일 러 두 기

o 행정안전부(정부통합전산센터)는 사이버 침해를 방지하기 위해 24×365일 쉴틈 없는 보안관제와 주기적인 보안점검 등 정보보호 활동을 수행하고 있습니다.

o 하지만, 사이버 침해의 주요대상이 되는 홈페이지의 웹 프로토콜은 기본적으로 누구에게나 개방되어 있기 때문에, 전통적인 침입차단 시스템을 이용한 방어가 용이하지 않으며, 이러한 특성을 이용한 새로운 위협이 지속적으로 발견되고 있습니다.

o 이러한 이유로, 행정안전부(정부통합전산센터)에서는 홈페이지에 대한 근본적인 보안 강화를 위해, 국가정보원, 한국인터넷진흥원, OWASP[1]등 국내외 전문기관에서 정의한 홈페이지 개발 보안 고려사항을 분류, 『웹 어플리케이션 개발 보안 가이드 2010』을 발간하게 되었습니다.

o 본 가이드에서는 12개 보안 사항에 대한 상세 설명과 함께 취약여부를 점검할 수 있는 방법 및 위험도, 보호대책을 제공합니다.

---

1) **OWASP : The Open Web Application Security Project (Http://www.owasp.org)**

웹 어플리케이션 개발 보안 항목 요약

o 국가정보원의 「홈페이지 보안관리 매뉴얼.」에 명시된 홈페이지 8대 취약점과 한국인터넷진흥원의 「홈페이지 개발 보안가이드」의 10대 취약점, OWASP에서 발표한 「10대 가장 심각한 웹 어플리케이션 보안 취약점」에 명시된 10가지 취약점 등 28가지 취약점을 정부통합전산센터의 환경을 고려하여, 웹 어플리케이션 개발 시 고려해야 할 12가지 점검항목으로 재분류하였습니다.

점 검 항 목	국가정보원	KISA	OWASP
① 스크립트 삽입(XSS)	O	O	O
② 스크립트 요청 참조(CSRF)			O
③ 악성 파일 실행	O		O
④ SQL 구문 삽입	O	O	O
⑤ URL/파라메터 조작		O	O
⑥ 파일 업로드	O	O	O
⑦ 파일 다운로드	O	O	O
⑧ URL강제접속/인증우회		O	O
⑨ 서비스 메소드 설정	O		
⑩ 에러처리 및 기타 정보 노출		O	O
⑪ ID/PW 관리	O		
⑫ 환경설정 보안 고려사항	O	O	O

※ 위 항목은 홈페이지 개발단계에서 고려하여야 할 보안점검항목으로 보안진단 항목과는 다를 수 있음.

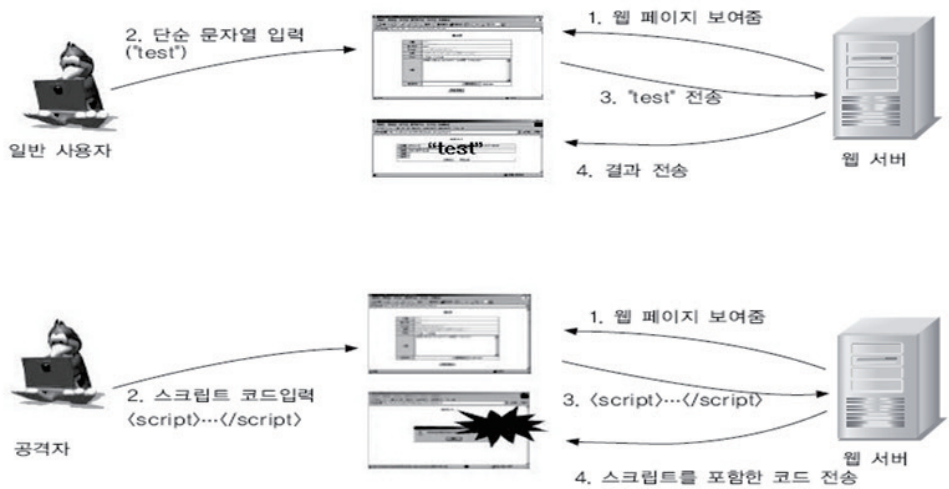
목 차

1. 스크립트 삽입(XSS)

☑ 개 요

o 스크립트 삽입(XSS:Cross Site Script)은 악성 스크립트를 홈페이지 게시판, 전자메일 등을 통해 유포하여, 사용자가 해당 게시물 또는 메일을 클릭하였을 때 악성 스크립트가 실행되는 것이며,

o 공격자는 스크립트 삽입을 이용하여 사용자의 개인정보, 로그인정보, 내부자료 등을 탈취하여 2차적인 공격을 수행한다.



[그림 1] 스크립트 삽입(XSS)

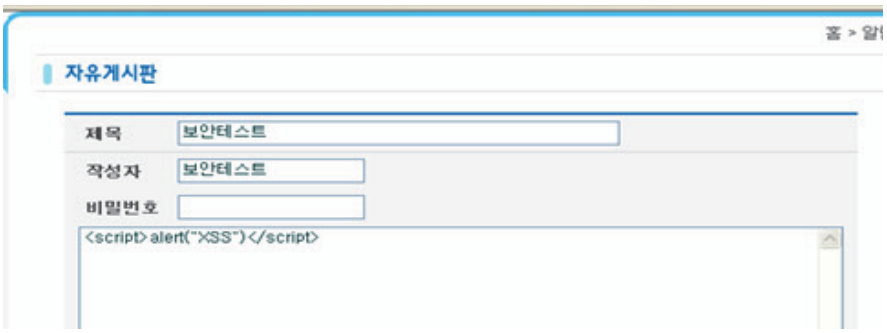
☑ 점검 예시

o 홈페이지 내 게시판에 글쓰기 시 작성자, 제목, 본문 등에 XSS 스크립트를 입력

예시) 1. <script>alert("XSS")</script>

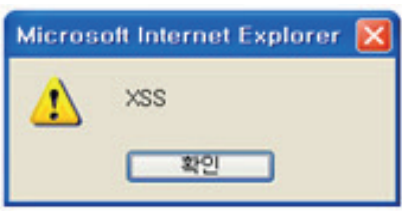
2. <sc<script>ript>alert("XSS")</sc</script>ript>

3. %3Cscript%3Ealert%28%22XSS%22%29%3C/script%3E



[그림 2] 게시물에 스크립트 삽입

o 취약 시스템의 경우 게시물 보기를 실행하면, 아래와 같이 팝업이 발생



[그림 3] XSS 취약시스템

☑ 공격대상

o 공격대상 언어 : 다음과 같은 스크립트나 언어가 스크립트 삽입 공격 대상이 된다.

JavaScript, VBScript, ActiveX, HTML, Flash

o 공격 시도 위치 : 다음과 같은 입력 부분에서 스크립트 삽입 공격을 시도한다.

- ◆ 게시판 글쓰기 화면(제목, 글 내용 등 데이터를 입력할 수 있는 모든 필드)
- ◆ 게시판 글 보기 화면에서 댓글 달기
- ◆ 쿠키 값
- ◆ 데이터베이스 질의문

☑ 피해유형

o 페이지 변조 : 이미지나 사운드를 삽입할 수 있다.
o 쿠키 변형 : 쿠키 자체를 변형시켜서 쿠키에 악의적인 코드를 넣어둘 수 있다.
o 쿠키 유출 : 사용자 쿠키 값을 획득하여, 인증우회에 사용한다.
o 개인정보 유출 : 웹 페이지 내의 패스워드나 신용 카드 번호와 같은 민감한 정보를 유출시킬 수 있다.
o 악성 프로그램 다운로드 : 스크립트 내에 악성 프로그램을 다운로드 받는 사이트를 링크시키며, 사용자가 모르는 사이 악성프로그램이 PC에 설치된다.

☑ 보호대책

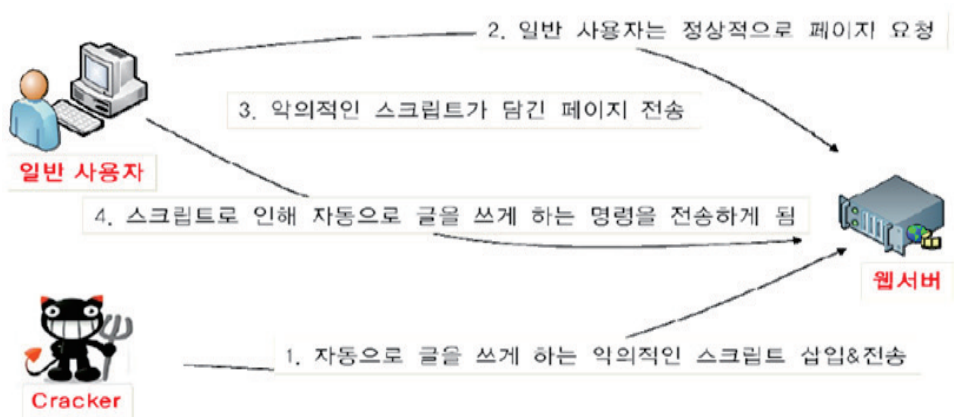
o 스크립트 삽입 공격에 대한 근본적인 대응방법은 사용자로부터 입력받는 모든 값을 서버에서 검증 후 입력 받도록 하는 방식을 사용하는 것이다.
o 입력 값 검증의 방법은 사용자 입력으로 사용 가능한 문자를 정해놓고, 나머지 모든 문자를 필터링하는 방법을 사용하여야 한다.
  - 검증 대상 입력 값 : 스크립트 정의어(<script>, <object>, <applet>, <embed> 등)
  - 어플리케이션에서 사용자들의 모든 입력값을 점검, 허용된 데이터만 입력할 수 있도록 구현
  - 입력되는 특수문자는 Entity형태로 변경하여 저장.
o 이때, 검증은 반드시 서버에서 이루어지도록 개발하여야 한다.

2. 스크립트 요청 참조(CSRF)

☑ 개 요

o 스크립트 요청 참조(CSRF)는 공격자가 사용자의 Cookie 값이나 Session정보를 의도한 사이트로 보내거나 특정한 동작을 유발하는 스크립트를 글에 삽입하여 사용자가 게시물 등을 클릭할 경우 공격자가 원하는 동작이 실행되는 것이며,

o 일반적으로 악용되는 태그들은 <Script>, <OBJECT>, <APPLET>,<EMBED>, <IMG>, <FORM> 등이 있다.



[그림 4] 스크립트 요청 참조(CSRF)

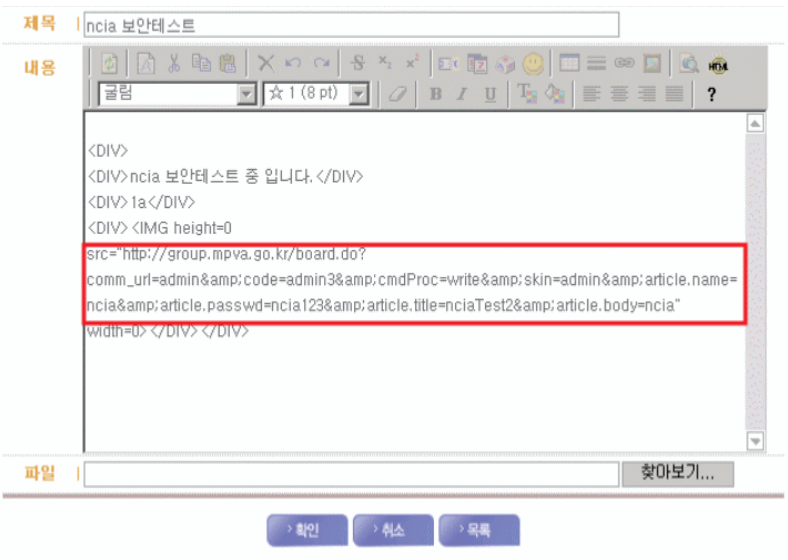
☑ 점검 방법

o 글쓰기 후 전송되는 URL과 파라미터 값을 복사한다.

- http://aa.co.kr/write_ok.php?cmdProc=write&name=test&passwd=test&title=test&body=test

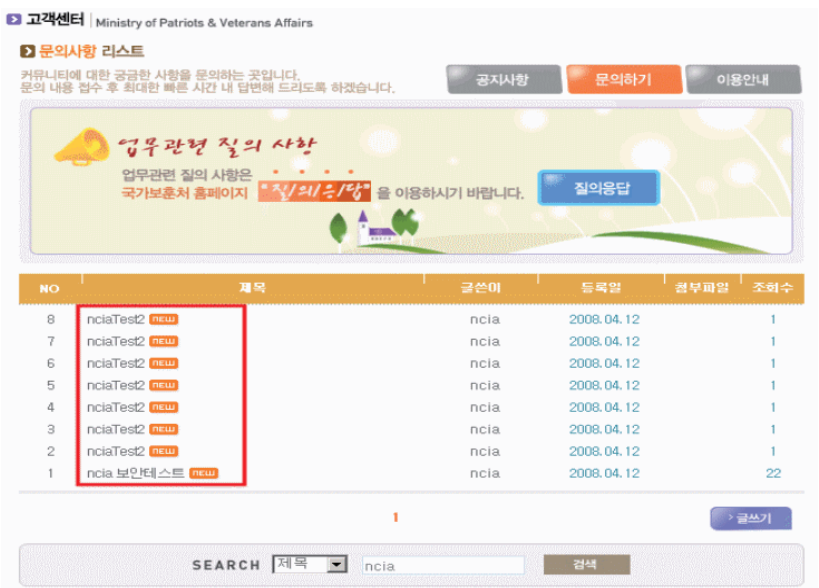
o 새로운 글을 작성하고, 글 내용 중에 <IMG> 태그를 사용하여, 아래와 같이 복사한 URL과 파라미터를 붙여 넣는다.

- <img width=0 height=0 src="http://aa.co.kr/write_ok.php?cmdProc=write&name=test&passwd=test&title=test&body=test">



[그림 5] <IMG> 태그 내 파라미터 값 삽입

o 등록된 게시물 열람 시 동일한 작업이 반복됨을 확인할 수 있다.



[그림 6] 동일 작업 반복 확인

☑ 공격대상

o 공격 대상이 되는 부분 : 글쓰기 시 HTML 태그가 허용된 게시판
o 공격 대상이 되는 태그 : <script>, <embed>, <object>, <applet>, <img> 등

☑ 피해유형

o 정보 노출 : 쿠키 또는 세션 정보 노출
o 동일 작업 반복 : 게시물 클릭 시 공격자가 원하는 동작 수행

☑ 보호대책

o CSRF를 예방할 수 있는 최선의 방안은 모든 입력 값을 상세히 검증하는 것으로 헤더, 쿠키, 질의 문, 폼 필드, 숨겨진 필드등과 같은 모든 파라미터들을 엄격한 규칙에 의해서 검증하여 HTML을 사용할 경우 태그 내에 html, ?, & 등이 포함되지 않도록 하여야 한다.
o 이때, 검증은 반드시 서버에서 이루어지도록 개발하여야 한다.

3. 악성 파일 실행

☑ 개 요

o URL 이나 파일시스템 참조 등 외부객체 참조를 사용하는 어플리케이션에서 입력파일 또는 입력 외부객체를 검증하지 않을 때 발생되며, PHP파일 연결, OS명령어 삽입, 제한되지 않은 파일 업로드 등 다양한 유형이 존재한다.

o 악성 파일 실행은 원격코드 실행, 원격 루트킷 설치와 시스템 손상 등을 야기한다.

☑ 점검 방법

o 테스트 서버에 임의의 파일을 저장한다.(파일명 : testfile.txt)

o URL중 외부객체를 참조하는 URL을 확인한다

- http://www.abc.com/activities_view.php?id=104&page=1&base_dir=/test

o 외부 객체를 참조하는 URL부분을 테스트 서버의 파일을 참조하도록 수정한다.

