

보안관리 시리즈 - 1

# 웹 서버 보안관리 가이드

Guide to Security Management of Web Servers

2003. 9

 정보통신부

 한국정보보호진흥원  
Korea Information Security Agency

# 발 간 사

최근 국가·사회적으로 중요한 업무들이 웹 기반으로 전환되고 있는 가운데 웹 서버 보안의 중요성이 강조되고 있습니다. 과거에는 단순히 정보의 공유를 목적으로 사용되었던 웹 서버가 이제는 행정, 금융, 통신 등 국민의 실생활에 꼭 필요한 사회적 기능의 기반 시스템으로 자리매김하게 되었습니다.

그러나 최근 들어 웹 서버에 대한 침해사고가 급증하고 있어 새로운 사회적 혼란의 우려를 낳고 있습니다. 웹 해킹을 통해 개인 신용정보가 유출되거나 불법적인 경제행위가 이루어지기도 하며, 국가간 마찰이 웹 해킹을 이용한 사이버 테러 형태로 번지기도 합니다. 이러한 사고들을 미루어 웹 해킹을 통한 국가 중요 행정 업무의 마비, 경제적 혼란 등 더 큰 사건들의 발생에 대해서도 예견해 볼 수가 있게 되었습니다.

한국정보보호진흥원에서는 최근에 급증하고 있는 웹 서버 침해사고에 적극적으로 대응하기 위하여 웹 서버 운영 및 개발의 실무자들이 안전한 웹 서버 운영환경을 구축할 수 있도록 『웹 서버 보안관리 가이드』를 발간하게 되었습니다.

이 가이드는 웹 서버 보안설정 방법, 최신 웹 해킹기술에 대한 대응방안 등 웹 서버를 해킹으로부터 안전하게 구축 및 운영하는데 필요한 정보를 제공함으로써 웹 서버의 보안을 한층 강화하는데 많은 도움이 되리라고 생각하며 그동안 많은 의견을 제시하여 주신 분께 깊이 감사드립니다.

2003년 9월

한국정보보호진흥원

원장 김 창 곤

이 보고서는 정보통신기반보호 지원사업의 일환으로 한국정보보호진흥원  
기반시설보호단 기반보호기획팀 연구원들이 참여하여 작성하였으며, 웹 보  
안 전문가 및 웹 서버 실무 운영자들의 많은 조언이 있었습니다.

2003년 9월

사업 책임자 : 이홍섭(책임연구원)

연구 책임자 : 노병규(선임연구원)

참여 연구원 : 박영우(선임연구원)

김성훈(선임연구원)

윤 준(연구원)

조영덕(연구원)

# 목 차

<b>제 1 장 서 론</b> .....	<b>1</b>
제 1 절 개요 .....	1
제 2 절 가이드 활용방법 .....	3
<b>제 2 장 웹 서버 보안현황</b> .....	<b>5</b>
제 1 절 웹 서버 사용현황 .....	5
제 2 절 웹 서버 해킹사고 발생현황 .....	7
제 3 절 웹 서버 침해사례 .....	9
<b>제 3 장 웹 서버 보안수칙</b> .....	<b>19</b>
제 1 절 호스트 OS 보안 .....	19
제 2 절 웹 서버 설치 보안 .....	22
제 3 절 웹 서버 운영 보안 .....	27
제 4 절 웹 어플리케이션 설계 보안 .....	29
<b>제 4 장 아파치 보안설정 방법</b> .....	<b>33</b>
제 1 절 서버보안 설정 .....	33
제 2 절 메인 설정 파일(httpd.conf) .....	36
제 3 절 .htaccess 파일 .....	48
제 4 절 인증 및 접근제어 .....	49
제 5 절 로깅 .....	64
<b>제 5 장 IIS 보안설정 방법</b> .....	<b>73</b>
제 1 절 서버 보안 설정 .....	73
제 2 절 IIS 보안 설정 .....	98
제 3 절 인증 및 접근제어 .....	118
제 4 절 로깅 .....	124

제 6 장 웹 어플리케이션 보안 .....	129
제 1 절 최신 취약성 분석 및 보안 대책 .....	129
제 2 절 서버측 스크립트 언어 보안 .....	158
제 7 장 웹 서버를 위한 안전한 네트워크 구성 .....	175
참 고 자 료 .....	179
부 록 .....	180

# 표 목 차

[표 4-1] 아파치 주요 디렉토리에 대한 접근권한 .....	33
[표 4-2] 아파치 주요 파일에 대한 접근권한 .....	34
[표 4-3] 아파치 에러 로그의 위험도 레벨 .....	66
[표 5-1] IIS 동작에 필요한 서비스 목록 .....	78
[표 5-2] IIS 웹 서버에서 사용중지를 권장하는 서비스 목록 .....	79
[표 5-3] IIS 웹 서버에서 접근제어를 설정하도록 권장하는 명령어 목록 .....	90
[표 5-4] IIS 관련 감사를 수행해야 할 정책 목록 .....	95
[표 5-5] IIS 예제들의 기본 위치 .....	102
[표 5-6] IIS에서 제거를 권장하는 매핑 목록 .....	104
[표 5-7] W3C 확장 로그 파일 형식의 속성 .....	126
[표 6-1] 주요 웹 어플리케이션 취약성과 대책 .....	130
[표 6-2] Web.config 파일의 설정 .....	170

# 그림 목 차

(그림 1-1) 제3장의 활용 예시 .....	4
(그림 2-1) 전세계 웹 서버 시장점유율 .....	5
(그림 2-2) 국내 정부기관이 사용하는 웹 서버 현황 .....	6
(그림 2-3) 웹사이트 침해 경험 .....	7
(그림 2-4) 웹사이트 침해 경로 .....	8
(그림 2-5) 해킹당한 일본 총무성 홈페이지 .....	9
(그림 2-6) 해킹당한 미국 노동부 홈페이지 .....	10
(그림 2-7) 해킹당한 미국 보건후생부 홈페이지 .....	10
(그림 2-8) 해킹당한 NASA 홈페이지 .....	11
(그림 2-9) 해킹당한 캘리포니아 주의회 공화당 간부회의 사이트 ....	12
(그림 2-10) 해킹당한 호주 Australian Institute of Marine Science 사이트 ..	12
(그림 2-11) 해킹당한 영국 Swindon Borough Council 사이트 .....	13
(그림 2-12) 해킹당한 국내 모 연구소 내부 웹 서버 .....	14
(그림 2-13) 해킹당한 국내 외국계 회사 홈페이지 .....	15
(그림 2-14) 해킹당한 국내 모 대기업 홈페이지 .....	16
(그림 6-1) 일반적인 웹 서비스 구성 .....	129
(그림 6-2) 관리자 페이지 노출 .....	138
(그림 6-3) XSS 취약점 예시 .....	144
(그림 6-4) 사이트의 오류 메시지 예 .....	152
(그림 6-5) 디렉토리 리스팅 예 - PHP .....	157
(그림 6-6) 데이터베이스 접속 계정 정보 노출 예 .....	164
(그림 6-7) 유효성 검사 예 .....	167
(그림 6-8) Tomcat 관리자 페이지 노출 예 .....	172
(그림 7-1) 웹 서버를 위한 네트워크 구성 예시-1 .....	175
(그림 7-2) 웹 서버를 위한 네트워크 구성 예시-2 .....	176

# 제 1 장 서 론

## 제 1 절 개 요

최근 많은 기업들이 대고객 업무를 웹 기반으로 운영하고 있고 국가도 행정 서비스를 웹 기반으로 전환하고 있는 가운데 웹 해킹 사고가 자주 발생하고 있어 웹 서버의 안전한 운영이 중요한 이슈로 부각되고 있다.

이제 웹 서버는 회사 홍보나 정보를 제공하는 단순한 기능에서 벗어나 다양한 사회적 기능의 기반 시스템으로 자리잡게 되었다. 사람들은 각자 자신의 집이나 회사에서 웹을 통해서 증권·금융 업무를 볼 수도 있고, 항공·호텔 예약이나 쇼핑, 정부 행정서비스 등의 업무도 볼 수 있다. 사회 활동이 일어나는 장소가 물리적 공간에서 웹으로 이동하고 있는 것이다.

이렇게 국가·사회적으로 중요한 기능들의 웹에 대한 의존도가 높아지고 있는 가운데 최근 웹 해킹 사고가 자주 발생하고 있어 새로운 사회적 혼란의 우려를 낳고 있다. 최근 들어 웹 해킹 사고가 언론에 자주 보도되고 있는데, 그 원인은 웹 서버가 사회적으로 중요한 기능을 담당하고 있는 상황에서 웹 서버에 대한 해커들의 관심도 부쩍 증대되었기 때문이라고 볼 수 있다. 또한 웹 서버는 다른 시스템에 비해 해킹이 비교적 쉬워서 초보 해커들도 손쉽게 해킹을 할 수 있다는 것과 해킹의 효과가 크다는 것이 중요한 원인으로 작용하고 있다.

웹 서비스는 기능의 특성상 다른 서비스와는 달리 반드시 외부에 노출되어 있어야 하고 방화벽의 보호를 받기 어렵다. 그리고 다양한 어플리케이션들이 웹 서비스와 연동되어 있어서 많은 보안 취약점들이 존재한다. 특히 새로운 웹 기술들이 개발되면서 전에 없던 새로운 형태의 보안 취약점들이 꾸준히 생겨나고 있다. 이러한 웹의 특징들은 해커들에게 끊임없는 흥미거리를 제공하여 웹 해킹의 매력을 높여주고 있다.

또한 웹사이트는 방송 매체와 같이 대중에 대한 정보전달력이 있어서, 다른 해킹 방법에 비해 해킹 효과가 크다. 개인이나 이익집단이 자신들의 요구사항이나 주장



을 사회에 전달하기 위해 웹 해킹이 동원되기도 한다. 특히 최근에 발생하고 있는 국가간 마찰과 관련하여 국가간 사이버 전쟁이 벌어지기도 하는데, 이때 해킹 대상으로 주로 이용되는 것이 의사전달 효과가 높은 웹 서버이다.

한국정보보호진흥원은 이와 같이 증가하고 있는 웹 서버에 대한 보안 위협에 효율적으로 대응하기 위하여 웹 서버 보안관리 가이드를 발간하게 되었다. 이 가이드에서는 현재 전세계적으로 가장 많이 사용되고 있는 아파치 웹 서버와 IIS 웹 서버를 중심으로 해킹의 위협으로부터 웹 서버를 안전하게 운영하는데 필요한 보안수칙을 제공한다. 또한 최신 웹 해킹 기법들이 주로 웹 어플리케이션을 대상으로 함에 따라 웹 어플리케이션을 안전하게 개발 및 운영하는데 필요한 웹 어플리케이션 보안에 관한 내용도 포함하고 있다<sup>1</sup>. 웹 어플리케이션 보안은 최신 웹 어플리케이션 취약점을 중심으로 설명한다.

이 가이드는 제2장 웹 서버 보안현황, 제3장 웹 서버 보안수칙, 제4장 아파치 보안설정 방법, 제5장 IIS 보안설정 방법, 제6장 웹 어플리케이션 보안, 제7장 웹 서버를 위한 안전한 네트워크 구성으로 구성되어 있다.

제2장 웹 서버 보안현황에서는 웹 서버 사용현황과 웹 해킹 사고 발생 현황, 분야별 웹 해킹 사례를 소개한다. 제3장 웹 서버 보안수칙에서는 웹사이트의 설계 및 개발, 운영 전반에 걸쳐 적용해야 하는 보안수칙들을 설명하고 있는데, 웹 서버 보안을 호스트 OS 보안, 웹 서버 설치 보안, 웹 서버 운영 보안, 웹 어플리케이션 설계 보안의 4단계로 나누어 각 단계에서 적용해야 하는 보안수칙을 기술하고 있다. 제4장과 제5장에서는 아파치 및 IIS의 설치 및 운영시 주의해야 하는 보안관련 사항들을 설명하고, 각 웹 서버에서 제공하는 보안 기능들을 자세한 예와 함께 설명하고 있다. 제6장 웹 어플리케이션 보안에서는 최근 웹 해킹에 많이 사용되고 있는 웹 어플리케이션 취약점과 관련하여 문제점을 분석하고 이를 해결하기 위한 보호대책을 설명한다. 마지막으로 제7장 웹 서버를 위한 안전한 네트워크 구성에서는 웹 서버의 안전을 고려한 네트워크 구성을 예를 들어 간단하게 설명하고 있다.

---

<sup>1</sup> 웹 해킹은 웹 서버 해킹과 웹어플리케이션 해킹으로 구분할 수 있다. 웹 서버 해킹은 아파치나 IIS 등 웹 서비스 프로그램에 존재하는 취약점을 이용한 공격이고, 웹어플리케이션 해킹은 JSP나 PHP 등 웹프로그래밍 언어와 데이터베이스 등을 연동하여 작성한 웹프로그램에 존재하는 보안 취약점을 이용한 공격이다.

## 제 2 절 가이드 활용방법

이 가이드는 구성상 처음부터 모든 장을 읽을 필요없이 필요한 부분만을 선택하여 읽을 수도 있다. 제4장에서는 아파치 보안설정 방법을 설명하고 있고, 제5장에서는 IIS 보안설정 방법을 설명하고 있다. 제6장에서는 최신 웹 어플리케이션의 취약성에 대한 분석과 대응방법을 설명하고 있다.

아파치 웹 서버 관리자는 “제4장 아파치 보안설정 방법”에 기술된 각 항목을 검토하면서 운영중인 서버가 올바르게 설정되어 있는지 점검하도록 한다. IIS 웹 서버 관리자도 제5장을 활용하여 동일한 방법으로 점검할 수 있다. 웹 어플리케이션 개발자는 최신 웹 해킹 기법으로부터 안전한 프로그램을 개발하기 위하여 웹 프로그래밍을 시작하기 전에 제6장을 검토하는 것이 좋다.

모든 웹 서버 환경에서 공통적으로 적용해야 하는 보안수칙은 제3장에서 설명하고 있다. 제3장은 웹 서버 보안을 호스트 OS 보안, 웹 서버 설치 보안, 웹 서버 운영 보안, 웹 어플리케이션 설계 보안의 4단계로 나누어서 각 단계에서 적용해야 하는 보안수칙을 기술하고 있다. 이 부분은 특정 웹 서버나 어플리케이션에 관련해서 설명한 것이 아니기 때문에, 이 가이드에서 다루지 않는 웹 서버의 관리자도 이 부분을 참고할 수 있다.

제3장의 각 항목들은 관련이 있는 제4장 ~ 제7장의 항목과 연결되어 있다<sup>2</sup>. 예를 들어 (그림 1-1)과 같이 제3장에 “샘플 파일, 매뉴얼 파일, 임시 파일의 제거”라는 항목이 있고, 항목 옆에 아파치 2, IIS 11, IIS 15이 표시되어 있다. 이것은 이 항목이 아파치 2번 항목, IIS 11번 항목, IIS 15번 항목과 관련이 있음을 나타내는 것이다.

제3장의 항목에서는 웹 서버의 안전을 위해 샘플 파일, 매뉴얼 파일, 임시 파일을 삭제하라고 권고하고 있다. 그런데 아파치 2번 항목에서는 어떤 디렉토리를 삭제해야 하는지 설명하고 있고, IIS 15번 항목에서는 IIS에 어떤 예제 응용 프로그램들이 있고, 이들을 어떻게 삭제할 수 있는지 구체적인 방법을 설명하고 있다. 이러한 구성을 활용하여 제3장을 읽다가 연결된 제4장과 제5장의 항목을 읽는다면 이

---

<sup>2</sup> 제4장과 제5장에서는 각 웹 서버의 보안관련 기능들 중 중요한 것들을 중심으로 기술하였다. 관련된 보안 기능이 없거나 상세한 설명이 필요없는 것들은 생략하였다. 따라서 제3장의 항목과 제4장 또는 제5장의 내용이 반드시 일대일 대응이 되지는 않는다.

해가 훨씬 빠를 것이다.

### 제 3 장 웹 서버 보안수칙

**■ 샘플 파일, 매뉴얼 파일, 임시 파일의 제거 (아파치 2, IIS 11, IIS 15)**

웹서버를 설치하면 기본적으로 설치되는 샘플 파일이나 매뉴얼 파일은 시스템 관련 정보를 노출하거나 해킹에 악용될 수 있다. 따라서 웹서버 설치 후에 반드시 이러한 파일들을 찾아서 삭제하도록 한다.

민감 관리 등의 이유로 앱을 통해 설명문서에 접근을 해야 한다면 접근제어를 통해 꼭 필요한 사용자만 접근을 허용하고 그 외의 사용자들은 접근하지 못하도록 설정한다.

또한 웹 서버를 정기적으로 검사하여 임시 파일들을 삭제하도록 한다. 특히 웹서버의 업데이트나 유지보수시 생성되는 백업파일이나 중요한 파일 등은 작업이 끝난 후 반드시 제거하도록 한다.

정확한 관리를 위해 폴더와 파일의 이름과 위치, 개수 등이 적혀있는 별도의 문서를 관리하는 것이 좋다. 그래서 문서에 등록되지 않은 불필요한 파일들을 점검해서 삭제하도록 한다.

**■ 웹서버에 대한 불필요한 정보 노출 방지 (아파치 8, IIS 7)**

웹서버 종류, 사용 OS, 사용자 계정 이름 등 웹서버와 관련된 불필요한 정보가 노출되지 않도록 한다. 이러한 정보가 사소한 것처럼 보일 수 있지만, 이러한 정보를 아는 것만으로도 공격에 필요한 나머지 정보를 수집하는데 도움이 될 수 있다.

뉴스그룹이나 메일링 리스트를 통해 웹서버 운영에 대한 질문을 할 경우, 조직의 네트워크와 시스템에 대한 상세정보가 유출되지 않도록 주의한다. 실제로 한 보안 컨설팅 회사는 인터넷에 공개된 정보를 기반으로 아씨의 네트워크에 대한 상세한 지도를 그릴 수 있었다. 이 정보가 비록 기밀성이 떨어지는 정보일지는 몰라도 해킹을 위한 좋은 출발점이 될 수 있다.

### → 제 5 장 IIS 보안설정 방법

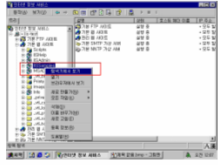
**[IIS 15] 모든 예제 응용 프로그램을 사용하지 않거나 제거**

IIS를 설치하면 기본적으로 예제와 설명서 등이 같이 설치된다. 이 폴더들은 해킹에 이용되거나 백도어가 숨어질 위험이 있으므로 제거해 주어야 한다. 다음 표는 IIS 예제들이 디폴트로 저장되는 위치이다.

예제	가상 디렉토리	위치
IIS 예제	\\\\IISamples	C:\inetpub\wwwroot\IISamples
IIS 설명서	\\\\IIShelp	C:\inetpub\wwwroot\IIShelp
데이터 액세스	\\\\MSADC	C:\Program Files\System\MSADC\files\system\msadc

이러한 가상 디렉토리와 실제폴더를 삭제하는 방법은 다음과 같다.

- ① [인터넷 서비스 관리자]에서 삭제하려는 가상 디렉토리(여기에서는 "IISamples")를 선택하고, 마우스 오른쪽 버튼을 클릭한 후 [탐색기에서 보기]를 선택한다. 그러면 해당 디렉토리를 보여주는 탐색기 창이 뜨게 된다.



- ② 우선 인터넷 서비스 관리자에서 아래와 같이 가상 디렉토리를 삭제한다.

(그림 1-1) 제3장의 활용 예시

제2장은 웹 보안 현황을 파악할 수 있는 자료들로 구성되어 있다. 그 중에서 웹 해킹 사례 부분은 최근 웹 해킹이 어떤 목적으로 발생하고 있고 얼마나 큰 피해를 가져올 수 있는지를 설명하고 있어서 웹 보안의 중요성을 깨닫는데 도움이 될 것이다.

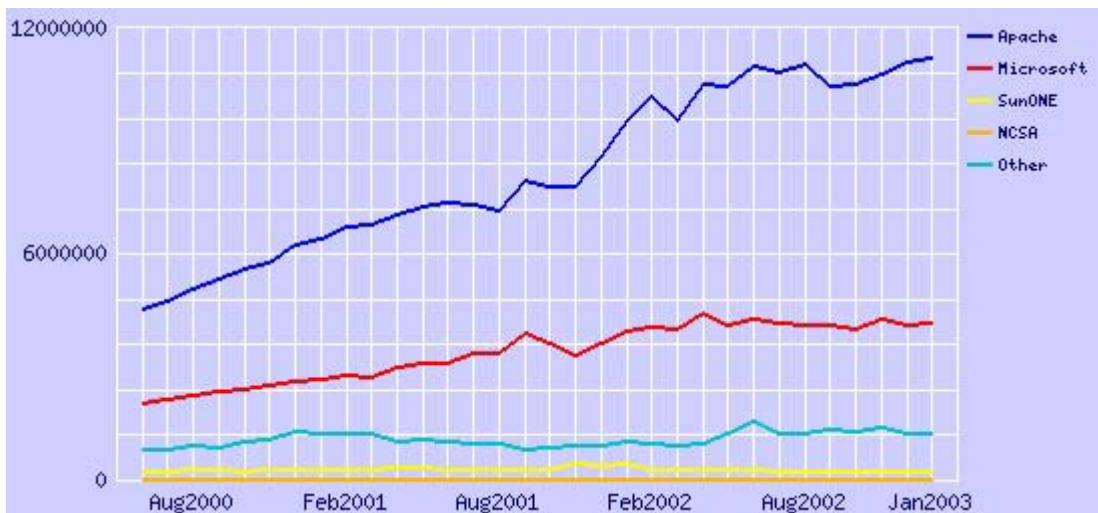
## 제 2 장 웹 서버 보안현황

### 제 1 절 웹 서버 사용현황

영국의 인터넷서비스 회사 Netcraft<sup>3</sup>는 매월 주요 웹 서버 제품의 세계 시장점유율을 조사해서 발표하고 있다. Netcraft의 조사에 따르면 2003년 1월 현재 인터넷에 연결된 웹 서버 중 66%를 아파치가 차지하고 있는 것으로 나타났다.

Netcraft는 인터넷에 연결된 호스트들을 대상으로 웹 서버 사용 통계를 조사해서 발표하고 있는데, 조사방법은 서버네임을 이용해서 서버에 http요청을 보내고 그 응답을 기반으로 웹 서버의 종류를 파악하는 것이다.

개발자	2002년 12월(%)		2003년 1월(%)		변동폭
Apache	11,065,427	(66.54)	11178,715	(66.42)	-0.12
Microsoft	4,113,590	(24.74)	412,101	(24.79)	0.05
Zeus	258,367	(1.55)	261,652	(1.55)	0.00
SunONE	229,081	(1.3)	233,105	(1.39)	0.01



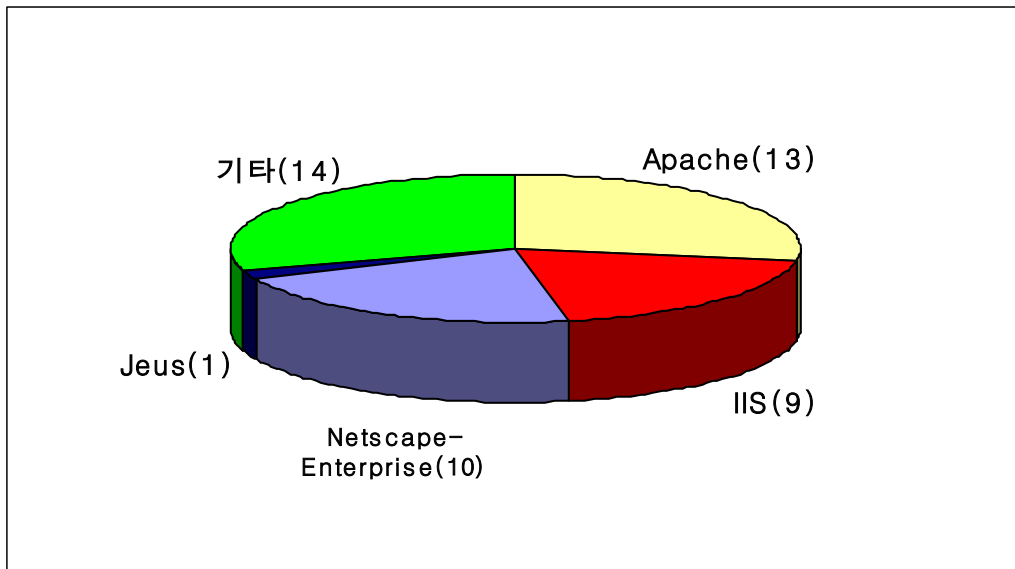
(그림 2-1) 전세계 웹 서버 시장점유율

<sup>3</sup> <http://www.netcraft.com/>

2003년 1월의 조사결과는 (그림 2-1)<sup>4</sup>과 같다. 총 35,424,956 사이트로부터 응답을 수신했고, 그 중에서 중복되는 사이트를 제외하고 1개의 IP를 하나의 사이트로 취급하여 계산하였다.

그림에서 보는 바와 같이 2003년 1월 전세계 웹 서버 시장은 아파치가 66%로 과반수 이상을 차지하고 있고, IIS, Zeus, SunONE이 그 뒤를 따르고 있다. 한가지 주목할 만한 사실은 아파치와 IIS가 시장점유율 91%로 전세계 웹 서버 시장의 대부분을 차지하고 있다는 것이다. 이에 이 가이드에서는 아파치와 IIS에 대한 보안설정 방법을 구체적으로 다루기로 했다.

한국정보보호진흥원이 조사한 국내 47개 주요 정부기관의 웹 서버 사용현황 조사 결과는 (그림 2-2)와 같다<sup>5</sup>. 국내 정부기관에서 사용하는 웹 서버 현황은 전세계 웹 서버 시장점유율과 조금 다르게 나타났다. 다음 그림에서 보는 바와 같이 정부기관들은 아파치, SunONE, IIS를 고루 사용하고 있었다.



(그림 2-2) 국내 정부기관이 사용하는 웹 서버 현황

<sup>4</sup> 조사결과에서 SunONE은 iPlanet-Enterprise, Netscape-Enterprise, Netscape-FastTrack, Netscape-Commerce, Netscape-Communications, Netsite-Commerce, Netsite-Communications을 포함하고, Microsoft는 Microsoft-IIS, Microsoft-IIS-W, Microsoft-PWS-95, Microsoft-PWS를 포함한다.

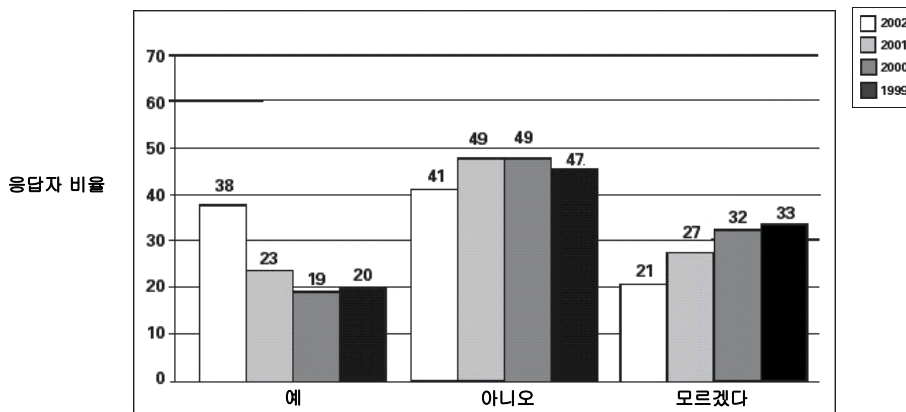
<sup>5</sup> 조사과정에서 47개 정부기관은 청와대 홈페이지 정부조직도에서 선정하였으며, 웹 서버 조사 방법은 각 기관의 도메인네임에 대해 Netcraft의 Web Server Query 기능을 이용하였다.

## 제 2 절 웹 서버 해킹사고 발생현황

최근에 발생하고 있는 해킹사고는 웹 해킹이 주류를 이루고 있다. 많은 업무들이 웹을 기반으로 발전함에 따라 전체적으로 웹사이트의 수가 증가하고 있고, 이에 비례하여 웹 해킹 사고도 점차 증가하고 있다.

본질적으로 웹 서버는 다양한 어플리케이션과 연동되어 있어 많은 보안 취약점이 존재하고, 서비스 특성상 외부에 노출되어야 하기 때문에 많은 해커들의 주요 공격 대상이 되고 있다. 또한 일반 서버 시스템을 공격하는 것보다 웹 서버를 공격하는 것이 웹 서버의 대중성을 고려할 때, 해킹 효과가 무척 크므로 웹 해킹을 선호하는 경향이 있다.

지난 1년 동안 당신의 웹사이트에 허가받지 않은 접근(Unauthorized Access)이나 오용(Misuse)이 발생한 적이 있었는가?



출처 : Computer Security Institute

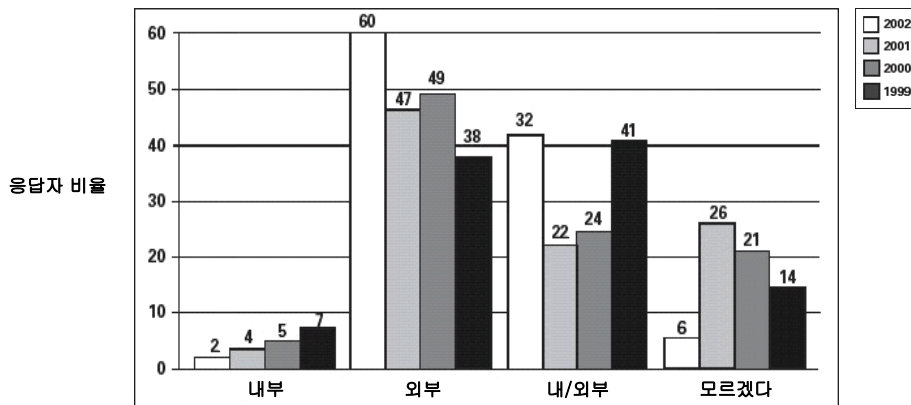
(그림 2-3) 웹사이트 침해 경험

미국의 컴퓨터 보안기관인 CSI(Computer Security Institute)가 미국 기업, 정부 기관, 금융기관, 의료기관, 대학의 실무자 503명을 대상으로 2002년에 실시한 설문 조사에서 응답자의 38%가 지난 일년동안 웹사이트 침해를 경험한 것으로 나타났다<sup>6</sup>.

<sup>6</sup> Richard Power, Computer Security Issues & Trends, Vol. VIII, No.1, Spring 2002.

이것은 2001년 23%에 비하면 큰 증가를 보인 것이다. 이것은 나머지 62%의 응답자 중 해킹사고를 당했지만 미처 발견하지 못한 경우를 고려할 때 매우 큰 수치라고 볼 수 있다. 또한 ‘예’라고 응답한 사람 중에 50% 정도가 5건 이상의 침해사고를 경험했다고 응답해 웹 서버에 대한 해킹이 크게 증가하고 있음을 보여주고 있다.

### 웹사이트 공격은 내부 혹은 외부로부터 오는가?



출처 : Computer Security Institute

(그림 2-4) 웹사이트 침해 경로

일반적으로 웹사이트 공격은 항상 외부로부터만 온다고 생각할 수 있으나 조사 결과는 그렇게 나타나지 않았다. 조사에 따르면 외부로부터만 공격이 왔다고 응답한 비율이 많은 부분을 차지했지만, 내부로부터 공격이 왔다고 응답한 비율이 2%, 내/외부로부터 공격이 왔다고 응답한 비율이 32%를 차지해, 내부로부터 오는 공격도 많은 부분을 차지하는 것으로 나타났다. 따라서 웹 서버 보안대책을 세울 때 외부로부터의 해킹만이 아니라 내부로부터의 해킹위협도 반드시 고려하도록 해야 할 것이다.

### 제 3 절 웹 서버 침해사례

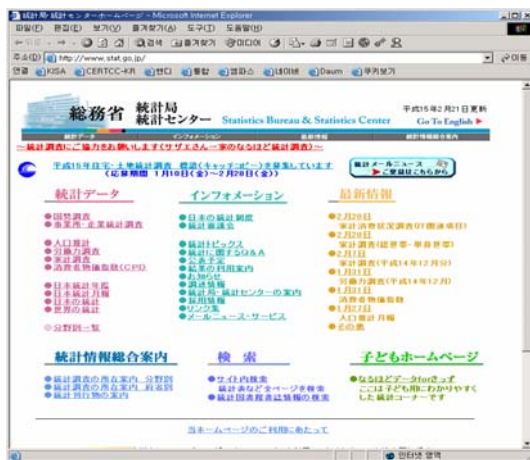
#### 1. 공공기관 침해사례

최근 전세계적으로 국가간 분쟁이 자주 발생하면서 사이버 공간에서도 이러한 분쟁과 관련해서 항의를 표시하는 일이 잦아지고 있다. 상대국가의 주요 사이트 게시판에 상대를 이해시키거나 자신의 주장을 펴기 위한 논리적인 글을 올리기도 하지만, 심한 욕설을 사용해서 상대국가를 비난하는 글을 올리기도 한다. 이러한 항의의 정도가 심해지면 웹 해킹이라는 사이버 테러가 발생하기도 하는데, 사이버 세계에서 그 나라의 얼굴이라고 할 수 있는 주요 공공기관 웹사이트가 주요 표적이 되고 있다.

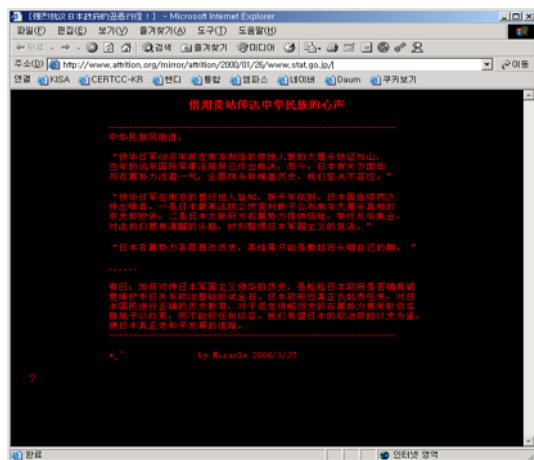
웹 서버는 다른 시스템과는 달리 대중에게 공개되어 있어 해킹을 할 경우 의사전달 효과가 높고, 해킹도 비교적 쉬운 편이라 항의표시 수단으로 쉽게 이용되고 있다. 다음은 최근 몇 년간 발생한 공공기관에 대한 웹 해킹 사례이다.

- 2000년 1월에 중국인으로 추정되는 해커가 일본 우익의 난징(남경)대학살 부정에 항의하여 일본 정부의 인터넷 홈페이지에 침입, 사이버 시위를 벌였다. 그는 일본 총무성이 개설한 홈페이지를 해킹하여 홈페이지의 첫화면을 일본을 비난하는 문구로 바꾸어 놓았다. 해커에 의해 변환된 화면에는 ‘(일본 우익단체의) 23일 난징대학살 부인 집회를 막지 않은 일본정부의 악랄한 행위를 고발하고 중화민족의 정의를 알린다’ 는 붉은색의 중국어 문구가 적혀 있었다.

정상 운영중인 사이트



해킹후 사이트



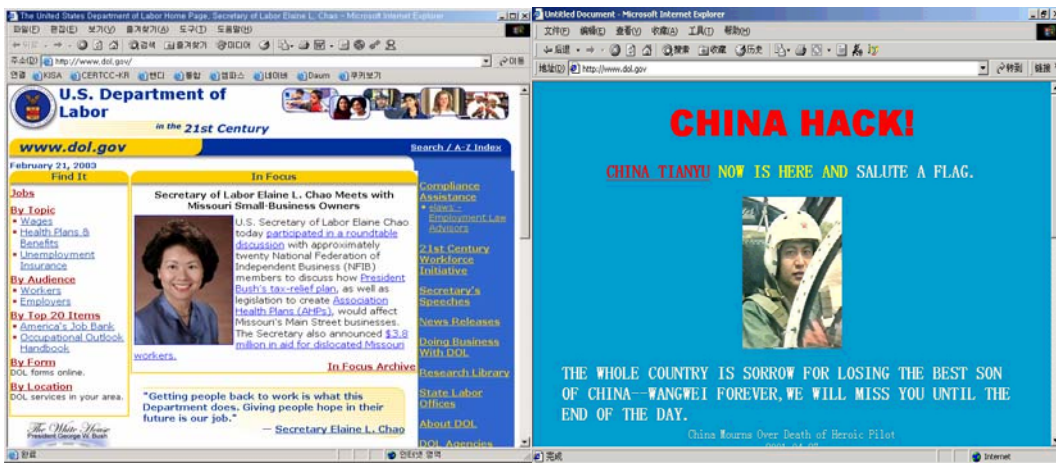
(그림 2-5) 해킹당한 일본 총무성 홈페이지



○ 2001년 4월 1일에 발생한 미국 경찰기와 중국 전투기의 충돌사건을 계기로 미국 해커와 중국 해커간에 사이버 전쟁이 벌어졌었다. 이 사이버 전쟁에서 미국 해커들의 선제 공격으로 최소 24개 중국 사이트가 침입을 당했고, 중국 해커들도 백악관, 노동부, FBI, NASA, CNN 등 37개 웹사이트에 대한 공격을 단행했다. 다음 그림은 2001년 4월에 중국인으로 추정되는 해커가 미국 노동부와 보건후생부의 웹사이트에 침입해 숨진 중국 전투기 조종사 왕웨이(王伟) 대위를 추모하는 글과 사진을 올려놓은 화면이다.

정상 운영중인 사이트

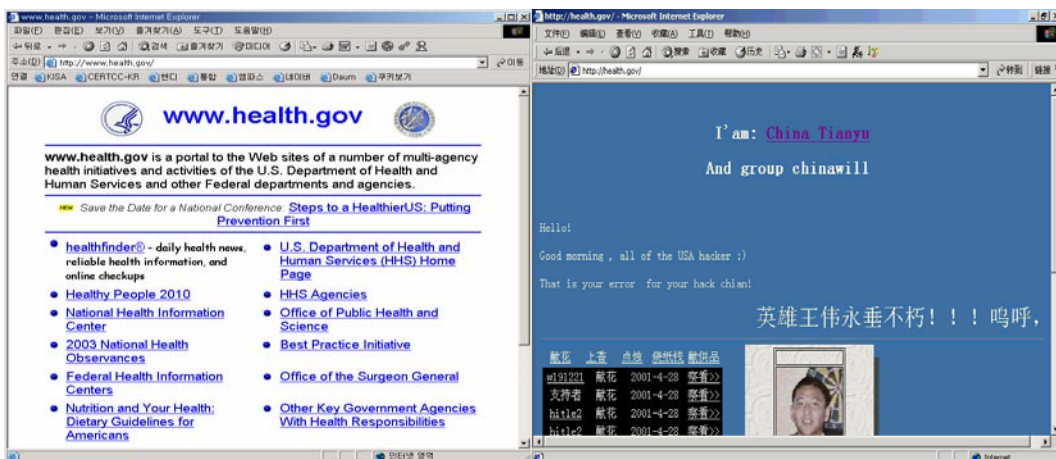
해킹후 사이트



(그림 2-6) 해킹당한 미국 노동부 홈페이지

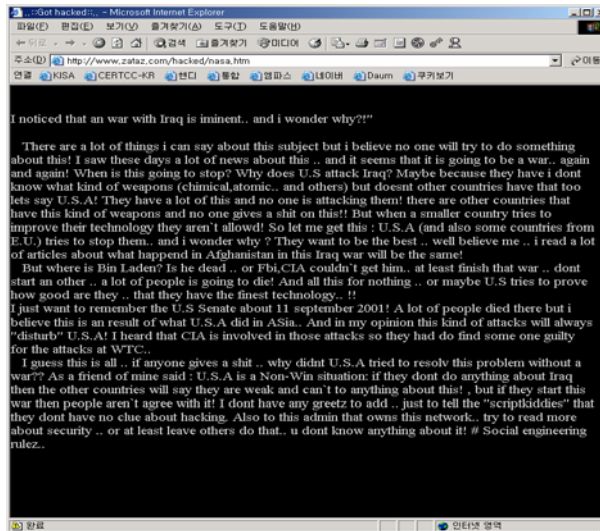
정상 운영중인 사이트

해킹후 사이트



(그림 2-7) 해킹당한 미국 보건후생부 홈페이지

- 2003년 2월 1일에 NASA의 홈페이지가 미국의 이라크 침공에 반대하는 해커 그룹에 의해 해킹당했다. NASA의 홈페이지는 몇 시간 동안 ‘왜 미국이 이라크를 침공하느냐’ 는 내용의 정치적인 메시지가 게시됐다. 이 사건의 주범은 자칭 ‘트리핀 스머프(Trippin Smurfs)’ 라는 해커 그룹이며, 이들은 다음 그림과 같은 이라크 침공과 관련된 메시지를 남겼다.



(그림 2-8) 해킹당한 NASA 홈페이지

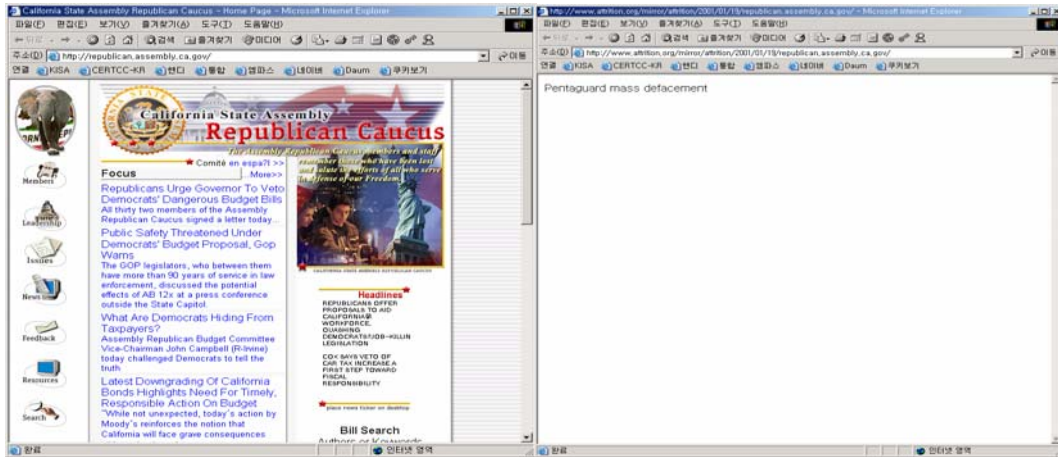
자신의 능력 과시와 우월감 표출은 과거부터 해커들의 전형적인 해킹 목적이었다. 사이버상에서 공권력의 상징인 정부 기관의 사이트에 대한 해킹은 공권력을 비웃고 자신의 우월감을 표출하기 위한 목적으로 현재에도 꾸준히 발생하고 있다.

- 2001년 1월에 미국, 영국, 호주 등 26개국 정부의 웹사이트가 펜타가드 (Pentagard)라는 해킹그룹에 의해 해킹을 당하는 사건이 발생했다. 펜타가드는 이전에도 정부나 군 웹사이트를 여러 번 해킹한 적이 있었다. 펜타가드는 해킹한 사이트에 원래의 내용을 지우고 ‘펜타가드 대량해킹’이라는 문구를 남겼으며, 자신들의 사이트를 통해 이번 공격은 정부사이트(.gov)와 군 사이트(.mil)에 대한 사상 최대규모의 해킹이라고 주장했다. 이들은 해킹 동기에 대해서는 밝히지 않았으나 ‘세계 모든 정부를 해킹하라. 맥주와 위스키, 보드카, 와인에서 힘을 얻은 우리는 다시 한번 정부와 군의 사이트를 해킹하러 왔다. 이들 사이트는 mp3도, 포르노도, 해적판 소프트웨어도 없는 쓸모없는 사이트다’ 라고 비아냥댔다. 이들은 1998년에 설립된 루마니아의 작

은 해킹단체이며 ‘월드와이드웹 전쟁’ 을 수행하고 있다고 주장했다. 다음 그림은 펜타가드에 의해 침해 당한 몇 개 사이트의 화면이다.

정상 운영중인 사이트

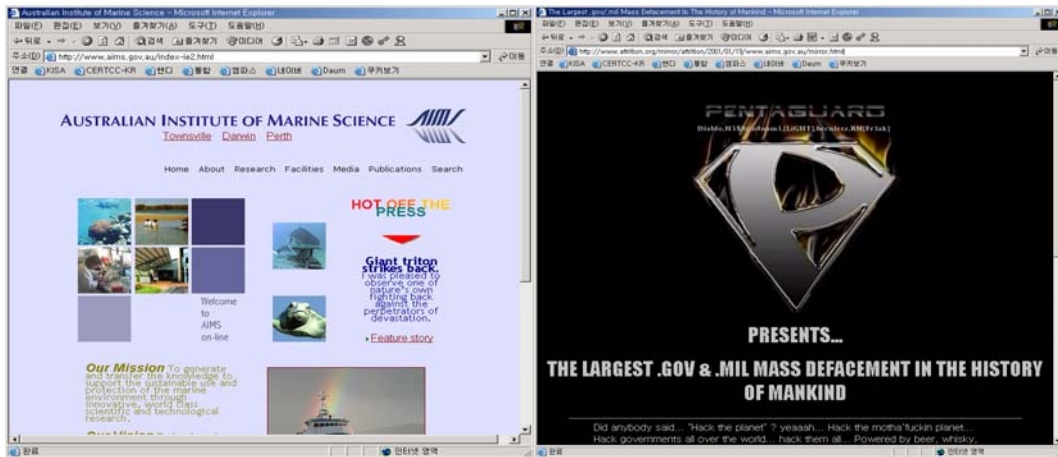
해킹후 사이트



(그림 2-9) 해킹당한 캘리포니아 주의회 공화당 간부회의 사이트

정상 운영중인 사이트

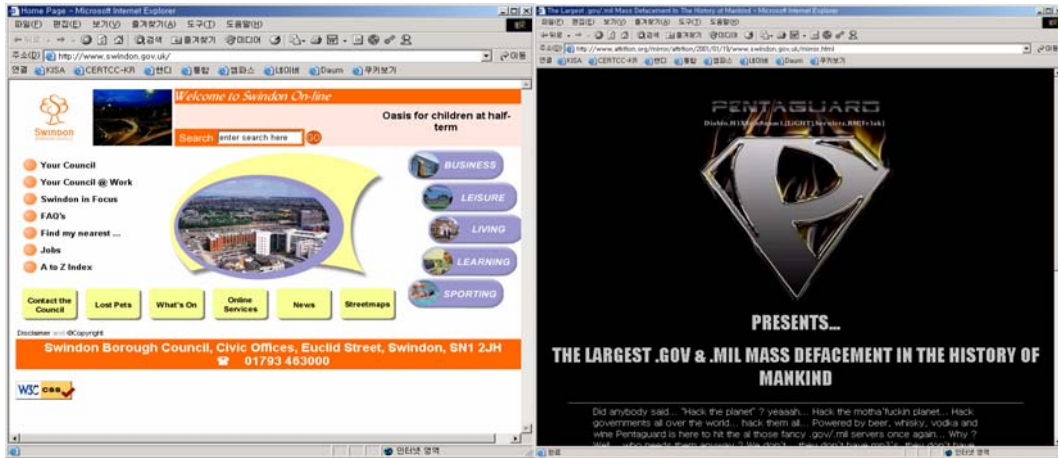
해킹후 사이트



(그림 2-10) 해킹당한 호주 Australian Institute of Marine Science 사이트

정상 운영중인 사이트

해킹후 사이트



(그림 2-11) 해킹당한 영국 Swindon Borough Council 사이트

## 2. 연구소 침해사례

규모가 큰 연구소의 경우 전산관리부서에서 연구소 웹 서버, 네트워크 장비 등을 외부침입으로부터 안전하도록 보안관리를 수행하고 있다. 그러나 각 연구부서에서 연구를 목적으로 운영하는 웹 서버나 연구원이 개인적인 목적으로 운영하는 웹 서버에 대해서는 보안관리를 소홀히 하고 있거나 파악조차 하고 있지 않아 이에 대한 침해사고가 자주 발생하고 있다.

