

---

# 한글 FreeBSD 핸드북

---

본 문서는 재 배포가 가능하며, 상업적인 용도로 인쇄물 등 기타 다른 용도로 사용할 수 없습니다.

본 번역물의 저작권은 역자에게만 있습니다.

## FreeBSD Handbook

이 영 옥 <[rick@rickinc.com](mailto:rick@rickinc.com)> 역,  
FreeBSD City <<http://FreeBSDCity.org>> 후원,  
공개소프트웨어지원센터 <<http://oss.or.kr>> 협찬.

2004 년 8 월 7 일.

문서갱신번호 1.1

본 문서는 FreeBSD Documentation Project 에서 발행하는 FreeBSD Handbook 을 번역한 것으로 2004 년 8 월 7 일자 수정본을 기준으로 합니다. FreeBSD Handbook 은 지속적으로 갱신되기 때문에, 본 문서의 발간 이후에 갱신된 내용이 무엇인지 알고자 한다면, CVS Repository 를 조회하여 해당일 이후에 갱신된 내용만을 쉽게 찾아볼 수 있습니다. 또한 표지에 표시되는 날짜와 문서갱신번호는 이 문서의 갱신상태를 나타냅니다. 날짜는 번역의 대상이 되는 오리지날 핸드북의 최종 수정일자를 표시하며, 문서갱신번호는 문서에 수정이 있을 때 증가하므로 사용자가 이 문서의 갱신상태를 쉽게 파악할 수 있도록 합니다. 본 문서는 이영옥이 관리, 배포하므로 내용에 오타자 및 오역이 발견될 경우엔 역자에게 알려주어 수정될 수 있도록 해 주십시오.

“FreeBSD Handbook”의 저작권은 “FreeBSD Documentation Project”에 있으며, 번역물인 “한글 FreeBSD 핸드북”의 저작권은 이영옥(Young-oak Lee)에게 있습니다.

*Copyright © 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004 The FreeBSD Documentation Project.*

---

# FreeBSD 문서 프로젝트

Copyright © 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004 The FreeBSD Documentation Project

FreeBSD의 세계에 온 것을 환영한다! 이 책은 *FreeBSD 4.10-RELEASE*에서 *FreeBSD 5.2.1-RELEASE*의 설치와 일상적으로 사용하는 방법을 설명한다. 여러 사람들이 이 매뉴얼을 작성하고 있지만 더 많은 섹션이 필요하고, 여기있는 섹션 중 몇개는 업데이트가 필요하다. 이 프로젝트에 참여하고 싶다면 FreeBSD 문서 프로젝트 메일링 리스트 (<http://lists.freebsd.org/mailman/listinfo/freebsd-doc>)에 메일을 보내기 바란다. 이 책의 가장 최신 버전은 FreeBSD 웹 사이트에서(<http://www.freebsd.org/>) 항상 볼 수 있다. 그리고 FreeBSD FTP 서버(<ftp://ftp.freebsd.org/pub/FreeBSD/doc/>)와 수 많은 미러 사이트 ([http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/handbook/mirrors-ftp.html](http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/mirrors-ftp.html))에서 다양한 형식으로 압축된 버전을 다운로드할 수 있다.

## 서문

### 독자들에게

FreeBSD를 처음 접하는 사람들을위해 이 가이드의 첫 번째 섹션에서 FreeBSD를 설치하는 과정 그리고 유닉스의 개념과 전통을 소개하고 있다. 여러분들의 관심과 새로 소개되는 컴퓨터 개념을 이해하는 능력이 이 섹션 전반에 필요하다.

이 책을 읽다보면 핸드북이 FreeBSD 시스템 관리자가 관심있어하는 모든 주제를 다루는 포괄적인 레퍼런스임을 알게된다. 대부분의 장에서 소개하는 내용을 원활히 이해하기위해 선행해서 읽어야되는 내용을 미리 제시하고 있으므로 참고하도록 한다.

### 초판에서 변경된 내용

두 번째 판은 FreeBSD 문서 프로젝트 멤버들이 2년동안 노력한 결실이다. 다음과 같은 사항이 이번에 변경되었다.

- 
- 완벽한 인덱스를 추가했다.
  - 모든 ASCII 삽화를 그래픽으로 바꾸었다.
  - 각 장에서 소개하는 내용을 미리 요약한 개요를 추가했다.
  - 전체 구성을 연관있는 5 개의 파트로 재 구성하였다: “시작”, “일반적인 업무”, “시스템 관리”, “네트워크 통신”, “부록”.
  - 2 장은 (“FreeBSD 설치”) 초보자들이 이해하기 쉽게 스크린샷을 많이넣어서 새로 작성했다.
  - 3 장에는 (“유닉스 기초”) 프로세스와 데몬 그리고 신호에 대한 내용을 추가했다.
  - 4 장에는 (“어플리케이션 설치”) 바이너리 패키지 관리에 대한 내용을 추가했다.
  - 5 장도 (“X 윈도우 시스템”) XFree86 4.X 에 KDE 와 GNOME 같은 최신 데스크톱 기술을 사용하는 방법 위주로 새롭게 작성했다.
  - 10 장에는 (“리눅스 호환성”) Oracle®과 SAP® R/3® 설치에 대한 내용을 추가했다.
  - 12 장도 (“FreeBSD 부팅 프로세스”) 내용을 보완했다.
  - 16 장은 (“스토리지”) 기존의 “디스크”와 “백업”을 하나로 통합했다. 이 주제를 통합해서 설명하는것이 이해하기 쉽다고 생각된다. 그리고 RAID 에(하드웨어와 소프트웨어) 대한 내용도 추가했다.
  - 20 장은 (“시리얼 통신”) 재 구성했고 내용도 FreeBSD 4.X/5.X 로 업데이트했다.
  - 21 장은 (“PPP 와 SLIP”) 대체적으로 내용을 업데이트했다.
  - 22 장에는 (“전자 메일”) **센드메일** 설정에 대한 내용을 추가했다.
  - 24 장에는 (“발전된 네트워크”) 새로운 내용을 많이 추가했다.

- 
- 다음과 같은 두 개의 주제를 이번에 추가 했다:

- 설정과 튜닝 (11 장)
- 멀티미디어 (7 장)

## 이 책의 구성

이 책은 논리적인 3 개의 섹션으로 나뉜다. 첫 번째 섹션 *시작하기*에서는 FreeBSD 설치와 기본적인 사용법을 설명하였다. 따라서 독자들은 이 내용을 순서대로 읽거나 관심있는 주제만 바로 읽어도 된다. 두 번째 섹션 *일반적인 업무*에서는 보편적으로 많이 사용되고있는 FreeBSD 기능을 설명하고있다. 역시 이 섹션과 모든 서브섹션을 읽는 순서가 따로 있지는않다. 각 섹션의 시작에는 각 장에서 다루는 내용과 독자들이 미리 알고있어야되는 내용을 간단히 요약한 설명이있다. 세 번째 섹션 *시스템 관리*에서는 FreeBSD 시스템을 관리하는 방법에 대해 설명한다. 네 번째 섹션 *네트워크 통신*은 네트워크와 서버에 대한 설명이다. 다섯 번째 섹션에는 참고할만한 부록을 담았다.

### 1 장, 소개

새로운 유저들에게 FreeBSD 를 소개하고있다. FreeBSD 프로젝트의 역사, 목표와 개발 모델을 설명한다.

### 2 장, 설치

이번 장에서 유저는 설치 과정을 따라하기만 하면된다. 그리고 시리얼 콘솔을 사용하여 설치하는 내용과 특별한 경우에 설치하는 방법을 소개한다.

### 3 장, 유닉스 기본

기본적인 유닉스 명령과 FreeBSD 운영체제의 기능에 대해 설명한다. 리눅스나 다른 종류의 유닉스에 익숙하다면 이 장을 지나쳐도 된다.

---

#### 4 장, 어플리케이션 설치

어플리케이션 설치를 혁신적으로 개선한 FreeBSD 의 “포트 컬렉션”과 표준 바이너리 패키지로 소프트웨어를 설치하는 방법을 소개한다.

#### 5 장, X 윈도우 시스템

X 윈도우의 일반적인 사항과 FreeBSD 에서 **XFree86** 을 어떻게 사용하는지 보여준다. 그리고 **KDE** 와 **GNOME** 같은 공통 데스크톱 환경도 설명한다.

#### 6 장, 데스크톱 어플리케이션

웹 브라우저와 사무실에서 사용하는 오피스 같은 공통 데스크톱 어플리케이션 몇 가지를 소개하고 FreeBSD 에 설치하는 방법을 설명한다.

#### 7 장, 멀티미디어

시스템을 지원하는 사운드와 비디오 재생기를 어떻게 설치하는지 보여준다. 그리고 샘플 오디오와 비디오 어플리케이션 몇 가지를 소개한다.

#### 8 장, FreeBSD 커널 설정

새로운 커널을 설정해야되는 이유를 설명하고 사용자 커널 설정, 빌드 그리고 설치에 대한 자세한 정보를 제공한다.

#### 9 장, 프린트

배너 페이지, 프린트 비용 계산 그리고 초기 설정에 대한 내용과 FreeBSD 에서 프린터를 관리하는 방법을 설명한다.

---

## 10 장, 리눅스 바이너리 호환성

FreeBSD 의 리눅스 호환성 기능을 설명한다. 그리고 **Oracle, SAP R/3** 와 **Mathematica**®처럼 유명한 리눅스 어플리케이션 설치도 자세히 설명한다.

## 11 장, 설정과 튜닝

최적의 성능을 유지하기위해 FreeBSD 를 튜닝하는데 사용할 수 있는 매개 변수를 설명한다. 그리고 다양한 설정 파일과 이런 파일을 어디서 찾을 수 있는지 설명한다.

## 12 장, 부팅 과정

FreeBSD 의 부팅 과정을 보여주고 설정 옵션으로 이들 과정을 어떻게 제어하는지 설명한다.

## 13 장, 유저와 기본 계정 관리

유저 계정을 생성하고 관리하는 방법을 설명한다. 그리고 유저와 다른 계정에 적용할 수 있는 사용 제한을 설명한다.

## 14 장, 보안

Kerberos, IPsec, 오픈 SSH 그리고 네트워크 방화벽을 포함하여 FreeBSD 시스템의 보안을 지원하는 여러가지 툴을 설명한다.

## 15 장, 필수 접근제어

필수 접근제어 (MAC)는 무엇이고 FreeBSD 시스템 보안에 이 메커니즘을 어떻게 사용하는지 설명한다.

---

## 16 장, 스토리지

FreeBSD 로 스토리지 미디어와 파일시스템을 어떻게 관리하는지 설명한다. 여기서는 물리적인 디스크, 레이드 어레이, 테잎 및 광학 미디어, 메모리 기반 디스크 그리고 네트워크 파일시스템이 포함된다.

## 17 장, Vinum 볼륨 매니저

장치에 독립적인 논리 디스크와 소프트웨어 RAID-0, RAID-1 과 RAID-5 를 제공하는 볼륨 매니저 Vinum 을 어떻게 사용하는지 설명한다.

## 18 장, 지역화

FreeBSD 에서 영어가 아닌 다른 언어를 어떻게 사용하는지 설명한다. 시스템과 어플리케이션 레벨에서 지역화하는 방법을 다룬다.

## 19 장, 최신 업데이트

FreeBSD-STABLE, FreeBSD-CURRENT 와 FreeBSD 릴리즈의 차이점을 설명한다. 또한 개발 시스템이 어떤 유저에게 이익이 되는지 그리고 관련된 과정을 소개한다.

## 20 장, 시리얼 통신

전화 연결(Dial-in) 서비스와 전화 접속(Dial-out)을위해 FreeBSD 시스템에 터미널과 모뎀을 어떻게 연결하는지 설명한다.

## 21 장, PPP 와 SLIP

PPP, SLIP 또는 PPP 를통한 이더넷을 사용하여 원격 시스템에 어떻게 접속하는지 설명한다.

---

## 22 장, 전자 메일

메일서버의 다양한 컴포넌트를 소개하고 가장 유명한 메일 서버 소프트웨어인 **센드 메일** 설정에 대해 간략히 설명한다.

## 23 장, 네트워크 서버

FreeBSD 머신을 네트워크 파일시스템 서버, 도메인 네임 서버, 네트워크 정보 시스템 서버 또는 시간 동기 서버로 설정하는 방법을 자세히 설명하고 설정 파일과 예제를 제공한다.

## 24 장, 발전된 네트워크

LAN 에있는 컴퓨터로 인터넷을 공유하고 발전된 라우팅에 관한 주제, 무선 네트워크, 블루투스, ATM, IPv6 와 다양한 내용을 포함한 네트워크 설정을 다룬다.

## 부록 A, FreeBSD 받기

FreeBSD CDROM, DVD 와 다운로드하여 설치할 수 있는 인터넷 사이트의 여러 자료들을 소개한다.

## 부록 B, 관련 서적소개

여기서는 더 자세한 설명을 원하는 유저를위해 여러가지 관련 도서를 소개한다.

## 부록 C, 인터넷의 자료이용

FreeBSD 유저들이 질문을 올리고 기술적으로 토론할 수 있는 다양한 포럼을 소개한다.



---

## 이 책에서 약정

일관되고 읽기 쉬운 텍스트를 제공하기 위해 이 책에서는 다음과 같은 약정을 사용한다.

## 인쇄상의 약정

### *Italic*

*italic* 체는 파일 이름, URL, 강조된 텍스트 그리고 기술적인 용어를 처음 소개할 때 사용했다.

### 바탕체

바탕체는 에러 메시지, 명령, 환경 변수, 포트 이름, 호스트이름, 유저 이름, 그룹 이름, 장치 이름, 변수와 코드에 사용했다.

### **Bold**

**bold** 체는 어플리케이션 명령과 키에 사용했다.

## 유저 입력

**bold** 체로 보여주는 키는 다른 텍스트의 표준 출력이다. 다음과 같이 키 사이에 '+'로 보여주는 것은 동시에 입력하라는 의미의 키 조합이다:

### **Ctrl + Alt + Del**

이 의미는 유저가 **Ctrl**, **Alt** 와 **Del** 키를 동시에 누르라는 것이다.

콤마(,)로 분리된 키를 순서대로 입력하라는 의미의 예제는 아래와 같다.

---

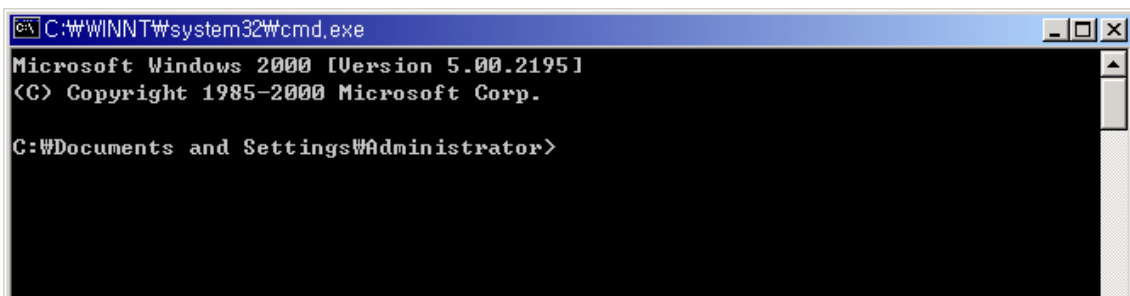
Ctrl+X, Ctrl+S

유저가 **Ctrl** 과 **X** 키를 동시에 누른 후 **Ctrl** 과 **S** 키를 동시에 누르는 것을 의미한다.

## 예제

E:W>로 시작하는 예제는 MS-DOS 명령이다. 특별히 명시하지 않으면 이런 명령은 Microsoft 윈도우 환경의 “명령 프롬프트” 윈도우에서 실행한다.

E:W> tools\Wfdimage floppies\Wkern.flp A:



```
C:\WINNT\system32\cmd.exe
Microsoft Windows 2000 [Version 5.00.2195]
<C> Copyright 1985-2000 Microsoft Corp.

C:\Documents and Settings\Administrator>
```

#로 시작하는 명령의 예제는 FreeBSD 에서 슈퍼 유저로 실행해야 된다. root 로 로그인 해서 명령을 입력하거나 일반 유저로 로그인해서 su(1)를 사용하여 슈퍼 유저 권한을 얻는다.

# dd if=kern.flp of=/dev/fd0

%로 시작하는 명령의 예제는 일반 유저 계정으로 실행한다. 특별히 명시하지 않으면 C 셸 구문이 환경 변수 설정과 다른 셸 명령에 사용된다.

% top

## 감사의 글

여러분이 보고있는 이 책은 전 세계 수많은 사람들이 노력한 결과다. 이들이 철자를 수정했

---

거나 완전한 챕터를 제출했던지 글을 제출한 사람들에게 감사하자!

몇몇 회사들이 이 자료를 개선하기위해 일하는 작가나 출판사 등에 비용을 지불해줬다. 특히 BSDi 는(Wind River System 이 인수한) 2000 년 3 월에 최초로 출판된 출판물(ISBN 1-57176-241-8) 간행을위해 일하는 FreeBSD 문서화 프로젝트 멤버들에게 비용을 지불해줬다. 그리고 Wind River Systems 는 챕터를 추가하는 작가들에게도 비용을 지불했다. 이 작업은 2001 년 11 월에 두 번째 출판물로(ISBN 1-57176-303-1) 종료되었다.

---

# 1. 시작 하기

FreeBSD 핸드북의 이 부분은 FreeBSD 가 처음인 유저와 관리자를위한 곳이다. 이 챕터에서는 다음과 같은 내용을 소개한다:

- FreeBSD 를 소개한다.
- FreeBSD 의 설치 과정을 설명한다.
- 유닉스의 기본과 원칙을 소개한다.
- 풍부한 어플리케이션을 FreeBSD 에 어떻게 설치하는지 보여준다.
- 유닉스 윈도우 시스템 X 를 소개하고 데스크톱 환경을 업무적으로 사용할 수 있는 설정도 자세히 설명한다.

## 1 장 FreeBSD 소개

### 1.1 개요

FreeBSD 에 대한 여러분들의 관심에 감사한다! 이번 장에서는 FreeBSD 프로젝트의 역사, 목표, 개발 모델 등과 같은 프로젝트의 다양한 활동을 소개한다.

이번 장을 읽고 다음과 같은 사항을 알 수 있다:

- FreeBSD 와 다른 컴퓨터 운영체제의 차이점.
- FreeBSD 프로젝트의 역사.
- FreeBSD 프로젝트의 목표.
- FreeBSD 의 오픈 소스 개발 모델.

- 
- 마지막으로 "FreeBSD"라는 이름의 유래.

## 1.2 FreeBSD 의 세계로 온 것을 환영한다.

FreeBSD는 인텔(x86 과 Itanium), AMD64, Alpha, Sun UltraSPARC 컴퓨터에서 운용되는 4.BSD-Lite 기반 운영체제이며 다른 아키텍처로 이식도 진행 중이다. 이 장에서는 FreeBSD의 역사나 현재 릴리즈에 관한 글도 읽을 수 있다. 프로젝트에 도움을 주고 싶다면 (코드, 하드웨어, 금전적으로) FreeBSD에 공헌하기라는 글을 살펴보자 ([http://www.freebsd.org/doc/en\\_US.ISO8859-1/articles/contributing/index.html](http://www.freebsd.org/doc/en_US.ISO8859-1/articles/contributing/index.html)).

### 1.2.1 FreeBSD 는 무엇을 할 수 있는가?

FreeBSD 는 주목할만한 여러가지 특징을 가지고있다. 어떤 특징이 있는지 살펴보자!

- 부하가 많이 걸린 상황에서도 어플리케이션과 유저가 공정하게 컴퓨터를 공유할 수 있도록 동적으로 우선순위를 조정하여 *우선순위 멀티태스킹*을 제공한다.
- 많은 사람들이 다양한 일에 FreeBSD 시스템을 동시에 사용할 수 있는 *멀티유저* 기능을 제공한다. 이 의미는 프린터, 테잎 장치와 같은 시스템 주변장치를 시스템과 네트워크에있는 모든 유저들이 공평하게 공유하고, 중요한 시스템 자원의 과중한 사용을 방지하기위해 유저 또는 그룹단위로 사용하는 자원을 제한할 수 있다.
- SLIP 와 PPP(21 장 참고), NFS 와 DHCP 그리고 NIS(23 장 참고)같은 산업 표준을 지원하는 강력한 *TCP/IP 네트워킹*을 제공한다. 이 의미는 FreeBSD 머신이 엔터프라이즈 서버로 동작하는 다른 시스템과 쉽게 연계하여 NFS, 메일 서비스처럼 중요한 기능을 제공한다. 그리고 인터넷 서비스, FTP, 방화벽(14 장 참고) 서비스를 사용하여 여러분의 그룹이 인터넷에 접속할 수 있다.
- 어플리케이션(또는 유저)이 다른 어플리케이션과 간섭하지 않도록 *메모리 보호*기능을 제공한다. 한쪽 어플리케이션의 문제가 어떤 경우든지 다른 어플리케이션에 영향을 주지않는다.
- FreeBSD 는 *32 비트* 운영체제(Alpha, Itanium, AMD64 와 UltraSPARC 에서는 *64 비트*

---

트 운영체제다)고 이런 배경에서 디자인되었다.

- 저렴한 VGA 카드와 모니터에 산업표준인 X 윈도우 시스템(5 장 참고)을 사용하여 그래픽 유저 인터페이스(GUI)를 제공한다.
- 리눅스, SCO, SVR4, BSDI 와 NetBSD 용으로 만들어진 다양한 프로그램과 바이너리 호환성을 제공한다. 다시 말해서 다른 유닉스 운영체제용으로 만들어진 프로그램을 그대로 사용할 수 있다.
- 바로 사용할 수 있는 수천 개의 어플리케이션을 FreeBSD 포트와 패키지 컬렉션(4 장에서 설명한다)으로 설치할 수 있다. 원하는 어플리케이션을 인터넷에서 찾을 필요없이 이곳에서 찾을 수 있다.
- 이식하기 쉬운 수천 개의 부가적인 어플리케이션을 인터넷에서 설치할 수 있다. FreeBSD 는 아주 유명한 상용 유닉스 시스템의 소스코드와 호환되기 때문에 대부분의 어플리케이션을 컴파일할 때 약간의 수정이 필요하거나 그대로 컴파일할 수 있다.
- Demand paged 가상 메모리와 병합된 가상 메모리/버퍼 캐시 디자인이 메모리를 많이 사용하는 어플리케이션을 효과적으로 지원하며 다른 유저에 대해서도 응답을 유지한다.
- SMP는 멀티 CPU 머신을 지원한다.
- C, C++, Fortran 그리고 Perl 처럼 서로 보완되는 다양한 개발 툴을 제공한다. 고급 연구와 개발에 사용하는 다른 언어도 포트와 패키지 컬렉션에서 설치할 수 있다.
- 전체 시스템 소스 코드가 제공되기 때문에 원하는 환경을 구축할 수 있다. 특정 벤더 솔루션에 의존하지않고 진정한 오픈 시스템을 사용할 수 있다.
- 광범위한 온라인 문서를 제공한다.
- 그리고 더 많은 특징들이 있다!

FreeBSD 는 캘리포니아 버클리대학의 컴퓨터 시스템 연구 그룹의(CSRG) 4.4BSD-Lite 로부터 BSD 시스템 개발의 뛰어난 전통을 지켜오고 있다. 게다가 FreeBSD 프로젝트는 CSRG

---

의 지원으로 최고 성능과 실제 부하에서 신뢰성을 유지하기 위해 수천 시간동안 시스템을 튜닝하고있다. 많은 거대 기업들이 성능과 신뢰성을 바탕으로 PC 운영체제 영역으로 진출하려고 하지만 FreeBSD 는 이러한 특징을 지금 제공하고 있다!

그리고 FreeBSD 에서 사용할 수 있는 어플리케이션이 상당히 제한적이라는것은 오직 여러분의 생각일 뿐이다. 소프트웨어 개발에서 공장 자동화, 원격 위성 안테나의 방위각을 교정하는 제품의 재고 관리까지 상업용 유닉스 제품으로 처리할 수 있다면 FreeBSD 도할 수 있다. 게다가 FreeBSD 를 사용하여 약간의 비용이나 비용이 전혀없이 리서치 센터와 전 세계 대학에서 개발한 수천 개의 고 가용성 어플리케이션을 으로 상당한 이익을 얻을 수 있다. 물론 매일 개발되는 수많은 상업용 어플리케이션도 사용할 수 있다.

FreeBSD 는 소스 코드도 자유롭게 이용할 수 있기 때문에 거의 알려지지않은 특별한 어플리케이션이나 프로젝트용으로 시스템을 변경할 수 있다. 이러한 방법으로 주요 상업용 벤더의 운영체제로 불가능한 것도 FreeBSD 에서는 가능하다. 여기서는 사람들이 현재 FreeBSD 에 사용하고있는 어플리케이션의 몇 가지 모델을 보여준다:

- *인터넷 서비스*: FreeBSD 에 내장된 강력한 TCP/IP 네트워킹으로 다음과 같은 다양한 인터넷 서비스에 이상적인 플랫폼이됐다:
  - FTP 서버
  - 웹 서버 (표준 또는 보안[SSL])
  - 방화벽과 NAT("IP 마스커레이딩") 게이트웨이
  - 전자 메일 서버
  - 유즈넷 뉴스 또는 전자 게시판 시스템
  - 더 많은 시스템에 사용되고 있다.

FreeBSD 를 사용하여 저렴한 386 급의 PC 로 소규모로 시작하여 사업의 성장에 맞추어 RAID 스토리지가 있는 4 개의 Xeon 프로세스로 업그레이드할 수 있다.

- *교육*: 여러분은 컴퓨터 공학이나 엔지니어링 필드와 연관된 학생인가? FreeBSD 는 운영체제와 컴퓨터 아키텍처, 네트워크를 공부하는 가장 좋은 환경을 제공한다. 무료로 사용할 수 있는 CAD 와 수학 그리고 그래픽 디자인 패키지도 주로 컴퓨터에

---

관심있는 사람들이 다른 작업을 할 때 유용하다.

- *연구*: 전체 시스템 소스 코드를 사용할 수 있기 때문에 FreeBSD는 컴퓨터 공학의 다른 분야처럼 운영체제 연구용으로 최고의 플랫폼이다. 자유롭게 이용할 수 있는 FreeBSD의 특징으로, 특별한 라이선스 계약이나 제한에 대한 걱정없이 원격그룹의 아이디어나 공유된 개발에 협력할 수 있다.
- *네트워킹*: 새로운 라우터, 네임서버(DNS) 또는 내부 네트워크를 지키기 위한 방화벽이 필요한가? 사용하지 않고 구석에 쌓아둔 386이나 486 PC에 FreeBSD를 설치하여 정교한 패킷 필터링이 가능한 고급 라우터로 쉽게 바꿀 수 있다.
- *X 윈도우 워크스테이션*: 무료 XFree86 서버나 훌륭한 상업용 서버 중 Xi Graphics가 제공하는 저렴한 X 터미널 솔루션으로 FreeBSD는 탁월한 선택이다. X 터미널과 달리, 원한다면 FreeBSD는 많은 어플리케이션을 로컬 머신에서 실행할 수 있기 때문에 중앙 서버의 부하를 줄일 수 있다. 또한 FreeBSD는 디스크 없이 부팅할 수 있기 때문에 여러대의 워크스테이션을 저렴하고 쉽게 관리할 수 있다.
- *소프트웨어 개발*: 기본적으로 FreeBSD 시스템은 유명한 GNU C/C++ 컴파일러와 디버거를 포함한 완벽한 개발 툴을 가지고있다.

## 1.2.2 누가 FreeBSD를 사용하는가?

FreeBSD는 다음 사이트들을 포함하여 인터넷의 거대한 사이트에 힘이되고있다:

- Yahoo! (<http://www.yahoo.com/>)
- Apache (<http://www.apache.org/>)
- Boue Mountain Arts (<http://www.bluemountain.com/>)
- Pair Networks (<http://www.pair.com/>)
- Sony Japan (<http://www.sony.co.jp/>)



- Netcraft (<http://www.netcraft.com/>)
- Weathernews (<http://www.supervalu.com/>)
- TELEHOUSE America (<http://www.telehouse.com/>)
- Sophos Anti-Virus (<http://www.sophos.com/>)
- JMA Wired (<http://www.jmawired.com/>)
- 그리고 국내에서는 드림위즈 (<http://www.dreamwiz.com>)와 세이클럽 (<http://www.sayclub.com>) 등 많은 사이트에 사용되고 있다.

### 드림위즈

OS, Web Server and Hosting History for dreamwiz.com					
http://dreamwiz.com was running Apache on FreeBSD when last queried at 28-Aug-2004 07:03:38 GMT - refresh now					FAQ
OS	Server	Last changed	IP address	Netblock Owner	
FreeBSD	Apache/1.3.28 (Unix)	14-Jul-2004	211.39.128.129	KRNIC	
FreeBSD	Apache/1.3.28 (Unix)	8-May-2004	211.39.128.139	KRNIC	
FreeBSD	Apache/1.3.28 (Unix)	18-Aug-2003	211.39.128.139	CENTRAL DATA COMMUNICATION OFFICE	
FreeBSD	Apache/1.3.26 (Unix)	28-Sep-2002	211.39.128.139	CENTRAL DATA COMMUNICATION OFFICE	
FreeBSD	Apache/1.3.26 (Unix)	9-Jul-2002	211.174.54.139	KOREAINTERNETDATACENTERInc.	
FreeBSD	Apache/1.3.12 (Unix)	31-Dec-2000	211.174.54.139	KRNIC	
FreeBSD	unknown	30-Dec-2000	211.174.54.139	KRNIC	
FreeBSD	Apache/1.3.12 (Unix)	18-Nov-2000	211.174.54.139	KRNIC	
FreeBSD	Apache/1.3.12 (Unix)	4-Nov-2000	211.62.252.142	DACOM	

### 세이클럽

OS, Web Server and Hosting History for sayclub.com						
http://sayclub.com was running Apache on FreeBSD when last queried at 28-Aug-2004 10:12:11 GMT - refresh now						FAQ
OS	Server	Last changed	IP address	Netblock Owner		
FreeBSD	Apache/1.3.28 (Unix) mod_ssl/2.8.15 OpenSSL/0.9.7c	28-Aug-2004	211.234.121.76	NeoWiz Corp.		
FreeBSD	Apache/1.3.31 (Unix) mod_ssl/2.8.19 OpenSSL/0.9.7c	21-Aug-2004	211.234.121.76	NeoWiz Corp.		
FreeBSD	Apache/1.3.28 (Unix) mod_ssl/2.8.15 OpenSSL/0.9.7c	19-Aug-2004	211.234.121.77	NeoWiz Corp.		
FreeBSD	Apache/1.3.29 (Unix) mod_ssl/2.8.16 OpenSSL/0.9.7c	18-Aug-2004	211.234.121.85	NeoWiz Corp.		
FreeBSD	Apache/1.3.28 (Unix) mod_ssl/2.8.15 OpenSSL/0.9.7c	16-Aug-2004	211.234.121.77	NeoWiz Corp.		
FreeBSD	Apache/1.3.29 (Unix) mod_ssl/2.8.16 OpenSSL/0.9.7c	13-Aug-2004	211.234.121.77	NeoWiz Corp.		
FreeBSD	Apache/1.3.28 (Unix) mod_ssl/2.8.15 OpenSSL/0.9.7c	11-Aug-2004	211.234.121.77	NeoWiz Corp.		
FreeBSD	Apache/1.3.29 (Unix) mod_ssl/2.8.16 OpenSSL/0.9.7c	9-Aug-2004	211.234.121.84	NeoWiz Corp.		
FreeBSD	Apache/1.3.28 (Unix) mod_ssl/2.8.15 OpenSSL/0.9.7c	7-Aug-2004	211.234.121.76	NeoWiz Corp.		
FreeBSD	Apache/1.3.29 (Unix) mod_ssl/2.8.16 OpenSSL/0.9.7c	4-Aug-2004	211.234.121.77	NeoWiz Corp.		

## 1.3 FreeBSD 프로젝트에 관하여

다음 섹션은 FreeBSD 프로젝트의 간략한 역사, 목표와 개발 모델을 포함하여 프로젝트에

---

대한 정보를 제공한다.

### 1.3.1 간략한 FreeBSD 의 역사

FreeBSD 프로젝트는 마지막 3 명의 패치킷 진행자(필자, Nate Williams 와 Rod Grimes)들의 “비공식 386BSD 패치킷”에서 파생되어 이른 1993 년에 시작되었다.

우리의 최초 목적은 386BSD 의 수많은 문제를 패치킷 메커니즘이 해결하지 못했기 때문에 386BSD 의 중간 스냅샷을 만드는 것이었다. 독자 중 몇명은 예전 프로젝트 제목 "386BSD 0.5" 또는 "386BSD Interim"을 기억할 것이다. 386BSD 는 거의 1 년 동안의 무관심으로 더 심각해진 Bill Jolitz 의 운영체제였다. 게다가 패치킷이 매일 증가되었기 때문에 우리는 새로운 시도를 해야된다고 동의했고 이 중간 단계를 정리하는 스냅샷을 제공해서 Bill 을 지원하기로 결정했다. 이 계획은 프로젝트가 무엇을 지원해야될지 확실히 결정하지않고 Bill Jolitz 가 갑자기 협의를 철회했을 때 돌연이 끝났다.

이 일을 계기로 우리는 Bill 의 지원없이도 목표를위해 노력할 수 있다고 판단해서 David Greenam 이 만든 "FreeBSD"라는 이름을 채택하였다. 그 당시 우리는 시스템 유저들의 컨설팅을 받은 후 최초 목표를 설정하였다. 프로젝트의 목표가 현실적인 방향으로 확실해진 후 나는 인터넷을 사용하기가 어려운 사람들을위해 FreeBSD 의 배포 채널을 개선할 방법으로 Walnut Creek CDROM 과 연락하였다. Walnut Creek CDROM 은 FreeBSD 를 CD 로 배포하겠다는 아이디어를 지원하고 프로젝트에서 사용할 머신과 빠른 인터넷 회선을 제공하였다. 그 당시 전혀 알려지지않은 프로젝트에 대한 Walnut Creek CDROM 의 전혀 예상하지 못한 신뢰가 FreeBSD 를 빠르게 발전시켰다.

첫 번째 CDROM 배포판(그리고 일반 네트워크 배포판)은 1993 년 12 월의 FreeBSD 1.0 릴리즈였다. 이 배포판은 386BSD 와 자유 소프트웨어 협회에서 제공하는 수많은 컴포넌트와 U.C Berkeley(버클리대학)의 4.3BSD-Lite("Net/2") 테잎에 기반하였다. 최초의 배포판이 상당히 성공하여 우리는 1994 년 5 월에 FreeBSD 1.1 을 아주 성공적으로 배포하였다.

이 시기에 Novell(우리에게는 넷웨어로 더 친숙하다)과 U.C.Berkeley 사이에 제기된 Berkeley Net/2 테잎에 대한 장기간의 소송으로 전혀 예상치 못한 폭풍 구름이 형성되었다. Net/2 의 주요 코드가 이전 AT&T 로부터 Novell 이 구입한 코드를 사용하였기 때문에 판정은 U.C.Berkely 의 패소로 확정되었다. Berkeley 가 해결해야되는 과제는 Novell 이 지적한 4.4BSD-Lite 릴리즈였기 때문에 모든 Net/2 유저는 신고되지않은 코드를 변경하게 되었다.

여기에 FreeBSD 도 포함되었기 때문에 프로젝트는 1994 년 7 월 마지막 날까지 Net/2 기반

---

제품의 적용을 중단해야 되었다. 이 협정 기간동안 프로젝트는 시한 이전에 마지막 배포판인 FreeBSD 1.1.5.1 을 발표할 수 있었다. 그리고 FreeBSD 는 사실상 불완전한 4.4BSD 의 코드를 완전히 새로 구성하는 힘든 작업을 시작하였다. Berkeley 의 CSRG 는 시스템을 실제로 부팅하기 위해 필요한 대부분의 코드를 삭제했기 때문에(법률상 필요로) 4.4 인텔 포트는 아주 불완전해서 "Lite" 릴리즈는 가벼웠다.

이렇게 되어 프로젝트는 네트워크와 CDROM 으로(늦은 12 월에) FreeBSD 2.0 을 배포하게 된 1994 년 11 월까지 코드를 변경했다. 완벽하게 마무리되지는 않았지만 이 릴리즈는 상당히 성공적이어서 1995 년 6 월에 더욱 강력하고 설치하기 쉬운 FreeBSD 2.0.5 를 배포하는 계기가 되었다.

우리는 1996 년 8 월에 FreeBSD 2.1.5 를 배포하였고 이 버전은 ISP 와 상용 커뮤니티 사이에서 2.1-STABLE 분기가 안정될 때까지 상당한 인기를 얻었다. FreeBSD 2.1.7.1 이 1997 년 2 월에 배포되어 2.1-STABLE 의 주요 개발이 끝났다. 이제 유지 모드로 이 분기의 보안 향상과 중요한 버그 패치도 끝났다.

FreeBSD 2.2 는 주요 개발라인에서("-CURRENT") 1996 년 11 월에 RELENG\_2\_2 로 분기되었고, 첫 번째 전체 릴리즈는(2.2.1) 1997 년 4 월에 배포되었다. 2.2 분기 이 후의 릴리즈는 97 년의 여름과 가을 사이에 끝났고 마지막은(2.2.8) 1998 년 11 월에 시작되었다. 첫 번째 공식 3.0 릴리즈가 1998 년 10 월에 시작되었기 때문에 2.2 분기가 끝났다고 말할 수 있었다.

이 버전은 1999 년 1 월 20 일에 4.0-CURRENT 와 3.X-STABLE 분기로 나누어졌다. 3.X-STABLE 부터 3.1 은 1999 년 2 월 15 일에, 3.2 는 1999 년 5 월 15 일에, 3.3 은 1999 년 9 월 16 일에, 3.4 는 1999 년 12 월 20 일에 배포되었고 마지막으로 보안 패치가된 **kerberos** 를 적용하기 위해 며칠 후 3.5.1 이 된 3.5 는 2000 년 6 월 24 일에 배포되었다.

**용어 설명:** Kerberos 는 사용자가 보안 서버의 서비스를 통해 직접 인증을 받을 수 있도록 네트워크에 추가된 시스템/프로토콜이다. 원격 로그인, 원격 복사 그리고 시스템과 관련된 파일을 안전하게 복사하고 난 위도가 높은 태스크를 상당히 안전하고 더 세심히 제어할 수 있다. 자세한 설명은 14.6 장을 읽는다.

4.X-STABLE 분기가 출현하게된 2000 년 3 월 13 일에 다른 분기가 시작되어 이제 "current-stable 분기"로 여겨졌다. 이때부터 몇 개의 릴리즈가 배포되었다. 4.0-RELEASE 는 2000 년 3 월에 그리고 가장 최근의 4.9-RELEASE 는 2003 년 10 월에 배포되었다. 2003 년에는 4.X-stable(RELENG\_4) 분기를따라 더 많은 분기가 있었다.

---

오랫동안 기다려온 5.0-RELEASE 는 2003 년 1 월 19 일에 발표되었다. 거의 3 년 동안의 작업 결과인 이 릴리즈부터 FreeBSD 는 발전된 멀티프로세서와 어플리케이션 스레드를 지원하고 UltraSPARC®과 ia64 플랫폼을 지원하게 되었다. 이 릴리즈 이 후로 2003 년 6 월에 5.1 이 나왔다. 다양하고 새로운 기능으로 5.X 릴리즈도 주요 시스템 아키텍처의 개발 노선 중 하나가 되었다. 그러나 이러한 발전으로 제대로 테스트되지않은 많은 양의 새로운 코드가 포함되었다. 이런 이유로 5.X 릴리즈는 “신 기술” 릴리즈라고 하지만 4.X 시리즈는 “제품” 릴리즈라고 한다. 조만간 5.X 는 stable 로 선언되고 다음 개발 분기 6.0-CURRENT 가 시작된다.

장기간의 개발 프로젝트는 당분간 5.X-CURRENT 분기를 유지하고 CDROM 으로된(물론 인터넷으로도) 5.X 의 스냅샷 릴리즈도 작업이 진행되는동안 스냅샷 서버에서 (<ftp://current.freebsd.org/pub/FreeBSD/snapshots/>) 이용할 수 있다.

### 1.3.2 FreeBSD 프로젝트의 목표

FreeBSD 프로젝트의 목표는 추가 조항없이 어떤 목적에든 사용할 수 있는 소프트웨어를 제공하는 것이다. 우리 모두는 코드에(그리고 프로젝트에) 상당한 노력을 기울였기 때문에 약간의 재정적인 보수를 바라는것은 확실히 아니지만 아주 부인하는것도 아니다. 우리에게 가장 중요한 사명은 모든 이들에게 그리고 어떤 목적에든 코드를 제공하는 것이기 때문에 코드는 가능한 폭넓게 사용되어 최대 이익을 제공해야된다. 이것이 자유소프트웨어의 가장 중요한 목표이고 우리가 열광적으로 지원하는 이유다.

GNU General Public License(GPL) 또는 Library General Public License(LGPL)를 가지고 있는 우리 소스 트리의 코드는 일반적인 반대 세력보다 사용하기 쉽겠지만 약간의 조항이 더 붙어 있다. GPL 소프트웨어를 상업적인 용도로 사용할 수 있지만 약간 더 복잡해지기 때문에 적절한 옵션이 필요하다면 소프트웨어를 더욱 자유로운 BSD 라이선스로 배포한다.

### 1.3.3 FreeBSD 개발 모델

사실상 우리의 공헌자 리스트에서 볼 수 있듯이

([http://www.freebsd.org/doc/en\\_US.ISO8859-1/articles/contributors/article.html](http://www.freebsd.org/doc/en_US.ISO8859-1/articles/contributors/article.html))

FreeBSD 개발은 전 세계 수백 명의 공헌자들에게 오픈되어 유연성있게 진행되고 있다. 우리는 계속해서 새로운 개발자와 아이디어를 찾고있으므로 관심있는 사람은 FreeBSD 기술 토론 메일링 리스트에(<http://lists.FreeBSD.org/mailman/listinfo/freebsd-hackers>) 등록해서

---

프로젝트와 더 가까워질 수 있다. FreeBSD 공지 메일링 리스트도

(<http://lists.FreeBSD.org/mailman/listinfo/freebsd-announce>) 프로젝트의 주된 사항을 다른 FreeBSD 유저에게 알리는데 사용할 수 있다. 메일링 리스트의 종류와 등록에 대한 자세한 정보는 [부록 C.1]을 참고한다.

혼자서 일을 하거나 같이 협력하던지 FreeBSD 프로젝트와 개발 진행에 대한 유용한 사항은 다음과 같은 곳에서 알 수 있다:

#### CVS 저장소

FreeBSD의 중앙 소스 트리는 FreeBSD에서 무료로 사용할 수 있는 소스 코드 제어 툴인 CVS로(<http://www.cvshome.org/>)(버전 관리 시스템) 관리된다. 주 CVS 저장소는(<http://www.FreeBSD.org/cgi/cvsweb.cgi>) 미국 Santa Clara CA의 머신에 있고 복제된 많은 미러 머신이 전세계에 걸쳐있다. CURRENT와 STABLE 트리를 가지고 있는 CVS 트리는 여러분의 머신에도 쉽게 복제할 수 있다. 복제하는 방법에 대한 더 많은 자료는 [소스 트리 동기화]섹션을 본다.

#### 커미터 리스트

커미터는 CVS 트리에 접근하여 FreeBSD 소스를 수정할 수 있는 권한을 가진 사람이다(용어 “committer”은 새롭게 변경한 내용을 CVS 저장소에 보관하는데 사용하는 cvs(1) commit 명령에서 왔다). 소스에 대한 재 검토를 요청하는 가장 좋은 방법은 커미터 리스트에 send-pr(1) 명령을 사용하는 것이다. 시스템에 문제가 발생했다면 FreeBSD 커미터 메일링 리스트에 메일을 보내서 해결할 수 있다.

#### FreeBSD 코어 팀

FreeBSD 프로젝트가 회사라면 FreeBSD 코어 팀은 위원회와 같다. 코어 팀의 최우선 업무는 프로젝트가 안정된 상태로 유지되도록 바른길로 인도하는 일이다. 새로운 코어 팀원을 채용하는 것만큼 헌신적이고 책임감있는 개발자를 우리 커미터 그룹에 영입하는것도 코어 팀의 임무 중 하나다. 현재 코어 팀은 2002년에 커미터 후보자 중에서 선출됐다. 선거는 매 2년마다 있다.

---

어떤 코어 팀원은 특정 영역의 의무도 가지고 있다. 이 말은 이들이 시스템 작업의 중요한 역할에 헌신하고 있다는 것이다. FreeBSD 개발자와 이들의 역할에 대한 전체적인 리스트는 공헌자 리스트를([http://www.freebsd.org/doc/en\\_US.ISO8859-1/articles/contributors/article.html](http://www.freebsd.org/doc/en_US.ISO8859-1/articles/contributors/article.html)) 본다.

**Note:** 대부분의 코어 팀원은 FreeBSD 개발과 프로젝트에서 금전적인 대가를 받지않는 봉사자이기 때문에 “헌신”을 “대가가 보장”된 것으로 오해하지 말아야 된다. 위의 “위원회”라는 말은 정확하지않고 이들은 FreeBSD 를위해 자신들의 생활을 포기한 사람이라는 표현이 더 어울릴 것이다.

#### 외부 공헌 자

가장 큰 개발자 그룹은 정보를 제공하고 버그를 수정해서 우리에게 보내주는 유저들 자신이다. FreeBSD 의 다른 개발에 지속적으로 관심을가지기 위해 이러한 주제가 토론되는 FreeBSD 기술 토론 메일링 리스트에 가입한다. FreeBSD 메일링 리스트에 대한 더 많은 정보는 [부록 C]를본다.

FreeBSD 공헌자 리스트는 상당히 길고 꾸준히 늘고있다. 망설이지 말고 참여하기 바란다.

코드를 제공하는것만이 프로젝트에 공헌하는 유일한 방법은 아니다. 필요한것에 대한 전체적인 리스트는 FreeBSD 프로젝트 웹사이트를 참고한다.

다시 요약하면 우리의 개발 모델은 중심화된 여유있는 집합으로 조직되어있다. 중심화된 모델은 공헌자가 아닌 FreeBSD 유저들이 하나의 메인 코드를 유지하기 쉽도록 편리하게 디자인 되어있다. 우리의 바램은 유저들이 쉽게 설치해서 사용할 수 있는 일관된 어플리케이션 프로그램으로 안정된 운영체제를 유지하는 것이다. 이 개발 모델이 우리의 목적을 이루는데 적합하다고 생각한다.

FreeBSD 개발자로 참여하려는 사람들에게 우리가 요구하는 사항은 사람들이 현재 참여하고있는 정도를 원할 뿐이다.

---

### 1.3.4 현재 FreeBSD 릴리즈

현재 FreeBSD 는 Intel i386™, i486™, Pentium®, Pentium Pro, Celeron®, Pentium II, Pentium III, Pentium 4 (또는 호환), Xeon™, DEC Alpha™ 그리고 Sun UltraSPARC 기반 컴퓨터 시스템에 자유롭게 사용할 수 있는 4.BSD-Lite 기반의 완전한 소스다. FreeBSD 는 NetBSD, OpenBSD, 386BSD, 자유 소프트웨어 협회의 소프트웨어와 U.C.Berkeley CSRG 그룹의 소프트웨어에 바탕을 두고있다.

늦은 94년 2.0 릴리즈부터 FreeBSD 의 성능, 기능과 안정성이 급격히 향상되었다. 가장 큰 변화는 병합된 가상메모리/파일 버퍼 케시로 가상 메모리 시스템의 혁신을 이룬것이다. 성능 향상뿐 아니라 메모리 접근을 쉽게할 수 있도록 5MB 로 설정하여 FreeBSD 의 메모리 누수도 줄였다. 또 다른 발전은 완벽한 NIS 클라이언트와 서버 지원, TCP 트랜잭션 지원, dial-on-demand PPP, 통합된 DHCP 지원, SCSI 서브시스템 향상, ISDN 지원, ATM 지원, FDDI, 기가 비트 이더넷(1000 Mbit) 아답터, 최신 Adaptec 컨트롤러 지원 향상 등 천여가지 이상의 버그들이 수정되었다.

기본 배포판에서는 미리 포팅되어있어서 일반적으로 사용할 수 있는 수천 개의 소프트웨어 컬렉션을 제공한다. 이 글을 쓸때는 10,500 개 이상의 포트가 있었다! 포트 리스트는 [http\(WWW\)](http://WWW) 서버부터 게임, 언어, 에디터와 거의 모든것을 포함한다. 전체 포트 컬렉션은 대략 300MB 의 공간이 필요하고 모든 포트는 실제 소스의 “델타”라고 한다. 이 시스템으로 예전 1.0 포트 컬렉션에서 요구되어왔던 쉬운 포트 업데이트와 디스크 공간을 엄청나게 줄였다.

포트를 컴파일하려면 단순히 설치하려는 프로그램의 디렉터리로 변경해서 `make install` 을 입력하면 나머지는 시스템이 처리한다. 빌드하는 각 포트의 실제 원본 소스는 CDROM 이나 FTP 사이트에서 동적으로 받아오기 때문에 원하는 포트를 빌드할 수 있는 충분한 디스크 공간만 필요하다. 또한 소스에서 포트 컴파일을 원하지않는 사람들을 위해서 대부분의 포트는 간단한 명령으로(`pkg_add`) 설치할 수 있는 미리 컴파일된 “패키지”를 제공한다. 패키지와 포트에 대한 더 많은 정보는 4 장에서 찾을 수 있다.

FreeBSD 설치와 사용에 유용한 다른 문서들도 FreeBSD 머신의 `/usr/share/doc` 디렉터리에서 찾을 수 있다. 그리고 다음 URL 에서도 매뉴얼을 볼 수 있다:

FreeBSD 핸드북

</usr/share/doc/handbook/index.html>

---

FreeBSD FAQ

`/usr/share/doc/faq/index.html`

가장 최근에 업데이트된 자료는 <http://www.FreeBSD.org/>에서 볼 수 있다.



---

## 2 장 FreeBSD 설치하기

이번 장에서는 FreeBSD 설치에 대해 설명한다. FreeBSD 와 유닉스를 처음 접하는 유저를 위해 많은 이미지로 다양한 상황에서 설치하는 방법을 소개한다. 그러나 인텔 호환 기종에서 설치하는 방법을 주로 다룬다.

### 2.1 개요

FreeBSD 는 **sysinstall** 이라는 사용하기 쉬운 텍스트 기반의 설치 프로그램을 제공한다. 벤더들이 원해서 그들의 무료 설치 프로그램을 제공하더라도 이 프로그램은 FreeBSD 의 기본 설치 프로그램이다. 이 장에서는 FreeBSD 설치에 **sysinstall** 을 어떻게 사용하는지 설명한다.

이 장을 읽고 다음 사항에 대해 알 수 있다:

- FreeBSD 설치 디스크는 어떻게 만들 수 있는가
- FreeBSD 가 하드 디스크를 나누도록 어떻게 지정하는가
- **sysinstall** 은 어떻게 시작하는가
- **sysinstall** 의 질문이 무엇을 의미하고 어떻게 대답하는가

이번 장을 읽기 전에 다음 사항에 대해 알고 있어야 된다.

- 설치하려는 FreeBSD 버전이 지원하는 하드웨어 리스트를 읽고 여러분의 하드웨어가 지원되는지 확인한다.

**Note:** 일반적으로 이 설치 문서는 i386("PC 호환") 아키텍처 컴퓨터를 위해 작성되었다. 다른 플랫폼에는 적절한 명령만(예를 들어 Alpha) 설명한다. 이 가이드가 최대한 업데이트 되어있지만 여기서 보여주는 것과 다른 인스톨러를 볼수 있을 것이다. 이번 장은 실제 설치 매뉴얼이 아닌 일반적인 가이드로 사용하길 권장한다.

---

## 2.2 설치전 작업

### 2.2.1 컴퓨터 조사

FreeBSD 를 설치하기 전에 컴퓨터의 부품 목록을 확인한다. FreeBSD 는 설치 과정에서 모델 번호와 제조사 별로 부품(하드 디스크, 네트워크 카드와 CDROM 드라이버 등)을 보여준다. 그리고 FreeBSD 는 사용하는 IRQ 나 IO 포트에 대한 정보를 포함하여 이들 장치에 정확한 설정을 시도한다. PC 하드웨어의 예측 불가능으로 이 과정이 항상 완벽하게 성공하는 것이 아니기 때문에 FreeBSD 의 설정에 대한 정확한 확인이 필요할 것이다.

윈도우나 리눅스 같은 운영체제가 설치되어 있다면 하드웨어가 어떻게 설정되어 있는지 확인하기 위해 이들 운영체제가 제공하는 기능을 사용하는 것도 좋은 생각이다. 확장 카드가 어떤 설정을 사용 중인지 확실하지 않는다면 카드에 프린트된 내용으로 찾을 수 있을 것이다. 일반적인 IRQ 번호는 3, 5, 7 이고 IO 포트 주소는 보통 0x330 형식의 16 진수로 적혀 있다.

FreeBSD 를 설치하기 전에 이런 정보를 프린트 하던지 적어두기를 권장한다. 다음과 같은 표를 이용하는 것도 도움이 될 것이다.

표 2-1. 샘플 장치 목록

장치 이름	IRQ	IO 포트	노트
첫 번째 하드 디스크	N/A	N/A	첫 번째 IDE 마스터의 시게이트 4GB 하드 디스크
CDROM	N/A	N/A	첫 번째 IDE 슬레이브
두 번째 하드 디스크	N/A	N/A	두 번째 IDE 마스터의 IBM 2GB 하드 디스크
첫 번째 IDE 컨트롤러	14	0x1f0	
네트워크 카드	N/A	N/A	Intel 10/100
모뎀	N/A	N/A	COM1 포트에 3Com 56K 팩스 모뎀

### 2.2.2 데이터 백업

중요한 데이터가 있는 곳에 FreeBSD 를 설치한다면 백업을 하고 FreeBSD 를 설치하기 전

---

에 백업한 것을 테스트한다. FreeBSD 설치 과정은 디스크에 데이터를 저장하기 전에 몇 차례에 걸쳐 문의하지만 설치가 시작된 후에는 되돌릴 수 없다.

## 2.2.3 FreeBSD 를 설치할 곳 결정

전체 디스크를 FreeBSD 로 사용한다면 이 부분에 대해 고민할 것 없이 다음 섹션으로 넘어가도 된다.

그러나 이미 설치되어있는 다른 운영체제와 FreeBSD 를 사용해야 된다면 데이터가 디스크에 어떻게 들어있는지 어떤 결과를 초래할지 어느 정도 이해하고 있어야 된다.

### 2.2.3.1 i386 의 디스크 레이아웃

PC 디스크는 여러 조각으로 분리할 수 있다. 이 조각들을 *파티션*이라고 한다. PC 는 디스크 당 4 개의 파티션만 지원할 수 있게 디자인 되어있다. 이러한 파티션을 *주 파티션*이라고 부른다. 이런 한계 때문에 4 개 이상의 파티션을 만들 수 있는 *확장 파티션*이라는 새로운 타입의 파티션이 만들어졌다. 디스크는 오직 하나의 확장 파티션만 가지고 있을 것이다. *논리 파티션*이라는 특별 파티션은 이 확장 파티션안에 만들 수 있다.

각 파티션은 파티션에있는 데이터 타입을 확인하는데 사용하는 *파티션 ID* 숫자를 가지고 있다. FreeBSD 파티션은 파티션 ID *165*를 가지고 있다.

보통 각 운영체제는 특별한 방법으로 파티션을 확인한다. 예를 들어 DOS 와 윈도우 같은 DOS 이후 버전은 각각의 주 파티션과 논리 파티션에 C:부터 시작되는 *드라이브 문자*를 할당한다.

FreeBSD 는 주 파티션에 설치해야 된다. FreeBSD 는 이 파티션에 사용자가 만드는 파일을 포함하여 모든 데이터를 가지고 있을 수 있다. 그러나 여러 개의 디스크를 가지고 있다면 전체 디스크에 FreeBSD 파티션을 생성하거나 부분적으로 만들 수 있다. FreeBSD 를 설치할 때 사용할 수 있는 파티션 하나는 가지고 있어야 된다. 이 파티션은 미리 준비하여 비어있는 파티션이거나 더 이상 사용하지 않는 데이터가있는 파티션일 것이다.

---

디스크의 모든 파티션을 사용하고 있다면 사용 중인 다른 운영체제에서(DOS 나 Windows 의 Fdisk) 제공하는 툴로 FreeBSD 를 설치할 파티션 하나를 비워야 된다.

여분의 파티션이 있다면 그것을 사용해도 된다. 그렇지 않으면 이미 나누어진 파티션 중 하나 이상의 공간을 줄여야 될 것이다.

FreeBSD 최소 설치에는 최소한 100MB 정도의 디스크 공간이 필요하다. 그러나 이 방법은 가장 최소화된 설치이기 때문에 사용자 파일에 필요한 공간이 거의 없을 것이다. 더욱 현실적인 최소설치는 그래픽 환경이 빠진 250MB 이고 그래픽 사용자 환경을 원하면 350MB 이상이 필요하다. 수많은 어플리케이션도 설치하기를 원한다면 더 많은 공간이 필요하다.

**PartitionMagic®** 같은 상업용 툴로 파티션을 다시 조정할 수 있다. CDROM 의 tools 디렉터리에 이 작업을 수행할 수 있는 **FIPS** 와 **PResizer** 라는 두 개의 무료 소프트웨어가 있다. **FIPS**, **PResizer** 와 **PartitionMagic** 은 FAT16 과 FAT32 파티션(MS-DOS 부터 Windows ME 에서 사용하는) 크기를 재 조정할 수 있다. **PartitionMagic** 만 유일하게 NTFS 를 재 조정할 수 있다고 한다. 이들 소프트웨어의 문서도 같은 디렉터리에 있다.

#### Symantec 사의 파티션 매직



**주의:** 이런 툴을 잘못 사용하면 디스크의 데이터를 삭제할 수 있다. 이런 툴을 사용하기 전에 백업을 해야된다.

#### [예제 2-1. 변경하지 않고 기존 파티션 이용]

윈도우가 설치된 4GB 디스크 하나만 있는 컴퓨터가 있고 이 디스크를 두 개의 드라이브 C:와 D:로 나누어서 각 디스크 크기는 2GB 라고 가정한다. C:에 1GB 의 데이터와 D:에 0.5GB 의 데이터를 가지고 있다.

---

이 의미는 디스크에 두 개의 파티션이 있고 각각 드라이브 문자를 가지고 있다. D:에 있는 모든 데이터를 C:로 복사해서 두 번째 파티션은 FreeBSD 용으로 비워 둘 수 있다.

### [예제 2-2. 기존 파티션 줄이기]

윈도우가 설치된 4GB 디스크가 하나 있는 컴퓨터를 가지고 있다고 가정한다. 윈도우를 설치할 때 4GB 크기 전체를 C: 드라이브로 만들었다. 현재 1.5GB의 공간을 사용 중이고 FreeBSD에 2GB를 할당하려고 한다.

FreeBSD를 설치하려면 다음 두 가지 중 한가지 작업이 필요하다:

- ① 윈도우 데이터를 백업하고 윈도우를 다시 설치할 때 2GB의 파티션을 만든다.
- ② 윈도우 파티션을 줄이기 위해 위에서 설명한 **Partition Magic** 같은 툴을 사용한다.

## 2.2.3.2 Alpha에서 디스크 레이아웃

Alpha에서는 FreeBSD 전용 디스크가 필요하다. 다른 운영체제와 디스크를 공유하는 것은 불가능하다. Alpha 머신에 따라서 부팅할 수 있는 디스크는 SCSI 또는 IDE 디스크다.

Digital/Compaq 매뉴얼 관례에 따라 모든 SRM 입력은 대문자로 보여준다. SRM은 대소문자를 구분하지 않는다.

머신에서 디스크 타입과 이름을 찾으려면 SRM 콘솔 프롬프트에서 *SHOW DEVICE* 명령을 사용한다:

```
>>>SHOW DEVICE

dka0.0.0.4.0          DKA0          TOSHIBA CD-ROM XM-57      3476
dkc0.0.0.1009.0      DKC0          RZ1BB-BS                  0658
dkc100.1.0.1009.0    DKC100        SEAGATE ST34501W         0015
dva0.0.0.0.1         DVA0
ewa0.0.0.3.0         EWA0          00-00-F8-75-6D-01
pkc0.7.0.1009.0      PKC0          SCSI Bus ID 7            5.27
pqa0.0.0.4.0         PQA0          PCI EIDE
```

---

pqb0.0.1.4.0	PQB0	PCI EIDE
--------------	------	----------

이 예제는 Digital Personal Workstation 433au 머신에 3 개의 디스크가 있음을 보여 준다. 첫 번째는 **DKA0** 라는 CDROM 드라이브이고 다른 두 개는 **DKC0** 와 **DKC100** 라는 디스크다.

DKx 라는 이름 형식은 SCSI 디스크다. 예를 들어 DKA100 은 첫 번째 SCSI 버스의 타겟 ID 가 1(A)인 SCSI 를 설명하기 때문에 DKC300 은 세 번째 SCSI(C) 버스의 SCSI ID 가 3 인 SCSI 다. 장치 이름 PKx 는 SCSI 호스트 버스 어댑터를 가리킨다. *SHOW DEVICE* 결과에서 보았듯이 SCSI CDROM 드라이브는 다른 SCSI 하드 디스크 드라이브로 간주된다.

IDE 디스크는 DQx 와 비슷한 이름을 가지고 있지만 PQx 가 IDE 컨트롤러와 연관 있다.

## 2.2.4 자세한 네트워크 구성 정보 모으기

FreeBSD 를 설치할 때 네트워크에 연결하려면(예를 들어 FTP 사이트나 NFS 서버에서 설치 한다면) 네트워크 구성을 알고 있어야 한다. 설치 중에 이 정보를 입력하여 네트워크에서 FreeBSD 를 설치할 수 있다.

### 2.2.4.1 이더넷 네트워크 또는 Cable/DSL 모뎀 연결

이더넷 네트워크, 케이블 또는 DSL 로 인터넷을 사용한다면 다음 정보가 필요하다:

- ① IP 주소
- ② 기본 게이트웨이 IP 주소
- ③ 호스트 이름
- ④ DNS 서버 IP 주소

이 정보를 모른다면 시스템 관리자나 서비스 공급자에게 문의한다. *DHCP* 로 이 정보가 자동으로 할당된다면 메모해 둔다.

---

## 2.2.4.2 모뎀으로 연결

표준 모뎀으로 ISP 에 연결한다면 시간만 오래 걸릴 뿐 인터넷으로 FreeBSD 를 설치할 수 있다.

다음 사항을 알고 있어야 한다:

- ① ISP 전화번호
- ② COM: 모뎀이 연결된 포트
- ③ ISP 계정과 패스워드

## 2.2.5 FreeBSD Errata 체크

각 릴리즈를 강화하기 위해 FreeBSD 프로젝트에서 모든 소스를 점검하지만 가끔 프로세스에 버그가 생기기도 한다. 이런 버그는 매우 드물게 설치 프로세스에 영향을 미친다. 이런 문제가 발견되어 수정된 후 FreeBSD Errata 에 (<http://www.freebsd.org/releases/5.2.1R/errata.html>) 게시된다. 설치 전에 errata 를 체크하여 기존 문제를 확인한다.

각 릴리즈의 errata 를 포함한 모든 정보는 FreeBSD 웹 사이트의 릴리즈 정보 섹션에서 (<http://www.freebsd.org/releases/index.html>) 찾을 수 있다.

## 2.2.6 FreeBSD 설치 파일 받기

FreeBSD 설치 프로세스는 다음 파일에서 FreeBSD 를 설치할 수 있다:

### 로컬 미디어

- CDROM
- 같은 컴퓨터의 DOS 파티션

- 
- SCSI 나 QIC 테잎
  - 플로피 디스크

#### 네트워크

- 필요하다면 방화벽을 지나거나 HTTP 프록시를 이용한 FTP 사이트
- NFS 서버
- 전용 패러럴이나 씨리얼 연결

FreeBSD CD 나 DVD 를 구입했다면 다음 섹션으로 넘어간다.

설치 파일을 얻지 못했다면 FreeBSD 설치 준비에 대해 설명한 섹션을 먼저 읽고 이곳으로 돌아온다.

### 2.2.7 부트 미디어 준비

FreeBSD 설치 프로세스는 다른 운영체제에서 실행하는 프로그램이 아니고 FreeBSD 인스톨러로 컴퓨터를 부팅해야된다. 보통 컴퓨터는 하드 디스크에 설치된 운영체제로 부팅하지만 부팅 가능한 플로피 디스크로 부팅하도록 설정할 수 있다. 그리고 CDROM 드라이브의 디스크에서도 부팅할 수 있다.

**Tip:** FreeBSD CDROM 이나 DVD(구입하였거나 스스로 준비한)가 있고 컴퓨터가 CDROM 이나 DVD 로(보통 BIOS 옵션에서 "Boot Order" 또는 비슷한 이름의) 부팅할 수 있다면 이번 섹션은 넘어가도 된다. FreeBSD CDROM 과 DVD 이미지로 부팅할 수 있기 때문에 특별한 준비없이 FreeBSD 를 설치할 수 있다.

#### [부트 플로피 이미지 생성]

##### 1. 부트 플로피 이미지 받기

부트 디스크는 설치 미디어의 floppies/ 디렉터리에있고 i386 아키텍처용 플로피 이미지는 <ftp://ftp.freebsd.org/pub/FreeBSD/releases/i386/5.2.1-RELEASE/floppies/>에서 그리고 Alpha 아키텍처용은



<ftp://ftp.freebsd.org/pub/FreeBSD/releases/alpha/5.2.1-RELEASE/floppies/>에서 다운로드 할수 있다.

플로피 이미지는 **.flp** 확장자를 가지고있다. floppies/ 디렉터리는 여러개의 이미지를 가지고있기 때문에 설치하려는 FreeBSD 버전과 하드웨어에 맞는 이미지가 필요하다. 보통 **kern.flp** 와 **mfsroot.flp** 파일 두개만 필요하지만 특정 시스템에는 추가적인 장치 드라이버가 필요할 수 있다. 이런 드라이버는 **drivers.flp** 이미지에서 제공된다. 이들 플로피 이미지에 대한 가장 최근 업데이트 정보는 같은 디렉터리의 README.TXT 를 체크한다.

**중요:** 이들 디스크 이미지를 다운로드 하려면 FTP 프로그램을 바이너리 모드로(*binary mode*) 사용해야 된다. 어떤 웹 브라우저는 디스크로 부팅할 수 없을 때 문제가되는 Text 모드(또는 *ASCII*)로 지정한다.

## 2. 플로피 디스크 준비

다운로드 한 이미지당 디스크 하나씩을 준비한다. 이런 디스크는 결함이 없어야 된다. 플로피를 테스트하는 가장 쉬운 방법은 이미 포맷된 플로피를 믿지말고 직접 포맷한다.

**중요:** FreeBSD 설치를 했을 때 설치 프로그램이 충돌, 반응이 없거나 이상하게 동작 한다면 첫째로 플로피를 의심해본다. 다른 디스크에 플로피 이미지를 넣고 다시 시도해본다.

## 3. 플로피 디스크에 이미지 파일넣기

**.flp** 파일은 디스크에 복사할 수 없는 완벽한 디스크의 내용이다. 이 의미는 DOS의 복사 명령으로 복사하지 못한다는 것이다. 대신 디스크에 이미지를 직접 작성하는 특별한 툴을 사용한다.

DOS/Windows 가 실행 중인 컴퓨터에서 플로피를 만든다면 우리는 **fdimage** 라는 툴을 제공한다.

E: 드라이브에 있는 CDROM 으로 플로피를 만든다면 다음 명령을 실행하다:

```
E:W> /tools/fdimage floppies/kern.flp A:
```

---

각 **.flp** 파일별로 플로피 디스크를 교체하면서 위 명령을 반복한다. 물론 디스크 라벨과 복사된 파일 이름은 일치해야된다. 필요 하다면 **.flp** 파일 위치에 따라 명령어 라인을 수정한다. CDROM이 없다면 fdimage를 FreeBSD FTP 사이트의 tools디렉터리에서(<ftp://ftp.FreeBSD.org/pub/FreeBSD/tools/>) 다운로드 할수 있다.

유닉스 시스템(다른 FreeBSD 시스템에서)에서 플로피를 만든다면 dd(1) 명령으로 이미지 파일을 디스크에 직접 작성할 수 있다. FreeBSD 에서 다음과 같이 실행한다:

```
# dd if=kern.flp of=/dev/fd0
```

FreeBSD 에서 /dev/fd0 는 첫 번째 플로피 디스크를 의미한다(A: 드라이브). 그리고 /dev/fd1 는 B: 드라이브를 말한다. 다른 유닉스에서는 플로피 디스크 장치를 다른 이름으로 부르기 때문에 필요하다면 그 시스템 매뉴얼을 참조한다.

이제 FreeBSD 설치를 시작할 준비가 되었다.

## 2.3 설치 시작

기본적으로 다음 메시지가 나올 때까지 FreeBSD 설치는 디스크의 내용을 변경하지 않는다.

```
Last Chance: Are you SURE you want continue the installation?
```

```
If you're running this on a disk with data you wish to save then WE STRONGLY  
ENCOURAGE YOU TO MAKE PROPER BACKUP before proceeding?
```

```
We can take no responsibility for lost disk contents!
```

하드 디스크의 데이터를 변경하지 않고 마지막 경고 메시지 이전에 언제라도 설치 과정을 중단할 수 있다.

---

## 2.3.1 부팅

### [i386 부팅 과정]

① 컴퓨터를 켜다. 시작하면서 보통 **F2, F10, Del** 또는 **Alt+S** 와 같은 키로 들어갈 수 있는 시스템 설정 메뉴나 BIOS 로 들어갈 수 있는 옵션을 보여준다. 어떤 키를 사용하는지 화면에 나타난다. 어떤 컴퓨터에서는 시작되는 동안 그래픽 화면이 나타날 수 있다. 보통 **Esc** 키로 그래픽 화면에서 빠져 나와서 필요한 메시지를 볼수 있다.

② 어떤 장치에서 시스템이 부팅할지 조정하는 설정을 찾는다. 이 설정은 보통 “Boot Order”라고 표시되어있고 *Floppy, CDROM, First Hard Disk* 처럼 장치들을 보여준다.

필요한 부트 플로피를 준비하였다면 플로피 디스크를 선택한다. CDROM 으로 부팅한다면 CDROM 을 선택한다. 의심스럽다면 컴퓨터나 마더보드 매뉴얼을 참고한다.

부팅 순서를 변경한 후 저장하고 빠져 나온다. 컴퓨터는 이제 다시 시작된다.

③ 2.2.7 장에서 설명한 것처럼 필요한 부트 플로피를 준비했다면 그중 하나는 첫 번째 부트 디스크며 **kern.flp** 를 포함한 디스크가 첫 번째일 것이다. 플로피 드라이브에 이 디스크를 넣는다.

CDROM 으로 부팅한다면 컴퓨터를 켜고 CDROM 을 바로 넣는다.

일반적으로 컴퓨터가 시작되어 이전 운영체제를 로드한다면 두 가지 경우 중 한 가지다:

1. 부팅할 때 디스크를 빨리 넣지 않았다면 디스크를 그대로두고 컴퓨터를 다시 시작한다.
2. BIOS 에서 제대로 변경하지 않았다. 맞는 설정이 될때까지 BIOS 옵션을 반복한다.
3. 특정 BIOS 는 원하는 미디어 부팅을 지원하지 않는다.

- ④ FreeBSD 가 부팅을 시작한다. CDROM 으로 부팅했다면 다음과 비슷한 화면을 볼 수 있다(버전 정보 생략):

```
Verifying DMI Pool Data .....
```

```
Boot from ATAPI CD-ROM :
```

```
1. FD 2.88MB System Tye- (00)
```

```
Uncompressing ... done
```

```
BTX loader 1.00 BTX version is 1.01
```

```
Console: internal video/keyboard
```

```
BIOS drive A: is disk0
```

```
BIOS drive B: is disk1
```

```
BIOS drive C: is disk2
```

```
BIOS drive C: is disk3
```

```
BIOS 639KB/261120kB available memory
```

```
FreeBSD/i386 bootstrap loader, Revision 0.8
```

```
/kernel text=0x277391 data=0x3268c+0x332a8 |
```

```
|
```

```
Hit [Enter] to boot immediately, or any other key for command prompt.
```

```
Booting [kernel] in 9 seconds... _
```

플로피 디스크로 부팅했다면 다음과 비슷한 화면을 볼 수 있다. (버전 정보 생략)

```
Verifying DMI Pool Data .....
```

```
BTX loader 1.00 BTX version is 1.01
```

```
Console: internal video/keyboard
```

```
BIOS drive A: is disk0
```

```
BIOS drive C: is disk1
```

```
BIOS 639kB/261120kB available memory
```

```
FreeBSD/i386 bootstrap loader, Revision 0.8
```

```
/kernel text=0x277391 data=0x3268c+0x332a8 |
```

please insert MFS root floppy and press enter:

**kern.flp** 디스크를 빼고 **mfsroot.flp** 디스크를 넣고 **Enter** 를 누르라는 지시를 따른다.

- ⑤ 플로피 또는 CDROM 부팅이든 상관없이 부트 프로세스는 이 부분을 가리킨다:

```
Hit [Enter] to boot immediately, or any other key for command prompt.  
Booting [kernel] in 9 seconds...._
```

10 초를 기다리거나 **Enter** 를 누른다. 이제 커널 설정 메뉴가 시작된다.

### [Alpha 에서 부팅 과정]

- ① 컴퓨터를 켜고 부트 모니터 프롬프트를 기다린다.
- ② 2.2.7 장에서 설명했던 부트 플로피를 준비했다면 그중 하나는 첫 번째 부트 디스크고 **kern.flp** 를 포함한것이 첫 번째일 것이다. 플로피 드라이브에 이 디스크를 넣고 디스크로 부팅하기 위해 다음 명령을 입력한다(필요하다면 플로피 드라이브 이름을 대신 넣는다):

```
>>>BOOT DVA0 -FLAGS " -FILE"
```

CDROM 으로 부팅한다면 CDROM 을 드라이브에 넣고 설치를 시작하기 위해 다음 명령을 입력한다(필요하다면 CDROM 드라이브 이름을 넣는다):

```
>>>BOOT DKA0 -FLAGS " -FILE"
```

- ③ 이제 FreeBSD 가 부팅을 시작한다. 플로피 디스크로 부팅했다면 특정 부분에서 다음 메시지를 보게된다:

```
Please insert MFS root floppy and press enter:
```

---

**kern.flp** 디스크를 꺼내고 **mfsroot.flp** 디스크를 넣은 후 **Enter** 를 누르라는 지시를 따른다.

- ④ 플로피 또는 CDROM 으로 부팅하던지 부트 프로세스는 다음 부분을 가리킨다:

```
Hit [Enter] to boot immediately, or any other key for command prompt.  
Booting [kernel] in 9 seconds... _
```

10 초를 기다리거나 **Enter** 를 누른다. 이제 커널 설정 메뉴를 시작한다.

## 2.3.2 커널 설정

*kernel*은 운영체제의 핵심이다. 이것은 하드 디스크, 네트워크 카드, 사운드 카드처럼 시스템에있는 모든 장치와 많은것을 제어한다. 각 하드웨어는 관련된 드라이버를 가지고있는 FreeBSD 커널에의해 지원된다. 각 드라이버는 SCSI 순차 접근 드라이버 *sa*, 시리얼 I/O 드라이버는(COM 포트를 관리하는) *sio* 처럼 두개나 새개의 문자 명을 가지고 있다.

**Note:** FreeBSD 버전 5.0 과 이후 버전은 새로운 *device.hints(5)* 기능으로 유저 설정이 많이 줄어들었다. *device.hints(5)*에 대한 더 많은 정보는 12.5 장을 확인한다.

커널이 시작될 때 각 드라이버는 시스템에있는 하드웨어를 지원하는지 확인하기위해 시스템을 체크한다. 그리고 드라이버는 하드웨어를 설정하고 커널의 나머지를 활성화한다.

이 체크 방식을 보통 *장치/탐색*이라고 한다. 불행히 장치 탐색이 항상 안전하지는 않다. 어떤 하드웨어 드라이버는 없기도하고 하드웨어 검색은 가끔 다른 드라이버를 불안정한 상태로 방치한다. 이것이 PC 디자인의 기본적인 한계다.

PCI 장치와 반대되는 오래된 장치를 ISA 장치라고 한다. ISA 방식은 칩에 내장되어 일반적으로 드라이버가 사용하는 인터럽트 요청 번호와(IRQ) IO 포트 주소가 필요하다. 이 정보는 보통 카드의 물리적인 점퍼를(jumpers) 사용하거나 DOS 기반의 유틸리티로 지정한다.

두 장치가 같은 IRQ 나 포트 주소를 공유하지 못하기 때문에 가끔 문제의 소지가 된다.

---

새로운 장치는 필요한 IRQ와 IO 포트 주소를 BIOS와 상호작용하여 지정하기 때문에 IRQ와 IO 포트 주소가 필요없는 PCI 방식을 따른다.

ISA 장치를 가지고 있다면 이 장치의 FreeBSD 드라이버는 카드에 설정되어있는 IRQ와 포트 주소로 설정해야 된다. 이 정보 때문에 하드웨어의 목록이(2.2.1 장을 본다)필요하다.

불행히 서로 다른 드라이버가 기본 IRQ와 메모리 포트를 사용한다면, 이 의미는 어떤 ISA 장치가 충돌하는 IRQ와 메모리 포트에 지정되어 있다는 것이다. 그러나 FreeBSD 드라이버에서 기본값은 제조사의 기본값을 반영해서 신중히 설정되므로 가능한 많은 장치가 동작한다.

커널이 가능한 많은 드라이버를 가지고 설치를 수행하기 때문에 많은 하드웨어 설정을 지원할 수 있어서 FreeBSD를 설치할 때는 문제가 되지 않는다. 이 의미는 이들 드라이버 중 어떤 것은 상반되는 설정을 가지게 된다는 말이다. 장치는 정확한 순서로 검색되기 때문에 나중에 검색한 장치가 이전에 검색된 것과 충돌한다면 하드웨어 상태가 이상하거나 FreeBSD를 설치할때 정확히 탐색되지 않았을 것이다.

왜냐하면 FreeBSD를 설치할 때 첫 번째로 검색한 것은 커널에 설정되어있는 드라이버 리스트에서 찾은것이고 그 중 몇 개는 원하지 않았기에 비활성 시켰거나 기본값이 잘못된 장치의 드라이버 설정을 확인했기 때문일 것이다

그림 2-1은 첫 번째 커널 설정 메뉴를 보여준다. 우리는 새로운 유저에게 가장 쉬운 인터페이스를 제공하는 **Start kernel configuration in full-screen visual mode** 옵션을 권장한다.

### 그림 2-1. 커널 설정 메뉴

```
Kernel Configuration Menu

Skip kernel configuration and continue with installation
Start kernel configuration in full-screen visual mode
Start kernel configuration in CLI mode

Here you have the chance to go into kernel configuration mode, making
any changes which may be necessary to properly adjust the kernel to
match your hardware configuration.

If you are installing FreeBSD for the first time, select Visual Mode
(press Down-Arrow then ENTER).

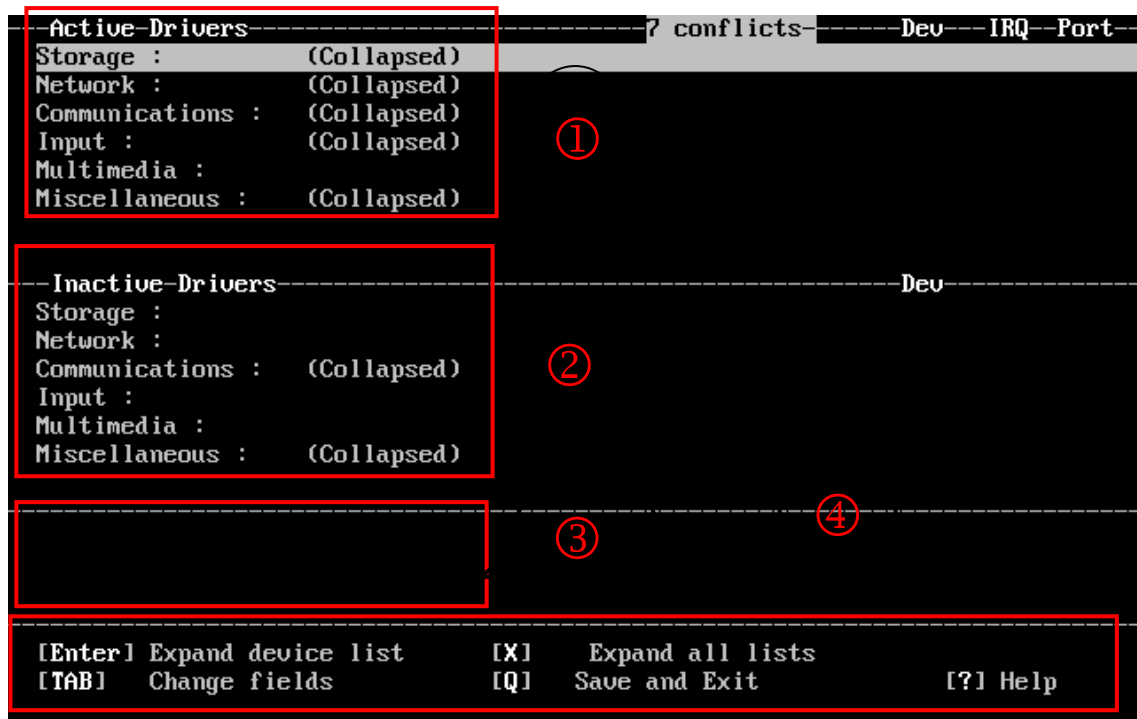
If you need to do more specialized kernel configuration and are an
experienced FreeBSD user, select CLI mode.

If you are certain that you do not need to configure your kernel
then simply press ENTER or Q now.
```

커널 설정 화면은(그림 2-2)4 개의 섹션으로 나누어 진다.

- ① 현재 “active” 표시가 되어있는 접을 수 있는 모든 드라이버 리스트는 *Storage* 와 *Network* 같은 그룹으로 다시 나뉘진다. 각 드라이버는 화면처럼 두개나 새개의 드라이버 이름과 드라이버가 사용하는 IRQ 및 메모리 포트를 보여준다. 그리고 활성화된 드라이버가 다른 활성화된 드라이버와 충돌 난다면 드라이버 이름 다음에 *CONF*가 보인다. 또한 이 섹션에서 현재 활성화되고 충돌되는 모든 드라이버를 보여준다.
- ② inactive 표시가 된 드라이버들은 커널에 남아있지만 커널이 시작될때 이 장치들을 탐색하지 않는다. 이것도 활성화된 드라이버 리스트처럼 세부적으로 나누어있다.
- ③ IRQ 와 메모리 포트 주소를 포함하여 현재 선택한 드라이버에 대해 더 자세히 보여 준다.
- ④ 여기서 사용할 수 있는 키 입력 정보를 보여 준다.

그림 2-2 커널 장치 설정 비주얼 인터페이스





이곳에서는 항상 충돌된 리스트가 있다. 이것은 이미 예상한 것이기 때문에 걱정하지 않아도 된다. 이전에 설명한 대로 모든 드라이버는 활성화 되어있고 드라이버 몇개는 다른 것과 충돌된다.

이제 드라이버 리스트를 통해 충돌을 해결해야된다.

### [드라이버 충돌 해결]

- ① X를 누른다. 이것은 드라이버 리스트를 완전히 확장하기 때문에 모든 드라이버를 볼수 있다. 활성화된 드라이버 리스트에서 전후로 움직이기 위해 방향키를 사용한다.

그림 2-3은 X를 눌렀을 때 결과를 보여 준다.

그림 2-3. 확장된 드라이버 리스트

```

Active Drivers-----? conflicts-----Dev---IRQ--Port--
Storage :
AdvanSys SCSI narrow controller          adv0
Adaptec 154x SCSI controller             aha0
Adaptec 152x SCSI and compatible sound cards  aic0
ATA/ATAPI compatible disk controller      ata0      14 0x1f0
ATA/ATAPI compatible disk controller      ata1      15 0x170
Buslogic SCSI controller                 bt0
Floppy disk controller                   fdc0      6  0x3f0
-----
Inactive Drivers-----Dev-----
Storage :
Network :
Communications : (Collapsed)
Input :
Multimedia :
Miscellaneous : (Collapsed)
-----

[Enter] Collapse device list  [C]   Collapse all lists
[TAB]  Change fields         [Q]   Save and Exit           [?] Help
  
```

- ② 컴퓨터에 없는 장치 드라이버를 비활성 한다. 드라이버를 비활성 하려면 방향키로 선택한 후 Del 키를 누른다. 이 드라이버는 *Inactive Drivers* 리스트로 이동된다.

**주의:** sc0 는 비활성하지 않는다. 이것은 화면을 조정하기 때문에 시리얼 케이블로 설치하지 않는다면 필요하다.

원하지 않는 장치를 비활성했다면 **Tab** 키를 눌러서 *Inactive Drivers* 리스트로 변경한 후 비활성한 드라이버를 선택하고 active 리스트로 보내기 위해 **Enter** 를 누른다.

**주의:** USB 키보드를 사용한다면 atkbd0 만 비활성 한다. 일반적인 키보드를 가지고 있다면 atkbd0 가 필요하다.

- ③ 충돌된 리스트가 없다면 이번 단계는 지나쳐도 된다. 그렇지 않으면 남아있는 충돌된 드라이버를 해결 해야 된다. 메시지 영역에 "allowed conflict"가 표시되지 않았다면 장치 탐색이 필요한 IRQ/주소를 변경하던지 하드웨어의 IRQ/주소를 변경해야 된다.

드라이버의 IRQ 와 IO 포트 주소 설정을 변경하려면 장치를 선택하고 **Enter** 를 누른다. 커서가 화면의 세 번째 섹션으로 이동하고 값을 변경할 수 있다. 하드웨어 목록을 만들 때 발견한 IRQ 와 포트 주소로 값을 입력한다. **Q** 를 눌러 장치 설정을 끝내고 active 드라이버 리스트로 돌아온다.

어떻게 해야 될지 확실하지 안다면 **-1** 을 사용해 본다. **-1** 값을 사용하여 어떤 FreeBSD 드라이버에 어떤 값이 적당한지 확인하기 위해 안전하게 하드웨어를 검색한다.

하드웨어 변경에서 이 절차는 장치에서 장치로 주소를 변경한다. 어떤 장치는 컴퓨터에서 카드를 빼서 점퍼나 DIP 스위치 설정을 변경해야 된다. 다른 카드는 카드 재 설정에 사용하는 DOS 플로피를 가지고 있을 것이다. 어떤 경우든 장치 매뉴얼을 참조한다. 이 과정이 끝나면 컴퓨터가 재 시작하기 때문에 카드를 재 설정하고 FreeBSD 설치 절차로 다시 들어가야 된다.

- ④ 모든 충돌이 해결되었을 때 화면은 다음 그림 2-4 와 비슷할 것이다.

**그림 2-4. 충돌 없는 드라이버 설정**

```

-----Active-Drivers-----Dev---IRQ---Port---
Storage :
ATA/ATAPI compatible disk controller  ata0   14 0x1f0
ATA/ATAPI compatible disk controller  ata1   15 0x170
Floppy disk controller                 fdc0    6  0x3f0
Network :
NE1000,NE2000,3C503,WD/SMC80xx Ethernet adapters  ed0    10 0x280
Communications :
Parallel Port chipset                  ppc0    7
-----Inactive-Drivers-----Dev-----
Storage : (Collapsed)
Network : (Collapsed)
Communications : (Collapsed)
Input :
Multimedia :
Miscellaneous : (Collapsed)

-----
Port address : 0x1f0
IRQ number   : 14
Flags       : 0x0

-----
[Enter] Edit device parameters  [DEL] Disable device
[TAB]  Change fields            [Q]   Save and Exit           [?] Help

```

화면은 실제로 가지고있는 드라이버만 리스트 되었기 때문에 active 드라이버 리스트는 이제 훨씬 적다.

변경한 것을 저장하고 다음 설치 단계로 이동한다. Q를 눌러 장치 설정 인터페이스를 빠져 나오면 다음 메시지가 나타난다:

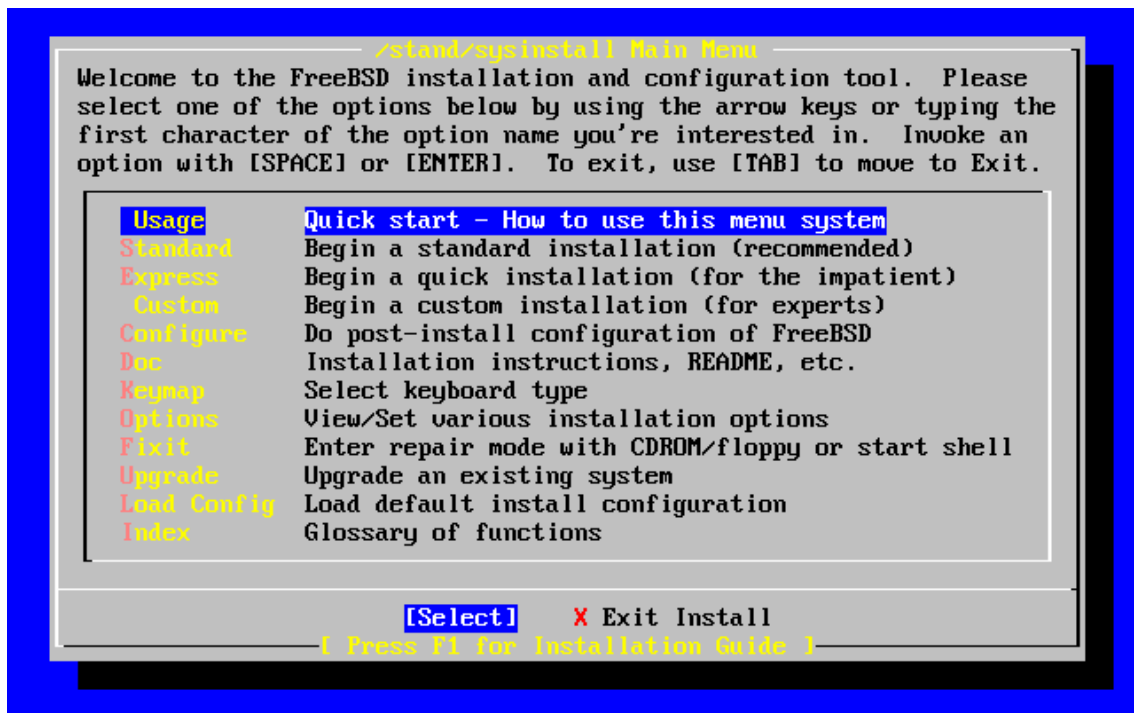
```

Save these parameters before exiting? ([Y]es/ [N]o/ [C]ancel)

```

매개 변수를 메모리에 저장하기 위해 Y를 누르면(설치가 끝나면 디스크에 저장된다) 탐색이 시작된다. 탐색 결과가 검은 화면에 하얀 문자로 표시된 후 **sysinstall**이 시작되고 메인 메뉴가 표시된다. (그림 2-5)

그림 2-5. Sysinstall 메인 메뉴



### 2.3.3 장치 탐색 결과 다시 확인하기

화면에 보여진 수백 라인은 저장되어있기 때문에 다시 볼수 있다.

버퍼를 다시 보려면 **Scroll Lock** 을 누른다. 이 키는 화면에서 스크롤을 켜다. 그리고 결과를 보기위해 방향키나 **PageUp** 과 **PageDown** 을 사용할 수 있다. **Scroll Lock** 를 다시 누르면 스크롤이 멈춘다.

이런 식으로 커널이 장치 탐색을 했을 때 스크롤이 꺼진 화면의 문자를 다시 볼수 있다. 컴퓨터에 가지고있는 장치에 따라 문자는 다르지만 그림 2-6 과 비슷한 문자를 볼 것이다.

그림 2-6. 전형적인 장치 탐색 결과

```

avail memory = 253050880 (247120K bytes)
Preloaded elf kernel "kernel" at 0xc0817000.
Preloaded mfs_root "/mfsroot" at 0xc0817084.
md0: Preloaded image </mfsroot> 4423680 bytes at 0xc03ddcd4

md1: Malloc disk
Using $PIR table, 4 entries at 0xc00fde60
npx0: <math processor> on motherboard
npx0: INT 16 interface
pcib0: <Host to PCI bridge> on motherboard
pci0: <PCI bus> on pcib0
pcib1: <VIA 82C598MVP (Apollo MVP3) PCI-PCI (AGP) bridge> at device 1.0 on pci0
pci1: <PCI bus> on pcib1
pci1: <Matrox MGA G200 AGP graphics accelerator> at 0.0 irq 11
isab0: <VIA 82C586 PCI-ISA bridge> at device 7.0 on pci0
isa0: <ISA bus> on isab0
atapci0: <VIA 82C586 ATA33 controller> port 0xe000-0xe00f at device 7.1 on pci0
ata0: at 0x1f0 irq 14 on atapci0
ata1: at 0x170 irq 15 on atapci0
uhci0 <VIA 83C572 USB controller> port 0xe400-0xe41f irq 10 at device 7.2 on pci0
usb0: <VIA 83572 USB controller> on uhci0
usb0: USB revision 1.0
uhub0: VIA UHCI root hub, class 9/0, rev 1.00/1.00, addr1
uhub0: 2 ports with 2 removable, self powered
pci0: <unknown card> (vendor=0x1106, dev=0x3040) at 7.3
dc0: <ADMtek AN985 10/100BaseTX> port 0xe800-0xe8ff mem 0xdb000000-0xeb0003ff irq 11 at device 8.0 on pci0
dc0: Ethernet address: 00:04:5a:74:6b:b5
miibus0: <MII bus> on dc0
ukphy0: <Generic IEEE 802.3u media interface> on miibus0
ukphy0: 10baseT, 10baseT-FDX, 100baseTX, 100baseTX-FDX, auto
ed0: <NE2000 PCI Ethernet (RealTek 8029)> port 0xec00-0xec1f irq 9 at device 10.0 on pci0
ed0 address 52:54:05:de:73:1b, type NE2000 (16 bit)
isa0: too many dependant configs (8)
isa0: unexpected small tag 14
orm0: <Option ROM> at iomem 0xc0000-0xc7fff on isa0
fdc0: <NEC 72065B or clone> at port 0x3f0-0x3f5,0x3f7 irq 6 drq2 on isa0
fdc0: FIFO enabled, 8 bytes threshold
fd0: <1440-KB 3.5" drive> on fdc0 drive 0
atkbd0: <Keyboard controller (i8042)> at port 0x60,0x64 on isa0
atkbd0: <AT Keyboard> flags 0x1 irq1 on atkbd0
kbd0 at atkbd0
psm0: <PS/2 Mouse> irq 12 on atkbd0
psm0: model Generic PS/2 mouse, device ID 0
vga0: <Generic ISA VGA> at port 0x3c0-0x3df iomem 0xa0000-0xbffff on isa0
sc0: <System console> at flags 0x100 on isa0
sc0: VGA <16 virtual consoles, flags=0x300>
sio0 at port 0x3f8-0x3ff irq 4 flags 0x10 on isa0
sio0: type 16550A
sio1 at port 0x2f8-0x2ff irq 3 on isa0
sio1: type 16550A
ppc0: <Parallel port> at port 0x378-0x37f irq 7 on isa0
pppc0: SMC-like chipset (ECP/EPP/PS2/NIBBLE) in COMPATIBLE mode
ppc0: FIFO with 16/16/15 bytes threshold
plip0: <PLIP network interface> on ppbus0
ad0: 8063MB <IBM-DHEA-38451> [16383/16/63] at ata0-master UDMA33
acd0: CD-RW <LITE-ON LTR-1210B> at ata1-slave P104
Mounting root from ufs:/dev/md0c
/stand/sysinstall running as init on vty0

```

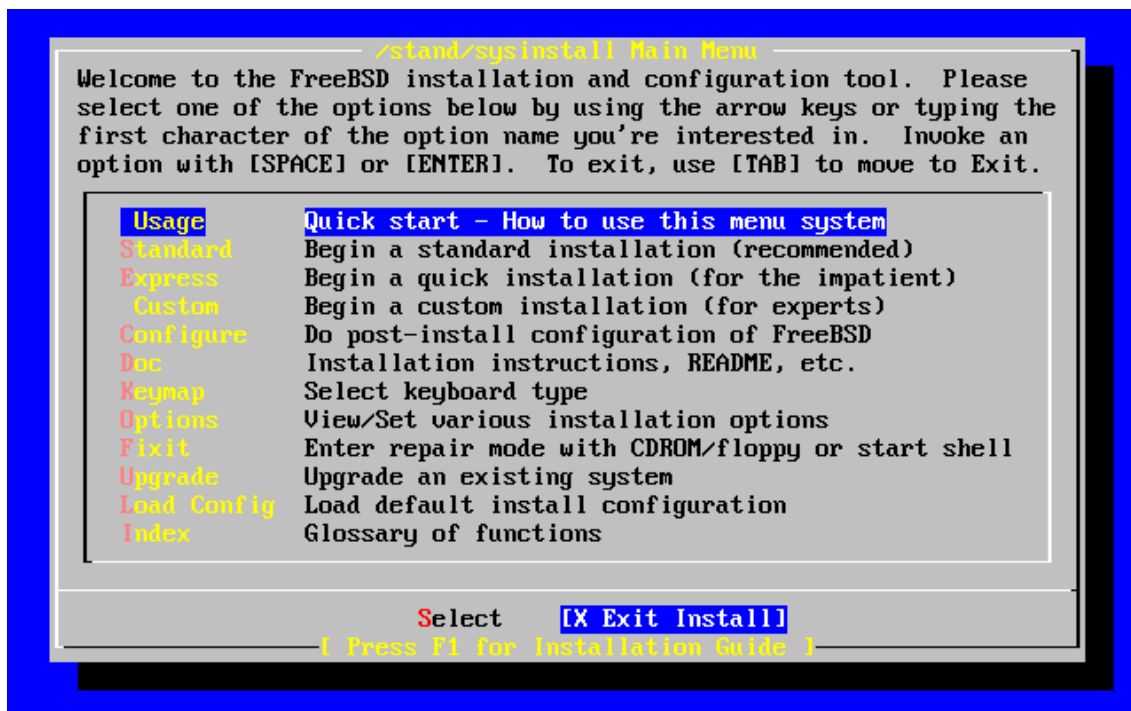
FreeBSD 가 예상했던 모든 장치를 찾았는지 탐색 결과를 주의 깊게 확인한다. 장치를 찾지

---

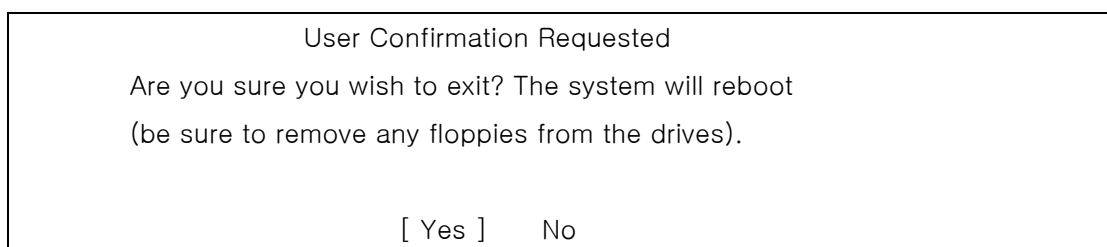
못했다면 표시되지 않는다. 장치 드라이버에 IRQ 및 포트 주소 설정이 필요하다면 체크하여 정확하게 입력한다.

UserConfig 장치 탐색에 변경이 필요하면 **sysinstall** 프로그램을 빠져나와 다시 시작하는 것은 쉽다. 또한 설치 프로세스와 더 친숙해지는것도 좋은 방법이다.

그림 2-7. Sysinstall Exit 선택



메인 설치 메뉴 화면에서 방향키로 **Exit Install** 을 선택한다. 그러면 다음 메시지가 나타난다:



CDROM 이 드라이브에 남아있고 **[Yes]**를 선택했다면 설치 프로그램은 다시 시작된다.

---

플로피로 부팅했다면 `mfsroot.flp` 플로피를 빼고 `kern.flp` 를 부팅하기 전에 넣어야 된다.

## 2.4 Sysinstall 소개

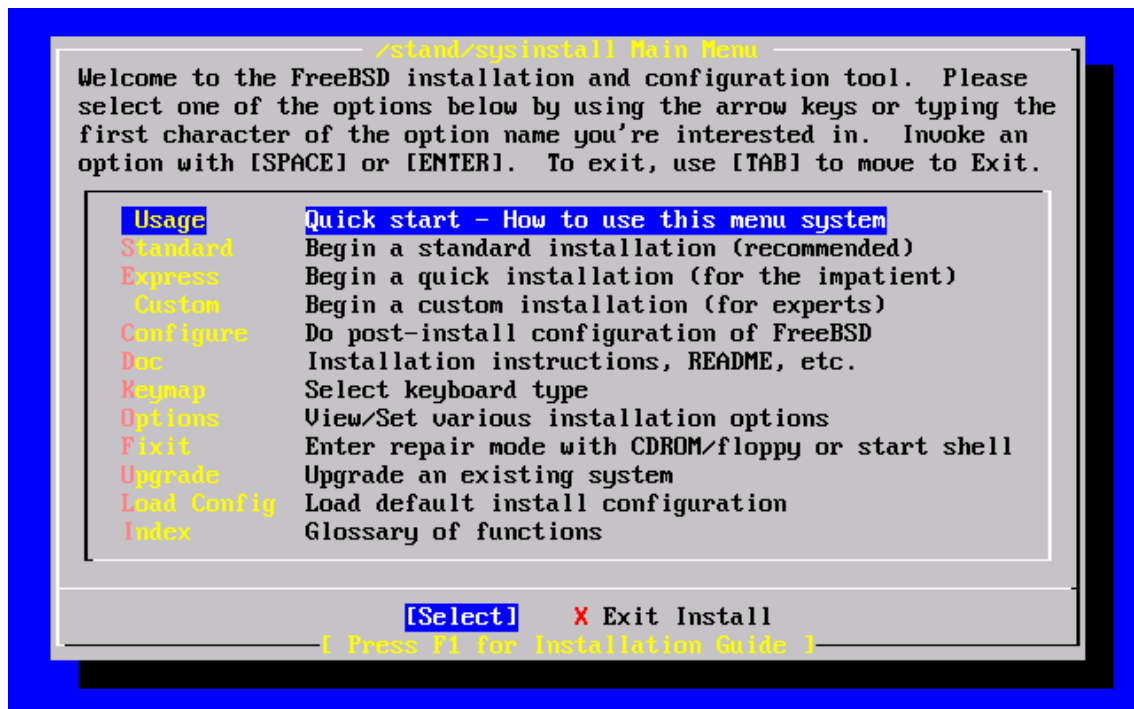
`sysinstall` 유틸리티는 FreeBSD 프로젝트에서 제공하는 설치 어플리케이션이다. 이것은 콘솔 기반이고 여러 메뉴로 나누어져 있으며 설치 프로세스를 설정하고 제어하는데 사용할 수 있는 화면이다.

`sysinstall` 메뉴 시스템은 방향키, **Enter**, **Space** 그리고 다른 키로 제어된다. 이런 키에 대한 자세한 설명과 용도는 `sysinstall` 의 사용법 정보에 포함되어 있다.

이 정보를 다시 보려면 그림 2-8 에서 보여주는 것처럼 **Usage** 엔트리가 밝게 되었을때 **[Select]** 버튼을 선택하고 **Enter** 를 누른다.

메뉴 시스템 사용법이 표시된다. 사용법을 다시 보고 **Enter** 를 눌러 메인 메뉴로 돌아간다.

그림 2-8. Sysinstall 메인 메뉴에서 Usage 선택

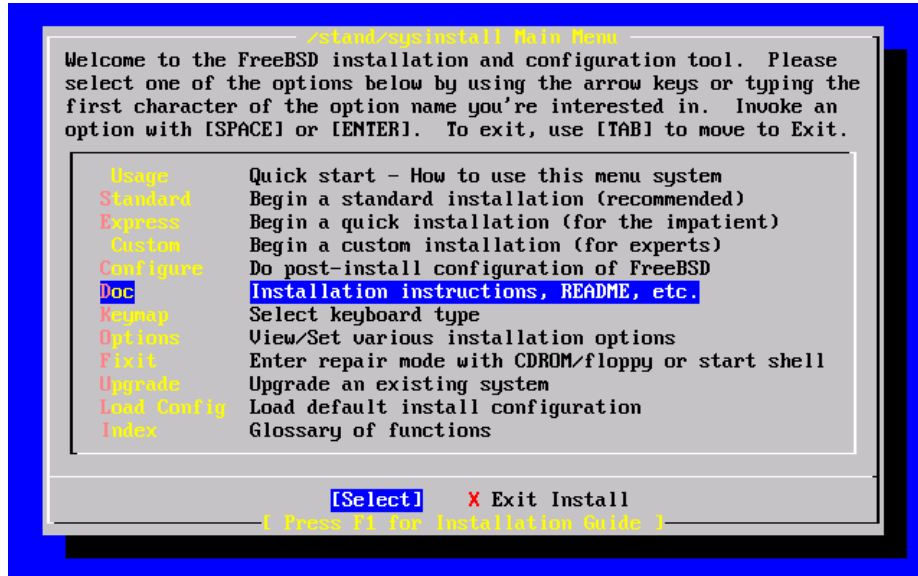


---

## 2.4.1 문서 메뉴 선택

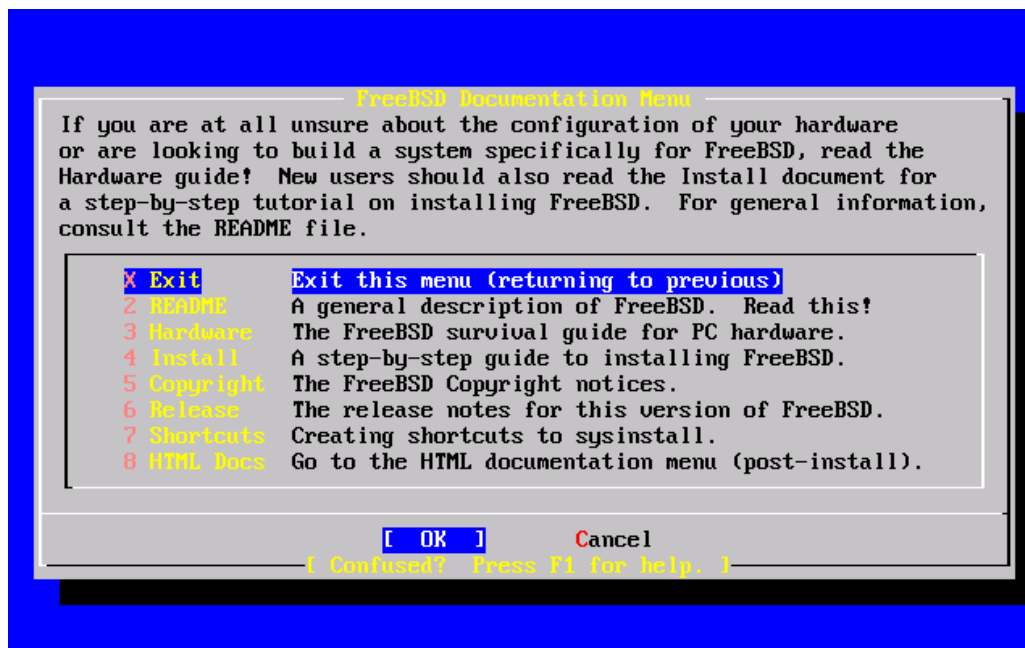
메인 메뉴에서 방향키로 Doc 를 선택하고 Enter 를 누른다.

그림 2-9. 문서 메뉴 선택



이 화면은 문서 메뉴를 보여준다.

그림 2-10. Sysinstall 문서 메뉴





---

정확한 설치를 위해 제공된 문서를 읽는 것도 중요하다.

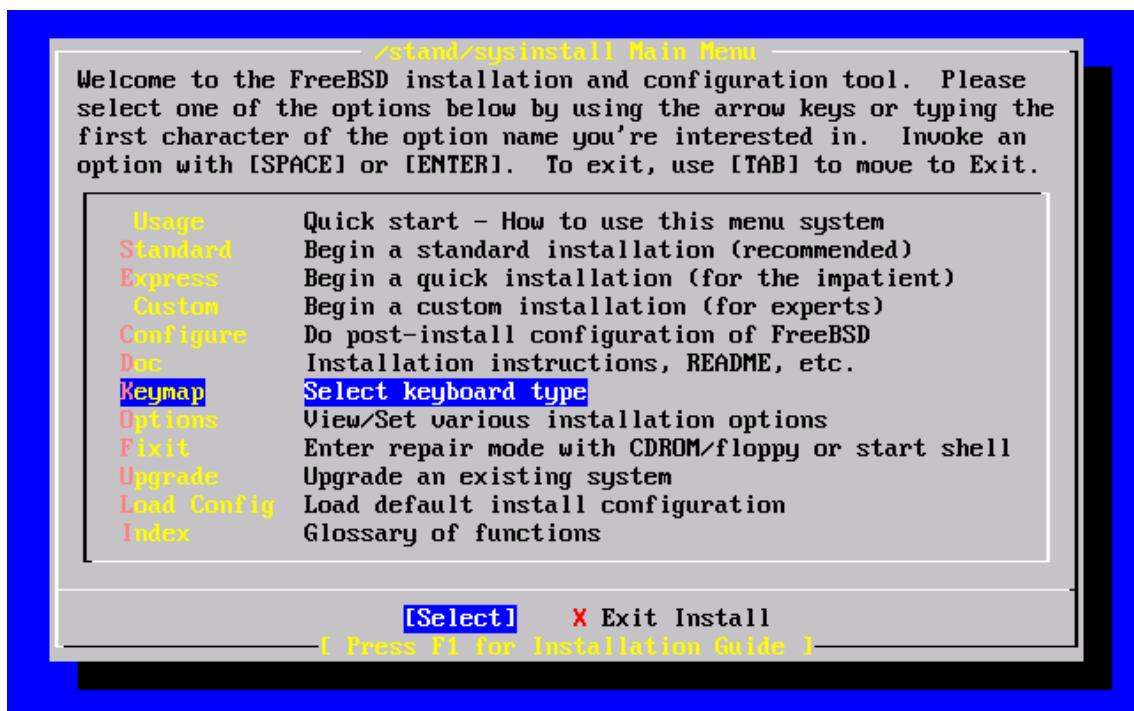
문서를 보려면 방향키로 선택하고 **Enter** 를 누른다. 문서 읽기가 끝나면 **Enter** 를 눌러서 문서 메뉴로 돌아온다.

메인 설치 메뉴로 돌아오려면 방향키로 **Exit** 를 선택하고 **Enter** 를 누른다.

## 2.4.2 keymap 메뉴 선택

키보드 맵을 변경하려면 메뉴에서 방향키로 **keymap** 을 선택하고 **Enter** 를 누른다.

그림 2-11. Sysinstall 메인 메뉴

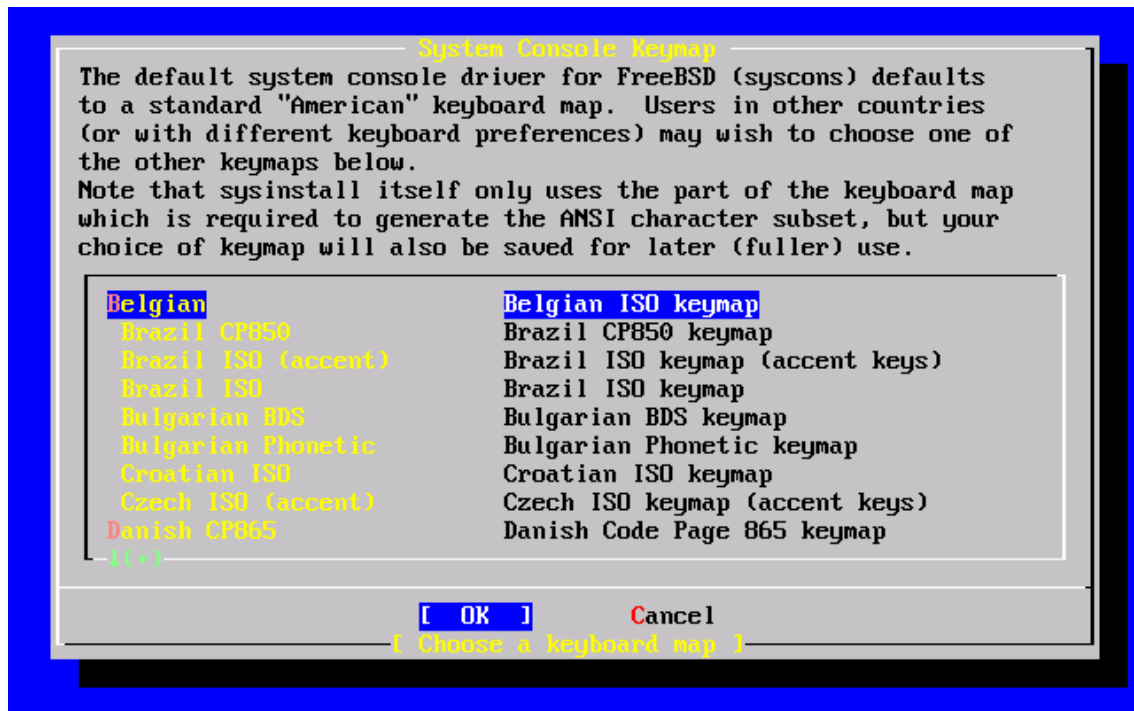


다른 키보드 맵은 up/down 방향키로 메뉴 아이템을 선택하고 **Space** 를 누른다. 다시 **Space** 를 누르면 아이템 선택이 취소된다. 끝나고 방향키로 **[OK]**를 선택한 후 **Enter** 를 누른다.

---

부분적인 리스트만 이 화면에 표시된다. [Cancel]을 선택하면 기본 키 맵을 사용하고 메인 설치 메뉴로 돌아간다.

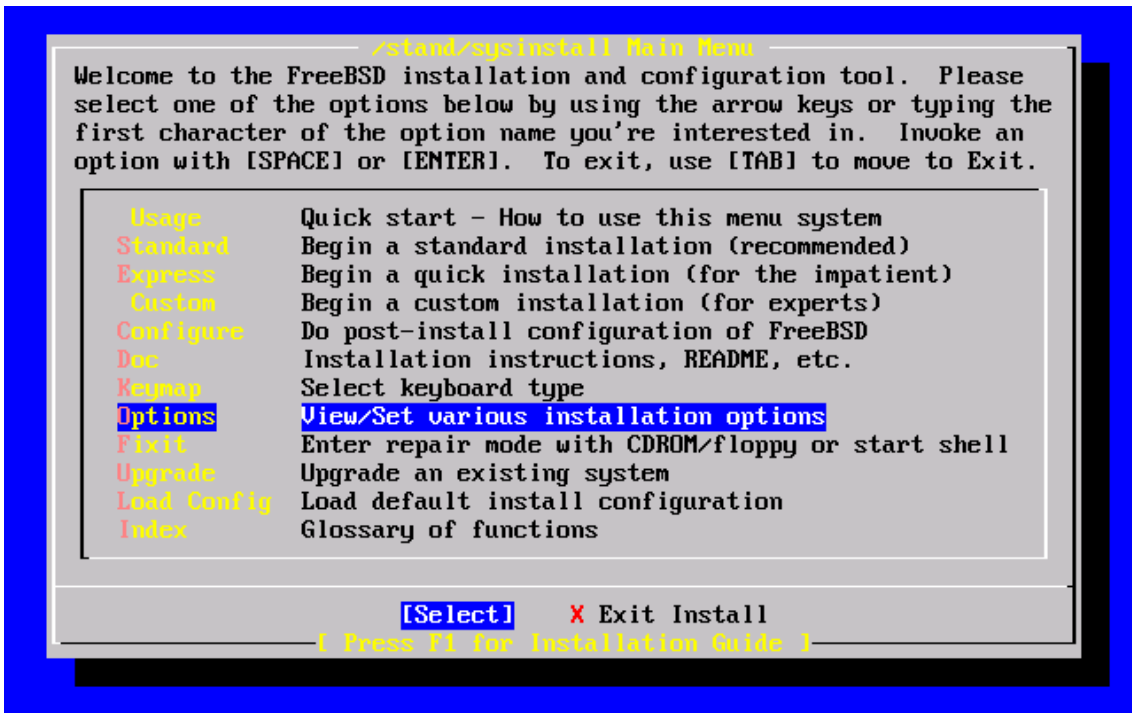
그림 2-12. Sysinstall Keymap 메뉴



### 2.4.3 설치 옵션 화면

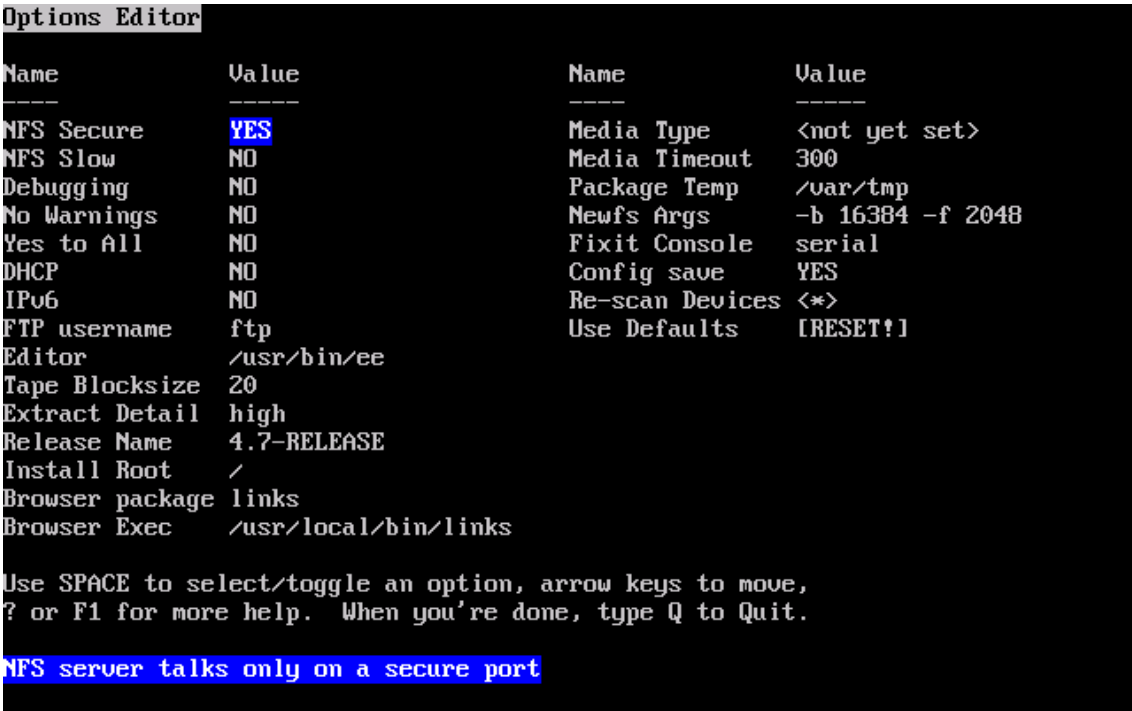
Options 을 선택하고 Enter 누른다.

그림 2-13. Sysinstall 메인 메뉴



보통 대부분의 유저에게 기본 설치 옵션이 적당하므로 변경할 필요는 없다. 릴리즈 이름은 설치 중인 버전에 따라 바뀐다.

그림 2-14. Sysinstall 옵션



---

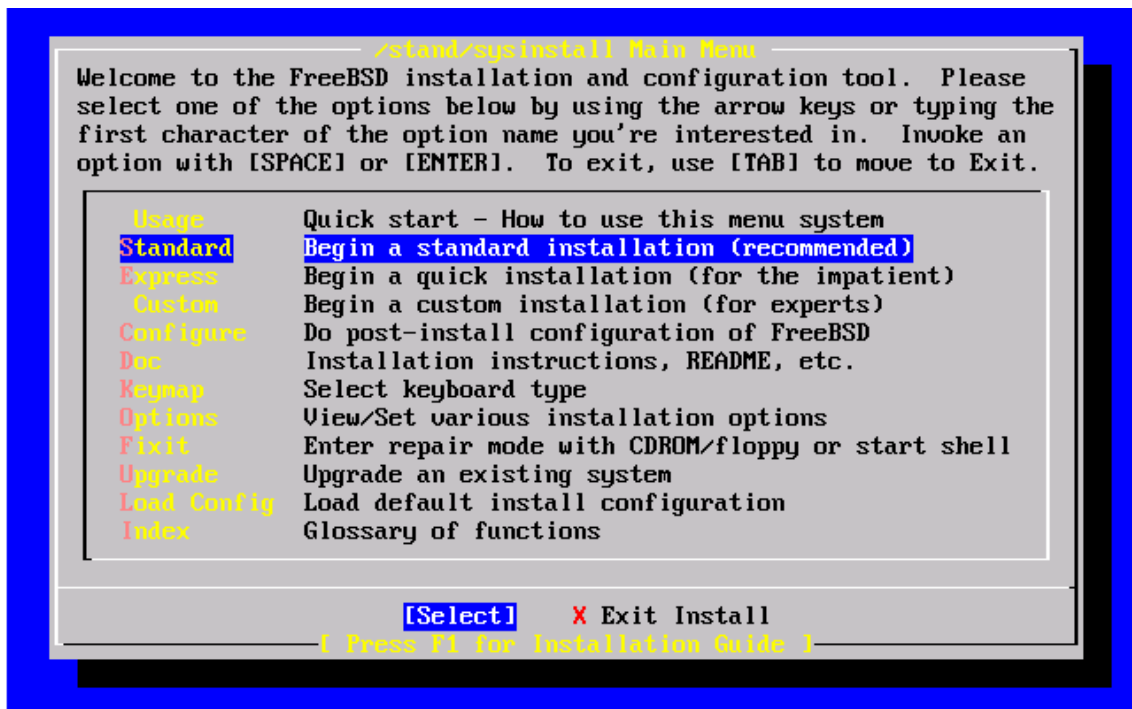
선택된 아이템 설명은 화면 하단의 파란색으로 빛나는 곳에 나타난다. 옵션 중 하나는 모든 값을 초기화해서 기본값으로 시작하게 하는 **Use Defaults** 다.

F1 키를 눌러 다양한 옵션에 대한 도움말 화면을 읽을 수 있다. Q를 누르면 메인 설치 메뉴로 되돌아간다.

## 2.4.4 Standard 설치 시작

Standard 설치는 UNIX 나 FreeBSD 가 처음인 유저에게 권장되는 옵션이다. 방향키로 Standard 를 선택하고 Enter 를 눌러서 설치를 시작한다.

그림 2-15. Standard 설치 시작



---

## 2.5 Disk 공간 할당

첫 번째 과업은 FreeBSD 에 디스크 공간을 할당하고 그 공간에 라벨을 붙여 **sysinstall** 이 설치 준비를 할수 있다. 이 작업을 진행하려면 FreeBSD 가 원하는 정보를 디스크에서 어떻게 찾는지 알고 있어야 한다.

### 2.5.1 BIOS 드라이브 번호 표기법

시스템에 FreeBSD 를 설치하고 설정하기 전에 특히 여러개의 하드 드라이브를 가지고 있다면 알고 있어야 할 중요한 정보가 있다.

PC 에서 MS-DOS 나 윈도우처럼 BIOS 에 의존적인 운영체제를 사용한다면 BIOS 에서 논리적으로 디스크의 순서를 바꿀 수 있고 운영체제는 이 변화를 따른다. 이 방법으로 사용자는 “Primary master”라고 부르는 드라이브가 아닌 다른 디스크 드라이브로 부팅할 수 있다. 이 방법은 단순하고 저렴하게 시스템을 백업하기 위해 **Ghost** 나 **XCOPY** 로 첫 번째 디스크를 똑 같은 두 번째 하드 드라이브에 복사하는 사용자에게 특히 편리하다. 바이러스 감염이나 운영체제 결함으로 첫 번째 드라이브가 실패한다면 BIOS 에서 논리적으로 두 드라이브를 바꿔서 시스템을 쉽게 복구할 수 있다. 이 방법은 드라이브의 케이블을 교환하는것과 비슷하지만 컴퓨터 케이스를 열 필요가 없다.

SCSI 컨트롤러가있는 더 비싼 시스템에서는 비슷한 방식으로 7 개까지 SCSI 드라이브를 재정렬할 수 있는 BIOS 확장을 포함하는 경우가 많다.

이런 기능의 장점에 익숙해진 사용자는 FreeBSD 를 사용한 결과가 기대한 것과 다른것에 놀라게 될지도 모른다. FreeBSD 는 BIOS 를 사용하지 않기 때문에 “BIOS 의 논리적인 드라이브 매핑”을 알지 못한다. 이것은 매우 난처한 상황을 야기할 수 있는데 특히 드라이브들이 물리적으로 동일한 결합구조로(geometry) 되어있고 서로가 다른 드라이브의 복사본 일 경우에 더욱 심각할 것이다. FreeBSD 를 사용할 때는 FreeBSD 설치전의 BIOS 드라이브 순서로 돌려 놓아야되고 이 후에도 순서를 그대로 두어야 한다. 드라이브 순서를 바꾸려면 컴퓨터의 케이스를 열어서 점퍼와 케이블을 바꿔야 한다.

Bill 과 Fred 의 예외적인 사건 파일에서 실례:

---

Bill 은 Fred 에게 다른 FreeBSD 박스를 만들어 주기위해 오래된 윈도우 박스를 지웠다. Bill 은 SCSI 유닛 0 인 SCSI 드라이브를 하나 설치하고 여기에 FreeBSD 를 설치했다.

Fred 는 시스템을 사용하기 시작하고 며칠 후 오래된 SCSI 드라이브에 다수의 소프트 에러가 있다는 메시지를 받고 이 사실을 Bill 에게 보고했다.

며칠이 더 지난 후 Bill 은 이 상황을 해결할 시간이라고 판단해서 디스크 드라이브 보관소에서 동일한 SCSI 드라이브를 가져왔다. 초기 표면 스캔 결과 드라이브가 정상적이었기 때문에 Bill 은 이 드라이브를 SCSI 유닛 4 로 설치하고 드라이브 0 에서 4 로 이미지를 복사했다. 이제 새로운 드라이브가 설치 되었고 정상적으로 작동하였기에 Bill 은 이 머신을 사용하기로 마음 먹고 SCSI BIOS 디스크 재 배치를 사용해서 시스템은 SCSI 유닛 4 에서 부팅되었다. FreeBSD 는 부팅되고 정상적으로 작동하였다.

Fred 는 며칠 동안 계속해서 일을했고 Bill 과 Fred 는 새로운 FreeBSD 버전으로 업그레이드할 시간이라고 결정했다. Bill 은 유닛 0 의 안정성이 떨어지기 때문에 보관소에서 동일한 다른 디스크로 교체했다. 그리고 Bill 은 새로운 버전의 FreeBSD 를 Fred 의 매직 인터넷 FTP 플로피를 사용하여 새로운 SCSI 유닛 0 에 설치했다. 설치는 정상적이었다.

Fred 는 새로운 버전의 FreeBSD 를 며칠 동안 사용한 후 기술부에서 사용하기에 적당하다고 생각했다. 예전 버전에서 모든 데이터를 복사 할 시간이 되었다. 그래서 Fred 는 SCSI 유닛 4 를 (예전 FreeBSD 버전의 마지막 복사본) 마운트 했다. Fred 는 SCSI 유닛 4 에 이전 데이터가 전혀 없어서 놀라웠다.

데이터는 어디로 갔을까?

Bill 이 원본 SCSI 유닛 0 을 SCSI 유닛 4 로 이미지를 복사 했을 때 유닛 4 는 “새로운 복사본” 이 되었다. Bill 이 SCSI BIOS 를 재정렬했기 때문에 SCSI 유닛 4 에서 부팅할 수 있었고 이 때 그는 혼자만의 바보짓을 한 것이다. FreeBSD 는 이 때까지 SCSI 유닛 0 에서 운용되었다. 이런 종류의 BIOS 변경은 Boot 와 Loader 코드 몇 가지 또는 전체를 BIOS 에서 선택한 드라이브로 변경하지만 FreeBSD 커널이 로드되면 BIOS 드라이브 번호를 무시하고 FreeBSD 는 일반적인 드라이브 번호로 되돌린다. 이 실례에서 시스템은 계속해서 최초의 SCSI 유닛 0 에서 운용되었고 Fred 의 모든 데이터는 SCSI 유닛 4 가 아닌 0 에 있었다. 시스템이 SCSI 유닛 4 에서 동작하는 것으로 나타났던 것은 단순히 인간의 인위적인 기대감이었다.

어쩌든 우리는 이 현상을 발견하여 어떤 데이터도 삭제되거나 피해를 입지 않아서 기쁘다. 예전

---

SCSI 유닛 0 이 복구되어 Fred 의 모든 데이터가 되돌려 졌다.

이 실례에 SCSI 드라이브가 사용되었다라도 IDE 드라이브의 개념도 동일하다.

## 2.5.2 Fdisk 로 슬라이스 생성

**Note:** 이 부분에서 디스크에 쓰기 동작을 하기 때문에 아무것도 변경하지 않는다. 실수 한 것 같아서 다시 시작하려면 메뉴에서 **sysinstall** 을 빠져 나가서 다시 시작하거나 **U** 를 눌러 되돌리기 옵션을 사용한다. 빠져 나가는것을 모를 경우 그냥 컴퓨터를 꺼도 된다.

**sysinstall** 에서 standard 설치 시작을 선택하면 다음과 같은 메시지를 볼 수 있다:

Message

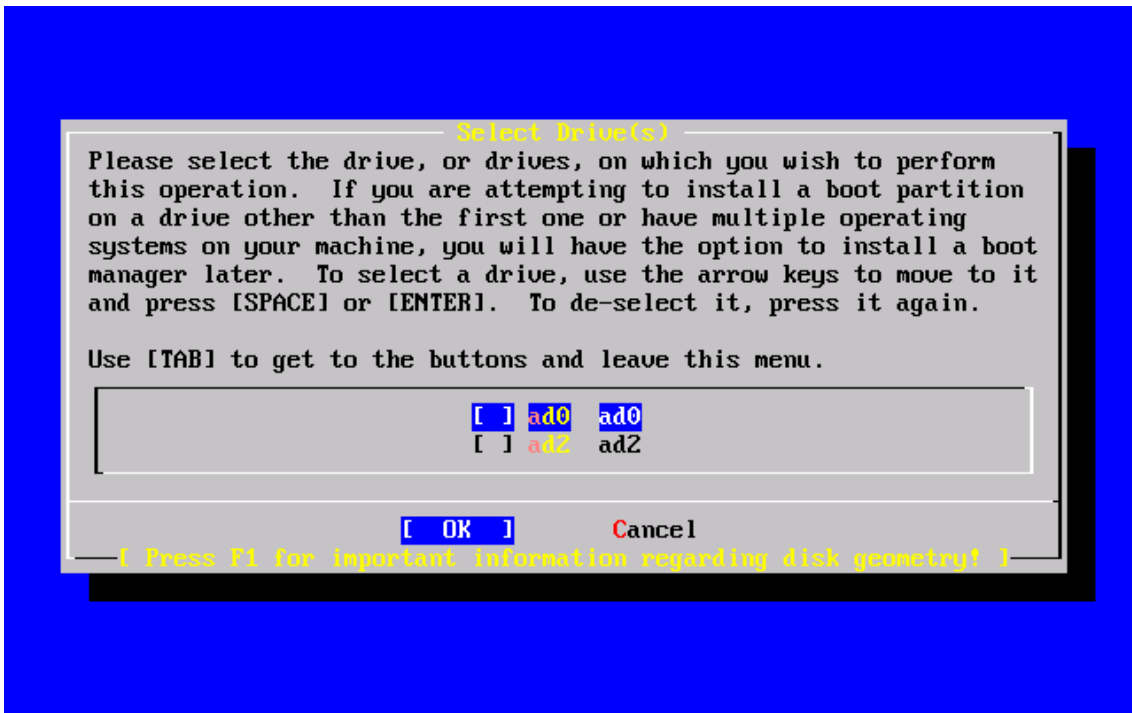
In the next menu, you will need to set up a DOS-style ("fdisk") partitioning scheme for your hard disk. If you simply wish to devote all disk space to FreeBSD (overwriting anything else that might be on the disk(s) selected) then use the (A)ll command to select the default partitioning scheme followed by a (Q)uit. If you wish to allocate only free space to FreeBSD, move to a partition marked "unused" and use the (C)reate command.

[ OK ]

[ Press enter or space ]

지시하는대로 **Enter** 를 누른다. 장치를 탐색할때 커널이 찾은 모든 하드 드라이브 리스트를 보게된다. 그림 2-16 의 예제는 두 개의 IDE 디스크 ad0 와 ad2 를 가지고있는 시스템을 보여준다.

그림 2-16. Fdisk 로 원하는 디스크 선택



이곳에 왜 ad1 이 없는지 이상하게 생각할 것이다. 왜 사라졌는지?

두 개의 IDE 하드디스크를 가지고 있다면 어떤 일이 발생하는지 생각해 보라, 하나는 첫 번째 IDE 컨트롤러의 마스터에 그리고 다른 하나는 두 번째 IDE 컨트롤러의 마스터에 있다. FreeBSD 가 찾은 순서대로 ad0 와 ad1 으로 번호를 지정했다면 모든것이 정상적으로 동작한다.

그러나 첫 번째 IDE 컨트롤러 슬레이브 장치에 세 번째 디스크를 추가했다면 이 디스크는 이제 ad1 이 되고 기존의 ad1 은 ad2 가 된다. 왜냐하면 장치 이름은(ad1s1a 처럼) 파일 시스템을 찾을때 사용하기 때문에 갑자기 어떤 파일 시스템이 더 이상 나타나지 않는것을 발견하고 FreeBSD 설정을 변경해야 될 것이다.

이런 이유로 커널은 이들 디스크가 검색된 순서가 아니고 존재하는 위치에 따라 IDE 디스크 이름을 설정할 수 있다. 그래서 두 번째 IDE 컨트롤러의 마스터 디스크는 ad0 와 ad1 장치가 없는 경우에도 항상 ad2 가 된다.

이 설정이 FreeBSD 커널에 기본값이기 때문에 앞의 화면은 ad0 와 ad2 를 보여준다. 이 스크린샷의 머신은 IDE 컨트롤러의 마스터 채널 양쪽에 IDE 디스크가 있고 슬레이브 쪽에는 디스크가 없다.



---

FreeBSD 설치에 사용하려는 디스크를 선택하고 [OK]를 누른다. Fdisk 는 그림 2-17 과 비슷한 화면으로 시작된다.

Fdisk 화면은 세 개의 섹션으로 나누어져 있다.

화면에서 처음 두 라인이되는 첫 번째 섹션은 FreeBSD 이름, 디스크 geometry, 디스크 크기를 포함하여 현재 선택한 디스크의 자세한 내용을 보여준다.

두 번째 섹션은 디스크의 시작과 끝, 크기, FreeBSD 가 지정한 이름 그리고 설명과 subtype 으로 디스크에 있는 슬라이스를 보여준다. 이 예제는 PC 의 디스크 레이아웃에 사용할 두 개의 작고 사용하지않는 슬라이스를 보여준다. 또한 DOS/윈도우의 C: 드라이브와 비슷한 하나의 큰 FAT 슬라이스와 DOS/윈도우의 다른 드라이브 문자를 가지고있는 확장 슬라이스를 보여준다.

세 번째 섹션은 Fdisk 에서 이용가능한 명령을 보여준다.

그림 2-17. 편집하기 전의 일반적인 Fdisk 파티션

```
Disk name:      ad0                      FDISK Partition Editor
DISK Geometry: 16383 cyls/16 heads/63 sectors = 16514064 sectors (8063MB)

Offset          Size(ST)          End          Name PType          Desc Subtype  Flags
-----
0               63               62          -    6      unused      0
63             4193217          4193279    ad0s1  2      fat        14  >
4193280         1008            4194287    -    6      unused      0  >
4194288        12319776        16514063    ad0s2  4      extended   15  >

The following commands are supported (in upper or lower case):

A = Use Entire Disk      G = set Drive Geometry  C = Create Slice      F = `DD' mode
D = Delete Slice        Z = Toggle Size Units   S = Set Bootable     I = Wizard m.
T = Change Type         U = Undo All Changes    Q = Finish

Use F1 or ? to get more help, arrow keys to select.
```

이제 원하는 방식에 따라 디스크에 슬라이스를 만들면 된다.

전체 디스크를 FreeBSD 가 사용한다면(설치 과정에서 **sysinstall** 이 계속 진행되도록 OK 를 눌렀다면 디스크의 기존 데이터는 지워진다) **Use Entire Disk** 옵션을 사용할 수 있도록 **A** 를 누른다. 기존에 있던 슬라이스는 지워지고 **unused** 플래그가 지정되어(다시 디스크 레이아웃을 구성할 수 있게) FreeBSD 가 사용할 하나의 큰 슬라이스가 만들어진다. 이렇게 했다면 방향키로 새로 만든 FreeBSD 슬라이스를 선택하고 **S** 를 눌러서 이 슬라이스로 부팅할 수 있도록 표시한다. 이 화면은 아래의 그림과 매우 비슷할 것이다.

이 슬라이스가 active 된 상태이고 부팅할 수 있는 상태라는것을 보여주는 Flag 칼럼의 **A** 를 확인한다.

FreeBSD 에 공간을 할당하기위해 슬라이스를 지우려면 방향키로 슬라이스를 선택하고 **D** 를 누른다. **C** 를 누르면 슬라이스 크기를 묻는 프롬프트가 나타난다. 적당한 숫자를 입력하고 **Enter** 를 누른다.

FreeBSD 를 위한 공간을 만들었다면(**Partition Magic** 같은 툴을 사용하여) **C** 를 눌러서 새로운 슬라이스를 만들 수 있다. 생성하려는 슬라이스 크기를 묻는 프롬프트가 다시 나타난다.

그림 2-18. 디스크 전체를 사용하는 Fdisk 파티션

```

Disk name:      ad0                      FDISK Partition Editor
DISK Geometry: 16383 cyls/16 heads/63 sectors = 16514064 sectors (8063MB)

Offset      Size(ST)      End      Name  PType      Desc  Subtype  Flags
-----
0           63           62      -     6          unused  0
63      16514001    16514063  ad0s1  3          freebsd 165      CA

The following commands are supported (in upper or lower case):

A = Use Entire Disk      G = set Drive Geometry  C = Create Slice      F = `DD' mode
D = Delete Slice        Z = Toggle Size Units   S = Set Bootable     I = Wizard m.
T = Change Type         U = Undo All Changes    Q = Finish

Use F1 or ? to get more help, arrow keys to select.

```

---

파티션 생성이 끝나면 Q를 누른다. 변경한 사항은 **sysinstall**에 저장되지만 아직 디스크에는 적용되지 않는다.

### 2.5.3 부트 매니저 설치

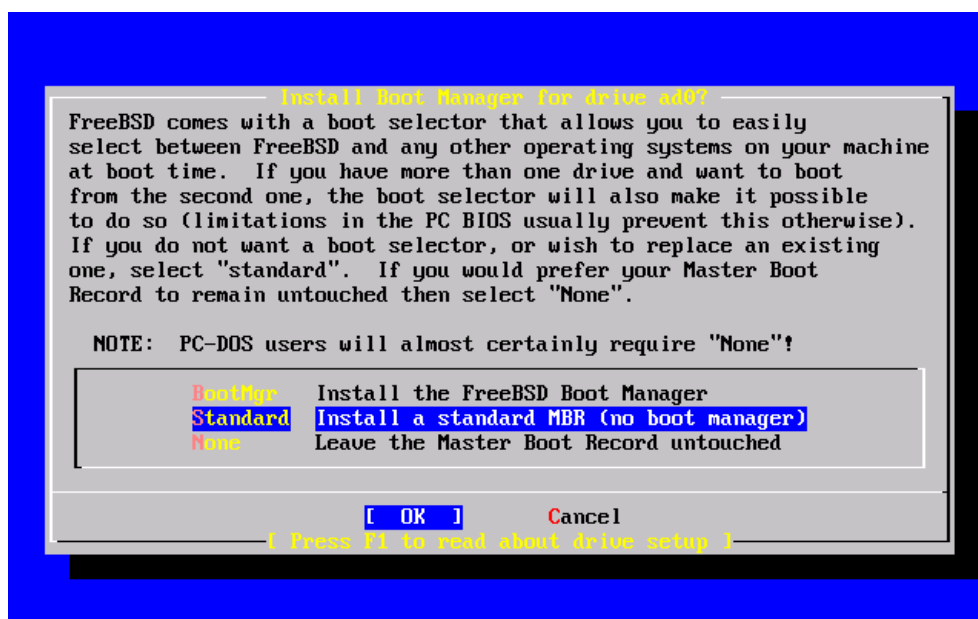
이제 부트 매니저 설치 옵션에 들어간다. 보통 FreeBSD 부트 매니저 설치는 다음과 같은 상황에 맞추어 선택할 수 있다:

- 하나 이상의 드라이브가 있고 FreeBSD를 첫 번째 드라이브에 설치한 경우.
- 같은 디스크에 다른 운영체제와 FreeBSD를 설치했고 컴퓨터가 시작될 때 FreeBSD나 다른 운영체제를 선택해서 시작하고 싶을 때.

FreeBSD가 머신의 유일한 운영체제고 첫 번째 하드 디스크에 설치했다면 표준 부트 매니저가 적당하다. FreeBSD를 부팅할 수 있는 다른 부트 매니저를 사용한다면 **None**을 선택한다.

원하는 옵션을 선택하고 **Enter**를 누른다.

그림 2-19. Sysinstall 부트 매니저 메뉴



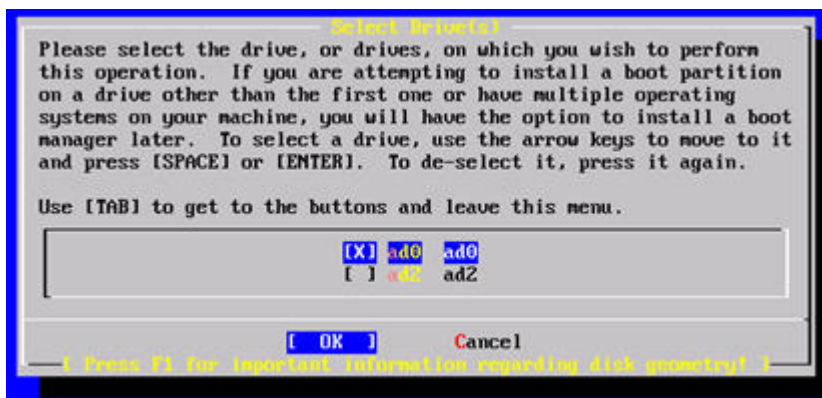
---

F1 을 눌러서 볼수 있는 도움말 화면은 두 운영체제가 하드 디스크를 공유하면 발생할 수 있는 문제에 대해 설명한다.

## 2.5.4 다른 드라이브에 슬라이스 생성

한 개 이상의 드라이브가 있다면 부트 매니저를 선택한 후 드라이브 선택 화면으로 되돌아 온다. 한 개 이상의 디스크에 FreeBSD 를 설치하려면 여기서 다른 디스크를 선택하고 Fdisk 로 슬라이스 생성 과정을 반복한다.

그림 2-20. 드라이브 선택 화면에서 나간다



Tab 키로 선택한 마지막 드라이브에서 [OK]와 [Cancel]을 변경할 수 있다.

Tab 키를 한번 눌러서 [OK]로 변경하고 Enter 를 눌러 설치를 계속한다.

## 2.5.5 Disklabel 을 이용한 파티션 생성

방금 생성한 각 슬라이스에 몇 개의 파티션을 만들어야 된다. 각 파티션은 a 부터 h 에 이르는 문자이고 파티션 b, c 와 d 는 지켜야되는 전통적인 의미가있다.

특정 어플리케이션은 한 개 이상의 디스크에 파티션을 생성했을때 얻을 수 있는 장점이 있

다. 그렇지만 이 장점을 위해 FreeBSD 를 처음 설치할 때부터 너무 많은 디스크 파티션을 생성할 필요는 없다. FreeBSD 를 설치하고 어떻게 사용하는지 배우기 시작했다는것이 더욱 중요하다. 운영체제와 더 친숙해졌을때 파티션 구성을 변경하기위해 FreeBSD 를 재 설치할 수 있다.

이번 구성은 4 개의 파티션을 만든다. 하나의 스왑 공간과 3 개의 파일시스템을 만든다.

표 2-2. 첫 번째 디스크의 파티션 레이아웃

파티션	파일 시스템	크기	설명
A	/	100MB	이것은 root 파일시스템이다. 다른 파일시스템들은 이 파일시스템 밑에 마운트된다. 100MB 가 이 파일시스템에 적당하다. 일반적으로 FreeBSD 를 설치한다면 이곳에 대략 40MB 정도의 데이터가 쌓이기 때문에 너무 많은 데이터를 두지 않는다. 남은 공간은 임시 데이터를 위해 필요하고 다음 FreeBSD 버전은 더 많은 / 공간을 필요로하기 때문에 여유있게 둔다.
B	N/A	2-3xRAM	시스템의 스왑 공간을 이 파티션에 있다. 적당한 양의 스왑공간을 선택하는 것은 예술에 가깝다. 스왑 공간을 만드는 가장 좋은 방법은 물리적인 메모리의(RAM) 두 배나 세배 정도의 공간을 할당한다. 그리고 최소한 64MB 이상의 공간을 할당해야 되기 때문에 컴퓨터에 32MB 보다 적은 RAM 이 있더라도 스왑의 양을 64MB 로 설정한다. 한 개 이상의 디스크를 가지고 있다면 각 디스크별로 공간을 할당할 수 있다. FreeBSD 는 각 디스크에서 스왑을 사용할 경우 효율적이다. 이 경우 필요한 스왑 양을(예: 128MB) 계산해서 가지고 있는 전체 디스크 수로 나누어 각 디스크 별로 공간을 할당하고, 이 예제에서는 디스크 별로 64MB 씩 할당한다.
E	/var	/50MB	/var 디렉터리는 계속해서 변하는 로그 파일과 다른 관리적인 파일을 가지고있다. 이들 대부분의 파일은 FreeBSD 가 하루하루 운용되는 동안 광범위하게 읽혀지거나 쓰여진다. 이런 파일을 다른 파일시스템과 따로 두면 동일한 사용 패턴을 가지고 있지않은 다른 디렉터리 파일에 영향을 주지않고 FreeBSD 가 이런 파일의 사용을 최적화할 수 있다.

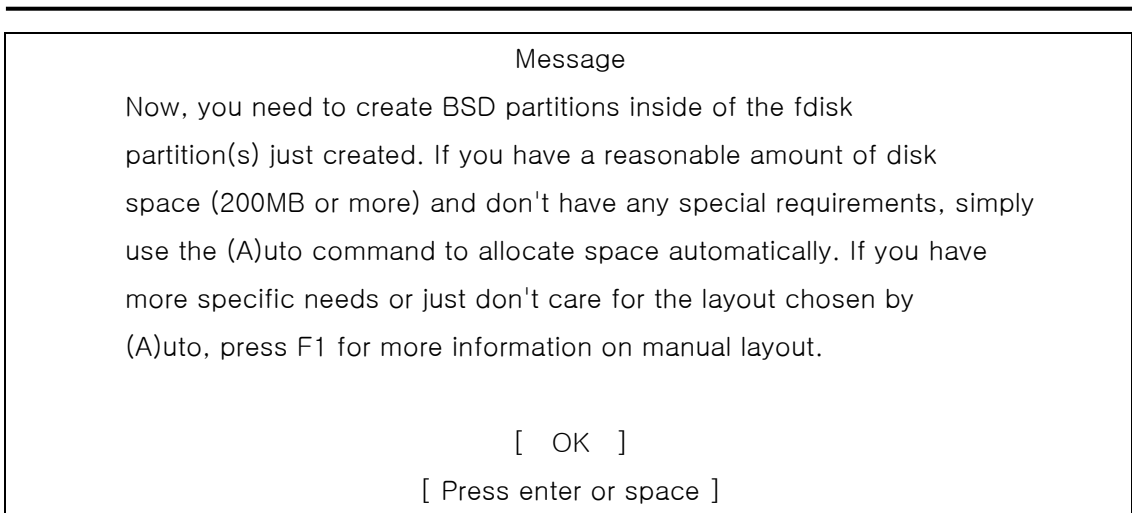
<i>F</i>	/usr	나머지 디스크	다른 모든 파일은 보통 /usr 과 /usr 서브 디렉터리에 저장된다
----------	------	------------	---

한 개 이상의 디스크에 FreeBSD 를 설치하려면 설정한 다른 슬라이스에도 파티션을 만들어야한다. 가장 쉬운 방법은 각 디스크에 하나는 스왑 공간을 그리고 다른 하나는 파일시스템으로 두 개의 파티션을 만든다.

표 2-3. 연속된 디스크의 파티션 레이아웃

파티션	파일시스템	크기	설명
<i>B</i>	N/A	설명을 본다	이미 설명했듯이 각 디스크에 스왑 공간을 나눌 수 있다. a 파티션에 여유 공간이 있더라도 관례적으로 스왑 공간은 b 파티션에 둔다.
<i>e</i>	/diskn	나머지 디스크	디스크의 나머지는 하나의 커다란 파티션으로 둔다. 이것은 e 파티션 대신 쉽게 a 파티션에 둘 수 있다. 그러나 관례적으로 슬라이스의 a 파티션은 root (/) 파일시스템으로 남겨 둔다. 이러한 관례를 꼭 따라야 되는 것은 아니지만 <b>sysinstall</b> 은 관례를 따르기 때문에 명료한 설치를 위해 이러한 관례를 따르도록한다. 이 파일시스템은 어느곳에든 마운트할 수 있다; 이 예제에서는 이 파일시스템을 /diskn에 마운트하도록 권장하고, <i>n</i> 은 변경한 각 디스크의 번호다. 그러나 다른 곳을 선호한다면 원하는 대로 한다.

파티션 레이아웃을 정했으면 **sysinstall** 로 레이아웃을 생성한다. 다음과 같은 메시지를 보게 된다.



Enter 를 눌러서 **Disklabel** 이라는 FreeBSD 파티션 편집기를 실행한다.

다음 그림은 **Disklabel** 을 처음 실행 했을때 화면이다. 이 화면은 새 개의 섹션으로 나누어진다.

처음 몇라인은 현재 작업하고있는 디스크의 이름과 파티션이 생성될 슬라이스를 보여준다 (여기서 **Disklabel** 은 슬라이스 이름이라고 하지않고 파티션 이름이라고 부른다). 그리고 이 화면에서 슬라이스의 여유 공간도 볼 수 있다. 이 공간은 슬라이스 안에 설정되어 있지만 아직 파티션으로 할당하지 않았다.

화면의 중간 부분은 생성된 파티션, 각 파티션이있는 파일시스템의 이름과 크기 그리고 파일시스템을 생성할때 적용한 몇 가지 옵션을 보여준다.

화면의 마지막 부분은 **Disklabel** 에서 사용할 수 있는 키 조합을 보여준다.

그림 2-21. Sysinstall 의 Disklabel 편집기

```

FreeBSD Disklabel Editor
Disk: ad0 Partition name: ad0s1 Free: 16514001 blocks (8063MB)
Part      Mount      Size Newfs  Part      Mount      Size Newfs
-----
-----

The following commands are valid here (upper or lower case):
C = Create      D = Delete    M = Mount pt.
N = Newfs Opts  Q = Finish    S = Toggle SoftUpdates
T = Toggle Newfs U = Undo      A = Auto Defaults  R = Delete+Merge

Use F1 or ? to get more help, arrow keys to select.

```

Disklabel 은 자동으로 파티션을 생성하고 기본 크기를 할당할 수 있다. 이제 **A** 를 눌러서 시도해 보자. 아래 그림과 비슷한 화면을 보게될 것이다. 디스크 크기에 따라 **default** 를 사용하는 것이 적합하거나 그렇지 않을 것이다. **default** 를 사용하지 않아도 문제는없다.

**Note:** FreeBSD 4.5 가 시작되면서 기본 파티션은 / 파티션의 일부를 /tmp 로 사용하지 않고 /tmp 파티션을 따로 할당한다. 이 기능으로 / 파티션이 임시 파일로 가득 차는것을 피할 수 있게되었다.

그림 2-22. 자동 기본값의 Sysinstall Disklabel 편집기

```

FreeBSD Disklabel Editor
Disk: ad0 Partition name: ad0s1 Free: 0 blocks (0MB)
Part      Mount      Size Newfs  Part      Mount      Size Newfs
-----
-----
ad0s1a    /           128MB UFS     Y
ad0s1b    swap        503MB SWAP
ad0s1e    /var        256MB UFS+S Y
ad0s1f    /tmp        256MB UFS+S Y
ad0s1g    /usr        6919MB UFS+S Y

The following commands are valid here (upper or lower case):
C = Create      D = Delete    M = Mount pt.  W = Write
N = Newfs Opts  Q = Finish    S = Toggle SoftUpdates
T = Toggle Newfs U = Undo      A = Auto Defaults  R = Delete+Merge

Use F1 or ? to get more help, arrow keys to select.

```



---

제시된 파티션을 삭제하고 원하는 파티션을 만들려면 방향키로 첫 번째 파티션을 선택하고 **D**를 눌러서 삭제한다. 제시된 모든 파티션을 삭제할 때까지 반복한다.

첫 번째 파티션을 생성하려면(*a, /*에 마운트) 화면 상단의 디스크 정보를 선택하고 **C**를 누른다. 새로운 파티션의 크기를 묻기 위해 Dialog 박스가 나타난다(다음 그림과 같이). 원하는 디스크 블록의 크기를 입력하거나 메가바이트는 *M*, 기가바이트는 *G*를 입력하거나 실린더는 *C*를 입력한다.

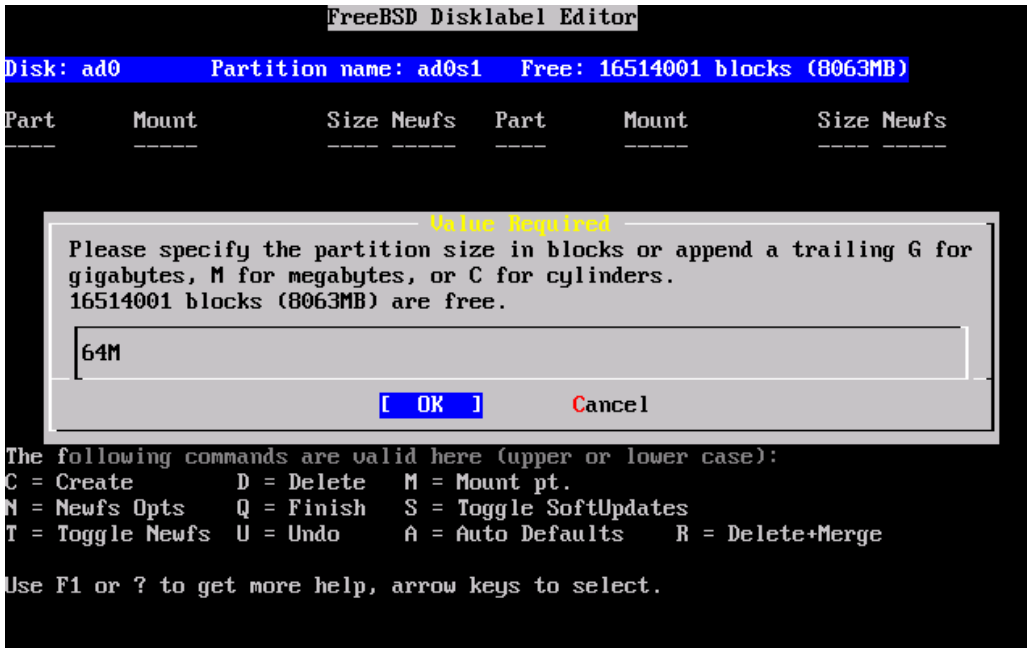
**Note:** FreeBSD 5.X가 시작되면서 유저는 *Custom Newfs (Z)* 옵션을 사용하여 UFS2를 선택할 수 있다. *Auto Defaults*로 라벨을 생성하고 *Custom Newfs* 옵션으로 수정하거나 일반적인 파티션을 생성할 때 *-O 2*를 추가한다. *Custom Newfs* 옵션을 사용한다면 *SoftUpdates*에 *-U*를 추가하는것도 잊지 않는다.

그림 2-23. root 파티션을 위한 빈 공간



보여준 기본 크기는 슬라이스의 남은 공간을 사용하는 파티션을 만든다. 이미 설명한 것처럼 파티션 크기를 사용한다면 **Backspace**로 지정된 숫자를 지우고 다음 그림처럼 64M를 입력한다. 그리고 **[OK]** 버튼을 누른다.

그림 2-24. root 파티션 크기 수정



선택한 파티션의 크기를 지정하면 이 파티션이 파일시스템을 포함할 것인지 스왑 공간인지 묻는 다음과 같은 화면이 나타난다. 첫 번째 파티션은 파일시스템을 포함하기 때문에 FS 를 선택하고 Enter 를 누른다.

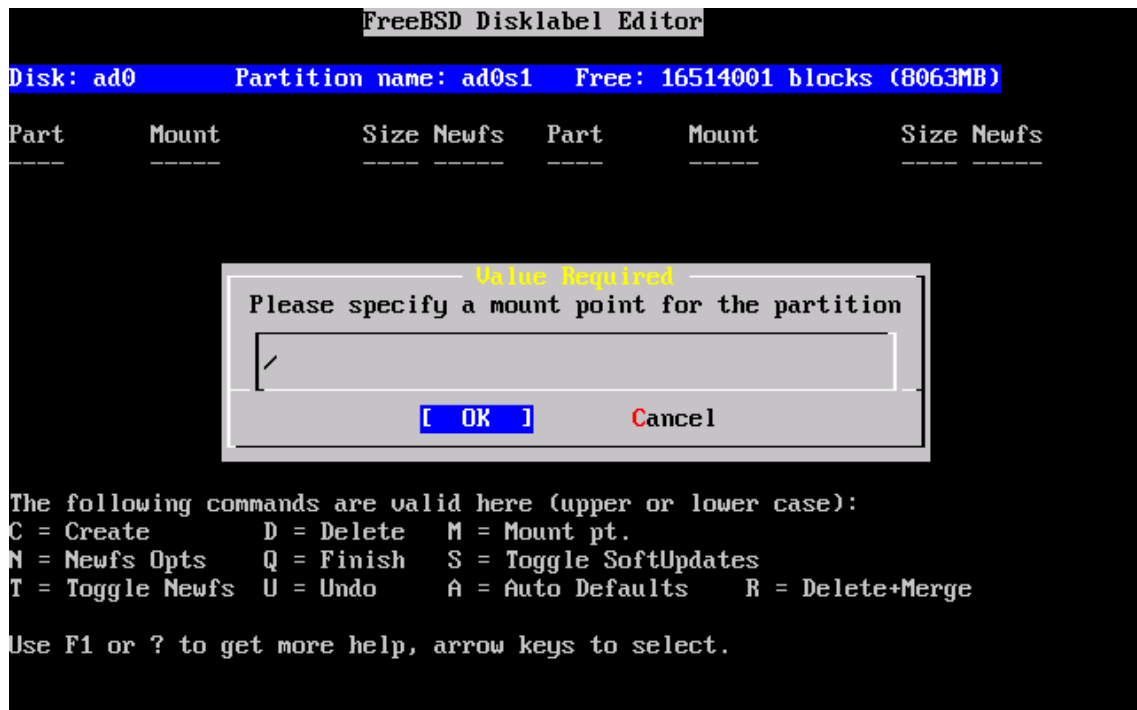
그림 2-25. root 파티션 타입 선택



---

마지막으로 파일시스템을 생성하는 것이므로 **Disklabel** 에게 어디에 마운트 할 것인지 다음과 같이 지정해야 된다. root 파일시스템의 마운트 위치는 /이기 때문에 /라고 입력한 후 **Enter** 를 누른다.

그림 2-26. root 마운트 위치 선택



새로 만든 파티션을 보여주기위해 화면이 업데이트 된다. 다른 파티션을 생성하도록 이 과정을 반복한다. 스왑 파티션을 생성하면 스왑 파티션은 마운트하지 않으므로 마운트 위치가 표시되지 않는다. 마지막 파티션 /usr 을 생성할때 제시된 크기를 지우고 남은 슬라이스의 전체를 사용한다.

선택한 값은 다르더라도 마지막 **FreeBSD Disklabel** 에디터는 다음 그림과 비슷한 화면으로 나타난다. 끝내기 위해 **Q** 를 누른다.

그림 2-27. Sysinstall Disklabel 에디터

```

FreeBSD Disklabel Editor
Disk: ad0 Partition name: ad0s1 Free: 0 blocks (0MB)
Part      Mount      Size Newfs  Part      Mount      Size Newfs
-----
ad0s1a    /           64MB UFS    Y
ad0s1b    swap        512MB SWAP
ad0s1e    /var        256MB UFS+S Y
ad0s1f    /usr        7231MB UFS+S Y

The following commands are valid here (upper or lower case):
C = Create      D = Delete    M = Mount pt.
N = Newfs Opts  Q = Finish    S = Toggle SoftUpdates
T = Toggle Newfs U = Undo      A = Auto Defaults  R = Delete+Merge

Use F1 or ? to get more help, arrow keys to select.

```

## 2.6 설치할 패키지 선택

### 2.6.1 배포본 선택

어떤 배포본을 설치할지 선택하는것은 시스템의 용도와 사용할 수 있는 디스크 공간에 따라 다르다. 가능한 가장 작은 설정부터 모든 설정까지 미리 옵션이 정해져있다. 여러분이 유닉스나 FreeBSD의 새로운 유저라면 이들 옵션 중 하나를 선택한다. 배포본을 원하는대로 수정하는것은 일반적으로 경험이있는 유저를 위한것이다.

배포본 옵션과 이들이 가지고 있는것이 무엇인지 관련된 다양한 정보는 **F1**을 눌러서 확인한다. 도움말을 확인한 후 **Enter**를 누르면 다시 배포본 선택 메뉴로 되돌아온다.

그래픽컬 유저 인터페이스(GUI)를 원한다면 *X*가 포함된 배포본을 선택한다. **XFree86**의 설정과 기본 데스크톱은 설치 과정 이후에 선택한다.

설치된 기본 **XFree86** 버전은 설치하려는 FreeBSD 버전에 따라 다르다. 4.6 이전 버전의 FreeBSD에는 **XFree86 3.X**가 설치되어 있다. FreeBSD 4.6 이후 버전에는 **XFree86 4.X**가

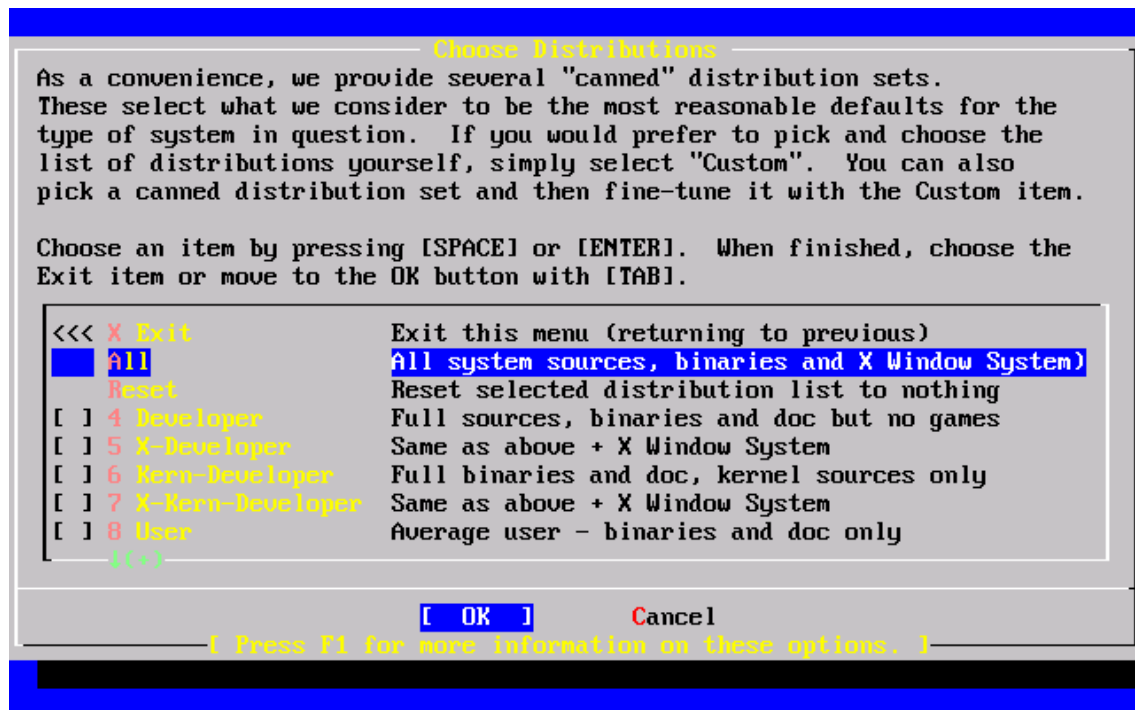
기본이다.

비디오 카드가 지원되는지 XFree86 웹 사이트에서(<http://www.xfree86.org/>) 체크한다. 설치하려는 기본 버전에서 비디오 카드가 지원되지 않는다면 설치를 끝내기 위해 X가 없는 배포본을 선택한다. 설치 후 포트 컬렉션으로 적당한 버전의 XFree86 을 설치하고 설정한다.

사용자 커널을 컴파일하려면 소스 코드를 포함하는 옵션을 선택한다. 사용자 커널을 왜 만들고 어떻게 만드는지 8 장을 본다.

물론 다목적 시스템은 모든것을 포함한다. 디스크 공간이 여유있다면 아래 그림과 같이 방향키로 All 을 선택한 후 Enter 를 누른다. 디스크 공간이 걱정된다면 상황에 맞는 옵션을 사용한다. 다른 배포본은 설치 후 추가할 수 있다.

그림 2-28. 배포본 선택



## 2.6.2 포트 컬렉션 설치

원하는 배포본을 선택한 후 FreeBSD 포트 컬렉션을 설치할 수 있는 기회가 제공된다. 포트

---

컬렉션은 소프트웨어를 설치하는 쉽고 편리한 방법으로 소프트웨어를 컴파일하는데 필요한 소스 코드는 가지고 있지않다. 이것은 자동으로 소스 코드를 다운로드하고 컴파일해서 설치하는 파일의 모음이다. 4 장에서 포트 컬렉션을 어떻게 사용하는지 설명한다.

설치 프로그램은 충분한 공간이 있다면 여유공간을 체크하지 않는다. 하드 디스크 공간이 충분하다면 이 옵션을 선택한다. FreeBSD 5.2.1 에서 FreeBSD 포트 컬렉션은 대략 300MB 의 디스크 공간이 필요하다.

User Confirmation Requested

Would you like to install the FreeBSD ports collection?

This will give you ready access to over 8,600 ported software packages, at a cost of around 210 MB of disk space when "clean" and possibly much more than that if a lot of the distribution tarballs are loaded (unless you have the extra CDs from a FreeBSD CD/DVD distribution available and can mount it on /cdrom, in which case this is far less of a problem).

The ports collection is a very valuable resource and well worth having on your /usr partition, so it is advisable to say Yes to this option.

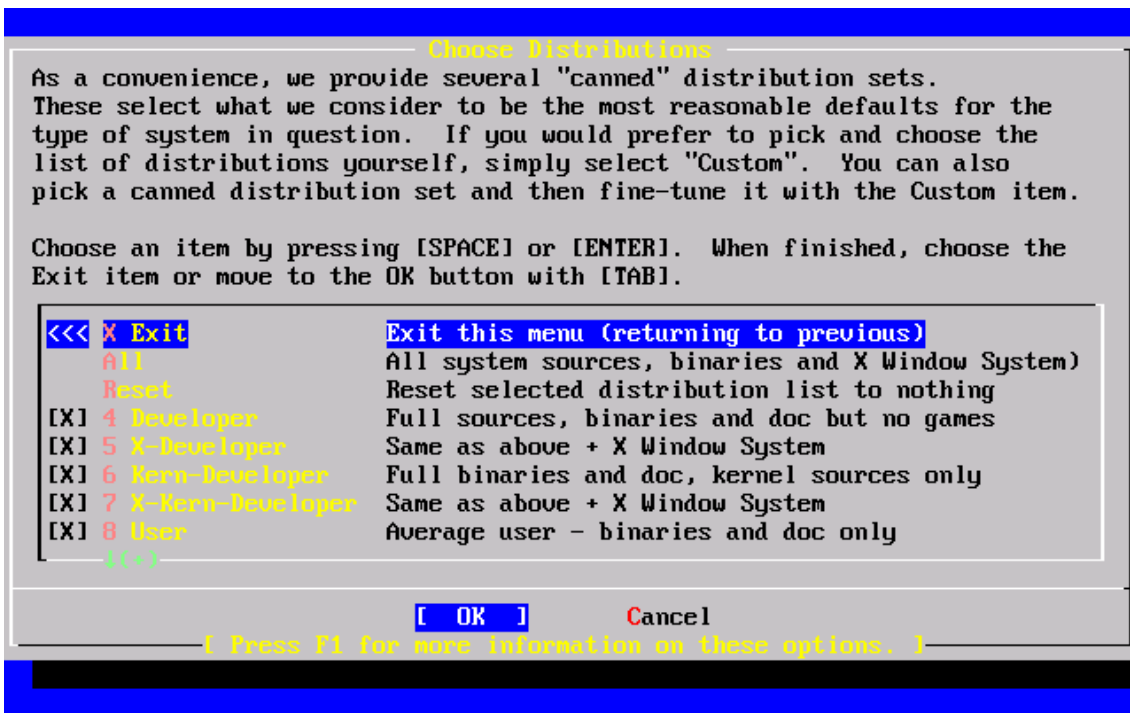
For more information on the ports collection & the latest ports, visit:

<http://www.FreeBSD.org/ports>

[ Yes ]    No

포트 컬렉션을 설치하기위해 방향키로 [Yes]를 선택하거나 [No]를 선택하여 이 옵션을 건너 뛴다. **Enter** 를 눌러 계속 진행하면 선택한 배포본 메뉴가 다시 나타난다.

그림 2-29. 배포본 확인



이 옵션이 맞다면 방향키로 Exit 를 선택한 후 [Ok]를 밝게하고 **Enter** 를 눌러 계속 진행 한다.

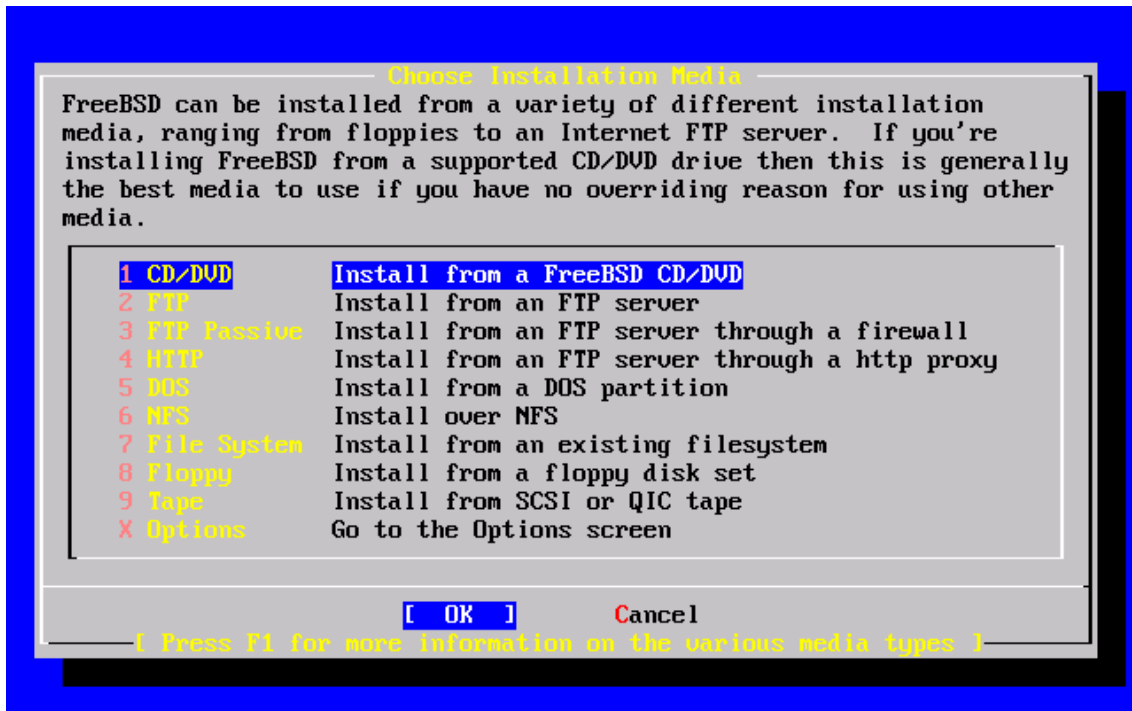
## 2.7 설치 미디어 선택

CDROM 에서 설치한다면 방향키로 FreeBSD CD/DVD 를 선택한다.

다른 설치 방법은 알맞은 옵션을 선택하고 지시를 따른다.

설치 미디어 온라인 도움말을 보려면 F1 을 누른다. **Enter** 를 눌러 미디어 선택 메뉴로 되돌아온다.

그림 2-30. 설치 미디어 선택



**FTP 설치 모드:** Active FTP, Passive FTP 또는 HTTP Proxy 를 경유하는 세가지 방법의 FTP 설치 모드 중 하나를 선택할 수 있다.

**FTP Active:** FTP 서버에서 설치

이 옵션은 모든 FTP 전송에 "Active" 모드를 사용한다. 이 모드는 방화벽을 통해서 동작하지 않지만 Passive 모드를 지원하지 않는 오래된 FTP 서버에서는 동작한다. Passive 모드에서 반응이 없다면(기본적인) active 모드로 시도한다.

**FTP Passive:** 방화벽을 거쳐 FTP 서버에서 설치

이 옵션은 모든 FTP 를 "Passive" 모드로 사용하도록 **sysinstall** 에게 지시한다. 이 모드는 랜덤 TCP 포트 주소로 입력되는 연결을 허용하지 않는 방화벽을 거쳐 사용할 수 있도록 한다.

**HTTP Proxy 를 경유한 FTP:** http Proxy 를 통해 FTP 서버에서 설치



---

이 옵션은 **sysinstall** 이 모든 FTP 를 HTTP 프로토콜로 Proxy 에 연결하도록 지시한다. proxy 는 요청을 변환하여 FTP 서버로 보낸다. 이 모드는 유저가 FTP 로 방화벽을 지나는 것은 허용하지 않지만 HTTP proxy 를 제공할 때 사용된다. 이 경우 proxy 와 FTP 서버를 지정해야 된다.

proxy FTP 서버는 보통 유저 이름 뒤에 "@" 문자를 넣고 원하는 서버 이름을 지정해야 된다. proxy 서버는 실제 서버를 대신하는 서버다. 예를 들어 <ftp.FreeBSD.org>에서 설치한다면 포트 1024 의 foo.example.com proxy FTP 서버를 사용한다.

이 경우 옵션 메뉴에서 FTP 유저 이름을 *ftp@ftp.FreeBSD.org* 로 설정하고 패스워드는 이 메일 주소를 지정한다. 설치 미디어에 FTP(proxy 가 지원한다면 Passive FTP)와 URL *ftp://foo.example.com:1234/pub/FreeBSD* 를 지정 한다.

왜냐하면 *ftp://ftp.freebsd.org* 에서 */pub/FreeBSD* 는 foo.example.com 에 프록시되기 때문에 이 장비에서 설치할 수 있다(설치 요청이 있으면 ftp.FreeBSD.org 로부터 파일이 패치된다).

## 2.8 설치 확인

여러분이 원한다면 이제부터 설치가 진행된다. 그러나 원하지 않는다면 하드 디스크의 내용을 건드리지 않고 안전하게 설치를 끝낼수 있다.

<p style="text-align: center;">User Confirmation Requested</p> <p style="text-align: center;">Last Chance! Are you SURE you want to continue the installation?</p> <p style="text-align: center;">If you're running this on a disk with data you wish to save then WE STRONGLY ENCOURAGE YOU TO MAKE PROPER BACKUPS before proceeding!</p> <p style="text-align: center;">We can take no responsibility for lost disk contents!</p> <p style="text-align: center;">[ Yes ]    No</p>
--

---

[Yes]를 선택하고 **Enter** 를 눌러서 설치를 진행한다.

설치 시간은 선택한 배포본과 사용된 설치 미디어 그리고 컴퓨터에 속도에 따라 다르다. 상태를 표시하는 일련의 메시지가 나타난다.

설치는 다음 메시지가 나타나면 끝난다.

Message

Congratulations! You now have FreeBSD installed on your system.

We will now move on to the final configuration questions.  
For any option you do not wish to configure, simply select No.

If you wish to re-enter this utility after the system is up, you may  
do so by typing: /stand/sysinstall .

[ OK ]

[ Press enter to continue ]

설치 후 설정으로 넘어가기 위해 **Enter** 를 누른다.

[No]를 선택하고 **Enter** 를 누르면 설치가 중단되기 때문에 시스템에 아무런 변화가 없다. 다음과 같은 메시지가 나타난다:

Message

Installation complete with some errors. You may wish to scroll  
through the debugging messages on VTY1 with the scroll-lock feature.  
You can also choose "No" at the next prompt and go back into the  
installation menus to retry whichever operations have failed.

[ OK ]

이 메시지는 아무것도 설치되지 않았기 때문에 발생된다. **Enter** 를 누르면 설치를 빠져나가

---

기위해 메인 설치 메뉴로 되돌아간다.

## 2.9 설치 후 설정

다양한 옵션을 설정하면 FreeBSD 가 성공적으로 설치된다. 옵션은 새로운 FreeBSD 시스템 이 부팅하기 전에 다시 입력해서 설정하거나 설치 후 /stand/sysinstall 을 사용하여 **Configure** 를 선택해서 설정할 수 있다.

### 2.9.1 네트워크 장치 설정

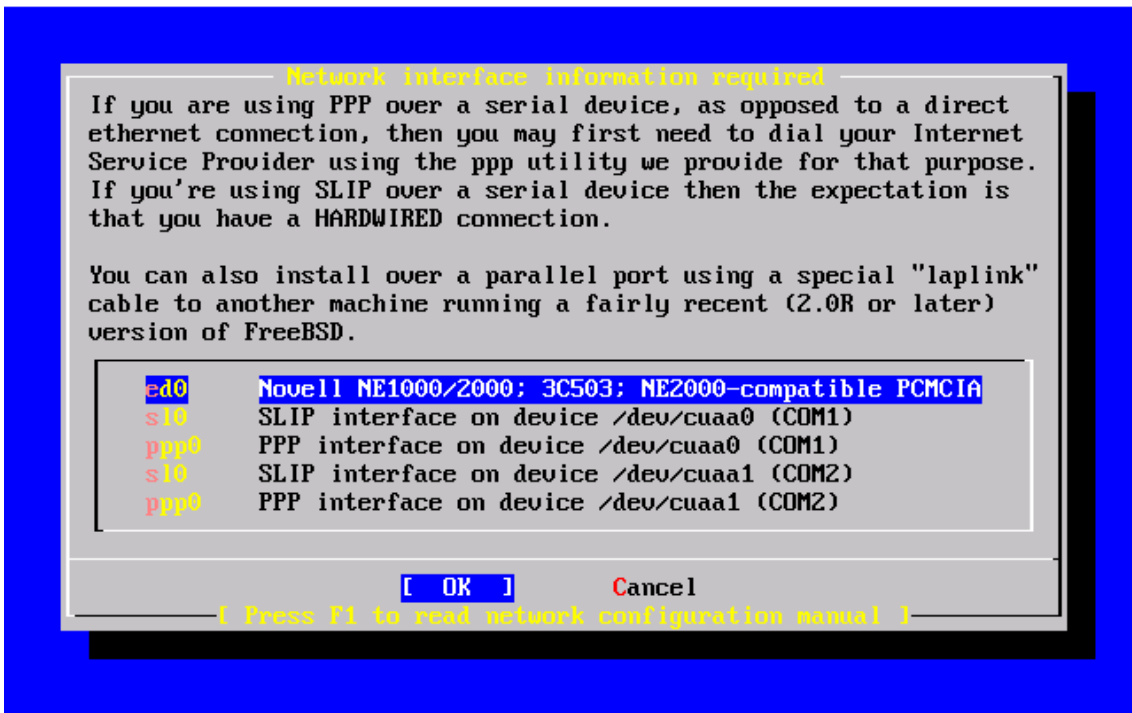
FTP 로 설치하기위해 PPP 를 설정했다면 이 화면이 나타나지않고 위에서 설명했듯이 나중에 설정할 수 있다.

로컬 네트워크의 게이트웨이/라우터로 FreeBSD 를 설정하는 자세한 정보는 고급 네트워크 섹션에서 설명한다.

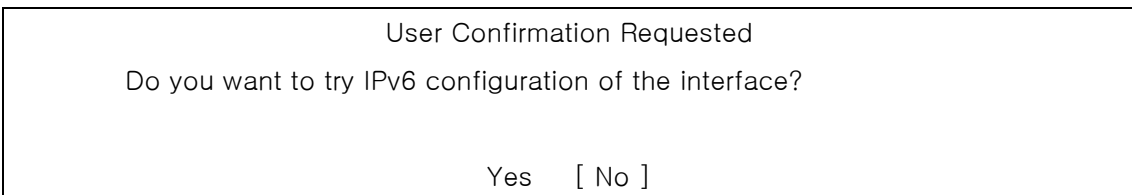
<p>User Confirmation Requested</p> <p>Would you like to configure any Ethernet or SLIP/PPP network devices?</p> <p>[ Yes ] No</p>
---

네트워크 장치를 설정하려면 [Yes]를 선택하고 **Enter** 를 누른다. 그렇지 않다면 [No]를 선택하여 계속 진행한다.

그림 2-31. 이더넷 장치 선택

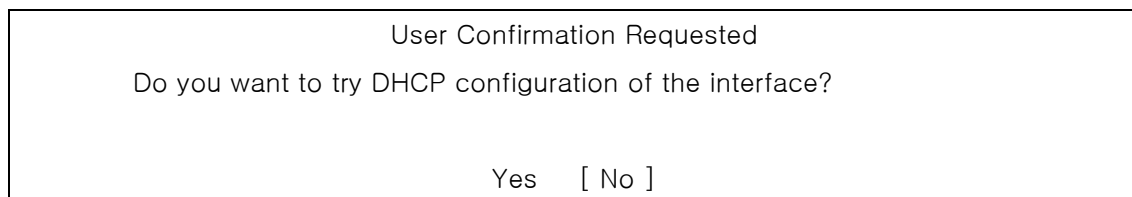


설정할 인터페이스를 방향키로 선택하고 **Enter** 를 누른다.



사실 네트워크에는 현재 인터넷 프로토콜이(IPv4) 적당하기 때문에 방향키로 [No]를 선택하고 **Enter** 를 누른다.

새로운 인터넷 프로토콜을(IPv6) 원한다면 [Yes]를 선택하고 **Enter** 를 누른다. RA 서버를 스캔하기 위해 몇 초 정도 필요하다.



DHCP(동적 호스트 구성 프로토콜)가 필요 없다면 방향키로 [No]를 선택하고 **Enter** 를 누른

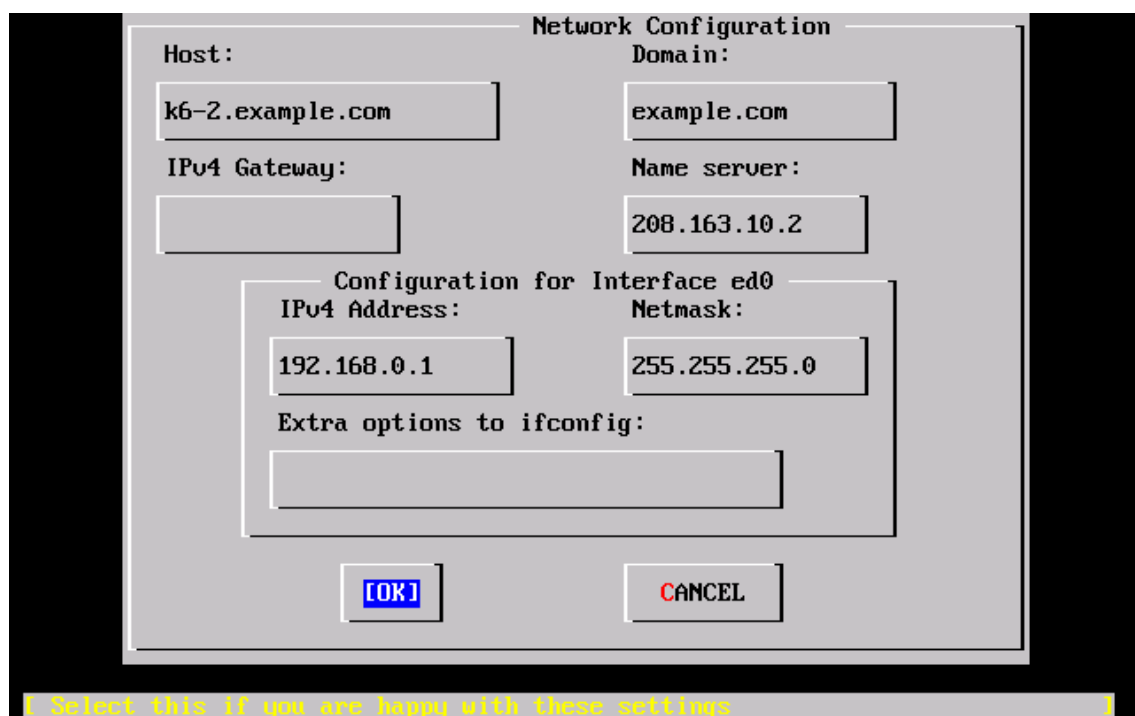
---

다.

[Yes]를 선택하면 **dhclient**가 실행되고 성공하면 네트워크 설정 정보가 자동적으로 채워진다. 더 자세한 정보는 23 장을 참고한다.

다음 네트워크 설정화면은 로컬 네트워크의 게이트웨이로 동작하는 시스템의 이더넷 장치 설정을 보여준다.

그림 2-32. ed0의 네트워크 설정



Tab 을 사용하여 원하는 정보 필드를 선택하고 알맞은 정보를 입력한다.

Host

k6-2.example.com 처럼 적절한 호스트 이름.

---

## Domain

example.com 처럼 머신이 있는곳의 도메인 이름.

## IPv4 Gateway

로컬이 아닌 외부로 패킷을 포워딩하는 호스트의 IP 주소. 머신이 네트워크의 노드라면 이곳을 채운다. 머신이 네트워크의 게이트웨이라면 이 필드를 빈 공간으로 둔다.

## Name server

로컬 DNS 서버의 IP 주소. 이 사설 네트워크에는 로컬 DNS 서버가 없기 때문에 ISP의 DNS 서버 IP 주소를 사용한다(208.163.10.2).

## IPv4 address

이 인터페이스에 사용되는 IP 주소는 192.168.0.1 이다.

## Netmask

이 로컬 네트워크에 사용되는 주소 블록은 C 클래스다(192.168.0.0-192.168.255.255). 기본 넷 마스크는 클래스 C 네트워크를 위한 것이다(255.255.255.0).

## ifconfig 추가 옵션

인터페이스에 특별한 옵션이 필요하면 ifconfig 에 옵션을 추가한다. 이 경우 아무런 옵션이 없다.

[OK]를 선택하기 위해 **Tab** 을 사용하고 끝나면 **Enter** 를 누른다.

---

User Confirmation Requested

Would you like to Bring Up the ed0 interface right now?

[ Yes ]   No

[Yes]를 선택하고 **Enter** 를 누르면 머신은 네트워크를 사용할 수 있도록 준비한다. 그러나 머신을 재 부팅해야되기 때문에 설치 도중에는 네트워크를 사용할 수 없다.

## 2.9.2 게이트웨이 설정

User Confirmation Requested

Do you want this machine to function as a network gateway?

[ Yes ]   No

머신이 로컬 네트워크의 게이트웨이 역할을해서 다른 머신 사이의 패킷을 포워딩한다면 [Yes]를 선택하고 **Enter** 를 누른다. 머신이 네트워크의 노드라면 [No]를 선택하고 **Enter** 를 눌러 계속 진행한다.

## 2.9.3 인터넷 서비스 설정

User Confirmation Requested

Do you want to configure inetd and the network services that it provides?

Yes   [ No ]

[No]를 선택했다면 **telnetd** 같은 다양한 서비스가 활성화되지 않는다. 이것은 원격지 유저가 **텔넷**으로 이 머신을 사용할 수 없음을 의미한다. 그렇지만 로컬 유저는 **텔넷**으로 원격 머신에 접근할 수 있다.

---

이런 서비스는 설치 후 /etc/inetd.conf 를 에디터로 편집하여 활성화할 수 있다.  
더 많은 정보는 23 장을 본다.

설치하는 동안 이런 서비스를 설정하고 싶다면 [Yes]를 선택한다. 추가적인 확인 화면이 나타난다.

User Confirmation Requested

The Internet Super Server (inetd) allows a number of simple Internet services to be enabled, including finger, ftp and telnetd. Enabling these services may increase risk of security problems by increasing the exposure of your system.

With this in mind, do you wish to enable inetd?

[ Yes ] No

계속 진행하도록 [Yes]를 선택한다.

User Confirmation Requested

inetd(8) relies on its configuration file, /etc/inetd.conf, to determine which of its Internet services will be available. The default FreeBSD inetd.conf(5) leaves all services disabled by default, so they must be specifically enabled in the configuration file before they will function, even once inetd(8) is enabled. Note that services for IPv6 must be separately enabled from IPv4 services.

Select [Yes] now to invoke an editor on /etc/inetd.conf, or [No] to use the current settings.

[ Yes ] No

[Yes]를 선택하고 라인 시작 부분의 #을 삭제하여 서비스를 추가할 수 있다.



그림 2-33. inetd.conf 편집

```
^_ (escape) menu      ^y search prompt    ^k delete line      ^p prev li         ^g prev page
^o ascii code        ^x search           ^l undelete line   ^n next li        ^v next page
^u end of file       ^a begin of line    ^w delete word     ^b back 1 char
^t begin of file     ^e end of line      ^r restore word    ^f forward 1 char
^c command           ^d delete char      ^j undelete char   ^z next word
=====
L: 1 C: 1
Sep 22 23:42:04 k6-2 login: ROOT LOGIN (root) ON ttyv121:13:33 obrien Exp $
#
# Internet server configuration database
#
# Define *both* IPv4 and IPv6 entries for dual-stack support.
# To disable a service, comment it out by prefixing the line with '#'.
# To enable a service, remove the '#' at the beginning of the line.
#
#ftp      stream  tcp      nowait  root    /usr/libexec/lukemftpd  ftpd -l -r
#ftp      stream  tcp      nowait  root    /usr/libexec/ftpd       ftpd -l
#ftp      stream  tcp6     nowait  root    /usr/libexec/ftpd       ftpd -l
#telnet   stream  tcp      nowait  root    /usr/libexec/telnetd    telnetd
#telnet   stream  tcp6     nowait  root    /usr/libexec/telnetd    telnetd
#shell    stream  tcp      nowait  root    /usr/libexec/rshd       rshd
#shell    stream  tcp6     nowait  root    /usr/libexec/rshd       rshd
#login    stream  tcp      nowait  root    /usr/libexec/rlogind    rlogind
#login    stream  tcp6     nowait  root    /usr/libexec/rlogind    rlogind
#finger   stream  tcp      nowait/3/10 nobody /usr/libexec/fingerd    fingerd -s
file "/etc/inetd.conf", 119 lines
```

원하는 서비스를 추가하고 **Esc** 를 누르면 **exit** 와 변경한 내용을 저장하는 메뉴가 나타난다.

## 2.9.4 익명 FTP

```

User Confirmation Requested
Do you want to have anonymous FTP access to this machine?

Yes    [ No ]
```

### 2.9.4.1 익명 FTP 거부

기본값인 [No]를 선택하고 **Enter** 를 누르면 계정과 패스워드를 가진 유저만 FTP 로 머신에 접근할 수 있다.

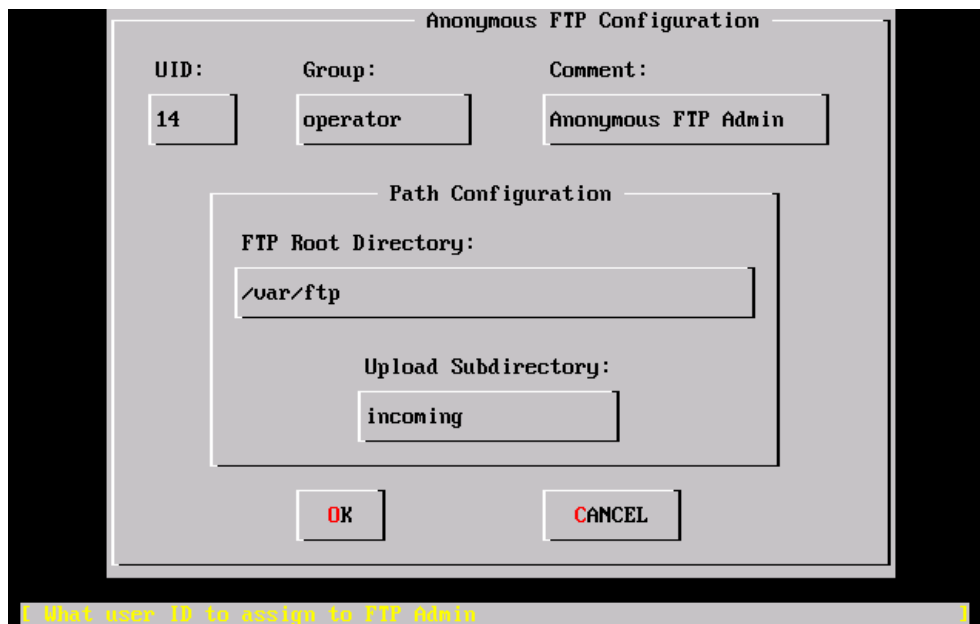
---

## 2.9.4.2 익명 FTP 허용

익명 FTP 를 허용하였다면 누구나 머신에 접근할 수 있다. 이 옵션을 활성화하기 전에 보안 과 밀접한 관련이 있으므로 심사숙고한다. 보안에 관한 더 많은 정보는 14 장을 본다.

익명 FTP 를 허용하려면 방향키로 [Yes]를 선택하고 **Enter** 를 누른다.  
다음 화면이(또는 비슷한) 나타난다.

그림 2-34. 기본 익명 FTP 설정



F1 을 누르면 도움말 화면이 나타난다:

This screen allows you to configure the anonymous FTP user.

The following configuration values are editable:

UID: The user ID you wish to assign to the anonymous FTP user.  
All files uploaded will be owned by this ID.

Group: Which group you wish the anonymous FTP user to be in.

Comment: String describing this user in /etc/passwd

FTP Root Directory:

Where files available for anonymous FTP will be kept.

Upload subdirectory:

Where files uploaded by anonymous FTP users will go.

FTP root 디렉터리는 기본적으로 /var 에 할당된다. FTP 에 필요한 공간이 충분하지 않다면 FTP Root Directory 를 /usr/ftp 로 설정하여 /usr 디렉터리를 사용할 수 있다.

적당한 값을 입력하고 **Enter** 를 눌러 계속한다.

User Confirmation Requested

Create a welcome message file for anonymous FTP users?

[ Yes ] No

[Yes]를 선택하고 **Enter** 를 눌렀다면 메시지를 편집하는 ee 라는 에디터가 자동으로 시작된다.

그림 2-35. FTP 환영 메시지 편집

```
^[ (escape) menu  ^y search prompt  ^k delete line    ^p prev line     ^g prev page
^o ascii code    ^x search         ^l undelete line  ^n next line     ^u next page
^u end of file   ^a begin of line  ^w delete word    ^b back char     ^z next word
^t begin of file ^e end of line    ^r restore word   ^f forward char
^c command       ^d delete char    ^j undelete char  ESC-Enter: exit
=====
Your welcome message here.
=====
file "/var/ftp/etc/ftpmotd", 1 lines, read only
```

---

지시대로 메시지를 편집하거나 나중에 다른 텍스트 에디터로 메시지를 변경할 수 있다. 에디터 화면 하단의 파일 이름과 위치를 적어둔다.

**Esc** 를 누르면 기본값이 **a) leave editor** 인 팝업 메뉴가 나타난다. **Enter** 를 눌러 빠져 나간 후 계속 진행한다.

## 2.9.5 네트워크 파일시스템 설정

네트워크 파일시스템(NFS)은 네트워크를 통해 파일을 공유한다. 머신은 서버, 클라이언트 또는 두 가지로 설정할 수 있다. 더 자세한 정보는 23 장을 참고한다.

### 2.9.5.1 NFS 서버

User Confirmation Requested

Do you want to configure this machine as an NFS server?

Yes    [ No ]

네트워크 파일시스템 서버나 클라이언트가 필요없으면 [No]를 선택하고 **Enter** 를 누른다. [Yes]를 선택하면 생성해야될 exports 파일(공유 파일)을 표시하는 메시지가 나타난다.

Message

Operating as an NFS server means that you must first configure an /etc/exports file to indicate which hosts are allowed certain kinds of access to your local filesystems.

Press [Enter] now to invoke an editor on /etc/exports

[ OK ]

**Enter** 를 눌러 계속 진행한다. 텍스트 에디터로 exports 파일을 생성하고 편집한다.

그림 2-36. exports 편집

```
^i (escape) menu ^y search prompt ^k delete line ^p prev line ^g prev page
^o ascii code ^x search ^l undelete line ^n next line ^v next page
^u end of file ^a begin of line ^w delete word ^b back char ^z next word
^t begin of file ^e end of line ^r restore word ^f forward char
^c command ^d delete char ^j undelete char ESC-Enter: exit
=====
#The following examples export /usr to 3 machines named after ducks,
#/home and all directories under it to machines named after dead rock stars
#and, finally, /a to 2 privileged machines allowed to write on it as root.
#/usr huey louie dewie
#/home -alldirs janice jimmy frank
#/a -maproot=0 bill albert
#
# You should replace these lines with your actual exported filesystems.

file "/etc/exports", 9 lines
```

지시대로 실제 export 할 파일시스템을 추가하거나 나중에 텍스트 에디터를 사용하면된다. 에디터 화면 하단의 파일 이름과 위치를 적어둔다.

Esc 를 누르면 a)leave editor 기본 팝업 메뉴가 나타난다. Enter 를 눌러 나가서 계속 진행한다.

### 2.9.5.2 NFS 클라이언트

NFS 클라이언트로 머신은 NFS 서버에 접근할 수 있다.

User Confirmation Requested

Do you want to configure this machine as an NFS client?

Yes [ No ]

방향키로 [Yes]와 [No] 중 원하는 것을 선택하고 Enter 를 누른다.

---

## 2.9.6 보안 프로필

“보안 프로필”은 특정 프로그램과 다른 설정들을 활성화하거나 비활성하여 원하는 보안 비용을 편리하게 조정하는 설정 옵션이다. 보안 프로필을 높게 설정하면 기본적으로 더 적은 프로그램이 활성화된다. 이것이 보안의 기본적인 원칙 중 하나이며 꼭 필요한 것 외에는 실행하지 않는다.

다음 보안 프로필은 단지 기본 설정임을 기억한다. 모든 프로그램은 FreeBSD 를 설치한 후 /etc/rc.conf 에 적당한 라인을 추가하거나 편집해서 활성화하고 비활성할 수 있다. 더 많은 정보는 rc.conf(5) 매뉴얼 페이지를 참고한다.

다음 표는 각 보안 프로필이 무엇을 의미하는지 설명한다. 종렬은 보안 프로필이고 횡렬은 활성화하거나 비활성할 수 있는 프로그램 또는 기능이다.

표 2-4. 가능한 보안 프로필

	최고	중간
Sendmail(8)	NO	YES
Sshd(8)	NO	YES
Portmap(8)	NO	MAYBE ❶
NFS server	NO	YES
Securelevel(8)	YES ❷	NO

❶ 설치하는 동안 NFS 서버나 클라이언트를 설정했다면 portmapper 가 활성화된다.  
❷ 보안 레벨이 "Extreme"이나 "High"로 설정된 보안 프로필을 선택했다면 그 의미를 이해하고 있어야 한다. init(8) 매뉴얼 페이지에있는 보안 레벨의 의미를 집중해서 읽지 않으면 나중에 중요한 문제에 부딪히게 된다.

User Confirmation Requested

Do you want to select a default security profile for this host (select No for "medium" security)?

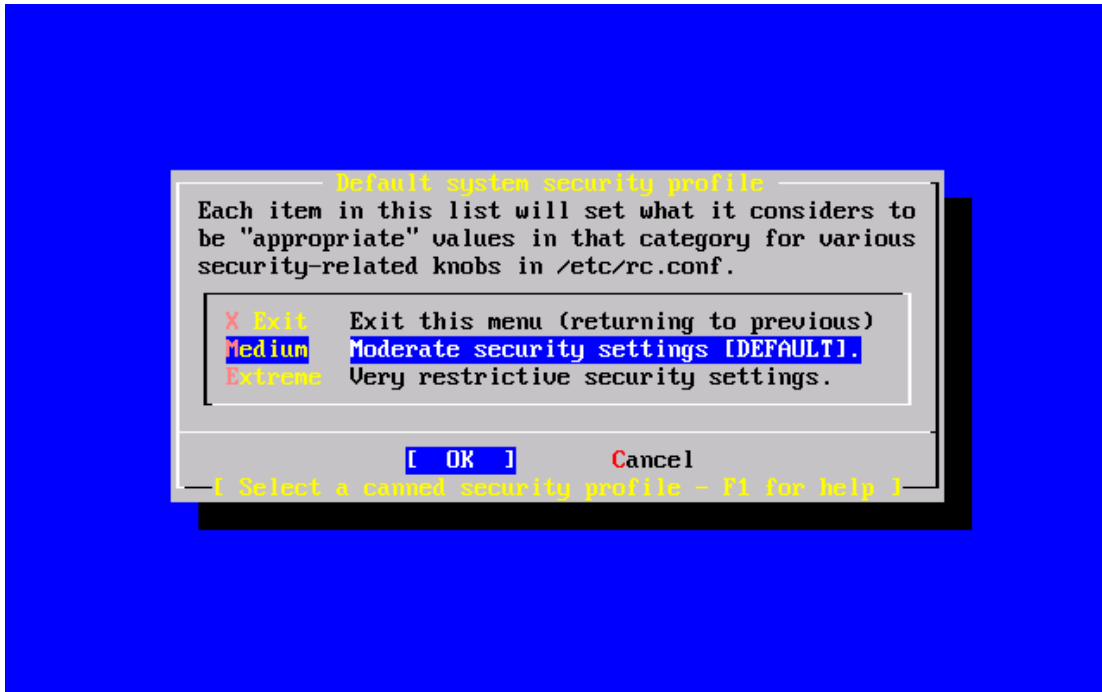
[ Yes ] No

[No]를 선택하고 **Enter** 를 누르면 보안 프로필이 중간으로 지정된다.

---

[Yes]를 선택하고 **Enter** 를 누르면 다른 보안 프로필을 선택할 수 있다.

그림 2-37. 보안 프로필 옵션



여기서 **F1** 을 누르면 도움말이 나타난다. **Enter** 를 누르면 다시 선택 메뉴로 돌아간다.

다른 레벨이 필요하지 않다면 방향키로 **Medium** 을 선택하고 [OK]에서 **Enter** 를 누른다.

선택한 보안 설정에 따라 적절한 확인 메시지가 나타난다.

중급(Medium)의 보안 프로필을 선택하면 다음 메시지가 나타난다.

```
Message

Moderate security settings have been selected.

Sendmail and SSHd have been enabled, securelevels are
disabled, and NFS server setting have been left intact.
PLEASE NOTE that this still does not save you from having
to properly secure your system in other ways or exercise
due diligence in your administration, this simply picks
a standard set of out-of-box defaults to start with.
```

---

To change any of these settings later, edit /etc/rc.conf

[OK]

가장 높은(Extreme) 보안 프로필을 선택하면 다음 메시지가 나타난다.

Message

Extreme security settings have been selected.

Sendmail, SSHd, and NFS services have been disabled, and securelevels have been enabled.

PLEASE NOTE that this still does not save you from having to properly secure your system in other ways or exercise due diligence in your administration, this simply picks a more secure set of out-of-box defaults to start with.

To change any of these settings later, edit /etc/rc.conf

[OK]

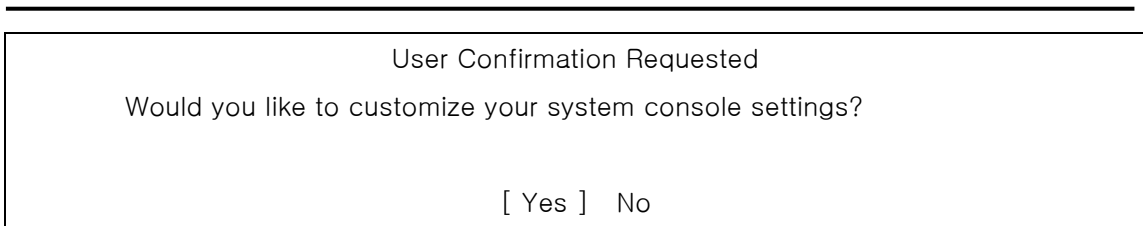
Enter 를 누르고 설치 후 설정을 계속한다.

**주의:** **Extreme** 으로 설정 했더라도 선택한 보안 프로필이 완벽한 보안을 보장하는 것이 아니다. 적절한 메일링 리스트([http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/handbook/eresources.html#ERESOURCES-MAIL](http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/eresources.html#ERESOURCES-MAIL))를 읽어서 보안 쟁점에 관심을 갖고 어려운 패스워드를 사용하는 좋은 보안 습관을 가지는 연습이 필요하다.

## 2.9.7 시스템 콘솔 설정

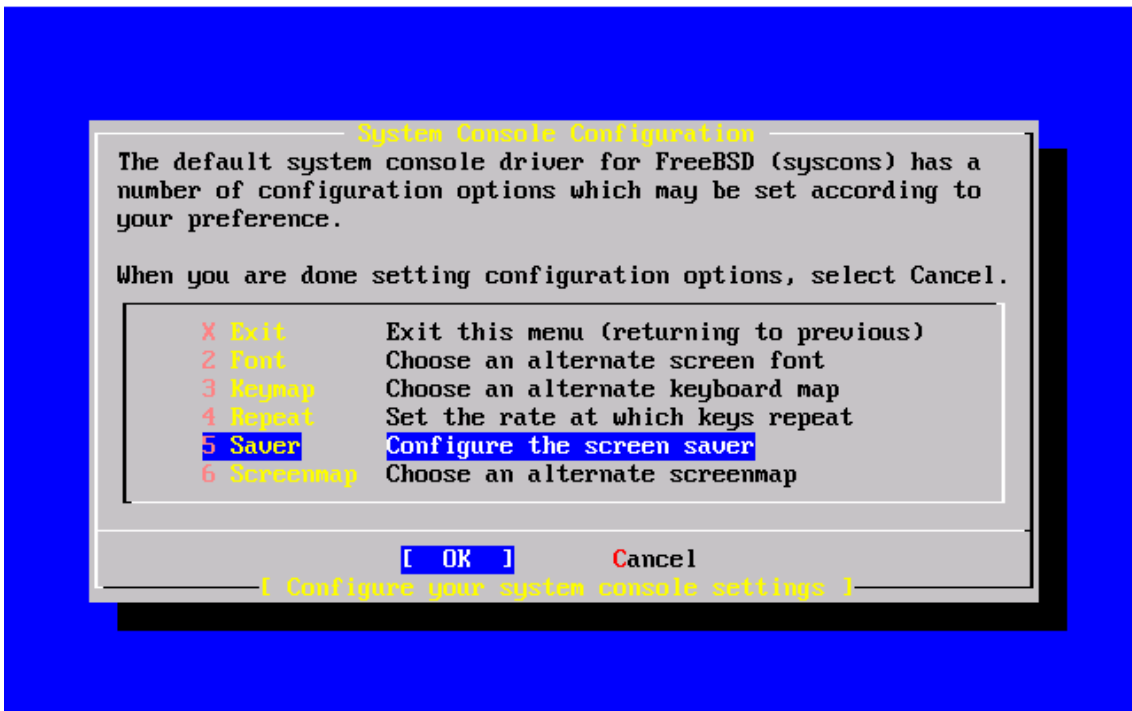
원하는 대로 시스템 콘솔을 수정할 수 있는 여러가지 옵션이 있다.





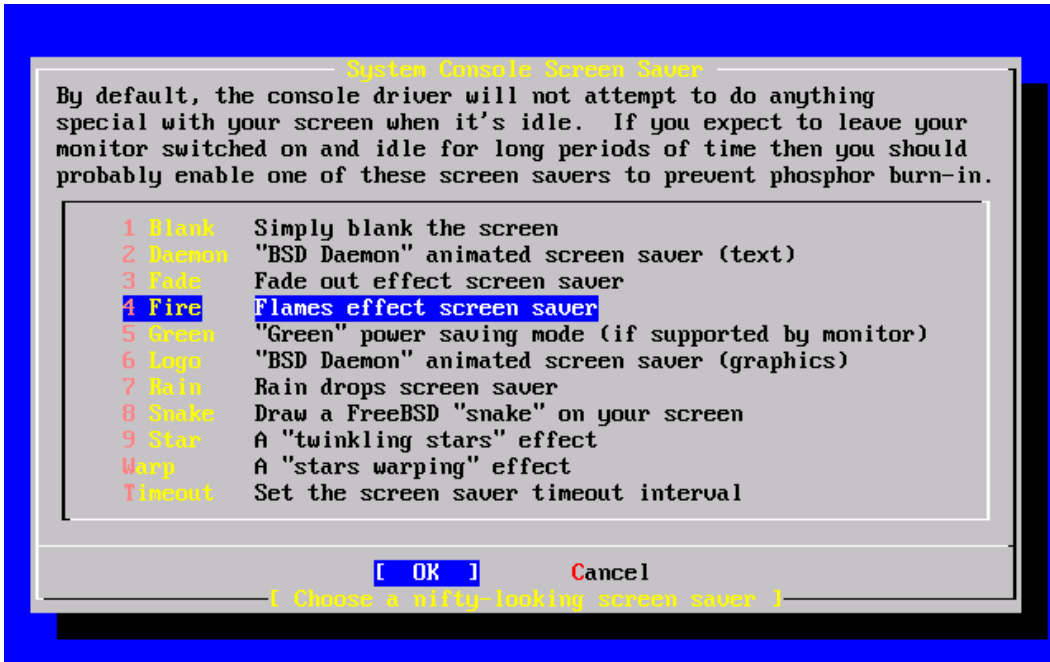
옵션을 확인한 후 설정하려면 [Yes]를 선택하고 **Enter** 를 누른다.

그림 2-38. 시스템 콘솔 설정 옵션



일반적으로 사용되는 옵션은 화면 보호기다. 방향키로 **Saver** 를 선택하고 **Enter** 를 누른다.

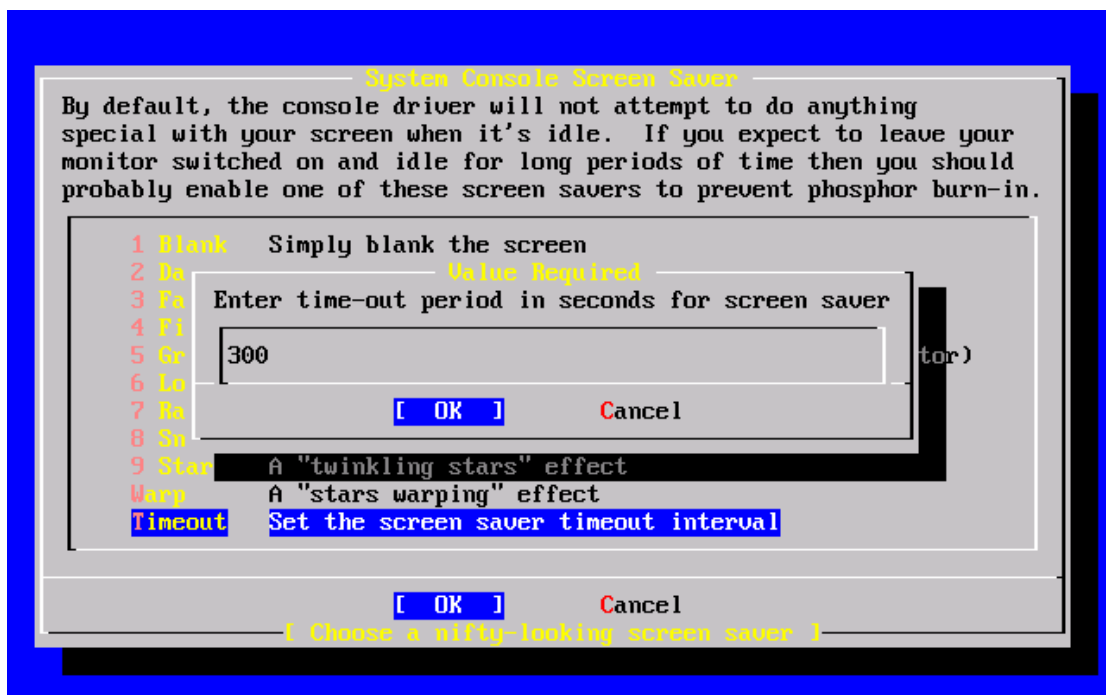
그림 2-39. 화면 보호기 옵션



원하는 화면 보호기를 선택하고 **Enter** 를 누른다. 시스템 콘솔 설정 메뉴가 다시 표시된다.

기본 시간 간격은 300 초이다. 시간 간격을 조정하고 **Saver** 를 다시 선택한다. 화면 보호기 옵션 메뉴에서 방향키로 **Timeout** 을 선택하고 **Enter** 를 누르면 팝업 메뉴가 나타난다:

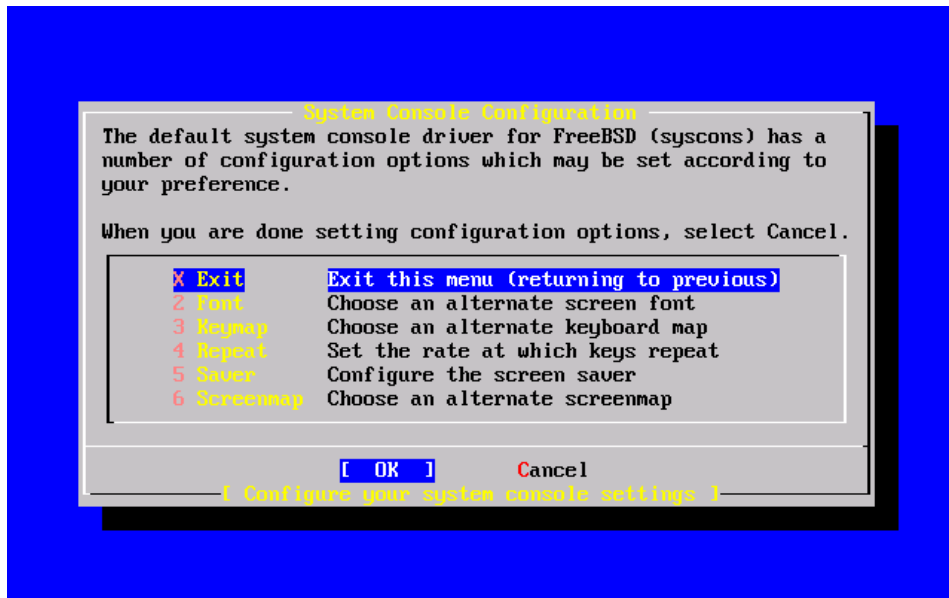
그림 2-40. 화면 보호기 타임아웃



---

값을 변경하고 [Ok]를 선택한 후 Enter 를 눌러 시스템 콘솔 설정 메뉴로 돌아온다.

그림 2-41. 시스템 콘솔 설정 나가기

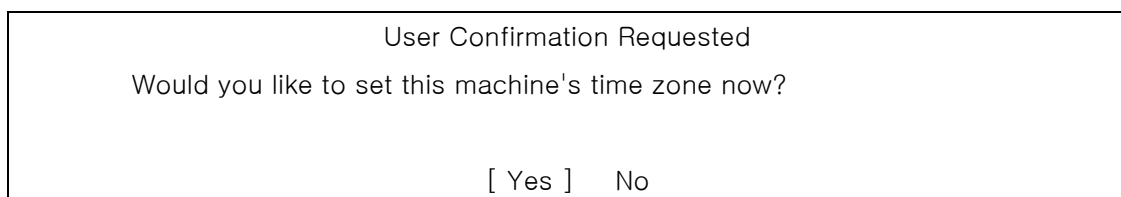


Exit 를 선택하고 Enter 를 누르면 설치 후 설정이 계속 진행된다.

## 2.9.8 시간대 설정

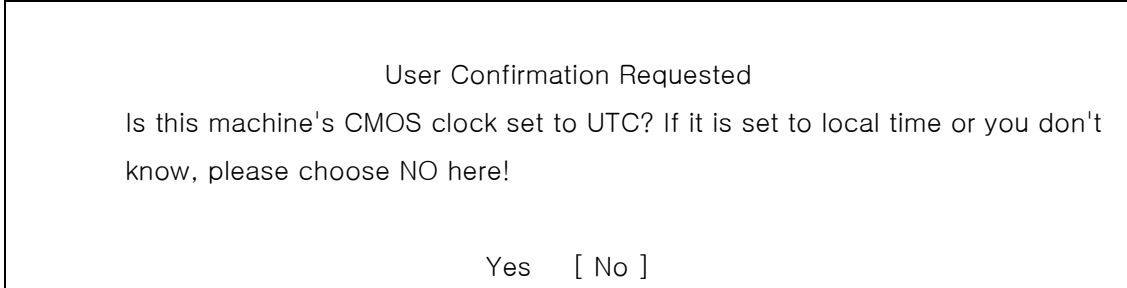
머신의 시간대를 설정하면 자동으로 특정 지역 시간으로 정확히 변경하고 적당한 기능에 맞춰 다른 시간대가 수행된다.

예제는 미국의 동부 시간대에 위치한 머신을 보여준다. 시간대 선택은 머신의 지질학적 위치를 따른다.



---

[Yes]를 선택하고 **Enter** 를 눌러 시간대를 설정한다.



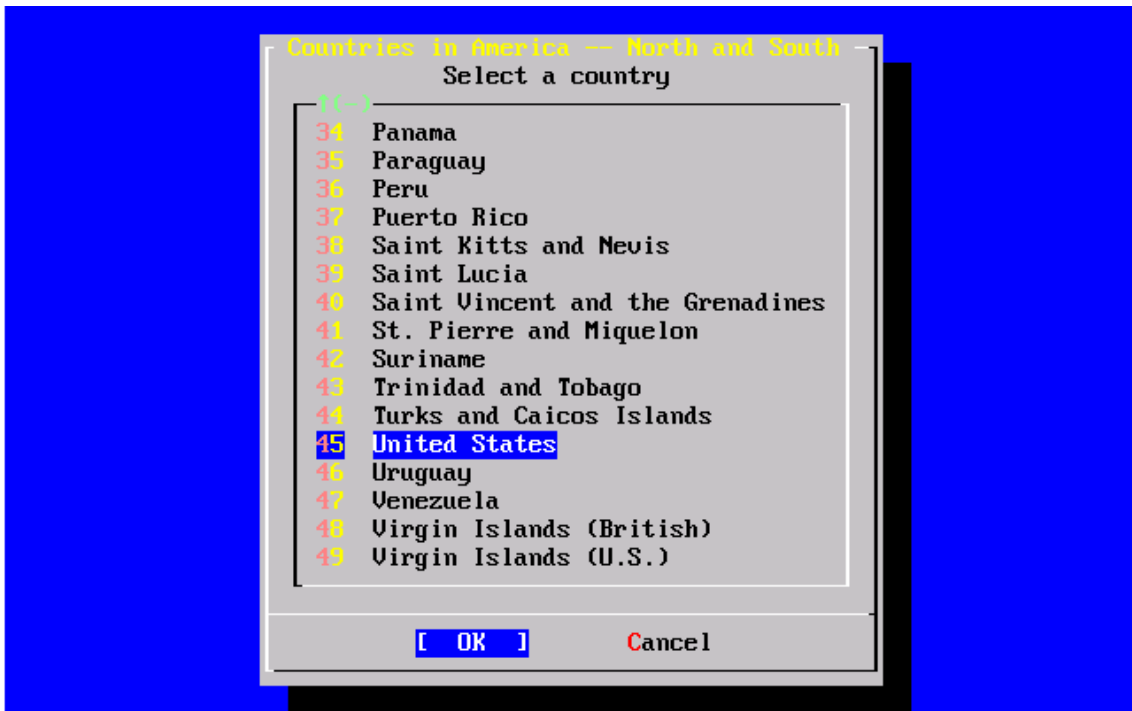
[Yes] 또는 [No]를 선택하여 머신의 시간을 설정하고 **Enter** 를 누른다.

그림 2-42. 지역 선택



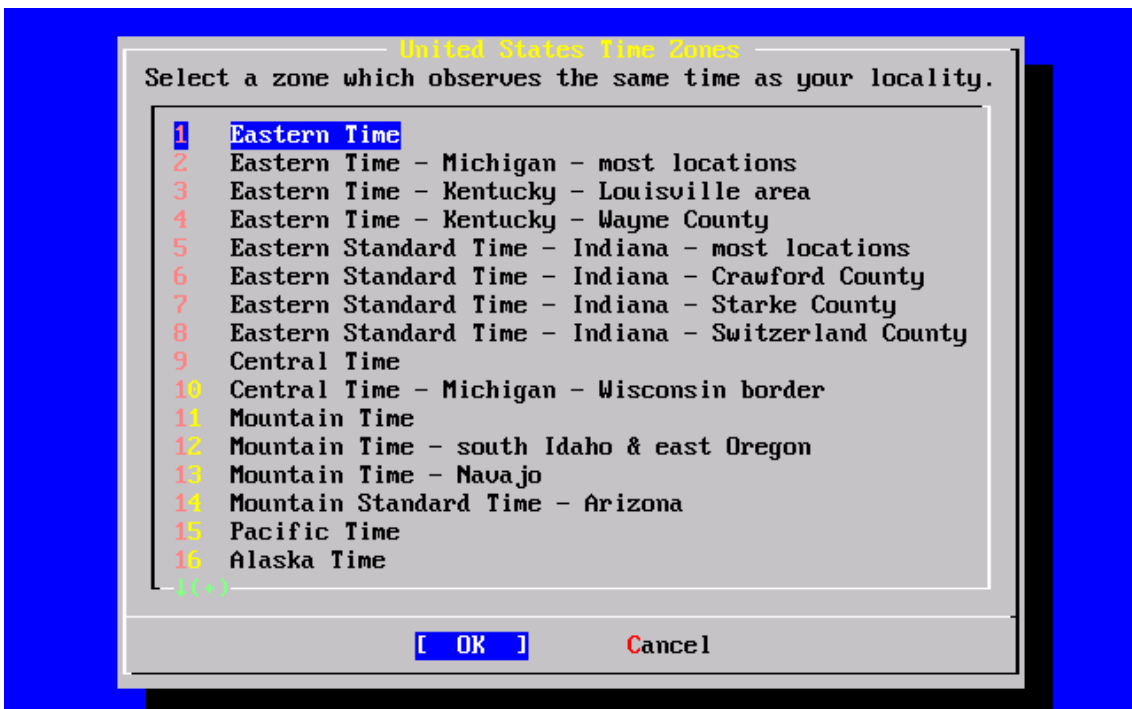
적당한 지역을 선택하고 **Enter** 를 누른다.

그림 2-43. 나라 선택



원하는 나라를 선택하고 Enter 를 누른다.

그림 2-44. 시간대 선택



---

적당한 시간대를 선택하고 **Enter** 를 누른다.

<p>Confirmation</p> <p>Does the abbreviation 'EDT' look reasonable?</p> <p>[ Yes ] No</p>
---

시간대가 정확한지 확인하는 메시지가 나타나므로 **Enter** 를 눌러 설치 후 설정을 계속한다.

## 2.9.9 Linux 호환성

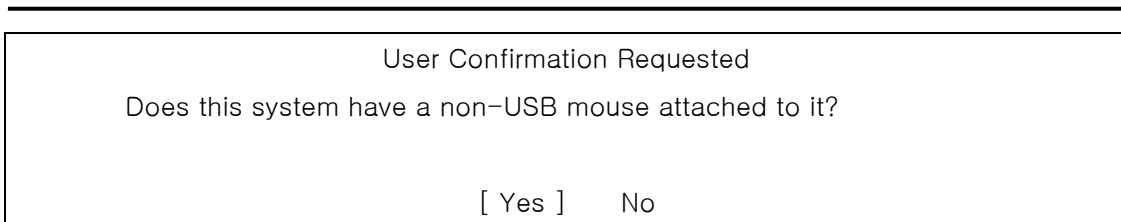
<p>User Confirmation Requested</p> <p>Would you like to enable Linux binary compatibility?</p> <p>[ Yes ] No</p>
--

[Yes]를 선택하고 **Enter** 를 누르면 FreeBSD 에서 리눅스 소프트웨어를 사용할 수 있다. 이 설치 과정은 리눅스와 호환을위해 적당한 패키지를 추가한다.

FTP 로 설치하려면 머신이 인터넷에 연결되어 있어야된다. 가끔 원격 FTP 사이트에 리눅스 바이러리 호환 파일 같은 배포본이 없을 수 있다. 이것도 필요하면 나중에 설치할 수 있다.

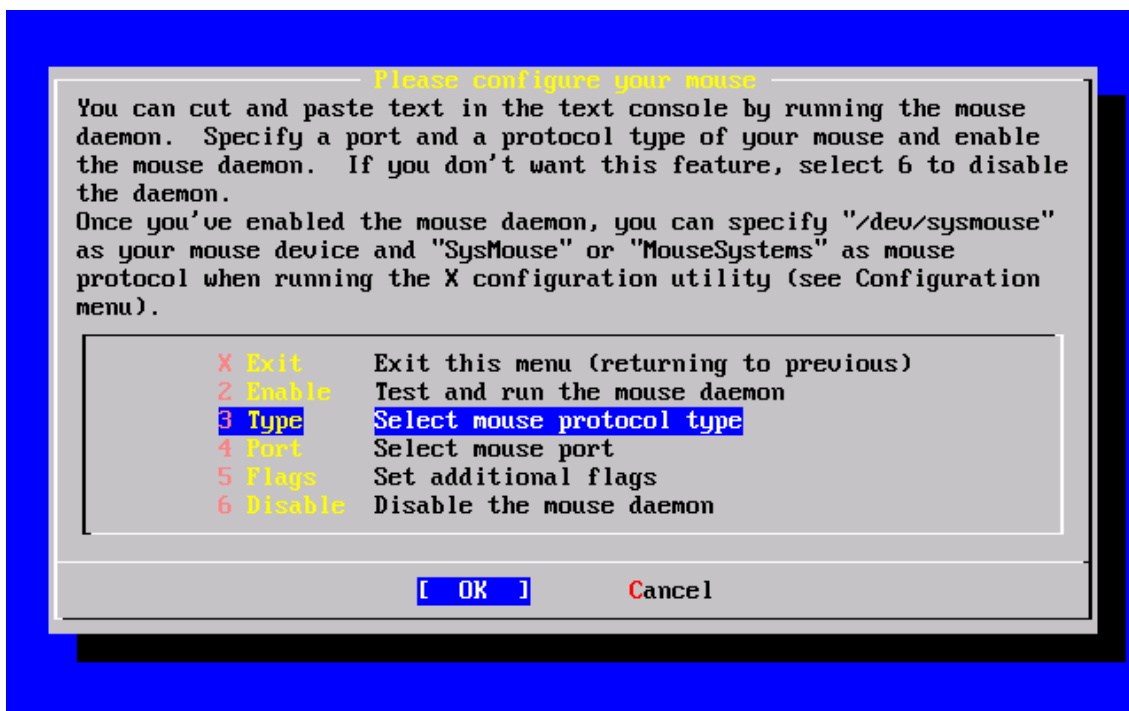
## 2.9.10 마우스 설정

이 옵션은 3 버튼 마우스로 콘솔과 유저 프로그램에서 텍스트를 잘라내고 붙일 수 있다. 2 버튼 마우스를 사용한다면 설치 후에 3 버튼 스타일의 에뮬레이팅에 관한 자세한 사항을 moused(8) 매뉴얼 페이지에서 참고한다. 이 예제는 USB 가 아닌 일반적인 마우스 설정을 (PS/2 또는 COM 포트 마우스 같은) 설명한다.



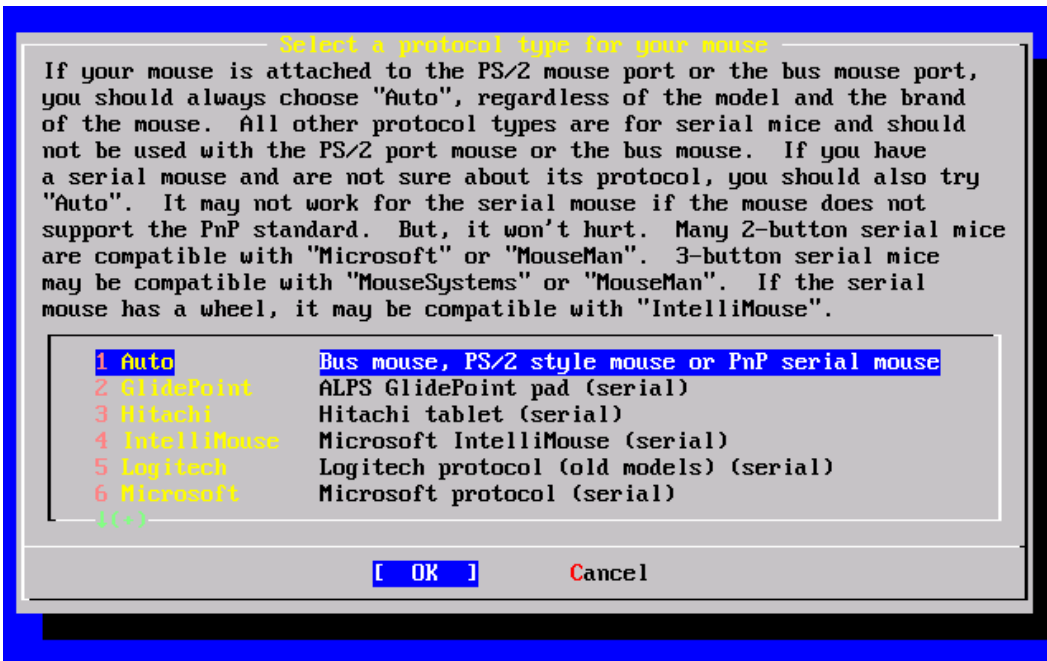
USB 가 아닌 마우스는 [Yes]를 선택하고 USB 마우스는 [No]를 선택한 후 **Enter** 를 누른다.

그림 2-45. 마우스 프로토콜 타입 선택



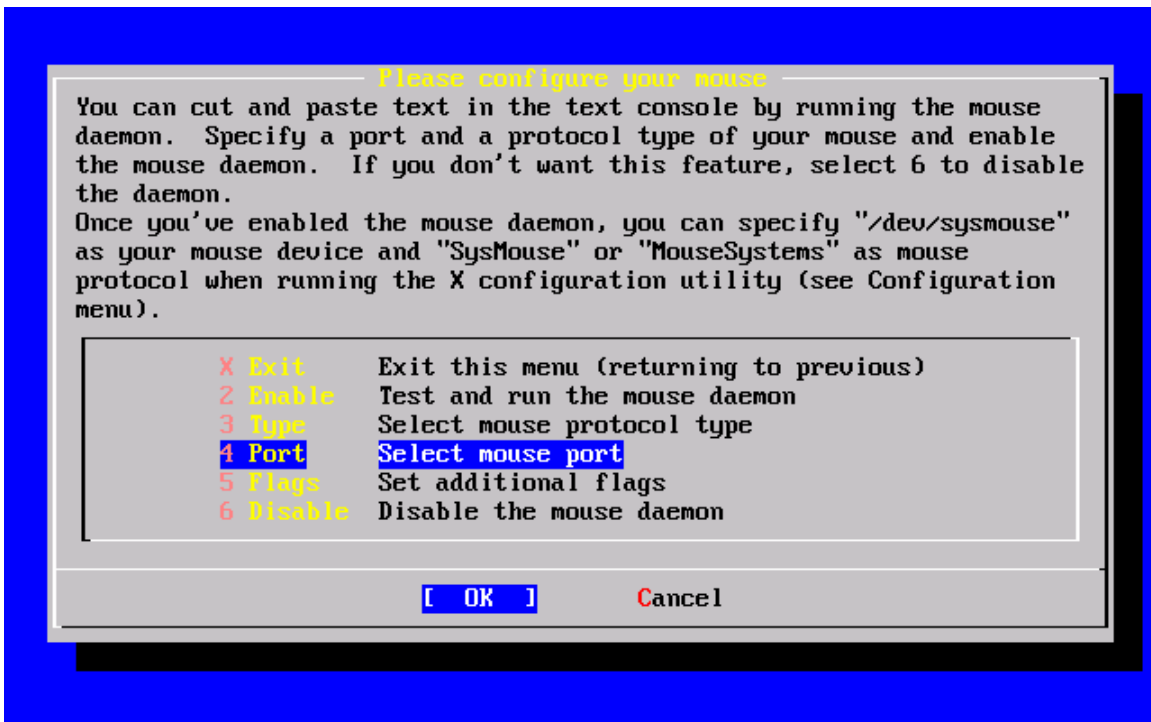
방향키로 **Type** 을 선택하고 **Enter** 를 누른다.

그림 2-46. 마우스 프로토콜 설정



이 예제에 사용된 마우스는 PS/2 타입이기 때문에 기본값인 **Auto**가 적당하다. 프로토콜을 변경하려면 다른 옵션을 선택한다. [OK]를 밝게하고 **Enter**를 눌러 이 메뉴에서 나온다.

그림 2-47. 마우스 포트 설정

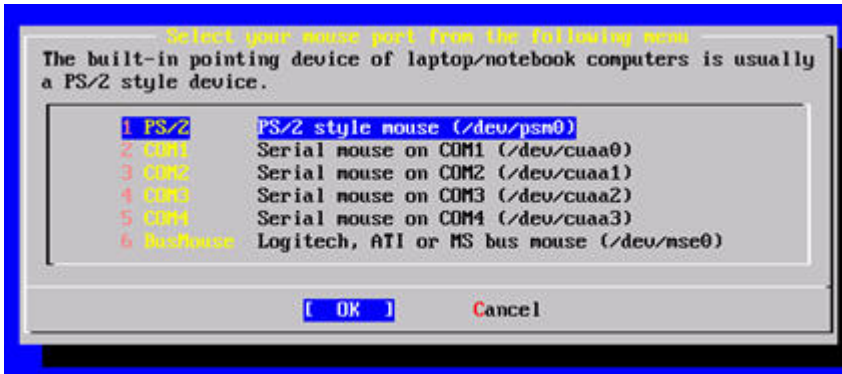




---

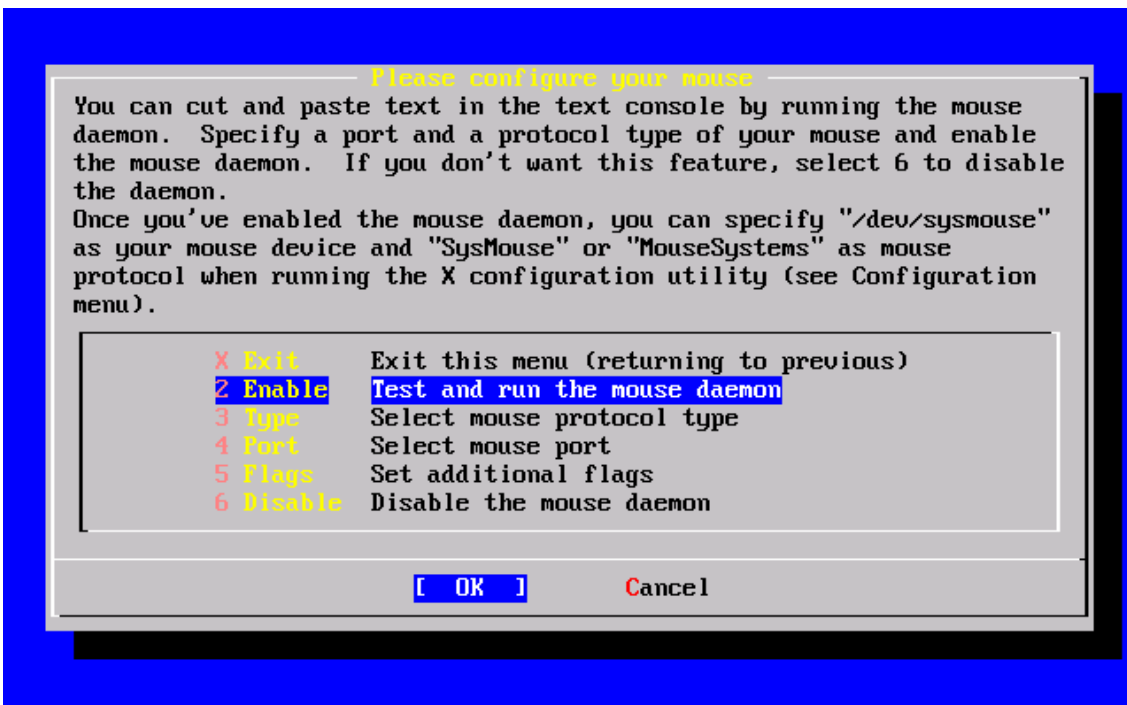
Port 를 선택하고 Enter 를 누른다.

그림 2-48. 마우스 포트 지정



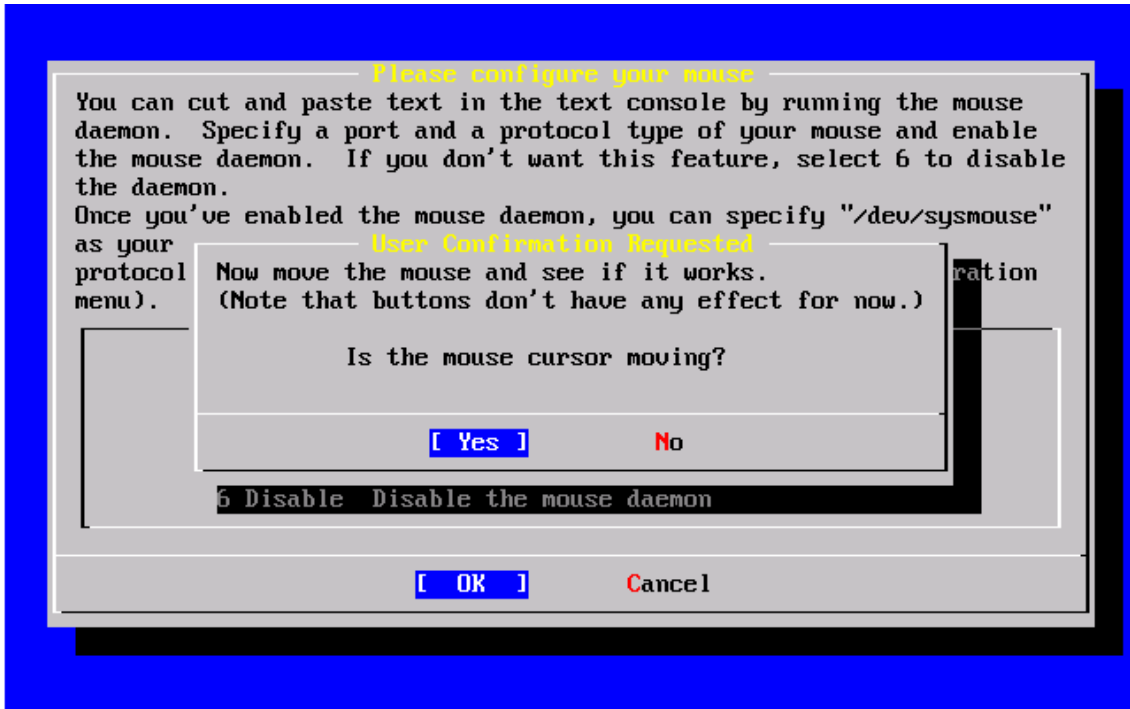
이 시스템은 PS/2 마우스를 가지고있기 때문에 기본값인 PS/2 가 적당하다. 포트를 변경하려면 방향키를 사용하고 Enter 를 누른다.

그림 2-49. 마우스 데몬 활성화



마지막으로 마우스 데몬을 활성화해서 테스트 한다.

그림 2-50. 마우스 데몬 테스트



마우스를 움직여 화면에서 커서가 움직이는것을 확인한다. 정상적으로 동작하면 [Yes]를 선택하고 **Enter** 를 누른다. 그렇지 않으면 마우스가 정확히 설정되지 않았기 때문에 [No]를 선택해서 다른 설정 옵션을 사용해본다.

[Yes]를 선택하여 이전 메뉴로 돌아와서 방향키로 **Exit** 를 선택하고 **Enter** 를 눌러 설치 후 설정을 계속한다.

## 2.9.11 추가적인 네트워크 서비스 설정

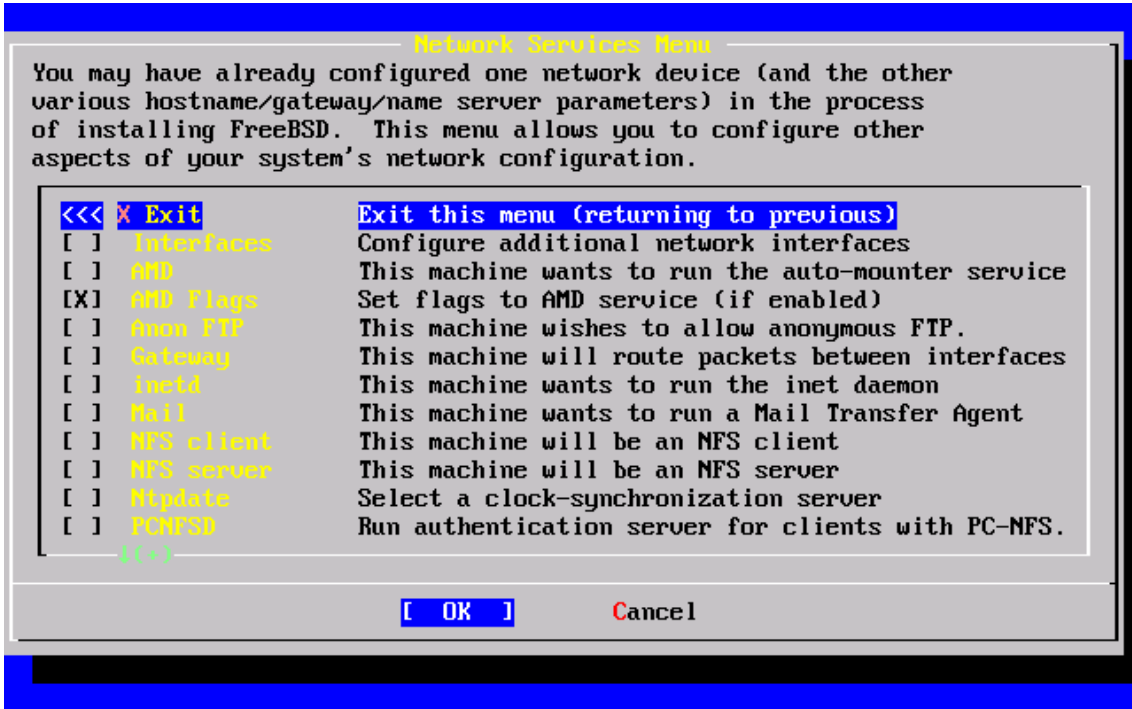
네트워크에 대한 지식이 부족한 새로운 유저에게 네트워크 서비스를 설정하는 것은 어려운 작업이다. 인터넷을 포함한 네트워크는 FreeBSD 를 포함하여 모든 현대 운영체제에게 중요하다. 따라서 FreeBSD 의 광범위한 네트워크 능력을 이해하는 것이 상당히 유용하다. 설치하는 동안 네트워크 서비스를 설정하면 다양한 서비스를 이해하는데 많은 도움이 된다.

네트워크 서비스는 네트워크의 어느 곳에서라도 연결을 허용하는 프로그램이다. 이런 프로그램이 악용되는 것을 방지하기위해 많은 노력을 하지만 프로그래머는 완벽하지 않기 때문에 네트워크 서비스의 버그는 악한 마음을 가지고있는 공격자에게 이용되기도 한다. 여러분

이 알고있고 필요한 네트워크 서비스만 활성화하는것이 중요하다. 의심된다면 필요할 때까지 네트워크 서비스를 비활성화하는것이 가장 좋다. 나중에 **sysinstall** 을 다시 실행하거나 `/etc/rc.conf` 파일에서 제공하는 옵션으로 언제든지 활성화할 수 있다.

네트워크 옵션을 선택하면 다음과 비슷한 메뉴가 나타난다.

그림 2-51. 상위 레벨 네트워크 설정



첫 번째 옵션 **Interfaces** 는 앞에서 설명하였기 때문에 이 옵션은 생략한다.

**AMD** 옵션은 BSD 자동 마운트 유틸리티 지원을 추가한다. 이 옵션은 보통 NFS 프로토콜 (아래 내용을 본다)과 결되하여 자동으로 원격 파일시스템을 마운트할 때 사용된다. 이곳에 특별한 설정은 필요없다.

다음 라인은 **AMD Flags** 옵션이다. 이 옵션을 선택하면 특정 AMD 플래그를 입력하는 메뉴가 나타난다. 이 메뉴는 기본 옵션을 가지고 있다.

```
-a /.amd_mnt -l syslog /host /etc/amd.map /net /etc/amd.map
```

`-a` 옵션은 여기서는 `/.amd_mnt` 로 지정된 기본 마운트 위치를 설정한다. `-l` 옵션은 기본 `log` 파일을 지정하지만 `syslogd` 가 사용되면 모든 로그는 시스템 로그 데몬으로 보내진다.

---

/host 디렉터리는 원격 호스트로 공유한 파일시스템을 마운트할 때 사용되지만 /net 디렉터리는 IP 주소로 공유한 파일시스템을 마운트할 때 사용된다. /etc/amd.map 파일은 AMD 공유에 기본 옵션을 정의한다.

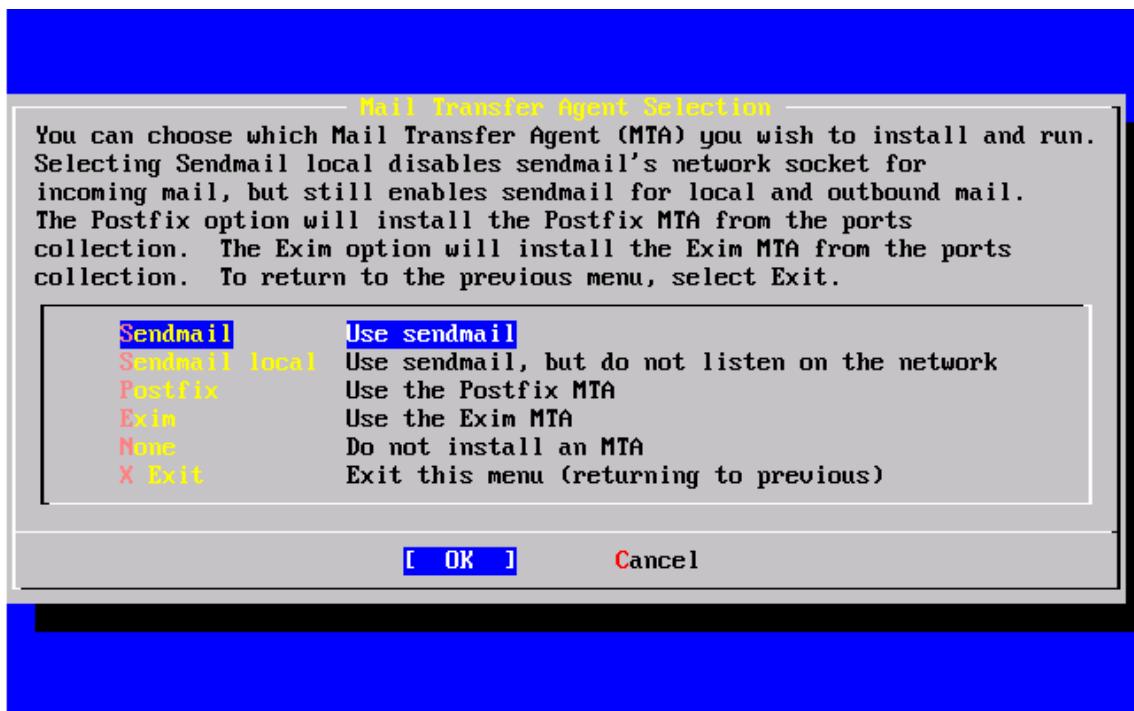
**Anon FTP** 옵션은 익명 FTP 연결을 허용한다. 이 옵션을 선택하면 이 머신을 익명 FTP 서버로 만든다. 이 옵션과 관련된 보안 위험을 생각해 본다. 보안 위험과 자세한 설정을 설명하기 위해 확인 메뉴가 표시된다.

**Gateway** 설정 메뉴는 머신을 이전에 설명한 게이트웨이로 설정한다. 설치하는 동안 실수로 게이트웨이 옵션을 선택했다면 여기서 선택한 게이트웨이 옵션을 취소할 수 있다.

**Inetd** 옵션은 위에서 설명한 inetd(8) 데몬을 설정하거나 완전히 비활성화할 때 사용한다.

**Mail** 옵션은 시스템의 기본 MTA 또는 메일 전송 에이전트를 설정할 때 사용한다. 이 옵션을 선택하면 다음 메뉴가 나타난다:

그림 2-52. 기본 MTA 선택



이곳에서 어떤 MTA 를 설치해서 기본 MTA 로 설정할지 선택할 수 있다. MTA 는 시스템이

---

나 인터넷의 유저에게 메일을 배달하는 메일 서버다.

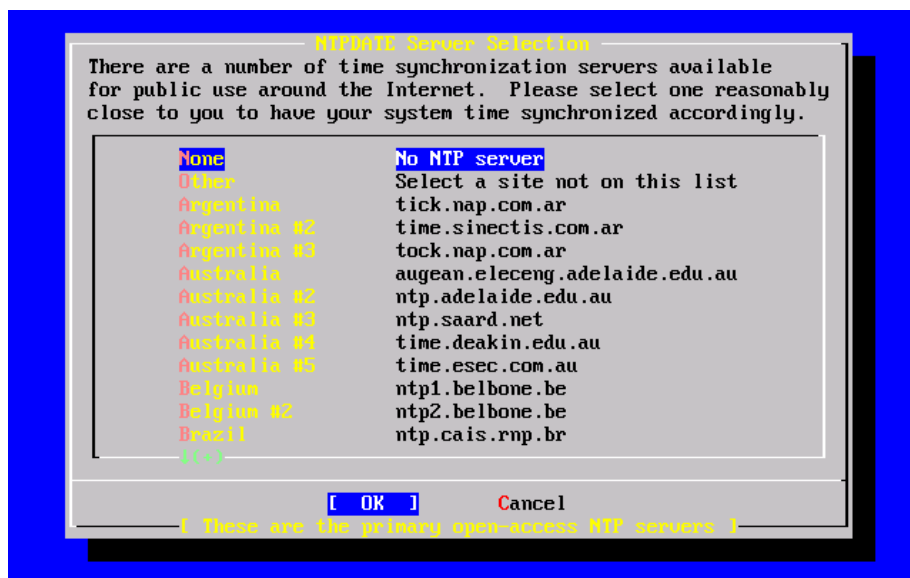
**Sendmail** 을 선택하면 FeeBSD 에 기본인 유명한 **sendmail** 서버를 설치한다. Sendmail local 옵션은 **sendmail** 을 기본 MTA 로 설정하지만 인터넷에서 배달되는 메일 수신 기능을 비활성한다. 이곳의 다른 옵션 Postfix 와 Exim 은 Sendmail 과 비슷한 기능을 수행한다. 이 프로그램들도 역시 메일을 배달하므로 어떤 유저는 **sendmail** MTA 대신 이들을 사용한다. MTA 를 선택하거나 그렇지 않더라도 네트워크 설정 메뉴는 다음 옵션 NFS 클라이언트를 나타낸다.

**NFS** 클라이언트 옵션은 NFS 를 통해 서버와 통신하도록 시스템을 설정한다. NFS 서버는 네트워크에있는 다른 머신이 NFS 프로토콜로 파일시스템을 사용할 수 있게한다. 독립된 머신이라면 이 옵션은 선택할 수 없다. 이 옵션을 선택했다면 나중에 더 많은 설정이 필요할 것이다. **NFS** 클라이언트와 서버 설정에 대한 더 많은 정보는 23 장을 본다.

아래있는 **NFS server** 옵션은 시스템을 NFS 서버로 설정한다. 이 옵션은 RPC 원격 프로시저 콜 서비스를 시작하기 위해 필요한 정보를 추가한다. RPC 는 호스트와 프로그램을 연결시킬 때 사용한다.

다음 라인은 시간 동기화와 관련된 **Ntpdate** 옵션이다. 이 옵션을 선택하면 다음과 같은 메뉴가 나타난다.

그림 2-53. Ntpdate 설정



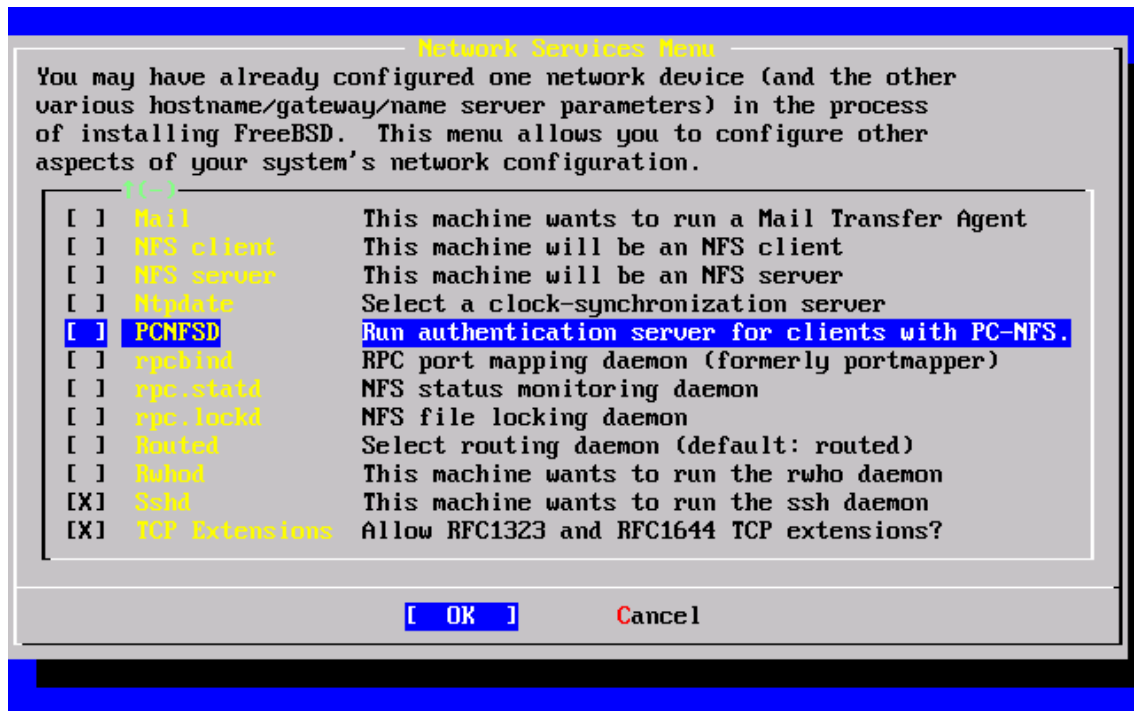
이 메뉴에서 여러분의 위치와 가장 가까운 서버를 선택한다. 가까운 서버를 선택해야 좀더

시간을 정확하게 동기화한다.

다음 옵션은 PCNFSD 로 포트 컬렉션에서 net/pcnfsd 패키지를 설치한다. 이 패키지는 Microsoft 의 MS-DOS 운영체제처럼 NFS 인증을 제공하지 못하는 시스템에게 NFS 인증 서비스를 제공하는 유용한 유틸리티다.

스크롤을 내려서 다른 옵션들을 확인한다.

그림 2-54. 아래에있는 네트워크 설정



rpcbind(8), rpc.statd(8) 그리고 rpc.lockd(8) 유틸리티는 모두 원격 프로시저 콜(RPC)에 사용한다. rpcbind 유틸리티는 NFS 서버와 클라이언트 통신을 관리하고 NFS 서버가 정확히 동작하는데 필요하다. rpc.statd 데몬은 상태를 모니터링하기 위해 다른 호스트의 rpc.statd 데몬과 상호 연동되고 상태 리포트는 보통 /var/db/statd.status 파일에 저장된다. 다음 옵션은 파일 잠금 서비스를 제공하는 rpc.lockd 옵션이다. 이 옵션은 보통 어떤 호스트가 얼마나 자주 잠금을 요청하는지 모니터링하기 위해 rpc.statd 와 사용된다. 마지막 두 개의 옵션은 디버깅에 사용하지만 NFS 서버와 클라이언트가 정확히 동작하기 위해 필요한 것은 아니다.

리스트 아래로 내려가면 다음 아이템은 라우팅 데몬 Routed 다. routed(8) 유틸리티는 멀티 캐스트 라우터를 인지하고 네트워크에 물리적으로 연결되어있는 호스트에서 요청이있으면

---

라우팅 테이블 복사본을 제공하는 네트워크 라우팅 테이블을 관리한다. 이 설정은 로컬 네트워크의 게이트웨이로 동작하는 머신에서 주로 사용된다(icmp(4)와 udp(4) 매뉴얼 페이지를 본다). 이 옵션을 선택하면 이 유틸리티의 기본 위치를 문의하는 메뉴가 나타난다. 기본 위치가 미리 정의되어 있기 때문에 **Enter** 키를 누르면 된다. 키를 누르면 **routed** 에 적용하고 싶은 플래그를 묻는 다른 메뉴가 나타난다. 기본값은 `-q` 이고 화면에 미리 표시되어있다.

다음 라인은 시스템을 초기화하는 동안 `rwhod(8)` 데몬을 시작하는 **Rwhod** 옵션이다. `rwhod` 는 시스템 메시지를 정기적으로 네트워크에 브로드캐스트 하거나 “consumer” 모드에서 이들 메시지를 모으는 유틸리티다. 더 많은 정보는 `ruptime(1)`과 `rwho(1)` 매뉴얼 페이지에서 찾을 수 있다.

리스트의 다음 옵션은 `sshd(8)` 데몬이다. 이것은 **OpenSSH** 보안 셸 서버로 표준 **telnet** 과 FTP 서버 대신 사용하도록 강력히 권장한다. `sshd` 서버는 암호화된 연결을 사용하여 호스트에서 호스트로 보안 연결을 생성하는데 사용된다.

마지막으로 **TCP Extensions** 옵션이 있다. 이것은 RFC 1323 과 RFC 1644 에 정의되어있는 TCP Extensions 을 활성화한다. 이 옵션으로 많은 호스트에서 연결 속도를 올릴 수 있지만 가끔 연결이 끊어지는 원인이된다. 서버에는 권장하지 않지만 독립된 머신에서는 유용할 수 있다.

이제 네트워크 서비스를 설정했기 때문에 다음 설정 섹션으로 이동할 수 있는 최 상단 아이탬으로 이동한다.

## 2.9.12 X 서버 설정

KDE, GNOME 같은 그래픽 유저 인터페이스를 사용하려면 X 서버 설정이 필요하다.

**Note:** **XFree86** 을 root 유저가 아닌 일반 유저에서 실행하려면 `x11/wrapper` 가 필요하다. 이 패키지는 FreeBSD4.7 부터 기본적으로 설치되고 그 이전 버전은 패키지 선택 메뉴에서 추가할 수 있다.

비디오 카드가 지원되는지 확인하기 위해 XFree86(<http://www.xfree86.org/>) 웹사이트를 체크한다.

---

User Confirmation Requested

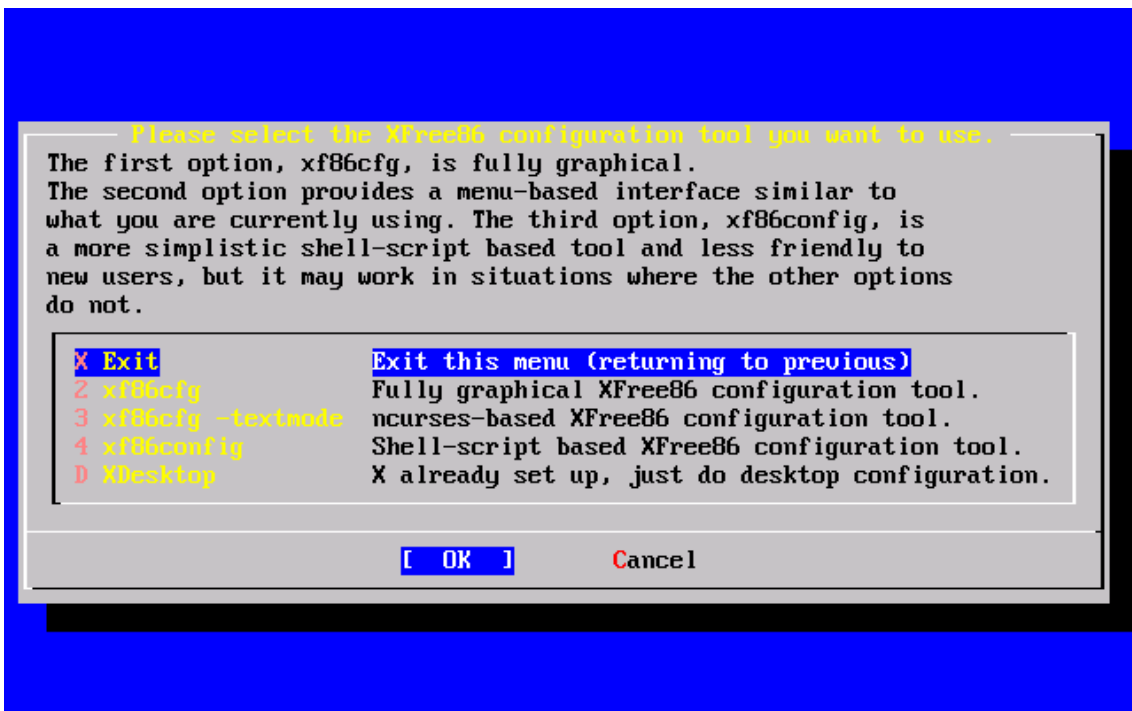
Would you like to configure your X server at this time?

[ Yes ] No

**주의:** 모니터의 설명서와 비디오 카드 정보가 필요하다. 설정이 정확하지 않으면 장비가 파손될 수 있다. 이런 정보를 가지고 있지 않다면 [No]를 선택하고 설치 후 필요한 정보가 있을 때 /stand/sysinstall 을 사용하여 **Configure** 을 선택하고 XFree86 을 선택한다. 이때 정확하지 않은 X 서버의 설정으로 머신이 정지될 수 있다. 설치를 끝낸 후 X 서버를 설정할 것을 권장하고있다.

그래픽 카드와 모니터 정보를 가지고 있다면 [Yes]를 선택하고 **Enter** 를 눌러 X 서버 설정을 진행한다.

그림 2-55. 설정 방법 메뉴 선택



X 서버를 설정하는 여러가지 방법이 있으므로 방법 중 하나를 선택하고 **Enter** 를 누른다. 모든 지시를 자세히 읽어야 된다.



---

`xf86cfg` 와 `xf86cfg-textmode` 는 화면을 몇 초 정도 어둡게하고 시작되기 때문에 기다려야 한다.

다음은 `xf86config` 설정 틀을 사용한 설명이다. 선택한 설정은 시스템의 하드웨어에 따라 다르기 때문에 이곳의 화면과 다를 것이다. 다른 방법은 5장에서 자세히 설명하고 있다.

Message

You have configured and been running the mouse daemon.  
Choose "/dev/sysmouse" as the mouse port and "SysMouse" or  
"MouseSystems" as the mouse protocol in the X configuration utility.

[ OK ]

[ Press enter to continue ]

이곳에 표시된 마우스 데몬은 앞에서 검색하여 설정한 것이다. **Enter** 를 눌러 계속한다.

`xf86config` 이 시작되고 간단한 소개가 나타난다:

This program will create a basic XF86Config file, based on menu selections you make.

The XF86Config file usually resides in /usr/X11R6/etc/X11 or /etc/X11. A sample XF86Config file is supplied with XFree86; it is configured for a standard VGA card and monitor with 640x480 resolution. This program will ask for a pathname when it is ready to write the file.

You can either take the sample XF86Config as a base and edit it for your configuration, or let this program produce a base XF86Config file for your configuration and fine-tune it.

Before continuing with this program, make sure you know what video card you have, and preferably also the chipset it uses and the amount of video memory on your video card. SuperProbe may be able to help with this.

---

Press enter to continue, or ctrl-c to abort.

**Enter** 를 누르면 마우스 설정이 시작된다. 다음 지시를 따르고 마우스 프로토콜에서 “마우스 시스템”을 선택한다. PS/2 마우스를 예로 설명하더라도 마우스 포트에 /dev/sysmouse 를 사용한다.

First specify a mouse protocol type. Choose one from the following list:

1. Microsoft compatible (2-button protocol)
2. Mouse Systems (3-button protocol) & FreeBSD moused protocol
3. Bus Mouse
4. PS/2 Mouse
5. Logitech Mouse (serial, old type, Logitech protocol)
6. Logitech MouseMan (Microsoft compatible)
7. MM Series
8. MM HitTablet
9. Microsoft IntelliMouse

If you have a two-button mouse, it is most likely of type 1, and if you have a three-button mouse, it can probably support both protocol 1 and 2. There are two main varieties of the latter type: mice with a switch to select the protocol, and mice that default to 1 and require a button to be held at boot-time to select protocol 2. Some mice can be convinced to do 2 by sending a special sequence to the serial port (see the ClearDTR/ClearRTS options).

Enter a protocol number: 2

You have selected a Mouse Systems protocol mouse. If your mouse is normally in Microsoft-compatible mode, enabling the ClearDTR and ClearRTS options may cause it to switch to Mouse Systems mode when the server starts.

Please answer the following question with either 'y' or 'n'.

Do you want to enable ClearDTR and ClearRTS? n

You have selected a three-button mouse protocol. It is recommended that you do not enable Emulate3Buttons, unless the third button doesn't work.

---

Please answer the following question with either 'y' or 'n'.

Do you want to enable Emulate3Buttons? y

Now give the full device name that the mouse is connected to, for example /dev/tty00. Just pressing enter will use the default, /dev/mouse.

On FreeBSD, the default is /dev/sysmouse.

Mouse device: /dev/sysmouse

키보드가 다음에 설정할 아이템으로 일반 101 키 모델이 설명되어있다. 다른 이름을 선택하거나 간단히 **Enter** 를 눌러 기본값을 사용한다.

Please select one of the following keyboard types that is the better description of your keyboard. If nothing really matches, choose 1 (Generic 101-key PC)

- 1 Generic 101-key PC
- 2 Generic 102-key (Intl) PC
- 3 Generic 104-key PC
- 4 Generic 105-key (Intl) PC
- 5 Dell 101-key PC
- 6 Everex STEPnote
- 7 Keytronic FlexPro
- 8 Microsoft Natural
- 9 Northgate OmniKey 101
- 10 Winbook Model XP5
- 11 Japanese 106-key
- 12 PC-98xx Series
- 13 Brazilian ABNT2
- 14 HP Internet
- 15 Logitech iTouch
- 16 Logitech Cordless Desktop Pro
- 17 Logitech Internet Keyboard
- 18 Logitech Internet Navigator Keyboard

- 
- 19 Compaq Internet
  - 20 Microsoft Natural Pro
  - 21 Genius Comfy KB-16M
  - 22 IBM Rapid Access
  - 23 IBM Rapid Access II
  - 24 Chicony Internet Keyboard
  - 25 Dell Internet Keyboard

Enter a number to choose the keyboard.

1

Please select the layout corresponding to your keyboard

- 1 U.S. English
- 2 U.S. English w/ ISO9995-3
- 3 U.S. English w/ deadkeys
- 4 Albanian
- 5 Arabic
- 6 Armenian
- 7 Azerbaidjani
- 8 Belarusian
- 9 Belgian
- 10 Bengali
- 11 Brazilian
- 12 Bulgarian
- 13 Burmese
- 14 Canadian
- 15 Croatian
- 16 Czech
- 17 Czech (qwerty)
- 18 Danish

Enter a number to choose the country.

---

Press enter for the next page

1

Please enter a variant name for 'us' layout. Or just press enter for default variant

us

Please answer the following question with either 'y' or 'n'.

Do you want to select additional XKB options (group switcher, group indicator, etc.)? n

그 다음으로 모니터를 설정한다. 주사율을 초과하면 모니터가 손상될 수 있으므로 모니터의 주사율을 초과하지 않는다. 걱정된다면 필요한 정보를 보고 설정한다.

Now we want to set the specifications of the monitor. The two critical parameters are the vertical refresh rate, which is the rate at which the whole screen is refreshed, and most importantly the horizontal sync rate, which is the rate at which scanlines are displayed.

The valid range for horizontal sync and vertical sync should be documented in the manual of your monitor. If in doubt, check the monitor database `/usr/X11R6/lib/X11/doc/Monitors` to see if your monitor is there.

Press enter to continue, or ctrl-c to abort.

You must indicate the horizontal sync range of your monitor. You can either select one of the predefined ranges below that correspond to industry-standard monitor types, or give a specific range.

It is VERY IMPORTANT that you do not specify a monitor type with a horizontal

---

sync range that is beyond the capabilities of your monitor. If in doubt, choose a conservative setting.

hsync in kHz; monitor type with characteristic modes

- 1 31.5; Standard VGA, 640x480 @ 60 Hz
- 2 31.5 – 35.1; Super VGA, 800x600 @ 56 Hz
- 3 31.5, 35.5; 8514 Compatible, 1024x768 @ 87 Hz interlaced (no 800x600)
- 4 31.5, 35.15, 35.5; Super VGA, 1024x768 @ 87 Hz interlaced, 800x600 @ 56 Hz
- 5 31.5 – 37.9; Extended Super VGA, 800x600 @ 60 Hz, 640x480 @ 72 Hz
- 6 31.5 – 48.5; Non-Interlaced SVGA, 1024x768 @ 60 Hz, 800x600 @ 72 Hz
- 7 31.5 – 57.0; High Frequency SVGA, 1024x768 @ 70 Hz
- 8 31.5 – 64.3; Monitor that can do 1280x1024 @ 60 Hz
- 9 31.5 – 79.0; Monitor that can do 1280x1024 @ 74 Hz
- 10 31.5 – 82.0; Monitor that can do 1280x1024 @ 76 Hz
- 11 Enter your own horizontal sync range

Enter your choice (1-11): 6

You must indicate the vertical sync range of your monitor. You can either select one of the predefined ranges below that correspond to industry-standard monitor types, or give a specific range. For interlaced modes, the number that counts is the high one (e.g. 87 Hz rather than 43 Hz).

- 1 50-70
- 2 50-90
- 3 50-100
- 4 40-150
- 5 Enter your own vertical sync range

Enter your choice: 2

You must now enter a few identification/description strings, namely an identifier, a vendor name, and a model name. Just pressing enter will fill in default names.

The strings are free-form, spaces are allowed.

---

Enter an identifier for your monitor definition: Hitachi

그 다음은 리스트에서 비디오 카드 드라이버를 선택하는 것이다. 설치되어있는 카드를 리스트에서 지나쳤다면 **Enter** 를 계속 누르면 리스트가 다시 나온다. 다음은 리스트에서 발췌한 것이다.

Now we must configure video card specific settings. At this point you can choose to make a selection out of a database of video card definitions. Because there can be variation in Ramdacs and clock generators even between cards of the same model, it is not sensible to blindly copy the settings (e.g. a Device section). For this reason, after you make a selection, you will still be asked about the components of the card, with the settings from the chosen database entry presented as a strong hint.

The database entries include information about the chipset, what driver to run, the Ramdac and ClockChip, and comments that will be included in the Device section. However, a lot of definitions only hint about what driver to run (based on the chipset the card uses) and are untested.

If you can't find your card in the database, there's nothing to worry about. You should only choose a database entry that is exactly the same model as your card: choosing one that looks similar is just a bad idea (e.g. a GemStone Snail 64 may be as different from a GemStone Snail 64+ in terms of hardware as can be).

Do you want to look at the card database? y

288	Matrox Millennium G200 8MB	mgag200
289	Matrox Millennium G200 SD 16MB	mgag200
290	Matrox Millennium G200 SD 4MB	mgag200
291	Matrox Millennium G200 SD 8MB	mgag200
292	Matrox Millennium G400	mgag400
293	Matrox Millennium II 16MB	mga2164w

---

294	Matrox Millennium II 4MB	mga2164w
295	Matrox Millennium II 8MB	mga2164w
296	Matrox Mystique	mga1064sg
297	Matrox Mystique G200 16MB	mgag200
298	Matrox Mystique G200 4MB	mgag200
299	Matrox Mystique G200 8MB	mgag200
300	Matrox Productiva G100 4MB	mgag100
301	Matrox Productiva G100 8MB	mgag100
302	MediaGX	mediagx
303	MediaVision Proaxcel 128	ET6000
304	Mirage Z-128	ET6000
305	Miro CRYSTAL VRX	Verite 1000

Enter a number to choose the corresponding card definition.

Press enter for the next page, q to continue configuration.

288

Your selected card definition:

Identifier: Matrox Millennium G200 8MB

Chipset: mgag200

Driver: mga

Do NOT probe clocks or use any Clocks line.

Press enter to continue, or ctrl-c to abort.

Now you must give information about your video card. This will be used for the "Device" section of your video card in XF86Config.

You must indicate how much video memory you have. It is probably a good idea to use the same approximate amount as that detected by the server you intend to use. If you encounter problems that are due to the used server not supporting the amount memory you have (e.g. ATI Mach64 is limited to



---

1024K with the SVGA server), specify the maximum amount supported by the server.

How much video memory do you have on your video card:

- 1 256K
- 2 512K
- 3 1024K
- 4 2048K
- 5 4096K
- 6 Other

Enter your choice: 6

Amount of video memory in Kbytes: 8192

You must now enter a few identification/description strings, namely an identifier, a vendor name, and a model name. Just pressing enter will fill in default names (possibly from a card definition).

Your card definition is Matrox Millennium G200 8MB.

The strings are free-form, spaces are allowed.

Enter an identifier for your video card definition:

해상도는 원하는 주사율을 설정한다. 일반적으로 유용한 범위는 640x480, 800x600 그리고 1024x768 이지만 비디오 카드 성능, 모니터 크기 그리고 눈에 적합한지에 따라 선택한다. 색상 농도를 선택할 때 카드가 지원하는 최고 해상도를 선택한다.

For each depth, a list of modes (resolutions) is defined. The default resolution that the server will start-up with will be the first listed mode that can be supported by the monitor and card.

Currently it is set to:

"640x480" "800x600" "1024x768" "1280x1024" for 8-bit

---

"640x480" "800x600" "1024x768" "1280x1024" for 16-bit

"640x480" "800x600" "1024x768" "1280x1024" for 24-bit

Modes that cannot be supported due to monitor or clock constraints will be automatically skipped by the server.

- 1 Change the modes for 8-bit (256 colors)
- 2 Change the modes for 16-bit (32K/64K colors)
- 3 Change the modes for 24-bit (24-bit color)
- 4 The modes are OK, continue.

Enter your choice: 2

Select modes from the following list:

- 1 "640x400"
- 2 "640x480"
- 3 "800x600"
- 4 "1024x768"
- 5 "1280x1024"
- 6 "320x200"
- 7 "320x240"
- 8 "400x300"
- 9 "1152x864"
- a "1600x1200"
- b "1800x1400"
- c "512x384"

Please type the digits corresponding to the modes that you want to select.

For example, 432 selects "1024x768" "800x600" "640x480", with a default mode of 1024x768.

Which modes? 432

You can have a virtual screen (desktop), which is screen area that is larger than the physical screen and which is panned by moving the mouse to the edge

---

of the screen. If you don't want virtual desktop at a certain resolution, you cannot have modes listed that are larger. Each color depth can have a differently-sized virtual screen

Please answer the following question with either 'y' or 'n'.

Do you want a virtual screen that is larger than the physical screen? n

For each depth, a list of modes (resolutions) is defined. The default resolution that the server will start-up with will be the first listed mode that can be supported by the monitor and card.

Currently it is set to:

"640x480" "800x600" "1024x768" "1280x1024" for 8-bit

"1024x768" "800x600" "640x480" for 16-bit

"640x480" "800x600" "1024x768" "1280x1024" for 24-bit

Modes that cannot be supported due to monitor or clock constraints will be automatically skipped by the server.

- 1 Change the modes for 8-bit (256 colors)
- 2 Change the modes for 16-bit (32K/64K colors)
- 3 Change the modes for 24-bit (24-bit color)
- 4 The modes are OK, continue.

Enter your choice: 4

Please specify which color depth you want to use by default:

- 1 1 bit (monochrome)
- 2 4 bits (16 colors)
- 3 8 bits (256 colors)
- 4 16 bits (65536 colors)

---

5 24 bits (16 million colors)

Enter a number to choose the default depth.

4

마지막으로 설정을 /etc/XF86Config 에 저장하기 위해 **Enter** 를 누른다.

I am going to write the XF86Config file now. Make sure you don't accidentally overwrite a previously configured one.

Shall I write it to /etc/X11/XF86Config? Y

설정에 실패하면 다음 메시지가 나타날 때 [Yes]를 선택하여 다시 설정할 수 있다.

User Confirmation Requested

The XFree86 configuration process seems to have failed. Would you like to try again?

[ Yes ]          No

**XFree86** 설정에 문제가 있다면 [No]를 선택한 후 **Enter** 를 누르고 설치를 계속한다. 설치 후 명령어라인 설정 유틸리티를 실행하기 위해 root 에서 **xf86cfg - textmode** 나 **xf86config** 를 사용할 수 있다. **XFree86** 를 설정하는 추가적인 방법은 5 장에 있다. 여기서 **XFree86** 설정을 선택하지 않았다면 다음은 패키지 선택 메뉴가 나타난다.

서버를 멈추는 기본 핫키 조합은 **Ctrl + Alt + Backspace** 이다. 이 키 조합으로 잘못된 서버 설정에 의한 하드웨어 손상을 막을 수 있다.

기본으로 설정된 해상도를 변경하려면 X 가 동작하는 동안 핫키 조합 **Ctrl+Alt++** 또는 **Ctrl+Alt+-** 로 해상도를 변경할 수 있다.

**XFree86** 을 실행하는 동안 **xvidtune** 을 사용하여 화면의 높이, 넓이 또는 중앙으로 위치를 조정할 수 있다.

---

적절하지 않은 설정은 장비에 부담을 줄수 있으므로 주의한다. 걱정된다면 설정하지 않고 X 윈도우 화면 조정은 모니터에서 조정한다. 텍스트 모드로 다시 변경할 때 화면이 달라질 수 있지만 장비에 부담을 주는것보다는 낫다.

어떤 설정이든 조정하기 전에 xvidtune(1) 매뉴얼 페이지를 읽는다.

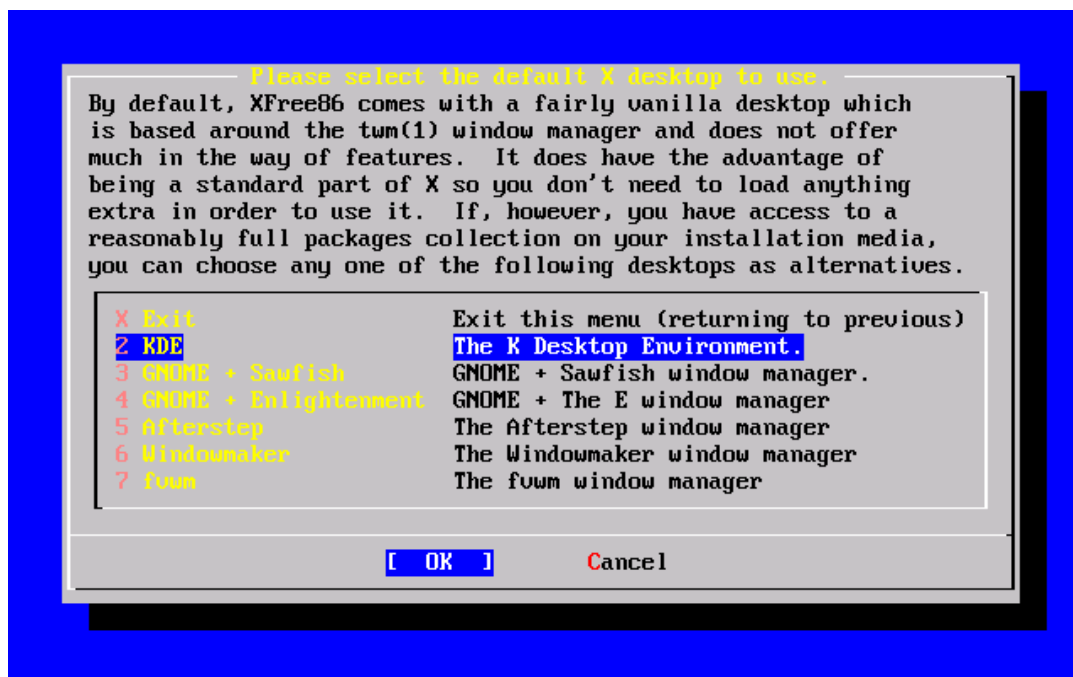
XFree86 설정이 끝나면 기본 데스크톱 선택으로 넘어간다.

### 2.9.13 기본 X 데스크톱 선택

다양한 윈도우 매니저를 사용할 수 있다. 이들은 매우 기본적인 환경부터 커다란 소프트웨어 스위트의 완벽한 데스크톱 환경까지 제공한다. 어떤 것은 최소한의 디스크 공간과 적은 메모리가 필요하지만 다양한 기능을 가진 매니저는 더 많은 자원이 요구된다. 가장 적절한 데스크톱 환경을 결정하는 가장 좋은 방법은 몇 가지 다른 환경을 사용해 보는 것이다. 이들은 설치가 끝나고 포트 컬렉션이나 패키지에서 추가하여 사용할 수 있다.

기본 데스크톱으로 유명한 데스크톱 중 하나를 선택하여 설치하고 설정할 수 있다. 이것은 설치 후 바로 시작된다.

그림 2-56. 기본 데스크톱 선택



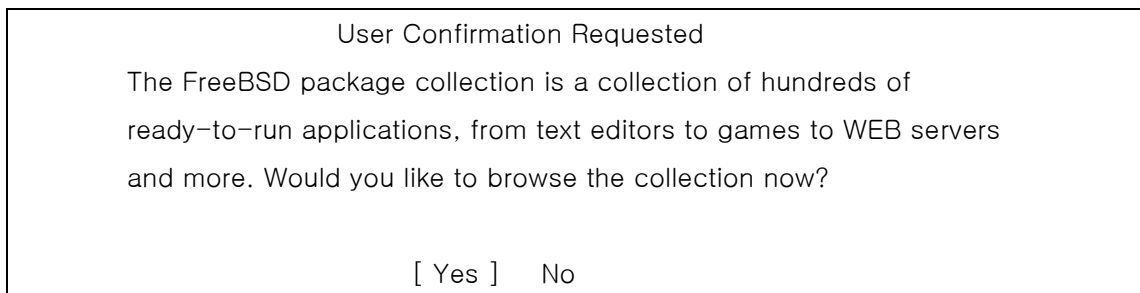
---

원하는 데스크톱을 선택하고 **Enter** 를 누르면 선택한 데스크톱 설치가 진행된다.

## 2.9.14 패키지 설치

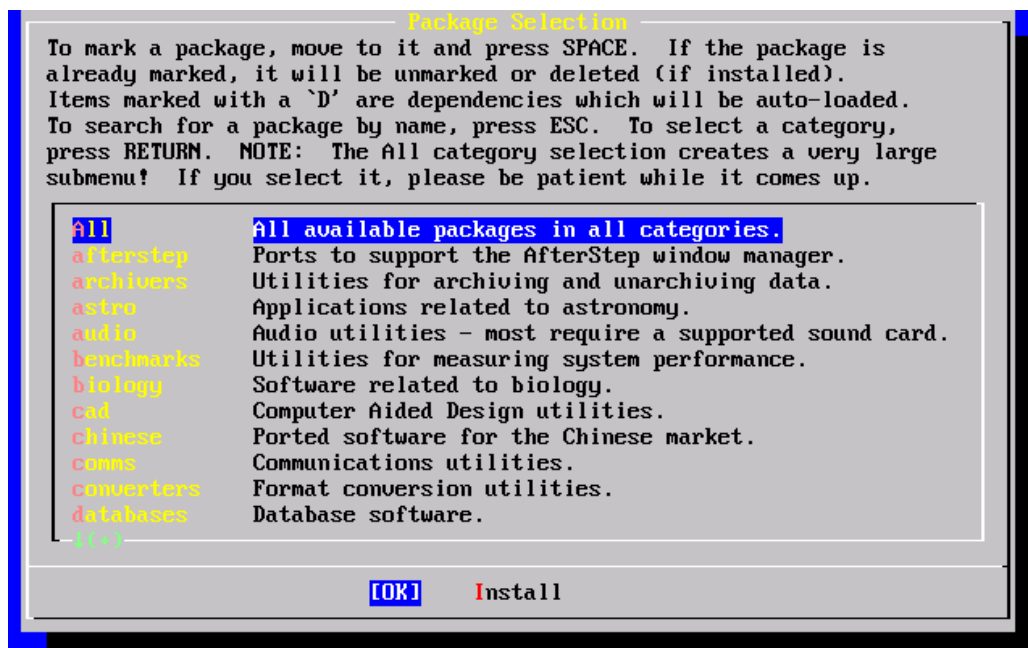
패키지는 이미 컴파일된 바이너리로 소프트웨어를 설치할 때 편리하다.

설명을 위해 하나의 패키지 설치를 보여주겠다. 원한다면 다른 패키지도 여기서 추가할 수 있다. 설치가 끝나고 패키지를 추가할 때도 `/stand/sysinstall` 을 사용할 수 있다.



[Yes]를 선택하고 **Enter** 를 누르면 패키지 선택 화면이 나타난다.

그림 2-57. 패키지 카테고리 선택

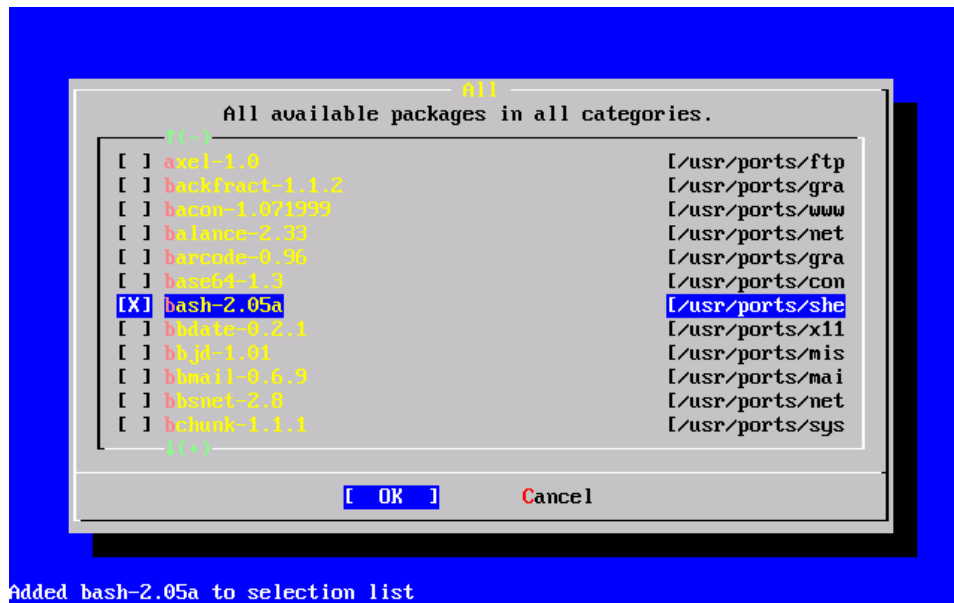


---

All 을 선택했다면 사용할 수 있는 모든 패키지가 표시되지만 특정 카테고리를 선택할 수 있다. 방향키로 선택하고 **Enter** 를 누른다.

메뉴는 선택할 수 있는 모든 패키지를 보여준다.

그림 2-58. 패키지 선택



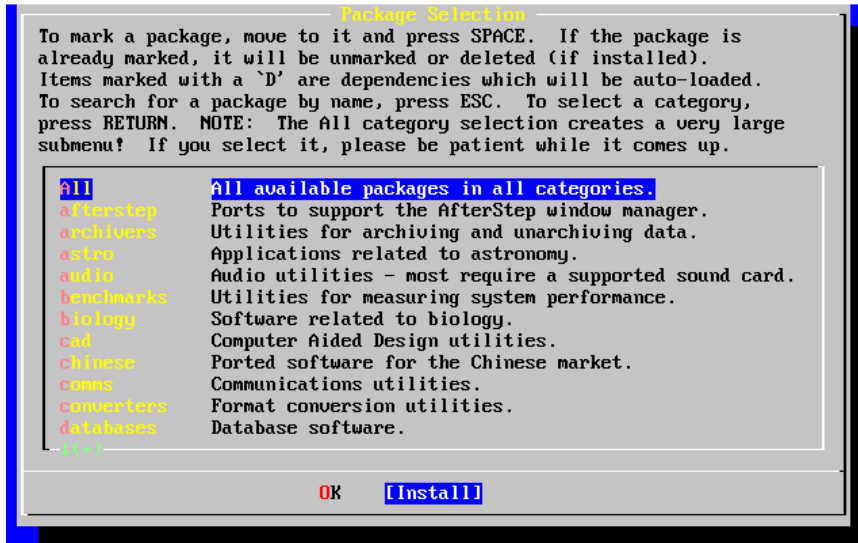
**bash** 셸을 선택 했다. **Space** 키를 눌러서 원하는 패키지 범위를 선택할 수 있다. 각 패키지에 대한 짧은 설명이 화면의 왼쪽 하단에 나타난다.

**Tab** 키를 누르면 마지막으로 선택한 패키지 사이에서 [OK]와 [Cancel]를 이동할 수 있다.

설치 하려는 패키지 선택이 끝나면 **Tab** 을 눌러 [OK]로 이동하고 **Enter** 를 눌러서 패키지 선택 메뉴로 다시 돌아올 수 있다.

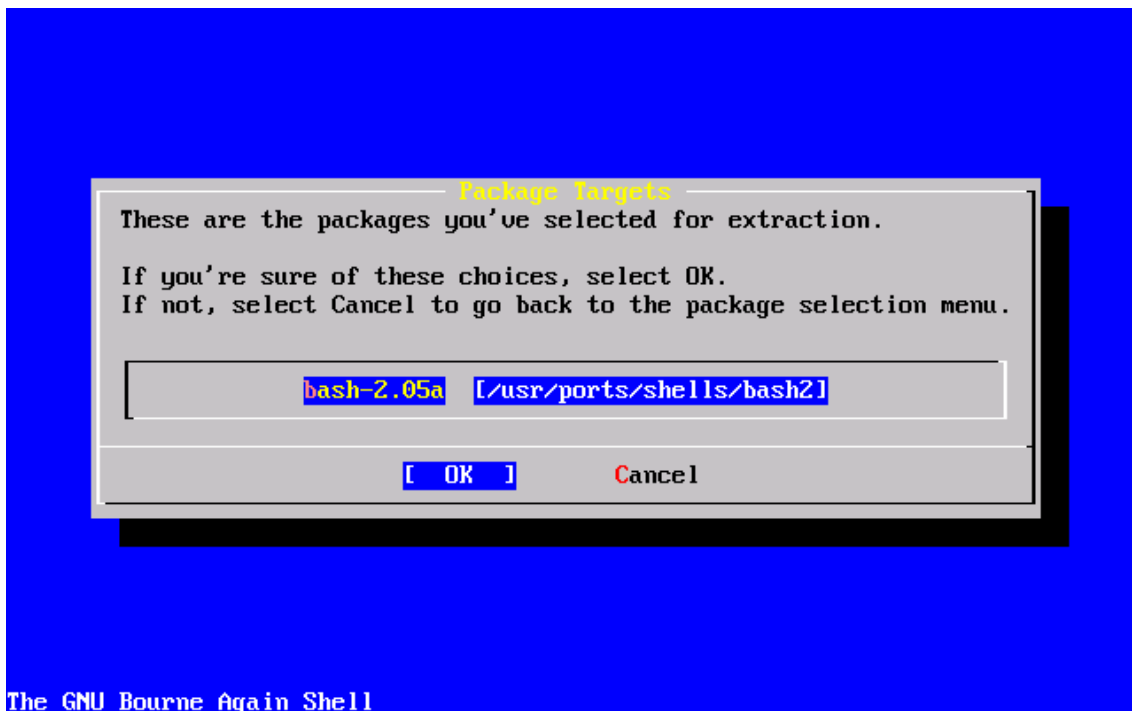
왼쪽과 오른쪽 방향키로도 [OK]와 [Cancel] 사이를 움직일 수 있다.

그림 2-59. 패키지 설치



[Install]를 선택하고 Enter를 누르면 설치하려는 패키지를 확인한다.

그림 2-60. 패키지 설치 확인



[OK]를 선택하고 Enter를 누르면 패키지 설치가 시작된다. 설치 메시지는 끝날 때까지 나타나고 에러 메시지가 있다면 적어둔다.