- http://www.abc.com/activities_view.php?id=104&page=1&base_dir=http://테스터서버/testfile.txt

o 테스트 서버에 저장해 놓은 파일의 내용이 브라우져에 보이는 경우 취약시스템

☑ 공격대상

o 홈페이지 소스 중 URL파일명 함수나 로컬 파일을 포함시기키 위해 사용자에게 파일명 선택을 허용하도록 개발된 코드

o 특히, PHP의 경우 아래와 같은 함수가 공격의 대상이 된다

include(), include_once(), require(), requier_once(), fopen(), imagecreatefromXXX(), file(), file_get_contents(), copy(), delete(), unlink(), upload_tmp_dir() $_FILES, move_uploaded_file()

☑ 피해유형

o 원격 루트킷 설치와 전체 시스템 손상 : 원격시스템에 악성 코드 등을 저장 후 실행시키게 되면 악성코드가 동작하면서 로컬시스템에 해킹 프로그램 설치 등이 가능하다. 공격자는 이를 통해 전체 시스템을 파괴할 수 있으며, 원격 코드의 종류에 따라 중요정보 노출 등의 위험이 존재한다

☑ 보호대책

o 다른 객체를 참고 할 때에는 참고 대상이 되는 객체이름 부분에 "URL형태" 또는 "정보시스템 외부의 파일"이 입력되지 않도록 구현하고,

o 특히 PHP언어로 웹 어플리케이션을 개발 할 때에는

- Allow_rul_fopen, allow_url_include 를 비활성화
- register_global 비활성화
- 사용자 입력에 아래의 함수를 포함한 입력 값 금지

include(), include_once(), require(), requier_once(), fopen(), imagecreatefromXXX(), file(), file_get_contents(), copy(), delete(), unlink(), upload_tmp_dir() $_FILES, move_uploaded_file()

4. SQL 구문 삽입

☑ 개 요

o SQL 구문 삽입(SQL Injection)은 URL의 파라미터 값 등의 전송되는 문자열에 대해 웹서버에서 유효성을 검증하지 않아, SQL 구문이 직접 DB서버로 전송되어 실행되는 것이며,

o 공격자는 SQL 구문 삽입을 이용하여 로그인 인증우회, 홈페이지 변조, 내부자료 유출 등을 시도한다

☑ 점검 방법

o 정상 URL/파라미터에 "'"을 추가하여 "SQL 오류 메시지"가 나타나면 공격에 취약함.

- 정상URL : http://xxx.com/view.jsp?srch_keyword=asdf
- URL조작 : http://xxx.com/view.jsp?srch_keyword=asdf**' or 2=2--**

o 또는, 사용자 로그인창에 아래와 같은 SQL구문을 입력하였을 때 로그인이 된다면 공격에 취약함

- ' or 1=1 -- (Oracle, Sybase), ' or 1=1#(MS-SQL), ' or 1=1;(MYSQL)

[그림 7] 로그인 창에 SQL 구문 입력

[그림 8] 로그인 성공

☑ 공격대상

o 대상 공격 삽입 부분 : 다음과 같은 부분이 공격 대상이 된다.

- ◆ 사용자 로그인 입력 폼
- ◆ 웹 URL 파라미터
- ◆ 쿠키 값

☑ 피해유형

o 데이터베이스 구조노출 : 의도적으로 "오류"가 발생되는 SQL 쿼리문을 삽입하여 데이터베이스 구조 파악

o 데이터베이스 자료 유출 : 주요 자료 외부유출

o 데이터베이스 데이터 변조 : SQL 쿼리문 조작을 통하여 홈페이지 내용을 변조.

o 인증우회 : 타 사용자로 로그인 성공

☑ 보호대책

o 데이터베이스와 연동하는 스크립트의 모든 파라미터들을 점검하여 사용자의 입력 값에 SQL 쿼리문이 삽입되지 않도록 특수문자(', ", ₩, ;, :, %, Space, --, +, <, >, (,), #, & 등)를 필터링 한다.

o 입력문자열 대한 길이를 제한한다.

o 데이터베이스와 연동하는 스크립트의 모든 파라미터들을 점검하여 사용자의 입력 값에 SQL 구문으로 사용되는 문자열(@variable, @@variable, print, set, or, union, and, insert, openrowset 등)을 필터링 한다.

o 데이터베이스의 에러 메시지를 사용자에게 보여주지 않도록 수정한다.

o 웹 어플리케이션이 사용하는 데이터베이스의 사용자 권한을 제한한다.

o php.ini 설정 중 magic_quotes_gpc 값을 On으로 설정한다.

- magic_quotes_gpc 옵션의 역할은 GPC(Get, Post, Cookie)를 통해 넘어오는 문자열 중에서 '(sing-quote)와 "(double-quote), ₩(backslash), NULL 값의 앞에 자동으로 백슬래쉬 문자를 붙여주는 기능을 한다.

```
; Magic quotes
; Magic quotes for incoming GET/POST/Cookie data.

magic_quotes_gpc = On ; Off에서 On으로 변경

; Magic quotes for runtime-generated data, e.g. data from SQL, form exec(), etc.

magic_quotes_runtime = Off

; Use Sybase-style magic quotes(escape ' with '' instead of \').

magic_quotes_sybase = Off
```

5. URL/파라메터 조작

☑ 개 요

o 공격자는 전송되는 URL 또는 URL의 파라미터를 조작하여 전송함으로써 웹서버로 하여금 공격자가 원하는 행위를 하도록 시도한다. 이때, 취약한 시스템에서는 인증우회, 관리자 권환 획득, 인가되지 않은 게시판에 글쓰기 등 인가되지 않은 시도가 성공한다.

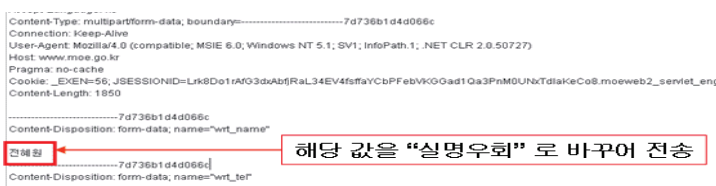
☑ 점검 방법

o 웹 프락시 도구를 이용하여 파라미터를 조작, 실명인증을 후회 시도

1. 글쓰기 전 실명인증 시도

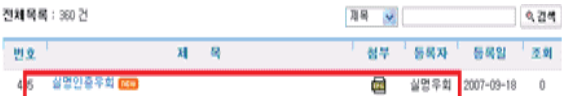
[그림 9] 실명인증 화면

2. 정상적인 실명인증 후 글쓰기 시도
3. 글쓰기 후 전송되는 패킷을 가로채어 "글쓴이"이름을 변경



[그림 10] 글쓴이 이름 변경

4. 글쓰기 완료 시 변경시킨 이름으로 글쓰기가 성공됨.



[그림 11] 실명인증 우회 성공

o URL의 파라미터 값을 조작하여 글쓰기, 수정 등이 가능한지 확인

1. 글쓰기 : board.php?mode=list에서 mode값을 write, add, insert 등으로 변경
2. 글 수정 : board.php?mode=view&idx=1에서 mode값을 update, modify, edit 등으로 변경
3. 글 삭제 : board.php?mode=delete&idx=1에서 idx값을 다른 게시물 번호로 변경

☑ 공격대상

o 대상 공격 삽입 부분 : 다음과 같은 부분이 공격의 대상이 된다.
- ◆ 웹 페이지에서 게시물을 조회하는 페이지
- ◆ 웹 게시판의 글쓰기 페이지

☑ 피해유형

o 인증 우회 : 정상적인 로그인이 아닌 파라미터 조작을 통하여 인증 우회가 가능하다. 이를 통해 공격자는 일반 사용자에게는 글쓰기 권한이 없는, "공지사항"등의 게시판에 게시물을 게시하는 등 홈페이지 조작이 가능하다.

o 게시물 조작 : 다른 사용자가 작성한 글을 수정 및 삭제 할 수 있다.

☑ 보호대책

o 글쓰기, 글 수정, 글 삭제 URL등 사용자의 조작에 의해 변경이 일어날 수 있는 페이지에 대해서는 정상적인 사용자인지 여부를 확인하는 인증절차를 삽입한다.

o 파라미터 및 URL을 입력 받을 때, 서버에서 직전에 응답한 정보와 입력되는 정보가 다른 부분이 존재하는지 여부를 체크하여, 실명인증 우회 등 인증우회를 방지한다.

6. 파일 업로드

☑ 개 요

o 파일 첨부를 할 수 있는 게시판에 일반적으로 허용된 파일(이미지파일, 워드파일 등) 이외에 악의적인 스크립트가 포함된 소스파일(.jsp, .php, .asp 등)을 첨부할 수 있게 되면,

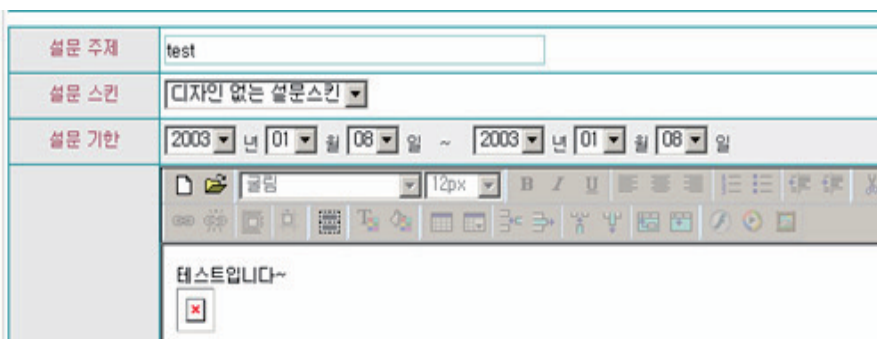
o 공격자는 악성 스크립트 등재 후, 서버 상에서 스크립트를 실행시켜 쉘 획득, 서버변조 등 웹서버를 장악할 수 있다.

☑ 점검 방법

o 사전준비

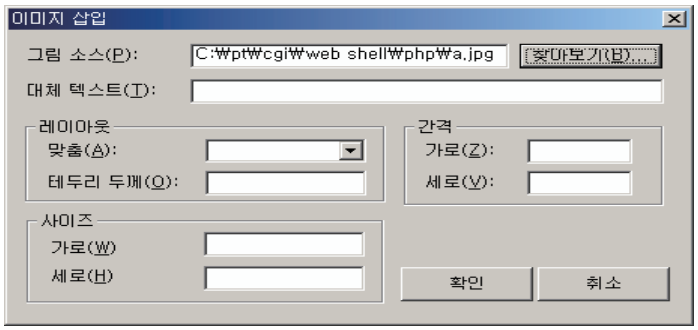
- 악성 스크립트 파일 준비 : a.php
- 악성 스크립트 파일을 복사하여 확장자를 이미지파일로 변경 : a.jpg
- 웹 프락시 도구 실행 : Paros 등

o 게시판 글쓰기 실행



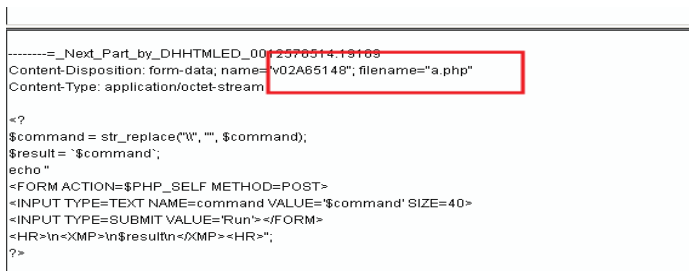
[그림 12] 게시물 작성

o 첨부파일 또는 글 편집기의 파일 업로드 버튼을 이용하여 파일 첨부 시도(a.jpg)



[그림 13] 첨부파일 첨부 화면

o 전송되는 패킷을 웹 프락시 도구를 이용하여 가로챈 후 파라미터 부분의 "a.jpg"를 "a.php"로 변경 후 전송



[그림 14] 파라미터 변조

o 게시물 작성 완료 후 첨부파일의 등록정보 등을 이용하여 첨부파일의 경로 확인

o 악성 스크립트 실행

[그림 15] 스크립트 실행

☑ 공격대상

o 대상 공격 부분 : 다음과 같은 기능이 있는 웹 게시판이 파일 업로드 공격의 대상이 된다.

- ◆ 파일 첨부 기능이 있는 웹 게시판
- ◆ 게시판 Editor의 이미지 첨부 기능이 있는 웹 게시판

☑ 피해유형

o 서버 원격 제어 : 쉘 프로그램을 통하여 웹 서버를 원격 제어할 수 있다.

o 홈페이지 위/변조 : 업로드 된 쉘 프로그램 실행을 통한 홈페이지 위/변조가 가능하다.

☑ 보호대책

o 첨부파일이 저장되는 Upload 디렉토리는 실행 권한을 제거하여 운영한다.

- IIS 보안 설정

  설정 ⇨ 제어판 ⇨ 관리도구 ⇨ 인터넷 서비스 관리자 선택 ⇨ 해당 Upload 폴더에 오른쪽 클릭을 하고 등록 정보 ⇨ 디렉토리 ⇨ 실행권한을 "없음"으로 설정

- Apache 설정 : 설정 후 Apache 데몬 Restart 해야 한다.

  Apache 설정 파일인 httpd.conf의 해당 디렉토리에 대한 문서 타입을 컨트롤하기 위해 Directory 세션의 AllowOverride 지시자에서 FileInfo 또는 All 추가

```
<Directory "/usr/local/apache">
AllowOverride FileInfo(또는 All) ...
...
</Directory>
```

- 파일 Upload 디렉토리에 .htaccess 파일을 만들고 다음과 같이 AddType 지시자를 이용, 현재 서버에서 운영되는 Server Side Script 확장자를 text/html로 MIME Type을 재조정하여 Upload된 파일이 실행되지 않도록 설정하거나 FileMatch 지시자를 이용하여 *.ph, *.inc, *.lib 등의 Server Side Script 파일에 대해서 직접 URL 호출을 금지시킨다.

```
<.htaccess>
<FileMatch "₩.(ph|inc|lib)">
Order allow, deny
Deny from all
</FileMatch>
AddType text/html .html .htm .php .php3 .php4 .phtml .phps .in .cgi .pl .shtml
.jsp
```

o 첨부파일의 확장자 필터링 처리

- 웹서버의 서비스 환경을 고려하여 office 문서, text, 이미지 등 업로드를 허용할 파일을 지정 한 후 지정된 확장자 이외의 파일이 업로드 되지 않도록 제한한다. 이 때, 확장자 점검은 반드시 서버 단에서 점검하여야 한다.

7. 파일 다운로드

☑ 개 요

o 웹 어플리케이션에서 상대경로를 사용할 수 있도록 설정되어 있는 경우, 상대경로 표시 문자열인 "../"를 통해 허가되지 않은 상위경로로 이동하여 시스템 주요 파일, 소스코드 등 중요자료의 열람이 가능하며,

o 공격자는 이러한 방법으로 /etc/passwd, /etc/shadow, /etc/host 등 시스템 정보가 포함되어 있는 주요파일 및 웹 소스파일을 다운로드, 시스템 계정 및 패스워드, DB접속정보 등을 획득하여 시스템 침투, 내부자료 유출 등 2차적인 공격에 악용할 수 있다.

☑ 점검 방법

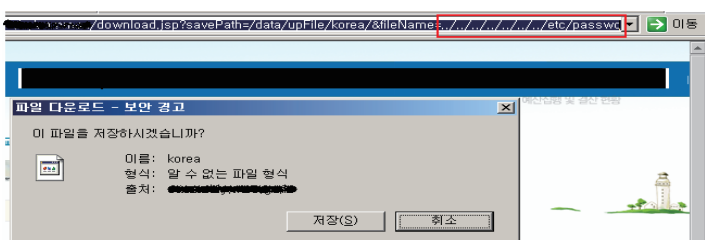
o 게시판의 파일다운로드 기능이 있는 게시물의 이름을 변경하여 시스템 파일 열람 시도

o 정상적인 게시판 파일 다운로드 URL확인

- http://www.kkk.com/download.jsp?pFilename =07년매출예산 결산 보고서
- http://www.kkk.com/download.jsp?savePath=/upload/ &fileName=보고서&upfileName=보고서

o 정상적인 URL을 "../"이용하여 조작

- http://www.kkk.com/download.jsp?pFilename= ./../../../etc/./../passwd
- http://www.kkk.com/download.jsp?savePath= /etc/&fileName=passwd&upfileName=passwd



[그림 16] 시스템 파일 다운로드 화면

☑ 공격대상

o 대상 공격 부분 : 다음과 같은 부분이 공격의 대상이 된다.