많은 경우에 각 부서나 개인이 운영하는 웹 서버는 해당부서나 개인이 자율적으로 운영하고 있는데, 이들은 보안담당 직원과는 달리 해킹의 가능성을 심각하게 고려하지 않는다. 따라서 이들 웹 서버에 대한 보안관리가 거의 이루어지지 않게 되고, 자연히 많은 취약점이 존재하게 된다. 해커는 연구소 메인 웹 서버보다 취약점이 많은 이들 웹 서버를 주 표적으로 삼는다.

이 웹 서버에 대한 침해사고는 단순히 그 웹 서버의 피해만으로 그치는 것이 아니라, 전산관리부서에서 안전하게 운영되고 있는 것으로 간주되는 연구소내의 다른 서버에 대한 침입의 발판으로 이용될 수 있어 주의를 요한다.

- 2002년 12월에 국내 K연구소에서 웹 서버 해킹사고가 발생하였다. 다음 그림은 연구소내의 한 부서에서 운영하고 있던 웹 서버가 해킹 당하여 메인 페이지가 변조된 화면이다. 이 해킹사고 발생 후, 며칠동안 전산담당부서나 웹 서버 관리 부서에서는 해킹사실을 모르고 있었으며 아무런 조치를 취하지 않고 있었다.

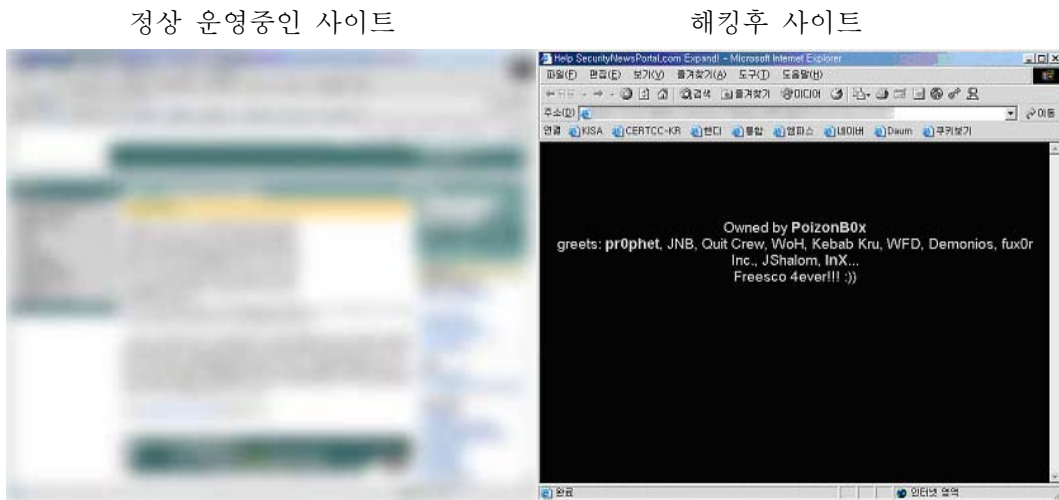


(그림 2-12) 해킹당한 국내 모 연구소 내부 웹 서버

### 3. 기업 침해사례

기업은 웹 서버 침해로 인한 피해가 연구소나 공공기관, 대학 등에 비해 큰 편이다. 기업의 웹 서버 침해는 직접적인 경제적 손실을 가져오거나 기업 이미지 하락으로 인한 매출 감소로 이어질 수 있다. 기업의 이미지는 판매촉진, 인재확보, 자금조달 등에 영향을 주어 기업의 성과에 중요한 역할을 하는데, 인터넷상에서 기업의 얼굴이라고 할 수 있는 홈페이지의 해킹은 기업의 명성과 신뢰도를 손상시킬 수 있다. 다음은 최근 몇 년간 국내 주요 기업에서 발생한 웹 해킹 사례이다.

- 다음 화면은 2001년 4월에 국내 대기업에서 운영하는 홈페이지와 외국계 회사의 홈페이지가 해킹당한 화면이다. 이러한 홈페이지 해킹사고는 빈번히 발생하고 있는데, 많은 기업들이 이미지 손상을 우려하여 해킹 사실을 비밀에 부치고 있다. 2001년 4월에 발생한 웹 해킹사고 중 국내 co.kr도메인을 가진 웹사이트에 대한 사고만 23건 이상이었다<sup>7</sup>.

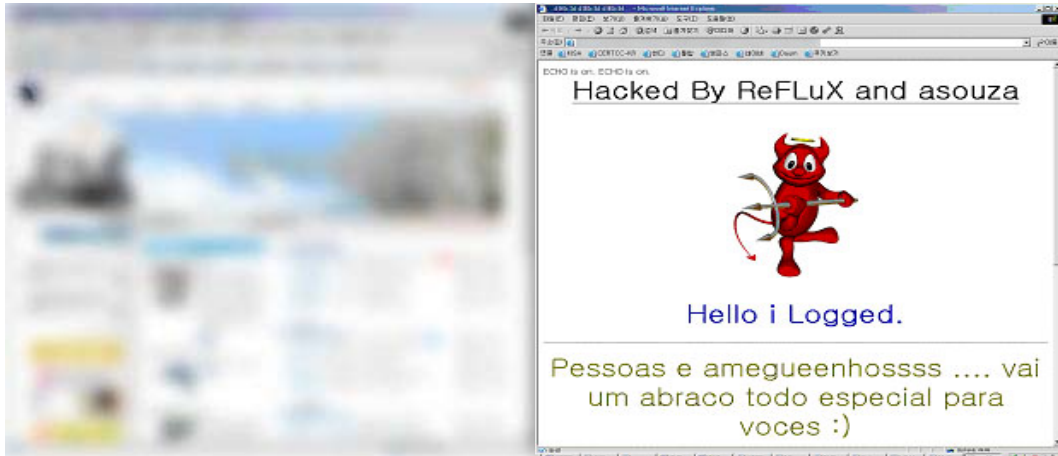


(그림 2-13) 해킹당한 국내 외국계 회사 홈페이지

<sup>7</sup> www.attrition.org 참조

정상 운영중인 사이트

해킹후 사이트



(그림 2-14) 해킹당한 국내 모 대기업 홈페이지

금융기관의 전산망에 해커가 침입할 경우 고객들의 금전적인 손실은 물론 금융업무의 마비가 초래될 가능성이 높다. 최근 금융기관들이 대고객 업무를 웹 기반으로 전환함에 따라 웹 서버의 결함은 곧 업무의 결함을 의미하게 되면서 웹사이트에 대한 침해가 매우 큰 경제적 손실을 가져올 수 있게 되었다.

- 2001년 8월에 국내 모 대학 전자계산소 연구원이 증권사의 웹 기반 증권거래 시스템을 해킹하여 4,300만원의 부당 이득을 챙기는 사고가 발생하였다. 그는 증권사에서 제공하는 웹 기반 증권거래 시스템에 대한 해킹 프로그램을 제작한 뒤 200만 번의 접속시도를 통해 모 증권사에서 사이버 거래를 하는 고객 200여명의 고객계좌와 비밀번호를 알아냈다. 그리고 나서 자신의 증권계좌에 미리 매수해 둔 모회사 주식에 대해 해킹한 계좌명의로 매수주문을 내 20억원 상당의 부당매매가 이뤄지도록 하는 방법으로 4300여만원의 부당 이득을 챙겼다. 웹 트레이딩 시스템은 이용자가 별도의 프로그램을 다운로드 하거나 별도의 프로그램 설치없이 곧바로 접속해서 거래할 수 있는 자바기반의 프로그램을 이용하게 되는데, 그는 자바(Java) 기반 프로그램의 취약점을 이용하였다. 프로그램 내용의 일부를 수정하여 일반 고객이 아닌 직원권한으로 침입하는데 성공, 무작위로 비밀번호를 찾아내는 해킹 프로그램을 가동해 10여일 만에 200여명에 이르는 고객의 계좌와 비밀번호를 알아냈다<sup>8</sup>. 2001년 7월 현재 사이버주식거래 규모가 68조원으로 전체 거래액의 66.4%를 차지하

<sup>8</sup> 이런 경우 특정 ID나 IP에서 단시간에 많은 접속시도가 있는 행위를 비정상 거래 행태로 간주하여 일정시간 접속을 차단하는 등의 대책이 필요하다.

고 있어 사이버주식거래 시스템에 대한 침해가 얼마나 심각한 경제적 혼란을 초래할지 짐작할 수 있다.

- 2001년 3월에 10대 해커 2명이 신용카드결제 승인처리업체 홈페이지에 침입, 회원 47만명의 신용카드 번호와 주민등록번호 등 핵심 신용정보를 훔쳐 이를 판매하려고 하였다. 신용카드결제 승인처리업체는 국내의 모든 신용카드회사와 연결되어 있어 침해 당할 경우 피해가 매우 클 수 있다. 이 10대 해커 두명은 신용카드결제 홈페이지에 접속, 국세청 의뢰로 이 회사에서 관리중인 신용카드 영수증 복권당첨자 47만명의 개인정보를 훔쳐냈다. 이어서 마케팅, 리서치전문업체 관계자 2,000여명에게 ‘마케팅과 리서치의 기본은 개인정보라고 합니다. …귀사에 정보를 판매하고자 합니다’ 라는 이메일을 보냈다. 이들은 이메일에서 ‘성명·주민번호 등 일반개인정보는 1인당 50원, 신용카드 번호 은행 계좌번호는 1인당 300원, 연봉 소득관련 정보는 1인당 600원에 판매한다’ 는 가격표까지 제시했다. 이들은 이 밖에도 다른 10개의 인터넷 사이트에서 780만명의 개인정보를 빼낸 것으로 밝혀졌다. 만약 이 개인정보가 악의를 가진 사람들에게 건네졌다면 전자상거래에 극심한 혼란이 초래됐을 것이다.

많은 기업들이 웹 기반으로 사업운영 및 영업전략을 펴나가면서 회원 정보를 웹 서버나 연동서버에 저장해두는 경우가 많아지고 있다. 인터넷 기업들이 무차별적으로 회원 유치경쟁을 벌이는 한편 수집된 개인정보에 대한 보안관리를 소홀히 하면서 개인정보 유출사태가 끊임없이 발생하고 있다.

- 2000년 12월에 고교 2년생인 해커가 국내 46개 인터넷사이트를 해킹, 630만명의 회원정보를 빼낸 뒤 해당 인터넷 관리자에게 메일을 통해 해킹사실을 알리며 금품을 요구하는 일이 있었다. 이 해커는 자신이 개발한 해킹 프로그램을 이용, 모 인터넷사이트를 해킹한 뒤 회원들의 이름과 주민등록번호, 주소, 출신학교, 직업 등이 입력된 개인정보를 빼냈다. 이 개인정보의 양은 당시 국내 전체 인터넷 이용자의 절반에 가까운 수치로서, 인터넷 운영자들이 수집한 회원정보에 대한 보안관리의 중요성을 깨닫게 해주고 있다.



#### 4. 대학 침해사례

대학은 일반 기업이나 공공기관에 비해 보안이 허술한 편이기 때문에 초보 해커들의 해킹연습장으로 이용되거나 해킹 추적을 피하기 위한 경유지로 많이 이용되어 왔다.

최근에는 중국-미국, 아프카니스탄-미국, 이라크-미국 등 국가간 분쟁이 자주 발생함에 따라 사이버테러나 지령하달의 중간 경유지로 활용될 가능성이 높아지고 있다.

대학에는 학사 업무를 위한 시스템 외에 많은 연구용 서버들과 수많은 학습용 PC들이 존재하기 때문에 전체 시스템에 대한 통합적인 보안관리가 어렵다. 또한 시스템 사용자가 어느 조직보다 많으며, 매년 20% 이상의 사용자가 새로운 사용자들로 바뀌기 때문에 다른 조직에 비해 체계적인 보안관리가 어려운 실정이다.

최근에 많은 대학에서 학사업무를 전산화하여 수강신청, 온라인 강의, 성적입력, 성적조회, 성적표 발급 등의 업무를 웹을 기반으로 수행하고 있다. 이에 따라 웹해킹을 통해 성적을 조작하거나 합격자 명단을 조작하는 등의 사례가 많이 발생하고 있다. 다음은 최근에 발생한 대학 웹 서버 침해사례이다.

- 2003년 1월에 서울대 합격발표 사이트가 해킹을 당하는 사고가 발생했다. 해커는 서울대 정시모집 최종합격자 발표 사이트를 해킹해서 합격자 공지사항에 ‘방법(네티즌 사이에서 쓰는 말로 ‘혼을 내주다’ 라는 뜻) 당했다’는 내용과 함께 ‘축하해용’이라는 농담 글을 게재하였고, 불합격자 공지사항에는 ‘힘내세요, 파이팅이에요~!’ 등의 글을 올렸다.

이번 사건에서는 큰 피해가 없었지만 만약 해커가 합격자 명단을 변조했었다면 수험생과 학부모들은 물론 입시관리 업무에 큰 혼란을 빚었을 것이다.

## 제 3 장 웹 서버 보안수칙

### 제 1 절 호스트 OS 보안

웹 서버를 설치하기 전에 호스트 OS의 보안을 먼저 확실히 해두어야 한다. 아무리 웹 서버를 안전하게 설정 및 운영한다 해도, 웹 서버가 설치될 OS가 안전하지 않다면 결코 웹 서버의 안전을 보장할 수 없다.

#### ■ OS에 대한 최신 패치 적용 (☞ IIS 10, IIS 22)

OS 벤더사이트나 보안 취약점 정보사이트<sup>9</sup>를 주기적으로 방문하여 현재 사용하고 있는 OS에 대한 최신 취약점 정보를 얻고, 패치 등 관련된 보안대책을 신속하게 적용하도록 한다<sup>10</sup>.

#### ■ OS 취약점 점검 (☞ IIS 21)

정기적으로 취약점 점검 도구와 보안 체크리스트를 사용하여 호스트 OS의 보안 취약점을 점검한다. 점검 결과로 발견된 취약점들은 보완조치하고 조치사항은 히스토리 관리를 위해 기록해둔다.

#### ■ 웹 서버 전용 호스트로 구성 (☞ 아파치 3, IIS 3, IIS 4, IIS 5)

웹 서버의 중요도를 고려하여 가급적이면 웹 서버 전용 호스트로 구성하도록 한다. 웹 서비스 운영에 필요한 최소한의 프로그램들만 남겨두고, 불필요한 서비스들은 반드시 제거하도록 한다. 시스템 사용을 목적으로 하는 일반 사용자 계정들은 모두 삭제하거나 최소의 권한만 할당한다. 오로지 관리자만이 로그인 가능하도록 한다.

---

<sup>9</sup> OS 벤더사이트나 보안 취약점 정보사이트는 부록을 참고하도록 한다.

<sup>10</sup> 가급적이면 운영중인 서버에 직접 적용하지 말고 별도의 테스트 서버를 이용하여 충분한 테스트를 거친 후에 적용하는 것이 좋다.

[참고] Anonymous FTP 사용시 주의

만약 한 서버에서 웹 서비스와 anonymous FTP를 동시에 사용해야 한다면, 웹 서비스의 문서 디렉토리가 anonymous FTP를 통해 접근이 불가능하도록 설정해야 한다. 웹 문서에 대해 설정한 접근제어가 anonymous FTP에 의해 위배될 수 있다.

[참고] 개발도구 제거

웹 서버를 구축한 후에는 컴파일러 같은 소프트웨어 개발 도구들을 제거하도록 한다. 이러한 도구들은 공격자가 서버에 침입한 후에 익스플로잇 코드를 컴파일하는 등 권한 상승을 위해 사용될 수 있다.

## ■ 서버에 대한 접근 제어 (☞ IIS 13)

관리목적의 웹 서버 접근은 콘솔 접근만을 허용하는 것이 가장 좋다. 그것이 불가능하다면 관리자가 사용하는 PC의 IP만 접근이 가능하도록 접근제어를 수행한다.

## ■ DMZ 영역에 위치

웹 서버를 DMZ 영역에 위치시키도록 한다. 웹 서버를 방화벽에 의해서 보호 받도록 하고<sup>11</sup>, 웹 서버가 침해 당하더라도 웹 서버를 경유해서 내부 네트워크로의 침입은 불가능하도록 구성한다. 네트워크 구성에 대한 상세한 설명은 ‘제7장 웹 서버를 위한 안전한 네트워크 구성’을 참고한다.

## ■ 강력한 관리자 계정 패스워드 사용

관리자 계정의 패스워드 보안은 모든 보안의 가장 기본이다. 하지만 이런 기본이 지켜지지 않아 여전히 해킹사고가 많이 발생하고 있다. 패스워드 보안은 모든 보안의 기본이자 가장 중요한 필수 보안 사항이다.

관리자 계정 패스워드는 유추가 불가능하고 패스워드 크랙으로도 쉽게 알아낼 수 없는 강력한 패스워드를 사용하도록 한다. 패스워드는 길이가 최소한 8자 이상이고,

<sup>11</sup> 웹 서비스가 방화벽을 통해 보호받을 수는 없지만, 방화벽을 통해 같은 서버에 존재하는 다른 서비스의 취약점을 이용한 공격은 막을 수 있다.

이름이나 계정명으로 유추할 수 없는 것이어야 한다. 또한 사전에 없는 단어를 사용하도록 하고, 기호문자를 최소 한 개 이상 포함시키도록 한다.

#### ■ 파일 접근권한 설정 (☞ 아파치 1)

관리자 계정이 아닌 일반 사용자 계정으로 관리자 계정이 사용하는 파일들을 변경할 수 없도록 해야 한다. 만약 관리자 계정보다 권한이 낮은 일반 계정으로 관리자가 실행하거나 쓰기를 수행하는 파일들을 변경할 수 있다면 관리자 권한 획득이 가능하다.

## 제 2 절 웹 서버 설치 보안

### ■ 소스코드 형태의 배포본 설치

웹 서버 소프트웨어가 소스코드와 바이너리 형태로 배포되는 경우, 보안상 가장 좋은 것은 소스코드를 다운로드 받아 필요한 기능만 설치하는 것이다.

### ■ 설치시 네트워크 접속 차단

웹 서버를 설치하기 전부터 보안설정을 안전하게 끝낼 때까지 호스트의 네트워크 접속을 차단하도록 한다. 보안설정이 완전히 끝나지 않은 상태에서 웹 서버가 외부에 노출될 경우 쉽게 해킹 당할 수 있으며, 그 이후에 취해지는 보안 조치들이 의미가 없게 될 수 있다.

### ■ 웹 프로세스의 권한 제한 (☞ 아파치 3, IIS 9, IIS 19)

시스템 전체적인 관점에서 웹 프로세스가 웹 서비스 운영에 필요한 최소한의 권한만을 갖도록 제한한다. 이렇게 하여 웹사이트 방문자가 웹 서비스의 취약점을 이용해 시스템에 대한 어떤 권한도 획득할 수 없도록 한다.

### ■ 로그 파일의 보호 (☞ 아파치 1, 아파치 10)

로그 파일은 침입 혹은 침입시도 등 보안 문제점을 파악하는데 중요한 정보를 제공한다. 이러한 로그 파일이 노출, 변조 혹은 삭제되지 않도록 불필요한 접근으로부터 보호한다.

## ■ 웹 서비스 영역의 분리 (☞ 아파치 4, IIS 1)

웹 서비스 영역과 시스템(OS)영역을 분리시켜서 웹 서비스의 침해가 시스템 영역으로 확장될 가능성을 최소화한다. 웹 서버의 루트 디렉토리와 OS의 루트 디렉토리를 다르게 지정한다.

웹컨텐츠 디렉토리는 OS 시스템 디렉토리는 물론 가급적 다른 웹 서버 디렉토리 와도 분리시킨다. 또한 로그 디렉토리와 설정 디렉토리는 웹 서비스를 통해 접근이 불가능한 곳에 위치시키도록 한다.

## ■ 링크 사용금지 (☞ 아파치 6, 아파치 11)

공개 웹 컨텐츠 디렉토리 안에서 서버의 다른 디렉토리나 파일들에 접근할 수 있는 심볼릭 링크, aliases, 바로가기 등을 사용하지 않는다.

## ■ 자동 디렉토리 리스팅 사용중지 (☞ 아파치 5, IIS 18)

디렉토리 요청시 디렉토리 내에 존재하는 파일 목록을 보여주지 않도록 설정한다. 디렉토리 내에 존재하는 DB 패스워드 파일이나 웹 어플리케이션 소스 코드 등 중요한 파일들에 대해 직접 접근이 가능하면 보안상 매우 위험하다. 이를 막기 위해 자동 디렉토리 리스팅 기능의 사용을 중지시킨다.

## ■ 기본 문서 순서 주의 (☞ 아파치 7, IIS 14)

웹 서버에서는 디렉토리 요청시 기본적으로 보여지는 파일들을 지정할 수 있도록 되어 있다. 이 파일 목록을 올바르게 지정하여 기본 문서가 악의적인 목적의 다른 파일로 변경되지 않도록 한다.

## ■ 샘플 파일, 매뉴얼 파일, 임시 파일의 제거 (☞ 아파치 2, IIS 11, IIS 15)

웹 서버를 설치하면 기본적으로 설치되는 샘플 파일이나 매뉴얼 파일은 시스템 관련 정보를 노출하거나 해킹에 악용될 수 있다. 따라서 웹 서버 설치 후에 반드시 이러한 파일들을 찾아서 삭제하도록 한다.

만약 관리 등의 이유로 웹을 통해 설명문서에 접근해야 한다면 접근제어를 통해 꼭 필요한 사용자만 접근을 허용하고 그 외의 사용자들은 접근하지 못하도록 설정한다.

또한 웹 서버를 정기적으로 검사하여 임시 파일들을 삭제하도록 한다. 특히 웹 서비스의 업데이트나 유지보수시 생성되는 백업파일이나 중요한 파일 등은 작업이 끝난 후 반드시 삭제하도록 한다.

정확한 관리를 위해 폴더와 파일의 이름과 위치, 개수 등이 적혀있는 별도의 문서를 관리하는 것이 좋다. 그래서 문서에 등록되지 않은 불필요한 파일들을 점검해서 삭제하도록 한다.

## ■ 웹 서버에 대한 불필요한 정보 노출 방지 (☞ 아파치 8, IIS 7, IIS 8)

웹 서버 종류, 사용 OS, 사용자 계정 이름 등 웹 서버와 관련된 불필요한 정보가 노출되지 않도록 한다. 이러한 정보가 사소한 것처럼 보일 수 있지만, 이러한 정보를 아는 것만으로도 공격에 필요한 나머지 정보를 수집하는데 도움이 될 수 있다.

뉴스그룹이나 메일링 리스트를 통해 웹 서버 운영에 대한 질의를 할 경우에도, 조직의 네트워크와 시스템에 대한 상세정보가 유출되지 않도록 주의한다<sup>12</sup>.

---

<sup>12</sup> 실제로 한 보안 컨설팅 회사는 인터넷에 공개된 정보를 기반으로 CIA의 네트워크에 대한 상세한 지도를 그릴 수 있었다. 이 정보가 비록 기밀성이 떨어지는 정보일지는 몰라도 해킹을 위한 좋은 출발점이 될 수 있다(<http://www.computerworld.com/printthis/2002/0,4814,68961,00.html>).

## ■ 업로드 제어

불필요한 파일 업로드는 허용하지 않는다. 파일 업로드를 허용해야 한다면, 대량의 업로드로 인한 서비스 불능상태가 발생하지 않도록 한다. 또한 업로드를 허용해야 하는 파일의 종류를 지정하여 그 외 종류의 파일들은 업로드가 불가능하도록 한다. 업로드된 파일은 웹 서버에 의해 바로 처리되지 못하도록 해야 한다. 처리되기 전에 반드시 수동이나 자동으로 파일의 보안성 검토를 수행하도록 한다.

## ■ 인증과 접근제어의 사용 (☞ 아파치 14, 아파치 15, IIS 23, IIS 24)

웹 서버에서 제공하는 인증 기능과 접근제어 기능을 필요한 곳에 적절하게 활용한다. 웹 서버에서는 사용자 아이디/패스워드 기반의 인증 기능과 특정 IP나 도메인에 대한 접근제어 기능을 제공한다.

## ■ 패스워드 설정 정책 수립

웹 서버의 인증 기능을 이용하는 경우에, 유추가 불가능한 패스워드를 사용하도록 한다. 패스워드 길이와 사용 문자에 대한 최소 복잡도를 설정하도록 하고, 사용자의 개인정보나 회사 공개정보를 이용한 비밀번호 사용을 금지하도록 한다. 또한 사용자들에게 웹사이트의 패스워드와 다른 중요한 것들의 패스워드(예를 들어, 은행이나 주식 관련 비밀번호)를 다르게 사용하도록 권장한다. 웹 서버 보안이 100% 완벽할 수 없기 때문에, 이렇게 함으로써 웹 서버 침해로 인한 더 큰 피해를 막을 수 있다.

## ■ 동적 콘텐츠 실행에 대한 보안 대책 수립(☞ 아파치 1, 아파치 2, 아파치 9, 아파치 11, 아파치 12, IIS 15, IIS 17)

동적 콘텐츠 처리 엔진들은 웹 서버의 일부로서 실행되면서 웹 서버와 동일한 권한으로 실행된다. 따라서 각 엔진 사용시 발생할 수 있는 모든 보안 취약점들을 파악하고 이와 관련된 보안 기능들을 설정해야 한다.

동적 콘텐츠와 관련된 기능 중 사용하지 않는 기능들은 제거를 하고 예제 파일들은 반드시 삭제한다. 가능하다면 동적 콘텐츠가 실행될 수 있는 디렉토리를 특정 디렉토리로 제한시키도록 하고, 콘텐츠의 추가 권한은 관리자로 제한하도록 한다.



## ■ 설치 후 패치 수행

웹 서버 기본 설치 후 알려진 취약점을 바로잡기 위해 취약점 정보사이트나 벤더 사이트<sup>13</sup>를 방문해서 웹 서버와 관련된 취약점 정보를 얻고, 패치나 업그레이드를 수행한다.

## ■ 설정 파일 백업 (☞ IIS 20)

웹 서버를 인터넷에 연결하기 전에 초기 설정 파일을 백업 받아서 보관해 둔다. 또한 변경이 있을 때마다 설정 파일을 백업하도록 한다. 이렇게 하여 해킹이나 실수가 발생해도 빠르게 복구할 수 있도록 한다.

## ■ SSL/TLS 사용

보안과 기밀성이 요구되는 경우 SSL이나 TLS를 사용하도록 한다<sup>14</sup>. 대부분의 경우에 SSL/TLS는 웹 서버에서 사용할 수 있는 가장 훌륭한 인증 및 암호 방법이다.

---

<sup>13</sup> 취약점 정보사이트나 벤더 사이트는 부록을 참고하도록 한다.

<sup>14</sup> modssl 사이트([www.modssl.or.kr](http://www.modssl.or.kr)) 및 MS사의 웹 서버에서 SSL 설정 문서 (<http://www.microsoft.com/korea/msdn/library/dnnetsec/html/SecNetHT16.asp>) 참조

## 제 3 절 웹 서버 운영 보안

### ■ 파일 무결성 점검 도구의 사용

웹 서버 설정 파일, 패스워드 파일, 스크립트 파일 등을 보호하기 위해 파일 무결성 점검을 수행하도록 한다. 업그레이드나 내용 변경시 파일 무결성 체크섬(checksum)을 업데이트 한다. 체크섬을 한번만 쓰기가 가능한 보호된 미디어에 저장하고, 정기적으로 체크섬을 비교하도록 한다.

### ■ 새로운 보안 취약점에 대한 지속적인 모니터링

아파치 보안 취약점 정보사이트를 주기적으로 방문하여 취약점 정보를 얻고, 새로운 취약점이 발표되면 패치 등 권고된 보안대책을 적용하도록 한다. 아파치 보안 취약점 정보사이트는 부록을 참고하도록 한다.

### ■ 주기적인 로그 점검 및 백업 (☞ 아파치 16, 아파치 17, IIS 12, IIS 25)

웹 서버 로그와 OS 로그를 주기적으로(가급적 매일) 점검하여 침입 혹은 침입 시도, 보안 문제점 등을 파악하도록 한다.

로그 점검시 다음과 같은 사항을 주의 깊게 검토하도록 한다.

- 불법적인 로그인 시도
- 접근이 제한된 파일에 대한 접근 시도
- 존재하지 않는 파일에 대한 접근 시도
- 허용되지 않은 PUT 메소드를 이용한 파일 업로드 시도
- 부적절한 입력값이 포함된 접근 시도
- 단시간 동안 특정 IP로부터의 다량 접근 시도
- 웹 서버의 뜻하지 않은 멈춤 혹은 시작

디스크 풀(full)을 막고 로그관리를 편하게 하기 위해 웹 서버 로그를 주기적으로 백업하도록 한다.

## ■ 안전한 동적 콘텐츠의 사용

모든 동적 콘텐츠들은 충분히 테스트되고 안전하다고 판단된 후에 운영중인 웹 서버에 설치해서 사용하여야 한다.

## ■ 웹 콘텐츠 승인 절차 수립

웹 콘텐츠 공개에 대한 승인 절차를 마련하여, 웹 서버 관리자나 개발자가 임의로 웹 콘텐츠를 추가 및 변경하지 못하도록 한다. 새로운 웹 콘텐츠를 공개하기 전에 불필요한 정보나 기밀성이 요구되는 정보가 없는지 검토하도록 한다.

## ■ 웹 콘텐츠 접근 매트릭스 관리

웹 콘텐츠 접근 매트릭스를 만들어, 웹콘텐츠 디렉토리안에 어떤 폴더와 어떤 파일들이 존재하고, 각각이 누구에 의해 접근이 가능하고 접근이 제한되어야 하는지 정의하도록 한다.

## ■ 관리자 PC에 대한 보안

관리자 PC에 대한 해킹은 곧 웹 서버에 대한 해킹과 동일한 결과를 가져올 수 있다. 따라서 웹 서버 보안에 못지 않게 관리자 PC에 대한 보안에도 신경을 쓰도록 한다.

## 제 4 절 웹 어플리케이션 설계 보안

웹 어플리케이션 보안을 위해 가장 중요한 것은 어플리케이션 설계 당시부터 보안을 고려하는 것이다. 훌륭한 개발자에 의해 성능 좋은 어플리케이션이 개발되어도 기본적인 보안 의식이 없이 프로그래밍 되어 있다면 웹 어플리케이션이 안전하게 동작함을 보장할 수 없다. 웹 어플리케이션의 취약성과 이에 대한 공격 기법은 매우 다양하고 고도화 되어 있다. 그러나 현재 웹 어플리케이션 취약점 점검도구는 시스템이나 네트워크 자체에 대한 취약점을 점검하는 도구들에 비해 공격 기법을 따라가지 못하고 있다. 가장 효율적인 방어는 웹 어플리케이션 설계 당시부터 보안을 고려하는 것이다.

### ■ 유효한 사용자 입력값 (☞ PHP 1, PHP 2, PHP 3, PHP 5, PHP 6, PHP 7, PHP 10, PHP 12, ASP.NET 1, ASP.NET 6)

웹 어플리케이션을 쉽게 공격하는 방법 중 한가지는 사용자 입력값의 조작이라고 할 수 있다. 일단 모든 사용자 입력을 의심해야 하고, 값의 유효성을 판단하기 위해 다중의 보안장치를 설계해야 한다.

URL 폼, 로그인 폼, 웹 게시판 등 사용자 입력값을 처리하는 모듈에서는 값을 입력하는 순간부터 철저한 유효성 검사(Validation Check)를 수행하도록 설정한다. 그리고, 입력될 수 없는 값을 체크하기 보다는 입력할 수 있는 값을 체크하여 필터링 하도록 설계한다.

그리고 클라이언트측에서 수행하는 유효성 검사를 신뢰해서는 안된다. 반드시 서버측에서도 입력값에 대한 유효성을 점검해야 한다.

### ■ 안전한 오류 환경 설계 (☞ PHP 13, ASP.NET 7)

어플리케이션이 오류로 인해 동작할 수 없을 경우, 기본적으로 어플리케이션의 상태를 서비스 거부(denial of service) 상태로 설계해야 한다. 예를 들어, 방화벽이 동작할 수 없을 경우 입출력 인터페이스의 모든 패킷이 폐기 되는것과 비슷하다. 따라서, 오류 상태의 어플리케이션에서는 사용자 로그인이 제한되거나, 서비스 이용이 불가능하도록 설계한다.

## ■ 간단하고 강력한 통제 인터페이스 설계

사용자 편의성을 고려하지 않는다면, 보안 통제는 일상화 되기 어렵다. 따라서 어플리케이션의 사용자와 관리자를 위한 통제 인터페이스는 가능한 사용하기 쉽게 설계되어야 한다.

## ■ 다중 보안 장치 적용 (☞ PHP 14, PHP 15, PHP 17, ASP.NET 4, ASP.NET 5)

웹 어플리케이션의 안전성을 제고하기 위해서는 다중의 보안 장치가 고려되어야 한다. 현재의 웹 어플리케이션에 대한 공격 기법은 매우 다양하고 고도화 되어 있다. 따라서 일회적인 보안 장치로는 어플리케이션의 보안을 안전성을 보장하기 어렵다.

예를 들어, 일단 로그인에 성공한 사용자라 할지라도, 개인정보를 변경하거나 보안이 요구되는 서비스를 이용할 경우 패스워드를 재확인 하는 등 추가적인 인증과정을 거치도록 설계한다.

## ■ 최소한의 권한 (☞ PHP 16, ASP.NET 2, JSP 2)

어플리케이션은 최소한의 권한으로 작동할 수 있도록 설계한다. 최소한의 권한으로 운영되는 어플리케이션은 공격 받을 경우 어플리케이션의 손상을 제한할 수 있기 때문이다.

또한, 사용자의 경우 허가된 서비스만 사용 가능하도록 접근을 제한하도록 한다.

## ■ 권한 분리 (☞ PHP 16, ASP.NET 2, ASP.NET 3, JSP 3)

프로세스나 자원에 대한 접근은 두가지 이상의 조건에 의존해 판단해야 한다. 그 래야 한가지 통제장치가 무력화 되어도 시스템에 대한 불법적인 접근을 차단 할 수 있기 때문이다.

웹 사이트 또한 제한영역과 공개영역으로 구분하여 설계한다.

## ■ 검증된 암호 모듈 사용(☞ PHP 7, PHP 8, PHP 11, PHP 14, PHP 15, JSP 1)

검증된 암호 알고리즘이라 할지라도 자체적으로 구현할 경우, 예상할 수 있는 취약성 검토가 어렵고 발견된 취약성에 대해서도 개발 지식이 있는 사람만이 패치 개발이 가능하기 때문에 가급적 공개적으로 보안성이 검증된 암호 모듈을 사용하도록 한다.

## ■ 최소한의 공통 메커니즘 (☞ PHP 9)

어플리케이션의 공유 메커니즘, 예를 들어 /tmp 나 /temp, /var/tmp 등의 사용을 최소화 한다. 공유 메커니즘은 의도하지 않은 상호작용을 포함하기 때문에 위험 할 수 있다.

세션의 경우, 공유 메커니즘을 통해 타인의 세션 ID 획득이 가능하기 때문에 설계 시 이를 피할 수 있는 방법이 고려되어야 한다.

이 페이지는 빈 페이지입니다.

## 제 4 장 아파치 보안설정 방법<sup>15</sup>

### 제 1 절 서버보안 설정

#### [아파치 1] 주요 디렉토리 및 파일에 대한 접근권한 설정

만일 root 만이 실행하거나 쓸 수 있는 파일을 root가 아닌 사용자가 수정할 수 있다면 시스템은 root 권한을 도용 당할 수 있다. 가령 누군가 httpd 실행파일을 다른 것으로 대체하고, 웹 서버 관리자가 다음에 이 파일을 실행한다면 의도하지 않은 임의의 코드가 실행될 수도 있다. 또한 root가 아닌 다른 사용자가 logs 디렉토리에 쓰기 권한이 있다면 로그파일을 다른 시스템 파일에 심블릭 링크 걸어서 root 권한으로 시스템 파일에 임의의 데이터를 덮어 쓰게 할 수도 있다. 그리고, 로그 파일 내용도 임의로 수정이 가능할 것이다. 따라서 주요 디렉토리와 파일을 보호하기 위해서 root 만이 쓰기가 가능하도록 설정할 필요가 있다.

주요 디렉토리 및 파일에 대한 접근권한을 가급적 [표 4-1], [표 4-2]와 같이 설정하도록 추천한다.

[표 4-1] 아파치 주요 디렉토리에 대한 접근권한

디렉토리	소유자	그룹	퍼미션
ServerRoot <sup>16</sup>	root	root	755
bin	root	root	755
conf	root	root	755
logs	root	root	755
cgi-bin	root	root	751
DocumentRoot <sup>17</sup>	root	root	775

※ /, /usr, /usr/local 도 root 만이 쓰기가 가능해야 한다.

<sup>15</sup> 본 장의 내용은 Apache 1.3.X Unix 버전을 기준으로 작성하였으며, 실제로 HancornLinux release 2.0에 Apache 1.3.27 Unix 버전을 설치하여 테스트하였다.

<sup>16</sup> ServerRoot는 아파치 서버가 활동하는 디렉토리로 httpd.conf 파일에서 설정된다.

<sup>17</sup> DocumentRoot는 아파치가 서비스할 파일들이 존재하는 디렉토리로 httpd.conf 파일에서 설정된다.



[표 4-2] 아파치 주요 파일에 대한 접근권한

파일	소유자	그룹	퍼미션
httpd	root	root	511
인증패스워드파일	root	nobody <sup>18</sup>	640
httpd.conf	root	nobody	640

[참고] 아파치 보안 설정시 사용될 수 있는 httpd 명령어의 몇 가지 기본적인 옵션들은 다음과 같다.

옵션	설명
-v	현재 httpd의 버전 출력
-V	컴파일시 설정된 값들 출력
-l	컴파일된 모든 모듈을 나열
-L	설정파일에서 사용 가능한 지시어 나열
-t	설정파일에 대해 문법(syntax) 체크 수행
-d ServerRoot	httpd.conf 파일에 설정되어 있는 ServerRoot 값을 사용하지 않고 커맨드 상에서 직접 ServerRoot를 지정
-f configfile	컴파일 타임에 결정된 설정 파일을 사용하지 않고 커맨드 상에서 직접 설정파일을 지정

## [아파치 2] 샘플 파일, 매뉴얼 파일, 임시 파일의 제거

아파치를 설치하면 htdocs 디렉토리와 cgi-bin 디렉토리가 디폴트로 설치된다. 이 디렉토리내의 파일들은 시스템 관련 정보를 노출하거나 해킹에 악용될 수 있어 삭제하도록 권고한다.

htdocs 디렉토리는 매뉴얼 파일을 담고 있는데, 매뉴얼 파일은 시스템에 대한 정보를 포함하고 있어서 해킹에 도움이 되는 정보를 제공할 수 있다. 따라서 아파치 설치후 매뉴얼 파일을 웹 서버에서 삭제하도록 한다. 만약 관리 등의 이유로 웹을 통해 매뉴얼 파일에 접근해야 한다면, 접근제어를 통해 꼭 필요한 사용자만 접근을 허용하고 외부 사용자들은 접근하지 못하도록 설정한다.

<sup>18</sup> nobody에 대한 설명은 『아파치 3』 웹 서버 실행을 위한 최소한의 권한을 가진 사용자 계정 생성』 참조

cgi-bin 디렉토리에는 샘플 스크립트 파일들이 설치되는데, 이 파일 중에는 시스템 관련 정보를 노출하거나 해킹에 악용될 수 있는 것들이 있다. 아파치 설치 후에 반드시 이러한 불필요한 파일들을 찾아서 삭제하도록 한다.

이외에도 업데이트나 유지보수시 생성되는 백업파일이나 중요 파일 등은 작업이 끝난 후 반드시 삭제하도록 한다.

## 제 2 절 메인 설정 파일(httpd.conf<sup>19</sup>)

### [아파치 3] 웹 서버 실행을 위한 최소한의 권한을 가진 사용자 계정 생성

아파치 웹 서버가 처음 실행될 때는 root 계정으로 실행되지만, 요청을 처리할 때는 웹 서버 전용 계정으로 실행되도록 설정하고 웹 서버 전용 계정은 권한을 최소화한다. 이렇게 설정하면 아파치 서버가 침해를 당해도 피해를 최소화 할 수 있다.

이를 위해 아파치 웹 서버를 위한 전용 사용자 아이디와 그룹 아이디를 생성한다. 일반적으로 웹 서버를 위한 사용자 아이디와 그룹 아이디로 nobody라는 이름을 사용하는데 반드시 이름이 nobody일 필요는 없다. 다만 다음과 같이 이 계정은 로그인 불가능하도록 설정한다.

```
          /etc/passwd  
nobody :x :99 :99 :Nobody :/ :
```

```
          /etc/shadow  
nobody :* : 11900 :0 :99999 :7 : : :
```

httpd.conf 파일에서 User 지시자와 Group 지시자를 이용하여 아파치 웹 서버가 요청을 처리할 때의 사용자 아이디와 그룹 아이디를 설정한다.

```
          httpd.conf  
User nobody  
Group nobody
```

---

<sup>19</sup> 아파치는 대부분의 보안 설정을 httpd.conf 파일에서 한다.

#### [아파치 4] DocumentRoot 설정

DocumentRoot는 웹 서버가 서비스할 파일들이 저장되어 있는 디렉토리를 설정하는 지시자이다. 만약 DocumentRoot가 /etc로 지정되면, 외부에서 /etc 디렉토리외 그 서브디렉토리에 존재하는 파일들에 접근할 수 있다. 따라서 외부에서 웹 콘텐츠만 접근할 수 있도록 DocumentRoot 디렉토리를 정확하게 설정하는 것이 중요하다.

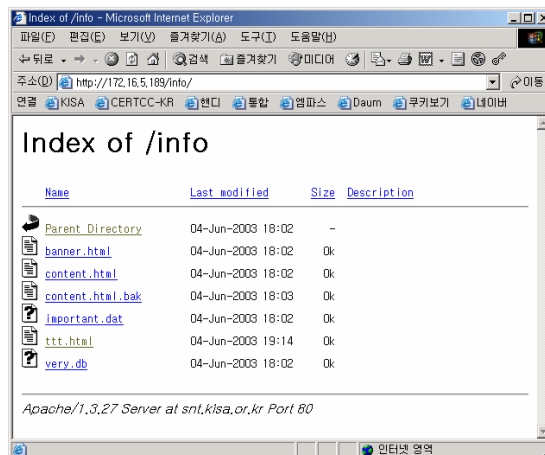
추가적으로 웹 콘텐츠 파일과 웹 어플리케이션 파일들을 분리하여 저장하고 웹 콘텐츠에 대한 요청만을 허락하도록 한다. 웹 콘텐츠가 저장되는 디렉토리는 ServerRoot나 chroot와는 별도의 파티션이나 하드드라이브를 사용하는 것이 좋다.

#### httpd.conf

```
#  
# DocumentRoot: The directory out of which you will server your  
# documents. By default, all requests are taken from this directory, but  
# symbolic links and aliases may be used to point to other locations.  
#  
DocumentRoot "/home/httpd/html/"
```

#### [아파치 5] 디렉토리 리스팅

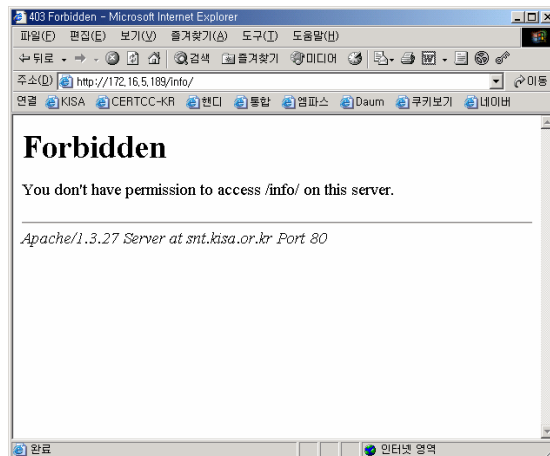
디렉토리 리스팅은 디렉토리에 대한 요청시 그 디렉토리에 DirectoryIndex(예, index.html)가 없을 경우, 디렉토리 내에 존재하는 모든 파일의 목록을 보여주는 것을 말한다.



디렉토리 리스팅이 허용되면 외부에서 디렉토리 내의 모든 파일에 접근이 가능하여 백업 파일이나 소스 파일 등 공개되어서는 안되는 중요한 파일들이 노출될 수 있다. 따라서 다음과 같은 방법으로 디렉토리 리스팅이 불가능하도록 설정한다. Directory 섹션의 Options 지시자에서 Indexes가 설정되어 있다면 제거한다.

```
httpd.conf
<Directory "DocumentRoot디렉토리" >
    Options Indexes, ...
    ... ..
</Directory>
```

Indexes를 제거하면 DirectoryIndex가 존재하지 않아도 디렉토리 리스팅을 하지 않고, 다음 그림처럼 접근불가 메시지를 보여준다.



## [아파치 6] FollowSymLinks

Options 지시자에서 다음과 같이 FollowSymLinks를 설정하면, 해당 디렉토리 내에 존재하는 심볼릭 링크를 따라갈 수 있다. FollowSymLinks는 ServerRoot와 DocumentRoot의 설정과는 무관하게, 중요 시스템 디렉토리로의 접근이 가능해지기 때문에 주의를 요한다.

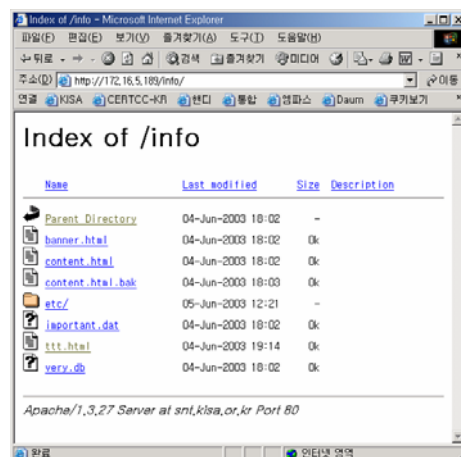
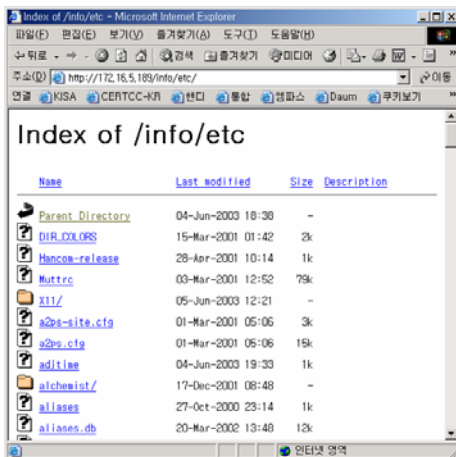
```

httpd.conf
ServerRoot "/usr/local/apache"

DocumentRoot "/home/httpd/html"

<Directory "/home/httpd/html/info" >
    Options FollowSymLinks, ...
    ... ..
</Directory>
    
```

위와 같이 설정된 상태에서 info 디렉토리내에 /etc 디렉토리에 대한 심볼릭 링크를 생성하면, 다음 화면과 같이 /etc 디렉토리에 접근이 가능해진다. 이것은 심각한 보안 문제를 야기 할 수 있다. 따라서 가급적 FollowSymLinks를 사용하지 말고 삭제할 것을 권고한다.



## [아파치 7] DirectoryIndex 설정

DirectoryIndex는 사용자가 디렉토리 이름 끝에 슬래쉬(/)를 붙여서 디렉토리의 인덱스를 요구한 경우, 리턴할 자원의 목록을 지정하는 지시자 이다.

예를 들어 DirectoryIndex 지시자가 다음과 같이 설정되어 있고 클라이언트가 디렉토리 인덱스 `http://your.server.com/info/` 를 요구한 경우, 아파치는 제일 먼저 `info` 디렉토리에서 `index.cgi` 파일을 찾아서 리턴해준다. 그 파일이 없으면 `index.shtml` 파일, `index.html` 순으로 찾게 된다.

```
httpd.conf
DirectoryIndex index.cgi index.shtml index.html
```

그런데 여기서 리스트의 순서에 주의해야 한다. `index.html` 파일이 존재한다 하더라도 공격자가 `index.cgi` 파일을 업로드 한다면, `index.html` 대신 `index.cgi` 파일이 처리될 것이다.

따라서 DirectoryIndex의 목록이 적절하게 배열되어 있는지 점검하도록 한다.

