❖ Apache 웹 서버 보안문제

- 웹 서버/클라이언트/애플리케이션 자체의 버그
- 웹 서버/클라이언트/애플리케이션 설정의 오류
- 침입차단시스템의 웹 서비스 오픈

- SANS TOP20 Vulnerabilities

Top Vulnerabilities to Windows Systems	Top Vulnerabilities to UNIX Systems
W1. Web Servers & Services	U1. Bind Domain Name Systems
W2. Workstation Services	***U2. Web Server***
W3. Windows Remote Access Services	U3. Authentication
W4. Microsoft SQL Server (MS-SQL)	U4. Version Control Systems
W5. Windows Authentication	U5. Mail Transport Services
W6. Web Browsers	U6. Simple Network Management Protocol
W7. File-Sharing Applications	U7. Open Secure Socket Layer
W8. LSASS Exposures	U8. Misconfiguration of Enterprise Services
W9. Mail Client	U9. Databases
W10. Instant Messaging	U10. Kernels

❖ Apache 웹 서버 프로세스를 위한 계정 관리

- Apache 서버의 설치 및 구동을 위한 계정
  - 운영체제에 로그인 하여 Apache 설치
  - 웹 서버 시작 및 종료
  - *웹 서비스를 위한 포트로 1024번 미만 포트번호를 사용하기 위해서는 이 계정이 root 이여야 함.*
- 웹 서버 프로세스를 위한 계정
  - 일반 사용자의 웹 접속을 처리하기 위하여 사용되는 프로세스가 이용
  - *반드시 로그인 할 수 없는 계정, 즉 쉘(Shell)이 없는 계정으로 설치*
  - 일반적으로 사용자 ID와 그룹으로 쉘이 없는 "*nobody*" 계정 이용

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

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

  - Apache 설정파일(http.conf)에 "User", "Group" 지시자(directive) 설정을 통해 활성화

```
User nobody
Group nobody
```

❖ Apache 웹 서버 홈 디렉터리 관리

- DocumentRoot
  - 모든 웹 컨텐트가 저장될 디렉터리 구조, 웹을 통해 공개
  - *시스템 루트 파일시스템 등과는 별도의 파일시스템 사용 필요*
  - Apache 초기 설치 시, DocumentRoot = *htdocs*
- DocumentRoot 변경
  - Apache 설정파일(http.conf)에서 임의의 디렉터리(/usr/local/www)로 변경

```
#DocumentRoot "/usr/local/apache/htdocs"
DocumentRoot "/usr/local/www"
```

- ChrootDir 설정
  - 웹 서버의 최상위 디렉터리를 임시 교체
  - 웹 서버를 통한 공격 발생 시 피해를 지정해 놓은 디렉터리 범위로 축소

❖ 불필요한 CGI 스크립트 관리

- Apache 초기 설치 시, 기본적으로 cgi-bin 디렉터리 존재하는 모든 스크립트 제거

❖ Apache 설정파일 관리 [1]

- 디렉터리 Listing 방지
  - 웹 브라우저를 통해 존재하지 않는 URL 입력 시, 기본적으로 디렉터리 리스트를 보여주는 것을 방지
  - 설정파일(http.conf) 변경
    "*Options*" 지시자(directive)에서 "*Indexes*" 옵션 제거
- Symbolic Link 사용 방지
  - 기존의 웹 문서 이외의 파일시스템에 접근하는 것이 가능
  - 가령, 시스템 Root 디렉터리(/)를 링크하게 되면, Nobody 권한으로 모든 파일시스템에 접근 허용
  - 설정파일(http.conf) 변경
    "*Options*" 지시자(directive)에서 "*FollowSymLinks*" 옵션 제거
- SSI (Server-Side Includes) 사용 제한
  - HTML 페이지 내에 존재하며, 동적인 웹 페이지 제공
  - "*exec cmd*"를 사용해서 임의의 스크립트나 프로그램 실행이 가능
  - 설정파일(http.conf) 변경
    "*Options*" 지시자(directive)에서 "*IncludesNoExec*" 옵션 제거
- CGI 실행 디렉터리 제한
  - 임의의 사용자들이 CGI 스크립트들을 어느 디렉터리에서나 실행 가능하도록 할 경우,
    악의적인 사용자가 CGI 업로드 후 이를 실행하여 공격 가능
  - CGI 프로그램의 실행은 관리자가 지정한 특정 디렉터리에서만 실행 가능하도록 제한
  - 설정파일(http.conf) 변경
    "*ScriptAlias*" 지시자(directive)에서 실행 가능한 디렉터리 제한

```
ScriptAlias /cgi-bin/ "/usr/local/apache/cgi-bin/"
```