- ◆ 게시판의 첨부파일 다운로드 기능
- ◆ 웹 URL

☑ 피해유형

o 주요파일 노출 : 패스워드파일 등 시스템 주요파일이 유출된다.

o 소스파일 노출 : 웹 어플리케이션 소스파일이 유출되어 공격자는 DB접속정보 등 내부 접속 정보를 획득 가능하다.

☑ 보호대책

o 파일 다운로드 시 파일명을 직접 URL에서 사용하거나 입력받지 않도록 하며 게시판 이름과 게시물 번호를 이용하여 서버 측에서 데이터베이스 재검색을 통하여 해당 파일을 다운로드 할 수 있도록 하여야 한다.

o 다운로드 위치는 지정된 데이터 저장소를 지정하여 사용하고, 데이터 저장소 상위 디렉토리로 이동되지 않도록 설정한다.

o PHP를 사용하는 경우 php.ini에서 magic_quotes_gpc를 On으로 설정하여 ".₩./"와 같은 문자에 대해 대응도 가능하도록 한다.

8. URL강제접속/인증우회

☑ 개 요

o 사용자 권한관리가 정상적으로 이루어지지 않는 홈페이지의 관리자 또는 사용자 인증 후 접속되는 페이지의 URL을 주소창에 직접 입력하거나, 쿠키를 조작하는 방법이며

o 공격자는 이러한 방법으로 관리자 메뉴페이지에 로그인 과정 없이 접속하여 관리자 페이지를 조작, 회원정보 열람 등 민감한 정보를 획득할 수 있고, 공지사항 등 인증이 필요한 글쓰기 게시판에 악의적인 게시물을 게시할 수 있다.

☑ 점검 방법

o 로그인을 하지 않고, 로그인 이후 접속되는 페이지의 URL을 입력하여 페이지가 정상적으로 표시되는지를 확인한다.

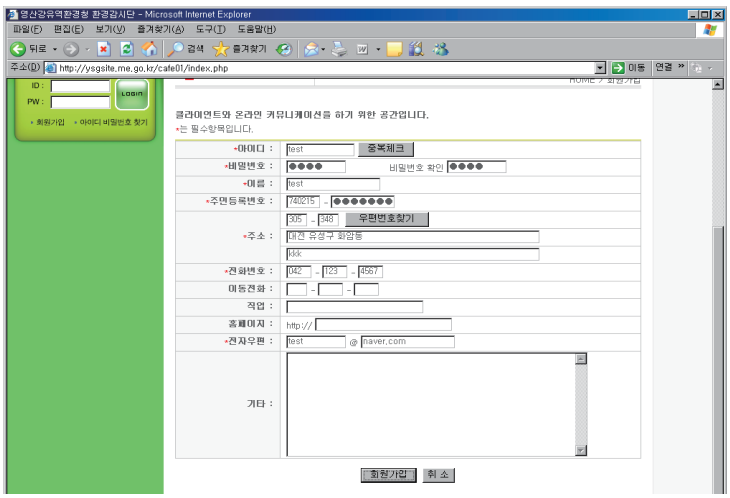
- 로그인 페이지 확인 : http://www.kkk.com/jsp/admin/admin_login.jsp

- 로그인 이후의 페이지를 유추하여 강제접속 : http://www.kkk.com/jsp/admin/admin.jsp

[그림 17] 관리자 페이지 강제접속

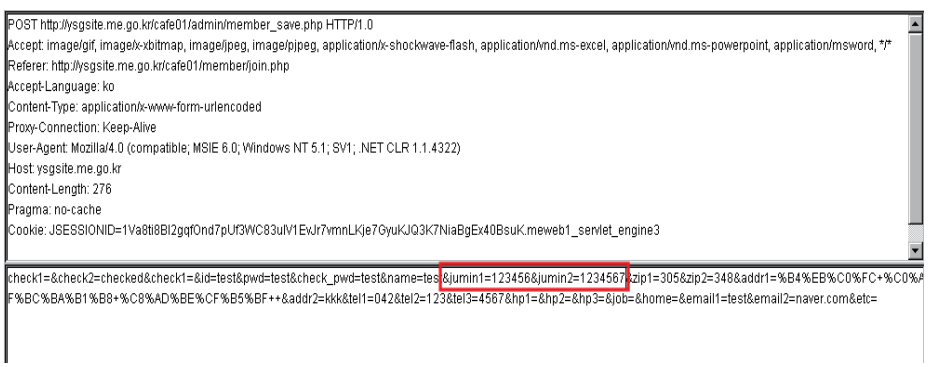
o 회원가입 시 실명 인증 또는 주민등록번호 유효성 검증을 우회하여 가명의 올바르지 않은 주민등록번호로 가입 가능한지 확인한다.

- 실명 인증 확인 후 보여지는 회원 가입 폼 화면이 직접 URL에 입력하여 보이는지 확인한다



[그림 18] 회원가입화면

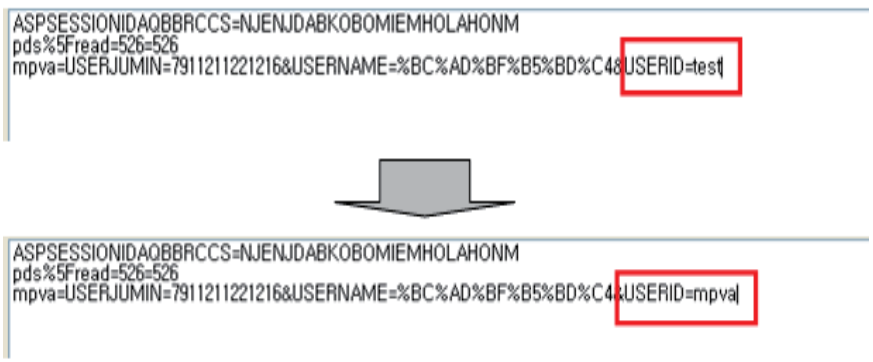
- 주민등록 번호 인증 우회 방법
  ① 회원가입 폼 화면을 열기 전 웹 프락시 툴을 이용하여 응답되는 패킷 중 주민등록번호 유효성 체크하는 자바 스크립트 부분을 삭제한다
  ② 회원가입 폼 화면에서 정상적인 주민등록번호 입력 후 전송되는 패킷을 웹 프락시 툴을 이용하여 주민등록번호 부분을 수정한다



[그림 19] 주민등록번호 조작 화면

o 세션의 쿠키 값이 평문(Plain text)으로 되어있는지 확인하고 중요 정보나 개인정보들이 포함되어 있지 않은지 확인하여 웹 프락시 도구를 사용하여 전송되는 쿠키 값 변조를 시도한다.

- 홈페이지에 자신의 계정 생성
- 로그인 시 웹 프락시 도구를 사용하여 전송되는 정보 중 쿠키 값에 개인 식별 정보가 포함되어 있는 지 확인
- 로그인 시 웹 프락시 도구를 사용하여, 쿠키 값을 변경



[그림 20] 쿠키 값 변경(웹 프락시 도구 사용)

- 다른 사용자의 정보 획득 및 수정이 가능하다.

[그림 21] 타 사용자의 정보 획득

☑ 공격대상

o 대상 공격 부분

- ◆ 관리자 메뉴 페이지
- ◆ 사용자 정보 페이지
- ◆ 회원 가입 페이지
- ◆ 사용자 정보(ID, PW, 권한 등)가 포함되어 평문으로 전송되는 쿠키 값

☑ 피해유형

o 회원 전용 페이지 노출 : 인증 처리를 하지 않는 회원 페이지에 강제 접속하여 인증 없이 회원 페이지를 열람할 수 있다.

o 관리자 권한 노출 : 인증 우회를 통하여 홈페이지 관리자 권한을 획득할 수 있다.

o 임의 사용자 등록 : 실명 인증 우회를 통하여 익명의 사용자 이름과 조작된 주민등록 번호로 회원 가입 가능하다.

o 시스템 명령 실행 : 웹 쉘이 업로드 된 경우 쿠키 조작을 통해 시스템 명령 실행이 가능하다.

o 게시판 변조 : 쿠키 값에 SQL 삽입 취약점을 이용하여 게시판 내용을 변조할 수 있다.

☑ 보호대책

o 회원전용 페이지나 관리자 페이지 등 모든 페이지에 접근 허용 금지 및 권한 설정을 하고, 인증을 거쳐서 접근할 수 있도록 한다. 이 때, 인증은 서버에서 실시하도록 한다.

o SSL과 같은 기술을 사용하여 로그인 트랜젝션 전체를 암호화 한다.

o 인증기능 사용시 쿠키를 이용하지 않는다.

9. 서비스 메소드 설정

☑ 개 요

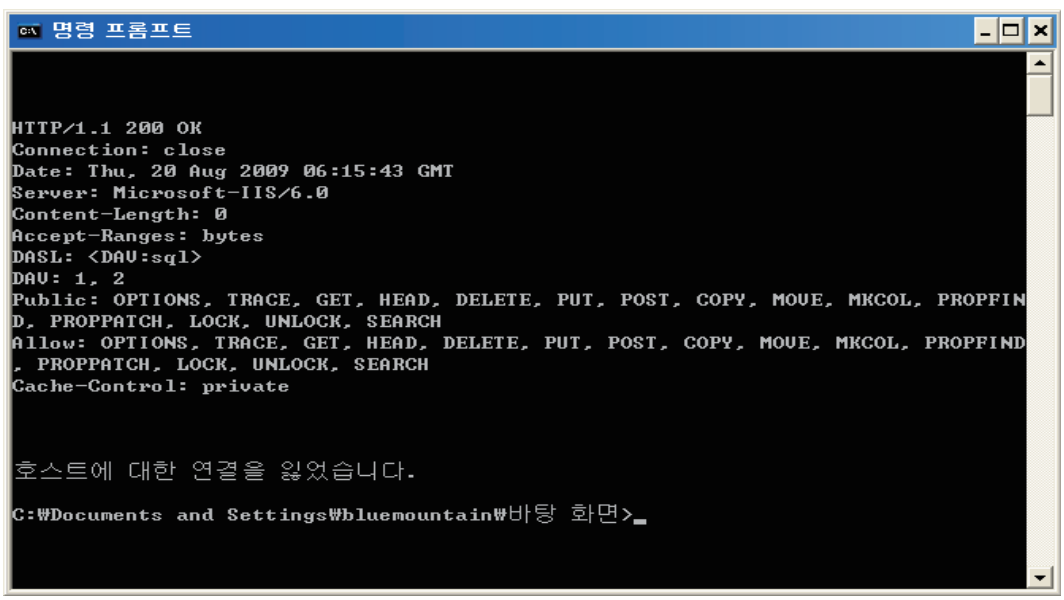
o Method는 웹 어플리케이션에서 기본적으로 제공하는 클라이언트와 통신하기 위한 도구로 GET, POST, PUT, MOVE, DELETE등 여러 가지 Method가 있으며 다양한 기능을 수행한다.

o 공격자는 웹서버에 허용되어있는 Method를 이용하여 파일업로드, 웹서버 파일삭제 등 웹서버를 인증 없이 조작할 수 있다.

☑ 점검 방법

o Command창에서 명령어로 Method 활성화 여부를 확인한다

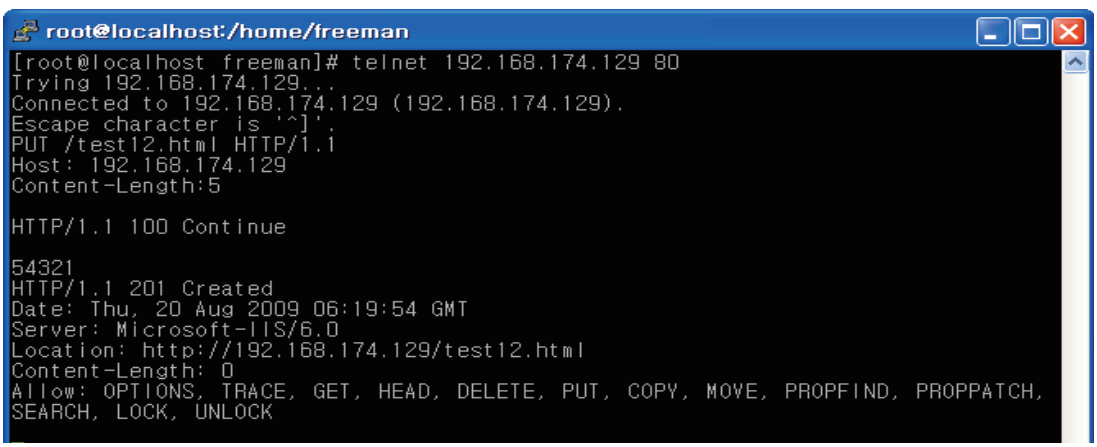
- telnet www.kkk.com 80으로 접속
- OPTIONS * HTTP/1.0



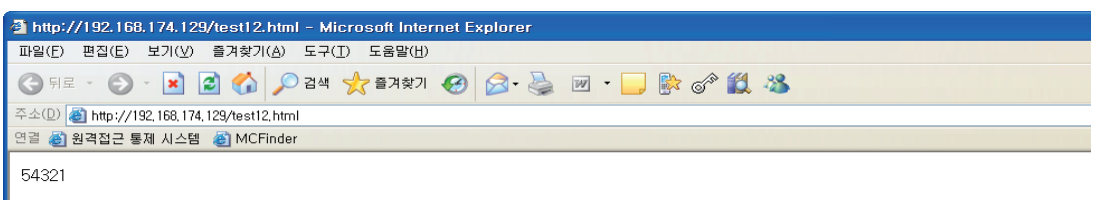
[그림 22] PUT Method 허용여부 확인

o PUT Method가 허용되어 있다 하더라도, 반드시 취약한 것은 아니며, 아래와 같은 테스트 방법을 이용, 점검하여야 한다.

- telnet victimhost httpport[enter]
  PUT /파일이름 HTTP/1.1[enter]
  Host: victimhost(IP가능)[enter]
  Content-Length:컨텐츠 길이[enter]
  [enter] : enter를 치게 되면, 서버에서 content를 입력받을 준비를 함. 이때 입력한 컨텐츠가 PUT 뒤에 지정한 파일이름으로 생성됨.
  (화면상에 HTTP/1.1 100 Continue가 표시됨, 단, HTTP/1.0에서는 빈화면만 보임)
  이후에 생성할 파일의 내용(컨텐츠)를 입력하면 됨. 이때, 길이는 컨텐츠 길이에서 지정한 길이만큼이 입력됨
- 공격이 성공한 경우, 201 Created 또는 200OK로 서버는 응답함.



[그림 23] PUT Method 테스트



[그림 24] PUT Method를 이용한 파일 생성 성공

☑ 공격대상

o 대상 공격 부분 : 각 Method에는 아래와 같은 취약점이 존재한다.

method	safe	idempotent	visible semantics	identifiable resource	cacheable
GET	X	X	X	X	X
HEAD	X	X	X	X	X
PUT		X	X	X	
POST					*
DELETE		X	X	X	
OPTIONS	X	X	X		
PROPFIND	X	X	X	X	*

[그림 25] Method별 취약점 여부

☑ 피해유형

o 홈페이지 변조 : PUT 메소드를 통한 웹페이지 파일 업로드가 가능하며, 이를 통해 홈페이지 변조가 가능하다.

☑ 보호대책

o POST, GET 외에 어플리케이션에서 사용되지 않는 HTTP 메소드들(PUT, Delete, Mkdir, Option 등)은 모두 제한해야 한다.

o 메소드 제거 방법은 다음과 같다.