## [아파치 8] 아파치 버전 숨기기

외부에서 아파치 서버에 접속해서 아파치 버전 및 OS의 종류와 버전을 알아내지 못하도록 `ServerTokens`와 `ServerSignature`를 다음과 같이 설정한다.

```
httpd.conf
ServerTokens ProductOnly

ServerSignature off
```

`ServerTokens`는 클라이언트에게 보낼 응답 헤더 필드에 OS 타입과 `compiled-in` 모듈에 대한 정보를 포함할지를 설정하는 지시자이다. `ServerTokens`는 디폴트로 `Full`로 설정되어 있는데, 설정값에 따라 노출되는 정보는 다음과 같다.

설정값	서버가 전송하는 정보
ServerTokens Prod[uctOnly]	Server: Apache
ServerTokens Min[imal]	Server: Apache/1.3.0
ServerTokens OS	Server: Apache/1.3.0 (Unix)
ServerTokens Full (혹은 설정되지 않음)	Server: Apache/1.3.0 (Unix) PHP/3.0 MyMod/1.2

ServerTokens를 Full로 설정하고 telnet 접속을 시도하면 다음 그림과 같이 아파치의 버전과 OS 종류 정보가 노출된다.

```

root@snt: /usr/local/apache/bin
[root@snt bin]# telnet localhost 80
Trying 127.0.0.1...
Connected to localhost.localdomain.
Escape character is '^'.
GET HTTP/1.1
HTTP/1.1 400 Bad Request
Date: Thu, 17 Oct 2002 01:58:41 GMT
Server: Apache/1.3.27 (Unix)
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<HTML><HEAD>
<TITLE>400 Bad Request</TITLE>
</HEAD><BODY>
<H1>Bad Request</H1>
Your browser sent a request that this server could not understand.<P>
The request line contained invalid characters following the protocol string.<P>
<P>
</BODY></HTML>
Connection closed by foreign host.
[root@snt bin]#
[영어][완성][두벌식]

```

ServerTokens를 ProductOnly로 설정하면 다음 그림처럼 웹 서버 종류가 아파치라는 정보만 보여주게 된다.

```

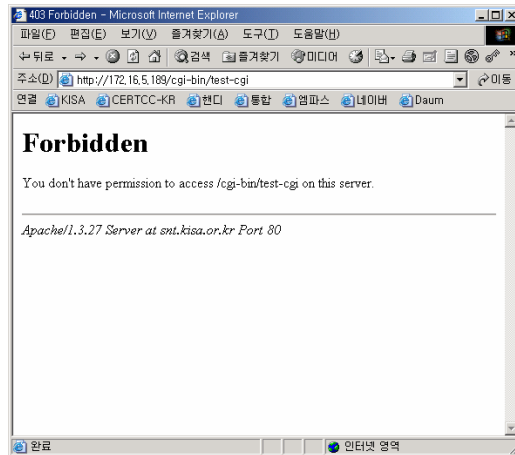
root@snt: /usr/local/apache/bin
[root@snt bin]# telnet localhost 80
Trying 127.0.0.1...
Connected to localhost.localdomain.
Escape character is '^'.
GET HTTP/1.1
HTTP/1.1 400 Bad Request
Date: Thu, 17 Oct 2002 02:03:03 GMT
Server: Apache
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<HTML><HEAD>
<TITLE>400 Bad Request</TITLE>
</HEAD><BODY>
<H1>Bad Request</H1>
Your browser sent a request that this server could not understand.<P>
The request line contained invalid characters following the protocol string.<P>
<P>
</BODY></HTML>
Connection closed by foreign host.
[root@snt bin]#
[영어][완성][두벌식]

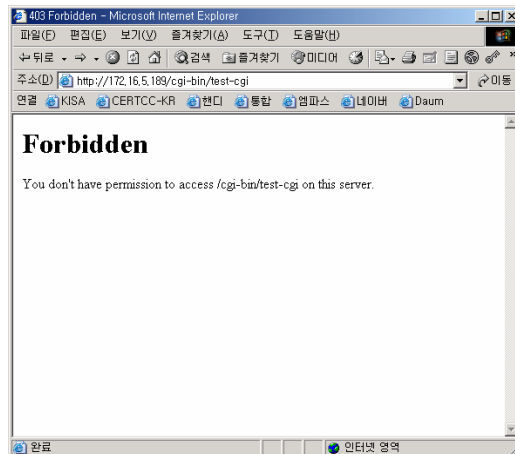
```



ServerSignature는 서버가 만든 문서의 꼬리말(trailing footer)을 설정하다. 이러한 꼬리말이 필요한 이유는 프락시 체인 형성시, 사용자가 실제로 어느 서버가 에러 메시지를 리턴했는지 알 수 있도록 하기 위해서다. 이 값이 on으로 설정되어 있으면 다음 그림과 같이 아파치 버전과 가상호스트의 ServerName, ServerAdmin이 노출된다.



ServerSignature를 off로 설정하면 다음 그림과 같이 아무런 정보를 보여주지 않는다.



## [아파치 9] Server Side Includes

SSI는 작은 양의 동적 콘텐츠를 생성시킬 때 사용하는 기능으로 httpd.conf 나 .htaccess 파일에서 다음과 같은 지시자 설정으로 사용이 가능하다.

```
httpd.conf
<Directory "/home/httpd/html/info" >
    Options +Includes,...
    ... ..
</Directory>
```

SSI에는 다음과 같은 몇 가지 보안 위험이 존재한다.

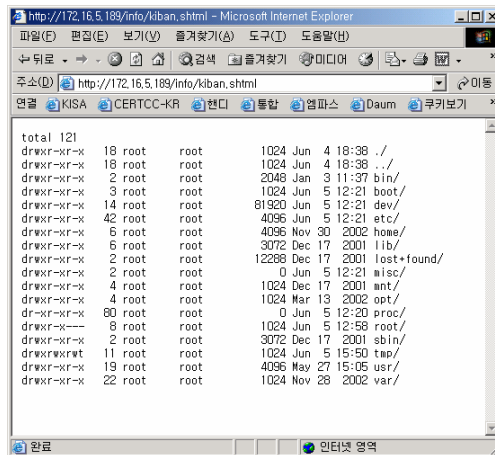
### ① 부하 증가

SSI가 enable되면 해당 디렉토리 안의 모든 파일들은 파일안에 SSI 지시자가 있건 없건 반드시 아파치에 의해 파싱(parsing)되어야 한다. 이것은 서버의 부하를 크게 증가시킬 수 있다.

### ② CGI 스크립트의 위험과 동일한 위험

“exec cmd” 요소를 사용하면 SSI-enabled 문서를 통해 CGI 스크립트나 임의의 명령어를 실행시킬 수 있다. 예를 들어 다음과 같은 코드가 삽입된 웹 문서를 만든 후, 그 파일을 요청하면 다음과 같은 결과를 얻을 수 있다.

```
<pre>
<!--#exec cmd=" ls -laF /" -->
</pre>
```



SSI 파일의 보안을 확장시키는 방법이 몇 가지 있다.

- ① .html이나 .htm 확장자를 가진 파일에 대해 SSI를 enable시키는 것은 매우 위험하다. 이것은 특히 공유되거나 트래픽이 높은 서버 환경에서 그렇다. SSI-enabled 파일은 반드시 별도의 확장자를 가지도록 한다. 이것은 서버 부하를 줄여주고 위험관리를 더 쉽게 한다. 다음 예제는 확장자가 .shtml인 파일들만 SSI 과성되도록 설정하는 것이다.

```

httpd.conf
AddType text/html .shtml
AddHandler server-parsed .shtml

```

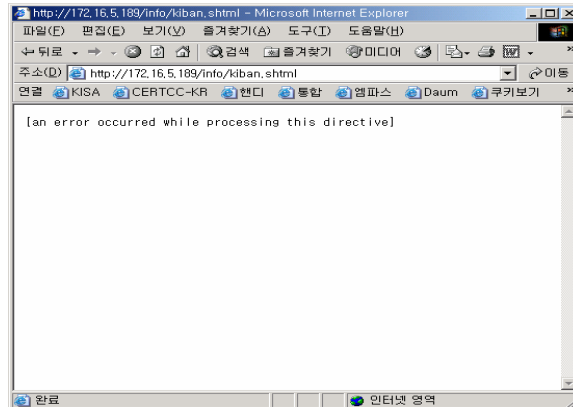
- ② SSI 페이지에서 스크립트나 프로그램의 실행을 불가능하게 한다. 이렇게 하기 위해서는 Options 지시자에서 Includes를 IncludesNOEXEC로 교체해야 한다.

```

httpd.conf
<Directory "/home/httpd/html/info" >
    Options +IncludesNOEXEC,...
    ... ..
</Directory>

```

이렇게 설정한 상태에서 위에서와 동일한 파일을 요구하면 다음과 같은 에러 메시지가 뜬다.



IncludesNOEXEC가 설정되어 있어도, ScriptAlias 지시자에 의해 지정된 디렉토리에 존재하는 CGI 스크립트를 다음과 같이 include하면 스크립트를 실행시킬 수 있다. 따라서 사용자가 웹 페이지의 내용을 편집할 수 있는 곳에서는 반드시 SSI 기능을 사용중지 시키도록 한다.

```
<pre>  
<!--#include virtual="스크립트파일" -->  
</pre>
```

- ③ SSI 파일의 위험을 제한시키기 위해 suexec<sup>20</sup>를 사용할 수 있다. 일반적으로 CGI 나 SSI 프로그램은 웹 서버 실행 계정과 동일한 계정으로 실행이 된다. Suexec 를 사용하면 웹 서버 실행 계정과는 다른 계정으로 CGI와 SSI 프로그램을 실행시킬 수 있다.

<sup>20</sup> <http://httpd.apache.org/docs/suexec.html> 참조

## [아파치 10] 로그 파일의 위치

로그 파일의 위치가 웹 서버 루트 디렉토리 밑에 있지 않도록 한다. 만약 로그 파일이 웹 서버 루트 디렉토리에 위치해 있으면 로그 파일이 노출될 수 있고, 웹 서버 침해시 로그에 대한 변조나 삭제의 위험이 있다. 따라서 로그 파일의 위치를 웹 서버 루트 디렉토리 외부에 위치시키는 것이 좋다.

이 가이드를 위한 테스트 웹 서버에서는 로그 파일의 위치를 다음과 같이 설정하였으니 참고하기 바란다.

```
httpd.conf
ServerRoot /usr/local/apache
... ..
ErrorLog /var/log/httpd/error_log
... ..
CustomLog /var/log/httpd/access_log common
```

## [아파치 11] Alias와 ScriptAlias 지시자 설정 점검

Alias는 특정 디렉토리에 매핑되는 URL prefix를 정의하는 것이고, ScriptAlias는 Alias의 기능에 그 디렉토리의 모든 파일들이 CGI 프로그램으로 간주되어 실행된다는 의미가 추가된다.

Alias와 ScriptAlias는 보통 DocumentRoot 디렉토리 밖에 있는 디렉토리에 접근하기 위해 사용된다. 따라서 불필요하게 Alias와 ScriptAlias를 설정하는 것은 보안에 큰 위험이 될 수 있으므로 주의를 요한다. 아파치 설치 후 httpd.conf에 설정된 모든 Alias와 ScriptAlias를 검토해서 사용하지 않는 것들은 반드시 제거하도록 한다.

## [아파치 12] CGI 스크립트 실행 제한

지정된 디렉토리에서만 CGI 스크립트가 실행될 수 있도록 제한한다. cgi-bin과 같은 CGI 바이너리 디렉토리를 지정해서, 그 디렉토리에 존재하는 CGI 스크립트만 실행되도록 해야 한다.

특정 디렉토리에 있는 CGI 스크립트 파일이 실행되도록 설정하는 방법은 두 가지이다. 한가지는 ScriptAlias 지시자를 설정하는 것이고, 다른 한가지는 Options 지시자에 ExecCGI를 설정하는 것이다.

ScriptAlias는 앞서 설명한 바와 같이 지정된 디렉토리에 저장된 모든 파일들을 CGI 스크립트로 간주하여 실행되도록 한다. 설정방법은 다음과 같다. 다음과 같이 설정한 경우 www.domain.com/cgi-bin/test-cgi 입력시 /home/httpd/cgi-bin/test-cgi 파일을 실행시켜 결과를 리턴한다.

```
httpd.conf
ScriptAlias /cgi-bin/ "/home/httpd/cgi-bin/"
```

Options지시자에 ExecCGI를 설정하는 방법은 다음과 같다. CGI스크립트의 실행을 허용할 디렉토리의 디렉토리 섹션에서 Options 지시자 옆에 ExecCGI를 명시하고, AddHandler로 CGI스크립트로 처리할 파일의 확장자를 지정해주면 된다.

```
httpd.conf
<Directory "/home/httpd/html/cgi-bin" >
    Options ExecCGI ...
    ... ..
</Directory>
... ..
AddHandler cgi-script .cgi
```

운영중인 웹서버의 경우 설정 파일을 분석하여 CGI스크립트가 실행될 필요가 없는 디렉토리에 대해 위와 같은 방법으로 CGI스크립트가 실행 가능하도록 설정되어 있는지 점검해볼 필요가 있다.

## 제 3 절 .htaccess 파일

### [아파치 13] .htaccess 파일 사용시 유의점

.htaccess 파일은 테스트와 디버깅용으로 사용하고, 운영시에는 .htaccess 파일을 삭제한 후 지시자들을 http.conf 파일의 <Directory> 컨테이너에 넣도록 한다.

.htaccess 파일을 많이 사용할수록 보안 설정시 고려해야 하는 파일의 수가 늘어나는 것이기 때문에 그만큼 보안관리에 허점이 생길 수 있다. 또한 .htaccess 파일을 많이 사용할수록 아파치에게 부하를 많이 주게 된다. 아파치에 대한 일관성 있는 보안관리를 수행하기 위해 httpd.conf 파일만 사용하도록 한다.

.htaccess 파일을 사용하지 못하도록 httpd.conf 파일에서 다음과 같이 설정할 수 있다.

```
httpd.conf
<Directory />
    AllowOverride None
    .....
</Directory>
```

이 설정은 디렉토리별로 .htaccess 파일이 enable된 곳을 제외하고 기본적으로 모든 디렉토리에서 .htaccess 파일의 사용을 막는다. 다른 디렉토리 섹션에서는 .htaccess 파일이 enable되지 않도록 AllowOverride를 사용하지 않는다.

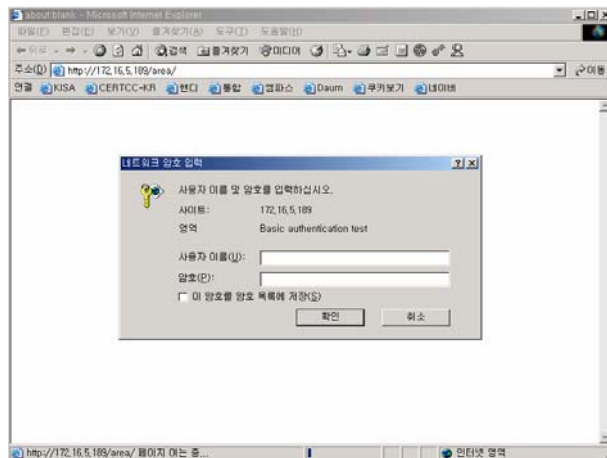
[참고] 만약 여러 명의 사용자가 하나의 웹 서버를 사용하고 있고 따라서 불가피하게 시스템의 일반 사용자에게 .htaccess의 사용을 허용해야 할 경우, 전체적으로 override를 허용하지 말고 반드시 필요한 override만을 허용하도록 한다. override 카테고리를 허용하기 전에 허용함으로써 미치는 영향을 반드시 검토하도록 한다. Override 카테고리가 세분화되어 있지 않기 때문에, 예상치 않게도 사용자에게 필요 이상의 지시자들을 허용하게 될 수 있다. 또한 .htaccess 파일들이 적절하게 보호가 되었는지 주기적으로 점검해야 한다. 그러나 철저한 보안을 생각한다면 .htaccess 파일을 사용하지 않도록 권고한다.

## 제 4 절 인증 및 접근제어

### [아파치 14] 인증

아파치에서는 3가지 인증방법을 사용할 수 있는데, Basic authentication, Digest authentication, Database authentication이 있다. 여기에서는 아파치 인증 절차와 각 인증방법의 사용법 및 장단점에 대해 설명하도록 한다.

아파치에서 일반적인 인증 절차는 다음과 같다. 클라이언트가 특정 자원을 요청하고 그 자원이 인증 기능을 통해서 보호되었다면, 아파치는 클라이언트의 요청에 대한 응답으로 401 Authentication Required 헤더를 보내서 클라이언트에게 사용자 증명서(credential)를 요구한다. 클라이언트는 401 response 헤더를 받으면, 다음 그림과 같이 사용자에게 아이디와 패스워드를 요구한다. 사용자가 아이디와 패스워드를 입력하면, 클라이언트는 그 값을 서버에게 전송한다. 그러면 서버는 아이디와 패스워드가 정확한지 점검한 후에, 요청된 자원을 클라이언트에게 전달한다.



HTTP 프로토콜이 stateless이기 때문에, 비록 동일한 요청이 같은 클라이언트로 부터 온다 하더라도 모든 요청은 위와 같은 절차에 의해 처리가 된다. 사용자는 다 행히도 세션당 한번만 아이디와 패스워드를 입력하면 그 뒤로는 브라우저가 알아서 처리해준다. 즉 사용자는 새로운 브라우저 창에서 동일한 사이트에 대해 단 한번만 아이디와 패스워드를 입력하면 된다.



서버는 인증을 요구하는 401 response와 함께 웹사이트의 보호 영역을 식별하는 인증영역 식별자를 클라이언트에게 전송한다. 인증영역 식별자는 위 그림에서 보는 바와 같이 팝업 박스에 (영역 : “Basic authentication test”) 나타난다. 클라이언트 브라우저는 이 인증영역 식별자를 사용자로부터 입력받은 아이디 및 패스워드와 함께 보관한다. 그래서 동일한 영역에 대한 요청이 다시 발생하게 되면 사용자에게 재입력을 요구하지 않고 저장된 아이디와 패스워드를 전송한다. 보통 이러한 캐싱(caching)은 현재의 브라우저 세션을 위한 것인데, 일부 브라우저에서는 영구적으로 저장하여 사용자에게 두 번 다시 입력을 요구하지 않는 경우도 있다.

## ■ Basic authentication

Basic authentication은 가장 간단한 인증 방법으로 오랫동안 사용되어 왔다. 그러나, 이 방법은 보안성이 크게 떨어지기 때문에 최근에는 다른 인증 방법들이 사용되고 있다.

Basic authentication을 사용하는 방법은 다음과 같다

### ① 패스워드 파일 생성

이 파일은 중요한 정보가 들어있기 때문에, 문서 디렉토리 밖에 저장하여 외부에서 접근이 불가능하도록 해야 한다. 비록 내용이 암호화되어 있더라도 우선 접근이 가능하면 패스워드를 알아낼 수 있는 기회가 주어질 수 있기 때문이다.

보통 사람들은 패스워드를 엄격하게 사용하지 않기 때문에 웹사이트 인증 패스워드와 동일한 패스워드를 은행 계좌에도 사용할 수 있다. 따라서 패스워드 파일의 보호를 허술히 하면, 비록 웹사이트의 콘텐츠가 중요하지 않다 하더라도 심각한 보안 위협이 될 가능성이 있는 것이다. 사용자들로 하여금 웹사이트의 패스워드와 다른 중요한 것들의 패스워드를 다르게 사용하도록 권장한다.

#### <새로운 패스워드 파일의 생성>

```
$ htpasswd -c /usr/local/apache/passwd/passwords jun
New password: “패스워드 첫 입력”
Re-type new password: “패스워드 두번째 입력”
Adding password for user jun
$
```

### <기존 패스워드 파일에 사용자 추가>

```
$htpasswd /usr/local/apache/passwd/passwords jun
New password: "패스워드 첫 입력"
Re-type new password: "패스워드 두번째 입력"
Adding password for user jun
$
```

패스워드가 암호화되어 저장되었다 하더라도, 반드시 패스워드 파일을 안전한 곳에 저장하고 최소의 퍼미션만 설정하도록 한다. 만약 웹 서버가 사용자 nobody, 그룹 nobody로 실행된다면, 웹 서버만 그 파일을 읽을 수 있도록 하고, root 만 쓰기를 할 수 있도록 퍼미션을 설정하도록 한다.

```
$chown root.nogroup /usr/local/apache/passwd/passwords
$chmod 640 /usr/local/apache/passwd/passwords
$
```

### ② 패스워드 파일 사용을 위해 configuration 설정

패스워드 파일이 생성되면, 아파치에게 이 패스워드 파일을 사용해서 사용자인증을 수행하도록 알려주어야 한다. 이 설정은 다음 지시자를 사용한다.

- AuthType : 인증 타입. 여기서는 Basic
- AuthName : 인증영역 식별자(realm)
- AuthUserFile : 패스워드 파일의 위치
- AuthGroupFile : 그룹 파일의 위치 (선택사항)
- Require : 허가를 주기 위해 만족시켜야 하는 요구사항들

httpd.conf파일의 <Directory> 섹션이나 .htaccess 파일에서 이 지시자들을 사용하여 설정하면 되는데, 다음은 httpd.conf 파일에서 area라는 디렉토리에 대해 Basic authentication을 설정한 예이다.

httpd.conf

```
<Directory "/home/httpd/html/area">
    .....
    AuthType Basic
    AuthName "Basic authentication test"
    AuthUserFile /usr/local/apache/passwd/passwords
    Require user jun
    .....
</Directory>
```

위 예제에서 AuthType의 Basic은 Basic authentication을 사용한다는 것을 나타내고, AuthName의 "Basic authentication test"는 인증영역 식별자로서 패스워드 팝업창에 표시된다. 또한 패스워드 파일의 위치는 /usr/local/apache/passwd/passwords 이고, 오직 아이디가 jun 인 사용자만이 패스워드가 일치할 경우 접근이 허가된다.

어떤 사용자든 아이디와 패스워드가 일치할 경우 로그인을 허용하려면 다음과 같이 "valid-user" 키워드를 사용한다.

httpd.conf

```
<Directory "/home/httpd/html/area">
    .....
    AuthType Basic
    AuthName "Basic authentication test"
    AuthUserFile /usr/local/apache/passwd/passwords
    Require valid-user
    .....
</Directory>
```

httpd.conf 파일을 변경한 경우 반드시 아파치 서버를 재시작 해야 한다.

### ③ 그룹 파일 생성(선택사항)

하나의 자원에 대한 접근 권한을 가진 사람들이 다수인 경우, 인증그룹을 사용해서 그룹단위로 사용자를 관리할 수 있다. 이렇게 하면 멤버의 추가 삭제시 그룹파일만 변경하면 되기 때문에 httpd.conf 파일을 변경할 필요도 없고, 따라서 매번 아파치를 재시작할 필요도 없다.

그룹단위로 사용자를 관리하기 위해서는 우선 그룹 파일을 생성하고, httpd.conf 파일의 Directory 섹션을 수정해주어야 한다.

그룹 파일은 다음과 같은 포맷을 갖는다. 이 예제에서는 두 개의 그룹 staff와 employee를 정의하고 각 그룹에 속한 사용자 아이디를 열거하였다.

```
①      ②  
staff: jun prmkim ksn  
employee: yschoi kikim wgjoo
```

① 그룹 이름    ② 사용자 아이디

이렇게 그룹파일을 만든 후, httpd.conf 파일에서 해당 디렉토리 섹션을 다음과 같이 수정한다. AuthGroupFile 지시자를 사용해서 그룹파일의 위치를 지정해주고, Require에 인증을 허용할 그룹명을 지정해준다. 다음 예제는 staff 그룹에 속한 사용자만이 /home/httpd/html/area 디렉토리 내의 자원에 접근할 수 있도록 설정한 것이다.

```
httpd.conf  
<Directory "/home/httpd/html/area">  
.....  
AuthType Basic  
AuthName "KISA staff"  
AuthUserFile /usr/local/apache/passwd/passwords  
AuthGroupFile /usr/local/apache/passwd/groups  
Require group staff  
.....  
</Directory>
```

이렇게 설정하면 인증 프로세스는 한 단계 더 많아 진다. 요청을 받으면 입력한 사용자 이름이 그룹에 존재하는지 알아내기 위해 그룹 파일을 먼저 점검한다. 그 다음에 사용자가 패스워드 파일에 존재하는지 점검하고, 있다면 패스워드가 일치하는지 점검한다.

Basic authentication의 경우 아이디와 패스워드를 브라우저가 캐싱하고 있기 때문에, 사용자에게 사용이 끝난 후 반드시 브라우저를 종료하도록 주지시킨다.

Basic authentication은 안전하지 않은 인증 방법이다. 패스워드가 암호화된 포맷으로 저장되지만, 클라이언트에서 서버로 네트워크를 따라 전송될 때는 암호화되지 않는다. 이것은 패킷 스니핑을 하면 누구나 사용자 이름과 패스워드를 알아낼 수 있다는 의미이다.

또한 사용자 아이디와 패스워드가 처음 한번만 전송되는 것이 아니라 자원에 대한 요청이 있을 때마다 전송되기 때문에 아이디와 패스워드를 스니핑하기가 매우 쉽다.

사용자 아이디와 패스워드뿐만 아니라 컨텐츠 자체도 암호화되지 않은 채로 네트워크에 전송되기 때문에, 사용자 이름과 패스워드를 몰라도 웹사이트의 중요한 정보에 접근할 수 있게 된다.

따라서 좀 더 완벽한 보안이 요구되는 곳이라면 basic authentication을 사용하지 않는 것이 좋다.

## ■ Digest authentication

Digest authentication<sup>21</sup>은 웹 컨텐츠를 보호하기 위한 대안적 방법을 제공한다. Digest authentication은 mod\_auth\_digest 모듈에 의해 구현되었다. 아파치 1.3.27 버전을 기본적으로 설치하면 mod\_auth\_digest가 포함되어 있지 않기 때문에 이 모듈을 추가 해야 한다. Digest authentication의 특징은 패스워드가 MD5 digest 형태로 전송된다는 점이다.

---

<sup>21</sup> mod\_auth\_digest는 mod\_digest의 업데이트 버전이다. mod\_auth\_digest는 현재 Experimental 상태로 아직 완전하게 테스트되지 않았다. mod\_auth\_digest는 mod\_digest와 일부 지시자를 공유하기 때문에, mod\_auth\_digest를 사용할 경우 mod\_digest를 사용하지 않도록 한다. digest authentication 의 상세 스펙은 RFC2617을 참조한다.

Digest authentication을 사용하는 방법은 다음과 같다. 설정 단계는 basic authentication과 비슷하다.

### ① 패스워드 파일 생성

Digest authentication을 위한 패스워드 파일 생성 방법은 다음과 같다. Basic authentication에서 패스워드 파일을 생성할 때와 다른 점은 인증영역 식별자가 아규먼트로 들어간다는 것이다.

< 새로운 digest 패스워드 파일 생성>

```
$htdigest -c /usr/local/apache/passwd/digest sec jun
Adding password for username in realm sec.
New password: “패스워드 첫 입력”
Re-type new password: “패스워드 두번째 입력”
$
```

sec은 인증영역 식별자로, httpd.conf 파일에서 AuthName 지시자를 사용해서 지정되는 이름과 동일해야 한다. 기존 패스워드 파일에 추가하는 경우에는 -c 플래그를 생략하도록 한다.

생성된 패스워드 파일은 반드시 문서 디렉토리 밖에 저장하여 외부에서 접근이 불가능하도록 해야 한다. 비록 내용이 암호화되어 있더라도 우선 접근이 가능하면 패스워드를 알아낼 수 있는 기회가 주어질 수 있기 때문이다. 또한 사용자들로 하여금 웹사이트의 패스워드와 다른 중요한 것들의 패스워드를 다르게 사용하도록 권장한다.

### ② 패스워드 파일 사용을 위해 configuration 설정

패스워드 파일이 생성되면, 아파치에게 이 패스워드 파일을 사용해서 사용자인증을 수행하도록 알려주어야 한다. 이 설정은 다음 지시자를 사용한다.

- AuthType : 인증 타입. 여기서는 Digest
- AuthName : 인증영역 식별자(realm)
- AuthDigestFile : 패스워드 파일의 위치
- AuthDigestGroupFile : 그룹 파일의 위치 (선택사항)
- Require : 허가를 주기 위해 만족시켜야 하는 요구사항들

httpd.conf파일의 <Directory> 섹션이나 .htaccess 파일에서 이 지시자들을 사용하여 설정하면 되는데, 다음은 httpd.conf 파일에서 area라는 디렉토리에 대해 Digest authentication을 설정한 예이다.

```

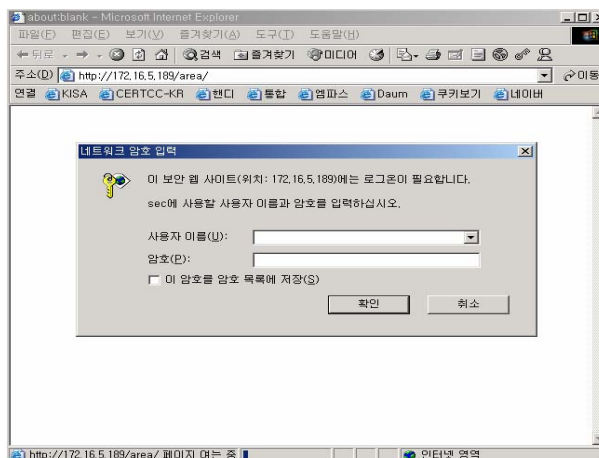
httpd.conf
<Directory "/home/httpd/html/area">
.....
AuthType Digest
AuthName "sec"
AuthDigestFile /usr/local/apache/passwd/digest
Require user jun
.....
</Directory>

```

위 예제에서 AuthType은 Digest authentication을 사용한다는 것을 명시하고, AuthName의 “Digest authentication test” 는 인증영역 식별자로 패스워드 팝업창에 표시된다. 또한 패스워드 파일의 위치는 /usr/local/apache/passwd /digest 이고, 오직 아이디가 jun 인 사용자만이 패스워드가 일치할 경우 접근이 허용된다.

httpd.conf 파일을 변경한 경우 반드시 아파치 서버를 재시작 해야 한다. .htaccess 파일을 변경한 경우에는 재시작이 필요 없으며 즉시 효과가 나타난다.

다음은 위와 같이 digest authentication을 설정했을 때, 디렉토리 접근시 나타나는 인증요구 화면이다.



### ③ 그룹 파일 생성(선택사항)

그룹 파일을 사용하는 방법도 basic authentication과 동일하다. 그룹파일의 형식은 다음과 같다.

```
①      ②  
staff: jun prmkim ksn  
employee: yschoi kikim wgjoo
```

① 그룹 이름 ② 사용자 아이디

이렇게 그룹파일을 만든 후, httpd.conf 파일에서 해당 디렉토리 섹션을 다음과 같이 수정한다. AuthDigestGroupFile 지시자를 사용해서 그룹파일의 위치를 지정해 주고, Require에 인증을 허용할 그룹명을 지정해준다. 다음 예제는 staff 그룹에 속한 사용자만이 /home/httpd/html/area 디렉토리 내의 자원에 접근할 수 있도록 설정한 것이다.

```
httpd.conf  
<Directory "/home/httpd/html/area">  
.....  
AuthType Digest  
AuthName "KISA staff"  
AuthUserFile /usr/local/apache/passwd/digest  
AuthDigestGroupFile /usr/local/apache/passwd/digest.groups  
Require group staff  
.....  
</Directory>
```

Digest authentication의 가장 큰 장점은 패스워드를 암호화 해서 네트워크로 전송한다는 것이다. 그러나 모든 브라우저가 이것을 지원하지는 않는다. Opera 4.0 이후 버전, IE 5.0이후 버전, Mozilla 1.0.1 과 Netscape 7 이후 버전, Amaya가 digest authentication을 지원하고, 다른 브라우저들은 지원하지 않는다.

Digest authentication의 단점은 패스워드는 암호화되어 전송되지만 데이터는 그렇지 않다는 것이다. 또한 패스워드도 digest 형태로 전송되기 때문에 HTTP 프로토콜에 익숙한 사람은 digest된 패스워드를 이용해서 콘텐츠에 접근할 수 있다. 이러



한 문제를 해결하려면 SSL<sup>22</sup>을 사용하는 것이 좋다.

#### ■ Database authentication modules

Basic authentication과 digest authentication은 둘 다 인증 정보를 저장하는데 텍스트 파일을 사용하는 단점이 있다. 텍스트 파일은 인덱싱이 안되기 때문에 검색 속도가 느리다. 또한 HTTP가 stateless이기 때문에 컨텐츠가 요청될 때마다 인증이 수행되어야 하고 텍스트 파일에서 검색을 수행해야 한다. 최악의 경우 사용자 이름이 텍스트 파일에 없으면 파일의 모든 라인을 점검해야 한다.

사용자가 적은 경우에는 이것이 큰 문제가 되지 않지만, 사용자가 많은 경우 웹 서버의 응답이 매우 느려진다. 실제로 많은 경우 정상적인 사용자 이름과 패스워드를 입력해도 인증 모듈이 그 파일에서 사용자를 찾는데 시간이 엄청나게 걸리기 때문에 아파치가 인증 실패를 리턴할 수 있다.

이런 경우에 다른 대안으로 데이터베이스를 사용하는 것이 좋다. 아파치는 다양한 데이터베이스를 사용해서 인증을 수행할 수 있도록 많은 모듈을 제공한다. 여기에서는 두가지 대표적인 모듈인 mod\_auth\_db와 mod\_auth\_dbm에 대해 설명한다.

이 두 모듈은 사용자이름과 패스워드를 DB나 DBM 파일에 저장할 수 있게 한다. DB파일과 DBM 파일은 활용상의 차이가 있는데, BSD, Linux 같은 시스템에서는 동일하다. 두 모듈 중 플랫폼에 적합한 모듈을 선택한다. 만약 사용하고 있는 플랫폼에서 DB 지원을 하지 않는다면 설치<sup>23</sup>를 해야 한다.

Berkeley 데이터 베이스 파일로 알려진 DB 파일은 가장 간단한 형태의 데이터베이스로서 HTTP 인증에 적합하다. DB 파일은 키와 값의 페어를 저장한다. 즉, 변수의 이름과 변수의 값을 저장한다. 다른 데이터베이스는 한 레코드에 많은 필드가 저장될 수 있지만, DB 파일은 키와 값의 페어만을 저장한다. 이것은 사용자 이름과 패스워드를 요구하는 인증에 아주 적합하다.

다음은 mod\_auth\_db의 사용 방법인데, 이것은 mod\_auth\_dbm에도 동일하게 적용된다. 단지 명령어나 파일 이름, 지시자만 db를 dbm으로, DB를 DBM으로 바꾸기만 하면 된다.

---

<sup>22</sup> modssl 사이트([www.modssl.or.kr](http://www.modssl.or.kr)) 참조

<sup>23</sup> [www.sleepycat.com](http://www.sleepycat.com) 참조

mod\_auth\_db 모듈은 디폴트로 컴파일되어 있지 않기 때문에, 이를 사용하기 위해서는 별도로 모듈을 컴파일한 후 추가해야 한다. 그러면 일반 인증과 거의 똑같은 방법으로 mod\_auth\_db 인증을 사용할 수 있다.

### ① 사용자 파일 생성

여기서 사용자 파일은 평범한 텍스트 파일이 아니고 DB 파일이다. 아파치는 이 사용자 파일을 관리할 수 있도록 간단한 유틸리티를 제공한다. 유틸리티 이름은 dbmmanage인데, bin 디렉토리에 위치한다.

dbmmanage는 htpasswd나 htdigest보다 복잡한 편이지만 그래도 사용이 비교적 간단하다. 다음은 jun이라는 사용자 계정을 추가하는 과정이다.

```
$dbmmanage /usr/local/apache/passwd/passwords.dat adduser jun
New password : "패스워드 첫 입력"
Re-type new password: "패스워드 두번째 입력"
User jun added with password encrypted to FdrQLsuCAcl6o Using crypt
$
```

펄(perl) 등의 언어로 패스워드 파일에 저장된 사용자와 패스워드 정보를 관리하는 도구를 작성할 수 있다.

### ② 인증을 위해 그 파일을 사용하도록 아파치 설정

패스워드 파일을 생성했다면 아파치가 그 파일을 사용해서 인증을 수행할 수 있도록 알려주어야 한다. 이것은 .htaccess 파일이나 httpd.conf 파일의 <Directory> 섹션에서 설정할 수 있는데, 다음은 httpd.conf 파일에서 설정한 예이다.

```
httpd.conf
<Directory "/home/httpd/html/area">
.....
AuthType Basic
AuthName "KISA staff"
AuthDBUserFile /usr/local/apache/passwd/passwords.dat
Require user jun
.....
</Directory>
```

### ③ 옵션으로 그룹 파일 생성

앞서 언급한 바와 같이 DB 파일은 키와 값의 페어를 저장한다. 그룹 파일의 경우 키는 사용자 아이디가 되고, 값은 콤마로 구분된 그룹 목록이 된다. 그룹 이름으로 저장하던 이전 그룹 파일과는 달리 사용자 이름을 인덱스로 해서 저장함으로써 레코드를 찾기가 쉬워졌다.

그룹 추가는 다음과 같이 할 수 있다.

```
$dbmmanage add groupfile jun one,two,three
$
```

그룹 파일 설정은 다음과 같이 한다. 여러 개의 그룹을 가지고 있어도 단 하나의 그룹만 요구하면 된다.

```
httpd.conf
<Directory "/home/httpd/html/area">
    .....
    AuthType Basic
    AuthName "KISA staff"
    AuthDBUserFile /usr/local/apache/passwd/passwords.dat
    AuthDBGroupFile /usr/local/apache/passwd/groups.dat
    require group three
    .....
</Directory>
```

패스워드와 그룹 정보를 위해 동일한 파일을 사용하고 싶다면 그렇게 할 수 있다. 그러나 Dbmmanage 유틸리티를 사용하기 전에 패스워드를 직접 암호화 해야 하기 때문에, 관리하기가 좀 더 복잡해진다.

## [아파치 15] 접근제어

접근제어는 사용자의 정체성이 아닌 호스트 이름이나 호스트 주소 등에 기반해서 접근을 제어하는 것이다. 즉 문서를 요청한 호스트의 호스트 이름이나 호스트 주소를 기반으로 접근을 허락하고 거부한다.

접근제어를 위해 사용되는 지시자로 Allow 와 Deny, Order 가 있다. Allow 와 Deny 는 접근을 허락하고 거부할 호스트 이름이나 호스트 주소를 지정한다. Order 는 Allow 와 Deny 지시자가 함께 사용될 때 어느 지시자가 우선순위를 가지고 먼저 처리될지를 지정한다.

이 지시자들의 사용법은 다음과 같다.

```
allow from address
```

*address* 는 IP 주소(혹은 부분 IP 주소)나 도메인네임(혹은 부분 도메인네임)이 될 수 있다. 필요하다면 여러 개의 주소나 도메인네임을 명시할 수 있다.

예를 들어 게시판에 스팸성 글을 올리는 사람이 있다면 다음과 같이 설정하여 그 사람의 주소로부터 게시판에 접근하지 못하도록 할 수 있다.

```
httpd.conf  
Order Allow,Deny  
Allow from all  
Deny from 11.22.33.44
```

이와는 반대로, 회사내부에서의 접속만을 허용하고 외부에서의 접근은 전부 허용하지 않고자 할 때는 다음과 같이 설정한다.

```
httpd.conf  
Order Deny,Allow  
Deny from all  
Allow from .company.com
```

IP 주소나 도메인 네임의 일부를 명시하여 한 도메인 전체를 블러킹하거나 혹은 전체 탑 레벨 도메인(tld, .com 이나 .gov 같은 탑 레벨 도메인)으로부터의 접근을 막을 수 있다.

```
httpd.conf
deny from 192.101.205
deny from exampleone.com exampletwo.com
deny from tld
```

Satisfy 지시자는 사용자 접근 허용을 결정할 때, 여러 개의 기준이 적용될지를 명시하는 지시자이다. 인자로 all 이나 any 중 하나를 취하는데, 디폴트는 all이다. All은 모든 기준들이 만족되어야만 접근을 허용한다는 의미이다. any인 경우 여러 개의 기준 중에 하나만 만족시켜도 접근을 허용한다.

다음 예제는 외부 네트워크로부터의 접근은 패스워드로 보호하고, 내부 네트워크로부터의 접근은 무조건 허용하는 것이다.

```
httpd.conf
<Directory /usr/local/apache/htdocs/sekrit>
    AuthType Basic
    AuthName intranet
    AuthUserFile /www/passwd/users
    AuthGroupFile /www/passwd/groups
    Require group customers
    Order allow,deny
    Allow from internal.com
    Satisfy any
</Directory>
```

<Limit> 지시자를 이용하면 호스트 이름이나 IP, 혹은 인증에 기반해서 메소드의 사용을 제한할 수 있다. 이 기능을 사용하면 불필요한 사용자의 파일 업로드를 제한할 수 있다.

보통 파일 업로드에는 PUT과 POST 메소드가 사용된다. 다음 예제는 211.232.27에서 접근한 IP에 대해서만 POST, PUT 메소드의 사용을 허용하는 것이다.

httpd.conf

```
<Limit GET POST PUT>
    order deny,allow
    deny from all
    allow from 211.232.27
</Limit>
```

<Limit>와 반대되는 지시자로 <LimitExcept>가 있다. <LimitExcept>는 지정된 메소드를 제외한 나머지 메소드에 대해서 사용을 제한한다.

## 제 5 절 로깅

### [아파치 16] 로그 설정 및 분석

관리자는 아파치의 로깅(logging) 기능을 이용하여 사이트 방문자, 방문자의 활동 등의 정보를 모니터링할 수 있다. 모든 침입 과정 및 접근 정보들이 로그에 기록되기 때문에 로그는 웹 서버 보안에 있어서 중요한 요소이다. 따라서 로그 파일을 주기적으로 검토하고 백업 및 안전하게 관리할 필요가 있다.

아파치는 두 개의 로그 파일을 사용하는데, 에러 로그와 액세스 로그이다. 에러 로그는 아파치 서버의 에러 정보를 기록하고, 액세스 로그는 아파치 서버가 처리하는 모든 요청에 대한 정보를 기록한다. 로그 파일의 위치는 httpd.conf 파일에서 지정한다.

httpd.conf

```
#
# ErrorLog : The location of the error log file.
# If you do not specify an ErrorLog directive within a <VirtualHost>
# container, error messages relating to that virtual host will be
# logged here. If you *do* define an error logfile for a <VirtualHost>
# container, that host's errors will be logged there and not here.
#
ErrorLog /var/log/httpd/error_log

#
# The location and format of the access logfile (Common Logfile Format).
# If you do not define any access logfiles within a <VirtualHost>
# container, they will be logged here. Contrariwise, if you *do*
# define per-<VirtualHost> access logfiles, transactions will be
# logged therein and *not* in this file.
CustomLog /var/log/httpd/access_log common
```

## ■ 에러 로그

에러 로그는 아파치가 서버 에러 정보를 저장하는 로그 파일로서 가장 중요한 로그 파일이다. 이곳에는 요청 처리 과정에서 발생한 에러 정보와 해결 방법이 저장된다. 서버 구동 중에 혹은 서버 운영 중에 어떤 문제가 발생하면 제일 먼저 살펴 보아야 할 것이 바로 이 파일이다.

에러 로그 파일의 이름과 위치는 httpd.conf 파일에서 ErrorLog 지시자를 통해 지정된다. 에러 로그 파일의 이름은 일반적으로 Unix계열시스템에서는 error\_log를 사용한다.

에러 로그 파일의 포맷은 비교적 자유로운 형식인데, 대부분의 경우 다음과 같은 정보가 포함된다.

```
① [Wed Oct 11 14:32:52 2000] ② [error] ③ [client 127.0.0.1] ④ client denied by server  
configuration: /export/home/live/ap/htdocs/test
```

- ① 메시지의 날짜와 시간
- ② 에러의 위험도
- ③ 에러를 발생시킨 클라이언트의 IP주소
- ④ 에러 메시지의 내용 (클라이언트가 요청한 문서를 파일 시스템 경로로 표현)

에러 로그에는 CGI 스크립트에서 발생하는 디버깅 정보도 저장될 수 있는데, CGI 스크립트에서 stderr를 사용하면 된다.

에러 로그는 위험도에 따라 [표 4-3]과 같이 8가지 수준으로 분류된다. emerg는 위험도가 가장 높은 에러이고, debug는 위험도가 가장 낮은 에러이다.



[표 4-3] 아파치 에러 로그의 위험도 레벨

수준	설명	예제
emerg	긴급 상황 - 시스템 사용불가	"Child cannot open lock file. Exiting"
alert	긴급 조치가 필요함	"getpwnid: couldn't determine user name from uid"
crit	위험한 상황	"socket: Failed to get a socket, exiting child"
error	에러 상황	"Premature end of script headers"
warn	경고 상황	"child process 1234 did not exit, sending another SIGHUP"
notice	정상이지만 중요한 상황	"httpd: caught SIGBUS, attempting to dump core in ..."
info	정보	"Server seems busy, (you may need to increase StartServers, or Min/MaxSpareServers)..."
debug	디버깅 메시지	"Opening config file ..."

에러 로그 파일에 기록될 에러의 위험도 수준은 다음과 같이 httpd.conf 파일에서 LogLevel 지시자를 이용하여 지정할 수 있다.

```

httpd.conf
#
# LogLevel : Control the number of messages logged to the error_log.
# Possible values include: debug, info, notice, warn, error, crit,
# alert, emerg.
#
LogLevel error
    
```

LogLevel 지시자의 디폴트 값은 warn이다. LogLevel 지시자가 error로 지정되면, 그보다 위험도가 높은 crit와 alert 수준의 에러들도 기록이 된다. 따라서 에러의 위험도가 낮을수록 로그 파일에 쌓이는 로그의 양이 방대해지기 때문에, 에러의 위험도를 적당하게 조절할 필요가 있다. 최소한 crit 보다 낮은 레벨을 지정할 것을 권고한다.

에러 로그를 커스터마이징하는 것은 어렵지만, 특정 요청의 경우 에러 로그에 저장되는 엔트리와 액세스 로그에 저장되는 엔트리가 동일하기 때문에, 액세스 로그를 커스터마이징해서 에러 조건에 대한 더 상세한 정보를 얻을 수 있다.

에러 로그를 지속적으로 모니터링 해야 하는 경우 다음과 같은 명령어를 사용하는 것이 좋다. tail 명령어는 파일의 마지막 부분부터 보여주기 때문에 최근에 기록된 로그부터 볼 수 있다.

```
$ tail -f error_log
```

## ■ 액세스 로그

액세스 로그는 서버가 처리하는 모든 요청에 대한 정보를 기록한다. 액세스 로그의 위치와 로그 포맷은 CustomLog 지시자를 통해 지정된다. LogFormat 지시자를 이용해서 다양한 로그 포맷을 만들어 놓고, 간단하게 선택하여 사용할 수 있다.

액세스 로그로 사용되는 공통적인 로그 포맷은 Common Log Format(CLF)과 Combined Log Format(CLF)이다.

다음은 Common Log Format(CLF)로 불리는 로그 포맷으로 많은 다른 웹 서버에서도 동일하게 생성되는 포맷이고, 많은 로그 분석 프로그램이 읽을 수 있는 포맷이다. httpd.conf 파일에서 다음과 같이 설정할 수 있다.

```
httpd.conf
LogFormat "%h %l %u %t W"%rW" %>s %b" common
CustomLog logs/access_log common
```

LogFormat 지시자는 하나의 포맷 스트링을 정의하고 common이라는 닉네임을 붙인다. CustomLog 지시자는 로그가 저장될 파일의 위치와 이름, 그리고 저장될 로그의 포맷을 정의한다.