❖ Apache 설정파일 관리 [2]

- "Options" 지시자(directive) 및 설정 값

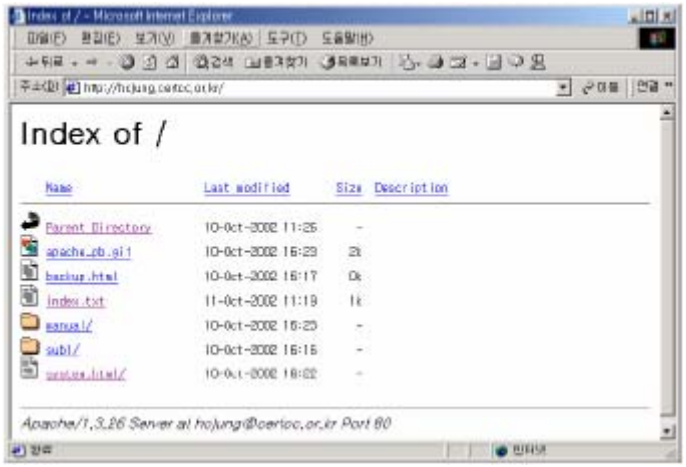
옵 션 값	설 명
All	MultiViews를 제외한 모든 옵션을 줌(default 설정값임)
None	옵션을 주지 않음
ExecCGI	CGI 프로그램 실행을 가능하게 함
FollowSymLinks	심볼릭 링크로의 이동을 가능하게 함
Includes	Server Side Includes를 가능하게 함
IncludesNOEXEC	Server-side includes는 가능하지만 CGI 스크립트나 프로그램들은 실행할 수 없도록 함.
Indexes	해당 디렉토리 안에 DirectoryIndex에 명기된 파일(index.html 등)이 없을 경우 디렉토리와 파일 목록을 보여줌
MultiViews	유사한 파일이름을 찾아 주는 기능을 실행함(예를들어 index라고만 입력하더라도 index.*를 찾아 보여줌)
SymLinksIfOwnerMatch	The server will only follow symbolic links for which the target file or directory is owned by the same user id as the link.

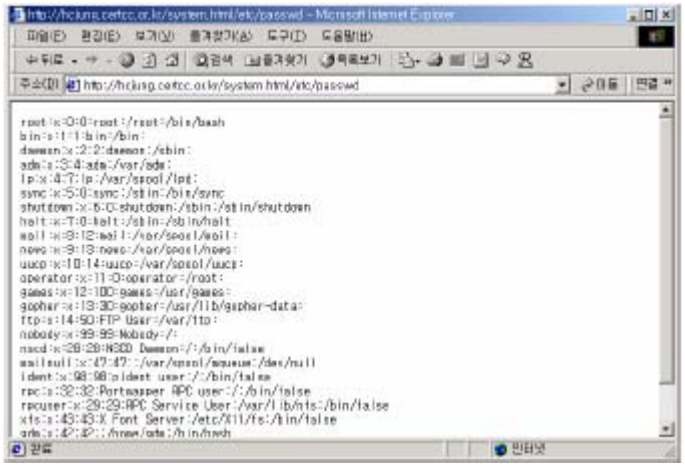
❖ Apache 설정파일 관리 [3]

- 보안관리가 잘못된 설정파일(http.conf) 예시
  - DocumentRoot 디렉터리의 설정이 "*Indexes*", "*FollowSymLinks*" 옵션을 가진 경우,

```
〈Directory "/usr/local/www"〉
    Options Indexes FollowSymLinks
〈/Directory〉
```

  - DirectoryIndex에 정의된 초기 파일(index.html)이 존재하지 않을 경우, 디렉터리 내의 파일 목록을 리스트 업
  - FollowSymLinks로 인해 시스템 루트 디렉터리(/)에 Symbolic Link된 system.html 파일 (ln –s system.html /)을 열었을 경우, DocumentRoot 디렉터리 상위의 정보 조회 가능