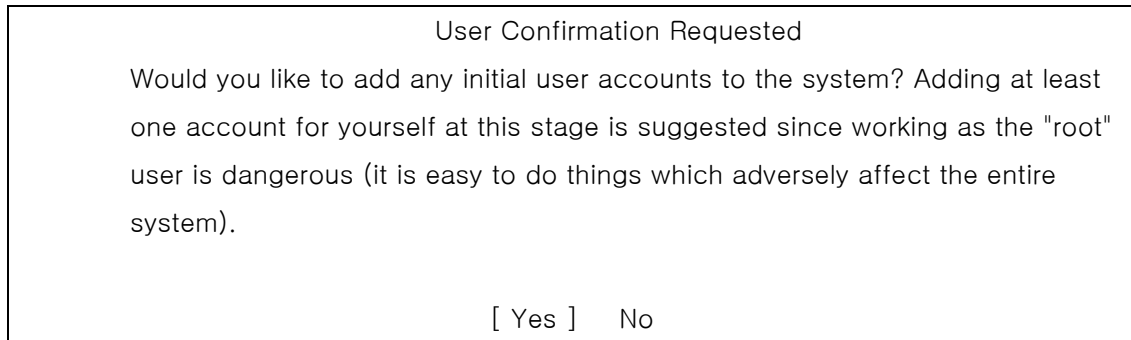


---

패키지 설치가 끝난 후 마지막 설정이 계속 된다. 어떤 패키지도 선택하지않고 다시 마지막 설정으로 되돌아 가더라도 install 을 선택해야 된다.

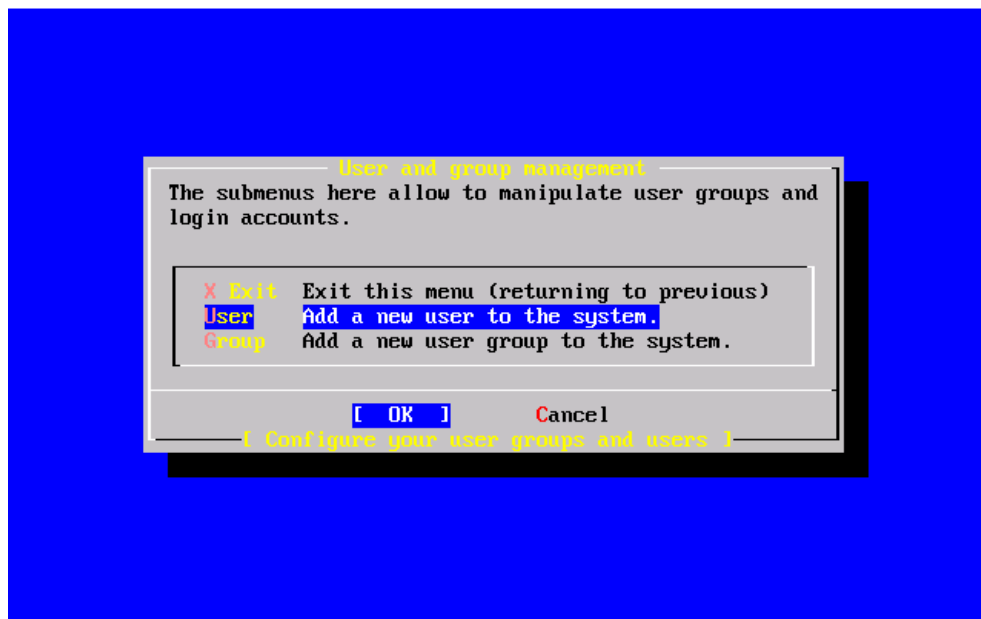
## 2.9.14 유저/그룹 추가

설치 중 최소한 한 명의 유저를 추가하여 root 로 로그인 하지않고 시스템을 사용할 수 있다. root 파티션은 보통 크기가 작기 때문에 root 로 어플리케이션을 실행하면 쉽게 파티션을 채울 수 있다.



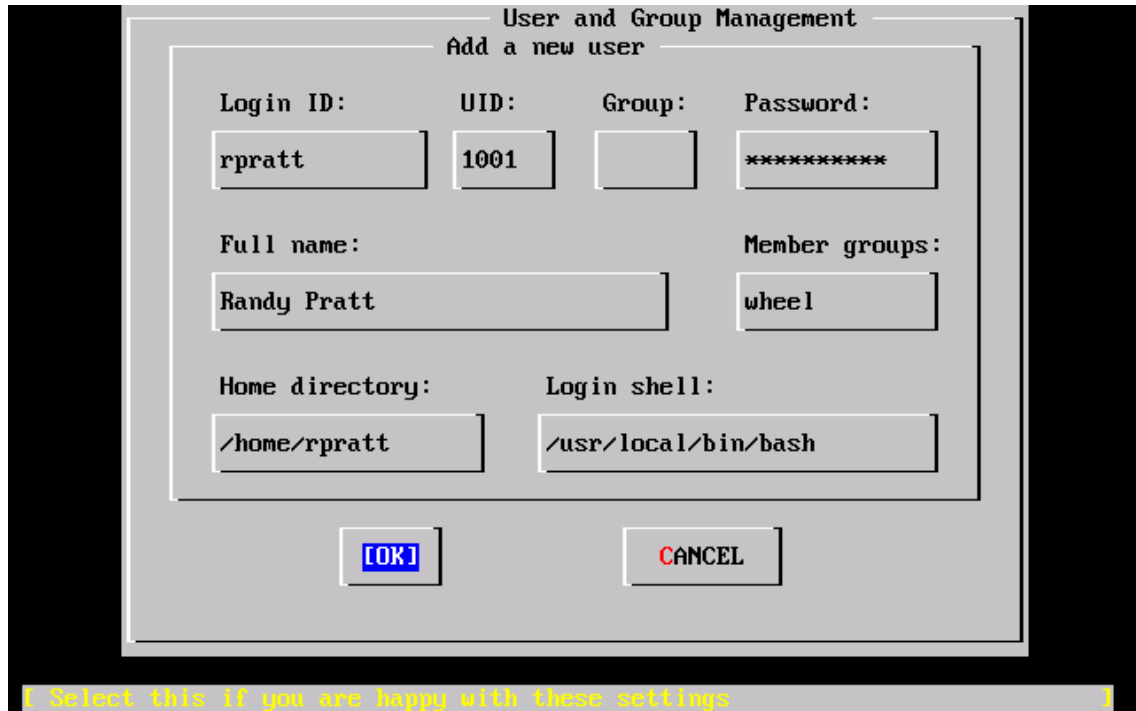
[Yes]를 선택하고 **Enter** 를 눌러 유저 추가를 계속한다.

그림 2-61. 유저 선택



방향키로 User 를 선택하고 Enter 를 누른다.

그림 2-62. 유저 정보 추가



Tab 키로 선택한 아이템에 대한 필요한 정보를 입력하는데 도움이 되도록 화면 하단에 설명이 나타난다.

#### Login ID

새로운 유저의 로그인 이름(필수).

#### UID

유저의 숫자 ID(자동 선택을 위해 공란으로 둔다).

#### Group

유저의 로그인 그룹 이름(자동 선택을 위해 공란으로 둔다).

---

**Password**

유저의 패스워드(주의 깊게 이 필드를 입력한다).

**Full name**

유저의 진짜 이름(설명을 입력한다).

**Member groups**

유저가 속한 그룹(접근 권한을 위해).

**Home directory**

유저의 홈 디렉터리(기본값을 위해 공란으로 둔다).

**Login shell**

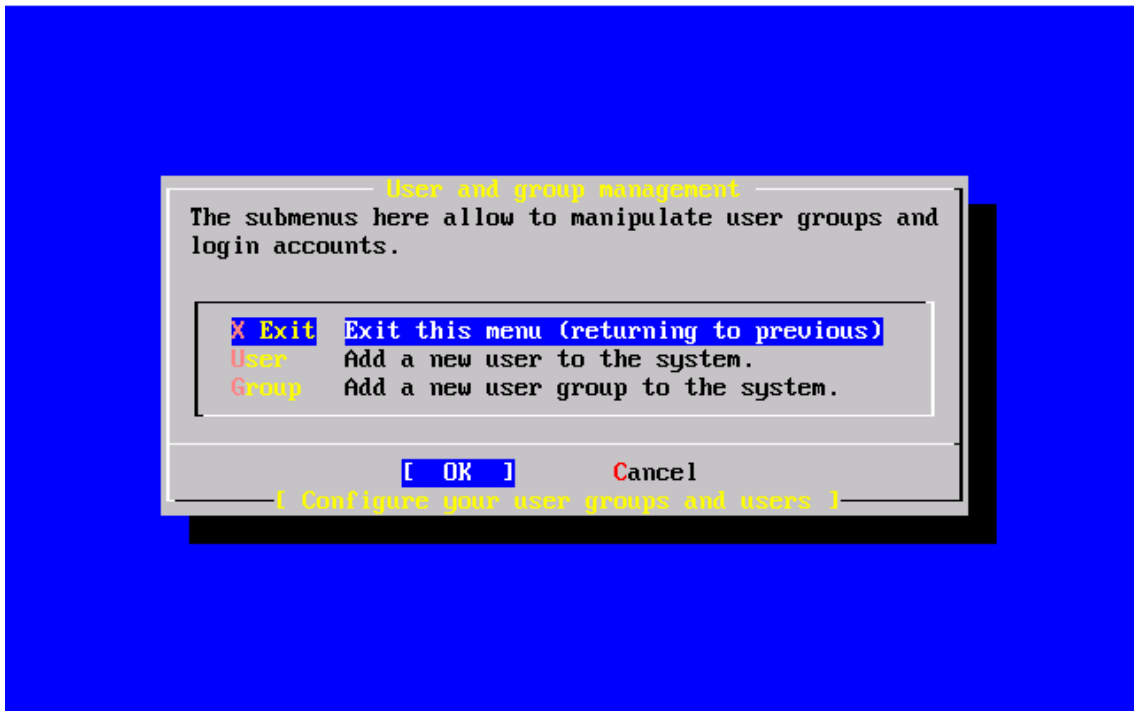
유저의 로그인 셸(기본값을 위해 공란으로 둔다. 예: /bin/sh).

로그인 셸은 패키지에서 미리 설치한 **bash** 셸을 사용하기 위해 /bin/sh 에서 /usr/local/bin/bash 로 변경 하였다. 설치되지 않은 셸을 사용한다면 로그인 할수 없다. BSD-세계에서 가장 유명한 셸은 /bin/tcsh 로 표시하는 C 셸이다.

그리고 유저가 root 권한으로 슈퍼 유저가 될수 있도록 wheel 그룹에 추가하였다.

설정이 적당하여 [OK]를 누르면 유저와 그룹 관리 메뉴가 다시 표시된다.

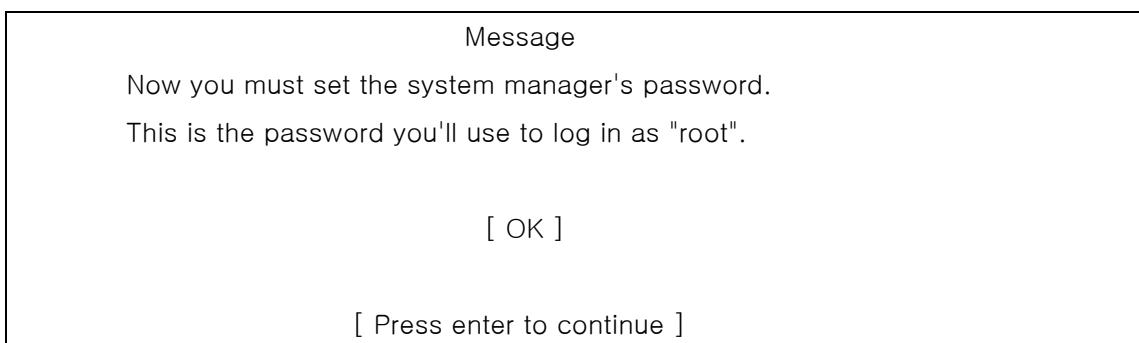
그림 2-63. 유저와 그룹 관리 나가기



그룹도 필요하다면 여기서 추가할 수 있다. 그렇지 않고 설치를 끝난 후 /stand/sysinstall 로 추가할 수 있다.

유저 추가를 끝내고 방향키로 Exit 를 선택한 후 **Enter** 를 눌러 설치를 계속한다.

## 2.9.16 root 패스워드 지정



**Enter** 를 눌러 root 패스워드를 설정한다.

패스워드는 두 번을 정확하게 입력한다. 추가적으로 패스워드를 잊었을때 찾을 수 있는 방

---

법이 있어야 한다.

Changing local password for root.

New password :

Retype new password :

패스워드를 성공적으로 입력한 후 설치는 계속된다.

## 2.9.16 설치 나가기

추가적으로 네트워크 장치 설정이 필요하거나 다른 설정이 필요하다면 이곳에서 또는 설치가 끝나고 /stand/sysinstall 에서 설정할 수 있다.

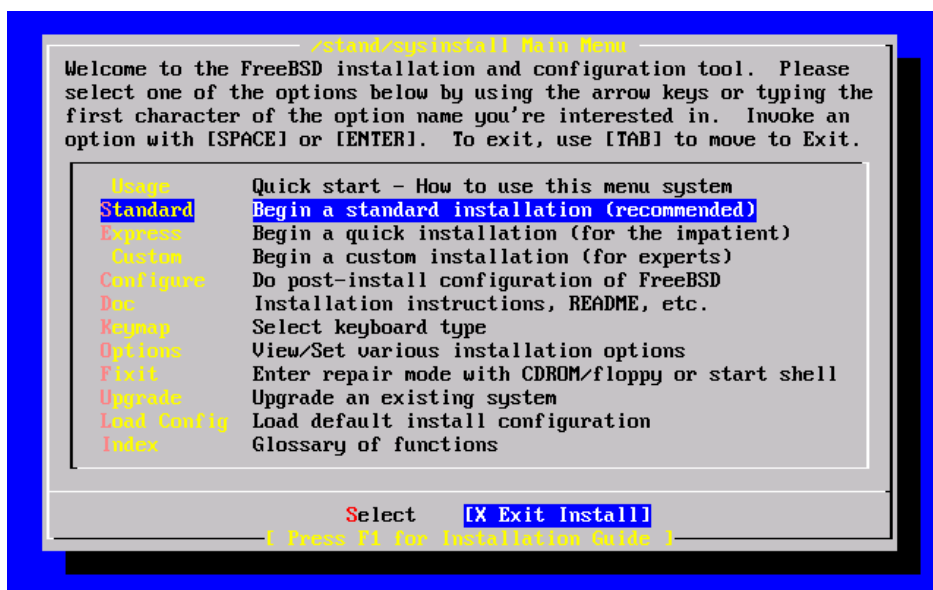
User Confirmation Requested

Visit the general configuration menu for a chance to set any last options?

Yes [ No ]

[No]를 선택하고 Enter 를 누르면 메인 설치 메뉴로 돌아간다.

그림 2-64. 설치 끝내기



---

[X Exit Install]를 선택하고 **Enter** 를 누르면 설치를 나가기 위해 확인을 묻는다.

User Confirmation Requested

Are you sure you wish to exit? The system will reboot (be sure to remove any floppies from the drives).

[ Yes ] No

[Yes]를 선택하고 플로피로 부팅했다면 플로피를 빼낸다. CDROM 드라이브는 머신이 재부팅할 때까지 잠겨있다. CDROM 드라이브 잠김이 풀리면 드라이브에서 꺼낼 수 있다(빨리).

## 2.9.18 FreeBSD 부팅

### 2.9.18.1 i386 에서 FreeBSD 부팅

모든 것이 정상이면 화면위를 지나가는 메시지를 볼수 있고 로그인 프롬프트에 도달한다. **Scroll-Lock** 를 누르고 **PgUp** 과 **PgDn** 으로 메시지의 내용을 볼수 있다. **Scroll-Lock** 을 다시 누르면 프롬프트로 돌아온다.

전체적인 메시지는 화면에 나타나지 않지만(버퍼의 한계로) 로그인 후 프롬프트에서 `dmesg` 를 입력하여 명령어 라인으로 메시지를 볼수 있다.

설치할 때 지정한 유저 이름과 패스워드로 로그인한다(이 예제에서는 `rpratt`). 필요한 경우를 제외하고 `root` 로그인 하는것은 가급적 피한다.

다음은 전형적인 부트 메시지다(버전 정보 생략):

Copyright (c) 1992-2002 The FreeBSD Project.  
Copyright (c) 1979, 1980, 1983, 1986, 1988, 1989, 1991, 1992, 1993, 1994  
The Regents of the University of California. All rights reserved.

Timecounter "i8254" frequency 1193182 Hz

---

```
CPU: AMD-K6(tm) 3D processor (300.68-MHz 586-class CPU)
  Origin = "AuthenticAMD"  Id = 0x580  Stepping = 0
  Features=0x8001bf<FPU,VME,DE,PSE,TSC,MSR,MCE,CX8,MMX>
  AMD Features=0x80000800<SYSCALL,3DNow!>
real memory  = 268435456 (262144K bytes)
config> di sn0
config> di lnc0
config> di le0
config> di ie0
config> di fe0
config> di cs0
config> di bt0
config> di aic0
config> di aha0
config> di adv0
config> q
avail memory = 256311296 (250304K bytes)
Preloaded elf kernel "kernel" at 0xc0491000.
Preloaded userconfig_script "/boot/kernel.conf" at 0xc049109c.
md0: Malloc disk
Using $PIR table, 4 entries at 0xc00fde60
npx0: <math processor> on motherboard
npx0: INT 16 interface
pcib0: <Host to PCI bridge> on motherboard
pci0: <PCI bus> on pcib0
pcib1: <VIA 82C598MVP (Apollo MVP3) PCI-PCI (AGP) bridge> at device 1.0 on pci0
pci1: <PCI bus> on pcib1
pci1: <Matrox MGA G200 AGP graphics accelerator> at 0.0 irq 11
isab0: <VIA 82C586 PCI-ISA bridge> at device 7.0 on pci0
isa0: <ISA bus> on isab0
atapci0: <VIA 82C586 ATA33 controller> port 0xe000-0xe00f at device 7.1 on pci0
ata0: at 0x1f0 irq 14 on atapci0
ata1: at 0x170 irq 15 on atapci0
uhci0: <VIA 83C572 USB controller> port 0xe400-0xe41f irq 10 at device 7.2 on pci0
usb0: <VIA 83C572 USB controller> on uhci0
usb0: USB revision 1.0
```

---

uhub0: VIA UHCI root hub, class 9/0, rev 1.00/1.00, addr 1  
uhub0: 2 ports with 2 removable, self powered  
chip1: <VIA 82C586B ACPI interface> at device 7.3 on pci0  
ed0: <NE2000 PCI Ethernet (RealTek 8029)> port 0xe800-0xe81f irq 9 at  
device 10.0 on pci0  
ed0: address 52:54:05:de:73:1b, type NE2000 (16 bit)  
isa0: too many dependant configs (8)  
isa0: unexpected small tag 14  
fdc0: <NEC 72065B or clone> at port 0x3f0-0x3f5,0x3f7 irq 6 drq 2 on isa0  
fdc0: FIFO enabled, 8 bytes threshold  
fd0: <1440-KB 3.5" drive> on fdc0 drive 0  
atkbd0: <keyboard controller (i8042)> at port 0x60-0x64 on isa0  
atkbd0: <AT Keyboard> flags 0x1 irq 1 on atkbd0  
kbd0 at atkbd0  
psm0: <PS/2 Mouse> irq 12 on atkbd0  
psm0: model Generic PS/2 mouse, device ID 0  
vga0: <Generic ISA VGA> at port 0x3c0-0x3df iomem 0xa0000-0xbffff on isa0  
sc0: <System console> at flags 0x1 on isa0  
sc0: VGA <16 virtual consoles, flags=0x300>  
sio0 at port 0x3f8-0x3ff irq 4 flags 0x10 on isa0  
sio0: type 16550A  
sio1 at port 0x2f8-0x2ff irq 3 on isa0  
sio1: type 16550A  
ppc0: <Parallel port> at port 0x378-0x37f irq 7 on isa0  
ppc0: SMC-like chipset (ECP/EPP/PS2/NIBBLE) in COMPATIBLE mode  
ppc0: FIFO with 16/16/15 bytes threshold  
ppbus0: IEEE1284 device found /NIBBLE  
Probing for PnP devices on ppbus0:  
plip0: <PLIP network interface> on ppbus0  
lpt0: <Printer> on ppbus0  
lpt0: Interrupt-driven port  
ppi0: <Parallel I/O> on ppbus0  
ad0: 8063MB <IBM-DHEA-38451> [16383/16/63] at ata0-master using UDMA33  
ad2: 8063MB <IBM-DHEA-38451> [16383/16/63] at ata1-master using UDMA33  
acd0: CDROM <DELTA OTC-H101/ST3 F/W by OIPD> at ata0-slave using PIO4  
Mounting root from ufs:/dev/ad0s1a



---

```
swapon: adding /dev/ad0s1b as swap device
Automatic boot in progress...
/dev/ad0s1a: FILESYSTEM CLEAN; SKIPPING CHECKS
/dev/ad0s1a: clean, 48752 free (552 frags, 6025 blocks, 0.9% fragmentation)
/dev/ad0s1f: FILESYSTEM CLEAN; SKIPPING CHECKS
/dev/ad0s1f: clean, 128997 free (21 frags, 16122 blocks, 0.0% fragmentation)
/dev/ad0s1g: FILESYSTEM CLEAN; SKIPPING CHECKS
/dev/ad0s1g: clean, 3036299 free (43175 frags, 374073 blocks, 1.3% fragmentation)
/dev/ad0s1e: filesystem CLEAN; SKIPPING CHECKS
/dev/ad0s1e: clean, 128193 free (17 frags, 16022 blocks, 0.0% fragmentation)
Doing initial network setup: hostname.
ed0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
    inet 192.168.0.1 netmask 0xfffff00 broadcast 192.168.0.255
    inet6 fe80::5054::5ff::fede:731b%ed0 prefixlen 64 tentative scopeid 0x1
    ether 52:54:05:de:73:1b
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
    inet6 fe80::1%lo0 prefixlen 64 scopeid 0x8
    inet6 ::1 prefixlen 128
    inet 127.0.0.1 netmask 0xff000000
Additional routing options: IP gateway=YES TCP keepalive=YES
routing daemons:.
additional daemons: syslogd.
Doing additional network setup:.
Starting final network daemons: creating ssh RSA host key
Generating public/private rsa1 key pair.
Your identification has been saved in /etc/ssh/ssh_host_key.
Your public key has been saved in /etc/ssh/ssh_host_key.pub.
The key fingerprint is:
cd:76:89:16:69:0e:d0:6e:f8:66:d0:07:26:3c:7e:2d root@k6-2.example.com
    creating ssh DSA host key
Generating public/private dsa key pair.
Your identification has been saved in /etc/ssh/ssh_host_dsa_key.
Your public key has been saved in /etc/ssh/ssh_host_dsa_key.pub.
The key fingerprint is:
f9:a1:a9:47:c4:ad:f9:8d:52:b8:b8:ff:8c:ad:2d:e6 root@k6-2.example.com.
setting ELF ldconfig path: /usr/lib /usr/lib/compat /usr/X11R6/lib
```

---

```
/usr/local/lib
a.out ldconfig path: /usr/lib/aout /usr/lib/compat/aout /usr/X11R6/lib/aout
starting standard daemons: inetd cron sshd usbd sendmail.
Initial rc.i386 initialization:.
rc.i386 configuring syscons: blank_time screensaver moused.
Additional ABI support: linux.
Local package initialization:.
Additional TCP options:.

FreeBSD/i386 (k6-2.example.com) (ttyv0)

login: rpratt
Password:
```

RSA 와 DSA 키를 생성할 때 머신이 느리면 약간의 시간이 더 필요하다. 이 현상은 새로 설치한 머신이 최초에 부팅할 때만 나타난다. 다음에 부팅할 때는 더 빠르다.

X 서버를 설정하고 기본 데스크톱을 선택했다면 명령어 라인에 **startx** 를 입력해서 X 윈도우를 시작할 수 있다.

## 2.9.18.2 Alpha 머신에서 FreeBSD 부팅

설치 절차가 끝나면 SRM 프롬프트에 다음과 같이 입력하여 FreeBSD 를 시작할 수 있다.

```
>>>BOOT DKC0
```

이 명령은 Firmware 에게 특정 디스크로 부팅할 것을 명령한다. 앞으로 FreeBSD 를 자동으로 부팅하려면 다음 명령을 사용한다:

```
>>> SET BOOT_OSFLAGS A
>>> SET BOOT_FILE"
>>> SET BOOTDEF_DEV DKC0
>>> SET AUTO_ACTION BOOT
```

---

부트 메시지는 i386 에서 FreeBSD 가 부팅하는 것과 비슷한 메시지를 보여준다.

## 2.9.19 FreeBSD 셧다운

운영체제를 정확히 셧다운하는것도 중요하다. 절대 파워를 그냥 끄지 않는다. 명령어 라인에서 su 를 입력하고 root 패스워드를 입력하여 슈퍼 유저가 된다. 이 명령은 wheel 그룹의 유저만 할 수 있다. 그렇지 않으면 root 로 로그인해서 **shutdown -h now** 를 사용한다.

The operating system has halted.  
Please press any key to reboot.

셧다운 명령이 "Please press any key to reboot" 메시지를 나타낸 후 안전하게 전원을 끄다. 전원 스위치를 끄는 대신 아무 키나 누르면 시스템은 다시 부팅한다.

그리고 **Ctrl+Alt+Del** 키 조합으로 시스템을 재 부팅할 수 있지만 이것은 일반적으로 운용할 때 권장하지 않는다.

## 2.10 지원되는 하드웨어

FreeBSD 는 현재 다양한 종류의 ISA, VLB, EISA 와 수많은 컴팩 Alpha 프로세서 머신과 인텔, AMD, Cyrix 나 NexGen"x86" 프로세서의 PCI 버스 기반 PC 에서 사용된다. 일반적인 IDE 와 ESDI 드라이브 설정, 다양한 SCSI 컨트롤러, PCMCIA 카드, USB 드라이브가 지원되고 네트워크와 시리얼 카드도 지원된다. FreeBSD 는 IBM 의 마이크로 채널(MCA) 버스도 지원하고 있다.

지원되는 하드웨어 리스트는 각 FreeBSD 릴리즈와 FreeBSD 하드웨어 노트로 제공된다. 이 문서는 보통 CDROM 이나 FTP 배포본의 최 상단 디렉터리의 HARDWARE.TXT 라는 파일 이름이나 **sysinstall** 의 문서 메뉴에서 찾을 수 있다. 이곳에는 아키텍처의 어떤 하드웨어 장치가 FreeBSD 의 각 릴리즈에 의해 지원되고 있는지 나열되어 있다. 다양한 릴리즈와 아키텍처에 지원되는 하드웨어 리스트를 FreeBSD 웹사이트의 릴리즈 정보 페이지에서 (<http://www.FreeBSD.org/releases/index.html>) 복사할 수 있다.

---

## 2.11 문제 해결

이번 섹션은 사람들이 보고하는 일반적인 문제처럼 기본적인 설치 문제를 다룬다. 그리고 FreeBSD 와 MS-DOS 를 듀얼 부팅하려는 사람들을 위한 몇 개의 질문과 답변이 있다.

### 2.11.1 문제가 있다면 어떻게 해야되는가?

PC 아키텍처의 다양한 한계로 100% 해결하는 것은 불가능 하지만 문제가 있을때 체크할 수 있는 몇 가지가 있다.

여러분의 FreeBSD 버전에서 하드웨어가 지원되고 있는지 하드웨어 노트를 체크한다.

하드웨어는 지원되지만 여전히 다른 문제가 있다면 컴퓨터를 재 부팅하고 **visual kernel configuration** 옵션이 나타나면 이것을 선택한다. 이 옵션으로 하드웨어 설정에서 이 문제와 관련있는 정보를 시스템에 제공한다. 부트 디스크의 커널은 대부분 하드웨어 장치의 IRQ, IO 주소와 DMA 채널이 공장 기본으로 설정되어 있다고 가정한다. 하드웨어 설정이 변경되었다면 설정 편집기를 사용하여 설정 정보를 찾을 수 있는 위치를 FreeBSD 에게 알려줘야 된다.

**주의:** 설치 중 디스플레이(sc0)처럼 필요한 드라이버를 비활성하지 않는다. 설정 에디터를 끝낸 후 설치가 옳히거나 이유를 모른채 실패한다면, 삭제하거나 변경하지 말아야 될 것을 삭제하거나 변경했을 것이다. 재 부팅하고 다시 시도한다.

새로운 장치를 탐색하면서 설치는 되어있지만 설정에 실패한 장치를 다시 탐색하게 된다. 이 경우 충돌되는 드라이버를 비활성하고 탐색한다.

**Note:** 어떤 설치 문제는 다양한 하드웨어 컴포넌트, 특히 메인보드의 펌웨어를 업데이트하여 피하거나 줄일 수 있다. 대부분 메인보드나 컴퓨터 제조업자의 웹사이트에서 BIOS 로 알려져있는 메인보드 펌웨어의 업그레이드 정보를 찾을 수 있다.

대부분의 제조사에서 특별한 이유가 없다면 메인보드 BIOS 를 업그레이드하지 말 것을 강력하게 권고한다. 업그레이드 절차가 문제를 야기해서 BIOS 칩에 영구적인

---

피해를 줄수 있기 때문이다.

설정 모드에서 다음과 같은 사항을 할 수 있다:

- 커널에 설치되어있는 장치 드라이버 리스트를 볼수 있다.
- 시스템에 없는 하드웨어 장치 드라이버를 비활성한다.
- 장치 드라이버가 사용하는 IRQ, DRQ, IO 포트 주소를 변경한다.

하드웨어 설정에 맞게 커널을 조정한 후 Q를 눌러 새로운 설정으로 부팅한다. 설치 과정이 끝나면 설정 모드에서 변경한것은 영구적으로 저장되기 때문에 부팅할 때마다 다시 설정할 필요가 없다. 그래도 사용자 커널을 빌드하는 것이 좋다.

## 2.11.2 MS-DOS 에서 설치하기

많은 유저가 Microsoft 기반의 운영체제가 설치되어있는 PC 에서 FreeBSD 를 설치하려고 한다. 이런 이유로 FreeBSD 는 **fips** 라는 유틸리티를 가지고있다. 이 유틸리티는 CD-ROM 의 tools 디렉터리에서 찾을 수 있거나 여러 FreeBSD 미러 사이트 중 한곳에서 다운로드 할 수 있다.

**FIPS** 유틸리티는 MS-DOS 파티션을 두 조각으로 나누어 기존 파티션은 보존하고 두 번째 여유 공간에 FreeBSD 를 설치할 수 있다. 첫째 MS-DOS 파티션을 Windows 의 **Disk Defragmenter** 유틸리티(익스플로러의 하드 드라이브에서 마우스 오른쪽을 클릭하고 하드 드라이브 조각 모음을한다)나 **Norton Disk Tools** 를 사용하여 조각 모음을한다. 이제 **FIPS** 유틸리티를 실행할 수 있다. 나머지 정보를 보여 주기 때문에 단지 화면의 지시를 따르면된다. 나중에 재 부팅해서 새로운 여유공간에 FreeBSD 를 설치한다. 원하는 설치 종류에따라 얼마나 많은 디스크 공간이 필요한지 배포본 메뉴를 확인한다.

PowerQuest(<http://www.powerquest.com>)의 **Partition Magic**이라는 매우 유용한 제품도 있다. 이 어플리케이션은 **FIPS** 보다 많은 기능을 가지고 있기 때문에 자주 운영체제를 추가/삭제할 계획이라면 강력히 추천한다. 구입해야 되므로 FreeBSD를 설치하고 유지하고자 한다면 **FIPS**가 유용할 것이다.

---

### 2.11.3 MS-DOS 파일시스템 사용

FreeBSD 는 아직 **Double Space** 어플리케이션으로 압축된 파일시스템을 지원하지 않는다. 그래서 이런 파일시스템은 FreeBSD 를 설치하기 전에 압축을 풀어야 한다. 이것은 **시작 > 프로그램 > 시스템 도구** 메뉴에있는 압축 에이전트로 압축을 풀 수 있다.

FreeBSD 는 MS-DOS 기반 파일시스템을 지원한다. MS-DOS 파일시스템을 지원하려면 `mount_msdos(8)`(FreeBSD 5.X 에서 이 명령은 `mount_msdosfs(8)`다) 필요한 매개변수를 사용해야 된다. 이 유틸리티의 일반적인 사용법은 다음과 같다.

```
# moutn_msdos /dev/ad0s1 /mnt
```

이 예제에서 MS-DOS 파일시스템은 주 하드 디스크의 첫 번째 파티션에 있다. 여러분의 상황은 다를 것이기 때문에 `dmesg` 와 `mount` 명령의 결과를 체크한다. 이 명령으로 파티션 레이아웃 정보를 충분히 볼수 있다.

**Note:** 확장된 MS-DOS 파일시스템은 보통 FreeBSD 파티션 다음에 생성된다. 다시 말하면 슬라이스 번호는 FreeBSD 가 사용하는 슬라이스 보다 하나 더 높을 것이다. 예를 들어 첫 번째 MS-DOS 파티션이 `/dev/ad0s1` 이면 FreeBSD 파티션이 `/dev/ad0s2` 이고 확장 MS-DOS 파티션이 `/dev/ad0s3` 에 있을 것이다. 처음에는 약간 혼란스러울 수 있다.

### 2.11.4 Alpha 유저의 질문과 답변

이번 섹션은 Alpha 시스템에 FreeBSD 를 설치하는 일반적인 질문의 답변이다.

#### 2.11.4.1 ARC 나 Alpha BIOS 콘솔에서 부팅할 수 있는가?

안된다, 컴팩의 Tru64 와 VMS 처럼 FreeBSD 는 SRM 콘솔에서만 부팅할 수 있다.

---

#### 2.11.4.2 설치에 필요한 여유공간이 없다. 첫째로 모든 것을 삭제해야 되는가?

불행히도 삭제해야 된다.

#### 2.11.4.3 컴팩 Tru64 나 VMS 파일시스템을 마운트할 수 있는가?

아직은 마운트할 수 없다.

## 2.12 고급 설치 가이드

이번 섹션은 예외적인 경우에 FreeBSD 를 어떻게 설치하는지 설명한다.

### 2.12.1 모니터나 키보드가 없는 시스템에 FreeBSD 설치하기

FreeBSD 를 설치하려는 머신에 모니터나 VGA 가 없기 때문에 이런 종류의 설치를 “headless install”이라고 부른다. 이런 설치가 어떻게 가능한지 묻는다면? 시리얼 콘솔을 사용하는 방법이다. 시리얼 콘솔은 시스템의 메인 화면과 키보드처럼 동작하는 근본적으로 다른 머신을 사용한다. 이렇게 설치하려면 다음 단계를 따르면 된다.

#### [Headless Install]

- ① 적당한 부트 플로피 이미지를 받는다.

적당한 디스크 이미지를 받아서 설치 프로그램으로 부팅할 수 있다. 부트 로더에게 VGA 장치로 콘솔 출력을 보내고 로컬 키보드에서 입력을 받는 대신 시리얼

포트를 통해 I/O 를 보내도록 한다. 이 설정으로 충분하기 때문에 이제 디스크 이미지를 받는다.

**kern.flp**는 <ftp://ftp.freebsd.org/pub/FreeBSD/releases/i386/5.2.1-RELEASE/floppies/kern.flp>를 **mfsroot.flp**는 <ftp://ftp.freebsd.org/pub/FreeBSD/releases/i386/5.2.1-RELEASE/floppies/mfsroot.flp> 사이트의 플로피 디렉터리 (<ftp://ftp.freebsd.org/pub/FreeBSD/releases/i386/5.2.1-RELEASE/floppies/>)에서 구할 수 있다.

## ② 플로피 디스크에 이미지 파일 작성하기

**kern.flp** 같은 이미지 파일은 디스크에 복사할 수 있는 일반적인 파일이 아니다. 이 파일들은 완벽한 디스크 이미지이다.

이 의미는 DOS 의 `copy` 같은 명령으로 파일을 작성할 수 없고 이미지를 디스크에 직접 작성하는 특정 툴을 사용해야 된다.

DOS 를 사용할 수 있는 컴퓨터에서 부팅 플로피를 만든다면 `fdimage` 라는 툴이 제공된다.

E: 드라이브에있는 CDROM 에서 플로피를 만든다면 다음과 같이 실행한다.

```
E:W> tools\fdimage floppies\kern.flp A:
```

매번 플로피 디스크를 교체하면서 모든 .flp 파일에 이 명령을 반복한다. 필요하다면 .flp 파일의 위치에 따라 명령어 라인을 적절히 수정한다. CDROM 이 없다면 `fdimage` 는 FreeBSD FTP 사이트의 `tools` 디렉터리에서 (<ftp://ftp.FreeBSD.org/pub/FreeBSD/tools/>) 다운로드할 수 있다.

유닉스 시스템(다른 FreeBSD 시스템)에서 플로피를 작성한다면 이미지 파일을 디스크에 직접 작성하는 `dd(1)` 명령을 사용한다. 다음 명령을 FreeBSD 에서 실행한다.

```
# dd if=kern.flp of=/dev/fd0
```



---

FreeBSD 의 /dev/fd0 는 첫 번째 플로피 디스크(A: 드라이브)를 /dev/fd1 는 B:드라이브를 의미한다. 다른 유닉스들은 플로피 장치에 다른 이름을 사용하므로 필요하다면 그 시스템 문서를 참고한다.

③ 시리얼 콘솔로 부팅하기 위해 부트 플로피를 활성화한다

**주의:** 쓰기 방지되어있는 플로피는 마운트하지 않는다.

방금 만든 플로피로 부팅했다면 FreeBSD 는 일반적인 설치 모드로 부팅한다. FreeBSD 가 시리얼 콘솔로 부팅해야 되므로 mount(8) 명령으로 kern.flp 플로피를 FreeBSD 시스템에 마운트한다.

```
# mount /dev/fd0 /floppy
```

플로피가 마운트되었기 때문에 플로피 디렉터리로 변경한다.

```
# cd /floppy
```

이곳에서 시리얼 콘솔로 부팅하도록 플로피를 설정한다. /boot/loader -h 를 포함하고 있는 boot.config 라는 파일을 만들어야 한다. 부트 로더가 시리얼 콘솔로 부팅하도록 플래그가 설정된다.

```
# echo "/boot/loader -h" > boot.config
```

정확하게 플로피가 설정되었기 때문에 umount(8) 명령으로 플로피를 언마운트한다:

```
# cd /
```

```
# umount /mnt
```

이제 플로피 드라이브에서 플로피를 꺼낸다.

④ 널 모뎀 케이블 연결.

---

두 머신의 시리얼 포트에 널 모뎀 케이블로 연결한다. 여기서 일반적인 시리얼 케이블은 동작하지 않기 때문에 크로스가된 널 모뎀 케이블이 필요하다.

⑤ 설치하기 위한 부팅

**Kern.flp** 플로피를 설치하려는 headless 머신의 플로피 드라이브에 넣고 머신의 전원을 켜다.

⑥ Headless 머신에 연결

cu(1)로 머신에 연결한다.

```
# cu -l /dev/cuaa0
```

이제 cu 세션을 통해 headless 머신을 조정할 수 있다. 장비에서 **mfsroot.flp** 를 넣어달라고 요청한 후 어떤 종류의 터미널을 사용할 것인지 선택하는 화면이 나타난다. FreeBSD 컬러 콘솔을 선택하여 설치를 진행한다.

## 2.13 설치 미디어 준비

FreeBSD 설치 미디어나 소스가 필요하다. 이것은 테잎처럼 물리적인 미디어거나 FTP 사이트 또는 로컬 MS-DOS 파티션 같은 **sysinstall** 로 파일을 받을 수 있는 소스다.

**Note:** 반복을 피하기 위해 “FreeBSD 디스크”라는 문장의 의미는 구입하거나 만든 FreeBSD CDROM 이나 DVD 를 가지고 있다는 말이다.

예를 들면 다음과 같은 상황이 될 수 있다.

- 로컬 네트워크에 연결되어 있는 많은 머신과 FreeBSD 디스크가 하나있다.  
FreeBSD 디스크의 내용을 가지고있는 로컬 FTP 사이트를 생성해서 머신들을 인터

---

넷에 연결하지 않고 로컬 FTP 사이트를 사용한다.

- FreeBSD 디스크를 가지고 있고 FreeBSD 는 CD/DVD 드라이브를 감지하지 못하지만 DOS/윈도우에서는 인식한다. 같은 컴퓨터의 DOS 파티션에 FreeBSD 설치 파일을 복사하고 이 파일을 사용하여 FreeBSD 를 설치한다.
- 설치하려는 컴퓨터에는 CD/DVD 드라이브나 네트워크 카드가 없지만 “Laplinsk-style” 시리얼이나 패러럴 케이블을 연결할 수 있다.
- FreeBSD 설치에 사용할 수 있는 테잎을 만들려고 한다.

## 2.13.1 설치 CDROM 만들기

각 릴리즈 별로 FreeBSD 프로젝트는 5 개의 CDROM 이미지를 만든다(“ISO 이미지”). 이들 이미지는 CD 레코더를 가지고 있다면 CD 로 만들어서(구워서) FreeBSD 설치에 사용할 수 있다. CD 레코더가 있고 통신 이용료가 저렴하다면 FreeBSD 를 설치하는 최상의 방법이다.

- ① 정확한 ISO 이미지 다운로드.

각 릴리즈의 ISO 이미지는 <ftp://ftp.FreeBSD.org/pub/FreeBSD/ISO-IMAGES-arch/version> 또는 가까운 미러 사이트에서 다운로드할 수 있다. *Arch* 와 *version* 은 알맞게 고쳐서 사용한다.

이 디렉터리에는 보통 다음과 같은 이미지가 있다:

표 2-5. FreeBSD ISO 이미지 이름과 의미

파일 이름	내용
<i>version</i> -mini.iso	FreeBSD 설치에 필요하다.
<i>version</i> -disc1.iso	FreeBSD 설치에 필요한 것과 디스크에 맞는 다양한 추가 패키지.

<i>version-</i> disc2.iso	<b>sysinstall</b> 의 “Repair”와 관련하여 쉽게 사용할 수 있는 “live filesystem”. FreeBSD CVS 트리의 복사본이다. 디스크에 맞는 다양한 추가 패키지.
------------------------------	--

mini ISO 이미지나 disc 1 의 이미지를 다운로드 해야된다. disc 1 의 이미지는 mini ISO 이미지 전체를 가지고있기 때문에 두 가지를 다운로드 하지 않는다.

인터넷 요금이 저렴하다면 mini ISO 이미지를 사용한다. FreeBSD 를 설치하고 필요하다면 포트/패키지 시스템으로(4 장을 본다) 추가적인 패키지를 설치할 수 있다.

디스크에서 다른 패키지를 선택해야 된다면 disc 1 의 이미지를 사용한다.

## ② CD 굽기

디스크에 CD 이미지를 구워야한다. 다른 FreeBSD 시스템에서 이 작업을 하려면 16 장에서 자세한 정보를 본다.

다른 플랫폼에서 이 작업을 하려면 그 플랫폼에 CD 레코더를 조정할 수 있는 유틸리티가 있어야한다. 제공되는 이미지는 다양한 CD 레코딩 프로그램이 지원하는 표준 ISO 포맷이다.

## 2.13.2 FreeBSD 디스크로 로컬 FTP 사이트 만들기

FreeBSD 디스크는 FTP 사이트와 같은 구조를 가지고 있다. 따라서 로컬 FTP 사이트를 쉽게 생성하여 FreeBSD 를 설치할때 네트워크의 다른 머신이 사용할 수 있다.

### [FreeBSD 디스크로 FTP 사이트 만들기]

- ① FTP 사이트가될 FreeBSD 컴퓨터의 드라이브에 CDROM 을 넣고 /cdrom 에 마운트한다.

```
# mount /cdrom
```

- 
- ② /etc/passwd 에 익명 FTP 사용자 계정을 추가한다. Vipw(8)로 /etc/passwd 를 편집하고 다음 라인을 추가한다.

```
ftp:*:99:99::0:0:FTP:/cdrom:/nonexistent
```

- ③ /etc/inetd.conf 에서 FTP 서비스를 활성화한다.

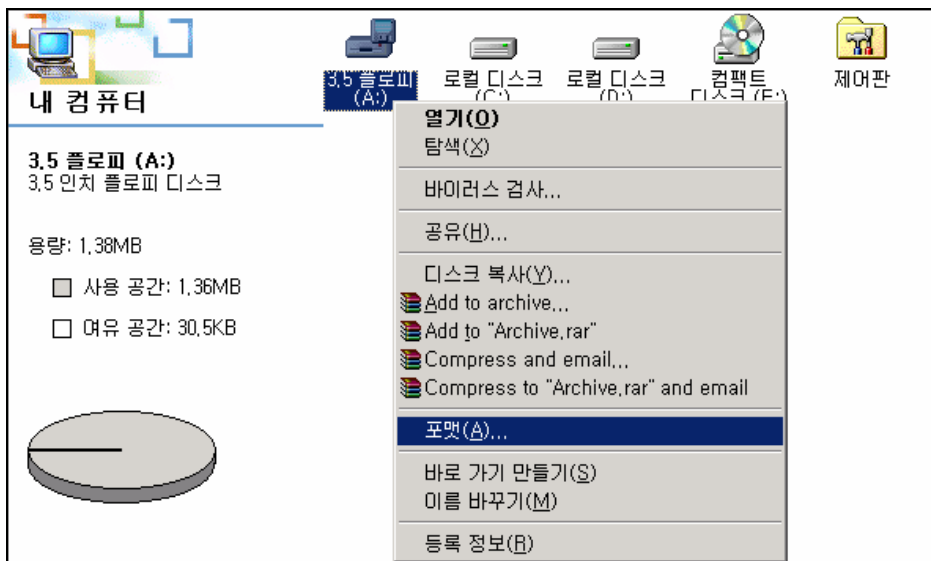
네트워크로 머신에 연결하는 모든 사람은 설치 과정에서 미디어 종류를 FTP 로 선택하고 FTP 사이트 메뉴에서 “Other”를 선택한 후 *ftp://your machine* 를 입력한다.

**주의:** 이 방법은 방화벽으로 보호되는 로컬 네트워크 머신에 적당하다. 인터넷으로 다른 머신에 FTP 서비스를 제공하는 것은 크랙커와 다른 사람의 주의를 끌기 때문에 이렇게 한다면 안전한 방법을 강구하기를 권장한다.

### 2.13.3 설치 플로피 만들기

플로피 디스크로 설치해야 된다면(이 방법을 사용하지 않기를 당부한다) 지원되지 않는 하드웨어 때문이거나 단순히 어려운 방법을 고집하는 이유일 것이다. 설치를 위해 몇 장의 플로피를 준비해야 된다.

최소한 bin 디렉터리(바이너리 배포본) 의 모든 파일을 저장할 만큼 1.44MB 나 1.2MB 플로피가 필요하다. DOS 에서 플로피를 준비한다면 MS-DOS 포맷 명령으로 포맷해야된다. 윈도우를 사용한다면 디스크를 포맷하기 위해 탐색기를 이용한다(A: 드라이브에서 마우스 오른쪽을 클릭하고 “포맷”을 선택).



공장에서 미리 포맷된 플로피는 믿지 않는다. 믿을 수 있게 다시 포맷한다. 예전에 유저들이 보고한 대부분의 문제는 적절하지 않게 포맷된 미디어가 문제였다.

다른 FreeBSD 머신에서 플로피를 만든다면 각 플로피를 DOS 파일시스템으로 만들 필요는 없지만 포맷 하는것도 괜찮은 생각이다. 이 방법 대신 `disklabel` 과 `newfs` 명령으로 아래 표시된 순서로(3.5" 1.44 MB 플로피용) UFS 파일시스템을 만들 수 있다.

```
# fdformat -f 1400 fd0.1440

# disklabel -w -r fd0.1440 floppy3

# newfs -t 2 -u 19 -l 1 -i 65536 /dev/fd0
```

**Note:** 5.25" 1.2MB 디스크는 `fd0.1200` 과 `floppy5` 를 사용한다.

이렇게 하면 다른 파일시스템처럼 마운트해서 쓸수 있다..

플로피를 포맷하고 파일을 복사한다. 배포된 파일은 적당한 크기로 나뉘지기 때문에 1.44MB 의 플로피에는 5 장이 적당하다. 모든 플로피에 원하는 배포본을 넣을 때까지 최대한 많은 파일을 각 플로피에 맞게 넣는다. 각 배포본은 플로피의 서브 디렉터리에 저장해아 된다 (예: `a:\bin\bin.aa` 와 `a:\bin\bin.ab` 처럼).

---

설치하는 중간에 미디어 화면을 만나면 "Floppy"를 선택하고 나머지 과정을 진행한다.

## 2.13.4 MS-DOS 파티션에서 설치하기

MS-DOS 파티션에서 설치하려면 배포본에서 root 디렉터리의 `freebsd` 라는 디렉터리로 파일을 복사한다. 예를 들면 `c:\freebsd` 와 같은 위치를 준비한다. CDROM 이나 FTP 사이트의 디렉터리 구조가 이 디렉터리와 같아야 되기 때문에 CD 에서 복사한다면 DOS 의 `xcopy` 명령을 권장한다. 예를 들어 최소 FreeBSD 설치를 준비하려면 다음 순서를 따른다.

```
C:\W> md c:\Wfreebsd
```

```
C:\W> xcopy e:\Wbin c:\Wfreebsd\Wbin\ /s
```

```
C:\W> xcopy e:\Wmanpages c:\Wfreebsd\Wmanpages\ /s
```

C:는 여유 공간을 E:는 CDROM 이 마운트된 곳을 의미한다.

CDROM 드라이브가 없다면 [ftp.FreeBSD.org](http://ftp.FreeBSD.org) 에서 배포본을 다운로드할 수 있다.

각 배포본은 디렉터리를 따로 가지고있다. 예를 들어 `base` 배포본은 `5.2.1/base/` 디렉터리에서([ftp://ftp.FreeBSD.org/pub/FreeBSd/releases/i386/5.2.1-RELEASE/base/](http://ftp://ftp.FreeBSD.org/pub/FreeBSd/releases/i386/5.2.1-RELEASE/base/)) 찾을 수 있다.

**Note:** FreeBSD 4.X 와 이전 릴리즈의 "base" 배포본은 "bin"이라고 부른다. 그래서 이런 버전을 사용하려면 샘플 명령과 URL 을 수정한다.

MS-DOS 파티션(가지고 있는 여유 공간)에서 설치하려는 다양한 배포본은 `C:\Wfreebsd` 에서 각각 설치한다. `BIN` 배포본은 최소 설치를 위해서만 필요하다.

## 2.13.5 설치 테잎 만들기

테잎에서 설치하는 것은 온라인 FTP 나 CDROM 에서 설치하는 것보다 가장 쉬운 방법일 것이다. 설치 프로그램은 단순히 파일을 `tar` 로 만들어 테잎에 넣는다. 관심있는 모든 배포

---

파일을 얻고나서 단순히 tar 로 만들어 테잎에 넣는다:

```
# cd /freebsd/distdir

# tar cvf /dev/rwt0 dist1 ... dist2
```

설치할 때 생성한 테잎의 모든 내용을 저장할 충분한 임시 디렉터리(선택할 수 있다)를 만들어야 된다. 이 설치 방법은 테잎의 순차 접근방식 때문에 상당히 많은 임시 공간이 필요하다. 테잎에 입력한 내용만큼의 공간이 필요함을 기억한다.

**Note:** 설치를 시작할때 부트 플로피로 부팅하기 전에 테잎을 드라이브에 넣는다. 설치 과정에서 테잎을 찾을 것이고 테잎이 없다면 실패할 것이다.

## 2.13.6 네트워크로 설치 준비하기

시리얼 포트(SLIP 나 PPP)와 패러럴 포트(PLIP (laplink 케이블)) 그리고 이더넷(표준 이더넷 컨트롤러(PCMCIA 를 포함한)) 방식의 3 가지 네트워크 설치가 있다.

SLIP 지원은 약간 오래되었기 때문에 노트북과 다른 컴퓨터를 연결하는 시리얼 케이블처럼 물리적으로 링크의 한계가 있다. 현재 SLIP 설치에서 다이얼링 기능을 제공하지 않기 때문에 링크는 물리적으로 연결해야된다.

모뎀을 사용한다면 PPP 만 유일하게 선택할 수 있다. 설치를 진행하기 전에 인터넷 서비스 업체에서 필요한 정보를 받아야 된다.

ISP(다시 말해 윈도우에서 스크립트없이 ISP에 연결할 수 있다면)에 연결하기 위해 PAP나 CHAP를 사용한다면 PPP 프롬프트에 dial이라고 입력한다. 그렇지 않으면 PPP 다이얼러가 아주 단순한 터미널 에뮬레이터만 제공하므로 모뎀에 맞는 "AT 명령"으로 ISP에 연결하는 방법을 알아야한다. 더 많은 정보는 user-ppp 핸드북과

FAQ([http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/faq/ppp.html](http://www.freebsd.org/doc/en_US.ISO8859-1/books/faq/ppp.html))를 참고한다. 문제가 있다면 set log local ....명령으로 화면에 로그를 뿌릴 수 있다.

물리적으로 연결된 다른 FreeBSD(2.0-R 이나 이후) 머신을 사용할 수 있다면 "laplink" 패러럴 포트 케이블로 설치하는것도 고려할 수 있다. 패러럴 포트를 통한 데이터 전송률이 일반



---

적인 시리얼 라인보다(50kbytes/sec 높게) 높아서 더 빨리 설치할 수 있다.

마지막으로 이더넷 카드를 사용하여 가장 빨리 설치할 수 있다. FreeBSD 는 가장 보편적인 PC 이더넷 카드를 지원한다. 지원되는 카드 리스트와 필요한 설정은 각 릴리즈 하드웨어 노트를 확인한다. 지원되는 PCMCIA 이더넷 카드 중 한가지를 사용한다면 노트북의 전원을 켜기 전에 장착해야 된다. 불행히 현재 FreeBSD 는 설치 중에 PCMCIA 카드의 *hot insertion* 을 제공하지 않는다.

그리고 네트워크 IP 주소, 넷 마스크 값과 머신의 이름도 필요하다. 고정 IP 가 아닌 PPP 로 설치한다면 IP 주소는 ISP 에서 자동으로 할당해 주기 때문에 걱정할 필요없다. IP 주소 대신 호스트 이름을 사용한다면 네임 서버와 게이트웨이(PPP 를 사용한다면 이것은 ISP 의 IP 주소가 된다) 주소가 필요하다. HTTP proxy 를 경유하여 FTP 로 설치하려면 proxy 주소가 필요하다. 위의 질문에 대한 대부분의 답을 모른다면 이 방법을 사용하기 전에 시스템 관리자 나 ISP 에 문의한다.

### 2.13.6.1 NFS 로 설치 준비하기

NFS 설치(23 장을 확인한다)는 상당히 직선적이다. 단순히 원하는 서버에 FreeBSD 배포본 을 복사해 두고 NFS 미디어를 선택한다.

이 서버가 “미리 할당된 포트”만(보통 Sun 워크스테이션에 기본적인) 지원한다면 설치가 진행되기 전에 **Options** 메뉴에 이 옵션을 설정한다.

매우 느린 전송률을 제공하는 이더넷 카드를 가지고 있다면 적절한 **Options flag** 로 변경한다.

NFS 를 사용하여 설치하려면 서버는 서브 디렉터리 마운트를 지원해야 된다. 예를 들면 FreeBSD 5.2.1 배포본이 `ziggy:/usr/archive/stuff/FreeBSD` 디렉터리에 있다면 `ziggy` 는 `/usr` 이나 `/usr/archive/stuff` 가 아닌 `/usr/archive/stuff/FreeBSD` 에 직접 마운트할 수 있어야 한다.

FreeBSD 의 `/etc/export` 파일에서 이것은 `-alldirs` 로 제어된다. 다른 NFS 서버는 다른 규정을 가지고 있을 것이다. 서버에서 "permission denied" 메시지가 보낸다면 정확한 설정이 필요하다.

---

## 3 장 유닉스 기본

### 3.1 개요

이번 장에서는 FreeBSD 운영체제의 기본적인 명령과 기능에 대해 다룬다. 이 내용은 다른 유닉스와 유사한 운영체제에 아주 밀접한 관련이 있다. 이런 내용에 대한 경험이 있다면 이번 장을 대충 살펴봐도 된다. FreeBSD 가 처음이라면 이번 장을 주의 깊게 읽어 보도록 한다.

이 장을 읽고 다음과 같은 사항을 알 수 있다.:

- FreeBSD 의 “가상 콘솔”을 어떻게 사용하는가
- 유닉스 파일 퍼미션이 어떻게 작동하는가
- 기본 FreeBSD 파일 시스템 레이아웃
- FreeBSD 디스크 구성
- 파일 시스템 마운트와 언마운트는 어떻게 하는가
- 프로세스, 데몬 그리고 신호는 무엇인가
- 셸이 무엇이고 기본 로그인 환경을 어떻게 변경하는가
- 기본 텍스트 에디터는 어떻게 사용하는가
- 장치와 장치 노드는 무엇인가
- 어떤 바이너리 포맷이 FreeBSD 에 사용되는가
- 더 많은 정보를 얻기 위해 매뉴얼 페이지는 어떻게 읽는가

---

## 3.2 가상 콘솔 & 터미널

FreeBSD 는 다양한 방법으로 사용할 수 있다. 그 중 하나는 텍스트 터미널로 명령을 입력하는 것이다. 유닉스 운영체제의 수 많은 유연성과 강력함은 이 방법으로 FreeBSD 를 사용 할때 쉽게 이용할 수 있다. 이번 장에서는 터미널과 콘솔이 무엇인지 FreeBSD 에서 이들을 어떻게 이용할 수 있는지 설명한다.

### 3.2.1 콘솔

시작하는 동안 그래픽 환경을 자동으로 시작하도록 FreeBSD 를 설정하지 않았다면 시스템은 부팅 후 시작 스크립트 실행이 끝나자마자 로그인 프롬프트를 보여준다. 다음과 비슷한 화면을 볼수 있다.

```
Additional ABI support:
Local package initialization:
Additional TCP options:

Fri Sep 20 13:01:06 EEST 2002

FreeBSD/i386 (pc3.example.org) (ttyv0)

login:
```

이 메시지는 여러분의 시스템과 약간 다르지만 비슷한 화면을 보게 된다. 마지막 두 라인에 우리는 관심을 가져야 된다. 다음은 마지막 두 번째 라인이다.

```
FreeBSD/i386 (pc3.example.org) (ttyv0)
```

이 라인은 부팅된 시스템에 관한 약간의 정보를 포함하고 있다. **x86 아키텍처**의 인텔이나 인텔 호환 프로세서에서 실행 중인 "FreeBSD" 콘솔을 보고있다. 이 머신의 이름은(모든 유닉스 머신은 이름을 가지고 있다) pc3.example.org 이고 여러분은 이 시스템 콘솔을 보고 있다 -- ttyv0 터미널.

마지막 라인은 항상 로그인 프롬프트다.

---

login:

이곳은 FreeBSD 에 로그인하기 위해 “유저 이름”을 입력하는 곳이다. 다음 장에서 어떻게 로그인 하는지 설명한다.

### 3.2.2 FreeBSD 에 로그인

FreeBSD 는 멀티유저 멀티프로세싱 시스템이다. 이 말은 머신 한대에서 수많은 프로그램을 동시에 실행하는 여러 사람들이 사용할 수 있는 시스템에 보통 주어지는 형식적인 표현이다.

모든 멀티유저 시스템은 나머지 사람들로 부터 사용자를 구분하는 몇 가지 방법이 필요하다. 이 방법은 FreeBSD 에서(그리고 유닉스와 유사한 모든 운영체제) 프로그램을 실행할 수 있기 전에 모든 유저는 시스템에 “로그인” 해야 된다. 모든 유저는 고유하고 개인적인 이름과(“유저 이름”) 보안 키를(“패스워드”) 가지고 있다. FreeBSD 는 유저가 어떤 프로그램이라도 실행할 수 있기 전에 이 두 가지를 물어본다.

FreeBSD 가 부팅하고 **시작 스크립트**가 실행을 끝낸 후 정확한 유저 이름을 물어보기 위해 프롬프트를 표시한다.

login:

이 예제에서 우리는 유저 이름이 john 이라고 가정한다. 이 프롬프트에 *john* 이라고 입력하고 **Enter** 를 누른다. 그러면 “password”를 입력하는 프롬프트가 나타난다:

login: john

Password:

이제 john 의 패스워드를 입력하고 **Enter** 를 누른다. 패스워드는 눈에 보이지 않는다! 이제 보안적인 사항을 만족시켰기 때문에 이에 대해 걱정할 필요가 없다.

패스워드를 정확하게 입력했다면 이제 FreeBSD 에 로그인되고 모든 명령을 입력할 준비가 된다.

MOTD(/etc/motd)나 명령 프롬프트(**#**, **\$** 또는 **%** 문자) 다음의 날짜 메시지를 보게 된다. 이것은 여러분이 성공적으로 FreeBSD 에 로그인 했음을 보여준다.

---

```

Password:
Last login: Sun Sep 12 19:12:15 from 211.112.127.83
Copyright (c) 1980, 1983, 1986, 1988, 1990, 1991, 1993, 1994
    The Regents of the University of California. All rights reserved.

Last login: Fri Sep 10 21:08:31 2004 from 210.90.81.126
Copyright (c) 1980, 1983, 1986, 1988, 1990, 1991, 1993, 1994
    The Regents of the University of California. All rights reserved.

FreeBSD 4.10-RELEASE (GENERIC) #0: Tue May 25 22:47:12 GMT 2004

Welcome to FreeBSD!

Before seeking technical support, please use the following resources:

o Security advisories and updated errata information for all releases are
  at http://www.FreeBSD.org/releases/ - always consult the ERRATA section
  for your release first as it's updated frequently.

o The Handbook and FAQ documents are at http://www.FreeBSD.org/ and,
  along with the mailing lists, can be searched by going to
  http://www.FreeBSD.org/search/. If the doc distribution has
  been installed, they're also available formatted in /usr/share/doc.

If you still have a question or problem, please take the output of
`uname -a`, along with any relevant error messages, and email it
as a question to the questions@FreeBSD.org mailing list. If you are
unfamiliar with FreeBSD's directory layout, please refer to the hier(7)
man page. If you are not familiar with man pages, type `man man`.

You may also use /stand/sysinstall to re-enter the installation and
configuration utility. Edit /etc/motd to change this login announcement.
You have new mail.
%■
```

### 3.2.3 여러 개의 콘솔

콘솔 하나에서 유닉스 명령을 실행하는 것도 괜찮지만 FreeBSD는 한번에 많은 프로그램을 실행할 수 있다. 동시에 여러 프로그램을 실행할 수 있는 FreeBSD와 같은 운영체제를 사용할 때 하나의 콘솔에서만 명령을 입력하는 것은 상당한 낭비가 된다. 이에 대해 “가상 콘솔”이 많은 도움을 줄 수 있다.

많은 가상 콘솔을 보여 주도록 FreeBSD를 설정할 수 있다. 키 보드의 키 쌍을 눌러 가상 콘솔 중 한곳에서 다른 곳으로 변경할 수 있다. 각 콘솔은 각자 다른 출력 채널을 가지고 있고 가상 콘솔을 다른 가상 콘솔로 변경하는 동안 FreeBSD는 적절한 키보드 입력과 모니터 출력으로 방향을 바꾼다.

특수 키 조합은 FreeBSD가 콘솔을 변경하도록 남겨 두었다. FreeBSD에서 **Alt-F1**, **Alt-**

---

F2 에서 **Alt-F8** 를 사용하여 다른 가상 콘솔로 변경할 수 있다.

**Note:** 상당히 기술적이고 정확한 FreeBSD 콘솔의 자세한 설명과 키보드 드라이버는 `syscons(4)`, `atkbd(4)`, `vidcontrol(1)`과 `kbdcontrol(1)`의 매뉴얼 페이지에서 찾을 수 있다. 여기서는 자세히 설명하지 않지만 흥미있는 독자는 어떻게 동작하는지 자세히 설명한 전체적인 내용을 매뉴얼 페이지에서 찾을 수 있다.

다음 콘솔로 변경하는 동안 FreeBSD 는 주의해서 화면 출력을 저장하고 복구한다. 이 결과가 여러 "가상" 화면과 FreeBSD 가 실행하도록 명령 입력에 사용할 수 있는 키보드의 환영이다. 가상 콘솔에서 실행한 프로그램은 콘솔에 보이지 않더라도 중지되지 않는다. 다른 가상 콘솔로 변경할 때도 프로그램은 계속 실행 중이다.

### 3.2.4 /etc/ttys 파일

FreeBSD 의 기본 설정은 8 개의 가상 콘솔을 시작한다. 이것은 하드웨어적인 설정이 아니어서 더 많게 혹은 더 적은 가상 콘솔로 부팅하도록 수정하기 쉽다. 가상 콘솔의 개수와 설정은 `/etc/tty` 파일에 설정되어 있다.

FreeBSD 가상 콘솔 설정에 `/etc/ttys` 파일을 사용할 수 있다. 이 파일에서 주석 처리되지 않은 각 라인은(`#` 문자로 시작되지 않은 라인) 싱글 터미널이나 가상 콘솔 설정을 가지고 있다. 이 파일의 기본 버전은 9 개의 가상 콘솔을 FreeBSD 가 설정하도록하고 이중 8 개가 활성화된다. 다음은 `ttysv` 파일에서 가상 콘솔의 내용이다.

```

#
# $FreeBSD: src/etc/etc.i386/ttys.v 1.8.2.1 2003/11/14 12:59:51 simokawa Exp $
#   @(#)ttys      5.1 (Berkeley) 4/17/89
#
# This file specifies various information about terminals on the system.
# It is used by several different programs.  Common entries for the
# various columns include:
#
# name The name of the terminal device.
#
# getty The program to start running on the terminal.  Typically a
#       getty program, as the name implies.  Other common entries
#       include none, when no getty is needed, and xdm, to start the
#       X Window System.
#
# type The initial terminal type for this port.  For hardwired
#       terminal lines, this will contain the type of terminal used.
#       For virtual consoles, the correct type is cons25.  Other
#       common values include network for network connections on
#       pseudo-terminals, dialup for incoming modem ports, and unknown
#       when the terminal type cannot be predetermined.
#
# status Must be on or off.  If on, init will run the getty program on
#        the specified port.  If the word "secure" appears, this tty
#        allows root login.
#
# name  getty                type    status      comments
#
# If console is marked "insecure", then init will ask for the root password
# when going to single-user mode.
console none                unknown off secure
#
ttyv0  "/usr/libexec/getty Pc"  cons25 on secure
# Virtual terminals
ttyv1  "/usr/libexec/getty Pc"  cons25 on secure
ttyv2  "/usr/libexec/getty Pc"  cons25 on secure
ttyv3  "/usr/libexec/getty Pc"  cons25 on secure
ttyv4  "/usr/libexec/getty Pc"  cons25 on secure
ttyv5  "/usr/libexec/getty Pc"  cons25 on secure
ttyv6  "/usr/libexec/getty Pc"  cons25 on secure
ttyv7  "/usr/libexec/getty Pc"  cons25 on secure
ttyv8  "/usr/X11R6/bin/xdm -nodaemon" xterm off secure

```

이 파일에있는 모든 칼럼의 자세한 설명과 가상 콘솔을 설정하는데 사용할 수 있는 옵션은 ttys(5) 매뉴얼 페이지를 참고한다.

### 3.2.5 싱글 유저 모드 콘솔

“싱글 유저 모드”는 무엇인가?에 대한 자세한 설명은 12.6.2 장에서 찾을 수 있다. 싱글 유저 모드에서 FreeBSD 를 운용할때 오직 하나의 콘솔은 의미가 없다. 이용할 수 있는 가상 콘솔이 없기 때문이다. 싱글 유저 모드 콘솔 설정도 /etc/ttys 파일에서도 찾을 수 있다. *console* 로 시작하는 라인을 확인한다:

```

# name  getty                                type  status  comments
#
# If console is marked "insecure", then init will ask for the root password
# when going to single-user mode.
console none                                unknown of secure

```

**Note:** 위의 *console* 라인이 보여주는 주석에서 이 라인의 *secure*를 *insecure*로 편집할 수 있다. 변경한다면 FreeBSD가 싱글 유저 모드로 부팅할 때 root 패스워드를 물어 본다.

따라서 *insecure*로 변경할 때 주의한다. root 패스워드를 잊어버렸다면 싱글 유저 모드로 부팅하는 것이 더욱 복잡해진다. 부팅은 가능 하지만 FreeBSD 부팅 프로세스와 관련된 프로그램을 잘 다루지 못하는 사용자들에게는 어려운 일이 될 것이다.

### 3.3 퍼미션

BSD 유닉스의 전통을 따르는 FreeBSD는 유닉스 개념의 몇 가지 키에 기반한다. 확실한 첫 번째는 FreeBSD는 멀티 유저 운영체제라는 것이다. 시스템은 전혀 연관이없는 태스크에서 여러 유저들의 모든 작업을 동시에 제어할 수 있다. 시스템은 하드웨어 장치, 주변 장치, 메모리에 대한 적절한 공유와 요청을 제어하고 CPU 시간을 각 유저에게 공정히 분배하는 책임을 가지고 있다.

시스템은 멀티 유저를 지원할 수 있기 때문에 누가 자원을 읽기, 쓰기 그리고 실행시킬 수 있는지 관리하는 퍼미션 세트를 가지고 있다. 이들 퍼미션에서 하나는 파일 주인을, 다른 하나는 파일이 속한 그룹을 그리고 마지막은 나머지를 나타내는 두 개의 8진수가 3개로 나누어져 저장되어있다. 숫자 표현은 다음과 같은 동작을 나타낸다.

값	퍼미션	디렉터리 리스트
0	No read, no write, no execute	---
1	No read, no write, execute	--x
2	No read, write, no execute	-w-
3	No read, write, execute	-wx
4	Read, no write, no execute	r--



5	Read, no write, execute	r-x
6	Read, write, no execute	rw-
7	Read, write, execute	rwX

소유자, 그룹 그리고 나머지 사람들을 위한 파일의 퍼미션 관련 칼럼과 정보를 포함하는 자세한 디렉터리 리스트를 보기 위해 `-l` 명령어 라인 인자를 `ls(1)`에 사용할 수 있다. 예를 들면 임의의 디렉터리에서 `ls -l`은 다음과 같은 내용을 보여 줄 것이다.

```
% ls -l
total 530
-rw-r--r-- 1 root  wheel   512 Sep  5 12:31 myfile
-rw-r--r-- 1 root  wheel   512 Sep  5 12:31 otherfile
-rw-r--r-- 1 root  wheel  7680 Sep  5 12:31 email.txt
...
```

이제 `ls -l`의 첫 번째 칼럼이 어떻게 나뉘는지 설명한다:

`-rw-r--r--`

첫 번째(왼쪽부터) 문자는 이 파일이 보통 파일, 디렉터리, 특별한 캐릭터(character) 장치, 소켓 또는 특별한 pseudo-file 장치인지 나타낸다. 이 경우 `-`는 보통 파일을 나타낸다. 이 예제에서 다음 3개의 문자 `rw-`는 파일 주인의 퍼미션을 나타낸다. 다음 세 번째 문자 `r--`는 파일이 속한 그룹 퍼미션을 마지막의 세 개의 문자 `r--`는 나머지 퍼미션이다. dash(-)의 의미는 퍼미션 설정이 안된 것을 의미한다. 이 파일의 경우 퍼미션이 지정되어서 파일 주인은 읽고 쓰기를, 그룹은 파일을 읽기만, 나머지도 읽기만 가능하다. 위 표에 따르면 이 파일의 퍼미션은 644다. 각 숫자는 파일 퍼미션의 3개 파트를 나타낸다.

이 설정이 모든 유저를 만족하겠지만 장치에서는 시스템이 퍼미션을 어떻게 제어할까? FreeBSD는 실제로 대부분의 하드웨어 장치를 프로그램이 열고, 읽고, 데이터를 쓸수 있는 다른 파일처럼 파일로 간주한다. 이들 특별한 장치 파일은 `/dev` 디렉터리에 저장되어 있다.

디렉터리도 파일로 간주되어 읽고, 쓰고, 실행할 수 있는 퍼미션을 가지고 있다.

디렉터리에서 실행 가능한 비트는 파일과 약간 다른 의미를 갖는다. 디렉터리에 실행 가능한 표시가 되어 있다면 이 의미는 디렉터리 안으로 들어갈 수 있다는 말이다. 또한 이 의미는 디렉터리에서 알고 있는 파일에 접근할 수 있다는 것이다(물론 파일에 퍼미션이

---

있어야 된다).

디렉터리 리스트를 보려면 특히 읽기 퍼미션이 디렉터리에 있어야 되고 반면에 이름을 알고 있는 파일을 삭제하려면 쓰기와 실행 가능한 퍼미션이 파일을 포함한 디렉터리에 있어야 된다.

더 많은 퍼미션 비트가 있지만 `setuid` 바이너리와 `sticky` 디렉터리는 특별한 상황에서 주로 사용된다. 파일 퍼미션과 퍼미션을 어떻게 지정하는지 더 많은 정보를 원한다면 `chmod(1)` 매뉴얼 페이지를 찾아본다.

### 3.3.1 심볼릭 퍼미션

가끔 심볼릭 표현이라고 말하는 심볼릭 퍼미션은 파일이나 디렉터리에 퍼미션을 할당하는 8 진수 값 대신 문자를 사용한다. 심볼릭 표현은 다음 값들을 사용할 수 있는 (누구에게) (어떻게) (퍼미션)의 구문을 사용한다.

옵션	문자	표현
(누구에게)	U	유저
(누구에게)	G	그룹
(누구에게)	O	나머지
(누구에게)	A	모두 (“전체”)
(어떻게)	+	퍼미션 추가
(어떻게)	-	퍼미션 삭제
(어떻게)	=	확실하게 퍼미션 설정
(퍼미션)	R	읽기
(퍼미션)	W	쓰기
(퍼미션)	X	실행
(퍼미션)	t	Sticky bit
(퍼미션)	s	Set UID 또는 GID

이들 값은 이전처럼 `chmod(1)`에 사용하지만 숫자 대신 문자를 사용한다. 예를들면 다른 유저들이 `FILE`에 접근하는 것을 막기 위해 다음 명령을 사용할 수 있다.

```
% chmod go= FILE
```

---

컴마로(,) 나누어진 리스트는 파일에 하나 이상의 설정을 변경할때 사용할 수 있다. 예를 들어 다음 명령은 *FILE*에서 그룹과 남은 사람들의 쓰기 퍼미션을 삭제하고 남은 사람들에게 실행 퍼미션을 추가한다.

```
% chmod go-w, a+w FILE
```

### 3.4 디렉터리 구조

FreeBSD 디렉터리 계층은 시스템을 전체적으로 이해하기 위한 기본이다. 가장 중요한 개념을 잡기 위해 root 디렉터리를 "/" 알아야 된다. 이 디렉터리는 부팅할때 처음 마운트되고 멀티 유저 운영을 준비하기 위해 필요한 기본 시스템을 포함하고 있다. 또한 root 디렉터리는 마운트하려는 모든 파일 시스템의 마운트 포인트를 가지고있다.

마운트 포인트는 추가적인 파일 시스템을 root 파일 시스템에 붙일 수 있는 디렉터리다. 표준 마운트 포인트는 /usr, /var, /mnt 와 /cdrom 을 포함한다. 이들 디렉터리는 보통 /etc/fstab 파일 엔트리를 참조한다. /etc/fstab 은 다양한 파일 시스템 테이블이고 시스템이 마운트 포인트를 참조한다. /etc/fstab 에있는 대부분의 파일 시스템은 *noauto* 옵션을 가지고있지 않다면 스크립트 rc(8)에 의해 부팅할때 자동으로 마운트된다. /etc/fstab 파일 포맷과 이 파일이 가지고있는 옵션에 대한 더 많은 정보는 fstab(5) 매뉴얼 페이지를 참고한다.

#### /etc/fstab 내용

```
# See the fstab(5) manual page for important information on automatic mounts
# of network filesystems before modifying this file.
#
# Device          Mountpoint      FStype  Options      Dump    Pass#
/dev/ad0s1b       none            swap    sw           0       0
/dev/ad0s1a       /                ufs     rw           1       1
#/dev/ad0s1h     /home           ufs     rw           2       2
/dev/ad0s1h       /home           ufs     rw,userquota 1       2
/dev/ad0s1e       /tmp            ufs     rw           2       2
/dev/ad0s1g       /usr            ufs     rw           2       2
/dev/ad0s1f       /var            ufs     rw           2       2
/dev/acd0c        /cdrom          cd9660  ro,noauto    0       0
proc              /proc           procfs  rw           0       0
```

파일 시스템 계층에 대한 완벽한 설명은 hier(7)에서 볼 수 있다. 이제 아주 일반적인 디렉터리에 대한 간략한 개요가 파일 시스템 계층에 대한 이해를 충족시킬 것이다.

디렉터리	설명
/	파일 시스템의 root 디렉터리
/bin/	싱글 유저와 멀티 유저를 위한 기본 유저 유틸리티
/boot/	운영체제가 부트 스트랩하는 동안 사용되는 프로그램과 설정 파일
/boot/defaults/	기본 부트스트랩 설정 파일. loader.conf(5)를 본다.
/dev/	장치 노드. intro(4)를 본다
/etc/	시스템 설정 파일과 스크립트
/etc/defaults/	기본 시스템 설정 파일. rc(8)을 본다
/etc/mail/	sendmail(8)같은 매일 전송 에이전트 설정 파일
/etc/namedb/	named 설정파일. named(8)을 본다
/etc/periodic/	cron(8)로 매일, 주간, 월간으로 실행되는 스크립트. periodic(8)을 본다
/etc/ppp/	ppp 설정 파일. ppp(8)을 본다
/mnt/	시스템 관리자가 사용하는 일반적인 빈 디렉터리.
/proc/	프로세스 파일 시스템. procfs(5)과 mount_procfs(8)을 본다
/root/	root 계정의 홈 디렉터리
/sbin/	싱글 유저와 멀티 유저 환경의 시스템 프로그램과 기본적인 관리자 유틸리티.
/stand/	Standalone 환경에서 사용되는 프로그램
/tmp/	임시 파일. 보통 mfs(8) 메모리 기반 파일 시스템 (/tmp의 내용은 보통 시스템이 재 부팅할때 보존되지 않는다).
/usr/	주요 유저 유틸리티와 어플리케이션.
/usr/bin/	일반적인 유틸리티, 프로그램 툴과 어플리케이션.
/usr/include/	표준 C include 파일
/usr/lib/	라이브러리 보관
/usr/libdata/	잡다한 유틸리티 데이터 파일
/usr/libexec/	시스템 데몬 & 시스템 유틸리티(다른 프로그램이 실행하는). 로컬에서 실행 가능한 라이브러리 등. 또한 FreeBSD 포트 프레임 워크의 기본 목적지로 사용된다.
/usr/local/	/usr/local에서 /usr가 사용하도록 hier(7)이 보편적으로 스케치한 레이아웃. /usr/local/share가 아닌 /usr/local의 man 디렉터리를 제외하고 포트 문서는 share/doc/port에 있다.
/usr/obj/	usr/src 트리를 만들 때 생성되는 아키텍처에 특별한 목적 트리.
/usr/ports	FreeBSD 포트 컬렉션(옵션).
/usr/sbin/	시스템 데몬 & 시스템 유틸리티 (유저가 실행하는)

/usr/share/	아키텍처 독립 파일
/usr/src/	BSD 로컬 소스 파일
/usr/X11R6/	실행 가능한 X11R6 배포본과 라이브러리 등 (옵션).
/var/	다양한 목적의 로그, 임시, 일시적인 파일과 스푼 파일.
/var/log/	잡다한 시스템 로그 파일.
/var/mail/	유저의 메일 박스 파일.
/var/spool/	잡다한 프린터와 메일 시스템 스푼링 디렉터리.
/var/tmp/	시스템이 재 부팅할 때까지의 임시 파일.
/var/yp	NIS 맵

### 3.5 디스크 구성

FreeBSD 가 파일을 찾기 위해 사용하는 가장 작은 단위는 파일 이름이다. 파일 이름 readme.txt 와 README.TXT 는 두개의 다른 파일로 대소 문자를 구분한다. FreeBSD 는 파일이 프로그램, 문서 또는 다른 데이터와 연관된 것인지 결정하기 위해 파일 확장자를(.txt) 사용하지 않는다.

**파일 확장자:** 윈도우를 예로 들면 \*.doc 로 끝나는 파일은 워드에서 열수 있고 \*.exe 로 끝나는 파일을 실행할 수 있다. 이처럼 윈도우에서는 확장자를 통하여 파일 종류를 선별하기 때문에 일반적인 파일은 더블클릭만 하면 해당 프로그램이 실행된다.

파일은 디렉터리에 저장되어 있다. 디렉터리에는 파일이 없을 수도 있고 수백 개의 파일이 있을 수 있다. 또한 디렉터리는 다른 디렉터리를 포함하거나 다른 디렉터리 계층에 디렉터리를 만들 수 있다. 이것은 데이터를 조직화하기 매우 쉽다.

파일과 디렉터리는 슬러시(/) 뒤에 주어진 파일이나 디렉터리 이름으로 참조되고 필요하다면 다른 디렉터리 이름 뒤에 올수 있다. readme.txt 파일을 가지고있는 bar 디렉터리를 포함하는 foo 디렉터리를 가지고 있다면 파일의 전체 이름이나 경로는 **foo/bar/readme.txt** 다.

디렉터리와 파일은 파일 시스템에 저장된다. 각 파일 시스템은 최상단에 *root* 디렉터리라고 부르는 정확히 하나의 디렉터리를 가지고 있다. 이 root 디렉터리는 다른 디렉터리를 포함할 수 있다.

이런 방식은 사용중인 다른 운영체제도 비슷할 것이다. 약간 다른 점은 예를들어 DOS 는

---

₩로 파일과 디렉터리 이름을 나누지만 MacOS 는 :를 사용한다.

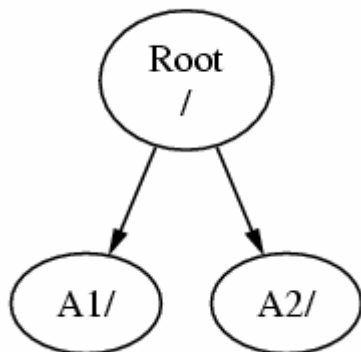
FreeBSD 는 경로에 드라이브 문자나 드라이브 이름을 사용하지 않는다. 따라서 FreeBSD 에서는 다음과 같이 입력하지 않는다.

c:/foo/bar/readme.txt

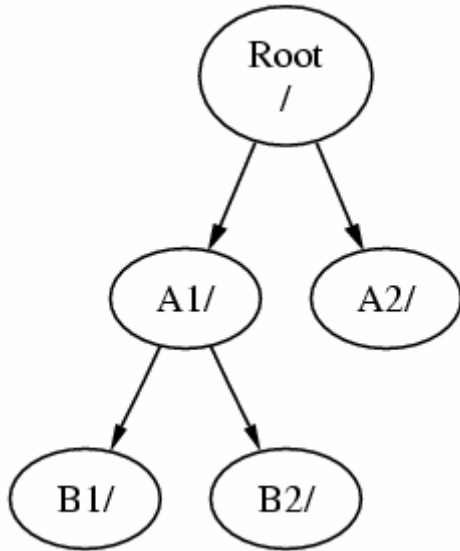
대신 하나의 파일 시스템은 root 파일 시스템으로 지정되어 있다. root 파일 시스템의 root 디렉터리는 / 라고 한다. 다른 모든 파일 시스템은 root 파일 시스템 아래에 마운트된다. FreeBSD 시스템이 많은 디스크를 가지고 있더라도 모든 디렉터리는 같은 디스크의 일부분으로 나타난다.

A, B, C라고 부르는 3개의 파일 시스템을 가지고 있다고 가정하고 각 파일 시스템은 A1과 A2(또는 B1, B2나 C1, C2) 두 개의 다른 디렉터를 포함하는 하나의 root 디렉터를 가지고 있다.

A root 파일 시스템에서 디렉터리 내용을 보기 위해 ls 명령을 사용했다면 두개의 서브 디렉터리 A1과 A2를 보게된다. 디렉터리 트리는 다음과 같다.

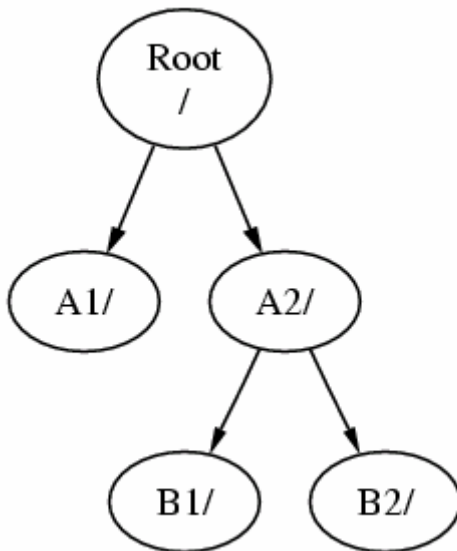


파일 시스템은 다른 파일 시스템의 디렉터리에 마운트 해야된다. 그래서 파일 시스템 B를 디렉터리 A1에 마운트 했다고 가정한다. B의 root 디렉터리는 A1이 되고 따라서 디렉터리는 B는 다음과 같은 위치가 된다.



필요하다면 *B1* 이나 *B2* 디렉터리에 있는 파일은 경로가 */A1/B1* 이나 */A1/B2* 가 된다.  
*/A1* 에 있던 파일은 일시적으로 안보이게 된다. 이 파일들은 *B*가 *A*에서 *언마운트*된다면 다시 나타난다.

*B*가 *A2*에 마운트되어 있다면 다음 도형과 같은 모습일 것이다.

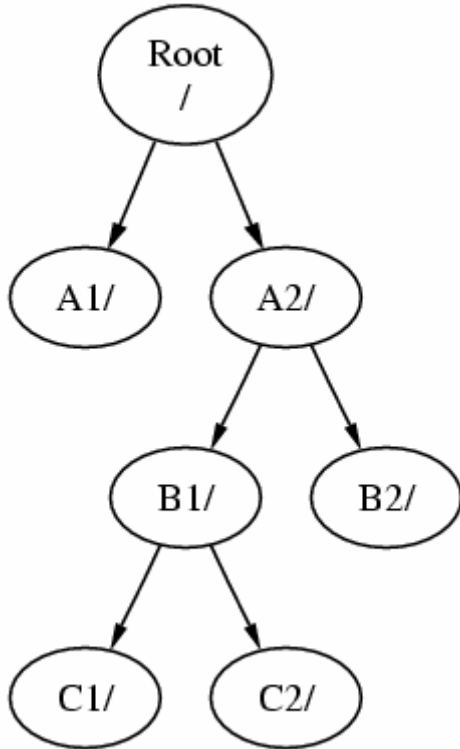


그리고 경로는 각자 */A2/B1* 과 */A2/B2* 가 될 것이다.

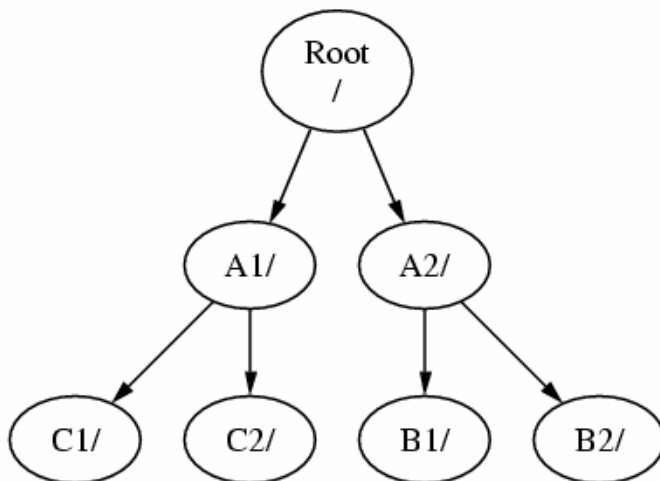
파일 시스템은 다른 파일 시스템의 최상 단계에 마운트할 수 있다. 마지막 예제를 계속하면

---

C 파일 시스템은 B 파일 시스템에서 B1 디렉터리의 최 상단에 마운트할 수 있다. 이해하기 쉽게 다음 그림을 준비한다.



그렇지 않으면 C는 A 파일 시스템의 A1 디렉터리 밑에 직접 마운트할 수 있다.



DOS 에 대해 잘 알고 있다면 완전히 같지는 않지만 `join` 명령과 비슷하다.



---

보통 별로 걱정할 필요는 없다. FreeBSD 를 설치할때 일반적인 파일 시스템을 만들어서 마운트할 곳을 결정하고 디스크를 새로 추가하지 않으면 절대 변경할 필요가 없다.

다른 파일 시스템은 만들지 않고 커다란 root 파일 시스템만 만들어도 된다. 이 방법은 약간의 결점과 하나의 장점이 있다.

### 여러 파일 시스템의 이점

- 다른 파일 시스템은 다른 *마운트 옵션*을 가질 수 있다. 예를 들면 주의 깊게 계획해서 root 파일 시스템은 읽기 전용으로만 마운트하여 부주의한 삭제나 중요한 파일을 변경하지 못하게할 수 있다. /home 디렉터리처럼 유저가 쓰기를 할수 있는 파일 시스템도 nosuid 로 마운트할 수 있다. 이 옵션은 파일 시스템에서 **suid/guid** 비트를 막아서 보안을 강화할 수 있다.

**suid/guid** 비트: 특정 프로그램을 실행하기 위해 root 나 시스템 유저 권한이 필요하다. 특히 패스워드 변경처럼 /etc/passwd 파일을 업데이트하기 위해서는 root 권한이 필요하다. Setuid 가 없다면 모든 패스워드를 root 가 변경해 줘야된다. 이처럼 특정 프로그램에 setuid 비트를 설정하여 그 프로그램이 실행되는 동안 일시적으로 root 나 시스템 유저 권한을 가지게 된다. Setgid 도 유저대신 특정 그룹의 권한을 갖는 것을 제외하고는 같다. 따라서 보안에 굉장히 민감하기 때문에 사용에 주의한다. 자세한 내용은 setuid 매뉴얼 페이지를 참고한다.

```
-r-sr-xr-x 2 root wheel 32888 May 26 06:30 /usr/bin/passwd
```

- FreeBSD 는 파일 시스템의 용도에 따라 파일 시스템의 파일 레이아웃을 자동으로 최적화 한다. 주기적으로 쓰기가 행해지는 수많은 작은 파일을 가지고있는 파일 시스템은 다른 형태의 파일 시스템과 다른 최적화가 수행된다. 하나의 큰 파일 시스템으로는 최적화할 수 없다.
- FreeBSD 의 파일 시스템은 전원 끊김에 강하다. 그러나 중요한 순간에 전원이 끊어지면 파일 시스템 구조가 깨질 수 있다. 데이터가 여러 개의 파일 시스템에 나눠 있다면 시스템은 부팅할 수 있고 백업된 데이터에서 복구하는 것도 쉽다.

### 싱글 파일 시스템의 이점

- 파일 시스템은 크기가 고정되어 있다. FreeBSD 를 설치할 때 크기를 지정하여 파일

---

시스템을 만들었다면 뒤늦게 더 큰 파티션이 필요하다는 것을 발견할 것이다. 백업하지 않고 새로운 크기로 파일 시스템을 만들고 백업 데이터를 복구하는 것은 어려운 작업이다.

**중요:** FreeBSD 4.4 와 그 후 버전의 기능 중 growfs(8) 명령으로 빠르게 파일 시스템 크기를 증가 시킬 수 있어 이런 한계를 극복할 수 있다.

파일 시스템은 파티션에 포함되어 있다. 이것은 FreeBSD 의 유닉스 전통 때문에 앞장에서 사용한 파티션 이라는 용어와 다른 의미를 가진다. 각 파티션은 a 에서 h 까지의 문자로 식별된다. 각 파티션은 하나의 파일 시스템만 포함할 수 있다. 이 의미의 파일 시스템은 파일 시스템 계층에서 그들의 마운트 포인트나 그들이 포함된 파티션의 문자 중 하나로 자주 설명된다.

그리고 FreeBSD 는 *스왑 공간*으로 디스크를 사용한다. 스왑 공간은 FreeBSD 에 *가상 메모리*를 제공한다. 이것은 컴퓨터가 실제보다 더 많은 메모리를 사용할 수 있게한다. FreeBSD 는 메모리 부족이되면 현재 사용하지않는 데이터를 스왑 공간으로 이동시키고 필요할때 다시 불러온다.

어떤 파티션은 특정 관례를 가지고 있다.

파티션	관 려
a	보통 root 파일 시스템을 가지고 있다.
b	보통 스왑 공간을 가지고 있다.
c	보통 슬라이스를 감싸는 크기와 같다. 이것은 유틸리티가 C 파티션의 전체 슬라이스에서 작업할 수 있도록 한다(예를들면 배드 블록 스캐너). 일반적으로 이 파티션에는 파일 시스템을 만들지 않는다.
d	파티션 d 는 더 이상 사용하지 않더라도 특별한 의미를 지니고 있다. 요즘의 어떤 툴은 파티션 d 에서 실행시키면 이상하게 동작하기 때문에 <b>sysinstall</b> 은 일반적으로 d 파티션을 만들지 않는다.

파일 시스템을 가지고있는 각 파티션은 FreeBSD 가 *슬라이스(slice)*라고 부르는곳에 저장되어있다. 슬라이스는 FreeBSD 가 예전 파티션을 부르는 용어고 FreeBSD 가 유닉스 기반이기 때문에 현재도 사용하고 있다. 슬라이스는 1 부터 4 까지의 번호다.

슬라이스 번호는 장치 이름 앞에 *s*를 붙이고 1 부터 시작한다. 그래서 "da0s1"는 첫 번째 SCSI 장치의 첫 번째 슬라이스다. 디스크에 4 개의 물리적인 슬라이스를 만들 수 있지만

적절한 종류의 물리적인 슬라이스 안에 논리적인 슬라이스를 만들 수 있다. 이렇게 확장된 슬라이스 번호는 5 부터 시작되기 때문에 "ad0s5"는 첫 번째 IDE 디스크의 첫 번째 확장 슬라이스다. 이들 장치는 슬라이스를 차지하고 있는 파일 시스템에 사용된다.

슬라이스와 다른 드라이브는 문자 a 에서 h 로 표현되는 *파티션*을 가지고 있다. 이 문자를 장치 이름에 덧붙이기 때문에 "da0a"는 첫 번째 da 드라이브 파티션이다. "ad1s3e"는 두 번째 IDE 디스크 드라이브에있는 새 번째 슬라이스의 다섯 번째 파티션이다.

마지막으로 시스템의 각 디스크는 동일하다. 디스크 이름은 디스크 종류를 표시하는 코드로 시작되고 어떤 디스크인지 번호가 표시된다. 슬라이스와 달리 디스크 번호는 0 에서 시작된다. 표 3-1 에서 일반적인 코드를 볼 수 있다.

FreeBSD 에 필요한 파티션을 언급할때 파티션을 가지고 있는 슬라이스와 디스크 이름도 표시해야되고 슬라이스를 언급할 때는 디스크 이름도 표시해야 된다. 디스크 이름을 리스트 할때 s + 슬라이스 번호 + 파티션 문자를 나열한다. 예제는 예제 3-1 에서 보여준다.

예제 3-2 는 이해를 돕기 위해 디스크 레이아웃의 개념적인 모델을 보여준다.

FreeBSD 를 설치하기 위해 첫째로 디스크 슬라이스를 설정하고, FreeBSD 를 설치할 슬라이스에 파티션을 만들고, 각 파티션에 파일 시스템을(또는 스왑 공간) 만들고 파일 시스템을 마운트할 곳을 결정한다.

**표 3-1. 디스크 장치 코드**

코드	의미
Ad	ATAPI (IDE) disk
Da	SCSI direct access disk
AcD	ATAPI(IDE) CDROM
Cd	SCSI CDROM
Fd	Floppy disk

**예제 3-1. 샘플 디스크, 슬라이스, 파티션 이름**

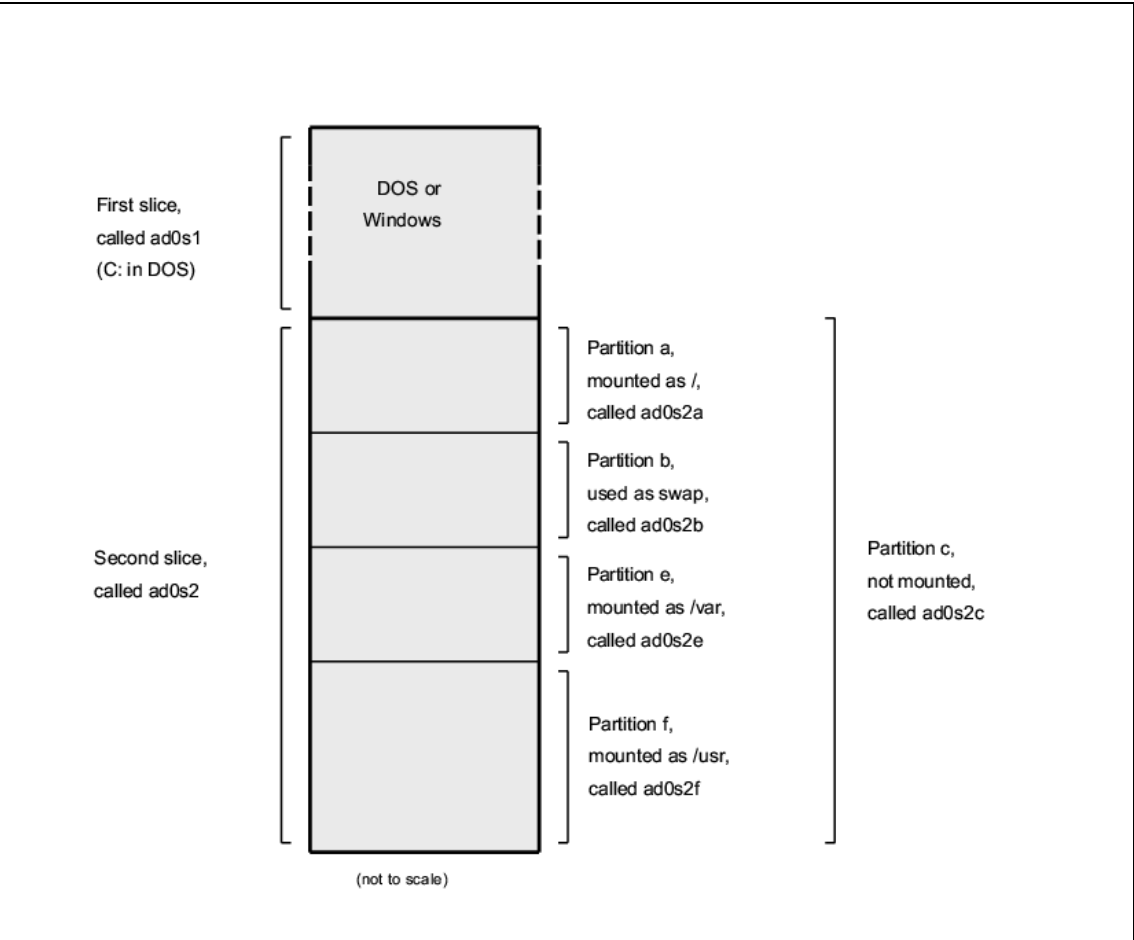
이름	의미
ad0s1a	첫 번째 IDE 디스크에서(ad0) 첫 번째 슬라이스의(s1) 첫 번째 파티션(a)

da1s2e	두 번째 SCSI 디스크(da1)에서 두 번째 슬라이스의(s2) 다섯 번째 파티션(e)
--------	--

**예제 3-2. 디스크의 개념적인 모델**

이 모형도는 FreeBSD 의 관점에서 시스템에 붙어있는 첫 번째 IDE 디스크를 나타낸다. 디스크는 크기가 4GB 이고 두개의 2GB 슬라이스를(DOS 파티션) 가지고 있다고 가정한다. 첫 번째 슬라이스는 DOS 디스크 C:를 포함하고 두 번째 슬라이스는 FreeBSD 설치를 포함한다. 이 FreeBSD 설치 예제는 새개의 파티션과 스왑 파티션을 가지고있다.

새개의 파티션은 각 파일 시스템을 가지고있다. 파티션 a 는 root 파일 시스템에 e 는 /var 디렉터리 계층 그리고 f 는 /usr 디렉터리 계층에 사용된다.



---

## 3.6 파일 시스템 마운트와 언마운트

파일 시스템에서 / 는 나무의 뿌리로 root 디렉터리에있는 /dev, /usr 과 다른 디렉터리는 /usr/local 처럼 자신만의 줄기를 가지고있는 줄기로 가장 잘 표현할 수 있다.

분리된 파일 시스템에 이들 디렉터를 두는 여러가지 이유가 있다. /var 은 log/, spool/ 디렉터리와 여러 종류의 임시 파일들로 채워져 있을 것이다. root 파일 시스템을 채우는것은 좋지 않기 때문에 /에서 /var 을 나누는것이 유리하다.

다른 파일 시스템에 특정 디렉터리 트리를 포함시키는 또 다른 이유는 이들이 분리되어있는 물리적인 디스크 또는 네트워크 파일 시스템 마운트처럼 가상 디스크에 있거나 CDROM 드라이브에 있기 때문이다.

### 3.6.1 fstab 파일

부팅이 진행되는 동안 /etc/fstab 에 리스트된 파일 시스템은 자동으로 마운트된다 (*noauto* 옵션으로 리스트되어 있지 않다면)

```
# See the fstab(5) manual page for important information on automatic mounts
# of network filesystems before modifying this file.
#
# Device          Mountpoint      FStype  Options      Dump    Pass#
/dev/amrd0s1b     none            swap    sw           0       0
/dev/amrd0s1a     /               ufs     rw           1       1
/dev/amrd0s1h     /home           ufs     rw           2       2
/dev/amrd0s1e     /tmp            ufs     rw           2       2
/dev/amrd0s1g     /usr            ufs     rw           2       2
/dev/amrd0s1f     /var            ufs     rw           2       2
/dev/acd0c        /cdrom          cd9660  ro,noauto    0       0
proc              /proc           procfs  rw           0       0
```

/etc/fstab 파일은 다음 포맷과 같은 라인 리스트를 가지고있다:

*device* */mount-point* *fstype* *options* *dumpfreq* *passno*

*device*

장치 이름(필요하다). 16.2 장에서 설명한다.

*mount-point*

파일 시스템을 마운트할 디렉터리(필요하다).

---

### *fstype*

mount(8)에 적용할 파일 시스템 타입. 기본 FreeBSD 파일 시스템은 *ufs* 다.

### *options*

파일 시스템을 읽고 쓰는 *rw* 또는 파일 시스템을 읽기 전용으로 만드는 *ro* 처럼 필요에 의해 적용되는 옵션들. 보통 부팅하는 동안 마운트하지 않아야 되는 파일 시스템의 일반적인 옵션은 *noauto* 다. 다른 옵션은 mount(8) 매뉴얼 페이지에 리스트 되어있다.

### *dumpfreq*

이 옵션은 어떤 파일 시스템의 덤프가 필요한지 결정할 때 dump(8)가 사용한다. 필드에 아무것도 없다면 0 값으로 채워진다.

### *passno*

어떤 파일 시스템을 체크해야 되는지 순서를 결정한다. 파일 시스템 체크가 필요 없다면 그 파일 시스템의 *passno* 를 0 으로 지정한다. root 파일 시스템(다른 것 보다 먼저 체크해야 된다) *passno* 를 1 로 지정하고 다른 파일 시스템의 *passno* 는 1 보다 큰 값으로 지정해야한다. 하나 이상의 파일 시스템이 같은 *passno* 를 가지고 있다면 fsck(8)는 가능하다면 병렬로 파일 시스템을 체크한다.

## 3.6.2. mount 명령

mount(8) 명령은 궁극적으로 파일 시스템을 마운트할때 사용한다.

다음은 마운트 명령을 사용할 때 가장 일반적인 형식이다.

```
# mount device mountpoint
```

mount(8) 매뉴얼 페이지에 풍부한 옵션이 있지만 아주 일반적인 것은 다음과 같다.

### 마운트 옵션

*-a*

/etc/fstab 에 리스트 되어있는 모든 파일 시스템을 마운트한다. "noauto"로 표시된

---

것은 제외하고 *-t* 플래그나 이미 마운트된것은 배제시킨다.

*-d*

실제로 시스템에 마운트하는것을 제외하고 관련된 모든 것을 체크한다. 이 옵션은 *-v* 플래그와 같이 사용하여 실제로 `mount(8)`를 해야되는지 결정할 때 유용하다.

*-f*

결함이 있는 파일 시스템을(위험한) 강제로 마운트 하거나 파일 시스템의 마운트 상태를 읽기-쓰기에서 읽기 전용으로 바꿀때 쓰기 접근을 강제로 취소한다.

*-r*

파일 시스템을 읽기 전용으로 마운트한다. 이것은 *rdonly* 인자를 *-o* 옵션으로 사용하는 것과 같다.

*-t fstype*

주어진 파일 시스템 타입으로 원하는 파일 시스템을 마운트하거나 *-a* 옵션이 있다면 주어진 타입으로만 파일 시스템을 마운트한다.

"ufs"는 기본 파일 시스템 타입이다.

*-u*

파일 시스템에서 마운트 옵션을 업데이트한다.

*-v*

대화형으로 명령을 실행한다.

*-w*

파일 시스템을 읽고-쓰기로 마운트한다.

*-o* 옵션은 다음 옵션을 포함한 옵션 리스트를 쉼표(,)로 나누어서 적용한다.

`nodev`

파일 시스템에서 특별한 장치를 해석하지 않는다. 보안 옵션에 유용하다.

`noexec`

---

이 파일 시스템에서 바이너리를 실행하지 않는다. 이것도 보안 옵션에 유용하다.

nosuid

파일 시스템에서 `setuid` 나 `setgid` 플래그를 해석하지 않는다. 이것도 보안 옵션에 유용하다.

### 3.6.3 umount 명령

`umount(8)` 명령은 매개 변수로 마운트 포인트, 장치 이름 또는 `-a` 나 `-A` 옵션을 가지고 있다.

모든 형식은 강제로 언마운트하는 `-f`와 대화형으로 진행하는 `-v`를 가지고 있다. `-f`는 보통 좋은 아이디어는 아니다. 강제로 파일 시스템을 언마운트 하는것은 컴퓨터가 충돌됐거나 파일 시스템의 데이터가 파괴되었을 때다.

`-a`와 `-A`는 `-t` 뒤에 나열된 파일 시스템 타입 때문에 수정되었을 마운트된 모든 파일 시스템을 언마운트 하는데 사용한다. 그러나 `-A`는 root 파일 시스템은 언마운트하지 않는다.

## 3.5 프로세스

FreeBSD는 멀티태스킹 운영체제다. 이 말은 동시에 하나 이상의 프로그램이 운용되는 것처럼 보인다. 같은 시간에 동작하는 각 프로그램을 *프로세스*라고 한다. 실행할 모든 명령은 최소 하나의 새로운 프로세스를 시작하고 시스템 기능을 유지하기 위해 항상 실행 중인 많은 시스템 프로세스가 있다.

각 프로세스는 파일처럼 *process ID*나 *PID*라고 하는 유일한 번호로 인식되고 각 프로세스는 하나의 소유자와 그룹을 갖는다. 소유자와 그룹 정보는 이전에 말했듯이 파일 퍼미션을 사용하여 프로세스가 어떤 파일이나 장치를 열수 있는지 결정하는데 사용된다. 그리고 대부분의 프로세스는 부모 프로세스를 가지고 있다. 부모 프로세스는 이들 프로세스를 시작한 프로세스다. 예를들어 쉘에서 명령을 입력하였다면 그 쉘은 프로세스가 되고 여러분이 실행한 명령도 프로세스가 된다. 이런 식으로 실행한 각 프로세스는 부모



---

프로세스로 쉘을 가지고 있다. 예외가 되는 것은 `init` 라는 특별한 프로세스다. `init` 는 항상 최초의 프로세스기 때문에 `PID` 는 항상 1 이다. `init` 는 FreeBSD 가 시작될 때 커널에 의해 자동으로 시작된다.

두개의 명령 `ps(1)`와 `top(1)`은 시스템에서 프로세스를 볼때 특히 유용하다. `ps(1)` 명령은 현재 동작 중인 프로세스를 정적으로 리스트해서 볼때 사용하고 그들의 `PID` 와 얼마나 많은 메모리를 사용하고 있는지 그리고 프로세스를 시작한 명령 라인 등을 보여 줄수 있다. `top(1)` 명령은 운용 중인 모든 프로세스를 보여주고 몇초 간격으로 프로세스 리스트를 업데이트하기 때문에 컴퓨터가 무엇을 하는지 동적으로 볼수 있다.

기본적으로 `ps(1)`는 여러분이 실행한 명령만 보여 준다. 예를 들면 다음과 같은 결과를 볼수 있다.

```
% ps
  PID  TT  STAT      TIME COMMAND
   298  p0  Ss      0:01.10 tcsh
   7078  p0  S        2:40.88 xemacs mdoc.xsl (xemacs-21.1.14)
  37393  p0  I        0:03.11 xemacs freebsd.dsl (xemacs-21.1.14)
  48630  p0  S        2:50.89 /usr/local/lib/netcape-linux/navigator-linux-4.77.bi
  48730  p0  IW       0:00.00 (dns helper) (navigator-linux-)
  72210  p0  R+       0:00.00 ps
   390  p1  Is       0:01.14 tcsh
   7059  p2  Is+      1:36.18 /usr/local/bin/mutt -y
   6688  p3  IWs      0:00.00 tcsh
  10735  p4  IWs      0:00.00 tcsh
  20256  p5  IWs      0:00.00 tcsh
   262  v0  IWs      0:00.00 -tcsh (tcsh)
   270  v0  IW+      0:00.00 /bin/sh /usr/X11R6/bin/startx -- -bpp 16
   280  v0  IW+      0:00.00 xinit /home/nik/.xinitrc -- -bpp 16
   284  v0  IW       0:00.00 /bin/sh /home/nik/.xinitrc
   285  v0  S        0:38.45 /usr/X11R6/bin/sawfish
```

이 예제에서 보듯이 `ps(1)`의 출력은 수많은 칼럼으로 구성되어 있다. `PID`는 이미 말했듯이 프로세스 ID 다. `PID` 는 1 부터 99999 까지 할당되고 동작이 끝나면 다시 시작 위치로 되돌려진다. `TT` 칼럼은 프로그램이 동작하는 `tty` 를 보여주고 잠시동안 신경 쓰지 않아도 된다. `STAT`는 프로그램의 상태를 보여주고 역시 신경쓰지 않아도 된다. `TIME`은 CPU 에서 프로그램이 실행된 시간의 양을 보여준다. 어떤 프로그램은 CPU 를 사용할 필요가 있기 전까지 많은 시간동안 어떤 이벤트가 발생하기를 기다리기 때문에 프로그램을 시작한 시간부터 경과된 시간을 보여주는 것은 의미가 없다. 마지막으로 `COMMAND`는 프로그램을 실행한 명령 라인이다.

`ps(1)`는 보여주는 정보를 변경하는 여러가지 옵션을 지원한다. 아주 유용한 옵션 세트 중 하나는 `auxww`다. 여러분의 프로세스와 동작 중인 모든 프로세스에 대한 정보를 보여준다. `u`는 메모리 사용량과 프로세스 소유자의 이름을 표시한다. `x`는 데몬 프로세스에 대한

정보를 보여주고 `ww`는 `ps(1)`가 전체 명령어 라인을 보여주기 때문에 너무 긴 명령어 라인을 화면에 맞게 잘라서 보여준다.

`top(1)`의 출력도 비슷하다. 샘플 세션은 아래와 비슷하다.

```
% top
last pid: 72257; load averages: 0.13, 0.09, 0.03 up 0+13:38:33 22:39:10
47 processes: 1 running, 46 sleeping
CPU states: 12.6% user, 0.0% nice, 7.8% system, 0.0% interrupt, 79.7% idle
Mem: 36M Active, 5256K Inact, 13M Wired, 6312K Cache, 15M Buf, 408K Free
Swap: 256M Total, 38M Used, 217M Free, 15% Inuse

  PID USERNAME PRI NICE  SIZE  RES STATE   TIME  WCPU   CPU COMMAND
 72257 nik      28  0  1960K 1044K RUN     0:00 14.86%  1.42% top
  7078 nik       2  0 15280K 10960K select  2:54  0.88%  0.88% xemacs-21.1.14
   281 nik       2  0 18636K  7112K select  5:36  0.73%  0.73% XF86_SVGA
   296 nik       2  0  3240K  1644K select  0:12  0.05%  0.05% xterm
 48630 nik       2  0 29816K  9148K select  3:18  0.00%  0.00% navigator-linu
   175 root       2  0   924K   252K select  1:41  0.00%  0.00% syslogd
  7059 nik       2  0  7260K  4644K poll   1:38  0.00%  0.00% mutt
...
```

출력은 섹션 두개로 나뉘 진다. 헤더는(첫 번째 5 라인) 실행 중인 마지막 프로세스의 PID, 시스템 부하 평균(시스템의 사용량이 얼마나 많은지 보여주는) 그리고 시스템 업타임과(마지막 부팅부터 경과된 시간) 현재 시간을 보여준다. 헤더에 관련된 다른 숫자는 몇 개의 프로세스가 운용 중인지(이 경우 47 개), 메모리와 스왑 공간은 얼마나 사용 중인지와 다른 CPU 상태에서 얼마나 많은 시간을 보내고 있는지 보여준다.

아래는 `ps(1)`의 출력과 비슷한 정보를 가진 연속된 칼럼이 있다. 앞에서 보았듯이 PID, 유저 이름, CPU 를 사용한 시간 그리고 실행한 명령을 보여준다. 그리고 `top(1)`은 프로세스가 사용 중인 메모리 공간의 양을 기본적으로 보여준다. 이중 하나는 총 크기와 상주하고 있는 크기로 두개의 칼럼으로 나누어진다. 총 크기는 어플리케이션이 필요로하는 메모리의 양을 상주하고있는 크기는 최근 어느정도의 메모리를 실제로 사용하고 있는지 보여준다. 이 예제에서 **Netscape** 가 거의 30MB 의 메모리를 요구하는것을 볼수 있지만 현재 9MB 만 사용하고 있다.

`top(1)`은 이 화면을 자동으로 매 2 초마다 업데이트한다. 이것은 `s` 옵션으로 바꿀 수 있다.

### 3.8 데몬, 시그널 그리고 프로세스 죽이기

사용하기 쉬운 에디터를 실행할 때 불러올 파일을 지시할 수 있다. 에디터가 이 기능을 제공하고 에디터는 터미널에 있기 때문에 이렇게 할수 있다. 어떤 프로그램은 계속해서 유저가 입력할 수 있도록 디자인되지 않았기 때문에 첫 번째에서 터미널 연결이 해제된다.

---

예를들어 웹 서버는 모든 시간을 웹 요청에 응답하는데 시간을 보내기 때문에 여러분의 입력이 필요없다. 사이트에서 사이트로 메일을 전송하는 프로그램도 이런 유형의 또 다른 예가 된다.

우리는 이런 프로그램을 *데몬*이라고 한다. 데몬은 그리스 신화의 캐릭터들이다. 그들은 선도 악도 아닌 영혼으로서 인류를 위해 간접적으로 크고 유용한 일들을 했다. 요즘의 웹 서버와 메일 서버처럼 유용하다. 이것이 아주 오랜시간 동안 왜 BSD 마스코트가 고무로된 장화를 신고 3 갈퀴의 포크를 가진 유쾌한 악마의 모습을 가지고 있는지 설명해 준다.

데몬으로 실행하는 프로그램의 이름에는 보통 "d"가 붙는 관례가 있다. **BIND**는 버클리 인터넷 네임 데몬(실행하는 실제 프로그램은 `named`라고 한다), Apache 웹 서버 프로그램은 `httpd`라고 부르고, 라인 프린터 스플링 데몬은 `lpd`라고 한다. 이것은 꼭 지켜야 되는 규칙이 아닌 관례다. 예를들어 샌드메일 어플리케이션의 메인 메일 데몬은 여러분이 상상하듯이 `maild`라고 하지않고 `sendmail`이라고 한다.

가끔 데몬 프로세스와 통신을 해야된다. 이런 통신을 *신호*라고 하고 신호를 보내서 데몬과(또는 실행 중인 프로세스) 통신을 할수 있다. 데몬에게 보낼 수 있는 여러가지 신호가 있다. 그 중 몇가지는 특별한 의미를 가지고 있고 다른 신호는 어플리케이션이 해석하기 때문에 어플리케이션 문서에서 어플리케이션이 신호를 어떻게 해석할지 알려준다. 여러분이 소유하고있는 프로세스에게만 신호를 보낼 수 있다. 다른 사람의 프로세스에 `kill(1)`이나 `kill(2)` 신호를 보내면 거부당한다. 여기서 예외는 모든 사용자의 프로세스에 신호를 보낼 수 있는 `root` 유저다.

FreeBSD 도 어떤 경우에 어플리케이션에게 신호를 보낸다. 어플리케이션 상태가 이상하고 예상치 못한 메모리 접근을 시도한다면 FreeBSD 는 그 프로세스에게 *Segmentation Violation* 신호를(`SIGSEGV`) 보낸다. 어플리케이션이 얼마간의 시간이 지나고 경고를 하기위해 `alarm(3)` 시스템 콜을 사용하고 있었다면 알람 신호를(`SIGALRM`) 계속해서 보낸다.

`SIGTERM`과 `SIGKILL` 두 개의 신호는 프로세스를 중지시키는데 사용할 수 있다. `SIGTERM`은 프로세스를 중지시키는 안전한 방법이다. 프로세스는 이 신호를 감지할 수 있고 섀다운 하기를 원한다는 것을 확인하여 열려있었던 로그 파일을 닫고 무엇이든 하고있던 작업을 섀다운 하기전에 끝낸다. 인터럽트 할수 없는 작업의 중간에 있다면 프로세스는 `SIGTERM` 신호를 무시할 수도 있다.

`SIGKILL`은 프로세스가 무시할 수 없다. 이것은 "네가 무슨 일을 하든 상관하지 않는다."

---

지금 바로 중단하라"는 신호다. *SIGKILL* 신호를 프로세스에게 보낸다면 FreeBSD 는 그 프로세스를 바로 중단시킨다.

**프로세스를 바로 중단시킨다:** 인터럽트 할 수 없는 몇 가지가 경우가 있기 때문에 이 말은 사실이 아니다. 예를들면 네트워크에 있는 다른 컴퓨터의 파일을 읽으려는 프로세스와 어떤 이유로 다른 컴퓨터가 없어졌다(꺼거나 네트워크에 문제가 생겼다면) 프로세스는 “인터럽트 할 수 없음”이라는 메시지를 보여준다. 결국 프로세스는 몇분 후 타임 아웃이 된다. 이 타임 아웃이 떨어지자 마다 프로세스는 중지 된다.

사용하려는 다른 신호는 *SIGHUP*, *SIGUSR1* 과 *SIGUSR2* 일 것이다. 이들은 일반적인 목적의 신호기 때문에 이들 신호가 보내졌을 때 다른 어플리케이션은 다른 동작을 할 것이다.

웹 서버의 설정 파일을 변경했다고 가정한다. 웹 서버가 설정 파일을 다시 읽기를 원할 것이다. *httpd* 를 중지하고 다시 시작할 수 있지만 원하지 않는 짧은 시간 동안 웹 서버를 중단시킨다. 대부분의 데몬은 *SIGHUP* 신호의 응답으로 설정 파일을 다시 읽도록 작성되었기 때문에 *httpd* 를 죽이고 다시 시작하는 대신 *SIGHUP* 신호를 보낸다. 이들 신호의 응답에는 표준적인 방법이라는 것이 없어서 다른 데몬은 다른 동작을 하기 때문에 이 질문은 해당 데몬 문서를 읽도록 한다.

#### [프로세스에게 신호 보내기]

이 예제는 *inetd(8)*에게 어떻게 신호를 보내는지 보여준다. *inetd(8)*의 설정 파일은 */etc/inetd.conf* 이고 *SIGHUP*를 보내면 *inetd(8)*은 이 설정 파일을 다시 읽는다.

- ① 신호를 보내려는 프로세스의 프로세스 ID 를 찾는다. *ps(1)*과 *grep(1)*을 사용하면 된다. *grep(1)* 명령은 출력에서 명시된 문자열을 찾는데 사용된다. 이 명령은 일반 유저에서 실행하고 *inetd(8)*은 root 로 실행되었기 때문에 *ps(1)*에 *ax* 옵션을 꼭 지정해야 된다.

```
% ps -ax | grep inetd
```

```
198 ?? IWs 0:00.00 inetd -wW
```

따라서 *inetd(8)* PID 는 198 이다. 어떤 경우 *grep inetd* 명령도 똑같은 출력 결과를 만든다. 이것도 *ps(1)*이 실행 중인 프로세스 리스트를 찾는 방법이기 때문이다.

② kill(1)로 신호를 보낸다. inetd(8)은 root 로 실행되었기 때문에 su(1)를 사용하여 root 가 되어야 한다.

```
% su
```

```
Password:
```

```
# /bin/kill -s HUP 198
```

대부분의 일반적인 유닉스 명령에서 성공했다면 kill(1)은 아무것도 출력하지 않는다. 소유권이 없는 프로세스에게 신호를 보낸다면 “kill: *PID*: Operation not permitted”라는 메시지를 보게된다. PID 를 잘못 입력했다면 나쁜 상황이 될수 있는 다른 프로세스에게 신호를 보내거나 운이 좋다면 현재 사용하지 않는 PID 에게 신호를 보낼 것이고 “kill: *PID*: No such process”를 보게된다.

**왜 /bin/kill 을 사용해야 되는가?** 많은 셸은 내장된 명령에서 kill 명령을 제공한다. 따라서 셸은 /bin/kill 을 실행하지않고 직접 신호를 보낸다. 이 방법은 매우 유용하지만 다른 셸들은 신호를 보내려는 이름을 다른 형식의 구문으로 지정한다. 모든 것들을 배우는것 보다 /bin/kill 명령을 직접 사용하는 것이 훨씬 단순할 수 있다.

다른 신호를 보내는 것도 매우 비슷하기 때문에 명령어 라인에서 *TERM*이나 *KILL* 대신 필요한 것을 사용한다.

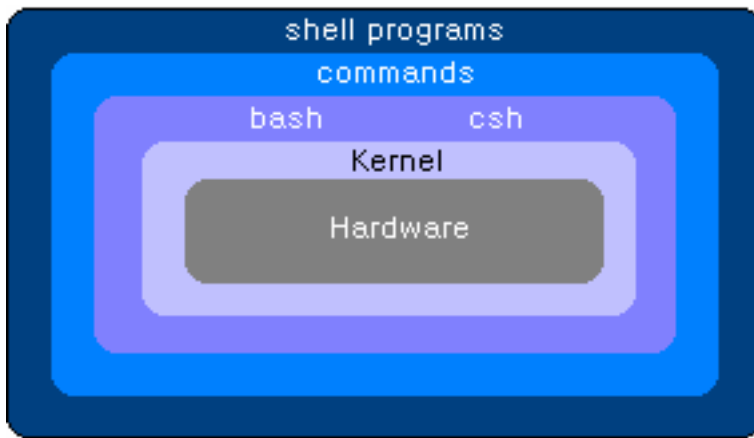
**중요:** 시스템의 프로세스를 함부로 죽이는 것은 상당히 위험하다. 특히 프로세스 ID 1 init(8)은 매우 특별하다. /bin/kill -s KILL 1 을 실행하는 것은 시스템을 셧다운하는 빠른 방법이다. **Return** 을 누르기 전에 kill(1)에 사용하는 인자를 항상 두번씩 체크 한다.

### 3.9 셸

FreeBSD 에서 메일 수행하는 수많은 작업은 셸 이라는 명령어 라인 인터페이스로 수행된다. 셸의 주된 기능은 입력 채널로 들어온 명령을 실행하는 것이다. 또한 셸에는 파일 관리, 파일 검색, 명령어 라인 편집, 명령 매크로와 환경 변수처럼 메일 수행하는 업무를 돕는 기능이 내장되어 있다. FreeBSD 는 본셸 sh 과 C-셸을 발전시킨 tcsh 같은 셸 세트를 가지고 있다. zsh 와 bash 같은 다른 셸은 FreeBSD 포트 컬렉션에서 이용할 수 있다.

---

## 운영체제에서 셸의 위치



어떤 셸을 사용할 것인가? 이것은 실제로 선호도 문제다. C 프로그래머라면 tcsh 처럼 C 와 호환성있는 셸을 편하게 느낄 것이다. 리눅스를 사용했거나 유닉스 명령어 라인이 처음이라면 bash 를 사용해 본다. 각 셸은 선호하는 작업 환경과 맞거나 안 맞을 수 있는 고유한 특징을 가지고 있으므로 어떤 셸을 사용할지 선택해야 된다.

셸의 일반적인 특징 중 한가지는 파일 이름 완성이다. 명령이나 파일 이름의 처음 몇 개의 문자가 주어지면 보통 키보드의 **Tab** 키를 눌러서 나머지 명령이나 파일 이름을 셸이 자동으로 완성시킨다. 여기 예제가 있다. foobar 와 foo.bar 라고 하는 두 개의 파일이 있다고 가정한다. foo.bar 를 지우기 위해 키보드에 입력해야 되는 것은 **rm fo** [Tab].[Tab]이다.

셸은 **rm fo**[BEEP].bar 를 출력할 것이다.

한 개 이상의 매칭되는 파일이 있기 때문에 파일 이름을 완전히 완성할 수 없다고 셸이 말하는 [BEEP]는 콘솔 벨이다. foobar 와 foo.bar 둘다 fo 로 시작되지만 *foo*로 완성될 수 있다. .을 입력하고 **Tab** 를 다시 누르면 셸이 나머지 파일 이름을 완성시켜 준다.

셸의 다른 특징은 환경 변수의 사용이다. 환경 변수는 변수 키 쌍이 셸의 환경 공간에 저장되어있는 것이다. 셸이 호출한 프로그램이 이 공간을 읽을 수 있기 때문에 수많은 프로그램 설정을 가지고 있다. 다음 표는 일반적인 환경 변수 리스트와 그들이 무엇을 의미하는지를 보여준다.

변수	설명
USER	현재 로그인한 유저 이름

PATH	바이너리를 찾기 위해 콜론(:)으로 나누어진 디렉터리 리스트
DISPLAY	가능하다면 연결하려는 x11 화면의 네트워크 이름
SHELL	현재 셸
TERM	유저 터미널 이름. 터미널의 특성을 결정하는데 사용된다.
TERMCAP	다양한 터미널 기능을 수행하기 위해 터미널을 벗어나는 코드의 데이터베이스 엔트리
OSTYPE	운영체제 타입. 예를 들어 FreeBSD.
MACHTYPE	시스템이 실행 중인 CPU 아키텍처
EDITOR	유저가 선호하는 텍스트 에디터
PAGER	유저가 선호하는 텍스트 페이지
MANPATH	매뉴얼 페이지를 찾는 콜론(:)으로 나누어진 디렉터리 리스트

환경 변수 지정은 셸별로 약간씩 다르다. 예를 들어 tcsh 나 csh 처럼 C 스타일의 셸에서 환경 변수 지정은 setenv 를 사용한다. sh 나 bash 처럼 본셸과 비슷한 셸에서 현재 환경 변수 지정은 export 을 사용한다. 예를 들어 csh 나 tcsh 에서 EDITOR 환경 변수를 /usr/local/bin/emacs 로 설정하는 것은 다음과 같다.

**% setenv EDITOR /usr/local/bin/emacs**

본셸에서는 다음명령을 사용한다.

**% export EDITOR="/usr/local/bin/emacs"**

또한 명령어 라인에서 환경 변수 앞에 \$ 문자를 두어서 대부분의 셸을 확장할 수 있다. 예를 들어 \$TERM 에 echo 를 적용해서 셸이 확장되기 때문에 echo \$TERM 은 \$TERM 이 설정된 상태를 출력한다.

셸은 메타 문자라고 부르는 수많은 특수 문자를 특별한 데이터 표현으로 간주한다. 아주 일반적인 것은 파일 이름에서 여러 문자를 표현하는 \* 문자다. 이런 특수 메타 문자는 파일 이름 검색에 사용된다. 예를 들어 echo \*라고 입력하는 것은 ls 라고 입력하는 것과 거의 같다. 왜냐하면 셸은 \*와 매칭되는 모든 파일을 echo 로 볼수 있도록 명령어 라인에 놓기 때문이다.

이런 특수 문자를 셸이 해석하지 않도록 하려면 특수 문자 앞에 백슬래시(\\) 두면 된다. echo \$TERM 터미널 설정을 출력한다. echo \\\$TERM 도 \$TERM 과 같은 내용을 출력한다.

---

### 3.9.1 셸 변경

셸을 변경하는 가장 쉬운 방법은 chsh 명령을 사용한다. chsh 를 실행하면 EDITOR 환경 변수에 지정한 에디터로 들어간다; 지정되지 않았다면 vi 로 들어간다. “Shell:” 라인을 적절히 변경한다

또한 chsh 에 -s 옵션을 줄 수 있다; 이것은 에디터에 입력할 필요없이 셸을 지정한다. 예를들어 셸을 bash 로 변경하려면 다음 명령을 실행한다.

```
% chsh -s /usr/local/bin/bash
```

chsh 를 아무런 매개 변수없이 실행하고 셸을 편집해도 된다.

```
#Changing user database information for kidp.  
Shell: /bin/csh  
Full Name: User &  
Office Location:  
Office Phone:  
Home Phone:  
Other information:
```

**Note:** 사용하려는 셸은 /etc/shells 파일에 있어야 된다. 포트 컬렉션에서 셸을 설치했다면 /etc/shells 파일에 셸이 존재한다. 직접 셸을 설치했다면 /etc/shells 파일에 직접 입력해야 된다.

예를들어 직접 bash 를 설치하고 bash 가 /usr/local/bin/에 있다면 다음과 같이 입력한다.

```
# echo "/usr/local/bin/bash" >> /etc/shells
```

그리고 다시 chsh 를 실행한다.

### 3.10 텍스트 에디터

FreeBSD 에서 다양한 설정은 텍스트 파일을 편집하면 된다. 이런 이유로 텍스트 에디터를



능숙하게 다루는것이 많은 도움이된다.

배우기 가장 쉽고 가장 단순한 에디터는 쉬운(easy) 에디터(editor)를 나타내는 **ee** 라는 에디터다. **ee** 를 시작하려면 명령어 라인에서 *ee filename* 을 입력한다. *filename* 은 편집하려는 파일 이름이다. 예를들어 /etc/rc.conf 를 편집하려면 ee /etc/rc.conf 를 입력한다. ee 에서 에디터의 기능을 사용하기 위한 모든 명령은 화면 상단에 리스트 되어있다.

```
^_ (escape) menu    ^y search prompt    ^k delete line      ^p prev li          ^g prev page
^o ascii code      ^x search            ^l undelete line    ^n next li          ^v next page
^u end of file      ^a begin of line     ^w delete word      ^b back 1 char
^t begin of file    ^e end of line       ^r restore word     ^f forward 1 char
^c command          ^d delete char       ^j undelete char    ^z next word
: 1 C: 1 =====
# -- sysinstall generated deltas -- # Sat Sep 11 02:44:14 2004
# Created: Sat Sep 11 02:44:14 2004
# Enable network daemons for user convenience.
# Please make all changes to this file, not to /etc/defaults/rc.conf.
# This file now contains just the overrides from /etc/defaults/rc.conf.
```

^ 문자는 키보드의 **Ctrl** 키를 의미하기 때문에 ^e 키 조합은 **Ctrl+e** 가 된다. ee 에서 나가려면 **Esc** 키를 누르고 에디터에서 떠나기를 선택한다. 파일을 수정했다면 에디터는 변경한 것을 저장할 것인지 묻는 프롬프트를 표시한다.

```
main menu
a) leave editor
b) help
c) file operations
d) redraw screen
e) settings
f) search
g) miscellaneous
press Esc to cancel
```

또한 FreeBSD 는 기본 시스템에서 vi 같은 더욱 강력한 에디터를 제공하지만 emacs 와 vim 같은 다른 에디터는 FreeBSD 포트 컬렉션에서 설치할 수 있다. 이런 에디터는 강력한 기능을 제공하지만 배우기에는 더 복잡하다. 그러나 수 많은 텍스트를 편집할 계획이라면 vim 이나 emacs 같은 더 강력한 에디터를 배우는 것이 오랜 기간 동안 시간을 절약할 수 있다.

### 3.11 장치와 장치 노드

장치는 디스크, 프린터, 그래픽 카드와 키보드를 포함하는 대부분 시스템 하드웨어와

---

관련하여 사용하는 용어다. FreeBSD 가 부팅할 때 주로 표시하는 것은 검색된 장치다. /var/run/dmesg.boot 에서 부트 메시지를 다시 볼수 있다.

예를 들어 acd0 는 첫 번째 IDE CDROM 드라이브지만 kbd0 는 키보드를 나타낸다.

유닉스 시스템에서 이들 대부분의 장치는 /dev 디렉터리에있는 장치 노드라고하는 특별한 파일을 통해 접근해야 된다.

### 3.11.1 장치 노드 생성

시스템에 새로운 장치를 추가하거나 추가적인 장치를 지원하도록 컴파일할 때 새로운 장치를 위해 하나 또는 하나 이상의 장치 노드를 생성해야 된다.

#### 3.11.1.1 MAKEDEV 스크립트

*DEVFS*가 없는 시스템에서(이것은 5.0 이전의 모든 FreeBSD 버전을 의미한다.) 장치 노드는 아래에서 보여주는 것처럼 MAKEDEV(8) 스크립트를 사용해서 생성한다.

```
# cd /dev
# sh MAKEDEV ad1
```

이 예제는 설치할 때 두 번째 IDE 드라이브에 적절한 장치 노드를 생성한다.

#### 3.11.1.2 DEVFS (장치 파일 시스템)

장치 파일 시스템 또는 *DEVFS*는 글로벌 파일 시스템 이름 공간에서 커널의 장치 이름 공간에 접근하도록 한다. 장치 노드를 생성하고 수정하는 대신 *DEVFS*은 이 특별한 파일 시스템을 관리한다.

더 많은 정보는 devfs(5) 매뉴얼 페이지를 본다.

*DEVFS*는 FreeBSD 5.0 과 이후 버전에서 기본적으로 사용된다.

---

## 3.12 바이너리 포맷

FreeBSD 가 왜 ELF 포맷을 사용하는지 이해하려면 우선 현재 유닉스에 실행할 수 있는 3 가지 주요 포맷에 대해 어느 정도 알고 있어야 된다.

- out(5)

가장 오래되고 고전적인 유닉스 오브젝트 포맷. 특정 포맷에(더 자세한 사항은 a.out(5)를 본다) 종종 사용되며 시작 부분에서 매직 번호(magic number)로 짧고 조밀한 헤더를 사용한다. 3 개의 로드된 세그먼트를 포함한다: 심볼 테이블과 문자열 테이블을 더한 .text 그리고 .data 와 .bss

- COFF

SVR3 오브젝트 포맷. 이제 헤더는 섹션 테이블을 포함하기 때문에 .text, .data 와 .bss 보다 많은 섹션을 가질 수 있다.

- elf(5)

멀티 섹션과 32-bit 또는 64-bit 값을 가능하도록 하는 기능으로 COFF 를 계승한다. 한가지 주요 결정: ELF 도 시스템 구조당 ABI 가 하나만 있다고 가정하고 디자인되었다. 이 가정은 실제로 전혀 맞지않고 상용 SYSV 세계에서(최소한 3 개의 ABI 를 갖는: SVR4, Solaris, SCO) 사실이 아니다.

FreeBSD 는 ABI 에 관한 정보로 실행할 수 있는, ELF 라고 하는 branding 유틸리티를 제공해서 이 문제를 다소 해결하였다. 더 많은 정보는 brandelf(1) 매뉴얼 페이지를 본다.

FreeBSD 는 “클래식” 캠프와 a.out(5) 포맷에 사용된 기술을 시도하여 3.X 분기가 시작될 때까지 사용하여 많은 세대의 BSD 릴리즈를 거치는 동안 증명되었다. 가끔 그 이전 FreeBSD 시스템에서 본래의 ELF 바이너리를(그리고 커널) 빌드해서 실행할 수 있었지만 FreeBSD 는 처음에 기본 포맷을 ELF 로 바꾸는 것을 반대했다. 왜냐하면 리눅스 캠프가 어렵게 ELF 로 변경할때 밴더와 마찬가지로 개발자들에게도 공유 라이브러리 구조를 매우 어렵게 만드는, 그 들의 유연성 없는 jump-table 기반 공유 라이브러리 메커니즘 때문에 a.out 의 실행 가능 포맷을 변경하기에 적절하지 않았다. ELF 툴을 공유 라이브러리 문제에

---

제공된 솔루션으로 이용할 수 있었기 때문에 "진행되는" 것처럼 보였고 어쩌든 필요한 변화를 수용하기 위한 이식 비용도 허용되었다. FreeBSD의 공유 라이브러리 메커니즘은 Sun의 SunOS 스타일 공유 라이브러리 메커니즘에 더욱 가까웠기 때문에 사용하기 쉽다.

그런데 왜 다양한 포맷이 있는가?

단순한 하드웨어만 있던 오래 전의 어두운 시대에 이 단순한 하드웨어는 간단하고 작은 시스템만 지원했었다. a.out은 이 단순한 시스템에서(PDP-11) 바이너리를 표현하는 업무에 완벽하게 적합했다. 이전 모토로라 68k와 VAXen 같은 아키텍처의 유닉스 포트에 적합했기 때문에 사람들은 이 단순한 시스템에서 유닉스로 포트할 때 a.out 포맷을 남겨 두었다.

그리고 어떤 뛰어난 하드웨어 엔지니어는 알팍한 트릭을 강제로 소프트웨어에 적용할 수 있다면 디자인의 한계를 깨고 CPU core가 빠르게 동작하게 할 수 있다고 생각했다. 새로운 종류의 하드웨어(요즘의 RISC로 알려진) 등장했지만 a.out은 이 하드웨어에 적합하지 않았기 때문에 단순하고 제한적인 a.out 포맷이 제공할 수 있는 것보다 이 하드웨어에서 더 좋은 성능을 얻을 수 있는 여러 포맷이 개발되었다. COFF, ECOFF와 다른 몇 개의 포맷이 개발되어서 ELF에 탑재하기 전에 성능 테스트가 되었다.

게다가 프로그램 크기는 계속 거대해지고 디스크는(그리고 물리적인 메모리) 아직 상대적으로 작았기 때문에 공유 라이브러리 개념이 탄생했다. VM 시스템도 더욱 복잡해졌다. 이들 발전 양상 중 각각은 a.out 포맷을 사용했지만 새로운 기능은 더욱 유용해졌다. 게다가 사람들은 실행하려고 할때 동적으로 로드되거나 init 코드가 실행된 후 필요없는 자원이 코어 메모리와 스왑 공간으로 되돌려 지기를 원했다. 언어가 더욱 복잡해졌기 때문에 사람들은 자동으로 실행되기 전에 코드를 불러와야 됐다. 많은 해커는 a.out 포맷으로 이런 모든 일을 처리하도록 수많은 시간을 작업했다. 이 시간 동안 a.out은 더욱 복잡한 코드 없이는 이런 문제를 제어하지 못했다. ELF는 이들 문제를 해결하였지만 기본적으로 동작하는 시스템에서 변경하는 것이 어려웠기 때문에 ELF로 이동하는 것보다 a.out을 유지하는 것이 더욱 어려워 질 때까지 ELF는 기다려야 됐다.

그러나 시간이 지나자 두 개의 병렬 트리에서 파생된 빌드 툴이(특히 어셈블러와 로더) 이들의 빌드 툴이 되었다. FreeBSD 트리에 공유 라이브러리가 추가되었고 몇몇 버그가 수정되었다. 최초로 이들 프로그램을 작성한 GNU 사람들은 크로스 컴파일러, 다른 포맷 플러그인 등 간단한 지원을 추가하여 이들 프로그램을 재 작성 했다. 많은 사람들이 FreeBSD를 목적으로 크로스 컴파일러 빌드를 원했지만 FreeBSD가 가지고 있는 오래된 소스 **as**와 **ld**가 상위 업무에 맞지 않았다. 새로운 GNU 툴들은(binutils) 크로스 컴파일, ELF, 공유 라이브러리 그리고 C++ 확장 등을 지원했다. 게다가 많은 벤더들이 ELF

---

바이너리를 릴리즈했기 때문에 FreeBSD 가 이들을 실행하기에 좋았다. ELF 는 a.out 보다 풍부하고 기본 시스템에서 더욱 확장할 수 있다. ELF 툴들은 유지하기 좋고 많은 사람들에게 중요한 크로스 컴파일을 지원한다. ELF 는 a.out 보다 약간 느리겠지만 큰 차이는 보이지 않는다. 또한 어떻게 이 두 가지가 페이지를 맵으로 해서 init 코드를 제어하는 등의 여러 가지 차이가 있다. 모두 중요하지는 않지만 차이가 있다. a.out 지원은 GENERIC 커널로 이동되고 레거시(legacy) a.out 프로그램의 필요성이 없다면 결국 커널에서 삭제될 것이다.

## 3.13 더 많은 정보

### 3.13.1 매뉴얼 페이지

FreeBSD 관한 더 포괄적인 문서는 매뉴얼 페이지 형식으로 되어있다. 시스템의 거의 모든 프로그램은 기본적인 사용법과 다양한 인자를 설명하고 있는 짧은 레퍼런스 매뉴얼을 가지고 있다. 이들 매뉴얼은 man 명령으로 볼수 있다. man 명령의 사용은 간단하다.

`% man command`

*command*는 배우려는 명령의 이름이다. 예를 들어 ls 명령에 대해 더 많이 배우려면:

`% man ls` 라고 입력한다.

온라인 매뉴얼은 여러 개의 섹션으로 나누어있다:

1. 유저 명령
2. 시스템 콜과 에러 번호
3. C 라이브러리에서의 함수
4. 장치 드라이버
5. 파일 포맷
6. 게임과 다른 오락도구
7. 잡다한 정보
8. 시스템 유지와 운용 명령
9. 커널 개발자

---

어떤 경우 온라인 매뉴얼의 하나 이상의 섹션에 같은 주제가 나타날 수 있다. 예를 들면 `chmod` 유저 명령과 `chmod()` 시스템 콜이 있다. 이 경우 원하는 섹션을 지정하여 `man` 명령을 사용할 수 있다.

**% man 1 chmod**

이것은 유저 명령 `chmod` 매뉴얼 페이지를 보여준다. 온라인 매뉴얼의 특정 섹션을 조회하는 것은 작성된 문서에서 전통적으로 괄호안에 두기 때문에 `chmod(1)`은 `chmod` 유저 명령을 `chmod(2)`는 시스템 콜을 조회한다.

명령어의 이름을 알고 있고 단순히 이 명령을 어떻게 사용하는지 알고 싶다면 괜찮지만 명령어 이름을 모른다면? 명령이 설명된 곳에서 `-k` 를 사용하여 키워드 검색을 할수 있다.

**% man -k mail**

이 명령으로 설명에 "mail" 이라는 키워드를 가진 명령어 리스트를 볼수 있다. 이것은 실제로 `apropos` 명령을 이용하는 것과 같다.

그래서 이런 식으로 원하는 명령을 `/usr/bin` 에서 찾을 수 있지만 이들이 무엇을 하는지 알 수 없다면 단순히 다음 명령을 사용한다:

**% cd /usr/bin**

**% man -f \***

또는 다음 명령을 사용한다:

**% cd /usr/bin**

**% whatis \***

### 3.13.2 GNU Info 파일

FreeBSD 는 자유 소프트웨어 협회에서(FSF) 만든 수많은 어플리케이션과 유틸리티를

---

가지고 있다. 이들 프로그램은 info 명령이나 emacs 를 설치하였다면 emacs 의 info 모드에서 볼수 있는 info 파일이라는 광범위한 하이퍼 텍스트 문서를 가지고 있다.

info(1) 명령의 사용은 단순히 다음 명령을 입력한다:

**% info**

간략한 소개는 *h* 를 입력한다. 빠른 명령어 레퍼런스는 *?* 를 입력한다.

---

## 4 장 어플리케이션 설치: 패키지와 포트

### 4.1 개요

이번 장에서는 FreeBSD 가 가지고 있는 특별한 기능인 패키지와 포트에 대해 알아보자. FreeBSD 는 다양한 시스템 툴을 기본적으로 가지고 있다. 그렇지만 이런 어플리케이션을 사용하기 위해 많은 준비가 필요하다. FreeBSD 는 시스템에 소프트웨어를 설치하는 포트 컬렉션과 바이너리 패키지라는 두 가지 기술을 제공한다. 여러분은 이 두 가지 기술을 사용하여 로컬 미디어(FreeBSD CD-ROM 이나 DVD-ROM)나 네트워크에서 최신 버전의 원하는 어플리케이션을 설치할 수 있다.

이번 장을 읽고 다음과 같은 사항을 알 수 있다:

- 바이너리 소프트웨어(어플리케이션) 패키지를 어떻게 설치하는가
- 포트 컬렉션에서 어떻게 소프트웨어를 설치할 수 있는가
- 이전에 설치한 패키지나 포트는 어떻게 삭제하는가
- 포트 컬렉션이 사용하는 기본적인 옵션을 어떻게 무시할 수 있는가
- 포트는 어떻게 업그레이드하는가

### 4.2 소프트웨어 설치 개요

이전에 유닉스나 리눅스 시스템을 사용해 보았다면 소프트웨어를 설치하는 전형적인 절차를 알고 있을 것이다. Z 쉘 설치를 예로 들어 보자:

- ① 웹 사이트(<http://www.zsh.org>) 또는 FTP 사이트(<ftp://ftp.zsh.org/zsh>)에서 소스 코드나 바이너리 형식의 소프트웨어를 다운로드한다. 여기서는 `zsh-4.2.1.tar.gz` 파일을 다운로드했다. 그리고 이 파일을 적당한 디렉터리로 이동시킨다.

```
# mv zsh-4.2.1.tar.gz /tmp
```



- 
- ② 다운로드 한 소프트웨어 형식이 압축되었다면 (보통 compress(1), gzip(1)이나 bzip2(1)형식으로 타볼로 압축된) 압축을 푼다:

```
# tar xzvf zsh-4.2.1.tar.gz
```

- ③ 압축을 푼 디렉터리의 문서 파일에서(보통 INSTALL 이나 README 파일, 또는 doc/ 서브 디렉터리에 있는 어떤 파일) 소프트웨어 설치 방법, 주의 사항과 옵션 등을 확인한다.

```
# cd zsh-4.2.1
# more INSTALL
```

- ④ 소스 형식의 소프트웨어를 다운로드하였다면 컴파일한다. 여기서 Makefile 을 수정하거나 configure 스크립트 실행 같은 다른 작업이 필요할 것이다:

```
# ./configure
```

- ⑤ 에러가 없는지 테스트하고 소프트웨어를 설치한다.

```
# make
# make install
```

위의 모든 절차가 정상이라면 다행이지만, FreeBSD 에 정확히 포트되지 않은(다시 말해 FreeBSD 시스템에 맞게 수정되지 않았다면) 소프트웨어 패키지를 설치하려면 코드를 적절히 수정해야 된다.

여러분들이 원한다면 위에서 설명한 “전통적인” 방법으로 소프트웨어를 계속 설치할 수 있다. 그렇지만 FreeBSD 는 여러분들의 수고를 덜어 주는 패키지와 포트라는 두 가지 기술을 제공하고 있다. 이 글을 쓰고 있을 때 10,550 개 이상의 어플리케이션을 이 두 가지 방법으로 설치할 수 있었다.

어떤 어플리케이션이라도 이 어플리케이션의 FreeBSD 패키지 파일 하나만 다운로드하면 된다. 패키지는 어플리케이션의 설정 파일이나 문서, 그리고 미리 컴파일 되어 있는 모든 명령어 세트를 가지고 있다. 다운로드한 패키지 파일은 pkg\_add(1), pkg\_delete(1), pkg\_info(1) 등의 FreeBSD 패키지 관리 명령으로 제어할 수 있다. 그래서 새로운 어플리케이션을 설치하려면 명령 하나만 실행하면 된다.

FreeBSD 포트는 소스 코드에서 어플리케이션을 자동으로 컴파일하도록 디자인되어 있는

---

파일의 모음이다.

만약 프로그램을 직접 컴파일한다면(파일을 다운로드해서 압축을 풀고 소스를 패치한 후 컴파일하여 설치한다) 일반적으로 수행해야 되는 여러 단계가 있다는 것을 기억한다. 포트로 만들어진 파일은 시스템이 수행해야 되는 필요한 정보를 모두 가지고있다. 그래서 여러분이 간단한 명령을 직접 실행하면, 시스템이 알아서 어플리케이션 소스 코드를 자동으로 다운로드해서 압축을 풀고 소스를 패치한 후 컴파일하여 설치한다.

그리고 나중에 간단히 설명하는 패키지 관리 명령으로 제어되는 패키지를 만들 때도 포트 시스템을 사용할 수 있다.

패키지와 포트는 **의존성**을 이해하고 있다. 여러분들이 직접 설치해야 되는 특정 라이브러리에 의존되는 어플리케이션을 설치하려 한다고 가정한다. 다행히 어플리케이션과 라이브러리는 FreeBSD 포트와 패키지로 설치할 수 있도록 되어 있다. 어플리케이션을 추가하는데 pkg\_add 명령이나 포트 시스템을 사용한다면, 둘 다 이 라이브러리가 설치되지 않았다고 알려 주고 다음과 같이 자동으로 라이브러리를 먼저 설치해준다.

```
-----  
Libraries have been installed in:  
/usr/local/lib
```

```
If you ever happen to want to link against installed libraries  
in a given directory, LIBDIR, you must either use libtool, and  
specify the full pathname of the library, or use the '-LLIBDIR'  
flag during linking and do at least one of the following:
```

- add LIBDIR to the 'LD\_LIBRARY\_PATH' environment variable during execution
- add LIBDIR to the 'LD\_RUN\_PATH' environment variable during linking
- use the '-Wl,--rpath -Wl,LIBDIR' linker flag

```
See any operating system documentation about shared libraries for  
more information, such as the ld(1) and ld.so(8) manual pages.  
-----
```

앞의 두 가지 기술이 비슷하기 때문에, 왜 FreeBSD 가 이 두 가지를 가지고 있는지 궁금할 것이다. 패키지와 포트는 서로 다른 장점을 가지고 있다. 이러한 이유로 여러분들은 단지 선호하는 것을 사용하거나 상황에 따라 선택하면 된다.

여러분들이 선택하기 쉽도록 패키지와 포트의 장점을 설명해준다.

## 4.2.1 패키지의 장점

- 
- 타볼(tar)로 압축되어 있는 패키지는, 어플리케이션 소스 코드를 타볼로 압축한 것보다 일반적으로 작다.
  - 소스 코드는 컴파일이 필요하지만 패키지는 컴파일이 필요없다. **Mozilla, KDE** 나 **GNOME** 처럼 사이즈가 큰 어플리케이션에서, 컴파일의 유무는 상당히 중요하고 시스템이 느리다면 특히 중요하다(자원 소모가 엄청나다).
  - 패키지를 사용한다면 앞에서 설명한 소프트웨어 설치 절차를 이해할 필요가 없다. 명령어 하나를 실행하면 시스템이 알아서 해준다.

## 4.2.2 포트의 장점

- 패키지는 광범위한 시스템에서 실행되기 때문에 일반적인 옵션으로 컴파일되어 있다. 포트에서 설치할 때 펜티엄 III 나 에슬론 프로세서에 최적화된 코드를 생성하도록 컴파일 옵션을 조정할 수 있다. 이로 인해 성능 향상과 버그 등 문제를 예방할 수 있다.
- 어떤 어플리케이션은 할 수 있는 것, 할 수 없는 것과 관련된 컴파일 시간 옵션을 가지고 있다. 예를 들어 **Apache** 웹 서버를 컴파일 할 때 광범위하고 다양한 컴파일 옵션을 지정할 수 있다. 포트에서 설치하면 기본 옵션을 사용하지 않고 원하는 옵션을 지정할 수 있다.

어떤 경우 어플리케이션에 특정 설정을 지정하기 위해 여러 가지 패키지가 있을 수 있다. 예를 들어 **Ghostscript** 는 X11 서버를 설치했는지에 따라 **ghostscript** 패키지와 **ghostscript-nox11** 패키지를 사용할 수 있다. 이런 종류의 간단한 튜닝은 패키지에서도 가능하지만, 어플리케이션이 한 개 이상의 다른 컴파일 시간 옵션을 가지고 있다면 불가능하다.

- 어떤 소프트웨어 배포 판의 라이선스 조항은 바이너리 배포를 금지한다. 이들은 소스 코드로만 배포된다.
- 어떤 사람들은 바이너리 배포 판을 믿지 않는다. 최소한 소스 코드를 읽어서 가능한 문제를 직접 찾는다.
- 패치를 가지고 있다면 패치를 적용하기 위해 소스가 필요하다.

- 
- 어떤 사람들은 코드를 좋아하기 때문에 심심하면 읽고 해킹해서 필요한 부분을 복사하는 등 여러 가지를 즐긴다.

포트가 업데이트되는 상황을 계속 지켜 보려면 FreeBSD 포트 메일링 리스트와 (<http://lists.FreeBSD.org/mailman/listinfo/freebsd-ports>) FreeBSD 포트 버그 메일링 리스트에 (<http://lists.FreeBSD.org/mailman/listinfo/freebsd-ports-bugs>) 가입한다. 메일링 리스트의 설명과 가입하는 방법은 [부록 C.1 메일링 리스트]를 참고한다.

**주의:** 어플리케이션을 설치하기 전에 어플리케이션의 보안에 관련된 문제를 <http://vuxml.freebsd.org/>에서(이곳에서는 FreeBSD와 FreeBSD 포트 컬렉션의 보안 문제를 소개한다) 확인한다.

또한 자동으로 모든 어플리케이션을 체크하여 알려진 **취약점**을 검사하고, 포트로 설치하기 전에도 체크해 주는 [security/portaudit](http://ports.freebsd.org/cgi/ports.cgi?qt=Q8T0ZqZ)를 설치해도 된다. 어떤 패키지를 설치한 후 **portaudit -F -a** 명령을 사용하면 취약점을 검사해준다.

이 장의 남은 부분에서는 패키지와 포트를 사용하여, 소프트웨어를 어떻게 설치하고 관리하는지 설명한다.

## 4.3 원하는 어플리케이션 찾기

어플리케이션을 설치하기 전에 어떤 기능을 원하고 어플리케이션 이름이 무엇인지 알고 있어야 된다.

FreeBSD 에서 사용할 수 있는 어플리케이션 리스트는 계속 증가하고 있다. 다행히 원하는 것을 찾을 수 있는 여러 가지 방법이 있다:

- FreeBSD 웹 사이트는 사용할 수 있는 모든 어플리케이션의 업데이트된 리스트를 <http://www.FreeBSD.org/ports/> 사이트에서 유지하고 있다. 포트는 카테고리 별로 나누어져 있어서, 이름을 알고 있다면 어플리케이션을 검색하거나 사용할 수 있는 모든 어플리케이션을 카테고리에서 볼 수 있다.

- 
- Dan Langille 는 <http://www.FreshPorts.org/>에 최신 포트를 유지하고있다. FreshPorts 는 어플리케이션의 변경사항을 포트 트리에서 확인하여 이들 포트가 업데이트 되면 메일로 통보해준다.
  - 원하는 어플리케이션의 이름을 모른다면 어플리케이션을 찾는데 FreshMeat(<http://www.freshmeat.net/>) 같은 사이트를 이용하고, 어플리케이션이 아직 포트되지 않았다면 FreeBSD 사이트로 다시 돌아와서 체크해본다.

## 4.4 패키지 시스템 사용

이번 섹션에서는 패키지를 사용하여 시스템에 프로그램을 설치하는 방법을 예제를 통해 보여준다. 그리고 패키지를 관리하고 삭제하는 기본적인 정보도 제공한다.

### 4.4.1 패키지 설치

FreeBSD 소프트웨어 패키지를 로컬 파일이나 네트워크의 서버에서 설치할 때 `pkg_add(1)` 유틸리티를 사용할 수 있다.

다음 예제를 따라 해보자

#### 예제 4-1. 패키지를 직접 다운로드해서 로컬에서 설치

```
# ftp -a ftp2.FreeBSD.org
Connected to ftp2.FreeBSD.org.
220 ftp2.FreeBSD.org FTP server (Version 6.00LS) ready.
331 Guest login ok, send your email address as password.
230-
230-   This machine is in Vienna, VA, USA, hosted by Verio.
230-   Questions? E-mail freebsd@vienna.verio.net.
230-
230-
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd /pub/FreeBSD/ports/packages/sysutils/
250 CWD command successful.
```

```
ftp> get lsof-4.56.4.tgz
local: lsof-4.56.4.tgz remote: lsof-4.56.4.tgz
200 PORT command successful.
150 Opening BINARY mode data connection for 'lsof-4.56.4.tgz' (92375 bytes).
100% |*****| 92375      00:00 ETA
226 Transfer complete.
92375 bytes received in 5.60 seconds (16.11 KB/s)
ftp> exit

# pkg_add lsof-4.56.4.tgz
```

로컬 패키지(FreeBSD CD-ROM 세트와 같은) 소스를 가지고 있지 않다면 pkg\_add(1)에 -r 옵션을 사용하는 것이 쉬울 것이다. 이 옵션을 사용하면 유틸리티가 정확한 파일을 결정하고 FTP 사이트에서 패키지를 가져와서 패치하고 설치한다.

```
# pkg_add -r lsof
```

위의 예제는 정확한 패키지를 다운로드하고 유저의 어떠한 개입도 없이 패키지를 설치한다. pkg\_add(1)는 fetch(3)를 사용하여 FTP\_PASSIVE\_MODE, FTP\_PROXY 와 FTP\_PASSWORD 를 포함하여 다양한 환경 변수가 설정된 파일을 다운로드한다.

방화벽 뒤쪽에 있다면 이들 중 한 개 또는 한 개 이상의 설정이 필요하거나, FTP/HTTP 프록시를 사용해야 된다. 옵션 리스트 전체는 fetch(3)를 본다. 위의 예제에서 lsof 는 lsof-4.56.4 대신 사용되었다. 원격 패치(다운로드) 기능을 사용할 때 패키지의 버전 번호는 삭제해야한다. pkg\_add(1) 명령은 가장 최신 버전의 어플리케이션을 자동으로 패치한다.

패키지 파일은 .tgz 형식으로 배포된다. 이들을 <ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/packages/> 사이트 또는 FreeBSD CD-ROM 배포 판에서 찾을 수 있다. FreeBSD 4-CD 세트의(그리고 PowerPak도) 모든 CD는, /packages 디렉터리에 패키지를 가지고있다. 패키지의 레이아웃(디렉터리 구조)은 /usr/ports 트리의 구조와 비슷하다. 각 카테고리는 각자 디렉터리를 가지고 있으며 모든 패키지는 All 디렉터리에서 찾을 수 있다.

패키지 시스템의 디렉터리 구조는 포트 레이아웃과 매칭되어 패키지/포트 시스템 전체를 형성하기 위해 같이 동작한다.

---

## 4.4.2 패키지 관리

pkg\_info(1)은 설치되어 있는 다양한 패키지 리스트와 간략한 설명을 보여 주는 유틸리티다.

```
# pkg_info
cvsup-16.1      A general network file distribution system optimized for CV
docbook-1.2    Meta-port for the different versions of the DocBook DTD
...
```

pkg\_version(1)은 설치되어 있는 모든 패키지의 버전을 요약해 주는 유틸리티다. 포트 트리에서 찾은 현재 버전과 패키지 버전을 비교해준다. 패키지 업그레이드 시기 결정에 도움이 된다.

```
# pkg_version
cvsup          =
docbook        =
...
```

두 번째 칼럼의 표식(=)은 설치된 버전과 로컬 포트 트리에서 사용할 수 있는 버전의 관계를 보여준다.

심볼	의미
=	설치되어 있는 패키지 버전이 로컬 포트 트리에서 찾은 것과 같다.
<	설치되어 있는 버전이 포트 트리에서 사용할 수 있는 것보다 예전 것이다.
>	설치되어 있는 버전이 로컬 포트 트리에서 찾아낸 것보다 새로운 것이다. (로컬 포트 트리 업데이트가 필요할 것이다.)
?	설치되어 있는 패키지를 포트 인덱스에서 찾을 수 없다(예를 들면 설치된 포트가 포트 컬렉션에서 삭제되었거나 이름이 바뀌었을 때 이런 일이 발생한다).
*	여러 가지 버전의 패키지가 있다.

## 4.4.3 패키지 삭제

이전에 설치한 패키지를 삭제할 때 pkg\_delete(1) 유틸리티를 사용한다.

```
# pkg_delete xchat-1.7.1
```

---

## 4.4.4 더 자세한 패키지 정보

모든 패키지 정보는 `/var/db/pkg` 디렉터리에 저장되어 있다. 설치되어 있는 파일 리스트와 각 패키지의 설명은 이 디렉터리에 있는 파일에서 찾을 수 있다.

## 4.5 포트 컬렉션 사용

이번 섹션에서는 포트 컬렉션을 사용하여 시스템에 프로그램을 설치하거나 삭제하는 기본적인 정보를 제공한다.

### 4.5.1 포트 컬렉션 받아오기

포트를 설치하기 전에 기본적으로 Makefiles 세트, 패치 그리고 `/usr/ports`의 설명 파일을 가지고 있는 포트 컬렉션을 먼저 받아 와야 된다.

FreeBSD 시스템을 설치하는 중간에 **sysinstall**이 포트 컬렉션을 설치할 것인지 물어 보았다. **no**를 선택했다면 다음 지시를 따라서 포트 컬렉션을 받아올 수 있다.

■ [예제: Sysinstall 를 사용하는 방법]

이 방법은 **sysinstall**을 다시 사용하여 포트 컬렉션을 직접 설치한다.

① root에서 다음과 같이 `/stand/sysinstall`을 실행한다:

```
# /stand/sysinstall
```



```

FreeBSD Configuration Menu
If you've already installed FreeBSD, you may use this menu to customize
it somewhat to suit your particular configuration. Most importantly,
you can use the Packages utility to load extra "3rd party"
software not provided in the base distributions.

X Exit          Exit this menu (returning to previous)
Distributions   Install additional distribution sets
Packages        Install pre-packaged software for FreeBSD
Root Password  Set the system manager's password
Fdisk          The disk Slice (PC-style partition) Editor
Label          The disk Label editor
User Management Add user and group information
Console        Customize system console behavior
Time Zone      Set which time zone you're in
Media          Change the installation media type
Mouse          Configure your mouse
Networking     Configure additional network services
Security       Select default system security profile
Startup        Configure system startup options
TTYs           Configure system ttys.
Options        View/Set various installation options
XFree86        Configure XFree86 Server
Desktop        Configure XFree86 Desktop
HTML Docs      Go to the HTML documentation menu (post-install)
Load KLD       Load a KLD from a floppy

[ OK ] [ Cancel ]
[ Press F1 for more information on these options ]

```

② 방향키로 아래로 이동하여 Configure 를 선택하고 Enter 를 누른다.

```

Usage          Quick start - How to use this menu system
Standard      Begin a standard installation (recommended)
Express        Begin a quick installation (for the impatient)
Custom        Begin a custom installation (for experts)
Configure      Do post-install configuration of FreeBSD
Doc           Installation instructions, README, etc.
Keymap        Select keyboard type

```

③ 방향키로 아래로 이동하여 Distributions 를 선택하고 Enter 를 누른다.

```

X Exit          Exit this menu (returning to previous)
Distributions   Install additional distribution sets
Packages        Install pre-packaged software for FreeBSD
Root Password  Set the system manager's password

```

④ 방향키로 ports 로 이동하여 Enter 를 누른다.

```

[ ] src         Sources for everything
[ ] ports       The FreeBSD Ports collection
[ ] local       Local additions collection
[ ] XFree86     The XFree86 distribution

```

⑤ 위로 이동하여 Exit 를 선택하고 Enter 를 누른다.

```

<< X Exit      Exit this menu (returning to previous)
  All          All system sources, binaries and X Window System
  Reset        Reset all of the below
[ ] bin        Binary base distribution (required)

```

⑥ CDROM, FTP 처럼 원하는 설치 미디어를 선택한다.

```

1 CD/DVD      Install from a FreeBSD CD/DVD
2 FTP        Install from an FTP server
3 FTP Passive Install from an FTP server through a firewall
4 HTTP       Install from an FTP server through a http proxy

```

⑦ Exit 로 이동해서 Enter 를 누른다.

```

X Exit      Exit this menu (returning to previous)
Distributions Install additional distribution sets
Packages   Install pre-packaged software for FreeBSD
Root Password Set the system manager's password

```

⑧ X 를 누르고 sysinstall 를 빠져 나간다.

```

Select [X] Exit Install
[ Press F1 for Installation Guide ]

```

업데이트된 포트 컬렉션을 가져오고 유지하는 다른 방법은 CVSup 을 사용한다.  
 /usr/share/examples/cvsup/ 디렉터리에 있는 파일을 확인한다. CVSup 사용법과 이 파일에  
 대한 자세한 정보는 [CVSup 사용] (부록 A.5)을 본다.

■ [CVSup 을 사용하는 방법]

이 방법은 CVSup 을 사용하여 포트 컬렉션을 빨리 받는 방법이다. 포트 트리를 최신으로  
 업데이트하거나 CVSup 에 대해 좀더 알고 싶다면 이전에 언급한 섹션을 읽는다.

① net/cvsup 포트를 설치한다. 더 자세한 내용은 CVSup 설치(부록 A.5.2)를 본다.

```

# cd /usr/ports/net/cvsup      [설치할 포트 디렉터리로 이동한다]
# make install                [포트 설치 명령]

```

② root 에서 /usr/share/exaples/cvsup/ports-supfile 를 /root 나 홈 디렉터리 같은 새로  
 운 위치로 복사한다.

```

# cp /usr/share/examples/cvsup/ports-supfile . [현재 디렉터리로 복사]

```

③ ports-supfile 를 수정한다.

```

# vi ports-supfile

```

---

④ `CHAGE_THIS.FreeBSD.org` 를 가장 가까운 **CVSup** 서버(`cvsup.kr.FreeBSD.org`)로 변경한다. 전체 미러 사이트 리스트는 [**CVSup** 미러 사이트]를(부록 A.5.7) 본다.

⑤ `cvsup` 을 실행한다:

```
# cvsup -g -L 2 /root/ports-supfile
```

⑥ 이 명령을 실행하면 가장 최근에 변경된 모든 것을 다운로드하고, 실제로 시스템에 포트를 다시 빌드하는 것을 제외하고 포트 컬렉션에 적용한다.

## 4.5.2 포트 설치

포트 컬렉션은 실제로 “스켈레톤(골격)”이라고 설명할 수 있다. 아주 간단히 말해서 포트 스킴레톤은, FreeBSD 시스템에 프로그램을 어떻게 컴파일하고 설치하는지 지정되어 있는 최소한의 파일 세트다. 포트 스킴레톤은 다음과 같은 파일과 디렉터리를 가지고 있다:

- [Makefile 파일] 이 파일은 어플리케이션을 어떻게 컴파일하고 시스템의 어느 곳에 설치해야 되는지 지정하는, 다양한 상태 정보를 가지고 있다.
- [distinfo 파일] 이 파일은 다운로드 해서 포트로 설치해야 되는 파일과, md5(1)를 사용하여 파일이 다운로드되는 동안 오류가 없는지 확인하는 체크섬 정보를 가지고 있다.
- [file 디렉터리] 이 디렉터리는 FreeBSD 시스템에 프로그램을 컴파일하고 설치하는 패치를 가지고 있다. 패치는 기본적으로 특정 파일을 변경하도록 지정하는 작은 파일이다. 이들은 평범한 텍스트 형식으로 되어 있고, 기본적으로 "라인 10 삭제" 또는 "라인 26 을 이렇게 바꾸어라" 등으로 설명 되어 있다. 패치는 diff(1) 프로그램으로 생성되기 때문에 "diffs"라고도 한다.

이 디렉터리는 포트 빌드에 사용되는 다른 파일도 가지고 있을 것이다.

- [pkg-descr 파일] 이 파일은 보통 여러 라인으로 더 자세히 프로그램을 설명한다.
- [pkg-plist 파일] 이 파일은 포트가 설치하는 모든 파일 리스트다. 그리고 삭제 명령으로 어떤 파일이 삭제되었는지 알려주는 포트 시스템이다.

---

어떤 포트는 pkg-message 같은 다른 파일을 가지고 있다. 포트 시스템은 특수한 상황을 제어할 때 이런 파일을 사용한다. 이들 파일과 일반적으로 포트에 대해 더 알고 싶다면 FreeBSD 포터의 핸드북을 읽어 보기 바란다  
([http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/porters-handbook/index.html](http://www.freebsd.org/doc/en_US.ISO8859-1/books/porters-handbook/index.html)).

이제 포트 컬렉션이 무엇에 사용되는지 알고 있으므로 첫 번째 포트를 설치해 보자. 포트 설치하는 두 가지 방법이 있고 아래에서 각각 설명한다.

그러나 시작하기 전에 설치하려는 포트를 선택해야 된다. FreeBSD 웹 사이트에 (<http://www.freebsd.org/ports/index.html>) 리스트 되어 있는 방법을 포함하여, 포트를 선택하는 몇 가지 방법이 있다. 이 사이트에 리스트 되어 있는 포트를 훑어 보거나 사이트의 검색 기능을 사용한다. 그리고 각 포트는 설명을 포함하고 있기 때문에 설치를 결정하기 전에 각 포트에 대해 약간씩 읽어 볼 수 있다.

다른 방법은 whereis(1) 명령을 사용하는 것이다. 단순히 whereis *file* 이라고 입력하고, 여기서 *file* 은 설치하려는 프로그램이다. 원하는 프로그램을 시스템에서 찾았다면 다음과 같이 어디에 있는지 보여준다:

```
# whereis lsof
```

```
lsof: /usr/ports/sysutils/lsof
```

이 명령은 lsof 를 /usr/ports/sysutils/lsof 디렉터리에서 찾을 수 있다고 말해준다.

특정 포트를 찾는 다른 방법은 포트 컬렉션에 내장되어 있는 검색 메커니즘을 사용하는 것이다. 검색 기능을 사용하려면 /usr/ports 디렉터리로 이동한다. 이 디렉터리에서 make search name=*program-name* 을 실행하고, 여기서 *program-name* 은 찾으려는 프로그램의 이름이다. 이제 lsof 를 찾아보자:

```
# cd /usr/ports
# make search name=lsof
Port: lsof-4.56.4
Path: /usr/ports/sysutils/lsof
Info: Lists information about open files (similar to fstat(1))
Maint: obrien@FreeBSD.org
Index: sysutils
B-deps:
```

---

R-deps:

출력 부분에서 특히 관심을 가질 곳은, 어디서 포트를 찾을 수 있는지 알려 주는 "Path:"라  
인이다. 다른 정보는 설치할 때 필요 없기 때문에 여기서 설명하지 않는다.

좀더 철저한 검색은 `make search key=string`를 사용할 수 있고, *string*은 검색하려는 문자  
열이다. 이 명령은 포트 이름, 주석, 설명과 의존성을 찾아준다. 또한 찾으려는 프로그램의  
이름을 모를 경우 특정 주제와 관련된 포트를 찾을 때 사용할 수 있다.

이 두 가지 경우 검색 문자열은 대소 문자를 구분하지 않는다. "LSOF" 검색은 "lsof" 검색과  
같은 결과를 보여준다.

**Note:** 포트 설치 는 root 에서 해야된다.

이제 설치하려는 포트를 찾아냈고 실제로 설치 준비가됐다. 포트는 소스 코드를 빌드하는  
설정 정보를 가지고 있지만 실제 소스 코드는 포함하지 않는다. 소스 코드는 CD-ROM 이나  
인터넷에서 가져올 수 있다. 소스 코드는 소프트웨어 개발자가 원하는 형식으로 배포된다.  
대부분의 소스 코드는 tar 와 gzip 파일 형식이지만, 다른 틀이나 압축하지 않았을 수도 있  
다. 프로그램 소스 코드가 어떤 형식이던지 "distfile" 라고 부른다. Distfile 은 CD-ROM 이나  
인터넷에서 받을 수 있다.

**주의:** 포트를 설치하기 전에 포트 컬렉션을 업데이트하고 포트와 관련된 보안 문제  
를 <http://vuxml.freebsd.org> 사이트에서 체크해야 된다.

보안 취약점은 어플리케이션을 새로 설치하기 전에 portaudit 로 자동으로 체크할 수  
있다. 이 틀은 포트 컬렉션(security/portaudit)에서 찾을 수 있다. 새로운 포트를 설  
치하기 전에 portaudit -F 를 실행하면 최신 취약성 데이터베이스로 패치한다. 보안  
공고와 데이터베이스 업데이트는 일일 보안 시스템 체크로 수행된다. 더 많은 정보  
는 portaudit(1)과 periodic(8) 매뉴얼 페이지를 참고한다.

#### 4.5.2.1 CD-ROM 에서 포트 설치

FreeBSD 프로젝트의 공식 CD-ROM 이미지에는 더 이상 distfile을 포함하지 않는다. 이  
CD-ROM들은 미리 컴파일되어 있어서 패키지에 적합한 많은 디렉토리를 가지고 있다.  
FreeBSD PowerPak같은 CD-ROM 제품은 distfile을 포함하고 있으며 이들 제품은 FreeBSD

---

물 (<http://www.freebsdmail.com/>) 같은 벤더에서 구입할 수 있다. 이번 섹션은 FreeBSD CD-ROM 세트를 가지고 있다고 가정한다.

FreeBSD CD-ROM 을 드라이브에 넣고 /cdrom 에(다른 마운트 포인트를 사용한다면 설치가 되지 않을 것이다) 마운트한다. 설치하려는 포트 디렉터리로 변경해서 설치를 시작한다:

```
# mount /cdrom
# cd /usr/ports/sysutils/Isf
```

Isf 디렉터리 안에서 포트 스케leton을 볼 수 있다. 다음 단계는 포트를 "빌드(컴파일)"한다. 이 단계는 프롬프트에서 단순히 make 를 입력한다. 명령을 입력하면 다음과 비슷한 메시지를 보게 된다:

```
# make
>> Isf_4.57D.freebsd.tar.gz doesn't seem to exist in /usr/ports/distfiles/.
>> Attempting to fetch from file:/cdrom/ports/distfiles/.
===> Extracting for Isf-4.57
...
[extraction output snipped]
...
>> Checksum OK for Isf_4.57D.freebsd.tar.gz.
===> Patching for Isf-4.57
===> Applying FreeBSD patches for Isf-4.57
===> Configuring for Isf-4.57
...
[configure output snipped]
...
===> Building for Isf-4.57
...
[compilation output snipped]
...
#
```

컴파일이 끝나면 프롬프트로 다시 되돌아온다. 다음 단계는 포트를 설치하는 단계로 install 이라는 간단한 단어를 make 명령에 붙인다:

```
# make install
===> Installing for Isf-4.57
```

```
...
[installation output snipped]
...
====>  Generating temporary packing list
====>  Compressing manual pages for Isof-4.57
====>  Registering installation for Isof-4.57
====>  SECURITY NOTE:
        This port has installed the following binaries which execute with
        increased privileges.
#
```

프롬프트가 나타나면 방금 설치한 어플리케이션을 실행할 수 있다. Isof 는 권한을 증가시키기 때문에 보안 경고가 뜬다. 포트를 빌드하고 설치하는 동안 발생할만한 다른 경고에 주의를 기울인다.

**Note:** 어떤 셸은 PATH 환경 변수에 나열되어 있는 디렉터리에서 실행할 명령을 케시에 담고 있다. 이런 셸을 사용하고 있다면 새로 설치한 명령을 사용하기 전에 포트를 설치한 후 rehash 명령을 실행해야 된다. 이런 셸들은 기본 시스템에 포함되어 있어서(tcsh 같은) 포트에서(예를 들면 shells/zsh) 사용할 수 있다.

#### 4.5.2.2 인터넷에서 포트 설치

마지막 섹션에서처럼 이번 섹션도 인터넷에 연결된 상태에서 작업을 한다고 가정한다. 연결되어 있지 않다면 CD-ROM 을 설치하거나 distfile 의 복사본을 /usr/ports/distfiles 에 직접 넣어야된다.

**Note:** 어떤 포트 라이선스는 CD-ROM 에 포함하는 것을 허용하지 않는다. 이런 포트는 다운로드하기 전에 등록 품을 채워야 되거나, 재 배포를 허용하지 않거나 다른 이 유일 수 있다. CD-ROM 이 아닌 포트에서 설치하려면 온라인에 연결되어 있어야 한다.

인터넷에서 포트를 설치하는 것은 CD-ROM 에서 설치하는 것과 같다. 둘 사이의 유일한 차이는 distfile 을 CD-ROM 에서 읽지 않고 인터넷에서 다운로드하는 것이다.

다음과 같이 설치 과정도 동일하다:

```
# make install
>> Isof_4.57D.freebsd.tar.gz doesn't seem to exist in /usr/ports/distfiles/.
>> Attempting to fetch from ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/distfiles/.
Receiving Isof_4.57D.freebsd.tar.gz (439860 bytes): 100%
439860 bytes transferred in 18.0 seconds (23.90 kBps)
===> Extracting for Isof-4.57
...
[extraction output snipped]
...
>> Checksum OK for Isof_4.57D.freebsd.tar.gz.
===> Patching for Isof-4.57
===> Applying FreeBSD patches for Isof-4.57
===> Configuring for Isof-4.57
...
[configure output snipped]
...
===> Building for Isof-4.57
...
[compilation output snipped]
...
===> Installing for Isof-4.57
...
[installation output snipped]
...
===> Generating temporary packing list
===> Compressing manual pages for Isof-4.57
===> Registering installation for Isof-4.57
===> SECURITY NOTE:
    This port has installed the following binaries which execute with
    increased privileges.
#
```

이전 예제에서 /usr/ports/distfiles/ 디렉터리에서 포트 distfile을 찾지 못했기 때문에 <ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/distfiles/> 사이트에서 받아온다.

패키지에서 설명했지만 포트 시스템도 FTP\_PASSIVE\_MODE, FTP\_PROXY 와



---

FTP\_PASSWORD 를 포함한 다양한 환경 변수가 지정된 파일을 다운로드하기 위해 `fetch(1)` 를 사용한다. 방화벽 안쪽에 있다면 한가지 또는 한가지 이상의 설정이 필요하거나, FTP/HTTP 프록시를 사용해야 된다. 전체 옵션 리스트는 `fetch(3)` 매뉴얼 페이지를 본다.

항상 인터넷에 접속할 수 없는 유저를 위해 `make fetch` 옵션이 제공된다. 디렉터리의 가장 상위 레벨에서(`/usr/ports`) 이 명령만 실행하면 필요한 파일이 다운로드된다. 이 명령은 `/usr/ports/net` 처럼 레벨이 낮은 카테고리에서도 실행된다. 포트가 라이브러리나 다른 포트에 의존된다면 포트의 의존성 포트나 라이브러리의 `distfile` 은 패치하지 않는다. 포트 의존성도 모두 패치하려면 `fetch` 를 `fetch-recursive` 로 변경한다.

**Note:** 앞서 말한 `make fetch` 처럼 최상위 디렉터리 `make` 를 실행하여 전체를 빌드할 수 있다. 그러나 어떤 포트는 동시에 빌드할 수 없기 때문에 위험한 방법이다. 다른 경우 어떤 포트는 같은 파일 이름으로 두 개의 다른 파일을 설치할 수 있다.

가끔 `MASTER_SITES` 가 아닌 다른 사이트에서(파일을 다운로드 하는 위치) 타볼(`tar` 로 압축된 소스코드)을 받아야 된다. 다음 명령으로 `MASTER_SITE` 옵션을 무시할 수 있다:

```
# cd /usr/ports/directory

# make MASTER_SITE_OVERRIDE= W
ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/distfiles/fetch
```

이 예제는 `MASTER_SITE` 옵션을 `ftp.FreeBSD.org/pub/FreeBSD/ports/distfiles/`로 변경한다.

**Note:** 어떤 포트는 어플리케이션에서 필요 없는 부분을 비활성하거나, 특정 보안 옵션과 튜닝을 활성화 또는 비활성할 수 있는 빌드 옵션을 제공한다. 영두 해 돌만한 포트 몇 개는 [www/mozilla](#), [security/gpgme](#) 와 [mail/sylpheed-claws](#) 가 있다. 이런 옵션을 이용할 수 있을 때 메시지가 나타난다.

### 4.5.2.3 기본 포트 디렉터리 무시하기

가끔 다른 `distfiles` 과 포트 디렉터리를 사용하는 것도 유용하다(또는 필수적이다). 다음과 같이 `PORTSDIR` 과 `PREFIX` 변수로 기본 디렉터리를 무시할 수 있다.

---

```
# make PORTSDIR=/usr/home/example/ports install
```

위 명령은 /usr/home/example/ports 에서 포트를 컴파일하고 /usr/local 아래에 설치한다.

```
# make PREFIX=/usr/home/example/local install
```

위 명령은 /usr/ports 에서 컴파일하고 /usr/home/example/local 에 설치한다.

```
# make PORTSDIRW=../ports PREFIX=../local install
```

위 명령은 두 개를 결합한다(이 페이지에서 전체 내용을 설명하기에 너무 길기 때문에 일반적인 사용법만 설명한다).

이런 변수를 환경 변수에 설정할 수 있다. 셸에서 환경 변수를 설정하는 방법은 매뉴얼 페이지를 참고한다.

#### 4.5.2.4 imake 문제

imake 를(X 윈도우 시스템의 일부) 사용하는 어떤 포트는, PREFIX 와 정상적으로 동작하지 않고 /usr/X11R6 에서 설치해야 된다는 메시지를 보여준다. 그리고 어떤 Perl 포트도 PREFIX 를 무시하고 Perl 트리에서 설치해야 된다. 이런 포트에는 PREFIX 를 사용하기가 어렵거나 불가능하다고 생각된다.

#### 4.5.3 설치되어 있는 포트 삭제하기

포트를 설치하는 방법을 알았기 때문에, 포트를 잘못 설치했다고 생각될 경우 어떻게 삭제하는지 궁금할 것이다. 이전 예제에서 설치한 포트를 삭제하는 예제를 보자 (별로 관심이 없는 Isof 같은 포트). 포트를 설치할 때처럼 포트 디렉터리(/usr/ports/sysutils/Isof )로 이동한다. 디렉터리를 변경 했으면 Isof 를 삭제할 준비가 되었다. make deinstall 명령을 이용하면 삭제할 수 있다:

```
# cd /usr/ports/sysutils/Isof           [삭제하려는 포트 디렉터리로 이동]
# make deinstall                       [삭제 명령 실행]
===> Deinstalling for Isof-4.57
```

---

이제 lsof 가 삭제되었다. 삭제한 포트를 다시 설치하려면 /usr/ports/sysutils/lsof 디렉터리에서 make reinstall 을 사용하여 다시 설치할 수 있다.

```
# make reinstall
```

make clean 를 실행하면 make deinstall 과 make reinstall 은 동작하지 않는다. 포트를 정리(clear)한 후 삭제하려면, 핸드북의 패키지 섹션에서 설명한 pkg\_delete(1)를 사용한다.

#### 4.5.4 포트와 디스크 공간

포트 컬렉션을 이용하면 많은 디스크 공간을 사용할 수 있다. 이런 이유로 make clean 옵션을 사용하여 작업한 디렉터를 항상 정리해야 된다. 이 명령은 포트를 빌드하고 설치한 후 작업한 디렉터를 삭제한다. 그리고 distfiles 디렉터리의 tar 파일을 삭제할 수 있고 포트 사용이 끝나면 설치된 포트를 삭제할 수 있다. 어떤 유저는 refuse 파일에 관련된 옵션을 입력하여 특정 포트 카테고리의 선택을 제한하기도 한다. 이 경우 CVSup 을 실행하면 그 카테고리의 파일은 다운로드하지 않는다.

#### 4.5.5 포트 업그레이드

항상 업데이트된 상태로 포트를 유지하는 것은(그래야 최신 버전의 어플리케이션을 설치할 수 있다) 상당히 피곤한 작업일 것이다. 예를 들어 포트를 업그레이드하려면 포트 디렉터리로 이동하여 포트를 빌드한 후 예전 포트를 삭제한다. 그리고 새로운 포트를 설치한 후 정리한다. 다섯 개의 포트만 생각해 보더라도 상당히 지겨울 것이다. 시스템 관리자들에게 이런 작업은 큰 문제였기 때문에 자동으로 포트를 업데이트하는 유틸리티를 제공한다. 예를 들면 [sysutils/portupgrade](#) 유틸리티는 자동으로 포트를 업데이트 해준다. 다른 포트처럼 make install clean 명령을 사용하여 이 유틸리티를 설치한다.

```
# cd /usr/ports/sysutils/portupgrade
# make install
# /usr/local/sbin/pkgdb -F ❶
---> Checking the package registry database
[Rebuilding the pkgdb <format:bdb1_btree> in /var/db/pkg ... - 13 packages found (-0
+13) .....done]

# /usr/local/sbin/portupgrade -a ❷
```

```
[Updating the portsdb <format:bdb1_btree> in /usr/ports ... - 10796 port entries
found .....1000.....2000.....3000.....4000.....5000.....6000.....7000....
.....8000.....9000.....10000..... done]
```

- ❶ 설치가 끝나면 `pkgdb -F` 명령으로 데이터베이스를 생성한다. 이 명령은 설치되어 있는 포트 리스트를 읽어서 `/var/.db/pkg` 디렉터리에 데이터베이스 파일을 생성한다.
- ❷ 그리고 `portupgrade -a` 를 실행하면 포트 데이터베이스와 포트 INDEX 파일을 읽는다. 마지막으로 **portupgrade** 는 다운로드를 시작하고 빌드한다. 그리고 기존 설정을 백업한 후 설치하고 업데이트된 포트를 정리한다.

**portupgrade** 는 여러 가지 경우에 사용할 수 있는 다양한 옵션을 가지고 있지만, 가장 중요한 옵션을 아래에서 설명한다.

전체 데이터베이스가 아닌 특정 어플리케이션만 업그레이드한다면 `portupgrade pkgname` 를 사용한다. 여기서 `pkgname` 은 원하는 어플리케이션 이름이다. **portupgrade** 가 입력한 패키지의 의존성과도 동작해야 된다면 `-r` 플래그를 사용하고, `-R` 은 입력한 패키지에 필요한 모든 패키지와 동작한다. 포트 대신 패키지를 사용하여 설치하려면 `-P` 옵션을 주고, 빌드와 설치는 하지 않고 `distfile` 만 패치하려면 `-F` 를 사용한다. 더 자세한 설명은 `portupgrade(1)` 매뉴얼 페이지를 확인한다.

**Note:** 데이터베이스가 일치하지 않는 문제를 수정하기 위해, `pkgdb -F` 를 사용하여 패키지 데이터베이스를 규칙적으로 업데이트하는 것이 중요하다(특히 **portupgrade** 가 필요할 때). 데이터베이스 무결성이 깨지기 때문에 패키지 데이터베이스를 업데이트하는 동안 **portupgrade** 를 중단하지 않는다.

`portupgrade` 같은 다른 유틸리티도 있으므로 `ports/sysutils` 디렉터리를 체크하고 어떤 포트가 가능한지 확인해본다.

## 4.6 설치 후 과정

새로운 어플리케이션을 설치한 후 일반적으로 어플리케이션에 포함되어있는 문서를 읽어본다. 그리고 필요한 설정을 편집하여 부팅할 때 (데몬 이라면) 그 어플리케이션이 자동으로 시작되기를 원할 것이다.

---

각 어플리케이션을 설정하는 정확한 단계는 분명히 다르다. 그러나 방금 어플리케이션을 새로 설치했다면 이제 무엇을 해야 되는가?라는 의문에 대해, 다음 설명이 도움이 될 것이다.:

- 어떤 파일이 어디에 설치되었는지 찾을 때 `pkg_info(1)`를 사용한다. 예를 들어 `FooPackage` 버전 1.0.0 을 바로 설치했다면 다음 명령으로 패키지가 설치한 모든 파일을 볼 수 있다.

```
# pkg_info -L foopackage-1.0.0 | less
```

매뉴얼 페이지는 `man/` 디렉터리를 설정파일은 `/etc` 디렉터리, 그리고 더욱 포괄적인 문서가 있는 `doc/` 디렉터리에 특별한 관심을 가진다.

방금 설치한 어플리케이션의 버전이 확실하지 않다면, 다음 명령으로 `foopackage` 라는 패키지가 설치한 모든 패키지를 볼 수 있다.

```
# pkg_info | grep -i foopackage
```

명령어 라인에서 `foopackage` 를 방금 설치한 어플리케이션으로 바꾼다.

- 어플리케이션의 매뉴얼 페이지가 설치된 곳을 확인하여 `man(1)`으로 다시 읽어본다. 그리고 샘플 설정 파일과 같이 제공된 추가적인 문서도 확인한다.
- 어플리케이션이 웹 사이트를 가지고 있다면 추가적인 문서와 빈번한 질문 등을 확인해본다. 웹 사이트 주소가 확실하지 않다면 다음 명령의 결과에 표시될 것이다.

```
# pkg_info foopackage-1.0.0
```

`WWW:` 라인이 어플리케이션의 웹사이트 URL 을 보여준다.

- 부팅할 때 시작해야 되는 포트는 보통 `/usr/local/etc/rc.d` 에 샘플 스크립트를 설치한다. 이 스크립트가 정확한지 확인하고 필요하다면 수정하거나 이름을 바꾼다. 더 자세한 내용은 시작 서비스를 확인한다.

---

## 4.7 깨진 포트문제 해결

포트가 정상적으로 동작하지 않는다면 체크할 만한 몇 가지 사항이 있다:

1. 수정하자! Porters 의 Handbook 은([http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/porters-handbook/index.html](http://www.freebsd.org/doc/en_US.ISO8859-1/books/porters-handbook/index.html)) "Ports" 구조에 대한 자세한 정보를 포함하기 때문에, 때때로 깨진 포트를 수정하거나 알맞게 고칠 수 있다.
2. 포트 관리자에게 메일을 보낸다. `make maintainer` 명령을 실행하거나 관리자의 메일 주소를 찾기 위해 Makefiles 을 읽어본다. 포트 이름 및 버전(Makefile 에서 `$FreeBSD:` 라인을 보낸다)과 관리자가 메일을 받았을 때 에러를 확인할 수 있도록 결과도 포함시켜야 된다. 관리자로부터 답변이 오지 않는다면 `send-pr(1)`을 사용하여 버그 리포트를 제출할 수 있다.
3. 가까운 FTP 사이트에서 패키지를 받는다. "마스터" 패키지 컬렉션은 [ftp.FreeBSD.org](ftp://ftp.FreeBSD.org) 의 패키지 디렉터리에 있지만 (<ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/packages/>), 가까운 미러 사이트를 먼저 체크해 보자! 이들 미러 사이트는 원활히 동작하고 더 빠르다. 그리고 다운받은 패키지를 설치하려면 이전에 설명한 `pkg_add(1)` 프로그램을 사용한다.

---

## 5 장 X 윈도우 시스템

### 5.1 개요

FreeBSD 는 강력한 그래픽 유저 인터페이스를 제공하기 위해 **XFree86** 을 사용한다. **XFree86** 은 X 윈도우 시스템의 오픈 소스 버전이다. 이번 장에서는 FreeBSD 시스템에 **XFree86** 의 설치와 설정을 다룬다. **XFree86** 과 비디오 하드웨어 지원에 대한 더 많은 정보는 **XFree86** 웹사이트를(<http://XFree86.org/>) 체크한다.

이번 장을 읽고 다음 사항에 대해 알 수 있다.

- X 윈도우 시스템의 다양한 컴포넌트와 이들의 기능을 알 수 있다.
- **XFree86** 의 설치와 설정
- 다른 윈도우 매니저는 어떻게 설치하고 사용하는가
- **XFree86** 에서 투루 타입 폰트를 어떻게 사용하는가
- 그래픽컬 로그인(**XDM**)은 시스템에 어떻게 설정하는가

이번 장을 읽기 전에 다음 사항을 알고 있어야 된다.

- 소프트웨어를 어떻게 설치하는지 알아야 된다(4 장).

### 5.2 X 이해

처음 X 를 사용하면 마이크로소프트 윈도우즈나 MacOS 와 같은 다른 그래픽컬 환경에 친숙한 사람들에게는 약간 당황스러울 수도 있다.

다양한 X 컴포넌트와 X 컴포넌트가 어떤 기능을 하는지 자세히 이해할 필요는 없다. 그러

---

나 기본적인 지식으로 X의 장점을 최대한 활용할 수가 있다.

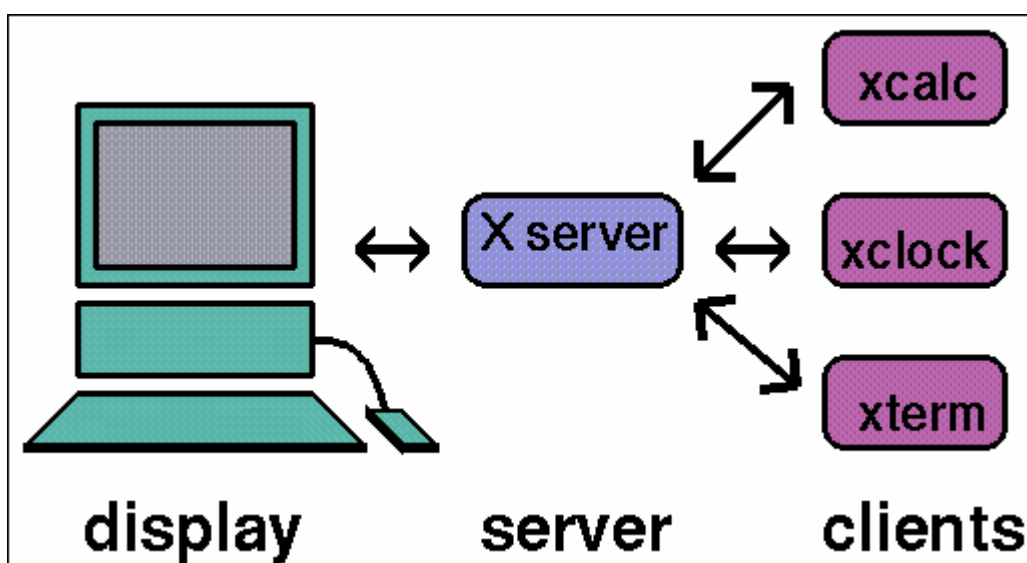
## 5.2.1 왜 X인가?

X는 유닉스용으로 작성된 최초의 윈도우 시스템은 아니지만 아주 유명하다. X의 최초 개발팀은 X를 작성하기 전에 다른 윈도우 시스템에서 작업을 했다. 그 시스템 이름은 "W"("Window")였다. 로마 알파벳에서 X는 단지 다음 문자다.

X는 "X", "X 윈도우 시스템", "X11" 그리고 다른 용어로 부를 수 있다. "X 윈도우"를 X11 이라고 부르는 것은 다른 사람들의 기분을 상하게 할 수 있다; 이와 관련된 더 자세한 내용은 X(7)를 참고한다.

## 5.2.2 X 클라이언트/서버 모델

X는 네트워크가 중심이되기 시작하던 시기에 디자인되어 "클라이언트-서버" 모델을 채택하였다. X 모델에서 "X 서버"는 키보드, 모니터 그리고 마우스가 붙어있는 컴퓨터에서 실행된다. 서버는 화면 관리, 키보드나 마우스 입력을 제어하는 등의 기능을 가지고 있다. 각 X 어플리케이션(XTerm 또는 Netscape 같은) "클라이언트"다. 클라이언트는 "이러한 좌표의 윈도우를 그려달라" 같은 메시지를 서버에게 보내고 서버는 "유저가 바로 OK 버튼을 클릭했다" 같은 메시지를 되돌려 보낸다.





---

가정이나 작은 사무 환경 같은 곳에 컴퓨터가 한대만 있다면 X 서버와 X 클라이언트는 같은 컴퓨터에서 실행된다. X 서버를 좀더 성능이 떨어지는 데스크톱 컴퓨터에서 그리고 X 어플리케이션(클라이언트) 사무실에서 제공하는 강력하고 비싼 머신에서 실행하는 것도 물론 가능하다. 이 시나리오에서 X 클라이언트와 서버 사이의 통신은 네트워크를 이용한다.

X 관련 용어는 기대보다 뒤떨어지기 때문에 어떤 사람들에게는 혼란스러울 수 있다. 그들은 "X 서버"는 전산실의 크고 강력한 머신에 그리고 "X 클라이언트" 머신은 그들의 책상 위에 있어야 된다고 생각한다.

X 서버는 키보드와 모니터가 있는 머신이고 X 클라이언트는 윈도우를 표시하는 프로그램이라는 것을 기억한다.

클라이언트와 서버가 같은 운영체제 또는 같은 타입의 컴퓨터라는 규정은 없다. X 서버를 마이크로소프트 윈도우나 애플의 MacOS 에서 실행하는 것도 가능하고 똑 같은 기능을 하는 다양한 무료 소프트웨어와 상업적인 어플리케이션도 사용할 수 있다.

FreeBSD 에 적용된 X 서버는 **XFree86** 이라고 하며 FreeBSD 라이선스와 매우 유사한 라이선스로 자유로이 사용할 수 있다. FreeBSD 에 상업용 X 서버도 사용할 수 있다.

### 5.2.3 윈도우 매니저

X 의 디자인 철학은 "수단이 아닌 도구다" 라는 유닉스 디자인 철학과 비슷하다. 이 의미에서 X 는 작업이 어떻게 수행될 수 있는지 알려주지 않는다. 대신 툴을 유저에게 제공하고 이런 툴을 어떻게 사용할지 결정하는 것은 유저의 책임이다.

이 철학은 화면의 윈도우가 어떻게 생기고 마우스로 어떻게 윈도우를 움직이며 윈도우 사이에서 어떤 키로 이동하는지(예: 마이크로소프트 윈도우에서는 **Alt+Tab**), 각 윈도우의 타이틀 바가 어떻게 생겼는지 그리고 윈도우가 닫기 버튼을 가지고 있는지 아닌지 등을 X 에게 알려주지 않는것으로 확장된다.

대신 X 는 이런 기능을 "윈도우 매니저"라고 부르는 어플리케이션에 위임한다. X 에 사용할 수 있는 **AfterStep, Blackbox, ctwm, Enlightenment, Fvwm, Sawfish, twm, Windows Maker** 와 더 다양한 윈도우 매니저가 있다. 이러한 윈도우 매니저들은 각자 다른 모양과 느낌을 제공한다. 이들 중 몇 개는 "가상 데스크톱"을 지원하고 어떤 것들은 데스크톱을 관리하기

---

위해 키를 지정할 수 있다. 어떤 윈도우 매니저는 "시작버튼" 같은 비슷한 장치를 가지고 있으며 다른 것들은 새로운 테마를 적용하여 보기 좋게 바꿀 수 있다. 이렇게 다양한 윈도우 매니저는 포트 컬렉션의 x11-wm 카테고리에서 사용할 수 있다.

게다가 KDE와 GNOME 데스크톱 환경은 데스크톱과 통합되는 각자의 윈도우 매니저를 가지고 있다.

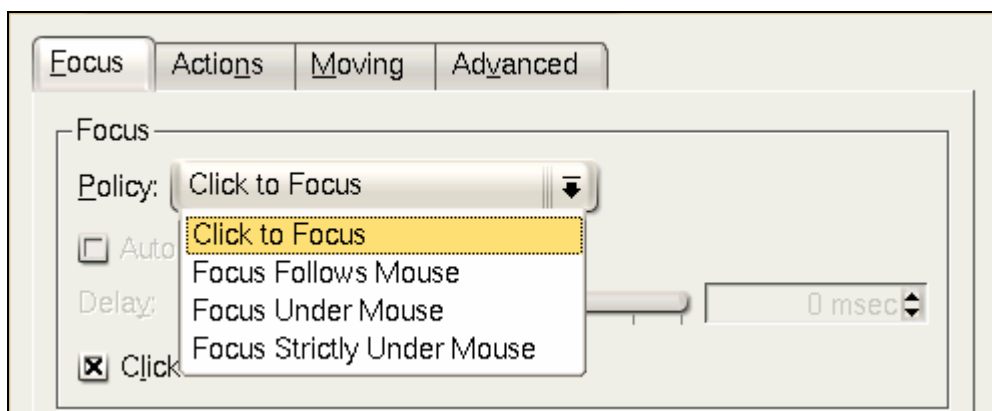
각 윈도우 매니저는 다른 설정 메커니즘 가지고 있다. 어떤 것은 직접 설정 파일을 작성해야 되고 다른 것은 대부분의 설정을 위한 GUI 툴 기능을 제공한다. 최소한 윈도우 매니저는(sawfish) 특수한 Lisp 언어로 작성된 설정파일을 가지고 있다.

**초점 방식:** 윈도우 매니저의 또 다른 기능은 마우스 "초점 방식"에 대한 기능이다. 모든 윈도우 시스템은 키 입력을 받을 수 있도록 윈도우를 선택하면 활성화된 윈도우를 보여준다.

이와 유사한 초점 방식은 "클릭으로 활성화"다. 이것은 마이크로소프트에서 사용하는 모델로 마우스로 클릭해서 윈도우를 활성화한다.

X는 특별한 활성화 방법을 지원하지 않는다. 대신 윈도우 매니저는 어떤 윈도우가 어느 때 초점이 맞춰있는지 제어한다. 다른 윈도우 매니저는 다른 초점 방식을 지원한다. 대부분은 클릭하면 활성화되는 방식을 지원하고 이들 중 대부분은 몇 가지 다른 방법도 지원한다.

#### 초점방식 설정 화면



---

가장 유명한 초점 방식은 다음과 같다.

#### focus-follow-mouse

마우스 포인터 아래의 윈도우가 활성화된다. 이것은 윈도우가 다른 윈도우의 가장 위에 있으면 필요 없을 것이다. 이 활성화는 클릭할 필요 없이 다른 윈도우를 지시하면 바뀐다.

#### sloppy-focus

이 방법은 focus-follows-mouse의 약간 확장된 방식이다. focus-follows-mouse에서 마우스가 root 윈도우(또는 백그라운드) 위로 이동하면 어떤 윈도우도 활성화되지 않고 키 입력은 단순히 사라진다. sloppy-focus에서 초점은 커서가 현재 위치한 윈도우가 아닌 새로운 윈도우에 들어갈때만 변경된다.

#### click-to-focus

활성화된 윈도우는 마우스 클릭으로 선택된다. 이 윈도우는 넓어져서 다른 윈도우 앞에 나타난다. 커서가 다른 윈도우로 이동하더라도 모든 키 입력은 이 윈도우로 입력된다.

많은 윈도우 매니저는 이렇게 다양한 방법을 지원한다. 윈도우 매니저의 문서를 참조한다.

## 5.2.4 위젯

"위젯"은 클릭할 수 있거나 어떤 방법으로 제어할 수 있는 유저 인터페이스의 모든 아이템을 의미하는 용어다; 버튼, 체크 박스, 라디오 버튼, 아이콘, 리스트 등을 의미한다. 마이크로소프트 윈도우는 이들을 "제어"라고 부른다.

마이크로소프트 윈도우와 애플의 MacOS는 아주 고정된 위젯 정책을 가지고 있다. 어플리케이션 개발자들은 어플리케이션이 똑같이 보여지고 느껴지기를 원한다. X에서는 특정 그래픽 스타일로 느낄 수 있거나 위젯 세트를 고정시키는 것을 고려하지 않았다.

이러한 이유로 X 어플리케이션이 똑 같은 느낌으로 보여지기를 기대하지 않는다. 최초의

---

MIT 아데나 위젯 세트, **Motif**( 마이크로소프트 윈도우가 모델로 채택한, 끝이 모두 각이 지고 회색의 3 단계 농도의 위젯 세트), **OpenLook** 과 다른 것들을 포함하여 인기 있는 위젯 세트와 변종이 많이 있다.

요즘 새로운 대부분의 X 어플리케이션은 KDE 에 사용되는 Qt 또는 GNOME 프로젝트에 사용되는 GTK 로 현대적으로 보이는 위젯 세트를 사용한다. 이러한 내용을 보면 유닉스 데스크톱의 모양을 초보자에게 쉽게 만드는 추세다.

## 5.3 XFree86 설치

XFree86 을 설치하기 전에 어떤 버전을 사용할지 결정해야 된다. XFree86 3.X 는 XFree86 의 개발분기에서 유지된다. 이 버전은 매우 안정적이고 수 많은 그래픽 카드를 지원한다. 그러나 이 소프트웨어에서 새로운 개발은 하지 않는다. XFree86 4.X 는 폰트지원이 좋아지고 화면 일그러짐을 방지하는 다양하고 새로운 특성을 가진 시스템으로 완벽하게 다시 디자인 됐다. 불행히 이 새로운 아키텍처에서 비디오 드라이버를 다시 작성 해야되기 때문에 3.X 에서 지원하던 오래된 카드 중 몇 개는 4.X 에서는 아직 지원하지 않는다. 이 분기에서 필요한 새로운 개발과 그래픽 카드 지원이 완료됐기 때문에 XFree86 4.X 는 이제 FreeBSD 4.X 윈도우 시스템의 기본버전이다.

FreeBSD 설치 프로그램은 유저에게 FreeBSD 를 설치 도중에 XFree86 4.X 를 설치하고 설정할 수 있는 기회를 제공한다(2.9.11 장에서 설명하였다). XFree86 3.X 설치하고 실행하려면 기본 FreeBSD 시스템이 설치될 때까지 기다린 후 XFree86 을 설치한다. 예를 들어 포트 컬렉션에서 XFree86 3.X 를 빌드하고 설치하려면 다음 명령을 실행한다.

```
# cd /usr/ports/x11/XFree86
# make all install clean
```

다른 방법은 XFree86 의 두 버전을 XFree86 웹사이트에서(<http://www.XFree86.org/>) 제공하는 FreeBSD 바이너리로 직접 설치할 수 있다. pkg\_add(1) 툴을 사용하는 바이너리 패키지 지도 XFree86 4.X 에 사용할 수 있다. pkg\_add(1)의 원격 패치 기능을 사용할때 패키지 버전번호를 삭제해야 한다. pkg\_add(1)는 자동으로 어플리케이션의 최신 버전으로 패치 하기 때문에 XFree86 4.X 의 패키지 패치와 설치는 다음 명령이면 된다.

```
# pkg_add -r XFree86
```

---

그리고 XFree86 4.X 를 설치하는데 포트 컬렉션도 사용할 수 있다. 포트 컬렉션 사용은 단순히 다음 명령을 입력하면 된다.

```
# cd /usr/ports/x11/XFree86-4
# make install clean
```

**Note:** 위 예제는 서버, 클라이언트, 폰트 등을 포함하여 모든 XFree86 배포본을 설치한다. 분리된 패키지와 XFree86 4.X 의 다른 포트도 사용할 수 있다.

이 장의 남은 부분에서 XFree86 을 어떻게 설정하고 효율적인 데스크톱 환경을 어떻게 설정하는지 설명한다.

## 5.4 XFree86 설정

### 5.4.1 시작하기 전에

XFree86 4.X 를 설정하기 전에 타겟 시스템에 관한 정보가 필요하다:

- 모니터 설명서
- 비디오 카드 칩셋
- 비디오 카드 메모리

모니터 설명서는 XFree86 을 실행할때 해상도와 주사율을 결정하는데 사용된다. 이러한 설명서는 일반적으로 모니터 매뉴얼이나 제조사의 웹사이트에서 얻을 수 있다. 필요한 두 가지 범위의 숫자는 수평 주사율과 수직 동조율이다.

비디오 카드 칩셋은 XFree86 이 어떤 드라이버 모듈을 사용하여 그래픽 하드웨어를 제어할지 정의한다. 대부분의 칩셋은 자동으로 설정되지만 정확히 검색되지 않으면 칩셋 정보를

---

알고있는것이 유용하다.

비디오 카드의 비디오 메모리는 해상도와 시스템이 구현할 수 있는 색상 농도를 규정한다. 역시 이 정보도 알고있는것이 유용하고 유저는 시스템의 한계를 알아야 한다.

## 5.4.2 XFree86 4.X 설정

XFree86 4.X의 설정은 여러 단계를 거친다. 첫 단계는 `-configure` 옵션으로 XFree86을 초기 설정 파일로 빌드한다. 슈퍼 유저에서 단순히 다음 명령을 실행한다.

```
# XFree86 -configure
```

위 명령은 /root 디렉터리에 XF86Config.new 라는(사용되는 디렉터리는 환경변수 \$HOME과 가지고있는 슈퍼유저 권한에 따라 다르다) 기본적인 XFree86 설정 파일을 생성한다. XFree86 프로그램은 시스템에서 그래픽 하드웨어를 탐색하고 검색된 하드웨어에 적당한 드라이버를 로드하기 위해 설정파일을 작성한다.

다음 단계는 타겟 시스템에서 XFree86이 그래픽 하드웨어와 동작할 수 있는지 확인하는 설정 테스트다. 이 작업을 수행하기 위해 유저는 다음 명령을 실행해야 된다:

```
# XFree86 -xf86config XF86Config.new
```

검고 회색의 격자눈금과 X 마우스 커서가 나타나면 설정은 성공적이다. 테스트를 빠져 나오기 위해 **Ctrl+Alt+Backspace**를 동시에 누른다.

**Note:** 마우스가 작동하지 않는다면 장치가 제대로 설정되었는지 확인한다. FreeBSD 설치에서 2.9.10 장을 본다.

다음은 원하는 대로 XF86Config.new 설정 파일을 조정한다. emacs(1)나 ee(1)와 같은 텍스트 에디터로 파일을 연다. 첫째로 타겟 시스템의 모니터 주파수를 추가한다. 이것은 보통 수평주파수와 수직 동조율로 나타난다. 이러한 값을 XF86Config.new 파일의 "Monitor" 섹션 아래에 추가한다:

Section "Monitor"	
Identifier	"Monitor0"

```

VendorName      "Monitor Vendor"
ModelName       "Monitor Model"
HorizSync       30-107
VertRefresh     48-120
EndSection

```

*HorizSync* 와 *VertRefresh* 키워드는 설정 파일에 없을 것이다. 키워드가 없다면 정확한 수직 동조율을 Horizsync 키워드 다음에 그리고 수평 주파수는 *VertRefresh* 키워드 다음에 추가한다. 위의 예제에서 타겟 모니터의 주사율을 입력하였다.

X는 가능한 모니터에서 DPMS(Energy Star) 기능을 사용할 수 있다. xset(1) 프로그램은 타임 아웃과 대기, 중지 또는 off 모드를 강제로 수행할 수 있다. 모니터에 DPMS 기능을 사용하려면 다음 라인을 모니터 섹션에 추가해야 된다:

```

Option          "DPMS"

```

XF86Config.new 설정 파일에 원하는 기본 해상도와 색상 농도를 선택한다. 이것은 "Screen" 섹션에 정의되어 있다:

```

Section "Screen"
    Identifier      "Screen0"
    Device          "Card0"
    Monitor         "Monitor0"
    DefaultDepth    24
    SubSection      "Display"
        Depth       24
        Modes       "1024x768"
    EndSubSection
EndSection

```

*DefaultDepth* 키워드는 기본적으로 실행하는 색상농도를 설명한다. 이것은 XFree86(1)에 맞게 -bpp 명령어 라인으로 수정할 수 있다. *Modes* 키워드는 주어진 색상 농도로 실행하도록 해상도를 표시한다. 타겟 시스템의 그래픽 하드웨어에 정의된 것처럼 오직 VESA 표준 모드만 지원된다. 위 예제의 기본 색상 농도는 픽셀당 24 비트이다. 이 색상 농도에서 허용되는 해상도는 1024 X 768 이다.

---

마지막으로 설정 파일을 작성하고 위에서처럼 테스트 모드로 테스트한다. 모든것이 정상이라면 설정파일을 XFree86(1)이 찾을 수 있는 일반적인 위치에 설치해야된다. 이곳은 보통 /etc/X11/XF86Config 나 /X11R6/etc/X11/XF86Config 다.

```
# cp XF86Config.new /etc/X11/XF86Config
```

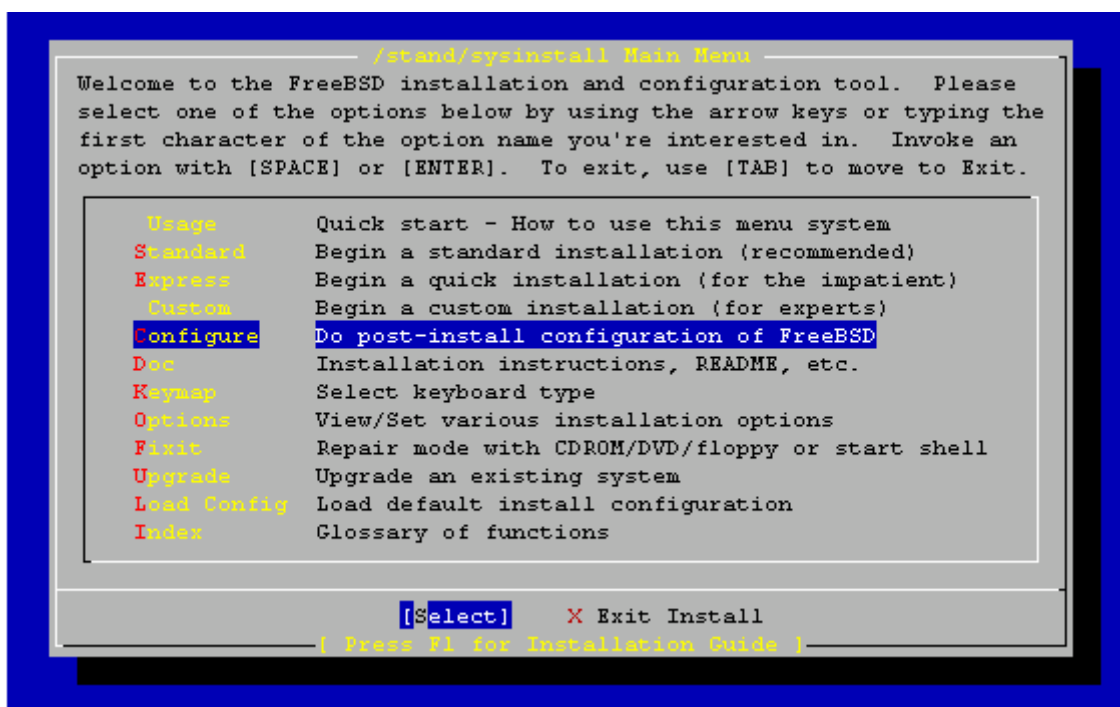
설정파일을 적절한 위치에 놓으면 설정이 끝난다. startx(1)로 XFree86 4.X 를 시작하려면 x11/wrapper 포트를 설치한다. 또한 XFree86 4.X 는 xdm(1)으로 실행할 수 있다.

**Note:** XFree86 4.X 배포 본에 xf86cfg(1)이라는 그래픽컬 설정 틀이있다. 적절한 드라이버와 설정을 선택하여 대화식으로 설정할 수 있다. 이 프로그램은 xf86cfg - textmode 명령으로 콘솔에서도 사용할 수 있다. 더 자세한 내용은 xf86cfg(1) 매뉴얼 페이지를 참고한다.

#### [예제: Sysinstall 을 이용한 XFree86 설정]

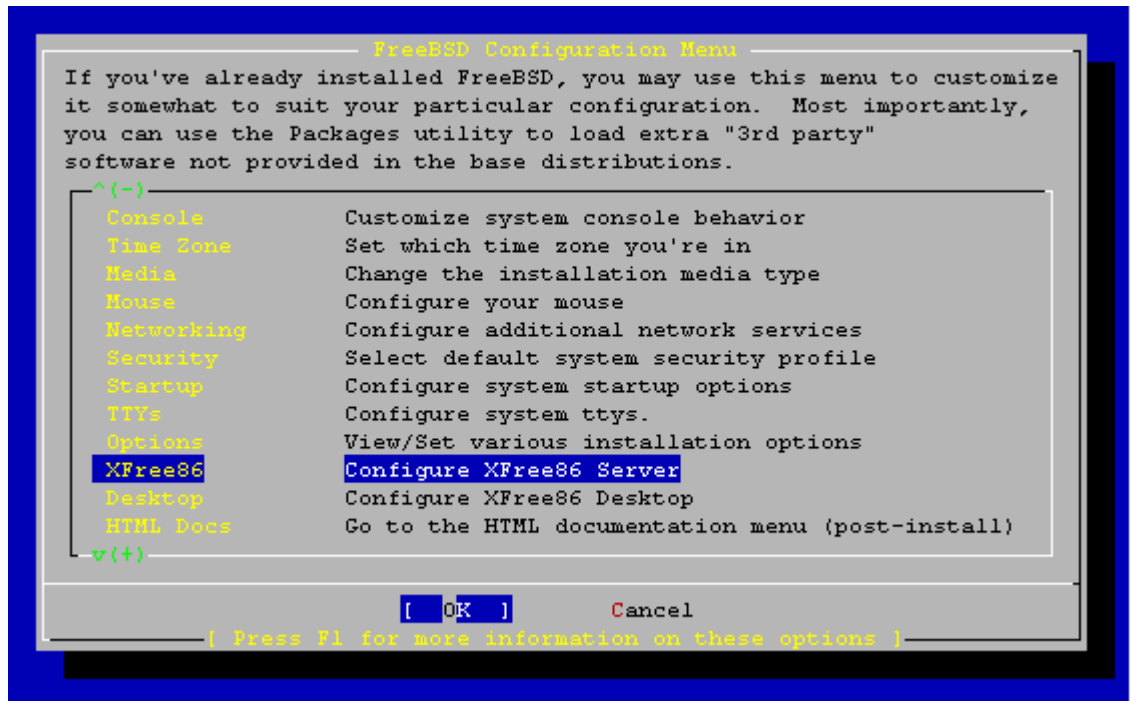
1. 먼저 다음 명령으로 Sysinstall 을 실행한다.

```
# /stand/sysinstall
```

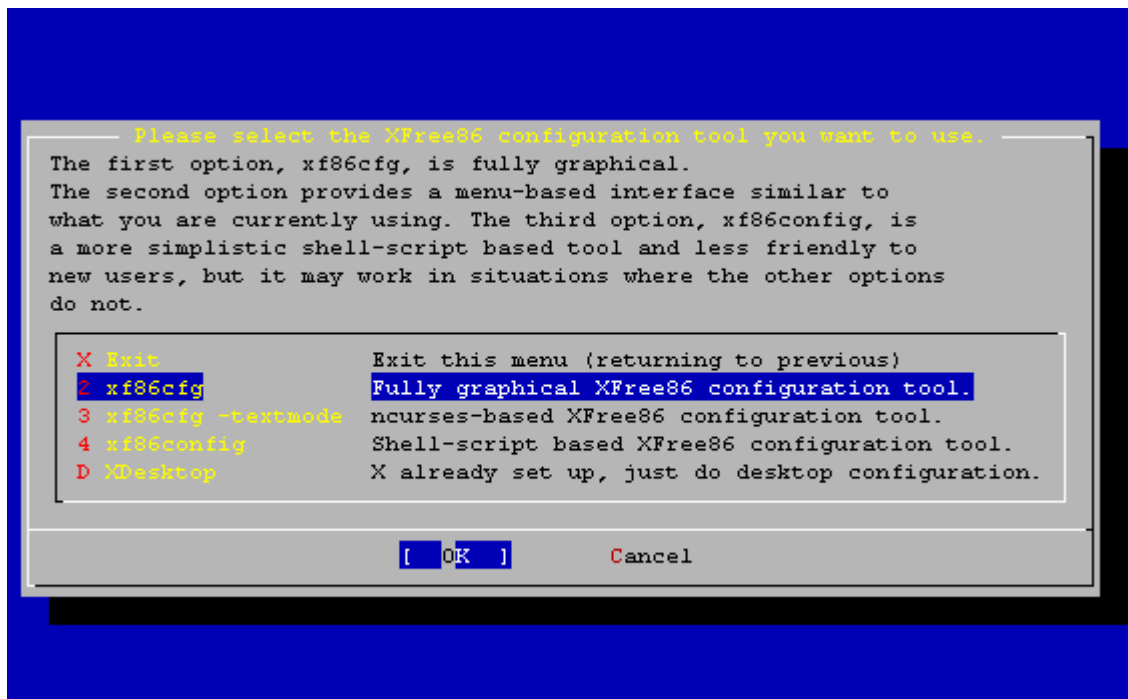


2. 아래 그림에서처럼 Configure → XFree86 을 선택한다.



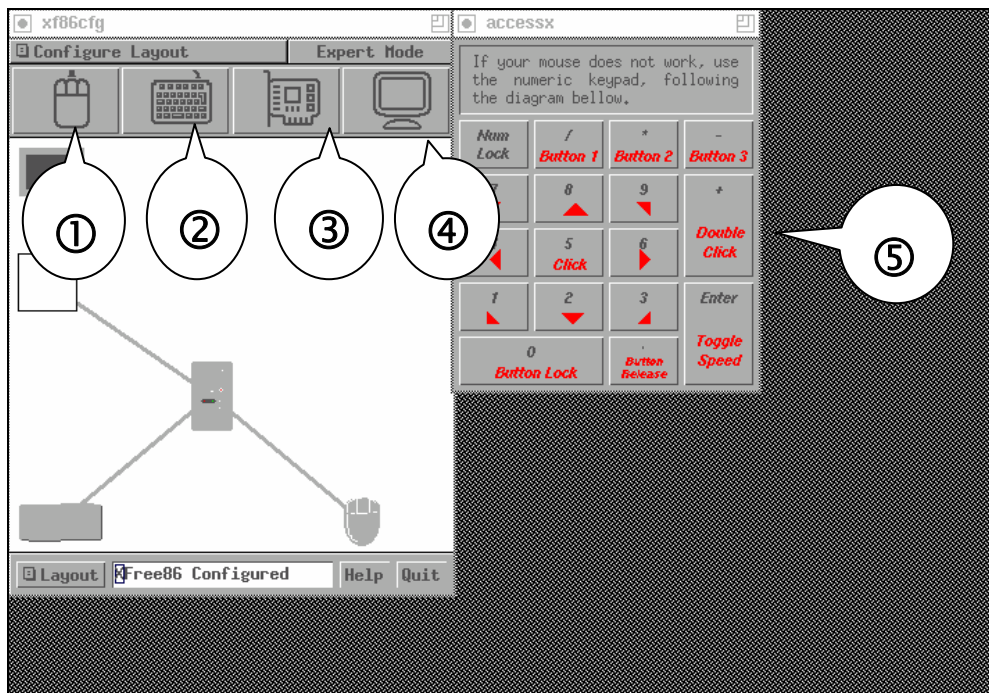


3. 그림처럼 XFree86 을 설정하는 3 가지 도구가 제공된다.



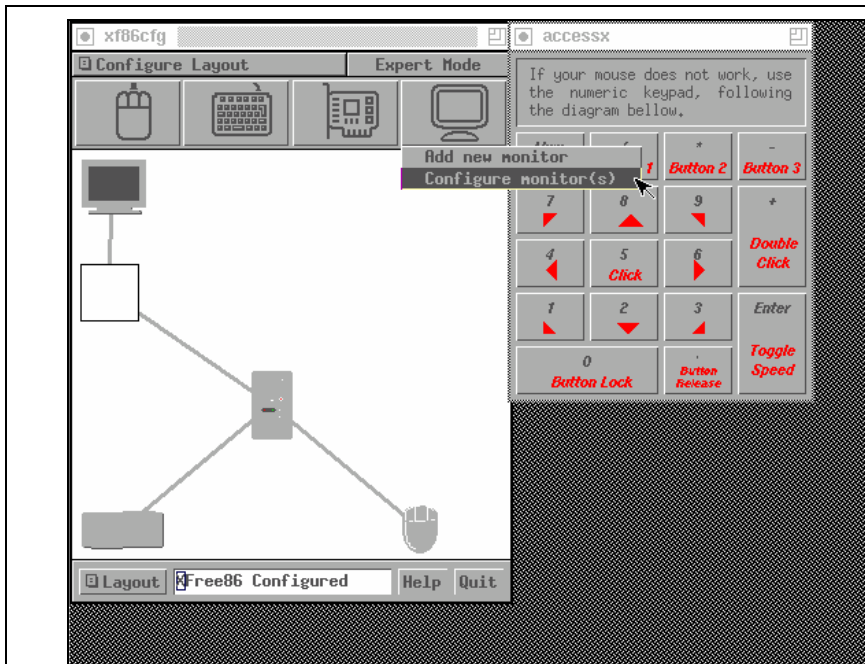
- **xf86cfg**: 그래픽으로된 설정 도구로 마우스와 키보드를 사용하여 설정할 수 있다.
- **xf86cfg -textmode**: sysinstall 이미지처럼 되어있다.
- **xf86config**: 완벽한 텍스트 기반의 스크립트다.

4. **xf86cfg** 를 선택하고 **Tab** 키를 눌러 [ OK ]를 누른다.

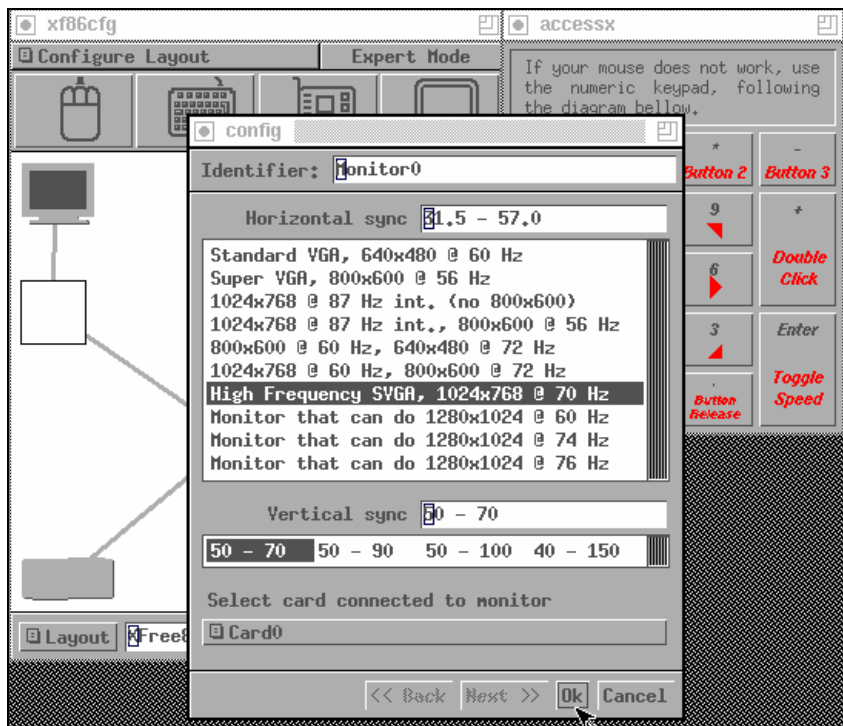


- ① 마우스를 설정할 수 있다.
- ② 키보드를 설정할 수 있다.
- ③ VGA 카드를 설정할 수 있다.
- ④ 마우스가 움직이지 않을때 그림처럼 숫자 키 패드를 사용하여 이동할 수 있다.

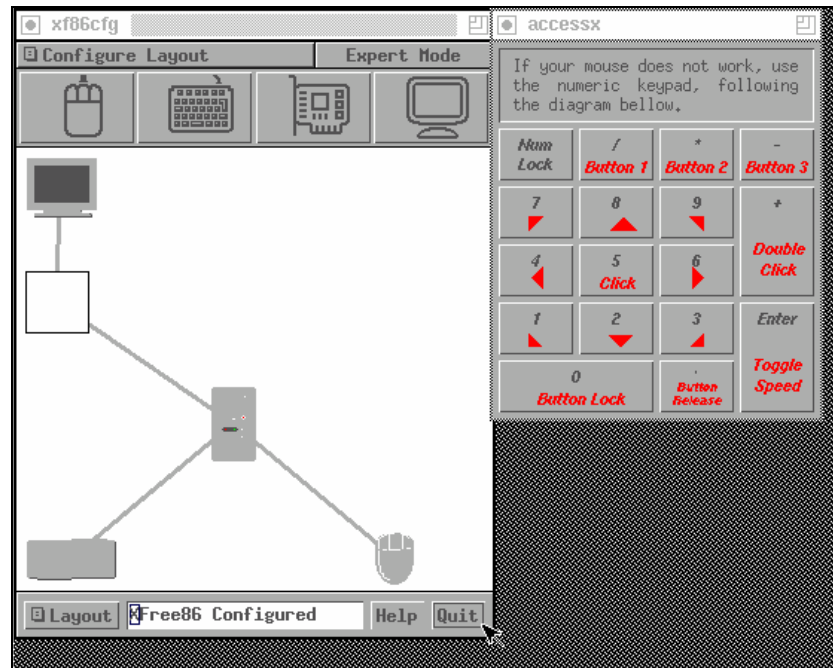
5. 그림과 같이 아이콘을 누르면 설정 메뉴가 나타난다.



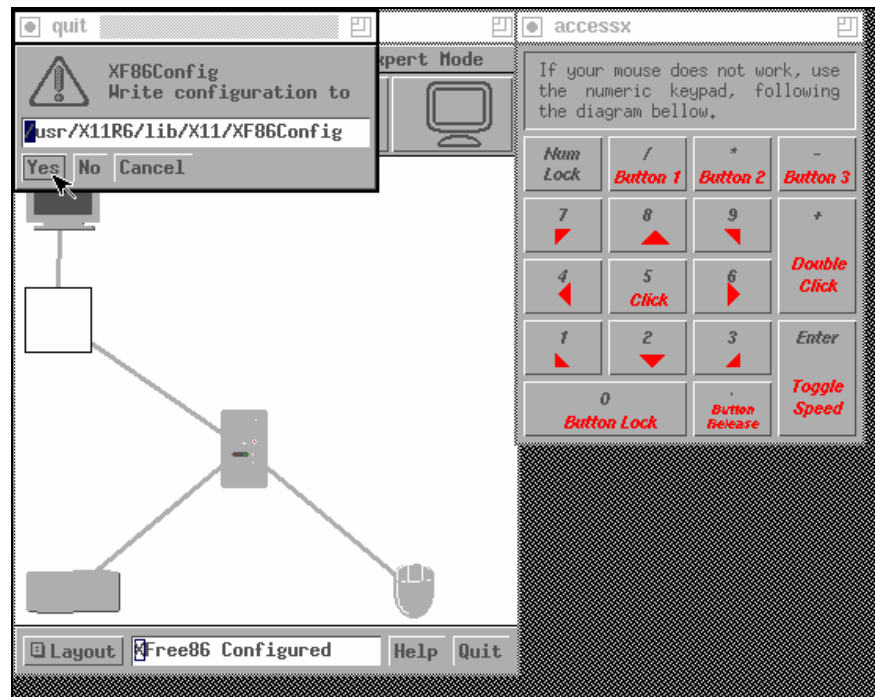
6. 아이콘 별로 설정하고 Ok 버튼을 클릭한다.



7. 설정을 나가기 위해 Quit 버튼을 클릭한다.



8. 설정한 내용을 파일로 작성할 것인지 묻는 대화상자가 나타난다. Yes 를 선택하고 **xf86cfg** 를 끝낸다.



---

## 5.4.3 고급 설정

### 5.4.3.1 Intel i810 그래픽 칩셋 설정

인텔 i810 통합 칩셋을 설정하려면 카드 드라이브에 맞는 **XFree86** 용 `agpgart` AGP 프로그래밍 인터페이스가 필요하다. `agp(4)` 드라이버는 4.8-RELEASE 와 5.0-RELEASE 부터 GENERIC 커널에 추가 되었다. 이전 릴리즈에서는 다음 라인을 커널 설정 파일에 추가하고 새로운 커널을 다시 빌드해야 된다:

```
device agp
```

부팅 할때 `agp.ko` 커널 모듈을 `loader(8)`를 사용하여 자동으로 로드하려면 다음 라인을 `/boot/loader.conf` 에 추가한다:

```
agp_load="YES"
```

그리고 FreeBSD 4.X 나 이전 버전을 사용 중 이라면 장치 노드가 프로그래밍 인터페이스를 생성해야 된다. AGP 장치 노드를 생성하려면 `/dev` 디렉터리에서 `MAKEDEV(8)`를 실행한다.

```
# cd /dev
```

```
# sh MAKEDEV agpgart
```

**Note:** FreeBSD 5.X 와 이후 버전은 `devfs(5)`로 장치 노드를 할당하기 때문에 `MAKEDEV(8)` 단계가 필요없다.

이 명령은 다른 그래픽 보드처럼 하드웨어를 설정한다. `agp(4)` 드라이버가 커널에 컴파일되지 않은 시스템에서 `kldload(8)`로 모듈을 로드하려는 시도는 수행되지 않는다. 이 드라이버는 부팅할 때부터 컴파일 될 때까지 커널에 있거나 `/boot/loader.conf` 를 사용한다.

**XFree86 4.1.0** 을 사용 중이고(또는 그 후 버전) `fbPictureInit` 처럼 이해할 수 없는 문자가 나타났다면 **XFree86** 설정파일의 `Driver "i810"` 다음에 아래 라인을 추가한다:

```
Option "NoDDC"
```

---

## 5.5 XFree86 에서 폰트 사용

### 5.5.1 타입 1 폰트

XFree86 에 적용된 기본 폰트는 전형적인 데스크톱 출판 어플리케이션에 필요한 것보다 작다. 크게 표시된 폰트는 지그재그처럼 어설프게 보이고 Netscape 의 작은 폰트는 아주 난해해 보인다. 그러나 XFree86 버전 3.X 나 4.X 에 무료로 쉽게 사용할 수 있는 고 품질의 타입 1 폰트(PostScript)가 몇개 있다. 예를들면 URW 폰트 컬렉션은([x11-fonts/urwfonts](#)) 고 품질의 표준 타입 1 폰트를(Times Roman, Helvetica, Palatino 와 더 많은 종류) 포함하고 있다. 이 무료 폰트 컬렉션은([x11-fonts/freefonts](#)) 많은 폰트를 가지고 있지만 이들 중 대부분은 Gimp 같은 그래픽 소프트웨어를 위한 것이어서 화면 폰트는 충분하지 않다. 그렇지만 약간의 노력으로 XFree86 이 투루 타입 폰트를 사용하도록 설정할 수 있다. 나중에 투루 타입 폰트에 관한 섹션을 본다.

위의 타입 1 폰트 모음을 포트 컬렉션에서 설치하려면 다음 명령을 실행한다:

```
# cd /usr/ports/x11-fonts/urwfonts
# make install clean
```

그리고 무료폰트나 다른 폰트 모음처럼 이런 폰트가 있다는 것을 X 서버에게 알려주기 위해 적당한 라인을 XF86Config( XFree86 버전 3 은 /etc/에 그리고 버전 4 는 /etc/X11/에) 파일에 추가한다:

```
FontPath "/usr/X11R6/lib/X11/fonts/URW/"
```

다른 방법은 X 섹션의 명령어 라인에서 다음 명령을 실행한다:

```
% xset fp+ /usr/X11R6/lib/X11/fonts/URW
% xset fp rehash
```

이 설정은 정상적으로 동작하지만 시작 파일에(보통 startx 세션은 ~/.xinitrc 에 그리고 XDM 같은 그래픽컬 로그인 매니저로 로그인할 때는 ~/.xsession 에 추가한다) 추가하지 않으면 X 세션이 닫힐때 없어진다. 세 번째 방법은 새로운 XftConfig 파일을 이용하는 것이다: 일그러짐 방지에 관한 섹션을 본다.

---

## 5.5.2 투루 타입 폰트

XFree86 4.X 는 투루 타입 폰트 표시 지원이 내장되어있다. 이 기능을 활성화하는 두 가지 모듈이있다. 이 예제에서는 배경의 다른 폰트와 더 어울리기 때문에 프리타입 모듈을 사용한다. 프리타입 모듈을 사용하려면 다음 라인을 /etc/X11/XF86Config 파일의 "Module" 섹션에 추가한다.

```
Load "freetype"
```

XFree86 3.3.X 는 투루 타입 폰트 서버를 분리해야 된다. Xfstt 는 보통 이러한 목적에 사용된다. Xfstt 설치 는 단순히 x11-servers/Xfstt 포트를 설치한다.

이제 투루 타입 폰트(예를 들면 /usr/X11R6/lib/X11/fonts/TrueType) 디렉터리를 만들고 모든 투루 타입 폰트를 이 디렉터리로 복사해 넣는다. 투루 타입 폰트는 직접 Macintosh 에서 가져올 수 없다; XFree86 이 사용하기 위해서는 유닉스/DOS/윈도우 포맷이어야 한다. 파일이 이 디렉터리에 복사되면 ttmkfdir 을 사용하여 fonts.dir 파일을 만들어서 X 폰트 표시기가 이들 새로운 파일이 설치된 것을 알수 있다. ttmkfdir 은 FreeBSD 포트 컬렉션의 x11-fonts/ttmkfdir에서 이용할 수 있다.

```
# cd /usr/X11R6/lib/X11/fonts/TrueType
# ttmkfdir > fonts.dir
```

이제 투루 타입 디렉터리를 폰트 경로에 추가한다. 이 방법은 위의 타입 1 폰트에서 설명한 것과 같다. 다음 명령을 이용하거나 FontPath 라인을 XF86Config 파일에 추가한다.

```
% xset fp+ /usr/X11R6/lib/X11/fonts/TrueType
% xset fp rehash
```

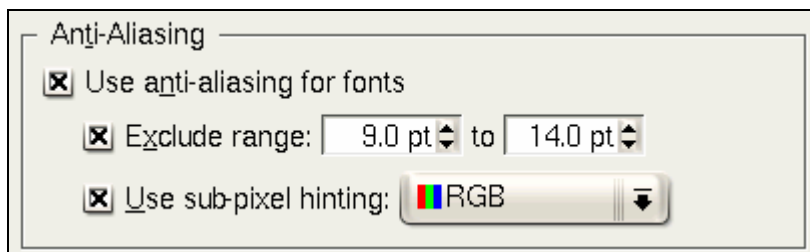
이제 Netscape, Gimp, StarOffice 와 다른 X 어플리케이션들이 설치된 투루 타입 폰트를 인식한다. 극히 작은 폰트(고해상도에서 웹 페이지 텍스트 같은)와 아주 커다란 폰트는 (StarOffice 에서) 이제 훨씬 보기 좋을것이다.

---

### 5.5.3 Anti-Aliased 폰트

Anti-aliasing 는 XFree86 4.0.2 부터 사용할 수 있었다. 그러나 폰트 설정은 **XFree86 4.3.0** 이 도입되기 전에는 귀찮은 일이었다. 버전 4.3.0 이 시작되면서, /usr/X11R6/lib/X11/fonts/의 모든 폰트와 ~/.fonts/는 자동으로 Xft-aware 어플리케이션에 anti-aliasing 을 사용할 수 있도록 만들어졌다. 모든 어플리케이션은 아니었지만 대부분 Xft 지원을 받았다. Xft-aware 어플리케이션의 예제는 Qt2.3 과 더 높은 버전(KDE 데스크톱 툴킷), Gtk+ 2.0 과 더 높은 버전(GNOME 데스크톱 툴킷) 그리고 **Mozilla** 1.2 와 더 높은 버전을 포함한다.

#### KDE 에서 Anti-Aliasing 폰트 사용



anti-alias 된 폰트를 제어하거나 anti-aliasing 기능을 설정하기 위해 /usr/X11R6/etc/fonts/local.conf 파일을 생성한다(이미 있다면 편집한다). Xft 폰트 시스템의 발전된 몇 가지 기능은 이 파일을 사용하여 조정할 수 있다. 이번 섹션은 몇가지 간단한 기능만 설명하므로 더 자세한 사항은 fonts-conf(5)를 본다.

이 파일은 XML 포맷이어야 된다. 이 사항을 주의하고 모든 태그는 정확히 달아야 된다. 파일은 일반 XML 헤더로 시작하고 DOCTYPE 정의를 따르며 <fontconfig>는 태그다:

```
<?xml version="1.0"?>
<!DOCTYPE fontconfig SYSTEM "fonts.dtd">
<fontconfig>
```

이전 상태에서 ~/.fonts/처럼 /usr/X11R6/lib/X11/fonts/의 모든 폰트도 Xft-aware 어플리케이션에서 사용할 수 있도록 만들어졌다. 이 두 디렉터리 트리밖에 다른 디렉터리를 추가하려면 /usr/X11R6/etc/fonts/local.conf 에 다음과 비슷한 라인을 추가한다:

```
<dir>/path/to/my/fonts</dir>
```



---

새로운 폰트를 추가하고 특히 새로운 폰트 디렉터리와 폰트 케시를 다시 빌드하기 위해 다음 명령을 실행한다:

```
# fc-cache -f
```

Anti-aliasing 은 매우 작은 텍스트를 읽기 쉽게 만들고 큰 텍스트에서 계단 같은 모양을 삭제하지만 보통 텍스트에 적용한다면 눈의 피로를 유발할 수 있는 가장자리를 약간 흐리게 만든다. anti-aliasing 에서 글자 크기가 14 포인트보다 작아지는 것을 방지하기 위해 다음 라인을 포함한다:

```
<match target="font">
    <test name="size" compare="less">
        <double>14</double>
    </test>
    <edit name="antialias" mode="assign">
        <bool>>false</bool>
    </edit>
</match>
<match target="font">
    <test name="pixelsize" compare="less" qual="any">
        <double>14</double>
    </test>
    <edit mode="assign" name="antialias">
        <bool>>false</bool>
    </edit>
</match>
```

모노 스페이스 폰트의 글자 간격은 anti-aliasing 에 적당하지 않을 것이다. 이것은 특히 KDE 와 문제가 되는것 같다. 이 문제를 해결하는 한가지는 강제로 이런 폰트 간격을 100 으로 고정한다. 다음 라인들을 추가하고

```
<match target="pattern" name="family">
    <test qual="any" name="family">
        <string>fixed</string>
    </test>
    <edit name="family" mode="assign">
```

---

```
<string>mono</string>
</edit>
</match>
<match target="pattern" name="family">
  <test qual="any" name="family">
    <string>console</string>
  </test>
  <edit name="family" mode="assign">
    <string>mono</string>
  </edit>
</match>
```

(이 엘리먼트에서 고정된 폰트의 보통 이름은 “mono”다), 다음 내용도 추가한다:

```
<match target="pattern" name="family">
  <test qual="any" name="family">
    <string>mono</string>
  </test>
  <edit name="spacing" mode="assign">
    <int>100</int>
  </edit>
</match>
```

Helvetica 같은 폰트는 anti-alias가 될 때 문제를 발생할 수 있다. 보통 이것은 폰트가 새로로 갈려져 보인다. 최악의 경우 **Mozilla** 같은 어플리케이션을 충돌하게 한다. 이 문제를 피하기 위해 local.conf 파일에 다음 라인을 추가한다:

```
<match target="pattern" name="family">
  <test qual="any" name="family">
    <string>Helvetica</string>
  </test>
  <edit name="family" mode="assign">
    <string>sans-serif</string>
  </edit>
</match>
```

---

local.conf 수정을 끝내고 </fontconfig> 태그로 파일을 끝냈는지 확인한다. 이렇게 하지 않으면 변경한것이 무시될 수 있다.

**XFree86** 의 기본 폰트 세트는 anti-aliasing 에 적절하지 않다. 더 좋은 기본 폰트 세트는 x11-fonts/bitstream-vera 폰트에서 찾을 수 있다. 이 폰트는 /usr/X11R6/etc/fonts/local.conf 파일이 없으면 설치한다. 파일이 없다면 폰트는 /usr/X11R6/etc/fonts/local.conf-vera 파일을 생성한다. 이 파일의 내용을 /usr/X11R6/etc/fonts/local.conf 에 넣고 Bitstream 폰트는 기본 **XFree86** Serif, Sans Serif 와 모노 스페이스 폰트로 대체된다.

마지막으로 유저는 개인적인 .fonts.conf 파일로 원하는 설정을 추가할 수 있다. 원하는 설정을 추가하려면 각 유저는 단순히 ~/.fonts.conf 을 생성한다. 이 파일도 XML 포맷이어야 한다.

**마지막 포인트:** LCD 화면에서 서브픽셀 샘플링이 요구될 것이다. 이것은 기본적으로 수평 해상도를 증진하기 위해 적색, 녹색과 파란색을 개별적인 컴포넌트로(수평분할) 간주한다; 이 결과는 획기적이다. 이것을 사용하려면 local.conf 파일 어딘가에 다음 라인을 추가한다:

```
<match target="font">
  <test qual="all" name="rgba">
    <const>unknown</const>
  </test>
  <edit name="rgba" mode="assign">
    <const>rgb</const>
  </edit>
</match>
```

**Note:** 화면 타입에 따라서 *rgb* 를 *bgr*, *vrgb* 또는 *vbgr* 로 변경할 필요가 있을 것이다; 어느 것이 가장 좋은지 직접 확인해본다.

Anti-aliasing 은 X 서버가 시작된 다음에 사용할 수 있다. 그러나 프로그램은 이 장점을 어떻게 활용하는지 알아야 한다. 현재 Qt 툴킷은 알고 있기 때문에 전체적인 KDE 환경은 anti-aliase 폰트를(자세한 내용은 5.7.3.2 장을 본다) 사용할 수 있다. Gtk+와 GNOME 도 "Font" capplet 으로 anti-aliasing 를(자세한 내용은 5.7.1.3 장을 본다) 사용할 수 있다. 기본적으로 Mozilla 1.2 와 이후 버전은 자동으로 anti-aliasing 을 이용한다. Anti-aliasing 을

---

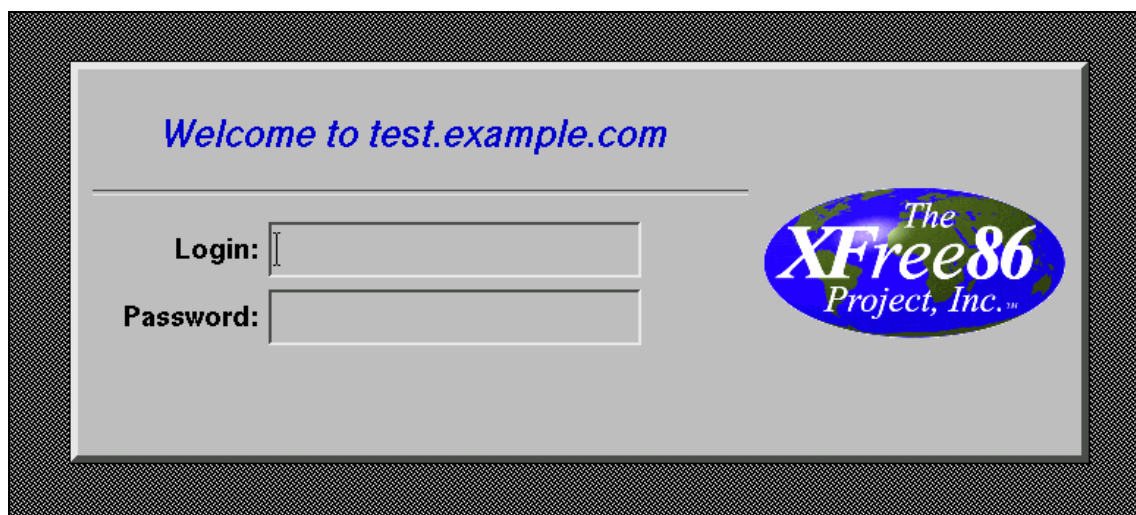
사용하지 않으려면 `-DWITHOUT_XFT` 플래그로 **Mozilla** 를 다시 빌드 한다.

## 5.6 X 디스플레이 매니저

### 5.6.1 개요

X 디스플레이 매니저는(XDM) 로그인 세션을 관리 하는 X 윈도우 시스템의 부가적인 부분이다. 이것은 최소한의 "X 터미널" 및 데스크톱과 대형 네트워크 화면 서버를 포함하는 몇가지 상황에 유용하다. 네트워크로 연결된 다른 머신에서 X 클라이언트와 서버를 운용하기 위해 광범위하고 다양한 설정이 가능하기 때문에 X 윈도우 시스템은 네트워크와 프로토콜에 영향을 받지않는다. XDM 은 디스플레이 서버를 연결한 후 로그인과 패스워드 조합 같은 인증정보를 입력하는 그래픽 인터페이스를 제공한다.

기본 XFree86 XDM



getty(8) 유틸리티에서 유저에게 같은 기능을 제공하는 XDM 을 생각해보자 (자세한 사항은 20.3.2 장을 본다). 이것은 디스플레이가 연결된 시스템에 로그인하고 유저를 대신해서 세션 매니저를 실행한다(일반적으로 X 윈도우 매니저). 그리고 XDM 은 이 프로그램에서 나가기 위해 유저가 모든 일을 끝나치고 디스플레이에서 빠져 나왔다는 신호를 기다린다. 여기서 XDM 은 다음 유저가 로그인 하도록 로그인 화면과 화면 선택을 표시한다.

---

## 5.6.2 XDM 사용

XDM 데몬 프로그램은 `/usr/X11R6/bin/xdm` 에 있다. 이 프로그램은 `root` 로 언제나 실행할 수 있고 로컬 머신에서 X 디스플레이 관리를 시작할 수 있다. XDM 을 머신이 부팅할 때마다 실행하려면 `/etc/ttys` 에 엔트리를 추가 한다. 이 파일의 포맷과 사용법에 관한 더 많은 정보는 20.3.2.1 장을 본다. `/etc/ttys` 파일에 가상 터미널에서 XDM 데몬을 실행하는 라인이 있다:

```
ttyv8  "/usr/X11R6/bin/xdm -nodaemon" xterm off secure
```

기본적으로 이 엔트리는 비활성되어 있다; 활성화하기 위해 필드 5 를 `off`에서 `on`으로 변경하고 20.3.2.2 장의 지시에 따라 `init(8)`을 다시 시작한다. 이 프로그램이 관리하는 터미널의 첫 번째 필드 이름은 `ttyv8`이다. 이것은 XDM 이 9 번째 가상 터미널에서 실행된다는 의미다.

## 5.6.3 XDM 설정

XDM 설정 디렉터리는 `/usr/X11R6/lib/X11/xdm` 에 있다. 이 디렉터리에 XDM 의 동작과 상황을 변경하는데 사용하는 몇 개의 파일이 있다. 일반적으로 다음과 같은 파일이다:

파일	설명
Xaccess	클라이언트 인증 룰 세트.
Xresources	기본 X 리소스 값.
Xservers	관리하려는 로컬과 원격 디스플레이 리스트.
Xsession	로그인을 위한 기본 세션 스크립트.
Xsetup_*	로그인 인터페이스가 시작하기 전에 실행하는 어플리케이션 스크립트.
xdm-config	이 머신에서 실행하는 모든 디스플레이의 전체 설정.
xdm-errors	서버 프로그램이 생성하는 에러.
xdm-pid	현재 실행중인 XDM 프로세스 ID.

또한 이 디렉터리에는 XDM 이 실행 중일 때 데스크톱 설정에 사용하는 몇개의 스크립트와 프로그램이 있다. 이런 파일들의 목적을 간단하게 설명하겠다. 이런 파일의 정확한 구문과 사용법은 `xdm(1)`에 설명되어있다.

---

기본 설정은 간단한 사각형 로그인 화면과 머신의 호스트 네임이 최상단에 표시되고 아래에 커다란 폰트로 "Login:"과 "Password:" 프롬프트가 나타난다.

### 5.6.3.1 Xaccess

디스플레이를 제어하기 위해 **XDM**에 연결한 프로토콜을 X 디스플레이 제어 연결 프로토콜이라고(XDMCP) 한다. 이 파일은 원격 머신에서 **XDMCP** 연결을 제어하기 위한 룰 세트다. 기본적으로 어떤 클라이언트라도 연결할 수 있지만 `xdm-config`로 원격 연결 대기를 변경하지 않는 이상 문제되지 않는다.

### 5.6.3.2 Xresources

이 파일은 디스플레이 선택자와 로그인 화면의 어플리케이션 기본 파일이다. 여기서 로그인 프로그램이 어디에 나타날지 수정할 수 있다. 이 포맷은 **XFree86** 문서에서 설명한 `app-default` 파일과 동일하다.

### 5.6.3.3 Xservers

이 파일은 원격 디스플레이 리스트고 선택자를 선택하면 제공된다.

### 5.6.3.4 Xsession

이 파일은 사용자가 로그인한 후 **XDM**을 실행하기 위한 기본 세션 스크립트다. 일반적으로 각 유저는 기본 세션 스크립트를 무시하는 최적화된 세션 스크립트를 `~/.xsession`에 가지고 있다.

### 5.6.3.5 Xsetup\_\*

이 파일은 선택자나 로그인 인터페이스가 나타나기 전에 자동으로 실행된다. `Xsetup_` 뒤에 로컬 디스플레이 번호가(예를 들면 `Xsetup_0`) 따라오며 각 디스플레이가 사용하는 스크립

---

트가 있다. 보통 이런 스크립트는 xconsole 처럼 백그라운드에서 한개 또는 두개의 프로그램을 실행한다.

### 5.6.3.6 xdm-config

이 파일은 설치 관리를 모든 디스플레이에 적용할 수 있는 app-defaults 폼 설정을 담고 있다.

### 5.6.3.7 xdm-errors

이 파일은 XDM 이 실행하려는 X 서버의 결과를 가지고 있다. XDM 화면이 어떤 이유에서 멈추었다면 이곳에서 에러 메시지를 확인할 수 있다. 그리고 이런 메시지는 세션마다 유저의 ~/.xsession-errors 파일에 작성되어 있다.

## 5.6.4 네트워크 디스플레이 서버 실행

디스플레이 서버에 다른 클라이언트를 연결하려면 접근 제어 룰을 편집하고 연결 리스너를 활성화한다. 기본적으로 이 곳에 보편적인 값을 설정한다. XDM 이 연결을 대기하도록 하려면 xdm-config 파일의 다음 내용을 주석처리 한다.

```
! SECURITY: do not listen for XDMCP or Chooser requests
! Comment out this line if you want to manage X terminals with xdm
DisplayManager.requestPort: 0
```

그리고 XDM 을 다시 시작한다. app-defaults 파일의 주석은 일반적으로 사용하는 “#”이 아닌 “!”문자로 시작된다는 것을 기억한다. 아주 강력한 접근제어가 요구될 수 있다. Xaccess 의 예제 엔트리를 확인하고 xdm(1) 매뉴얼 페이지를 참조한다.

## 5.6.5 XDM 교체

기본 XDM 프로그램을 대체하는 몇 개의 프로그램이 있다. 그 중 하나인 KDM 은(KDE 에 포함된) 다음 장에서 설명한다. KDM 은 로그인할 때 유저가 윈도우 매니저를 선택할 수 있으며 다양하고 시각적으로 보기 좋게 꾸미는 기능을 제공한다.

---

## 5.7 데스크톱 환경

이번 장은 FreeBSD의 X에 이용할 수 있는 다른 데스크톱 환경을 설명한다. "데스크톱 환경"은 간단한 윈도우 매니저부터 KDE와 GNOME처럼 완벽한 데스크톱 어플리케이션까지 어떤 것이라도 의미할 수 있다.

### 5.7.1 GNOME

#### 5.7.1.1 GNOME에 대하여

GNOME은 사용자가 쉽게 사용하고 컴퓨터를 설정할 수 있게 하는 사용자 친화적인 데스크톱 환경이다. GNOME은 패널(Panel)(어플리케이션을 시작하고 상태를 표시하는 곳), 데스크톱(데이터와 어플리케이션을 놓을 수 있는 곳), 표준 데스크톱 툴과 어플리케이션 세트 그리고 어플리케이션이 연동되어 일관성을 유지할 수 있도록 하는 관례를 포함한다. 다른 운영체제나 다른 환경의 유저는 GNOME이 지원하는 강력한 그래픽 운용환경을 각자의 환경에서 사용하여 좋은 느낌을 얻을 수 있다. FreeBSD의 GNOME에 관한 더 많은 정보는 FreeBSD GNOME 프로젝트의 웹 사이트에서(<http://www.FreeBSD.org/gnome>) 찾을 수 있다.





---

### 5.7.1.2 GNOME 설치하기

GNOME 을 설치하는 가장 쉬운 방법은 2 장에서 설명된 FreeBSD 설치 과정 중 "Desktop Configuration" 메뉴를 통해서다. 패키지나 포트 컬렉션에서도 쉽게 설치할 수 있다.

네트워크에서 GNOME 패키지를 설치하는 것은 다음과 같이 입력하면 된다.

```
# pkg_add -r gnome2
```

소스에서 GNOME 을 빌드 하려면 포트 트리를 사용한다.

```
# cd /usr/ports/x11/gnome2
# make install clean
```

GNOME 이 설치되면 X 서버가 기본 윈도우 매니저 대신 GNOME 을 시작하도록 설정해야 된다. `.xinitrc` 가 있다면 간단히 현재 윈도우 매니저를 시작하는 라인을 `/usr/X11R6/bin/gnome-session` 으로 대체한다. 설정 파일에 특별한 내용이 없다면 다음과 같이 간단히 입력하면 된다:

```
% echo "/usr/X11R6/bin/gnome-session" > ~/.xinitrc
```

그런 다음 `startx` 를 입력하면 GNOME 데스크톱 환경이 시작될 것이다.

**Note:** XDM 같은 디스플레이 매니저가 사용되고 있다면 이 방법은 동작하지 않는다. 대신 위의 명령으로 실행할 수 있는 `.xsession` 파일을 생성한다. 이렇게 하려면 파일을 편집하고 존재하는 윈도우 매니저 명령을 `/usr/X11R6/bin/gnome-session` 로 대신한다:

```
% echo "#!/bin/sh" > ~/.xsession
% echo "/usr/X11R6/bin/gnome-session" >> ~/.xsession
% chmod +x ~/.xsession
```

다른 옵션은 로그인할 때 윈도우 매니저를 선택할 수 있도록 디스플레이 매니저를 설정한다; KDE 관련 섹션에서 KDE 의 디스플레이 매니저인 `kdm` 을 어떻게 설정하는지 자세히 설명한다.

---

### 5.7.1.3 Anti-aliase 폰트와 GNOME

버전 4.0.2 가 시작되어 XFree86 은 anti-aliasing 을 "RENDER" 확장으로 지원한다. Gtk+2.0 과 그 후 버전은(GNOME 이 이용하는 툴킷) 이 기능을 사용할 수 있다. Anti-aliasing 설정 은 5.5.3 장에서 설명하였기 때문에 anti-aliasing 과 업데이트된 소프트웨어는 GNOME 데스크톱과 사용할 수 있다. Applications -> Desktop Preferences -> Font 에서 Best shapes, Best contrast 나 Subpixel smoothing(LCD) 중 하나를 선택한다. GNOME 데스크톱의 일부가 아닌 Gtk+ 어플리케이션은 프로그램을 시작하기 전에 환경변수 GDK\_USE\_XFT를 1로 지정한다.

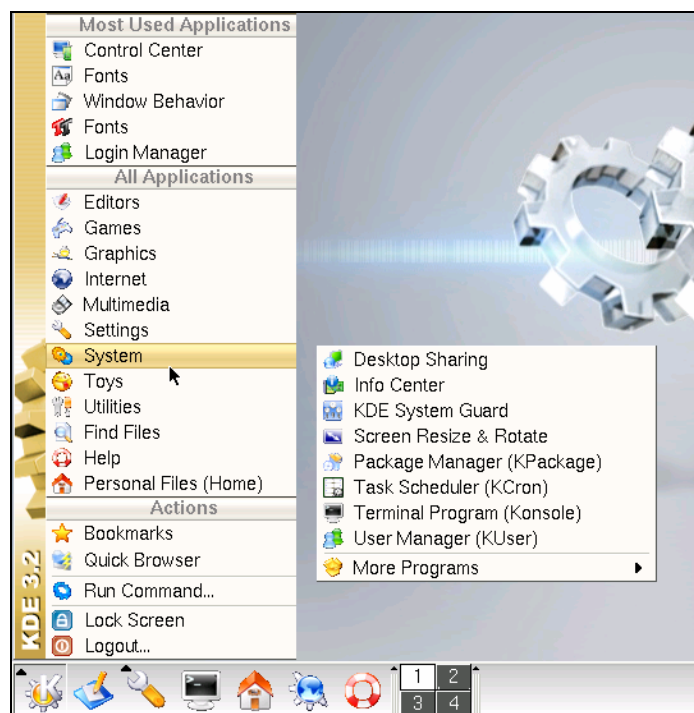
## 5.7.2 KDE

### 5.7.2.1 KDE 에 대하여

KDE 는 사용하기 쉬운 최신 데스크톱 환경이다. KDE 가 유저에게 제공하는 몇 가지는 다음과 같다.

- 아름다운 최신 데스크톱
- 데스크톱은 완벽한 네트워크 투명성을 보여준다.
- KDE 데스크톱과 어플리케이션에서 일관되게 접근할 수 있는 통합된 도움말 시스템.
- 모든 KDE 어플리케이션의 일관된 느낌.
- 표준적인 메뉴와 툴바, 키 결합, 컬러-도표 등.
- 세계화: KDE 는 40 개 이상의 언어로 이용할 수 있다.
- 중심화된 dialog 로 구성된 데스크톱 설정.
- 엄청난 수의 유용한 KDE 어플리케이션.

KDE 는 스프레드 시트, 프레젠테이션, 오거나이저, 뉴스 클라이언트와 더욱 다양한 프로그램으로 이루어진 KDE 의 "KParts" 기술에 기반한 오피스 어플리케이션을 가지고 있다. 또한 KDE 는 유닉스 시스템에있는 다른 웹 브라우저와 경쟁하며 견고함을 의미하는 Konqueror 라는 웹 브라우저를 가지고 있다. KDE 에 관한 더 많은 정보는 KDE 웹 사이트 (<http://www.kde.org/>)에서 찾을 수 있다. KDE 에서 FreeBSD 에 특별한 정보와 리소스는 FreeBSD-KDE 팀(<http://freebsd.kde.org/>)의 웹사이트를 참고한다.



### 5.7.2.2 KDE 설치.

GNOME 이나 다른 데스크톱 환경처럼 KDE 를 설치하는 가장 쉬운 방법은 2.9.12 장에서 설명하였듯이 FreeBSD 설치 과정 중 "Desktop Configuration" 메뉴를 사용하는 것이다. 다시 말하지만 소프트웨어는 패키지나 포트 컬렉션에서 쉽게 설치할 수 있다.

네트워크에서 KDE 패키지를 설치하려면 단순히 다음 명령을 입력한다:

```
# pkg_add -r kde
```

pkg\_add(1)는 자동으로 어플리케이션의 최신 버전으로 패치한다.

소스에서 KDE 빌드는 포트 트리를 사용한다:

---

```
# cd /usr/ports/x11/kde3
# make install clean
```

KDE 가 설치된 후 X 서버에게 기본 윈도우 매니저 대신 이 어플리케이션을 실행하도록 지정해야 된다. 이것은 .xinitrc 파일을 편집하면 된다:

```
% echo "exec startkde" > ~/.xinitrc
```

이제 X 윈도우 시스템이 startx 로 호출될 때 KDE 가 기본 데스크톱이 된다.

xdm 같은 디스플레이 매니저를 사용하였다면 설정이 약간 다르다. 대신 .xsession 파일을 편집한다. kdm 에 관한 설명은 아래에서 소개한다.

### 5.7.3 KDE 관한 상세 설명.

이제 KDE 가 시스템에 설치되었으므로 대부분의 사항을 도움말 페이지나 다양한 메뉴를 클릭하여 찾을 수 있다. 윈도우나 Mac 유저는 상당히 친숙할 것이다.

KDE 를 위한 최고의 레퍼런스는 온라인 문서다. KDE 는 **Konqueror** 라는 웹 브라우저와 다양하고 유용한 어플리케이션 및 폭넓은 문서를 가지고 있다. 이 섹션의 남은 부분에서는 변칙적인 탐구로 배우기 힘든 기술적인 아이템에 대해 설명한다

#### 5.7.3.1 KDE 디스플레이 매니저

멀티 유저 시스템의 관리자는 유저를 환영하기 위한 그래픽 로그인 화면을 원할 것이다. 앞에서 설명한대로 xdm 을 사용할 수 있다. 그러나 KDE 는 xdm 대신 더욱 매력적이고, 로그인할때 더 많은 옵션을 가지고 있는 kdm 을 가지고 있다. 특히 유저는 로그인 후 어떤 데스크톱 환경을(KDE, GNOME 또는 다른 것들) 실행할지 쉽게 선택할 수 있다.

시작은 root 에서 KDE 제어 패널인 kcontrol 을 실행한다. 일반적으로 root 에서 전체 X 환경을 실행하는 것은 불안하다. 대신 일반 유저로 윈도우 매니저를 실행하여 터미널 윈도우를 열고(xterm 이나 KDE 의 konsole 같은) su 로 root 가 된 후(이를 위해 유저는 /etc/group 에서 wheel 그룹이어야 한다) kcontrol 를 입력한다.

---

왼쪽에 System 으로 표시된 곳에서 아이콘을 클릭하고 Login manager 를 클릭한다. 우측에는 KDE 매뉴얼에서 완벽하게 설명하는 다양한 설정 옵션이 있다. 우측의 session 을 클릭한다. 다양한 윈도우 매니저와 데스크톱 환경을 추가하려면 New type 을 클릭한다. 이들은 단지 라벨이기 때문에 **startkde** 나 **gnome-session** 이 아닌 KDE 와 GNOME 이라고 할수 있다. *failsafe* 라벨을 포함한다.

다른 메뉴도 있지만 주로 표면적이고 설명적이다. 끝나면 바닥의 **Apply** 를 클릭하고 제어 센터를 빠져나간다.

**kdm** 이 라벨의(KDE, GNOME 등) 의미를 이해하도록 **xdm** 이 사용하는 파일을 수정한다.

**Note:** KDE 2.2 에서 이 사항은 바뀌었다; **kdm** 은 이제 자신의 설정 파일을 직접 이용한다. 더 자세한 사항은 KDE 2.2 문서를 본다.

root 로 터미널 윈도우에서 /usr/X11R6/lib/X11/xdm/Xsession 파일을 편집한다. 중간부분 섹션은 다음과 같다:

```
case $# in
1)
    case $1 in
    failsafe)
        exec xterm -geometry 80x24-0-0
        ;;
    esac
esac
```

몇 개의 라인을 이 섹션에 추가해야 된다. 사용하고 있는 라벨을 "KDE"와 "GNOME" 이라고 가정하면 다음 내용을 사용한다.

```
case $# in
1)
    case $1 in
    kde)
        exec /usr/local/bin/startkde
        ;;
    GNOME)
```

---

```
        exec /usr/X11R6/bin/gnome-session
        ;;
    failsafe)
        exec xterm -geometry 80x24-0-0
        ;;
    esac
esac
```

KDE 에 로그인할때 아름다운 백그라운드 데스크톱이 나타나도록 다음 라인을 /usr/X11R6/lib/X11/xdm/Xsetup\_0 에 추가한다:

```
/usr/local/bin/kdmdesktop
```

이제 **kdm** 이 다음에 부팅할 때부터 시작되도록 /etc/ttys 에 설정해야 된다. 이렇게 하려면 단순히 xdm 의 이전 섹션에서 지시한 내용을 따르고 /usr/X11R6/bin/xdm 프로그램이 /usr/local/bin/kdm 을 참조하도록 교체한다.

### 5.7.3.2 Anti-aliase 폰트

4.0.2 가 시작되면서 **XFree86** 은 anti-aliasing 을 "RENDER" 확장으로 지원하고 **Qt**(KDE 가 사용하는 툴킷) 버전 2.3 부터 이 확장을 지원한다. 이 설정은 5.5.3 장의 antialiasing X11 폰트에서 설명하였다. 그래서 anti-aliasing 으로 업데이트된 소프트웨어는 **KDE** 데스크톱에서 이용할 수 있다. **KDE** 메뉴에서 **Preferences** -> **Look and Feel** -> **Fonts** 로 이동하여 **Use Anti-Aliasing for Fonts and Icons** 의 체크 박스를 클릭한다. **KDE** 의 일부가 아닌 Qt 어플리케이션은 프로그램을 시작하기 전에 환경변수 **QT\_XFT** 를 *true* 로 지정해야 된다.

## 5.7.4 XFce

### 5.7.4.1 XFce 에 관하여

**XFce** 는 **GNOME** 이 사용하는 **GTK** 툴킷 기반의 데스크톱 환경으로 더 가볍고 단순하지만 사용과 설정이 쉬운 효과적인 데스크톱이다. 보기엔 상용 유닉스 시스템의 **CDE** 와 비슷하다. **XFce** 의 몇 가지 특징은 다음과 같다.

- 간단하면서도 쉬운 데스크톱 관리.

- 
- 마우스의 드래그 & 드롭 등으로 완벽한 설정가능.
  - CDE 와 비슷한 메인 패널, 메뉴, 애플릿과 어플리케이션 실행.
  - 통합 윈도우 매니저, 파일 매니저, 사운드 매니저, **GNOME** 과 호환되는 모듈과 다른 것들.
  - 테마이용(GTK 를 사용하기 때문에).
  - 빠르고, 가볍고 효율적이다: 오래되고/느린 머신이나 메모리 한계가있는 머신에서 이상적이다.

XFce 에 관한 더 많은 정보는 XFce 웹사이트(<http://www.xfce.org/>)에서 찾을 수 있다.

#### 5.7.4.2 XFce 설치

XFce 바이너리 패키지가 있다(이 글을 쓰고 있을 때). 설치는 간단히 다음과 같이 입력한다:

```
# pkg_add -r xfce
```

대신 소스에서 빌드 하려면 포트 컬렉션을 이용한다.

```
# cd /usr/ports/x11-wm/xfce
```

```
# make install clean
```

이제 X 를 시작할때 **XFce** 를 시작하도록 X 서버에게 알려야 한다. 간단히 다음 내용을 입력한다:

```
% echo "/usr/X11R6/bin/startxfce" > ~/.xinitrc
```

X 가 다음에 시작될때 **XFce** 가 기본 데스크톱이 될것이다. 그전에 xdm 같은 디스플레이 매니저를 사용하였다면 **GNOME** 섹션에서 설명하였듯이 **.xsession** 을 생성하지만 `/usr/X11R6/bin/startxfce` 명령을 사용한다; 또는 kdm 섹션에서 설명하였듯이 로그인 할 때 데스크톱을 선택할 수 있도록 디스플레이 매니저를 설정한다.

---

## II 일반적인 업무

이제 기본적인 내용은 설명했고 여기서는 자주 사용되는 FreeBSD 의 기능에 대해 설명한다. 이번 장에서는 다음과 같은 내용을 배울 수 있다.

- X 윈도우 시스템을 더욱 자세히 설명하고 GNOME과 KDE 같은 현대적인 데스크톱 환경을 소개한다.
- FreeBSD에서 사용할 수 있는 다양한 멀티미디어 툴을 소개한다.
- 시스템의 추가적인 기능을 활성화하는 FreeBSD 사용자 커널 빌드 프로세스를 설명한다.
- 데스크톱과 네트워크에 연결되어있는 프린터 설치에 대해 자세히 설명한다.
- FreeBSD 시스템에서 리눅스 어플리케이션을 어떻게 실행하는지 설명한다.

이들 챕터 중 어떤 것은 각 챕터의 시작부분에 미리 읽어야 될 내용을 권장하기도 한다.

## 6 장 데스크톱 어플리케이션

### 6.1 개요

FreeBSD 는 브라우저나 워드 프로세서처럼 방대하고 다양한 데스크톱 어플리케이션을 실행할 수 있다. 이들 중 대부분은 패키지로 사용할 수 있거나 포트 컬렉션에서 자동으로 빌드할 수 있다. 새로운 유저들은 이들 어플리케이션을 자신들의 데스크톱에서 사용할 수 있기를 기대한다. 이번 장에서는 유명한 데스크톱 어플리케이션 몇 가지를 패키지나 포트 컬렉션에서 쉽게 설치하는 것을 보여준다.

포트에서 프로그램을 설치할 때 이들 프로그램은 소스에서 컴파일 된다. 이것은 컴파일 하는 프로그램과 머신의 CPU 에 따라 다소간의 시간을 필요로 한다. 소스에서 빌드 한다면 엄청나게 오랜 시간을 필요로 하기 때문에 이미 빌드 된 패키지인 포트 컬렉션에서 대부분의 프로그램을 설치할 수 있다.



---

FreeBSD 의 리눅스 바이너리 호환 특성으로 리눅스용으로 개발된 많은 어플리케이션을 데스크톱에서 사용할 수 있다. 리눅스 어플리케이션을 설치하기 전에 10 장을 읽어보기를 강력히 권장한다. 리눅스 바이너리 호환성을 사용하는 많은 포트는 "linux-"로 시작된다. 예를들어 whereis(1)로 특정 포트를 검색할 때 이것을 기억한다. 다음 문서는 리눅스 어플리케이션을 설치하기 전에 리눅스 바이너리 호환성을 활성화했다고 가정한다.

다음은 이번 장에서 다룰 내용이다:

- 브라우저 (Mozilla, Netscape, Opera와 같은).
- 업무를 위한 어플리케이션 (koffice, AbiWord, Gimp, OpenOffice.org와 같은).
- 문서 뷰어 (Acrobat Reader, gv, Xpdf, GQview와 같은).
- 재무 회계 (GnuCash, Gnumeric, Abacus와 같은).

이번 장을 읽기 전에 다음과 같은 사항을 알고 있어야 한다:

- 소프트웨어를 어떻게 설치할 수 있는가 (4장)
- 리눅스 소프트웨어를 어떻게 설치할 수 있는가 (10장)

멀티미디어 환경 구축에 대한 정보는 7 장을 읽는다. 전자 메일을 설정하고 사용하려면 22 장을 참고한다.

## 6.2 브라우저

FreeBSD 에는 특정 브라우저가 미리 설치되어 있지 않다. 대신 www 디렉터리(<http://www.FreeBSD.org/ports/www.html>)의 포트 컬렉션에 수많은 브라우저가 설치할 수 있도록 준비되어있다. 모든 것을(어떤 경우는 오랜 시간을 필요로 한다) 컴파일 할 시간이 없다면 대부분을 패키지로 설치할 수 있다.

KDE 와 GNOME 은 HTML 브라우저를 제공하고 있다. 이처럼 완벽한 데스크톱 설정에 대한 더 많은 정보는 5.7 장을 참고한다.

가벼운 브라우저를 찾고 있다면 포트 컬렉션의 [www/dillo](http://www/dillo), [www/links](http://www/links) 나 [www/w3m](http://www/w3m) 을 확인해보도록 한다.

이번 섹션은 이런 어플리케이션에 대해 다룬다:

어플리케이션 이름	필요한 자원	포트에서 설치	주된 의존성
모질라	무겁다	무겁다	Gtk+
넷스케이프	무겁다	가볍다	리눅스 바이너리 호환성
오페라	가볍다	가볍다	FreeBSD 버전: 없다 리눅스 버전: 리눅스 바이너리 호환성과 <code>linux-openmotif</code>

## 6.2.1 모질라

아마 모질라가 FreeBSD 에 가장 적합한 브라우저일 것이다. 현대적이면서 안정적이고 FreeBSD 에 완벽하게 포트 되었다. 모질라의 특징은 매우 표준적이고 완만한 HTML 디스플레이 엔진이다. 모질라는 메일과 뉴스 리더를 지원한다. 웹 페이지를 직접 작성한다면 모질라는 HTML 편집 도구도 제공한다. 넷스케이프 유저는 두 브라우저가 같은 기반을 공유하기 때문에 Communicator 스위트와 비슷함을 느낄 것이다.



CPU 속도가 233MHz 보다 느리거나 64MB 이하의 RAM 을 가진 느린 머신에서는 모질라를

---

완벽하게 사용하기에 너무 많은 자원을 소모할 수 있다. 대신 다음 장에서 설명하는 **오페라** 브라우저를 사용하기를 원할 것이다.

어떤 이유에서 모질라를 컴파일 하기를 원하지 않는다면, FreeBSD GNOME 팀이 미리 여러분을 위해 컴파일 해두었기 때문에 네트워크에서 다음 명령으로 패키지를 설치한다:

```
# pkg_add -r mozilla
```

패키지를 사용할 수 없고 충분한 시간과 디스크 공간을 가지고 있다면 시스템에 모질라를 컴파일하고 설치할 수 있는 소스를 받을 수 있다. 다음 명령을 수행한다:

```
# cd /usr/ports/www/mozilla
# make install clean
```

**모질라** 포트는 root 권한으로 chrome 등록 설정을 실행하여 정확히 초기화할 수 있다. 그렇지만 마우스 동작과 같은 몇 가지 추가 패치를 원한다면 root 로 **모질라**를 실행하여 정확히 설치해야 된다.

**모질라** 설치가 끝나면 root 권한은 더 이상 필요없다. **모질라** 브라우저를 다음 명령으로 시작할 수 있다:

```
% mozilla
```

아래 명령처럼 메일과 뉴스리더를 직접 시작할 수 있다:

```
% mozilla -mail
```

## 6.2.2 Mozilla, Java™, and Macromedia® Flash™

**모질라** 설치는 간단하지만 불행히 자바와 마이크로미디어 플레시 같은 것을 지원하도록 **모질라**를 설치하는 것은 시간과 디스크 공간을 필요로 한다.

첫째로 모질라와 같이 사용할 파일을 다운로드 한다. 현재 웹 브라우저에 <http://www.sun.com/software/java2/download.html>을 입력하고 이 웹 사이트의 계정을 생성한다. 다음에도 필요할 수 있기 때문에 여기서 유저이름과 패스워드를 저장하도록 한다.

---

라이선스 제한 때문에 자동으로 포트를 패치 할 수 없기 때문에 파일 j2sdk-1\_3\_1-src.tar.gz를 다운로드하고 /usr/ports/distfiles/에 복사한다. 또한 우리는 <http://java.sun.com/webapps/download/Display?BundleId=7905> 에서 " java environment "을 다운로드 해야 된다. 파일 이름은 j2sdk-1\_3\_1\_08-linux-i586.bin이고 크기가 크다(대략 25MB). 아까처럼 이 파일도 /usr/ports/distfiles/에 넣어 둔다. 마지막으로 " java patchkit"을 <http://www.eyesbeyond.com/freebsd/ports/java/>에서 다운로드하고 /usr/ports/distfiles/에 복사한다.

[java/jdk13](#) 포트를 표준 **make install clean** 으로 설치하고 [www/flashpluginwrapper](#) 포트도 설치한다. 이 포트는 커다란 [emulators/linux\\_base](#) 를 필요로 한다. 다른 **플레시** 플러그인이 있지만 동작하지 않는다.

**모질라**를 설치하지 않았다면 [www/mozilla](#) 포트를 설치한다.

이제 플레시 플러그인 파일을 다음 명령으로 복사한다:

```
# cp /usr/local/lib/flash/libflashplayer.so ₩
    /usr/X11R6/lib/browser_plugins/libflashplayer_linux.so

# cp /usr/local/lib/flash/ShockwaveFlash.class ₩
    /usr/X11R6/lib/browser_plugins/
```

이제 다음 라인을 **모질라** 시작 스크립트 /usr/X11R6/bin/mozilla 의 가장 상단에(#!/bin/sh 바로 밑에) 추가한다:

```
LD_PRELOAD=/usr/local/lib/libflashplayer.so.1
export LD_PRELOAD
```

이것은 **플레시** 플러그인을 활성화한다.

이제 다음 명령으로 **모질라**를 시작하면 된다:

```
% mozilla &
```

그리고 **Help** 메뉴에서 **About Plug-ins** 옵션으로 간다. 이 리스트는 현재 사용할 수 있는 모든 플러그인을 보여준다. **자바**와 **마이크로미디어 플레시**가 리스트 되어 있을 것이다.

---

## 6.2.3 넷스케이프

포트 컬렉션에는 넷스케이프 브라우저의 몇 가지 버전이 있다. FreeBSD 에 기본적인 버전이 한때 심각한 보안 버그를 가지고 있었기 때문에 이들을 설치하면 크게 실망하게 된다. 대신 좀더 최근의 리눅스나 디지털 유닉스 버전을 사용한다.

요즘 넷스케이프 브라우저의 가장 안정된 릴리즈는 **넷스케이프 7**이다. 이것은 포트 컬렉션으로 설치할 수 있다:

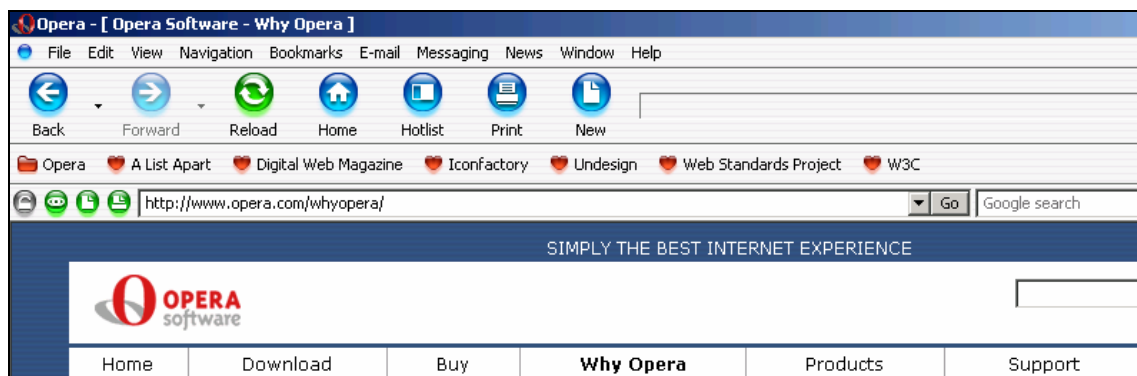
```
# cd /usr/ports/www/netscape7
# make install clean
```

프랑스, 독일과 일본 카테고리에 지역화된 버전이 있다.

**주의:** 넷스케이프 4.X 버전은 현대 표준에 맞지 않기 때문에 권장하지 않는다. 그러나 넷스케이프 7.x 와 새로운 버전은 i386 플랫폼에서만 사용할 수 있다.

## 6.2.4 오페라

오페라는 매우 빠르다는 강점을 가진 표준 호환 브라우저다. 오페라에는 두 가지 장점이 있다: 일반적인 FreeBSD 버전과 리눅스 에뮬레이터에서 운용되는 버전이 있다. 광고가 화면에 나타나는 무료 브라우저 버전과 오페라 웹사이트에서(<http://www.opera.com/>) 구입할 수 있는 버전이 있다.



FreeBSD 버전 **오페라**로 웹을 탐색하려면 패키지를 설치한다:

---

```
# pkg_add -r opera
```

어떤 FTP 사이트는 모든 패키지를 가지고 있지 않지만 다음 명령으로 포트 컬렉션에서 받을 수 있다:

```
# cd /usr/ports/www/opera  
# make install clean
```

리눅스 버전의 **오페라**를 설치하려면 위의 예제 opera 에 linux-opera 를 사용한다. 리눅스 버전은 **어도비 아크로벳 리더**처럼 리눅스에서만 사용할 수 있는 플러그인이 필요한 상황에서 유용하다. 모든 면에서 FreeBSD 와 리눅스 버전은 기능적으로 동일하다.

## 6.3 업무용 어플리케이션

업무용 어플리케이션에서 새로운 유저는 종종 관찮은 오피스 툴이나 친근한 워드 프로세서를 찾는다. KDE 같은 데스크톱 환경이 오피스 툴을 지원하고 있지만 기본 어플리케이션은 없다. FreeBSD 는 데스크톱 환경에 상관없이 필요한 모든 것을 제공한다.

이 섹션은 이러한 어플리케이션에 대해 다룬다:

어플리케이션 이름	필요한 자원	포트에서 설치	주된 의존성
KOffice	가볍다	무겁다	KDE
AbiWord	가볍다	가볍다	Gtk+또는 GNOME
Gimp	가볍다	무겁다	GTK+
OpenOffice.org	무겁다	엄청나다	GCC3.1, JDK1.3, Mozilla

### 6.3.1 KOffice

KDE 공동체는 데스크톱 환경과 KDE 에서 사용할 수 있는 오피스 툴을 제공한다. 이 오피스 툴은 다른 오피스 툴에서 찾을 수 있는 4 개의 표준 컴포넌트를 포함한다. KWord 는 워드프로세서, KSpread 는 스프레드시트 프로그램, KPresenter 는 프레젠테이션 그리고 Kontour 로 그래픽 문서를 만들 수 있다.

최신 KOffice 를 설치하기 전에 KDE 버전을 업데이트 해야 된다.

---

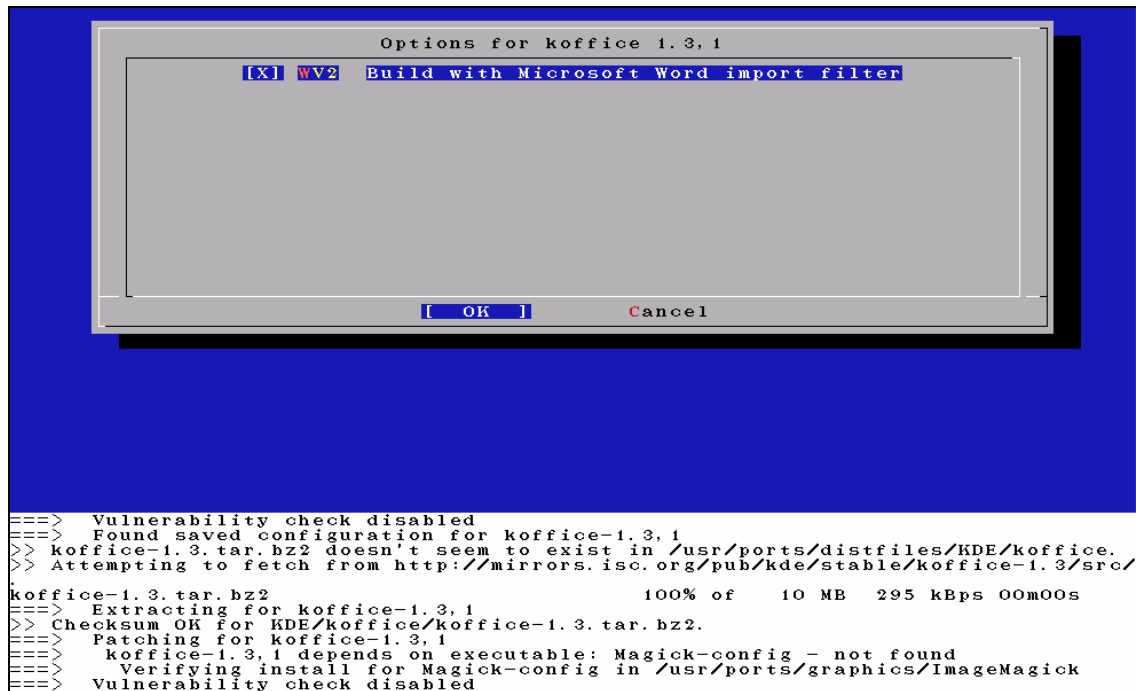
패키지에서 **KOffice** 를 설치하려면 다음 명령을 사용한다:

```
# pkg_add -r koffice
```

패키지를 사용할 수 없다면 포트 컬렉션을 사용할 수 있다. 예를 들어 KDE3 용 **KOffice** 를 설치하려면 다음 명령을 사용한다:

```
# cd /usr/ports/editors/koffice-kde3
# make install clean
```

위 명령을 실행하면 **Microsoft Word import** 필터를 설치할 것인지 문의한다.



### 6.3.2 AbiWord

**AbiWord** 는 마이크로소프트 워드와 비슷한 무료 워드 프로세서 프로그램이다. 이것은 편지, 레포트, 메모 같은 것을 입력할때 편리하다. 또한 매우 빠르고 많은 특성을 가지고 있으며 사용자 친화적이다. **AbiWord** 는 마이크로 소프트웨어의 .doc 포맷을 포함하여 수많은 파일 포맷으로 불러오거나 내보낼 수 있다. **AbiWord** 는 패키지에서 사용할 수 있기 때문에 다음 명령으로 설치할 수 있다:

---

```
# pkg_add -r AbiWord-gnome
```

패키지를 사용할 수 없다면 포트 컬렉션에서 컴파일 할 수 있다. 포트 컬렉션은 업데이트 되어 있어야 한다. 다음 명령으로 설치할 수 있다:

```
# cd /usr/ports/editors/AbiWord
# make install clean
```

### 6.3.3 김프

이미지를 만들거나 사진을 편집하는 김프는 매우 세련된 이미지 편집 프로그램이다. 김프는 간단한 페인트 프로그램이지만 고 품질의 사진 편집 툴로 사용되고 있다. 다양한 플러그인과 스크립트 인터페이스 기능을 지원한다. 김프는 폭 넓은 범위의 파일 포맷을 읽고 쓸수 있다. 스캐너와 타블렛 인터페이스를 지원한다.

다음 명령으로 패키지에서 설치할 수 있다:

```
# pkg_add -r gimp
```

FTP 사이트가 이 패키지를 가지고 있지 않다면 포트 컬렉션을 사용할 수 있다. 포트 컬렉션의 graphics 디렉터리는(<http://www.FreeBSD.org/ports/graphics.html>) 김프 매뉴얼도 가지고 있다. 여기서 김프를 어떻게 설치하는지 보여준다:

```
# cd /usr/ports/graphics/gimp1
# make install clean
# cd /usr/ports/graphics/gimp-manual-pdf
# make install clean
```

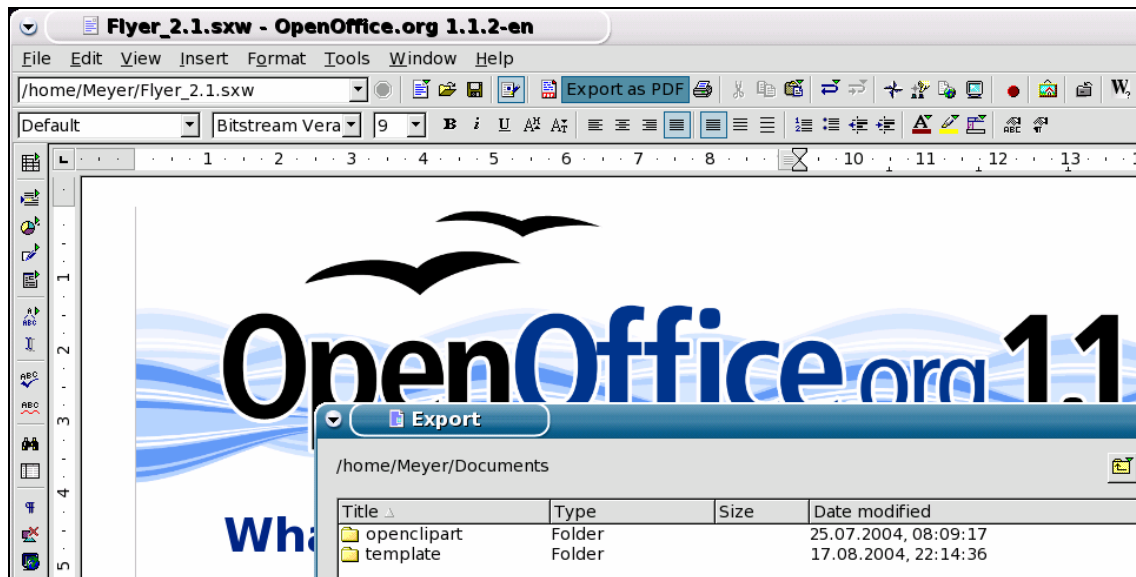
**Note:** 포트 컬렉션의 graphics(<http://www.FreeBSD.org/ports/graphics.html>) 디렉터리는 개발 버전의 김프를 [graphics/gimp-devel](#)에 가지고 있다. HTML 과 포스트스크립트 버전의 김프 매뉴얼은 [graphics/gimp-manual-html](#) 과 [graphics/gimp-manual-ps](#)에 있다.



---

### 6.3.4 오픈 오피스

OpenOffice.org 는 완벽한 오피스 제품에 필수적인 모든 어플리케이션을 가지고 있다: 워드 프로세서, 스프레드시트, 프레젠테이션 관리자와 드로잉 프로그램. 유저 인터페이스는 다른 오피스 툴들과 매우 비슷하며 다양하고 유명한 파일 포맷을 불러오고 내보낼 수 있다. 인터페이스, 스펠 체크와 사전을 포함한 많은 것을 다른 언어로 사용할 수 있다.



OpenOffice.org의 워드 프로세서는 호환성과 유연성을 증가시키기 위해 XML 파일 포맷을 사용한다. 스프레드시트 프로그램은 매크로 언어와 외부 데이터베이스에 연결할 수 있는 특징을 가지고 있다. OpenOffice.org는 이미 윈도우, 솔라리스, 리눅스, FreeBSD와 Mac OS X 에서 안정되고 자연스럽게 실행된다. OpenOffice.org에 대한 더 많은 정보는 오픈 오피스 웹사이트에서(<http://www.openoffice.org/>) 찾을 수 있다. FreeBSD에 맞는 정보와 직접 패키지를 다운로드 하려면 FreeBSD OpenOffice Porting Team의 웹 사이트를 방문한다 (<http://projects.imp.ch/openoffice/>).

OpenOffice.org 를 설치하려면 다음 명령을 입력한다:

```
# pkg_add -r openoffice
```

패키지가 설치되면 setup 프로그램을 실행하고 *standard workstation installation* 을 선택한다. 이 명령은 OpenOffice.org 를 사용하려는 유저에서 실행한다:

```
% openoffice-setup
```

---

오픈 오피스 패키지를 사용할 수 없다면 포트에서 컴파일 할수 있다. 그렇지만 많은 디스크 공간과 컴파일 시간이 소요되는 점을 감안해야 된다.

```
# cd /usr/ports/editors/openoffice
# make install clean
```

이것이 끝나면 **OpenOffice.org** 를 사용하려는 유저에서 다음 명령으로 `setup` 를 실행하고 *standard workstation installation* 을 선택한다:

```
% cd /usr/ports/editors/openoffice
% make install-user
```

지역화된 버전을 사용하려면 사용할 수 있는 포트가 있다:

언어	포트
아라비아어	editors/openoffice-ar
덴마크어	editors/openoffice-dk
스페인어	editors/openoffice-es
그리스어	editors/openoffice-gr
이탈리아어	editors/openoffice-it
네덜란드어	editors/openoffice-nl
폴란드어	editors/openoffice-pl
스위스어	editors/openoffice-se
터키어	editors/openoffice-tr
프랑스어	french/openoffice
독일어	german/openoffice
일본어	japanese/openoffice
한국어	korean/openoffice
포르투갈어	portuguese/openoffice
러시아어	russian/openoffice

## 6.4 문서 뷰어

---

새로운 몇 가지 문서 포맷이 최근에 인기를 얻었다. 기본 시스템에서 사용할 수 없는 표준 뷰어도 필요할 것이다. 이번 섹션에서는 이들 뷰어를 어떻게 설치하는지 보여준다.