이 포맷에 의해 생성된 로그는 다음과 같다.

```

①      ② ③ ④      ⑤      ⑥ ⑦
172.16.5.100 - jun [08/Apr/2003:16:03:43 +0900] "GET /php HTTP/1.1" 301 313

```

- ① 클라이언트의 IP 주소(%h) - HostnameLookups이 On으로 설정되어 있으면 IP주소 대신 호스트네임을 찾아서 저장한다(이 설정으로 서버가 크게 느려질 수 있기 때문에 가능하면 사용하지 않도록 한다).
- ② 클라이언트의 identity(%l) - 클라이언트 컴퓨터의 identd에 의해 결정된 클라이언트 identity. IdentityCheck가 On으로 설정되어 있지 않으면 이 정보를 찾지 않는다(이 설정으로 서버가 느려질 수 있고, identity 정보도 신뢰하기 어렵기 때문에 가능하면 사용하지 않도록 한다).
- ③ HTTP 인증을 받은 사용자의 ID(%u) - 인증을 받지 못한 경우에(상태 코드가 401인 경우) 이 값은 부정확하다. 또한 요청받은 문서가 인증을 요구하지 않는 경우에는 ‘-’ 로 표시된다.
- ④ 서버가 요청 처리를 끝낸 시간(%t) - [일/월/년:시:분:초 지역]
- ⑤ 클라이언트의 요청 내용(W" %rW" ) - 사용한 메소드, 요청한 자원, 사용한 프로토콜
- ⑥ 상태코드(%>s) - 서버가 클라이언트에게 보낸 상태 코드. 2XX(성공), 3XX(redirectation), 4XX(클라이언트에 의한 에러), 5XX(서버에 의한 에러). 상세한 상태 코드는 부록을 참고한다.
- ⑦ 클라이언트에게 전송된 콘텐츠의 크기 - response header 부분은 포함되지 않는다. 클라이언트에게 전송된 콘텐츠가 없으면 이 값은 ‘-’ 로 표시된다.

다음은 Combined Log Format이다. httpd.conf 파일에서 다음과 같이 설정할 수 있다.

```

httpd.conf
LogFormat "%h %l %u %t W"%rW" %>s %b W"%{Referer}iW" W"%{User-agent}iW"
combined
CustomLog log/acces_log combined

```

이 포맷은 두 개의 필드를 제외하면 Common Log Format과 동일하다. 추가된 필드는 퍼센트 지시자 *%{header}i*를 사용하고 있는데, *header*는 HTTP request header 중 일부가 될 수 있다. 이 포맷에 의해 생성된 로그는 다음과 같다.

```
172.16.5.100 - jun [08/Apr/2003:16:03:43 +0900] "GET /php HTTP/1.1" 301 313
```

```
① - " ② "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)"
```

- ① 클라이언트가 요청한 자원이 include되었거나 링크된 페이지(“” %`{Referer}`I“” ) - 위 예제에서는 그러한 페이지가 없음
- ② 클라이언트 브라우저에 대한 정보(“” %`{User-agent}`I“” )

아파치에서는 액세스 로그 파일을 여러 개 만들 수 있다. 예를 들어 httpd.conf 파일에서 다음과 같이 정의하면 저장되는 정보가 각기 다른 3개의 액세스 로그 파일이 생성된다. 다음 예제에서는 logs 디렉토리 밑에 access\_log, referer\_log, agent\_log 가 생성된다.

httpd.conf

```
LogFormat "%h %l %u %t W"%rW" %>s %b" common
CustomLog logs/access_log common
CustomLog logs/referer_log "%{Referer}i -> %U"
CustomLog logs/agent_log "%{User-agent}i"
```

## ■ 로그 설정시 유의사항

웹 서버의 일반 시스템사용자가 아파치의 로그 파일이 저장되는 디렉토리에 대해 쓰기 권한을 가진다면, 아파치 서버가 시작될 때 갖게 되는 UID와 동일한 권한(대부분의 경우 root)을 얻을 수 있다. 아파치는 대부분의 경우 root권한으로 로깅을 수행하는데, 시스템사용자는 아파치의 로그 파일을 다른 중요 시스템 파일에 대한 링크로 대체하여, root 권한으로 다른 중요 시스템 파일의 내용을 변경할 수 있다. 따라서 일반사용자는 로그가 저장되는 디렉토리에 대해 쓰기 권한이 없도록 설정해야 한다.

또한 로그 파일에 클라이언트가 제공하는 데이터가 들어갈 수 있으므로, 로그 데이터 처리시 주의하도록 한다. 악의적인 클라이언트가 제어문자 등을 로그 파일에 삽입하여 웹 서버를 침해할 수 있다. 특히 클라이언트가 웹 서비스를 통해서 아파치의 로그 파일을 볼 수 없도록 해야 한다.

## [아파치 17] 로그 백업

클라이언트의 요청이 증가하면 로그 파일에 저장되는 정보의 양도 크게 증가한다. 액세스 로그 파일의 경우 10,000개의 요청에 대해 로그 파일이 1MB이상 증가한다. 따라서 주기적으로 로그 파일을 백업 받고 기존 파일을 삭제해주어야 한다. 이렇게 로그 파일을 교체한 후에는 반드시 아파치를 재시작해야 한다. 만약 재시작을 하지 않는다면, 아파치는 기존 파일에 대한 파일 오픈을 유지하면서 계속 쓰기를 시도할 것이다.

Graceful restart의 경우 기존에 유지된 연결을 잃지 않고 새로운 로그 파일을 열게 된다. 그러나 아파치 서버는 기존 요청의 처리가 끝날 때까지 기존 로그 파일에 쓰기를 계속한다. 따라서 재시작을 한 후에 로그 파일에 대한 어떤 처리를 수행하기 전에 일정시간 기다릴 필요가 있다. 다음은 로그 파일을 교환한 후 기존 로그 파일을 압축하는 시나리오이다.

```
$mv access_log access_log.old
$mv error_log error_log.old
$apachectl graceful
$sleep 600
$gzip access_log.old error_log.old
```

로그를 교환하는 또 다른 방법은 파이프 로그를 이용하는 것이다. 이 방법을 이용하면 에러 로그와 액세스 로그를 파이프를 통해서 다른 프로세스에게 전달할 수 있다.

파이프 로그를 사용하는 방법은 다음과 같이 단순히 로그 파일 이름 대신 파이프 문자 ‘|’ 와 실행파일 이름을 지정해주면 된다. 실행 파일은 로그를 표준 입력으로 받아서 처리한다. 다음 예제는 아파치에서 제공하는 rotatelog 프로그램을 사용해서 24시간마다 로그를 rotation시키는 설정이다.

```
httpd.conf
CustomLog "|/usr/local/apache/bin/rotatelog /var/log/access_log 86400"
common
```

파이프 로그 프로세스는 아파치의 부모 프로세스가 시작시키기 때문에, 부모 프로세스의 user ID를 상속받는다. 만약 아파치가 root로 시작된다면 파이프 로그 프로세스도 root로 시작된다. 따라서 파이프 로그 프로그램을 간단하고 안전하게 유지하는 것이 보안상 매우 중요하다.

파이프 로그의 가장 큰 장점은 서버를 재시작하지 않고도 로그 rotation을 할 수 있다는 것이다.

이 페이지는 빈 페이지입니다.

## 제 5 장 IIS 보안설정 방법<sup>24</sup>

### 제 1 절 서버 보안 설정

IIS는 Windows OS의 다른 컴포넌트가 제공하는 기능들을 사용하거나 Windows 아키텍처의 많은 기술들을 사용하기 때문에 IIS는 MS Windows OS와 통합된 서비스라고 볼 수 있다. 따라서 IIS 보안을 다룰 때는 반드시 MS Windows OS의 보안도 함께 고려해야 한다. MS Windows의 취약점은 곧 IIS의 취약점으로 간주될 수 있으며, IIS보안은 OS 보안 설정과 밀접한 관련을 가지고 있다.

본 절에서는 MS Windows OS 보안 사항 중 가장 기본적이면서도 중요한 사항과 IIS와 관련해서 꼭 필요한 사항만을 다루도록 한다. MS Windows OS 보안에 대한 상세한 내용은 별도의 보안문서를 참고하기 바란다.

#### [IIS 1] 부팅파티션과 웹 서비스 파티션의 분리

부팅파티션과 데이터파티션을 분리하도록 한다. 서로 다른 디스크를 사용하거나 최소한 파티션을 나누어서 설치하도록 한다. 이렇게 하면 웹 서비스의 피해가 서버 전체의 피해로 확산되는 것을 최소화 할 수 있다.

#### [IIS 2] NTFS 파일 시스템의 사용

Windows 서버 운영체제에서는 두가지 파일 시스템을 제공한다. FAT(File Allocation Table)과 NTFS(NT File System)인데, 이 중에서 NTFS가 보안, 성능, 로깅(logging) 측면에서 우수하다.

특히 보안 측면에서 NTFS는 파일단위의 암호화 및 권한 설정이 가능하고, 디스크 할당량을 설정할 수 있다. 따라서 반드시 NTFS를 사용해서 웹 서버 보안을 강화하

---

<sup>24</sup> 본 장의 내용은 IIS 5.0을 기준으로 하여 작성하였으며, 실제로 Windows 2000 server에 IIS 5.0을 설치하여 테스트하였다.



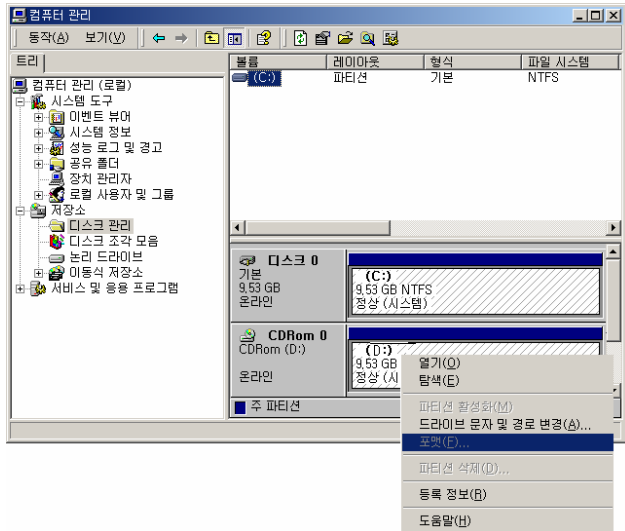
도록 한다. 웹 서버를 운영하는 시스템에서는 최소한 부트 파티션과 웹 서비스를 제공하는 파티션은 NTFS를 사용하도록 한다.

다음은 NTFS 파일 시스템을 사용하는 3가지 방법이다.

- ① OS를 설치하지 않은 경우 OS 설치과정에서 파일 시스템을 NTFS로 포맷한다.
- ② OS가 이미 설치된 경우에는 Convert 유틸리티를 사용해서 FAT 파티션을 NTFS로 변환한다. Convert 유틸리티 사용 예제는 다음과 같다.

```
C:\wconvert D: /FS:NTFS
```

- ③ 데이터가 없는 파티션의 파일 시스템을 변환하는 경우 디스크 관리자를 이용해서 파티션을 NTFS로 재포맷한다. [시작] -> [프로그램] -> [관리도구] -> [컴퓨터 관리]에서 [디스크 관리]를 선택한 후, 해당 드라이브에 대해 마우스 오른쪽 버튼을 클릭한 후 [포맷]을 선택하면 된다.



### [IIS 3] 필요한 구성요소만을 설치

불필요한 구성요소의 설치는 보안과 시스템 성능 측면에서 도움이 되지 않는다. 웹 서비스에 꼭 필요한 요소만을 설치하여 보안관리를 용이하게 하고 예측하기 어려운 위험에 대한 노출을 최소화 하도록 한다.

인터넷 정보 서비스의 구성요소들은 다음과 같다.

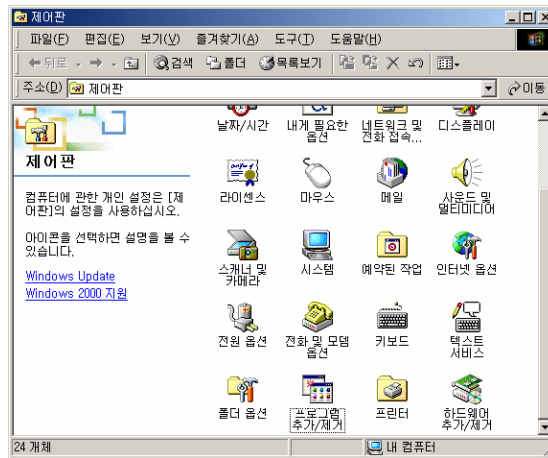
- 공용파일
- 설명서
- 인터넷 서비스 관리자(HTML)
- 인터넷 정보 서비스 스냅인
- FTP 서버
- FrontPage 2000 Server Extension
- NNTP Service
- SMTP Service
- Visual InterDev RAD Remote Deployment Support
- World Wide Web 서버

이 중 다음 3가지 요소는 웹 서버 운영에 필수적인 요소로서, 이 3가지만으로도 웹 서버 운영이 가능하다. 그 외의 구성요소는 필요에 따라 추가하면 된다.

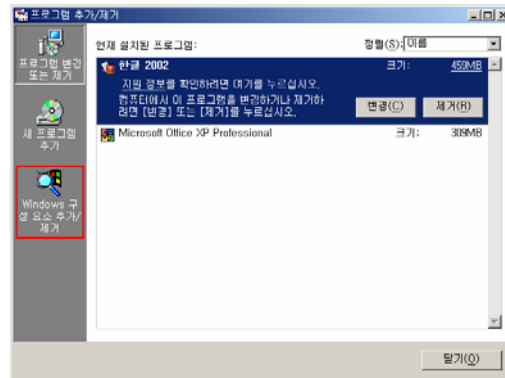
- 공용파일
- 인터넷 정보서비스 스냅인
- World Wide Web 서버

Windows 2000 Server의 설치시 인터넷 정보 서비스의 구성요소를 결정할 수 있고, Windows 2000 Server 설치 후에는 다음과 같은 방법으로 구성요소를 추가하거나 제거할 수 있다.

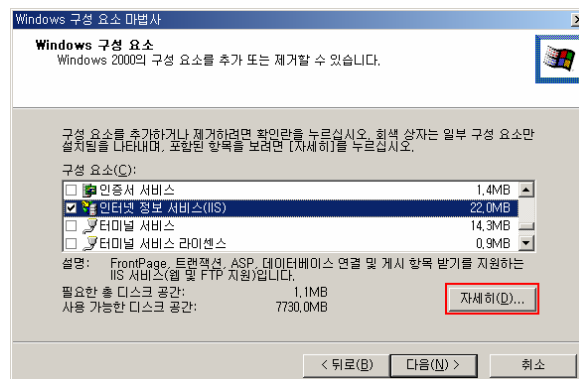
① [제어판]에서 [프로그램 추가/제거] 아이콘을 실행한다.



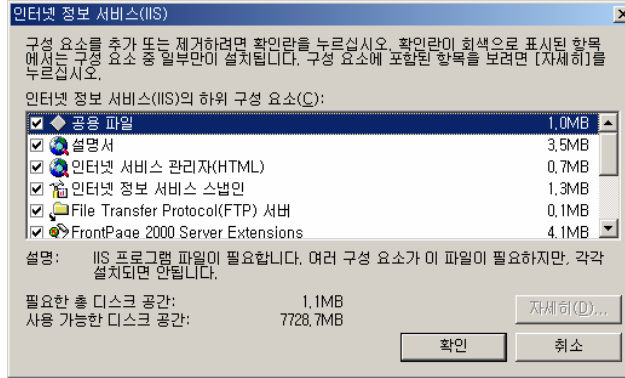
② [프로그램 추가/제거] 창에서 좌측 [Windows 구성 요소 추가/제거] 메뉴를 클릭한다.



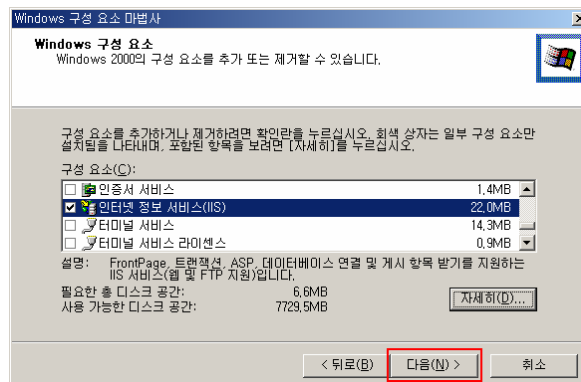
③ [Windows 구성 요소 마법사] 창에서 [인터넷 정보 서비스(IIS)]를 선택한 후 [자세히] 버튼을 클릭한다.



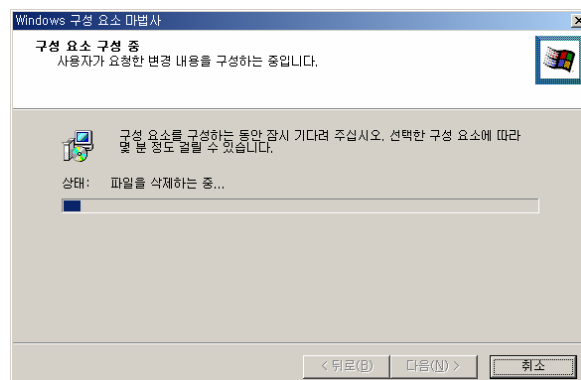
- ④ 이제 추가할 구성요소는 선택하고, 제거할 구성요소는 비선택을 한 후 [확인] 버튼을 클릭한다.



- ⑤ 구성요소의 추가 및 제거가 끝났으면 [다음] 버튼을 클릭한다.



그러면 다음 그림과 같이 구성요소 변경 과정이 시작된다.



#### [IIS 4] 웹전용 서버로 구성(불필요한 서비스 제거)

가급적이면 서버를 웹전용 서버로 구성하도록 한다. 웹 서버 운영에 필요한 서비스들만 구동시키고 불필요한 서비스들을 중지시켜서 보안 문제의 발생 가능성을 최소화 시킨다. Windows 2000 Server를 설치하면 디폴트로 구동되는 서비스 가운데 웹 서버 운영에 필요 없는 서비스들이 있다. 이러한 서비스들을 점검하여 중지시키도록 한다.

[표 5-1]은 IIS가 동작하기 위해 필요한 서비스들이다.

[표 5-1] IIS 동작에 필요한 서비스 목록

서비스명
Event Log
License Logging Service
Windows NT Lanman(NTLM) Security Support Provider
Remote Procedure Call(RPC) Service
Windows NT Server or Windows NT Workstation
IIS Admin Service
Microsoft Distributed Transaction Coordinator(MSDTC)
Protected Storage

현재 운영하려는 웹 서버에 반드시 필요한 서비스가 어떤 것이고 어떤 서비스가 불필요한지를 결정하기 위해 MS에서 제공하는 다음 문서를 참고하도록 한다.

[http://www.microsoft.com/korea/technet/prodtechnol/windows2000serv/deploy/p  
rodspecc/win2ksvc.asp](http://www.microsoft.com/korea/technet/prodtechnol/windows2000serv/deploy/p<br/>rodspecc/win2ksvc.asp)

(혹은 Technet 사이트내 검색 -> “Windows 2000 서비스” 입력 -> 검색결과에서 [기술 자료] 카테고리내의 [Windows 2000 서비스]를 클릭)



이 문서에서는 Windows 2000 Server 제품 계열의 운영 체제에 포함되어 있는 약 100개의 서비스가 알파벳 순서로 요약되어 있다. 여기에서는 각 서비스가 운영체제의 기능과 어떻게 관련되어 있는지에 대해 일반적으로 설명하고, 각 서비스를 사용하지 않는 경우 발생하는 주요 결과에 대해 설명한다.

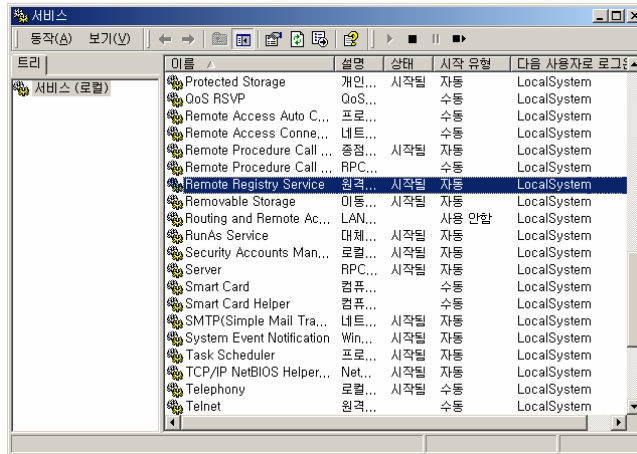
일반적으로 사용중지를 권장하는 서비스들에는 다음과 같은 것들이 있다.

[표 5-2] IIS 웹 서버에서 사용중지를 권장하는 서비스 목록

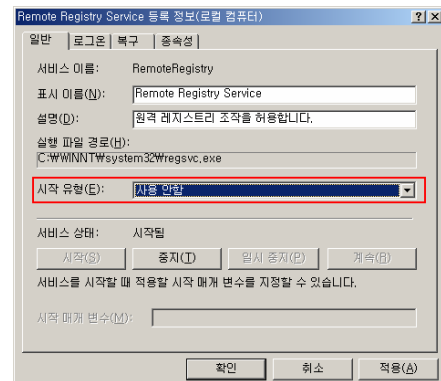
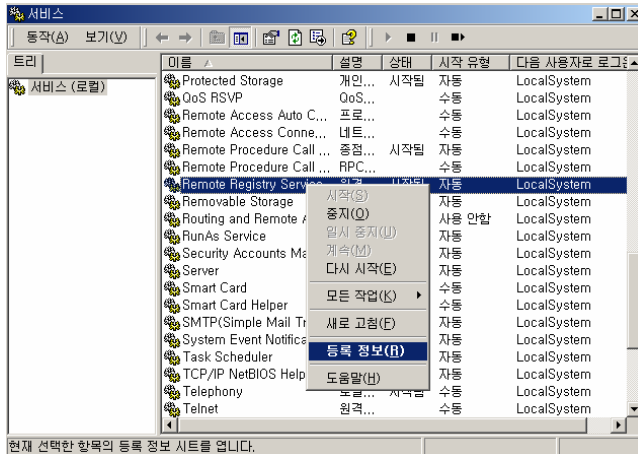
서비스명
Application Management
Clipboard Service
DHCP Client
Fax
Messenger
Print Spooler
Remote Registry Service
Smart Card
Smart Card Helper
Telnet

불필요한 서비스를 중지시키는 방법은 다음과 같다.

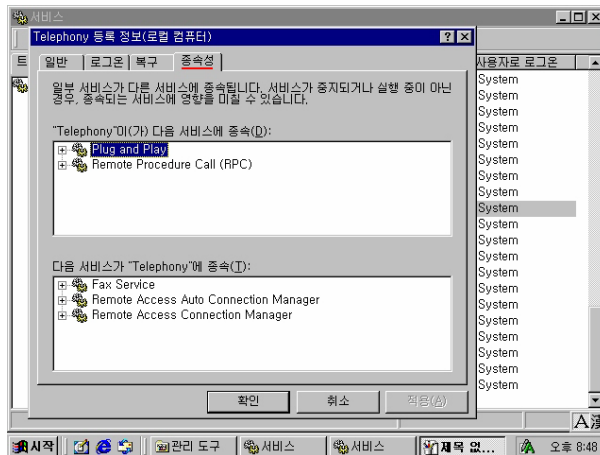
- ① [시작] -> [프로그램] -> [관리도구] -> [서비스] 프로그램을 실행한다.



- ② 중지하려는 서비스에 대해 마우스 오른쪽 버튼을 클릭하고 [등록 정보]를 선택한 후, 시작 유형에서 [사용 안함]을 선택한다.



서비스간의 의존성에 따라 서비스 사용중지가 불가능한 경우가 있는데, 이런 경우에는 각 서비스에 대한 서비스 종속성 옵션을 살펴볼 수 있다. 각 서비스의 [등록정보]의 [종속성] 탭을 선택하면 다음과 같이 그 서비스가 종속된 서비스 목록과 그 서비스에 종속된 서비스 목록을 볼 수 있다.



## [IIS 5] 계정의 수와 권한을 최소화

웹 서버에는 꼭 필요한 계정만을 만들고, 각 계정에 대해서는 필요한 최소한의 권한만 주도록 한다. 일반적으로 웹 서버에는 관리자 계정과 웹 서버 구동을 위한 계정만 남기고 일반 사용자 계정은 삭제하도록 하는 것이 좋다.

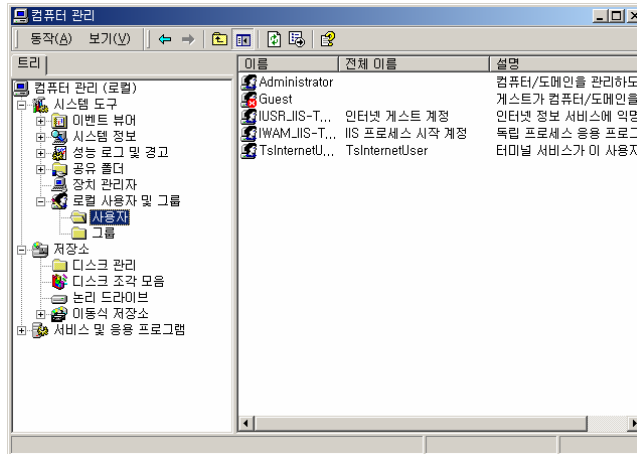
IIS를 설치하면 두 개의 익명 계정이 생성되는데, 'IUSR\_컴퓨터이름' 과 'IWAM\_컴퓨터이름' 이다. 'IUSR\_컴퓨터이름' 은 웹자원에 대한 익명 접근을 허용 하는데 사용되는 계정이다. 'IWAM\_컴퓨터이름' 은 MTS(Microsoft Transaction Server)와 다양한 IIS 개체가 사용하는 계정이다. 이 두 계정에 대해서는 최소한의 권한만 유지하도록 한다.

디폴트 계정 중에 불필요한 계정은 사용안함으로 설정하고 디폴트 이름을 가지고 있는 주요 계정은 이름을 변경하도록 한다. 예를 들어 Guest계정은 사용안함으로 설정하고 Administrator 계정은 해커가 추측하기 어려운 계정명으로 바꾸도록 한다.

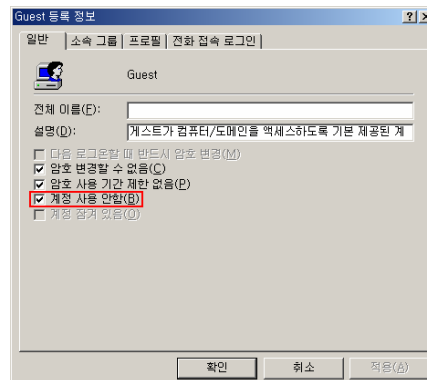


Guest 계정을 사용안함으로 하기 위해서는 다음과 같이 설정한다.

- ① [시작] -> [프로그램] -> [관리도구] -> [컴퓨터 관리]에서 [로컬 사용자 및 그룹]을 선택한 후 [사용자]를 선택한다.

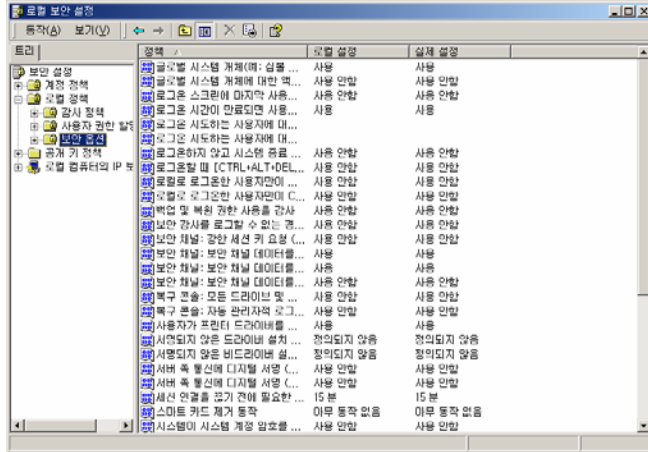


- ② 오른쪽에서 Guest 계정에 대해 마우스 오른쪽 버튼을 클릭한 후 [등록 정보]를 선택한다. 여기에서 [계정 사용 안함]을 선택한다.

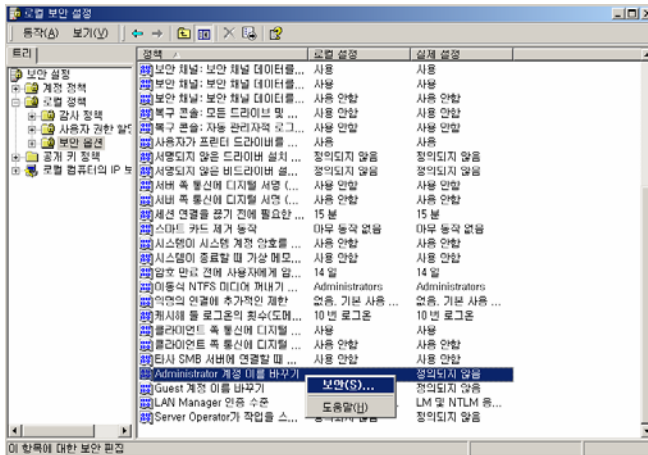


Administrator 계정을 다른 이름으로 바꾸는 방법은 다음과 같다.

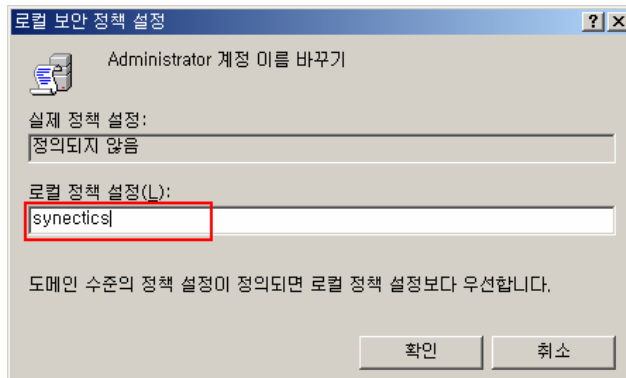
- ① [프로그램] -> [관리도구] -> [로컬 보안 정책]에서 [로컬 정책]을 선택하고 [보안 옵션]을 선택한다.



- ② “Administrator 계정 이름 바꾸기”에 대해 마우스 오른쪽 버튼을 클릭하고 [보안]을 선택한다.



- ③ 로컬 보안 정책 설정 화면에서 새로운 관리자 계정 이름을 입력한다.



시스템을 재시작하면 Administrator 계정이 바뀌어 있음을 확인할 수 있다.

## [IIS 6] 공유 사용 안함

많은 바이러스가 윈도우의 네트워크 공유를 통해 감염되기 때문에 웹 서버에서는 공유를 절대 사용하지 않도록 한다. 일반 공유뿐만 아니라 관리 공유의 경우도 반드시 필요한 것만 사용하고 나머지는 제거하도록 한다.

관리 공유는 관리자와 운영 체제 서비스가 네트워크에서 컴퓨터 환경을 관리하는데 사용하도록 만들어진 것이다. 관리 공유는 설정을 해제할 수 있지만 컴퓨터를 다시 시작하면 공유가 다시 설정되어 있다.

관리 공유에는 다음과 같은 것들이 있다.

- 루트 파티션 또는 볼륨
- 시스템 루트 폴더
- FAX\$ 공유
- IPC\$ 공유
- NETLOGON 공유
- PRINT\$ 공유

루트 파티션과 볼륨은 \$ 기호가 추가된 드라이브 문자 이름으로 공유된다. 예를 들어, C 와 D 드라이브는 C\$와 D\$로 공유된다. 시스템 루트 폴더 (%SYSTEMROOT%)는 ADMIN\$로 공유된다. 이 관리 공유를 사용하면 관리자는 네트워크를 통해 시스템 루트 폴더 계층에 쉽게 액세스할 수 있다. FAX\$ 공유는 팩스를 보내는 과정에서 팩스 클라이언트가 사용한다. 이 공유 폴더는 파일을 캐싱하고 파일 서버에 저장된 표지 페이지에 액세스한다. IPC\$ 공유는 네트워크 프로그램 간 통신에 명명된 파이프를 통해 클라이언트와 서버 사이를 임시로 연결하는 데 사용된다. 이것은 네트워크 서버의 원격 관리에 주로 사용된다. NETLOGON 공유는 Netlogon 서비스가 로그인 요청을 처리하는 데 사용된다. PRINT\$ 공유는 프린터의 원격 관리에 사용된다.

이 중 [드라이브 문자]\$, PRINT\$, FAX\$는 웹 서버에서 반드시 제거하도록 하고 나머지 공유도 사용하지 않는다면 제거하도록 한다<sup>25</sup>.

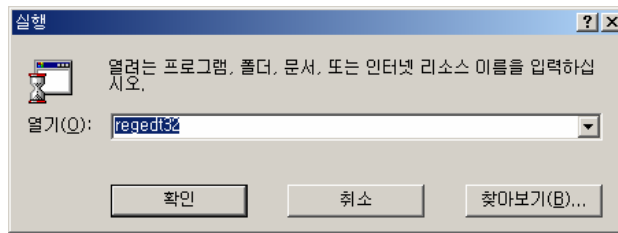
---

<sup>25</sup> IPC\$는 시스템을 시작할 때마다 제거해주어야 한다.

일부 서비스와 프로그램이 관리 공유를 일부 필요로 할 수 있다. 따라서 관리 공유를 설정 해제한 후에 프로그램과 서비스의 기능을 테스트하도록 한다.

관리 공유 설정을 해제하는 방법은 다음과 같다.

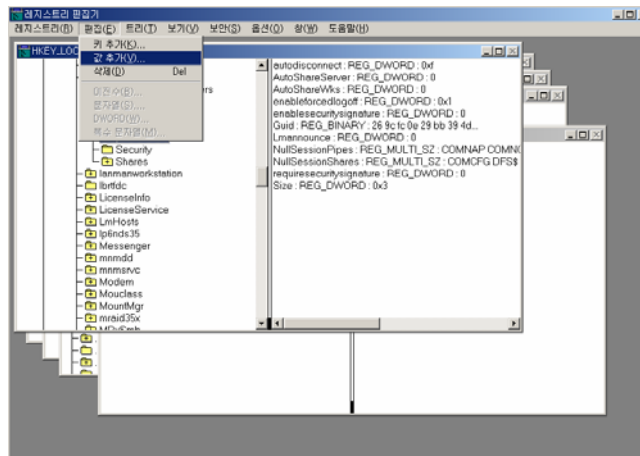
- ① 레지스트리 편집기(regedt32<sup>26</sup>)를 실행한다.



- ② HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\LanmanServer\parameters에서 다음 두 값들을 생성하도록 한다.

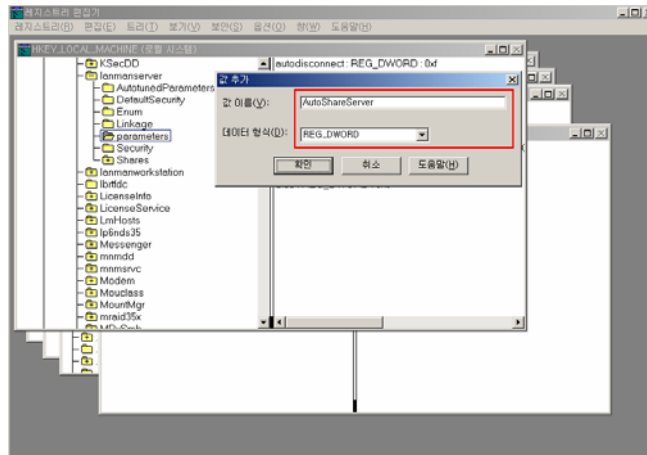
```
AutoShareServer : 0
AutoShareWks : 0
```

생성방법은 다음과 같다. 우선 [편집] 메뉴에서 [값 추가]를 선택한다.

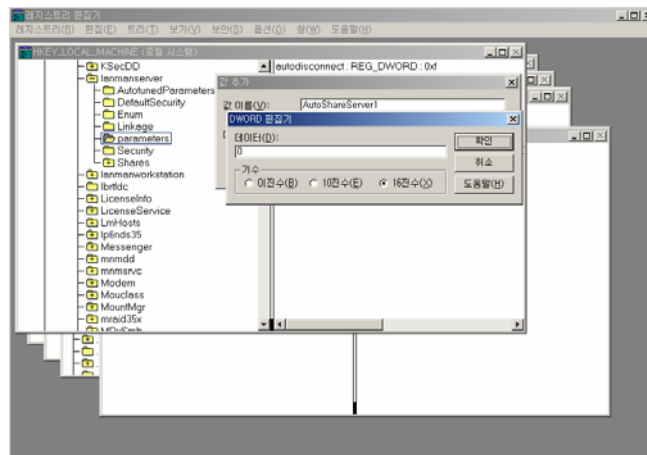


<sup>26</sup> MS사는 Windows NT 4.0과 Windows 2000을 사용할 경우, regedt32.exe를 사용해서 레지스트리를 편집하도록 권고한다. regedit.exe를 사용하면 레지스트리 키에 대한 보안 설정이 불가능하다. regedit.exe는 검색시에만 활용하고, 레지스트리 편집은 regedit32.exe를 이용하도록 한다. MS 기술자료 141377 “Differences Between Regedit.exe and Regedt32.exe” 참조

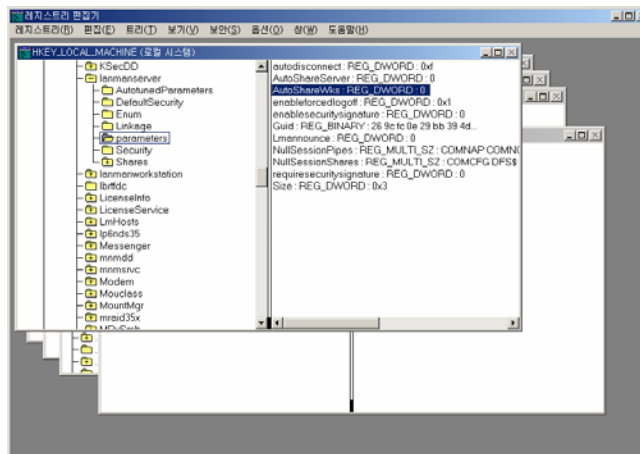
[값이름]으로 'AutoShareServer' 를 입력하고, [데이터 형식]으로 REG\_DWORD를 선택한 후 [확인]을 누른다.



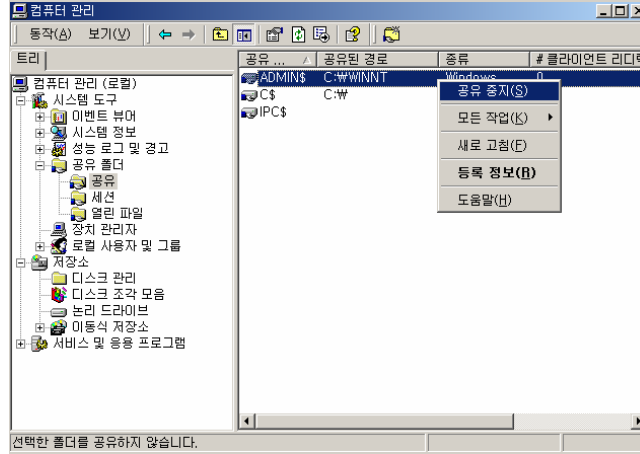
[데이터]로 '0' 을 입력하고 [확인]을 누른다.



위와 동일한 방법으로 AutoShareWks도 생성한다.

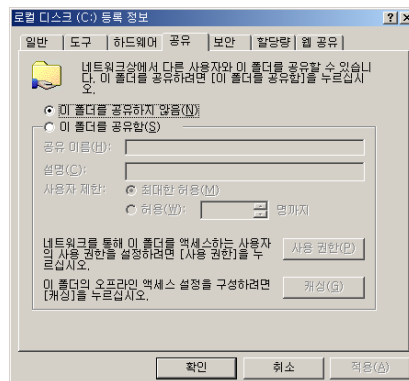


- ③ [시작] -> [프로그램] -> [관리 도구] -> [컴퓨터 관리]에서 [공유 폴더]와 [공유]를 차례로 선택하면 오른쪽에 관리 공유 목록이 보인다. 제거하려는 공유에 대해서 마우스 오른쪽 버튼을 클릭하고 [공유 중지]를 선택한다.



PRINT\$의 경우 Print Spooler 서비스를 사용중지 해줘야 완전한 삭제가 가능하다. 이 서비스를 제거하는 방법은 『[IIS 4] 불필요한 서비스 제거』를 참고하도록 한다.

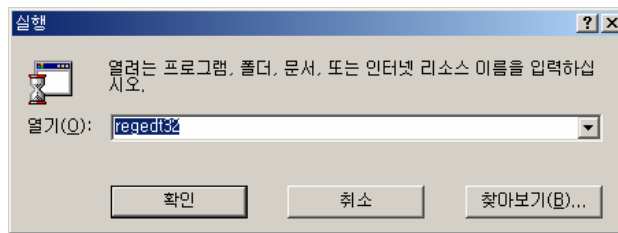
이와 같은 방법으로 관리 공유를 삭제하면 재부팅을 해도 다음 그림과 같이 관리 공유 설정이 다시 살아나지 않는다.



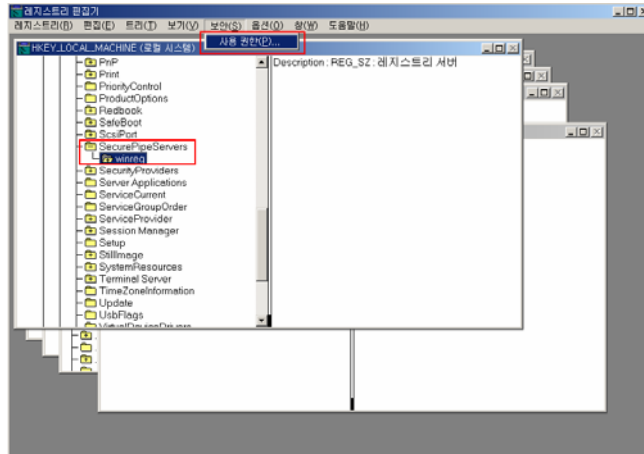
## [IIS 7] 레지스트리 원격 접근 제한

기본 사용 권한에서는 레지스트리의 원격 액세스를 제한하지 않는다. Windows 2000 레지스트리 편집 도구에서 기본적으로 원격 액세스를 지원하므로, 레지스트리 원격 액세스 권한은 관리자에게만 부여해야 한다. 레지스트리에 대한 네트워크 액세스를 제한하는 방법은 다음과 같다.

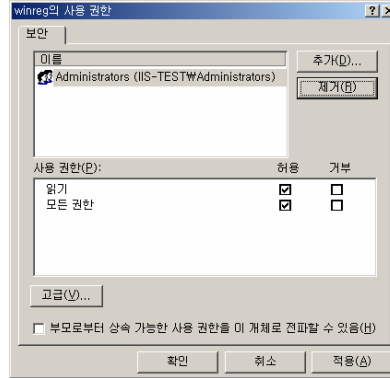
- ① 레지스트리 편집기(regedt32)를 실행한다.



- ② HKEY\_LOCAL\_MACHINE \SYSTEM\CurrentControlSet\Control\SecurePipeServers 에서 winreg 를 선택하고 [보안] 메뉴를 선택한 다음, [사용 권한]을 선택한다.



- ③ Administrators 사용 권한을 모든 권한(Full Control)으로 설정하고, 다른 사용자나 그룹이 표시되지 않는 지 확인한 다음 [확인]을 클릭한다.



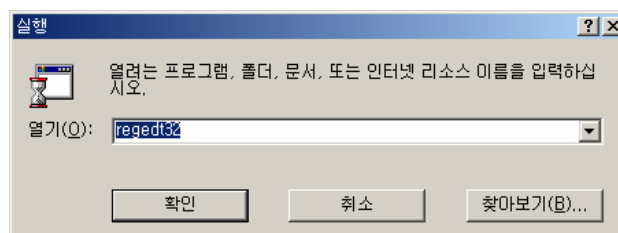
이 키에 설정된 보안 권한은 원격 레지스트리 액세스를 위하여 시스템에 연결할 수 있는 사용자나 그룹을 정의한다.

AllowedPaths 하위 키에는 winreg 키의 보안 권한과 별도로 Everyone 그룹의 구성원이 액세스할 수 있는 키 목록이 있다. 이렇게 해서 winreg 레지스트리 키에서의 액세스 제한에 관계없이 프린터 상태 확인과 같은 특정 시스템 기능을 사용할 수 있다. AllowedPaths 레지스트리 키의 기본 보안 설정은 Administrators만이 경로를 관리할 수 있도록 되어 있다.

#### [IIS 8] 공개 로컬 보안 인증(LSA)의 정보에 대한 접근 제한

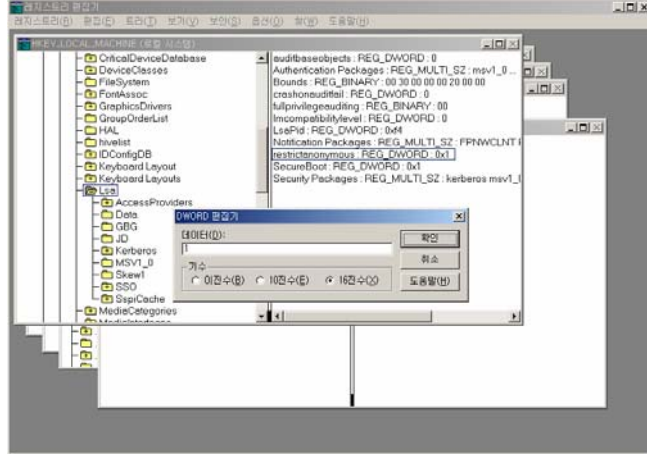
익명 사용자가 Windows NT Security Subsystem의 LSA 구성 요소에 대해 얻을 수 있는 공개 정보를 최소화해야 한다. LSA는 로컬 컴퓨터의 액세스와 사용 권한을 포함한 보안 관리 항목을 처리한다. 이 제한을 설정하는 방법은 다음과 같다.

- ① 레지스트리 편집기(regedt32)를 실행한다.





- ② HKEY\_LOCAL\_MACHINE\System\CurrentControlSet\Control\Lsa에서 restrictanonymous를 더블클릭한 후 값 데이터로 1을 입력하고 [확인] 버튼을 누른다.



**[IIS 9] 시스템 실행 파일에 대한 제한**

사용되지 않는 실행 파일들을 모두 제거하거나 관리자만 실행 가능하도록 설정하여 잠재적인 위험 요소를 제거한다.

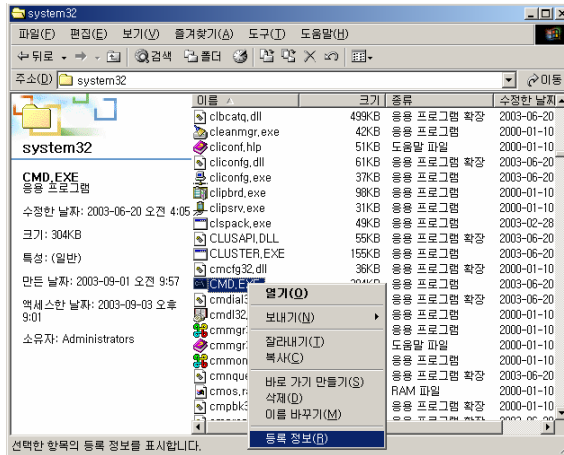
다음 실행 파일들에 대해 관리자만 실행이 가능하도록 설정한다.

**[표 5-3] IIS 웹 서버에서 접근제어를 설정하도록 권장하는 명령어 목록**

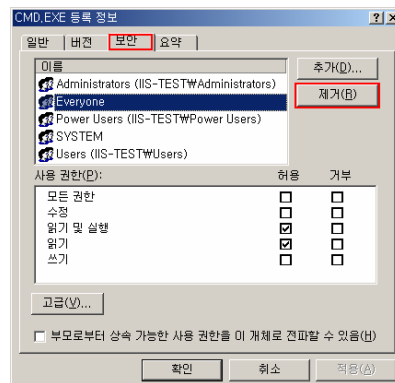
arp.exe	at.exe	cmd.exe	edit.com
edlin.exe	finger.exe	ftp.exe	ipconfig.exe
net.exe	netstat.exe	nslookup.exe	ping.exe
qbasic.exe	rcp.exe	regedit.exe	regedt32.exe
rexec.exe	route.exe	runas.exe	rsh.exe
syskey.exe	telnet.exe	tracert.exe	tftp.exe
xcopy.exe			

관리자만 실행이 가능하도록 설정하는 방법은 다음과 같다.