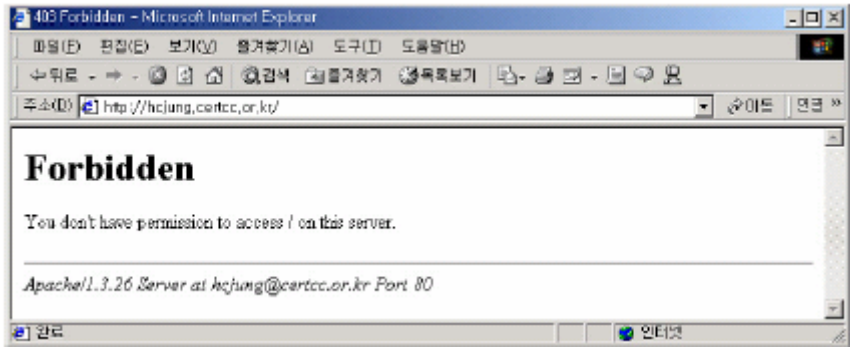


❖ Apache 설정파일 관리 [4]

- 보안관리가 올바른 설정파일(http.conf) 예시
  - "*Indexes*" 옵션과 "*FollowSymLinks*" 옵션 제거
  - "*IncludesNoExec*" 옵션 추가

```
〈Directory "/usr/local/www" 〉
    Options IncludesNoExec
〈/Directory〉
```

- DirectoryIndex에 정의된 초기 파일(index.html)이 존재하지 않을 경우, 디렉터리 내의 파일 목록을 보여주는 것이 아니라, 오류 창을 띄움



❖ Apache 설정파일 관리 [5]

▪ 웹 서버 응답 메시지 헤더 정보 숨기기

- 웹 서버 헤더 정보: 클라이언트가 웹 서버에 접속했을 때, 웹 서버에서 내보내는 응답 메시지의 헤더
- 웹 서버 버전, 배너, 구동되고 있는 응용 프로그램의 취약성을 식별하는데 이용

```
[root@hcjung conf]# telnet xxx.xxx.xxx.xxx 80
Trying xxx.xxx.xxx.xxx...
Connected to xxx.xxx.xxx.xxx.
Escape character is '^]'.
GET / HTTP/1.1

HTTP/1.1 400 Bad Request
Date: Tue, 15 Oct 2002 11:25:10 GMT
Server: Apache/1.3.19 (Unix) PHP/4.0.4pl1
```

- 설정파일(http.conf) 변경

"*ServerToken*" 지시자(directive)를 수정함으로써 헤더에 의해 전송되는 정보 변경

키워드	제공하는 정보	예
Prod[uctOnly]	웹서버 종류	Server: Apache
Min[imal]	Prod 키워드 제공 정보 + 웹서버 버전	Server: Apache/1.3.0
OS	Min 키워드 제공 정보 + 운영체제	Server: Apache/1.3.0 (Unix)
Full	OS 키워드 제공 정보 + 설치된 모듈(응용프로그램) 정보	Server: Apache/1.3.0 (Unix) PHP/3.0 MyMod/1.2

❖ Apache 웹 서버 사용자 인증 관리 [1]

- 기본 사용자 인증 (Basic Authentication)
  - Apache에서 제공되는 *htpasswd*를 이용하여 사용자 계정을 생성하고 인증하는 방법
  - 클라이언트에서 서버로 전송되는 도중 패스워드는 평문이므로 노출이 쉬움
  - 패스워드는 암호화되어 저장
- 다이제스트 사용자 인증 (Digest Authentication)
  - 기본 사용자 인증 방법과 동일
  - 클라이언트에서 서버로 전송되는 *패스워드를 MD5 해쉬하여 전송 [데이터는 평문]*
- 애플리케이션에서의 인증
  - 전자서명 인증, 생체 인증, 데이터베이스 적용

❖ Apache 웹 서버 사용자 인증 관리 [2]

▪ 기본 사용자 인증 (Basic Authentication)

❶패스워드 파일 생성

- htpasswd 명령을 이용하여 패스워드 파일 생성 [첫 생성시에만 –c 옵션 사용]

```
사용법: htpasswd [-cmdps] passwordfile username
```

```
[root@hcjung bin]# ./htpasswd -c /usr/local/apache/passwords hcjung
New password:
Re-type new password:
Adding password for user hcjung
```

