발간사

최근 해킹동향이나 보안의 화두는 웹 보안과 개발 보안에 집중되고 있습니다. 이것은 과거에 비해 보안 담당자 및 시스템 운영자의 보안 의식 수준이 많이 향상되었으며 시스템 및 네트워크 공격에 대비를 철저히 하고 있어 크래커들의 관심이 웹으로 변화된데 기인 합니다.

결국, 보안을 강화하기 위해서는 보안의 적용 포인트를 이미 구현이 완료된 단계보다는 웹 어플리케이션이 개발되고 구현되는 시점으로 변환되어야 한다는 것입니다.

또한, 한 보안 전문가의 연구결과에 따르면 개발의 각 단계 중 설계 단계에서 정보보호가 고려될 경우 21%, 구현 단계에서는 15%, 시험 단계에서는 12%의 비용이 절감될 수 있음이 실증적으로 확인되어 비용적인 측면에서도 그 우수성이 검증 되었습니다.

이에 과학기술정보보호센터(S&T-SEC)는 주요 웹 취약점 항목을 기준으로 홈페이지 개발 시 필요한 어플리케이션 언어별 보안 요구사항과 홈페이지 개발시 안전한 프로그래밍 예와 취약한 프로그래밍 예를 들어 홈페이지 개발 시 개발자가 실제로 참고가 될 수 있도록 PHP, ASP, JSP 프로그래밍에 대한 보안사항을 프로그래밍 예시를 통해 제시하고자 합니다.

2007. 11. 30

과학기술정보보호센터장

황 일 선

본 가이드는 최신 해킹 기술 및 침해시도 사례를 기반으로 과학기술 분야 특성에 맞는 안전한 홈페이지 운영 및 관리를 위하여 과학기술정보보호센터의 연구원들이 참여하였으며, 한국정보보호진흥원이 발간한 “홈페이지 개발보안 가이드”를 참고하여 제작되었습니다.

2007년 11월 30일

주관연구기관명 : 한국과학기술정보연구원
주관연구책임자 : 황일선 (고성능연구망사업단 단장)
참 여 연 구 원 : 이혁로(선임기술원) 박학수(선임연구원)
이행곤(선임연구원) 정기문(연 구 원)
최상수(연 구 원) 이호선(연 구 원)
김주범(연 구 원)

Part IV
홈페이지 개발 시 보안 매뉴얼

Part IV 홈페이지 개발 시 보안 매뉴얼

표 목차

그림 목차

1. 개요

가. 개요

미국은 홈페이지 서버를 포함한 정보시스템을 개발 과정에서 정보보호의 반영 및 구현에 관한 본격적인 제도화를 1997년부터 국방부를 중심으로 시행 하였으며, 미 상무부 산하의 NIST(National Institute of Standards and Technology)에서는 연방기관 및 민간에서 정보시스템 개발 시에 정보보호 요건을 평가하고 승인하는데 적용할 수 있는 각종 지침을 2002년 이후 본격적으로 개발하고 있다.

특히 2002년 7월에 개정된 OECD 정보보호가이드라인에서는 정보시스템 및 네트워크의 설계 구현 단계부터의 정보보호를 고려하도록 권고 하고 있다.

정보시스템의 정보보호를 제공하는 방법중에 하나인 내장(embedded) 방식은 정보시스템 계획 단계에서부터 정보보호 요구사항을 파악하여 정보시스템 분석 및 설계에 정보보호 기능을 구현하는 방식으로, 초기에는 정보보호 요구사항 파악 및 기능 구현을 위한 시간과 노력이 요구되나 다른 정보시스템 기능과 원활한 상호 운용성을 제공할 수 있어 결과적으로 비용효과적인 방식이라 할 수 있다

그러나 이러한 큰 장점에도 불구하고 많은 웹 개발자들은 홈페이지를 구축시의 효율성 및 편의성에 치중하여 설계 구축 단계에서 정보보호를 고려하지 못하고 있으며, 이로 인하여 대다수의 홈페이지들이 많은 취약점을 잠재하고 있는 것이다.

본 장에서는 홈페이지 개발 시 개발자들이 구현하는 언어별 보안대책을 제시함으로써 개발이후 보안취약점을 사전에 차단하고자 한다.

나. 적용범위

과학기술부 산하 정보보호 대상기관에서 기 구축 운영중인 홈페이지와 향후 개편 및 재개발 예정인 홈페이지를 대상으로 한다.

다. 활용방법

본 장에서는 주요 웹 취약점 항목을 기준으로 홈페이지 개발 시 필요한 어플리케이션 언어별 보안 요구사항과 개발시 안전한 프로그래밍 예 및 취약한 프로그래밍 예를 들어 개발자가 실제로 참고가 될 수 있도록 하였다. 즉, PHP, ASP, JSP 프로그래밍에 대한 보안사항을 프로그래밍 예시를 통해 제시하고자 한다.