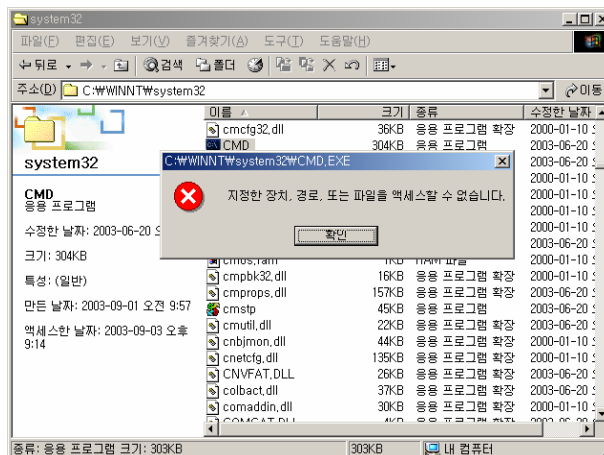
① 실행 파일에 대해 마우스 오른쪽 버튼을 클릭한 후 [등록정보]를 선택한다.



② 등록정보에서 [보안]탭을 선택하고 Administrator와 SYSTEM을 제외한 모든 사용자를 제거한다.



이렇게 설정하면 해당 실행 파일을 일반 사용자권한으로 실행시키려고 할 때 다음과 같은 에러가 발생한다.



[참고] 가장 많이 악용되는 IIS 취약점들은 URL을 통해 cmd.exe를 실행하도록 하는 것이다. 공격자는 cmd.exe의 파라미터로 명령어를 주어서, IIS서버에서 임의의 명령어를 실행시킨다. 이러한 공격을 막기 위한 좋은 방법은 cmd.exe를 삭제하거나, 이 파일이 필요하다면 이 파일의 위치를 다른 곳으로 옮기는 것이다.

그런데 Windows 2000에서는 WFP(Windows File Protection) 메커니즘이 cmd.exe 파일이 이동되거나 이름변경 혹은 삭제되는 경우 재배치를 수행한다. 따라서 Windows 2000에서 cmd.exe를 보호하는 가장 좋은 방법은 이 파일에 대한 접근 권한을 관리자에게만 허용하는 것이다.

### [IIS 10] 최신 서비스팩과 핫픽스 설치

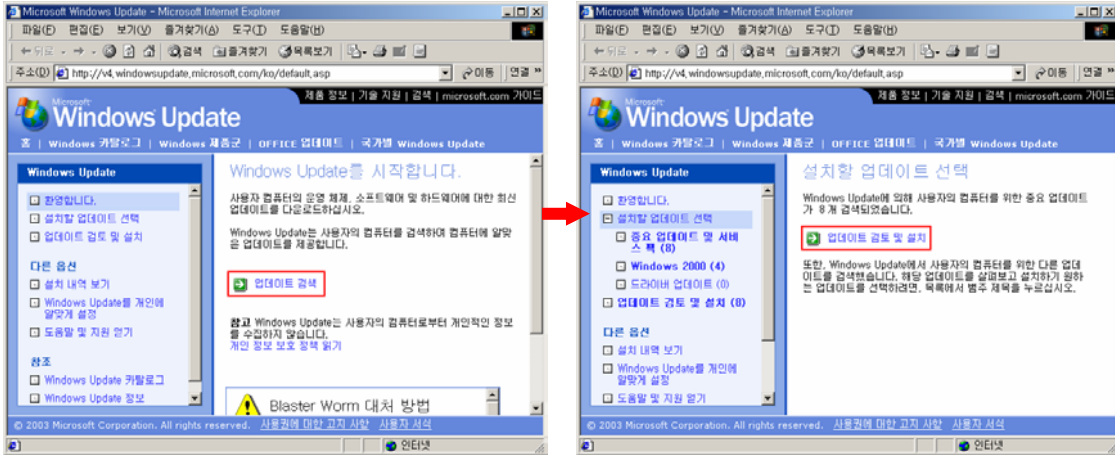
최근까지 발표된 서비스팩과 핫픽스를 설치하기 위해 Windows Update 기능을 이용한다. Windows Update<sup>27</sup>를 실행시키면 해당 서버에 필요한 서비스팩과 핫픽스 목록을 보여준다.

① [시작] -> [Windows Update] 메뉴를 선택한다.

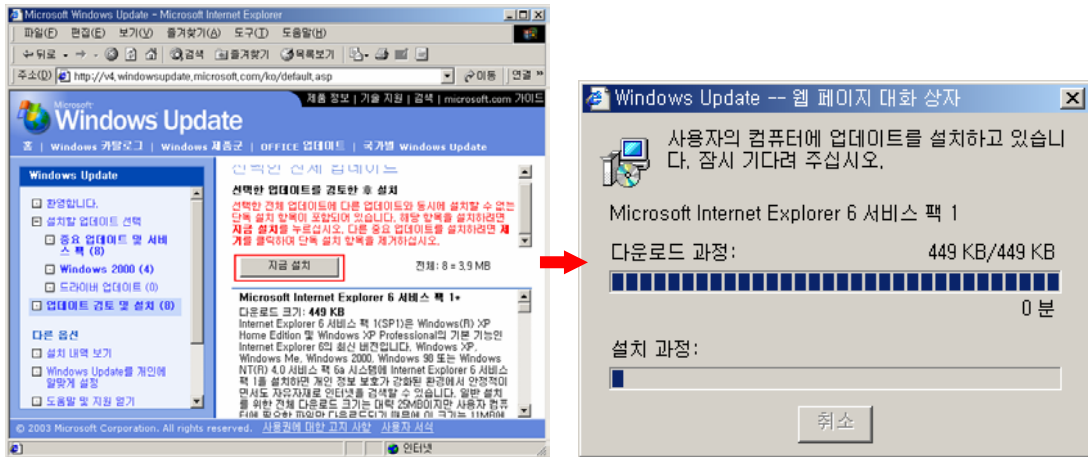


<sup>27</sup> 이 도구의 단점은 MS IIS 웹 서버 어플리케이션은 다루지 않는다는 것이다.

② 그러면 다음과 같이 자동으로 MS사의 Windows Update 사이트에 접속이 된다. 여기에서 [업데이트 검색]을 클릭한다. 업데이트 검색이 끝나면 [업데이트 검토 및 설치]를 클릭한다.



③ 업데이트 항목을 선택한 후 [지금 설치]를 클릭하면 설치가 시작된다. 설치과정 중에 계약 [동의]나 [다음] 버튼을 클릭하면 설치가 진행된다.

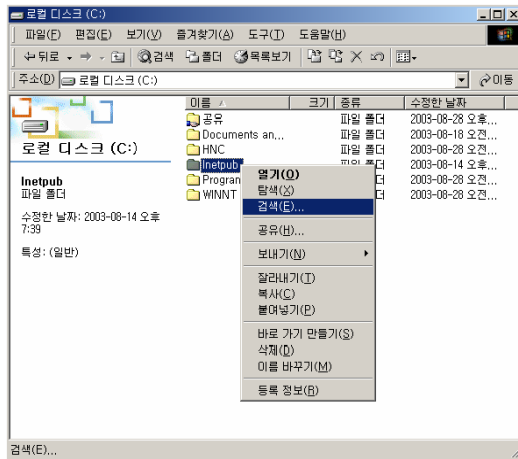


## [IIS 11] 임시 파일의 제거

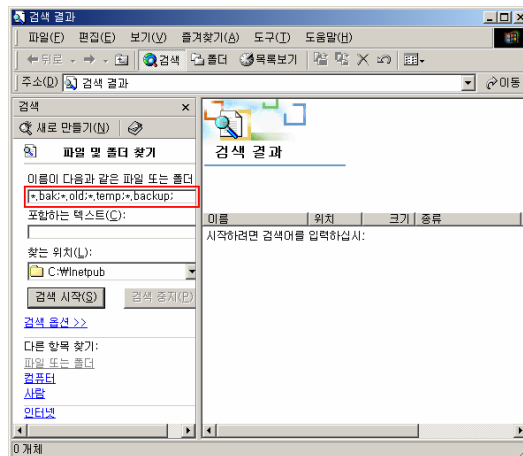
웹 서버를 정기적으로 검사하여 유지보수, 업데이트 등을 위해 임시로 생성했던 파일들을 삭제해주시도록 한다. 정확한 관리를 위해 IIS와 관련된 폴더와 파일의 이름, 위치, 개수 등이 적혀있는 별도의 문서를 관리하는 것이 좋다. 그래서 문서에 등록되지 않은 불필요한 파일들을 점검해서 삭제하도록 한다.

다음과 같은 방법으로 불필요한 파일들을 찾아서 삭제하도록 한다.

- ① 웹 서비스에 사용되는 폴더에 대해 마우스 오른쪽 버튼을 누른 후 [검색]을 선택한다.



- ② 찾고자 하는 파일들의 확장자를 입력한 후 [검색 시작] 버튼을 누른다. (예, bak, log, tmp, sav 등)



- ③ 검색된 파일들 중에 불필요한 파일이 있는지 점검하여 삭제하도록 한다.
- ④ 확장자로 검색하는 것 외에도 웹 서비스에 사용되는 폴더들을 일일이 검사하여 불필요한 파일들을 삭제하도록 한다.

**[IIS 12] Windows 이벤트 로그 점검**

IIS는 Windows와 통합된 서비스이기 때문에, 관리자들은 웹 서버 보안을 위해 IIS의 로그뿐만 아니라 Windows의 로그도 분석할 필요가 있다.

MS사에서는 IIS 5.0 보안과 관련해서 다음 정책 목록에 대해 감사를 수행하도록 권고하고 있다<sup>28</sup>.

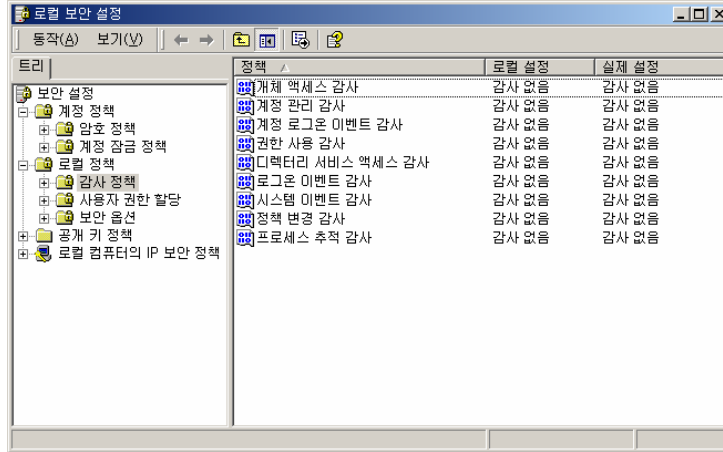
**[표 5-4] IIS 관련 감사를 수행해야 할 정책 목록**

감사를 수행할 정책 목록	시도	
	성공	실패
계정 로그인	사용함	사용함
계정 관리	사용 안 함	사용함
디렉터리 서비스 액세스	사용 안 함	사용함
로그온	사용함	사용함
개체 액세스	사용 안 함	사용 안 함
정책 변경	사용함	사용함
사용 권한 사용	사용 안 함	사용함
프로세스 추적	사용 안 함	사용 안 함
시스템	사용 안 함	사용 안 함

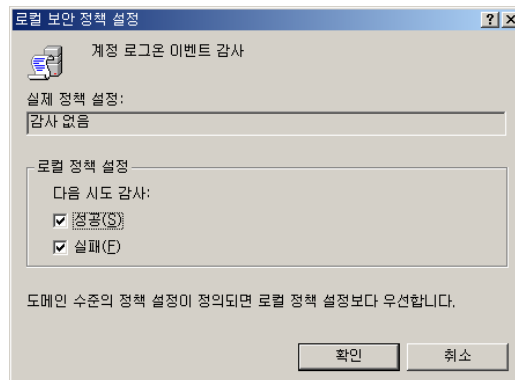
위 정책 목록들에 대해 감사를 설정하는 방법은 다음과 같다.

<sup>28</sup> MS 기술자료 300549 “Windows 보안 감사 설정 및 적용” 참조

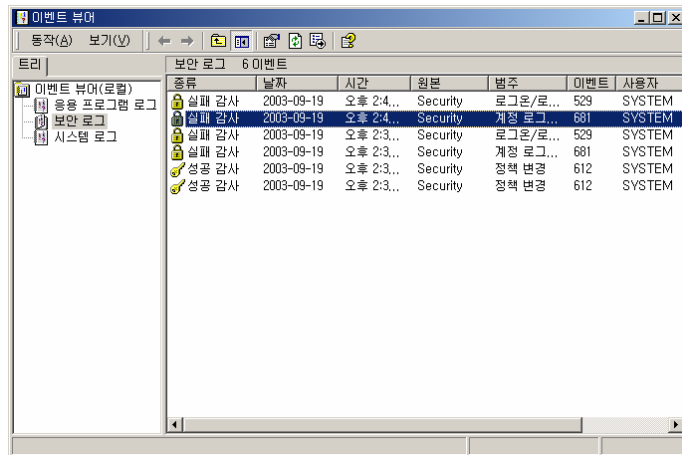
- ① [시작] -> [프로그램] -> [관리 도구] -> [로컬 보안 정책]을 실행시키고, [보안 설정] -> [로컬 정책] -> [감사 정책]을 선택한다.



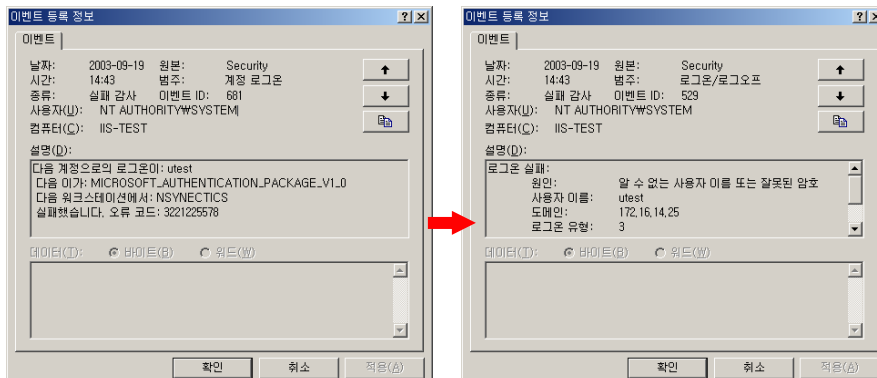
- ② 오른쪽 창에서 설정하거나 해제하려는 정책을 두 번 클릭한다. 예를 들어 “계정 로그인 이벤트 감사” 를 두 번 클릭하면 다음과 같은 설정 창이 뜬다. 여기에서 감사를 수행할 시도를 선택한다.



위와 같은 방법을 통해서 제안된 모든 정책 목록에 대해 감사를 설정한 후에 테스트를 위해 잘못된 아이디와 패스워드로 로그인을 시도하였더니 다음과 같은 로그를 이벤트 뷰어에서 볼 수 있었다. Windows의 이벤트 로그는 [시작] -> [프로그램] -> [관리 도구] -> [이벤트 뷰어]에서 볼 수 있다.



한번의 로그온 시도에 대해 두 개의 이벤트가 발생했는데, 각 이벤트를 두 번 클릭하면 다음과 같은 [이벤트 등록 정보]를 볼 수 있다. 첫번째로 발생한 이벤트는 ‘컴퓨터명이 NSYNECTICS인 컴퓨터에서 utest라는 계정으로 로그온을 시도했는데 실패했다’는 기록이고, 두번째로 발생한 이벤트는 ‘로그온 실패의 원인이 알 수 없는 사용자 이름 또는 잘못된 암호’라는 기록이다.

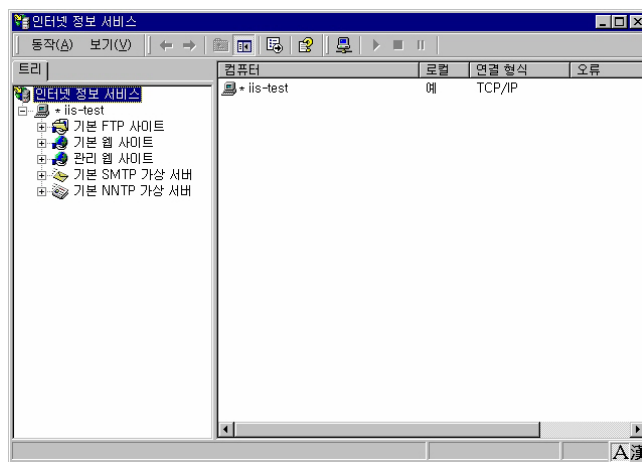




## 제 2 절 IIS 보안 설정

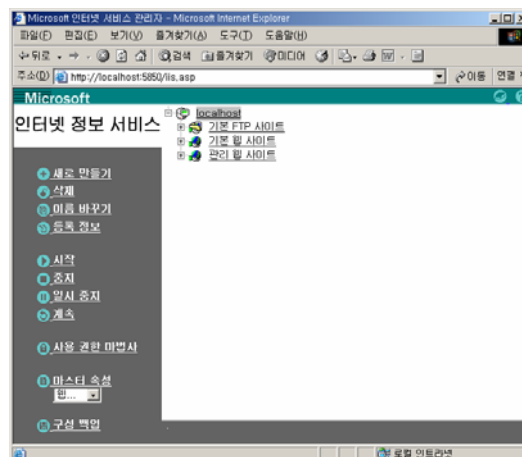
IIS를 관리하기 위한 도구로 인터넷 서비스 관리자가 있다. 인터넷 서비스 관리자는 프로그램 그룹의 관리도구 그룹에 있는데, 이 도구를 통해서 IIS에 대한 대부분의 보안 설정을 수행할 수 있다.

[시작] -> [프로그램] -> [관리 도구] -> [인터넷 서비스 관리자]



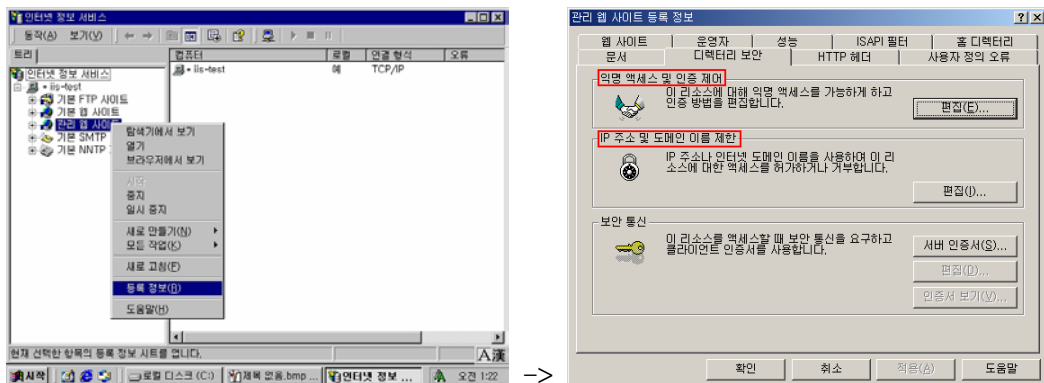
### [IIS 13] HTMLA에 대한 접근 제어

인터넷 서비스 관리자(HTMLA)는 웹 기반의 웹 서버 관리도구로서 원격에서 웹 서버를 관리하는데 사용된다.

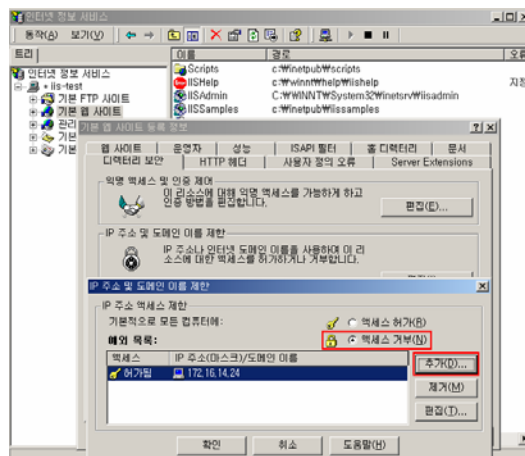


그런데 외부에서 누구든 이 관리 웹사이트에 접근할 수 있다면 웹 서버의 모든 설정을 변경할 수 있어서 치명적인 보안문제를 발생시킬 수 있다. IIS 5.0은 디폴트 설치시 관리 웹사이트에 대한 모든 외부 접근이 불가능하도록 설정되어 있다. 불가피하게도 외부에서 관리 웹사이트에 접근해야 할 필요가 있다면, [IP주소 및 도메인 이름 제한]과 [익명 액세스 및 인증 제어] 기능을 이용하여 접근 가능한 IP주소와 사용자를 제한하도록 한다.

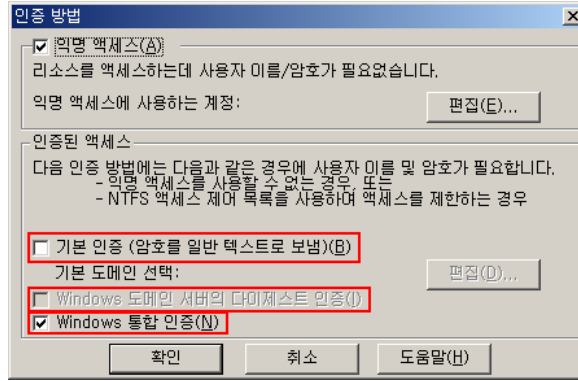
- ① [시작] -> [프로그램] -> [관리 도구] -> [인터넷 서비스 관리자]를 실행하고, [관리 웹사이트]를 마우스 오른쪽 버튼으로 클릭한 후 [등록정보]를 선택한다. 등록 정보에서 [디렉터리 보안] 탭을 선택하면 [IP주소 및 도메인 이름 제한]과 [익명 액세스 및 인증 제어] 기능을 설정할 수 있다.



- ② 우선 [IP주소 및 도메인 이름 제한]에서 [편집]을 클릭하여 특정 컴퓨터(관리자 PC)에서만 접근이 가능하도록 설정한다. 다음 그림처럼 기본적으로 모든 컴퓨터에 대해 [액세스 거부]를 설정하고, 접근을 허용할 IP주소(관리자 PC의 IP주소)를 추가한다. 접근제어에 대한 상세한 설명은 『[IIS 24] 접근제한』 부분을 참고하도록 한다.

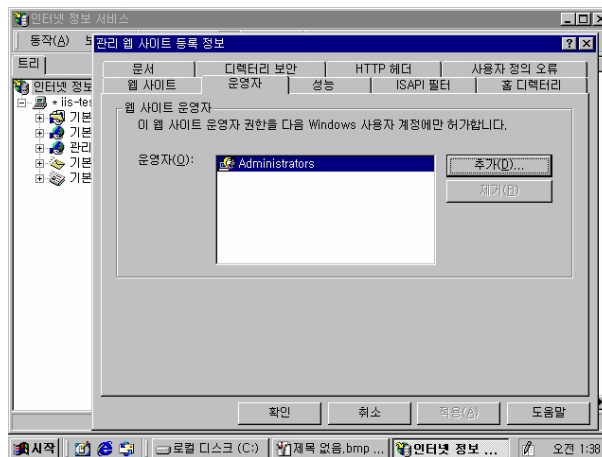


- ③ 다음으로 관리자만 접근이 가능하도록 사용자 인증을 설정하도록 한다. [익명 액세스 및 인증 제어]에서 [편집]을 클릭하면 다음 그림과 같이 3가지 인증방법을 선택할 수 있다. (기본 인증, 다이제스트 인증, 통합인증)



기본 인증방법은 암호를 평문으로 전달하기 때문에 보안상 위험하다. 다이제스트 인증은 암호를 해시값으로 전달하기 때문에 기본 인증 방법에 비해 비교적 안전하다. 통합인증<sup>29</sup>은 네트워크를 통해 암호를 전달하지 않고 자신이 암호를 알고 있음을 증명하는 방식이기 때문에 보안상 안전하다. 따라서 가급적 인증방법을 다이제스트 인증이나 통합인증을 선택하도록 한다. 인증에 대한 상세한 설명은 『[IIS 23] 인증』 부분을 참고하도록 한다.

- ④ 마지막으로 웹사이트 운영자 권한을 가급적 기본설정인 Administrators만 허용하도록 한다.

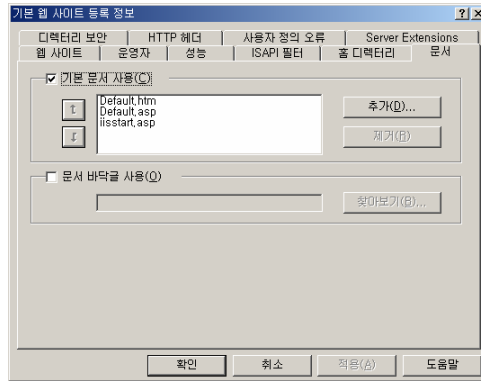


<sup>29</sup> 통합인증은 브라우저로 반드시 익스플로러만을 사용해야 한다는 단점이 있다.

## [IIS 14] 기본 문서 설정

기본 문서는 브라우저 요청에서 디렉토리만 지정하고 문서 이름을 지정하지 않았을 때 기본적으로 보여지는 문서를 말한다. 예를 들어 기본 문서로 default.htm이 지정되어 있는 경우 `http://your.server.com/info/`를 요구하면 IIS는 `http://your.server.com/info/default.htm`을 리턴해준다.

기본 문서를 설정하기 위해서는 [시작] -> [프로그램] -> [관리 도구] -> [인터넷 정보 서비스]를 실행시키고, [기본 웹 사이트]에 대해 마우스 오른쪽 버튼을 클릭한 후 [등록정보]를 선택한다. 다음 그림과 같이 [등록정보]에서 [문서]탭을 선택하면, 기본 문서를 추가 혹은 제거할 수 있다.



기본 문서가 여러 개 지정된 경우 리스트에서 상위에 있는 문서가 우선 순위가 높다. 위 그림의 경우 Default.htm이 존재하면 그 파일을 리턴해주고, Default.htm이 존재하지 않으면 Default.asp 파일을 리턴해준다. Default.asp 파일도 없으면 iisstart.asp 파일을 찾아서 리턴해준다.

그런데 여기서 리스트의 순서에 주의해야 한다. 위와 같이 설정한 상태에서 디렉토리에 Default.asp 파일만 존재하고 Default.htm 파일이 존재하지 않는 경우, 만약 공격자가 Default.htm 파일을 업로드 한다면 관리자가 의도했던 Default.asp 대신 공격자가 업로드한 Default.htm 파일이 처리된다.

따라서 기본 문서의 목록이 적절하게 배열되어 있는지 점검하도록 한다.

[IIS 15] 모든 예제 응용 프로그램을 제거

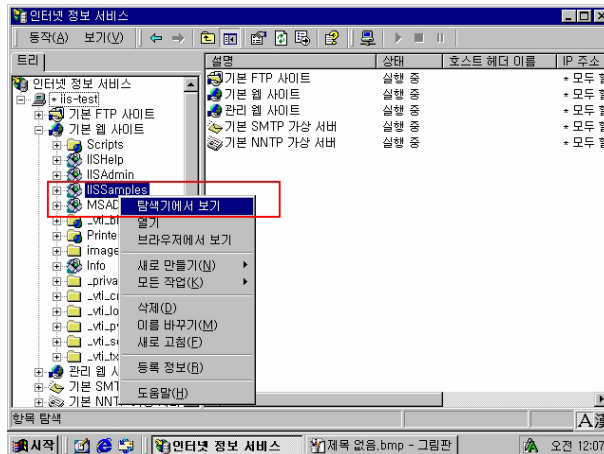
IIS를 설치하면 기본적으로 예제와 설명서 등이 같이 설치된다. 이 폴더들은 해킹에 이용되거나 백도어가 심어질 위험이 있으므로 제거해 주어야 한다. [표 5-5]는 IIS 예제들이 저장되는 기본 위치이다.

[표 5-5] IIS 예제들의 기본 위치

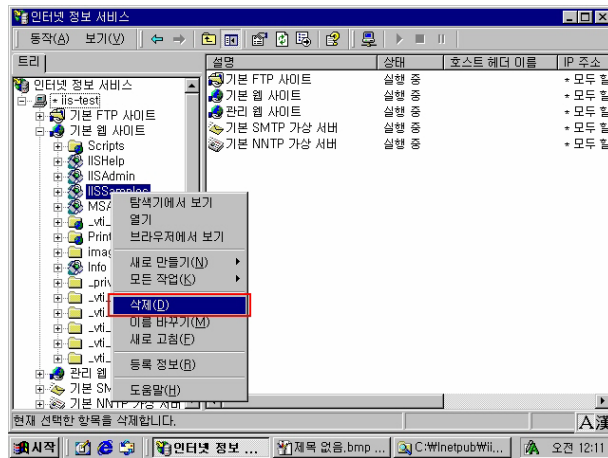
예제	가상 디렉토리	위치
IIS 예제	WIISamples	c:\WinetpubWiissamples
IIS 설명서	WIISHelp	c:\Wwinnt\help\wiishelp
데이터 액세스	WMSADC	c:\Wprogram files\common files\system\wmsadc

이러한 가상 디렉토리와 실제폴더를 삭제하는 방법은 다음과 같다.

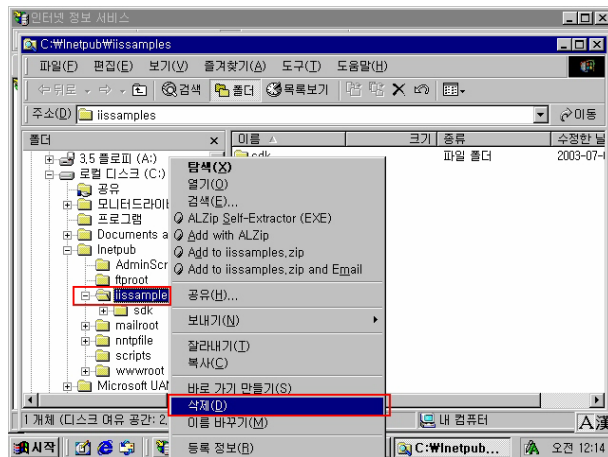
- ① [인터넷 서비스 관리자]에서 삭제하려는 가상 디렉토리 (여기에서는 “IISamples”)를 선택하고, 마우스 오른쪽 버튼을 클릭한 후 [탐색기에서 보기]를 선택한다. 그러면 해당 디렉토리를 보여주는 탐색기 창이 뜨게 된다.



② 우선 [인터넷 서비스 관리자]에서 다음과 같이 가상 디렉토리를 삭제한다.



③ 그리고 나서 탐색기로 열어두었던 실제 디렉토리도 제거한다.



### [IIS 16] iisadmpwd 가상 디렉토리 제거

iisadmpwd 디렉토리는 IIS 4.0 설치시 기본적으로 생성되는 가상 디렉토리로서, 원격에서 웹 서버를 통하여 Windows NT와 Windows 2000의 암호를 재설정할 수 있게 만든다. 이는 주로 인트라넷 환경을 전제로 설계된 기능으로, IIS 5 설치시에는 기본적으로 설치되지 않지만, IIS 4서버를 IIS 5로 업그레이드 할 때 제거되지 않고 남아 있다. 인트라넷을 사용하지 않거나 서버를 웹에 연결할 경우 이 가상 디렉토리를 제거한다. 가상 디렉토리 제거 방법은 『[IIS 15] 모든 예제 응용 프로그램을 사용하지 않거나 제거』를 참조한다.

## [IIS 17] 사용하지 않는 스크립트 매핑 제거

사용하지 않는 스크립트 매핑은 보안에 위협이 될 수 있으므로 개발자와 협의하여 불필요한 매핑인지 확인한 후에 제거하도록 한다.

.asp나 .shtm 과 같은 확장자들은 특정 DLL 파일과 매핑되어 처리되어진다. 이러한 매핑가운데는 사용되지 않는 것들이 있어서 이를 제거해주는 것이 보안에 도움이 된다.

대부분의 사이트에서 다음 매핑들을 제거하도록 권장한다. 하지만 실제 사이트에 적용하기 전에 해당 매핑을 제거해도 되는 것인지 개발자와 협의를 한 후에 제거해야 한다.

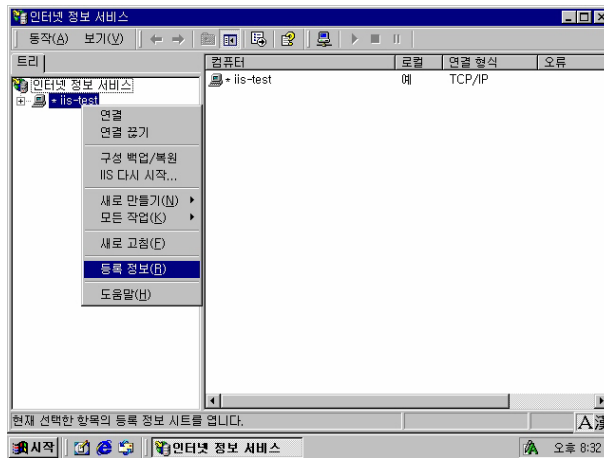
[표 5-6] IIS에서 제거를 권장하는 매핑 목록

IIS가 사용하지 않는 것	제거할 매핑
웹 기반 암호 재설정	.htr
인터넷 데이터베이스 커넥터	.idc
Server-side Includes	.stm, .shtm, .shtml
인터넷 인쇄	.printer
인덱스 서버	.htw, .ida, .idq

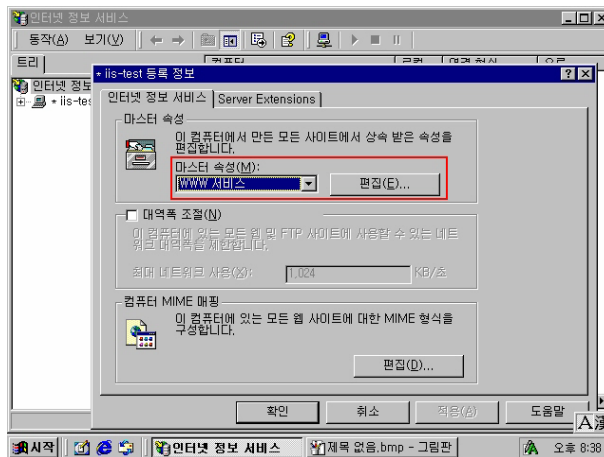
[참고] 인터넷 인쇄 기능은 인터넷 서비스 관리자뿐만 아니라 그룹 정책을 통해서도 구성할 수 있다. 그룹 정책 설정과 인터넷 서비스 관리자 설정 간의 충돌이 있을 경우, 그룹 정책 설정이 우선권을 가진다. 인터넷 서비스 관리자에서 인터넷 인쇄 기능을 제거할 경우, 로컬 또는 도메인 그룹 정책에 의해 다시 사용되지 않는지 확인하도록 한다.

다음 절차에 따라 사용하지 않는 스크립트 매핑을 제거한다.

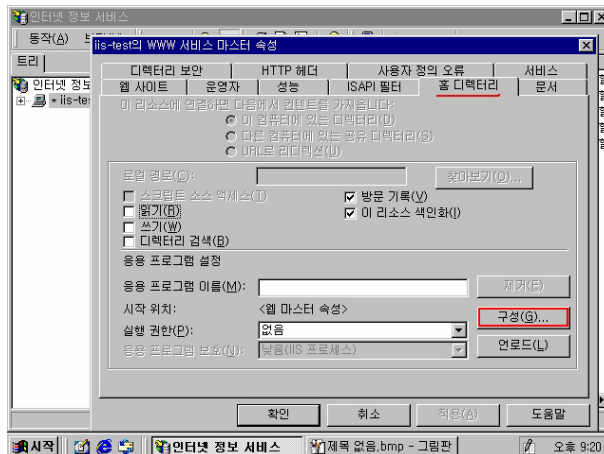
- ① [시작] -> [프로그램] -> [관리 도구] -> [인터넷 정보 서비스]를 열고, 웹 서버에 대해 마우스 오른쪽 버튼을 클릭한 후 [등록 정보]를 선택한다.



- ② 등록 정보에서 [인터넷 정보 서비스] 탭을 선택한 후, [마스터 속성]으로 “WWW 서비스”를 선택하고 [편집]을 클릭한다.

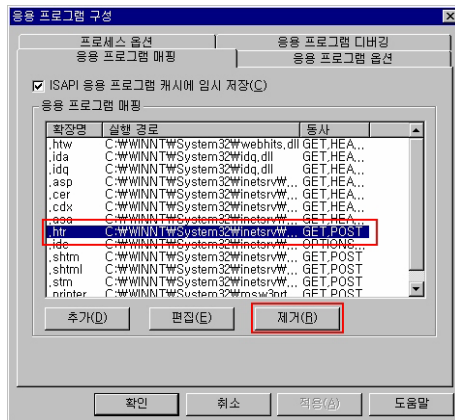


- ③ WWW 서비스 마스터 속성에서 [홈 디렉토리] 탭을 선택하고 [구성]을 클릭한다.





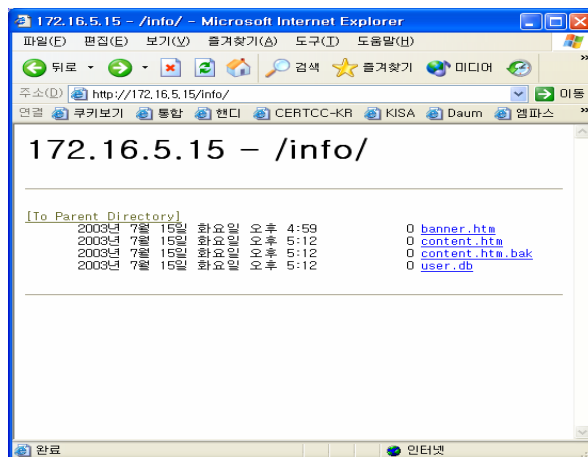
- ④ [응용 프로그램 구성]에서 [응용 프로그램 매핑] 탭을 선택하고, 사용하지 않는 스크립트 매핑을 찾아서 선택한 후 [제거] 버튼을 클릭한다.



- ⑤ 사용하지 않는 스크립트 매핑을 제거한 후, [확인]을 클릭하여 변경사항을 저장한다. IIS서비스를 재시작하면 변경사항이 적용된다.

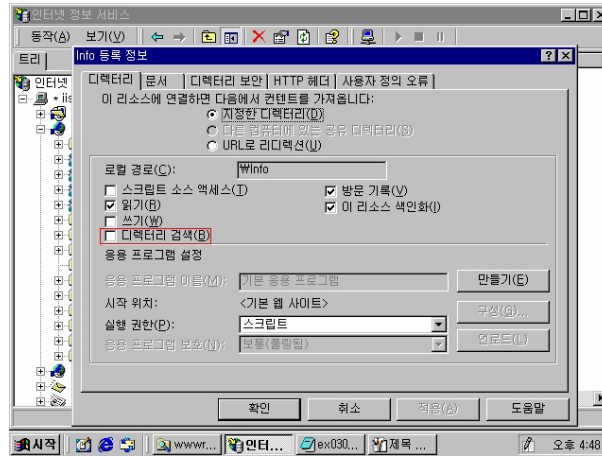
## [IIS 18] 디렉토리 검색

디렉토리 검색은 디렉토리에 대한 요청시 그 디렉토리에 기본 문서가 존재하지 않을 경우 디렉토리 내에 존재하는 모든 파일들의 목록을 보여주는 것을 말한다.

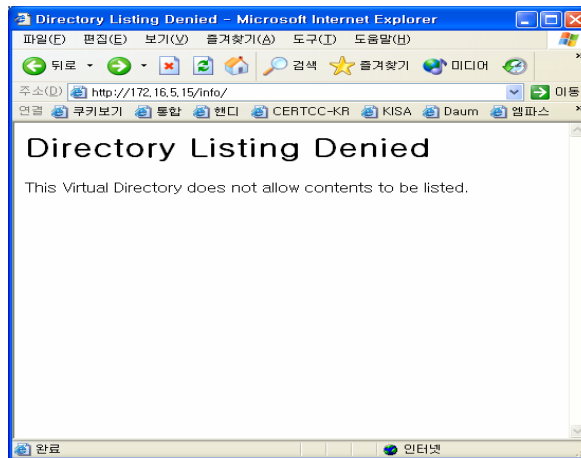


디렉토리 검색이 허용되면 외부에서 디렉토리 내의 모든 파일에 대한 접근이 가능하여 백업 파일이나 소스 파일 등 공개되어서는 안되는 중요한 파일들이 노출될 수 있다. 따라서 다음과 같은 방법으로 디렉토리 검색이 불가능하도록 설정한다.

[시작] -> [프로그램] -> [관리 도구] -> [인터넷 정보 서비스]를 실행시킨 후 [기본 웹 사이트]에 대해 마우스 오른쪽 버튼을 클릭한 후 [등록정보]를 선택한다. [디렉토리]탭에서 [디렉토리 검색]을 다음 그림과 같이 체크되지 않은 상태를 유지하도록 한다.



이렇게 하면 해당 디렉토리에 대한 접근시 다음과 같이 접근불가 메시지를 보여 준다.

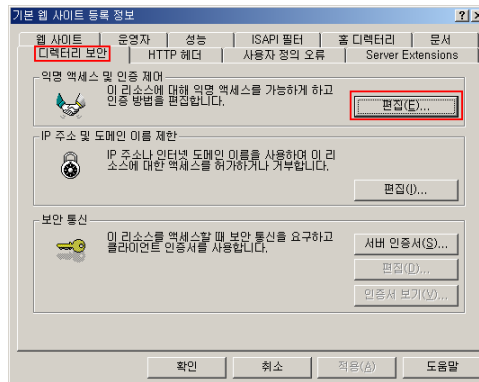


## [IIS 19] 익명 사용자 계정 권한 제한

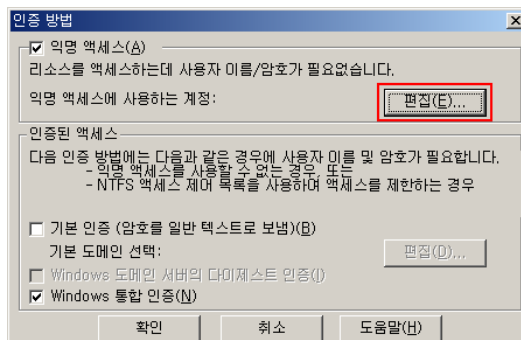
익명 사용자 계정은 익명 액세스를 허용하는 경우에 방문자가 사용하게 되는 Windows 2000 사용자 계정이다. 기본적으로 IIS를 설치하면 IUSR\_서버이름 계정이 생성된다. IUSR\_서버이름 계정은 가장 제한적인 권한만을 갖고 있어야 한다.

익명 사용자 계정은 다음 위치에서 설정할 수 있다.

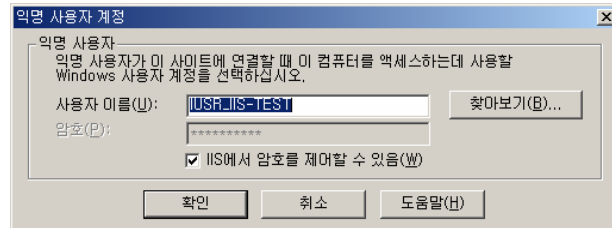
- ① [시작] -> [프로그램] -> [관리 도구] -> [인터넷 정보 서비스]를 실행하고, 기본 웹 사이트에 대해 마우스 오른쪽 버튼을 클릭한 후 [등록정보]를 선택한다.
- ② [기본 웹 사이트 등록 정보]에서 [디렉토리 보안] 탭을 선택한 후 [익명 액세스 및 인증 제어]의 [편집] 버튼을 클릭한다.



- ③ [인증 방법]에서 [익명 액세스]의 [편집] 버튼을 클릭한다.



- ④ 익명 사용자 이름이 디폴트로 IUSR\_서버이름으로 설정되어 있는 것을 볼 수 있다. 익명 사용자 이름으로 권한 있는 계정을 설정하지 않도록 유의한다.



여기에서 [IIS에서 암호를 제어할 수 있음]을 선택하면 사용자 관리자에서 IUSR\_서버이름 계정의 암호를 변경할 경우 자동으로 수정된 암호가 웹사이트에 반영된다. 이 옵션을 선택하지 않으면 사용자 관리자에서 익명 사용자 계정의 암호를 변경할 때 이곳에서도 암호를 변경해 주어야 한다.

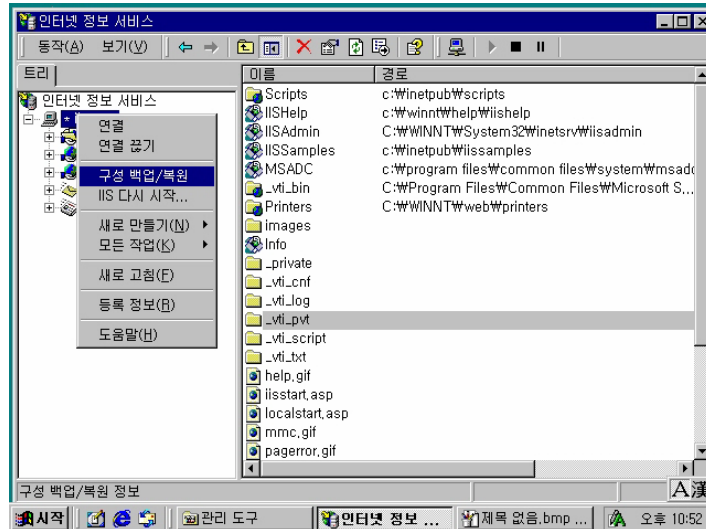
IIS 익명 액세스에 사용되는 계정의 이름을 IUSR\_서버이름에서 다른 이름으로 변경하여 외부에서 추측할 수 없도록 하는 것이 좋다. 그리고 이 계정에 대한 원격 로그인이 불가능하도록 설정하는 것이 좋다.

## [IIS 20] 웹 서버의 설정값 백업

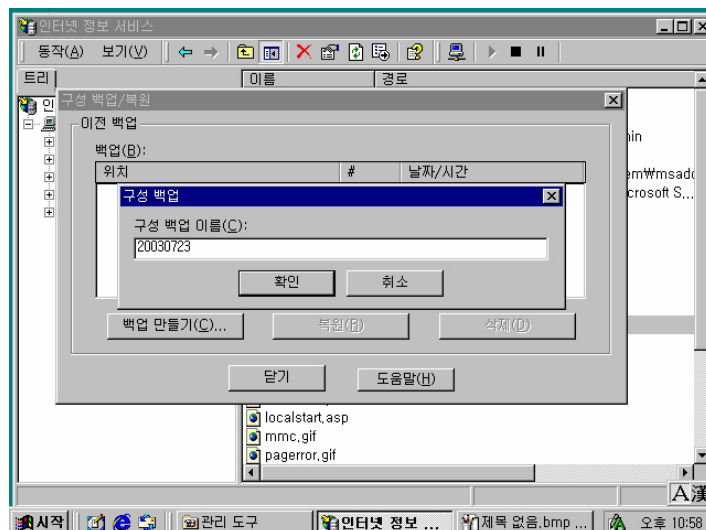
IIS는 설정정보를 IIS 메타베이스에 저장한다. 메타베이스는 Windows 레지스트리와 비슷하지만 IIS에 한정된다. IIS가 재설치 되거나 재설정되어야 하는 경우, 백업받은 메타베이스를 이용해서 빠르게 IIS를 원상 복구할 수 있다.

메타베이스의 백업 및 복원 방법은 다음과 같다.

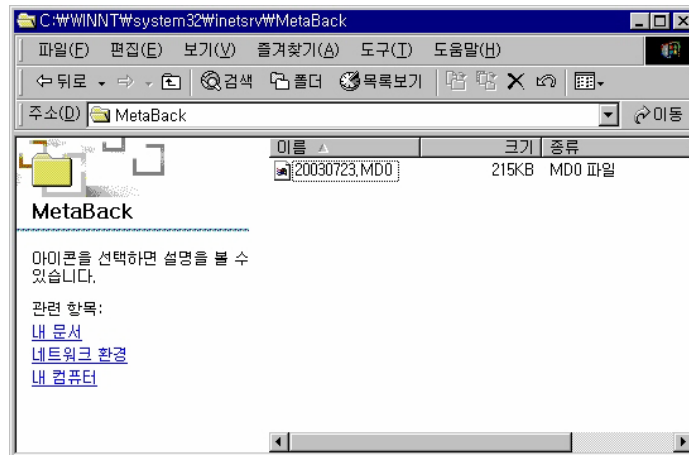
- ① [시작] -> [프로그램] -> [관리 도구] -> [인터넷 정보 서비스]를 실행하고, 서버를 마우스 오른쪽 버튼으로 클릭한 후 [구성 백업/복원]메뉴를 선택한다.