어플리케이션 이름	필요한 자원	포트에서 설치	주된 의존성
아크로벳리더	가볍다	가볍다	리눅스 바이너리 호환성
gv	가볍다	가볍다	Xaw3d
Xpdf	가볍다	가볍다	FreeType
GQview	가볍다	가볍다	Gtk+ 또는 GNOME

### 6.4.1 아크로벳 리더

많은 문서는 이제 배포하기 편리한 문서 포맷의 표준인 PDF 파일로 배포된다. 이런 종류의 파일을 보기 위해 권장되는 뷰어는 어도비사가 리눅스용으로 릴리즈한 **아크로벳 리더**다. FreeBSD 는 리눅스 바이너리를 실행할 수 있기 때문에 FreeBSD 용으로 사용할 수 있다.

아크로벳 리더 5 패키지를 설치하려면 다음 명령을 실행한다

```
# pkg_add -r acroread5
```

일반적인 패키지를 이용할 수 없거나 최신버전을 원한다면 포트 컬렉션을 이용할 수 있다:

```
# cd /usr/ports/print/acroread5  
# make install clean
```

**Note:** 아크로벳 리더는 여러가지 다른 버전을 사용할 수 있다. 이 문서를 작성하는 동안: [print/acroread](#)(버전 3.0.2), [print/acroread4](#) (버전 4.0.5)와 [print/acroread5](#) (버전 5.0.6)이 있다. 이들 모두가 여러분이 가지고있는 FreeBSD 버전에 맞도록 패키지되지 않았을 수도 있다. 포트 컬렉션은 항상 최신 버전을 가지고 있다.

### 6.4.2 gv

gv 는 포스트스크립트와 PDF 뷰어다. 이것은 원래 **ghostview** 기반이었으나 보기 좋은 Xaw3d 라이브러리를 갖게 되었다. **gv** 는 빠르고 인터페이스는 깨끗하다. **gv** 는 적응하기 쉽고, 종이크기, scale 또는 antialias 와 같은 많은 특징을 가지고 있다. 거의 모든 동작은

---

키보드나 마우스로 할 수 있다.

패키지에서 `gv` 를 설치하려면 다음 명령을 사용한다:

```
# pkg_add -r gv
```

패키지를 가지고 있지 않다면 포트 컬렉션을 이용할 수 있다:

```
# cd /usr/ports/print/gv
```

```
# make install clean
```

### 6.4.3 Xpdf

작은 FreeBSD PDF 뷰어를 원한다면 **Xpdf** 가 가볍고 효율적인 뷰어다. 이것은 매우 적은 자원을 필요로하고 매우 안정적이다. 표준 X 폰트를 사용하기 때문에 **Motif** 나 다른 X 툴킷이 필요없다.

**Xpdf** 패키지를 설치하려면 다음 명령을 사용한다:

```
# pkg_add -r xpdf
```

패키지를 사용할 수 없거나 포트 컬렉션을 사용한다면 다음 명령을 사용한다:

```
# cd /usr/ports/graphics/xpdf
```

```
# make install clean
```

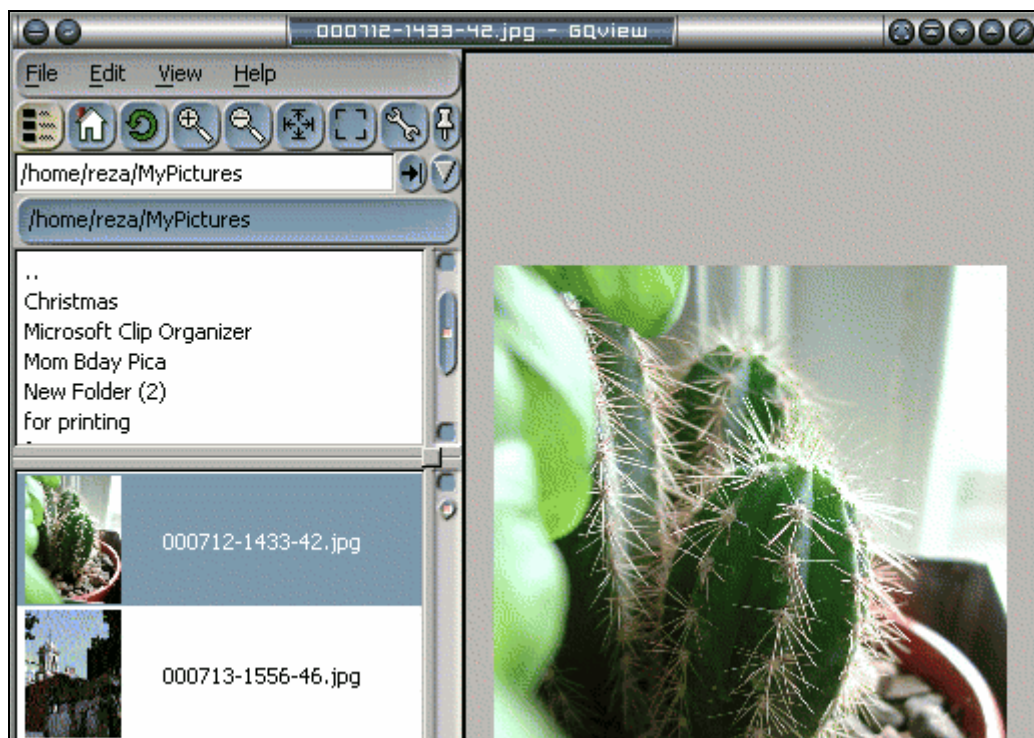
설치가 끝나면 마우스 오른쪽 버튼으로 메뉴를 활성화시켜서 **Xpdf** 를 시작하고 사용할 수 있다.

### 6.4.4 GQview

**GQview** 는 이미지 관리기다. 한번 클릭으로 파일을 볼수 있으며 외부 에디터 실행과 썸네일 미리 보기 등을 할수 있다. 슬라이드 쇼 모드와 몇 가지 기본 파일 운용 기능이 있다. 이미지 컬렉션을 관리하여 중복된 것을 쉽게 찾을 수 있다. **GQview** 는 전체 화면

---

보기와 국제화를 지원할 수 있다.



GQview 패키지를 설치하려면 다음 명령을 입력한다:

```
# pkg_add -r gqview
```

패키지를 사용할 수 없거나 포트 컬렉션을 사용한다면 다음 명령을 입력한다:

```
# cd /usr/ports/graphics/gqview  
# make install clean
```

## 6.5 재무 회계

어떤 이유에서든 FreeBSD 데스크톱에서 개인적인 재정 관리를 원한다면 강력하고 사용하기 쉬운 몇 가지 어플리케이션이 설치될 준비가 되어있다. 이들 중 몇가지는 **Quicken** 이나 **Excel** 문서처럼 광범위한 파일 포맷과 호환된다.

이번 섹션은 이런 어플리케이션에 대해 다룬다.

---

어플리케이션 이름	필요한 자원	포트에서 설치	주된 의존성
GnuCash	가볍다	무겁다	GNOME
Gnumeric	가볍다	무겁다	GNOME
Abacus	가볍다	가볍다	Tcl/Tk

## 6.5.1 GnuCash

GnuCash 는 사용하기에 쉽지는 않지만 사용자들에게 강력한 어플리케이션을 제공하고자 하는 GNOME 프로젝트 노력의 일부분이다. GnuCash 에서 수입과 지출, 은행 구좌 또는 주식 내역을 관리할 수 있다. GnuCash 는 직관적이지만 매우 전문적이다.

GnuCash 는 훌륭한 등록기, 계층적인 계좌 시스템, 많은 키보드 가속기와 자동 완성기능을 제공한다. 이것은 하나의 트랜잭션을 여러개로 더 자세히 나눌 수 있다. GnuCash 는 Quicken QIF 파일을 가져오거나 융합할 수 있다. 또한 거의 모든 나라의 날짜와 통화 포맷을 제어한다.

GnuCash 를 시스템에 설치하려면 다음 명령을 실행한다:

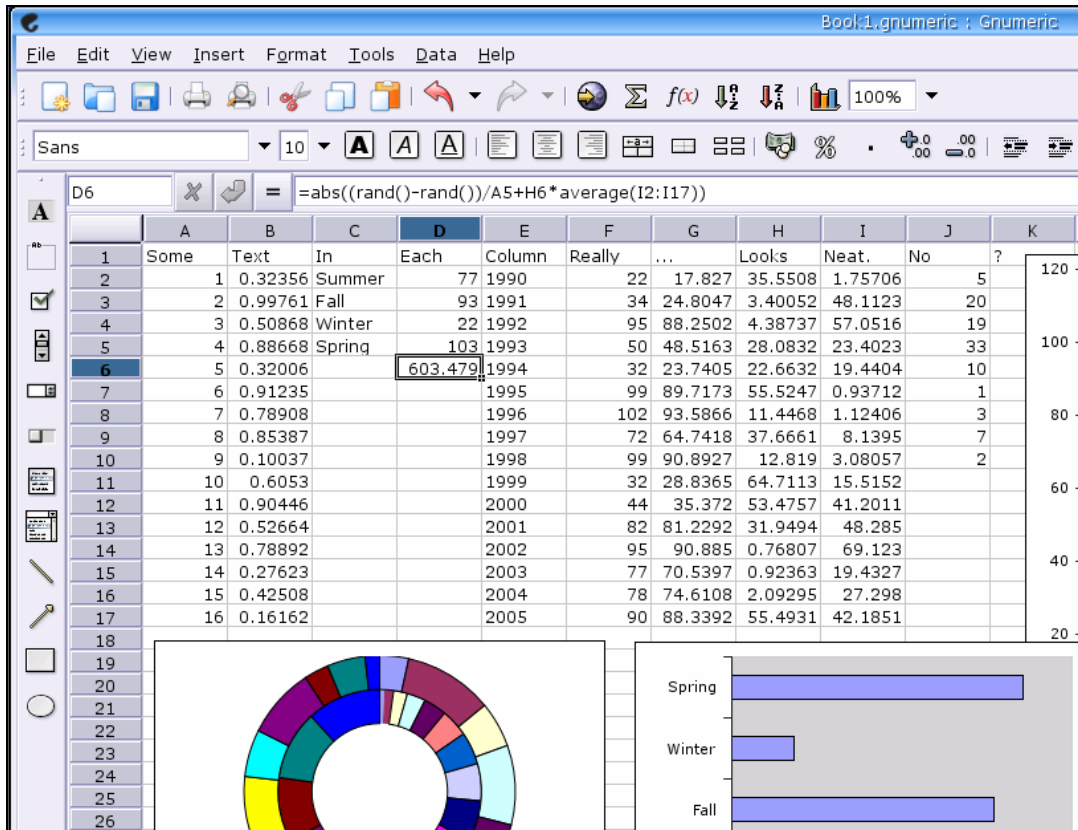
```
# pkg_add -r gnuCash
```

패키지를 사용할 수 없다면 포트 컬렉션을 이용할 수 있다:

```
# cd /usr/ports/finance/gnuCash
# make install clean
```

## 6.5.2 Gnumeric

Gnumeric 는 GNOME 데스크톱 환경의 일부인 스프레드시트다. 이 스프레드시트는 사용하기 편하게 셀 포맷에 따라 자동으로 유저의 입력을 "추측"하고 여러가지 순서에 맞춰 자동으로 채워 넣는 시스템을 가지고 있다. 이것은 Excel, Lotus 1-2-3 또는 Quattro Pro 와 같은 유명한 많은 포맷 파일을 불러올 수 있다. Gnumeric 는 [math/guppi](#) 그래픽 프로그램으로 그래픽을 지원한다. 또한 숫자, 통화, 날짜, 시간 등의 다양한 내장 함수로 일반적인 모든 셀 포맷을 사용할 수 있다.



Gnumeric 를 패키지에서 설치하려면 다음을 입력한다:

```
# pkg_add -r gnumeric
```

패키지를 사용할 수 없다면 다음 명령으로 포트 컬렉션을 이용할 수 있다:

```
# cd /usr/ports/math/gnumeric
# make install clean
```

### 6.5.3 Abacus

**Abacus** 는 작고 사용하기 쉬운 스프레드시트다. 이것은 통계, 제정 그리고 수학 같은 다양한 범위의 유용하고 수많은 내장 함수를 포함하고 있다. 또한 **Excel** 파일 포맷도 불러오고 내보낼 수 있다. **Abacus** 는 포스트스크립트 형식으로 생성할 수 있다.

패키지에서 **Abacus** 를 설치하려면 다음 명령을 입력한다:

---

```
# pkg_add -r abacus
```

패키지를 사용할 수 없다면 다음 명령으로 포트 컬렉션을 이용할 수 있다:

```
# cd /usr/ports/dekutils/abacus
# make install clean
```

## 6.6 요약

FreeBSD가 ISP사이에서 성능과 안정성으로 유명하지만 데스크톱으로 사용할 수 있도록 충분한 준비가 되어있다. 수천만의 어플리케이션을 패키지 (<http://www.FreeBSD.org/where.html>)나 포트에서 (<http://www.FreeBSD.org/ports/index.html>) 사용할 수 있기 때문에 필요한 모든 것으로 완벽한 데스크톱을 만들 수 있다.

데스크톱 설치를 끝냈다면 `misc/instant-workstation`으로 더 많은 것을 원할 것이다. 이 "메타-포트"로 워크스테이션을 위한 전형적인 포트 세트를 빌드할 수 있다. 포트 세트는 `/usr/ports/cisc/instant-workstation/Makefile`을 편집하여 수정할 수 있다. 다음 구문은 포트 추가 또는 삭제를 위한 기본 세트로 사용되기 때문에 일반적인 절차로 빌드한다. 결국 원하는 데스크톱에 맞는 매우 적합하고 커다란 패키지를 생성해서 다른 워크스테이션에 설치할 수 있다.

다음은 이번 장에서 다룬 모든 데스크톱 어플리케이션에 대해 간단히 요약한 내용이다:

어플리케이션 이름	패키지 이름	포트 이름
Mozilla	<i>mozilla</i>	www/mozilla
Netscape	<i>linux-netscape7</i>	www/netscape7
Opera	<i>linux-opera</i>	www/linux-opera
KOffice	<i>koffice-kde3</i>	editors/koffice-kde3
AbiWord	<i>AbiWord-gnome</i>	editors/AbiWord
The GIMP	<i>gimp</i>	graphics/gimp1
OpenOffice.org	<i>openoffice</i>	editors/openoffice
Acrobat Reader	<i>acroread5</i>	print/acroread5



---

<b>gv</b>	<i>gv</i>	print/gv
<b>Xpdf</b>	<i>xpdf</i>	graphics/xpdf
<b>GQview</b>	<i>gqview</i>	graphics/gqview
<b>GnuCash</b>	<i>gnucash</i>	finance/gnucash
<b>Gnumeric</b>	<i>gnumeric</i>	math/gnumeric
<b>Abacus</b>	<i>abacus</i>	deskutils/abacus

---

## 7 장 멀티미디어

### 7.1 개요

FreeBSD 는 광범위하고 다양한 사운드 카드를 지원하여 컴퓨터에서 고음질의 출력으로 여러분을 즐겁게 할수 있다. FreeBSD 는 MPEG 오디오 레이어 3(MP3), WAV 그리고 Ogg Vorbis 포맷과 다른 수 많은 포맷으로도 오디오를 레코드하고 플레이할 수 있는 기능을 가지고 있다. 또한 FreeBSD 포트 컬렉션은 레코드된 오디오 편집과 사운드 효과 추가 그리고 MIDI 장치를 제어하는 다양한 어플리케이션을 가지고 있다.

개인적인 경험에 의하면 FreeBSD 는 비디오와 DVD 파일을 플레이 하도록 지원한다. 다양한 비디오 미디어를 인코딩하고 변환하여 플레이하는 어플리케이션은 사운드 어플리케이션보다 적다. 예를들어 이 글을 쓰는 동안 FreeBSD 포트 컬렉션의 [audio/sox](#)에는 포맷간에 변환할 수 있는 쓸만한 재 인코딩 어플리케이션이 없다. 그렇지만 이 부분의 소프트웨어에 대한 전망은 급속히 좋아지고 있다.

이번 장에서는 사운드 카드를 설정하기 위해 필요한 단계를 설명한다. **XFree86**(5 장)의 설정과 설치는 더 좋은 플레이를 위해 약간의 튜닝이 필요하겠지만 여러분의 비디오 카드에 맞는 주의점에 대해 다루었다.

이번 장을 읽고 다음과 같은 사항을 알 수 있다:

- 시스템이 사운드 카드를 감지하도록 어떻게 설정하는가.
- 샘플 어플리케이션을 사용하여 카드를 테스트 한다.
- 사운드 카드 설정 문제를 어떻게 해결하는가.
- MP3와 다른 오디오를 어떻게 플레이하고 인코딩하는가.
- **XFree86**으로 비디오 카드를 어떻게 지원하는가.
- 쓸만한 비디오 플레이어/인코더 포트
- DVD, .mpg 그리고 .avi 파일은 어떻게 플레이 하는가.

- 
- CD와 DVD 정보를 파일로 어떻게 바꾸는가.

이번 장을 읽기 전에 다음 사항을 알고 있어야 된다:

- 새로운 커널을 어떻게 설정하고 설치하는지 알아야 한다 (8장)

비디오 섹션은 **XFree86 4.X** 가(x11/XFree86-4) 설치되었다고 가정한다. **XFree86 3.X** 도 동작하겠지만 이번 장에서 설명하는 내용으로 테스트하지 않았다. 여기서 설명하는 내용이 **XFree86 3.X** 에서 동작한다면 우리에게 알려주기 바란다.

**주의:** 오디오 CD 나 비디오 DVD 를 mount(8) 명령으로 마운트하면 최소한 에러가 발생하고 운이없는 경우 커널 패닉이 발생한다. 이런 미디어는 보통 ISO-파일시스템과 다른 특별한 인코딩이되어 있다.

## 7.2 사운드 카드 설정

### 7.2.1 정확한 장치 찾기

시작하기 전에 가지고있는 카드의 모델, 사용하는 칩 그리고 PCI/ISA 카드인지 알고 있어야 한다. FreeBSD 는 광범위하고 다양한 PCI 와 ISA 카드를 지원하기 때문에 다음 리스트에서 여러분의 카드를 보지 못했다면 pcm(4) 매뉴얼 페이지를 체크한다. 다음 리스트는 아니지만 아주 일반적인 카드 리스트다.

- Crystal 4237, 4236, 4232, 4231
- Yamaha OPL-Sax
- OPTi931
- Ensoniq AudioPCI 1370/1371
- ESS Solo-1/1E
- NeoMagic 256AV/ZX
- Sound Blaster Pro, 16, 32, AWE64, AWE128, Live
- Creative ViBRA16
- Advanced Asound 100, 110, and Logic ALS120

- 
- ES 1868, 1869, 1879, 1888
  - Gravis UltraSound
  - Aureal Vortex 1 or 2

사운드 장치를 사용하려면 적당한 장치 드라이버를 로드해야 된다. 드라이버 로드는 두 가지 방법 중 한가지로 가능하다. 가장 쉬운 방법은 단순히 사운드 카드의 커널 모듈을 `kldload(8)`로 로드하는 것이다. 그렇지 않으면 커널이 사운드 카드를 지원하도록 정적으로 컴파일 한다. 아래 섹션에서는 이처럼 하드웨어 지원을 추가하는데 필요한 정보를 제공한다. 커널 재 컴파일에 대한 더 많은 정보는 8장을 본다.

### 7.2.1.1 Creative, Advance 와 ESS 사운드 카드

위의 카드 중 한 가지를 가지고 있다면 다음 내용을 커널 설정 파일에 추가해야 된다:

#### **device pcm**

PnP ISA 카드를 가지고 있다면 다음을 추가해야 된다:

#### **device sbc**

PnP ISA 카드가 아닌 경우 다음을 커널 설정 파일에 추가한다:

#### **device pcm**

**device sbc0 at isa? port 0x220 irq 5 drq 1 flags 0x15**

위에서 보여준 설정은 기본값이다. 가지고 있는 카드에 맞게 IRQ 나 설정을 변경해야 된다. 더 많은 정보는 `sbc(4)` 매뉴얼 페이지를 본다.

**Note:** 사운드 블레스터 라이브는 이 섹션에서 다루지않는 패치 없이는 FreeBSD 4.0 에서 지원되지 않는다. 이 카드를 사용하기 전에 최신 STABLE 로 업데이트 할 것을 권장한다.

### 7.2.1.2 Gravis 울트라 사운드 카드

---

PnP ISA 카드는 다음 내용을 커널 설정 파일에 추가해야 된다:

**device pcm**

**device gusc**

PnP ISA 카드가 아니라면 다음 내용을 커널 설정 파일에 추가한다:

**device pcm**

**device gus0 at isa? port 0x220 irq 5 drq 1 flags 0x13**

가지고있는 카드에 맞도록 IRQ 나 다른 설정을 변경해야 된다. 더 많은 정보는 gusc(4) 매뉴얼 페이지를 참고한다.

### 7.2.1.3 크리스털 사운드 카드

크리스털 사운드 카드는 다음 내용을 커널 설정파일에 추가해야 된다:

**device pcm**

**device csa**

### 7.2.1.4 일반적인 지원

PnP ISA 나 PCI 카드는 다음 내용을 커널 설정 파일에 추가해야 된다:

**device pcm**

브리지 드라이버를 가지고 있지 않은 PnP ISA 사운드 카드가 아닌 경우 다음 내용을 커널 설정 파일에 추가해야 된다:

**device pcm0 at isa? irq 10 drq 1 flags 0x0**

가지고 있는 카드에 맞게 IRQ 나 다른 설정을 변경한다.

---

## 7.2.1.5 Onboard 사운드 카드

마더보드에 사운드 장치가 부착되어있는 시스템은 커널 설정에 다음 옵션이 필요할 것이다:

### options PNPBIOS

**Note:** FreeBSD 5.0 이나 이 후 버전을 사용 중 이면 *PNPBIOS* 옵션은 필요없다. 이 옵션 삭제되었지만 기능은 항상 활성화 되어있다.

## 7.2.2 장치 노드 생성 및 테스트

제 부팅하고 로그인 후 `/var/run/dmesg.boot` 파일에서 다음과 같이 장치를 체크한다:

```
# grep pcm /var/run/dmesg.boot
```

```
pcm0: <SB16 DSP 4.11> on sbc0
```

여러분 시스템의 결과는 다를 것이다. `pcm` 장치가 보이지 않는다면 무엇인가 잘못되어있을 것이다. 이런 일이 발생한다면 커널 설정 파일에서 정확한 장치를 선택했는지 다시 확인한다. 일반적인 문제는 7.2.2.1 장에 설명한다.

**Note:** FreeBSD 5.0 이나 새로운 버전을 사용한다면 이 섹션의 남은 부분을 지나쳐도 된다. 이들 5.0 버전은 `devfs(5)`를 사용하여 자동으로 장치 노드를 생성한다.

이전 명령이 `pcm0` 를 보여준다면 `root` 에서 다음 명령을 실행해야 된다:

```
# cd /dev
```

```
# sh MAKEDEV snd0
```

이전 명령이 `pcm1` 을 보여준다면 `snd0` 를 `snd1` 로 바꾸어서 위에서 보여준대로 실행한다.

**Note:** 위의 명령은 `/dev/snd` 장치를 생성하지 않는다!

MAKEDEV 는 다음 사항을 포함하는 장치 노드 그룹을 생성한다:

장치	설명
----	----

/dev/audio	SPARC-호환 오디오 장치.
/dev/dsp	디지털 음성 장치.
/dev/dspW	/dev/dsp 와 같지만 샘플당 16 bits 가 아니다.
/dev/midi	Raw midi 접근 장치.
/dev/mixer	포트 믹서 장치 제어.
/dev/music	레벨 2 sequencer 인터페이스.
/dev/sequencer	Sequencer 장치.
/dev/pss	프로그램 가능한 장치 인터페이스.

모든 것이 정상이라면 사운드 카드 기능을 사용할 수 있다. CD-ROM 이나 DVD-ROM 드라이브와 사운드 카드가 정확히 설정되었다면 CD 를 드라이브에 넣고 cdcontrol(1)로 플레이 할수 있다:

```
% cdcontrol -f /dev/acd0c play 1
```

audio/workman 처럼 다양한 어플리케이션이 더 좋은 인터페이스를 제공한다. MP3 오디오 파일을 듣기 위해 audio/mpg123 같은 어플리케이션을 설치할 수 있다.

### 7.2.2.1 일반적인 문제

에러	해결책
unsupported subdevice XX	하나 이상의 장치 노드가 정확하게 생성되지 않았다. 위의 단계를 반복한다.
sb_dspwr(XX) timed out	I/O 포트가 정확히 설정되지 않았다.
bad irq XX	IRQ 가 정확하게 설정되지 않았다. 사운드 카드의 IRQ 와 맞게 IRQ 를 설정한다.
xxx: gus pcm not attached, out of memory	장치를 사용하기 위한 충분한 메모리가 없다.
xxx: can't open /dev/dsp!	다른 어플리케이션이 장치를 사용하고 있다면 fstat  grep dsp 로 체크한다. 특별히 문제가 되는 것은 <b>esound</b> 와 <b>KDE</b> 의 사운드 지원이다.

### 7.2.3 멀티플 사운드 소스 사용

**esound** 또는 **artsd** 가 특정 어플리케이션과 사운드 장치 공유를 지원하지 않을때 동시에

---

플레이 할수 있도록 멀티플 사운드 소스를 설정하는 것도 바람직하다.

FreeBSD 는 `sysctl(8)`로 설정할 수 있는 *가상 사운드 채널*로 멀티플 소스를 설정하게 된다. 가상 채널은 커널에서 사운드를 섞어 사운드 카드의 플레이 채널을 입체적으로 만든다.

root 유저라면 여러 가상 채널을 설정할 수 있는 두 개의 `sysctl` 이 있다:

```
# sysctl hw.snd.pcm0.vchans=4
# sysctl hw.snd.maxautovchans=4
```

위의 예제는 보편적으로 사용하기에 적당한 4 개의 가상 채널을 할당한다.

`hw.snd.pcm0.vchans` 은 `pcm0` 이 가지고 있는 가상 채널 개수이고 장치를 붙여서 설정할 수 있다. `hw.snd.maxautovchans` 는 `kldload(8)`로 새로운 오디오 장치를 추가했을 때의 가상 채널 개수다. `pcm` 모듈은 하드웨어 드라이버를 독립적으로 로드할 수 있기 때문에 `hw.snd.maxautovchans` 는 나중에 추가된 장치가 가지고 있는 가상 채널 개수를 저장할 수 있다.

`devfs(5)`를 사용하지 않는다면 어플리케이션을 `/dev/dsp0.x` 로 지정해야 된다. 여기서 `x` 는 `hw.snd.pcm.0.vchans` 가 위의 예제처럼 4 로 설정되었다면 0 에서 3 까지다. 시스템에서 `devfs(5)`를 사용한다면 위의 내용은 유저에게 자동으로 할당된다.

## 7.3 MP3 오디오

MP3(MPEG 레이어 3 오디오)는 CD 수준의 음질을 제공하기 때문에 여러분의 FreeBSD 워크스테이션이 지원하도록 해야 된다.

### 7.3.1 MP3 플레이어

가장 유명한 XFree86 MP3 플레이어는 XMMS(X 멀티미디어 시스템)다. GUI 가 Nullsoft 의 Winamp 와 거의 비슷하기 때문에 Winamp 스킨을 XMMS 에 사용할 수 있다. 또한 XMMS 플러그인을 지원한다.





**XMMS** 는 [multimedia/xmms](#) 포트나 패키지에서 설치할 수 있다.

**XMMS** 의 인터페이스는 직관적인 플레이 리스트와 그래픽 이퀄라이저 등을 가지고 있다. 이렇게 **Winamp** 와 유사하기 때문에 **XMMS** 가 사용하기 편하다는 것을 알게 될 것이다.

[audio/mpg123](#) 포트는 명령어 라인 MP3 플레이어다.

**mpg123** 는 명령어 라인에서 사운드 장치와 MP3 파일을 아래와 같이 지정할 수 있다:

```
# mpg123 -a /dev/dsp1.0 Foobar-GreatestHits.mp3
```

High Performance MPEG 1.0/2.0/2.5 Audio Player for Layer 1, 2 and 3.

Version 0.59r (1999/Jun/15). Written and copyrights by Michael Hipp.

Uses code from various people. See 'README' for more!

THIS SOFTWARE COMES WITH ABSOLUTELY NO WARRANTY! USE AT YOUR OWN RISK!

Playing MPEG stream from BT - Foobar-GreastHits.mp3 ...

MPEG 1.0 layer III, 128 kbit/s, 44100 Hz joint-stereo

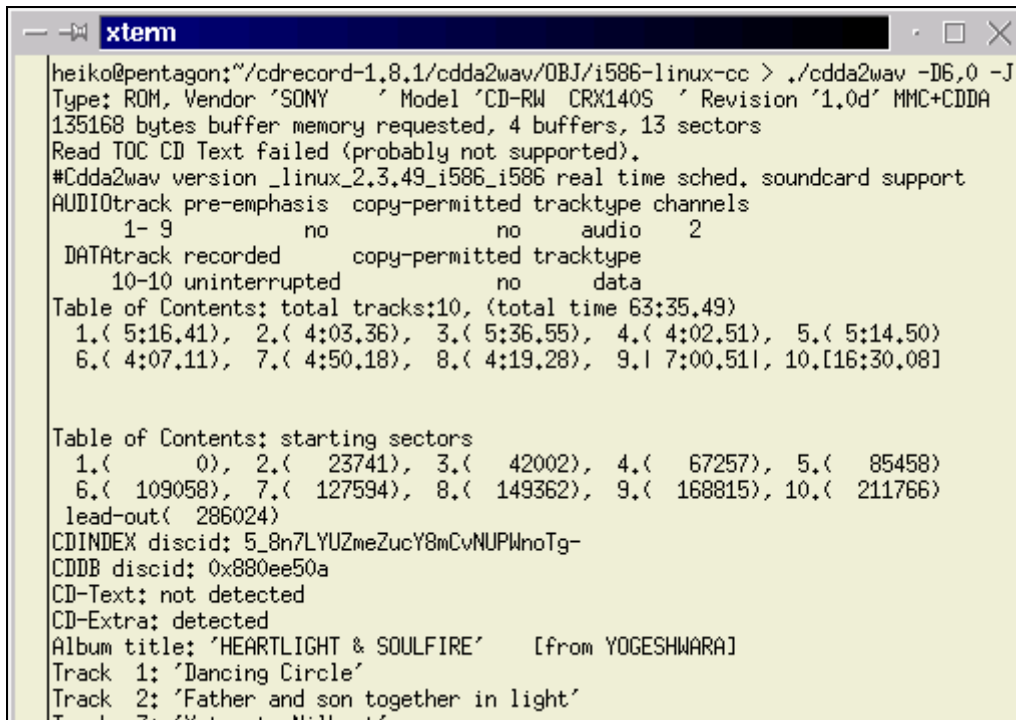
`/dev/dsp1.0` 은 여러분 시스템의 dsp 장치 엔트리로 대체한다.

---

## 7.3.2 CD 오디오 트랙 추출

CD를 인코딩 하거나 CD 트랙을 MP3로 변환하기 전에 CD의 오디오 데이터를 하드 드라이브에 추출해야 된다. raw CDDA(CD 디지털 오디오) 데이터를 WAV 파일로 추출하면 된다.

[sysutils/cdrtools](#)에 포함된 `cdda2wav` 툴은 CD와 관련된 오디오 정보를 CD에서 추출 하는데 사용된다.



```
heiko@pentagon:~/cdrecord-1.8.1/cdda2wav/OBJ/i586-linux-cc > ./cdda2wav -D6,0 -J
Type: ROM, Vendor 'SONY' Model 'CD-RW CRX140S' Revision '1.0d' MMC+CDDA
135168 bytes buffer memory requested, 4 buffers, 13 sectors
Read TOC CD Text failed (probably not supported).
#Cdda2wav version _linux_2.3.49_i586_i586 real time sched. soundcard support
AUDIOtrack pre-emphasis copy-permitted tracktype channels
  1- 9      no      no      audio  2
DATAtrack recorded copy-permitted tracktype
 10-10  uninterrupted      no      data
Table of Contents: total tracks:10, (total time 63:35.49)
  1.( 5:16.41),  2.( 4:03.36),  3.( 5:36.55),  4.( 4:02.51),  5.( 5:14.50)
  6.( 4:07.11),  7.( 4:50.18),  8.( 4:19.28),  9.( 7:00.51), 10.(16:30.08)

Table of Contents: starting sectors
  1.( 0),  2.( 23741),  3.( 42002),  4.( 67257),  5.( 85458)
  6.( 109058),  7.( 127594),  8.( 149362),  9.( 168815), 10.( 211766)
lead-out( 286024)
CDINDEX discid: 5_8n7LYUZmeZucY8mCvNUPWnoTg-
CDDB discid: 0x880ee50a
CD-Text: not detected
CD-Extra: detected
Album title: 'HEARTLIGHT & SOULFIRE' [from YOGESHWARA]
Track 1: 'Dancing Circle'
Track 2: 'Father and son together in light'
```

오디오 CD를 드라이브에 넣고 다음 명령으로 (root에서) 전체 CD를 각각의(트랙 단위로) WAV 파일로 추출할 수 있다:

```
# cdda2wav -D 0,1,0 -B
```

`cdda2wav`는 ATAPI(IDE) CDROM 드라이브를 지원한다. IDE 드라이브에서 추출하려면 SCSI 유닛 번호 대신 장치 이름을 지정한다. 예를 들어 IDE 드라이브에서 트랙 7을 추출하려면 다음 명령을 사용한다.

---

```
# cdda2wav -D /dev/acd0a -t 7
```

-D 0,1,0 는 cdrecord -scanbus 의 출력에 맞는 SCSI 장치 0,1,0 을 나타낸다.

각각의 트랙을 추출하려면 아래와 같이 -t 옵션을 사용한다:

```
# cdda2wav -D 0,1,0 -t 7
```

이 예제는 오디오 CDROM 의 트랙 7 을 추출한다. 예를 들어 1 에서 7 의 트랙 범위를 추출하려면 범위를 지정한다:

```
# cdda2wav -D 0,1,0 -t 1+7
```

유틸리티 dd(1)도 ATAPI 드라이브에서 오디오 트랙을 추출하는데 사용한다. 더 많은 정보는 16.5.5 장을 읽는다.

### 7.3.3 MP3 인코딩

요즘 mp3 인코더의 선택은 **lame** 다. **Lame** 는 포트 트리의 [audio/lame](#) 에서 찾을 수 있다.

추출한 WAV 파일을 사용하기 위해 다음 명령이 audio01.wav 를 audio01.mp3 로 변환한다:

```
# lame -h -b 128 \  
--tt "Foo Song Title" \  
--ta "FooBar Artist" \  
--tl "FooBar Album" \  
--ty "2001" \  
--tc "Ripped and encoded by Foo" \  
--tg "Genre" \  
audio01.wav audio01.mp3
```

128 kbits 는 표준 MP3 비트율로 사용되고 있지만 160 이나 192 비트율의 고 품질을 즐길 수 있다. 더 높은 비트율은 MP3 가 많은 디스크 공간을 소비하지만 음질은 더 좋다. -h 옵션은 "고 음질이지만 좀더 느린" 모드다. -t 옵션으로 시작하는 것은 MP3 파일에 저장된 일반적인 노래 정보를 가지고 있는 ID3 태그를 보여준다. 추가적인 인코딩 옵션은 lame 매뉴얼 페이지를 참고하여 찾을 수 있다.

---

## 7.3.4 MP3 디코딩

MP3 에서 오디오 CD 를 레코딩 하려면 압축되지 않은 WAV 포맷으로 변환시켜야 한다. XMMS 와 mpg123 은 MP3 의 출력을 압축되지 않은 파일 포맷으로 만든다.

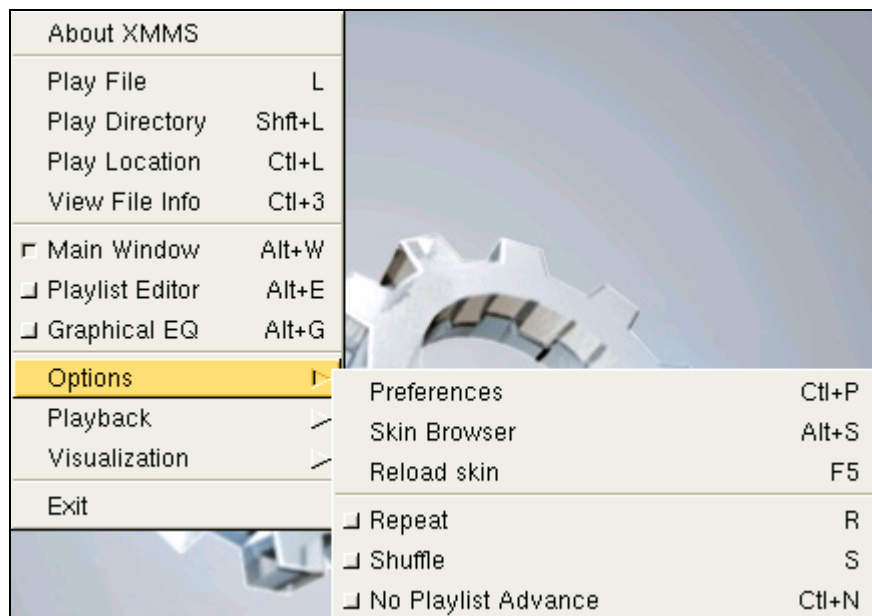
**XMMS** 에서 디스크를 만들려면 다음 절차를 따른다.

[예제: XMMS 에서 디스크 만들기]

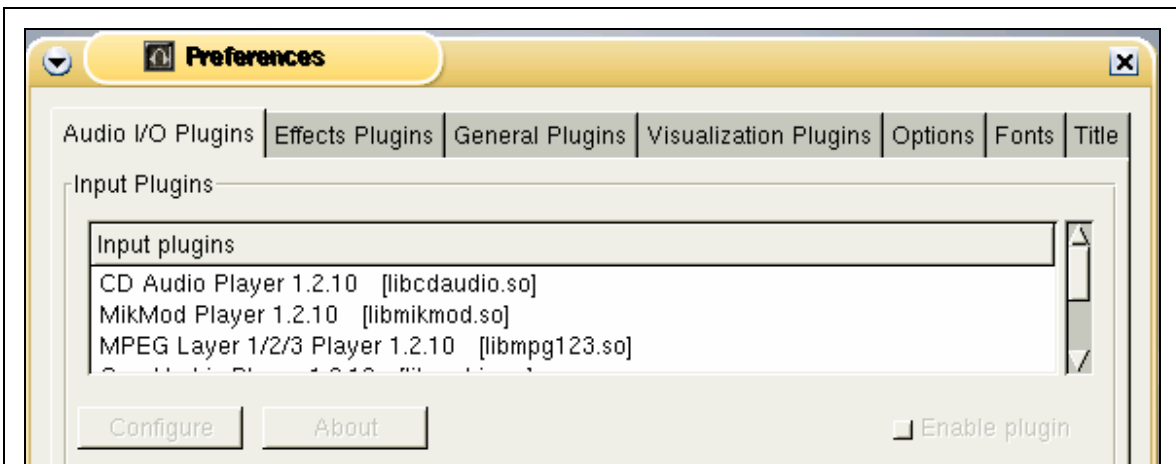
1. **XMMS** 를 실행한다.

% **xmms &**

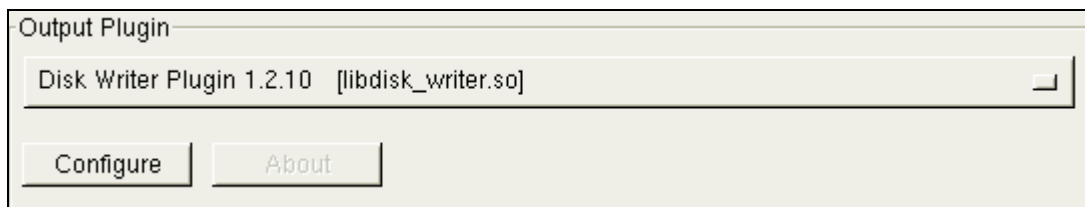
2. **XMMS** 메뉴를 불러오기 위해 윈도우에서 오른쪽 마우스를 클릭한다.



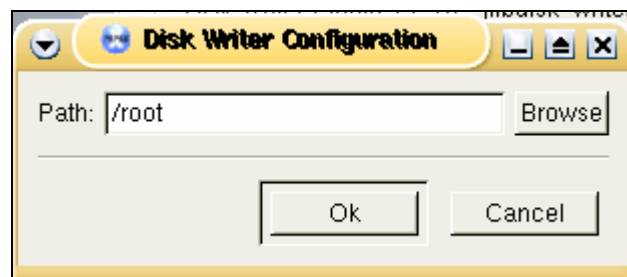
3. *Options*에서 *Preference*를 선택한다.



- 출력 플러그인을 "Disk Writer Plugin"으로 변환한다.



- Configure**를 누른다.
- 압축되지 않은 파일을 저장할 디렉터리를 입력한다(또는 브라우저를 선택한다).



- 볼륨은 100%, EQ 설정을 끄고 일반적인 방법으로 MP3 파일을 **XMMS**로 로드 한다.
- Play*를 누르면 **--XMMS**는 MP를 플레이 하지만 음악은 들을 수 없다. 실제로 MP3를 파일로 플레이 하는 것이다.



9. MP3로 음악을 다시 듣기 전에 기본 출력 플러그인 설정을 되돌려놓는것을 잊지 않는다.

**mpg123** 에서 표준 출력 작성하려면 다음 명령을 실행한다.

```
% mpg123 -s audio01.mp3 > audio01.pcm
```

**XMMS** 는 WAV 포맷으로 파일을 작성하지만 **mpg123** 은 MP3 를 raw PCM 오디오 데이터로 변환한다. 이들 포맷은 **cdrecord** 로 오디오 CD 를 생성하는데 사용할 수 있다. raw PCM 은 **burncd(8)**로 사용해야 된다. WAV 파일을 사용한다면 각 트랙의 시작 부분에서 WAV 파일의 헤더 소리인 작게 튀는 소리를 듣게 된다. **SoX** 유틸리티로([audio/sox](#) 포트나 패키지로 설치할 수 있다) WAV 파일의 헤더를 간단히 삭제할 수 있다:

```
% sox -t wav -r 44100 -s -w -c 2 track.wav track.raw
```

FreeBSD 에서 CD 레코더 사용에 대한 더 많은 정보는 16.6 장을 읽는다.

## 7.4 비디오 재생

비디오 재생은 매우 새롭고 급속하게 개발되는 어플리케이션 영역이다. 모든 것이 사운드처럼 자연스럽게 작동하지 않기 때문에 지금은 참아야 된다.

시작하기 전에 가지고 있는 비디오 카드 모델과 사용하는 칩셋을 알고 있어야 한다.

**XFree86** 이 광범위하고 다양한 비디오 카드를 지원하지만 좋은 성능을 제공하는 것은 몇 개 되지 않는다. X 서버가 여러분의 카드를 사용하여 지원할 수 있는 범위에 관한 리스트를 보려면, X11 이 작동하는 동안 **xvinfo(1)** 명령을 사용한다.

---

다양한 플레이어와 옵션을 평가하기 위해 테스트 파일로 이용할 수 있는 짧은 MPEG 파일을 가지고 있는 것도 좋은 아이디어다. 어떤 DVD 플레이어는 DVD 미디어를 기본적으로 /dev/devd 에서 찾거나 이 장치 이름이 내장되어 있기 때문에, 적당한 장치에 심볼릭 링크를 거는 것이 유용 할 수 있다:

```
# ln -sf /dev/acd0c /dev/dvd
# ln -sf /dev/racd0c /dev/rdvd
```

devfs(5)를 사용하는 FreeBSD 5.X 는 권장하는 링크 설정과 약간 다르다:

```
# ln -sf /dev/acd0c /dev/dvd
# ln -sf /dev/acd0c /dev/rdvd
```

이렇게 직접 생성한 링크는 devfs(5)의 특성으로 시스템을 재 부팅하면 없어지기 때문에 시스템을 부팅할 때마다 자동으로 심볼릭 링크를 생성하도록 /etc/devfs.conf 에 다음 라인을 추가한다:

```
link acd0 dvd
link acd0 rdvd
```

추가적으로 특별한 DVD-ROM 기능이 필요한 DVD 디코딩은 DVD 장치에 쓰기 권한이 필요하다.

어떤 포트는 정확하게 빌드하기 위해 다음 커널 옵션이 필요하다. 빌드하기 전에 커널 설정 파일에 이들 옵션을 추가하고 새로운 커널을 빌드 후 재 부팅한다:

```
option CPU_ENABLE_SSE
option USER_LDT
```

**Note:** option *USER\_LDT* 는 FreeBSD 5.X 에 없다.

X11 인터페이스 공유 메모리를 높이기 위해 몇몇 sysctl(8) 변수를 증가해야 된다:

```
kern.ipc.shmmax=67108864
kern.ipc.shmall=32768
```

---

## 7.4.1 비디오 성능 결정

X11 에서 비디오를 플레이 할수 있는 몇 가지 방법이 있다. 실제 동작은 하드웨어에 따라 다르다. 아래에서 설명하는 각각의 방법은 하드웨어에 따라 품질도 변한다. 두 번째로 X11 에서의 비디오를 플레이하는 것이 최근 많은 주목을 받고있기 때문에 XFree86 의 각 버전에서 상당히 발전될 것이다.

일반적인 비디오 인터페이스 리스트:

- ① X11: 공유 메모리를 사용하는 일반적인 X11 출력
- ② XVideo: X11 drawabel에서 비디오를 지원하는 X11 인터페이스 확장
- ③ SDL: 간단한 직접 미디어 레이어
- ④ DGA: 직접 그래픽 접근
- ⑤ SVGALib: 저 레벨 콘솔 그래픽 레이어

### 7.4.1.1 XVideo

**XFree86 4.X** 는 특별한 가속기를 통해 표현할 수 있는 오브젝트에서 비디오를 직접 플레이 하는 *XVideo* 로(Xvideo, Xv, xv 로 알려진) 확장된다. 이 확장으로 낮은 성능의 머신에서도(예를들어 내 PIII 400 Mhz 노트북에서도) 매우 좋은 품질의 플레이를 제공한다. 불행히 이 기능이 지원되는 카드 리스트는 현재 생산되지 않는다:

- ① 3DFX 부두 3
- ② 인텔 i810과 i815
- ③ 몇몇 S3칩 (세비지/IX와 세비지/MX 같은)

가지고 있는 카드가 이들 중에 없더라도 아직 실망하기는 이르다. **XFree86 4.X** 는 새로운 xv 기능을 각 릴리즈에 추가했다. xvinfo 로 확장되는지 체크한다:

% **xvinfo**

다음과 비슷한 결과를 보게 되면 XVideo 는 여러분의 카드를 지원한다:

```
X-Video Extension version 2.2
screen #0
Adaptor #0: "Savage Streams Engine"
```



---

number of ports: 1  
port base: 43  
operations supported: PutImage  
supported visuals:  
depth 16, visualID 0x22  
depth 16, visualID 0x23  
number of attributes: 5  
"XV\_COLORKEY" (range 0 to 16777215)  
client settable attribute  
client gettable attribute (current value is 2110)  
"XV\_BRIGHTNESS" (range -128 to 127)  
client settable attribute  
client gettable attribute (current value is 0)  
"XV\_CONTRAST" (range 0 to 255)  
client settable attribute  
client gettable attribute (current value is 128)  
"XV\_SATURATION" (range 0 to 255)  
client settable attribute  
client gettable attribute (current value is 128)  
"XV\_HUE" (range -180 to 180)  
client settable attribute  
client gettable attribute (current value is 0)  
maximum XvImage size: 1024 x 1024  
Number of image formats: 7  
id: 0x32595559 (YUY2)  
guid: 59555932-0000-0010-8000-00aa00389b71  
bits per pixel: 16  
number of planes: 1  
type: YUV (packed)  
id: 0x32315659 (YV12)  
guid: 59563132-0000-0010-8000-00aa00389b71  
bits per pixel: 12  
number of planes: 3  
type: YUV (planar)  
id: 0x30323449 (I420)  
guid: 49343230-0000-0010-8000-00aa00389b71

---

bits per pixel: 12  
number of planes: 3  
type: YUV (planar)  
id: 0x36315652 (RV16)  
guid: 52563135-0000-0000-0000-000000000000

bits per pixel: 16  
number of planes: 1  
type: RGB (packed)  
depth: 0  
red, green, blue masks: 0x1f, 0x3e0, 0x7c00  
id: 0x35315652 (RV15)  
guid: 52563136-0000-0000-0000-000000000000

bits per pixel: 16  
number of planes: 1  
type: RGB (packed)  
depth: 0  
red, green, blue masks: 0x1f, 0x7e0, 0xf800  
id: 0x31313259 (Y211)  
guid: 59323131-0000-0010-8000-00aa00389b71

bits per pixel: 6  
number of planes: 3  
type: YUV (packed)  
id: 0x0  
guid: 00000000-0000-0000-0000-000000000000

bits per pixel: 0  
number of planes: 0  
type: RGB (packed)  
depth: 1  
red, green, blue masks: 0x0, 0x0, 0x0

또한 리스트 되어있는 포맷은(YUV2, YUV12 등) 모든 Xvideo 로 표현되지 않기 때문에 어떤 플레이어에는 역효과를 발생한다.

결과가 다음과 비슷하다면:

X-Video Extension version 2.2  
screen #0

---

no adaptors present

XVideo 가 여러분의 카드를 지원하지 못할 것이다.

XVideo 가 카드를 지원하지 않는다면 이 의미는 화면이 비디오 표현 요청을 제어하기가 더 어렵다는 것이다. 비디오 카드와 프로세서에 따라 다르지만 충분한 경험으로 해결할 수 있을 것이다. 성능 향상 방법에 대해 7.4.3 장에서 읽을 수 있다.

### 7.4.1.2 간단한 직접 미디어 레이어

사운드와 그래픽을 효과적으로 사용할 수 있도록 크로스 플랫폼 어플리케이션으로 개발되고 있는 간단한 직접 미디어 레이어 SDL 은 Microsoft Windows, BeOS 그리고 유닉스 사이에서 포팅 레이어로 되어있다. SDL 레이어는 가끔 X11 인터페이스보다 더욱 효과적일 수 있는 낮은 레벨의 추상개념을 하드웨어에 제공한다.

SDL 은 [devel/sdl12](#)에서 찾을 수 있다.

### 7.4.1.3 직접 그래픽 접근

직접 그래픽 접근은 프로그램이 X 서버를 우회하여 직접 프레임 버퍼를 수정할 수 있는 XFree86 확장이다. 낮은 레벨 메모리 매핑에 놓여있기 때문에 효과적으로 공유하기 위해 프로그램을 root 로 실행해서 이 확장을 사용한다.

DGA 확장은 dga(1)로 테스트하고 벤치마크 할수 있다. dga 가 실행되면 키를 누를 때마다 화면의 색상을 바꾼다. 끝내기 위해 q 를 사용한다.

## 7.4.2 비디오 관련 포트와 패키지

이 섹션은 FreeBSD 포트 컬렉션에서 설치할 수 있고 비디오 플레이에 사용할 수 있는 소프트웨어에 대해 다룬다. 비디오 플레이는 아주 적극적으로 개발되고 있는 소프트웨어 영역이기 때문에 다양한 어플리케이션 기능이 여기서 설명하는 것과 약간 다르다.

---

첫째로 FreeBSD 에서 실행되는 대부분의 비디오 어플리케이션은 예전에 리눅스용 어플리케이션으로 개발되었다는 점을 알고 있어야 된다. 이들 중 대부분은 아직도 배타 품질에 머무르고 있다. FreeBSD 비디오 패키지에서 부딪힐 수 있는 문제는 다음과 같은 사항을 포함하고 있다:

- ① 어플리케이션이 다른 어플리케이션이 만든 파일을 플레이 할수 없다.
- ② 어플리케이션이 직접 만든 파일을 플레이 할수 없다.
- ③ 두 머신의 똑같은 어플리케이션은 각 머신에 맞도록 다시 빌드하고 같은 파일을 다르게 플레이 한다.
- ④ 이미지 크기 재 조정처럼 사소해 보이는 필터가 재 조정 루틴의 버그로 아주 나쁜 결과를 야기한다.
- ⑤ 어플리케이션은 항상 코어를 덤프한다
- ⑥ 문서가 포트에 설치되지 않고 웹이나 포트의 work에서 찾을 수 있다.

이들 어플리케이션 대부분도 “Linux-isms”를 나타낼 것이다. 리눅스 배포본으로 실행되는 어떤 표준 라이브러리나 어플리케이션의 개발자에게 전유 당해온 리눅스 커널의 어떤 기능으로 문제가 있을 것이다. 다음과 같은 문제를 야기하는 이들 문제가 항상 통보된 것은 아니었다.

- ① /proc/cpuinfo 사용은 프로세서 특성을 찾기 위한 것이다.
- ② 프로그램이 완벽하게 끝나지 않고 정지할 수 있는 쓰레드의 잘못된 사용.
- ③ 어플리케이션과 보편적으로 같이 사용되는 FreeBSD 포트 컬렉션에 아직 없는 소프트웨어.

게다가 이런 어플리케이션 개발자는 포트 관리자와 최소한의 작업으로 필요한 포팅을 하기 위해 제후해 왔기 때문에 이런 문제는 아직도 예상할 수 있다.

### 7.4.2.1 MPlayer

**MPlayer** 는 최근에 급속하게 비디오 플레이로 개발되고 있다. **MPlayer** 팀의 목적은 리눅스와 다른 유닉스에서 속도와 유연성을 갖는 것이다. 프로젝트는 팀이 그 당시 사용할 수 있는 플레이어에서 좋지 않은 플레이 성능에 질렸을 때 시작되었다. 어떤 사람은 인터페이스가 유선형 디자인을 위해 희생되었다고 말하지만 명령어 라인 옵션과 키 입력으로 제어를 해보면 꽤 괜찮다는 것을 알게 될 것이다.

---

[예제: MPlayer 설치 및 사용하기]

## 1. MPlayer 빌드

MPlayer 는 [multimedia/mplayer](#) 에 있다. MPlayer 는 빌드하는 동안 다양한 하드웨어를 체크해서 이 시스템에서 다른 시스템으로 이식할 수 없는 바이너리를 생성한다. 그래서 바이너리 패키지를 사용하지 않고 포트에서 빌드해야 된다. 게다가 빌드를 시작할때 발생하는 많은 옵션을 make 에 지정할 수 있다.

```
# cd /usr/ports/multimedia/mplayer
```

```
# make
```

You can enable additional compilation optimizations

by defining WITH\_OPTIMIZED\_CFLAGS

You can enable GTK GUI by defining WITH\_GUI.

You can enable DVD support by defining WITH\_DVD.

You can enable SVGALIB support by defining WITH\_SVGALIB.

You can enable VORBIS sound support by defining WITH\_VORBIS.

You can enable XAnim DLL support by defining WITH\_XANIM.

[x11-toolkits/gtk12](#) 가 설치되어 있다면 GUI 를 활성화할 수 있다. 설치 되지 않았다면 설치할 필요는 없다. MPlayer 로 DVD 를(CSS 인코딩) 플레이 하려면 DVD 지원 옵션을 활성화해야 된다. 몇 가지 옵션은 다음과 같다:

```
# make WITH_DVD=yes WITH_SVGALIB=yes
```

이 문서를 작성할 당시 MPlayer 포트는 HTML 문서와 실행할 수 있는 한 개의 mplayer 를 빌드했다. 또한 비디오를 다시 인코딩하는 틀인 mencoder 도 빌드한다. Makefile 를 수정하여 활성화할 수 있으며 포트의 다음 버전에서는 기본적으로 활성화 될 것이다.

MPlayer HTML 문서는 매우 유용하다. 독자가 비디오 하드웨어와 인터페이스에 대해 이번 장에서 부족한 정보를 찾는다면 MPlayer 문서가 필요한 정보를 완벽하게 제공해 준다. 유닉스에서 비디오 지원에 대한 정보를 찾는다면 MPlayer 문서를 읽기 위해 시간을 투자해야 한다.

## 2. MPlayer 사용

---

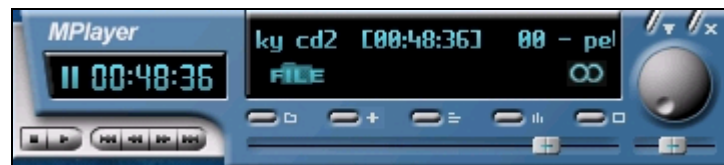
**MPlayer** 유저는 홈 디렉터리에 `.mplayer` 서브디렉터리를 설정해야 된다. 필요한 서브디렉터리를 생성하려면 다음 명령을 사용한다:

```
% cd /usr/ports/multimedia/mplayer
```

```
% make install-user
```

`mplayer` 명령 옵션은 매뉴얼 페이지에 나와있으며 더 자세한 HTML 문서까지 있다.

이 섹션에서는 일반적인 사용법만 설명해준다.



`testfile.avi` 와 같은 파일에서 플레이 하려면 비디오 인터페이스 중 하나는 `-vo` 설정을 해야 된다:

```
% mplayer -vo xv testfile.avi
```

```
% mplayer -vo sdl testfile.avi
```

```
% mplayer -vo x11 testfile.avi
```

```
# mplayer -vo dga testfile.avi
```

```
# mplayer -vo 'sdl:dga' testfile.avi
```

성능은 여러 가지 요인과 특히 하드웨어에 따라 다르기 때문에 이들 모든 옵션을 사용해 보는 것이 좋다.

DVD 에서 플레이 하려면 `testfile.avi` 를 `-dvd N DEVICE` 로 변경한다. `N` 은 플레이 할 제목의 번호고 `DEVICE` 는 DVD-ROM 의 장치 노드다. 예를 들어 `/dev/dvd` 에서 제목 3 을 플레이 하려면 다음 명령을 실행한다:

```
# mplayer -vo dga -dvd 2 /dev/dvd
```

정지, 잠시 멈춤 등은 `mplayer -h` 의 출력 키 바인딩을 참고하거나 매뉴얼 페이지를 읽는다.

플레이에 추가적으로 중요한 옵션은 풀 스크린 모드와 성능을 높여주는 `-framedrop` 과 연관된 `-fs -zoom` 이다.

`mplayer` 명령어 라인이 너무 길어지지 않도록 유저는 `.mplayer/conf` 파일을 생성하고 기본

---

옵션을 설정할 수 있다:

```
vo=xv  
fs=yes  
zoom=yes
```

마지막으로 mplayer 는 DVD 타이틀을 .vob 파일로 추출하는데 사용할 수 있다. DVD 에서 두 번째 타이틀을 추출하려면 아래와 같이 입력한다:

```
# mplayer -dumpstream -dumpfile out.vob -dvd 2 /dev/dvd
```

출력파일 out.vob 는 MPEG 이고 이 섹션에서 설명한 다른 패키지로 조정할 수 있다.

### **7.4.2.1.3 mencoder**

빌드할때 mencoder 를 설치했다면 이것은 경험이 필요하다는 것을 충고해준다.

mencoder 를 사용하기 전에 HTML 문서 옵션에 익숙해지는 것은 좋은 생각이다. 매뉴얼 페이지가 있지만 HTML 없이는 별로 유용하지 않다. 품질을 증진, 낮은 비트율 그리고 포맷 변환을 위한 몇 가지 방법이 있고 이들 트릭 중 몇 가지는 성능을 좋거나 나쁘게 만들 수 있다. 여기 몇 가지 예제가 있다. 첫 번째는 단순히 복사만 한다:

```
% mencoder input.avi -oac copy -ovc copy -o output.avi
```

적절하지 못한 명령어 라인 옵션은 mplayer 로도 플레이 할수 없는 파일을 생성한다. 그러므로 단지 파일만 추출하려면 mplayer 에 *-dumpfile* 을 사용한다.

input.avi 로 변환하려면 MPEG4 코덱과 MPEG3 오디오 인코딩이 필요하다([audio/lame](#) 이 필요하다.):

```
% mencoder input.avi -oac mp3lame -lameopts br=192 \  
-ovc lavc -lavcopts vcodec=mpeg4:vhq -o output.avi
```

이것은 mplayer 와 xine 로 플레이 할수 있는 파일을 생성한다.

input.avi 는 *-dvd 1 /dev/dvd* 로 바꾸고 DVD 타이틀을 다시 인코딩 하도록 root 로 실행할 수 있다. 처음 결과에 대해 별로 만족하지 못하기 때문에 타이틀을 파일로 덤프하고 이 파일로 작업하는 것을 권장한다.

---

### 7.4.2.2 xine 비디오 플레이어

**xine** 비디오 플레이어는 통합 비디오 솔루션에 다시 사용할 수 있는 라이브러리와 플러그인으로 확장되어 실행할 수 있는 모듈을 만드는 프로젝트다. 이것은 패키지와 포트 [multimedia/xine](#)에서 사용할 수 있다.

**Xine** 플레이어가 아직 마무리는 덜 되었지만 사용하기 좋게 만들어졌다. 사실 **xine**는 빠른 CPU와 비디오카드가 필요하거나 **XVideo** 확장 지원이 필요하다. GUI를 사용할 수 있지만 약간 어색하다.

이 문서를 작성하는 동안 DVD에 인코딩된 CSS를 플레이하는 입력 모듈이 **xine**에 적용된 것은 없다. 빌드할 수 있는 서드파티 모듈은 있지만 아직 FreeBSD 포트 컬렉션에 포함되지 않았다

**MPlayer**와 비교하면 **xine**이 사용하기 좀더 편하지만 유저들이 제어할 수 있는 좀더 나은 인터페이스가 필요하다. **xine** 비디오 플레이어도 **XVideo** 인터페이스에서 최상의 성능을 발휘한다.

기본적으로 **xine**는 그래픽 인터페이스로 시작된다. 메뉴는 특정 파일을 사용하는데 사용할 수 있다:

```
% xine
```

또는 GUI 없이 명령으로 즉시 파일을 플레이 할 수 있다.

```
% xine -g -p mymovie.avi
```

### 7.4.2.3 transcode 유틸리티

**transcode** 소프트웨어는 플레이어가 아닌 .avi를 .mpg 파일로 다시 인코딩 하는데 적당한 툴이다. **transcode**는 비디오 파일 합체와 깨진 파일 복구 기능이 있고 stdin/stdout 스트림 인터페이스로 명령어 라인 툴을 사용한다.

**MPlayer**처럼 **transcode**도 포트 [multimedia/transcode](#)에서 빌드해야되는 매우 실험적인 소프트웨어다. make 명령에 매우 많은 옵션이 사용되지만 개인적으로 다음 명령을 권장한다:



---

**# make WITH\_LIBMPEG2=yes**

multimedia/avifile를 설치할 계획이라면 아래와 같이 *WITH\_AVIFILE* 옵션을 make 명령에 추가한다:

**# make WITH\_AVIFILE=yes WITH\_LIBMPEG2=yes**

여기 **transcode** 로 재조정 된 출력을 만드는 비디오 변환에 관한 두 가지 예제가 있다. 첫 번째는 출력파일을 openDIVX AVI 파일로 인코드 하지만 두 번째는 더욱 호환성있는 MPEG 포맷으로 인코드한다.

```
% transcode -i input.vob -x vob -V -Z 320x240 \  
-y opendivx -N 0x55 -o output.avi
```

```
% transcode -i input.vob -x vob -V -Z 320x240 \  
-y mpeg -N 0x55 -o output.tmp
```

```
% tcplex -o output.mpg -i output.tmp.m1v -p output.tmp.mpa -m 1
```

transcode 은 매뉴얼 페이지가 있지만 같이 설치된 다양한 tc\* 유틸리티용은(tcplex 와 같은) 몇 개의 문서만 있다. *-h* 명령어 라인 옵션은 명령어 사용법만 보여준다.

비교해보면 transcode 는 mencoder 보다 느리게 실행되지만 좀더 호환 가능한 파일로 만들 수 있다. 예를 들어 transcode 로 생성한 MPEG 파일은 예전 윈도우 미디어 플레이어와 Apple 의 QuickTime 으로 플레이 되었다.

### 7.4.3 더 많은 자료

FreeBSD 용으로 다양한 비디오 소프트웨어 패키지가 엄청나게 개발되고 있다. 가까운 시기에 이번 장의 내용이 더 이상 맞지 않을 것이다. 그때까지 FreeBSD 의 최신 A/V 기능을 원하는 유저는 여러 FAQ 와 튜토리얼 및 다른 어플리케이션을 사용해 보면서 조합해야 된다.

이 섹션은 독자에게 더 많은 것을 배울 수 있는 몇 가지 링크를 제공한다.

---

Mplayer 문서는(<http://www.mplayerhq.hu/DOCS/>) 매우 기술적인 정보를 제공한다. 이들 문서는 유닉스 비디오의 전문적인 고급 기술을 원하는 사람들을 위한 문서다. **MPlayer** 메일링 리스트는 문서 읽기를 좋아하는 사람에게 적당하지 않기 때문에 버그 리포트를 보낼 계획이라면 신청한다.

xine 하우투는([http://dvd.sourceforge.net/xine-howto/en\\_GB/html/howto.html](http://dvd.sourceforge.net/xine-howto/en_GB/html/howto.html)) 모든 플레이어에 일반적인 성능 향상 정보를 가지고 있다.

마지막으로 여러분이 관심 있어 할만한 유망한 어플리케이션이 있다:

- 포트 [multimedia/avifile](http://multimedia/avifile)에 있는 **Avifile**(<http://avifile.sourceforge.net/>)
- 포트 [multimedia/ogle](http://multimedia/ogle)에 있는 **Ogle**(<http://www.dtek.chalmers.se/groups/dvd/>)
- **Xtheater**(<http://xtheater.sourceforge.net/>)

## 7.5 TV 카드 설치

### 7.5.1 소개

TV 카드는 방송이나 케이블 TV 를 컴퓨터로 볼수 있게 한다. 대부분은 RCA 또는 S-비디오 입력을 합성하거나 이중 어떤 카드는 FM 라디오 튜너도 가지고 있다.

FreeBSD 는 bktr(4) 드라이버로 Brooktree Bt848/849/878/879 또는 Conexant CN-878/Fusion 878a 비디오 캡처 칩을 사용하는 PCI 기반 TV 카드 지원을 제공한다. 튜너가 지원되는 보드도 지원하기 때문에 지원되는 튜너 리스트는 bktr(4) 매뉴얼 페이지를 참고한다.

### 7.5.2 드라이버 추가

카드를 사용하려면 다음 라인을 /boot/loader.conf 파일에 추가하여 bktr(4) 드라이버를 로드해야 된다:

```
bktr_load="YES"
```

---

다른 방법은 커널에 TV 카드 지원을 정적으로 컴파일 하고 이 경우 다음 라인을 커널 설정 파일에 추가한다:

```
device bktr
device iicbus
device iicbb
device smbus
```

카드 컴포넌트가 I2C 버스를 통해 연결되기 때문에 이들 장치 드라이버가 필요하다. 그리고 새로운 커널을 빌드하여 설치한다.

시스템에 지원을 추가하고 머신을 재 부팅해야 된다. 부팅하는 동안 다음과 같이 TV 카드가 나타난다:

```
bktr0: <BrookTree 848A> mem 0xd7000000-0xd7000fff irq 10 at device 10.0 on pci0
iicbb0: <I2C bit-banging driver> on bti2c0
iicbus0: <Philips I2C bus> on iicbb0 master-only
iicbus1: <Philips I2C bus> on iicbb0 master-only
smbus0: <System Management Bus> on bti2c0
bktr0: Pinnacle/Miro TV, Philips SECAM tuner.
```

물론 이들 메시지는 하드웨어에 따라 다르다. 그러나 튜너가 정확히 감지 되었는지 확인해야 된다. `sysctl(8)` MIB 와 커널 설정 파일 옵션으로 감지된 매개변수를 변경하는 것도 가능하다. 예를들어 튜너를 Philips SECAM 튜너로 지정하려면 커널 설정 파일에 다음 라인을 추가한다:

```
options OVERRIDE_TUNER=6
```

또는 직접 `sysctl(8)`을 사용할 수 있다:

```
# sysctl hw.bt848.tuner=6
```

가능한 옵션에 대한 더 자세한 사항은 `bktr(4)` 매뉴얼 페이지와 `/usr/src/sys/conf/NOTES` 파일을 확인한다. (FreeBSD 4.X 라면 `/usr/src/sys/conf/NOTES` 는 `/usr/src/sys/i386/conf/LINT` 로 대체된다)

---

## 7.5.3 유용한 어플리케이션

TV 카드를 사용하기 위해 다음 어플리케이션 중 한가지를 설치해야 된다.

- [multimedia/fxtv](#)로 윈도우에서 TV를 볼수 있고 이미지/오디오/비디오 기능도 된다.
- [multimedia/xawtv](#)도 [fxtv](#)와 같은 기능의 TV 어플리케이션이다.
- [misc/alevt](#)는 디코드해서 Videotext/Teletext를 표시한다.
- [audio/xmradio](#)는 몇몇 TV 카드에 붙어있는 FM 라디오 튜너를 사용하는 어플리케이션이다.
- [audio/wmtune](#)는 라디오 튜너용 핸디 데스크톱 어플리케이션이다.

FreeBSD 포트 컬렉션에서 더 많은 어플리케이션을 사용할 수 있다.

## 7.5.4 문제 해결

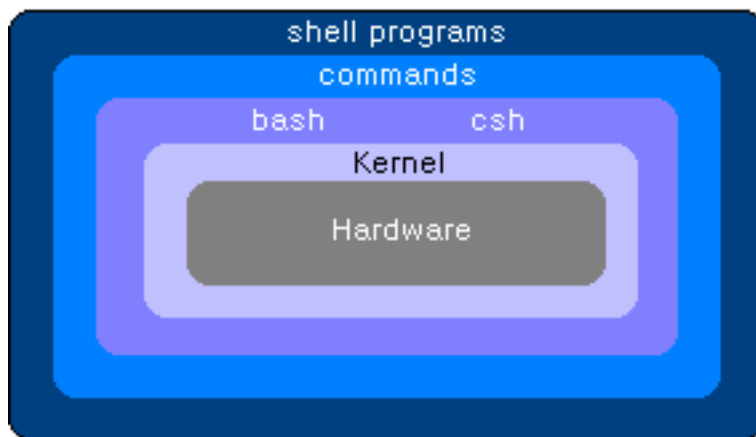
TV 카드와 문제가 발생한다면 첫째로 비디오 캡처 칩과 튜너가 [bktr\(4\)](#) 드라이버로 정확히 지원되는지 설정 옵션은 정확한지 확인한다. TV 카드에 대한 더 많은 지원과 다양한 질문은 [freebsd-multimedia](#) 메일링 리스트에(<http://lists.freebsd.org/mailman/listinfo/freebsd-multimedia>) 가입하고 아카이브를 검색한다.

---

## 8 장 FreeBSD 커널 설정

### 8.1 개요

커널은 FreeBSD 운영체제의 핵심이다. 메모리 관리, 보안 제어 시행, 네트워크, 디스크 사용과 더 많은 것들을 관리한다. FreeBSD 는 대부분 동적으로 설정할 수 있지만 커널을 다시 설정해서 컴파일하는 것이 아직도 가끔 필요하다.



이번 장을 읽은 후 다음과 같은 사항을 알 수 있다:

- 왜 사용자 커널을 빌드 해야 되는가
- 커널 설정 파일을 작성하거나 가지고있는 설정파일을 어떻게 변경하는가
- 새로운 커널을 생성하고 변경 하기 위해 커널 설정 파일을 어떻게 이용하는가
- 새로운 커널은 어떻게 설치하는가
- 필요할 수 있는 /dev 엔트리를 어떻게 생성하는가
- 문제가 생기면 어떻게 해결하는가

---

## 8.2 왜 사용자 커널을 빌드 해야 되는가?

전통적으로 FreeBSD는 “모놀리틱(monolithic)” 커널이라는 것을 가지고 있다. 이 의미는 커널이 지원하는 장치 리스트가 고정된 하나의 커다란 프로그램이기 때문에 커널 기능을 변경하려면 새로운 커널을 컴파일해서 컴퓨터를 재 부팅해야 된다.

요즘 FreeBSD는 필요할때 동적으로 로드하고 언 로드 할수 있도록 대부분의 커널 기능을 모듈에 포함시키는 방향으로 빠르게 변화하고 있다. 이러한 변화는 커널을 새로운 하드웨어에 빠르게 적용할 수 있도록(노트북에서 PCMCIA 카드와 같은) 하거나 처음 커널을 컴파일 할때 필요없던 새로운 기능을 커널에 넣을 수 있다. 이러한 커널을 모듈러(modular) 커널이라고하며 일상적으로는 KLD 라고 부른다.

상황이 이렇더라도 아직 정적인 커널 설정이 필요하다. 어떤 경우 기능적으로 필요한 것을 동적으로 로드 할수 없도록 커널에 묶어두기 때문이다. 또 다른 이유는 기능상 동적으로 로드 할수 있는 커널 모듈 개발을 시작할 수 있는 시간적인 여유를 가진 사람이 아직 없기 때문이다.

사용자 커널 빌드는 대부분의 유닉스 유저들이 꼭 치러야 되는 가장 중요한 통과 의례 중 하나다. 이 절차는 시간을 필요로 하지만 FreeBSD 시스템에 수 많은 장점을 제공한다. 다양한 하드웨어를 지원하는 GENERIC 커널과 달리 사용자 커널은 여러분 PC의 하드웨어만을 지원한다. 따라서 다음과 같은 많은 이점을 가져다 준다:

- 빠른 부팅시간. 커널은 시스템에있는 하드웨어만 찾기 때문에 시스템이 부팅하는 시간을 획기적으로 줄여준다.
- 더 적은 메모리 사용량. 사용자 커널은 종종 GENERIC 커널 보다 더 적은 메모리를 사용한다. 커널은 실제 메모리에 항상 로드되기 때문에 이것은 중요하다. 이러한 이유로 사용자 커널은 특히 RAM의 양이 적은 시스템에 유용하다.
- 추가적인 하드웨어 지원. 사용자 커널은 GENERIC 커널에 없는 사운드 카드 같은 장치 지원을 추가할 수 있다.

## 8.3 사용자 커널 빌드와 설치

첫째로 커널 빌드 디렉터리를 빠르게 탐색해 보자. 언급된 모든 디렉터리는 /sys를 통해서

도 접근할 수 있는 /usr/src/sys 디렉터리와 주로 연관이 있다. 이곳에 커널의 다른 곳을 표현하는 많은 서브 디렉터리가 있지만 우리에게 가장 중요한 곳은 사용자 커널 설정을 수정하고 커널을 컴파일하는 곳인 arch/conf 다. arch는 i386 이나 alpha 또는 pc98 을(일본에서 유명한 PC 하드웨어의 다른 개발 분기) 의미한다. 모든 특정 아키텍처의 디렉터리는 해당 아키텍처와 관련이 있다. 나머지 코드는 잠재적으로 FreeBSD 에 포트 할수 있는 모든 플랫폼에 공통적이다. 디렉터리 구조의 논리적인 조직과 지원되는 각 장치 그리고 파일 시스템 및 옵션은 각자의 서브 디렉터리에 있다. FreeBSD 5.X 와 그 후 버전은 sparc64 를 지원하고 몇가지 다른 아키텍처는 개발 중이다.

**Note:** 시스템에 /usr/src/sys 디렉터리가 없다면 커널 소스가 설치되지 않은 것이다. 커널 소스를 설치하는 가장 쉬운 방법은 root 에서 /stand/sysinstall 을 실행 → Configure → Distributions → src → sys 를 선택한다.

```
<<< X Exit      Exit this menu (returning to previous)
      All      Select all of the below
      Reset    Reset all of the below
[ ] base      top-level files in /usr/src
[ ] contrib   /usr/src/contrib (contributed software)
[ ] gnu       /usr/src/gnu (software from the GNU Project)
[ ] etc       /usr/src/etc (miscellaneous system files)
[ ] games     /usr/src/games (the obvious!)
[ ] include   /usr/src/include (header files)
[ ] lib       /usr/src/lib (system libraries)
[ ] libexec   /usr/src/libexec (system programs)
[ ] release   /usr/src/release (release-generation tools)
[ ] bin       /usr/src/bin (system binaries)
[ ] sbin      /usr/src/sbin (system binaries)
[ ] scrypto   /usr/src/crypto (contrib encryption sources)
[ ] share     /usr/src/share (documents and shared files)
[ ] skrb4     /usr/src/kerberosIV (sources for KerberosIV)
[ ] skrb5     /usr/src/kerberos5 (sources for Kerberos5)
[ ] ssecure   /usr/src/secure (BSD encryption sources)
[X] sys      /usr/src/sys (FreeBSD kernel)
[ ] tools     /usr/src/tools (miscellaneous tools)
[ ] ubin      /usr/src/usr.bin (user binaries)
[ ] usbin     /usr/src/usr.sbin (aux system binaries)
```

sysinstall 을 좋아하지 않는다면 공식적인 FreeBSD CDROM 에서 명령어 라인으로 소스를 설치할 수 있다.

```
# mount /cdrom
# mkdir -p /usr/src/sys
# ln -s /usr/src/sys /sys
# cat /cdrom/src/ssys.[a-d]* | tar -xzf -
```

그 다음 arch/conf 디렉터리로 이동해서 GENERIC 설정 파일을 원하는 이름으로 복사한다. 예를 들면 다음 명령과 같다.

```
# cd /usr/src/sys/i386/conf
```

---

# cp GENERIC MYKERNEL

관례적으로 이름은 모두 대문자고 다른 하드웨어에 설치된 다양한 FreeBSD 머신을 관리한다면 머신의 호스트 이름을 사용하는 것도 좋은 방법이다. 이 예제의 목적을 위해 우리는 MYKERNEL 이라고 이름을 붙였다.

**TIP:** /usr/src 에 커널 설정파일을 저장하지 않는 것이 좋다. 문제가 발생할 것 같아서 /usr/src 를 지우고 다시 시작하고 싶은 생각이 있을 수도 있다. 5 초 후 사용자 커널 설정 파일을 지웠다는 것을 인식할 것이다. 다음에 소스 트리를 업데이트할 때 덮어쓰기 하여 커널 설정을 잃어버리므로 GENERIC 를 직접 편집하지 않는다.

커널 설정 파일을 유지하려면 i386 디렉터리에 심볼릭 링크를 만든다.

예들 들어 다음 명령을 따른다.

```
# cd /usr/src/sys/i386/conf
# mkdir /root/kernels
# cp GENERIC /root/kernels/MYKERNEL
# ln -s /root/kernels/MYKERNEL
```

**Note:** 위 명령과 다음 명령은 root 계정에서 실행하지 않으면 퍼미션 거부 에러를 보게 된다.

이제 MYKERNEL 을 에디터로 수정한다. 처음 이라면 사용할 수 있는 에디터가 여기서 설명하기에 너무 복잡하지만 많은 책에서 설명이 잘 되어있는 vi 일 것이다. 그러나 여러분이 초보자로서 선택할 수 있는 ee 라는 쉬운 에디터를 FreeBSD 가 제공한다. 설정 파일 상단에 여러분의 설정을 설명하도록 주석문을 변경하거나 GENERIC 과 다르게 만들 수 있다.

SunOS 나 다른 BSD 운영체제에서 커널을 빌드해 보았다면 이런 파일이 아주 익숙할 것이다. 반면에 DOS 같은 다른 운영체제를 사용해 왔다면 GENERIC 설정파일은 어려울 수 있으므로 설정 파일 섹션의 설명을 천천히 그리고 주의 깊게 따른다.

**Note:** 소스 트리를 FreeBSD 프로젝트의 마지막 소스 트리와 동기화 하였다면 업데이트 과정을 수행하기 전에 /usr/src/UPDATING 파일을 항상 체크한다. 이 파일에 FreeBSD 업데이트와 관련된 모든 중요한 사항이 입력되어 있다.



---

`/usr/src/UPDATING` 는 항상 여러분의 FreeBSD 소스 버전과 일치하기 때문에 핸드북 보다 정확하다.

이제 커널 소스 코드를 컴파일 해야 된다. 커널을 다시 빌드해야 되는 이유와 관련된 한가지 방법과 운용 중인 FreeBSD 버전에 따라 컴파일 할수 있는 두 가지 절차가 있다.

- 오직 커널 소스코드만 설치했다면 절차 1 을 따른다.
- FreeBSD 4.0 이전 버전을 사용하고 `make world` 로 FreeBSD 4.0 이나 더 높은 버전으로 업그레이드 하지 않았다면 절차 1 을 이용한다.
- 소스코드는 업데이트 하지 않고 새로운 커널만 빌드 하였다면(아마 IPFIREWALL 처럼 새로운 옵션만 추가했을 것이다) 둘 중 한가지를 사용할 수 있다.
- `make world` 프로세스로 커널을 다시 빌드 하였다면 절차 2 를 사용한다.

[예제: 커널 빌드하기]

#### 절차 1. 전통적인 방법으로 커널 빌드

- ① 커널 소스 코드를 생성하기 위해 `config(8)`를 실행한다.

```
# /usr/sbin/config MYKERNEL
```

- ② 빌드 디렉터리로 변경한다. 이곳은 앞서 말한 명령을 실행하면 출력된다.

```
# cd ../compile/MYKERNEL
```

FreeBSD 버전 5.0 이전에서는 위 명령 대신 다음 명령을 사용한다.

```
# cd ../../compile/MYKERNEL
```

- ③ 커널 컴파일

```
# make depend  
# make
```

- ④ 새로운 커널 설치

```
# make install
```

- ⑤ 시스템을 재 부팅하여 다음 명령으로 아래와 같이 바뀐 커널 이름과 에러 메시지가 없는지 확인한다.

```
# dmesg |grep MYKERNEL
```

```
Copyright (c) 1992-2004 The FreeBSD Project.  
Copyright (c) 1979, 1980, 1983, 1986, 1988, 1989, 1991, 1992, 1993, 1994  
The Regents of the University of California. All rights reserved.  
FreeBSD 4.10-RELEASE #1: Tue Sep 7 12:14:39 KST 2004  
root@www.trustedbsd.co.kr:/usr/src/sys/compile/MYKERNEL  
Timecounter "i8254" frequency 1193182 Hz  
Timecounter "TSC" frequency 933354324 Hz  
CPU: Intel Pentium III (933.35-MHz 686-class CPU)
```

## 절차 2. 새로운 방법으로 커널 빌드

- ① /usr/src 디렉터리로 변경한다.

```
# cd /usr/src
```

- ② 커널 컴파일

```
# make buildkernel KERNCONF=MYKERNEL
```

- ③ 새로운 커널 설치

```
# make installkernel KERNCONF=MYKERNEL
```

- ④ 절차 1의 ⑤번에서처럼 바뀐 커널 이름 및 에러 메시지가 없는지 확인한다.

**Note:** FreeBSD 4.2와 이전 버전에서는 *KERNCONF* = 을 *KERNEL* =로 바꾸어야 된다. 4.2-STABLE은 2월 2일 이전에 패치되었기 때문에 *KERNCONF*=를 인지하지 못한다.

마지막으로 *buildworld-installworld* 를 성공적으로 완료한 후 어떤 방법으로든 소스 트리를

---

업그레이드(CVSup, CTM 을 실행하지 않았거나 anoncvs 을 사용하지 않았다면) 하지 않았다면 `config → make → depend → make → make install` 명령을 순서대로 실행한다.

**Note:** 새로운 장치를(사운드 카드 같은) 추가하였고 FreeBSD4.X 나 이전 버전을 사용한다면 이들을 사용하기 전에 /dev 디렉터리에 어떤 장치 노드를 추가해야 된다. 더 많은 정보는 장치 노드 생성 섹션을 본다.

새로운 커널은 root 디렉터리의 /kernel 에 복사되고 예전 커널은 /kernel.old 로 옮겨진다. 이제 시스템을 셧다운하여 새로운 커널로 재 부팅 한다. 문제가 발생한다면 이번 장의 마지막에서 문제해결을 위한 몇 가지 방법을 제시한다. 새로운 커널로 부팅하지 않는다면 복구 방법에 대한 설명을 읽는다.

**Note:** FreeBSD 5.0 에서 커널은 모듈과 함께 /boot/kernel 에 설치되고 예전 커널은 /boot/kernel.old 로 백업된다. 부트 loader(8)처럼 부트 프로세스와 관련된 다른 파일과 설정도 /boot 에 저장된다. 유저가 컴파일된 커널과 동기화된 모듈을 가지고 있는 것이 매우 중요하다고 생각하더라도 사용자 모듈은 /boot/modules 에 놓이게 된다. 컴파일된 커널과 연동하도록 설정하지 않은 모듈은 불안하거나 정확하지 않을 것이다.

## 8.4 설정 파일

설정 파일의 일반적인 포맷은 상당히 간단하다. 각 라인은 키워드와 하나 또는 하나 이상의 인수를 포함한다. 단순히 대부분의 라인은 인수 하나만을 가지고 있다. # 다음은 주석으로 무시된다고 생각하며 이번 섹션은 각 키워드에 대해 설명한다. 몇 개의 관련 키워드가 한 개의 섹션으로 그룹화되어 있거나(Networking 처럼) 이들이 실제로 GENERIC 파일에 흩어져 있더라도 보통 이들은 순서대로 GENERIC 에 리스트 되어있다. 완벽한 옵션 리스트와 장치 라인의 더욱 자세한 설명은 GENERIC 과 같은 디렉터리에 있는 LINT 설정 파일에 설명되어있다. 라인의 목적이나 필요성이 의심된다면 먼저 LINT 를 체크한다.

**Note:** FreeBSD 5.X 와 상위 버전에 LINT 는 없다. 아키텍처에 의존되는 옵션은 NOTE 파일을 본다. 주로 아키텍처에 의존되는 몇개의 옵션은 /usr/src/sys/conf/NOTES 파일에 저장되어 있다. 또한 이곳에서 옵션을 검토하는

---

것도 권장할만하다.

다음은 이해를 돕기 위해 설명을 추가한 GENERIC 커널 설정파일의 예제다. 이 예제는 여러분이 복사한 `/usr/src/sys/i386/conf/GENERIC` 과 상당히 흡사하다. 가능한 모든 커널 옵션에 대한 자세한 내용은 `/usr/src/sys/i386/conf/LINT` 를 본다.

```
#
# GENERIC -- Generic kernel configuration file for FreeBSD/i386
#
# For more information on this file, please read the handbook section on
# Kernel Configuration Files:
#
#   http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/kernelconfig-
#   config.html
#
# The handbook is also available locally in /usr/share/doc/handbook
# if you've installed the doc distribution, otherwise always see the
# FreeBSD World Wide Web server (http://www.FreeBSD.org/) for the
# latest information.
#
# An exhaustive list of options and more detailed explanations of the
# device lines is also present in the ../../conf/NOTES and NOTES files.
# If you are in doubt as to the purpose or necessity of a line, check first
# in NOTES.
#
# $FreeBSD: src/sys/i386/conf/GENERIC,v 1.380 2003/03/29 13:36:41 mdodd Exp $
```

다음은 빌드하는 모든 커널에서 필요한 필수 키워드다.

**machine**      **i386**

이것은 머신의 아키텍처를 의미한다. *i386*, *pc98*, *sparc64*, *alpha*, *ia64*, *amd64* 또는 *powerpc* 가 된다.

```
cpu      I486_CPU
cpu      I586_CPU
cpu      I686_CPU
```

위의 옵션은 시스템에 가지고 있는 CPU 타입을 지정한다. CPU 라인의 인수를 여러 줄로 할 수 있지만(예: 확실하지 않다면 *I586\_CPU*나 *I686\_CPU*를 사용한다) 사용자 커널에는 가지고 있는 CPU 만 지정한다. CPU 타입에 대해 확실하지 않다면 부팅 메시지를 볼 수 있는 `/var/run/dmesg.boot` 파일을 확인 한다.

```
The Regents of the University of California. All rights reserved.
FreeBSD 4.10-RELEASE #1: Tue Sep  7 12:14:39 KST 2004
root@www.trustedbsd.co.kr:/usr/src/sys/compile/MYKERNEL
Timecounter "i8254" frequency 1193182 Hz
Timecounter "TSC" frequency 933354324 Hz
CPU: Intel Pentium III (933.35-MHz 686-class CPU)
  Origin = "GenuineIntel" Id = 0x68a Stepping = 10
  Features=0x383f9ff<FPU,VME,DE,PSE,TSC,MSR,PAE,MCE,CX8,SEP,MTRR,PGE,MCA,
real memory = 335478784 (327616K bytes)
avail memory = 322916352 (315348K bytes)
```

*I386\_CPU*는 FreeBSD 소스에서 아직도 지원하고 있지만 기본적으로 STABLE 과 CURRENT 에서 비활성되어 있다. 이 의미는 386 급 CPU 에 FreeBSD 를 설치 하려면 다음 옵션을 따라야 된다:

- ① 예전 FreeBSD 릴리즈를 설치하고 8.3 장에서 설명한 것처럼 소스를 다시 빌드 한다.
- ② 새로운 머신에서 유저 영역과 커널을 빌드하고 미리 컴파일한 `/usr/obj` 파일을 사용하여 386 머신에 설치한다.
- ③ 설치 CD-ROM 의 커널이 지원하는 *I386\_CPU*를 포함하는 FreeBSD 릴리즈를 운용한다.

이들 중 첫 번째 옵션이 다른 것들보다 쉽겠지만 386 급 머신에서 구하기 어려운 디스크 공간이 많이 필요하다.

```
ident      GENERIC
```

이 옵션은 커널 이름을 확인한다. 이전 예제의 지시를 따랐다면 여러분의 커널에 붙인 이름으로 변경한다. 예: *MYKERNEL*. *Ident* 문자열에 넣은 값이 커널이 부팅할 때 출력되기 때문에 일반적인 커널과 차이를 두고 싶다면 새로운 커널에 다른 이름을 주는 것이 유용하다.

**maxusers**            *n*

*maxusers* 옵션은 중요한 시스템 테이블의 크기를 숫자로 지정한다. 이 숫자는 대략 머신에 동시에 접속하는 유저의 숫자와 일치해야 된다.

FreeBSD 4.5 가 시작되면서 이 옵션을 0으로(자동 튜닝 알고리즘 세트 *maxuser* 은 최소 32 에서 최대 384 로 시스템의 메모리 양과 같다.) 설정하였다면 시스템이 자동으로 설정한다.

FreeBSD 5.X 에서는 특별히 값을 지정하지 않았다면 *maxusers* 는 0이다. FreeBSD 4.5 보다 이전 버전을 사용하거나 직접 *maxusers* 를 관리하려면 특히 X 윈도우 시스템이나 소프트웨어를 컴파일 한다면 *maxusers* 를 최소한 4 이상으로 지정해야 된다. 그 이유는 *maxusers* 로 설정하는 가장 중요한 테이블은  $20 + 16 * maxusers$  로 설정하는 최대 프로세스이기 때문에, *maxusers* 를 1 로 지정했다면 시스템이 시작할 때 18 개와 X 윈도우 시스템을 시작할 때 생성되는 15 개를 포함하여 36 개의 프로세스만 동시에 시작할 수 있다. 매뉴얼 페이지 읽기처럼 간단한 업무 조차도 필터와 압축 풀기 그리고 읽기 위해 9 개의 프로세스가 필요하다. *maxusers* 를 64 로 설정하는 것이 거의 모든 유저에게 충분한 1044 프로세스를 동시에 수행할 수 있다. 그러나 다른 프로그램을 시작하려고 하거나 많은 동시 유저가 접속하여 서버가 운용될 때(ftp.FreeBSD.org 같은) 프로세스 테이블이 꽉 차는 것이 걱정된다면 이 숫자를 증가시키고 다시 빌드 할 수 있다.

**Note:** *maxusers* 는 머신에 로그인할 수 있는 유저 수를 제한하지 않는다. 단순히 시스템에 로그인하는 최대 유저와 얼마나 많은 프로세스를 각 유저들이 실행할지 고려하여 다양한 테이블 크기를 적당한 값으로 설정한다. 원격 로그인과 X 터미널 윈도우 동시 접속자를 제한하는 한가지 키워드는 *pseudo-device pty 16* 이다. FreeBSD 5.X 에서는 *pty(4)* 드라이버가 “자동-복제”되기 때문에 이 숫자에 대해 걱정할 필요 없고 설정 파일에 *device pty* 를 사용해야 된다.

---

```
# Floating point support – do not disable.
```

```
device          npx0          at nexus? port IO_NPX irq 13
```

*npx0*는 FreeBSD 에서 부동 소수점 계산을 하기 위한 하드웨어 코-프로세서이거나 소프트웨어 계산 에뮬레이터다. 이것은 필수다.

```
# Pseudo devices – the number indicates how many units to allocate.
```

```
device  loop          # Network loopback
```

위 옵션은 TCP/IP 의 일반적인 루프백 장치다. 텔넷이나 FTP 를 localhost 로(127.0.0.1) 한다면 이 장치를 거쳐 되돌아 온다. 이것은 필수다. FreeBSD 4.X 에서는 *pseudo-device loop* 라인을 사용한다.

다음은 대부분 옵션이다. 더 많은 정보는 아래의 노트나 그 다음의 각 옵션을 본다.

```
#To statically compile in device wiring instead of /boot/device.hints
```

```
#hints          "GENERIC.hints"          #Default places to look for devices.
```

FreeBSD 5.X 와 새로운 버전은 *device.hints(5)*가 장치 드라이버 옵션설정에 사용된다. *loader(8)*가 부팅할때 체크하는 기본 위치는 */boot/device.hints* 다. *hints* 옵션을 사용하여 이들 *hints*를 커널에 정적으로 컴파일 할수 있다. 따라서 */boot* 파일에 *device.hints*를 만들 필요는 없다.

```
#makeoptions    DEBUG=-g          #Build kernel with gdb(1) debug
symbols
```

FreeBSD 의 일반적인 빌드 과정은 커널이 링크된 후 빌드되어 대부분의 심볼들을 *STRIPS*할때 설치장소의 공간을 줄이기 위해 디버깅 정보를 포함하지 않는다. CURRENT 분기에서 커널을 테스트하거나 FreeBSD 커널에 여러분이 수정한 것을 적용하려면 이 라인의 주석을 푼다. 이 옵션은 *gcc(1)*의 디버깅 정보를 활성화하도록 *-g* 옵션을 사용한다. 예전 커널 빌드 과정을(*config; make depend; 등*) 사용한다면 *config(8)* *-g* 옵션으로

---

같은 결과를 얻을 수 있다.

**options                    MATH\_EMULATE                    #Support for x87 emulation**

위 라인은 컴퓨터에 코-프로세서가 없다면(386 이나 486SX) 커널이 코-프로세서를 시뮬레이트 할수 있다. 486DX, 386, 486SX(387 이나 487 칩으로 나누는) 또는 더 높은 장비를(펜티엄, 펜티엄 2 등) 가지고 있다면 이 라인을 주석처리 할수 있다.

**Note:** FreeBSD 가 가지고 있는 일반적인 계산 코-프로세서 에뮬레이터 루틴은 정확성이 떨어진다. 계산 코-프로세서를 가지고 있지 않고 정확성이 요구된다면 라이선스 문제로 기본적으로 포함하지 않고 GNU 계산 지원에 사용하는 *GPL\_MATH\_EMULATE*로 이 옵션을 변경한다.

오래된 CPU 는 기본적인 부동 소수점 계산이 일반적으로 지원되지 않고 대부분의 경우 다른 부가옵션 없이 GENERIC 커널이 지원하지 않기 때문에 FreeBSD 5.0 에서 계산 에뮬레이션은 보통 비활성되어 있다.

**options                    INET                    #InterNETworking**

네트워크 지원. 네트워크에 연결할 계획이 아니라면 그대로 둔다. 대부분의 프로그램은 최소한 루프백 네트워크가(예: PC 와 네트워크 연결을 하기 위해) 필요하기 때문에 위 옵션은 기본적으로 필요하다.

**options                    INET6                    #IPv6 communications protocols**

위 옵션은 IPv6 통신 프로토콜(현재 IP 의 다음 세대 버전을 의미한다)을 활성화한다.

**options                    FFS                    #Berkeley Fast Filesystem**  
**options                    FFS\_ROOT                    #FFS usable as root device [keep this!]**



위 옵션은 기본적인 하드 드라이브 파일시스템이다. 하드 디스크로 부팅한다면 그대로 둔다.

**Note:** FreeBSD 5.0 에서 FFS\_ROOT 는 더 이상 필요없다.

**options                    UFS\_ACL                    #Support for access control lists**

FreeBSD 5.0 에만 있는 이 옵션은 접근 제어 리스트를 커널이 지원하도록 한다. 위 옵션은 확장된 기능과 UFS2 를 사용하며 특징은 14.12 장에서 자세히 설명한다. ACL 이 기본적으로 활성화되므로 파일 시스템에 사용하고 있었다면 예상하지 못한 방법으로부터 파일을 보호하는 접근 제어 리스트를 커널에서 비활성하지 않는다.

**options                    UFS\_DIRHASH                #Improve performance on big directories**

이 옵션은 메모리를 추가적으로 사용하여 대형 디렉터리에서 디스크 속도를 올리는 기능을 가지고 있다. 일반적으로 대형 서버나 상호 작용하는 워크스테이션에 이 옵션을 사용하고 방화벽처럼 메모리가 중요하고 디스크 접근 속도는 중요하지 않은 작은 시스템에 FreeBSD 를 사용한다면 이 옵션을 삭제한다.

**options                    SOFTUPDATES                #Enable FFS Soft Updates support**

이 옵션은 커널에서 소프트 업데이트를 활성화하여 디스크 쓰기 속도를 증가시킨다. 소프트 업데이트는 기본적으로 4.X 분기에서 활성화됐지만 사용하지 않았을 것이다. 소프트 업데이트를 활성화했다면 mount(8) 출력에서 확인할 수 있다.

```
/dev/da0s1a on / (ufs, local)
/dev/da0s1f on /tmp (ufs, local, soft-updates)
/dev/da0s1g on /usr (ufs, local, soft-updates)
/dev/da0s1e on /var (ufs, local, soft-updates)
procfs on /proc (procfs, local)
```

soft-updates 옵션을 보지 못했다면 새로운 파일시스템에 tuneefs(8)이나 newfs(8)을 사용하여 활성화해야 된다.

---

options	MFS	#Memory Filesystem
options	MD_ROOT	#MD is a potential root device

이 옵션은 메모리 파일시스템이다. 이것은 임시 파일을 빠르게 저장하기 위한 것이고 기본적으로 RAM 디스크이다. 많은 스왑 공간을 가지고 있고 이와 같은 이점을 원한다면 유용하다. MFS 파티션을 마운트하는 완벽한 장소는 /tmp 디렉터리이다. 왜냐하면 수많은 프로그램이 이곳에 임시 데이터를 저장하기 때문이다. /tmp 에 MFS RAM 디스크를 마운트 하려면 다음 라인을 /etc/fstab 에 추가한다:

```
/dev/ad1s2b /tmp mfs rw 0 0
```

이제 간단히 재부팅 하거나 mount /tmp 명령을 실행한다.

**Note:** FreeBSD 5.X 에서 md(4)-backed UFS 파일시스템은 MFS 보다 메모리 파일 시스템을 위해 사용한다. MD-backed 파일시스템의 설정 정보는 mdconfig(8), mdmfs(8) 매뉴얼 페이지와 16.12 장에서 찾을 수 있다. 이런 이유로 MFS 옵션은 더 이상 지원되지 않는다.

options	NFS	#Network Filesystem
options	NFS_ROOT	#NFS usable as root device, NFS required

위 옵션은 네트워크 파일시스템을 의미한다. TCP/IP 로 유닉스 파일 서버에서 파티션을 마운트하지 않는다면 이들을 주석처리 한다.

options	MSDOSFS	#MSDOS Filesystem
---------	---------	-------------------

MS-DOS 파일 시스템이다. 부팅할때 DOS 포맷된 하드 드라이브를 마운트할 계획이 아니라면 이것을 주석처리 한다. 위에서 설명했듯이 DOS 파티션을 처음 마운트할때 자동으로 로드된다. 또한 뛰어난 **mttools** 소프트웨어(포트 컬렉션에서)로 마운트와 언마운트하지 않고 DOS 플로피를(*MSDOSFS* 는 필요 없이) 사용할 수 있다.

options	CD9660	#ISO 9660 Filesystem
options	CD9660_ROOT	#CD-ROM usable as root, CD9660 required

---

CDROM 을 위한 ISO 9660 파일 시스템이다. CDROM 드라이브를 가지고 있지 않거나 종 종 데이터 CD 만 마운트 한다면(왜냐하면 처음 데이터 CD 를 마운트할때 동기적으로 마운트 되기 때문이다) 주석처리 한다. 오디오 CD 는 이 파일시스템이 필요 없다.

**options**                    **PROCFS**                    **#Process filesystem**

프로세스 파일시스템이다. 이것은 어떤 프로세스가 실행 중인지 정보를 보여주는 ps(1) 같은 프로그램이 사용하는 /proc 에 마운트 한 것과 같다. 대부분의 디버깅과 모니터링 툴은 *PROCFS* 없이 실행될 수 있도록 수정하였기 때문에 FreeBSD 5.X 에서 *PROCFS* 은 대부분 사용할 필요가 없다. 게다가 5.X-CURRENT 커널은 *PROCFS* 를 사용하도록 했기 때문에 *PSEUDofs* 지원도 이제 포함해야 된다.

**options**                    **PSEUDofs**                    **#Pseudo-filesystem framework**

*PSEUDofs* 은 FreeBSD 4.X 에서 사용할 수 없다. FreeBSD 4.X 와 달리 FreeBSD 5.X 의 새로운 설치 는 기본적으로 프로세스 파일시스템을 마운트하지 않는다.

**options**                    **COMPAT\_43**                    **#Compatible with BSD 4.3 [KEEP THIS!]**

4.3 BSD 와 호환성을 위한 옵션이다. 이것을 그대로 둔다; 이곳을 주석처리 한다면 어떤 프로그램은 이상하게 동작할 것이다.

**options**                    **COMPAT\_FREEBSD4**                    **#Compatible with FreeBSD4**

위 옵션은 FreeBSD 5.X i386 과 alpha 시스템에서 예전 시스템 콜 인터페이스를 사용하는 이전 버전의 FreeBSD 에 설치된 어플리케이션 지원에 필요하다. 이 옵션은 예전 어플리케이션을 사용하는 모든 i386 과 alpha 시스템에 사용하도록 권장한다; ia64 와 Sparc64 같은 5.X 에서만 지원하는 플랫폼은 이 옵션이 필요 없다.

**options**                    **SCSI\_DELAY=15000**                    **#Delay (in ms) before probing SCSI**

---

이 옵션 때문에 커널이 각 SCSI 장치를 탐색하기 전에 15 초 동안 잠시 멈춘다. IDE 하드 드라이브만 가지고 있다면 이 옵션을 무시하고 그렇지 않고 부팅 속도를 높이기 위해 5 초 정도로 낮은 숫자를 입력할 수 있다. 물론 이렇게 설정해 보고 FreeBSD 가 SCSI 장치 인지에 문제가 있다면 숫자를 늘린다.

**options**                    **UCONSOLE**                    **#Allow users to grab the console**

X 유저에게 유용하도록 유저가 콘솔을 볼 수 있게 한다. 예를 들면 커널이 보내는 다른 콘솔 메시지처럼 write(1)와 talk(1) 그리고 여러분이 받은 다른 메시지를 보여줄 콘솔 **xterm** 을 **xterm -c** 를 입력하여 생성할 수 있다.

**Note:** FreeBSD 5.X 에서 *UCONSOLE* 은 더 이상 필요 없다.

**options**                    **USERCONFIG**                    **#boot -c editor**

위 옵션은 부트 메뉴에서 부팅 설정을 편집할 수 있다.

**options**                    **VISUAL\_USERCONFIG**                    **#visual boot -c editor**

이 옵션은 부트 메뉴에서 부팅 설정을 비주얼 설정 에디터로 편집할 수 있다.

**Note:** FreeBSD 버전 5.0 과 이후 버전에서 *USERCONFIG* 옵션이 새로운 device.hints(5)의 이점을 감소 시켰다. device.hints(5)에 대한 더 많은 정보는 12.5 장을 확인한다.

**options**                    **KTRACE**                    **#ktrace(1) support**

위 옵션은 디버깅할때 유용한 커널 프로세스 추적을 활성화한다.

**options**                    **SYSVSHM**                    **#SYSV-style shared memory**

---

이 옵션은 System V 공유 메모리를 지원한다. 가장 일반적인 사용은 X에서 수많은 그래픽 집약적인 프로그램이 자동으로 속도를 증가 시키는 XSHM 확장이다. X를 사용한다면 이 옵션을 사용해야 된다.

**options**                    **SYSVSEM**                    **#SYSV-style semaphores**

System V 세마포어 지원. 자주 사용되지 않지만 커널에 몇 백 바이트만 추가된다.

**options**                    **SYSVMSG**                    **#SYSV-style message queues**

System V 메시지 지원. 역시 커널에 몇 백 바이트만 추가된다.

**Note:** `ipcs(1)` 명령은 이들 System V 특성을 사용하는 모든 프로세스를 리스트한다.

**options**    **P1003\_1B**                    **#Posix P1003\_1B real-time extensions**  
**options**    **\_KPOSIX\_PRIORITY\_SCHEDULING**

1993 POSIX 에 추가된 실시간 확장. 포트 컬렉션에서 특정 어플리케이션이(스타오피스 같은) 이 기능을 사용한다.

**Note:** FreeBSD 5.X 에서 이 기능의 모든 것이 `_KPOSIX_PRIORITY_SCHEDULING` 옵션으로 지원 되기 때문에 `P1003_1B`는 더 이상 필요 없다.

**options**                    **ICMP\_BANDLIM**                    **#Rate limit bad replies**

이 옵션은 ICMP 응답 에러의 대역폭을 제한한다. 서비스 거부 패킷 공격으로부터 머신을 보호하기 때문에 보통 이 옵션을 사용한다.

**Note:** FreeBSD 5.X 에서 이 기능은 기본적으로 활성화 되어있기 때문에

---

*ICMP\_BANDLIM* 옵션은 필요 없다.

**# To make an SMP kernel, the next two are needed**

**#options SMP # Symmetric MultiProcessor Kernel**

**#options APIC\_IO # Symmetric (APIC) I/O**

위의 두 가지는 SMP(멀티 CPU) 지원에 필요하다.

**device isa**

FreeBSD 가 지원하는 모든 PC 는 ISA 장치를 갖고 있다. IBM PS/2(마이크로 채널 아키텍처)를 가지고 있다면 FreeBSD 는 지금 약간 제한된 지원만 가능하다. MCA 지원에 대한 더 많은 정보는 /usr/src/sys/i386/conf/LINT 를 본다.

**device eisa**

EISA 마더보드를 가지고 있다면 이 옵션을 활성화한다. 이것은 EISA 버스의 모든 장치를 자동으로 탐색하고 설정지원을 활성화한다.

**device pci**

PCI 마더보드를 가지고 있다면 이 옵션을 활성화한다. 이것은 PCI 카드와 PCI 부터 ISA 버스에 이르는 게이트웨이 자동 검색을 활성화한다.

**device agp**

시스템에 AGP 카드가 있다면 이 옵션을 활성화한다. 이것은 AGP 와 이런 기능을 가진 보드의 AGP GART 지원을 활성화한다.

**# Floppy drives**

---

```
device      fdc0      at isa? port IO_FD1 irq 6 drq 2
device      fd0      at fdc0 drive 0
device      fd1      at fdc0 drive 1
```

이 옵션들은 플로피 드라이브 컨트롤러다. *fd0*는 플로피 A: 드라이브를 *fd1*은 B: 드라이브다.

```
device      ata
```

이 드라이버는 모든 ATA와 ATAPI 드라이브를 지원한다. 현대 머신에서는 모든 PCI ATA/ATAPI 장치를 커널이 탐색하도록 *device ata* 라인만 필요하다.

```
device      atadisk      # ATA disk drives
```

위 옵션은 ATA 디스크 드라이브를 위해 *device ata*가 함께 필요하다.

```
device      atapicd      # ATAPI CDROM drives
```

위 옵션은 ATAPI CDROM 드라이브를 위해 *device ata*가 필요하다.

```
device      atapifd      # ATAPI floppy drives
```

이 옵션도 ATAPI 플로피 드라이브를 위해 *device ata*가 필요하다.

```
device      atapist      # ATAPI tape drives
```

이 옵션도 ATAPI 테이프 드라이브를 위해 *device ata*가 필요하다.

```
options     ATA_STATIC_ID      #Static device numbering
```

---

이 옵션은 컨트롤러 번호를 정적으로(예전 드라이버 같은) 만들거나 장치 번호를 동적으로 할당한다.

**# SCSI Controllers**

```
device      ahb      # EISA AHA1742 family
device      ahc      # AHA2940 and onboard AIC7xxx devices
device      amd      # AMD 53C974 (Teckram DC-390(T))
device      dpt      # DPT Smartcache - See LINT for options!
device      isp      # Qlogic family
device      ncr      # NCR/Symbios Logic
device      sym      # NCR/Symbios Logic (newer chipsets)

device      adv0     at isa?
device      adw
device      bt0      at isa?
device      aha0     at isa?
device      aic0     at isa?
```

SCSI 컨트롤러. 시스템에 없는 것은 주석처리 한다. 시스템에 IDE 만 있다면 모든 내용을 삭제할 수 있다.

**# SCSI peripherals**

```
device      scbus    # SCSI bus (required)
device      da       # Direct Access (disks)
device      sa       # Sequential Access (tape etc)
device      cd       # CD
device      pass     # Passthrough device (direct SCSI
access)
```

SCSI 주변장치. 역시 IDE 하드웨어만 있다면 모두 주석 처리하거나 SCSI 주변장치를 삭제할 수 있다.



---

**# RAID controllers**

```
device      ida      # Compaq Smart RAID
device      amr      # AMI MegaRAID
device      mlx      # Mylex DAC960 family
```

지원되는 RAID 컨트롤러. RAID 컨트롤러를 사용하지 않는다면 이들을 삭제할 수 있다.

**# atkbdc0 controls both the keyboard and the PS/2 mouse**

```
device      atkbdc0   at isa? port IO_KBD
```

AT 키보드와 PS/2 스타일의 입력 장치에 I/O 서비스를 제공하는 키보드 컨트롤러. 이 컨트롤러는 키보드 드라이버(*atkbd*)와 PS/2 스타일 입력 장치 드라이버(*psm*)에 필요하다.

```
device      atkbd0    at atkbdc? irq 1
```

*atkbdc* 컨트롤러와 *atkbd* 드라이버는 AT 84 키보드나 AT 키보드 컨트롤러에 연결된 발전된 AT 키보드를 사용하게 한다.

```
device      psm0      at atkbdc? irq 12
```

마우스가 PS/2 마우스 포트에 꽂혀 있다면 이 장치를 사용한다.

```
device      vga0      at isa?
```

비디오 카드 드라이버다.

**# splash screen/screen saver**

```
device      splash
```

시작할때 스플래시 화면(스플래시 화면을 등록해두면 부팅하는 동안 등록해둔 스플래시 이미지가 화면에 나타난다). 이 옵션도 화면 보호기가 필요하다. FreeBSD 4.X 에는

---

*pseudo-device splash* 라인을 사용한다.

```
# syscons is the default console driver, resembling an SCO console
device          sc0          at isa?
```

SCO 콘솔과 닮은 *sc0*는 기본 콘솔 드라이버다. 왜냐하면 대부분의 전체 화면 프로그램은 *termcap* 같은 터미널 데이터베이스 라이브러리를 거쳐 콘솔에 접근하기 때문에 이것을 사용하거나 *VT220* 콘솔 드라이버와 호환되는 *vt0*를 사용하던지 문제가 되지 않는다. 로그인 했을때 이 콘솔에서 전체 화면 프로그램을 사용하는데 문제가 있다면 *TERM* 변수를 *scoansi*로 설정한다.

```
# Enable this and PCVT_FREEBSD for pcvt vt220 compatible console driver
#device          vt0          at isa?
#options         XSERVER      # support for X server on a vt console
#options         FAT_CURSOR   # start with block cursor
# If you have a ThinkPAD, uncomment this along with the rest of the PCVT lines
#options         PCVT_SCANSET=2 # IBM keyboards are non-std
```

위 옵션들은 거꾸로 *VT100/102*와 호환되는 *VT220* 호환 콘솔 드라이버다. *sc0*와 호환되지 않는 하드웨어를 가진 어떤 노트북은 제대로 작동한다. 또한 로그인할때 *TERM* 변수를 *vt100*이나 *vt220*으로 변경한다. 이 드라이버는 가끔 *sc0* 장치의 *termcap*이나 *terminfo* 엔트리를 이용할 수 없는 네트워크를 통해 다른 여러 머신에 연결할때 유용할 것이다. -- *vt100*은 사실상 모든 플랫폼에서 사용할 수 있어야 한다.

```
# Power management support (see LINT for more options)
device          apm0          at nexus? disable flags 0x20 # Advanced Power
Management
```

노트북에 유용한 발전된 전원관리를 지원한다.

```
# PCCARD (PCMCIA) support
device          card
```

```
device      pcic0    at isa? irq 10 port 0x3e0 iomem 0xd0000
device      pcic1    at isa? irq 11 port 0x3e2 iomem 0xd4000 disable
```

PCMCIA 지원. 노트북을 사용한다면 이 옵션들을 사용해야 될 것이다.

#### # Serial (COM) ports

```
device      sio0     at isa? port IO_COM1 flags 0x10 irq 4
device      sio1     at isa? port IO_COM2 irq 3
device      sio2     at isa? disable port IO_COM3 irq 5
device      sio3     at isa? disable port IO_COM4 irq 9
```

이들은 MS-DOS/윈도우 에서 COM1 에서 COM4 로 부르는 4 개의 시리얼포트다.

**Note:** 내부 모뎀을 COM4 와 시리얼 포트 COM2 에 가지고 있다면 FreeBSD 에서 모뎀을 사용하기 위해 모뎀 IRQ 를 2 번으로(확실치 않지만 기술적으로 IRQ2=IRQ9) 변경해야 된다. 여러 개의 시리얼 카드를 가지고 있다면 이들 라인에 적당한 값에 대한 더 많은 정보는 sio(4) 매뉴얼 페이지를 참고한다. 0x\*2e8 형식의 IO 주소를 사용하는 어떤 비디오 카드(S3 칩에 기반한)와 가격이 싼 여러 시리얼 카드는 16 비트 IO 주소로 정확히 번역되지 않기 때문에 이러한 카드와 충돌하여 COM4 포트를 사실상 사용하지 못하게 한다.

각 시리얼 포트는 유일한 IRQ 가 필요하기 때문에(공유 인터럽트가 지원되는 멀티포트 카드 중 하나라도 사용하지 않는다면) 기본적으로 COM3 와 COM4 는 사용할 수 없다.

#### # Parallel port

```
device      ppc0     at isa? irq 7
```

이것은 ISA 버스 패러럴 포트 인터페이스다.

```
device      ppbus    # Parallel port bus (required)
```

패러럴 포트 버스를 지원한다.

---

```
device      lpt      # Printer
```

패러럴 포트 프린터를 지원한다.

**Note:** 패러럴 프린터 지원에 위의 3 가지가 모두 필요하다.

```
device      plip     # TCP/IP over parallel
```

이 옵션은 패러럴 네트워크 인터페이스 드라이버다.

```
device      ppi      # Parallel port interface device
```

일반적인 목적의 I/O("geek port")+IEEE1284 I/O.

```
#device     vpo      # Requires scbus and da
```

이것은 lomega ZIP 드라이브를 위한 것이다. *scbus* 와 *da* 지원이 필요하다. EPP 1.9 모드에서 최대 성능을 발휘한다.

```
# PCI Ethernet NICs.
```

```
device      de       # DEC/Intel DC21x4x ("Tulip")
```

```
device      fxp     # Intel EtherExpress PRO/100B (82557, 82558)
```

```
device      tx      # SMC 9432TX (83c170 "EPIC")
```

```
device      vx      # 3Com 3c590, 3c595 ("Vortex")
```

```
device      wx      # Intel Gigabit Ethernet Card ("Wiseman")
```

다양한 PCI 네트워크 카드. 이들 중 시스템에 없는 것은 모두 주석 처리한다.

```
# PCI Ethernet NICs that use the common MII bus controller code.
```

---

```
device      miibus      # MII bus support
```

MII 버스 지원은 어떤 PCI 10/100 이더넷 네트워크 카드에 필요하다. 즉 MII-컴플라이언트 트랜시버를 사용하거나 MII 처럼 작동하는 트랜시버 컨트롤 인터페이스다. *device miibus* 를 커널 설정에 추가하면 일반 miibus API 와 특히 개별적인 드라이버로 제어하지 못하는 일반적인 PHY 를 포함하여 모든 PHY 드라이버 지원을 얻게 된다.

```
device      dc          # DEC/Intel 21143 and various workalikes
device      rl          # RealTek 8129/8139
device      sf          # Adaptec AIC-6915 ("Starfire")
device      sis         # Silicon Integrated Systems SiS 900/SiS 7016
device      ste         # Sundance ST201 (D-Link DFE-550TX)
device      tl          # Texas Instruments ThunderLAN
device      vr          # VIA Rhine, Rhine II
device      wb          # Winbond W89C840F
device      xl          # 3Com 3c90x ("Boomerang", "Cyclone")
```

MII 버스 제어 코드를 사용하는 드라이버다.

자신의 네트워크 카드가 어떤 것인지 모른다면 dmesg 로 검색된 드라이버를 찾아본다.

```
isab1: <PCI to ISA bridge (vendor=1106 device=3057)> at device
xl0: <3Com 3c905C-TX Fast Etherlink XL> port 0x1400-0x147f mem
xl0: Ethernet address: 00:01:03:39:00:2d
miibus0: <MII bus> on xl0
ukphy0: <Generic IEEE 802.3u media interface> on miibus0
ukphy0: 10baseT, 10baseT-FDX, 100baseTX, 100baseTX-FDX, auto
```

```
# ISA Ethernet NICs.
```

```
device      ed0      at isa? port 0x280 irq 10 iomem 0xd8000
```

```
device      ex
```

```
device      ep
```

```
# WaveLAN/IEEE 802.11 wireless NICs. Note: the WaveLAN/IEEE really
# exists only as a PCMCIA device, so there is no ISA attachment needed
# and resources will always be dynamically assigned by the pccard code.
```

```
device      wi
```

```

# Aironet 4500/4800 802.11 wireless NICs. Note: the declaration below will
# work for PCMCIA and PCI cards, as well as ISA cards set to ISA PnP
# mode (the factory default). If you set the switches on your ISA
# card for a manually chosen I/O address and IRQ, you must specify
# those parameters here.
device      an
# The probe order of these is presently determined by i386/isa/isa_compat.c.
device      ie0    at isa? port 0x300 irq 10 iomem 0xd0000
device      fe0    at isa? port 0x300
device      le0    at isa? port 0x300 irq 5 iomem 0xd0000
device      lnc0   at isa? port 0x280 irq 10 drq 0
device      cs0    at isa? port 0x300
device      sn0    at isa? port 0x300 irq 10
# requires PCCARD (PCMCIA) support to be activated
#device     xe0    at isa?

```

ISA 이더넷 드라이버. 어떤 드라이버가 어떤 카드를 지원하는지  
/usr/src/sys/i386/conf/LINT 를 본다.

```

device ether      # Ethernet support

```

이더넷 카드를 가지고 있다면 *ether* 만 필요하다. 이것은 일반적인 이더넷 프로토콜 코드를 포함한다. FreeBSD 4.X 에서는 *pseudo-device ether* 라인을 사용한다

```

device sl        1    # Kernel SLIP

```

*s/*은 SLIP 를 지원한다. 이것은 대부분 설정하기 쉽고 모뎀과 모뎀 연결에 적합하고 더 강력한 PPP 로 대체되고 있다. *sl* 다음 번호는 얼마나 많은 SLIP 세션을 동시에 지원할지 지정한다. FreeBSD 4.X 에서는 *pseudo-device sl* 라인을 사용한다.

```

device ppp       1    # Kernel PPP

```

이 옵션은 전화 연결을 위해 커널 PPP 를 지원한다. 또한 *tun* 을 사용하는 유저기반 어플

---

리케이션과 전화 걸기 요청처럼 더 많은 유연성과 기능을 제공하는 PPP 버전도 있다. *ppp* 다음의 숫자는 PPP 연결을 동시에 얼마나 지원 할 것인가를 지정한다. FreeBSD 4.X에서는 *pseudo-device ppp*를 사용한다.

```
device tun          # Packet tunnel.
```

이 옵션은 유저기반 PPP 소프트웨어에 사용된다. *tun* 다음의 숫자는 지원할 동시 PPP 세션의 수를 지정한다. 더 많은 정보는 이 책의 PPP 섹션을 본다. FreeBSD 4.X에서는 *pseudo-device tun*을 사용한다.

```
device pty          # Pseudo-ttys (telnet etc)
```

이것은 "pseudo-terminal" 또는 시뮬레이트된 로그인 포트다. 이 옵션은 telnet 과 rlogin 요청 세션, **xterm** 과 **Emacs** 같은 어플리케이션들이 사용한다. *pty* 다음 숫자는 생성하려는 *ptys* 번호를 보여준다. **xterm** 윈도우나 원격 로그인이 동시에 기본값 16 보다 더 필요하다면 최대 256 까지 숫자를 증가시킬 수 있다.

```
device md           # Memory ``disks''
```

메모리 디스크 *pseudo-devices*. FreeBSD 4.X에서는 *pseudo-device md* 라인을 사용한다

```
device gif          # IPv6 and IPv4 tunneling
```

이 옵션은 IPv6 를 IPv4 로 터널링, IPv4 를 IPv6 로 터널링, IPv4 를 IPv4 로 터널링 그리고 IPv6 를 IPv6 로 터널링 한다. FreeBSD 4.4 가 시작되면서 *gif* 장치는 자동으로 복제되었기 때문에 *pseudo-device gif* 라인을 사용한다. 이전 FreeBSD 버전은 *pseudo-device gif 4* 처럼 숫자가 필요하다.

```
device faith        # IPv6-to-IPv4 relaying (translation)
```

---

이 `pseudo-device` 는 패킷을 IPv4/IPv6 변환해서 데몬에 보내고 변환되는 패킷을 캡처한다. FreeBSD 4.X 에서는 `pseudo-device faith 1` 라인을 사용한다.

```
# The `bpf' pseudo-device enables the Berkeley Packet Filter.  
# Be aware of the administrative consequences of enabling this!  
device bpf # Berkeley packet filter
```

이것은 버클리 패킷 필터다. 이 `pseudo-device` 는 네트워크 인터페이스를 무작위 (promiscuous) 모드로 변환하고 브로드 캐스트 네트워크에서 (예: 이더넷) 모든 패킷을 캡처 한다. 이러한 패킷은 캡처되어 디스크에 저장되거나 `tcpdump(1)` 프로그램으로 분석 된다. FreeBSD 4.X 에서는 `pseudo-device bpf` 라인을 사용한다.

**Note:** `bpf(4)` 장치도 기본 라우터 IP 주소 등을 가져오는 `dhclient(8)`가 사용한다. DHCP 를 사용한다면 이곳을 주석처리 하지 않는다.

```
# USB support  
#device uhci # UHCI PCI->USB interface  
#device ohci # OHCI PCI->USB interface  
#device usb # USB Bus (required)  
#device ugen # Generic  
#device uhid # ``Human Interface Devices''  
#device ukbd # Keyboard  
#device ulpt # Printer  
#device umass # Disks/Mass storage - Requires scbus and da  
#device ums # Mouse  
# USB Ethernet, requires mii  
#device aue # ADMtek USB ethernet  
#device cue # CATC USB ethernet  
#device kue # Kawasaki LSI USB ethernet
```

다양한 USB 장치(USB 키보드, 마우스, 프린터 등) 지원.

더 많은 정보와 FreeBSD 가 지원하는 추가적인 장치는 `/usr/src/sys/i386/conf/LINT` 를 본다.



---

## 8.4.1 대형 메모리 설정(PAE)

대형 메모리를 설정한 머신은 유저 + 커널 가상 주소에서 4기가 바이트 제한 이상을 접근해야 된다. 이 제한 때문에 Intel은 Pentium Pro와 이 후의 CPU 생산 라인에 36-bit 물리적인 주소 공간을 사용하도록 지원을 추가했다.

Intel Pentium Pro와 이후 CPU의 물리적 주소 확장(PAE) 기능은 64기가 바이트까지 메모리를 설정할 수 있게 한다. FreeBSD는 4.X 시리즈의 FreeBSD 4.9-RELEASE와 5.X 시리즈의 FreeBSD 5.1-RELEASE에서 사용할 수 있는 PAE 커널 설정 옵션으로 이 기능을 지원한다. 따라서 Intel 메모리 아키텍처의 한계는 4기가 이상이나 미만까지는 차이가 없다. 4기가 바이트 이상의 메모리 할당은 단순히 가능한 메모리 풀(pool)을 추가한다.

커널에서 PAE 지원을 활성화하려면 커널 설정파일에 단순히 다음 라인을 추가한다:

**options**            **PAE**

**Note:** FreeBSD에서 PAE 지원은 Intel IA-32 프로세서에서만 사용할 수 있다. 또한 PAE 지원은 다양하게 테스트 되지 않아서 FreeBSD의 안정된 다른 기능보다 안정성이 떨어진다는 것을 감안해야 된다.

FreeBSD에서 PAE 지원은 약간의 한계가 있다:

- 프로세스 하나가 4기가 바이트 이상의 VM 공간을 사용할 수 없다.
- KLD 모듈은 PAE가 활성화된 커널에 로드할 수 없기 때문에 모듈과 커널의 빌드 프레임워크는 달라진다.
- bus\_dma(9) 인터페이스를 사용하지 않는 장치 드라이버는 PAE가 활성화된 커널에서 데이터 무결성을 깨뜨리기 때문에 사용을 권장하지 않는다. 이런 이유로 PAE 커널 설정 파일은 PAE가 활성화된 커널에서 동작할 수 없는 모든 드라이버를 배제한 FreeBSD 5.X에서 지원된다.

- 
- 어떤 시스템은 사용할 수 있는 물리적인 메모리 양에 따라 메모리 자원 사용량 한계를 튜닝 할 수 있다. 이런 튜닝은 PAE 시스템의 대형 메모리에 따라 더 많이 할당하지 않는다면 필요 없을 수 있다. 이런 예제 하나는 커널에서 최대의 vnode를 제어하게 하는 `kern.maxvnodes sysctl`이다. 이 옵션과 튜닝 할 수 있는 다른 것들을 적절한 값으로 조정하기 바란다.
  - 커널 가상 주소(KVA) 공간을 증가 시키거나 KVA 소모를 방지하기 위해 대량으로 사용되는 특정 커널 자원의 양을 줄일 필요가 있을 것이다. KVA 공간을 증가시키기 위해 `KVA_PAGES` 커널 옵션을 사용할 수 있다.

성능과 안정성을 위해 `tuning(7)` 매뉴얼 페이지를 참고 한다. `pae(4)` 매뉴얼 페이지는 FreeBSD PAE 지원에 대한 최신 정보를 가지고 있다.

## 8.5 장치 노드 생성

**Note:** FreeBSD 5.0 이나 이 후 버전을 사용한다면 이번 섹션을 지나쳐도 된다. 이들 버전은 장치 노드 할당에 `devfs(5)`를 사용한다.

커널의 거의 모든 장치는 `/dev` 디렉터리에 매칭되는 "node" 엔트리를 가지고 있다. 이들 노드는 일반적인 파일처럼 보이지만 실제로 어떤 프로그램이 장치에 접근할 수 있는지와 관련된 특별한 엔트리가 커널에 입력되어있다. 처음 운영체제를 설치 했을때 실행되는 셸 스크립트 `/dev/MAKEDEV`가 지원되는 거의 모든 장치 노드를 생성한다. 그러나 이것은 모든 노드를 생성하지 못하기 때문에 새로운 장치를 지원하도록 추가할때 적절한 엔트리가 이 디렉터리에 있는지 확인하고 없다면 추가한다. 여기 간단한 예제가 있다.

커널이 IDE CD-ROM 을 지원하도록 하려면 설정파일에 다음 라인을 추가한다.

```
device acd0
```

이 의미는 `/dev` 디렉터리에서 `acd0` 로 시작되고 아마 `c` 같은 문자가 따라오거나 "raw" 장치를 의미하는 `r`이 먼저 나오는 어떤 엔트리를 찾아야 된다. 이들 파일이 없어서 기능이 비활성되어 있으므로 `/dev` 디렉터리로 이동 후 다음을 입력해야 된다.

```
# sh MAKEDEV acd0
```

---

이 스크립트가 끝나면 `acd0c` 와 `racd0c` 엔트리가 새로 추가된 것을 `/dev` 에서 찾을 수 있고 정상적으로 작동하는 것을 확인할 수 있다.

사운드 카드를 위해 다음 명령이 적당한 엔트리를 생성한다:

```
# sh MAKEDEV snd0
```

**Note:** 사운드 카드와 같은 장치 노드를 생성할때 다른 사람이 머신을 사용하면 이 사용자들을 `/etc/fstab` 파일에 추가해서 외부에서 장치를 사용하지 못하도록 하는것이 바람직할 것이다. `fstab(5)`에서 더 많은 정보를 참고한다.

엔트리를 없고 `GENERIC` 장치가 아닌 것들은 이 절차를 따른다.

**Note:** 모든 SCSI 컨트롤러는 `/dev` 엔트리의 같은 세트를 사용하기 때문에 장치 노드를 만들 필요가 없다. 또한 네트워크 카드와 `SLIP/PPP pseudo-devices` 도 `/dev` 에 엔트리를 가지고 있지 않기 때문에 이에 대해 걱정할 필요가 없다.

## 8.6 문제가 있다면

사용자 커널을 빌드할때 5 종류의 문제가 발생할 수 있다. 이들은 다음과 같다:

### config 실패:

커널에 지시를 하였을때 `config(8)` 명령이 실패했다면 아마도 어딘가에서 단순한 실수가 있었을 것이다. 다행히 `config(8)`은 문제가 되는 라인 번호를 출력하기 때문에 `vi` 에서 빨리 찾을 수 있다. 예를들어 다음 메시지를 보게 되면

```
config: line 17: syntax error
```

`vi` 의 명령어 모드에서 `17G` 를 입력하여 바로 문제가 되는 부분으로 갈수 있다. 물론 키워드를 입력하는 것은 `GENERIC` 커널이나 다른 레퍼런스와 비교하여 정확히 입력한다.

---

### make 실패:

make 명령이 실패했다면 보통 커널이 에러를 알려주지만 config(8)가 문제를 파악하기에 충분치 않다. 다시 설정을 확인하고 그래도 문제를 해결하지 못했다면 FreeBSD 일반 질문 메일링 리스트 (<http://lists.freebsd.org/mailman/listinfo/freebsd-questions>)에 빨리 파악될 수 있도록 커널 설정과 함께 보낸다.

### 새로운 커널 설치 실패:

커널 컴파일이 양호하지만 설치에(make install 이나 make installkernel 명령 실패) 실패했다면 첫째로 시스템의 보안 레벨이 1 이나 더 높은 값인지(init(8)) 체크한다. 커널 설치시 커널에서 변경 불능 플래그를 삭제하고 새로운 커널에 변경불능 플래그를 설정한다. 왜냐하면 보안 레벨이 1 이나 더 높은 경우 시스템에서 변경불능 플래그를 가지고 있는 파일의 설정 해제를 방지하기 때문에 커널 설치시 보안레벨 0 이나 더 낮은 상태에서 수행해야 된다.

### 커널이 부팅하지 않을 때:

패닉이 아니고 새로운 커널이 부팅하지 않거나 장치 탐색에 실패했다면 다행히 FreeBSD 는 호환되지 않는 커널로부터 복구하는 완벽한 메커니즘을 가지고 있다. 간단히 FreeBSD 부트 로더에서 원하는 커널을 선택한다. 시스템 카운트가 10 부터 작아질때 커널을 선택한다. Enter 키를 제외한 키를 누른 후 unload 를 입력하고 *boot kernel.old*나 정확히 부팅할 수 있는 다른 커널 파일 이름을 입력한다. 커널을 다시 설정할때 기존 커널을 가지고 있는 것은 좋은 생각이다.

적당한 커널로 부팅 후 설정 파일을 체크하고 다시 빌드 할수 있다. 도움이 될만한 내용은 성공적인 부팅에서 모든 커널 메시지만 저장하는 /var/log/messages 파일이다. dmesg(8) 명령으로도 현재 부팅한 커널 메시지를 출력할 수 있다.

**Note:** 커널 빌드에 문제가 있다면 GENERIC 이나 다음에 빌드할때 삭제되지 않도록 다른 이름으로 변경해둔 정상적으로 동작하는 커널을 가지고 있어야 한다. 새로운 커널을 설치할때 kernel.old 는 정상적으로 동작하지 않는 마지막에 설치

---

된 커널로 덮어쓰기 때문에 믿을 수 없다. 게다가 가능한 빨리 동작중인 커널을 적당한 kernel 위치로 이동시키지 않으면 ps(1) 같은 명령은 정확히 동작하지 않는다. make 로 설치하는 (다른 커널을 영구히 이동시키기 위해) 커널 파일의 잠금을 해제하는 적당한 명령은 다음과 같다.

```
# chflags noschg /kernel
```

이렇게 할수 없다면 securelevel(8)이 레벨 0 보다 높은 상태일 것이다. /etc/rc.conf 에서 kern\_securelevel 를 -1 로 설정하고 재부팅 한다. 새로운 커널이 만족스러울때 securelevel(8)을 기존 설정으로 바꿀 수 있다.

그리고 새로운 커널이나 문제되는 파일을 잠궈서 삭제되거나 잘못되는 것을 방지하려면 다음 명령을 실행한다.

```
# chflags schg /kernel
```

FreeBSD 5.X 에서 커널은 시스템의 변경 불능 플래그가 설정되지 않기 때문에 문제의 소지가 된다.

#### 커널은 작동하지만 ps(1)는 작동하지 않는다:

예를 들어 3.X 시스템에 4.X 커널의 경우처럼 시스템 유틸리티가 빌드한 것과 다른 버전의 커널을 설치하였다면 ps(1)와 vmstat(8) 같은 수 많은 시스템 상태 명령이 동작하지 않는다. 이런 유틸리티와 libkvm 라이브러리는 다시 컴파일 해야 된다. 운영체제의 나머지 부분과 다른 버전의 커널을 사용하는 것은 좋은 생각이 아니다.

---

## 9 장 프린트

### 9.1 개요

FreeBSD 는 오래된 도트 프린터부터 최신의 레이저 프린터까지 다양한 프린터를 사용할 수 있고 어플리케이션을 실행하여 고품질의 내용을 출력할 수 있다.

FreeBSD 를 네트워크 프린터 서버로도 설정할 수 있다; FreeBSD 는 이런 기능으로 다른 FreeBSD 컴퓨터를 포함하여 윈도우와 MacOS 호스트까지 다양한 컴퓨터로부터 출력요청을 처리할 수 있다. FreeBSD 는 내용을 출력해서 어떤 유저와 머신이 가장 많이 출력 했는지 통계를 낼수 있고 누구의 출력물인지 보여주는 "배너" 페이지를 만들 수 있다.

이번 장을 읽고 다음과 같은 사항을 알수 있다:

- FreeBSD 에서 프린트 스플러는 어떻게 설정하는가
- 입력된 문서를 프린터가 이해하는 출력포맷으로 변환하는 것을 포함하여 특수한 출력파일을 제어하는 프린터필터를 어떻게 설치하는가
- 헤더페이지나 출력물에서 배너페이지는 어떻게 활성화하는가
- 다른 컴퓨터에 연결된 프린터를 어떻게 사용하는가
- 네트워크에 직접 연결된 프린터를 어떻게 사용하는가
- 출력할 파일크기 제한을 포함하여 프린터제한을 설정하고 특정 유저의 프린터 사용을 어떻게 제어하는가
- 프린터 사용통계와 사용량을 어떻게 계산하는가
- 출력문제는 어떻게 해결하는가

이번 장을 읽기 전에 다음 사항을 알고 있어야 한다:

- 
- 새로운 커널을 어떻게 설치하고 설정하는가(8 장)

## 9.2 소개

FreeBSD 로 프린터를 사용하려면 LPD 스푼링 시스템이라고 하는 버클리 라인프린터 스푼링 시스템으로 동작하도록 프린터를 설정해야 된다. 이 설정이 FreeBSD 에서 표준 프린터 제어시스템이다. 이번 장에서는 간단히 LPD 라고 하는 LPD 스푼링 시스템에 대해 소개하고 설정할 수 있도록 안내한다.

이미 LPD 나 다른 프린터 스푼링 시스템을 잘 알고 있다면 이번 섹션을 지나서 스푼링 시스템 설정으로 넘어가도 된다.

LPD 는 호스트의 프린터에 대한 모든 것을 제어한다. 다음과 같은 사항을 포함하고 있다:

- 직접 연결되어 있는 프린터와 네트워크의 다른 호스트에 연결되어 있는 프린터 사용 제어.
- 유저가 파일을 출력할 수 있도록 한다; 이러한 작업을 *잡*이라고 한다.
- 각 프린터 *큐*를 관리하여 여러 유저가 동시에 프린터를 사용하는 것을 방지한다.
- *헤더페이지*를(*배너* 또는 *burst* 페이지로 알려진) 출력하기 때문에 출력물 중에서 유저는 자신의 잡을 쉽게 찾을 수 있다.
- 시리얼 포트에 연결된 프린터의 통신 매개변수 관리.
- 네트워크를 통해 다른 호스트의 LPD 스푼러에 잡을 보낼 수 있다.
- 다양한 프린터 언어 또는 프린터 특성에 맞게 출력 잡을 포맷하는 특수필터를 실행할 수 있다.
- 프린터 사용량을 계산할 수 있다.

---

설정 파일과(/etc/printcap) 특별한 필터 프로그램을 제공하여 다양한 프린터 하드웨어가 위의 모든 기능 또는 필요한 기능을 갖추도록 LPD 시스템을 설정할 수 있다.

## 9.2.1 왜 스플러를 사용하는가

시스템의 유일한 유저라면 접근 제어, 헤더페이지 또는 프린터 계산 등이 필요 없는데 왜 스플러로 귀찮게 하는지 궁금 할 것이다. 프린터를 직접 사용하는 것도 가능하지만 다음과 같은 이유로 스플러를 사용하는 것이 좋다:

- 백그라운드에서 LPD 프린트 잡을 수행한다; 프린터에 데이터가 복사될 때까지 기다릴 필요가 없다.
- LPD 는 필터를 통해 날짜/시간을 헤더에 추가하거나 프린터가 이해하는 특수파일 포맷으로(TeX DVI 파일 같은) 쉽게 변환할 수 있다. 이런 단계를 직접 수행할 필요는 없다.
- 시스템에서 스플러와 통신하여 출력기능을 제공하는 수많은 무료 또는 상업용 프로그램이 있다. 스플링 시스템을 설정하여 나중에 추가하거나 사용하고 있는 다른 소프트웨어를 쉽게 지원할 수 있다.

## 9.3 기본 설정

LPD 스플링 시스템으로 프린터를 사용하려면 프린터 하드웨어와 LPD 소프트웨어를 설정해야 된다. 이 문서에서는 두 가지 레벨의 설정을 설명한다:

- 프린터와 어떻게 연결하는지 알기 위해 간단한 프린터설정 섹션을 보면 LPD 가 어떻게 통신하고 프린터로 평범한 텍스트 파일을 출력하는지 알 수 있다.
- 다양한 특수파일 포맷출력, 헤더페이지 출력, 네트워크를 통한 출력, 프린터 사용 제어 그리고 프린터 계산을 어떻게 하는지 발전된 프린터 설정섹션을 본다.



---

## 9.3.1 간단한 프린터 설정

이번 섹션은 프린터 하드웨어와 LPD 소프트웨어를 어떻게 설정하는지 설명한다. 여기서는 기본적인 것만 가르친다:

- 하드웨어 설정 섹션은 프린터를 컴퓨터의 포트에 연결하는 몇 가지 힌트를 제공한다
- 소프트웨어 설정 섹션은 LPD 스플러 설정파일(/etc/printcap) 어떻게 설정하는지 보여준다.

데이터를 프린터로 보내기 위해 시리얼이나 패러럴 인터페이스 대신 네트워크 프로토콜을 사용한다면 네트워크 데이터 흐름 인터페이스로 출력하기를(9.4.3.2 장) 본다.

이번 섹션이 "간단한 프린터 설정"이지만 실제로 상당히 복잡하다. 컴퓨터와 LPD 스플러로 프린터를 작동하는 것이 가장 어려운 부분이다. 헤더페이지 및 계산하기처럼 발전된 프린터는 이 설정이 제대로 동작하면 상당히 쉽다.

### 9.3.1.1 하드웨어 설정

이번 섹션은 PC 에 프린터를 연결하는 다양한 방법을 소개한다. 포트와 케이블의 종류에 대해 설명하고 FreeBSD 가 프린터와 통신하도록 커널 설정도 필요할 것이다.

다른 운영체제에서 프린터를 사용하고 있다면 이번 섹션을 지나 소프트웨어 설정 섹션으로 넘어가도 된다.

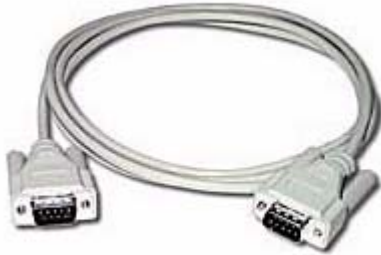
#### [포트와 케이블 특성 소개]

#### 1. 포트와 케이블

거의 모든 프린터는 다음 인터페이스 중 한가지 또는 두 가지를 사용하여 PC 에 연결할 수 있다:

- *시리얼* 인터페이스는 데이터를 프린터로 보내기 위해 컴퓨터의 시리얼 포트를 사용한다. 시리얼 인터페이스는 컴퓨터 산업에서 보편적이기 때문에 케이블을 쉽게 구할 수 있고 만들 수도 있다. 시리얼 인터페이스는 가끔 특수 케이블이

필요하고 복잡한 통신 옵션 설정이 필요할 수도 있다.



- *패러럴* 인터페이스는 데이터를 프린터로 보내기 위해 컴퓨터의 패러럴 포트를 사용한다. 패러럴 인터페이스는 PC 가게에서 쉽게 구할 수 있다. 케이블은 쉽게 구할 수 있지만 직접 만들기는 어렵다. 패러럴 인터페이스는 일반적으로 통신 옵션이 없기 때문에 설정이 매우 단순하다.

패러럴 인터페이스는 가끔 프린터의 연결 종류에 따라 "Centronics" 인터페이스라는 이름을 붙인다.



- Universal Serial Bus 를 의미하는 USB 인터페이스는 패러럴이나 RS232 시리얼 인터페이스보다 더 빠르게 동작할 수 있다. 케이블은 단순하고 싸다. USB 는 RS232 시리얼과 패러럴보다 성능이 좋지만 유닉스 시스템에서는 지원이 잘 안 된다. 이러한 문제를 피하기 위해 USB 인터페이스와 패러럴 인터페이스를 둘다 가지고 있는 프린터를 구입한다.



일반적으로 시리얼 인터페이스는 패러럴 인터페이스 보다 느리다. 패러럴 인터페이스는 보통 단방향 통신을(컴퓨터에서 프린터로) 제공하지만 시리얼과 USB 는 쌍방향을 제공한다. 새로운 패러럴 포트와(EPP 와 ECP) 프린터는 FreeBSD 에서 IEEE1284 케이블을 사용할 때 쌍방향으로 통신할 수 있다.

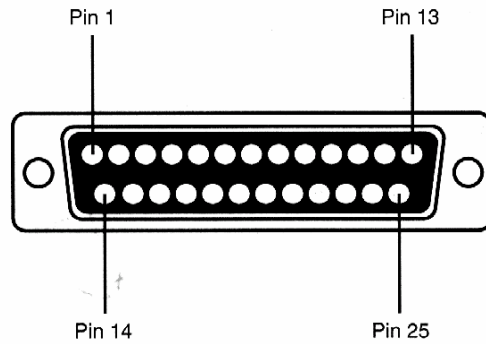
패러럴 포트를 이용한 쌍방향 통신 프린터는 보통 쌍방향 중 한쪽만 사용한다. 양쪽을 사용하는 첫 번째 방법은 사용자가 빌드한 프린터 드라이버를 FreeBSD 에서 사용한다. 이것은 잉크젯 프린터에서 일반적이고 잉크 양과 다른 상태 정보를 알려줄 수 있다. 두 번째 방법은 프린터가 포스트스크립트를 지원할 때 사용한다.

사실 포스트스크립트 잡은 실제 프로그램을 프린터에게 보내는 것이다; 결과를 직접 컴퓨터로 돌려주기 때문에 출력물 생성이 필요 없을 수 있다. 포스트스크립트도 포스트스크립트 프로그램에서 에러가 발생하거나 종이가 잦 걸리는 문제를 컴퓨터에게 통보하기 위해 쌍방향 통신을 사용한다. 유저가 이런 정보를 받을 수 있을 것이다. 게다가 포스트스크립트를 효과적으로 계산하려면 프린터는 쌍방향 통신이 필요하다: 프린터에서 페이지 양을 확인하고 (얼마나 많은 페이지를 출력했는지) 유저의 잡을 보내고 다시 페이지양을 확인한다. 두 값의 차이를 계산해서 유저에게 얼마나 많은 사용료를 부과 할 것인지 알수 있다.

## 2. 패러럴 포트

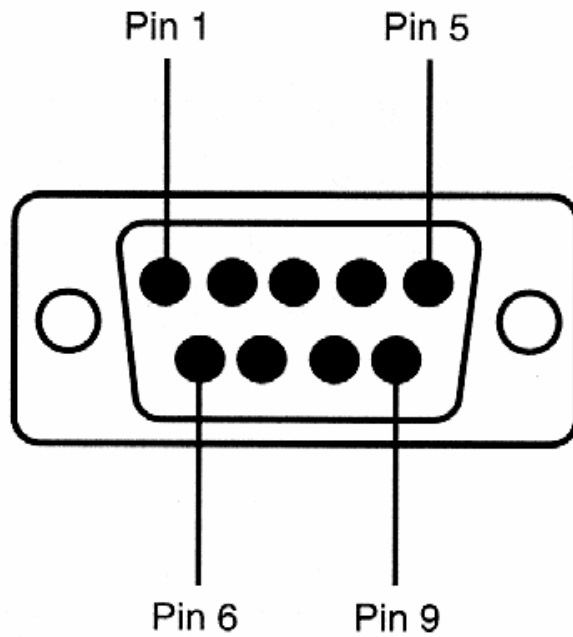
패러럴 인터페이스로 프린터를 사용 하려면 컴퓨터와 프린터 사이를 Centronics 케이블로 연결한다. 프린터나 컴퓨터 매뉴얼이 완벽한 길잡이를 제시한다.

컴퓨터에서 어떤 패러럴 포트를 사용하는지 기억한다. 첫 번째 패러럴 포트는 FreeBSD 에서 /dev/ppc0 고 두 번째는 /dev/ppc1 이다. 프린터 장치 이름도 같은 스키마를 사용한다: /dev/lpt0 는 첫 번째 패러럴 포트의 프린터다.



### 3. 시리얼 포트

시리얼 인터페이스로 프린터를 사용하려면 프린터와 컴퓨터 사이를 적절한 시리얼 케이블로 연결한다. 프린터나 컴퓨터의 매뉴얼이 완벽한 길잡이를 제시한다.



“적당한 시리얼 케이블”이 무엇인지 확실하지 않다면 다음 안내문 중 하나를 참고해 본다.

- 모뎀 케이블은 케이블의 끝 단자를 각 핀에 연결하고 그와 똑같이 반대편도 끝 단자의 핀에 연결한다. 이러한 종류의 케이블은 "DTE - DCE" 케이블이라고 한다.

- *널 모뎀* 케이블 연결은 몇 핀들은 직접 연결하고 몇 핀들은 교체하며(예를 들면 데이터 보내기를 받기로) 몇 핀들은 각각의 커넥터 후드(hood)에 고의적으로 단락(Short) 시킨다. 이런 종류의 케이블도 "DTE - DTE" 케이블이라고 한다.
- 어떤 특별한 프린터에 사용하는 *시리얼* 케이블은 *널 모뎀* 케이블과 비슷하지만 고의적으로 단락 시키지 않고 각자 대응되는 부분으로 신호를 보낸다.

프린터에 설정해야 되는 통신 매개변수도 보통 앞쪽의 제어 패널이나 프린터의 DIP 스위치로 설정한다. 프린터와 컴퓨터가 지원하는 최대 *bps* 율을(초당 bit 수를 말하며 종종 *보드(baud)* 율 이라고 한다) 선택한다. 7 이나 8 선택; none, even, odd parity; 그리고 1 또는 2 stop 비트. 흐름 제어 프로토콜도 선택한다; none 또는 XON/XOFF(또한 "in-band" 나 "소프트웨어"로 알고 있는) 흐름 제어. 소프트웨어 설정에 대한 내용은 아래 내용을 참조한다

### 9.3.1.2 소프트웨어 설정

이번 섹션은 FreeBSD 에서 LPD 스플링 시스템으로 출력하기 위해 필요한 소프트웨어 설정을 설명한다.

이 단계에 대한 개요를 소개한다.

- ① 필요하다면 프린터가 사용할 포트를 위해 커널을 설정한다; 커널 설정 섹션에서 필요한 것을 설명한다.
- ② 패러럴 포트를 사용한다면 패러럴 포트 통신모드를 설정한다; 패러럴 포트 통신모드 설정섹션에서 자세히 설명할 것이다.
- ③ 운영체제가 프린터에 데이터를 보낼 수 있는지 테스트한다. 프린터 통신체크 섹션에서 설명한다.
- ④ /etc/printcap 파일을 수정하여 프린터에 맞는 LPD 를 설정한다. 다음 장에서 설명한다.

---

## [커널 설정 소개]

### 커널 설정

운영체제 커널을 특정 장치 설정으로 동작 하도록 컴파일 한다. 프린터의 패러럴이나 시리얼 인터페이스는 위 설정 중 일부분이다. 그래서 커널이 포트에 맞게 설정되어 있지 않다면 시리얼이나 패러럴 포트 지원을 추가해야 될 것이다.

현재 사용중인 커널이 시리얼 인터페이스를 지원하는지 확인하려면 다음 명령을 입력한다:

```
# grep sioN /var/run/dmesg.boot
```

N은 0 부터 시작되는 시리얼 포트 번호다. 다음과 비슷한 결과가 있다면 커널은 포트를 지원한다.

```
sio2 at port 0x3e8-0x3ef irq 5 on isa  
sio2: type 16550A
```

커널이 패러럴 인터페이스를 지원하는지 확인 하려면 다음 명령을 입력한다:

```
# grep ppcN /var/run/dmesg.boot
```

N은 0 부터 시작되는 패러럴 포트 번호다. 다음과 비슷한 결과를 보게 된다면 커널은 포트를 지원한다.

```
ppc0: <Parallel port> at port 0x378-0x37f irq 7 on isa0  
ppc0: SMC-like chipset (ECP/EPP/PS2/NIBBLE) in COMPATIBLE mode  
ppc0: FIFO with 16/16/8 bytes threshold
```

운영체제가 프린터에 사용중인 패러럴이나 시리얼 포트를 감지해서 사용하도록 하려면 커널을 다시 설정해야 된다.

시리얼 포트 지원을 추가하려면 커널 설정에 관한 섹션을 본다. 패러럴 포트 지원을 추가하려면 그 섹션과 다음 섹션을 본다.

---

### 9.3.1.3 포트에 맞는 /dev 엔트리 추가

**Note:** FreeBSD 5.0 은 필요할 때 자동으로 장치 노드를 생성하는 *devfs* 파일시스템을 가지고 있다. *devfs* 가 활성화된 FreeBSD 버전을 사용한다면 이 섹션을 건너뛰어도 된다.

커널이 시리얼이나 패러럴 포트 통신을 지원하더라도 시스템의 프로그램이 데이터를 보내고 받을 수 있는 소프트웨어 인터페이스가 필요하다. 이것은 /dev 디렉터리에 있는 엔트리와 관련된다.

[예제: /dev 엔트리 추가, 통신모드 설정 그리고 프린터 체크하기]

#### 1. 포트에 맞는 /dev 엔트리 추가

① su(1) 명령으로 root 가 된다. 프롬프트가 나타나면 root 패스워드를 입력한다.

② /dev 디렉터리로 이동한다:

```
# cd /dev
```

③ 다음을 입력한다:

```
# ./MAKEDEV port
```

*port* 는 원하는 포트의 장치 엔트리다. 첫 번째 패러럴 포트 프린터는 *lpt0* 를 사용하고 두 번째 포트 프린터는 *lpt1* 이다; 마찬가지로 첫 번째 시리얼 포트는 *tyd0* 를 두 번째는 *tyd1* 다.

④ 다음을 입력한다:

```
# ls -l port
```

장치 엔트리가 생성 되었는지 확인한다.

## 2. 패러럴 포트 통신모드 설정

패러럴 인터페이스를 사용할 때 FreeBSD 가 interrupt-driven 방식 또는 polled 방식으로 프린터와 통신할 것인지 선택할 수 있다. FreeBSD 4.X 와 5.X 에서 보편적인 프린터 장치 드라이버는 ppc(4) 드라이버로 포트 칩셋을 제어하는 ppbus(4) 시스템을 사용한다.

- *interrupt-driven* 방식은 GENERIC 커널에서 기본값이다. 운영체제는 이 방법에서 프린터가 데이터를 받을 준비가 되었는지 결정하기 위해 IRQ 라인을 사용한다.
- *polled* 방식은 더 많은 데이터를 받을 준비가 되었는지 운영체제가 계속적으로 직접 프린터를 체크한다. 준비가 되었다는 응답이오면 커널은 더 많은 데이터를 보낸다.

interrupt-driven 방식은 일반적으로 약간 빠르지만 중요한 IRQ 라인을 사용한다. 어떤 새로운 HP 프린터는 인터럽트 모드에서 정확히 동작하지 않고 특히 타이밍 문제가(정확히 이해하지 못하는) 있다. 이런 프린터는 polled 모드가 필요하다. 어떤 프린터는 두 가지 모드에서 동작하지만 인터럽트 모드에서는 상당히 느리다.

커널을 설정 하거나 lptcontrol(8) 프로그램을 사용하여 위의 두가지 방식으로 통신모드를 설정할 수 있다.

**커널을 수정하여 통신모드를 설정하려면 다음 절차를 따른다:**

- ① 커널 설정 파일을 편집하고 *ppc0* 엔트리를 찾는다. 두 번째 패러럴 포트를 설정한다면 *ppc1* 을 사용하고 세 번째 포트는 *ppc2* 다.

- interrupt-driven 모드를 원하면 FreeBSD 4.X 는 *irq* 를 지정해야 된다:

```
device ppc0 at isa? irq N
```

*N*은 컴퓨터의 패러럴 포트 IRQ 번호다.

FreeBSD 5.X 는 다음 라인을 수정한다:

```
hint.ppc.0.irq="N"
```



---

/boot/device.hints 파일에서 적당한 IRQ 번호로 *N*을 변경한다. 커널 설정 파일도 ppc(4) 드라이버를 가지고 있어야 한다:

**device ppc**

- polled 모드를 원하면 *irq*를 추가하지 않는다:

FreeBSD 4.X는 커널 설정 파일에 다음 라인을 사용한다:

**device ppc0 at isa?**

FreeBSD 5.X는 단순히 다음과 같은 라인을 /boot/device.hints 파일에서 삭제한다:

**hint.ppc.0.irq="*N*"**

어떤 경우에는 FreeBSD 5.X에서 polled 모드로 포트를 설정하는 것이 적당하지 않다. 대부분이 acpi(4) 드라이버를 사용하기 때문에 이 문자를 검색한 후 장치에 붙여서 프린터 포트 접근 모드를 제어한다. 이 문제를 확인하기 위해 acpi(4) 설정을 체크한다.

- ② 파일을 저장하고 설정, 빌드해서 커널을 설치한 후 재부팅 한다. 더 자세한 내용은 커널 설정을 본다.

**lptcontrol(8)으로 통신모드를 설정하려면 다음 절차를 따른다:**

- ① 다음 명령을 실행한다:

```
# lptcontrol -i -d /dev/lptN
```

*lptN*을 interrupt-driven 모드로 설정한다.

- ② 다음 명령을 실행한다:

---

```
# lptcontrol -p -d /dev/lptN
```

*lptN*을 polled-모드로 설정한다.

시스템이 부팅할 때 마다 이 모드가 설정되도록 `/etc/rc.local` 파일에 이 명령을 입력한다. 더 많은 정보는 `lptcontrol(8)`을 본다.

### 3. 프린터 통신체크

스플링 시스템을 설정하기 전에 운영체제가 성공적으로 프린터에 데이터를 보내는지 확인해야 된다. 프린터 통신과 스플링 시스템을 개별적으로 디버그하는 것이 쉽기 때문이다.

프린터를 테스트하려면 텍스트를 보낸다. 프린터가 보낸 문자들을 즉시 출력하면 `lptest(1)` 프로그램은 완벽하다: 이 명령은 출력할 수 있는 ASCII 문자 96 개를 96 라인에 출력한다.

포스트스크립트(또는 다른 언어 기반) 프린터는 더 복잡한 테스트가 필요하다. 다음과 같은 작은 포스트스크립트면 충분하다:

```
%!PS
100 100 moveto 300 300 lineto stroke
310 310 moveto /Helvetica findfont 12 scalefont setfont
(Is this thing working?) show
showpage
```

위의 포스트스크립트 코드는 파일로 변환되어 다음 섹션의 예제에서처럼 사용된다.

**Note:** 이 문서에서 프린터 언어는 휴렛팩커드의 PCL 이 아닌 포스트스크립트 같은 언어를 말한다. PCL 이 다양한 기능을 가지고 있어도 `escape` 시퀀스로 평범한 텍스트를 섞을 수 있다. 포스트스크립트는 직접 평범한 텍스트를 출력할 수 없기 때문에 이런 종류의 프린터 언어는 특수한 조정이 필요하다.

### 4. 패러럴 프린터 체크

---

이번 섹션은 FreeBSD 가 패러럴 포트로 연결된 프린터와 통신할 수 있는지 체크하는 방법을 설명한다.

패러럴 포트에서 프린터를 테스트하려면 다음 절차를 따른다:

- ① su(1)로 root가 된다.
- ② 프린터로 데이터를 보낸다.
  - 프린터가 평범한 텍스트를 출력할 수 있다면 lptest(1)로 다음 명령을 입력한다:

```
# lptest > /dev/lptN
```

*N*은 0 부터 시작되는 패러럴 포트 번호다.

- 프린터가 포스트스크립트나 다른 프린터 언어를 이해한다면 작은 프로그램을 프린터에 보내기 위해 다음 명령을 입력한다:

```
# cat > /dev/lptN
```

그리고 *RETURN*이나 *ENTER*를 누른 후 라인을 편집할 수 없기 때문에 라인당 입력된 프로그램을 주의 깊게 입력한다. 프로그램 입력이 끝나면 *CONTROL+D*나 파일을 끝내는 키를 누른다.

다른 방법은 파일에 프로그램을 입력하고 다음과 같이 실행할 수 있다:

```
# cat file > /dev/lptN
```

*file*은 프린터에 보내고 싶은 프로그램을 가지고 있는 파일 이름이다.

무엇인가 출력되는 것을 보게 된다. 텍스트가 제대로 보이지 않더라도 걱정하지 않는다; 조금 후에 수정할 것이다.

## 5. 시리얼 프린터체크

이 섹션은 FreeBSD 가 시리얼 포트에서 프린터와 통신할 수 있는지 체크하는 방법을 설명한다.

시리얼 포트에서 프린터를 테스트하려면 다음 절차를 따른다:

- ① su(1)로 root가 된다.
- ② /etc/remote 파일에 다음 엔트리를 추가한다:

```
printer:dv=/dev/port:br#bps-rate:pa=parity
```

*port*는 시리얼 포트의 (*ttyd0*, *ttyd1* 등) 장치 엔트리, *bps-rate*는 프린터 통신의 초당 비트율 그리고 *parity*는 프린터에 필요한 패리티다(*even*, *odd*, *none* 이나 *zero* 중 하나다)

여기 세 번째 시리얼 포트에 19200bps로 패리티 없이 시리얼 라인으로 연결된 프린터의 샘플 엔트리가 있다:

```
printer:dv=/dev/ttyd2:br#19200:pa=none
```

- ③ tip(1)으로 프린터를 연결하기 위해 다음 명령을 입력한다:

```
# tip printer
```

이 단계가 동작하지 않는다면 /etc/remote 파일을 수정하고 /dev/ttyd*N* 대신 /dev/cuaa*N*을 사용하여 다시 시도한다.

- ④ 프린터에 데이터를 보낸다.
  - 프린터가 평범한 텍스트를 출력할 수 있다면 lptest(1)로 다음 명령을 입력한다:

```
% $lptest
```

- 프린터가 포스트스크립트나 다른 프린터 언어를 이해한다면 프린터에 작은 프로그램을 보낸다. 백스페이스나 다른 편집 키는 프린터에서 매우 중요하게 작용할 수 있기 때문에 아주 주의하여 라인 별로 프로그램을 입력한다.

---

전체 프로그램을 보냈다는 것을 통보하기 위해 프린터에게 파일 끝을 알리는 키를 입력해야 된다. 포스트스크립트는 *CONTROL + D*를 누른다.

다른 방법은 파일에 프로그램을 넣고 다음과 같이 실행한다:

```
% >file
```

*file*은 프로그램을 가지고있는 파일 이름이다. tip(1)이 파일을 보낸 후 파일 끝을 알리는 키를 눌러야 한다.

출력 결과를 볼 수 있다. 텍스트가 제대로 보이지 않더라도 걱정하지 않는다; 이 후에 고칠 것이다.

#### 9.3.1.4 스플러 활성화: /etc/printcap 파일

이제 프린터가 연결되었고 통신을 하도록 커널을 설정하였기 때문에(필요하다면) 프린터에 간단한 데이터를 보낼 수 있다. 프린터 사용제어를 위해 **LPD**를 설정할 준비가 됐다.

/etc/printcap 파일을 수정하여 **LPD**를 설정한다. **LPD** 스플링 시스템은 스플러가 사용될 때마다 이 파일을 읽기 때문에 파일 업데이트 효과는 즉시 나타난다.

printcap(5) 파일 포맷은 직선적이다. /etc/printcap를 수정하려면 텍스트 에디터를 사용한다. 포맷은 /usr/share/misc/termcap와 /etc/remote 같은 특정 파일과 동일하다. 포맷에 대한 완벽한 정보는 cgetent(3)를 본다.

간단한 스플러 설정은 다음과 같은 단계로 이루어진다:

- ① 프린터 이름을 선택해서(그리고 몇 개의 엘리머스도) /etc/printcap 파일에 입력한다; 이름에 대한 더 많은 정보는 프린터 이름 섹션을 본다.
- ② *sh* 문자열로 헤더페이지를(기본적으로 켜 있는) 끌 수 있다; 더 많은 정보는 헤더페이지 끄기 섹션을 본다.
- ③ 스플링 디렉토리를 만들고 위치를 *sd* 문자열로 지정한다; 더 많은 정보는 스플링

---

디렉터리 만들기 섹션을 본다.

- ④ 프린터에 `/dev` 엔트리를 사용하도록 설정하고 `lp` 문자열로 `/etc/printcap` 에 입력한다; 더 많은 정보는 프린터 장치 확인하기를 본다. 또한 프린터가 시리얼 포트에 연결되어 있다면 스플러 통신 매개변수 설정 섹션에서 설명하는 `ms#` 문자열로 통신 매개변수를 설정한다.
- ⑤ 평범한 텍스트 입력필터 설치; 자세한 사항은 텍스트 필터 설치 섹션을 본다.
- ⑥ `lpr(1)` 명령으로 출력하여 설정을 테스트한다. 더 자세한 내용은 문제 해결 섹션에서 볼 수 있다.

**Note:** 포스트스크립트 프린터처럼 언어 기반 프린터는 직접 평범한 텍스트를 출력할 수 없다. 간단한 설정에 대한 개요는 위에서 설명했고 다음 섹션은 프린터가 이해할 수 있는 파일만 출력한다고 가정한다.

유저는 가끔 시스템에 어떤 프린터가 연결되어 있더라도 평범한 텍스트는 출력할 수 있다고 생각한다. LPD 와 관련된 프로그램도 같은 개념이다. 이런 프린터가 설치되어 있고 프린터 언어와 평범한 텍스트 잡을 출력 하려면 위의 개요에 간단한 설정 단계를 추가해야 된다: 자동으로 평범한 텍스트를 포스트스크립트로(또는 다른 프린터 언어로) 변환하는 프로그램을 설치한다. 포스트스크립트 프린터에서 평범한 텍스트 잡 출력하기 섹션에서 방법을 설명한다.

#### [예제: `/etc/printcap` 파일의 세부적인 설정]

##### 1. 프린터에 이름 붙이기

첫 단계(쉽다)는 프린터의 이름을 선택한다. 프린터에 다양한 엘리어스를 제공할 수 있기 때문에 기능적이든 기묘한 이름을 선택하던지 문제는 없다.

`/etc/printcap` 의 프린터 중 하나는 `lp` 엘리어스를 가져야 한다. 이것은 기본 프린터의 이름이다. 유저가 `PRINTER` 환경변수를 사용하지 않았거나 `LPD` 명령의 명령어 라인에서 프린터 이름을 지정하지 않았다면 `lp` 는 기본 프린터가 사용한다.

또한 일반적인 관례상 프린터의 마지막 엘리어스는 모델을 포함한 프린터의 자세한 설명을 넣는다.

---

이름과 일반적인 엘리어스를 선택해서 /etc/printcap 파일에 입력한다. 프린터 이름은 거의 왼쪽 종렬에서 시작된다. 수직 막대기(|)로 엘리어스를 각각 분리하고 마지막 엘리어스는 콜론(:)으로 끝낸다.

다음 예제는 두대의 프린터가(Diablo 630 라인 프린터와 Panasonic KX-P4455 포스트스크립트 레이저 프린터) 지정되어 있는 /etc/printcap 의 시작부분이다:

```
#
# /etc/printcap for host rose
#
rattan|line|diablo|lp|Diablo 630 Line Printer:

bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:
```

이 예제에서 첫 번째 프린터 이름은 *rattan* 이고 *line*, *diablo*, *lp* 와 *Diablo 630 Line Printer* 를 엘리어스로 가지고 있다. 또한 *lp* 엘리어스도 가지고 있기 때문에 기본 프린터가 된다. 두 번째 이름은 *bamboo* 고 *ps*, *PS*, *S*, *panasonic* 과 *Panasonic KX-P4455 PostScript v51.4* 를 엘리어스로 가지고 있다.

## 2. 헤더페이지 끄기

LPD 스플러 시스템은 기본적으로 각 잡의 헤더페이지를 출력한다. 헤더페이지는 잡을 요청한 유저 이름, 잡을 보낸 호스트와 잡의 이름을 아주 큰 문자로 출력한다. 불행히 이 부가적인 텍스트는 단순히 프린터 설정을 디버깅하는 방법이기 때문에 우리는 헤더페이지를 끈다.

헤더페이지를 끄려면 *sh* 문자열 엔트리를 /etc/printcap 파일에 추가한다. 여기 *sh* 가 추가된 /etc/printcap 예제가 있다:

```
#
# /etc/printcap for host rose - no header pages anywhere
#
rattan|line|diablo|lp|Diablo 630 Line Printer:W
      :sh:

bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:W
```

---

`:sh:`

우리가 정확히 어떤 포맷을 사용했는지 적어둔다: 거의 왼쪽 칼럼에서 시작된 첫 번째 라인과 다음 라인은 TAB 한번으로 들여쓰기 한다. 엔트리에서 마지막 라인을 제외하고 모든 라인은 백슬러시 문자로 끝낸다.

### 3. 스푼링 디렉터리 생성

간단한 스푼러 설정에서 다음 단계는 잡이 출력될 때까지 보관되는 곳이며 스푼러를 지원하는 다른 파일이 있는 스푼링 디렉터리를 만든다.

다양한 스푼링 디렉터리는 관례상 `/var/spool` 디렉터리에 만들기 때문에 스푼링 디렉터리 내용은 백업할 필요 없고 다시 디렉터리를 만드는 것은 단순히 `mkdir(1)`을 사용한다.

또한 관례적으로 디렉터리 이름은 아래와 같이 프린터 이름을 확인할 수 있는 이름을 사용한다:

```
# mkdir /var/spool/printer-name
```

그러나 네트워크에 많은 프린터를 가지고 있다면 LPD 로 출력할 수 있도록 하나의 디렉터리 밑에 스푼링 디렉터리를 만들 수 있다. 우리는 두 예제 프린터 *rattan* 과 *bamboo* 를 위해 다음과 같이 실행했다:

```
# mkdir /var/spool/lpd
# mkdir /var/spool/lpd/rattan
# mkdir /var/spool/lpd/bamboo
```

**Note:** 유저가 개인적으로 출력하는 것이 걱정된다면 스푼링 디렉터리를 보호하여 공개적으로 접근하지 못하게 해야 된다. 스푼링 디렉터리는 유저 데몬과 그룹 데몬을 제외하고 다른 사람들이 읽기, 쓰기, 찾기를 못 해야 된다. 우리는 예제 프린터에 다음과 같이 설정했다:

```
# chown daemon:daemon /var/spool/lpd/rattan
# chown daemon:daemon /var/spool/lpd/bamboo
# chmod 770 /var/spool/lpd/rattan
# chmod 770 /var/spool/lpd/bamboo
```



---

마지막으로 `/etc/printcap` 파일에 이런 디렉토리를 지정하여 LPD 에게 알려 줘야 한다. `sd` 문자열로 스포링 디렉터리 경로 이름을 지정한다:

```
#
# /etc/printcap for host rose - added spooling directories
#
rattan|line|diablo|lp|Diablo 630 Line Printer:W
      :sh:sd=/var/spool/lpd/rattan:

bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:W
      :sh:sd=/var/spool/lpd/bamboo:
```

프린터 이름은 첫 번째 칼럼에서 시작하지만 다른 모든 엔트리는 탭으로 들여쓰기 하고 백슬러시로 각 라인을 끝낸다.

`sd`로 스포링 디렉토리를 지정하지 않았다면 스포링 시스템은 `/var/spool/lpd` 를 기본값으로 사용한다.

#### 4. 프린터 장치 확인

포트 섹션에 `/dev` 엔트리를 추가하여 FreeBSD 가 프린터와 통신할 때 `/dev` 디렉터리의 어떤 엔트리를 사용하는지 확인한다. 이제 LPD 정보를 설명한다. 스포링 시스템이 출력할 잡을 가지고 있을때 필터 프로그램에(데이터를 프린터에 보내는 기능을 가진) 지정된 장치를 연다.

`/etc/printcap` 파일에 `/dev` 엔트리 경로이름은 `lp` 문자열을 사용하여 지정한다.

우리 예제에서 `rattan`은 첫 번째 패러럴 포트에 `bamboo`는 여섯 번째 시리얼 포트에 있다고 가정한다; 다음 내용을 `/etc/printcap` 에 추가한다:

```
#
# /etc/printcap for host rose - identified what devices to use
#
rattan|line|diablo|lp|Diablo 630 Line Printer:W
      :sh:sd=/var/spool/lpd/rattan:W
```

`:lp=/dev/lpt0:`

`bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:W`

`:sh:sd=/var/spool/lpd/bamboo:W`

`:lp=/dev/ttyd5:`

`/etc/printcap` 파일에 `/p` 문자열로 프린터를 지정하지 않았다면 `LPD` 는 `/dev/lp` 를 기본값으로 사용한다. `/dev/lp` 는 현재 FreeBSD 에 존재하지 않는다.

설치하려는 프린터가 패러럴 포트에 연결되어 있다면 이번 섹션을 넘어서 텍스트 필터 설치하기로 간다. 그렇지 않다면 다음 섹션의 지시를 따른다.

## 5. 스플러 통신 매개변수 설정

시리얼 포트에 있는 프린터를 위해 `LPD` 는 데이터를 프린터에 보내는 필터 프로그램에 `bps` 율, 패리티 그리고 다른 시리얼 통신 매개변수를 설정할 수 있다. 이것은 다음과 같은 이유로 이점이 된다:

- 간단히 `/etc/printcap` 파일을 수정해서 다른 통신 매개변수를 사용할 수 있다; 필터 프로그램을 다시 컴파일 할 필요는 없다.
- 다른 시리얼 통신 설정을 가진 여러대의 프린터가 같은 필터 프로그램을 사용하도록 스플링 시스템을 활성화한다.

다음 `/etc/printcap` 는 `/p` 문자열로 나열된 장치의 시리얼 통신 매개변수를 제어 한다:

**`br#bps-rate`**

장치의 통신 속도를 초당 50, 75, 110, 134, 150, 200, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, 38400, 57600 이나 115200 로 설정할 수 있는 `bps-rate` 로 설정한다.

**`ms#stty-mode`**

장치를 열고 터미널 장치 옵션을 설정한다. `stty(1)` 매뉴얼 페이지에 유효한 옵션

설명이 있다.

LPD 가 *lp* 문자열로 지정된 장치를 열었을때 *ms#* 문자열에 지정된 것으로 장치 특성을 설정한다. 관련 특성은 *stty(1)* 매뉴얼 페이지에 설명되어 있는 *parenb*, *parodd*, *cs5*, *cs6*, *cs7*, *cs8*, *cstopb*, *crtcts* 와 *ixon* 모드다.

여섯 번째 시리얼 포트에 예제 프린터를 추가하고 bps 율을 38400 으로 설정한다. 모드는 *-parenb*로 패리티 없음, *cs8*로 8-bit 캐릭터, *clocal*로 모뎀 제어 없고 *crtcts*로 하드웨어 흐름을 제어한다:

```
bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:₩
      :sh:sd=/var/spool/lpd/bamboo:₩
      :lp=/dev/ttyd5:ms#-parenb cs8 clocal crtcts:
```

## 6. 텍스트 필터 설치

이제 LPD 에게 어떤 텍스트 필터를 사용하여 프린터에 잡을 보낼지 알려줘야 한다. *입력 (input) 필터*로 알려져 있는 *텍스트 필터*는 프린트할 잡이 있을 때 LPD 가 실행하는 프로그램이다. 프린터의 LPD 가 텍스트 필터를 실행할 때 잡을 출력하기 위해 필터의 표준 입력을 입력으로 *lp* 문자열로 지정된 프린터 장치를 표준 출력으로 설정한다. 필터는 기본 입력으로 작업을 읽고 프린터에 필요한 변환을 수행하여 기본 출력으로 결과를 보낼 것이다. 텍스트 필터에 대한 더 많은 정보는 필터 섹션을 본다.

우리의 간단한 프린터 설정 텍스트 필터는 */bin/cat* 을 실행하여 프린터에게 잡을 보내는 작은 셸 스크립트다. FreeBSD 는 백스페이스와 언더라인(\_)같은 문자 흐름을 제대로 제어하지 못하는 프린터를 위해 *lpf* 라는 필터를 가지고 있다. 물론 원한다면 다른 필터 프로그램을 사용할 수 있다. *lpf* 필터는 텍스트 필터 섹션에서 자세히 설명한다.

첫째 간단한 텍스트 필터가 될 */usr/local/libexec/if-simple* 셸 스크립트를 만든다. 텍스트 에디터로 다음 내용을 위 파일에 넣는다:

```
#!/bin/sh
#
# if-simple - Simple text input filter for lpd
# Installed in /usr/local/libexec/if-simple
#
```

---

```
# Simply copies stdin to stdout.  Ignores all filter arguments.
```

```
/bin/cat && exit 0
```

```
exit 2
```

파일을 실행할 수 있도록 한다:

```
# chmod 555 /usr/local/libexec/if-simple
```

그리고 /etc/printcap 에 if 문자열로 지정하여 LPD 가 사용하도록 한다. 우리는 이것을 우리가 가진 두 대의 프린터에 추가해서 /etc/printcap 예제는 다음과 같다:

```
#
```

```
# /etc/printcap for host rose - added text filter
```

```
#
```

```
rattan|line|diablo|lp|Diablo 630 Line Printer:W
```

```
    :sh:sd=/var/spool/lpd/rattan:W :lp=/dev/lpt0:W
```

```
    :if=/usr/local/libexec/if-simple:
```

```
bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:W
```

```
    :sh:sd=/var/spool/lpd/bamboo:W
```

```
    :lp=/dev/ttyd5:ms#-parenb cs8 clocal crtscts:W
```

```
    :if=/usr/local/libexec/if-simple:
```

## 7. LPD 켜기

lpd(8)는 /etc/rc 로 실행하고 *lpd\_enable* 변수로 제어된다. 이 변수의 기본값은 *NO*다. 설정하지 않았다면 다음 라인을 /etc/rc.conf 에 추가한다:

```
lpd_enable="YES"
```

그리고 머신을 다시 시작하거나 lpd(8)을 실행한다.

```
# lpd
```

---

## 8. 테스트

간단한 LPD 설정의 마지막에 도달했지만 아직 끝난것이 아니다. 왜냐하면 설정을 테스트 하고 문제가 있으면 해결해야 되기 때문이다. 설정을 테스트 하려면 출력을 해본다. LPD 시스템에서 출력하려면 출력할 잡을 보내는 `lpr(1)` 명령을 사용한다.

테스트 텍스트를 출력하기 위해 프린터 통신체크 섹션에서 설명한 `lpr(1)`과 `lptest(1)` 프로그램을 조합할 수 있다.

간단한 LPD 설정을 테스트하려면 다음 명령을 실행한다:

```
# lptest 20 5 | lpr -Pprinter-name
```

*printer-name* 은 `/etc/printcap` 에 지정한 프린터 이름이다(또는 엘리어스). 기본 프린터를 테스트 하려면 `-P` 인자 없이 `lpr(1)`을 입력한다. 다시 포스트스크립트를 테스트 하려면 `lptest(1)` 대신 포스트스크립트 프로그램을 보낸다. 파일에 프로그램을 넣고 `lpr file`을 입력하여 프로그램을 보낼 수 있다.

포스트스크립트 프린터는 프로그램 결과를 출력한다. `lptest(1)`를 사용했다면 결과는 다음과 비슷할 것이다:

```
!#$%&'()*+,-./01234  
"#$%&'()*+,-./012345  
#$%&'()*+,-./0123456  
$%&'()*+,-./01234567  
%&'()*+,-./012345678
```

프린터를 더 테스트 하려면 큰 프로그램을(언어 기반 프린터를 위해) 다운로드 하거나 다른 인자로 `lptest(1)`를 실행한다. 예를 들어 `lptest 80 60`은 60 라인에 각각 80 문자를 생성한다.

프린터가 작동하지 않으면 문제 해결 섹션을 본다.

## 9.4 발전된 프린터 설정

---

이 섹션은 특별하게 포맷된 파일, 헤더페이지, 네트워크를 통한 프린트 그리고 프린터 사용량의 제한과 사용량 확인을 위한 필터를 설명한다.

## 9.4.1 필터

LPD가 네트워크 프로토콜, 큐, 사용제어 그리고 프린터의 다른 부분을 제어 하더라도 대부분의 실제 작업은 필터에서 이루어진다. 필터는 프린터의 통신 장치에 관련된 부분을 제어하는 특수한 필수품이다. 간단한 프린터 설정에서 대부분의 프린터에서(텍스트 필터 설치하기 섹션을 본다) 작동하는 평범한 텍스트 필터를 아주 간단히 설치하였다.

그러나 포맷변환과 출력물 계산 그리고 특수한 quirks 프린터 등의 장점을 살리기 위해 필터가 어떻게 동작하는지 알아야 한다. 이런 기능을 제어하는 것이 필터의 역할이다. 그리고 나쁜 소식은 대부분 필터를 직접 제공해야 된다는 것이다. 좋은 소식은 대부분의 필터를 일반적으로 사용할 수 있다; 아니면 쉽게 작성할 수 있다.

또한 FreeBSD는 많은 프린터에서 동작하여 평범한 텍스트를 출력할 수 있는 `/usr/libexec/lpr/lpf`를(이것은 파일에서 백스페이스와 탭을 제어하고 출력물 계산까지 하지만 모든 프린터에서 동작하지는 않는다.) 가지고 있다. FreeBSD 포트 컬렉션에도 여러가지 필터와 필터 컴포넌트가 있다.

여기서 이번 섹션에서 얻을 수 있는 것을 소개한다:

- 필터가 어떻게 동작하는가 섹션에서는 출력 과정에서 필터의 역할을 설명한다. 이 섹션을 읽고 LPD가 필터를 사용할 때 어떤 일이 발생하는지 이해할 수 있다. 이 지식은 프린터에 더 많은 필터를 설치 할 때마다 부딪힐 수 있는 문제를 방지하고 해결할 수 있도록 돕는다.
- LPD는 기본적으로 모든 프린터가 평범한 텍스트를 출력할 수 있게 한다. 이것은 평범한 텍스트를 직접 출력할 수 없는 포스트스크립트 프린터에(또는 다른 언어 기반 프린터) 문제가 된다. 포스트스크립트 프린터에서 평범한 텍스트 잡을 출력하기 섹션에서 문제를 해결하기 위한 방법을 설명한다. 포스트스크립트 프린터를 가지고 있다면 이 섹션을 읽도록 한다.
- 포스트스크립트는 다양한 프로그램의 유명한 출력포맷이다. 어떤 사람들은 직접 포스트스크립트 코드를 작성 하지만 포스트스크립트 프린터는 비싸다. 일반 프린터에

---

서 포스트스크립트 시뮬레이팅하기 섹션은 일반 프린터가 포스트스크립트 데이터를 받아서 출력하도록 프린터의 텍스트 필터를 어떻게 수정하는지 알려준다. 일반적인 프린터를 가지고 있다면 이번 섹션을 읽도록 한다.

- 파일변환 섹션은 그래픽이나 typesetting 데이터 같은 특수한 파일포맷을 프린터가 이해하는 포맷으로 자동으로 변환하는 방법에 대해 설명한다. 이 섹션을 읽고 사용자가 troff 데이터 출력은 `lpr -t` 를 TeX DVI 데이터는 `lpr -d` 그리고 도트 방식 이미지 데이터는 `lpr -v` 등으로 출력하도록 프린터를 설정할 수 있다. 이 섹션을 읽도록 개인적으로 권장한다.
- 출력필터 섹션은 자주 사용하지 않는 LPD의 특성에 대해 설명한다: 헤더페이지를 출력하지 않는다면 이번 섹션을 건너 뛰어도 된다.
- `lpf` 섹션에서는 텍스트 필터 `lpf` 를 설명한다. 라인 프린터를 위한(그리고 라인 프린터처럼 작동하는 레이저 프린터) 간단한 텍스트 필터가 FreeBSD 에 필요하다면 텍스트 필터 `lpf` 가 거의 완벽하다. 평범한 텍스트를 계산하는 빠른 방법이 필요하거나 백스페이스 문자를 만났을때 오 동작하는 프린터를 가지고 있으면 `lpf` 를 사용한다.

#### 9.4.1.1 필터가 어떻게 작동되는가

이미 설명했듯이 필터는 LPD 가 실행하는 프로그램으로 프린터 통신장치와 관련된 부분을 제어한다.

LPD 는 파일을 출력하려고할때 필터 프로그램을 시작한다. 이것은 파일을 출력하기 위해 필터의 기본입력은 파일로 표준출력을 프린터 그리고 표준에러는 에러 로그파일로 설정한다 (기본적으로 `/etc/printcap` 또는 `/dev/console` 에서 `#` 문자열로 지정한).

LPD 가 사용하는 필터와 필터인자는 `/etc/printcap` 파일에 나열된 것을 따르고 잡에 지정하는 인자는 `lpr(1)` 명령어 라인을 따른다. 예를 들면 사용자가 `lpr -t` 를 입력하였다면 LPD 는 목적 프린터의 `#` 문자열에 따라 troff 필터를 시작한다. 사용자가 평범한 텍스트를 출력 한다면 LPD 는 `#` 필터를(이것은 거의 사실이다: 자세한 사항은 출력필터를 본다) 시작한다.

`/etc/printcap` 에 지정할 수 있는 3 종류의 필터가 있다:

- 
- 혼동할 수 있지만 LPD 문서에서 *입력필터*라고 하는 텍스트필터는 일반 텍스트 출력력을 제어한다. 이것을 기본필터처럼 생각한다. 텍스트필터의 기능은 백스페이스와 탭 또는 다른 특수문자를 프린터가 잘못 인지하는 것을 방지하여 LPD가 모든 프린터에서 기본적으로 텍스트를 출력할 수 있도록 한다. 프린터 사용량을 계산해야 된다면 텍스트필터는 일반적으로 출력된 라인 수와 프린터가 지원하는 페이지당 라인을 비교하여 출력된 페이지를 계산한다. 텍스트필터는 다음 인자 리스트로 시작된다:

```
filter-name [-c] -width -length -indent -n login -h host acct-file
```

*-c*

잡을 lpr -i 로 실행하면 나타난다.

*width*

/etc/printcap 에 기본값이 132 로 설정되어있는 *pw* 문자열의(페이지의 넓이) 값이다.

*length*

기본값이 66 으로 설정된 *pl* 문자열의(페이지 길이) 값이다.

*indent*

기본값이 0 으로 설정된 lpr -i 로부터 들여쓰기(종이에서 처음으로 출력되는 위치) 값이다..

*login*

파일을 출력하는 유저계정의 이름이다.



---

*host*

잡을 보낸 호스트의 이름이다.

*acct-file*

*af* 문자열에서 파일계정의 이름이다.

- *변환필터*는 프린터가 종이로 출력할 수 있는 파일포맷으로 변환한다. 예를 들어 *ditroff* 글씨체 데이터는 직접 출력할 수 없지만 *ditroff* 파일의 데이터를 프린터가 출력할 수 있는 형식으로 변환하는 *ditroff* 파일 변환필터를 설치할 수 있다. 변환 필터 섹션에서 변환필터에 대해 설명한다. 또한 출력량 계산이 필요하다면 계산에도 변환필터가 필요하다. 변환필터는 다음 인자로 시작된다:

*filter-name -xpixel-width -ypixel-height -n login -h host acct-file*

*pixel-width*는 *px* 문자열의(기본값은 0) 값이고 *pixel-height*는 *py* 문자열의(기본값은 0) 값이다.

- 출력(output)필터는 텍스트필터가 없거나 헤더페이지가 필요할 때만 사용된다. 내 경험상 출력필터는 별로 사용하지 않는다. 출력필터 섹션에서 출력필터에 대해 설명한다. 출력필터에는 두개의 인자만 있다:

*filter-name -width -length*

텍스트필터와 동일하게 *-w*와 *-l* 인자를 사용한다.

필터는 *exit* 문으로 끝낼 수 있다:

*exit 0*

---

파일이 성공적으로 출력됐을 때

exit 1

필터가 파일출력은 실패했지만 LPD 가 파일출력을 다시 시도했을 때. 이 상태가 되면 LPD 는 필터를 다시 시작한다.

exit 2

필터가 파일출력에 실패하고 LPD 가 다시 시도하지 않을 때. LPD 는 파일을 버린다.

FreeBSD 릴리즈에 포함된 /usr/libexec/lpr/lpf 텍스트 필터는 폼 피드(feed)를 보내고 프린터 사용량 계산 방법을 결정할 때 width 와 length 인자의 이점을 사용한다. 계정 엔트리를 생성하는데 로그인과 호스트 그리고 계정 파일인자를 사용한다.

필터를 구매한다면 LPD 와 호환되는지 확인한다. 필터가 호환된다면 위에서 설명한 인자들을 지원해야 된다. 일반적인 용도로 필터를 작성할 계획이면 이 필터들은 같은 인자와 exit 코드를 지원해야 된다.

#### 9.4.1.2 포스트스크립트 프린터에서 평범한 텍스트 잡 처리

컴퓨터와 포스트스크립트 프린터의(또는 다른 언어기반의) 유일한 유저로서 절대 프린터에 평범한 텍스트를 보내지 않고, 평범한 텍스트를 프린터에 보낼 수 있는 다양한 프로그램들의 기능을 사용하지 않겠다면 이번 섹션은 걱정할 필요가 없다.

그러나 프린터에 포스트스크립트와 평범한 텍스트 잡을 보낼 생각이면 프린터에 인자를 설정해야 된다. 이렇게 하기 위해 우리는 평범한 텍스트나 포스트스크립트 잡이 입력될 때 감지하는 텍스트 필터를 가지고 있다. 모든 포스트스크립트 잡은 %/로(다른 언어는 프린터 문서를 본다) 시작한다. 잡에 이들 두 문자가 있다면 포스트스크립트 파일이기 때문에 직접 나머지 잡을 처리할 수 있다. 파일에 이 두 문자가 없다면 필터는 텍스트를 포스트스크립트로 변환하고 결과를 출력한다.

---

우리가 할수 있는 방법?

시리얼 프린터를 가지고 있다면 가장 좋은 방법은 lprps 를 설치하는 것이다. lprps 는 프린터와 쌍방향 통신을 하는 포스트스크립트 프린터필터다. 이것은 프린터의 다양한 정보로 프린터 상태파일을 업데이트 하여 관리자가 정확하게 프린터 상태를(토너가 없다거나 종이가 걸린 것 같은) 알수 있다. 그러나 더 중요한 것은 입력되는 잡이 텍스트인지 확인하는 psif 와(lprps 에 달려 온 다른 프로그램) 포스트스크립트로 변환하는 textps 라는 프로그램을 가지고 있다. 프린터에 잡을 보낼때 lprps 를 사용한다.

lprps 는 FreeBSD 포트 컬렉션에 있다(포트 컬렉션을 본다). 포트 컬렉션에서 설치하거나 직접 빌드해서 설치할 수 있다. lprps 를 설치하고 lprps 의 일부인 psif 프로그램에 경로 이름을 지정한다. lprps 를 포트 컬렉션에서 설치했다면 /etc/printcap 의 시리얼 포스트스크립트 프린터 엔트리에 다음 라인을 사용한다:

```
:if=/usr/local/libexec/psif:
```

또한 LPD 가 프린터를 입력-출력 모드로 열도록 *rw* 문자열을 지정할 수 있다.

패러럴 포스트스크립트 프린터를(프린터와 쌍방향 통신을 못하기 때문에 lprps 가 필요한) 가지고 있다면 텍스트 필터로 다음 쉘 스크립트를 사용할 수 있다:

```
#!/bin/sh
#
# psif - Print PostScript or plain text on a PostScript printer
# Script version; NOT the version that comes with lprps
# Installed in /usr/local/libexec/psif
#

IFS="" read -r first_line
first_two_chars=`expr "$first_line" : 'W(..W)'\`

if [ "$first_two_chars" = "%!" ]; then
#
# PostScript job, print it.
#
echo "$first_line" && cat && printf "W004" && exit 0
```

```

    exit 2
else
    #
    # Plain text, convert it, then print it.
    #
    ( echo "$first_line"; cat ) | /usr/local/bin/textps && printf "W004" && exit 0
    exit 2
fi

```

위 스크립트에서 textps 는 우리가 평범한 텍스트를 포스트스크립트로 변환하도록 따로 설치한 프로그램이다. 원하는 텍스트-포스트스크립트 변환 프로그램을 사용할 수 있다. FreeBSD 포트 컬렉션은 텍스트-포스트스크립트 변환하는 모든 기능을 가진 a2ps 라는 프로그램을 가지고 있다.

#### 9.4.1.3 일반 프린터에서 포스트스크립트 시뮬레이트

포스트스크립트는 고품질의 인쇄와 출력을 하는 *de facto* 표준이다. 그러나 포스트 스크립트는 기본적으로 비싸다. 다행히 Aladdin Enterprises 는 FreeBSD 에서 작동하고 무료로 포스트스크립트처럼 동작하는 **Ghostscript** 를 가지고 있다. 고스트스크립트는 대부분의 포스트스크립트 파일을 읽을 수 있고 이들 페이지를 다양한 벤더의 일반 프린터를 포함하여 여러 장치로 보낼 수 있다. 고스트스크립트를 설치하고 특수한 텍스트 필터를 사용하여 일반 프린터를 포스트스크립트처럼 동작 시킬 수 있다.

고스트스크립트는 FreeBSD 포트 컬렉션에 있기 때문에 원한다면 포트 컬렉션에서 설치할 수 있다. 직접 패치하여 빌드 후 설치하는 것도 쉽다.

포스트스크립트를 시뮬레이트 하려면 포스트스크립트 파일을 감지하는 텍스트 필터가 있어야 한다. 그렇지 않으면 필터는 직접 파일을 프린터에 보낸다. 필터는 고스트스크립트로 파일을 프린터가 이해하도록 변환한다.

여기 예제가 있다: 다음 스크립트는 휴렛팩커드 데스크 젯 500 프린터를 위한 텍스트 필터다. 다른 프린터는 `-sDEVICE` 인자를 `gs`(고스트스크립트) 명령에 사용한다(`gs -h` 를 입력하면 현재 고스트스크립트가 지원하는 장치 리스트를 보여준다.).

```
#!/bin/sh
```

```

#
# ifhp - Print Ghostscript-simulated PostScript on a DeskJet 500
# Installed in /usr/local/libexec/ifhp

#
# Treat LF as CR+LF:
#
printf "\W033&k2G" || exit 2

#
# Read first two characters of the file
#
IFS="" read -r first_line
first_two_chars=`expr "$first_line" : '\W(..\W)'\`

if [ "$first_two_chars" = "%!" ]; then
    #
    # It is PostScript; use Ghostscript to scan-convert and print it.
    #
    # Note that PostScript files are actually interpreted programs,
    # and those programs are allowed to write to stdout, which will
    # mess up the printed output. So, we redirect stdout to stderr
    # and then make descriptor 3 go to stdout, and have Ghostscript
    # write its output there. Exercise for the clever reader:
    # capture the stderr output from Ghostscript and mail it back to
    # the user originating the print job.
    #
    exec 3>&1 1>&2
    /usr/local/bin/gs -dSAFER -dNOPAUSE -q -sDEVICE=djet500 \W
        -sOutputFile=/dev/fd/3 - && exit 0
else
    #
    # Plain text or HP/PCL, so just print it directly; print a form feed
    # at the end to eject the last page.
    #
    echo "$first_line" && cat && printf "\W033&IOH" &&

```

---

```
exit 0
```

```
fi
```

```
exit 2
```

마지막으로 `if` 문자열로 LPD 에게 필터를 알려줘야 한다:

```
:if=/usr/local/libexec/lfhp:
```

이로서 `lpr plain.text` 와 `lpr whatever.ps` 를 입력하여 성공적으로 출력할 수 있다.

### 9.4.1.4 변환필터

간단한 프린터설정에서 설명한 설정을 끝낸 후 원하는 파일포맷을 위해(ASCII 텍스트와 같은) 변환필터를 설치한다.

#### [변환필터에 대한 자세한 설명]

##### 1. 변환필터는 왜 설치하는가?

변환필터는 다양한 종류의 파일을 쉽게 출력한다. 예를 들면 TeX typesetting 시스템으로 많은 작업을 하고 포스트스크립트 프린터를 가지고 있다고 가정한다. 항상 TeX 에서 DVI 파일을 생성하기 때문에 DVI 파일을 포스트스크립트로 변환하기 전까지는 직접 출력할 수 없다. 명령순서는 다음과 비슷하다:

```
% dvips seaweed-analysis.dvi
```

```
% lpr seaweed-analysis.ps
```

DVI 파일 변환필터를 설치해서 각각의 변환과정을 LPD 로 건너뛸 수 있다. 이제 DVI 파일이 있을 때마다 한 단계를 건너뛸 수 있다:

```
% lpr -d seaweed-analysis.dvi
```

`-d` 옵션을 지정하여 DVI 파일을 변환하는 LPD 가 있다. 포맷과 변환옵션 섹션에 변환옵션이 나열되어 있다.

프린터가 지원하기를 원하는 각각의 변환옵션은 *변환필터*를 설치하고 `/etc/printcap` 에 경로 이

름을 지정하면 된다. 변환필터는 텍스트파일을 제외하고 프린터가 이해하는 포맷으로 파일을 변환하는 간단한 프린터설정을 위한 텍스트 필터와 같다.

## 2. 어떤 변환필터를 설치해야 되는가?

사용하려는 변환필터를 설치해야 된다. 많은 DVI 데이터를 출력하려면 DVI 변환필터가 필요하다. 풍부한 troff 를 출력하려면 troff 필터가 필요할 것이다.

다음은 LPD 와 동작하고 /etc/printcap 파일에서 관련된 문자 그리고 lpr 명령과 어떤 관계인지 요약한 표다:

파일 종류	/etc/printcap 문자열	lpr 옵션
cifplot	<i>cf</i>	<i>-c</i>
DVI	<i>df</i>	<i>-d</i>
plot	<i>gf</i>	<i>-g</i>
ditroff	<i>nf</i>	<i>-n</i>
FORTTRAN text	<i>rf</i>	<i>-f</i>
troff	<i>rf</i>	<i>-f</i>
raster	<i>vf</i>	<i>-v</i>
plain text	<i>if</i>	none, <i>-p</i> 또는 <i>-l</i>

우리가 예제에서 lpr -d 를 사용한다는 의미는 프린터에게 /etc/printcap 의 엔트리에 *df* 문자열이 필요하다는 것이다.

내용이 어떠한 FORTRAN text 와 plot 는 사용하지 않을 것이다. 사이트에서 사용자 필터를 설치해서 이런 포맷옵션에 새로운 의미를 부여할 수 있다. 예를 들어 Printerleaf 파일을 (Interleaf 데스크톱 출판 프로그램 파일) 직접 출력하려고 하지만 plot 파일은 절대 출력하지 않는다고 가정한다. *gf* 문자열로 Printerleaf 변환필터를 설치하고 유저들에게 lpr -g 는 Printerleaf 파일을 출력한다고 교육한다.

## 3. 변환필터 설치

변환필터는 기본 FreeBSD 영역 밖의 프로그램이기 때문에 /usr/local 에 설치된다.

/usr/local/libexec 디렉터리는 LPD 가 실행하는 프로그램들만 있는 일반적인 위치다; 일반 유

---

저는 이들을 실행할 필요가 없다.

변환필터를 사용하려면 /etc/printcap 의 원하는 프린터에 적절한 문자열로 경로 이름을 지정한다.

예제에서 우리는 프린터 이름 *bamboo* 엔트리에 DVI 변환필터를 추가한다. 이곳은 프린터 *bamboo* 를 위해 새로운 *df* 문자열로 /etc/printcap 파일을 다시 수정한 예제다.

```
#
# /etc/printcap for host rose - added df filter for bamboo
#
rattan|line|diablo|lp|Diablo 630 Line Printer:W
    :sh:sd=/var/spool/lpd/rattan:W
    :lp=/dev/lpt0:W
    :if=/usr/local/libexec/if-simple:

bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:W
    :sh:sd=/var/spool/lpd/bamboo:W
    :lp=/dev/ttyd5:ms#-parenb cs8 clocal crtscts:rw:W
    :if=/usr/local/libexec/psif:W
    :df=/usr/local/libexec/psdf:
```

DVI 필터는 /usr/local/libexec/psdf 이라는 쉘 스크립트다. 여기 쉘 스크립트가 있다:

```
#!/bin/sh
#
# psdf - DVI to PostScript printer filter
# Installed in /usr/local/libexec/psdf
#
# Invoked by lpd when user runs lpr -d
#
exec /usr/local/bin/dvips -f | /usr/local/libexec/lprps "$@"
```

이 스크립트는 잡을 출력하는 표준입력에 dvips 를 필터모드로(-f 인자) 실행한다. 그리고 LPD 를 인자로 붙여서 포스트스크립트 프린터필터 lprps 를(포스트스크립트 프린터에서 평범한 텍스트 출력하기 섹션을 본다) 실행한다. lprps 는 출력된 페이지를 계산하는데 이런 인자를 사용한



---

다.

#### 4. 더 많은 변환필터 예제

변환필터를 설치하는데 고정된 설정단계가 없기 때문에 더 많은 예제를 제공한다. 여러분의 필터를 만들기 위해 이 문서를 가이드처럼 사용한다. 괜찮다면 이들 예제를 직접 사용한다.

이 예제 스크립트는 휴렛팩커드 레이저젯 III-SI 프린터를 위한 도트방식(실제로 GIF 파일) 변환필터다:

```
#!/bin/sh
#
# hpvf - Convert GIF files into HP/PCL, then print
# Installed in /usr/local/libexec/hpvf

PATH=/usr/X11R6/bin:$PATH; export PATH
giftopnm | ppmtopgm | pgmtopbm | pbmtolj -resolution 300 ₩
    && exit 0 ₩
    || exit 2
```

GIF 파일을 이식 가능한 anymap 변환하고 변환된 것을 다시 graymap, bitmap 그리고 레이저젯/PCL 호환 데이터로 변환한다.

여기 위의 필터를 사용하는 프린터의 /etc/printcap 파일 엔트리가 있다:

```
#
# /etc/printcap for host orchid
#
teak|hp|laserjet|Hewlett Packard LaserJet 3Si:₩
    :lp=/dev/lpt0:sh:sd=/var/spool/lpd/teak:mx#0:₩
    :if=/usr/local/libexec/hpif:₩
    :vf=/usr/local/libexec/hpvf:
```

다음 스크립트는 포스트스크립트 프린터 *bamboo* 의 groff typesetting 시스템을 troff 데이터로 변환하는 필터다:

---

```
#!/bin/sh
#
# pstf - Convert groff's troff data into PS, then print.
# Installed in /usr/local/libexec/pstf
#
exec grops | /usr/local/libexec/lprps "$@"
```

위의 스크립트는 프린터와의 통신제어에 lprps 를 다시 사용한다. 프린터가 패러럴 포트에 있다면 이 스크립트를 사용한다:

```
#!/bin/sh
#
# pstf - Convert groff's troff data into PS, then print.
# Installed in /usr/local/libexec/pstf
#
exec grops
```

필터를 사용하기 위해 /etc/printcap 에 추가할 엔트리가 있다:

```
:tf=/usr/local/libexec/pstf:
```

여기 FORTRAN blush 에서 숙련자가 만든 예제가 있다. 직접 평범한 텍스트를 출력할 수 있는 프린터의 FORTRAN-텍스트 필터다. 우리는 프린터 *teak*에 이 필터를 설치한다:

```
#!/bin/sh
#
# hprf - FORTRAN text filter for LaserJet 3si:
# Installed in /usr/local/libexec/hprf
#
printf "\W033&k2G" && fpr && printf "\W033&l0H" &&
  exit 0
exit 2
```

프린터 *teak*이 필터를 사용하도록 /etc/printcap 에 라인을 추가한다:

---

```
:rf=/usr/local/libexec/hprf:
```

마지막으로 약간 복잡한 예제가 있다. 이전에 소개한 레이저젯 프린터 *teak*에 DVI 필터를 추가한다. 첫째 쉬운 부분으로 /etc/printcap 에 DVI 필터의 위치를 입력한다:

```
:df=/usr/local/libexec/hpdf:
```

이제 어려운 부분으로 필터를 만들기 위해 DVI 를 레이저젯/PCL 로 변환하는 프로그램이 필요하다. FreeBSD 포트 컬렉션은(포트 컬렉션을 본다) 이런 필터를 가지고 있다: dvi2xx 라는 이름의 패키지다. 이 패키지를 설치하면 우리에게 필요한 DVI 를 레이저젯 IIp, 레이저젯 III 와 레이저젯 2000 호환 코드로 바꾸는 프로그램 dviij2p 도 설치된다.

dviij2p 는 표준입력으로 읽지 못하기 때문에 hpdf 필터를 약간 복잡하게 만든다. 이것은 파일 이름으로 동작한다. 파일이름은 마지막이 .dvi 로 끝나기 때문에 /dev/fd/0 를 사용한 표준 입력은 문제가 있다. 우리는 임시파일 이름을(.dvi 로 끝나는) /dev/fd/0 에 링크하여(심볼릭 링크) dviij20 가 표준입력을 강제로 읽도록 하여 문제를 해결할 수 있다.

우리가 간과한 또 다른 사실은 임시링크에 /tmp 를 사용할 수 없다는 것이다. 심볼릭 링크는 bin 유저와 그룹의 소유다. 필터는 daemon 유저로 실행된다. 그리고 /tmp 디렉터리는 sticky bit 설정을 가지고 있다. 필터는 링크를 생성할 수 있지만 끝나고 정리할 수 없기 때문에 링크가 다른 유저에 속하게 되었을 때 삭제한다.

대신 필터는 스펙링 디렉터리인(/etc/printcap 에서 *sd* 문자열로 지정된) 현재 작업디렉터리에 심볼릭 링크를 만든다. 이곳은 /tmp 보다 여유 공간이 더 많기(종종) 때문에 필터가 동작하기에 가장 적절한 위치다.

마지막으로 여기 필터가 있다:

```
#!/bin/sh
#
# hpdf - Print DVI data on HP/PCL printer
# Installed in /usr/local/libexec/hpdf

PATH=/usr/local/bin:$PATH; export PATH

#
```

---

```
# Define a function to clean up our temporary files.  These exist
# in the current directory, which will be the spooling directory
# for the printer.
#
cleanup() {
    rm -f hpdf$$dvi
}

#
# Define a function to handle fatal errors: print the given message
# and exit 2.  Exiting with 2 tells LPD to do not try to reprint the
# job.
#
fatal() {
    echo "$@" 1>&2
    cleanup
    exit 2
}

#
# If user removes the job, LPD will send SIGINT, so trap SIGINT
# (and a few other signals) to clean up after ourselves.
#
trap cleanup 1 2 15

#
# Make sure we are not colliding with any existing files.
#
cleanup

#
# Link the DVI input file to standard input (the file to print).
#
ln -s /dev/fd/0 hpdf$$dvi || fatal "Cannot symlink /dev/fd/0"

#
```

---

```
# Make LF = CR+LF
#
printf "W033&k2G" || fatal "Cannot initialize printer"

#
# Convert and print. Return value from dvi2p does not seem to be
# reliable, so we ignore it.
#
dvi2p -M1 -q -e- dfhp$$dvi

#
# Clean up and exit
#
cleanup
exit 0
```

## 5. 자동변환: 다른 변환필터

위의 모든 변환필터는 다양한 출력환경에 적절하지만 사용자가 사용할 것을 지정해야 된다(lpr(1) 명령어 라인에). 특히 사용자가 컴퓨터와 익숙하지 않은 경우 필터옵션을 지정하는 것은 귀찮은 일이다. 설상가상으로 정확하지 않은 필터옵션은 파일종류와 다른 필터를 실행하고 프린터가 많은 종이를 낭비하게 된다.

모든 변환필터를 설치하는 것보다 파일종류를 감지해서 자동으로 정확한 변환필터를 실행하는 텍스트필터를 설치할 수 있다. file 같은 툴이 도움이 될 것이다. 물론 파일종류를 결정하는 것이 어떤 파일종류에서는 어렵기 때문에 이 파일들을 위한 변환필터를 제공할 수 있다.

FreeBSD 포트 컬렉션에는 자동으로 변환하는 apsfiler 라는 텍스트필터가 있다. 이것은 평범한 텍스트와 포스트스크립트 그리고 DVI 파일을 식별하여 적절히 변환해서 출력할 수 있다.

### 9.4.1.5 출력(output) 필터

LPD 스플링 시스템은 아직 우리가 설명하지 않은 다른 종류의 필터를 지원한다: *출력필터*. 출력필터는 텍스트필터처럼 평범한 텍스트를 출력하는데 사용되지만 매우 단순하다. 텍스트

---

필터 대신 출력필터를 사용하면 다음과 같은 이점이 있다.

- LPD 는 잡에서 파일을 한번에 하나씩 처리하지 않고 한번에 전체작업을 처리한다.
- LPD 는 잡에서 출력필터가 파일의 시작과 끝을 감지하도록 어떤 표시도 하지 않는다.
- LPD 는 필터에 유저 로그인이나 호스트를 적용하지 않기 때문에 계산을 하지 않는다. 그래서 오직 두개의 인자만 가진다.

`filter-name -width -length`

*width* 는 *pw* 문자열 *length* 는 *pl* 문자열 값이다.

출력필터가 단순하더라도 주의한다. 잡의 각 파일을 다른 페이지에서 시작하려면 출력필터는 동작하지 않는다. 대신 텍스트필터(입력필터로 알려진) 사용한다; 자세한 내용은 텍스트필터 설치하기 섹션을 본다. 게다가 특수 플래그문자를 적용하여 신호를 보내려면 byte 스트림을 체크해야 되고 LPD 의 이점을 얻기 위해 신호를 보내야 되는 출력필터는 실제로 더 복잡하다.

그렇지만 헤더페이지를 출력하기 원하고 escape 시퀀스를 보내야 되거나 헤더페이지를 출력하기 위해 다른 초기화 문자열이 필요하다면 출력필터가 필요하다(LPD 는 출력필터에게 유저나 호스트정보를 주지 않기 때문에 유저계정의 요청으로 헤더페이지를 계산하려면 출력필터가 필요 없다).

LPD 는 현대의 프린터에서 출력필터와 텍스트 또는 다른 필터를 사용할 수 있다. 이러한 경우 LPD 는 헤더페이지를(헤더페이지 섹션을 보아라) 출력하기 위해서만 출력필터를 실행한다. 그리고 LPD 는 필터에 2 바이트를 보내서 출력필터를 정지시킬 수 있다: ASCII 001 뒤에 오는 ASCII 031. 출력필터가 이 2 바이트(031, 001)를 보게 되면 SIGSTOP를 보내서 직접 정지시킨다. LPD 가 다른 필터실행을 끝냈을 때 SIGCONT를 보내서 출력필터를 다시 시작한다.

텍스트필터가 아닌 출력필터가 있고 LPD 가 평범한 텍스트에서 동작한다면 LPD 는 이 잡을 수행하는데 출력필터를 사용한다. 이전 상태에서 출력필터는 종이 공급이나 다른 종이를 전진시키지 않고 순서대로 잡의 각 파일을 출력한다. 대부분의 경우 텍스트필터가 필요하다.

---

우리가 앞서 텍스트필터로 설명한 lpf 프로그램도 출력필터처럼 실행할 수 있다. 빠른 출력 필터가 필요하지만 탐지 바이트를 작성하지 않고 코드신호를 보내지 않으려면 lpf 를 사용한 다. 프린터가 필요로하는 초기화 코드를 제어하도록 lpf 를 쉘 스크립트로 감쌀 수 있다.

#### 9.4.1.6 lpf: 텍스트필터

FreeBSD 바이너리 배포판 프로그램 /usr/libexec/lpr/lpf 는 출력을(lpr -i 명령) 들여쓰기, 원래 문자 적용(lpr -l 명령), 잡에서 출력위치를 백스페이스와 탭으로 조정할 수 있으며 출력 된 페이지를 계산할 수 있는 텍스트필터다(입력필터). 또한 출력필터처럼 동작할 수 있다.

lpf 는 많은 프린트 환경에 적당하다. 프린터에 초기화 시퀀스를 보낼 수 없지만 필요한 초기화를 쉘 스크립트로 작성해서 lpf 를 실행하기 쉽다.

lpf 가 페이지계산을 정확하게 하려면 /etc/printcap 파일의 pw와 pl 문자열에 정확한 값을 입력해야 된다. 한 페이지에 적당한 텍스트 양과 유저가 출력한 페이지 양을 결정하기 위해 이런 값을 사용한다. 프린터 계산에 관한 더 많은 정보는 프린터 사용량 계산을 본다.

### 9.4.2 헤더페이지

많은 유저가 있고 모든 유저가 여러 프린터를 사용한다면 헤더페이지가 필요한지 생각할 것이다.

배너나 *burst* 페이지로 알고 있는 헤더페이지로 누구의 잡인지 확인할 수 있다. 헤더페이지 는 크고 볼드체로 가장자리에 장식이 되어 있어서 출력되기 때문에 출력물에서 헤더페이지 는 실제문서와 떨어져 있을 것이다. 헤더페이지는 유저가 잡을 빨리 찾게 한다. 헤더페이지 의 가장 큰 단점은 모든 잡마다 출력해야 되지만 이들의 필요성은 잠시뿐이고 결과적으로 재활용 통이나 휴지통에 버려진다(헤더페이지는 잡의 각 파일마다 생성되지 않고 각 잡마다 생성되기 때문에 종이 낭비가 덜하다).

프린터가 직접평범한 텍스트를 출력할 수 있다면 LPD 시스템은 자동으로 헤더페이지를 제공할 수 있다. 포스트스크립트 프린터를 가지고 있다면 헤더페이지를 생성하는 부수적인 프로그램이 필요하다; 포스트스크립트 프린터에서 헤더페이지를 본다(9.4.2.4 장).

---

### 9.4.2.1 헤더페이지 사용

간단한 프린터설정 섹션에서 `/etc/printcap` 파일에 `sh` 문자열을(의미는 "헤더페이지 출력 안 함") 지정하여 헤더페이지를 꺼두었다. 헤더페이지를 사용하려면 그냥 `sh` 문자열을 삭제한다.

프린터에 초기화 문자열을 보내기 위해 출력필터를 제공해야 될 것이다. 여기 휴렛팩커드 PCL 호환 프린터를 위한 출력필터 예제가 있다:

```
#!/bin/sh
#
# hpof - Output filter for Hewlett Packard PCL-compatible printers
# Installed in /usr/local/libexec/hpof

printf "W033&k2G" || exit 2
exec /usr/libexec/lpr/lpf
```

`of` 문자열로 출력필터 경로를 지정한다. 더 많은 정보는 출력필터 섹션을 본다.

우리가 전에 소개한 프린터 `teak`의 `/etc/printcap` 파일 예제가 있다; 헤더페이지를 사용하고 위의 출력필터를 추가한다:

```
#
# /etc/printcap for host orchid
#
teak|hp|laserjet|Hewlett Packard LaserJet 3Si:W
    :lp=/dev/lpt0:sd=/var/spool/lpd/teak:mx#0:W
    :if=/usr/local/libexec/hpif:W
    :vf=/usr/local/libexec/hpvf:W
    :of=/usr/local/libexec/hpof:
```

이제 사용자가 `teak`로 잡을 출력할 때 헤더페이지도 출력된다. 사용자가 출력물을 찾는데 시간이 들어도 된다면 `lpr -h`로 헤더페이지 감출 수 있다; 더 많은 `lpr(1)` 옵션은 헤더페이지 옵션 섹션을 본다.

**Note:** LPD는 헤더페이지를 출력한 후 폼 피드(feed) 문자를 출력한다. 프린터가 페



---

이지를 배출하기 위해 다른 문자나 문자열을 사용한다면 /etc/printcap 에 ff 문자열로 지정한다.

### 9.4.2.2 헤더페이지 제어

헤더페이지를 사용하면 LPD 는 페이지에 가득 찬 커다란 문자로 유저와 호스트 그리고 출력 잡을 확인할 수 있는 긴 헤더를 생성한다. 여기 예제가 있다(호스트 rose 에서 kelly 가 출력한 잡 이름은 outline 이다.):

```

k
k
k
k  k    eeee
k  k    e  e
k k    eeeee
kk k    e
k  k    e  e
k  k    eeee    |||    |||
                     y  y
                     y  y
                     y  y
                     y  y
                     y  yy
                     yyy y
                      y
                      y
                      yyy

        t
        t
0000  u  u  tttt    ||    i
o  o  u  u  t
o  o  u  u  t
o  o  u  u  t
o  o  u  uu  t  t    |||    iii    n  nnn    eeee
0000  uu u  tt      n  nn  n  e  e
                     n  n  eeeee
                     n  n  e
                     n  n  e  e
                     n  n  eeee

r rrr    0000    ssss    eeee
rr  r    o  o    s  s    e  e
r        o  o    ss    eeeee
r        o  o    ss    e
r        o  o    s  s    e  e
r        0000    ssss    eeee

Job:  outline
Date: Sun Sep 17 11:04:58 1995

```

---

LPD 는 이 텍스트 다음에 폼 피드(from feed)를 덧붙이기 때문에 잡은 새로운 페이지에서 시작된다(원하는 프린터의 /etc/printcap 엔트리에 *sf* 문자열을(form feed 제거) 지정 하지 않았다면) 새로운 페이지에서 시작된다.

LPD 가 짧은 헤더를 생성하기를 원하면 /etc/printcap 파일에 *sh* 문자열을 지정한다. 헤더 페이지는 아래와 비슷할 것이다:

```
rose:kelly Job: outline Date: Sun Sep 17 11:07:51 1995
```

또한 기본적으로 LPD 는 헤더페이지를 먼저 출력하고 잡을 출력한다. 반대로 하려면 /etc/printcap 에 *h/* 문자열을(헤더가 마지막) 지정한다.

### 9.4.2.3 헤더페이지 계산

LPD 에 내장된 헤더페이지를 사용하면 출력물을 계산할 때 특정 패러다임이 수행된다: 헤더 페이지는 사용료에서 공제해야 된다.

왜냐하면?

계산해야 되는 헤더페이지가 언제 출력되는지 제어하는 유일한 프로그램이 출력필더고 유저 나 호스트 정보 또는 계산파일을 제공하지 않기 때문에 프린터 사용량을 누구에게 부과할지 도움이 되지 못한다. 또한 유저가 `lpr -h` 로 헤더페이지를 감출 수 있기 때문에 텍스트필터 나 변환필터에 "페이지 한 장"을 추가할 수 없다. 유저들에게 그들이 출력하지 않은 헤더페이지 요금을 부과할 수 있다. 기본적으로 `lpr -h` 은 환경적으로 선호하는 옵션이지만 이 옵션을 사용하도록 권장할 수 없다.

각 필터마다 헤더페이지를(그래서 유저들에게 요금을 부과할 수 있다) 생성하기에 아직 부족하다. 유저가 `lpr -h` 옵션으로 헤더페이지를 감추기를 원하지만 LPD 는 필터에게 `-h` 옵션의 정보를 주지 않기 때문에 헤더페이지는 계속 출력되고 요금이 부과된다.

그래서 사용할 수 있는 옵션은 다음과 같다.

- LPD 패러다임을 허용하고 헤더페이지를 무료로 한다.

- LPD 대신 LPRng 같은 프로그램을 설치한다. 표준 스플러 대안 섹션에서 LPD 대신 사용할 수 있는 다른 스플링 소프트웨어에 관한 더 많은 정보를 준다.
- 고성능 출력필터작성. 보통 출력필터는 프린터 초기화나 어떤 간단한 문자변환 이외의 의미는 갖지 않는다. 이런 필터는 헤더페이지와 평범한 텍스트작업에만(텍스트(입력) 필터가 없을 때) 적당하다. 그러나 평범한 텍스트 잡을 위한 텍스트필터가 있다면 LPD 는 헤더페이지용으로 출력필터를 시작한다. 그리고 출력필터는 어떤 유저나 호스트에게 요금을 부과할지 결정할 수 있도록 LPD 가 생성한 헤더페이지를 해석할 수 있다. 이 방법의 유일한 문제는 출력필터가 어떤 계산 파일을 사용할지 (*af* 문자열에 파일 이름을 적용하지 않는다) 아직 모르지만 유명한 계산파일을 가지고 있다면 출력필터에 넣을 수 있다. 문장 해석단계를 쉽게 하도록 */etc/printcap* 에 *sh* 문자열(짧은 헤더) 사용한다. 그리고 다시 더 많은 문제가 발생되면 유저는 헤더페이지를 무료로 해주는 더욱 관대한 시스템관리자를 찾을 것이다.

#### 9.4.2.4 포스트스크립트 프린터에서 헤더페이지

위에서 설명했듯이 LPD 는 다양한 프린터에 적절하도록 평범한 텍스트 헤더페이지를 생성할 수 있다. 물론 포스트스크립트는 직접 평범한 텍스트를 출력할 수 없기 때문에 LPD 의 헤더페이지 기능을 거의 사용하지 못한다.

헤더페이지를 출력할 수 있는 방법은 모든 변환필터와 텍스트필터로 헤더페이지를 생성하는 것이다. 필터는 적절한 헤더페이지를 생성하기 위해 유저와 호스트 인자를 사용한다. 이런 방법의 단점은 유저가 *lpr -h* 로 출력하더라도 항상 헤더페이지를 출력하는 것이다.

이 방법에 대해 알아보자. 다음 스크립트는 3 개의 인자를(유저 로그인 이름, 호스트 이름 그리고 잡 이름) 가지고 간단한 포스트스크립트 헤더페이지를 만든다:

```
#!/bin/sh
#
# make-ps-header - make a PostScript header page on stdout
# Installed in /usr/local/libexec/make-ps-header
#
#
# These are PostScript units (72 to the inch).  Modify for A4 or
```

---

```
# whatever size paper you are using:
#
page_width=612
page_height=792
border=72

#
# Check arguments
#
if [ $# -ne 3 ]; then
    echo "Usage: `basename $0` <user> <host> <job>" 1>&2
    exit 1
fi

#
# Save these, mostly for readability in the PostScript, below.
#
user=$1
host=$2
job=$3
date=`date`

#
# Send the PostScript code to stdout.
#
exec cat <<EOF
%!PS

%
% Make sure we do not interfere with user's job that will follow
%
save

%
% Make a thick, unpleasant border around the edge of the paper.
%
```

---

```

$border $border moveto
$page_width $border 2 mul sub 0 rlineto
0 $page_height $border 2 mul sub rlineto
currentscreen 3 -1 roll pop 100 3 1 roll setscreen
$border 2 mul $page_width sub 0 rlineto closepath
0.8 setgray 10 setlinewidth stroke 0 setgray

%
% Display user's login name, nice and large and prominent
%
/Helvetica-Bold findfont 64 scalefont setfont
$page_width ($user) stringwidth pop sub 2 div $page_height 200 sub moveto
($user) show

%
% Now show the boring particulars
%
/Helvetica findfont 14 scalefont setfont
/y 200 def
[ (Job:) (Host:) (Date:) ] {
200 y moveto show /y y 18 sub def }
forall

/Helvetica-Bold findfont 14 scalefont setfont
/y 200 def
[ ($job) ($host) ($date) ] {
270 y moveto show /y y 18 sub def
} forall

%
% That is it
%
restore
showpage
EOF

```

---

이제 각 변환필터와 텍스트필터는 첫 번째 헤더페이지를 생성하기 위해 이 스크립트를 호출해서 유저의 잡을 출력한다. 여기 이 문서의 앞부분에서 설명했던 헤더페이지를 만들도록 수정된 DVI 변환필터가 있다:

```
#!/bin/sh
#
# psdf - DVI to PostScript printer filter
# Installed in /usr/local/libexec/psdf
#
# Invoked by lpd when user runs lpr -d
#

orig_args="$@"

fail() {
    echo "$@" 1>&2
    exit 2
}

while getopts "x:y:n:h:" option; do
    case $option in
        x|y) ;; # Ignore
        n)   login=$OPTARG ;;
        h)   host=$OPTARG ;;
        *)   echo "LPD started `basename $0` wrong." 1>&2
            exit 2
            ;;
    esac
done

[ "$login" ] || fail "No login name"
[ "$host" ] || fail "No host name"

( /usr/local/libexec/make-ps-header $login $host "DVI File"
  /usr/local/bin/dvips -f ) | eval /usr/local/libexec/lprps $orig_args
```

---

유저와 호스트 이름을 확인하기 위해 필터가 인자리스트를 어떻게 분석하는지 알아보자. 다른 변환필터 분석과 동일하다. 그렇지만 텍스트필터는 약간 다른 인자설정이 필요하다(필터가 어떻게 작동하는지 섹션을 본다).

이전 스키마에서 우리가 아주 간단하게 언급했듯이 lpr에는 "헤더페이지 감추기"를 사용하지 못하게 하는 옵션이 있다(-h 옵션). 유저가 요금을 아끼려고(또는 헤더페이지에 금액을 부과하고 싶다면 조금만 받는다) 하지만 모든 필터가 모든 잡마다 헤더페이지를 출력하기 때문에 유저들은 그렇게 하지 못한다.

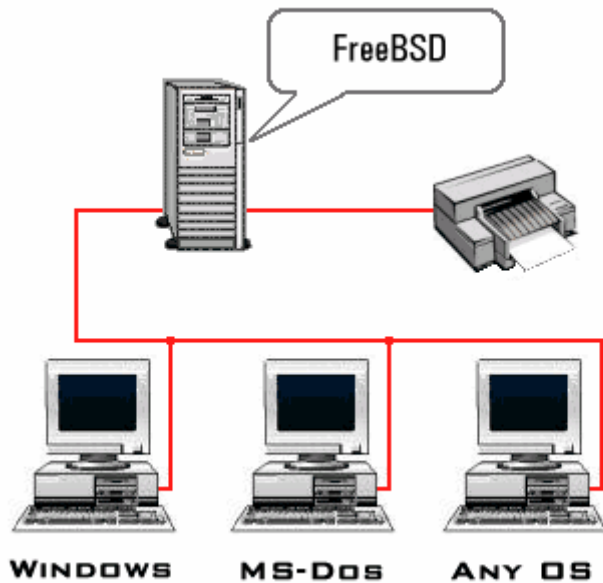
유저가 잡마다 헤더페이지 끄는 것을 허용하려면 헤더페이지 계산 섹션에서 설명한 요령을 사용해야 된다: LPD가 생성하는 헤더페이지를 분석해서 포스트스크립트 버전을 생성하는 출력필터를 작성한다. 유저가 lpr -h로 잡을 보내면 LPD는 헤더페이지를 생성하지 않고 출력필터도 마찬가지로 헤더페이지를 생성하지 않는다. 그렇지 않으면 출력필터는 텍스트를 LPD로부터 읽어서 프린터에게 적당한 포스트스크립트 헤더페이지 코드를 보낸다.

시리얼 라인에 포스트스크립트 프린터가 있다면 출력필터 psof와 위의 lprps를 사용할 수 있다. psof는 헤더페이지 사용료를 부과하지 않는다.

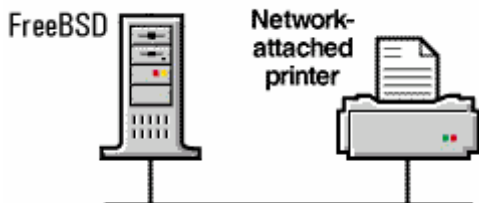
### 9.4.3 네트워크 프린트

FreeBSD는 네트워크 프린트를 지원한다: 잡을 원격 프린터로 보낸다. 네트워크 프린트는 일반적으로 두 가지 다른 의미를 가지고 있다:

- 원격 호스트의 프린터를 사용한다. 일반적인 시리얼이나 패러럴 인터페이스를 가지고 있는 호스트에 프린터를 설치한다. 그리고 네트워크의 다른 호스트에서 프린터를 사용할 수 있도록 LPD를 설정한다. 원격 호스트에 프린터설치 섹션에서 어떻게 설치하는지 설명한다.



- 네트워크에 직접 연결되어있는 프린터에 사용. 어떤 프린터는 일반적인 시리얼이나 패러럴 인터페이스와 네트워크 인터페이스도 가지고 있다. 이런 프린터는 다음과 같이 작동한다:



- ◆ LPD 프로토콜과 원격호스트의 큐 잡도 이해할 것이다. 이 경우 일반 호스트에서 LPD 가 작동하는 것처럼 동작한다. 원격 호스트에 프린터 설치하기 섹션을 따라서 이런 프린터를 설치한다.
- ◆ 네트워크 연결도 지원할 것이다. 이 경우 잡을 스펙링해서 프린터에 보내는 역할을 하는 호스트에 프린터를 붙여서 네트워크에 연결한다. 네트워크 인터페이스를 가진 프린터 섹션에서 이런 프린터를 설치하는 몇가지 방법을 소개한다.



---

### 9.4.3.1 원격 호스트에 설치된 프린터

LPD 스펙링 시스템은 LPD 가(또는 LPD 와 호환되는) 실행중인 다른 호스트에 잡을 보내는 내장된 지원시스템을 가지고 있다. 이 기능은 한대의 호스트에 프린터를 설치하고 다른 호스트에서 접근할 수 있도록 한다. 또한 네트워크 인터페이스를 가지고 LPD 프로토콜을 이해하는 프린터에서도 작동한다.

이런 종류의 원격 프린트를 활성화 시키려면 간단한 프린터 설치 섹션에서 설명한 방법으로 호스트에 프린터를 설치한다. 발전된 프린터 설정에서 필요한 설정을 한다. 활성화시킨 LPD 기능으로 프린터가 제대로 동작하는지 테스트한다. 또한 *원격호스트에서*(원격 프린터에서 출력 제한하기를 본다) LPD 서비스를 이용하는 인증을 *로컬 호스트가* 가지고 있는지 확인한다.

LPD 와 호환되는 네트워크 인터페이스를 가진 프린터를 사용한다면 아래에서 설명하는 프린터 호스트는 프린터 자신이고 프린터 이름은 프린터에 설정한 이름이다. 프린터나 프린터 네트워크 인터페이스 매뉴얼 문서를 본다.

**Tip:** 휴렛패커드 레이저젯을 사용하고 있다면 *text* 라는 프린터가 자동으로 LF 를 CRLF 로 변환하기 때문에 *hplj* 스크립트가 필요 없다.

그리고 다른 호스트에서 프린터를 사용하려면 */etc/printcap* 파일에 다음과 같은 엔트리를 추가한다:

- ① 원하는 이름의 엔트리를 추가한다. 단순히 프린터 호스트의 엘리어스 같은 이름을 사용할 수 있다.
- ② *lp* 문자열은 확실히 공란으로 둔다(*:lp=*).
- ③ 스펙링 디렉터리를 만들고 *sd* 문자열로 위치를 지정한다. LPD 는 프린터 호스트에 데이터를 보내기 전에 데이터를 이곳에 저장한다.
- ④ *rm* 문자열에 프린터 호스트 이름을 입력한다.
- ⑤ *rp* 문자열의 프린터 호스트란에 프린터 이름을 입력한다.

---

/etc/printcap 파일에 변환필터와 페이지 넓이 등을 지정할 필요가 없다.

여기 예제가 있다. 호스트 rose 는 *bamboo*와 *rattan* 두 대의 프린터를 가지고 있다. 그리고 orchid 호스트의 유저가 출력할 수 있도록 한다. orchid 의(헤더페이지 사용 섹션에서) /etc/printcap 파일은 다음과 같고 이 파일에는 프린터 *teak*의 엔트리를 가지고 있다; 호스트 rose 에 있는 두 대의 프린터 엔트리를 추가한다:

```
#
# /etc/printcap for host orchid – added (remote) printers on rose
#
#
# teak is local; it is connected directly to orchid:
#
teak|hp|laserjet|Hewlett Packard LaserJet 3Si:W
    :lp=/dev/lpt0:sd=/var/spool/lpd/teak:mx#0:W
    :if=/usr/local/libexec/ifhp:W
    :vf=/usr/local/libexec/vfhp:W
    :of=/usr/local/libexec/ofhp:

#
# rattan is connected to rose; send jobs for rattan to rose:
#
rattan|line|diablo|lp|Diablo 630 Line Printer:W
    :lp=:rm=rose:rp=rattan:sd=/var/spool/lpd/rattan:

#
# bamboo is connected to rose as well:
#
bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:W
    :lp=:rm=rose:rp=bamboo:sd=/var/spool/lpd/bamboo:
```

그리고 orchid 에 스푼링 디렉터리를 만들어야 한다:

```
# mkdir -p /var/spool/lpd/rattan /var/spool/lpd/bamboo
```

---

```
# chmod 770 /var/spool/lpd/rattan /var/spool/lpd/bamboo
# chown daemon:daemon /var/spool/lpd/rattan /var/spool/lpd/bamboo
```

이제 orchid의 유저는 *rattan*과 *bamboo*로 출력할 수 있다. 예를 들어 orchid 유저가 다음과 같이 입력했다면 orchid의 LPD 시스템은 DVI 잡을 */var/spool/lpd/bamboo* 스펴링 디렉터리로 복사한다.

```
% lpr -P bamboo -d sushi-review.dvi
```

호스트 *rose*가 *bamboo*의 스펴링 디렉터리에 잡을 보내자마자 두 개의 LPD는 파일을 *rose*에게 전달한다. 파일은 출력될 때까지 *rose*의 큐에서 대기하고 *rose*의 DVI에서 포스트스크립트로 변환된다(*bamboo*가 포스트스크립트 프린터기 때문에).

### 9.4.3.2 네트워크 인터페이스를 가진 프린터

프린터용 네트워크 인터페이스 카드를 구입할 때 두 가지 버전을 구입할 수 있다: 하나는 스펴러를 에뮬레이트하는 버전(더 비싼 버전)이고 다른 하나는 시리얼이나 패러럴 포트(싼 버전)를 사용할 때처럼 데이터를 보내기만하는 버전이 있다. 이번 섹션은 가격이 싼 버전을 어떻게 활용하는지 설명한다. 더 비싼 버전에 대해서는 앞의 원격호스트에 프린터 설치하기를 본다.

*/etc/printcap* 파일에 사용하는 시리얼이나 패러럴 인터페이스(시리얼 인터페이스를 사용하면), 보드율, 흐름 제어, 탭 지연, 새로운 라인 변환 등을 명시한다. 그렇지만 TCP/IP나 다른 네트워크 포트로 프린터를 연결하도록 지정하지 못한다.

네트워크 프린터에 데이터를 보내려면 텍스트와 변환필터라고 부르는 통신 프로그램을 개발해야 된다. 여기 관련된 예제가 있다; 스크립트 *netprint*는 표준입력으로 모든 데이터를 받아서 네트워크에 연결된 프린터로 보낸다. 프린터 호스트 이름을 *netprint*의 첫번째 인자로 지정하고 포트 번호를 두번째 인자로 지정한다. 많은 프린터가 쌍방향 통신을 지원하기 때문에 쌍방향 통신의 장점을(프린터 상태, 계산하기 등) 원하겠지만 이것은 단방향 통신만 지원한다(FreeBSD에서 프린터로).

```
#!/usr/bin/perl
#
# netprint - Text filter for printer attached to network
```

```

# Installed in /usr/local/libexec/netprint
#
$#ARGV eq 1 || die "Usage: $0 <printer-hostname> <port-number>";

$printer_host = $ARGV[0];
$printer_port = $ARGV[1];

require 'sys/socket.ph';

($ignore, $ignore, $protocol) = getprotobyname('tcp');
($ignore, $ignore, $ignore, $ignore, $address)
    = gethostbyname($printer_host);

$sockaddr = pack('S n a4 x8', &AF_INET, $printer_port, $address);

socket(PRINTER, &PF_INET, &SOCK_STREAM, $protocol)
    || die "Can't create TCP/IP stream socket: $!";
connect(PRINTER, $sockaddr) || die "Can't contact $printer_host: $!";
while (<STDIN>) { print PRINTER; }
exit 0;

```

그리고 이 스크립트를 다양한 필터에 사용할 수 있다. 네트워크에 연결된 Diablo 750-N 라인 프린터를 가지고 있다고 가정한다. 프린터는 데이터를 포트 번호 5100으로 출력한다. 프린터의 호스트이름은 scrivener이다. 여기 이 프린터용 텍스트필터가 있다:

```

#!/bin/sh
#
# diablo-if-net - Text filter for Diablo printer `scrivener' listening
# on port 5100. Installed in /usr/local/libexec/diablo-if-net
#
exec /usr/libexec/lpr/lpf "$@" | /usr/local/libexec/netprint scrivener 5100

```

#### 9.4.4 프린터 사용량 제한

---

이번 섹션은 프린터 사용량 제한에 대해 설명한다. LPD 시스템은 누가 프린터에 논리적 또는 원격으로 접근하여 여러 장을 출력할 수 있는지, 얼마나 큰 용량을 출력하고 몇개의 프린터 큐를 사용하는지 제어할 수 있다.

#### 9.4.4.1 여러장 출력 제한

LPD 시스템은 사용자가 파일을 여러 장씩 쉽게 출력할 수 있도록 한다. 예를 들어 유저는 `lpr -#5` 로 각 파일을 5 장씩 출력할 수 있다. 이 기능이 편리한지는 관리자에게 달려있다.

여러 장씩 출력하는 것이 부당하다면 `/etc/printcap` 파일에 `sc` 문자열을 추가하여 `lpr(1) -#` 옵션을 사용하지 못하게 할 수 있다. 유저가 `-#` 옵션을 사용하려고 하면 다음과 같은 메시지를 보게 된다:

```
lpr: multiple copies are not allowed
```

원격에서(원격 호스트에 설치된 프린터 섹션을 본다) 프린터를 사용하도록 설정 한다면 `/etc/printcap` 파일에 `sc` 문자열이 필요하고 그렇지 않으면 유저는 다른 호스트에서 여러 장씩 출력할 수 있다.

여기 예제가 있다. 이것은 호스트 `rose` 의 `/etc/printcap` 파일이다. 프린터 `rattan` 은 사용량이 별로 없기 때문에 여러 장씩 출력할 수 있도록 허용하지만 레이저 프린터 `bamboo` 는 사용량이 많기 때문에 `sc` 문자열을 추가하여 여러 장씩 출력하지 못하게 한다:

```
#
# /etc/printcap for host rose - restrict multiple copies on bamboo
#
rattan|line|diablo|lp|Diablo 630 Line Printer:W
    :sh:sd=/var/spool/lpd/rattan:W
    :lp=/dev/lpt0:W
    :if=/usr/local/libexec/if-simple:

bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:W
    :sh:sd=/var/spool/lpd/bamboo:sc:W
    :lp=/dev/ttyd5:ms#--parenb cs8 clocal crtscts:rw:W
    :if=/usr/local/libexec/psif:W
```

---

```
:df=/usr/local/libexec/psdf:
```

또한 호스트 orchid 의 /etc/printcap 에도(그리고 프린터 teak 에도 여러 장씩 출력하지 못하게 한다) sc 문자열을 추가해야 된다:

```
#
# /etc/printcap for host orchid - no multiple copies for local
# printer teak or remote printer bamboo
teak|hp|laserjet|Hewlett Packard LaserJet 3Si:W
    :lp=/dev/lpt0:sd=/var/spool/lpd/teak:mx#0:sc:W
    :if=/usr/local/libexec/ifhp:W
    :vf=/usr/local/libexec/vfhp:W
    :of=/usr/local/libexec/ofhp:

rattan|line|diablo|lp|Diablo 630 Line Printer:W
    :lp=:rm=rose:rp=rattan:sd=/var/spool/lpd/rattan:

bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:W
    :lp=:rm=rose:rp=bamboo:sd=/var/spool/lpd/bamboo:sc:
```

sc 문자열로 lpr -# 사용을 막을 수 있지만 사용자가 lpr(1)을 여러번 사용하거나 같은 파일을 아래와 같이 여러번 출력하는 것은 막지 못한다:

```
% lpr forsale.sign forsale.sign forsale.sign forsale.sign forsale.sign
```

이런 식의 사용법을(이런 명령을 무시하는 것을 포함하여) 제한하는 다양한 방법이 있다.

#### 9.4.4.2 프린터 접근 제한

유닉스 그룹 메커니즘과 /etc/printcap 의 rg 문자열로 누가 어떤 프린터로 출력할 수 있는지 제어할 수 있다. 프린터를 사용하려는 유저를 특정 그룹에 넣고 rg 문자열에 그룹 이름을 적는다.

그룹에(root 를 포함한) 포함되지 않은 유저가 제한된 프린터에 출력을 보내면: “lpr: Not a member of the restricted group” 메시지가 나타난다.

---

*sc* 문자열(여러 장씩 출력 제한)로 제한하는 것이 적당하다고 생각되면(원격 호스트에 설치된 프린터 섹션을 본다) 프린터를 사용해야 되는 원격 호스트에서 *rg* 문자열을 지정해야 된다.

예를 들어 프린터 *rattan*은 누구나 사용할 수 있지만 그룹 *artists*만 *bamboo*를 사용할 수 있도록 한다. 여기 호스트 *rose*의 */etc/printcap*와 유사한 파일이 있다:

```
#
# /etc/printcap for host rose - restricted group for bamboo
#
rattan|line|diablo|lp|Diablo 630 Line Printer:W
    :sh:sd=/var/spool/lpd/rattan:W
    :lp=/dev/lpt0:W
    :if=/usr/local/libexec/if-simple:

bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:W
    :sh:sd=/var/spool/lpd/bamboo:sc:rg=artists:W
    :lp=/dev/ttyd5:ms#--parenb cs8 clocal crtscts:rw:W
    :if=/usr/local/libexec/psif:W
    :df=/usr/local/libexec/psdf:
```

다른 */etc/printcap* 예제 파일은(호스트 *orchid*) 그대로 둔다. 물론 *orchid*에 있는 사람은 *bamboo*로 출력할 수 있다. 어쨌든 프린터를 사용해야 되는 사람만 허가했다.

**Note:** 프린터마다 하나의 그룹만 제한할 수 있다.

### 9.4.4.3 잡 크기 제어

프린터를 사용하는 유저가 많이 있다면 유저가 사용할 수 있는 파일크기를 제한할 필요가 있을 것이다. 어쨌든 스프링 디렉터리로 사용할 많은 여유공간과 다른 유저 잡을 위한 디렉터리도 파일시스템에 필요하다.

LPD는 잡에서 최대 파일크기를 *mx* 문자열로 제한할 수 있다. 단위는 1024 바이트 *BUFSIZ* 블록이다. 이 문자열을 0으로 설정하면 파일 크기에 제한이 없지만 *mx* 문자열이

---

지정되지 않았다면 기본 제한값 1000 블록이 적용된다.

**Note:** 제한은 전체 잡 크기가 아닌 *파일*에만 적용된다.

LPD 는 프린터에 걸어둔 제한보다 큰 파일을 거부하지 않는다. 대신 출력할 큐에 쌓이는 파일을 제한한다. 나머지는 버릴 것이다. 이것이 적당한지 생각해 본다.

우리는 우리의 예제 프린터 *rattan* 과 *bamboo* 에 제한을 추가한다. 왜냐하면 이런 예술가들의 포스트스크립트 파일은 너무 큰 경향이 있기 때문에 5 MB 로 제한할 것이다. 평범한 텍스트 라인프린터는 제한하지 않는다:

```
#
# /etc/printcap for host rose
#
#
# No limit on job size:
#
rattan|line|diablo|lp|Diablo 630 Line Printer:W
      :sh:mx#0:sd=/var/spool/lpd/rattan:W
      :lp=/dev/lpt0:W
      :if=/usr/local/libexec/if-simple:

#
# Limit of five megabytes:
#
bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:W
      :sh:sd=/var/spool/lpd/bamboo:sc:rg=artists:mx#5000:W
      :lp=/dev/ttyd5:ms#-parenb cs8 clocal crtscts:rw:W
      :if=/usr/local/libexec/psif:W
      :df=/usr/local/libexec/psdf:
```

이 제한은 로컬 유저에게만 적용된다. 원격에서 프린터를 사용하도록 설정하였다면 원격 유저에게는 적용되지 않는다. 따라서 원격 /etc/printcap 파일에도 *mx* 문자열을 지정해야 된다. 원격 출력에 대한 더 많은 정보는 원격 호스트에 설치된 프린터 섹션을 참고한다.



---

원격 프린터에서 작업 크기를 제한하는 또 다른 방법이 있다; 원격 프린터에서 잡 제한하기 섹션을 본다.

#### 9.4.4.4 원격 프린터에서 잡 제한

LPD 스펙링 시스템은 원격 호스트로부터 잡을 제한하는 몇 가지 방법을 제공한다:

##### 호스트 제한

어떤 원격 호스트가 로컬 LPD 에 잡을 요청할 수 있는지 `/etc/hosts.equiv` 와 `/etc/hosts.lpd` 파일로 제어할 수 있다. LPD 는 이들 파일에 나열된 호스트의 요청인지 체크해서 그렇지 않다면 LPD 는 요청을 무시 한다.

이 파일들의 포맷은 간단하다: 라인당 하나의 호스트 이름을 적는다. 파일 `/etc/hosts.equiv` 는 `ruserok(3)` 프로토콜에 사용되어 `rsh(1)`와 `rcp(1)` 같은 프로그램에 영향을 주기 때문에 주의한다.

예를 들어 호스트 `rose` 의 `/etc/hosts.lpd` 파일이 있다:

```
orchid
violet
madrigal.fishbaum.de
```

이 의미는 `rose` 가 호스트 `orchid` 와 `violet` 그리고 `madrigal.fishbaum.de` 의 요청을 허용한다. 다른 호스트가 `rose` 의 LPD 에 접근 하려면 거부당한다.

##### 크기 제한

스플링 디렉터리가 있는 파일시스템에 여유 공간이 얼마나 많이 필요한지 결정할 수 있다. 로컬 프린터의 스펙링 디렉터리에 `minfree` 라는 파일을 생성한다. 여유 디스크 블록을 얼마나 원격 작업에 할당할지 이 파일에 숫자로 지정한다.

이것은 권한이 없는 원격 유저가 파일시스템을 가득 채우지 못하게 한다. 또한 로컬

---

유저에게 잡의 우선권을 줄 수도 있다: 우선권은 minfree 파일에 지정한 양 이상으로 여유 디스크 공간을 할당해서 원격 잡이 큐에서 오래 대기하도록 한다.

예를들면 프린터 *bamboo*에 minfree 파일을 추가한다. 이 프린터의 스펙링 디렉터리를 `/etc/printcap`에서 찾도록 한다; 여기 *bamboo*의 엔트리가 있다:

```
bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:W
      :sh:sd=/var/spool/lpd/bamboo:sc:rg=artists:mx#5000:W
      :lp=/dev/ttyd5:ms#-parenb cs8 clocal crtscts:rw:mx#5000:W
      :if=/usr/local/libexec/psif:W
      :df=/usr/local/libexec/psdf:
```

스펙링 디렉터리는 *sd* 문자열로 지정되어있다. 우리는 LPD가 원격 작업에 사용하도록 파일시스템의 여유 디스크 공간을 3 MB로(6144 디스크블록) 만들었다:

```
# echo 6144 > /var/spool/lpd/bamboo/minfree
```

## 유저 제한

`/etc/printcap`에 *rs* 문자열로 어떤 원격 유저가 로컬 프린터를 사용할 수 있는지 제어할 수 있다. *rs*가 로컬 프린터 엔트리에 있을때 유저가 로컬 호스트와 같은 계정 이름으로 잡을 보냈다면 LPD는 원격 호스트의 잡을 허용한다. 그렇지 않으면 LPD는 잡을 거부한다.

이런 기능은 네트워크로 공유하는 다른 부서 환경에서 특히 유용하고 어떤 유저는 부서간 경계를 넘나들 수 있다. 이들에게 시스템 계정을 줘서 자신이 속한 부서의 프린터를 사용할 수 있도록 한다. 여러분의 프린터만 사용하고 컴퓨터 자원은 사용하지 못하게 하려면 *hlp* 디렉터리가 없고 `/usr/bin/false`처럼 사용할 수 없는 쉘을 할당하는 “token” 계정을 줄수 있다.

## 9.4.5 프린터 사용량 계산

종이와 잉크 값 그리고 프린터 유지보수 때문에 출력에 대한 요금을 부과해야 된다. 사용패턴과 유지보수 비용을 페이지당 가격으로 산출한다. 이제 실제로 출력물에 대해 어떻게 계

---

산 하는지 알아보자

불행히 LPD 스플링 시스템은 이 부분에 많은 도움이 되지 못한다. 요금 계산은 사용하는 프린터의 종류와 출력되는 포맷 그리고 프린터 사용량에 따라 다르다.

요금을 계산하려면 페이지를 계산하거나 출력된 페이지를 산출하도록 프린터의 텍스트 필터와(평범한 텍스트 잡 요금) 변환필터를(다른 파일포맷 요금) 수정해야 된다. 간단한 출력필터는 계산을 못하기 때문에 사용하지 못한다. 필터섹션을 본다.

일반적으로 두 가지 계산방법이 있다:

- *주기적인 계산*은 쉽기 때문에 일반적인 방법이다. 잡을 출력할 때마다 필터는 유저, 호스트 그리고 파일페이지를 로그로 남긴다. 매월, 반 년간, 연간 또는 원하는 기간 동안 유저가 출력한 페이지의 총계를 모아서 요금을 부과한다. 그리고 모든 로그파일을 삭제하고 다음 기간을 시작한다.
- *시기적인 계산*은 좀더 어렵기 때문에 많이 사용되지 않는다. 이 방법은 유저가 프린터를 사용하자마자 요금을 부과하는 필터를 가지고 있다. 디스크 쿼타처럼 요금 계산은 순간적으로 처리된다. 유저계정이 사용량을 초과했을때 프린터를 사용하지 못하게 해서 유저가 자신의 "프린터 사용 쿼타"를 체크하고 조정하도록 할수 있다. 그러나 이 방법은 유저와 유저의 쿼타를 추적하는 데이터베이스 코드가 필요하다.

LPD 스플링 시스템은 두 가지 방법을 쉽게 지원한다: 그리고 필터를(대부분) 제공해야 되기 때문에 계산 코드도 제공해야 된다. 그러나 긍정적인 부분이 있다: 계산 방법에 엄청난 유연성을 가지고 있다. 예를 들면 주기적 또는 시기적인 계산을 선택한다. 로그로 남길 정보를 선택한다: 유저 이름, 호스트 이름, 잡 종류, 출력된 페이지, 사용된 종이의 종류와 프린터를 사용한 시간 등을 선택한다. 그리고 이런 정보를 저장하도록 필터를 수정하면 된다.

#### 9.4.5.1 대략적으로 프린터 요금 빠르게 계산하기

FreeBSD 는 주기적인 계산에 간단히 설정할 수 있는 두가지 프로그램이 있다. 이들은 "lpf: 텍스트 필터" 섹션에서 설명한 텍스트필터와 프린터 계산파일에서 전체 엔트리를 모으는 pac(8) 프로그램이다.

---

필터섹션에서 언급했듯이 **LPD**는 필터명령어 라인에 계산파일의 이름을 사용하도록 텍스트와 변환필터를 실행한다. 필터는 계산파일 엔트리를 어느 곳에 작성할지 확인하기 위해 이 인자를 사용할 수 있다. `/etc/printcap` 파일의 `af` 문자열에 있는 이 파일 이름을 절대 경로로 지정하지 않았다면 스펠링 디렉터리와 상대경로가 된다.

**LPD**는 페이지 넓이와 길이 인자로(`pw`와 `p/` 문자열에서) `lpf`를 시작한다. `lpf`는 필요한 종이를 결정하는데 이들 인자를 사용한다. 프린터에 파일을 보낸 후 계산 파일에 계산 엔트리를 작성한다. 이 엔트리는 다음과 비슷하다:

```
2.00 rose:andy
3.00 rose:kelly
3.00 orchid:mary
5.00 orchid:mary
2.00 orchid:zhang
```

`lpf`는 파일 잠금 로직을 생성하지 않기 때문에 각 프린터의 계산파일을 사용하여 동시에 같은 파일을 작성한다면 두 `lpf`는 각자의 엔트리를 침범할 것이다.

각 프린터의 계산파일을 보장하는 쉬운 방법은 `/etc/printcap`에 `af=acct`를 사용한다. 그리고 각 계산 파일은 프린터 스펠링 디렉터리의 `acct`라는 파일이름이 된다.

유저에게 프린터 요금을 부과할 준비가 되었을 때 `pac(8)` 프로그램을 실행한다. 수집을 원하는 프린터의 스펠링 디렉터리로 이동해서 `pac`를 입력한다. 다음과 비슷한 달러 문자로 나뉜 요약본을 볼수 있다:

Login	pages/feet	runs	price
orchid:kelly	5.00	1	\$ 0.10
orchid:mary	31.00	3	\$ 0.62
orchid:zhang	9.00	1	\$ 0.18
rose:andy	2.00	1	\$ 0.04
rose:kelly	177.00	104	\$ 3.54
rose:mary	87.00	32	\$ 1.74
rose:root	26.00	12	\$ 0.52
total	337.00	154	\$ 6.74

여기 `pac(8)`에 사용하는 인자들이 있다:

---

-*Pprinter*

*printer*은 요약할 프린터다. 이 옵션은 */etc/printcap*의 *af* 문자열에 절대 경로가 있다면 작동한다.

-*c*

유저 이름의 알파벳 순서대신 요금 별로 정렬한다.

-*m*

계산파일에서 호스트 이름을 무시한다. 이 옵션으로 호스트 *alpha*의 유저 *smith*는 호스트 *gamma*의 유저 *smith*와 같다. 이 옵션을 사용하지 않으면 이들은 다른 유저가 된다.

-*pprice*

*/etc/printcap*의 *pc* 문자열 요금대신 페이지당 또는 피트당 *price* 달러로 요금을 계산하거나 2 센트로(기본값) 요금을 부과한다. 부동 소수로 *price*를 지정할 수 있다.

-*r*

거꾸로 정렬한다

-*s*

계산 요약파일을 만들고 계산파일을 결산한다.

---

*name...*

주어진 유저 *names* 의 계산정보만 출력한다.

pac(8)가 산출한 기본적인 요약에서 다양한 호스트의 각 유저들이 출력한 페이지의 양을 볼수 있다. 여러분의 사이트에서 호스트에 의미가 없다면(왜냐하면 유저가 어떤 호스트라도 사용할 수 있기 때문에) 다음과 같이 요약하기 위해 `pac -m` 을 실행한다:

Login	pages/foot	runs	price
andy	2.00	1	\$ 0.04
kelly	182.00	105	\$ 3.64
mary	118.00	35	\$ 2.36
root	26.00	12	\$ 0.52
zhang	9.00	1	\$ 0.18
total	337.00	154	\$ 6.74

총액을 계산하기 위해 pac(8)는 `/etc/printcap` 파일의(기본적으로 종이당 200 또는 2 센트) `pc` 문자열을 사용한다. 특히 종이당 또는 피트당 가격을 백 센트 단위로 부과할 때 이 문자열이 사용된다. `-p` 옵션으로 pac(8)를 실행할 때 이 값을 무시할 수 있다. `-p` 옵션의 단위는 백센트가 아닌 달러다. 예를 들어 다음 명령은 종이당 가격을 1 달러 50 센트로 적용한다.

#### # **pac -pl.50**

이 옵션으로 실재로 많은 이익을 낼 수 있다.

마지막으로 `pac -s` 를 실행하면 프린터의 계정파일과 같은 이름으로 요약정보를 생성하지만 이름에 `_sum` 이 추가된다. 그리고 계산파일을 종료한다. `pac(8)`을 다시 실행하면 총액을 계산하기 위해 요약파일을 다시 읽고 일반 계산파일에 정보를 추가한다.

### 9.4.5.2 출력된 페이지는 어떻게 계산하는가?

원격지에 있더라도 정확하게 계산하기 위해 잡에 사용된 종이 양을 결정할 수 있어야 한다.

---

이것이 프린터 계산의 근본적인 문제이다.

평범한 텍스트 잡에서는 이 문제를 해결하기가 어렵지 않다: 잡에있는 라인과 프린터가 지원하는 용지당 라인을 비교해서 계산한다. 라인을 გადა 출력한 파일에서 백스페이스 계산이나 한개 또는 한개 이상의 물리적인 라인으로 둘러싸인 긴 논리적인 라인 계산을 잊지 않는다.

텍스트필터 `lpf` 는(`lpf`: 텍스트 필터에서 소개한) 계산할 때 이러한 것을 계산한다. 계산에 필요한 텍스트필터를 작성한다면 `lpf` 의 소스코드를 확인하는 것도 좋을 것이다.

그렇지만 다른 파일포맷은 어떻게 조정할 것인가?

DVI 를 레이저젯 또는 DVI 를 포스트스크립트로 변환하는 것은 `dvilj` 나 `dvips` 의 진단결과를 분석하여 변환된 페이지를 확인할 수 있다. 다른 파일포맷과 변환프로그램도 이와 비슷하게 계산할 수 있을 것이다.

그러나 사실 이런 방법으로 실제 프린터가 이들 페이지를 모두 출력하지 않은 것을 알게 될 것이다. 예를들어 종이 걸림이나 토너부족 또는 파열된 것을 출력하고 유저는 계속 요금을 부과 당한다.

그래서 어떻게 할수 있는가?

정확히 계산할 수 있는 한 가지 방법이 있다. 사용한 용지 양을 알려주는 시리얼 라인이나 네트워크에 연결되어 있는 프린터를 사용한다. 거의 모든 포스트스크립트 프린터는 이런 기능을 가지고 있다. 다른 모델들도 비슷한 기능을 가지고 있다. 이런 프린터가 잡을 처리한 후 페이지 사용량을 확인하고 계산할 수 있는 정보를 로그로 남기도록 필터를 수정한다. 라인 계산에 애러가 없다면 파일을 확인할 필요가 없다.

물론 관대하게 모든 출력물을 무료로 할수 있다.

## 9.5 프린터 사용

이 섹션은 FreeBSD 에 설치한 프린터 사용법을 알려준다. 여기서는 유저레벨 명령을 소개한다:

---

lpr(1)

잡을 출력한다.

lpq(1)

프린터 큐를 체크한다.

lprm(1)

프린터 큐에서 잡을 삭제한다.

LPD 스플러 관리하기 섹션에서 설명한 프린터와 프린터 큐 제어에 사용하는 관리자 명령 lpc(8)도 있다.

lpr(1), lprm(1) 그리고 lpq(1) 3 가지 명령은 /etc/printcap 파일에 나열 되어있듯이 사용할 프린터/큐를 지정하는 옵션 *-P printer-name* 을 사용한다. 이것은 다양한 프린터에 잡을 보내고 삭제 및 체크할 수 있다. *-P* 옵션을 사용하지 않는다면 이 명령은 PRINTER 환경변수에 지정된 프린터를 사용한다. 마지막으로 PRINTER 환경변수도 없다면 이 명령은 기본 프린터 *lp* 를 사용한다.

앞으로 *기본 프린터*의 의미는 PRINTER 환경변수의 프린터 이름이나 PRINTER 환경변수가 없을때 프린터 이름 *lp* 를 의미한다.

## 9.5.1 출력

파일을 출력하려면 다음 명령을 입력한다:

```
% lpr filename ...
```

이것은 나열한 파일을 기본 프린터로 출력한다. 파일을 나열하지 않았다면 lpr(1)은 기본입력에서 데이터를 읽어온다. 예를 들어 이 명령은 몇가지 중요한 시스템파일을 출력한다:



---

% `lpr /etc/host.conf /etc/hosts.equiv`

특정 프린터를 지정하려면 다음 명령을 입력한다:

% `lpr -P printer-name filename ...`

이 예제는 프린터이름 *rattan*으로 현재 디렉터리의 자세한 리스트를 출력한다:

% `ls -l | lpr -P rattan`

왜냐하면 `lpr(1)` 명령에 파일을 나열하지 않았기 때문에 `lpr`은 `ls -l` 명령으로의 출력을 기본 입력으로 받아서 읽는다.

`lpr(1)` 명령은 포맷제어, 파일변환 적용, 여러 장씩 출력 등을 제어하는 다양한 옵션을 사용할 수 있다. 더 많은 정보는 출력옵션 섹션을 본다.

## 9.5.2 잡 체크

`lpr(1)`로 출력할 때, 출력하려는 데이터를 **LPD** 스폰링 시스템에 보내기 위해 "출력 잡"이라는 패키지로 묶는다. 각 프린터는 잡 큐를 가지고 있어서 여러분과 다른 유저의 잡은 이곳에서 대기한다. 프린터는 첫 번째로 입력된 잡을 첫 번째로 하여 순서대로 출력한다.

기본 프린터의 큐를 체크하려면 `lpq(1)`을 입력한다. 특정 프린터는 `-P` 옵션을 이용한다. 예를 들어 다음 명령은 프린터이름 *bamboo*의 큐를 보여준다.

% `lpq -P bamboo`

다음은 `lpq` 명령의 결과를 보여준다:

bamboo is ready and printing					
Rank	Owner	Job	Files	Total Size	
active	kelly	9	/etc/host.conf, /etc/hosts.equiv	88 bytes	
2nd	kelly	10	(standard input)	1635 bytes	
3rd	mary	11	...	78519 bytes	

---

위 명령은 *bamboo*의 큐에있는 3개의 잡을 보여준다. 첫 번째 잡은 유저 *kelly*가 실행했고 잡 번호 9를 가지고 있다. 프린터의 모든 잡은 유일한 잡 번호를 가진다. 보통은 잡 번호를 사용할 필요가 없지만 잡을 취소하기 위해서 필요하다. 더 자세한 사항은 잡 삭제하기 섹션을 본다.

잡 번호 9는 파일이 두 개로 되어있다; 여러 파일을 *lpr(1)* 명령어 라인에 주면 하나의 잡으로 간주한다. 현재 활성화된 잡이라는(단어 *active* 아래의 "Rank" 칼럼을 확인한다) 의미는 프린터가 이 잡을 현재 출력 중이라는 것이다. 두 번째 잡은 *lpr(1)* 명령의 표준입력에 데이터를 입력하고 있다. 세 번째 잡은 유저 *mary*가 보냈다; 이것은 좀더 큰 잡이다. 그녀가 출력하려는 파일의 경로이름이 너무 길기 때문에 *lpq(1)* 명령은 간단히 3개의 점(.)으로 보여준다.

*lpq(1)* 출력의 첫 번째 라인도 유용하다: 프린터가 현재 무엇을 하는지 알려준다(또는 최소한 프린터가 무엇을 하는지 **LPD**가 알고 있다)

또한 *lpq(1)* 명령은 자세하고 긴 리스트를 생성하기 위해 *-l* 옵션을 지원한다. 여기 *lpq -l*의 예제가 있다:

```
waiting for bamboo to become ready (offline ?)
kelly: 1st                [job 009rose]
    /etc/host.conf                73 bytes
    /etc/hosts.equiv              15 bytes

kelly: 2nd                [job 010rose]
    (standard input)              1635 bytes

mary: 3rd                 [job 011rose]
    /home/orchid/mary/research/venus/alpha-regio/mapping 78519 bytes
```

### 9.5.3. 잡 삭제

출력하려는 마음이 바뀌었다면 *lprm(1)* 명령으로 큐에서 잡을 삭제할 수 있다. 종종 활성화된 잡이라도 *lprm(1)*로 삭제할 수 있지만 약간 또는 대부분의 잡이 그냥 출력된다.

---

기본 프린터에서 잡을 삭제하려면 `lpq(1)`로 잡 번호를 찾는다. 그리고 다음과 같이 입력한다:

```
% lprm job-number
```

특정 프린터에서 잡을 삭제하려면 `-P` 옵션을 추가한다. 다음 명령은 프린터 *bamboo*의 큐에서 잡 번호 10을 삭제한다:

```
% lprm -P bamboo 10
```

`lprm(1)` 명령은 몇 가지 쉬운 명령을 가지고 있다:

```
lprm -
```

여러분이 실행한 모든 잡을(기본 프린터에서) 삭제한다.

```
lprm user
```

*user*가 실행한 모든 잡을(기본 프린터에서) 삭제한다. 슈퍼 유저는 다른 유저의 잡을 삭제할 수 있다; 여러분은 자신의 잡만 삭제할 수 있다.

```
lprm
```

명령어 라인에 잡 번호나 유저 이름 또는 `-`을 사용하지 않으면 `lprm(1)`은 여러분이 실행하고 현재 기본프린터에서 활성화된 잡을 삭제한다. 슈퍼 유저는 활성화된 어떤 잡이라도 삭제할 수 있다.

위의 쉬운 명령에 `-P` 옵션을 사용하여 기본 프린터 대신 특정프린터를 사용한다. 예를 들어 다음 명령은 프린터 이름 *rattan*의 큐에서 현재 유저의 모든 잡을 삭제한다:

```
% lprm -P rattan -
```

---

**Note:** 네트워크 환경에서 작업 중이면 `lprm(1)`은 다른 호스트에서 같은 프린터를 사용할 수 있지만 잡을 실행한 호스트에서만 잡을 삭제하도록 한다. 다음 명령의 순서에서 이 사항을 보여준다:

```
% lpr -P rattan myfile
% rlogin orchid
% lpq -P rattan
Rank   Owner      Job  Files          Total Size
active seeyan   12   ...          49123 bytes
2nd    kelly      13   myfile         12 bytes
% lprm -P rattan 13
rose: Permission denied
% logout
% lprm -P rattan 13
dfA013rose dequeued
cfA013rose dequeued
```

## 9.5.4 다양한 출력옵션

`lpr(1)` 명령은 텍스트포맷, 그래픽변환과 다른 파일포맷, 여러 장씩 출력 그리고 잡 제어 등을 위한 옵션을 지원한다. 이번 섹션은 이 옵션을 설명한다.

### 9.5.4.1 포맷과 변환옵션

다음 `lpr(1)` 옵션은 잡의 파일포맷을 제어한다. 잡이 평범한 텍스트를 가지고 있지 않거나 `pr(1)` 유틸리티로 평범한 텍스트포맷을 원하면 이런 옵션을 사용한다.

예를 들어 다음 명령은 DVI 파일이름(TeX typesetting 시스템에서) `fish-report.dvi` 를 프린터 *bamboo* 로 출력한다:

```
% lpr -P bamboo -d fish-report.dvi
```

---

이들 옵션이 잡의 모든 파일에 적용되기 때문에 잡에서 DVI와 ditroff 파일을 같이 섞지 못한다. 대신 다른 잡처럼 각 잡에 다른 변환옵션을 사용한다.

**Note:** 이들 옵션 중 *-p*와 *-T* 옵션을 제외한 모든 옵션은 대상 프린터에 변환필터가 필요하다. 예를 들어 *-d* 옵션은 DVI 변환필터가 필요하다. 변환필터 섹션에서 더욱 자세히 설명한다.

*-c*

cifplot 파일을 출력한다.

*-d*

DVI 파일을 출력한다.

*-f*

FORTTRAN 텍스트파일을 출력한다.

*-g*

plot 데이터를 출력한다.

*-i number*

*number* 칼럼만큼 출력이 들여쓰기 된다; *number*를 생략하면 들여쓰기는 8이 기본값이다. 이 옵션은 오직 특정 변환필터와 동작한다.

**Note:** *-i*와 숫자 사이에 공간을 두지 않는다

*-l*

---

---

특성을 포함하여 텍스트 데이터를 그대로 출력한다.

`-n`

ditroff(장치에 독립적인 troff) 데이터를 출력한다.

`-p`

출력하기 전에 pr(1)을 사용하여 평범한 텍스트로 포맷한다. 더 많은 정보는 pr(1)을 본다.

`-T title`

파일 이름 대신 pr(1) 헤더에 *title* 를 사용한다. 이 옵션은 `-p` 옵션과 사용할 때만 효과가 있다.

`-t`

troff 데이터를 출력한다.

`-v`

raster 데이터를 출력한다.

여기 예제가 있다: 다음 명령은 보기 좋게 포맷된 버전의 ls(1) 매뉴얼 페이지를 기본 프린터로 출력한다:

```
% zcat /usr/share/man/man1/ls.1.gz | troff -t -man | lpr -t
```

---

zcat(1) 명령은 ls(1) 매뉴얼 페이지 소스의 압축을 해제하고 그 소스를 포맷하여 GNU troff 출력물로 만들어서 lpr 에(잡 을 LPD 스플러로 보내는 역할을 하는) 보내는 역할을 하는 troff(1) 명령으로 보낸다. 왜냐하면 *-t* 옵션을 lpr(1)에 사용했기 때문에 스플러는 GNU troff 출력물 포맷을 기본 프린터가 이해할 수 있는 포맷으로 변환한다.

### 9.5.4.2 잡 제어 옵션

다음 옵션은 lpr(1)에게 LPD 가 잡을 특수하게 제어하도록 한다:

*-# copies*

한 장씩 출력하지 않고 잡의 각 파일을 여러 장씩 출력한다. 관리자는 프린터 마모를 줄이고 복사기 사용량을 증가 시키려고 이 옵션을 비활성 할 것이다. 여러 장씩 출력제한 섹션을 본다.

이 예제는 parser.c 와 parser.h 를 기본 프린터에서 3 장씩 출력한다:

```
% lpr -#3 parser.c parser.h
```

*-m*

잡 출력이 완전히 끝나면 메일을 보낸다. 이 옵션으로 LPD 시스템은 잡 제어가 끝났을때 계정으로 메일을 보낸다. 이 메시지로 잡이 성공적으로 끝났는지 또는 에러가 발생했고 에러는 무엇이라고 알려준다.

*-s*

파일을 스플링 디렉터리에 복사하지 않고 심볼릭 링크를 생성한다.

커다란 잡을 출력한다면 이 옵션을 사용하고 싶을 것이다. 스플링 디렉터리(스플링 디렉터리가 있는 파일시스템의 여유공간을 잡이 전부 소모할 것이다) 공간을 절약하게 한다. LPD 가 모든 잡을 스플링 디렉터리에 복사하지 않기 때문에 시간도 절약한다.

---

반면 단점도 있다: **LPD**가 원본파일을 직접 참조하기 때문에 출력하기 전에 원본파일을 삭제하거나 수정할 수 없다.

**Note:** 원격 프린터로 출력한다면 결국 **LPD**는 로컬 호스트에서 원격 호스트로 파일을 복사해야 되기 때문에 `-s` 옵션은 원격이 아닌 로컬 스펠링 디렉터리 공간을 절약하게 되지만 그래도 유용하다.

`-r`

잡을 스펠링 디렉터리에 복사하거나 `-s` 옵션으로 출력 후 파일을 삭제한다. 이 옵션은 주의해야 된다!

### 9.5.4.3 헤더페이지 옵션

이들 옵션은 `lpr(1)`에게 보통 잡의 헤더페이지에 나타나는 텍스트를 조정하도록 한다. 목적 프린터에서 헤더페이지를 사용하지 않는다면 이 옵션은 아무런 효과가 없다. 헤더페이지 설정에 대한 정보는 헤더페이지 섹션을 본다.

`-C text`

`text`로 헤더페이지의 호스트 이름을 바꾼다. 호스트 이름은 보통 잡을 보낸 호스트 이름이다.

`-J text`

`text`로 헤더페이지의 잡 이름을 바꾼다. 잡 이름은 보통 잡의 첫 번째 파일명이거나 표준입력으로 출력한다면 `stdin`이다.

`-h`

헤더페이지를 출력하지 않는다.



---

**Note:** 어떤 사이트에서는 이 옵션이 헤더페이지 생성과 연관이 없다. 더 자세한 사항은 헤더페이지를 본다.

## 9.5.5 프린터 관리

프린터 관리자로서 프린터를 설치하고 테스트해야 된다. `lpc(8)` 명령으로 프린터를 더욱 세세히 운용할 수 있다. `lpc(8)`을 사용하여 다음과 같은 사항을 제어할 수 있다:

- 프린터 시작/정지
- 프린터 큐 켜기/끄기
- 각 큐에서 잡 순서를 다시 정렬한다.

첫째 용어에 관해 정리한다: 프린터가 정지됐다면 큐에있는 어떤 잡도 출력하지 않는다. 유저는 프린터가 시작되거나 큐가 정리될 때까지 대기하는 잡을 계속 보낼 수 있다.

큐를 꺼 두었다면 유저는(`root`를 제외한) 프린터로 잡을 보낼 수 없다. 큐를 켜면 잡을 보낼 수 있다. 큐를 끄고 프린터를 시작할 수 있고 이 경우 큐가 비어질 때까지 잡을 출력한다.

보통 `lpc(8)` 명령을 사용하려면 `root` 권한이 필요하다. 일반 유저는 프린터상태를 확인해서 반응이 없는 프린터를 다시 시작할 때만 `lpc(8)` 명령을 사용할 수 있다.

여기 `lpc(8)` 명령을 요약했다. 대부분의 명령은 운용할 프린터를 지시하기 위해 *printer-name* 인자를 갖는다. *printer-name*에 *all*을 사용하여 `/etc/printcap`에 나열된 모든 프린터를 사용할 수 있다.

`abort printer-name`

현재 잡을 취소하고 프린터를 정지시킨다. 유저는 큐가 활성화되어 있으면 계속 잡을 보낼 수 있다.

---

`clean printer-name`

프린터 스펠링 디렉터리에서 오래된 파일을 삭제한다. 가끔 잡을 생성하는 파일은 특히 출력하는 동안 발생한 예러나 관리목적의 작업이 많을 때 **LPD** 로 완전히 삭제되지 않는다. 이 명령은 스펠링 디렉터리에 속하지 않는 파일을 찾아서 삭제한다.

`disable printer-name`

새로운 잡 큐를 비활성 한다. 프린터가 작동 중이라면 큐에 남아있는 모든 잡을 출력한다. 슈퍼 유저는(root) 큐를 비활성 했더라도 항상 잡을 보낼 수 있다.

이 명령은 새로운 프린터나 필터설치를 테스트하는 동안 유용하다: root 로 큐를 비활성하고 잡을 보낸다. 다른 유저는 슈퍼 유저가 테스트를 끝나치고 큐를 enable 명령으로 다시 활성화시킬 때까지 잡을 보내지 못한다.

`down printer-name message`

프린터를 다운시킨다. 프린터를 disable 하고 stop 한 것과 같다. *message* 는 프린터 큐를 lpq(1)나 lpc status 로 상태를 체크할 때마다 나타난다.

`enable printer-name`

프린터 큐를 활성화한다. 유저는 잡을 보낼 수 있지만 프린터는 큐가 시작될 때까지 출력하지 못한다.

`help command-name`

*command-name* 명령의 도움말을 보여준다. *command-name* 이 없으면 사용할 수 있는 명령어를 요약해준다.

---

`restart printer-name`

프린터를 시작한다. **LPD**가 이상한 상태로 멈춰있다면 일반 사용자가 이 명령을 사용할 수 있지만 유저는 `stop`이나 `down` 명령으로 정지시킨 프린터는 시작할 수 없다. `restart` 명령은 `abort` 후 `start`와 동일하다.

`start printer-name`

프린터 시작. 프린터는 큐의 잡을 출력한다.

`stop printer-name`

프린터 정지. 프린터는 현재 잡을 마치고 더 이상 출력하지 않는다. 프린터가 정지되었더라도 유저는 활성화된 큐에 잡을 보낼 수 있다.

`topq printer-name job-or-username`

`printer-name`의 큐를 `job` 번호나 잡이 속한 `username`으로 나열하여 순서를 바꾼다. 이 명령의 `printer-name`에는 `all`을 사용할 수 없다.

`up printer-name`

프린터를 올린다; `down` 명령과 반대다. `start` 후 `enable`과 같은 명령이다.

`lpc(8)` 명령어 라인으로 위 명령을 사용할 수 있다. 어떤 명령도 입력하지 않으면 `lpc(8)`은 `exit`, `quit`로 파일의 끝을 입력할 때까지 대화식 모드로 들어간다.

## 9.6 표준 스플러 대체

---

이 매뉴얼을 순차적으로 읽어왔다면 이제 FreeBSD의 LPD 스펴링 시스템에 대한 모든 것을 배웠다. 따라서 다음과 같은 질문을 던지게 될 것이다: "다른 스펴링 시스템은 어떤 것이 있는가(그리고 FreeBSD에서 작동하는가)?"

## LPRng

"LPR: 다음 세대"를 의미하는 **LPRng**는 완벽하게 PLP를 다시 작성한 것이다. Patrick Piwell과 Justin Mason이(PLP의 주요 관리자) **LPRng**를 공동으로 만들었다. **LPRng**의 메인 사이트는 <http://www.lprng.org>다.

## CUPS

Common UNIX Printing System **CUPS**는 유닉스 기반 운영체제에 이식 가능한 프린팅 레이어를 제공한다. 모든 유닉스 벤더와 유저에게 표준 프린팅 솔루션을 장려하기 위해 Easy Software Products가 개발했다.

**CUPS**는 프린트 잡과 큐를 관리하기 위해 기본적으로 Internet Printing Protocol (IPP)를 사용한다. Line Printer Daemon(LPD) Server Message Block (SMB) 그리고 AppSocket (a.k.a JetDirect) 프로토콜도 약간의 기능 제한으로 지원된다. **CUPS**는 유닉스에서 실질적인 프린팅을 지원하기 위해 네트워크 프린터 브라우저와 PostScript Printer Description (PPD) 기반 프린팅 옵션을 추가했다.

**CUPS**의 메인 사이트는 <http://www.cups.org/>

## 9.7 문제 해결

lpstest(1)로 간단히 테스트했을 때 정확히 출력되지 않고 아래와 같은 결과를 보게 될 수 있다.

**잠시 동안 동작하지만 용지전체를 배출하지는 않는다.**

프린터가 내용은 출력하지만 잠시 후 작동하지 않는다. 이 말은 프린터의 PRINT REMAINING이나 FORM FEED 버튼을 눌러야 됴을 의미한다.

---

이 경우 프린터는 다른 것을 출력하기 전에 더 많은 데이터가 있는지 확인하려고 대기 중 이었다. 이 문제를 해결하려면 텍스트필터로 FORM FEED 문자를(또는 필요한 경우) 프린터에 보내야 한다. 이로서 프린터는 내부 버퍼에 남아있는 텍스트를 즉시 출력한다. 또한 각 잡을 완전히 끝내기 때문에 다음 잡은 이전 잡의 마지막 페이지 중간부분이 끝나기 전까지 시작되지 않는다.

/usr/local/libexec/if-simple 을 대체하는 다음 셸 스크립트는 잡을 프린터에 보낸 후 폼 피드를 출력한다:

```
#!/bin/sh
#
# if-simple - Simple text input filter for lpd
# Installed in /usr/local/libexec/if-simple
#
# Simply copies stdin to stdout. Ignores all filter arguments.
# Writes a form feed character (Wf) after printing job.

/bin/cat && printf "Wf" && exit 0
exit 2
```

### "계단 효과" 생성

용지에서 다음과 같은 결과를 볼 수 있다:

```
!#$%&'()*+,-./01234
      "$%&'()*+,-./012345
            #%&'()*+,-./0123456
```

특정 문자가 새로운 라인으로 잘못 해석되기 때문에 여러분은 계단효과의 또 다른 희생자가 되었다. 유닉스 스타일의 운영체제는 새로운 라인 표시에 하나의 문자만 사용한다: ASCII 코드 10, 라인 피드(LF). 그러나 MS-DOS, OS/2 그리고 다른 운영체제가 사용하는 문자 쌍은 ASCII 코드 10 과 ASCII 코드 13 이다(carriage return 또는 CR). 대부분의 프린터는 새로운 라인을 표현할 때 MS-DOS 방식을 따른다.

---

FreeBSD 에서 출력할 때 이런 텍스트는 라인 피드 문자로 사용된다. 프린터는 라인 피드 문자 설정에 따라 종이를 한 라인씩 전진시키지만 다음 문자를 출력하기 위해 용지에 같은 수평위치를 유지한다. 캐리지 리턴의 기능은 출력하려는 다음 문자의 위치를 종이의 왼쪽 가장자리로 옮기는 것이다.

다음은 FreeBSD 가 원하는 프린터의 동작이다:

Printer received CR	Printer prints CR
Printer received LF	Printer prints CR + LF

이렇게 설정을 변경하는 몇 가지 방법이 있다:

- 프린터의 설정 스위치나 조정 패널로 이런 문자의 해석을 변경한다. 설정을 변경하기 위해 프린터 매뉴얼을 확인한다.

**Note:** FreeBSD 와 다른 운영체제를 멀티 부팅한다면 다른 운영체제가 CR 과 LF 문자를 해석할 수 있도록 프린터를 다시 설정해야 된다. 아래의 다른 해결책을 찾을 수 있다.

- 시리얼 라인에서만 FreeBSD 의 시리얼 라인 드라이버가 자동으로 LF 를 CR+LF 로 변환한다. 이 기능을 사용하려면 /etc/printcap 파일에서 *ms#* 문자열을 사용하고 *onlcr* 모드를 설정한다.
- *escape code* 를 보내서 일시적으로 프린터가 LF 문자를 다르게 처리하도록 한다. 프린터가 지원하는 *escape* 코드는 프린터 매뉴얼을 참고한다. 적당한 *escape* 코드를 발견하면 최초 코드를 보내도록 텍스트필터를 수정하고 출력할 잡을 보낸다.

여기 휴렛팩커드 PCL *escape* 코드를 이해하는 프린터의 텍스트필터 예제가 있다. 이 필터는 프린터가 LF 문자를 LF 와 CR 로 인식하게 한다; 그리고 잡을 보낸다; 마지막으로 잡의 마지막 페이지를 배출하도록 폼 피드를 보낸다. 이 예제는 거의 모든 휴렛팩커드 프린터에서 동작한다.

```
#!/bin/sh
#
```

---

```

# hpif - Simple text input filter for lpd for HP-PCL based printers
# Installed in /usr/local/libexec/hpif
#
# Simply copies stdin to stdout. Ignores all filter arguments.
# Tells printer to treat LF as CR+LF. Ejects the page when done.

printf "\033&k2G" && cat && printf "\033&l0H" && exit 0
exit 2

```

여기 orchid 라는 호스트의 /etc/printcap 예제가 있다. 이 호스트에는 teak 이라는 이름의 휴렛팩커드 레이저젯 3Si 가 첫 번째 패러럴 포트에 연결되어 있다. 위 스크립트를 텍스트필터처럼 이용한다:

```

#
# /etc/printcap for host orchid
#
teak|hp|laserjet|Hewlett Packard LaserJet 3Si:\W
        :lp=/dev/lpt0:sh:sd=/var/spool/lpd/teak:mx#0:\W
        :if=/usr/local/libexec/hpif:

```

각 라인을 과다 출력한다.

프린터는 절대 라인을 전진시키지 않는다. 모든 텍스트가 가장 상단의 한 라인에 계속 출력된다.

이 문제는 위에서 설명한 계단효과와 반대로 더욱 드문 경우다. FreeBSD 가 라인의 끝을 표시하기 위해 사용하는 LF 문자가 용지의 좌측처음으로 되돌아오게 하는 CR 문자처럼 사용되고 다음 라인으로도 변경되지 않는다.

프린터의 설정 스위치나 조정 패널로 LF 와 CR 문자를 다음과 같이 해석하도록 변경한다:

Printer receives	Printer prints
------------------	----------------

---

CR

CR

---

LF

CR + LF

### 프린터가 문자를 빠뜨린다.

출력하는 동안 프린터가 각 라인 별로 문자를 몇개씩 출력하지 않는다. 이 문제는 프린터를 사용할수록 더 많은 문자를 빠뜨리는 결과를 낳는다.

이 문제는 컴퓨터에서 시리얼 라인을 통해 보내온 데이터를 프린터가 정상적인 속도로 처리하지 못하기 때문이다(이 문제는 프린터가 패러럴 포트에 연결되어 있으면 발생하지 않는다). 이 문제를 해결하는 두 가지 방법이 있다:

- 프린터가 XON/XOFF 흐름제어를 지원한다면 *ms#* 문자열에서 *ixon* 모드를 지정하여 FreeBSD 에서 사용한다.
- 프린터가 캐리어(carrier) 흐름제어를 지원한다면 *ms#* 문자열에서 *crtscs* 모드를 지정한다. 프린터와 컴퓨터간의 연결 케이블은 정확히 캐리어 흐름 제어를 할수 있는 케이블 이어야 된다.

### 알아볼 수 없는 내용을 출력한다.

원하는 텍스트가 아니고 일정하지 않은 내용을 출력한다.

이것은 보통 시리얼 프린터에서 통신 매개변수를 잘못 사용한 또 다른 증상이다. *br* 문자열에서 *bps* 율을 *ms#* 문자열의 페리티 설정을 다시 체크한다; 프린터가 */etc/printcap* 파일과 같은 설정을 사용하는지 확인한다.

### 아무 일도 발생하지 않는다.

아무 일도 발생하지 않는다면 하드웨어 문제가 아닌 FreeBSD 문제일 것이다. */etc/printcap* 파일에 프린터를 디버깅하기 위해 로그 파일(/) 문자열 엔트리를 추가한다. 예를 들면 여기 *rattan* 를 위한 *lf* 문자열 엔트리가 있다:

```
rattan|line|diablo|lp|Diablo 630 Line Printer:W
```



---

```
:sh:sd=/var/spool/lpd/rattan:W
:lp=/dev/lpt0:W
:if=/usr/local/libexec/ifsimple:W
:lf=/var/log/rattan.log
```

그리고 다시 출력한다. 발생한 에러메시지를 확인하기 위해 로그파일을(우리 예제에서 /var/log/rattan.log) 체크한다. 이 메시지를 확인하여 정확히 문제를 파악한다.

If 문자열을 지정하지 않았다면 **LPD**는 기본적으로 /dev/console 를 사용한다.

## 10 장 리눅스 바이너리 호환성

### 10.1 요약

FreeBSD 는 리눅스를 포함하여 여러 다른 유닉스 운영체제와 호환되는 바이너리를 제공한다. 여기서 엄밀하게 왜 FreeBSD 가 리눅스 바이너리를 실행할 수 있어야 하는지? 자문할지도 모른다. 대답은 질문이 아주 단순하다는 것이다. 컴퓨터 세계에서 마지막 이슈가 리눅스기 때문에 수많은 회사와 개발자들은 리눅스를 위해서만 개발한다. 이 문제는 대부분의 이런 회사들이 FreeBSD 버전도 만들었다면 얼마나 많은 사람들이 그들의 제품을 사용할지 모르기 때문에 대부분 리눅스를 위해서만 개발하고 있다. 그래서 FreeBSD 유저는 무엇을 해야 되는가? 이것이 FreeBSD 가 리눅스 바이너리 호환성을 제공하는 이유다.

요약해 보면 이 호환성으로 FreeBSD 유저가 모든 리눅스 어플리케이션의 90%를 수정하지 않고 실행시킬 수 있다. 이들은 **StarOffice™**, 리눅스 버전의 **Netscape®**, **Adobe® Acrobat®**,

---

RealPlayer® 5 와 7, VMware™, Oracle®, WordPerfect®, Doom, Quacke 등 다수의 어플리케이션을 포함한다. 특정 상황에서는 리눅스 보다 FreeBSD 에서 리눅스 바이너리 성능이 더 좋다고 보고된 것도 있다.

그러나 FreeBSD 에서 지원하지 않는 특정 리눅스 운영체제에서만 실행되는 것도 있다. 리눅스 /proc 파일 시스템이나 가상 8086 모드(8086 의 확장된 개념으로 8086 에서 만들어진 프로그램과 호환될 수 있도록 하는 방식) 활성화처럼 i386™의 특정 콜을 과도하게 사용하는 리눅스 바이너리는 FreeBSD 에서 동작하지 않는다.

이번 장을 읽고 다음과 같은 사항을 알수 있다:

- 시스템에서 리눅스 바이너리 호환성을 어떻게 활성화하는가.
- 추가적인 리눅스 공유 라이브러리는 어떻게 설치하는가.
- FreeBSD 시스템에 리눅스 어플리케이션을 어떻게 설치하는가.
- FreeBSD의 리눅스 호환성에 대한 자세한 사항

이번 장을 읽기 전에 다음 사항을 알고 있어야 한다:

- 소프트웨어를 어떻게 설치하는가 (4장)

## 10.2 설치

리눅스 바이너리 호환성은 기본적으로 꺼져 있다. 이 기능을 사용하는 가장 쉬운 방법은 *linux* KLD 오브젝트를("커널이 로드할 수 있는 오브젝트") 로드한다. 명령 프롬프트에서 *linux* 만 입력하여 이 모듈을 로드할 수 있다. 설치하는 동안 리눅스 바이너리 호환성을 설치했다면 다음과 같은 결과를 볼 수 있다:

```
# linux
Linux driver already loaded
```

리눅스 호환성을 항상 사용하려면 다음 라인을 */etc/rc.conf* 에 추가한다:

---

linux\_enable="YES"

kldstat(8) 명령은 KLD 가 로드 되었는지 확인하는데 사용할 수 있다:

```
% kldstat
Id Refs Address      Size      Name
  1    2 0xc0100000 16bdb8   kernel
  7    1 0xc24db000 d000     linux.ko
```

어떤 이유로 KLD 로드를 원치 않거나 로드할 수 없다면 커널 설정파일에 *options COMPAT\_LINUX*를 추가하여 리눅스 바이너리 호환성을 커널에 정적으로 링크한다. 8 장의 설명대로 새로운 커널을 설치한다.

## 10.2.1 리눅스 런타임 라이브러리 설치

두 가지 방법 중 한가지를 사용하여 설치할 수 있다; linux\_base 포트를 사용하거나 이들을 수동으로 설치하면 된다.

### 10.2.1.1 linux\_base 포트를 사용하여 설치하기

이것이 런타임 라이브러리를 설치하는 가장 쉬운 방법이다. 단순히 포트 컬렉션에서 다른 포트처럼 설치하면 된다. 다음 명령을 실행한다:

```
# cd /usr/ports/emulators/linux_base
# make install distclean
```

이제 리눅스 바이너리 호환성이 작동한다. 어떤 프로그램은 정확하지 않은 시스템 라이브러리 버전으로 문제가 발생한다. 그러나 이것은 보통 문제가 되지 않는다.

**Note:** 다른 버전의 리눅스 배포본처럼 여러 가지 버전의 emulators/linux\_base 포트가 있을 것이다. 설치를 원하는 리눅스 어플리케이션에서 필요한 가장 최신 포트를 설치하면 된다.

---

### 10.2.1.2 수동으로 라이브러리 설치

"포트" 컬렉션이 설치되지 않았다면 수동으로 라이브러리를 설치할 수 있다. 런타임 링커(linker)에 의존하는 프로그램을 사용할때 리눅스 공유 라이브러리가 필요하다. 또한 FreeBSD 시스템의 리눅스 라이브러리를 위해 "shadow root" 디렉터리 /compat/linux 를 생성해야 된다. FreeBSD 에서 실행되는 모든 리눅스 프로그램은 공유 라이브러리를 열고 이 트리를 처음으로 확인한다. 그래서 예를 들면 리눅스 프로그램이 /lib/libc.so 를 로드한다면 FreeBSD 는 /compat/linux/lib/libc.so 를 처음으로 열고 이 파일이 없다면 /lib/libc.so 를 연다. 공유 라이브러리는 리눅스 ld.so 리포트의 경로보다 shadow 트리 /compat/linux/lib 에 설치되어야 한다.

FreeBSD 시스템에 설치한 리눅스 바이너리가 처음에만 의존하는 공유 라이브러리를 찾아야 한다. 잠시 후 추가적인 작업 없이 새롭게 설치된 리눅스 바이너리를 실행할 수 있는 만족할 만한 리눅스 공유 라이브러리를 시스템에 설치할 수 있다.

### 10.2.1.3 부가적인 공유 라이브러리를 어떻게 추가하는가

linux\_base 포트를 설치했지만 아직도 어플리케이션이 다른 공유 라이브러리를 원한다면 어떤 공유 라이브러리를 리눅스 바이너리가 필요로 하고 어디서 구할 수 있는지 어떻게 알 수 있는가? 기본적으로 2 가지 가능성이 있다(FreeBSD 시스템에서 다음 지시 사항을 따를 때 root 권한이 필요하다).

리눅스 시스템을 사용할 수 있다면 어떤 공유 라이브러리가 어플리케이션에 필요한지 살펴보고 FreeBSD 시스템에 복사한다. 다음 예제를 확인한다:

리눅스 바이너리의 **Doom** 을 FTP 로 받아왔고 이것을 리눅스 시스템에 넣었다고 가정한다. 어떤 공유 라이브러리가 필요한지 ldd linuxdoom 을 실행해서 다음과 같이 체크할 수 있다:

```
% ldd linuxdoom
libXt.so.3 (DLL Jump 3.1) => /usr/X11/lib/libXt.so.3.1.0
libX11.so.3 (DLL Jump 3.1) => /usr/X11/lib/libX11.so.3.1.0
libc.so.4 (DLL Jump 4.5pl26) => /lib/libc.so.4.6.29
```

마지막 칼럼까지 모든 파일을 가져와서 심볼릭 링크가 지정하듯이 /compat/linux 밑에

---

넣는다. 이 말은 드디어 FreeBSD 시스템에 이들 파일을 가질 수 있다는 의미다:

```
/compat/linux/usr/X11/lib/libXt.so.3.1.0
/compat/linux/usr/X11/lib/libXt.so.3 -> libXt.so.3.1.0
/compat/linux/usr/X11/lib/libX11.so.3.1.0
/compat/linux/usr/X11/lib/libX11.so.3 -> libX11.so.3.1.0
/compat/linux/lib/libc.so.4.6.29
/compat/linux/lib/libc.so.4 -> libc.so.4.6.29
```

**Note:** 이미 ldd 출력의 첫 번째 칼럼과 똑같은 버전의 리눅스 공유 라이브러리를 가지고 있다면 공유 라이브러리가 동작하고 있기 때문에 마지막 칼럼에 있는 이름의 파일을 시스템에 복사할 필요가 없다. 새로운 버전의 공유 라이브러리면 어쨌든 복사하는 것을 권장한다. 새로운 것을 심볼릭 링크로 만들었다면 예전 것을 삭제할 수 있다.

```
/compat/linux/lib/libc.so.4.6.27
/compat/linux/lib/libc.so.4 -> libc.so.4.6.27
```

그래서 이러한 공유 라이브러리를 시스템에 가지고 있다면 ldd의 출력에 따라 새로운 버전의 필요한 바이너리를 찾는다:

```
libc.so.4 (DLL Jump 4.5pl26) -> libc.so.4.6.29
```

마지막 한 두 개의 버전만 다르다면 약간 예전 버전의 프로그램에서 정확히 동작하기 때문에 /lib/libc.so.4.6.29를 복사하면 된다. 그러나 원한다면 libc.so로 교체할 수 있다.

```
/compat/linux/lib/libc.so.4.6.29
/compat/linux/lib/libc.so.4 -> libc.so.4.6.29
```

**Note:** 심볼릭 링크 메커니즘은 리눅스 바이너리에만 필요하다. FreeBSD 런타임 링커는 주 버전 번호가 매칭되는지 확인하기 때문에 걱정할 필요는 없다.

## 10.2.2 리눅스 ELF 바이너리 설치

---

ELF 바이너리는 가끔 추가적인 "브랜딩(branding)" 단계가 필요하다. 브랜드가 없는 ELF 바이너리를 실행하려고 하면 다음과 같은 에러 메시지를 보게 된다:

```
% ./my-linux-elf-binary
ELF binary type not known
Abort
```

리눅스 바이너리에서 FreeBSD ELF 바이너리를 FreeBSD 커널이 구분할 수 있도록 `brandelf(1)` 유틸리티를 사용한다.

```
% brandelf -t Linux my-linux-elf-binary
```

이제 GNU 툴 체인이(toolchain) 적당한 브랜딩 정보를 ELF 바이너리에 자동으로 넣기 때문에 나중에 이 단계가 필요없다.