▸ IIS 5.0 HTTP 메소드 제한

- WebDAV 사용안함 처리

① 레지스트리 편집기(Regedt32.exe)를 시작

② 레지스트리에서 다음 키를 검색 함

HKEY_LOCAL_MACHINE₩SYSTEM₩CurrentControlSet₩Services₩W3SVC₩Parameters

③ 편집 메뉴에서 값 추가를 누른 후 다음 레지스트리 값을 추가함

값 이름 : DisableWebDAV

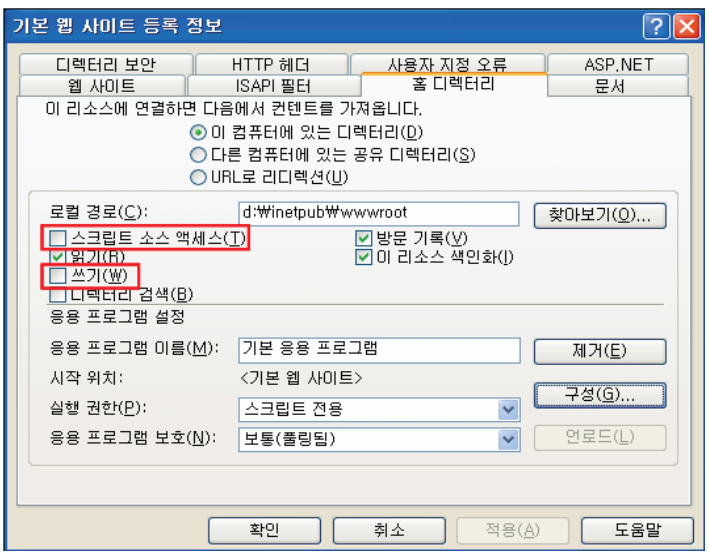
데이터 형식 : DWORD

값 데이터 : 1

④ IIS를 다시 시작한다.

[그림 26] WebDAV 사용안함 설정

- 웹 서비스 사용 폴더에 쓰기 권한 및 스크립트 소스 엑세스 권한 제거

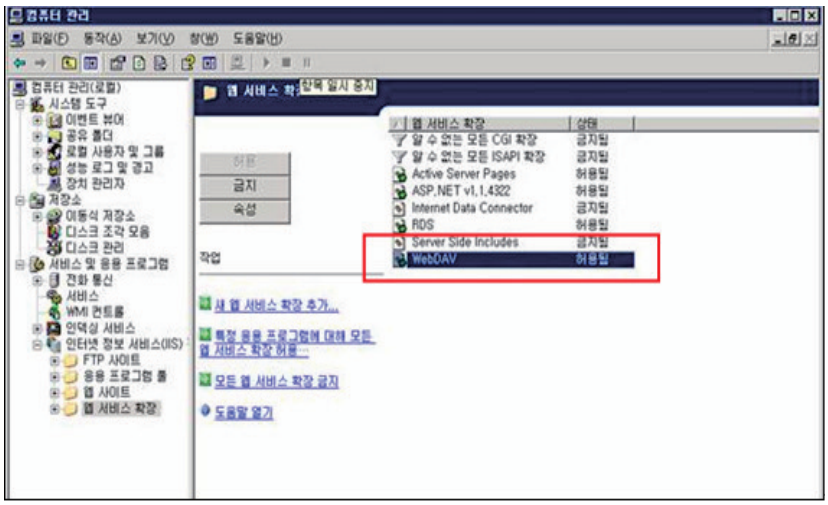


[그림 27] 웹 폴더 쓰기 권한 설정

※ File System(NTFS/FAT32/FAT16)의 쓰기 권한이 존재하면 게시판 혹은 업 로드 컴포넌트를 이용한 업로드 경로에서도 문제가 발생하지 않음

▸ IIS 6.0 HTTP 메소드 제한

- WebDAV 서비스를 제거하면 PUT, DELETE 같은 위험한 메소드 외의 GET, POST, TRACE, OPTIONS 메소드만 존재함



[그림 28] IIS 6.0 HTTP 메소드 제한

▸ Apache의 httpd.conf 파일 설정 방법

```
<Directory />

<LimitExcept GET POST>
        Order allow,deny
        deny from all
</LimitExcept>

</Directory>
```

- 위와 같이 설정하면 정상적으로 Apache 웹 서버에 로그인 권한을 가진 사용자 외에 다른 사용자는 제한된 Method를 사용할 수 없게 된다.

▸ WebToB 설정 방법

- WebToB는 기본적으로 "GET, POST, HEAD" 등을 지원하고 있으나, PUT, DELETE 등을 삭제할 때는 http.m 파일 내의 "*NODE" 절에서 아래와 같이 설정, "wscfl -i http.m" 명령어를 실행한 후 WebToB를 재기동해야 한다.
- Method = "-PUT, -HEAD, -DELETE, -TRACE"

```
webmain WEBTOBDIR = "/usr/local/webtob",
SHMKEY = 69000,
```

```
DOCROOT = "/usr/local/webto/docs",
User = "nobody",
Group = "nobody",
IndexName = "Index.html",
UserDir = "public_html",
DirIndex = "Index",
Method = "-PUT, -DELETE, -MOVE, -OPTIONS",
LanguagePrio = "kr"
```

▸ IPlanet 설정 방법

- 불필요한 메소드 제거 방법은 obj.conf 파일에서 아래와 같은 방법으로 제거하고자 하는 메소드를 입력하여 설정한다.

```
<Client method = "TRACE">
    AuthTrans fn = "set-variable" remove-headers = "transfer-encoding"
    set-headers = "content-length: -1" error = "501"
</Client>

<Client method = "PUT">
    AuthTrans fn = "set-variable" remove-headers = "transfer-encoding"
    set-headers = "content-length: -1" error = "501"
</Client>

<Client method = "DELETE">
    AuthTrans fn="set-variable" remove-headers = "transfer-encoding"
    set-headers = "content-length: -1" error = "501"
</Client>

<Client method = "MOVE">
    AuthTrans fn = "set-variable" remove-headers = "transfer-encoding"
    set-headers = "content-length: -1" error = "501"
</Client>

<Client method = "MKDIR">
```

```
        AuthTrans fn = "set-variable" remove-headers = "transfer-encoding"
        set-headers = "content-length: -1" error = "501"
</Client>

<Client method = "RMDIR">
        5AuthTrans fn = "set-variable" remove-headers = "transfer-encoding"
        set-headers = "content-length: -1" error = "501"
</Client>
```

▸ Tomcat 설정 방법

- 불필요한 메소드 제거 방법은 Tomcat 5.5₩conf₩web.xml 파일에 서 아래와 같이 설정되어 있는 메소드를 지우거나 주석 처리하여 설정한다.

```
<security-constraint>
<display-name>Example Security Constraint</display-name>
<web-resource-collection>
<web-resource-name>Protected Area</web-resource-name>
<!-- Define the context-relative URL(s) to be protected -->
<url-pattern>/jsp/security/protected/*</url-pattern>

<!-- If you list http methods, only those methods are protected -->
    <http-method>DELETE</http-method>
    <http-method>GET</http-method>
    <http-method>POST</http-method>
    <http-method>PUT</http-method>
</web-resource-collection>

<auth-constraint>
<!-- Anyone with one of the listed roles may access this area -->
<role-name></role-name>
</auth-constraint>
</security-constraint>
```

10. 에러처리 및 기타 정보 노출

☑ 개 요

o 에러페이지는 사용자가 웹서버에 비정상적인 요청을 하였을 때, 웹서버에서 해당 비정상 요청에 대해 알려주는 페이지이며, 300번대, 400번대, 500번대의 숫자로 표시된다.

o 이 중, 500번대의 코드는 서버 내부오류를 표시하는 코드로, 특히 DB에 잘못된 요청을 하였을 때, DB쿼리의 오류를 보여주며, 이를 통해 DB의 구조가 노출될 우려가 있다.

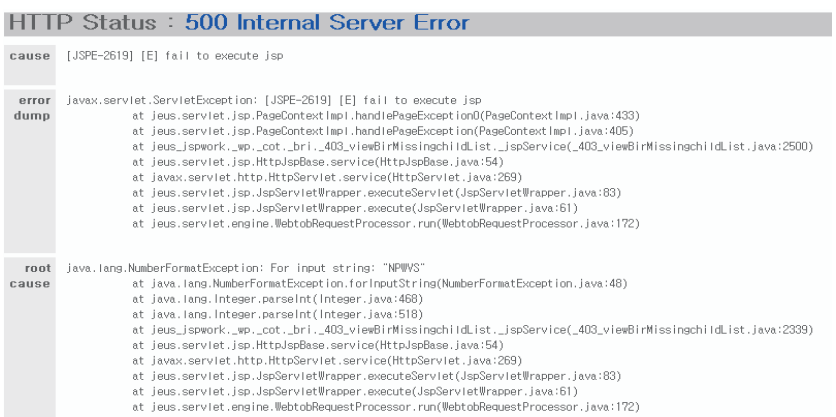
o 또한, 특정 시스템의 취약점 또는 작성자의 실수로 인하여 이름과 주민등록번호, 통장계좌번호, 신용카드 번호 등이 노출되어 해당 사용자의 정보가 직간접적으로 이용될 수 있다.

☑ 점검 방법

o URL의 파라미터를 변경하여 내부오류를 발생시킨다.

- 파라미터 값 변경 시도 : ttp://www.kkk.com/board/common.htmlapp&c=1201&page=' or 1=2--'

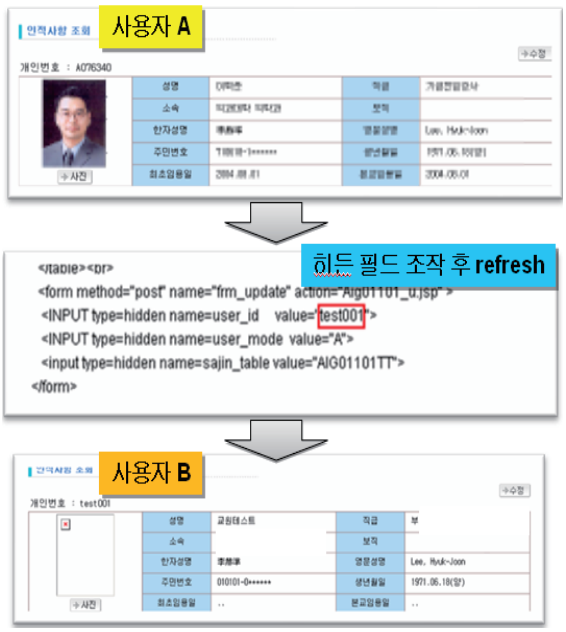
o 검색 창 또는 ID/PW 입력 창에 특수문자 등을 입력하여 에러 메시지 발생 여부를 확인한다.



[그림 29] 500에러 발생화면

o 사용자 개인정보보기 등 화면에서 마우스 오른쪽 버튼을 클릭한 후 "소스보기"를 선택하거나, 웹 프락시 도구를 사용하여 웹브라우저로부터 웹서버로 전송되는 내용을 확인하여 type=hidden으로 되어 있는 필드 값을 조작한다.

o 아래 화면에서 user_id 필드의 type이 hidden으로 되어있으며, 필드 값(value)을 웹 프락시 도구를 사용하여 조작하여 전송한다.



[그림 30] 히든 필드 조작

☑ 공격대상

o 대상 공격 부분 : Defult 에러 페이지를 통해 내부 서버의 구조가 노출될 가능성이 있다.

- ◆ 개인정보가 포함된 페이지
- ◆ 소스 코드에 남아 있는 주석문
- ◆ 웹 서버 정보
- ◆ 사용자 정보 보기 화면
- ◆ 로그인 시도 화면
- ◆ 게시물 수정 화면

☑ 피해유형

o 시스템 정보 노출 : 에러 페이지를 통해 웹서버정보 등 시스템 정보 노출 가능성이 있다.
o DB 정보 노출 : DB 접속 에러 메시지를 통한 DB 접속 정보 또는 테이블 명, 필드 명 등 노출 가능성이 있다.
o 개인 정보 노출 : 개인정보 및 중요한 정보들이 노출될 수 있다.
o 데이터 변조 : Paros 등의 웹 프락시 툴을 이용하여 히든 필드로 사용하는 고정 값(예: 금액, 코드, 아이디, 비밀번호 등)의 조작이 가능하다.
o 권한 획득 : 사용자의 히든 필드 값을 수정하여 인증을 우회할 수 있다.

☑ 보호대책

o 통일된 메시지를 출력한다.
  - 접근된 파일이 존재하지 않거나 거부되는 경우 : "access denied" 출력
  - 로그인 시 ID나 패스워드가 틀린 경우 : "로그인에 실패 했습니다" 출력
  - 웹 어플리케이션의 인수에 특수문자 입력 시 : "특수문자 입력 불가" 메시지 출력
o 최종 사용자에게 스택 추적 정보, 경로 정보와 디버그 정보 제공을 금지한다.
o 에러 메시지를 특정 URL로 리다이렉트 또는 예외 호출을 설정한다.
o 히든 필드 값을 그대로 사용하지 말고, 데이터베이스에서 재검색 하여 값을 새로 얻어 오거나 히든 필드로 전송된 값들에 대한 검증을 하도록 소스를 수정한다.
o 히든 필드로 값을 넘기는 것보다 세션의 속성을 통하여 변수를 저장 관리하는 것이 안전하다.

11. ID/PW 관리

☑ 개 요

o 취약한 ID/PW 취약점은 일반적으로 사용자 계정 또는 관리자 계정 생성 시 관리의 편의성을 위해 유추하기 쉬운 ID/PW를 사용하는 취약점이며,

o 공격자는 해당 취약점을 이용하여 관리자 또는 사용자 ID/PW를 유추하여 인가되지 않은 페이지에 접근을 시도한다.

☑ 점검 방법

o 로그인 페이지에 ID와 비밀번호를 유추하여 입력한 후 로그인 성공여부를 점검한다.

- test/test, admin/admin

o 운영 시 주기적으로 사용자 계정테이블을 점검, 유추하기 쉬운 ID 등에 대해 수정을 권고한다.

☑ 공격대상

o 대상 공격 부분

◆ 로그인 페이지

☑ 피해유형

o 관리자 권한 획득 및 개인 정보 유출 : 사용자나 관리자의 ID/PW 노출로 인한 개인 정보 유출 및 관리자 권한 노출이 발생할 수 있다.

☑ 보호대책

o 웹 서비스의 회원 가입 정책 중 사용자의 비밀번호 길이 및 형태에 패스워드 정책을 적용하여 비밀번호의 길이, ID 포함 검사, 영문과 숫자의 혼합 여부를 체크한다.

12. 환경설정 및 보안고려사항

☑ 개 요

o 위에서 열거한 점검항목 이외에 웹서버 운영상 보안적으로 고려해야 할 사항들을 기술 한다

☑ 중요 정보를 보여주는 페이지는 캐시를 사용하지 못하도록 설정

o 중요 정보를 보여주는 화면에 no-cache 설정을 하지 않을 경우, 로그아웃을 한 이후에도 [뒤로 가기] 버튼을 사용해서 해당 내용을 볼 수 있는 위험이 존재한다.

o no-cache 설정을 위해서 HTML HEAD 부분에 아래 내용을 추가한다.

<meta HTTP-EQUIV="Pragma" CONTENT="no-cache">

☑ Java Class 역 컴파일 문제

o Java 언어의 Byte-code 특성으로 인하여 Java class는 쉽게 역 컴파일이 가능하다. 만약 Java Applet에 중요 정보(예: 원격지 접속을 위한 ID/PW, DB 쿼리문, 직접 제작한 암호화 알고리즘, 프로그램 로직 등)를 hard-coding 했다면, 이를 발견한 공격자는 해당 정보를 악용할 수 있는 위험성이 존재한다.

o Sun Microsystems에서 Java 프로그램 개발 시 고려해야 할 다양한 보안 사항을 제공한다.