- ② [구성 백업/복원] 대화상자에서 [백업 만들기]버튼을 클릭하면 다음 그림과 같은 구성 백업 대화상자가 나타난다. 현재 웹 서비스의 설정을 저장할 파일의 이름을 입력한다. 여기에서는 백업날짜를 입력하였다. [확인]버튼을 누르고 백업 목록에 백업 파일이 만들어진 것을 확인한 후 [닫기]버튼을 클릭하여 상자를 닫는다.



메타베이스 백업 파일은 C:\WINNT\system32\inet\_srv\MetaBack에 저장된다.



복원은 백업절차와 동일하다. [구성 백업/복원] 대화상자에서 [백업 만들기]버튼 대신 [복원] 버튼을 누르면 된다.

IIS 설정의 백업은 허가받은 관리자만 접근할 수 있는 안전한 영역에 저장되어야 한다.

#### [IIS 21] MBSA를 사용하여 스캐닝

MBSA(Microsoft Base Line Security Analyzer)는 잘못된 보안설정과 적용되지 않은 핫픽스를 식별하는 스캐너이다<sup>30</sup>. 스캔 대상은 다음과 같다.

- Windows NT 4.0과 Windows 2000, Windows XP,
- SQL Server 7.0과 2000
- IE 5.01과 그 이후 버전
- Office 2000과 2002
- IIS 4.0, 5.0

<sup>30</sup> MBSA는 HFNetChk(Microsoft Network Security Hotfix Checker)를 사용해서 적용되지 않은 핫픽스와 서비스팩을 식별한다.

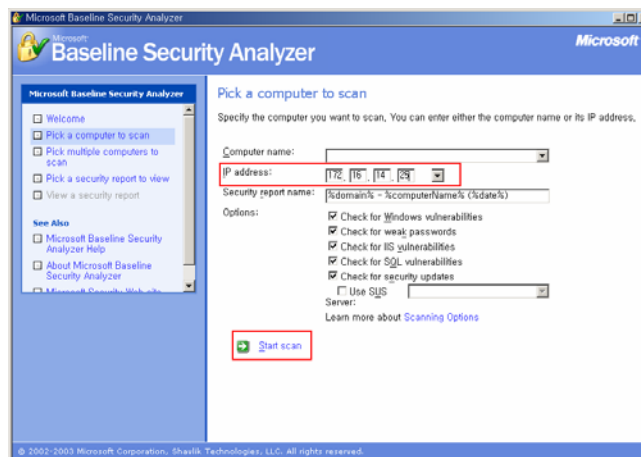
마이크로소프트사 한국어 사이트에 들어가서 “MBSA” 로 검색하면 MBSA를 다운로드 받을 수 있는 페이지를 찾을 수 있다. MBSA를 다운받아 설치하고 다음과 같은 방법으로 스캐닝을 수행하도록 한다.

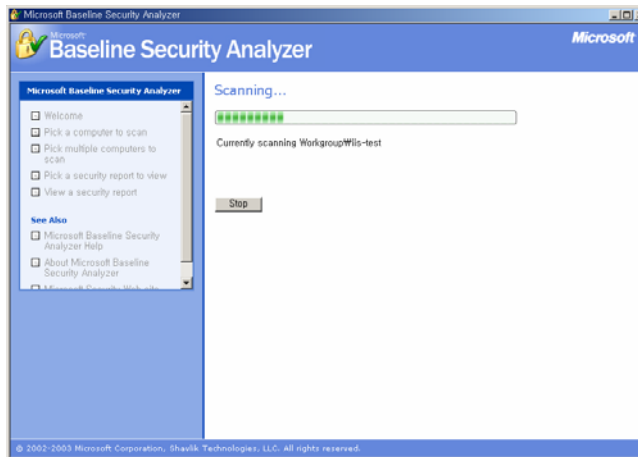
다음은 Windows 2000 Server와 IIS 5.0만을 디폴트로 설치하고 1주일 전에 Windows Updates를 수행한 시스템에 대해 MBSA 스캐닝을 수행하는 과정이다.

① MBSA를 실행시키고 [Scan a computer]를 선택한다.(IIS 웹 서버가 현대인 경우)

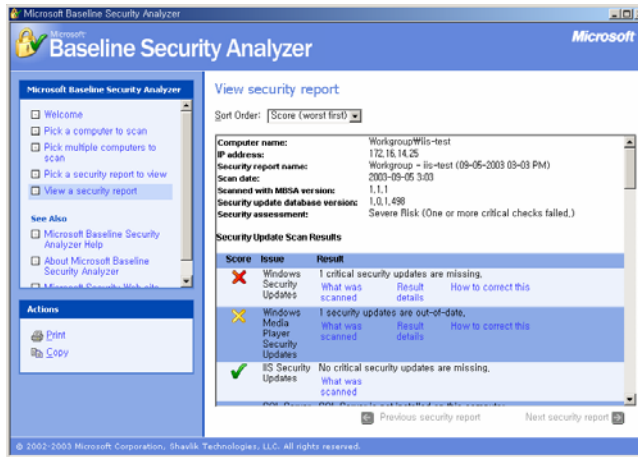


② 웹 서버의 컴퓨터의 IP주소를 입력한 후 [Start scan]을 클릭한다.

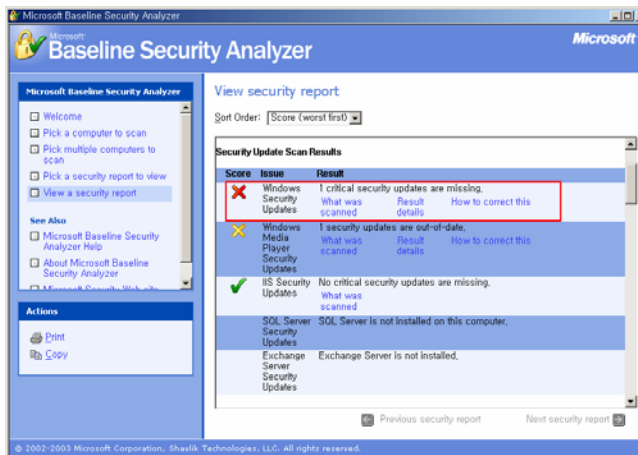




스캔 결과화면은 다음과 같다. 스캔결과는 Windows와 IIS에 대해서 미수행된 중요 보안 업데이트 정보와 발견된 취약점 정보들을 보여준다.

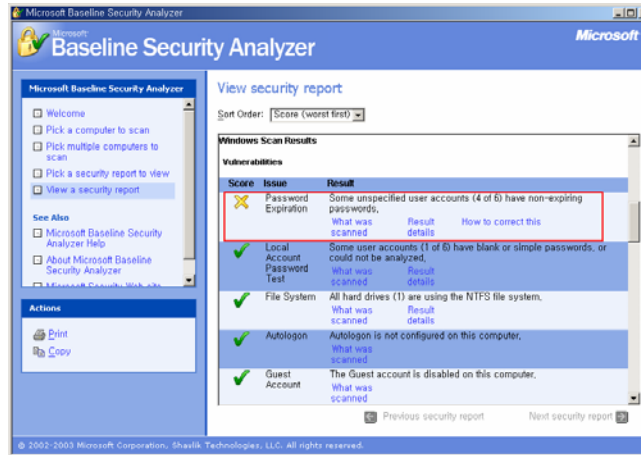


Security Update Scan Results : 수행되지 않은 Windows Security updates가 하나 있고, IIS Security Updates는 없는 것으로 나타났다.

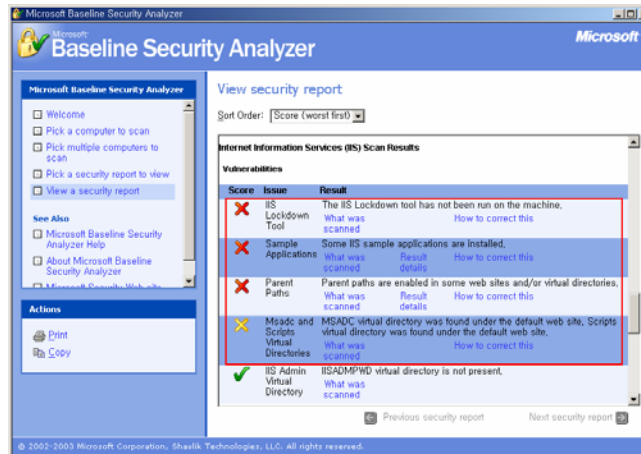




Windows Scan Results : 4개의 사용자 계정이 패스워드 만료가 설정되어 있지 않는 것으로 나타났다.



IIS Scan Results : IIS Lockdown 도구가 실행되지 않고 있고, 일부 IIS 샘플 어플리케이션이 설치되어 있으며, Parent paths가 설정되어 있는 곳이 몇 군데 있는 것으로 나타났다.



각 문제점에 대해서 “How to correct” 를 클릭하면 이 문제를 해결하기 위해 무엇을 어떻게 해야 하는지 상세한 설명을 볼 수 있다.

## [IIS 22] MS사의 보안 알람 서비스 가입

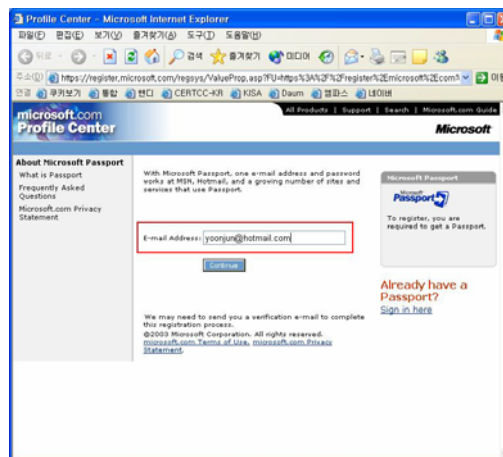
적용해야 할 새로운 패치가 없도록 끊임없이 새로운 취약점과 패치 정보에 주의 를 기울이도록 한다<sup>31</sup>. 이를 위해 MS사에서 제공하는 알람 서비스에 가입하도록 권 고한다.

가입 방법은 다음과 같다.

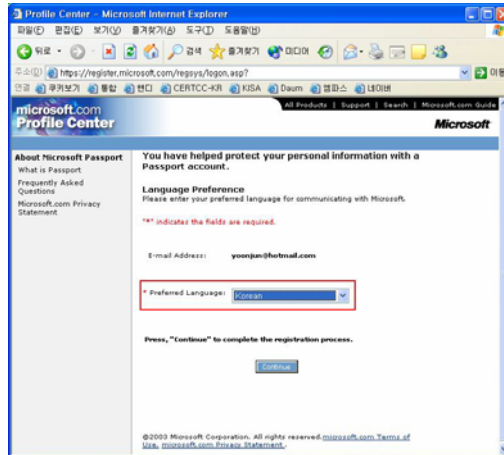
- ① <http://www.microsoft.com/korea/technet/default.asp> 를 방문한다.
- ② 오른쪽에 보안센터의 보안뉴스레터 신청을 클릭한다.



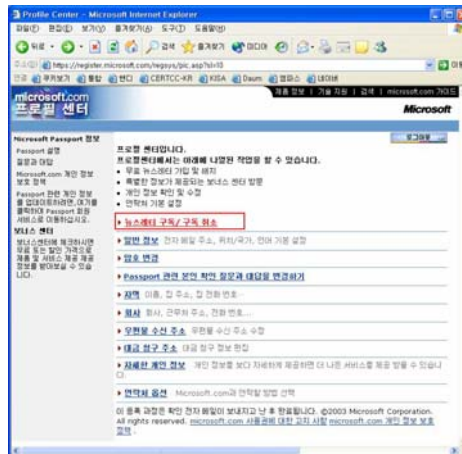
- ③ 로그인 창이 나오면 .net passport를 입력하고 확인을 누르면 언어 설정하는 부 분이 나온다.



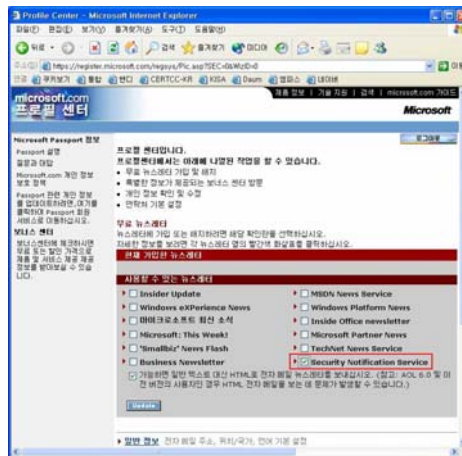
<sup>31</sup> 가급적 운영중인 서버에 직접 적용하지 말고 별도의 테스트 서버를 이용하여 충분한 테스트를 거 친 후에 적용하는 것이 좋다.



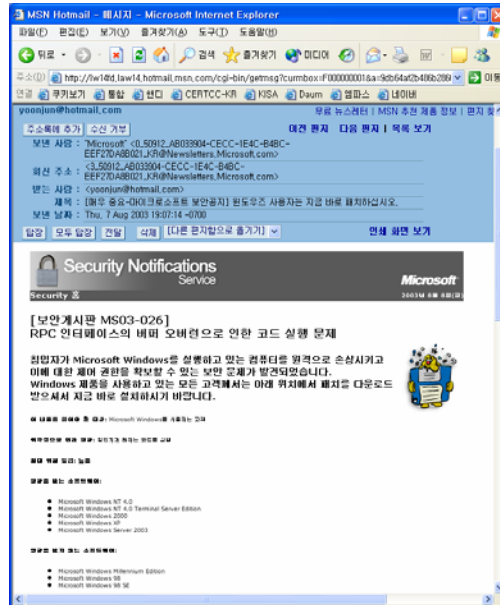
④ 프로필 센터가 나오면 뉴스레터 구독/구독 취소를 선택한다.



⑤ Security Notification Service를 선택하고 Update 버튼을 누른다.



다음은 위 절차에 따라 등록된 메일 계정으로 새로 발견된 보안 문제에 대한 패치가 릴리즈 되었음을 알리는 메일이 도착한 화면이다.



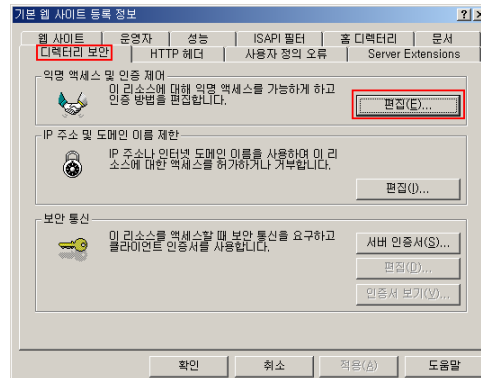
## 제 3 절 인증 및 접근제어

### [IIS 23] 인증

IIS에서는 웹 사이트의 허가를 얻기 위한 인증 방법으로 3가지를 사용할 수 있는데, 기본 인증, 다이제스트 인증, Windows 통합 인증이 있다.

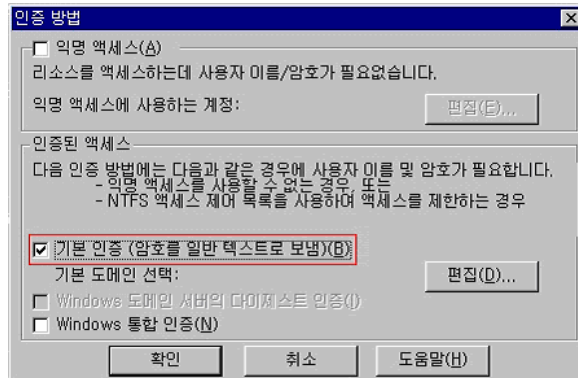
인증방법을 선택 및 설정하려면 다음과 같은 절차를 따른다.

- ① [시작] -> [프로그램] -> [관리 도구] -> [인터넷 정보 서비스]를 실행하고, 기본 웹 사이트에 대해 마우스 오른쪽 버튼을 클릭한 후 [등록정보]를 선택한다.
- ② [기본 웹 사이트 등록 정보]에서 [디렉토리 보안] 탭을 선택한 후 [익명 액세스 및 인증 제어]의 [편집] 버튼을 클릭한다.



- ③ 그리고 나서 다음 인증방법 중 한가지를 선택한다.

## ■ 기본 인증



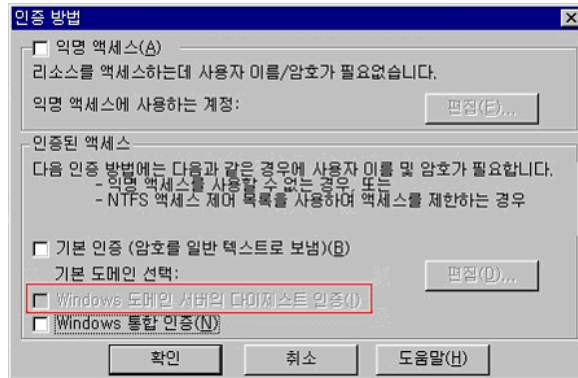
웹 서버에 사용자 계정이 등록된 사용자들에게만 웹 사이트의 내용을 볼 수 있도록 하는 방법이다. 만약 NTFS를 사용한다면 사용자가 접근하려는 파일에 대해서 적절한 NTFS 사용권한도 있어야 접근이 가능하다.

기본 인증의 단점은 암호를 네트워크로 전송할 때 암호화하지 않고 평문으로 보내기 때문에 패킷 스니핑 방법으로 암호를 쉽게 알아낼 수 있다는 것이다. 따라서 중요한 보안을 요구하는 사이트에서 기본인증 방법만 사용하는 것은 바람직하지 못하다.

기본 인증에 사용되는 사용자 계정이 웹 서버가 속한 도메인과 다른 경우에 어느 도메인의 사용자들을 기본 인증에 사용할 것인지 지정하려면 옆에 위치한 [편집] 버튼을 눌러서 설정한다.

기본 인증을 설정할 때 반드시 익명 액세스를 체크하지 않도록 한다. 기본 인증과 익명 액세스가 함께 체크되면 익명 액세스가 우선한다.

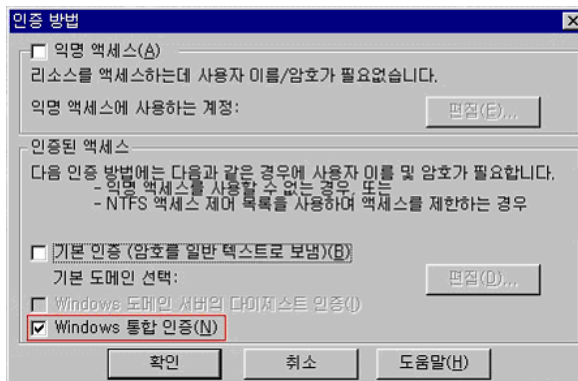
■ 다이제스트 인증



IIS 5.0에서만 지원되는 인증 방법으로 Windows 2000에 등록된 사용자들만 웹사이트를 사용할 수 있다. 네트워크를 통해 암호를 전달하지 않고 해싱방법을 사용하므로 안전하게 사용자를 인증할 수 있다.

다이제스트 인증은 인터넷 익스플로러 5.0 웹 브라우저에서만 사용된다. 또한 다이제스트 인증을 사용하기 위해서는 Windows 2000 도메인 컨트롤러가 필요하다.

■ Windows 통합 인증



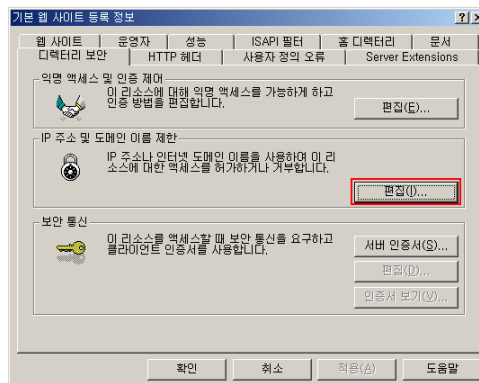
IIS 4.0에서 사용하던 Windows NT Challenge/Response 인증방법으로 이름이 Windows 통합 인증으로 변경되었다. 인터넷 익스플로러 2.0 이상에서만 사용 가능한 인증방법이다. 다이제스트 인증과 마찬가지로 Windows 2000에 등록된 사용자만이 웹사이트를 이용할 수 있다.

기본 인증과 통합 인증이 함께 선택되면 통합인증이 우선 적용된다.

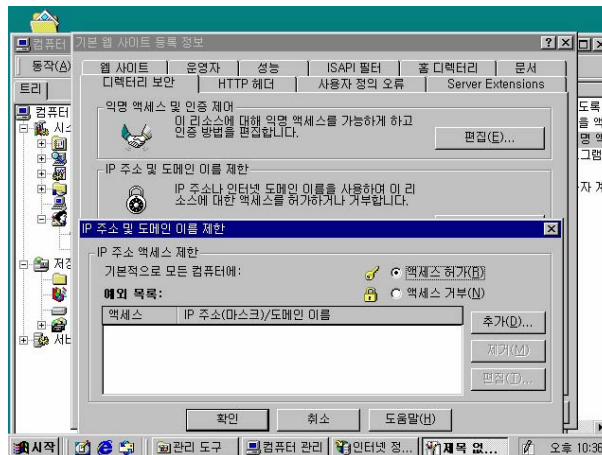
## [IIS 24] 접근제어

IP주소나 도메인 이름에 대해서 접근제한을 설정하는 방법은 다음과 같다.

- ① [시작] -> [프로그램] -> [관리 도구] -> [인터넷 정보 서비스]를 실행하고, 기본 웹 사이트에 대해 마우스 오른쪽 버튼을 클릭한 후 [등록정보]를 선택한다.
- ② [기본 웹 사이트 등록 정보]에서 [디렉토리 보안] 탭을 선택한 후 [IP 주소 및 도메인 이름 제한]의 [편집] 버튼을 클릭한다.



- ③ 기본 액세스 제한 방식을 선택한다.



[액세스 허가]를 선택하면 기본적으로 모든 컴퓨터가 웹사이트에 접근하는 것을 허용한다. 이 경우 예외 목록에는 액세스를 거부하는 컴퓨터의 목록이 나타난다.

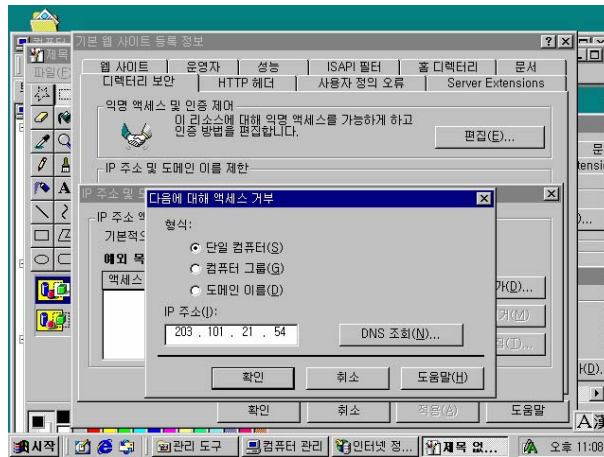
[액세스 거부]를 선택하면 기본적으로 모든 컴퓨터가 웹사이트에 접근하는 것을 막는다. 이 경우 예외 목록에는 액세스를 허용하는 컴퓨터의 목록이 나타난다.



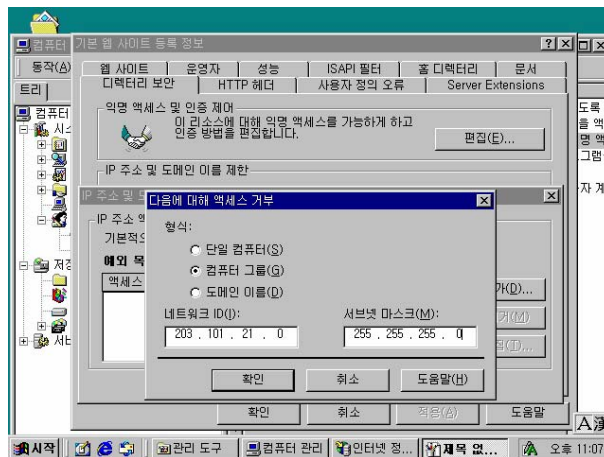
액세스 거부나 허용은 단일 컴퓨터, 컴퓨터 그룹, 도메인 이름에 대해서 설정할 수 있는데 [추가]나 [편집], [제거] 버튼을 이용한다. 여기에서는 기본적으로 모든 컴퓨터의 웹사이트 접근을 허용한 상태에서 특정 컴퓨터 혹은 컴퓨터 그룹, 도메인 이름에 대해 접근을 거부하는 방법을 설명하도록 하겠다. [액세스 허가]를 선택하도록 한다.

④ [액세스 허가]를 선택한 상태에서 [추가]버튼을 선택하면 [단일 컴퓨터], [컴퓨터 그룹] [도메인 이름]을 선택할 수 있다.

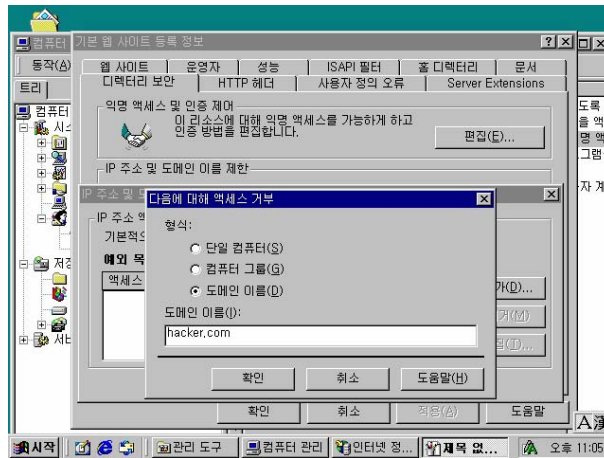
[단일 컴퓨터]는 1대의 호스트만 접근을 거부할 때 사용한다. 해당 컴퓨터의 IP 주소를 지정하는데, 해당 컴퓨터의 IP주소를 모른다면 [DNS 조회]를 통해서 도메인 이름을 IP주소로 변환할 수 있다.



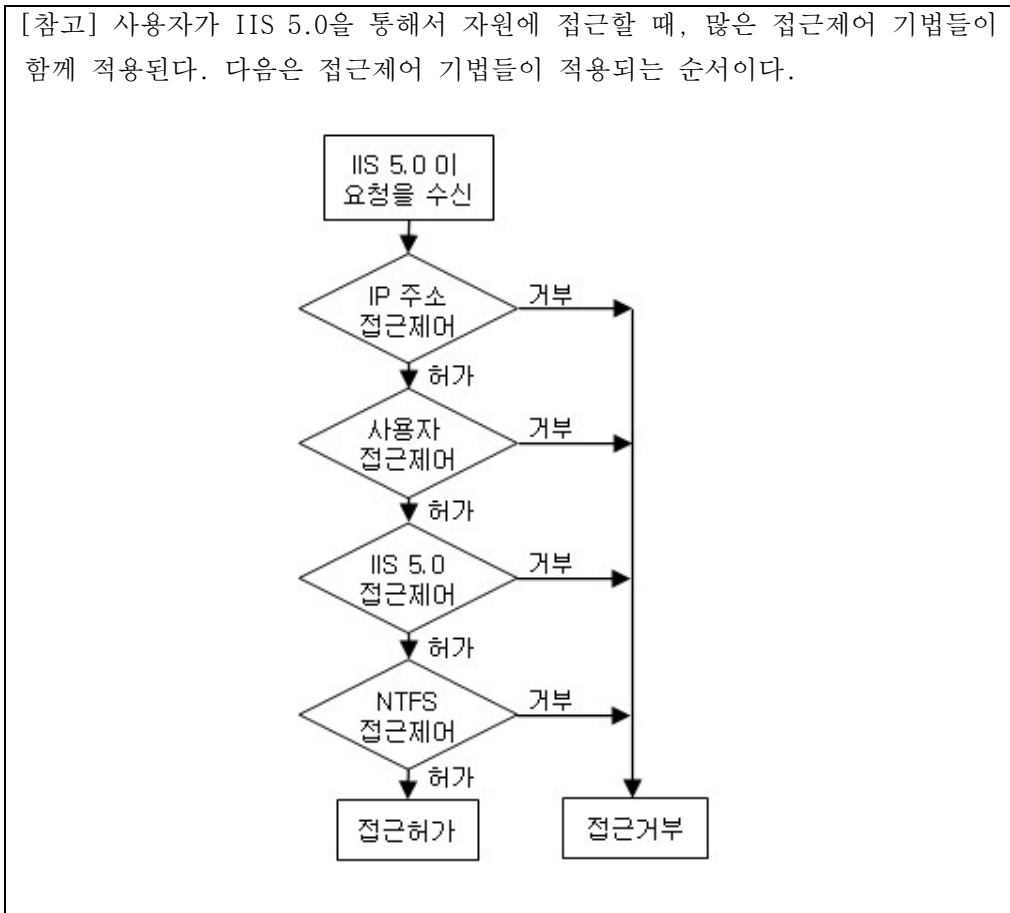
[컴퓨터 그룹]은 네트워크 ID와 서브넷 마스크를 사용해서 특정 IP그룹의 접속을 제한할 때 사용한다.



[도메인 이름]은 해당 도메인 이름을 사용하는 모든 호스트들의 접근을 제한할 때 사용한다.



[참고] 사용자가 IIS 5.0을 통해서 자원에 접근할 때, 많은 접근제어 기법들이 함께 적용된다. 다음은 접근제어 기법들이 적용되는 순서이다.



## 제 4 절 로깅

### [IIS 25] 로그 설정 및 분석

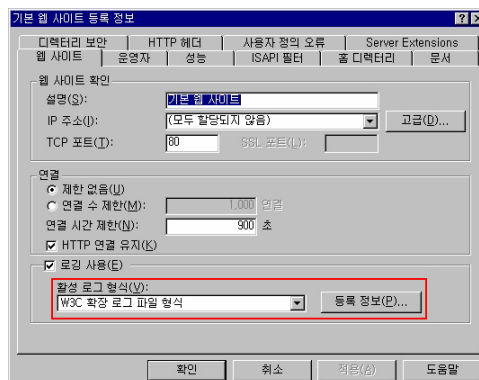
관리자는 IIS의 로깅 기능을 이용하여 사이트 방문자, 방문자의 활동 등의 정보를 모니터링할 수 있다. 모든 침입 과정 및 접근 정보들이 로그에 기록되기 때문에 로그는 웹 서버 보안에 있어서 중요한 요소이다. 로그 파일을 주기적으로 검토하고 백업 및 안전하게 관리할 필요가 있다.

IIS 5.0에서는 4가지 로그 형식을 선택할 수 있다.

- Microsoft IIS 로그 파일 형식
- NCSA 공통 로그 파일 형식
- ODBC 로깅<sup>32</sup>
- W3C 확장 로그 파일 형식

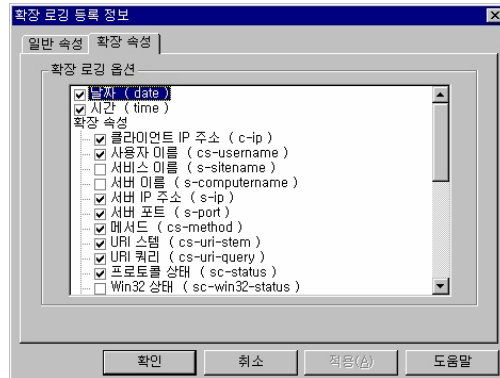
이 중 IIS 로그 형식으로 가장 널리 사용되고 있고 추천되는 로그 형식은 “W3C 확장 로그 파일 형식”이다. 본 문서에서는 이 로그 형식의 사용에 대해 설명하도록 한다. 로그 설정방법은 다음과 같다.

- ① [시작] -> [프로그램] -> [관리 도구] -> [인터넷 정보 서비스]에서 [기본 웹 사이트]를 마우스 오른쪽 버튼으로 클릭 한 후 [등록정보]를 선택한다.
- ② [웹 사이트]탭에서 [로깅 사용]을 선택하고, 활성 로그 형식으로 “W3C 확장 로그 파일 형식”을 선택한다.



<sup>32</sup> 데이터베이스에 로깅되는 고정 형식이다. 별도의 SQL서버에 로깅을 하는 경우 네트워크문제가 발생하면 로그 파일이 손상되거나 수행 성능에 문제가 발생할 수 있다.

③ 로그 형식 옆의 [등록정보]를 누른 후 [확장 속성]탭에서 최소한 다음 속성들을 선택하도록 한다.



[확장 속성]탭에서 반드시 선택해야 하는 속성들

필드 이름
클라이언트 IP주소
사용자 이름
메소드
URI 스템
HTTP 상태
Win32 Error
사용자 에이전트
서버 IP주소
서버 포트

마지막 두 속성은 한 대의 컴퓨터에서 여러 웹 서버를 호스트할 경우에만 사용하도록 한다.

각 속성에 대한 설명은 [표 5-7]과 같다.

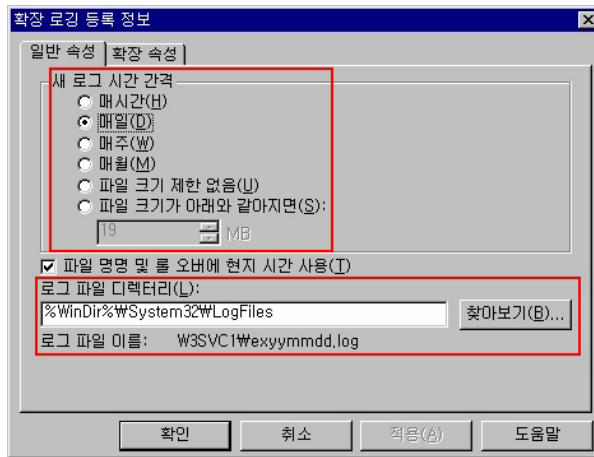
[표 5-7] W3C 확장 로그 파일 형식의 속성

속성	설명
날짜(date)	클라이언트의 접속 날짜(2002-12-25)
시간(time)	클라이언트의 접속 시간(01:10:48)
클라이언트 IP주소(c-ip)	서버를 액세스한 클라이언트의 IP 주소
사용자 이름(cs-username)	서버에 액세스한 사용자 이름(-는 anonymous를 의미)
서비스 이름(s-sitename)	클라이언트에서 실행한 인터넷 서비스 이름 (W3SVC1은 기본 웹 사이트를 의미)
서버 이름(s-computername)	로그가 만들어진 웹 서버 이름(서버가 2대 이상 일 경우)
서버 IP주소(s-ip)	로그가 만들어진 웹 서버의 IP주소
서버 포트(s-port)	클라이언트가 연결한 웹 서버의 TCP 포트번호
메소드(cs-method)	클라이언트가 실행한 메소드
URI 스템(cs-uri-stem)	클라이언트가 액세스한 자원
URI 쿼리(cs-uri-query)	클라이언트가 시도한 쿼리
프로토콜 상태(sc-status)	HTTP 프로토콜의 작동상태
Win32 상태(sc-status) <sup>33</sup>	Windows 2000의 작동상태
보낸 바이트(sc-byte)	웹 서버가 클라이언트에게 보낸 바이트 수
받은 바이트(cs-byte)	클라이언트가 웹 서버에게 보낸 바이트 수
걸린시간(time-taken)	작업에 걸린 시간
프로토콜 버전(cs-version)	클라이언트가 사용한 프로토콜 버전
호스트(cs-host)	클라이언트의 호스트이름
사용자 에이전트(cs user-agent)	클라이언트가 사용하는 브라우저
쿠키(cs cookie)	보내거나 받은 쿠키의 내용
참조 페이지(cs referrer)	이 웹사이트로 이동하기 위해서 사용자가 클릭한 링크가 있던 웹 사이트
프로세스 이벤트(s-event)	이벤트를 발생시킨 프로세스의 종류
프로세스 형식(s-process-type)	이벤트의 종류
전체 사용자 시간(s-user-time)	클라이언트가 프로세스를 사용한 전체 시간 (단위 초)
전체 커널 시간(s-kernel-time)	클라이언트가 사용한 커널 모드 프로세스 전체 시간(단위 초)
전체 페이지 오류(s-page-faults)	메모리 오류가 발생한 메모리 참조 수

<sup>33</sup> Win32 상태 속성은 디버깅시 유용하다. Win32 오류가 어떤 의미를 가지는지 알아보려면 명령줄에서 net helpmsg err를 입력한다. 여기서 err는 알아보려는 오류 번호이다. 예를 들어 err로 5번을 입력하면 “액세스가 거부되었습니다”가 출력된다.

전체 프로세스(s-total-procs)	CGI 또는 프로세스 응용프로그램의 수
활성 프로세스(s-active-procs)	로그가 기록되는 동안 CGI 또는 프로세스 응용프로그램의 수
전체 종료된 프로세스 (s-stopped-proc)	중지된 CGI 또는 프로세스 응용프로그램의 수

④ [일반 속성]탭에서 로그 파일이 저장되는 위치와 새 로그 시간 간격을 지정한다.



IIS 로그 파일의 디폴트 위치는 c:\winnt\system32\logfiles 디렉토리고 로그 파일 이름은 현재 날짜에 기반해서 yymmdd.log로 만들어진다

새 로그 시간 간격은 디폴트로 [매일]이 지정되는데, 사용자가 많아서 로그 파일의 크기가 빠르게 증가하는 경우에는 [파일 크기가 아래와 같아지면]을 선택하여 로그 파일을 일정 크기로 제한할 수 있다.

로깅설정의 변경을 적용하는 시점부터 해당 로깅설정이 바로 적용된다는 점을 주의하도록 한다.

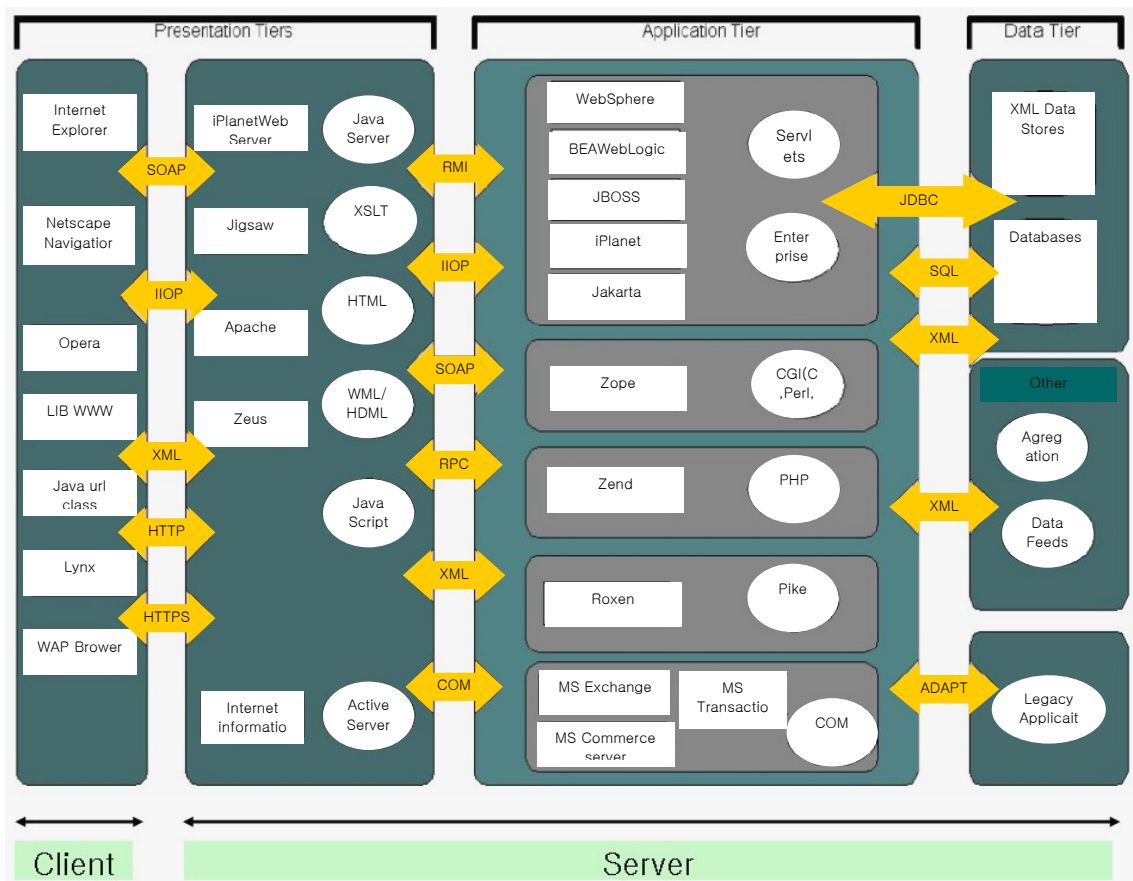
표준 백업 프로시저에 따라 로그 파일들을 백업 받아두었다가 웹 서버에 있는 로그가 침해당했을 때 사용되도록 한다.

이 페이지는 빈 페이지입니다.

## 제 6 장 웹 어플리케이션 보안

### 제 1 절 최신 취약성 분석 및 보안 대책

웹 어플리케이션은 웹 기반 어플리케이션(Web-base application)을 약칭하는 말로써, HTTP를 기반으로 사용자간 또는 다른 시스템간의 정보 교환을 위한 클라이언트/서버 소프트웨어이다. 사용자는 Internet Explorer(IE)나 Netscape 같은 클라이언트 프로그램을 통해 서비스를 요청하게 되고, 서버 프로그램에서는 이를 처리하기 위해서 로컬 디렉토리를 검색하거나, 다른 어플리케이션 서버의 콘텐츠를 참조한다.



(그림 6-1) 일반적인 웹 서비스 구성



초기의 웹 어플리케이션은 CGI(Common Gateway Interface) 기반으로 데이터베이스와 연동하여 웹 서비스를 제공하였다. 현재는 IIS나 아파치 기반의 웹 서버에 Java나 PHP(hypertext preprocessor), ASP(active server pages)와 같이 웹 환경에 최적화된 언어로 구현된 어플리케이션을 사용한다. 특히, 사용자 요구 사항에 따라 다양한 종류의 데이터 타입이나 비즈니스 로직을 다루기 위해서 웹 어플리케이션은 (그림 6-1)과 같이 복잡한 양상을 띠게 된다.

[참고] 웹 어플리케이션은 논리적으로 3개의 기능으로 나뉠 수 있다. 웹 서버는 사용자가 요구한 데이터를 데이터 서버에 질의하여 해당 데이터를 획득한 후, 사용자의 요구에 맞게 전달하는 역할을 한다. 사용자의 웹브라우저에서는 수신한 데이터를 사용자가 인지할 수 있도록 표현하게 되고, 사용자 브라우저는 때때로 필요한 정보를 웹 서버로 보내기도 한다. Presentation-tier는 웹 서버의 아파치, IIS와 클라이언트측의 IE, Netscape와 같은 브라우저, page layout을 만드는 어플리케이션 컴포넌트를 포함한다. 즉, 데이터를 사용자나 서버 시스템에 표현하는 기능을 담당한다. Application-tier는 웹 어플리케이션의 엔진 역할을 한다. 주로 비즈니스 로직 기능을 담당하는데, 사용자 입력값의 처리, 의사결정, 사용자의 presentation-tier 내에서 표현될 데이터를 획득하는 등 역할을 하게 된다. Application-tier 기능은 CGI나 Java(J2EE), ASP.NET 서비스, PHP, ColdFusion과 같은 기술로 구현되고 IBM WebSphere, WebLogic, JBOSS, ZEND 등의 상용 제품이 있다. Data-tier는 application-tier에서 필요한 데이터를 임시 또는 영구적으로 저장하는 기능을 담당하는데, 대부분의 웹 서비스 환경에서는 웹 서버와는 별도로 데이터베이스를 구축하고 운영한다. 근래의 시스템은 다른 시스템과의 호환성을 고려하여 원시 데이터를 XML(extension markup language) 형식으로 저장한다.

본 절에서는 웹 서버를 기반으로 동작하는 클라이언트/서버 소프트웨어의 주요 취약성을 [표 6-1]과 같이 10가지로 분류하여 분석하고, 최근에 문제가 되고 있는 위협사례를 파악하여 이에 대한 대응책을 제시하고자 한다.

[표 6-1] 주요 웹 어플리케이션 취약성과 대책

취 약 성	위 험	대 책
부적절한 파라미터	쿠키값 조작 폼 필드 조작 HTTP헤더 조작 URL 조작 Query string 조작	- 사용자 입력값을 신뢰하지 않는다. - 클라이언트측 유효성 검사를 신뢰 하지 않는다. - 정형화 문제를 고려한다. - 전체 입력 인터페이스에 대한 유효

	유니코드 인코딩 소스코드 조작	성 검사 모듈을 통합하여 설계한다.
취약한 접근제어	권한 상승 퍼미션 악용	<ul style="list-style-type: none"> <li>- 제한/공개영역 사이트로 구분한다.</li> <li>- 강력한 패스워드 정책을 실행한다.</li> <li>- 최소한의 권한으로 시스템을 사용 토크 설정한다.</li> <li>- 권한 분리 정책을 설정한다.</li> </ul>
세션 관리	Brute force 공격 계정 추측하기 세션 가로채기	<ul style="list-style-type: none"> <li>- 중요한 cookie에 대해서는 SSL 통신을 하거나, 세션변수를 암호화하도록 한다.</li> <li>- 세션의 타임아웃을 설정한다.</li> <li>- 세션에 대한 비인가 접근을 막는다.</li> </ul>
Cross-Site Scripting(XSS) 취약점	XSS 공격	<ul style="list-style-type: none"> <li>- 사용자는 접근하려는 URL을 직접 타 이핑 하도록 한다.</li> <li>- 웹 페이지의 특수문자를 필터링 한다.</li> </ul>
버퍼 오버플로우 취약점	버퍼 오버플로우 포맷 스트링	- 어플리케이션 벤더나 보안 사이트의 메일링리스트에 가입하여 최신의 보안패치를 유지한다.
악의적인 커맨드 주입 공격	SQL injection OS command injection	<ul style="list-style-type: none"> <li>- 사용자 입력값에 대한 유효성 검사를 철저히 한다.</li> <li>- 일반 사용자의 시스템 접근을 차단한다.</li> </ul>
부주의한 에러처리	쿠키값 조작 폼 필드 조작 HTTP헤더 조작 Query string 조작	<ul style="list-style-type: none"> <li>- 시스템 정보가 노출되지 않도록 에러처리를 한다.</li> <li>- 구체적인 에러 정보는 서버의 디렉토리에 파일로 남기도록 한다.</li> </ul>
암호의 안전치 못한 사용	알려진 암호취약성 공격 암호 키의 획득	<ul style="list-style-type: none"> <li>- 검증된 암호 알고리즘을 사용한다.</li> <li>- 키 관리에 주의한다.</li> </ul>
원격 관리 취약점	노출된 시스템 정보 악용	<ul style="list-style-type: none"> <li>- 가급적 최소한의 원격관리 기능을 사용한다.</li> <li>- 원격관리 되는 어플리케이션은 root 권한으로 작동시키지 않도록 한다.</li> </ul>
잘못된 설정	관리자 페이지 노출 부적절한 설정파일 접근	<ul style="list-style-type: none"> <li>- 정기적으로 두 개 이상의 다른 취약점 점검도구로 사이트를 점검 한다.</li> <li>- 보안 메일링리스트에 가입하여 최신의 보안 패치를 유지한다.</li> </ul>

## 1. 부적절한 파라미터(Invalidated parameters)

웹 어플리케이션은 응답 메시지를 결정하기 위해서 클라이언트가 보낸 HTTP request 메시지의 정보를 이용한다. 이때, 공격자는 HTTP request 메시지의 정보들 - URL, query string, headers, cookies, form fields, hidden field - 을 알아내고, 값을 조작하여 사이트의 보안 메커니즘을 우회하고자 한다. 만약 웹 어플리케이션이 HTTP request 정보에 대해 강력한 유효성 검사(validation check)를 하지 않는다면 부적절한 파라미터에 대한 취약성은 계속 존재하게 된다.