### 10.2.3 호스트 네임 리졸버(resolver) 설정

DNS 가 작동하지 않거나 다음과 같은 메시지를 보게 된다면:

```
resolv+: "bind" is an invalid keyword resolv+
"hosts" is an invalid keyword
```

`/compat/linux/etc/host.conf` 파일에 다음 내용을 추가해야 된다:

```
order hosts, bind
multi on
```

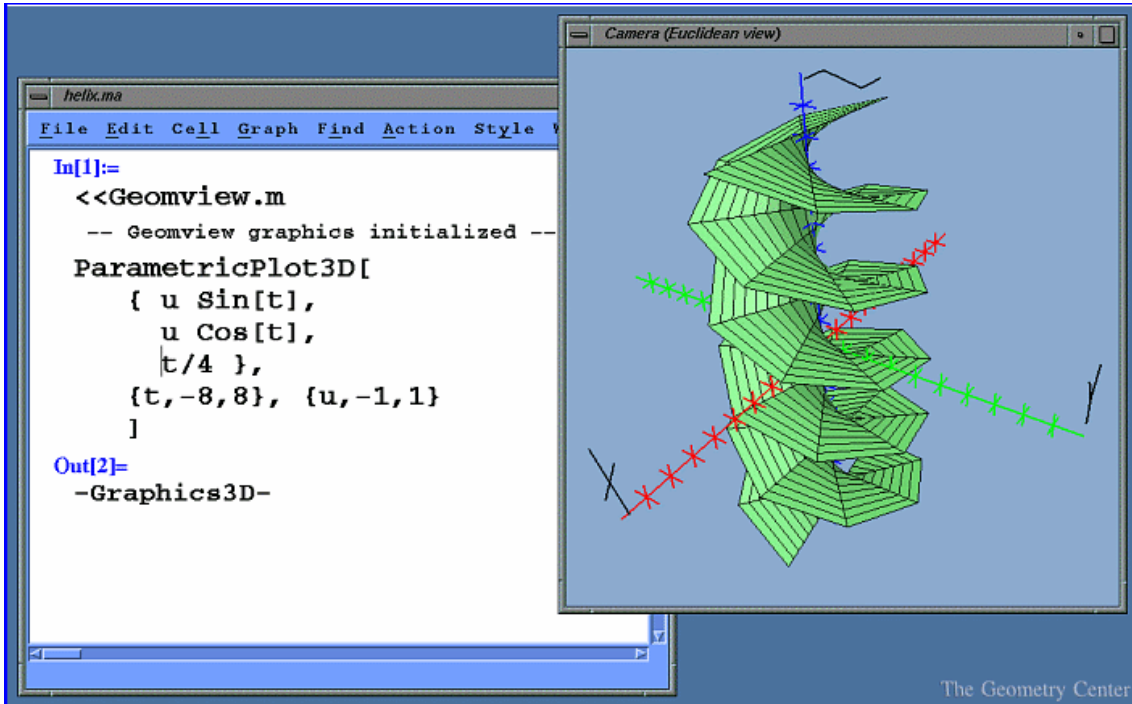
여기서 순서는 처음에 `/etc/hosts` 를 검색하고 DNS 는 두 번째로 검색하도록 지정한다. `/compat/linux/etc/host.conf` 가 설치되지 않았을 때 리눅스 어플리케이션은 FreeBSD 의 `/etc/host.conf` 를 찾고 맞지 않는 FreeBSD 구문에 대해 알려준다. `/etc/resolv.conf` 파일로 네임서버를 설정하지 않았다면 `bind(DNS 서버)`를 삭제한다.

---

## 10.3 Mathematica® 설치

이 문서는 리눅스 버전의 **Mathematica®** (4.X)를 FreeBSD 시스템에 설치하는 단계를 설명한다.

### Mathematica 샘플 예제



리눅스 버전의 **Mathematica**는 FreeBSD에서 완벽하게 동작하지만 바이너리에 Wolfram 브랜드 적용이 필요하기 때문에 FreeBSD는 리눅스 ABI로 이 바이너리를 실행하는 것을 알고 있다.

리눅스 버전의 **Mathematica**나 학생용 **Mathematica**는 직접 Wolfram <http://www.wolfram.com/>에서 구입할 수 있다.

### 10.3.1 리눅스 바이너리 브랜딩

리눅스 바이너리는 Wolfram이 배포하는 **Mathematica** CDROM의 Unix 디렉터리에 있다. 이 디렉터리 트리를 로컬 하드 드라이브로 복사해서 인스톨러를 실행하기 전에 `brandelf(1)`으로 리눅스 바이너리를 브랜드 할 수 있다:

---

```
# mount /cdrom
# cp -rp /cdrom/Unix/ /localdir/
# brandelf -t Linux /localdir/Files/SystemFiles/Kernel/Binaries/Linux/*
# brandelf -t Linux /localdir/Files/SystemFiles/FrontEnd/Binaries/Linux/*
# brandelf -t Linux /localdir/Files/SystemFiles/Installation/Binaries/Linux/*
# brandelf -t Linux /localdir/Files/SystemFiles/Graphics/Binaries/Linux/*
# brandelf -t Linux /localdir/Files/SystemFiles/Converters/Binaries/Linux/*
# brandelf -t Linux /localdir/Files/SystemFiles/LicenseManager/Binaries/Linux/mathlm
# cd /localdir/Installers/Linux/
# ./MathInstaller
```

다른 방법은 다음 명령으로 브랜드되지 않은 모든 바이너리에 기본 ELF 브랜드를 설정할 수 있다:

```
# sysctl kern.fallback_elf_brand=3
```

위 명령은 FreeBSD 가 브랜드 되지 않은 ELF 바이너리에 리눅스 ABI 를 사용하도록 하기 때문에 CDROM 에서 직접 인스톨러를 실행할 수 있다.

## 10.3.2 Mathematica 패스워드 받기

**Mathematica** 를 실행하기 전에 머신 ID 에 맞는 패스워드를 Wolfram 에서 받아야 한다.

리눅스 호환성 런타임 라이브러리를 설치한 후 설치 디렉터리의 압축이 풀린

**Mathematica** 에서 mathinfo 프로그램을 실행하여 머신 ID 를 받는다. 이 머신 ID 는 첫 번째 이더넷 카드의 MAC 주소와 관련 있다.

```
# cd /localdir/Files/SystemFiles/Installation/Binaries/Linux
# mathinfo
disco.example.com 7115-70839-20412
```

메일, 전화 또는 팩스로 Wolfram 에 등록할때 머신 ID 를 그들에게 주면 그룹으로 이루어진 그룹 패스워드를 받는다. 플랫폼에 상관없이 처음으로 **Mathematica** 를 실행할 때 이 정보를 입력한다.



---

### 10.3.3 네트워크로 Mathematica 를 전면에서 운용하기

**Mathematica** 는 표준 폰트세트로(정수, 합, 그리스 문자 등) 표현하지 못하는 문자를 표시하기 위해 특수한 폰트를 사용한다. X 프로토콜은 이들 문자가 로컬에 설치되어 있어야 한다. 이 의미는 이들 폰트를 CDROM 또는 **Mathematica** 가 설치된 호스트로부터 로컬 머신에 복사해야 된다는 것이다. 이들 폰트는 보통 CDROM 의 /cdrom/Unix/Files/SystemFiles/Fonts 에 있거나 하드 드라이브의 /usr/local/mathematica/SystemFiles/Fonts 에 있다. 실재 폰트는 Type1 과 X 서브 디렉터리에 있다. 위에서 설명했듯이 이들을 사용하는 여러 가지 방법이 있다.

첫 번째 방법은 /usr/X11R6/lib/X11/fonts 의 폰트디렉터리 한곳에 복사하는 것이다. 이 방법은 fonts.dir 파일에 폰트이름을 넣고 첫 번째 라인의 수많은 폰트번호를 바꾸어야 한다. 그렇지 않으면 폰트를 복사해 넣은 디렉터리에서 mkfontdir(1)을 실행하면 된다.

두 번째 방법은 디렉터리를 /usr/X11R6/lib/X11/fonts 에 복사하는 것이다:

```
# cd /usr/X11R6/lib/X11/fonts
# mkdir X
# mkdir MathType1
# cd /cdrom/Unix/Files/SystemFiles/Fonts
# cp X/* /usr/X11R6/lib/X11/fonts/X
# cp Type1/* /usr/X11R6/lib/X11/fonts/MathType1
# cd /usr/X11R6/lib/X11/fonts/X
# mkfontdir
# cd ../MathType1
# mkfontdir
```

이제 새로운 폰트디렉터리를 폰트경로에 추가한다:

```
# xset fp+ /usr/X11R6/lib/X11/fonts/X
# xset fp+ /usr/X11R6/lib/X11/fonts/MathType1
# xset fp rehash
```

XFree86™ 서버를 사용 중이라면 XF86Config 파일에 이들 폰트 디렉터리를 추가하여

---

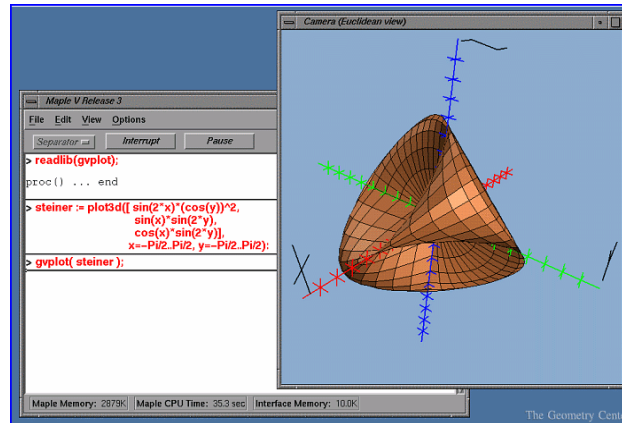
자동으로 로드할 수 있다.

/usr/X11R6/lib/X11/fonts/Type1 이라는 디렉터리를 가지고 있지 않다면 위의 예제에서 MathType1 의 이름을 Type1 로 바꿀 수 있다.

## 10.4 Maple™ 설치

Maple™ 은 Mathematica®와 비슷한 상용 계산프로그램이다. 이 소프트웨어를 <http://www.maplesoft.com/>에서 구입하고 라이선스 파일을 등록해야 된다.

### Maple 샘플 예제



이 소프트웨어를 FreeBSD 에 설치하려면 단순히 다음 단계를 따른다.

### [예제: Maple 설치]

- ① 제품에서 INSTALL 쉘 스크립트를 실행한다. 설치 프로그램에 프롬프트가 나올 때 "RedHat" 옵션을 선택한다. 일반적인 설치 디렉터리는 /usr/local/maple이다.
- ② 설치가 끝나지 않는다면 Maple Waterloo Software (<http://register.maplesoft.com>)에서 라이선스를 신청해서 이 라이선스를 /usr/local/mple/license/license.dat에 복사 한다.
- ③ Maple에있는 INSTALL\_LIC 설치 스크립트를 실행하여 FLEXlm 라이선스를 설치한다. 라이선스 서버에게 머신의 주 호스트 이름을 지정한다.

- ④ 다음 내용으로 /usr/local/mple/bin/mple.system.type 파일을 패치한다:

```
----- snip -----
*** maple.system.type.orig      Sun Jul  8 16:35:33 2001
--- maple.system.type    Sun Jul  8 16:35:51 2001
*****
*** 72,77 ****
--- 72,78 -----
        # the IBM RS/6000 AIX case
        MAPLE_BIN="bin.IBM_RISC_UNIX"
        ;;
+   "FreeBSD"|W
    "Linux")
        # the Linux/x86 case
        # We have two Linux implementations, one for Red Hat and
----- snip end of patch -----
```

*"FreeBSD"|W* 뒤에 공백이 들어가지 않도록 주의한다.

이 패치는 **Maple** 이 "FreeBSD"를 리눅스 시스템의 일종으로 감지하도록 한다.

bin/maple 쉘 스크립트는 운영체제 이름을 찾기 위해 `uname -a` 를 수행하는 bin/maple.system.type 쉘 스크립트를 불러온다. OS 이름에 따라 어떤 바이너리를 사용할지 찾는다.

- ⑤ 라이선스 서버 시작

/usr/local/etc/rc.d/lmgrd.sh 에 설치되어있는 다음 스크립트가 lmgrd 를 시작하는 편리한 방법이다:

```
----- snip -----

#!/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/X11R6/bin
PATH=${PATH}:/usr/local/maple/bin:/usr/local/maple/FLEXlm/UNIX/LINUX
export PATH
```

```

LICENSE_FILE=/usr/local/maple/license/license.dat
LOG=/var/log/lmgrd.log

case "$1" in
start)
    lmgrd -c ${LICENSE_FILE} 2>> ${LOG} 1>&2
    echo -n " lmgrd"
    ;;
stop)
    lmgrd -c ${LICENSE_FILE} -x lmdown 2>> ${LOG} 1>&2
    ;;
*)
    echo "Usage: `basename $0` {start|stop}" 1>&2
    exit 64
    ;;
esac

exit 0

----- snip -----

```

⑥ **Maple** 테스트 시작:

```

% cd /usr/local/maple/bin
% ./xmaple

```

Maplesoft 에게 여러분은 FreeBSD 버전의 프로그램이 필요하다고 메일을 보낸다.

### 10.4.1 일반적인 함정

- **FLEXlm** 라이선스 매니저는 동작시키기 어려운 툴이다. 추가적인 문서는 <http://www.globetrotter.com/>에서 찾을 수 있다.

- 
- 문제가 발생했다면 Imgrd는 라이선스 파일과 core dump에 대해 까다로운 것으로 알려져 있다. 정확한 라이선스 파일은 다음과 같이 생겼다:

```
# =====  
# License File for UNIX Installations ("Pointer File")  
# =====  
SERVER chillig ANY  
#USE_SERVER  
VENDOR mapleimg  
  
FEATURE Maple mapleimg 2000.0831 permanent 1 XXXXXXXXXXXX W  
    PLATFORMS=i86_r ISSUER="Waterloo Maple Inc." W  
    ISSUED=11-may-2000 NOTICE=" Technische Universitat Wien" W  
    SN=XXXXXXXXXX
```

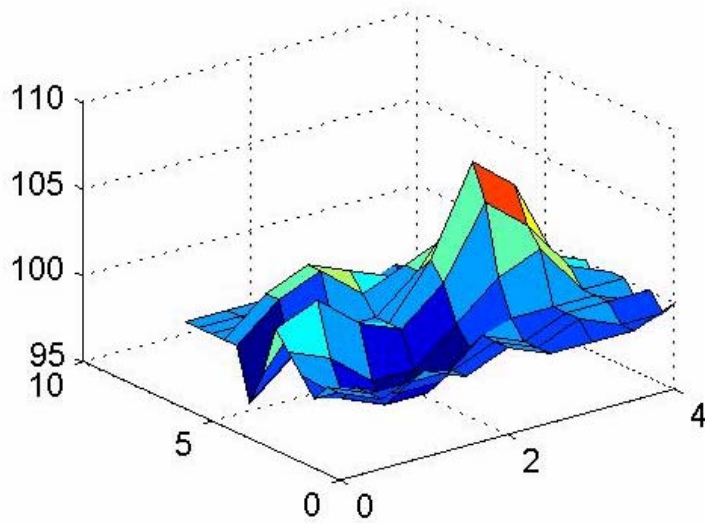
**Note:** 시리얼 번호와 키는 "X"로 나타냈다. chillig 는 호스트 네임이다.

"FEATURE" 라인을(라이선스 키로 보호되기 때문에) 건드리지 않고 라이선스 파일을 수정한다.

## 10.5 MATLAB® 설치

이 문서는 리눅스 버전의 **MATLAB®** 버전 6.5 를 FreeBSD 시스템에 설치하는 절차에 대해 설명한다. **Java** 가상 머신을(10.5.3 장을 본다) 제외하고 정확히 동작한다.

**MATLAB 샘플 예제**



리눅스 버전의 **MATLAB** 는 MathWorks <http://www.mathworks.com> 에서 직접 구입할 수 있다. 또한 라이선스 파일을 받거나 생성하는 방법도 알 수 있다. 그들에게 여러분은 FreeBSD 버전의 소프트웨어가 필요하다고 얘기한다.

## 10.5.1 MATLAB 설치

**MATLAB** 설치는 다음 절차를 따른다:

- ① 설치 스크립트에서 권장하듯이 root에서 설치 CD를 넣고 마운트한다. 설치 스크립트를 시작하려면 다음 명령을 입력한다:

```
# /compat/linux/bin/sh /cdrom/install
```

**Tip:** 인스톨러는 그래픽으로 보여준다. 디스플레이를 열 수 없다는 에러 메시지를 보여 주면 `setenv HOME ~ USER`를 입력한다. `USER`는 `su(1)`를 수행하기 전의 유저다.

- ② **MATLAB** root 디렉터리를 요청하면 `/compat/linux/usr/local/matlab`을 입력한다:

**Tip:** 나머지 설치 절차에서 입력하기 쉽게 셸 프롬프트에 입력한다: `set`

---

MATLAB=/compat/linux/usr/local/matlab

- ③ **MATLAB** 라이선스를 받았을 때 설명한 대로 라이선스 파일을 수정한다.

**Tip:** 인스톨러가 라이선스 파일을 수정하라고 요청하기 전에 좋아하는 에디터로 이 파일을 준비해서 \$MATLAB/license.dat 에 복사한다.

- ④ 설치 절차를 끝낸다.

이제 **MATLAB** 설치가 끝났다. 다음 단계는 FreeBSD 시스템에 "glue"를 적용한다.

## 10.5.2 라이선스 매니저 시작

- ① 라이선스 매니저 스크립트에 심볼릭 링크를 생성한다:

```
# ln -s $MATLAB/etc/lmboot /usr/local/etc/lmboot_TMW
# ln -s $MATLAB/etc/lmdown /usr/local/etc/lmdown_TMW
```

- ② /usr/local/etc/rc.d/flexlm.sh에 시작 파일을 생성한다. 아래 예제는 배포된 \$MATLAB/etc/rc.lm.glnx86의 수정된 버전이다. 파일 위치를 변경하고 리눅스 에뮬레이션에서 라이선스 매니저를 시작한다.

```
#!/bin/sh
case "$1" in
  start)
    if [ -f /usr/local/etc/lmboot_TMW ]; then
      /compat/linux/bin/sh /usr/local/etc/lmboot_TMW -u
      username && echo 'MATLAB_lmgrd'
    fi
    ;;
  stop)
    if [ -f /usr/local/etc/lmdown_TMW ]; then
      /compat/linux/bin/sh /usr/local/etc/lmdown_TMW > /dev/null
    fi
  2>&1
```

---

```
    fi
        ;;
*)
    echo "Usage: $0 {start|stop}"
    exit 1
    ;;
esac

exit 0
```

**중요:** 파일을 실행할 수 있어야 한다:

```
# chmod +x /usr/local/etc/rc.d/flexlm.sh
```

위의 *username* 은 시스템에 있는 유저 이름을(*root* 가 아닌) 넣는다.

③ 다음 명령으로 라이선스 매니저를 시작한다:

```
# /usr/local/etc/rc.d/flexlm.sh start
```

### 10.5.3 Java 런타임 환경 링크

FreeBSD 에서 동작하도록 Java 런타임 환경의 링크를 변경한다:

```
# cd $MATLAB/sys/java/jre/glnx86/
```

```
# unlink jre; ln -s ./jre1.1.8 ./jre
```

### 10.5.4 MATLAB 시작 스크립트 생성

① 다음 시작 스크립트를 /usr/local/bin/matlab에 입력한다:

```
#!/bin/sh
/compat/linux/bin/sh /compat/linux/usr/local/matlab/bin/matlab "$@"
```

② 그리고 `chmod +x /usr/local/bin/matlab` 명령을 실행한다.



---

**Tip:** `emulators/linux_base` 버전에 따라 이 스크립트를 실행할 때 에러를 볼 수 있다. 에러를 피하기 위해 `/compat/linux/usr/local/matlab/bin/matlab` 파일을 다음과 같이 변경한다:

```
if [ `expr "$lscmd" :'.*->.*'` -ne 0 ];
```

위의 라인을 (버전 13.0.1에서는 410 라인이다.) 다음 라인으로 변경한다:

```
if test -L $newbase; then
```

## 10.5.5 MATLAB 섯다운 스크립트 생성

다음은 MATLAB 가 정확히 끝나지 않는 문제를 해결하기 위해 필요하다.

- ① `$MATLAB/toolbox/local/finish.m` 파일을 만들고 다음 라인을 입력한다:

```
! $MATLAB/bin/finish.sh
```

**Note:** `$MATLAB` 는 문자다.

**Tip:** 빠져 나가기 전의 작업 환경을 저장하는 `finis sav.m` 과 `finishdig.m` 파일을 같은 디렉터리에서 찾을 수 있다. 이들 중 하나를 사용한다면 `save` 명령을 입력하고 즉시 위 라인을 입력한다.

- ② 다음 내용이 있는 `$MATLAB/bin/finish.sh` 파일을 생성한다:

```
#!/usr/compat/linux/bin/sh
(sleep 5; killall -1 matlab_helper) &
exit 0
```

- ③ 파일을 실행할 수 있게 만든다:

```
# chmod +x $MATLAB/bin/finish.sh
```

---

## 10.5.6 MATLAB 사용

matlab 를 입력해서 사용할 준비가 됐다.

## 10.6 오라클 설치

### 10.6.1 서문

이 문서는 리눅스용 오라클 8.0.5 와 오라클 8.0.5.1 엔터프라이즈 에디션을 FreeBSD 머신에 설치하는 과정을 설명한다.

### 10.6.2 리눅스 환경 설치

emulators/linux\_base 와 devel/linux\_devtools 를 포트 컬렉션에서 설치해야 된다. 이들을 포트에서 설치하기 어렵다면 패키지나 포트 컬렉션에서 예전 버전을 사용할 수 있다.

인텔리전트 에이전트를 사용하겠다면 Red Hat Tcl 패키지도 설치해야 된다: tcl-8.0.3-20.i386.rpm. 공식 RPM 포트에서(archivers/rpm) 패키지를 설치하는 일반적인 명령은 다음과 같다:

```
# rpm -i --ignoreos --root /compat/linux --dbpath /var/lib/rpm package
```

패키지 설치에 어려움이 없어야 한다.

### 10.6.3 오라클 환경 생성

오라클을 설치하기 전에 적당한 환경을 설정해야 된다. 이 문서는 오라클 설치 가이드가 아닌 리눅스용 오라클을 FreeBSD 에 설치할때 특별히 필요한 것에 대해 설명한다.

#### 10.6.3.1 커널 튜닝

오라클 설치 가이드에서 설명하듯이 최대 크기의 공유 메모리를 설정해야 된다. FreeBSD 에서는 *SHMMAX*를 사용하지 않는다. *SHMMAX*는 단지 *SHMMAXPGS*와 *PGSIZE*를 계산할 뿐이다. 따라서 *SHMMAXPGS*를 정의해야 된다. 여기에 사용할 수 있는 다른 모든 옵션은 가이드에 설명되어 있다. 예를 들면 다음과 같이 정의한다:

```
options SHMMAXPGS=10000
options SHMMNI=100
options SHMSEG=10
options SEMMNS=200
options SEMMNI=70
options SEMMSL=61
```

이들 옵션을 사용하려는 오라클 용도에 맞게 설정한다.

그리고 다음 옵션을 커널 설정 파일에 지정한다:

```
options SYSVSHM #SysV shared memory
options SYSVSEM #SysV semaphores
options SYSVMSG #SysV interprocess communication
```

### 10.6.3.2 오라클 계정

다른 계정을 만들듯이 *oralc* 계정을 만든다. *oracle* 계정은 특별하기 때문에 리눅스 셸을 줘야 된다. */etc/shells* 에 */compat/linux/bin/bash*를 추가하고 *oracle* 계정 셸을 */compat/linux/bin/bash*로 설정한다.

### 10.6.3.3 환경

*ORACLE\_HOME*과 *ORACLE\_SID* 같은 일반적인 오라클 환경 외에도 다음 환경 변수를 지정해야 된다:

변수	값
<i>LD_LIBRARY_PATH</i>	<i>\$ORACLE_HOME/lib</i>
<i>CLASSPATH</i>	<i>\$ORACLE_HOME/jdbc/lib/classes111.zip</i>

PATH	<pre> /compat/linux/bin /compat/linux/sbin /compat/linux/usr/bin /compat/linux/usr/sbin /bin /sbin /usr/bin /usr/sbin /usr/local/bin \$ORACLE_HOME/bin </pre>
------	---

모든 환경 변수를 .profile 에 설정할 것을 권장한다. 전체적인 예제는 다음과 같다:

```

ORACLE_BASE=/oracle; export ORACLE_BASE
ORACLE_HOME=/oracle; export ORACLE_HOME
LD_LIBRARY_PATH=$ORACLE_HOME/lib
export LD_LIBRARY_PATH
ORACLE_SID=ORCL; export ORACLE_SID
ORACLE_TERM=386x; export ORACLE_TERM
CLASSPATH=$ORACLE_HOME/jdbc/lib/classes111.zip
export CLASSPATH
PATH=/compat/linux/bin:/compat/linux/sbin:/compat/linux/usr/bin
PATH=$PATH:/compat/linux/usr/sbin:/bin:/sbin:/usr/bin:/usr/sbin
PATH=$PATH:/usr/local/bin:$ORACLE_HOME/bin
export PATH

```

## 10.6.4 오라클 설치

리눅스 에뮬레이터의 약간의 불일치로 인스톨러를 시작하기 전에 /var/tmp 에 .oracle 이라는 이름의 디렉토리를 생성해야 된다. 누구나 쓰기를 할수 있거나 oracle 유저만 쓰기를 하도록 한다. **오라클**을 아무런 문제없이 설치할 수 있다. 문제가 발생 한다면 첫째로 오라클 배포판이나 설정을 체크한다. **오라클**을 설치하고 다음 두 개의 서브 섹션에서 설명하는 패치를 적용한다.

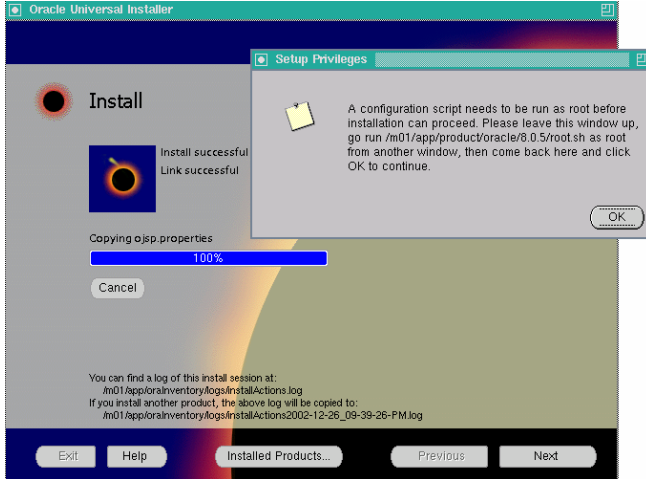
가끔 발생하는 문제는 TCP 프로토콜 아답터가 제대로 설치되지 않는 것이다. 그 결과 어떤 TCP 리스너(listener)도 시작하지 못한다. 다음은 이 문제를 해결하는데 도움을 준다:

```

# cd $ORACLE_HOME/network/lib
# make -f ins_network.mk ntcontab.o
# cd $ORACLE_HOME/lib
# ar r libnetwork.a ntcontab.o
# cd $ORACLE_HOME/network/lib
# make -f ins_network.mk install

```

root.sh 를 다시 실행하는 것을 잊지 않는다!



#### 10.6.4.1 root.sh 패치

오라클을 설치할 때 root 에서 실행해야 되는 몇 가지가 root.sh 라는 스크립트에 저장되어 있다. 이 스크립트는 oraInst 디렉터리에 작성되어 있다. 다음 패치를 root.sh 에 적용하고 chown 를 사용할 적당한 위치로 이동시키거나 리눅스 셸에서 스크립트를 실행한다.

```
*** oraInst/root.sh.orig Tue Oct 6 21:57:33 1998
--- oraInst/root.sh Mon Dec 28 15:58:53 1998
*****
*** 31,37 ****
# This is the default value for CHOWN
# It will redefined later in this script for those ports
# which have it conditionally defined in ss_install.h
! CHOWN=/bin/chown
#
# Define variables to be used in this script
--- 31,37 -----
# This is the default value for CHOWN
# It will redefined later in this script for those ports
# which have it conditionally defined in ss_install.h
! CHOWN=/usr/sbin/chown
```

---

```
#
# Define variables to be used in this script
```

오라클을 CD 에서 설치하지 않았다면 root.sh 소스를 패치할 수 있다. 이것은 rthd.sh 라고 하고 소스 트리의 orainst 디렉터리에 있다.

### 10.6.4.2 genclntsh 패치

스크립트 genclntsh 는 싱글 공유 클라이언트 라이브러리를 생성하는데 사용된다. demo 를 빌드할 때 사용된다. PATH 주석을 풀어서 다음 패치를 적용한다:

```
*** bin/genclntsh.orig Wed Sep 30 07:37:19 1998
--- bin/genclntsh Tue Dec 22 15:36:49 1998
*****
*** 32,38 ***
#
# Explicit path to ensure that we're using the correct commands
#PATH=/usr/bin:/usr/ccs/bin export PATH
! PATH=/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin export PATH
#
# each product MUST provide a $PRODUCT/admin/shrept.lst
--- 32,38 ----
#
# Explicit path to ensure that we're using the correct commands
#PATH=/usr/bin:/usr/ccs/bin export PATH
! #PATH=/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin export PATH
#
# each product MUST provide a $PRODUCT/admin/shrept.lst
```

### 10.6.5 오라클 실행

이 지시를 따랐다면 리눅스에서 실행하는 것처럼 오라클을 실행할 수 있다.

---

## 10.7 SAP® R/3® 설치

SAP® 시스템을 FreeBSD 에 설치하는 것은 SAP 지원 팀이 지원하지 않는다.

### 10.7.1 서문

이 문서는 FreeBSD 와 **오라클** 설치를 포함하여 FreeBSD 머신에 **SAP R/3** 시스템과 리눅스용 **오라클** 데이터베이스를 설치할 수 있는 방법을 설명한다. 두 가지 설정이 설명된다:

- FreeBSD 4.3-STABLE에 **오라클 8.0.5**와 **SAP R/3 4.6B (IDES)**
- FreeBSD 4.5-STABLE에 **오라클 8.1.7**과 **SAP R/3 4.6C**

이 문서가 중요한 모든 단계를 아주 자세히 설명하더라도 **오라클**과 **SAP R/3** 설치 가이드를 대체할 수 없다.

**오라클**과 **SAP OSS** 리소스뿐만 아니라 **SAP** 는 **SAP R/3** 리눅스용 에디션의 문서와 **오라클** 관련 질문도 참고한다.

### 10.7.2 소프트웨어

다음 CD-ROM 들이 **SAP** 설치에 사용된다:

#### 10.7.2.1 SAP R/3 4.6B, 오라클 8.0.5

이름	번호	설명
KERNEL	51009113	SAP Kernel Oracle / Installation / AIX, Linux, Solaris
RDBMS	51007558	Oracle / RDBMS 8.0.5.X / Linux
EXPORT1	51010208	IDES / DB-Export / Disc 1 of 6
EXPORT2	51010209	IDES / DB-Export / Disc 2 of 6

EXPORT3	51010210	IDES / DB-Export / Disc 3 of 6
EXPORT4	51010211	IDES / DB-Export / Disc 4 of 6
EXPORT5	51010212	IDES / DB-Export / Disc 5 of 6
EXPORT6	51010213	IDES / DB-Export / Disc 6 of 6

추가적으로 사실 필요하지 않은 오라클 8 서버 CD 와(리눅스용 버전 8.0.5, 커널 버전 2.0.33) FreeBSD 4.3-STABLE 를 사용했다(4.3 릴리즈 보다 며칠 전의).

### 10.7.2.2 SAP R/3 4.6C SR2, 오라클 8.1.7

이름	번호	설명
KERNEL	51014004	SAP Kernel Oracle / SAP Kernel Version 4.6D / DEC, Linux
RDBMS	51012930	Oracle 8.1.7/ RDBMS / Linux
EXPORT1	51013953	Release 4.6C SR2 / Export / Disc 1 of 4
EXPORT1	51013953	Release 4.6C SR2 / Export / Disc 2 of 4
EXPORT1	51013953	Release 4.6C SR2 / Export / Disc 3 of 4
EXPORT1	51013953	Release 4.6C SR2 / Export / Disc 4 of 4
LANG1	51013954	Release 4.6C SR2 / Language / DE, EN, FR / Disc 1 of 3

설치 하려는 언어에 따라 언어 CD 가 추가적으로 필요할 것이다. 우리는 DE 와 EN 을 사용하기 때문에 첫 번째 언어 CD 만 필요하다. 4 개의 EXPORT CD 번호는 모두 동일하다. 또한 3 개의 언어 CD 도 같은 번호를 가지고 있다 (이것은 4.6B IDES 릴리즈 CD 부터 달라졌다). 이 문서를 작성할 때 FreeBSD 4.5-STABLE 에서(20.03.2002) 설치했다.

### 10.7.3 SAP 노트

다음 노트를 SAP R/3 를 설치하기 전에 읽으면 설치하는 동안 유용하다:

#### 10.7.3.1 SAP R/3 4.6B, 오라클 8.0.5



번호	제목
0171356	SAP Software on Linux: Essential Comments
0201147	INST: 4.6C R/3 Inst. on UNIX – Oracle
0373203	Update / Migration Oracle 8.0.5 --> 8.0.6/8.1.6 LINUX
0072984	Release of Digital UNIX 4.0B for Oracle
0130581	R3SETUP step DIPGNTAB terminates
0144978	Your system has not been installed correctly
0162266	Questions and tips for R3SETUP on Windows NT / W2K

### 10.7.3.2 SAP R/3 4.6C 오라클 8.1.7

번호	제목
0015023	Initializing table TCPDB (RSXP0004) (EBCDIC)
0045619	R/3 with several languages or typefaces
0171356	SAP Software on Linux: Essential Comments
0195603	RedHat 6.1 Enterprise version: Known problems
0212876	The new archiving tool SAPCAR
0300900	Linux: Released DELL Hardware
0377187	RedHat 6.2: important remarks
0387074	INST: R/3 4.6C SR2 Installation on UNIX
0387077	INST: R/3 4.6C SR2 Inst. on UNIX – Oracle
0387078	SAP Software on UNIX: OS Dependencies 4.6C SR2

### 10.7.4 필요한 하드웨어

다음 장비는 SAP R/3 시스템 설치에 적당하다. 제품을 사용하기 위해 더 많은 공간이 필요하다:

컴포넌트	4.6B	4.6C
Processor	2 x 800MHz Pentium® III	2 x 800MHz Pentium III
Memory	1GB ECC	2GB ECC
Hard Disk Space	50-60GB (IDES)	50-60GB (IDES)

---

제품을 사용하기 위해 대형 케이스의 Xeon 프로세서와 고속 디스크 접근(SCSI, RAID 하드웨어 컨트롤러) 그리고 USV 와 ECC-RAM 이 권장된다. 설치하는 동안 27GB 의 데이터베이스 파일을 생성하는 거대한 양의 하드 디스크 공간이 IDES 시스템에 이미 설정되어 있다. 또한 이 공간은 제품 시스템과 어플리케이션 데이터를 초기화하는데 충분하다.

#### 10.7.4.1 SAP R/3 4.6B, 오라클 8.0.5

다음은 규격 하드웨어를 사용했다: 2 개의 800 MHz 펜티엄 III 프로세서가 있는 듀얼 프로세서 보드, Adaptec 29160 울트라 160 SCSI 아답터(40/80 GB DLT 테잎 드라이브와 CDROM 을 사용하기 위해), Mylex AcceleRAID (2 채널, 펌웨어 6.00-1-00 과 32MB 램), Mylex RAID 컨트롤러를 두 개의 17GB 하드 디스크와(미러 된) 4 개의 36GB 하드 디스크에(RAID 레벨 5) 연결한다.

#### 10.7.4.2 SAP R/3 4.6C, 오라클 8.1.7

여기서는 Dell PowerEdge 2500 이 사용되었다: 두 개의 1G 펜티엄 III 프로세서의(256 kB 케이스) 듀얼 프로세서 보드, 2GB PC 133ECC SDRAM, 128MB 의 PERC/3 DC PCI RAID 컨트롤러 그리고 EIDE DVD-ROM 드라이브. RAID 컨트롤러에 두 개의 18GB 하드 디스크와(미러 된) 4 개의 36GB 하드 디스크를(RAID 레벨 5) 붙인다.

### 10.7.5 FreeBSD 설치

첫째 FreeBSD 를 설치해야 된다. FreeBSD 를 설치하는 여러 가지 방법이 있다(FreeBSD 4.3 은 FTP 를 통해 그리고 FreeBSD 4.5 는 릴리즈 CD 에서 직접 설치하였다). 더 많은 정보는 2.13 장을 본다.

#### 10.7.5.1 디스크 레이아웃

단순히 같은 디스크 레이아웃을 SAP R/3 46B 와 SAP R/3 46C SR2 설치에 사용했다. 다른 하드웨어에(dev/da 와 /dev/amr 을 각각 사용했기 때문에 AMI MegaRAID 를 사용한다면

---

/dev/da0s1a 대신 /dev/amr0s1a 를 볼수 있다) 설치했기 때문에 장치 이름만 변경했다:

파일 시스템	크기(1k-블록)	크기(GB)	마운트 포인트
/dev/da0s1a	1.016.303	1	/
/dev/da0s1b		6	swap
/dev/da0s1e	2.032.623	2	/var
/dev/da0s1f	8.205.339	8	/usr
/dev/da1s1e	45.734.361	45	/compat/linux/oracle
/dev/da1s1f	2.032.623	2	/compat/linux/sapmnt
/dev/da1s1g	2.032.623	2	/compat/linux/usr/sap

미리 Mylex 또는 PERC/3 RAID 소프트웨어로 두 개의 논리 드라이브를 설정하고 초기화한다. 이 소프트웨어는 BIOS 가 부팅하는 동안 시작된다.

SAP 는 **오라클** 서브 디렉터리에 나누어서 마운트 하도록 제안하지만 우리는 단순히 실제 서브 디렉터리에 생성하기로 결정했기 때문에 이 디스크 레이아웃은 SAP 권장 사항과 약간 다르다.

### 10.7.5.2 make world 와 새로운 커널

마지막 -STABLE 소스를 다운로드 한다. 커널 설정 파일을 설정한 후 world 와 사용자 커널로 다시 빌드한다. SAP R/3 와 **오라클**에 필요한 커널 매개 변수를 포함한다.

## 10.7.6 리눅스 환경 설치

### 10.7.6.1 리눅스 기본 시스템 설치

첫째 linux\_base 포트 설치가 필요하다(root 에서):

```
# cd /usr/ports/emulators/linux_base
# make install distclean
```

---

### 10.7.6.2 리눅스 개발 환경 설치

10.6 장에 따라 FreeBSD 에 오라클을 설치하려면 리눅스 개발환경이 필요하다:

```
# cd /usr/ports/devel/linux_devtools
# make install distclean
```

리눅스 개발 환경은 SAP R/3 46B IDES 설치를 위해서만 필요하다. FreeBSD 시스템에 오라클 DB 가 다시 링크되지 않는다면 필요없다. 이 환경은 리눅스 시스템에서 오라클 타볼(tarball)을 사용하는 경우다.

### 10.7.6.3 필요한 RPM 설치

R3SETUO 프로그램을 시작하려면 PAM(리눅스 인증 모듈) 지원이 필요하다. FreeBSD 4.3-STABLE 에 처음 SAP 를 설치하는 동안 우리는 필요한 모든 패키지와 PAM 설치를 시도했지만 결국 동작하는 PAM 패키지를 강제로 설치하였다. SAP R/3 4.6C SR2 에 우리는 동작하는 PAM RPM(레드햇 패키지 매니저)을 강제로 설치하였기 때문에 의존되는 패키지가 필요 없는 것으로 보인다:

```
# rpm -i --ignoreos --nodeps --root /compat/linux --dbpath /var/lib/rpm W
pam-0.68-7.i386.rpm
```

오라클 8.0.5 에 인텔리전트 에이전트를 실행하기 위해 우리는 RedHat Tcl 패키지 tcl-8.0.5-30.i386.rpm 도(설치하지 않으면 오라클을 설치하는 동안 링크가 동작하지 않는다) 설치했다. 오라클 링크에 관한 몇가지 다른 문제가 있지만 그것은 FreeBSD 가 아닌 리눅스와 오라클 문제다.

### 10.7.6.4 몇 가지 추가적인 힌트

/etc/fstab 에 linprocfs 를 추가하는 것도 좋은 아이디어다. 더 많은 정보는 linprocfs(5) 매뉴얼 페이지를 본다. 다른 매개변수 설정은 /etc/sysctl.conf 파일에 kern.fallback\_elf\_brand=3 을 설정한다.

---

## 10.7.7 SAP R/3 환경 생성

### 10.7.7.1 필요한 파일시스템과 마운트 포인트 생성

단순히 설치하는 다음 파일시스템을 생성하면 된다:

마운트 포인트	GB 크기
/compat/linux/oracle	45 GB
/compat/linux/sapmnt	2 GB
/compat/linux/usr/sap	2 GB

몇 가지 링크생성도 필요하다. 링크를 생성하지 않으면 **SAP** 인스톨러는 생성된 링크를 체크하도록 요청한다:

```
# ln -s /compat/linux/oracle /oracle
# ln -s /compat/linux/sapmnt /sapmnt
# ln -s /compat/linux/usr/sap /usr/sap
```

다음은 링크를 생성하지 않아서 설치 중 발생할 수 있는 에러메시지다(시스템 PRD 와 **SAP R/3 4.6C SR2** 설치):

```
INFO 2002-03-19 16:45:36 R3LINKS_IND_IND SyLinkCreate:200
    Checking existence of symbolic link /usr/sap/PRD/SYS/exe/dbg to
    /sapmnt/PRD/exe. Creating if it does not exist...

WARNING 2002-03-19 16:45:36 R3LINKS_IND_IND SyLinkCreate:400
    Link /usr/sap/PRD/SYS/exe/dbg exists but it points to file
    /compat/linux/sapmnt/PRD/exe instead of /sapmnt/PRD/exe. The
    program cannot go on as long as this link exists at this
    location. Move the link to another location.

ERROR 2002-03-19 16:45:36 R3LINKS_IND_IND Ins_SetupLinks:0
    can not setup link '/usr/sap/PRD/SYS/exe/dbg' with content
    '/sapmnt/PRD/exe'
```

---

### 10.7.7.2 유저와 디렉터리 생성

SAP R/3 는 두 개의 유저와 3 개의 그룹이 필요하다. 유저이름은 3 개의 문자로 이루어진 SAP 시스템 ID 에(SID) 의존된다. 이들 SID 중 몇 개는 SAP 에 의해 예약되어있다(예를 들어 SAP 와 NIX. 전체 리스트는 SAP 문서를 본다). 시스템을 업무에 사용할 수 있도록 IDES 설치에는 *IDS* 를 그리고 4.6C SR2 설치에는 *PRD* 를 사용한다. 그래서 우리는 다음과 같은 그룹을 갖게 되었다(그룹 ID 는 다를 것이다. 이들은 실제 설치에 사용한 값이다):

그룹 ID	그룹 이름	설명
100	dba	데이터베이스 관리자
101	sapsys	SAP 시스템
102	oper	데이터베이스 운영자

기본 오라클 설치에 dba 그룹만 사용했다. oper 그룹으로도 dba 그룹을 사용했다(더 많은 정보는 오라클과 SAP 문서를 본다)

그리고 다음 유저들도 필요하다:

유저 ID	유저 이름	일반 이름	그룹	추가적인 그룹	설명
1000	idsadm/prdadm	sidadm	sapsys	oper	SAP 관리자
1002	oraids/oraprd	Orasid	dba	oper	오라클 관리자

SAP 관리자"에 필요한 다음 엔트리를 adduser(8)로 추가한다:

```
Name: sidadm
Password: *****
Fullname: SAP Administrator SID
Uid: 1000
Gid: 101 (sapsys)
Class:
Groups: sapsys dba
HOME: /home/sidadm
Shell: bash (/compat/linux/bin/bash)
```

---

그리고 "오라클 관리자"에는 다음 엔트리가 필요하다:

```
Name: orasid
Password: *****
Fullname: Oracle Administrator SID
Uid: 1002
Gid: 100 (dba)
Class:
Groups: dba
HOME: /oracle/sid
Shell: bash (/compat/linux/bin/bash)
```

dba 와 oper 두 그룹을 사용한다면 그룹 oper 도 포함해야 된다.

### 10.7.7.3 디렉터리 생성

이들 디렉터리는 보통 분리된 파일시스템에 생성한다. 우리는 간단히 같은 RAID 5 에 디렉터리를 생성하기로 했다.

첫째 몇몇 디렉터리의 소유권과 권한을 설정한다(root 유저에서):

```
# chmod 775 /oracle
# chmod 777 /sapmnt
# chown root:dba /oracle
# chown sidadm:sapsys /compat/linux/usr/sap
# chmod 775 /compat/linux/usr/sap
```

두 번째로 유저 *orasid*로 디렉터리를 생성한다. 이들 디렉터리는 모두 */oracle/SID*의 서브디렉터리가 된다:

```
# su - orasid
# cd /oracle/SID
# mkdir mirrlogA mirrlogB origlogA origlogB
# mkdir sapdata1 sapdata2 sapdata3 sapdata4 sapdata5 sapdata6
# mkdir saparch sapreorg
```

---

```
# exit
```

오라클 8.1.7 을 설치할 때 몇 개의 추가적인 디렉터리가 필요하다:

```
# su - orasid
# cd /oracle
# mkdir 805_32
# mkdir client stage
# mkdir client/80x_32
# mkdir stage/817_32
# cd /oracle/SID
# mkdir 817_32
```

**Note:** client/80x\_32 디렉터리는 정확하게 이 이름으로 사용된다. *x*를 다른 번호로 바꾸지 않는다.

세 번째 단계에서 유저 *sidadm* 으로 디렉터리를 생성한다:

```
# su - sidadm
# cd /usr/sap
# mkdir SID
# mkdir trans
# exit
```

#### 10.7.7.4 /etc/services 엔트리

SAP R/3 는 FreeBSD 에 설치할 때 정확하게 설정되지 않는 /etc/services 파일에 몇 가지 엔트리가 필요하다. 다음 엔트리를 추가한다(최소한 인스턴트 번호와 일치하는 엔트리가 필요하다. 이 경우 00을 추가하지만 *dp*, *gw*, *sp*와 *ms*에 00에서 99까지 모든 엔트리를 추가해도 문제는 없다). SAProuter 를 사용하거나 SAP OSS 에 접근해야 된다면 타겟 시스템에서 포트 3299 가 보통 SAProuter 에 사용되기 때문에 99도 필요하다:

sapdp00	3200/tcp # SAP Dispatcher.	3200 + Instance-Number
sapgw00	3300/tcp # SAP Gateway.	3300 + Instance-Number
sapsp00	3400/tcp #	3400 + Instance-Number



---

sapms00	3500/tcp #	3500 + Instance-Number
sapmsS/D	3600/tcp # SAP Message Server.	3600 + Instance-Number
sapgw00s	4800/tcp # SAP Secure Gateway	4800 + Instance-Number

### 10.7.7.5 필요한 지역

SAP 는 RedHat 의 기본 설치에서 설정되지 않는 최소한 두 개의 지역이 필요하다.

SAP 는 FTP 서버에서 필요한 RPM 다운로드를 제공한다(OSS 에 접근할 수 있는 소비자만 다운로드할 수 있는). 필요한 RPM 리스트는 노트 0171356 을 본다.

또한 적당한 링크를 생성하는 것도 가능하지만 업무용 시스템에는 권장하지 않는다(IDES 시스템과 문제없이 작동하지만). 다음 지역 설정이 필요하다:

```
de_DE.ISO-8859-1
en_US.ISO-8859-1
```

따라서 다음과 같이 링크를 생성한다:

```
# cd /compat/linux/usr/share/locale
# ln -s de_DE de_DE.ISO-8859-1
# ln -s en_US en_US.ISO-8859-1
```

이들이 없다면 설치할 때 몇 가지 문제가 발생한다. 계속해서 무시한다면(파일 CENTRDB.R3S 에서 OK가 아닌 상태를 선택하여) 추가적인 노력 없이 SAP 시스템에 로그인하는 것은 불가능 하다.

### 10.7.7.6 커널 튜닝

SAP R/3 시스템은 수 많은 리소스가 필요하다. 그래서 커널 설정파일에 다음과 같은 매개변수를 추가했다:

```
# Set these for memory pigs (SAP and Oracle):
options MAXDSIZ="(1024*1024*1024)"
options DFLDSIZ="(1024*1024*1024)"
```

---

```

# System V options needed.
options SYSVSHM #SYSV-style shared memory
options SHMMAXPGS=262144 #max amount of shared mem. pages
#options SHMMAXPGS=393216 #use this for the 46C inst.parameters
options SHMMNI=256 #max number of shared memory ident if.
options SHMSEG=100 #max shared mem.segs per process
options SYSVMSG #SYSV-style message queues
options MSGSEG=32767 #max num. of mes.segments in system
options MSGSSZ=32 #size of msg-seg. MUST be power of 2
options MSGMNB=65535 #max char. per message queue
options MSGTQL=2046 #max amount of msgs in system
options SYSVSEM #SYSV-style semaphores
options SEMMNU=256 #number of semaphore UNDO structures
options SEMMNS=1024 #number of semaphores in system
options SEMMNI=520 #number of semaphore identifiers
options SEMUME=100      #number of UNDO keys

```

SAP 의 문서에 최소값이 지정되어있다. 다만 리눅스를 위한 설명이 없기 때문에 더 많은 정보는 HP-UX 섹션(32-bit)을 본다. 4.6C SR2 를 설치하려는 시스템에 더 많은 메인 메모리가 있고 공유 세그먼트는 클수록 SAP 와 오라클에 좋지 때문에 거대한 양의 공유 메모리 페이지를 선택한다.

**Note:** *MAXDSIZ* 와 *DFLDSLZ* 을 최대 1GB 로 설정하고 i386 에 FreeBSD 4.5 를 기본적으로 설치한다. 그렇지 않으면 “ORA-27102: out of memory” 와 “Linux Error: 12: Cannot allocate memory” 같은 에러가 계속 발생한다.

## 10.7.8 SAP R/3 설치

### 10.7.8.1 SAP CDROM 준비

설치하는 동안 마운트와 언 마운트해야되는 여러 장의 CDROM 이 있다. 충분한 CDROM 드라이브가 있다면 모두 마운트한다. 우리는 CDROM 내용을 적당한 디렉터리에 복사했다:

```
/oracle/SID/sapreorg/cd-name
```

---

4.6B/IDES 를 설치할 때 *cd-name*은 KERNEL, RDBMS, EXPORT1, EXPORT2, EXPORT3, EXPORT4, EXPORT5 와 EXPORT6 그리고 4.6C SR2 를 설치할 때 *cd-name*은 KERNEL, RDBMS, DISK1, DISK2, DISK3, DISK4 와 LANG 중 하나다. CD 로 마운트된 모든 파일이름은 대문자를 사용해야 되고 그렇지 않으면 마운트에 *-g* 옵션을 사용한다. 다음 명령을 사용해 보자:

```
# mount_cd9660 -g /dev/cd0a /mnt
# cp -R /mnt/* /oracle/SID/sapreorg/cd-name
# umount /mnt
```

### 10.7.8.2 설치 스크립트 실행

첫째 install 디렉터리를 준비해야 된다:

```
# cd /oracle/SID/sapreorg
# mkdir install
# cd install
```

그리고 관련된 모든 파일을 install 디렉터리로 복사하는 설치스크립트를 시작한다:

```
# /oracle/SID/sapreorg/KERNEL/UNIX/INSTTOOL.SH
```

IDES 설치(4.6B) SAP R/3 데모 시스템에 완벽하게 튜닝 되어서 3 장의 EXPORT CD 대신 6 장이 있다. 여기서 IDES 중앙 인스턴스가 아닌 설치템플릿 CENTRDB.R3S 는 표준 중앙 인스턴스(R/3 와 데이터베이스) 위한 것이므로 EXPORT1 에서 정확한 CENTRDB.R3S 를 복사해야 되고 그렇지 않으면 R3SETUP 은 오직 3 개의 EXPORT CD 만 요청한다.

새로운 **SAP 4.6C SR2** 릴리즈는 4 장의 EXPORT CD 로 되어있다. 설치단계를 제어하는 매개변수 파일은 CENTRAL.R3S 다. 반대로 이전 릴리즈에는 데이터베이스가 있는 중앙 인스턴스 또는 데이터베이스가 없는 중앙 인스턴스를 위해 나누어진 설치템플릿이 없다. **SAP** 는 데이터베이스 설치에 분리된 템플릿을 사용한다. 그러나 나중에 설치를 다시 하겠다면 원본파일로 다시 시작하면 된다.

설치 중이나 설치 후에 **SAP** 는 전체 도메인네임이 아닌 컴퓨터 이름만 돌려받기 위해

hostname 이 필요하다. 그래서 호스트네임을 적절히 설정하거나 *orasid*와 *sidadm*에(또는 최소한 설치단계는 root 로 수행해야 되기 때문에 root 에) `alias hostname='hostname -s'`로 엘리어스를 설정한다. SAP 를 설치하는 동안 두 유저에게 설치된 .profile 와 .login 파일을 수정하는 것도 가능하다.

### 10.7.8.3 R3SETUP 4.6B 시작

LD\_LIBRARY\_PATH 를 정확히 설정한다:

```
# export LD_LIBRARY_PATH=/oracle/IDS/lib:/sapmnt/IDS/exe:/oracle/805_32/lib
```

설치 디렉터리에서 root 로 R3SETUP 를 시작한다:

```
# cd /oracle/IDS/sapreorg/install
# ./R3SETUP -f CENTRDB.R3S
```

그러면 이 스크립트가 몇 가지 질문을 한다(기본값은 가로로 묶여있고 그 다음은 실제로 입력한 값이다):

질문	기본값	입력
Enter SAP System ID	[C11]	IDSEnter
Enter SAP Instance Number	[00]	Enter
Enter SAPMOUNT Directory	[/sapmnt]	Enter
Enter name of SAP central host	[troubadix.domain.d e]	Enter
Enter name of SAP db host	[troubadix]	Enter
Select character set	[1] (WE8DEC)	Enter
Enter Oracle server version (1) Oracle 8.0.5, (2) Oracle 8.0.6, (3) Oracle 8.1.5, (4) Oracle 8.1.6		1Enter
Extract Oracle Client archive	[1] (Yes, extract)	Enter
Enter path to KERNEL CD	[/sapcd]	/oracle/IDS/sapreorg/KER NEL
Enter path to RDBMS CD	[/sapcd]	/oracle/IDS/sapreorg/RD BMS

Enter path to EXPORT1 CD	[/sapcd]	/oracle/IDS/sapreorg/EXPORT1
Directory to copy EXPORT1 CD	[/oracle/IDS/sapreorg/CD4_DIR]	<b>Enter</b>
Enter path to EXPORT2 CD	[/sapcd]	/oracle/IDS/sapreorg/EXPORT2
Directory to copy EXPORT2 CD	[/oracle/IDS/sapreorg/CD5_DIR]	<b>Enter</b>
Enter path to EXPORT3 CD	[/sapcd]	/oracle/IDS/sapreorg/EXPORT3
Directory to copy EXPORT3 CD	[/oracle/IDS/sapreorg/CD6_DIR]	<b>Enter</b>
Enter path to EXPORT4 CD	[/sapcd]	/oracle/IDS/sapreorg/EXPORT4
Directory to copy EXPORT4 CD	[/oracle/IDS/sapreorg/CD7_DIR]	<b>Enter</b>
Enter path to EXPORT5 CD	[/sapcd]	/oracle/IDS/sapreorg/EXPORT5
Directory to copy EXPORT5 CD	[/oracle/IDS/sapreorg/CD8_DIR]	<b>Enter</b>
Enter path to EXPORT6 CD	[/sapcd]	/oracle/IDS/sapreorg/EXPORT6
Directory to copy EXPORT6 CD	[/oracle/IDS/sapreorg/CD9_DIR]	<b>Enter</b>
Enter amount of RAM for SAP + DB		850 <b>Enter</b> (in Megabytes)
Service Entry Message Server	[3600]	<b>Enter</b>
Enter Group-ID of sapsys	[101]	<b>Enter</b>
Enter Group-ID of oper	[102]	<b>Enter</b>
Enter Group-ID of dba	[100]	<b>Enter</b>
Enter User-ID of sidadm	[1000]	<b>Enter</b>
Enter User-ID of orasid	[1002]	<b>Enter</b>
Number of parallel procs	[2]	<b>Enter</b>

CD 를 다른 위치에 복사해 두지 않았다면 **SAP** 인스톨러는 필요한 CD 를 찾지 못해서(CD 에서 LABEL.ASC 파일로 확인되는) CD 를 넣고 마운트 후 확인하거나 마운트 경로를 입력하라고 요청한다.

CENTRDB.R3S 는 에러가 없을 것이다. 우리의 경우 EXPORT4 CD 를 다시 요구했지만 정확한 키를(6\_LOCATION 그리고 7\_LOCATION 등) 입력했기 때문에 정확한 값으로 계속 진행할 수 있었다.

아래에서 언급된 몇 가지 문제와 별개로 오라클 데이터베이스 소프트웨어가 설치되는 곳까지 진행한다.

#### 10.7.8.4 R3SETUP 4.6C SR2 시작

LD\_LIBRARY\_PATH 를 정확히 설정한다. 이것은 오라클 8.0.5 로 4.6B 를 설치하는 것과 다른 값이다:

```
# export LD_LIBRARY_PATH=/sapmnt/PRD/exe:/oracle/PRD/817_32/lib
```

설치 디렉터리에서 root 로 R3SETUP 시작한다:

```
# cd /oracle/PRD/sapreorg/install
# ./R3SETUP -f CENTRAL.R3S
```

이 스크립트는 몇 가지 질문을 한다(기본값은 가로로 묶여있고 그 다음은 실제로 입력한 값이다):

질문	기본 값	입력
Enter SAP System ID	[C11]	PRDEnter
Enter SAP Instance Number	[00]	Enter
Enter SAPMOUNT Directory	[/sapmnt]	Enter
Enter name of SAP central host	[majestix]	Enter
Enter Database System ID	[PRD]	PRDEnter
Enter name of SAP db host	[majestix]	Enter
Select character set	[1] (WE8DEC)	Enter
Enter Oracle server version (2) Oracle 8.1.7		2Enter
Extract Oracle Client archive	[1] (Yes, extract)	Enter

Enter path to KERNEL CD	[/sapcd]	/oracle/PRD/sapreorg/KERNEL
Enter amount of RAM for SAP + DB	2044	1800Enter (in Megabytes)
Service Entry Message Server	[3600]	Enter
Enter Group-ID of sapsys	[100]	Enter
Enter Group-ID of oper	[101]	Enter
Enter Group-ID of dba	[102]	Enter
Enter User-ID of oraprd	[1002]	Enter
Enter User-ID of prdadm	[1000]	Enter
LDAP support		3Enter (no support)
Installation step completed	[1] (continue)	Enter
Choose installation service	[1] (DB inst,file)	Enter

설치하는 동안 OSUSERDBSID\_IND\_ORA(유저 *orasid* 생성) 그리고 OSUSERSIDADM\_IND\_ORA(유저 *sidadm* 생성) 구문에서 에러가난다면 유저를 생성한다.

아래에서 언급된 몇 가지 문제와 별개로 오라클 데이터베이스 소프트웨어가 설치되는 곳까지 진행한다.

## 10.7.9 오라클 8.0.5 설치

리눅스와 오라클에서 가능한 문제에 대해서는 적당한 SAP 노트와 오라클 readme 를 본다. 보통 호환되지 않는 라이브러리 문제는 없다.

오라클 설치에 대한 더 많은 정보는 오라클 설치 챕터를 참고한다.

### 10.7.9.1 오라클 8.0.5 와 orainst 설치

오라클 8.0.5 가 사용된다면 오라클 8.0.5 가 예전의 glibc(RedHat 6.0)와 링크된 것처럼 성공적으로 링크를 생성하기 위해 몇 가지 추가적인 라이브러리가 필요하지만 RedHat 6.1 은 새로운 glibc 를 사용하고 있다. 그래서 링크가 동작하도록 다음의 추가적인 패키지를 설치해야 된다:

```
compat-libs-5.2-2.i386.rpm
```

---

```
compat-glibc-5.2-2.0.7.2.i386.rpm
```

```
compat-egcs-5.2-1.0.3a.1.i386.rpm
```

```
compat-egcs-c++-5.2-1.0.3a.1.i386.rpm
```

```
compat-binutils-5.2-2.9.1.0.23.1.i386.rpm
```

더 많은 정보는 적당한 SAP 노트나 오라클 Readme 를 본다. 옵션이 없다면(설치하는 동안 우리는 체크할 수 있는 충분한 시간이 없었다) 원본 바이너리를 사용할 수 있거나 원래 RedHat 시스템에서 링크된 바이너리를 사용한다.

인텔리전트 에이전트를 컴파일 하려면 RedHat Tcl 패키지가 설치되어 있어야 한다. tcl-8.0.3-20.i386.rpm 이 없다면 RedHat 6.1 의 tcl-8.0.5-30.i386.rpm 같은 새로운 것도 사용할 수 있다.

다시 링크하는 부분에서 설치는 상당히 직관적이다:

```
# su - oraids
# export TERM=xterm
# export ORACLE_TERM=xterm
# export ORACLE_HOME=/oracle/IDS
# cd /ORACLE_HOME/orainst_sap
# ./orainst
```

리눅스에서 현재 이용할 수 없기 때문에 *Oracle On-Line Text Viewer* 를 제외하고 소프트웨어가 설치될 때까지 Enter 를 눌러 모두 확인한다. 그리고 오라클은 유효한 gcc, egcs 나 i386-redhat-linux-gcc 대신 gcc, egcs 또는 i386-redhat-linux-gcc 와 링크되어야 한다.

시간 때문에 우리는 **오라클 8.0.5 PreProduction** 릴리즈 바이너리를 사용하기로 했다. RDBMS CD 에서 버전을 받으려는 첫 번째 이후의 시도는 실패했고 그 당시 정확한 RPM 을 찾는 것은 우리가 많았다.



---

## 10.7.9.2 리눅스용(커널 2.0.33) 오라클 8.0.5 Pre-production

### 릴리즈 설치

이 설치 는 상당히 쉽다. CD 를 마운트하고 인스톨러를 시작한다. 오라클 홈 디렉터리의 위치를 물어보고 모든 바이너리를 홈 디렉터리에 복사한다. 이전에 설치한 RDBMS 가 남아있었지만 나중에 오라클 데이터베이스는 아무런 문제없이 시작할 수 있었다.

## 10.7.10 오라클 8.1.7 리눅스 타볼 설치

oracle81732.tgz 타볼을 가지고 리눅스 시스템의 설치 디렉터리 /oracle/SID/817\_32/에 타볼을 푼다.

### 10.7.11.1 SAP R/3 설치 계속

첫째로 유저 idsamd(sidadm)과 oraids(orasid)의 설정 환경을 체크한다. 이들은 모두 hostname 을 사용하는 .profile, .login 과 .cshrc 를 가지고 있다. 이 경우 파일 3 개의 모든 hostname 을 완벽하게 유효한 이름 hostname -s 로 변경해야 된다.

### 10.7.11.1 데이터베이스 로드

exit 를 선택하거나 안 하느냐에 따라 R3SETUP 를 재 시작 하거나 계속 진행할 수 있다. R3SETUP 는 테이블 스페이스를 생성하고 데이터와(46B IDES 는 EXPORT1 에서 EXPORT6 까지 그리고 46C 는 DISK1 에서 DISK4 까지) R3load 를 데이터베이스에 넣어서 로드한다.

데이터베이스 로드가 끝났을 때(몇 시간 정도가 필요할 것이다) 어떤 패스워드가 요구된다. 테스트 설치를 위해 유명한 기본 패스워드를 사용할 수 있다(보안 문제가 있다면 다른 것을 사용한다):

질문	입력
Enter Password for sapr3	sapEnter
Confirum Password for sapr3	sapEnter

Enter Password for sys	change_on_installEnter
Confirm Password for sys	change_on_installEnter
Enter Password for system	managerEnter
Confirm Password for system	managerEnter

여기서 우리가 4.6B 를 설치하는 동안 dipgntab 에서 몇 가지 문제가 있었다.

### 10.7.11.2 리스너

다음과 같이 유저 *orasisd* 로 오라클 리스너를 시작한다:

```
% umask 0; lsnrctl start
```

그렇지 않으면 소켓에 정확한 퍼미션이 할당되지 않기 때문에 ORA-12546 에러를 보게 된다. 자세한 사항은 SAP 노트 072984 를 본다.

### 10.7.11.3 MNLS 테이블 업데이트

Latin-1 이 아닌 언어를 SAP 시스템에 탑재하기를 원치 않는다면 여러 나라의 언어지원 테이블을 업데이트 한다. 이 사항은 SAP OSS 노트 15023 과 45619 에 설명되어 있다. 그렇지 않으면 SAP 를 설치하는 동안 이 질문을 건너뛰어도 된다.

**Note:** MNLS 가 필요 없다면 테이블 TCPDB 를 체크해야 되고 이것이 끝나지 않는다면 초기화 해야 된다. 더 많은 정보는 SAP 노트 0015023 과 0045619 를 본다.

## 10.7.12 설치 후 단계

### 10.7.12.1 SAP R/3 라이선스 키 요청

SAP R/3 라이선스 키를 요청해야 된다. 설치 중에 지정한 라이선스는 4 주 동안만 일시적으로 유효하므로 키를 요청해야 된다. 첫째로 하드웨어 키를 받는다. 유저

---

idsadm 으로 로그인해서 saplicense 를 실행한다:

```
# /sapmnt/IDS/exe/saplicense -get
```

saplicense 를 매개변수 없이 실행하며 옵션 리스트를 보여준다. 받은 라이선스 키에 따라 설치할 수 있다:

```
# /sapmnt/IDS/exe/saplicense -install
```

다음 값을 입력해야 된다:

SAP SYSTEM ID	= SID, 3 chars
CUSTOMER KEY	= hardware key, 11 chars
INSTALLATION NO	= installation, 10 digits
EXPIRATION DATE	= yyyyymmdd, usually "99991231"
LICENSE KEY	= license key, 24 chars

### 10.7.12.2 유저 생성

클라이언트 000 에서 유저를 생성한다(이 작업을 끝내기 위해 클라이언트 000 에서 몇 가지 작업이 필요하지만 유저 sap\*와 ddic 는 다르다). 유저 네임에 우리는 보통 wartung(또는 영어에서 service)를 선택했다. 필요한 프로파일은 sap\_new 와 sap\_all 이다. 기본 유저 패스워드의 추가적인 안전을 위해 모든 클라이언트를 변경해야 된다(이것은 유저 sap\*와 ddic 를 포함한다).

### 10.7.12.3 전송 시스템, 프로파일, 운용 모드 등 설정

ddic 및 sap\*와 다른 유저는 클라이언트 000 에서 최소한 다음과 같은 작업이 필요하다:

작업	트랜잭션
전송 시스템 설정, 예: Stand-Alone Transport Domain Entity	STMS
시스템을 위한 프로파일 생성/수정	RZ10
운영 모드와 인스턴스 유지	RZ04

---

이들과 다른 모든 설치 후 단계는 SAP 설치 가이드에 완벽하게 설명되어있다.

### 10.7.12.4 initsid.sap(initIDS.sap) 편집

파일 /oracle/IDS/dbs/initIDS.sap 는 SAP 백업 프로필을 가지고 있다. 여기서 사용되는 테잎 크기와 압축된 테잎 등은 정의가 필요하다. sapdba/brbackup 으로 실행되는 동안 정의하기 위해 우리는 다음 값을 변경하였다:

```
compress = hardware
archive_function = copy_delete_save
cpio_flags = "-ov --format=newc --block-size=128 --quiet"
cpio_in_flags = "-iuv --block-size=128 --quiet"
tape_size = 38000M
tape_address = /dev/nsa0
tape_address_rew = /dev/sa0
```

설명:

*compress*: 우리가 사용하는 테잎은 하드웨어 압축을 하는 HP DLT1 이다.

*archive\_function*: 이 옵션은 오라클 아카이브 로그를 저장하기 위한 기본동작을 정의한다: 새로운 로그파일은 테잎에 저장하고 이미 저장된 로그는 다시 한번 저장된 후 삭제된다. 데이터베이스를 복구해야될때 이 옵션은 아카이브 테잎 중 하나가 이상하더라도 수많은 문제를 방지한다.

*cpio\_flags*: 기본값은 블록크기를 5120 Bytes 로 설정하는 *-B* 를 사용한다. HP 는 DLT 테잎에 최소한 32K 블록크기를 권장하기 때문에 우리는 64K 로 설정하도록 *--block-size=128* 을 사용한다. inode 번호가 65535 보다 크기 때문에 *--format=newc* 가 필요하다. 마지막 옵션 *-quiet* 가 필요하고 사용하지 않는다면 brbackup 이 cpio 저장된 블록번호를 출력하자마자 요청할 것이다.

*cpio\_in\_flags*: 플래그는 테잎에서 데이터를 다시 로딩하기 위해 필요하다. 포맷은 자동으로 감지된다.

*tape\_size*: 이 옵션은 보통 저 용량 테잎에 사용된다. 보안적인 이유로(우리는 하드웨어

---

압축을 사용한다) 이 값은 실제 값보다 약간 낮다.

*tape\_address*: rewind 할 수 없는 장치는 cpio 로 사용한다.

*tape\_address\_rew*: rewind 할 수 있는 장치는 cpio 로 사용한다.

### 10.7.12.5 설치 후 설정 문제

다음 SAP 매개변수를 설치 후에 켜야 된다(예를 들어 IDES 46B 는 1GB 메모리):

이름	값
ztta/roll_extension	250000000
abap/heap_area_dia	300000000
abap/heap_area_nondia	400000000
em/initial_size_MB	256
em/blocksize_kB	1024
ipc/shm_psize_40	70000000

SAP Note 0013026:

이름	값
ztta/dynpro_area	2500000

SAP Note 0157246:

이름	값
rdisp/ROLL_MAXFS	16000
rdisp/PG_MAXFS	30000

**Note:** 위의 매개변수로 1GB 의 메모리를 가지고 있는 시스템은 다음과 비슷한 메모리 소비량을 볼수 있다:

Mem: 547M Active, 305M Inact, 109M Wired, 40M Cache, 112M Buf, 3492K Free

---

## 10.7.13 설치 중 문제

### 10.7.13.1 문제 해결 후 R3SETUP 재 시작

에러가 발생했다면 R3SETUP 을 정지한다. 정확한 로그파일을 보고 에러를 해결했다면 R3SETUP 다시 시작한다. 보통 마지막 단계에서 REPEAT 를 선택한 것에 대해 R3SETUP 가 문제를 제기한다.

R3SETUP 를 재 시작하려면 적당한 R3S 파일로 시작한다:

```
# ./R3SETUP -f CENTRDB.R3S
```

4.6B 는

```
# ./R3SETUP -f CENTRAL.R3S
```

4.6C 는 CENTRAL.R3S 나 DATABASE.R3S 에서 에러가 발생하더라도 문제가 없다.

**Note:** 어떤 단계에서 R3SETUP 는 데이터베이스와 **SAP** 프로세스가 운용 중 이라고 간주한다(이러한 단계가 이미 끝났기 때문에). 예를 들어 데이터베이스가 시작되지 않았다면 에러가 발생한다. 따라서 에러를 수정한 후 R3SETUP 를 다시 시작하기 전에 데이터베이스와 **SAP** 를 수동으로 시작해야 된다.

**오라클** 리스너도 정지되었다면(예를 들면 시스템 재 부팅이 필요했다면) 오라클 리스너도 다시 시작하는 것을 잊지 않는다(*orasid*에서 `umask 0; lsnrctl start`).

### 10.7.13.2 R3SETUP 동안 OSUSERSIDADM\_IND\_ORA

이 단계에서 R3SETUP 이 문제가 발생하면 R3SETUP 이 이때 사용하는 템플릿 파일을 수정한다(CENTRDB.R3S(4.6B), CENTRAL.R3S 또는 DATABASES.R3S(4.6C)).

[*OSUSERSIDADM\_IND\_ORA*] 위치의 내용이나 *STATUS=ERROR* 엔트리만 찾아서 다음 값으로 수정한다:

---

```
HOME=/home/sidadm (was empty)
STATUS=OK (had status ERROR)
```

그리고 R3SETUP 를 다시 시작한다.

### 10.7.13.3 R3SETUP 동안 OSUSERDBSID\_IND\_ORA

R3SETUP 은 이 단계에서도 문제가 발생할 수 있다. 여기 에러는 구문 OSUSERSIDADM\_IND\_ORA 과 비슷하다. 이때 R3SETUP 이 사용하는 템플릿 파일을 수정한다(CENTRDB.R3S(4.6B)또는 CENTRAL.R3S 또는 DATABASES.R3S(4.6C)). [OSUSERDBSID\_IND\_ORA] 위치의 내용이나 STATUS=ERROR 엔트리를 찾아서 다음 값으로 수정한다:

```
STATUS=OK
```

그리고 R3SETUP 를 다시 시작한다.

### 10.7.13.4 오라클을 설치하는 동안 “oraview.vrf FILE NOT

#### FOUND”

설치를 시작하기 전에 *Oracle On-Line Text Viewer* 선택을 해제하지 않았다. 리눅스용에서 현재 사용할 수 없더라도 이 옵션은 설치에서 표시되어있다. **오라클** 설치메뉴에서 이 제품 선택을 해지하고 설치를 계속한다.

### 10.7.13.5 R3SETUP, RFC 또는 SAPguil 를 시작하는 동안

#### “TEXTENV\_INVALID”

이 에러가 발생했다면 정확한 지역이 빠진 것이다. SAP 노트 0171356 에는 설치가 필요한 RPM 을 나열하고 있다(예: RedHat 6.1 의 saplocales-1.0-3, saposcheck-1.0-1.) 이 경우 관련된 모든 에러를 무시하고 R3SETUP 이 문제를 표시할 때마다 **ERROR** 을 **OK** 로 적당히

---

설정하고 R3SETUP 를 다시 시작한다. **SAP** 시스템은 정확하게 설정되지 않았기 때문에 시스템이 시작되었어도 SAPgui 로 시스템에 연결할 수 없다. 주어진 다음 메시지를 따르고 예전 리눅스 **SAPgui** 로 연결을 다시 시도한다:

```
Sat May 5 14:23:14 2001
*** ERROR => no valid userarea given [trgmsg0. 0401]
Sat May 5 14:23:22 2001
*** ERROR => ERROR NR 24 occurred [trgmsgi. 0410]
*** ERROR => Error when generating text environment. [trgmsgi. 0435]
*** ERROR => function failed [trgmsgi. 0447]
*** ERROR => no socket operation allowed [trxio.c 3363]
Speicherzugriffsfehler
```

이 동작은 **SAP R/3** 에 지역할당이 정확히 되지 않았고 적절한 설정도 필요하다(몇몇 데이터베이스 테이블에서 빠진 엔트리). **SAP** 와 연결하도록 DEFAULT.PFL 파일에(노트 0043288 을 본다) 다음 엔트리를 추가한다:

```
abap/set_etct_env_at_new_mode = 0
install/collate/active = 0
rscp/TCP0B = TCP0B
```

**SAP** 시스템을 다시 시작한다. 지역 지정과 언어 설정이 동작하지 않더라도 이제 시스템에 연결할 수 있다. 국가설정을(그리고 정확한 지역을 지원하는) 정확히 한후 이러한 엔트리를 DEFAULT.PFL 에서 삭제할 수 있고 **SAP** 시스템을 재 시작할 수 있다.

### 10.7.13.6 ORA-00001

위 에러는 FreeBSD 4.5 의 **오라클 8.1.7** 에서만 발생하는 에러다. **오라클** 데이터베이스를 적절히 초기화하지 못하고 충돌했기 때문에 시스템의 세마포어와 공유 메모리를 제거한다. 그 다음 데이터베이스를 시작하고 ORA-00001 을 입력한다.

ipcs -a 로 이들을 찾고 ipcrm 으로 삭제한다.



---

### 10.7.13.7 ORA-00445 (백그라운드 프로세스 PMON 이 시작되지 않았다)

이 에러는 **오라클 8.1.7** 에서 발생한다. 이 에러는 데이터베이스가 유저 `prdadm` 에서 `startsap` 스크립트로(예를 들면 `startsap_majestix_00`) 시작되었을 때 발생한다.

가능한 방법은 `svrmgrl` 대신 유저 `oraprd` 로 데이터베이스를 시작한다:

```
% svrmgrl
SVRMGR> connect internal;
SVRMGR> startup;
SVRMGR> exit
```

### 10.7.13.8 ORA-12546 (정확한 퍼미션으로 리스너 시작)

유저 `oraids` 에서 다음 명령으로 **오라클** 리스너를 시작한다:

```
# umask 0; lsnrctl start
```

그렇지 않으면 소켓에 정확한 퍼미션이 할당되지 않기 때문에 ORA-12546 에러를 보게 된다. SAP 노트 0072984 를 본다.

### 10.7.13.9 ORA-27102 (메모리 부족)

이 에러는 `MAXDSIZ`와 `DFLDSIZ` 값이 1GB 보다 큰 값을 사용하려고 할 때(1024x1024x1024) 발생한다. 추가적으로 우리는 "Linux Error 12: Cannot allocate memory"를 보았다.

### 10.7.13.10 R3SETUP 동안 [DIPGNTAB\_IND\_IND]

일반적으로 SAP 노트 0130581 을 본다 (R3SETUP 단계에서 `DIPGNTAB`이 끝난다). IDES-기능을 설치하는 동안 몇 가지 이유로 설치 단계가 적절한 **SAP** 시스템네임 "IDS"를

---

사용하지 못하고 빈 문자열 ""을 사용한다. 경로는 *S/D*를(여기서는 IDS) 동적으로 사용하여 생성되기 때문에 디렉터리 접근과 관련된 몇 가지 사소한 문제가 발생한다. 그래서 다음 경로 대신:

```
/usr/sap/IDS/SYS/...  
/usr/sap/IDS/DVMGS00
```

아래의 경로를 사용한다:

```
/usr/sap//SYS/...  
/usr/sap/D00
```

설치를 계속하기 위해 우리는 추가적인 디렉터리에 링크를 생성했다:

```
# pwd  
/compat/linux/usr/sap  
# ls -l  
total 4  
drwxr-xr-x 3 idsadm sapsys 512 May 5 11:20 D00  
drwxr-x--x 5 idsadm sapsys 512 May 5 11:35 IDS  
lrwxr-xr-x 1 root sapsys 7 May 5 11:35 SYS -> IDS/SYS  
drwxrwxr-x 2 idsadm sapsys 512 May 5 13:00 tmp  
drwxrwxr-x 11 idsadm sapsys 512 May 4 14:20 trans
```

또한 우리는 SAP 노트(0029227 과 0008401)에서 이 동작에 대한 설명을 발견했다. 이러한 문제를 **SAP 4.6C** 설치에서는 보지 못했다.

### 10.7.13.11 R3SETUP 동안 [RFCRSWBOINI\_IND\_IND]

**SAP 4.6C**를 설치하는 동안 발생한 이 에러는 이전에 설치하는 동안 발생한 다른 에러가 원인이 됐다. 이 경우 적절한 로그파일을 보고 정확한 실제문제를 찾는다.

로그파일을 확인한 후 이 에러가 발생하는 실제원인을 찾았다면 문제 단계의 *STATUS*를 *ERROR*에서 *OK*로(파일 CENTRDB.R3S) 설정할 수 있고 R3SETUP를 다시 시작한다. 설치를 끝낸 후 트랜잭션 SE38로부터 *RSWBOINS* 리포트를 실행해야 된다.

---

*RFCRSWBOINI*와 *RFCRADDBDIF* 구문에 대한 추가적인 정보는 SAP 노트 0162266 을 본다.

### 10.7.13.12 R3SETUP 동안 [RFCRADDBDIF\_IND\_IND]

역시 비슷한 방법을 적용한다: 로그 파일을 확인해서 이전 문제가 원인이 되지 않았는지 확인한다.

SAP 노트 0162266 에 적용된 것을 확인하였다면 문제단계의 *STATUS* 설정을 *ERROR*에서 *OK*(파일 *CENTRDB.R3S*)로 바꾸고 R3SETUP 를 다시 시작한다. 설치 후 트랜잭션 SE38 에서 RADDBDIF 리포트를 실행해야 된다.

### 10.7.13.13 sigaction sig31: 파일 크기제한 초과

이 에러는 SAP 프로세스를 *disp+work*로 시작하는 동안 발생한다. SAP 가 startsap 스크립트로 시작되었다면 서버 프로세스가 분리되어 다른 모든 SAP 프로세스의 시작을 방해한다. 이 결과로 문제가 생겨도 스크립트 자체가 에러를 보여주지 못한다.

SAP 프로세스가 정확히 시작되었는지 체크하기 위해 모든 오라클과 SAP 프로세스 상태를 보여주는 `ps ax | grep SID`로 살펴본다. SAP 시스템에 연결할 수 없고 어떤 프로세스를 잃어버렸다면 `/usr/sap/SID/DEVBMGSnr/work/`에서 찾을 수 있는 적절한 로그파일을 살펴본다. 이 파일은 `dev_ms`와 `dev_disp`에서 찾을 수 있다.

오라클과 SAP 가 사용하는 공유메모리의 양이 커널 설정파일에 정의한 양을 초과했다면 31 신호가 발생하기 때문에 더 큰 값으로 해결할 수 있다:

```
# larger value for 46C production systems:
options SHMMAXPGS=393216
# smaller value sufficient for 46B:
#options SHMMAXPGS=262144
```

### 10.7.13.14 saposcol 시작 실패

---

프로그램 `saposcol` 과 몇 개의 문제가(버전 4.6D) 있다. **SAP** 시스템은 시스템성능에 관련된 데이터를 수정하기 위해 `saposcol` 을 사용한다. 이 프로그램을 **SAP** 시스템에 사용할 필요가 없기 때문에 이 문제는 사소한 문제가 될 수 있다. 이전 버전(4.6B)은 동작하지만 모든 데이터를(예를 들면 CPU 사용량과 같은 많은 콜은 단지 0 을 되돌려준다) 수집하지는 않는다.

## 10.8 발전된 주제

어떻게 리눅스 바이너리 호환성이 작동하는지 궁금하다면 이 섹션에 원하는 설명이 있다.

### 10.8.1 어떻게 동작하는가?

FreeBSD 는 “execution class loader”라는 추상적 개념을 가지고 있다. 이것이 `execve(2)` 시스템 콜의 발단이 됐다.

셸 인터프리터나 셸 스크립트를 실행하기 위해 FreeBSD 는 `#!` 로더에 의해 저장된 싱글 로더 대신에 로더 리스트를 가지고 있다. 역사적으로 유닉스 플랫폼의 유일한 로더는 시스템이 알고 있는 바이너리인지 확인해서 바이너리라면 바이너리 로더를 실행하는 매직번호(magic number)를(보통 파일의 처음 4 또는 8 바이트) 확인한다. 시스템의 바이너리 종류가 아니라면 `execve(2)`가 실패를 되돌려주고 셸이 셸 스크립트처럼 실행한다.

이 가정은 현재의 셸이 무엇이든 간에 기본이다.

나중에 첫 번째 2 캐릭터를 확인하도록 `sh(1)`에 `hack` 이 만들어졌고 2 캐릭터가 `:/bin` 이면 `csh(1)` 셸을 대신 실행한다(SCO 가 처음으로 이 `hack` 을 만들었다고 생각된다).

이제 FreeBSD 가 하는 것은 로더 리스트를 거치는 것이다. 이것은 인터프리터에 대해 알고 있는 일반적인 `#!` 로더인데 이들은 공백들에 의해 구분되는 문자들과 `/bin/sh` 에 저장된 내용들로 구성된다.

리눅스 ABI 를 지원하기 위해 FreeBSD 는 ELF 바이너리로(FreeBSD, Solaris™, 리눅스 등 ELF 이미지 타입을 가지고 있는 다른 OS 는 차이가 없다) 매직번호를 확인한다.

SVR4/Solaris ELF 바이너리에 없는 ELF 로더는 ELF 이미지 섹션에 설명되어있는 특화된 *brand* 를 찾는다.

---

리눅스 바이너리가 동작하도록 이들을 `brandelf(1)`에서 Linux 형식으로 brand 해야 된다:

# `brandelf -t Linux file`

brand 가 끝나면 ELF 로더는 파일에서 *Linux brand* 를 찾을 수 있다.

ELF 로더가 *Linux brand* 를 확인하면 로더는 *proc* 구조의 포인터를 변경한다. 모든 시스템 콜은 이 포인터로 인덱스 된다(전통적인 유닉스 시스템에서 이것은 시스템 콜을 포함하고 있는 *sysent[]* 구조배열이다). 게다가 트램폴린(trampoline) 코드 신호의 trap vector 를 특별히 제어하는 플래그가 프로세스에 할당되고 리눅스 커널 모듈이 제어하는 몇 가지를 수정했다.

리눅스 시스템 콜 벡터는 주소가 커널 모듈에 있는 *sysent[]* 엔트리 리스트를 가지고 있다.

리눅스 바이너리가 시스템 콜을 호출할 때 내장된 코드는 시스템 콜 기능 포인터가 *proc* 구조를 참조하지 않게 하고 FreeBSD 가 아닌 리눅스 시스템 콜 엔트리 포인트를 갖게 한다.

게다가 리눅스 모드는 동적으로 *reroots* 를 검색한다; 이 결과로 파일시스템을 마운트할 때 *union* 옵션이 파일시스템 마운트에 사용된다(*unionfs* 파일시스템 타입이 아니다). 처음 */compat/linux/original-path* 디렉터리에서 파일을 검색하고 검색에 실패하면 */original-path* 디렉터리에서 검색은 끝난다. 이것은 다른 바이너리에 필요한 바이너리를 실행할 수 있다(예: 리눅스 툴 체인은 리눅스 ABI 지원에서 모두 실행할 수 있다). 적당한 리눅스 바이너리가 없다면 리눅스 바이너리는 FreeBSD 바이너리를 로드해서 실행할 수 있고, 리눅스에서 리눅스 바이너리가 실행되지 않았다고 경고를 보내지 못하도록 `uname(1)` 명령을 */compat/linux* 디렉터리에 둘 수 있다.

이 결과로 FreeBSD 커널에 리눅스 커널이 있다; 커널과 FreeBSD 시스템 콜 테이블 엔트리 그리고 리눅스 시스템 콜 테이블 엔트리가 제공하는 모든 서비스를 실행하는 파일시스템 운용, 가상 메모리 운용, 신호 전달, System V IPC 등 다양하고 근본적인 기능이 있다. 유일한 차이점은 FreeBSD 바이너리는 FreeBSD *glue* 기능을 가지고 있고 리눅스 바이너리는 리눅스 *glue* 기능을 가지고 있다(대부분의 예전 OS 는 자신만의 *glue* 기능을 가지고 있다.

**Glue:** 프로세스가 만드는 콜의 *proc* 구조에서 동적으로 초기화된 포인터를 끄는, 참조되지 않는 주소기능 대신 static global *sysent[]* 구조 배열의 주소기능

---

어느 것이 진짜 FreeBSD ABI 인지 문제되지 않는다. 기본적으로 유일한 차이점은(현재: 이것은 새로운 릴리즈에서 쉽게 변경할 수 있고 아마도 다음 버전부터 가능할 것이다) FreeBSD *glue* 기능은 정적으로 커널에 링크되어있고 리눅스 *glue* 기능도 정적으로 링크 시키거나 커널 모듈을 통해 접근할 수 있다.

그러나 이 기능은 사실 에뮬레이션이 아닌가? 이것은 에뮬레이션이 아니고 ABI 기능이다. 에뮬레이터는(또는 시뮬레이션) 관련이 없다.

그렇다면 왜 가끔 “리눅스 에뮬레이션” 이라고 하는가? 이 용어가 FreeBSD 의 이미지를 버렸다. 그 당시 이 기능을 표현할 수 있는 적절한 단어가 없었다; 코드를 컴파일 하지 않거나 모듈을 로드하지 않는다면 FreeBSD 가 리눅스 바이너리를 실행하는 것은 거짓이어서 로드 되는 것을 설명하기 위한 단어가 필요했고 따라서 “리눅스 에뮬레이터”가 탄생했다.

---

## III 시스템 관리

FreeBSD 핸드북의 나머지 장에서는 FreeBSD 시스템관리에 초점을 맞추고 있다. 각 장은 각 장을 읽고서 배울 수 있는 것과 내용에 대한 자세한 설명으로 시작한다.

이러한 장들은 필요할 때 읽을 수 있도록 디자인 되었다. 특별한 순서로 읽을 필요는 없고 FreeBSD 를 처음 사용하기 위해 모든 장을 읽을 필요도 없다.

## 11 장 설정과 튜닝

### 11.1 개요

FreeBSD 의 중요한 관점 중 하나는 시스템설정이다. 정확한 시스템 설정은 나중에 업그레이드할 때 두통거리를 방지한다. 이번 장에서는 FreeBSD 시스템을 튜닝할 수 있는 몇 가지 매개변수를 포함하여 FreeBSD 설정방법을 자세히 설명한다.

이번 장을 읽은 후 다음과 같은 사항을 알 수 있다:

- 파일 시스템과 스왑 파티션으로 효과적으로 작업할 수 있는 방법
- rc.conf 설정과 /usr/local/etc/rc.d 시작 시스템의 기본
- 네트워크 카드 설정과 테스트는 어떻게 하는가
- 네트워크 장치에서 가상호스트 설정은 어떻게 하는가
- /etc 에 있는 다양한 설정파일은 어떻게 사용하는가
- sysctl 변수를 사용하여 FreeBSD 를 어떻게 튜닝하는가
- 디스크 성능튜닝과 커널 제한을 어떻게 수정하는가

---

이번 장을 읽기 전에 다음 사항을 알고 있어야 한다:

- 유닉스와 FreeBSD 기본(3 장).
- 커널의 설정/컴파일의 기본(8 장).

## 11.2 초기 설정

### 11.2.1 파티션 레이아웃

#### 11.2.1.1 기본 파티션

disklabel(8)이나 sysinstall(8)로 파일시스템 레이아웃을 지정할 때 하드 드라이브는 바깥쪽 트랙에서 안쪽트랙으로 더 빠르게 데이터를 전송한다. 그래서 더 작고 사용량이 많은 파일 시스템은 드라이브 바깥쪽으로 가까워야 하지만 /usr 처럼 커다란 파티션은 안쪽으로 향해야 된다. 작은 순서대로 파티션을 만드는 것도 좋은 생각이다: root, swap, /var, /usr.

```
FreeBSD Disklabel Editor
Disk: ad1          Partition name: ad1s1    Free: 0 blocks (0MB)
Part  Mount        Size  Newfs  Part  Mount
----  -
ad1s1e /              500MB UFS    N
ad1s1b swap           512MB SWAP
ad1s1a /tmp           500MB UFS+S N
ad1s1f /var          2000MB UFS+S N
ad1s1g /usr          5000MB UFS+S N
ad1s1h /home        11027MB UFS+S N
```

/var 크기는 머신의 사용량을 반영한다. /var 은 메일박스, 로그파일 그리고 프린터 스펠로 사용된다. 메일박스과 로그파일은 사용자 수와 로그파일 보관주기에 따라 예상치 못하게 커질 수 있다. 대부분의 유저는 GByte 용량이 필요하지 않겠지만 /var/tmp 는 패키지를 포함할 수 있도록 충분한 크기를 할당해야 된다.

/usr 파티션은 port(7) 컬렉션과(권장) 소스 코드(선택적)처럼 시스템 지원에 필요한 수많은 파일을 가지고 있다. 두 가지는 모두 설치할 때 선택사항이다. 이 파티션에 최소한 2GB 이상을 권장한다.

파티션 크기를 선택할 때 필요한 공간을 생각한다. 다른 프로그램을 사용하는 동안 파티션



---

공간이 부족하면 문제가 발생된다.

**Note:** 어떤 유저는 `sysinstall(8)`의 *Auto-defaults*가 `/var`과 `/` 파티션을 필요한 공간보다 보통 적게 할당하는 것을 알게 된다. 파티션은 충분히 할당해야 된다.

```
FreeBSD Disklabel Editor
Disk: ad1 Partition name: adis1 Free: 0 blocks
Part      Mount      Size  Newfs  Part      Mount
-----
adis1a    /           128MB UFS     Y
adis1b    swap        1014MB SWAP
adis1e    /var        256MB  UFS+S  Y
adis1f    /tmp        256MB  UFS+S  Y
adis1g    /usr        17885MB UFS+S  Y
```

### 11.2.1.2 스왑 파티션

스왑 파티션을 할당하는 가장 좋은 방법은 시스템 메모리 크기의(RAM) 대략 두 배를 할당한다. 예를 들어 머신이 128MByte의 메모리를 가지고 있다면 스왑 파일은 256MByte가 되어야 한다. 메모리가 적은 시스템은 더 많은 스왑으로 좋은 성능을 발휘한다. 256MByte 이하의 스왑은 권장하지 않고 메모리 확장을 고려한다. 커널의 VM 페이징 알고리즘은 스왑 파티션이 최소한 메모리 크기의 두배가 되었을 때 최고의 성능을 발휘하도록 튜닝 되어있다. 너무 작은 스왑 설정은 가상메모리 페이지 스캐닝코드에 비효율적이고 더 많은 메모리를 추가한다면 나중에 문제가 될 것이다.

여러 개의 SCSI 디스크가(또는 서로 다른 컨트롤러에 여러 개의 IDE 디스크를 사용중인)있는 대형 시스템에는 스왑을 각 드라이브 별로 할당하는 것을 권장한다(4개 드라이브 이상). 이 스왑 파티션은 대략 같은 크기가 되어야 한다. 커널이 임의적인 크기를 제어할 수 있지만 내부 데이터구조 크기는 가장 큰 스왑 파티션의 4배다. 거의 같은 크기의 스왑 파티션을 생성하면 커널이 스왑 공간을 각 디스크에 최적으로 스트라이프(stripe) 한다. 스왑을 모두 사용하지 못하더라도 스왑 크기는 커야 된다. 스왑 크기가 크면 강제로 재 부팅되기 전에 잘못된 프로그램을 복구하기가 쉬울 것이다.

### 11.2.1.3 왜 파티션인가?

어떤 유저들은 한 개의 대형 파티션이 좋다고 생각하지만 이것이 문제가 되는 몇 가지 이유가 있다. 첫째 각 파티션은 운용상 다른 기능을 수행하기 때문에 파티션을 나누면 파일시스템을 적절히 튜닝할 수 있다. 예를 들면 `root`와 `/usr` 파티션에는 쓰기가 거의 없고 주로 읽

---

기만 한다. 반면에 읽기와 쓰기가 /var 과 /var/tmp 에서 많이 발생할 수 있다.

시스템을 적절히 파티션하면 쓰기가 많은 더 작은 파티션의 조각난 파일들이 대부분 읽기만 하는 파티션으로 퍼지지 않게 된다. 쓰기가 많은 파티션을 디스크 끝에 가깝게 유지하면 쓰기가 자주 발생하는 파티션에서 I/O 성능을 향상시킬 것이다. 큰 파티션에 I/O 성능이 필요하겠지만 이 들을 디스크 끝 쪽으로 더 이동시키면 /var 을 끝으로 너무 이동시키기 때문에 성능향상을 기대하기 어렵다. 마지막은 안전과 관련이 있다. 더 작고 대부분 읽기만 하는 정돈된 root 파티션은 갑작스런 시스템 장애에도 문제가 발생할 확률이 더 적다.

## 11.3 핵심 설정

시스템 설정정보의 중요한 위치는 /etc/rc.conf 다. 이 파일은 주로 시스템이 시작될 때 시스템 설정에 사용되는 전체적인 설정정보를 가지고 있다. 각 rc\* 파일들의 이름은 파일정보를 의미한다.

/etc/defaults/rc.conf 의 기본설정을 무시하려면 rc.conf 파일에 엔트리를 만들어야 한다. 이 파일에는 예제가 아닌 실제 기본값을 가지고 있기 때문에 /etc 로 복사하지 않는다. 전체 시스템의 특성변경은 rc.conf 파일을 사용한다.

관리자의 업무를 간소화시키기 위해 시스템의 특정설정에서 광범위한 설정으로 여러 가지 방법이 여러 사이트에 적용될 것이다. 권장되는 접근법은 사이트에 널리 퍼진 설정을 /etc/rc.conf.site 같은 파일로 바꾸고 이 파일을 시스템 특성정보만 가지고 있는 /etc/rc.conf 에 포함시킨다.

rc.conf 는 sh(1)가 읽기 때문에 설정을 변경하는 방법은 쉽다. 예를 들면 다음과 같다.

- rc.conf:

```
. rc.conf.site
hostname="node15.example.com"
network_interfaces="fxp0 lo0"
ifconfig_fxp0="inet 10.1.1.1"
```

- rc.conf.site:

---

```
defaultrouter="10.1.1.254"
saver="daemon"
blanktime="100"
```

rc.conf.site 파일을 rsync 같은 프로그램을 사용해서 모든 시스템으로 분배할 수 있지만 rc.conf 파일은 유일해야 된다.

sysinstall(8)이나 make world 를 사용한 시스템업그레이드는 rc.conf 파일을 덮어쓰지 않기 때문에 시스템 설정정보를 잃어버리지는 않는다.

## 11.4 어플리케이션 설정

일반적으로 설치된 어플리케이션은 구문에 맞는 각자의 설정파일을 가지고 있다. 이런 파일은 기본시스템에 나누어있기 때문에 패키지 관리 툴로 쉽게 설치해서 관리할 수 있는 이점을 가지고 있다.

보통 이런 파일은 /usr/local/etc 에 설치되어있다. 어플리케이션이 수많은 설정파일을 가지고 있는 경우 서브 디렉터리에 이들 파일을 가지고 있다.

일반적으로 포트나 패키지로 설치하면 샘플 설정파일도 설치된다. 이들 샘플파일은 보통 .default 가 이름 뒤에 붙는다. 어플리케이션의 설정파일이 없다면 .default 파일을 복사해서 만든다.

예를 들면 /usr/local/etc/apache 디렉터리의 내용을 보면 다음과 같은 파일들이 있다:

-rw-r--r--	1	root	wheel	2184	May 20	1998	access.conf
-rw-r--r--	1	root	wheel	2184	May 20	1998	access.conf.default
-rw-r--r--	1	root	wheel	9555	May 20	1998	httpd.conf
-rw-r--r--	1	root	wheel	9555	May 20	1998	httpd.conf.default
-rw-r--r--	1	root	wheel	12205	May 20	1998	magic
-rw-r--r--	1	root	wheel	12205	May 20	1998	magic.default
-rw-r--r--	1	root	wheel	2700	May 20	1998	mime.types
-rw-r--r--	1	root	wheel	2700	May 20	1998	mime.types.default

```
-rw-r--r-- 1 root wheel 7980 May 20 1998 srm.conf
-rw-r--r-- 1 root wheel 7933 May 20 1998 srm.conf.default
```

파일크기로 srm.conf 파일만 변경됐음을 볼 수 있다. 나중에 **Apache** 포트 업그레이드는 변경된 이 파일을 덮어쓰지 않는다.

## 11.5 서비스 시작

시스템이 다양한 서비스를 제공하도록 일반적인 기능들을 지원한다. 이들은 여러 가지 방식으로 시작될 것이고 각자 장점을 가지고 있다.

포트나 패키지 컬렉션으로 설치된 소프트웨어는 보통 시스템이 시작될 때 *start* 인자로 시작되고 시스템이 셧다운 될 때 *stop* 인자로 호출되는 스크립트를 /usr/local/etc/rc.d 에 설치한다. 이 방법은 root 로 실행해야 되는 다양한 서비스를 시스템이 실행할 때 권장된다. 이런 스크립트는 패키지를 설치할 때 등록되고 패키지를 제거할 때 삭제된다.

/usr/local/etc/rc.d 의 일반적인 시작 스크립트는 다음과 비슷하다:

```
#!/bin/sh
echo -n ' FooBar'

case "$1" in
start)
    /usr/local/bin/foobar
    ;;
stop)
    kill -9 `cat /var/run/foobar.pid`
    ;;
*)
    echo "Usage: `basename $0` {start|stop}" >&2
    exit 64
    ;;
esac
```

---

```
exit 0
```

FreeBSD 시작스크립트는 root 로 실행해야 되는 *sh* 확장자를 가진 스크립트를 /usr/local/etc/rc.d 에서 찾는다. 찾은 스크립트는 목적에 맞게 시작할 때 *start*를 섰다운 할 때 *stop* 옵션을 호출한다. 그래서 시스템이 시작하는 적당한 시간에 위의 샘플스크립트가 실행되기를 원하면 /usr/local/etc/rc.d 에 FooBar.sh 라는 파일에 저장하고 실행가능 하도록 만든다. 다음과 같이 chmod(1)로 쉘 스크립트를 실행할 수 있도록 만들 수 있다:

```
# chmod 755 FooBar.sh
```

어떤 서비스는 적당한 포트에 서비스요청이 들어왔을 때 inetd(8)가 실행한다. 이 방법은 메일을 읽는 서버에(POP 과 IMAP 등) 일반적이다. 이런 서비스는 /etc/inetd.conf 파일을 수정해서 서비스할 수 있다. 이 파일 편집에 대한 자세한 사항은 inetd(8)를 본다.

어떤 추가적인 시스템 서비스는 /etc/rc.conf 를 변경하여 실행할 수 없을 것이다. 이들은 전통적으로 이 서비스를 호출하는 명령을 /etc/rc.local 에 두면 된다. FreeBSD 3.1 에는 기본 /etc/rc.local 파일이 없기 때문에 관리자가 이 파일을 생성했다면 사용할 수 있지만 별로 권장하지 않는다. rc.local 은 일반적으로 서비스를 시작하는 가장 좋은 위치다. 서비스를 시작하기 위해 더 좋은 장소가 있다면 그 위치를 이용해도 된다.

**Note:** /etc/rc.conf 에 어떤 명령도 두지 않는다. 데몬을 시작하거나 부팅할 때 특정 명령을 실행하려면 /usr/local/etc/rc.d 에 스크립트를 저장한다.

시스템서비스를 시작하기 위해 cron(8) 데몬을 사용할 수도 있다. cron(8)은 프로세스를 crontab 소유자의 권한으로 실행할 수 있기 때문에 root 가 아닌 일반 유저로 서비스를 실행하고 관리할 수 있는 많은 장점이 있다.

cron(8)은 특정시간을 지정하는 대신 사용할 수 있는 *@reboot* 옵션을 지원하는 특별한 기능을 제공한다. 이 방법으로 시스템이 부팅하고 cron(8)이 시작되자마자 작업을 실행한다.

## 11.6 cron 유틸리티 설정

FreeBSD 에서 가장 유용한 유틸리티 중 하나는 cron(8)이다. cron 유틸리티는 백그라운드에서 실행되고 /etc/crontab 파일을 계속해서 체크한다. cron 유틸리티는 새로운 crontab 파

---

일을 검색하기 위해 /var/cron/tabs 디렉터리도 체크한다. 이러한 crontab 파일은 cron 이 실행해야 되는 시간과 명령정보를 저장한다.

/etc/crontab 파일을 살펴보자:

```
# /etc/crontab - root's crontab for FreeBSD
#
# $FreeBSD: src/etc/crontab,v 1.32 2002/11/22 16:13:39 tom Exp $
# ❶
#
SHELL=/bin/sh
PATH=/etc:/bin:/sbin:/usr/bin:/usr/sbin ❷
HOME=/var/log
#
#
#minute hour    mday    month    wday    who command ❸
#
#
*/5 * * * * root    /usr/libexec/atrun ❹
```

- ❶ 대부분의 FreeBSD 설정파일처럼 #문자는 주석문을 의미한다. 주석문은 파일에서 무엇을 왜 수행하는지 기억하게 한다. 주석문은 명령이나 명령의 일 부분으로 해석해야 되는 라인과 같이 두지 않는다; 주석문은 항상 새로운 라인에 둔다. 공란은 무시된다.
- ❷ 환경이 먼저 정의되어 있어야 한다. equals(=) 문자는 이 예제에서 사용된 SHELL, PATH 와 HOME 옵션처럼 환경설정을 정의하는데 사용된다. 쉘 라인이 생략되었다면 cron 은 기본적으로 sh 를 사용한다. PATH 변수가 생략되었다면 기본설정은 아무것도 없고 파일의 위치가 절대적으로 필요하다. HOME 이 생략되었다면 cron 은 유저의 홈 디렉터리를 사용한다.
- ❸ 이 라인은 총 7 개의 필드를 정의한다. 여기에 리스트된 값은 *minute*, *hour*, *mday*, *month*, *wday*, *who* 와 *command* 이고 이들은 대부분 의미를 암시할 수 있다. *minute* 는 명령이 실행되는 분을 의미한다. *hour* 는 *minute* 옵션과 비슷하게 단지 시간을, *mday* 는 날짜를, *month* 는 *hour* 와 *minute* 처럼 달을 명시한다. *Wday* 옵션은 주를 의미한다. 이런 모든 필드는 숫자로 된 값이어야 되고 24 시간으로

---

지정한다. 오직 `/etc/crontab` 파일에만 있는 `who` 필드는 특별하다. 이 필드는 어떤 사용자가 명령을 실행할지 명시한다. 사용자가 `crontab` 파일을 설치해도 이 옵션은 없다. 마지막으로 `command` 옵션이 나열되어 있다. 이것은 마지막 필드이므로 보통 실행될 명령이 명시된다.

- ④ 이 마지막 라인은 위에서 설명한 값을 정의한다. 우리는 여러 개의 \* 문자가 따라 오는 `*/5` 를 지정하였다. 이 \* 문자의 의미는 "처음부터 끝까지"이고 모든 시간으로 해석할 수 있다. 이 라인에 의해 `atrun` 명령은 root 권한으로 어느 날이나 달에 관계없이 매 5 분마다 실행하는 것으로 판단한다. `atrun` 에 관한 더 많은 정보는 `atrun(8)` 매뉴얼 페이지를 본다.

명령어에 여러 개의 옵션을 적용할 수 있다; 그러나 명령어를 여러 라인으로 확장할 경우 백슬러시 "W"를 사용하여 문자가 계속됨을 표시해야 된다.

한가지 차이점이 있지만 위 설정은 모든 `crontab` 파일의 기본설정이다. 유저 이름을 지정한 필드번호 6 은 시스템의 `/etc/crontab` 파일에만 있다. 이 필드는 각 유저의 `crontab` 파일에 서는 생략한다.

## 11.6.1 crontab 설치

새로 작성한 `crontab` 을 설치하려면 `crontab` 유틸리티를 사용한다. 아주 일반적인 사용법은 다음과 같다:

```
# crontab crontab
```

설치된 `crontab` 파일을 나열하는 옵션은 `crontab` 에 `-l` 을 붙이고 출력 값을 확인한다.

템플릿을 사용하지 않고 `crontab` 파일을 직접 작성하려는 유저는 `crontab -e` 옵션을 사용할 수 있다. 이 명령은 선택한 에디터로 빈 파일을 불러온다. 파일을 저장하면 이 파일은 `crontab` 명령으로 자동으로 설치된다.

## 11.7 FreeBSD 5.X 에서 rc 사용

---

FreeBSD 는 최근 시스템 초기화에 NetBSD rc.d 시스템을 통합했다. 사용자가 `/etc/rc.d` 디렉터리에 나열되어 있는 이들 파일에 관심을 가져야 한다. 이런 파일 중 대부분은 `start`, `stop` 과 `restart` 옵션으로 제어할 수 있는 기본적인 서비스를 위한 것이다. 예를 들면 `sshd(8)`은 다음 명령으로 다시 시작할 수 있다:

```
# /etc/rc.d/sshd restart
```

이 절차는 다른 서비스도 비슷하다. 물론 서비스는 보통 `rc.conf(5)`에 지정되어있기 때문에 자동으로 시작된다. 예를 들면 시작할 때 네트워크 주소변환 데몬을 사용하려면 단순히 다음 라인을 `/etc/rc.conf` 에 추가하면 된다:

```
natd_enable="YES"
```

`natd_enable="NO"` 라인이 있다면 단순히 `NO`를 `YES`로 변경한다. rc 스크립트는 아래에서 설명하듯이 의존되는 다른 서비스들도 다음에 재 부팅하는 동안 자동으로 로드한다.

rc.d 시스템은 주로 시스템이 시작되고/셴다운될 때 서비스도 시작/정지하도록 하였기 때문에 `/etc/rc.conf` 에 변수를 적절히 지정했다면 표준 `start`, `stop` 과 `restart` 옵션이 수행된다. 예를 들면 `/etc/rc.conf` 에 `sshd_enable` 이 `YES`로 설정되어 있다면 위의 `sshd restart` 명령만 동작한다. `/etc/rc.conf` 의 설정과 관계없이 서비스를 `start`, `stop` 또는 `restart`하려면 이 명령을 "force"로 지정해야 된다. 예를 들면 `/etc/rc.conf` 의 현재 설정에 상관없이 `sshd` 를 재 시작하려면 다음 명령을 실행한다:

```
# /etc/rc.d/sshd forcerestart
```

서비스가 `/etc/rc.conf` 에서 활성화되어 있다면 옵션 `rcvar`로 적절한 rc.d 스크립트를 실행해서 체크하기 쉽다. 그래서 관리자는 `sshd` 가 `/etc/rc.conf` 에 활성화되어 있는 것을 다음 명령을 실행해서 체크할 수 있다:

```
# /etc/rc.d/sshd rcvar
# sshd
$sshd_enable=YES
```

**Note:** 두 번째 라인은(`# sshd`) `sshd` 명령의 결과다; root 콘솔이 아니다.

서버가 실행 중인지 확인하려면 `status` 옵션을 사용한다. 예를 들면 `sshd` 가 실제로 시작되



---

있는지 확인하려면 다음 명령을 실행한다:

```
# /etc/rc.d/sshd status
sshd is running as pid 433.
```

서비스를 *reload* 하는 것도 가능하다. 이 방법은 서비스가 설정파일을 강제로 다시 읽도록 각 서비스에 신호를 보낸다. 보통 이 의미는 *SIGHUP* 신호를 서비스에 보내는 것이다.

**rcNG** 구조는 네트워크 서비스만을 위한 것이 아니고 시스템 초기화의 대부분에도 사용된다. 예를 들면 *bgfsck* 파일을 생각해 본다. 이 스크립트가 실행될 때 다음과 같은 메시지를 출력한다:

```
Starting background file system checks in 60 seconds.
```

그래서 이 파일은 시스템을 초기화하는 동안만 백그라운드 파일시스템 체크에 사용된다.

다양한 시스템 서비스는 완벽한 기능을 위해 다른 서비스와 의존된다. 예를 들면 NIS와 다른 RPC 기반 서비스는 *rpcbind* 서비스가 시작될 때까지 시작하지 못한다. 이 문제를 해결하는 의존관계와 다른 메타데이터에 관한 정보는 각 스크립트의 최상단에 주석문으로 설명되어 있다. *rcorder(8)* 스크립트는 시스템을 초기화하는 동안 어떤 시스템 서비스를 의존성에 안전하게 호출해야 하는지 순서를 결정하도록 주석문을 해석하는데 사용된다. 다음 단어가 각 시작파일의 최상단에 포함되어 있을 것이다:

- *PROVIDE*: 서비스 이름
- *REQUIRE*: 이 기능에 필요한 서비스 이름. 이 파일은 지정된 서비스 *이후에* 실행된다.
- *BEFORE*: 이 서비스와 의존되는 서비스 리스트. 이 파일은 지정된 서비스보다 *이전에* 실행된다.
- *KEYWORD*: FreeBSD 나 NetBSD. 이것은 \*BSD 의존성에 사용된다.

이 방법으로 관리자는 다른 유닉스 운영체제처럼 런레벨(runlevels) 때문에 혼동하지 않고 시스템서비스를 쉽게 제어할 수 있다.

---

FreeBSD 5.X 의 rc.d 시스템에 관한 추가적인 정보는 rc(8)과 rc.subr(8) 매뉴얼 페이지에서 찾을 수 있다.

## 11.8 네트워크 카드 설정

요즘 우리는 네트워크가 안 되는 컴퓨터를 상상할 수 없다. 네트워크 카드를 추가하고 설정하는 것은 FreeBSD 관리자에게 일반적인 업무다.

### 11.8.1 정확한 드라이버 설정

시작하기 전에 카드모델, 사용하고 있는 칩과 PCI 또는 ISA 카드인지 알고 있어야 된다. FreeBSD 는 광범위하고 다양한 PCI 와 ISA 카드를 지원한다. 카드가 지원되는지 가지고 있는 릴리즈의 하드웨어 호환리스트를 체크한다.

카드가 지원되면 적당한 드라이버를 선택해야 된다. `usr/src/sys/i386/conf/LINT` 는 지원되는 칩셋/카드에 대한 몇 가지 정보로 네트워크 카드 드라이버 리스트를 보여준다. 드라이버가 정확한지 의심된다면 드라이버 매뉴얼 페이지를 읽는다. 매뉴얼 페이지는 지원되는 하드웨어와 문제가 발생할 가능성이 있는 정보를 준다.

일반적인 카드를 가지고 있다면 드라이버를 찾기 위해 많은 시간을 투자할 필요가 없다. 일반적인 네트워크 카드 드라이버는 GENERIC 커널에 있기 때문에 부팅할 때 다음과 같이 카드를 보여준다:

```
dc0: <82c169 PNIC 10/100BaseTX> port 0xa000-0xa0ff mem 0xd3800000-0xd38000ff irq 15 at device 11.0 on pci0
dc0: Ethernet address: 00:a0:cc:da:da:da
miibus0: <MII bus> on dc0
ukphy0: <Generic IEEE 802.3u media interface> on miibus0
ukphy0: 10baseT, 10baseT-FDX, 100baseTX, 100baseTX-FDX, auto
dc1: <82c169 PNIC 10/100BaseTX> port 0x9800-0x98ff mem 0xd3000000-0xd30000ff irq 11 at device 12.0 on pci0
dc1: Ethernet address: 00:a0:cc:da:da:db
miibus1: <MII bus> on dc1
```

```
ukphy1: <Generic IEEE 802.3u media interface> on miibus1
ukphy1: 10baseT, 10baseT-FDX, 100baseTX, 100baseTX-FDX, auto
```

이 예제에서 우리는 dc(4) 드라이버를 사용하는 두 개의 카드가 시스템에 있는 것을 볼 수 있다

네트워크 카드를 사용하려면 적당한 드라이버를 로드해야 된다. 두 가지 방법 중 한 가지로 로드할 수 있다. 가장 쉬운 방법은 kldload(8)로 네트워크 카드 커널 모듈을 로드하는 것이다. 모듈은 모든 네트워크 드라이버에(예를 들어 ISA 카드와 ed(4)드라이버를 사용하는 카드) 사용할 수 없다. 또 다른 방법은 가지고 있는 카드를 지원하도록 커널에 정적으로 컴파일 한다. 커널 설정파일에 추가할 것을 체크하기 위해 /usr/src/sys/i386/conf/LINT 와 드라이버 매뉴얼 페이지를 참고한다. 커널 재 컴파일에 대한 더 많은 정보는 8 장을 본다. 부팅 할 때 카드가 커널에(GENERIC) 검색되었다면 새로운 커널을 빌드 할 필요는 없다.

## 11.8.2 네트워크 카드 설정

정확한 네트워크 카드 드라이버가 로드되었으면 카드를 설정해야 된다. 다른 것과 마찬가지로 설치할 때 네트워크 카드도 **sysinstall** 을 통해 설정하였다.

시스템의 네트워크 인터페이스 설정을 확인하려면 다음 명령을 입력한다:

```
% ifconfig
① dc0: flags=8843<①UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
    ②inet 192.168.1.3 ③netmask 0xfffff00 ④broadcast 192.168.1.255
    ⑤ether 00:a0:cc:da:da:da
    ⑥media: Ethernet autoselect (100baseTX <full-duplex>)
    ⑦status: active
② dc1: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
    inet 10.0.0.1 netmask 0xfffff00 broadcast 10.0.0.255
    ether 00:a0:cc:da:da:db
    media: Ethernet 10baseT/UTP
    status: no carrier
③ lp0: flags=8810<POINTOPOINT,SIMPLEX,MULTICAST> mtu 1500
④ lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
    inet 127.0.0.1 netmask 0xff000000
```

---

```
⑤ tun0: flags=8010<POINTOPOINT,MULTICAST> mtu 1500
```

**Note:** 예전 FreeBSD 버전은 `ifconfig(8)`에 `-a` 옵션이 필수였기 때문에 `ifconfig(8)`의 구문에 관한 더 자세한 설명은 매뉴얼 페이지를 참고한다. IPv6 의(*inet6* 등) 정확한 내용은 이 예제에서 생략한다.

이 예제에서 다음 장치가 표시되었다:

- ① dc0: 첫 번째 이더넷 인터페이스
- ② dc1: 두 번째 이더넷 인터페이스
- ③ lp0: 패러럴 포트 인터페이스
- ④ lo0: 루프백 장치
- ⑤ tun0: **ppp** 가 사용하는 터널 장치

FreeBSD 는 네트워크 카드 이름에 커널이 부팅할 때 검색된 순서를 드라이버 이름에 추가한다. 예를 들면 `sis2` 는 `sis(4)` 드라이버를 사용하는 시스템의 3 번째 네트워크 카드다.

이 예제에서 `dc0` 장치는 `up` 상태고 정상적으로 작동하고 있다. 키의 의미는 다음과 같다.

- ❶ `UP`은 카드가 설정되었고 준비되었다는 의미다.
- ❷ 카드는 인터넷(*inet*) 주소를(여기서는 192.168.1.3) 가지고 있다.
- ❸ 유효한 서브 넷 마스크를(*netmask*; 0xfffff00 는 255.255.255.0 와 같다) 가지고 있다.
- ❹ 유효한 브로드캐스트 주소를 가지고 있다(이 경우 192.168.1.255).
- ❺ 카드의(ether) MAC 주소는 00:a0:cc:da:da:da 다.
- ❻ 물리적인 미디어 선택은 자동선택 모드다(*media: Ethernet autoselect*)

---

(*10baseTX <full-duplex>*)). dc1 은 *10baseT/UTP* 미디어로 동작하도록 설정되어있다. 드라이버에 사용할 수 있는 미디어타입에 대한 더 많은 정보는 매뉴얼 페이지를 참고한다.

- ⑦ 링크(*status*) 상태는 *active* 다. dc1 에서 우리는 *status: no carrier* 를 볼 수 있다. 이더넷 케이블이 카드에 연결되지 않았을 때 일반적인 상태다.

ifconfig(8) 출력이 다음과 비슷하다면 카드가 설정되지 않았음을 나타낸다.

```
dc0: flags=8843<BROADCAST,SIMPLEX,MULTICAST> mtu 1500
      ether 00:a0:cc:da:da:da
```

카드를 설정하려면 root 권한이 필요하다. 네트워크 카드는 ifconfig(8) 명령어 라인으로 설정할 수 있지만 시스템이 재 부팅할 때마다 다시 설정해야 된다. 파일 `/etc/rc.conf` 에 네트워크 카드설정을 추가한다.

좋아하는 에디터로 `/etc/rc.conf` 를 연다. 시스템에 있는 각 네트워크 카드라인을 추가해야 한다. 우리의 경우를 예로 들면 다음 라인을 추가했다:

```
ifconfig_dc0="inet 192.168.1.3 netmask 255.255.255.0"
ifconfig_dc1="inet 10.0.0.1 netmask 255.255.255.0 media 10baseT/UTP"
```

여러분의 카드에 맞는 장치로 dc0, dc1 등을 바꾸고 주소도 적절히 바꾼다. 사용할 수 있는 옵션에 대한 더 자세한 사항은 카드 드라이버와 ifconfig(8)을 읽고 `/etc/rc.conf` 구문에 대한 더 많은 정보는 `rc.conf(5)` 매뉴얼 페이지를 본다.

설치하는 동안 네트워크를 설정하였다면 네트워크 카드에 관한 몇 라인이 있을 것이다. 라인을 새로 추가하기 전에 `/etc/rc.conf` 를 다시 체크한다.

LAN 에 있는 여러 머신의 이름과 IP 주소 등 필요한 정보가 없다면 `/etc/hosts` 파일에 추가해야 된다. 더 많은 정보는 `hosts(5)`과 `/usr/share/examples/etc/hosts` 를 참고한다.

안전하게 FreeBSD 설치에서처럼 `sysinstall` 을 이용하여 네트워크 카드를 설정할 수 있다.

```
# /stand/sysinstall
```

Configure → NetWorkings → Interfaces 를 선택한다. Interfaces 에서 네트워크 카드를 선택 하면 FreeBSD 설치에서와 같은 화면이 나타난다. 관련된 필드를 입력한다.

```
Network Configuration
Host: test.example.com
Domain: example.com
IPv4 Gateway: 10.1.100.1
Name server: 10.1.100.1
Configuration for Interface fxp0
IPv4 Address: 10.1.100.245
Netmask: 255.255.255.0
Extra options to ifconfig:
[OK] [CANCEL]
```

### 11.8.3 테스트와 문제해결

/etc/rc.conf 의 필요한 내용을 변경하고 시스템을 재 부팅한다. 이 방법은 변경한 것을 인터페이스에 적용하고 설정에러 없이 시스템이 재 시작하는 것을 확인할 수 있다.

시스템이 재 부팅되면 네트워크 인터페이스를 테스트한다.

#### 11.8.3.1 이더넷 카드 테스트

이더넷 카드가 정확히 설정되었는지 확인하려면 두 가지를 체크한다. 첫째로 인터페이스에 핑을 보내고 LAN 의 다른 머신에 핑을 보낸다.

첫째 다음 명령으로 로컬 인터페이스를 테스트한다:

```
% ping -c5 192.168.1.3
PING 192.168.1.3 (192.168.1.3): 56 data bytes
64 bytes from 192.168.1.3: icmp_seq=0 ttl=64 time=0.082 ms
64 bytes from 192.168.1.3: icmp_seq=1 ttl=64 time=0.074 ms
64 bytes from 192.168.1.3: icmp_seq=2 ttl=64 time=0.076 ms
64 bytes from 192.168.1.3: icmp_seq=3 ttl=64 time=0.108 ms
```

---

```
64 bytes from 192.168.1.3: icmp_seq=4 ttl=64 time=0.076 ms

--- 192.168.1.3 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.074/0.083/0.108/0.013 ms
```

이제 LAN 의 다른 머신에 핑을 보낸다:

```
% ping -c5 192.168.1.2
PING 192.168.1.2 (192.168.1.2): 56 data bytes
64 bytes from 192.168.1.2: icmp_seq=0 ttl=64 time=0.726 ms
64 bytes from 192.168.1.2: icmp_seq=1 ttl=64 time=0.766 ms
64 bytes from 192.168.1.2: icmp_seq=2 ttl=64 time=0.700 ms
64 bytes from 192.168.1.2: icmp_seq=3 ttl=64 time=0.747 ms
64 bytes from 192.168.1.2: icmp_seq=4 ttl=64 time=0.704 ms

--- 192.168.1.2 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.700/0.729/0.766/0.025 ms
```

/etc/hosts 파일을 설정하였다면 192.168.1.2 대신 머신 이름을 사용할 수 있다.

### 11.8.3.2 문제 해결

하드웨어와 소프트웨어 설정문제 해결은 간단한 사항을 사전에 체크해서 골치 아픈 사항을 감소시킬 수 있다. 네트워크 케이블이 카드에 꽂혀 있는가? 네트워크 서비스 설정을 정확히 하였는가? 방화벽설정을 정확히 하였는가? 사용하고 있는 카드를 FreeBSD 가 지원하는가? 버그리포트를 보내기 전에 항상 하드웨어노트를 확인한다. FreeBSD 버전을 마지막 STABLE 버전으로 업데이트 한다. 메일링 리스트를 체크하거나 인터넷을 검색한다.

카드가 작동하지만 성능이 좋지 않다면 tuning(7) 매뉴얼 페이지를 읽어 볼 것을 권장한다. 또한 정확하지 않은 네트워크 설정이 속도를 감소시키기 때문에 네트워크 설정을 체크할 수 있다.

어떤 유저는 어떤 카드에서 일반적인 "device timeouts"을 한두 번 경험한다. 증상이 계속

---

발생하거나 귀찮게 한다면 다른 장치와 충돌하지 않는지 확인한다. 케이블 연결도 다시 체크한다. 다른 카드가 필요할 수도 있다.

어떤 때 유저는 몇 개의 "watchdog timeout" 에러를 보게 된다. 첫째 네트워크 케이블을 체크 한다. 대부분의 카드는 버스마스터를 지원하는 PCI 슬롯이 필요하다. 어떤 오래된 마더 보드는 오직 PCI 슬롯 한 개만 지원한다(보통 슬롯 0). 문제가 있는지 확인하려면 네트워크 카드와 마더보드 문서를 체크한다.

"No route to host" 메시지는 시스템이 목적지 호스트에 패킷을 라우트하지 못하면 발생한다. 이것은 기본 라우트가 없거나 케이블이 연결되지 않았을 때 발생한다. `netstat -rn` 의 결과를 확인하고 접근하려는 호스트에 맞는 라우트가 있는지 확인한다. 라우트가 없다면 24 장을 읽는다.

"ping: sendto: Permission denied" 에러 메시지는 종종 잘못된 방화벽설정으로 발생한다. `ipfw` 가 커널에서 활성화 되었지만 룰이 정의되지 않았다면 기본정책이 ping 응답을 포함한 모든 트래픽을 거부한다. 더 많은 정보는 14.8 장을 읽는다.

가끔 카드성능이 떨어지거나 수준 미달일 수 있다. 이 경우 `autoselect` 를 정확한 미디어로 선택하여 미디어 선택모드를 정확히 설정한다. 대부분의 하드웨어에서 발생하는 일이지만 누구나 이 문제를 해결할 수 있는 것은 아니다. 다시 모든 네트워크설정을 확인하고 `tuning(7)` 매뉴얼 페이지를 읽는다.

## 11.9 가상 호스트

FreeBSD 를 사용하는 매우 일반적인 방법은 서버 한대로 네트워크에 여러 서버처럼 보이게 하는 가상 사이트 호스팅이다. 여러 개의 네트워크 주소를 인터페이스 하나에 할당하면 된다.

주어진 네트워크에서 인터페이스는 한 개의 "실제" 주소를 가지고 있고 많은 수의 "alias" 주소를 가지고 있을 것이다. 이런 엘리어스는 보통 `/etc/rc.conf` 에 엘리어스 엔트리를 추가하면 된다.

인터페이스 `fxp0` 의 엘리어스 엔트리는 다음과 같다:



```
ifconfig_fxp0_alias0="inet xxx.xxx.xxx.xxx netmask xxx.xxx.xxx.xxx"
```

엘리어스 엔트리는 alias0 부터 시작해서 순서대로 높여간다(예를 들면 \_alias1, \_alias2 등으로). 설정 과정은 처음으로 빠진 번호에서 멈춘다.

엘리어스의 넷 마스크 계산은 중요하지만 다행히 아주 간단하다. 주어진 인터페이스에 정확히 네트워크의 넷 마스크를 표현하는 한 개의 주소가 있어야 된다. 이 네트워크에 할당된 모든 주소는 1개의 넷 마스크를 가져야 된다.

예를 들면 fxp0 인터페이스는 두 개의 네트워크에 연결되어있다. 10.1.1.0 네트워크에 넷 마스크 255.255.255.0 가 그리고 202.0.75.15 네트워크에 넷 마스크 255.255.255.240 가 할당되어 있다. 우리는 시스템에 10.1.1.1 부터 10.1.1.5 까지 그리고 202.0.75.17 에서 202.0.75.20 까지 IP 를 할당하려고 한다.

다음 엔트리로 이렇게 준비하기 위해 아답터를 정확히 설정한다:

```
ifconfig_fxp0="inet 10.1.1.1 netmask 255.255.255.0"
ifconfig_fxp0_alias0="inet 10.1.1.2 netmask 255.255.255.255"
ifconfig_fxp0_alias1="inet 10.1.1.3 netmask 255.255.255.255"
ifconfig_fxp0_alias2="inet 10.1.1.4 netmask 255.255.255.255"
ifconfig_fxp0_alias3="inet 10.1.1.5 netmask 255.255.255.255"
ifconfig_fxp0_alias4="inet 202.0.75.17 netmask 255.255.255.240"
ifconfig_fxp0_alias5="inet 202.0.75.18 netmask 255.255.255.255"
ifconfig_fxp0_alias6="inet 202.0.75.19 netmask 255.255.255.255"
ifconfig_fxp0_alias7="inet 202.0.75.20 netmask 255.255.255.255"
```

## 11.10 설정 파일

### 11.10.1 /etc 레이아웃

설정 정보를 가지고 있는 수 많은 디렉터리가 있다:

/etc	일반적인 시스템 설정정보; 이곳 데이터는 시스템과 연관된다.
------	-----------------------------------

/etc/defaults	시스템 설정파일의 기본버전
/etc/mail	추가적인 sendmail(8) 설정과 다른 MTA 설정파일
/etc/ppp	유저와 커널 ppp 프로그램 설정
/etc/namedb	named(8) 데이터를 위한 기본위치. 보통 named.conf 와 zone 파일이 이곳에 저장된다.
/usr/local/etc	설치된 어플리케이션의 설정파일. 어플리케이션마다 서브디렉터리를 가지고 있을 것이다.
/usr/local/etc/rc.d	설치된 어플리케이션의 start/stop 스크립트
/var/db	패키지 데이터베이스, 위치 데이터베이스처럼 자동으로 생성되고 시스템과 관련있는 데이터베이스 파일.

## 11.10.2 Hostnames

### 11.10.2.1 /etc/resolve.conf

/etc/resolve.conf 는 FreeBSD 의 리졸버가 인터넷 도메인 시스템에(DNS) 어떻게 접근하는지 보여준다.

resolve.conf 에 가장 보편적인 엔트리는 다음과 같다.

#### *nameserver*

리졸버가 질의할 네임서버 IP 주소. 서버는 최대 3 개의 나열된 서버 순서대로 질의를 받는다.

#### *search*

호스트 이름을 찾기 위해 검색하는 리스트. 이것은 보통 로컬 호스트의 도메인에 의해 결정된다.

#### *domain*

로컬 도메인 이름

전형적인 resolve.conf:

```
search example.com
```

---

```
nameserver 147.11.1.11
nameserver 147.11.100.30
```

**Note:** 오직 하나의 *search*와 *domain* 옵션만 사용된다.

DHCP 를 사용한다면 dhclient(8)는 보통 DHCP 서버에서 받은 정보로 resolv.conf 를 다시 작성한다.

### 11.10.2.2 /etc/hosts

/etc/hosts 는 옛날 인터넷을 회상하게 하는 간단한 텍스트 데이터베이스다. 이 파일은 이름을 IP 주소로 매핑해주는 DNS 및 NIS 와 결합하여 작동한다. LAN 으로 연결되어 있는 로컬 컴퓨터는 named(8) 서버를 설정하는 대신 아주 단순히 네임서버 목적으로 hosts 파일을 설정할 수 있다. 추가적으로 /etc/hosts 는 일반적으로 접근하는 이름을 외부에 질의할 필요 없이 인터넷 네임의 로컬 레코드를 제공하는데 사용할 수 있다.

```
# $FreeBSD$
#
# Host Database
# This file should contain the addresses and aliases
# for local hosts that share this file.
# In the presence of the domain name service or NIS, this file may
# not be consulted at all; see /etc/nsswitch.conf for the resolution order.
#
#
::1                localhost localhost.my.domain myname.my.domain
127.0.0.1          localhost localhost.my.domain myname.my.domain

#
# Imaginary network.
#10.0.0.2          myname.my.domain myname
#10.0.0.3          myfriend.my.domain myfriend
#
# According to RFC 1918, you can use the following IP networks for
# private nets which will never be connected to the Internet:
```

---

```
#
# 10.0.0.0 - 10.255.255.255
# 172.16.0.0 - 172.31.255.255
# 192.168.0.0 - 192.168.255.255
#
# In case you want to be able to connect to the Internet, you need
# real official assigned numbers. PLEASE PLEASE PLEASE do not try
# to invent your own network numbers but instead get one from your
# network provider (if any) or from the Internet Registry (ftp to
# rs.internic.net, directory `/templates').
#
```

/etc/hosts 는 간단한 포맷을 가지고 있다:

[Internet address] [official hostname] [alias1] [alias2] ...

예를 들면 다음 라인과 같다.

```
10.0.0.1 myRealHostname.example.com myRealHostname foobar1 foobar2
```

더 많은 정보는 hosts(5)를 참고한다.

## 11.10.3 log 파일 설정

### 11.10.3.1 syslog.conf

syslog.conf 는 syslogd(8) 프로그램의 설정파일이다. 특정 로그파일로 저장할 syslog 메시지 종류를 나타낸다.

```
# $FreeBSD$
#
# Spaces ARE valid field separators in this file. However,
# other *nix-like systems still insist on using tabs as field
# separators. If you are sharing this file between systems, you
```

```

#      may want to use only tabs as field separators here.
#      Consult the syslog.conf(5) manual page.
*.err;kern.debug;auth.notice;mail.crit      /dev/console
*.notice;kern.debug;lpr.info;mail.crit;news.err /var/log/messages
security.*                                   /var/log/security
mail.info                                    /var/log/maillog
lpr.info                                     /var/log/lpd-errs
cron.*                                       /var/log/cron
*.err                                        root
*.notice;news.err                           root
*.alert                                      root
*.emerg                                      *
# uncomment this to log all writes to /dev/console to /var/log/console.log
#console.info                                /var/log/console.log
# uncomment this to enable logging of all log messages to /var/log/all.log
#*. *                                         /var/log/all.log
# uncomment this to enable logging to a remote log host named loghost
#*. *                                         @loghost
# uncomment these if you're running inn
# news.crit                                   /var/log/news/news.crit
# news.err                                    /var/log/news/news.err
# news.notice                                 /var/log/news/news.notice
!startslip
*. *                                         /var/log/slip.log
!ppp
*. *                                         /var/log/ppp.log

```

더 많은 정보는 `syslog.conf(5)`를 참고한다.

### 11.10.3.2 newsyslog.conf

`newsyslog.conf`는 보통 `cron(8)`에 스케줄되어 실행되는 프로그램 `newsyslog(8)`의 설정파일이다. `newsyslog(8)`은 로그파일 보관이나 재 배열이 언제 필요한지 결정한다. `logfile`은 `logfile.0`으로 변경되고 `logfile.0`는 `logfile.1`로 계속해서 변경된다. 그렇지 않으면 `log` 파일은 `gzip(1)`포맷으로 보관되어 이름이 `logfile.0.gz`, `logfile.1.gz` 등으로 바뀐다.

---

newsyslog.conf 는 어떤 log 파일이 관리되고 얼마나 많은 파일을 보관하며 언제 log 파일들이 만들어 졌는지 보여준다. log 파일은 일정한 크기나 시간/날짜가 될 때까지 재 배열되거나 보관할 수 있다.

```
# configuration file for newsyslog
# $FreeBSD$
#
# filename          [owner:group]   mode count size when [ZB] [pid_file]
[sig_num]
/var/log/cron              600 3    100 *    Z
/var/log/amd.log           644 7    100 *    Z
/var/log/kerberos.log     644 7    100 *    Z
/var/log/lpd-errs         644 7    100 *    Z
/var/log/maillog          644 7    *    @T00 Z
/var/log/sendmail.st      644 10   *    168 B
/var/log/messages         644 5    100 *    Z
/var/log/all.log          600 7    *    @T00 Z
/var/log/slip.log         600 3    100 *    Z
/var/log/ppp.log          600 3    100 *    Z
/var/log/security         600 10   100 *    Z
/var/log/wtmp             644 3    *    @01T05 B
/var/log/daily.log        640 7    *    @T00 Z
/var/log/weekly.log       640 5    1    $W6D0 Z
/var/log/monthly.log      640 12   *    $M1D0 Z
/var/log/console.log      640 5    100 *    Z
```

더 많은 정보는 newsyslog(8) 매뉴얼 페이지를 참고한다.

## 11.10.4 sysctl.conf

sysctl.conf 는 rc.conf 와 거의 흡사하며 값은 *variable=value* 형식으로 설정한다. 지정된 값은 시스템이 멀티유저 모드에서 설정되고 모든 변수도 이 모드에서 설정할 수 있다.

샘플 sysctl.conf 는 치명적인 상태 로그를 끄고 실제로 FreeBSD 에서 실행하는 리눅스 프

---

로그를 로그로 지정하였다:

```
kern.logsigexit=0      # Do not log fatal signal exits (e.g. sig 11)
compat.linux.osname=FreeBSD
compat.linux.osrelease=4.3-STABLE
```

## 11.11 sysctl 튜닝

sysctl(8)은 사용중인 FreeBSD 시스템을 변경시킬 수 있는 인터페이스다. 이것은 TCP/IP 스택 옵션의 다양한 장점과 노련한 시스템관리자들이 성능을 엄청나게 향상시킬 수 있는 가상 메모리 시스템을 포함한다. 500 가지 이상의 시스템변수를 sysctl(8)로 읽고 설정할 수 있다.

sysctl(8) 서버는 시스템설정을 읽고 변경하는 핵심적인 두 가지 기능이 있다.

읽을 수 있는 모든 변수를 보려면:

```
% sysctl -a
```

예를 들어 특정변수 kern.maxproc 를 읽으려면 다음 명령을 사용한다:

```
% sysctl kern.maxproc
kern.maxproc: 1044
```

특정변수를 설정하려면 직관적으로 *variable=value* 구문을 사용한다:

```
# sysctl kern.maxfiles=5000
kern.maxfiles: 2088 -> 5000
```

sysctl 변수는 보통 문자열, 숫자 또는 불린(Booleans)값 중 하나로 설정한다(불린에서 1은 yes 를 0은 no 를 의미한다).

---

## 11.11.1 읽기 전용 sysctl(8)

어떤 경우 sysctl(8) 값을 읽기 전용으로 수정하는 것이 바람직할 것이다. 권장하지는 않지만 종종 불가피하다.

예를 들면 어떤 노트북 모델의 cardbus(4) 장치는 메모리 범위를 찾지 못하고 다음과 비슷한 에러메시지와 함께 실패한다:

```
cbb0: Could not map register memory  
  
device_probe_and_attach: cbb0 attach returned 12
```

위와 같은 경우 특정 기본 sysctl(8) 설정을 읽기 전용으로 수정 할 필요가 있다. 이러한 상황을 해결하기 위해 유저는 로컬 /boot/loader.conf.local 에 sysctl(8) "OLDs"를 넣을 수 있다. 기본 설정은 /boot/defaults/loader.conf 파일에 있다.

위에서 언급한 문제를 해결하려면 앞서 말한 파일에 *hw.pci.allow\_unsupported\_io\_range=1* 이라고 설정해야 된다. 이제 cardbus(4)는 정확하게 작동한다.

## 11.12 디스크 튜닝

### 1.12.1 Sysctl 변수

#### 11.12.1.1 *vfs.vmiodirenable*

*vfs.vmiodirenable sysctl* 변수는 0 (off) 또는 1 (on)로 설정되어 있을 것이다; 기본값은 1 이다. 이 변수는 시스템이 디렉터리를 어떻게 캐시하는지 제어한다. 대부분의 디렉터리는 작고 파일시스템에서는 조각(fragment(보통 1K) 하나를 그리고 버퍼 캐시에는 더 작은 조각을 (보통 512 bytes) 사용한다. 그러나 기본모드로 운용할 때 버퍼 캐시는 수많은 메모리를 가지고 있더라도 고정된 수의 디렉터리만 캐시한다. 이 sysctl 을 켜면 버퍼 캐시가 디렉터리를 캐시하기 위해 VM 페이지 캐시를 사용하고 모든 메모리를 디렉터리 캐시에 사용할 수 있다. 그러나 코어(core) 메모리에서 디렉터리 캐시로 사용되는 최소량은 512bytes 보다 큰



---

물리적인 페이지 크기다(일반적으로 4k). 수많은 파일을 제어하는 서비스를 운용한다면 이 옵션을 켜기를 권장한다. 이런 서비스는 웹 캐시, 대형 메일시스템과 뉴스시스템을 포함한다. 이 옵션을 켜면 메모리 낭비를 가져올 수 있지만 일반적으로 성능을 저하시키지 않는다. 그러나 테스트를 통해 메모리 낭비도 해결할 수 있다.

### 11.12.1.2 *vfs.write\_behind*

*vfs.write\_behind* sysctl 변수의 기본값은 1 (on)이다. 이 변수는 연속적인 대형 파일을 쓸 때 일반적으로 발생하는 전체 클러스터가 모이면 파일시스템에게 미디어 쓰기가 발생했음을 통보해 준다. 이 아이디어는 I/O 성능이 개선되지 않을 때 불필요한 버퍼로 버퍼 캐시가 소모되는 것을 방지한다. 그러나 이것은 프로세스를 지연시킬 수 있기 때문에 특정 상황에서 끄려고 할 수도 있다.

### 11.12.1.3 *vfs.hirunningspace*

*vfs.hirunningspace* sysctl 변수는 주어진 상황에서 얼마나 많은 쓰기 I/O 를 디스크 컨트롤러 시스템의 큐에 저장할지 결정한다. 기본값이 보통 적당하지만 많은 디스크를 가진 머신에서 4 나 5 메가바이트로 올리기를 원할 것이다. 너무 큰 값으로(엄청난 버퍼 캐시를 쓰는 발단이 된다) 설정하면 클러스터링 성능이 아주 떨어뜨릴 수 있다. 이렇게 큰 값으로 설정하지 않는다! 높은 쓰기 값으로 같은 시간에 읽기도 추가될 수 있다.

다양한 버퍼 캐시와 VM 페이지 캐시가 sysctl 과 연관되어있다. 이러한 값을 변경하지 않기를 권한다. FreeBSD 4.3 에서 VM 시스템은 자동으로 튜닝된 것이 성능에 아주 좋다.

### 11.12.1.4 *vm.swap\_idle\_enabled*

*vm.swap\_idle\_enabled* sysctl 변수는 많은 유저가 로그인해서 사용하는 시스템과 수많은 idle 프로세스를 가지고 있는 대형 멀티유저 시스템에 유용하다. 이런 시스템은 수많은 프로세스를 생성하여 여유 메모리보존을 저하시킨다. 이 기능을 켜고 *vm.swap\_idle\_threshold1* 과 *vm.swap\_idle\_threshold2* 로 swapout hysteresis 를(idle 초 동안) 튜닝하면 idle 프로세스와 연관된 메모리 페이지의 우선순위를 일반적인 페이지 알고리즘보다 더 빨리 낮출 수 있다. 따라서 페이지 아웃 데몬을 돕는다. 본질적으로 pre-page 메모리를 바로 교체하도록 해서 결과적으로 스왑과 디스크 대역폭을 더 많이 사용하기 때문에 필요하지 않다면 이 옵션

---

션을 켜지 않는다. 작은 시스템에서 이 옵션은 효과를 확신할 수 있지만 이미 적당한 페이지링을 하고있는 대형 시스템에서 이 옵션은 VM 시스템이 전체 프로세스로 진행되어 메모리가 쉽게 바닥나게 한다.

### 11.12.1.5 *hw.ata.wc*

FreeBSD 4.3에서는 IDE 쓰기 캐싱을 시험적으로 꺼두었다. 이 변수는 IDE 디스크의 쓰기 대역폭을 줄이지만 심각한 데이터일관성에 대한 논의가 하드 드라이브 벤더에 의해 제기되었기 때문에 필요성이 고려되었다. 문제는 IDE 드라이브 쓰기가 완료되었을 때 잘못된 결과를 통보하는 것이다. IDE 쓰기 캐시를 켜두면 IDE 하드 드라이브는 디스크에 쓰기를 못 할 뿐만 아니라 종종 디스크 부하가 많을 때 몇몇 블록의 쓰기를 무기한으로 지연시킨다. 시스템이 다운되거나 전원이 끊어지면 심각한 파일시스템 문제의 원인이 될 것이다. FreeBSD의 기본 정책은 안정성으로 변하였다. 불행히 이 결과는 릴리즈 이후 쓰기 캐싱을 기본으로 되돌려서 엄청난 성능감소를 유발하였다. 시스템에서 *hw.ata.wc sysctl* 변수를 확인하여 기본으로 되어있는지 체크한다. IDE 쓰기 캐싱이 꺼져있다면 커널 변수를 1로 되돌려서 쓰기 캐싱을 사용할 수 있다. 이 설정은 부팅할 때 부트로더에 의해 적용된다. 커널 부트 후 아무런 효과가 없다면 재 부팅 한다.

더 많은 정보는 *ata(4)*를 본다.

### 11.12.1.6 *SCSI\_DELAY (kern.cam.scsi\_delay)*

*SCSI\_DELAY* 커널 설정은 시스템 부팅시간을 줄이는데 사용될 수 있다. 기본값은 상당히 높고 부트 프로세스에서 15+ 초 안에 응답할 수 있다. 이 설정을 5초로 줄여도 보통 동작한다(특히 요즘 드라이브에서는). FreeBSD의 새로운 버전은(5.0+) *kern.cam.scsi\_delay*를 사용 부팅할 때 조정할 수 있다. 커널 설정옵션은 *seconds*가 아닌 *milliseconds* 값으로 조정할 수 있다.

## 11.12.2 Soft Updates

*tunefs(8)* 프로그램을 파일시스템 튜닝 도구로 사용할 수 있다. 이 프로그램은 다양한 옵션을 가지고 있지만 여기서는 다음 명령으로 수행할 수 있는 소프트웨어업데이트 켜기와 끄는 것만 설명한다:

---

```
# tuneufs -n enable /filesystem
# tuneufs -n disable /filesystem
```

파일시스템이 마운트 된 상태에서는 tuneufs(8)로 파일시스템을 변경할 수 없다. 소프트웨어 업데이트를 활성화하기 적당한 시간은 어떤 파티션도 마운트되지 않은 싱글 유저모드에서다.

**Note:** FreeBSD 4.5 에서 newfs(8)에 -u 옵션을 사용하여 파티션을 생성하는 동안 소프트웨어 업데이트를 활성화할 수 있다.

소프트웨어 업데이트는 메모리 캐시를 사용하여 주로 파일을 생성하고 삭제하는 메타데이터 성능을 확실히 향상시킨다. 모든 파일시스템에 소프트웨어 업데이트 사용을 권장한다. 알고 있어야 되는 소프트웨어 업데이트의 두 가지 단점이 있다: 첫째 소프트웨어 업데이트는 시스템에 문제가 생겨도 파일시스템의 무결성은 보장하지만 물리적으로 디스크를 업데이트하는 몇 초 동안(또는 몇 분 동안)에 발생한 문제에 대해서는 무결성을 보장하지 못한다. 시스템에 문제가 발생하면 다른 것보다 중요한 작업을 잃어버리게 될 것이다. 둘째로 소프트웨어 업데이트는 파일시스템 블록의 freeing 을 늦춘다. 거의 꽉 찬 상태의 파일시스템 블록(root 파일 시스템 같은)을 가지고 있고 make installworld 같은 중요한 업데이트를 수행하면 파일시스템 공간 부족으로 업데이트가 실패할 수 있다.

### 11.12.2.1 소프트웨어 업데이트에 관한 자세한 설명

파일시스템 메타데이터를 디스크(메타데이터 업데이트는 컨텐츠가 아닌 inodes 나 디렉터리 같은 데이터를 업데이트한다)에 쓰는 전통적인 두 가지 접근방식이 있다.

전통적으로 기본적인 동작은 메타데이터 업데이트를 동기적으로 수행한다. 디렉터리가 변경되었다면 시스템은 바뀐 것을 실제로 디스크에 작성할 때까지 기다린다. 파일 데이터 버퍼가 버퍼 캐시에 전달되면 나중에 동기적으로 디스크를 백업한다. 이런 방식의 장점은 안전하게 운용된다는 것이다. 업데이트할 때 실패했다라도 메타데이터는 항상 무결성 상태가 유지된다. 파일은 완벽하게 생성되었거나 그렇지 않을 것이다. 파일의 데이터블록은 문제가 있을 때 버퍼 캐시가 없는 디스크에 저장하는 방법을 못 찾고 fsck(8)은 문제를 감지하고 파일길이를 0으로 설정하여 파일시스템을 복원할 수 있다. 게다가 이러한 동작은 단순 명료하다. 단점은 메타데이터 변경이 느리게 진행된다는 것이다. 예를 들어 rm -r 은 디렉터리 순서로 모든 파일을 삭제하지만 각 디렉터리의 변경(파일 삭제)은 디스크에 동기적으로 작성된다. 이것은 디렉터리와 inode 테이블 업데이트하고 파일에 의해 할당된 간접적인 블록

---

도 포함할 것이다. 비슷하게 고려해야 될 사항은 전개된 커다란 계층(`tar -x`)에도 적용하는 것이다.

두 번째 경우는 비 동기적인 메타데이터 업데이트다. 이것은 리눅스/`ext2fs` 와 \*BSD `ufs` 의 `mount -o async` 에 기본이다. 모든 메타데이터 업데이트는 단순히 버퍼 캐시로 이동되기 때문에 이 결과 이들은 파일내용 데이터 업데이트와 혼합된다. 이 방식의 장점은 각 메타데이터 업데이트가 디스크에 작성될 때까지 기다릴 필요가 없기 때문에 모든 운용은 많은 메타 데이터를 업데이트할 때 동기방식보다 더욱 빠르게 진행된다. 또한 이 방식은 명확하고 단순해서 코드 속으로 버그가 유입될 위험이 적어진다. 단점은 파일시스템의 무결성을 보장하지 못한다는 것이다. 많은 메타데이터 업데이트에 실패(전원불량 또는 누군가 리셋 버튼을 눌렀다면)했다면 파일시스템은 예측할 수 없는 상태로 남아 있을 것이다. 시스템이 다시 부팅될 때 파일시스템의 상태에 대해 고려할 기회가 없다; 파일의 데이터 블록은 이미 디스크에 작성되었지만 `inode` 테이블의 업데이트나 관련된 디렉터리는 그렇지 않다. 따라서 실제로 문제(필요한 정보가 디스크에 없기 때문이다)를 해결할 수 있는 `fsck` 실행이 불가능하다. 파일시스템이 복구가 불가능할 정도로 파괴되었다면 유일한 방법이 `newfs(8)`을 이용하여 백업데이터에서 복구한다.

이 문제를 위한 일반적인 해결책은 *journaling* 이라고 부르는 *dirty region logging* 도구를 이용한다. 이 용어가 무결성에 사용된 적이 없더라도 종종 로그 트랜잭션의 다른 형식에 적용되었다. 메타데이터 업데이트는 아직도 동기적으로 수행되지만 디스크의 작은 공간에만 적용된다. 나중에 이들은 적당한 위치로 이동된다. 왜냐하면 로그 영역은 작고 디스크에서 인접하는 지역이며 디스크 헤드를 움직일 만큼 충분한 거리가 없고 사용량이 많기 때문에 이 옵션은 동기적인 업데이트보다 빠르다. 게다가 실행은 상당히 단순하기 때문에 버그가 존재할 위험이 낮다. 단점은 모든 메타데이터가 두 번씩(한번은 로그 영역에 또 한번은 적당한 위치에) 작성되기 때문에 일반적인 작업을 위한 성능은 “비관적”이다. 이와 반대로 문제가 발생한 경우 미결중인 모든 메타데이터 운용은 빠르게 롤백되거나 `fast` 파일시스템 시작의 결과로 시스템이 다시 돌아온 후 로그 영역으로부터 완료될 수 있다.

버클리 FFS 의 개발자인 Kirk McKusick 는 이 문제를 소프트 업데이트로 해결하였다: 모든 미결중인 메타데이터 업데이트는 메모리에 잡아두고 순서대로(“메타데이터 업데이트 순서”) 디스크에 작성된다. 이것은 부하가 많은 메타데이터 운용의 경우 이전 데이터가 아직도 메모리에 있고 디스크에 아직 작성되지 않았다면 나중에 업데이트된 데이터가 이전 것을 덮어쓰는 결과를 불러왔다. 따라서 디렉터리는 업데이트가 디스크에 작성되기 전에 보통 메모리에 작성된다(데이터블록은 위치에 따라 분류되기 때문에 그들의 메타데이터 이전에 디스크에 저장되지 않는다). 시스템에 문제가 발생했다면 이로 인해 무조건 "로그를 되돌리는" 원인이 되었다: 모든 작업이 한번도 디스크에 저장되지 않았다면 디스크에 저장하지 못하게

---

된다. 무결한 파일시스템 상태는 30 에서 60 초 전의 상태로 보존된다. 이 알고리즘은 사용되고 있는 모든 리소스를 그들의 적당한 비트맵(블록과 inode)으로 표시하도록 보장하는데 사용된다. 문제 발생 후 유일하게 발생하는 리소스 할당 에러는 실제로 사용되지 않은 리소스를 "사용 중"으로 표시하는 것이다. fsck(8)은 이 상황을 인지하고 더 이상 사용하지 않는 리소스를 풀어준다. mount -f 를 사용하여 강제로 마운트한 상태에서 충돌이 발생한 후 파일시스템의 지저분한 상태를 무시해도 된다. 사용하지 않는 리소스를 되돌리기 위해 fsck(8)를 나중에 실행해야 된다. 이것이 백그라운드 fsck 의 아이디어다: 시스템이 시작할 때 오직 파일시스템의 스냅샷만 저장되어 있다. fsck 는 이 후에 시작할 수 있다. 모든 파일시스템은 결함이 있는 상태로 마운트할 수 있기 때문에 시스템 시작 프로세스는 멀티 유저 모드에 있다. 백그라운드 fsck 는 사용하지 않는 리소스를 풀기 위해 fsck 가 필요한 모든 파일시스템에 스케줄 되어 있다(소프트업데이트를 사용하지 않는 파일시스템은 보통 포그라운드 fsck 를 사용해야 된다).

장점은 메타데이터 운용이 거의 동기화 업데이트(예: 메타데이터 작성을 두 번씩 해야 되는 *logging* 보다 빠르다)처럼 빠르다는 것이다. 단점은 코드(유저 데이터의 유실에 민감한 지역에서는 버그로 인한 높은 위험이 있다)의 복잡성과 높은 메모리 소모다. 게다가 사용된 몇 가지 기능이 있다. 문제 발생 후 파일시스템은 약간 예전 상태로 나타난다. 표준 동기적인 접근방식은 메타데이터와 파일 내용이 디스크에 한번도 작성되지 않았기 때문에 fsck 이후 소프트웨어 업데이트 파일시스템에 남아 있지 않아야 되는 크기가 0 인 파일이 남아있는 원인이 된다. 디스크 공간은 업데이트가 디스크에 적용될 때까지 사용하지 못한다. 이것은 모든 파일을 두 번씩 작성할 수 있는 충분한 공간이 없는 파일시스템에 많은 데이터를 설치할 때 문제가 발생할 수 있다.

## 11.13 커널 제한 튜닝.

### 11.13.1 파일/프로세스 제한

#### 11.13.1.1 *kern.maxfiles*

*kern.maxfiles* 은 시스템의 요구에 따라 증가하거나 감소 시킬 수 있다. 이 변수는 시스템의 디스크립터(descriptor) 수를 가리킨다. 파일 디스크립터 테이블이 가득 차면 dmesg 명령으로 볼수 있는 시스템 메시지 버퍼에서 file: table is full 이라는 메시지를 계속적으로 보여준

---

다.

각 열린 파일, 소켓 또는 fifo 는 하나의 파일 디스크립터(descriptor)를 사용한다. 대규모의 서버 제품은 동시에 운용되는 서비스의 수와 종류에 따라 수천 개의 파일 디스크립터가 요구될 것이다.

*Kern.maxfiles* 의 기본값은 커널 설정파일에서 *MAXUSERS* 옵션으로 지정한다.

*Kern.maxfiles* 은 *MAXUSERS* 의 값에 비례하여 커진다. 사용자 커널을 컴파일 할 때 시스템의 용도에 따라 이 커널 설정옵션을 지정하는 것도 좋은 생각이다. 이 숫자로 대부분의 커널에는 이미 제한이 설정되어 있다. 머신이 한번에 256 유저를 실제로 연결할 수 없더라도 필요한 리소스는 고성능 웹 서버와 비슷할 것이다.

**Note:** FreeBSD 4.5 에서 커널 설정파일의 *MAXUSERS* 가 0으로 설정된 것은 시스템이 가지고 있는 RAM 의 양에 따라 적절하게 지정된 기본값이다.

### 11.13.1.2 *kern.ipc.somaxconn*

*kern.ipc.somaxconn* sysctl 변수는 새로운 TCP 연결을 허용하는 listen 큐의 크기를 제한한다. 기본 값 128은 부하가 많은 웹 서버 환경에서 새로운 연결을 제공하기에는 보통 너무 낮다. 이런 환경에서 이 값을 1024 나 더 큰 값으로 변경하기를 권장한다. 서비스 데몬이 listen 큐 크기를(예: sendmail(8)이나 Apache) 제한하지만 가끔 큐 크기를 조정하는 설정파일을 가지고 있기도 하다. 대형 listen 큐는 서비스 거부 공격을(DOS) 피하기 좋을 것이다.

## 11.13.2 네트워크 제한

*NMBCLUSTERS* 커널 설정옵션은 시스템에서 사용할 수 있는 네트워크 Mbufs 양을 보여준다. 낮은 Mbufs 와 트래픽이 많은 서버는 FreeBSD 의 효율성을 저하시킨다. 각 클러스터는 대략 메모리의 2K 정도를 표시하기 때문에 1024 값은 네트워크 버퍼를 위해 남겨둔 2Mbytes 의 커널 메모리를 나타낸다. 간단한 계산법으로 필요한 양에 따라 계산할 수 있다. 동시에 1000 개의 연결이 필요한 웹 서버를 가지고 있고 각 연결은 수신에 16k 와 송신에 16k 버퍼가 필요하다면 웹 서버에 대략 32MB 의 네트워크 버퍼가 필요하다. 가장 좋은 방법은 두 배를 설정해서  $2 \times 32\text{MB} / 2\text{KB} = 64\text{MB} / 2\text{KB} = 32768$  이 된다. 우리는 4096 에서 32768 사이의 값을 가장 큰 메모리 양을 가지고 있는 머신에 권장한다. 이런 상황이 아닌 경우 이 매개변수에 임의대로 높은 값을 지정한다면 이것은 부팅할 때 충돌을 유발할 수 있

---

다. `netstat(1)`에 `-m` 옵션은 네트워크 클러스터 사용을 감지하는데 사용할 수 있을 것이다.

튜닝할 수 있는 `kern.ipc.nmbclusters` 로더는 부팅할 때 튜닝한다. 오직 예전 버전의 FreeBSD 만 `NMBCLUSTERS` 커널 `config(8)` 옵션이 필요하다.

`sendfile(2)` 시스템 콜을 엄청나게 사용하는 부하가 많은 서버는 `NSFBUFS` 커널 설정옵션 또는 `/boot/loader.conf` 에서(자세한 사항은 `loader(8)`을 본다) 값을 수정하여 `sendfile(2)` 버퍼의 숫자를 늘려야 할 것이다. 이 매개변수에 조정이 필요한 때는 프로세스가 “`sbufa`” 상태로 보일 때다. `sysctl` 변수 `kern.ipc.nsfbufs` 는 커널 설정변수에서 읽기전용이다. 이 매개변수는 명목상 `kern.maxusers` 로 조정되는 것 같지만 적당한 튜닝이 필요할 것이다.

**중요:** 소켓이 non-blocking 으로 표시되었더라도 non-blocking 소켓에서 `sendfile(2)`를 호출하는 것은 `sendfile(2)`가 `struct sf_buf`를 사용할 수 있을 때까지 blocking 을 호출하기 때문이다

### 11.13.2.1 *net.inet.ip.porrange.\**

`net.inet.ip.porrange.*` `sysctl` 변수는 TCP 와 UDP 소켓에 따라 자동으로 포트번호 범위를 제어한다. 3 가지 범위가 있다: 낮은 범위, 기본 범위, 높은 범위. 대부분의 네트워크 프로그램은 `net.inet.ip.porrange.first`와 `net.inet.ip.porrange.last`가 제어하는 1024 에서 5000 까지의 기본 범위를 사용한다. 제한된 포트 범위는 외부연결에 사용되고 시스템의 포트가 부족한 특별한 경우에서도 운용할 수 있다. 이것은 부하가 큰 웹 proxy 를 운용할 때 보통 발생한다. 포트 범위는 보통 웹 서버처럼 주로 수신 연결만 제어하거나 메일 릴레이처럼 제한적으로 송신하는 서버를 운용할 때 문제가 발생하지 않는다. 포트가 부족한 상태에서 운용 중일 때 `net.inet.ip.porrange.last`를 증가시킬 것을 권장한다. 10000, 20000 또는 30000 정도가 적당할 것이다. 포트 범위를 변경할 때 방화벽의 효과를 고려한다. 어떤 방화벽은 많은 포트(일반적으로 낮은 번호의 포트) 범위를 막고 외부연결에 높은 포트범위를 사용할 것으로 예상한다. 이러한 이유로 `net.inet.ip.porrange.first`를 낮출 것을 권장한다.

### 11.13.2.2 TCP 대역폭 지연 제품

TCP 대역을 지연하는 제품 제한은 NetBSD 의 TCP/Vegas 와 비슷하다.

`net.inet.tcp.inflight_enable` `sysctl` 변수를 1 로 설정해서 활성화 할 수 있다. 시스템은 최적의 처리량을 유지하기 위해 각 연결당 대역폭을 지연하는 제품을 계산해서 네트워크 큐의

---

데이터 양을 제한한다.

이 기능은 모뎀, 기가 비트 이더넷 또는 고속 WAN 링크로(또는 고대역 지연 제품의 다른 링크) 데이터를 서비스할 때, 특히 윈도우를 사용하거나 거대한 윈도우를 보내도록 설정했을 때 유용하다. 이 옵션을 활성화 시키면 `net.inet.tcp.inflight_debug` 를 0으로(디버깅을 비활성으로) 설정하고 제품이 `net.inet.tcp.inflight_min` 설정을 사용하면 최소 6144의 이득을 얻을 것이다. 그러나 최소 설정은 연결에 따라 대역폭 제한을 효과적으로 비활성할 것이다. 기능 제한은 라우트와 스위치 패킷 큐 사이에서 빌드하는 데이터 양을 감소시키고 로컬 호스트의 인터페이스 큐에서 빌드하는 데이터 양도 감소시킨다. 특히 매우 느린 모뎀을 사용한 몇 개의 패킷이 큐에 쌓인 쌍방향 연결도 *Round Trip Times*으로 운용할 수 있다. 그러나 이 기능은 데이터 전달에만(업로드/서버 사이에서) 효과가 있다. 데이터를 받을때는(다운로드) 전혀 효과가 없다.

`net.inet.tcp.inflight_stab` 를 조정하는 것은 권장하지 않는다. 이 매개변수는 기본적으로 20 이고 2는 대역폭 지연 제품 윈도우 계산에 추가된 최대 패킷을 의미한다. 추가적인 윈도우는 알고리즘을 안정시키고 상황을 변경시키기 위한 응답을 개선하기 위해 필요하지만 높은 핑 응답이 있는 느린 연결의(빠른 알고리즘이 없는 것보다 아직도 느리지만) 결과일 수 있다. 이러한 경우 이 매개변수를 15, 10 또는 5로 낮추기를 원할 것이다; 그리고 원하는 효과를 얻기 위해 `net.inet.tcp.inflight_min` 을(예를 들면 3500으로) 낮출 것이다. 이들 매개변수를 낮추는 것은 마지막 수단이다.

## 11.14 스왑 공간 추가

완벽한 계획을 가지고 있더라도 시스템은 종종 원하는 대로 작동하지 않는다. 더 많은 스왑 공간이 필요하다면 추가하는 것은 간단하다. 스왑 공간을 추가하는 3가지 방법이 있다: 새로운 하드 드라이브 추가, NFS를 스왑으로 이용 그리고 가지고 있는 파티션에 스왑 파일 생성.

### 11.14.1 새로운 하드 드라이브에 스왑 생성

스왑을 추가하는 가장 좋은 방법은 다른 하드 드라이브를 추가하는 것이다. 다른 하드 드라이브를 사용할 수 있다면 스왑을 어떻게 준비하는지 제안하는 핸드북의 초기 설정 섹션의



---

(11.2 장) 스왑 공간에(11.2.1.2 장) 대한 내용을 다시 읽는다.

## 11.14.2 NFS 를 이용한 스왑

NFS 를 이용하는 스왑 방식은 스왑으로 사용할 로컬 하드디스크가 없을 때만 권장한다. NFS 를 이용한 스왑은 FreeBSD 4.X 이전 버전에서는 느리고 비효율적이다. 4.0-RELEASE 와 새로운 버전에서는 빠르고 효과적이다. 새로운 FreeBSD 버전이라도 NFS 스왑은 이용할 수 있는 네트워크 대역과 추가적으로 NFS 서버의 부하에 따라 제한적이다.

## 11.14.3 스왑 파일

스왑 파일로 사용할 특정 크기의 파일을 만들 수 있다. 예제에서 우리는 64MB 의 /usr/swap0 라고 부르는 파일을 사용한다. 물론 원하는 어떤 이름이라도 사용할 수 있다.

### [예제 스왑 파일 생성하기]

#### 1. FreeBSD 4.X 에서 스왑 파일 생성

- ① 커널 설정이 vnode 드라이버를 포함해야 된다. 이 설정은 최근 버전의 GENERIC 에 없다.

```
pseudo-device vn 1 #Vnode driver (turns a file into a device)
```

- ② vn-device 를 생성한다:

```
# cd /dev  
# sh MAKEDEV vn0
```

- ③ 스왑 파일 생성 (/usr/swap0):

```
# dd if=/dev/zero of=/usr/swap0 bs=1024k count=64
```

- ④ 적당한 퍼미션 설정 (/usr/swap0):

```
# chmod 0600 /usr/swap0
```

- ⑤ /etc/rc.conf 에서 스왑파일 활성화:

```
swapfile="/usr/swap0" # Set to name of swapfile if aux swapfile desired.
```

- ⑥ 머신을 재 부팅 하거나 다음 명령을 입력하여 스왑 파일을 즉시 활성화한다:

```
# vnconfig -e /dev/vn0b /usr/swap0 swap
```

## 2. FreeBSD 5.X 에서 스왑 파일 생성

- ① 커널 설정이 메모리 디스크 드라이버 (md(4))를 포함해야 된다. 이것은 기본 GENERIC 커널에 있다.

```
device md # Memory "disks"
```

- ② 스왑 파일 생성 (/usr/swap0):

```
# dd if=/dev/zero of=/usr/swap0 bs=1024k count=64
```

- ③ 적절한 퍼미션 설정 (/usr/swap0):

```
# chmod 0600 /usr/swap0
```

- ④ /etc/rc.conf 에서 스왑 파일 활성화하기:

```
swapfile="/usr/swap0" # Set to name of swapfile if aux swapfile desired.
```

- ⑤ 머신을 재 부팅하거나 다음 명령으로 스왑 파일을 즉시 활성화한다:

```
# mdconfig -a -t vnode -f /usr/swap0 -u 0 && swapon /dev/md0
```

---

## 11.15 전원과 리소스 관리

효과적인 방법으로 하드웨어 리소스를 이용하는 것은 매우 중요하다. ACPI 를 소개하기 전에 운영체제가 전원 사용량과 시스템의 온도를 적절히 조절하는 것은 매우 어렵다. 하드웨어는 *플러그엔 플레이 BIOS(PNPBIOS)* 또는 *Advanced Power Management(APM)* 같은 인터페이스가 내장된 몇몇 BIOS 가 제어한다. 전원과 리소스 관리는 현대 운영체제를 구성하는 하나의 키 컴포넌트다. 예를 들어 시스템 온도가 예상치 못하게 증가한다면 운영체제가 시스템 상태를(그리고 경고하는) 모니터 하기를 원할 것이다.

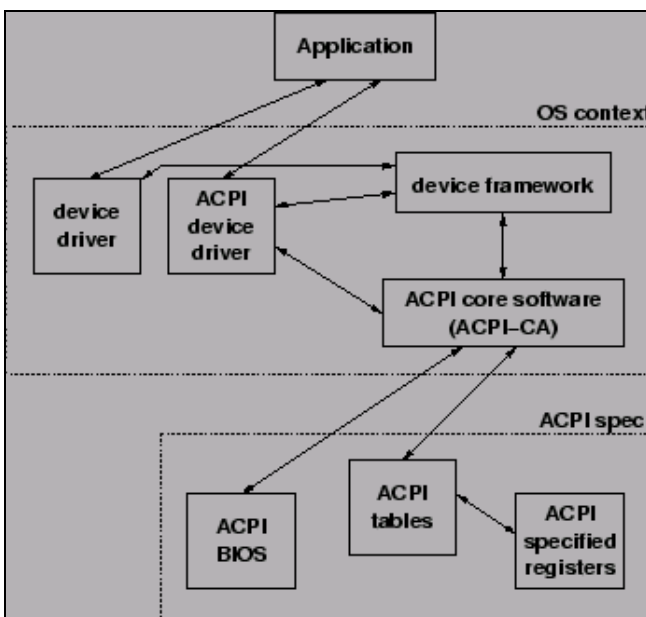
FreeBSD 핸드북의 이번 섹션은 ACPI 에 대한 포괄적인 정보를 제공한다. 레퍼런스는 마지막 부분에서 더 많은 읽을 거리를 제공한다. ACPI 는 FreeBSD 5.X 와 이후 버전의 기본 커널 모듈에서 이용할 수 있다는 것을 기억한다. FreeBSD 4.9 에서는 커널 설정파일에 *device acpi* 라인을 추가한 후 다시 빌드하여 ACPI 를 활성화 할 수 있다.

### 11.15.1 ACPI 는 무엇인가?

발전된 설정과 전원 인터페이스는(ACPI) 하드웨어 리소스와 전원 관리를 위해 표준 인터페이스를 제공하도록 벤더들이 협의하여 표준을 재정하였다.

*Operating System-directed configuration and Power Management* 가 키 요소다. 이 표준은 운영체제에(OS) 더 많은 제어와 유연성을 제공한다.

#### ACPI 아키텍처



---

현재 플러그 앤 플레이 인터페이스의 한계를(FreeBSD 4.X 에서 사용한 APM 같은) 확장한 현대 시스템이 ACPI 를 소개하였다. ACPI 는 APM 을(발전된 전원 관리) 직접 계승한다.

## 11.15.2 발전된 전원 관리의(APM) 결점

발전된 전원 관리(APM) 시스템의 활동에 따라 전원 사용량을 제어한다. APM BIOS 는 시스템 벤더에서 특정 하드웨어 플랫폼에 제공된다. OS 에서 APM 드라이버는 전원 레벨을 관리하는 APM 소프트웨어 인터페이스에 접근하는 매개체다.

APM 에 4 개의 중요한 문제가 있다. 첫째로 전원 관리는 BIOS 가(벤더에서 지정한) 하고 OS 는 이에 대해 아무것도 모른다. 이에 대한 하나의 예는 유저가 APM BIOS 에서 하드 드라이브 idle-time 을 설정하면 값이 초과되었을 때 BIOS 는 OS 의 허락없이 하드 드라이브 회전 수를 줄인다. 둘째로 APM 로직은 BIOS 에 내장되어 있어서 OS 영역 밖에서 운용된다. 이 의미는 유저가 ROM 에 새로운 내용을 입력해야 APM BIOS 의 문제를 해결할 수 있다는 것이다; 해결에 실패하면 시스템이 복구 불능 상태가 될 수 있는 매우 위험한 프로시저다. 셋째로 APM 은 벤더의 특정 기술이다. 이 의미는 벤더의 BIOS 에 패리티와(두 배의 수고) 버그가 있다면 다른 것에서도 해결할 수 없을 것이다. 마지막으로 APM BIOS 는 복잡한 전원 정책에 맞는 충분한 옵션이 없기 때문에 머신의 목적에 맞게 개조해야 될 것이다.

*플러그 앤 플레이 BIOS(PNPBIOS)*는 많은 상황에서 신뢰할 수 없다. PNPBIOS 는 16 비트 기술이기 때문에 OS 는 PNPBIOS 방법과 맞추기 위해 16 비트 에뮬레이션을 사용해야 된다.

FreeBSD APM 드라이버는 apm(4) 매뉴얼 페이지에 설명되어 있다.

## 11.15.3 ACPI 설정

acpi.ko 드라이버는 커널에 컴파일하지 않고 loader(8)로 시작할 때 기본적으로 적재된다. 이 의미를 생각해 보면 커널을 재 빌드하지 않고 다른 acpi.ko 로 변경했다면 모듈은 작동하기 더 쉽다. 이것은 테스트하기 쉽다는 장점이 있다. 또 다른 이유는 시스템이 시작된 후 ACPI 를 시작하는 것은 별로 유용하지 않고 어떤 경우는 치명적일 수 있다. 걱정된다면 모든 ACPI 를 비활성 한다. 시스템 버스가 다양한 하드웨어와 상호작용에 사용되기 때문에 드라이버는 내릴수도 없고 내려서도 안 된다. ACPI 는 acpiconf(8) 유틸리티로 비활성할 수 있다. 사실상 ACPI 와의 대부분의 상호작용은 acpiconf(8)로 할 수 있다. 기본적으로 이 의

---

미는 ACPI 에 관련된 어떤 것이라도 `dmesg(8)`에서 출력된다면 보통 ACPI 가 실행 중이라는 것이다.

**Note:** ACPI 와 APM 은 공존할 수 없고 따로 사용해야 된다. 드라이버에서 다른 드라이버가 실행 중이라고 경고 하면 마지막에 로드한 것은 중지된다.

ACPI 는 `acpicnf(8)`에 `-s` 플래그와 `1-5` 옵션으로 시스템을 절전모드로 바꿀 수 있다. 대부분의 유저는 `1` 만 필요할 것이다. 옵션 `5` 는 다음 명령처럼 작동하는 `soft-off` 다:

```
# halt -p
```

다른 옵션도 이용할 수 있다. 더 많은 정보는 `acpicnf(8)` 매뉴얼 페이지를 체크한다.

## 11.16 FreeBSD ACPI 사용과 디버깅

ACPI 는 장치 발견, 전원 사용량 관리 그리고 BIOS 가 이전에 관리하던 다양한 하드웨어에 표준적인 접근방식을 제공하는 근본적으로 새로운 방법이다. ACPI 는 모든 시스템으로 개발이 계속 진행되고 있지만 어떤 마더보드와 FreeBSD 의 커널 서브시스템에서의 불안정한 버그가 있고 Intel ACPI-CA 인터프리터에서 계속 버그가 나타나고 있다.

이 문서는 문제의 핵심을 확인하고 해결책을 개발할 수 있도록 FreeBSD ACPI 관리자를 지원한다.

### 11.16.1 디버깅 정보 제출

**Note:** 문제를 보내기 전에 가능하다면 BIOS 와 컨트롤러 펌웨어를 최신 버전으로 사용해 본다.

문제를 제출할 때 다음 정보를 `freebsd-acpi@FreeBSD.org` 에 보낸다

- 시스템 종류와 모델 등 버그를 유발할 수 있는 모든 것을 포함하여 버그 증상을 설명한다. 또한 새로운 버그라면 언제 버그가 발생했는지 정확히 적는다.

- 
- 버그로 인해 생성된 에러메시지를 포함하여 “boot -v” 후의 dmesg 결과를 적는다.
  - ACPI 를 비활성해서 문제가 없어졌다면 ACPI 를 비활성하고 “boot -v”에서 dmesg 결과를 적는다.
  - “sysctl hw.acpi”의 결과도 보낸다. 이것은 시스템이 어떤 기능을 제공하는지 확인하기 좋다.
  - ASL 을 찾을 수 있는 URL. ASL 은 너무 크기 때문에 직접 보내지 않는다. 다음 명령으로 ASL 을 복사한다:

```
# acpidump -t -d > $NAME-$SYSTEM.asl
```

(\$NAME 는 로그인 이름으로 그리고 \$SYSTEM 은 제조사/모델로 대체한다. 예를 들면: njl-Fooco6000.asl)

대부분의 개발자들이 freebsd-current 메일링 리스트를 확인하지만 확실히 볼 수 있도록 acpi 에 리스트 한다. 대부분의 개발자들이 직업을 가지고 있기 때문에 오랫동안 기다려야 될수 있다. 문제에 대한 답변이 바로 오지 않는다면 send-pr(1)을 사용하여 PR 를 통해 문의할 수 있다. PR 을 입력할 때 위에서 요청한 정보를 포함한다. 이렇게 해주는 것이 문제를 분석해서 해결할 수 있도록 돕는 방법이다. 우리는 PR 을 리포팅 메커니즘으로 사용하지 않고 남아있는 문제를 위해 사용하기 때문에 처음에 acpi 리스트에 메일을 보내지 않았다면 PR 을 보내지 않는다. 다른 사람이 여러분과 같은 문제에 대해 먼저 질문했을 수도 있다.

## 11.16.2 백그라운드

ACPI 는 ia32 (x86), ia64 (Itanium)과 adm64(AMD) 아키텍처를 따르는 모든 현대 컴퓨터에 있다. 완벽한 표준은 CPU 성능 관리, 단계별 전원 제어, 온도 범위, 다양한 배터리 시스템, 임베디드 컨트롤러와 버스 목록을 포함하는 다양한 기능을 가지고 있다. 대부분의 시스템 도구는 완전한 표준보다 적다. 예를 들면 데스크톱 시스템은 오직 버스관리 부분만 가지고 있고 반면 노트북은 쿨러와 배터리관리 부분을 지원한다.

ACPI 시스템은 다양한 컴포넌트를 가지고 있다. BIOS 와 칩셋 벤더는 APIC 맵(SMP 에 사용하는), 설정 레지스터 그리고 단순한 설정 값처럼 지정하는 다양한 fixed table 을(예: FADT)

---

메모리에서 제공한다. 추가적으로 바이트 코드(DSDT) 테이블을 트리처럼 지정하는 장치와 메소드 이름 공간이 제공된다.

ACPI 드라이버는 fixed table 분석, 바이트 코드 인터프리터 수행, 장치 드라이버를 수정하고 커널이 ACPI 정보에 접근하도록 해야 된다. FreeBSD 의 경우 Intel 이 리눅스와 NetBSD 에 공유되는 인터프리터를(ACPI-CA) 제공하고 있다. ACPI-CA 소스 코드의 경로는 src/sys/contrib/dev/acpica 다. FreeBSD 에서 ACPI-CA 가 작동하게 하도록 하는 글루(glue) 코드는 src/sys/dev/acpica/Osd 에 있다. 마지막으로 다양한 ACPI 장치를 실행하는 드라이버는 src/sys/dev/acpica 에서 찾을 수 있다.

### 11.16.3 일반적인 문제

ACPI 가 정확히 작동하려면 모든 부분이 정확히 작동해야 된다. 여기서는 종종 발생하는 문제를 해결할 수 있도록 일반적인 문제를 다룬다.

#### 11.16.3.1 대기모드/정상모드

ACPI 는 RAM(STR) 상태에 세 개의 대기모드 *S1-S3* 를, 디스크 상태에는 *S4* 라고 부르는 한 개의 대기모드(*STD*)를 그리고 시스템에 전원은 연결했지만 시동은 하지 않은 일반적인 “soft off” 상태인 *S5* 를 가지고 있다. *S4* 는 실제로 두 가지로 분리된다. *S4* BIOS 는 디스크 대기모드를 지원하는 BIOS 다. *S4* OS 는 전체를 운영 체제가 실행한다.

대기모드와 관련된 아이템 체크를 위해 `sysctl hw.acpi` 를 시작한다. 여기 저자의 Thinkpad 결과가 있다:

```
hw.acpi.supported_sleep_state: S3 S4 S5
hw.acpi.s4bios: 0
```

이 의미는 *S3*, *S4* OS 와 *S5* 를 테스트하기 위해 `acpicconf -s` 를 사용할 수 있다는 것이다. `s4bios` 가 1(1) 이라면 *S4* OS 대신 *S4* BIOS 를 지원한다.

대기/정상 모드를 테스트할 때 지원된다면 *S1* 으로 시작한다. 더 많은 드라이버 지원이 필요없기 때문에 이 상태는 대부분 동작한다. 아무도 *S2* 실행을 가지고 있지 않지만 여러분이 가지고 있다면 *S1* 과 비슷하다. 다음은 *S3* 다. 이것은 가장 심오한 STR 상태이고 하드웨

---

어를 적절히 다시 초기화하도록 수많은 드라이버를 지원해야 된다. 다시 문제가 발생한다면 수많은 드라이버/하드웨어에 더 많은 테스트와 작업이 필요하기 때문에 문제가 해결될 것이라는 기대는 하지말고 acpi 리스트에 메일을 보낸다.

이 문제 분석을 위해 가능한 모든 드라이버를 삭제한다. 문제가 발생하지 않는다면 문제가 다시 발생할 때까지 드라이버를 로드한다. nvidia.ko, X11 디스플레이 드라이버 그리고 USB 같은 바이너리 드라이버가 일반적으로 문제를 유발하는 반면 이더넷 인터페이스는 보통 제대로 동작한다. 이상 없이 드라이버를 로드/언 로드할 수 있다면 적절한 명령을 `/etc/rc.suspend` 와 `/etc/rc.resume` 에 추가하여 자동으로 수행할 수 있다. 주석 처리된 드라이버 로드와 언 로드 예제가 있다. 정상모드로 돌아온 후 화면이 영망이라면 `hw.acpi.reset_video` 를 0 으로 설정한다. `hw.acpi.sleep_delay` 값을 크거나 작게 설정하면서 확인한다.

시도해 볼만한 다른 것은 같은 하드웨어에서 그들의 대기/정상 모드를 지원하고 테스트하는 최신 리눅스 배포본의 ACPI 를 로드 해 본다. 리눅스에서는 동작한다면 이것은 FreeBSD 드라이버 문제이기 때문에 어떤 드라이버가 문제를 유발하는지 쉽게 찾을 수 있다. 그 ACPI 관리자는 보통 다른 드라이버를(예: 사운드, ATA 등) 관리하지 않기 때문에 드라이버 문제는 결국 `freebsd-current` 리스트에 올려지고 드라이버 관리자에게 메일이 전송된다. 문제 해결을 좋아한다면 문제를 분석하기 위해 의심스러운 드라이버에 디버그 `printf(3)`를 입력해서 어느 곳에서 정상으로 되돌아오는 기능이 정지 하는지 확인한다.

마지막으로 ACPI 를 비활성하고 대신 APM 을 활성화한다. 대기/정상 모드가 APM 과 동작한다면 APM 을 사용하는 것이 좋을 것 같고 특히 예전 하드웨어는(2000년 이전의) 더욱 그렇다. ACPI 를 정확히 지원 받기 위해 벤더에 문의해야 되고 오래된 하드웨어는 ACPI 와 BIOS 문제가 대부분이다.

### 11.16.3.2 시스템 정지(일시적 또는 영구적)

시스템이 정지하는 대부분의 이유는 인터럽트를 잃어버리거나 대량의 인터럽트로 발생한다. 부팅하기 전에 BIOS 가 인터럽트를 어떻게 설정하는지, APIC (MADT) 테이블의 정확함과 SCI 의 라우팅에 따라 칩셋은 수많은 문제가 발생한다.

`vmstat -r`의 출력에서 "acpi0"를 가진 라인을 확인하여 대량의 인터럽트는 인터럽트 유실과 구분할 수 있다. 숫자가 몇초 동안 증가한다면 대량의 인터럽트 발생이다. 시스템이 정지한 것 같다면 DDB 를(콘솔에서 **CTRL + ALT + ESC**) 멈추고 `show interrupts` 를 입력한다.



---

인터럽트 문제에서는 가장 최우선으로 loader.conf 에 `hint.apic.0.disabled="1"`로 APIC 지원을 비활성 한다.

### 11.16.3.3 패닉

패닉은 ACPI 에 상당히 드문 경우지만 가장 긴급하게 수정할 부분이다. 첫 번째 단계는 패닉을 줄이고(가능 하다면) 문제를 추적하기 위해 패닉 상황에 최대한 접근한다. `options DDB`를 활성화하고 시리얼 콘솔을 설정하거나 `dump(8)` 파티션을 설정한다. `tr`로 DDB 에서 흔적을 찾을 수 있다. 흔적을 직접 찾는다면 최소한 상단에서 5 줄 하단에서 5 줄을 복사한다.

그리고 ACPI 를 비활성하고 부팅하여 문제를 명확히 한다. 정상이라면 `debug.acpi.disable`의 다양한 값을 사용하여 ACPI 서브 시스템을 확인할 수 있다. 예제는 `acpi(4)` 매뉴얼 페이지를 확인한다.

### 11.16.3.4 대기모드나 셧다운 후 시스템 전원 켜기

첫째로 loader.conf(5)에서 `hw.acpi.disable_on_poweroff="0"`으로 설정한다. 이것은 셧다운 하는 동안 다양한 이벤트를 비활성할때 ACPI 를 유지한다. 어떤 시스템은 같은 이유로 이 값을 "1"로(기본값) 설정해야 된다. 이것은 보통 대기모드나 전원을 끈 후 갑자기 전원이 다시 들어오는 문제를 수정한다.

### 11.16.3.5 다른 문제

ACPI 와 다른 문제가(docking station 과 동작, 장치가 검색되지 않는 등) 있다면 문제를 설명한 메일을 메일링 리스트에 보낸다. 그러나 이들 중 어떤 문제는 ACPI 시스템에서 완료 되지 않은 부분일 수 있기 때문에 시간이 걸릴 수 있다. 참고 기다리며 우리가 보낼 패치 테스트를 준비한다.

## 11.16.4 ASL, acpidump 그리고 IASL

---

대부분의 일반적인 문제는 BIOS 벤더가 정확하지 않은(버그가 있는) 바이트 코드를 제공하는 것이다. 이것은 보통 다음과 같이 커널 콘솔메시지로 증상이 나타난다:

```
ACPI-1287: *** Error: Method execution failed [WW_SB_.PCI0.LPC0.FIGD._STA] (Node
0xc3f6d160), AE_NOT_FOUND
```

종종 BIOS 를 최신 버전으로 업데이트하여 이런 문제를 해결할 수 있다. 대부분의 콘솔 메시지는 큰 문제가 없지만 배터리 상태 작동불능 같은 다른 문제가 발생한다면 AML 이 문제를 찾는 시작 포인트가 된다. AML 이라고 알려진 바이트 코드는 ASL 이라는 소스 언어에서 컴파일 되었다. AML 은 DSDT 라고 알려진 테이블에서 찾는다. ASL 을 복사하려면 `acpidump(8)`을 사용한다. `-f` (fixed tables 의 내용을 보여주는)와 `-d` (AML 을 ASL 로 분해하는) 두 가지 옵션을 사용한다. 예제 구문은 디버깅 정보 제출을 본다.

여러분이 가장 간단히 처음으로 체크할 수 있는 것은 에러를 체크하기 위해 ASL 을 다시 컴파일 한다. 경고 메시지는 보통 무시할 수 있지만 에러는 보통 ACPI 가 정확히 동작하는 데 지장이 되는 것들이다. ASL 을 다시 컴파일하려면 다음 명령을 사용한다:

```
# iasl your.asl
```

## 11.16.5 ASL 수정

오랫동안 우리의 목표는 ACPI 가 유저의 간섭없이 대부분의 사항을 해결하기를 바래 왔다. 그러나 우리는 BIOS 벤더가 만든 일반적인 실수에 대한 개발을 아직도 진행 중이다. Microsoft 인터프리터 (`acpi.sys` 와 `acpiec.sys`)는 표준을 정확히 지키지 않기 때문에 윈도 위에서만 ACPI 를 테스트한 많은 BIOS 벤더는 ASL 을 수정하지 않는다. 우리는 계속해서 Microsoft 의 인터프리터가 허용하는 표준적이지 않은 동작을 확인해서 문서화하여 복사하기 때문에 FreeBSD 는 유저가 강제로 ASL 을 수정하지 않아도 동작하기를 기대한다. 우리가 동작을 확인해서 문제를 해결하기 쉽도록 ASL 을 직접 수정할 수 있다. 이것이 동작한다면 우리가 ACPI-CA 에서 버그를 해결하여 여러분이 수정할 필요가 없도록 예전것과 새로운 ASL 의 `diff(1)`을 보내 주기 바란다.

여기 일반적인 에러 메시지의 발생 원인과 해결책이 있다:

---

### 11.16.5.1 \_OS 의존성

어떤 AML 에서 세계는 다양한 윈도우 버전으로만 이루어 졌다고 간주한다. 이것이 문제를 해결하는 방법이라면 FreeBSD 에게 그냥 OS 라고 지정할 수 있다. 이것을 지정하는 가장 쉬운 방법은 /boot/loader.conf 에서 `hw.acpi.osname="Windows 2001"`로 지정하거나 ASL 에서 찾은 비슷한 문자열로 지정한다.

### 11.16.5.2 Return 문 빠짐

어떤 방법은 표준적으로 필요한 값을 정확히 되돌려 주지 않는다. ACPI-CA 가 이것을 제어 하지 않기 때문에 FreeBSD 는 값을 무조건 되돌려 주도록 하여 해결하고 있다. 어떤 값을 되돌려 받아야 되는지 알고 있다면 필요한 곳에 정확한 Return 문을 추가할 수 있다. 강제로 iasl 을 ASL 에 컴파일하려면 `-f` 플래그를 사용한다.

### 11.16.5.3 기본 AML 덮어쓰기

your.asl 을 수정한 후 컴파일하려면 다음 명령을 실행한다:

```
# iasl your.asl
```

컴파일하는 동안 에러가 발생하더라도 `-f` 플래그를 추가하여 강제로 AML 을 생성할 수 있다. 어떤 에러(예: Return 문이 빠진)는 인터프리터가 자동으로 해결한다.

DSDT.aml 은 iasl 의 기본 출력파일 이름이다. /boot/loader.conf 를 다음과 같이 수정하여 BIOS 의 버그 복사본(플래시 메모리에 계속 존재하는) 대신 이것을 로드할 수 있다:

```
acpi_dsdt_load="YES"
acpi_dsdt_name="/boot/DSDT.aml"
```

DSDT.aml 을 /boot 디렉터리로 복사한다.

### 11.16.6 ACPI 에서 디버그 결과 값 얻기

---

ACPI 드라이버는 매우 유연한 디버깅 도구를 제공한다. 서브시스템 세트를 설정하고 설명 레벨도 지정할 수 있다. 디버그하려는 서브시스템은 “layers”에서 지정하고 ACPI-CA 컴포넌트(ACPI\_ALL\_COMPONENTS)와 ACPI 하드웨어 지원(ACPI\_ALL\_DRIVERS)으로 나뉜다. “level”은 비트 마스크(bitmask)이기 때문에 여러 옵션을 스페이스로 나뉘어서 한번에 지정할 수 있다. 실제로 콘솔메시지가 상당히 길다면 로그를 출력하는데 시리얼 콘솔을 사용할 수 있다. 개별적인 레이어의 전체 리스트와 레벨은 acpi(4) 매뉴얼 페이지에서 찾을 수 있다.

디버깅 출력은 기본적으로 비활성되어 있다. ACPI가 커널에 컴파일되어 있고 디버깅 출력을 활성화하려면 `options ACPI_DEBUG`을 커널 설정에 추가한다. 전체적으로 활성화하기 위해 `/etc/make.conf`에 `options ACPI_DEBUG=1`을 추가할 수 있다. 모듈이라면 다음과 같이 `acpi.ko` 모듈을 다시 컴파일 한다:

```
# cd /sys/modules/acpi/acpi
&& make clean &&
make ACPI_DEBUG=1
```

`/boot/kernel`에 `acpi.ko`를 설치하고 `loader.conf`에 원하는 레벨과 레이어를 추가한다. 이 예제는 모든 ACPI-CA 컴포넌트와 ACPI 하드웨어 드라이버(CPUm LID 등)의 디버그 메시지를 활성화한다. 최소한의 설명 레벨로 에러메시지만 출력한다.

```
debug.acpi.layer="ACPI_ALL_COMPONENTS ACPI_ALL_DRIVERS"
debug.acpi.level="ACPI_LV_ERROR"
```

특정 이벤트로 발생하는 정보를 원하면 부팅한 후 레이어와 레벨을 지정하기 위해 `loader.conf`를 변경하지 않고 `sysctl`을 사용해서 특정 이벤트를 위해 시스템을 준비한다. `sysctls`는 `loader.conf`에서 조정할 수 있는 이름과 같다.

## 11.16.7 레퍼런스

ACPI에 대한 더 많은 정보는 다음 위치에서 찾을 수 있다:

- ACPI 메일링 리스트. [freebsd-acpi@FreeBSD.org](mailto:freebsd-acpi@FreeBSD.org)
- ACPI 메일링 리스트 아카이브. <http://lists.freebsd.org/pipermail/freebsd-acpi/>

- 
- 예전 ACPI 메일링 리스트 아카이브. [http://home.jp.FreeBSD.org/mail-list/acpi-jp/](http://home.jp.FreeBSD.org/mail-list/acpi-<u>jp/</u>)
  - ACPI 2.0 과 관련된 자료. <http://acpi.info/spec.htm/>
  - FreeBSD 매뉴얼 페이지: acpi(4), acpi\_thermal(4), acpidump(8), iasl(8), acpidb(8)
  - DSDT 디버깅 리소스([http://www.cpqlinux.com/acpi-howto.html#fix\\_broken\\_dsdt](http://www.cpqlinux.com/acpi-howto.html#fix_broken_dsdt) ). (예제로 Compaq 을 사용하지만 일반적으로 유용하다)

---

## 12 장 FreeBSD 부팅 과정

### 12.1 개요

컴퓨터를 시작하고 운영체제를 로드하는 프로세스를 “부트스트랩 프로세스” 또는 간단히 “부팅”이라고 한다. FreeBSD 부트 프로세스는 같은 컴퓨터에 설치되어있는 다른 운영체제를 선택하거나 같은 운영체제 또는 설치된 커널의 다른 버전을 선택할 수 있을 정도로 시스템을 시작할 때 여러 가지를 선택할 수 있는 탁월한 유연성을 제공한다.

이번 장에서는 설정할 수 있는 자세한 옵션과 FreeBSD 부트 프로세스를 어떻게 수정할 수 있는지 설명한다. 이것은 FreeBSD 커널이 시작되고 장치 탐색 그리고 `init(8)`가 시작될 때까지 발생하는 모든 사항을 포함한다. 이 상황이 언제 발생하는지 확실하지 않다면 텍스트 색상이 밝은 하얀색에서 회색으로 변할 때다.

이번 장을 읽고 다음과 같은 사항을 알 수 있다:

- FreeBSD 부트스트랩 시스템은 어떤 컴포넌트이고 어떻게 상호작용하는가
- 부트 프로세스를 제어하기 위해 FreeBSD 부트스트랩 컴포넌트에 설정할 수 있는 옵션
- `device.hints(5)`의 기본

**x86:** 이번 장에서는 인텔 x86 시스템에서 운용 중인 FreeBSD의 부트 프로세스에 대해 설명한다.

### 12.2 부팅 문제

컴퓨터를 켜고 운영체제가 시작되면 흥미있는 딜레마에 빠진다. 정의에 따르면 컴퓨터는 운영체제가 시작될 때까지 어떤 일을 할지 전혀 알지 못한다. 이것은 디스크에서 프로그램을 실행하는 것도 포함된다. 그래서 컴퓨터는 운영체제 없이 디스크에서 프로그램을 실행시키지 못하고, 디스크에 운영체제가 있다면 운영체제는 어떻게 시작되는가?

이 문제는 *Adventures of Baron Munchausen* 책의 내용과 유사하다. 주인공이 맨홀에 빠졌고 그는 부트스트랩을 잡고 어렵게 빠져 나온다. 함축해서 “부팅”이라는 용어가 된 예전 컴퓨터 용어 **부트스트랩**은 운영체제를 로드하는 메커니즘에 적용됐다.

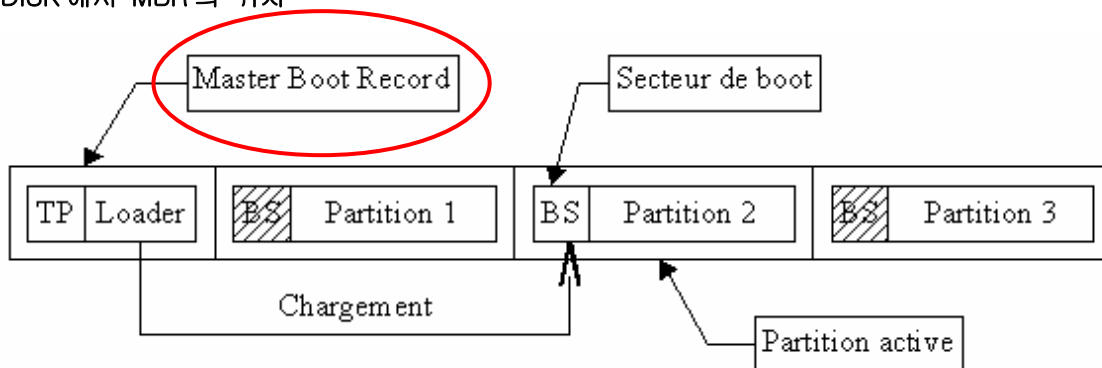
x86 하드웨어에서 기본 입/출력 시스템(BIOS)은 운영체제를 로드하는 기능을 가지고 있다. 운영체제를 로드하려고 BIOS는 디스크의 특정 위치에 있는 마스터 부트 레코드를(MBR) 찾기 위해 하드 디스크를 탐색한다. BIOS는 MBR을 로드하고 실행하기에 충분한 기능을 가지고 있고 MBR은 운영체제 로딩에 관련된 나머지 태스크를 수행할 수 있다.

## PC BIOS

AMIBIOS NEW SETUP UTILITY - VERSION 3.31a		[ Setup Help ]
Frequency/Voltage Control		
D.O.T Ranger	Private	Warning! Dynamic OverClocking is an advanced overclocking function. Any damage or risk resulted from inpropriety or overclocking are not guaranteed. Please make sure your peripherals can afford different settings.
D.O.T Mode	CPU Only	
Performance Mode	Normal	
CPU Ratio Selection	8.0x	
DRAM Frequency (Mhz)	Auto	
Spread Spectrum	Enabled	
Adjust CPU Bus Clock(Mhz)	100	
DDR Clock(Mhz)	0	
Adjust AGP/PCI Clock(Mhz)	66.66/33.33	
CPU Vcore Adjust	No	
CPU Voltage (V)		
DDR Power Voltage		
AGP Power Voltage		
F1:Help    F11:Select Item    +/-:Change Values    F7:Setup Defaults Esc:Previous Menu    Enter:Select ▸Sub-Menu    F10:Save & Exit		

하나의 운영체제를 디스크에 설치하였다면 표준 MBR 이면 된다. 이 MBR은 디스크에서 첫 번째로 부팅할 수 있는 슬라이스를 찾아서 나머지 운영체제를 로드하는 코드를 실행한다.

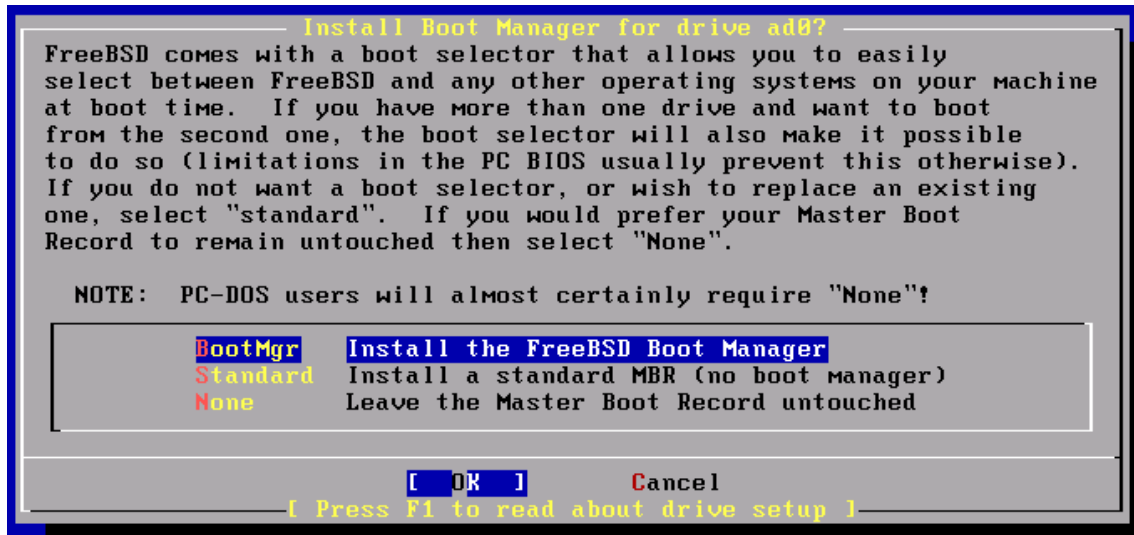
## DISK에서 MBR의 위치



---

디스크에 여러 개의 운영체제를 설치하였다면 다른 MBR 을 설치할 수 있으며 운영체제 리스트를 보고 부팅하려는 운영체제를 선택할 수 있다. FreeBSD 는 설치할 수 있는 MBR 을 가지고 있고 다른 운영체제 벤더도 각자 MBR 을 제공한다.

### FreeBSD Boot Manager



FreeBSD 부트스트랩 시스템의 나머지는 세 단계로 나누어진다. 첫 번째 단계는 컴퓨터를 특정 단계로 보내서 두 번째 단계를 실행하도록 하는 MBR 이 실행하는 단계다. 두 번째 단계는 세 번째 단계를 실행하기 전에 좀더 많은 일을 할 수 있다. 세 번째 단계는 운영체제로딩 태스크를 끝낸다. 표준 PC 는 단계 1 과 2 에서 실행할 수 있는 프로그램 크기를 제한하기 때문에 이 태스크는 세 단계로 나누어진다. FreeBSD 는 연속된 태스크를 같이 수행할 수 있도록 더 유연한 로더를 제공한다.

커널이 시작되면 장치를 탐색해서 사용하기 위해 장치를 초기화한다. 커널 부트 프로세스가 끝나면 커널은 디스크를 사용할 수 있도록 하는 유저 프로세스 init(8)에게 제어권을 넘긴다. init(8)은 파일시스템 마운트, 네트워크로 통신할 수 있도록 네트워크 카드설정 그리고 FreeBSD 시스템이 시작될 때 일반적으로 실행되는 모든 프로세스를 시작하는 유저 레벨 리소스 설정을 시작한다.

## 12.3 MBR 과 부트 단계 1, 2, 3



---

## 12.3.1 MBR, /boot/boot0

FreeBSD MBR 은 /boot/boot0 에 있다. 실제 MBR 은 FreeBSD 영역밖에 디스크의 특정 위치에 있어야 되기 때문에 이것은 MBR 복사본이다.

boot0 는 매우 단순하다. MBR 의 프로그램은 오직 512 바이트 밖에 안되기 때문에 boot0 는 매우 단순하다. 하드 디스크에 FreeBSD MBR 을 설치하고 여러 운영체제를 설치했다면 부팅할 때 다음과 비슷한 화면을 볼 수 있다:

### 예제 12-1. boot0 스크린 샷

```
F1 DOS
F2 FreeBSD
F3 Linux
F4 ??
F5 Drive 1

Default: F2
```

다른 운영체제 특히 윈도우 95, 98 은 이전 MBR 을 자신의 것으로 덮어쓴다. 이러 일이 발생했거나 이전의 MBR 을 FreeBSD MBR 로 바꾸고 싶다면 다음 명령을 이용한다:

```
# fdisk -B -b /boot/boot0 device
```

*device* 에 ad0 는 첫 번째 IDE 디스크, ad2 는 두 번째 IDE 컨트롤러의 첫 번째 IDE 디스크, da0 는 첫 번째 SCSI 디스크 등으로 부팅할 수 있는 장치를 의미한다.

그러나 여러분이 리눅스 유저이고 LILO 가 부트 프로세스를 제어하기를 원한다면 /etc/lilo.conf 파일을 FreeBSD 를 위해 편집할 수 있거나 FreeBSD 설치과정 중에 *Leave The Master Boot Record Untouched* 를 선택한다. FreeBSD 부트 매니저를 설치하였다면 리눅스로 다시 부팅하여 LILO 설정파일 /etc/lilo.conf 에 다음 옵션을 추가한다:

```
other=/dev/hdXY
table=/dev/hdb
loader=/boot/chain.b
```

---

```
label=FreeBSD
```

그러면 리눅스에서 **LILO** 를 이용하여 FreeBSD 를 부팅할 수 있다. 우리는 예제에서 드라이브 번호와 파티션을 결정할 때 *XY*를 사용한다. SCSI 드라이브를 사용한다면 */dev/hdXY*를 *XY*구문을 다시 사용하는 */dev/sdXY*와 비슷하게 읽을 수 있도록 변경하기를 원할 것이다. 같은 드라이브에 두 개의 운영체제가 있다면 *loader=/boot/chain.b*는 생략할 수 있다. 새로 변경한 내용을 시스템에 적용하도록 */sbin/lilo -v*를 실행할 수 있고 이것은 콘솔메시지로 확인할 수 있다.

### 12.3.2 단계 1: /boot/boo1 과 단계 2: /boot/boot2

개념상 첫 번째와 두 번째 단계는 같은 디스크영역에 있는 같은 프로그램의 일부분이다. 왜냐하면 공간적인 제약으로 두 개의 단계로 나누어놓았지만 항상 같이 설치된다.

이들은 boot0 에 있는 부트 슬라이스의 부트 섹터에서 찾을 수 있거나 MBR 에서 부트 프로세스를 계속 진행하도록 하는 프로그램을 찾는 프로그램이다. /boot 디렉터리의 파일은 FreeBSD 파일시스템 외부에 저장되어있는 실제 파일의 복사본이다.

boot1 역시 512 바이트 크기만 허용되기 때문에 boot1 은 매우 단순하고 boot2 를 찾아서 실행하는데 필요한 슬라이스 정보를 가지고 있는 FreeBSD disklabel 에 대해 충분히 알고 있다.

boot2 는 조금 더 복잡하고 파일을 찾을 수 있을 정도로 FreeBSD 파일시스템을 알고 있으며 커널이나 로더를 실행할 수 있도록 선택할 수 있는 간단한 인터페이스를 제공한다.

#### 예제 12-2. boot2 스크린 샷

```
>> FreeBSD/i386 BOOT
Default: 0:ad(0,a)/kernel
boot:
```

boot1 을 boot2 로 바꿀 필요가 있다면 `disklabel(8)`을 사용한다:

```
# disklabel -B disklice
```

---

ad0s1 은 첫 번째 IDE 디스크의 첫 번째 슬라이스처럼 *diskslice* 에 부팅할 수 있는 디스크와 슬라이스를 입력한다.

**Dangerously Dedicated Mode:** `disklabel(8)` 명령에서 `ad0` 같은 디스크 이름만 사용하면 슬라이스없이 `dangerously dedicated disk` 를 생성하게 된다. 이것은 원하지 않는 결과를 발생시키기 때문에 **Return** 을 누르기 전에 `disklabel(8)` 명령을 다시 체크한다.

### 12.3.3 단계 3: /boot/loader

로더는 3 단계 부트스트랩 중 마지막 단계이고 일반적으로 파일시스템의 `/boot/loader` 에 있다.

로더는 더욱 복잡한 명령어 세트로 사용자 친화적인 설정방법, 사용하기 쉬운 내장형 명령어 세트 사용 그리고 더욱 강력한 인터프리터에 의한 백업을 지향한다.

#### 12.3.3.1 로더 프로그램의 흐름

초기화하는 동안 로더는 콘솔과 디스크를 탐색해서 어떤 디스크로 부팅하는지 이해한다. 로더는 변수를 적절히 지정하고 유저명령을 스크립트나 상호작용으로 적용할 수 있는 곳에서 인터프리터를 시작한다.

로더는 변수에 적절한 기본값을 설정하는 `/boot/default/loader.conf` 를 기본적으로 읽는 `/boot/loader.rc` 를 읽은 후 이들 변수의 로컬변수로 변경하는 `/boot/loader.conf` 를 읽는다. 그 다음 `loader.rc` 는 이들 변수로 동작해서 선택한 모듈과 커널을 로딩한다.

기본적으로 로더는 키 입력을 위해 10 초 동안 기다리고 인터럽트가 없다면 커널을 부트 한다. 인터럽트가 있다면 유저는 변수를 조정하고 모든 모듈 언 로드, 모듈 로드 그리고 마지막으로 부트 또는 재 부팅할 수 있는 사용하기 쉬운 명령어 세트 프롬프트에 놓이게 된다.

#### 12.3.3.2 로더에 내장된 명령

다음은 대부분 일반적으로 사용되는 로더 명령이다. 이용할 수 있는 모든 명령에 대한 완벽

---

한 설명은 loader(8)을 본다.

autoboot *seconds*

주어진 짧은 시간 동안 인터럽트가 없다면 커널 부트로 진행한다. 이것은 카운트다운을 보여주고 기본 시간은 10 초다.

boot [-options] [*kernelname*]

커널 이름이 입력되면 즉시 입력된 옵션으로 커널 부트 한다.

boot-conf

부팅할 때 어떤 일을 할지 설정한 변수에 따라 모듈의 설정도 자동으로 같아진다. unload 를 처음으로 사용하는 것과 같고 대부분의 일반적인 kernel 변수를 변경한다.

help[*topic*]

/boot/loader.help 의 도움말을 읽어서 보여준다. *index* 라는 주제를 입력했다면 입력된 주제에 이용할 수 있는 것을 보여준다.

include *filename* ...

입력한 파일 이름의 파일로 진행되어 파일 라인을 읽고 해석한다. 에러가 발생하면 명령을 포함한 체 즉시 멈춘다.

load [-t *type*] *filename*

입력한 커널, 커널 모듈 또는 파일을 로드 한다. 파일 이름 뒤의 인수는 파일에 적용된다.

ls [-/] [*path*]

주어진 경로의 파일 리스트를 출력하거나 경로가 지정되지 않았다면 root 디렉터리를 출력한다. -/이 지정되었다면 파일 크기도 보여 준다.

---

lsdev [-v]

로드할 수 있는 모듈에서 모든 장치를 나열한다. -v를 지정하면 더 자세히 출력한다.

lsmod [-v]

로드 된 모듈 출력. -v를 지정하면 더 자세히 보여 준다.

more *filename*

각 *LINES*가 보이면 잠시 멈추어서 지정된 파일을 보여 준다.

reboot

시스템을 즉시 재 부팅한다.

set *variable*, set *variable=value*

로더의 환경변수 설정

unload

로드 된 모든 모듈을 삭제한다.

### 12.3.3.3 로더 예제

로더를 사용하는 몇 가지 예제가 여기 있다:

- 단순히 일반 커널로 부팅하지만 싱글 유저모드로 들어가려면 다음 명령을 입력한다:

**boot -s**

- 일반 커널과 모듈을 내린 후 예전(또는 다른) 커널을 로드하려면 다음 명령을 입력한다:

---

```
unload
load kernel.old
```

설치 디스크의 일반 커널을 지칭하는 `kernel.GENERIC` 이나 이전에 설치된 커널을 (예를 들어 업그레이드를 했을 때 또는 직접 커널을 설정했다면) 지칭하는 `kernel.old` 를 사용할 수 있다.

**Note:** 일반 모듈과 다른 커널을 로드하려면 다음 라인을 사용한다:

```
unload
set kernel="kernel.old"
boot-conf
```

- 커널 설정 스크립트를 로드하려면 (커널 boot-time configurator 에서 자동으로 만들어져서 일반적인 작업을 하는 스크립트) 다음 명령을 입력한다:

```
load -t userconfig_script /boot/kernel.conf
```

## 12.4 부팅 시 커널과 상호작용

커널이 로더나(보통) `boot2` 로(로더로 바이패스 한다) 로드되고 부트 플래그가 있다면 플래그를 검토해서 필요한 움직임을 조정한다.

### 12.4.1 커널 부트 플래그

여기 아주 일반적인 부트 플래그가 있다:

`-a`

커널을 초기화할 때 `root` 파일시스템 같은 장치들을 마운트할지 문의한다.

`-C`

CDROM 에서 부팅한다.

---

-c

부팅할 때 커널을 설정하는 **UserConfig** 를 실행한다.

-v

커널이 시작될 때 자세한 설명을 보여준다.

**Note:** 다른 부트 플래그에 대한 자세한 내용은 `boot(8)`을 읽는다.

## 12.5 장치 힌트

**Note:** 이것은 이전 버전에는 없는 FreeBSD 5.0 과 이후 버전의 기능이다.

시스템이 처음 시작되는 동안 `boot loader(8)`는 `device.hints(5)` 파일을 읽는다. 이 파일은 변수 같은 커널 부트정보를 저장하고 있으며 종종 장치 힌트라고 부른다. 이들 “장치 힌트”는 장치를 설정하기 위해 장치 드라이버가 사용한다.

또한 장치 힌트는 단계 3의 부트 로더 프롬프트에도 지정될 있을 것이다. 변수는 `set`을 사용하여 추가할 수 있고 `unset`으로 삭제하며 `show` 명령으로 볼 수 있다. `/boot/device.hints` 파일에 변수를 지정하여 이 설정을 무시할 수 있다. 부트 로더에서 입력한 장치 힌트는 영구적이지 않고 다음에 부팅할 때 삭제된다.

시스템이 부팅되면 `kenv(1)` 명령은 모든 변수를 덤프하는데 사용할 수 있다.

`/boot/device.hints` 파일의 구문은 주석 표시에 표준 해시“#”를 사용하여 라인당 변수를 한 개씩 지정한다. 라인 구성은 다음과 같다:

```
hint.driver.unit.keyword="value"
```

단계 3의 부트 로더 구문은 다음과 같다:

```
set hint.driver.unit.keyword=value
```

---

driver 는 장치 드라이버 이름, *unit* 은 장치 드라이버 유닛번호 그리고 *keyword* 는 힌트 키워드다. 키워드는 다음과 같은 옵션으로 이루어질 것이다:

- *at*: 장치가 붙어있는 버스 지정
- *port*: 사용할 I/O 의 시작주소 지정
- *irq*: 사용할 인터럽트 요청번호 지정
- *drq*: DMA 채널번호 지정
- *maddr*: 장치가 사용할 물리적인 메모리주소 지정
- *flags*: 장치의 변수 플래그비트 설정
- *disabled*: 1로 설정되어 있다면 장치는 비활성 된다.

장치 드라이버는 여기에 나열된 것보다 더 많은 힌트를 가지고 있으므로(또는 필요하고) 매뉴얼 페이지를 보도록 한다. 더 많은 정보는 `device.hints(5)`, `kenv(1)`, `loader.conf(5)`와 `loader(8)` 매뉴얼 페이지를 참고한다.

## 12.6 Init: 프로세스 컨트롤 초기화

커널 부팅이 끝나면 `/sbin/init` 에 있는 유저 프로세스 `init(8)` 또는 `loader` 의 `init_path` 변수에 지정한 경로에 있는 프로그램에게 제어권을 넘긴다.

### 12.6.1 자동 재 부팅 순서

자동 재 부팅은 시스템의 결함이 없는 상태에서 파일시스템을 이용할 수 있게 한다. 자동으로 재 부팅되지 않으면 `fsck(8)`은 결함을 수정하지 못하고 `init(8)`은 시스템관리자가 문제를 직접 수정하도록 시스템을 싱글 유저모드로 넘긴다.



---

## 12.6.2 싱글 유저 모드

자동 재 부팅 순서를 거쳐서 또는 유저가 `-s` 옵션으로 부팅하거나 로더에서 `boot_single` 변수를 설정하여 싱글 유저모드로 들어갈 수 있다.

또한 멀티 유저모드에서 재 부팅(`-r`)이나 `halt(-h)` 옵션 없이 `shutdown(8)`을 실행시켜서 싱글 유저모드로 들어갈 수 있다.

시스템 콘솔이 `/etc/ttys`에서 `insecure`로 설정되어있다면 싱글 유저모드를 초기화하기 전에 `root` 패스워드를 요청하는 시스템 프롬프트가 나타난다.

### 예제 12-3. /etc/ttys 에서 Insecure 콘솔

# name	getty	type	status	comments
#				
#	If console is marked "insecure", then init will ask for the root password			
#	when going to single-user mode.			
console	none		unknown of	<b>insecure</b>

**Note:** *insecure* 콘솔의 의미는 콘솔의 물리적인 보안을 비 보안으로 설정하여 `root` 패스워드를 알고 있는 사람만 싱글 유저모드로 들어갈 수 있다. 이것은 콘솔을 보안상 불안하게 운용한다는 의미가 아니다. 그러므로 안전을 생각한다면 *secure*가 아닌 *insecure*를 선택한다.

## 12.6.3 멀티 유저 모드

`init(8)`가 파일시스템을 순서대로 찾았거나 유저가 싱글 유저모드를 끝냈다면 시스템의 리소스 설정이 시작되는 멀티 유저모드로 들어간다.

### 12.6.3.1 리소스 설정(rc)

리소스 설정 시스템은 `/etc/defaults/rc.conf`에서 기본 설정을 읽고 자세한 시스템 특성을

---

/etc/rc.conf 에서 읽는다. 그 다음 /etc/fstab 에서 지시한 파일 시스템 마운트, 네트워크 서비스 시작, 잡다한 시스템 데몬 시작 그리고 마지막으로 설치된 패키지를 시작 스크립트로 실행한다.

rc(8) 매뉴얼 페이지는 스크립트 자체를 분석할 수 있어서 시스템의 리소스 설정에 좋은 레퍼런스가 된다.

## 12.7 섯다운 순서

shutdown(8)로 섯다운 명령이 떨어지면 init(8)은 /etc/rc.shutdown 스크립트를 실행해서 모든 프로세스에게 *TERM* 신호를 보내고 시간 내에 중지하지 않는 서비스에게 *KILL* 신호를 보낸다.

전원 관리를 지원하는 아키텍처와 시스템에서 FreeBSD 머신의 전원을 끄으려면 간단히 **shutdown -p now** 명령을 사용하면 즉시 전원이 끊어진다. FreeBSD 시스템을 재 부팅만 하려면 **shutdown -r now** 를 사용한다. shutdown(8)을 실행하려면 root 나 operator 그룹의 멤버여야 된다. halt(8)와 reboot(8)명령도 사용할 수 있다. 더 많은 정보는 halt(8), reboot(8) 그리고 shutdown(8) 매뉴얼 페이지를 참고 한다.

**Note:** 전원 관리를 위해 FreeBSD 5.X 는 acpi(4)를 그리고 FreeBSD 4.X 는 apm(4) 을 지원하는 커널이나 모듈을 로드해야 된다.

---

## 13 장 유저와 기본계정 관리

### 13.1 개요

FreeBSD 는 같은 시간에 여러 유저들이 컴퓨터를 사용할 수 있도록 한다. 이들 유저 중 한 명만 모니터 앞에 앉아서 키보드를 이용할 수 있지만 다른 수많은 유저는 네트워크로 작업할 수 있다. 시스템을 이용하려는 모든 유저는 계정을 가지고 있어야 한다.

이번 장을 읽고 다음과 같은 사항을 알 수 있다:

- FreeBSD 시스템의 다양한 다른 유저계정
- 유저계정을 어떻게 추가하는가
- 유저계정을 어떻게 삭제하는가
- 유저의 전체이름이나 원하는 쉘처럼 어떻게 계정을 세세히 설정하는가
- 계정과 계정 그룹이 접근하여 사용할 수 있는 메모리와 CPU 같은 리소스를 제한하기 위해 계정 기반 제한을 어떻게 적용할 수 있는가
- 계정 관리를 쉽게 하려면 그룹을 어떻게 사용해야 되는가

이번 장을 읽기 전에 다음 사항을 이해해야 된다:

- UNIX 와 FreeBSD(3 장)의 기본을 이해하고 있어야 한다.

### 13.2 소개

시스템의 모든 접근은 계정으로 이루어지고 모든 프로세스는 유저에 의해 실행되기 때문에 유저와 계정 관리는 FreeBSD 에서 없어서는 안될 중요한 업무다.

---

FreeBSD 시스템의 모든 계정은 계정을 확인할 수 있는 명확한 정보를 가지고 있다.

#### User name

유저 이름을 login: 프롬프트에 입력한다. 유저 이름은 컴퓨터에서 유일해야 된다; 같은 이름을 가진 두 명의 유저는 없을 것이다. 유효한 유저 이름을 만드는 다양한 방법이 passwd(5)에 문서화되어 있다; 일반적으로 모든 문자는 소문자이고 8 개 또는 더 적은 문자로 이루어진 유저 이름을 사용할 수 있다.

#### Password

각 계정은 관련된 패스워드를 가지고 있다. 패스워드가 없다면 패스워드 없이 시스템을 사용할 수 있다. 이것은 보안상 좋지 않은 생각이다; 모든 계정은 패스워드를 가지고 있어야 한다.

#### User ID(UID)

UID 는 시스템의 유저가 유일함을 증명하기 위해 사용되는 0 에서 65536 까지의 번호다(4294967295 처럼 큰 UID/GID 를 사용할 수 있지만 ID 값을 예상하고 만든 소프트웨어와 심각한 문제가 발생할 수 있다). FreeBSD 는 유저를 확인하기 위해 내부적으로 UID 를 사용한다. 유저 이름을 지정해야 되는 모든 FreeBSD 명령은 작업 전에 UID 로 변경된다. 이 의미는 다른 유저 이름으로 여러 계정을 가질 수 있지만 UID 는 같다. FreeBSD 는 이런 계정을 한 명의 유저로 생각한다. 이런 일은 절대 필요 없을 것이다.

```
%  
%id  
uid=1001(rick4u) gid=1001(rick4u) groups=1001(rick4u), 0(wheel)
```

#### Group ID (GID)

GID 는 유저가 속한 주 그룹을 확인하기 위해 사용하는 0 에서 65536 까지의 번호다. 그룹은 유저의 UID 보다 그들의 GID 기반으로 리소스 접근을 제어하는 메커니즘이

---

다. 이 방법으로 설정 파일들을 줄일 수 있다. 또한 유저는 한 개 이상의 그룹에 포함될 것이다.

#### Login class

로그인 클래스는 다른 유저들을 위해 시스템을 적절히 수정할 때 추가적인 유연성을 제공하는 그룹 메커니즘으로 확장된다.

#### Password change time

기본적으로 FreeBSD 는 강제로 유저들이 주기적으로 패스워드를 변경하도록 하지 않는다. 강제로 어떤 유저나 전체 유저에게 특정 시간이 지난 후 그들의 패스워드를 변경하도록 유저 기반에서 시행할 수 있다.

#### Account expiry time

기본적으로 FreeBSD 는 계정을 말소시키지 않는다. 예를 들어 학생 계정을 가지고 있는 학교에서 시간적인 한계를 가진 계정을 만들었다면 계정이 언제 말소될지 지정할 수 있다. 계정의 디렉터리와 파일이 남아 있더라도 말소 시간이 지난 후 이 계정으로 시스템에 로그인할 수 없다.

#### User's full name

FreeBSD 에서 유저 이름은 유일함이 증명되어야 하지만 유저의 실제이름을 반영할 필요는 없다. 이 정보는 계정과 관련될 수 있다.

#### Home directory

홈 디렉터리는 유저가 시스템에 로그인 했을 때 시작되는 시스템 디렉터리의 전체 경로다. 관례상 일반적으로 /home/username 이나 /usr/home/username 에 모든 유저의 홈 디렉터리를 만든다. 유저는 개인적인 파일을 유저 홈 디렉터리에 저장하고 홈 디렉터리에는 어떤 디렉터리라도 만들 수 있다.

---

## User shell

셸은 사용자가 시스템과 상호 작용할 수 있는 기본적인 환경을 제공한다. 여러 가지 종류의 셸이 있으며 경험이 있는 유저는 그들이 선호하는 것을 선택한다.

주로 3 종류의 계정이 있다: 슈퍼 유저, 시스템 유저 그리고 일반 유저계정. 일반적으로 root 로 부르는 슈퍼 유저 계정은 제한이 없는 특권으로 시스템을 관리한다. 시스템 유저는 서비스를 실행한다. 마지막으로 일반유저 계정은 로그인해서 매일을 읽는 등 실제 사람들이 사용한다.

## 13.3 슈퍼 유저 계정

root 라고 부르는 슈퍼 유저 계정은 메일 송.수신과 시스템의 일반적인 점검 또는 프로그래밍 같은 하루하루 작업에 사용되지 않는 시스템 관리를 쉽게할 수 있도록 미리 설정되어있다.

일반 유저와 차이가 있는 슈퍼 유저는 제한 없이 운용될 수 있기 때문에 슈퍼 유저 계정을 잘못 사용하면 치명적인 결과를 유발할 수 있기 때문이다. 유저 계정은 실수로 시스템을 파괴할 수 없기 때문에 추가적인 권한이 필요하지 않다면 가능하면 일반유저를 사용하는 것이 가장 좋다.

뛰어 쓰거나 잘못된 문자는 돌이킬 수 없는 데이터 유실을 의미하기 때문에 슈퍼 유저에서는 항상 명령을 두세 번씩 체크해야 된다.

따라서 이번 장을 읽은 후 첫 번째로 해야 될 일은 일상적인 사용을 위한 일반 계정이 없다면 계정을 생성한다. 이 사항은 멀티유저나 싱글 유저에서 똑같이 적용되며 어떻게 계정을 생성하고 일반유저와 슈퍼 유저 사이를 변환하는지 설명한다.

## 13.4 시스템 계정

시스템 유저는 DNS, 메일, 웹 서버 같은 서비스를 실행하는데 사용된다. 이렇게 하는 이유

---

는 보안 때문이다: 슈퍼 유저로 모든 서비스가 실행된다면 서비스는 제한 없이 동작할 것이다.

시스템 유저의 예제는 daemon, operator, bind(도메인 네임서비스를 위한)와 news 다. 어떤 시스템관리자는 웹 서버를 실행하기 위해 httpd 를 생성한다.

nobody 는 보통 권한이 없는 시스템 유저다. 그러나 더 많은 서비스에 nobody 를 사용하게 되면 더 많은 파일과 프로세스가 이 유저로 관리되고 따라서 더 많은 권한을 유저가 가지게 된다.

### 시스템 계정 예제

```
daemon:*:1:1:Owner of many system processes:/root:/sbin/nologin
operator:*:2:5:System &:/sbin/nologin
bin:*:3:7:Binaries Commands and Source:/sbin/nologin
tty:*:4:65533:Tty Sandbox:/sbin/nologin
kmem:*:5:65533:KMem Sandbox:/sbin/nologin
games:*:7:13:Games pseudo-user:/usr/games:/sbin/nologin
news:*:8:8:News Subsystem:/sbin/nologin
man:*:9:9:Mister Man Pages:/usr/share/man:/sbin/nologin
sshd:*:22:22:Secure Shell Daemon:/var/empty:/sbin/nologin
snnsp:*:25:25:Sendmail Submission User:/var/spool/clientmqueue:/sbin/nologin
mailnull:*:26:26:Sendmail Default User:/var/spool/mqueue:/sbin/nologin
bind:*:53:53:Bind Sandbox:/sbin/nologin
uucp:*:66:66:UUCP pseudo-user:/var/spool/uucppublic:/usr/libexec/uucp/uucico
xten:*:67:67:X-10 daemon:/usr/local/xten:/sbin/nologin
pop:*:68:6:Post Office Owner:/nonexistent:/sbin/nologin
www:*:80:80:World Wide Web Owner:/nonexistent:/sbin/nologin
nobody:*:65534:65534:Unprivileged user:/nonexistent:/sbin/nologin
```

## 13.5 유저 계정

유저 계정은 시스템에 실제 사람이 접근하는 것을 주로 의미하고 이들 계정은 유저와 유저 환경을 고립하여 시스템이나 다른 유저로부터의 피해를 막고 다른 유저에게 영향을 주지 않으면서 원하는 환경을 만들 수 있다.

시스템에 접근하는 모든 사람은 유일한 유저 계정을 가지고 있다. 이것은 누가 무엇을 하는지 알 수 있고 유저들이 다른 유저의 설정이나 메일을 읽는 것을 방지하는 등을 할 수 있다.

각 유저는 시스템을 편하게 사용할 수 있도록 셸, 에디터, 키보드 조합과 언어 등을 바꿔 사용하면서 원하는 환경을 설정할 수 있다.

---

## 13.6 계정 수정

유닉스 환경에서 유저 계정을 조정하기 위해 여러 가지 명령을 사용할 수 있다. 보통 사용하는 일반적인 명령은 아래에 요약되어 있으며 사용법과 예제로 더욱 자세히 설명한다.

명령	요약
adduser(8)	새로운 유저를 추가할 때 권장하는 명령어 라인 어플리케이션.
rmuser(8)	유저를 삭제할 때 권장하는 명령어 라인 어플리케이션.
chpass(1)	유저 데이터베이스 정보를 변경하는 유연한 툴.
passwd(1)	유저 패스워드를 변경하는 간단한 명령어 라인 툴.
pw(8)	유저 계정의 모든 것을 수정하는 강력하고 유연한 툴.

### 13.6.1 adduser

adduser(8)은 새로운 유저를 추가하는 간단한 프로그램이다. 이 프로그램은 시스템의 passwd 와 group 파일에 엔트리를 생성한다. 또한 새로운 유저를 위한 홈 디렉토리를 생성하며 /usr/share/skel 에서 기본 설정파일(“. 파일”)을 복사하고 부가적으로 유저 환영메시지를 유저 메일에 보내 줄 수 있다.

초기 설정파일을 생성하려면 adduser -s -config\_create(-s는 adduser(8)의 설명을 없앤다. 기본값을 변경하려면 -v 를 사용한다.)를 사용한다. 그 다음 adduser(8)을 기본으로 설정하고 root 사용으로 인한 불안감을 해결하기 위해 첫 번째 유저 계정을 생성한다.

#### 예제 13-1. adduser 설정하고 FreeBSD 4.X 에 유저 추가

```
# adduser -v
Use option "--silent" if you don't want to see all warnings and questions.
Check /etc/shells
Check /etc/master.passwd
Check /etc/group
Enter your default shell: csh date no sh tcsh zsh [sh]: zsh
Your default shell is: zsh -> /usr/local/bin/zsh
Enter your default HOME partition: [/home]:
Copy dotfiles from: /usr/share/skel no [/usr/share/skel]:
Send message from file: /etc/adduser.message no
[/etc/adduser.message]: no
Do not send message
Use passwords (y/n) [y]: y

Write your changes to /etc/adduser.conf? (y/n) [n]: y
```



---

```
Ok, let's go.
Don't worry about mistakes. I will give you the chance later to correct any input.
Enter username [a-z0-9_-]: jru
Enter full name []: J. Random User
Enter shell csh date no sh tcsh zsh [zsh]:
Enter home directory (full path) [/home/jru]:
Uid [1001]:
Enter login class: default []:
Login group jru [jru]:
Login group is ``jru''. Invite jru into other groups: guest no
[no]: wheel
Enter password []:
Enter password again []:

Name:      jru
Password:  ****
Fullname:  J. Random User
Uid:       1001
Gid:       1001 (jru)
Class:
Groups:    jru wheel
HOME:      /home/jru
Shell:     /usr/local/bin/zsh
OK? (y/n) [y]: y
Added user ``jru''
Copy files from /usr/share/skel to /home/jru
Add another user? (y/n) [y]: n
Goodbye!
#
```

요약해 보면 기본 셸을 zsh(포트 컬렉션에서 찾을 수 있는 추가적인 셸)로 바꾸었고 새로운 유저에게 환영 메일은 보내지 않는다. 그 다음 우리는 설정을 저장하고 jru 라는 계정을 생성해서 jru 를 wheel 그룹으로(그래서 su(1) 명령으로 root 가 될 수 있다) 만들었다.

**Note:** 입력하는 패스워드는 \* 문자로도 보이지 않는다. 두번 잘못 입력하지 않게 주의한다.

**Note:** 이제부터는 adduser(8)를 인수없이 그냥 사용하고 기본값으로 변경할 필요가 없다. 프로그램이 기본값으로 변경할지 물어본다면 프로그램을 빠져나가고 -s 옵션으로 다시 시도한다.

---

### 예제 13-2. FreeBSD 5.X 에서 유저 추가

```
# adduser
Username: jru
Full name: J. Random User
Uid (Leave empty for default):
Login group [jru]:
Login group is jru. Invite jru into other groups? []: wheel
Login class [default]:
Shell (sh csh tcsh zsh nologin) [sh]: zsh
Home directory [/home/jru]:
Use password-based authentication? [yes]:
Use an empty password? (yes/no) [no]:
Use a random password? (yes/no) [no]:
Enter password:
Enter password again:
Lock out the account after creation? [no]:
Username   : jru
Password   : ****
Full Name  : J. Random User
Uid        : 1001
Class      :
Groups     : jru wheel
Home       : /home/jru
Shell      : /usr/local/bin/zsh
Locked     : no
OK? (yes/no): yes
adduser: INFO: Successfully added (jru) to the user database.
Add another user? (yes/no): no
Goodbye!
#
```

## 13.6.2 rmuser

시스템에서 유저를 완전히 삭제하기 위해 `rmuser(8)`를 사용할 수 있다. `rmuser(8)`는 다음과 같은 단계를 수행한다:

- 
- ① 유저의 crontab(1) 엔트리 삭제
  - ② 유저에 속한 at(1) 잡 삭제
  - ③ 유저가 실행한 모든 프로세스를 죽인다.
  - ④ 시스템의 로컬 패스워드 파일에서 유저 삭제.
  - ⑤ 유저의 홈 디렉터리 삭제.
  - ⑥ /var/mail 에서 유저의 메일 파일 삭제
  - ⑦ /tmp 같은 임시파일 공간에서 유저에 속한 모든 파일 삭제
  - ⑧ 마지막으로 /etc/group 에 있는 모든 그룹에서 유저이름 삭제

**Note:** 그룹에 다른 유저는 없고 그룹 이름이 유저 이름과 같다면 이 그룹은 삭제된다: 이것은 adduser(8)가 생성된 유저에게 유일한 그룹을 할당했다.

엄청난 문제를 발생시키기 때문에 rmuser(8)은 슈퍼 유저 계정을 삭제할 수 없다.

기본적으로 무엇을 하고 있는지 알려 주는 대화식 모드가 사용된다.

#### 예제 13-2. rmuser 대화식 계정 제거

```
# rmuser jru
Matching password entry:
jru:*:1001:1001::0:0:J. Random User:/home/jru:/usr/local/bin/zsh
Is this the entry you wish to remove? y
Remove user's home directory (/home/jru)? y
Updating password file, updating databases, done.
Updating group file: trusted (removing group jru -- personal group is
empty) done.
Removing user's incoming mail file /var/mail/jru: done.
Removing files belonging to jru from /tmp: done.
```

```
Removing files belonging to jru from /var/tmp: done.  
Removing files belonging to jru from /var/tmp/vi.recover: done.  
#
```

### 13.6.3 chpass

chpass(1)은 패스워드, 셸 그리고 개인적인 정보 같은 유저 데이터베이스 정보를 변경한다.

슈퍼 유저처럼 시스템관리자만 다른 유저의 정보와 패스워드를 chpass(1)로 변경할 수 있다.

유저 이름에서 옵션을 주지 않으면 chpass(1)는 에디터에 유저 정보를 보여준다. 유저가 에디터에서 빠져나갈 때 유저 데이터베이스는 새로운 정보로 업데이트된다.

**Note:** FreeBSD 5.X 에서 슈퍼 유저가 아니라면 편집기를 끝낼 때 패스워드를 물어본다.

#### 예제 13-4. 슈퍼 유저에 의한 대화식 chapps

```
#Changing user database information for jru.  
Login: jru  
Password: *  
Uid [#]: 1001  
Gid [# or name]: 1001  
Change [month day year]:  
Expire [month day year]:  
Class:  
Home directory: /home/jru  
Shell: /usr/local/bin/zsh  
Full Name: J. Random User  
Office Location:  
Office Phone:  
Home Phone:  
Other information:
```

일반 유저는 자신의 정보 중 아주 작은 부분만 수정할 수 있다.

#### 예제 13-5. 일반 유저에 의한 대화식 chpass

```
#Changing user database information for jru.  
Shell: /usr/local/bin/zsh
```

---

```
Full Name: J. Random User
Office Location:
Office Phone:
Home Phone:
Other information:
```

**Note:** chfn(1)과 chsh(1)은 ypchpass(1), ypchfn(1)과 ypchsh(1)처럼 chpass(1)에 링크된다. NIS 는 자동으로 지원되므로 명령어 전에 yp 를 지정할 필요없다. 이것이 혼란스럽다면 NIS 는 24 장에서 설명하기 때문에 걱정하지 않아도 된다.

## 13.6.4 passwd

passwd(1)는 유저에서 패스워드를 바꾸거나 슈퍼 유저에서 다른 유저의 패스워드를 변경하는 일반적인 방법이다.

**Note:** 허가되지 않은 사람이 유저가 콘솔을 잠시 비웠을 때 패스워드를 변경하는 것을 막기 위해 유저는 패스워드를 바꾸기 전에 기존 패스워드를 입력해야 된다.

### 예제 13-5. 패스워드 변경

```
% passwd
Changing local password for jru.
Old password:
New password:
Retype new password:
passwd: updating the database...
passwd: done
```

### 예제 13-6. 슈퍼 유저에서 다른 유저 패스워드 변경

```
# passwd jru
Changing local password for jru.
New password:
Retype new password:
passwd: updating the database...
passwd: done
```

**Note:** chpass(1), yppasswd(1)은 단지 passwd(1)와 연결해 주기 때문에 NIS 는 두 가지 중 하나의 명령으로 작동한다.

---

## 13.6.5 pw

pw(8)은 유저와 그룹을 생성, 삭제, 수정하고 볼 수 있는 명령어 라인 유틸리티이다. 시스템 유저와 그룹 파일의 처음부터 끝까지 모든 것을 수정할 수 있다. pw(8)은 셸 스크립트에 사용하기 적당하고 아주 강력한 명령어 라인 옵션 세트를 가지고 있지만 새로운 유저는 여기 있는 다른 명령어보다 더 복잡하다는 것을 깨닫게 될 것이다.

## 13.7 유저 제한

유저를 가지고 있다면 시스템 사용 제한의 필요성을 느낄 것이다. FreeBSD는 관리자가 개개인이 사용하는 시스템 리소스 양을 제한할 수 있는 몇 가지 방법을 제공한다. 이런 제한은 두 가지 섹션으로 나뉜다: 디스크 쿼타와 다른 리소스 제한.

디스크 쿼타는 유저에게 디스크 사용량을 제한하고 항상 사용량을 계산할 필요없이 빠르게 체크하는 방법을 제공한다. 쿼타는 16.14장에서 설명한다.

다른 리소스 제한은 CPU와 메모리 그리고 유저가 소비하는 다른 리소스 제한을 포함한다. 이들은 로그인 클래스를 사용해서 정의하고 여기서 설명한다.

로그인 클래스는 /etc/login.conf에 정의되어있다. 정확한 의미는 이번 섹션의 범위를 벗어나지만 login.conf(5) 매뉴얼 페이지에 자세하게 설명되어있다. 각 유저는 로그인 클래스에 할당되어있고 각 로그인 클래스는 관련된 로그인 기능이 설정되어 있다. 로그인 기능은 *name=value* 쌍이고, *name*은 증명에 필요하고 *value*는 이름에 따른 임의적인 문자열이다. 로그인 클래스 설정과 기능은 직선적이고 login.conf(5)에 설명되어있다.

리소스 제한은 두 가지 방법에서 실제적인 로그인 기능과 다르다. 첫째 모든 제한은 소프트웨어(현재) 하드 제한이 있다. 소프트웨어 제한은 사용자나 어플리케이션에 의해 제어 되겠지만 하드 제한보다는 높지 않을 것이다. 후자는 유저에 의해 낮춰 지겠지만 절대 높아지지 않는다. 두 번째 대부분의 리소스 제한은 전체 유저가 아닌 특정 유저의 프로세스마다 제한을 적용할 수 있다. 그러나 이들의 차이점은 로그인 기능 프레임 워크 실행이 아닌 제한을 적용할 것을 지정한다(예: 이들은 실제로 로그인 기능의 특별한 경우가 아니다).

아래는 아주 일반적으로 사용되는 리소스 제한이다(나머지 로그인 기능은 login.conf(5)에서 찾을 수 있을 것이다.)

---

### *coredumpsiz*

프로그램이 생성하는 코어파일 크기 제한은 디스크 사용량에 종속적인 다른 제한이다(예: 파일크기나 디스크 쿼타). 그렇지만 디스크 공간 사용 제한의 일반적인 방법처럼 사용된다: 유저는 종종 삭제하지 않는 코어파일을 직접 생성할 수 없기 때문에 이렇게 설정하면 거대한 프로그램의(예: **emacs**) 충돌로 인한 디스크 공간 부족을 방지할 것이다.

### *cputime*

이 옵션은 유저 프로세스가 사용하는 최대 CPU 사용시간이다. 오 동작하는 프로세스는 커널에 의해 정지 될 것이다.

**Note:** 이것은 `top(1)`과 `ps(1)`가 보여주는 몇몇 필드처럼 CPU 퍼센트가 아닌 CPU 사용 *시간*을 제한한다. 이 글을 쓰고 있을 때 전자의 제한은 불가능하고 더 비효율적이다: 아마 적절한 태스크에 컴파일러는 가끔 100%의 CPU를 사용한다.

### *filesize*

이 옵션은 유저가 사용할 파일의 최대 크기다. 디스크 쿼타와 달리 이 제한은 유저가 가지고 있는 모든 파일이 아닌 각 파일을 제한한다.

### *maxproc*

이 옵션은 유저가 사용할 프로세스의 최대 번호다. 이것은 포그라운드와 백그라운드 같은 프로세스를 포함하고 `kern.maxproc sysctl(8)`에 지정된 시스템 한계치를 넘지 못할 것이다. 또한 너무 작게 설정하였다면 유저의 생산 활동을 저해할 것이다: 여러 번 로그인 하거나 파이프 라인을 실행하는 것이 가끔 유용하다. 거대한 프로그램 컴파일처럼 특정 태스크에서도 여러 프로세스를 생성한다(예: `make(1)`와 `cc(1)` 등).

---

### *memorylocked*

이 옵션은 프로세스가 주 메모리 속에 상주하기를 요청하는 메모리의 최대 양이다 (예: `mlock(2)`). `amd(8)`같은 어떤 중요한 시스템 프로그램은 이벤트가 발생하는 동안 주 메모리에 상주하면서 문제가 있을 때 시스템에 악 영향을 주지 않는다.

### *memoryuse*

이 옵션은 프로세스가 주어진 시간동안 소비할 최대 메모리 양이다. 코어 메모리와 스왑 사용량도 포함된다. 이것은 모든 메모리 소비제한을 적용하지 않지만 권장은 선택이다.

### *openfiles*

이 옵션은 프로세스 하나가 열수있는 파일의 최대 개수다. FreeBSD 에서 파일은 소켓과 IPC 채널을 표현하는데도 사용된다. 그래서 너무 낮게 설정하지 않도록 주의한다. 시스템 전반에 걸친 제한은 `kern.maxfiles sysctl(8)`에 정의되어있다.

### *sbsize*

이 옵션은 mbufs 처럼 유저가 소비하는 네트워크 메모리 양의 제한이다. 이것은 수많은 소켓을 생성하는 오래된 DoS 공격에 대한 대응으로 시작되었지만 보통 네트워크 통신제한에 사용되었다.

### *stacksize*

이 옵션은 프로세스 스택이 커지는 최대 크기다. 이 옵션만으로 프로그램이 사용하는 메모리 양을 제한하기에 효과적이지 않았기 때문에 다른 제한과 연계하여 사용한다.

리소스를 제한할 때 주의해야 될 몇 가지가 있다. 다음은 몇 개의 일반적인 팁과 잡다한 해설이다.



- 
- 시스템이 시작될 때 `/etc/rc` 가 시작하는 프로세스는 `daemon` 로그인 클래스에 할당되어 있다.
  - 시스템의 `/etc/login.conf` 가 대부분의 제한을 적용하기에 가장 적당하지만 관리자만이 그 시스템에 적절한 것을 알 수 있다. 제한을 너무 높게 설정하면 권한을 남용할 수 있으며 반면에 너무 낮게 설정하면 생산성을 저해할 것이다.
  - X 윈도우 시스템(X11) 유저는 다른 유저에 비해 더 많은 자원을 할당해야 될 것이다. X11 은 자체적으로 수 많은 리소스를 소모하지만 유저는 동시에 더 많은 프로그램을 실행할 것이다.
  - 유저의 전체 프로세스가 아닌 각각의 프로세스에 제한을 적용할 수 있음을 기억한다. 예를 들면 `openfiles` 를 50 으로 설정하면 유저가 실행하는 각 프로세스가 50 개의 파일만 열수 있다는 의미다. 따라서 유저가 열려는 총 파일의 양은 `maxproc` 값에 `openfiles` 의 값을 곱한 것이다. 이것도 메모리 사용 제한에 적용할 수 있다.

일반적인 리소스 제한과 로그인 클래스 및 특성에 대한 더 많은 정보는 관련된 매뉴얼 페이지를 참고한다: `cap_mkdb(1)`, `getrlimit(2)`, `login.conf(5)`

## 13.8 유저 개인화

지역 설정은 시스템 관리자나 유저가 다른 언어, 문자 설정, 날짜와 시간 등의 환경을 설정한다. 이 사항은 지역 설정에서 설명한다.

## 13.9 그룹

그룹은 간단히 유저의 리스트다. 그룹은 그룹 이름과 GID(Group ID)로 확인된다. FreeBSD 에서(그리고 대부분의 다른 유닉스 시스템) 프로세스가 할 수 있는 것을 커널이 결정하기 위해 사용하는 두 가지 요소는 유저 ID 와 이 유저가 속한 그룹 리스트다. 유저 ID 와 다른 프로세스는 관련된 그룹 리스트를 가지고 있다. 유저나 프로세스의 그룹 ID 를 들어본 적이 있을 것이다; 대부분 이것은 리스트의 첫 번째 그룹을 의미한다.

그룹 이름의 그룹 ID 맵은 `/etc/group` 이다. 이 파일은 4 개의 콜론으로(:)나누어진 필드로

---

되어 있는 평범한 텍스트 파일이다. 첫 번째 필드는 그룹 이름, 두 번째는 암호화된 패스워드, 세 번째는 그룹 ID 그리고 네 번째는 콤마(,)로 나누어진 멤버 리스트다. 이 파일은 직접(물론 구문에러만 만들지 않는다면) 안전하게 편집할 수 있다. 더 정확한 구문에 대한 설명은 group(5) 매뉴얼 페이지를 본다.

/etc/group 을 직접 편집하지 않으려면 그룹추가와 편집에 pw(8) 명령을 사용할 수 있다. 예를 들면 teamtwo 라는 그룹을 추가하고 확인하는데 사용할 수 있다:

#### 예제 13-7. pw(8)를 사용해서 그룹 추가하기

```
# pw groupadd teamtwo
# pw groupshow teamtwo
teamtwo:*:1100:
```

위의 번호 1100은 그룹 teamtwo 의 그룹 ID 다. 지금 teamtwo 는 멤버를 가지고 있지 않기 때문에 별로 쓸모가 없다. jru 를 teamtwo 그룹으로 초대해서 변경해 보자.

#### 예제 13-8. pw(8)를 사용해서 유저를 그룹에 추가하기

```
# pw groupmod teamtwo -M jru
# pw groupshow teamtwo
teamtwo:*:1100:jru
```

인수 *-M* 옵션은 콤마(,)로 나누어진 그룹 멤버인 유저 리스트다. 이전 섹션에서 우리는 각 유저의 패스워드 파일도 그룹을 포함해야 된다고 설명했다. 후자는(유저) 시스템에 의해 자동으로 그룹 리스트에 추가된다; pw(8)에 *groupshow* 명령을 사용할 때 유저는 멤버로 나타나지 않지만 정보가 id(1)나 비슷한 툴로 질의될 때는 나타난다. 다시 말해서 pw(8)은 /etc/group 파일만 수정한다; 이 툴은 절대 /etc/passwd 에서 추가적인 데이터를 읽으려고 하지 않는다.

#### 예제 13-9. 그룹 멤버를 결정할 때 id(1) 사용하기

```
% id jru
uid=1001(jru) gid=1001(jru) groups=1001(jru), 1100(teamtwo)
```

보았듯이 jru 는 jru 와 teamtwo 그룹의 멤버이다.

pw(8)에 대한 더 많은 정보는 매뉴얼 페이지를 참고하고 /etc/group 포맷에 관한 더 많은 정보는 group(5) 매뉴얼 페이지를 참고한다.

---

## 14 장 보안

### 14.1 개요

이번 장에서는 일반적인 정책 중 가장 최우선인 시스템 보안 개념과 FreeBSD 의 몇 가지 발전된 주제를 소개한다. 여기서 다루는 많은 주제는 시스템에 적용할 수 있고 일반적인 인터넷 보안에도 적용할 수 있다. 인터넷의 모든 사람들이 여러분들의 친절한 이웃이기를 원하겠지만 더 이상 안전한 곳이 아니다. 시스템 보안은 데이터, 지적 재산, 시간 등을 해커 같은 사람으로부터 지키는 필수요소다.

FreeBSD 는 시스템과 네트워크의 보안을 강화하기 위해 여러 유틸리티와 메커니즘을 제공한다.

이번 장을 읽고 다음 사항을 알 수 있다:

- FreeBSD 에 고려할 기본적인 시스템 보안개념
- FreeBSD 에서 이용할 수 있는 DES 와 MD5 같은 다양한 암호 메커니즘
- one-time 패스워드 인증은 어떻게 설정하는가
- FreeBSD 5.0 이전 릴리즈에 **KerberosIV** 는 어떻게 설정하는가
- FreeBSD 5.0 이후 릴리즈에 **Kerberos5** 를 어떻게 설정하는가
- IPFW 을 사용한 방화벽 생성
- IPsec 를 설정하고 FreeBSD/Windows 머신 사이에 VPN 은 어떻게 생성하는가
- FreeBSD 에서 SSH 를 사용하기 위해 **OpenSSH** 는 어떻게 설정하는가
- 파일시스템 ACL 은 무엇이고 어떻게 사용하는가

- 
- FreeBSD 보안 권고를 어떻게 이용하는가

이번 장을 읽기 전에 다음 내용을 알고 있어야 된다:

- FreeBSD 와 인터넷의 기본.

## 14.2 소개

보안은 시스템 관리자가 해야 될 업무의 시작과 끝이다. 모든 BSD 유닉스와 멀티유저 시스템이 고유한 보안 메커니즘을 가지고 있지만 유저들이 "정직"하도록 추가적인 보안 메커니즘을 구축하고 관리하는 일이 시스템 관리자의 가장 큰 업무일 것이다. 머신은 보안에 관심을 가진 만큼 안전해지지만 보안은 편리함과 상반된다. 보통 유닉스 시스템은 동시에 수많은 프로세스를 운용할 수 있고 이들 프로세스 중 대부분은 서버로 운용된다. 이 말은 외적인 요소와 연결되어 통신할 수 있음을 의미한다. 과거의 미니 컴퓨터와 메인 프레임은 오늘날의 데스크톱이 되고 컴퓨터가 네트워크와 인터넷에 이용됨으로 보안 문제가 더욱 커지고 있다.

보안은 층으로 이루어진 양파껍질 같은 접근을 통하여 가장 완벽하게 수행된다. 간결하게 말해서 해야 될 일은 사용하기 편할 정도의 많은 보안계층을 만들고 침입에 대비하여 주의 깊게 시스템을 살펴보는 것이다. 지나치게 보안사항을 많이 만들거나 보안 메커니즘의 가장 중요한 관점 중 하나인 탐색에 방해가 되기를 원하지 않을 것이다. 예를 들어 각각의 시스템 바이너리에 schg 플래그(chflags(1))를 설정하면 안 된다. 왜냐하면 일시적으로는 바이너리들을 보호하겠지만 공격자들에 의한 변화를 쉽게 잡아내지 못하여 결국 보안시스템이 공격자들을 탐지하지 못한다.

### 다 계층 보안 레이어



---

시스템 보안은 시스템다운이나 시스템을 이용할 수 없도록 시도하는 것을 포함하여 다양한 공격 양상과 관련이 있다. 보안문제는 몇 개의 카테고리로 나뉘어진다:

- ① 서비스 거부 공격
- ② 사용자 계정 획득
- ③ 접근할 수 있는 서버를 통해 root 권한 획득
- ④ 유저 권한을 통한 root 권한 획득
- ⑤ 백 도어 생성

서비스 거부 공격은 머신에 필요한 리소스를 빼앗는 행위다. 전형적인 DoS 공격은 머신이 다운되도록 하거나 다른 말로 서버나 네트워크 스택이 반응할 수 없도록 하여 머신을 사용할 수 없도록 하는 맹목적인 메커니즘이다. 어떤 DoS 공격은 네트워크 스택에서 싱글 패킷으로 머신이 다운되는 버그를 이용한다. 후자는 커널의 버그를 패치하여 해결할 수 있다. 서버에 대한 공격은 종종 시스템에 악의적인 상황을 초래하는 서버 부하를 적절히 제한하는 옵션으로 해결할 수 있다. 맹목적인 네트워크 공격은 해결하기 어렵다. 예를 들어 스푸핑 (spoof) 공격은 멈추기가 거의 불가능하여 인터넷으로부터 시스템을 계속해서 끊어놓는다. 머신을 다운시킬 수 없겠지만 인터넷 연결을 집중적으로 공격할 수 있다.

유저 권한 획득은 DoS 공격보다 더 빈번하다. 많은 시스템 관리자는 아직도 표준 telnetd, rlogind, rshd 와 ftp 서버를 머신에서 운용한다. 기본적으로 이러한 서버는 연결을 암호화해서 운용하지 않는다. 그 결과 다소간의 유저를 가지고 있다면 원격에서(시스템에 로그인하는 아주 일반적이고 편리한 방법) 시스템에 로그인하는 유저 중 한 명 이상의 유저가 패스워드를 스니프(sniff) 당할 것이다. 주의 깊은 시스템 관리자는 로그인에 성공한 유저라도 의심스러운 소스를 찾으려고 원격접근 로그를 분석할 것이다.

한번이라도 공격자가 유저 계정으로 접근했다면 공격자가 항상 root 권한을 획득할 수 있다고 가정할 수 있다. 그러나 사실은 보안상 잘 관리되는 시스템이라도 root 권한을 획득하기 위해 유저 계정에 접근해야 되는 것은 아니다. root 에 접근하지 못한 공격자는 그의 흔적을 숨기지 못하고 자신이 획득한 유저 파일을 삭제하는 외에 머신을 다운시키지 못하기 때문에 이 차이는 중요하다. 유저는 시스템 관리자의 경고를 무시하는 경향이 있기 때문에 유저 계정 획득은 매우 빈번하다.

---

시스템 관리자는 잠재적으로 머신에서 root 를 깰 수 있는 다양한 방법이 있음을 생각해야 된다. 공격자가 root 패스워드를 알 수 있고 root 계정으로 실행 중인 서버의 버그를 찾아서 네트워크를 통해 그 서버에 연결하여 root 를 깰 수도 있다. 또한 공격자가 유저 계정을 획득하였다면 root 를 깰 수 있는 suid root 프로그램의 버그를 알고 있을 것이다. 공격자가 머신에서 root 를 깨는 방법을 찾았다면 공격자는 백 도어를 설치할 필요가 없을 것이다. 그러나 root 취약점들은 공격자가 자신의 흔적을 지우기 위해 많은 노력이 필요하기 때문에 대부분의 공격자들은 백 도어를 설치한다. 백 도어는 공격자가 쉽게 root 로 다시 시스템에 접근할 수 있는 방법을 제공하지만 영리한 시스템 관리자에게는 그들의 침투를 쉽게 알 수 있는 방법을 제공한다. 공격자가 처음 공격한 취약점을 막을 수도 있기 때문에 백 도어를 설치하지 못하게 하는 것이 실제로 보안에 나쁜 영향을 미칠 수도 있다.

보안 개선 사항은 다양한 층으로 된 양파 껍질처럼 접근할 수 있는 방법이어야 되고 다음과 같이 분류할 수 있다:

- ① root 와 스태프(staff) 계정 보안
- ② root 로 실행되는 서버와 suid/sgid 바이너리 보안
- ③ 유저 계정 보안
- ④ 패스워드 파일 보안
- ⑤ 커널 코어, raw 장치와 파일시스템 보안
- ⑥ 부적절한 시스템 변화를 빨리 감지한다.
- ⑦ 편집증

이번 장의 다음 섹션은 위에서 열거한 내용에 대해 심도 있게 다룬다.

## 14.3 FreeBSD 보안 강화

---

**명령어 대 프로토콜:** 이 문서에서는 명령어와 어플리케이션에 **bold** 체를 사용했다. ssh 는 명령어일 뿐만 아니라 프로토콜에 가깝기 때문에 ssh 와 같은 보기에도 사용했다.

이 섹션은 위에서 언급한 FreeBSD 시스템 보안 개선사항에 대해 설명한다.

### 14.3.1 root 계정과 스텝 계정 보안

첫째로 root 계정의 보안을 강화하지 않았다면 스텝 계정 사용자들을 귀찮게 하지 않는다. 대부분의 시스템은 root 계정에 패스워드를 할당한다. 첫째로 패스워드는 *항상* 노출될 수 있음을 생각한다. 이 의미는 패스워드를 삭제하라는 뜻이 아니다. 패스워드는 콘솔을 통해 머신에 접속하기 위해 항상 필요하다. 이 말은 패스워드 없이 콘솔 밖에서 su(1) 명령을 사용할 수 있게 해서는 안된다. 예를 들어 /etc/ttys 파일의 pty 를 insecure 지정해서 telnet 이나 rlogin 을 사용하여 직접 root 로 로그인할 수 없도록 한다. **sshd** 같은 다른 로그인 서비스도 root 로 직접 로그인하는 것을 막는다. /etc/ssh/sshd\_config 파일에서 *PermitRootLogin* 을 *NO* 로 설정해서 직접 로그인을 막을 수 있다. FTP 같은 접근 방법은 종종 크랙(cracks) 될 수 있다. 따라서 직접 root 로그인 은 시스템 콘솔로만 접속하도록 해야 된다.

물론 시스템 관리자는 root 가 될 수 있어야 하기 때문에 몇 가지 관문을 열어 놓는다. 그러나 이러한 관문을 사용하기 위해 추가적인 패스워드 인증이 필요하도록 한다. root 가 될 수 있도록 하는 한가지 방법은 적당한 스텝 계정을 wheel 그룹에(/etc/group 에서) 추가한다. wheel 그룹에 있는 스텝 멤버만 su 로 root 가 될 수 있다. 절대 일반적인 스텝 멤버의 패스워드 엔트리에 wheel 그룹을 넣어서 root 로 접근하는 권한을 주지 않는다. 스텝 계정은 staff 그룹에 있어야 하고 /etc/group 파일에서 wheel 그룹을 추가한다. 실제로 root 접근이 필요한 스텝 멤버만 wheel 그룹에 넣어야 된다. 또한 Kerberos 같은 인증방법을 사용한다면 Kerberos 를 사용하는 root 계정의 .k5login 파일은 유저가 wheel 그룹에 있지 않아도 ksu(1)로 root 가 될 수 있다. 침입자가 패스워드 파일을 가지고 스텝 계정으로 침입했다면 wheel 메커니즘으로 침입자는 root 를 파괴할 수 있기 때문에 Kerberos 가 더 좋은 방법이다. 그러나 wheel 메커니즘이라도 가지고 있는 것이 아무런 보안 옵션을 가지고 있지 않은 것보다 좋다.

최후의 root 접근은 다른 로그인 접근 방법을 사용하고, 스텝 계정을 간접적으로 안전하게 만들기 위해 "\*"로 알려진 암호화된 패스워드를 스텝 계정에 사용한다. vipw(8) 명령을 사용

---

하면 하나의 "\*" 문자로 암호화된 패스워드의 각 인수를 바꿀 수 있다. 이 방법은 /etc/master.passwd 파일을 업데이트해서 user/password 데이터베이스가 패스워드 인증 로그인을 비활성한다.

스텝 계정 엔트리는 다음과 같다:

```
foobar:R9DT/Fa1/LV9U:1000:1000::0:0:Foo Bar:/home/foobar:/usr/local/bin/tcsh
```

이것을 다음과 같이 바꾼다:

```
foobar*:1000:1000::0:0:Foo Bar:/home/foobar:/usr/local/bin/tcsh
```

암호화된 패스워드는 절대 "\*"와 일치하지 않기 때문에 이렇게 변경하면 일반적인 로그인을 막을 수 있다. 이렇게 되면 스텝 멤버는 인증을 받기 위해 kerberos(1)나 공인/개인 키 쌍을 사용하는 ssh(1) 같은 다른 메커니즘을 사용해야 된다. Kerberos 같은 것을 사용할 때 보통 Kerberos 서버를 실행하는 머신과 데스크톱 워크스테이션은 안전해야 된다. ssh에 공인/개인키 쌍을 사용할 때 일반적인 로그인에 사용되는 머신도(보통 누군가의 워크스테이션) 보안에 안전해야 된다. 보안의 추가적인 계층은 ssh-keygen(1)로 키 쌍을 생성할 때 패스워드 보호 키 쌍으로 키 쌍을 추가할 수 있다. 스텝 계정의 패스워드를 "\*"로 하는 것도 설정해둔 안전한 접근방법으로 스텝 멤버만 로그인하는 것을 보장한다. 이 방법은 모든 스텝 멤버가 많은 침입자들이 사용하는 중요한 취약점을 달고 안전하고 암호화된 연결을 강제로 사용하게 한다: 안전하지 않은 머신에서 네트워크가 탐지되기 때문이다.

더욱 간접적인 보안 메커니즘도 보안이 강화된 서버에서 보안설정이 별로 없는 서버로 로그인 하는 것이다. 예를 들어 메인 박스에 모든 종류의 서버가 운용 중이라면 워크스테이션에는 운용하는 것이 없어야 된다. 워크스테이션이 안전하도록 운용하는 서버가 가능한 없어야 되고 패스워드 화면보호기를 운용해야 된다. 물론 워크스테이션에 물리적인 접근이 가능하면 공격자는 설치한 모든 종류의 보안을 깨뜨릴 수 있다. 이 사항을 고려해야 되지만 워크스테이션이나 서버에 물리적으로 접근할 수 없는 사람들로 부터 주로 네트워크를 통해 원격지에서 공격이 이루어진다는 사실 또한 고려해야 된다.

Kerberos 같은 것을 사용해서 한곳에서 스텝 계정의 패스워드 변경을 허용하거나 방지할 수 있고 스텝 멤버의 계정을 가지고 있는 모든 머신에 즉시 적용된다. 스텝 멤버 계정이 획득 당했다면 획득 당한 계정의 패스워드가 모든 머신에서 변경됨을 고려해야 된다. 따로 운영하는 패스워드를 N 대의 머신에서 변경하는 것은 엄청난 작업일 것이다. Kerberos로 패스워드를 다시 만들 수 있다: Kerberos는 일정 시간이 지나면 타임아웃 되도록 할 뿐 아니라



---

Kerberos 시스템은 특정 시간이 경과하면(한 달에 한번 정도) 유저가 새로운 패스워드로 변경하도록 할 수 있다.

## 14.3.2 root 로 실행되는 서버와 SUID/SGID 바이너리 보안

### 강화

신중한 시스템 관리자는 많지도 적지도 않게 오직 필요한 서버만 운용한다. 추가적인 서버는 종종 상당한 버그를 가지고 있다. 예를 들면 예전 버전의 **imapd** 나 **popper** 를 운용하는 것은 모든 사람들에게 완벽한 root 티켓을 주는 것과 같다. 주의 깊게 체크하지 않은 서버는 절대 운용하지 않는다. 많은 서버는 root 로 작동하지 않아도 된다. 예를 들어 **ntalk**, **comsat** 와 **finger** 데몬은 특수한 유저 *sandboxes*로 운용할 수 있다. sandbox 는 완벽하지 않아서 수많은 문제를 수반하겠지만 양파 껍질처럼 안전한 접근방식을 가지고 있다: 누군가 sandbox 로 운용 중인 서버를 깰 수 있다면 그들은 sandbox 만 파괴할 수 있다. 더 높은 층에 대한 공격은 성공하기 어렵기 때문에 공격자의 성공은 보잘것 없어진다. root 보안 취약점은 사실상 기본적인 서버를 포함해서 root 로 운용중인 모든 서버에서 역사적으로 발견되어 왔다. **telnetd**, **rshd** 나 **rlogind** 를 통해 로그인할 수 없고 **sshd** 로만 사람들이 로그인할 수 있다면 이들 서비스를 종료시킨다.

FreeBSD 는 이제 기본적으로 **ntalkd**, **comsat** 과 **finger** 를 sandbox 에서 운용한다. sandbox 로 운용할 다른 서버는 아마 **named(8)**가 될 것이다. */etc/defaults/rc.conf* 에는 **named** 를 sandbox 로 운용하기 위한 인수가 주석 처리되어 있다. 새로운 시스템을 설치하였거나 기존 시스템을 업그레이드 함에 따라 이렇게 *sandboxes* 로 사용되는 특수 유저 계정이 설치되지 않았을 것이다. 신중한 시스템 관리자는 가능하면 서버에 *sandboxes* 를 사용하려고 할 것이다.

일반적으로 *sandboxes* 로 운용되지 않는 다른 서버들이 있다: **sendmail**, **popper**, **imapd**, **ftpd** 와 그 외 다른 서버들이다. 이들 서버 중 어떤 서버를 설치하는 것은 시스템 관리자가 일반적으로 하는 작업보다 더 많은 작업이 필요하다. 이러한 서버는 root 로 운용해야 되고 보안상 문제가 될만한 곳을 찾는 다른 메커니즘을 사용해야 될 것이다.

시스템에서 가능성이 가장 큰 다른 root 보안 취약점은 시스템에 설치되어 있는 **suid-root** 와 **sgid** 바이너리이다. 대부분의 **rlogin** 처럼 이들 바이너리는 */bin*, */sbin*, */usr/bin* 또는 */usr/sbin* 에 존재한다. 반면 100% 안전하지 않지만 시스템에 기본적으로 설치된 **suid** 와 **sgid** 바이너리는 신뢰할만하다. 아직도 root 보안 취약점이 한번씩 이들 바이너리에서 발견

---

되고 있다. **xterm**을(전형적인 **suid**) 공격하기 쉽게 만드는 **root** 보안 취약점이 1998년에 *Xlib*에서 발견되었다. 편안함을 추구하기 보다 안전이 중요하기 때문에 신중한 시스템 관리자는 **suid** 바이너리를 스텝들만 실행하고 스텝들만 접근할 수 있도록 제한하여 **nobody**가 사용하는 **suid** 바이너리는 제거(**chmod 000**)한다. 화면에 디스플레이가 필요가 없는 서버는 보통 **xterm** 바이너리가 필요 없다. **sgid** 바이너리는 대부분 위험하다. 침입자가 **sgid-kmem** 바이너리를 깰 수 있다면 침입자는 **/dev/kmem**을 읽을 수 있기 때문에 암호화된 패스워드 파일을 읽어서 잠재적으로 계정 패스워드를 획득할 수 있다. 다른 방법으로 **kmem**을 파괴할 수 있는 침입자는 안전한 방법으로 로그인하는 유저가 사용하는 **pty**를 포함하여 **pty**를 통해 입력되는 모든 키 입력을 모니터 할 수 있다. **tty** 그룹을 깨뜨린 침입자는 대부분의 유저 **tty**를 사용할 수 있다. 유저가 터미널 프로그램이나 키보드 시뮬레이션으로 에뮬레이터를 사용한다면 침입자는 그 유저처럼 유저 터미널의 명령을 보여 줄 수 있는 데이터 스트림(**stream**) 발생기를 만들 수 있다.

### 14.3.3 유저 계정 보호

유저 계정은 일반적으로 보호하기 매우 힘들다. 스텝에 엄격한 접근 제한을 강요하고 그들의 패스워드를 "\*"로 할 수 있지만 일반적인 유저 계정에 이렇게 하는 것은 불가능하다. 충분한 제어를 하고 있다면 유저 계정을 적절하게 보호할 수 있을 것이다. 그렇지 않다면 단순히 이들 계정을 잠도 자지 못하고 모니터링 해야 된다. 유저 계정에 **ssh**와 **kerberos**를 사용하는 것은 추가적인 관리와 기술 지원이 필요하기 때문에 더욱 문제가 되지만 아직은 암호화된 패스워드 파일과 비교하면 매우 좋은 솔루션이다.

### 14.3.4 패스워드 파일 보호

유일하게 완벽한 방법은 가능한 많은 패스워드를 \*로 하고 이들 계정 접근에 **ssh**나 **kerberos**를 사용한다. 그리고 암호화된 패스워드 파일(**/etc/spwd.db**)을 **root**만 읽을 수 있고 공격자가 **root**의 쓰기 권한을 획득할 수 없더라도 침입자가 이 파일의 읽기 권한을 획득하는 것도 가능하다.

보안 스크립트는 패스워드 파일이(아래의 파일 무결성 체크를 본다) 변경되었는지 항상 체크하고 보고해야 된다.

---

### 14.3.5 커널 Core, Raw 장치와 파일시스템 보안

공격자가 root 를 꺾었다면 원하는 모든 것을 할 수 있지만 특별한 방법이 있다. 예를 들면 대부분의 현대 커널에는 패킷 스니핑(sniffing) 장치 드라이버라는 것이 커널에 내장되어 있다. FreeBSD 에서는 bpf 장치라고 부른다. 침입자는 보통 획득하려는 머신에 패킷 스니퍼를 실행한다. 대부분의 시스템은 bpf 장치를 컴파일 할 필요가 없기 때문에 침입자에게 패킷 스니퍼를 사용할 수 있는 기회를 줄 필요가 없다.

그러나 bpf 장치를 꺼 두었더라도 /dev/mem 과 /dev/kmem 을 가지고 있기 때문에 신경 써야 된다. 문제는 침입자가 raw 디스크 장치에 쓰기를 할 수 있다는 것이다. 또한 모듈 로더라고(kldload(8)) 부르는 다른 커널 기능도 있다. 고난도의 침입자는 운용 중인 커널에 자신만의 bpf 장치나 다른 스니핑(sniffing) 장치를 설치하기 위해 KLD 모듈을 사용할 수 있다. 이러한 문제를 피하기 위해 최소한 보안 레벨 1 이상의 높은 보안 레벨에서 커널을 운용해야 된다. 보안 레벨은 kern.securelevel 변수에 sysctl 로 설정할 수 있다. 보안 레벨을 1 로 설정하면 raw 장치에 쓰기 접근이 거부되고 schg 같은 특수 chflags 플래그가 강제로 설정된다. 또한 중요한 시작 바이너리 디렉터리와 스크립트 파일에 schg 플래그를 설정한다; 보안 레벨이 설정된 채로 모든 것이 실행된다. 너무 심한 보안 레벨이 설정되면 시스템을 업그레이드할 때 더욱 어려울 것이다. 높은 보안 레벨에서 시스템을 운용하더라도 모든 시스템 파일과 디렉터리에 schg 플래그를 설정하지 않는다. 다른 방법은 단순히 /와 /usr 를 읽기 전용으로 마운트하는 것이고 너무 엄격한 보호는 침입자의 중요한 흔적을 찾지 못하게 할 수 있다.

### 14.3.6 파일, 바이너리, 설정파일 등의 무결성 체크

일단 침입이 발생하면 심각해지기 전에 주요 시스템구성이나 제어파일들만 보호할 수 있다. 예를 들면 chflags 로 /와 /usr 의 대부분의 파일에 schg 비트를 설정하는 것은 역효과를 초래 할 것이다. 왜냐하면 파일을 보호하는 동안 탐지창도 닫기 때문이다. 다 계층 보안의 마지막 층이자 가장 중요한 것은 탐지다. 침입을 탐지하지 못했다면 보안계층의 나머지는 별로 유용하지 않다(또는 안전 불감증으로 더 나쁠 것이다). 양파 보안계층의 반은 공격자를 막기보다 속도를 줄이는 것이고 그가 움직이는 만큼 잡을 수 있는 탐지기회를 주기 위해서이다.

침입을 탐지하기 위해 가장 좋은 방법은 변경되거나 실수로 만들어진 또는 예상하지 못한 파일을 찾는 것이다. 변경된 파일을 찾는 가장 좋은 방법은 접근이 제한된(가끔 중앙화 된) 다른 시스템에서다. 추가적으로 보안이 강화되어 제한된 시스템에 보안 스크립트를 추가하

---

는 것은 가능한 공격을 대부분 차단하기 때문에 중요하다. 최대의 효과를 얻으려면 중요한 머신에 접근할 때 머신의 NFS 를 읽기 전용으로 공유하거나 접근이 제한된 머신에서 다른 머신에 ssh 로 접근할 수 있도록 ssh 키 쌍을 설정해야 된다. 이것이 네트워크 트래픽을 제외하고 NFS 가 최소한으로 노출되는 방법이다. 또한 사실상 탐색되지 않는 각 클라이언트 박스에서 파일시스템을 모니터 할 수 있다. 접근이 제한된 서버가 스위치를 통하여 클라이언트 머신에 연결되어 있다면 NFS 를 사용하는 것이 가끔 더 좋은 선택이 된다. 접근이 제한된 서버가 허브를 통하여 클라이언트와 연결되어있거나 몇 개의 라우팅 계층을 통하여 연결된다면 NFS 를 사용하는 것은 매우 보안에 취약하기 때문에 ssh 를 이용하는 것이 더 좋은 선택일 것이다.

접근이 제한된 머신이 있다면 모니터 하기 위해 클라이언트 시스템에 최소한 읽기 권한이 필요하고 실제 모니터링은 스크립트를 작성해야 된다. 주어진 NFS 마운트에서 find(1)와 md(1) 같은 단순한 시스템 유틸리티로 스크립트를 작성할 수 있다. 클라이언트 머신의 파일을 md5 로 체크하는 것은 최소 하루에 한번 그리고 /etc 와 /usr/local/etc 같은 곳에서 찾을 수 있는 제어 파일테스트는 더욱 자주하는 것이 좋다. 의심스러운 것이 발견되었고 접근이 제한된 머신의 md5 기반 관련정보를 믿을 수 있다면 시스템 관리자가 이것을 체크했을 때 긴장하게 될 것이다. 좋은 보안 스크립트는 적절하지 않은 suid 바이너리와 /, /usr 같은 시스템 파티션에서 추가되거나 삭제된 파일을 체크하는 것이다.

NFS 가 아닌 ssh 를 사용할 때 보안 스크립트를 작성하는 것이 더 어렵다. 기본적으로 이들 보안 스크립트를 실행해서 결과를 보여주고 안전하게 이들 스크립트를 사용하려면 scp 바이너리가(find 같은) 클라이언트 머신에 필요하다. 클라이언트 머신의 ssh 클라이언트가 이미 공격 당했을 수도 있다. 취약한 연결에 ssh 를 사용해야 되겠지만 역시 쉬운 일은 아니다.

좋은 보안 스크립트는 유저와 스텝 멤버의 접근 설정파일의 변화를 체크하는 것이다: .rhosts, .shosts, .ssh/authorized\_keys 와 더 많은 파일은 MD5 체크 범위를 벗어나면 실패할 것이다.

거대한 유저 디스크 공간을 가지고 있다면 이들 파티션의 모든 파일까지 체크하기에는 너무 많은 시간이 필요할 것이다. 이 경우 suid 바이너리와 장치를 이들 파티션에 마운트하지 못하도록 마운트 플래그를 설정하는 것도 좋은 생각이다. nodev 와 nosuid 옵션(mount(8))을 눈 여겨 보아야 된다. 어쨌든 최소 일주일에 한번씩 스캔 해야 된다. 왜냐하면 이 계층의 목적은 침입할 수 있는지 침입할 수 없는지 효과적으로 탐지하는 것이기 때문이다.

프로세스 계정(accton(8)을 본다)은 침입이 있을 후 분석 메커니즘을 돕는 운영체제의 비교

---

적 낮은 오버헤드 기능이다. 침입이 발생한 후 파일이 손상되지 않았다면 침입자가 실제로 어떻게 시스템에 침입했는지 추적하는데 특히 유용하다.

마지막으로 보안 스크립트는 로그파일을 처리해야 되고 로그는 가능한 보호된 상태에서 생성해야 한다. 이러한 목적에 원격 syslog 가 매우 유용할 것이다. 침입자는 그의 자취를 없애려고 하고 로그파일은 시스템 관리자가 최초 침입 시간과 방법을 추적하는데 중요하다. 로그파일을 영구적으로 기록하는 방법은 시스템 콘솔을 시리얼 포트로 운용하고 안전한 머신의 모니터링 콘솔을 통해 영구적으로 정보를 모은다.

### 14.3.7 편집증

약간의 불신은 절대 해가 되지 않는다. 일반적으로 시스템 관리자는 편리함에 영향을 미치지 않는다면 개수에 관계없이 보안관련 사항을 추가할 수 있으며 편리함에 영향을 미치는 것과 동시에 보안적인 요소가 증가하는 것도 추가할 수 있다. 보안 관리자는 이러한 것을 섞어서 사용해야 된다. 여기서 알려준 방법을 사용한다면 여러분의 방법은 이 문서를 읽은 공격자에게 알려지게 된다.

### 14.3.8 서비스 거부 공격

이번 섹션은 서비스 거부 공격에 대해 다룬다. DoS 공격은 전형적인 패킷 공격이다. 현재 패킷 속임수로 네트워크를 소모시키는 공격을 방어할만한 것은 없고 보통 서버가 공격으로 다운되지 않도록 피해를 감소시킬 수는 있다.

- ① 서버 forks 제한
- ② 공격 수단의 제한(ICMP 응답 공격과 핑 브로드캐스트 등)
- ③ 커널 라우트 캐시

보통 DoS 공격은 머신이 죽을 때까지 서버 프로세스 및 파일기술자(file descriptor: 흔히 파일핸들(file handle)이라고 부르며 유닉스 운영체제는 프로세스가 사용할 수 있는 파일기술자 개수를 제한한다)와 메모리를 소비하게 한다. `inetd`(`inetd(8)`)는 이런 공격을 제한하는 몇 가지 옵션을 가지고 있지만 머신이 다운되는 것을 막을 수 없고 공격으로부터 서비스 중단을 막는 것도 보통 불가능하다. `inetd` 매뉴얼 페이지를 주의 깊게 읽고 `-c`, `-C`와 `-R` 옵션

---

션을 눈 여겨본다. `init -C` 옵션은 변조된 IP 공격을 피할 수 있기 때문에 보통 이런 옵션을 조합하여 사용한다. 어떤 단독 서버는 자체적으로 `fork` 제한 매개변수를 가지고 있다.

**Sendmail** 은 `sendmail` 의 로드제한 옵션을 사용하는 것보다 더 효과적으로 로드를 줄여주는 `-OMaxDaemonChildren` 옵션을 가지고 있다. **sendmail** 을 실행할 때 예상한 부하를 충분히 제어하도록 `MaxDaemonChildren` 매개변수를 지정하고 컴퓨터가 제어하지 못하도록 너무 높게 지정하지 않는다. `sendmail` 을 큐 모드에서(`-ODeliveryMode=queued`) 운용하고 데몬을(`sendmail -bd`) 큐의 운용에서(`sendmail -q15m`) 분리하여 실행하는 것도 현명한 생각이다. 실시간 배달을 원한다면 `-q1m` 처럼 더욱 낮은 간격으로 큐를 운용할 수 있지만 `sendmail` 의 단계별 실패를 막기 위해 `MaxDaemonchildren` 옵션을 적당한 값으로 지정한다.

**Syslogd** 는 직접 공격당 할 수 있기 때문에 가능하면 언제나 `-s` 옵션을 이용하기를 강력히 권장하고 그렇지 않으면 `-a` 옵션을 사용한다.

또한 직접 공격 받을 수 있는 **tcpwrapper** 의 역방향 `identd` 같은 역방향 연결 서비스도 (`connect-back services`) 아주 조심한다. 보통 이런 이유로 **tcpwrappers** 의 역방향 `ident` 기능을 사용하지 않을 것이다.

**Tcp wrapper:** TCP 를 기반으로 한 네트워크 서비스(`finger`, `ftp`, `telnet` 등) 요청을 받아 그 서비스를 실행하기 전에 요청한 호스트에 대해 보안상으로 필요한 검사를 해서 서비스가 실행되기 이전에 공격을 막을 수 있도록 해주는 프로그램이다.

라우터 끝에 방화벽을 두어 외부 접근에서 내부 서비스를 보호하는 것은 아주 좋은 생각이다. 이 아이디어는 LAN 밖에서의 공격을 방지하지만 네트워크 기반의 `root` 획득으로부터 내부 서비스를 보호하기에는 적합하지 않다. 항상 방화벽을 배타적으로 설정한다; 예를 들어 "방화벽은 포트 A, B, C, D 와 M-Z 를 제외한 모든 포트를 열어둔다". 이 방법으로 **named**(존의 메인 서버라면), **ntalkd**, **sendmail** 과 인터넷으로 접근할 수 있는 다른 서비스처럼 특정 서비스를 제외한 낮은 포트(1024 안의 포트)를 열어줄 수 있다. 포용적이고 호의적으로 방화벽을 설정하려면 몇 개의 서비스를 막는 것을 잊거나 새로운 내부 서비스를 추가하고 방화벽 업데이트를 잊으면 된다. 낮은 포트를 열어주지 않고 유연하게 방화벽을 운용하려면 높은 번호의 포트범위를 계속 열어줄 수 있다. FreeBSD 는 복잡한 방화벽을 쉽게 설정할 수 있도록 `net.inet.ip.portrange sysctl` 을(`sysctl -a | fgrep portrange`) 이용하여 동적으로 포트범위를 제어할 수 있다. 예를 들어 일반적으로 4000 에서 5000 사이의 첫 번째/마지막 범위와 49152 에서 65535 의 높은 포트범위를 사용하고 방화벽에서(물론 확실히 지정된 인터넷 서비스 포트를 제외하고) 4000 번 아래는 막는다.

---

일반적인 다른 DoS 공격은 스프링보드(springboard) 공격이라고 한다; 서버를 공격하기 위한 방법으로 서버가 응답하도록 하여 서버나 로컬 네트워크 또는 다른 머신의 부하를 증가시킨다. 이들 중 가장 보편적인 공격은 ICMP 핑 브로드캐스트 공격이다. 공격자는 실제로 공격하려는 소스 IP 주소를 핑 패킷에 설정하여 LAN 의 브로드캐스트 주소에 보낸다. 외곽의 라우터가 브로드캐스트 주소 핑에 대해 stomp 설정이 되어있지 않으면 LAN 은 공격 당하는 머신을 마비시키도록 변조된 소스 주소로 충분한 응답을 생성한다. 특히 공격자가 한번에 여러 개의 다른 네트워크에서 여러 개의 다른 브로드캐스트에 같은 트릭을 사용한다면 공격하려는 머신을 마비시킬 수 있다. 백 개 이상의 브로드캐스트와 20Mbit 공격이 적당하다. 두 번째로 일반적인 스프링보드 공격은 ICMP 에러 리포팅 시스템이다. ICMP 에러 응답으로 생성되는 패킷을 짜맞춰서 공격자는 서버의 입력 네트워크를 포화상태로 만들 수 있기 때문에 ICMP 응답으로 나가는 네트워크도 포화상태가 된다. 이런 종류의 공격은 mbuf 의 (네트워크 버퍼에 사용되는 구조체 이름으로 초기 크기는 kern.maxusers 에 의해 결정된다) 고갈로 서버가 다운될 수 있다. 특히 서버가 ICMP 응답을 처리하지 못한다면 쉽게 다운될 수 있다. FreeBSD 커널은 이런 종류의 피해를 줄이기 위해 *ICMP\_BANDLIM*이라는 새로운 커널 컴파일 옵션을 가지고 있다. 마지막으로 주요 스프링보드 공격은 udp 에코 서비스처럼 특정 내부 inetd 와 관련된 것이다. 서버 A 와 B 가 같은 LAN 에 있으면 공격자는 단순히 소스 주소를 서버 A 의 에코 포트에 그리고 목적지 주소를 서버 B 의 에코 포트로 설정하여 UDP 패킷을 변조한다. 두 서버는 이 패킷을 각자에게 되돌려 준다. 공격자는 이 방법에서 간단히 약간의 패킷을 추가하여 두 서버와 그 서버들의 LAN 을 과부화 상태로 만들 수 있다. 비슷한 문제가 내부 **chargen** 포트에서 나타난다. 유능한 시스템 관리자는 이런 inetd 내부 테스트 서비스를 끌 것이다.

패킷 변조 공격은 커널 라우트 캐시의 부하를 높일 때도 사용된다. *net.inet.ip.rtxpire*, *rtminexpire* 와 *rtmaxcache* sysctl 매개변수를 참고한다. 무작위로 소스 IP 를 사용하는 패킷 거부공격은 라우트 테이블에서 임시로 캐시된 라우트를 커널이 생성하도록 하고 **netstat - rna | fgrep W3** 로 볼 수 있다. 이런 라우트는 전형적으로 1600 초안에 타임아웃 된다. 커널이 캐시된 라우트 테이블이 너무 크다는 것을 발견했다면 *rtexpire* 로 동적으로 감소시키지만 *rtminexpire* 보다 절대 적어지지 않는다. 여기 두 가지 문제가 있다:

- ① 커널은 가벼운 부하상태인 서버가 갑자기 공격 당했을 때 빠르게 대응하지 못한다.
- ② 커널이 일관된 공격에서 살아남도록 *rtminexpire* 를 충분히 낮춰야 한다.

서버가 T3 나 더 좋은 인터넷에 연결되어있다면 직접 sysctl(8)로 *rtexpire* 와 *rtminexpire* 를

---

설정하는 것이 좋을 수 있다. 매개변수를 절대 0으로(머신이 다운되기를 원하지 않는다면) 설정하지 않는다. 두 매개변수를 2로 설정하면 공격으로부터 라우트 테이블을 보호하기에 충분할 것이다.

### 14.3.9 kerberos 와 SSH 접근 문제

Kerberos 와 SSH 를 사용하려고 하면 Kerberos 와 ssh 를 사용한다고 알려져 되는 몇 가지 문제가 있다. Kerberos V 는 훌륭한 인증 프로토콜이지만 kerberized 된 **telnet** 과 **rlogin** 어플리케이션에서 바이너리 스트림과 맞지 않는 버그가 있다. 또한 기본 Kerberos 는 `-x` 옵션을 사용하지 않으면 세션을 암호화하지 않는다. **ssh** 는 기본적으로 모든 것을 암호화한다.

기본적으로 암호 키를 포워딩하는 것을 제외하고 ssh 를 사용한 작업은 상당히 안전하다. 이 의미는 나머지 시스템에 접근할 수 있도록 암호 키를 가지고 있는 안전한 워크스테이션이 있고 불안정한 머신에 ssh 를 사용한다면 암호 키는 유용하다. 실제 키는 노출되지 않지만 로그인이 지속되는 동안 ssh 는 포워딩 포트를 설치하고 공격자가 불안정한 머신에서 root 를 획득했다면 다른 머신에 접속하기 위해 키를 사용하려고 포트를 사용할 수 있다.

가능하다면 스텝(staff) 로그인에 ssh 와 Kerberos 를 같이 사용하도록 권장한다. **ssh** 가 Kerberos 를 지원하도록 컴파일 할 수 있다. 이 방법은 Kerberos 로 패스워드를 보호하는 동안 잠재적으로 ssh 키가 노출되는 것을 줄인다. ssh 키는 안전한 머신에서(Kerberos 는 이 일에 적당하지 않다) 자동화된 태스크에 사용해야 된다. 또한 ssh 설정에서 키 포워딩을 끄거나 키를 특정 머신에서 로그인할 때만 사용하도록 `authorized_keys` 파일에서 `from=IP/DOMAIN` 옵션을 사용하기를 권장한다.

## 14.4 DES, MD5 와 암호화

유닉스 시스템의 모든 유저는 계정에 맞는 패스워드를 가지고 있다. 이런 패스워드는 유저와 실제 운영체제만 알고 있어야 한다. 이런 패스워드를 비밀리에 유지하기 위해 쉽게 암호화만 될 수 있고 복호화는 될 수 없는 일방적인 해시(hash)로 암호화된다. 바꿔 말하면 우리가 방금 설명한 내용은 정확히 진실은 아니다: 운영 체제는 실제로 패스워드를 알지 못하고 단지 패스워드로부터 암호화된 것을 알고 있다. 평범한 텍스트 패스워드를 아는 유일한 방법은 가능한 패스워드를 강제로 검색하는 것이다.



---

불행히 패스워드를 암호화하는 유일하게 안전한 방법은 유닉스가 데이터 암호화 표준인 DES 기반일 때였다. DES 소스 코드를 미국 이외로 수출할 수 없었기 때문에 미국에 살고 있는 사람들에게는 문제가 되지 않았지만 FreeBSD는 미국 수출법과 DES를 계속 사용하는 다른 모든 유닉스와 호환을 유지하는 방법을 찾아야 했다.

해결책은 암호화 라이브러리를 나누어서 미국 유저는 DES 라이브러리를 설치하여 DES를 사용할 수 있지만 다국적 유저는 수출할 수 있는 암호화 방법을 가지고 있다. 여기서 어떻게 FreeBSD가 기본 암호화 기법으로 MD5를 사용하게 되었는지 설명한다. MD5가 DES보다 좀더 안전하다고 생각되었기 때문에 DES를 설치하는 것은 주로 호환성 때문이다.

### 14.4.1 보안 메커니즘 확인

FreeBSD 4.4 이 전에 `libcrypt.a`는 암호화에 사용되는 라이브러리를 심볼릭으로 링크하였다. FreeBSD 4.4에서는 `libcrypt.a`가 인증해서 라이브러리에 설정할 수 있는 패스워드를 제공한다. 현재 이 라이브러리는 DES, MD5와 Blowfish 해시기능을 지원한다. 기본적으로 FreeBSD는 패스워드 암호화에 MD5를 사용한다.

어떤 암호화 기법이 FreeBSD에 설정되고 사용되는지 확인하는 방법은 상당히 쉽다. `/etc/master.passwd` 파일에서 암호화된 패스워드를 확인하는 것도 하나의 방법이다. MD5 해시로 암호화된 패스워드는 DES 해시로 암호화된 것보다 길고 `$1$`로 시작된다. `$2$`로 시작하는 패스워드는 Blowfish 해시기능으로 암호화된 것이다. DES 패스워드 문자열은 확인할 수 있는 어떠한 특수 문자도 가지고 있지 않지만 MD5 패스워드보다 짧고 `$` 문자를 포함하지 않은 64개 알파벳이 코드 되어있어서 달러 문자로(`$`) 시작하지 않고 짧은 문자열은 대부분 DES 패스워드다.

새로운 패스워드에 사용하는 패스워드 포맷은 `/etc/login.conf`에서 `"des"`, `"md5"` 또는 `"blf"`의 값을 갖는 `"passwd_fmt"` 로그인 설정으로 제어된다. 로그인 설정에 대한 더 많은 정보는 `login.conf(5)` 매뉴얼 페이지를 본다.

## 14.5. 일회용 패스워드(One-Time Password)

S/Key는 단 방향 해시기능 기반의 일회용 패스워드다. FreeBSD는 호환성을 위해 MD4 해시를 사용하지만 다른 시스템은 MD5와 DES-MAC를 이용한다. S/Key는 FreeBSD 1.1.5

---

부터 FreeBSD 기본 시스템에 편입되었고 수많은 운영체제에 채택되고 있다. S/Key 는 벨 통신 연구소의 등록상표다.

FreeBSD 버전 5.0 부터 S/Key 가 OPIE(모든 것에서 일회용 비밀번호)와 기능적으로 동일하게 되었다. OPIE 는 기본적으로 MD5 해시를 사용한다.

아래에서 3 가지 종류의 비밀번호를 설명한다. 첫째는 보통 유닉스 스타일 또는 Kerberos 비밀번호다; 우리는 이것을 유닉스 비밀번호라고 부른다. 두 번째 종류는 S/Key 키 프로그램이나 OPIE opiekey(1) 프로그램이 생성하고 keyinit 또는 opiepasswd(1) 프로그램과 로그인 프롬프트가 인증한다; 우리가 "일회용 비밀번호(one-time 비밀번호)"라고 부르는 일회용 비밀번호다. 마지막 비밀번호는 일회용 비밀번호를 만들 때 사용되는 key/opiekey 프로그램을 사용하는 비밀 비밀번호다; 우리는 이것을 "비밀 비밀번호" 또는 그냥 "비밀번호"라고 부른다.

비밀 비밀번호는 유닉스 비밀번호와 아무런 관련이 없다; 두 개를 동일하게 설정할 수 있지만 권장하지 않는다. S/Key 와 OPIE 비밀 비밀번호는 예전 유닉스 비밀번호(FreeBSD 표준 로그인 비밀번호는 128 개의 문자 길이를 가지고 있을 것이다)처럼 8 개 문자의 한계가 없이 원하는 길이만큼 사용할 수 있다. 6 이나 7 개 문자의 긴 어구로 된 비밀번호가 일반적이다. 대부분 S/Key 나 OPIE 시스템은 유닉스 비밀번호와 완벽하게 독립적으로 운영된다.

비밀번호와 달리 S/Key 와 OPIE 에 중요한 두 가지 다른 데이터가 있다. 하나는 두 개의 문자와 5 개의 숫자로 이루어진 "seed" 또는 "key"로 알려진 것이다. 다른 하나는 "반복 카운트"라고 하는 1 에서 100 까지의 번호다. S/Key 는 seed 와 비밀 비밀번호를 연결하여 일회용 비밀번호를 생성하고 반복 카운트에 지정된 만큼 MD4/MD5 해시에 적용하여 6 개의 짧은 영어단어를 결과로 출력한다. 이 6 개의 영어단어가 일회용 비밀번호다. 인증 시스템은 (PAM 이 메인 인증 시스템) 마지막으로 일회용 비밀번호를 사용한 흔적을 가지고 유저에 제공된 해시 비밀번호가 이전 비밀번호와 같다면 유저는 인증에 통과한다. 왜냐하면 성공적으로 사용된 비밀번호가 저장되어있다면 여기에 사용된 단 방향 해시는 나중에 일회용 비밀번호를 만드는 것이 불가능하기 때문이다; 반복 카운터는 성공적인 각각의 로그인후 유저와 로그인 프로그램을 동기화하기 위해 감소된다. 반복 카운트가 감소되어 1 이 되면 S/Key 와 OPIE 는 다시 초기화된다.