- Secure Code guidelines(SUN Microsystems) : http://www.java.sun.com/securit/seccodeguide.html

☑ ASP(Visual Basic, C++, C# 등을 사용한 모든 ASP에 적용)

o include 파일을 보호하기 위해 일반적인 디렉토리(/lib, /include, /library 등)를 사용하지 않도록 한다.

o include 파일들의 확장자를 .inc나 .lib 등을 사용하는 경우 웹 페이지 상에서 텍스트 파일로 인식하지 않도록 .asp를 붙여서 사용한다.(예: config.inc.asp, lib.inc.asp 등)

o ASPError 객체의 output을 사용자에게 전달하지 않도록 한다.

o SQL 쿼리를 ASP에서 직접 생성하는 것을 지양하고, Stored procedure를 사용하도록 한다.

```
직접 생성 방식
 strQuery = "SELECT something FROM table WHERE foo = '" + var1 + "' AND var = '" + var2 + "'";

Stored procedure를 사용한 생성 방식
 strQuery = sp_comefunc(var1, var2)
```

☑ PHP

o [PHP 4.0 이상] 환경 설정(php.ini) 내용 중 register_global을 "on"으로 설정할 경우, PHP 스크립트의 변수 값을 임의로 변경할 수 있는 취약성이 존재한다. 따라서 register_global은 "off"로 설정한 후 , $_GET, $_POST 문을 사용해서 사용자가 전달한 값을 얻어야 한다.

o PHP스크립트 오류를 사용자에게 보내지 않기 위해서 PHP 환경설정(php.ini)에서 아래와 같이 설정한다.

```
log_error = On
display_errors = Off
```

o 특정 파일의 내용보기 방지

- 오류 메시지가 발생된 CGI의 물리적 위치와 에러 부분이 표시 되는데, 이를 이용하여 공격자는 /lib, /inc, /admin 등 보여 지지 말아야 할 정보가 노출되는 위험성이 존재한다. 이를 제거하기 위해 php.ini 내의 설정 중에서 display_errors 값을 Off로 설정한다.

- 각 코딩 라인에 @를 사용하여 해당 라인의 에러 메시지를 출력하지 않는 방법을 사용한다

```
$abc = @mysql_connect($connect, $id, $pw);
@$abc = mysql_connect($connect, $id, $pw);
```

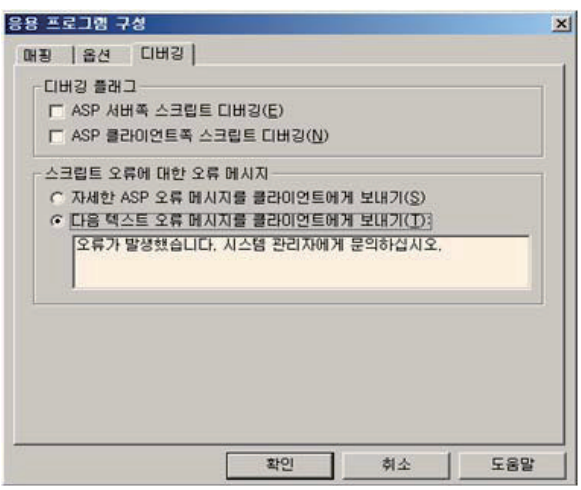
o include 파일을 보호하기 위해 일반적인 디렉토리(/lib, /include, /library 등)를 사용하지 않도록 한다.

o include 파일들의 확장자를 .inc나 .lib 등을 사용하는 경우 웹 페이지 상에서 텍스트 파일로 인식하지 않도록 .php를 붙여서 사용한다.(예: config.inc.php, lib.inc.php 등)

☑ MASS SQL Injection 예방을 위한 IIS웹서버 및 MS-SQL 설정

o 자세한 오류내용 표시 차단

- IIS 웹 서버에서는 기본적으로 웹 서비스의 오류가 발생 할 때, 자세한 오류 메시지를 접속자에게 표시하게 되어 있다. 이 설정을 변경하여 공격자가 오류 메시지를 통해 유용한 정보를 수집할 수 없도록 수정해야 한다.



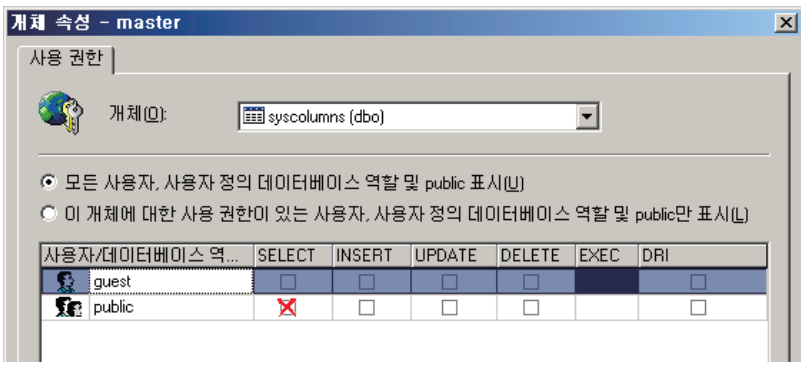
[그림 31] 스크립트 오류 메시지 설정화면

- 일반적인 SQL Injection 공격의 경우 오류 메시지를 기반으로 정보를 추출하게 되므로, 이 설정 변경만으로도 방어효과를 볼 수 있다. 다만, Blindfolded SQL Injection이나 시스템

명령어를 수행하는 SQL Injection공격은 차단할 수 없으므로 반드시 프로그램 수정의 보완조치로 활용해야 한다.

o SQL 서버 보안 강화

- 웹 페이지와 MS-SQL 서버를 연동 할 때, 데이터베이스의 관리자 계정인 SA 계정을 사용하게 되면 공격자가 악용할 수 있으므로 매우 위험하다. 그러므로 반드시 사용자 계정을 사용하고 최소 권한만을 할당하여 사용해야 한다.
- 또한, MASS SQL Injection 스크립트의 경우 시스템 테이블인 Syscolumns와 Sysobjects의 정보를 이용하고 있으므로, 반드시 필요하지 않은 경우라면 사용자 계정이나 "public"계정에 할당되어 있는 "SELECT" 권한을 제거하는 것이 안전하다.



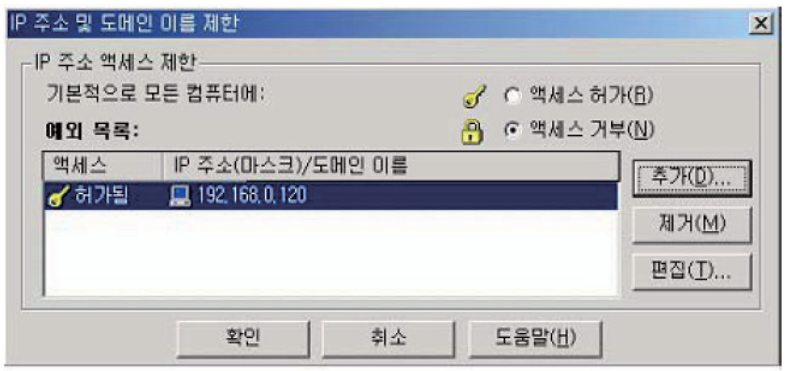
[그림 32] Public 계정 권한 제어

☑ 관리자 페이지에 대한 접근제어

o 관리자 페이지에는 접근권한을 가진 IP에서만 접근 가능하도록 ACL(Access Control List)을 설정한다.

▸ IIS 웹 서버에서 보호 대책

- 설정 ⇨ 제어판 ⇨ 관리도구 ⇨ 인터넷 서비스 관리자 선택
- 해당 관리자 페이지 폴더에 오른쪽 클릭하고 등록정보 ⇨ 디렉토리 보안 ⇨ IP 주소 및 도메인 이름 제한 ⇨ 편집 버튼을 클릭
- 액세스 거부를 선택하고 추가 버튼을 클릭하여 관리자 호스트 IP 또는 서브넷을 등록



[그림 33] IIS 보호 설정

▸ Apache 웹 서버에서 보호 대책

- Apache 웹 서버의 환경설정 파일인 httpd.conf 파일의 Directory 세션의 AllowOverride 지시자에서 AuthConfig 또는 All을 추가하여 .htaccess를 통하여 사용자 계정, 사용자 패스워드를 등록한 사용자만 접근이 가능하도록 하고 관리자 디렉토리(admin)에 대해 특정 IP에 대해서만 접근이 가능하도록 설정한다.

```
#먼저 접근을 제어하고자 하는 디렉토리에 대한 상위 디렉토리 정의에
#AllowOverride 부분이 'All', 'AuthConfig', 'FileInfo' 등으로 설정

<Directory /home/www/admin/>
   AllowOverride AuthConfig
   Order deny, allow
   Deny from all
   Allow from 10.10.100.7 10.10.2.1/24
</Directory>

AccessFileName .htaccess
<File ~ "^₩ht">
Order allow, deny
Deny from all
</Files>
<.htaccess>
AuthName "인증이 필요한 관리자 페이지입니다."
AuthType Basic
```

```
AuthUserFile /home/www/admin/.htpasswd
AuthGroupFile /dev/null
require valid-user
Order deny, allow
Deny from all
Allow from 10.10.100.7 10.10.2.1/24
```

- 관리자 페이지와 같이 인증이 필요한 디렉토리에 .htaccess 파일을 만들고 admin 계정의 패스워드 ~apache/bin/htpasswd를 이용하여 사용자 정보 파일(.htpasswd)을 생성한다.

```
<Directory /home/www/admin/>
# ~apache/bin/htpasswd -c /home/www/admin/.htpasswd [사용자명]
   New password: ********
   Re-type new password: ********
   Adding password for user [사용자명]
#
```

※ 주의사항

- Apache 서버의 경우 AllowOverride 지시자를 변경 시 apache restart가 필요하다.

- 관리자 페이지의 디렉토리명 변경 시 웹 프로그램에서 경로명을 지정하는 경우 수정하여야 한다.

- 관리자 디렉토리에는 일반 사용자의 접근이 필요한 파일이 존재하지 않아야 한다.

o 관리자 인증 후 접속할 수 있는 페이지의 경우 해당 페이지 주소를 직접 입력하여 접속하지 못하도록 관리자 페이지 각각에 대하여 관리자 인증을 위한 세션을 관리한다.

☑ 디렉토리 리스팅 방지 설정

o IIS

- [제어판] ⇨ [관리도구]의 [인터넷 서비스 관리자](혹은 [인터넷 정보 서비스]) 메뉴의 [기본 웹 사이트]에서 마우스 오른쪽 클릭, '속성' 부분을 보면 '기본 웹 사이트 등록 정보'에서 '홈 디렉토리' 부분을 클릭 한 후 '디렉토리 검색(B)' 부분의 체크를 해지한다.

o Apache
- 서버에서 "httpd.conf" 라는 파일을 찾음
- 파일 내용 중 Options 항목 뒤에 Indexes라는 단어를 지우고 파일을 저장한다. 이 때, Options는 디렉토리 별로 설정할 수 있게 되어 있으므로 모든 디렉토리에 대해서 Options 항목의 Indexes를 제거 후 Http 프로세스를 재기동 한다.

```
...
<Directory "/usr/local/www">
  Options Indexes <- 제거
</Directory>
...
```

☑ Default Page 제거

o 초기 서비스 설치 시 디폴트로 설치되는 컨텐츠 및 서비스들을 Disable 시킨다.
o 패스워드 변경, 경로 변경 등 접근제한 설정을 한다.
o 디폴트 디렉토리(/htdocs, /cgi-bin 등)를 삭제 또는 변경한다.
o 디폴트 페이지(manual, sample 등)를 삭제 또는 변경한다.

웹 어플리케이션별 Default Page 종류

- Web.xml 페이지
  http://www.kkk.com/WEB-INF/web.xml
- Apache Tomcat Default Page
  http://www.kkk.com:8080/index.jsp
- Apache Tomcat 관리자 페이지
  http://www.kkk.com:8080/admin
  http://www.kkk.com:8080/manager/html

※ 특히, 웹 어플리케이션별 관리자 페이지는 삭제하지 않을 경우, 이를 이용, 홈페이지 변조, 악성 프로그램 설치 등 심각한 피해가 우려되므로 반드시 제거하여야 한다.

☑ 백업파일 삭제

o 백업파일은 .bak, .back, .org, .orgin, .~, .log 등 웹 서버의 개발/유지보수 시 사용한 테스트 파일을 삭제하지 않을 경우, 공격자는 웹서버의 소스를 다운로드하여 DB접속정보, 웹서버 정보 등을

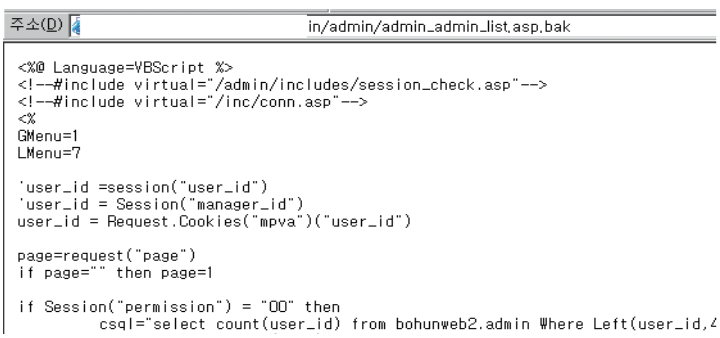
분석, 웹서버 침투, 자료유출 등 2차 공격에 악용할 수 있다.

o 웹 디렉토리와 다른 곳에서 백업 본을 생성하여 소스 수정 후 소스만 업로드 하도록 하여 웹 브라우저를 통해 보여 지는 디렉토리에는 html, asp, php, cgi, jsp 등만 올려놓도록 한다.

o 불필요한 파일 관리를 위해 httpd.conf 파일을 수정한다.

```
<FIles ~"₩.bak$">
    Order allow, deny
    Deny from all
</Files>
```

o 홈페이지 서버에 테스트 파일과 같은 불필요한 파일을 삭제하고 홈페이지 서비스와 관련 없는 디렉토리(백업 디렉토리 등)는 일반 사용자가 접근이 불가능 하도록 적절한 권한(디렉토리 또는 파일 접근 권한)을 설정한다.



[그림 34] 백업파일 예

<별첨1> ActiveX Control 개발가이드

ActiveX Control은 웹을 통해 다양한 기능과 미려한 디자인을 제공하기 위해 사용하는 프로그램으로 인터넷 뱅킹, 포털, 게임, 쇼핑몰 등 대부분의 웹사이트에서 사용하고 있다

이러한 ActiveX Control은 보안의 관점에서 웹페이지나 전자메일에 포함된 스크립트에 의해 사용자 PC 내부의 파일 엑세스, 레지스트리 수정 등 잠재적 악용가능성이 있어, ActiveX Control을 사용하도록 개발된 홈페이지는 개발 시 특별한 주의가 요구 된다

[출처 : ActiveX Control 개발보안가이드라인(2008.11, 국가사이버안전센터)]

☑ ActiveX Control 개발 가이드

o 문자열 입력값에 대한 크기 검증

- ActiveX Control은 보통 인터넷 익스플로러 웹브라우저에서 스크립트를 통해 실행된다. HTML 상에서 <param> 태그를 통하여 ActiveX Control이 초기화 될 때의 파라미터를 설정할 수 있고, <script> 태그를 통하여 변수를 사용하듯이 프로퍼티(property)를 설정할 수 있고, 함수를 사용하듯이 메소드를 호출할 수 있다.
- 문자열 입력 값에 대한 크기 검증이란 파라미터, 프로퍼티, 메소드를 통해 ActiveX Control에 입력되는 문자열이 미리 할당된 메모리의 크기보다 클 때 입력 값을 필터링 하는 행위를 말한다. .
- 즉, 반드시 미리 허용된 길이의 문자열만을 입력받고, 이후 문자열을 처리하는 과정에서도 할당한 버퍼보다 더 긴 문자열을 복사하지 않도록 프로그램을 제작하여야 한다.
- 미리 할당된 메모리보다 큰 문자열이 입력으로 들어왔을 때는 오류를 리턴하고, 문자열을 복사할 때는 미리 할당된 버퍼 크기보다 문자열이 크진 않은지 항상 확인하며, strcpy 함수 대신에 strncpy_s 함수를, memcpy 함수 대신에 memcpy_s 함수를 사용하는 등 문자열 복사에 보다 안전한 함수를 사용하여 프로그램을 구현하는 것이 바람직하다.