- 웹 서버 자체가 읽을 수 있는 최소한의 권한 만을 부여 [Nobody 사용자/그룹]

```
[root@hcjung bin]# chown root.nobody /usr/local/apache/passwords
[root@hcjung bin]# chmod 640 /usr/local/apache/passwords
```

❖ Apache 웹 서버 사용자 인증 관리 [3]

▪ 기본 사용자 인증 (Basic Authentication)

❷ 패스워드 파일을 사용 가능하도록 환경 설정

- 각 디렉터리별로 사용자 인증
- 설정파일(http.conf) 변경

"*AllowOverride*" 지시자(directive)를 "*None*"에서 "*AuthConfig*"로 설정

```
<Directory "/usr/local/www">
  AllowOverride AuthConfig
</Directory>
```

- 사용자 인증이 필요한 디렉터리에 다음의 지시자(directive)들이 포함된 .htaccess 파일 생성

지 시 자	설 명
AuthType	인증 형태(Basic 또는 Digest)
AuthName	인증 영역(웹 브라우저의 인증창에 표시됨)
AuthUserFile	사용자 패스워드 파일의 위치
AuthGroupFile	그룹 파일의 위치(옵션)
Require	접근을 허용할 사용자 또는 그룹 정의 ex) Require user userid [userid] ... Require group group-name [group-name] ... Require valid-user

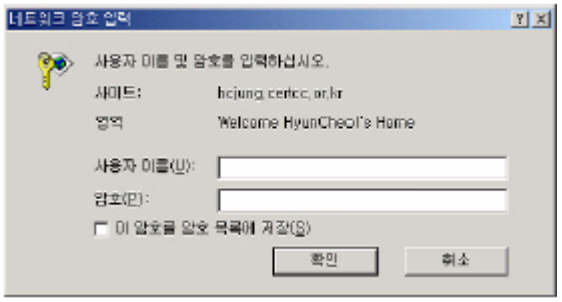
❖ Apache 웹 서버 사용자 인증 관리 [4]

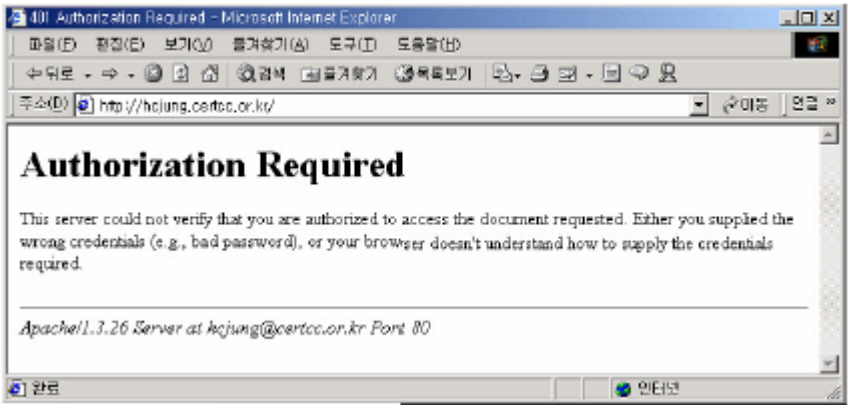
- 기본 사용자 인증 (Basic Authentication)

❸ 패스워드 파일에 등록된 hcjung와 webmaster만이 웹 서버에 접속할 수 있도록 필요한 설정

```
[root@hcjung /root]# cd /usr/local/www
[root@hcjung www]# vi .htaccess
AuthType Basic
AuthName "Welcome HyunCheol's Home"
AuthUserFile /usr/local/apache/passwords
Require user hcjung webmaste
```

- 정상적으로 사용자 인증 설정 완료 시, 다음과 같은 인증 창 생성
- 인증 실패 시, 다음과 같은 경고 창 생성





❖ SSL 인증서 또는 웹 암호화 솔루션 적용 관리

- Apache에서는 mod_ssl을 이용하여 암호화 지원
- 지원 인증서 규격
  - OpenSSL을 통해 자체적으로 생성한 인증서
  - 유료 공인 인증서 (NPKI/GPKI)