아래에서 설명할 각 시스템과 관련된 3 개의 프로그램이 있다. key 와 opiekey 프로그램은 반복 카운트, seed, 비밀 비밀번호 그리고 일회용 비밀번호를 생성하거나 일회용 비밀번호의 연속적인 리스트를 허용한다. keyinit 와 opiepasswd 프로그램은 각각 S/Key 와 OPIE 초기화에 쓰이고 비밀번호, 반복 카운트 또는 seed 변경에 사용된다; 이들은 비밀 비밀번호

---

구문이나 반복 카운트, seed 및 일회용 비밀번호 중 하나를 사용한다. keyinfo 와 opieinfo 프로그램은 관련된 증명파일을 확인해서(/etc/skeykeys 또는 /etc/opiekeys) 유저의 현재 반복 카운트와 seed 를 출력한다.

우리가 설명할 네 가지 종류의 사용법이 있다. 첫째는 안전한 연결을 통하여 처음으로 일회용 비밀번호를 설정하거나, 비밀번호나 seed 를 변경하는데 keyinit 또는 opiepasswd 를 사용하는 것이다. 두 번째 사용법은 안전한 연결의 key 나 opiekey 를 사용하여 안전하지 않은 연결에 keyinit 또는 opiepasswd 를 사용하는 것과 같다. 세 번째는 안전하지 않은 연결에서 key/opiekey 를 사용하는 것이다. 네 번째는 안전하지 않은 연결로 어떤 곳에 연결할 때, 적어둘 수 있거나 출력할 수 있는 여러 개의 키를 생성하기 위해 key 나 opiekey 를 사용하는 것이다.

## 14.5.1 보안 연결에서(ssh 처럼 암호화된 연결) 초기화하기

처음 S/Key 를 초기화하려면 안전한 연결로 로그인 되어있는 동안 비밀번호를 바꾸거나 seed 를 변경하고(예: 머신의 콘솔이나 ssh 를 통해) 아니면 직접 로그인 되어있는 동안 아무런 인수 없이 keyinit 명령을 사용한다:

```
% keyinit
Adding unfurl:
Reminder - Only use this method if you are directly connected.
If you are using telnet or rlogin exit with no password and use keyinit -s.
Enter secret password:          ❶
Again secret password:
ID unfurl s/key is 99 to17757    ❸
DEFY CLUB PRO NASH LACE SOFT
```

OPIE 에서는 opiepasswd 를 대신 사용한다:

```

% opiepasswd -c
[grimreaper] ~ $ opiepasswd -f -c
Adding unfurl:
Only use this method from the console; NEVER from remote. If you are using
telnet, xterm, or a dial-in, type ^C now or exit with no password.
Then run opiepasswd without the -c parameter.
Using MD5 to compute responses.
Enter new secret pass phrase: ❷
Again new secret pass phrase:
ID unfurl OTP key is 499 to4268 ❸
MOS MALL GOAT ARM AVID COED

```

❷ Enter new secret pass phrase: 또는 ❶ Enter secret password: 프롬프트에 패스워드를 입력한다. 이것은 로그인할 때 사용하는 패스워드가 아니고 일회용 로그인 키를 생성하기 위해 사용된다. ❸ ID 라인에는 특별한 인스턴스의 매개변수를 보여준다; 로그인 이름, 반복 카운트와 seed. 로그인할 때 시스템이 이들 매개변수를 기억하기 때문에 따로 기억할 필요는 없다. 마지막 라인에는 이들 매개변수 및 비밀 패스워드와 일치하는 특별한 일회용 패스워드를 입력한다; 즉시 다시 로그인 했다면 이 일회용 패스워드를 사용할 것이다.

## 14.5.2 안전하지 않은 연결에서(telnet 처럼 암호화되지 않은 연결) 초기화하기

안전하지 않은 연결에서 초기화하거나 비밀 패스워드를 변경하려면 key 나 opiekey 를 실행할 수 있는 공간을 위해 안전한 연결이 필요하다; 이것은 Macintosh 의 데스크 액세서리 형식이거나 신뢰된 머신의 쉘 프롬프트일 것이다. 또한 반복 카운트를(100 정도의 값이 좋을 것이다) 만들어야 되고 자신만의 seed 를 구성하거나 무작위로 만들어진 것을 사용할 것이다. 안전하지 않은 연결에서(초기화하려는 머신) **keyinit -s** 명령을 사용한다:

```

% keyinit -s
Updating unfurl:
Old key: to17758
Reminder you need the 6 English words from the key command.
Enter sequence count from 1 to 9999: 100
Enter new key [default to17759]:

```

---

```
s/key 100 to 17759
s/key access password:
s/key access password:CURE MIKE BANE HIM RACY GORE
```

OPIE 에는 opiepasswd 를 사용해야 된다:

```
% opiepasswd

Updating unfurl:
You need the response from an OTP generator.
Old secret pass phrase:
    otp-md5 498 to4268 ext
    Response: GAME GAG WELT OUT DOWN CHAT
New secret pass phrase:
    otp-md5 499 to4269
    Response: LINE PAP MILK NELL BUOY TROY

ID mark OTP key is 499 gr4269
LINE PAP MILK NELL BUOY TROY
```

기본 seed 를(keyinit 프로그램을 혼동되게 key 라고 부르는) 사용하려면 **Enter** 를 누른다. 그리고 접근 패스워드를 입력하기 전에 안전한 연결이나 S/Key 데스크 액세스서리로 이동해서 같은 매개변수를 입력한다:

```
% key 100 to17759
Reminder - Do not use this program while logged in via telnet or rlogin.
Enter secret password: <비밀번호 입력>
CURE MIKE BANE HIM RACY GORE
```

OPIE 는 다음 명령을 사용한다:

```
% opiekey 498 to4268
Using the MD5 algorithm to compute response.
Reminder: Don't use opiekey from telnet or dial-in sessions.
Enter secret pass phrase:
GAME GAG WELT OUT DOWN CHAT
```

---

이제 안전하지 않은 연결로 되돌아와서 관련된 프로그램이 생성한 일회용 패스워드를 복사한다.

### 14.5.3 일회용 패스워드 하나 생성하기

S/Key 나 OPIE 를 초기화하면 로그인할 때 다음과 같은 프롬프트가 나타난다:

```
% telnet example.com
Trying 10.0.0.1...
Connected to example.com
Escape character is '^]'.

FreeBSD/i386 (example.com) (ttya)

login: <username>
s/key 97 fw13894
Password:
```

OPIE 는 다음과 같이 나타난다:

```
% telnet example.com
Trying 10.0.0.1...
Connected to example.com
Escape character is '^]'.

FreeBSD/i386 (example.com) (ttya)

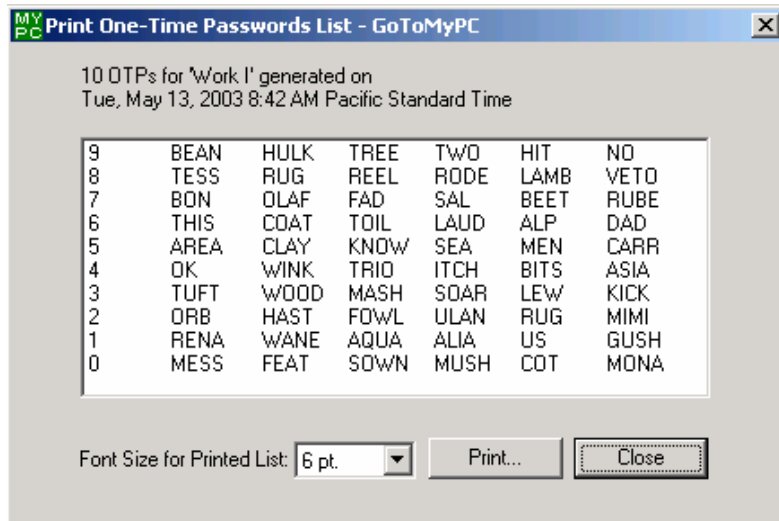
login: <username>
otp-md5 498 gr4269 ext
Password:
```

S/Key 와 OPIE 프롬프트는 유용한 기능을(여기서는 설명하지 않는다) 가지고 있다: 패스워드 프롬프트에서 **Return** 을 누르면 프롬프트는 에코로 다시 보여주기 때문에 무엇을 입력했는지 볼 수 있다. 이것은 직접 패스워드를 입력했다면 **printout** 처럼 아주 유용할 수 있다.

---

여기서 이 로그인 프롬프트에 대한 응답으로 일회용 패스워드를 생성해야 된다. 신뢰된 시스템에서 key 나 opiekey 를 실행하면 된다(DOS, 윈도우와 MacOS 에서도 다양한 버전이 있다).

#### 윈도우 버전 일회용 패스워드 생성기



여기서 반복 카운트와 seed 가 명령어 라인 옵션으로 필요하다. 로그인 프롬프트에서 잘라내서 로그인 하려는 머신에 붙일 수 있다.

신뢰할 수 있는 연결이나 시스템에서 다음 명령을 실행한다:

```
% key 97 fw13894
```

Reminder - Do not use this program while logged in via telnet or rlogin.

Enter secret password:

```
WELD LIP ACTS ENDS ME HAAG
```

OPIE 는 다음 명령을 사용한다:

```
% opiekey 498 to4268
```

Using the MD5 algorithm to compute response.

Reminder: Don't use opiekey from telnet or dial-in sessions.

Enter secret pass phrase:

```
GAME GAG WELT OUT DOWN CHAT
```

---

이제 계속 로그인할 수 있는 일회용 패스워드를 갖게 된다:

```
login: <username>
s/key 97 fw13894
Password: <echo 를 활성화하기 위해 Enter 를 누른다>
s/key 97 fw13894
Password [echo on]: WELD LIP ACTS ENDS ME HAAG
Last login: Tue Mar 21 11:56:41 from 10.0.0.2 ...
```

## 14.5.4 여러 개의 일회용 패스워드 생성

가끔 신뢰관계가 아닌 머신이나 보안상 안전한 연결에 접근해야 될 때가 있다. 이러한 경우 key 와 opiekey 명령을 사용하여 일회용 패스워드를 미리 생성해서 사용할 수 있다. 예를 들면 다음과 같이 실행한다:

```
% key -n 5 30 zz99999
Reminder - Do not use this program while logged in via telnet or rlogin.
Enter secret password: <secret password>
26: SODA RUDE LEA LIND BUDD SILT
27: JILT SPY DUTY GLOW COWL ROT
28: THEM OW COLA RUNT BONG SCOT
29: COT MASH BARR BRIM NAN FLAG
30: CAN KNEE CAST NAME FOLK BILK
```

OPIE 는 다음 명령을 사용한다:

```
% opiekey -n 5 30 zz99999
Using the MD5 algorithm to compute response.
Reminder: Don't use opiekey from telnet or dial-in sessions.
Enter secret pass phrase: <secret password>
26: JOAN BORE FOSS DES NAY QUIT
27: LATE BIAS SLAY FOLK MUCH TRIG
28: SALT TIN ANTI LOON NEAL USE
29: RIO ODIN GO BYE FURY TIC
30: GREW JIVE SAN GIRD BOIL PHI
```



---

-n 5는 순서대로 5 개 키를 요청하고 30은 반복해야 될 마지막 숫자를 지정한다. 이들은 마지막으로 사용된 낱말의 역순으로 출력된다. 의심된다면 결과를 직접 적어둘 수 있고 아니면 잘라내서 lpr 에 붙일 수 있다. 각 라인은 반복 카운트와 일회용 패스워드를 보여준다; 이런 방법을 사용하거나 패스워드를 직접 찾을 것이다.

## 14.5.5 유닉스 패스워드 사용제한

S/Key 는 호스트 이름, 유저 이름, 터미널 포트 또는 로그인 세션의 IP 주소에 따라 유닉스 패스워드 사용을 제한할 수 있다. 이들 제한은 /etc/skey.access 설정파일에서 찾을 수 있다. skey.access(5) 매뉴얼 페이지에 파일의 전체 포맷에 대한 더 많은 정보가 있고 이 파일을 사용하기 전에 알고 있어야 되는 몇 가지 보안권고도 있다.

/etc/skey.access 파일이(FreeBSD 4.x 시스템에서 기본적인) 없다면 모든 유저는 유닉스 패스워드를 사용할 수 있다. 파일이 있다면 skey.access 파일 문구에 유닉스 패스워드를 사용할 수 있도록 설정되어있는 경우를 제외하고 모든 유저는 S/Key 를 사용해야 된다.

여기 설정구문 중 아주 일반적인 3 가지를 보여주는 샘플 skey.access 설정파일이 있다.

```
permit internet 192.168.0.0 255.255.0.0
permit user fnord
permit port ttyd0
```

첫 번째 라인은(*permit internet*) 유닉스 패스워드를 사용하도록 지정된 값과 IP 소스 주소와 mask 가 일치하는 유저만 허용한다. 이것을 보안 메커니즘에 고려하지 않아도 되지만 승인된 유저들에게 그들이 안전하지 않은 네트워크를 사용하고 있으며 인증을 위해 S/key 를 사용해야 된다는 것을 알려줘야 된다.

여기서는 **fnord** 인 두 번째 라인(*permit user*)은 지정된 유저 이름이 언제나 유닉스 패스워드를 사용할 수 있도록 허용한다. 일반적으로 말하면 이 옵션은 dumb 터미널처럼 key 프로그램을 사용하지 못하거나 교육할 수 없는 사람들에게만 사용해야 된다.

세 번째 라인은(*permit port*) 유닉스 패스워드를 사용하도록 지정된 터미널 라인의 모든 유저 로그인을 허용한다; 이것은 전화 접속에 사용될 것이다.

---

OPIE 도 S/Key 처럼 로그인 세션의 IP 주소에 따라 유닉스 패스워드 사용을 제한할 수 있다. 관련된 파일은 FreeBSD 5.0 와 새로운 시스템에서 기본적으로 가지고 있는 `/etc/opieaccess` 이다. 이 파일에 대한 더 많은 정보와 사용할 때 알아야 되는 보안권고에 대해 `opieaccess(5)`를 체크한다.

여기 샘플 `opieaccess` 파일이 있다:

```
permit 192.168.0.0 255.255.0.0
```

지정된 값과 IP 소스 주소와(스푸핑(spoofing)에 공격 받기 쉬운) mask 가 일치하는 유저만 언제나 유닉스 패스워드를 사용할 수 있다.

`opieaccess` 의 모든 룰에 맞지 않으면 기본값은 OPIE 가 아닌 모든 로그인을 거부한다.

## 14.6 KerberosIV

Kerberos 는 유저들이 보안 서버의 서비스를 통해 직접 인증을 받을 수 있는 네트워크에 추가된 시스템/프로토콜이다. 원격 로그인, 원격 복사, 내부 보안 시스템파일 복사 및 위험성이 높은 서비스를 더 안전하게 정확히 제어할 수 있다.

다음 설명은 FreeBSD 에 배포된 Kerberos 를 어떻게 설정하는지 가이드로 사용할 수 있다. 그러나 완벽한 설명은 관련된 매뉴얼 페이지를 참고한다.

### 14.6.1 KerberosIV 설치

Kerberos 는 FreeBSD 에 추가적인 컴포넌트다. 이 소프트웨어를 설치하는 가장 쉬운 방법은 FreeBSD 를 처음 설치하는 과정 중 `sysinstall` 에서 `krb4` 나 `krb5` 배포본을 선택하면 된다. 이 배포본은 Kerberos 요소 'eBones'(KerberosIV)이나 'Heimdal'(Kerberos5)를 설치한다. 이런 요소는 미국/캐나다 밖에서 개발되어 미국의 암호화 코드 수출제한이 있던 시기에 두 나라밖의 시스템 관리자들이 사용할 수 있었기 때문에 포함되었다.

대신 MIT Kerberos 요소는 포트 컬렉션 [security/krb5](#) 에서 사용할 수 있다.

---

## 14.6.2 최초의 데이터베이스 생성

이 사항은 Kerberos 서버에만 해당한다. 첫째 어떠한 예전 Kerberos 데이터베이스도 가지고 있지 않아야 된다. /etc/kerberosIV 로 디렉터리를 변경하고 다음 파일들만 있는지 체크한다:

```
# cd /etc/kerberosIV
# ls
README      krb.conf      krb.realms
```

다른 파일이(principal.\*이나 master\_key 같은) 있다면 예전 Kerberos 데이터베이스를 삭제하기 위해 kdb\_destroy 명령을 사용하거나 Kerberos 가 실행 중이 아니라면 간단히 이들 파일을 삭제하면 된다.

이제 Kerbers 영역을 정의하기 위해 krb.conf 와 krb.realms 파일을 수정한다. 우리의 경우 영역은 *EXAMPLE.COM*이고 서버는 grunt.example.com 이다. 우리는 다음 krb.conf 파일을 수정하거나 생성했다:

```
# cat krb.conf
❶EXAMPLE.COM
❷EXAMPLE.COM ❸grunt.example.com admin server
CS.BERKELEY.EDU okeeffe.berkeley.edu
ATHENA.MIT.EDU kerberos.mit.edu
ATHENA.MIT.EDU kerberos-1.mit.edu
ATHENA.MIT.EDU kerberos-2.mit.edu
ATHENA.MIT.EDU kerberos-3.mit.edu
LCS.MIT.EDU kerberos.lcs.mit.edu
TELECOM.MIT.EDU bitsy.mit.edu
ARC.NASA.GOV trident.arc.nasa.gov
```

우리의 경우 이곳에 다른 영역은 필요 없다. 이 예제는 머신이 여러 개의 영역을 어떻게 인지하는지 보여준다.

❶번 라인은 이 시스템이 동작하는 영역의 이름이다. 다른 라인은 realm/host 엔트리를 포함하고 있다. 라인의 ❷번 내용은 영역이고 ❸번은 "키 분산 센터"처럼 동작하는 영역의 호

---

스트다. 단어 *admin server* 다음의 호스트 이름이 의미하는 것은 호스트도 관리 데이터베이스 서버를 제공한다는 것이다. 이들 용어에 대한 자세한 사항은 Kerberos 매뉴얼 페이지를 참고한다.

이제 `grunt.example.com` 을 `EXAMPLE.COM` 영역에 추가하고 `EXAMPLE.COM` 영역의 `.example.com` 도메인에 있는 모든 호스트 엔트리를 추가한다. 따라서 `krb.realms` 파일은 다음과 같이 업데이트될 것이다:

```
# cat krb.realms
grunt.example.com EXAMPLE.COM
.example.com EXAMPLE.COM
.berkeley.edu CS.BERKELEY.EDU
.MIT.EDU ATHENA.MIT.EDU
.mit.edu ATHENA.MIT.EDU
```

역시 다른 영역은 이곳에 필요 없다. 이 파일은 머신이 여러 영역을 어떻게 인지하는지에 대한 예제였다.

첫 번째 라인은 특정 시스템을 영역으로 바꾼다. 나머지 라인은 특정 서브 도메인의 기본 시스템을 영역으로 어떻게 바꾸는지 보여준다.

이제 데이터베이스를 생성할 준비가 되었다. 이것은 Kerberos 서버에서만(또는 키 분산 센터) 실행하면 된다. `kdb_init` 명령을 다음과 같이 실행한다:

```
# kdb_init
Realm name [default ATHENA.MIT.EDU ]: EXAMPLE.COM
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.

Enter Kerberos master key:
```

이제 키를 저장해서 로컬 머신의 서버가 키를 로드하도록 해야 된다. `kstash` 명령을 다음과 같이 사용한다:

---

```
# kstash
```

```
Enter Kerberos master key:
```

```
Current Kerberos master key version is 1.
```

```
Master key entered. BEWARE!
```

위 명령은 암호화된 마스터 패스워드를 `/etc/kerberosIV/master_key` 에 저장한다.

### 14.6.3 모두 실행하기

Kerberos 로 각 시스템을 안전하게 만들기 위해 두 가지 중요한 요소를 데이터베이스에 추가해야 된다. 이들 이름은 `kpasswd`와 `rcmd`이고 두 요소는 각 시스템의 이름을 참조하여 생성된다.

이들 데몬 `kpasswd` 와 `rcmd` 는 다른 시스템에서 Kerberos 패스워드를 변경하고 `rcp`, `rlogin` 과 `rsh` 같은 명령을 실행할 수 있도록 한다.

이제 이런 엔트리를 추가해 보자:

```
# kdb_edit
```

```
Opening database...
```

```
Enter Kerberos master key:
```

```
Current Kerberos master key version is 1.
```

```
Master key entered. BEWARE!
```

```
Previous or default values are in [brackets] ,  
enter return to leave the same, or new value.
```

```
Principal name: passwd
```

```
Instance: grunt
```

---

<Not found>, Create [y] ? **y**

Principal: passwd, Instance: grunt, kdc\_key\_ver: 1

New Password:

Verifying password

New Password:

Random password [y] ? **y**

Principal's new key version = 1

Expiration date (enter yyyy-mm-dd) [ 2000-01-01 ] ?

Max ticket lifetime (\*5 minutes) [ 255 ] ?

Attributes [ 0 ] ?

Edit O.K.

Principal name: **rcmd**

Instance: grunt

<Not found>, Create [y] ?

Principal: rcmd, Instance: grunt, kdc\_key\_ver: 1

New Password:

Verifying password

New Password:

Random password [y] ?

Principal's new key version = 1

Expiration date (enter yyyy-mm-dd) [ 2000-01-01 ] ?

Max ticket lifetime (\*5 minutes) [ 255 ] ?

Attributes [ 0 ] ?

Edit O.K.

Principal name: <----- 아무것도 입력하지 않으면 빠져나간다.

---

## 14.6.4 서버 파일 생성

이제 각 머신에 정의된 모든 서비스 인스턴스를 추출해야 된다. 추출하기 위해 `ext_srvtab` 명령을 이용한다. 이 명령은 각 Kerberos 클라이언트의 `/etc/kerberosIV` 디렉터리로 *안전하게* 복사하거나 옮겨야 되는 파일을 생성한다. 이 파일은 각 서버와 클라이언트에 필요하고 Kerberos 운용에 중요하다.

```
# ext_srvtab grunt
Enter Kerberos master key:

Current Kerberos master key version is 1.

Master key entered. BEWARE!
Generating 'grunt-new-srvtab'....
```

위 명령은 `srvtab` 로 이름을 바꿔야 되는 임시파일을 생성하기 때문에 모든 서버가 이 파일을 로드할 수 있다. `mv(1)` 명령으로 이 파일을 최초 시스템으로 이동시킨다:

```
# mv grunt-new-srvtab srvtab
```

이 파일이 클라이언트 시스템에 있고 네트워크가 안전하지 않다고 생각되면 `client-new-srvtab` 를 이동 가능한 미디어에 복사해서 물리적으로 안전하게 이동시킨다. 클라이언트의 `/etc/kerberosIV` 디렉터리에서 `srvtab` 로 다시 이름을 바꾸고 퍼미션을 600 으로 변경한다:

```
# mv grumble-new-srvtab srvtab
# chmod 600 srvtab
```

## 14.6.5 데이터베이스 생성

이제 몇몇 유저 엔트리를 데이터베이스에 추가해야 된다. 첫째로 유저 `jane` 엔트리를 생성하자. `Kdb_edit` 명령을 다음과 같이 사용한다:

```
# kdb_edit
Opening database...

Enter Kerberos master key:

Current Kerberos master key version is 1.

Master key entered.  BEWARE!
Previous or default values are in [brackets] ,
enter return to leave the same, or new value.

Principal name: jane
Instance:

<Not found>, Create [y] ? y

Principal: jane, Instance: , kdc_key_ver: 1
New Password:          <----- 보안 패스워드를 입력한다
Verifying password

New Password:          <----- 패스워드를 다시 입력한다
Principal's new key version = 1
Expiration date (enter yyyy-mm-dd) [ 2000-01-01 ] ?
Max ticket lifetime (*5 minutes) [ 255 ] ?
Attributes [ 0 ] ?
Edit O.K.
Principal name:        <----- 아무것도 입력하지 않으면 빠져나간다
```

## 14.6.6 테스트

첫째 Kerberos 데몬을 실행해야 한다. /etc/rc.conf 를 정확히 수정했다면 재 부팅할 때 자동으로 데몬이 실행된다. 이것은 오직 Kerberos 서버에만 필요하다. Kerberos 클라이언트는 /etc/kerberosIV 디렉터리에서 필요한 것을 자동으로 수행한다.



---

```
# kerberos &
```

```
Kerberos server starting
```

```
Sleep forever on error
```

```
Log file is /var/log/kerberos.log
```

```
Current Kerberos master key version is 1.
```

```
Master key entered. BEWARE!
```

```
Current Kerberos master key version is 1
```

```
Local realm: EXAMPLE.COM
```

```
# kadmin -n &
```

```
KADM Server KADM0.0A initializing
```

```
Please do not use 'kill -9' to kill this job, use a  
regular kill instead
```

```
Current Kerberos master key version is 1.
```

```
Master key entered. BEWARE!
```

이제 위에서 만든 ID jane 의 티켓을 kinit 명령으로 얻을 수 있다:

```
% kinit jane
```

```
MIT Project Athena (grunt.example.com)
```

```
Kerberos Initialization for "jane"
```

```
Password:
```

실제로 인증을 받았다면 체크할 수 있도록 klist 를 사용하여 결과를 리스트 한다:

```
% klist
```

```
Ticket file: /tmp/tkt245
```

```
Principal: jane@EXAMPLE.COM
```

Issued	Expires	Principal
Apr 30 11:23:22	Apr 30 19:23:22	krbtgt.EXAMPLE.COM@EXAMPLE.COM

---

**kpasswd** 데몬이 Kerberos 데이터베이스 인증을 받을 수 있다면 **passwd** 를 사용해서 패스워드를 바꿀 수 있는지 확인한다:

```
% passwd
realm EXAMPLE.COM
Old password for jane:
New Password for jane:
Verifying password
New Password for jane:
Password changed.
```

### 14.6.7 su 권한 추가

Kerberos 는 root 권한이 필요한 각 유저에게 *각자의* su 패스워드를 할당할 수 있다. 이제 su(1)로 root 인증이 필요한 ID 를 추가할 수 있다. 이것은 주체(principal)와 관련된 root 인스턴스를 가지고 제어된다. **kdb\_edit** 를 사용하여 Kerberos 데이터베이스에 *jane.root* 엔트리를 생성한다:

```
# kdb_edit
Opening database...

Enter Kerberos master key:

Current Kerberos master key version is 1.

Master key entered.  BEWARE!
Previous or default values are in [brackets] ,
enter return to leave the same, or new value.

Principal name: jane
Instance: root
```

---

<Not found>, Create [y] ? y

Principal: jane, Instance: root, kdc\_key\_ver: 1

New Password: <----- 보안 패스워드를 입력한다

Verifying password

New Password: <----- 패스워드를 다시 입력한다

Principal's new key version = 1

Expiration date (enter yyyy-mm-dd) [ 2000-01-01 ] ?

Max ticket lifetime (\*5 minutes) [ 255 ] ? 12 <---- 이 값을 유지한다

Attributes [ 0 ] ?

Edit O.K.

Principal name: <----- 아무것도 입력하지 않으면 빠져나간다

이제 동작하는지 확인해 보자:

```
# kinit jane.root
```

```
MIT Project Athena (grunt.example.com)
```

```
Kerberos Initialization for "jane.root"
```

```
Password:
```

유저를 root 의 .klogin 파일에 추가해야 된다:

```
# cat /root/.klogin
```

```
jane.root@EXAMPLE.COM
```

이제 su(1)를 사용해 보자:

```
% su
```

```
Password:
```

---

그리고 어떤 결과가 나타나는지 확인한다:

```
# klist
Ticket file:    /tmp/tkt_root_245
Principal:     jane.root@EXAMPLE.COM

    Issued          Expires          Principal
May  2 20:43:12  May  3 04:43:12  krbtgt.EXAMPLE.COM@EXAMPLE.COM
```

## 14.6.8 다른 명령어 이용

이전 예제에서 *jane* 이라고 부르는 주체와 *root* 인스턴스를 만들었다. 이것은 주체와 같은 유저 이름이고 Kerberos 의 기본값이다; *<principal>.<instance>* 의 형식 *<username>.root* 는 *root* 홈 디렉터리의 *.klogin* 파일에 필요한 엔트리가 있다면 *<username>*이 *su(1)*로 *root* 가 되는 것을 허용한다.

```
# cat /root/.klogin
jane.root@EXAMPLE.COM
```

유저 개인의 홈 디렉터리에 그 형식과 같은 라인이 있다면 마찬가지로:

```
% cat ~/.klogin
jane@EXAMPLE.COM
jack@EXAMPLE.COM
```

이 설정은 *jane* 나 *jack*(*kinit* 를 통해) 인증을 가지고 있는 *EXAMPLE.COM* 의 사람이 *rlogin(1)*, *rsh(1)*, *rcp(1)*를 사용하여 *jane* 의 계정이나 이 시스템(*grunt*) 파일에 접근하는 것을 허용한다.

예를 들어 *jane* 는 이제 Kerberos 로 다른 시스템에 로그인할 수 있다:

---

```
% kinit
MIT Project Athena (grunt.example.com)
Password:
% rlogin grunt
Last login: Mon May  1 21:14:47 from grumble
Copyright (c) 1980, 1983, 1986, 1988, 1990, 1991, 1993, 1994
    The Regents of the University of California.  All rights reserved.

FreeBSD BUILT-19950429 (GR386) #0: Sat Apr 29 17:50:09 SAT 1995
```

그렇지 않으면 같은 머신에서 Jack 이 Jane 의 계정으로 로그인할 수 있다(jane 는 .klogin 파일을 위와 같이 설정하고 Kerberos 가 적용되는 사람은 null 인스턴스로 *jack* 주체를 설정한다)

```
% kinit
% rlogin grunt -l jane
MIT Project Athena (grunt.example.com)
Password:
Last login: Mon May  1 21:16:55 from grumble
Copyright (c) 1980, 1983, 1986, 1988, 1990, 1991, 1993, 1994
    The Regents of the University of California.  All rights reserved.

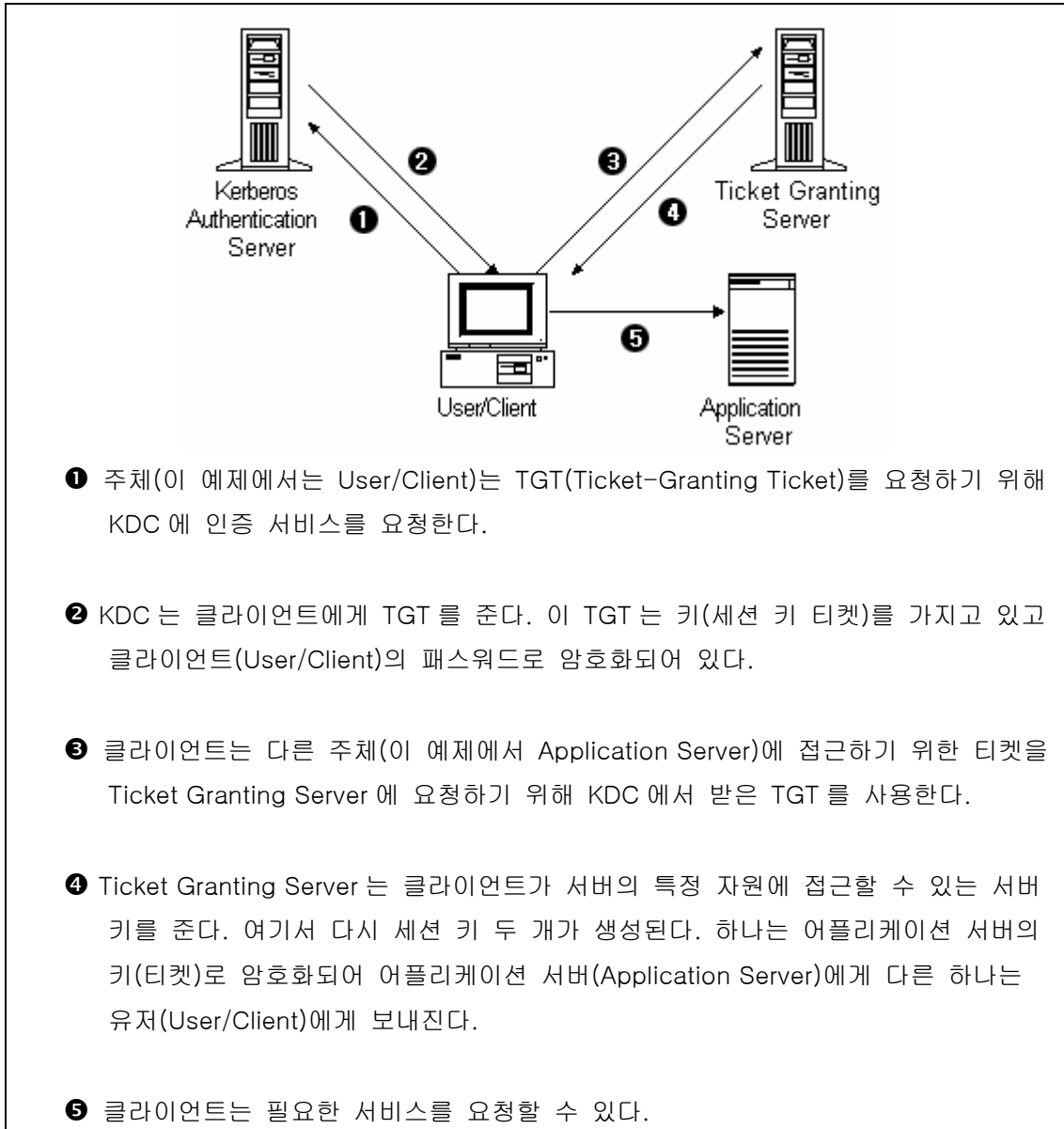
FreeBSD BUILT-19950429 (GR386) #0: Sat Apr 29 17:50:09 SAT 1995
```

## 14.7 Kerberos5

FreeBSD-5.1 이상의 모든 FreeBSD 릴리즈는 **Kerberos5** 만 지원한다. 따라서 **Kerberos5** 가 유일하게 포함된 버전이고 설정은 **KerberosIV** 와 대부분 비슷하다. 다음 정보는 FreeBSD-5.0 릴리즈 이후의 **Kerberos5** 에만 적용된다. KerberosIV 를 사용하려는 유저는 [security/krb4](#) 포트를 설치한다.

**Kerberos** 는 유저들이 보안 서버의 서비스를 통해 직접 인증을 받을 수 있도록 네트워크에 추가된 시스템/프로토콜이다. 원격 로그인, 원격 복사, 내부 보안 시스템파일 복사와 위험성이 높은 서비스를 더 안전하고 정확히 제어할 수 있다.

## Kerberos 인증 절차 요약



- 1 주체(이 예제에서는 User/Client)는 TGT(Ticket-Granting Ticket)를 요청하기 위해 KDC 에 인증 서비스를 요청한다.
- 2 KDC 는 클라이언트에게 TGT 를 준다. 이 TGT 는 키(세션 키 티켓)를 가지고 있고 클라이언트(User/Client)의 패스워드로 암호화되어 있다.
- 3 클라이언트는 다른 주체(이 예제에서 Application Server)에 접근하기 위한 티켓을 Ticket Granting Server 에 요청하기 위해 KDC 에서 받은 TGT 를 사용한다.
- 4 Ticket Granting Server 는 클라이언트가 서버의 특정 자원에 접근할 수 있는 서버 키를 준다. 여기서 다시 세션 키 두 개가 생성된다. 하나는 어플리케이션 서버의 키(티켓)로 암호화되어 어플리케이션 서버(Application Server)에게 다른 하나는 유저(User/Client)에게 보내진다.
- 5 클라이언트는 필요한 서비스를 요청할 수 있다.

**Kerberos** 는 인증을 확인하는 프록시 시스템이라고 설명할 수 있고 신뢰되는 추가적인 인증 시스템이라고도 한다. **Kerberos** 는 네트워크에서 안전한 유저 인증이라는 오직 한가지 기능만 제공한다. 권한 기능이나(유저가 무엇을 할 수 있는지) 감사 기능은(유저가 무엇을 했는지) 제공하지 않는다. 클라이언트와 서버가 **Kerberos** 인증을 사용하면 사생활과 데이터 보호를 위해 모든 통신을 암호화할 수 있다.

따라서 **Kerberos** 는 권한과 감사 서비스를 제공하는 다른 보안기능과 같이 사용하도록 강력히 권장한다.

---

다음 설명은 분산된 FreeBSD 에 **Kerberos** 를 어떻게 설정하는지 가이드로 사용할 수 있다. 그렇지만 완벽한 설명은 관련된 매뉴얼 페이지를 참고한다.

**Kerberos** 설치방법을 설명하기 위해 다양한 이름 공간(name spaces)을 다음과 같이 제어한다:

- DNS 도메인(“존”)은 example.org 다.
- **Kerberos** 영역은 EXAMPLE.ORG 다.

**Note:** 내부적으로 서비스를 하더라도 **Kerberos** 를 설정할 때 실제 도메인 이름을 사용한다. 이 방법으로 DNS 문제를 피하고 다른 **Kerberos** 영역과 상호작용을 보장한다.

## 14.7.1 역사적 배경

**Kerberos** 는 네트워크 보안 문제의 해결책으로 MIT 에서 만들었다. Kerberos 프로토콜은 강력한 암호화를 사용하기 때문에 클라이언트는 안전하지 않은 네트워크를 통해서 서버에게 인증을 받을 수 있다.

**Kerberos** 는 네트워크 인증 프로토콜과 툴 프로그램을(예를 들어 **Kerberos** 텔넷) 설명하는 프로그램이다. 현재 버전의 프로토콜은 RFC 1510 에 설명되어있는 버전 5 다.

광범위한 운영체제에서 사용되는 이 프로토콜의 여러 가지 툴도 무료로 사용할 수 있다. Massachusetts Institute of Technology(MIT)가 최초로 **Kerberos** 를 개발해서 **Kerberos** 패키지를 계속해서 개발하고 있다. 일반적으로 미국의 암호화 제품에 사용되어서 역사적으로 미국 수출법에 영향을 미쳤다. MIT **Kerberos** 는 포트([security/krb5](#))에서 사용할 수 있다. Heimdal **Kerberos** 는 버전 5 의 다른 도구이고 미국 수출법을 피해서 미국 밖에서 개발되었다(따라서 비영리 유닉스 변종에 많이 포함되었다). Heimdal Kerberos 배포본은 포트([security/heimdal](#))에서 사용할 수 있고 기본 FreeBSD 설치에 포함되어 있다.

다양한 독자를 위해 여기서는 FreeBSD 에 포함된 Heimdal 배포본 사용에 대해 다룬다.

---

## 14.7.2 Heimdal KDC 설정

키 배포 센터(KDC)는 **Kerberos** 가 제공하는(**Kerberos** 티켓) 인증 서비스를 중앙화한다. KDC 는 **Kerberos** 영역의 모든 컴퓨터에 의해 신뢰되기 때문에 강력한 보안이 필요하다.

**Kerberos** 서버가 실행되는 동안 약간의 자원이 필요하지만 KDC 로만 작동하는 머신이 보안 상 권장된다.

KDC 설정을 시작하기 위해 KDC 로(여러분의 시스템을 반영하도록 경로를 적절히 수정해야 될 것이다) 동작하도록 `/etc/rc.conf` 파일에 적절한 설정이 필요하다.

```
kerberos5_server_enable="YES"
kadmind5_server_enable="YES"
kerberos_stash="YES"
```

**Note:** `Kerberos_stash` 는 FreeBSD 4.X 에서만 사용할 수 있다.

**Kerberos** 설정 파일 `/etc/krb5.conf` 를 설정한다:

```
[libdefaults]
    default_realm = EXAMPLE.ORG
[realms]
    EXAMPLE.ORG = {
        kdc = kerberos.example.org
    }
[domain_realm]
    .example.org = EXAMPLE.ORG
```

이 `/etc/krb5.conf` 파일은 KDC 가 완벽한 호스트 이름 `kerberos.example.org` 를 가지고 있음을 보여준다. KDC 가 다른 호스트 이름을 가지고 있다면 존 파일에 CNAME(alias) 엔트리를 추가해야 된다.

**Note:** BIND DNS 서버로 적절히 설정한 대형 네트워크에는 위의 예제를 다음과 같이 수정할 수 있다:



```
[libdefaults]
default_realm = EXAMPLE.ORG
```

다음 라인으로 example.org 존 파일을 확장한다:

```
_kerberos._udp      IN  SRV      01 00 88 kerberos.example.org.
_kerberos._tcp      IN  SRV      01 00 88 kerberos.example.org.
_kpasswd._udp       IN  SRV      01 00 464 kerberos.example.org.
_kerberos-adm._tcp  IN  SRV      01 00 749 kerberos.example.org.
_kerberos            IN  TXT      EXAMPLE.ORG.
```

그리고 **Kerberos** 데이터베이스를 생성한다. 이 데이터베이스는 마스터 패스워드로 암호화된 주체 키를 모두 가지고 있다. 이 패스워드는 파일에(/var/heimdal/m-key) 저장되기 때문에 기억할 필요는 없다. 마스터 키를 생성하기 위해 `kstash` 를 실행하고 패스워드를 입력한다.

마스터 키가 생성되면 `kadmin` 프로그램에 `-f` 옵션을 사용하여 데이터베이스를 초기화할 수 있다. 이 옵션은 `kadmind` 네트워크 서비스를 통하지 않고 직접 데이터베이스 파일을 수정하도록 한다. 이 설정으로 데이터베이스를 생성하기 전에 발생하는 연결 문제를 제어할 수 있다. `kadmin` 프롬프트에서 최초의 영역 데이터베이스를 생성하기 위해 `init` 명령을 사용한다.

마지막으로 `kadmin` 에서 `add` 명령을 사용하여 첫 번째 주체 키를 생성한다. 지금은 주체에 기본 옵션을 사용하고 `modify` 명령으로 나중에 변경할 수 있다. 가능한 옵션을 보기 위해 언제나 `?` 명령을 사용할 수 있다.

샘플 데이터베이스를 생성하는 세션은 다음과 같다:

```
# kstash
Master key: xxxxxxxx
Verifying password - Master key: xxxxxxxx

# kadmin -l
kadmin> init EXAMPLE.ORG
Realm max ticket life [unlimited]:
kadmin> add tillman
Max ticket life [unlimited]:
```

```
Max renewable life [unlimited]:
Attributes []:
Password: xxxxxxxx
Verifying password - Password: xxxxxxxx
```

이제 KDC 서비스를 시작한다. 서비스를 시작하기 위해 `/etc/rc.d/kerberos start` 와 `/etc/rc.d/kadmind start` 를 실행한다. 지금은 kerberized 데몬이 실행되지 않지만 KDC 명령어 라인으로 방금 생성한 주체(유저)의 티켓 리스트를 확인하여 KDC 기능을 체크할 수 있다:

```
% k5init tillman
tillman@EXAMPLE.ORG's Password:

% k5list
Credentials cache: FILE:/tmp/krb5cc_500
  Principal: tillman@EXAMPLE.ORG

  Issued          Expires          Principal
Aug 27 15:37:58  Aug 28 01:37:58  krbtgt/EXAMPLE.ORG@EXAMPLE.ORG
```

### 14.7.3 Heimdal 서비스로 Kerberos 서버 활성화하기

첫째 **Kerberos** 설정파일 `/etc/krb5.conf` 를 복사해야 한다. 복사하는 것은 단순히 KDC 에서 클라이언트 컴퓨터로 안전하게(`scp(1)`같은 네트워크 유틸리티를 사용하거나 플로피 디스크로 물리적으로 안전하게) 복사한다.

다음으로 `/etc/krb5.keytab` 파일이 필요하다. 이 사항이 **Kerberos** 를 제공하기 위해 데몬이 활성화된 서버와 워크스테이션의 가장 큰 차이점이다. 서버는 keytab 파일이 필요하다. 이 파일은 KDC 가 각자의 인증을 확인하도록 하는 서버 호스트 키를 가지고 있다. 이 키가 공개되면 서버의 보안이 깨지기 때문에 보안에 유념하여 서버에 전송해야 된다. 이 말은 FTP 처럼 평범한 텍스트 채널을 통해 전송하는 것이 굉장히 어리석다는 것을 의미한다.

보통 keytab 은 `kadmin` 프로그램을 사용하는 서버에 전송한다. `kadmin` 을 사용하여 호스트

---

주체도(krb5.keytab 의 끝인 KDC) 생성해야 되기 때문에 유용하다.

앞에서 받은 티켓을 가지고 있어야 되고 이 티켓으로 kadmind.acl 에서 kademin 인터페이스를 사용할 수 있어야 된다. 접근 제한 리스트 디자인에 대한 자세한 사항은 Heimdal 정보 페이지에서 "원격 관리" 섹션을 본다. 원격 kadmin 접근을 활성화하지 않는다면 단순히 안전하게 KDC 에 연결하여(로컬 콘솔, ssh(1) 또는 **Kerberos telnet(1)**) **kadmin -i** 을 사용하여 관리한다.

/etc/krb5.conf 파일을 설치한 후 **Kerberos** 서버에서 kadmin 을 사용할 수 있다. **add --random-key** 명령은 서버 호스트 주체를 추가하고 **ext** 명령은 서버 호스트 주체의 keytab 만 추출할 수 있다. 예를 들면 다음과 같다:

```
# kadmin
kadmin> add --random-key host/myserver.example.org
Max ticket life [unlimited]:
Max renewable life [unlimited]:
Attributes []:
kadmin> ext host/myserver.example.org
kadmin> exit
```

ext 명령은("extract"의 약자) 기본적으로 추출한 키를 /etc/krb5.keytab 에 저장한다. KDC 에서(보안상의 이유로) kadmind 를 실행하지 않아서 원격에서 kadmin 에 접속할 수 없다면, 호스트 주체를 KDC 에 직접 추가한 후 다음과 같은 명령을 사용하여 임시 파일로(KDC 의 /etc/krb5.keytab 을 덮어쓰지 않고) 추출할 수 있다:

```
# kadmin
kadmin> ext --keytab=/tmp/example.keytab host/myserver.example.org
kadmin> exit
```

이제 안전하게 keytab 을 서버 컴퓨터에 복사한다(예를 들어 scp 나 플로피를 사용하여). KDC 의 keytab 을 덮어 쓰지 않기 위해 기본 keytab 이름이 아닌 다른 이름을 지정한다.

서버가 KDC 와(krb5.conf 파일로) 통신할 수 있기 때문에 인증을 받을(krb5.keytab 파일로) 수 있다. 이제 **Kerberos** 서비스를 활성화할 준비가 되었다. 예를 들면 아래와 같은 라인을 /etc/inetd.conf 에 넣고 /etc/rc.d/inetd restart 로 inetd(8) 서비스를 다시 시작하여 telnet 서비스를 활성화한다:

---

```
telnet stream tcp nowait root /usr/libexec/telnetd telnetd -a user
```

중요 비트를 `-a` 로(인증을 위해) `user` 에게 설정한다. 더 자세한 사항은 `telnet(8)` 매뉴얼 페이지를 참고한다.

## 14.7.4 Heimdal 로 클라이언트 Kerberos 활성화

클라이언트 컴퓨터를 설정하는 것은 상당히 쉽다. 모든 설정은 `/etc/krb5.conf` 에 있는 **Kerberos** 설정파일만 필요하다. 단순히 KDC 에서 이 파일을 클라이언트 컴퓨터에 안전하게 복사한다.

위에서 생성한 주체 티켓을 가져와서 보여주고 삭제하기 위해 `kinit`, `klist` 와 `kdestroy` 를 사용해서 클라이언트 컴퓨터를 테스트한다. 연결이 되지 않고 티켓을 받아 오는 것이 클라이언트나 KDC 의 문제가 아닌 서버의 문제라도 **Kerberos** 가 활성화된 서버에 연결할 수 있도록 **Kerberos** 어플리케이션을 사용할 수 있다.

`telnet` 같은 어플리케이션을 테스트할 때 패스워드가 보이는지 확인하기 위해 패킷 스니퍼(`tcpdump(1)` 같은)를 사용한다. 전체 데이터 흐름을 암호화하는(`ssh` 와 비슷한) `-x` 옵션으로 `telnet` 을 사용한다.

**Kerberos** 클라이언트 핵심 어플리케이션(전통적으로 `kinit`, `klist`, `kdestroy` 와 `kpasswd` 라는 이름으로) FreeBSD 기본 시스템에 설치된다. FreeBSD 5.0 이전 버전은 `k5init`, `k5list`, `k5destroy`, `k5passwd` 와 `k5stash` 라고 되어 있다.

코어(core)뿐만 아니라 다양한 **Kerberos** 클라이언트 어플리케이션들도 기본적으로 설치된다. 이것이 최소한의 Heimdal 기본 구성이며 텔넷은 유일하게 **Kerberos** 가 가능한 서비스다.

Heimdal 포트는 클라이언트 어플리케이션 중에서 빠진 것을 추가할 수 있다: **Kerberos** 가 활성화되는 버전의 `ftp`, `rsh`, `rcp`, `rlogin` 과 다른 일반적인 프로그램 몇 개. MIT 포트도 **Kerberos** 클라이언트 어플리케이션 전체 스위트(suite)를 포함한다.

## 14.7.5 유저 설정파일: `.k5login` 과 `.k5users`

---

영역에 있는 유저는 전형적으로 로컬 유저 계정과(tillman 이라는 로컬 계정) 매핑되는 **Kerberos** 주체를(tillman@EXAMPLE.ORG) 가지고 있다. telnet 같은 클라이언트 어플리케이션에는 보통 유저 이름이나 주체가 필요 없다.

그러나 가끔 **Kerberos** 주체와 대응하지 않는 로컬 유저 계정에 접근하기를 원한다. 예를 들면 tillman@EXAMPLE.ORG 가 로컬 유저 계정 webdevelopers 에 접근할 필요가 있다. 다른 주체도 이 로컬 계정에 접근할 필요가 있을 수 있다.

유저 홈 디렉터리에 있는 .k5login 과 .k5users 파일은 .hosts 와 .rhosts 의 강력한 결합처럼 사용할 수 있다. 예를 들면 다음과 같은 내용의 .k5login 이 유저 webdevelopers 의 홈 디렉터리에 있다면 나열되어 있는 두 개의 주체는 필요한 공유 패스워드 없이 계정에 접근할 수 있다.

```
tillman@example.org
jdoe@example.org
```

이들 명령의 매뉴얼 페이지를 참고하도록 한다. ksu 매뉴얼 페이지에서 .k5users 를 확인할 수 있다.

## 14.7.6 Kerberos 팁: 티켓과 문제해결

- Heimdal 이나 MIT Kerberos 포트를 사용할 때 PATH 환경변수에 **Kerberos** 버전의 클라이언트 어플리케이션을 시스템 버전의 클라이언트 이전에 나열해야 된다.
- 시간이 동기화 되었는가? 시간이 동기화되지(5 분 이내로) 않았다면 인증은 실패한다.
- 표준 프로토콜이 아닌 kadmin 을 제외하고 MIT 와 Heimdal 은 정확하게 상호작용한다.
- 호스트 이름을 변경한다면 host/ 주체를 변경해야 되고 keytab 을 업데이트 한다. 이 사항을 Apache 의 [www/mod\\_auth\\_kerb](#) 에 사용하는 www/ 주체 같은 특별한 keytab 엔트리에도 적용한다.

- 
- 영역에 있는 모든 호스트는 DNS 로(또는 최소한 /etc/hosts 로) 분해할(순방향과 역방향으로) 수 있어야 된다. CNAME 가 동작하더라도 A 와 PTR 레코드는 정확한 위치에 있어야 된다. 관련 에러 메시지가 별로 직관적이지 않기 때문에 주의한다:  
"refuses authentication because Read req failed: Key table entry not found"
  - KDC 의 클라이언트로 동작할지도 모르는 어떤 운영체제들은 setuid 로 root 가 될 수 있도록 ksu 에 퍼미션을 설정하지 않는다. 이 의미는 ksu 가 작동하지 않아서 보안상 안전하겠지만 상당히 귀찮아 진다는 뜻이다. 이것은 KDC 의 에러가 아니다
  - MIT Kerberos 에서 주체가 기본 값인 10 시간 이상의 티켓을 가지고 있기를 원한다면 두 주체의 최대 유효기간을 변경하기 위해 kadmin 에서 modify\_principal 을 사용해야 된다. 그리고 주체는 좀더 오래 지속되는 티켓을 요청하기 위해 kinit 에 - / 옵션을 사용할 수 있다.

**Note:** 문제 해결을 위해 KDC 에서 패킷 스니퍼를 실행하고 워크스테이션에서 kinit 를 실행했다면 이 전에 패스워드를 입력했더라도 kinit 가 실행되자마자 TGT(Ticket Granting Ticket)는 발송된다. 이 말은 **Kerberos** 서버는 인증을 받지 않은 요청에 TGT 를 자유롭게 전송한다; 그러나 모든 TGT 는 유저 패스워드에서 파생된 키로 암호화 된다. 따라서 유저가 패스워드를 입력하면 KDC 로 전송되지 않고 kinit 가 받은 TGT 를 암호화 하는데 사용된다. 디코딩하여 유효한 시간에 유효한 티켓을 얻었다면 유저는 유효한 **Kerberos** 인증을 갖게 된다. 이 인증은 나중에 **Kerberos** 서버와 안전하게 통신할 수 있는 세션 키와 실제 **Kerberos** 서버의 키로 암호화된 실제 티켓이 포함되어 있다. 이 암호화의 두 번째 레이어는 유저에게 알려져 있지 않지만 각 TGT 의 인증을 **Kerberos** 서버가 확인할 수 있도록 한다.

- 영역에 있는 모든 컴퓨터의 시간을 동기화해야 된다. 여기에는 NTP 가 적당할 것이다. NTP 에 대한 더 많은 정보는 23.7 장을 본다.
- 기간이 긴 티켓을(예를 들어 일주일) 사용하려면 티켓이 저장되어있는 머신에 연결하기 위해 **OpenSSH** 를 사용하고 ssh\_config 에서 **Kerberos TicketCleanup** 을 no 로 설정해야 된다. 그렇지 않고 로그아웃 하면 티켓은 바로 삭제된다.
- 호스트 주체는 기간이 좀더 긴 티켓을 가질 수 있다. 유저 주체는 일주일짜리 티켓을 가지고 있지만 연결하려는 호스트가 9 시간짜리라면 호스트 주체의 캐시가 만기 되면 티켓 캐시도 동작하지 않는다.

- 
- 안전하지 않은 특정 패스워드 사용을 방지하기 위해 krb5.dict 파일을 설정할 때 (kadmind 매뉴얼 페이지에 간단한 설명이 있다) 패스워드 정책이 할당된 주체에만 적용됨을 기억한다. krb5.dict 파일 포맷은 간단하다: 라인당 한 문자열씩 입력한다. 그리고 /usr/share/dic/words 에 심볼릭 링크를 생성하는 것이 유용할 것이다.

## 14.7.7 MIT 포트와 차이점

MIT 와 Heimdal 설치의 가장 큰 차이점은 다른 명령 설정을(그러나 동일한) 가지고 다른 프로토콜을 사용하는 kadmin 프로그램과 관련된 것이다. 이 말은 많은 것을 함축하고 있다. KDC 가 MIT 라면 원격에서 KDC 를 관리하기 위해 Heimdal kadmin 프로그램을 사용할 수 없다.

클라이언트 어플리케이션도 같은 태스크를 수행하는데 약간 다른 명령라인 옵션을 가지고 있을 것이다. MIT Kerberos 웹 사이트(<http://web.mit.edu/Kerberos/www/>)의 지시를 따른다. 그리고 경로 문제를 주의한다: MIT 포트는 기본적으로 /usr/local에 설치되기 때문에 PATH 환경변수에 시스템 디렉터리가 먼저 나열되어 있으면 MIT 대신 “일반” 시스템 어플리케이션이 실행된다.

**Note:** FreeBSD 가 제공하는 MIT [security/krb5](#) 포트에서 왜 telnetd 와 klogind 로 로 그인하는 것이 약간 다른지 궁금하다면 포트로 설치한 /usr/local/share/doc/krb5/README.FreeBSD 파일을 확인한다. 케시 파일에서 부정확한 퍼미션을 수정하려면 인증에 사용하는 login.krb5 바이너리가 필요하고 포워드 된 인증에 맞게 소유권을 적절히 변경할 수 있다.

## 14.7.8 Kerberos 의 문제를 완화하기

### 14.7.8.1 Kerberos 에 모두 접근할 수 있거나 아무도 접근할 수 없다.

네트워크에서 활성화된 모든 서비스는 **Kerberos** 와(또는 네트워크 공격에 안전한) 동작하도록 수정하고 그렇지 않으면 유저 인증이 획득 당해서 다시 사용될 것이다. 예를 들면 모든

---

원격 셸에(예를 들어 rsh 와 telnet) **Kerberos** 를 활성화할 수 있지만 패스워드를 평범한 텍스트로 보내는 POP3 메일 서버는 변경하지 못한다.

#### 14.7.8.2 Kerberos 는 싱글 유저 워크스테이션에 계획되었다

멀티 유저 환경에서 Kerberos 는 보안이 떨어진다. 모든 유저들이 읽을 수 있는 /tmp 디렉터리에 티켓을 저장하기 때문이다. 유저가 여러 사람들과 동시에 컴퓨터를 공유한다면(예: 멀티 유저) 유저의 티켓을 다른 유저가 볼 수(복사) 있다.

이 문제는 명령라인 옵션에 `-c` 파일 이름이나 KRB5CCNAME 환경변수로 해결할 수 있지만 흔히 사용하지 않는다. 주로 유저의 홈 디렉터리에 티켓을 저장하고 간단한 파일 퍼미션으로 이 문제를 해결한다.

#### 14.7.8.3 KDC 가 문제의 원인이다.

KDC 는 마스터 패스워드 데이터베이스를 가지고 있기 때문에 가장 안전해야 된다. KDC 에서는 어떤 서비스도 실행하지 말아야 되며 물리적으로도 안전해야 된다. **Kerberos** 는 같은 키로(“마스터” 키) 암호화한 모든 패스워드를 KDC 에 파일로 저장하기 때문에 위험성이 높다.

노출된 마스터 키는 생각처럼 위험하지 않을 수 있다. 마스터 키는 **Kerberos** 데이터베이스를 암호화하는데 사용되고 난수 발생기의 소스로 사용된다. 공격자가 마스터 키에 접근할 수 없도록 KDC 에 접근하는 것은 안전해야 된다

추가적으로, KDC 를 사용할 수 없다면(서비스 거부 공격이나 네트워크 문제일 것이다) 인증을 받을 수 없기 때문에 네트워크 서비스를 사용할 수 없고 따라서 서비스 거부 공격의 목표가 된다. 이것은 여러 대의 KDC 나(하나의 마스터와 여러 슬레이브) 대체인증(PAM 이 적당하다)으로 방지할 수 있다.

#### 14.7.8.4 Kerberos 단점

Kerberos 는 유저, 호스트 그리고 서비스가 직접 인증을 받을 수 있다. 그러나 유저, 호스트와 서비스가 KDC 에 인증을 받는 메커니즘은 없다. 이 의미는 트로이(Trojan) kinit 가(예를



---

들어) 모든 유저 이름과 패스워드를 기록할 수 있다. [security/tripwire](http://security/tripwire) 또는 다른 파일시스템 무결성 체크 툴을 여기에 사용할 수 있다.

### 14.7.9 참고할 만한 자료

- Kerberos FAQ(<http://www.faqs.org/faqs/kerberos-faq/>)
- 인증 시스템 디자인: a Dialogue in Four Scenes (<http://web.mit.edu/Kerberos/www/dialogue.html>)
- RFC 1510, The Kerberos 네트워크 인증 서비스(V5), RFC 1510
- MIT Kerberos 홈페이지 (<http://www.ietf.org/rfc/rfc1510.txt?number=1510>)
- Heimdal Kerberos 홈페이지 (<http://www.pdc.kth.se/heimdal/>)

## 14.8 방화벽

방화벽은 인터넷에 연결되어 있으며 사설 네트워크에 안전한 보안을 제공하는 어플리케이션을 찾는 사람들에게 관심이 증가되는 영역이다. 이번 장에서는 방화벽이 무엇이고 어떻게 사용하며 FreeBSD 커널에서 제공하는 방화벽 기능을 어떻게 사용하는지 설명한다.

**Note:** 사람들은 보통 내부 네트워크와 "크고 위험한 인터넷"의 모든 보안 문제를 방화벽이 해결할 것이라고 생각한다. 보안 문제를 도울 수 있지만 엉성하게 설정된 방화벽은 없는 것보다 더 큰 문제를 유발할 수 있다. 방화벽은 시스템에 다른 보안 계층을 추가할 수 있지만 굳은 결심으로 내부 네트워크를 뚫으려는 크래커는 막을 수 없다. 방화벽을 뚫을 수 없다고 너무 믿으면 크래커가 쉽게 내부 네트워크를 뚫게 만들어서 내부 보안에 실패할 것이다.

### 14.8.1 방화벽은 무엇인가?

오늘날 인터넷에서 일반적으로 사용하는 두 종류의 방화벽이 있다. 첫 번째는 *패킷 필터링*

---

라우터로 더 알려져 있다. 이런 종류의 방화벽은 *multi-homed* 머신(네트워크에 접속하기 위해 여러 IP 주소와 네트워크 인터페이스를 가지고 있는 컴퓨터 호스트를 의미한다. Multi-homed 호스트는 동일한 네트워크나 다른 네트워크에 물리적으로 연결되어 있다)을 사용하고 각각의 패킷을 포워드 하거나 막기 위해 룰을 설정한다. 프록시 서버로 알려져 있는 두 번째 종류는 인증을 제공하고 패킷을 포워드하는 데몬이 패킷 포워딩이 비활성 되어있는 커널을 가진 multi-homed 머신에 설치되어 있을 것이다.

가끔 사이트에서 두 종류의 방화벽을 결합하기 때문에 허가된 머신만(*bastion host*로 알려진) 패킷 필터링 라우터를 통해서 내부 네트워크로 패킷을 보낼 수 있다. 프록시 서비스는 일반 인증 메커니즘 보다 일반적으로 더 보안에 강한 베스천(bastion) 호스트에서 실행된다.

이번 장의 남은 부분에서 다루게 될 커널 패킷 필터링(IPFW로 알려진) FreeBSD는 가지고 있다. 프록시 서버는 추가적인 소프트웨어로 FreeBSD에 설치할 수 있지만 다양한 프록시 서버가 있기 때문에 이번 섹션에서 모든 것을 설명하기는 불가능하다.

### 14.8.1.1 패킷 필터링 라우터

라우터는 두 개 이상의 네트워크 사이에 패킷을 전달하는 머신이다. 패킷 필터링 라우터는 패킷 전달 여부를 결정하기 전에 각 패킷을 룰 리스트와 비교하는 프로그램이다. 대부분의 최근 IP 라우팅 소프트웨어는 기본적으로 모든 패킷을 전달하는 패킷 필터 기능을 가지고 있다. 필터를 활성화하려면 룰을 설정해야 된다.

방화벽은 패킷을 통과시킬지 결정하기 위해 설정된 룰과 패킷의 헤더 내용이 같은지 관찰한다. 룰에 대응하는 패킷을 발견하면 룰에 설정된 동작을 따른다. 룰의 동작은 패킷 삭제, 패킷 포워드 또는 패킷의 원래 주인에게 ICMP 메시지를 보낼 수도 있다. 룰은 대응된 순서에 따라 처음으로 대응된 룰이 적용된다. 따라서 룰 리스트는 "룰 체인"이라고 할 수 있다.

패킷 대응 규칙은 사용되는 소프트웨어에 따라 다르지만 일반적으로 패킷의 소스 IP 주소, 목적지 IP 주소, 소스 포트번호, 목적지 포트 번호(포트를 지원하는 프로토콜에) 또는 패킷 타입에(UDP, TCP, ICMP 등등) 따라 룰을 지정할 수 있다.

### 14.8.1.2 프록시 서버

프록시 서버는 일반적인 시스템 데몬(telnet, ftpd 등) 대신 특별한 서버를 가진 머신이다. 이

---

런 서버는 보통 전 방향(한쪽으로만) 연결만 허용하기 때문에 프록시 서버라고 부른다. 방화벽 호스트에 프록시 텔넷 서버를 운용(예를 들어)하면 사람들은 외부에서 방화벽으로 접속하기 위해 텔넷을 사용할 수 있으며 어떤 인증 메커니즘을 통해 내부 네트워크로 접근할 수 있다(다른 방식으로 프록시 서버는 내부 네트워크에서 오는 신호를 외부로 보내는 역할을 할 수 있다).

프록시 서버는 보통 일반 서버보다 보안에 강하고 때때로 원-타임 패스워드 시스템을 포함하는 광범위하고 다양한 인증 메커니즘을 이용할 수 있다. 따라서 누군가 여러분이 사용하는 패스워드를 발견했다라도 한번 사용한 후 패스워드가 즉시 소멸되어 그들이 이 패스워드로 시스템에 접근할 수 없다. 프록시 서버는 유저들이 실제로 호스트 머신에 접근하지 못하도록 하여 보안 시스템에 백도어를 설치하는 것은 아주 어렵게 된다.

프록시 서버는 종종 접근제한 이상의 방법을 가지고 있어서 지정한 호스트만 서버에 접근할 수 있다. 관리자는 보통 어떤 유저가 어떤 머신과 통신할 수 있는지 지정할 수 있다. 역시 어떤 프록시 소프트웨어를 선택하느냐에 따라서 특별한 기능을 사용할 수도 있다.

## 14.8.2 IPFW 기능은 무엇인가?

FreeBSD 에서 제공하는 IPFW 는 커널 안에 있는 패킷 필터링과 계정시스템으로 유저 기반의 제어 유틸리티 ipfw(8)를 가지고 있다. 이 두 가지로 커널이 라우팅을 결정할 때 사용하도록 룰을 정의하고 질의할 수 있다.

IPFW 에 관련된 두 개의 분야가 있다. 방화벽 섹션은 패킷 필터링을 수행한다. 그리고 방화벽 섹션에서 사용하는 것과 비슷한 룰 기반에서, 라우터 사용량을 추적하는 IP 계정 섹션이 있다. 예를 들면 얼마나 많은 라우터 트래픽이 특정 머신에서 오는지 또는 얼마나 많은 WWW 트래픽을 포워딩하는지 관리자가 모니터 할 수 있다.

IPFW 의 디자인으로 라우터가 없는 머신의 입력과 출력 연결에 IPFW 을 패킷 필터링처럼 사용할 수 있다. 이것은 IPFW 를 특이하게 사용하는 경우고 상황에 맞는 명령과 기술을 사용해야 된다.

## 14.8.3 FreeBSD 에서 IPFW 사용

커널 IPFW 시스템의 원하는 기능에 따라 커널 설정파일에 하나 이상의 옵션을 추가하고 커

---

널을 다시 컴파일한다. 커널을 다시 컴파일하는 방법에 대한 자세한 사항은 "커널 다시 설정하기(8 장)"를 본다.

**주의:** 모든 IP를 거부하는 것이 IPFW의 기본 정책이다. 시작할 때 접근을 허용하는 룰을 추가하지 않았다면 재 부팅 후 방화벽이 활성화된 커널로 바뀌어서 서버에 접근할 수 없다. 방화벽 기능을 처음으로 사용할 때 /etc/rc.conf 파일에 `firewall_type=open`으로 설정하고 새로운 커널 기능이 제대로 작동하는 것을 테스트한 후 /etc/rc.firewall의 방화벽 룰을 깨끗이 정리할 것을 권장한다. ssh를 사용하는 것보다 안전하게 로컬 콘솔에서 방화벽 설정을 초기화하는 것이 좋을 것이다. 커널을 빌드하는 다른 옵션은 `IPFIREWALL`과 `IPFIREWALL_DEFAULT_TO_ACCEPT` 옵션을 사용한다. 이 방법은 기본 IPFW의 룰을 변경해서 모든 IP를 허용하여 서버에 접근하지 못하는 것을 피할 수 있다.

IPFW와 관련된 4 가지 커널 설정 옵션이 있다:

*options IPFIREWALL*

패킷 필터링 코드를 커널에 컴파일한다.

*options IPFIREWALL\_VERBOSE*

패킷 로그는 `syslogd(8)`을 통해 남긴다. 필터 룰에 패킷 로그를 지정하였다더라도 이 옵션이 없으면 로그를 남기지 않는다.

*options IPFIREWALL\_VERBOSE\_LIMIT=10*

`syslogd(8)`를 통해 로그로 남기는 패킷의 수를 엔트리 별로 제한한다. 적대적인 환경에서 방화벽 활동을 로그로 남기길 원하겠지만 `syslog`가 가득 차는 서비스 거부 공격을 당하도록 서비스를 열어두고 싶지 않을 것이다.

체인 엔트리가 지정된 패킷 제한에 도달하면 로그는 특정 엔트리가 될 때까지 꺼진다. 다시 로그를 기록하려면 `ipfw(8)` 유틸리티를 사용하여 관련된 카운터를 리셋해

---

야 된다:

```
# ipfw zero 4500
```

4500 은 로그를 계속 기록하려는 체인 엔트리다.

*options IPFWALL\_DEFAULT\_TO\_ACCEPT*

“거부”에서 “허가”로 기본 룰을 변경한다. 이 옵션은 *IPFWALL* 을 지원하는 커널로 부팅했지만 방화벽을 아직 설정하지 않았다면 방화벽에 접근이 거부되는 것을 방지한다. 증가하는 특정 문제의 필터로 ipfw(8)을 사용하는 것도 매우 유용하다. 이 옵션은 방화벽을 열고 동작을 변경시키기 때문에 사용에 유의한다.

**Note:** FreeBSD 의 이전 버전은 *IPFWALL\_ACCT* 옵션을 가지고 있다. 방화벽 코드가 자동으로 계정 기능을 포함하기 때문에 이제 사용하지 않는다.

## 14.8.4 IPFW 설정

IPFW 소프트웨어 설정은 ipfw(8) 유틸리티를 사용할 수 있다. 이 명령의 구문은 아주 복잡해 보이지만 구조를 알면 비교적 단순하다.

이 유틸리티에 사용되는 4 개의 명령어 카테고리가 있다: 추가/삭제, 리스트, 플러싱(flushing) 그리고 클리어(clear). 추가/삭제는 패킷을 허용, 거부, 로그로 남기는지 제어하는 룰을 생성하는데 사용된다. 리스트는 룰 설정의(다른 말로 체인) 내용과 패킷 카운터를(계정) 검사하는데 사용된다. 플러싱은 체인에서 모든 엔트리를 삭제하는데 사용한다. 클리어는 하나 이상의 계정 엔트리를 0 으로 만드는데 사용한다.

### 14.8.4.1 IPFW 룰 변경

명령어 구문 형식은 다음과 같다:

```
ipfw[-N] 명령 [index] 동작 [log] 프로토콜 주소 [options]
```

---

이 형식의 명령을 사용할 때 유효한 플래그가 하나 있다:

**-N**

출력에서 주소와 서비스 이름을 분석한다.

주어진 **명령**은 아주 짧은 형식으로 줄일 수 있다. 유효한 **명령**은 다음과 같다:

**add**

방화벽/계정 룰 리스트에 엔트리 추가

**delete**

방화벽/계정 룰 리스트에서 엔트리 삭제

이전 버전의 IPFW 는 방화벽과 계정 엔트리로 나누어져 있었다. 현재 버전은 각 방화벽 엔트리별로 패킷 계정을 제공한다.

**index** 값이 제공되었다면 체인에서 엔트리를 특정 위치로 옮기는데(앞에서 패킷은 룰 리스트 중 처음으로 대응하는 룰의 동작을 따른다고 했다. 따라서 룰 순서를 지정하는데 사용된다) 사용된다. 그렇지 않으면 엔트리는 마지막 체인 엔트리 보다 100 이 큰 인덱스의 마지막 체인에 지정된다.

**IPFIREWALL\_VERBOSE**를 커널에 컴파일 하였다면 **log** 옵션은 룰에 대응하는 패킷을 시스템 콘솔에 출력한다.

유효한 **동작**은 다음과 같다:

**reject**

---

패킷을 버리고 패킷을 보낸 호스트에게 ICMP 호스트나 포트에 도달할 수 없다는 패킷을 보낸다.

#### allow

패킷을 그냥 통과 시킨다(엘리어스: *pass*, *permit* 와 *accept*).

#### deny

패킷을 버린다. 소스에게 ICMP 메시지로 통보하지 않는다(따라서 패킷은 절대 목적지에 도달할 수 없다).

#### count

패킷 카운터를 업데이트 하지만 이 를 기반으로 패킷의 허가/거부는 하지 않는다. 다음 체인 엔트리를 계속 검색한다.

각 동작은 가장 짧게 알아볼 수 있도록 지정한다.

지정할 수 있는 **프로토콜**은 다음과 같다:

#### all

모든 IP 패킷 대응

#### icmp

ICMP 패킷 대응

---

tcp

TCP 패킷 대응

udp

UDP 패킷 대응

주소는 다음과 같이 범위를 지정할 수 있다:

*address/mask [port]*에서 *address/mask [port]*까지 [*interface*를 통해]

지원하는 포트와 *protocols*를 결합할 때만 *port*를 지정할 수 있다(UDP와 TCP).

*interface*는 옵션으로 해당 인터페이스로 유입되는 패킷만 처리하도록 IP 주소나 로컬 IP 인터페이스의 도메인 이름 또는 인터페이스 이름을 지정한다. 인터페이스 유닛 번호는 와일드카드(\*) 지정할 수 있다. 예를 들어 *ppp\**는 모든 커널 *ppp* 인터페이스와 대응된다.

*address/mask* 지정에 사용되는 구문은 3 가지 중 하나다:

*address*

*address/mask-bits*

*address:mask-pattern*

유효한 호스트 이름을 IP 주소 대신 지정할 수 있다. *mask-bits*는 어떤 비트를 주소 마스크에 설정해야 되는지 숫자로 표현한다. 예를 들어 192.216.222.1/24로 지정하는 것은 C 클래스 서브넷에(이 경우 192.216.222) 매칭되는 모든 주소를 허가하는 마스크를 생성한다. *mask-pattern*은 주어진 주소와 논리적으로 AND 연산된 IP 주소다. 키워드 *any*는 "모든 IP 주소"를 지정할 때 사용한다.



---

차단하려는 **포트 번호**는 다음과 같이 지정한다:

*port* [,*port* [,*port* [...]]]

위와 같이 하나의 포트나 여러 개의 포트 지정할 수 있고 포트 범위를 지정하려면 다음과 같이 시작과 끝나는 포트를 입력한다.

*port-port*

사용 가능한 *options* 은 다음과 같다:

**frag**

패킷이 데이터그램(데이터 또는 패킷을 의미한다)의 첫 번째 조각이 아니라면 대응한다.

**in**

입력 패킷이면 대응한다.

**out**

출력 패킷이면 대응한다.

**ipoptions *spec***

IP 헤더가 *spec*에 나열한 옵션을 가지고 있다면 대응한다. 지원되는 IP 옵션: *ssrr* (strict source route), *lsrr* (loose source route), *rr* (record packet route)와 *ts* (time stamp). 특별한 옵션이 없으면 읽기로 지정될 것이다.

**established**

---

패킷이 이미 연결된 TCP 연결의 일부분이면(다시 말해 RST 나 ACK 비트 설정을 가지고 있다면) 대응한다. *established* 룰을 체인의 앞부분에 지정하여 방화벽의 성능을 최적화할 수 있다.

#### setup

패킷이 TCP 연결을(SYN 비트는 설정 되지만 ACK 비트는 아니다) 만들려고 하면 대응한다.

#### tcpflags *flags*

TCP 헤더가 콤마로 나뉘어진 *flags* 리스트를 포함하면 대응한다. 지원되는 플래그는 *fin*, *syn*, *rst*, *psh*, *ack* 와 *urg* 다. 특별한 플래그가 없으면 읽기로 표시될 것이다.

#### icmptypes *types*

ICMP 종류가 *types* 리스트에 있다면 대응한다. 이 리스트는 콤마로(,) 나뉘어진 각 범위를 and/or 로 결합한 것처럼 나뉘어져 있을 것이다. 일반적으로 사용되는 ICMP 종류: 0 에코 응답(핑 응답), 3 목적지 도달불가, 5 리다이렉트, 8 에코 요청(핑 요청)과 11 시간 초과 (traceroute(8)에서 TTL 을 표현하는데 사용된다).

### 14.8.4.2 IPFW 룰 리스트

명령어 구문은 다음과 같다:

```
ipfw [-a][-c][-d][-e][-t][-N][-S] list
```

이 형식의 명령을 사용할 때 7 개의 유효한 플래그가 있다:

-a

---

나열하는 동안 카운터 값을 보여준다. 이 옵션이 계정 카운터를 볼 수 있는 유일한 방법이다.

-c

조밀한 형식으로 룰을 보여준다.

-d

동적인 룰과 추가로 정적인 룰도 보여준다.

-e

-d가 지정되어 있다면 소멸된 동적 룰도 보여준다.

-t

각 체인 엔트리에 마지막으로 대응된 시간을 보여준다. 이 시간 리스트는 ipfw(8) 유틸리티로 사용되는 입력구문과 호환되지 않는다.

-N

주어진 주소와 서비스 이름을 분석한다.

-S

각 룰에 속한 설정을 보여준다. 이 플래그가 지정되지 않았다면 비활성된 룰은 나열되지 않는다.

---

### 14.8.4.3 IPFW 룰들 삭제

체인을 삭제하기 위한 구문은 다음과 같다:

```
ipfw flush
```

이 명령은 커널에 의해(인덱스 65535) 강제로 고정된 기본 정책을 제외한 방화벽 체인의 모든 엔트리를 삭제한다. 룰을 정리할 때 경고 사용; 기본 거부 정책은 허용 엔트리가 체인에 추가될 때까지 네트워크에서 시스템을 차단한다.

### 14.8.4.4 IPFW 패킷 카운터 정리

하나 또는 더 많은 패킷 카운트를 정리하기 위한 구문은 다음과 같다:

```
ipfw zero [index]
```

*index* 인수 없이 사용하면 모든 패킷 카운터를 정리한다. *index* 를 제공하였다면 정리 동작은 지정한 체인 엔트리에만 영향을 미친다.

### 14.8.5 ipfw 예제 명령

이 명령은 호스트 evil.crackers.org 로부터 nice.people.org 호스트의 텔넷 포트로 모든 패킷을 거부한다:

```
# ipfw add deny tcp from evil.crackers.org to nice.people.org 23
```

다음 예제는 crackers.org 전체 네트워크로부터(C 클래스) nice.people.org 머신으로(모든 포트)의 모든 TCP 트래픽을 거부하고 로그로 남긴다.

```
# ipfw add deny log tcp from evil.crackers.org/24 to nice.people.org
```

사람들이 내부 네트워크로(C 클래스의 서브넷) X 세션을 보내지 못하도록 하려면 다음 명령

---

이 필요한 필터링을 할 것이다:

```
# ipfw add deny tcp from any to my.org/28 6000 setup
```

계정 레코드를 보려면 다음 명령을 사용한다:

```
# ipfw -a list
```

또는 짧은 형식의 명령은 다음과 같다:

```
# ipfw -a l
```

체인 엔트리가 마지막으로 대응된 시간을 볼 수 있다:

```
# ipfw -at l
```

## 14.8.6 패킷 필터링 방화벽 만들기

**Note:** 다음은 단지 가정이다. 각 방화벽의 필요 조건이 다르기 때문에 여러분의 특정 필요조건에 맞는 방화벽을 어떻게 만드는지 설명할 수 없다.

방화벽을 처음 설정할 때 안전한 환경에서 방화벽을 설정할 수 없고 테스트할 공간이 없다면 로그가 남는 명령어 버전을 사용해서 커널이 로그를 남기도록 강력히 권장한다. 이 방법으로 문제 영역을 빨리 확인할 수 있고 큰 혼란 없이 처리할 수 있다. 초기설정이 끝난 후라도 가능한 공격에 대한 흔적인 “거부”를 로그로 남기고 필요한 경우 방화벽의 룰을 수정하는 것도 로그로 남기도록 한다.

**Note:** accept 된 것을 로그로 남긴다면 많은 로그 데이터를 만든다는 것을 알고 있어야 한다. 로그 엔트리는 방화벽을 지나는 모든 패킷별로 로그 엔트리를 하나씩 생성하기 때문에 거대한 FTP/http 전송 등은 시스템을 아주 느리게 한다. 또한 패킷이 지나가기 전에 커널에 의해 더 많은 작업이 필요하기 때문에 이러한 패킷에 대한 로그가 증가하게 되어있다. 또한 **syslogd** 는 추가적인 모든 데이터를 디스크에 로그로 남길 때 프로세서를 오랫동안 사용하고 /var/log 파티션을 쉽게 채울 수 있다.

---

/etc/rc.conf.local 이나 /etc/rc.conf 에서 방화벽을 활성화한다. 관련 매뉴얼 페이지에서 방화벽을 정지하고 미리 설정된 설정을 어떻게 보는지 설명한다. 미리 설정되어 있는 것을 사용하지 않겠다면 rc.conf 에 지정할 수 있는 파일에 **ipfw list** 명령으로 현재 룰 세트를 출력한다. 방화벽을 활성화하기 위해 /etc/rc.conf.local 이나 /etc/rc.conf 를 사용하지 않는다면 IP 인터페이스를 설정하기 전에 방화벽이 활성화되어 있어야 된다.

다음 문제는 방화벽이 실제로 무엇을 해야 되는지 설정하는 것이다. 이것은 외부에서 네트워크 접근을 원하는지 그리고 내부에서 외부로 얼마나 많이 접근할 것 인가에 따라 다르다. 일반적인 몇 가지 룰은 다음과 같다:

- TCP 1024 포트 아래로 들어오는 패킷은 모두 막는다. 대부분 보안과 민감한 서비스가 finger, SMTP(메일)와 telnet 같은 것이기 때문이다.
- 들어오는 모든 UDP 트래픽을 막는다. UDP 와 관련된 유용한 서비스와 트래픽은 매우 적고 보안을 위협하기 때문이다(예: Suns RPC 와 NFS 프로토콜). 또한 UDP 는 비 연결 지향 프로토콜이기 때문에 단점도 될 수 있다; 입력되는 UDP 트래픽을 막는 것은 UDP 응답으로 나가는 트래픽도 막게 된다. 이 방법은 외부에 있는 서버를 사용하는 사람들에게(내부에 있는) 문제가 될 수 있다. 외부 서버에 접근을 허용하려면 방화벽을 통해 내부 UDP 포트 191 에서 1525 번으로 들어오는 패킷을 허가해야 된다. **ntp** 는 포트 123 으로 입력되는 다른 서비스이기 때문에 이 포트도 열어 줘야 될 것이다.
- 외부로부터 포트 6000 으로 들어오는 트래픽을 막는다. 포트 6000 은 X11 서버에 접근하는데 사용되기 때문에 보안을 위협할 수 있다(특히 워크스테이션에서 xhost +를 습관적으로 사용하는 사람). X 11 은 실제로 6000 부터 시작하는 포트 범위를 사용할 수 있기 때문에 6000 이상을 제한하여 머신에서 사용할 수 있는 X 디스플레이 개수를 제한할 수 있다. RFC 1700 에(할당된 번호) 정의하는 상한선은 6063 이다.
- 어떤 포트를 내부 서버가 사용하는지(예: SQL 서버 등) 체크한다. 보통 위에서 지정한 1~1024 범위를 벗어나기 때문에 이들 포트도 막는 것이 좋을 것이다.

방화벽 설정을 위한 다른 체크 리스트는 CERT

([http://www.cert.org/tech\\_tips/packet\\_filtering.html](http://www.cert.org/tech_tips/packet_filtering.html))에서 확인할 수 있다.

---

위의 내용은 오직 가이드라인만 제시한다. 어떤 필터 룰을 방화벽에 사용할지 직접 결정해야 된다. 위에서 제공한 충고를 따르고 누군가 네트워크에 들어왔더라도 어떤 책임도 갖지 않는다.

### 14.8.7 IPFW 오버헤드와 최적화

많은 사람들은 얼마나 많은 IPFW 오버헤드가 시스템에 추가되는지 알고 싶어한다. 이에 대한 대답은 대부분 룰 설정과 프로세서 속도에 따라 다르다. 그리고 이더넷과 관련된 대부분의 어플리케이션과 작은 룰 셋에 대해서는 무시해도 된다. 여러분의 호기심을 만족시키고 실제 필요한 평가를 위해 다음 내용을 읽도록 한다.

다음 평가는 486-66 에 2.2.5-STABLE 를 사용하였다(IPFW 가 FreeBSD 의 마지막 릴리즈에서 약간 변경되었지만 비슷한 속도로 수행되고 있다). IPFW 는 *ip\_fw\_chk* 루틴에 사용되는 시간을 측정해서 1000 패킷마다 결과를 콘솔에 출력하도록 수정하였다.

룰 셋 두 가지를 룰별로 1000 번씩 테스트하였다. 첫 번째는 룰이 반복되는 최악의 상황을 보여주도록 디자인 하였다:

```
# ipfw add deny tcp from any to any 55555
```

위 명령은 마지막으로 패킷이 룰에 대응하지(포트 번호의 장점) 않는다고 결정하기 전에 대부분의 IPFW 의 패킷 체크 루틴이 수행되는 최악의 시나리오를 보여준다. 이 룰이 999 번 반복되고 모든 IP 접근(any 에서 any 까지)을 허용한다.

두 번째 룰 설정은 룰 체크를 빨리 중단하도록 디자인 하였다:

```
# ipfw add deny ip from 1.2.3.4 to 1.2.3.4
```

위의 룰에 대응하지 않는 소스 IP 주소는 이런 룰을 매우 빨리 지나가도록 한다. 전과 마찬가지로 1000 번째 룰은 모든 IP 를 허용한다.

앞의 경우 패킷별로 프로세스 오버헤드는 대략 2.703 ms/packet 또는 대략 룰 별로 2.7 밀리 초다. 그래서 이론상 이러한 룰에 의한 패킷 프로세싱 제한은 초당 대략 370 패킷이 된다. 10Mbps 이더넷에 ~1500 바이트 패킷 크기는 오직 55.5%의 대역폭만 이용하게 된다.

---

나중의 경우 각 패킷은 대략 1.172ms로 진행하거나 대충 룰별로 1.2 밀리 초가 소요된다. 이론적인 패킷 프로세스 제한은 10Mbps 이더넷 대역을 소비할 수 있는 초당 대략 853 패킷이다.

과도한 수의 룰 테스트와 이러한 룰은 실제 세계에서는 통하지 않는다. 이러한 결과는 시간 정보만 보여주는데 사용된다. 여기 효율적인 룰을 설정할 때 고려할 몇 가지가 있다:

- TCP 트래픽을 주로 제어하기 위해 *established* 룰을 먼저 적용한다. 이 룰 전에 어떤 *allow tcp* 문도 두지 않는다.
- 룰 설정에서 거의 사용하지 않는 룰 이전에 자주 사용되는 룰을 둔다(물론 방화벽 변경 없이). 어떤 룰이 보통 사용되는지 `ipfw -a 1`로 패킷 카운팅 통계를 체크해 볼 수 있다.

## 14.9 OpenSSL

많은 유저가 간과하는 기능 중 한가지는 FreeBSD에 포함된 **OpenSSL** 툴킷이다. **OpenSSL**은 일반 통신 레이어에 암호화 전송 레이어를 제공한다; 따라서 많은 네트워크 어플리케이션 및 서비스와 **OpenSSL**이 상호 작동하게 한다.

**OpenSSL**을 사용하는 방법에는 메일 클라이언트의 암호화된 인증, 신용카드 결제와 같은 웹 기반 트랜잭션 등이 포함될 것이다. [www/apache13-ssl](http://www/apache13-ssl)과 [mail/sylpheed-claws](http://mail/sylpheed-claws)와 같은 포트들은 **OpenSSL**을 지원하도록 컴파일 한다.

**Note:** 대부분의 경우 포트 컬렉션은 `WITH_OPENSSL_BASE` make 변수를 “yes”로 설정하지 않았다면 [security/openssl](http://security/openssl)을 빌드한다.

FreeBSD에 포함되어 있는 **OpenSSL** 버전은 보안 소켓 레이어 v2/v3 (SSLv2/SSLv3), 전송 레이어 보안 v1(TLSv1) 네트워크 보안 프로토콜 그리고 어플리케이션에 사용할 수 있도록 일반적인 암호화 라이브러리를 제공한다.

**Note:** **OpenSSL**이 IDEA 알고리즘을 지원하지만 미국의 특허권 때문에 기본적으로 비활성되어 있다. 사용하기를 원하면 라이선스를 다시 확인하고 제한을 수락할 수 있다면 `make.conf`에 `MAKE_IDEA` 변수를 설정한다.



---

아마도 **OpenSSL** 을 가장 일반적으로 사용하는 방법은 어플리케이션을 사용할 수 있도록 증명서를 제공하는 것이다 이들 증명서는 기업체나 개인의 자격을 증명한다. 증명서에 여러 증명 기관이나 CA(Certificate Authority)의 검증이 없다면 보통 경고 메시지가 뜬다. 증명 기관은 기업체나 개인의 유효한 자격을 검증하기 위해 증명서에 서명을 해주는 VeriSign 과 같은 회사다. 이러한 절차는 비용이 소요되므로 증명서를 사용하기 위해 꼭 필요한 것은 아니다; 그러나 의심이 많은 사람들에게 두려움을 덜어 줄 수 있다.

### 14.10.1 증명서 만들기

증명서를 만들기 위해 다음 명령을 사용할 수 있다:

```
# openssl req -new -nodes -out req.pem -keyout cert.pem
Generating a 1024 bit RSA private key
.....++++++
.....++++++
writing new private key to 'cert.pem'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:PA
Locality Name (eg, city) []:Pittsburgh
Organization Name (eg, company) [Internet Widgits Pty Ltd]:My Company
Organizational Unit Name (eg, section) []:Systems Administrator
Common Name (eg, YOUR name) []:localhost.example.org
Email Address []:trhodes@FreeBSD.org

Please enter the following 'extra' attributes
to be sent with your certificate request
```

---

```
A challenge password []:SOME PASSWORD
```

```
An optional company name []:Another Name
```

“Common name” 프롬프트가 나타나면 도메인 이름을 입력한다. 이 프롬프트에는 증명서를 검증하는 서버 이름이 필요하다; 원하는 문자를 입력하면 되지만 도메인 이름을 입력하면 쓸데없는 증명서를 만들게 된다. 예를 들어 만기일이나 다른 암호화 알고리즘 같은 다른 옵션도 사용할 수 있다. 완전한 옵션 리스트는 `openssl(1)` 매뉴얼 페이지에서 찾을 수 있다.

`cert.pem` 파일은 앞서 명령을 입력한 디렉터리에 없다. 이 증명서는 수많은 CA(인증 기관) 중 한곳의 서명을 받도록 보내졌을 것이다.

CA의 서명이 필요 없다면 자체적으로 서명이 된 증명서를 만들 수 있다. 첫째로 CA 키를 생성한다:

```
# openssl gendsa -des3 -out W
myca.key 1024
```

증명서를 만들기 위해 이 키를 사용한다:

```
# openssl req -new -x509 -days 365 -key W
myca.key -out new.crt
```

두 개의 새로운 파일이 디렉터리에 생성되었다: 인증 기관 서명 파일 `myca.key`와 증명서 `new.crt`가 생성되었다. 이들 파일은 `root`만 읽을 수 있도록 하여 `/etc`와 같은 적당한 디렉터리로 이동시킨다. `chmod` 유틸리티로 `0600` 퍼미션이 적당할 것이다.

## 14.10.2 증명서를 사용하는 예제

이들 파일로 무엇을 할 수 있는가? 적절한 사용법은 `Sendmail` MTA의 연결을 암호화하는데 사용한다. 그래서 로컬 MTA를 통해 메일을 보내는 유저가 깨끗한 텍스트 인증을 사용하지 않게 한다.

**Note:** 어떤 MUA는 로컬에 증명서를 설치하지 않으면 에러가 발생하기 때문에 이것은 이상적인 사용법이 아니다. 증명서 설치에 대한 더 자세한 정보는 소프트웨어에 포함되어 있는 문서를 참고한다.

---

다음 라인을 로컬 `.mc` 파일에 입력한다:

```
dnl SSL Options
define(`confCACERT_PATH',`/etc/certs')dnl
define(`confCACERT',`/etc/certs/new.crt')dnl
define(`confSERVER_CERT',`/etc/certs/new.crt')dnl
define(`confSERVER_KEY',`/etc/certs/myca.key')dnl
define(`confTLS_SRV_OPTIONS', `V')dnl
```

`/etc/certs/`는 증명서와 키 파일을 저장할 디렉터리다. 이제 남은 단계는 로컬 `.cf` 파일을 다시 빌드한다. `/etc/mail` 디렉터리에서 **make install** 을 입력하여 쉽게 빌드할 수 있다. 그리고 **Sendmail** 데몬을 다시 시작하는 **make restart** 명령을 사용한다.

`/var/log/maillog` 파일에 에러가 없고 모든 것이 정상이라면 **Sendmail** 이 프로세스 리스트에 나타난다.

간단한 테스트를 위해 `telnet(1)` 유틸리티를 사용하여 메일 서버에 연결한다:

```
# telnet example.com 25
Trying 192.0.34.166...
Connected to example.com.
Escape character is '^]'.
220 example.com ESMTP Sendmail 8.12.10/8.12.10; Tue, 31 Aug 2004 03:41:22 -0400
(EDT)
ehlo example.com
250-example.com Hello example.com [192.0.34.166], pleased to meet you
250-ENHANCEDSTATUSCODES
250-PIPELINING
250-8BITMIME
250-SIZE
250-DSN
250-ETRN
250-AUTH LOGIN PLAIN
250-STARTTLS
250-DELIVERBY
```

```
250 HELP
quit
221 2.0.0 example.com closing connection
Connection closed by foreign host.
```

“STARTTLS” 라인이 출력에 나타나면 모든 것이 정상적으로 동작하는 것이다.

## 14.10 IPsec 로 VPN 만들기

FreeBSD 게이트웨이를 사용하여 인터넷으로 나뉜 두 개의 네트워크에 VPN 을 생성한다.

### 14.10.1 IPsec 이해

이 섹션은 FreeBSD 와 **Microsoft Windows 2000/XP** 머신 환경에서 안전하게 통신할 수 있도록 IPsec 를 설정하여 사용하는 과정을 설명한다. IPsec 설정은 사용자 커널 설정에(8 장) 대해 이해하고 있어야 된다.

IPsec 는 인터넷 프로토콜 레이어의 최 상단에 있는 프로토콜로서 두 대 이상의 머신이 안전하게 통신할 수 있게 한다. FreeBSD IPsec “network stack”은 IPv4 와 IPv6 를 지원하는 KAME(<http://www.kame.net/>) 프로젝트 기반이다.

**Note:** FreeBSD 5.X 는 OpenBSD 에서 가져온 “Fast IPsec”라는 “하드웨어 가속” IPsec 스택을 가지고 있다. IPsec 의 성능을 최적화하기 위해 crypto(4) 서브시스템으로 하드웨어 암호화를 적용한다. 이 서브시스템은 새로운 것이기 때문에 KAME 버전의 IPsec 에서 사용할 수 있는 모든 기능을 지원하지 않는다. 그러나 하드웨어 가속 IPsec 를 활성화하려면 커널 설정파일에 다음 옵션을 추가해야 된다:

```
options FAST_IPSEC # new IPsec (cannot define w/ IPSEC)
```

현재 KAME 버전의 IPsec 와 lue 에서 “Fast IPsec” 서브시스템을 사용하는 것은 불가능하다. 더 많은 정보는 fast\_ipsec(4) 매뉴얼 페이지를 참고한다.

IPsec 는 두 개의 서브 프로토콜로 이루어진다:

- *Encapsulated Security Payload(ESP)*는 대칭 암호화 알고리즘으로(Blowfish 와 3DES 같은) 내용을 암호화하여 불법적인 패킷 캡처 시도로부터 IP 패킷 데이터를 보호한다.
- *Authentication Header(AH)*는 암호화 체크섬 계산과 보안 해싱(hashing) 기능으로 IP 패킷 헤더 필드를 해싱하여 불법적인 패킷 캡처 시도와 스푸핑(변조)으로부터 IP 패킷 헤더를 보호한다.

ESP 와 AH 는 환경에 따라 같이 사용하기도하고 따로 사용할 수도 있다.

IPsec 는 두 호스트 사이의 트래픽을 직접 암호화할 수 있고 안전한 통신을 위해 두 개의 서브넷에 터널모드(*Tunnel Mode*) 라는 “가상 터널”을 생성할 수 있다. 후자는 일반적으로 가상 사설 망(*Virtual Private Network (VPN)*)으로 더 유명하다. FreeBSD 의 IPsec 서브시스템에 대한 더 자세한 정보는 ipsec(4) 매뉴얼 페이지를 참고한다.

커널에 IPsec 지원을 추가하려면 커널 설정파일에 다음 옵션을 추가한다:

```
options IPSEC #IP security
options IPSEC_ESP #IP security (crypto; define w/ IPSEC)
```

IPsec 디버깅을 지원하려면 다음 커널 옵션도 추가해야 된다:

```
options IPSEC_DEBUG #debug for IP security
```

## 14.10.2 문제

VPN 을 구성하는 표준은 없고 장점과 단점을 가진 여러 가지 기술을 사용하여 구축할 수 있다. 이 문서에서는 여러 가지 시나리오와 각 시나리오에 따른 VPN 구축 전략을 보여준다.

## 14.10.3 시나리오 #1: 인터넷에 연결되어 하나처럼 동작하는 두 개의 네트워크

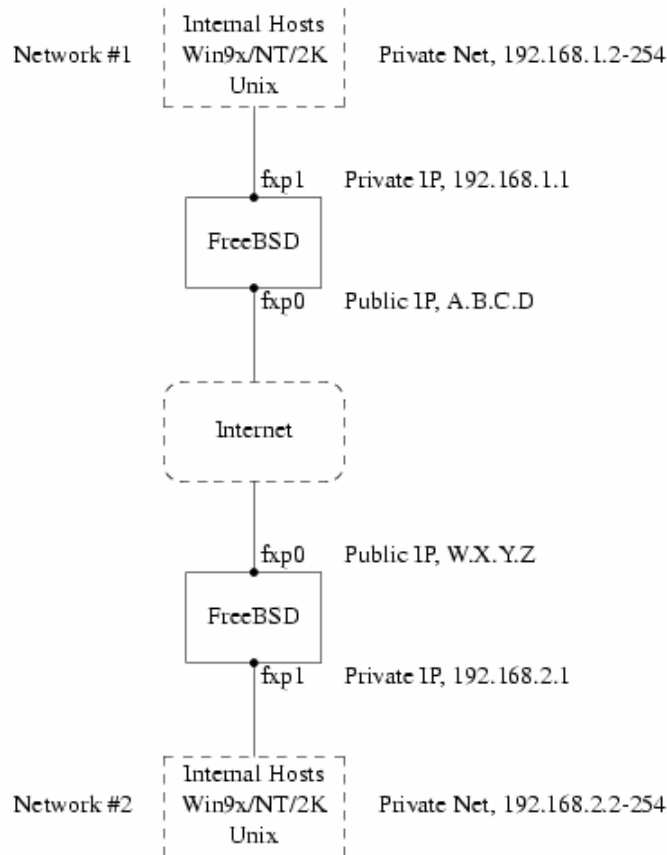
---

이것이 내가 처음으로 VPN에 관심을 가지게 된 시나리오다. 다음과 같은 전제를 기반으로 한다:

- 최소한 두 개의 사이트가 있다.
- 두 사이트는 사설 IP를 사용한다
- 두 사이트는 FreeBSD가 동작하는 게이트웨이를 통해 인터넷에 연결되어 있다.
- 각 네트워크의 게이트웨이는 최소한 한 개의 공인 IP 주소를 가지고 있다.
- 두 네트워크의 내부 주소는 공인이든 사설 IP 주소든지 상관없다. 필요하다면 게이트웨이 머신에서 NAT를 실행할 수 있다.
- 두 네트워크의 내부 IP 주소는 충돌하지 않는다. 이 작업에 VPN 기술과 NAT를 조합하여 사용하는 것도 이론상 가능하지만 엄청난 설정 작업이 수반될 것이라고 생각한다.

양쪽 내부 네트워크에 같은 범위의 사설 IP 주소를 사용하고 있었다면 한쪽은 번호를 바꿔야 된다.

네트워크 토폴로지는 다음과 비슷할 것이다:



---

그림에서 두 개의 공인 IP 주소(Public IP A.B.C.D 와 Public IP W.X.Y.Z)를 주의한다. 이 문서에서는 두 개의 공인 IP 를 의미하는 문자로 바꾸어 사용했다. 이 문서에서 이들 문자를 보게 되면 여러분의 공인 IP 주소로 바꾼다. 그리고 두 개의 게이트웨이 머신이 가지고 있는 .1 내부 IP(192.168.1.1 과 192.168.2.1 을 의미한다)도 마찬가지로 변경한다. 두 개의 네트워크는 서로 다른 사설 IP 주소를(각각 192.168.1.x 와 192.168.2.x) 가지고 있다. 사설 네트워크의 모든 머신은 .1 머신을 기본 게이트웨이로 사용하였다.

네트워크 관점에서, 가끔 패킷을 잃어버리는 경향이 있는 약간 느린 라우터에 머신들이 직접 연결되어있더라도 각 네트워크는 다른 네트워크에 머신이 있는 것처럼 보여야 된다.

이 의미는 머신 192.168.1.20 에서 다음과 같이 ping 을 실행했을 때 동작해야 된다.

#### **ping 192.168.2.34**

Windows 머신은 로컬 네트워크 머신을 탐색할 수 있는 것처럼 다른 네트워크에 있는 머신을 볼 수 있고 공유 파일을 탐색할 수 있어야 한다.

그리고 전체가 보안상 안전해야 된다. 이 의미는 두 개의 네트워크 트래픽을 암호화해야 된다는 것이다.

두 네트워크 사이에 VPN 을 생성하려면 다음과 같이 여러 단계가 필요하다:

- ① 인터넷으로 연결되어 있는 두 개의 네트워크 사이에 “가상” 네트워크를 생성한다. 제대로 동작하는지 ping(8) 같은 툴을 사용하여 테스트한다.
- ② 필요하다면 두 네트워크 사이의 트래픽을 암호화하고 복호화하는 보안 정책을 적용한다. 이 트래픽이 암호화되었는지 tcpdump(1) 같은 툴을 사용하여 테스트한다.
- ③ Windows 머신이 VPN 을 통해 다른 네트워크를 볼 수 있도록 FreeBSD 게이트웨이 에 부가적인 소프트웨어를 설정한다.

### **14.10.3.1 단계 1: “가상” 네트워크 생성 및 테스트**

네트워크 #1 의 게이트웨이 머신에 로그인해서 IP 주소 W.X.Y.Z 머신의 사설 주소로 ping 192.168.2.1 을 실행한다. 이 명령이 동작하려면 무엇이 필요한가?

- 
- ① 게이트웨이 머신은 192.168.2.1 에 어떻게 도달하는지 알아야 된다. 다시 말해 192.168.2.1 의 라우트를 알아야 된다.
  - ② 192.168.x 같은 사설 IP 주소 범위는 인터넷에 나타나지 않는다. 대신 192.168.2.1 로 보내는 각 패킷은 다른 패킷에 감싸진다. 이 패킷은 A.B.C.D 에서 나타나서 W.X.Y.Z 로 보내져야 한다. 이 절차를 *캡슐화(encapsulation)*라고 한다.
  - ③ 이 패킷이 W.X.Y.Z 에 도달하면 *캡슐해제(unencapsulated)*하여 192.169.2.1 에 전송해야 된다.

두 개의 네트워크 사이에 “터널”이 필요하다고 생각할 수 있다. 두 “터널 입구”는 IP 주소 A.B.C.D 와 W.X.Y.Z 이고 사설 IP 주소가 거쳐갈 수 있도록 알려줘야 한다. 터널은 공인 인터넷을 거쳐 사설 IP 주소로 트래픽을 전송한다.

이 터널은 일반적인 인터페이스나 FreeBSD 의 **gif** 장치를 사용하여 생성된다. 각 게이트웨이 호스트의 gif 인터페이스는 4 개의 IP 주소로 설정된다고 생각할 수 있다; 공인 IP 주소 두 개와 사설 IP 주소 두 개.

gif 장치를 지원하도록 양쪽 머신의 FreeBSD 커널에 컴파일해야 된다. 양쪽 머신의 커널 설정파일에 다음 라인을 추가하고 컴파일하여 설치한 후 재 부팅한다.

**pseudo-device gif**

커널 설정은 두 단계로 이루어진다. 첫째, 터널은 gifconfig(8)을 사용하여 외부 IP 주소(공인)가 무엇인지 알려줘야 한다. 그리고 사설 IP 주소는 ifconfig(8)을 사용하여 설정해야 된다.

네트워크 #1 의 게이트웨이 머신에서 터널을 설정하기 위해 다음 두 명령을 실행한다.

```
# gifconfig gif0 A.B.C.D W.X.Y.Z
# ifconfig gif0 inet 192.168.1.1 192.168.2.1 netmask 0xffffffff
```

다른 머신에서도 같은 명령을 실행하지만 IP 주소의 순서를 반대로 한다.

```
# gifconfig gif0 W.X.Y.Z A.B.C.D
```



---

```
# ifconfig gif0 inet 192.168.2.1 192.168.1.1 netmask 0xffffffff
```

그리고 다음 명령을 실행하여 설정을 볼 수 있다. 예를 들어 네트워크 #1 게이트웨이에서 다음과 같은 내용을 보게 된다:

```
# gifconfig gif0
gif0: flags=8011<UP,POINTTOPOINT,MULTICAST> mtu 1280
inet 192.168.1.1 --> 192.168.2.1 netmask 0xffffffff
physical address inet A.B.C.D --> W.X.Y.Z
```

위에서 보았듯이 터널은 물리적인 주소 A.B.C.D 와 W.X.Y.Z 사이에 생성되고 192.168.1.1 과 192.168.2.1 사이의 트래픽은 터널을 통과하게 된다.

여기서도 **netstat -rn**으로 확인할 수 있는 양쪽 머신의 라우팅 테이블에 엔트리를 추가해야 된다. 다음은 네트워크 #1 의 게이트웨이 호스트를 출력한 것이다.

```
# netstat -rn
Routing tables

Internet:
Destination      Gateway          Flags    Refs    Use    Netif  Expire
...
192.168.2.1      192.168.1.1    UH        0        0     gif0
...
```

"Flags" 값이 보여주듯이 이것은 호스트 라우트다. 이 의미는 각 게이트웨이가 다른 게이트웨이에 어떻게 접근하는지 알고 있지만 각 네트워크의 내부에 어떻게 접근하는지 알지 못하는 뜻이다. 이 문제는 금방 해결된다.

양쪽 머신에 방화벽을 운용하는 것과 비슷하게 VPN 트래픽을 우회시켜야 된다. 양쪽 네트워크 사이의 모든 트래픽을 허용하거나 VPN 한쪽 끝에서 다른 쪽 끝까지 방화벽 룰에 포함시킨다.

VPN 을 통하는 모든 트래픽을 허용하도록 방화벽을 설정했다면 테스트는 아주 쉽다. 게이트웨이 머신에 ipfw(8)을 사용한다면 다음 명령이 양쪽 VPN 사이의 모든 트래픽을 허용한다.

---

```
# ipfw add 1 allow ip from any to any via gif0
```

이 명령을 양쪽 게이트웨이 호스트에서 실행한다.

이것은 각 게이트웨이 머신이 서로 ping 을 허용하도록 한다. 192.168.1.1 에서 다음 명령을 실행해서 응답을 받을 수 있어야 되고 반대편 게이트웨이에서도 똑 같은 결과를 받아야 한다.

```
# ping 192.168.2.1
```

그러나 양쪽 네트워크의 내부 머신에는 아직 연결하지 못한다. 이유는 라우팅 때문이다. 게이트웨이 머신이 반대편에 어떻게 연결하는지 알고 있더라도 양쪽 게이트웨이 내부에 연결하는 방법은 알지 못한다.

이 문제를 해결하기 위해 각 게이트웨이 머신에 고정(static) 라우트를 추가해야 된다. 첫 번째 게이트웨이에서 고정 라우트를 추가하는 명령은 다음과 같다:

```
# route add 192.168.2.0 192.168.2.1 netmask 0xfffff00
```

이 명령은 “네트워크 192.168.2.0 에 있는 호스트에 연결하기 위해 호스트 192.168.2.1 에 패킷을 보낸다”. 다른 쪽 게이트웨이에서도 비슷한 명령을 실행해야 하지만 192.168.1.x 주소 대신 사용한다.

IP 트래픽은 한쪽 네트워크의 호스트에서 다른 네트워크에 도달할 수 있다.

이제 두 네트워크 사이에 2/3 정도의 VPN 이 생성되었다. 아직 사설 쪽은 설정하지 않았다. ping(8)와 tcpdump(1)을 사용하여 이 설정을 테스트 한다. 게이트웨이 호스트에 로그인하여 다음 명령을 실행한다:

```
# tcpdump dst host 192.168.2.1
```

같은 호스트의 다른 로그인 세션에서는 다음 명령을 실행한다:

```
# ping 192.168.2.1
```

다음과 비슷한 결과를 보게 된다:

---

```
16:10:24.018080 192.168.1.1 > 192.168.2.1: icmp: echo request
16:10:24.018109 192.168.1.1 > 192.168.2.1: icmp: echo reply
16:10:25.018814 192.168.1.1 > 192.168.2.1: icmp: echo request
16:10:25.018847 192.168.1.1 > 192.168.2.1: icmp: echo reply
16:10:26.028896 192.168.1.1 > 192.168.2.1: icmp: echo request
16:10:26.029112 192.168.1.1 > 192.168.2.1: icmp: echo reply
```

결과에서 보았듯이 ICMP 메시지는 암호화되지 않은 채 되 돌아온다. 패킷에서 더 많은 데이터 바이트를 캡처하기 위해 tcpdump(1)에 `-s` 매개변수를 사용하였다면 더 많은 정보를 볼 수 있다.

다음 섹션에서 두 네트워크 사이의 보안 링크를 설명하고 나면 모든 트래픽이 자동으로 암호화된다.

#### 요약:

- “pseudo-device gif”로 양쪽 커널을 설정한다
- 게이트웨이 호스트 # 1 의 `/etc/rc.conf` 에 다음 라인을(필요하다면 IP 주소를 변경한다) 추가한다.

```
gifconfig_gif0="A.B.C.D W.X.Y.Z"
ifconfig_gif0="inet 192.168.1.1 192.168.2.1 netmask 0xffffffff"
static_routes="vpn"
route_vpn="192.168.2.0 192.168.2.1 netmask 0xfffff00"
```

- 양쪽 호스트의 방화벽 스크립트에서(`/etc/rc.firewall` 또는 비슷한) 다음 명령을 실행한다.

```
# ipfw add 1 allow ip from any to any via gif0
```

- IP 순서를 반대로해서 게이트웨이 호스트 #2 의 `/etc/rc.conf` 도 변경한다

---

### 14.10.3.2 단계 2: 보안 링크

안전한 링크를 위해 IPsec 를 사용한다. IPsec 는 두 개의 호스트에 동일한 암호화 키를 제공하고 두 호스트 사이의 데이터 암호화에 이 키를 사용하는 메커니즘을 제공한다.

여기서 고려해야 될 두 가지 설정 영역이 있다.

- ① 사용할 암호화 메커니즘에 동의하는 두 호스트를 위한 메커니즘이 있어야 한다. 이 메커니즘에 두 호스트가 동의한다면 이 둘 사이에는 “보안 관계”가 성립한다고 말한다.
- ② 어떤 트래픽을 암호화해야 되는지 지정하는 메커니즘이 있어야 한다. 외부로 나가는 모든 트래픽을 암호화하기를 원치 않을 것이다. VPN 의 일부 트래픽만 암호화하기를 원할 것이다. 어떤 트래픽을 암호화할지 결정하는 룰을 “보안 정책”이라고 한다.

보안 관계와 보안 정책은 커널이 관리하고 유저 기반 프로그램으로 수정할 수 있다. 그러나 사용하기 전에 IPsec 와 Encapsulated Security Payload(ESP) 프로토콜을 지원하도록 커널을 설정해야 된다. 커널 설정파일에 다음 내용을 추가해서 컴파일하고 설치한 후 재 부팅하면 된다:

```
options IPSEC
options IPSEC_ESP
```

양쪽 게이트웨이 호스트의 커널을 위와 같이 설정해야 된다.

보안 관계를 설정할 때 두 가지를 선택할 수 있다. 두 호스트 사이의 암호화 알고리즘과 암호화 키 등을 선택하는 설정이나 Internet Key Exchange 프로토콜(IKE)을 실행하는 데몬을 사용할 수 있다.

개인적으로 후자를 권장하지만 개별적으로 설정하는 것도 어렵지 않다.

보안 정책을 편집하고 표시하려면 setkey(8)을 실행한다. route(8)이 커널의 라우팅 테이블이기 때문에 setkey 는 커널의 보안 정책 테이블이라고 유추할 수 있다. 또한 setkey 는 현재 보안 관계를 보여줄 수 있고 이러한 관점에서 netstat -r 과 유사하다.

FreeBSD 에서 보안 관계를 관리하는 여러 가지 데몬이 있다. 이 문서에서는 이들 데몬 중

---

하나인 racoon 을 어떻게 사용하는지 설명한다. racoon 은 FreeBSD 포트 컬렉션 security/카테고리에 있으며 일반적인 방법으로 설치한다.

racoon 을 양쪽 게이트웨이 호스트에서 실행해야 된다. 양쪽 호스트에서 VPN 의 반대편 IP 와 보안 키로(양쪽 게이트웨이에 동일한 키) 설정한다.

두 데몬은 각자 연결하여 서로 누구인지 확인한다(설정한 보안 키를 사용하여). 그리고 데몬은 새로운 보안 키를 생성해서 VPN 을 통한 트래픽 암호화에 이 키를 사용한다. 키를 정기적으로 변경하기 때문에 두 데몬이 다른 키를 선택했을 때 공격자가 이전 키를 크랙했다 라도 큰 영향을 받지 않는다.

racoon 설정은  $\${PREFIX}/etc/racoon$  에 저장되므로 약간만 수정하면 되는 설정파일을 이곳에서 찾을 수 있다. 변경해야 되는 racoon 의 다른 컴포넌트는 “pre-shared key”다.

기본 racoon 설정은 이 키를  $\${PREFIX}/etc/racoon/psk.txt$  에서 찾는다. pre-shared key 는 VPN 링크를 통과하는 트래픽을 암호화할 때 사용하는 키가 아니고 키 관리 데몬이 다른 데몬과 신뢰하는데 사용하는 단순한 토큰이다.

psk.txt 에는 여러분이 관심을 가져야 되는 각 원격 사이트를 위한 라인을 가지고 있다. 이 예제에서는 상대 사이트를 위한 한 라인을(VPN 의 양쪽 단은 다른 쪽과 통신하기 때문이다) 각 psk.txt 파일에 가지고 있는 두 사이트가 있다.

게이트웨이 호스트 #1 의 라인은 다음과 비슷할 것이다:

W.X.Y.Z	secret
---------	--------

이것은 반대편의 *공인* IP 주소고 보안을 제공하는 텍스트 문자열이다. 여러분은 “secret”를 키로 사용하지 않는다.

A.B.C.D	secret
---------	--------

역시 같은 보안 키를 가진 반대편의 공인 IP 다. racoon 을 실행하기 전에 psk.txt 는 0600 모드여야(root 만 읽고 쓸 수 있도록) 된다.

양쪽 게이트웨이 머신에서 racoon 을 실행한다. ISAKMP(Internet Security Association Key

---

Management Protocol) 포트로 UDP 를 전송하는 IKE 트래픽을 허용하도록 방화벽 룰을 추가해야 된다.

```
# ipfw add 1 allow udp from A.B.C.D to W.X.Y.Z isakmp
# ipfw add 1 allow udp from W.X.Y.Z to A.B.C.D isakmp
```

racoon 이 실행되면 ping 을 한쪽 게이트웨이에서 다른 쪽으로 테스트할 수 있다. 연결은 아직 암호화되지 않지만 racoon 은 양쪽 호스트 사이에 보안 관계를 설정한다. 따라서 시간이 좀 소요될 것이고 ping 명령에 대한 응답을 받기 전에 약간 지체될 수 있다.

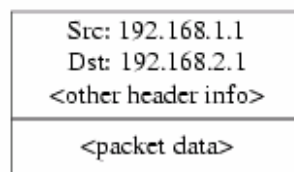
보안 관계가 설정되면 setkey(8)을 사용하여 설정된 것을 볼 수 있다. 보안 관계 정보를 보려는 호스트에서 **setkey -D** 를 실행한다.

이제 받은 해결하였다. 남은 문제는 보안 정책설정이다.

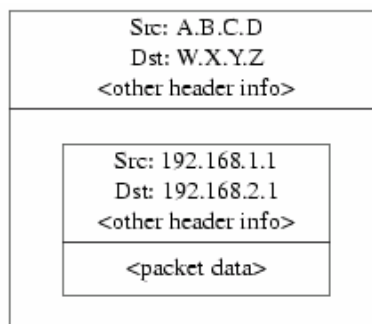
현명한 보안 정책을 수립하기 위해 설정한 것을 다시 확인한다

여러분이 보낸 각 IP 패킷에는 패킷에 대한 정보를 가지고 있는 헤더를 가지고 있다. 헤더는 소스와 목적지 IP 주소를 가지고 있다. 이미 알고 있듯이 192.168.x.y 와 같은 사설 IP 주소 범위는 공인 인터넷에 나타나지 않는다. 대신 이들 IP 는 다른 패킷에 캡슐화(encapsulate) 된다. 이 패킷은 공인 소스와 목적지 IP 주소로 사설 주소를 대체한다.

외부로 나가는 패킷이 다음과 같이 생겼다면:



다음과 같이 다른 패킷으로 캡슐화된다:



---

이 캡슐화는 gif 장치로 수행한다. 그림에서처럼 패킷은 이제 실제 IP 를 가지고 외부로 나가고 최초의 패킷은 인터넷으로 나가는 패킷 속에 싸여있다.

그리고 우리는 VPN 사이의 모든 트래픽이 암호화되기를 원한다. 이 말을 다시 설명하면:

“A.B.C.D 에서 출발하고 목적지가 W.X.Y.Z 라면 필요한 보안 관계를 사용하여 암호화시킨다”

“패킷이 W.X.Y.Z 에서 도착하고 목적지가 A.B.C.D 라면 필요한 보안 관계를 사용하여 복호화시킨다”

이 설정은 거의 근접했지만 정확하지는 않다. 이렇게 했다면 VPN 의 일부가 아닌 W.X.Y.Z 에서 출발하거나 W.X.Y.Z 에 도착하는 모든 트래픽이 암호화된다. 이 설정은 여러분이 원하는 것이 아니다. 정확한 정책은 다음과 같다:

“다른 패킷으로 캡슐화된 패킷이 A.B.C.D 에서 출발하고 목적지가 W.X.Y.Z 라면 필요한 보안 관계로 암호화한다”

“다른 패킷으로 캡슐화된 패킷이 W.X.Y.Z 에서 도착하고 목적지가 A.B.C.D 라면 필요한 보안 관계로 복호화한다”

약간의 변경이지만 필요한 요소다.

보안 정책도 setkey(8)을 사용하여 설정할 수 있다. setkey(8) 기능은 정책을 정의하기 위한 언어다. 표준입력으로 설정을 입력하거나 설정 정보를 가진 파일 이름을 지정하기 위해 -f 옵션을 사용할 수 있다.

W.X.Y.Z 로 나가는 모든 트래픽을 암호화시키는 게이트웨이 호스트 #1 의 설정은 다음과 같다:

```
spdadd A.B.C.D/32 W.X.Y.Z/32 ipencap -P out ipsec esp/tunnel/A.B.C.D-W.X.Y.Z/require;
```

위의 내용을 파일(예를 들어 /etc/ipsec.conf)에 넣고 다음 명령을 실행한다:

```
# setkey -f /etc/ipsec.conf
```

---

*spdadd*는 보안 정책 데이터베이스에 룰을 추가하도록 *setkey(8)*에게 지시한다. 이 라인의 나머지는 이 정책에 어떤 패킷이 대응되는지 지정한다. A.B.C.D/32 와 W.X.Y.Z/32 는 IP 주소와 이 정책을 적용할 네트워크나 호스트를 확인하는 넷 마스크다. 우리는 이들 두 호스트 사이의 트래픽에 적용하기를 원한다. *ipencap*는 다른 패킷을 캡슐화하는 패킷에만 이 정책을 적용하라고 커널에게 지시한다. *-P out*은 외부로 나가는 패킷에만 이 정책을 적용하라는 것이고 *ipsec*는 패킷이 안전하다는 것을 보여준다.

두 번째 라인은 이 패킷을 어떻게 암호화하는지 지정한다. *esp*는 사용할 프로토콜이지만 *tunnel*은 패킷이 IPsec 패킷으로 캡슐화된다는 것을 보여준다. 계속 사용되는 A.B.C.D 와 W.X.Y.Z는 사용할 보안 관계를 선택하는데 사용되고 마지막 *require*는 이 룰에 대응된다면 이 패킷을 암호화하라는 명령이다.

이 룰은 외부로 나가는 패킷에만 적용된다. 내부로 들어오는 패킷에도 비슷한 룰이 필요하다.

```
spdadd W.X.Y.Z/32 A.B.C.D/32 ipencap -P in ipsec esp/tunnel/W.X.Y.Z-A.B.C.D/require;
```

이 경우 *out* 대신 *in*을 사용하고 IP 주소를 반대로 입력한다.

다른 게이트웨이 호스트에도(공인 IP 주소 W.X.Y.Z를 가지고 있는) 비슷한 룰이 필요하다.

```
spdadd W.X.Y.Z/32 A.B.C.D/32 ipencap -P out ipsec esp/tunnel/W.X.Y.Z-A.B.C.D/require;
```

```
spdadd A.B.C.D/32 W.X.Y.Z/32 ipencap -P in ipsec esp/tunnel/A.B.C.D-W.X.Y.Z/require;
```

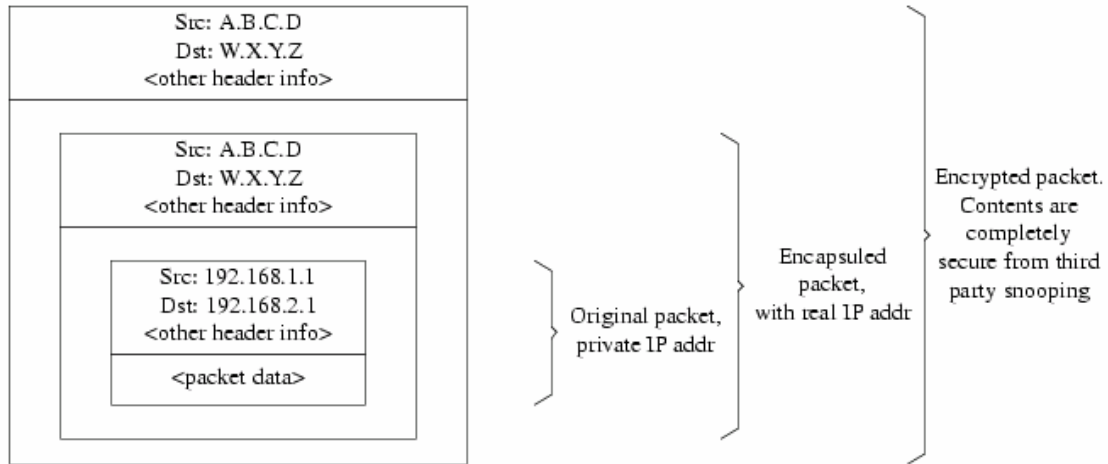
마지막으로 ESP 와 IPENCAP 패킷을 정방향과 역방향으로 허용하도록 방화벽에 룰을 추가한다. 이들 룰을 양쪽 호스트에 추가해야 된다.

```
# ipfw add 1 allow esp from A.B.C.D to W.X.Y.Z
# ipfw add 1 allow esp from W.X.Y.Z to A.B.C.D
# ipfw add 1 allow ipencap from A.B.C.D to W.X.Y.Z
# ipfw add 1 allow ipencap from W.X.Y.Z to A.B.C.D
```

룰이 대칭되기 때문에 각 게이트웨이 호스트에 같은 룰을 사용할 수 있다.



외부로 나가는 패킷은 이제 다음과 비슷할 것이다:



이들 패킷이 VPN의 끝에 도착하면 복호화(racoon으로 설정한 보안 관계를 사용하여)된다. 그리고 이들 패킷은 내부 네트워크에서 이동할 수 있는 패킷만 남을 때까지 두 번째 레이어를 벗겨내는 gif 인터페이스에 들어간다.

이전처럼 ping(8) 테스트를 사용하여 상태를 체크할 수 있다. A.B.C.D 게이트웨이 머신에 로그인하여 다음 명령을 실행한다:

```
# tcpdump dst host 192.168.2.1
```

같은 호스트의 다른 로그인 세션에서는 다음 명령을 실행한다:

```
# ping 192.168.2.1
```

이때 다음과 같은 결과를 볼 수 있다:

```
XXX tcpdump output
```

앞에서 보았듯이 tcpdump(1)가 ESP 패킷을 보여준다. -s 옵션을 사용했다면 암호화되었기 때문에 알 수 없는 문자를(외관상) 보게 된다.

축하한다! 두 개의 원격 사이트 사이에 VPN을 설정했다.

---

## 요약

- 양쪽 커널을 다음 내용으로 설정한다:

```
options IPSEC
options IPSEC_ESP
```

- `security/racoon` 을 설치한다. 양쪽 게이트웨이 호스트의 `${PREFIX}/etc/racoon/psk.txt` 파일에 원격 호스트의 IP 주소 엔트리와 양쪽 호스트가 알고 있는 보안 키를 추가한다. 이 파일을 0600 모드로 변경한다.
- 각 호스트의 `/etc/rc.conf` 에 다음 라인을 추가한다:

```
ipsec_enable="YES"
ipsec_file="/etc/ipsec.conf"
```

- 필요한 `spdadd` 라인을 가지고 있는 `/etc/ipsec.conf` 파일을 각 호스트에 생성한다.

게이트웨이 호스트 #1 에서는 다음과 같다:

```
spdadd A.B.C.D/32 W.X.Y.Z/32 ipencap -P out ipsec
    esp/tunnel/A.B.C.D-W.X.Y.Z/require;
spdadd W.X.Y.Z/32 A.B.C.D/32 ipencap -P in ipsec
    esp/tunnel/W.X.Y.Z-A.B.C.D/require;
```

게이트웨이 호스트 #2 는 다음과 같다:

```
spdadd W.X.Y.Z/32 A.B.C.D/32 ipencap -P out ipsec
    esp/tunnel/W.X.Y.Z-A.B.C.D/require;
spdadd A.B.C.D/32 W.X.Y.Z/32 ipencap -P in ipsec
    esp/tunnel/A.B.C.D-W.X.Y.Z/require;
```

- 
- 양쪽 호스트 사이의 IKE, ESP 와 IPENCAP 트래픽을 허용하는 룰을 방화벽에 추가한다:

```
# ipfw add 1 allow udp from A.B.C.D to W.X.Y.Z isakmp
# ipfw add 1 allow udp from W.X.Y.Z to A.B.C.D isakmp
# ipfw add 1 allow esp from A.B.C.D to W.X.Y.Z
# ipfw add 1 allow esp from W.X.Y.Z to A.B.C.D
# ipfw add 1 allow ipencap from A.B.C.D to W.X.Y.Z
# ipfw add 1 allow ipencap from W.X.Y.Z to A.B.C.D
```

이전에 선행해야 되는 두 단계는 VPN 을 up 해서 실행할 수 있어야 된다. 각 네트워크에 있는 머신은 IP 주소를 사용하여 다른 쪽의 머신을 참조할 수 있어야 되고 링크를 통과하는 모든 트래픽은 자동으로 암호화된다.

## 14.11 OpenSSH

OpenSSH 는 원격 머신에 안전하게 접근하는데 사용하는 네트워크 연결 툴이다. 이 툴은 rlogin, rcp 와 telnet 을 대체할 수 있다. 게다가 다른 TCP/IP 연결도 SSH 를 통하여 안전하게 터널/포워드 할 수 있다. OpenSSH 로 암호화된 모든 트래픽은 감청되거나 연결 탈취와 다른 네트워크 레벨의 공격을 효과적으로 막을 수 있다.

OpenSSH 는 OpenBSD 프로젝트에 의해 관리되고 최근에 모든 버그가 수정된 SSH v1.2.12 기반으로 업데이트되었다. 이것은 SSH 프로토콜 1 과 2 에 호환된다. OpenSSH 는 FreeBSD 4.0 부터 기본 시스템에 포함되었다.

### 14.11.1 OpenSSH 사용의 이점

보통 telnet(1)이나 rlogin(1)을 사용할 때 데이터는 암호화되지 않은 평범한 텍스트 형태로 네트워크에 전송된다. 네트워크 스니퍼는 클라이언트 서버간 어느 곳에서든 유저/패스워드 정보나 데이터 전송을 캡처할 수 있다. OpenSSH 는 이런 일이 발생하지 않도록 다양한 인증과 암호화 방법을 제공한다.

---

## 14.11.2 sshd 활성화하기

rc.conf 파일에 다음 라인을 추가한다:

```
sshd_enable="YES"
```

시스템이 다음에 초기화될 때 **OpenSSH** 데몬 프로그램 `sshd(8)`을 로드한다. 아니면 명령어 라인에서 `sshd`를 입력하여 직접 **sshd** 데몬을 실행할 수 있다.

## 14.11.3 SSH 클라이언트

`ssh(1)` 유틸리티는 간단히 `rlogin(1)`과 동작한다.

```
# ssh user@example.com
Host key not found from the list of known hosts.
Are you sure you want to continue connecting (yes/no)? yes
Host 'example.com' added to the list of known hosts.
user@example.com's password: *****
```

세션이 `rlogin`이나 `telnet`을 사용하여 만들어졌다면 로그인 세션은 계속 진행된다. SSH는 클라이언트가 연결할 때 서버의 인증을 확인하기 위해 키 인증 시스템을 사용한다. 유저는 처음 연결할 때 프롬프트에 `yes`만 입력하면 된다. 이 후의 모든 로그인은 저장된 인증 키로 확인된다. SSH 클라이언트는 나중에 로그인할 때 받은 인증키가 저장된 인증키와 다를 경우 경고를 준다. 인증키는 `~/.ssh/known_hosts` 저장되거나 SSH v2 인증키는 `~/.ssh/known_hosts2`에 저장된다.

기본적으로 **OpenSSH** 서버는 SSH v1와 SSH v2 연결을 허용하도록 설정되어있다. 그러나 클라이언트는 두 가지 중 하나를 선택할 수 있다. 버전 2는 이전 버전보다 더 강력하고 안전하다.

`ssh`는 `-1` 또는 `-2` 인수를 지정하여 v1과 v2 프로토콜을 강제로 선택할 수 있다.

---

## 14.11.4 안전한 복사

scp(1) 명령은 rcp(1)와 비슷하게 동작한다; 보안을 제외하고 원격 머신에서 파일을 복사할 수 있다.

```
# scp user@example.com:/COPYRIGHT COPYRIGHT
user@example.com's password: *****
COPYRIGHT          100% |*****| 4735
00:00
#
```

왜냐하면 인증키는 이전 예제에서 이미 호스트에 저장되어있기 때문에 여기서 scp(1)를 사용하면 바로 확인된다.

첫 번째 인자에 소스 파일을 지정하고 두 번째에 목적지 파일을 지정하는 scp(1) 인자는 cp(1)와 비슷하다. 왜냐하면 SSH 를 사용하는 네트워크를 거쳐 파일이 복사되므로 한 개 또는 더 많은 파일인자를 이 형식에 추가할 수 있다.

기본 형식은 다음과 같다.

```
user@host:<path_to_remote_file>.
```

## 14.11.5 설정

OpenSSH 데몬과 클라이언트 시스템의 다양한 설정파일은 /etc/ssh 디렉터리에 있다.

ssh\_config 는 클라이언트 설정이지만 sshd\_config 는 데몬 설정이다.

추가적으로 sshd\_program(기본적으로 /usr/sbin/sshd)과 sshd\_flags rc.conf 옵션은 더 많은 설정레벨을 제공한다.

## 14.11.6 ssh-keygen

패스워드를 사용하는 대신 ssh-keygen(1)으로 유저 인증에 사용하려는 RSA 키를 생성할 수 있다.

```
% ssh-keygen -t rsa1
Initializing random number generator...
Generating p: .++ (distance 66)
Generating q: .....++ (distance 498)
Computing the keys...
Key generation complete.
Enter file in which to save the key (/home/user/.ssh/identity):
Enter passphrase:
Enter the same passphrase again:
Your identification has been saved in /home/user/.ssh/identity.
...
```

ssh-keygen(1)은 인증에 사용되는 공개키와 개인키 쌍을 생성한다. 개인키는 ~/.ssh/identity 에 공개키는 ~/.ssh/identity.pub 에 저장된다. 공개키를 연결하려는 원격 머신의 ~/.ssh/authorized\_keys 에 저장한다.

따라서 패스워드 대신 RSA 인증으로 원격 머신에 연결할 수 있다.

**Note:** `-t rsa1` 옵션은 SSH 프로토콜 1 버전을 사용하도록 RSA 키를 생성한다. SSH 프로토콜 2 버전의 RSA 키를 사용하려면 `ssh-keygen -t rsa` 명령을 사용한다.

ssh-keygen(1)에 패스워드를 사용했다면 유저는 개인키를 사용하기 위해 매번 패스워드를 입력하는 프롬프트를 보게 된다.

SSH 프로토콜 버전 2 DSA 키는 같은 목적을 위해 `ssh-keygen -d` 명령으로 생성할 수 있다. 이 명령은 SSH 프로토콜 버전 2 세션만 사용하도록 공개/개인 DSA 키를 생성한다. 공개 키는 ~/.ssh/id\_dsa.pub 에 개인키는 ~/.ssh/id\_dsa 에 저장된다.

DSA 공개키는 원격 머신의 ~/.ssh/authorized\_keys 에 있어야 된다.

ssh-agent(1)과 ssh-add(1)은 여러 개의 패스워드로 된 개인키 관리에 사용된다.

**주의:** 시스템의 OpenSSH 버전에 따라 다양한 옵션과 파일을 지정할 수 있다. 문제를 예방하기 위해 ssh-keygen(1) 매뉴얼 페이지를 참고한다.

---

## 14.11.7 SSH 터널링

OpenSSH 는 암호화 세션에서 다른 프로토콜을 캡슐화하는 터널을 생성할 수 있다.

다음 명령이 ssh(1)가 **telnet** 터널을 생성하는 것을 보여준다.

```
% ssh -2 -N -f -L 5023:localhost:23 user@foo.example.com
%
```

ssh 명령은 다음 옵션들과 사용된다.

**-2**

강제로 ssh 프로토콜 버전 2 를 사용한다(예전의 SSH 서버에는 사용하지 않는다).

**-N**

명령이 없고 터널만 있음을 명시한다. 생략했다면 ssh 는 일반 세션을 시작한다.

**-f**

강제로 ssh 를 백그라운드에서 실행한다.

**-L**

로컬 터널을 *localport:remotehost:remoteport* 형식으로 보여준다.

*user@foo.example.com*

원격 SSH 서버를 의미한다.

---

SSH 터널은 localhost의 지정된 포트에 listen 소켓을 생성한다. 그리고 로컬 호스트/포트에 SSH를 통해 요청되는 모든 연결을 지정된 원격 호스트와 포트로 포워드한다.

예제에서 localhost의 포트 5023은 원격 머신의 포트 23번으로 포워드된다. 23번은 telnet이기 때문에 SSH 터널을 통해 보안 telnet 세션을 생성한다.

이 방법으로 SMTP, POP3, FTP 등과 같은 비 보안 TCP 프로토콜을 감쌀 수 있다.

#### 예제 14-1. SMTP 보안 터널 생성에 SSH 사용하기

```
% ssh -2 -N -f -L 5025:localhost:25 user@mailserver.example.com
user@mailserver.example.com's password: *****
% telnet localhost 5025
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 mailserver.example.com ESMTP
```

이 명령을 ssh-keygen(1)와 사용하여 더욱 깨끗하고/안전한 SSH 터널링 환경에서 유저 계정을 추가할 수 있다. 패스워드를 입력하는 대신 키를 사용할 수 있고, 터널은 각 유저 단위로 실행할 수 있다.

#### [예제: 실질적인 SSH 터널링 예제]

##### 1. POP3 서버에 안전하게 접근하기

외부의 접근을 허용하는 SSH 서버가 사무실에 있다. 같은 사무실 내부 네트워크에 POP3 서버가 운용되는 메일 서버가 있다. 로컬 네트워크나 여러분의 집과 사무실 사이의 네트워크 경로는 완벽하게 신뢰할 수 있거나 신뢰할 수 없을 것이다. 안전하게 메일을 체크하려면 사무실의 SSH 서버와 SSH로 연결해서 터널을 거쳐 메일 서버에 도달할 수 있다.

```
% ssh -2 -N -f -L 2110:mail.example.com:110 user@ssh-server.example.com
user@ssh-server.example.com's password: *****
```



---

터널이 실행되고있을 때 메일 클라이언트의 POP3 요청을 localhost 포트 2110 으로 지정하여 보낼 수 있다. 이곳의 연결은 안전하게 터널을 거쳐 mail.example.com 으로 포워드된다.

## 2. 엄격한 방화벽 우회하기

어떤 네트워크 관리자는 들어오는 연결과 나가는 연결을 필터링하는 아주 강력한 방화벽 룰을 정하여 적용한다. 외부로 연결되는 것은 오직 포트 22 번과 80 번으로 SSH 와 웹 서핑만할 수 있다.

음악을 듣기 위해 Ogg Vorbis 서버 같은 서비스(업무와 관련 없는)를 사용하고 싶을 것이다. 이 Ogg Vorbis 서버가 22 이나 80 이 아닌 다른 포트로 음악을 서비스한다면 사용할 수 없을 것이다.

해결책은 방화벽 외부 머신에 SSH 터널을 생성해서 이 터널을 이용하여 Ogg Vorbis 서버를 사용한다.

```
% ssh -2 -N -f -L 8888:music.example.com:8000 user@unfirewalled-system.example.org
user@unfirewalled-system.example.org's password: *****
```

이제 music.example.com 의 8000 포트로 포워드하는 localhost 8888 포트로 클라이언트 프로그램을 지정하여 방화벽을 우회할 수 있다.

## 14.11.8 더 읽을만한 내용

OpenSSH(<http://www.openssh.com/>)

ssh(1) scp(1) ssh-keygen(1) ssh-agent(1) ssh-add(1)

sshd(8) sftp-server(8)

---

## 14.12 파일시스템 접근 제어 리스트

스냅샷과 같은 파일시스템의 증가로 FreeBSD 5.0 과 이후 버전은 파일시스템 접근제어 리스트(ACL) 보안을 제공한다.

접근제어 리스트는 높은 호환성으로(POSIX@.1e) 표준 유닉스 퍼미션 모델을 확장한다. 이 기능으로 관리자는 더욱 강력한 보안 모델의 이점을 얻을 수 있다.

UFS 파일시스템에 ACL 을 지원하려면 다음 라인을 커널에 컴파일해야 된다:

`options UFS_ACL`

이 옵션을 컴파일하지 않고 ACL 을 지원하는 파일시스템을 마운트하려고 하면 경고 메시지가 나타난다. 이 옵션은 GENERIC 커널에 포함되어있다. ACL 은 파일시스템에서 확장된 기능이다. 확장된 기능은 차세대 유닉스 파일시스템 UFS2 에서 자연스럽게 지원된다.

ACL 은 /etc/fstab 에 추가해야 되는 mount-time 관리 플래그로(*ac/s*) 활성화된다. mount-time 플래그는 파일시스템 헤더의 슈퍼블록 ACL 플래그를 수정하는 tunefs(8)을 사용하여 자동으로 영구히 설정할 수도 있다. 보통 여러 가지 이유에서 슈퍼블록 플래그를 사용하는 것이 더 선호된다:

- mount-time ACL 플래그는 remount 로(mount(8) *-u*) 변경할 수 없고 완벽하게 umount(8)한 후 새로 mount(8)해야 된다. 이 의미는 부팅 후 ACL 을 root 파일 시스템에서 활성화할 수 없다는 것이다. 그리고 한번 사용되면 파일시스템의 배열을 변경할 수도 없다.
- 슈퍼블록 플래그를 설정하면 fstab 엔트리가 없거나 장치가 재 배열되더라도 항상 ACL 이 활성화된 채 파일시스템이 마운트된다. 이 방법은 ACL 이 활성화되지 않은 파일시스템이 갑자기 마운트되어 ACL 상태가 적절하지 않아서 유발할 수 있는 보안 문제를 방지한다.

**Note:** 완벽하게 새로 mount(8)하지 않아도 플래그를 허용하도록 ACL 동작을 변경할 수 있지만 ACL 이 정확히 활성화되지 않고 갑자기 마운트되면 도끼로 발등 찍히는 일이 될 수 있다. 따라서 ACL 이 활성화됐다면 비활성하고 확장된 속성을 그대로 두고 다시 활성화한다. 일반적으로 파일시스템에서 ACL 을 활성화했다면 다시 비활성하지 않아야 된다. 왜냐하면 파일 보호가 시스템의 유저와 호환되지

---

않게 되고 ACL 을 다시 활성화하면 퍼미션이 변경된 후의 이전 ACL 이 파일에 추가되어 결과를 예측할 수 없게 된다.

ACL 이 활성화된 파일시스템은 다음과 같이 파일 퍼미션 설정을 볼 때 +(플러스) 표시로 나타난다.

```
drwx----- 2 robert  robert  512 Dec 27 11:54 private
drwxrwx---+ 2 robert  robert  512 Dec 23 10:57 directory1
drwxrwx---+ 2 robert  robert  512 Dec 22 10:20 directory2
drwxrwx---+ 2 robert  robert  512 Dec 27 11:57 directory3
drwxr-xr-x  2 robert  robert  512 Nov 10 11:54 public_html
```

여기 보고 있는 directory1, directory2 와 directory3 디렉터리는 모두 ACL 의 이점을 가지고 있다. public\_html 디렉터리는 아니다.

## 14.12.1 ACL 사용

파일시스템 ACL 은 `getfacl(1)` 유틸리티로 볼 수 있다. 예를 들면 `test` 파일의 ACL 설정을 보려면 다음 명령을 사용한다:

```
% getfacl test
#file:test
#owner:1001
#group:1001
user::rw-
group::r--
other::r-
```

이 파일의 ACL 설정을 변경하려면 `setfacl(1)` 유틸리티를 실행한다.

```
% setfacl -k test
```

`-k` 옵션은 파일이나 파일시스템에 현재 설정되어 있는 모든 ACL 을 삭제한다. 좀더 바람직한 방법은 ACL 의 기본 필드는 남겨두는 `-b` 를 사용한다.

---

```
% setfacl -m u:trhodes:rwx,group:web:r--,o::--- test
```

앞서 설명한 명령의 `-m` 옵션은 기본 ACL 엔트리를 수정할 때 사용된다. 이전 명령으로 모든 정의가 삭제되어 이전에 정의된 엔트리가 없기 때문에 이 명령은 기본 옵션을 복구하고 나열된 옵션을 할당한다. 시스템에 없는 유저나 그룹을 추가한다면 “Oinvalid argument” 에러가 표준 출력으로 표시되므로 주의한다.

## 14.13 FreeBSD 보안 권고문

다른 운영체제처럼 FreeBSD 도 “보안 권고문”을 발행한다. 보통 이들 권고문은 메일을 사용하여 통보되고 적절한 릴리즈가 패치 된 후 Errata 에 보관된다. 이번 섹션에서는 권고문이 무엇이고 어떻게 이해해야 되며 시스템을 패치하기 위해 무엇이 필요한지 설명한다.

### 14.13.1 권고문은 어떻게 생겼는가?

FreeBSD 보안 권고문은 아래와 비슷하게 생겼고 FreeBSD 보안 공고 메일링 리스트에서 (<http://lists.freebsd.org/mailman/listinfo/freebsd-security-notifications>) 볼 수 있다.

---

FreeBSD-SA-XX:XX.UTIL

Security Advisory

The FreeBSD Project

Topic: denial of service due to some problem ❶

Category: core ❷

Module: sys ❸

Announced: 2003-09-23 ❹

Credits: Person@EMAIL-ADDRESS ❺

Affects: All releases of FreeBSD ❻

---

FreeBSD 4-STABLE prior to the correction date

Corrected: 2003-09-23 16:42:59 UTC (RELENG\_4, 4.9-PRERELEASE)  
2003-09-23 20:08:42 UTC (RELENG\_5\_1, 5.1-RELEASE-p6)  
2003-09-23 20:07:06 UTC (RELENG\_5\_0, 5.0-RELEASE-p15)  
2003-09-23 16:44:58 UTC (RELENG\_4\_8, 4.8-RELEASE-p8)  
2003-09-23 16:47:34 UTC (RELENG\_4\_7, 4.7-RELEASE-p18)  
2003-09-23 16:49:46 UTC (RELENG\_4\_6, 4.6-RELEASE-p21)  
2003-09-23 16:51:24 UTC (RELENG\_4\_5, 4.5-RELEASE-p33)  
2003-09-23 16:52:45 UTC (RELENG\_4\_4, 4.4-RELEASE-p43)  
2003-09-23 16:54:39 UTC (RELENG\_4\_3, 4.3-RELEASE-p39) ⑦

FreeBSD only: NO ③

For general information regarding FreeBSD Security Advisories,  
including descriptions of the fields above, security branches, and the  
following sections, please visit  
<http://www.FreeBSD.org/security/>.

I. Background ⑨

II. Problem Description ⑩

III. Impact(11)

IV. Workaround(12)

---

V. Solution(13)

VI. Correction details(14)

VII. References(15)

- ❶ *Topic* 필드는 문제가 정확히 무엇인지 보여준다. 기본적으로 현재 보안 권고안을 설명하고 취약점과 유틸리티를 표시한다.
- ❷ *Category*는 문제가 되는 시스템 부분(*core*, *contrib* 또는 *ports*가 될 수 있다)을 알려준다. *core* 카테고리의 의미는 취약점이 FreeBSD 운영체제의 핵심 컴포넌트에 영향을 미친다는 것이다. *contrib* 카테고리의 의미는 취약점이 **sendmail** 처럼 FreeBSD 프로젝트에 의해 작성된 소프트웨어에 영향을 미친다는 것이다. 마지막으로 *ports* 카테고리는 포트 컬렉션으로 사용할 수 있는 소프트웨어에 영향을 미친다는 것이다.
- ❸ *Module* 필드는 예를 들어 *sys* 처럼 컴포넌트 위치를 설명한다. 이 예제에서는 우리가 보고 있는 *sys*에 영향을 받기 때문에 이 취약점은 커널이 사용하는 컴포넌트에 영향을 끼친다.
- ❹ *Announced* 필드는 보안 권고가 발행되거나 세계로 발표된 날짜를 반영한다. 이 의미는 보안팀이 이 문제를 확인했고 이 패치가 FreeBSD 소스 저장소에 반영됐다는 것이다.
- ❺ *Credits* 필드는 취약점을 발견해서 보고한 개인이나 단체를 명시해준다.
- ❻ *Affects* 필드는 이 취약점이 어떤 FreeBSD 릴리즈에 영향을 주는지 설명한다. 커널의 경우 영향을 받는 파일의 *ident* 결과를 빨리 조회하면 버전을 확인하기 쉽다. 포트는 */var/db/pkg*의 포트이름 뒤에 나열된 버전번호를 확인한다. 시스템을

---

FreeBSD CVS 저장소와 동기화되지 않고 한번씩 다시 빌드했다면 확인할 수 없다.

- ⑦ *Corrected* 필드는 날짜와 시간 그리고 시간 오프셋과 일치하는 릴리즈를 표시한다.
- ⑧ *FreeBSD only* 필드는 이 취약점이 FreeBSD 에만 영향을 주는지 또는 다른 운영체제에도 영향을 주는지 표시한다.
- ⑨ *Background* 필드는 정확히 영향을 받는 유틸리티가 무엇인지 알려준다. 이 유틸리티가 왜 FreeBSD 에 있으며 무엇에 쓰는지 그리고 개발된 배경에 대한 간략한 정보를 준다.
- ⑩ *Problem Description* 필드는 보안 문제에 대한 더 자세한 정보를 제공한다. 여기서는 결함이 있는 코드나 어떻게 이 유틸리티가 보안 문제를 유발하는 악의적인 목적에 사용되었는지에 대한 정보를 포함한다.
- (11) *Impact* 필드는 어떤 종류의 영향을 시스템에 끼칠 수 있는지 설명한다. 예를 들어 서비스 거부 공격, 유저가 더 많은 권한 이용가능, 공격자의 슈퍼유저 권한획득 등이 포함된다.
- (12) *Workaround* 필드는 시스템을 업그레이드할 수 없는 시스템 관리자가 문제를 해결할 수 있는 방법을 제공한다. 이것은 시간적인 제약이나 네트워크 유용성 또는 다른 많은 이유로 인한 것이다. 어떤 경우든 보안을 가볍게 치부하면 안되고 영향을 받는 시스템은 패치하거나 보안 문제를 해결해야 된다.
- (13) *Solution* 필드에서는 시스템 패치를 설명한다. 이것은 단계적으로 테스트되었고 시스템을 패치하여 보안 문제를 검증한다.
- (14) *Correction Details* 필드는 문자에 밑줄을 쳐서 변경된 기간별로 CVS 분기나 릴리즈 이름을 표시한다. 각 분기별로 영향을 받는 파일의 수정된 횟수도 보여준다.
- (15) *References* 필드는 소스의 다른 정보를 제공한다. 이곳에는 웹 URL, 책, 메일링 리스트와 뉴스그룹을 포함할 수 있다.

---

## 15 강제 접근제어

### 15.1 개요

더 넓은 세계로 뻗어 나가기 위해 더 안전한 보안 환경의 필요성이 증가하고 있다. 이러한 요구로 보안 이외의 사항은 고려하지 않는 TrustedBSD 프로젝트가 시작되었다.

TrustedBSD 프로젝트는 POSIX@.1e 초안에 기반한 유저기반 유틸리티와 커널 인터페이스를 개발하여 FreeBSD 5.X 에 합병하는 것을 목표로 하고 있다. 아직은 개발 단계지만 대부분의 기능이 거의 안정화되었다. 여기에 파일시스템 접근제어 리스트(ACLs)와 강제 접근제어(MAC) 메커니즘이 포함된다.

그럼 MAC 이란 무엇인가? 강제 접근제어(Mandatory Access Controls)는 인증을 받지 않고 시스템이나 유저 데이터에 접근하는 것을 금지하기 위해 유저를 제어하는 룰이다; 또는 시스템 주체(object)나 객체(subject)에 완벽한 무결성을 제공하기 위한 것이다. 주체와 객체의 정의는 아래 내용을 본다. 정의의 필수요소는 관리자와 시스템이 제어하는 일방적인 접근제어이고 유저는 결정권이 없다.

이번 장 전체는 여기서 간단히 MAC 이라고 하는, 주로 강제 접근제어(Mandatory Access Control) 프레임 워크 기능에 대해 설명한다. 더 많은 기능이 FreeBSD 5.X 에 추가되기 때문에 추가된 기능은 이곳에서 추가적으로 설명할 것이다.

이번 장을 읽고 다음 사항을 알 수 있다:

- 현재 FreeBSD 에 포함되어 있는 MAC 모듈을 소개하고 관련된 정책을 설명한다.
- 라벨이 할당된 곳과 그렇지 않은 곳의 정책 사이에 어떤 MAC 정책을 사용할 수 있는가
- MAC 프레임 워크를 사용하도록 시스템을 어떻게 효과적으로 설정할 수 있는가
- MAC 모듈을 사용하여 다른 정책은 어떻게 설정하는가
- MAC 프레임 워크를 사용하여 안전한 환경을 어떻게 구축하는가



- 
- 시스템이 적절히 동작하는지 확인하기 위해 MAC 설정은 어떻게 테스트하는가

이번 장을 읽기 전에 다음 사항을 알고 있어야 된다:

- 유닉스와 FreeBSD 기본을 이해해야 된다(3 장).
- 최신 FreeBSD 소스로 유지할 수 있어야 되고 커널 설정/컴파일의 기본을 알고 있어야 된다(8 장).
- 보안 사항을 이해하고 FreeBSD 에 적용할 수 있어야 한다(14 장).

**주의:** 다음 정보를 잘못 적용하면 시스템을 사용하기 어렵게 되어 유저들이 불편을 호소하고 Xfree86 이 제공하는 기능을 사용할 수 없게 된다. 그리고 MAC 이 완벽한 시스템 보안을 제공하는 것이 아니라는 점을 염두 해둔다. MAC 프레임 워크는 기존 보안정책을 증가시킬 뿐이다: 적절한 보안 정책과 정기적인 보안체크 없이 시스템이 완벽하게 안전하다고 믿는 것은 정말 바보 같은 짓이다.

이번 장에 포함되어 있는 예제는 단지 예제라는 것을 염두 해둔다. 이러한 설정을 중요한 시스템에 적용하지 않는다. 이런 정책을 시행하는 것은 고려해 볼만하지만 모든 것이 어떻게 동작하는지 정확하게 이해하지 못하는 사람은 전체 시스템을 되 돌리고 수많은 파일과 디렉터리를 다시 설정해야 되는 것을 깨닫게 될 수 있다.

### 15.1.1 여기서 설명하지 않는 것은 무엇인가?

이번 장에서는 MAC 프레임 워크에 관련된 폭넓은 보안에 대해 설명한다. MAC 정책을 개발 하는 것은 설명하지 않는다. 새로운 정책 작성은 이 글의 최초 목적을 완전히 벗어난다. 또한 MAC 프레임 워크에 포함된 몇몇 정책은 설명하지 않는다. 여기에는 `mac_test(4)`, `mac_stub(4)`와 `mac_none(4)` 모듈/정책이 포함되어 있다.

이들 모듈과 이들이 제공하는 다양한 메커니즘에 대한 더 많은 정보는 매뉴얼 페이지를 참고한다.

---

## 15.2 이번 장의 핵심 용어

이번 장을 소개하기 전에 핵심 용어에 대해 설명한다. 이로서 발생할 혼란을 방지하고 새로운 용어와 정보의 갑작스런 출현을 예방한다. 다음 용어는 이번 장 전반에 걸쳐 사용되고 있다:

- *구획(compartment)*: 구획은 사용자가 시스템의 특정 컴포넌트에 접근할 수 있도록 주어진 프로그램 세트와 파티션 되거나 나누어진 데이터를 의미한다. 구획은 또한 워크 그룹, 부서, 프로젝트나 주제처럼 그룹화를 의미한다.
- *무결성(Integrity)*: 중요한 개념으로서 무결성은 데이터에 둘 수 있는 신뢰 레벨이다. 데이터의 무결성이 증가하면 데이터의 신뢰성도 증가한다.
- *라벨*: 라벨은 파일, 디렉터리와 시스템의 다른 항목에 적용할 수 있는 보안 속성이다. 항목에 라벨을 붙여 기밀성을 표시할 수 있다; 파일에 라벨을 붙이면 그 파일에 대한 보안 레벨을 표시하여 파일, 유저, 리소스 등만 사용할 수 있다. 라벨의 의미와 판단은 정책에 따라 다르다: 어떤 정책은 라벨을 무결성을 나타내거나 비밀 주체로 처리하겠지만 다른 정책은 접근하기 위한 룰로 사용할 것이다.
- *레벨*: 보안속성의 설정을 증가시키거나 감소시킨다. 레벨을 증가시키면 보안도 향상된다.
- *멀티라벨*: *멀티라벨*은 tunefs(8) 유틸리티로 싱글 유저모드에 설정할 수 있는 파일 시스템 옵션이다; 부팅 중 fstab(5) 파일을 사용하여 설정하거나 새로운 파일시스템을 생성하는 동안 설정한다. 이 옵션은 파일시스템의 파일과 디렉터리에 멀티 MAC 라벨을 설정한다. 그리고 보안을 강화하기 위해 라벨을 사용하는 정책에만 적용한다.
- *주체(object)*: 주체 또는 시스템 주체는 *객체(subject)*를 목표로 하는 정보 흐름의 실체다. 여기에는 디렉터리, 파일, 필드, 화면, 키보드, 메모리, 스토리지, 프린터 또는 다른 데이터 스토리지/이동용 장치가 포함된다. 기본적으로 주체는 데이터 저장소 또는 시스템 리소스다; 주체를 효과적으로 사용한다는 의미는 데이터를 효과적으로 사용한다는 것이다.
- *정책*: 주체가 어떻게 취급되는지 정의한 룰 모음이다. *정책*은 보통 특정 아이템이 어떻게 제어되는지 적어놓은 것이다. 이번 장에서 정책이라는 용어는 *보안 정책*을

---

의미한다.

- *민감도(sensitivity)*: MLS 를 설명할 때 보통 사용된다. 민감도 레벨은 데이터가 얼마나 중요하고 비밀을 유지해야 되는지를 설명하는 용어다. 민감도 레벨이 증가되면 데이터의 중요도도 높아진다.
- *싱글 라벨*: 싱글 라벨은 데이터 흐름에 접근제어를 시행하기 위해 파일시스템 전체에 하나의 라벨만 사용하는 것이다. *멀티라벨(multilabel)* 옵션이 설정되지 않았을 때 파일시스템이 이런 식으로 설정되면 모든 파일은 같은 라벨로 설정된다.
- *객체(subject)*: 객체는 주체 사이의 정보 흐름으로 유발된 활동적인 존재다: 예를 들어 유저, 유저 프로세서, 시스템 프로세스 등이 포함된다.

## 15.3 MAC 설명

이 모든 용어를 생각하며 MAC 프레임 워크가 어떻게 시스템의 보안을 향상시키는지 생각해 보자. MAC 프레임 워크가 제공하는 다양한 정책은 네트워크, 파일시스템, 특정 포트나 소켓에 유저 접근제어 등에 사용된다. 아마도 정책을 가장 제대로 사용하는 방법은 다중 계층(multi-layer) 보안 환경을 위해 이들 정책을 혼합하여 사용하는 것이다. 유일한 단점은 다양한 파일시스템 라벨과 유저별로 네트워크 접근제어를 설정할 때 관리적인 업무가 증가한다.

이러한 단점은 프레임 워크의 영구적인 효과를 생각하면 별 것 아니다. 필요 없는 정책 지원을 감소시키면 선택의 폭을 넓힐 뿐만 아니라 시스템의 전체적인 성능을 향상시킬 수 있다. 그리고 이상적인 적용 방법은 필요한 모든 보안 요소와 프레임 워크가 제공하는 다양한 정책을 효과적으로 적용하는 것이다.

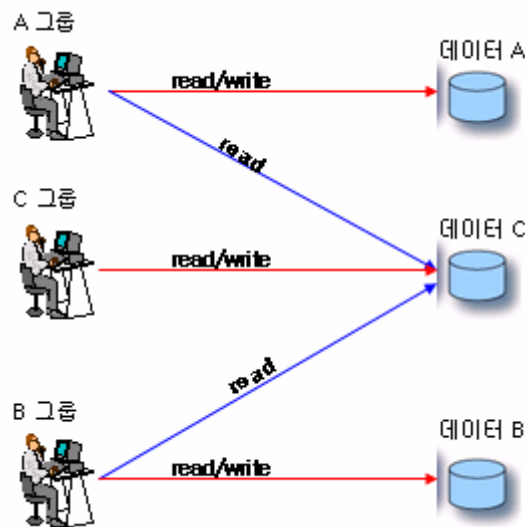
따라서 시스템에 사용되는 MAC 기능은 최소한 유저가 자신의 의지로 보안속성을 변경할 수 없어야 된다. 그리고 모든 유저 유틸리티 및 프로그램과 스크립트는 선택한 정책이 제공하는 접근 룰의 제한에서 동작해야 되며 모든 MAC 접근 룰은 시스템 관리자가 제어할 수 있어야 된다.

정확한 정책을 조심스럽게 선택하는 것은 시스템 관리자의 총 책임이다. 어떤 환경에는 네트워크를 통한 접근제어가 필요할 것이다; 이 경우 `mac_portacl(4)`, `mac_ifoff(4)`와

mac\_biba(4)를 생각해 볼만하다. 그렇지 않으면 파일시스템의 보안 아이템이 필요할 것이다. mac\_bsdextended(4)와 mac\_mls(4)와 같은 정책이 이러한 목적에 사용되고 있다.

정책 결정은 네트워크 설정을 따른다. 아마도 특정 유저 한 명만 ssh(1)가 제공하는 기능으로 네트워크나 인터넷을 사용하게 할 수 있을 것이다. mac\_portacl(4)는 이런 상황에 선택하는 정책이다. 파일시스템의 경우는 어떻게 해야 되는가? 특정 디렉토리를 다른 그룹이나 유저가 사용할 수 있도록 해야 되는가? 또는 특정 주체를 등급으로 나누어서 유저나 유틸리티의 특정 파일 사용을 제한해야 되는가?

파일시스템의 경우 주체를 사용하는 것은 다른 사람이 아닌 특정 유저에게만 허용할 것이다. 예를 들어 대형 개발 팀은 각각의 작은 그룹으로 나누어 질 것이다. 프로젝트 A의 개발자들은 프로젝트 B의 개발자들이 작성한 주체 사용은 금지될 것이다. 그러나 프로젝트 C 개발자들이 생성한 주체는 사용할 수 있어야 될 것이다; 상당히 난처한 상황이다. 이러한 경우 MAC 프레임 워크가 제공하는 다른 정책을 사용한다; 유저들을 이들 그룹으로 나누고 정보 유출에 대한 걱정 없이 적절한 영역을 사용할 수 있는 권한을 준다.



따라서 각 정책은 정보흐름과 관련된 유일한 방법을 가지게 된다. 정책 선택은 심사 숙고한 보안 정책에 전적으로 의존된다. 대부분 환경에 맞게 전체 정책을 수정하여 네트워크에 이행해야 된다. MAC 프레임 워크가 제공하는 여러 가지 정책을 이해하면 상황에 따른 최고의 정책을 선택하는데 도움이 된다.

기본 FreeBSD 커널은 MAC 프레임 워크 옵션을 포함하지 않는다; 따라서 다음 커널 옵션을 추가해야 된다:

```
options    MAC
```

---

그리고 커널을 다시 빌드하고 설치한다.

**주의:** 다양한 MAC 모듈 매뉴얼 페이지에서는 커널에 빌드하도록 설명하지만 시스템에서 네트워크와 다른 요소들을 사용하지 못하게 될 수 있다. MAC 을 시행하는 것은 방화벽 시행과 비슷하지만 시스템을 완전히 잠그는 것은 피해야 된다. 이전 설정으로 되돌리는 능력을 고려해야 되겠지만 원격에서 MAC 을 시행하는 것은 굉장한 주의가 요구된다.

## 15.4 MAC 라벨 이해

MAC 라벨은 사실 시스템의 전체 주체와 객체에 적용할 수 있는 보안 속성이다.

라벨을 설정할 때 유저는 이것이 무엇이고 무엇과 연관되는지 확실히 이해해야 된다. 주체에 적용할 수 있는 속성은 로드된 정책에 따라 달라지고 따라서 속성의 해석도 달라진다. 이해 부족으로 설정을 잘못했다면 시스템이 예상하지 못한 동작을 하게 된다. 그리고 라벨을 너무 높거나 낮게 설정하면 디렉터리와 파일시스템의 전체 보안 구조에 결함이 생긴다

주체의 보안 라벨은 보안 접근 제어를 결정하는 일부분으로 정책에 의해 사용된다. 어떤 정책에서 라벨은 결정에 필요한 모든 정보를 가지고 있다; 다른 모델에서 라벨은 거대한 룰 세트의 일부분으로 처리된다.

예를 들어 파일에 *biba/low* 라벨을 설정하는 것은 "low" 값의 Biba 정책으로 라벨이 관리됨을 의미한다. 그리고 주체와 객체에 정책 이름과 자격을 설정하기 때문에 라벨 포맷은 간단하다.

FreeBSD 는 라벨 기능을 지원하는 정책으로 미리 정의된 라벨(low, high 와 equal 라벨) 3 가지가 제공된다. 이들은 각 정책과 다른 방식으로 접근 제어를 시행하지만 low 라벨은 가장 낮은 설정, equal 라벨은 객체와 주체를 비활성 하거나 영향을 받지 않도록 설정하고 high 라벨은 가능한 가장 높은 설정을 시행한다.

싱글 라벨 파일시스템 환경에서는 오직 하나의 라벨만 주체와 객체에 사용될 것이다. 여기서는 접근 퍼미션 한 세트를 전체 시스템과 세트 전체가 필요할 수 있는 많은 환경에 시행한다. 그러나 멀티 라벨이 시스템의 주체나 객체에 설정될 수 있는 몇 가지 경우가 있다.

---

이러한 경우 *멀티 라벨* 옵션이 `tunefs(8)`에 적용될 것이다.

Biba 와 MLS 의 경우 제어 계층의 정확한 레벨을 보여주기 위해 숫자 라벨이 설정될 것이다. 이 숫자 레벨은 파티션하거나 권한을 가진 다른 그룹으로 정보를 나누어서, 그 그룹이 나 더 상위 그룹만 접근을 허용하는 등급으로 분류한다.

대부분의 경우 관리자는 파일시스템 전반에 싱글 라벨만 설정한다; 사실 이 제한은 조금 있다 설명할 *멀티라벨* 옵션을 필요로 한다.

*이 방법은 DAC 와 비슷하다. MAC 은 관리자에게 엄격한 제어를 제공한다.* root 는 제어와 정책을 설정할 수 있는 사람이기 때문에 유저를 적절한 카테고리/접근 레벨에 할당할 수 있다. 그렇지만 다양한 정책으로 root 유저도 제한할 수 있다. 주체를 통한 기본 제어는 그룹을 제외하지만 root 는 언제라도 설정을 변경할 수 있다. 이것은 Biba 와 MLS 같은 정책에서 다루는 계층적/인증 모델이다.

## 15.4.1 라벨 설정

사실상 라벨 정책을 설정하는 모든 사항은 4 개의 명령어 세트를 사용하여 시행된다. 이들 명령은 주체나 객체를 설정하거나 설정을 조정하고 확인하기 위한 간단한 인터페이스를 제공한다.

모든 사항은 `setfmac(8)`과 `setpmac(8)` 유틸리티를 각각 사용하여 설정할 수 있다. `setfmac` 명령은 시스템 주체에 MAC 라벨을 설정하는데 사용하지만 시스템 객체에 라벨을 설정할 때는 `setpmac` 명령을 사용한다. 확인해 보자:

```
# setfmac biba/high test
```

위 명령에서 에러가 없으면 프롬프트가 나타나지만 에러가 발생하면 반응을 보인다; `chmod(1)`과 `chwon(8)` 명령과 비슷하다. 제한된 주체에 라벨을 설정하거나 수정할 때 "Permission denied"가 보통 발생한다. 시스템 관리자는 이 문제를 해결하기 위해 다음 명령을 사용할 수 있다:

```
# setfmac biba/high test
``Permission denied''
# setpmac biba/low setfmac biba/high test
# getfmac test
test: biba/high
```

위에서 확인 하였듯이 setpmac 는 관련된 프로세스에 다른 라벨을 할당하여 정책 설정을 덮어 쓰는데 사용할 수 있다. getpmac 유틸리티는 **sendmail** 처럼 현재 실행 중인 프로세스에 보통 사용된다: 명령어 대신 프로세스 ID 를 사용하지만 비슷한 논리다. 사용자가 주체나 객체를 소유하고 있다면 그 유저만 정책 라벨을 무시할 수 있다. 유저가 자신의 접근 레벨이 아닌 파일을 조정하려고 하면 mac\_set\_link 기능으로 “Operation not permitted” 에러가 표시된다.

#### 15.4.1.1 유저와 라벨 설정

유저가 직접 라벨을 가지고 있어야 되기 때문에 그들의 파일과 프로세스도 시스템에 정의된 보안 정책과 적절히 상호작용해야 된다. 이 설정은 로그인 클래스를 사용하여 login.conf 파일을 통해 설정된다. 라벨을 사용하는 모든 정책은 유저 클래스 설정을 시행한다.

모든 정책을 가진 예제 엔트리는 아래에 나열되어 있다:

```
default:W
:copyright=/etc/COPYRIGHT:W
:welcome=/etc/motd:W
:setenv=MAIL=/var/mail/$,BLOCKSIZE=K:W
:path=~:/bin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/local/sbin:/usr/local/bin:W
:manpath=/usr/share/man /usr/local/man:W
:nologin=/usr/sbin/nologin:W
:cputime=1h30m:W
:datasize=8M:W
:vmemoryuse=100M:W
:stacksize=2M:W
:memorylocked=4M:W
:memoryuse=8M:W
:filesize=8M:W
```

```
:coredumpsize=8M:W
:openfiles=24:W
:maxproc=32:W
:priority=0:W
:requirehome:W
:passwordtime=91d:W
:umask=022:W
:ignoretime@:W
:label=partition/13,mls/5,biba/10(5-15),lmac10[2]:
```

*label* 옵션은 MAC 이 시행하는 유저 클래스의 기본 라벨을 설정하는데 사용한다. 유저가 이 값을 절대 수정할 수 없기 때문에 유저의 경우는 필수가 될 수 있다. 그러나 실제 설정에서 관리자는 모든 정책을 활성화하기를 원치 않을 것이다. 이들 설정을 시행하기 전에 이번 장의 나머지 내용도 검토하기 바란다.

**Note:** 유저는 처음으로 로그인해서 그들의 라벨을 변경할 것이다; 그러나 이 변경은 객체를 제한하는 정책이다. 위의 예제에서 프로세스의 무결성은 최소 5 에서 최대 15 까지고 기본적인 라벨은 10 이라고 Biba 정책에 지시한다. 따라서 그 프로세스는 라벨을 변경할 때까지 10 에서 실행되고 유저가 `setpmac` 명령을 사용한다면 로그인할 때 Biba 에 의해 강제로 범위가 정해진다.

`login.conf` 를 변경한 후 `cap_mkdb` 를 사용하여 로그인 클래스 특성 데이터베이스를 다시 빌드해야 되고 이것은 앞으로 보게 될 예제와 설명에 반영된다.

특히 많은 유저가 있는 여러 사이트에서 여러 개의 다른 유저 클래스가 필요할 때 유용할 것이다. 이 방법은 관리하기가 어렵기 때문에 상당히 심사숙고 해야 된다.

### 15.4.1.2 네트워크 인터페이스와 라벨 설정

네트워크를 통한 데이터 흐름을 제어하기 위해 네트워크 인터페이스에도 라벨을 설정할 수 있을 것이다. 주체에 대한 정책 기능처럼 모든 경우에 작용한다. 예를 들어 *biba* 에서 높게 설정된 유저는 낮은 라벨로 설정된 네트워크 인터페이스에 접근할 수 없다.

네트워크 인터페이스에 MAC 라벨이 설정되면 *maclabel*이 `ifconfig` 에 적용될 것이다. 예를 들면 다음과 같다:



---

```
# ifconfig bge0 maclabel biba/equal
```

위 명령은 bge(4) 인터페이스에 *biba/equal* MAC 라벨을 설정한다. *biba/high(low-high)*와 비슷한 설정을 사용할 때 전체 라벨을 인용부호로 감싸야 한다. 그렇지 않으면 에러가 발생한다.

라벨을 지원하는 각 정책은 네트워크 인터페이스에서 MAC 라벨을 비활성 할 때 사용할 수 있는 tunable 를 가지고 있다. 라벨을 *equal*로 설정하면 비슷한 효과를 가져온다. sysctl 의 결과나 정책 매뉴얼 페이지 또는 이번 장 다음에 찾을 수 있는 이들 tunables 에 대한 정보를 확인한다.

## 15.4.2 싱글라벨 또는 멀티라벨

기본적으로 시스템은 싱글라벨 옵션을 사용한다. 그러나 관리자에게 이것은 어떤 의미인가? 시스템 보안 모델의 유연성을 위해 허용과 거부권을 제공하기 때문에 관리자 권한에서는 약간 다르다.

싱글라벨은 오직 하나의 라벨만 허용한다. 예를 들어 각 객체나 주체에 *biba/high*가 사용된다. 이 방법은 관리하기 편하겠지만 라벨을 지원하는 정책의 유연성을 떨어뜨린다. 많은 관리자들이 보안 정책에 멀티라벨 옵션을 사용하기를 원할 것이다.

멀티라벨 옵션은 전체 파티션에 오직 하나의 라벨만 허용하는 표준 싱글라벨 대신 각 객체나 주체가 각자의 독립적인 MAC 라벨을 가질 수 있도록 허용한다. 멀티라벨과 싱글라벨 옵션은 Biba, Lomac, MLS 와 SEBSD 정책을 포함하여 라벨 기능을 시행하는 정책에만 필요하다.

대부분 멀티라벨을 다 설정할 필요는 없을 것이다. 다음과 같은 상황과 보안 모델을 생각해 보자:

- MAC 프레임워크와 다양한 정책을 섞어 사용하는 FreeBSD 웹 서버
- 이 머신은 시스템의 모든 사항에 *biba/high* 라벨 하나만 필요하다. 이곳의 파일 시스템은 싱글라벨이 항상 영향을 끼치고 있기 때문에 멀티라벨 옵션은 필요 없다.

- 
- 그러나 이 머신이 웹 서버가 되기 때문에 게시(write up) 기능을 방지하기 위해 *biba/low*에서 웹 서버를 운용해야 된다. Biba 정책과 이 정책의 기능은 나중에 설명하기 때문에 이해하기 어렵다면 일단 계속 읽은 후 다시 되돌아온다. 실행 상태가 아니면 서버는 대부분 *biba/low*에서 분리된 파티션 세트를 사용할 수 있다. 이 예제 설정과 유저 설정은 많이 부족하다(예를 들어 데이터 제한 같은). 그러나 이것은 단지 앞에서 설명한 것을 증명하는 예제다.

라벨이 아닌 정책이 사용된다면 멀티라벨 옵션은 필요 없다. 여기에는 *seeotheruids*, *portacl* 과 *partition* 정책이 포함된다.

파티션과 멀티라벨 기능 기반의 보안모델을 수립할 때 멀티라벨을 사용하면 파일시스템의 모든 것들이 라벨을 갖기 때문에 관리가 어려워질 수 있다. 이것은 디렉터리, 파일과 장치 노드까지 포함한다.

다음 명령은 멀티라벨을 갖기 위해 파일시스템에 멀티라벨을 설정한다. 이것은 싱글 유저 모드에서만 설정할 수 있을 것이다.

```
# tunefs -l enable /
```

스왑 파일시스템에는 필요 없다.

**Note:** 어떤 유저는 root 파티션에 멀티라벨 플래그를 설정할 때 문제가 있었다. 이러한 경우 이번 장의 문제해결 섹션(15.16)을 참고한다.

### 15.4.3 Tunables 로 MAC 제어하기

모듈을 로드하지 않고도 sysctl 인터페이스로 설정할 수 있는 MAC 이 있을 것이다:

- *security.mac.enforce\_fs*는 기본적으로 활성화되고 파일시스템에 MAC 파일시스템 정책이 시행된다.
- *security.mac.enforce\_kld*는 기본적으로 활성화되고 동적 커널 링커에 MAC 커널 링크 정책이 시행된다([kld\(4\)](#)를 본다).
- *security.mac.enforce\_network*는 기본적으로 활성화되고 MAC 네트워크 정책이 시행된다.

- 
- *security.mac.enforce\_pipe* 는 기본적으로 활성화되고 파이프(pipe)에 MAC 정책이 시행된다.
  - *security.mac.enforce\_process* 는 기본적으로 활성화되고 프로세스간의 통신에 사용되는 프로세스에 MAC 정책이 시행된다.
  - *security.mac.enforce\_socket* 는 기본적으로 활성화되고 소켓에 MAC 정책이 시행된다([socket\(2\)](#) 매뉴얼 페이지를 본다).
  - *security.mac.enforce\_system* 는 기본적으로 활성화되고 accounting 과 재 부팅 같은 시스템 동작에 MAC 정책이 시행된다
  - *security.mac.enforce\_vm* 는 기본적으로 활성화되고 가상 메모리 시스템에 MAC 정책이 시행된다.

**Note:** 모든 정책이나 MAC 옵션은 tunables 를 지원한다. 이들은 보통 *security.mac.<polycyname>* 트리에 놓인다. MAC 에서 모든 tunables 를 보려면 다음 명령을 사용한다:

```
# sysctl -da | grep mac
```

기본 MAC 정책이 일반적으로 모두 시행되기 때문에 설명해야 됐다. 모듈을 커널에 빌드했다면 시스템은 상당히 제한되기 때문에 대부분 로컬 네트워크나 인터넷 연결 등과 같은 통신을 할 수 없다. 이것이 모듈을 커널에 빌드하지 말라는 이유다. 따라서 sysctl 로 비활성할 수 있는 기능을 제한하는 것이 아니고 다시 빌드하여 새로운 시스템을 설치할 필요 없이 관리자가 시스템의 정책을 일시적으로 변경할 수 있다.

## 15.5 모듈 설정

MAC 프레임 워크에 포함된 모든 모듈은 위에서 말했듯이 커널에 포함되어 있거나 런타임 커널 모듈에 의해 로드된다. 권장하는 방법은 /boot/loader.conf 파일에 모듈 이름을 추가하여 부팅할 때 로드하는 것이다.

다음 섹션은 다양한 MAC 모듈과 특징을 설명한다. 이들을 특정 환경에서 사용하는 것도 이번 장에서 고려하고 있다. 어떤 모듈은 “이것은 허용되지만 저것은 안 된다”처럼 라벨을 시행하여 접근 제어하는 라벨을 지원한다. 라벨 설정파일에서는 파일이 어떻게 접근되며 네트워크로 통신이 가능하지 등을 제어할 것이다. 이전 섹션에서는 파일이나 파티션 접근제어 별로 활성화하기 위해 멀티레벨 플래그를 파일시스템에 어떻게 설정하는지 보여줬다.

---

싱글라벨 설정은 오직 하나의 라벨만 시스템에 시행하기 때문에 tunefs 옵션을 멀티라벨이라고 한다.

## 15.5.1 MAC seeotheruids 모듈

모듈 이름: `mac_seeotheruids.ko`

커널 설정 라인: `options MAC_SEEOTHERUIDS`

부트 옵션: `mac_seeotheruids_load="YES"`

`mac_seeotheruids(4)` 모듈은 `security.bsd.see_other_uids` 과 `security.bsd.see_other_gids` sysctl tunables 을 모방하여 확장한다. 이 옵션을 설정하기 전에 라벨을 설정할 필요 없이 다른 모듈과 동작할 수 있다.

이 모듈을 로드한 후 다음 sysctl tunables 이 기능을 제어하기 위해 사용될 것이다:

- `security.mac.seeotheruids.enabled` 는 모듈의 기능을 활성화하고 기본 설정을 사용한다. 이들 기본 설정은 다른 유저의 프로세스와 소켓을 볼 수 없게 한다.
- `security.mac.seeotheruids.specificgid_enabled` 는 특정 그룹을 이 정책에서 제외시킨다. 이 정책에서 특정 그룹을 제외시키려면 `security.mac.seeotheruids.specificgid=XXX` sysctl tunable 을 사용한다. 위 예제에서 `XXX` 는 제외시키려는 그룹 ID 번호로 대체한다.
- `security.mac.seeotheruids.primarygroup_enabled` 는 이 정책에서 특정 주 그룹을 제외시킬 때 사용한다. 이 tunable 를 사용할 때 `security.mac.seeotheruids.specificgid_enabled` 는 설정되지 않을 것이다.

root 유저를 이 정책에서 제외할 수 없음을 기억한다. 이것이 MAC 버전과 기본적으로 `security.bsd.seeotheruids` 에 포함된 표준 tunable 버전과의 가장 큰 차이점 중 하나다.

---

## 15.6 MAC bsdextended 모듈

모듈 이름: `mac_bsdextended.ko`

커널 설정 라인: `options MAC_BSDEXTENDED`

부트 옵션: `mac_bsdextended_load="YES"`

`mac_bsdextended(4)` 모듈은 파일시스템 방화벽을 시행한다. 이 모듈 정책은 표준 파일시스템 퍼미션 모델을 확장하고 파일시스템 계층에서 관리자가 파일과 유틸리티 그리고 디렉터리를 보호하기 위해 방화벽 같은 룰 세트 생성을 허용한다.

정책은 `ipfw(8)`과 비슷한 구문을 가지고 있는 `ugidfw(8)` 유틸리티를 사용하여 생성할 수 있을 것이다. 더 많은 룰은 `libugidfw(3)` 라이브러리 기능을 사용하여 작성할 수 있다.

이 모듈을 적용할 때 아주 조심해야 된다. 잘못된 사용은 파일시스템의 특정 부분을 사용할 수 없게 된다.

### 15.6.1 예제

`mac_bsdextended(4)` 모듈이 로드되면 다음 명령이 현재 룰 설정을 나열하기 위해 사용될 것이다:

```
# ugidfw list
0 slots, 0 rules
```

예상대로 룰은 정의되어있지 않다. 이 의미는 모든 것에 접근할 수 있다는 것이다. `root`는 영향을 받지 않고 모든 유저로부터 모든 접근을 막기 위해 룰을 생성하려면 단순히 다음 명령을 실행한다:

```
# ugidfw add subject not uid root new object not uid root mode n
```

---

**Note:** FreeBSD 5.3 이전 릴리즈에는 add 매개변수가 없다. 이 경우 set 변수를 대신 사용한다. 아래 예제를 참고한다.

ls 처럼 매우 단순한 명령이라도 모든 유저가 실행할 수 없기 때문에 좋아 보이지는 않는다. 좀더 효과적인 룰은 다음과 같을 것이다:

```
# ugidfw set 2 subject uid user object uid trhodes mode n
# ugidfw set 3 subject uid user object gid trhodes mode n
```

이 명령은 유저 이름 user 로 trhodes 의 홈 디렉터리에서 디렉터리 리스트를 포함하여 모든 접근을 막는다.

user 대신 *not uid trhodes* 를 적용할 수 있다. 이 명령도 위와 동일한 제한을 모든 유저에게 시행한다.

**Note:** root 유저는 이 설정으로 영향을 받지 않는다.

여기서는 파일시스템을 강화하기 위해 mac\_bsdextended(4) 모듈을 어떻게 사용해야 되는지 일반적인 아이디어를 제공한다. 더 많은 정보는 mac\_bsdextended(4)와 ugidfw(8) 매뉴얼 페이지를 본다.

## 15.7 MAC ifoff 모듈

**모듈 이름:** mac\_ifoff.ko

**커널 설정 라인:** *options MAC\_IFOFF*

**부트 옵션:** *mac\_ifoff\_load="YES"*

mac\_ifoff(4) 모듈은 시스템이 최초로 부팅하는 동안 네트워크 인터페이스를 비활성화하기 위해 단독으로 존재한다. 시스템에 라벨을 설정할 필요 없고 다른 MAC 모듈에 의존하지도 않는다.

---

대부분은 아래에 나열된 `sysctl tunables` 를 통해 제어된다.

- `security.mac.ifflo.enabled`은 루프백(lo(4)) 인터페이스의 모든 트래픽을 활성화/비활성 한다.
- `security.mac.ifflo.bpfrecv.enabled`은 버클리 패킷 필터 인터페이스(bpf(4))의 모든 트래픽을 활성화/비활성 한다.
- `security.mac.ifflo.other.enabled`은 다른 모든 인터페이스의 트래픽을 활성화/비활성 한다.

`mac_ifflo(4)`를 가장 일반적으로 사용하는 방법은 부팅하는 동안 네트워크 트래픽이 허용되지 않는 환경에서 네트워크를 모니터링 하는데 사용한다. 다른 권장되는 사용법은 보호되고 있는 디렉터리에서 새로운 파일이나 수정된 파일이 발견되면 자동으로 네트워크 트래픽을 차단하기 위해 `security/aide`를 사용하는 스크립트 작성에 사용하는 것이다.

## 15.8 MAC portacl 모듈

모듈 이름: `mac_portacl.ko`

커널 설정 라인: `MAC_PORTACL`

부트 옵션: `mac_portacl_load="YES"`

`mac_portacl(4)` 모듈은 다양한 `sysctl` 변수를 사용하여 로컬 TCP 와 UDP 포트 바인딩을 제약할 때 사용할 수 있다. 사실 `mac_portacl(4)`는 root 가 아닌 유저가 미리 정의된 특정 포트에(1024 보다 낮은 포트) 바인드 할 수 있게 한다.

로드 되면 이 모듈은 모든 소켓에 MAC 정책을 활성화한다. 다음과 같은 `tunables` 을 활성화할 수 있다:

- `security.mac.portacl.enabled` 정책을 완벽히 활성화/비활성 한다(버그 때문에

---

*security.mac.portacl.enabled* sysctl 변수는 FreeBSD 5.2.1 이나 이전 릴리즈에서 작동하지 않는다).

- *security.mac.portacl.port\_high* 는 *mac\_portacl(4)*가 보호하려는 가장 높은 번호를 설정한다.
- *security.mac.portacl.suser\_exempt* 는 0 이 아닌 값이 설정됐을 때 root 유저를 이 정책에서 제외시킨다.
- *security.mac.portacl.rules* 는 실제 *mac\_portacl* 정책을 지정한다; 아래를 본다.

*security.mac.portacl.rules* sysctl 에서 지정하였듯이 실제 *mac\_portacl* 정책은 *rule[,rule,...]* 형식의 텍스트 문자열로 필요한 룰을 정의한다. 각 룰은 *idtype:id:protocol:port* 형식으로 되어있다. *idtype* 매개변수는 각각 유저 id 나 그룹 id 가 되는 *uid*나 *gid*가 되어 *id* 매개변수를 해석하는데 사용된다. *protocol* 매개변수는 *tcp*나 *udp*로 매개변수를 지정하여 룰을 TCP 에 적용할지 UDP 에 적용할지 결정하는데 사용한다. 마지막 *port* 매개변수는 유저나 그룹이 바인드할 수 있도록 지정된 포트번호다.

**Note:** 룰 셋이 커널에 의해 직접 해석되기 때문에 오직 숫자 값만 유저 ID, 그룹 ID, 그리고 포트 매개변수에 사용할 수 있다. 다시 말해 유저, 그룹 그리고 포트 서비스 이름은 사용할 수 없다.

유닉스와 같은 운영체제에서 1024 이하의 포트는 특정 권한을 가진 프로세스에 바인드 하는데 사용한다. 즉 root 가 실행하는 프로세스를 의미한다. *mac\_portacl(4)*는 권한이 없는 프로세스를 1024 이하의 포트에 바인드하기 위해 표준 유닉스 제한을 비활성 한다. 이것은 *sysctl(8)* 변수 *net.inet.ip.portrange.reservedlow* 와 *net.inet.ip.portrange.reservedhigh* 을 0 으로 설정하면 된다.

더 많은 정보는 아래 예제나 *mac\_portacl(4)* 매뉴얼 페이지를 참고 한다.

## 15.8.1 예제

다음 예제는 위의 설명을 이해하기 쉽게 해준다:



---

```
# sysctl security.mac.portacl.port_high=1023
# sysctl net.inet.ip.portrange.reservedlow=0 net.inet.ip.portrange.reservedhigh=0
```

첫째 표준 권한을 가진 포트에 `mac_portacl(4)`를 설정하고 일반적인 유닉스 바인드 제한을 비활성 한다.

```
# sysctl security.mac.portacl.suser_exempt=1
```

`root` 유저는 이 정책의 영향을 받지 않기 때문에 `security.mac.portacl.suser_exempt`을 0 이 아닌 값으로 설정한다. 이제 `mac_portacl(4)` 모듈은 기본적으로 유닉스처럼 동작하도록 시스템 동작을 설정한다.

```
# sysctl security.mac.portacl.rules=uid:80:tcp:80
```

UID 80 의(보통 `www` 유저) 유저가 포트 80 번과 바인드할 수 있게 한다. 따라서 `www` 유저가 `root` 권한 없이 웹 서버를 실행할 수 있다.

```
# sysctl security.mac.portacl.rules=uid:1001:tcp:110,uid:1001:tcp:995
```

1001 의 UID 유저가 TCP 포트 110("pop3")번과 995("pop3s")번에 바인드 하게한다. 이 명령은 이 유저가 포트 110 과 995 번에서 연결을 허용하는 서버를 시작하도록 한다.

## 15.9 라벨 기능과 MCA 정책

다음 섹션 몇 가지는 라벨을 사용하는 MAC 정책에 대해 설명한다.

이 섹션부터 `mac_biba(4)`, `mac_lomac(4)`, `mac_partition(4)`, 그리고 `mac_mls(4)`의 기능에 대해 설명한다.

**Note:** 이것은 예제 설정이기 때문에 중요한 제품에는 사용하지 않는다. 이 문서의 목적은 시행과 테스트를 위한 구문과 예제를 보여 주는 것이다.

---

이들 정책이 정확히 작동하기 위해 몇 가지 준비가 필요하다.

## 15.9.1 라벨 정책 준비

다음 내용이 login.conf 파일에 필요하다:

- *insecure* 클래스나 비슷한 종류의 다른 클래스를 추가해야 된다. Insecure 로그인 클래스는 필요 없지만 여기 예제에서만 사용한다; 다른 설정은 다른 클래스 이름이 필요할 것이다.
- *Insecure* 클래스에는 다음 설정과 정의가 필요하다. 이들 중 몇 가지는 수정할 수 있지만 기본 라벨을 정의하는 라인은 필요하다.

```
insecure:W
:copyright=/etc/COPYRIGHT:W
:welcome=/etc/motd:W
:setenv=MAIL=/var/mail/$,BLOCKSIZE=K:W
:path=~:/bin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/local/sbin:/usr/local/bin:W
:manpath=/usr/share/man /usr/local/man:W
:nologin=/usr/sbin/nologin:W
:cputime=1h30m:W
:datasize=8M:W
:vmemoryuse=100M:W
:stacksize=2M:W
:memorylocked=4M:W
:memoryuse=8M:W
:filesize=8M:W
:coredumpsize=8M:W
:openfiles=24:W
:maxproc=32:W
:priority=0:W
:requirehome:W
:passwordtime=91d:W
:umask=022:W
:ignoretime@:W
:label=partition/13,mls/5,biba/low:
```

---

유저가 새로운 클래스로 변경되기 전에 `cap_mkdb(1)` 명령을 `login.conf(5)`로 실행해야 된다.

`root`도 `login` 클래스에 두어야 된다. 그렇지 않으면 `root`가 실행하는 거의 모든 명령은 `setpmac`를 사용해야 된다.

**주의:** `login.ocnf` 데이터베이스를 다시 빌드하면 데몬 클래스와 나중에 어떤 에러가 발생할 수 있다. 간단히 데몬 계정을 주석처리하고 데이터베이스를 다시 빌드하는 것이 이 문제를 줄일 수 있다.

- MAC 라벨을 시행하는 모든 파티션은 멀티라벨을 지원해야 된다. 여기 있는 많은 예제는 테스트 목적의 다른 라벨을 가지고 있기 때문에 이렇게 해야 된다. 예방책으로 `mount` 명령의 결과를 확인한다.
- 새로운 유저 클래스에 시행된 높은 보안 메커니즘을 가진 유저를 변경한다. `pw(8)`이나 `vipw(8)`로 이 쉽게 변경할 수 있다.

## 15.10 MAC 파티션 모듈

**모듈 이름:** `mac_partition.ko`

**커널 설정 라인:** `options MAC_PARTITION`

**부트 옵션:** `mac_partition_load="YES"`

`mac_partition(4)` 정책은 프로세스를 MAC 라벨 기반의 특정 “파티션”에서 벗어나지 못하게 한다. 비교 하기는 어렵겠지만 이것을 특별한 종류의 `jail(8)`로 생각한다.

이것은 `loader.conf(5)` 파일에 추가해야 되는 하나의 모듈이기 때문에 부팅하는 동안 정책을 로드하고 활성화한다.

이 정책을 위한 대부분의 설정은 아래에서 설명하는 `setpmac(8)` 유틸리티를 사용하여 설정된다. 다음 `sysctl tunable`을 이 정책에 사용할 수 있다:

- 
- `security.mac.partition.enabled`는 MAC 프로세스 파티션의 시행을 활성화한다.

이 정책이 활성화되면 유저는 자신의 프로세스만 볼 수 있게 되고 특정 유틸리티로 작업 하지 못하게 된다. 예를 들면 `insecure` 클래스 위의 유저는 `top` 명령과 다른 프로세스를 생성하는 많은 명령을 사용할 수 없게 된다.

유틸리티를 파티션 라벨에 설정하기 위해 `setpmac` 유틸리티를 사용한다:

```
# setpmac partition/13 top
```

이 명령은 `insecure` 클래스 유저에게 라벨 셋으로 `top` 명령을 추가한다. 유저가 생성하는 `insecure` 클래스의 모든 프로세스는 `partition/13` 라벨에서 실행된다.

## 15.10.1 예제

다음 명령은 파티션 라벨과 프로세스 리스트를 보여준다:

```
# ps Zax
```

다음 명령은 다른 유저의 프로세스 파티션 라벨과 그 유저가 현재 실행하는 프로세스를 보여 준다:

```
# ps -ZU trhodes
```

**Note:** 유저는 `mac_seeotheruids(4)` 정책이 로드되지 않는다면 `root` 라벨의 프로세스를 볼 수 있다.

아주 교묘하게 실행하는 방법은 `/etc/rc.conf` 에서 모든 서비스를 비활성하고 적절한 라벨 세트로 시작되는 스크립트를 실행한다.

**Note:** 다음 정책은 제공되는 3 개의 기본라벨 대신 정수 설정을 지원한다. 제한을 포함하고 있는 이들 옵션은 모듈 매뉴얼 페이지에 더 자세히 설명되어 있다.

---

## 15.11 MAC 다중 계층 보안 모듈

모듈 이름: `mac_mls.ko`

커널 설정 라인: `options MAC_MLS`

부트 옵션: `mac_mls_load="YES"`

`mac_mls(4)` 정책은 정보 흐름 정책을 엄격하게 시행하여 시스템의 객체와 주체에 접근하는 것을 제어한다.

MLS(다중 계층 보안) 환경에서 “clearance” 레벨을 구현에 따라 각 객체나 주체의 라벨에 설정한다. 이들 clearance 나 sensibility 레벨은 6000 이상의 번호에 영향을 미친다; 이들 레벨을 각 객체나 주체에 완벽하게 설정하는 것은 시스템 관리자에게 엄청난 일이 된다. 다행히 3 개의 “instant” 라벨은 이미 이 정책에 포함되어 있다.

이들 라벨은 `mls/low` 와 `mls/equal` 그리고 `mls/high` 다. 이들 라벨은 매뉴얼 페이지에 자세히 설명되어 있기 때문에 여기서는 간단히 설명한다:

- `mls/low` 라벨은 다른 모든 주체에 의해 제어되는 낮은 설정을 포함한다. `mls/low` 로 라벨된 모든 것들은 낮은 clearance 레벨을 가지고 있기 때문에 높은 레벨의 정보에 접근할 수 없다. 게다가 이 라벨은, 이들 라벨에 쓰거나 정보를 추가하는 높은 clearance 레벨 주체를 허용하지 않는다.
- `mls/equal` 라벨은 정책에서 제외해야 되는 주체에 적용해야 된다.
- `mls/high` 라벨은 가능한 가장 높은 레벨의 clearance 다. 이 라벨에 할당된 주체는 시스템의 다른 모든 주체보다 우선권을 가진다; 그러나 낮은 클래스 주체에 정보를 누출시키지 못하게 한다.

MLS 는 다음과 같은 것을 제공한다:

- 계층이 없는 카테고리 설정으로 계층적인 보안 레벨을 제공한다.

- 
- 읽을 수 없고 작성할 수 없는 고정된 룰을 제공한다(객체는 자신의 레벨이나 하위 주체에 접근하기 위해 읽을 수 있지만 상위 레벨은 읽을 수 없다).
  - 비밀을 제공한다(부적절한 데이터 공개를 방지한다).
  - 시스템 디자인의 기본적인 목적은 여러 개의 민감한 레벨에서(비밀과 신뢰 사이에서 정보 누출 없이) 데이터를 동시에 제어하기 위함이다.

다음 `sysctl tunables` 은 특별한 서비스와 인터페이스를 설정하는데 사용할 수 있다:

- `security.mac.mls.enabled` 는 MLS 정책을 활성화/비활성 한다.
- `security.mac.mls.ptys_equal` 는 `mls/equal` 을 생성하는 동안 모든 `pty(4)` 장치에 라벨을 할당한다.
- `security.mac.mls.revocation_enabled` 은 라벨이 낮은 계층의 라벨로 변경된 후 주체의 접근을 폐지한다.
- `security.mac.mls.max_compartments` 는 주체에 최대치의 구획 레벨을 설정할 때 사용된다. 그리고 기본적으로 최대 구획 번호는 시스템에서 허용된다.

MLS 라벨을 조정하기 위해 `setfmac(8)` 명령이 제공되었다. 주체에 라벨을 할당하기 위해 다음 명령을 사용한다:

```
# setfmac mls/5 test
```

파일 `test` 에 MLS 라벨을 할당하려면 다음 명령을 사용한다:

```
# getfmac test
```

여기서는 MLS 의 정책 기능을 요약하였다. 또 다른 방법은 MLS 정책 정보를 지정하는 마스터 정책파일을 `/etc` 파일에 생성하고 `setfmac` 명령에 이 파일을 적용한다. 이 방법은 모든 정책을 다룬 후 설명한다.

---

**확인:** 낮은 clearance 의 주체는 높은 clearance 프로세스를 볼 수 없다. 기본 정책은 쓰기가 필요 하더라도 읽을 수 없는 모든 것에 *mls/high*를 시행한다. 읽기가 필요하더라도 쓸 수 없는 모든 것에 *mls/low*를 시행한다. 그리고 나머지에 *mls/equal*을 시행한다. *Insecure*로 표시된 모든 유저는 *mls/low*에 설정해야 된다.

## 15.12 MAC Biba 모듈

**모듈 이름:** `mac_biba.ko`

**커널 설정 라인:** `options MAC_BIBA`

**부트 옵션:** `mac_biba_load="YES"`

`mac_biba(4)` 모듈은 MAC Biba 정책을 로드한다. 이 정책은 정보흐름이 약간 반대로 되어 있는 것을 제외하고 MLS 정책과 거의 비슷하게 동작한다. 이 정책은 민감한 정보가 아래로 흐르는 것을 방지 하지만 MLS 정책은 민감한 정보가 위로 흐르는 것을 방지한다; 따라서 이 섹션의 대부분을 두 정책에 적용할 수 있다.

Biba 환경에서 “무결성” 라벨을 각 객체나 주체에 설정한다. 이들 라벨은 계층적인 등급과 계층적이지 않은 구획을 생성한다. 주체나 객체의 등급이 올라가기 때문에 무결성도 마찬가지로 올라간다.

아래에서 설명하듯이 지원되는 라벨은 *biba/low*, *biba/equal* 그리고 *biba.high* 다:

- *biba/low* 라벨은 주체나 가지고 있을 객체의 무결성도를 가장 낮게 설정한다. 이 것을 주체나 객체에 설정하면 높게 표시된 주체나 객체에 쓰기 접근을 방지한다.
- 이 정책에서 제외시키기 위해 *biba/equal* 라벨을 주체에 설정할 수 있다.
- *biba/high* 라벨은 낮은 라벨로 설정된 주체에 쓰기를 허용하지만 이 주체를 읽을 수 없다. 이 라벨을 전체 시스템의 무결성에 영향을 주는 주체에 설정할 것을 권장한다.

---

Biba 는 다음과 같은 사항을 제공한다:

- 계층적이지 않은 무결성 카테고리를 설정하여 계층적인 무결성 레벨을 제공한다.
- 작성할 수 없고 읽을 수 없는 고정된 룰을 제공한다(MLS 와 반대). 객체는 상위가 아닌 자신의 레벨에서 주체에 쓰기 접근을 할 수 있다. 비슷하게 객체는 하위가 아닌 자신의 레벨 위의 주체에 읽기 접근을 할 수 있다;
- 무결성(데이터의 부적합한 변경을 방지한다).
- 무결성 레벨(민감한 레벨의 MLS 대신).

다음 sysctl tunables 는 Biba 정책을 조정하는데 사용할 수 있다.

- *security.mac.biba.enabled* 는 타겟 머신에 Biba 정책 시행을 활성화/비활성 하는데 사용할 것이다.
- *security.mac.biba.ptys\_equal* 는 pty(4) 장치에 Biba 정책을 비활성 하는데 사용할 것이다.
- *security.mac.biba.revocation\_enabled* 는 라벨이 객체를 제어하도록 변경되었다면 주체의 접근을 강제로 취소한다.

시스템 주체에 설정된 Biba 정책에 접근하기 위해 setfmac 와 getfmac 명령을 사용한다:

```
# setfmac biba/low test
# getfmac test
test: biba/low
```

**확인:** 무결성이 낮은 객체는 무결성이 높은 객체에 쓰기를 할 수 없다; 무결성이 높은 객체는 확인할 수 없거나 무결성이 낮은 주체를 읽을 수 없다.



---

## 15.13 MAC LOMAC 모듈

모듈 이름: `mac_lomac.ko`

커널 설정 라인: `options MAC_LOMAC`

부트 옵션: `mac_lomac_load="YES"`

MAC Biba 정책과 다르게 `mac_lomac(4)` 정책은 무결성 레벨이 낮아진 후 무결성 률을 깨지 않고 무결성이 더 낮은 주체로 접근하는 것을 허용한다.

예전 `lomac(4)` 시행과 혼동하지 말아야 되는 MAC 버전의 낮은-위터마크 무결성 정책은 예비 등급 구획으로 객체 등급을 낮추기 위해 플로팅(floating) 라벨을 사용하는 것을 제외하고 Biba 와 아주 비슷하다. 이 두 번째 구획은 `[auxgrade]` 형식을 가지고 있다. 예비 등급(auxiliary grade)으로 `lomac` 정책을 할당할 때 숫자 2가 예비 등급인 `lomac10[2]` 형식과 비슷하다.

MAC LOMAC 정책은 무결성이 낮은 주체를 객체가 읽을 수 있게 하고 나중에 무결성이 높은 주체에 쓰기를 방지하기 위해 객체의 라벨을 강등시키는 무결성 라벨과 시스템의 모든 주체의 라벨에 산재해 있다. 이것은 위에서 설명한 `[auxgrade]` 옵션이기 때문에 정책은 더 많은 호환성을 제공하고 Biba 보다 초기 설정이 적다.

### 15.13.1 예제

Biba 와 MLS 정책처럼 `setfmac` 와 `setpmac` 유틸리티가 시스템 주체에 라벨을 적용할 때 사용될 것이다.

```
# setfmac /usr/home/trhodes lomac/high[low]
# getfmac /usr/home/trhodes lomac/high[low]
```

여기서 예비 등급은 `low`이고 MAC LOMAC 정책으로만 제공되는 기능이다.

---

## 15.14 MAC 으로 안전한 환경 구축

다음 데모는 적절히 설정된 정책과 다양한 MAC 모듈을 사용하여 구축한 안전한 환경이다. 이것은 오직 테스트이므로 실제 환경에는 적용하지 않는다. 이 말을 무시한다면 절대 작동하지 않고 실제 환경에서 문제가 될 수 있다.

이 설명을 진행하기 전에 *멀티라벨* 옵션을 각 파일시스템에 설정해야 된다. 설정하지 않으면 에러가 발생한다.

### 15.14.1 insecure 유저 클래스 생성

다음 유저 클래스를 /etc/login.conf 파일에 추가한다:

```
insecure:W
:copyright=/etc/COPYRIGHT:W
:welcome=/etc/motd:W
:setenv=MAIL=/var/mail/$,BLOCKSIZE=K:W
:path=~:/bin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/local/sbin:/usr/local/bin
:manpath=/usr/share/man /usr/local/man:W
:nologin=/usr/sbin/nologin:W
:cputime=1h30m:W
:datasize=8M:W
:vmemoryuse=100M:W
:stacksize=2M:W
:memorylocked=4M:W
:memoryuse=8M:W
:filesize=8M:W
:coredumpsize=8M:W
:openfiles=24:W
:maxproc=32:W
:priority=0:W
:requirehome:W
:passwordtime=91d:W
:umask=022:W
:ignoretime@:W
```

---

```
:label=partition/13,mls/5:
```

그리고 다음 라인을 기본 유저 클래스에 추가한다:

```
:label=mls/equal,biba/equal,partition/equal:
```

이것이 끝나면 데이터베이스를 다시 빌드하도록 다음 명령을 실행한다:

```
# cap_mkdb /etc/login.conf
```

## 15.14.2 정확한 모듈로 부팅

다음 라인을 /boot/loader.conf 에 추가하여 시스템이 초기화하는 동안 필요한 모듈을 로드한다:

```
mac_biba_load="YES"
mac_mls_load="YES"
mac_seeotheruids_load="YES"
mac_partition_load="YES"
```

## 15.14.3 모든 유저를 Insecure 로 설정

root 나 시스템 유저가 아닌 모든 유저 계정은 이제 로그인 클래스가 필요하다. 로그인 클래스가 필요하지만 없다면 유저는 vi(1)처럼 일반적인 명령을 사용하는데 제약을 받는다. 다음 sh 스크립트를 사용하면 된다:

```
# for x in `awk -F: '($3 >= 1001) && ($3 != 65534) { print $1 }' \W
  /etc/passwd`; do pw usermod $x -L insecure; done;
```

위 명령이 실행된 후 /etc/master.passwd 로 cap\_mkdb 명령을 실행해야 된다.

---

## 15.14.4 완벽한 설정

이제 contexts 파일을 생성해야 된다; 다음 예제는 Robert Watson 의 예제 정책으로 /etc/policy.contexts 를 생성한 것이다

```
# This is the default BIBA/MLS policy for this system.

.*                biba/high,mls/high
/sbin/dhclient    biba/high(low),mls/high(low)
/dev(/.*)?        biba/equal,mls/equal
# This is not an exhaustive list of all "privileged" devices.
/dev/mdctl        biba/high,mls/high
/dev/pci          biba/high,mls/high
/dev/k?mem        biba/high,mls/high
/dev/io           biba/high,mls/high
/dev/agp.*        biba/high,mls/high
(/var)?/tmp(/.*)? biba/equal,mls/equal
/tmp/W.X11-unix   biba/high(equal),mls/high(equal)
/tmp/W.X11-unix/. * biba/equal,mls/equal
/proc(/.*)?      biba/equal,mls/equal
/mnt.*           biba/low,mls/low
(/usr)?/home     biba/high(low),mls/high(low)
(/usr)?/home/. * biba/low,mls/low
/var/mail(/.*)?  biba/low,mls/low
/var/spool/mqueue(/.*)? biba/low,mls/low
(/mnt)?/cdrom(/.*)? biba/high,mls/high
(/usr)?/home/(ftp|samba)(/.)? biba/high,mls/high
/var/log/sendmailW.st biba/low,mls/low
/var/run/utmp     biba/equal,mls/equal
/var/log/(lastlog|wtmp) biba/equal,mls/equal
```

이 정책은 왼쪽에 나열된 디렉터리의 유틸리티가 보내고 받는 정보 흐름에 제한을 설정하여 보안을 시행한다.

이 파일을 다음 명령으로 시스템에 읽어 들인다:

---

```
# setfsmac -ef /etc/policy.contexts /
# setfsmac -ef /etc/policy.contexts /usr
```

**Note:** 위의 파일 시스템 레이아웃은 환경에 따라 다를 것이다.

/etc/mac.conf 파일은 메인 섹션에서 다음과 같이 수정해야 된다:

```
default_labels file ?biba,?mls
default_labels ifnet ?biba,?mls
default_labels process ?biba,?mls,?partition
default_labels socket ?biba,?mls
```

## 15.14.5 설정 테스트

이 테스트를 위해 `adduser` 명령으로 유저를 추가하여 *insecure* 클래스에 넣는다.

아래 예제는 root 와 일반 유저를 혼합하여 테스트 한다; 구분하기 위해 프롬프트를 확인한다.

### 15.14.5.1 기본 라벨 테스트

```
% getpmac
biba/15(15-15),mls/15(15-15),partition/15
# setpmac partition/15,mls/equal top
```

**Note:** top 프로세스는 다른 top 프로세스를 실행하기 전에 종료된다.

### 15.14.5.2 MAC Seeotheruids 테스트

---

```
% ps Zax
```

```
biba/15(15-15),mls/15(15-15),partition/15 1096 #C: S 0:00.03 -su (bash)
biba/15(15-15),mls/15(15-15),partition/15 1101 #C: R+ 0:00.01 ps Zax
```

다른 유저들의 프로세스를 보는 것은 허용하지 않는다.

### 15.14.5.3 MAC Partition 테스트

나머지 테스트를 위해 MAC *seeotheruids* 정책을 비활성 한다:

```
# sysctl security.mac.seeotheruids.enabled=0
```

```
% ps Zax
```

LABEL	PID	TT	STAT	TIME
COMMAND				
biba/equal(low-high),mls/equal(low-high),partition/15	1122	#C:	S+	0:00.02
top				
biba/15(15-15),mls/15(15-15),partition/15	1096	#C:	S	0:00.05
-su (bash)				
biba/15(15-15),mls/15(15-15),partition/15	1123	#C:	R+	0:00.01
ps Zax				

Biba 정책은 높은 라벨이 적용된 주체를 읽을 수 있게 한다.

```
# setpmac partition/15,mls/equal,biba/low top
```

```
% ps Zax
```

LABEL	PID	TT	STAT	TIME	COMMAND
biba/15(15-15),mls/15(15-15),partition/15	1096	#C:	S	0:00.07	-su (bash)
biba/15(15-15),mls/15(15-15),partition/15	1226	#C:	R+	0:00.01	ps Zax

Biba 정책은 낮게 라벨 된 주체를 읽지 못하게 하지만 MLS 는 읽을 수 있게 한다.

---

```
% ifconfig bge0 | grep maclabel
maclabel biba/low(low-low),mls/low(low-low)
% ping -c 1 192.0.34.166
PING 192.0.34.166 (192.0.34.166): 56 data bytes
ping: sendto: Permission denied
```

유저는 이러한 이유로 example.com 이나 다른 도메인에 핑을 보낼 수 없다.

이 에러가 발생하는 것을 방지하기 위해 다음 명령을 실행한다:

```
# sysctl security.mac.biba.trust_all_interfaces=1
```

이 명령은 기본 인터페이스 라벨을 insecure 모드로 설정하기 때문에 기본 Biba 정책 라벨은 시행되지 않는다.

```
# ifconfig bge0 maclabel biba/equalW(low-highW),mls/equalW(low-highW)
% ping -c 1 192.0.34.166
PING 192.0.34.166 (192.0.34.166): 56 data bytes
 64 bytes from 192.0.34.166: icmp_seq=0 ttl=50 time=204.455 ms
--- 192.0.34.166 ping statistics ---
 1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max/stddev = 204.455/204.455/204.455/0.000 ms
```

더 정확한 라벨을 설정하여 ping 을 보낼 수 있다.

몇 가지 읽기와 쓰기 테스트를 위해 몇 개의 파일을 생성한다:

```
# touch test1 test2 test3 test4 test5
# getfmac test1
test1: biba/equal,mls/equal
# setfmac biba/low test1 test2; setfmac biba/high test4 test5; W
  setfmac mls/low test1 test3; setfmac mls/high test2 test4
# setfmac mls/equal,biba/equal test3 && getfmac test?
test1: biba/low,mls/low
```

---

```
test2: biba/low,mls/high
test3: biba/equal,mls/equal
test4: biba/high,mls/high
test5: biba/high,mls/equal
# chown testuser:testuser test?
```

이 모든 파일은 testuser 유저가 소유해야 된다. 읽기 테스트를 해보자:

```
% ls
test1 test2 test3 test4 test5
% ls test?
ls: test1: Permission denied
ls: test2: Permission denied
ls: test4: Permission denied
test3 test5
```

이제 쌍을 유지하지 못한다; 예: (*biba/low,mls/low*)와 (*biba/low,mls/high*) 그리고 (*biba/high,mls/high*)쌍. 물론 읽기 접근도 거부된다. 쓰기 테스트를 해보자:

```
% for i in `echo test*`; do echo 1 > $i; done
-su: test1: Permission denied
-su: test4: Permission denied
-su: test5: Permission denied
```

읽기 테스트처럼 쓰기 접근도 쓰기 쌍으로 허용되지 않는다; 예: (*biba/low,mls/high*)와 (*biba/equal,mls/equal*).

```
% cat test?
cat: test1: Permission denied
cat: test2: Permission denied
1
cat: test4: Permission denied
```

이제 root 에서 다음 명령을 실행한다:



---

```
# cat test2
1
```

## 15.15 다른 예제: 웹 서버를 고립시키기 위해 MAC 사용

유저가 접근할 수 있는 웹 데이터를 저장할 분리된 위치를 지정한다. 여기서 *biba/high* 프로세스가 웹 데이터에 접근할 수 있도록 한다.

웹 데이터를 저장할 디렉토리를 생성한다:

```
# mkdir /usr/home/cvs
```

이제 cvs 로 초기화 한다:

```
# cvs -d /usr/home/cvs init
```

첫 번째 목적은 *biba* 정책을 활성화하는 것이기 때문에 */boot/loader.conf* 에 *mac\_biba\_enable="YES"*를 입력한다. 여기서는 MAC 지원을 커널에서 활성화했다고 가정한다.

여기서 시스템의 모든 것에 *biba/high* 를 기본적으로 설정해야 된다.

*login.conf* 파일의 기본 유저 클래스 아래를 다음과 같이 변경한다:

```
:ignoretime@:W
:umask=022:W
:label=biba/high:
```

모든 유저를 기본 클래스에 등록해야 된다; 다음 명령을 사용하면 몇 분 후 완료된다.

```
# for x in `awk -F: '($3 >= 1001) && ($3 != 65534) { print $1 }' W
/etc/passwd`; do pw usermod $x -L default; done;
```

기본 클래스를 복사하여 *biba/low* 라벨 설정으로 web 클래스를 생성한다.

**cv**s 저장소에 저장된 메인 웹 데이터에 작업하는 유저를 생성한다. 이 유저를 새로운 로그인 클래스 web 에 등록한다.

모든 곳에 *biba/high*가 기본이기 때문에 저장소도 마찬가지다. 웹 데이터도 마찬가지이므로 유저가 웹 데이터를 읽고 쓸 수 있다; 그러나 웹 서버는 *biba/high* 유저가 접근해야 되는 데이터를 서비스하기 때문에 전체 데이터를 강등시켜야 된다.

이를 위한 완벽한 툴은 FreeBSD 에서 기본적으로 제공하는 sh(1)과 cron(8)이다. 다음 스크립트가 원하는 것을 해결해준다:

```
PATH=/bin:/usr/bin:/usr/local/bin; export PATH;
CVSROOT=/home/repo; export CVSROOT;
cd /home/web;
cvs -qR checkout -P htdocs;
exit;
```

**Note:** 보통 cvs ID 태그를 웹 사이트 데이터 파일에 적용해야 된다.

이 스크립트를 web 의 홈 디렉터리에 두고 다음 crontab(1) 엔트리를 추가한다:

```
# Check out the web data as biba/low every twelve hours:
0 * * * * web /home/web/checkout.sh
```

이 명령은 매 12 시간마다 머신에서 HTML 소스를 확인한다.

웹 서버를 시작하는 기본 시작 방법도 *biba/low* 에서 프로세스를 시작하도록 수정해야 된다. 이것은 /usr/local/etc/rc.d/apache.sh 스크립트에 다음 명령을 입력하면 된다:

```
command="setpmac biba/low /usr/local/sbin/httpd"
```

---

**Apache** 설정을 *biba/low* 정책으로 동작하도록 수정해야 된다. 이 경우 *biba/low*로 설정된 디렉터리에 로그파일을 추가하도록 소프트웨어를 수정해야 되고 그렇지 않으면 “access denied” 에러를 보게 된다.

**Note:** 이 예제는 *docroot*가 */home/web/htdocs* 를 지정하도록 설정한다. 그렇지 않으면 **Apache** 는 제공할 문서를 저장하려고 할 때 실패한다.

PID 파일, *Scoreboardfile*, *DocumentRoot*, log 파일 위치 또는 쓰기 접근에 필요한 다른 변수를 포함한 다양한 설정도 수정해야 된다. *biba*를 사용할 때 *biba/low*가 설정되지 않은 곳의 모든 쓰기 접근은 거부된다.

## 15.16 MAC 프레임 워크 문제 해결

개발 단계 동안 몇 명의 유저가 일반적인 설정 문제를 보고했다. 이들 문제 중 몇 가지를 설명한다:

### 15.16.1 /에 *multilabel* 옵션을 활성화할 수 없다.

*mltilabel* 플래그가 내 *root(/)* 파티션에서 활성화되지 않았다

50 명 중 한 명의 유저가 이 문제를 겪게 됐고 우리도 초기 설정을 하는 동안 이 문제를 겪게 됐다. 이 문제를 자세히 확인한 결과 이 버그는 정확하지 않은 문서나 문서를 제대로 이해하지 못해서 발생하는 것이었다. 어쨌든 다음 절차로 이 문제를 해결할 수 있다:

- ① *root* 파티션을 읽기만할 수 있도록 */etc/fstab* 에 *ro*로 설정한다.
- ② 싱글 유저모드로 재 부팅한다.
- ③ /에서 *tunefs -l enable* 을 실행한다.
- ④ 보통모드로 시스템을 재 부팅한다.
- ⑤ *mount -urw /*를 실행하고 */etc/fstab* 의 *ro*를 *rw*로 변경한 후 시스템을 다시 재

---

부팅한다.

- ⑥ root 파일시스템에 *멀티라벨*이 제대로 설정되었는지 확인하기 위해 mount 결과를 다시 확인한다.

## 15.16.2 MAC 설정 이후 Xfree86 이 시작되지 않는다.

MAC 으로 안전한 환경을 구축한 후 Xfree86 을 실행할 수 없다.

이 문제는 MAC *파티션* 정책이나 MAC 라벨정책 중 잘못된 라벨로 발생할 수 있다. 디버그 하려면 다음 사항을 따른다:

- ① 에러 메시지를 확인한다; 사용자가 *insecure* 클래스에 있다면 *파티션* 정책이 문제가 된다. 유저의 클래스를 다시 *default* 클래스로 설정하고 `cap_mkdb` 명령으로 데이터베이스를 다시 빌드한다. 그래도 해결되지 않는다면 다음 단계를 시행한다.
- ② 라벨 정책을 다시 확인한다. 정책이 유저, XFree86 어플리케이션 그리고 `/dev` 엔트리에 정확히 설정되어있는지 확인한다.
- ③ 둘 다 문제를 해결하지 못한다면 에러 메시지와 환경을 설명한 메일을 TrustedBSD 웹 사이트(<http://www.trustedbsd.org/>)에 있는 TrustedBSD 토론 리스트나 FreeBSD 일반 질문 메일링 리스트 (<http://lists.freebsd.org/mailman/listinfo/freebsd-questions>)에 보낸다.

---

## 16 장 스토리지

### 16.1 개요

이번 장에서는 FreeBSD 에서 디스크 사용에 대해 다룬다. 여기에는 메모리와 네트워크 기반 디스크 그리고 표준 SCSI/IDE 스토리지 장치가 포함된다.

이번 장을 읽고 다음과 같은 사항을 알 수 있다:

- 물리적인 디스크에서(파티션과 슬라이스) 데이터 구성을 설명하기 위해 FreeBSD 에서 사용하는 용어.
- 시스템에 디스크를 어떻게 추가하는가
- USB 저장 장치를 사용하기 위해 FreeBSD 를 어떻게 설정하는가
- 메모리 디스크와 같은 가상 파일 시스템은 어떻게 설정하는가
- 디스크 사용량을 제한하기 위해 쿼타는 어떻게 사용하는가
- 침입에 대비하여 디스크는 어떻게 암호화하는가
- FreeBSD 에서 CD 와 DVD 는 어떻게 생성하는가
- 백업을 위한 다양한 스토리지 미디어 옵션
- FreeBSD 에서 백업 프로그램은 어떻게 사용하는가
- 플로피 디스크에 어떻게 백업하는가.
- 스냅샷이 무엇이고 효과적으로 사용방법은 무엇인가

이번 장을 읽기 전에 다음 사항을 알고 있어야 된다:

- 새로운 FreeBSD 커널을 어떻게 설정하고 설치하는지 알고 있어야 된다 (8 장).

---

## 16.2 장치 이름

다음은 FreeBSD 가 지원하는 물리적인 스토리지 장치 리스트고 장치 이름은 이들 장치와 관련이 있다.

테이블 16 -1. 물리적인 디스크 이름 관례

드라이브 종류	드라이브 장치 이름
IDE 하드 드라이브	<i>ad</i>
IDE CDROM 드라이브	<i>acd</i>
SCSI 하드 드라이브와 USB 스토리지 장치	<i>da</i>
SCSI CDROM 드라이브	<i>cd</i>
여러 가지 비 표준 CDROM 드라이브	Mitsumi CD-ROM 은 <i>mcd</i> , Sony CD-ROM 은 <i>scd</i> , Matsushita/Panasonic CD-ROM 은 <i>matcd</i> ❖
Floppy 드라이브	<i>fd</i>
SCSI 테잎 드라이브	<i>sa</i>
IDE 테잎 드라이브	<i>ast</i>
Flash 드라이브	DiskOnChip® Flash 장치의 <i>fla</i> .
RAID 드라이브	Adaptec® AdvancedRAID 는 <i>aacd</i> , Mylex®는 <i>mlx</i> d 와 <i>mly</i> d, AMI MegaRAID®는 <i>amrd</i> , Compaq Smart RAID 는 <i>idad</i> , 3ware® RAID 는 <i>twed</i> .
<b>Notes:</b> ❖ <i>matcd</i> (4) 드라이버는 2002년 10월 5일에 FreeBSD 4.X 분기에서 삭제되었기 때문에 FreeBSD 5.0 과 5.1 에는 없다. 그러나 이 드라이버는 2003년 6월 16일부터 5.X 분기에 추가되었다.	

## 16.3 디스크 추가

드라이브가 하나만 있는 머신에 새로운 SCSI 디스크를 추가하려고 한다. 컴퓨터를 끄고 컴퓨터와 컨트롤러 그리고 드라이브 제조사의 매뉴얼대로 드라이브를 설치한다. 드라이브 설치하는 아주 다양한 절차가 필요하기 때문에 자세한 설명은 이 문서의 범위를 벗어난다.

`root` 로 로그인 한다. 드라이브를 설치하고 `/var/run/dmesg.boot` 에서 새로운 디스크를 감지

---

했는지 확인한다. 예제를 계속하면 새로 추가된 드라이브는 da1 이 되고 우리는 /1 에(IDE 드라이브를 추가했다면 장치 이름은 4.0 이전 시스템에서는 wd1 이 되거나 대부분의 4.X 시스템에서는 ad1 이 된다.) 마운트하려고 한다.

FreeBSD 는 IBM-PC 호환 컴퓨터에서 운용되기 때문에 PC BIOS 파티션에 등록을 해야 된다. 이 의미는 전통적인 BSD 파티션과 다르다. PC 디스크는 4 개까지 BIOS 파티션 엔트리를 가지고 있다. 디스크를 FreeBSD 전용으로 사용한다면 *dedicated* 모드를 사용할 수 있다. 그렇지 않다면 FreeBSD 는 PC BIOS 파티션 중 하나가 된다. FreeBSD 는 PC BIOS 파티션을 슬라이스라고 부르기 때문에 전통적인 BSD 파티션과 혼동하지 않도록 한다. FreeBSD 전용 디스크에서도 슬라이스를 사용하겠지만 다른 운영체제가 설치된 컴퓨터에서도 사용된다. 따라서 다른 운영체제의 fdisk 유틸리티와 혼동하지 않는다.

슬라이스의 경우 드라이브는 /dev/da1s1e 처럼 추가된다. 이것을 SCSI 디스크, 유닛 번호 1(두 번째 SCSI 디스크), 슬라이스 1(PC BIOS 파티션 1) 그리고 BSD 파티션 e 로 읽는다. FreeBSD 전용의 경우 이 드라이브는 간단히 /dev/da1e 에 추가된다.

### 16.3.1 sysinstall(8)을 사용하여 디스크 추가하기

#### ① Sysinstall 탐색

/stand/sysinstall 의 사용하기 쉬운 메뉴로 새로운 디스크를 파티션해서 라벨을 할 수 있다. root 유저 또는 su 명령을 사용하여 /stand/sysinstall 을 실행하고 *Configure* 메뉴에 들어간다. FreeBSD *Configuration* 메뉴에서 아래로 이동하여 *Fdisk* 옵션을 선택한다.

```
Disk name: da0 FDISK Partition Editor
DISK Geometry: 4467 cyls/255 heads/63 sectors = 71762355 sectors (35040MB)

Offset      Size(ST)      End      Name PType      Desc Subtype  Flags
-----
0 71775284 71775283 - 6 unused 0

The following commands are supported (in upper or lower case):
A = Use Entire Disk      G = set Drive Geometry  C = Create Slice      F = `DD` mode
D = Delete Slice        Z = Toggle Size Units   S = Set Bootable     I = Wizard m.
T = Change Type         U = Undo All Changes    W = Write Changes
```

### ② fdisk 파티션 편집기

fdisk 안에서 A를 눌러 전체 디스크를 FreeBSD에 할당할 수 있다. "remain cooperative with any future possible operating systems"라는 질문에 YES로 대답한다. W로 변경된 내용을 디스크에 저장하고 q를 눌러서 FDISK에서 빠져 나온다. 다음은 마스터 부트 레코드에 대한 질문이다. 실행중인 시스템에 디스크를 추가했기 때문에 None을 선택한다.

```
Disk name: da0 FDISK Partition Editor
DISK Geometry: 4467 cyls/255 heads/63 sectors = 71762355 sectors (35040MB)

Offset      Size(ST)      End      Name  PType      Desc  Subtype  Flags
      0         63          62      -     6      unused    0
      63    71762292    71762354  da0s1  3      freebsd   165     C
71762355    12929      71775283  -     6      unused    0

The following commands are supported (in upper or lower case):
A = Use Entire Disk      G = set Drive Geometry  C = Create Slice      F = "DD" mode
D = Delete Slice        Z = Toggle Size Units   S = Set Bootable     I = Wizard m.
T = Change Type         U = Undo All Changes    W = Write Changes
```

### ③ 디스크 라벨 편집기

sysinstall를 빠져 나온 후 다시 시작한다. 여기서 Label 옵션을 선택하더라도 위에서 설명한 곳으로 이동한다. Disk Label Editor에 들어간다. 여기서 전통적인 BSD 파티션을 생성한다. 하나의 디스크는 라벨 a-h로 8개의 파티션으로 나눌 수 있다. 파티션 라벨 중 몇 개는 특별하게 사용된다. a 파티션은 root 파티션(/)으로 사용된다. 그래서 시스템 디스크는(다시 말해 부팅할 수 있는 디스크) a 파티션을 가지고 있어야 된다. b 파티션은 스왑 파티션으로 사용되기 때문에 스왑 파티션으로 많은 디스크를 가지고 있을 것이다. c 파티션은 전체 디스크를 FreeBSD 전용 모드 또는 슬라이스 모드에서 전체를 FreeBSD 슬라이스로 사용할 것을 의미한다. 다른 파티션은 일반적으로 사용된다.

sysinstall의 라벨 편집기는 파티션 e를 root나 스왑 파티션으로 사용하지 않는다. 라벨 편집기에서 C를 입력해서 파일시스템 하나를 생성한다. 이것이 FS(파일시스템) 또는 swap 용인지 묻는 프롬프트가 나타나면 FS를 선택하고 마운트 포인트를(예: /mnt) 입력한다. 설치 후 모드에서 디스크를 추가했을 때 sysinstall이



/etc/fstab 에 엔트리를 생성하지 않았기 때문에 마운트 포인트 지정은 중요하지 않다.

```

FreeBSD Disklabel Editor
Disk: da0      Partition name: da0s1  Free: 71762292 blocks (35040MB)
Part      Mount      Size Newfs  Part      Mount      Size Newfs
-----

```

Value Required

Please specify the partition size in blocks or append a trailing G for gigabytes, M for megabytes, or C for cylinders.  
71762292 blocks (35040MB) are free.

71762292

[ OK ]      [ Cancel ]

```

The
C = Create      D = Delete    M = Mount pt.  W = Write
N = Newfs Opts  Q = Finish    S = Toggle SoftUpdates
T = Toggle Newfs U = Undo      A = Auto Defaults  R = Delete+Merge

```

이제 **W** 를 입력해서 새로운 라벨을 디스크에 작성하고 파일시스템을 만든다. **sysinstall** 의 에러를 무시하면 새로운 파티션을 마운트할 수 없다. 라벨 편집기에서 나와서 **sysinstall** 를 완전히 빠져 나온다.

#### ④ 끝내기

마지막 단계는 새로운 디스크 엔트리를 /etc/fstab 에 추가한다.

```

# See the fstab(5) manual page for important information on automatic mounts
# of network filesystems before modifying this file.
#
# Device      Mountpoint  FStype  Options      Dump  Pass#
/dev/da0s1b  none       swap    sw           0     0
/dev/da0s1a  /          ufs     rw           1     1
/dev/da0s1f  /tmp       ufs     rw           2     2
/dev/da0s1g  /usr       ufs     rw           2     2
/dev/da0s1e  /var       ufs     rw           2     2
/dev/acd0c   /cdrom     cd9660  ro,noauto    0     0
proc        /proc      procfs  rw           0     0

```

## 16.3.2 명령어 라인 유틸리티를 사용하여 디스크추가

### 16.3.2.1 슬라이스 사용

---

이 설정은 컴퓨터에 설치된 다른 운영체제와 디스크가 정확히 작동하도록 하기 때문에 다른 운영체제의 fdisk 유틸리티와 혼동하지 않는다. 새로운 디스크를 설치할 때 이 방법을 사용하도록 권장한다. 이 방법을 사용하겠다면 *dedicated* 모드만 사용한다.

```
# dd if=/dev/zero of=/dev/da1 bs=1k count=1
# fdisk -Bl da1 ❶
# disklabel -B -w -r da1s1 auto ❷
# disklabel -e da1s1 ❸
# mkdir -p /1
# newfs /dev/da1s1e ❹
# mount /dev/da1s1e /1 ❺
# vi /etc/fstab ❻
```

- ❶ 새로운 디스크를 초기화한다.
- ❷ 라벨을 할당한다.
- ❸ 방금 생성한 파티션의 라벨을 수정한다.
- ❹ 원하는 파티션을 생성할 때까지 이 명령을 반복한다.
- ❺ 파티션을 마운트한다.
- ❻ 적절한 엔트리를 /etc/fstab 에 추가한다.

IDE 디스크를 가지고 있으면 da 대신 ad 를 사용한다. 이전 FreeBSD 4.X 시스템에는 wd 를 사용한다.

### 16.3.2.2 FreeBSD 전용모드

다른 운영체제와 새로운 드라이브를 같이 사용하지 않는다면 *dedicated* 모드를 사용한다. 이 모드는 Microsoft 운영체제가 혼동할 수 있겠지만 손상을 입히지는 않을 것이다. 그러나 IBM 의 OS/2 는 이 파티션을 이해하지 못한다.

```
# dd if=/dev/zero of=/dev/da1 bs=1k count=1
# disklabel -Brw da1 auto
# disklabel -e da1 ❶
# newfs -d0 /dev/da1e
# mkdir -p /1
# vi /etc/fstab ❷
```

---

```
# mount /1
```

- ❶ e 파티션을 생성한다.
- ❷ /etc/fstab 에 /dev/da1e 엔트리를 추가한다.

다른 방법은:

```
# dd if=/dev/zero of=/dev/da1 count=2
# disklabel /dev/da1 | disklabel -BrR da1 /dev/stdin
# newfs /dev/da1e
# mkdir -p /1
# vi /etc/fstab                                ❶
# mount /1
```

- ❶ /etc/fstab 에 /dev/da1e 엔트리를 추가한다.

**Note:** FreeBSD 5.1-릴리즈 이후부터 `bsdlabel(8)` 유틸리티가 예전 `disklabel(8)` 프로그램으로 대체한다. `bsdlabel(8)`로 쓰지 않는 다양한 옵션과 매개변수가 없어졌다; 위의 예제에서 옵션 `-r`은 `bsdlabel(8)`로 제거됐다. 더 많은 정보는 `bsdlabel(8)` 매뉴얼 페이지를 참고한다.

## 16.4 RAID

### 16.4.1 소프트웨어 레이드

#### 16.4.1.1 연속된 디스크 드라이버 (CCD) 설정

대형 스토리지 솔루션을 선택할 때 가장 중요한 요소는 속도, 안정성 그리고 가격을 검토해야 된다. 3 가지를 모두 갖춘 제품은 찾기가 힘들다; 보통 빠르고 안정적인 대형 스토리지 장치는 비싸기 때문에 비용을 줄이기 위해 속도나 안정성을 감소시켜야 된다.

아래에서 설명하는 시스템 디자인은 가격을 최우선으로 꼽고 속도와 안정성을 따진다. 이

---

시스템에서 데이터 전송 속도는 결국 네트워크에 따라 다르다. 반면 안정성이 매우 중요하기 때문에 아래에서 설명하는 CCR 드라이브는 이미 CD-R 에 풀 백업되어 쉽게 대체할 수 있는 온라인 데이터를 제공한다.

필요한 사항을 규정하는 것이 대형 스토리지 솔루션 선택의 첫 번째 단계다. 속도나 안정성을 비용보다 중요시 한다면 이번 섹션에서 설명하는 시스템과 다를 것이다.

### 16.4.1.2 하드웨어 설치

대략 90GB 의 온라인 스토리지를 제공하기 위해 아래에서 설명하는 CCD 디스크의 코어 방식으로 Western Digital 30GB 5400 RPM IDE 디스크를 시스템에 추가한다. 최고의 성능을 위해 각 IDE 디스크는 각각의 컨트롤러와 케이블에 연결되어 있어야 하지만 최소 가격을 위해 추가적인 IDE 컨트롤러를 사용하지 않는다. 대신 디스크를 점퍼로 설정하여 각 IDE 컨트롤러에 하나는 마스터를 다른 하나는 슬레이브를 연결한다.

재 부팅하면 시스템 BIOS 는 자동으로 추가된 디스크를 감지한다. 더욱 중요한 것은 FreeBSD 가 재 부팅할 때 디스크를 검색한다는 것이다:

```
ad0: 19574MB <WDC WD205BA> [39770/16/63] at ata0-master UDMA33
ad1: 29333MB <WDC WD307AA> [59598/16/63] at ata0-slave UDMA33
ad2: 29333MB <WDC WD307AA> [59598/16/63] at ata1-master UDMA33
ad3: 29333MB <WDC WD307AA> [59598/16/63] at ata1-slave UDMA33
```

**Note:** FreeBSD 가 모든 디스크를 감지하지 못했다면 점퍼를 정확하게 설정했는지 확인한다. 대부분의 IDE 드라이브는 "케이블 선택" 방식의 점퍼를 가지고 있다. 이것은 마스터/슬레이브와 관련된 점퍼가 아니다. 정확한 점퍼를 확인하기 위해 드라이브 문서를 참고한다.

파일시스템에 이들을 어떻게 추가할지 생각한다. `vinum(8)`과(17 장) `ccd(4)`를 사용한다. 여기서는 `ccd(4)`를 사용한다.

### 16.4.1.3 CCD 설정

---

ccd(4) 드라이버는 몇 개의 동일한 디스크를 연결하여 하나의 논리적인 파일시스템으로 만들 수 있다. ccd(4)를 사용하려면 ccd(4)를 지원하도록 커널을 빌드해야 된다. 다음 라인을 커널 설정파일에 추가하고 커널을 다시 빌드해서 설치한다:

```
pseudo-device  ccd    4
```

5.X 시스템에서는 다음 라인을 대신 사용한다:

```
device  ccd
```

**Note:** FreeBSD 5.X에서는 ccd(4) 장치 드라이버가 자가 복제를 하기 때문에 ccd(4) 장치 개수를 지정할 필요가 없다. 새로운 장치 인스턴스는 요청이 있으면 자동으로 생성된다.

FreeBSD 3.0 또는 이후 버전에서 ccd(4) 지원은 커널이 로드할 수 있는 모듈처럼 로드할 수 있다.

ccd(4)를 설정하려면 첫째로 디스크에 라벨을 할당하기 위해 disklabel(8)을 사용해야 된다:

```
# disklabel -r -w ad1 auto
# disklabel -r -w ad2 auto
# disklabel -r -w ad3 auto
```

이 명령은 ad1c, ad2c 그리고 ad3c 를 하나의 디스크로 연결하도록 디스크 라벨을 생성한다.

다음 단계는 디스크 라벨 종류를 변경한다. 디스크를 편집하기 위해 disklabel(8)을 사용할 수 있다:

```
# disklabel -e ad1
# disklabel -e ad2
# disklabel -e ad3
```

이 명령은 EDITOR 환경변수에 지정된 에디터로(보통 vi(1)) 각 디스크의 현재 디스크 라벨을 연다.

---

수정하지 않은 디스크 라벨은 다음과 비슷하게 보인다:

```
8 partitions:
#          size  offset  fstype  [fsize bsize bps/cpg]
c: 60074784      0  unused      0    0    0 # (Cyl.  0 - 59597)
```

ccd(4)가 사용할 수 있도록 새로운 파티션 *e*를 추가한다. 이것은 보통 *c* 파티션에서 복사할 수 있지만 *fstype*은 4.2BSD로 만들어야 한다. 이제 디스크 라벨은 다음과 비슷하게 보인다:

```
8 partitions:
#          size  offset  fstype  [fsize bsize bps/cpg]
c: 60074784      0  unused      0    0    0 # (Cyl.  0 - 59597)
e: 60074784      0  4.2BSD      0    0    0 # (Cyl.  0 - 59597)
```

#### 16.4.1.4 파일시스템 생성

ccd0c의 장치 노드가 없기 때문에 다음 명령을 실행해서 생성해야 된다:

```
# cd /dev
# sh MAKEDEV ccd0
```

**Note:** FreeBSD 5.0에서 devfs(5)가 /dev의 장치 노드를 자동으로 관리하기 때문에 MAKEDEV는 필요없다.

이제 모든 디스크 라벨을 할당했으므로 ccd(4)를 빌드해야 된다. 빌드하려면 다음과 비슷한 옵션으로 ccdconfig(8)를 사용한다:

```
# ccdconfig ccd01 322 03 /dev/ad1e4 /dev/ad2e /dev/ad3e
```

각 옵션의 사용과 의미는 다음과 같다:

- 
- ❶ 첫 번째 인자는 이 예제의 경우 `/dev/ccd0c` 인 설정하려는 장치를 의미한다. `/dev/` 부분은 옵션이다.
  - ❷ 파일시스템의 인터리브(**interleave**: 성능을 높이기 위해 데이터가 서로 인접하지 않도록 배열하는 방법). 인터리브는 디스크 블록에서 스트라이프(stripe) 크기를 정의하고 보통 512 바이트다. 그래서 32의 인터리브는 16,384 바이트가 된다.
  - ❸ `ccdconfig(8)`의 플래그. 드라이브 미러링을 원한다면 여기에 특정 플래그를 지정할 수 있다. 이 설정은 `ccd(4)`에 미러링을 제공하지 않기 때문에 0으로 설정한다.
  - ❹ `ccdconfig(8)`의 마지막 인자는 장치를 어레이로 만든다. 각 장치는 절대 경로를 사용한다.

`ccdconfig(8)`를 실행하면 `ccd(4)`가 설정된다. 그래서 파일시스템을 설치할 수 있다. 옵션은 `newfs(8)`를 참고하거나 단순히 다음 명령을 실행한다:

```
# newfs /dev/ccd0c
```

### 16.4.1.5 자동화하기

보통 재 부팅할 때마다 `ccd(4)`를 자동으로 마운트하려면 다음 명령으로 현재 설정을 `/etc/ccd.conf`에 작성한다:

```
# ccdconfig -g > /etc/ccd.conf
```

`/etc/ccd.conf`가 있다면 재 부팅하는 동안 `/etc/rc` 스크립트는 `ccdconfig -C`를 실행한다. 따라서 자동으로 `ccd(4)`를 설정하여 마운트할 수 있다.

**Note:** `ccd(4)`를 `mount(8)`하기 전에 싱글 유저모드로 부팅했다면 어레이를 설정하도록 다음 명령을 실행해야 된다:

```
# ccdconfig -C
```

자동으로 `ccd(4)`를 마운트하도록 `ccd(4)` 엔트리를 `/etc/fstab`에 생성해서 부팅할 때 마운트 한다:

---

<code>/dev/ccd0c</code>	<code>/media</code>	<code>ufs</code>	<code>rw</code>	<code>2</code>	<code>2</code>
-------------------------	---------------------	------------------	-----------------	----------------	----------------

### 16.4.1.6 The Vinum 볼륨 매니저

가상 디스크 드라이브 톨인 Vinum 볼륨 매니저는 블록 장치 드라이버이다. 디스크 하드웨어를 블록 장치 인터페이스와 맵 데이터에서 분리하여 전통적인 디스크 스토리지와 비교하여 유연성과 성능 그리고 안정성을 증가시켰다. `vinum(8)`은 RAID-0, RAID-1 과 RAID-5 모델을 따로 또는 결합하여 구성한다.

`vinum(8)`에 대한 더 많은 정보는 17 장을 본다.

### 16.4.2 하드웨어 RAID

그리고 FreeBSD 는 다양한 하드웨어 레이드 컨트롤러를 지원한다. 이런 장치는 FreeBSD 에 어레이 관리를 위한 소프트웨어 없이 RAID 서브시스템을 제어한다.

카드에 있는 BIOS 를 사용하여 카드는 디스크의 대부분을 제어한다. 다음은 Promise IDE RAID 컨트롤러를 사용하기 위한 설정을 간단히 요약한 것이다. 이 카드를 설치하면 시스템이 시작될 때 필요한 정보를 요청하는 프롬프트를 보여준다. 지시에 따라 카드의 설정 화면으로 들어간다. 여기서는 모든 드라이브를 결합하는 기능을 사용해서 디스크가 FreeBSD 에 하나의 드라이브처럼 보인다. 다른 RAID 레벨도 적절히 설정할 수 있다.

### 16.4.3 ATA RAID1 어레이 다시 빌드하기

FreeBSD 는 어레이에서 문제가 발생한 디스크를 바로 교체할 수 있게 한다. 이것은 재 부팅하기 전에 감지해야 된다.

다음과 비슷한 메시지를 `/var/log/messages` 또는 `dmesg(8)` 결과에서 볼 수 있다:

<pre>ad6 on monster1 suffered a hard error. ad6: READ command timeout tag=0 serv=0 - resetting ad6: trying fallback to PIO mode</pre>
---



---

```
ata3: resetting devices .. done
ad6: hard error reading fsbn 1116119 of 0-7 (ad6 bn 1116119; cn 1107 tn 4 sn 11)
status=59 error=40
ar0: WARNING - mirror lost
```

atacontrol(8)로 더 많은 정보를 체크한다:

```
# atacontrol list
ATA channel 0:
  Master:      no device present
  Slave:      acd0 <HL-DT-ST CD-ROM GCR-8520B/1.00> ATA/ATAPI rev 0

ATA channel 1:
  Master:      no device present
  Slave:      no device present

ATA channel 2:
  Master:      ad4 <MAXTOR 6L080J4/A93.0500> ATA/ATAPI rev 5
  Slave:      no device present

ATA channel 3:
  Master:      ad6 <MAXTOR 6L080J4/A93.0500> ATA/ATAPI rev 5
  Slave:      no device present

# atacontrol status ar0
ar0: ATA RAID1 subdisks: ad4 ad6 status: DEGRADED
```

- ① 첫째로 어레이에서 디스크를 분리하여 안전하게 제거할 수 있다:

```
# atacontrol detach 3
```

- ② 디스크를 분리한다.

- 
- ③ 예비 디스크로 교체한다:

```
# atacontrol attach 3
Master:  ad6 <MAXTOR 6L080J4/A93.0500> ATA/ATAPI rev 5
Slave:   no device present
```

- ④ 어레이를 다시 빌드한다:

```
# atacontrol rebuild ar0
```

- ⑤ 다시 빌드하는 명령은 끝날 때까지 멈춰있다. 그렇지만 다른 터미널을(Alt+Fn 사용) 열고 다음 명령으로 진행 사항을 체크할 수 있다:

```
# dmesg | tail -10
[output removed]
ad6: removed from configuration
ad6: deleted from ar0 disk1
ad6: inserted into ar0 disk1 as spare

# atacontrol status ar0
ar0: ATA RAID1 subdisks: ad4 ad6 status: REBUILDING 0% completed
```

- ⑥ 이 작업이 끝날 때까지 기다린다.

## 16.5 USB 스토리지 장치

요즘 수많은 외장형 스토리지 솔루션들이 Universal Serial Bus(USB)를 사용한다: 하드 드라이브, USB 스토리지, CD-R 레코더 등. FreeBSD는 이들 장치를 지원한다.

---

## 16.5.1 설정

USB 스토리지 장치 드라이버 `umass(4)`가 USB 스토리지 장치를 지원한다. GENERIC 커널을 사용한다면 설정을 변경할 필요 없다. 사용자 커널을 사용한다면 커널 설정파일에 다음 라인들이 필요하다:

```
device scbus
device da
device pass
device uhci
device ohci
device usb
device umass
```

`umass(4)` 드라이버는 USB 스토리지 장치를 사용하기 위해 SCSI 서브시스템을 사용하므로 USB 장치는 시스템에서 SCSI 장치처럼 보인다. 마더보드의 USB 칩셋에 따라 `device uhci`와 `device ohci` 중 하나만 필요하지만 커널 설정파일에 둘 다 가지고 있어도 문제는 없다. 라인을 추가하고 새로운 커널을 컴파일해서 설치해야 된다.

**Note:** USB 장치가 CD-R 레코더면 SCSI CD-ROM 드라이버 `cd(4)`를 다음 라인으로 커널에 추가해야 된다:

```
device cd
```

레코더가 SCSI 드라이브처럼 보이기 때문에 `atapicam(4)` 드라이버를 커널 설정에 사용하지 않는다.

USB 2.0 컨트롤러는 FreeBSD 5.X와 FreeBSD 4.X는 FreeBSD 4.10-RELEASE부터 지원한다. 다음 라인을 커널 설정파일에 추가한다:

```
device ehci
```

USB 1.X를 지원하려면 `uhci(4)`와 `ohci(4)` 드라이버가 필요하다.

**Note:** FreeBSD 4.X에서 USB 데몬(`usbd(8)`)이 USB 장치를 감지할 수 있도록 실행

---

해야 된다. 이렇게 하려면 `usb_enable="YES"`를 `/etc/rc.conf` 파일에 넣고 머신을 재 부팅한다.

## 16.5.2 설정 테스트

설정을 테스트할 준비가 됐다: USB 장치를 꽂으면 시스템의 메시지 버퍼에(`dmesg(8)`) 다음과 같이 드라이브가 나타난다:

```
umass0: USB Solid state disk, rev 1.10/1.00, addr 2
GEOM: create disk da0 dp=0xc2d74850
da0 at umass-sim0 bus 0 target 0 lun 0
da0: <Generic Traveling Disk 1.11> Removable Direct Access SCSI-2 device
da0: 1.000MB/s transfers
da0: 126MB (258048 512 byte sectors: 64H 32S/T 126C)
```

물론 장치 노드(`da0`)와 다른 사항은 여러분의 설정과 다르다.

USB 장치가 SCSI 장치로 보이기 때문에 시스템에 꽂혀 있는 USB 스토리지 장치를 보기 위해 `camcontrol` 명령을 사용할 수 있다:

```
# camcontrol devlist
<Generic Traveling Disk 1.11>      at scbus0 target 0 lun 0 (da0,pass0)
```

장치가 파일시스템이면 마운트할 수 있다. 필요하다면 16.3 장을 따라 USB 드라이브에 파티션을 생성하고 포맷할 수 있다.

장치를 빼면(디스크를 먼저 언마운트 한다) 시스템 메시지 버퍼에서 다음과 비슷한 내용을 보게 된다:

```
umass0: at uhub0 port 1 (addr 2) disconnected
(da0:umass-sim0:0:0:0): lost device
(da0:umass-sim0:0:0:0): removing device entry
GEOM: destroy disk da0 dp=0xc2d74850
umass0: detached
```

---

### 16.5.3 더 많은 정보

디스크 추가(16.3 장), 파일시스템 마운트 및 언마운트 섹션(3.6 장)과 다양한 매뉴얼 페이지가 유용하다: `umass(4)`, `camcontrol(8)`, 그리고 `usbdevs(8)`.

## 16.6 광학 미디어 생성과 사용 (CD & DVD)

### 16.6.1 소개

CD는 전통적인 디스크와 다른 여러 가지 특징을 가지고 있다. 처음에는 CD에 유저가 데이터를 작성할 수 없었다. CD는 헤드가 트랙 사이를 움직이는 동안 지체되지 않고 계속해서 읽을 수 있도록 디자인되었다. CD는 시스템 사이에서 그 당시 비슷한 크기의 다른 미디어보다 쉽게 데이터를 전송했다.

CD는 트랙을 가지고 있지만 계속해서 읽기 위한 데이터의 섹션을 의미할 뿐 디스크의 물리적인 특성을 의미하지 않는다. FreeBSD에서 CD를 만들려면 CD 트랙에 생성할 데이터 파일을 준비하여 CD 트랙에 작성한다.

ISO 9660 파일시스템은 이런 이유로 디자인되었다. 불행히 일반적인 것보다 파일시스템 분류에 한계가 있다. 다행히 CD를 알맞게 작성할 수 있도록 이런 한계를 보충하는 확장 메커니즘을 제공하지만 이런 확장을 지원하지 않는 시스템이 아직도 존재한다.

`sysutils/mkisofs` 프로그램은 ISO 9660 파일시스템을 가지고 있는 데이터 파일을 생성하는데 사용된다. 이 프로그램은 다양한 확장을 지원하는 옵션을 가지고 있고 아래에서 설명할 것이다. `sysinstalls/mkisofs` 포트에서 설치할 수 있다.

CD 레코더의 방식에 따라 CD를 레코드하는 툴이 결정된다. ATAPICD 레코더는 기본 시스템에 포함된 `burncd` 프로그램을 사용한다. SCSI와 USB CD 레코더는 `sysutils/cdrtools` 포트에서 `cdrecord`를 사용한다.

`burncd`는 지원하는 드라이브 수에 제한이 있다. 드라이브가 지원되는지 보려면 지원하는

---

CD-R/RW 드라이브 리스트를 본다(<http://www.freebsd.dk/ata/>).

**Note:** FreeBSD 4.8-릴리즈 버전이나 더 높은 FreeBSD 5.X 를 사용한다면 `cdrecord` 와 ATAPI/CAM 모듈의 ATAPI 하드웨어에서 SCSI 드라이버용도 사용할 수 있다.

그래픽 유저 인터페이스(GUI) CD 레코딩 소프트웨어를 찾는다면 **X-CD-Roast** 나 **K3b** 을 확인해 본다. 이들 둘은 패키지나 `sysutils/xcdroast` 와 `sysutils/k3b` 포트에서 사용할 수 있다. **X-CD-Roast** 와 **K3b** 는 ATAPI 하드웨어와 ATAPI/CAM 모듈이 필요하다.

## 16.6.2 mkisofs

`sysutils/mkisofs` 는 유닉스 파일시스템에 디렉터리 트리 이미지인 ISO 9660 파일시스템을 생성한다. 간단한 사용 법은 다음과 같다:

```
# mkisofs -o imagefile.iso /path/to/tree
```

이 명령은 `/path/to/tree` 를 복사한 ISO 9660 파일시스템을 가지고 있는 `imagefile.iso` 를 생성한다. 이 과정에서 표준 ISO 9660 파일시스템에 맞는 이름으로 파일 이름을 변경하고 ISO 파일시스템 특성이 아닌 이름을 가진 파일은 제외시킨다.

다양한 옵션으로 이런 한계를 극복할 수 있다. 특히 `-R` 은 유닉스 시스템에 일반적인 Rock Ridge 를 확장, `-J` 는 Microsoft 시스템이 사용하는 Joliet 확장을 활성화하고 `-hfs` 는 MacOS 가 사용하는 HFS 파일시스템을 생성하는데 사용할 수 있다.

FreeBSD 시스템에서만 사용하려는 CD 는 `-U` 로 모든 파일이름 제한을 해결할 수 있다. `-R` 을 사용하면 ISO 9660 표준을 깨는 방법이라도 여러분이 시작한 FreeBSD 트리와 동일한 파일시스템 이미지를 만든다.

보통 사용하는 마지막 옵션은 `-b` 다. 이 옵션은 "티 Torito" 부팅 가능한 CD 를 만들 때 사용하도록 부트 이미지 위치를 지정할 때 사용된다. 이 옵션은 CD 로 작성할 트리의 최상단에서 부트 이미지 경로를 인자로 갖는다. 지정한 `/tmp/myboot` 는 `/tmp/myboot/boot/cdboot` 의 부트 이미지로 부팅 가능한 FreeBSD 시스템을 가지고 있다. 따라서 다음과 같은 명령으로 `/tmp/bootable.iso` 에 ISO 9660 파일시스템 이미지를 만들 수 있다:

---

```
# mkisofs -U -R -b boot/cdboot -o /tmp/bootable.iso /tmp/myboot
```

위의 명령이 끝나고 커널에 vn(FreeBSD 4.X)이나 md(FreeBSD 5.X)를 설정하였다면 다음 명령으로 파일시스템을 마운트할 수 있다:

FreeBSD 4.X 는 아래 명령을 사용한다:

```
# vnconfig -e vn0c /tmp/bootable.iso
# mount -t cd9660 /dev/vn0c /mnt
```

FreeBSD 5.X 에서는 다음 명령을 사용한다:

```
# mdconfig -a -t vnode -f /tmp/bootable.iso -u 0
# mount -t cd9660 /dev/md0 /mnt
```

여기서 /mnt 와 /tmp/myboot 가 동일함을 확인할 수 있다.

미세한 조정을 위해 [sysinstalls/mkisofs](#) 에 사용할 수 있는 다양한 옵션이 있다. 특히 ISO 9660 레이아웃을 수정하고 Joliet 와 HFS 디스크를 작성할 수 있다. 더 자세한 사항은 mkisofs(8) 매뉴얼 페이지를 본다.

### 16.6.3 CD 굽기

ATAPI CD 레코더를 가지고 있다면 ISO 이미지를 CD 로 레코드하기 위해 burncd 명령을 이용할 수 있다. burncd 는 기본 시스템의 /usr/sbin/burncd 에 설치되어 있다. 옵션이 몇 개 없으므로 사용법은 매우 단순하다:

```
# burncd -f cddevice data imagefile.iso fixate
```

위 명령은 *imagefile.sio* 를 *cddevice* 에 레코드한다. 기본 장치는 /dev/acd0c 다. 레코드 속도와 레코드 후 CD 꺼내기 그리고 오디오 데이터 작성에 대한 설정 옵션은 burncd(8)을 확인한다.

---

## 16.6.4. cdrecord

ATAPI CD 레코더를 가지고 있지 않다면 CD를 레코드하기 위해 `cdrecord`를 사용해야 된다. `cdrecord`는 기본 시스템에 없어서 `sysutils/cdrtools` 포트나 적당한 패키지로 설치해야 된다. 기본 시스템을 변경하면 이 프로그램의 바이너리에 문제가 발생할 수 있다. 그래서 시스템을 업그레이드할 때 포트를 업그레이드 하거나, **STABLE** 버전을 사용한다면 새로운 버전을 사용할 수 있을 때 포트를 업그레이드 한다.

`cdrecord`는 많은 옵션을 가지고 있지만 기본 사용법은 `burncd`보다 간단하다. ISO 9660 레코드는 다음 명령을 이용한다:

```
# cdrecord dev=device imagefile.iso
```

`cdrecord`를 제대로 이용하는 방법은 사용할 `dev`를 찾을 때 사용한다. 적당한 설정을 찾으려면 다음과 비슷한 결과를 출력하는 `cdrecord`에 `-scanbus` 플래그를 사용한다.

```
# cdrecord -scanbus
Cdrecord 1.9 (i386-unknown-freebsd4.2) Copyright (C) 1995-2000 Jorg Schilling
Using libscg version 'schily-0.1'
scsibus0:
  0,0,0  0) 'SEAGATE ' 'ST39236LW      ' '0004' Disk
  0,1,0  1) 'SEAGATE ' 'ST39173W      ' '5958' Disk
  0,2,0  2) *
  0,3,0  3) 'iomega  ' 'jaz 1GB        ' 'J.86' Removable Disk
  0,4,0  4) 'NEC      ' 'CD-ROM DRIVE:466' '1.26' Removable CD-ROM
0,5,0   5) *
  0,6,0  6) *
  0,7,0  7) *
scsibus1:
  1,0,0 100) *
  1,1,0 101) *
  1,2,0 102) *
  1,3,0 103) *
  1,4,0 104) *
  1,5,0 105) 'YAMAHA  ' 'CRW4260      ' '1.0q' Removable CD-ROM
  1,6,0 106) 'ARTEC   ' 'AM12S        ' '1.06' Scanner
```



---

1,7,0 107) *
--------------

리스트에 있는 장치의 *dev* 값으로 이 리스트가 적당하다. CD 레코더를 설치하고 *dev* 값에 쉼표(,)로 나뉜 3 개의 번호를 사용한다. 여기서 CRW 장치는 1,5,0 이기 때문에 적당한 입력은 *dev=1,5,0* 이다. 이 값을 지정하는 쉬운 방법은 `cdrecord(1)` 매뉴얼 페이지에 있다. 또한 오디오 트랙 작성, 속도 제어와 다른 것에 대한 정보도 찾을 수 있다.

## 16.6.5 오디오 CD 복제

CD 에서 오디오 데이터를 연속된 파일로 추출해서 공 CD 에 레코딩하여 오디오 CD 를 복제할 수 있다. 이 과정은 ATAPI 와 SCSI 드라이브에서 약간 다르다.

### SCSI 드라이브

- ① 오디오를 추출하기 위해 `cdda2wav` 를 사용한다

```
% cdda2wav -v255 -D2,0 -B -Owav
```

- ② `.wav` 파일을 레코딩하기 위해 `cdrecord` 를 사용한다.

```
% cdrecord -v dev=2.0 -dao -useinfo *.wav
```

16.6.4 장에서 설명했듯이 2.0이 적당한 설정이다.

### ATAPI 드라이브

- ① ATAPI CD 드라이브는 각 트랙을 `/dev/acd $d$ tnn` 에 작성한다. *d* 는 드라이브 번호고 *nn* 은 필요하다면 0 을 앞에 붙인(예: 02) 두 개의 10 진수로 작성하는 트랙번호다. 따라서 첫 번째 디스크의 첫 번째 트랙은 `/dev/acd0t01`, 두 번째는 `/dev/acd0t02`, 세 번째는 `/dev/acd0t03` 등으로 변환된다.

`/dev` 에 적당한 파일이 있는지 확인한다.

---

```
# cd /dev
# sh MAKEDEV acd0t99
```

**Note:** FreeBSD 5.0 에서 `devfs(5)`가 `/dev`의 엔트리를 자동으로 생성하고 관리하기 때문에 `MAKEDEV`는 필요 없다.

- ② `dd(1)`로 각 트랙을 추출한다. 파일을 추출할 때 블록 크기도 지정해야 된다.

```
# dd if=/dev/acd0t01 of=track1.cdr bs=2352
# dd if=/dev/acd0t02 of=track2.cdr bs=2352
...
```

- ③ `burncd`로 추출한 파일을 레코드한다. 오디오 파일임을 지정해야 되고 `burncd`는 레코딩이 끝나고 디스크를 닫는다.

```
# burncd -f /dev/acd0 audio track1.cdr track2.cdr ... fixate
```

## 16.6.6 데이터 CD 복제

데이터 CD를 `sysutils/mkisofs`로 생성한 이미지 파일과 기능적으로 동등한 이미지 파일로 복사할 수 있고 이 방법으로 어떤 데이터 CD라도 복제할 수 있다. 여기 주어진 예제의 CDROM 장치는 `acd0`다. 실제 CDROM 장치로 변경한다. FreeBSD 4.X에서 `c`는 전체 파티션 또는 이 경우 CDROM이고 전체 디스크를 표시하기 위해 장치 이름 끝에 붙인다.

```
# dd if=/dev/acd0c of=file.iso bs=2048
```

이제 이미지를 갖게 되었고 위에서 설명한대로 CD에 레코드한다.

## 16.6.7 데이터 CD 사용하기

이제 표준 데이터 CDROM을 만들었기 때문에 마운트해서 데이터를 읽으려고 할 것이다. 기본적으로 `mount(8)`은 파일시스템을 `ufs`로 생각한다. 다음 명령을 사용하면 “Incorrect super block”이라는 메시지가 나타나고 마운트되지 않는다.

---

```
# mount /dev/cd0c /mnt
```

CDROM 은 *UFS* 파일시스템이 아니기 때문에 마운트하려는 시도는 위와 같이 실패한다. `mount(8)`에 파일시스템이 *ISO9660* 이라고 지정하면 모든 것이 정상으로 동작한다. `mount(8)`에 옵션 `-t cd9660` 을 지정하면 된다. 예를 들어 CDROM 장치 `/dev/cd0c` 를 `/mnt` 에 마운트하려면 다음 명령을 사용한다:

```
# mount -t cd9660 /dev/cd0c /mnt
```

CDROM 이 사용하는 인터페이스에 따라 장치 이름이(이 예제에서는 `/dev/cd0c`) 다를 수 있다. 또한 `-t cd9660` 옵션은 `mount_cd9660(8)`를 실행한다. 위의 예제를 짧게 만들 수 있다:

```
# mount_cd9660 /dev/cd0c /mnt
```

이 방법으로 어떤 벤더의 데이터 CDROM 이라도 보통 사용할 수 있다. 그러나 특정 *ISO 9660* 으로 확장된 디스크는 이상하게 수행된다. 예를 들어 *Joliet* 디스크는 모든 파일 이름을 2-byte 유니코드 문자로 저장한다. *FreeBSD* 커널은 아직 유니코드를 지원하지 못하기 때문에 영문자가 아닌 문자는 물음표(?)로 나타난다(*FreeBSD 4.3* 이나 이후 버전을 사용하면 *CD9660* 드라이버는 적당한 유니코드 변환 테이블을 가지고 있다. 몇몇 일반적인 암호화 모듈은 `sysutils/cd9660_unicode` 포트에서 사용할 수 있다).

때때로 CDROM 을 마운트할 때 “Device not configured”를 보게 될 것이다. 이 의미는 보통 트레이에 디스크가 없거나 버스에 드라이브가 없는 것으로 CDROM 드라이브가 인지한다. CDROM 드라이브가 실제로 확인하려면 몇 초 정도가 소요되므로 끈기 있게 기다린다.

종종 SCSI CDROM 은 버스 리셋에 응답할 충분한 시간이 없기 때문에 실패한다. SCSI CDROM 을 가지고 있다면 커널 설정에 다음 옵션을 추가하고 커널을 다시 빌드한다.

```
options SCSI_DELAY=15000
```

이것은 SCSI 버스에게 부팅 시 15 초 동안 기다리게 하여 CDROM 드라이브에게 버스 리셋에 응답할 수 있는 모든 가능한 기회를 주기 위해서다.

---

## 16.6.8 raw 데이터 CD 레코드

ISO 9660 파일시스템을 생성하지 않고 파일을 직접 CD 로 레코드하도록 선택할 수 있다. 어떤 사람들은 백업 목적으로 이렇게 하고 있다. 이것은 표준 CD 레코드보다 더 빨리 실행된다:

```
# burncd -f /dev/acd1c -s 12 data archive.tar.gz fixate
```

데이터를 CD 같은 곳에 저장하려면 데이터를 raw 장치 노드에서 읽어야 한다:

```
# tar xzvf /dev/acd1c
```

일반 CDROM 처럼 이 디스크를 마운트할 수 없다. 이런 CDROM 은 FreeBSD 를 제외한 어떤 운영체제에서도 읽을 수 없다. CD 를 마운트하거나 다른 운영체제와 데이터를 공유하려면 위에서 설명한대로 [sysutils/mkisofs](#) 을 사용해야 된다.

## 16.6.9 ATAPI/CAM 드라이버 사용

이 드라이버는 SCSI 서브시스템을 통해 ATAPI 장치에(CD-ROM, CD-RW, DVD 장치 등) 접근하도록 해서 [sysutils/cdrdao](#) 또는 `cdrecord(1)` 같은 어플리케이션을 사용할 수 있도록 한다.

이 드라이버를 사용하려면 커널 설정파일에 다음 라인을 추가해야 된다:

```
device atapicam
device scbus
device cd
device pass
```

또한 다음 라인도 커널 설정파일에 필요하다:

```
device ata
```

그리고 다시 빌드하고 새로운 커널을 설치한 후 머신을 재 부팅한다. 부팅할 때 레코더는 다음과 같이 나타난다:

---

```
acd0: CD-RW <MATSHITA CD-RW/DVD-ROM UJDA740> at ata1-master PIO4
cd0 at ata1 bus 0 target 0 lun 0
cd0: <MATSHITA CDRW/DVD UJDA740 1.00> Removable CD-ROM SCSI-0 device
cd0: 16.000MB/s transfers
cd0: Attempt to query device size failed: NOT READY, Medium not present - tray closed
```

이 드라이브는 이제 `/dev/cd0` 장치 이름으로 접근할 수 있고, 예를 들어 CD-ROM 을 `/mnt` 에 마운트하려면 다음 명령을 입력한다:

```
# mount -t cd9660 /dev/cd0 /mnt
```

root 에서 레코더의 SCSI 주소를 알기 위해 다음 명령을 실행할 수 있다:

```
# camcontrol devlist
<MATSHITA CDRW/DVD UJDA740 1.00> at scbus1 target 0 lun 0 (pass0,cd0)
```

그래서 `1,0,0`이 `cdrecord(1)`과 다른 SCSI 어플리케이션이 사용하는 주소가 된다.

ATAPI/CAM 과 SCSI 시스템에 대한 더 많은 정보는 `atapicam(4)`와 `cam(4)` 매뉴얼 페이지를 참고한다.

## 16.7 광학 미디어(DVD) 생성과 사용

### 16.7.1 소개

CD 와 비교하여 DVD 는 차세대 광학 미디어 스토리지 기술이다. DVD 는 CD 보다 많은 데이터를 저장하고 요즘 비디오 배포의 표준이다.

물리적으로 레코드 할 수 있는 5 가지 포맷을 레코드하는 DVD 에 정의할 수 있다:

- **DVD-R:** 이것은 처음으로 DVD를 레코드 할 수 있던 포맷이다. DVD-R 표준은 DVD 포럼(<http://www.dvdforum.com/forum.shtml>)에 정의되어 있다. 이 포맷은 한 번만 레코드 한다.

- 
- **DVD-RW**: 이것은 DVD-R 표준의 다시 레코드 할 수 있는(rewriteable) 버전이다. DVD-RW 는 1000 회 정도 다시 레코드 할 수 있다.
  - **DVD-RAM**: 이것도 DVD 포럼이 지원하는 다시 레코드 할 수 있는 포맷이다. DVD-RAM 은 이동할 수 있는 하드 드라이브처럼 보인다. 그러나 이 미디어는 대부분의 DVD-ROM 드라이브와 DVD 비디오 플레이어와 호환되지 않는다; 오직 몇 개의 DVD 레코더만 DVD-RAM 을 지원한다.
  - **DVD+RW**: 이것은 DVD+RW Alliance(<http://www.dvdrw.com/>)가 정의한 다시 레코드 할 수 있는 포맷이다. DVD+RW는 대략 1000 회 정도 다시 레코드 할 수 있다.
  - **DVD+R**: 이 포맷은 다양한 DVD+RW 포맷을 한번만 레코드 한다.

한 장의 DVD 는 실제 4.38GB 또는 4485MB 에(1 kilobyte 는 1024bytes) 달하는 4,700,000,000 바이트를 저장할 수 있다.

**Note:** 물리적인 미디어와 어플리케이션에 따라 차이가 있다. 예를 들어 DVD 비디오 는 레코딩하는 물리적인 미디어에 작성할 수 있는 특정 파일 레이아웃이다: DVD-R, DVD+R, DVD-RW 등. 미디어 종류를 선택하기 전에 레코딩 소프트웨어와 DVD 비디오 플레이어가(표준 플레이어나 컴퓨터의 DVD-ROM) 미디어와 호환되는지 확인해야 된다.

## 16.7.2 설정

프로그램 growisofs(1)은 DVD 레코딩에 사용된다. 이 명령은 **dvd+rw-tools** 유틸리티의 ([sysutils/dvd+rw-tools](#)) 일부분이다. **dvd+rw-tools** 는 모든 DVD 미디어 종류를 지원한다

이들 툴은 장치에 접근하기 위해 SCSI 서브시스템을 사용하므로 커널에 ATAPI/CAM 지원을 추가해야 된다.

또한 ATAPI 장치를 사용하기 위해 DMA 도 활성화해야 된다. 다음 라인을 /boot/loader.conf 파일에 추가하면 된다:

---

```
hw.ata.atapi_dma="1"
```

`dvd+rw-tools` 를 사용하기 전에 여러분의 DVD 레코더와 관련된 정보를 얻기 위해 `dvd+rw-tools` 의 하드웨어 호환 노트를 참고한다.

### 16.7.3 데이터 DVD 굽기

`growisofs(1)` 명령은 파일시스템 레이아웃을 만들어서 DVD 에 작성하는 `mkisofs(8)`을 실행한다. 이 의미는 DVD 를 굽기 전에 데이터 이미지를 만들 필요가 없다는 것이다.

`/path/to/data` 디렉터리의 데이터를 DVD+R 이나 DVD-R 로 굽기 위해 다음 명령을 사용한다:

```
# growisofs -dvd-compat -Z /dev/cd0 -J -R /path/to/data
```

옵션 `-J-R`은 파일시스템을 생성하기 위해(이 예제에서는 Joliet 와 Rock Ridge 로 확장하여 ISO 9660 파일시스템을 생성) `mkisofs(8)`에 적용한다. 더 자세한 내용은 `mkisofs(8)` 매뉴얼 페이지를 참고한다.

옵션 `-Z`는 멀티 세션이던지 아니던지 상관없이 초기 세션을 레코딩 할 때 사용한다. DVD 장치 `/dev/cd0`는 여러분의 설정에 따라 변경해야 된다. `-dvd-compat` 매개변수는 디스크를 달아서 추가적으로 레코딩 할 수 없다. 대신 이것은 더 많은 DVD-ROM 드라이브와 미디어 호환성을 제공한다.

이미 생성한 이미지를 레코드 할 수도 있다. 예를 들면 `imagefile.iso` 이미지를 굽기 위해 다음 명령을 실행한다:

```
# growisofs -dvd-compat -Z /dev/cd0=imagefile.iso
```

쓰기 속도는 미디어에 따라 자동으로 감지되어 설정된다. 쓰기 속도를 강제로 조정하려면 `-speed=` 매개변수를 사용한다. 더 많은 정보는 `growisofs(1)` 매뉴얼 페이지를 읽는다.

### 16.7.4 DVD 비디오 굽기

---

DVD 비디오는 ISO 9660 과 micro-UDF(M-UDF) 기술서를 기반으로 하는 특정 파일 레이아웃이다. 또한 DVD 비디오에는 특정 데이터 구조 계층이 있다. 이것이 DVD 를 작성하기 위해 [sysutils/dvdauthor](#) 같은 특정 프로그램을 사용하는 이유다.

DVD 비디오 파일시스템의 이미지를 가지고 있다면 이전 섹션의 예제에서 다른 이미지처럼 굽는다. 예를 들어 DVD 이미지를 만들었고 그 결과가 `/path/to/video` 디렉터리에 있다면 DVD 비디오로 굽기 위해 다음 명령을 사용한다:

```
# growisofs -Z /dev/cd0 -dvd-video /path/to/video
```

`-dvd-video` 옵션은 `mkisofs(8)`에 적용되어 DVD 비디오 파일시스템 레이아웃을 생성한다. `-dvd-video` 옵션은 `growisofs(1)` `-dvd-compat` 옵션과 동일하다.

## 16.7.5 DVD+RW 사용하기

CD-RW 와 달리 DVD+RW 는 처음 사용하기 전에 포맷을 해야 된다. `growisofs(1)` 프로그램이 자동으로 적절하게 포맷하기 때문에 권장하는 방법이다. 그러나 DVD+RW 를 포맷하기 위해 `dvd+rw-format` 명령을 사용할 수 있다:

```
# dvd+rw-format /dev/cd0
```

이 옵션을 딱 한번만 실행하면 되기 때문에 최초의 DVD+RW 미디어에만 포맷이 필요하다. 그리고 이전 섹션에서 보여준 것처럼 DVD+RW 를 굽는다.

새로운 데이터를 DVD+RW 에 굽기 위해 미디어의 내용을 삭제할 필요 없이 다음과 같이 이전 레코딩을 덮어쓰면 된다:

```
# growisofs -Z /dev/cd0 -J -R /path/to/newdata
```

DVD+RW 포맷은 이전 레코딩에 데이터를 쉽게 추가할 수 있다. 멀티 세션으로 레코딩하지 않고 기존 데이터에 새로운 세션을 합한다. `growisofs(1)`은 미디어에 있던 ISO 9660 파일시스템을 확장한다.

예를 들어 이전 DVD+RW 에 데이터를 추가하려면 다음 명령을 사용해야 된다:



---

```
# growisofs -M /dev/cd0 -J -R /path/to/nextdata
```

초기 세션을 레코딩 할 때 사용했던 mkisofs(8) 옵션을 다음에 레코딩 할 때도 사용해야 된다.

**Note:** DVD-ROM 미디어가 더 많은 드라이브와 호환되기를 원한다면 `-dvd-compat` 옵션을 사용할 수 있다. DVD+RW 에서는 이 옵션이 있어도 데이터를 추가할 수 있다.

특별한 이유로 미디어의 데이터를 삭제하려면 다음 명령을 사용한다:

```
# growisofs -Z /dev/cd0=/dev/zero
```

## 16.7.6 DVD-RW 사용

DVD-RW 는 두 가지 디스크 포맷을 허용한다: 연속적으로 증가하는 것과 덮어쓰기를 제한한 것. 기본적으로 DVD-RW 디스크는 연속적인(추가할 수 있는) 포맷이다.

최초의 DVD-RW 는 포맷 없이 바로 사용할 수 있지만 연속 포맷으로 사용한 DVD-RW 는 새로운 세션을 작성하기 전에 데이터를 삭제해야 된다.

연속 모드에서 DVD-RW 데이터를 삭제하기 위해 다음 명령을 실행한다:

```
# dvd+rw-format -blank=full /dev/cd0
```

**Note:** 1x 미디어에서 전체를 삭제하는데(`-blank=full`) 대략 한 시간 정도 걸린다. DVD-RW 가 Disk-At-Once(DAO) 모드에서 작성됐다면 `-blank` 옵션으로 빠르게 삭제할 수 있다. DAO 모드에서 DVD-RW 를 구우려면 다음 명령을 사용한다:

```
# growisofs -use-the-force-luke=dao -Z /dev/cd0=imagefile.iso
```

growisofs(1)이 최소(빠른 삭제) 미디어로 감지하여 DAO 에서 작성하기 때문에 `-use-the-force-luke=dao` 옵션은 필요 없다.

이 포맷이 기본값인 연속적으로 증가하는 것보다 더 유연하기 때문에 어떤 DVD-

---

RW 든 덮어 쓰기 제한 모드로 사용하는 것이 좋다.

연속 DVD-RW 에 데이터를 작성하려면 다른 DVD 포맷과 같은 명령을 사용한다:

```
# growisofs -Z /dev/cd0 -J -R /path/to/data
```

이전에 레코딩 한 데이터에 다른 데이터를 추가하려면 `growisofs(1)`에 `-M` 옵션을 사용한다. 그러나 연속 증가모드에서 DVD-RW 에 데이터를 추가한다면 새로운 세션이 디스크에 생성되어 멀티 세션 디스크가 된다.

덮어쓰기 제한 포맷에서 DVD-RW 는 새로운 세션을 생성하기 전에 데이터를 삭제할 필요 없이 DVD+RW 의 경우와 비슷하게 `-Z` 옵션으로 디스크를 덮어쓴다. `-M` 옵션을 DVD+RW 에 사용한 것처럼 디스크에 있는 이전 ISO 9660 파일시스템에 추가할 수 있다. 결과는 단일 세션 DVD 가 된다.

DVD-RW 를 덮어쓰기 제한으로 포맷하려면 다음 명령을 사용한다:

```
# dvd+rw-format /dev/cd0
```

이전의 연속 포맷으로 변환하려면 다음 명령을 사용한다:

```
# dvd+rw-format -blank=full /dev/cd0
```

### 16.7.7 멀티 세션

매우 적은 DVD-ROM 과 DVD 비디오 플레이어만 멀티세션 DVD 를 지원한다. 대부분 이들은 첫 번째 세션만 읽는다. 연속 포맷에서 DVD+R, DVD-R 과 DVD-RW 는 멀티 세션을 허용한다. 덮어쓰기 제한 포맷에서는 DVD+RW 와 DVD-RW 에 멀티 세션이라는 개념이 없다.

연속 포맷의 DVD+R, DVD-R 또는 DVD-RW 에서 초기화 세션 이후에 다음 명령을 사용하면 디스크에 새로운 세션을 추가한다:

```
# growisofs -M /dev/cd0 -J -R /path/to/nextdata
```

덮어쓰기 제한 모드에서 DVD+RW 또는 DVD-RW 에 이 명령 라인을 사용하면 이전 데이터

---

에 새로운 세션을 합병하여 데이터를 추가한다. 이 결과는 단일 세션 디스크가 된다. 이것이 이들 미디어를 작성한 후 데이터를 추가하는 방법이다.

**Note:** 미디어의 어떤 공간이 각 세션의 처음과 끝에 사용된다. 따라서 미디어 공간을 최적화하기 위해 대형 데이터를 단일 세션으로 추가한다. DVD-R 에서 세션 개수는 대략 200 개 그리고 DVD+R 에서는 154 개 제한이 있다.

## 16.7.8 더 많은 정보

DVD 에 대한 더 많은 정보는 `dvd+rw-mediainfo` 를 확인한다.

`dvd+rw-tools` 에 대한 더 많은 정보는 `growisofs(1)` 매뉴얼 페이지와 `dvd+rw-tools` 웹 사이트 그리고 `cdwrite` 메일링 리스트 모음에서 찾을 수 있다.

**Note:** 레코딩 결과나 미디어 문제에 대한 `dvd+rw-mediainfo` 출력은 어떠한 문제 보고서에도 필수다. 이 결과 없이 여러분을 돕는 것은 거의 불가능하다.

## 16.8 플로피 디스크 생성과 사용

예를 들어 누군가 이동할 수 있는 스토리지 미디어가 없을 때 다른 컴퓨터로 작은 양의 데이터를 전송해야 된다면 플로피 디스크에 데이터를 저장하는 것도 유용하다.

이 섹션은 FreeBSD 에서 플로피를 어떻게 사용하는지 설명한다. 여기서는 3.5 인치 DOS 플로피 포맷을 사용하는 방법에 대해 주로 다루지만 개념은 다른 플로피 디스크 포맷과 비슷하다.

### 16.8.1 플로피 포맷

#### 16.8.1.1 장치

플로피도 다른 장치처럼 `/dev` 엔트리로 접근한다. FreeBSD 4.X 와 이전 릴리즈에서 `raw` 플

---

로피 디스크를 사용하려면 `/dev/fdN`이나 `/dev/fdNX`를 사용한다. *N*은 보통 0인 드라이브 번호고 *X*는 문자를 표시한다.

5.0과 새로운 릴리즈에는 간단히 `/dev/fdN`을 사용한다.

## 5.0 이전 릴리즈와 5.0과 이후 릴리즈의 디스크 크기 설정

### ① 4.X와 이전 릴리즈에서 디스크 크기

`/dev/fdN.size` 장치가 있다. *size*는 킬로 바이트의 플로피 디스크 크기다. 이런 엔트리는 디스크 크기를 결정하기 위해 `low` 레벨로 포맷할 때 사용된다. 1440KB는 다음 예제에서 사용할 크기다.

가끔 `/dev` 아래의 엔트리를 생성해야 된다. 생성하려면 다음 명령을 실행한다:

```
# cd /dev && ./MAKEDEV "fd*"
```

### ② 5.0 또는 새로운 릴리즈에서 디스크 크기

5.0에서는 `devfs(5)`가 `/dev`에서 자동으로 장치 노드를 관리하기 때문에 `MAKEDEV`는 필요 없다.

원하는 디스크 크기는 `fdformat(1)`의 `-f` 플래그로 설정한다. 지원되는 크기는 `fdcontrol(8)`에 나열되어 있지만 가장 정확히 작동하는 1440kB를 권장한다.

## 16.8.1.2 포맷

플로피 디스크는 사용하기 전에 `low` 레벨 포맷을 해야 된다. 보통 벤더에서 미리 포맷을 하지만 포맷은 미디어의 무결성을 체크하는 좋은 방법이다. 큰 디스크 크기를 지정하여 강제로 포맷할 수 있지만 대부분의 디스크가 1440kB 용으로 디자인되어있다.

플로피를 `low`-레벨 포맷하려면 `fdformat(1)`를 사용해야 된다. 이 유틸리티는 장치 이름을 인자로 인식한다.

에러 메시지가 발생했다면 적어두고 이로서 디스크가 좋은지 나쁜지 결정하는데 도움이 된다.

---

## 5.0 이전 릴리즈와 5.0 과 이후 릴리즈의 디스크 포맷하기

### ① 4.X 와 이전 릴리즈에서 포맷

플로피를 포맷 하려면 `/dev/fdN.size` 를 사용한다. 드라이브에 3.5 인치 디스크를 넣고 다음 명령을 실행한다:

```
# /usr/sbin/fdformat /dev/fd0.1440
```

### ② 5.0 과 새로운 릴리즈에서 포맷

플로피를 포맷하기 위해 `/dev/fdN` 장치를 사용한다. 3.5 인치 플로피 디스크를 드라이브에 넣고 다음 명령을 실행한다:

```
# /usr/sbin/fdformat -f 1440 /dev/fd0
```

## 16.8.2 디스크 라벨

디스크를 low-레벨 포맷하고 디스크 라벨을 할당해야 된다. 이 디스크 라벨은 이후에 삭제되지만 시스템이 디스크 크기와 구조를 결정하기 위해 필요하다.

새로운 디스크 라벨이 전체 디스크에 할당되어 플로피 디스크의 구조에 대해 필요한 정보를 얻게 된다. 디스크 라벨의 구조 값은 `/etc/disktab` 에 나열되어 있다.

이제 다음과 같이 `disklabel(8)`을 실행한다:

```
# /sbin/disklabel -B -r -w /dev/fd0 fd1440
```

## 16.8.3 파일시스템

이제 플로피에 high-레벨 포맷을 하기 위한 준비가 되었다. **FreeBSD** 가 디스크를 읽고 쓸 수 있도록 새로운 파일시스템을 생성한다. 새로운 파일시스템을 생성한 후 디스크 라벨이 삭제되기 때문에 디스크를 다시 포맷하려면 디스크 라벨을 다시 생성해야 된다.

플로피의 파일시스템은 **UFS** 나 **FAT** 로 만들 수 있다. 보통 **FAT** 가 플로피에 적절하다.

---

플로피에 새로운 파일시스템을 작성하려면 다음 명령을 실행한다:

```
# /sbin/newfs_msdos /dev/fd0
```

이제 디스크를 사용할 준비가 되었다.

## 16.8.4 플로피 사용

플로피를 사용하려면 `mount_msdos(8)`(4.X와 이전 릴리즈에서) 또는 `mount_msdosfs(8)`로 (5.0과 새로운 릴리즈에서) 마운트한다. 포트 컬렉션에서 [emulators/mtools](#)도 사용할 수 있다.

## 16.9 데이터 테이프 생성과 사용

주로 사용하는 테이프 미디어는 4mm, 8mm, QIC, 미니 카트리지와 DLT다.

### 16.9.1 4mm (DDS: 디지털 데이터 스토리지)

4mm 테이프는 워크스테이션의 백업 미디어에서 QIC를 대체하고 있다. 이런 경향은 QIC 드라이브의 생산을 주도하는 Conner purchased Archive사가 QIC 드라이브 생산을 중단했을 때 더욱 가속화되었다. 4mm 드라이브는 작고 조용하지만 안정성에 대한 평판이 좋지 않아 8mm 드라이브를 즐겨 사용한다. 이 카트리는 8mm 카트리지보다 더 싸고 작다(3 x 2 x 0.5 인치, 76 x 51 x 12 mm). 8mm와 비슷한 4mm는 둘 다 나선형 스캔 방식을 사용하기 때문에 헤드 수명이 비교적 짧다.

IMATION DDS 3 4mm DAT TAPE



---

이런 드라이브에서 데이터 처리량은 ~150 kB/s 에서 최고 500kB/s 에 이른다. 데이터 저장 용량은 1.3GB 에서 2.0GB 다. 이들 드라이브 대부분에서 사용할 수 있는 하드웨어 압축은 대략 두 배 용량이다. 멀티 드라이브 테이프 라이브러리 유닛은 케비넷 하나에 드라이브 6 개를 가지고 자동으로 테이프를 교체한다. 라이브러리 저장 용량은 40GB 에 달한다.

DDS-3 표준은 이제 12GB 이상의(또는 압축해서 24GB) 용량을 지원한다.

테이프는 2,000 회가 지나거나 100 번의 전체 백업에 사용하면 사용할 수 없다.

## 16.9.2 8mm (Exabyte)

8mm 테이프는 보편적인 SCSI 테이프 드라이브로 테이프를 교체하는 방식의 최고의 선택이 되고 있다. 거의 모든 사이트는 2GB 8mm 테이프 드라이브를 가지고 있다. 8mm 드라이브는 안정적이고 상당히 편리하지만 카트리지가 비싸고 작다(4.8 x 3.3 x 0.6 인치; 122 x 84x 15mm). 8mm 테이프의 단점은 헤드를 스치는 테이프의 많은 움직임으로 헤드와 테이프의 수명이 비교적 짧다.

### Verbatim 8mm DAT TAPE



데이터를 처리하는 범위는 ~250kB/s 에서 ~500 kB/s 다. 데이터 크기는 300MB 에서 7GB 에 달한다. 대부분의 이들 드라이브에서 사용할 수 있는 하드웨어 압축은 대략 두 배 용량이다. 이들 드라이브는 싱글 유닛 또는 드라이브 6 개로 된 멀티 드라이브 테이프 라이브러리와 120 개의 테이프를 싱글 케비넷에 사용할 수 있다. 테이프는 유닛에 의해 자동으로 교체되고 라이브러리 용량은 840+ GB 에 달한다.

---

대형 "맘모스" 모델은 하나의 테잎에(압축해서 24GB) 12GB 를 지원하고 가격은 기존 테잎 드라이브의 대략 두 배 정도다.

데이터는 나선 스캔으로 테잎에 저장되고 헤드는 미디어의(대략 6도 정도) 각진 부분에 위치한다. 테잎은 대략 헤드를 잡고 있는 스펀의 270도 각도로 감겨 있다. 스펀은 테잎이 스펀 위를 미끄러지는 동안 강긴다. 그 결과 데이터의 밀도는 높고 한쪽 끝에서 다른 쪽으로 테잎을 교차하여 뾰뾰하게 트랙에 저장된다.

### 16.9.3 QIC

QIC-150 테잎과 드라이브는 아주 보편적인 테잎 드라이브와 미디어일 것이다. QIC 테잎 드라이브는 가장 싼 가격의 백업 드라이브 시리즈다. 단점은 미디어의 가격이다. QIC 테잎은 8mm 나 4mm 테잎에 비해 비싸고 GB 데이터 스토리지당 5 배 정도 비싸다. 그러나 6 개 정도의 테잎으로 만족한다면 QIC 가 정확한 선택일 것이다. QIC 는 아주 보편적인 테잎 드라이브기 때문에 모든 사이트는 QIC 드라이브를 다른 장비만큼 가지고 있다. QIC 는 물리적으로 테잎과 비슷한(종종 똑 같은) 큰 밀도를 가진다. QIC 드라이브는 조용하지 않고 데이터를 레코드하기 전과 읽기, 쓰기 또는 검색할 때는 시끄럽다. QIC 테잎 치수는 6 x 4 x 0.7 인치; 15.2 x 10.2 x 1.7 mm 다. 테잎 라이브러리와 교환기는 사용할 수 없다.

#### Verbatim QIC TAPE



데이터 처리량은 ~150kB/s 에서 ~500 kB/s 이고 용량은 40MB 에서 15GB 다. 하드웨어 압축은 새로운 QIC 드라이브에서 사용할 수 있으며 점점 사용하는 사이트가 적어지고 있다; 이들은 DAT 드라이브로 교체되고 있다.

데이터는 테잎의 트랙에 저장되고 트랙은 테잎 미디어의 긴 축을 따라 끝에서 끝으로 동작한다. 트랙은 테잎의 용량에 따라 넓이가 다양하다. 대부분의 새로운 드라이브는 예전 것과 읽기(종종 쓰기도) 호환을 제공한다. QIC 는 데이터 보존에 대한(이 메커니즘은 나선 스캔 드라이브 보다 더 단순하고 강하다) 좋은 평판을 가지고 있다.



---

테잎은 5,000 번 백업 이후에 폐기된다.

## 16.9.4 DLT

DLT 는 여기에 나열된 모든 드라이브 중 가장 빠른 데이터 전송율을 가지고 있다.

1/2"(12.5mm) 테잎은 싱글 스펴 카트리지에(4 x 4 x 1 인치; 100 x 100 x 25mm) 포함되어 있다. 카트리는 카트리지 입구에 흔들리는 뚜껑을 하나 가지고 있다. 이 드라이브 메커니즘은 테잎 리더를 추출하기 위해 이 뚜껑을 연다. 테잎 리더는 테잎을 끌어오기 위해 사용하는 타원형의 구멍을 가지고 있으며 스펴을 팽팽히 당겨서 테잎 드라이브로 들어간다. 여기에 나열된(9 트랙 테잎 만 제외하고) 모든 다른 테잎 카트리는 테잎 카트리지 내부에서 스펴을 밀고 당긴다.

### MAXELL 40GB DLT TAPE



데이터 전송량은 대략 1.5MB/s 이어서 4mm, 8mm 나 QIC 테잎 드라이브의 3 배의 전송량이 된다. 데이터 저장 용량은 싱글 드라이브에서 10GB 에서 20GB 에 달한다. 드라이브는 멀티 테잎 교환기와 멀티 테잎 라이브러리는 5 에서 900 개의 테잎을 1 에서 20 개의 드라이브에 사용할 수 있어서 50GB 에서 9TB 의 저장 용량을 제공한다.

DLT 테잎 IV 포맷으로 압축하면 70GB 이상의 용량을 지원한다.

데이터는 테잎의 트랙에 병렬로 직접(QIC 테잎처럼) 저장되어 한번에 두 개의 트랙에 레코드 되고 읽기/쓰기 헤드의 수명은 비교적 긴 편이다; 테잎의 움직임이 정지되면 헤드와 테잎 사이의 움직임도 없다.

---

## 16.9.5 AIT

AIT 는 Sony 의 새로운 포맷이고 테잎당 50GB 까지(압축해서) 저장한다. 테잎은 내용 색인을 가지고 있는 메모리 칩을 포함한다. 이 색인으로 테잎 드라이브가 파일의 위치를 빨리 찾아서 다른 테잎에서 파일 검색에 필요한 몇 분 정도를 단축할 수 있다.

### SONY AIT TAPE



SAMS:Alexandria 같은 소프트웨어는 40 개 또는 더 많은 AIT 테잎 라이브러리를 관리할 수 있고 화면에 내용을 출력하기 위해 테잎의 메모리 칩과 직접 통신한다. 그리고 어떤 테잎에 어떤 파일이 백업되었는지 확인하여 정확한 테잎 위치를 찾아서 로드한 후 테잎에서 데이터를 복구할 수 있다.

## 16.9.6 새로운 테잎을 처음으로 사용하기

완벽한 공 테잎에 처음으로 읽기 또는 쓰기를 하면 제대로 동작하지 않는다. 콘솔 메시지는 다음과 비슷할 것이다:

```
sa0(ncr1:4:0): NOT READY asc:4,1
sa0(ncr1:4:0): Logical unit is in process of becoming ready
```

테잎은 체크 블록을(블록 번호 0) 가지고 있지 않다. 모든 QIC 테잎 드라이브에 QIC-525 표준을 채용하여 테잎에 체크 블록을 작성한다. 여기 두 가지 방법이 있다:

- `mt fsf 1` 명령으로 테잎의 체크 블록을 작성한다.

- 
- 테잎을 꺼내기 위해 앞 패널의 버튼을 사용한다.
  - 다시 테잎을 넣고 dump 데이터를 테잎에 받는다.
  - dump 는 “DUMP: End of tape detected” 리포트를 보여주고 콘솔에 “HARDWARE FAILURE info:280 asc:80,96” 보여준다.
  - mt rewind 명령으로 테잎을 되감는다.
  - 다른 테잎들도 똑같이 실행한다.

## 16.10 플로피에 백업하기

### 16.10.1 데이터를 백업하기 위해 플로피를 사용할 수 있는가?

플로피 디스크는 다음과 같은 이유로 백업용으로 적당하지 않다:

- 미디어의 안정성이 떨어지고 특히 장시간 보관에 취약하다.
- 백업과 복구가 매우 느리다.
- 용량의 한계가 크다.

그러나 데이터를 백업할 만한 다른 방법이 없다면 플로피 디스크를 이용한 백업이 하지 않는 것보다 좋다.

플로피를 사용한다면 고품질의 플로피를 사용해야 된다. 몇 년 동안 사무실에 방치하던 플로피는 좋은 선택이 아니다. 이상적인 방법은 평판이 좋은 회사의 새 제품을 이용하는 것이 좋다.