o 임의의 프로세스를 실행할 수 있는 기능 금지

- 임의의 프로세스를 실행할 수 있는 기능이란 ActiveX Control이 HTML의 <param> 태그나 스크립트 상에서 프로퍼티, 메소드 인자를 통하여 사용자 시스템에 있는 임의의 실행 파일에 대한 경로, 이름 및 실행 인자를 입력으로 받고 해당 파일을 실행하는 기능을 의미한다. 일반적으로 임의의 프로세스를 실행하는 기능은 보안상 매우 위험하므로 ActiveX Control의 메소드 형태로 구현하지 말아야 하지만, 보안을 고려하지 않는 개발자들이 쉽게 범할 수 있는 실수이고, 또한 자주 발견되고 있는 취약점이다.
- 프로세스 실행 취약점을 방지하기 위해서는 해당 ActiveX Control의 실행에 꼭 필요한 파일에 대해서만 프로세스 실행 기능을 제공하도록 프로그램을 개발하여야 한다. HTML 또는 스크립트 상에서 실행 파일의 경로와 이름 또는 실행 인자를 입력 받지 않도록 프로그램 소스 상에 고정하는 방법이 가장 안전하다.
- 불가피하게 HTML 또는 스크립트 상의 입력 값을 통한 프로세스 실행 기능이 필요하다면, 입력 값을 개인키로 암호화하여 변조될 수 없게 보호하거나, 경로나 이름으로 필터링 하여 기능을 제한하여야 한다. 또한, 실행 인자를 입력 받는 경우에는 실행 인자에 대한 크기를 검증하고 특수문자를 사용한 경우 지정된 범위를 벗어나지 않도록 구현에 주의가 필요하다.

o 임의의 파일 내용에 대한 읽기 기능 금지

- 임의의 파일 내용에 대한 읽기 기능이란 ActiveX Control이 HTML의 <param> 태그나 스크립트 상에서 메소드 인자, 프로퍼티를 통하여 사용자 시스템에 있는 임의의 파일에 대한 경로와 이름을 입력 받고 해당 파일의 내용을 메소드의 리턴 값이나 프로퍼티로 출력하는 기능을 의미한다.
- 임의의 파일 내용에 대한 읽기 기능은 범용으로 사용될 수 있다는 장점 때문에 ActiveX Control 개발자들이 일반적으로 구현하게 되는 대표적인 취약점 유형 중 하나이다.
- 파일의 내용을 읽어서 출력하는 기능이 제한되어 있지 않고 임의의 파일을 읽을 수 있게 설계되어 있다면 공격자는 취약한 해당 ActiveX Control을 악용하여 사용자 시스템의 중요 파일의 내용을 유출시킬 수 있다.
- 파일 읽기 취약점을 방지하기 위해서는 해당 ActiveX Control의 실행에 꼭 필요한 파일에 대해서만 파일 읽기 기능을 제공하도록 프로그램을 작성하여야 한다. HTML 또는 스크립트

상에서 파일 경로 및 이름을 입력 받지 않도록 프로그램 소스 상에 파일 경로 및 이름을 고정하는 방법이 가장 안전하다.

- 불가피하게 HTML 또는 스크립트 상의 입력 값을 통한 파일 읽기 기능이 필요하다면, 입력 값을 개인키로 암호화하여 변조될 수 없게 보호하거나, 폴더 이름, 파일 이름, 확장자 등으로 필터링 기능을 제한하여야 한다.

o 임의의 레지스트리 내용 읽기 기능 금지

- 임의의 레지스트리 내용 읽기 기능이란 ActiveX Control이 HTML의 <param> 태그나 스크립트 상에서 메소드 인자, 프로퍼티를 통하여 사용자 시스템에 있는 임의의 레지스트리에 대한 경로 및 이름을 입력 받고 해당 레지스트리의 내용을 메소드의 리턴 값이나 프로퍼티로 출력하는 기능을 의미한다.
- 임의의 레지스트리 내용에 대한 읽기 기능은 범용으로 사용될 수 있다는 장점 때문에 ActiveX Control 개발자들이 일반적으로 구현하게 되는 또 하나의 대표적인 보안 취약점 유형 중 하나이다.
- 레지스트리의 내용을 읽어서 출력하는 기능이 제한되어 있지 않고 임의의 레지스트리 내용을 읽을 수 있게 설계되어 있다면 레지스트리 읽기 취약점이 발생하게 된다. 레지스트리 읽기 취약점이 발생하면, 공격자는 취약한 해당 ActiveX Control을 악용하여 사용자 시스템의 중요 레지스트리의 내용을 외부로 유출시킬 수 있다.
- 레지스트리 읽기 취약점을 방지하기 위해서는 해당 ActiveX Control의 실행에 꼭 필요한 레지스트리에 대해서만 레지스트리 읽기 기능을 제공하도록 프로그램을 수정하여야 한다.
- HTML 또는 스크립트 상에서 레지스트리 경로 및 이름을 입력 받지 않도록 프로그램 소스 상에 레지스트리 경로 및 이름을 고정하는 방법이 가장 안전하다. 불가피하게 HTML 또는 스크립트 상의 입력 값을 통한 레지스트리 읽기 기능이 필요하다면, 입력 값을 개인키로 암호화하여 변조될 수 없게 보호하거나, 경로나 이름으로 필터링을 가하여 기능을 제한하여야 한다. 필터링을 구현할 때는 레지스트리의 경로가 ".."으로 우회되지 않도록 구현에 주의가 필요하다.

o 업데이트 파일에 대한 신뢰성 검증

- 최근 ActiveX Control들은 대부분 자동 업데이트 기능을 포함하고 있으며, 이러한 업데이트

기능은 HTML의 <param> 태그나 스크립트 상에서 메소드 인자, 프로퍼티를 통하여 업데이트 서버의 URL이나 설치 파일 등의 업데이트 정보를 입력 받아서 동작하는 것이 일반적이다.

- 만약 업데이트 정보(업데이트 서버의 URL, 설치 파일, 업데이트 되는 파일 등)를 공격자가 변조할 수 있다면, 정상적인 업데이트 기능을 악용하여 악성코드를 설치하는 것이 가능하다.
- ActiveX Control 프로그램의 업데이트를 공격자로부터 보호하기 위해서는 업데이트 파일에 대한 신뢰성 검증이 필요하며, 이를 위한 방법으로는 서명 값을 검증하는 방법이 가장 바람직하다.
- 서명 값 검증이란 해당 ActiveX Control 개발업체에서 공개키/개인키 쌍을 발급받고, 업데이트 파일에 대해서는 개인키로 서명한 값을 추가하여 전달하고, 사용자 시스템인 클라이언트 측에서는 전달된 업데이트 파일에 대하여 서명 값이 올바른지 공개키로 검증하는 방법을 의미한다. 즉, 업데이트 파일에 디지털 서명을 하여 업데이트 파일에 대한 신뢰성을 보장받는 것이다.
- 서명 값 검증 방법을 구현하기 어려운 경우에는 업데이트 URL을 HTML이나 스크립트 상에서 입력받지 아니하고 프로그램 소스 상에서 고정하여 사용하는 방법이 한 가지 대안이 될 수 있다.

o 관리자 권한의 폴더에 ActiveX Control 프로그램 설치

- ActiveX Control은 설치된 웹 사이트에서 실행될 뿐만 아니라 웹 게시물이나 이메일의 열람을 통해서도 손쉽게 실행이 가능하다. 또한, ActiveX Control을 개발하는 목적은 인터넷 익스플로러가 접근하지 못하는 사용자 시스템의 자원을 다루기 위해서가 대부분이다.
- 만약 이러한 ActiveX Control이 공격자에 의해 변조되어 악성코드로 변해 있다면 사용자 시스템에 치명적인 위협이 될 것이 자명하다.
- 불행히도 일부 ActiveX Control 개발자들은 이러한 위협에 대하여 고려하지 않은 채 업데이트나 사용상의 편의를 위해 인터넷 익스플로러에서도 접근이 가능한 낮은 권한의 폴더에 프로그램을 설치하기도 한다. 하지만, 낮은 권한의 폴더에 있는 파일은 쉽게 수정이 가능하기 때문에 잠재적으로 변조될 위험성이 크다 할 수 있다. ActiveX Control 프로그램을 공격자의 변조 공격으로부터 보호하기 위해서는 ActiveX Control 프로그램을 관리자 권한의 폴더에 설치하는 것이 중요하다.
- 이 때 ActiveX Control 프로그램이란 실행코드인 exe, dll, ocx 등의 파일을 의미하고, 관리자 권한의 폴더란 C:₩Windows₩의 하위 폴더 또는 C:₩Program Files₩의 하위폴더를 의미한다.

o 실행 가능한 웹 사이트의 제한

- 대부분의 ActiveX Control은 설치된 웹 사이트에서만 필요하며 그 외의 웹 사이트에서는 실행될 필요가 없다. 하지만, ActiveX Control은 설치된 웹 사이트에서 뿐만 아니라 다른 모든 웹 사이트나 이메일을 통해서도 손쉽게 실행이 가능하다. 문제는 특정 ActiveX Control이 보안 취약점을 가지는 경우 해당 ActiveX Control을 설치하고 사용했던 웹 사이트뿐만이 아니라 모든 웹 사이트와 이메일을 통해서도 공격이 가능해진다는 점이다.
- 따라서 ActiveX Control의 악용을 방지하기 위해서는 ActiveX Control의 실행 가능한 웹 사이트를 제한하여야만 한다. 실행 가능한 웹 사이트의 제한은 ActiveX Control의 보안 취약점을 제거해 줄 수는 없지만, 공격자의 공격을 효과적으로 방어해줄 수 있다. 왜냐하면 ActiveX Control의 악용은 해당 ActiveX Control이 설치되고 사용되는 사이트와는 관계가 없는 공격자의 웹서버 또는 이메일을 통해서 대부분 이루어지기 때문이다.
- ActiveX Control의 실행 가능한 웹 사이트를 제한하기 위해서는 ActiveX Control이 실행되는 시점에 웹 사이트 URL에 대한 필터링을 구현하여야 하며, 이를 위해 SiteLock Template을 활용할 것을 적극 권장한다.
- SiteLock Template이란 ActiveX Control이 실행되는 도메인을 제한해주기 위해 Microsoft 사에서 제공하는 ATL 템플릿으로써 실행 가능한 웹 사이트 리스트의 제한뿐만 아니라, 추가적으로 실행 가능한 인터넷 영역의 제한 및 실행 가능한 유효 기간 설정 등의 기능을 제공해 준다.
- SiteLock Template을 활용하지 않고 직접 웹 사이트 URL에 대한 필터링을 구현할 수 있겠으나 사용이 간편하고 안정적이며 URL이 우회될 수 있는 실수를 방지해 주기 때문에 SiteLock Template을 이용하는 편이 바람직하다. SiteLock Template을 설치할 수 있는 URL은 아래와 같다(또는 MSDN 사이트에서 SiteLock Template을 검색하면 찾을 수 있다).

  * SiteLock Template 설치 URL
    - http://www.microsoft.com/downloads/details.aspx? FamilyID=43cd7e1e-5719-45c0-88d9-ec9ea7fefbcb&displaylang=en

o '신뢰할 수 있는 사이트 추가' 남용 금지

- 윈도우 Vista의 강화된 보안정책으로 ActiveX Control의 설치 및 실행에 제한을 받게 되자 개발자들은 대대적인 수정작업에 들어가야 했다. 불행히도 일부 개발자들은 윈도우 Vista용 ActiveX Control을 수정하는 대신 자신의 ActiveX Control이 설치되거나 실행되는

사이트를 신뢰할 수 있는 사이트로 추가하였다. 신뢰할 수 있는 사이트에서는 윈도우 Vista에서 추가된 보안기능들이 대부분 해제되어 기존의 ActiveX Control들도 대부분 정상적으로 동작하기 때문이다.

- 인터넷 익스플로러는 웹 사이트를 보안 수준에 따라 인터넷 영역, 로컬 인트라넷 영역, 신뢰할 수 있는 사이트 영역, 제한된 사이트 영역의 4가지 영역으로 나누고 있다. 대부분의 사이트는 인터넷 영역에서 실행되며, 인터넷 영역에서는 높은 보안 수준이 제공된다.
- 신뢰할 수 있는 사이트는 사용자 스스로 해당 사이트가 안전하다고 특별히 지정하는 사이트로써, 인터넷 영역보다 보안 설정이 낮아지며 특히 윈도우 Vista에서는 보호모드가 해제된다. 보호모드가 해제되면 윈도우 Vista에서 추가된 대부분의 보안정책이 무력화된다.
- 따라서 이 문제를 해결하기 위해서는 보안설정을 낮추는 방식이 아니라, 신뢰할 수 있는 사이트가 아니더라도 ActiveX Control의 정상적인 실행이 가능하도록 ActiveX Control을 수정하여야 한다.

o 권한 상승 창에 대한 우회 금지

- 윈도우 Vista의 보호모드 하에서 인터넷 익스플로러는 낮은 Integrity의 사용자 권한으로 실행되며, ActiveX Control 또한 낮은 Integrity의 사용자 권한을 가진다. 이로 인하여, ActiveX Control이 "Program Files" 하위의 파일이나 "HKEY_LOCAL_MACHINE" 하위의 레지스트리와 같은 높은 권한을 가지는 로컬 자원을 다루기 위해서는 권한 상승이 필요하다.
- 일반적인 사용자 입장에서 빈번하게 팝업 되는 권한 상승 창은 번거로운 것이 사실이다. 따라서 개발자는 사용자 편의성을 위해 권한 상승 창이 팝업 되는 수가 최소화되도록 ActiveX Control을 설계하고 싶어 하며, 이런 의미에서 자동 권한 상승은 매력적이다. 하지만, ActiveX Control의 암시적 권한 상승은 보안적인 관점에서 보았을 때 위험하다.
- 왜냐하면, 보통의 경우 ActiveX Control이 보안 취약점을 가지고 있다 할지라도 보호모드 때문에 높은 권한으로 동작하는 것이 제한되는 반면, 암시적인 자동 권한 상승을 사용하는 ActiveX Control이 보안 취약점을 가지는 경우는 사용자 동의 없이도 높은 권한으로 악성 행위를 수행하게 할 수 있기 때문이다. 따라서 ActiveX Control 개발 시 이러한 암시적인 권한 상승의 위험성을 반드시 고려해야 할 것이다.

o ActiveX Control의 남용 금지

- 불필요한 ActiveX Control의 개발을 줄이고 대체기술로 전환하는 것이 바람직하다. ActiveX Control은 사용자 시스템의 자원을 접근해야 하고 다른 대체수단이 없는 경우에만 선택되어야 하며, 그 외에는 Ajax, Flash/FLEX, SilverLight 등의 더 나은 대체기술로 전환되어야 옳다. 그리고 부득이하게 ActiveX Control을 개발할 때에는 개발 초기부터 보안을 염두에 두고 설계 및 구현해야 한다. 안전한 ActiveX Control의 개발을 위해서는 개발자와 보안책임자 스스로 보안의식과 책임감을 가지는 것이 무엇보다도 필요하다.

o Windows Vista에서 안전한 ActiveX Control개발 체크리스트

내 용	세부 점검항목	비고
문자열 입력 값에 대한 크기 검증	■ HTML에서 <param> 태그를 통해 파라미터로 문자열을 입력으로 받아 처리하는 경우 프로그램에서 할당한 크기를 넘지 않는지 확인하는 루틴이 항상 포함되어 있는가?	
	■ 스크립트에서 프로퍼티(property)가 문자열을 입력으로 받아 처리하는 경우 프로그램에서 할당한 크기를 넘지 않는지 확인하는 루틴이 항상 포함되어 있는가?	
	■ 스크립트에서 메소드(method)가 인자로 문자열을 입력으로 받아 처리하는 경우 프로그램에서 할당한 크기를 넘지 않는지 확인하는 루틴이 항상 포함되어 있는가?	
임의의 프로세스 실행 기능 금지	■ 임의의 프로세스 실행 기능을 제한하고 있는가? ※ HTML의 <param> 태그나 스크립트 상에서 메소드, 프로퍼티를 통하여 사용자 시스템에 있는 임의의 실행 파일에 대한 경로 및 이름, 실행 인자를 입력으로 받고 해당 실행 파일을 실행하는 기능	
	■ 특정 경로내의 실행파일에 대하여 프로세스 실행 기능을 제공하는 경우 입력 값에 ".."을 사용할 수 없도록 제한하고 있는가? ※ HTML의 <param> 태그나 스크립트 상에서 메소드, 프로퍼티를 통하여 실행 파일에 대한 경로 및 이름, 실행 인자를 입력으로 받고 해당 실행 파일을 실행하는 기능	
임의의 파일 읽기 기능 금지	■ 임의의 파일 읽기 기능을 제한하고 있는가? ※ HTML의 <param> 태그나 스크립트 상에서 메소드, 프로퍼티를 통하여 사용자 시스템에 있는 임의의 파일에 대한 경로 및	