웹 어플리케이션의 대부분이 입력값의 유효성을 입증하기 위하여 클라이언트 측(client-side validation mechanisms)<sup>34</sup>에서 검사하게 된다. 클라이언트 측에서의 인증 방식은 성능이나 사용자 측면에서 좋은 방법이지만, 쉽게 우회할 수 있어서 보안상으로는 큰 취약성을 가지고 있다. 따라서 서버 측에서도 파라미터 조작 공격(parameter manipulation attacks)에 대한 방어가 필요하다.

대부분의 웹 환경에서는 다양한 인코딩 방식을 지원하는데, 예를 들어 c:\temp\test.txt는 다음 스트링과 같이 표현될 수 있다.

```
>test.txt
>c:\temp\subdir\..\test.txt
>c:\temp\ test.txt
>..\test.txt
>c%3A%5Ctemp%5Csubdir%5C%2E%5Ctest.txt
(:은 %3A, \는 %5C, . 는 %2E의 16진수로 표현함)
```

이처럼 다양하게 표현된 입력값은 값들의 유효성을 검사하기 이전에 간단한 형태로 단일화 되어 처리되는 것이 일반적이다. 이것을 정형화(canonicalization)라고

---

<sup>34</sup> 클라이언트측 유효성 검사란, 웹 어플리케이션의 form에 대해 사용자가 입력 필드를 무시하지 않도록 하거나 또는 입력한 값에 대해 속성, 범위, 패턴 등이 유효한지에 대해 통제하고 오류사항에 대해서는 적절히 대처하도록 클라이언트측 스크립트에서 검사하게 된다. 서버의 성능향상을 위하여 대부분의 사이트에서 유효성 검사를 클라이언트측에 의존하고 서버에서는 가장 문제가 될 일부의 입력에 대해서만 검사하게 된다. 예를 들어, 많은 사이트에서 자바스크립트로 보안체크를 하고 있다. 자바스크립트는 클라이언트에서 실행되는 언어이므로 변조가 가능하고 우회할 수도 있다. 따라서, 서버측 스크립트 언어에서 다시 보안검사를 하도록 해야한다.

한다. 이런 경우, 부적절한 값들은 유효성 검사를 피할 수 있는 기회가 되기 때문에 어플리케이션 설계 단계에서도 주의해야 한다.

### (1) 취약성 판단하기

- 웹 어플리케이션의 소스 코드를 자동화된 소스 검사 도구로 점검하거나, 수동적으로 알려진 취약성에 대해서 점검한다. 이때 주로, 사용자 입력값을 처리하는 모듈이나 HTTP request 메시지 정보를 호출하는 부분, 변수를 다루는 부분, 사이트를 업데이트 하는 부분 등의 소스코드를 정밀하게 검사한다. 만약 호출된 변수가 그 값의 유효성을 검사하지 않고 어플리케이션에서 사용된다면 취약하다고 할 수 있다.

예> Perl의 경우 "taint"(-t) 옵션으로 찾을 수 있다.  
J2EE에서는 HttpServletRequest class를 조사한다.

- 모의 침투방식으로 파라미터 조작에 대한 취약성을 점검할 수 있다.
- 웹 취약점 점검 도구를 활용해서 사이트의 취약성을 점검한다.

### (2) 보호대책

- 개발자가 웹 어플리케이션을 개발할 당시에 점검하는 방법이 가장 효과적이다. 특히, 보안에 위반되는 사항을 체크하는 것보다 수용할 수 있는 조건을 정해서 해당 조건을 만족하는 파라미터만을 사용하는 것이 중요하다.

예> 게시판에 올릴 수 없는 파일을 점검하는 것보다 올릴 수 있는 파일 조건을 설정하고 조건에 맞지 않은 파일들은 업로드 하지 못하도록 설계하는 것이 바람직하다.

- 파라미터 조작을 방지하기 위한 최상의 방법은 파라미터가 사용되기 전에 다음 사항들을 점검하는 것이다.
  - data type
  - character set
  - 최대/최소 길이
  - 널(Null)문자 허용 여부

- 추가적인 파라미터의 요구사항
- 중복 정보
- 숫자의 한계 범위
- 지정된 범위의 값(enumeration)
- 특정한 패턴(regular expressions)

- 파일명, 경로명, URL처럼 이름값을 입력 받을 경우, 정형화 문제에 대해 주의한다.

예> 웹페이지에 링크되는 파일의 경로명은 사용자에게 의해서 변경될 수 없도록 절대경로를 쓰도록 한다.

입력되는 파일 이름 자체가 웹 어플리케이션에서 적절한지 판단한다.

- 파일을 다운 받을 수 있는 디렉토리를 특정 디렉토리로 한정하여 다른 디렉토리에서는 파일을 다운 받을 수 없도록 어플리케이션을 수정한다. 그리고, 다운로드 경로는 공격자에 의해 수정이 불가능하도록 절대경로를 사용하도록 한다.

### (3) 위협 사례

#### ■ Cookie 조작(Cookie manipulation)

cookie는 클라이언트와 서버 사이의 연결을 유지하는 방법 중 한가지로서 일정 시간 메모리에 유지되는 정보이기 때문에 사용자에게 의해 수정이 가능하다. cookie 조작은 웹 사이트에 대한 접근 권한을 획득하기 위해 주로 이루어진다.

이 공격에 대한 보호대책은 다음과 같다.

- ▶ SSL을 통한 통신은 cookie 조작의 공격을 막을 수 있다.
- ▶ cookie를 HMAC(Hash message authentication code) 등의 알고리즘으로 암호화한다.
- ▶ Cookie 보다는 세션 통신이 보다 안전한 방법이다. 또는 cookie와 세션을 함께 검사하도록 설계한다.
- ▶ 사용자가 자신의 호스트에 저장되어 있는 cookie 정보를 수정하여 공격할 수도 있기 때문에, 서버측에서도 cookie 정보의 조작 여부를 확인해야 한다.

## ■ Form field 조작

HTML form field<sup>35</sup>의 값들은 HTTP POST 방식으로 서버에 텍스트로 전달된다. 여기에는 hidden form field가 포함되어 있는데, 어떤 타입이든지 form field 값들은 쉽게 수정할 수 있고, 클라이언트 측에서 수행하는 유효성 검사를 우회할 수 있다. 결과적으로, form field 값들을 기초로 서버의 보안 의사결정(security decision)을 하는 경우는 form field 조작 공격에 취약하다고 할 수 있다.

이 공격에 대한 보호대책은 다음과 같다.

- ▶ 서버와 클라이언트 사이의 인증을 hidden form field 대신에 보다 안전한 세션변수를 사용한다.
- ▶ 웹 페이지를 수정하는 단계에서는 반드시 사용자 인증을 다시 한번 하도록 한다.

## ■ HTTP 헤더 조작

HTTP 헤더는 클라이언트와 서버 사이에서 오고 가는 정보들을 갖고 있다. 클라이언트는 서비스 요청에 대해서, 서버는 서비스 응답에 대한 헤더를 만들게 된다. 만약 웹 어플리케이션의 보안의사 결정이 HTTP request 메시지의 헤더 정보를 기초로 한다면 어플리케이션은 HTTP 헤더 조작 공격에 취약하다고 할 수 있다.

이 공격에 대한 보호대책은 다음과 같다.

- ▶ HTTP 헤더를 기본으로 보안의사 결정 로직을 설계하지 않는다. 예를 들어, 클라이언트의 출처를 결정하기 위해서 HTTP Referer를 신뢰하지 않도록 한다. 왜냐하면, HTTP Referer은 쉽게 조작 될 수 있기 때문이다.

<sup>35</sup> HTML form의 hidden form field는 클라이언트와 서버 사이의 연결을 유지하는 방법 중 한가지로서, 사용자 정보를 hidden field에 넣어 지정된 페이지로 넘길 때 사용한다. 정보를 간단히 넘길 때 편리한 방법이지만, 보낼 수 있는 정보의 제한성과 클라이언트 브라우저의 소스 보기로 쉽게 접근할 수 있다는 점에서 보안이 필요한 경우에는 사용하지 않는다.

## ■ URL 조작

웹 사이트에서 다운로드 페이지나 업로드 페이지에서 URL을 일부 조작하여 패스워드 파일을 비롯한 주요 파일을 획득할 수 있다.

이런 공격이 가능한 이유는 cgi, php, php3, jsp와 같은 서버측 스크립트 파일의 기능을 이용해서 URL 구문을 “..” 나 “%2e”, “%2f” 등과 같은 특수 문자로 적절히 조작하여 발생하게 된다.

이런 특수문자들을 정확히 필터링 하지 않았기 때문에 웹루트 디렉토리를 벗어나서 임의의 위치에 있는 파일을 열람하거나 다운 받는 것이 가능해진다.

이 공격에 대한 보호대책은 다음과 같다.

- ▶ 파일을 다운 받을 수 있는 디렉토리를 특정 디렉토리로 한정한다.
- ▶ 사용자가 요청한 URL 구문(Requested URL)에 대해서 완전하게 디코딩 한 후, 보안 검사를 하도록 한다.
- ▶ 특수문자에 대해 필터링 하도록 한다.

## ■ Query string 조작

브라우저의 URL 주소창에 query string이 그대로 보이기 때문에, 사용자가 GET 방식으로 query string 값을 조작하여 서버로 보내는 것은 쉽다. 따라서, 만약 웹 어플리케이션이 query string을 그대로 사용하여 보안의사 결정을 하거나, 결제액 같은 민감한 데이터를 URL 주소창에 그대로 보여주는 경우라면 query string 조작 공격에 취약하다고 할 수 있다.

이 공격에 대한 보호대책은 다음과 같다.

- ▶ 민감한 데이터나 서버의 보안로직에 영향을 미치는 query string을 GET 방식으로 URL 구문에 포함시키지 않도록 한다.
- ▶ form에 대한 submit 타입을 입력할 경우, GET 방식 대신 POST 방식으로 전송한다.

## ■ 유니코드 공격

웹 환경에서 정형화 문제는 URL 인코딩에서 IP 주소 변환까지 매우 다양하다. 그 중에서도 한때 이슈가 되었던 유니코드 공격은 대표적인 정형화 취약성을 이용한 공격법으로 알려져 있다.

유니코드 취약점은 IIS4.0과 5.0버전의 웹 서버와 Windows2000 SP1과 SP2에 존재하는 것으로 알려져 있다.

이 공격에 대한 보호대책은 다음과 같다.

- ▶ 웹 서버를 패치 하도록 한다.

## ■ 소스코드 조작

웹 사이트는 대부분이 불특정 다수에게 공개되는 특성이 있다. 서버측 스크립트 언어 - PHP, ASP(ASP.NET), JSP 등 - 는 클라이언트 브라우저에서 소스코드를 직접적으로 볼 수는 없지만, 브라우저의 HTML 소스코드 보기 기능을 통해서 클라이언트 웹 사이트가 어떻게 작동하는지에 대한 정보는 노출 될 수 있다. 이런 경우는 소스코드 자체의 취약점 보다는 어플리케이션의 동작 과정의 취약성을 통해 관리자 권한을 획득할 수도 있다.

어플리케이션의 구조적인 문제를 발견하는 일은 그리 쉽지 않으며, 진단도구 역시 전무하기 때문에 설계 당시에 보안성 검토를 하는 것이 가장 중요하다고 볼 수 있다.

- ▶ 클라이언트측 브라우저에서 소스코드 보기 기능을 제한하도록 설정한다. 모든 소스보기 기능을 제한할 필요는 없으며, 어플리케이션의 성격에 따라 판단하도록 한다.
- ▶ 소스코드 조작은 클라이언트 페이지에서 발생하기 때문에, 조작된 페이지의 업로드를 막기 위해 사용자 인증을 다시 하도록 설계한다.



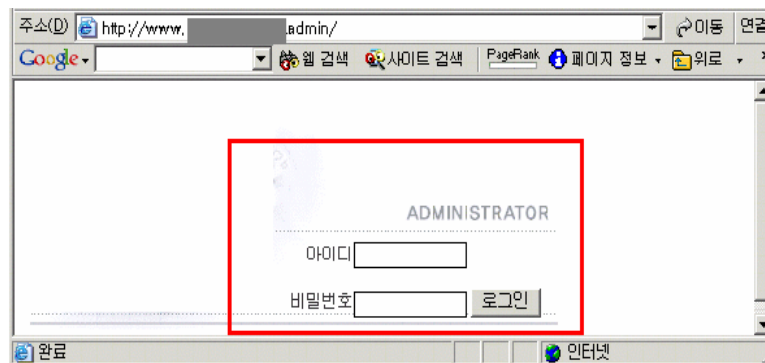
## 2. 취약한 접근제어(Broken access control)

접근제어 또는 권한(authorization)은 인증(authentication)된 사용자에게 허가된 적절한 자원이나 서비스를 제공하는 과정을 의미한다. 즉, 권한은 인증된 사용자들에게 무엇을 허락할 것인가에 대한 문제이다. 웹 어플리케이션의 접근제어 모델은 그 사이트가 제공하는 콘텐츠와 기능에 매우 밀접하게 연관된다. 또한 사용자들은 소속 그룹에 따라 다른 권한과 권리를 가지게 된다.

대부분의 개발자들은 믿을 수 있는 접근제어의 구현을 어렵지 않게 여기는 경향이 있는데, 정교하게 구현되지 못한 접근제어는 악의적인 사용자들에게 쉽게 노출되고 공격의 수단이 된다.

### (1) 취약성 판단하기

- 사용자나 그룹에 따른 접근제어 정책이 문서화 되어 있는지 검토한다.
- 접근제어 정책에 따라 구현된 소스코드를 자세하게 검토해야 한다.
- 어플리케이션의 환경설정 페이지 또는 관리자 페이지가 노출되어 있는지 검토한다. 그리고 디폴트로 설치된 URL이 변경이 안되어 누구에게나 접근이 가능하면 취약하다고 할 수 있다.
- 모의침투 테스트를 통해 취약성 정도를 판단 할 수 있다.
  - URL에 임의의 디렉토리 이름을 추측하여(guessing) 입력했을 때, 디렉토리가 목록에 나타나면 접근제어가 취약하다고 할 수 있다.



(그림 6-2) 관리자 페이지 노출

## (2) 보호대책

- 웹 어플리케이션 접근제어에 대한 정보보호 정책을 문서화 한다. 문서 내용에는 사용자를 분류하고 분류된 사용자 그룹에게 제공될 수 있는 서비스와 자원을 명시한다.
- 웹 어플리케이션의 접근제어 모델 설계 및 운영시 다음 사항을 고려한다.
  - 접근제어의 기준을 쉽게 노출될 수 있는 아이디(ID)에 기반을 두지 않도록 설계한다.
  - 신상정보변경 페이지 같은 특정한 URL에 접근할 경우 반드시 접근권한을 재점검 하도록 한다.
  - 웹 서버에 저장되어 있는 설정 파일이나 데이터 파일의 파일속성을 최소한의 권한으로 설정한다.
  - 중요한 정보들은 캐쉬(caching)되지 않도록 설계한다.
- 관리자 페이지의 경우, 관리자 호스트 IP만 접근하도록 설정하고, 관리자 페이지의 URL은 추측하기 어려운 이름으로 정한다.
- 관리자 페이지의 접속 포트 역시 제한하도록 한다. 이때 내외부 인터페이스의 접근 포트를 제한하도록 한다.
- 사용자의 경우, 사이트의 URL창이나 로그인 폼, 개인 신상정보 등록 폼 등의 자동완성 기능을 비활성화 하도록 브라우저를 설정한다.

### 3. 세션 관리

클라이언트와 서버 사이의 관계를 유지하기 위한 개념<sup>36</sup> 중 한가지가 세션이다. HTTP 프로토콜은 상태를 유지하는 프로토콜이 아니기(stateless) 때문에 세션 매커니즘은 웹 어플리케이션에서 지원하게끔 설계해야한다.

클라이언트가 서버에 최초로 접속했을 때 세션변수를 부여 받으면 클라이언트와 서버 양쪽에 저장되어 연속된 HTTP request/response 관계를 유지하게 된다. 즉, cookie처럼 클라이언트 정보가 사용자측에만 저장되는 것과는 다른 개념이다. 서버와 클라이언트 양쪽에 같은 정보가 저장되기 때문에 Cookie에 비해 보안 면에서 보다 안전한 방법이라고 할 수 있다.

만약 공격자가 세션을 가로채어 세션 ID를 조작할 수 있다면 정상적인 사용자로 위장하여 서비스를 받을 수 있다.

#### (1) 취약성 판단하기

- 세션정보가 URL 구문에 명시되어 GET 방식으로 전달되는지 점검한다.
- 클라이언트 소스보기 기능으로 hidden form field의 세션정보가 노출되는지 점검한다.
- 네트워크 스니핑 도구를 사용해서 서버와 클라이언트 사이의 세션 정보가 스니핑되는지 점검한다.

---

<sup>36</sup> 웹 서비스에서 서버와 클라이언트의 상태를 유지하기 위한 방법은 hidden form field, URL rewriting, cookie, 세션 등이 있다. hidden form field는 POST 방식으로 데이터를 넘길 때 hidden input 양식에 클라이언트를 구분할 수 있는 구분자를 포함시키는 방법이다. URL rewriting은 클라이언트를 구분하기 위해 하나의 페이지에서 다른 페이지로 이동할 경우 URL의 뒷부분에 정보를 가지고 다니는 것을 말한다. 이 방식은 GET 방식에서 사용된다. Cookie는 서버가 클라이언트에 전송하여 저장하는 텍스트 파일로, 간단한 텍스트 파일에 클라이언트를 구분할 정보를 담아 전송하게 된다.

## (2) 보호대책

- 로그인 과정부터 모든 세션을 SSL(Secure socket layer)로 암호화 한다.
- SSL이 적용되기 어려울 경우에는 세션변수가 브라우저에 의해 캐싱되어 URL이나 referer header에 포함되지 않도록 한다.
- 인증 페이지는 누군가 로그인 페이지를 백업하기 위해서 사용자 브라우저의 “뒤로” 버튼을 누를 경우에 대비하여 HTTP 헤더의 “no cache tag”와 관련된 모든 것을 활성화 해야 한다.

## 4. Cross-Site Scripting(XSS) 취약점

Cross-site scripting은 XSS<sup>37</sup>라고도 불리는데, 웹 어플리케이션이 사용자 입력으로 받은 악성 코드를 필터링하지 않고 그대로 동적으로 생성된 웹페이지에 포함하여 사용자에게 재전송하는 것이다. 자바스크립트처럼 클라이언트측에서 실행되는 언어<sup>38</sup>로 작성된 코드를 사용자 입력으로 주게 되면, 이 코드가 그대로 클라이언트 컴퓨터에게 전달되어 브라우저 상에서 실행이 된다.

XSS는 크게 두가지 형태가 있다. 공격 대상 서버에 악의적인 코드를 영구적으로 주입하는 방식과 사용자가 주입된 공격 코드를 클릭함으로써 사용자 브라우저에서 공격 코드가 활성화 되는 방식이 있다.

XSS 공격의 가장 큰 위험은 사용자의 세션과 쿠키 정보, 사용자 파일의 노출과 트로이안 목마 프로그램의 설치, 다른 웹 페이지로의 이동, 콘텐츠의 수정 등으로 볼 수 있다.

### (1) 취약성 판단하기

- XSS 취약점이 존재하는 사이트의 특징은 사용자의 입력을 검증하지 않고 그대로 동적으로 생성되는 웹페이지에 포함해서 다시 보여준다는 점이다.
- XSS 취약점을 의심할 수 있는 웹 어플리케이션은 주로 다음과 같다.
  - 입력한 검색어를 다시 보여주는 검색엔진
  - 입력한 스트링을 함께 보여주는 에러 페이지
  - 입력한 값을 사용자에게 다시 돌려주는 Form
  - 사용자에게 메시지 포스팅이 허용된 웹보드

---

<sup>37</sup> Cross-site scripting의 명명은 CERT 보안 권고문 "CA-2000-02 Malicious HTML tags embedded in client web requests"에서 시작했다.

<sup>38</sup> 클라이언트 컴퓨터에서 실행되는 Client-Side Script 언어에는 JavaScript, VBScript, Flash, ActiveX, XML/XSL, DHTML 등이 있다.

## (2) 보호대책

- 웹 어플리케이션의 소스코드에 대한 유효성 검사(validation check)를 수행 한다. HTTP headers, cookies, query string, form field, hidden field 에 대해 입력을 허락할 수 있는 필터링 조건을 설계하는 것이 필요하다.
- 웹 전용 취약점 점검도구를 사용하여 웹페이지를 점검한다.
- 사용자 브라우저의 스크립트 활성화 기능을 비활성화 시킨다.
- 스크립트 코드의 다음 문자를 필터링하여 검출하고, 이 문자들을 다음과 같이 변경하도록 한다.

대상문자	<	>	(	)	#	&
변경값	&lt;	&gt;	&#40	&#41	&#35	&#38

- 웹사이트 방문시 메일이나 웹문서에 포함된 링크를 클릭하지 말고, 직접 브라우저 주소창에 URL을 입력한다.
- XSS 공격은 대체로 메일이나 팝업창에 Javascript가 포함되어 사용자에게 전달 된다. 특히, 이메일에 포함되는 ActiveX(OLE) 객체, VBscript, Shockwave, Flash 등과 같은 임베디드 액티브 콘텐츠는 방화벽이나 IDS를 통과하기 때문에 사용자는 신뢰할 수 있는 발신자의 이메일에 포함된 링크나 이미지, Flash라도 함부로 클릭하지 않도록 주의할 필요가 있다.
- 브라우저의 보안 패치를 최신으로 유지한다.

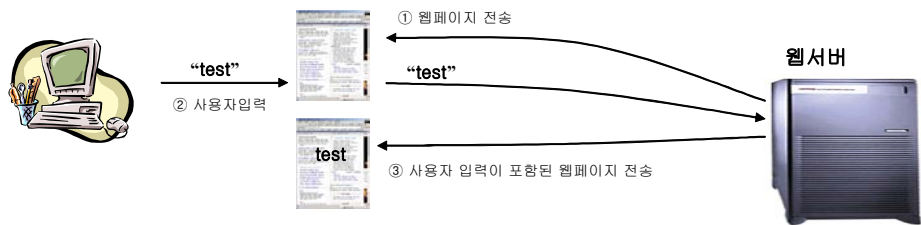
## (3) 위협 사례

XSS 공격은 시스템 자체 보다는 사용자를 대상으로 한 공격이다. 공격 시나리오는 다음과 같다.

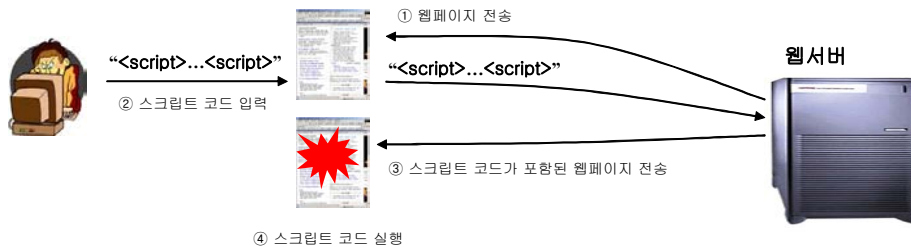
공격자는 XSS취약점이 존재하는 웹페이지의 사용자 입력으로 “test” 와 같이 정상적인 스트링을 입력하는 것이 아니라 <script>로 시작하는 악성 스크립트 코드를

입력한다. 웹 서버는 공격자가 입력한 악성 스크립트 코드가 포함된 웹페이지를 생성해서 클라이언트에게 되돌려준다. 이 웹페이지에 포함된 스크립트 코드는 클라이언트측 브라우저상에서 실행된다.

공격자는 웹메일이나 게시판 등을 이용하여 리턴되는 웹페이지를 공격 목표가 되는 일반사용자에게 전달한다. 공격자는 웹메일에 악성 스크립트 코드를 포함하여 전달하거나, 게시판에 악성 스크립트 코드가 포함된 글을 포스팅한 후 그 글을 읽도록 한다. 일반 사용자가 공격자로부터 수신한 웹메일을 여는 순간 혹은 공격자가 포스팅한 게시물을 읽는 순간, 해당 웹페이지에 포함된 스크립트 코드가 실행된다.



<정상 스트링 입력>



<악성 스크립트 입력>

(그림 6-3) XSS 취약점 예시

이러한 방법 외에도 악성 스크립트를 URL에 포함시켜서 전달하는 방법도 있다. 공격자는 URL의 변수값으로 악성 스크립트를 넣은 링크를 공격 대상자에게 전달하여 그 링크를 클릭하도록 한다. 공격자는 이 링크를 공격 대상자에게 친숙한 내용의 HTML 메일에 포함시켜 전달하는데, 메일은 공격 대상자가 반드시 그 링크를 클릭하도록 내용을 구성한다. 공격 대상자가 그 링크를 클릭함과 동시에 악성 스크립트가 포함된 웹페이지를 수신하게 된다.

## 5. 버퍼 오버플로우 취약점

버퍼는 일반적으로 배열 형식의 메모리 블록이다. 배열 크기가 확인되지 않으면 할당된 버퍼 외부에 작성할 수 있게 되는데 이러한 동작이 버퍼보다 상위의 메모리 주소에서 발생하는 경우가 버퍼 오버플로우이다. 공격자가 교묘한 입력값을 웹 어플리케이션에 주입하면 어플리케이션은 할당할 수 있는 메모리 크기를 초과하여 임의의 메모리 주소로 분기하는데, 이때 분기된 주소가 공격자가 실행하려고 하는 데몬이나 유해한 프로그램의 시작주소가 될 경우 공격자는 웹 서버의 권한을 획득할 수 있다.

또한 버퍼 오버플로우 취약점은 프로세스의 자원을 고갈시켜 시스템으로 하여금 서비스를 제공하지 못하게 하는 서비스 거부공격(Denial of service)으로 확대 될 수 있다.

웹 어플리케이션에 존재하는 버퍼 오버플로우 취약점을 발견하는 것은 쉬운 일이 아니다. 특히, JSP와 같은 스크립트 언어에서의 버퍼 오버플로우 취약점을 찾는 것은 상당히 어렵다고 알려져 있다. 또한 취약점이 발견됐다 하더라도 그 취약점을 이용하여 익스플로잇 하는 것도 어렵다. 하지만 익스플로잇이 성공했을 경우 매우 치명적인 결과를 가져올 수 있으므로 이 취약점에 대해 항상 주의 해야 할 것이다.

버퍼 오버플로우 공격과 유사한 포맷 스트링 공격이 알려져 있다.

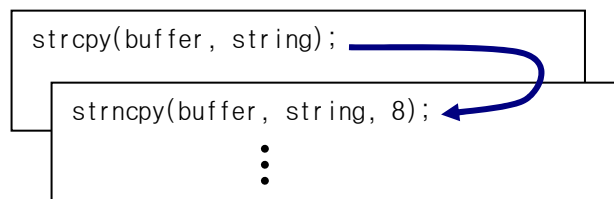
### (1) 취약성 판단하기

- PHP, ASP(ASP.NET), JSP 등과 같은 스크립트 언어로 작성된 사이트 보다는 C나 C++과 같은 언어로 작성된 사이트의 경우, 버퍼 오버플로우 취약성이 존재할 확률이 높다.
- 기존의 알려진 버퍼 오버플로우 취약성과 관련한 보안 권고문을 참고로, 현재 웹 서버와 웹 어플리케이션의 운영체제와 소스코드의 버전을 확인한다.

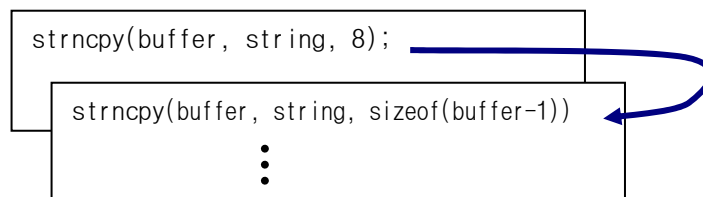


## (2) 보호대책

- IDS에서 버퍼 오버플로우 공격의 징후를 판단할 수 있도록 IDS 정책을 설정한다.
- 웹 서버에 대한 최신 취약점 정보에 항상 주의를 기울이고 발표된 보안 패치를 적용한다.
- 웹 서버와 어플리케이션이 제한된 권한을 가진 계정으로 실행되도록 한다.
- 직접 개발한 어플리케이션에 대해서는 모든 외부 입력에 대해 적절한 한계체크를 수행하고, 버퍼 오버플로우 취약점이 존재하는 함수의 사용을 금한다.
  - 가급적이면 프로그래밍 도구로 C가 아닌 Perl, Python, Java와 같이 자동화된 바운드 체크 기능을 제공하는 언어를 사용한다. 그러나 거의 모든 OS가 C로 작성되어 있어서 이것이 불가능하거나 실용적이지 못한 경우가 있을 수 있다. 시스템 내부의 하드웨어 접근이 필요한 경우가 특히 그러하다.
  - 취약점이 존재하는 라이브러리 함수의 사용을 금한다. 예를 들어 표준 C 라이브러리에서 `get()`, `strcpy()`, `strcmp()` 함수는 인자에 대해 한계 체크를 수행하지 않는다. 이러한 함수는 다음 예제와 같이 취약점이 없는 함수로 대체하도록 한다.



- 보안을 최우선적으로 고려하여 소프트웨어를 설계한다. 이를테면 위 예제를 다음과 같은 방법으로 버퍼를 좀 더 안전하게 만들 수 있다.



- 소스 코드 스캐닝 도구를 사용한다. 소스 코드 점검 도구는 소스 코드를 분석하여 버퍼 오버플로우를 일으킬 수 있는 코드 부분을 찾아내 준다. 소스 코드 스캐닝 도구로 PurifyPlus 등이 있다<sup>39</sup>.
- 안전한 라이브러리 모듈을 사용한다. C++ 표준 템플릿 라이브러리는 내부적으로 한계 체크를 수행하는 String 클래스를 제공한다. 이와 같이 취약한 표준 문자열 처리 함수보다는 좀 더 안전한 스트림 처리 함수를 제공하는 라이브러리 모듈을 찾아서 이용하도록 한다.
- 스택 실행이 불가능하도록 만든다. 스택 실행이 불가능하도록 하기 위해서는 OS 커널에 대한 패치가 필요하다. 스택 실행을 불가능하게 만드는 패치가 일부 UNIX 버전에 대해서 존재한다. 대부분의 버퍼 오버플로우 공격이 실행 가능한 스택을 이용하기 때문에 이것은 효과적인 예방 방법이 될 것이다. 리눅스 커널에 대한 패치도 존재한다<sup>40</sup>.
- 시스템에 대한 점검을 수행한다. 어플리케이션이 실행될 시스템에 무엇이 존재하고 누가 그것을 실행시킬 권한을 갖는지 파악하도록 한다. SUID가 root로 설정되었거나 root 소유의 world writable 파일들과 디렉토리들은 공격의 대상이 될 수 있다. 이러한 파일과 디렉토리들을 찾아서 버퍼 오버플로우 취약점에 대한 테스트를 수행하도록 한다.

---

<sup>39</sup> memory leak를 발견하도록 개발된 툴로서 버퍼 오버플로우를 일으킬 수 있는 unchecked buffer나 코딩 에러를 찾아낼 수 있다([www.rational.com](http://www.rational.com)).

<sup>40</sup> Openwall Project, [www.openwall.com](http://www.openwall.com)

## 6. 악의적인 명령어 주입 공격

명령어 주입 공격은 웹 어플리케이션에 악의적인 코드를 입력하여 시스템을 무력화 시키는 공격 방법이다. 웹 어플리케이션의 구조에 따라 때때로 운영체제에 접근하기도 하는데, 이때 악의적인 값이 입력되면 웹 어플리케이션 뿐만 아니라 운영체제의 권한까지도 공격 당하게 된다.

### (1) 취약성 판단하기

- 시스템함수 호출, 셸 명령어, SQL 처리문 등 주로 외부 명령어를 입력 받아 실행하는 웹 어플리케이션이 취약하다.
- system, exec, fork, Runtime.exec, SQL 쿼리 등 외부 자원을 호출하는 부분의 소스코드를 검토한다.
- SQL injection 과 같은 모의 해킹을 통해서 시스템의 취약성을 판단할 수 있다.

### (2) 보호대책

- 웹 어플리케이션에는 셸 명령어나 시스템 함수를 호출하는 코드 또는 라이브러리를 가능한 피하는 것이 좋다.
- 웹 페이지에 만약 명령어 입력이 필요한 부분이라면 보안 여부가 보증되어야 한다. 테스트 과정에서 입력에 대한 산출물과 리턴 파라미터, 에러메세지 등을 꼼꼼히 점검해야 한다.
- 웹 어플리케이션에서 명령어를 실행할 수 있는 권한을 제한하도록 한다.
- 가급적 시스템 명령어를 입력받는 인터페이스는 웹 페이지에 표시되지 않는 것이 좋다.
- 웹 어플리케이션을 최신 보안 패치하도록 한다.

### (3) 위협 사례

#### ■ SQL injection

최근 웹 프로그래밍은 자료의 효율적인 저장 및 검색을 위해 DBMS를 필수적으로 사용하고 있다. 주로 PHP, JSP, ASP 등의 스크립트 언어를 이용하여 DBMS와 연동하는데, 이러한 웹 어플리케이션에서 클라이언트의 잘못된 입력값을 검증하지 않아 비정상적인 SQL 쿼리가 발생할 수 있다.

이러한 비정상적인 쿼리는 사용자 인증을 우회하거나 데이터베이스에 저장된 데이터를 노출시킬 수 있다. 공격자는 SQL injection 취약점을 이용하여 아이디와 암호를 몰라도 웹 기반 인증을 통과할 수 있고, 데이터베이스에 저장된 데이터를 열람해 볼 수 있다. 또한 SQL 명령어를 입력하고 실행시킬 수도 있다.

이러한 SQL injection 공격은 방어하기가 비교적 쉬움에도 불구하고 현재 인터넷에 연결된 많은 웹사이트들이 이 공격에 취약하다.

SQL injection 공격 방법 중 가장 간단한 방법은 로그인 아이디와 패스워드를 조작하여 웹 어플리케이션의 인증 기능을 우회하는 것이다. 공격자는 아이디와 패스워드 입력에 조작된 SQL문을 입력하여 인증을 통과하는데, 이 공격 방법을 통해 특정 사용자의 권한으로 로그인할 수도 있고, 임의의 사용자 특히 관리자의 권한으로 로그인 할 수도 있다.

웹 어플리케이션 코드를 보면 사용자가 입력한 데이터를 SQL문에 포함하여 처리하는 부분이 있는데, SQL injection 공격은 이 부분을 이용하는 것이다. 공격자는 웹 어플리케이션 코드내의 SQL문이 자신이 원하는 악의적인 SQL문으로 변경될 수 있도록, 조작한 스트링을 이곳에 입력으로 주어서 사용자 인증을 우회하거나 데이터베이스에 저장된 모든 테이블들의 구조와 내용을 열람할 수 있다.

그리고 SQL 서버에서 제공하는 stored procedure를 이용하면 SQL 서버에서 임의의 명령어를 실행할 수도 있는데, 이것이 가능할 경우 데이터베이스의 삭제는 물론 서버 OS에 대한 접근 및 조작도 가능하다.

이 공격에 대한 보호대책은 다음과 같다.

- ▶ SQL injection 공격을 방지하기 위해 사용자 입력을 필터링 한다.
- ▶ SQL 서버의 에러 메시지를 사용자에게 보여주지 않도록 설정한다.
- ▶ 웹 어플리케이션이 사용하는 데이터베이스 사용자의 권한을 제한한다.
- ▶ 데이터베이스 서버에 대한 보안 설정을 수행한다.

SQL Injection에 대한 가장 훌륭한 해결책은 포괄적인 입력 검증을 수행하는 것이다. 모든 스크립트에 존재하는 모든 파라미터들을 점검하여 사용자의 입력값이 SQL injection을 발생시키지 않도록 수정한다.

웹 어플리케이션 코드를 수정하여 사용자 입력이 SQL 문장으로 사용되지 않도록 한다. 모든 사용자 입력의 앞뒤에 따옴표를 붙이도록 하고, 사용자 입력에 존재하는 따옴표를 제거하도록 한다. 따옴표를 제거하는 것보다 더 확실한 방법은 사용자 입력에 따옴표가 존재하는 경우 이를 에러처리 하도록 하는 것이다. 만약 따옴표가 제거된 사용자 입력을 계속 처리하도록 허용한다면 이를 우회한 공격이 가능할 수 있다.

따옴표에 대한 검증을 수행한 다음에는 사용자 입력으로 사용이 불가능한 스트링을 제거하도록 하고, 사용 가능한 문자집합에 포함되지 않은 모든 문자들을 제거하도록 한다. 이 경우에도 마찬가지로 허용되지 않는 문자열이나 허용되지 않는 문자가 포함된 경우를 에러로 처리하는 것이 더욱 안전하다.

현재 알려져 있는 공격 패턴을 기반으로 해서 사용자 입력으로 사용이 불가능한 스트링을 결정한다. 또한 새로운 공격 기술에 대비해서 사용자 입력으로 사용이 가능한 최소의 문자집합을 결정한다. 사용이 허용된 문자들의 조합으로 공격 스트링을 만들 수 있기 때문에 이 두 가지 방법을 동시에 사용해야 한다. 최소 사용가능 문자집합에는 가급적 숫자만을 포함하도록 하고, 그것이 어렵다면 숫자와 문자만을 포함하도록 한다. 만약 사용자 입력으로 기호문자(symbol)나 구두문자(punctuation)같은 문자들을 사용해야 한다면, 꼭 필요한 최소의 문자만을 포함하도록 하고 허용된 문자를 HTML 문자로 변환해서 처리하도록 한다.

공격자는 리턴되는 에러 메시지에 대한 역공학 분석을 통하여 공격에 성공할 수 있는 SQL Injection 스트링을 알아낸다. 따라서 SQL 서버의 에러 메시지를 외부에 제공하지 않도록 한다. 많은 경우 개발과정에서 디버깅을 목적으로 에러가 리턴되도록 허용했다가 개발이 끝난 후 그 기능을 제거하는 것을 잊어버리는 경우가 많다. 에러가 발생한 경우 에러 발생사실을 보여주기 보다 메인 페이지로 돌아가도록 하는 것이 좋다. 500 Internal Server Error page 자체가 그 서버에 대해 SQL Injection 공격이 가능하다는 것을 의미할 수 있다.

웹 어플리케이션이 사용하는 데이터베이스 유저의 권한을 제한한다. 가능하면 일반 유저 권한으로는 모든 system stored procedures에 접근하지 못하도록 한다. 이렇게 하여 웹 어플리케이션의 SQL Injection 취약점을 이용하여 데이터베이스 전체에 대한 제어권을 얻거나 데이터베이스를 운용중인 서버에 대한 접근이 불가능하도록 한다. 이외에도 데이터베이스 보안에 대한 문서를 참조하여 데이터 베이스 서버의 보안 수준을 제고하도록 한다.

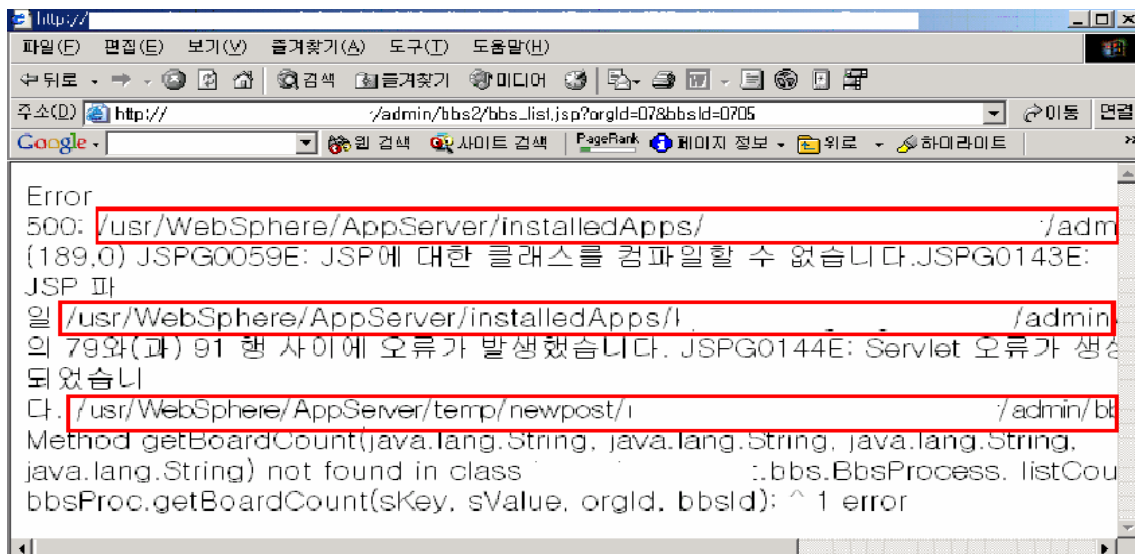
## 7. 부적절한 에러처리

부적절한 에러처리는 웹사이트의 보안문제를 임의의 사용자에게 드러낼 수 있다. 예를 들어, 특정 파일 접근을 시도할 때, 파일이 존재하지 않을 경우 “file not found” 라는 오류메시지가 표시되지만, 파일이 존재하고 파일접근권한이 없는 경우 “access denied” 라는 오류메시지가 표시된다. 공격자는 오류메시지만으로 파일의 존재유무와 속성에 관한 정보를 획득할 수 있는 것이다.

좋은 에러처리 방법은 간단한 에러메시지를 표시하도록 하고 그 발생원인에 대해서는 파일형식의 로그기록을 남기는 것이다. 그리고, 사용자 입력값 뿐만 아니라 시스템 호출, DB query, 기타 내부기능에 의해 발생 가능한 에러도 포함하여 처리해야 한다.

### (1) 취약성 판단하기

- 웹 어플리케이션에서 제어할 수 없는 코드를 실행시켜 에러에 빠지게 되었을 경우, 발생하는 오류메시지로부터 자세한 시스템 정보를 얻게 되는지 판단한다.
- 어플리케이션 개발 당시 디버그 정보를 제거하지 않고 그대로 남길 경우, 아래와 같은 디버그 에러 정보가 화면에 뿌려지게 될 수도 있다.



(그림 6-4) 사이트의 오류 메시지 예

## (2) 보호대책

- 웹 어플리케이션 개발자는 생길 수 있는 모든 경우의 에러에 대해 대응할 수 있도록 설계해야 한다.
- 모든 상황에 대해 발생 가능한 에러를 검토하여 그에 대한 조치를 문서화한다.
- 실제로 에러가 발생할 경우에는 필요한 경우 시스템에 대한 최소한의 정보만을 표시한다.
- 발생한 에러의 정도나 횟수에 따라 악의적인 공격을 탐지할 수 있도록 설계한다.
- 어플리케이션이 오류로 인해 제 기능을 다하지 못할 경우에는 서비스 거부 상태가 되도록 설계한다.



## 8. 암호의 안전치 못한 사용

비밀번호가 메모리에 저장되어 있는 상황이나 중요한 데이터를 암호화 시키지 않은 경우 등이 암호를 사용함에 있어 적절하게 사용하지 않는 예이다. 또한 복호화가 용이한 알고리즘을 사용하거나 암호화 과정에 필요한 키 관리가 제대로 안되는 경우도 암호를 안전하게 사용하지 못한 사례이다.

### (1) 취약성 판단하기

- 웹 어플리케이션에서 비밀번호, 신용카드번호, 계좌번호, 개인정보와 같은 중요한 데이터를 입력 받을 경우, 입력 데이터가 그대로 노출되는지 여부를 검토한다.
- 사용된 암호화 알고리즘이 공개적으로 보안성이 검증된 알고리즘인지를 검토한다.
- 암호화/복호화에 사용되는 키의 관리가 적절한지 검토한다.

### (2) 보호대책

- 중요한 정보에 대해서만 암호화를 수행하거나 또는 물리적으로 다른 곳에 중요 정보를 암호화 하여 백업하도록 한다.
- 신용카드번호나 개인신상의 중요 정보를 웹 서버에 암호화하여 저장하는 대신, 간단하게 사용자로부터 입력을 받도록 설계하는 편이 좋다.
- 토큰, 세션변수, 쿠키 같은 예측이 가능한 정보들을 추측하려는 시도를 막을 수 있도록 설계한다.
- 일반 사용자에게 암호화 전 과정이 노출되지 않도록 설계한다.

## 9. 원격 관리 취약점

웹 서버 관리자를 위해 제공되는 원격관리 기능은 여러모로 유용하지만 시스템이 심각하게 공격 받을 수 있는 많은 경로를 제공하기도 한다.

관리자가 편의를 위하여 외부에서 접속할 수 있도록 되어있는 경우 완벽하게 보안이 되어 있지 않다면 해커의 입장에서 손쉽게 웹 서버의 모든 제어권을 가질 수 있다.

### (1) 취약성 판단하기

- 웹 서버 및 웹사이트가 어떻게 관리되는지를 점검한다.
- 원격 인터페이스에 대한 인증정책이 취약한지 여부를 검토한다.
- 웹페이지를 변경하거나 웹 서버 설정을 변경할 때 원격에서 접속가능한지 여부를 판단한다.
- 원격 인터페이스에서 시스템 함수를 호출하는 명령어가 사용될 수 있는지를 검토한다.
- 관리자 페이지의 노출 여부를 검토한다.
- 시스템/네트워크 취약성 점검도구를 이용해서 사이트의 취약성을 주기적으로 점검한다.

### (2) 보호대책

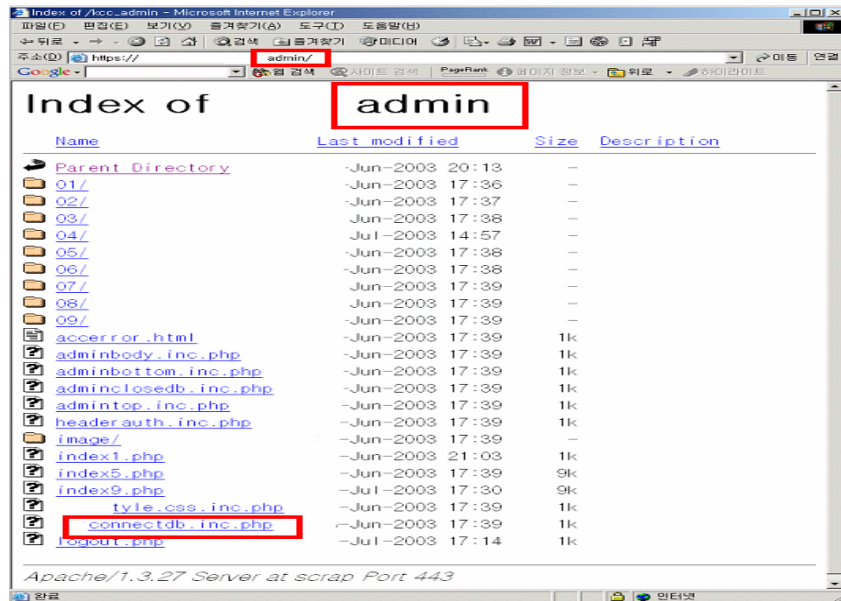
- 가능한 원격에서의 관리자 권한으로의 접근은 없도록 설정한다.
- 부득이하게 원격에서의 접속이 필요한 경우는 VPN을 사용하거나 관리자의 터미널 IP주소만 접근하도록 설정한다.
- 만약 웹 기반의 원격접근이 가능하다면, 인증서나 토큰 기반의 인증 방식이 채택되도록 설계한다.
- 관리자와 일반 사용자 인터페이스가 분리되도록 설계한다.

## 10. 서버의 구성설정

웹 어플리케이션에 대한 구성설정은 개발과 운영의 양면에서 고려해야 한다. 개발자가 기존에 알려진 프로그램들의 보안성 검토 없이 소스코드를 그대로 사용하거나 운영자가 웹 어플리케이션의 설치 당시의 디폴트 값을 수정 없이 적용하여 운영할 경우 취약하다고 할 수 있다.