---

## 16.10.2 플로피에 데이터를 어떻게 백업하는가?

플로피에 백업하는 가장 좋은 방법은 여러 플로피에 나눠서 백업하는 `-M` 옵션으로(멀티 볼륨) `tar(1)`를 사용하는 것이다.

현재 디렉터리와 서브디렉터리 전체를 백업하려면 다음 명령을 사용한다(root 에서):

```
# tar Mcvf /dev/fd0 *
```

첫 번째 플로피가 꽂 차면 `tar(1)`는 다음 볼륨을(왜냐하면 `tar(1)`은 미디어와 독립적이기 때문에 이것은 볼륨을 의미한다; 이 문맥에서는 플로피 디스크를 의미한다) 요청하는 프롬프트를 보여준다.

```
Prepare volume #2 for /dev/fd0 and hit return:
```

지정한 모든 파일을 저장할 때까지 이 메시지는 계속된다(볼륨 번호가 증가하면서).

## 16.10.3 압축해서 백업할 수 있는가?

불행히 `tar(1)`는 멀티 볼륨을 저장할 때 `-z` 옵션을 사용하지 못한다. 물론 모든 파일을 `gzip(1)`으로 압축하고 이 파일을 `tar(1)`로 플로피에 저장해서 `gunzip(1)`으로 압축을 풀 수 있다.

## 16.10.4 백업한 내용은 어떻게 복구하는가?

전체 데이터를 복구하려면 다음 명령을 사용한다:

```
# tar Mxvf /dev/fd0
```

지정한 파일만 복구하는 두 가지 방법이 있고 첫 번째는 첫 번째 플로피에서 시작할 수 있다:

```
# tar Mxvf /dev/fd0 filename
```

---

유틸리티 tar(1)는 필요한 파일을 찾을 때까지 다음 플로피를 넣으라는 프롬프트를 보여준다.

다른 방법으로 어떤 플로피의 파일인지 알고 있다면 간단히 플로피를 넣고 위와 같은 명령을 사용한다. 플로피의 첫 번째 파일이 이전 플로피에서 계속된다면 파일을 요청하지 않았어도 tar(1)는 복구하지 못한다는 메시지를 보여준다.

## 16.11 백업 기본

주요 백업 프로그램 3 가지는 dump(8), tar(1)와 cpio(1)다

### 16.11.1 덤프와 복구

전통적인 유닉스 백업 프로그램은 dump 와 restore 다. 이 프로그램들은 드라이브에서 디스크 블록과 파일 시스템이 생성한 링크 및 디렉터리를 모은다. dump 는 전체 파일시스템을 장치에 백업한다. 파일시스템의 일부분이나 하나 이상의 새로운 디렉터리 트리(Incremental Backup)만 백업할 수 없다. dump 는 파일과 디렉터리를 테잎에 넣을 수 없고 대신 파일과 디렉터리를 포함한 raw 데이터 블록을 넣을 수 있다.

**Note:** root 디렉터리에서 dump 를 사용한다면 /home, /usr 이나 다른 디렉터리는 백업되지 않는다. 왜냐하면 이들 디렉터리는 보통 다른 파일시스템에 마운트되거나 심볼릭으로 링크되기 때문이다.

dump 는 이전 AT&T 유닉스 버전 6 의(1975 년경) 특징을 가지고 있다. 기본 매개변수는 9 트랙 테잎에(6250 bpi) 적당하고 요즘의(62,182 ftpi 이상의) 고밀도 미디어에서는 사용할 수 없다. 이런 기본적인 제한은 현대 테잎 드라이브의 용량을 사용하기 위해 명령어 라인에서 무시해야 된다.

다른 컴퓨터의 테잎 드라이브에 rdump 와 rrestore 로 네트워크를 통해 데이터를 백업하는 것도 가능하다. 두 가지 프로그램은 원격 테잎 드라이브를 사용하기 위해 rcmd 와 ruserok 를 사용한다. 그래서 유저가 원격 머신의 .rhosts 파일에 나열되어 있어야 백업을 수행할 수 있다. rdump 와 rrestore 인자는 원격 컴퓨터를 사용하기에 적절해야 된다. FreeBSD 컴퓨터에서 rdumping 으로 Exabyte 테잎 드라이브가 연결된 Sun 의 komodo 에 사용하려면 다음 명령을 입력한다:

---

```
# /sbin/rdump 0dsbfu 54000 13000 126 komodo:/dev/nsa8 /dev/da0a 2>&1
```

주의: .rhosts 인증을 허가하는 것은 보안과 관련이 있다.

더욱 안전한 ssh 로 dump 와 restore 를 사용할 수 있다.

#### 예제 16-1. ssh 를 통한 dump 사용

```
# /sbin/dump -0uan -f - /usr | gzip -2 | ssh1 -c blowfish ₩
targetuser@targetmachine.example.com dd of=/mybigfiles/dump-usr-l0.gz
```

## 16.11.2 tar

tar(1)도 AT&T 유닉스 버전 6 부터(1975 년경) 사용되었다. Tar 명령은 파일시스템을 다룬다; 파일과 디렉터리를 테잎에 저장한다. tar 는 cpio(1)에 사용할 수 있는 완벽한 옵션을 지원하지 않지만 cpio 가 사용하는 일반적이지 않은 파이프라인 명령은 필요 없다.

tar 버전 대부분은 네트워크 백업을 지원하지 않는다. FreeBSD 가 사용할 수 있는 GNU 버전의 tar 는 rdump 와 같은 구문으로 원격 장치를 지원한다. tar 로 Exabyte 테잎 드라이브가 연결된 Sun komodo 사용하려면 다음 명령을 사용한다:

```
# /usr/bin/tar cf komodo:/dev/nsa8 . 2>&1
```

원격 장치를 지원하지 않는 버전에서 원격 테잎 드라이브에 데이터를 보내기 위해 파이프라인과 rsh 를 사용할 수 있다.

```
# tar cf - . | rsh hostname dd of=tape-device obs=20b
```

네트워크 백업의 보안이 걱정된다면 rsh 대신 ssh 를 사용한다.

## 16.11.3 cpio

cpio(1)는 마그네틱 미디어에 맞는 최초의 유닉스 파일 교환 테잎 프로그램이다. cpio 는 파일 교환 수행, 수많은 기록 포맷을 지원하고 데이터를 다른 프로그램에 보내는 파이프 옵션

---

을 가지고 있다. 마지막 기능 때문에 `cpio`가 설치할 때 사용하는 최고의 미디어가 되었다. `cpio`는 디렉터리 트리와 파일 리스트가 어떻게 동작하는지 모르기 때문에 표준 입력을 지원 해야 된다.

`cpio`는 네트워크 백업을 지원하지 않아서 원격 테잎 드라이브에 데이터를 보내기 위해 파이프 라인과 `rsh`를 사용할 수 있다.

```
# for f in directory_list; do
find $f >> backup.list
done
# cpio -v -o --format=newc < backup.list | ssh user@host "cat > backup_device"
```

`directory_list`는 백업하려는 디렉터리 리스트며 `user@host`는 백업을 수행하는 유저/호스트 이름 그리고 `backup_device`는 백업할 위치다(예: `/dev/nsa0`).

## 16.11.4 pax

`pax(1)`는 `tar`와 `cpio`의 IEEE/POSIX 버전이다. 세월이 흐르고 `tar`와 `cpio`의 다양한 버전들이 약간씩 호환되지 않았다. 그래서 완벽한 표준을 제시하여 POSIX는 새로운 기록 유틸리티를 만들었다. `pax`는 다양한 `cpio`와 `tar` 포맷을 읽고 쓸 수 있으며 자신만의 새로운 포맷을 추가했다. 이 명령어 세트는 `tar`보다 `cpio`와 비슷하다.

## 16.11.5 Amanda

**Amanda**는(발전된 메릴랜드 네트워크 디스크 기록자) 한가지 프로그램 이라기보다 클라이언트/서버 백업 시스템이다. **Amanda** 서버는 **Amanda** 클라이언트로 **Amanda** 서버와 네트워크로 연결된 수많은 컴퓨터의 데이터를 싱글 테잎 드라이브에 저장한다. 사이트에서 대형 디스크와 발생하는 일반적인 문제는 테잎에 데이터를 직접 백업할 시간이 태스크에 할당된 시간의 양을 초과하는 것이다. **Amanda**는 이 문제를 해결하였다. **Amanda**는 같은 시간에 여러 파일시스템을 백업하기 위해 "holding disk"를 사용할 수 있다. 그리고 **Amanda**는 "기록 세트"를 생성한다: 테잎 그룹이 **Amanda**의 설정 파일에 나열된 모든 파일시스템의 Full 백업에 일정 시간을 사용한다. "기록 세트"도 밤 마다 모든 파일시스템의 incremental(또는 differential) 백업을 수행한다. 피해가 발생한 파일시스템을 복구하려면 가장 최근의 Full 백업과 incremental 백업이 필요하다.

---

설정 파일은 백업과 **Amanda** 가 만들어내는 네트워크 트래픽을 알맞게 조정한다. 또한 데이터를 테잎에 저장하기 위해 위의 백업 프로그램을 사용한다. **Amanda** 는 기본적으로 설치되지 않지만 포트나 패키지로 사용할 수 있다.

## 16.11.6 어떤 백업 프로그램이 최고인가?

dump(8)가 사용되던 시기에 Elizabeth D.Zwicky 는 여기서 설명하는 모든 백업 프로그램을 테스트하였다. 모든 데이터와 유닉스 파일시스템의 모든 속성을 보존하는 완벽한 프로그램은 dump 다. Elizabeth 는 다양하고 거대한 파일시스템으로 특별한 상황을 만들고(일반적인 것도 포함한) 이들 파일시스템의 백업과 복구로 각 프로그램을 테스트했다. 포함된 특성은: 문제가 있는 파일, 파일 이름을 재미있는 문자로 만든 파일, 읽을 수 없고 쓰기를 못하는 파일, 장치, 백업하는 동안 변경된 파일크기, 백업하는 동안 생성/삭제된 파일 등이다. 그녀는 1991 년 9 월에 LISA V 에 결과를 올렸다. 테스트 백업과 백업프로그램을 확인해 본다 (<http://berdmann.dyndns.org/zwicky/testdump.doc.html>).

## 16.11.8. 비상시 복구 절차

### 16.11.8.1 재난이 발생하기 전에

발생할 만한 재난에 대비하기 위한 4 가지 단계가 있다.

첫째, 각 디스크(예: disklabel da0 | lpr)의 라벨, 파일시스템 테이블과(/etc/fstab) 모든 부트 메시지를 각각 두 개씩 출력해둔다.

둘째, 모든 장치를 가지고 있는 부팅 플로피를(boot.flp 와 fixit.flp) 준비한다. 체크하는 가장 쉬운 방법은 플로피 드라이브에 플로피를 넣고 머신을 재 부팅해서 부트 메시지를 체크한다. 모든 장치가 나열되고 동작한다면 단계 3 을 건너뛰어도 된다.

그렇지 않으면 모든 디스크를 마운트하고 테잎 장치를 사용할 수 있는 커널을 가진 두 개의 부팅 플로피를 만들어야 한다. 이들 플로피는 다음과 같은 명령도 가지고 있어야 한다: fdisk, disklabel, newfs, mount 와 사용하려는 백업 프로그램. 이들 프로그램은 정적으로 링크되어 있어야 한다. dump 를 사용한다면 플로피는 restore 도 가지고 있어야 된다.

셋째, 일반적인 백업 테잎을 만든다. 마지막 백업을 수행한 후 변경된 것은 복구할 수 없을



---

것이다. 백업 테잎에 쓰기 방지 한다.

넷째, 플로피와(boot.flp 와 fixit.flp 또는 단계 2 에서 만든 두 개의 사용자 부팅 플로피 중 하나) 백업 테잎을 테스트한다. 그리고 복구 절차를 적어 둔다. 부팅 가능한 플로피, 출력한 정보와 백업 테잎을 이들 노트와 같이 보관한다. 복구할 때 너무 긴장하여 백업 테잎을 망가뜨리는 것을 적어둔 노트로 방지할 수 있다(어떻게? tar xvf /dev/sa0 대신 순간적으로 tar cvf /dev/sa0 로 백업 테잎을 덮어쓰기 할 것이다).

보안 평가를 추가하려면 부트 플로피와 백업 테이프를 만들어서 원격지에 따로 저장한다. 원격지는 같은 사무실 빌딩이 아니다. 세계 무역 센터의 수많은 상업적 회사들은 비싼 값에 이런 사항을 확인하였다. 원격지는 물리적으로 컴퓨터와 디스크 드라이브가 있는 곳에서 일정 거리 이상을 두어야 한다.

#### 예제 16-3. 부팅 플로피를 만드는 스크립트

```
#!/bin/sh
#
# create a restore floppy
#
# format the floppy
#
PATH=/bin:/sbin:/usr/sbin:/usr/bin

fdformat -q fd0
if [ $? -ne 0 ]
then
    echo "Bad floppy, please use a new one"
    exit 1
fi

# place boot blocks on the floppy
#
disklabel -w -B /dev/fd0c fd1440

#
# newfs the one and only partition
#
```

---

```
newfs -t 2 -u 18 -l 1 -c 40 -i 5120 -m 5 -o space /dev/fd0a

#
# mount the new floppy
#
mount /dev/fd0a /mnt

#
# create required directories
#
mkdir /mnt/dev
mkdir /mnt/bin
mkdir /mnt/sbin
mkdir /mnt/etc
mkdir /mnt/root
mkdir /mnt/mnt          # for the root partition
mkdir /mnt/tmp
mkdir /mnt/var

#
# populate the directories
#
if [ ! -x /sys/compile/MINI/kernel ]
then
    cat << EOM
The MINI kernel does not exist, please create one.
Here is an example config file:
#
# MINI -- A kernel to get FreeBSD onto a disk.
#
machine          "i386"
cpu              "I486_CPU"
ident            MINI
maxusers         5

options          INET                # needed for _tcp _icmpstat _ipstat
```

---

```
                                #      _udpstat _tcpstat _udb
options      FFS                #Berkeley Fast File System
options      FAT_CURSOR          #block cursor in syscons or picons
options      SCSI_DELAY=15      #Be pessimistic about Joe SCSI device
options      NCONS=2            #1 virtual consoles
options      USERCONFIG         #Allow user configuration with -c XXX

config       kernel  root on da0 swap on da0 and da1 dumps on da0

device       isa0
device       pci0

device       fdc0    at isa? port "IO_FD1" bio irq 6 drq 2 vector fdintr
device       fd0    at fdc0 drive 0

device       ncr0

device       scbus0

device       sc0    at isa? port "IO_KBD" tty irq 1 vector scintr
device       npx0   at isa? port "IO_NPX" irq 13 vector npxintr

device       da0
device       da1
device       da2

device       sa0

pseudo-device loop          # required by INET
pseudo-device gzip         # Exec gzipped a.out's
EOM
    exit 1
fi

cp -f /sys/compile/MINI/kernel /mnt
```

---

```
gzip -c -best /sbin/init > /mnt/sbin/init
gzip -c -best /sbin/fsck > /mnt/sbin/fsck
gzip -c -best /sbin/mount > /mnt/sbin/mount
gzip -c -best /sbin/halt > /mnt/sbin/halt
gzip -c -best /sbin/restore > /mnt/sbin/restore

gzip -c -best /bin/sh > /mnt/bin/sh
gzip -c -best /bin/sync > /mnt/bin/sync

cp /root/.profile /mnt/root

cp -f /dev/MAKEDEV /mnt/dev
chmod 755 /mnt/dev/MAKEDEV

chmod 500 /mnt/sbin/init
chmod 555 /mnt/sbin/fsck /mnt/sbin/mount /mnt/sbin/halt
chmod 555 /mnt/bin/sh /mnt/bin/sync
chmod 6555 /mnt/sbin/restore

#
# create the devices nodes
#
cd /mnt/dev
./MAKEDEV std
./MAKEDEV da0
./MAKEDEV da1
./MAKEDEV da2
./MAKEDEV sa0
./MAKEDEV pty0
cd /

#
# create minimum file system table
#
cat > /mnt/etc/fstab <<EOM
/dev/fd0a / ufs rw 1 1
```

```
EOM

#
# create minimum passwd file
#
cat > /mnt/etc/passwd <<EOM
root:*:0:0:Charlie &:/root:/bin/sh
EOM

cat > /mnt/etc/master.passwd <<EOM
root::0:0::0:0:Charlie &:/root:/bin/sh
EOM

chmod 600 /mnt/etc/master.passwd
chmod 644 /mnt/etc/passwd
/usr/sbin/pwd_mkdb -d/mnt/etc /mnt/etc/master.passwd

#
# umount the floppy and inform the user
#
/sbin/umount /mnt
echo "The floppy has been unmounted and is now ready."
```

### 16.11.8.2 사고 발생 후

핵심 질문은 하드웨어는 이상이 없는가? 이다. 일반적인 백업을 받아 두었기 때문에 소프트웨어에 대해서는 걱정할 필요가 없다.

하드웨어가 파손 되었다면 파손된 부분을 교체한다.

하드웨어에 이상이 없다면 플로피를 체크한다. 사용자 부트 플로피를 사용하려면 싱글 유저 모드(boot: 프롬프트에서 `-s` 를 입력한다)로 부팅한다. 다음절을 건너뛴다.

boot.flp 와 fixit.flp 플로피를 사용한다면 계속 읽어야 된다. 플로피 드라이브에 boot.flp 플로피를 넣고 컴퓨터를 부팅한다. 화면에 최초의 설치 메뉴가 나타난다. *Fixit--Repair mode*

---

with *CDROM or floppy* 옵션을 선택하고 프롬프트가 나타나면 `fixit.flp` 를 넣는다. 필요한 `restore` 와 다른 프로그램은 `/mnt2/stand` 에 있다.

각 파일 시스템을 복구한다.

첫 번째 디스크의 root 파티션을 mount 한다(예: `mout /dev/da0a /mnt`). 디스크 라벨이 손상 되었다면 출력해서 저장한 라벨과 맞추어 디스크를 다시 파티션하고 라벨을 할당하도록 `disklabel` 를 사용한다. 파일시스템을 다시 생성하기 위해 `newfs` 을 사용한다. 플로피의 root 파티션을 읽기와 쓰기로(`mount -u -o rw /mnt`) 다시 마운트한다. 이 파일시스템을(예: `restore vrf /dev/sa0`) 복구하기 위해 백업 프로그램과 백업 테이프를 사용한다. 파일시스템을(예: `umount /mnt`) 언마운트 한다. 손상된 각 파일시스템 별로 반복한다.

시스템이 다시 살아나면 새로운 테이프에 데이터를 백업한다. 다시 시스템에 문제가 발생하거나 데이터 손실이 발생할 수 있다. 몇 시간을 투자하였기 때문에 문제가 발생하면 복구 시간을 단축할 수 있다.

## 16.12 네트워크, 메모리와 File-Backed 파일 시스템

### 템

물리적으로 컴퓨터에 있는 플로피, CD, 하드 드라이브 등을 제외하고 FreeBSD 가 알고 있는 다른 종류의 디스크는 *가상 디스크*다.

여기에는 네트워크 파일시스템과 메모리기반 파일시스템 그리고 file-backed 파일시스템이 포함된다.

사용 중인 FreeBSD 버전에 따라 file-backed 과 메모리 기반 파일시스템을 생성해서 사용하기 위해 다른 툴을 사용해야 된다.

**Note:** FreeBSD 4.X 유저는 필요한 장치를 생성하기 위해 `MAKEDEV(8)`을 사용해야 된다. FreeBSD5.0 과 새로운 버전은 `devfs(5)`를 사용하여 장치 노드를 생성한다.

### 16.12.1 FreeBSD 4.X 에서 File-Backed 파일시스템 사용

---

vnconfig(8) 유틸리티는 vnode pseudo-disk 장치를 설정하고 활성화한다. *vnode*는 파일을 표현하고 파일의 활동에 초점을 맞춘다. 이 의미는 vnconfig(8)가 파일시스템을 생성하고 운용하기 위해 파일을 이용한다는 것이다. 가능한 한가지 사용 방법은 플로피를 마운트하거나 CD 이미지를 파일로 보관하는 것이다.

vnconfig(8)을 사용하려면 커널 설정파일이 vn(4)을 지원해야 된다:

```
pseudo-device vn
```

파일시스템 이미지를 마운트 하려면 다음 예제를 확인한다:

예제 16-4. FreeBSD 4.X 에서 파일시스템 이미지를 마운트하기 위해 vnconfig 사용하기

```
# vnconfig vn0 diskimage  
  
# mount /dev/vn0c /mnt
```

vnconfig(8)로 새로운 파일시스템을 생성하려면 다음 예제를 확인한다:

예제 16-5. vnconfig 로 새로운 File-Backed 디스크 생성

```
# dd if=/dev/zero of=newimage bs=1k count=5k  
5120+0 records in  
5120+0 records out  
# vnconfig -s labels -c vn0 newimage  
# disklabel -r -w vn0 auto  
# newfs vn0c  
Warning: 2048 sector(s) in last cylinder unallocated  
/dev/vn0c: 10240 sectors in 3 cylinders of 1 tracks, 4096 sectors  
5.0MB in 1 cyl groups (16 c/g, 32.00MB/g, 1280 i/g)  
super-block backups (for fsck -b #) at:  
32  
# mount /dev/vn0c /mnt  
# df /mnt
```

Filesystem	1K-blocks	Used	Avail	Capacity	Mounted on
/dev/vn0c	4927	1	4532	0%	/mnt

## 16.12.2 FreeBSD 5.X 에서 File-Backed 파일시스템 사용

유틸리티 mdconfig(8)는 메모리 디스크를 설정해서 활성화하는데 사용하고 FreeBSD 5.X 에서는 md(4)를 대신 사용한다. mdconfig(8)을 사용하려면 md(4) 모듈을 로드하거나 지원하도록 커널 설정파일에 추가해야 된다:

```
device md
```

mdconfig(8) 명령은 3 종류의 메모리 기반 가상 디스크를 지원한다: 메모리 디스크는 malloc(9)로 할당하고 메모리 디스크도 backing 처럼 파일이나 스왑 공간을 사용한다. 가능한 사용법은 플로피를 마운트하거나 CD 이미지를 파일로 보관하는 것이다.

파일시스템을 마운트하려면 다음 예제를 확인한다:

### 예제 16-5. FreeBSD 5.X 에서 파일시스템 이미지를 마운트하기 위해 mdconfig 사용하기

```
# mdconfig -a -t vnode -f diskimage -u 0
# mount /dev/md0c /mnt
```

mdconfig(8)로 새로운 파일시스템 이미지를 생성하려면 다음 예제를 확인한다:

### 예제 16-6. mdconfig 로 새로운 File-Backed 디스크 생성하기

```
# dd if=/dev/zero of=newimage bs=1k count=5k
5120+0 records in
5120+0 records out
# mdconfig -a -t vnode -f newimage -u 0
# disklabel -r -w md0 auto
# newfs md0c
/dev/md0c: 5.0MB (10240 sectors) block size 16384, fragment size 2048
using 4 cylinder groups of 1.27MB, 81 blks, 256 inodes.
```



---

```
super-block backups (for fsck -b #) at:
```

```
32, 2624, 5216, 7808
```

```
# mount /dev/md0c /mnt
```

```
# df /mnt
```

Filesystem	1K-blocks	Used	Avail	Capacity	Mounted on
/dev/md0c	4846	2	4458	0%	/mnt

-u 옵션으로 유닛 번호를 지정하지 않았다면 mdconfig(8)은 사용하지 않는 장치를 선택하기 위해 md(4) 자동할당을 사용한다. 할당된 유닛 이름은 md4 처럼 표준 출력으로 출력된다. mdconfig(8)의 더 자세한 사항은 매뉴얼 페이지를 참고한다.

**Note:** FreeBSD 5.1-RELEASE 부터 bsdlable(8) 유틸리티가 예전 disklabel(8) 프로그램을 대체한다. bsdlable(8) 때문에 사용하지 않는 옵션과 매개변수가 없어졌다; 위 예제의 옵션 -r은 삭제해야 된다. 더 많은 정보는 bsdlable(8) 매뉴얼 페이지를 참고한다.

mdconfig(8) 유틸리티는 매우 유용하지만 file-backed 파일시스템을 생성하기 위해 수많은 명령어 라인을 문의한다. 또한 FreeBSD 5.0은 mdmfs(8)라는 툴을 가지고 있다. 이 프로그램은 디스크가 mdconfig(8)와 newfs(8)을 사용하여 UFS 파일시스템을 생성하고 mount(8)로 마운트하도록 md(4)를 설정한다. 예를 들어 위와 같은 파일시스템 이미지를 생성하고 마운트한다면 다음과 같이 입력한다:

#### 예제 16-8. mdmfs 로 File-Backed 디스크 설정해서 마운트하기

```
# dd if=/dev/zero of=newimage bs=1k count=5k
```

```
5120+0 records in
```

```
5120+0 records in
```

```
5120+0 records out
```

```
# mdmfs -F newimage -s 5m md0 /mnt
```

```
# df /mnt
```

Filesystem	1K-blocks	Used	Avail	Capacity	Mounted on
/dev/md0	4846	2	4458	0%	/mnt

유닛 번호 없이 md 옵션을 사용했다면 mdmfs(8)는 사용하지 않는 장치를 자동으로 선택하기 위해 md(4) 자동-유닛 기능을 사용한다. mdmfs(8)에 대한 더 자세한 사항은 매뉴얼 페이지를 참고한다.

---

### 16.12.3 FreeBSD 4.X 에서 메모리 기반 파일시스템 사용

md(4) 드라이버는 간단해서 FreeBSD 4.X 에 메모리 기반 파일시스템을 생성하는데 효과적이다. malloc(9)는 메모리를 할당하는데 사용된다.

예를 들어 단순히 vnconfig(8)로 준비한 파일시스템을 사용할 수 있다:

#### 예제 16 -9. FreeBSD 4.X 에서 md 메모리 디스크 사용하기

```
# dd if=newimage of=/dev/md0
5120+0 records in
5120+0 records out
# mount /dev/md0c /mnt
# df /mnt
Filesystem 1K-blocks    Used    Avail Capacity  Mounted on
/dev/md0c      4927         1    4532     0%    /mnt
```

더 자세한 사항은 md(4) 매뉴얼 페이지를 참고한다.

### 16.12.4 FreeBSD 5.X 에서 메모리 기반 파일시스템 사용

메모리 기반과 file-backed 파일시스템은 같은 틀을 사용한다: mdconfig(8)나 mdmfs(8). 메모리 기반 파일시스템을 위한 메모리는 malloc(9)로 할당된다.

#### 예제 16-10. mdconfig 로 새로운 메모리 기반 디스크 생성하기

```
# mdconfig -a -t malloc -s 5m -u 1
# newfs -U md1
/dev/md1: 5.0MB (10240 sectors) block size 16384, fragment size 2048
    using 4 cylinder groups of 1.27MB, 81 blks, 256 inodes.
    with soft updates
super-block backups (for fsck -b #) at:
    32, 2624, 5216, 7808
# mount /dev/md1 /mnt
# df /mnt
Filesystem 1K-blocks Used Avail Capacity  Mounted on
```

---

```
/dev/md1      4846    2 4458    0%   /mnt
```

#### 예제 16-11. mdmfs 로 새로운 메모리 기반 디스크 생성하기

```
# mdmfs -M -s 5m md2 /mnt
# df /mnt
Filesystem 1K-blocks Used Avail Capacity Mounted on
/dev/md2    4846    2 4458    0%   /mnt
```

mdconfig(8)에서 *malloc*을 *swap*으로 바꾸면 malloc(9)이 backed 파일시스템을 사용하지 않고 스왑을 사용하는 것도 가능하다. mdmfs(8) 유틸리는 기본적으로(-M없이) 스왑 기반 디스크를 생성한다. 더 자세한 사항은 mdconfig(8)와 mdmfs(8) 매뉴얼 페이지를 참고한다.

### 16.12.5 시스템에서 메모리 디스크 삭제

메모리 기반 또는 파일 기반 파일시스템을 사용하지 않을 때 모든 리소스를 시스템 되돌려 줘야 한다. 리소스를 되돌려 주려면 첫째로 파일시스템을 언마운트하고 시스템에서 디스크를 떼어내기 위해 mdconfig(8)을 사용하여 리소스를 되돌려 준다.

예를 들어 모든 리소스를 되돌리기 위해 /dev/md4 를 사용한다:

```
# mdconfig -d -u 4
```

명령어 라인 mdconfig -i 로 md(4) 장치 설정에 대한 정보를 나열하는 것도 가능하다.

FreeBSD 4.X 에서 vnconfig(8)은 장치를 삭제하는데 사용한다. 예를 들어 /dev/vn4 가 사용한 리소스를 되돌리는 방법은 다음 명령을 사용한다:

```
# vnconfig -u vn4
```

---

## 16.12 파일시스템 스냅샷

FreeBSD 5.0 은 소프트웨어 업데이트와 결합된 새로운 기능을 제공한다: 파일시스템 스냅샷

유저는 지정한 파일시스템의 이미지를 스냅샷으로 생성해서 이미지를 파일처럼 다룰 수 있다. 스냅샷 파일은 운용중인 파일시스템에서 생성해야 되고 유저는 파일시스템 당 20 개 이하의 스냅샷만 생성할 수 있을 것이다. 운용중인 스냅샷은 슈퍼 블록에 저장되기 때문에 시스템이 재 부팅할 때 언 마운트 했다가 다시 마운트할 때 사라지지 않는다. 스냅샷이 더 이상 필요 없다면 표준 `rm(1)` 명령으로 삭제할 수 있다. 스냅샷은 이 명령으로 삭제되지만 사용되던 공간은 다른 스냅샷이 이런 블록 중 몇 개를 요청할 것이기 때문에 반환되지 않는다.

`root` 도 스냅샷에 쓰기를 못하도록 최초에 생성될 때 `schg` 플래그가(`chflags(1)` 매뉴얼 페이지를 본다) 설정된다. `unlink(1)` 명령은 `schg` 플래그 설정으로 스냅샷을 삭제할 수 있도록 스냅샷 파일에 예외를 만들기 때문에 스냅샷 파일을 삭제하기 전에 `schg` 플래그를 제거할 필요가 없다.

스냅샷은 `mount(8)` 명령으로 생성된다. `/var` 스냅샷을 `/var/snapshot/snap` 파일에 두려면 다음 명령을 따른다:

```
# mount -u -o snapshot /var/snapshot/snap /var
```

다른 방법은 스냅샷을 생성하기 위해 `mksnap_ffs(8)`를 사용할 수 있다:

```
# mksnap_ffs /var /var/snapshot/snap
```

스냅샷이 생성되면 몇 가지 유용한 사용법이 있다:

- 스냅샷을 CD 나 테잎에 전송할 수 있기 때문에 어떤 관리자는 백업 목적으로 스냅샷 파일을 사용한다.
- `fsck(8)`은 스냅샷에서 동작 할 것이다. 파일시스템이 마운트 됐을 때 파일시스템은 깨끗하다고 보기 때문에 항상 깨끗한(변하지 않는) 결과를 유지할 수 있다. 이것이 근본적으로 백그라운드에서 `fsck(8)` 프로세스가 작동하게 된 이유다.
- 스냅샷에서 `dump(8)` 유틸리티 실행. `dump` 는 파일시스템과 스냅샷 타임스탬프의 일관성을 유지한다. `dump(8)`도 명령어 라인에 `-L` 플래그를 사용하여 `dump` 이미지

---

를 생성하고 스냅샷을 삭제할 수 있다.

- mount(8)로 정지된 파일시스템의 이미지처럼 스냅샷을 마운트한다.  
/var/snapshot/snap 에 스냅샷을 mount(8)하기 위해 다음 명령을 실행한다:

```
# mdconfig -a -t vnode -f /var/snapshot/snap -u 4
# mount -r /dev/md4 /mnt
```

이제 /mnt 에 마운트 되어있는 정지된 /var 파일시스템의 계층을 둘러볼 수 있다. 스냅샷을 생성하던 시기와 모든 것이 같은 상태다. 유일한 예외는 이전 스냅샷이 크기가 0 인 파일로 나타나는 것이다. 한계가 있는 스냅샷을 사용할 때 다음 명령으로 언 마운트할 수 있다:

```
# umount /mnt

# mdconfig -d -u 4
```

기술적인 문서를 포함하여 소프트웨어 업데이트와 파일시스템 스냅샷에 대한 더 많은 정보는 Marshall Kirk McKusick 의 웹사이트를 방문한다 <http://www.mckusick.com>.

## 16.14 파일시스템 쿼타

쿼타는 파일시스템 기반에서 유저나 그룹 멤버가 사용할 수 있는 디스크 공간과 파일의 개수를 제한할 수 있는 운영체제의 부가적인 기능이다. 이 기능은 유저나 유저 그룹에게 할당되는 리소스의 양을 제한할 필요가 있는 대부분의 시분할 시스템에서 자주 사용된다. 쿼타는 유저나 유저 그룹이 사용할 수 있는 디스크 공간이 모두 소모되는 것을 방지한다.

### 16.14.1 디스크 쿼타를 활성화하도록 시스템 설정하기

디스크 쿼타를 적용하기 전에 쿼타가 커널에 설정되어 있어야 된다. 이것은 다음 라인을 커널 설정파일에 추가하면 된다:

```
options QUOTA
```

---

표준 GENERIC 커널에서는 기본적으로 활성화되지 않기 때문에 디스크 쿼타를 사용하기 위해 사용자 커널을 설정하고 빌드 후 설치해야 된다. 커널 설정에 대한 더 많은 정보는 8 장을 참고한다.

그리고 `/etc/rc.conf` 에서 디스크 쿼타를 활성화해야 된다. 다음 라인을 추가하면 된다:

```
enable_quotas="YES"
```

쿼타 시작과 적당한 조정을 위한 추가적인 설정 매개변수가 있다. 보통 부팅할 때 각 파일 시스템의 쿼타 무결성은 `quotacheck(8)` 프로그램이 체크한다. `quotacheck(8)`은 쿼타 데이터베이스 내의 데이터가 정확히 파일시스템의 데이터와 일치하도록 한다. 이것은 시간을 많이 필요로 하는 절차여서 시스템 부팅시간에 상당한 영향을 끼친다. 이 단계를 건너뛰려면 목적에 맞게 `/etc/rc.conf` 의 변수를 변경해야 된다:

```
check_quotas="NO"
```

FreeBSD 3.2 릴리즈 이전 버전을 사용한다면 하나의 변수로만 이루어져 설정은 단순하다. 다음을 `/etc/rc.conf` 설정한다:

```
check_quotas="YES"
```

마지막으로 파일시스템 별로 디스크 쿼타를 활성화하기 위해 `/etc/fstab` 를 편집해야 된다. 이 설정으로 모든 파일시스템에 유저나 그룹 쿼타 또는 둘 다 적용할 수 있다.

시스템에서 유저 별로 쿼타를 적용하려면, 쿼타를 적용하려는 파일시스템의 `/etc/fstab` 엔트리 옵션 필드에 `userquota` 옵션을 추가한다. 예를 들어 다음과 같이 변경한다:

```
/dev/da1s2g /home ufs rw,userquota 1 2
```

그룹 쿼타를 적용하려면 비슷한 방법으로 `userquota` 대신 `groupquota` 를 사용한다. 유저와 그룹 모두에게 쿼타를 적용하려면 다음과 같이 엔트리를 변경한다:

```
/dev/da1s2g /home ufs rw,userquota,groupquota 1 2
```

---

기본적으로 쿼타 설정파일은 유저와 그룹 쿼타에 맞도록 각각 `quota.user` 와 `quota.group` 라는 이름으로 파일시스템의 `root` 디렉터리에 저장되어 있다. 더 많은 정보는 `fstab(5)`을 본다. `fstab(5)` 매뉴얼 페이지에서는 쿼타 파일의 위치를 지정할 수 있다고 하지만 다양한 쿼타 유틸리티가 정확한 위치를 파악하지 못하는 것 같아 권장하지는 않는다.

여기서 새로운 커널로 시스템을 재 부팅한다. `/etc/rc` 는 `/etc/fstab` 에서 활성화한 모든 쿼타의 최초 쿼타 파일을 생성하기 위해 적절한 명령을 자동으로 실행하기 때문에 크기가 0인 쿼타 파일을 직접 만들 필요가 없다.

일반적인 운용에서는 `quotacheck(8)`, `quotaon(8)` 또는 `quotaoff(8)` 명령을 직접 실행할 필요가 없지만 사용법을 이해할 수 있도록 매뉴얼 페이지를 읽는 것도 좋은 방법이다.

## 16.14.2 쿼타 제한 설정

시스템에 쿼타를 활성화하도록 설정하였으므로 실제로 활성화됐는지 확인한다. 확인하는 방법은 다음 명령을 실행한다:

```
# quota -v
```

디스크 사용량과 쿼타가 활성화된 각 파일시스템의 현재 쿼타 제한을 요약한 라인을 보게 된다.

```
Disk quotas for user root (uid 0):
```

Filesystem	usage	quota	limit	grace	files	quota	limit	grace
/home	2158046	0	0		16	0	0	

이제 `edquota(8)` 명령으로 쿼타 제한을 적용할 준비가 되었다.

디스크 공간을 유저나 그룹에 어떻게 할당할 것인가, 얼마나 많은 파일을 유저와 그룹이 생성할 것인가에 따라 여러 옵션이 있다. 디스크 공간이나 파일 개수 또는 두 가지를 결합하여 제한을 적용할 수 있다. 이런 제한은 두 개의 카테고리로 나누어진다: 하드와 소프트 제한.

하드 제한을 초과하지는 못한다. 유저가 하드 제한에 도달하면 이 제한이 있는 파일시스템에는 더 이상 파일을 생성하지 못한다. 예를 들어 유저가 파일시스템에 500 블록의 하드

---

제한을 가지고 있고 현재 490 블록을 사용 중이면 유저는 추가적으로 10 블록의 파일만 생성할 수 있다. 11 블록을 추가하려면 실패한다.

다른 방법의 소프트 제한은 일정한 시간 동안 초과할 수 있다. 이 기간은 기본적으로 일주일이며 유예기간(*grace period*)으로 알려져 있다. 유저가 유예기간을 어긴다면 소프트 제한은 하드 제한으로 바뀌고 더 이상 파일을 할당하지 못한다. 유저가 소프트 제한 이하로 용량을 줄이면 유예기간도 복구된다.

다음은 `edquota(8)` 명령을 실행했을 때 보게 될 예제를 보여준다. `eduquota(8)` 명령이 실행되면 쿼타 제한을 편집하도록 `EDITOR` 환경변수에 지정된 에디터로 들어가거나 `EDITOR` 변수가 설정되지 않았다면 `vi` 로 들어간다.

```
# edquota -u test
```

```
Quotas for user test:
```

```
/usr: blocks in use: 65, limits (soft = 50, hard = 75)
```

```
    inodes in use: 7, limits (soft = 50, hard = 60)
```

```
/usr/var: blocks in use: 0, limits (soft = 50, hard = 75)
```

```
    inodes in use: 0, limits (soft = 50, hard = 60)
```

보통 쿼타가 활성화된 각 파일시스템 별로 두 라인씩 보게 된다. 하나는 블록 제한 그리고 다른 하나는 `inode` 제한이다. 쿼타 제한을 수정하려면 단순히 값을 변경하고 업데이트한다. 예를 들어 유저의 블록 제한을 소프트 제한 50 과 하드 제한 75 에서 소프트제한 500 과 하드 제한 600 으로 올리기 위해 다음 내용을 아래와 같이 변경한다:

```
/usr: blocks in use: 65, limits (soft = 50, hard = 75)
```

```
위의 내용을 다음과 같이 변경한다:
```

```
/usr: blocks in use: 65, limits (soft = 500, hard = 600)
```

새로운 쿼타 제한은 에디터를 끝낼 때 적용된다.

종종 `UID` 의 범위에 쿼타 제한을 설정하는 것도 바람직하다. `-p` 옵션을 `edquota(8)` 명령에 사용하면 된다. 첫째로 유저에게 원하는 쿼타 제한을 할당하고 `edquota -p protouser(모델`



---

이 되는 유저) startuid(시작되는 ID)-enduid(끝나는 ID)를 실행한다. 예를 들어 유저 **test** 에 게 쿼타 제한을 설정하고 **UID 10,000** 부터 **19,999** 까지 똑 같이 제한할 때 다음 명령을 사용할 수 있다:

```
# edquota -p test 10000-19999
```

더 많은 정보는 **edquota(8)** 매뉴얼 페이지를 본다.

### 16.14.3 쿼타 제한과 디스크 사용량 체크

쿼타 제한과 디스크 사용량을 체크하기 위해 **quota(1)**나 **repquota(8)** 명령을 사용할 수 있다. **quota(1)** 명령은 각 유저나 그룹 쿼타와 디스크 사용량을 체크할 수 있다. 유저는 자신의 쿼타만 그리고 그룹 쿼타는 유저가 속한 그룹의 쿼타만 확인할 수 있다. 슈퍼 유저는 모든 유저와 그룹 쿼타를 볼 수 있다. **repquota(8)** 명령은 쿼타가 활성화된 파일시스템의 모든 쿼타와 디스크 사용량을 요약해서 볼 수 있다.

다음은 두 개의 파일시스템에 쿼타 제한을 가지고 있는 유저의 샘플 **quota -v** 명령의 결과다.

Disk quotas for user test (uid 1002):

Filesystem	blocks	quota	limit	grace	files	quota	limit	grace
/usr	65*	50	75	5days	7	50	60	
/usr/var	0	50	75		0	50	60	

위 예제의 **/usr** 파일시스템에서 이 유저는 50 블록의 소프트 제한에 현재 15 블록을 초과하여 유예 기간이 5 일 남았다. 별표(\*)는 유저가 현재 쿼타 제한을 초과했음을 보여준다.

유저가 전혀 사용하지 않는 일반적인 파일시스템은 이 파일시스템에 쿼타 제한이 적용되었더라도 **quota(1)** 명령의 출력에 아무것도 나오지 않는다. **-v** 옵션은 위 예제의 **/usr/var** 파일시스템처럼 이들 파일시스템을 보여줄 것이다.

### 16.14.4 NFS 에서 쿼타

쿼타 서브시스템으로 **NFS** 서버에 쿼타가 강제로 적용된다. **rpc.rquotad(8)** 데몬은 **NFS** 클라

---

이언트에서 `quota(1)` 명령으로 사용자가 자신의 쿼타 통계를 볼 수 있도록 한다.

`/etc/inetd.conf` 에서 `rpc.rquotad` 를 활성화하려면 다음 내용을 추가한다:

```
rquotad/1      dgram rpc/udp wait root /usr/libexec/rpc.rquotad rpc.rquotad
```

이제 `inetd` 를 다시 시작한다:

```
# kill -HUP `cat /var/run/inetd.pid`
```

## 16.15 디스크 파티션 암호화

FreeBSD 는 인증 받지 않은 데이터에 접근하는 것을 막기 위해 훌륭한 온라인 보호를 제공한다. 파일 퍼미션과 Mandatory Access Control (MAC)이(15 장을 본다) 컴퓨터가 켜져 있고 운영체제가 운용 중일 때 인증 받지 않은 어플리케이션으로부터 데이터 접근을 방지한다. 그러나 공격자가 물리적으로 컴퓨터에 접근하여 간단히 컴퓨터의 하드 드라이브를 다른 시스템에 복사해서 귀중한 데이터를 분석한다면 운영체제가 적용한 퍼미션은 의미가 없다.

공격자가 하드 드라이브를 소유하거나 컴퓨터의 전원을 끄고 중요한 자원에 높은 자극을 주더라도 **GEOM** 기반 디스크 암호화(**gbde**)는 컴퓨터 파일시스템의 데이터를 보호할 수 있다. 몇 개의 파일에만 암호화할 수 있는 귀찮은 방법과 달리 **gbde** 는 전체 파일시스템을 완벽하게 암호화한다.

### 16.15.1 커널에서 **gbde** 를 활성화하기

[예제: 커널에서 **gbde** 를 활성화하기]

① **root** 가 된다.

**gbde** 설정은 슈퍼 유저 권한이 필요하다.

```
% su -
```

```
Password:
```

② 운영체제 버전을 확인한다.

gbde(4)는 FreeBSD 5.0 또는 새로운 버전이 필요하다.

```
# uname -r
5.0-RELEASE
```

③ 커널 설정파일에 gbde(4) 지원을 추가한다.

좋아하는 텍스트 에디터로 커널 설정파일에 다음 라인을 추가한다:

```
options GEOM_BDE
```

설정을 추가해서 다시 컴파일 한 후 FreeBSD 커널을 설치한다. 이 절차는 8장에서 설명하였다. 마지막으로 새로운 커널로 재 부팅한다.

## 16.15.2 암호화할 하드 드라이브 준비

다음 예제는 새로운 하드 드라이브를 추가하여 파티션을 암호화한다고 가정한다. 이 파티션은 /private 에 마운트 할 것이다. gbde 는 /home 과 /var/mail 을 암호화 하는데도 사용할 수 있지만 이 소개서의 범위를 초과하는 더 복잡한 설명이 필요하다.

### [예제: 하드 드라이브를 추가하여 파티션 암호화하기]

① 새로운 하드 드라이브 추가.

16.3 장에서 설명했듯이 새로운 드라이브를 시스템에 설치한다. 이 예제의 목적에 따라 새로운 하드 드라이브 파티션이 /dev/ad4s1c 에 추가되었다. /dev/ad0s1 \* 장치는 이 예제 시스템의 표준 FreeBSD 파티션을 나타낸다.

```
# ls /dev/ad*
/dev/ad0          /dev/ad0s1b     /dev/ad0s1e     /dev/ad4s1
/dev/ad0s1       /dev/ad0s1c     /dev/ad0s1f     /dev/ad4s1c
/dev/ad0s1a      /dev/ad0s1d     /dev/ad4
```

② **gbde** 잠금(Lock) 파일을 저장할 디렉터리를 생성한다.

```
# mkdir /etc/gbde
```

**gbde** 잠금 파일은 **gbde** 가 암호화 파티션에 접근하기 위해 필요한 정보를 가지고 있다. 잠금 파일이 없는 **gbde** 는 암호화된 파티션에서 소프트웨어로 지원되지 않는 엄청난 수작업 없이 암호화된 데이터를 해석하지 못한다. 암호화된 파티션들은 각자의 잠금 파일을 사용한다.

③ **gbde** 파티션 초기화

**gbde** 파티션을 사용하기 전에 초기화해야 된다. 초기화는 한번만 수행하면 된다:

```
# gbde init /dev/ad4s1c -i -L /etc/gbde/ad4s1c
```

**gbde(8)**은 에디터를 열고 템플릿에서 다양한 옵션을 설정할 수 있게 한다. **UFS1** 이나 **UFS2** 를 사용하려면 **sector\_size** 를 2048 로 설정한다:

```
$FreeBSD: src/sbin/gbde/template.txt,v 1.1 2002/10/20 11:16:13 phk Exp $
#
# Sector size is the smallest unit of data which can be read or written.
# Making it too small decreases performance and decreases available space.
# Making it too large may prevent filesystems from working. 512 is the
# minimum and always safe. For UFS, use the fragment size
#
sector_size      =      2048
[...]
```

**gbde(8)**은 데이터 보안에 사용할 패스워드를 두 번 입력하도록 요구한다. 패스워드는 두 번다 동일해야 된다. **gbde** 의 데이터 보호능력은 선택한 패스워드의 난이도에 따라 달라진다.

이 예제에서는 **/etc/gbde/ad4s1c** 에 저장하였듯이 **gbde init** 명령은 **gbde** 파티션에 잠금 파일을 생성한다.

**경고:** **gbde** 잠금 파일은 암호화된 파티션의 내용과 함께 백업해야 된다. 잠금 파일만 삭제하면 의지를 가진 공격자가 **gbde** 파티션을 해독할 수도 있지만, 잠금 파일이 없으면 정당한 주인도 **gbde(8)**가 전혀 지원하지 못하는 엄청난 작업 없이 암호화된 파티션의 데이터에 접근하지 못한다.

④ 암호화된 파티션을 커널에 붙인다.

```
# gbde attach /dev/ad4s1c -l /etc/gbde/ad4s1c
```

암호화된 파티션을 초기화하는 동안 선택한 패스워드를 입력하도록 한다. 암호화된 새로운 장치는 `/dev` 에서 `/dev/device_name.bde` 로 나타난다:

```
# ls /dev/ad*
```

```
/dev/ad0          /dev/ad0s1b      /dev/ad0s1e      /dev/ad4s1
/dev/ad0s1        /dev/ad0s1c      /dev/ad0s1f      /dev/ad4s1c
/dev/ad0s1a       /dev/ad0s1d      /dev/ad4          /dev/ad4s1c.bde
```

⑤ 암호화된 장치에 파일시스템 생성하기

커널에 암호화된 장치가 올라오면 장치에 파일시스템을 생성할 수 있다. 암호화된 장치에 파일시스템을 생성하려면 **newfs(8)**를 사용한다. **newfs(8)**에 `-O2` 옵션을 사용하면 예전 **UFS1** 파일시스템을 초기화하는 것보다 새로운 **UFS2** 파일시스템을 초기화할 때 더 빠르기 때문에 권장한다.

**Note:** `-O2` 옵션은 FreeBSD 5.1-릴리즈와 새로운 버전에서 기본이다.

```
# newfs -U -O2 /dev/ad4s1c.bde
```

**Note:** **newfs(8)** 명령은 `*.bde` 확장자가 장치 이름에 붙은 **gbde** 파티션에서 실행해야 된다.

⑥ 암호화된 파티션 마운트하기

암호화된 파일시스템의 마운트 포인트를 생성한다

```
# mkdir /private
```

암호화된 파일시스템을 마운트한다.

```
# mount /dev/ad4s1c.bde /private
```

⑦ 암호화된 파일시스템을 사용할 수 있는지 확인한다

이제 암호화된 파일시스템을 df(1)로 보고 사용할 수 있다

```
% df -H
```

Filesystem	Size	Used	Avail	Capacity	Mounted on
/dev/ad0s1a	1037M	72M	883M	8%	/
/devfs	1.0K	1.0K	0B	100%	/dev
/dev/ad0s1f	8.1G	55K	7.5G	0%	/home
/dev/ad0s1e	1037M	1.1M	953M	0%	/tmp
/dev/ad0s1d	6.1G	1.9G	3.7G	35%	/usr
/dev/ad4s1c.bde	150G	4.1K	138G	0%	/private

암호화된 파티션

### 16.15.3 암호화된 파일시스템 마운트하기

부팅 후 파일시스템을 사용하기 전에 암호화된 파일시스템을 커널에 다시 붙이고 에러 체크 후 마운트해야 된다. 필요한 명령은 root 로 실행한다.

#### [예제: 암호화된 파일시스템 마운트하기]

① 커널에 gbde 파티션 붙이기

```
# gbde attach /dev/ad4s1c -l /etc/gbde/ad4s1c
```

암호화된 gbde 파티션을 초기화 동안 선택한 패스워드를 입력한다.

② 파일시스템 체크하기

암호화된 파일시스템을 자동으로 마운트할 수 있도록 /etc/fstab 에 추가하지 않았기 때문에 마운트하기 전에 수동으로 fsck(8)를 실행하여 이 파일시스템의 에러를 체크

---

한다.

```
# fsck -p -t ffs /dev/ad4s1c.bde
```

③ 암호화된 파일시스템 마운트하기

```
# mount /dev/ad4s1c.bde /private
```

이제 암호화된 파일시스템을 사용할 수 있다.

### 16.15.3.1 암호화된 파티션 자동으로 마운트하기

암호화된 파티션을 자동으로 붙여서 체크한 후 마운트하는 스크립트를 만드는 것이 가능하지만 보안을 위해 이 스크립트에 **gbde(8)** 패스워드는 입력하지 않는다. 대신 콘솔이나 **ssh(1)**를 통해 패스워드를 입력하여 스크립트를 직접 실행하는 것이 권장된다.

## 16.15.4 gbde 에 의한 암호 보호

**gbde(8)**는 CBC 모드에서 섹터당 128 비트의 AES 를 사용하여 암호화한다. 디스크에서 각 섹터는 다른 AES 키로 암호화된다. 사용자가 입력한 패스워드에서 섹터 키가 파생되는 과정을 포함하여 **gbde** 의 암호화 디자인에 대한 더 많은 정보는 **gbde(4)**를 본다.

## 16.15.5 호환성 문제

**sysinstall(8)**은 **gbde** 로 암호화된 장치와 호환되지 않는다. 모든 \*.bde 장치는 **sysinstall(8)** 을 시작하기 전에 커널에서 분리시켜야 되고 그렇지 않으면 장치를 최초로 검색할 때 문제가 발생한다. 예제에서 사용한 암호화된 장치를 분리시키기 위해 다음 명령을 사용한다:

```
# gbde detach /dev/ad4s1c
```

또한 **vinum(4)**은 **geom(4)** 서브시스템을 사용하지 않기 때문에 **vinum** 볼륨에 **gbde** 를 사용하지 못한다.

---

## 17 장 Vinum 볼륨 매니저

### 17.1 개요

어떤 디스크를 가지고 있더라도 항상 한계는 있다:

- 디스크가 너무 작을 수 있다
- 디스크가 너무 느릴 수 있다.
- 디스크가 너무 신뢰할 수 없을 수 있다.

위의 문제에 대한 답변으로 이번 장에서는 Vinum 볼륨 매니저를 소개한다.

### 17.2 디스크가 너무 작다.

볼륨 매니저라고 하는 Vinum 은 이들 세가지 문제를 해결할 수 있는 가상 디스크다. 좀더 자세히 알아보자.

디스크가 계속 커지고 있지만 그래도 데이터 스토리지는 필요하다. 가끔 사용할 수 있는 디스크보다 더 큰 파일시스템이 필요할 수 있다. 인정하는 바와 같이 이 문제는 10년 전보다 시급하지 않지만 아직도 존재하고 있다. 어떤 시스템은 수많은 디스크에 데이터를 저장하는 추상적인 장치를 생성하여 이 문제를 해결하였다.

### 17.3 병목 현상

현대 시스템은 수많은 사용자가 데이터에 동시에 접근해야 될 필요가 자주 있다. 예를 들어 대형 FTP 나 HTTP 서버는 동시접속자가 수천에 이르는 세션을 관리하고 외부와 몇 100Mbit/s 로 연결되어 있기 때문에 대부분 지속적으로 디스크 전송률을 초과하고 있다.



---

현재 디스크 드라이버는 순차적으로 데이터 전송율을 70 MB/S 이상으로 올릴 수 있지만 독립적인 프로세스가 많이 접근하는 드라이브 환경에서 이 값은 별로 중요하지 않다. 이 경우 디스크 서브시스템의 관점에서 이 문제를 보는 것이 더욱 중요하다: 중요한 매개변수는 서브 시스템에서 발생하는 전송 로드, 다시 말해 드라이브가 전송에 관여되는데 소비되는 시간이다.

디스크에서 데이터를 전송할 때 드라이브는 첫 번째 섹터가 헤드에 읽혀질 때까지 기다린 후 전송을 시작한다.

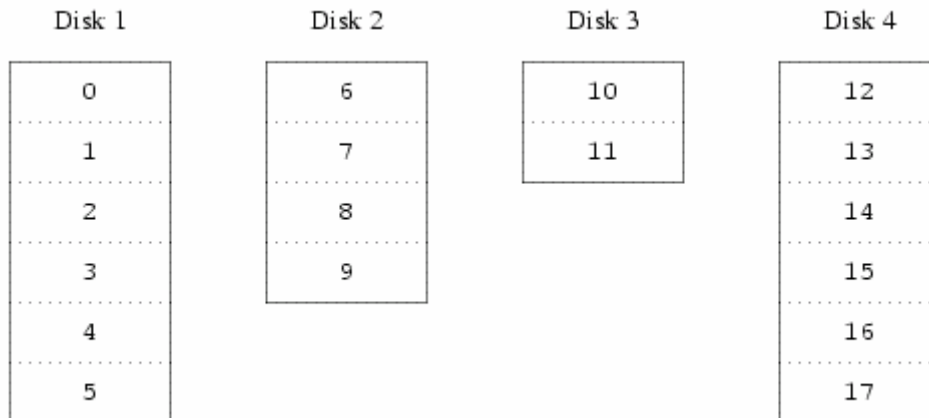
일반적으로 10kB 를 전송하는 것을 생각해 보자: 현 세대의 고속 디스크는 평균 3.5 ms 속도로 헤드를 움직일 수 있다. 가장 빠른 드라이브 회전속도는 15,000rpm 에 달하므로 회전 지연 시간(Rotational Latency: 섹터를 찾는 시간)은 2 ms 다. 70MB/s 를 전송하는데 대략 150 $\mu$ s 가 소요되므로 위치를 잡는 시간과 비교되지 않는다. 이러한 경우 효율적인 전송률은 1MB/s 이하로 떨어지고 전적으로 전송 크기에 의존된다.

이런 병목을 해결하기 위한 전통적이고 확실한 방법은 "더욱 빠른 회전이다": 큰 디스크 하나를 사용하는 것보다 여러 개의 작은 디스크를 같은 스토리지 공간에 묶는 것이 좋다. 각 디스크 별로 데이터를 독립적으로 전송할 수 있기 때문에 여러 개의 디스크를 사용해서 처리량을 효과적으로 증가시킬 수 있다.

정확한 데이터 처리량 향상은 처리에 사용되는 디스크 개수와 비례하지 않는다: 병렬로 연결했을 때 각 드라이브의 전송량이 증가 하더라도 요청을 드라이브에 고르게 분배할 수 없다. 불가피하게 하나의 드라이브 부하가 다른 것보다 높게 된다.

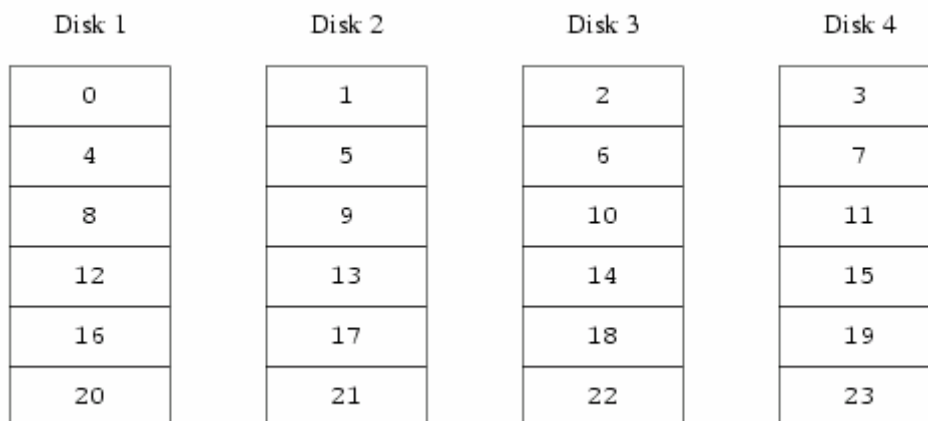
균등한 디스크 부하는 데이터가 드라이브에 공유되는 방법에 크게 의존된다. 다음 설명은 디스크 스토리지를 책의 페이지처럼 번호로 주소를 할당할 수 있는 수많은 데이터 섹터처럼 생각하는 것이 이해하기 쉽다. 가장 확실한 방법은 큰 책을 작은 섹션으로 나누는 것처럼 별개의 물리적인 디스크의 연속된 섹터 크기로 가상 디스크를 그룹화해서 저장한다. 이런 방법을 *concatenation* 라고 하고 디스크를 특정 크기로 지정할 필요가 없는 장점이 있다. 가상 디스크를 사용하려는 요청이 주소 영역으로 공평하게 분배될 때 성능이 향상된다. 한 곳에 요청이 집중되면 성능은 별로 향상되지 않는다. 그림 17-1 은 스토리지 유닛이 어떤 순서로 할당되었는지 보여준다.

그림 17-1 연결 구조



다른 매핑 방법은 주소 공간을 더 작고 같은 크기의 컴포넌트로 나누어서 다른 장치에 순서대로 저장한다. 예를 들어 첫 번째 256 섹터는 첫 번째 디스크에 저장하고 다음 256 섹터는 다음 디스크에 저장하는 방식이다. 마지막 디스크가 가득찬 후 이 프로세스는 디스크가 가득 찰 때까지 반복한다. 이 매핑을 *스트라이핑(Striping)* 또는 RAID-0 라고 한다. 스트라이핑은 데이터를 저장하는데 약간의 작업이 더 필요하다. 따라서 다수의 디스크에 데이터를 분산시켜야 되는 추가적인 I/O 가 발생하지만 로드를 더욱 균등하게 디스크에 할당할 수 있다. 그림 17-2 는 스트라이프 구조에서 스토리지 유닛이 할당된 순서를 보여준다.

그림 17-2. 스트라이프된 구조



## 17.4 데이터 무결성

현대 디스크의 마지막 문제는 신뢰성이다. 디스크 드라이브의 신뢰성은 지난 몇 년간 엄청난 속도로 발전했지만 아직도 서버를 다운시키는 핵심요소가 되고 있다. 디스크 문제가 발생했을 때 이 결과는 비극적일 수 있다: 디스크 드라이브를 교체하고 데이터를 복구하는데 몇 일을 소비할 수 있다.

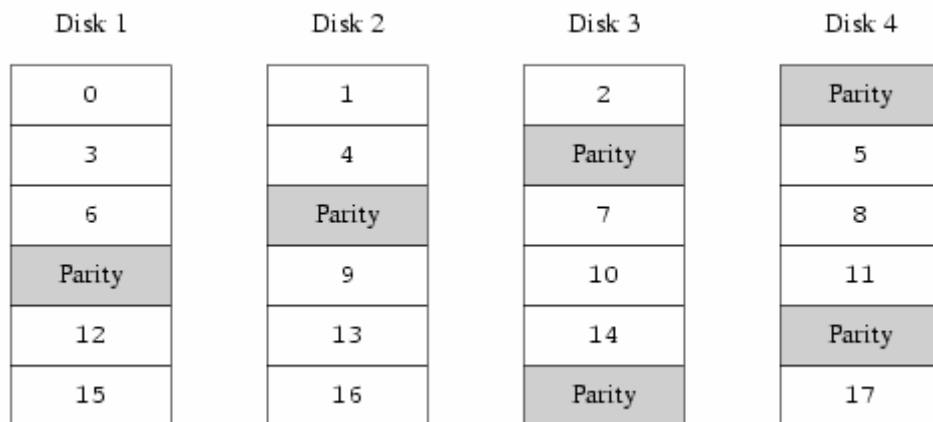
이 문제를 해결하는 전통적인 방법은 두 개의 물리적인 디스크를 서로 복제하는 미러링(mirroring)이다. RAID 레벨의 장점을 가지고 있기 때문에 이 기술은 RAID 레벨 1 이나 RAID-1 이라고 부른다. 양쪽 디스크에 데이터를 기록하므로 쓰기 속도는 상당히 느리지만 양쪽에서 읽을 수 있어 읽기 속도는 만족할만하다. 그리고 드라이브 하나가 실패하더라도 다른 드라이브의 데이터를 사용할 수 있다.

미러링은 두 가지 문제가 있다:

- 중복된 솔루션으로 두 배의 디스크 스토리지가 필요하다.
- 성능 효과; 빠르게 보이지만 성능 면에서 이점은 없다.

다른 솔루션은 RAID 레벨 2, 3, 4 와 5 에서 사용하는 *패리티*(parity)다. 여기서 RAID-5 가 가장 흥미롭다. Vinum 은 각 스트라이프의 블록 하나를 다른 블록의 패리티로 사용하는 스트라이프 구조에서 변형된 것이다. 각 스트라이프에 패리티 블록이 포함되어 RAID-5 가 실행되는 것을 제외하고 Vinum 에서 RAID-5 플렉스(plex)는 스트라이프된 플렉스(plex)와 비슷하다. RAID-5 에 필요하기 때문에 이 패리티 블록의 위치는 하나의 스트라이프에서 다음으로 이동한다. 데이터 블록에서 번호는 상대적인 블록 번호를 나타낸다.

그림 17-3. RAID-5 구조



---

미러링과 비교하여 RAID-5의 장점은 필요한 스토리지 공간이 눈에 띄게 적어진다. 읽기 속도는 스트라이프 구조와 비슷하지만 쓰기 속도는 대략 읽기 속도의 25%가 떨어진다. 드라이브 하나에 문제가 발생하더라도 어레이는 느린 속도로 계속 동작할 수 있다: 한 개의 디스크에 장애가 생기면 RAID-4나 RAID-5는 여유 디스크(spare disk)에 기존의 데이터 정보를 기초로 한 패리티 연산으로 원래의 데이터를 살려낸다: 남아있는 드라이브에서는 계속 읽을 수 있지만 문제가 발생한 드라이브에서 읽기는 남아있는 모든 드라이브의 일치하는 블록에서 재계산된다.

## 17.5 Vinum 오브젝트

이러한 문제를 해결하기 위해 Vinum은 4단계의 오브젝트를 실행한다:

- 가장 확실한 오브젝트는 **볼륨(volume)**이라는 가상 디스크다. 볼륨은 약간의 차이가 있더라도 본질적으로 유닉스 디스크 드라이브와 같은 특성을 가지고 있다. 크기 제한이 없다.
- 볼륨은 볼륨의 전체 주소 공간을 표현하는 **플렉스(plexes)**로 구성되어 있다. 따라서 계층적으로 이 레벨에서 백업이 제공된다. 플렉스를 미러된 어레이에서 각각의 디스크로 생각하면 각 디스크는 같은 데이터를 가지고 있다.
- Vinum이 유닉스 디스크 스토리지 프레임워크에 존재하기 때문에 멀티 디스크 플렉스 블록을 생성할 때 유닉스 파티션을 사용할 수 있지만 유연성이 너무 떨어져서 사용하지 않는다: 유닉스 디스크는 제한된 파티션 수를 가진다. 대신 Vinum은 싱글 유닉스 파티션을 **서브디스크(플렉스 블록을 생성하는데 사용한다)**라는 연속된 영역으로 나눈다.
- 서브디스크는 Vinum **드라이브(현재 유닉스 파티션)**에 존재한다. Vinum 드라이브는 많은 서브디스크를 포함할 수 있다. 설정과 상태 정보를 저장하는데 사용되는 드라이브 시작부분의 작은 영역을 제외하고 전체 드라이브를 데이터 저장에 사용할 수 있다.

다음 섹션은 이러한 목적을 지원하기 위해 필요한 Vinum의 기능에 대해 설명한다.

---

## 17.5.1 볼륨 크기 고려

Vinum 설정에서 플렉스는 모든 드라이브에 퍼져있는 여러 개의 서브디스크를 포함할 수 있다. 이 결과 각 드라이브는 플렉스의 크기를 제한하지 않기 때문에 볼륨의 제한도 없다.

## 17.5.2 중복 데이터 스토리지

Vinum 은 여러 개의 플렉스를 볼륨에 붙여서 미러링하고 각 플렉스는 볼륨의 데이터를 표현한다. 따라서 볼륨은 1 에서 8 개의 플렉스를 포함할 것이다.

플렉스가 볼륨의 전체 데이터를 표현하더라도 표현되는 부분이 디자인이나(플렉스의 서브디스크로 정의하지 않아서) 사고로 인해(드라이브 실패의 결과로) 물리적으로 사라질 수도 있다. 그러나 플렉스 하나가 전체 볼륨의 데이터를 제공할 수 있어서 볼륨이 완벽하게 동작한다.

## 17.5.3 성능 문제

Vinum 은 플렉스 레벨에서 concatenation 과 striping 을 실행한다.

- 연속된 플렉스(*concatenated plex*)는 각 서브디스크를 번갈아 사용한다.
- 스트라이프된 플렉스(*striped plex*)는 데이터를 각 서브 디스크로 나눈다. 서브 디스크들은 같은 크기여야 되고 연속된 플렉스에서 구분하기 위해 최소한 두 개의 서브 디스크 있어야 된다.

## 17.5.4 어떤 플렉스 구조인가?

FreeBSD 5.2.1 에 제공되는 Vinum 버전은 두 종류의 플렉스를 실행한다:

- 연속된 플렉스는 아주 유연하다: 이들은 여러 개의 서브디스크를 포함할 수 있고 이 서브디스크는 다른 크기일 수 있다. 플렉스는 추가적인 서브디스크를 추가하여 확장될 것이다. CPU 오버헤드를 측정할 수 없더라도 스트라이프된 플렉스보다

CPU 사용시간이 적다. 한편 이 디스크 중 하나는 사용량이 많고 다른 디스크는 그렇지 않아서 한쪽이 과열되기 쉽다.

- 스트라이프된(RAID-0) 플렉스의 가장 큰 장점은 과열을 줄이는 것이다: 최적의 스트라이프(대략 256KB) 크기를 선택하여 드라이브에 부하를 고르게 분산시킬 수 있다. 이러한 방식의 단점은 더욱 복잡한 코드와 서브디스크의 제한이다: 이 서브디스크들은 모두 같은 크기여야 되고 새로운 서브디스크를 추가하여 확장한 플렉스는 너무 복잡하기 때문에 현재 Vinum 이 지원하지 못한다. 추가적으로 Vinum 에는 사소한 한계가 있다: 최소한 두 개의 서브디스크를 가지고 있지 않으면 스트라이프된 플렉스를 연속된 플렉스와 구분할 수 없기 때문에 두 개의 서브디스크가 필요하다.

표 17-1 에 각 플렉스 구조의 장점과 단점을 요약했다.

표 17-1. Vinum 플렉스 구조

플렉스 종류	최소 서브디스크	서브디스크 추가 가능?	같은 크기 필요?	어플리케이션
Concatenated	1	Yes	No	최고의 배치 유연성과 적절한 성능의 대형 데이터 스토리지
Striped	2	No	Yes	높은 동시 접근성과 결합되어 고 성능이다

## 17.6 몇 가지 예제

Vinum 은 각 시스템이 알고 있는 오브젝트가 설명되어 있는 *설정 데이터베이스*를 관리한다. 유저는 `vinum(8)` 유틸리티 프로그램으로 하나 또는 하나 이상의 설정파일에서 최초로 설정 데이터베이스를 생성한다. Vinum 은 설정 데이터베이스의 복사본을 각 디스크 슬라이스에 (Vinum 은 *장치*라고 부른다) 저장한다. 이 데이터베이스는 각 단계가 변경될 때 업데이트 되기 때문에 각 Vinum 오브젝트 상태를 정확하게 복구하고 재 시작한다.

---

## 17.6.1 설정 파일

설정파일은 각각의 Vinum 오브젝트를 설명한다. 간단한 볼륨의 정의는 다음과 같다:

```
drive a device /dev/da3h
  volume myvol
    plex org concat
      sd length 512m drive a
```

이 파일은 4 개의 Vinum 오브젝트를 설명한다:

- *drive* 라인은 디스크 파티션과(*drive*) 하드웨어에서 상대 경로를 설명한다. 여기서는 심볼릭 이름 *a*를 할당했다. 장치 이름에서 심볼릭 이름을 분리하여 혼동 없이 디스크를 한곳에서 다른 곳으로 옮길 수 있다.
- *volume* 라인은 볼륨을 설명한다. 이름만 필요하기 때문에 여기서는 *myvol*이다.
- *plex* 라인은 플렉스를 정의한다. 구조 매개변수만 필요하기 때문에 여기서는 *concat*이다. 이름은 입력하지 않는다: 시스템은 볼륨이름에 *.px*를 붙여서 자동으로 이름을 생성한다. *x*는 볼륨에서 플렉스 번호다. 그래서 이 플렉스는 *myvol.p0*라고 부른다.
- *sd* 라인은 서브디스크를 설명한다. 최소한의 설정은 저장될 드라이브 이름과 서브디스크 크기다. 플렉스이기 때문에 이름은 지정하지 않는다: 시스템은 플렉스 이름에서 파생된 이름에 *.sx*를 붙여서 자동으로 이름을 할당한다. *x*는 플렉스에서 서브디스크 번호다. 그래서 Vinum은 이 서브디스크 이름을 *myvol.p0.s0*로 할당한다.

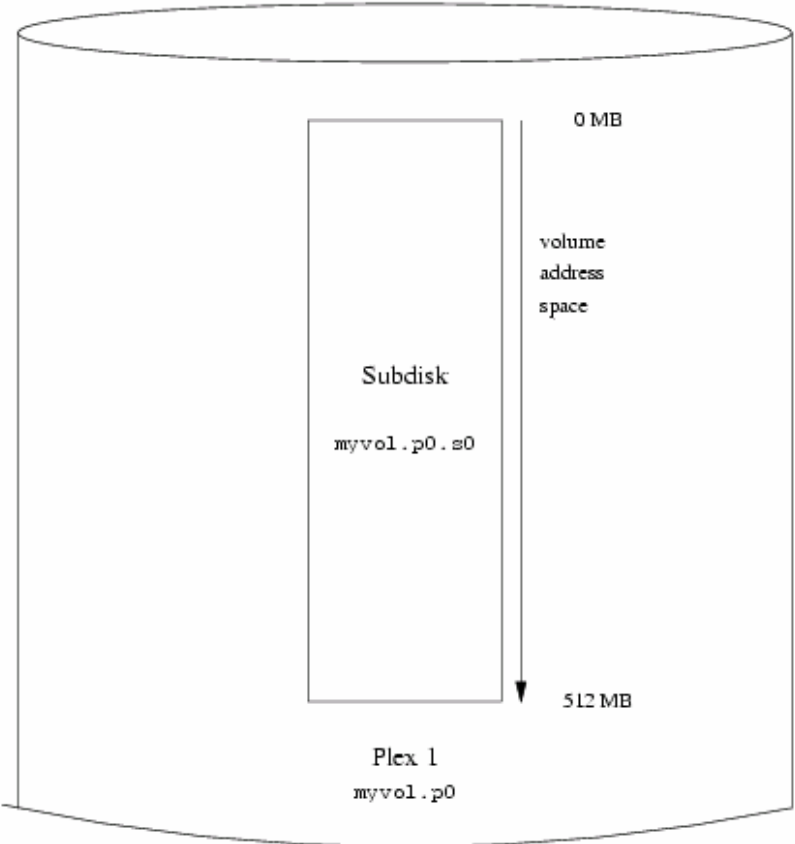
이 파일 처리가 끝나면 `vinum(8)`은 다음 내용을 출력한다:

```
# vinum -> create config1
Configuration summary
Drives:      1 (4 configured)
Volumes:     1 (4 configured)
Plexes:      1 (8 configured)
```

Subdisks:		1 (16 configured)		
D a	State: up	Device /dev/da3h	Avail:	
2061/2573 MB (80%)				
V myvol	State: up	Plexes: 1	Size:	512 MB
P myvol.p0	C State: up	Subdisks: 1	Size:	512 MB
S myvol.p0.s0	State: up	PO: 0	B Size:	512 MB

이 결과는 vinum(8) 포맷을 간략하게 나열해주고 그림 17-4 에서 그래픽으로 표현해 보았다.

그림 17-4. 간단한 Vinum 볼륨





이 그림과 다음 그림은 서브디스크를 번갈아 사용하는 플렉스를 가진 볼륨을 표현한다. 이 평범한 예제의 볼륨은 플렉스 하나를 가지고 있고 플렉스는 서브디스크를 하나 가지고 있다.

이 볼륨의 플렉스는 싱글 서브디스크를 가지고 있어서 전통적인 디스크 파티션의 스토리지 할당과 전혀 다르지 않다. 따라서 전통적인 디스크 파티션에 비해 큰 장점이 없다. 다음 섹션은 다양하고 더욱 흥미로운 설정 방법을 제시한다.

### 17.6.2 복원력 증가: 미러링

볼륨의 복원력은 미러링으로 증가시킬 수 있다. 미러된 볼륨의 위치를 지정할 때 각 플렉스의 서브디스크를 다른 드라이브에 놓는 것이 중요하다. 그래서 드라이브에 문제가 발생하더라도 두 플렉스가 동시에 다운되지 않는다. 다음은 볼륨을 미러하는 설정이다:

```
drive b device /dev/da4h
  volume mirror
    plex org concat
      sd length 512m drive a
    plex org concat
      sd length 512m drive b
```

Vinum의 모든 오브젝트 트랙이 설정 데이터베이스에 있으므로 이 예제에서 드라이브 *a*를 다시 정의할 필요가 없다. 이 정의 후 설정파일은 다음과 비슷할 것이다:

```
Drives:          2 (4 configured)
Volumes:         2 (4 configured)
Plexes:          3 (8 configured)
Subdisks:        3 (16 configured)

D a              State: up      Device /dev/da3h    Avail: 1549/2573 MB (60%)
D b              State: up      Device /dev/da4h    Avail: 2061/2573 MB (80%)

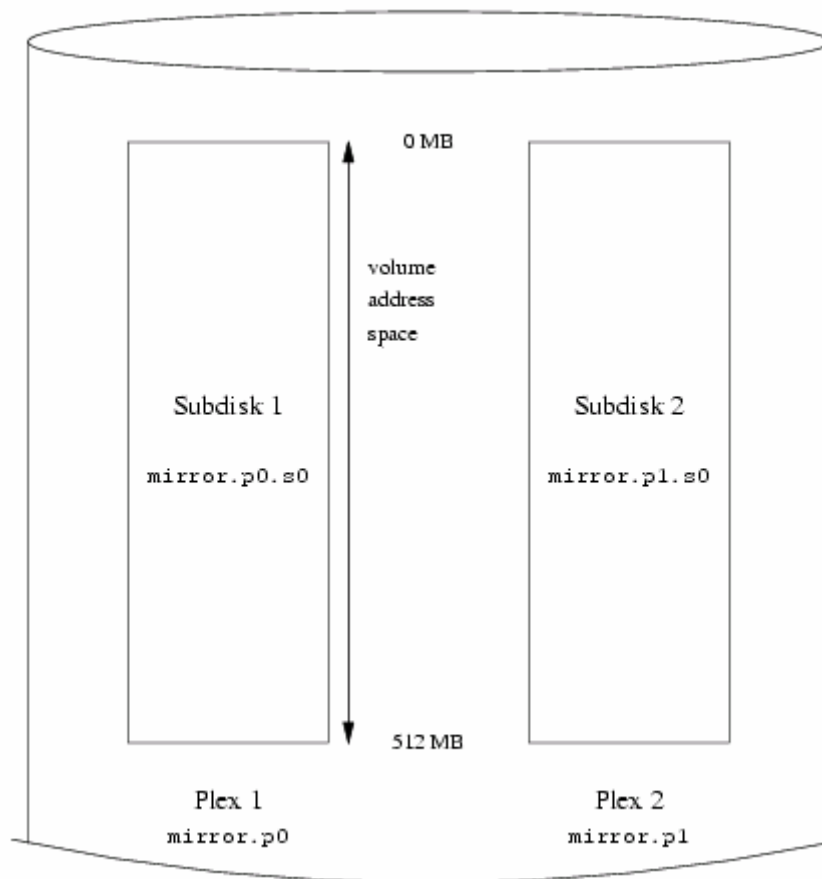
V myvol          State: up      Plexes:      1 Size:      512 MB
V mirror         State: up      Plexes:      2 Size:      512 MB

P myvol.p0       C State: up      Subdisks:    1 Size:      512 MB
```

P mirror.p0	C State: up	Subdisks: 1	Size: 512 MB
P mirror.p1	C State: initializing	Subdisks: 1	Size: 512 MB
S myvol.p0.s0	State: up	PO: 0	B Size: 512 MB
S mirror.p0.s0	State: up	PO: 0	B Size: 512 MB
S mirror.p1.s0	State: empty	PO: 0	B Size: 512 MB

그림 17-5 에서 위의 구조를 그래픽으로 보여준다.

그림 17-5 미러된 Vinum 볼륨



이 예제에서 각 플렉스는 완전한 512MB 주소공간을 가지고 있다. 이전 예제에서 각 플렉스는 싱글 서브디스크만 포함했었다.

---

### 17.6.3 성능 최적화

이전 예제에서 미러된 볼륨은 미러되지 않은 볼륨에 비해 좀더 강했지만 성능은 더 떨어졌다: 전체 디스크 대역폭의 균형된 사용을 위해 양쪽 드라이브에 데이터를 기록해야 된다. 성능 향상을 위해서 다른 방법이 요청되었다: 미러링 대신 데이터를 가능한 많은 디스크 드라이브에 스트라이프 한다. 다음은 4 개의 디스크 드라이브에 플렉스로 스트라이프된 볼륨 설정을 보여준다:

```
drive c device /dev/da5h
drive d device /dev/da6h
volume stripe
plex org striped 512k
sd length 128m drive a
sd length 128m drive b
sd length 128m drive c
sd length 128m drive d
```

앞에서 Vinum 이 알고 있기 때문에 드라이브를 다시 정의할 필요가 없다. 이 정의 절차가 끝난 후 설정은 다음과 비슷할 것이다:

```
Drives:          4 (4 configured)
Volumes:         3 (4 configured)
Plexes:          4 (8 configured)
Subdisks:        7 (16 configured)

D a              State: up      Device /dev/da3h    Avail: 1421/2573 MB (55%)
D b              State: up      Device /dev/da4h    Avail: 1933/2573 MB (75%)
D c              State: up      Device /dev/da5h    Avail: 2445/2573 MB (95%)
D d              State: up      Device /dev/da6h    Avail: 2445/2573 MB (95%)

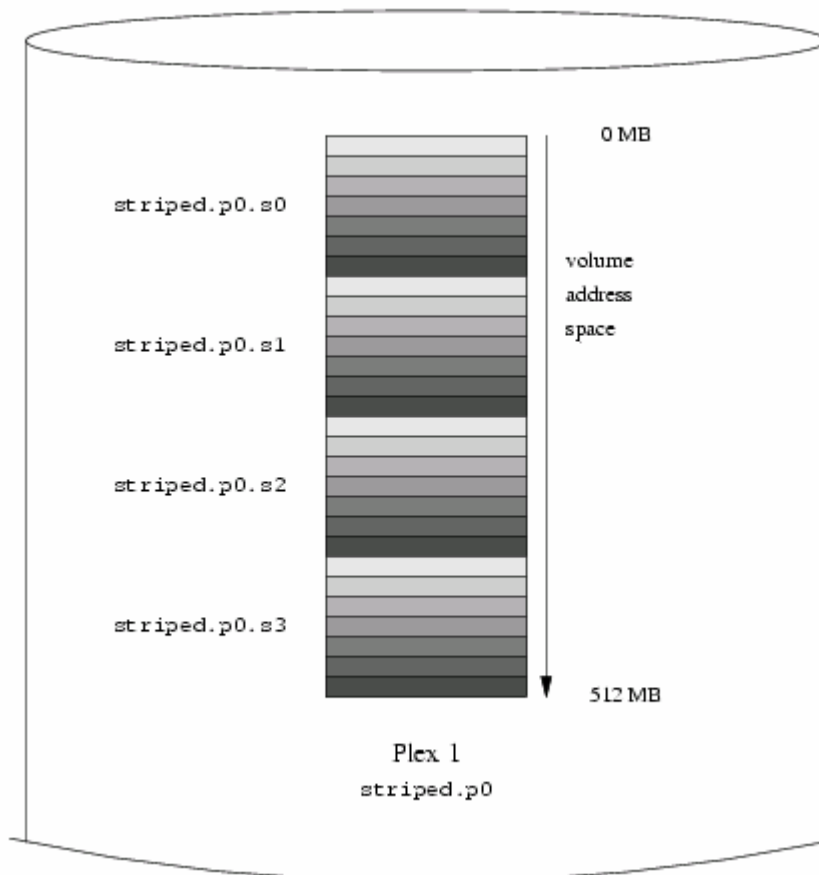
V myvol          State: up      Plexes:      1 Size:      512 MB
V mirror         State: up      Plexes:      2 Size:      512 MB
V striped        State: up      Plexes:      1 Size:      512 MB

P myvol.p0       C State: up    Subdisks:    1 Size:      512 MB
P mirror.p0      C State: up    Subdisks:    1 Size:      512 MB
```

P mirror.p1	C State: initializing	Subdisks:	1	Size:	512 MB
P striped.p1	State: up	Subdisks:	1	Size:	512 MB
S myvol.p0.s0	State: up	PO:	0	B Size:	512 MB
S mirror.p0.s0	State: up	PO:	0	B Size:	512 MB
S mirror.p1.s0	State: empty	PO:	0	B Size:	512 MB
S striped.p0.s0	State: up	PO:	0	B Size:	128 MB
S striped.p0.s1	State: up	PO:	512 kB	Size:	128 MB
S striped.p0.s2	State: up	PO:	1024 kB	Size:	128 MB
S striped.p0.s3	State: up	PO:	1536 kB	Size:	128 MB

이 볼륨을 그림 17-6 에서 그래픽으로 다시 보여준다.

그림 17-6. 스트라이프된 Vinum 볼륨



---

스트라이프의 어두운 부분은 플렉스 주소 공간의 위치를 나타낸다: 밝은 스트라이프가 처음에 오고 어두운 부분은 나중에 온다.

## 17.6.4 복원력과 성능

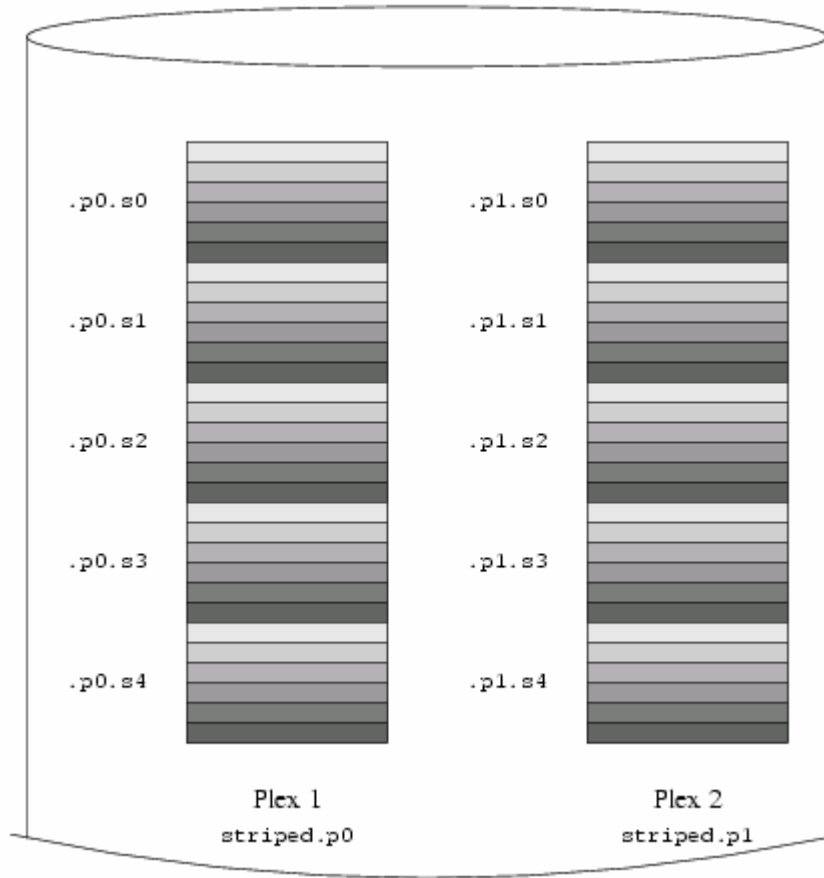
표준 유닉스 파티션과 비교하여 복원력과 성능이 증가된 볼륨을 충분한 하드웨어로 만들 수 있다. 일반적인 설정파일은 다음과 비슷할 것이다:

```
volume raid10
  plex org striped 512k
    sd length 102480k drive a
    sd length 102480k drive b
    sd length 102480k drive c
    sd length 102480k drive d
    sd length 102480k drive e
  plex org striped 512k
    sd length 102480k drive c
    sd length 102480k drive d
    sd length 102480k drive e
    sd length 102480k drive a
    sd length 102480k drive b
```

두 번째 플렉스의 서브 디스크는 이러한 첫 번째 플렉스의 두 번째 드라이브에 의해 오프셋(offset) 된다: 이 설정으로 두 드라이브의 전송량을 증가하더라도 같은 서브디스크에 기록하지 않게 된다.

그림 17-7에서 이 볼륨의 구조를 그래픽으로 표현했다.

그림 17-7. 스트라이프되고 미러된 Vinum 볼륨



## 17.7 오브젝트에 이름 할당하기

앞에서 설명했듯이 덮어쓰게 되더라도 Vinum은 플렉스와 서브디스크에 기본 이름을 할당한다. 기본 이름을 덮어쓰는 것은 권장하지 않는다: 임의 적인 이름의 오브젝트를 허용하는 VERITAS 볼륨 매니저를 겪어 봤다면 이 유연성은 중요한 장점이 아닌 혼동을 초래할 수 있다.

이름에 공백 문자가 아닌 문자를 지정할 수 있다. 그러나 이름을 할당할 때 문자, 숫자와 언더 스코어(\_) 문자로 제한하는 것을 권장한다. 볼륨과 플렉스 그리고 서브디스크의 이름은 64 캐릭터 이상이고 드라이브 이름은 32 캐릭터 이상일 것이다.

Vinum 오브젝트는 /dev/vinum 계층에 장치 노드를 할당한다. 위에서 보여준 설정으로 Vinum은 다음과 같은 장치 노드를 생성할 것이다:

- `vinum(8)`과 `Vinum` 데몬이 각각 사용하는 컨트롤 장치는 `/dev/vinum/control` 과 `/dev/vinum/controld` 이다.
- 각 볼륨의 블록과 캐릭터 장치 엔트리는 `Vinum` 이 사용하는 메인 장치다. 블록 장치 이름은 볼륨 이름이지만 캐릭터 장치 이름은 BSD 전통에 따라 이름 앞에 문자 `r`을 붙인다. 그래서 위의 설정은 블록 장치 `/dev/vinum/myvol`, `/dev/vinum/mirror`, `/dev/vinum/striped`, `/dev/vinum/raid5` 와 `/dev/vinum/raid10` 이다. 그리고 캐릭터 장치는 `/dev/vinum/rmyvol`, `/dev/vinum/rmirror`, `/dev/vinum/rstriped`, `/dev/vinum/rraid5` 와 `/dev/vinum/rraid10` 이다. 여기에 문제가 있다: `r`과 `rr`이라는 두 개의 볼륨을 가질 수 있지만 생성하려는 장치 노드 `/dev/vinum/rr` 와 충돌한다: 이것이 캐릭터 장치의 볼륨 `r`인지 또는 블록 장치의 볼륨 `rr`인가? `Vinum` 은 현재 이 문제를 설명하지 않는다: 결과는 첫 번째로 정의된 볼륨이 이름을 갖게 된다.
- 각 드라이브 엔트리와 디렉터리 `/dev/vinum/volume` 을 생성한다. 이들 엔트리는 사실 디스크 노드와 일치하는 심볼릭 링크다.
- 각 볼륨 엔트리와 디렉터리 `/dev/vinum/volume` 을 생성한다. 이 디렉터리에는 컴포넌트 서브디스크로 변경되는 각 플렉스의 서브디렉터리가 있다.
- 각 플렉스와 블록의 블록 장치 노드 그리고 각 서브디스크의 캐릭터 장치 노드를 가지고 있는 디렉터리 `/dev/vinum/plex`, `/dev/vinum/sd` 와 `/dev/vinum/rsd` 를 생성한다.

예를 들어 다음 설정파일을 확인해 보자:

```
drive drive1 device /dev/sd1h
  drive drive2 device /dev/sd2h
  drive drive3 device /dev/sd3h
  drive drive4 device /dev/sd4h
volume s64 setupstate
  plex org striped 64k
    sd length 100m drive drive1
    sd length 100m drive drive2
    sd length 100m drive drive3
```

---

**sd length 100m drive drive4**

이 파일에 따라 vinum(8)은 /dev/vinum 에 다음 구성을 생성한다:

```
brwx----- 1 root wheel 25, 0x40000001 Apr 13 16:46 Control
brwx----- 1 root wheel 25, 0x40000002 Apr 13 16:46 control
brwx----- 1 root wheel 25, 0x40000000 Apr 13 16:46 controld
drwxr-xr-x 2 root wheel 512 Apr 13 16:46 drive
drwxr-xr-x 2 root wheel 512 Apr 13 16:46 plex
crwxr-xr-- 1 root wheel 91, 2 Apr 13 16:46 rs64
drwxr-xr-x 2 root wheel 512 Apr 13 16:46 rsd
drwxr-xr-x 2 root wheel 512 Apr 13 16:46 rvol
brwxr-xr-- 1 root wheel 25, 2 Apr 13 16:46 s64
drwxr-xr-x 2 root wheel 512 Apr 13 16:46 sd
drwxr-xr-x 3 root wheel 512 Apr 13 16:46 vol
```

/dev/vinum/drive:

total 0

```
lrwxr-xr-x 1 root wheel 9 Apr 13 16:46 drive1 -> /dev/sd1h
lrwxr-xr-x 1 root wheel 9 Apr 13 16:46 drive2 -> /dev/sd2h
lrwxr-xr-x 1 root wheel 9 Apr 13 16:46 drive3 -> /dev/sd3h
lrwxr-xr-x 1 root wheel 9 Apr 13 16:46 drive4 -> /dev/sd4h
```

/dev/vinum/plex:

total 0

```
brwxr-xr-- 1 root wheel 25, 0x10000002 Apr 13 16:46 s64.p0
```

/dev/vinum/rsd:

total 0

```
crwxr-xr-- 1 root wheel 91, 0x20000002 Apr 13 16:46 s64.p0.s0
crwxr-xr-- 1 root wheel 91, 0x20100002 Apr 13 16:46 s64.p0.s1
crwxr-xr-- 1 root wheel 91, 0x20200002 Apr 13 16:46 s64.p0.s2
crwxr-xr-- 1 root wheel 91, 0x20300002 Apr 13 16:46 s64.p0.s3
```

/dev/vinum/rvol:



```

total 0
crwxr-xr--  1 root  wheel  91,  2 Apr 13 16:46 s64

/dev/vinum/sd:
total 0
brwxr-xr--  1 root  wheel  25, 0x20000002 Apr 13 16:46 s64.p0.s0
brwxr-xr--  1 root  wheel  25, 0x20100002 Apr 13 16:46 s64.p0.s1
brwxr-xr--  1 root  wheel  25, 0x20200002 Apr 13 16:46 s64.p0.s2
brwxr-xr--  1 root  wheel  25, 0x20300002 Apr 13 16:46 s64.p0.s3

/dev/vinum/vol:
total 1
brwxr-xr--  1 root  wheel  25,  2 Apr 13 16:46 s64
drwxr-xr-x  3 root  wheel      512 Apr 13 16:46 s64.plex

/dev/vinum/vol/s64.plex:
total 1
brwxr-xr--  1 root  wheel  25, 0x10000002 Apr 13 16:46 s64.p0
drwxr-xr-x  2 root  wheel      512 Apr 13 16:46 s64.p0.sd

/dev/vinum/vol/s64.plex/s64.p0.sd:
total 0
brwxr-xr--  1 root  wheel  25, 0x20000002 Apr 13 16:46 s64.p0.s0
brwxr-xr--  1 root  wheel  25, 0x20100002 Apr 13 16:46 s64.p0.s1
brwxr-xr--  1 root  wheel  25, 0x20200002 Apr 13 16:46 s64.p0.s2
brwxr-xr--  1 root  wheel  25, 0x20300002 Apr 13 16:46 s64.p0.s3

```

플렉스와 서브디스크에는 특정 이름을 할당하지 않아야 되지만 Vinum 드라이브에는 이름을 할당해야 된다. 이렇게 해서 드라이브를 다른 위치로 이동시켜도 자동으로 인지할 수 있다. 드라이브 이름은 32 캐릭터 이상일 것이다.

## 17.7.1 파일시스템 생성하기

한가지 예외를 제외하고 볼륨은 디스크와 동일하다. 유닉스 드라이브와 달리 파티션 테이블을 포함하지 않는 Vinum 은 볼륨을 파티션 하지 않는다. 따라서 특정 디스크 유틸리티를 수

---

정해야 된다. 특히 이전에 Vinum 볼륨의 마지막 문자인 파티션 확인자를 해석하려는 newfs(8)다. 예를 들어 디스크 드라이브는 /dev/ad0a 또는 /dev/da2h 와 비슷한 이름을 가질 수 있다. 이들 이름은 첫 번째(0) IDE 디스크(ad)의 첫 번째 파티션(a)과 세 번째(2) SCSI 디스크(da)의 여덟 번째 파티션(h)을 각각 나타낸다. 대조적으로 Vinum 볼륨은 파티션 이름과 관련이 없는 /dev/vinum/concat 라고 부른다.

일반적으로 newfs(8)은 디스크 이름을 해석해서 해석하지 못하면 메시지를 보여준다. 예를 들면 다음과 같은 메시지를 보여준다:

```
# newfs /dev/vinum/concat
newfs: /dev/vinum/concat: can't figure out file system partition
```

**Note:** 다음은 FreeBSD 5.0 이전 버전에서만 유효하다:

이 볼륨에 파일시스템을 생성하려면 newfs(8)에 `-v` 옵션을 사용한다:

```
# newfs -v /dev/vinum/concat
```

## 17.8 Vinum 설정

GENERIC 커널은 Vinum 을 포함하지 않는다. Vinum 을 포함하는 특별한 커널을 빌드할 수 있지만 권장하지 않는다. Vinum 을 시작하는 표준적인 방법은 커널 모듈이다(kld). 그리고 Vinum 에 kldload(8)을 사용할 필요는 없다: vinum(8)을 시작할 때 모듈이 로드 되었는지 체크해서 안되었다면 자동으로 로드한다.

### 17.8.1 Startup

Vinum 은 설정파일과 같은 형식으로 디스크 슬라이스에 설정정보를 저장한다. 설정 데이터 베이스에서 읽을 때 Vinum 은 설정파일에서 허용하지 않는 키워드 번호를 인지한다. 예를 들면 디스크 설정은 다음과 같은 텍스트를 포함할 것이다:

---

```
volume myvol state up
volume bigraid state down
plex name myvol.p0 state up org concat vol myvol
plex name myvol.p1 state up org concat vol myvol
plex name myvol.p2 state init org striped 512b vol myvol
plex name bigraid.p0 state initializing org raid5 512b vol bigraid
sd name myvol.p0.s0 drive a plex myvol.p0 state up len 1048576b driveoffset 265b
plexoffset 0b
sd name myvol.p0.s1 drive b plex myvol.p0 state up len 1048576b driveoffset 265b
plexoffset 1048576b
sd name myvol.p1.s0 drive c plex myvol.p1 state up len 1048576b driveoffset 265b
plexoffset 0b
sd name myvol.p1.s1 drive d plex myvol.p1 state up len 1048576b driveoffset 265b
plexoffset 1048576b
sd name myvol.p2.s0 drive a plex myvol.p2 state init len 524288b driveoffset
1048841b plexoffset 0b
sd name myvol.p2.s1 drive b plex myvol.p2 state init len 524288b driveoffset
1048841b plexoffset 524288b
sd name myvol.p2.s2 drive c plex myvol.p2 state init len 524288b driveoffset
1048841b plexoffset 1048576b
sd name myvol.p2.s3 drive d plex myvol.p2 state init len 524288b driveoffset
1048841b plexoffset 1572864b
sd name bigraid.p0.s0 drive a plex bigraid.p0 state initializing len 4194304b driveoff
set 1573129b plexoffset 0b
sd name bigraid.p0.s1 drive b plex bigraid.p0 state initializing len 4194304b driveoff
set 1573129b plexoffset 4194304b
sd name bigraid.p0.s2 drive c plex bigraid.p0 state initializing len 4194304b driveoff
set 1573129b plexoffset 8388608b
sd name bigraid.p0.s3 drive d plex bigraid.p0 state initializing len 4194304b driveoff
set 1573129b plexoffset 12582912b
sd name bigraid.p0.s4 drive e plex bigraid.p0 state initializing len 4194304b driveoff
set 1573129b plexoffset 16777216b
```

이곳에서 확실한 차이는 위치 정보 표시, 이름 할당(둘 다 허용되지만 사용자가 사용하지는 못한다) 그리고 상태 정보다. Vinum 은 드라이브에 대한 정보를 설정정보에 저장하지 않는다: Vinum 레벨로 파티션하기 위해 설정된 디스크 드라이브를 스캔하여 드라이브를 찾는다.

---

따라서 다른 유닉스 드라이브 ID 를 할당 받았더라도 Vinum 이 드라이브를 정확히 인지하게 된다.

### 17.8.1.1 자동 시작

시스템이 부팅할 때 Vinum 을 자동으로 시작하려면 /etc/rc.conf 에 다음 라인을 추가해야 된다:

```
start_vinum="YES"      # set to YES to start vinum
```

/etc/rc.conf 파일이 없다면 이 내용을 입력하여 생성한다. 그래서 시작할 때 Vinum kld 를 시스템에 로드하고 설정에 참조된 오브젝트를 시작한다. 그리고 파일시스템을 마운트하기 전에 끝나서 자동으로 fsck(8)를 수행하고 Vinum 볼륨에 파일시스템을 마운트할 수 있다.

**vinum start** 명령으로 Vinum 을 시작할 때 Vinum 은 하나의 Vinum 드라이브에서 설정 데이터베이스를 읽는다. 일반적인 상황에서 각 드라이브는 설정 데이터베이스의 동일한 복사본을 가지고 있다. 그래서 어떤 드라이브를 읽더라도 문제가 되지 않는다. 그러나 문제가 발생한 후 Vinum 은 어떤 드라이브가 가장 최근에 업데이트되었고 이 드라이브에서 설정을 읽었는지 확인해야 된다. 필요하다면 진행하기 전의 예전 드라이브에서 설정을 업데이트해야 된다.

## 17.9 Root 파일시스템에 Vinum 사용하기

머신의 파일시스템을 Vinum 으로 완벽하게 미러하기 위해 root 파일시스템도 미러해야 된다. 설정은 임의적인 파일시스템에 비해 다음과 같은 이유로 약간 복잡하다:

- root 파일시스템은 부팅하는 동안 사용해야 되기 때문에 Vinum 의 인프라스트럭처를 부팅하기 전에 사용할 수 있어야 된다
- Vinum 의 자세한 사항에 대하여 이해하지 못하기 때문에 root 파일시스템을 가지고 있는 볼륨은 기본 유틸리티를(예: PC 급 머신에서 BIOS) 사용하여 읽을 수 있는 부트스트랩과 커널도 가지고 있어야 한다.

---

이번 섹션의 "root 볼륨"이라는 용어는 일반적으로 root 파일시스템을 포함하는 Vinum 볼륨을 의미한다. 이 볼륨에 "root"라는 이름을 사용하는 것이 좋은 생각이지만 어쩌든 기술적으로 필요한 것은 아니다. 이번 섹션에서 모든 예제 명령은 이 이름이라고 가정한다.

## 17.9.1 root 파일시스템을 위해 Vinum 빨리 실행하기

이 사항을 측정할 수 있는 몇 가지 방법이 있다:

- Vinum 은 부팅할 때 커널에서 이용할 수 있어야 한다. 그래서 17.8.1.1 장에서 설명한 Vinum 을 자동으로 시작하는 방법은 여기에 적용할 수 없고 다음 설정이 준비되지 않았을 때 `start_vinum` 매개변수는 실제로 설정되지 않는다. 첫 번째 옵션은 Vinum 을 커널에 정적으로 컴파일 한다. 그래서 아무 때나 이용할 수 있지만 일반적으로 바람직하지 않다. 커널이 시작되기 전에 `/boot/loader` 가(12.3.3 장) `vinum` 을 커널 모듈에 로드하는 옵션도 있다. 이것은 다음 라인을 `/boot/loader.conf` 에 넣으면 된다.

```
vinum_load="YES"
```

- root 파일시스템에 볼륨을 제공해야 되기 때문에 Vinum 은 이전에 초기화되어야 한다. 기본적으로 Vinum 커널 부분은 관리자가(또는 시작 스크립트 중 하나) `vinum start` 명령을 사용할 때까지 Vinum 볼륨 정보를 가지고 있는 드라이브를 찾지 않는다.

**Note:** 다음 단락은 FreeBSD 5.x 와 이 후 버전에 필요한 단계의 요점이다. FreeBSD 4.x 의 설정은 달라서 17.9.5 장에서 설명한다.

Vinum 이 커널 시작 부분에서 Vinum 정보를 검색하기 위해 모든 드라이브를 자동으로 탐색하도록 다음 라인을 `/boot/loader.conf` 에 입력한다.

```
vinum.autostart="YES"
```

커널에 root 파일시스템을 찾는 곳을 지정할 필요는 없다. `/boot/loader` 는 `/etc/fstab` 에서

---

root 장치 이름을 검색해서 이 정보를 커널에 보낸다. root 파일시스템을 마운트할 때 커널은 제공된 장치 이름에서 어떤 드라이버가 내부 장치 ID 로(major/minor 번호) 변환될지 결정한다.

## 17.9.2 Vinum 기반 root 볼륨이 부트스트랩에 접근할 수 있도록 만들기

현재 FreeBSD 부트스트랩은 7.5KB의 코드뿐이고 미리 UFS 파일시스템에서 파일을 읽어야 되기 때문에 Vinum 내부 구조에 대하여 알려주는 것은 불가능하다. 이 문제는 Vinum 설정 데이터를 해석하여 부트 볼륨 요소에 대해 해결할 수 있다. 그래서 root 파일시스템을 포함하고 있는 표준 "a" 파티션을 착각하도록 하는 부트스트랩 코드를 제공하기 위해 몇 가지 기교가 필요하다.

이러한 모든 문제를 해결하기 위해 root 볼륨을 다음과 같이 설정해야 된다:

- root 볼륨을 스트라이프 하거나 RAID-5로 설정하지 않는다.
- root 볼륨은 플렉스 별로 하나 이상의 연속된 서브디스크를 포함하지 않는다.

각각 root 파일시스템 복사본을 하나씩 가지고 있는 여러 개의 플렉스가 바람직하고 가능하다. 그러나 부트스트랩 프로세스는 결국 커널이 root 파일시스템을 마운트할 때까지 부트스트랩과 모든 파일을 찾을 때 이들 복사본 중 하나만 이용한다. 이들 플렉스에 있는 싱글 서브디스크는 각각의 장치가 부팅할 수 있도록 "a" 파티션의 일루전(illusion)을 가지고 있어야 된다. root 볼륨 플렉스를 가지고 있는 다른 장치와 비교해서 이들 거짓(faked) "a" 파티션이 장치의 같은 옵셋에 있을 필요는 없다. 그러나 혼동되지 않게 미러된 장치가 대칭되도록 이런 식으로 Vinum 볼륨을 생성하는 것도 좋은 아이디어다.

각 장치들이 root 볼륨을 포함하도록 "a" 파티션을 설정하기 위해 다음과 같은 사항이 필요하다:

- ① root 볼륨의 일부인 이 장치의 서브디스크 위치와 크기를 확인하기 위해 다음 명령을 사용한다.

---

```
# vinum l -rv root
```

Vinum 옵션과 크기는 바이트(bytes) 단위를 사용한다. `disklabel` 명령에 사용할 블록 번호를 알기 위해 이들을 512로 나뉘어야 한다.

- ② root 볼륨과 연관된 각 장치를 위해 다음 명령을 실행한다.

```
# disklabel -e devname
```

*devname*은 슬라이스 테이블이 없는 디스크 이름이거나(da0 처럼) 슬라이스 이름이다(ad0s1 과 같은).

장치에(아마도 이미 Vinum root 파일시스템을 가지고 있는) "a" 파티션이 있다면 다른 이름을 할당해서 접근이 가능 하지만(이 경우) 시스템의 부트스트랩은 기본적으로 더 이상 사용하지 않는다. 활성화된 파티션에(현재 마운트된 root 파일시스템과 같은) 다시 이름을 할당할 수 없기 때문에 "Fixit" 중간이나(미러된 상황에서) 현재 디스크가 부트되지 않고 초기화되는 두 번째 단계에서 실행해야 된다.

이 장치에 있는 Vinum 파티션 옵션을 이 장치에 있는 각 root 볼륨 서브디스크의 옵션에 추가해야 된다. 결과 값은 새로운 "a" 파티션의 "offset" 값이 된다. 이 파티션의 "size" 값은 위의 계산에서 예상할 수 있다. "fstype"는 4.2BSD 다. 이 문장에서 별로 중요하지는 않더라도 "fsize", "bsize"와 "cpg" 값은 실제 파일시스템에 맞는 선택을 한다.

따라서 새로운 "a" 파티션은 이 장치의 Vinum 파티션과 일치된다. Vinum 파티션이 "vinum" 파일시스템 타입을 사용하도록 정확히 표시되어 있다면 `disklabel`은 이렇게 일치하는 곳만 허용한다.

- ③ 이제 거짓 "a" 파티션은 root 볼륨의 복사본을 하나 가지고 있는 각 장치에 있다. 다음과 같은 명령을 이용하여 결과를 다시 확인해야 된다.

```
# fsck -n /dev/devnamea
```

---

새로운 Vinum root 볼륨을 설정할 때, 현재 활성화된 root 파일시스템과 매칭되지 않을 수 있는 Vinum 볼륨에서 제어 정보를 가지고 있는 모든 파일들은 root 파일시스템과 관련이 있다. 그래서 /etc/fstab 와 /boot/loader.conf 파일은 특히 주의해야 된다.

따라서 다음에 재 부팅할 때 부트 스트랩은 새로운 Vinum 기반 root 파일시스템에서 적절한 제어정보를 해석한다. 모든 장치가 표시된 후 커널 초기화 프로세스의 마지막에서 이 설정이 성공했음을 보여주는 다음과 같은 메시지가 나타난다:

```
Mounting root from ufs:/dev/vinum/root
```

### 17.9.3 Vinum 기반 root 설정 예제

Vinum root 볼륨을 설정한 후 `vinum l -rv root` 의 출력은 다음과 비슷할 것이다:

```
...
Subdisk root.p0.s0:
    Size:          125829120 bytes (120 MB)
    State: up
    Plex root.p0 at offset 0 (0 B)
    Drive disk0 (/dev/da0h) at offset 135680 (132 kB)

Subdisk root.p1.s0:
    Size:          125829120 bytes (120 MB)
    State: up
    Plex root.p1 at offset 0 (0 B)
    Drive disk1 (/dev/da1h) at offset 135680 (132 kB)
```

offset(파티션 /dev/da0h 에 관련된) 값은 135680이다. disklabel 의 용어에서 이 값은 265에서 512 바이트로 디스크 블록을 변환하는 것이다. 역시 이 root 볼륨의 크기는 245760 512-바이트 블록이다. 이 볼륨의 두 번째 복사본을 가지고 있는 /dev/da1h 는 대칭적인 설정을 가지고 있다.

이들 장치의 디스크 라벨은 다음과 비슷할 것이다:



...

8 partitions:

#	size	offset	fstype	[fsize bsize bps/cpg]				
a:	245760	281	4.2BSD	2048 16384	0	# (Cyl.	0*- 15*)	
c:	71771688	0	unused	0 0		# (Cyl.	0 - 4467*)	
h:	71771672	16	vinum			# (Cyl.	0*- 4467*)	

거짓 "a" 파티션이 위 개요의 값과 일치하도록 "size" 매개변수를 보존할 수 있지만 "offset" 매개변수는 Vinum 파티션 "h"의 오프셋과 장치에 있는 이 파티션의(또는 슬라이스) 오프셋을 더한 값이다. 이것은 17.9.4.3 장에서 설명한 문제를 피하기 위해 필요한 전형적인 설정이다. "a" 파티션 전체는 이 장치의 모든 Vinum 데이터를 가지고 있는 파티션 "h"에 있음을 볼 수 있다.

위의 예제에서 장치 전체가 Vinum 전용이라면 Vinum 설정의 일부분이 되는 새로운 디스크를 설정했기 때문에 이전 Vinum root 파티션은 남아있지 않는다.

## 17.9.4 문제 해결

문제가 있다면 이 상황을 해결할 수 있는 방법이 있다. 다음 리스트는 몇 가지 잘 알려진 실수에 대한 해결책을 제시한다.

### 17.9.4.1 부트 스트랩은 로드하지만 시스템이 부팅하지 않는다.

어떤 이유로 시스템이 부팅하지 않는다면 10 초를 기다리는 동안 **space** 키를 눌러서 부트 스트랩을 중지할 수 있다. 로더 변수는(*vinum.autostrt*와 같은) **show** 를 사용하여 검사할 수 있고 **set** 이나 **unset** 명령으로 제어할 수 있다.

자동으로 로드되는 모듈 리스트에 Vinum 커널 모듈이 없다면 간단히 **load vinum** 이 도움을 될 수 있다.

준비가 된 후 **boot -as** 로 부트 프로세스를 계속할 수 있다. 옵션 **-as** 는 커널이 root 파일 시스템을 마운트(-a) 하도록 요청하고 root 파일시스템이 읽기 전용으로 마운트되는 싱글 유저 모드에서(-s) 부트 프로세스를 중지시킨다. 그 결과 멀티 플렉스 볼륨 중 하나의 플렉스만 마운트 되더라도 플렉스 사이의 데이터가 일치하지 않는 것은 위험하다.

---

root 파일시스템을 마운트하도록 요청하는 단계에서 정확한 root 파일시스템을 가지고 있는 장치로 들어갈 것이다. /etc/fstab 가 정확하게 설정되어 있다면 기본값은 `ufs:/dev/vinum/root`와 비슷하다. 일반적인 다른 선택은 이전 Vinum root 파일시스템을 포함하는 가상 파티션 `ufs:da0d`와 비슷할 것이다. Vinum root 장치의 서브디스크를 실제로 참조하는 엘리어스 "a" 파티션 중 하나가 이곳에 있다면 미러된 설정에서 미러된 root 장치의 일부만 마운트하기 때문에 주의한다. 이 파일시스템이 읽기와 쓰기로 마운트 되었다면 이들 플렉스는 일치하지 않는 데이터를 전송하기 때문에 Vinum root 볼륨의 다른 플렉스를 삭제해야 된다.

#### 17.9.4.2 메인 부트 스트랩만 로드한다.

/boot/loader 를 로드하는데 실패하지만 메인 부트스트랩이 로드되었다면(부트 프로세스가 시작된 후 화면 우측의 칼럼 왼쪽에서 - 문자를 볼 수 있다면) 이 부분에서 **space** 키를 사용하여 메인 부트 스트랩을 중지할 수 있다. 이 방법으로 두 번째 단계에서 부트스트랩을 멈추고 12.3.2 장을 본다. 여기서 위의 "a"에서 삭제된 이전 root 파일시스템을 가지고 있는 파티션과 다른 파티션에서 부팅하는 것을 취소할 수 있다.

#### 17.9.4.3 부팅하지 않고 부트 스트랩이 패닉 된다.

이 상황은 Vinum 을 설치하여 부트스트랩이 삭제된 경우에 발생한다. 불행히 Vinum 은 현재 Vinum 헤더 정보를 작성하기 전에 여유 파티션의 시작부분에 4KB 만 남길 수 있다. 그러나 첫 번째에서 두 번째 부트 스트랩으로 넘어가는 과정에서 디스크 라벨을 할당하기 위해 8KB 가 필요하다. 따라서 Vinum 파티션이 부팅 가능한 슬라이스나 디스크의 오프셋 0 에서 시작되었다면 Vinum 설정은 부트스트랩을 삭제한다.

비슷한 예를 들어 "Fixit" 중간 단계에서 위의 상황을 복구했고 12.3.2 장에서 설명했듯이 **disklabel -B** 를 사용하여 부트스트랩을 다시 설치했다면, 부트 스트랩이 Vinum 헤더를 삭제하여 Vinum 은 더 이상 디스크를 찾지 않는다. 실제 Vinum 설정 데이터가 아니라면 Vinum 볼륨의 데이터가 삭제되더라도 똑같은 Vinum 설정 데이터로 다시 들어가서 모든 데이터를 복구할 수 있지만 상당히 어렵다. Vinum 헤더와 시스템 부트 스트랩 충돌을 막기 위해 최소 4kb 만 남기고 전체 Vinum 파티션을 이동시켜야 된다.

---

## 17.9.5 FreeBSD 4.x 와 차이

FreeBSD 4.x 에서는 빠진 디스크를 Vinum 이 자동으로 탐색하도록 몇 가지 내부 기능이 필요하고, root 장치의 내부 ID 를 확인하기 위한 코드는 /dev/vinum/root 와 같은 이름을 자동으로 제어하기에 적절하지 않다. 그 결과 약간의 차이점이 있다.

Vinum 은 /boot/loader.conf 에 다음과 같은 라인을 사용하여 탐색할 디스크를 지정해야 한다:

```
vinum.drives="/dev/da0 /dev/da1"
```

모든 드라이브가 Vinum 데이터를 포함할 수 있다고 지정하는 것도 중요하다. 더 많은 드라이브를 나열하더라도 문제없고, Vinum 은 유효한 Vinum 헤더에 이름이 할당된 드라이브의 모든 슬라이스와 파티션을 탐색하기 때문에 각 슬라이스와 파티션을 추가할 필요도 없다.

root 파일시스템의 이름을 해석하는데 사용되는 루틴과 파생된 장치 ID(major/minor 번호) 는 /dev/ad0s1a 와 같은 "예전" 장치 이름만 제어하도록 준비되었기 때문에 /dev/vinum/root 와 같은 root 볼륨 이름은 처리하지 못한다. 이유는 Vinum 이 초기화하는 동안 root 장치의 ID 를 갖는 내부 커널 매개변수의 초기 설정이 필요하기 때문이다. 이 초기 설정은 로더 변수 *vinum.root* 에 root 볼륨의 이름을 전달해서 요청한다. 이러한 것이 가능하도록 /boot/loader.conf 엔트리는 다음과 비슷할 것이다:

```
vinum.root="root"
```

이제 커널이 초기화되는 동안 마운트하기 위해 root 장치를 찾을 때 몇몇 커널 모듈이 커널 매개변수로 미리 초기화되었는지 확인한다. 초기화되었고 장치가 root 장치의 문자열에 적용한 것을(우리의 경우 "vinum") 해석한 것과 드라이버의 주 번호와 root 장치가 매칭된다고 하면, 자체적으로 해결하지 않고 미리 할당된 장치 ID 를 사용한다. 그래서 자동으로 시작하는 동안 root 파일시스템에 Vinum root 볼륨을 마운트할 수 있다.

그러나 boot -a 가 root 장치의 이름을 수동으로 입력하도록 할 때 이 루틴은 Vinum 볼륨을 참조하는 이름을 아직 실제로 해석할 수 없다. Vinum 장치를 참조하지 않은 장치 이름을 입력했다면 이미 할당된 root 매개변수와 주어진 이름에서 해석된 드라이버의 주 번호 사이의 불일치는, 이 루틴을 일반적인 해석기로 만들기 때문에 ufs:da0d 처럼 입력한 문자열은

---

정확하게 동작한다. 이 것이 실패한다면 해석할 수 없기 때문에 ufs:vinum/root 문자열을 다시 입력하는 것이 불가능하다. 유일한 방법은 다시 부팅해서 위의 내용을 다시 시작하는 것이다("askroot" 프롬프트에서 최초의 /dev/는 항상 생략할 수 있다).

---

## 18 장 지역화 - I18N/L10n 사용과 설정

### 18.1 개요

FreeBSD 는 유저와 공헌자들이 전 세계에 분포되어 있는 매우 분산된 프로젝트다. 이번 장에서는 영어를 사용하지 않는 유저들이 실제 작업에 사용할 수 있도록 FreeBSD 의 국제화와 지역화 기능에 대해 설명한다. 시스템과 어플리케이션 레벨에서 사용할 수 i18n 의 다양한 기능이 있기 때문에 특정 소스의 문서를 참고하도록 한다.

이번 장을 읽고 다음과 같은 사항을 알 수 있다:

- 다른 언어와 지역 설정이 어떻게 현대 운영체제에 코드화되었는가
- 로그인 쉘의 지역 설정은 어떻게 지정하는가
- 영어가 아닌 다른 언어로 어떻게 콘솔을 설정하는가
- X 윈도우에서 다른 언어를 효과적으로 사용하기
- i18n을 따르는 어플리케이션 작성에 대한 더 많은 정보는 어디서 찾을 수 있는가

이번 장을 읽기 전에 다음 사항을 알고 있어야 한다:

- 어플리케이션을 어떻게 설치하는지 알고 있어야 한다(4장)

### 18.2 기본 정보

#### 18.2.1 I18N/L10N 은 무엇인가?

개발자들은 국제화를 국제화의(internationalization) 첫 번째 문자부터 마지막 문자를 계산하여 I18N 이라는 짧은 용어로 사용한다. L10N 도 같은 분류 방법을 사용하여 지역화(localization)에서 나온 이름이다. I18N/L10N 설정과 프로토콜을 조합하여 유저들이 선택한 언어로 어플리케이션을 사용할 수 있다.

I18N 어플리케이션은 라이브러리에서 I18N 키트를 사용하도록 프로그램 되어있다. 따라서 개발자들은 간단한 파일을 작성하여 표시되는 메뉴와 텍스트를 각 언어로 변환할 수 있다. 우리는 프로그래머가 이 전통을 따르도록 권장하고 있다.

---

## 18.2.2 왜 I18N/L10N 을 사용해야 되는가?

I18N/L10N 은 영어가 아닌 다른 언어에서 데이터를 보고, 입력, 진행 시킬 때 사용된다.

## 18.2.3 I18N 의 노력으로 어떤 언어가 지원되는가?

I18N 과 L10N 은 FreeBSD 의 기능이 아니다. 현재 중국, 독일, 일본, 한국, 프랑스, 베트남과 더 많은 국가들이 포함되어 있어서 세계에서 가장 주된 언어를 선택할 수 있다

## 18.3 지역화 사용

I18N 은 FreeBSD 의 기능이 아닌 관례다. FreeBSD 가 이러한 관례를 따를 수 있도록 여러분의 도움을 기대하고 있다.

지역화 설정은 3 가지 주된 용어를 바탕으로 한다: 언어 코드, 국가 코드 그리고 인코딩이다. 지역 이름은 다음과 같은 부분으로 되어있다:

`LanguageCode_CountryCode.Encoding`

### 18.3.1 언어와 국가 코드

FreeBSD 시스템을 특정 언어로(또는 유닉스를 지원하는 다른 I18N 으로) 지역화하려면 특정 국가와 언어(국가 코드는 어플리케이션에게 사용할 언어변수를 알려준다) 코드를 찾아야 한다. 게다가 웹 브라우저, SMTP/POP 서버, 웹 서버 등 지역화를 설정할 어플리케이션도 결정한다.

다음은 언어/국가 코드의 예제다:

언어/국가 코드	설명
en_US	영어 - 미국
ru_RU	러시아인을 위한 러시아어

## 18.3.2 인코딩

8-비트 방식의 ASCII 인코딩을 사용하지 않고 다양한 멀티바이트 문자를 사용하는 몇몇 언어에 대한 더 자세한 사항은 `multibyte(3)`를 본다. 오래된 어플리케이션은 이들을 인지하지 못해서 문자 제어에 오류가 있다. 새로운 어플리케이션은 보통 8 비트 문자를 인식한다. 용도에 따라 유저는 어플리케이션을 다양한 멀티 바이트 문자를 지원하도록 컴파일해야 될 필요가 있거나 정확하게 설정해야 될 것이다. 다양한 멀티바이트 문자를 입력하고 진행할 수 있도록 FreeBSD 포트 컬렉션에서(<http://www.freebsd.org/ports/index.html>) 각 언어와 프로그램을 제공하고 있다. 각 FreeBSD 포트에서 I18N 문서를 참고한다.

특히 유저는 정확한 설정이나 `configure/Makefile/compiler` 에 적용하는 정확한 값을 알기 위해 어플리케이션 문서를 찾아봐야 한다.

다음과 같은 몇 가지를 주의해야 된다:

- 언어에 특별한 싱글 C chars 문자 세트(`multibyte(3)`를 본다). 예: ISO-8859-1, ISO-8859-15, KO18-R, C437.
- 다양한 멀티바이트 인코딩. 예: EUC, Big5.

문자 세트의 변경 리스트를 IANA 레지스트리(<http://www.iana.org/assignments/character-sets>)에서 체크할 수 있다.

**Note:** FreeBSD 버전 4.5 와 새로운 버전은 X11 지역 호환 인코딩을 대신 사용한다.

## 18.3.3 I18N 어플리케이션

FreeBSD 포트와 패키지 시스템에서 I18N 어플리케이션은 확인하기 쉽게 I18N 이라고 적혀있다. 그러나 필요한 언어를 항상 지원하는 것은 아니다.

---

## 18.3.4 지역 설정

일반적으로 로그인 셸의 LANG 에 지역 이름 변수를 지정하는 것으로 충분하다. 이러한 설정을 유저의 ~/.login\_conf 파일이나 유저 셸의(~/.profile, ~/.bashrc, ~/.cshrc) 시작 파일에 넣으면 된다. LC\_CTYPE, LC\_CTME 처럼 지역에 부분적인 설정까지는 필요 없다. 더 많은 정보는 FreeBSD 의 언어관련 문서를 참고한다.

설정파일에 다음 두 환경 변수를 설정해야 된다:

- POSIX setlocal(3) 관련 기능은 LANG 설정
- 어플리케이션의 MIME 문자 세트는 MM\_CHARSET를 설정

이것은 유저 셸 설정, 특정 어플리케이션 설정 그리고 X11 설정을 포함한다.

### 18.3.4.1 지역 설정 방법

지역을 설정하는 두 가지 방법이 있고 아래에서 설명할 것이다. 첫 번째는(권장사항) 로그인 클래스에 환경변수를 할당하는 것이고 두 번째는 시스템의 셸 시작 파일에 환경 변수 할당을 추가하는 것이다.

[예제: 지역 설정하기]

#### 1 로그인 클래스를 사용하는 방법

이 방법은 필요한 사항을 각 셸 시작 파일에 할당하지 않고 모든 셸에 로컬 이름과 MIME 문자를 설정하는 환경 변수를 한번에 지정할 수 있다. 유저 레벨설정은 유저가 직접 지정할 수 있고 관리자 레벨 설정은 슈퍼 유저 권한이 있어야 한다.

##### 1.1 유저 레벨 설정

Latin-1 인코딩을 설정하기 위해 두 개의 변수 설정을 가진 유저 홈 디렉터리의 .login\_conf 파일 예제가 있다:



```
me:₩
:charset=ISO-8859-1:₩
:lang=de_DE.ISO8859-1:
```

그리고 전통적인 중국 BIG-5 인코딩 변수를 설정한 .login\_conf 의 예제가 있다. 어떤 소프트웨어는 중국어, 일본어와 한국어에 맞는 지역 변수를 정확하게 고려하지 않기 때문에 더 많은 변수 설정에 주의해야 된다.

```
#Users who do not wish to use monetary units or time formats
#of Taiwan can manually change each variable
me:₩
lang=zh_TW.Big5:₩
lc_all=zh_TW.Big5:₩
lc_collate=zh_TW.Big5:₩
lc_ctype=zh_TW.Big5:₩
lc_messages=zh_TW.Big5:₩
lc_monetary=zh_TW.Big5:₩
lc_numeric=zh_TW.Big5:₩
lc_time=zh_TW.Big5:₩
charset=big5:₩
xmodifiers="@im=xcin": #Setting the XIM Input Server
```

더 자세한 사항은 관리자급 설정과 login.conf(5)를 본다.

## 1.2 관리자급 설정

유저의 로그인 클래스가 정확한 언어로 설정되었는지 /etc/login.conf 에서 확인한다. 다음 설정이 /etc/login.conf 에 있는지 확인한다:

```
language_name:accounts_title:₩
:charset=MIME_charset:₩
:lang=locale_name:₩
:tc=default:
```

그래서 Latin-1 을 사용하는 이전 예제는 다음과 같다:

```
german:German Users Accounts:W
:charset=ISO-8859-1:W
:lang=de_DE.ISO8859-1:W
:tc=default:
```

### 1.2.1 vipw(8)로 로그인 클래스 변경

새로운 유저를 추가하려면 vipw 를 사용하여 다음과 같은 엔트리를 생성한다:

```
user:password:1111:11:language:0:0:User Name:/home/user:/bin/sh
```

### 1.2.2 adduser(8)로 로그인 클래스 변경

새로운 유저를 추가하려면 adduser 를 사용하고 다음 사항을 따른다:

- /etc/adduser.conf에 *defaultclass = language*를 설정한다. 이 경우 다른 언어를 사용하는 모든 유저를 위해 *default* 클래스를 명시해야 된다.
- 다른 방법은 adduser(8)에서 지정된 언어가 나타나도록 한다

```
Enter login class: default []:
```

- 또 다른 방법은 다른 언어를 사용하려는 유저를 위해 다음 명령을 사용한다:

```
# adduser -class language
```

### 1.2.3 pw(8)로 로그인 클래스 변경

새로운 유저를 추가하기 위해 pw(8)를 사용한다면 다음과 같은 형식으로 사용한다.

```
# pw useradd user_name -L language
```

---

## 2 셸 시작 파일을 사용하는 방법

**Note:** 선택 가능한 각 셸 프로그램 별로 다른 설정이 필요하기 때문에 이 방법은 권장하지 않는다. 대신 로그인 클래스를 사용하도록 한다.

지역 이름과 MIME 문자 세트를 추가하려면 아래에서 보여주는 두 개의 환경 변수를 `/etc/profile` 과 `/etc/csh.login` 셸 시작 파일에 설정한다. 아래 예제에서는 독일어를 사용한다:

`/etc/profile` 에 다음 내용을 추가한다:

```
LANG=de_DE.ISO8859-1; export LANG
MM_CHARSET=ISO-8859-1; export MM_CHARSET
```

아니면 `/etc/csh.login` 에 다음 내용을 입력한다:

```
setenv LANG de_DE.ISO8859-1
setenv MM_CHARSET ISO-8859-1
```

그렇지 않고 위의 내용을 `/usr/share/shel/dot.profile`(위의 `/etc/profile` 과 비슷한)에 추가하거나 `/usr/share/skel/dot.login`(위의 `/etc/csh.login` 과 비슷한)에 추가할 수 있다.

X11 설정은 `$HOME/.xinitrc` 에 다음 내용을 추가한다:

```
LANG=de_DE.ISO8859-1; export LANG
```

또는 다음 명령을 사용한다:

```
# setenv LANG de_DE.ISO8859-1
```

---

## 18.3.5 콘솔 설정

모든 싱글 C chars 문자 세트는 `/etc/rc.conf` 에 다음 내용으로 정확한 콘솔 폰트를 설정한다:

```
font8x16=font_name
font8x14=font_name
font8x8=font_name
```

여기서 *font\_name* 은 `/usr/share/syscons/fonts` 디렉터리에서 `.fnt` 확장자를 제외하고 가져온 것이다.

또한 싱글 C chars 문자 세트는 `/stand/sysinstall` 로 정확한 키 맵과 스크린 맵을 설정한다. `sysinstall` 에서 **Configure** 를 선택하고 **Console** 를 선택한다. 아니면 `/etc/rc.conf` 에 다음 내용을 추가할 수 있다:

```
scrnmap=screenmap_name
keymap=keymap_name
keychange="fkey_number sequence"
```

이곳의 *screenmap\_name* 은

`/usr/share/syscons/scrnmpas` 디렉터리에서 `.scm` 확장자를 제외하고 가져온 것이다. 맵 폰트와 맞는 스크린 맵은 일반적으로 pseudographics 영역의 VGA 아답터 폰트 문자 매트릭스를 8 bit 에서 9bit 로 확장하기 위해 필요하다. 예를 들어 스크린 폰트를 8bit 칼럼으로 사용한다면 이 영역 밖으로 문자를 옮긴다.

`/etc/rc.conf` 의 **moused** 데몬을 다음 설정으로 활성화했다면 다음 단락에서 마우스 커서 정보에 대해 생각해본다:

```
moused_enable="YES"
```

기본적으로 마우스 커서 `syscons(4)` 드라이버는 문자 세트의 `0Xd0-0xd3` 범위를 사용한다. 여러분의 언어가 이 범위를 사용한다면 커서의 범위를 다른 곳으로 이동시켜야 한다. FreeBSD 5.0 이전 버전에서 커서 범위를 이동시키려면 커널 설정에 다음 라인을 추가한다:

---

options	SC_MOUSE_CHAR=0x03
---------	--------------------

FreeBSD 버전 4.4 와 이전 버전은 다음 라인을 /etc/rc.conf 에 넣는다:

mousechar_start=3
-------------------

이곳의 *keymap\_name* 은 /usr/share/syscons/keymaps 디렉터리에서 .kbd 를 제외하고 가져 온 것이다. 어떤 키 맵을 사용할지 확실하지 않다면 재 부팅 없이 키 맵을 테스트하는 kdbmap(1)을 사용할 수 있다.

기능키 순서를 키 맵에 정의할 수 없기 때문에 터미널 타입에 프로그램 기능키가 맞도록 *keychange* 가 보통 필요하다.

그리고 정확한 콘솔 터미널 종류도 /etc/ttys 의 모든 *ttym\** 엔트리에 설정해야 된다. 현재 미리 설정되어 있는 것은 아래와 같다:

문자 세트	터미널 종료
ISO-8859-1 또는 ISO-8859-15	cons25i1
ISO-8859-2	cons25i2
ISO-8859-7	cons25i7
KOI8-R	cons25r
KOI8-U	cons25u
CP437 (VGA default)	cons25
US-ASCII	cons25w

다양한 멀티바이트 문자에 맞도록 /usr/ports/language 디렉터리의 정확한 FreeBSD 포트를 사용한다. 시스템이 시리얼 vty 의 폰트를 확인하는 동안 어떤 포트가 콘솔에 나타나기 때문에 X11 과 pseudo 시리얼 콘솔을 위해 vty 를 충분히 남겨둬야 한다. 다음은 콘솔에서 다른 언어를 사용하는 어플리케이션을 위한 리스트다:

언어	위치
----	----

---

전통적인 중국어(BIG-5)	chinese/big5con
일어	japanese/ja-kon2-* 또는 japanese/Mule_Wnn
한국어	Korea/ko-han

### 18.3.6. X11 설정

X11 이 FreeBSD 프로젝트의 일부가 아니더라도 이곳에 FreeBSD 유저를 위한 몇 가지 정보를 기재하였다. 더 자세한 사항은 XFree86 웹 사이트(<http://www.xfree86.org>) 또는 사용중인 X11 서버를 참고한다.

~/Xresources 에 어플리케이션의 특별한 i18N 설정을(다시 말해서 폰트와 메뉴 등) 추가할 수 있다.

#### 18.3.6.1 폰트 표시

X11 투루 타입 일반 서버를(x11-servers/XtXF86srv-common) 설치하고 투루 타입 언어를 설치한다. 정확한 지역을 설정하면 선택한 언어로 메뉴 같은 것을 볼 수 있다.

#### 18.3.6.2 비 영문자 입력

X11 입력 메소드(XIM) 프로토콜은 모든 X11 클라이언트에 새로운 표준이다. 모든 X11 어플리케이션은 XIM 입력 서버로 입력을 받도록 XIM 클라이언트로 작성해야 된다. 다른 언어로 사용할 수 있는 몇 개의 XIM 서버가 있다.

### 18.3.7 프린터 설정

어떤 싱글 C chars 문자 세트는 프린터에 보통 입력되어 있다. 멀티바이트 문자 세트는 특별한 설정이 필요하기 때문에 **apsfilter** 를 사용하도록 한다. 특정 언어 변환기를 사용하여 포스트스크립트나 PDF 포맷으로 문서를 변환할 수 있을 것이다.

---

## 18.3.8 커널과 파일시스템

FreeBSD fast 파일시스템은(FFS) 8 비트기 때문에 어떠한 싱글 C chars 문자 세트도(multibyte(3)를 본다) 사용할 수 있지만 문자 세트 이름이 파일시스템에 저장되어 있지 않다; 즉 raw 8-비트기 때문에 인코딩 순서에 대해 전혀 알지 못한다. FFS 는 아직 멀티바이트 문자 세트 형식을 지원하지 못한다. 그러나 어떤 멀티바이트 문자 세트는 이런 지원을 할 수 있도록 FFS 의 독립된 패치를 가지고 있다. 이들 패치는 일시적인 솔루션이거나 해킹된 것이기 때문에 소스 트리에 이들을 추가하지 않았다. 더 많은 정보와 패치 파일에 대해서는 각 언어의 웹 사이트를 참고한다.

FreeBSD MS-DOS 파일시스템은 MS-DOS, 유니코드 문자세트 그리고 선택한 FreeBSD 파일시스템 문자 세트 사이를 변환할 수 있는 기능을 가지고 있다. 자세한 사항은 mount\_msdos(8)을 본다.

## 18.4 I18N 프로그램 컴파일

많은 FreeBSD 포트는 I18N 을 지원하도록 포트 되어있다. 이들 중 몇몇은 포트 이름에 -I18N 이 붙어있다. 이들과 다른 프로그램은 특별한 설정 없이 I18N 을 지원하도록 만들어졌다.

그러나 MySQL 과 같은 어떤 어플리케이션은 특별한 문자 세트로 makefile 을 설정해야 된다. 이것은 보통 Makefile 이나 소스의 **configure** 에 옵션을 설정하면 된다.

## 18.5 특정 언어로 FreeBSD 지역화

### 18.5.1 러시아 어 (KOI8-R 인코딩)

KOI8-R 인코딩에 대한 더 많은 정보는 KOI8-R 레퍼런스를 본다(<http://koi8.pp.ru/>).

#### 18.5.1.1 지역 설정

---

다음 라인을 ~/.login\_conf 파일에 넣는다:

```
me:My Account:W
    :charset=KOI8-R:W
    :lang=ru_RU.KOI8-R:
```

지역설정의 예제는 이번 장의 앞부분을 본다.

### 18.5.1.2 콘솔 설정

- FreeBSD 5.0 이전 버전은 커널 설정파일에 다음 라인을 추가한다:

```
options      SC_MOUSE_CHAR=0x03
```

FreeBSD 4.4 와 이전 버전은 다음 라인을 /etc/rc.conf 에 넣는다:

```
mousechar_start=3
```

- /etc/rc.conf에 다음 설정을 사용한다:

```
keymap="ru.koi8-r"
scrnmap="koi8-r2cp866"
font8x16="cp866b-8x16"
font8x14="cp866-8x14"
font8x8="cp866-8x8"
```

- /etc/ttys의 각 *ttyv\** 엔트리 터미널 타입에 *cons25r*을 사용한다.

콘솔 설정 예제는 이번 장의 앞부분을 본다.



---

### 18.5.1.3 프린터 설정

대부분의 러시아어 프린터는 하드웨어 코드 페이지 CP8666 을 가지고 있기 때문에 KOI8-R 을 CP8666 으로 변환하기 위한 특별한 출력필터가 필요하다. 이런 필터는 기본적으로 /usr/libexec/lpr/ru/koi2alt 에 설치된다. 러시아어 프린터 /etc/printcap 엔트리는 다음과 비슷하다:

```
lp|Russian local line printer:W
    :sh:of=/usr/libexec/lpr/ru/koi2alt:W
    :lp=/dev/lpt0:sd=/var/spool/output/lpd:lf=/var/log/lpd-errs:
```

자세한 설명은 printcap(5)를 본다.

### 18.5.1.4 MS-DOS FS 과 러시아어 파일 이름

다음 예제는 마운트된 MS-DOS 파일시스템에서 러시아어 파일 이름을 지원할 수 있도록 설정한 fstab(5) 엔트리 예제다:

```
/dev/ad0s2    /dos/c  msdos  rw,-Wkoi2dos,-Lru_RU.KOI8-R 0 0
```

-L 옵션으로 사용하는 로컬 이름을 선택하고 -W는 문자 변환 테이블을 설정한다. 변환 테이블이 /usr/libdata/msdosfs 에 있기 때문에 -W 옵션을 사용하려면 MS-DOS 파티션을 마운트하기 전에 /usr 을 마운트해야 된다. 자세한 설명은 mount\_msdos(8)을 본다.

### 18.5.1.5 X11 설정

- ① 앞에서 설명했듯이 X 지역 설정이 아닌 것을 먼저 한다.

**Note:** 러시아어 KOI8-R 지역 설정은 예전 XFree86 릴리즈에서(3.3 보다 낮은) 작동하지 않을 것이다. XFree86 4.X 는 이제 FreeBSD 의 기본 X 윈도우 시스템 버전이기 때문에 여러분이 예전 FreeBSD 버전을 사용하지 않는다면 문제없다.

- 
- ② ru 디렉터리에서 다음 명령을 입력한다:

```
# make install
```

위 포트는 KOI8-R 폰트의 최신 버전을 설치한다. XFree86 3.3 은 KOI8-R 폰트를 몇 개 가지고 있지만 위의 버전이 더 좋다.

/etc/XF86config 파일에서 "Files" 섹션을 체크한다. 다음 라인을 다른 FontPatch 엔트리 이전에 추가해야 된다:

```
FontPath "/usr/X11R6/lib/X11/fonts/cyrillic/misc"
FontPath "/usr/X11R6/lib/X11/fonts/cyrillic/75dpi"
FontPath "/usr/X11R6/lib/X11/fonts/cyrillic/100dpi"
```

고해상도 모드를 사용한다면 75 dpi 와 100 dpi 라인을 교체한다.

- ③ 러시아어 키보드를 활성화하려면 XF86Config 파일의 "keyboard" 섹션에 다음 설정을 추가한다.

XFree86 3.X 에서는 다음 설정을 추가한다:

```
XkbLayout "ru"
XkbOptions "grp:caps_toggle"
```

XFree86 4.X 에서는 다음 설정을 추가한다:

```
Option "XkbLayout" "ru"
Option "XkbOptions" "grp:caps_toggle"
```

또한 이곳에서 *XkbDisable* 를 끈다(주석을 붙인다).

RUS/LAT 는 CapsLock 으로 변경된다. 예전 CapLock 기능은 Shift+CapsLock 로(LAT 모드에서만) 사용할 수 있다.

---

키보드에 "Windows@" 키를 가지고 있다면 RUS 모드에서 알파벳이 아닌 어떤 키는 부 정확하게 매핑되기 때문에 XF86Config 파일에 다음 라인을 추가한다.

XFree86 3.X 은 다음 라인을 추가한다:

```
XkbVariant "winkeys"
```

XFree86 4.X 은 다음 라인을 추가한다:

```
Option "XkbVariant" "winkeys"
```

**Note:** 러시아어 XKB 키보드는 예전 XFree86 버전에서 작동하지 않을 것이다. 더 많은 정보는 위의 내용을 본다. 러시아어 XKB 키보드도 지역적화되지 않은 어플리케이션에서 제대로 작동하지 않을 것이다. 최소한으로 지역화된 어플리케이션은 XtsetLanguageProc(NULL, NULL, NULL)을 호출한다; 예전 프로그램의 기능. X11 어플리케이션 지역화에 대한 더 많은 정보는 X 윈도우용(<http://koi8.pp.ru/xwin.html>) KOI8-R 을 본다.

## 18.5.2 대만어를 위한 전통 중국 지역화

FreeBSD 대만 프로젝트는 <http://freebsd.sinica.edu.tw/~ncvs/zh-l10n-tu/>에 중국 포트를 사용하는 FreeBSD용 I18N/L10N 튜토리얼을 가지고 있다. zh-L10N-tut 편집자는 Clive Lin<Clive@CirX.org>이다. freebsd.sinica.edu.tw에서 다음 컬렉션의 cvsup을 사용할 수 있다:

컬렉션	설명
outta-port tag=.	중국어용 베타버전 포트 컬렉션
zh-L10N-tut tag=.	BIG-5 전통 중국어로 FreeBSD 튜토리얼 지역화
zh-doc tag=.	FreeBSD 문서를 BIG-5 전통 중국어로 번역

Chuan-Hsing Shen<s874070@mail.yzu.edu.tw>는 대만 FreeBSD 의 zh-L10N-tut 을 사용하여 중국어 FreeBSD 컬렉션을(CFC) (<http://cnpa.yzu.edu.tw/~cfc/>) 생성하였다. 이 패키지과 스크립트 파일은 <ftp://ftp.csie.ncu.edu.tw/OS/FreeBSD/taiwan/CFC/>에서 사용할

---

수 있다.

### 18.5.3 독일어 지역화 (모든 ISO 8859-1 언어용)

Slaven Rezic <eserte@cs.tu-berlin.de>는 FreeBSD 머신에서 움 라우트 문자를 사용하는 방법에 대한 튜토리얼을 작성했다. 이 튜토리얼은 독일어로 작성되었고 <http://www.de.FreeBSD.org/de/umlaute/>에서 볼 수 있다.

### 18.5.4 일본어와 한국어 지역화

한국어는 <http://www.kr.FreeBSD.org/>를 그리고 일본어는 <http://www.jp.FreeBSD.org/>를 참고한다.

### 18.5.5 비 영어권 FreeBSD 문서

어떤 FreeBSD 공헌자는 FreeBSD의 일부분을 다른 언어로 번역했다. 이들은 메인 사이트 링크나 `/usr/share/doc`에서 볼 수 있다.

---

## 19 장 서버 업데이트

### 19.1 개요

FreeBSD 는 릴리즈 사이에서 계속해서 개발되고 있다. 최신 기술을 원하는 사람들에게 최신 개발버전으로 시스템을 동기화할 수 있는 여러 가지 쉬운 메커니즘이 있다.

최신 기술은 모든 사람에게 필요한 것이 아니다. 이번 장에서는 개발 중인 시스템을 사용할 것인지 릴리즈된 버전 중 하나를 사용할지 결정하는데 도움을 준다.

이번 장을 읽고 다음과 같은 사항을 알 수 있다:

- 두 개발 분기의 차이점: FreeBSD-STABLE과 FreeBSD-CURRENT.
- CVSsup, CVS 또는 CTM으로 시스템 업데이트하기
- make world로 전체 기본시스템을 다시 빌드해서 설치하는 방법

이번 장을 읽기 전에 다음 사항을 알고 있어야 한다:

- 네트워크를 정확히 설정할 수 있어야 된다(24장).
- 추가적인 소프트웨어를 설치하는 방법을 알아야 된다 (4장)

### 19.2 FreeBSD-CURRENT 대 FreeBSD-STABLE

FreeBSD 에는 두 개의 개발 분기가 있다: FreeBSD-CURRENT 와 FreeBSD-STABLE. 이 섹션은 각각에 대해 간단히 소개하고 시스템을 각각의 트리로 업데이트하는 방법을 설명한다. FreeBSD-CURRENT 를 처음에 설명하고 FreeBSD-STABLE 를 설명한다.

#### 19.2.1 FreeBSD 를 현재 버전으로 유지하기

이 글을 읽게 되면 FreeBSD-CURRENT 가 가장 최근에 개발된 FreeBSD 버전임을 알게 된다. FreeBSD-CURRENT 유저는 가장 고난도의 기술을 가지고 있어야 되고 시스템의

---

어려운 문제를 해결할 수 있어야 한다. 여러분이 새로운 FreeBSD 유저라면 CURRENT 를 설치하기 전에 다시 생각해 본다.

### 19.2.1.1 FreeBSD-CURRENT 는 무엇인가?

FreeBSD-CURRENT 는 가장 최근에 작업한 FreeBSD 소스다. 이것은 진행 중이며 실험적인 설정과 다음 정식 릴리즈에 추가하거나 추가하지 않을 수 있는 과도기적인 메커니즘을 포함하고 있다. 많은 FreeBSD 개발자들이 FreeBSD-CURRENT 소스 코드를 메일 컴파일 하지만 소스가 빌드되지 않을 때도 있다. 이런 문제는 가능한 신속히 해결되지만 FreeBSD-CURRENT 가 엄청난 문제를 일으키거나 간절히 원하던 기능이 소스코드를 받는 순간 문제가 될 수 있다.

### 19.2.1.2 FreeBSD-CURRENT 는 누구에게 필요한가?

FreeBSD-CURRENT 는 3 가지의 주된 관심을 가진 그룹에 유용하다:

- ① 소스 트리의 일부에서 활동하는 FreeBSD 그룹의 멤버와 절대적으로 "current"를 유지해야 되는 사람.
- ② 가능한 FreeBSD-CURRENT를 강건하게 만들기 위해 문제 해결에 시간을 투자해야 되는 FreeBSD 테스터 그룹 멤버. 패치에 특정 설정을 적용하려고 하거나 FreeBSD의 일반적인 방향을 제시하려는 사람.
- ③ 단지 눈으로 확인하거나 참고적인 목적으로 current 소스를 사용하려는 사람(다시 말해 사용하지 않고 읽으려는). 이들은 종종 코드에 대해 충고하거나 직접 참여하기도 한다.

### 19.2.1.3 FreeBSD-CURRENT 에서 안 되는 것은?

- ① 새로운 기능이 있다는 얘기를 듣고 사용해 보려고 처음으로 새로운 릴리즈를 사용한다. 새로운 기능을 위해서 처음으로 사용해 본다는 의미는 새로운 버그를 처음으로 경험하는 것이다.

- 
- ② 버그를 수정하는 빠른 방법. 제공된 버전의 FreeBSD-CURRENT는 버그를 수정하기 위해 새로운 버그를 소개하는 것과 같다.
  - ③ 어쩌든 "공식적으로 지원된다". FreeBSD-CURRENT 그룹의 3가지 규정 중 하나에 맞게 우리는 사람들을 돕기 위해 진심으로 최선을 다하지만 단순히 기술 지원을 위해 시간을 할애하지 않는다. 이 의미는 단순히 하루에 수백 개의 질문에 답변하려고 FreeBSD에서 일하는 것이 아니다. 수많은 질문과 메시지 중 FreeBSD를 개선할 수 있는 방법을 찾는 것이다.

#### 19.2.1.4 FreeBSD-CURRENT 사용하기

- ① freebsd-current와(<http://lists.freebsd.org/mailman/listinfo/freebsd-current>) cvs-all 리스트에(<http://lists.freebsd.org/mailman/listinfo/cvs-all>) 등록한다. 이것은 괜찮은 아이디어가 아니고 기본이다. freebsd-current 리스트에 등록하지 않았다면 사람들이 current 시스템에 대해 논평한 내용을 확인할 수 없어서 다른 사람들이 찾아놓은 해결책을 볼 수 없다. 더 중요한 것은 시스템의 안정성 유지에 중요한 빌드를 실수하게 된다.

cvs-all 리스트에서 적절한 정보에 따라 가능한 효과적으로 변경된 각각의 로그 엔트리를 볼 수 있다.

이들 리스트나 다른 리스트에 가입하려면

<http://lists.FreeBSD.org/mailman/listinfo>에서 가입하고 싶은 리스트를 선택한다. 나머지 과정에 대한 설명도 그곳에서 볼 수 있다.

- ② FreeBSD 미러 사이트에서([http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/handbook/mirrors.html](http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/mirrors.html)) 소스를 받는다. 두 가지 방법 중 한 가지를 사용한다:

a. /usr/share/examples/cvsup에서 찾을 수 있는 standard-supfile 파일로 cvsup 프로그램을 사용한다. 전체 소스를 한번에 받거나 변경된 것만 받을 수 있기 때문에 가장 권장하는 방법이다. 많은 사람들은 cron으로 cvsup을 수행하여 소스를 자동으로 업데이트한다. 위의 샘플 supfile을 수정하여 환경에 맞게 cvsup을 설정한다.

---

b. 인터넷 연결 상태가 좋지 않다면(가격이 비싸거나 메일만 된다면) CTM 기능을 사용한다. CTM 은 옵션이다. 많은 문제가 발생할 수 있고 깨진 파일을 받을 수도 있다. 상당히 오랜 시간 동안 제대로 작동되지 않아서 CTM 을 좀처럼 사용하지 않게 되었다. 9600 bps 모뎀이나 더 빠른 회선을 가지고 있다면 CVSup 을 사용하도록 권장한다.

- ③ 사용해 보려고 소스를 받는다면 특정 부분만 선택하지 않고 FreeBSD-CURRENT 전체를 받는다. 소스의 다양한 부분은 업데이트에 영향을 받기 때문에 일부분만 컴파일하면 문제가 발생할 수 있다.

FreeBSD-CURRENT 를 컴파일하기 전에 /usr/src 에 있는 Makefile 를 주의 깊게 읽어본다. 최소한 업그레이드 과정 중 새로운 커널을 설치하고 전체를(build the world) 빌드해야 된다. FreeBSD-CURRENT 메일링 리스트를 읽고 /usr/src/UPDATING 으로 다음 릴리즈에 포함하려는 다른 부트스트랩 프로시저를 업데이트한다.

## 19.2.2 FreeBSD 를 stable 로 유지하기

### 19.2.2.1 FreeBSD-STABLE 은 무엇인가?

FreeBSD-STABLE 는 공식적인 릴리즈가 만들어지는 우리의 개발 분기다. 아직도 개발되고 있지만 언젠가 FreeBSD-STABLE 소스는 특정 목적에 적합하거나 적합하지 않게 될 것이라는 의미다. 이것은 단지 엔지니어용으로 개발중인 것 중 하나고 최종 사용자를 위한 리소스는 아니다.

### 19.2.2.2 FreeBSD-STABLE 은 누구에게 필요한가?

여러분이 FreeBSD 개발 프로세스를 따르거나 특히 다음 FreeBSD 릴리즈에 기여하기를 원하면 FreeBSD-STABLE 사용에 대해 고려해 본다.

보안 문제도 FreeBSD-STABLE 분기에서 해결되었기 때문에 보안 문제로 FreeBSD-STABLE 을 사용할 필요는 없다. FreeBSD 를 위한 모든 보안 권고가 릴리즈에 영향을[1] 미치는 문제를 어떻게 해결하는지 설명하기 때문에 단지 보안문제로 전제 개발 분기를



---

유지한다면 원하지 않는 여러 가지 변경을 가져오게 된다.

[1]: 위의 설명은 정확하지 않다. FreeBSD의 예전 릴리즈를 영원히 지원하지는 않지만 이들 릴리즈를 여러해 동안 지원해왔다. 예전 FreeBSD 릴리즈의 현재 보안 정책을 완벽히 설명한 <http://www.FreeBSD.org/security/>를 본다.

FreeBSD-STABLE 분기가 항상 컴파일되고 운용되도록 노력했지만 아직은 완벽하게 보장하지 못한다. 게다가 코드가 FreeBSD-STABLE에 포함되기 전에 FreeBSD-CURRENT에서 개발되었지만 FreeBSD-CURRENT보다 FreeBSD-STABLE를 더 많은 사람들이 사용하기 때문에 FreeBSD-CURRENT에서 발생하지 않던 버그가 FreeBSD-STABLE에서 종종 발견될 수밖에 없다.

이런 이유로 무턱대고 FreeBSD-STABLE을 사용하지 않기를 권장하고 여러분의 개발환경에서 철저히 테스트하지 않고 서버를 FreeBSD-STABLE로 업데이트하지 않는다.

STABLE를 사용하지 않는다면 가장 최근의 FreeBSD 릴리즈를 사용하고, 릴리즈에서 릴리즈로 업데이트할 때 바이너리 업데이트 메커니즘을 사용하도록 권장한다.

### 19.2.2.3 FreeBSD-STABLE 사용하기

- ① `freebsd-stable` 리스트에(<http://lists.freebsd.org/mailman/listinfo/freebsd-stable>) 가입한다. FreeBSD-STABLE이나 특별히 주의가 필요한 곳에서 나타날 수 있는 빌드 의존성 관련 정보를 이곳에서 얻을 수 있다. 개발자들도 논쟁의 여지가 있는 변경이나 업데이트를 계획할 때 제안된 변경에 대한 유저들의 응답을 들을 수 있도록 이 메일링 리스트에 공고할 수 있다.

`cvs-all` 리스트에서 적절한 정보에 따라 가능한 효과적으로 변경된 각각의 로그 엔트리를 볼 수 있다.

이들 리스트나 다른 리스트에 가입하려면

<http://lists.FreeBSD.org/mailman/listinfo>에서 가입하고 싶은 리스트를 선택한다.

나머지 과정에 대한 설명도 그곳에서 볼 수 있다.

- ② 새로운 시스템을 설치하고 `stable`로 가능한 유지하려면

---

ftp://snapshots.jp.FreeBSD.org/pub/FreeBSD/snapshots/에서 분기의 가장 최근 스냅샷을 받아서 다른 릴리즈처럼 설치한다. 미러 사이트에서([http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/handbook/mirrors.html](http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/mirrors.html)) 가장 최근의 FreeBSD-STABLE 릴리즈를 설치하고 다음 지시에 따라 가장 최근의 FreeBSD-STABLE 코드로 업데이트해도 된다.

이전 릴리즈의 FreeBSD 를 사용하고 소스를 통해 업그레이드하려면 FreeBSD 미러 사이트를 통해 쉽게 업그레이드할 수 있다. 다음 두 가지 중 한가지 방법을 사용할 수 있다:

a. /usr/share/examples/cvsup 에서 찾을 수 있는 standard-supfile 파일로 cvsup 프로그램을 사용한다. 전체 소스를 한번에 받거나 변경된 것만 받을 수 있기 때문에 가장 권장하는 방법이다. 많은 사람들은 cron 으로 cvsup 을 수행하여 소스를 자동으로 업데이트한다. 위의 샘플 supfile 을 수정하여 환경에 맞도록 cvsup 를 설정한다.

b. CTM 을 사용한다. 인터넷 회선이 빠르지 않고 비싸다면 시도해 볼만한 방법이다.

- ③ 기본적으로 필요할 때 즉시 소스에 접근할 수 있고 통신 대역폭을 고려할 필요가 없다면 cvsup이나 ftp를 사용하고 그렇지 않다면 CTM을 사용한다.
- ④ FreeBSD-STABLE를 컴파일하기 전에 /usr/src에있는 Makefile를 주의 깊게 읽어 본다. 최소한 업그레이드 과정 중에 새로운 커널을 설치하고 세상 만들기를 해야 된다. FreeBSD-STABLE 메일링 리스트를 읽고 /usr/src/UPDATING으로 다음 릴리즈에 포함하려는 다른 부트스트랩 프로시저를 업데이트한다.

## 19.3 소스 동기화

FreeBSD 프로젝트 소스 중 특정 영역이나 관심 있는 모든 영역의 소스를 인터넷으로(또는 email) 업데이트하는 다양한 방법이 있다. 주로 우리가 서비스하는 것은 익명 CVS([http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/handbook/anoncvs.html](http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/anoncvs.html)), CVSup([http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/handbook/cvsup.html](http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/cvsup.html))

---

그리고 CTM([http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/handbook/ctm.html](http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/ctm.html))이다.

**주의:** 소스 트리 중 특정 부분만 업데이트하는 것도 가능하지만 유일하게 지원되는 업데이트 프로시저는 전체 트리를 업데이트하고 유저 기반과(다시 말해 /bin 과 /sbin 에서 운용중인 모든 유저 프로그램) 커널 소스를 다시 컴파일하는 것이다. 소스 트리 중 특정 부분, 커널 또는 유저 기반 프로그램만 업데이트하면 종종 문제가 발생된다. 이들 문제는 컴파일 에러에서부터 커널 패닉이나 데이터 손실 등이 발생한다.

익명 CVS 와 CVSup 는 *pull* 모델의 소스 업데이트를 사용한다. 이러한 경우 CVSup 은 유저가 cvsup 프로그램을 실행하면 cvsupd 서버와 연동하여 파일을 업데이트한다. 업데이트에는 약간의 시간이 소요되고 원할 때만 업데이트해도 된다. 관심 있는 특정 파일이나 디렉터리만 업데이트할 수 있다. 어떤 파일을 가지고 있고 무엇을 업데이트하는가에 따라 업데이트는 서버에 부하를 유발한다. CVSup 은 원격 CVS 저장소에서 변경된 디렉터리를 정확히 받아오도록 단지 CVS 를 확장한 것이기 때문에 익명 CVS 는 CVSup 보다 더 간단하다. CVSup 이 더 효과적이지만 익명 CVS 가 사용하기 쉽다.

반면 CTM 은 마스터에 저장되어 있는 전체 소스와 여러분이 가지고 있는 소스를 비교하지 않는다. 대신 마지막으로 실행된 이후부터 변경된 파일을 확인하는 스크립트가 마스터 CTM 머신에서 하루에 여러 번씩 실행되어 변경된 것을 압축한 후 순서대로 번호를 부여하고 메일로(출력할 수 있는 ASCII 기 때문에) 보내기 위해 암호화한다. 이들 CTM 데이터를 받은 뒤 자동으로 암호를 풀고 변경된 것을 유저 소스에 복사, 적용, 확인하는 ctm\_rmail(1) 유틸리티에게 넘긴다. 이 프로세스는 CVSup 보다 더욱 효과적이고 *pull* 모델이 아닌 *push* 모델이기 때문에 서버 자원을 적게 사용한다.

물론 또 다른 차이가 있다. 부주의하게 데이터를 처리하였다면 CVSup 은 손상이 있는 부분을 감지하여 다시 빌드한다. CTM 은 이런 기능이 없고 소스 트리에서 특정 부분을 지웠다면(백업하지 않고) 단순히 삭제한 비트를 다시 동기화하기 위해 CTM 이나 익명 CVS 로 다시 빌드해야 된다.

## 19.4 make world 사용

로컬 소스 트리를 FreeBSD 의 특정 버전에(FreeBSD-STABLE, FreeBSD-CURRENT 등)

---

맞도록 동기화했다면 시스템을 다시 빌드하기 위해 소스 트리를 사용할 수 있다.

**백업:** make world 하기 전에 시스템을 백업하는 것이 굉장히 중요하다. world 를(다음의 지시를 따른다면) 다시 빌드하는 것은 쉬운 일이지만 실수를 했거나 불가피한 상황이 발생하면 시스템이 부팅하지 못한다.

백업을 하고 부팅 플로피를 만든다. 이 플로피를 사용할 일이 없을 수도 있지만 후회하는 것보다 안전한 것이 낫다.

**적당한 메일링 리스트에 가입한다:** 기본적으로 FreeBSD-STABLE 과 FreeBSD-CURRENT 분기는 개발 중이다. FreeBSD 에 공헌하는 사람들도 종종 실수를 유발하기도 한다.

가끔 이런 실수는 대수롭지 않게 시스템에 진단 메시지를 출력하는 원인이 되기도 한다. 아니면 변경된 것이 큰 문제를 유발하여 시스템이 부팅하지 못하거나 파일시스템을 파괴할 수 있다(최악의 경우).

이런 문제가 발생했다면 문제를 설명하고 어떤 시스템과 관련 있는지 적절한 메일링 리스트에 경고 메시지가 붙는다. 그리고 모든 문제가 해결되었을 때 경고 해제 발표가 난다.

FreeBSD-STABLE 이나 FreeBSD-CURRENT 를 유지하면서 FreeBSD-STABLE 또는 FreeBSD-CURRENT 메일링 리스트를 읽지 않는다면 문제에 부딪치게 된다.

## 19.4.1 시스템을 업데이트하는 일반적인 방법

시스템을 업데이트 하려면 다음 절차를 따른다:

```
# make buildworld
# make buildkernel
# make installkernel
# reboot
```

싱글 유저모드로(예를 들면 로더 프롬프트에서 boot -s 를 사용) 부팅한 후 다음 명령을

---

실행한다:

```
# mergemaster -p
# make installworld
# mergemaster
# reboot
```

**더 자세한 설명:** 위에서 설명한 순서는 시스템 업데이트를 시작할 수 있도록 짧게 요약한 내용이다. 그러나 각 단계를 정확히 이해할 수 있도록 다음 단계를 읽고 특히 사용자 커널 설정을 사용한다면 완벽히 이해해야 된다.

## 19.4.2 /usr/src/UPDATING 을 읽는다

시작하기 전에 /usr/src/UPDATING 을(또는 언제나 소스 코드에서 복사할 수 있는 이와 같은 파일) 읽는다. 이 파일은 발생하게 될 문제나 특정 명령을 실행하는 순서에 대한 중요한 정보를 가지고 있다. UPDATING 이 이 글과 약간 모순된다면 UPDATING 을 따른다.

**중요:** UPDATING 을 읽는다고 이전에 설명한 정확한 메일링 리스트에 가입되는 것이 아니다. 두 가지 필수사항은 배타적이 아닌 상호 보완적이다.

## 19.4.3 /etc/make.conf 체크

/etc/defaults/make.conf 와 /etc/make.conf 를 확인한다. 첫 번째 파일에는 기본 정의 몇 가지가 있다; 대부분은 주석처리 되어 있다. 소스에서 시스템을 다시 빌드할 때 이들 정의를 사용하려면 /etc/make.conf 에 추가한다. /etc/make.conf 에 추가하는 모든 것은 make 를 실행할 때마다 사용되기 때문에 기본적인 내용은 시스템에 설정해두는 것이 좋다.

일반적인 관리자는 /etc/defaults/make.conf 에서 CFLAGS 와 NOPROFILE 라인을 /etc/make.conf 에 복사하여 주석을 푼다.

다른 정의를(COPTFLAGS, NOPORTDOCS 등) 확인해서 필요한 것인지 결정한다.

---

## 19.4.4 /etc 파일 업데이트

/etc 디렉터리는 시스템이 시작될 때 사용되는 스크립트와 시스템 설정 정보의 중요한 부분을 포함한다. 이들 스크립트 중 몇 개는 FreeBSD 버전에 따라 변경된다.

또한 설정파일 중 몇 가지, 특히 /etc/group 과 같은 파일은 시스템이 하루하루 운용되는 동안 사용된다.

"make world"를 설치하는 동안 특정 유저이름과 그룹이 있어야 될 때 문제가 발생되어 왔다. 업데이트할 때 이들 유저나 그룹이 존재하지 않다면 문제가 된다.

최근의 사례는 smmsp 유저가 추가되었을 때다.mtree(8)가 /var/spool/clientmqueue 를 생성하려고 할 때 설치 과정이 실패했었다.

해결방법은 /usr/src/etc/group 를 확인해서 여기에 나열된 그룹과 여러분의 그룹을 비교한다. 여러분의 파일에 없는 그룹이 새로운 파일에 있다면 이것을 복사한다. 또한 /etc/group 에서 /usr/src/etc/group 과 이름은 다르지만 같은 GID 를 가지고 있는 그룹은 이름을 바꾼다.

4.6-RELEASE 부터 world 를 빌드하기 이전 모드에서 -p 옵션을 제공하기 때문에 mergemaster(8)을 실행할 수 있다. 이것은 buildworld 나 installworld 를 성공시키는데 필수적인 파일만 비교한다. 이전 버전의 mergemaster 가 -p 를 지원하지 않는다면 처음 실행할 때 소스 트리에서 새로운 버전을 사용한다:

```
# cd /usr/src/usr.sbin/mergemaster
# ./mergemaster.sh -p
```

**Tip:** 특히 걱정된다면 이름을 변경하거나 삭제하려는 파일이 어느 그룹인지 확인한다:

```
# find / -group GID -print
```

위 명령은 그룹 *GID* 의(그룹 이름이나 숫자로 된 그룹 ID) 모든 파일을 보여준다

## 19.4.5 싱글 유저모드로 들어간다.

---

싱글 유저모드에서 시스템을 컴파일 하려고 할 것이다. 시스템을 다시 설치하려면 모든 표준 시스템 바이너리, 라이브러리, 파일 등을 포함하여 수많은 시스템 파일을 다루기 때문에 싱글 유저모드에서 빠르게 컴파일 할 수 있다. 운용중인 시스템에서(특히 그 시간에 작업하는 유저가 있다면) 이런 컴파일은 문제를 발생할 수 있다.

시스템 컴파일은 멀티 유저모드에서 수행하고 설치하는 싱글 유저모드에서 수행하는 방법이 있다. 이 방법을 원한다면 단순히 빌드가 끝날 때까지 다음 단계를 따른다.  
installkernel 이나 installworld 를 해야 될 때 싱글 유저모드로 변경한다.

슈퍼유저에서 다음 명령을 실행한다:

```
# shutdown now
```

위 명령으로 운영중인 시스템에서 싱글 유저모드로 들어간다.

다른 방법은 시스템을 재 부팅하고 부트 프롬프트에서 `-s` 플래그를 입력해도 된다.  
시스템이 싱글 유저모드로 부팅하면 쉘 프롬프트에서 다음 명령을 수행한다:

```
# fsck -p  
# mount -u /  
# mount -a -t ufs  
# swapon -a
```

위 명령은 파일시스템을 체크해서 읽기 쓰기로 /를 다시 마운트한 후 /etc/fstab 에 있는 다른 UFS 파일시스템을 마운트하고 스왑을 켜다.

**Note:** CMOS 시간이 GMT 가(date(1) 명령의 출력이 정확한 시간이나 위치를 나타내지 않는다면) 아닌 로컬 시간으로 설정되어있다면 다음 명령을 실행해야 될 것이다:

```
# adjkerntz -i
```

이 명령은 로컬 시간을 정확하게 설정한다; 정확하지 않은 시간은 나중에 문제를 유발한다.

---

## 19.4.6 /usr/obj 삭제

빌드된 시스템의 일부는 /usr/obj 디렉터리에(기본적으로) 저장된다. 이 디렉터리는 /usr/src 의 복사본이다.

이 디렉터리를 삭제하여 "make world" 진행 속도를 높일 수 있고 문제의 소지를 없앨 수 있다.

/usr/obj 의 어떤 파일은 먼저 삭제해야 되는 삭제방지 플래그가 설정되어있다(더 많은 정보는 chflags(1)을 본다).

```
# cd /usr/obj
# chflags -R noschg *
# rm -rf *
```

## 19.4.7 소스를 다시 컴파일하기

### 19.4.7.1 출력 저장

make(1)을 실행하면서 생긴 출력을 다른 파일에 저장하는 것도 좋은 생각이다. 문제가 발생하면 에러 메시지를 복사할 수 있다. 어떤 문제인지 진단할 때 직접 도움이 안되더라도 FreeBSD 메일링 리스트에 문제를 게시한다면 도움이 된다.

출력을 저장하는 가장 쉬운 방법은 모든 출력을 저장할 파일 이름을 script(1) 명령에 지정해서 사용한다. world 를 다시 빌드하기 전에 빨리 script(1) 명령을 실행하고 진행이 끝나면 exit 를 입력한다.

```
# script /var/tmp/mw.out
Script started, output file is /var/tmp/mw.out
# make TARGET
... compile, compile, compile ...
# exit
Script done, ...
```



---

이렇게 해서 출력을 /tmp 에 저장하지 않는다. 이 디렉터리는 다음에 재 부팅할 때 정리되기 때문이다. 더 좋은 장소는 /var/tmp(이전의 예제처럼) 또는 root 의 홈 디렉터리에 저장한다.

## 19.4.6.2 기본 시스템 컴파일하기

/usr/src 디렉터리로 이동한다:

```
# cd /usr/src
```

(물론 소스 코드가 다른 곳에 있다면 소스 코드가 있는 디렉터리로 변경한다)

world 를 다시 빌드하려면 make(1) 명령을 사용한다. 이 명령은 FreeBSD 를 어떻게 다시 빌드해야 되는지 빌드되는 순서 등이 설명된 Makefile 의 설정을 읽는다.

입력할 일반적인 명령어 라인 형식은 다음과 같다:

```
# make -x -DVARIABLE target
```

이 예제에서 -x 는 make(1)에 적용하는 옵션이다. 적용할 수 있는 옵션 예제는 make(1) 매뉴얼 페이지를 본다.

-*DVARIABLE*은 Makefile 에 적용하는 변수고 Makefile 의 동작은 이들 변수로 제어된다. 이들은 /etc/make.conf 에 지정하는 변수와 같고 설정하는 다른 방법을 제공한다.

```
# make -DNOPROFILE target
```

위 명령은 빌드하지 말아야 되는 라이브러리 프로필을 지정하는 다른 방법이고 /etc/make.conf 의 다음 라인과 일치한다.

```
NOPROFILE= true # Avoid compiling profiled libraries
```

*target*은 make(1)에게 무엇을 해야 되는지 알려준다. 각 Makefile 에 다른 여러 타겟을 지정하고 어떤 타겟에 무엇을 해야 되는지 지정한다.

---

어떤 타겟은 Makefile 에 나열되어 있지만 실행해야 되는 것은 아니다. 대신 시스템이 세부 단계를 다시 빌드하기 위해 필요한 단계로 빠져 나가도록 빌드 프로세스가 사용한다.

대부분 make(1)에 매개변수를 적용할 필요가 없기 때문에 명령은 다음과 비슷하다:

#### **# make target**

FreeBSD 버전 2.2.5 부터(실제로 FreeBSD-CURRENT 분기에서 최초로 만들어졌고 FreeBSD-STABLE 2.2.2 와 2.2.5 사이에서 적용되었다) world 타겟은 두 개로 나뉘었다: buildworld 와 installworld.

이름이 암시하듯 buildworld 는 /usr/obj 에 새로운 트리를 완벽히 빌드하고 installworld 는 현재 머신에 이 트리를 설치한다.

이것은 2 가지 이유 때문에 매우 유용하다. 첫째 운영중인 시스템의 어떠한 컴포넌트도 영향을 받지 않도록 빌드를 안전하게 한다. 멀티 유저모드로 운영중인 머신에서 신경 쓰지 않고 buildworld 를 안전하게 실행할 수 있다. 그렇지만 아직도 installworld 부분은 싱글 유저모드에서 실행하는 것이 권장된다.

두 번째로 네트워크에 있는 여러 머신을 NFS 마운트로 업그레이드할 수 있다. A, B 와 C 3 대의 머신을 업그레이드한다면 A 에서 make buildworld 와 make installworld 를 실행한다. B 와 C 는 /usr/src 와 /usr/obj 를 A 에서 NFS 로 마운트하고 make installworld 를 실행해서 빌드 결과를 B 와 C 에 설치한다.

world 타겟이 있지만 사용하지 않는다.

다음 명령을 실행한다:

#### **# make buildworld**

그리고 여러 개의 프로세스를 동시에 실행하는 -j 옵션을 make 에 지정할 수 있다. 이것은 여러 개의 CPU 가 있을 때 아주 유용하다. 그러나 대부분의 컴파일과정이 CPU 역할 보다는 IO 절차가 많기 때문에 CPU 가 하나만 있는 머신에서도 유용하다.

전형적으로 CPU 가 하나만 있는 머신에서 다음 명령을 실행한다.

---

# `make -j4 buildworld`

`make(1)`은 한번에 4 개의 프로세스를 실행한다. 메일링 리스트에 있는 경험적인 증거로 위 명령이 보통 최고의 성능을 보장하는 것으로 나타났다.

머신에 여러 개의 CPU 가 있다면 SMP 를 사용하도록 설정된 커널로 6 에서 10 사이의 값을 사용하여 얼마나 속도가 증가하는지 확인한다.

이것은 아직도 실험적이어서 소스 트리가 이 기능으로 종종 깨질 수 있다. 이 매개변수를 사용하여 `world` 를 컴파일 하는데 실패했다면 문제를 메일링 리스트에 올리기 전에 이 매개변수 없이 다시 시도한다.

### 19.4.7.3 빌드 시간

여러 가지 요인이 빌드 시간에 영향을 주지만 어떤 트릭이나 단축키를 사용하지 않고 현재 500MHz 펜티엄 III 와 128MB 의 메모리로 `FreeBSD-STABLE` 트리를 빌드하는데 대략 2 시간이 걸린다. `FreeBSD-CURRENT` 트리는 약간 더 걸린다.

## 19.4.8 새로운 커널 컴파일과 설치

새로운 시스템을 완벽히 적용하려면 커널을 다시 컴파일한다. 이것은 특정 메모리 구조가 변경되기 때문에 특히 필요하고 `ps(1)`와 `top(1)`같은 프로그램은 커널과 소스 코드 버전이 같아질 때까지 작동하지 않는다.

커널을 다시 컴파일하는 가장 간단하고 안전한 방법은 `GENERIC` 기반 커널에서 빌드하고 설치한다. `GENERIC` 은 시스템에 필요한 모든 장치를 가지고 있지 않아서 시스템을 싱글 유저모드로 다시 부팅하여 필요한 모든 것을 추가해야 된다. 새로운 시스템이 정확히 동작하는지 테스트하는 것도 좋은 생각이다. `GENERIC` 으로 부팅해서 시스템이 동작하는지 확인하고 일반 커널 설정파일에 새로운 커널을 빌드할 수 있다.

현대 버전의 `FreeBSD` 에서 새로운 커널을 빌드하기 전에 `world` 를 빌드해야 된다.

**Note:** 미리 설정된 파일로 사용자 커널을 빌드하려면 다음과 같이 `KERNCONF=MYKERNEL` 를 사용한다:

---

```
# cd /usr/src
# make buildkernel KERNCONF=MYKERNEL
# make installkernel KERNCONF=MYKERNEL
```

FreeBSD 4.2 또는 예전 버전에서는 *KERNCONF=* 를 *KERNEL=*로 변경해야 된다.  
4.2-STABLE 은 2001 년 2 월 이전에 패치 되어서 *KERNCONF=*를 인식하지 못한다.

**kern.securelevel** 을 1 이상으로 올렸고 커널 바이너리에 **noshchg** 이나 비슷한 플래그를 설정하였다면 싱글 유저모드에서 **installkernel** 을 사용해야 된다. 플래그를 설정하지 않았다면 이들 명령을 멀티 유저모드에서 문제 없이 수행할 수 있다. **kern.securelevel** 에 대한 자세한 사항은 **init(8)**를 그리고 다양한 파일 플래그는 **chflags(1)**를 확인한다.

FreeBSD 4.0 이전 버전으로 업그레이드 한다면 예전 커널 빌드 프로시저를 사용한다.  
그러나 아래와 같은 명령어 라인을 사용하는 새로운 버전의 **config(8)**를 사용하도록 권장한다.

```
# /usr/obj/usr/src/usr.sbin/config/config KERNELNAME
```

## 19.4.8 싱글 유저모드로 재 부팅

새로운 커널이 동작하는지 테스트하기 위해 싱글 유저모드로 재 부팅한다. 싱글 유저모드로 부팅하는 방법은 19.4.5 의 지시를 따른다.

## 19.4.9 새로운 시스템 바이너리 설치

**make buildworld** 를 실행하여 최신 버전의 FreeBSD 로 빌드하였다면 새로운 시스템 바이너리 설치에 **installworld** 를 사용한다.

다음 명령을 실행한다:

```
# cd /usr/src
# make installworld
```

---

**Note:** make buildworld 명령어 라인에 옵션을 지정했다면 make installworld 명령어 라인에도 같은 옵션을 지정해야 된다. 다른 옵션을 지정할 필요는 없다; 예를 들어 -j 는 installworld 에 절대 사용하지 않는다.

예를 들어 다음 명령을 사용했다면:

```
# make -DNOPROFILE buildworld
```

설치할 때도 같은 옵션을 사용한다:

```
# make -DNOPROFILE installworld
```

그렇지 않으면 make buildworld 하는 동안 빌드하지 않은 라이브러리를 설치하려고 한다.

## 19.4.10 make world 로 업데이트되지 않은 파일 업데이트

world 를 다시 만드는 것은 새로 생성되거나 변경된 설정파일과 특정 디렉토리를(특히 /etc, /var 과 /usr) 업데이트하지 않는다.

수동으로 업데이트하는 것도 가능하지만 이들 파일을 업데이트하는 가장 쉬운 방법은 mergemaster(8)을 사용한다. 어떤 방법을 선택하던지 어떤 문제가 발생할지 모르기 때문에 /etc 를 백업한다.

### 19.4.11.1 mergemaster

mergemaster(8) 유틸리티는 /etc 디렉토리에 있는 여러분의 설정파일과 소스 트리에 있는 /usr/src/etc 에서 설정파일의 차이를 확인하는데 도움을 주는 본 셸 스크립트이다. 이것은 이런 위치에 있는 시스템 설정파일을 소스 트리로 업데이트하는데 권장하는 솔루션이다.

mergemaster 는 3.3-릴리즈와 3.4-릴리즈 사이에 FreeBSD 기본 시스템에 통합되었다. 이 의미는 3.3 이후의 모든 -STABLE 과 -CURRENT 시스템에 이 유틸리티가 있다는 것이다.

시작하려면 프롬프트에서 mergemaster 를 입력하고 시작되는 것을 확인한다.

mergemaster 은 / 아래에 일시적인 root 환경을 빌드하고 다양한 시스템 설정파일을 이곳에 놓는다. 이들 파일은 현재 시스템에 설치되어있는 것과 비교된다. 여기서 추가되거나 수정된 라인은 + 표시로 -는 완전히 삭제됐거나 새로운 라인으로 대체되어 차이가 있는 파일을 diff(1) 포맷으로 보여준다. diff(1) 구문과 파일의 차이점을 보여주는 것에 대한 자세한 정보는 매뉴얼 페이지를 참고한다.

mergemaster(8)은 변경된 각 파일을 보여주고 여기서 새로운 파일 삭제(임시 파일이라고 한), 수정하지 않은 상태에서 임시 파일 설치, 현재 설치된 파일과 임시 파일 합병 또는 diff(1) 결과를 다시 볼 수 있는 옵션을 보여준다.

### /etc/hosts 파일 비교화면

```

Use 'd' to delete the temporary ./etc/hosts
Use 'i' to install the temporary ./etc/hosts
Use 'm' to merge the temporary and installed versions
Use 'v' to view the diff results again

Default is to leave the temporary file to deal with by hand
How should I deal with this? [Leave it for later] m
:::1                localhost.example.com localhost | #FreeBSD: src/etc/hosts.v 1.11.2.4 2003/02/06 20:36:58 dbak
127.0.0.1           localhost.example.com localhost | #
10.1.100.244        test.example.com test | # Host Database
10.1.100.244        test.example.com. | #
| # This file should contain the addresses and aliases for loca
> # share this file. Replace "my.domain" below with the domain
> # machine.
> #
> # In the presence of the domain name service or NIS, this fil
> # not be consulted at all; see /etc/host.conf for the resolut
> #
> #
> :::1                localhost localhost.my.domain
> 127.0.0.1           localhost localhost.my.domain

```

임시 파일 삭제를 선택하면 mergemaster(8)에게 변경되지 않은 현재 파일을 유지하고 새로운 버전을 삭제하도록 한다. 이 선택은 현재 파일을 변경하지 말아야 될 이유가 없다면 권장하지 않는다. ?를 입력하여 mergemaster(8) 프롬프트에서 언제라도 도움말을 볼 수 있다. 유저가 파일 건너뛰기를 선택했다면 모든 파일이 처리된 후 다시 나타난다.

### 도움말 화면

```

%?
l:      use the left version
r:      use the right version
e l:    edit then use the left version
e r:    edit then use the right version
e b:    edit then use the left and right versions concatenated
e:      edit a new version
s:      silently include common lines
v:      verbosely include common lines
q:      quit

```

수정하지 않은 임시 파일을 설치하도록 선택하면 현재 파일을 새로운 것으로 바꾼다. 파일을 수정하지 않았다면 이렇게 하여 새로운 파일로 대체한다.

---

파일 합병을 선택하면 텍스트 에디터에 양쪽 파일의 내용이 나타난다. 스크린 양쪽의 파일을 보면서 합병할 수 있고 양쪽에서 일부분을 선택하여 새로운 파일을 생성할 수 있다. 파일을 양쪽에서 비교할 때 **l** 키는 왼쪽 내용을 **r** 키는 오른쪽 내용을 선택한다. 마지막으로 출력되는 내용은 양쪽 파일을 조합한 것으로 설치될 파일이다. 이 선택은 사용자가 설정을 변경한 파일에 일반적으로 사용된다.

diff(1) 결과를 다시 보겠다고 선택하면 mergemaster(8)은 선택할 옵션을 보여 줄 때처럼 파일의 차이를 보여준다.

mergemaster(8)은 시스템 파일의 변경을 끝낸 후 다른 옵션을 보여준다. mergemaster(8)은 패스워드 파일을 다시 빌드할 것인지 그리고 FreeBSD 5.0 이전 버전을 운용한다면 MAKEDEV(8)을 실행할 것인지 문의하고 임시 파일을 삭제하는 옵션을 보여주고 끝난다.

#### mergemaster 끝내기

```
*** Comparison complete
*** Files that remain for you to merge by hand:
/var/tmp/temproot/etc/hosts
/var/tmp/temproot/etc/motd

Do you wish to delete what is left of /var/tmp/temproot? [no] █
```

### 19.4.11.2 수동 업데이트

수동으로 업데이트하는 것은 /usr/src/etc 파일을 /etc 로 복사하면 되는 것이 아니다. 이들 파일 중 몇 개는 "설치"를 해야 된다. 왜냐하면 /usr/src/etc 디렉터리는 /etc 디렉터리의 복사본이 아니기 때문이다. 게다가 /etc 에만 있고 /usr/src/etc 에 없는 파일이 있다.

mergemaster(8)을 사용한다면 다음 섹션까지 건너뛸 수 있다.

업데이트를 수동으로 하는 가장 간단한 방법은 파일을 새로운 디렉터리에 설치하고 차이점을 찾는 것이다.

**/etc 디렉터리 백업:** 이론상 이 디렉터리의 어떤 것도 자동으로 건드리지 않게 되어 있지만 확실하게 하는 것이 좋다. 그래서 /etc 디렉터리를 다음 명령으로 안전한 곳으로 복사한다:

---

```
# cp -Rp /etc /etc.old
```

-R 은 서브 디렉터리까지 복사 -p 은 파일 변경시간과 소유자의 정보를 유지한다.

새로운 /etc 와 다른 파일을 설치할 임시 디렉터리가 필요하다. /var/tmp/root 가 적당하고 이 디렉터리에 수많은 서브 디렉터리가 존재하게 된다.

```
# mkdir /var/tmp/root
# cd /usr/src/etc
# make DESTDIR=/var/tmp/root distrib-dirs distribution
```

위 명령은 필요한 디렉터리 구조를 빌드하고 파일을 설치한다. /var/tmp/root 아래에 생성된 수많은 서브 디렉터리 중 비어있는 디렉터리는 삭제해야 된다. 삭제하는 가장 쉬운 방법은 다음 명령을 사용한다:

```
# cd /var/tmp/root
# find -d . -type d | xargs rmdir 2>/dev/null
```

이 명령은 비어있는 모든 디렉터리를 삭제한다(이 디렉터리들이 비어있지 않다는 경고를 막기 위해 표준 에러는 /dev/null 로 리다이렉트 한다)

/var/tmp/root 는 이제 / 아래의 적당한 위치에 놓을 모든 파일을 포함한다. 이제 이들 파일에서 이전 파일과의 차이점을 확인한다.

/var/tmp/root 에 설치될 어떤 파일은 "."으로 시작되는 파일을 가지고 있다. 이 글을 작성할 때 다른 파일이 있을 수도 있지만 이와 같은 파일은 /var/tmp/root 와 /var/tmp/root/root 에 있는 쉘 시작 파일뿐 이다. ls -a 로 이들을 확인할 수 있다.

가장 쉬운 방법은 diff(1)로 두 파일을 비교하는 것이다:

```
# diff /etc/shells /var/tmp/root/etc/shells
```

이 명령은 /etc/shells 파일과 새로운 /var/tmp/root/etc/shells 파일을 비교해서 차이점을 보여준다. 변경한 것을 합병하거나 새로운 파일로 덮어쓰기를 결정하는데 이 방법을 사용한다.



---

새로운 root 디렉터리(/var/tmp/root) 이름에 변경 시간을 포함해서 양쪽 버전의 차이점을 쉽게 비교할 수 있다: 주기적으로 world 를 다시 빌드한다는 의미는 /etc 를 주기적으로 업데이트하는 것이기 때문에 귀찮은 일이 될 수 있다.

/etc 에 합병하여 마지막으로 변경된 파일 세트의 복사본으로 이 절차의 속도를 높일 수 있다. 다음 프로시저에서 하나의 아이디어를 제공한다.

- ① 일반적으로 world를 만든다. /etc와 다른 디렉터리를 업데이트할 때 현재 날짜가 반영된 이름으로 타겟 디렉터리를 생성한다. 이 방법을 사용한다면 1998년 2월 14일은 다음과 같다:

```
# mkdir /var/tmp/root-19980214
# cd /usr/src/etc
# make DESTDIR=/var/tmp/root-19980214 W
distrib-dirs distribution
```

- ② 위에서 설명한 것처럼 이 디렉터리에서 변경한 것을 합병한다. 그리고 /var/tmp/root-19980214 디렉터리는 삭제하지 않는다.
- ③ 마지막 버전의 소스를 다운로드 해서 다시 빌드했을 때 위의 단계 1을 따른다. 따라서 /var/tmp/root-19980221이라고 부르게 될 새로운 디렉터리를 생성한다(업데이트 간격이 일주일 이라면).
- ④ 두 디렉터리 사이의 차이를 확인하기 위해 새롭게 추가된 주 사이에 변경된 것은 diff(1)를 사용하여 볼 수 있다.

```
# cd /var/tmp
# diff -r root-19980214 root-19980221
```

일반적으로 이렇게 나온 결과는 /var/tmp/root-19980221/etc 와 /etc 의 다른 세트보다 더욱 작다. 세트의 차이점이 적기 때문에 변경된 것을 여러분의 /etc 디렉터리로 합병하는 것은 더 쉽다.

- ⑤ 이제 예전 두 개의 /var/tmp/root-\* 디렉터리를 삭제할 수 있다:
- ```
# rm -rf /var/tmp/root-19980214
```

- 
- ⑥ 변경된 것을 /etc에 합병할 필요가 있을 때마다 이 절차를 반복한다.

디렉터리 이름을 자동으로 생성하기 위해 date(1)를 사용할 수 있다:

```
# mkdir /var/tmp/root-`date "+%Y%m%d"`
```

## 19.4.12 /dev 업데이트

**Note:** FreeBSD 5.0 이나 이후 버전을 운용 중 이라면 이 섹션을 건너뛰어도 된다. 이들 버전은 장치 노드를 할당하기 위해 devfs(5)를 사용한다.

대부분 mergemaster(8) 틀은 장치 노드 업데이트가 필요할 때를 감지하여 자동으로 수행한다. 여기서는 장치 노드를 수동으로 어떻게 업데이트하는지 알려준다.

안전하게 수행하도록 여러 단계가 있다.

- ① /var/tmp/root/dev/MAKEDEV를 /dev에 복사한다:

```
# cp /var/tmp/root/dev/MAKEDEV /dev
```

/etc 를 업데이트하기 위해 mergemaster(8)을 사용한다면 MAKEDEV 스크립트가 필요할 경우 직접 체크해서 문제없이 복사하여 업데이트한다.

- ② 이제 현재 /dev 스냅샷을 가져온다. 이 스냅샷은 퍼미션, 소유권, 각 파일 이름의 major와 minor 번호를 포함해야 되지만 변경 시간은 포함하지 않는다. 스냅샷을 가져오는 가장 쉬운 방법은 awk(1)를 사용하여 몇몇 정보를 잘라낸다:

```
# cd /dev
# ls -l | awk '{print $1, $2, $3, $4, $5, $6, $NF}' > /var/tmp/dev.out
```

- ③ 모든 장치 노드를 다시 생성한다:

```
# sh MAKEDEV all
```

- ④ 다른 디렉터리의 스냅샷을 작성한다. 여기서는 /var/tmp/dev2.out이다. 이제 잘못

---

생성된 장치 노드가 있는지 두 개의 파일을 살펴본다. 필요는 없겠지만 후회하는 것보다 낫다.

```
# diff /var/tmp/dev.out /var/tmp/dev2.out
```

디스크 슬라이스가 일치하지 않기 때문에 다음 명령으로 슬라이스 전체를 다시 생성한다.

```
# sh MAKEDEV sd0s1
```

여러분의 상황에 따라 약간 다를 것이다.

## 19.4.13 /stand 업데이트

**Note:** 이 단계는 마무리에 포함된다. FreeBSD 5.2 또는 이후 버전을 사용한다면 /rescue 디렉터리는 make installworld 를 수행하는 동안 정적으로 컴파일 된 바이너리로 자동으로 업데이트되기 때문에 /stand 를 업데이트 할 필요는 없다. 안전하게 생략할 수 있다.

완벽하게 끝내기 위해 /stand 파일까지 업데이트하기를 원할 것이다. 이들 파일은 /stand/sysinstall 바이너리에 하드 링크되어 있다. 이 바이너리는 정적으로 링크되어 있기 때문에 다른 파일시스템이(특히 /usr) 마운트되지 않은 상태에서 동작할 수 있다.

```
# cd /usr/src/release/sysinstall
```

```
# make all install
```

## 19.4.14 재 부팅

이제 모든 것이 끝났다. 모든 것이 정확한 위치에 있다면 시스템을 재 부팅한다. 그냥 shutdown (8)을 사용한다:

```
# shutdown -r now
```

---

## 19.4.15 끝내기

이제 FreeBSD 시스템을 성공적으로 업데이트하였다. 축하한다!

약간의 문제가 발생했다면 시스템의 특정 부위를 다시 빌드하는 것은 쉽다. 예를 들어 업그레이드나 /etc 를 합병할 때 /etc/magic 을 우연히 삭제하였다면 file(1) 명령은 동작하지 않는다. 이 경우 다음과 같이 수정할 수 있다:

```
# cd /usr/src/usr.bin/file
# make all install
```

## 19.5.16 자주 있는 질문

### 19.4.16.1 변경할 때 마다 world 를 다시 만들어야 하는가?

이것은 변경한 것에 따라 다르기 때문에 쉽지 않은 질문이다. 예를 들어 CVSup 를 실행해서 다음과 같은 파일이 업데이트 되었다면 전체 world 를 다시 빌드할 필요가 없을 것이다:

```
src/games/cribbage/instr.c
src/games/sail/pl_main.c
src/release/sysinstall/config.c
src/release/sysinstall/media.c
src/share/mk/bsd.port.mk
```

적당한 서브디렉터리로 가서 make all install 을 수행하면 된다. 그러나 예를 들어 src/lib/libc/stdlib 와 같이 변경된 것이 중요하다면 world 를 다시 만들거나 정적으로 링크된 부분을 다시 만들어야 한다(정적 링크도 추가했다면).

그리고 이것은 얼마나 자주 업그레이드 할 것인지 그리고 FreeBSD-STABLE 또는 FreeBSD-CURRENT 를 사용하는지에 따라 다르다.

---

## 19.4.16.2 수많은 11 에러신호와 함께(또는 다른 에러 신호)

### 컴파일에 실패했다. 무슨 일인가?

이것은 보통 하드웨어 문제를 보여준다. world 를 만드는 것은 하드웨어에 효과적으로 스트레스를 테스트하는 방법이고 종종 메모리 문제가 나타난다. 보통 컴파일러가 이상한 신호를 보내면서 이유 없이 죽기 때문에 하드웨어 상태를 알 수 있다.

make 를 다시 시작했고 프로세스가 다른 부분에서 죽는다면 하드웨어 문제가 확실하다.

이 보기에서 머신이 어디서 문제가 발생하는지 확인하기 위해 컴포넌트에서 스왑을 켜는 것을 제외하고 여러분이 해볼 수 있는 것은 매우 적다.

## 19.4.16.3 world 만들기를 끝내고 /usr/obj 를 삭제할 수 있는가?

간단히 대답하면 삭제할 수 있다.

/usr/obj 는 컴파일 하는 동안 생성된 모든 오브젝트 파일을 가지고 있다. 보통 "make world"의 첫 번째 단계는 이 디렉토리를 삭제하고 다시 시작하는 것이다. 만들기를 끝낸 후 /usr/obj 를 가지고 있다면 수많은 디스크 공간을 낭비하는 것이다(현재 대략 340MB 정도).

그러나 여러분이 기억하고 있다면 "make world"에서 이 단계를 건너 뛰어도 된다.

/usr/obj 로 대부분의 소스를 다시 컴파일 할 필요가 없기 때문에 세부적인 빌드를 더욱 빠르게 할 수 있다. /usr/obj 를 삭제해야 되는 이유는 이상한 의존성 문제가 발생하여 빌드가 실패한다. 이 문제는 FreeBSD 메일링 리스트에 종종 보고되고 있다.

## 19.4.16.4 다시 시작하기 위해 빌드를 멈출 수 있는가?

문제를 발견하기 전에 얼마나 진행 되었는가에 따라 다르다.

일반적으로(이것은 꼭 지켜야 되는 룰은 아니다) "make world" 은 기본적인 툴과(gcc(1)와 make(1) 같은) 시스템 라이브러리의 새로운 복사본을 빌드한다. 따라서 이들 툴과 라이브러리가 설치된다. 새로운 툴과 라이브러리는 다시 빌드하는데 사용되고 다시 설치된다. 전체 시스템은(ls(1) 또는 grep(1)과 같은 일반적인 유저 프로그램을 제외하고)

---

새로운 시스템 파일로 다시 빌드된다.

마지막 단계에 있다면(저장되고 있는 출력을 살펴보아서 알 수 있다) 다음과 같이 할 수 있다(상당히 안정적으로):

```
... fix the problem ...  
# cd /usr/src  
# make -DNOCLEAN all
```

이 명령이 "make world"를 이전으로 되돌려 놓지는 않는다.

"make world"출력에서 다음 메시지를 보았다면:

```
-----  
Building everything..  
-----
```

이것은 상당히 안전한 상태다.

이 메시지를 보지 못했거나 확실하지 않다면 후회하는 것보다 안전한 것이 낫기 때문에 빌드를 다시 시작한다.

#### 19.4.16.5 world 를 만들 때 어떻게 속도를 높일 수 있는가?

- 싱글 유저모드로 실행한다.
- /usr/src와 /usr/obj 디렉터리를 다른 디스크의 파일시스템에 둔다. 가능하다면 이들 디스크도 다른 디스크 컨트롤러에 둔다.
- 더 좋은 방법은 이들 파일 시스템을 ccd(4) 장치(디스크 드라이버를 연결시키는) 사용하여 여러 디스크에 둔다.
- 프로필을 끈다(/etc/make.conf에서 ``NOPROFILE=true``). 보통 이렇게까지 할 필요는 없다.

- 
- 또한 /etc/make.conf에 `-O -pipe`처럼 CFLAGS를 설정한다. `-O2`로 튜닝하면 더 느리고 `-O`에서 `-O2` 사이로 다르게 튜닝하는 것도 마찬가지다. `-pipe`는 컴파일러에게 임시 파일보다 디스크 사용을 자제하는(메모리 사용이 많아지는) 파이프를 사용하여 통신하도록 한다.
  - `-jn` 옵션을 `make(1)`에 적용하여 여러 프로세스를 병렬로 수행한다. 이 옵션은 싱글이나 멀티 프로세서 머신이나 상관없이 일반적으로 도움이 된다.
  - /usr/src를 가지고 있는 파일시스템을 `noatime` 옵션으로 마운트(또는 다시 마운트)할 수 있다. 이 방법은 파일시스템을 사용한 시간을 저장하지 않는다. 어쨌든 이 정보가 필요 없을 것이다.

```
# mount -u -o noatime /usr/src
```

**주의:** 이 예제는 /usr/src가 자체 파일시스템에 있다고 가정한다. 그렇지 않다면(예를 들어 /usr의 일부이면) /usr/src가 아닌 다른 파일시스템 마운트 포인트가 필요하다.

- /usr/obj를 가지고 있는 파일시스템은 `async` 옵션으로 마운트(또는 다시 마운트)할 수 있다. 이 방법은 비 동기적으로 디스크 쓰기를 수행한다. 다시 말해서 쓰기는 즉시 이루어지지만 데이터는 몇 초 후에 디스크에 저장된다. 클러스터로 쓰기를 할 수 있어서 극적인 성능 향상을 가져온다.

**주의:** 이 옵션을 사용하면 파일시스템이 깨지기 쉽다. 이 옵션을 사용하는 동안 전원 공급이 끊어지면 머신이 재 시작했을 때 파일시스템은 복구할 수 없는 상태가 될 수 있다.

이 파일시스템에 /usr/obj만 있다면 문제가 되지 않는다. 그렇지 않고 이 파일 시스템에 유용한 데이터가 있다면 이 옵션을 사용하기 전에 백업한다.

```
# mount -u -o async /usr/obj
```

그리고 위에서처럼 /usr/obj가 자체적인 파일시스템에 없다면 적당한 마운트 포인트로 옮긴다.

---

## 19.4.16.6 문제가 생긴다면 어떻게 해야 되는가?

여러분의 환경이 이전 빌드와 많이 다르지 않다면 단순히 다음 명령들을 실행한다.

```
# chflags -R noschg /usr/obj/usr
# rm -rf /usr/obj/usr
# cd /usr/src
# make cleandir
# make cleandir
```

위에서 make cleandir 은 실제로 두 번 실행한다.

그리고 전체 과정을 다시 시작하고 make buildworld 를 시작한다.

그래도 문제가 있다면 에러와 uname -a 의 출력을 FreeBSD 일반 질문 메일링 리스트로 보낸다. 관련된 단계의 다른 사람의 답을 확인한다.

## 19.5 여러 대의 머신 업데이트하기

같은 소스 트리로 만들고 싶은 여러 대의 머신이 있다면 모든 머신에 소스를 다운로드해서 모든 것을 다시 빌드하는 것은 자원 낭비다: 디스크 공간, 네트워크 대역폭과 CPU. 해결책은 대부분의 작업을 한대의 머신에서 수행하고 나머지 머신은 NFS 를 통해서 작업한다. 이 섹션은 이 방법에 대해 설명한다.

### 19.5.1 준비과정

첫째 우리가 *빌드 세트*라고 하는 같은 바이너리 세트를 운용하려는 머신들을 확인한다. 각 머신이 사용자 커널을 가지고 있지만 같은 유저기반 바이너리를 운용할 것이다. 이들 머신 중에서 world 와 커널을 빌드하려는 빌드 머신을 선택한다. make world 를 수행할 충분한 CPU 를 가지고 있는 빠른 머신이어야 된다. 실제 서버에 적용하기 전에 소프트웨어 업데이트를 테스트 할 머신도 선택한다. 이 머신은 일정시간 동안 다운시켜도 되는 머신이어야 된다.



---

이 빌드 세트에서 모든 머신은 똑 같은 머신의 `/usr/obj` 와 `/usr/src` 를 같은 위치에 마운트해야 된다. 여러 개의 빌드 세트를 가지고 있다면 `/usr/src` 는 한대의 빌드 머신에 있어야 되고 나머지는 이곳에 NFS 로 마운트해야 된다.

마지막으로 모든 빌드 세트 머신의 `/etc/make.conf` 는 빌드 머신과 일치해야 된다. 이 의미는 빌드 세트 머신에 설치할 모든 기본 시스템을 빌드 머신에서 빌드해야 된다. 또한 각 빌드 머신은 `/etc/make.conf` 에 `KERNCONF` 로 커널 이름 세트를 가지고 있어야 되며, 빌드 머신은 자신의 커널을 처음에 그리고 모든 커널 이름을 차례로 `KERNCONF` 에 나열해야 된다. 빌드 머신이 다른 머신의 커널을 빌드하려면 `/usr/src/sys/arch/conf` 에 각 머신의 커널 설정파일을 가지고 있어야 한다.

## 19.5.2 기본 시스템

이제 빌드 할 모든 준비가 끝났다. 19.4.7.2 장에서 설명한대로 빌드 머신에서 커널과 `world` 를 빌드하지만 아무것도 설치하지 않는다. 빌드가 끝난 후 테스트 머신에 방금 빌드한 커널을 설치한다. 이 머신이 `/usr/src` 와 `/usr/obj` 를 NFS 로 마운트 했다면 재부팅하여 싱글 유저모드로 들어갔을 때 네트워크를 활성화하고 이들을 마운트한다. 이 들을 마운트하는 가장 쉬운 방법은 멀티 유저모드로 부팅하고 싱글 유저모드로 가기 위해 `shutdown now` 를 실행한다. 그곳에서 새로운 커널과 `world` 를 설치하고 일반적으로 `mergemaster` 를 실행할 수 있다. 끝나고 이 머신을 멀티 유저모드로 되돌리기 위해 재부팅한다.

테스트 머신에서 모든 것이 정상이라면 같은 절차로 빌드 세트의 각 머신에 새로운 소프트웨어를 설치한다.

## 19.5.3 포트

포트 트리도 같은 아이디어를 사용할 수 있다. 첫 번째로 중요한 단계는 `/usr/ports` 를 동일한 머신에서 빌드 세트의 모든 머신에 마운트한다. `/etc/make.conf` 를 적절히 설정하여 `distfiles` 를 공유한다. NFS 마운트로 매핑된 `root` 유저가 쓰기를 할 수 있도록 일반적인 공유 디렉터리에 `DISTDIR` 를 설정한다. 각 머신은 로컬 빌드 디렉터리에 `WRKDIRPREFIX` 를 설정한다. 마지막으로 패키지를 빌드하고 배포한다면 `DISTDIR` 처럼 디렉터리에 `PACKAGES` 를 설정한다.

---

## IV. 네트워크 통신

FreeBSD 는 고성능 네트워크 서버로 아주 광범위하게 사용되고 있는 운영체제 중 하나다. 이번 장에서는 다음과 같은 사항에 대해 다룬다:

- 시리얼 통신
- PPP와 PPPoE
- 전자 메일
- 네트워크 서버 운용
- 다른 고급 네트워킹

이곳은 여러분에게 필요한 정보가 있을 때마다 읽을 수 있도록 디자인되어 있다. 특별한 순서로 읽거나 네트워크 환경에서 FreeBSD 를 사용하기 전에 이 모든 장들을 읽을 필요는 없다.

## 20 장 시리얼 통신

### 20.1 개요

유닉스는 항상 시리얼 통신을 지원해 왔다. 최초의 유닉스 머신은 실제로 사용자 입력과 출력에 시리얼 라인을 사용했다. 초당 평균 10 개의 문자를 시리얼 프린터와 키보드로 전송하던 때부터 일반 터미널은 많은 것이 변하였다. 이번 장에서는 FreeBSD 에서 사용하는 시리얼 통신의 몇 가지 방법에 대해 다룬다.

이번 장을 읽고 다음과 같은 사항을 알 수 있다:

- 터미널로 FreeBSD 시스템에 어떻게 연결하는가
- 원격 호스트에 다이얼-아웃하기 위해 모뎀은 어떻게 사용하는가
- 원격의 유저가 모뎀을 사용하여 시스템에 어떻게 로그인하는가
- 시리얼 콘솔로 시스템을 어떻게 부팅하는가

이번 장을 읽기 전에 다음과 같은 사항을 알고 있어야 된다:

- 
- 새로운 커널 설정과 설치(9장).
  - 유닉스 퍼미션과 프로세스의 이해(3장).
  - FreeBSD에서 사용할 시리얼 하드웨어용(모뎀 또는 멀티 포트 카드) 기술문서가 필요하다.

## 20.2 소개

### 20.2.1 전문 용어

#### bps

초당 비트율 - 데이터의 전송 속도를 나타내는 단위.

#### DTE

데이터 터미널 장비 -- 예를 들어 컴퓨터.

#### DCE

데이터 통신 장비 -- 모뎀.

#### RS-232

하드웨어간 시리얼 통신 EIA 표준규격.

데이터 통신율에 대해 설명할 때 이번 장에서는 "보드(baud)"라는 용어를 사용하지 않는다. 보드는 일정 시간 동안의 전자적인 상태 변화를 나타내는 숫자이기 때문에 "bps(초당 비트율)"가 사용하기에 정확하다.

### 20.2.2 케이블과 포트

FreeBSD 시스템에 모뎀이나 터미널로 연결하려면 컴퓨터에 시리얼 포트가 있어야 되고 시리얼 장치와 연결하기 위한 적당한 케이블이 필요하다. 하드웨어와 필요한 케이블에 대해 잘 알고 있다면 이번 섹션을 지나쳐도 된다.

---

## 20.2.2.1 케이블

여러 종류의 시리얼 케이블이 있다. 목적에 맞는 아주 일반적인 두 종류는 널 모뎀 케이블과 표준("직렬") RS-232 케이블이다. 하드웨어 문서에 필요한 케이블의 종류에 대한 설명이 있을 것이다.

### [케이블 종류와 특성]

#### 1. 널 모뎀 케이블

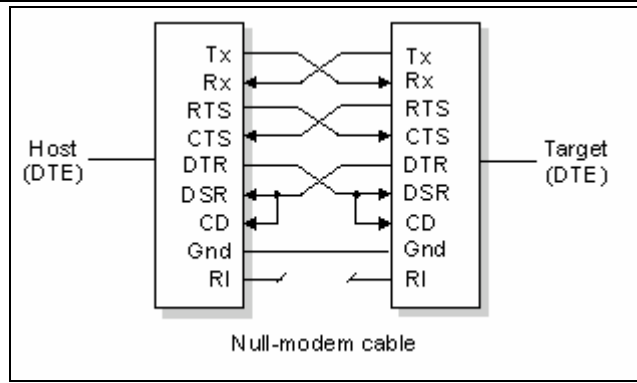
널 모뎀 케이블은 "신호용 접지(Signal Ground)"와 같은 어떤 신호는 직접 흐르게 하지만 다른 신호로 바꾸기도 한다. 예를 들어 한쪽 끝 단의 핀에서 "보낸 데이터"는 다른 끝 단의 "받는 데이터"가 된다

케이블을 직접 만든다면 터미널에 사용하기 위해 널 모뎀 케이블을 조립할 수 있다. 이 표는 RS-232C 신호 이름과 DB-25 커넥터의 핀 번호를 보여준다.

| 신호와 의미                                        | 핀 번호 | 핀 번호 | 신호  |
|-----------------------------------------------|------|------|-----|
| SG; 신호용 접지(Signal Ground)                     | 7    | 7    | SG  |
| TxD; DTE 로부터의 송신 데이터(Transmitted Data)        | 2    | 3    | RxD |
| RxD; DCE 로부터의 수신 데이터(Received Data)           | 3    | 2    | TxD |
| RTS; DTE 가 컨트롤한 송신 요구(Request to Send)        | 4    | 5    | CTS |
| CTS; DCE 가 컨트롤한 송신 허가(Clear to Send)          | 5    | 4    | RTS |
| DTR; DTE 가 컨트롤한 단말 장치 완료(Data Terminal Ready) | 20   | 6    | DSR |
| DCD; 모뎀으로부터의 반송파 검출(Data Carrier Detect)      | 8    | 6    | DSR |
| DSR; DCE 가 컨트롤한 수신 준비 완료(Data Set Ready)      | 6    | 20   | DTR |

**Note:** "Data Set Ready(DSR)"와 "Data Carrier Detect(DCD)"는 내부적인 연결을 의미하지만 "Data Terminal Ready(DTR)"은 외부적이다.

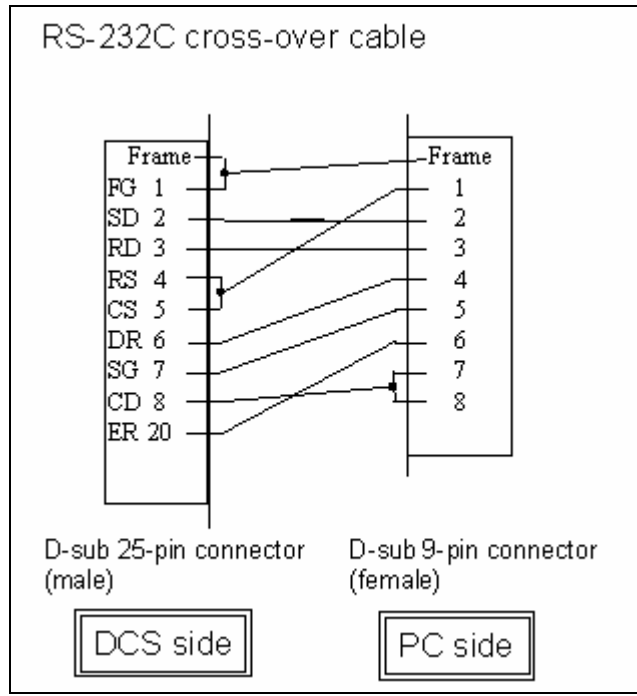
널 모뎀 케이블 핀 연결도



**2. 표준 RS-232C 케이블**

표준 시리얼 케이블은 모든 RS-232C 신호를 통과 시킨다. 케이블 끝 단의 "보낸 데이터"는 다른 쪽 끝 단에서도 "보낸 데이터"가 된다. 이것은 모뎀으로 FreeBSD 시스템에 연결하는데 사용되는 케이블 종류이고 특정 터미널용으로도 적당하다.

**RS-232C 크로스 케이블**



---

## 20.2.2.2 포트

시리얼 포트는 FreeBSD 호스트 컴퓨터와 터미널 사이의 데이터가 전송되는 장치다. 이번 섹션에서는 포트 종류와 이들 포트가 FreeBSD의 어디에 있는지 설명한다.

### [포트 종류와 포트 이름]

#### 1. 포트 종류

몇 가지 포트 종류가 있다. 케이블을 구입하거나 만들기 전에 터미널과 FreeBSD 시스템의 포트에 적당한지 확인해야 된다.

대부분의 터미널은 DB25 포트를 가지고 있다. FreeBSD를 사용 중인 PC를 포함한 개인용 컴퓨터는 DB25나 DB9 포트를 가지고 있다. PC에 멀티 포트 시리얼 카드를 가지고 있다면 RJ-12나 RJ-45 포트가 있을 것이다.

사용하는 포트 종류에 대한 설명은 하드웨어 매뉴얼 페이지를 참고한다. 포트를 육안으로 확인하는 것도 도움이 된다.

#### 2. 포트 이름

FreeBSD에서는 /dev 디렉터리의 엔트리를 거쳐 각 시리얼 포트에 접근한다. 두 가지 종류의 엔트리가 있다:

- 입력 포트(Call-in)는 /dev/ttyd*N*라고 하고, *N*은 0부터 시작하는 포트 번호다. 일반적으로 터미널에 연결할 때는 입력 포트를 사용한다. 입력 포트가 정확히 동작하려면 데이터 반송파 검출(DCD) 신호가 *on*이 되어야 한다.
- 출력 포트(Call-out)는 /dev/cuaa*N*라고 한다. 보통 출력 포트는 터미널 연결에 사용하지 않고 모뎀 연결에 사용한다. 시리얼 케이블이나 터미널이 반송파 검출 신호를 지원하지 않으면 출력 포트를 사용한다.

첫 번째 시리얼 포트(MS-DOS에서 COM1)에 터미널이 연결되어 있다면 /dev/ttyd0를 사용한다. 두 번째 시리얼 포트(MS-DOS에서 COM2)에 터미널이 있다면 /dev/ttyd1을 사용한다.

---

## 20.2.3 커널 설정

FreeBSD 는 기본적으로 4 개의 시리얼 포트를 지원한다. MS-DOS 에서는 이 포트를 COM1, COM2, COM3 와 COM4 로 부른다. FreeBSD 는 현재 Digiboard 와 Stallion Technologies 에서 만든 좀더 세련된 멀티 포트 그리고 BocaBoard 1008 과 2016 처럼 "dumb" 멀티포트 시리얼 인터페이스 카드를 지원한다. 그러나 기본 커널은 표준 COM 포트만 찾는다.

커널이 시리얼 포트를 감지하는지 확인하려면 커널이 부팅하는 동안 메시지를 확인하거나 `/sbin/dmesg` 명령으로 커널 부트 메시지를 다시 볼 수 있다. 특히 `sio` 로 시작되는 메시지를 찾는다.

**Tip:** 단어 `sio` 가 있는 메시지를 찾으려면 다음 명령을 사용한다:

```
# /sbin/dmesg | grep 'sio'
```

예를 들어 시스템에 4 개의 시리얼 포트가 있다면 시리얼 포트에 맞는 커널 부트 메시지는 다음과 같다:

```
sio0 at 0x3f8-0x3ff irq 4 on isa
sio0: type 16550A
sio1 at 0x2f8-0x2ff irq 3 on isa
sio1: type 16550A
sio2 at 0x3e8-0x3ef irq 5 on isa
sio2: type 16550A
sio3 at 0x2e8-0x2ef irq 9 on isa
sio3: type 16550A
```

커널이 모든 시리얼 포트를 감지하지 못한다면 여러분의 시스템에 맞는 FreeBSD 사용자 커널 설정이 필요할 것이다. 커널 설정에 대한 정보는 8 장을 본다.

FreeBSD 4.X 의 커널 설정파일에서 관련 장치 라인은 다음과 같다:

```
device    sio0    at isa? port IO_COM1 irq 4
device    sio1    at isa? port IO_COM2 irq 3
device    sio2    at isa? port IO_COM3 irq 5
device    sio3    at isa? port IO_COM4 irq 9
```

---

그리고 FreeBSD 5.X 에서는 아래와 비슷하다:

|        |     |
|--------|-----|
| device | sio |
|--------|-----|

FreeBSD 4.X 에서는 가지고 있지 않는 장치 리스트를 주석 처리하거나 완벽하게 삭제한다; FreeBSD 5.X 에서는 시리얼 포트를 설정하기 위해 /boot/device.hints 파일을 수정한다. 시리얼 포트와 멀티 포트 보드 설정에 대한 더 많은 정보는 sio(4) 매뉴얼 페이지를 참고한다. 이전 버전의 FreeBSD 에 맞는 설정파일을 사용했다면 장치 플래그와 구문이 버전간에 변경되었기 때문에 주의한다.

**Note:** 각 시리얼 포트에 아주 일반적으로 사용되는 포트 주소 *port 10\_COM1* 은 *port 0x3f8* 을, *IO\_COM2* 는 *0x2f8*, *IO\_COM3* 은 *0x38* 그리고 *IO\_COM4* 는 *0x2e8* 을 나타낸다; 인터럽트 4, 3, 5 와 9 는 COM1 부터 COM4 에 아주 보편적으로 사용되는 IRQ 다. 그리고 ISA 버스 PC 에서 보통 시리얼 포트는 하나의 IRQ 를 공유하지 못한다(멀티 포트 시리얼 보드는 하나나 두 개의 IRQ 를 보드의 모든 16550A 포트에 공유할 수 있다).

## 20.2.4 장치 특수 파일

대부분의 커널 장치는 /dev 디렉터리에 있는 "장치 특수 파일"을 통해 사용된다. sio 장치는 /dev/ttydM(dial-in)과 /dev/cuaaM(call-out) 장치를 통해 접근된다. 또한 FreeBSD 는 장치 초기화(/dev/ttyidN과 /dev/cuaiaN)와 잠금 장치(/dev/ttyldN 과 /dev/cual0N)도 제공한다. 장치 초기화는 흐름 제어에 RTS/CTS 신호를 사용하는 모뎀용 crtscts(하드웨어 흐름 제어)처럼 포트가 열릴 때마다 통신 포트 매개변수를 초기화하는데 사용된다. 잠금 장치는 유저나 프로그램이 특정 매개변수를 변경하지 못하도록 포트에 잠금 플래그를 사용한다; 터미널 설정, 잠금과 초기화, 터미널 설정 옵션에 대한 더 많은 정보는 termios(4), sio(4)와 stty(1) 매뉴얼 페이지를 본다.

### 20.2.4.1 장치 특수 파일 생성

/dev 디렉터리의 MAKEDEV 라는 쉘 스크립트는 장치 특수 파일을 관리한다. MAKEDEV 로 COM1(포트 0)의 다이얼-업(dial-up) 특수 파일을 만든다면 디렉터리를 /dev 로 변경하고 MAKEDEV ttyd0 명령을 사용한다. 마찬가지로 COM2(포트 1)의 다이얼 업 장치 특수



---

파일은 MAKEDEV ttyd1 를 사용한다.

**Note:** FreeBSD 5.0 은 필요할 때 자동으로 장치 노드를 생성하는 devfs(5) 파일시스템을 가지고 있다. devfs가 활성화된 이런 버전의 FreeBSD 를 사용하고 있다면 이 섹션을 지나쳐도 된다.

MAKEDEV 는 /dev/ttydN 장치 특수 파일과 /dev/cuaaN, /dev/cuaiaN, /dev/cualaN, /dev/ttyldN과 /dev/ttyidN 노드를 생성한다.

새로운 장치 특수 파일을 생성한 후 이들 장치 특수 파일에 접근할 필요가 있는 유저만 읽고 쓸 수 있도록 파일 퍼미션을 설정해야(특히 /dev/cua\* 파일들) 된다 -- 아마 보통 유저가 모뎀으로 다이얼-아웃하는 것을 원하지 않을 것이다. /dev/cua\*의 기본 퍼미션은 다음과 비슷하면 된다:

|             |        |        |                      |             |
|-------------|--------|--------|----------------------|-------------|
| crw-rw----- | 1 uucp | dialer | 28, 129 Feb 15 14:38 | /dev/cuaa1  |
| crw-rw----- | 1 uucp | dialer | 28, 161 Feb 15 14:38 | /dev/cuaia1 |
| crw-rw----- | 1 uucp | dialer | 28, 193 Feb 15 14:38 | /dev/cuala1 |

이 퍼미션은 유저 uucp 와 그룹 dialer 만 call-out(전화연결) 장치를 사용하도록 한다.

## 20.2.5 시리얼 포트 설정

ttydN(또는 cuaaN) 장치는 어플리케이션이 사용하기를 원하는 일반 장치다. 프로세스가 장치를 열면 터미널 I/O 설정의 기본 설정을 가진다. 이들 설정은 다음 명령으로 볼 수 있다:

```
# stty -a -f /dev/ttyd1
```

이 장치 설정을 변경하면 이 설정은 장치가 닫힐 때까지 유지된다. 이 장치가 다시 열리면 기본 설정으로 되돌아간다. 기본 설정으로 변경하려면 "최초 상태(initial state)" 장치 설정을 열고 조정할 수 있다. 예를 들어 ttyd5 에 CLOCAL 모드, 8bit 통신 그리고 XON/XOFF 흐름 제어를 기본적으로 키려면 다음 명령을 입력한다:

```
# stty -f /dev/ttyid5 clocal cs8 ixon ixoff
```

---

시리얼 장치는 /etc/rc.serial 파일로 시스템에서 전체적으로 초기화할 수 있다. 이 파일은 시리얼 장치의 기본 설정에도 영향을 미친다.

어플리케이션이 특정 설정을 변경하지 못하게 하려면 "잠금 상태(lock state)" 장치를 조정한다. 예를 들어 ttyd5의 속도를 57600 bps로 고정하려면 다음 명령을 입력한다:

```
# stty -f /dev/ttyd5 57600
```

이제 ttyd5를 열고 포트 속도를 변경하려는 어플리케이션은 57600 bps로 고정된다.

당연히 초기화 상태와 잠금 상태 장치는 root 계정만 쓸 수(write) 있어야 한다.

## 20.3 터미널

터미널은 여러분의 컴퓨터에 콘솔이 없거나 네트워크에 연결되어 있지 않을 때 편리하게 FreeBSD 시스템에 접근할 수 있는 방법이다. 이 섹션은 FreeBSD로 터미널을 어떻게 사용하는지 설명한다.

### 20.3.1 터미널 종류에 따른 사용

최초의 유닉스 시스템은 콘솔을 가지고 있지 않았다. 대신 사람들은 컴퓨터의 시리얼 포트에 연결된 터미널로 로그인해서 프로그램을 실행했다. 모뎀을 사용하는 터미널 소프트웨어로 원격 시스템에 다이얼-인하여 텍스트 작업만 하는 것과 매우 비슷하다.

요즘 PC는 고품질의 그래픽 콘솔 케이블을 가지고 있지만 아직도 거의 모든 유닉스 스타일 운영체제에서는 시리얼 포트에 로그인 세션을 연결하는 기능을 가지고 있다. 사용하지 않는 시리얼 포트에 연결된 터미널을 사용하여 로그인 한 후 콘솔이나 X 윈도우 시스템의 xterm에서 실행하는 텍스트 프로그램을 사용할 수 있다.

업무용 사용자를 위해 고용자들의 데스크톱에 있는 터미널들로 FreeBSD 시스템에 연결할 수 있다. 가정용 사용자는 오래된 IBM PC나 매킨토시 같은 컴퓨터를 FreeBSD가 운용되는 좀더 강력한 컴퓨터에 연결하는 터미널로 사용할 수 있다. 그렇지 않으면 싱글 유저 컴퓨터를 강력한 멀티 유저 시스템으로 바꿀 수 있을 것이다.

---

FreeBSD 에 사용할 수 있는 3 종류의 터미널이 있다:

- Dumb 터미널
- 터미널처럼 작동하는 PC
- X 터미널

남은 섹션에서는 각 터미널 종류를 설명한다.

### 20.3.1.1 Dumb 터미널

Dumb 터미널은 시리얼 라인을 통해 컴퓨터에 연결할 수 있는 전문적인 하드웨어다. 텍스트를 표시하고 보내기와 받는 기능만 있으면 되기 때문에 이들은 “dumb”이라고 부른다. dumb 에서는 어떤 프로그램도 실행할 수 없다. 텍스트 에디터, 컴파일러, e 메일, 게임 등을 할 수 있는 컴퓨터에 연결된 컴퓨터이기 때문이다.

Digital Equipment Corporations 의 VT-100 과 Wyse 의 WY-75 를 포함하여 다양한 생산자들이 만들어낸 몇 백 종류의 dumb 터미널이 있다. 이런 종류의 터미널이면 FreeBSD 와 동작할 수 있다. 어떤 고급 터미널은 그래픽까지 표시할 수 있지만 특정 소프트웨어 패키지로만 이런 기능을 사용할 수 있다.

Dumb 터미널은 X 윈도우 시스템이 제공하는 그래픽 어플리케이션이 필요 없는 작업 환경에서 유명하다.

### 20.3.1.2 터미널처럼 동작하는 PC

dumb 터미널이 텍스트를 표시하고 보내기와 받는 기능이면 충분하기 때문에 어떤 개인용 컴퓨터도 dumb 터미널처럼 사용할 수 있다. 필요한 것은 적당한 케이블과 컴퓨터에서 실행되는 터미널 에뮬레이션 소프트웨어다.

이런 설정이 가정에서 인기가 있을 것 같다. 예를 들어 여러분의 배우자가 FreeBSD 시스템 콘솔에서 바쁘게 작업하고 있다면 같은 시간에 여러분은 좀더 성능이 떨어지는 개인용 컴퓨터로 FreeBSD 터미널로 연결하여 텍스트 작업을 할 수 있다.

---

### 20.3.1.3 X 터미널

X 터미널은 사용할 수 있는 터미널 중 가장 세련된 종류다. 시리얼 포트에 연결하는 대신 보통 이더넷처럼 네트워크로 연결한다. 그리고 이 터미널은 텍스트 어플리케이션 대신 X 어플리케이션을 표현할 수 있다.

마무리하기 위해 X 터미널을 소개하였지만 이번 장에서는 X 터미널의 구축, 설정 또는 사용에 대해서는 다루지 않는다.

### 20.3.2 설정

이 섹션은 터미널을 사용하여 로그인할 수 있도록 FreeBSD 시스템의 필요한 설정에 대해 설명한다. 이미 터미널로 연결되어 있는 시리얼 포트를 지원하도록 커널을 설정하고 터미널을 연결한 것으로 간주한다.

12 장에서 시스템이 시작할 때 init 프로세스가 모든 프로세스를 제어하고 초기화하는 임무를 가지고 있다고 했다. Init 가 수행하는 태스크 중 하나는 /etc/ttys 파일을 읽은 후 사용할 수 있는 터미널에서 getty 프로세스를 시작하는 것이다. getty 프로세스는 로그인 이름을 읽고 login 프로그램을 시작하는 기능을 가지고 있다.

그래서 root 에서 FreeBSD 시스템의 터미널을 설정해야 된다:

- ① 시리얼 포트의 /dev 디렉터리 엔트리에 /etc/ttys 라인이 없다면 추가한다.
- ② /usr/libexec/getty에 실행하는 포트를 지정하고 /etc/gettytab 파일에는 적당한 getty 타입을 지정한다.
- ③ 기본 터미널 타입을 지정한다.
- ④ 포트를 "on"으로 설정한다.
- ⑤ 포트를 "secure"로 지정한다.
- ⑥ /etc/ttys 파일을 init가 다시 읽게 한다.

추가적으로 단계 ②에서 /etc/gettytab 엔트리를 생성하여 사용자 getty 타입을 생성할 수 있다. 이번 장에서는 이 방법은 설명하지 않는다; 더 많은 정보는 gettytab(5)와 getty(8) 매뉴얼 페이지를 읽는 것이 좋을 것이다.

---

### 20.3.2.1 /etc/ttys 에 엔트리 추가

/etc/ttys 파일에는 로그인을 원하는 FreeBSD 시스템의 모든 포트가 나열되어 있다. 예를 들어 첫 번째 가상 콘솔 ttyv0 는 이 파일에 엔트리를 가지고 있다. 이 엔트리를 사용하여 콘솔로 로그인할 수 있다. 또한 이 파일은 다른 가상 콘솔, 시리얼 포트 그리고 pseudo-ttys 엔트리도 가지고 있다. 물리적으로 연결된 터미널은 /dev 부분(예를 들어 /dev/ttyv0 는 ttyv0 로 나열된다)이 없고 단지 시리얼 포트의 /dev 엔트리만 나열되어 있다.

FreeBSD 를 기본값으로 설치하면 ttyd0 에서 ttyd3 까지 첫 번째 4 개의 시리얼 포트를 지원하는 /etc/ttys 파일을 가지고 있다. 이들 포트 중 한곳에 터미널로 연결한다면 다른 엔트리를 추가할 필요 없다.

#### 예제 20-1. /etc/ttys 터미널 엔트리 추가

두 개의 터미널로 시스템에 연결한다고 가정한다: Wyse-50 과 오래된 286 IBM PC 에 사용 중인 VT-100 터미널을 에뮬레이션하는 Procomm 터미널 소프트웨어. Wyse 를 두 번째 시리얼 포트에 그리고 286 을 여섯 번째 시리얼 포트(멀티 시리얼 카드의 포트)에 연결한다. /etc/ttys 파일의 일치하는 엔트리는 아래와 비슷할 것이다:

```
ttyd1 ① "/usr/libexec/getty std.38400" ② wy50 ③ on ④ insecure ⑤  
ttyd5  "/usr/libexec/getty std.19200" vt100 on insecure
```

- ① 첫 번째 필드에는 보통 /dev 에서 찾을 수 있는 터미널 특수 파일의 이름을 지정한다.
- ② 두 번째 필드는 이 라인을 실행하는 명령(보통 getty(8))이다. getty 는 초기화와 라인 오픈, 속도 설정 그리고 유저 이름을 입력하는 프롬프트를 나타내고 login(1) 프로그램을 실행한다.

getty 프로그램은 명령어 라인에 하나(임의적인)의 매개변수 *getty* 타입만 받아들인다. *getty* 타입은 터미널 라인에 bps 율과 패리티 같은 특성을 설정한다. getty 프로그램은 이런 특성을 /etc/gettytab 파일에서 읽어 들인다.

파일 /etc/gettytab 는 오래되거나 새로운 터미널 라인을 위한 수많은 엔트리를 가지고 있다. 대부분의 이 엔트리는 *std* 텍스트로 시작되며 물리적으로 연결된 터미널에 동작한다. 이들 엔트리는 패리티를 무시한다. 110 에서 115200 에 달하는 각 bps 율의 *std* 엔트리가 있다. 물론 이 파일에 원하는 엔트리를 추가할 수 있다. gettytab(5)

---

매뉴얼 페이지에서 더 많은 정보를 제공한다.

`/etc/ttys` 파일에 `getty` 타입을 설정할 때 통신 설정은 터미널과 일치해야 된다.

예제에서 Wyse-50 은 패리티를 사용하지 않고 38400 bps 속도로 연결한다. 286 PC 는 패리티를 사용하지 않고 19200 bps 속도로 연결한다.

- ③ 세 번째 필드는 일반적으로 tty 라인에 연결된 터미널 종류다. 다이얼-업(dial-up) 포트는 유저가 실제로 특정 터미널이나 소프트웨어로 다이얼-업하기 때문에 전형적으로 `unknown` 이나 `dialup` 이 이 필드에 사용된다. 물리적으로 연결된 터미널은 터미널 타입이 변하지 않기 때문에 이 필드의 `termcap(5)` 데이터베이스 파일에 실제 터미널 타입을 설정할 수 있다.

예제에서 Wyse-50 은 실제 터미널 타입을 사용하는 반면 **Procomm** 을 사용하는 286 PC 는 VT-100 을 에뮬레이트 하도록 설정한다.

- ④ 네 번째 필드는 포트를 활성화하려면 지정한다. `on` 을 설정하면 `init` 프로세스는 두 번째 필드에서 프로그램 `getty` 를 시작한다. 이 필드를 `off` 로 설정하면 `getty` 가 없을 것이고 따라서 이 포트로 로그인할 수 없다.

- ⑤ 마지막 필드는 포트가 안전한지 지정하는데 사용한다. 포트를 `secure` 로 설정하는 것은 이 포트를 통해 `root` 계정(또는 유저 ID가 0 인)이 로그인 할 만큼 안전하다는 의미다. 안전하지 않은 포트는 `root` 로그인을 허용하지 않아야 된다. `Insecure` 포트에서 유저는 특권이 없는 계정으로 로그인 한 후 슈퍼 유저 권한을 얻기 위해 `su(1)` 나 비슷한 메커니즘을 사용한다.

안전한 터미널이더라도 "insecure"를 사용할 것을 권장한다. 슈퍼 유저 권한이 필요하다면 로그인 해서 `su` 를 사용하는 것은 굉장히 간단하다.

### 20.3.2.2 /etc/ttys 를 init 가 다시 읽도록 하기

`/etc/ttys` 파일의 필요 사항을 변경하고 `init` 가 설정파일을 다시 읽도록 `SIGHUP` 신호를 보낸다:

```
# kill -HUP 1
```

---

**Note:** init 는 시스템에서 처음으로 시작되는 프로세스이기 때문에 항상 PID 1 을 가진다.

설정이 모두 정확하고 모든 케이블이 연결되어 있으며 터미널 전원이 들어왔다면 `getty` 프로세스는 각 터미널에 실행되고 이쯤에서 로그인 프롬프트를 터미널에서 볼 수 있다.

### 20.3.3 연결 문제 해결

모든 설정을 정확히 체크했다라도 터미널을 설정하는 동안 문제가 있을 수 있다. 여기 증상과 해결을 위한 몇 가지 방법이 있다.

#### 20.3.3.1 로그인 프롬프트가 나타나지 않는다.

터미널이 연결되어 있고 전원이 들어왔는지 확인한다. 개인용 컴퓨터를 터미널처럼 사용한다면 정확한 시리얼 포트에서 터미널 에뮬레이션 소프트웨어가 동작하는지 확인한다.

또한 케이블이 정확히 터미널과 FreeBSD 컴퓨터에 연결되어 있는지 그리고 케이블 종류가 맞는지 확인한다.

터미널과 FreeBSD 의 bps 을 그리고 패리티 설정이 일치하는지 확인한다. 그래픽을 표시하는 터미널을 가지고 있다면 명암과 밝기가 조정됐는지 체크한다. 터미널을 프린터로 출력한다면 종이와 잉크가 공급되는지 체크한다.

`getty` 프로세스가 실행되고 터미널을 지원하는지 체크한다. 예를 들어 실행 중인 `getty` 프로세스 리스트를 `ps` 로 보려면 다음과 같이 입력한다:

```
# ps -axww|grep getty
```

터미널 엔트리를 볼 수 있다. 예를 들어 다음 화면은 두 번째 시리얼 포트 `ttyd1` 에서 `getty` 가 동작하고 `/etc/gettytab` 에 `std.38400` 엔트리를 사용하는 것을 보여 준다:

```
22189  d1  ls+   0:00.03 /usr/libexec/getty std.38400 ttyd1
```

`getty` 프로세스가 동작하지 않는다면 `/etc/ttys` 에서 포트를 활성화했는지 체크한다. 그리고 `ttys` 파일을 수정한 후 `kill -HUP 1` 을 실행하는 것을 잊지 않는다.

---

getty 프로세스가 동작하지만 터미널이 로그인 프롬프트를 표시하지 않거나 프롬프트는 표시되지만 입력할 수 없다면 터미널이나 케이블이 하드웨어 신호 변경(handshaking)을 지원하지 못하는 것이다. /etc/ttyd의 엔트리를 *std.38400*에서 *3wire.38400*으로 변경하고 kill -HUP 1을 실행한다. *3wire* 엔트리는 *std*와 비슷하지만 하드웨어 신호 변경을 무시한다. *3wire*를 사용할 때 버퍼 오버플로우를 방지하기 위해 보드율(baud rate)이나 소프트웨어 흐름 제어를 활성화해야 된다.

### 20.3.3.2 로그인 프롬프트 대신 알 수 없는 문자가 나타난다면.

터미널과 FreeBSD에서 bps율과 패리티 설정이 정확한지 확인한다. 정확한 getty 종류를 사용하고 있는지 getty 프로세스를 체크한다. 그렇지 않으면 /etc/ttyd를 수정하고 kill -HUP 1을 실행한다.

### 20.3.3.3 문자가 두 번씩 나타난다; 입력 했을 때 패스워드가

나타난다.

터미널을 "half duplex"나 "local echo"에서 "full duplex"로(또는 터미널 에뮬레이션 소프트웨어를) 변경한다.

## 20.4 다이얼-인(Dial-in) 서비스

다이얼-인 서비스용으로 FreeBSD 시스템을 설정하는 것은 터미널 대신 모뎀으로 연결하는 것을 제외하고 터미널 연결과 매우 비슷하다.

### 20.4.1 외장형 모뎀 VS 내장형 모뎀

외장형 모뎀은 가끔 비 휘발성 메모리에 저장된 매개변수로 반 영구적으로 설정할 수 있고 보통 중요한 RS-232 신호 상태를 나타내는 발광소자를 제공한다. 그래서 외장형 모뎀이 다이얼-업 연결에 매우 편리해 보인다. 점멸등은 데이터 흐름 상태를 나타내지만 모뎀이



---

정확하게 동작하는지 확인하는데도 유용하다.

내장형 모뎀은 보통 비 휘발성 메모리가 부족하여 설정에 한계가 있어서 DIP 스위치로만 설정한다. 내장형 모뎀에 신호를 표시하는 램프가 있더라도 시스템의 뚜껑을 닫았을 때 이 램프를 보기가 어려울 것이다.

### 20.4.1.1 모뎀과 케이블

외장형 모뎀을 사용한다면 적당한 케이블이 필요하다. 표준 RS-232C 시리얼 케이블이면 일반적인 모든 신호에 적당하다:

- Transmitted Data (SD)
- Received Data (RD)
- Request to Send (RTS)
- Clear to Send (CTS)
- Data Set Ready (DSR)
- Data Terminal Ready (DTR)
- Carrier Detect (CD)
- Signal Ground (SG)

FreeBSD 는 2400bps 이상의 전송 속도에서 흐름제어에 RTS 와 CTS 신호가 필요하다. CD 신호는 콜이 응답되거나 라인 끊김을 감지하기 위해 그리고 DTR 신호는 세션이 끝난 후 모뎀을 리셋(초기화) 하기 위해 필요하다. 어떤 케이블은 필요한 신호가 모두 없어도 연결되기 때문에 라인이 끊어졌을 때 로그인 세션이 없어지지 않는 것과 같은 문제가 발생한다면 케이블에 문제가 있을 것이다.

다른 유닉스와 유사한 운영체제처럼 FreeBSD 도 콜이 응답되거나 라인이 언제 끊어지고 콜 후에 모뎀이 리셋 됐는지 감지하기 위해 하드웨어 신호를 사용한다. FreeBSD 는 직접 모뎀에 명령하거나 모뎀의 상태 리포트를 받지 않는다.

### 20.4.2 시리얼 인터페이스에 대해

FreeBSD 는 NS8250-, NS16450-, NS16550- 그리고 NS16550A-기반 EIA RS-232C(CCITT V.24) 통신 인터페이스를 지원한다. 8250 과 16450 장치는 싱글 캐릭터

---

버퍼를 가지고 있다. 16550 장치는 좀더 좋은 시스템 성능을 제공하는 16 캐릭터 버퍼를 제공한다. 싱글 캐릭터 버퍼 장치는 16 캐릭터 버퍼 장치보다 운영체제에 더 많은 부하를 주기 때문에 16550A-기반 시리얼 인터페이스 카드를 더 선호한다. 여러 개의 시리얼 포트를 시스템에서 사용하거나 부하가 많이 발생한다면 16550A- 기반 카드를 사용하여 통신의 에러 발생율을 낮출 수 있다.

### 20.4.3 개요

터미널에서 init 는 다이얼-인(dial-in) 연결을 위해 설정된 각 시리얼 포트에 getty 프로세스를 생성한다. 예를 들어 모뎀이 /dev/ttyd0 에 연결되어 있다면 명령어 ps ax 는 다음과 같은 결과일 것이다:

```
4850 ?? |      0:00.09 /usr/libexec/getty V19200 ttyd0
```

유저가 모뎀에 전화를 걸어 모뎀끼리 연결되면 CD(반송파 검출) 라인이 모뎀에 의해 통보된다. 커널은 캐리어가 감지되었고 gettys 가 포트를 열었다고 알린다. getty 는 최초로 지정된 라인 속도로 login: 프롬프트를 보낸다. 그리고 적절한 문자가 입력되는지 대기하고 일반적인 설정에서 이상한 신호를 받으면 getty 는 적절한 문자를 입력 받을 때까지 라인 속도를 조정한다.

유저가 로그인 이름을 입력하면 getty 는 유저의 패스워드를 물어보고 완벽히 로그인한 후 유저의 셸을 시작하는 /usr/bin/login 을 실행한다.

### 20.4.4 파일 설정

/etc 디렉터리에는 FreeBSD 시스템에 다이얼-업 연결을 허용하기 위해 수정해야 되는 3 개의 시스템 설정파일이 있다. 첫째로 /etc/gettytab 는 /usr/libexec/getty 데몬의 설정 정보를 가지고 있다. 그리고 /etc/ttys 는 어떤 tty 장치에 getty 프로세스를 실행해야 되는지 /sbin/init 에게 알려주는 정보를 가지고 있다. 마지막으로 /etc/rc.serial 스크립트에 포트 초기화 명령을 넣을 수 있다.

유닉스 시스템에 다이얼-업 모뎀을 고려해야 되는 두 학교가 있다. 한쪽 그룹은 모뎀과 시스템 설정을 좋아해서 원격유저가 어떤 속도로 전화연결 하는지 상관없기 때문에 로컬 컴퓨터와 모뎀의 RS-232 인터페이스를 고정된 속도로 실행한다. 이 설정의 장점은 원격

---

유저가 항상 로그인 프롬프트를 즉시 볼 수 있다는 것이다. 단점은 유저의 실제 데이터율이 어떤지 시스템이 모르기 때문에 Emacs 같은 풀(Full) 스크린 프로그램은 느린 연결에 맞도록 화면 표시 방법을 조정하지 않는다.

다른 학교는 모뎀의 RS-232 인터페이스를 원격유저의 연결 속도에 맞춘다. 예를 들어 V.32bis(14.4 kbps)로 모뎀이 연결하면 모뎀 RS-232 인터페이스를 19.2 kbps 에서 실행하고 2400 bps 로 연결하면 모뎀의 RS-232 인터페이스를 2400 bps 에서 실행한다. getty 는 특정 모뎀의 연결 속도에 대해 이해하지 못하기 때문에 getty 는 최초의 속도로 login: 메시지를 주고 응답해오는 문자를 확인한다. 유저가 이상한 메시지를 보게 된다면 적절한 프롬프트를 볼 때까지 Enter 키를 눌러야 된다. 데이터 전송율이 맞지 않다면 getty 는 유저가 입력하는 모든 것을 쓸데없는 문자로 인식하고 그 다음 속도로 login: 프롬프트를 보여준다. 이 절차가 계속되면 지겨울 수 있지만 보통 한 두 번의 키 입력으로 유저는 적절한 프롬프트를 볼 수 있다. 물론 이 로그인 순서가 이전의 "속도를 고정시킨" 방법만큼 깨끗해 보이지 않지만 낮은 속도로 연결된 유저는 풀 스크린 프로그램에서 더 좋은 응답을 얻게 된다.

이 섹션에서 균형 잡힌 설정 정보를 보여 주려고 노력했지만 모뎀의 데이터 전송율에 치우친 경향이 있다.

#### 20.4.4.1 /etc/gettytab

/etc/gettytab 은 getty(8)의 설정 정보로 termcap(5) 스타일 파일이다. 파일의 포맷과 기능에 관한 완벽한 정보는 gettytab(5) 매뉴얼 페이지를 본다.

##### [고정된 속도와 연결 속도에 맞추는 설정]

###### 1. 고정된 속도 설정

모뎀과 컴퓨터간의 통신 속도를 특정 속도로 고정했다면 /etc/gettytab 를 변경할 필요가 없을 것이다.

###### 2. 연결 속도에 맞추는 설정

모뎀과 컴퓨터의 연결 속도에 대한 정보를 getty 에게 주기 위해 /etc/gettytab 엔트리를 설정한다. 2400bps 모뎀을 가지고 있다면 2400D 엔트리를 사용할 수 있을 것이다.

```
#
# Fast dialup terminals, 2400/1200/300 rotary (can start either way)
#
D2400|d2400|Fast-Dial-2400:W
:nx=D1200:tc=2400-baud:
3|D1200|Fast-Dial-1200:W
:nx=D300:tc=1200-baud:
5|D300|Fast-Dial-300:W
:nx=D2400:tc=300-baud:
```

고속 모뎀을 가지고 있다면 /etc/gettytab 에 엔트리를 추가해야 될 것이다; 다음 엔트리는 14.4kbps 모뎀을 인터페이스의 최고 속도 19.2kbps 로 사용하기 예제다:

```
#
# Additions for a V.32bis Modem
#
um|V300|High Speed Modem at 300,8-bit:W
:nx=V19200:tc=std.300:
un|V1200|High Speed Modem at 1200,8-bit:W
:nx=V300:tc=std.1200:
uo|V2400|High Speed Modem at 2400,8-bit:W
:nx=V1200:tc=std.2400:
up|V9600|High Speed Modem at 9600,8-bit:W
:nx=V2400:tc=std.9600:
uq|V19200|High Speed Modem at 19200,8-bit:W
:nx=V9600:tc=std.19200:
```

이 설정을 사용하면 패리티 없이 8-bit 로 연결한다.

위의 예제는 19.2 kbps(V.32bis) 전송율로 모뎀과 컴퓨터의 연결을 시도하여 9600 bps, 2400 bps, 1200 bps, 300 bps 그리고 다시 19.2 kbps 으로 순환된다. 전송율 사이클은 *nx*="next table") 구문으로 수행된다. 각 라인은 특정 전송율에 나머지 "표준" 설정을 선택하도록 *tc*="table Continuation") 엔트리를 사용한다.

28.8 kbps 모뎀을 가지고 있거나 14.4 kbps 모뎀에서 압축하기를 원한다면 19.2 kbps 보다 높은 전송율을 사용해야 된다. 아래에 57.6 kbps 로 연결을 시도하는 gettytab

엔트리 예제가 있다:

```
#
# Additions for a V.32bis or V.34 Modem
# Starting at 57.6 Kbps
#
vm|VH300|Very High Speed Modem at 300,8-bit:W
:nx=VH57600:tc=std.300:
vn|VH1200|Very High Speed Modem at 1200,8-bit:W
:nx=VH300:tc=std.1200:
vo|VH2400|Very High Speed Modem at 2400,8-bit:W
:nx=VH1200:tc=std.2400:
vp|VH9600|Very High Speed Modem at 9600,8-bit:W
:nx=VH2400:tc=std.9600:
vq|VH57600|Very High Speed Modem at 57600,8-bit:W
:nx=VH9600:tc=std.57600:
```

CPU 가 느리거나 부하가 많이 걸리는 시스템에 16550A 기반 시리얼 포트가 없다면 57.6kbps 연결에서 sio "silo" 에러를 보게 될 것이다.

#### 20.4.4.2 /etc/ttys

/etc/ttys 파일의 설정에 대해서는 예제 20-1 에서 다루었다. 모뎀 설정은 비슷하지만 getty 에 다른 인자를 주고 터미널 타입도 다르게 지정해야 된다. 고정된 속도와 연결 속도에 맞추는 설정의 일반적인 포맷은 다음과 같다:

```
ttyd0 "/usr/libexec/getty xxx" dialup on
```

위 라인의 첫 번째 내용은 이 엔트리의 장치 특수 파일이다. ttyd0 는 이 getty 가 주시하는 파일이 /dev/ttyd0 라는 의미다. 두 번째 내용 *"/usr/libexec/getty xxx"* (xxx 는 최초의 gettytab 문자로 대체된다)는 init 가 장치에 실행할 프로세스다. 세 번째 내용 *dialup* 은 기본 터미널 타입이다. 네 번째 매개변수 *on* 은 init 에게 운용중인 라인임을 알려준다. 다섯 번째 매개변수에 *secure* 가 있을 수 있지만 물리적으로 안전한(시스템 콘솔 같은) 터미널에만 사용해야 된다.

---

기본 터미널 타입(위의 예제에서 *dialup*)은 로컬 유저의 설정(preference)에 따라 달라진다. *dialup*은 다이얼-업 회선에 전통적으로 기본 터미널 타입이기 때문에 유저는 터미널이 *dialup*이면 통보하고 자동으로 터미널 타입을 조정하도록 로그인 스크립트를 수정할 것이다. 그러나 필자의 사이트에서는 대부분의 유저가 VT102 에뮬레이션을 사용하기 때문에 기본 터미널 타입을 *vt102*로 지정하였다.

*/etc/ttys*를 변경한 후 파일을 다시 읽도록 다음 명령으로 HUP 신호를 *init* 프로세스에게 보낸다.

```
# kill -HUP 1
```

시스템을 최초로 설정하는 것이라면 *init* 신호를 보내기 전에 모뎀이 적절히 설정되고 연결될 때까지 기다려야 한다.

## [고정된 속도와 연결 속도에 맞추는 설정]

### 1. 고정된 속도 설정

속도를 고정시키도록 설정하려면 *getty*에 제공되는 고정된 속도 엔트리가 *ttys*에 필요하다. 포트 속도가 19.2 kbps로 고정된 모뎀의 *ttys* 엔트리는 다음과 비슷할 것이다:

```
ttyd0 "/usr/libexec/getty std.19200" dialup on
```

모뎀이 다른 데이터 전송율로 고정되어 있다면 *std.19200* 대신 *std.speed*를 적당한 값으로 변경한다. */etc/gettytab*에 나열된 유효한 타입을 사용해야 된다.

### 2. 연결 속도에 맞추는 설정

연결 속도에 맞추는 설정에서 *ttys* 엔트리는 */etc/gettytab*의 "auto-baud"(sic)로 시작되는 적절한 엔트리를 참조해야 된다. 예를 들어 위에서 제시된 엔트리를 19.2Kbps(*gettytab* 엔트리는 *V19200* 시작 위치를 포함한다)로 시작하는 연결 속도에 맞추는 모뎀용으로 추가한다면 *ttys* 엔트리는 다음과 비슷할 것이다:

```
ttyd0 "/usr/libexec/getty V19200" dialup on
```

---

### 20.4.4.3 /etc/rc.serial

V.32, V.32bis 그리고 V.34 같은 고속 모뎀은 하드웨어(RTS/CTS) 흐름 제어를 사용해야 된다. FreeBSD 커널에서 모뎀 포트에 하드웨어 흐름 제어 플래그를 지정하려면 `stty` 명령을 `/etc/rc.serial` 에 추가한다.

예를 들어 `termios` 플래그 `crtcscts` 를 시리얼 포트 #1(COM2)의 다이얼-인(dial-in)과 다이얼-아웃(dial-out) 초기화 장치에 설정하려면 다음 라인을 `/etc/rc.serial` 에 추가한다:

```
# Serial port initial configuration
stty -f /dev/ttyid1 crtcscts
stty -f /dev/cuaia1 crtcscts
```

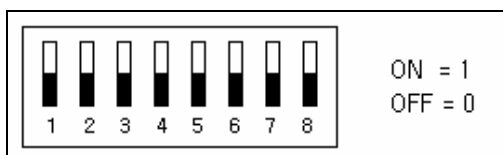
### 20.4.5 모뎀 설정

매개변수가 비 휘발성 메모리에 영구적으로 설정되는 모뎀을 가지고 있다면 매개변수를 설정하기 위해 터미널 프로그램(MS-DOS 에서 `Telix` 또는 FreeBSD 에서 `tip` 과 같은)을 사용해야 된다. `getty` 가 사용하는 최초의 속도로 모뎀을 연결하고 다음과 같은 요구 사항에 맞도록 모뎀의 비 휘발성 메모리를 설정한다:

- 연결하면 CD 신호가 *on*.
- 연결하면 DTR이 *on*; 사용이 끝나면 DTR이 *off* 되어 연결을 끊고 모뎀을 리셋한다.
- 송신할 때 데이터 흐름제어에 CTS 사용.
- XON/XOFF(흐름제어에 사용되는 프로토콜) 흐름 제어를 비활성.
- 수신할 때 데이터 흐름 제어에 RTS 사용.
- Quite mode(결과 코드를 돌려주지 않음).
- 명령 에코를 돌려주지 않음.

어떤 명령이나 DIP 스위치로 위와 같이 설정하는지 모뎀 매뉴얼을 확인한다.

#### DIP 스위치 설정



---

예를 들어 위의 매개변수를 USRobotics Sportster 14,400 외부 모뎀에 설정하려면 다음 명령을 모뎀에 보낸다:

**ATZ**

**AT&C1&D2&H1&I0&R2&W**

V.42bis 또는 MNP5 압축을 사용하는 것처럼 모뎀의 다른 것을 조정하여 위와 같이 설정할 수 있다.

USR Sportster 14,400 외장형 모뎀도 설정해야 되는 몇 개의 DIP 스위치를 가지고 있다; 다른 모뎀에는 다음 예를 사용할 수 있을 것이다:

- Switch 1: UP -- 표준 DTR.
- Switch 2: N/A (문자형식 결과 코드/수치형식 결과 코드).
- Switch 3: UP -- 결과 코드를 돌려주지 않음.
- Switch 4: DOWN -- 라인 off 명령의 에코를 돌려주지 않음.
- Switch 5: UP -- 자동 착신
- Switch 6: UP -- 표준 CD.
- Switch 7: UP -- 비 휘발성 램에서 기본값 로드.
- Switch 8: N/A (Smart Mode/Dumb Mode)

getty 가 실수로 명령어 모드에 있는 모뎀에게 login: 프롬프트를 보내서 모뎀이 명령어나 결과 코드를 되돌려주지 않도록 결과코드를 비활성 해야 된다. 이 결과 getty 와 모뎀의 통신 설정이 잘못될 수 있다.

### 20.4.5.1 고정된 속도로 설정

고정된 속도로 설정하려면 통신 상태와 무관하면서 모뎀과 컴퓨터의 데이터 율을 일정하게 유지하도록 모뎀을 설정해야 된다. USR Sporster 14,400 외장형 모뎀에서 다음 명령은 모뎀과 컴퓨터 데이터 율을 이 명령의 속도로 고정한다:

**ATZ**

**AT&B1&W**



---

## 20.4.5.2 연결 속도에 맞추는 설정

유동적인 속도로 설정하기 위해 모뎀의 시리얼 포트 데이터 비율을 입력 콜 비율에 대응되도록 조정해야 된다. USR Sportster 14,400 외장형 모뎀에서 다음 명령은 모뎀의 error-corrected 데이터 비율을 사용된 명령의 속도로 고정시키지만 시리얼 포트 비율은 non-error-corrected 연결에 맞도록 수정한다.

ATZ

AT&B2&W

## 20.4.5.3 모뎀 설정 체크

대부분의 고속 모뎀은 모뎀의 현재 운용 매개변수를 사람이 읽을 수 있는 명령으로 제공한다. USR Sportster 14,400 외장형 모뎀에서 명령 AT15는 비 휘발성 메모리에 저장되어 있는 설정을 보여준다. 모뎀의 실제 운용 매개변수를 보려면 ATZ와 AT14 명령을 사용한다.

다른 회사의 모뎀을 가지고 있다면 모뎀의 매개변수 설정을 어떻게 체크할 수 있는지 매뉴얼 페이지를 본다.

## 20.4.6 문제해결

시스템의 전화연결 모뎀을 체크할 수 있는 몇 가지 단계가 있다.

### 20.4.6.1 FreeBSD 시스템 체크

모뎀을 FreeBSD에 장착한 후 시스템이 부팅해서 모뎀의 상태를 나타내는 표시기가 발광한다면, 시스템 콘솔에 login: 프롬프트가 나타났을 때 모뎀의 DTR 표시기가 발광하는지 확인한다—발광한다면 이것은 FreeBSD의 적절한 통신 포트에 getty 프로세스가 시작되었고 모뎀의 허가를 기다린다는 의미다.

DTR 표시기가 발광하지 않는다면 FreeBSD 시스템에 콘솔로 로그인해서 FreeBSD가 정확한 포트에 getty 프로세스를 실행하는지 ps ax 명령으로 본다. 프로세스가 표시되는 동안 다음과 비슷한 라인을 볼 수 있다:

---

```
114 ?? |      0:00.10 /usr/libexec/getty V19200 ttyd0
115 ?? |      0:00.10 /usr/libexec/getty V19200 ttyd1
```

이것과 약간 다른 다음과 같은 라인을 보게 되면 모뎀이 아직 콜을 허용하지 않았다는 것이다. :

```
114 d0 |      0:00.10 /usr/libexec/getty V19200 ttyd0
```

이 말은 getty 가 통신 포트를 완벽하게 열었다는 것이다. getty 는 CD(반송파 검출) 신호가 on 이 될 때까지 통신 포트를 열수 없기 때문에 이것은 케이블이나 잘못된 모뎀 설정문제를 보여줄 수 있다.

요청한 ttydN 포트를 열 때까지 getty 프로세스가 기다리지 않는다면 /etc/ttys 엔트리를 다시 확인한다. 또한 init 나 getty 로부터 문제에 관한 로그 메시지가 없는지 /var/log/messages 파일을 확인한다. 메시지가 있다면 실수나 잘못된 엔트리 또는 잘못된 장치 특수 파일이 있는지 설정파일 /etc/ttys 와 /etc/gettytab 을 다시 확인하고 적절한 장치 특수 파일 /dev/ttydN도 확인한다.

## 20.4.6.2 다이얼-인 하기

시스템에 다이얼-인(dial-in)하려면 원격 시스템에 8 bits, 패리티 없이 1 stop bit 를 사용한다. 즉시 프롬프트가 나타나지 않거나 잘못된 문자가 나오면 초당 한번씩 Enter 를 눌러본다. 몇 초 후까지 login: 프롬프트를 볼 수 없다면 BREAK 신호를 보낸다. 전화연결(dialing)에 고속 모뎀을 사용한다면 전화연결 모뎀의 인터페이스 속도(예를 들어 USR Sportster 모뎀에서 AT&B1 로)를 고정시킨 후 다시 연결한다.

그래도 login: 프롬프트를 볼 수 없다면 /etc/gettytab 을 다시 확인하고 다음 사항도 역시 체크한다.

- /etc/ttys에 지정된 이름이 /etc/gettytab에 있는 이름과 일치하는지 확인.
- 각 nx=엔트리가 다른 gettytab 문자 이름과 일치하는지 확인.
- 각 tc=엔트리가 다른 gettytab 문자 이름과 일치하는지 확인.

다이얼-인을 했지만 FreeBSD 시스템의 모뎀이 응답하지 않는다면 DTR 이 on 이 되었을 때

---

응답하도록 모뎀을 설정했는지 확인한다. 모뎀이 정확하게 설정되었다면 DTR 이 *on* 이 되었는지 모뎀의 상태 표시기를 체크한다.

많은 시간을 허비했지만 아직도 작동하지 않는다면 잠시 휴식 후 나중에 다시 확인한다. 역시 작동하지 않으면 FreeBSD 일반적인 질문 메일링 리스트에 메일을 보내서 모뎀과 문제에 대해 설명하면 여러 사람들이 도와줄 것이다.

## 20.5 다이얼-아웃(Dial-out) 서비스

다음 내용은 다른 컴퓨터에 모뎀으로 연결할 수 있도록 여러분의 호스트를 설정하는 것이다. 원격 호스트에 터미널로 연결하기 적당해서 BBS 에 로그인 하는데 유용하다.

이런 종류의 연결은 PPP 에 문제가 있고 인터넷에서 파일을 다운로드 할 때 유용하다. FTP 가 필요하지만 PPP 연결이 끊어졌다면 터미널 세션을 FTP 로 사용한다. 그리고 Z 모뎀을 사용하여 이 파일을 전송한다.

### 20.5.1 Stock Hayes 모뎀이 지원되지 않는다. 어떻게 해야 되는가?

사실 매뉴얼 페이지의 `tip`(`tip` 명령은 다른 머신에 full-duplex 로 연결한다)은 유효기간이 지났다. 일반적인 Hayes 다이얼러(dialer)는 이미 내장되어 있다. 그냥 `/etc/remote` 파일에 `at=hayes` 라고 사용하면 된다.

Hayes 드라이버는 새로운 모뎀(*BUSY, NO DIALTONE* 이나 *CONNECT 115200* 과 같은 메시지는 혼란스러울 것이다)의 특정 기능을 사용하기에 너무 단순하다. 이런 메시지는 `tip(ATX0&W` 을 사용하여)을 사용할 때 끄도록 한다.

그리고 `tip` 의 다이얼 타임아웃이 60 초기 때문에 모뎀의 타임 아웃 설정을 이보다 짧게 해야 되고 그렇지 않으면 `tip` 은 통신문제라고 판단한다. `ATS7=45&W` 로 다시 시도한다.

**Note:** 이러한 이유로 `tip` 은 아직 완벽하게 Hayes 모뎀을 지원하지 못한다.  
해결책은 `/usr/src/usr.bin/tip/tip` 디렉터리의 `tipconf.h` 파일을 수정해야 되고

---

파일을 수정하려면 배포된 소스가 필요하다.

라인 `#define HAYES 0`을 `#define HAYES 1`로 변경한다. 그리고 `make` 하고 `make install` 한다. 이후 모든 것이 완벽하게 동작한다.

## 20.5.2 이들 AT 명령을 어떻게 입력하는가?

`/etc/remote` 파일에 "direct"라는 엔트리를 만든다. 예를 들어 모뎀이 첫 번째 시리얼 포트 `/dev/cuaa0` 에 연결되어 있다면 다음 라인을 넣는다:

```
cuaa0:dv=/dev/cuaa0:br#19200:pa=none
```

`br` 문자에 모뎀이 지원하는 최고 bps 을 사용한다. 그리고 `tip cuaa0` 를 입력하면 모뎀에 연결된다.

시스템에 `/dev/cuaa0` 가 없다면 다음 명령을 실행한다:

```
# cd /dev
# sh MAKEDEV cuaa0
```

아니면 `root` 에서 `cu` 로 다음 명령을 수행한다:

```
# cu -lline -sspeed
```

*line* 은 시리얼 포트(예: `/dev/cuaa0`)를 그리고 *speed* 는 속도를 의미한다(예: 57600). AT 명령 입력을 끝내고 나가기 위해 `~.`을 입력한다.

## 20.5.3 pn 기능의 @ 기호가 동작하지 않는다!

전화 번호 기능 중에서 `@` 기호는 `tip` 이 전화번호를 `/etc/phones` 에서 찾도록 한다. 그러나 `@` 기호도 `/etc/remote` 와 같은 파일 문자열의 특수 문자다. 백슬러시로 빠져 나오는 다음 명령을 사용한다:

```
pn=W@
```

---

## 20.5.4 명령어 라인에서 전화 번호는 어떻게 입력하는가?

"generic"라는 엔트리를 /etc/remote 파일에 넣는다. 예를 들어 다음과 같이 추가한다:

```
tip115200|Dial any phone number at 115200 bps:W
:dv=/dev/cuaa0:br#115200:at=hayes:pa=none:du:
tip57600|Dial any phone number at 57600 bps:W
:dv=/dev/cuaa0:br#57600:at=hayes:pa=none:du:
```

그리고 다음 명령을 실행한다:

```
# tip -115200 5551234
```

tip 대신 cu 를 사용하려면 일반적인 cu 엔트리를 사용한다:

```
cu115200|Use cu to dial any number at 115200bps:W
:dv=/dev/cuaa1:br#57600:at=hayes:pa=none:du:
```

그리고 다음 명령을 실행한다:

```
# cu 5551234 -s 115200
```

## 20.5.5 항상 bps 율을 입력해야 되는가?

*tip1200* 또는 *cu1200* 엔트리를 입력하여 적절한 통신 속도를 br 문자열로 설정한다. tip 은 가장 알맞은 기본값을 1200bps 로 알고 있기 때문에 *tip1200* 엔트리를 찾는다. 그러나 1200 bps 로 사용할 필요는 없다.

## 20.5.6 터미널 서버를 경유하여 다양한 호스트에 접근한다.

연결될 때까지 기다려서 CONNECT <host>를 매번 입력하지 않고 tip 의 *cm* 문자열을 사용한다. 예를 들어 /etc/remote 의 엔트리는 다음과 비슷하다:

---

```
pain|pain.deep13.com|Forrester's machine:W
:cm=CONNECT painWn:tc=deep13:
muffin|muffin.deep13.com|Frank's machine:W
:cm=CONNECT muffinWn:tc=deep13:
deep13:Gizmonics Institute terminal server:W
:dv=/dev/cuaa2:br#38400:at=hayes:du:pa=none:pn=5551234:
```

위의 설정으로 tip pain 이나 tip mffin 을 실행하면 호스트 pain 이나 muffin 에 연결할 수 있기 때문에 tip deep13 을 입력하여 터미널 서버에 연결한다.

## 20.5.7 각 사이트에 연결하기 위해 여러 회선에 tip 을 사용할 수 있는가?

이 방법은 대학교에 여러 개의 모뎀 라인이 있고 수 천명의 학생이 이 라인을 이용할 때 종종 문제가 되고 있다.

이 대학교를 위해 /etc/remote 에 엔트리를 생성하고 *pn* 문자열에 @를 사용한다:

```
big-university:W
:pn=W@:tc=dialout
dialout:W
:dv=/dev/cuaa3:br#9600:at=courier:du:pa=none:
```

그리고 대학교 전화번호를 /etc/phones 에 나열한다:

```
big-university 5551111
big-university 5551112
big-university 5551113
big-university 5551114
```

tip 은 나열된 순서대로 연결을 시도하여 연결할 수 없으면 포기한다. 재 시도를 원하면 tip 을 while 문으로 실행한다.

---

## 20.5.8 Ctrl+P 을 한번 보내기 위해 왜 두 번의 Ctrl+P 를 눌러야 하는가?

Ctrl+P 는 tip 에게 다음 문자가 문자 데이터임을 명시해주는 기본적인 “force” 문자다. “변수를 설정한다”는 의미를 가진 ~s 문자로 force 문자대신 다른 문자를 설정할 수 있다.

~sforce=*single-char* 를 입력하여 다음 라인으로 변경한다. *single-char* 은 임의적인 1 바이트 문자다. *single-char* 을 생략하면 force 문자는 **Ctrl+2** 나 **Ctrl+Space** 를 입력하여 생성하는 널(nul) 문자다.

*single-char* 에 적당한 값은 몇몇 터미널 서버에서만 사용하는 **Shift+Ctrl+6** 다.

\$HOME/.tiprc 파일에 다음 라인을 명시하여 원하는 문자를 force 문자로 입력할 수 있다:

```
force=<single-char>
```

## 20.5.9 갑자기 입력하는 모든 것이 대문자가 됐다?

Caps-Lock 키가 깨진 사람들을 위해 특별히 디자인된 tip 의 “raise character”인 **Ctrl+A** 를 누른 것이다. 위의 ~s 를 사용하고 변수 *raisechar* 를 적절히 설정한다. 이러한 기능을 절대 사용하지 않는다면 force 문자와 같이 설정할 수 있다.

다음은 **Ctrl+2** 와 **Ctrl+A** 를 자주 입력하는 Emacs 유저를 위한 .tiprc 파일 샘플이 있다:

```
force=^^  
raisechar=^^
```

^^는 **Shift+Ctrl+6** 다.

## 20.5.10 tip 으로 파일은 어떻게 전송할 수 있는가?

다른 유닉스 시스템에는 ~p(put)과 ~t(take)로 파일을 보내고 받을 수 있다. 이들 명령은

---

원격 시스템에 접근해서 파일을 보낼 때 `cat` 과 `echo` 를 실행한다. 구문은 다음과 같다:

`~p` 내부 파일 [원격 파일]

`~t` 원격 파일 [내부 파일]

에러 체크가 없기 때문에 `z` 모뎀과 같은 다른 프로토콜을 사용해야 된다.

## 20.5.11 tip 으로 `z` 모뎀은 어떻게 실행하는가?

파일을 받으려면 원격 시스템에서 송신 프로그램을 실행한다. 그리고 내부에서 `~C rz` 를 입력하면 파일 수신이 시작된다.

파일을 보내려면 원격 시스템에서 수신 프로그램을 실행한다. 그리고 `~C sz` *파일이름* 을 입력하면 원격 시스템으로 파일 송신이 시작된다.

## 20.6 시리얼 콘솔 설정

### 20.6.1 소개

FreeBSD 는 콘솔처럼 시리얼 포트의 `dumb` 터미널로만 시스템을 부팅할 수 있다. 이러한 설정은 두 가지 유형의 사람들에게 유용하다: 키보드가 없거나 모니터가 없는 머신에 FreeBSD 를 설치하려는 시스템 관리자와 커널이나 장치 드라이버 버그를 수정하려는 개발자.

12 장에서 설명했듯이 FreeBSD 는 3 단계 부트스트랩을 사용한다. 처음 두 단계는 부트디스크의 FreeBSD 슬라이스 시작부분에 저장되어있는 부트 블록 코드에 있다. 부트 블록은 세 번째 단계의 코드인 부트 로더(`/boot/loader`)를 로드하고 실행한다.

시리얼 콘솔을 설정하기 위해 부트 블록 코드, 부트 로더 코드 그리고 커널을 설정해야 된다.



---

## 20.6.2 간단한 시리얼 콘솔 설정

이번 섹션은 기본 설정을 사용하고 시리얼 포트에 어떻게 연결하는지 알고 있으며 단지 시리얼 콘솔의 간략한 개요를 알고 싶다고 가정한다. 이번 단계가 어렵다고 생각되면 모든 옵션과 고급 설정을 자세히 설명하는 다음 섹션을 본다.

- ① 시리얼 포트에 연결한다. 시리얼 콘솔은 COM1이 된다.
- ② 부트 로더와 커널이 시리얼 콘솔을 활성화하도록 `echo -h > /boot.config` 를 실행한다.
- ③ `/etc/ttys`에서 `ttyd0` 엔트리를 `off`에서 `on`으로 변경한다. 이것은 비디오 콘솔이 일반적으로 설정된 것처럼 시리얼 콘솔에 로그인 프롬프트를 보여준다.
- ④ `shutdown -r now` 명령을 실행하여 시리얼 콘솔로 시스템을 재 부팅한다.

## 20.6.3 시리얼 콘솔 설정

- ① 시리얼 케이블 준비  
널 모뎀 케이블이나 표준 시리얼 케이블 그리고 널 모뎀 아답터가 필요하다. 시리얼 케이블을 설명한 섹션을 참고한다.
- ② 키보드를 뺀다.  
대부분의 PC 시스템은 전원이 들어오고 자체 테스트를 하는 동안 키보드를 찾고 키보드를 감지하지 못하면 에러를 표시한다. 어떤 머신은 키보드가 없는 것을 큰 소리로 알려주고 키보드를 꽂을 때까지 부팅하지 않는다.

여러분의 컴퓨터가 에러는 표시하지만 어째든 부팅을 계속한다면 특별히 필요한 것은 없다(Phoenix BIOS 가 설치된 어떤 머신은 "keyboard failed"만 나타내고 계속 부팅한다).

**Tip:** BIOS 에서 키보드 설정을 "Not installed"로 설정하는 것은 키보드를 사용하지 못한다는 것이 아니다. 이 설정은 전원이 들어왔을 때 BIOS 에게 키보드를 탐색하지 말라는 것으로 키보드가 없더라도 에러를 표시하지 않는다. 이 플래그를 "Not installed"로 설정하였더라도 키보드가 꽂혀 있다면 키보드는 정상적으로 동작한다.

---

키보드가 없어서 컴퓨터가 부팅하지 않는다면 BIOS 를 설정해야 된다. 그래서 가능하다면 이 에러를 무시한다. 에러를 무시하는 방법은 마더보드 매뉴얼 페이지를 참조한다.

**Note:** 시스템에 PS/2 마우스가 있다면 키보드와 같이 마우스를 빼놓을 수 있다. 왜냐하면 PS/2 마우스는 키보드와 어떤 하드웨어를 공유하므로 그대로 꽂아둔 마우스 때문에 키보드가 꽂혀있다고 잘못 감지하게 된다. AMI Bios 를 가진 Gateway 2000 펜티엄 90Mhz 시스템을 이런 방법으로 사용할 수 있다. 일반적으로 키보드가 없는 마우스는 별로 쓸모가 없기 때문에 이것은 문제가 되지 않는다.

③ dumb 터미널을 COM1(sio0)에 꽂는다.

dumb 터미널이 없다면 예전 PC/XT 에 모뎀 프로그램을 사용하거나 다른 유닉스 박스의 시리얼 포트를 사용할 수 있다. COM1(sio0) 포트가 없으면 생성한다. 여기서 부트 블록을 다시 컴파일 하지 않고 부트 블록에 COM1 포트가 아닌 다른 포트를 선택할 수 없다. 다른 장치가 COM1 포트를 사용하고 있다면 이 장치를 일시적으로 삭제하고 새로운 부트 블록과 커널을 설치한 후 FreeBSD 를 시작한다(COM1 을 파일/컴퓨터/터미널 서버에 이용할 수 있다고 가정한다; COM1 을 다른 것에 사용해야 된다면(그리고 COM1 을 사용하는 것을 COM2 로(sio1) 바꾸지 못한다면) 귀찮더라도 어쩔 수 없다).

④ 커널 설정파일의 COM1(sio0)에 적절한 플래그가 필요하다.

관련 플래그는 다음과 같다:

*0x10*

이 포트의 콘솔 지원을 활성화한다. 다른 콘솔 플래그는 이 설정이 없다면 무시된다. 현재 하나의 유닛만 콘솔이 지원할 수 있다; 이 플래그가 설정된 첫 번째(설정 파일 순서에서) 유닛만 우선적으로 지원된다. 이 옵션 하나로는 시리얼 포트의 콘솔을 생성하지 못한다. 다음 플래그나 아래에 설명된 *-h* 옵션을 이 플래그와 같이 설정한다.

*0x20*

아래에서 설명하는 *-h* 옵션에 관계없이 이 포트를 콘솔(다른 우선순위 콘솔이 없다면)로 지정한다. 이 플래그는 FreeBSD 버전 2.X의 *COMCONSOLE* 옵션을 대체한다. 플래그 *0x20*은 *0x10* 플래그와 같이 사용해야 된다.

---

0x40

이 포트(0x10 과 조합하여)를 지정하여 일반적으로 접근하지 못하게 한다. 시리얼 콘솔로 사용하기를 원하는 시리얼 포트에 이 플래그를 설정하지 않는다. 이 플래그는 원격 커널 디버그를 위해 디자인하였다. 원격 디버깅에 대한 상세한 정보는 개발자 핸드북([http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/developers-handbook/index.html](http://www.freebsd.org/doc/en_US.ISO8859-1/books/developers-handbook/index.html))을 본다.

**Note:** FreeBSD 4.0 이나 이후 버전에서 플래그 0x40의 의미가 약간 달라서 원격 디버그 시리얼 포트에 다른 플래그를 명시해야 된다.

예제:

```
device sio0 at isa? port IO_COM1 flags 0x10 irq 4
```

더 자세한 사항은 sio(4) 매뉴얼 페이지를 본다.

플래그가 설정되어있지 않다면 **UserConfig**(다른 콘솔에서)를 실행하거나 커널을 다시 컴파일 해야 된다.

- ⑤ 부트 드라이브의 a 파티션에 있는 root 디렉터리에 boot.config를 생성한다.

이 파일은 시스템을 어떻게 부트해야 되는지 부트 블록 코드에게 지시한다. 시리얼 콘솔을 활성화하려면 하나 또는 하나 이상의 다음 옵션이 필요하다. 여러 개의 옵션을 지정하려면 같은 라인에 지정한다:

-h

내부와 시리얼 콘솔을 변경한다. 이 옵션을 사용하여 콘솔 장치를 변경할 수 있다. 예를 들어 내부 콘솔(비디오)로 부팅하였다면 -h를 사용하여 부트 로더와 커널이 콘솔 장치로 시리얼 포트를 사용하도록 지시할 수 있다. 그렇지 않고 시리얼 포트로 부팅하였다면 -h를 사용하여 부트 로더와 커널이 콘솔 대신 비디오 화면을 사용하도록 할 수 있다.

-D

싱글 콘솔과 듀얼 콘솔 설정을 변경한다. 싱글 설정에서 콘솔(비디오 화면)은 -h 옵션 상태에 따라 내부 콘솔이거나 시리얼 포트일 것이다. 듀얼 콘솔

---

설정은 `-h` 옵션 상태에 관계없이 비디오 화면과 시리얼 포트가 동시에 콘솔이 된다. 그러나 듀얼 콘솔을 설정하는 것은 부트 블록이 실행 중일 때만 효과가 있다. 부트 로더가 제어를 하게 되면 `-h` 옵션으로 콘솔이 유일한 콘솔로 지정된다.

`-P`

부트 블록이 키보드를 찾도록 한다. 키보드를 찾지 못하면 `-D`와 `-h` 옵션이 자동으로 설정된다.

**Note:** 현재 버전의 부트 블록 공간이 제한되어 있기 때문에 `-P` 옵션은 추가된 키보드만 감지할 수 있다. 101 키 보다 적은 키보드(F11 과 F12 키가 없는)는 감지되지 않는다. 어떤 노트북 컴퓨터의 키보드는 이러한 한계로 정확히 감지되지 않는다. 시스템이 이런 경우에 해당된다면 `-P` 옵션을 사용하는 것은 단념한다. 유감스럽게 이 문제에 대한 해결책은 없다.

콘솔을 자동으로 선택하도록 `-P` 옵션을 사용하거나 `-h` 옵션으로 시리얼 콘솔을 활성화한다.

추가적으로 `boot(8)`에 설명되어있는 다른 옵션을 사용할 수 있을 것이다.

`-P`를 제외하고 다른 옵션들은 부트 로더(`/boot/loader`)에게 전달된다. 부트 로더는 `-h` 옵션의 상태에 따라 내부 비디오와 시리얼 포트 중 어느 쪽이 콘솔이 되는지 결정한다. 이 의미는 `/boot.config`에 `-h` 옵션이 아닌 `-D` 옵션을 지정했다면 부트 블록이 동작하는 동안만 시리얼 포트를 콘솔처럼 사용할 수 있다; 부트 로더는 내부 비디오 화면을 콘솔로 사용한다.

#### ⑥ 머신 부팅

FreeBSD 박스를 시작할 때 부트 블록은 콘솔에 `/boot.config`의 내용을 보여준다. 예를 들면 다음과 같은 화면을 볼 수 있다:

```
/boot.config: -P
Keyboard: no
```

`-P`를 `/boot.config`에 넣었다면 두 번째 라인은 키보드가 있다(presence)/없다(absence)로 표시된다. 이런 메시지는 `/boot.config` 옵션에

---

따라 시리얼이나 내부 콘솔 또는 양쪽에서 나타난다.

| 옵션                   | 메시지가 나타나는 곳 |
|----------------------|-------------|
| none                 | 내부 콘솔       |
| -h                   | 시리얼 콘솔      |
| -D                   | 시리얼과 내부 콘솔  |
| -Dh                  | 시리얼과 내부 콘솔  |
| -P, keyboard present | 내부 콘솔       |
| -P, keyboard absent  | 시리얼 콘솔      |

위의 메시지 이후에 부트 블록이 계속 부트 로더를 로딩하고 콘솔에 다른 메시지가 출력되기 전에 잠시 동안 정지한다. 일반적인 상황에서 부트블록에 인터럽트를 걸 필요는 없지만 설정이 정확한지 확인하기 위해 인터럽트를 걸 수 있다.

부트 프로세스를 인터럽트 하기 위해 콘솔에서 Enter 를 제외하고 아무 키나 누른다. 부트 블록은 다른 동작을 위해 프롬프트를 보여준다. 다음과 비슷한 화면을 보게 될 것이다:

```
>> FreeBSD/i386 BOOT
Default: 0:wd(0,a)/boot/loader
boot:
```

/boot.config 에 넣은 옵션에 따라 위의 메시지가 시리얼이나 내부 콘솔 또는 양쪽에 나타나는지 확인한다. 정확한 콘솔에 메시지가 나타난다면 부트 프로세스를 계속 진행하도록 Enter 를 누른다.

시리얼 콘솔을 원했지만 시리얼 터미널에서 프롬프트를 볼 수 없다면 설정에 문제가 있을 것이다. 인터럽트를 걸었을 때 -h를 입력하고 Enter/Return(가능하다면)를 눌러 부트 블록(그리고 부트 로더와 커널)에게 시리얼 포트를 콘솔로 선택하도록 한다. 시스템이 부팅하면 무엇이 문제인지 확인한다.

부트 로더가 로드 된 후 부트 프로세스가 세 번째 단계에 이르렀더라도 부트 로더의 적절한 환경변수 설정에 의해 내부 콘솔과 시리얼 콘솔을 변경할 수 있다.

---

## 20.6.4 요약

다음은 이번 섹션에서 설명한 다양한 설정과 마지막으로 선택되는 콘솔에 대해 요약한 것이다.

### 20.6.4.1 경우 1: sio0 에 플래그 0x10 을 설정했다.

```
device sio0 at isa? port IO_COM1 flags 0x10 irq 4
```

| /boot.config 에서 옵션   | 부트 블록 콘솔 | 부트 로더 콘솔 | 커널 콘솔 |
|----------------------|----------|----------|-------|
| 없다                   | 내부       | 내부       | 내부    |
| -h                   | 시리얼      | 시리얼      | 시리얼   |
| -D                   | 시리얼과 내부  | 내부       | 내부    |
| -Dh                  | 시리얼과 내부  | 시리얼      | 시리얼   |
| -P, keyboard present | 내부       | 내부       | 내부    |
| -P, keyboard absent  | 시리얼과 내부  | 시리얼      | 시리얼   |

### 20.6.4.2 경우 2: sio0 에 플래그 0x30 을 설정했다.

```
device sio0 at isa? port IO_COM1 flags 0x30 irq 4
```

| /boot.config 에서 옵션   | 부트 블록 콘솔 | 부트 로더 콘솔 | 커널 콘솔 |
|----------------------|----------|----------|-------|
| 없다                   | 내부       | 내부       | 시리얼   |
| -h                   | 시리얼      | 시리얼      | 시리얼   |
| -D                   | 시리얼과 내부  | 내부       | 시리얼   |
| -Dh                  | 시리얼과 내부  | 시리얼      | 시리얼   |
| -P, keyboard present | 내부       | 내부       | 시리얼   |
| -P, keyboard absent  | 시리얼과 내부  | 시리얼      | 시리얼   |

---

## 20.6.5 시리얼 콘솔 팁

### 20.6.5.1 시리얼 포트 속도를 빠르게 설정

기본적으로 시리얼 포트 설정은: 9600 보드, 8bits, 패리티 없이 1 stop bit 이다. 속도를 바꾸려면 최소한 부트 블록을 다시 컴파일 해야 된다. 다음 라인을 /etc/make.conf 에 추가하고 새로운 부트 블록으로 컴파일 한다:

```
BOOT_COMCONSOLE_SPEED=19200
```

새로운 부트 블록을 빌드하고 설치하는 자세한 방법은 다음 섹션을 본다.

시리얼 콘솔이 *-h*로 부팅하지 않고 다른 설정으로 되어 있거나 커널이 사용하는 시리얼 콘솔이 부트 블록이 사용하는 것과 다르면 커널 설정파일에 다음 라인을 추가하고 새로운 커널로 컴파일 해야 된다:

```
options CONSPEED=19200
```

### 20.6.5.2 sio0 가 아닌 다른 시리얼 포트를 콘솔에 사용

sio 가 아닌 다른 포트를 콘솔에 사용하려면 몇 가지를 다시 컴파일 해야 된다. 어떤 이유에서든 다른 시리얼 포트를 사용하려면 부트 블록, 부트 로더 그리고 커널을 다음과 같이 다시 컴파일 한다.

- ① 커널 소스를 받아온다 (19장을 본다)
- ② /etc/make.conf에 사용하려는 포트 주소(0x3F8, 0x2F8, 0x3E8 or 0x2E8)를 *BOOT\_COMCONSOLE\_PORT*에 설정한다. sio0에서 sio3(COM1에서 COM4까지)까지만 사용할 수 있고 멀티 포트 시리얼 카드는 동작하지 않는다. 인터럽트가 없는 설정이 필요하다.
- ③ 사용자 커널 설정파일을 생성하고 사용하려는 시리얼 포트에 적당한 플래그를 추가한다. 예를 들어 sio1(COM2)를 콘솔로 만들려면 다음과 같이 설정한다:

---

```
device sio1 at isa? port IO_COM2 flags 0x10 irq 3
```

또는

```
device sio1 at isa? port IO_COM2 flags 0x30 irq 3
```

다른 시리얼 포트에 콘솔 플래그를 설정하지 않는다.

- ④ 다시 컴파일하고 부트 블록과 부트 로더를 설치한다:

```
# cd /sys/boot
# make clean
# make
# make install
```

- ⑤ 다시 빌드하고 커널을 설치한다.

- ⑥ disklabel(8)로 부트 디스크에 부트블록을 작성하고 새로운 커널로 부팅한다.

### 20.6.5.3 시리얼 라인으로 DDB 디버거 들어가기

시리얼 콘솔에서(원격 진단법에 유용하지만 시리얼 포트에 이상한 BREAK 신호를 보내면 위험하다) 커널 디버거로 들어가려면 다음 옵션으로 커널을 컴파일 한다:

```
options BREAK_TO_DEBUGGER
options DDB
```

### 20.6.5.4 시리얼 콘솔에서 로그인 프롬프트 확인

필요하지는 않지만 시리얼 라인을 통해서 login 프롬프트를 확인하고 싶을 것이다. 이제 시리얼 콘솔을 통해 부트 메시지와 커널 디버그 세션으로 들어갈 수 있다. 여기서 어떻게 하는지 보여준다.



---

/etc/ttys 파일을 에디터로 열고 다음 라인으로 이동한다:

```
ttyd0 "/usr/libexec/getty std.9600" unknown off secure
ttyd1 "/usr/libexec/getty std.9600" unknown off secure
ttyd2 "/usr/libexec/getty std.9600" unknown off secure
ttyd3 "/usr/libexec/getty std.9600" unknown off secure
```

*ttyd0*에서 *ttyd3*는 COM1에서 COM4와 일치한다. 원하는 포트를 *off*에서 *on*으로 바꾼다. 시리얼 포트의 속도를 바꾸었다면 *std.9600*을 현재 설정에 맞도록 *std.19200*와 같이 변경한다.

터미널 타입도 *unknown*에서 실제 시리얼 터미널 타입으로 바꿀 수 있다.

파일을 수정한 후 이 변경 사항을 적용하기 위해 `kill -HUP 1`을 입력한다.

## 20.6.6 부트 로더에서 콘솔 변경

이전 섹션에서 부트 블록을 조정하여 시리얼 콘솔을 어떻게 설정하는지 설명하였다. 이번 섹션은 부트 로더에서 몇몇 명령과 환경변수를 입력하여 콘솔을 지정하는 것을 보여준다. 부트 블록 후에 부트 프로세스의 세 번째 단계에서 부트 로더가 호출되기 때문에 부트 로더의 설정이 부트 블록의 설정을 덮어쓴다.

### 20.6.6.1 시리얼 콘솔 설정

/boot/loader.rc에 라인 하나를 추가하여 부트 로더와 커널이 시리얼 콘솔을 사용하도록 쉽게 지정할 수 있다:

```
set console=comconsole
```

이것은 이전 섹션에서 설명한 부트 블록의 설정에 상관없이 영향을 끼친다.

/boot/loader.rc의 첫 번째 라인에 위의 라인을 입력하는 것이 좋다. 그래서 가능한 쉽게 시리얼 콘솔에서 부트 메시지를 볼 수 있다.

---

마찬가지로 다음과 같이 내부 콘솔을 지정할 수 있다:

```
set console=vidconsole
```

부트 로더 환경변수 `console` 을 설정하지 않았다면 부트 로더와 그 후의 커널은 부트 블록에서 `-h` 옵션에 따라 어느 쪽 콘솔이든지 보여준다.

3.2 이후 버전에서 `/boot/loader.rc` 보다 `/boot/loader.conf.local` 이나 `/boot/loader.conf` 에 콘솔을 지정해야 된다. 이러한 방법에서 `/boot/loader.rc` 는 다음과 비슷할 것이다:

```
include /boot/loader.4th
start
```

그리고 `/boot/loader.conf.local` 를 생성하고 다음 라인을 추가한다.

```
console=comconsole

또는

console=vidconsole
```

더 많은 정보는 `loader.conf(5)`를 본다.

**Note:** 최근의 부트 로더는 부트 블록에 옵션이 없으면 `-P` 옵션과 동일하다. 그리고 키보드가 있느냐에 따라 자동으로 내부 콘솔과 시리얼 콘솔을 선택하는 설정은 없다.

### 20.6.6.2 콘솔용 시리얼 포트로 `sio0` 가 아닌 다른 포트 사용

시리얼 콘솔을 `sio0` 가 아닌 다른 시리얼 포트로 사용하려면 부트 로더를 다시 컴파일 해야 된다. 섹션 “`sio0` 가 아닌 다른 시리얼 포트를 콘솔에 사용”에서 설명하는 절차를 따른다.

---

## 20.6.7 주의

여기서 설명하는 아이디어는 그래픽이나 키보드가 필요 없는 전용서버를 설정할 수 있도록 하고 있다. 대부분의 시스템은 키보드 없이 부팅할 수 있지만 아주 소수의 시스템만 그래픽 카드 없이 부팅할 수 있다. AMI BIOS 가 있는 머신은 간단히 "graphics adapter"를 CMOS 에서 "Not installed"로 변경하여 간단히 그래픽 카드 없이 부팅할 수 있도록 설정할 수 있다.

그러나 대부분의 머신은 이 옵션을 지원하지 못하고 시스템에 그래픽 카드가 없다면 부팅을 거부한다. 이러한 머신에서는 모니터를 붙이지 않더라도 그래픽 카드(사용하지 않는 모노 그래픽 카드라도)를 끼워 두어야 된다. AMI BIOS 를 설치할 수도 있을 것이다.

---

## 21 장 PPP 와 SLIP

### 21.1 개요

FreeBSD 는 컴퓨터를 다른 컴퓨터와 연결하는 다양한 방법을 가지고 있다. 다이얼-업(dial-up) 모뎀으로 네트워크나 인터넷에 연결하려면 PPP 나 SLIP 를 사용해야 된다. 이번 장에서는 이런 모뎀 기반 통신 서비스 설정에 대해 자세히 다룬다.

이번 장을 읽고 다음과 같은 사항을 알 수 있다:

- 유저 PPP는 어떻게 설정하는가
- 커널 PPP는 어떻게 설정하는가
- PPPoE(PPP over Ethernet)는 어떻게 설정하는가
- PPPoA (PPP over ATM)는 어떻게 설정하는가
- SLIP 클라이언트와 서버는 어떻게 설정하는가

이번 장을 읽기 전에 아래 사항을 알고 있어야 한다:

- 기본적인 네트워크 용어를 이해하고 있어야 된다.
- 기본적인 다이얼-업 연결과 PPP나 SLIP에 대해 이해하고 있어야 된다.

유저 PPP 와 커널 PPP 의 가장 큰 차이점이 무엇인지 궁금할 것이다. 유저 PPP 프로세스는 커널 보다 userland 에서 데이터를 인바운드(inbound)하고 아웃바운드(outbound) 한다. 커널과 userland 사이의 데이터를 복사하는 것은 노력이 따르지만 더 많은 기능을 수행할 수 있다 - 다양한 PPP 실행. 유저 PPP 는 외부와 통신할 때 tun 장치를 사용하지만 커널 PPP 는 ppp 장치를 사용한다.

**Note:** 이번 장 전반에 걸쳐 `pppd` 처럼 다른 PPP 소프트웨어와 구분할 필요가 없다면 유저 PPP 는 단순히 `ppp` 로 표시한다. 그리고 따로 표시하지 않는다면 이 섹션에서 설명하는 모든 명령은 root 로 실행한다.

### 21.2 유저 PPP 사용

---

## 21.2.1 유저 PPP

### 21.2.1.1 전제 조건

이문서는 다음과 같은 사항으로 가정한다:

- PPP를 사용하여 연결하는 ISP 계정을 가지고 있다.
- 모뎀이나 다른 장치가 시스템에 설치되어 있고 ISP와 연결할 수 있도록 정확하게 설정되어 있다.
- ISP의 전화번호를 알고 있다.
- 로그인 이름과 패스워드(보통 유닉스 스타일 로그인과 패스워드 또는 PAP나 CHAP 로그인과 패스워드).
- 한 개 또는 한 개 이상의 네임 서버 IP주소. 보통 네임 서버로 사용하도록 ISP에서 두 개의 IP를 받았을 것이다. ISP에서 IP를 주지 않았다면 `ppp.conf`에서 `enable dns` 명령을 사용하여 PPP가 네임 서버를 설정하도록 한다. 이 기능은 ISP의 PPP가 DNS 감지(negotiation)를 지원하는가에 따른다.

다음 정보는 ISP 에서 제공 받겠지만 모두 필요한 것은 아니다:

- ISP의 게이트웨이 IP주소. 게이트웨이는 여러분이 연결하는 *default route*로 설정되는 머신이다. 이 정보가 없다면 연결됐을 때 ISP PPP 서버가 정확한 값을 제공하도록 만들 수 있다.

이 IP 번호는 PPP 에 의해 *HISADDR*로 참조된다.

- 사용해야 될 넷 마스크. ISP에서 제공하지 않는다면 무난하게 255.255.255.255를 사용할 수 있다.
- ISP에서 고정 IP와 호스트 이름을 제공했다면 입력한다. 그렇지 않으면 적당해 보이는 값을 설정한다.

필요한 정보를 모른다면 없다면 ISP 에 문의한다.

**Note:** 이 섹션 전반에 걸쳐 보여 주는 많은 예제의 설정파일 내용은 라인에 번호를 부여했다. 이들 번호는 설명을 돕기 위해 부여한 것이므로 실제 파일을 의미하지 않는다. 또한 탭과 공백 문자로 들여 쓰는 것이 중요하다.