	이름을 입력으로 받고 해당 파일의 내용을 메소드의 리턴 값이나 프로퍼티로 출력하는 기능	
	■ 특정 폴더내의 파일에 대하여 읽기 기능을 제공하는 경우 입력 값에 "..."을 사용할 수 없도록 제한하고 있는가? ※ HTML의 <param> 태그나 스크립트 상에서 메소드, 프로퍼티를 통하여 파일에 대한 경로 및 이름을 입력으로 받고 해당 파일의 내용을 메소드의 리턴 값이나 프로퍼티로 출력하는 기능	
임의의 레지스트리 읽기 기능 금지	■ 임의의 레지스트리 읽기 기능을 제한하고 있는가? ※ HTML의 <param> 태그나 스크립트 상에서 메소드, 프로퍼티를 통하여 사용자 시스템에 있는 임의의 레지스트리에 대한 경로 및 이름을 입력으로 받고 해당 레지스트리의 내용을 메소드의 리턴 값이나 프로퍼티로 출력하는 기능	
	■ 특정 경로내의 레지스트리에 대하여 레지스트리 읽기 기능을 제공하는 경우 입력 값에 "..."을 사용할 수 없도록 제한하고 있는가? ※ HTML의 <param> 태그나 스크립트 상에서 메소드, 프로퍼티를 통하여 레지스트리에 대한 경로 및 이름을 입력으로 받고 해당 레지스트리의 내용을 메소드의 리턴 값이나 프로퍼티로 출력하는 기능	
업데이트 파일에 대한 신뢰성 검증	■ HTML의 <param> 태그나 스크립트 상에서 메소드 인자, 프로퍼티를 통하여 업데이트 URL 정보를 입력 받는 경우, 디지털 서명을 이용하여 업데이트 파일의 신뢰성을 확보하였는가?	
관리자 권한의 폴더에 프로그램 설치	■ ActiveX Control 프로그램(exe, dll, ocx 등의 파일)은 관리자 권한의 폴더(C:₩Windows₩의 하위 폴더 또는 C:₩Program Files₩의 하위폴더)에 설치되도록 제한하였는가?	
실행 가능한 웹 사이트 제한	■ Microsoft의 SiteLock Template를 활용하여 ActiveX Control이 실행 가능한 웹 사이트를 제한하였는가?	
신뢰할 수 있는 사이트 추가 남용 금지	■ ActiveX Control의 설치 및 실행을 위해서 해당 웹 사이트를 신뢰할 수 있는 사이트에 추가하는 기능을 제한하고 있는가?	
	■ ActiveX Control의 설치를 위해서 보안설정을 낮출 필요가 있는 경우 설치 후에 반드시 보안설정을 원래대로 복구하고 있는가?	
	■ ActiveX Control의 실행을 위해서 보안설정을 낮출 필요가 있는 경우 보안설정을 낮추는 방법 대신 권한 상승 창을 통해 ActiveX Control의 권한을 높이는 방법으로 구현하고 있는가?	
권한 상승 창 우회 금지	■ 권한 상승 창을 통해 사용자 동의 없이 ActiveX Control이 Medium Integrity의 사용자 권한을 가지는 프로세스(.exe)를	

	실행할 수 있도록 Elevation Policy 레지스트리 키를 사용하지 않았음을 확인 하였는가?	
	■ Medium Integrity 이상의 권한을 가진 대리자 프로세스(.exe)를 사용자 시스템에 상주하게 하여, ActiveX Control이 사용자 동의 없이 높은 권한이 필요한 일을 처리하도록 개발하지 않았음을 확인 하였는가?	
불필요한 ActiveX Control 남용 금지	■ 개발 되는 ActiveX Control이 사용자 시스템의 자원을 이용할 필요가 명확히 있고, 다른 대체수단이 없는 것을 확인하였는가?	
	■ ActiveX Control은 정상적인 사용자뿐만 아니라 악의적인 공격자 또한 이용할 수 있음을 염두하고, 해당 기능이 악용될 소지는 없는지 충분히 고려하였는가?	

<별첨2> 보안코딩 소스코드

1 스크립트 삽입(XSS), 스크립트 요청참조(CSRF)

o ASP

- Server.HTMLEncode() 함수를 사용하여, 특정 문자열에 대한 HTML encoding을 수행
- 적용 가능한 IIS : IIS 5.0이상
- 사용법 : <%= Server.HTMLEncode("입력값") %>
- 개발코드 예시

```
If use_html Then     'HTML tag를 사용하게 할 경우 부분 허용
memo = Server.HTMLEncode(memo)  'HTML encoding 수행

memo = replace(memo, "&lt;p&gt;", "<p>")
memo = replace(memo, "&lt;P&gt;", "<P>")
memo = replace(memo, "&lt;br&gt;", "<br>")
memo = replace(memo, "&lt;BR&gt;", "<BR>")

Else   'HTML tag를 사용하지 못하게 할 경우
memo = Server.HTMLEncode(memo)  'HTML encoding 수행
memo = replace(memo, "<", "&lt;")
memo = replace(memo, ">", "&gt;")
End If

Response.write "게시물 내용-" & memo & "<BR>"
```

o PHP

- htmlspecialchars() 함수를 이용하여 특정 문자열에 대한 HTML encoding을 수행.
- strip_tags() 함수를 이용하여 문자열로부터 HTML tag와 PHP tag를 제거
- 개발코드 예시

```
$use_tag = "img, font, p, br"; // 허용할 HTML tag
if($use_html == 1) // HTML tag를 사용하게 할 경우 부분 허용
$memo = str_replace("<", "&lt;", $memo); // HTML tag를 모두 제거
$tag = explode(",", $use_tag);
for($i=0; $i<count($tag); $i++) // 허용할 tag만 사용 가능하게 변경
```

```
$memo = eregi_replace("&lt;".$tag[$i]." ", "<".$tag[$i]." ", $memo);
$memo = eregi_replace("&lt;".$tag[$i].">", "<".$tag[$i].">", $memo);
$memo = eregi_replace("&lt;/".$tag[$i], "</".$tag[$i], $memo);
else // HTML tag를 사용하지 못하게 할 경우
// $memo = htmlspecialchars($memo);
// htmlspecialchars() 사용 시 일부 한글이 깨어지는 현상이 발생 할 수 있음
$memo = str_replace("<", "&lt;", $memo);
$memo = str_replace(">", "&gt;", $memo);
echo "게시물 내용-" . $memo . "<BR>₩n";
```

o JSP

- 개발코드 예시

```
if(use_html) // HTML tag를 사용하게 할 경우 부분 허용
memo = memo.replaceAll("<","&lt;"); //HTML tag를 모두 제거
memo = memo.replaceAll(">","&gt;");
// 허용할 HTML tag만 변경
memo = memo.replaceAll("&lt;p&gt;", "<p>");
memo = memo.replaceAll("&lt;P&gt;", "<P>");
memo = memo.replaceAll("&lt;br&gt;", "<br>");
memo = memo.replaceAll("&lt;BR&gt;", "<BR>");
else // HTML tag를 사용하지 못하게 할 경우
memo = memo.replaceAll("<","&lt;");
memo = memo.replaceAll(">","&gt;");
out.print("게시물 내용-" + memo + "<BR>");
```

2. SQL 구문 삽입

o ASP

```
prodId = Request.QueryString("productId")
prodId = replace(prodId, "'", "''") '특수문자 제거
prodId = replace(prodId, ";", "")
```

```
prodId = replace(prodId, "--", "")
prodId = replace(prodId, "+", "")
prodId = replace(prodId, "%", "")
prodId = replace(prodId, "<", "&lt;")
prodId = replace(prodId, ">", "&gt;")
prodId = replace(prodId, "(", "&#40;")
prodId = replace(prodId, ")", "&#41;")
prodId = replace(prodId, "#", "&#35;")
prodId = replace(prodId, "&", "&#38;")
prodId = replace(LCase(prodId), "@@variable", "")
prodId = replace(LCase(prodId), "@variable", "")
prodId = replace(LCase(prodId), "print", "")
prodId = replace(LCase(prodId), "set", "")
prodId = replace(LCase(prodId), "or", "")
prodId = replace(LCase(prodId), "union", "")
prodId = replace(LCase(prodId), "and", "")
prodId = replace(LCase(prodId), "insert", "")
prodId = replace(LCase(prodId), "openrowset", "")
set conn = server.createObject("ADODB.Connection")
set rs = server.createObject("ADODB.Recordset")
query = "select prodName from products where id = " & prodId
conn.Open "Provider = SQLOLEDB; Data Source = (local); Initial Catalog =
productDB; User Id = dbid; Password = "
rs.activeConnection = conn
rs.open query
If not rs.eof Then response.write "제품명" & rs.fields("prodName").value
Else response.write "제품이 없습니다"
End If
```

o PHP

```
$query = sprintf("SELECT id, password, username FROM user_table WHERE
id = '%s';", addslashes($id));
// id 변수를 문자형으로 받고, id 변수의 특수문자를 일반문자로 변환
// @로 php 에러 메시지를 막는다.
$result = @OCIParse($conn, $query);
if(!@OCIExecute($result)) error("SQL 구문 에러");
exit;
@OCIFetchInto($result, &$rows);
```

o JSP

```
String sql = "SELECT * FROM user_table" + " WHERE id = ?" + " AND password
= ?";
ResultSet rs = null;
PreparedStatement pstmt = null;
try
conn = DBManager.getConnection();
pstmt = conn.prepareStatement(sql);
pstmt.setString(1, request.getParameter("id"));
pstmt.setString(2, request.getParameter("password"));
rs = pstmt.executeQuery();
```

6. 파일 업로드

o ASP

```
Set Up = Server.CreateObject("SiteGalaxyUpload.Form")
Path1 = server.mappath(".") & "₩upload₩"
Fname = Up("file1")
if Fname <> "" then '파일 첨부가 되었으면
if Up("file1").Size > 10240 then '용량 제한
Response.Write "용량 초과"
Response.End
end if
if Up("file1").MimeType <> "image" then '이미지만 업로드 허용
Response.Write "이미지 파일이 아닙니다."
Response.End
end if
Filename=Mid(Fname,InstrRev(Fname,"₩")+1) '파일이름부분 추출
'중복 시에 파일이름부분을 변경하기 위해 분리
Farry=split(Filename,".") '.을 기준으로 분리
preFname=Farry(0) '파일이름 앞부분
extFname=Farry(1) '파일의 확장자
'저장할 전체 path를 만듦, 파일이름을 구함
Path2 = Path1 & Filename
saveFname=preFname & "." & extFname
```

```
Set fso = CreateObject("Scripting.FileSystemObject")
countNo = 0 '파일 중복될 경우 세팅 값
fExist=0 '같은 이름의 파일 존재 체크
Do until fExist = 1
If(fso.FileExists(Path2)) Then
countNo = countNo + 1
Path2 = Path1 & preFname & countNo & "." & extFname
saveFname=preFname & countNo & "." & extFname
else fExist=1
End If
Loop
Up("file1").SaveAs(Path2)
response.write(saveFname & " 저장완료")
else
response.write("Error")
end if
Set Up = nothing
```

o PHP

```
$uploaddir = '/var/www/uploads/';
//파일 사이즈가 0byte 보다 작거나 최대 업로드 사이즈보다 크면 업로드를 금지 시킴
if($_FILES['userfile']['name'])
if($_FILES['userfile']['size'] <= 0) //최대 업로드 사이즈 체크 삽입
print "파일 업로드 에러";
exit;
//파일 이름의 특수문자가 있을 경우 업로드를 금지시킴
if (eregi("[^a-z0-9₩._₩-]",$_FILES['userfile']['name']))
print "파일 이름의 특수문자 체크";
exit;
//파일 확장자중 업로드를 허용할 확장자를 정의
$full_filename = explode(".", $_FILES['userfile']['name']);
$extension = $full_filename[sizeof($full_filename)-1];
/* PHP의 경우 확장자 체크를 할 때 strcmp(확장자,"php3"); 로 체크를 하게 되면
pHp3 이나 phP3는 구별을 하지 못하게 되므로 strcasecmp처럼 대소문자 구별을 하지
않고 비교하는 함수를 사용. 또한 .를 기준으로 하여 확장자가 하나로 간주하고
```

```
프로그램을 할 경우 file.zip.php3 이라고 올린다면 zip파일로 인식하고 그냥 첨부가
되므로 아래와 같이 제일 끝에 존재하는 확장자를 기준으로 점검하도록 함 */
$extension= strtolower($extension);
if (!( ereg($extension","hwp") || ereg($extension","pdf") || ereg($extension","jpg")) )
print "업로드 금지 파일입니다";
exit;
$uploadfile = $uploaddir. $_FILES['userfile']['name'];
if (move_uploaded_file($_FILES['userfile']['tmp_name'], $uploadfile))
print "파일이 존재하고, 성공적으로 업로드 되었습니다.";
print_r($_FILES);
else
print "파일 업로드 공격의 가능성이 있습니다! 디버깅 정보입니다:₩n";
print_r($_FILES);
```

o JSP

```
<%@ page contentType = "text/html; charset=euc-kr" %>
<%@ page import="com.oreilly.servlet.MultipartRequest,com.oreilly.servlet.multipart.
DefaultFileRenamePolicy, java.util.*" %>
<%
String savePath = "/var/www/uploads"; //업로드 디렉토리
int sizeLimit = 5 * 1024 * 1024; //업로드 파일 사이즈 제한
try
MultipartRequest multi = new MultipartRequest(request, savePath, sizeLimit,
"euc-kr", new DefaultFileRenamePolicy());
Enumeration formNames = multi.getFileNames(); //폼의 이름 반환
String formName = (String)formNames.nextElement();
String fileName = multi.getFilesystemName(formName); //파일의 이름 얻기
String file_ext = fileName.substring(fileName.lastIndexOf('.') + 1);
if(!( file_ext.equalsIgnoreCase("hwp") || file_ext.equalsIgnoreCase("pdf") ||
file_ext.equalsIgnoreCase("jpg")) )
out.print("업로드 금지 파일");
if(fileName == null)
out.print("파일 업로드 실패");
```

```
else
fileName = new String(fileName.getBytes("8859_1"),"euc-kr"); //한글 인코딩
out.print("File Name : " + fileName);
catch(Exception e)
%>
```

7. 파일 다운로드

o ASP

```
file = Request.Form ("file") '파일 이름
Response.ContentType = "application/unknown" 'ContentType 선언
Response.AddHeader "Content-Disposition","attachment; filename=" & file
Set objStream = Server.CreateObject("ADODB.Stream") 'Stream 이용
strFile = Server.MapPath("./upfiles/") & "₩" & file '서버 절대경로
strFname = Mid(Fname,InstrRev(file,"₩")+1) '파일 이름 추출, ..₩ 등의 하위 경로 탐색은 제거 됨
strFPath = Server.MapPath("./upfiles/") & "₩" & strFname '웹 서버의 파일 다운로드 절대 경로
If strFile = strFPath Then '사용자가 다운 받는 파일과 웹 서버의 파일 다운로드 경로가 맞는지 비교
objStream.Open
objStream.Type = 1
objStream.LoadFromFile strFile
download = objStream.Read
Response.BinaryWrite download
End If
Set objstream = nothing '객체 초기화
```

o PHP

```
if (preg_match("/[^a-z0-9_-]/i",$up_dir))
print "디렉토리에 특수문자 체크";
exit;
if (preg_match("/[^₩xA1-₩xFEa-z0-9._-]|₩.₩./i",urldecode($dn_file_name)))
print "파일이름에 특수문자 체크";
```

```
exit;
$dn_path = "/var/www/data/$up_dir/$dn_file_name";
if (!file_exists($dn_path))
print "파일이 존재여부 체크";
exit;
//파일 전송 루틴
header("Content-Type: doesn/matter");
header("Content-Length: ".filesize("$dn_path"));
header("Content-Disposition: filename = ".$dn_file_name]);
header("Content-Transfer-Encoding: binary₩r₩n");
header("Pragma: no-cache");
header("Expires: 0");
```

o JSP

```
String UPLOAD_PATH = "/var/www/upload/";
String filename = response.getParameter("filename");
String filepathname = UPLOAD_PATH + filename;
if(filename.equalsIgnoreCase("..") || filename.equalsIgnoreCase("/"))
//파일 이름 체크
return 0;
//파일 전송 루틴
response.setContentType("application/unknown; charset = euc-kr");
response.setHeader("Content-Disposition","attachment;filename = " + filename + ";");
response.setHeader("Content-Transfer-Encoding:" , "base64");
try
BufferedInputStream in = new BufferedInputStream(new
FileInputStream(filepathname));
.....
catch(Exception e)
//에러 체크 [파일 존재 유무 등]
```