### (1) 취약성 판단하기

- 시스템/네트워크 취약점 점검도구를 이용해 웹 어플리케이션을 주기적으로 점검 하도록 한다.
- 잘 알려진 보안체크리스트를 적극 활용해서 웹 어플리케이션을 점검한다.
- 잘못된 구성설정의 대표적인 사례는 다음과 같다.
  - 어플리케이션의 보안 패치를 하지 않은 경우
  - 디렉토리 리스팅과 경로 유출 공격을 허용하는 소프트웨어의 잘못된 설정
  - 불필요한 스크립트, 유틸리티, 구성파일, 웹 페이지의 기본 설정값과 백업, 샘플파일의 설치
  - 파일과 디렉토리의 부적절한 권한설정
  - 컨텐츠 관리와 원격 관리와 같은 불필요한 서비스의 제공
  - 설치시 기본적으로 생성되는 계정과 패스워드의 존재
  - 활성화된 관리 및 디버깅 기능이 접근 가능한 경우
  - 시스템 정보가 드러나는 에러 메시지의 노출
  - 잘못 설정된 SSL 인증서 및 암호화 설정
  - 설치시 제공되는 기본적인 인증서의 사용
  - 개발 당시나 설치 당시 테스트 목적으로 만들어 놓은 설정



(그림 6-5) 디렉토리 리스팅 예 - PHP

## (2) 보호대책

- 벤더가 제공하는 보안권고문을 항상 검토하여 어플리케이션을 하도록 한다.
- 보안권고문을 자동으로 받아볼 수 있도록 메일링 리스트에 가입하고, 항상 모니터링 한다.
- 정기적으로 두 개 이상의 다른 취약점 점검도구를 이용해 웹 서버 내부와 외부에서 점검하도록 한다.
- 웹 사이트를 구축 및 운영할 때, 본 가이드를 참고로 하여 보안설정을 하도록 한다.

## 제 2 절 서버측 스크립트 언어 보안

### 1. PHP

#### [PHP 1] register\_globals

php.ini 환경 설정 파일에서 register\_globals이라는 환경 설정변수가 있다. 변수를 공통으로 쓸 것인지에 대한 설정 부분인데, 이 설정은 PHP 4.2.0 이후 버전에는 register\_globals=off 가 디폴트로 설정되어 있다. 즉, 변수를 공통적으로 쓰는게 아니라, 변수가 GET방식인지 POST 방식인지 혹은 전역변수인지를 모두 체크하여 외부에서 값을 변경할 수 없도록 되어 있다.

그런데 이전 PHP버전에서는 register\_globals=on 이 디폴트 설정이었기 때문에 개발자들은 예전에 구현한 프로그램과의 호환성을 위해 register\_globals=on 으로 바꾸고 있는 실정이다.

그렇다면, register\_globals=on 설정을 바꾸지 않고 막을 수 있는 방법은 if문으로 파일명 같은 변수를 검사하여 지정한 값들이 아니면 오류를 표시하고 버리도록 프로그래밍 해야 한다.

#### [PHP 2] 외부 파일 불러오기 방지

외부에서 파일을 불러오는 문제는 php.ini 설정에서 allow\_url\_fopen=off로 설정해두면, 외부에서 파일을 가져올 수 없다.

#### [PHP 3] 특수문자 필터링

GET 방식으로 입력되는 “..” 는 일반적으로 필터링 하도록 설계되고 있다. 그런데 셸에서 “.w./” 는 “..” 와 동일하게 간주된다. 따라서 특수문자에 대한 전반적인 필터링 장치가 필요하다.

php.ini 중 magic\_quotes\_gpc=on 으로 설정하면, “w” 가 “www” 으로 처리된다.

#### [PHP 4] 위험한 함수 사용 방지

공격자는 PHP 취약성을 파악하려고 `phpinfo()` 함수나 `passthru()` 함수를 이용해 웹 서버에 대한 권한을 얻으려고 한다. 이처럼, 공격에 악용될 수 있는 함수<sup>41</sup>들은 `php.ini` 설정 중 `disable_functions` 라인에 추가하면 위험을 줄일 수 있다.

#### [PHP 5] 파일 업로드 방지

웹 보드나 자료실 등에 파일을 업로드 하여 PHP를 실행하는 방법은 현재까지 가장 많이 사용되는 공격 기법이다. 꾸준히 보안 패치가 나오고 있지만, 허술하게 패치한 곳이 많아 공격이 용이하다.

- 자료실이나 게시판 어플리케이션을 작성할 때 PHP 실행파일은 올리지 못하도록 막는다. 올리지 못하는 파일을 검사하는 것보다는 올릴 수 있는 파일을 검사하는 방법이 효과적이다.
- 파일의 확장자 검사를 할 때, “`strcmp(확장자, “php3” );`” 로 검사하면 `pHp3` 나 `phP3`는 구별하지 못한다. 따라서 “`strcasecmp()`” 과 같은 대소문자를 구별하지 않고 비교하는 함수를 사용해야 한다.
- 파일의 확장자 검사를 할 때, “.” 를 기준으로 파일명과 확장자를 구별하는 경우가 많다. 파일명의 앞에서부터 “.” 를 검사할 경우, 만약 “`file.zip.php3`” 로 파일을 올린다면 `zip` 파일로 인식하고 파일은 업로드 된다. 따라서 다음과 같이 프로그래밍 하도록 한다.

```
$check=explode( ".", $a);
if( !strcasecmp($check[sizeof($check)-1], "php3") )
{
echo "php3 확장자는 올릴 수 없음;
exit;
}
```

<sup>41</sup> 주로 `passthru()`, `phpinfo()`, `system()`, `shell_exec()`, `exec()`, `proc_open()` 등이 대상이 된다.

## [PHP 6] superglobal array

URL과 폼, 쿠키의 값은 \$\_GET와 \$\_POST, \$\_COOKIE 등 superglobal array<sup>42</sup>를 통해 접근한다.

superglobal array로부터 값을 사용하기 전에 배열값에 대해 유효성 검사를 한다. 예를 들어, 우편번호의 경우 6자리 숫자 또는 3자리 숫자, 하이픈, 3자리 숫자가 입력 될 것이다. 이런 식으로 입력값의 정규표현을 고려한다면 데이터의 유효성을 쉽게 파악할 수 있게 된다.

## [PHP 7] 해쉬- 파라미터 조작 검증

만약 cookie 필드값이나 hidden form field 등 민감한 데이터를 다룰 경우 해쉬 값을 보냄으로써 제3자에 의해 변경된 값이 아님을 확인해야 한다.

일단 데이터와 해쉬값을 수신한 후, 데이터를 재해쉬하여 수신한 해쉬값이 이전 값과 동일한지를 확인한다.

```
// sending the cookie
$secret_word = 'gargamel';
$id = 123745323;
$hash = md5($secret_word.$id);
setcookie('id',$id.'-'. $hash);

// receiving and verifying the cookie
list($cookie_id,$cookie_hash) = explode('-',$COOKIE['id']);
if (md5($secret_word.$cookie_id) == $cookie_hash) {
    $id = $cookie_id;}
else {
    die('Invalid cookie.');
```

---

<sup>42</sup> 4.1.0 버전 이후 PHP는 웹 서버, 환경설정, 사용자 입력과 관련된 미리 선언된 배열 변수 집합을 추가적으로 제공한다. 여기에 속하는 변수들은 전역변수이며, 이런 변수를 superglobal array(슈퍼 전역변수)라고 부른다.

만약 누군가 cookie의 ID 값을 변경한다면 해쉬 결과는 서로 일치하지 않을 것이다. 운영자는 키값인 “\$secret\_word”가 노출되지 않도록 주의 해야한다. 따라서, 파일로 저장할 경우 누구에게도 노출되지 않도록 보호하는 것이 중요하며, 주기적으로 변경 해야 한다.

## [PHP 8] 접근제어 모듈 - PEAR

자체 개발한 접근제한 모듈보다는 PEAR(PHP Extension and Application Repository)<sup>43</sup> 모듈을 사용한다. “Auth”는 사용자를 위한 cookie 기반 인증 방식이고, “Auth\_HTTP”는 브라우저 기반의 인증 방식이다.

## [PHP 9] 안전한 세션 관리

안전하고 표준화된 세션을 관리하기 위해서 내장된 세션관리 함수를 사용하도록 한다. 주의할 점은, 서버가 세션정보를 어떻게 저장하는지 설정사항을 파악해야 한다. 예를 들어, 세션 콘텐츠가 /tmp 디렉토리에 누구든지 읽을 수 있도록 저장되어 있다면, 서버에 로그가 있는 모든 사용자들은 모든 세션을 볼 수 있다. 따라서, 세션정보를 데이터베이스에 저장하거나 신뢰할 수 있는 사용자만이 접근할 수 있는 파일시스템의 한곳에 저장하도록 한다.

- Php.ini 중 session.save\_path=/tmp가 디폴트로 설정되어 있다. 이 부분을 /tmp/session 처럼 디렉토리를 따로 만들고 session.save\_path=/tmp/session 과 같이 설정 파일을 변경한 후에, /tmp/session 디렉토리를 user와 group 값을 nobody로 설정하고, 퍼미션을 750으로 조정하도록 한다.
- 클라이언트와의 현재 세션값을 재확인 할 경우, 클라이언트의 IP를 저장하여 IP와 세션값을 함께 검사하도록 설계하는 것이 더욱 안전하다.
- 개인신상정보를 변경하는 부분에서는 다시 한번 인증 절차를 거치도록 설계하고, 패스워드와 주민번호는 절대 웹페이지에 보여서는 안된다.
- 네트워크 스니퍼에 의해 세션변수가 스푸핑 되지 않도록, 세션관련 트래픽은 반드시 SSL 통신을 하도록 한다. 모든 PHP 통신을 SSL 통신으로 할 이유는 없으며, 웹 어플리케이션의 보안성을 고려하여 설계하도록 한다.

<sup>43</sup> <http://pear.php.net/package-info.php?package=Auth>

[http://pear.php.net/package-info.php?package=Auth\\_HTTP](http://pear.php.net/package-info.php?package=Auth_HTTP)



## [PHP 10] Cross-site scripting (XSS) 대책

입력값을 필터링 하지 않고 그대로 화면에 보여주지 않도록 한다. 값이 hidden form field나 query string, 웹에 포함되기 전에 반드시 필터링 하는 단계를 거치도록 한다.

PHP에서는 신뢰할 수 없는 데이터에 대한 필터링 툴이 제공되고 있다.

- htmlspecialchars()는 특수문자를 HTML 엔티티 형태로 변환한다. 선택적인 두번째 인자 quote\_style은 작은 따옴표와 큰 따옴표를 어떻게 사용할 지를 말해준다. 기본 모드인 ENT\_COMPAT 은 단지 큰 따옴표만 변환하고 작은 따옴표는 그대로 내버려 둔다. ENT\_QUOTES 가 지정되면, 작은 따옴표와 큰 따옴표 모두를 번역하며, ENT\_NOQUOTES는 작은 따옴표와 큰 따옴표 모두를 번역하지 않는다.
- stripslashes()는 원하는 문자를 필터링 한다. ( ' 과 )을 HTML 엔티티로 변경하는 것은 XSS 공격을 피할 수 있는 보호대책이 된다.

```
$safer = stripslashes($untrusted, array('(' => '&040;', ') => '&041;'));
```

- strip\_tags()는 스트링에서 HTML과 PHP 태그를 제거한다.
- utf8\_decode()는 유니코드 UTF-8 인코딩 스트링내의 ISO-8859-1 문자를 1바이트의 ASCII문자로 변경한다. 때때로 XSS 공격자는 그들의 공격 내용을 유니코드 인코딩으로 숨기려는 시도를 하기 때문에 utf8\_decode()로 악성코드를 발견할 수 있게 된다.

## [PHP 11] 패치

PHP는 런타임 메모리 할당이나 포인터의 개념이 없기 때문에 C 코드와 같은 버퍼 오버플로우는 없다. 그러나 PHP에서 제공하는 함수 자체 또는 확장 버전에서 발생할 수 있는 버퍼 오버플로우를 피하기 위해서는 PHP 모듈에 대한 패치가 중요하다. 따라서, 보안 메일링 리스트<sup>44</sup>에 가입하여 공지되는 패치와 보안권고문을 참고하도록 한다.

---

<sup>44</sup> PHP Mailing Lists: <http://www.php.net/mailling-lists.php>

## [PHP 12] 명령어 주입 공격에 대한 방어

명령어 주입 공격은 필터링 하지 않은 악의적인 명령어를 시스템이나 데이터 베이스에 넘겼을 때 발생한다. 악의적인 명령어 주입 공격을 피하기 위해서 사용자 입력값을 시스템이나 데이터베이스에 넘기기 전에 검사하도록 한다.

현재 웹페이지에 셸을 통해 명령어를 입력 받고 있다면, 우선은 정말로 그러한 인터페이스가 필요한지에 대해 재점검한다. 만약 외부의 프로세스를 실행시키기 위한 인수 등을 사용자로부터 입력 받아야 하는 경우라면, 입력값을 `escapeshellcmd()`와 `escapeshellarg()`을 사용해 점검하도록 한다.

외부 프로그램을 실행하거나 외부 파일을 열기 전에, `realpath()`로 실행시키려는 프로그램이나 파일의 경로명을 정형화 해야 한다. 이런 방법은 모든 `.`(현재 디렉토리)와 `..`(부모디렉토리)를 검출하고, `//`(이중의 디렉토리 구분자)를 제거하게 된다. 일단 경로명을 정형화 한 후에는 기준에 충족하는지를 확인할 수 있다.

만약 사용자 입력값을 SQL query에 입력할 경우, SQL 모듈로 넘기기 전에 `addslashes()`로 입력값을 점검한다. 만약 MySQL을 사용할 경우, `mysql_real_escape_string()`로 점검할 수도 있다(PHP 버전은 4.3.0).

## [PHP 13] 에러 처리

만약 PHP나 연결된 데이터베이스, 외부 프로그램 등의 에러메시지가 임의의 사용자가 볼 수 있도록 화면에 나타난다면, 이것은 시스템의 정보를 노출하기 때문에 공격의 수단으로 활용될 수 있다. 따라서, 사용자에게 에러메시지를 보여주기 보다는 서버의 에러 로그로 남기도록 설정을 바꿔주어야 한다. 따라서 `php.ini` 설정 파일에서 다음과 같이 설정하도록 한다.

```
log_errors = on
display_errors = off
```

## [PHP 14] 암호화

mcrypt extension 함수는 표준화된 많은 암호화 알고리즘을 제공한다. 자신이 개발한 고유한 암호화 알고리즘을 사용하는 대신에 mcrypt()를 사용하도록 한다.

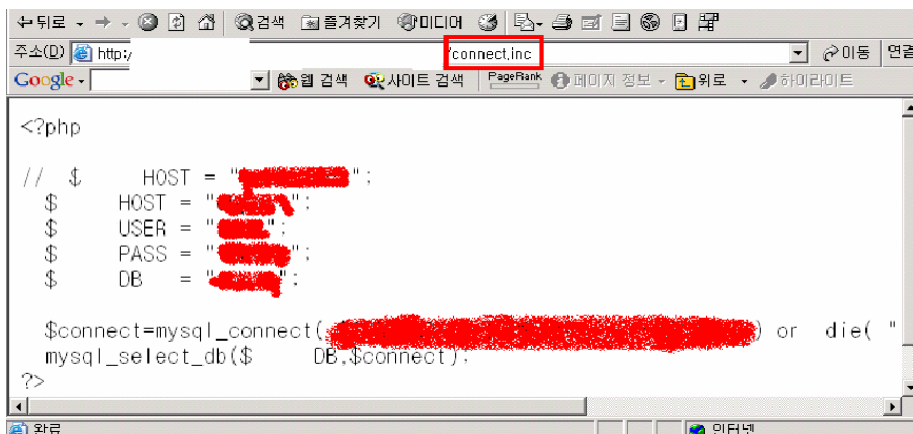
또한 키 관리에도 주의해야 한다. 좋은 방법은 키를 서버에 저장하지 않으며, 사용자가 입력할 수 있는 프롬프트 화면을 화면에 보이지 않도록 하는 것이다.

## [PHP 15] SSL 설정

원격 관리 기능을 실행할 경우 콘텐츠의 스니핑을 막기 위해서 SSL 연결을 하도록 한다. 원격 관리 모듈이 third-party 소프트웨어로 설치하는 경우라면, 디폴트로 설정된 관리자 이름과 패스워드를 변경하도록 한다. 그리고, 디폴트로 설정된 관리자 접속 URL을 반드시 변경하도록 한다.

## [PHP 16] 데이터베이스의 분리

자료실이나 각종 문서가 링크되어 있는 페이지에서는 다운로드시 데이터베이스와의 연동 경로가 노출되지 않도록 주의해야 한다.



```
<?php
// $ HOST = " [REDACTED] ";
$ HOST = " [REDACTED] ";
$ USER = " [REDACTED] ";
$ PASS = " [REDACTED] ";
$ DB = " [REDACTED] ";

$connect=mysql_connect([REDACTED]) or die( "
mysql_select_db($ DB,$connect);
?>
```

(그림 6-6) 데이터베이스 접속 계정 정보 노출 예

## [PHP 17] 환경 설정 파일

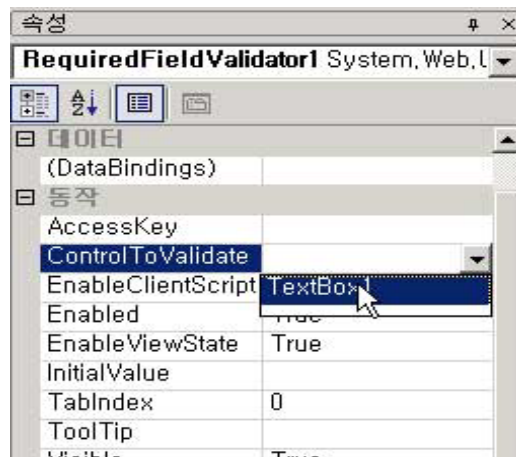
PHP의 환경 설정 파일인 두개의 php.ini 샘플 파일(`php.ini-dist`와 `php.ini-recommended`) 중 `php.ini-recommended`를 사용하도록 한다.

## 2. ASP.NET

### [ASP.NET 1] 입력값의 유효성 검사

사용자 입력값의 타입, 길이, 포맷, 범위 등에 대한 유효성 검사를 한다. 입력값에 대한 유효성 검사 요령은 안전한 데이터 기준에 대해서 우선 점검한 후, 수용할 수 없는 악성 데이터 기준을 점검한다. 그리고, 클라이언트측 유효성 검사를 신뢰해서는 안된다.

- string form field 입력값은 정규표현(regular expression)을 사용해서 유효성 검사를 한다(예를 들어, RegularExpressValidator 컨트롤을 사용한다)
- 정규 HTML 컨트롤, query strings, cookies, form field 등의 입력값에 대해 Regex 클래스를 사용하여 입력값을 검사한다.
- 데이터가 입력되는 곳에는 RequiredFieldValidator 컨트롤을 사용하도록 한다.
- 서버 컨트롤의 범위 체크는 RangeValidator 컨트롤을 사용한다.
- free form에 대한 입력 부분에서는 보안통제를 우회하는 악성 데이터의 입력에 주의해야 한다.
- 사용자가 입력한 값을 화면에 다시 보여줄 경우, HtmlEncode와 UriEncode로 인코딩 한다.
- 지정된 가상 경로 혹은 상대 경로를 서버의 실제 디렉토리에 매핑하는 MapPath는 다른 어플리케이션을 매핑할 경우, 적절한 위치에서 사용을 제한하도록 해야 한다.
- ASP.NET version 1.1 validateRequest 옵션은 활성화 한다.
- URLScan을 웹 서버에 설치하여 정기적으로 점검한다.
- HttpOnly cookie 옵션은 XSS 공격을 방어할 수 있다(IE6.1 이후에 적용).



(그림 6-7) 유효성 검사 예

## [ASP.NET 2] 인증

- 사이트를 공개영역과 인증을 요구하는 제한영역으로 구별하여 설계한다.
- 보안과 비보안폴더로 구별된 사이트에서는 절대경로의 URL을 사용하고, 주요 페이지로 이동할 경우 사용자 인증을 다시 한번 하도록 설계한다.
- credentials나 authentication cookie의 경우, SSL을 사용하거나 암호화 하고 무결성 체크(Protection="ALL")를 하도록 한다.
- slidingExpiration 속성은 "false"로 설정하고, SSL로 보호되지 않는 authentication cookie는 타임아웃을 설정하도록 한다.
- form authentication cookie는 requireSSL 속성과 Secure cookie property를 사용해서 HTTPS 연결로 제한하도록 한다.
- application cookie는 유일한 이름과 경로의 조합으로 되어 있어야 한다.
- 패스워드는 사용자 공간에 저장하지 않는다. 대신 패스워드에 대한 힌트를 저장하도록 한다.
- 강력한 패스워드 정책을 세운다.
- <credentials> 구문은 form authentication을 위한 <forms>구문 안에 사용하지 않도록 한다.

### [ASP.NET 3] 권한

- URL authorization은 페이지와 디렉토리의 접근제어를 위해 사용되어야 한다.
- 파일 권한 정책은 Windows 인증과 함께 사용되어야 한다.
- 주요 클래스와 해당 멤버에 대한 안전한 접근을 위해 권한 설정을 하도록 한다.

### [ASP.NET 4] 민감한 데이터 관리

- 중요 정보를 다룰 때는 SSL을 사용하도록 한다.
- 민감한 정보는 cookie나 hidden form field, query string에 저장하지 않는다.
- 민감한 정보는 캐쉬에 저장하지 않는다. output cache는 off 상태로 설정한다.
- Web.config와 Machine.config 파일 안에 텍스트 패스워드를 피하도록 한다.

### [ASP.NET 5] 세션 관리

- session cookie는 인증을 요구하는 모든 페이지에서 SSL로 보호되어야 한다.
- 필요 없다면, session state service는 비활성화 시키고, session state service는 least-privileged account로 운영한다.
- SQL 서버에 연결할 때 Windows 인증이 선행되도록 설계한다.
- connection string은 AspNet\_setreg.exe로 암호화 되어야 한다.

### [ASP.NET 6] 파라미터 조작 방지

- view state는 MAC(message authentication code)으로 보호되어야 한다.
- 서버의 주요 정보가 query string에 포함되려면 query string의 해쉬 정보로 대신해야 한다.
- 모든 입력 파라미터에 대해 유효성 검사를 수행한다.

## [ASP.NET 7] 에러 처리

- 구조화된 에러 핸들링을 설계한다.
- 자세한 에러 사항은 사용자 페이지에 뿌리지 말고, 서버에 로그 파일로 남기도록 한다.
- 페이지 레벨 혹은 어플리케이션 레벨의 에러 핸들러를 구현하도록 한다.
- 어플리케이션이 에러와 예외 조건을 구별하여 처리해야 한다.

## [ASP.NET 8] 환경 설정

- 설정파일에 대한 수정시도는 HttpForbiddenHandler를 사용해 막도록 한다.
- ASP.NET는 least-privileged account로 운영하도록 한다.
- 개인신상정보 등은 Aspnet\_setreg.exe에 의해 <processModel>상에서 암호화되어야 한다.
- machine-wide 정책을 설정하기 위해서, Web.config 세팅을 Machine.config 내 allowOverride="false"로 잠근다.



[표 6-2] Web.config 파일의 설정

구 문	설 정 권 고
<trace enabled=" false" >	서버의 tracing 기능을 비활성화 시킨다.
<globalization>	Request와 Response 인코딩 방식을 적절하게 설정한다.
<httpRuntime>	maxLength을 적절히 설정하면 사용자가 큰 파일을 업로드하는 것을 막을 수 있다(선택사항임).
<compilation debug=" false" .../>	서버의 디버그 컴파일 옵션은 비활성화 시킨다.
① <pages enableViewState=" false" .. /> ② <pages enableViewState=" true" enableViewStateMac=" true" />	① 만약 어플리케이션이 view state를 사용하지 않는다면, enableViewState 옵션을 false로 설정한다. ② 만약 어플리케이션이 view state를 사용한다면, enableViewState 옵션을 true로 설정하고, enableViewStateMac 옵션도 true로 설정한다.
<customErrors mode=" on" />	사용자정의 오류(custom error) 페이지에는 자세한 오류 정보를 제외하고 전달하도록 한다.
<authentication>	Authentication mode는 어플리케이션의 요구사항을 충족할 수 있도록 적절하게 설정하는데, 특정한 authentication 타입을 사용하기 위해서 allowOverride=" false" 로 설정된 <location> 요소를 사용한다.  <location path=" " allowOverride=" false" > <system web> <authentication mode=" Windows" /> </system.web> </location>
<forms>	Form authentication 설정은 보호되어야 한다.  <forms loginUrl=" RestrictedWlogin.aspx" protection=" All" requireSSL=" true" timeout=" 10" name=" AppNameCookie" path=" WFormAuth" slidingExpiration=" true" />

<p>&lt;identity&gt;</p>	<p>가장ID(Impersonation identities)<sup>45</sup>는 Aspnet_setreg.exe를 사용해서 레지스트리 값에서 암호화 되어야 한다.</p> <pre>&lt;identity impersonate="true" userName="registry:HKLMSOFTWARE\YourApp\identity\AS PNETSETREG, userName" password="registry:HKLMSOFTWARE\YourApp\identity\AS PNET_SETREG,password"/&gt;</pre>
<p>&lt;authorization&gt;</p>	<p>Role name의 현재 포맷을 검사한다.</p>
<p>&lt;machineKey&gt;</p>	<p>만약 다수의 ASP.NET 어플리케이션이 같은 웹 서버에서 운영될 때, IsolateApps 옵션을 세팅하여 각 어플리케이션의 비밀키를 생성하도록 해야 한다.</p> <pre>&lt;machineKey validationKey="AutoGenerate, IsolateApps" decryptionKey="AutoGenerate, IsoIateApps" validation="SHA1" /&gt;</pre>
<p>&lt;sessionState&gt;</p>	<p>① 만약 mode=StateServer라면, credential은 Aspnet_setreg.exe로 암호화 하여 레지스트리에 저장한다.  ② 만약 mode=SQLServer라면, windows 인증은 state store database에 연결할 때 사용되고, credential은 Aspnet_setreg.exe.를 사용해서 암호화하여 레지스트리에 저장한다.</p>
<p>&lt;httpHandler&gt;</p>	<p>사용안된 파일 타입은 파라미터 조작 공격을 막기 위해서 HttpForbiddenHandler에 매핑 시킨다.</p> <p>예)</p> <pre>&lt;add verb="*" path="*.rem" type="System.Web.HttpForbiddenHandler"/&gt;</pre>
<p>&lt;webServices&gt;</p>	<p>사용하지 않는 프로토콜은 비활성화 시킨다.  자동으로 만들어지는 WSDL(Web services description language)는 비활성화 시킨다(선택사항임)</p>

<sup>45</sup> ‘가장(impersonation)’의 의미는 말 그대로 계정을 흉내 낸다는 뜻이다. 가장이 필요한 어플리케이션은 대개 서버측 어플리케이션으로서 클라이언트가 서버에 대한 권한이 있는지를 손쉽게 검사하기 위해 사용되는 방법이다.

### 3. JSP(Java server pages)

#### [JSP 1] 패치

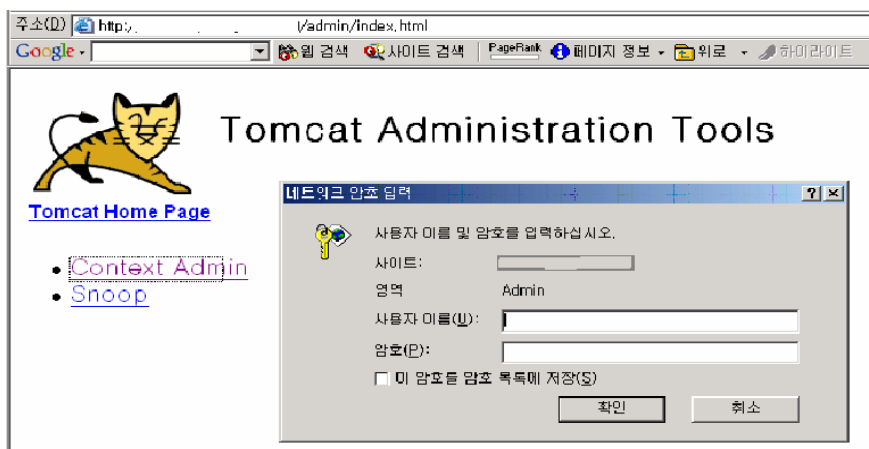
한때 GET 방식으로 “%3f.jsp” 를 넘겨주면 디렉토리가 리스팅 되는 Tomcat 버그가 발견되어 대부분의 JSP 기반의 웹 서버가 공격 당한 경험이 있다. 이 버그는 Tomcat 모듈 자체의 문제기 때문에 패치가 되기 전까지는 어쩔 수 없이 방치될 수밖에 없다.

만약 아파치+Tomcat+JSP 환경의 웹 서버라면, 아파치에서 다음과 같이 설정하도록 하거나, Tomcat 모듈의 보안 패치를 최신으로 유지하도록 한다.

```
<LocationMatch "/(%3f|W?)W.jsp">  
    AllowOverride None  
    Deny from all  
</LocationMatch>
```

#### [JSP 2] admin 디렉토리 관리

아파치+Tomcat+JSP 환경에서는 /admin 디렉토리가 자동으로 마운트되기 때문에 상당히 주의해야 한다. 또한, Tomcat을 root 권한으로 운영해서는 안된다.



(그림 6-8) Tomcat 관리자 페이지 노출 예

### [JSP 3] 디렉토리 구분

JSP를 쓰면서 PHP를 쓰게 해놓는 사이트의 경우, 같은 디렉토리에 JSP와 PHP를 함께 두는 경우 심각한 보안문제가 야기 될 수 있다.

80번 포트에서는 아파치와 연동된 모듈로서 JSP가 동작하여 문제가 없지만, 8080 포트에서는 PHP가 인식되지 않아 8080포트로 접속한 후 PHP 파일을 불러내면 소스 코드를 화면에 보여주게 된다. 이를 막기 위해서는 JSP와 PHP 디렉토리를 다르게 하여 설치해야 한다.

만약 같은 디렉토리를 써야 할 경우라면, 다음과 같이 설정한다.

- 아파치 환경에서 JSP 디렉토리를 링크하는 방법

```
>ln -s JSP디렉토리 아파치 디렉토리
```

- 아파치 환경에서 디렉토리를 인식 시키도록 설정하는 방법

```
ScriptAlias /jsp/ "/usr/local/jsp/dics/"
```

### [JSP 4] 디렉토리 리스팅

디렉토리 리스팅을 대수롭지 않게 생각할 수 있지만, 웹 페이지의 중요 소스의 백업 파일이나 민감한 정보가 드러날 수 있기 때문에 다음과 같이 환경을 설정하는 것이 중요하다.

- Tomcat의 경우 web.xml

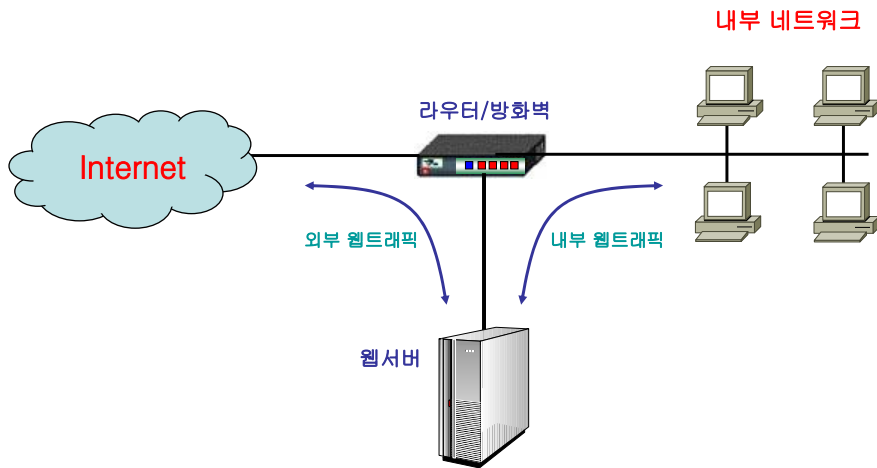
```
<init-param>  
  <param-name>listings</param-name>  
  <param-value>>false</param-value>  
</init-param>
```

- Resin 의 경우 resin.conf

```
<directory-servlet>>false</directory-servlet>
```

## 제 7 장 웹 서버를 위한 안전한 네트워크 구성

웹 서버는 서비스의 특성상 인터넷에 연결되어 외부에 공개되어야 하기 때문에, 보안이 잘 설정되어 있어도 해킹의 가능성이 항상 존재한다. 대부분의 경우 웹 서버는 전세계 어느 곳에서든지 접근이 가능하기 때문에 끊임없는 해킹 시도가 있을 수 있고, 웹 기술의 발전에 따라 항상 새로운 취약점이 발견될 가능성이 있다. 따라서 웹 서버의 해킹 가능성을 인정하고, 웹 서버에 해킹이 발생하더라도 더 큰 피해를 막을 수 있도록 안전한 네트워크 구성이 필요하다.



(그림 7-1) 웹 서버를 위한 네트워크 구성 예시-1

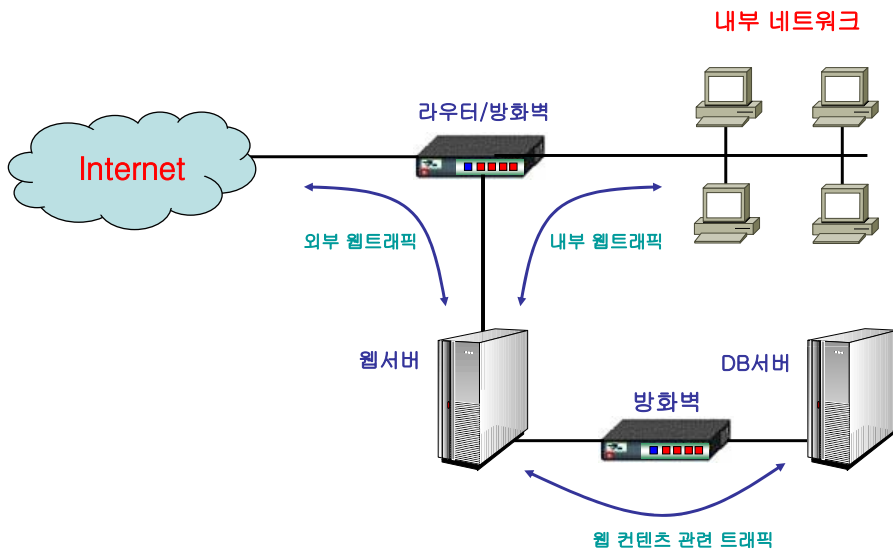
웹 서버를 효율적으로 보호하고, 웹 서버 해킹으로부터 내부 네트워크를 보호하기 위해 (그림 7-1)과 같이 웹 서버를 별도의 보호된 서브넷에 위치시키는 것이 안전하다. 라우터/방화벽단에서 웹 서버와 내부 네트워크를 분리시키고 불필요한 트래픽 흐름을 제한한다.

인터넷과 웹 서버간의 트래픽이 내부 네트워크로 흘러 들어가지 못하도록 하고, 내부 네트워크 트래픽이 웹 서버에서 관찰되지 못하도록 한다. 웹 서버쪽 서브넷은 웹트래픽만 송수신할 수 있도록 하고, 웹 서버에서 내부 네트워크로 접속을 시도할 수 없도록 한다. 라우터/방화벽이 하나 이상인 경우 웹트래픽 필터링 전용과 내부

네트워크와 인터넷간의 트래픽 필터링 전용을 분리하여 운영하는 것이 좋다.

네트워크를 이렇게 구성하면 웹 서버가 해킹 당했다 하더라도 내부 네트워크에서 흘러다니는 기밀 정보가 노출될 염려가 없고, 웹 서버에서 내부 네트워크로의 침투가 어려워진다. 또한 라우터/방화벽에서 웹 트래픽만을 별도로 관찰하고 제어하기가 용이하다.

웹 서버로 들어오는 연결은 기본 연결포트인 80/tcp 포트만 허용한다. 인증과 암호화를 위해 SSL을 사용하는 경우 443/tcp 포트도 허용하도록 한다. 웹 서버에서 나가는 TCP 연결은 기본적으로 모두 블로킹(blocking)한다. 또한 모든 UDP와 ICMP 트래픽을 차단하도록 한다. 만약 호스트네임 찾기(lookup)를 위해 DNS를 사용해야 한다면 53/udp만을 허용한다. 이 경우 웹 서버에서 내부 DNS서버로의 연결만을 허용하고, 내부 DNS가 요청을 외부 DNS로 릴레이 하도록 한다.



(그림 7-2) 웹 서버를 위한 네트워크 구성 예시-2

웹사이트에서 이메일 서비스, 디렉토리 서비스, 데이터베이스 서비스를 사용해야 한다면, 이 서비스를 제공하는 서버들이 웹 서버하고만 통신할 수 있도록 구성하는 것이 좋다. (그림 7-2)처럼 이 서버들을 별도의 보호된 서브넷에 위치시키고 웹 서버하고만 데이터를 주고 받을 수 있도록 구성한다. 인터넷과 내부 네트워크에서 이 서버들에게 직접 트래픽을 보낼 수 없도록 하고, 중간에 위치한 방화벽에서 허용된

프로토콜 이외의 모든 트래픽을 차단한다. 또한 웹 서버와 이 서버들 사이의 트래픽이 내부 네트워크로 흘러가지 않도록 한다.

추가적으로 라우터와 방화벽에서 소스라우팅(source routing)을 금지시키고, 웹 서버 및 웹 서버와 연동되는 다른 서버들에서 IP 포워딩(IP forwarding)과 소스라우팅을 금지시킨다.



이 페이지는 빈 페이지입니다.

## 참 고 자 료\*

- [1] Apache HTTP Server Version 1.3 Documentation
- [2] Tracy, M., Jansen, W. and McLarnon, M., Guidelines on Securing Public Web Servers, NIST, 2002.
- [3] Klaus-Peter Kossakowski and Julia Allen, Securing Public Web Servers, CERT, 2000
- [4] Richard Power, “2002 CSI/FBI Computer Crime and Security Survey” , Computer Security Issues & Trends, 2002.
- [5] A Guide to Building Secure Web Applications, version 1.0, OWASP.
- [6] The Ten Most Critical Web Application Security Vulnerabilities, OWASP, 2003.
- [7] Security at the Next Level - Are Your Web Applications Vulnerable, SPI LABS.
- [8] Mark E. Donaldson, Inside the Buffer Overflow Attack: Mechanism, Method, & Prevention, SANS, 2002.
- [9] Ken Coar, Using .htaccess Files with Apache, 2000.
- [10] Ken Coar, Securing Your Web Pages with Apache, 2000.
- [11] 차주연, 윈도우 웹 서버 보안, 대림출판사, 2002.
- [12] 유형수, 웹 서버 관리자를 위한 IIS 5.0, 혜지원출판사, 2001.
- [13] PHP and OWASP Top ten security vulnerabilities, OWASP 2003
- [14] Improving web security - threats and countermeasures, Microsoft, MSDN Home > MSDN Library > .NET Development > .NET Security
- [15] 안전한 프로그래밍 지침, 2002,  
<http://doc.kldp.org/HOWTO//html/Secure-Programs-HOWTO/index.html>
- [16] 이승혁 저, PHP 웹 프로그래밍 가이드, 마이트press, 2001
- [17] PHP 매뉴얼, <http://www.php.net/>
- [18] 태요의 ASP 강좌, <http://www.taeyo.pe.kr/>
- [19] Shinichi Sato 저, IIS5 ASP Scripting guide, 영진출판사, 2001
- [20] 김태영 외 1인 공저, Taeyo’ s ASP 입문, 삼양출판사, 2002
- [21] ASP.NET 보안,  
<http://ko.gotdotnet.com/quickstart/aspplus/doc/tracelogapp.aspx>
- [22] 소설 같은 JSP, <http://www.jabook.org/index.html>
- [23] 이재갑 외 3인 공저, about JSP, 영진출판사, 2001
- [24] 유경상, ASP.NET과 IIS, 그리고 윈도우 2000의 보안 관계, ZDNet Korea, 2002

---

\* 일부 참고자료는 내부 협의에 의해 생략하였음

이 페이지는 빈 페이지입니다.

## 부 록

### 부 록 1. OS 벤더 보안사이트

OS 종류	사이트 주소
FreeBSD	<a href="http://www.freebsd.org/security/">http://www.freebsd.org/security/</a>
NetBSD	<a href="http://www.netbsd.org/Security/">http://www.netbsd.org/Security/</a>
OpenBSD	<a href="http://openbsd.org/security.html">http://openbsd.org/security.html</a> <a href="ftp://ftp.openbsd.org/pub/OpenBSD/patches/">ftp://ftp.openbsd.org/pub/OpenBSD/patches/</a>
HP Tru64	<a href="http://h30097.www3.hp.com/unix/security-download.html">http://h30097.www3.hp.com/unix/security-download.html</a>
IBM AIX	<a href="http://techsupport.services.ibm.com/rs6000/notifications">http://techsupport.services.ibm.com/rs6000/notifications</a>
Solaris	<a href="http://sunsolve.sun.com/pub-cgi/show.pl?target=home">http://sunsolve.sun.com/pub-cgi/show.pl?target=home</a>
SGI IRIX	<a href="http://www.sgi.com/support/security/">http://www.sgi.com/support/security/</a>
Microsoft	<a href="http://www.microsoft.com/korea/security/">http://www.microsoft.com/korea/security/</a> <a href="http://www.microsoft.com/security/">http://www.microsoft.com/security/</a>
Caldera	<a href="http://www.caldera.com/support/security/">http://www.caldera.com/support/security/</a>
Debian	<a href="http://www.debian.org/security/">http://www.debian.org/security/</a>
Mandrake	<a href="http://www.mandrakesecure.net/en/advisories/">http://www.mandrakesecure.net/en/advisories/</a>
Redhat	<a href="http://www.redhat.com/apps/support/errata/">http://www.redhat.com/apps/support/errata/</a>
Slackware	<a href="http://www.slackware.com/security/">http://www.slackware.com/security/</a>
Suse	<a href="http://www.suse.com/us/security/">http://www.suse.com/us/security/</a>
Turbo	<a href="http://www.turbolinux.com/security/">http://www.turbolinux.com/security/</a>

## 부 록 2. 취약점 정보사이트

사이트 명	사이트 주소
Securityfocus	<a href="http://www.securityfocus.com/">http://www.securityfocus.com/</a>
CERTCC-KR	<a href="http://www.certcc.co.kr/">http://www.certcc.co.kr/</a>
Securitymap	<a href="http://www.securitymap.net/">http://www.securitymap.net/</a>
AusCERT	<a href="http://www.auscert.org.au/">http://www.auscert.org.au/</a>
Linuxsecurity	<a href="http://www.linuxsecurity.com/">http://www.linuxsecurity.com/</a>
Macromedia Security Zone	<a href="http://www.macromedia.com/devnet/security/security_zone/">http://www.macromedia.com/devnet/security/security_zone/</a>
securiTeam.com	<a href="http://www.securiteam.com/">http://www.securiteam.com/</a>
symantec security response	<a href="http://securityresponse.symantec.com/">http://securityresponse.symantec.com/</a>
ICAT	<a href="http://icat.nist.gov/">http://icat.nist.gov/</a>

## 부 록 3. 동적 콘텐츠 보안 사이트

사이트 명	사이트 주소
asp alliance	<a href="http://www.aspalliance.com/">http://www.aspalliance.com/</a>
www.cgisecurity.com	<a href="http://www.cgisecurity.com/lib/">http://www.cgisecurity.com/lib/</a>
Exploiting Common Vulnerabilities in PHP Application	<a href="http://www.securereality.com.au/studyinscarlet.txt">http://www.securereality.com.au/studyinscarlet.txt</a>
Java Security	<a href="http://java.sun.com/security/">http://java.sun.com/security/</a>
Java Security Frequently Asked Questions	<a href="http://www.cs.princeton.edu/sip/faq/java-faq.php3">http://www.cs.princeton.edu/sip/faq/java-faq.php3</a>
Open Web Application Security Project	<a href="http://www.owasp.org/">http://www.owasp.org/</a>
www.asp.net	<a href="http://www.asp.net/">http://www.asp.net/</a>

#### 부 록 4. 아파치 웹 서버 보안 사이트

사이트 명	사이트 주소
ApacheWeek Security	<a href="http://www.apacheweek.com/security/">http://www.apacheweek.com/security/</a>
Apache Tutorials	<a href="http://httpd.apache.org/docs/misc/tutorials.html">http://httpd.apache.org/docs/misc/tutorials.html</a>
Apache SSL	<a href="http://www.apache-ssl.org/">http://www.apache-ssl.org/</a>
Securing Your Web Pages with Apache	<a href="http://apache-server.com/tutorials/LPauth1.html">http://apache-server.com/tutorials/LPauth1.html</a>
The Apache Korea Group	<a href="http://www.apache.kr.net/">http://www.apache.kr.net/</a>
Apache-server.com	<a href="http://apache-server.com/">http://apache-server.com/</a>

#### 부 록 5. IIS 웹 서버 보안 사이트

사이트 명	사이트 주소
IIS 5.0 Resource Guide - Chapter 9 Security	<a href="http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/iis/iis5/reskit/iis50rg/iischp9.asp">http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/iis/iis5/reskit/iis50rg/iischp9.asp</a>
IIS 5.0 Baseline Security Checklist*	<a href="http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/tools/chklist/iis5cl.asp">http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/tools/chklist/iis5cl.asp</a>
IIS 5.0 Security	<a href="http://www.microsoft.com/windows2000/en/server/iis/default.asp?url=/WINDOWS2000/en/server/iis/htm/core/iiabts.sc.htm">http://www.microsoft.com/windows2000/en/server/iis/default.asp?url=/WINDOWS2000/en/server/iis/htm/core/iiabts.sc.htm</a>
Secure Internet Information Services 5 Checklist	<a href="http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/tools/chklist/iis5chk.asp">http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/tools/chklist/iis5chk.asp</a>
NSA Guide to the Secure Configuration and Administration of Microsoft IIS 5.0	<a href="http://nsa1.www.conxion.com/win2k/guides/w2k-14.pdf">http://nsa1.www.conxion.com/win2k/guides/w2k-14.pdf</a>
eEye Advisories and Alerts	<a href="http://www.eeye.com/html/Research/Advisories/index.html">http://www.eeye.com/html/Research/Advisories/index.html</a>

\* 우측 사이트 주소가 변경되었다면 MS사 사이트에서 접속해서 좌측 사이트 명에 대해 검색을 수행하여 찾으십시오.

부 록 6. HTTP/1.1 상태코드\*

상태 코드	설명
100	Continue
101	Switching Protocols
200	OK
201	Created
202	Accepted
203	Non-Authoritative Information
204	No Content
205	Reset Content
206	Partial Content
300	Multiple Choice
301	Moved Permanently
302	Found
303	See Other
304	Not Modified
305	Use Proxy
306	(Unused)
307	Temporary Redirect
400	Bad Request
401	Unauthorized
402	Payment Required
403	Forbidden
404	Not Found
405	Method Not Allowed
406	Not Acceptable
407	Proxy Authentication Required
408	Request Timeout
409	Conflict
410	Gone
411	Length Required
412	Precondition Failed
413	Request Entity Too Large
414	Request-URI Too Long

---

\* RFC2616 참조

415	Unsupported Media Type
416	Requested Range Not Satisfiable
417	Expectation Failed
500	Internal Server Error
501	Not Implemented
502	Bad Gateway
503	Service Unavailable
504	Gateway Timeout
505	HTTP Version Not supported



# 웹 서버 보안관리 가이드

---

2003년 9월 인쇄

2003년 9월 발행

- 발행인 : 김 창 곤
- 발행처 : 한국정보보호진흥원  
서울시 송파구 가락동 78번지  
IT 벤처 타워 서관  
전화 : (02)4055-114
- 인쇄처 : 호정씨앤피  
전화 : (02)2277-4718

---

<비매품>

1. 본 연구보고서는 정보통신부의 출연금으로 수행한 정보통신 연구 개발사업의 연구결과입니다.
2. 본 연구보고서의 내용을 발표할 때에는 반드시 정보통신부의 정보통신개발 사업의 연구 결과임을 밝혀야 합니다.
3. 본 연구보고서는 한국정보보호진흥원이 판권을 소유하고 있으며, 당 원의 허가 없이 무단 전재 및 복사를 금합니다.