---

### 21.2.1.2 PPP 장치 노드 생성

일반적인 상황에서 대부분의 유저는 오직 하나의 tun 장치(/dev/tun0)만 필요하다. tun0 는 아래에서 tunN으로 바뀔 것이다. N은 여러분의 시스템에 맞는 유닛 번호다.

devfs(5)가 활성화되지 않은 FreeBSD(FreeBSD 4.X 와 이전 버전)는 tun0 장치가 있는지 확인해야 된다(devfs(5)가 활성화되었다면 요청에 의해 장치 노드가 생성되기 때문에 확인할 필요는 없다).

tun0 장치가 정확하게 설정되었는지 확인하는 가장 쉬운 방법은 장치를 다시 만드는 것이다. 장치를 다시 만들기 위해 다음 명령을 따른다:

```
# cd /dev
# sh MAKEDEV tun0
```

커널에 16 개의 터널 장치가 필요하면 다음 명령을 사용하여 생성한다:

```
# cd /dev
# sh MAKEDEV tun15
```

### 21.2.1.3 자동 PPP 설정

ppp 와 pppd(커널 레벨의 PPP 기능)는 /etc/ppp 디렉터리에 있는 설정파일을 사용한다. 유저 ppp 예제는 /usr/share/examples/ppp/에서 찾을 수 있다.

ppp 를 설정할 때 수정해야 될 파일들은 요구 사항에 따라 다르다. 설정할 것은 ISP 가 고정 IP 를(다시 말해 IP 주소 하나를 주어서 항상 이 IP 만 사용한다) 할당하는지 유동 IP 를(다시 말해 ISP 에 연결할 때마다 IP 주소가 바뀐다) 할당하는지에 따라 다르다.

[PPP 설정]

## 1. PPP 와 고정 IP 주소

/etc/ppp/ppp.conf 파일을 수정해야 된다. 이것은 아래 예제와 비슷하다.

**Note:** :로 끝나는 시작 라인을 제외하고 다른 라인들은 아래에서 보여 준 대로 스페이스 또는 탭 키를 사용하여 들여쓰기 해야 된다.

```
1  default:
2      set log Phase Chat LCP IPCP CCP tun command
3      ident user-ppp VERSION (built COMPILATIONDATE)
4      set device /dev/cuaa0
5      set speed 115200
6      set dial "ABORT BUSY ABORT NOWWsCARRIER TIMEOUT 5 W
7              W"W" AT OK-AT-OK ATE1Q0 OK WWdATDTWWT TIMEOUT 40
CONNECT"
8      set timeout 180
9      enable dns
10
11  provider:
12      set phone "(123) 456 7890"
13      set authname foo
14      set authkey bar
15      set login "TIMEOUT 10 W"W" W"W" gin:--gin: WWU word: WWP col: ppp"
16      set timeout 300
17      set ifaddr x.x.x.x y.y.y.y 255.255.255.255 0.0.0.0
18      add default HISADDR
```

### 라인 1:

default 엔트리 확인. 이 엔트리의 명령은 ppp 가 동작할 때 자동으로 실행된다.

### 라인 2:

로그인 매개변수 활성화. 설정이 정확하게 동작하면 너무 큰 로그 파일을 만들지 않도록 이 라인이 메시지를 요약한다.

```
set log phase tun
```

---

**라인 3:**

PPP 가 peer 에게 어떻게 인증을 받을 수 있는지 지시한다. 교신과 링크 설정에 문제가 생기면 PPP 는 직접 peer 에게 인증을 받고 peer 관리자가 이런 문제를 조사할 때 유용한 정보를 제공한다.

**라인 4:**

모뎀이 접속하고 있는 장치를 확인한다. COM1 은 /dev/cuaa0 이고 COM2 는 /dev/cuaa1 이다.

**라인 5:**

연결하려는 속도를 설정한다. 115200 이 동작하지 않으면(새로운 모뎀에서 논리적인 이유로) 38400 을 사용해 본다.

**라인 6 & 7:**

다이얼 문자열. 유저 PPP 는 chat(8) 프로그램과 비슷한 expect/send 구문을 사용한다. 이 언어의 특성에 대한 더 많은 정보는 매뉴얼 페이지를 참고한다.

이 명령은 읽기 쉽게 다음 라인에서 계속된다. ppp.conf 의 명령은 라인의 마지막 문자가 "W" 문자이면 다음 라인으로 계속된다.

**라인 8:**

링크의 idle 타임아웃을 설정한다. 180 초가 기본값이기 때문에 이 라인은 상당히 직선적이다.

**라인 9:**

로컬 리졸버 설정을 peer 가 확인하도록 PPP 가 요청하도록 지시한다. 로컬 네임서버를 운용 중이라면 이 라인을 주석처리 하거나 삭제한다.

**라인 10:**

읽기 쉽도록 빈 라인이다. 빈 라인은 PPP 가 무시한다.

**라인 11:**

"provider"라고 부르는 라인은 provider 엔트리를 확인한다. 이곳을 ISP 이름으로 변경할 수 있기 때문에 연결을 시작하기 위해 나중에 *load ISP*를 사용할 수 있다.

**라인 12:**



이 공급자의 전화번호를 설정한다. 여러 개의 전화번호는 콜론(:)이나 파이프 문자(|)로 나눠서 지정한다. 이 분리 방법의 차이는 ppp(8)에 나와있다. 요약해보면 번호를 교대로 사용하려면 콜론(:)을 사용한다. 첫 번째 번호를 처음으로 사용하고 첫 번째 번호가 실패해서 다른 번호를 사용하려면 파이프 문자를 사용한다. 위에서 본 것처럼 전체 전화번호를 설정하려면 인용부호를 사용한다. 전화번호에서 공백으로 들여쓰기를 했다면 인용부호(")로 둘러싸야 한다.

**라인 13 & 14:**

유저 이름과 패스워드 확인. 유닉스 스타일 로그인 프롬프트를 사용하여 연결할 때 이런 값은 set login 명령에 WU 와 WP 변수로 참조된다. PAP 나 CHAP 로 연결하여 인증할 때 이런 값이 사용된다.

**라인 15:**

PAP 나 CHAP 를 사용한다면 여기서 로그인할 수 없기 때문에 이 라인을 주석처리 하거나 삭제한다. 더 자세한 사항은 PAP 와 CHAP 인증을 참고한다.

로그인 문자열은 다이얼 문자열과 같은 chat 와 비슷한 구문이다. 이 예제에서 이 문자열은 다음과 같은 로그인 세션을 가진 유저를 위해 동작한다:

```
J. Random Provider
login: foo
password: bar
protocol: ppp
```

필요에 따라 이 스크립트를 수정한다. 처음으로 이 스크립트를 작성할 때 "chat" 로그를 활성화해서 대화가 제대로 되는지 확인할 수 있다.

**라인 16:**

연결의 기본 idle 타임아웃(초단위로)을 설정한다. 여기서 연결은 동작이 없이 300 초가 지나면 자동으로 끊어진다. 타임아웃을 원치 않으면 이 값을 0 으로 설정하거나 `-ddial` 명령 라인으로 변경한다.

**라인 17:**

인터페이스 주소 설정. 문자열 `x.x.x.x`는 ISP 에서 공급하는 IP 주소로 변경한다. 문자열 `y.y.y.y`은 ISP 의 게이트웨이(여러분이 연결하는 머신) IP 주소로 변경한다. ISP 에서 게이트웨이 주소를 제공하지 않았다면 `10.0.0.2/0` 을 사용한다. "추측해서"

주소를 사용해야 된다면 유동 IP 주소를 사용한 PPP 에서 지시한대로 /etc/ppp/ppp.linkup 에 엔트리를 생성한다. 이 라인이 생략됐다면 PPP 는 *-auto* 모드로 동작하지 않는다.

#### 라인 18:

기본 경로로 ISP 의 게이트웨이를 추가한다. 특수한 단어 *HISADDR*은 라인 9 에서 지정한 게이트웨이 주소로 대체된다. 9 라인 이후에 이 라인이 없으면 *HISADDR*은 초기화되지 않기 때문에 중요하다.

ppp 를 *-auto* 로 사용하지 않으려면 이 라인을 ppp.linkup 파일로 이동시킨다.

고정 IP 로 ppp 가 *-auto* 모드에서 동작하면 연결하기 전에 라우팅 테이블 엔트리가 정확하게 설정되기 때문에 ppp.linkup 에 엔트리를 추가할 필요가 없다. 그러나 연결 후에 프로그램을 실행하기 위해 엔트리를 생성할 수 있다. 이것은 나중에 샌드메일 예제에서 설명한다.

예제 설정파일은 /usr/share/examples/ppp/ 디렉터리에서 찾을 수 있다.

## 2. PPP 와 유동 IP 주소

인터넷 서비스 공급자가 고정 IP 주소를 할당하지 않는다면 로컬과 원격 주소 감지(negotiate)에 ppp 를 설정할 수 있다. "추측한" IP 주소를 설정하여 연결 후에 ppp 가 IP 설정 프로토콜(IPCP)을 사용하여 정확하게 설정하도록 한다. ppp.conf 설정은 고정 IP 를 사용한 PPP 와 같으므로 다음과 같이 변경한다:

```
17      set ifaddr 10.0.0.1/0 10.0.0.2/0 255.255.255.255
```

레퍼런스이기 때문에 역시 라인 번호를 포함하지 않는다. 최소한 한 칸 이상의 공백으로 들여 쓰기 해야 된다.

#### 라인 17:

/ 문자 이후의 숫자는 ppp 가 사용하려는 주소비트 번호다. 상황에 맞게 정확한 IP 주소를 사용하려고 하겠지만 위의 예제는 항상 동작한다.

마지막 인자(0.0.0.0)는 PPP 에게 10.0.0.1 이 아닌 0.0.0.0 을 사용하여 교신을 시작하도록 하고 어떤 ISP 들에게 필요하다. PPP 가 *-auto* 모드에서 초기 경로를

설정하지 못하므로 `set ifaddr` 명령의 첫 번째 인자로 `0.0.0.0`을 지정하지 않는다.

`-auto` 모드에서 사용하지 않는다면 연결이 된 후에 사용되는 `/etc/ppp/ppp.linkup`에 엔트리를 생성해야 된다. 여기서 `ppp`가 인터페이스 주소에 할당되면 이제 라우팅 테이블 엔트리를 추가하는 것도 가능하다:

```
1 provider:
2 add default HISADDR
```

#### 라인 1:

연결을 설정할 때 `ppp`는 다음과 같은 규정에 따라 `ppp.linkup`에서 엔트리를 찾는다: 첫째 `ppp.conf`에 사용한 라벨과 같은 대응되는 라벨이 있는지 찾는다. 검색에 실패하면 게이트웨이 IP 주소 엔트리를 찾는다. 이 엔트리는 4개의 8진수 IP 스타일 라벨이다. 그래도 엔트리를 찾지 못하면 `MYADDR` 엔트리를 찾는다.

#### 라인 2:

이 라인은 `HISADDR`가 지정하는 경로를 기본 경로로 추가하도록 `ppp`에게 지시한다. `HISADDR`은 IPCP에서 결정된 게이트웨이 IP 주소로 대체된다.

더 자세한 예제는 파일 `/usr/share/examples/ppp/ppp.conf.sample`과 `/usr/share/examples/ppp/ppp.linkup.sample`에서 `pmdemand` 엔트리를 본다.

### 3. 입력 콜 받기

LAN에 연결된 머신에서 입력 콜을 받도록 `ppp`를 설정할 때 패킷을 LAN으로 포워드 할 것인지 결정해야 된다. 포워드 하겠다면 LAN 서브넷의 IP 주소를 `peer`에게 할당하고 `/etc/ppp/ppp.conf` 파일에서 `enable proxy` 명령을 사용한다. 또한 `/etc/rc.conf` 파일에 다음 내용이 있는지 확인한다:

```
gateway_enable="YES"
```

### 4. 어떤 `getty` 인가?

FreeBSD를 다이얼-업 서비스로 설정하기 섹션에서 `getty(8)`를 사용한 다이얼-업 서비스 활성화를 설명한다.

---

또 다른 `getty`는 다이얼-업을 위해 개선된 버전인 `mgetty`(<http://www.leo.org/~doering/mgetty/index.html>)다.

`mgetty`를 사용하는 장점은 `mgetty`가 모뎀과 활발히 통신한다. 이 의미는 `/etc/ttys`에서 포트가 꺼져있다면 모뎀은 응답하지 않는다.

최근 `mgetty` 버전(0.99 베타 버전부터)은 클라이언트 스크립트가 서버에 접근하는 횟수를 줄일 수 있도록 `ppp` 흐름을 자동으로 감지한다.

`mgetty`에 대한 더 많은 정보는 `Mgetty`와 `Autoppp`를 참고한다.

## 5. ppp 퍼미션

`ppp` 명령은 보통 `root` 유저에서 실행한다. 그러나 아래에서 설명하듯이 `ppp`를 서버 모드에서 일반 유저로 실행하려면 `/etc/group`의 `network` 그룹에 유저를 추가하여 `ppp`를 실행할 수 있는 권한을 할당한다.

그리고 `allow` 명령으로 유저들이 설정파일에서 하나 이상의 섹션에 접근할 수 있도록 해야 된다:

```
allow users fred mary
```

이 명령이 `default` 섹션에서 사용된다면 지정된 유저가 모든 것에 접근할 수 있다.

## 6. 유동 IP 유저를 위한 PPP 셸

다음 내용으로 `/etc/ppp/ppp-shell` 파일을 만든다:

```
#!/bin/sh
IDENT=`echo $0 | sed -e 's/^.*-W(.*)$/W1/'`
CALLEDAS="$IDENT"
TTY=`tty`

if [ x$IDENT = xdialup ]; then
    IDENT=`basename $TTY`
fi

echo "PPP for $CALLEDAS on $TTY"
echo "Starting PPP for $IDENT"

exec /usr/sbin/ppp -direct $IDENT
```

이 스크립트는 실행할 수 있어야 되고 다음 명령으로 ppp-dialup 이라는 심볼릭 링크를 만든다:

```
# ln -s ppp-shell /etc/ppp/ppp-dialup
```

그리고 모든 다이얼-업 유저 *찰로* 이 스크립트를 이용한다. 다음은 /etc/passwd 에서 pchilds 라는(직접 패스워드 파일을 수정하지 않고 vipw 를 사용한다) 유저의 다이얼-업 PPP 예제다.

```
pchilds:*:1011:300:Peter Childs PPP:/home/ppp:/etc/ppp/ppp-dialup
```

다음과 같이 누구나 읽을 수 있는 0 바이트 파일을 가진 /home/ppp 디렉터리를 생성한다:

```
-r--r--r--  1 root    wheel      0 May 27 02:23 .hushlogin
-r--r--r--  1 root    wheel      0 May 27 02:22 .rhosts
```

## 7. 고정 IP 유저를 위한 PPP 셸

위에서처럼 ppp-shell 파일을 생성하고 고정 IP 를 할당 받은 각 계정의 ppp-shell 로 심볼릭 링크를 생성한다.

---

예를 들어 fred, sam 과 mary 3 명의 다이얼-업 고객이 있고 C 클래스 네트워크를 할당하려면 다음과 같이 입력한다:

```
# ln -s /etc/ppp/ppp-shell /etc/ppp/ppp-fred
# ln -s /etc/ppp/ppp-shell /etc/ppp/ppp-sam
# ln -s /etc/ppp/ppp-shell /etc/ppp/ppp-mary
```

이러한 유저들의 다이얼-업 계정 쉘을 위에서 생성한 심볼릭 링크로 설정한다(예를 들면 mary 의 쉘은 /etc/ppp/ppp-mary 다).

## 8. 유동 IP 유저를 위한 ppp.conf 설정

/etc/ppp/ppp.conf 파일에 다음과 같은 라인이 필요하다:

```
default:
    set debug phase lcp chat
    set timeout 0

ttyd0:
    set ifaddr 203.14.100.1 203.14.100.20 255.255.255.255
    enable proxy

ttyd1:
    set ifaddr 203.14.100.1 203.14.100.21 255.255.255.255
    enable proxy
```

**Note:** 들여쓰기가 중요하다.

*default:* 섹션이 각 섹션에 로드 된다. /etc/ttys 에서 활성화시킨 각 다이얼-업 연결에 위의 *ttyd0:*와 비슷한 엔트리를 생성해야 된다. 각 라인은 유동 IP 유저를 위한 IP 주소 풀(pool)에서 유일한 IP 주소를 받아야 된다.

## 9. 고정 IP 유저를 위한 ppp.conf 설정

위의 샘플 /usr/share/examples/ppp/ppp.conf 의 내용에 따라 각각의 고정 IP 가 할당된 다이얼-업 유저에게 섹션을 추가해야 된다. fred, sam 과 mary 예제로 계속 설명한다.

```
fred:
    set ifaddr 203.14.100.1 203.14.101.1 255.255.255.255

sam:
    set ifaddr 203.14.100.1 203.14.102.1 255.255.255.255

mary:
    set ifaddr 203.14.100.1 203.14.103.1 255.255.255.255
```

필요하다면 /etc/ppp/ppp.linkup 파일에 각 고정 IP 유저에 대한 라우팅 정보도 있어야 된다. 아래 라인은 클라이언트의 ppp 링크를 경유하는 C 클래스 203.14.101.0 네트워크의 경로를 추가한다.

```
fred:
    add 203.14.101.0 netmask 255.255.255.0 HISADDR

sam:
    add 203.14.102.0 netmask 255.255.255.0 HISADDR

mary:
    add 203.14.103.0 netmask 255.255.255.0 HISADDR
```

## 10. mgetty 와 AutoPPP

*AUTO\_PPP* 옵션으로 mgetty 를 설정하고 컴파일하면 PPP 연결의 LCP(종료 패킷을 교환하여 링크를 끊는데 사용된다) 상태를 mgetty 가 감지하고 자동으로 ppp 쉘을 생성한다. 그러나 기본 로그인/패스워드 절차가 없다면 유저는 PAP 나 CHAP 인증을 사용해야 된다.

이 섹션은 유저가 *AUTO\_PPP* 옵션(v0.99 배타나 이후 버전)으로 mgetty 를 성공적으로 설정, 컴파일하고 설치 했다고 가정한다.

다음 내용이 /usr/local/etc/mgetty+sendfax/login.conf 파일에 필요하다:

```
/AutoPPP/ - - /etc/ppp/ppp-pap-dialup
```

이 설정은 감지된 PPP 연결에 mgetty 가 ppp-pap-dialup 스크립트를 실행하도록 한다.

다음 내용이 있는 /etc/ppp/ppp-pap-dialup 파일(이 파일은 실행할 수 있어야 된다)을 생성한다:

```
#!/bin/sh
exec /usr/sbin/ppp -direct pap$IDENT
```

/etc/ttys 에서 활성화된 각 다이얼-업을 위해 /etc/ppp/ppp.conf 에 일치하는 엔트리를 생성한다. 이것은 다행히 위에서 정의한 것과 상호 존재한다.

```
pap:
  enable pap
  set ifaddr 203.14.100.1 203.14.100.20-203.14.100.40
  enable proxy
```

이 방법으로 로그인한 각 유저는 /etc/ppp/ppp.secret 파일에 유저이름/패스워드가 필요하거나 /etc/password 파일에서 PAP 를 통해 유저가 인증을 받도록 다음 옵션을 추가해야 된다.

```
enable passwdauth
```

특정 유저에게 고정 IP 를 할당하려면 /etc/ppp/ppp.secret 의 세 번째 인자에 번호를 지정할 수 있다. 예제는 /usr/share/examples/ppp/ppp.secret.sample 를 본다.

## 11. MS 확장

요청에 따라 DNS 와 NetBIOS 네임 서버 주소를 할당하도록 PPP 를 설정할 수 있다.

PPP 버전 1.x 에서 이렇게 확장하려면 다음 라인을 /etc/ppp/ppp.conf 의 관련 섹션에 추가해야 된다.



```
enable msextd
set ns 203.14.100.1 203.14.100.2
set nbns 203.14.100.5
```

그리고 PPP 버전 2 이상에서는 다음 내용을 추가한다:

```
accept dnst
set dns 203.14.100.1 203.14.100.2
set nbns 203.14.100.5
```

이 설정은 클라이언트에게 첫 번째와 두 번째 네임서버 주소 그리고 NetBIOS 네임서버 호스트를 알려준다.

버전 2 와 그 이상에서 *set dns* 라인이 생략됐다면 PPP 는 /etc/resolv.conf 에 있는 네임서버 주소를 사용한다.

## 12. PAP 와 CHAP 인증

어떤 ISP 는 인증 시스템을 설치해서 여러분의 연결 인증에 PAP 나 CHAP 인증 메커니즘을 사용한다. 이 경우 연결이 되면 ISP 는 login: 프롬프트를 보여주지 않지만 즉시 PPP 통신이 시작된다.

PAP 는 패스워드를 평범한 텍스트로 보내서 CHAP 보다 안전성이 떨어지지만 시리얼 라인을 통해서만 전송하기 때문에 패스워드는 문제가 되지 않는다.

다시 PPP 와 고정 IP 주소 또는 PPP 와 유동 IP 주소 섹션을 참고해서 다음 내용을 수정한다:

```
13      set authname MyUserName
14      set authkey MyPassword
15      set login
```

라인 13:

이 라인에 여러분의 PAP/CHAP 유저 이름을 지정한다. *MyUserName* 에 정확한 값을 입력해야 된다.

---

#### 라인 14:

이 라인은 여러분의 PAP/CHAP 패스워드를 지정한다. *MyPassWord*에 정확한 값을 입력한다. 정확히 설정하려고 다음과 같은 라인을 추가하려고 하겠지만 기본적으로 PAP 와 CHAP 양쪽에서 허용된다.

```
16      accept PAP

또는

16      accept CHAP
```

#### 라인 15:

여러분이 PAP 나 CHAP 를 사용한다면 ISP 서버에 보통 로그인하지 않아도 된다. 따라서 "set login"문자열을 비활성 해야 된다.

### 13. 동작중인 PPP 설정 변경하기

ppp 프로그램이 백그라운드에서 실행되는 동안 ppp 프로그램과 통신하는 것이 가능하지만 적절히 통신할 수 있는 포트가 설정되어 있을 때만 가능하다. 통신할 수 있는 포트를 설정하려면 설정에 다음 라인을 추가한다:

```
set server /var/run/ppp-tun%d DiagnosticPassword 0177
```

이것은 PPP 에게 특정 유닉스 도메인 소켓을 주시하고 클라이언트의 접근을 허용하기 전에 명시된 패스워드를 물어보도록 한다. 이름에서 %d는 사용하는 tun 장치 번호로 대체한다.

소켓이 설정되면 스크립트의 pppctl(8) 프로그램을 사용하여 실행중인 프로그램을 조정할 수 있을 것이다.

### 21.2.1.4 PPP 네트워크 주소 변환 기능 사용

PPP 는 커널을 변경할 필요 없이 내부 NAT 를 사용할 수 있는 기능이 있다. 이런 기능은

---

/etc/ppp/ppp.conf 에 다음 라인을 추가하면 활성화될 것이다:

```
nat enable yes
```

다른 방법으로 PPP NAT 는 명령어 라인 옵션 *-nat*으로 활성화할 수 있다. 또한 기본적으로 활성화되도록 /etc/rc.conf 에 *ppp\_nat*를 추가할 수 있다.

이 기능을 사용한다면 다음과 같이 입력되는 연결을 포워딩 하거나:

```
nat port tcp 10.0.0.2:ftp ftp
nat port tcp 10.0.0.2:http http
```

외부의 모든 입력을 막는 /etc/ppp/ppp.conf 에 유용한 옵션을 찾을 수 있을 것이다.

```
nat deny_incoming yes
```

### 21.2.1.5 마지막 시스템 설정

ppp 를 설정하였지만 실행하기 전에 해야 되는 몇 가지가 있다. /etc/rc.conf 파일 수정과 관련된 일이다.

이 파일의 상단에 *hostname=* 라인을 설정해야 된다 예:

```
hostname="foo.example.com"
```

여러분의 ISP 가 고정 IP 주소와 이름을 부여했다면 호스트 이름으로 이 이름이 적절하다.

*network\_interfaces* 변수를 본다. 요청에 따라 시스템이 ISP 에 연결하도록 설정하려면 tun0 장치가 리스트에 추가되어 있는지 확인하고 그렇지 않으면 삭제한다.

```
network_interfaces="lo0 tun0"
ifconfig_tun0=
```

---

**Note:** *ifconfig\_tun0* 변수는 비어 있어야 하고 */etc/start\_if.tun0* 라는 파일을 생성해야 된다. 이 파일에는 다음 라인이 필요하다:

```
ppp -auto mysystem
```

이 스크립트는 네트워크를 설정할 때 실행되고 자동 모드에서 ppp 데몬이 시작한다. 이 머신이 LAN의 게이트웨이라면 *-alias* 스위치를 사용할 수 있다. 더 자세한 사항은 매뉴얼 페이지를 참고한다.

*/etc/rc.conf*의 다음 라인에서 라우터 프로그램을 *NO*로 설정한다:

```
router_enable="NO"
```

routed가 ppp로부터 생성된 기본 라우팅 테이블 엔트리를 삭제하기 때문에 routed 데몬을 시작하면 안 된다.

*sendmail\_flags* 라인에 *-q* 옵션을 포함하지 않는 것도 중요할 수 있다. 그렇지 않으면 sendmail이 주소를 확인하기 위해 머신이 아이얼-아웃하는 원인이 될 것이다. 다음과 같이 설정할 수 있다:

```
sendmail_flags="-bd"
```

이 설정의 단점은 ppp가 연결될 때마다 다음 명령으로 sendmail이 메일 큐를 다시 확인하도록 해야 된다:

```
# /usr/sbin/sendmail -q
```

자동으로 메일 큐를 확인하도록 *ppp.linkup*에 *!bg* 명령을 사용할 수 있다:

```
1 provider:
2 delete ALL
3 add 0 0 HISADDR
4 !bg sendmail -bd -q30m
```

---

이렇게 하지 않고 SMTP 트래픽을 막도록 "dfilter"를 설정하는 것도 가능하다. 더 자세한 사항은 샘플 파일을 참고한다.

이제 유일하게 남은 것은 머신을 재 부팅하는 일이다.

재 부팅 후 다음 명령을 입력하여 dial provider 로 PPP 세션을 시작하거나:

```
# ppp
```

외부와의 통신이 발생할 때(그리고 start\_if.tun0 스크립트를 생성하지 않았다면) ppp 세션이 자동으로 연결되기를 원한다면 다음 명령을 입력한다:

```
# ppp -auto provider
```

### 21.2.1.6 요약

ppp 를 처음 설정할 때 다음 단계를 따라야 한다:

클라이언트 쪽:

- ① tun 장치를 커널에 빌드한다.
- ② /dev 디렉터리에 tunN 장치 파일이 있어야 된다.
- ③ /etc/ppp/ppp.conf에 엔트리를 생성한다. pmdemand 예제는 대부분의 ISP 설정에 적당하다.
- ④ 유동 IP를 가지고 있다면 /etc/ppp/ppp.linkup에 엔트리를 생성한다.
- ⑤ /etc/rc.conf 파일을 업데이트한다.
- ⑥ 전화접속 요청이 필요하면 start\_if.tun0 스크립트를 생성한다.

서버 쪽:

- 
- ① tun 장치를 커널에 빌드 한다.
  - ② tunN 장치 파일을 /dev 디렉터리에서 사용할 수 있어야 한다.
  - ③ /etc/passwd(vipw(8) 프로그램을 이용하여)에 엔트리를 생성한다.
  - ④ ppp -direct direct-server나 비슷한 명령을 실행하는 프로필을 이 유저의 홈 디렉터리에 생성한다.
  - ⑤ /etc/ppp/ppp.conf에 엔트리를 생성한다. direct-server 예제가 적당할 것이다.
  - ⑥ /etc/ppp/ppp.linkup에 엔트리를 생성한다.
  - ⑦ /etc/rc.conf 파일을 업데이트 한다.

## 21.3 커널 PPP 사용

### 21.3.1 커널 PPP 설정

머신에 PPP 를 설정하기 전에 pppd 가 /usr/sbin 에 있고 /etc/ppp 디렉터리가 있어야 된다.

pppd 는 두 가지 모드로 동작할 수 있다:

- ① "클라이언트" - PPP 시리얼 회선이나 모뎀 라인으로 머신을 외부와 연결하기를 원할 것이다.
- ② "서버" - 머신이 네트워크에 연결되어 있고 ppp를 사용하여 다른 컴퓨터와의 연결에 사용된다.

두 가지 경우에 옵션 파일을 설정해야 된다(etc/ppp/options 이나 머신에서 한 명 이상의 유저가 PPP 를 사용한다면 ~/.ppprc 를 설정한다).

그리고 원격 호스트에 전화 연결하여 통신할 수 있는 모뎀/시리얼

---

소프트웨어(comms/kermit 이 좀더 적합하다)가 필요하다.

## 21.3.2 pppd 클라이언트로 사용

다음 /etc/ppp/options 은 시스코 PPP 라인 터미널 서버에 연결할 때 사용할 수 있을 것이다.

```
crtscts      # enable hardware flow control
modem        # modem control line
noipdefault  # remote PPP server must supply your IP address.
              # if the remote host does not send your IP during IPCP
              # negotiation, remove this option
passive      # wait for LCP packets
domain ppp.foo.com  # put your domain name here

:<remote_ip>  # put the IP of remote PPP host here
              # it will be used to route packets via PPP link
              # if you didn't specified the noipdefault option
              # change this line to <local_ip>:<remote_ip>

defaultroute # put this if you want that PPP server will be your
              # default router
```

접속은 다음 단계를 따른다:

- ① **kermit**으로(또는 다른 모뎀 프로그램) 원격 호스트에 전화 연결해서 유저 이름과 패스워드를(또는 원격호스트에 PPP 연결에 필요한) 입력한다.
- ② **kermit**을 종료한다(접속 상태를 유지하고).
- ③ 다음 명령을 실행한다:

```
# /usr/src/usr.sbin/pppd.new/pppd /dev/tty01 19200
```

---

적당한 속도와 장치 이름을 사용한다.

이제 컴퓨터가 PPP 로 접속되었다. 접속이 실패했다면 /etc/ppp/options 파일에 *debug* 옵션을 추가할 수 있고 문제를 추적하는 메시지를 콘솔에서 체크할 수 있다.

다음의 /etc/ppp/pppup 스크립트는 3 단계 모두 자동으로 수행한다:

```
#!/bin/sh
ps ax |grep pppd |grep -v grep
pid=`ps ax |grep pppd |grep -v grep|awk '{print $1;}'`
if [ "X${pid}" != "X" ] ; then
    echo 'killing pppd, PID=' ${pid}
    kill ${pid}
fi
ps ax |grep kermit |grep -v grep
pid=`ps ax |grep kermit |grep -v grep|awk '{print $1;}'`
if [ "X${pid}" != "X" ] ; then
    echo 'killing kermit, PID=' ${pid}
    kill -9 ${pid}
fi

ifconfig ppp0 down
ifconfig ppp0 delete

kermit -y /etc/ppp/kermit.dial
pppd /dev/tty01 19200
```

/etc/ppp/kermit.dial 은 전화 연결해서 원격 호스트에 필요한 모든 인증을 받는 **kermit** 스크립트다(이와 같은 예제 스크립트는 이 문서의 마지막에 있다)

PPP 라인을 끊기 위해 다음 /etc/ppp/pppdown 스크립트를 사용한다:



---

```
#!/bin/sh
pid=`ps ax |grep pppd |grep -v grep|awk '{print $1;}'`
if [ X${pid} != "X" ] ; then
    echo 'killing pppd, PID=' ${pid}
    kill -TERM ${pid}
fi

ps ax |grep kermit |grep -v grep
pid=`ps ax |grep kermit |grep -v grep|awk '{print $1;}'`
if [ "X${pid}" != "X" ] ; then
    echo 'killing kermit, PID=' ${pid}
    kill -9 ${pid}
fi

/sbin/ifconfig ppp0 down
/sbin/ifconfig ppp0 delete
kermit -y /etc/ppp/kermit.hup
/etc/ppp/ppptest
```

다음과 비슷한 /usr/etc/ppp/ppptest 를 실행하여 PPP 가 아직 동작 중인지 체크한다:

```
#!/bin/sh
pid=`ps ax | grep pppd |grep -v grep|awk '{print $1;}'`
if [ X${pid} != "X" ] ; then
    echo 'pppd running: PID=' ${pid-NONE}
else
    echo 'No pppd running.'
fi
set -x
netstat -n -l ppp0
ifconfig ppp0
```

모뎀을 정지시키려면 다음 내용이 있는 /etc/ppp/kermit.hup 을 실행한다:

---

```
set line /dev/tty01 ; put your modem device here
set speed 19200
set file type binary
set file names literal
set win 8
set rec pack 1024
set send pack 1024
set block 3
set term bytesize 8
set command bytesize 8
set flow none

pau 1
out +++
inp 5 OK
out ATH0W13
echo W13
exit
```

여기 kermit 대신 chat 을 사용한 다른 방법이 있다.

다음 두 개의 파일이 pppd 에 접속하는데 충분하다.

/etc/ppp/options:

```
/dev/cuaa1 115200

rtscts      # enable hardware flow control
modem       # modem control line
connect "/usr/bin/chat -f /etc/ppp/login.chat.script"
noipdefault # remote PPP serve must supply your IP address.
             # if the remote host doesn't send your IP during
             # IPCP negotiation, remove this option
passive     # wait for LCP packets
domain <your.domain> # put your domain name here
```

---

```

:      # put the IP of remote PPP host here
      # it will be used to route packets via PPP link
      # if you didn't specified the noipdefault option
      # change this line to <local_ip>:<remote_ip>

defaultroute  # put this if you want that PPP server will be
              # your default router

```

/etc/ppp/login.chat.script:

**Note:** 다음은 한 라인이다.

```

ABORT BUSY ABORT 'NO CARRIER' "" AT OK ATDT<phone.number>
CONNECT "" TIMEOUT 10 ogin:wwrogin: <login-id>
TIMEOUT 5 sword: <password>

```

이 스크립트들을 정확히 수정해서 설치하고 이제 필요한 것은 다음과 같이 pppd 를 실행한다:

```
# pppd
```

### 21.3.3 pppd 서버로 사용

/etc/ppp/options 은 다음 내용과 비슷한 내용을 가지고 있다:

```

crtscts          # Hardware flow control
netmask 255.255.255.0  # netmask ( not required )
192.114.208.20:192.114.208.165 # ip's of local and remote hosts
                                # local ip must be different from one
                                # you assigned to the ethernet ( or other )
                                # interface on your machine.
                                # remote IP is ip address that will be
                                # assigned to the remote machine

domain ppp.foo.com      # your domain

```

---

|         |                |
|---------|----------------|
| passive | # wait for LCP |
| modem   | # modem line   |

다음 /etc/ppp/pppserv 스크립트는 pppd 를 서버로 동작하게 한다:

```
#!/bin/sh
ps ax |grep pppd |grep -v grep
pid=`ps ax |grep pppd |grep -v grep|awk '{print $1;}'`
if [ "X${pid}" != "X" ] ; then
    echo 'killing pppd, PID=' ${pid}
    kill ${pid}
fi
ps ax |grep kermit |grep -v grep
pid=`ps ax |grep kermit |grep -v grep|awk '{print $1;}'`
if [ "X${pid}" != "X" ] ; then
    echo 'killing kermit, PID=' ${pid}
    kill -9 ${pid}
fi

# reset ppp interface
ifconfig ppp0 down
ifconfig ppp0 delete

# enable autoanswer mode
kermit -y /etc/ppp/kermit.ans

# run ppp
pppd /dev/tty01 19200
```

서버를 정지 시킬 때 /etc/ppp/pppservdown 스크립트를 사용한다:

---

```
#!/bin/sh
ps ax |grep pppd |grep -v grep
pid=`ps ax |grep pppd |grep -v grep|awk '{print $1;}'`
if [ "X${pid}" != "X" ] ; then
    echo 'killing pppd, PID=' ${pid}
    kill ${pid}
fi
ps ax |grep kermit |grep -v grep
pid=`ps ax |grep kermit |grep -v grep|awk '{print $1;}'`
if [ "X${pid}" != "X" ] ; then
    echo 'killing kermit, PID=' ${pid}
    kill -9 ${pid}
fi
ifconfig ppp0 down
ifconfig ppp0 delete

kermit -y /etc/ppp/kermit.noans
```

다음 **kermit** 스크립트(/etc/ppp/kermit.ans)는 모뎀의 자동응답 모드를 활성화/비활성 한다:

```
set line /dev/tty01
set speed 19200
set file type binary
set file names literal
set win 8
set rec pack 1024
set send pack 1024
set block 3
set term bytesize 8
set command bytesize 8
set flow none

pau 1
out +++
inp 5 OK
```

---

```
out ATH0W13
inp 5 OK
echo W13
out ATS0=1W13 ; change this to out ATS0=0W13 if you want to disable
                ; autoanswer mode inp 5 OK
echo W13
exit
```

/etc/ppp/kermit.dial 스크립트는 원격호스트로 전화 연결해서 인증 받는데 사용된다. 이 파일을 필요에 따라 수정해야 된다. 이 스크립트에 로그인 이름과 패스워드를 넣고 모뎀과 원격호스트의 응답에 따라 입력문을 변경해야 된다.

```

;
; put the com line attached to the modem here:
;
set line /dev/tty01
;
; put the modem speed here:
;
set speed 19200
set file type binary          ; full 8 bit file xfer
set file names literal
set win 8
set rec pack 1024
set send pack 1024
set block 3
set term bytesize 8
set command bytesize 8
set flow none
set modem hayes
set dial hangup off
set carrier auto             ; Then SET CARRIER if necessary,
set dial display on         ; Then SET DIAL if necessary,
set input echo on
set input timeout proceed
```

---

```

set input case ignore
def W%x 0                ; login prompt counter
goto slhup

:slcmd                    ; put the modem in command mode
echo Put the modem in command mode.
clear                    ; Clear unread characters from input buffer
pause 1
output +++               ; hayes escape sequence
input 1 OKW13W10        ; wait for OK
if success goto slhup
output W13
pause 1
output atW13
input 1 OKW13W10
if fail goto slcmd      ; if modem doesn't answer OK, try again

:slhup                    ; hang up the phone
clear                    ; Clear unread characters from input buffer
pause 1
echo Hanging up the phone.
output ath0W13          ; hayes command for on hook
input 2 OKW13W10
if fail goto slcmd      ; if no OK answer, put modem in command mode

:sdial                    ; dial the number
pause 1
echo Dialing.
output atdt9,550311W13W10 ; put phone number here
assign W%x 0            ; zero the time counter

:look
clear                    ; Clear unread characters from input buffer
increment W%x           ; Count the seconds
input 1 {CONNECT }
if success goto sllogin

```

---

---

```

reinput 1 {NO CARRIERW13W10}
if success goto sldial
reinput 1 {NO DIALTONEW13W10}
if success goto slnodial
reinput 1 {W255}
if success goto slhup
reinput 1 {W127}
if success goto slhup
if < W%x 60 goto look
else goto slhup

:slogin                                ; login
assign W%x 0                          ; zero the time counter
pause 1
echo Looking for login prompt.

:sloop
increment W%x                          ; Count the seconds
clear                                   ; Clear unread characters from input buffer
output W13
;
; put your expected login prompt here:
;
input 1 {Username: }
if success goto sluid
reinput 1 {W255}
if success goto slhup
reinput 1 {W127}
if success goto slhup
if < W%x 10 goto slloop                ; try 10 times to get a login prompt
else goto slhup                        ; hang up and start again if 10 failures

:sluid
;
; put your userid here:
;

```



```

output ppp-loginW13
input 1 {Password: }
;
; put your password here:
;
output ppp-passwordW13
input 1 {Entering SLIP mode.}
echo
quit

:slnodial
echo W7No dialtone. Check the telephone line!W7
exit 1

; local variables:
; mode: csh
; comment-start: "; "
; comment-start-skip: "; "
; end:

```

## 21.4 PPP 접속 문제 해결

모뎀을 사용하여 접속할 때 사용자가 해결해야 될 여러 가지 문제가 있다. 이 섹션은 발생할 수 있는 몇 가지 문제에 대해 다룬다. 예를 들어 전화접속으로 시스템에 로그인할 때 어떤 프롬프트가 나타나는지 정확히 알고 있어야 될 필요가 있을 것이다. 어떤 ISP는 *ssword* 프롬프트를 보여주고 다른 ISP는 *password*를 보여 준다; ppp 스크립트를 정확히 작성하지 않았다면 로그인은 실패한다. ppp 연결을 디버그하는 가장 일반적인 방법은 직접 연결하는 것이다. 다음 정보는 직접 단계적으로 연결을 수행한다.

### 121.4.1 장치 노드 체크

커널을 다시 설정했다면 sio 장치를 다시 확인한다. 커널을 설정하지 않았다면 걱정할 필요가 없다. 단지 다음 명령으로 dmesg 출력에서 모뎀 장치를 체크한다:

---

**#dmesg | grep sio**

sio 장치에 관련된 적절한 결과를 보게 되고 이 결과는 우리에게 필요한 COM 포트다. 모뎀이 표준 시리얼 포트처럼 동작한다면 sio1 이나 COM2 에 나열된 것을 보게 된다. 그렇다면 커널을 다시 빌드 할 필요 없이 시리얼 장치를 만들면 된다. /dev 디렉터리로 이동하여 위에서처럼 MAKEDEV 스크립트를 실행하여 시리얼 장치를 만들 수 있다. 이제 다음 명령으로 시스템에 시리얼 장치를 만든다:

**# sh MAKEDEV cuaa0 cuaa1 cuaa2 cuaa3**

DOS 에서 sio 모뎀이 sio1 나 COM2 와 매칭되면 모뎀 장치는 /dev/cuaa1 이 된다.

## 21.4.2 직접 연결

직접 ppp 를 제어하여 인터넷에 연결하는 것은 쉽고 빠르며, 연결을 디버그 하는 가장 좋은 방법으로 ISP 에서 ppp 클라이언트 연결을 어떻게 처리하는지 정보를 얻게 된다. 명령어 라인에서 PPP 를 시작한다. 이곳의 모든 예제에서 PPP 가 동작하는 머신의 호스트 이름으로 *example* 을 사용한다. 그냥 ppp 를 입력하여 ppp 를 시작한다:

**# ppp**

이제 ppp 가 시작되었다.

ppp ON example> **set device /dev/cuaa1**

모뎀 장치를 설정하였고 여기서 모뎀 장치는 cuaa1 이다.

ppp ON example> **set speed 115200**

연결 속도 설정에서 이 경우 115,200 kbps 를 사용한다.

ppp ON example> **enable dns**

ppp 가 리졸버를 설정할 수 있도록 /etc/resolv.conf 에 네임서버 라인을 추가하도록 한다.

---

ppp 가 호스트 이름을 결정하지 못했다면 나중에 직접 설정할 수 있다.

ppp ON example> **term**

"터미널" 모드로 변경하여 직접 모뎀을 제어할 수 있다.

```
deflink: Entering terminal mode on /dev/cuaa1
type '~h' for help
```

**at**

**OK**

**atdt 123456789**

at 을 사용하여 모뎀을 초기화하고 atdt 와 ISP 번호를 사용하여 전화 걸기를 시작하게 한다.

CONNECT

연결을 확인하고 하드웨어와 관련 없는 문제가 발생한다면 여기서 문제를 해결할 것이다.

ISP Login:**myusername**

유저 이름을 입력하는 프롬프트에 ISP 가 제공한 유저 이름을 넣고 Enter 를 누른다.

ISP Pass:**mypassword**

패스워드를 입력하는 프롬프트에 ISP 가 제공한 패스워드를 입력한다. FreeBSD 에 로그인 하는 것처럼 패스워드는 나타나지 않는다.

Shell or PPP:**ppp**

ISP 에 따라서 이 프롬프트는 절대 나타나지 않을 수 있다. 여기서 ISP 에게 쉘 사용을 요청하거나 ppp 를 시작할 수 있다. 이 예제에서는 인터넷에 연결하려고 ppp 사용을 선택했다.

Ppp ON example>

---

이 예제에서 첫 번째 P 는 대문자임을 주의한다. 성공적으로 ISP 에 연결됐음을 보여준다.

PPp ON example>

성공적으로 ISP 인증을 거쳤고 IP 주소 할당을 기다린다.

PPP ON example>

IP 주소를 받았고 연결은 성공적으로 완료됐다.

PPP ON example>**add default HISADDR**

현재 peer 만 유일하게 연결되었기 때문에 외부와 연결하기 전에 기본 경로(라우트)를 추가해야 된다. 예전 경로 때문에 기본 경로를 추가하는데 실패했다면 *add* 앞에 ! 문자를 입력할 수 있다. 아니면 실제로 연결하기 전에 이렇게 설정하여 새로운 경로를 적절히 설정할 수 있다.

모든 것이 정상이면 **CTRL + z** 를 사용하여 백그라운드로 바꿀 수 있는 인터넷 연결이 활성화되고 PPP 가 ppp 로 바뀌면 연결은 끊어진다. 연결 상태를 보여주기 때문에 확인하는데 유용하다. 대문자 P 는 ISP 와 연결되었음을 소문자 p 는 어떤 이유에서건 연결이 끊어진 것을 보여준다. ppp 는 2 가지 상태만 있다.

### 21.4.2.1 디버깅

직접 연결은 했지만 연결이 안된 것처럼 보이면 *set ctsrts off* 로 하드웨어 흐름제어에 사용되는 CTS/RTS 를 끈다. 주된 이유는 데이터를 통신 링크에 보내려고 하면 PPP 가 멈추는 어떤 PPP 터미널 서버에 연결되어 있어서 절대 오지 않는 CTS 나 Clear To Send 신호를 기다리게 된다. 그러나 이 옵션을 사용한다면 끝 단에서 끝 단까지 특정 문자(XON/XOFF 등)를 보낼 때 하드웨어 의존성을 없애기 위해 필요할 수 있는 *set accmap* 옵션도 사용해야 된다. 이 옵션에 대한 정보와 사용법은 ppp(8) 매뉴얼 페이지를 참고한다.

오래된 모뎀을 가지고 있다면 *set parity even* 을 사용해야 될 것이다. 패리티가 기본적으로 none 으로 설정되므로 오래된 모뎀과 몇몇 ISP 의 에러 체크에 사용된다.

---

ISP 가 교신을 시작하기 위해 여러분 측의 준비를 기다리는 동안 PPP 는 보통 교신 에러가 있는 명령어 모드를 되돌려 주지 않을 것이다. 여기서 ~p 명령을 사용하면 강제로 ppp 가 설정 정보를 보낸다.

로그인 프롬프트를 보지 못했다면 위의 예제에서 유닉스 스타일의 인증 대신 PAP 나 CHAP 인증을 사용해야 된다. PAP 나 CHAP 를 사용하려면 터미널 모드로 들어가기 전에 PPP 에게 다음 옵션을 추가한다:

```
ppp ON example> set authname myusername
```

*myusername* 은 ISP 에서 할당해준 유저 이름으로 대체한다.

```
ppp ON example> set authkey mypassword
```

*mypassword* 는 ISP 에게 할당 받은 패스워드로 대체한다.

연결은 잘 되었지만 도메인 이름을 찾지 못하는 것처럼 보인다면 IP 주소로 ping(8)을 사용하여 결과를 확인한다. 100% 패킷을 잃어버린다면 대부분 기본 경로를 할당하지 않았을 것이다. 연결되어 있는 동안 옵션 *add default HISADDR* 가 설정되어 있는지 다시 체크한다. IP 주소로 연결할 수 있다면 리졸버 주소가 */etc/resolve.conf* 에 추가되지 않았을 것이다. 이 파일은 다음과 비슷하다:

```
domain example.com
nameserver x.x.x.x
nameserver y.y.y.y
```

*x.x.x.x* 와 *y.y.y.y* 는 ISP 의 DNS 서버 IP 주소로 대체한다. 이 정보는 ISP 와 계약할 때 제공받았거나 그렇지 않았을 수 있지만 ISP 와 통화하여 해결한다.

syslog(3)으로 PPP 연결에 로그 기능을 제공할 수 있다. 그냥 다음 라인을 */etc/syslog.conf* 에 추가한다:

```
!ppp
*. * /var/log/ppp.log
```

---

대부분 이 기능이 이미 설정되어 있다.

## 21.5 PPP over Ethernet(PPPoE) 사용

이번 섹션은 PPP over Ethernet(PPPoE)를 어떻게 설정하는지 설명한다.

### 21.5.1 커널 설정

PPPoE 를 위해 특별히 커널을 설정할 필요는 없다. **netgraph** 지원이 필요하다면 커널에 빌드하지 않고 **ppp** 를 통해 동적으로 로드 한다.

**Netgraph:** Netgraph 는 잘 정의된 각각의 태스크를 수행하도록 디자인된 툴들을 조합하여 강력하고 유연한 네트워킹 기능을 수행하는 in-kernel 네트워킹 서브시스템이다.

### 21.5.2 ppp.conf 설정

여기에 정확히 동작하는 ppp.conf 의 예제가 있다:

```
default:
  set log Phase tun command # you can add more detailed logging if you wish
  set ifaddr 10.0.0.1/0 10.0.0.2/0

name_of_service_provider:
  set device PPPoE:xl1 # replace xl1 with your ethernet device
  set authname YOURLOGINNAME
  set authkey YOURPASSWORD
  set dial
  set login
  add default HISADDR
```

### 21.5.3 PPP 실행

---

root 에서 다음 명령을 실행한다:

```
# ppp -ddial name_of_service_provider
```

## 21.5.4 부팅할 때 PPP 시작하기

다음 내용을 /etc/rc.conf 파일에 추가하다:

```
ppp_enable="YES"
ppp_mode="ddial"
ppp_nat="YES" # if you want to enable nat for your local network, otherwise NO
ppp_profile="name_of_service_provider"
```

## 21.5.5 PPPoE 서비스 태그 사용

연결하기 위해 서비스 태그(service tag)를 사용해야 될 경우가 종종 있다. 서비스 태그는 네트워크에 있는 다른 PPPoE 서버들을 구분하는데 사용된다.

ISP 가 제공한 문서에서 필요한 서비스 태그 정보를 받는다. 이 정보가 없다면 ISP 의 기술 지원을 받는다.

마지막으로 포트 컬렉션에서 찾을 수 있는 Roaring Penguin PPPoE

프로그램(<http://www.roaringpenguin.com/pppoe/>)이 제시하는 방법을 시도해 볼 수 있다.

그러나 이 방법은 모뎀 프로그램을 사용하지 못하게 될 수 있기 때문에 시도하기 전에 다시 생각해 본다. 그냥 여러분의 ISP가 제공한 모뎀에 맞는 프로그램을 설치한다. 그리고 프로그램에서 System 메뉴에 접근한다. 여러분의 프로필 이름에 보통 *ISP*라고 설정되어 있다.

프로필 이름(서비스 태그)은 set device 명령(더욱 자세한 내용은 ppp(8) 매뉴얼 페이지를 본다)의 공급자 부분으로 ppp.conf 의 PPPoE 설정 엔트리에 사용된다. 다음과 비슷하다:

```
set device PPPoE:x11:ISP
```

*x11* 을 여러분의 이더넷 카드에 맞게 변경하는 것을 잊지 않는다.

---

ISP를 위에서 찾은 프로파일로 변경하는 것을 잊지 않는다.

추가적인 정보는 아래 사이트를 참고한다:

- Renaud Waldura의 Cheaper Broadband with FreeBSD on DSL(<http://renaud.waldura.com/doc/freebsd/pppoe/>).
- Udo Erdelhoff (독일인) Nutzung von T-DSL und T-Online mit FreeBSD(<http://www.ruhr.de/home/nathan/FreeBSD/tdsl-freebsd.html>).

## 21.5.6 3Com® HomeConnect® ADSL Modem Dual

### Link 로 PPPoE 설정하기

이 모뎀은 RFC 2515(<http://www.fags.org/rfcs/rfc2516.html>) (L.Mamakos, K.Lidl, J.Evarts, D.Carrel, D.Simone 그리고 R.Wheeler이 작성한 PPP over Ethernet(PPPoE)을 따르지 않는다. 대신 이더넷 프레임에 다른 패킷 타입 코드를 사용한다. 이 모뎀이 PPPoE 규정을 따라야 한다고 생각하면 3Com에(<http://www.3com.com/>) 항의한다.

FreeBSD가 이 장치와 통신하도록 하려면 `sysctl` 을 설정해야 된다. `sysctl` 설정은 `/etc/sysctl.conf` 를 수정하여 부팅할 때 자동으로 수행한다:

```
net.graph.nonstandard_pppoe=1
```

아니면 `sysctl net.graph.nonstandard_pppoe=1` 명령을 사용하여 즉시 결과를 알 수 있다.

시스템 전반에 걸친 설정이기 때문에 일반적인 PPPoE 클라이언트나 서버를 3Com® HomeConnect® ADSL Modem 과 함께 여기서 설명하는 것은 불가능하다.

## 21.6 PPP 를 ATM 으로 사용 (PPPoA)

이 섹션은 PPP over ATM(PPPoA)을 어떻게 설정하는지 다룬다. PPPoA 는 유럽의 DSL



---

공급자들이 흔히 사용한다.

## 21.6.1 Alcatel SpeedTouch™ USB 로 PPPoA 사용하기

펌웨어가 Alcatel의 라이선스([http://www.alcatel.com/consumer/dsl/disclaimer\\_ix.htm](http://www.alcatel.com/consumer/dsl/disclaimer_ix.htm))로 배포되어서 FreeBSD의 기본 시스템에 무료로 재 배포할 수 없기 때문에 이 장치를 지원하는 PPPoA는 FreeBSD 포트에서 공급한다.

소프트웨어를 설치하려면 단순히 포트 컬렉션을 이용한다. [net/pppoa](#) 포트를 설치하고 제공된 지시를 따른다.

많은 USB 장치처럼 Alcatel SpeedTouch™ USB 도 호스트 컴퓨터에서 운영에 맞는 펌웨어를 다운로드 해야 된다. FreeBSD 에서 자동으로 다운로드 할 수 있기 때문에 장치가 USB 포트에 연결될 때마다 펌웨어는 자동으로 로드 된다. 자동으로 펌웨어를 전송하도록 다음 정보를 `/etc/usbd.conf` 파일에 추가한다. 이 파일은 root 유저에서 수정해야 된다.

```
device "Alcatel SpeedTouch USB"
  devname "ugen[0-9]+"
```

vendor 0x06b9  
product 0x4061  
attach "/usr/local/sbin/modem\_run -f /usr/local/libdata/mgmt.o"

USB 데몬 `usbd` 를 활성화하려면 다음 라인을 `/etc/rc.conf` 에 추가한다:

```
usbd_enable="YES"
```

시작할 때 PPP 가 다이얼-업 하도록 설정하는 것도 가능하다. 다이얼-업 하도록 설정하려면 다음 라인을 root 유저로 `/etc/rc.conf` 에 추가한다.

```
ppp_enable="YES"
ppp_mode="ddial"
ppp_profile="adsl"
```

정확하게 동작하도록 [net/pppoa](#) 포트에서 제공하는 샘플 `ppp.conf` 를 사용한다.

---

## 21.6.2 mpd 사용

다양한 서비스(특히 PPTP)에 연결하기 위해 **mpd** 를 사용할 수 있다. **mpd** 는 포트 컬렉션의 [net/mpd](#) 에서 찾을 수 있다. 다양한 ADSL 모뎀은 모뎀과 컴퓨터 사이에 생성된 PPTP 터널을 필요로 하고 이러한 모뎀 중 하나가 Alcatel SpeedTouch Home 이다.

첫째로 포트를 설치하고 여러분의 필요성과 ISP 의 설정에 따라 **mpd** 를 설정한다. 포트의 PREFIX/etc/mpd/에 문서화가 잘 되어있는 샘플 설정파일 세트가 있다. PREFIX 의 의미는 포트가 설치되어 있는 디렉터리를 의미하고 기본적으로 /usr/local/에 위치한다. 완벽한 **mpd** 설정 가이드는 포트가 설치된 후 PREFIX/share/mpd/에서 HTML 포맷으로 확인할 수 있다. 여기에 **mpd** 로 ADSL 서비스에 연결하는 샘플 설정이 있다. 이 설정은 두 개의 파일로 나뉘어있고 첫 번째는 mpd.conf 이다:

```
default:
    load adsl

adsl:
    new -i ng0 adsl adsl
    set bundle authname username ❶
    set bundle password password ❷
    set bundle disable multilink

    set link no pap acfcomp protocomp
    set link disable chap
    set link accept chap
    set link keep-alive 30 10

    set ipcp no vjcomp
    set ipcp ranges 0.0.0.0/0 0.0.0.0/0

    set iface route default
    set iface disable on-demand
    set iface enable proxy-arp
    set iface idle 0
```

---

```
open
```

- ❶ ISP 인증에 사용되는 유저 이름.
- ❷ ISP 인증에 사용되는 패스워드.

mpd.links 파일은 연결을 원하는 링크에 대한 정보를 가지고 있다. 예제 mpd.links 는 위의 예제에 덧붙인 것이다.

```
adsl:
  set link type pptp
  set pptp mode active
  set pptp enable originate outcall
  set pptp self 10.0.0.1 ❶
  set pptp peer 10.0.0.138 ❷
```

- ❶ mpd 를 사용할 FreeBSD 컴퓨터의 IP 주소
- ❷ ADSL 모뎀의 IP 주소. Alcatel SpeedTouch Home 의 주소는 기본적으로 10.0.0.138 이다.

root 에서 다음 명령으로 연결을 쉽게 초기화할 수 있다.

```
# mpd -b adsl
```

다음 명령으로 연결 상태를 볼 수 있다.

```
% ifconfig ng0
ng0: flags=88d1<UP,POINTOPOINT,RUNNING,NOARP,SIMPLEX,MULTICAST> mtu 1500
    inet 216.136.204.117 --> 204.152.186.171 netmask 0xffffffff
```

mpd 를 사용하여 FreeBSD 에서 ADSL 서비스에 연결하는 것을 권장한다.

## 21.6.3 pptpclient 사용

[net/pptpclient](#) 를 사용하여 FreeBSD 를 다른 PPPoA 서비스에 연결하는 것도 가능하다.

---

`net/pptpclient`를 사용하여 DSL 서비스에 연결하려면 포트나 패키지를 설치하고 `/etc/ppp/ppp.conf`를 수정한다. 이들 작업은 root 권한이 필요하다. `ppp.conf`의 예제 섹션은 아래에 있다. 더 많은 `ppp.conf` 옵션은 **ppp** 매뉴얼 페이지 `ppp(8)`를 참고한다.

```
adsl:
set log phase chat lcp ipcp ccp tun command
set timeout 0
enable dns
set authname username ❶
set authkey password ❷
set ifaddr 0 0
add default HISADDR
```

- ❶ DSL 공급자의 유저계정 이름
- ❷ 계정의 패스워드

**주의:** `ppp.conf` 파일에 평범한 텍스트 형식으로 계정의 패스워드를 입력해야 되므로 아무도 이 파일의 내용을 볼 수 없도록 해야 된다. 다음 명령들은 이 파일을 root 계정으로만 읽을 수 있도록 한다. `chmod(1)`과 `chown(8)`에 대한 더 많은 정보는 매뉴얼 페이지를 참고한다.

```
# chown root:wheel /etc/ppp/ppp.conf
# chmod 600 /etc/ppp/ppp.conf
```

이 설정은 DSL 라우터의 PPP 세션 터널을 연다. 이더넷 DSL 모뎀에는 여러분이 연결할 LAN IP가 미리 설정되어 있다. Alcatel SpeedTouch Home의 경우 주소는 10.0.0.138이다. 라우터 문서에서 장치가 어떤 IP를 사용하는지 확인한다. 터널을 열고 `ppp` 세션을 시작하려면 다음 명령을 실행한다.

```
# pptp address isp
```

**Tip:** `pptp`는 프롬프트를 돌려주지 않기 때문에 이전 명령의 마지막에 앰퍼샌드("&")를 추가할 수 있다.

tun 가상 터널 장치는 `pptp`와 `ppp` 프로세스 사이에서 생성된다. 프롬프트를 되돌려

---

받았거나 **pptp** 프로세스가 연결을 확인했다면 다음과 같이 터널을 확인할 수 있다.

```
% ifconfig tun0
tun0: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1500
      inet 216.136.204.21 --> 204.152.186.171 netmask 0xfffff00
      Opened by PID 918
```

연결할 수 없다면 보통 telnet 이나 웹 브라우저로 라우터의 설정을 체크한다. 그래도 연결할 수 없다면 **pptp** 명령의 출력과 **ppp** 로그 파일 `/var/log/ppp.log` 의 내용을 검토해본다.

## 21.7 SLIP 사용

### 21.7.1 SLIP 클라이언트 설정

다음은 FreeBSD 머신을 고정 호스트 네트워크에서 SLIP(serial Line Internet Protocol: TCP/IP 네트워크에 다이얼-업 접속을 할 수 있도록 해주는 데이터링크 프로토콜) 클라이언트로 설정하는 방법이다. 유동적인 호스트이름 할당에 대해서는(다이얼-업 할 때 마다 IP 주소가 변경되는) 더욱 복잡한 설정이 필요할 것이다.

우선 어떤 시리얼 포트에 모뎀을 연결할지 결정한다. 많은 사람들은 `/dev/modem` 과 같은 심볼릭 링크를 실제 이름 `/dev/cuaaN` 에 설정한다. 이 방법은 실제 장치 이름을 추상화하여 모뎀을 다른 포트로 이동시킬 필요가 없게 된다. 시스템의 `/etc` 파일들과 `.kermrc` 파일을 전체적으로 수정해야 될 때 상당히 귀찮은 일이 될 수 있다.

**Note:** `/dev/cuaa0` 는 COM1, `cuaa1` 은 COM2 등으로 대응된다.

다음 내용을 커널 설정파일에 추가한다:

```
pseudo-device  sl      1
```

FreeBSD 5.X 에서는 다음 라인을 대신 사용한다:

---

```
device sl
```

위 라인은 GENERIC 커널에 포함되어 있어서 이 내용을 지우지 않았다면 문제가 없다.

### 21.7.1.1 최초 설정

- ① 머신의 게이트웨이와 네임서버를 `/etc/hosts` 파일에 추가한다. 필자의 경우 `hosts` 파일은 다음과 같다:

```
127.0.0.1          localhost loghost
136.152.64.181    water.CS.Example.EDU water.CS water
136.152.64.1      inr-3.CS.Example.EDU inr-3 slip-gateway
128.32.136.9      ns1.Example.EDU ns1
128.32.136.12     ns2.Example.EDU ns2
```

- ② FreeBSD 5.0 전 버전에서는 `bind` 전에 `hosts`가 `/etc/host.conf` 파일에 나열되어 있어야 된다. FreeBSD 5.0부터 이 파일 대신 `/etc/nsswitch.conf` 파일을 대신 사용하므로 이 파일의 `dns` 전의 `hosts` 라인에 `files`가 있어야 된다. 이러한 매개변수가 없다면 재미있는 일이 발생할 것이다.

- ③ `/etc/rc.conf` 파일을 수정한다.

- A. 다음 라인을 수정하여 호스트 이름을 설정한다:

```
hostname="myname.my.domain"
```

머신의 완벽한 인터넷 호스트 이름이 이곳에 필요하다.

- B. 다음 라인을 변경하여 `sl0`를 네트워크 인터페이스 리스트에 추가한다:

```
network_interfaces="lo0" 라인을
network_interfaces="lo0 sl0"으로 변경한다.
```

- 
- C. 다음 라인을 추가하여 sl0의 시작 플래그를 설정한다:

```
ifconfig_sl0="inet ${hostname} slip-gateway netmask 0xfffff00 up"
```

- D. 다음 라인을 수정하여 기본 라우터를 설정한다.

```
defaultrouter="NO" 라인을  
defaultrouter="slip-gateway"으로 수정한다.
```

- ④ 다음 내용을 포함하는 /etc/resolv.conf 파일을 생성한다.

```
domain CS.Example.EDU  
nameserver 128.32.136.9  
nameserver 128.32.136.12
```

이들 설정은 네임 서버 호스트 설정이다. 물론 실제 도메인 이름과 주소는 여러분의 환경을 따른다.

- ⑤ root와 toor 패스워드를 설정한다(그리고 패스워드가 없는 다른 계정들까지).
- ⑥ 머신을 재 부팅하고 정확한 호스트 이름인지 확인한다.

### 21.7.1.2 SLIP 접속

- ① 프롬프트에 slip를 입력하여 다이얼-업 한 후 머신 이름과 패스워드를 입력한다. 필요한 사항은 여러분의 환경에 따라 다르다. kermite를 사용한다면 다음과 같은 스크립트를 사용한다:

```
# kermite setup  
set modem hayes  
set line /dev/modem  
set speed 115200
```

---

```
set parity none
set flow rts/cts
set terminal bytesize 8
set file type binary
# The next macro will dial up and login
define slip dial 643-9600, input 10 =>, if failure stop, -
output slipWx0d, input 10 Username:, if failure stop, -
output silviaWx0d, input 10 Password:, if failure stop, -
output ***Wx0d, echo Wx0aCONNECTEDWx0a
```

물론 여러분의 환경에 맞게 호스트 이름과 패스워드를 변경한다. 이렇게 한 후 해 kermit 프롬프트에 slip 을 입력하여 연결할 수 있다.

**Note:** 파일시스템에 평범한 텍스트로 패스워드를 두는 것은 일반적으로 좋지 못한 생각이다. 위험 요소가 된다.

- ② kermit을 그 상태로 두고(**Ctrl-z**를 사용하여 중단할 수 있다) root에서 다음 명령을 입력한다:

```
# slattach -h -c -s 115200 /dev/modem
```

라우터 밖의 다른 호스트에 ping 이 나간다면 연결이 된 것이다. ping 이 나가지 않는다면 slattach 인자를 *-c* 대신 *-a*로 바꾸고 다시 시도한다.

### 21.7.1.3 연결을 어떻게 끊는가

slattach 를 중지하기 위해 다음 명령을 실행한다:

```
# kill -INT `cat /var/run/slattach.modem.pid`
```

위 명령은 root 에서 실행해야 된다. kermit 으로(중단 시켰다면 fg 를 실행하여) 되돌아가서 빠져나간다(q).

slattach 매뉴얼 페이지에서는 인터페이스를 다운시키는데 ifconfig sl0 down 을 사용하도록 하지만 이것은 여기서 알려 준 방법과 차이가 없어 보인다(ifconfig sl0 도 같은 내용을



---

출력한다).

종종 모뎀이 전송을 끊는 것을 거부한다(가끔). 이 경우 단순히 kermit 를 시작하고 다시 빠져나간다. 보통 두 번 정도하면 끊어진다.

#### 21.7.1.4 문제 해결

작동하지 않는다면 메일링 리스트에 문의한다. 이제까지 사람들이 보고한 내용:

- slattach에서 `-c`나 `-a`를 사용하지 않는다(이것은 치명적이지 않지만 어떤 유저는 자신들의 문제를 해결했다고 보고했다).
- `s10` 대신 `s100`을 사용했다(어떤 폰트에서는 차이점을 찾기가 어려울 것이다).
- 인터페이스 상태를 보기 위해 `ifconfig s10`를 사용한다. 예를 들어 다음과 같이 입력한다:

```
# ifconfig s10
s10: flags=10<POINTOPOINT>
    inet 136.152.64.181 --> 136.152.64.1 netmask ffffff00
```

- ping 테스트에서 "no route to host"라는 메시지를 받았다면 라우팅 테이블과 관련된 문제일 것이다. `netstat -r` 명령을 사용하여 현재 경로를 볼 수 있다:
- 

```
# netstat -r
Routing tables
Destination      Gateway          Flags    Refs    Use  IfaceMTU
Rtt    Netmasks:

(root node)
(root node)

Route Tree for Protocol Family inet:
(root node) =>
default          inr-3.Example.EDU  UG          8    224515  s10 -    -
```

|                                         |    |          |     |   |   |
|-----------------------------------------|----|----------|-----|---|---|
| localhost.Exampl localhost.Example. UH  | 5  | 42127    | lo0 | - |   |
| 0.438                                   |    |          |     |   |   |
| inr-3.Example.ED water.CS.Example.E UH  | 1  | 0        | sl0 | - | - |
| water.CS.Example localhost.Example. UGH | 34 | 47641234 | lo0 | - |   |
| 0.438                                   |    |          |     |   |   |
| (root node)                             |    |          |     |   |   |

앞의 예제는 부하가 많은 시스템이다. 여러분 시스템의 숫자는 네트워크 활동 변화에 따라 다르다.

## 21.7.2 SLIP 서버 설정

이 문서는 FreeBSD 시스템에서 SLIP 서버 서비스를 제공하도록 설정을 지원한다. 전적으로 이 의미는 원격 클라이언트의 로그인에 따라 자동으로 연결이 시작되도록 시스템을 설정한다는 것이다.

### 21.7.2.1 설정하기 전 전제조건

이 섹션은 매우 기술적이어서 백그라운드 지식이 필요하다. 특히 여러분이 TCP/IP 네트워크 프로토콜 및 네트워크와 노드 주소, 네트워크 주소 마스크, 서브 넷, 라우팅과 RIP 같은 라우팅 프로토콜에 대해 어느 정도 알고 있다고 가정한다. 다이얼-업 서버에서 SLIP 서비스 설정은 이러한 개념의 지식이 필요하고 이런 지식이 없다면 Craig Hunt의 O'Reilly 에서 발간한 TCP/IP 네트워크 관리(ISBN Number 0-937175-82-X)나 Douglas Comer 의 TCP/IP 프로토콜 책을 본다.

그리고 이미 모뎀을 설치했고 모뎀을 통해 로그인할 수 있도록 시스템 파일을 정확히 설정했다고 가정한다. 아직 준비하지 않았다면 다이얼-업 서비스 설정 튜토리얼을 본다; 웹 브라우저를 사용할 수 있다면 <http://www.FreeBSD.org/docs.html> 에서 튜토리얼 리스트를 볼 수 있다. 또한 시리얼 포트 장치 드라이버 정보는 `sio(4)` 매뉴얼 페이지를 본다. 모뎀에 로그인할 수 있도록 관련된 시스템 설정 정보는 `ttys(5)`, `gettytab(5)`, `getty(8)` & `init(8)`를 그리고 시리얼 포트 매개변수 설정정보는 `stty(1)` 매뉴얼 페이지를 체크하여 얻을 수 있다.

---

## 21.7.2.2 간단한 개요

일반적인 설정에서 FreeBSD 를 사용한 SLIP 서버는 다음과 같이 동작한다: SLIP 유저는 FreeBSD SLIP 서버 시스템에 다이얼-업하고 특수 유저 쉘 `/usr/sbin/sliplogin` 을 사용하는 특수한 SLIP 로그인 ID 로 로그인 한다. `sliplogin` 프로그램은 특수한 유저와 매칭되는 라인을 찾기 위해 `/etc/sliphome/slip.hosts` 파일을 검색한다. 매칭되는 라인을 찾았다면 이용할 수 있는 SLIP 인터페이스에 시리얼 라인을 연결하고 SLIP 인터페이스를 설정하기 위해 `/etc/sliphome/slip.login` 쉘 스크립트를 실행한다.

### 21.7.2.2.1 SLIP 서버 로그인 예제

예를 들어 SLIP 유저 ID 가 `Shelmerg` 라면 `/etc/master.passwd` 에서 `Shelmerg` 의 엔트리는 다음과 비슷하다:

```
Shelmerg:password:1964:89::0:0:Guy Helmer - SLIP:/usr/users/Shelmerg:/usr/sbin/sliplogin
```

`Shelmerg` 가 로그인하면 `sliplogin` 은 유저 ID 와 매칭되는 라인을 `/etc/sliphome/slip.hosts` 에서 찾는다; 예를 들어 `/etc/sliphome/slip.hosts` 의 라인은 다음과 같을 것이다:

```
Shelmerg      dc-slip sl-helmer      0xfffffc00      autocomp
```

`sliplogin` 이 매칭되는 라인을 찾으면 시리얼 라인은 이용할 수 있는 다음 SLIP 인터페이스에 연결한다. 그리고 다음과 같이 `/etc/sliphome/slip.login` 을 실행한다.

```
/etc/sliphome/slip.login 0 19200 Shelmerg dc-slip sl-helmer 0xfffffc00 autocomp
```

정상적이면 `/etc/sliphome/slip.login` 은 연결되어 있는 SLIP 인터페이스(위의 예제에서 `slip.login` 에 나열되어 첫 번째 매개변수가 주어진 `slip` 인터페이스 0)에 로컬 IP 주소(`dc-slip`), 원격 IP 주소(`sl-helmer`) 그리고 SLIP 인터페이스에 네트워크 마스크(`0xfffffc00`)와 부가적인 플래그(`autocomp`)를 설정하기 위해 `ifconfig` 를 사용한다. 문제가 발생하면 `sliplogin` 은 `syslog` 데몬을 통해서 쓸 만한 정보 메시지를 보통 `/var/log/messages` 에 로그로 남긴다(어떤 `syslogd` 가 로그로 남고 어디에 남는지 확인하려면 `/etc/syslog.conf` 를 체크하고 `syslogd(8)`와 `syslog.conf(5)` 매뉴얼 페이지를 참고한다).

이 정도로 해두고 시스템 설정으로 들어가보자.

---

### 21.7.2.3 커널 설정

FreeBSD의 기본 커널에는 보통 두 개의 SLIP 인터페이스(sl0와 sl1)가 정의되어 있다. 이들 인터페이스가 정의되어 있는지 netstat -i를 통해볼 수 있다.

netstat -i의 샘플 출력:

| Name | Mtu   | Network     | Address         | lpkts  | lerrs | Opkts  | Oerrs | Coll |
|------|-------|-------------|-----------------|--------|-------|--------|-------|------|
| ed0  | 1500  | <Link>      | 0.0.c0.2c.5f.4a | 291311 | 0     | 174209 | 0     | 133  |
| ed0  | 1500  | 138.247.224 | ivory           | 291311 | 0     | 174209 | 0     | 133  |
| lo0  | 65535 | <Link>      |                 | 79     | 0     | 79     | 0     | 0    |
| lo0  | 65535 | loop        | localhost       | 79     | 0     | 79     | 0     | 0    |
| sl0* | 296   | <Link>      |                 | 0      | 0     | 0      | 0     | 0    |
| sl1* | 296   | <Link>      |                 | 0      | 0     | 0      | 0     | 0    |

netstat -i에서 보여 주는 sl0와 sl1 인터페이스는 커널에 두 개의 SLIP 인터페이스가 빌드되어있음을 보여준다(sl0와 sl1 뒤의 \*는 인터페이스가 "다운"되어 있음을 보여준다).

그러나 FreeBSD의 기본 커널은 패킷을 포워드(기본적으로 FreeBSD 머신은 라우터로 동작하지 않는다) 하도록 설정되지 않았기 때문에 인터넷 RFC가 인터넷 호스트에(RFC 1009 [인터넷 게이트웨이에 필요한], 1122 [인터넷 호스트에 필요한 통신 레이어] 그리고 1127[호스트에서 필요한 RFC를 볼 수 있는]를 본다) 필요하다. FreeBSD SLIP 서버가 라우터로 동작하기를 원한다면 /etc/rc.conf 파일을 수정하고 gateway\_enable 변수를 YES로 변경한다.

그리고 재 부팅하여 새로운 설정을 적용해야 된다.

기본 커널 설정 파일(/sys/i386/conf/GENERIC)의 거의 마지막 부분에 있는 다음과 같은 라인을 주의한다:

```
pseudo-device sl 2
```

이것은 커널에서 사용할 수 있는 SLIP 장치 번호를 정의한다; 마지막 라인의 숫자는 운영체제가 동시에 운용할 수 있는 최대 SLIP 연결 숫자이다.

---

커널을 재 설정하는 것은 8 장의 FreeBSD 커널 설정을 참고한다.

## 21.7.2.4 Sliplogin 설정

앞에서 언급했듯이 /usr/sbin/sliplogin(sliplogin의 실제 매뉴얼 페이지는 sliplogin(8)을 본다) 설정의 일부분인 3 개의 파일이 /etc/sliphome 디렉터리에 있다: slip.hosts 는 SLIP 유저와 관련된 IP 주소를 정의한다; slip.login 은 보통 SLIP 인터페이스 설정이다; 그리고(부가적으로) slip.logout 은 시리얼 접속이 종료되면 slip.login 의 설정을 되돌린다.

### [3 개의 파일 설정]

#### 1. slip.hosts 설정

/etc/sliphome/slip.hosts 는 공백으로 나누어진 최소한 4 가지 내용을 포함하는 라인이다:

- SLIP 유저의 로그인 ID
- SLIP 링크의 로컬 주소
- SLIP 링크의 원격 주소
- 네트워크 마스크

로컬과 원격주소는 호스트 이름일 것이고(/etc/hosts 에 의해 IP 주소로 분해되거나, FreeBSD 5.X 에서는 /etc/nsswitch.conf 그리고 FreeBSD 4.X 에서는 /etc/host.conf 의 설정에 따른 도메인 네임 서비스다) 네트워크 마스크는 /etc/networks 검색에 의해 분석할 수 있는 이름일 것이다. 샘플 시스템에서 /etc/sliphome/slip.hosts 는 다음과 같다:

```
#
# login local-addr      remote-addr      mask             opt1      opt2
#                               (normal,compress,noicmp)
#
Shelmerg dc-slip        sl-helmerg       0xffffc00       autocomp
```

각 라인의 마지막에는 다음의 하나 이상의 옵션을 지정할 수 있다.

- 
- *normal* -- 헤더를 압축하지 않는다.
  - *compress* - 헤더를 압축한다.
  - *autocomp* -- 원격에서 허락한다면 헤더를 압축한다.
  - *noicmp* -- ICMP 패킷을 비활성 한다(그래서 어떠한 "ping" 패킷도 여러분의 대역폭을 사용하지 못한다).

여러분의 SLIP 링크에서 로컬과 원격 주소를 선택하는 것은, TCP/IP 서브넷을 사용할 것인가 또는 SLIP 서버에 "proxy ARP"를 사용할 것인지에 따라 달라진다(이것은 "진짜" proxy ARP 가 아니지만 설명을 위해서 이 용어를 사용한다). 어떤 방법을 사용할지 또는 IP 주소를 어떻게 할당할지 정확하지 않다면 SLIP 서버를 설명하기 전에 언급한 내용들을 참고하고 여러분의 네트워크 IP 관리자에게 도움을 청한다.

SLIP 클라이언트를 서브넷으로 나눈다면 여러분에게 할당된 네트워크 IP 번호의 서브넷을 만들고 이 서브넷 내의 IP 주소를 각 SLIP 클라이언트에 할당한다. 그리고 이 SLIP 서브넷으로부터 SLIP 서버를 경유하여 가장 가까운 IP 라우터의 경로를 정적으로 설정해야 될 것이다.

그렇지 않고 "proxy ARP" 방법을 사용한다면 여러분의 SLIP 서버 서브넷 범위의 IP 를 SLIP 클라이언트 IP 주소로 할당한다. 그리고 arp(8)을 사용하여 SLIP 서버의 ARP 테이블에서 proxy-ARP 엔트리를 관리하도록 /etc/sliphome/slip.login 과 /etc/sliphome/slip.logout 스크립트를 수정해야 된다.

## 2. slip.login 설정

전형적인 /etc/sliphome/slip.login 파일은 다음과 같다:

```

#!/bin/sh -
#
#      @(#)slip.login  5.1 (Berkeley) 7/1/90
#
# generic login file for a slip line.  sliplogin invokes this with
# the parameters:
#      1      2      3      4      5      6      7-n
#  slipunit ttyspeed loginname local-addr remote-addr mask opt-args
#
/sbin/ifconfig sl$1 inet $4 $5 netmask $6

```

이 slip.login 파일은 SLIP 인터페이스의 로컬, 원격 주소 그리고 네트워크 마스크를 ifconfig 명령으로 설정한다.

"proxy ARP" 방법(SLIP 클라이언트를 서브넷으로 나누는 대신)을 사용하기로 결정했다면 필요한 /etc/sliphome/slip.login 파일은 다음과 비슷할 것이다:

```

#!/bin/sh -
#
#      @(#)slip.login  5.1 (Berkeley) 7/1/90
#
# generic login file for a slip line.  sliplogin invokes this with
# the parameters:
#      1      2      3      4      5      6      7-n
#  slipunit ttyspeed loginname local-addr remote-addr mask opt-args
#
/sbin/ifconfig sl$1 inet $4 $5 netmask $6
# Answer ARP requests for the SLIP client with our Ethernet addr
/usr/sbin/arp -s $5 00:11:22:33:44:55 pub

```

slip.login 의 추가적인 라인 arp -s \$5 00:11:22:33:44:55 pub 은 SLIP 서버의 ARP 테이블에 ARP 엔트리를 생성한다. 이 ARP 엔트리는 이더넷의 다른 IP 노드에서 SLIP 클라이언트의 IP 주소를 질의할 때 SLIP 서버의 이더넷 맥 주소로 응답하도록 한다.

위의 예제를 사용할 때 이더넷 맥 주소(00:11:22:33:44:55)를 여러분의 시스템의 이더넷 카드 맥 주소로 대체한다. 그렇지 않으면 여러분의 "proxy ARP"는 절대로 동작하지 않는다! netstat -i 를 실행한 결과에서 여러분의 SLIP 서버 이더넷 맥 주소를 확인할 수 있다; 출력의 두 번째 라인이 다음과 비슷할 것이다:

```
ed0 1500 <Link>0.2.c1.28.5f.4a 191923 0 129457 0 116
```

이것은 이 시스템의 이더넷 맥 주소가 00:02:c1:28:5f:4a 임을 보여준다. netstat -i 에서 보여 준 이더넷 맥 주소에서 마침표를 콜론(:)으로 바꾸고 arp(8)이 요구하는 형식으로 주소를 변환하기 위해 하나의 16 진수 숫자에는 0 을 추가한다; 사용법에 대한 완벽한 정보는 arp(8) 매뉴얼 페이지를 본다.

**Note:** /etc/sliphome/slip.login 과 /etc/sliphome/slip.logout 을 생성하고 실행 가능한 비트(chmod 755 /etc/sliphome/slip.login /etc/sliphome/slip.logout)를 설정해야 된다. 그렇지 않으면 sliplogin 은 실행할 수 없다.

### 3. slip.logout 설정

/etc/sliphome/slip.logout 이 꼭 필요하지 않지만("proxy ARP"를 사용하지 않는다면) 생성하기로 했다면 기본 slip.logout 스크립트 예제가 여기 있다:

```
#!/bin/sh -
#
#      slip.logout
#
# logout file for a slip line.  sliplogin invokes this with
# the parameters:
#      1      2      3      4      5      6      7-n
#      slipunit tty speed loginname local-addr remote-addr mask opt-args
#
/sbin/ifconfig sl$1 down
```

"proxy ARP"를 사용한다면 /etc/sliphome/slip.logout 가 SLIP 클라이언트의 ARP 엔트리를 삭제하기를 원할 것이다:



```
#!/bin/sh -
#
#      @(#)slip.logout

#
# logout file for a slip line.  sliplogin invokes this with
# the parameters:
#      1      2      3      4      5      6      7-n
#  slipunit ttyspeed loginname local-addr remote-addr mask opt-args
#
/sbin/ifconfig sl$1 down
# Quit answering ARP requests for the SLIP client
/usr/sbin/arp -d $5
```

arp -d \$5 는 SLIP 클라이언트가 로그인 했을때 slip.login 에 추가된 "proxy ARP" 엔트리를 삭제한다.

/etc/sliphome/slip.logout 을 생성하고 실행할 수 있도록(다시말해서 chmod 755 /etc/sliphome/slip.logout) 다시 설정한다.

### 21.7.2.5 라우팅에서 고려할 사항

SLIP 클라이언트와 나머지 네트워크(아마도 인터넷) 사이의 패킷 라우팅에 "proxy ARP" 방법을 사용하지 않는다면, SLIP 서버를 경유하여 SLIP 클라이언트가 속하는 서브넷까지의 경로를 가장 가까운 기본 라우터에 정적인 경로로 추가해야 될 것이다.

[정적인 경로에서 고려할 사항]

## 1. 정적인 경로

정적인 경로를 가장 가까운 기본 라우터에 추가하는 것이 문제가 될 수 있다(또는 권한이 없다면 불가능하다). Cisco 와 Proteon 에서 만든 몇몇 라우터처럼 여러분의 조직에 다중 라우터로 구성된 네트워크가 있다면 정적인 경로를 SLIP 서브넷에서 사용하도록 라우터를 설정하고 이 정적 경로를 다른 라우터에 알려야 되기 때문에 정적 경로에 기반하는 라우팅(static-route-based routing)이 동작하도록 특정 전문지식과 문제해결/시스템 변경 능력이 필요할 것이다.

## 2. GateD® 실행

정적 경로의 또 다른 문제는 FreeBSD SLIP 서버에 **GateD** 를 설치하고 적절한 라우팅 프로토콜(RIP/OSPF/BGP/EGP)을 사용하여 SLIP 서브넷에 대한 경로 정보를 다른 라우터에 알리도록 설정하는 것이다. gated 를 설정하기 위해 /etc/gated.conf 파일을 작성해야 된다; 여기 필자가 FreeBSD SLIP 서버에 사용하는 것과 비슷한 샘플이 있다:

**Note: GateD®**는 이제 소유권이 있는 소프트웨어이기 때문에 더 이상 공개적으로 소스코드를 이용할 수 없다(더 많은 정보는 GateD 웹 사이트에 있다). 이 섹션은 아직 사용되고 있는 예전 버전과 호환되도록 존재한다

```
#
# gated configuration file for dc.dsu.edu; for gated version 3.5alpha5
# Only broadcast RIP information for xxx.xxx.yy out the ed Ethernet interface
#
#
# tracing options
#
traceoptions "/var/tmp/gated.output" replace size 100k files 2 general ;

rip yes {
    interface sl noripout noripin ;
    interface ed ripin ripout version 1 ;
    traceoptions route ;
};

#
# Turn on a bunch of tracing info for the interface to the kernel:
```

```

kernel {
    traceoptions remnants request routes info interface ;
};

#
# Propagate the route to xxx.xxx.yy out the Ethernet interface via RIP
#

export proto rip interface ed {
    proto direct {
        xxx.xxx.yy mask 255.255.252.0 metric 1; # SLIP connections
    };
};

#
# Accept routes from RIP via ed Ethernet interfaces

import proto rip interface ed {
    all ;
};

```

위의 샘플 `gated.conf` 파일은 SLIP 서브넷 `xxx.xxx.yy` 대한 경로 정보를 RIP 를 사용하여 이더넷에 브로드캐스트 한다; `ed` 드라이버 대신 다른 이더넷 드라이버를 사용한다면 레퍼런스의 `ed` 인터페이스를 적절히 변경해야 된다. 이 샘플 파일은 **GateD** 의 활동을 디버깅하기 위해 `/var/tmp/gated.output` 에 로그로 남기도록 설정되어있다; **GateD** 가 정확히 동작한다면 로그 옵션을 삭제할 수 있다. `xxx.xxx.yy` 를 여러분의 SLIP 서브넷 네트워크 주소로 변경해야 된다(*proto direct* 문자열의 넷 마스크도 변경한다).

시스템에 **GateD** 를 설치하고 설정했다면 FreeBSD 시작 스크립트가 **routed** 대신 **GateD** 를 실행하도록 해야 된다. 가장 쉬운 방법은 `/etc/rc.conf` 에 `router` 와 `router_flags` 변수를 설정한다. **GateD** 의 명령어 매개변수에 대한 정보는 매뉴얼 페이지를 참고한다.

---

## 22 장 전자 메일

### 22.1 개요

email 로 유명한 전자 메일은 현재 가장 광범위하게 사용되는 통신 수단이다. 이번 장에서는 FreeBSD 에서 메일 서버를 운용하는데 필요한 기본 지침과 FreeBSD 로 메일을 보내고 받는 것도 설명한다; 그러나 완벽한 레퍼런스가 아니기 때문에 실제로 중요하게 고려해야 될 많은 사항이 생략되었다. 더 많은 사항은 부록 B 에 나열되어 있는 적절한 책을 참고한다.

이번 장을 읽고 다음과 같은 사항을 알 수 있다:

- 전자 메일을 보내고 받는데 연관된 소프트웨어 컴포넌트는 무엇인가
- 기본 **sendmail** 설정 파일은 FreeBSD의 어느곳에 있는가
- 원격과 로컬 메일 박스의 차이점
- 릴레이로 메일 서버를 불법적으로 사용하는 스팸머는 어떻게 막는가
- 시스템에 **sendmail** 대신 다른 메일 전송 에이전트는 어떻게 설치하고 설정하는가
- 일반적인 메일 서버 문제 해결
- UUCP로 SMTP를 어떻게 사용하는가
- 시스템이 메일을 보낼 수만 있도록 어떻게 설정하는가
- 다이얼-업 접속으로 메일을 어떻게 사용하는가
- 보안에 좀더 안전하도록 SMTP 인증은 어떻게 설정하는가
- 메일을 보내고 받을 수 있도록 **mutt**와 같은 메일 유저 에이전트(Mail User Agent)를 어떻게 설치하고 사용하는가

- 
- 원격 POP이나 IMAP 서버에서 메일을 어떻게 다운로드 하는가
  - 받는 메일에 자동으로 필터와 룰을 적용하는 방법

이번 장을 읽기 전에 다음 사항을 확인해야 된다.:

- 네트워크를 적절히 설정한다(24장).
- 메일 호스트에 맞도록 DNS 정보를 적절히 설정한다(24장).
- 소프트웨어를 어떻게 설치하는지 알아야 된다 (4장)

## 22.2 전자 메일 사용

전자 메일 교환과 관련된 5 가지의 주요 파트가 있다. 이들은 유저 프로그램, 서버 데몬, DNS, 원격 또는 로컬 메일 박스 그리고 메일 호스트다.

### 22.2.1 유저 프로그램

유저 프로그램에는 **mutt**, **pine**, **elm** 와 **mail** 과 같은 명령어 라인 프로그램 그리고 **balsa**, **xfmail** 같은 GUI 프로그램 몇 개와 WWW 브라우저처럼 더욱 복잡한 것이 포함된다. 이들 프로그램은 이용할 수 있는 서버 데몬 중 하나를 호출하거나 TCP 를 통해서 단순히 email 트랜잭션을 로컬 "mailhost"에 전달한다.

### 22.2.2 메일 호스트 서버 데몬

FreeBSD 에는 **sendmail** 이 기본적으로 설치되지만 다음과 같은 데몬을 포함하여 여러 개의 메일 데몬을 지원한다:

- **exim**;
- **postfix**;

---

- **qmail**

서버 데몬은 보통 전달되는 메일을 받고 보내는 메일을 전송하는 두 가지 기능이 있다. 그러나 메일을 읽을 수 있도록 POP 또는 IMAP 과 같은 프로토콜로 메일을 모으는 기능이 없고 로컬 mbox 이나 메일 박스 Maildir 연결도 허용하지 않는다. 이를 위해 추가적인 데몬이 필요할 것이다.

**주의:** 예전 버전의 **sendmail** 에는 공격자가 로컬이나 원격에서 머신에 접근할 수 있는 몇 가지 심각한 보안 문제를 가지고 있다. 이런 문제를 피하기 위해 최신 버전을 사용해야 된다. FreeBSD 포트 컬렉션에서 다른 MTA 를 설치해도 된다.

### 22.2.3 Email 과 DNS

도메인 네임 시스템과(DNS) named 데몬은 메일 전송에 큰 역할을 담당한다. 여러분의 사이트에서 다른 곳으로 메일을 전송하기 위해 서버 데몬은 메일을 받는 목적지 호스트를 결정할 때 DNS 에서 사이트를 검색한다.

DNS 는 호스트 네임을 IP 주소로 매핑하는 기능과 MX 레코드로 알려져 있는 메일 배달에 사용되는 정보를 저장한다. MX(메일 익스체인지) 레코드는 특정 도메인의 메일을 받는 호스트를 지정한다. 호스트 네임이나 도메인의 MX 레코드를 지정하지 않았다면 메일은 A 레코드로 지정된 호스트 IP 로 직접 배달된다.

다음과 같이 host(1) 명령으로 도메인의 MX 레코드를 볼 수 있을 것이다:

```
% host -t mx FreeBSD.org
```

```
FreeBSD.org mail is handled (pri=10) by mx1.FreeBSD.org
```

### 22.2.4 메일 받기

여러분 도메인의 메일은 메일 호스트가 받는다. 여러분의 도메인에 보낸 모든 메일을 모아서 설정에 따라 mbox(메일을 저장하는 기본적인 방법) 또는 Maildir 포맷으로 저장한다. 메일이 저장되면 mail(1)이나 **mutt** 와 같은 어플리케이션을 사용하여 로컬에서 읽거나 POP 이나 IMAP 와 같은 프로토콜로 원격에서 가져간다. 이 의미는 로컬에서만 메일을 읽는다면 POP 또는 IMAP 서버 설치가 필요 없다는 것이다.

---

### 22.2.4.1 POP 과 IMAP 으로 원격 메일 박스 사용

원격에서 메일 박스에 접근하려면 POP 또는 IMAP 서버를 사용해야 된다. 이들 프로토콜은 사용자가 원격에서 메일 박스에 쉽게 연결할 수 있도록 한다. POP 과 IMAP 둘 다 사용자가 원격에서 메일 박스에 접근하도록 하지만 IMAP 이 다음 사항을 포함하여 더 많은 장점을 가지고 있다:

- IMAP은 메시지를 가져오기도 하지만 원격 서버에 저장할 수 있다.
- IMAP은 실시간 업데이트를 지원한다.
- IMAP은 사용자가 메시지를 다운로드 하지 않고 메시지 구조를 가져오기 때문에 느린 속도에서 아주 유용할 수 있다; 클라이언트와 서버간의 데이터 전송을 최소화하기 위해 서버에서 검색할 수 있다.

POP 이나 IMAP 서버를 설치하기 위해 다음 단계를 수행해야 된다:

- ① 조건에 맞는 IMAP이나 POP 서버를 선택한다. 다음 POP과 IMAP 서버는 유명하고 좋은 예제가 될 것이다:
  - `qpopper`;
  - `teapop`;
  - `imap-uw`;
  - `courier-imap`;
- ② 선택한 POP이나 IMAP 데몬을 포트 컬렉션에서 설치한다.
- ③ POP이나 IMAP 서버를 로드 하기 위해 필요한 `/etc/inetd.conf`를 수정한다.

**주의:** POP 과 IMAP 을 사용하여 유저 이름과 패스워드를 포함하는 증명서 정보를 텍스트로 보낼 때 유의한다. 이 의미는 이들 프로토콜로 전송하는 정보가 보안에 안전하기를 원한다면 `ssh(1)`을 통한 터널링 세션 사용을 고려해야 된다. 터널링 세션은 14.12.7 장에서 설명한다.

---

## 22.2.4.2 로컬 메일 박스 사용

메일 박스는 메일 박스가 있는 서버에서 직접 MUA 를 사용하여 로컬에서 사용할 수 있다. 이것은 **mutt** 나 **mail(1)**같은 어플리케이션을 사용하면 된다.

## 22.2.5 메일 호스트

메일 호스트는 여러분 호스트의 메일을 전송하고 받는 기능을 하는 서버에 주어진 이름이다.

## 22.3 sendmail 설정

**sendmail(8)**은 FreeBSD 의 기본 메일 전송 에이전트(MTA)다. **sendmail** 의 기능은 메일 유저 에이전트(MUA)로부터 메일을 받아서 설정 파일에 정의된 적절한 메일러에 전송한다. 그리고 **sendmail** 은 네트워크 연결을 허용하여 메일을 로컬 메일 박스나 다른 프로그램에 배달할 수 있다.

**sendmail** 은 다음 설정 파일을 사용한다:

| 파일이름                      | 기능                                 |
|---------------------------|------------------------------------|
| /etc/mail/access          | <b>sendmail</b> 접근 데이터베이스 파일       |
| /etc/mail/aliases         | 메일 박스 옐리어스(alias)                  |
| /etc/mail/local-host-name | <b>sendmail</b> 이 메일을 허용하는 호스트 리스트 |
| /etc/mail/mailer.conf     | 메일러 프로그램 설정                        |
| /etc/mail/mailertable     | 메일러 배달 테이블                         |
| /etc/mail/sendmail.cf     | <b>sendmail</b> 마스터 설정 파일          |
| /etc/mail/virtusertable   | 가상 유저와 도메인 테이블                     |

### 22.3.1 /etc/mail/access

접근 데이터베이스는 어떤 호스트 또는 IP 주소가 로컬 메일 서버에 접근하고 어떤 종류의 접근을 할 수 있는지 정의한다. 호스트를 *OK*, *REJECT*, *RELAY*로 나열하거나 단순히 주어진 메일러 에러로 에러를 제어하는 **sendmail** 의 에러 제어 루틴으로 보낼 수 있다. 기본값인 *OK*로 지정된 호스트는 메일의 최종 목적지가 로컬 머신이면 메일을 이 호스트에



---

보낼 수 있다. *REJECT*로 지정된 호스트는 모든 메일 연결을 거부당한다. *RELAY* 옵션을 가진 호스트는 이 메일 서버를 통해 어떤 목적지라도 메일을 보낼 수 있다.

#### 예제 22-1. sendmail 접근 데이터베이스 설정

|                        |                                        |
|------------------------|----------------------------------------|
| cyberspammer.com       | 550 We don't accept mail from spammers |
| FREE.STEALTH.MAILER@   | 550 We don't accept mail from spammers |
| another.source.of.spam | REJECT                                 |
| okay.cyberspammer.com  | OK                                     |
| 128.32                 | RELAY                                  |

이 예제에는 5 개의 엔트리가 있다. 테이블의 왼쪽에서 매칭되는 메일 발신자는 테이블 오른쪽 동작의 영향을 받는다. 처음 두 개의 예제는 에러코드를 **sendmail**의 에러 제어 루틴으로 전달한다. 메일이 테이블의 왼쪽과 매칭될 때 메시지가 원격 호스트에 출력된다. 다음 엔트리는 인터넷의 지정된 호스트(another.source.of.spam)로부터 메일을 거부한다. 다음 엔트리는 위의 cyberspammer.com 보다 자세하게 호스트 okay.cyberspammer.com 의 메일 연결을 허용한다. 더 정확히 매칭되면 모호하게 매칭되는 것을 무시한다. 마지막 엔트리는 128.32.로 시작되는 IP 주소에 포함된 호스트의 전자 메일 릴레이를 허용한다. 이들 호스트는 이 메일 서버를 통해 목적지인 다른 메일 서버로 메일을 보낼 수 있다.

이 파일을 변경하면 데이터베이스를 업데이트하도록 /etc/mail/에서 make 를 실행해야 된다.

### 22.3.2 /etc/mail/aliases

엘리어스 데이터베이스는 다른 유저, 파일, 프로그램 또는 다른 엘리어스로 확장되는 가상 메일 박스 리스트를 가지고 있다. /etc/mail/aliases 에 사용할 수 있는 몇 개의 예제가 있다.

#### 예제 22-2. 메일 엘리어스

```
root: localuser
ftp-bugs: joe,eric,paul
bit.bucket: /dev/null
procmail: "|/usr/local/bin/procmail"
```

파일 포맷은 간단하다; 콜론(:)왼쪽의 메일 박스 이름은 오른쪽의 타겟으로 배달된다. 첫 번째 예제는 단순히 root 메일 박스를 엘리어스 데이터베이스를 다시 검색하는 localuser

---

메일 박스에 배달한다. 매칭되는 것이 없다면 메시지는 로컬 유저 `localuser`에게 배달된다. 다음 예제는 메일 리스트를 보여 준다. 메일 박스 `ftp-bugs`로 가는 메일은 3개의 로컬 메일 박스 `joe`, `eric`과 `paul`에게 배달된다. 원격 메일 박스는 [user@example.com](mailto:user@example.com)과 같이 지정할 수 있다. 그 다음 예제는 메일을 파일에 전달하는데, 여기서는 `/dev/null`로 보내는 것을 보여 준다. 마지막 예제는 메일을 프로그램에게 보내는 것을 보여 준다. 이 경우 메일 메시지는 유닉스 파이프를 거쳐 `/usr/local/bin/procmail`의 표준 입력으로 작성된다.

이 파일을 변경하면 데이터베이스를 업데이트하기 위해 `/etc/mail/`에서 `make`를 실행해야 된다.

### 22.3.3 /etc/mail/local-host-names

이 파일은 `sendmail(8)`이 로컬 호스트 네임처럼 허용하는 호스트 네임 리스트이다. `sendmail`이 메일을 받아야 되는 도메인이나 호스트를 적는다. 예를 들면 이 메일 서버가 도메인 `example.com`과 호스트 `mail.example.com`의 메일을 허용해야 된다면 `local-host-names`는 다음과 비슷할 것이다:

```
example.com
mail.example.com
```

이 파일을 변경하면 변경 사항을 읽도록 `sendmail(8)`을 다시 시작해야 된다.

### 22.3.4 /etc/mail/sendmail.cf

`sendmail`의 마스터 설정 파일 `sendmail.cf`는 메일 주소를 다시 작성하는 것부터 원격 메일 서버에 거부 메시지를 보내는 것을 포함한 `sendmail`의 모든 동작을 제어한다. 매우 다양한 기능으로 이 설정 파일은 약간 복잡하기 때문에 자세한 사항은 이 섹션의 범위를 초과한다. 다행히 표준 메일 서버를 구성하는 경우 이 파일을 약간만 수정하면 된다.

마스터 `sendmail` 설정 파일은 `sendmail`의 특성과 동작을 정의하는 `m4(1)` 매크로에서 빌드할 수 있다. 더 자세한 사항은 `/usr/src/contrib/sendmail/cf/README`를 확인한다.

이 파일을 변경하고 변경된 것을 적용하기 위해 `sendmail`을 재 시작해야 된다.

---

## 22.3.5 /etc/mail/virtusertable

virtusertable 은 가상 도메인과 메일 박스를 실제 메일 박스로 매핑하는 메일 주소다. 이들 메일 박스에는 로컬, 원격 그리고 /etc/mail/aliases 에 정의된 엘리어스나 파일을 사용할 수 있다.

### 예제 22-3. 가상 도메인 메일 매핑 표 예제

|                        |                            |
|------------------------|----------------------------|
| root@example.com       | root                       |
| postmaster@example.com | postmaster@noc.example.net |
| @example.com           | joe                        |

위의 예제는 도메인 example.com 과 매핑 된다. 이 파일은 처음부터 순서대로 매칭(매칭되는 룰이 여러 개라면 최초에 매칭되는 룰이 적용된다)된다. 첫 번째 아이템은 *root@example.com* 을 로컬 메일 박스 root 로 매핑 한다. 다음 엔트리는 *postmaster@example.com* 을 noc.example.net 의 호스트 postmaster 로 매핑 한다. 마지막으로 더 이상 example.com 과 매칭되는 것이 없다면 다른 모든 메일 메시지가 매칭되는 마지막 매핑으로 매칭된다. 이것은 로컬 메일 박스 joe 와 맵핑 된다.

## 22.4 메일 전송 에이전트 변경

앞에서 언급했듯이 FreeBSD 는 MTA 로(메일 전송 에이전트) **sendmail** 이 이미 설치되어 있다. 그래서 기본적으로 **sendmail** 이 매일 보내기와 받기를 담당하고 있다.

그러나 여러 가지 이유로 어떤 시스템 관리자는 시스템의 MTA 를 변경하기를 원한다. 이런 이유는 단순히 다른 MTA 를 사용해 보고 싶은 것부터 특정 기능이 필요하거나 다른 메일러에 있는 패키지를 원하기 때문이다. 이유야 어떻든 다행히 FreeBSD 는 MTA 를 쉽게 변경할 수 있다.

### 22.4.1 새로운 MTA 설치

다양하게 MTA 를 선택할 수 있다. 여러 가지 MTA 를 선택할 수 있는 FreeBSD 포트 컬렉션에서부터 찾기 시작하는 것이 좋을 것이다. 물론 어느 곳에서든 원하는 MTA 를

---

선택하여 FreeBSD 에 설치해서 사용할 수 있다.

새로운 MTA 를 설치하는 것으로 시작한다. 설치 한 후 **sendmail** 의 역할을 인수하기 전에 새로 설치한 MTA 가 원하는 기능을 제공하는지 확인하고 새로운 소프트웨어를 설정할 수 있는 기회가 주어진다. 새로운 소프트웨어를 설치할 때 /usr/bin/sendmail 과 같은 시스템 바이너리를 덮어쓰지 말아야 된다. 그렇지 않으면 새로운 메일 소프트웨어가 설정하기 전에 서비스하게 된다.

선택한 소프트웨어 설정 방법에 대한 정보는 선택한 MTA 의 문서를 참고한다.

## 22.4.2 sendmail 비활성

**sendmail** 을 시작하는데 사용하는 프로시저는 4.5-릴리즈와 4.6-릴리즈에서 약간 변경되었다. 따라서 비활성 하는 프로시저도 약간 다르다.

### 22.4.2.1 2002/4/4 이전의 FreeBSD 4.5-STABLE 과 이전 버전

#### (4.5-릴리즈와 이전 버전 포함)

/etc/rc.conf 에 다음 내용을 입력한다.

```
sendmail_enable="NO"
```

이것은 **sendmail** 의 메일 수신 서비스는 비활성 하지만 /etc/mail/mailer.conf 가(아래를 본다) 변경되지 않았다면 e-mail 을 보내기 위해 **sendmail** 이 계속 사용된다.

### 22.4.2.2 2002/4/4 이후의 FreeBSD 4.5-STABLE (4.6-릴리즈와

#### 이후 버전 포함)

완벽하게 **sendmail** 을 비활성 하려면 다음 내용을 /etc/rc.conf 에 사용한다.

---

```
sendmail_enable="NONE"
```

**주의:** `sendmail` 의 메일 전송 서비스를 비활성 했다면 대체 메일 배달 시스템을 사용해야 된다. 그렇지 않았다면 `periodic(8)`과 같은 시스템의 기능은 메일로 결과를 배달할 수 없다. 시스템의 수 많은 부분은 `sendmail` 호환 시스템 기능을 가지고 있다. `sendmail` 을 비활성 한 후 어플리케이션이 계속해서 `sendmail` 의 바이너리를 사용하여 메일을 보내려고 한다면 메일은 동작하지 않는 `sendmail` 큐에 쌓이게 되고 절대 배달되지 않는다.

`sendmail` 의 받는 메일 서비스만 중지하려면 다음과 같이 `/etc/rc.conf` 에 설정한다.

```
sendmail_enable="NO"
```

더 많은 `sendmail` 의 시작 옵션은 `rc.sendmail(8)` 매뉴얼 페이지에서 확인할 수 있다.

### 22.4.3. 부팅할 때 새로운 MTA 실행

부팅할 때 새로운 MTA 를 실행하기 위해 두 가지 방법을 선택할 수 있고 운용하는 FreeBSD 버전에 따라 선택할 수 있다.

#### 22.4.3.1 2002/4/11 일 이전의 FreeBSD 4.5-STABLE(4.5-

##### 릴리즈와 이전 버전 포함).

`/usr/local/etc/rc.d/`에 root 로 실행할 수 있는 `.sh` 로 끝나는 스크립트를 추가한다. 이 스크립트는 `start`와 `stop` 매개변수로 실행할 수 있어야 된다. 수동으로 서버를 시작 할 때도 사용할 수 있는 이 시스템 스크립트는 시스템이 시작될 때 다음 명령을 실행한다.

```
# /usr/local/etc/rc.d/supermailer.sh start
```

서버를 수동으로 정지시킬 때 사용할 수 있는 이 시스템 스크립트는 첫 다운할 때 명령에 `stop` 옵션을 사용한다.

```
# /usr/local/etc/rc.d/supermailer.sh stop
```

---

## 22.4.3.2 2002/4/11 이후 FreeBSD 4.5-STABLE (4.6-릴리즈와 그 후 버전을 포함하는)

FreeBSD 의 이후 버전에서 위의 방법을 사용할 수 있거나 `/etc/rc.conf` 에 `mta_start_script="filename"`을 설정할 수 있다. `filename`은 부팅할 때 MTA 를 시작하기 위해 실행하는 스크립트 이름이다.

## 22.4.4 sendmail 을 대신하는 시스템의 기본 메일러 설정

`sendmail` 프로그램은 유닉스 시스템에 표준 소프트웨어로 되어 있기 때문에 어떤 소프트웨어는 이미 `sendmail` 이 설치되고 설정되어 있다고 가정한다. 이런 이유로 다른 많은 MTA 는 `sendmail` 명령어 라인 인터페이스와 호환되는 자신만의 기능을 제공한다; 이것은 `sendmail` 교체하는데 용이하게 사용된다.

그래서 다른 메일러를 사용한다면 `/usr/bin/sendmail` 과 같은 표준 `sendmail` 바이너리를 실행하려는 소프트웨어가 실제로 여러분이 선택한 메일러를 실행하도록 해야 된다. 다행히 FreeBSD 는 이런 기능을 하는 `mailwrapper(8)`라는 시스템을 제공한다.

`sendmail` 이 설치되어 운용 중일 때 `/etc/mail/mailler.conf` 에서 다음과 같은 내용을 찾을 수 있다:

|                         |                                             |
|-------------------------|---------------------------------------------|
| <code>sendmail</code>   | <code>/usr/libexec/sendmail/sendmail</code> |
| <code>send-mail</code>  | <code>/usr/libexec/sendmail/sendmail</code> |
| <code>mailq</code>      | <code>/usr/libexec/sendmail/sendmail</code> |
| <code>newaliases</code> | <code>/usr/libexec/sendmail/sendmail</code> |
| <code>hoststat</code>   | <code>/usr/libexec/sendmail/sendmail</code> |
| <code>purgestat</code>  | <code>/usr/libexec/sendmail/sendmail</code> |

이 의미는 이들 일반적인 명령(`sendmail` 같은)이 실행될 때 시스템은 실제로 `mailler.conf` 를 체크해서 `/usr/libexec/sendmail/sendmail` 을 대신 실행하는 `sendmail` 이라는 이름의 `mailwrapper` 복사본을 호출한다. 이 시스템은 이런 기본 `sendmail` 기능이 수행될 때 어떤 바이너리를 실제로 실행할지 변경하기 쉽게 한다.

그래서 `sendmail` 대신 `/usr/local/supermailer/bin/sendmail-compat` 를 실행하려면

---

/etc/mail/mailer.conf 를 변경할 수 있다:

```
sendmail    /usr/local/supermailer/bin/sendmail-compat
send-mail   /usr/local/supermailer/bin/sendmail-compat
mailq       /usr/local/supermailer/bin/mailq-compat
newaliases  /usr/local/supermailer/bin/newaliases-compat
hoststat    /usr/local/supermailer/bin/hoststat-compat
purgestat   /usr/local/supermailer/bin/purgestat-compat
```

## 22.4.5 마무리

원하는 모든 설정을 끝냈다면 더 이상 필요 없는 **sendmail** 프로세스를 죽이고 새로운 소프트웨어 프로세스를 시작하거나 간단히 재 부팅한다. 재 부팅은 시스템이 부팅할 때 새로운 MTA 를 자동으로 실행하도록 정확하게 설정되어 있는지 확인할 수 있는 기회를 제공한다.

## 22.5 문제 해결

### 22.5.1 사이트의 hosts 에 왜 FQDN 를 사용해야 되는가?

호스트가 실제로 도메인과 다르다는 것을 알게 될 것이다; 예를 들어 여러분이 foo.bar.edu 에 있고 bar.edu 도메인의 mumble 라는 호스트에 메일을 보내려면 mumble 대신 전체 도메인 이름 mumble.bar.edu 를 사용해야 된다.

전통적으로 이 방법은 BSD BIND 리졸버(resolver)에서 허용되었다. 그러나 FreeBSD 에 적용된 현재 버전의 **BIND** 는 완벽한 도메인 네임이 아닌 기본 도메인의 축약된 이름을 더 이상 허용하지 않는다. 그래서 부적절한 호스트 이름 mumble 는 mumble.foo.bar.edu 에서 찾거나 root 도메인에서 검색된다.

mumble.bar.edu 와 mumble.edu 는 이전과 다른 검색 방법으로 찾아진다. 이 방법이 왜 나쁜지 그리고 보안 문제가 발생할 수 있는지 RFC 1535 를 찾아본다.

이 문제를 해결하는 쉬운 방법은 /etc/resolv.conf 에서 다음 라인을 아래 라인처럼 완벽한

---

도메인 네임으로 바꾼다.

`domain foo.bar.edu`

위의 내용을 아래와 같이 완벽한 도메인 이름으로 바꾼다.

`search foo.bar.edu bar.edu`

그러나 RFC 1535 에서 명시하듯이 검색 순서가 “local 과 public 관리 사이의 경계”를 넘어가지 않게 한다.

## 22.5.2 sendmail 이 메시지 “mail loops back to myself” 를 보여 준다.

이 문제는 **sendmail** FAQ 에 다음과 같이 답변되었다:

아래와 같은 예러 메시지를 받았다.

553 MX list for domain.net points back to relay.domain.net

554 <user@domain.net>... Local configuration error

이 문제를 어떻게 해결할 수 있는가?

MX 레코드를 통해 지정된 호스트(여기서는 relay.domain.net)로 메일을 전달하도록 요청했지만 릴레이 머신이 domain.net 에 대해 알지 못한다. domain.net 을 /etc/mail/local-host-names 에 추가하거나(8.10 이전 버전에서는 /etc/sendmail.cw 로 알려져 있다) “Cw domain.net”(FEATURE(use\_cw\_file)를 사용한다면)을 /etc/mail/sendmail.cf 에 추가한다.

**sendmail** FAQ 는 <http://www.sendmail.org/faq/>에서 찾을 수 있고 메일 설정을 변경하려면 FAQ 를 읽어 본다.

## 22.5.3 다이얼-업 PPP 호스트에서 메일 서버를 어떻게 운용하는가?



---

LAN의 FreeBSD 박스를 인터넷에 연결하기를 원한다. FreeBSD 박스는 LAN의 메일 게이트웨이가 된다. PPP 전용선이 아니다.

최소한 두 가지 방법이 있고 하나는 UUCP를 이용하는 것이다.

다른 하나는 도메인의 2차 MX 서비스를 제공하는 Full-time(계속 운용 중인) 인터넷 서버가 있으면 된다. 예를 들어 회사의 도메인이 example.com 이고 인터넷 공급자가 도메인의 2차 MX 서비스를 제공하도록 example.net 을 설정했다면 하나의 호스트만 최종 수령자로 지정한다(example.com 의 /etc/mail/sendmail.cf 에 *Cw example.com* 를 추가한다):

|              |    |    |              |
|--------------|----|----|--------------|
| example.com. | MX | 10 | example.com. |
|              | MX | 20 | example.net. |

**sendmail** 이 메일을 보낼 때는 모뎀 링크로 여러분에게(example.com) 메일을 배달하려고 한다. 이 경우 여러분이 온라인에 없기 때문에 대부분 타임 아웃에 걸린다. **sendmail** 프로그램은 자동으로 2차 MX 사이트로 메일을 배달한다. 예를 들어 인터넷 서비스 공급자(example.net)의 메일 서버에 배달한다. 2차 MX 사이트는 1차 MX 호스트(example.com)로 메일을 배달하려고 주기적으로 여러분의 호스트에 연결을 시도한다.

로그인 스크립트에 다음과 같이 사용할 수 있다:

```
#!/bin/sh
# Put me in /usr/local/bin/pppmyisp
( sleep 60 ; /usr/sbin/sendmail -q ) &
/usr/sbin/ppp -direct pppmyisp
```

유저 별로 로그인 스크립트를 따로 작성하려면 위 스크립트 대신 `sendmail -qRexample.com` 를 사용할 수 있다. 이 방법은 example.com 의 큐에 있는 모든 메일을 즉시 처리하게 한다.

좀더 자세한 내용은 다음과 같다:

FreeBSD 인터넷 공급자의 메일링 리스트에서 가져온 메시지.

---

> 우리는 고객에게 2 차 MX 를 제공한다. 고객은 메일을 1 차 MX 로 받아 가기 위해  
> 하루에 몇 번씩 자동으로 우리 서비스에 접속한다. 우리 샌드메일은 메일 큐에 있는  
> 메일을 30 분마다 전송한다. 이때 모든 메일을 1 차 MX 에 보내기 위해 고객은 온라인  
> 상태를 30 분간 유지해야 된다.  
>  
> 샌드메일이 모든 메일을 지금 보내도록 하는 명령이 있는가? 물론 유저는 머신에 root  
> 권한이 없다.

sendmail.cf 의 “privacy flags” 섹션에 Opgoaway,restrictqrun 에 대한 정의가 있다.  
root 가 아닌 일반 유저가 메일 큐 프로세스를 시작할 수 있도록 허용하려면 restrictqrun  
삭제한다. 그리고 MX 레코드를 다시 정렬해야 될 것이다. 우리는 고객의 사이트에 대한  
제일 높은 우선순위의 MX 이므로 다음과 같이 정의했다.

```
# 우리가 호스트의 최적의 MX 라면 로컬 설정 에러를 만들지 않고 직접 시도한다.  
#  
OwTrue
```

이 방법으로 원격 사이트는 고개의 접속 없이 여러분에게 메일을 전달한다. 그리고  
여러분의 고객에게 전달한다. 호스트로서 고객 메일 서버의 DNS 에 “customer.com”과  
“hostname.customer.com”을 추가한다. 그리고 DNS 의 A 레코드에 “customer.com”을  
입력한다.

## 22.5.4 다른 호스트에 메일을 보냈을 때 “Relaying

### Denied” 에러는 왜 발생하는가?

FreeBSD 가 기본값으로 설치된 상태에서 **sendmail** 은 **sendmail** 이 운용 중인 호스트의  
메일만 보낼 수 있도록 설정되어 있다. 예를 들어 POP3 서버가 설치되어 있다면 유저는  
학교, 직장 또는 다른 원격지에서 메일을 체크할 수 있지만 외부에서 메일을 보낼 수 없다.  
일반적으로 메일 보내기를 한 후 메일러 데몬으로부터 “5.7 Relaying Denied” 에러  
메시지를 받게 된다.

이것을 해결하는 여러 가지 방법이 있다. 가장 직관적인 해결책은 릴레이 도메인 파일  
/etc/mail/relay-domains 에 여러분 ISP 의 IP 주소를 넣는 것이다. IP 를 추가하는 빠른

---

방법은 다음 명령을 실행한다:

```
# echo "your.isp.example.com" > /etc/mail/relay-domains
```

이 파일을 생성하거나 수정한 후 **sendmail** 를 재 시작해야 된다. 여러분이 서버 관리자이고 로컬에서 메일 보내기를 원치 않거나 다른 머신 또는 다른 ISP 의 클라이언트/시스템을 지정하여 사용하려면 엄청난 작업이 된다. 한 두 개의 메일 계정을 설정한다면 매우 유용하다. 추가해야 될 많은 주소가 있다면 좋아하는 에디터로 이 파일을 열고 라인당 하나씩 도메인을 추가한다.

```
your.isp.example.com
other.isp.example.net
users-isp.example.org
www.example.org
```

이제 리스트에 있는 호스트(시스템의 유저가 가지고 있는 계정을 지원하는)에서 여러분의 시스템을 통해 메일 보내기가 성공한다. SPAM 을 보내려는 사람들을 제외하고 원격 유저가 시스템을 통해 메일을 보낼 수 있는 매우 좋은 방법이다.

## 22.6 고급 주제

이번 섹션은 전체 도메인의 메일 설정과 구축처럼 더욱 복잡한 주제에 대해 다룬다.

### 22.6.1 기본 설정

/etc/resolv.conf 를 설정하거나 네임 서버를 직접 운용해서 외부 호스트로 메일을 보낼 수 있다. 메일이 여러분의 FreeBSD 호스트에서 운용 중인 MTA(예: **sendmail**)로 배달되기를 원한다면 두 가지 방법이 있다:

- 네임 서버를 직접 운용하고 도메인을 갖는다. 예를 들면 FreeBSD.org
- 호스트에 직접 배달된 메일을 본다. 이것은 여러분 머신의 현재 DNS 네임에 직접 메일을 배달하면 된다.

---

위 사항 중 어떤 것을 선택하든지 호스트에 직접 배달되는 메일을 받으려면 영구적인 고정 IP 주소를 가지고 있어야 된다(대부분의 PPP 다이얼-업 설정처럼 유동적인 주소가 아닌). 방화벽 안에 있다면 SMTP 트래픽을 전송할 수 있어야 한다. 호스트에서 메일을 직접 받으려면 다음 두 가지 중 하나가 필요하다:

- DNS에 있는 MX 레코드에 호스트 IP 주소를 지정해야 된다.
- 호스트를 위한 MX 엔트리가 DNS에 없어야 한다.

위의 두 가지 중 한가지로 호스트에서 메일을 직접 받을 수 있다.

다음과 같이 실행한다:

```
# hostname
example.FreeBSD.org
# host example.FreeBSD.org
example.FreeBSD.org has address 204.216.27.XX
```

머신이 위의 메시지만 출력했다면 <[yourlogin@example.FreeBSD.org](mailto:yourlogin@example.FreeBSD.org)> 메일은 문제 없이 동작하는 것을 볼 수 있다(sendmail이 example.FreeBSD.org에서 정상적으로 운용되고 있다고 가정한다).

대신 다음과 같은 메시지를 보게 된다면 호스트에(example.FreeBSD.org) 보낸 모든 메일은 호스트에 직접 보내 지지 않고 같은 유저 이름의 hub 에 모아 지게 된다:

```
# host example.FreeBSD.org
example.FreeBSD.org has address 204.216.27.XX
example.FreeBSD.org mail is handled (pri=10) by hub.FreeBSD.org
```

위의 정보는 DNS 서버로 제어된다. 메일 라우팅 정보를 전송하는 DNS 레코드는 *Mail eXchange* 엔트리다. MX 레코드가 없다면 메일은 IP 주소에 따라 직접 목적지 호스트에 배달된다.

freefall.FreeBSD.org 의 MX 엔트리는 다음과 비슷할 것이다:

|          |    |    |                      |
|----------|----|----|----------------------|
| freefall | MX | 30 | mail.crl.net         |
| freefall | MX | 40 | agora.rdrop.com      |
| freefall | MX | 10 | freefall.FreeBSD.org |
| freefall | MX | 20 | who.cdrom.com        |

freefall 은 많은 MX 엔트리를 가지고 있다. 가장 낮은 MX 번호는 가능하다면 메일을 직접 받는 호스트다. 어떤 이유로 이 호스트에 접근할 수 없다면 다른 호스트(종종 "backup MX"라고 하는)가 메시지를 일시적으로 받고 낮은 번호의 호스트를 이용할 수 있을 때 낮은 번호 호스트로 전송해서 결국 가장 낮은 번호의 호스트로 전송된다.

다른 MX 사이트는 유용하게 사용할 수 있도록 여러분의 인터넷 회선과 분리되어 있어야 된다. 여러분의 ISP 또는 다른 친근한 사이트에서 이 서비스를 제공한다면 문제 없다.

## 22.6.2 도메인용 메일

"메일 호스트"(별칭: 메일 서버)를 설정하려면 여러 대의 워크스테이션으로 보내는 모든 메일을 이 서버에서 받도록 한다. 기본적으로 여러분 도메인에 있는 호스트(여기서는 \*.FreeBSD.org)로 전송되는 메일을 메일 서버에 넘겨주면 유저는 마스터 메일 서버에서 메일을 받을 수 있다.

사용하기 편하도록 *username* 과 같은 유저 계정이 양쪽 머신에 있어야 한다. adduser(8)을 사용한다.

사용할 메일 호스트는 네트워크에 있는 각 워크스테이션의 메일 익스체인지로 디자인되어 있어야 된다. 다음과 같이 DNS 를 설정하면 된다:

|                     |    |               |                            |
|---------------------|----|---------------|----------------------------|
| example.FreeBSD.org | A  | 204.216.27.XX | ; Workstation              |
|                     | MX | 10            | hub.FreeBSD.org ; Mailhost |

이 설정은 A 레코드가 어느 곳을 지정하던지 상관없이 워크스테이션의 메일을 메일 호스트로 리다이렉트 한다. 메일은 MX 호스트로 보내 진다.

직접 DNS 서버를 운용하지 않는다면 이 방법을 사용할 수 없다. DNS 서버를 운용하지 않거나 DNS 서버를 운용할 수 없다면 DNS 를 제공하는 ISP 에 문의한다.

---

가상 메일 호스팅을 하고 있다면 다음 정보가 유용하다. 이 예제에서 여러분은 customer1.org 라는 도메인이 있는 고객이 있다고 가정하고 customer1.org 의 모든 메일을 여러분의 메일 호스트 mail.myhost.com 에 보내려고 한다. 그렇다면 DNS 엔트리는 다음과 비슷하다:

|               |    |    |                 |
|---------------|----|----|-----------------|
| customer1.org | MX | 10 | mail.myhost.com |
|---------------|----|----|-----------------|

이 도메인의 메일만 제어한다면 customer1.org 의 A 레코드는 필요 없다.

**Note:** A 레코드가 없으면 customer1.org 에 대한 ping 은 가지 않는다.

마지막으로 해야 될 내용은 어떤 도메인 또는 호스트 네임의 메일을 **sendmail** 이 허용해야 되는지 지정해야 된다. 다음 중 한가지를 사용하면 된다:

- FEATURE(use\_cw\_file)을 사용한다면 /etc/mail/local-host-names 파일에 호스트를 추가한다. 8.10 버전 보다 이전 **sendmail**을 사용한다면 이 파일은 /etc/sendmail.cw이다.
- 8.10 또는 이후 버전의 **sendmail**을 사용한다면 /etc/sendmail.cf 또는 /etc/mail/sendmail.cf 파일에 *Cwyour.host.com* 라인을 추가한다.

## 22.7 UUCP 로 SMTP 사용

FreeBSD 에서 실행되는 **sendmail** 은 인터넷에 직접 연결되어 있는 사이트에 맞게 디자인되어 있다. UUCP(UNIX-to-UNIX Copy Protocol )로 메일을 보내고 받으려는 사이트는 다른 **sendmail** 설정 파일을 설치해야 된다.

/etc/mail/sendmail.cf 를 직접 수정하는 것은 난이도가 상당히 높다. **sendmail** 버전 8 은 실제 상위 레벨에서 설정하는 m4(1) 전처리 프로세스로 설정 파일을 생성한다. m4(1) 설정 파일은 /usr/src/usr.sbin/sendmail/cf 에서 찾을 수 있다.

소스로 설치하지 않았다면 **sendmail** 설정 세트는 여러 개의 타볼로 배포된다. 마운트 된 CDROM 에 FreeBSD 소스 코드가 있다고 가정하면 다음과 같이 실행한다:

---

```
# cd /cdrom/src
```

```
# cat scontrib.?? | tar xzf --C /usr/src/contrib/sendmail
```

이 명령은 몇 백 킬로바이트만 추출한다. cf 디렉터리의 README 파일에서 m4 설정을 위한 기본적인 정보를 찾을 수 있다.

UUCP 배달을 지원하는 가장 좋은 방법은 *mailertable* 기능을 사용하는 것이다. 이것은 **sendmail** 이 라우팅(경로)을 결정하는데 사용할 수 있는 데이터베이스를 생성한다.

우선 .mc 파일을 생성한다. 디렉터리 /usr/src/usr.sbin/sendmail/cf/cf 에 몇 개의 예제가 있다. foo.mc 라는 이름의 파일을 생성했다면 적절한 sendmail.cf 로 변환하기 위해 필요한 명령은 다음과 같다:

```
# cd /usr/src/usr.sbin/sendmail/cf/cf
```

```
# make foo.cf
```

```
# cp foo.cf /etc/mail/sendmail.cf
```

일반적인 .mc 파일은 다음과 같다:

```
VERSIONID('Your version number') OSTYPE(bsd4.4)

FEATURE(accept_unresolvable_domains)
FEATURE(nocanonify)
FEATURE(mailertable, `hash -o /etc/mail/mailertable')

define(`UUCP_RELAY', your.uucp.relay)
define(`UUCP_MAX_SIZE', 200000)
define(`confDONT_PROBE_INTERFACES')

MAILER(local)
MAILER(smtp)
MAILER(uucp)

Cw    your.alias.host.name
Cw    youruucpnodename.UUCP
```

---

*accept\_unresolvable\_domains*, *nocanonify*와 *confDONT\_PROBE\_INTERFACES* 라인이 메일을 배달하는 동안 DNS 를 사용하는 것을 방지한다. *UUCP\_RELAY* 절은 UUCP 배달을 지원하는데 필요하다. 단순히 제어하려는 .UUCP 가상 도메인 주소를 *UUCP\_RELAY* 절에 할당한다; 보통 여러분 ISP 의 메일 릴레이를 넣는다.

이렇게 설정 했다면 /etc/mail/mailertable 파일이 필요하다. 여러분의 모든 메일에 사용되는 외부 링크가 하나 있다면 다음 파일이 적당하다:

```
#
# makemap hash /etc/mail/mailertable.db < /etc/mail/mailertable
.                uucp-dom:your.uucp.relay
```

더 복잡한 예제는 다음과 같을 것이다:

```
#
# makemap hash /etc/mail/mailertable.db < /etc/mail/mailertable
#
horus.interface-business.de    uucp-dom:horus
.interface-business.de        uucp-dom:if-bus
interface-business.de          uucp-dom:if-bus
.heep.sax.de                   smtp8:%1
horus.UUCP                     uucp-dom:horus
if-bus.UUCP                    uucp-dom:if-bus
.                               uucp-dom:
```

처음 3 라인은 도메인으로 향하게 된 메일을 기본 라우트로 보내지 않고 배달 경로가 짧아지도록 근처 UUCP 로 보낸다. 그 다음 라인은 SMTP 로 배달할 수 있는 로컬 이더넷 도메인의 메일을 제어한다. 마지막으로 근처의 UUCP 는 *uucp-neighbor !recipient* 가 기본 룰을 무시하도록 .UUCP 가상 도메인으로 표시된다. 마지막 라인은 항상 모든 것이 매칭되는 하나의 점(.)이다. 이것은 외부로 나가는 일반적인 메일의 게이트웨이를 제공하는 근처 UUCP 호스트로 메일을 배달한다. *uucp-dom:* 키워드 뒤의 모든 노드 이름은 *uname* 명령으로 확인할 수 있는 적절한 UUCP 인접 호스트여야 된다.

이 파일을 사용하기 전에 DBM 데이터베이스로 변환해야 된다. 이 명령어 라인은 mailertable 파일의 최 상단에 표시되어 있다. mailertable 파일을 변경할 때마다 항상 이 명령을 실행해야 된다.



---

마지막 힌트: 특정 메일 라우팅이 동작하는지 확실하지 않다면 **sendmail** 에 **-bt** 옵션을 사용한다. 이것은 **sendmail** 을 주소 테스트 모드로 시작한다; 단순히 메일 라우팅을 테스트하려는 주소를 3,0 뒤에 입력한다. 마지막 라인은 이 에이전트의 목적지 호스트를 호출하는데 사용되는 내부 메일 에이전트와 주소(변환된 주소)를 알려 준다. 이 모드는 **Ctrl+D** 를 입력하여 종료한다.

% **sendmail -bt**

ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)

Enter <ruleset> <address>

> **3,0 foo@example.com**

canonicalize           input: foo @ example . com

...

parse                   returns: \$# uucp-dom \$#@ your.uucp.relay \$: foo < @ example . com .

>

> ^D

## 22.8 보내기 전용으로 설정

릴레이를 통해 메일만 보내기를 원하는 여러 가지 이유가 있다. 몇 가지는 다음과 같다:

- 여러분의 컴퓨터가 데스크톱 머신 이지만 **send-pr(1)**같은 프로그램을 사용하려고 한다면 ISP의 메일 릴레이를 사용한다.
- 컴퓨터가 메일을 로컬에서 제어하지 않는 서버지만 모든 메일을 릴레이에 전달해야 된다.

대부분의 MTA 가 이에 대한 기능을 갖고 있다. 불행히 완벽하게 MTA 를 설정하는 것은 매우 어렵기 때문에 offloading 메일만 제어한다. **sendmail** 과 **postfix** 와 같은 프로그램은 이런 용도로 쉽게 사용된다.

게다가 일반적인 인터넷 서비스를 사용한다면 계약서에 여러분은 “메일 서버”를 운용하지 못하게 되어 있다.

---

필요한 것을 해결하는 가장 쉬운 방법은 mail/ssmtp 포트를 설치한다. root 에서 다음 명령을 실행한다:

```
# cd /usr/ports/mail/ssmtp
# make install replace clean
```

mail/ssmtp 를 설치한 후 /usr/local/etc/ssmtp/ssmtp.conf 의 4 개의 라인으로 설정할 수 있다:

```
root=yourrealemail@example.com
mailhub=mail.example.com
rewriteDomain=example.com
hostname=_HOSTNAME_
```

root 에 실제 메일 주소를 입력한다. mail.example.com 에(어떤 ISP 는 “외부 메일 서버” 또는 “SMTP” 서버라고 부른다) ISP 의 외부 메일 릴레이를 입력한다.

/etc/rc.conf 에서 *sendmail\_enable*="NONE"으로 설정하여 **sendmail** 을 비활성 한다.

mail/ssmtp 에 다른 옵션도 사용할 수 있다. 예제와 더 많은 정보는 /usr/local/etc/ssmtp 의 설정파일 예제나 **ssmtp** 의 매뉴얼 페이지를 본다.

**ssmtp** 를 이렇게 설정하면 여러분 ISP 의 사용량 정책을 침해하거나 스팸머에 의해 컴퓨터가 도용되지 않고 컴퓨터에서 메일 보내기 기능이 필요한 소프트웨어는 메일을 보낼 수 있다.

## 22.8 전화접속으로 메일 사용

고정 IP 주소를 가지고 있다면 기본 사항에서 수정할 필요가 없다. 호스트 이름을 할당된 인터넷 이름으로 설정하면 나머지는 **sendmail** 이 알아서 한다

유동적으로 할당되는 IP 번호를 가지고 있고 전화접속 PPP 로 인터넷에 연결한다면 ISP 의 메일 서버에 메일 박스를 가지고 있을 것이다. ISP 의 도메인이 example.net, 여러분의 유저 이름이 user 그리고 bsd.home 이라는 머신을 가지고 있다면 ISP 는 메일 릴레이로

---

relay.example.net 을 사용하라고 할 것이다.

메일 박스에서 메일을 가져오려면 메일을 가져오는 에이전트를 설치해야 된다. **fetchmail** 유틸리티는 다양한 프로토콜을 지원하기 때문에 적절한 선택이다. 이 프로그램은 패키지나 포트 컬렉션([mail/fetchmail](#))으로 사용할 수 있다. ISP 는 보통 POP3 를 지원한다. 유저 PPP 를 사용한다면 /etc/ppp/ppp.linkup 의 다음 엔트리로 인터넷이 연결됐을 때 자동으로 메일을 가져올 수 있다:

```
MYADDR:  
!bg su user -c fetchmail
```

로컬 계정이 아닌 곳으로 메일을 배달하는데 **sendmail** 을 사용한다면(아래와 같이) 인터넷에 연결되자마자 **sendmail** 이 여러분의 메일을 배달하기를 원할 것이다. 이렇게 하려면 /etc/ppp/ppp.linkup 의 fetchmail 명령 뒤에 다음 명령을 입력한다:

```
!bg su user -c "sendmail -q"
```

bsd.home 에 user 라는 계정을 가지고 있다고 가정한다. bsd.home 의 user 홈 디렉터리에 다음 내용이 있는 .fetchmailrc 파일을 생성한다:

```
poll example.net protocol pop3 fetchall pass MySecret
```

이 파일에 패스워드 *MySecret* 이 있기 때문에 user 외엔 아무도 읽을 수 없어야 한다.

*from:* 헤더로 메일을 보내려면 **sendmail** 이 *user@bsd.home* 대신 *user@example.net* 을 사용하도록 해야 된다. 또한 메일을 빠르게 송신하기 위해 **sendmail** 이 relay.example.net 을 통해 메일을 전송하도록 한다.

다음 .mc 파일이 적절하다:

```
VERSIONID('bsd.home.mc version 1.0')  
OSTYPE(bsd4.4)dnl  
FEATURE(nouucp)dnl  
MAILER(local)dnl  
MAILER(smtp)dnl
```

```
Cwlocalhost
Cwbsd.home
MASQUERADE_AS(`example.net')dnl
FEATURE(allmasquerade)dnl
FEATURE(masquerade_envelope)dnl
FEATURE(nocanonify)dnl
FEATURE(nodns)dnl
define(`SMART_HOST', `relay.example.net')
Dmbsd.home
define(`confDOMAIN_NAME', `bsd.home')dnl
define(`confDELIVERY_MODE', `deferred')dnl
```

이 .mc 파일과 sendmail.cf 파일을 변환하는 자세한 설명은 이전 섹션을 참고한다. 그리고 sendmail.cf 를 업데이트한 후 **sendmail** 을 재 시작한다.

## 22.10 SMTP 인증

SMTP 인증으로 메일 서버는 수많은 이점을 얻게 된다. SMTP 인증은 **sendmail** 에 다른 레이어 보안을 추가할 수 있고 매번 메일 클라이언트를 다시 설정할 필요 없이 호스트를 변경하는 모바일 유저에게 같은 메일 서버를 사용할 수 있는 기능을 제공한다.

- ① 포트에서 security/cyrus-sasl을 설치한다. security/cyrus-sasl은 많은 컴파일 옵션을 선택할 수 있고 목적에 맞게 여기서는 *pwcheck* 옵션을 선택한다.
- ② security/cyrus-sasl을 설치한 후 /usr/local/lib/sasl/Sendmail.conf에(또는 파일이 없다면 생성한다) 다음 라인을 추가한다:

```
pwcheck_method: passwd
```

이 방법은 **sendmail** 이 FreeBSD passwd 데이터베이스 인증을 사용하도록 한다. 이 설정으로 SMTP 인증을 사용해야 되는 각 유저의 새로운 유저이름과 패스워드를 생성하지 않고 로그인과 메일 패스워드를 같이 사용할 수 있다.

- ③ 이제 /etc/make.conf에 다음 라인을 추가한다:

---

```
SENDMAIL_CFLAGS=-I/usr/local/include/sasl1 -DSASL
SENDMAIL_LDFLAGS=-L/usr/local/lib
SENDMAIL_LDADD=-lsasl
```

이들 라인은 컴파일 할 때 `cyrus-sasl` 과 링크하도록 `sendmail` 에게 정확한 설정 옵션을 준다. `sendmail` 를 다시 컴파일 하기 전에 `cyrus-sasl` 이 설치되어 있어야 한다.

- ④ 다음 명령을 실행하여 `sendmail` 을 다시 컴파일 한다:

```
# cd /usr/src/usr.sbin/sendmail
# make cleandir
# make obj
# make
# make install
```

`/usr/src` 를 광범위하게 변경하지 않았고 필요한 공유 라이브러리를 이용할 수 있다면 `sendmail` 의 컴파일은 문제가 없다.

- ⑤ `sendmail` 을 컴파일하고 재 설치한 후 `/etc/mail/freebsd.mc` 파일을 수정한다(또는 `.mc` 파일로 사용하는 파일. 많은 관리자는 파일이 유일하도록 `hostname(1)` 의 출력을 `.mc` 파일 이름에 사용한다) 다음의 라인을 추가한다:

```
dnl set SASL options
TRUST_AUTH_MECH('GSSAPI DIGEST-MD5 CRAM-MD5 LOGIN')dnl
define(`confAUTH_MECHANISMS', `GSSAPI DIGEST-MD5 CRAM-MD5 LOGIN')dnl
define(`confDEF_AUTH_INFO', `/etc/mail/auth-info')dnl
```

이들 옵션은 유저 인증을 `sendmail` 에 맞는 방법으로 설정한다. `pwcheck` 가 아닌 다른 방법을 사용한다면 포함된 문서를 읽도록 한다.

- ⑥ 마지막으로 `/etc/mail` 에서 `make(1)` 을 실행한다. 이 명령은 새로운 `.mc` 파일을 실행해서 `freebsd.cf` 라는 `.cf` 파일을 생성한다(또는 `.mc` 파일에 사용하고 싶은 이름으로). 파일을 `sendmail.cf` 로 복사하고 `sendmail` 을 적절히 재 시작하는 `make`

---

install restart 명령을 사용한다. 이 절차에 대한 더 많은 정보는 /etc/mail/Makefile을 참고한다.

모든 것이 정확하다면 메일 클라이언트에서 로그인 정보를 입력하고 테스트 메시지를 보낼 수 있다. 추가적으로 더 자세히 검사하려면 **sendmail**의 *LogLevel*을 13으로 설정하고 /var/log/maillog에 에러가 있는지 본다.

/etc/rc.conf에 다음 라인을 추가하여 시스템이 부팅할 때마다 이 서비스를 사용할 수 있다:

```
sasl_pwcheck_enable="YES"
sasl_pwcheck_program="/usr/local/sbin/pwcheck"
```

이 설정은 시스템이 부팅됨에 따라 SMTP\_AUTH를 초기화한다.

SMTP 인증에 대한 더 많은 정보는 **sendmail** 페이지를 본다.

## 22.11 메일 유저 에이전트

메일 유저 에이전트는(MUA) 메일을 보내고 받는데 사용하는 어플리케이션이다. 더 나아가 메일이 발전되고 더욱 복잡해짐에 따라 MUA도 메일과 관련하여 점점 강력해졌다; 이것은 유저에게 강력한 기능과 유연성을 제공한다. FreeBSD는 포트 컬렉션으로 쉽게 설치할 수 있는 많은 메일 유저 에이전트를 가지고 있다. 유저는 **evolution** 또는 **balsa**와 같은 그래픽 메일 클라이언트; **mutt**, **pine**이나 mail과 같은 콘솔 기반 클라이언트; 또는 몇몇 거대 그룹에서 사용하는 웹 인터페이스 중 하나를 선택할 것이다.

### 22.11.1 mail

mail(1)은 FreeBSD에서 기본 메일 유저 에이전트다. 첨부 파일 추가와 로컬 메일 박스만 지원할 수 있는 제한이 있지만 텍스트 기반 메일을 보내고 받는데 필요한 기본적인 모든 기능을 제공하는 콘솔 기반 MUA다.

mail이 POP이나 IMAP 서버를 기본적으로 지원하지 않지만 이들 메일 박스는 다음 장에서

---

설명할(22.12 장) **fetchmail** 과 같은 어플리케이션으로 로컬 mbox 파일로 다운로드 할 수 있다.

메일을 보내고 받으려면 다음 예제처럼 단순히 mail 명령을 사용한다:

**% mail**

/var/mail 의 메일 박스 내용은 mail 유틸리티가 자동으로 읽는다. 메일 박스가 비어있다면 유틸리티는 메일이 없다는 메시지를 보여준다. 메일 박스를 읽은 후 어플리케이션 인터페이스가 시작되고 메시지 리스트를 보여준다. 메시지에선 다음 예제처럼 자동으로 숫자가 붙는다:

```
Mail version 8.1 6/6/93.  Type ? for help.
"/var/mail/marcs": 3 messages 3 new
>N  1 root@localhost      Mon Mar  8 14:05  14/510  "test"
  N  2 root@localhost      Mon Mar  8 14:05  14/509  "user account"
  N  3 root@localhost      Mon Mar  8 14:05  14/509  "sample"
```

이제 메시지는 mail 명령 **t** 에 보여주는 메시지 번호를 붙여서 읽는다. 이 예제에서는 첫 번째 메일을 읽는다:

**& t 1**

```
Message 1:
From root@localhost  Mon Mar  8 14:05:52 2004
X-Original-To: marcs@localhost
Delivered-To: marcs@localhost
To: marcs@localhost
Subject: test
Date: Mon,  8 Mar 2004 14:05:52 +0200 (SAST)
From: root@localhost (Charlie Root)
```

This is a test message, please reply if you receive it.

위의 예제에서 보았듯이 **t** 키가 메시지 헤더 전체를 보여준다. 메시지 리스트를 다시 표시하려면 **h** 키를 사용한다.

---

답변이 필요한 메일이면 **R** 이나 **r** mail 키로 답변할 수 있다. **R** 키는 메일을 보낸 사람에게만 답변을 보내지만 **r** 은 보낸 사람과 다른 수령자들에게도 메시지를 보낸다. 답변을 보내려는 메일 번호를 명령 뒤에 붙여도 된다. 명령을 입력한 후 답변 메시지를 작성하고 메시지 끝 다음 라인에 점(.) 하나를 입력한다. 다음과 같은 예제가 있다:

& R 1

To: root@localhost

Subject: Re: test

Thank you, I did get your email.

.

EOT

새로운 메일을 보내려면 **m** 키를 사용하고 수령자의 메일 주소를 입력한다. 여러 명의 수령자는 콤마(,)로 각 주소를 분리하여 지정한다. 메시지 제목을 입력하고 내용을 입력한다. 메시지 끝 다음 라인에 점(.) 하나를 입력해야 된다.

& mail root@localhost

Subject: I mastered mail

Now I can send and receive email using mail ... :)

.

EOT

mail 유틸리티를 사용 중간에 **?** 명령은 언제나라도 도움말을 표시한다. mail(1) 매뉴얼 페이지에서 mail 에 대한 더 많은 정보를 볼 수 있다.

**Note:** 앞에서 설명했듯이 mail(1) 명령은 파일 첨부을 지원하지 않기 때문에 파일은 첨부 되지 않는다. **mutt** 처럼 새로운 MUA 는 더욱 지능적으로 파일 첨부을 제어한다. 그래도 mail 명령을 사용하겠다면 [converters/mpak](#) 포트도 사용해보길 권한다.

## 22.11.2 mutt

**mutt** 는 작지만 다음과 같은 유용한 기능을 가진 강력한 메일 유저 에이전트다:



- 연속적인 메시지 처리;
- 디지털 사인과 메일 암호화를 위한 PGP 지원;
- MIME 지원;
- Maildir 지원;
- 원하는 대로 고치기 쉽다.

이 모든 특징이 **mutt**를 가장 발전된 메일 유저 에이전트 중 하나로 만들었다. **mutt**에 대한 더 많은 정보는 <http://www.mutt.org>를 참고한다.

안정 버전의 **mutt**는 `mail/mutt` 포트로 설치할 수 있지만 현재 개발 버전은 `mail/mutt-devel` 포트로 설치할 수 있을 것이다. 포트로 설치한 후 **mutt**는 다음 명령으로 실행할 수 있다:

```
% mutt
```

**mutt**는 `/var/mail`의 유저 메일박스 내용을 자동으로 읽어서 내용을 표시한다. 유저 메일박스에 메일이 없다면 **mutt**는 유저의 명령을 기다린다. 아래 예제는 **mutt**가 메시지 리스트를 보여준다.

```
q:Quit  d:Del  u:Undel  s:Save  m:Mail  r:Reply  g:Group  ?:Help
 1 N   Mar 09 Super-User  ( 1) test
 2 N   Mar 09 Super-User  ( 1) user account
 3 N   Mar 09 Super-User  ( 1) sample

*-Mutt: /var/mail/marcs [Msgs:3 New:3 1.6K]---(date/date)----- (all)---
```

메일을 읽으려면 단순히 커서 키로 메일을 선택하고 **Enter** 키를 누른다. **mutt**가 표시하는 메일 예제는 다음과 같다:

```
i:Exit -:PreoPg <Space>:NextPg v:View Attachm. d:Del r:Reply j:Next ?:Help
X-Original-To: marcs@localhost
Delivered-To: marcs@localhost
To: marcs@localhost
Subject: test
Date: Tue, 9 Mar 2004 10:28:36 +0200 (SAST)
From: Super-User <root@localhost>

This is a test message, please reply if you receive it.

-N - 1/1: Super-User      test      -- (all)
```

mail(1) 명령으로 **mutt** 는 메일을 보낸 사람과 모든 수령자들에게 유저가 답변을 보낼 수 있게 한다. 보낸 사람에게만 답변을 보내려면 **r** 키를 사용한다. 최초로 메일을 보낸 사람과 모든 메시지 수령자에게 답변을 보내려면 **g** 키를 사용한다.

**Note:** **mutt** 는 메일을 작성하고 답변을 입력하는 에디터로 **vi(1)** 명령을 사용할 수 있다. 이것은 유저가 홈 디렉터리에 자신만의 **.muttrc** 를 만들어서 **editor** 변수를 설정하면 된다.

새로운 메시지를 작성하려면 **m** 을 누른다. 적절한 제목을 입력하면 **mutt** 가 **vi(1)** 를 실행해서 메일을 작성할 수 있다. 메일 내용을 입력하고 저장한 후 **vi** 를 빠져 나오면 **mutt** 는 전송할 메일을 요약하여 보여준다. 메일을 보내려면 **y** 를 누른다. 요약 화면 예제는 다음과 같다:

```
y:Send q:Abort t:To c:CC s:Subj a:Attach file d:Descrip ?:Help
From: Marc Silver <marcs@localhost>
To: Super-User <root@localhost>
Cc:
Bcc:
Subject: Re: test
Reply-To:
Fcc:
Security: Clear

-- Attachments
- I 1 /tmp/mutt-bsd-c0hobscQ [text/plain, 7bit, us-ascii, 1.1K]

-- Mutt: Compose [Approx. msg size: 1.1K Atts: 1]
```

---

또한 ? 키를 입력하여 사용할 수 있는 광범위한 도움말을 **mutt** 도 가지고 있다. 가장 상위 라인에서도 키보드 단축키를 보여준다.

## 22.11.3 pine

**pine** 은 초보자를 겨냥하였지만 몇 가지 발전된 특징도 가지고 있다.

**주의:** **pine** 소프트웨어는 특수하게 준비된 메일을 보내서, 원격에서 로컬 시스템의 유저처럼 임의적인 코드를 실행할 수 있는 몇 개의 원격 취약점이 과거에 발견되었다. 이처럼 발견된 문제는 수정되었지만 **pine** 코드는 매우 불안정한 스타일로 작성되어서 FreeBSD 보안 관리자는 발견되지 않은 다른 취약점이 있다고 믿고 있다. **pine** 을 설치하면 직접 책임을 가져야 한다.

현재 버전의 **pine** 는 [mail/pine4](#) 포트로 설치할 수 있다. 포트를 설치한 후 **pine** 은 다음 명령으로 실행할 수 있다:

% **pine**

처음 **pine** 이 실행되면 간단한 소개와 함께 환영 메시지를 보여주고, 얼마나 많은 유저가 그들의 클라이언트를 사용하는지 파악하기 위한 익명 메일을 **pine** 개발 팀에 보내달라는 요청도 보여준다. 이 익명 메시지를 보내려면 **Enter** 를 누르고 메시지를 보내지 않고 환영 페이지를 빠져나가려면 **E** 를 누른다. 환영 페이지 예제는 다음과 같다:

```
PINE 4.58      GREETING TEXT      No Messages
<<<This message will appear only once>>>
Welcome to Pine ... a Program for Internet News and Email
We hope you will explore Pine's many capabilities. From the Main Menu,
select Setup/Config to see many of the options available to you. Also
note that all screens have context-sensitive help text available.
SPECIAL REQUEST: This software is made available world-wide as a public
service of the University of Washington in Seattle. In order to justify
continuing development, it is helpful to have an idea of how many people
are using Pine. Are you willing to be counted as a Pine user? Pressing
Return will send an anonymous (meaning, your real email address will not
be revealed) message to the Pine development team at the University of
Washington for purposes of tallying.
Pine is a trademark of the University of Washington.
[ALL of greeting text]
? Help      E Exit this greeting      PrevPage  P Print
Ret [Be Counted!]      Spc NextPage
```

환영 페이지를 빠져나오면 유저는 커서 키로 쉽게 이동할 수 있는 메인 메뉴에 있다. 이 메인 메뉴는 새로운 메일 작성, 메일 디렉터리 탐색 그리고 주소록 관리에 대한 단축키까지 제공한다. 메인 메뉴 아래에는 특정 작업을 수행할 수 있는 키보드 단축키를 보여준다.

pine 이 여는 기본 디렉터리는 inbox 다. 메시지 인덱스를 보려면 I 을 누르거나 아래와 같이 "MESSAGE INDEX" 옵션을 선택한다:

```

PINE 4.58  MAIN MENU                               Folder: INBOX  3 Messages
?  HELP          - Get help using Pine
C  COMPOSE MESSAGE - Compose and send a message
I  MESSAGE INDEX - View messages in current folder
L  FOLDER LIST   - Select a folder to view
A  ADDRESS BOOK  - Update address book
S  SETUP         - Configure Pine Options
Q  QUIT         - Leave the Pine program

Copyright 1989-2003.  PINE is a trademark of the University of Washington.
? Help          P PrevCmd          R RelNotes
O OTHER CMDS > [Index] N NextCmd        K KBlock
  
```

메시지 인덱스는 현재 디렉터리의 메시지를 보여주고 커서 키로 탐색할 수 있다. 밝게 표시된 메시지는 Enter 키를 눌러서 읽을 수 있다.

```

PINE 4.58  MESSAGE INDEX                           Folder: INBOX  Message 1 of 3 ANS
A  1 Mar  9 Super-User          (471) test
A  2 Mar  9 Super-User          (479) user account
A  3 Mar  9 Super-User          (473) sample

? Help          < FldrList    P PrevMsg      | PrevPage  D Delete     R Reply
O OTHER CMDS > [ViewMsg] N NextMsg      Spc NextPage  J Undelete   F Forward
  
```

아래 스크린 샷에 **pine** 이 표시한 샘플 메시지가 있다. 키보드 단축 키는 화면 하단에 표시된다. 이들 단축 키 예제 중 하나는 현재 표시하고 있는 메시지의 답변을 MUA 에게 보내는 **r** 키다.

```

PINE 4.58 MESSAGE TEXT Folder: INBOX Message 1 of 3 ALL ANS
Date: Tue, 9 Mar 2004 10:28:36 +0200 (SAST)
From: Super-User <root@localhost>
To: marcs@localhost
Subject: test

This is a test message, please reply if you receive it.

[ALL of message]
? Help      < MsgIndex  P PrevMsg      - PrevPage D Delete      R Reply
O OTHER CMDS > ViewAttch  N NextMsg      Spc NextPage J Undelete   F Forward

```

**pine** 에서 답장 메일은 기본적으로 **pine** 과 같이 설치되는 **pico** 에디터를 사용한다. **pico** 유틸리티는 메시지 사이를 이동하기 편하고 초보자에게 **vi(1)**나 **mail(1)**보다 쉽다. 답장이 끝나면 **Ctrl + X**를 눌러서 메시지를 보낼 수 있다. **pine** 어플리케이션은 확인을 요청한다.

```

PINE 4.58 COMPOSE MESSAGE REPLY Folder: INBOX 3 Messages
To      : Super-User <root@localhost>
Cc      :
Attchmnt:
Subject : Re: test
----- Message Text -----

I did recieve your message...

^G Get Help  ^X Send      ^R Read File ^Y Prev Pg   ^K Cut Text  ^O Postpone
^C Cancel    ^J Justify   ^W Where is  ^U Next Pg   ^U UnCut Text ^T To Spell

```

**pine** 어플리케이션은 메인 메뉴의 **SETUP** 옵션으로 적절히 조정할 수 있다. 더 많은 정보는 <http://www.washington.edu/pine/>을 본다.

---

## 22.12. fetchmail 사용

**fetchmail** 은 IMAP 과 POP 서버에서 메일을 자동으로 다운받아서 로컬 메일박스에 저장하는 완벽한 기능의 IMAP 과 POP 클라이언트다. **fetchmail** 은 [mail/fetchmail](#) 포트로 설치할 수 있고 다음과 같은 기능을 포함하여 여러 가지 기능을 제공한다:

- POP3, APOP, KPOP, IMAP, ETRN 그리고 ODMR 프로토콜 지원.
- 일반적으로 필터링, 포워딩 그리고 엘리머싱 기능을 허용하는 SMTP를 사용한 메일 포워딩 기능.
- 정기적으로 새로운 메시지를 체크하는 데몬 모드로 실행할 수 있을 것이다.
- 여러 개의 메일박스를 검색하여 설정에 따라 다른 로컬 유저에게 포워딩 할 수 있다.

**fetchmail** 의 모든 기능을 이 문서에서 설명하는 것은 범위를 벗어나기 때문에 기본적인 몇 가지 기능만 설명한다. **fetchmail** 유틸리티를 정확히 실행하기 위해 `.fetchmailrc` 라는 설정 파일이 필요하다. 이 파일은 서버 정보와 로그인 정보도 가지고 있다. 따라서 이 파일의 내용은 민감하기 때문에 다음 명령으로 파일 소유자만 읽을 수 있도록 만드는 것이 좋다.

```
% chmod 600 .fetchmailrc
```

다음 `.fetchmailrc` 는 POP 으로 유저 메일 박스 하나를 다운로드 하는 예제를 보여준다. **fetchmail** 에게 유저 이름 `joesoap` 와 패스워드 `XXX`로 `example.com` 에 연결하도록 한다. 이 예제는 `joesoap` 이 로컬 시스템의 유저도 된다고 가정한다.

```
poll example.com protocol pop3 username "joesoap" password "XXX"
```

다음 예제는 여러 개의 POP 과 IMAP 서버에 연결해서 적절한 로컬 유저 이름으로 리다이렉트 한다.

---

```
poll example.com proto pop3:
user "joesoap", with password "XXX", is "jsoap" here;
user "andrea", with password "XXXX";
poll example2.net proto imap:
user "john", with password "XXXXX", is "myth" here;
```

**fetchmail** 유틸리티는 `.fetchmailrc` 파일에 나열된 서버에 **fetchmail** 이 폴링하는 간격을(초 단위) 뒤에 입력하는 `-d` 플래그를 사용하여 데몬 모드로 실행 시킬 수 있다.

```
% fetchmail -d 60
```

**fetchmail** 에 대한 더 많은 정보는 <http://www.catb.org/~esr/fetchmail/>에서 찾을 수 있다.

## 22.13 procmail 사용

**procmail** 유틸리티는 받는 메일 필터로 사용하는 아주 강력한 어플리케이션이다. 받는 메일 중 매칭되는 메일이 특정 기능 수행하거나 다른 메일박스 또는 메일 주소로 다시 라우트 하도록 유저가 룰을 정의할 수 있다. **procmail** 은 `mail/procmail` 포트로 설치할 수 있다. 설치되면 대부분의 MTA 에 직접 통합된다; 더 많은 정보는 MTA 문서를 참고한다. 그렇지 않고 **procmail** 기능을 사용하는 유저 홈 디렉터리의 `.forward` 에 다음 라인을 추가하여 **procmail** 을 통합할 수 있다:

```
"|exec /usr/local/bin/procmail || exit 75"
```

다음 섹션은 기본적인 몇 개의 **procmail** 룰을 보여주고 간단히 기능을 설명한다. 이들 룰과 다른 것들은 유저 홈 디렉터리에 있는 `.procmailrc` 파일에 입력해야 된다.

이들 대부분의 룰은 `procmailex(5)` 매뉴얼 페이지에서 찾을 수 있다.

다음은 [user@example.com](mailto:user@example.com)에서 오는 모든 메일을 외부 메일 [goodmail@example2.com](mailto:goodmail@example2.com)으로 포워딩 한다:

---

```
:0
* ^From.*user@example.com
! goodmail@example2.com
```

1000 bytes 이하의 모든 메일은 외부 메일 [goodmail@example2.com](mailto:goodmail@example2.com)에 포워드한다:

```
:0
* < 1000
! goodmail@example2.com
```

[alternate@example.com](mailto:alternate@example.com)에 보낸 모든 메일을 alternate 메일 박스에 보낸다:

```
:0
* ^TOalternate@example.com
alternate
```

제목이 "Spam"인 모든 메일을 /dev/null 에 보낸다:

```
:0
^Subject:.*Spam
/dev/null
```

FreeBSD.org 메일링 리스트를 분석해서 적절한 메일 박스에 넣는 유용한 방법:

```
:0
* ^Sender:.owner-freebsd-W/[^@]+@FreeBSD.ORG
{
  LISTNAME=${MATCH}
  :0
  * LISTNAME??^W/[^@]+
  FreeBSD-${MATCH}
}
```



---

## 23 장 네트워크 서버

### 23.1 개요

이번 장은 유닉스 시스템에서 자주 사용되는 네트워크 서비스 중 몇 가지를 다룬다. 여러 가지 다른 종류의 네트워크 서비스를 어떻게 설치, 설정, 테스트하고 유지하는지 설명한다. 이번 장 전반에 설정 파일 예제도 포함되어 있다.

이번 장을 읽고 다음과 같은 사항을 알 수 있다:

- inetd 데몬은 어떻게 관리하는가
- 유저 계정을 공유하기 위한 네트워크 정보 서버는 어떻게 설치하는가
- DHCP를 사용하여 자동 네트워크 설정은 어떻게 하는가
- 도메인 네임서버는 어떻게 설치하는가
- 아파치 웹 서버는 어떻게 설치하는가
- 파일 전송 프로토콜(FTP) 서버는 어떻게 설치하는가
- 삼바(Samba)를 사용하여 윈도우 클라이언트를 위한 파일과 프린터 서버는 어떻게 설치하는가
- NTP 프로토콜로 시간과 날짜를 동기화 하고 타임 서버는 어떻게 설치하는가

이번 장을 읽기 전에 다음 사항을 알고 있어야 된다:

- /etc/rc 스크립트의 기본적인 이해
- 기본적인 네트워크 용어 이해
- 부가적인 소프트웨어 설치(4장)

### 23.2 inetd "슈퍼-서버"

#### 23.2.1 개요

inetd(8)은 여러 서비스의 연결을 관리하기 때문에 "인터넷 슈퍼-서버"라고 한다. 네트워크 서비스를 제공하는 프로그램들은 일반적으로 데몬으로 알려져 있다. **inetd** 는 다른 데몬을

---

관리하는 서버다. **inetd** 로 연결이 요청되면 어떤 데몬으로 연결해야 되는지 결정해서 해당 데몬을 실행하고 소켓을 생성한다. stand-alone 모드에서 각 데몬을 개별적으로 운영하는 것과 비교하여 하나의 **inetd** 인스턴스로 시스템의 전체 로드를 줄인다.

주로 **inetd** 는 다른 데몬을 실행하는데 사용되지만 **chargen**, **auth** 와 **daytime** 과 같은 여러 개의 평범한 프로토콜은 직접 관리한다.

이 섹션은 **inetd** 의 기본 설정부터 명령어 라인 옵션과 설정 파일 `/etc/inetd.conf` 까지 다룬다.

## 23.2.2 설정

**inetd** 는 `/etc/rc.conf` 시스템으로 초기화된다. 기본적으로 `inetd_enable` 옵션은 "NO"로 설정되어 있지만 **sysinstall** 로 보안 프로필을 중간으로 설정하면 활성화된다.

`/etc/rc.conf` 에 다음과 같이 설정하면 부팅할 때 **inetd** 를 활성화하거나 비활성 한다.

```
inetd_enable="YES"
또는
inetd_enable="NO"
```

게다가 `inetd_flags` 옵션으로 다른 명령어라인 옵션도 **inetd** 에 적용할 수 있다.

## 23.2.3 명령어 라인 옵션

**inetd** 개요:

```
inetd [-d] [-l] [-w] [-W] [-c maximum] [-C rate] [-a address / hostname] [-p filename] [-R rate] [configuration file]
```

-d

디버깅 켜기.

---

-l

성공적인 연결을 로그로 남긴다.

-w

외부 서비스에 TCP Wrapping 켜기(기본적으로).

-W

**inetd** 에 내장된 내부 서비스에 TCP Wrapping 켜기(기본적으로).

-c maximum

기본적으로 동시에 실행되는 각 서비스의 최대 값을 지정한다; 기본값은 무제한.  
*max-child* 매개변수를 사용하여 서비스 별로 제한할 수 있다.

-C rate

1 분 동안 하나의 IP 주소로부터 요청될 수 있는 서비스의 기본 최대값을 지정;  
기본값은 무제한. *max-connections-per-ip-per-minute* 매개변수를 사용하여  
서비스 별로 제한할 수 있다.

-R rate

1 분 동안 서비스가 발생할 수 있는 최대값 지정; 기본값은 256. 값 0 은  
무제한으로 허용한다.

-a

바인드 할 특정 IP 주소 하나를 지정한다. 그렇지 않고 IPv4 나 IPv6 주소가  
사용되는 호스트 이름과 일치할 경우 호스트 이름을 사용할 수 있다. 보통 호스트  
이름은 **inetd** 가 jail(8) 내부에서 운용될 때 지정하고 이 경우 호스트 이름은 jail(8)  
환경과 일치한다.

호스트 이름을 지정하여 사용하고 IPv4 와 IPv6 바인딩이 둘 다 요구될 때,  
*/etc/inetd.conf* 의 각 서비스를 위해 각 바인딩의 적절한 프로토콜 타입 엔트리가  
하나 필요하다. 예를 들어 TCP 기반 서비스는 *tcp4* 와 *tcp6* 프로토콜 엔트리가  
하나씩 필요하다.

-p

프로세스 ID 를 저장할 파일지정

이들 옵션은 */etc/rc.conf* 의 *inetd\_flags* 옵션으로 **inetd** 에 적용할 수 있다. 기본적으로

---

*inetd\_flags*는 *inetd*의 외부와 내부 서비스에 TCP Wrapping을 작동시키는 "-wW"를 설정한다. 경험이 없는 유저들을 위해 이들 매개변수는 수정할 필요가 없거나 */etc/rc.conf*에 입력되어 있다.

**Note:** 외부 서비스는 연결이 되면 실행되는 *inetd*의 외부 데몬이다. 한편 내부 서비스는 *inetd*가 자체적으로 처리한다.

## 23.2.4 *inetd.conf*

*inetd*의 설정은 */etc/inetd.conf* 파일로 제어된다.

*/etc/inetd.conf*를 수정하면 다음과 같이 *inetd* 프로세스에게 HangUP 신호를 보내서 *inetd*가 강제로 설정을 다시 읽도록 할 수 있다:

예제 23-1. *inetd*에 HangUP 신호 보내기

```
# kill -HUP `cat /var/run/inetd.pid`
```

설정 파일의 각 라인은 각 데몬을 지정한다. 파일에서 주석은 "#" 표시가 앞에 있다. */etc/inetd.conf*의 포맷은 다음과 같다:

```
service-name
socket-type
protocol
{wait|nowait}[/max-child[/max-connections-per-ip-per-minute]]
user[:group][:/login-class]
server-program
server-program-arguments
```

IPv4를 사용하는 *ftpd* 데몬 예제는 다음과 같다:

```
ftp      stream  tcp     nowait  root    /usr/libexec/ftpd    ftpd -l
```

**service-name**

이것은 특정 데몬의 서비스 이름이다. */etc/services*에 나열된 서비스와 일치해야

---

된다. 이것으로 **inetd** 가 어떤 포트를 준비해야 되는지 결정한다. 새로운 서비스를 생성한다면 `/etc/services` 에 먼저 등록해야 된다.

### socket-type

*stream*, *dgram*, *raw* 또는 *seqpacket* 중 하나. *stream* 은 연결 기반의 TCP 데몬에만 사용되지만 *dgram* 은 UDP 전송 프로토콜을 이용하는 데몬에 사용된다.

### protocol

다음 중 하나가 된다:

| 프로토콜      | 설명            |
|-----------|---------------|
| tcp, tcp4 | TCP IPv4      |
| udp, udp4 | UDP IPv4      |
| tcp6 TCP  | IPv6          |
| udp6 UDP  | IPv6          |
| tcp46     | TCP IPv4 와 v6 |
| udp46     | UDP IPv4 와 v6 |

### {wait|nowait}[/max-child[/max-connections-per-ip-per-minute]]

*wait/nowait* 은 **inetd** 가 실행한 데몬이 자체적인 소켓을 제어할 수 있는지 없는지를 보여 준다. *dgram* 소켓 타입은 *wait* 옵션을 사용해야 되지만 보통 멀티 스레드인 *stream* 소켓 데몬은 *nowait* 을 사용해야 된다. *wait* 은 보통 하나의 데몬이 여러 소켓을 사용하지만 *nowait* 은 새로운 각 소켓에 자식 데몬을 생성한다.

**inetd** 가 생성하는 자식 데몬의 최대값은 *max-child* 옵션을 사용하여 설정할 수 있다. 특정 데몬의 인스턴스를 10 으로 제한한다면 *nowait* 뒤에 */10* 이 필요하다.

게다가 한 곳에서 특정 데몬으로의 최대 연결을 제한하는 다른 옵션을 *max-child* 에 줄 수 있다. 이것은 *max-connections-per-ip-per-minute* 로 제한한다. 이곳의 값 10 은 특정 IP 주소에서 특정 서비스로 분 당 연결 시도를 10 회로 제한한다. 이 방법은 고의적이거나 고의적이지 않은 리소스 소비와 머신에 대한 서비스 거부 공격을(DOS) 방지하는데 유용하다.

---

이 필드에서 *wait* 이나 *nowait* 은 필수다. *max-child* 와 *max-connections-per-ip-per-minute* 은 옵션이다.

*max-child* 와 *max-connections-per-ip-per-minute* 제한이 없는 스트림 타입 멀티 쓰레드 데몬은 간단히 *nowait* 이다.

최대 제한이 10 인 같은 데몬은: *nowait /10* 이다.

게다가 IP 주소별로 분당 연결을 20 개로 제한하고 최대 10 개의 자식 프로세스로 제한한 데몬은: *nowait/10/20* 이다.

이들 옵션은 **fingerd** 데몬의 기본 설정으로 다음과 같이 이용되고 있다:

```
finger stream tcp nowait/3/10 nobody /usr/libexec/fingerd fingerd -s
```

#### user

*user* 는 특정 데몬을 운영하는 유저 이름이다. 대부분 데몬은 *root* 유저로 실행된다. 보안을 위해 몇몇 서버를 *daemon* 유저나 최소한의 권한을 가진 *nobody* 유저로 실행하는 것을 볼 수 있다.

#### server-program

연결 요청을 받았을 때 데몬을 실행하는 전체 경로다. 데몬이 *inetd* 에 의해 내부적으로 서비스된다면 *internal* 를 사용해야 한다.

#### server-program-arguments

이것은 데몬을 실행할 때 *argv[0]* 으로 시작하는 인자를 지정하여 *server-program* 과 결합하여 작동한다. **mydaemon -d** 가 명령어 라인이라면 *mydaemon -d* 는 *server-program-arguments* 의 값이 된다. 역시 데몬이 내부 서비스라면 *internal* 을 여기에 사용한다.

## 23.2.5 보안

설치할 때 보안 프로필을 선택했느냐에 따라서 많은 *inetd* 데몬이 기본적으로 활성화될 것이다. 필요 없다면 특정 데몬은 비활성 한다. 데몬의 시작 부분에 "#" 를 입력하고 *hangup* 신호를 *inetd* 에 보낸다. **fingerd** 와 같은 어떤 데몬은 공격에 대한 너무 많은

---

정보가 제공되었기 때문에 어떤 경우든 필요 없을 것이다.

어떤 데몬은 오랫동안 보안에 대해 신경 쓰지 않았거나 연결 시도에 대해 타임아웃이 없다. 따라서 공격자는 특정 데몬에 눈치채지 못하게 연결을 시도하여 공격해서 이용 가능한 자원을 소모할 수 있다. 특정 데몬에 *ip-per-minute* 와 *max-child* 한계를 두는 것은 좋은 생각이다.

기본적으로 TCP wrapping 은 켜 있다. **inetd** 로 운용되는 다양한 데몬에 TCP 제한을 적용하는 정보는 `hosts_access(5)` 매뉴얼 페이지를 참고한다.

## 23.2.6 내부 서비스 데몬들

**daytime**, **time**, **echo**, **discard**, **chargen** 와 **auth** 는 모두 **inetd** 의 내부 서비스를 제공 받는다.

**auth** 서비스는 네트워크 서비스 인증을 제공하고 특정 레벨까지 설정할 수 있다.

좀더 자세한 정보는 `inetd(8)` 매뉴얼 페이지를 참고한다.

## 23.3 NFS

FreeBSD 는 다양한 다른 파일시스템 중 NFS 로 유명한 네트워크 파일시스템을 지원한다. NFS 는 시스템이 디렉터리와 파일 시스템을 네트워크를 통하여 공유하도록 한다. NFS 로 유저와 프로그램은 대부분의 원격 시스템의 파일을 로컬 파일처럼 사용할 수 있다.

NFS 가 제공하는 중요한 장점 몇 가지는 다음과 같다:

- 일반적으로 사용되는 데이터를 머신 한대에 저장해서 네트워크로 접근할 수 있기 때문에 로컬 워크스테이션은 더 적은 디스크 공간을 사용한다.
- 네트워크에 있는 모든 머신에 유저의 홈 디렉터리를 생성할 필요 없다. NFS 서버에 홈 디렉터리를 설정해서 네트워크로 사용할 수 있다.
- 플로피 디스크, CDROM 드라이버와 ZIP 드라이브 같은 스토리지 장치는 다른

---

머신에서 네트워크로 사용할 수 있다. 네트워크를 통해 여러 개의 이동용 미디어 드라이브를 줄일 수 있다.

### 23.3.1 NFS 는 어떻게 동작하는가

NFS 는 최소한 두 개의 메인 파트로 구성된다: 서버와 하나 이상의 클라이언트. 클라이언트는 서버 머신에 저장되어 있는 데이터를 원격에서 접속한다. 이러한 기능이 정확히 작동하기 위해 몇 가지 프로세스를 설정하고 실행해야 된다:

**Note:** FreeBSD 5.X 에서 **portmap** 유틸리티는 **rpcbind** 유틸리티로 대체되었다. 그래서 FreeBSD 5.X 유저는 다음 예제들의 모든 **portmap** 인스턴스를 **rpcbind** 로 모두 변경한다.

서버에서는 다음 데몬들이 동작해야 된다:

| 데몬             | 설 명                                                  |
|----------------|------------------------------------------------------|
| <b>nfsd</b>    | NFS 클라이언트로부터의 NFS 서비스 요청을 처리하는 데몬                    |
| <b>mountd</b>  | nfsd(8)이 요청해서 실행되는 NFS 마운트 데몬                        |
| <b>portmap</b> | portmapper 데몬은 NFS 서버가 사용하는 포트를 NFS 클라이언트가 감지하도록 한다. |

클라이언트도 **nfsiod** 로 알려진 데몬을 실행할 수 있다. **nfsiod** 데몬은 NFS 서버의 요청을 서비스한다. 이것은 부가적이고 성능을 향상시키지만 일반적이고 정확한 운용에는 필요 없다. 더 많은 정보는 **nfsiod(8)** 매뉴얼 페이지를 본다.

### 23.3.2 NFS 설정

NFS 설정은 비교적 직선적이다. **/etc/rc.conf** 파일을 약간 수정해서 시스템이 시작할 때 필요한 사항을 모두 실행할 수 있다.

NFS 서버의 **/etc/rc.conf** 파일에 다음 옵션을 설정한다:



```
portmap_enable="YES"
nfs_server_enable="YES"
mountd_flags="-r"
```

NFS 서버가 활성화되면 **mountd** 가 자동으로 실행된다.

클라이언트의 `/etc/rc.conf` 에는 다음 옵션이 필요하다:

```
nfs_client_enable="YES"
```

NFS 로 공유할 파일시스템을 `/etc/exports` 파일에 지정한다. `/etc/exports` 의 각 라인에 공유되는 파일시스템과 어떤 머신이 이 파일 시스템에 접근하는지 정의한다. 이렇게 해서 어떤 머신이 파일시스템에 접근하는지 접근 옵션도 지정할 수 있다. 이 파일에 사용할 수 있는 다양한 옵션이 있지만 여기서는 몇 가지만 언급한다. `exports(5)` 매뉴얼 페이지에서 다른 옵션을 발견할 수 있다.

여기 몇 개의 `/etc/exports` 엔트리 예제가 있다:

설정은 여러분의 환경과 네트워크 상황에 따라 다르겠지만 다음 예제는 파일시스템을 어떻게 공유하는지 보여준다. 예를 들어 서버와 같은 도메인(각각을 위한 도메인 이름의 부족으로)을 가지고 있거나 `/etc/hosts` 파일에 엔트리가 있는 3 대의 예제 머신에 `/cdrom` 디렉터리를 공유한다. `-ro` 플래그는 파일시스템을 읽기 전용으로 공유한다. 이 플래그 때문에 원격 시스템은 공유된 이 파일시스템의 어떤 것도 변경할 수 없다.

```
/cdrom -ro host1 host2 host3
```

다음 라인은 IP 주소로 3 대의 호스트에 `/home` 을 공유한다. DNS 서버 설정이 없는 사설 네트워크에 있다면 유용한 설정이다. 부가적으로 `/etc/hosts` 파일에 내부 호스트 이름을 설정할 수 있다; 더 많은 정보는 `hosts(5)` 를 본다. `-alldirs` 플래그는 서브 디렉터리를 마운트 포인트로 사용할 수 있게 한다. 다시 말해서 서브 디렉터리는 마운트 할 수 없지만 클라이언트가 필요로 하는 디렉터리만 마운트 하도록 한다.

```
/home -alldirs 10.0.0.2 10.0.0.3 10.0.0.4
```

다음 라인은 `/a` 를 공유하기 때문에 다른 도메인의 클라이언트 2 대가 파일시스템에 접근할

---

것이다. `-maproot=root` 플래그는 원격 시스템의 root 유저가 root 로 공유된 파일시스템에 적성을 할 수 있게 한다. `-maproot=root` 플래그가 없다면 유저가 원격 시스템의 root 권한을 가지고 있더라도 공유된 파일시스템을 수정할 수 없다.

```
/a -maproot=root host.example.com box.example.org
```

클라이언트가 공유된 파일시스템에 접근하려면 클라이언트는 퍼미션을 가지고 있어야 한다. `/etc/exports` 파일에 원하는 클라이언트를 지정한다.

`/etc/exports` 에서 각 라인은 파일시스템 하나를 호스트 하나에 공유하는 정보를 나타낸다. 원격 호스트는 파일시스템 별로 한번씩 지정할 수 있고 하나의 기본 엔트리를 가질 것이다. 예를 들어 `/usr` 이 하나의 파일시스템이라고 가정하면 다음의 `/etc/exports` 는 유효하지 않다:

```
/usr/src client
/usr/ports client
```

하나의 파일시스템 `/usr` 이 같은 호스트 `client` 에 두 라인으로 공유되어 있다. 이 상황에 맞는 정확한 포맷은 다음과 같다:

```
/usr/src /usr/ports client
```

파일시스템에 포함되어 있는 디렉터리들은 원하는 호스트에 공유하려면 모두 한 라인에 지정한다. 클라이언트가 지정되지 않은 라인은 싱글 호스트로 취급된다. 이것은 파일시스템 공유를 제한하지만 대부분의 사람들에게 문제는 없다.

다음 예제는 `/usr` 과 `/exports` 가 로컬 파일시스템인 유효한 공유 리스트다:

```
# src 와 ports 를 client01 과 client 02 에 공유하지만 client01 만 root 권한을 갖는다.
/usr/src /usr/ports -maproot=root client01
/usr/src /usr/ports client02
# client 머신은 root 권한을 가지고 있어서 /export 의 어느 곳이든 마운트 할 수 있다.
# 모든 사람들이 /exports/obj 를 읽기 권한으로 마운트 할 수 있다.
/exports -alldirs -maproot=root client01 client02
/exports/obj -ro
```

---

/etc/exports 를 수정할 때마다 **mountd** 를 재 시작해서 변경된 내용을 반영해야 된다.  
이것은 mountd 프로세스에게 HUP 신호를 보내면 된다:

```
# kill -HUP `cat /var/run/mountd.pid`
```

그렇지 않고 재 부팅할 필요가 없더라도 재 부팅하게 되면 FreeBSD 의 모든 설정을 정확히 반영한다. 다음 명령을 root 에서 실행하면 모든 것이 시작된다.

NFS 서버에서 아래 명령을 실행한다:

```
# portmap
# nfsd -u -t -n 4
# mountd -r
```

NFS 클라이언트에서 다음 명령을 실행한다:

```
# nfsiod -n 4
```

이제 실제로 원격 파일시스템을 마운트 할 수 있도록 모든 준비가 되었다. 이 예제에서 서버 이름은 *server* 고 클라이언트의 이름은 *client* 다. 원격 파일시스템을 일시적으로 마운트 하거나 설정을 테스트만 하려면 클라이언트에서 root 로 다음 명령을 실행한다:

```
# mount server:/home /mnt
```

이 명령은 서버의 /home 디렉터리를 클라이언트의 /mnt 에 마운트 한다. 모든 설정이 정확하다면 클라이언트의 /mnt 에서 서버에 있는 모든 파일을 볼 수 있다.

컴퓨터가 부팅할 때마다 원격 파일시스템을 자동으로 마운트하려면 /etc/fstab 파일에 파일 시스템을 추가한다. 여기 예제가 있다:

|                                             |
|---------------------------------------------|
| <pre>server:/home  /mnt  nfs rw  0  0</pre> |
|---------------------------------------------|

fstab(5) 매뉴얼 페이지에서 이용할 수 있는 모든 옵션을 찾을 수 있다.

---

### 23.3.3 실용적인 사용

NFS 는 실제로 많이 사용된다. 이들 중 매우 보편적인 것을 아래에 적었다:

- 몇 대의 머신에 CDROM을 공유하도록 설정하거나 다른 미디어(플로피나 테잎)를 이들 머신에 공유한다. 이 방법은 저렴하고 가끔 여러 대의 머신에 소프트웨어를 설치할 때 편리한 방법이다.
- 대형 네트워크에서 모든 유저의 홈 디렉터리를 저장할 중앙 NFS 서버를 설정하면 아주 편리할 것이다. 이런 홈 디렉터리는 네트워크로 공유할 수 있기 때문에 유저가 어떤 워크스테이션에 로그인 하더라도 항상 같은 홈 디렉터리를 사용할 수 있다.
- 몇 대의 머신에 /usr/ports/distfiles 디렉터를 공유할 수 있다. 여러 대의 머신에서 포트를 설치해야 된다면 이 방법으로 각각의 머신에 소스를 다운로드 할 필요 없이 빠르게 설치할 수 있다.

### 23.3.4 amd 로 자동 마운트

amd(8)은(자동 마운트 데몬) 파일이나 디렉터리가 있는 파일시스템에 접근할 때마다 원격 파일시스템을 자동으로 마운트 한다. 파일시스템이 일정 시간 동안 사용되지 않으면 **amd** 는 자동으로 언 마운트 한다. 단순히 **amd** 의 기능을 사용하거나 영구적으로 마운트 하려면 보통 /etc/fstab 에 나열하면 된다.

**adm** 는 NFS 서버의 /host 와 /net 디렉터리에서 동작한다. 파일이 이들 디렉터리 중 한곳에 접근하면 **amd** 는 적절한 원격 마운트 위치를 찾아서 자동으로 마운트한다. /net 은 IP 주소로 공유 파일시스템을 마운트 하는데 사용되지만 /host 은 원격 호스트 이름으로 마운트할 때 사용된다.

/host/foobar/usr 에 있는 파일에 접근하려면 **amd** 에게 공유된 /usr 을 호스트 foobar 에 마운트 하도록 한다.

#### 예제 23-2. amd 로 공유 디렉터리 마운트

showmount 명령으로 원격 호스트의 이용할 수 있는 마운트를 볼 수 있다. 예를 들어 호스트 이름 foobar 의 마운트를 보기 위해 다음 명령을 사용할 수 있다:

---

```
% showmount -e foobar
```

```
Exports list on foobar:
```

```
/usr          10.10.10.0
/a           10.10.10.0
```

```
% cd /host/foobar/usr
```

예제에서 보았듯이 showmount 는 /usr 를 공유된 것으로 보여 준다. /host/foobar/usr 로 디렉터리를 변경하면 amd 는 호스트 이름 foobar 를 해석하고 자동으로 원하는 공유 디렉터리를 마운트 한다.

amd 는 다음 라인을 /etc/rc.conf 에 추가하여 시작 스크립트로 시작할 수 있다:

```
amd_enable="YES"
```

추가적으로 사용자 플래그를 amd\_flags 옵션으로 amd 에 적용할 수 있다. 기본적으로 amd\_flags 는 다음과 같이 설정된다:

```
amd_flags="-a /.amd_mnt -l syslog /host /etc/amd.map /net /etc/amd.map"
```

/etc/amd.map 파일은 공유 디렉터리가 마운트 될 때의 기본 옵션을 정의한다.

/etc/mad.conf 파일은 amd 의 몇 가지 고급 기능을 정의한다.

더 많은 정보는 amd(8)과 amd.conf(5) 매뉴얼 페이지를 참고한다.

### 23.3.5 다른 시스템과 통합 문제

ISA PC 시스템의 특정 이더넷 카드는 특히 NFS 와 관련하여 심각한 네트워크 문제를 유발하는 한계를 가지고 있다. 애매하게도 FreeBSD 에 명확히 나타나지 않지만 FreeBSD 시스템도 영향을 받는다.

(FreeBSD)PC 시스템이 Silicon Graphics 나 Sun Microsystems 이 만든 고성능 워크스테이션과 네트워크로 연결되었을 때 이 문제가 거의 발생한다. NFS 마운트 동작은 양호하고 몇 가지 동작도 성공적이지만 다른 시스템이 계속해서 요청을 하더라도 갑자기 서버가 클라이언트에게 응답을 하지 않는다. 이 문제는 클라이언트가 FreeBSD

---

시스템이거나 워크스테이션일 때 클라이언트에 발생한다. 이 문제가 발생하면 클라이언트를 안전하게 셧 다운할 수 있는 방법이 없다. NFS 문제가 해결되지 않기 때문에 유일한 방법은 클라이언트를 재 부팅해야 된다. "정확한" 해결책은 FreeBSD 시스템에 고성능 이더넷 카드를 붙이면 되지만 이런 예러가 발생하지 않게 하는 간단한 조치가 있다. FreeBSD 시스템이 서버라면 클라이언트에서 마운트 할 때 `-w=1024` 옵션을 준다. FreeBSD 시스템이 클라이언트라면 NFS 파일 시스템을 옵션 `-r=1024`로 마운트 한다. 자동으로 마운트 하도록 이들 옵션을 클라이언트 `fstab`의 4 번째 필드에 지정하여 사용하거나 직접 마운트 할 때 마운트 명령에 `-o` 매개변수를 사용한다.

NFS 서버와 클라이언트가 다른 네트워크에 있을 때 다른 문제가 있다. 이런 경우라면 필요한 UDP 정보를 라우터가 라우팅 해야되고 그렇지 않으면 사용하지 못한다.

다음 예제에서 `fastws`은 고성능 워크스테이션 호스트(인터페이스) 이름이고 `freebox`는 저급 이더넷 카드를 가진 FreeBSD 시스템의 호스트(인터페이스) 이름이다. 또한 `/sharedfs`는 NFS 파일시스템(`exports(5)`를 본다)으로 공유되고 `/project`는 클라이언트에서 공유 디렉터리를 마운트하는 마운트 포인트이다. 이와 같은 경우 `hard`나 `soft` 그리고 `bg`와 같은 추가적인 옵션을 어플리케이션이 요구할 것이다.

FreeBSD 시스템이(`freebox`) 클라이언트인 `freebox`의 `/etc/fstab` 예제는 다음과 같다:

```
fastws:/sharedfs /project nfs rw,-r=1024 0 0
```

`freebox`에서 직접 마운트 명령을 사용하려면 다음 명령을 실행한다:

```
# mount -t nfs -o -r=1024 fastws:/sharedfs /project
```

FreeBSD 시스템이 서버인 `fastws`의 `/etc/fstab`의 예제는 아래와 같다:

```
freebox:/sharedfs /project nfs rw,-w=1024 0 0
```

`fastws`에서 직접 마운트한다면 다음 명령을 사용한다:

```
# mount -t nfs -o -w=1024 freebox:/sharedfs /project
```

거의 모든 16 비트 이더넷 카드는 읽기와 쓰기 크기에 위의 제한이 없이 동작된다.

---

문제가 발생하면 어떤 일이 발생하고 왜 해결할 수 없는지 설명하면, NFS 는 일반적으로 8k 의 "블록" 크기로 동작한다(더 작은 크기의 조각으로 동작할 수 있지만). 최대 이더넷 패킷이 대략 1500 바이트기 때문에 NFS "블록"은 상위 계층 코드에서 하나의 유닛이 되지만, 여러 개의 이더넷 패킷으로 나누어서 받은 후 재 조합해서 일반적으로 인정되는 유닛이 된다. 고성능 워크스테이션은 NFS 유닛을 구성하는 패킷을 최대한 표준에 맞추어 패킷 간격을 조밀하게 하나씩 내보낸다. 저급 카드가 있는 호스트에 이들 패킷이 전송된 후 모든 패킷이 다시 조합되어 인정되기 전에 나중에 보낸 패킷이 같은 유닛의 이전 패킷을 오버런한다. 이 결과 워크스테이션은 타임아웃 되어 다시 시도하지만 8K 유닛 전체를 다시 시도하게 되고 이 절차는 계속 반복된다.

유닛 크기를 이더넷 패킷 크기 이하로 제한하여 완전히 받은 이더넷 패킷이 교착 상태를 피하고 각각 승인될 수 있다.

오버런은 고성능 워크스테이션이 PC 시스템으로 데이터를 보내면 계속 발생할 수 있지만, 더 좋은 카드에서도 NFS 는 이런 오버런을 보장하지 않는다. 오버런이 발생하면 영향을 받은 유닛은 재 전송되어 받은 후 재 조합해서 인정된다.

## 23.4 NIS/YP

### 23.4.1 NIS 는 무엇인가?

네트워크 정보 서비스를 나타내는 NIS 는 유닉스(원래 SunOS) 시스템을 중앙에서 관리하기 위해 Sun Microsystems 에서 개발하였다. 이제 산업 표준이 되어 주요 주요 유닉스 시스템들이 NIS 를 지원한다(Solaris, HP-UX, AIX, Linux, NetBSD, OpenBSD, FreeBSD 등).

NIS 는 원래 옐로우 페이지로 알려졌지만 상표 문제로 Sun 이 이름을 바꾸었다. 이 예전 용어는(yp) 아직도 종종 사용되고 있다.

NIS 는 NIS 도메인의 머신들이 일반적인 설정 파일을 공유하도록 하는 RPC 기반 클라이언트/서버 시스템이다. 따라서 시스템 관리자는 최소한의 데이터 설정으로 NIS 클라이언트 시스템을 설정하고 한 곳에서 설정 데이터를 추가, 삭제 수정할 수 있다.

이것은 Windows NT 도메인 시스템과 비슷하다; 내부적인 수행은 다르지만 기본 기능은

---

동일하다.

## 23.4.2 알고 있어야 할 용어/프로세스

FreeBSD 를 NIS 서버나 NIS 클라이언트로 만들 때 수행해야 되는 여러 가지 중요한 유저 프로세스와 용어가 있다:

| 용어            | 설명                                                                                                                                                                                                                                                     |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| NIS 도메인 이름    | NIS 마스터 서버와 모든 클라이언트는(슬레이브 서버를 포함하여) NIS 도메인 이름을 가지고 있다. Windows NT 도메인 이름과 비슷하고 NIS 도메인 이름은 DNS 와 어떤 연관도 없다.                                                                                                                                          |
| portmap       | RPC(NIS 가 사용하는 네트워크 프로토콜인 원격 프로시저 콜)를 활성화하기 위해 사용해야 된다. portmap 이 실행 중이지 않으면 NIS 서버를 실행할 수 없거나 NIS 클라이언트로 사용할 수 없다.                                                                                                                                    |
| yplib         | NIS 클라이언트를 NIS 서버에 연결시킨다. yplib 는 시스템에서 NIS 도메인 이름을 받아서 RPC 를 사용하여 서버에 연결한다. 그리고 yplib 는 NIS 환경에서 클라이언트-서버 통신의 핵심이다; 클라이언트 머신에서 yplib 가 정지했다면 NIS 서버에 접근할 수 없다.                                                                                        |
| yplib         | 오직 NIS 서버에서만 실행된다; 이것이 NIS 서버 프로세스다. yperv(8)이 죽는다면 서버는 더 이상 NIS 요청에 응답할 수 없다(다행히 이 장애를 극복할 수 있는 슬레이브 서버가 있다). 지금까지 사용하던 서버가 죽었다면 다른 서버로 다시 연결하지 않는 NIS 의 몇 가지 문제가 있다. 가끔 이 문제를 해결할 수 있는 유일한 방법은 서버 프로세스(또는 서버 전체를) 또는 클라이언트에서 yplib 프로세스를 재 시작하는 것이다. |
| rpc.yppasswdd | NIS 마스터 서버에서만 실행해야 되는 다른 프로세스; NIS 클라이언트에서 유저가 NIS 패스워드를 변경할 수 있게 하는 데몬이다. 이 데몬이 실행되지 않는다면 유저는 NIS 마스터 서버에 로그인하여 클라이언트의 패스워드를 변경해야 된다.                                                                                                                 |

## 23.4.3 어떻게 동작하는가?

NIS 환경에 3 종류의 호스트가 있다: 마스터 서버, 슬레이브 서버와 클라이언트. 서버는 호스트 설정 정보를 중앙에 저장하는 저장소처럼 동작한다. 마스터 서버는 신뢰할 수



---

있도록 이 정보를 가지고 있지만 슬레이브 서버는 이 정보를 여분으로 미러한다.  
클라이언트는 서버로부터 이 정보를 제공받아 동작한다.

이 방법을 이용하여 다양한 파일 정보를 공유할 수 있다. `master.passwd`, `group` 과 `hosts` 파일이 일반적으로 NIS 를 통해 공유된다. 보통 이런 파일에서 찾을 수 있는 정보가 필요할 때마다 클라이언트는 NIS 서버에 요청하지 않고 직접 바운드(연결)한다.

### 23.4.3.1 머신 종류

- *NIS 마스터 서버.* Windows NT 주 도메인 컨트롤러와 유사한 이 서버는 모든 NIS 클라이언트가 사용하는 파일을 관리한다. 마스터 서버에 있는 `passwd`, `group` 그리고 다양한 다른 파일은 NIS 클라이언트가 사용한다.

**Note:** 머신 한대가 하나 이상의 NIS 도메인의 마스터 서버가 될 수 있다. 그러나 여기서는 소규모의 NIS 환경을 다루기 때문에 이 교제에서는 다루지 않는다.

- *NIS 슬레이브 서버.* NT 백업 도메인 컨트롤러와 비슷한 NIS 슬레이브 서버는 NIS 마스터의 데이터 파일 복사본을 관리한다. NIS 슬레이브 서버는 중요한 환경을 백업으로 제공하고 마스터 서버의 로드를 분산하는데 도움이 된다: NIS 클라이언트는 항상 처음으로 응답하는 NIS 서버에 연결되고 여기에 슬레이브 서버의 응답도 포함된다.
- *NIS 클라이언트.* 대부분의 Windows NT 워크스테이션처럼 NIS 클라이언트도 로그인 하기 위해 NIS 서버 인증이 필요하다(또는 Windows NT 워크스테이션의 경우에는 Windows NT 도메인 컨트롤러의 인증).

### 23.4.4 NIS/YP 사용

이 섹션은 샘플 NIS 환경 설정을 설명한다.

**Note:** 이번 섹션은 FreeBSD 3.3 또는 이후의 버전을 사용한다고 가정한다.  
여기서 설명하는 내용은 3.0 이후의 FreeBSD 버전에서 동작하겠지만 확실하지는 않다.

---

### 23.4.4.1 계획

여러분이 대학의 작은 연구실의 관리자라고 가정한다. 15 대의 FreeBSD 머신으로 이루어진 이 연구실은 현재 중앙 관리가 되지 않고 있다; 각 머신은 각자의 `/etc/passwd` 와 `/etc/master.passwd` 를 가지고 있다. 이 파일들은 수동으로 서로 동기화한다; 현재 연구실에 유저를 추가하려면 15 대의 머신에서 `adduser` 를 실행해야 된다. 이렇게 변경해야 되기 때문에 연구실의 머신 두 대를 서버로 사용하여 NIS 를 도입하기로 결정했다.

따라서 연구실의 설정은 다음과 비슷하다:

| 머신 이름     | IP 주소         | 머신 역할        |
|-----------|---------------|--------------|
| ellington | 10.0.0.2      | NIS 마스터      |
| coltrane  | 10.0.0.3      | NIS 슬레이브     |
| basie     | 10.0.0.4      | 교원전용 워크스테이션  |
| bird      | 10.0.0.5      | 클라이언트 머신     |
| cli[1-11] | 10.0.0.[6-17] | 다른 클라이언트 머신들 |

처음으로 NIS 를 설정할 계획이라면 어떻게 진행할 것인지 생각해 보는 것이 좋다. NIS 를 생성하기 위해 필요한 사항이 얼마 되지 않기 때문에 네트워크 크기에 대해서는 걱정하지 않는다.

#### [NIS 환경을 구축할 때 주의 사항]

##### 1. NIS 도메인 네임 선택

이 이름은 사용하려는 “도메인 네임”이 아니다. 엄밀히 말해서 "NIS 도메인 네임" 이다. 클라이언트가 서버에게 정보를 요청하기 위해 브로드캐스트할 때 자신이 속하는 NIS 도메인 네임도 포함되어 있다. 따라서 하나의 네트워크에 여러 대의 서버가 있을 때 어떤 서버가 요청을 처리할지 결정할 수 있다. 관련이 있는 호스트를 그룹화 할 수 있는 이름을 NIS 도메인네임으로 생각해본다.

어떤 단체들은 NIS 도메인 이름으로 인터넷 도메인을 선택한다. 네트워크 문제를 디버깅할 때 혼란스러울 수 있기 때문에 권장하지 않는다. NIS 도메인 이름은 여러분의 네트워크에서 유일해야 되고 머신 그룹을 설명하는 이름이 유용하다. 예를 들어 학교에서 예술부는 "acme-art" NIS 도메인 일 것이다. 이 예제에서는 *test-domain* 을 선택했다고 가정한다.

그러나 어떤 운영체제(특히 SunOS)는 NIS 도메인 이름을 인터넷 도메인 이름처럼

---

사용한다. 네트워크에서 하나 이상의 머신이 이러한 제한을 가지고 있다면 NIS 도메인 이름으로 인터넷 도메인 이름(DNS)을 사용해야 된다.

## 2. 물리적인 서버 요구 사항

NIS 서버로 사용할 머신을 선택할 때 몇 가지 유의사항이 있다. NIS 대한 한가지 불행한 것은 서버에 관한 클라이언트의 의존성 레벨이다. 클라이언트가 NIS 도메인 서버에 연결할 수 없다면 가끔 머신을 사용할 수 없다. 유저와 그룹 정보의 부족으로 대부분의 시스템이 일시적으로 정지된다. 그래서 자주 재 부팅 하거나 개발용으로 사용하던 머신은 선택에서 배제해야 된다. 트래픽이 아주 심하지 않은 네트워크라면 다른 서비스에 사용하는 머신에도 NIS 서버를 올릴 수 있지만 NIS 서버를 사용할 수 없게 되면 모든 NIS 클라이언트도 사용하지 못한다는 것을 생각해야 된다.

### 23.4.4.2 NIS 서버

모든 NIS 정보는 규칙적으로 복사되어 NIS 마스터 서버라고 부르는 하나의 머신에 저장된다. 정보를 저장하는데 사용되는 데이터베이스는 NIS 맵이라고 한다. FreeBSD 에서 이들 맵은 `/var/yp[domainname]`에 저장되고 `[domainname]`은 NIS 도메인 이름이다. 하나의 NIS 서버에서 여러 개의 도메인을 한번에 지원할 수 있기 때문에 지원되는 각 도메인 별로 디렉터리를 가질 수 있다. 따라서 각 도메인은 자신만의 독립적인 맵 세트를 가진다.

NIS 마스터와 슬레이브 서버는 `ypserv` 데몬으로 모든 NIS 요청을 제어한다. `ypserv` 는 NIS 클라이언트의 요청에 응답하고, 요청된 도메인과 맵 이름을 적절한 데이터베이스 파일 경로로 해석하여 데이터베이스에서 클라이언트로 데이터를 다시 돌려보낸다.

#### [NIS 서버 설정]

---

## 1. NIS 마스터 서버 설정

NIS 마스터 서버를 설정하는 것은 상당히 직관적이고 목적에 따라 다르다. FreeBSD 는 NIS 의 모든 것을 지원한다. 필요한 모든 것은 /etc/rc.conf 에 다음 라인을 추가하면 나머지는 FreeBSD 가 처리한다.

① nisdomainname="test-domain"

이 라인은 네트워크 설정에 따라 NIS 도메인 이름을 test-domain 으로 설정한다(재부팅 후).

② nis\_server\_enable="YES"

이 라인은 네트워크가 다시 로드 될 때 FreeBSD 가 NIS 서버 프로세스를 시작한다.

③ nis\_yppasswdd\_enable="YES"

이것은 위에서 언급한, 클라이언트 머신에서 사용자가 그들의 NIS 패스워드를 변경할 수 있게 하는 rpc.yppasswdd 데몬을 활성화한다.

**Note:** 여러분의 NIS 설정에 따라 더 많은 엔트리를 추가해야 된다. 아래의 NIS 클라이언트가 되기도 하는 NIS 서버에 대한 섹션에서 좀더 자세히 다룬다.

이제 슈퍼 유저에서 /etc/netstart 명령을 실행하면 된다. 이 명령은 /etc/rc.conf 에 지정된 값으로 필요한 모든 것을 설정한다.

## 2. NIS 맵 초기화

NIS 맵은 /var/yp 디렉터리에 저장된 데이터베이스 파일이다. 이들 파일은 /etc/master.passwd 파일을 제외하고 NIS 마스터의 /etc 디렉터리의 설정파일로부터 생성된다. root 와 다른 관리용 계정의 패스워드를 NIS 도메인에서 모든 서버로 전파시키지 않는 것이 보안상 안전하므로 NIS 맵을 초기화하기 전에 다음 내용을 따라야 한다:

```
# cp /etc/master.passwd /var/yp/master.passwd
# cd /var/yp
```

---

```
# vi master.passwd
```

NIS 클라이언트로 전파시키기를 원하지 않는 계정(예를 들어 root 와 UID 0(슈퍼 유저)인 다른 계정)과 모든 전체 시스템 계정(bin, tty, kemem, games 등)을 삭제한다.

**Note:** /var/yp/master.passwd 는 그룹이나 다른 사람들이 읽지 못하도록(모드 600) chmod 명령을 사용한다.

끝나고 나면 NIS 맵을 초기화한다! FreeBSD 는 초기화하기 위해 ypinit 라는 스크립트를 가지고 있다(더 많은 정보는 매뉴얼 페이지를 참고한다). 이 스크립트는 대부분의 유닉스 운영체제에서 사용할 수 있지만 모든 유닉스에서 사용하지는 못한다. Digital 유닉스/Compaq Tru64 유닉스는 ypsetup 이라고 한다. NIS 마스터 맵을 생성하는 것이므로 ypinit 에 옵션 *-m* 을 붙인다. NIS 맵을 생성하기 위해 위의 단계를 이미 수행한 걸로 간주하고 다음을 실행한다:

```
ellington# ypinit -m test-domain
Server Type: MASTER Domain: test-domain
Creating an YP server will require that you answer a few questions.
Questions will all be asked at the beginning of the procedure.
Do you want this procedure to quit on non-fatal errors? [y/n: n] n
Ok, please remember to go back and redo manually whatever fails.
If you don't, something might not work.
At this point, we have to construct a list of this domains YP servers.
rod.darktech.org is already known as master server.
Please continue to add any slave servers, one per line. When you are
done with the list, type a <control D>.
master server   : ellington
next host to add: coltrane
next host to add: ^D
The current list of NIS servers looks like this:
ellington
coltrane
Is this correct? [y/n: y] y

[..output from map generation..]
```

```
NIS Map update completed.
```

```
ellington has been setup as an YP master server without any errors.
```

ypinit 는 /var/yp/Makefile.dist 로부터 /var/yp/Makefile 를 생성한다. 이 파일은 오직 FreeBSD 머신으로 싱글 NIS 서버 환경을 운용한다고 간주한다. *test-domain* 은 슬레이브 서버도 가지고 있기 때문에 /var/yp/Makefile 를 수정해야 된다:

```
ellington# vi /var/yp/Makefile
```

다음 라인이 주석 처리 되어 있지 않다면 주석 처리해야 된다

```
NOPUSH = "True"
```

### 3. NIS 슬레이브 서버 설정

NIS 슬레이브 서버 설정은 마스터 서버 설정보다 간단하다. 슬레이브 서버에 로그인해서 이전처럼 /etc/rc.conf 를 수정한다. 유일한 한가지 차이점은 ypinit 를 실행할 때 *-s* 옵션을 사용하는 것이다. *-s* 옵션은 NIS 마스터의 이름이 필요하기 때문에 명령어 라인은 다음과 같다:

```
coltrane# ypinit -s ellington test-domain
```

```
Server Type: SLAVE Domain: test-domain Master: ellington
```

```
Creating an YP server will require that you answer a few questions.
```

```
Questions will all be asked at the beginning of the procedure.
```

```
Do you want this procedure to quit on non-fatal errors? [y/n: n] n
```

```
Ok, please remember to go back and redo manually whatever fails.
```

```
If you don't, something might not work.
```

```
There will be no further questions. The remainder of the procedure should take a few minutes, to copy the databases from ellington.
```

```
Transferring netgroup...
```

```
ypxfr: Exiting: Map successfully transferred
```

```
Transferring netgroup.byuser...
```

---

ypxfr: Exiting: Map successfully transferred  
Transferring netgroup.byhost...

ypxfr: Exiting: Map successfully transferred  
Transferring master.passwd.byuid...

ypxfr: Exiting: Map successfully transferred  
Transferring passwd.byuid...

ypxfr: Exiting: Map successfully transferred  
Transferring passwd.byname...

ypxfr: Exiting: Map successfully transferred  
Transferring group.bygid...

ypxfr: Exiting: Map successfully transferred  
Transferring group.byname...

ypxfr: Exiting: Map successfully transferred  
Transferring services.byname...

ypxfr: Exiting: Map successfully transferred  
Transferring rpc.bynumber...

ypxfr: Exiting: Map successfully transferred  
Transferring rpc.byname...

ypxfr: Exiting: Map successfully transferred  
Transferring protocols.byname...

ypxfr: Exiting: Map successfully transferred  
Transferring master.passwd.byname...

ypxfr: Exiting: Map successfully transferred  
Transferring networks.byname...

ypxfr: Exiting: Map successfully transferred  
Transferring networks.byaddr...

ypxfr: Exiting: Map successfully transferred  
Transferring netid.byname...

ypxfr: Exiting: Map successfully transferred  
Transferring hosts.byaddr...

ypxfr: Exiting: Map successfully transferred  
Transferring protocols.bynumber...

ypxfr: Exiting: Map successfully transferred  
Transferring ypservers...

ypxfr: Exiting: Map successfully transferred  
Transferring hosts.byname...

```
ypxfr: Exiting: Map successfully transferred
```

```
coltrane has been setup as an YP slave server without any errors.
```

```
Don't forget to update map ypservers on ellington.
```

이제 `/var/yp/test-domain` 이라는 디렉터리가 생성되었다. NIS 마스터 서버의 맵이 이 디렉터리에 복사되어 있다. 그리고 최근에 변경된 사항도 업데이트한다. 다음 라인들을 슬레이브 서버의 `/etc/crontab` 에 추가하여 항상 업데이트 할 수 있다:

```
20 * * * * root /usr/libexec/ypxfr passwd.byname
21 * * * * root /usr/libexec/ypxfr passwd.byuid
```

이 두 라인은 마스터 서버의 맵과 슬레이브 서버의 맵을 동기화한다. 패스워드 정보는 서버 시스템에 아주 중요하기 때문에 마스터 서버는 NIS 맵이 변경되었다면 슬레이브 서버와 통신을 시도한다. 이런 엔트리들이 필수적이지 않지만 강제로 업데이트하는 것도 좋은 생각이다. 이 방법은 맵 업데이트가 항상 완벽히 수행되지 않는 사용량이 많은 네트워크에서 더욱 중요하다.

이제 슬레이브 서버에서 NIS 서버를 다시 시작하는 `/etc/netstart` 명령을 수행한다.

### 23.4.4.3 NIS 클라이언트

NIS 클라이언트는 `ypbind` 데몬을 사용하여 특정 NIS 서버와 바인딩(결합)이라는 연결을 한다. `ypbind` 는 시스템의 기본 도메인(`domainname` 명령으로 설정된)을 체크하고 로컬 네트워크에서 RPC 요청 브로드캐스트를 시작한다. 이들 요청에는 `ypbind` 가 바인드 하려는 도메인 이름을 지정한다. 요청하고 있는 도메인을 지원하도록 설정된 서버가 브로드캐스트 패킷을 수신하여 응답하면 `ypbind` 는 서버의 주소를 기록한다. 여러 대의 서버(예를 들어 하나의 마스터와 여러 대의 슬레이브)가 응답하면 `ypbind` 는 첫 번째 서버의 주소를 사용한다. 그래서 클라이언트는 모든 NIS 요청을 이 서버로 지정한다. `ypbind` 는 종종 서버가 운용 중인지 "ping"을 날려본다. 특정 시간 동안 ping 에 대한 응답이 없다면 `ypbind` 는 응답이 없음을 표시하고 다른 서버를 향해 브로드캐스팅을 시작한다.

#### 23.4.4.3.1 NIS 클라이언트 설정

FreeBSD 머신을 NIS 클라이언트로 설정하는 것은 아주 직관적이다.





---

ypserv(8)은 지정된 호스트만 접근할 수 있도록 제한하는 securenets 기능을 지원한다. 시작할 때 ypserv(8)은 /var/yp/securenets 에서 securenets 정보를 읽는다.

**Note:** 이 경로 변경은 `-p` 옵션으로 지정된 경로에 의존된다. 이 파일에는 네트워크 설정과 공백으로 나누어진 네트워크 마스크로 이루어진 엔트리가 포함되어 있다. "#"으로 시작하는 라인은 주석이고 샘플 securenets 파일은 다음과 비슷할 것이다:

```
# allow connections from local host -- mandatory
127.0.0.1      255.255.255.255
# allow connections from any host
# on the 192.168.128.0 network
192.168.128.0 255.255.255.0
# allow connections from any host
# between 10.0.0.0 to 10.0.15.255
# this includes the machines in the testlab
10.0.0.0      255.255.240.0
```

이 룰에 맞는 주소로부터 요청을 받는다면 ypserv(8)은 보통 요청을 처리한다. 주소가 룰에 맞지 않는다면 이 요청은 무시되고 경고 메시지를 로그로 남긴다. /var/yp/securenets 파일이 없으면 ypserv 는 어떤 호스트의 연결이든 허용한다.

그리고 ypserv 프로그램은 Wietse Venema 의 **tcpwrapper** 패키지를 지원한다. 따라서 관리자는 접근 제어에 /var/yp/securenets 대신 **tcpwrapper** 설정 파일을 사용할 수 있다.

**Note:** 이 두 가지의 접근 제어 메커니즘은 이미 지정된 포트 테스트와 같은 몇 가지 보안을 제공하지만 "IP spoofing" 공격에 취약하다. 모든 NIS 관련 트래픽은 방화벽에서 막아야 한다.

/var/yp/securenets 를 사용하는 서버는 오래된 TCP/IP 를 사용하는 정당한 NIS 클라이언트를 지원하지 못할 것이다. 이들 중 어떤 것은 브로드캐스트 하거나 브로드캐스트 주소를 계산할 때 서브 넷 마스크 감지에 실패하면 모든 호스트 비트를 0으로 만든다. 이 문제 중 몇 가지는 클라이언트 설정을 변경해서 수정할 수 있지만 다른 문제는 클라이언트 시스템을 사용하지 못하거나 /var/yp/securenets 를 포기해야 될 것이다.

---

서버에서 오래된 TCP/IP 로 `/var/ypsecurenets` 를 사용하는 것은 아주 적절하지 못하고 거대한 네트워크에서 NIS 기능을 상실하는 원인이 된다.

`tcpwrapper` 패키지로 NIS 서버를 숨길 수 있다. 그러나 클라이언트 프로그램이 타임아웃이 걸릴 만큼 지연될 수 있고 특히 사용량이 많은 네트워크나 느린 NIS 서버에서 심각하다. 하나 이상의 클라이언트 시스템이 이런 증상을 가지고 있다면 클라이언트 시스템이 NIS 슬레이브 서버로 쿼리를 보내도록 바꾸고 이들을 강제로 바인드 한다.

### 23.4.6 로그인에서 특정 유저 제외시키기

연구실에 `basie` 라는 교원 전용의 워크스테이션 있다고 가정한다. 이 머신을 NIS 도메인에서 제외할 계획은 없고 마스터 NIS 서버의 `passwd` 파일에 교원과 학생의 계정이 모두 있다. 어떻게 하면 좋을 것인가?

NIS 데이터베이스에 계정이 있더라도 머신에 로그인할 수 없게 하는 방법이 있다. 이 방법은 클라이언트 머신의 `/etc/master.passwd` 파일 끝에 로그인을 허용하지 않는 유저의 이름 `-username` 을 추가한다. `vipw` 는 `/etc/master.passwd` 변경을 체크해주고 수정이 끝나면 자동으로 패스워드 데이터베이스를 다시 빌드하므로 `vipw` 를 사용하는 것이 좋다. 예를 들어 `basie` 에 로그인하는 유저 `bill` 이 로그인하지 못하게 하려면 다음 절차를 따른다:

```
basie# vipw
```

```
[add -bill to the end, exit]
```

```
vipw: rebuilding the database...
```

```
vipw: done
```

```
basie# cat /etc/master.passwd
```

```
root:[password]:0:0::0:0:The super-user:/root:/bin/csh
toor:[password]:0:0::0:0:The other super-user:/root:/bin/sh
daemon:*:1:1::0:0:Owner of many system processes:/root:/sbin/nologin
operator:*:2:5::0:0:System &:/sbin/nologin
bin:*:3:7::0:0:Binaries Commands and Source,,:/sbin/nologin
tty:*:4:65533::0:0:Tty Sandbox:/sbin/nologin
kmem:*:5:65533::0:0:KMem Sandbox:/sbin/nologin
games:*:7:13::0:0:Games pseudo-user:/usr/games:/sbin/nologin
news:*:8:8::0:0:News Subsystem:/sbin/nologin
man:*:9:9::0:0:Mister Man Pages:/usr/share/man:/sbin/nologin
bind:*:53:53::0:0:Bind Sandbox:/sbin/nologin
uucp:*:66:66::0:0:UUCP pseudo-
user:/var/spool/uucppublic:/usr/libexec/uucp/uucico
xten:*:67:67::0:0:X-10 daemon:/usr/local/xten:/sbin/nologin
pop:*:68:6::0:0:Post Office Owner:/nonexistent:/sbin/nologin
nobody:*:65534:65534::0:0:Unprivileged user:/nonexistent:/sbin/nologin
+:::
-bill
```

```
basie#
```

### 23.4.7 넷 그룹(Net Group) 사용

매우 적은 수의 유저나 머신에 특별한 룰이 필요하다면 이전 섹션에서 보여 준 방법으로 정확히 작동할 것이다. 거대한 네트워크에서 민감한 머신에 로그인하는 특정 유저를 거부하도록 설정하는 것을 잊어버리거나 각 머신을 따로 수정해야 된다면 NIS의 최대 장점인 중앙 관리를 못하게 되는 것이다.

NIS 개발자들은 이 문제를 *넷 그룹*이라고 부르는 방법으로 해결하였다. 이들의 목적과 의미는 유닉스 파일시스템이 사용하는 일반 그룹과 비교할 수 있다. 가장 큰 차이점은 숫자 id가 필요 없고 유저 계정과 다른 넷 그룹 포함하여 넷 그룹을 정의할 수 있다.

넷 그룹은 수백 명의 유저와 머신으로 거대하고 복잡한 네트워크를 제어하기 위해 개발되었다. 따라서 이런 상황이라면 좋은 해결책이 된다. 그러나 다른 한편으로 이 복잡성은 아주 간단한 예제로 넷 그룹을 설명하는 것을 거의 불가능하게 만든다. 이 섹션의 나머지에 사용되는 예제로 이 문제를 설명한다.

여러분이 NIS를 완벽하게 설명하여 다수의 사람들이 관심을 보였다고 가정한다. 다음에 할 작업은 캠퍼스의 다른 머신을 커버하도록 NIS 도메인을 확장하는 것이다. 다음 두 개의 테이블은 새로운 유저와 머신 이름을 포함하여 상황을 간략한 소개한다.

| 유저 이름                    | 설명              |
|--------------------------|-----------------|
| alpha, beta              | IT 학과의 일반적인 직원들 |
| charli, delta            | IT 학과의 새로운 견습생  |
| echo, foxtrott, golf,... | 일반적인 직원들        |
| able, baker,....         | 현재 인턴들          |

| 머신 이름                                  | 설명                                            |
|----------------------------------------|-----------------------------------------------|
| war, death, famine, pollution          | 가장 중요한 서버들. IT 직원들만 이들 머신에 로그인할 수 있다.         |
| pride, greed, envy, wrath, lust, sloth | 덜 중요한 서버들. IT 학과의 전원이 이들 머신에 로그인할 수 있다.       |
| one, two, three, four,...              | 일반적인 워크스테이션. 오직 실제 직원들만 이들 머신에 로그인할 수 있다.     |
| trashcan                               | 중요한 데이터가 없는 아주 오래된 머신. 인턴들도 이 머신들에 로그인할 수 있다. |

각 유저 별로 따로 제한한다면, 시스템에 로그인을 허용하지 않는 유저들은 각 시스템의 passwd에 *-user*로 추가한다. 하나의 엔트리라도 빼먹는다면 문제가 발생할 수 있다. 처음 설정할 때는 정확히 설정할 수 있겠지만 매일 매일 운용 중 새로운 유저를 위한 라인을 빼먹을 수 있다.

넷 그룹으로 이 상황을 제어하면 여러 가지 이점을 제공한다. 각 유저를 따로 제어할

---

필요가 없다; 유저를 하나 이상의 넷 그룹에 할당하고 넷 그룹 멤버 모두를 로그인 허용 또는 금지할 수 있다. 새로운 머신을 추가했다면 넷 그룹 로그인 제한만 정의하면 된다. 새로운 유저를 추가했다면 유저를 하나 이상의 넷 그룹에 추가하면 된다. 이러한 변경은 각자에게 독립적이다; 더 이상 유저와 머신이 연관되지 않는다. " 여러분의 NIS 설정이 세심하게 계획되었다면 머신에 접근을 허용하거나 거부하기 위해 하나의 중앙 설정파일만 수정하면 된다.

첫 번째 단계는 NIS 맵 넷 그룹을 초기화한다. FreeBSD의 ypinit(8)가 기본적으로 이 맵을 생성하지 않지만 맵이 생성된 후 ypinit의 NIS는 맵을 지원한다. 빈 맵을 생성하려면 단순히 다음 명령을 입력한다.

```
ellington# vi /var/yp/netgroup
```

그리고 목록을 추가한다. 우리의 예제에는 최소한 4개의 넷 그룹이 필요하다: IT 직원, IT 견습생, 일반 근로자와 인턴들.

```
IT_EMP  (.alpha,test-domain)  (.beta,test-domain)
IT_APP  (.charlie,test-domain) (.delta,test-domain)
USERS   (.echo,test-domain)   (.foxtrott,test-domain) *
        (.golf,test-domain)
INTERNS (.able,test-domain)   (.baker,test-domain)
```

*IT\_EMP*, *IT\_APP* 등은 넷 그룹의 이름이다. 각 중괄호 둘러싸서 하나 이상의 유저 계정을 그룹에 추가한다. 그룹 내의 3개의 필드는 다음과 같다:

- ① 다음 항목이 유효한 호스트의 이름. 호스트 이름을 지정하지 않았다면 이 엔트리는 모든 호스트에 유효하다. 호스트 이름을 지정했다면 어둠과 공포 그리고 완벽한 혼돈의 세계로 들어가는 것이다.
- ② 이 넷 그룹에 포함된 계정 이름.
- ③ 계정의 NIS 도메인. 여러분이 하나 이상의 NIS 도메인으로 불행한 사람이라면 다른 NIS 도메인에서 여러분의 넷 그룹으로 계정을 받아(import)올 수 있다.

각각의 이들 필드는 와일드카드(\*)를 포함할 수 있다. 자세한 사항은 netgroup(5)를 본다.

---

**Note:** NIS 도메인에 다른 운영체제도 같이 사용하고 있다면 넷 그룹 이름에 8 개 이상의 문자를 사용하지 않는다. 이름은 대 소문자를 구분한다; 넷 그룹 이름에 대문자를 사용하면 유저와 머신 그리고 넷 그룹 이름을 구분하기 쉽다.

어떤 NIS 클라이언트(FreeBSD 가 아닌)는 수많은 엔트리의 넷 그룹을 제어할 수 없다. 예를 들어 넷 그룹이 15 개 이상의 엔트리를 가지고 있으면 오래된 버전의 어떤 SunOS 가 문제를 유발한다. 15 유저 이하의 서브 넷 그룹 여러 개를 생성하여 이런 제한을 피할 수 있고 실제 넷 그룹이 서브 넷 그룹을 포함한다:

```
BIGGRP1 (.joe1, domain) (.joe2, domain) (.joe3, domain) [...]  
BIGGRP2 (.joe16, domain) (.joe17, domain) [...]  
BIGGRP3 (.joe31, domain) (.joe32, domain)  
BIGGROUP BIGGRP1 BIGGRP2 BIGGRP3
```

넷 그룹 하나에 225 이상의 유저가 필요하면 이 과정을 반복한다.

새로운 NIS 맵을 활성화하고 배포하는 것은 쉽다:

```
ellington# cd /var/yp  
ellington# make
```

다음 명령은 3 개의 NIS 맵 netgroup, netgroup.byhost 와 netgroup.byuser 를 생성한다. 새로운 NIS 맵을 이용할 수 있는지 ypcat(1)로 체크한다:

```
ellington% ypcat -k netgroup  
ellington% ypcat -k netgroup.byhost  
ellington% ypcat -k netgroup.byuser
```

첫 번째 명령의 결과는 /var/yp/netgroup 의 내용과 비슷해야 된다. 두 번째 명령은 호스트에 특정 넷 그룹을 지정하지 않았다면 출력 결과를 생성하지 않는다. 세 번째 명령은 넷 그룹의 유저 리스트를 가져오는데 사용할 수 있다.

클라이언트 설정은 상당히 간단하다. war 서버를 설정하려면 vipw(8)만 실행하고 다음 라인을 아래와 같이 변경한다.

---

```
+:::~::~:
```

```
+@IT_EMP:::~::~:
```

이제 넷 그룹 *IT\_EMP*에서 사용자가 정의한 데이터만 *war*의 패스워드 데이터베이스에 읽혀서 이들 유저만 로그인할 수 있다.

불행히 이 제한은 셸의 ~ 기능 그리고 유저 이름과 유저 ID 사이를 변환하는 모든 루틴에 적용된다. 다시 말해 `cd ~user`은 동작하지 않고 `ls -l`은 유저 이름 대신 숫자 id를 보여 주며 `find . -user joe -print`는 "No such user"와 같은 결과를 출력하고 실패한다. 이것을 해결하려면 서버에 로그인이 허락된 유저 엔트리를 제외한 모든 유저 엔트리를 읽어 들여야 된다.

이 문제는 `/etc/master.passwd`에 다른 라인을 추가해서 해결할 수 있다. 이 라인에는 다음 내용이 포함되어 있어야 된다:

```
+:::~::~:/sbin/nologin
```

이 의미는 "모든 엔트리를 읽어오지만 엔트리에서 `/sbin/nologin` 셸은 대체하지 않는다". `/etc/master.passwd`의 값을 기본값으로 해서 패스워드 엔트리의 모든 필드를 대체할 수 있다.

**주의:** `+:::~::~:/sbin/nologin`은 `+@IT_EMP:::~::~:` 뒤에 둔다. 그렇지 않으면 NIS에서 읽어온 모든 유저 계정의 로그인 셸은 `/sbin/nologin`이 된다.

이것을 변경한 후 새로운 직원이 IT 학부에 들어왔다면 NIS 맵 하나만 변경하면 된다. 좀더 덜 중요한 서버에서 `/etc/master.passwd`의 오래된 `+:::~::~:`를 다음과 같이 변경하여 비슷한 결과를 얻을 수 있다:

```
+@IT_EMP:::~::~:
```

```
+@IT_APP:::~::~:
```

```
+:::~::~:/sbin/nologin
```

일반적인 워크스테이션의 라인은 다음과 같을 것이다:



---

```
+@IT_EMP:.....  
+@USERS:.....  
+...../sbin/nologin
```

그리고 몇 주 후 정책이 변경될 때까지 모든 것이 정상이다: IT 학부에서 인턴 고용을 시작한다. IT 학과의 인턴들은 일반 워크스테이션과 덜 중요한 서버를 사용할 수 있다; 그리고 IT 수습생들은 메인 서버에 로그인할 수 있다. 새로운 넷 그룹 IT\_INTERN 을 추가하고 이 넷 그룹에 새로운 IT 인턴들을 추가하여 모든 머신의 설정을 변경하기 시작한다. 이전에 언급했듯이 “중앙 관리에서 예러는 전체적인 혼란을 초래 한다”.

다른 넷 그룹으로부터 새로운 넷 그룹을 생성하는 NIS 의 기능으로 이런 상황을 방지할 수 있다. 한가지 가능성은 역할별로 넷 그룹을 생성하는 것이다. 예를 들면 중요한 서버의 로그인 제한을 정의한 BIGSRV 라는 넷 그룹, 덜 중요한 서버는 MALLSRV 라는 다른 넷 그룹 그리고 보통 워크스테이션은 USERBOX 라고 부르는 세 번째 넷 그룹을 생성할 수 있다. 이들 넷 그룹 각각은 이들 머신에 로그인을 허용하는 넷 그룹을 포함한다. NIS 맵 넷 그룹의 새로운 엔트리는 다음과 비슷할 것이다:

```
BIGSRV   IT_EMP  IT_APP  
SMALLSRV IT_EMP  IT_APP  ITINTERN  
USERBOX  IT_EMP  ITINTERN USERS
```

이러한 방법의 로그인 제한은 동일한 제한이 필요한 머신 그룹을 정의할 수 있을 때 편리할 것이다. 불행히 이것은 룰이 아니고 예외다. 대부분 머신 별로 로그인 제한을 정의할 수 있는 능력이 필요하기 때문이다.

머신에 특별한 넷 그룹을 정의하는 것은 위의 개요를 변경하는 정책과 또 다른 문제를 유발할 수 있다. 이 시나리오에서 각 머신의 /etc/master.passwd 는 "+"로 시작하는 두 라인을 포함한다. 이들 중 첫 번째는 이 머신에 로그인할 수 있는 계정으로 넷 그룹을 추가하고 두 번째는 /sbin/nologin 셸을 가진 다른 모든 계정이다. 넷 그룹 이름처럼 머신 이름도 모두 대문자로 사용하는 것이 좋은 생각이다. 다시 말해 이 라인은 다음과 비슷할 것이다:

```
+@BOXNAME:.....  
+...../sbin/nologin
```

---

모든 머신에서 이 태스크를 끝내고 다시 로컬의 /etc/master.passwd 를 수정할 필요는 없다. 더 자세한 변경은 NIS 맵을 수정해서 제어할 수 있다. 여기 몇 개의 아이디어를 추가한 이 시나리오에 맞는 넷 그룹 맵 예제가 있다.

```
# Define groups of users first
IT_EMP    (,alpha,test-domain)    (,beta,test-domain)
IT_APP    (,charlie,test-domain)  (,delta,test-domain)
DEPT1     (,echo,test-domain)     (,foxtrott,test-domain)
DEPT2     (,golf,test-domain)     (,hotel,test-domain)
DEPT3     (,india,test-domain)    (,juliet,test-domain)
ITINTERN  (,kilo,test-domain)     (,lima,test-domain)
D_INTERNS (,able,test-domain)     (,baker,test-domain)
#
# Now, define some groups based on roles
USERS     DEPT1    DEPT2    DEPT3
BIGSRV    IT_EMP  IT_APP
SMALLSRV  IT_EMP  IT_APP  ITINTERN
USERBOX   IT_EMP  ITINTERN  USERS
#
# And a groups for a special tasks
# Allow echo and golf to access our anti-virus-machine
SECURITY  IT_EMP  (,echo,test-domain) (,golf,test-domain)
#
# machine-based netgroups
# Our main servers
WAR       BIGSRV
FAMINE    BIGSRV
# User india needs access to this server
POLLUTION BIGSRV (,india,test-domain)
#
# This one is really important and needs more access restrictions
DEATH     IT_EMP
#
# The anti-virus-machine mentioned above
ONE       SECURITY
#
```

---

```
# Restrict a machine to a single user
TWO      (,hotel,test-domain)
# [...more groups to follow]
```

유저 계정을 관리하기 위해 어떤 데이터베이스를 사용한다면 데이터베이스의 리포트 톨로 맵의 첫 번째 부분을 생성할 수 있을 것이다. 이 방법으로 새로운 유저는 자동으로 머신에 접근할 수 있다.

**경고:** 항상 머신 기반 넷 그룹을 사용하라고 권고할 수 없다. 20 대 또는 수백 대의 머신을 학생들 연구실에 배치한다면 적당한 제한에서 NIS 맵 크기를 유지하기 위해 머신 별 넷 그룹 대신 역할별 넷 그룹을 사용해야 된다.

## 23.4.8 기억해야 될 중요 사항

NIS 환경에서 따로 해야 될 일이 아직 존재한다.

- 연구실에 유저를 추가할 때 마스터 NIS 서버에만 추가하고 NIS 맵을 다시 빌드해야 된다. 이것을 잊는다면 새로운 유저는 NIS 마스터를 제외하고 어떤 곳에도 로그인할 수 없다. 예를 들어 실험실에 "jsmith"라는 새로운 유저를 추가해야 된다면 다음 명령을 입력한다:

```
# pw useradd jsmith
# cd /var/yp
# make test-domain
```

pw useradd jsmith 대신 adduser jsmith 를 실행할 수 있다.

- 관리용 계정을 NIS 맵에서 삭제한다. 관리자 계정에 접근할 필요가 없는 유저가 있는 머신에 관리자 계정과 패스워드가 전파되는 것을 원치 않을 것이다.
- NIS 마스터와 슬레이브의 보안을 유지하고 다운 시간을 최소화한다. 해커나 간단히 이들 머신의 전원을 끌 수 있는 사람이 있다면 이들은 효과적으로 수많은 사람들을 연구실에 로그인하지 못하게 할 수 있다.

이것이 중앙 관리 시스템 관리자의 문제다. NIS 서버를 보호하지 않는다면 수많은 유저의

---

야유를 받게 될 것이다.

## 23.4.9 NIS v1 호환

FreeBSD의 **ypserv**는 NIS v1 클라이언트를 부분적으로 지원한다. FreeBSD의 NIS는 오직 NIS v2 프로토콜만 사용하지만 v1 프로토콜을 포함한 다른 오래된 시스템을 지원한다. **ypbind** 데몬은 이들 시스템에 실제로 필요 없더라도 NIS v1 서버와 연결할 수 있도록 한다(그리고 이들이 v2 서버에서 응답을 받은 후에도 브로드캐스팅을 지속할 것이다). 보통 클라이언트의 요청이 지원되더라도 이 버전의 **ypserv**는 v1 맵 전송 요청을 지원하지 않는다; 따라서 오직 v1 프로토콜만 지원하는 오래된 NIS 서버가 있는 환경에 마스터나 슬레이브로 사용할 수 없다. 다행히 이런 서버는 요즘 사용하지 않을 것이다.

## 23.4.10 NIS 클라이언트가 되기도 하는 NIS 서버

서버 머신이 NIS 클라이언트도 되는 멀티 서버 도메인에서 **ypserv**를 실행할 때 주의한다. 이들 서버가 바인드 요청 브로드캐스트를 하도록 두지 않고 강제로 자신들에게 바인드 하도록 하는 것도 좋은 생각이고 가능할 것이다. 서버끼리 의존관계가 있어서 서버 하나가 다운되고 다른 서버들이 이 서버와 연관되어 있다면 이상한 실패 모드가 발생할 수 있다. 결국 모든 클라이언트가 타임아웃 되어 다른 서버와 바인드 하려고 하지만 상당히 지연된다. 그리고 서버들이 다시 각자 바인드 하기 때문에 서비스는 계속 사용할 수 없을 것이다.

-S 플래그로 **ypbind**를 실행해서 호스트를 특정 서버에 강제로 바인드 할 수 있다. NIS 서버를 재 부팅할 때마다 직접 입력하지 않으려면 `/etc/rc.conf`에 다음 라인을 입력할 수 있다:

```
nis_client_enable="YES" # run client stuff as well
nis_client_flags="-S NIS domain,server"
```

더 많은 정보는 `ypbind(8)`을 본다.

## 23.4.11 패스워드 포맷

사람들이 NIS를 실행하려고 할 때 가장 보편적인 문제는 패스워드 포맷 호환성이다.

---

여러분의 NIS 서버가 DES 암호화 패스워드를 사용한다면 DES 를 사용하는 클라이언트만 지원한다. 예를 들어 네트워크에 Solaris NIS 클라이언트가 있다면 대부분 DES 암호화 패스워드를 사용해야 된다.

서버와 클라이언트가 어떤 포맷을 사용하는지 체크하려면 `/etc/login.conf` 를 본다. 호스트가 DES 암호화 패스워드를 사용하도록 하려면 `default` 클래스는 다음과 같은 엔트리를 포함할 것이다:

```
default:W
    :passwd_format=des:W
    :copyright=/etc/COPYRIGHT:W
    [Further entries elided]
```

`passwd_format` 문자열에 포함할 수 있는 다른 값은 `blf`와 `md5`가 있다(각각 Blowfish 와 MD5 암호화 패스워드).

`/etc/login.conf` 를 변경하려면 `root` 로 다음 명령을 실행해서 로그인 기능 데이터베이스를 다시 빌드해야 된다:

```
# cap_mkdb /etc/login.conf
```

**Note:** `/etc/master.passwd` 의 패스워드 포맷은 로그인 기능 데이터베이스가 다시 빌드 된 후 유저가 자신들의 패스워드를 변경할 때까지 업데이트 되지 않는다.

패스워드가 여러분이 선택한 포맷으로 암호화되었는지 확인하려면 `/etc/auth.conf` 의 `crypt_default` 에 선택한 순서대로 패스워드 포맷이 있는지 확인한다. 패스워드 포맷을 선택하려면 리스트의 첫 번째에 원하는 포맷을 둔다. 예를 들어 DES 암호화 패스워드를 사용할 때 엔트리는 다음과 같다:

```
crypt_default = des blf md5
```

NIS 서버와 클라이언트 기반의 각 FreeBSD 에서 위의 단계를 따랐다면 여러분의 네트워크에서 어떤 패스워드 포맷을 사용하는지 서로 일치해야 된다. NIS 클라이언트에서 인증 문제가 발생한다면 패스워드 포맷 문제부터 분석을 시작한다. 다른 네트워크에 NIS 서버를 두려면 저 사양에서 일반적인 표준이기 때문에 DES 를 모든 시스템이 사용해야 될 것이다.

---

## 23.5 DHCP

### 23.5.1 DHCP 란 무엇인가?

동적 호스트 구성 프로토콜인 DHCP 는 시스템이 네트워크에 연결해서 네트워크와 통신에 필요한 정보를 받아올 수 있다는 의미다. FreeBSD 는 ISC(Internet Software Consortium) DHCP 를 실행하기 때문에 여기서 설명하는 것들은 ISC 배포본을 사용한다.

### 23.5.2 이 섹션은 무엇에 대해 설명하는가

이 섹션은 클라이언트단과 서버 단의 ISC DHCP 시스템 컴포넌트를 설명한다.

클라이언트단 프로그램 dhclient 는 FreeBSD 에 통합되었고 서버부분은 [net/isc-dhcp3-server](#) 포트에서 사용할 수 있다. dhclient(8), dhcp-options(5), dhclient.conf(5) 매뉴얼 페이지와 아래의 추가적인 레퍼런스에서 유용한 정보를 제공한다.

### 23.5.3 어떻게 동작하는가

DHCP 클라이언트 dhclient 가 클라이언트 머신에서 실행되면 설정 정보를 요청하기 위해 브로드캐스팅을 시작한다. 기본적으로 이들은 UDP 포트 68 번을 사용하여 요청한다. 그러면 서버는 UDP 67 로 클라이언트 IP 주소, 넷 마스크, 라우터 그리고 DNS 서버 같은 관련된 네트워크 정보를 되돌려준다. 이러한 모든 정보는 DHCP 형식으로 뿌려지고 일정시간 동안만(DHCP 서버 관리자가 설정한) 유효하다. 이런 식으로 더 이상 네트워크에 연결되어 있지 않은 클라이언트 IP 주소는 자동으로 회수된다.

DHCP 클라이언트는 서버로부터 여러 가지 정보를 받을 수 있다. 완벽한 리스트는 dhcp-option(5)에서 찾을 수 있을 것이다.

### 23.5.4 FreeBSD 통합

FreeBSD 는 ISC DHCP 클라이언트인 dhclient 와 완벽하게 통합되어있다. DHCP 클라이언트 지원은 인스톨러와 기본 시스템에서 제공하여 DHCP 서버가 사용되는 네트워크에서 자세한 네트워크 설정에 대한 지식이 필요 없게 된다. dhclient 는 FreeBSD 3.2 부터 모든 FreeBSD 에 포함되어 있다.

---

DHCP 는 `sysinstall` 에서 지원된다. `sysinstall` 로 네트워크 인터페이스를 설정할 때 첫 번째 질문이 “Do you want to try DHCP configuration of this interface?”이고, 긍정적인 대답(YES)은 `dhclient` 를 실행하여 성공하면 자동으로 네트워크 설정 정보를 채운다.

시스템이 시작될 때 DHCP 를 사용하려면 다음 두 가지를 설정해야 된다:

- `bpf` 장치가 커널에 컴파일 되어 있어야 한다. 컴파일은 커널 설정파일에 `device bpf` (FreeBSD 4.X에서는 `pseudo-device bpf`)을 추가하고 커널을 다시 빌드 한다. 커널 빌드에 대한 더 많은 정보는 8장을 본다.

`bpf` 장치는 FreeBSD GENERIC 커널에 포함되어 있으므로 사용자 커널을 가지고 있지 않다면 DHCP 작동을 위해 추가할 필요는 없다.

**Note:** 특히 보안에 민감하다면 `bpf` 도 패킷 스니퍼가 정확하게 동작하도록 하는 장치라는 것을 인식한다(root 로 실행해야 되더라도). 보안에 민감하다면 DHCP 를 정말 사용할 때까지 `bpf` 를 커널에 추가하면 안 된다.

- 다음 라인을 포함하도록 `/etc/rc.conf`를 수정한다:

```
ifconfig_fxp0="DHCP"
```

**Note:** 11 장에서 설명하였듯이 동적으로 설정하기를 원하는 인터페이스로 `fxp0` 를 변경한다.

`dhclient` 를 다른 위치에서 사용하거나 추가적인 플래그를 `dhclient` 에 적용한다면 다음 라인을 추가한다(필요하다면 수정한다):

```
dhcp_program="/sbin/dhclient"  
dhcp_flags=""
```

DHCP 서버 `dhcpcd` 는 포트 컬렉션의 `net/isc-dhcp3-server` 포트에 포함되어 있다. 이 포트에는 ISC DHCP 서버와 관련 문서가 있다.

---

## 23.5.5 파일

- `/etc/dhclient.conf`

`dhclient` 는 설정파일 `/etc/dhclient.conf` 가 필요하다. 보통 이 파일에는 설명만 있으며 일반적인 설정에 기본값이 적당하다. 이 설정 파일은 `dhclient.conf(5)` 매뉴얼 페이지에 설명되어 있다.

- `/sbin/dhclient`

`dhclient` 는 고정적으로 `/sbin` 안에 링크되어 있다. `dhclient(8)` 매뉴얼 페이지에서 `dhclient` 에 대한 더 많은 정보를 제공한다.

- `/sbin/dhclient-script`

`dhclient-script` 는 FreeBSD 에 맞는 DHCP 클라이언트 설정 스크립트다. `dhclient-script(8)`에 설명이 있지만 특별히 유저가 수정할 필요는 없다.

- `/var/db/dhclient.leases`

DHCP 클라이언트는 이 파일에 필요한 데이터베이스를 로그로 기록한다. `dhclient.leases(5)`에 좀더 자세한 설명이 있다.

## 23.5.6 더 많은 정보

DHCP 프로토콜은 RFC2131 에(<http://www.freesoft.org/CIE/RFC/2131/>) 완벽하게 설명되어 있다. 관련 정보는 `dhcp.org` 에(<http://www.dhcp.org/>) 게시되어 있다.

## 23.5.7 DHCP 서버 설치 및 설정



---

### 23.5.7.1 이 섹션에서 무엇을 설명하는가

이 섹션은 ISC(Internet Software Consortium) DHCP 스위트를 사용하여 FreeBSD 시스템을 어떻게 DHCP 서버로 동작하도록 설정하는지 설명한다.

서버 부분은 FreeBSD의 기본 시스템에서 제공하지 않기 때문에 이 서비스를 제공하려면 [net/isc-dhcp3-server](#) 포트를 설치해야 한다. 포트 컬렉션 사용에 대한 더 많은 정보는 4장을 본다.

### 23.5.7.2 DHCP 서버 설치

FreeBSD 시스템을 DHCP 서버로 설정하려면 bpf(4) 장치를 커널에 컴파일 해야 된다. 커널에 컴파일 하려면 *device bpf* (FreeBSD 4.X에서는 *pseudo-device bpf*)를 커널 설정 파일에 추가하고 커널을 다시 빌드 한다. 커널 빌드에 대한 더 많은 정보는 8장을 본다.

bpf 장치는 FreeBSD GENERIC 커널에 포함되어 있으므로 DHCP 동작을 위해 사용자 커널을 생성할 필요는 없다.

**Note:** 특히 보안에 민감하다면 bpf도 패킷 스니퍼가 정확하게 동작하도록 하는 장치라는 것을 인식한다(root로 실행해야 되더라도). 보안에 민감하다면 DHCP를 정말 사용할 때까지 bpf를 커널에 추가하면 안 된다.

다음에 해야 될 일은 [net/isc-dhcp3-server](#) 포트로 설치된 샘플 dhcpd.conf를 수정해야 된다. 기본적으로 이 파일은 /usr/local/etc/dhcpd.conf.sample 파일이고 변경하기 전에 이 파일을 /usr/local/etc/dhcpd.conf로 복사한다.

### 23.5.7.3 DHCP 서버 설정하기

dhcpd.conf는 서브넷과 호스트에 관한 선언을 포함하고 있으며, 예제를 이용하여 아주 쉽게 설명되어있다:

```

option domain-name "example.com"; ❶
option domain-name-servers 192.168.4.100; ❷
option subnet-mask 255.255.255.0; ❸

default-lease-time 3600; ❹
max-lease-time 86400; ❺
ddns-update-style none; ❻

subnet 192.168.4.0 netmask 255.255.255.0 {
    range 192.168.4.129 192.168.4.254; ❼
    option routers 192.168.4.1; ❽
}

host mailhost {
    hardware ethernet 02:03:04:05:06:07; ❾
    fixed-address mailhost.example.com; ❿
}

```

- ❶ 이 옵션은 기본 검색 도메인으로 클라이언트에 제공되는 도메인을 지정한다. 이 의미에 대한 더 많은 정보는 `resolv.conf(5)`를 본다.
- ❷ 이 옵션은 클라이언트가 사용해야 되는 DNS 서버 리스트를 쉼마(,)로 나누어 지정한다.
- ❸ 클라이언트에 제공할 넷 마스크
- ❹ 클라이언트는 유효한 임대기간으로 특정 기간을 요청할 것이다. 그렇지 않으면 서버는 이 만료 값을(초 단위) 임대 기간으로 할당한다.
- ❺ 이것은 서버가 제공하는 최대 임대기간이다. 오직 *max-lease-time* 초가 유효하지만 클라이언트가 더 오랜 임대기간을 요청해야 되기 때문에 문제가 된다.
- ❻ 이 옵션은 임대가 허용되거나 다시 임대될 때 DHCP 서버가 DNS 를 업데이트 할지 지정한다. ISC 에서 이 옵션이 *필요하다*.

- ⑦ 이것은 클라이언트에 할당하기 위해 남겨 둔 풀(pool)에서 어떤 IP 주소를 사용할지를 의미한다. 한 쪽은 시작되는 IP 주소고 다른 쪽은 끝나는 주소다.
- ⑧ 클라이언트에 공급할 기본 게이트웨이를 선언한다.
- ⑨ 호스트의 하드웨어 MAC 주소(그래서 DHCP 서버는 요청이 들어왔을 때 호스트를 인지할 수 있다).
- ⑩ 특정 호스트에게 항상 같은 IP 주소를 할당하도록 지정한다. 제공한 정보를 회수하기 전에 DHCP 서버는 호스트 이름을 분석하기 때문에 이곳에 호스트 이름도 사용할 수 있다.

dhcpd.conf 작성이 끝나면 다음 명령으로 서버를 시작할 수 있다:

```
# /usr/local/etc/rc.d/isc-dhcpd.sh start
```

나중에 서버 설정을 변경할 때 대부분의 데몬처럼 설정을 다시 로드하는 동안 **dhcpd** 는 *SIGHUP* 신호를 보내도 응답하지 않는다. *SIGTERM* 신호를 보내어 프로세스를 정지시키고 위의 명령을 사용하여 재 시작해야 된다.

#### 23.5.7.4 파일

- /usr/local/sbin/dhcpd

**dhcpd** 는 /usr/local/sbin 내에 정적으로 링크되어 있다. 포트로 설치된 dhcpd(8) 매뉴얼 페이지에서 dhcpd 에 대한 더 많은 정보를 제공한다.

- /usr/local/etc/dhcpd.conf

**dhcpd** 는 클라이언트에게 서비스를 제공하기 전에 /usr/local/etc/dhcpd.conf 파일의 설정이 필요하다. 이 파일은 서버 운용에 대한 정보와 함께 서비스를 받는 클라이언트에게 제공할 모든 정보를 포함해야 된다. 이 설정 파일은 포트로 설치된 dhcpd.conf(5) 매뉴얼 페이지에 설명되어 있다.

- 
- `/var/db/dhcpd.leases`

DHCP 서버는 로그처럼 작성된 이 파일에 필요한 데이터베이스를 가지고 있다. 포트로 설치한 `dhclient.leases(5)` 매뉴얼 페이지에 좀더 자세한 설명이 있다.

- `/usr/local/sbin/dhcrelay`

**dhcrelay** 는 하나의 DHCP 서버가 클라이언트의 요청을 다른 네트워크의 DHCP 서버에게 포워드하는 고급 환경에 사용된다. 이 기능이 필요하다면 [net/isc-dhcp3-server](#) 포트를 설치한다. 포트로 제공되는 `dhcrelay(8)` 매뉴얼 페이지에 더 자세한 정보가 있다.

## 23.6 DNS

### 23.6.1 요약

FreeBSD 는 아주 일반적인 DNS 프로토콜을 실행하는 BIND(Berkeley Internet Name Domain) 버전을 기본적으로 사용한다. DNS 는 이름을 IP 주소로 매핑시키는 프로토콜이다. 예를 들어 `www.FreeBSD.org` 로 쿼리를 보내면 FreeBSD 의 웹 서버의 IP 주소를 응답으로 받지만 `ftp.FreeBSD.org` 로 쿼리 하면 그에 맞는 FTP 머신의 IP 주소를 돌려준다. 이와 반대의 경우가 발생할 수 있다. IP 주소로 쿼리를 보내면 호스트 이름으로 해석된다. 시스템에서 DNS lookup 을 수행하려고 네임서버를 운용할 필요는 없다.

DNS 는 좀더 복잡한 시스템의 authoritative root 네임서버와 호스트 및 각각의 도메인 정보를 캐시하는 작은 규모의 네임서버와 인터넷을 통해 공조한다.

이 문서는 FreeBSD 에서 사용되는 안정 버전인 BIND 8.x 를 설명한다. FreeBSD 에서 BIND 9.x 는 [net/bind9](#) 포트로 설치할 수 있다.

RFC 1034 와 RFC 1035 에 DNS 프로토콜이 명시되어있다.

현재 BIND 는 인터넷 소프트웨어 컨소시엄에서([www.isc.org](http://www.isc.org)) 관리한다.

---

## 23.6.2 용어

이 문서를 이해하려면 DNS 와 관련된 몇 가지 용어를 이해해야 된다.

| 용어                | 정의                                                                           |
|-------------------|------------------------------------------------------------------------------|
| 순방향 DNS           | 호스트 이름을 IP 주소로 매핑                                                            |
| Origin            | 특정 존 파일로 커버하는 도메인을 의미한다.                                                     |
| named, BIND, 네임서버 | FreeBSD 에서 BIND 네임서버 패키지를 부르는 일반적인 이름.                                       |
| 리졸버               | 머신이 존 정보를 네임서버에 쿼리 하는 시스템 프로세스                                               |
| 역방향 DNS           | 순방향 DNS 의 반대; IP 주소를 호스트 이름으로 매핑                                             |
| root 존            | 인터넷 존 계층의 시작점. 파일 시스템에서 root 디렉터리 아래로 모든 파일이 있는 것과 비슷하게 모든 존은 root 존 아래에 있다. |
| 존                 | 개개의 도메인, 서브 도메인 또는 같은 권한을 가진 DNS 관할 영역                                       |

존 예제:

- .은 root 존이다.
- org.은 root 존 아래의 존이다.
- example.org는 org. 존의 아래 존이다.
- foo.example.org.은 example.org. 존 아래의 서브 도메인이다.
- 1.2.3.in-addr.arpa는 3.2.1.\* IP 공간 아래에 있는 모든 IP 주소를 의미하는 존이다.

위에서 보았듯이 더 명확한 호스트 이름 부분이 왼쪽에 나타난다. 예를 들어 example.org.은 org.보다 더 명확하고 org.은 root 존보다 더 명확하다. 각 호스트 이름의 레이아웃은 파일 시스템과 더 비슷하다: /dev 디렉터리는 root 아래에 있다.

## 23.6.3. 네임서버를 운용하는 이유

네임서버는 보통 두 종류가 있다: 권한을 가진 네임서버와 캐싱 네임서버.

---

권한을 가진 네임서버는 다음과 같은 경우에 필요하다:

- 쿼리에 대해 신뢰할 수 있는 DNS 정보를 전 세계에 제공하고자 할 때.
- example.org와 같은 도메인이 등록되어 있고 이 도메인 아래의 호스트 이름에 IP 주소를 할당해야 될 때.
- IP 주소 블록이 역방향 DNS 엔트리를 필요로 할 때(IP를 호스트 이름으로).
- 슬레이브라고 하는 백업 네임서버는 주 네임서버가 다운되거나 접근할 수 없을 때 쿼리에 응답해야 된다.

캐싱 네임서버는 다음과 같은 때 필요하다:

- 로컬 DNS 서버는 캐시 하여 외부의 네임서버에 쿼리 하는 것보다 더 빠르게 응답한다.
- 네트워크 전체의 트래픽을 감소해야 될 때(DNS 트래픽은 전체 인터넷 트래픽의 5% 정도를 감소시킨다.)

www.FreeBSD.org 를 쿼리 할 때 리졸버는 보통 위 단의 ISP 네임서버에 쿼리해서 응답을 전달한다. 캐싱 DNS 서버인 로컬 네임서버는 DNS 서버를 캐싱해서 외부에 한번만 쿼리 하면 된다. 정보가 로컬에 캐시되어 있기 때문에 추가적인 쿼리는 로컬 네트워크에서 해결된다.

## 23.6.4 어떻게 동작 하는가

FreeBSD 에서 BIND 데몬은 특별한 이유로 **named** 라고 한다.

| 파일    | 설명            |
|-------|---------------|
| named | BIND 데몬       |
| ndc   | 네임 데몬 제어 프로그램 |

---

|                        |                     |
|------------------------|---------------------|
| /etc/namedb            | BIND 존 정보가 위치한 디렉터리 |
| /etc/namedb/named.conf | 데몬 설정파일             |

존 파일은 보통 /etc/namedb 디렉터리에 있고 네임서버가 제공하는 DNS 존 정보를 가지고 있다.

## 23.6.5 BIND 시작

BIND 는 기본적으로 설치되기 때문에 모든 설정은 비교적 간단하다.

네임 데몬을 부팅할 때 시작하려면 /etc/rc.conf 를 다음과 같이 수정한다:

```
named_enable="YES"
```

데몬을 수동으로 시작하려면(설정을 한 후) 다음 명령을 사용한다.

```
# ndc start
```

## 23.6.6 설정 파일

### 23.6.6.1 make-localhost 사용

로컬 역방한 DNS 존 파일을 /etc/namedb/localhost.rev 에 정확하게 생성한다.

```
# cd /etc/namedb
# sh make-localhost
```

### 23.6.6.2 /etc/namedb/named.conf

---

```

// $FreeBSD$
//
// Refer to the named(8) manual page for details.  If you are ever going
// to setup a primary server, make sure you've understood the hairy
// details of how DNS is working.  Even with simple mistakes, you can
// break connectivity for affected parties, or cause huge amount of
// useless Internet traffic.

options {
    directory "/etc/namedb";

// In addition to the "forwarders" clause, you can force your name
// server to never initiate queries of its own, but always ask its
// forwarders only, by enabling the following line:
//
//     forward only;

// If you've got a DNS server around at your upstream provider, enter
// its IP address here, and enable the line below.  This will make you
// benefit from its cache, thus reduce overall DNS traffic in the
Internet.
/*
    forwarders {
        127.0.0.1;
    };
*/

```

주석문에서 설명하듯이 업 링크 캐시의 장점을 활용하도록 여기서 *forwarders*를 활성화할 수 있다. 일반적인 상황에서 네임서버는 찾고 있는 응답을 얻을 때까지 특정 네임서버에 반복적으로 쿼리를 보낸다. 이것이 활성화되면 업 링크 네임서버에(또는 제공된 네임서버) 처음으로 쿼리 해서 캐시의 장점을 얻을 수 있다. 업 링크 네임서버에 과부하가 걸렸다면 빠른 네임서버를 활성화하면 좋을 것이다.

**주의:** 127.0.0.1 은 여기서 동작하지 않는다. 이 IP 를 여러분의 업 링크 네임서버의 주소로 변경한다.



---

```
/*
 * If there is a firewall between you and name servers you want
 * to talk to, you might need to uncomment the query-source
 * directive below. Previous versions of BIND always asked
 * questions using port 53, but BIND 8.1 uses an unprivileged
 * port by default.
 */
// query-source address * port 53;

/*
 * If running in a sandbox, you may have to specify a different
 * location for the dumpfile.
 */
// dump-file "s/named_dump.db";
};

// Note: the following will be supported in a future release.
/*
host { any; } {
    topology {
        127.0.0.0/8;
    };
};
*/

// Setting up secondaries is way easier and the rough picture for this
// is explained below.
//
// If you enable a local name server, don't forget to enter 127.0.0.1
// into your /etc/resolv.conf so this server will be queried first.
// Also, make sure to enable it in /etc/rc.conf.

zone "." {
    type hint;
    file "named.root";
};
```



---

```
// chown bind:bind /etc/namedb/s
// chmod 750 /etc/namedb/s
```

샌드 박스에서 BIND 를 운용하는 방법에 대한 더 많은 정보는 샌드 박스에서 네임서버 운용하기를 본다([http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/handbook/network-dns.html#NETWORK-NAMED-SANDBOX](http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/network-dns.html#NETWORK-NAMED-SANDBOX))

```
/*
zone "example.com" {
    type slave;
    file "s/example.com.bak";
    masters {
        192.168.1.1;
    };
};

zone "0.168.192.in-addr.arpa" {
    type slave;
    file "s/0.168.192.in-addr.arpa.bak";
    masters {
        192.168.1.1;
    };
};
*/
```

named.conf 에서 이 설정은 순방향과 역방향 존의 슬레이브 엔트리 예제다.

새로 추가된 각 존의 새로운 존 엔트리를 named.conf 에 추가해야 한다.

예를 들어 example.org 의 가장 간단한 존 엔트리는 다음과 비슷할 수 있다:

```
zone "example.org" {
    type master;
    file "example.org";
};
```

---

*type* 구문에서 보여주듯이 이 존은 마스터다. 존 정보는 *file* 구문에서 보여주듯이 `/etc/namedb/example.org` 에 있다.

```
zone "example.org" {
    type slave;
    file "example.org";
};
```

슬레이브의 경우 존 정보는 마스터 서버의 특정 존을 전송해서 지정된 파일에 저장한다. 마스터 서버가 죽었거나 접속할 수 없을 슬레이브 네임서버는 전송된 존 정보를 제공할 수 있다.

### 23.6.6.3 존 파일

`example.org(/etc/namedb/example.org` 에 있는)의 마스터 존 예제는 다음과 같다:

```
$TTL 3600

example.org. IN SOA ns1.example.org. admin.example.org. (
                    5                ; Serial
                    10800             ; Refresh
                    3600              ; Retry
                    604800            ; Expire
                    86400 )          ; Minimum TTL

; DNS Servers
@           IN NS       ns1.example.org.
@           IN NS       ns2.example.org.

; Machine Names
localhost  IN A        127.0.0.1
ns1        IN A        3.2.1.2
ns2        IN A        3.2.1.3
mail       IN A        3.2.1.10
```

```

@           IN A      3.2.1.30

; Aliases
www        IN CNAME   @

; MX Record
@           IN MX     10      mail.example.org.

```

"."으로 끝나는 모든 호스트 이름은 정확한 호스트 이름이고, 반대로 끝에 "."이 없는 것은 Origin을 참조한다. 예를 들어 *www*는 *www + Origin*이다. 가상 존 파일에서 origin은 *example.org*기 때문에 *www*는 [www.example.org](http://www.example.org)로 변경된다.

존 파일 포맷은 다음과 같다:

```

recordname      IN recordtype  value

```

아주 일반적으로 사용되는 DNS 레코드:

|              |                             |
|--------------|-----------------------------|
| <b>SOA</b>   | 존의 권한 시작                    |
| <b>NS</b>    | 권한을 가진 네임서버                 |
| <b>A</b>     | 호스트 주소                      |
| <b>CNAME</b> | 엘리어스의 캐노니컬(canonical) 이름    |
| <b>MX</b>    | 메일 익스체인저                    |
| <b>PTR</b>   | 도메인 네임 포인터 (역방향 DNS 에 사용되는) |

```

example.org. IN SOA ns1.example.org. admin.example.org. (
                    5                ; Serial
                    10800             ; Refresh after 3 hours
                    3600              ; Retry after 1 hour
                    604800            ; Expire after 1 week
                    86400 )          ; Minimum TTL of 1 day

```

|                  |                          |
|------------------|--------------------------|
| example.org.     | 도메인 이름이면서 이 존 파일의 Origin |
| ns1.example.org. | 이 존에 대한 주/권한을 가진 네임서버    |

|                           |                                                                                                                                                                                                                               |
|---------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>admin.example.org.</i> | 이 존에 대한 책임이 있는 사람, “@의 메일주소가 .으로 대체된다. 따라서 <admin@example.org>는 admin.example.org 로 바뀐다.                                                                                                                                      |
| 5                         | 파일의 시리얼 번호. 이것은 존 파일이 수정될 때마다 증가해야 된다. 요즘 많은 관리자는 시리얼 번호로 <i>yyyymmddrr</i> 포맷을 선호한다. <i>2001041002</i> 는 2001년 04월 10일에 마지막으로 수정되었음을 의미하고, 마지막의 02는 이날 존 파일이 수정된 횟수를 의미한다. 시리얼 번호는 존이 업데이트 되었을 때 슬레이브 네임서버에게 알려 주기 때문에 중요하다. |

|   |       |                  |
|---|-------|------------------|
| @ | IN NS | ns1.example.org. |
|---|-------|------------------|

이것은 NS 엔트리다. 존에 대한 응답 권한을 가진 모든 네임서버는 이런 엔트리를 하나씩 가지고 있다. 여기서 본 @ 표시는 example.org.가 될 수 있다. @ 표시는 Origin 으로 변경된다.

|           |      |           |
|-----------|------|-----------|
| localhost | IN A | 127.0.0.1 |
| ns1       | IN A | 3.2.1.2   |
| ns2       | IN A | 3.2.1.3   |
| mail      | IN A | 3.2.1.10  |
| @         | IN A | 3.2.1.30  |

레코드는 머신 이름을 보여 준다. 위에서 보았듯이 ns1.example.org 는 3.2.1.2 로 해석된다. 역시 Origin 표시 @가 여기서도 사용되었기 때문에 example.org 의미는 3.2.1.30 으로 해석된다.

|     |          |   |
|-----|----------|---|
| www | IN CNAME | @ |
|-----|----------|---|

케노니컬 이름 레코드는 보통 머신에 엘리어스를 할당하는데 사용된다. 예제에서 www 는 origin 으로 표시된 머신이나 example.org(3.2.1.30)의 엘리어스다. CNAMES 는 호스트 이름의 엘리어스를 제공하거나 라운드 로빈의 여러 머신 중 호스트 이름 하나다.

|   |       |    |                   |
|---|-------|----|-------------------|
| @ | IN MX | 10 | mail.example.org. |
|---|-------|----|-------------------|

MX 레코드는 존으로 들어오는 메일을 어떤 메일 서버가 제어하는지 지정한다. mail.example.org 는 메일 서버의 호스트 이름이고 10 은 메일 서버의 우선순위다.

---

3, 2, 1의 우선순위로 여러 대의 메일 서버를 가지고 있을 수 있다. 메일 서버는 처음으로 example.org의 가장 높은 우선 순위의 MX에 전달을 시도하고 두 번째 우선 순위 등으로 메일이 정확하게 전달될 때까지 시도한다.

80.90.210.in-addr.arpa 존 파일에는(역방향 DNS) A 또는 CNAME 대신 PTR 엔트리를 사용하는 것을 제외하고 같은 포맷이 사용된다.

```
$TTL 3600
```

```
1.2.3.in-addr.arpa. IN SOA ns1.example.org. admin.example.org. (  
                    5                ; Serial  
                    10800            ; Refresh  
                    3600             ; Retry  
                    604800           ; Expire  
                    3600 )           ; Minimum
```

```
@      IN NS   ns1.example.org.
```

```
@      IN NS   ns2.example.org.
```

```
2      IN PTR ns1.example.org.
```

```
3      IN PTR ns2.example.org.
```

```
10     IN PTR mail.example.org.
```

```
30     IN PTR example.org.
```

이 파일은 적당한 IP 주소를 가상의 도메인 호스트 이름으로 매핑해 준다.

## 23.6.7 캐싱 네임서버

캐싱 네임서버는 존에 대한 권한이 없는 네임서버다. 단순히 쿼리하고 나중에 사용하기 위해 쿼리한 내용을 기억하고 있다. 설정은 존을 포함하지 않은 일반적인 네임서버로 설정한다.

---

## 23.6.8 Sandbox 에서 named 운용

named(8)의 보안을 강화하기 위해 특권이 없는 유저로 실행하고 sandbox 디렉터리로 chroot(8)하도록 설정한다. 이 설정은 sandbox 외부에서 **named** 데몬으로 모든 접근을 차단한다. 이렇게 해서 **named** 에서 발생할 피해를 줄이는데 도움이 된다. 기본적으로 **named** 를 실행하는 유저로 FreeBSD 는 bind 라는 유저와 그룹을 가지고 있다.

**Note:** 많은 사람들이 **named** 를 chroot 로 설정하는 대신 jail(8) 내부에서 **named** 를 운용하도록 권장한다. 이 섹션에서는 이 상황에 대해 다루지 않는다.

**named** 는 sandbox 외부로(공유 라이브러리와 로그 소켓 등) 모든 접근이 차단되기 때문에 **named** 기능이 정확하게 동작하도록 몇 개의 단계가 필요하다. 다음 체크 리스트에서 sandbox 의 경로는 /etc/named 이고 이 디렉터리의 내용을 수정하지 못하도록 설정하였다고 가정한다. root 로 다음 단계를 수행한다.

- **named**에 필요한 모든 디렉터리를 생성한다:

```
# cd /etc/namedb
# mkdir -p bin dev etc var/tmp var/run master slave
# chown bind:bind slave var/* ❶
```

❶ **named** 만 이들 디렉터리에 쓰기 권한이 필요하다.

- 기본 존과 설정파일을 다시 준비하고 생성한다:

```
# cp /etc/localtime etc ❷
# mv named.conf etc && ln -sf etc/named.conf
# mv named.root master
# sh make-localhost && mv localhost.rev localhost-v6.rev master
# cat > master/named.localhost
$ORIGIN localhost.
$TTL 6h
@ IN SOA localhost. postmaster.localhost. (
    1 ; serial
    3600 ; refresh
```



---

```
        1800      ; retry
        604800   ; expiration
        3600 )   ; minimum
IN NS localhost.
IN A          127.0.0.1
```

- ② 이 설정은 **named** 가 **syslogd(8)**에 정확한 시간으로 로그를 남기도록 한다.
- FreeBSD 4.9-RELEASE 이전 버전을 사용 중이라면 **named-xfer**를 정적으로 링크한 복사본을 빌드해서 **sandbox**에 복사한다:

```
# cd /usr/src/lib/libisc
# make cleandir && make cleandir && make depend && make all
# cd /usr/src/lib/libbind
# make cleandir && make cleandir && make depend && make all
# cd /usr/src/libexec/named-xfer
# make cleandir && make cleandir && make depend && make
NOSHARED=yes all
# cp named-xfer /etc/namedb/bin && chmod 555
/etc/namedb/bin/named-xfer ③
```

정적으로 링크한 **named-xfer** 가 설치된 후 소스 트리에 라이브러리나 프로그램의 필요 없는 복사본이 남지 않도록 설치된 것을 정리해야 된다:

```
# cd /usr/src/lib/libisc
# make cleandir
# cd /usr/src/lib/libbind
# make cleandir
# cd /usr/src/libexec/named-xfer
# make cleandir
```

- ③ 이 단계에서 문제가 발생한다고 종종 보고되었다. 문제가 발생하면 다음 명령을 사용한다:

```
# cd /usr/src && make cleandir && make cleandir
```

---

그리고 /usr/obj 트리를 삭제한다:

```
# rm -fr /usr/obj && mkdir /usr/obj
```

이렇게 소스 트리에서 "cruft"를 정리한 후 위의 단계를 다시 시도하면 정확히 동작한다.

FreeBSD 버전 4.9-RELEASE 나 이 후 버전을 사용한다면 named-xfer 을 /usr/libexec 에 복사하면 정적으로 링크가 되고, 이것을 sandbox 에 복사할 때 간단히 cp(1) 명령을 사용할 수 있다.

- named가 쓰기를 할 수 있도록 dev/null을 만든다:

```
# cd /etc/namedb/dev && mknod null c 2 2
# chmod 666 null
```

- /var/run/ndc를 /etc/namedb/var/run/ndc로 심볼릭 링크한다:

```
# ln -sf /etc/namedb/var/run/ndc /var/run/ndc
```

**Note:** 위 설정은 단순히 ndc(8)을 실행할 때마다 -c 옵션을 지정할 필요가 없게 한다. 부팅할 때 /var/run의 내용이 삭제되기 때문에 유용하다고 생각되면 @reboot 옵션을 사용하도록 이 명령을 root crontab에 추가할 수 있다. 이에 관한 더 많은 정보는 crontab(5)에서 볼 수 있다.

- named가 작성할 수 있는 추가적인 log 소켓을 생성하도록 syslogd(8)을 설정한다. 설정은 /etc/rc.conf의 syslogd\_flags 변수에 -l /etc/namedb/dev/log을 추가한다.
- named가 시작해서 sandbox로 chroot하도록 다음 라인을 /etc/rc.conf에 추가하여 준비한다:

```
named_enable="YES"
named_flags="-u bind -g bind -t /etc/namedb /etc/named.conf"
```

**Note:** 설정파일 /etc/named.conf에 sandbox와 관련된 절대경로를 표시한다. 예를 들면 위의 라인에서 실제파일은 /etc/namedb/etc/named.conf 임을



```

    type hint;
    file "master/named.root";
};
zone "private.example.net" in {
    type master;
    file "master/private.example.net.db";
    allow-transfer { 192.168.10.0/24; };
};
zone "10.168.192.in-addr.arpa" in {
    type slave;
    masters { 192.168.10.2; };
    file "slave/192.168.10.db"; ❹
};

```

- ❶ *directory* 구문은 **named** 에게 필요한 모든 파일이 이 디렉터리에 있다는 것을 지시하기 위해 /로 지정되어 있다(일반 유저의 /etc/namedb 와 동일함을 기억한다).
- ❷ **named-xfer** 바이너리의 절대경로 지정(**named** 의 프레임 레퍼런스에서). 이 설정은 기본적으로 **named** 가 /usr/libexec 에서 **named-xfer** 를 찾도록 컴파일 되어 있기 때문에 필요하다.
- ❸ **named** 가 이 존의 존 파일을 찾을 수 있는 파일이름을 지정한다(위의 *directory* 구문과 관련 있다).
- ❹ 존 파일이 마스터 서버에서 완벽하게 전송된 후 **named** 가 존 파일의 복사본을 작성하는 파일이름을 지정한다(위의 *directory* 구문과 관련 있다). 이것은 slave 디렉터리의 소유자를 위의 설정 단계에서 왜 bind 로 왜 바뀌어야 했는지를 보여준다.

위 단계를 마치고 서버를 재 부팅하거나 **syslogd(8)**을 재 시작한 후 **named(8)**을 시작하고 *syslogd\_flags* 와 *named\_flags* 에 지정한 새로운 옵션이 사용되는지 확인한다. 이제 **named** 를 복사한 sandbox 를 운용할 수 있다.

## 23.6.9 보안

BIND 가 DNS 로 가장 보편적으로 사용되고 있지만 항상 보안 문제가 발생했다. 따라서

---

앞으로도 취약점이 발견될 가능성은 충분하다.

CERT 의(<http://www.cert.org/>) 보안 권고를 읽고 현재 인터넷과 FreeBSD 보안 문제에 대한 최신 이슈를 접할 수 있도록 FreeBSD 보안 광고 메일링 리스트([http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/handbook/eresources.html#ERESOURCES-MAIL](http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/eresources.html#ERESOURCES-MAIL))에 가입하는 것도 좋은 생각이다.

**Tip:** 보안 문제가 발생하면 최신 소스를 받아서 **named** 를 새로 빌드하여 피해를 최소화할 수 있다.

## 23.6.10 더 많은 자료

BIND/named 매뉴얼 페이지: `ndc(8)`, `named(8)`, `named.conf(5)`

- 공식 ISC BIND 페이지(<http://www.isc.org/products/BIND/>).
- BIND FAQ(<http://www.nominum.com/getOpenSourceResource.php?id=6>).
- 오라일리 DNS와 BIND 4th 에디션 (<http://www.oreilly.com/catalog/dns4/>).
- RFC1034 - 도메인 네임 - 컨셉과 유용성(<ftp://ftp.isi.edu/in-notes/rfc1034.txt>).
- RFC1035 - 도메인 네임 - 틀과 설명서(<ftp://ftp.isi.edu/in-notes/rfc1035.txt>).

## 23.6.11 BIND9 과 FreeBSD

FreeBSD 5.3 릴리즈 버전에는 BIND9 DNS 서버 소프트웨어가 배포판에 포함되어있다. 이 소프트웨어에는 새로운 보안 기능과 파일 시스템 레이아웃 그리고 자동으로 생성된 `chroot(8)` 설정이 포함되어있다. 이 섹션은 두 개의 파트로 구성되어있으며, 첫 번째는 새로운 기능과 설정을 그리고 두 번째는 5.3 으로의 업그레이드를 돕는 내용을 작성하였다. 여기서 BIND9 서버는 단순히 `named(8)`로 언급한다. 이 섹션은 앞장에서 설명한 용어와 이론적인 설명은 제외한다. 따라서 더 읽기 전에 앞장을 참고하기 바란다.

`named` 설정 파일은 현재 `/var/named/etc/namedb/`에 있으며 수정이 필요하다.

## 24.7.1 마스터 존 설정

---

마스터 존을 설정하기 위해 /var/named/etc/namedb/에서 다음 명령을 실행한다:

```
# sh make-localhost
```

위 명령이 정상적으로 실행되면 새로운 파일이 master 디렉터리에 생성된다; 로컬 도메인 네임을 위한 파일 이름은 `localhost.rev` 이고 IPv6 설정을 위한 이름은 `localhost-v6.rev` 다. `named.conf` 파일에 이 파일들에 대한 설정이 있으므로 따로 수정할 필요는 없다.

### 24.7.2 슬레이브 존 설정

추가적인 도메인이나 서브 도메인 설정은 슬레이브 존으로 설정하면 될 것이다. 일반적으로 `master/localhost.rev` 파일을 `slave` 디렉터리에 복사한 후 수정한다. 파일을 수정한 후 `named.conf` 파일에도 아래 `example.com` 예제와 같이 추가해야 된다.

```
zone "example.com" {
    type slave;
    file "slave/example.com";
    masters {
        10.0.0.1;
    };
};

zone "0.168.192.in-addr.arpa" {
    type slave;
    file "slave/0.168.192.in-addr.arpa";
    masters {
        10.0.0.1;
    };
};
```

이 예제에서 마스터 IP(10.0.0.1)은 이 도메인에 존의 설정을 전해주는 주 도메인 서버의 IP 를 지정한다.

---

### 24.7.3 시스템 초기화 설정

시스템이 부팅할 때 **named** 데몬도 시작하도록 설정하려면 `rc.conf` 파일(`/etc/rc.conf`)에 다음 옵션을 입력한다.

```
named_enable="YES"
```

다른 옵션도 지정할 수 있지만 기본적으로 이 설정이면 된다. 이곳에 입력할 수 있는 다른 옵션은 `rc.conf(5)` 매뉴얼 페이지를 참고한다. `rc.conf` 파일에 아무것도 입력하지 않았다면 다음 명령을 실행하여 직접 **named** 데몬을 실행할 수 있다:

```
# /etc/rc.d/named start
```

### 24.7.4 BIND9 보안

FreeBSD 가 자동으로 **named** 를 `chroot(8)`에서 실행하지만 가능한 DNS 서비스 공격을 방지할 수 있는 다른 보안 메커니즘이 있다.

#### 24.7.4.1 쿼리 접근 제어 리스트

쿼리 접근 제어 리스트는 존에 대한 쿼리를 제한하는데 사용할 수 있다. 설정은 'Acl 토큰'에 지정한 내부 네트워크와 존 설정에 원하는 IP 주소를 지정하여 설정한다. 예제 호스트에 쿼리를 허용하는 도메인을 지정하려면 다음과 같이 설정한다:

```
acl "example.com" {
    192.168.0.0/24;
};

zone "example.com" {
    type slave;
    file "slave/example.com";
    masters {
        10.0.0.1;
```

```
};
allow-query { example.com; };
};

zone "0.168.192.in-addr.arpa" {
    type slave;
    file "slave/0.168.192.in-addr.arpa";
    masters {
        10.0.0.1;
    };
    allow-query { example.com; };
};
```

#### 24.7.4.2 버전 제한

DNS 서버에서 버전을 확인할 수 있도록 허용하는 것은 공격자에게 좋은 정보가 될 수 있다. 악의적인 유저는 서버를 해킹하기 위해 알려진 익스플로잇 (exploits)이나 버그를 찾는데 이 정보를 사용할 것이다.

**주의:** 버전 정보를 거짓으로 설정해도 익스플로잇 공격으로부터 서버를 보호하지 못한다. 취약성이 없는 버전으로 업그레이드하는 것만이 서버를 보호할 수 있다.

거짓 버전은 `named.conf` 의 'options' 섹션에 설정할 수 있다:

```
options {
    directory          "/etc/namedb";
    pid-file           "/var/run/named/pid";
    dump-file          "/var/dump/named_dump.db";
    statistics-file    "/var/stats/named.stats";
    version            "None of your business";
};
```



---

## 23.7 아파치 웹 서버

### 23.7.1 개요

FreeBSD 는 세계의 가장 유명한 웹 사이트 중 몇 곳을 운영하는데 사용된다. 인터넷에서 가장 많이 사용되는 웹 서버는 **아파치 웹 서버**다. **아파치** 소프트웨어 패키지는 FreeBSD 설치 미디어에 포함되어 있다 FreeBSD 를 처음 설치할 때 **아파치**를 설치하지 않았다면 [www/apache13](http://www.apache13) 또는 [www/apache2](http://www/apache2) 포트에서 설치할 수 있다.

**아파치**가 성공적으로 설치되면 설정을 해야 된다.

**Note:** 이 섹션은 FreeBSD에서 가장 많이 사용되는 **아파치 웹 서버 1.3.X** 버전에 대해 설명한다. 아파치 2.X가 새로운 기술을 많은 소개하겠지만 여기서는 설명하지 않는다. 아파치 2.X에 대한 더 많은 정보는 <http://httpd.apache.org/>를 본다.

### 23.7.2 설정

메인 아파치 웹 서버 설정파일은 FreeBSD 의 `/usr/local/etc/apache/httpd.conf` 에 설치된다. 이 파일은 # 문자로 시작되는 주석 라인이 있는 전형적인 유닉스 텍스트 설정파일이다. 가능한 모든 설정 옵션의 포괄적인 설명은 이 책의 범위를 벗어나기 때문에 여기서는 가장 일반적인 설정 방법을 설명한다.

**ServerRoot "/usr/local"**

이곳은 아파치 설치의 기본 디렉터리 구조를 지정한다. 바이너리는 서버 root(/usr/local 을 의미한다)의 bin 과/sbin 서브 디렉터리에 저장되고 설정파일은 etc/apache 에 저장된다.

**ServerAdmin [you@your.address](mailto:you@your.address)**

이 주소는 서버에 문제가 있을 때 메일을 보내는 곳이다. 이 주소는 에러 페이지처럼 서버가 생성하는 몇몇 페이지에 나타난다.

---

ServerName [www.example.com](http://www.example.com)

ServerName 는 설정한 호스트 이름이 다르다면(다시 말해 호스트의 실제 이름 대신 “www”를 사용) 서버의 클라이언트에게 보여주는 호스트 이름을 설정한다.

DocumentRoot `"/usr/local/www/data"`

DocumentRoot 는 여러분의 문서를 제공하는 디렉터리다. 기본적으로 모든 요청은 이 디렉터리에서 가져오지만 다른 위치를 지정하기 위해 심볼릭 링크와 앨리어스를 사용할 수 있다.

변경하기 전에 아파치 설정파일을 백업으로 복사해 두는 것이 좋다. 설정이 만족스럽다면 **아파치**를 시작할 준비가 되었다.

### 23.7.3 아파치 시작

**아파치**는 다른 네트워크 서버처럼 `inetd` 슈퍼 서버에서 실행되지 않는다. 클라이언트 웹 브라우저의 HTTP 요청을 더 빨리 처리하기 위해 `standalone` 로 실행하도록 설정한다. 셸 스크립트로 가능한 간단히 시작, 정지, 재 시작할 수 있다. **아파치**를 처음으로 시작하려면 다음 명령을 실행한다:

```
# /usr/local/sbin/apachectl start
```

다음 명령을 이용하여 언제든지 서버를 정지 시킬 수 있다:

```
# /usr/local/sbin/apachectl stop
```

어떤 이유로 설정파일을 변경하면 서버를 재 시작해야 된다:

```
# /usr/local/sbin/apachectl restart
```

시스템이 시작될 때 **아파치**를 실행시키려면 다음 라인을 `/etc/rc.conf` 에 추가 한다:

---

```
apache_enable="YES"
```

시스템이 부팅할 때 추가적인 명령어 라인 옵션을 아파치 httpd 프로그램에 지정하려면 rc.conf 에 추가적인 라인으로 옵션을 지정한다:

```
apache_flags=""
```

이제 웹 서버가 시작되고 웹 브라우저에 `http://localhost/`를 지정하여 웹 사이트를 볼 수 있다. 표시되는 기본 웹 페이지는 `/usr/local/www/data/index.html` 이다.

## 23.7.4 아파치 모듈

기본 서버에 기능을 추가하기 위해 수많은 아파치 모듈을 사용할 수 있다. FreeBSD 포트 컬렉션은 유명한 add-on 모듈을 **아파치**와 같이 설치하는 쉬운 방법을 제공한다.

### 23.7.4.1 mod\_ssl

`mod_ssl` 모듈은 Secure Sockets Layer(SSL v2/v3)와 Transport Layer Security(TLS v1) 프로토콜을 통해 강력한 암호화를 제공하기 위해 OpenSSL 라이브러리를 사용한다. 이 모듈은 신뢰되는 증명서를 제공하는 곳에 증명서를 요청하기 위해 필요한 모든 것을 제공하기 때문에 FreeBSD 에서 보안 웹 서버를 운용할 수 있다.

아직 아파치를 설치하지 않았다면 `mod_ssl` 이 포함된 **아파치** 버전을 [www/apache13-modssl](http://www.apache13-modssl) 포트로 설치할 수 있을 것이다.

### 23.7.4.2 mod\_perl

**아파치/펄** 통합 프로젝트는 강력한 펄 프로그래밍 언어와 아파치 웹 서버를 통합하고 있다. `mod_perl` 모듈로 전적으로 펄을 이용한 아파치 모듈을 작성할 수 있다. 게다가 서버에 내장된 인터프리터로 외부 인터프리터의 오버헤드와 펄을 시작할 때의 페널티(penalty)를 최소화할 수 있다.

---

아직 **아파치**를 설치하지 않았다면 `mod_perl`이 포함된 아파치 버전을 [www/apache13-modperl](#) 포트로 설치할 수 있을 것이다.

### 23.7.4.3 PHP

“PHP: Hypertext Preprocessor” PHP는 오픈 소스의 일반적인 스크립트 언어에 광범위하게 사용되고 특히 웹 개발과 HTML에 내장되는 곳에 적합하다. 구문이 C, Java 그리고 펄과 비슷하기 때문에 배우기 쉽다. 언어의 가장 주된 목적은 웹 개발자들이 동적으로 생성되는 웹 페이지를 빠르게 만들 수 있게 하지만 PHP로 더 많은 것을 할 수 있다.

PHP는 [lang/php5](#) 포트로 설치할 수 있을 것이다.

## 23.8 파일 전송 프로토콜 (FTP)

### 23.8.1 개요

파일 전송 프로토콜(FTP)은 FTP 서버로부터 유저에게 간편하게 파일을 전송하는 방법을 제공한다. FreeBSD는 기본 시스템에 FTP 서버 소프트웨어 `ftpd`가 포함되어 있다. 여기서는 FreeBSD에서 FTP 서버를 설치하고 관리하도록 아주 직선적으로 설명한다.

### 23.8.2 설정

가장 중요한 설정 단계는 FTP 서버를 어떤 계정이 사용할 수 있도록 할 것인지 선택하는 것이다. 일반적인 FreeBSD 시스템은 다양한 데몬에 사용하는 수많은 시스템 계정을 가지고 있지만 `unknown` 유저가 이들 계정에 로그인할 수 없어야 된다. `/etc/ftpusers` 파일은 FTP를 사용할 수 없는 유저를 나열하고 있다. 기본적으로 앞에서 말한 시스템 계정이 포함되어 있지만 FTP를 사용하지 말아야 되는 유저를 이곳에 추가할 수도 있다.

어떤 유저의 FTP 사용을 완전히 막지 않고 제한하고 싶을 것이다. 이것은 `/etc/ftpchroot` 파일로 설정할 수 있다. 이 파일에 FTP 사용을 제한하려는 유저와 그룹을 나열한다. `ftpchroot(5)` 매뉴얼 페이지에 자세한 설명이 있기 때문에 여기서는 더 이상 설명하지 않는다.

---

여러분의 서버에 익명 FTP 사용을 허용하려면 FreeBSD 시스템에 ftp 라는 이름의 유저를 생성해야 된다. 그러면 유저는 유저 이름 ftp 나 anonymous 와 패스워드(전통적으로 패스워드에 유저의 전자 메일 주소를 사용한다)로 여러분의 FTP 서버에 로그인할 수 있다. 익명 유저가 로그인하면 ftp 유저의 홈 디렉터리만 사용하도록 제한하기 위해 FTP 서버는 chroot(2)를 호출한다.

FTP 클라이언트에 보여주기 위해 환영 메시지를 입력할 수 있는 두 개의 텍스트 파일이 있다. /etc/ftpwelcom 파일의 내용이 로그인 프롬프트가 나타나기 전에 유저들에게 보여진다. 로그인에 성공하면 /etc/ftpmotd 파일의 내용을 보여준다. 이 파일 경로는 로그인 환경과 상대적이기 때문에 익명 유저에게 ~ftp/etc/ftpmotd 가 보여진다.

FTP 서버를 정확히 설정한 후 /etc/inetd.conf 를 활성화해야 된다. 활성화하는 방법은 단순히 ftpd 라인 앞부분의 # 표시를 삭제한다:

```
ftp    stream  tcp nowait  root    /usr/libexec/ftpd  ftpd -
```

예제 23-1 에서 설명한 것처럼 이 설정파일을 수정한 후 inetd 에게 설정파일을 다시 읽도록 신호를 보내야 된다.

다음과 같이 입력하여 FTP 서버에 로그인할 수 있다:

```
% ftp localhost
```

### 23.8.3 관리

ftpd 데몬은 로그 메시지에 syslog(3)를 사용한다. 기본적으로 시스템 로그 데몬은 /var/log/xferlog 파일의 FTP 관련된 위치에 메시지를 저장한다. FTP 로그는 /etc/syslog.conf 파일의 다음 라인을 변경하여 수정할 수 있다:

```
ftp.info    /var/log/xferlog
```

익명 FTP 서버를 운영하면 중요한 문제가 생길 수 있다는 것을 알고 있어야 된다. 특히 익명 유저가 파일을 업로드 할 수 있도록 하는 것에 대해 다시 생각해 본다. 여러분의 FTP 사이트가 불법 소프트웨어의 거래 장소가 되는 것을 알게 될 것이다. FTP 업로드를 허용할

---

필요가 있다면 퍼미션을 설정하여 다른 익명 유저들이 이들 파일을 볼 수 없도록 한다.

## 23.9 Microsoft® Windows® 클라이언트를 위한 파일과 프린트 서비스 (Samba)

### 23.9.1 개요

**삼바**는 마이크로소프트 윈도우 클라이언트에 파일과 프린트 서비스를 제공하는 유명한 오픈 소스 소프트웨어 패키지다. 이러한 클라이언트는 FreeBSD 의 로컬 디스크 드라이브나 로컬 프린터에 연결하여 사용할 수 있다.

**삼바** 소프트웨어 패키지는 FreeBSD 설치 미디어에 포함되어 있다. FreeBSD 를 처음 설치할 때 삼바를 설치하지 않았다면 [net/samba3](#) 포트 패키지로 설치할 수 있다.

### 23.9.2 설정

기본 삼바 설정 파일은 `/usr/local/etc/smb.conf.default` 에 설치된다. 이 파일을 `/usr/local/etc/smb.conf` 에 복사하여 삼바를 사용하기 전에 수정한다.

`smb.conf` 파일은 프린터와 윈도우 클라이언트에 공유하려는 “파일시스템” 정의와 같은 **삼바**의 런타임 설정정보를 포함하고 있다. **삼바** 패키지는 `smb.conf` 파일을 쉽게 설정할 수 있는 **swat** 라는 웹 기반 툴을 포함하고 있다.

#### 23.9.2.1 삼바 웹 기반 툴 사용 (SWAT)

삼바 웹 기반 툴(SWAT)은 `inetd` 데몬으로 실행된다. 따라서 삼바 설정에 `swat` 을 사용하기 전에 `/etc/inetd.conf` 에서 다음 라인의 주석을 풀어야 된다:

```
swat stream tcp nowait/400 root /usr/local/sbin/swat
```

예제 23-1 에서 설명하듯이 이 설정파일을 수정한 후 `inetd` 가 설정파일을 다시 읽도록

---

해야 된다.

inetd.conf에서 **swat**이 활성화되면 브라우저에 <http://localhost:901> 을 입력하여 연결할 수 있다. 처음엔 시스템 root 계정으로 로그인해야 된다.

메인 삼바 설정 페이지에 성공적으로 로그인한 후 시스템 문서를 탐색하거나 'Globals' 탭을 클릭하여 설정을 시작할 수 있다. 이 Globals 섹션은 /usr/local/etc/smb.conf 의 *[global]*에 설정된 변수와 일치한다.

### 23.9.2.2 Global 설정

**swat** 를 사용하거나 /usr/local/etc/smb.conf 를 직접 수정하더라도 **삼바**를 설정할 때 마주치게 될 첫 번째 설정은 다음과 같다:

#### workgroup

이 서버를 사용할 컴퓨터의 NT 도메인 이름이나 Workgroup 이름 설정.

#### netbios name

삼바 서버가 알려질 NetBIOS 이름을 설정한다. 기본적으로 호스트 DNS 이름의 첫 번째 컴포넌트와 같다.

#### server string

*net view* 명령과 서버에 대한 설명을 찾는 다른 네트워크 툴에 표시되는 문자열을 설정한다.

### 23.9.2.3 보안 설정

/usr/local/etc/smb.conf 에서 가장 중요한 두 가지 설정은 보안 모델 선택과 클라이언트 유저의 패스워드 포맷이다. 다음 설정이 이들 옵션을 제어한다:

---

## security

여기서 가장 일반적인 두 가지 옵션은 *security = share* 와 *security = user* 다. FreeBSD 머신의 유저이름과 같은 유저 이름을 클라이언트가 사용한다면 user 레벨 보안을 사용하려고 할 것이다. 이것이 기본 보안 정책이고 공유된 자원에 접근하기 전에 클라이언트는 로그인에 필요하다.

share 레벨 보안에서 공유된 자원에 연결하기 전에 클라이언트는 유효한 유저 이름과 패스워드로 서버에 로그인할 필요가 없다. 이 설정이 예전 **삼바** 버전의 기본 보안 모델이다.

## passwd backed

**삼바**는 여러 개의 backend 인증 모델을 가지고 있다. LDAP, NIS+, SQL 데이터베이스나 수정된 패스워드 파일로 클라이언트를 인증할 수 있다. 기본 인증 방법은 *smbpasswd* 이고 여기서 설명한다.

*smbpasswd*를 사용한다면 **삼바**가 클라이언트를 인증할 수 있도록 `/usr/local/private/smbpasswd` 파일을 생성해야 된다. 모든 유닉스 유저 계정들에게 접근 권한을 주려면 다음 명령을 사용한다:

```
# cat /etc/passwd | grep -v "^#" | make_smbpasswd > /usr/local/private/smbpasswd
# chmod 600 /usr/local/private/smbpasswd
```

설정 옵션에 대한 추가적인 정보는 **삼바** 문서를 확인해 본다. 이곳의 기본적인 개요에서 **삼바**를 실행하기 위해 필요한 모든 것을 얻을 수 있다.

## 23.9.3 삼바 시작

시스템이 부팅할 때 삼바를 시작하려면 `/etc/rc.conf` 에 다음 라인을 추가한다:

```
samba_enable="YES"
```



---

다음 명령을 실행하여 언제든지 **삼바**를 시작할 수 있다:

```
# /usr/local/etc/rc.d/samba.sh start
Starting SAMBA: removing stale tdb's :
Starting nmbd.
Starting smbd.
```

**삼바**는 사실 3 개의 데몬으로 구성되어 있다. **nmbd** 와 **smbd** 데몬은 samba.sh 스크립트가 시작한다. smb.conf 에서 winbind name resolution 서비스를 활성화했다면 winbind 데몬도 시작되는 것을 볼 수 있다.

다음 명령으로 언제든지 **삼바**를 정지 시킬 수 있다:

```
# /usr/local/etc/rc.d/samba.sh stop
```

**삼바**는 마이크로소프트 윈도우 네트워크와 기능적으로 통합되는 복잡한 소프트웨어다. 여기서 설명하는 기본 설치 이상의 기능에 대한 더 많은 정보는 <http://www.samba.org>를 확인한다.

## 23.10 NTP

### 23.10.1 개요

시간이 지나면 컴퓨터의 시간은 정확성이 떨어진다. NTP(네트워크 시간 프로토콜)는 시간을 정확하게 하는 한가지 방법이다.

많은 인터넷 서비스는 컴퓨터 시간과 연관성이 있고 시간이 정확해야 이점을 얻을 수 있다. 예를 들어 웹 서버는 파일이 특정 시간 이후에 수정되었다면 이 파일을 보내달라는 요청을 받을 것이다. cron(8)과 같은 서비스는 주어진 시간에 명령을 수행한다. 시간이 맞지 않는다면 이들 명령은 예정된 시간에 실행되지 않는다.

FreeBSD 는 머신의 시간을 맞추기 위해 다른 NTP 서버에 쿼리를 보내거나 다른 머신에 시간 서비스를 제공하는데 사용되는 NTP 서버 ntpd(8)을 가지고 있다.

---

## 23.10.2 적당한 NTP 서버 선택

시간을 동기화하려면 한대 또는 한대 이상의 NTP 서버를 찾아야 한다. 여러분의 네트워크 관리자나 ISP가 이런 목적으로 NTP 서버를 설정하였을 것이다 -- 이러한 경우 그들이 제공하는 문서를 확인한다. 여러분에게 가까운 NTP 서버를 찾아서 공적으로 사용할 수 있는 NTP 서버 리스트가 있다(<http://www.eecis.udel.edu/~mills/ntp/servers.html>). 선택한 서버의 정책을 인지하고 필요한 경우 허락을 얻는다.

여러분이 사용하던 서버를 사용할 수 없거나 시간이 부정확할 수 있기 때문에 연결이 되지 않은 여러 개의 NTP 서버를 선택하는 것도 좋은 생각이다. ntpd(8)은 다른 서버의 응답을 지능적으로 받는다 -- 이것은 안정된 서버보다 불안정한 서버에 더 유용하다.

## 23.10.3 머신 설정

### 23.10.3.1 기본 설정

머신이 부팅할 때 시간을 동기화하려면 ntpdate(8)을 사용할 수 있다. 이것은 자주 재부팅하고 가끔 동기화가 필요한 몇몇 데스크톱 머신에는 적당하지만 대부분의 머신은 ntpd(8)을 사용하는 것이 좋다.

부팅할 때 ntpdate(8)을 사용하는 것은 ntpd(8)을 운용하는 머신에도 좋은 생각이다. ntpd(8)은 시간을 점차적으로 변경하지만 ntpdate(8)은 시간을 설정한다. 따라서 현재 머신에 설정된 시간이 정확한 시간과 얼마나 큰 차이가 있는지 문제되지 않는다.

부팅할 때 ntpdate(8)을 활성화하려면 `/etc/rc.conf` 에 `ntpdate_enable="YES"`를 추가한다. 그리고 동기화하려는 모든 서버를 ntpdate(8)에 적용할 플래그와 함께 `ntpdate_flags`에 지정해야 된다.