8. URL강제접속/인증우회(로그인)

o ASP

```
<html>
<head>
<title> Login </title>
<script>
function check_submit()
if(!login.user_id.value)
alert("아이디를 입력 하세요");
login.user_id.focus();
return false;
if(!login.password.value)
alert("아이디를 입력 하세요");
login.password.focus();
return false;
return true;
</script>
</head>
<body>
<form method=post action=login.asp onsubmit="return check_submit();"
name=login>
<TABLE border=0>
<TR>
<TD>아이디</TD>
<TD><input type=text name="user_id" value="" maxlength=12 size=19></TD>
</TR>
<TR>
<TD>패스워드</TD>
<TD><input type=password name="password" value="" maxlength=12
size=19><input type=submit value="login"></TD>
</TR>
</TABLE>
</form>
</body>
</html>
```

```
<% Option Explicit %>
<%
Dim user_id, password
user_id = Request.Form("user_id") '사용자로부터 입력 받은 아이디
password = Request.Form("password") '사용자로부터 입력 받은 패스워드
If UserAuth(user_id, password) <> 1 Then
Response.redirect("/login.html") '인증 실패 시 인증 페이지로 Redirect
Else
If Session("logged_in") <> 1 Then '인증된 사용자 인지 체크
Session("logged_in") = 1 '인증에 성공했을 경우 logged_in 에 1의 값을 세팅
Session("user_id") = user_id '사용자 ID 저장
Session("user_ip") = Request.Servervariables("REMOTE_ADDR") 'IP 저장
End If
Response.redirect("/main.asp") '인증 성공 시 Main 페이지로 Redirect
End If
%>
<%
Function stripQuotes(strWords)
stripQuotes = replace(strWords, "'", "''") '특수문자 제거
End Function
Function UserAuth(user_id, user_pwd) '사용자 인증
Dim objConn, objRs
Dim strConnection, strQuery
Set objConn = Server.CreateObject("ADODB.Connection")
Set objRs = Server.CreateObject("ADODB.RecordSet")
'DB 연결 정보, 별도의 헤더 파일로 관리하여 INCLUDE
strConnection = "DSN = MEMBER; uid = DBUSER; pwd = DBPASSWD"
On Error Resume Next '에러가 생길경우
objConn.Open strConnection
objRs.ActiveConnection = objConn
strQuery = "SELECT * FROM user_tbl WHERE user_id= '" &_
stripQuotes(user_id) & "' AND password='" &_
stripQuotes(user_pwd) & "'"
objRs.Open strQuery
If objRs.BOF or objRs.EOF Then' 올바른 사용자를 찾지 못했을 경우
UserAuth = 0
Else
```

```
UserAuth = 1
End If
objRs.Close ' DB 연결 해제
Set objRs = Nothing
objConn.Close
Set objConn = Nothing
End Function
%>
```

```
<%
If Session("user_ip") = Request.Servervariables("REMOTE_ADDR") AND
Session("logged_in") = 1 Then
Response.Write Session("user_id") & "님은 " & Session("user_ip") & "에서 접속하셨
습니다."
'인증에 성공한 IP와 사용자 IP를 비교, 인증 여부 비교
'... 중략 ...
Else
Response.write "허가되지 않은 사용자 입니다."
End If
%>
```

o PHP

```
<html>
<head>
<title> Login </title>
<script>
function check_submit()
if(!login.user_id.value)
alert("아이디를 입력하세요");
login.user_id.focus();
return false;
if(!login.password.value)
alert("아이디를 입력하세요");
login.password.focus();
```

```
return false;
return true;
</script>
</head>
<body>
<form method=post action=login.php onsubmit="return check_submit();"
name=login>
<TABLE border=0>
<TR>
<TD>아이디</TD>
<TD><input type=text name="user_id" value="" maxlength=12 size=19></TD>
</TR>
<TR>
<TD>패스워드</TD>
<TD><input type=password name="password" value="" maxlength=12
size=19><input type=submit value="login"></TD>
</TR>
</TABLE>
</form>
</body>
</html>
```

```
<?PHP
@session_cache_limiter('nocache');
@session_start(); //세션 데이터를 초기화
// form 에서 사용자 id와 사용자 password를 아래 변수로 전달
if(!UserAuth($_POST['user_id'],$_POST['password'])) //DB 에서 사용자 인증 처리하는 부
분
header("Location: login.html");
exit; //인증 실패 시 종료
//인증에 성공한 경우 처리해야 되는 부분
if (!session_is_registered("logged_in"))
$logged_in = 1; //인증에 성공했을 경우 logged_in 에 1의 값을 세팅
$user_id = $_POST["user_id"];
```

```
$user_ip = $_SERVER["REMOTE_ADDR"];
session_register("logged_in"); //인증 결과 저장
session_register("user_id"); //사용자 ID를 저장
session_register("user_ip"); //사용자 IP를 저장
header("Location: main.php");
?>
<?PHP
function UserAuth($userid, $userpwd)
$connect = mysql_connect("localhost","DBUSER","DBPASSWD");
mysql_select_db("MEMBER");
$strQuery = "SELECT * FROM user_tbl WHERE user_id ='" . addslashes($userid) . "'
AND password='" . addslashes($userpwd) . "'";
$result = @mysql_query($strQuery);
if($result)
if(mysql_num_rows($result))
$data = mysql_fetch_array($result);
$userLevel = $data["level"];
@mysql_free_result($result);
@mysql_close($connect);
return 1;
return 0;
@mysql_close($connect);
return 0;
?>
```

```
<?PHP
@session_start();
if(strcmp($_SESSION['user_ip'], $_SERVER['REMOTE_ADDR']) == 0 &&
session_is_registered('logged_in'))
//인증에 성공한 IP와 사용자 IP를 비교, 인증 여부 비교
//... 중략 ...
echo $_SESSION['user_id'] . "님은 " . $_SESSION['user_ip'] . "에서 접속하셨습니다.";
else
echo "허가되지 않은 사용자 입니다.";
exit;
?>
```

o JSP

```
<html>
<head>
<title> Login </title>
<script>
function check_submit()
if(!login.user_id.value)
alert("아이디를 입력 하세요");
login.user_id.focus();
return false;
if(!login.password.value)
alert("아이디를 입력 하세요");
login.password.focus();
return false;
return true;
</script>
</head>
<body>
<form method=post action=login.jsp onsubmit="return check_submit();"
name=login>
<TABLE border=0>
<TR>
<TD>아이디</TD>
<TD><input type=text name="user_id" value="" maxlength=12 size=19></TD>
</TR>
<TR>
<TD>패스워드</TD>
<TD><input type=password name="password" value="" maxlength=12
size=19><input type=submit value="login"></TD>
</TR>
</TABLE>
</form>
</body>
</html>
```

```
<%@ page contentType="text/html;charset=euc-kr" %>
<%@ page import="java.util.*" %>
<%@ page import="java.sql.* " %>
<%
String DB_URL = "jdbc:mysql://127.0.0.1/MEMBER"; //DB 연결 정보, 별도의 헤더 파일로 관리하여 INCLUDE
String DB_USER = "DBUSER";
String DB_PASSWORD= "DBPASSWD";
//HttpSession session = request.getSession(true); //Servlet의 경우만 추가
String user_ip = request.getRemoteAddr(); //연결된 사용자의 IP 획득
String user_id = null;
Connection conn;
PreparedStatement pstmt = null;
ResultSet rs = null;
try
Class.forName("org.gjt.mm.mysql.Driver"); //드라이버 등록
conn = DriverManager.getConnection(DB_URL, DB_USER, DB_PASSWORD); //DB연결
String query = "SELECT * FROM user_tbl WHERE user_id = ? AND password = ?";
pstmt = conn.prepareStatement(query);
pstmt.setString(1, request.getParameter("user_id")); //사용자 입력 값 전달
pstmt.setString(2, request.getParameter("password"));
rs = pstmt.executeQuery();
if(rs.next())
user_id = rs.getString(1); //DB에서 사용자 정보 획득
if(user_id != null)
if(session.getValue("logged_in") != "1")  //인증된 사용자 인지 체크
session.putValue("logged_in", "1"); //세션에 사용자 정보 기록
session.putValue("user_id", user_id);
session.putValue("user_ip", user_ip);
//LogSave(user_id, user_ip); //인증에 성공한 사용자 정보 기록
response.sendRedirect("/main.jsp"); //인증 성공 시 Main 페이지로 Redirect
else
response.sendRedirect("/login.html"); //인증 실패 시 인증 페이지로 Redirect
catch(Exception ex) //에러처리
out.println(ex);
finally //DB연결 종료
```

```
if(rs != null) try rs.close(); catch(SQLException ex)
if(pstmt != null) try pstmt.close(); catch(SQLException ex)
%>
```

```
<%@ page contentType="text/html;charset=euc-kr" %>
<%@ page import="java.util.*" %>
<%
//HttpSession session = request.getSession(true);
if(session.getValue("user_ip") == request.getRemoteAddr() &&
session.getValue("logged_in") == "1")
//인증에 성공한 IP와 사용자 IP를 비교, 인증 여부 비교
//...
out.println(session.getValue("user_id") + " 님은 " + session.getValue("user_ip") + "
에서 접속하셨습니다.");
//... 중략 ...
else
response.sendRedirect("/login.html"); //인증 실패 시 인증 페이지로 Redirect
%>
```

8. URL강제접속/인증우회(쿠키사용방지)

o ASP

```
<%
'form 에서 사용자 id와 사용자 password를 아래 변수로 전달
If myfunc_userauth(userid, userpw) <> 1 Then 'DB 에서 사용자 인증을 처리하는 부분
Response.write "인증 실패"
Else
'인증에 성공한 경우 처리해야 되는 부분
If Session("logged_in") <> 1 Then
```

```
Session("logged_in") = 1 '인증에 성공했을 경우 logged_in 에 1의 값을 세팅
Session("userid") = userid
Session("user_ip") = Request.Servervariables("REMOTE_ADDR")
End If
End If
...
%>
```

```
<%
IF Session("user_ip") = Request.Servervariables("REMOTE_ADDR") AND
Session("logged_in") = 1 Then
'인증에 성공한 IP와 사용자 IP를 비교, 인증 여부 비교
'...
Else
Response.write "허가되지 않은 사용자 입니다."
End If
%>
```

o PHP

```
<?PHP
@session_start(); //세션 데이터를 초기화
// form 에서 사용자 id와 사용자 password를 아래 변수로 전달
if(!myfunc_userauth($userid,$userpw)) //DB 에서 사용자 인증을 처리하는 부분
print "인증 실패";
exit; //인증 실패 시 종료
//인증에 성공한 경우 처리해야 되는 부분
if (!session_is_registered("logged_in"))
$logged_in = 1; //인증에 성공했을 경우 logged_in 에 1의 값을 세팅
$user_ip = $_SERVER["REMOTE_ADDR"];
session_register("logged_in"); //인증 결과 저장
session_register("userid"); //사용자 ID를 저장
```

```
session_register("user_ip"); //사용자 IP를 저장
...
?>
```

```
<?PHP
session_start();
if(strcmp($_SESSION['user_ip'], $_SERVER['REMOTE_ADDR']) == 0 &&
session_is_registered('logged_in'))
//인증에 성공한 IP와 사용자 IP를 비교, 인증 여부 비교
//...
else
print "허가되지 않은 사용자 입니다.";
exit;
?>
```

o JSP

```
<%@ page contentType="text/html;charset=euc-kr" %>
<%@ page import="java.util.*" %>
<%@ page import="java.sql.* " %>
<%
//HttpSession session = request.getSession(true);
// form 에서 사용자 id와 사용자 password를 아래 변수로 전달
if(!myfunc_userauth(userid, userpw)) //DB 에서 사용자 인증을 처리하는 부분
out.println "인증 실패";
else
//인증에 성공한 경우 처리해야 되는 부분
session.putValue('logged_in',"1");
session.putValue('userid',userid);
session.putValue('user_ip',request.getRemoteAddr());
...
%>
```

```
<%@ page contentType="text/html;charset=euc-kr" %>
<%@ page import="java.util.*" %>
<%@ page import="java.sql.* " %>
<%
//HttpSession session = request.getSession(true);
String user_ip = session.getValue("user_ip");
if(user_ip.equals(request.getRemoteAddr()) && logged_in.equals("1"))
//인증에 성공한 IP와 사용자 IP를 비교, 인증 여부 비교
//...
else
out.println "허가되지 않은 사용자 처리.";
%>
```

12. 환경설정 및 보안 고려사항

o ASP

```
<%
If myfunc_userauth(userid, userpw) <> 1 Then 'DB에서 사용자 인증을 처리
Response.write "인증 실패"
Else
If Request.ServerVariables("REMOTE_ADDR") <> "10.10.1.1" Then '관리자 IP 확인
Response.write "관리자 IP가 아닙니다."
Response.write "인증실패"
LogSave(userid, user_ip, 0) '접속에 실패한 ID 및 IP 기록
Else
Session("logged_in") = 1 '인증에 성공했을 경우 logged_in 에 1의 값을 세팅
Session("userid") = userid
Session("user_ip") = Request.ServerVariables("REMOTE_ADDR")
LogSave($userid, $user_ip) '접속에 사용한 ID 및 IP 기록
... 중략 ...
End If
End If
%>
```

o PHP

```
<?PHP
@session_start(); //세션 데이터를 초기화
if(!myfunc_userauth($userid, $userpw) || $_SERVER["REMOTE_ADDR'] != "10.10.1.1")
//DB 에서 사용자 인증을 처리, 관리자 IP인지 확인
print "인증 실패";
LogSave(userid, user_ip, 0) '접속에 실패한 ID 및 IP 기록
exit; //인증 실패 시 종료
//인증에 성공한 경우 처리해야 되는 부분
if (!session_is_registered("logged_in"))
$logged_in = 1; //인증에 성공했을 경우 logged_in 에 1의 값을 세팅
$user_ip = $_SERVER["REMOTE_ADDR"];
session_register("logged_in"); //인증 결과 저장
session_register("userid"); //사용자 ID를 저장
session_register("user_ip"); //사용자 IP를 저장
LogSave($userid, $user_ip); // 접속한 사용자 ID 및 IP 기록
... 중략 ...
?>
```

o JSP

```
<%@ page contentType = "text/html; charset = euc-kr" %>
<%@ page import="java.util.* " %>
<%@ page import="java.sql.* " %>
<%
//HttpSession session = request.getSession(true);
String user_ip = request.getRemoteAddr();
// form 에서 사용자 id와 사용자 password를 아래 변수로 전달
if(!myfunc_userauth(userid, userpw) || !user_ip.equals("10.10.1.1"))
//DB 에서 사용자 인증을 처리, 관리자 IP인지 확인
out.println "인증 실패";
LogSave(userid, user_ip, 0) '접속에 실패한 ID 및 IP 기록
else
//인증에 성공한 경우 처리해야 되는 부분
session.putValue("logged_in","logok");
session.putValue("userid",userid);
```

```
session.putValue("user_ip", user_ip);
LogSave(userid, user_ip); //접속한 사용자 ID 및 IP기록
%>
```

웹 응용프로그램 개발 보안가이드 2010

발행일자 : 2010년 1월 1일

발행기관 : 행정안전부

홈페이지 : 행정안전부 (http://www.mopas.go.kr)

정부통합전산센터 (http://www.ncia.go.kr)

전자우편 : nciacert@mopas.go.kr

전화번호 : 042 - 250 - 5770~3