❖ 보안 패치 관리

- Apache 버전별 취약점 확인
  - ApacheWeek, http://www.apacheweek.com/security/
- 주기적인 취약점 패치
  - http://www.apache.org/dist/httpd/patches/

❖ 설정 파일 및 데이터 백업 관리

- Apache 각종 환경설정 파일
- Apache 설치 과정에서 사용된 Install 파일
  - Rebuild 시간 단축 효과
- 사용자 프로그램 소스
  - PHP, JSP, CGI 등
- 웹 서비스 연관 데이터베이스

❖ 로그 설정 및 분석 관리 [1]

- Apache 로그 파일
  - *에러 로그 (error log)*: Apache 서버의 에러 정보 기록
  - *액세스 로그 (access log)*: Apache 서버가 처리하는 모든 요청에 대한 기록
  - "*ErrorLog*" 및 "*CustomLog*" 지시자(directive)를 통해 로그 파일의 위치 설정 (http.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
```

❖ 로그 설정 및 분석 관리 [2]

- 에러 로그 (error log) 파일
  - 파일 포맷은 비교적 자유로운 형식이나, 다음과 같은 내용 포함

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

1. 메시지의 날짜와 시간
2. 에러의 위험도
3. 에러를 발생시킨 클라이언트 주소
4. 에러 메시지의 내용
   (클라이언트가 요청한 문서를 파일시스템 경로로 표현)

  - "*LogLevel*" 지시자(directive)를 통해 에러의 위험도 수준 설정 (http.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
```

❖ 로그 설정 및 분석 관리 [3]

- 액세스 로그 (access log) 파일
  - "*LogFormat*" 지시자(directive)를 통해 파일 포맷 선택
    - *Common Log Format*
    - *Combined Log Format*

- Common Log Format
  - 웹 서버에서 공통으로 사용하는 로그 포맷
  - 로그 분석 프로그램이 읽을 수 있는 포맷
  - 설정파일(http.conf) 변경

```
LogFormat "%h %l %u %t ₩"%r₩" %>s %b" common
CustomLog logs/access_log common
```

  - "LogFormat" 지시자(directive)는 하나의 포맷 스트링을 정의하고 common이라는 닉네임을 가짐
  - "CustomLog" 지시자(directive)가 로그가 저장될 파일의 위치와 이름, 로그 포맷을 정의

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

1. 클라이언트 IP 주소 (%h): 서버의 성능저하 (off 권고)
2. 클라이언트 신분 (%l): 서버의 성능저하, 신뢰도 낮음 (off 권고)
3. HTTP 인증 받은 사용자 ID (%u): 인증 받지 못한 경우 부정확, 인증을 요구하지 않는 경우 '-' 표시
4. 서버가 요청 처리를 끝낸 시간 (%t): [일/월/년:시:분:초 지역]
5. 클라이언트 요청 내용 (\ %r\): 사용한 메쏘드, 요청한 자원, 사용한 프로토콜 표시
6. 상태코드 (%>s): 2XX(성공), 3XX(redirection), 4XX(클라이언트에 의한 에러), 5XX(서버에 의한 에러)
7. 클라이언트에 전송된 컨텐트 크기 (%b): 없으면 '-' 표시

❖ 로그 설정 및 분석 관리 [4]

- Combined Log Format
  - 설정파일(http.conf) 변경

```
LogFormat "%h %l %u %t ₩"%r₩" %>s %b ₩"%{Referer}i₩" ₩"%{User-
agent}i₩"" combined
CustomLog log/acces_log combined
```

  - "LogFormat" 지시자(directive)는 하나의 포맷 스트링을 정의하고 combined이라는 닉네임을 가짐
  - "CustomLog" 지시자(directive)가 로그가 저장될 파일의 위치와 이름, 로그 포맷을 정의

```
172.16.5.100 - jun [08/Apr/2003:16:03:43 +0900] "GET /php HTTP/1.1" 301
313 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)"
```

1. 클라이언트가 요청한 자원이 include 되었거나 링크된 페이지: (\ %{Referer}I\)
2. 클라이언트 브라우저에 대한 정보: (\ %{User-agent}I\)

- 로그 파일의 보호
  - 일반 사용자는 로그 저장 디렉터리에 쓰기 권한이 없도록 설정
  - 클라이언트가 웹 서비스를 통해 로그 파일을 볼 수 없도록 설정