### 23.10.3.2 일반적인 설정

NTP는 `ntp.conf(5)`에 설명된 포맷으로 `/etc/ntp.conf`에 설정한다. 여기 샘플 예제가 있다:

```
server ntplocal.example.com prefer
server timeserver.example.org
server ntp2a.example.net

driftfile /var/db/ntp.drift
```

*server* 옵션으로 어떤 서버를 사용할지 각 라인에 서버 한대씩 지정한다. 서버가 ntplocal.example.com 과 같이 prefer 인자로 지정되어 있다면 이 서버는 다른 서버보다 우선권이 있다. 우선권이 있는 서버의 응답이 다른 서버의 응답과 많은 차이가 있다면 폐기되고 그렇지 않으면 다른 응답을 고려하지 않고 사용된다. *prefer* 인자는 보통 특별한 시간 모니터링 하드웨어처럼 아주 정확하다고 알려진 NTP 서버에 사용된다.

*driftfile* 옵션은 시스템 시간의 빈번한 옵셋을 저장할 파일을 지정한다. ntpd(8)은 시간이 맞지 않는 것을 자동으로 보상하는데 이 옵션을 사용하고, 외부의 모든 시간이 일정 시간 동안 차단되더라도 적절히 정확한 설정을 유지한다.

*driftfile* 옵션은 사용중인 NTP 서버의 기존 응답에 대한 정보를 저장할 파일을 지정한다. 이 파일은 NTP의 내적인 정보를 포함한다. 다른 프로세스가 이 파일을 수정하면 안 된다.

### 23.10.3.3 서버 접근 제어

기본적으로 NTP 서버는 인터넷의 모든 호스트가 접근할 수 있다. /etc/ntp.conf의 *restrict* 옵션으로 서버에 접근할 머신을 제어할 수 있다.

NTP 서버에 모든 머신의 접근을 거부하려면 다음 라인을 /etc/ntp.conf에 추가한다:

```
restrict default ignore
```

여러분의 네트워크에 있는 머신만 서버로 시간을 동기화하고, 서버를 설정하거나 동기화를 거부하지 않을 것이라면 다음 라인을 추가한다:

```
restrict 192.168.1.0 mask 255.255.255.0 notrust nomodify notrap
```

192.168.1.0은 여러분의 네트워크 주소고 255.255.255.0는 넷 마스크다.

---

`/etc/ntp.conf` 에 여러 개의 *restrict* 옵션을 지정할 수 있다. 더 자세한 사항은 `ntp.conf(5)`의 접근제어 지원에 대한 부분을 본다.

## 23.10.4 NTP 서버 운용

부팅할 때 NTP 서버를 시작하려면 `xntpd_enable="YES"` 라인을 `/etc/rc.conf` 에 추가한다. 추가적인 플래그를 `ntpd(8)`에 적용하려면 `/etc/rc.conf` 의 `xntpd_flags` 매개변수를 수정한다.

머신을 재 부팅하지 않고 서버를 시작하려면 `/etc/rc.conf` 의 `xntpd_flags` 에 부가적인 매개변수를 지정하고 `ntpd` 를 실행한다. 예를 들면 다음과 같이 실행한다:

```
# ntpd -p /var/run/ntpd.pid
```

**Note:** FreeBSD 5.X 에서 `/etc/rc.conf` 의 여러 가지 옵션 이름이 바뀌었다. 따라서 `xntpd`의 모든 인스턴스를 위 옵션의 `ntpd`로 대체한다.

## 23.10.5 일시적인 인터넷 연결에 ntpd(8) 사용

`ntpd` 의 적절한 기능 때문에 인터넷에 계속적으로 연결할 필요가 없다. 그러나 요청에 의해 전화접속으로 일시적으로 연결한다면 전화 걸기를 시작하거나 연결을 유지하는 동안 NTP 트래픽을 방지하는 것도 좋은 생각이다. 유저 PPP 를 사용한다면 `/etc/ppp/ppp.conf` 에 *filter* 기능을 사용할 수 있다. 예를 들면 다음과 같이 설정할 수 있다:

```
set filter dial 0 deny udp src eq 123
# Prevent NTP traffic from initiating dial out
set filter dial 1 permit 0 0
set filter alive 0 deny udp src eq 123
# Prevent incoming NTP traffic from keeping the connection open
set filter alive 1 deny udp dst eq 123
# Prevent outgoing NTP traffic from keeping the connection open
set filter alive 2 permit 0/0 0/0
```

더 자세한 사항은 `ppp(8)`의 패킷 필터링 섹션과 `/usr/share/examples/ppp/`의 예제를

---

확인한다.

**Note:** 어떤 ISP 는 미리 정의되어 있는 포트번호를 막아서 응답이 머신에 도달하지 못하므로 NTP 기능을 사용하지 못한다.

## 23.10.6 더 많은 정보

NTP 서버 문서는 /usr/share/doc/ntp/에서 HTML 문서를 찾을 수 있다.

---

## 24 장 고급 네트워킹

### 24.1 개요

이번 장에서는 고급 네트워킹에 대해 다룬다.

이번 장을 읽고 아래와 같은 사항을 알 수 있다:

- 게이트웨이와 라우트의 기본
- IEEE802.11과 블루투스 장치는 어떻게 설정하는가
- FreeBSD를 어떻게 브리지처럼 동작 시키는가
- diskless 머신이 네트워킹으로 부팅하도록 어떻게 설정하는가
- 네트워크 주소 변환(NAT)은 어떻게 설정하는가
- PLP로 두 대의 컴퓨터를 어떻게 연결하는가
- FreeBSD 머신에서 IPv6 설정
- FreeBSD 5.X로 ATM 설정

이번 장을 읽기 전에 알고 있어야 할 사항:

- /etc/rc 스크립트의 기본을 알고 있어야 된다.
- 기본 네트워크 용어에 익숙해야 된다.
- 새로운 FreeBSD 커널 설정과 설치 (8장)
- 소프트웨어 설치 (4장)

---

## 24.2 게이트웨이와 라우트(경로)

네트워크를 통해서 머신이 다른 머신을 찾을 수 있도록, 한 머신에서 다른 머신을 어떻게 찾는지 설명된 메커니즘이 있어야 된다. 이것을 라우팅(경로) 이라고 한다. "경로"는 목적지와 게이트웨이 주소 쌍으로 정의되어 있다. 이 주소 쌍은 여러분이 특정 목적지에 접속한다면 이 *게이트웨이*를 경유하여 통신하는 것을 보여준다. 목적지에는 각각의 호스트와 서브 넷 그리고 "기본 라우트" 이렇게 3 종류가 있다. "기본 라우트(경로)"는 적용할 다른 경로가 없을 때 사용된다. 나중에 기본 라우트에 대해 좀더 이야기할 것이다. 그리고 게이트웨이에도 역시 3 종류가 있다: 각각의 호스트와 인터페이스("링크"라고 부르기도 하는) 그리고 이더넷 하드웨어 주소(맥(MAC) 주소).

### 24.2.1 예제

다른 관점에서 라우터를 설명하기 위해 다음 예제의 `netstat` 를 활용한다:

```
% netstat -r
Routing tables
```

| Destination       | Gateway          | Flags | Refs | Use   | Netif  | Expire |
|-------------------|------------------|-------|------|-------|--------|--------|
| default           | outside-gw       | UGSc  | 37   | 418   | ppp0   |        |
| localhost         | localhost        | UH    | 0    | 181   | lo0    |        |
| test0             | 0:e0:b5:36:cf:4f | UHLW  | 5    | 63288 | ed0    | 77     |
| 10.20.30.255      | link#1           | UHLW  | 1    | 2421  |        |        |
| example.com       | link#1           | UC    | 0    | 0     |        |        |
| host1             | 0:e0:a8:37:8:1e  | UHLW  | 3    | 4601  | lo0    |        |
| host2             | 0:e0:a8:37:8:1e  | UHLW  | 0    | 5     | lo0 => |        |
| host2.example.com | link#1           | UC    | 0    | 0     |        |        |
| 224               | link#1           | UC    | 0    | 0     |        |        |

처음 두 라인은 기본 라우트(다음 섹션에서 다루게 될)와 localhost 경로를 명시한다.

이 라우팅 테이블에서 *localhost*의 인터페이스(*Netif* 컬럼)는 lo0 이고 루프백 장치로 알려져 있다. 이것은 시작과 끝이 같기 때문에 모든 트래픽을 LAN 을 통해 외부로 보내지 않고 내부로 보낸다.

그 다음은 0:e0:으로 시작되는 주소다. 이들은 맥 주소로 알려져 있는 이더넷 하드웨어 주소다. FreeBSD 는 자동으로 로컬 이더넷의 임의의 호스트(예제에서는 test0)를 찾아서 이 호스트로 가는 경로를 직접 이더넷 인터페이스 ed0 에 추가한다. 일정 시간동안 호스트로부터 응답이 없을 때 사용되는 이런 종류의 라우트와 관련된 타임 아웃도(*Expire* 칼럼)있다. 타임 아웃이 발생하면 이 호스트의 경로는 자동으로 삭제된다. 이들 호스트는 로컬 호스트에서 가장 짧은 경로를 계산하는 RIP 라는 메커니즘을 사용하여 식별된다.

FreeBSD 도 로컬 서브네트워크(10.20.30.255 는 서브네트워크 10.20.30 의 브로드캐스트 주소이고 example.com 은 이 서브네트워크에 연결되어 있는 도메인 이름이다)의 경로 정보를 추가할 수 있다. 명시된 *link#1* 은 머신의 첫 번째 이더넷 카드를 의미한다. 이곳에 추가적으로 지정된 인터페이스가 없음을 의미한다.

이들 두 그룹(로컬 네트워크 호스트와 로컬 서브네트워크) 모두 **routed** 라는 데몬에 의해 자동적으로 경로가 설정된다. **routed** 가 동작하지 않는다면 정적으로 정의한(즉 입력한) 경로만 존재하게 된다.

*host1* 라인은 이더넷 주소로 알 수 있듯이 우리의 호스트를 의미한다. 우리가 호스트에 데이터를 보내면 FreeBSD 는 이더넷 인터페이스로 보내기 보다는 루프백 인터페이스(lo0)를 사용하는 것을 알고 있다.

두 번째 *host2* 라인은 우리가 ifconfig(8) 엘리어스(사용해야 되는 이유는 이더넷 섹션을 본다)를 사용할 때 발생하는 예제다. lo0 인터페이스 이후의 => 표시는 루프백(이 주소도 로컬 호스트를 의미하기 때문이다)을 사용한다는 것이지만 엄밀히 말해서 엘리어스다. 이러한 경로는 엘리어스를 지원하는 호스트만 보여준다. 로컬 네트워크의 다른 호스트는 이런 경로로 *link#1* 라인만 가지고 있다.

다른 섹션에서 다루게 될 마지막 라인(목적지 서브네트워크 224) 멀티캐스팅과 관련 있다.

마지막으로 각 경로의 다양한 속성은 *Flags* 칼럼에서 볼 수 있다. 아래 내용은 이들 플래그 몇 개와 의미를 설명한 것이다:

|          |                                                      |
|----------|------------------------------------------------------|
| <b>U</b> | Up: 이 경로가 활성화됐다.                                     |
| <b>H</b> | Host: 이 경로의 목적지는 싱글 호스트다.                            |
| <b>G</b> | Gateway: 어디서 보낸 것인지 확인하기 위해 이 목적지의 원격 시스템에 무엇이든 보낸다. |



|          |                                                                          |
|----------|--------------------------------------------------------------------------|
| <b>S</b> | Static: 이 경로는 시스템이 자동으로 생성한 것이 아니고 수동으로 설정한 것이다                          |
| <b>C</b> | Clone: 머신에 연결했을 때 이 경로로 기반으로 새로운 경로가 만들어진다. 이런 종류의 경로는 보통 로컬 네트워크에 사용된다. |
| <b>W</b> | WasCloned: 로컬 네트워크(클론) 경로를 기반으로 자동으로 설정된 경로를 나타낸다.                       |
| <b>L</b> | Link: 이더넷 하드웨어에 대한 참조를 포함하는 경로.                                          |

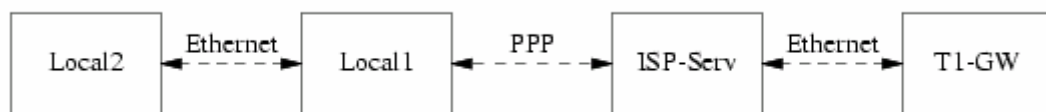
## 24.2.2 기본 라우트

로컬 시스템이 원격호스트에 연결할 때 경로를 결정하기 위해 알고 있는 경로가 있는지 라우팅 테이블을 체크한다. 도달하는 방법을 알고 있는 서브네트워크(복제된 경로)에 원격 호스트가 있다면 시스템은 이 인터페이스에 연결할 수 있는지 체크한다.

알고 있는 모든 경로로 실패했다면 시스템에는 마지막 기본 라우트(경로)라는 옵션이 하나 있다. 이 경로는 특별한 종류의 게이트웨이 경로(보통 시스템에 유일한)로서 플래그 필드에 항상 *c*가 표시되어 있다. 로컬 네트워크의 호스트를 위해 이 게이트웨이는 머신이 외부(PPP 링크, DSL, 케이블 모뎀, T1 또는 다른 네트워크 인터페이스를 통해서)로 직접 연결할 수 있도록 설정되어 있다.

외부로 나가는 게이트웨이 역할을 하는 머신에 기본 라우트를 설정한다면 기본 라우트는 인터넷 서비스 공급자(ISP)의 게이트웨이 머신이 된다.

기본 라우트의 예제를 보자. 이것은 일반적인 설정이다:



호스트 Local 1 과 Local 2 는 여러분의 사이트에 있다. Local 1 은 다이얼-업 PPP 를 통해 ISP 에 연결되어 있다. 이 PPP 서버 컴퓨터는 로컬 영역 네트워크를 지나서 다른 게이트웨이 컴퓨터에 연결되어 있고 외부 인터페이스를 거쳐 ISP 인터넷 망에 연결된다.

각 머신의 기본 게이트웨이는 아래와 같다:

| 호스트  | 기본 게이트웨이 | 인터페이스 |
|------|----------|-------|
| 로컬 2 | 로컬 1     | 이더넷   |

|      |       |     |
|------|-------|-----|
| 로컬 1 | T1-GW | PPP |
|------|-------|-----|

일반적으로 "왜 로컬 1의 기본 게이트웨이를 ISP의 서버로 설정하지 않고 T1-GW로 설정하는가?"라고 자주 듣게 되었다.

PPP 인터페이스는 내부 연결에 ISP의 로컬 네트워크(여러분 쪽의) 주소를 사용하기 때문에 ISP 로컬 네트워크에 있는 모든 머신의 경로는 자동으로 생성되고 있다. 그래서 T1-GW 머신에 어떻게 도달하는지 이미 알고 있기 때문에 ISP 서버에 데이터를 보내는 중간 단계가 필요 없어진다.

마지막으로 주소 X.X.X.1을 로컬 네트워크의 기본 게이트웨이로 사용하는 것이 일반적이다. 그래서(같은 예제를 사용하여) 로컬 C클래스 주소 공간이 10.20.30이고 ISP가 10.9.9를 사용하였다면 기본 라우트는 다음과 같다:

| 호스트                          | 기본 라우트            |
|------------------------------|-------------------|
| 로컬 2 (10.20.30.2)            | 로컬 1 (10.20.30.1) |
| 로컬 1 (10.20.30.1, 10.9.9.30) | T1-GW (10.9.9.1)  |

/etc/rc.conf 파일을 통해 기본 라우트를 쉽게 지정할 수 있다. 예제에서 Local2 머신의 /etc/rc.conf에 다음 라인을 추가한다:

```
defaultrouter="10.20.30.1"
```

아니면 route(8) 명령으로 직접 지정할 수도 있다:

```
# route add default 10.20.30.1
```

### 24.2.3 이중 네트워크 호스트

두 개의 다른 네트워크에 있는 호스트에 관해 우리가 다루어야 할 한가지 설정이 있다. 기술적으로 게이트웨이 기능을 하는 어떤 머신(PPP 연결을 사용하는 위의 예제에서)이라도 이중 네트워크 호스트(dual-homed host)라고 할 수 있다. 그러나 이 용어는 두 개의 로컬 영역 네트워크에 연결되어 있는 머신을 언급할 때만 사용한다.

머신에 두 개의 이더넷 카드가 있으면 각 이더넷 카드는 서브네트워크로 나누어진 주소를

---

가지고 있다. 그렇지 않으면 머신은 오직 하나의 이더넷 카드를 가지고 있고 `ifconfig(8)` 엘리머싱을 사용할 것이다. 전자는 물리적으로 나누어진 두 개의 이더넷 네트워크에 사용되고 후자는 물리적인 하나의 네트워크지만 논리적으로 두 개로 나누어진 서브네트워크에서 사용된다.

양쪽에 라우팅 테이블을 설정하여 각 서브네트워크는 이 머신이 다른 서브네트워크의 게이트웨이(내부 라우트)로 설정되었음을 알고 있다. 두 서브네트워크의 라우터로 동작하는 머신을 설정하는 것은 패킷 필터링 또는 어느 한 방향이나 양쪽으로 방화벽이 필요할 때 가끔 사용된다.

이 머신이 실제로 두 인터페이스 사이의 패킷을 포워드 하기를 원한다면 FreeBSD 가 이 기능을 활성화하도록 해야 된다.

## 24.2.4 라우터 구축

네트워크 라우터는 한쪽 인터페이스에서 다른 쪽으로 패킷을 포워드하는 단순한 시스템이다. 인터넷 표준과 바람직한 엔지니어링 관습이 FreeBSD 프로젝트가 기본적으로 FreeBSD 에서 라우터 기능을 활성화하지 못하게 했다. 이 기능은 `rc.conf(5)`에서 다음 변수를 `YES`로 변경하여 활성화할 수 있다:

```
gateway_enable=YES           # Set to YES if this host will be a gateway
```

이 옵션은 `sysctl(8)`변수 `net.inet.ip.forwarding`를 `1`로 설정한다. 일시적으로 라우팅을 멈추려면 이 설정을 잠시 `0`으로 변경할 수 있다.

새로운 라우터는 데이터를 어디로 보낼지 알아야 한다. 네트워크가 아주 단순하다면 정적인 경로를 사용할 수 있다. FreeBSD 도 RIP(버전 1 과 버전 2 두 가지로) 및 IRDP 와 통신하는 표준 BSD 라우팅 데몬 `routed(8)`을 가지고 있다. BGP v4, OSPF v2 와 매우 정교한 다른 라우팅 프로토콜은 `net/zebra` 패키지로 사용할 수 있다. 더욱 복잡한 네트워크 라우팅 솔루션으로 `GateD` 와 같은 상용 제품도 사용할 수 있다.

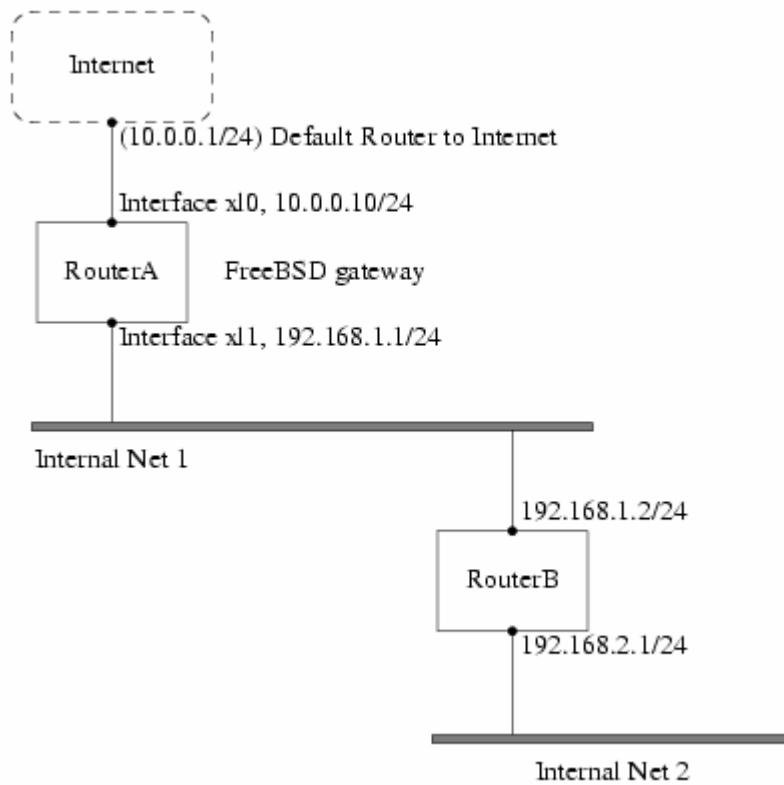
FreeBSD 를 라우터로 설정했더라도 라우터에 필요한 완벽한 인터넷 표준을 따르지 않는다. 그러나 일반적으로 사용하기에는 적당하다.

---

## 24.2.5 정적인 경로 설정

### 24.2.5.1 직접 설정

다음과 같은 네트워크를 가지고 있다고 가정한다:



이 시나리오에서 RouterA 는 나머지 인터넷에 라우터로 동작하는 FreeBSD 머신이다. 이 머신은 외부로 연결할 수 있도록 10.0.0.1 로 설정된 기본 라우트를 가지고 있다. 그리고 RouterB 는 이미 적절하게 설정되어 있어서 필요한 곳에 어떻게 도달하는지 알고 있다고 가정한다(이 그림에서 이렇게 설정하는 것은 간단하다. 게이트웨이로 192.168.1.1 을 사용하도록 RouterB 의 기본 라우트를 추가한다.)

RouterA 의 라우팅 테이블을 확인해 보면 다음과 비슷할 것이다:

---

```
% netstat -nr
```

```
Routing tables
```

```
Internet:
```

| Destination  | Gateway   | Flags | Refs | Use   | Netif | Expire |
|--------------|-----------|-------|------|-------|-------|--------|
| default      | 10.0.0.1  | UGS   | 0    | 49378 | xl0   |        |
| 127.0.0.1    | 127.0.0.1 | UH    | 0    | 6     | lo0   |        |
| 10.0.0/24    | link#1    | UC    | 0    | 0     | xl0   |        |
| 192.168.1/24 | link#2    | UC    | 0    | 0     | xl1   |        |

현재 RouterA 의 라우팅 테이블로는 Internal Net 2 에 도달할 수 없다. 192.168.2.0/24 로의 경로가 없지만 직접 추가할 수 있다. 다음 명령은 192.168.1.2 를 다음 홉(hop)으로 사용하여 Internal Net 2 네트워크를 RouterA 의 라우팅 테이블에 추가한다:

```
# route add -net 192.168.2.0/24 192.168.1.2
```

이제 RouterA 는 192.168.2.0/24 네트워크의 호스트에 도달할 수 있다.

## 24.2.5.2 설정 유지

위 예제는 시스템에 정적인 라우트를 설정하기에 적당하다. 그러나 FreeBSD 머신을 재부팅 하면 라우팅 정보가 유지되지 않는다. 해결할 수 있는 방법은 /etc/rc.conf 파일에 고정 경로를 추가하는 것이다:

```
# Add Internal Net 2 as a static route
static_routes="internalnet2"
route_internalnet2="-net 192.168.2.0/24 192.168.1.2"
```

*static\_routes* 설정 변수는 스페이스(space)로 나뉜 문자열 리스트다. 각 문자열은 경로 이름을 의미한다. 위의 예제는 *static\_routes* 하나만 있고 문자열은 *internalnet2*다. 그리고 route(8) 명령에 지정하는 모든 설정 매개변수를 지정하는 곳에 *route\_internalnet2* 라는 설정 변수를 추가한다. 위의 예제에 다음 명령을 사용한다:

```
# route add -net 192.168.2.0/24 192.168.1.2
```

---

그래서 `"-net 192.168.2.0/24 192.168.1.2"`가 필요하다.

위에서 말했듯이 `static_routes`에 하나 이상의 문자열을 지정할 수 있다. 따라서 여러 개의 정적인 경로를 추가할 수 있다. 다음 라인은 192.168.0.0/24 와 192.168.1.0/24 네트워크에 정적인 경로를 추가하는 예를 보여 준다.

```
static_routes="net1 net2"  
route_net1="-net 192.168.0.0/24 192.168.0.1"  
route_net2="-net 192.168.1.0/24 192.168.1.1"
```

## 24.2.6 라우팅 전달

외부와의 경로를 어떻게 정의하면 되는지 이미 설명했지만 외부에서 우리를 어떻게 찾는지 설명하지 않았다.

라우팅 테이블 설정 방법을 알고 있기 때문에 특정 주소(예제에서 C 클래스 서브넷)로 향하는 모든 트래픽을 내부로 포워드 하는 네트워크의 특정 호스트로 보낼 수 있다.

여러분의 사이트에 주소를 할당할 때 서비스 공급자가 그들의 라우팅 테이블을 설정하기 때문에 여러분 서브네트워크로 향하는 모든 트래픽은 PPP 링크를 통해 여러분의 사이트로 보내진다. 그러나 다른 나라에 있는 사이트에서 여러분의 ISP 로 데이터를 어떻게 보내는가?

할당된 모든 주소에 관한 정보를 가지고 있는 시스템이 있고(DNS 정보 배포와 같은) 인터넷 백본에 연결 위치를 지정한다. "백본"은 인터넷 트래픽을 전 세계로 전달하는 주 라인이다. 각 백본 머신은 특정 네트워크에서 특정 백본으로 연결되는 트래픽 그리고 그 백본에서 여러분의 ISP 까지 연결되는 모든 서비스 제공자들을 거치는 마스터 경로 테이블의 복사본을 가지고 있다.

여러분 사이트와 연결되는 곳을 백본 사이트에 알리는 것은 인터넷 서비스 공급자의 몫이고 이것을 경로 전달이라고 한다.

---

## 24.2.7 문제 해결

가끔 경로 전달에 문제가 있어서 어떤 사이트는 연결할 수 없다. 라우팅이 다운된 곳을 찾기 위해 사용할 수 있는 가장 유용한 명령은 `traceroute(8)` 명령이다. 원격 머신에 연결되지 않은 것처럼 보인다면 역시 유용하다(다시 말해 `ping(8)`이 실패한 경우).

`traceroute(8)` 명령은 연결을 원하는 머신 이름으로 실행한다. 결국에 목적 호스트에 도달하던가 연결이 끊겨서 중단되더라도 `traceroute(8)` 명령은 게이트웨이 호스트를 따라가는 길을 보여준다.

더 많은 정보는 `traceroute(8)` 매뉴얼 페이지를 본다.

## 24.2.7 멀티캐스트 라우팅

FreeBSD 는 멀티캐스트 어플리케이션과 멀티캐스트 라우팅을 기본적으로 지원한다. 멀티캐스트 어플리케이션을 사용하기 위해 FreeBSD 를 특별히 설정할 필요는 없다; 어플리케이션은 보통 FreeBSD 와 상관없이 실행된다. 멀티캐스트 라우팅은 커널에 컴파일 해서 지원해야 한다:

`options MROUTING`

게다가 멀티캐스트 라우팅 데몬 `mouted(8)`은 `/etc/mouted.conf` 파일로 터널과 DVMRP 로 설정해야 된다. 멀티캐스트 설정에 대한 자세한 사항은 `mouted(8)` 매뉴얼 페이지에서 찾을 수 있다.

## 24.3 무선 네트워킹

### 24.3.1 소개

네트워크 케이블이 없이 항상 컴퓨터를 사용할 수 있다는 것은 매우 유용하다. FreeBSD 는 무선 클라이언트로 사용할 수 있고 "액세스 포인트"로도 사용할 수 있다.

---

## 24.3.2 무선 모드 작동

무선 장치 802.11 을 설정하는 BSS 와 IBSS 두 가지 방법이 있다.

### 24.3.2.1 BSS 모드

BSS 모드는 일반적으로 사용되는 모드이며 인프라스트럭처(infrastructure) 모드라고도 한다. 이 모드에서 다양한 무선 액세스 포인트가 유선 네트워크에 연결된다. 각 무선 네트워크는 고유한 이름을 가지고 있다. 이 이름을 네트워크의 SSID 라고 한다.

무선 클라이언트는 이들 무선 액세스 포인트에 연결되며 IEEE 802.11 표준은 무선 네트워크 연결에 사용하는 프로토콜을 정의하고 있다. 무선 클라이언트는 SSID 를 설정하여 특정 네트워크에 연결할 수 있지만 정확히 설정하지 않고도 어떤 네트워크에든 연결할 수 있다.

### 24.3.2.2 IBSS 모드

ad-hoc 모드(무선 클라이언트 상호간의 통신을 지원하기 때문에 액세스 포인트가 필요 없다)로 부르기도 하는 IBSS 모드는 포인트와 포인트를 연결하기 위해 디자인 되었다. 실제로 두 종류의 ad-hoc 모드가 있다. 첫 번째는 ad-hoc 또는 IEEE ad-hoc 모드라고 부르기도 하는 IBSS 모드다. 이 모드는 IEEE 802.11 표준에 의해 정의된다. 두 번째는 demo ad-hoc 모드 또는 Lucent ad-hoc(그리고 좀 혼란스럽게 가끔 ad-hoc 모드로 부르기도 한다) 모드로 부른다. 이것은 예전 pre-802.11 ad-hoc 모드이고 래거시(legacy) 설치에만 사용되어 왔다. ad-hoc 모드는 더 이상 설명하지 않는다.

## 24.3.3 인프라스트럭처 모드(Infrastructure Mode)

### 24.3.3.1 액세스 포인트

액세스 포인트는 하나 이상의 무선 클라이언트가 중앙 허브로 사용할 수 있는 무선 네트워크 장치다. 액세스 포인트를 사용할 때 모든 클라이언트는 액세스 포인트를 통해 통신한다. 여러 개의 액세스 포인트로 집, 사무실, 공원 등을 무선 네트워크로 완벽하게 커버할 수 있다.



---

액세스 포인트는 일반적으로 여러 개의 네트워크에 연결된다: 무선카드 그리고 나머지 네트워크에 연결하기 위한 하나 또는 하나 이상의 유선 네트워크 카드다.

액세스 포인트는 미리 빌드된 것을 구입하거나 FreeBSD 에 지원되는 무선 카드로 직접 빌드 할 수 있다. 여러 벤더에서 다양한 무선 액세스 포인트와 카드를 생산한다.

### 24.3.3.2 FreeBSD 액세스 포인트 빌드

#### [액세스 포인트를 빌드해서 확인하는 방법]

##### 1. 필요한 사항

FreeBSD 로 액세스 포인트를 만들려면 호환되는 무선 카드가 필요하다. 현재 유일하게 지원되는 카드는 Prism 칩셋이다. 그리고 FreeBSD 에서 지원되는 무선 네트워크 카드도 필요하다(FreeBSD 가 다양한 종류의 장치를 지원하기 때문에 어렵지 않다). 이 가이드에서는 네트워크에 연결되어 있는 네트워크 카드 사이의 모든 트래픽을 무선 장치와 유선으로 중개하는 bridge(4)를 설명한다.

FreeBSD 가 액세스 포인트를 사용하는 hostap 기능은 특정 버전의 펌웨어에서 가장 제대로 동작한다. Prism 2 카드는 펌웨어 버전 1.3.4 나 새로운 버전을 사용해야 된다. Prism 2.5 와 Prism 3 카드는 펌웨어 1.4.9 를 사용해야 된다. 예전 버전의 펌웨어는 정확하게 작동하지 않을 것이다. 여기서 카드를 업데이트하는 유일한 방법은 카드 생산자가 제공하는 Windows 펌웨어 업데이트 유틸리티를 사용해야 된다.

##### 2. 설치

첫째로 시스템이 무선 카드를 인식해야 된다:

```
# ifconfig -a
wi0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
    inet6 fe80::202:2dff:fe2d:c938%wi0 prefixlen 64 scopeid 0x7
    inet 0.0.0.0 netmask 0xff000000 broadcast 255.255.255.255
    ether 00:09:2d:2d:c9:50
    media: IEEE 802.11 Wireless Ethernet autoselect (DS/2Mbps)
    status: no carrier
    ssid ""
```

```
stationname "FreeBSD Wireless node"
channel 10 authmode OPEN powersavemode OFF powersavesleep 100
wepmode OFF weptxkey 1
```

지금은 자세한 사항에 대해 신경 쓰지 않고 설치된 무선 카드를 보여준다는 것만 알고 있자. 무선 인터페이스를 인지하고 PC 카드를 사용하는데 문제가 있다면 `pccardc(8)`을 확인하고 `pccardd(8)`에 대한 더 자세한 사항은 매뉴얼 페이지를 참고한다.

그리고 액세스 포인트로 만들기 위해 FreeBSD의 브리지 부분을 준비하도록 모듈을 로드 해야 된다. `bridge(4)` 모듈을 로드 하려면 간단히 다음 명령을 실행한다:

```
# kldload bridge
```

모듈을 로드 할 때 에러 메시지가 없어야 된다. 에러 메시지가 나타나면 커널에 `bridge(4)` 코드를 컴파일 해야 된다. 핸드북의 브릿지 섹션에서 이 태스크를 해결하도록 도와줄 것이다.

이제 브리지 기능이 갖추어졌다. FreeBSD 커널에게 어떤 인터페이스가 브리지인지 `sysctl(8)`로 명시해야 된다:

```
# sysctl net.link.ether.bridge=1
# sysctl net.link.ether.bridge_cfg="wi0 xl0"
# sysctl net.inet.ip.forwarding=1
```

FreeBSD 5.2-RELEASE 와 이후 버전에서는 다음 옵션을 사용한다:

```
# sysctl net.link.ether.bridge.enable=1
# sysctl net.link.ether.bridge.config="wi0,xl0"
# sysctl net.inet.ip.forwarding=1
```

무선 카드 설정은 다음 명령이 카드를 액세스 포인트로 설정한다:

```
# ifconfig wi0 ssid my_net channel 11 media DS/11Mbps mediaopt hostap up
stationname "FreeBSD AP"
```

`ifconfig(8)` 명령으로 `wi0` 인터페이스를 up, SSID 를 `my_net`으로 설정 그리고 스테이션

---

이름을 *FreeBSD AP*로 설정한다. *media DS/11Mbps*는 카드를 11Mbps 모드로 설정하고 이 설정은 *mediaopt* 효과를 위해 필요하다. *mediaopt hostap* 옵션은 인터페이스를 액세스 포인트 모드로 설정한다. *channel 11* 옵션은 802.11b 채널을 사용하도록 설정한다. *wicontrol(8)* 매뉴얼 페이지에서 도메인을 규정하는데 유효한 채널 옵션을 보여준다.

이제 액세스 포인트가 동작하여 완벽하게 기능을 수행한다. *wicontrol(8)*, *ifconfig(8)*과 *wi(4)*를 읽어서 더 많은 정보를 얻을 수 있다.

### 3. 상태 정보

액세스 포인트가 설정되어 동작하면 운영자는 액세스 포인트에 연결된 클라이언트를 확인하고 싶을 것이다. 언제라도 다음 명령을 입력하여 확인할 수 있다:

```
# wicontrol -l
1 station:
00:09:b7:7b:9d:16 asid=04c0, flags=3<ASSOC,AUTH>, caps=1<ESS>,
rates=f<1M,2M,5.5M,11M>, sig=38/15
```

이것은 매개변수로 하나의 스테이션이 연결되어 있는 것을 보여준다. 신호는 상대적인 새기(strength)만 보여주는데 사용된다. dBm 로 변환하거나 다른 펌웨어에서 사용하는 다른 단위로 변경된다.

### 24.3.3.3 클라이언트

무선 클라이언트는 액세스 포인트나 다른 클라이언트에 직접 접근하는 시스템이다. 일반적으로 무선 클라이언트는 하나의 무선 네트워크 카드 장치를 가지고 있다.

무선 클라이언트를 설정하는 몇 가지 다른 방법이 있다. 이들은 보통 BSS 와(액세스 포인트가 필요한 인프라스트럭처 모드) IBBS 의(ad-hoc 또는 peer - to - peer 모드) 다른 무선 모드에 기반한다. 예제에서는 액세스 포인트와 통신하기 위한 두 가지 중 가장 유명한 BSS 모드를 사용한다.

---

## [무선 클라이언트 설정]

### 1. 필요한 사항

FreeBSD 를 무선 클라이언트로 설정하기 위해 FreeBSD 가 지원하는 무선 카드가 필요하다.

### 2. 무선 FreeBSD 클라이언트 설정

시작하기 전에 알고 있어야 되는 몇 가지가 있다. 이 예제에서는 암호화를 끄고 *my\_net*이라는 이름의 네트워크에 연결한다.

**Note:** 이 예제는 데이터를 암호화하지 않아서 보안적으로 위험하다. 다음 섹션에서는 어떻게 암호화하고 왜 암호화가 중요한지 그리고 어떤 암호화 기술은 왜 아직도 완벽하게 통신을 보호할 수 없는지 설명한다.

카드가 FreeBSD 에서 감지되어야 한다:

```
# ifconfig -a
wi0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
    inet6 fe80::202:2dff:fe2d:c938%wi0 prefixlen 64 scopeid 0x7
    inet 0.0.0.0 netmask 0xff000000 broadcast 255.255.255.255
    ether 00:09:2d:2d:c9:50
    media: IEEE 802.11 Wireless Ethernet autoselect (DS/2Mbps)
    status: no carrier
    ssid ""
    stationname "FreeBSD Wireless node"
    channel 10 authmode OPEN powersavemode OFF powersavesleep 100
    wepmode OFF weptxkey 1
```

그리고 네트워크에 맞게 카드를 정확히 설정한다:

```
# ifconfig wi0 inet 192.168.0.20 netmask 255.255.255.0 ssid my_net
```

192.168.0.20 과 255.255.255.0 를 여러분의 무선 네트워크에 맞는 IP 와 넷 마스크로 바꾼다. 우리의 액세스 포인트는 무선 네트워크와 유선 네트워크 사이의 데이터 브리지 역할을 하기 때문에 유선 네트워크에 다른 장치처럼 나타난다.

이 설정이 끝나면 표준 유선 연결처럼 무선 네트워크에서도 호스트에 ping 을 보낼 수 있다.

무선 연결에 문제가 있다면 액세스 포인트와 관련된 것을(연결된) 체크한다:

```
# ifconfig wi0
```

다음과 같은 어떤 정보가 나타나므로 확인한다:

```
status: associated
```

정보가 나타나지 않는다면 액세스 포인트의 범위를 벗어났거나, 데이터를 암호화하도록 했거나 설정 문제일 것이다.

#### 24.3.3.4 암호화

완벽히 보호되는 영역으로 네트워크를 유지할 수 없기 때문에 무선 네트워크를 암호화하는 것은 중요하다. 무선 데이터는 인근지역 전체로 브로드캐스트 되므로 이 데이터를 누구나 읽을 수 있다. 이런 이유로 암호화가 필요하다. 데이터를 암호화해서 전파를 보내면 데이터를 잡기가 더 어려워진다.

클라이언트와 액세스 포인트 사이의 데이터를 암호화하는 아주 일반적인 두 가지 방법은 WEP 와 ipsec(4)다.

##### [무선 데이터 암호화하기]

###### 1. WEP

WEP 는 Wired Equivalency Protocol 의 약어다. WEP 는 유선 네트워크처럼 무선 네트워크를 안전하게 만든다. 그러나 이것은 크랙되어서 깨지기 쉽다. 따라서 중요한 데이터를 암호화하는 곳에 사용하지 않는다.

암호화하지 않는 것 보다는 안전하기 때문에 새로운 FreeBSD 액세스 포인트에서 다음 명령으로 WEP 를 켜다:

```
# ifconfig wi0 inet up ssid my_net wepmode on wepkey 0x1234567890 media
```

---

## DS/11Mbps mediaopt hostap

그리고 다음 명령으로 클라이언트에서 WEP 를 켤 수 있다:

```
# ifconfig wi0 inet 192.168.0.20 netmask 255.255.255.0 ssid my_net wepmode on  
wepkey 0x1234567890
```

0x1234567890 는 유일한 키로 바꾼다.

## 2. IPsec

ipsec(4)는 네트워크 사이의 데이터를 암호화하는 더욱 안전하고 강력한 툴이다.

이것이 무선 네트워크 데이터를 암호화하는 확실한 선택이다. ipsec(4) 보안과 사용법은 이 핸드북의 ipsec 섹션에서 더 읽을 수 있다.

## 24.3.3.5 툴

무선 네트워크 설정과 디버깅에 사용할 수 있는 몇 가지 툴이 있고 여기서 이 중 몇 가지를 소개한다.

### 1. bsd-airtools 패키지

**bsd-airtools** 패키지는 WEP 키 크래킹과 액세스 포인트 탐지 등을 포함하여 완벽한 무선 점검 툴 세트다.

**bsd-airtools** 유틸리티는 [net/bsd-airtools](#) 포트에서 설치할 수 있다. 포트에서 설치하는 정보는 핸드북 4 장에서 찾을 수 있다.

**dstumbler** 프로그램은 액세스 포인트를 감지해서 신호 대 잡음 비를 기록할 수 있는 패키지 툴이다. 액세스 포인트를 올리고 실행하는데 어려움을 겪는다면 **dstumbler** 이 문제해결을 도와줄 수 있다.

무선 네트워크 보안을 테스트하려면 여러분의 무선 네트워크 보안에 WEP 이 적당한지 결정하는데 도움을 줄 수 있는 "dweputils"(dwepcrack, dwepdump 와 dwepkeygen) 선택할 수 있다.

---

## 2. wicontrol, ancontrol 과 raycontrol 유틸리티

이 유틸리티들은 무선 네트워크에서 무선 카드를 제어하는 툴이다. 위의 예제에서 무선 카드가 wi0 인터페이스이기 때문에 wicontrol(8)을 선택했다. Cisco 무선 장치를 가지고 있다면 an0 로 나타나기 때문에 ancontrol(8)을 사용한다.

## 3. ifconfig 명령

ifconfig(8) 명령은 wicontrol(8)과 같은 다양한 옵션을 사용할 수 있지만 몇 가지 옵션이 빠져있다. 명령어 라인 매개변수와 옵션은 ifconfig(8)을 체크한다.

### 24.3.3.6 지원되는 카드

#### 액세스 포인트

현재 BSS 모드를 지원하는 유일한 카드는 Prism 2, 2.5 또는 3 칩셋 기반의 장치다. 정확한 리스트는 wi(4)를 본다.

#### 클라이언트

거의 모든 802.11b 무선 카드가 현재 FreeBSD 에서 지원된다. Prism, Spectrum24, Hermes, Aironet 과 Raylink 기반의 대부분의 카드는 IBSS 모드에서(ad-hoc, peer-to-peer 과 BSS) 무선카드로 작동한다.

## 24.4 블루투스

### 24.4.1 소개

블루투스는 라이선스가 없는 10M 범위의 2.4Ghz 밴드에서 개인 네트워크를 생성할 때 사용하는 무선 기술이다. 네트워크는 보통 핸드폰, PDA 와 노트북 컴퓨터 같은 휴대용 장치에서 ad-hoc 형식으로 사용된다. 다른 유명한 무선 기술(Wi-Fi)과 다르게 블루투스는 FTP 같은 파일 서버, 파일 공유, 음성 전달, 시리얼 라인 에뮬레이션 등 고급 레벨의 서비스를 제공한다.

FreeBSD 에서 블루투스 스택은 Netgraph 프레임워크를(netgraph(4)를 본다) 사용한다.

---

블루투스의 광범위한 USB 동글은(dongles) ng\_ubt(4) 드라이버로 지원된다. Broadcom BCM2033 칩 기반 블루투스 장치는 ubtbcmfw(4)와 ng\_ubt(4) 드라이버로 지원된다. 3Com 블루투스 PC 카드 3CRWB60-A는 ng\_bt3c(4) 드라이버로 지원된다. 시리얼과 UART 기반 블루투스 장치는 sio(4), ng\_h4(4) 그리고 hcseriald(8)이 지원한다. 이번 장은 USB 블루투스 동글을 사용하는 방법을 설명한다. 블루투스 지원은 FreeBSD 5.0 과 새로운 버전에서 사용할 수 있다.

## 24.4.2 장치 연결

기본적으로 블루투스 장치 드라이버는 커널 모듈로 사용할 수 있다. 장치를 연결하기 전에 드라이버를 커널에 로드 해야 된다.

```
# kldload ng_ubt
```

블루투스 장치가 시스템이 시작될 때 나타나면 /boot/loader.conf 에서 모듈을 로드 한다.

```
ng_ubt_load="YES"
```

USB 동글을 연결한다. 다음과 비슷한 출력 결과가 콘솔에(또는 syslog 에) 나타난다.

```
ubt0: vendor 0x0a12 product 0x0001, rev 1.10/5.25, addr 2
ubt0: Interface 0 endpoints: interrupt=0x81, bulk-in=0x82, bulk-out=0x2
ubt0: Interface 1 (alt.config 5) endpoints: isoc-in=0x83, isoc-out=0x3,
      wMaxPacketSize=49, nframes=6, buffer size=294
```

/usr/share/examples/netgraph/bluetooth/rc.bluetooth 를 /etc/rc.bluetooth 와 같은 적당한 위치로 복사한다. 이 스크립트는 블루투스 스택을 시작하고 정지하는데 사용된다. 장치를 빼기 전에 스택을 정지 시키는 것이 좋지만 일반적으로 치명적이지는 않다. 스택을 시작할 때 다음과 비슷한 내용을 보게 된다:

```
# /etc/rc.bluetooth start ubt0
BD_ADDR: 00:02:72:00:d4:1a
Features: 0xff 0xff 0xf 00 00 00 00 00
<3-Slot> <5-Slot> <Encryption> <Slot offset>
```



```

<Timing accuracy> <Switch> <Hold mode> <Sniff mode>
<Park mode> <RSSI> <Channel quality> <SCO link>
<HV2 packets> <HV3 packets> <u-law log> <A-law log> <CVSD>
<Paging scheme> <Power control> <Transparent SCO data>
Max. ACL packet size: 192 bytes
Number of ACL packets: 8
Max. SCO packet size: 64 bytes
Number of SCO packets: 8

```

### 24.4.3 호스트 컨트롤러 인터페이스 (HCI)

호스트 컨트롤러 인터페이스는(HCI) 베이스밴드(Baseband) 컨트롤러와 링크 관리자 그리고 하드웨어 상태 확인과 제어를 등록하는 일반적인 인터페이스다. 이 인터페이스는 블루투스 베이스밴드 기능에 접근하는 일관된 방법을 제공한다. 호스트의 HCI 레이어는 블루투스 하드웨어에 HCI 펌웨어와 데이터 및 명령을 전달한다. 호스트 컨트롤러 전송 레이어(즉 물리적인 버스) 드라이버는 각자 정보를 교환할 수 있도록 HCI 레이어를 양쪽에 제공한다.

*hci* 타입의 Netgraph 노드 하나는 한 대의 블루투스 장치에 생성된다. HCI 노드는 보통 블루투스 장치 드라이버 노드(down 스트림)와 L2CAO 노드(up 스트림)에 연결된다. 모든 HCI 운용은 장치 드라이버 노드가 아닌 HCI 노드에서 수행해야 된다. HCI 노드의 기본 이름은 “devicehci”다. 더 자세한 사항은 `ng_hci(4)` 매뉴얼 페이지를 확인한다.

가장 일반적인 태스크는 RF 접근 방식으로 블루투스 장치를 인지하는 것이다. 이 동작을 *inquiry*라고 한다. *inquiry*와 HCI 관련된 다른 동작은 `hccontrol(8)` 유틸리티로 실행된다. 아래 예제는 어떤 블루투스 장치가 범위에 있는지 발견하는 것을 보여준다. 몇 초 안 장치 리스트를 보여준다. 원격 장치가 *discoverable* 모드에 있다면 *inquiry*에 반응한다.

```

% hccontrol -n ubt0hci inquiry
Inquiry result, num_responses=1
Inquiry result #0
BD_ADDR: 00:80:37:29:19:a4
    Page Scan Rep. Mode: 0x1
    Page Scan Period Mode: 00
    Page Scan Mode: 00
    Class: 52:02:04

```

---

```
Clock offset: 0x78ef
Inquiry complete. Status: No error [00]
```

*BD\_ADDR*은 네트워크 카드의 MAC 주소와 비슷한 블루투스 장치의 유일한 주소다. 이 주소는 장치와 통신할 때 필요하다. 사람이 읽을 수 있는 이름을 *BD\_ADDR*에 할당할 수 있다. `/etc/Bluetooth/hosts` 파일은 감지한 블루투스 호스트에 대한 정보를 가지고 있다. 다음 예제는 사람이 읽을 수 있도록 원격 장치에 할당되어있는 이름을 어떻게 가져오는지 보여준다.

```
% hccontrol -n ubt0hci remote_name_request 00:80:37:29:19:a4
BD_ADDR: 00:80:37:29:19:a4
Name: Pav's T39
```

원격 블루투스 장치에 `inquiry`를 실행하면 “`your.host.name(ubt0)`”으로 찾는다. 로컬 장치에 할당된 이름은 언제라도 변경할 수 있다.

블루투스 시스템은 일대 일(`point-to-point`) 연결(오직 두 개의 블루투스 유닛만 연결된다)이나 일대 다중(`point-to-multipoint`) 연결을 제공한다. 일대 다중(`Point-to-multipoint`) 연결은 여러 대의 블루투스 장치와 동시에 연결된다. 다음 예제는 로컬 장치에 연결되어 있는 베이스밴드 연결 리스트를 어떻게 가져오는지 보여준다.

```
% hccontrol -n ubt0hci read_connection_list
Remote BD_ADDR   Handle Type Mode Role Encrypt Pending Queue State
00:80:37:29:19:a4  41  ACL   0 MAST  NONE      0    0 OPEN
```

*연결 제어*는 베이스밴드 연결을 종료해야 될 때 유용하다. 보통 직접 종료할 필요는 없다. 스택이 활동하지 않는 베이스밴드 연결을 자동으로 종료한다.

```
# hccontrol -n ubt0hci disconnect 41
Connection handle: 41
Reason: Connection terminated by local host [0x16]
```

`hccontrol help` 명령이 사용할 수 있는 HCI 명령의 전체 리스트를 보여준다. 대부분의 HCI 명령은 슈퍼 유저 권한이 필요 없다.

---

## 24.4.4 Logical Link Control 와 Adaptation Protocol (L2CAP)

논리적 링크 제어와 Adaptation Protocol(L2CAP)은 프로토콜 다중화(multiplexing), 프로토콜 분할과 재 조합으로 상위 레이어 프로토콜에 연결 지향과 비 연결 데이터(connectionless data) 서비스를 제공한다. L2CAP 는 상위 레벨 프로토콜과 어플리케이션이 64 킬로바이트 이상의 L2CAP 데이터 패킷을 보내고 받는 것을 허용한다.

L2CAP 는 *채널(channels)* 개념을 기반으로 한다. 채널은 베이스밴드 연결의 최 상단에 논리적으로 연결된 것이다. 각 채널은 다 대 일(many-to-one) 형식으로 하나의 프로토콜로 바운드 된다. 여러 채널이 같은 프로토콜로 바운드될 수 있지만 하나의 채널은 여러 프로토콜로 바운드될 수 없다. 채널에 도착하는 각각의 L2CAP 패킷은 적절한 상위 레벨 프로토콜로 인계된다. 여러 개의 채널을 같은 베이스밴드 연결에 공유할 수 있다.

하나의 *l2cap* 타입 Netgraph 노드가 블루투스 장치 하나에 생성된다. L2CAP 노드는 보통 블루투스 HCI 노드(down 스트림)와 블루투스 소켓 노드(up 스트림)에 연결된다. L2CAP 노드의 기본 이름은 "device\_l2cap"다. 더 자세한 내용은 ng\_l2cap(4) 매뉴얼 페이지를 참고 한다.

유용한 명령은 다른 장치에 ping 을 보낼 때 사용하는 *l2ping(8)*이다. 어떤 블루투스 도구는 전송한 모든 데이터에 대한 결과를 되돌려 주지 않을 것이기 때문에 다음 예제의 *0 bytes* 가 일반적인 결과다.

```
# l2ping -a 00:80:37:29:19:a4
0 bytes from 0:80:37:29:19:a4 seq_no=0 time=48.633 ms result=0
0 bytes from 0:80:37:29:19:a4 seq_no=1 time=37.551 ms result=0
0 bytes from 0:80:37:29:19:a4 seq_no=2 time=28.324 ms result=0
0 bytes from 0:80:37:29:19:a4 seq_no=3 time=46.150 ms result=0
```

*l2control(8)* 유틸리티는 L2CAP 노드를 다양하게 운용하는데 사용된다. 이 예제는 논리적 연결과(채널) 로컬 장치의 베이스밴드 연결 리스트를 어떻게 가져오는지 보여 준다.

```

% I2control -a 00:02:72:00:d4:1a read_channel_list
L2CAP channels:
Remote BD_ADDR      SCID/ DCID  PSM  IMTU/ OMTU State
00:07:e0:00:0b:ca  66/  64    3   132/  672 OPEN
% I2control -a 00:02:72:00:d4:1a read_connection_list
L2CAP connections:
Remote BD_ADDR      Handle Flags Pending State
00:07:e0:00:0b:ca  41 O          0 OPEN

```

다른 진단 툴은 btsockstat(1)이다. netstat(1)과 비슷하지만 블루투스 네트워크와 관련된 데이터 구조를 보여준다. 아래 예제는 위의 I2control(8)과 같은 논리적인 연결을 보여준다.

```

% btsockstat
Active L2CAP sockets
PCB      Recv-Q Send-Q Local address/PSM      Foreign address  CID  State
c2afe900  0      0 00:02:72:00:d4:1a/3   00:07:e0:00:0b:ca 66   OPEN
Active RFCOMM sessions
L2PCB    PCB      Flag MTU  Out-Q DLCs State
c2afe900 c2b53380 1   127  0   Yes  OPEN
Active RFCOMM sockets
PCB      Recv-Q Send-Q Local address      Foreign address  Chan DLCI State
c2e8bc80  0      250 00:02:72:00:d4:1a 00:07:e0:00:0b:ca 3    6   OPEN

```

## 24.4.5 RFCOMM 프로토콜

RFCOMM 프로토콜은 L2CAP 프로토콜을 통해 시리얼 포트 에뮬레이션을 제공한다. 이 프로토콜은 ETSI(European Telecommunications Standards Institute) 표준 TS 07.10 에 기반한다. RFCOMM 는 RS-232(EIATIA-232-E) 9 핀 시리얼 포트를 에뮬레이팅하는 간단한 추가적인 전송 프로토콜이다. RFCOMM 는 60 개의 블루투스 장치를 동시에 연결(RFCOMM 채널들)할 수 있다.

RFCOMM 의 목적에 맞는 완전한 통신 경로는 다른 장치에서(통신 종단 점) 실행되는 두 개의 어플리케이션과 그들 사이의 통신 세그먼트도 포함한다. RFCOMM 은 장치의 시리얼 포트를 사용하는 어플리케이션을 다루도록 디자인 되어있다. 통신 세그먼트는 장치에서 장치로 연결되는(직접 연결) 블루투스 링크다.

---

RFCOMM 은 장치 사이를 직접 연결하는 경우나 네트워크에서 장치와 모뎀의 연결만  
관련한다. RFCOMM 은 한쪽은 블루투스 무선 기술을 사용하여 통신하고 다른 쪽에는 유선  
인터페이스를 제공하는 통신 모듈처럼 다른 설정을 지원할 수 있다.

FreeBSD 에서 RFCOMM 프로토콜은 블루투스 소켓 레이어에서 실행된다.

## 24.4.6 장치 Pairing

기본적으로 블루투스 통신은 인증이 되지 않아서 어떤 장치라도 다른 장치와 통신할 수  
있다. 블루투스 장치에(예를 들어 핸드폰) 특정 서비스를(예를 들면 전화 접속 서비스)  
제공하기 위해 인증이 필요할 것이다. 블루투스는 *PIN 코드*로 인증할 수 있다. PIN 코드는  
16 문자 길이의 ASCII 문자열이다. 유저는 양쪽 장치에 같은 PIN 코드를 입력해야 된다.  
유저가 PIN 코드를 입력하면 양쪽 장치는 *link key*를 생성한다. 그리고 링크 키는 각  
장치나 특정 스토리지에 저장할 수 있다. 다음부터 양쪽 장치는 이전에 생성한 링크 키를  
사용한다. 앞에서 설명한 절차를 *pairing*이라고 하고 링크 키를 잃어 버렸다면 *pairing*을  
다시 해야 된다.

hcsecd(8) 데몬이 블루투스 인증 요청을 처리할 수 있다. 기본 설정 파일은  
/etc/Bluetooth/hcsecd.conf 다. 임의적으로 "1234"라는 PIN 코드를 설정한 핸드폰 예제가  
아래에 있다.

```
device {
    bdaddr 00:80:37:29:19:a4;
    name    "Pav's T39";
    key     nokey;
    pin     "1234";
}
```

PIN 코드에는(길이를 제외한) 제한이 없다. 어떤 장치에는(예를 들면 블루투스 헤드 셋)  
고정된 PIN 코드가 내장되어 있을 것이다. *-d* 옵션이 hcsecd(8) 데몬을 강제로  
포그라운드로 유지시키기 때문에 어떤 일이 발생하는지 확인하기 쉽다. 원격 장치가  
*pairing*에 응답하도록 설정해서 원격 장치에 블루투스 연결을 시작한다. 원격 장치는  
*pairing*이 허용됐음을 알리고 PIN 코드를 요청한다. hcsecd.conf 에 있는 PIN 코드와 같은  
PIN 코드를 입력하면 PC와 원격 장치가 연결된다. 그렇지 않으면 원격 장치에서 *pairing*을

---

시작할 수 있다. 다음 내용은 **hcsecd** 데몬의 샘플 출력이다.

```
hcsecd[16484]: Got Link_Key_Request event from 'ubt0hci', remote bdaddr
0:80:37:29:19:a4
hcsecd[16484]: Found matching entry, remote bdaddr 0:80:37:29:19:a4, name 'Pav's
T39', link key doesn't exist
hcsecd[16484]: Sending Link_Key_Negative_Reply to 'ubt0hci' for remote bdaddr
0:80:37:29:19:a4
hcsecd[16484]: Got PIN_Code_Request event from 'ubt0hci', remote bdaddr
0:80:37:29:19:a4
hcsecd[16484]: Found matching entry, remote bdaddr 0:80:37:29:19:a4, name 'Pav's
T39', PIN code exists
hcsecd[16484]: Sending PIN_Code_Reply to 'ubt0hci' for remote bdaddr
0:80:37:29:19:a4
```

## 24.4.7 서비스 감지 프로토콜 (SDP)

서비스 감지 프로토콜(SDP)은 클라이언트 어플리케이션에서 서버 어플리케이션이 제공하는 서비스와 이들 서비스의 특성을 감지한다는 의미다. 서비스의 특성은 종류나 제공되는 서비스 등급과 메커니즘 또는 서비스를 사용하기 위해 필요한 프로토콜 정보를 포함한다.

SDP는 SDP 서버와 SDP 클라이언트 사이의 통신에 연관된다. 서버는 서버와 관련된 서비스의 특징을 설명하는 서비스 레코드 리스트를 관리한다. 각 서비스 레코드는 각 서비스에 대한 정보를 가지고 있다. 클라이언트는 SDP 서버가 관리하는 서비스 레코드에 SDP를 요청하여 정보를 갱신한다. 클라이언트 또는 클라이언트와 관련된 어플리케이션이 서비스를 사용하려면 서비스 공급자에게 연결해야 된다. SDP는 서비스와 서비스의 특징을 감지하는 메커니즘을 제공하지만 이들 서비스를 사용하는 메커니즘은 제공하지 않는다.

보통 SDP 클라이언트는 원하는 특징의 서비스를 검색한다. 그러나 SDP 서버의 서비스 레코드에 설명되어 있는 서비스 타입을, 서비스에 대한 사전정보 없이 감지하는 적절한 시간이 있다. 제공되는 서비스를 찾는 절차를 *browsing*이라고 한다.

블루투스 SDP 서버 `sdpd(8)`과 명령어 라인 클라이언트 `sdpcontrol(8)`은 표준 FreeBSD 설치에 포함되어 있다. 다음 예제는 SDP 브라우저 쿼리를 어떻게 수행하는지 보여준다.

---

```

% sdpcontrol -a 00:01:03:fc:6e:ec browse
Record Handle: 00000000
Service Class ID List:
    Service Discovery Server (0x1000)
Protocol Descriptor List:
    L2CAP (0x0100)
        Protocol specific parameter #1: u/int/uuid16 1
        Protocol specific parameter #2: u/int/uuid16 1

Record Handle: 0x00000001
Service Class ID List:
    Browse Group Descriptor (0x1001)

Record Handle: 0x00000002
Service Class ID List:
    LAN Access Using PPP (0x1102)
Protocol Descriptor List:
    L2CAP (0x0100)
    RFCOMM (0x0003)
        Protocol specific parameter #1: u/int8/bool 1
Bluetooth Profile Descriptor List:
    LAN Access Using PPP (0x1102) ver. 1.0

```

각 서비스에는 특성 리스트가(예를 들어 RFCOMM 채널) 있다. 서비스에 따라 몇 가지 특성을 기억해야 될 것이다. 어떤 블루투스 도구는 서비스 브라우징을 지원하지 않기 때문에 리스트를 되돌려 주지 않을 것이다. 이런 경우 특정 서비스를 검색하는 것도 가능하다. 다음 예제는 OBEX Object Push(OPUSH) 서비스를 어떻게 검색하는지 보여준다.

```
% sdpcontrol -a 00:01:03:fc:6e:ec search OPUSH
```

FreeBSD에서는 sdpd(8) 서버로 블루투스 클라이언트에게 서비스를 제공한다.

```
# sdpd
```

블루투스 서비스를 원격 클라이언트에 제공하려는 로컬 서버는 로컬 SDP 데몬으로 서비스를 등록한다. 이러한 어플리케이션에는 rfcmm\_pppd(8)이 있다. 이 어플리케이션이

---

실행되면 로컬 SDP 데몬으로 블루투스 LAN 서비스를 등록된다.

로컬 SDP 서버로 등록된 서비스 리스트는 로컬 제어 채널을 통한 SDP 브라우저 쿼리로 볼 수 있다.

```
# sdpcontrol -l browse
```

## 24.4.8 다이얼-업 네트워크와(DUN) PPP(LAN) 프로파일로 네트워크 접근

다이얼-업 네트워크(DUN) 프로파일은 대부분 모뎀과 핸드폰에 사용된다. 이 프로파일로 다루는 시나리오는 다음과 같다:

- 다이얼-업 인터넷 서버나 다른 다이얼-업 서비스에 연결하기 위한 컴퓨터의 무선 모뎀으로 핸드폰이나 모뎀을 사용한다.
- 데이터를 받기 위해 컴퓨터에 핸드폰이나 모뎀을 사용한다.

PPP(LAN) 프로파일로 네트워크에 접근하는 것은 다음과 같은 상황에 사용할 수 있다:

- 하나의 블루투스 장치로 LAN에 접근할 때.
- 여러 개의 블루투스 장치로 LAN에 접근할 때.
- PC와 PC를 연결(시리얼 케이블 에뮬레이션을 통한 PPP 네트워크를 사용한다)할 때.

FreeBSD 에서 양쪽 프로파일은 RFCOMM 블루투스 연결을 PPP 로 변환하는 `ppp(8)`와 `rfcomm_pppd(8) -a` 로 감싸서 사용한다. 그리고 프로파일 사용하기 전에 `/etc/ppp/ppp.conf` 에 새로운 PPP 라벨을 만들어야 된다. 예제는 `rfcomm_pppd(8)` 매뉴얼 페이지를 참고한다.

다음 예제에서 `rfcomm_pppd(8)`은 DUN RFCOMM 채널에서 `BD_ADDR`



---

00:80:37:29:19:a4 로 원격 장치에 RFCOMM 연결을 여는데 사용된다. 실제 RFCOMM 채널 번호는 원격 장치에서 SDP 를 통해 받아온다. RFCOMM 채널을 직접 지정하는 것도 가능하고 이 경우 rfcomm\_pppd(8)은 SDP 쿼리를 수행하지 않는다. 그리고 원격 장치에서 RFCOMM 채널을 찾을 때 sdpcntrl(8)을 사용한다.

```
# rfcomm_pppd -a 00:80:37:29:19:a4 -c -C dun -l rfcomm-dialup
```

PPP(LAN)로 네트워크 접근 서비스를 제공하기 위해 sdpd(8) 서버가 실행되고 있어야 된다. LAN 클라이언트의 위한 새로운 엔트리를 /etc/ppp/ppp.conf 파일에 생성하고 예제는 rfcomm\_pppd(8) 매뉴얼 페이지를 참고한다. 마지막으로 RFCOMM PPP 서버는 유효한 RFCOMM 채널 번호에서 실행한다. RFCOMM PPP 서버는 로컬 SDP 데몬을 통해 자동으로 블루투스 LAN 서비스에 등록된다. 아래 예제는 RFCOMM PPP 서버를 어떻게 시작하는지 보여준다.

```
# rfcomm_pppd -s -C 7 -l rfcomm-server
```

## 24.4.9 OBEX Object Push (OPUSH) 프로파일

OBEX 는 무선 장치 사이에서 파일을 전송하는데 광범위하게 사용되는 프로토콜이다. 주된 용도로 노트북이나 Palm 무선 장치 사이에서는 일반적인 파일 전송에, 핸드폰과 다른 장치 사이에서는 PIM 어플리케이션으로 명함이나 달력 엔트리를 전송하는 적외선 통신에 사용된다.

OBEX 서버와 클라이언트는 [comms/obexapp](#) 포트에서 사용할 수 있는 **obexapp** 패키지를 사용할 수 있다.

OBEX 클라이언트는 OBEX 서버에 주체를 보내(push)고 받는(pull)데 사용된다. 예를 들면 주체는 명함이나 시간 약속일 것이다. OBEX 클라이언트는 SDP 를 통해 원격 장치에서 RFCOMM 채널 번호를 가져올 수 있다. 여기서 RFCOMM 채널 번호 대신 서비스 이름을 지정할 수 있다. 지원되는 서비스 이름은 IrMC, FTRN 그리고 OPUSH 다. RFCOMM 채널을 번호로 지정할 수 있다. 아래는 장치 정보 주체를 핸드폰에서 가져오고 새로운 주체(명함)를 전화의 디렉터리에 보내는 OBEX 세션의 예제다.

---

```
% obexapp -a 00:80:37:29:19:a4 -C IrMC
obex> get
get: remote file> telecom/devinfo.txt
get: local file> devinfo-t39.txt
Success, response: OK, Success (0x20)
obex> put
put: local file> new.vcf
put: remote file> new.vcf
Success, response: OK, Success (0x20)
obex> di
Success, response: OK, Success (0x20)
```

OBEX 주체를 보내(push)는 서비스를 제공하려면 sdpd(8) 서버가 실행되고 있어야 된다. 입력되는 모든 주체가 저장되는 root 폴더를 생성해야 되고 기본 경로는 /var/spool/obex 다. 마지막으로 유효한 RFCOMM 채널번호에서 OBDEX 서버를 시작한다. 아래 예제에서 OBEX 서버를 어떻게 시작하는지 보여준다.

```
# obexapp -s -C 10
```

## 24.4.10 시리얼 포트 프로파일(SPP)

시리얼 포트 프로파일(SPP)은 블루투스 장치가 RS232(또는 비슷한) 시리얼 케이블 에뮬레이션을 수행하도록 한다. 이 프로파일로 다루는 시나리오는 케이블을 대신하여 가상 시리얼 포트 개념으로 블루투스를 사용하는 레거시 어플리케이션이다.

rfcomm\_sppd(1) 유틸리티는 시리얼 포트 프로파일 실행한다. pseudo tty 가 가상 시리얼 포트 개념에 사용된다. 아래 예제는 원격 장치의 시리얼 포트 서비스에 어떻게 연결하는지 보여준다. RFCOMM channel-rfcomm\_sppd(1)는 SDP 를 통해 원격 장치에서 가져올 수 있기 때문에 지정할 필요가 없다. 이것을 무시하려면 명령어 라인에서 RFCOMM 채널을 지정한다.

```
# rfcomm_sppd -a 00:07:E0:00:0B:CA -t /dev/tty6
rfcomm_sppd[94692]: Starting on /dev/tty6...
```

연결된 pseudo tty 는 시리얼 포트 로 사용할 수 있다.

---

```
# cu -l ttty6
```

## 24.4.11 문제 해결

### 24.4.11.1 원격 장치에 연결하지 못한다.

예전의 어떤 블루투스 장치는 role 스위칭을 지원하지 못한다. 기본적으로 FreeBSD 가 새로운 연결을 허용했을 때 블루투스 장치는 role 스위칭을 수행해서 마스터가 된다. 이 기능을 지원하지 못하는 장치는 연결할 수 없다. 새로운 연결이 완료되면 role 스위칭이 수행되기 때문에 원격 장치가 role 스위칭을 지원하는지 확인하는 것은 불가능하다. 로컬에서 role 스위칭을 비활성하는 HCI 옵션이 있다.

```
# hccontrol -n ubt0hci write_node_role_switch 0
```

### 24.4.11.2 무엇인가 잘못되면 확인할 수 있는가?

확인할 수 있다. [http://www.geocities.com/m\\_evmenkin/](http://www.geocities.com/m_evmenkin/)에서 다운로드 할 수 있는 `hcidump-1.5` 패키지를 사용한다. `hcidump` 유틸리티는 `tcpdump(1)`과 비슷하다. 블루투스 패킷의 내용을 터미널에 표시하고 블루투스 패킷을 파일로 덤프하는데 사용할 수 있다.

## 24.5 브리지

### 24.5.1 소개

IP 서브넷을 생성하지 않고 하나의 물리적인 네트워크(이더넷 구간과 같은)를 두 개의 네트워크 구간으로 나누고, 라우터를 사용하여 두 개의 구간을 연결하는 것이 유용한 경우가 있다. 두 개의 네트워크를 연결하는 장치를 여기서는 "브리지"라고 부른다. 두 개의 네트워크 인터페이스를 가진 FreeBSD 는 브리지로 동작할 수 있다.

브리지는 각 네트워크 인터페이스의 맥 어드레스(이더넷 주소)로 동작한다. 브리지는 소스와 목적지가 다른 네트워크에 있을 때만 트래픽을 포워드 한다.

---

많은 것을 고려하면 브리지는 매우 적은 포트를 가진 이더넷 스위치와 같다.

## 24.5.2 브리지는 어떤 곳에 적당한가

요즘에 일반적으로 브리지가 사용되는 두 가지 상황이 있다.

### 24.5.2.1 구간의 트래픽이 높을 때

첫 번째 경우는 물리적인 네트워크 구간이 트래픽으로 과부하 상태지만 어떤 이유에서든 서브넷으로 네트워크를 나누고 라우터로 서브넷 사이를 연결하지 않을 때다.

신문사를 예로 들면 편집부와 생산부서가 같은 서브 네트워크에 있다고 가정하자. 편집부의 모든 유저는 서버 A를 파일 서비스로 사용하고 생산부서의 유저들은 서버 B를 사용한다. 이더넷을 사용하여 모든 유저가 연결되므로 네트워크의 부하가 심해서 느리고 종종 다운된다.

하나의 네트워크에서 편집부 유저는 한쪽 네트워크 구간을 사용하고 생산부서 유저를 다른 네트워크를 사용하도록 분리할 수 있다면 두 네트워크 구간을 브리지로 연결할 수 있다. 네트워크 패킷의 목적지가 브리지의 다른 쪽 인터페이스(다른 부서)에 있을 때만 다른 네트워크로 보내기 때문에 각 네트워크 구간의 과부하를 줄일 수 있다.

### 24.5.2.2 방화벽 필터링이 필요한 곳

일반적인 두 번째 경우는 네트워크 주소 변환(NAT) 기능이 없는 방화벽이 필요한 곳이다.

DSL이나 ISDN으로 ISP에 연결된 작은 회사를 예로 들어보자. 이 회사는 ISP로부터 13개의 공인 IP를 할당 받았고 네트워크에 10대의 PC가 있다. 이 상황에서 라우터 기반 방화벽을 사용하는 것은 서브넷 문제 때문에 어렵다.

IP 번호 문제 없이 DSL/ISDN 라우터의 다운(down) 스트림 경로에 브리지 기반 방화벽을 설치할 수 있다.

---

## 24.5.3. 브리지 설정

### 24.5.3.1 네트워크 인터페이스 카드 선택

브리지는 기능상 최소 두 개의 네트워크 카드가 필요하다. 불행히 FreeBSD 4.0에서는 모든 네트워크 인터페이스 카드가 브리지를 지원하지 못한다. 지원되는 카드의 자세한 리스트는 `bridge(4)`를 읽는다.

계속 진행하기 전에 두 개의 네트워크 카드를 설치하고 테스트한다.

### 24.5.3.2 커널 설정 변경

커널이 브리지를 지원하도록 다음 라인을 추가한다:

```
options BRIDGE
```

커널 설정파일에 명시하고 커널을 다시 빌드한다.

### 24.5.3.3 방화벽 기능

브리지에 방화벽 기능을 추가하여 사용하려면 `IPFIREWALL` 옵션도 추가해야 된다.

브리지를 방화벽으로 설정하는 일반적인 정보는 14 장을 읽는다.

IP 패킷이 아닌 다른 패킷(ARP 와 같은)이 브리지를 통과하도록 하려면 방화벽 옵션을 설정해야 된다. 이 옵션은 `IPFIREWALL_DEFAULT_TO_ACCEPT`이고 모든 패킷이 접근할 수 있도록(방화벽 기능을 활성화하면 모든 패킷을 거부한다) 방화벽의 기본 룰을 변경한다. 설정하기 전에 변경하는 방법과 변경하려는 룰의 의미를 알고 있어야 된다.

### 24.5.3.4 트래픽(대역폭) 제어 지원

브리지를 트래픽 제어기로 사용하려면 커널 설정에 `DUMMYNET` 옵션을 추가해야 된다. 더 많은 정보는 `dumynet(4)`를 읽는다.

---

## 24.5.4 브리지 활성화

시작할 때 브리지가 활성화되도록 다음 라인을 `/etc/sysctl.conf` 에 추가한다:

```
net.link.ether.bridge=1
```

그리고 지정된 인터페이스에서 브리지가 활성화되도록 다음 라인도 추가한다(*if1* 과 *if2* 는 여러분의 네트워크 인터페이스에 맞도록 변경한다):

```
net.link.ether.bridge_cfg=if1,if2
```

브리지가 ipfw(8)로 패킷 필터링을 하게 하려면 다음 라인을 추가한다:

```
net.link.ether.bridge_ipfw=1
```

FreeBSD 5.2-RELEASE 와 이후의 버전은 다음 라인을 대신 사용한다:

```
net.link.ether.bridge.enable=1  
net.link.ether.bridge.config=if1,if2  
net.link.ether.bridge.ipfw=1
```

## 24.5.5 다른 정보

네트워크에서 텔넷으로 브릿지에 접근하려면 네트워크 카드 하나에 IP 주소를 할당하면 된다. 두 카드의 주소를 동일하게 할당하지 않는다.

네트워크에 여러 개의 브리지가 있다면 두 대의 워크스테이션간의 경로는 하나뿐이어야 된다. 기술적으로 이 의미는 스패닝 트리(spanning tree: 브리지가나 스위치 네트워크에서 발생한 논리적인 루프를 감지하고 계산하는데 사용되는 IEEE 802.1D 표준) 링크 관리가 지원되지 않는다.

브리지는 잠재적으로 ping 응답 시간을 증가시킬 수 있고 특히 한쪽 구획에서 다른 구획으로의 트래픽을 증가시킬 수 있다.

---

## 24.6 디스크 없이 운용하기(Diskless Operation)

FreeBSD 머신은 네트워크를 통해 부팅할 수 있기 때문에 NFS 서버로부터 마운트 한 파일 시스템을 사용하여 로컬 디스크 없이 운용할 수 있다. 표준 설정파일 이외에 시스템을 수정할 필요 없다. 이런 시스템은 필요한 모든 요소를 사용할 수 있도록 준비 되어 있기 때문에 상당히 설정하기 쉽다:

- 네트워크를 통해 커널을 로드 할 수 있는 두 가지 방법이 있다:
  - ✓ PXE: Intel® Preboot Execution Environment system은 네트워크 카드나 마더보드에 다양한 기능이 내장된 부트 롬이다. 더 자세한 사항은 pxeboot(8)를 본다.
  - ✓ **etherboot** 포트(net/etherboot)는 네트워크를 통해 ROM-able 코드를 부트 커널에 생성한다. 코드를 네트워크 카드의 부트 PROM이나 로컬 플로피(또는 하드) 디스크 드라이브에 저장할 수 있으며 사용중인 MS-DOS 시스템에서 로드 할 수 있다. 다양한 네트워크 카드가 지원된다.
- 샘플 스크립트(/usr/share/examples/diskless/clone\_root)는 생성하기 쉽고 서버에서 워크스테이션의 root 파일시스템을 관리한다. 이 스크립트를 약간 수정해야겠지만 시작하기 쉽다.
- /etc에있는 표준 시스템 시작파일은 디스크가 없는 시스템을 감지해서 시스템의 부팅을 지원한다.
- 스왑이 필요하다면 NFS 파일이나 로컬 디스크를 사용할 수 있다.

디스크가 없는 워크스테이션을 설정하는 여러 가지 방법이 있다. 다양한 요소가 관련이 있겠지만 대부분은 로컬 시스템(클라이언트)의 상황에 맞게 수정할 수 있다. 다음은 표준 FreeBSD 시작 스크립트와의 호환성을 강조하여 전체 시스템의 설정 변수를 설명한 것이다. 설명하는 시스템은 다음과 같은 특성을 가지고 있다:

- 디스크가 없는 워크스테이션은 읽기 전용으로 공유되어 있는 root 파일시스템과 /usr을 사용한다.

---

root 파일시스템은 디스크가 없이 운용하는데 알맞게 수정한 설정파일이거나 워크스테이션이 가지고 있던 설정파일을 수정한 표준 FreeBSD root 의 복사본이다.

root 파일시스템에서 쓰기 기능이 필요한 곳은 FreeBSD 4.X 에서는 mfs(8), FreeBSD 5.X 에서는 md(4) 파일시스템으로 덮어써야 된다. 이렇게 변경해도 시스템이 재 부팅하면 원상태로 복구 된다.

- 커널은 전송되어 **Etherboot**나 PXE 프로그램에 의해 로드 되고 상화에 따라 두 프로그램 중에서 한가지가 사용된다.

**주의:** 앞에서 설명하였듯이 이 시스템은 보안상 안전하지 않다. 안전한 네트워크에 두고 다른 호스트와는 신뢰관계를 맺지 않는다.

이번 장의 모든 정보는 FreeBSD 4.9-RELEASE 와 5.2.1-RELEASE 에서 테스트됐다. 이 글은 주로 4.X 사용에 중점을 두고 작성되었다.

## 24.6.1 백그라운드 정보

디스크가 없는 워크스테이션을 설정하는 것은 상당히 직선적이고 에러가 발생하기 쉽다. 그리고 여러 가지 이유가 있기 때문에 원인을 찾아내기도 어렵다:

- 컴파일 시간 옵션은 실행할 때 다르게 동작할 수 있다.
- 에러 메시지는 종종 애매하거나 없을 것이다.

이러한 상황에서 메커니즘과 관련된 백그라운드 정보는 발생할 수 있는 문제 해결에 매우 유용하다.

성공적인 부팅을 위해 여러 가지 동작이 수행되어야 한다:

- 머신에 IP 주소, 실행할 수 있는 파일 이름, 서버 이름 그리고 root 경로 등과 같은 초기화 매개변수가 필요하다. 이 정보는 DHCP 또는 BOOTP 프로토콜을 사용하면 된다. DHCP는 BOOTP에서 확장된 것이기 때문에 포트 번호와 기본 패킷 포맷이 같다.



---

BOOTP 만 사용하도록 시스템을 설정할 수도 있다. bootpd(8) 서버 프로그램은 FreeBSD 기본 시스템에 포함되어있다.

그러나 DHCP 는 BOOTP(PXE 를 사용할 것이고 디스크가 없이 운용하는데 직접 연관은 없지만 다양한 기능을 추가하여 사용할 수 있는 쓸만한 설정파일) 이상의 다양한 장점을 가지고 있기 때문에, 여기서는 주로 DHCP 를 설정하는 방법 위주로 설명하고 가능하다면 bootpd(8)을 사용한 예제도 보여줄 것이다. 샘플 설정에서는 **ISC DHCP** 소프트웨어 패키지를(3.0.1.r12 가 테스트 서버에 설치되어 있다) 사용한다.

- 머신은 하나나 여러 개의 프로그램을 로컬 메모리에 전송해야 된다. 여기에 TFTP나 NFS가 사용된다. TFTP와 NFS를 선택하는 것은 컴파일 시간 옵션과 관련이 있다. 일반적으로 발생하는 에러의 원인은 잘못된 프로토콜에 파일이름을 지정하는 것이다: TFTP는 보통 서버 디렉터리의 한곳에 있는 모든 파일을 전송하기 때문에, 파일 이름은 이 디렉터리와 상대적이지만 NFS는 절대 파일 경로를 사용한다.
- 부트스트랩 중간의 프로그램과 커널을 초기화하고 실행해야 된다. 이 부분에서 몇 가지를 변경해야 된다:
  - ✓ PXE는 FreeBSD의 세 번째 로더의 수정된 버전인 pxeboot(8)을 로드 한다. 전송 제어를 하기 전에 loader(8)은 시스템 시작에 필요한 대부분의 매개변수를 가져와서 커널 환경에 둔다. 여기서 GENERIC 커널을 사용해도 된다.
  - ✓ **Etherboot**는 특별한 준비 없이 커널을 직접 로드한다. 특정 옵션으로 커널을 빌드 해야 된다.

PXE 와 **Etherboot** 는 4.X 시스템에서와 동일하게 동작한다. 왜냐하면 5.X 커널은 보통 loader(8)에게 더 많은 태스크를 시키기 때문에 PXE 는 5.X 시스템을 지원한다.

BIOS 와 네트워크 카드가 PXE 를 지원한다면 이것을 사용하는 것이 좋다. 그렇지만 **Etherboot** 로 5.X 시스템을 시작하는 것도 가능하다.

- 마지막으로 머신이 파일시스템에 접근해야 된다. 이 모든 경우에 NFS가 사용된다.

---

그리고 `diskless(8)` 매뉴얼 페이지도 참고한다.

## 24.6.2 명령 설정

### 24.6.2.1 ISC DHCP 를 사용한 설정

ISC DHCP 서버는 BOOTP 와 DHCP 요청에 응답할 수 있다.

릴리즈 4.9 에서 **ISC DHCP 3.0** 은 기본 시스템에 포함되어 있지 않다. [net/isc-dhcp3-server](http://net/isc-dhcp3-server) 포트나 적당한 패키지를 먼저 설치해야 된다. 포트와 패키지에 대한 일반적인 정보는 4 장을 참고한다.

ISC DHCP 가 설치된 후 실행하기 위해 설정파일이 필요하다(보통 `/usr/local/etc/dhcpd.conf`). 호스트 `margaux` 는 **Etherboot** 를 그러나 호스트 `corbieres` 는 PXE 를 사용하는 예제가 있다:

```
default-lease-time 600;
max-lease-time 7200;
authoritative;

option domain-name "example.com";
option domain-name-servers 192.168.4.1;
option routers 192.168.4.1;

subnet 192.168.4.0 netmask 255.255.255.0 {
    use-host-decl-names on; ❶
    option subnet-mask 255.255.255.0;
    option broadcast-address 192.168.4.255;

    host margaux {
        hardware ethernet 01:23:45:67:89:ab;
        fixed-address margaux.example.com;
        next-server 192.168.4.4; ❷
        filename "/data/misc/kernel.diskless"; ❸
    }
}
```

```
option root-path "192.168.4.4:/data/misc/diskless"; ❹
}
host corbieres {
    hardware ethernet 00:02:b3:27:62:df;
    fixed-address corbieres.example.com;
    next-server 192.168.4.4;
    filename "pxeboot";
    option root-path "192.168.4.4:/data/misc/diskless";
}
}
```

- ❶ 이 옵션은 **dhcp** 에게 디스크가 없는 호스트의 호스트 이름으로 *host* 선언부의 값을 보내도록 한다. 그렇지 않으면 *host* 선언에 *option host-name margaux* 를 추가한다.
- ❷ *next-server* 는 로더나 커널 파일을 로딩하기 위해 사용하는 TFTP 또는 NFS 서버를 가리킨다(기본값은 DHCP 서버에서와 같은 호스트를 사용한다).
- ❸ *filename* 은 **Etherboot** 또는 PXE 가 다음 실행단계에서 로드 해야 될 파일을 정의한다. 사용하는 전송 방법에 따라 지정해야 된다. **Etherboot** 는 NFS 나 TFTP 를 사용하도록 컴파일 할 수 있고 FreeBSD 포트는 기본적으로 NFS 로 설정된다. 여기서 상대적인 파일 이름을 사용하는 것은 PXE 가 TFTP 를 사용하기 때문이다(이것은 TFTP 서버 설정과 관련이 있겠지만 대부분은 일반적인 설정을 가지고 있다). 그리고 PXE 는 커널이 아닌 pxeboot 도 로드 한다. pxeboot 를 로딩하는 것처럼 상당히 흥미 있는 것들이 FreeBSD CD-ROM 의 /boot 디렉터리에(pxeboot(8)은 GENERIC 커널을 로드 할 수 있기 때문에 PXE 를 사용하여 원격 CD-ROM 에서 부팅하는 것도 가능하다) 있다.
- ❹ *root-path* 옵션은 보통 NFS 표기에서 root 파일시스템 경로를 정의한다. PXE 를 사용할 때 BOOTP 커널 옵션을 활성화하지 않았다면 호스트의 IP 를 그대로 둘 수 있다. NFS 서버는 TFTP 와 같다.

### 24.6.2.2 BOOTP 사용 설정

여기 /etc/bootptab 에서 찾을 수 있는 **bootpd** 와 동일한 설정이 있다.

---

BOOTP 를 사용하기 위해 기본 옵션이 아닌 *NO\_DHCP\_SUPPORT* 옵션으로 **Etherboot** 를 컴파일 하는 것에 주의하고 PXE 는 DHCP 가 필요하다. **bootpd** 의 확실한 장점은 bootpd 가 기본 시스템에 있다는 것이다.

```
.def100:W
    :hn:ht=1:sa=192.168.4.4:vm=rfc1048:W
    :sm=255.255.255.0:W
    :ds=192.168.4.1:W
    :gw=192.168.4.1:W
    :hd="/tftpboot":W
    :bf="/kernel.diskless":W
    :rp="192.168.4.4:/data/misc/diskless":

margaux:ha=0123456789ab:tc=.def100
```

### 24.6.2.3 Etherboot 로 부트 프로그램 준비

Etherboot 의 웹 사이트(<http://etherboot.sourceforge.net/>)에는 주로 리눅스 시스템을 위한 문서가 많이 있지만 그래도 유용한 정보가 많이 있다. 다음은 FreeBSD 시스템에서 **Etherboot** 를 어떻게 사용할지 요점을 제시한다.

첫째로 [net/etherboot](#) 패키지나 포트를 설치한다. **Etherboot** 포트는 보통 `/usr/ports/net/etherboot` 에서 찾을 수 있다. 시스템에 포트 트리가 설치되어 있다면 이 디렉터리에서 `make` 명령만 입력하고 모든 것을 주의 깊게 관찰한다. 그 밖의 포트와 패키지에 대한 정보는 4 장을 참고한다.

**Etherboot** 소스 디렉터리의 Config 파일을 수정하여 etherboot 설정을(다시 말해 NFS 대신 TFTP 를 사용하도록) 변경할 수 있다.

여기서는 부트 플로피를 사용할 것이다. 다른 방법(PROM 이나 DOS 프로그램) **Etherboot** 문서를 참고한다.

부트 플로피를 생성하는 방법은 **Etherboot** 가 설치된 머신의 드라이브에 플로피를 넣고, 현재 디렉터리를 **Etherboot** 트리의 src 디렉터리로 변경한 후 다음 명령을 입력한다:

---

```
# gmake bin32/devicetype.fd0
```

*devicetype*은 디스크가 없는 워크스테이션의 이더넷 카드 종류를 말한다. 정확한 *devicetype*은 같은 디렉터리의 NIC 파일을 참고한다.

#### 24.6.2.4 PXE 로 부팅

기본적으로 pxeboot(8) 로더는 NFS 로 커널을 로드 한다. /etc/make.conf 에 *LOADER\_TFTP\_SUPPORT* 옵션을 지정하여 TFTP 를 대신 NFS 를 사용하도록 컴파일 할 수 있다. 관련된 지시 사항은 /etc/defaults/make.conf 의(또는 5.X 시스템은 /usr/share/examples/etc/make.conf) 설명을 본다.

시리얼 콘솔로 디스크가 없는 머신을 설정하는데 유용하고 문서화되지 않은 *BOOT\_PXELDR\_PROBE\_KEYBOARD* 와 *BOOT\_PXELDR\_ALWAYS\_SERIAL*(후자는 FreeBSD 5.X 에만 있다) 두 가지 make.conf 옵션이 있다.

머신이 시작될 때 PXE 를 사용하려면 보통 BIOS 설정에서 *Boot from network* 옵션을 선택하거나 PC 를 초기화하는 동안 기능 키를 입력한다.

#### 24.6.2.5 TFTP 와 NFS 서버 설정

TFTP 를 사용하도록 Etherboot 나 PXE 를 설정했다면 파일 서버에서 tftpd 를 활성화해야 된다:

- ① tftpd가 파일을 제공할 디렉터리를 생성한다. 예: /tftpboot
- ② /etc/inetd.conf에 다음 라인을 추가한다:

```
tftp dgram udp wait root /usr/libexec/tftpd tftpd -s /tftpboot
```

**Note:** 어떤 PXE 버전은 TCP 버전의 TFTP 를 사용해야 되는 경우가 있다. 이 경우 *gram udp* 를 *stream tcp* 로 교체한다.

- ③ inetd가 설정파일을 다시 읽도록 한다:  
# kill -HUP `cat /var/run/inetd.pid`

---

tftpboot 디렉터리를 서버의 원하는 곳에 생성할 수 있다. 이 위치를 inetd.conf 와 dhcpd.conf 두 곳에 설정해야 된다.

NFS 를 활성화하고 NFS 서버로 적당한 파일시스템을 공유한다.

- ① 다음 라인을 /etc/rc.conf에 추가한다:

```
nfs_server_enable="YES"
```

- ② 디스크가 없는 워크스테이션에 파일시스템을 공유하는 것은 다음 라인을 /etc/exports에 추가하면 된다(볼륨의 마운트 위치를 조정하고 *margaux corbieres* 대신 디스크가 없는 워크스테이션의 이름을 사용한다):

```
/data/misc -alldirs -ro margaux corbieres
```

- ③ **mountd**가 설정파일을 다시 읽게 한다. 첫 단계에서 실제로 /etc/rc.conf를 사용하여 NFS를 활성화해야 되고 재 부팅하는 대신 다음 명령을 사용할 수 있다.

```
# kill -HUP `cat /var/run/mountd.pid`
```

### 24.6.2.6 디스크가 없는 커널 빌드

**Etherboot** 를 사용한다면 디스크가 없는 클라이언트를 위해 다음 옵션을 추가하여 커널 설정파일을 생성해야 된다(일반적으로):

```
options      BOOTP          # Use BOOTP to obtain IP address/hostname
options      BOOTP_NFSROOT # NFS mount root filesystem using BOOTP info
```

또한 *BOOTP\_NFSV3*, *BOOT\_COMPAT* 그리고 *BOOTP\_WIRED\_TO*를 사용해도 된다(4.X 는 LINT 를 5.X 는 NOTE 참고한다).

**Note:** PXE 를 사용할 때 위의 옵션으로 커널을 빌드할 필요는 없다(그러나 권장한다). 이들 옵션을 활성화하면 특별한 경우에 pxeboot(8)이 검색한 것과 새로운

---

값이 일치하지 않아서 발생할 수 있는 위험을 줄이고, 커널이 시작되는 동안 더 많은 DHCP 요청을 허용한다.

이들 옵션은 사실 커널에서 DHCP와 BOOTP를 일반적으로 활성화하기 때문에 역사적으로 약간의 혼란을 야기했다(강제로 BOOTP 또는 DHCP를 사용하도록 할 수 있다).

커널을 빌드(8장을 본다)해서 `dhcpd.conf`에 지정한 곳에 복사한다.

**Note:** Etherboot로 로드할 수 있도록 5.X 커널에 장치 힌트(device hint)가 컴파일되어 있어야 된다. 설정파일에 다음 옵션을 추가한다(NOTES 파일을 본다):

```
hints      "GENERIC.hints"
```

### 24.7.2.7 root 파일시스템 준비

`dhcpd.conf`의 `root-path`에 디스크가 없는 워크스테이션을 위한 root 파일시스템을 생성해야 된다.

#### [root 파일시스템을 생성하는 두 가지 방법]

##### 1. clone\_root 스크립트를 사용하는 방법

이 방법이 root 파일시스템을 생성하는 가장 빠른 방법이지만 현재 FreeBSD 4.X에서만 지원된다. 이 쉘 스크립트는 `/usr/share/examples/diskless/clone_root`에 있으며 약간의 수정이 필요하고 최소한 파일시스템을 생성할 위치가 어느 곳인지(`DEST` 변수) 지정해야 된다.

필요한 사항은 스크립트 최 상단의 설명을 참고한다. 이곳에는 기본 시스템을 어떻게 빌드해야 되고 디스크가 없는 운용, 서브 네트워크 또는 각각의 워크스테이션 버전에 맞게 어떤 파일을 선택해야 되는지 설명한다. 또한 디스크가 없는 운용에 맞는 예제 설정파일 `/etc/fstab`와 `/etc/rc.conf`도 제공한다.

`/usr/share/examples/diskless`의 README 파일에 다양한 백그라운드 정보가 있지만 `diskless` 디렉터리의 다른 예제는 `clone_root`가 사용하는 것과 `/etc`의 시스템 시작 스크립트와 구별되는 설정 방법을 제시한다. 여기서 제시하는 `rc` 스크립트를 수정하는 방법을 사용하지 않는다면 이들 문서는 참고만 한다.

## 2 표준 make world 프로시저 사용

이 방법은 FreeBSD 4.X 나 5.X 에 적용할 수 있고 최초 시스템을(root 파일시스템뿐만 아니고) DESTDIR 에 완벽하게 설치한다. 단순히 다음 스크립트를 실행한다:

```
#!/bin/sh
export DESTDIR=/data/misc/diskless
mkdir -p ${DESTDIR}
cd /usr/src; make world && make kernel
cd /usr/src/etc; make distribution
```

실행이 끝나면 필요에 따라 /etc/rc.conf 와 /etc/fstab 를 수정해서 DESTDIR 에 둔다.

### 24.6.2.8 스왑 설정

필요하다면 NFS 를 통해 서버에 있는 스왑 파일에 접근할 수 있다. 일반적으로 여기서 사용되는 방법 중 몇 가지는 5.X 에서는 더 이상 진행되지 않는다.

#### [NFS 를 통해 스왑 파일사용하기]

##### 1. FreeBSD 4.X 에서 NFS 스왑 사용

스왑 파일 위치와 크기는 FreeBSD 에만 있는 BOOTP/DHCP 옵션 128 과 129 로 지정할 수 있다. ISC DHCP 3.0 이나 **bootpd** 의 설정파일 예제는 다음 절차를 따른다:

- ① dhcpd.conf에 다음 라인을 추가한다:

```
# Global section
option swap-path code 128 = string;
option swap-size code 129 = integer 32;

host margaux {
    ... # Standard lines, see above
    option swap-path "192.168.4.4:/netswapvolume/netswap";
    option swap-size 64000;
}
```



---

*swap-path*는 스왑 파일이 있는 디렉터리 경로다. 각 파일은 *swap.client-ip*라는 이름으로 되어있다.

**dhcpcd**의 예전 버전은 *option option-128* "...처럼 더 이상 지원되지 않는 구문을 사용한다.

대신 */etc/bootptab*은 다음 구문을 사용한다:

```
T128="192.168.4.4:/netswapvolume/netswap":T129=64000
```

**Note:** */etc/bootptab*에서 스왑 크기는 16 진수 포맷으로 표현해야 된다.

- ② NFS 스왑 파일 서버에서 스왑 파일을 생성한다

```
# mkdir /netswapvolume/netswap
# cd /netswapvolume/netswap
# dd if=/dev/zero bs=1024 count=64000 of=swap.192.168.4.6
# chmod 0600 swap.192.168.4.6
```

*192.168.4.8*은 디스크가 없는 클라이언트 IP 다.

- ③ NFS 스왑 파일 서버의 */etc/exports*에 다음 라인을 추가한다:

```
/netswapvolume -maproot=0:10 -alldirs margaux corbieres
```

그리고 위에서처럼 **mountd**가 공유 파일을 다시 읽도록 한다.

## 2. FreeBSD 5.X에서 NFS 스왑 사용

커널은 부팅할 때 NFS 스왑을 활성화하지 않는다. 스왑은 쓰기가 가능한 파일시스템으로 마운트하여, 스왑을 생성한 후 활성화하는 시작 스크립트로 활성화해야 된다. 적당한 크기로 스왑을 생성하려면 다음과 같이 실행한다:

```
# dd if=/dev/zero of=/path/to/swapfile bs=1k count=1 oseek=100000
```

---

활성화하려면 rc.conf 에 다음 라인을 추가해야 된다:

```
swapfile=/path/to/swapfile
```

### 26.6.2.9 잡다한 문제들

다음과 같이 특별한 경우가 있을 것이다.

① /usr를 읽기 전용으로 운용하기

디스크가 없는 워크스테이션이 X를 실행할 수 있도록 설정되어있다면 기본적으로 /usr 에 에러 로그를 남기는 xdm 설정파일을 수정해야 된다.

② FreeBSD가 아닌 서버 운용하기

root 파일시스템 서버가 FreeBSD 에서 운용되지 않으면 FreeBSD 머신에서 root 파일시스템을 생성해서 tar 나 cpio 로 타겟 머신에 복사해야 된다.

이 경우 /dev 에 있는 특수 파일의 major/minor 정수 크기가 달라지는 문제가 있다. 이 문제를 해결하는 것은 FreeBSD 가 아닌 서버에서 디렉터리를 공유해서 FreeBSD 머신에 마운트하고, 정확한 장치 엔트리를 생성하도록 FreeBSD 머신에서 MAKEDEV 를 실행한다(FreeBSD 5.0 과 이 후 버전은 devfs(5)로 장치 노드를 생성하기 때문에 이들 버전에서 MAKEDEV 를 실행할 필요가 없다).

## 24.7 ISDN

ISDN 기술과 하드웨어에 대한 자세한 정보는 Dan Kegel 의 ISDN 페이지(<http://www.alumni.caltech.edu/~dank/isdn/>)에 있다.

ISDN 의 간단한 로드 맵은 다음을 사항을 따른다:

- 여러분이 유럽에 살고 있다면 ISDN 카드 섹션을 확인해 보기 바란다.
- 다이얼-업 기반의 ISDN으로 인터넷 공급자(전용선을 기반이 아닌)를 통해

---

인터넷에 연결할 계획이라면 터미널 아답터를 확인해본다. 이 방법을 사용하면 큰 유연성을 가지고 있어서 인터넷 공급자를 바꾸더라도 별문제가 없을 것이다.

- 두 개의 랜을 같이 연결하거나 ISDN 전용선으로 인터넷에 연결한다면 stand alone 라우터/브리지 옵션을 고려해본다.

사용할 솔루션을 결정하기 가격이 중요한 요소가 되고 있다. 아래 설명은 최소 가격에서 최대 가격까지 순서대로 제시해 줄 것이다.

## 24.7.1 ISDN 카드

FreeBSD 의 ISDN 은 passive 카드만을 사용하는 DSS1/Q.931(또는 Euro-ISDN) 표준만 지원한다. FreeBSD 4.4 부터는 최초로 지원된 Primary Rate(PRI) ISDN 카드를 포함하여 펌웨어가 다른 신호 프로토콜을 지원하는 몇몇 active 카드도 지원된다.

**isdn4bsd** 소프트웨어는 raw HDLC(high level data link control 의 약어로 동기식 고속 데이터 전송을 능률적으로 실행하기 위한 제어방식)를 사용한 IP 또는 동기식 PPP 를 사용하여 다른 ISDN 라우터에 연결할 수 있다: *isppp* 로 커널 PPP 를 사용하거나 수정된 *sppp(4)* 드라이버를 사용하거나 유저 기반 *ppp(8)*을 사용한다. 유저 기반 *ppp(8)*를 사용하여 두 개나 그 이상의 ISDN B-채널에서 **채널 본딩(channel bonding)**도 가능하다. 300 Baud 모뎀 소프트웨어처럼 자동 응답 전화 어플리케이션도 사용할 수 있다.

**채널 본딩(channel bonding):** beowulf 프로젝트에서 선보였던 이 기술은 이더넷 2 개를 같은 IP 로 할당하여 2 배의 전송속도를 낸다. 이더넷 1 개는 outbound 로 다른 하나는 Inbound 로 사용하는 방식이다.

많은 ISDN PC 카드가 FreeBSD 에서 지원하기 때문에 유럽을 포함하여 전 세계에서 성공적으로 사용되고 있다.

지원되는 passive ISDN 카드는 Infineon(이전 Siemens) ISAN/HSCX/IPAC ISDN 칩셋, Cologne 칩(ISA 버스만), W6692 칩의 Winbond 의 PCI 카드, Tiger300/320/ISAC 칩셋과 결합된 카드 그리고 AVM Fritz!Catd PCI v.1.0 과 AVM Fritz!Card PnP 와 같은 특정 벤더 칩셋 기반 카드도 가능하다.

현재 무난하게 지원되는 ISDN 카드는 AVM B1(ISA 와 PCI) BRI 카드와 AVM T1 PCI PRI

---

카드다.

**isdn4bsd** 에 대한 문서는 FreeBSD 시스템의 `/usr/share/examples/isdn/` 디렉터리를 찾거나 **erratas** 와 **isdn4bsd**(<http://people.freebsd.org/~hm/>) 핸드북과 같은 문서 정보가 있는 **isdn4bsd**(<http://www.freebsd-support.de/i4b/>) 홈페이지에서 찾는다.

현재 지원되지 않는 ISDN PC 카드나 **isdn4bsd** 성능을 개선하는 것처럼 다른 ISDN 프로토콜을 지원하도록 하려면 Hellmuth Michaelis <[hm@FreeBSD.org](mailto:hm@FreeBSD.org)>에게 연락한다.

**isdn4bsd** 설치, 설정과 문제 해결에 대한 질문은 `freebsd-isdn` 메일링 리스트(<http://lists.freebsd.org/mailman/listinfo/freebsd-isdn>)를 이용할 수 있다.

## 24.7.2 ISDN 터미널 아답터

터미널 아답터(TA)는 일반적인 전화라인을 사용하는 ISDN 모뎀이다.

대부분의 TA 는 표준 Hayes 모뎀 AT 명령 세트를 사용하므로 모뎀에 옮겨서 사용할 수 있다.

TA 는 기본적으로 연결과 속도가 예전 모뎀보다 더 빠른 점을 제외하고 모뎀과 동일하게 동작한다. 따라서 모뎀의 경우와 똑같이 PPP 를 설정해야 된다. 그리고 가능한 최고 속도로 시리얼 속도를 설정한다.

TA 를 사용하여 인터넷 공급자에게 연결하는 가장 큰 장점은 유동적인 PPP 를 사용할 수 있다는 것이다. IP 주소가 점점 부족해지기 때문에 대부분의 인터넷 서비스 공급자들은 더 이상 고정 IP 를 제공하지 않는다. 대부분의 stand-alone 라우터는 유동 IP 할당을 적용하지 않는다.

TA 의 연결특성과 안정성은 운용중인 PPP 데몬에 따라 다르다. 그래서 PPP 를 설정했다면 FreeBSD 머신에서 사용중인 모뎀에서 ISDN 으로 쉽게 업그레이드할 수 있다. 그러나 사용중인 PPP 프로그램에서 문제가 있었다면 이 문제는 업그레이드 후에도 지속된다.

최고의 안정성을 원한다면 userland PPP 가 아닌 커널 PPP 옵션을 사용한다.

다음 TA 는 FreeBSD 와 작동한다.

- 
- Motorola BitSurfer 와 Bitsurfer Pro.
  - Adtran.

TA 벤더들이 자신들의 제품에서 거의 모든 표준 모뎀 AT 명령이 실행되도록 노력하기 때문에 대부분의 다른 TA 도 정확하게 동작할 것이다.

외장형 TA 의 실질적인 문제는 모뎀처럼 고급 시리얼 카드가 컴퓨터에 필요하다.

시리얼 장치의 자세한 이해, 동기식과 비동기식 시리얼 포트의 차이점을 알려면 FreeBSD 시리얼 하드웨어 튜토리얼을([http://www.freebsd.org/doc/en\\_US.ISO8859-1/articles/serial-uart/index.html](http://www.freebsd.org/doc/en_US.ISO8859-1/articles/serial-uart/index.html)) 읽어 본다.

128 kbs 회선이 있더라도 TA 가없는 표준 PC 시리얼 포트(비동기)는 115.2kbs 로 제한된다. ISDN 의 128kbs 를 완벽하게 사용하려면 TA 를 동기식 시리얼 카드에 연결해야 된다.

내장형 TA 를 구입하여 동기화/비동기화 문제를 피하려고 하지 않는다. 내장형 TA 에는 단순히 표준 PC 시리얼 포트 칩이 내장되어 있다. 내장형 TA 의 장점은 다른 시리얼 케이블을 구입하여 사용하지 않아도 되고 빈 콘센트를 찾지 않아도 된다는 것뿐이다.

TA 와 동기식 카드는 단순한 386 FreeBSD 머신에서도 최소한 stand-alone 라우터만큼 빠르고 더 유연한 설정이 가능할 것이다.

동기식카드/TA와 stand-alone 라우터를 선택하는 것은 양심적인 문제에 달려있다. 메일링 리스트에 이 문제에 대한 설명이 있고 완벽한 설명은 검색에서(<http://www.freebsd.org/search/index.html>) 찾아본다.

### 24.7.3 Stand-alone ISDN 브리지/라우터

ISDN 브리거나 라우터는 FreeBSD 또는 다른 운영체제에 특별한 기능이 아니다. 라우터와 브리지 기술에 대한 완벽한 설명은 네트워크 레퍼런스 책을 참고한다.

이 페이지의 문맥에서 라우터와 브리지라는 용어는 상호 교환하여 사용할 수 있다.

---

ISDN 라우터/브리지의 가격이 계속 떨어지기 때문에 쉽게 선택할 수 있다. ISDN 라우터는 로컬 이더넷 네트워크에 직접 연결하는 작은 장치이고 다른 브리지/라우터로 연결되는 것을 제어한다. PPP 와 다른 일반적인 프로토콜을 사용하여 통신할 수 있는 소프트웨어가 내장되어 있다.

라우터는 완벽한 동기식 ISDN 회선을 사용하기 때문에 표준 TA 보다 처리량이 훨씬 빠르다.

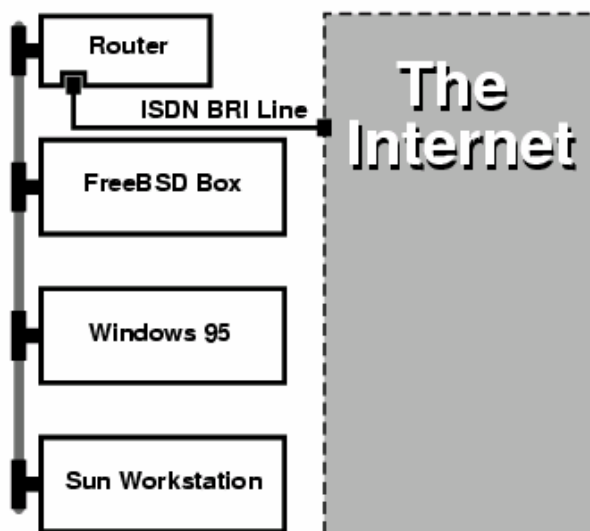
ISDN 라우터와 브리지의 주요 문제는 생산자간의 서비스 상호이용 문제가 아직 존재한다. 인터넷 공급자에게 연결할 계획이라면 이들과 상의해 보아야 될 것이다.

양쪽을 연결할 네트워크 카드만 구입하면 확실하게 동작하기 때문에 홈 네트워크를 사무실 네트워크에 연결하는 것처럼 두 개의 네트워크 구간을 연결할 계획이라면, 이 방법이 가장 간단한 관리 솔루션이다.

예를 들어 집에 있는 컴퓨터나 지정 사무실의 네트워크를 본점 사무실 네트워크에 연결하려면 다음 설정을 따르면 된다.

#### 예제 24-1. 지정 사무실 또는 홈 네트워크

네트워크는 10 base 2 개의 이더넷으로 버스 기반 토폴로지를 사용한다. 필요하다면 라우터를 AUI/10BT 트랜시버 네트워크 케이블에 연결한다.

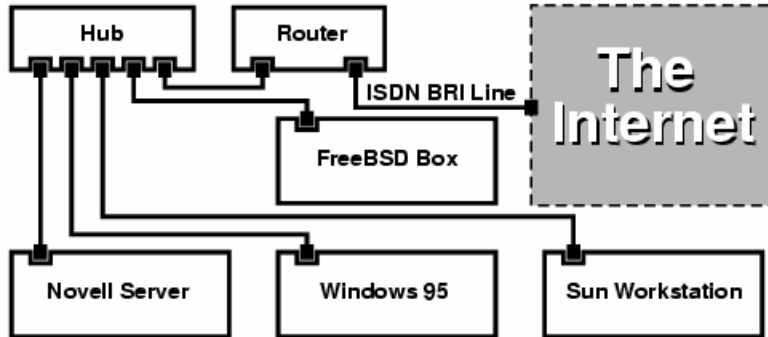


홈/지정 사무실에 한대의 컴퓨터만 있다면 크로스 케이블을 사용하여 stand-alone 라우터에 연결할 수 있다.

---

## 예제 24-2. 본점 사무실 또는 다른 네트워크

네트워크는 10 base T 이더넷으로 스타 토폴로지를 사용한다.



대부분의 라우터/브리지의 가장 큰 장점은 두 개의 독립된 PPP 회선으로 분리된 2 개의 사이트에 동시에 연결할 수 있다는 것이다. 이 설정은 두 개의 시리얼 포트를 가진 특정 모델을 제외하고 대부분의 TA가 지원하지 않는다. 채널 본딩, MMP 등과 혼돈하지 않는다.

예를 들어 사무실에 ISDN 전용회선이 있고 다른 ISDN 회선이 필요하지 않다면 이 설정은 매우 유용한 기능이다. 사무실에 있는 라우터를 인터넷과 연결된 B 채널(64Kbps) 전용선에 연결하고 분리된 데이터 연결에 다른 B 채널을 사용할 수 있다. 두 번째 B 채널은 다이얼-인, 다이얼-아웃 또는 더 많은 대역폭을 위해 첫 번째 B 채널과 동적으로 본딩(MPP 등)할 수 있다.

그리고 이더넷 브리지는 IP 트래픽 뿐만 아니라 더 많은 것을 전달할 수 있다. IPX/SPX와 사용하고 있는 다른 프로토콜도 보낼 수 있다.

## 24.8 네트워크 주소 변환

### 24.8.1 요약

일반적으로 natd(8)로 알려져 있는 FreeBSD의 네트워크 주소 변환 데몬은 들어오는 로(raw) IP 패킷을 허용하고 그 소스를 로컬머신으로 변경하며 이러한 패킷들을 외부 IP 패킷 스트림으로 다시 내보내는 데몬이다. 말하자면 natd(8)은 데이터를 돌려줄 때 원래 데이터의 위치를 결정할 수 있기 때문에 소스 IP 주소와 포트를 변경하여 최초의 요청자에게 재 전송해 준다.

---

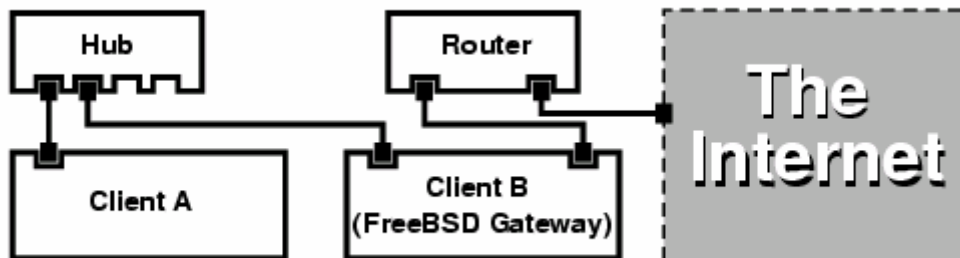
NAT 의 일반적인 용도는 인터넷을 공유하는데 사용한다.

## 24.8.2 설정

IPv4 에서 IP 주소 부족이나 케이블 또는 DSL 과 같은 고속 인터넷 유저의 증가로 사람들은 인터넷 공유 솔루션의 필요성을 많이 느끼게 됐다. 하나의 인터넷 회선과 IP 주소로 여러 대의 컴퓨터를 온라인에 연결하는 기능이 natd(8)을 선택하는 이유가 되었다.

유저는 보통 하나의 IP 주소와 케이블 또는 DSL 라인에 연결된 한대의 머신으로 LAN 에 있는 여러 대의 컴퓨터를 인터넷에 연결할 수 있기를 바란다.

이렇게 회선을 공유하려면 인터넷에 연결되어 있는 FreeBSD 머신이 게이트웨이처럼 동작해야 된다. 이 게이트웨이 머신은 두 개의 네트워크 카드가 필요하다 -- 하나는 인터넷 라우터에 다른 하나는 LAN 에 연결된다. LAN 의 모든 머신은 허브나 스위치를 통해 연결된다.



일반적으로 인터넷을 공유하는데 이런 식의 설정이 사용된다. LAN 에 있는 머신 중 한대가 인터넷에 연결되어 있고 나머지 머신은 "게이트웨이" 머신을 통해 인터넷에 접속된다.

## 24.8.3 설정

다음 옵션이 커널 설정파일에 필요하다:

```
options IPFIREWALL
options IPDIVERT
```



---

게다가 선택사항으로 다음 내용을 추가하는 것도 괜찮을 것 같다:

```
options IPFIREWALL_DEFAULT_TO_ACCEPT
options IPFIREWALL_VERBOSE
```

다음 내용을 /etc/rc.conf 에 추가한다:

```
gateway_enable="YES" ❶
firewall_enable="YES" ❷
firewall_type="OPEN" ❸
natd_enable="YES" ❹
natd_interface="fxp0" ❺
natd_flags=""
```

|                       |                                                                               |
|-----------------------|-------------------------------------------------------------------------------|
| gateway_enable="YES"  | 머신이 게이트웨이로 동작하도록 설정한다. sysctl net.inet.ip.forwarding=1 을 실행해도 같은 효과를 얻을 수 있다. |
| firewall_enable="YES" | 부팅할 때 /etc/rc.firewall 에서 방화벽 룰을 활성화한다.                                       |
| firewall_type="OPEN"  | 모든 패킷을 허용하도록 방화벽에 미리 내장된 룰을 지정한다. 추가적인 종류는 /etc/rc.firewall 을 확인한다.           |
| natd_interface="fxp0" | 어떤 인터페이스로 패킷이 포워드 되는지 보여준다(이 인터페이스는 인터넷에 연결되어 있다).                            |
| natd_flags=""         | 부팅할 때 natd(8)에 설정하는 추가적인 옵션.                                                  |

/etc/rc.conf 에 지정된 이전 옵션이 부팅할 때 natd -interface fxp0 를 실행한다. 이 명령을 수동으로도 실행할 수 있다.

**Note:** natd(8)에 너무 많은 옵션이 있기 때문에 설정파일을 사용할 수 있다. 이 경우 설정파일은 /etc/rc.conf 에 정의되어 있어야 된다:

```
natd_flags="-f /etc/natd.conf"
```

/etc/natd.conf 파일에는 라인 별로 설정옵션이 하나씩 지정되어 있다. 예를 들면 다음 섹션의 경우 다음 파일을 사용한다:

---

```
redirect_port tcp 192.168.0.2:6667 6667
redirect_port tcp 192.168.0.3:80 80
```

설정파일에 대한 자세한 정보는 `natd(8)` 매뉴얼 페이지의 `-f` 옵션을 참고한다.

LAN의 뒤 단에 있는 각 머신과 인터페이스는 RFC 1918에 정의되어있는 사설 네트워크 IP 주소와 `natd` 머신의 내부 IP 주소를 기본 게이트웨이로 할당 받는다.

예를 들어 LAN 뒷단의 클라이언트 A와 B는 192.168.0.2와 192.168.0.3의 IP 주소를 할당 받지만 `natd` 머신의 LAN 인터페이스는 192.168.0.1의 IP 주소로 설정한다. 클라이언트 A와 B의 기본 게이트웨이를 `natd` 머신의 192.168.0.1로 설정해야 된다. `natd` 머신의 외부 또는 인터넷 인터페이스에는 `natd(8)`가 작동하기 위해 특별히 수정할 것은 없다.

## 24.8.4 포트 리다이렉션

`natd`의 단점은 인터넷에서 LAN에 있는 클라이언트에게 접근할 수 없다는 것이다. LAN에 있는 클라이언트는 외부로 연결할 수 있지만 외부에서 연결할 수 없다. 이것은 LAN에 있는 클라이언트 머신으로 인터넷 서비스를 할 때 문제가 된다. 이 문제를 해결하는 단순한 방법은 `natd` 머신의 선택된 인터넷 포트를 LAN에 있는 클라이언트로 리다이렉트한다.

예를 들어 IRC 서버가 클라이언트 A에서 운용되고 웹 서버는 클라이언트 B에서 운용된다면 정확히 작동하도록 포트 6667(IRC)과 80(web)으로 들어오는 연결을 각 머신으로 리다이렉트 해야 된다.

적절한 옵션으로 `-redirect_port`를 `natd(8)`에 적용한다. 구문은 다음과 같다:

```
-redirect_port proto targetIP:targetPORT[-targetPORT]
                [aliasIP:]aliasPORT[-aliasPORT]
                [remoteIP[:remotePORT[-remotePORT]]]
```

위의 예제에서 인자는 다음과 같다:

```
-redirect_port tcp 192.168.0.2:6667 6667
-redirect_port tcp 192.168.0.3:80 80
```

이 인자는 적절한 tcp 포트를 LAN 에있는 클라이언트 머신에게 리다이렉트한다.

*-redirect\_port* 인자는 각 포트뿐만 아니라 포트의 범위를 지정하는데 사용할 수 있다. 예를 들어 *tcp 192.168.0.2:2000-3000 2000-3000* 은 포트 2000 에서 3000 으로 들어오는 모든 연결을 클라이언트 A 의 2000 에서 3000 번으로 모두 리다이렉트한다.

이들 옵션은 natd(8)을 실행할 때 직접 입력, /etc/rc.conf 의 *natd\_flags=""*에 입력하여 또는 설정파일을 통해 적용할 수 있다.

더 많은 설정옵션은 natd(8)을 참고한다.

## 24.8.5 주소 리다이렉션

주소 리다이렉션은 여러 개의 IP 주소를 사용할 수 있을 때 유용하지만 머신 한대에 이 주소들을 할당해야 된다. 이렇게 해서 natd(8)은 각 LAN 클라이언트에 각자의 외부 IP 주소를 할당할 수 있다. natd(8)은 LAN 에있는 클라이언트로부터 외부로 나가는 패킷을 적절한 외부 IP 주소로 재 작성하고, 특정 IP 주소로 입력되는 모든 트래픽을 지정된 LAN 클라이언트로 리다이렉트한다. 이러한 방식을 고정적 NAT 라고도 한다. 예를 들면 IP 주소 128.1.1.1, 128.1.1.2, 그리고 128.1.1.3 이 **natd** 게이트웨이 머신에 있다. 128.1.1.1 을 **natd** 게이트웨이 머신의 외부 IP 주소로 사용할 수 있지만 128.1.1.2 와 128.1.1.3 은 LAN 클라이언트 A 와 B 에 포워드된다.

*-redirect\_address* 구문은 다음과 같다:

```
-redirect_address localIP publicIP
```

|          |                          |
|----------|--------------------------|
| localIP  | LAN 클라이언트의 내부 IP 주소      |
| publicIP | LAN 클라이언트에 대응되는 외부 IP 주소 |

예제에서 이 인자는 다음과 같다:

```
-redirect_address 192.168.0.2 128.1.1.2
-redirect_address 192.168.0.3 128.1.1.3
```

`-redirect_port` 처럼 이들 인자도 `/etc/rc.conf` 의 `natd_flags=""` 옵션이나 설정파일을 통해 적용할 수 있다. 주소 리다이렉션에서 특정 IP 주소로 들어오는 모든 데이터는 리다이렉트되기 때문에 포트 리다이렉션은 필요 없다.

natd 머신의 외부 IP 주소를 활성화하고 외부 인터페이스로 엘리어스 되어야 한다. `rc.conf(5)`을 확인한다.

## 24.9 Parallel Line IP (PLIP)

PLIP 는 양쪽 패러럴 포트에서 TCP/IP 를 사용할 수 있게 한다. 네트워크 카드가 없는 머신이나 노트북에 설치할 때 유용하다. 이 섹션에서는 다음과 같은 사항을 다룬다:

- 패러럴 케이블(laplink) 만들기
- PLIP로 두 대의 컴퓨터 연결하기

### 24.9.1 패러럴 케이블 만들기

대부분의 컴퓨터 소매점에서 패러럴 케이블을 구입할 수 있다. 구입할 수 없거나 직접 만드는 방법을 알고 싶다면, 다음 표에서 일반 패러럴 프린터 케이블이 아닌 네트워크용 패러럴 케이블 만드는 것을 보여준다

표 24-1. 네트워크용 패러럴 케이블 배선

| A-name | A-End | B-End | Descr. | Post/Bit |
|--------|-------|-------|--------|----------|
| DATA0  | 2     | 15    | Data   | 0/0x01   |
| -ERROR | 15    | 2     |        | 1/0x08   |
| DATA1  | 3     | 13    | Data   | 0/0x02   |
| +SLCT  | 13    | 3     |        | 1/0x10   |

|               |         |         |        |                  |
|---------------|---------|---------|--------|------------------|
| DATA2<br>+PE  | 4<br>12 | 12<br>4 | Data   | 0/0x04<br>1/0x20 |
| DATA3<br>-ACK | 5<br>10 | 10<br>5 | Strobe | 0/0x08<br>1/0x40 |
| DATA4<br>BUSY | 6<br>11 | 11<br>6 | Data   | 0/0x10<br>1/0x80 |
| GND           | 18-25   | 18-25   | GND    | -                |

## 24.9.2 PLIP 설정

랩 링크(laplink) 케이블을 준비하자. 그리고 양쪽 컴퓨터의 커널에서 lpt(4) 드라이버를 지원하는지 확인한다:

```
# grep lp /var/run/dmesg.boot
lpt0: <Printer> on ppbus0
lpt0: Interrupt-driven port
```

FreeBSD 4.X 에서 패러럴 포트는 interrupt driven 포트(포트의 인터럽트를 사용하여 데이터를 전송하는)여야 되고 커널 설정파일에 다음과 같은 라인이 있는지 확인한다:

```
device ppc0 at isa? irq 7
```

FreeBSD 5.X 에서는 /boot/device.hints 파일에 다음 라인이 필요하다:

```
hint.ppc.0.at="isa"
hint.ppc.0.irq="7"
```

커널 설정파일에 *device plip* 라인이 있는지 또는 *plip.ko* 커널 모듈이 로드 되었는지 확인한다. 이 두 가지 경우 패러럴 네트워크 인터페이스는 *ifconfig(8)* 명령을 직접 사용할 때 나타나야 한다. FreeBSD 4.X 에서는 다음과 같이 나타나고:

```
# ifconfig lp0
lp0: flags=8810<POINTOPOINT,SIMPLEX,MULTICAST> mtu 1500
```

---

그리고 FreeBSD 5.X 는 아래와 같이 나타난다:

```
# ifconfig plip0
```

```
plip0: flags=8810<POINTOPOINT,SIMPLEX,MULTICAST> mtu 1500
```

**Note:** 패러럴 인터페이스에 사용하는 장치 이름은 FreeBSD 4.X (lpx)와 FreeBSD 5.X (plipx)에서 다르다.

랩 링크 케이블을 두 컴퓨터의 패러럴 인터페이스에 꽂는다.

root 유저에서 양쪽 네트워크 인터페이스의 매개변수를 설정한다. 예를 들면 FreeBSD 4.X 로 운용되는 호스트 host1 과 FreeBSD 5.X 로 운용되는 호스트 host2 를 연결하려면 다음과 같이 설정한다:

|                                          |
|------------------------------------------|
| host1 <-----> host2                      |
| IP Address     10.0.0.1         10.0.0.2 |

host1 에서는 다음과 같이 인터페이스를 설정한다:

```
# ifconfig lp0 10.0.0.1 10.0.0.2
```

host2 에서는 다음과 같이 인터페이스를 설정한다:

```
# ifconfig lp0 10.0.0.2 10.0.0.1
```

이제 연결이 정상적으로 동작한다. 더 자세한 사항은 lp(4)와 lpt(4) 매뉴얼 페이지를 참고한다.

그리고 양쪽 호스트를 /etc/hosts 에 추가해야 된다:

|           |                               |
|-----------|-------------------------------|
| 127.0.0.1 | localhost.my.domain localhost |
| 10.0.0.1  | host1.my.domain host1         |
| 10.0.0.2  | host2.my.domain               |

연결이 정상인지 확인하려면 각 호스트에서 상대 컴퓨터로 ping 테스트를 한다. 예를 들어

---

host1 에서 다음과 같이 실행한다:

```
# ifconfig lp0
lp0: flags=8851<UP,POINTOPOINT,RUNNING,SIMPLEX,MULTICAST> mtu 1500
    inet 10.0.0.1 --> 10.0.0.2 netmask 0xff000000
# netstat -r
Routing tables

Internet:
Destination          Gateway              Flags      Refs      Use      Netif Expire
host2                 host1                UH         4        127592    lp0
# ping -c 4 host2
PING host2 (10.0.0.2): 56 data bytes
64 bytes from 10.0.0.2: icmp_seq=0 ttl=255 time=2.774 ms
64 bytes from 10.0.0.2: icmp_seq=1 ttl=255 time=2.530 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=255 time=2.556 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=255 time=2.714 ms

--- host2 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 2.530/2.643/2.774/0.103 ms
```

## 24.10 IPv6

IPv6(또한 IPng "IP next generation"으로 유명한)는 유명한 IP 프로토콜(IPv4 로 알고 있는)의 다음 버전이다. 현재 다른 \*BSD 시스템처럼 FreeBSD 도 KAME IPv6 레퍼런스 실행을 포함하고 있다. 그래서 FreeBSD 시스템은 IPv6 에 필요한 모든 것을 가지고 있다. 이번 섹션은 IPv6 설정과 운용에 대해 초점을 맞췄다.

1990 년대 초기에 사람들은 IPv4 의 주소가 급속히 감소하는 것을 감지하였다. 인터넷의 팽창에 따라 두 가지 주된 문제가 나타나고 있다:

- 주소 부족. 요즘 이 문제는 사설 IP 주소와(10.0.0.0/8, 192.168.0.0/24등) 네트워크 주소 변환(NAT)으로 더 이상 염려되지 않는다.
- 라우팅 테이블 엔트리가 점점 커진다. 이것은 요즘에도 문제가 되고 있다.

---

IPv6 에서 이들 문제와 다른 장점:

- 128 bit 주소 공간. 다시 말해서 이론적으로 340,282,366,920,938,463,374,607,431,768,211,456의 주소를 사용할 수 있다. 이 의미는 우리 행성에서 평방 미터당 대략  $6.67 * 10^{27}$ 의 IPv6 주소가 있다는 것이다.
- 라우터는 라우팅 테이블에 네트워크 어그리게이션(aggregation) 주소만 저장하면 되기 때문에 라우팅 테이블의 평균 공간을 8192 엔트리로 줄일 수 있다.

다음과 같이 IPv6 의 다양하고 유용한 특성이 있다:

- 주소 자동 설정 (RFC2462).
- 애니캐스트(Anycast) 주소
- 필수적인 멀티캐스트(Multicast) 주소
- IPsec (IP 보안)
- 간략한 헤더 구조
- 모바일 IP
- IPv4를 IPv6로 변환하는 메커니즘

더 많은 정보는 다음 사이트를 참고한다:

- playground.sun.com에서(<http://www.sun.com/>) IPv6 요약
- KAME.net (<http://www.kame.net/>)
- 6bone.net (<http://www.6bone.net/>)

## 24.10.1 IPv6 주소에 대한 백그라운드

유니캐스트(Unicast), 애니캐스트(Anycast)와 멀티캐스트(Multicast)의 IPv6 주소가 있다.

유니캐스트 주소는 일반적인 주소다. 패킷을 유니캐스트 주소로 보내면 주소에 속한 인터페이스에 정확하게 도달한다.

애니캐스트 주소는 유니캐스트 주소에서 구문적으로는 구분할 수 없지만 이들 인터페이스의 그룹 주소로는 구분할 수 있다. 애니캐스트 주소로 향하는 패킷은 가장 가까운(라우터 측정) 인터페이스에 도달한다. 애니캐스트 주소는 라우터에서만 사용될 것이다.



멀티캐스트 주소는 인터페이스 그룹을 식별한다. 멀티캐스트 주소로 향하는 패킷은 멀티캐스트 그룹에 속하는 모든 인터페이스에 도달한다.

**Note:** IPv4 브로드캐스트 주소는(보통 xxx.xxx.xxx.255) IPv6 에서 멀티캐스트 주소로 표현된다.

표 24-2. 예약되어 있는 IPv6 주소

| IPv6-주소          | 미리 예약되어 있는 길이(Bits) | 설 명                        | 비 고                                                    |
|------------------|---------------------|----------------------------|--------------------------------------------------------|
| ::               | 128 비트              | 지정되어 없음 (unspecified)      | 예: IPv4 에서 0.0.0.0                                     |
| ::1              | 128 비트              | 루프백 주소                     | 예: IPv4 에서 127.0.0.1                                   |
| ::00:xx:xx:xx:xx | 96 비트               | IPv4 에 내장되어있음              | 낮은 32 비트 주소들은 IPv4 주소다. 그리고 IPv6 주소와 호환되는 IPv4 라고 부른다. |
| ::ff:xx:xx:xx:xx | 96 비트               | IPv4 를 IPv6 주소로 매핑         | 낮은 32 비트 주소들은 IPv4 주소다. IPv6 를 지원하지 않는 호스트를 위한 것이다.    |
| fe80:: - feb::   | 10 비트               | link-local                 | cf. loopback address in IPv4                           |
| fec0:: - fef::   | 10 비트               | site-local                 |                                                        |
| ff::             | 8 비트                | 멀티캐스트                      |                                                        |
| 001 (base 2)     | 3 비트                | 글로벌 유니캐스트 (global unicast) | 모든 글로벌 유니캐스트 주소는 이 풀(pool)에서 할당된다. 처음 3 비트는 "001"이다.   |

## 24.10.2 IPv6 주소 읽기

규정된 형식은 x:x:x:x:x:x:x:x 으로 표현되고 각 "x"은 16 비트 16 진수 값이다. 예를 들면 FEBC:A574:382B:23C1:AA49:4592:4EFE:9982 과 같이 읽는다.

---

가끔 주소는 모두 0 인 긴 서브스트링을 가지고 있으므로 각 서브스트링은 ":"으로 줄일 수 있다. 예를 들어 fe80::1 은 규정된 형식 fe80:0000:0000:0000:0000:0000:0000:0001 과 동일하다.

세 번째 형식은 IPv4 스타일로 유명한(10 진수) 점 "."으로 32Bit 씩 분리하여 작성한다. 예를 들어 2002::10.0.0.1 는 2002::a00:1 과 동일하게 규정된 표현식 2002:0000:0000:0000:0000:0000:0a00:0001 과 일치한다.

이제 다음 내용을 읽을 수 있다:

```
# ifconfig
rlo: flags=8943<UP,BROADCAST,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu
1500
    inet 10.0.0.10 netmask 0xfffff00 broadcast 10.0.0.255
    inet6 fe80::200:21ff:fe03:8e1%rlo prefixlen 64 scopeid 0x1
    ether 00:00:21:03:08:e1
    media: Ethernet autoselect (100baseTX )
    status: active
```

fe80::200:21ff:fe03:8e1%rlo 은 로컬 주소로 링크되도록 자동으로 설정된다. 자동 설정의 일부분으로 MAC 주소에서 생성된다.

IPv6 주소의 구조에 대한 더 자세한 사항은 RFC2373 을 참고한다

## 24.10.3 연결

현재 다른 IPv6 호스트와 네트워크에 연결하는 4 가지 방법이 있다:

- 실험적으로 6bone에 등록한다.
- ISP로부터 IPv6 네트워크를 제공받고 설명은 인터넷 공급자에게 문의한다.
- IPv6을 IPv4로 변경한다.
- 다이얼-업 연결을 사용한다면 [net/freenet6](#) 포트를 사용한다.

현재 가장 일반적인 방법이므로 여기서는 6bone 에 연결하는 방법을 설명한다

---

첫째로 6bone 사이트(<http://www.6bone.net/>)에서 가장 가까운 6bone 회선을 찾는다. 약간의 행운이 있다면 책임자에게 글을 보내서 설정하는 방법을 들을 수 있다(<http://www.ipv6.or.kr/> 사이트를 참고한다). 보통 GRE (gif)터널 설정이 필요하다.

일반적인 gif(4) 터널을 설정하는 예제가 있다:

```
# ifconfig gif0 create
# ifconfig gif0
gif0: flags=8010<POINTOPOINT,MULTICAST> mtu 1280
# ifconfig gif0 tunnel MY_IPv4_ADDR HIS_IPv4_ADDR
# ifconfig gif0 inet6 alias MY_ASSIGNED_IPv6_TUNNEL_ENDPOINT_ADDR
```

대문자 단어는 6bone 노드에서 받은 정보로 교체한다.

위 설정으로 구축한 터널이 작동하는지 ping6(8) 'ing ff02::1%gif0 로 체크하면 두 개의 핑이 되돌아오는 것을 볼 수 있다.

**Note:** ff02:1%gif0 주소에 관심이 있다면 이것은 멀티캐스트 주소다. %gif0 은 네트워크 인터페이스 gif0 에 멀티캐스트 주소가 사용됨을 나타낸다. 터널의 다른 끝에 멀티캐스트 주소로 ping 을 보내기 때문에 응답이 오는 것이다.

라우터를 6bone 업 링크로 설정하는 것은 상당히 직관적이다:

```
# route add -inet6 default -interface gif0
# ping6 -n MY_UPLINK
# traceroute6 www.jp.FreeBSD.org
(3ffe:505:2008:1:2a0:24ff:fe57:e561) from 3ffe:8060:100::40:2, 30 hops max, 12 byte
packets
 1  atnet-meta6  14.147 ms  15.499 ms  24.319 ms
 2  6bone-gw2-ATNET-NT.ipv6.tilab.com  103.408 ms  95.072 ms *
 3  3ffe:1831:0:ffff::4  138.645 ms  134.437 ms  144.257 ms
 4  3ffe:1810:0:6:290:27ff:fe79:7677  282.975 ms  278.666 ms  292.811 ms
 5  3ffe:1800:0:ff00::4  400.131 ms  396.324 ms  394.769 ms
 6  3ffe:1800:0:3:290:27ff:fe14:cdee  394.712 ms  397.19 ms  394.102 ms
```

---

이 결과는 머신마다 다르다. 모질라 같은 브라우저에서 IPv6 를 활성화했다면 IPv6 사이트 [www.kame.net](http://www.kame.net) 에 접속해서 출처는 거북이를 볼 수 있다.

## 24.10.4 IPv6 에서 DNS

IPv6 를 위한 두 가지 종류의 DNS 레코드가 사용되었다. IETF 에서 A6 레코드를 사용하지 않는다고 선언해서 이제 AAAA 레코드가 표준이다.

AAAA 레코드를 사용하는 것도 역시 상당히 직관적이다. 호스트 네임에 새로운 IPv6 주소를 할당하려면 다음 내용을 주 DNS 존 파일에 추가한다.

```
MYHOSTNAME          AAAA    MYIPv6ADDR
```

이 경우 여러분의 DNS 존을 DNS 공급자에게 제공할 필요가 없다. 현재 **bind** 버전과(버전 8.3 과 9) [dns/djbdns](#)(IPv6 패치)에서 AAAA 레코드를 지원한다.

## 24.10.5 /etc/rc.ocnf 변경

### 24.10.5.1 IPv6 클라이언트 설정

여기서는 LAN 에서 클라이언트로 동작하는 머신을 설정하는 방법을 지원한다. 부팅할 때 인터페이스에 `rtsol(8)`이 자동으로 설정되도록 하려면 다음 라인을 추가한다:

```
ipv6_enable="YES"
```

`fxp0` 인터페이스에 `2001:471:1f11:251:290:27ff:fee0:2093` 과 같은 IP 를 고정적으로 할당하려면 다음을 추가한다:

```
ipv6_ifconfig_fxp0="2001:471:1f11:251:290:27ff:fee0:2093"
```

`2001:471:1f11:251::1` 의 기본 라우터를 추가하려면 `/etc/rc.conf` 에 다음 내용을 추가한다:

```
ipv6_defaultrouter="2001:471:1f11:251::1"
```

---

## 24.10.5.2 IPv6 라우터 / 게이트웨이 설정

여기서는 6bone(<http://www.6bond.net/>)같은 터널 제공자가 보내준 자료를 설정하고 재부팅하더라도 설정이 지속되도록 도와준다. 시작할 때 터널을 복원하기 위해 다음과 같은 설정을 /etc/rc.conf에서 사용한다:

설정하려는 일반적인 터널링 인터페이스를 나열한다. 예를 들어 gif0 는 다음과 같이 추가한다:

```
gif_interfaces="gif0"
```

로컬 쪽 끝단 *MY\_IPv4\_ADDR*을 원격지의 끝단 *REMOTE\_IPv4\_ADDR*로 인터페이스를 설정하려면 다음 내용을 추가한다:

```
gif_config_gif0="MY_IPv4_ADDR REMOTE_IPv4_ADDR"
```

할당한 IPv6 주소를 여러분의 IPv6 터널 끝단에 적용하려면 다음 내용을 추가한다:

```
ipv6_ifconfig_gif0="MY_ASSIGNED_IPv6_TUNNEL_ENDPOINT_ADDR"
```

마지막으로 IPv6 기본 라우트를 설정한다. 다음은 반대 쪽 IPv6 터널이다:

```
ipv6_defaultrouter="MY_IPv6_REMOTE_TUNNEL_ENDPOINT_ADDR"
```

## 24.10.6 라우터 알림(Router Advertisement)와 호스트 자동 설정

이 섹션은 IPv6 기본 라우트를 알리도록 rtadvd(8)를 설정한다.

```
라우터 알림(Router Advertisement): IPv6 라우터는 라우터 알림 메시지를 정기적으로 또는 라우터 요청(Router Solicitation) 메시지에 대한 응답으로 보내며, 여기에는 hop limit,
```

---

링크 접두사, 링크 MTU, 라우터 작동시간 등 호스트에서 필요로 하는 정보가 포함된다.

rtadvd(8)을 활성화하려면 /etc/rc.conf 에 다음 라인을 추가한다:

```
rtadvd_enable="YES"
```

IPv6 라우터 요청(Router solicitation)을 하는 인터페이스를 지정하는 것은 중요하다.

**라우터 요청(Router Solicitation):** 라우터 요청 메시지는 링크에 있는 IPv6 라우터를 검색하기 위해 IPv6 호스트에서 전송하며, 호스트는 정기적인 라우터 메시지를 기다리지 않고 멀티캐스트 라우터 요청 메시지로 IPv6 라우터에게 즉시 응답하라는 메시지(Router Advertisement)를 보낸다.

예를 들어 rtadvd(8)에게 fxp0 를 사용하도록 하려면 다음 라인을 추가한다:

```
rtadvd_interfaces="fxp0"
```

이제 /etc/rtadvd.conf 설정파일을 생성해야 된다. 여기 예제가 있다:

```
fxp0:W  
:addrs#1:addr="2001:471:1f11:246::":prefixlen#64:tc=ether:
```

fxp0 대신 사용하려는 인터페이스를 지정한다.

2001:471:1f11:246::는 여러분에게 할당된 값으로 변경한다.

/64 서브넷 전체를 사용한다면 변경할 것이 없지만 그렇지 않으면 *prefixlen#*를 정확한 값으로 변경한다.

## 24.11 FreeBSD 5.X 에서 ATM

### 24.11.1 ATM(PVCs)을 통한 클래식컬 IP 설정

---

Classical IP over ATM(CLIP)은 IP 로 ATM 을 사용하는 가장 단순한 방법이다. switched connections(SVCs)과 permanent connections(PVCs)을 사용할 수 있다. 이 섹션은 PVCs 기반 네트워크를 어떻게 설정하는지 설명한다.

### 24.11.1.1 완전한 네트워크 설정

PVCs 로 CLIP 를 설정하는 첫 번째 방법은 전용 PVC 로 각 시스템을 다른 시스템과 연결한다. 이 방법은 설정하기 쉽지만 시스템이 많아지면 실용성이 떨어진다. 예를 들어 네트워크에 4 대의 시스템이 있고 각 시스템은 ATM 아답터 카드로 ATM 네트워크와 연결되어 있다고 가정한다. 첫 단계는 IP 주소 할당 계획과 시스템간의 ATM 연결이다. 다음 호스트와 IP 를 사용한다고 가정한다:

| 호스트   | IP 주소         |
|-------|---------------|
| hostA | 192.168.173.1 |
| hostB | 192.168.173.2 |
| hostC | 192.168.173.3 |
| hostD | 192.168.173.4 |

완벽하게 연결되어 있는 네트워크를 만들기 위해 각 시스템 쌍에 ATM 을 연결해야 된다:

| Machines      | VPI.VCI couple |
|---------------|----------------|
| hostA - hostB | 0.100          |
| hostA - hostC | 0.101          |
| hostA - hostD | 0.102          |
| hostB - hostC | 0.103          |
| hostB - hostD | 0.104          |
| hostC - hostD | 0.105          |

각 연결의 끝에서 VPI 와 VCI 값은 다를 수 있지만 여기서는 간단히 같다고 여긴다. 그리고 각 호스트의 ATM 인터페이스를 설정해야 된다:

---

```
hostA# ifconfig hatm0 192.168.173.1 up
hostB# ifconfig hatm0 192.168.173.2 up
hostC# ifconfig hatm0 192.168.173.3 up
hostD# ifconfig hatm0 192.168.173.4 up
```

모든 호스트의 ATM 인터페이스가 hatm0 이라고 가정하고 PVCs 를 hostA 에 설정한다(이미 ATM 스위치가 설정되어 있다고 가정하고 스위치 설정은 매뉴얼 페이지를 참고한다).

```
hostA# atmconfig natm add 192.168.173.2 hatm0 0 100 llc/snap ubr
hostA# atmconfig natm add 192.168.173.3 hatm0 0 101 llc/snap ubr
hostA# atmconfig natm add 192.168.173.4 hatm0 0 102 llc/snap ubr

hostB# atmconfig natm add 192.168.173.1 hatm0 0 100 llc/snap ubr
hostB# atmconfig natm add 192.168.173.3 hatm0 0 103 llc/snap ubr
hostB# atmconfig natm add 192.168.173.4 hatm0 0 104 llc/snap ubr

hostC# atmconfig natm add 192.168.173.1 hatm0 0 101 llc/snap ubr
hostC# atmconfig natm add 192.168.173.2 hatm0 0 103 llc/snap ubr
hostC# atmconfig natm add 192.168.173.4 hatm0 0 105 llc/snap ubr

hostD# atmconfig natm add 192.168.173.1 hatm0 0 102 llc/snap ubr
hostD# atmconfig natm add 192.168.173.2 hatm0 0 104 llc/snap ubr
hostD# atmconfig natm add 192.168.173.3 hatm0 0 105 llc/snap ubr
```

주어진 ATM 아답터로 이들을 지원하는데 UBR 이 아닌 다른 트래픽 contracts 를 사용할 수 있다. 이 같은 경우 트래픽 contract 의 이름을 트래픽 매개변수 뒤에 지정한다. 다음 명령으로 atmconfig(8) 틀 도움말을 보거나 atmconfig(8) 매뉴얼 페이지를 참고한다.

```
# atmconfig help natm add
```

같은 설정을 /etc/rc.conf 로도 할 수 있다. hostA 는 다음과 비슷하다:



---

```
network_interfaces="lo0 hatm0"
ifconfig_hatm0="inet 192.168.173.1 up"
natm_static_routes="hostB hostC hostD"
route_hostB="192.168.173.2 hatm0 0 100 llc/snap ubr"
route_hostC="192.168.173.3 hatm0 0 101 llc/snap ubr"
route_hostD="192.168.173.4 hatm0 0 102 llc/snap ubr"
```

모든 CLIP 라우트의 현재 상태는 다음 명령으로 볼 수 있다:

```
hostA# atmconfig natm show
```

---

## III. 부록

### 부록 A. FreeBSD 받기

#### A.1 CDROM 과 DVD 판매

##### A.1.1 박스 제품

여러 판매자의 FreeBSD 박스 제품(FreeBSD CD, 추가적인 소프트웨어, 매뉴얼) 구입할 수 있다:

- CompUSA  
WWW:<http://www.compusa.com/>
- Frys Electronics  
WWW:<http://www.frys.com/>

##### A.1.2 CD 와 DVD

FreeBSD CD 와 DVD 를 여러 온라인 판매 사이트에서 구입할 수 있다.

- BSD Mall by Daemon News  
PO Box 161  
Nauvoo, IL 62354  
USA  
Phone: +1 866 273-6255  
Fax: +1 217 453-9956  
Email: <sales@bsdmail.com>  
WWW: <http://www.bsdmail.com/freebsd1.html>
- BSD-Systems  
Email: <info@bsd-systems.co.uk>

---

WWW: <http://www.bsd-systems.co.uk>

- fastdiscs.com  
6 Eltham Close  
Leeds, LS6 2TY  
United Kingdom  
Phone: +44 870 1995 171  
Email: <[sales@fastdiscs.com](mailto:sales@fastdiscs.com)>  
WWW: <http://fastdiscs.com/freebsd/>
- FreeBSD Mall, Inc.  
3623 Sanford Street  
Concord, CA 94520-1405  
USA  
Phone: +1 925 674-0783  
Fax: +1 925 674-0821  
Email: <[info@freebsdmail.com](mailto:info@freebsdmail.com)>  
WWW: <http://www.freebsdmail.com/>
- FreeBSD Services Ltd  
11 Lapwing Close  
Bicester  
OX26 6XR  
United Kingdom  
WWW: <http://www.freebsd-services.com/>
- Hinner EDV  
St. Augustinus-Str. 10  
D-81825 Munchen  
Germany  
Phone: (089) 428 419  
WWW: <http://www.hinner.de/linux/freebsd.html>

- 
- Ikarios  
22-24 rue Voltaire  
92000 Nanterre  
France  
WWW: <http://ikarios.com/form/#freebsd>
  
  - Ingram Micro  
1600 E. St. Andrew Place  
Santa Ana, CA 92705-4926  
USA  
Phone: 1 (800) 456-8000  
WWW: <http://www.ingrammicro.com/>
  
  - JMC Software  
Ireland  
Phone: 353 1 6291282  
WWW: <http://www.thelinuxmall.com>
  
  - The Linux Emporium  
Hilliard House, Lester Way  
Wallingford  
OX10 9TA  
United Kingdom  
Phone: +44 1491 837010  
Fax: +44 1491 837016  
WWW: <http://www.linuxemporium.co.uk/products/freebsd/>
  
  - Linux System Labs Australia  
21 Ray Drive  
Balwyn North

---

VIC - 3104  
Australia  
Phone: +61 3 9857 5918  
Fax: +61 3 9857 8974  
WWW: <http://www.isl.com.au>

- LinuxCenter.Ru  
Galernaya Street, 55  
Saint-Petersburg  
190000  
Russia  
Phone: +7-812-3125208  
Email: <[info@linuxcenter.ru](mailto:info@linuxcenter.ru)>  
WWW: <http://linuxcenter.ru/freebsd>
  
- UNIXDVD.COM LTD  
57 Primrose Avenue  
Sheffield  
S5 6FS  
United Kingdom  
WWW: <http://www.unixdvd.com/>

## A.1.3 배포사

FreeBSD CDROM 제품을 직접 판매한다면 연락을 주기 바란다:

- Cylogistics  
809B Cuesta Dr., #2149  
Mountain View, CA 94040  
USA  
Phone: +1 650 694-4949  
Fax: +1 650 694-4953  
Email: <[sales@cylogistics.com](mailto:sales@cylogistics.com)>

---

WWW: <http://www.cylogistics.com/>

- FreeBSD Services Ltd  
11 Lapwing Close  
Bicester  
OX26 6XR  
United Kingdom  
WWW: <http://www.freebsd-services.com/>
- Kudzu, LLC  
7375 Washington Ave. S.  
Edina, MN 55439  
USA  
Phone: +1 952 947-0822  
Fax: +1 952 947-0876  
Email: <[sales@kudzuenterprises.com](mailto:sales@kudzuenterprises.com)>
- LinuxCenter.Ru  
Galernaya Street, 55  
Saint-Petersburg  
190000  
Russia  
Phone: +7-812-3125208  
Email: <[info@linuxcenter.ru](mailto:info@linuxcenter.ru)>  
WWW: <http://linuxcenter.ru/freebsd>
- Navarre Corp  
7400 49th Ave South  
New Hope, MN 55428  
USA  
Phone: +1 763 535-8333  
Fax: +1 763 535-0341

---

WWW: <http://www.navarre.com/>

## A.2 FTP 사이트

FreeBSD 공식 소스는 전 세계에 설치되어 있는 익명 FTP 미러 사이트를 통해 받을 수 있다. <ftp://ftp.FreeBSD.org/pub/FreeBSD/> 사이트가 연결이 잘되고 수많은 사람들이 접속할 수 있지만 더 빠르고 가까운 미러 사이트를 찾을 수 있을 것이다(특히 여러분이 어떤 종류의 미러 사이트를 직접 구축한다면).

정적인 호스트 리스트보다 DNS로부터 정보를 가져오기 때문에 FreeBSD 미러 사이트 데이터베이스(<http://mirrorlist.freebsd.org/FBSDsites.php>)가 핸드북에 나열된 것보다 더 정확한 미러 사이트 리스트를 가지고 있다.

추가적으로 다음 미러 사이트에서 익명 FTP를 통해 FreeBSD를 사용할 수 있다. 익명 FTP를 통해 FreeBSD를 받으려면 가까운 미러 사이트를 찾아본다. 미러 사이트에 “Primary Mirror Sites”로 리스트된 곳이 FreeBSD 전체 아카이브(archive)를 가지고 있지만(각 아키텍처에 현재 사용할 수 있는 모든) 여러분에게 가까운 사이트에서 더 빠르게 다운로드 할 수 있다. 각 나라에 있는 사이트는 가장 유명한 아키텍처의 최근 버전은 가지고 있지만 FreeBSD 아카이브를 모두 가지고 있지 않을 것이다. 모든 사이트를 익명 FTP로 사용할 수 있지만 어떤 사이트는 다른 방법을 제공한다. 각 사이트를 사용할 수 있는 방법은 각 나라의 사이트에서 제공한다.

### 중앙 서버

- <ftp://ftp.FreeBSD.org/pub/FreeBSD/> (ftp)

### 주 서버

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다

<[mirror-admin@FreeBSD.org](mailto:mirror-admin@FreeBSD.org)>.

<ftp://ftp1.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp2.FreeBSD.org/pub/FreeBSD/> (ftp)

---

<ftp://ftp3.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp4.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp5.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp6.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp7.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp8.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp9.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp10.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp11.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp12.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp13.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp14.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 아르헨티나

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@ar.FreeBSD.org](mailto:hostmaster@ar.FreeBSD.org)>.

<ftp://ftp.ar.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 호주

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@au.FreeBSD.org](mailto:hostmaster@au.FreeBSD.org)>.

<ftp://ftp.au.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp2.au.FreeBSD.org/pub/FreeBSD/> (ftp)



---

<ftp://ftp3.au.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 오스트리아

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@at.FreeBSD.org](mailto:hostmaster@at.FreeBSD.org)>.

<ftp://ftp.at.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://))

<ftp://ftp2.at.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://)/rsync)

#### 브라질

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@br.FreeBSD.org](mailto:hostmaster@br.FreeBSD.org)>.

<ftp://ftp.br.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://))

<ftp://ftp2.br.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp3.br.FreeBSD.org/pub/FreeBSD/> (ftp/rsync)

<ftp://ftp4.br.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp5.br.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp6.br.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp7.br.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 캐나다

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@ca.FreeBSD.org](mailto:hostmaster@ca.FreeBSD.org)>.

<ftp://ftp.ca.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp2.ca.FreeBSD.org/> (ftp)

#### 중국

---

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[phj@cn.FreeBSD.org](mailto:phj@cn.FreeBSD.org)>.

- <ftp://ftp.cn.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp2.cn.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 크로아티아

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@hr.FreeBSD.org](mailto:hostmaster@hr.FreeBSD.org)>.

<ftp://ftp.hr.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 체코 공화국

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@cz.FreeBSD.org](mailto:hostmaster@cz.FreeBSD.org)>.

<ftp://ftp.cz.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://)/rsync)

#### 덴마크

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@dk.FreeBSD.org](mailto:hostmaster@dk.FreeBSD.org)>.

<ftp://ftp.dk.FreeBSD.org/pub/FreeBSD/> (ftp/ftpv6/[http](http://))

<ftp://ftp2.dk.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 에스토니아

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@ee.FreeBSD.org](mailto:hostmaster@ee.FreeBSD.org)>.

<ftp://ftp.ee.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 핀란드

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@fi.FreeBSD.org](mailto:hostmaster@fi.FreeBSD.org)>.

---

<ftp://ftp.fi.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp3.fi.FreeBSD.org/pub/FreeBSD/> (ftp)

## 프랑스

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@fr.FreeBSD.org](mailto:hostmaster@fr.FreeBSD.org)>.

<ftp://ftp.fr.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp2.fr.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp3.fr.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp5.fr.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp6.fr.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp8.fr.FreeBSD.org/pub/FreeBSD/> (ftp)

## 독일

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다 <[de-bsd-hubs@de.FreeBSD.org](mailto:de-bsd-hubs@de.FreeBSD.org)>.

<ftp://ftp.de.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp2.de.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://)/rsync)

<ftp://ftp3.de.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp4.de.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://)/rsync)

<ftp://ftp5.de.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp6.de.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp7.de.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://))

---

<ftp://ftp8.de.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 그리스

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@gr.FreeBSD.org](mailto:hostmaster@gr.FreeBSD.org)>.

<ftp://ftp.gr.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp2.gr.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 홍콩

<ftp://ftp.hk.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 헝가리

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[mohacsi@ik.bme.hu](mailto:mohacsi@ik.bme.hu)>.

<ftp://ftp.hu.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://)/rsync)

<ftp://ftp2.hu.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 아이슬란드

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@is.FreeBSD.org](mailto:hostmaster@is.FreeBSD.org)>.

<ftp://ftp.is.FreeBSD.org/pub/FreeBSD/> (ftp/rsync)

#### 아일랜드

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@ie.FreeBSD.org](mailto:hostmaster@ie.FreeBSD.org)>.

<ftp://ftp.ie.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp2.ie.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://)/rsync)

---

<ftp://ftp3.ie.FreeBSD.org/pub/FreeBSD/> (ftp/[http/rsync](http://rsync))

#### 이탈리아

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@it.FreeBSD.org](mailto:hostmaster@it.FreeBSD.org)>.

<ftp://ftp.it.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 일본

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@jp.FreeBSD.org](mailto:hostmaster@jp.FreeBSD.org)>.

<ftp://ftp.jp.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp2.jp.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp3.jp.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp4.jp.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp5.jp.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp6.jp.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp7.jp.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp8.jp.FreeBSD.org/pub/FreeBSD/> (ftp)

<ftp://ftp9.jp.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 대한민국

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@kr.FreeBSD.org](mailto:hostmaster@kr.FreeBSD.org)>.

- <ftp://ftp.kr.FreeBSD.org/pub/FreeBSD/> (ftp/[rsync](http://rsync))

- <ftp://ftp2.kr.FreeBSD.org/pub/FreeBSD/> (ftp)

---

## 리투아니아

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@lt.FreeBSD.org](mailto:hostmaster@lt.FreeBSD.org)>.

- <ftp://ftp.lt.FreeBSD.org/pub/FreeBSD/> (ftp)

## 네덜란드

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@nl.FreeBSD.org](mailto:hostmaster@nl.FreeBSD.org)>.

- <ftp://ftp.nl.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://)/rsync)
- <ftp://ftp2.nl.FreeBSD.org/pub/FreeBSD/> (ftp)

## 노르웨이

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@no.FreeBSD.org](mailto:hostmaster@no.FreeBSD.org)>.

- <ftp://ftp.no.FreeBSD.org/pub/FreeBSD/> (ftp/rsync)
- <ftp://ftp3.no.FreeBSD.org/pub/FreeBSD/> (ftp)

## 폴란드

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@pl.FreeBSD.org](mailto:hostmaster@pl.FreeBSD.org)>.

- <ftp://ftp.pl.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp2.pl.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp3.pl.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp5.pl.FreeBSD.org/pub/FreeBSD/> (ftp)

## 포르투갈

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@pt.FreeBSD.org](mailto:hostmaster@pt.FreeBSD.org)>.

- <ftp://ftp.pt.FreeBSD.org/pub/FreeBSD/> (ftp)

- 
- <ftp://ftp2.pt.FreeBSD.org/pub/freebsd/> (ftp)
  - <ftp://ftp4.pt.FreeBSD.org/pub/ISO/FreeBSD/> (ftp)

#### 루마니아

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@ro.FreeBSD.org](mailto:hostmaster@ro.FreeBSD.org)>.

- <ftp://ftp.ro.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp1.ro.FreeBSD.org/pub/FreeBSD/> (ftp/ftpv6)

#### 러시아

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@ru.FreeBSD.org](mailto:hostmaster@ru.FreeBSD.org)>.

- <ftp://ftp2.ru.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp3.ru.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp4.ru.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 사우디 아라비아

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[ftpadmin@isu.net.sa](mailto:ftpadmin@isu.net.sa)>.

- <ftp://ftp.isu.net.sa/pub/mirrors/ftp.freebsd.org/> (ftp)

#### 싱가포르

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@sg.FreeBSD.org](mailto:hostmaster@sg.FreeBSD.org)>.

- <ftp://ftp.sg.FreeBSD.org/pub/FreeBSD/> (ftp/[http/rsync](http://rsync))

#### 슬로바키아 공화국

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@sk.FreeBSD.org](mailto:hostmaster@sk.FreeBSD.org)>.

- 
- <ftp://ftp.sk.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 슬로베니아

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@si.FreeBSD.org](mailto:hostmaster@si.FreeBSD.org)>.

- <ftp://ftp.si.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp2.si.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 남아프리카

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@za.FreeBSD.org](mailto:hostmaster@za.FreeBSD.org)>.

- <ftp://ftp.za.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp2.za.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp3.za.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp4.za.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 스페인

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@es.FreeBSD.org](mailto:hostmaster@es.FreeBSD.org)>.

- <ftp://ftp.es.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp2.es.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp3.es.FreeBSD.org/pub/FreeBSD/> (ftp)

#### 스웨덴

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@se.FreeBSD.org](mailto:hostmaster@se.FreeBSD.org)>.

- <ftp://ftp.se.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp2.se.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp3.se.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp5.se.FreeBSD.org/pub/FreeBSD/> (ftp/[http/rsync](http://rsync))



---

## 스위스

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@ch.FreeBSD.org](mailto:hostmaster@ch.FreeBSD.org)>.

- <ftp://ftp.ch.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp2.ch.FreeBSD.org/pub/FreeBSD/> (ftp)

## 대만

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@tw.FreeBSD.org](mailto:hostmaster@tw.FreeBSD.org)>.

- <ftp://ftp.tw.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp2.tw.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://)/rsync)
- <ftp://ftp3.tw.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp4.tw.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp5.tw.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp6.tw.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://))
- <ftp://ftp7.tw.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp8.tw.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp9.tw.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp10.tw.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp11.tw.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://))
- <ftp://ftp12.tw.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp13.tw.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp14.tw.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp15.tw.FreeBSD.org/pub/FreeBSD/> (ftp)

## 터키

- <ftp://ftp.tr.FreeBSD.org/pub/FreeBSD/> (ftp)

## 우크라이나

- <ftp://ftp.ua.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp2.ua.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp3.ua.FreeBSD.org/pub/FreeBSD/> (ftp)

- 
- <ftp://ftp5.ua.FreeBSD.org/pub/FreeBSD/> (ftp)
  - <ftp://ftp6.ua.FreeBSD.org/pub/FreeBSD/> (ftp)
  - <ftp://ftp7.ua.FreeBSD.org/pub/FreeBSD/> (ftp)
  - <ftp://ftp8.ua.FreeBSD.org/pub/FreeBSD/> (ftp)

## 영국

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@uk.FreeBSD.org](mailto:hostmaster@uk.FreeBSD.org)>.

- <ftp://ftp.uk.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp2.uk.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://)/rsync)
- <ftp://ftp3.uk.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp4.uk.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp5.uk.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp6.uk.FreeBSD.org/pub/FreeBSD/> (ftp)

## 미국

문제가 있다면 이 도메인의 호스트 관리자에게 연락한다  
<[hostmaster@us.FreeBSD.org](mailto:hostmaster@us.FreeBSD.org)>.

- <ftp://ftp1.us.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp2.us.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp3.us.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp4.us.FreeBSD.org/pub/FreeBSD/> (ftp/ftpv6)
- <ftp://ftp5.us.FreeBSD.org/pub/FreeBSD/> (ftp/rsync)
- <ftp://ftp6.us.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://))
- <ftp://ftp7.us.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://)/rsync)
- <ftp://ftp8.us.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp9.us.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://))
- <ftp://ftp10.us.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp11.us.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp12.us.FreeBSD.org/pub/FreeBSD/> (ftp/rsync)
- <ftp://ftp13.us.FreeBSD.org/pub/FreeBSD/> (ftp/[http](http://)/rsync)
- <ftp://ftp14.us.FreeBSD.org/pub/FreeBSD/> (ftp)
- <ftp://ftp15.us.FreeBSD.org/pub/FreeBSD/> (ftp)

---

## A.3 익명 CVS

### A.3.1 소개

익명 CVS 는 원격 CVS 저장소와 동기화하기 위해 FreeBSD 에 번들로 포함된 CVS 유틸리티가 제공하는 기능이다. 특별한 권한 없이 FreeBSD 프로젝트의 공식 익명 CVS 서버 중 한곳에서 읽기전용 CVS 태스크를 수행할 수 있다. 익명 CVS 서버를 사용하려면 CVSROOT 환경 변수를 적당한 익명 서버로 지정하고 cvs login 명령과 유명한 패스워드 “anoncvs”을 입력하여 로컬 저장소처럼 cvs(1) 명령으로 접근한다.

**Note:** cvs login 명령은 CVS 서버 인증에 사용되는 패스워드를 여러분의 HOME 디렉터리에 .cvspass 라는 파일로 저장한다. 이 파일이 없으면 cvs login 을 처음 사용할 때 에러가 발생할 것이다. 빈 .cvspass 파일을 생성하고 다시 로그인 한다.

CVSup 과 anoncvs 서비스는 기본적으로 동일한 기능을 수행하지만 유저가 동기화 방법을 선택하는데 영향을 끼치는 다양한 요소가 있다. 가격 면에서는 아니지만 CSVup 은 네트워크 리소스 사용량에서 더 효과적이고 둘 중 기술적으로 더 가치가 있어 보인다. CVSup 을 사용하기 위해서, CVSup 가 *클렉션*이라고 하는 상당히 큰 비트를 받기 전에 특별한 클라이언트를 먼저 설치하고 설정해야 된다.

Anoncvs 와 비교하면 CVSup 은 CVS 모듈 이름을 참조하여 각 파일에서 특정 프로그램(is 나 grep 같은)까지 모든 부분에 사용할 수 있다. 물론 anoncvs 도 CVS 저장소에서 읽기는 잘 하지만 FreeBSD 프로젝트에 공유된 저장소 하나를 로컬 개발 지원에 사용하려고 한다면 CVSup 만 사용할 수 있다.

### A.3.2 익명 CVS 사용

익명 CVS 저장소를 사용하기 위한 cvs(1) 설정은 단순히 CVSROOT 환경 변수에 FreeBSD 프로젝트의 anoncvs 서버를 설정하면 된다. 이 글을 작성할 때 다음과 같은 서버를 사용할 수 있다:

- *오스트리아:* pserver:anoncvs@anoncvs.at.FreeBSD.org:/home/ncvs (cvs login을 사용하고 패스워드 프롬프트가 나타나면 아무 패스워드나 넣는다).

- 
- 프랑스: pserver:anoncvs@anoncvs.fr.FreeBSD.org:/home/ncvs (pserver (password ``anoncvs"), ssh (패스워드 없음)).
  - 독일: pserver:anoncvs@anoncvs.de.FreeBSD.org:/home/ncvs (cvs login을 사용하고 패스워드 프롬프트가 나타나면 "anoncvs"를 입력한다).
  - 독일: pserver:anoncvs@anoncvs2.de.FreeBSD.org:/home/ncvs (rsh, pserver, ssh, ssh/2022).
  - 일본: pserver:anoncvs@anoncvs.jp.FreeBSD.org:/home/ncvs (cvs login을 사용하고 패스워드 프롬프트가 나타나면 "anoncvs"를 입력한다)
  - 스웨덴: freebsdanoncvs@anoncvs.se.FreeBSD.org:/home/ncvs (ssh만 사용하고 패스워드 없음)
  - USA: freebsdanoncvs@anoncvs.FreeBSD.org:/home/ncvs (ssh만 사용하고 패스워드 없음)

CVS 는 어떤 버전의 FreeBSD 소스가 존재했는지 한번만 체크하기 때문에 cvs(1)의 수정(revision) 플래그(-r)와 FreeBSD 프로젝트 저장소가 허용하는 값을 능숙하게 처리해야 된다.

수정(revision) 태그와 분기(branch) 태그 두 종류가 있다. 수정 태그는 수정된 것을 의미한다. 이 의미는 날짜 별로 동일하다는 것이다. 이와 반대로 분기 태그는 특정 시간에 특정 개발 라인에서 가장 최근에 수정된 것을 의미한다. 분기 태그는 특별히 수정된 것을 의미하지 않기 때문에 오늘과 내일의 버전이 약간 다르다.

섹션 A.6 에는 유저가 흥미 있어할 만한 분기 태그를 포함하고 있다. 다시 말해 포트 컬렉션에 여러 개의 수정 본이 있지 않기 때문에 이들을 포트 컬렉션으로 사용할 수 없다.

분기 태그를 지정하면 보통 그 개발 라인의 가장 최근에 수정된 파일을 받게 된다. 과거의 특정 버전을 원한다면 `-D date` 플래그로 날짜를 지정할 수 있다. 더 자세한 내용은 cvs(1) 매뉴얼 페이지를 본다.

### A.3.3 예제

설정을 하기 전에 cvs(1) 매뉴얼 페이지를 읽어볼 것을 권장하고, 여기서는 익명 CVS 를 사용하는 몇 가지 예제를 보여준다.

---

**예제 A-1. -CURRENT 에서 (ls(1)) 무언가를 체크해서 다시 삭제한다:**

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.FreeBSD.org:/home/ncvs
% cvs login
At the prompt, enter the password ``anoncvs``.
% cvs co ls
% cvs release -d ls
% cvs logout
```

**예제 A-2. 3.X-STABLE 분기에서 ls(1)의 버전 체크:**

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.FreeBSD.org:/home/ncvs
% cvs login
At the prompt, enter the password ``anoncvs``.
% cvs co -rRELENG_3 ls
% cvs release -d ls
% cvs logout
```

**예제 A-3. ls(1)의 변경된 리스트 생성:**

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.FreeBSD.org:/home/ncvs
% cvs login
At the prompt, enter the password ``anoncvs``.
% cvs rdiff -u -rRELENG_3_0_0_RELEASE -rRELENG_3_4_0_RELEASE ls
% cvs logout
```

**예제 A-4. 어떤 모듈 이름을 사용할 수 있는지 찾는다:**

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.FreeBSD.org:/home/ncvs
% cvs login
At the prompt, enter the password ``anoncvs``.
% cvs co modules
% more modules/modules
% cvs release -d modules
% cvs logout
```

---

## A.3.4 다른 리소스

추가적으로 다음 리소스가 CVS 사용법을 배우는 것을 도울 수 있다:

Cal Poly의 [CVS Tutorial](#).

[CVS Home](#), CVS 개발과 지원 커뮤니티.

[CVSweb](#)은 CVS를 위한 FreeBSD 프로젝트 웹 인터페이스다.

## A.4 CTM 사용

CTM은 중앙의 디렉터리 트리와 원격 디렉터리 트리를 동기화하는 방법이다. FreeBSD 소스 트리에 사용하기 위해 개발되었지만, 시간이 지나면서 다른 사람들이 다른 목적에도 유용하다는 것을 발견했을 것이다. Deltas 생성 절차에 대한 문서가 있기 때문에, CTM을 다른 목적에 사용하기를 원한다면, ctm-users 메일링 리스트에서 (<http://lists.freebsd.org/mailman/listinfo/ctm-users>) 더 많은 정보를 얻을 수 있다.

### A.4.1 CTM을 왜 사용해야 되는가?

CTM은 FreeBSD 소스 트리를 로컬에 복사해 준다. 트리를 사용할 수 있는 여러 가지 방법이 있다. 전체 CVS 트리를 받을 것인지 또는 특정 분기만을 받을 것인지에 따라, CTM이 정보를 제공할 수 있다. 여러분이 FreeBSD에서 활발히 활동중인 개발자고 TCP/IP 연결이 없거나 형편없다면, 또는 자동으로 여러분에게 보내 주는 것으로 변경하고 싶다면 CTM이 좋은 선택이다. 아주 활동적인 분기라면 하루에 3번의 deltas를 받아야 된다. 그러나 이들 deltas를 자동으로 어떻게 메일로 보내야 되는지 고려해야 된다. 이것은 보통 5K 미만이고, 종종 10-50K이고(10번에 한번) 이제 100K+ 이상이 되었다.

---

## A.4.2 CTM 을 어느 곳에 사용해야 되는가?

두 가지가 필요하다: **CTM** 프로그램과 최초 deltas(현재 레벨 이상의 것을 받기 위해).

**CTM** 프로그램은 버전 2.0 부터 FreeBSD 에 포함되었고, 소스를 복사했다면 /usr/src/usr.sbin/ctm 에 둔다.

FreeBSD 2.0 이전의 버전을 사용한다면, 다음 사이트에서 직접 현재 CTM 으로 패치할 수 있다:

<http://www.FreeBSD.org/cgi/cvsweb.cgi/src/usr.sbin/ctm/>

**CTM** 에게 “deltas”를 제공하기 위해 두 가지 방법을 사용할 수 있다: FTP 나 전자 메일. 일반적으로 FTP 를 사용할 수 있다면 다음 FTP 사이트로 CTM 을 사용하거나 미리 ([http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/handbook/ctm.html#MIRRORS-CTM](http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/ctm.html#MIRRORS-CTM)) 섹션을 본다:

<ftp://ftp.FreeBSD.org/pub/FreeBSD/CTM/>

FTP 관련 디렉터리와 패치는 README 파일에 나와있는 곳에서 할 수 있다.

전자 메일을 통해 deltas 를 받으려면:

**CTM** 배포 리스트 중 한곳에 등록한다. ctm-cvs-cur

(<http://lists.freebsd.org/mailman/listinfo/ctm-cvs-cur>)가 전체 CVS 트리를 지원한다.

ctm-src-sur은 개발 분기의 헤드(head) 지원한다. ctm-src-4 는

(<http://lists.freebsd.org/mailman/listinfo/ctm-src-4>) 4.X 릴리즈 분기를 제공한다(리스트에 등록하는 방법을 모른다면, 위의 리스트 이름을 클릭하거나

<http://lists.FreeBSD.org/mailman/listinfo>에서 등록하고 싶은 리스트를 클릭한다. 리스트 페이지는 등록에 필요한 설명도 포함하고 있다).

메일로 **CTM** 업데이트를 받기 시작할 때 ctm\_rmail 프로그램을 사용하여 압축을 해제하고 적용할 수 있다. 완전히 자동으로 프로세스를 실행시키려면 /etc/aliases 의 엔트리에서 직접 ctm\_rmail 프로그램을 사용할 수 있다. 더 자세한 사항은 ctm\_rmail 매뉴얼 페이지를 참고 한다.

---

**Note:** CTM deltas 를 받기 위해 어떤 방법을 쓰더라도 문제없고, ctm-announce 메일링 리스트에 등록 하기 바란다. 나중에 유일하게 CTM 시스템 운용에 관한 통보가 게시되는 곳이다. 위의 리스트 이름을 클릭하고 지시에 따라 리스트에 등록한다.

### A.4.3 처음으로 CTM 사용하기

순서대로 deltas 를 생성하기 위해 시작 위치가 필요하다.

첫째로 무엇을 가지고 있는지 결정해야 된다. 모든 사람들이 빈 디렉터리에서부터 시작한다. 지원되는 트리로 CTM 을 시작하려면 최초 “빈” delta 를 사용해야 된다. deltas 를 시작하는 방법 중 하나는 배포된 CD 를 통해서지만, 지금은 배포하지 않는다.

트리가 수십 MB 이기 때문에 가지고 있는 것으로 시작하는 것이 좋다. -RELEASE CD 를 가지고 있다면, 이것을 복사하거나 초기 소스를 추출할 수 있다. 중요한 데이터를 전송하여 저장한다.

이들 “starter” deltas 는 X에 추가된 번호로(예를 들어 src-cur.3210XEmpty.gz) 인지할 수 있다. 다음 의미는 초기 “seed”의 유래에 X가 대응한다. Empty 는 빈 디렉터리다. Empty 에서 변화된 를 기반은 모든 100 deltas 를 생성한다. 이때 이들 70 에서 80 MB 의 gzip 데이터는 XEmpty deltas 에 공통이다.

시작에 기본 delta 를 선택하면 이것을 따르는 높은 번호의 모든 deltas 도 필요하다.

### A.4.4 일상적으로 CTM 사용하기

Deltas 를 적용하려면 다음 명령을 실행한다:

```
# cd /where/ever/you/want/the/stuff
# ctm -v -v /where/you/store/your/deltas/src-xxx.*
```



---

gzip 으로 압축된 deltas 를 CTM 이 알고 있기 때문에 gunzip 을 할 필요가 없고, 디스크 공간을 절약한다.

전체 과정이 매우 안전하다고 감지 되지 않으면 **CTM** 은 트리를 건드리지 않는다. delta 를 확인하기 위해 `-c` 플래그를 사용할 수 있고 실제로 **CTM** 은 트리를 건드리지 않는다; 이것은 단지 delta 통합을 확인하고 여러분의 현재 트리에 정확히 적용되었는지 확인한다.

**CTM** 에 다른 옵션들도 있기 때문에 더 많은 정보는 매뉴얼 페이지나 소스를 참고 한다.

새로운 delta 를 받을 때마다 위 명령을 실행하면 **CTM** 은 소스를 최신으로 유지한다.

deltas 를 다시 다운로드 하기가 어렵다면 삭제하지 않는다. 문제가 생길지 모르기 때문에 deltas 를 가지고 있기를 원할 것이다. 플로피 디스크만 가지고 있다면 복사하기 위해 `fdwrite` 사용을 생각해본다.

## A.4.5 로컬 변경 유지하기

소스 트리에서 파일 변경 경험이 있는 개발자라면, **CTM** 은 제한된 방법으로 로컬 변경을 지원한다: `foo` 파일이 있는지 확인하기 전에 `foo.ctm` 을 찾아야 된다. 이 파일이 있다면 **CTM** 은 `foo` 대신 이것을 사용한다.

이 동작은 로컬 변경을 간단히 관리하도록 한다: 확장자 `.ctm` 이 붙은 파일 이름과 일치하는, 수정하려는 파일을 단순히 복사한다. 그리고 **CTM** 이 `.ctm` 파일을 업데이트 하는 동안 자유롭게 코드를 해킹할 수 있다.

## A.4.6 흥미 있는 다른 CTM 옵션

---

## A.4.6.1 업데이트하는 파일이 정확히 어떤 것인지 확인하기

CTM 에 `-r` 옵션을 사용하여 CTM 이 소스 저장소에서 변경할 리스트를 확인할 수 있다.

이전 또는 이후 절차에서 파일이 수정되었는지 확인하거나 약간 의심될 때, 변경된 내용을 로그로 남기기 위해 유용하다.

## A.4.6.2 업데이트하기 전에 백업

가끔 CTM 업데이트가 변경할 파일을 백업하기를 원할 것이다.

`-B backup-file` 옵션으로, CTM 은 backup-file 에 주어진 CTM delta 가 업데이트할 모든 파일을 백업한다.

## A.4.6.3 업데이트 하는 파일 제한

가끔 주어진 CTM 업데이트의 범위 제한을 하거나 deltas 순서에서 약간의 파일만 추출하는 것에 관심을 가질 수 있다.

`-e` 와 `-x` 옵션을 사용하여 지정한 정규 표현 식 필터로 CTM 이 동작하는 파일 리스트를 제어할 수 있다.

예를 들어 저장된 CTM deltas 컬렉션에서 최신 lib/libc/Makefile 복사본을 추출하려면 다음 명령을 실행한다:

```
# cd /where/ever/you/want/to/extract/it/  
# ctm -e '^lib/libc/Makefile' ~ctm/src-xxx.*
```

CTM delta 에서 지정된 모든 파일을 위해, `-e` 와 `-x` 옵션이 주어진 명령어 라인에 순서대로

---

적용된다. eligible 로 표시되었다면 `-e` 와 `-x` 옵션이 적용된 후 이 파일은 **CTM** 으로만 진행 된다

## A.4.7 CTM 의 나중 계획

CTM 시스템에 어떤 인증을 사용하기 때문에, 거짓 CTM 업데이트를 감지 할 수 있다. CTM 옵션은 혼돈되고 직관적이지 않기 때문에 깨끗이 정리한다.

## A.4.8 잡다한 것들

포트 컬렉션을 위한 deltas 순서가 있지만 별로 중요하지는 않다.

## A.4.9 CTM 미러

CTM/FreeBSD 는 다음 미러 사이트에서 익명 FTP 를 통해 사용할 수 있다. 익명 FTP 를 통해 **CTM** 을 받는다면 가까운 사이트를 찾는다.

문제가 발생하면 `ctm-users` 메일링 리스트에 연락한다.

캘리포니아, Bay Area, 공식 소스

<ftp://ftp.FreeBSD.org/pub/FreeBSD/development/CTM/>

남아프리카, 예전 deltas 백업 서버

<ftp://ftp.za.FreeBSD.org/pub/FreeBSD/CTM/>

대만/R.O.C.

<ftp://ctm.tw.FreeBSD.org/pub/FreeBSD/development/CTM/>

<ftp://ctm2.tw.FreeBSD.org/pub/FreeBSD/development/CTM/>

<ftp://ctm3.tw.FreeBSD.org/pub/FreeBSD/development/CTM/>

---

미러 사이트를 찾지 못했거나 불완전하다면, [alltheweb](http://www.alltheweb.com/)(<http://www.alltheweb.com/>)같은 검색엔진을 사용해 본다.

## A.5 CVSUp 사용

### A.5.1 소개

**CVSUp**은 마스터 CVS 저장소에서 원격 서버 호스트의 소스 트리를 배포하고 업데이트 하기 위한 소프트웨어 패키지다. FreeBSD 소스는 캘리포니아의 중앙 개발 머신의 CVS 저장소에서 관리하고 있다. **CVSUp**으로 FreeBSD 유저는 자신의 소스 트리를 업데이트 할 수 있다.

**CVSUp**은 *pull* 모델이라는 업데이트를 사용한다. Pull 모델에서 각 클라이언트는 원할 때 서버에 업데이트 요청을 한다. 서버는 클라이언트로부터 업데이트 요청을 기다린다. 따라서 모든 업데이트는 클라이언트가 주도한다. 서버는 절대 업데이트 요청을 보내지 않는다. 유저는 직접 업데이트하기 위해 **CVSUp** 클라이언트를 실행하거나 보통 자동으로 실행하기 위해 cron 잡을 설정해야 된다.

**CVSUp** 용어는 전체 소프트웨어 패키지를 의미한다. 메인 컴포넌트는 각 유저 머신에서 실행하는 클라이언트 `cvsup` 이고, 각 FreeBSD 미러 사이트에서 실행하는 `cvsupd` 다.

FreeBSD 문서와 메일링 리스트를 읽어서 알겠지만 **sup** 레퍼런스를 읽어 보는 것도 좋다. **sup**은 **CVSUp**의 전 모델이고 비슷한 기능을 제공했다. **CVSUp**은 **sup**과 같은 방식으로 사용되기 때문에 **sup**와 호환되는 설정 파일을 사용한다. **CVSUp**이 빠르고 더 유연하기 때문에 **sup**은 더 이상 FreeBSD 프로젝트에서 사용하지 않는다.

### A.5.2 설치

**CVSUp**을 설치하는 가장 쉬운 방법은 FreeBSD 패키지 컬렉션에 컴파일 되어 있는 [net/cvsup](#) 패키지를 사용한다. 소스에서 **CVSUp**을 빌드하려면 [net/cvsup](#) 포트를 사용한다.

---

그러나 미리 경고하면: [net/cvsup](#) 포트는 다운로드 하고 빌드 하는데 상당한 양의 디스크 공간과 시간이 필요한 Modula-3 시스템에 의존된다.

**Note:** 서버처럼 XFree86 이 설치되어 있지 않은 머신에 CVSup 을 사용하려면 CVSup GUI 가 포함되지 않은 [net/cvsup-without-gui](#) 를 사용한다.

## A.5.3 CVSup 설정

CVSup 운용은 supfile 이라는 설정 파일로 제어한다. /usr/share/examples/cvsup/ 디렉터리에 샘플 supfiles 이 있다.

supfile 에는 CVSup 의 다음과 같은 질문에 대한 정보를 가지고 있다:

- 어떤 파일을 받으려고 하는가?
- 어떤 버전의 파일을 원하는가?
- 어느 곳에서 파일을 받으려는가?
- 머신의 어느 곳에 이 파일을 받으려는가?
- 상태(status) 파일을 어느 곳에 둘 것인가?

다음 섹션에서 우리는 이들 질문에 답하는 전형적인 supfile 을 생성한다. 첫째 우리는 supfile 의 모든 구조를 설명한다.

supfile 은 텍스트 파일이고 #로 시작되는 주석은 라인의 끝까지다. 공백 라인과 주석은 무시된다.

나머지 각 라인은 유저가 받으려는 파일을 설명한다. "collection"으로 시작되는 라인은 서버가 정의한 파일의 논리적인 그룹이다. 컬렉션 이름은 서버에게 원하는 파일을 설명한다. 컬렉션 이름 뒤에 0 이나 더 많은 필드는 공백으로 나눈다. 이들 필드는 위에서 리스트 한 질문의 답이다. 두 종류의 필드가 있다: 플래그 필드와 값 필드. 플래그 필드는 키워드 값으로만 이루어져 있다. 예: *delete* 또는 *compress*. 값 필드도 키워드로 시작하지만, 키워드는 공백 없이 = 문자로 연결된다. 예를 들면 *release=cvs*.

supfile 은 일반적으로 하나 이상의 컬렉션을 지정한다. supfile 를 생성하는 한가지 방법은 각 컬렉션에 관련된 모든 필드를 명시하는 것이다. 그러나 이 방법은 supfile 라인을 상당히

---

길게 만들고, supfile 에서 모든 컬렉션의 대부분의 필드는 같기 때문에 불편하다. CVSup 은 이러한 문제를 해결하는 기본 메커니즘을 제공한다. 특수 pseudo-collection 이름 *\*default* 로 시작되는 라인은, supfile 에서 subsequent 컬렉션에 기본적으로 사용되는 플래그와 값을 설정하는데 사용된다. 기본값은 컬렉션에 다른 값을 지정하여 각 컬렉션에서 무시할 수 있다. 기본값은 변경하거나 supfile 중간에 *\*default* 라인을 추가하여 확장할 수 있다.

이를 바탕으로 우리는 FreeBSD-CURRENT 의 메인 소스 트리를 받아서 업데이트 하는 supfile 을 생성한다.

어떤 파일을 받으려고 하는가?

CVSup 으로 사용할 수 있는 파일은 “collections”이라는 그룹으로 구성된다. 사용할 수 있는 컬렉션은 다음 섹션에서 설명한다. 이 예제에서 우리는 FreeBSD 시스템에 전체 메인 소스 트리를 받는다. 우리에게 모든 소스를 주는 대형 컬렉션 *src-all*이 있다. 우리의 supfile 을 생성하기 위한 첫 단계로 라인당 컬렉션 하나씩을 입력한다(여기서는 오직 한 라인만):

```
src-all
```

어떤 버전의 파일을 원하는가?

CVSup 으로 어떤 버전이든지 받을 수 있다. cvsupd 서버는 모든 버전을 담고 있는 CVS 저장소에서 운용되기 때문에 가능하다. 이들 중 어떤 것을 원하는지 *tag=* 와 *date=* 값 필드를 사용하여 지정 한다.

**주의:** 몇몇 태그는 특정 컬렉션의 파일에만 적용할 수 있기 때문에 *tag=* 필드를 지정할 때 주의해야 된다. 부정확하거나 철자가 빠진 태그를 지정하면 원하지 않는 파일을 CVSup 이 삭제 한다. 특히 *tag=*은 *ports-\** 컬렉션에만 사용한다.

*tag=*필드 이름은 저장소의 심볼릭 태그다. 수정(revision) 태그와 분기(branch) 태그 두 종류가 있다. 수정 태그는 수정된 것을 의미한다. 이 의미는 날짜 별로 동일하다는 것이다. 이와 반대로 분기 태그는 특정 시간에 특정 개발 라인에서 가장 최근에 수정된 것을 의미한다. 분기 태그는 특별히 수정된 것을 의미 하지 않

---

기 때문에 오늘과 내일 버전이 약간 다르다.

섹션 A.6 은 유저가 흥미 있어할 만한 분기 태그를 가지고 있다. CVSup 설정 파일에 태그를 지정할 때 *tag=* 이라는 구문을 먼저 적는다(*RELENG\_4*은 *tag=RELENG\_4*가 된다). *tag=.*은 포트 컬렉션과 관계 있다는 것을 꼭 기억한다.

**주의:** 보여준 것처럼 정확하게 태그 이름을 입력한다. CVSup 은 유효한 태그인지 아닌지 분간하지 못한다. 태그 철자를 잘못 입력했다면, 어떤 파일도 지정하지 않는 유효한 태그처럼 CVSup 이 작동한다.

분기 태그를 지정하면 보통 그 개발 라인의 가장 최근 버전의 파일을 받게 된다. 이전 버전을 원한다면 *date=*값 필드로 특정 날짜를 지정할 수 있다. *cvsup(1)* 매뉴얼 페이지에서 자세히 설명한다.

우리 예제에서 우리는 FreeBSD-CURRENT 를 받으려고 한다. 우리는 *supfile* 의 시작 라인에 다음 라인을 추가한다:

```
*default tag=.
```

*tag=* 필드와 *date=* 필드를 지정하지 않으면 작동하는, 특별히 중요한 경우가 있다. 이 경우 서버의 CVS 저장소에서 특정 버전을 받지 않고 실제 RCS 파일을 받게 된다. 개발자들은 보통 이 모드를 선호 한다. 저장소 복사본을 자신들의 시스템에서 운영하여 수정된 상황과 이전 버전 파일을 확인할 수 있다. 그러나 이것은 엄청난 디스크 공간이 소모된다.

어느 곳에서 파일을 받으려는가?

업데이트를 어디에서 받을 것인지 *cvsup* 에게 알려주는 *host=* 필드를 사용한다. 모든 CVSup 미러 사이트에서 받을 수 있지만 가장 가까운 곳을 설정한다. 이 예제에서 우리는 가상의 FreeBSD 배포 사이트 *cvsup666.FreeBSD.org* 를 사용한다:

```
*default host=cvsup666.FreeBSD.org
```

CVSup 을 실행하기 전에 실제 존재하는 호스트로 변경해야 된다. 명령어 라인에 *-h hostname* 으로 호스트 설정을 무시할 수 있다.

---

머신의 어느 곳에 이 파일을 받을 것인가?

*prefix*= 필드는 파일을 어느 곳에 받을지 **CVSup** 에게 알려 준다. 이 예제에서 우리는 직접 소스 파일을 우리의 메인 소스 트리 `/usr/src` 에 받는다. `src` 디렉터리는 우리가 받으려는 컬렉션을 의미하기 때문에 다음 라인이 정확하다:

```
*default prefix=/usr
```

상태(status) 파일을 어느 곳에 둘 것인가?

**CVSup** 클라이언트는 특정 상태 파일을 “base”라는 디렉터리에 보관한다. 이들 파일은 업데이트한 파일의 기록을 가지고 있어서 **CVSup** 이 더 효과적으로 동작하도록 한다. 우리는 표준 base 디렉터리 `/usr/local/etc/cvsup` 을 사용한다:

```
*default base=/usr/local/etc/cvsup
```

이 설정이 `supfile` 에 지정되어 있지 않더라도 기본적으로 사용되기 때문에 위의 라인은 사실 필요 없다. base 디렉터리가 없다면 이제 생성하면 된다. **CVSup** 클라이언트는 base 디렉터리가 없다면 실행을 거부한다.

잡다한 `supfile` 설정:

일반적으로 `supfile` 에 있어야 되는 하나 이상의 라인이 있다:

```
*default release=cvs delete use-rel-suffix compress
```

*release=cvs* 는 서버가 메인 FreeBSD CVS 저장소 밖에서 정보를 받는다는 것을 표시한다. 사실 대부분의 경우가 이렇지만 이 설명서의 범위를 벗어나는 다른 가능성도 있다.

*delete* 는 파일을 삭제할 수 있는 퍼미션을 **CVSup** 에게 준다. 항상 이렇게 지정하여 **CVSup** 이 소스 트리를 완전히 업데이트 하도록 한다. **CVSup** 은 적합한 이들 파일만 삭제하기 위해 주의하고, 추가 적인 파일을 가지고 있다면 완전히 따로 남



---

는다.

*use-rel-suffix*는 난해하다. 이에 관해 알고 싶다면 `cvsup(1)` 매뉴얼 페이지를 확인한다. 그렇지 않으면 그냥 지정하고 신경 쓰지 않는다.

*compress*는 통신 채널에 `gzip` 타입의 압축 사용을 활성화 한다. 네트워크 링크가 T1 이나 그 이상이면 압축을 사용하지 않는 것이 좋다.

모두 모으기:

여기 전체 `supfile` 예제가 있다:

```
*default tag=.  
*default host=cvsup666.FreeBSD.org  
*default prefix=/usr  
*default base=/usr/local/etc/cvsup  
*default release=cvsv delete use-rel-suffix compress  
  
src-all
```

### A.5.3.1 refuse 파일

위에서 말했듯이 **CVSup**은 *pull* 방식을 사용한다. 기본적으로 이 의미는 **CVSup** 서버에 연결하면 서버가 받아갈 수 있는 것을 알려주고, 클라이언트는 원하는 것을 받아간다. 기본 설정에서 **CVSup** 클라이언트는 설정 파일에서 선택한 컬렉션과 태그에 관련된 모든 파일을 받아간다. 그러나 원하는 것을 항상 받아갈 수 있는 것은 아니다. 특히 `doc`, `ports`, 또는 `www` 트리 -- 대부분의 사람들이 4 ~ 5 개 언어를 읽지 못하기 때문에 언어와 연관된 파일은 다운로드 할 필요 없다. 포트 컬렉션을 **CVSup** 한다면 각 컬렉션을 지정하여 받을 수 있다(예: 단순히 *ports-all*이라고 하지 않고 *ports-astrology*, *port-biology* 등). 그러나 `doc`와 `www` 트리는 언어와 관련된 컬렉션을 가지고 있지 않기 때문에 **CVSup**의 다양한 기능 중 한가지를 사용해야 된다: `refuse` 파일

`refuse` 파일은 본질적으로 **CVSup**에게 컬렉션에서 모든 싱글 파일을 받지 않도록 한다; 다

---

시 말해 클라이언트에게 서버의 특정 파일을 거부하도록 한다. refuse 파일은 *base/sup/*에서 찾을 수 있다(아직 디렉터리를 지정하지 않았다면 지정한다). *base*는 *supfile*에 정의한다; 기본적으로 *base*가 */usr/local/etc/cvsup*이라는 말은, 기본 refuse 파일이 */usr/local/etc/cvsup/sup/refuse*이라는 말이다.

refuse 파일은 매우 단순한 포맷을 가지고 있다; 단순히 다운로드 하지 않으려는 파일이나 디렉터리 이름을 가지고 있다. 예를 들어 영어와 독일어 약간을 제외한 모든 언어와, 독일어 어플리케이션(또는 영어를 제외한 다른 언어용 어플리케이션) 사용이 필요 없을 거라면 refuse 파일에 다음 목록을 입력한다:

ports/chinese  
ports/french  
ports/german  
ports/hebrew  
ports/hungarian  
ports/japanese  
ports/korean  
ports/polish  
ports/portuguese  
ports/russian  
ports/ukrainian  
ports/vietnamese  
doc/da\_\*  
doc/de\_\*  
doc/el\_\*  
doc/es\_\*  
doc/fr\_\*  
doc/it\_\*  
doc/ja\_\*  
doc/nl\_\*  
doc/no\_\*  
doc/pl\_\*  
doc/pt\_\*  
doc/ru\_\*  
doc/sr\_\*  
doc/zh\_\*

---

다른 언어도 이런 식으로 설정한다(FreeBSD CVS 저장소를 확인하여 전체 리스트를 볼 수 있다).

유용한 이런 기능으로, 인터넷이 느리거나 인터넷을 사용한 시간으로 요금을 계산하는 유저는, 사용하지 않는 파일을 다운로드 하지 않아서 시간을 절약 할 수 있다. `refuse` 파일과 **CVSup** 의 다른 유용한 기능은 매뉴얼 페이지를 참고한다.

## A.5.4 CVSup 실행

이제 업데이트 준비가 끝났다. 업데이트 하는 명령어 라인은 상당히 간단 하다:

```
# cvsup supfile
```

`supfile` 은 방금 생성한 `supfile` 의 이름이다. 여러분이 X11 에서 실행한다면, `cvsup` 은 일반적으로 사용하는 버튼을 가진 GUI 윈도우를 보여준다. `go` 버튼을 누르고 실행을 확인한다.

이 예제에서 실제 `/usr/src` 를 업데이트하기 때문에 `root` 로 프로그램을 실행해야 되고, 따라서 `cvsup` 은 파일을 업데이트 할 수 있는 퍼미션을 가지게 된다. 방금 생성한 설정 파일로 이 프로그램을 한번도 실행해 보지 않았다면 약간 불안 할 것이다. 실제 중요한 파일을 건드리지 않고 쉽게 테스트할 수 있는 방법이 있다. 편리한 곳에 빈 디렉토리를 생성하고 명령어 라인의 추가적인 인자로 이 이름을 입력한다:

```
# mkdir /var/tmp/dest
# cvsup supfile /var/tmp/dest
```

방금 지정한 디렉터리가 모든 파일 업데이트의 목적 디렉터리로 사용된다. **CVSup** 은 `/usr/src` 에서 일반적인 파일을 검토하지만 파일을 수정하거나 삭제하지 않는다. 파일 업데이트는 `/var/tmp/dest/usr/src` 에서 시행된다. 또한 **CVSup** 는 기본 디렉터리의 상태 파일도 수정하지 않는다. 이들 파일의 새로운 버전은 지정된 디렉터리에 받아진다. 이런 식의 테스트는 `/usr/src` 에 읽기 퍼미션만 있으면 `root` 계정으로 실행할 필요가 없다.

X11 을 실행하지 않았거나 GUI 를 싫어한다면, `cvsup` 을 실행할 때 몇 개의 옵션을 추가해

---

아 된다:

```
# cvsup -g -L 2 supfile
```

`-g`는 **CVSup**에게 GUI를 사용하지 않도록 한다. X11를 사용하지 않는다면 자동으로 적용되는 옵션이지만, 그렇지 않으면 지정해야 된다.

`-L 2`는 **CVSup**에게 업데이트 되는 모든 파일을 자세히 표시하도록 한다. 3 단계의 자세한 표시가 있다: `-L 0`에서 `-L 2`까지. 기본값은 에러 메시지만 표시하는 `0`이다.

다양한 옵션이 있다. 간단히 이들 옵션 리스트를 확인하려면 `cvsup -H`를 입력한다. 더 자세한 설명은 매뉴얼 페이지를 확인한다.

이 방식의 업데이트가 마음에 든다면 `cron(8)`을 사용하여 정기적으로 업데이트 할 수 있다. 물론 `cron(8)`을 통해 업데이트 할 때 **CVSup**은 GUI를 사용하면 안 된다.

## A.5.5 CVSup 파일 컬렉션

**CVSup**을 사용할 수 있는 파일 컬렉션은 계층으로 구성되어 있다. 몇 개의 큰 컬렉션이 있고, 이들은 작은 서브-컬렉션으로 나누어진다. 큰 컬렉션을 받는 것은 서브-컬렉션 각각을 받는 것과 같다. 컬렉션 사이의 계층 관계는 아래 리스트에서 들여쓰기를 사용하여 표현한다.

아주 일반적으로 사용되는 컬렉션은 `src-all`과 `ports-all`이다. 다른 컬렉션은 특별한 목적의 작은 그룹의 사람들이 사용하기 때문에, 어떤 미러 사이트는 이들을 가지고 있지 않을 것이다.

```
cvs-all release=cvs
```

암호화 코드를 포함하고 있는 메인 FreeBSD 저장소.

```
distrib release=cvs
```

배포와 FreeBSD 미러링과 연관된 파일

---

*doc-all release=cvs*

FreeBSD 핸드북 소스와 다른 문서. FreeBSD 웹 사이트의 파일은 포함하지 않는다.

*ports-all release=cvs*

FreeBSD 포트 컬렉션

**중요:** *ports-all* 전체를 업데이트 하지는 않지만, 아래 나와 있는 서브 컬렉션 하나를 사용하려면 항상 *ports-base*를 업데이트 해야 된다. 포트 빌드 인프라스트럭처가 변경될 때마다 *ports-base*에 표시되기 때문에, 변경된 내용이 실제 포트에서 사용된다. 따라서 실제 포트만 업데이트 하고 이들이 새로운 기능만 사용한다면, 알 수 없는 에러 메시지와 함께 빌드가 실패 할 수 있다. 이렇게 되었을 때 가장 처음 해야 될 사항이 *ports-base* 서브 컬렉션을 업데이트 한다.

*ports-archivers release=cvs*

압축 툴

*ports-astro release=cvs*

천문학관련 포트

*ports-audio release=cvs*

사운드 지원

*ports-base release=cvs*

포트 컬렉션은 인프라스트럭처를 빌드 한다 - /usr/ports 서브 디렉터리의 Mk/와 Tools/에 있는 다양한 파일들.

**Note:** 위의 중요 사항을 확인한다: FreeBSD 포트 컬렉션을 업데이트 할 때마다, 이 서브 컬렉션을 항상 업데이트 해야 된다.

*ports-benchmarks release=cvs*

벤치마크

*ports-biology release=cvs*

생물학

---

*ports-cad release=cvs*

컴퓨터 디자인 툴

*ports-chinese release=cvs*

중국어 지원

*ports-comms release=cvs*

통신 소프트웨어

*ports-converters release=cvs*

문자 코드 변환기

*ports-databases release=cvs*

데이터베이스

*ports-deskutils release=cvs*

컴퓨터가 발명되기 전에 책상에서 사용하던 것

*ports-devel release=cvs*

개발 유틸리티

*ports-dns release=cvs*

DNS 관련 소프트웨어

*ports-editors release=cvs*

에디터

*ports-emulators release=cvs*

다른 운영 체제 에뮬레이터

*ports-finance release=cvs*

---

재무와 관련된 필수 어플리케이션

*ports-ftp release=cv*

FTP 클라이언트 서버 유틸리티

*ports-games release=cv*

게임

*ports-german release=cv*

독일어 지원

*ports-graphics release=cv*

그래픽 유틸리티

*ports-hungarian release=cv*

헝가리어 지원

*ports-irc release=cv*

Internet Relay Chat 유틸리티

*ports-japanese release=cv*

일본어 지원

*ports-java release=cv*

Java™ 유틸리티

*ports-korean release=cv*

한국어 지원

*ports-lang release=cv*

프로그래밍 언어

---

*ports-mail release=cv*s

메일 소프트웨어

*ports-math release=cv*s

숫자 계산 소프트웨어

*ports-mbone release=cv*s

MBone 어플리케이션

*ports-misc release=cv*s

잡다한 유틸리티

*ports-multimedia release=cv*s

멀티미디어 소프트웨어

*ports-net release=cv*s

네트워크 소프트웨어

*ports-news release=cv*s

유즈넷 뉴스 소프트웨어

*ports-palm release=cv*s

Palm™ 시리즈 지원 소프트웨어

*ports-polish release=cv*s

폴란드어 지원

*ports-portuguese release=cv*s

포르투갈어 지원

*ports-print release=cv*s

---



---

프린트 소프트웨어

*ports-russian release=cvs*

러시아어 지원

*ports-security release=cvs*

보안 유틸리티

*ports-shells release=cvs*

명령어 라인 셸

*ports-sysutils release=cvs*

시스템 유틸리티

*ports-textproc release=cvs*

텍스트 프로세싱 유틸리티 (데스크톱 출판은 포함하지 않는다).

*ports-vietnamese release=cvs*

베트남어 지원

*ports-www release=cvs*

웹 서비스와 관련된 소프트웨어

*ports-x11 release=cvs*

X 윈도우 시스템 지원 포트

*ports-x11-clocks release=cvs*

X11 clocks.

*ports-x11-fm release=cvs*

X11 파일 매니저

---

*ports-x11-fonts release=cvs*

X11 폰트와 폰트 유틸리티

*ports-x11-toolkits release=cvs*

X11 툴킷

*ports-x11-servers*

X11 서버

*ports-x11-wm*

X11 윈도우 매니저

*src-all release=cvs*

암호화 코드를 포함한 메인 FreeBSD 소스

*src-base release=cvs*

/usr/src 최상 단의 잡다한 파일

*src-bin release=cvs*

싱글 유저 모드에 필요할 수 있는 유저 유틸리티 ([/usr/src/bin](#)).

*src-contrib release=cvs*

상대적으로 수정이 될 된 FreeBSD 프로젝트 외부의 유틸리티와 라이브러리 ([/usr/src/contrib](#)).

*src-crypto release=cvs*

상대적으로 수정이 될 된 FreeBSD 프로젝트 외부의 암호화 유틸리티와 라이브러리 ([/usr/src/crypto](#)).

*src-eBones release=cvs*

---

Kerberos 와 DES ([/usr/src/eBones](#)). 현재 릴리즈의 FreeBSD 에 사용되지 않는다.

*src-etc release=cvs*

시스템 설정 파일 ([/usr/src/etc](#)).

*src-games release=cvs*

게임 ([/usr/src/games](#)).

*src-gnu release=cvs*

GNU 라이선스를 가지고 있는 유틸리티 ([/usr/src/gnu](#)).

*src-include release=cvs*

헤더 파일 ([/usr/src/include](#)).

*src-kerberos5 release=cvs*

Kerberos5 보안 패키지 ([/usr/src/kerberos5](#)).

*src-kerberosIV release=cvs*

KerberosIV 보안 패키지 ([/usr/src/kerberosIV](#)).

*src-lib release=cvs*

라이브러리 ([/usr/src/lib](#)).

*src-libexec release=cvs*

보통 다른 프로그램들이 실행하는 시스템 프로그램 ([/usr/src/libexec](#)).

*src-release release=cvs*

FreeBSD 릴리즈를 만들기 위해 필요한 파일 ([/usr/src/release](#)).

*src-sbin release=cvs*

싱글 유저 모드를 위한 시스템 유틸리티 ([/usr/src/sbin](#)).

---

*src-secure release=cv*s

암호화 라이브러리와 명령 ([/usr/src/secure](#)).

*src-share release=cv*s

여러 시스템으로 공유 할 수 있는 파일 ([/usr/src/share](#)).

*src-sys release=cv*s

커널 ([/usr/src/sys](#)).

*src-sys-crypto release=cv*s

커널 암호 코드 ([/usr/src/sys/crypto](#)).

*src-tools release=cv*s

FreeBSD 관리를 위한 다양한 툴 ([/usr/src/tools](#)).

*src-usrbin release=cv*s

유저 유틸리티 ([/usr/src/usr.bin](#)).

*src-usrsbin release=cv*s

시스템 유틸리티 ([/usr/src/usr.sbin](#)).

*www release=cv*s

FreeBSD WWW 사이트 소스

*distrib release=sel*f

**CVSup** 서버의 설정 파일. **CVSup** 미러 사이트가 사용한다.

*gnats release=cur*rent

GNATS 버그 추적 데이터베이스

*mail-archive release=cur*rent

---

FreeBSD 메일링 리스트 저장소

*www release=current*

이미 처리가 된 FreeBSD WWW 사이트 파일 (소스 파일이 아니다). WWW 미러 사이트에서 사용한다.

## A.5.6 더 많은 정보

CVSup 에 대한 더 많은 정보와 CVSup FAQ 는 CVSup 홈페이지를 (<http://www.polstra.com/projects/freeware/CVSup/>) 참고한다.

FreeBSD 에 관련된 CVSup 설명은 FreeBSD 기술 설명 메일링 리스트를 (<http://lists.freebsd.org/mailman/listinfo/freebsd-hackers>) 참고 한다. 새로운 버전의 소프트웨어는 이곳과 FreeBSD 공고 메일링 리스트에서 (<http://lists.freebsd.org/mailman/listinfo/freebsd-announce>) 발표된다.

질문과 버그 리포트는 프로그램 작성자에게 문의한다 [cvsup-bugs@polstra.com](mailto:cvsup-bugs@polstra.com).

## A.5.7 CVSup 사이트

FreeBSD 를 위한 CVSup 서버는 다음 리스트에서 운용 중 이다.

(2004/06/24 13:15:27 UTC)

중앙 서버

[cvsup.FreeBSD.org](http://cvsup.FreeBSD.org)

메인 미러 사이트

[cvsup1.FreeBSD.org](http://cvsup1.FreeBSD.org)

---

[cvsup2.FreeBSD.org](http://cvsup2.FreeBSD.org)

[cvsup3.FreeBSD.org](http://cvsup3.FreeBSD.org)

[cvsup4.FreeBSD.org](http://cvsup4.FreeBSD.org)

[cvsup5.FreeBSD.org](http://cvsup5.FreeBSD.org)

[cvsup6.FreeBSD.org](http://cvsup6.FreeBSD.org)

[cvsup7.FreeBSD.org](http://cvsup7.FreeBSD.org)

[cvsup8.FreeBSD.org](http://cvsup8.FreeBSD.org)

[cvsup9.FreeBSD.org](http://cvsup9.FreeBSD.org)

[cvsup10.FreeBSD.org](http://cvsup10.FreeBSD.org)

[cvsup11.FreeBSD.org](http://cvsup11.FreeBSD.org)

[cvsup12.FreeBSD.org](http://cvsup12.FreeBSD.org)

[cvsup13.FreeBSD.org](http://cvsup13.FreeBSD.org)

[cvsup14.FreeBSD.org](http://cvsup14.FreeBSD.org)

[cvsup15.FreeBSD.org](http://cvsup15.FreeBSD.org)

[cvsup16.FreeBSD.org](http://cvsup16.FreeBSD.org)

[cvsup18.FreeBSD.org](http://cvsup18.FreeBSD.org)

아르헨티나

[cvsup.ar.FreeBSD.org](http://cvsup.ar.FreeBSD.org)

호주

[cvsup.au.FreeBSD.org](http://cvsup.au.FreeBSD.org)

[cvsup2.au.FreeBSD.org](http://cvsup2.au.FreeBSD.org)

---

[cvsup3.au.FreeBSD.org](http://cvsup3.au.FreeBSD.org)

[cvsup4.au.FreeBSD.org](http://cvsup4.au.FreeBSD.org)

[cvsup5.au.FreeBSD.org](http://cvsup5.au.FreeBSD.org)

[cvsup6.au.FreeBSD.org](http://cvsup6.au.FreeBSD.org)

[cvsup7.au.FreeBSD.org](http://cvsup7.au.FreeBSD.org)

오스트리아

[cvsup.at.FreeBSD.org](http://cvsup.at.FreeBSD.org)

[cvsup2.at.FreeBSD.org](http://cvsup2.at.FreeBSD.org)

브라질

[cvsup.br.FreeBSD.org](http://cvsup.br.FreeBSD.org)

[cvsup2.br.FreeBSD.org](http://cvsup2.br.FreeBSD.org)

[cvsup3.br.FreeBSD.org](http://cvsup3.br.FreeBSD.org)

[cvsup4.br.FreeBSD.org](http://cvsup4.br.FreeBSD.org)

[cvsup5.br.FreeBSD.org](http://cvsup5.br.FreeBSD.org)

캐나다

[cvsup.ca.FreeBSD.org](http://cvsup.ca.FreeBSD.org)

중국

[cvsup.cn.FreeBSD.org](http://cvsup.cn.FreeBSD.org)

[cvsup2.cn.FreeBSD.org](http://cvsup2.cn.FreeBSD.org)

[cvsup3.cn.FreeBSD.org](http://cvsup3.cn.FreeBSD.org)

[cvsup4.cn.FreeBSD.org](http://cvsup4.cn.FreeBSD.org)

---

cvsup5.cn.FreeBSD.org

코스타리카

cvsup1.cr.FreeBSD.org

체코 공화국

cvsup.cz.FreeBSD.org

덴마크

cvsup.dk.FreeBSD.org

cvsup2.dk.FreeBSD.org

에스토니아

cvsup.ee.FreeBSD.org

핀란드

cvsup.fi.FreeBSD.org

cvsup2.fi.FreeBSD.org

프랑스

cvsup.fr.FreeBSD.org

cvsup2.fr.FreeBSD.org

cvsup3.fr.FreeBSD.org

cvsup4.fr.FreeBSD.org

cvsup5.fr.FreeBSD.org

cvsup8.fr.FreeBSD.org

독일



---

[cvsup.de.FreeBSD.org](http://cvsup.de.FreeBSD.org)

[cvsup2.de.FreeBSD.org](http://cvsup2.de.FreeBSD.org)

[cvsup3.de.FreeBSD.org](http://cvsup3.de.FreeBSD.org)

[cvsup4.de.FreeBSD.org](http://cvsup4.de.FreeBSD.org)

[cvsup5.de.FreeBSD.org](http://cvsup5.de.FreeBSD.org)

[cvsup6.de.FreeBSD.org](http://cvsup6.de.FreeBSD.org)

[cvsup7.de.FreeBSD.org](http://cvsup7.de.FreeBSD.org)

[cvsup8.de.FreeBSD.org](http://cvsup8.de.FreeBSD.org)

그리스

[cvsup.gr.FreeBSD.org](http://cvsup.gr.FreeBSD.org)

[cvsup2.gr.FreeBSD.org](http://cvsup2.gr.FreeBSD.org)

헝가리

[cvsup.hu.FreeBSD.org](http://cvsup.hu.FreeBSD.org)

아이슬란드

[cvsup.is.FreeBSD.org](http://cvsup.is.FreeBSD.org)

인도네시아

[cvsup.id.FreeBSD.org](http://cvsup.id.FreeBSD.org)

아일랜드

[cvsup.ie.FreeBSD.org](http://cvsup.ie.FreeBSD.org)

이탈리아

[cvsup.it.FreeBSD.org](http://cvsup.it.FreeBSD.org)

---

---

일본

cvsup.jp.FreeBSD.org  
cvsup2.jp.FreeBSD.org  
cvsup3.jp.FreeBSD.org  
cvsup4.jp.FreeBSD.org  
cvsup5.jp.FreeBSD.org  
cvsup6.jp.FreeBSD.org

한국

cvsup.kr.FreeBSD.org  
cvsup2.kr.FreeBSD.org  
cvsup3.kr.FreeBSD.org

쿠웨이트

cvsup1.kw.FreeBSD.org

크르기스스탄

cvsup.kg.FreeBSD.org

라티비아

cvsup.lv.FreeBSD.org

리투아니아

cvsup.lt.FreeBSD.org  
cvsup2.lt.FreeBSD.org  
cvsup3.lt.FreeBSD.org

---

네덜란드

[cvsup.nl.FreeBSD.org](http://cvsup.nl.FreeBSD.org)  
[cvsup2.nl.FreeBSD.org](http://cvsup2.nl.FreeBSD.org)  
[cvsup3.nl.FreeBSD.org](http://cvsup3.nl.FreeBSD.org)  
[cvsup4.nl.FreeBSD.org](http://cvsup4.nl.FreeBSD.org)  
[cvsup5.nl.FreeBSD.org](http://cvsup5.nl.FreeBSD.org)

뉴질랜드

[cvsup.nz.FreeBSD.org](http://cvsup.nz.FreeBSD.org)

노르웨이

[cvsup.no.FreeBSD.org](http://cvsup.no.FreeBSD.org)

필리핀

[cvsup1.ph.FreeBSD.org](http://cvsup1.ph.FreeBSD.org)

폴란드

[cvsup.pl.FreeBSD.org](http://cvsup.pl.FreeBSD.org)  
[cvsup2.pl.FreeBSD.org](http://cvsup2.pl.FreeBSD.org)  
[cvsup3.pl.FreeBSD.org](http://cvsup3.pl.FreeBSD.org)

포르투갈

[cvsup.pt.FreeBSD.org](http://cvsup.pt.FreeBSD.org)  
[cvsup2.pt.FreeBSD.org](http://cvsup2.pt.FreeBSD.org)  
[cvsup3.pt.FreeBSD.org](http://cvsup3.pt.FreeBSD.org)

루마니아

---

cvsup.ro.FreeBSD.org

cvsup1.ro.FreeBSD.org

cvsup2.ro.FreeBSD.org

cvsup3.ro.FreeBSD.org

러시아

cvsup.ru.FreeBSD.org

cvsup2.ru.FreeBSD.org

cvsup3.ru.FreeBSD.org

cvsup4.ru.FreeBSD.org

cvsup5.ru.FreeBSD.org

cvsup6.ru.FreeBSD.org

산 마리노

cvsup.sm.FreeBSD.org

싱가폴

cvsup.sg.FreeBSD.org

슬로바키아 공화국

cvsup.sk.FreeBSD.org

cvsup2.sk.FreeBSD.org

슬로베니아

cvsup.si.FreeBSD.org

cvsup2.si.FreeBSD.org

---

남아프리카

[cvsup.za.FreeBSD.org](http://cvsup.za.FreeBSD.org)

[cvsup2.za.FreeBSD.org](http://cvsup2.za.FreeBSD.org)

스페인

[cvsup.es.FreeBSD.org](http://cvsup.es.FreeBSD.org)

[cvsup2.es.FreeBSD.org](http://cvsup2.es.FreeBSD.org)

[cvsup3.es.FreeBSD.org](http://cvsup3.es.FreeBSD.org)

스웨덴

[cvsup.se.FreeBSD.org](http://cvsup.se.FreeBSD.org)

[cvsup3.se.FreeBSD.org](http://cvsup3.se.FreeBSD.org)

스위스

[cvsup.ch.FreeBSD.org](http://cvsup.ch.FreeBSD.org)

대만

[cvsup.tw.FreeBSD.org](http://cvsup.tw.FreeBSD.org)

[cvsup3.tw.FreeBSD.org](http://cvsup3.tw.FreeBSD.org)

[cvsup4.tw.FreeBSD.org](http://cvsup4.tw.FreeBSD.org)

[cvsup5.tw.FreeBSD.org](http://cvsup5.tw.FreeBSD.org)

[cvsup6.tw.FreeBSD.org](http://cvsup6.tw.FreeBSD.org)

[cvsup7.tw.FreeBSD.org](http://cvsup7.tw.FreeBSD.org)

[cvsup8.tw.FreeBSD.org](http://cvsup8.tw.FreeBSD.org)

[cvsup9.tw.FreeBSD.org](http://cvsup9.tw.FreeBSD.org)

---

[cvsup10.tw.FreeBSD.org](http://cvsup10.tw.FreeBSD.org)

[cvsup11.tw.FreeBSD.org](http://cvsup11.tw.FreeBSD.org)

[cvsup12.tw.FreeBSD.org](http://cvsup12.tw.FreeBSD.org)

[cvsup13.tw.FreeBSD.org](http://cvsup13.tw.FreeBSD.org)

태국

[cvsup.th.FreeBSD.org](http://cvsup.th.FreeBSD.org)

터키

[cvsup.tr.FreeBSD.org](http://cvsup.tr.FreeBSD.org)

우크라이나

[cvsup2.ua.FreeBSD.org](http://cvsup2.ua.FreeBSD.org)

[cvsup3.ua.FreeBSD.org](http://cvsup3.ua.FreeBSD.org)

[cvsup4.ua.FreeBSD.org](http://cvsup4.ua.FreeBSD.org)

[cvsup5.ua.FreeBSD.org](http://cvsup5.ua.FreeBSD.org)

[cvsup6.ua.FreeBSD.org](http://cvsup6.ua.FreeBSD.org)

[cvsup7.ua.FreeBSD.org](http://cvsup7.ua.FreeBSD.org)

영국

[cvsup.uk.FreeBSD.org](http://cvsup.uk.FreeBSD.org)

[cvsup2.uk.FreeBSD.org](http://cvsup2.uk.FreeBSD.org)

[cvsup3.uk.FreeBSD.org](http://cvsup3.uk.FreeBSD.org)

[cvsup4.uk.FreeBSD.org](http://cvsup4.uk.FreeBSD.org)

미국

---

---

[cvsup1.us.FreeBSD.org](http://cvsup1.us.FreeBSD.org)

[cvsup2.us.FreeBSD.org](http://cvsup2.us.FreeBSD.org)

[cvsup3.us.FreeBSD.org](http://cvsup3.us.FreeBSD.org)

[cvsup4.us.FreeBSD.org](http://cvsup4.us.FreeBSD.org)

[cvsup5.us.FreeBSD.org](http://cvsup5.us.FreeBSD.org)

[cvsup6.us.FreeBSD.org](http://cvsup6.us.FreeBSD.org)

[cvsup7.us.FreeBSD.org](http://cvsup7.us.FreeBSD.org)

[cvsup8.us.FreeBSD.org](http://cvsup8.us.FreeBSD.org)

[cvsup9.us.FreeBSD.org](http://cvsup9.us.FreeBSD.org)

[cvsup10.us.FreeBSD.org](http://cvsup10.us.FreeBSD.org)

[cvsup11.us.FreeBSD.org](http://cvsup11.us.FreeBSD.org)

[cvsup12.us.FreeBSD.org](http://cvsup12.us.FreeBSD.org)

[cvsup13.us.FreeBSD.org](http://cvsup13.us.FreeBSD.org)

[cvsup14.us.FreeBSD.org](http://cvsup14.us.FreeBSD.org)

[cvsup15.us.FreeBSD.org](http://cvsup15.us.FreeBSD.org)

[cvsup16.us.FreeBSD.org](http://cvsup16.us.FreeBSD.org)

[cvsup18.us.FreeBSD.org](http://cvsup18.us.FreeBSD.org)

---

## A.6 CVS Tags

`cv`s 에서 소스를 받거나 업데이트 할 때 수정(revision) 태그를(날짜 참조) `CVSup` 에 지정해야 된다.

수정 태그는 FreeBSD 개발의 특정라인이나 특정 시간을 의미한다. 첫 번째 종류는 “분기(branch) 태그”라고 하고, 두 번째는 “릴리즈 태그”라고 한다.

### A.6.1 분기 태그

`HEAD` 를(항상 유효한 태그) 제외한 이 모든 것들은 `src/` 트리에만 적용할 수 있다. `ports/`, `doc/`와 `www/` 트리는 분기 되지 않는다.

#### HEAD

메인 라인의 심볼릭 이름 또는 `FreeBSD-CURRENT`. 수정 태그가 지정되지 않았을 때 기본적으로 적용된다.

`CVSup` 에서 이 태그는 `.`로 표시된다(구두점이 아니고 문자상의 `.` 문자다).

**Note:** CVS 에서 이것은 수정 태그를 지정하지 않으면 기본적으로 적용된다. 의도적이지 않다면, `STABLE` 머신에서 `CURRENT` 소스를 체크하거나 업데이트 하는 것은 좋지 않다.

#### RELENG\_5\_2

`FreeBSD-5.2` 와 `FreeBSD-5.2.1` 릴리즈 분기는 보안 권고와 중요한 패치에만 사용된다.

#### RELENG\_5\_1

`FreeBSD-5.1` 릴리즈 분기는 보안 권고와 중요한 패치에만 사용된다.

#### RELENG\_5\_0

`FreeBSD-5.0` 릴리즈 분기는 보안 권고와 중요한 패치에만 사용된다.



---

#### RELENG\_4

FreeBSD-STABLE 로도 알려져 있는 FreeBSD-4.X 개발 라인.

#### RELENG\_4\_10

FreeBSD-4.10 릴리즈 분기는 보안 권고와 중요한 패치에만 사용된다.

#### RELENG\_4\_8

FreeBSD-4.8 릴리즈 분기는 보안 권고와 중요한 패치에만 사용된다.

#### RELENG\_4\_7

FreeBSD-4.7 릴리즈 분기는 보안 권고와 중요한 패치에만 사용된다.

#### RELENG\_4\_6

FreeBSD-4.6 과 FreeBSD-4.6.2 릴리즈 분기는 보안 권고와 중요한 패치에만 사용된다.

#### RELENG\_4\_5

FreeBSD-4.5 릴리즈 분기는 보안 권고와 중요한 패치에만 사용된다.

#### RELENG\_4\_4

FreeBSD-4.4 릴리즈 분기는 보안 권고와 중요한 패치에만 사용된다.

#### RELENG\_4\_3

FreeBSD-4.3 릴리즈 분기는 보안 권고와 중요한 패치에만 사용된다.

#### RELENG\_3

3.X-STABLE 로도 알려져 있는 FreeBSD-3.X 개발 라인

#### RELENG\_2\_2

2.2-STABLE 로도 알려져 있는 FreeBSD-2.2.X 개발 라인. 이 분기는 거의 사용하지 않는다.

## A.6.2 릴리즈 태그

---

이들 태그는 특정 버전의 FreeBSD 가 릴리즈 될 때, FreeBSD src/ 트리와(그리고 ports/, doc/ 와 www/ 트리) 일치한다.

RELENG\_4\_10\_0\_RELEASE

FreeBSD 4.10

RELENG\_5\_2\_1\_RELEASE

FreeBSD 5.2.1

RELENG\_5\_2\_0\_RELEASE

FreeBSD 5.2

RELENG\_4\_9\_0\_RELEASE

FreeBSD 4.9

RELENG\_5\_1\_0\_RELEASE

FreeBSD 5.1

RELENG\_4\_8\_0\_RELEASE

FreeBSD 4.8

RELENG\_5\_0\_0\_RELEASE

FreeBSD 5.0

RELENG\_4\_7\_0\_RELEASE

FreeBSD 4.7

RELENG\_4\_6\_2\_RELEASE

FreeBSD 4.6.2

---

RELENG\_4\_6\_1\_RELEASE

FreeBSD 4.6.1

RELENG\_4\_6\_0\_RELEASE

FreeBSD 4.6

RELENG\_4\_5\_0\_RELEASE

FreeBSD 4.5

RELENG\_4\_4\_0\_RELEASE

FreeBSD 4.4

RELENG\_4\_3\_0\_RELEASE

FreeBSD 4.3

RELENG\_4\_2\_0\_RELEASE

FreeBSD 4.2

RELENG\_4\_1\_1\_RELEASE

FreeBSD 4.1.1

RELENG\_4\_1\_0\_RELEASE

FreeBSD 4.1

RELENG\_4\_0\_0\_RELEASE

FreeBSD 4.0

RELENG\_3\_5\_0\_RELEASE

FreeBSD-3.5

RELENG\_3\_4\_0\_RELEASE

---

FreeBSD-3.4

RELENG\_3\_3\_0\_RELEASE

FreeBSD-3.3

RELENG\_3\_2\_0\_RELEASE

FreeBSD-3.2

RELENG\_3\_1\_0\_RELEASE

FreeBSD-3.1

RELENG\_3\_0\_0\_RELEASE

FreeBSD-3.0

RELENG\_2\_2\_8\_RELEASE

FreeBSD-2.2.8

RELENG\_2\_2\_7\_RELEASE

FreeBSD-2.2.7

RELENG\_2\_2\_6\_RELEASE

FreeBSD-2.2.6

RELENG\_2\_2\_5\_RELEASE

FreeBSD-2.2.5

RELENG\_2\_2\_2\_RELEASE

FreeBSD-2.2.2

RELENG\_2\_2\_1\_RELEASE

FreeBSD-2.2.1

---

---

RELENG\_2\_2\_0\_RELEASE

FreeBSD-2.2.0

## A.7 AFS 사이트

FreeBSD AFS 서버는 다음 사이트에서 운용 중 이다:

스웨덴

파일 경로는: /afs/stacken.kth.se/ftp/pub/FreeBSD/

|                 |                                      |
|-----------------|--------------------------------------|
| stacken.kth.se  | # Stacken Computer Club, KTH, Sweden |
| 130.237.234.43  | #hot.stacken.kth.se                  |
| 130.237.237.230 | #fishburger.stacken.kth.se           |
| 130.237.234.3   | #milko.stacken.kth.se                |

관리자 <ftp@stacken.kth.se>

## A.8 rsync 사이트

다음 사이트는 rsync 프로토콜을 사용하여 FreeBSD 를 사용할 수 있게 한다. **rsync** 유틸리티는 rcp(1) 명령과 거의 비슷하게 동작하지만, 더 많은 옵션을 가지고 있고 두 파일 세트의 차이만 전송하는 rsync 원격 업데이트 프로토콜을 사용하기 때문에, 엄청난 속도로 네트워크를 통해 동기화 한다. 이것은 여러분이 FreeBSD FTP 서버나 CVS 저장소 미러 사이트를 운용한다면 굉장히 유용하다. **rsync** 는 많은 운영체제에서 사용할 수 있고, FreeBSD 에서는 [net/rsync](#) 포트나 패키지를 사용한다.

체코 공화국

rsync://ftp.cz.FreeBSD.org/

---

사용할 수 있는 컬렉션:

ftp: FreeBSD FTP 서버 일부분 미러  
FreeBSD: FreeBSD FTP 서버 전체 미러

독일

rsync://grappa.unix-ag.uni-kl.de/

사용할 수 있는 컬렉션:

freebsd-cvs: FreeBSD CVS 저장소 전체

이 머신은 NetBSD 와 OpenBSD 프로젝트 CVS 저장소도 미러 한다.

네덜란드

rsync://ftp.nl.FreeBSD.org/

사용할 수 있는 컬렉션:

vol/3/freebsd-core: FreeBSD FTP 서버 전체 미러

영국

rsync://rsync.mirror.ac.uk/

사용할 수 있는 컬렉션:

ftp.FreeBSD.org: FreeBSD FTP 서버 전체 미러

미국

---

rsync://ftp-master.FreeBSD.org/

이 서버는 FreeBSD 메인 미러 사이트만 사용할 것이다.

사용할 수 있는 컬렉션:

FreeBSD: FreeBSD FTP 의 마스터 저장소

acl: FreeBSD 마스터 ACL 리스트

rsync://ftp13.FreeBSD.org/

사용할 수 있는 컬렉션:

FreeBSD: FreeBSD FTP 서버의 완전한 미러 사이트

## 부록 B. 관련 서적

매뉴얼 페이지가 FreeBSD 운영체제의 부분별 레퍼런스를 제공하지만, 전체 운영체제를 사용할 수 있도록, 이들 매뉴얼 페이지를 어떻게 사용해야 되는지 설명하지 않는다. 그래서 유닉스 시스템 관리자에 관한 쓸만한 책과 유저 매뉴얼이 없다.

### B.1 FreeBSD 와 연관된 책과 잡지

*국제적인 책과 잡지:*

[Using FreeBSD](http://jdli.tw.freebsd.org/publication/book/freebsd2/index.htm) (<http://jdli.tw.freebsd.org/publication/book/freebsd2/index.htm>)

(중국어).

---

FreeBSD for PC 98'ers (일본어), published by SHUWA System Co, LTD. ISBN 4-87966-468-5 C3055 P2900E.

FreeBSD (일본어), published by CUTT. ISBN 4-906391-22-2 C3055 P2400E.

[Complete Introduction to FreeBSD](#)

(<http://www.shoeisha.com/lib/redirect.asp?url=http://www.seshop.com/detail.asp?bid=650>) (일본어), published by [Shoeisha Co., Ltd](#)(<http://www.shoeisha.co.jp/>). ISBN 4-88135-473-6 P3600E.

[Personal UNIX Starter Kit FreeBSD](#)

(<http://www.ascii.co.jp/pb/book1/shinkan/detail/1322785.html>) (일본어), published by [ASCII](#) (<http://www.ascii.co.jp/>). ISBN 4-7561-1733-3 P3000E.

FreeBSD Handbook (일본어 번역), published by [ASCII](#). ISBN 4-7561-1580-2 P3800E.

FreeBSD mit Methode (in German), published by [Computer und Literatur Verlag](#)/Vertrieb Hanser (<http://www.cul.de/>), 1998. ISBN 3-932311-31-0.

[FreeBSD 4 - Installieren, Konfigurieren, Administrieren](#)

(<http://www.cul.de/freebsd.html>) (독일어), published by [Computer und Literatur Verlag](#) (<http://www.cul.de/>), 2001. ISBN 3-932311-88-4.



---

[FreeBSD 5 – Installieren, Konfigurieren, Administrieren](http://www.cul.de/) (<http://www.cul.de/> )  
(독일어), published by [Computer und Literatur Verlag](http://www.cul.de/), 2003. ISBN 3-936546-06-1.

[FreeBSD de Luxe](http://www.mitp.de/vmi/mitp/detail/pWert/1343/) (<http://www.mitp.de/vmi/mitp/detail/pWert/1343/> ) (독일어),  
published by [Verlag Moderne Industrie](http://www.mitp.de/) (<http://www.mitp.de/>), 2003. ISBN 3-8266-1343-0.

[FreeBSD Install and Utilization Manual](http://www.pc.mycom.co.jp/FreeBSD/install-manual.html)  
(<http://www.pc.mycom.co.jp/FreeBSD/install-manual.html> ) (일본어),  
published by [Mainichi Communications Inc.](http://www.pc.mycom.co.jp/) (<http://www.pc.mycom.co.jp/>).

Onno W Purbo, Dodi Maryanto, Syahrial Hubbany, Widjil Widodo [Building Internet Server with FreeBSD](http://maxwell.itb.ac.id/) (<http://maxwell.itb.ac.id/> ) (인도네시아 어),  
published by [Elex Media Komputindo](http://www.elexmedia.co.id/) (<http://www.elexmedia.co.id/>).

영문 책과 잡지:

[Absolute BSD: The Ultimate Guide to FreeBSD](http://www.absolutebsd.com/)  
(<http://www.absolutebsd.com/>), published by [No Starch Press](http://www.nostarch.com/)  
(<http://www.nostarch.com/>), 2002. ISBN: 1886411743

[The Complete FreeBSD](http://www.nostarch.com/) (<http://www.nostarch.com/>), published by  
[O'Reilly](http://www.oreilly.com/) (<http://www.oreilly.com/>), 2003. ISBN: 0596005164

---

[The FreeBSD Corporate Networker's Guide](http://www.freebsd-corp-net-guide.com/) (<http://www.freebsd-corp-net-guide.com/>), published by [Addison-Wesley](http://www.awl.com/aw/) (<http://www.awl.com/aw/>), 2000. ISBN: 0201704811

[FreeBSD: An Open-Source Operating System for Your Personal Computer](http://andrsn.stanford.edu/FreeBSD/introbook/) (<http://andrsn.stanford.edu/FreeBSD/introbook/>), published by The Bit Tree Press, 2001. ISBN: 0971204500

Teach Yourself FreeBSD in 24 Hours, published by [Sams](http://www.sampublishing.com/), 2002. ISBN: 0672324245

FreeBSD unleashed, published by [Sams](http://www.sampublishing.com/) (<http://www.sampublishing.com/>), 2002. ISBN: 0672324563

FreeBSD: The Complete Reference, published by [McGrawHill](http://books.mcgraw-hill.com/) (<http://books.mcgraw-hill.com/>), 2003. ISBN: 0072224096

## B.2 사용자 가이드

Computer Systems Research Group, UC Berkeley. *4.4BSD User's Reference Manual*. O'Reilly & Associates, Inc., 1994. ISBN 1-56592-075-9

---

Computer Systems Research Group, UC Berkeley. *4.4BSD User's Supplementary Documents*. O'Reilly & Associates, Inc., 1994. ISBN 1-56592-076-7

*UNIX in a Nutshell*. O'Reilly & Associates, Inc., 1990. ISBN 093717520X

Mui, Linda. *What You Need To Know When You Can't Find Your UNIX System Administrator*. O'Reilly & Associates, Inc., 1995. ISBN 1-56592-104-6

[Ohio State University](http://www-wks.acs.ohio-state.edu/) (<http://www-wks.acs.ohio-state.edu/>) has written a [UNIX Introductory Course](http://www-wks.acs.ohio-state.edu/unix_course/unix.html) ([http://www-wks.acs.ohio-state.edu/unix\\_course/unix.html](http://www-wks.acs.ohio-state.edu/unix_course/unix.html)) which is available online in HTML and PostScript format.

An Italian [translation](http://www.freebsd.org/doc/it_IT.ISO8859-15/books/unix-introduction/index.html) ([http://www.freebsd.org/doc/it\\_IT.ISO8859-15/books/unix-introduction/index.html](http://www.freebsd.org/doc/it_IT.ISO8859-15/books/unix-introduction/index.html)) of this document is available as part of the FreeBSD Italian Documentation Project.

[Jpman Project, Japan FreeBSD Users Group](http://www.jp.freebsd.org/) (<http://www.jp.freebsd.org/>). [FreeBSD User's Reference Manual](http://www.pc.mycom.co.jp/FreeBSD/urm.html) (<http://www.pc.mycom.co.jp/FreeBSD/urm.html>) (Japanese translation). [Mainichi Communications Inc.](http://www.pc.mycom.co.jp/) (<http://www.pc.mycom.co.jp/>), 1998. ISBN4-8399-0088-4 P3800E.

---

[Edinburgh University](http://www.ed.ac.uk/) (<http://www.ed.ac.uk/>) has written an [Online Guide](http://unixhelp.ed.ac.uk/) (<http://unixhelp.ed.ac.uk/>) for newcomers to the UNIX environment.

## B.3 관리자 가이드

Albitz, Paul and Liu, Cricket. *DNS and BIND*, 4th Ed. O'Reilly & Associates, Inc., 2001. ISBN 1-59600-158-4

Computer Systems Research Group, UC Berkeley. *4.4BSD System Manager's Manual*. O'Reilly & Associates, Inc., 1994. ISBN 1-56592-080-5

Costales, Brian, et al. *Sendmail*, 2nd Ed. O'Reilly & Associates, Inc., 1997. ISBN 1-56592-222-0

Frisch, Aileen. *Essential System Administration*, 2nd Ed. O'Reilly & Associates, Inc., 1995. ISBN 1-56592-127-5

Hunt, Craig. *TCP/IP Network Administration*, 2nd Ed. O'Reilly & Associates, Inc., 1997. ISBN 1-56592-322-7

Nemeth, Evi. *UNIX System Administration Handbook*. 3rd Ed. Prentice Hall, 2000. ISBN 0-13-020601-6

---

Stern, Hal *Managing NFS and NIS* O'Reilly & Associates, Inc.,  
1991. ISBN 0-937175-75-7

[Jpman Project, Japan FreeBSD Users Group. FreeBSD System Administrator's Manual](#) (Japanese translation). [Mainichi Communications Inc.](#), 1998. ISBN4-8399-0109-0 P3300E.

Dreyfus, Emmanuel. [Cahiers de l'Admin: BSD](#)  
(<http://www.editions-eyrolles.com/php.informatique/Ouvrages/9782212112443.php3>  
) (in French), Eyrolles, 2003. ISBN 2-212-11244-0

## B.4 프로그래머 가이드

Asente, Paul, Converse, Diana, and Swick, Ralph. *X Window System Toolkit*. Digital Press, 1998. ISBN 1-55558-178-1

Computer Systems Research Group, UC Berkeley. *4.4BSD Programmer's Reference Manual*. O'Reilly & Associates, Inc., 1994. ISBN 1-56592-078-3

Computer Systems Research Group, UC Berkeley. *4.4BSD Programmer's Supplementary Documents*. O'Reilly & Associates, Inc., 1994. ISBN 1-56592-079-1

---

Harbison, Samuel P. and Steele, Guy L. Jr. *C: A Reference Manual*.  
4rd ed. Prentice Hall, 1995. ISBN 0-13-326224-3

Kernighan, Brian and Dennis M. Ritchie. *The C Programming  
Language..* PTR Prentice Hall, 1988. ISBN 0-13-110362-9

Lehey, Greg. *Porting UNIX Software*. O'Reilly & Associates, Inc.,  
1995. ISBN 1-56592-126-7

Plauger, P. J. *The Standard C Library*. Prentice Hall, 1992. ISBN  
0-13-131509-9

Spinellis, Diomidis. [Code Reading: The Open Source Perspective](http://www.spinellis.gr/codereading/)  
(<http://www.spinellis.gr/codereading/>). Addison-Wesley, 2003.  
ISBN 0-201-79940-5

Stevens, W. Richard. *Advanced Programming in the UNIX  
Environment*. Reading, Mass. : Addison-Wesley, 1992. ISBN 0-  
201-56317-7

Stevens, W. Richard. *UNIX Network Programming*. 2nd Ed, PTR  
Prentice Hall, 1998. ISBN 0-13-490012-X

---

Wells, Bill. ``Writing Serial Drivers for UNIX''. *Dr. Dobb's Journal*.  
19(15), December 1994. pp68-71, 97-99.

## B.5 운영 체제 내부

Andleigh, Prabhat K. *UNIX System Architecture*. Prentice-Hall, Inc.,  
1990. ISBN 0-13-949843-5

Jolitz, William. ``Porting UNIX to the 386''. *Dr. Dobb's Journal*.  
January 1991-July 1992.

Leffler, Samuel J., Marshall Kirk McKusick, Michael J Karels and  
John Quarterman *The Design and Implementation of the  
4.3BSD UNIX Operating System*. Reading, Mass. : Addison-  
Wesley, 1989. ISBN 0-201-06196-1

Leffler, Samuel J., Marshall Kirk McKusick, *The Design and  
Implementation of the 4.3BSD UNIX Operating System: Answer  
Book*. Reading, Mass. : Addison-Wesley, 1991. ISBN 0-201-  
54629-9

McKusick, Marshall Kirk, Keith Bostic, Michael J Karels, and John  
Quarterman. *The Design and Implementation of the 4.4BSD  
Operating System*. Reading, Mass. : Addison-Wesley, 1996.  
ISBN 0-201-54979-4

---

(Chapter 2 of this book is available [online](http://www.freebsd.org/doc/en_US.ISO8859-1/books/design-44bsd/book.html) ([http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/design-44bsd/book.html](http://www.freebsd.org/doc/en_US.ISO8859-1/books/design-44bsd/book.html)) as part of the FreeBSD Documentation Project, and chapter 9 [here](#).)

Stevens, W. Richard. *TCP/IP Illustrated, Volume 1: The Protocols*. Reading, Mass. : Addison-Wesley, 1996. ISBN 0-201-63346-9

Schimmel, Curt. *Unix Systems for Modern Architectures*. Reading, Mass. : Addison-Wesley, 1994. ISBN 0-201-63338-8

Stevens, W. Richard. *TCP/IP Illustrated, Volume 3: TCP for Transactions, HTTP, NNTP and the UNIX Domain Protocols*. Reading, Mass. : Addison-Wesley, 1996. ISBN 0-201-63495-3

Vahalia, Uresh. *UNIX Internals -- The New Frontiers*. Prentice Hall, 1996. ISBN 0-13-101908-2

Wright, Gary R. and W. Richard Stevens. *TCP/IP Illustrated, Volume 2: The Implementation*. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-63354-X

## B.6 보안 레퍼런스



---

Cheswick, William R. and Steven M. Bellovin. *Firewalls and Internet Security: Repelling the Wily Hacker*. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-63357-4

Garfinkel, Simson and Gene Spafford. *Practical UNIX & Internet Security*. 2nd Ed. O'Reilly & Associates, Inc., 1996. ISBN 1-56592-148-8

Garfinkel, Simson. *PGP Pretty Good Privacy* O'Reilly & Associates, Inc., 1995. ISBN 1-56592-098-8

## B.7 하드웨어 레퍼런스

Anderson, Don and Tom Shanley. *Pentium Processor System Architecture*. 2nd Ed. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-40992-5

Ferraro, Richard F. *Programmer's Guide to the EGA, VGA, and Super VGA Cards*. 3rd ed. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-62490-7

Intel Corporation publishes documentation on their CPUs, chipsets and standards on their [developer web site](http://developer.intel.com/) (<http://developer.intel.com/>), usually as PDF files.

---

Shanley, Tom. *80486 System Architecture*. 3rd ed. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-40994-1

Shanley, Tom. *ISA System Architecture*. 3rd ed. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-40996-8

Shanley, Tom. *PCI System Architecture*. 4th ed. Reading, Mass. : Addison-Wesley, 1999. ISBN 0-201-30974-2

Van Gilluwe, Frank. *The Undocumented PC*, 2nd Ed. Reading, Mass: Addison-Wesley Pub. Co., 1996. ISBN 0-201-47950-8

Messmer, Hans-Peter. *The Indispensable PC Hardware Book*, 4th Ed. Reading, Mass: Addison-Wesley Pub. Co., 2002. ISBN 0-201-59616-4

## B.8 유닉스 역사

Lion, John *Lion's Commentary on UNIX, 6th Ed. With Source Code*. ITP Media Group, 1996. ISBN 1573980137

Raymond, Eric S. *The New Hacker's Dictionary, 3rd edition*. MIT Press, 1996. ISBN 0-262-68092-0. Also known as the [Jargon File](http://www.catb.org/~esr/jargon/html/index.html) (<http://www.catb.org/~esr/jargon/html/index.html>)

---

Salus, Peter H. *A quarter century of UNIX*. Addison-Wesley Publishing Company, Inc., 1994. ISBN 0-201-54777-5

Simon Garfinkel, Daniel Weise, Steven Strassmann. *The UNIX-HATERS Handbook*. IDG Books Worldwide, Inc., 1994. ISBN 1-56884-203-1

Don Libes, Sandy Ressler *Life with UNIX* -- special edition. Prentice-Hall, Inc., 1989. ISBN 0-13-536657-7

*The BSD family tree.*

<http://www.FreeBSD.org/cgi/cvsweb.cgi/src/share/misc/bsd-family-tree>

(<http://www.freebsd.org/cgi/cvsweb.cgi/src/share/misc/bsd-family-tree>) or </usr/share/misc/bsd-family-tree> on a modern FreeBSD machine.

*The BSD Release Announcements collection.* 1997.

<http://www.de.FreeBSD.org/de/ftp/releases/>

*Networked Computer Science Technical Reports Library.*

<http://www.ncstrl.org/>

*Old BSD releases from the Computer Systems Research group (CSRG).* <http://www.mckusick.com/csrg/>: The 4CD set covers

---

all BSD versions from 1BSD to 4.4BSD and 4.4BSD-Lite2 (but not 2.11BSD, unfortunately). As well, the last disk holds the final sources plus the SCCS files.

## B.9 잡지와 정기 간행물

*The C/C++ Users Journal*. R&D Publications Inc. ISSN 1075-2838

*Sys Admin -- The Journal for UNIX System Administrators* Miller  
Freeman, Inc., ISSN 1061-2688

*freeX -- Das Magazin für Linux - BSD - UNIX* (in German)  
Computer- und Literaturverlag GmbH, ISSN 1436-7033

## 부록 C. 인터넷의 리소스

빠른 속도로 발전하는 FreeBSD는 프린트 미디어를 쓸모 없게 만든다. 항상 그렇지는 않지만 최신의 정보를 유지하는 것은 전자 리소스가 최고다. FreeBSD는 전자 메일과 커뮤니티에 접근하는 가장 효과적인 유즈넷 뉴스를 통한, 봉사자들의 노력의 결과다. 또한 유저 커뮤니티는 "기술적인 지원 부서" 정도가 되기도 한다.

FreeBSD 유저 커뮤니티에 연락하는 가장 중요한 포인트는 아래에 요약되어 있다. 여기서 설명하지 않는 다른 리소스를 알고 있다면 FreeBSD 문서 프로젝트 메일링 리스트에 (<http://lists.freebsd.org/mailman/listinfo/freebsd-doc>) 알려주기 바란다.

---

## C.1 메일링 리스트

많은 FreeBSD 개발자들이 유즈넷을 읽지만, comp.unix.bsd.freebsd.\* 그룹 중 한곳에 질문을 올렸다면 제 시간에 읽는다고 보장하지 못한다. 적절한 메일링 리스트에 질문을 올리면 우리와 FreeBSD 애호가들로부터 빠른 응답을 받을 수 있다.

다양한 리스트 철택어는 이 문서 아래에 있다. *리스트에 등록하거나 메일을 보내기 전에 철택어를 읽어보기 바란다.* 불필요한 사항을 배제하고 적절한 정보만 유지하기 위해 사용하는 룰에 의해, 대부분의 리스트 등록자들은 하루에 수백 통의 FreeBSD 관련 메시지를 받게 된다.

저장소는 모든 메일링 리스트를 가지고 있고 FreeBSD 웹 서버를 통해 검색할 수 (<http://www.freebsd.org/search/index.html>) 있다. 키워드 검색 저장소는 종종 질문된 답변을 찾는 훌륭한 방법을 제공하기 때문에 질문을 올리기 전에 확인해 본다.

### C.1.1 리스트 요약

*일반적인 리스트:* 다음은 누구나 등록할 수 있는 일반적인 리스트다:

| 리스트                | URL                                                                                                                                     | 목적                                 |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------|------------------------------------|
| cvs-all            | <a href="http://lists.FreeBSD.org/mailman/listinfo/cvs-all">http://lists.FreeBSD.org/mailman/listinfo/cvs-all</a>                       | FreeBSD 소스트리 변경                    |
| freebsd-advocacy   | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-advocacy">http://lists.FreeBSD.org/mailman/listinfo/freebsd-advocacy</a>     | FreeBSD 전도                         |
| freebsd-announce   | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-announce">http://lists.FreeBSD.org/mailman/listinfo/freebsd-announce</a>     | 중요한 이벤트와 프로젝트 이벤트                  |
| freebsd-arch       | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-arch">http://lists.FreeBSD.org/mailman/listinfo/freebsd-arch</a>             | 아키텍처와 디자인 토론                       |
| freebsd-bugbusters | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-bugbusters">http://lists.FreeBSD.org/mailman/listinfo/freebsd-bugbusters</a> | FreeBSD 문제 리포트 데이터베이스와 관련 툴에 대한 토론 |
| freebsd-bugs       | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-bugs">http://lists.FreeBSD.org/mailman/listinfo/freebsd-bugs</a>             | 버그 리포트                             |

|                                |                                                                                                                                                                 |                                |
|--------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------|
| freebsd-chat                   | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-chat">http://lists.FreeBSD.org/mailman/listinfo/freebsd-chat</a>                                     | 기술적이지 않은 FreeBSD 커뮤니티에 관련된 아이템 |
| freebsd-config                 | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-config">http://lists.FreeBSD.org/mailman/listinfo/freebsd-config</a>                                 | FreeBSD 설치와 설정 툴 개발            |
| freebsd-current                | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-current">http://lists.FreeBSD.org/mailman/listinfo/freebsd-current</a>                               | FreeBSD-CURRENT 사용에 관한 토론      |
| freebsd-isp                    | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-isp">http://lists.FreeBSD.org/mailman/listinfo/freebsd-isp</a>                                       | FreeBSD를 사용한 인터넷 서비스 공급자들 이슈   |
| freebsd-jobs                   | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-jobs">http://lists.FreeBSD.org/mailman/listinfo/freebsd-jobs</a>                                     | FreeBSD 고용자와 기회 컨설팅            |
| freebsd-newbies                | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-newbies">http://lists.FreeBSD.org/mailman/listinfo/freebsd-newbies</a>                               | 새로운 FreeBSD 유저 활동과 토론          |
| freebsd-policy                 | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-policy">http://lists.FreeBSD.org/mailman/listinfo/freebsd-policy</a>                                 | FreeBSD 코어 팀 정책 결정. 읽기 전용      |
| freebsd-questions              | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-questions">http://lists.FreeBSD.org/mailman/listinfo/freebsd-questions</a>                           | 유저 질문과 기술적인 지원                 |
| freebsd-security-notifications | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-security-notifications">http://lists.FreeBSD.org/mailman/listinfo/freebsd-security-notifications</a> | 보안 공고                          |
| freebsd-stable                 | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-stable">http://lists.FreeBSD.org/mailman/listinfo/freebsd-stable</a>                                 | FreeBSD-STABLE 사용에 관한 토론       |
| freebsd-test                   | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-test">http://lists.FreeBSD.org/mailman/listinfo/freebsd-test</a>                                     | 실제 리스트가 아닌 메시지를 테스트하는 곳        |

*기술적인 리스트:* 다음 리스트는 기술적인 토론에 대한 것이다. 사용법과 내용에 대한 가이드라인이 있기 때문에, 등록하거나 메일을 보내기 전에 각 리스트를 주의 깊게 읽어 본다.

| 리스트             | URL                                                                                                                           | 목적                       |
|-----------------|-------------------------------------------------------------------------------------------------------------------------------|--------------------------|
| freebsd-acpi    | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-acpi">http://lists.FreeBSD.org/mailman/listinfo/freebsd-acpi</a>   | ACPI 와 전원관리 개발           |
| freebsd-afs     | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-afs">http://lists.FreeBSD.org/mailman/listinfo/freebsd-afs</a>     | AFS 를 FreeBSD 에 포팅       |
| freebsd-aic7xxx | <a href="http://lists.FreeBSD.org/mailman/listinfo/aic7xxx">http://lists.FreeBSD.org/mailman/listinfo/aic7xxx</a>             | Adaptec AIC 7xxx 드라이버 개발 |
| freebsd-alpha   | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-alpha">http://lists.FreeBSD.org/mailman/listinfo/freebsd-alpha</a> | Alpha 에 FreeBSD 를 포팅     |

|                   |                                                                                                                                       |                                           |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------|
|                   | <a href="#">eebsd-alpha</a>                                                                                                           |                                           |
| freebsd-amd64     | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-amd64">http://lists.FreeBSD.org/mailman/listinfo/freebsd-amd64</a>         | FreeBSD 를 AMD64 시스템에 포팅                   |
| freebsd-arm       | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-arm">http://lists.FreeBSD.org/mailman/listinfo/freebsd-arm</a>             | FreeBSD 를 ARM 프로세서에 포팅                    |
| freebsd-atm       | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-atm">http://lists.FreeBSD.org/mailman/listinfo/freebsd-atm</a>             | FreeBSD 로 ATM 네트워크 사용                     |
| freebsd-audit     | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-audit">http://lists.FreeBSD.org/mailman/listinfo/freebsd-audit</a>         | Audit 프로젝트 소스 코드                          |
| freebsd-binup     | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-binup">http://lists.FreeBSD.org/mailman/listinfo/freebsd-binup</a>         | 바이너리 업데이트 시스템 디자인과 개발                     |
| freebsd-cluster   | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-cluster">http://lists.FreeBSD.org/mailman/listinfo/freebsd-cluster</a>     | 클러스터 환경에서 FreeBSD 사용                      |
| freebsd-cvsweb    | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-cvsweb">http://lists.FreeBSD.org/mailman/listinfo/freebsd-cvsweb</a>       | CVSweb 관리                                 |
| freebsd-database  | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-database">http://lists.FreeBSD.org/mailman/listinfo/freebsd-database</a>   | FreeBSD 에서 데이터베이스 사용과 개발 토론               |
| freebsd-doc       | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-doc">http://lists.FreeBSD.org/mailman/listinfo/freebsd-doc</a>             | FreeBSD 관련 문서 생성                          |
| freebsd-emulation | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-emulation">http://lists.FreeBSD.org/mailman/listinfo/freebsd-emulation</a> | 리눅스/DOS 와 같은 다른 시스템 에뮬레이션                 |
| freebsd-firewire  | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-firewire">http://lists.FreeBSD.org/mailman/listinfo/freebsd-firewire</a>   | FreeBSD FireWire (iLink, IEEE 1394) 기술 토론 |
| freebsd-fs        | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-fs">http://lists.FreeBSD.org/mailman/listinfo/freebsd-fs</a>               | 파일 시스템                                    |
| freebsd-geom      | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-geom">http://lists.FreeBSD.org/mailman/listinfo/freebsd-geom</a>           | GEOM 관련 토론                                |
| freebsd-gnome     | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-gnome">http://lists.FreeBSD.org/mailman/listinfo/freebsd-gnome</a>         | <b>GNOME</b> 과 <b>GNOME</b> 어플리케이션 포팅     |
| freebsd-hackers   | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-hackers">http://lists.FreeBSD.org/mailman/listinfo/freebsd-hackers</a>     | 일반적인 기술 토론                                |
| freebsd-hardware  | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-hardware">http://lists.FreeBSD.org/mailman/listinfo/freebsd-hardware</a>   | FreeBSD 에서 지원되는 하드웨어에 대한 일반적인 토론          |
| freebsd-i18n      | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-i18n">http://lists.FreeBSD.org/mailman/listinfo/freebsd-i18n</a>           | FreeBSD 국제화                               |

|                     |                                                                                                                                           |                                                          |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------|
| freebsd-ia32        | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-ia32">http://lists.FreeBSD.org/mailman/listinfo/freebsd-ia32</a>               | IA-32 (Intel® x86) 플랫폼의 FreeBSD                          |
| freebsd-ia64        | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-ia64">http://lists.FreeBSD.org/mailman/listinfo/freebsd-ia64</a>               | FreeBSD 를 인텔 IA64 시스템에 포팅                                |
| freebsd-ipfw        | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-ipfw">http://lists.FreeBSD.org/mailman/listinfo/freebsd-ipfw</a>               | IP 방화벽 코드 재 디자인에 대한 기술적인 토론                              |
| freebsd-isdn        | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-isdn">http://lists.FreeBSD.org/mailman/listinfo/freebsd-isdn</a>               | ISDN 개발자                                                 |
| freebsd-java        | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-java">http://lists.FreeBSD.org/mailman/listinfo/freebsd-java</a>               | Java 개발자와 JDK 를 FreeBSD 로 포팅하는 사람들                       |
| freebsd-kde         | <a href="http://freebsd.kde.org/mailman/listinfo/kde-freebsd">http://freebsd.kde.org/mailman/listinfo/kde-freebsd</a>                     | KDE 와 KDE 어플리케이션 포팅                                      |
| freebsd-lfs         | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-lfs">http://lists.FreeBSD.org/mailman/listinfo/freebsd-lfs</a>                 | LFS 를 FreeBSD 로 포팅                                       |
| freebsd-libh        | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-libh">http://lists.FreeBSD.org/mailman/listinfo/freebsd-libh</a>               | 차세대 설치 및 패키지 시스템                                         |
| freebsd-mips        | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-mips">http://lists.FreeBSD.org/mailman/listinfo/freebsd-mips</a>               | FreeBSD 를 MIPS 에 포팅                                      |
| freebsd-mobile      | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-mobile">http://lists.FreeBSD.org/mailman/listinfo/freebsd-mobile</a>           | 모바일 컴퓨터에 대한 토론                                           |
| freebsd-mozilla     | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-mozilla">http://lists.FreeBSD.org/mailman/listinfo/freebsd-mozilla</a>         | Mozilla 를 FreeBSD 에 포팅                                   |
| freebsd-multimedia  | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-multimedia">http://lists.FreeBSD.org/mailman/listinfo/freebsd-multimedia</a>   | 멀티미디어 어플리케이션                                             |
| freebsd-new-bus     | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-new-bus">http://lists.FreeBSD.org/mailman/listinfo/freebsd-new-bus</a>         | Bus 아키텍처에 대한 기술적인 토론                                     |
| freebsd-net         | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-net">http://lists.FreeBSD.org/mailman/listinfo/freebsd-net</a>                 | 네트워크 토론과 TCP/IP 소스 코드                                    |
| freebsd-performance | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-performance">http://lists.FreeBSD.org/mailman/listinfo/freebsd-performance</a> | 높은 성능/부하를 위한 성능 튜닝 질문                                    |
| freebsd-perl        | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-perl">http://lists.FreeBSD.org/mailman/listinfo/freebsd-perl</a>               | <b>OpenOffice.org</b> 와 <b>StarOffice</b> 를 FreeBSD 에 포팅 |
| freebsd-platforms   | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-platforms">http://lists.FreeBSD.org/mailman/listinfo/freebsd-platforms</a>     | Intel 아키텍처가 아닌 플랫폼에 포팅                                   |
| freebsd-ports       | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-ports">http://lists.FreeBSD.org/mailman/listinfo/freebsd-ports</a>             | 포트 컬렉션에 대한 토론                                            |



|                    |                                                                                                                                         |                                     |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------|
| freebsd-ports-bugs | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-ports-bugs">http://lists.FreeBSD.org/mailman/listinfo/freebsd-ports-bugs</a> | 포트 bugs/PRs 토론                      |
| freebsd-ppc        | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-ppc">http://lists.FreeBSD.org/mailman/listinfo/freebsd-ppc</a>               | FreeBSD 를 PowerPC 에 포팅              |
| freebsd-qa         | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-qa">http://lists.FreeBSD.org/mailman/listinfo/freebsd-qa</a>                 | 보통 릴리즈 중 품질 보장 토론                   |
| freebsd-realtime   | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-realtime">http://lists.FreeBSD.org/mailman/listinfo/freebsd-realtime</a>     | FreeBSD 를 실시간 확장으로 개발               |
| freebsd-scsi       | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-scsi">http://lists.FreeBSD.org/mailman/listinfo/freebsd-scsi</a>             | 스카시 서브시스템                           |
| freebsd-security   | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-security">http://lists.FreeBSD.org/mailman/listinfo/freebsd-security</a>     | FreeBSD 에 영향을 주는 보안 이슈              |
| freebsd-small      | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-small">http://lists.FreeBSD.org/mailman/listinfo/freebsd-small</a>           | 임베디드 어플리케이션에 FreeBSD 사용             |
| freebsd-smp        | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-smp">http://lists.FreeBSD.org/mailman/listinfo/freebsd-smp</a>               | [A]Symmetric MultiProcessing 디자인 토론 |
| freebsd-sparc64    | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-sparc64">http://lists.FreeBSD.org/mailman/listinfo/freebsd-sparc64</a>       | Sparc 기반 시스템에 FreeBSD 포팅            |
| freebsd-standards  | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-standards">http://lists.FreeBSD.org/mailman/listinfo/freebsd-standards</a>   | C99 과 FreeBSD 에서의 POSIX® 표준 스펙팅 일치  |
| freebsd-threads    | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-threads">http://lists.FreeBSD.org/mailman/listinfo/freebsd-threads</a>       | FreeBSD 에서 스레드                      |
| freebsd-testing    | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-testing">http://lists.FreeBSD.org/mailman/listinfo/freebsd-testing</a>       | FreeBSD 성능과 안정성 테스트                 |
| freebsd-tokenring  | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-tokenring">http://lists.FreeBSD.org/mailman/listinfo/freebsd-tokenring</a>   | FreeBSD 에서 토큰 링(Token Ring) 지원      |
| freebsd-x11        | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-x11">http://lists.FreeBSD.org/mailman/listinfo/freebsd-x11</a>               | FreeBSD 에서 X11 유지 및 지원              |

제한된 리스트: 다음 리스트는 더 전문화된 사용자를 위한 곳이어서, 일반적인 사람들은 관심이 없을 것이다. 이들 제한된 리스트에 가입하기 전에 기술적인 리스트에 출석하여 관련된 통신 에티켓을 이해하는 것도 괜찮은 생각이다.

| 리스트          | URL                                                                                                                         | 목적             |
|--------------|-----------------------------------------------------------------------------------------------------------------------------|----------------|
| freebsd-hubs | <a href="http://lists.FreeBSD.org/mailman/listinfo/freebsd-hubs">http://lists.FreeBSD.org/mailman/listinfo/freebsd-hubs</a> | 미러 사이트를 운용하는 사 |

|                     |                                                               |                                                         |
|---------------------|---------------------------------------------------------------|---------------------------------------------------------|
|                     | fo/freebsd-hubs                                               | 람들                                                      |
| freebsd-user-groups | http://lists.FreeBSD.org/mailman/listinfo/freebsd-user-groups | 유저 그룹 coordination                                      |
| freebsd-vendors     | http://lists.FreeBSD.org/mailman/listinfo/freebsd-vendors     | 릴리즈 엔지니어와 FreeBSD 기반 제품 개발자의 협력                         |
| freebsd-www         | http://lists.FreeBSD.org/mailman/listinfo/freebsd-www         | <a href="http://www.FreeBSD.org">www.FreeBSD.org</a> 관리 |

*간추린 리스트:* 위의 모든 리스트는 간추린 포맷에서 사용할 수 있다. 리스트에 등록하면, 여러분의 계정 옵션 섹션에서 간추린 옵션을 변경할 수 있다.

*CVS 리스트:* 다음 리스트는 소스 트리에서 다양한 영역의 변경 로그 메시지에 관심 있는 사람들을 위한 것이다. 이들은 *읽기 전용*이기 때문에 메일을 보내지 않는다.

| 리스트          | URL                                                    | 소스 영역                                 | 영역 설명(소스 목적)        |
|--------------|--------------------------------------------------------|---------------------------------------|---------------------|
| cvb-all      | http://lists.freebsd.org/mailman/listinfo/cvb-all      | /usr/(CVSROOT doc ports projects src) | 트리의 모든 변경           |
| cvb-doc      | http://lists.freebsd.org/mailman/listinfo/cvb-doc      | /usr/(doc www)                        | Doc 와 www 트리의 모든 변경 |
| cvb-ports    | http://lists.freebsd.org/mailman/listinfo/cvb-ports    | /usr/ports                            | 포트 트리의 모든 변경        |
| cvb-projects | http://lists.freebsd.org/mailman/listinfo/cvb-projects | /usr/projects                         | 프로젝트 트리의 모든 변경      |
| cvb-src      | http://lists.freebsd.org/mailman/listinfo/cvb-src      | /usr/src                              | src 트리의 모든 변경       |

## C.1.2 가입하는 방법

리스트에 가입하려면, 위의 리스트 이름의 URL을 입력하거나

<http://lists.FreeBSD.org/mailman/listinfo>에서 관심 있는 리스트를 클릭한다. 리스트 페이지는 등록에 필요한 사항을 가지고 있다.

---

실제로 리스트에 질문을 올리기 위해서는 간단히 <[listname@FreeBSD.org](mailto:listname@FreeBSD.org)>에 메일을 보낸다. 그러면 전 세계의 메일링 리스트 멤버들에게 다시 배포된다.

리스트에서 등록을 해지하려면, 리스트에서 받은 모든 메일의 하단에서 찾을 수 있는 URL을 클릭한다. 직접 등록을 해지하기 위해 `freebsd-[listname]-unsubscribe@FreeBSD.org`에 메일을 보내도 된다.

우리는 여러분들이 기술적인 진로에 대해, 기술적인 메일링 리스트에 올린 내용에 대해 답변할 것이다. 여러분이 중요한 공고만 관심이 있다면 FreeBSD 공고 메일링 리스트에 (<http://lists.freebsd.org/mailman/listinfo/freebsd-announce>) 가입하기를 권장한다.

### C.1.3 리스트 약정

모든 FreeBSD 메일링 리스트는 사용자들이 지켜야 될 특정 기본 룰을 가지고 있다. 이들 가이드라인의 요구를 따르지 않으면 FreeBSD Postmaster <[postmaster@FreeBSD.org](mailto:postmaster@FreeBSD.org)>로부터 두 번의 경고를 받고, 3번째가 되면 관리자는 이들이 보낸 메일을 FreeBSD 메일링 리스트에서 모두 삭제하고 필터링 한다. 이런 룰과 판정이 필요하다는 것은 유감이지만 요즘 인터넷은 보기에 상당히 위험하고, 이런 메커니즘들이 깨지기 쉽다는 것을 보통 식별하지 못한다.

지켜야 될 룰:

게시하는 주제는 리스트의 기본 약정을 준수 한다. 예를 들어 리스트가 기술적인 문제에 대한 것이라면 여러분이 게시하는 것도 기술적인 것이어야 된다. 엉뚱한 잡담이나 모든 사람들을 위한 메일링 리스트의 의미를 훼손하는 글 등은 허용되지 않는다. 특정 주제가 아닌 자유로운 대화는, FreeBSD 채팅 메일링 리스트를(<http://lists.freebsd.org/mailman/listinfo/freebsd-chat>) 자유롭게 이용하기 바란다.

2 개 이상의 메일링 리스트에 같은 글을 올리지 않고, 양쪽 리스트에 꼭 글을 올려야 되면 오직 2 개만 사용한다. 대부분의 리스트는 이해하기 힘든 문자를 조합한 것을(“- stable & -scsi” 이런 식의) 제외하고, 이미 다른 가입자들의 토론으로 넘쳐나기 때문에 한번에 하나 이상의 글을 올릴 필요는 없다. 여러

---

메일링 리스트가 Cc 라인에 나타나는 형식으로, 메시지가 여러분에게 전달됐다면 다시 이 메시지를 보내기 전에 Cc 라인을 제거해야 된다. 누가 최초로 글을 올렸는지 상관없이 여러분의 cross-postings 에 대해 책임이 있다.

유저뿐만 아니고 개발자도 마찬가지로 개인적인 공격과 신성모독은 용납되지 않는다. 표절이나 허용이 안된 개인적인 메일을 다시 게시해서 문제가 발생하는 넷 티켓 위반은 주의가 요구 되지만 강제적인 것은 아니다. 그러나 매우 드물지만 이러한 내용이 리스트의 약정에 맞는 경우도 있기 때문에 경고만 있는 경우도 있다.

FreeBSD 와 관련 없는 제품 광고나 서비스는 강력히 금지되고, 이에 따라 위반자가 스팸으로 광고를 한다면 즉시 금지된다.

*리스트 별 약정:*

freebsd-acpi (<http://lists.FreeBSD.org/mailman/listinfo/freebsd-acpi>)

*ACPI 와 전원 관리 개발*

freebsd-afs (<http://lists.FreeBSD.org/mailman/listinfo/freebsd-afs>)

*Andrew File System*

이 리스트는 CMU/Transarc 에서 AFS 를 포팅해서 사용하는 것에 대한 토론이다.

freebsd-announce (<http://lists.FreeBSD.org/mailman/listinfo/freebsd-announce>)

*Important events / milestones*

이것은 중요한 FreeBSD 이벤트의 공고에만 관심 있는 사람들을 위한 메일링 리스트다. 이것은 스냅샷과 다른 릴리즈에 대한 공고를 포함한다. 새로운 FreeBSD 기능에 대한 공고도 포함한다. 또한 도움을 요청하는 내용 등이 올라갈 것이다. 적절히 제어되는 메일링 리스트다.

---

freebsd-arch

*아키텍처와 디자인 토론*

이 리스트는 FreeBSD 아키텍처에 대한 토론이다. 메시지는 대부분 아주 기술적이다. 적당한 주제에 대한 예는:

Heidemann 레이어가 동작하도록 VFS 로 무엇을 수정해야 되는가

많은 버스와 아키텍처에서 같은 드라이버를 사용할 수 있도록, 장치 드라이버 인터페이스를 어떻게 변경 할 수 있는가

네트워크 드라이버를 어떻게 작성하는가

freebsd-audit (<http://lists.FreeBSD.org/mailman/listinfo/freebsd-audit>)

*소스 코드 감사 프로젝트*

이것은 FreeBSD 소스 코드 감사 프로젝트의 메일링 리스트다. 처음에는 보안 관련 변경 사항에 대한 토의였지만, 변경된 코드에 대한 재 검토로 확장되었다.

이 리스트는 매우 자주 폐치 되고, 일반적인 FreeBSD 유저들은 별로 관심이 없을 것이다. 보안 토론은 freebsd-security 에서 유지하고 있는 특정 소스코드와 관련 없다. 거꾸로 모든 개발자는 재검토를 위해 자신들의 폐치를 이곳으로 보내도록 하고, 특히 버그가 시스템에 영향을 주는 부분을 수정한 경우 폐치를 이곳으로 보내도록 한다.

freebsd-binup (<http://lists.FreeBSD.org/mailman/listinfo/freebsd-binup>)

*FreeBSD 바이너리 업데이트 프로젝트*

---

이 리스트는 바이너리 업데이트 시스템이나 **binup** 토론을 제공하기 위한 것이다. 디자인 이슈, 자세한 실행, 패치, 버그 리포트, 상태 리포트, 특징 요청, commit 로그, 그리고 **binup** 과 관련된 다른 것들.

freebsd-bugbusters (<http://lists.FreeBSD.org/mailman/listinfo/freebsd-bugbusters>)

*문제 리포트 제어 노력의 조화*

이 리스트는 Bugmeister 를 위한 조화와 토론 포럼, Bugbusters, 그리고 PR 데이터 베이스에 관심을 가지고 있는 사람들을 위해 제공된다. 이 리스트는 특정 버그, 패치나 PR 에 대해서 토론하지 않는다.

freebsd-bugs (<http://lists.FreeBSD.org/mailman/listinfo/freebsd-bugs>)

*버그 리포트*

FreeBSD 버그를 보고하기 위한 메일링 리스트다. 버그가 발생하면 send-pr(1) 명령이나 웹 인터페이스(<http://www.freebsd.org/send-pr.html>)를 사용하여 버그를 보고한다.

freebsd-chat

*FreeBSD 커뮤니티와 관련 있는 기술적이지 않은 아이템*

이 리스트는 다른 리스트에 비해 기술적이지 않은 사회적인 다양한 정보가 넘쳐나고 있다. Jordan 이 toon ferret 처럼 보이는지, 누가 커피를 너무 많이 마시는지, 어디서 최고의 맥주를 양조하는지, 누가 지하에서 맥주를 양조하는지 등이 포함된다. 가끔 통보하는 중요한 이벤트가(다가 오는 파티, 결혼식, 생일, 새로운 직업 등과 같은) 기술적인 리스트에 통보 될 수 있지만, 다음 활동들은 이 채팅 리스트에 의해 방향이 정해진다.

freebsd-core

---

*FreeBSD 코어 팀*

핵심 멤버를 위한 내부적인 메일링 리스트다. 심각한 FreeBSD 관련 문제의 조정이 필요 하거나 확인이 필요할 때 메시지를 보낼 수 있다.

freebsd-current (<http://lists.freebsd.org/mailman/listinfo/freebsd-current>)

*FreeBSD-CURRENT 에 대한 토론*

FreeBSD-CURRENT 유저를 위한 메일링 리스트다. -CURRENT 에 새로 추가된 기능 중 유저에게 영향을 끼치는 기능에 대한 경고와, 스텝들에 대한 지시 사항을 포함한다. "CURRENT"를 사용하는 유저는 이 리스트에 가입해야 된다. 이것은 아주 기술적인 내용을 기대할 수 있는 기술적인 메일링 리스트다.

freebsd-cvsweb (<http://lists.freebsd.org/mailman/listinfo/freebsd-cvsweb>)

*FreeBSD CVSweb 프로젝트*

FreeBSD-CVSweb 사용, 개발과 관리에 대한 기술적인 토론

freebsd-doc (<http://lists.freebsd.org/mailman/listinfo/freebsd-doc>)

*문서 프로젝트*

이 메일링 리스트는 FreeBSD 문서 작성 프로젝트와 이슈를 토론하는 곳이다. 이 메일링 리스트의 멤버는 보통 "FreeBSD 문서 프로젝트" 라고 부른다. 이것은 오픈된 리스트기 때문에 자유롭게 등록하여 기고할 수 있다.

freebsd-firewire (<http://lists.freebsd.org/mailman/listinfo/freebsd-firewire>)

*FireWire (iLink, IEEE 1394)*

---

FreeBSD 의 FireWire(IEEE 1394) 서브시스템의 디자인과 시행에 대한 토론을 하는 메일링 리스트다. 관련 주제는 표준, 버스 장치와 프로토콜, 아답터 보드/카드/칩셋, 그리고 아키텍처와 적절한 지원 코드 시행을 포함한다.

freebsd-fs (<http://lists.freebsd.org/mailman/listinfo/freebsd-fs>)

#### *파일 시스템*

FreeBSD 파일 시스템에 대한 토론. 이것은 아주 기술적인 내용을 기대할 수 있는 기술적인 메일링 리스트다.

freebsd-geom (<http://lists.freebsd.org/mailman/listinfo/freebsd-geom>)

#### *GEOM*

GEOM 과 실행에 관련된 특별한 토론. 이것은 아주 기술적인 내용을 기대할 수 있는 기술적인 메일링 리스트다.

freebsd-gnome (<http://lists.freebsd.org/mailman/listinfo/freebsd-gnome>)

#### *GNOME*

FreeBSD 시스템의 **GNOME** 데스크톱 환경에 대한 토론. 이곳은 아주 기술적인 내용을 기대할 수 있는 기술적인 메일링 리스트다.

freebsd-ipfw (<http://lists.freebsd.org/mailman/listinfo/freebsd-ipfw>)

#### *IP Firewall*

FreeBSD 에서 IP 방화벽 코드의 재 디자인에 대한 기술적인 토론이다. 이곳은 아주 기술적인 내용을 기대할 수 있는 기술적인 메일링 리스트다.



---

freebsd-ia64 (<http://lists.freebsd.org/mailman/listinfo/freebsd-ia64>)

*FreeBSD 를 IA64 에 포팅*

FreeBSD 를 Intel 에서 IA-64 플랫폼으로 포팅하는 각각의 활동을 위한 메일링 리스트다. 기술적인 토론에 대해 관심 있는 사람들은 언제든지 환영한다.

freebsd-isdn (<http://lists.freebsd.org/mailman/listinfo/freebsd-isdn>)

*ISDN 커뮤니티*

FreeBSD 를 지원하는 ISDN 개발에 대한 메일링 리스트다.

freebsd-java (<http://lists.freebsd.org/mailman/listinfo/freebsd-java>)

*Java 개발*

FreeBSD 에 중요한 Java 어플리케이션 개발과 JDK 포팅과 관리에 대한 메일링 리스트다.

freebsd-jobs (<http://lists.freebsd.org/mailman/listinfo/freebsd-jobs>)

*일자리 제공*

FreeBSD 와 관련된 구인란과 이력서를 올리는 포럼이다. 다시 말해 FreeBSD 와 관련된 구인을 원하거나 직업을 원한다면 이곳에 광고 하면 된다. 일반적인 고용을 위한 메일링 리스트는 다른 곳에 있고, 이곳은 일반적인 고용 문제를 위한 메일링 리스트가 아니다.

다른 FreeBSD.org 메일링 리스트처럼 이 리스트도 전 세계로 뿌려지기 때문에, 원하는 지역이나, 재택 근무 또는 이주하는 것을 지원할 수 있는지 정확히 입력한다.

---

메일은 오픈 포맷만 사용한다 - 평범한 텍스트가 바람직하지만 기본 PDF, HTML, 그리고 몇 가지 다른 포맷도 허용된다. Microsoft Word (.doc) 와 같은 포맷은 메일링 리스트 서버에서 거부 당한다.

freebsd-kde (<http://freebsd.kde.org/mailman/listinfo/kde-freebsd>)

*KDE*

FreeBSD 시스템의 **KDE** 에 대한 토론. 이곳은 아주 기술적인 내용을 기대할 수 있는 기술적인 메일링 리스트다.

freebsd-hackers (<http://lists.freebsd.org/mailman/listinfo/freebsd-hackers>)

*기술적인 토론*

FreeBSD 와 관련된 기술적인 토론을 위한 포럼이다. 이곳은 메인 기술적인 메일링 리스트다. FreeBSD 에 대한 활동과 문제 해결이나 다른 대안 솔루션에 대해 토론한다. 기술적인 토론에 관심 있는 사람들을 환영한다. 이것은 아주 기술적인 내용을 기대할 수 있는 기술적인 메일링 리스트다.

freebsd-hardware (<http://lists.freebsd.org/mailman/listinfo/freebsd-hardware>)

*FreeBSD 하드웨어 일반적인 토론*

FreeBSD 가 운영되는 다양한 종류의 하드웨어와, 다양한 문제를 피할 수 있는 하드웨어 제안에 대한 토론.

freebsd-hubs (<http://lists.freebsd.org/mailman/listinfo/freebsd-hubs>)

*미래 사이트*

---

FreeBSD 미러 사이트를 운영중인 사람을 위한 통보와 토론

freebsd-isp (<http://lists.freebsd.org/mailman/listinfo/freebsd-isp>)

*인터넷 서비스 공급자들의 이슈*

이 메일링 리스트는 FreeBSD 를 사용하는 인터넷 서비스 공급자에(ISP) 대한 토론을 위한 것이다. 이것은 아주 기술적인 내용을 기대할 수 있는 기술적인 메일링 리스트다.

freebsd-newbies (<http://lists.freebsd.org/mailman/listinfo/freebsd-newbies>)

*Newbies 활동에 대한 토론*

우리는 다른 어디서도 다루어지지 않은 새로운 유저들의 활동을 다룬다. 독립적인 학습과 문제 해결 능력, 자원 발견과 사용, 그리고 다른 사람에게 도움 요청, 메일링 리스트를 어떻게 사용하고 어떤 리스트를 사용해야 되는가, 일반적인 채팅, 실수, 거만, 아이디어 공유, 스토리, 도덕적인 지원과 FreeBSD 커뮤니티 중 활동적인 커뮤니티에 등록하는 것을 포함한다. 우리는 우리의 문제를 받아서 FreeBSD 문제들을 지원하며, 다른 방법으로 FreeBSD 에 새로운 유저들을 다른 사람들과 만나게 하여 우리가 하는 것과 같은 일을 하게 한다 (서로 경험한 바를 바탕으로 서로의 경험과 아이디어를 공유시키기 위해...).

freebsd-openoffice (<http://lists.freebsd.org/mailman/listinfo/freebsd-openoffice>)

*OpenOffice.org*

OpenOffice.org 와 StarOffice 포팅과 관리에 대한 토론.

freebsd-performance (<http://lists.freebsd.org/mailman/listinfo/freebsd-performance>)

*FreeBSD 튜닝과 속도 증가에 대한 토론*

이 메일링 리스트는 해커, 관리자, 그리고 FreeBSD 의 속도에 관심 있는 단체들을

---

위한 곳이다. 부하가 많은 곳이나 제한 적인 곳에서 FreeBSD 설치를 포함하여 허용되는 주제는 경험적인 성능 문제를 다룬다. FreeBSD 성능을 증가 시키려는 단체들은 이 리스트에 등록할 것을 강력히 권장한다. 이것은 이상적으로 아주 기술적인 리스트기 때문에 경험이 많은 FreeBSD 유저, 해커, 그리고 FreeBSD 속도, 안정성 등에 관심이 있는 관리자에게 적합하다. 이 리스트는 질문 답변 대신 문서로 되어 있지만, 답변이 안된 성능 관련 주제에 대한 질문에 답변할 수 있다.

freebsd-platforms (<http://lists.freebsd.org/mailman/listinfo/freebsd-platforms>)

*Intel 플랫폼이 아닌 머신에 포팅*

FreeBSD 크로스 플랫폼 이슈에 대한 일반적인 토론과, Intel 머신이 아닌 다른 머신으로 FreeBSD 포팅 제안과 관련 있다.

freebsd-policy (<http://lists.freebsd.org/mailman/listinfo/freebsd-policy>)

*코어 팀 정책 결정*

FreeBSD 코어 팀 정책 결정 메일링 리스트는 읽기 전용이다.

freebsd-ports (<http://lists.freebsd.org/mailman/listinfo/freebsd-ports>)

*“포트”에 대한 토론*

FreeBSD “포트 컬렉션” (/usr/ports), 포트 인프라스트럭처, 그리고 일반적인 포트 관련 토론. 이것은 아주 기술적인 내용을 기대할 수 있는 기술적인 메일링 리스트다.

freebsd-ports-bugs (<http://lists.freebsd.org/mailman/listinfo/freebsd-ports-bugs>)

*“포트” 버그에 대한 토론*

FreeBSD 의 “포트 컬렉션” (/usr/ports)의 문제 리포트, 포트 제안, 또는 포트 수정

---

에 관한 토론이다. 이것은 아주 기술적인 내용을 기대할 수 있는 기술적인 메일링 리스트다.

freebsd-questions (<http://lists.freebsd.org/mailman/listinfo/freebsd-questions>)

*유저 질문*

이것은 FreeBSD 에 관한 질문 메일링 리스트다. 상당히 기술적인 질문이 아니라면, 아주 막연한 질문은 보내지 않는다.

freebsd-scsi (<http://lists.freebsd.org/mailman/listinfo/freebsd-scsi>)

*SCSI 서브시스템*

이것은 SCSI 서브시스템에서 FreeBSD 를 운영 하는 사람들을 위한 메일링 리스트다. 이것은 아주 기술적인 내용을 기대할 수 있는 기술적인 메일링 리스트다.

freebsd-security (<http://lists.freebsd.org/mailman/listinfo/freebsd-security>)

*보안 문제*

FreeBSD 컴퓨터 보안 문제 (DES, Kerberos, 보안 구멍과 수정 등). 이것은 아주 기술적인 내용을 기대할 수 있는 기술적인 메일링 리스트다. 질문 답변 리스트는 아니지만 FAQ 에서는 환영한다(질문 과 답변 둘 다).

freebsd-security-notifications (<http://lists.freebsd.org/mailman/listinfo/freebsd-security-notifications>)

*보안 통보*

FreeBSD 보안 문제와 수정에 대한 통보. 이곳은 토론 리스트가 아니고, 토론 리스트는 FreeBSD-security 다.

---

freebsd-small (<http://lists.freebsd.org/mailman/listinfo/freebsd-small>)

*임베디드 어플리케이션에 FreeBSD 사용*

이 리스트는 보통 작고 임베디드 FreeBSD 설치에 관한 토론이다. 이것은 아주 기술적인 내용을 기대할 수 있는 기술적인 메일링 리스트다.

freebsd-stable (<http://lists.freebsd.org/mailman/listinfo/freebsd-stable>)

*FreeBSD-STABLE 사용에 대한 토론*

이것은 FreeBSD-STABLE 유저를 위한 메일링 리스트다. -STABLE 에 새로이 추가되어 유저에게 영향을 주는 기능, -STABLE 에 남아있는 기능에 대한 경고를 포함한다. "STABLE"를 운용하는 사람은 이 리스트에 등록하는 것이 좋다. 이것은 아주 기술적인 내용을 기대할 수 있는 기술적인 메일링 리스트다.

freebsd-standards (<http://lists.freebsd.org/mailman/listinfo/freebsd-standards>)

*C99 & POSIX Conformance*

이것은 FreeBSD 가 C99 와 POSIC 표준을 따르는 것에 대한 기술적인 토론이다.

freebsd-user-groups (<http://lists.freebsd.org/mailman/listinfo/freebsd-user-groups>)

*유저 그룹 Coordination 리스트*

이것은 각 로컬 유저 그룹의 운영자들이 서로간에 혹은 핵심 멤버(Core team)들로부터, 특정 사람들과 문제들을 토론할 수 있게 하는 메일 리스트이다. 이 메일링 리스트는 유저 그룹들간의 회의 개요와 프로젝트 방안 등으로 제한된다.

---

freebsd-vendors (<http://lists.freebsd.org/mailman/listinfo/freebsd-vendors>)

*벤더*

FreeBSD 프로젝트와 소프트웨어 및 하드 웨어 벤더간의 이슈에 관한 토론.

## C.1.4 메일링 리스트에서 필터링

FreeBSD 메일링 리스트는 스팸, 바이러스, 그리고 다른 원하지 않는 메일을 배포하지 않도록 여러 가지 방법으로 필터링 되고 있다. 메일링 리스트 보호에 사용되는 모든 것을 포함 하지는 않지만, 필터링 동작을 이 섹션에서 설명한다.

특정 타입의 첨부 파일만 메일링 리스트에 허용된다. 아래 리스트에 없는 MIME 컨텐츠 타입의 모든 첨부 파일은, 메일링 리스트에서 메일이 배포되기 전에 제거된다.

application/octet-stream

application/pdf

application/pgp-signature

application/x-pkcs7-signature

message/rfc822

multipart/alternative

multipart/related

multipart/signed

text/html

text/plain

---

text/x-diff

text/x-patch

**Note:** 어떤 메일링 리스트는 다른 MIME 콘텐츠 타입의 첨부 파일을 허용하겠지만, 위의 리스트는 대부분의 메일링 리스트에 적용할 수 있다.

메일이 HTML 과 평범한 텍스트 버전을 가지고 있다면, HTML 버전은 삭제 된다. 메일이 HTML 버전만 가지고 있다면 평범한 텍스트로 변환된다.

## C.2 유즈넷 뉴스 그룹

두 개의 FreeBSD 뉴스 그룹뿐만 아니고, FreeBSD 가 토론되고 FreeBSD 유저와 관련 있는 다른 많은 뉴스 그룹이 있다. 키 워드 검색 아카이브를 사용할 수 있는 뉴스 그룹 중 몇 곳은 Warren Toomey <wkt@cs.adfa.edu.au>의 호의에 의한 것이다.

### C.2.1 BSD 관련 뉴스 그룹

comp.unix.bsd.freebsd.announce (<news:comp.unix.bsd.freebsd.announce>)

comp.unix.bsd.freebsd.misc (<news:comp.unix.bsd.freebsd.misc>)

de.comp.os.unix.bsd (독일어) (<news:de.comp.os.unix.bsd>)

fr.comp.os.bsd (프랑스어) (<news:fr.comp.os.bsd>)

it.comp.os.freebsd (이탈리아어) (<news:it.comp.os.freebsd>)



---

## C.2.2 다른 유닉스 뉴스 그룹

comp.unix (<news:comp.unix>)

comp.unix.questions (<news:comp.unix.questions>)

comp.unix.admin (<news:comp.unix.admin>)

comp.unix.programmer (<news:comp.unix.programmer>)

comp.unix.shell (<news:comp.unix.shell>)

comp.unix.user-friendly (<news:comp.unix.user-friendly>)

comp.security.unix (<news:comp.security.unix>)

comp.sources.unix (<news:comp.sources.unix>)

comp.unix.advocacy (<news:comp.unix.advocacy>)

comp.unix.misc (<news:comp.unix.misc>)

comp.bugs.4bsd (<news:comp.bugs.4bsd>)

comp.bugs.4bsd.ucb-fixes (<news:comp.bugs.4bsd.ucb-fixes>)

comp.unix.bsd (<news:comp.unix.bsd>)

## C.2.3 X 윈도우 시스템

comp.windows.x.i386unix (<news:comp.windows.x.i386unix>)

comp.windows.x (<news:comp.windows.x>)

---

comp.windows.x.apps (<news:comp.windows.x.apps>)

comp.windows.x.announce (<news:comp.windows.x.announce>)

comp.windows.x.intrinsics (<news:comp.windows.x.intrinsics>)

comp.windows.x.motif (<news:comp.windows.x.motif>)

comp.windows.x.pex (<news:comp.windows.x.pex>)

comp.emulators.ms-windows.wine (<news:comp.emulators.ms-windows.wine>)

## C.3 웹 서버

주 서버

<http://www.FreeBSD.org/>

아르헨티나

<http://www.ar.FreeBSD.org/>

호주

<http://www.au.FreeBSD.org/>

<http://www2.au.FreeBSD.org/>

오스트리아

<http://www.at.FreeBSD.org/>

<http://www2.at.FreeBSD.org/> (IPv6)

---

벨기에

<http://freebsd.unixtech.be/>

브라질

<http://www.br.FreeBSD.org/>

<http://www2.br.FreeBSD.org/www.freebsd.org/>

<http://www3.br.FreeBSD.org/>

캐나다

<http://www.ca.FreeBSD.org/>

<http://www2.ca.FreeBSD.org/>

중국

<http://www.cn.FreeBSD.org/>

코스타리카

<http://www1.cr.FreeBSD.org/>

체코 공화국

<http://www.cz.FreeBSD.org/>

덴마크

<http://www.dk.FreeBSD.org/> (IPv6)

<http://www3.dk.FreeBSD.org/>

에스토니아

<http://www.ee.FreeBSD.org/>

핀란드

---

<http://www.fi.FreeBSD.org/>

<http://www2.fi.FreeBSD.org/>

프랑스

<http://www.fr.FreeBSD.org/>

독일

<http://www.de.FreeBSD.org/>

<http://www1.de.FreeBSD.org/>

<http://www2.de.FreeBSD.org/> (IPv6)

그리스

<http://www.gr.FreeBSD.org/>

<http://www.FreeBSD.gr/>

홍콩

<http://www.hk.FreeBSD.org/>

헝가리

<http://www.hu.FreeBSD.org/>

<http://www2.hu.FreeBSD.org/>

아이슬란드

<http://www.is.FreeBSD.org/>

인도네시아

<http://www.id.FreeBSD.org/>

아일랜드

---

<http://www.ie.FreeBSD.org/>

<http://www2.ie.FreeBSD.org/>

이탈리아

<http://www.it.FreeBSD.org/>

<http://www.gufi.org/mirrors/www.freebsd.org/data/>

일본

<http://www.jp.FreeBSD.org/www.FreeBSD.org/> (IPv6)

대한민국

<http://www.kr.FreeBSD.org/>

<http://www2.kr.FreeBSD.org/>

쿠웨이트

<http://www.kw.FreeBSD.org/>

키르기스스탄

<http://www.kg.FreeBSD.org/>

라티비아

<http://www.lv.FreeBSD.org/>

리투아니아

<http://www.lt.FreeBSD.org/>

네덜란드

<http://www.nl.FreeBSD.org/>

<http://www2.nl.FreeBSD.org/>

---

뉴질랜드

<http://www.nz.FreeBSD.org/>

노르웨이

<http://www.no.FreeBSD.org/>

<http://www2.no.FreeBSD.org/> (IPv6)

필리핀

<http://www.FreeBSD.org.ph/>

폴란드

<http://www.pl.FreeBSD.org/>

<http://www2.pl.FreeBSD.org/>

포르투갈

<http://www.pt.FreeBSD.org/>

<http://www1.pt.FreeBSD.org/>

<http://www4.pt.FreeBSD.org/>

<http://www5.pt.FreeBSD.org/>

루마니아

<http://www.ro.FreeBSD.org/>

<http://www1.ro.FreeBSD.org/>

<http://www2.ro.FreeBSD.org/>

<http://www3.ro.FreeBSD.org/>

러시아

---

---

<http://www.ru.FreeBSD.org/>

<http://www2.ru.FreeBSD.org/>

<http://www3.ru.FreeBSD.org/>

<http://www4.ru.FreeBSD.org/>

산마리노

<http://www.sm.FreeBSD.org/>

싱가포르

<http://www2.sg.FreeBSD.org/>

슬로바키아 공화국

<http://www.sk.FreeBSD.org/>

<http://www2.sk.FreeBSD.org/>

슬로베니아

<http://www.si.FreeBSD.org/>

<http://www2.si.FreeBSD.org/>

남 아프리카

<http://www.za.FreeBSD.org/>

<http://www2.za.FreeBSD.org/>

스페인

<http://www.es.FreeBSD.org/>

<http://www2.es.FreeBSD.org/>

<http://www3.es.FreeBSD.org/>

---

스웨덴

<http://www.se.FreeBSD.org/>

<http://www2.se.FreeBSD.org/>

스위스

<http://www.ch.FreeBSD.org/>

<http://www2.ch.FreeBSD.org/>

대만

<http://www.tw.FreeBSD.org/>

<http://www2.tw.FreeBSD.org/>

<http://www3.tw.FreeBSD.org/>

<http://www4.tw.FreeBSD.org/>

태국

<http://www.th.FreeBSD.org/>

터키

<http://www.tr.FreeBSD.org/>

<http://www2.tr.FreeBSD.org/>

<http://www.enderunix.org/freebsd/>

우크라이나

<http://www.ua.FreeBSD.org/>

<http://www2.ua.FreeBSD.org/>

<http://www5.ua.FreeBSD.org/>



---

<http://www4.ua.FreeBSD.org/>

영국

<http://www.uk.FreeBSD.org/>

<http://www2.uk.FreeBSD.org/>

<http://www3.uk.FreeBSD.org/>

<http://www4.uk.FreeBSD.org/>

<http://www1.uk.FreeBSD.org/> (IPv6)

미국

<http://www2.us.FreeBSD.org/>

<http://www4.us.FreeBSD.org/> (IPv6)

<http://www5.us.FreeBSD.org/> (IPv6)

## C.4 메일 주소

다음 유저 그룹은 멤버에게 FreeBSD 관련 메일 주소를 제공한다. 리스트 된 관리자는 메일 주소를 남용할 경우 주소를 취소할 수 있는 권한을 가진 관리자다.

| 도메인                 | 기능     | 유저 그룹                          | 관리자                                  |
|---------------------|--------|--------------------------------|--------------------------------------|
| ukug.uk.FreeBSD.org | 포워드 전용 | <freebsd-users@uk.FreeBSD.org> | Lee Johnston<br><lee@uk.FreeBSD.org> |

---

## C.5 쉘 계정

다음은 FreeBSD 프로젝트를 지원하는 사람들에게 쉘 계정을 제공하는 유저 그룹이다. 리스트 된 관리자는 쉘 계정을 남용할 경우 계정을 취소할 수 있는 권한을 가진 관리자다.

| 도메인                     | 접근             | 유저 그룹                  | 관리자                                  |
|-------------------------|----------------|------------------------|--------------------------------------|
| storm.uk.FreeBSD.org    | SSH 만 접근       | 읽기 전용 cvs, 개인 웹 공간, 메일 | Lee Johnston<br><lee@uk.FreeBSD.org> |
| dogma.freebsd-uk.eu.org | Telnet/FTP/SSH | 메일, 웹 공간, 익명 FTP       | Lee Johnston<br><lee@uk.FreeBSD.org> |