라. 용어정의

1) 스크립트(script) : 컴퓨터 프로세서나 컴파일러가 아닌 다른 프로그램에 의해 번역되고 수행되는 명령문의 집합이다.
2) PHP(Personal Hypertext Preprocessor) : 하이퍼텍스트 생성 언어(HTML)에 포함되어 동작하는 스크립팅 언어. 별도의 실행 파일을 만들 필요 없이 HTML 문서 안에 직접 포함시켜 사용하며, C, 자바, 펄 언어 등에서 많은 문장 형식을 준용하고 있어 동적인 웹 문서를 빠르고 쉽게 작성할 수 있다. ASP(Active Server Pages)와 같이 스크립트에 따라 내용이 다양해서 동적 HTML 처리 속도가 빠르며, PHP 스크립트가 포함된 HTML 페이지에는 .php, .php3, .phtml이 붙는 파일 이름이 부여된다. 처음에는 'Personal Home Page Tools'이라 불렸으며, 공개된 무료 소스이다.
3) ASP.NET(active server pages.net) : 하나 이상의 작은 내장 프로그램(스크립트)를 갖고 있는 HTML페이지가 사용자에게 보여지기 위해 서버에서 수행되는 것이다. 서버의 웹 스크립트는 사용자의 요구에 따라 데이터베이스에서 찾은 결과로 순간적으로 웹페이지를 만든다.
4) JSP(Java Server Page) : 웹 서버에 있는 서브릿(servlet)을 사용해 웹 페이지의 내용과 모양을 제어하는 기술. 서브릿 응용 프로그램 인터페이스(API)로서 ASP(Active Server Page)와 유사하나 자바 서버 페이지(JSP)는 자바 프로그램

을 호출하고 ASP는 스크립트(VBScript나 JScript)를 사용한다. 자바 서브릿과 연결된 하이퍼텍스트 생성 언어(HTML) 페이지는 .JSP라는 파일 이름이 붙는다. 일명 서브릿이라고 한다.

5) 인증(authentication) : 컴퓨터에서 전자화된 정보로 상대편의 신원을 확인하는 방법이다.
6) 세션(session) : 망 환경에서 사용자 간 또는 컴퓨터 간의 대화를 위한 논리적 연결, 프로세스들 사이에서 통신을 하기 위해 메시지 교환을 통해 서로를 인식한 이후부터 통신을 마칠 때까지의 기간이다.
7) 버퍼오버플로우(Buffer Overflow) : 루트권한과 관련된 프로그램에 예상치 못한 입력값을 보내 해당 프로그램의 에러를 유발하는 것이다.

2. 프로그래밍 언어 보안

가. PHP 언어보안

(1) register_globals

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

그런데 이전 PHP버전에서는 register_globals=on 이 디폴트 설정이었기 때문에 개발자들은 예전에 구현한 프로그램과의 호환성을 위해 register_globals=on으로 바꾸고 있다.

register_globals=on 설정을 바꾸지 않고 막을 수 있는 방법은 if문으로 파일명 같은변수를 검사하여 지정한 값들이 아니면 오류를 표시하고 버리도록 구현 해야 한다.

```
egister_globals = on
가능 -> $_POST['user_name']
가능 -> $user_name

register_globals = off
가능 -> $_POST['user_name']
불가능 -> $user_name // 다만 extract()로 처리하면 가능

==================================
헤더를 담당하는 부분에서,
// php.ini 의 register_globals=off 일 경우 (php >= 4.1.0)
@extract($_GET);
@extract($_POST);
@extract($_SERVER);
위와 같이 처리하여 사용할 수 있다
```

(2) 외부 파일 불러오기 방지

외부에서 파일을 불러오는 문제는 php.ini 설정에서 allow_url_fopen=off로 설정해두면, 외부에서 파일을 가져올 수 없다.

```
php.ini
===============
// Fopen 의 설정
===============
// URL(http:// 나 ftp://)을 파일로서 취급할지를 결정한다.
Allow_url_fopen = off
```

(3) 특수문자 필터링

GET 방식으로 입력되는 "../"는 일반적으로 필터링 하도록 설계되고 있다. 그런데 쉘에서 ".₩./"는 "../"와 동일하게 간주된다. 따라서 특수문자에 대한 전반적인 필터링 장치가 필요하다.

php.ini 중 magic_guotes_gpc=on 으로 설정하면, "₩"가 "₩₩"으로 처리된다.

```
php.ini
===============
// magic_guotes_gpc 의 설정
===============
Allow_url_fopen = on
```

(4) 위험한 함수 사용 방지

공격자는 PHP 취약성을 파악하려고 phpinfo() 함수나 passthru() 함수를 이용해 웹 서버에 대한 권한을 얻으려고 한다. 이처럼, 공격에 악용될 수 있는 함수[1]들은

[1] 주로 passthru(), phpinfo(), system(), shell_exec(), exec(), proc_open() 등이 대상이 된다.

php.ini 설정 중 disable_functions 라인을 추가하면 위험을 줄일 수 있다.

```
php.ini
===============
// disable_fuctions 의 설정. 기본값 없음. 특정함수의 사용여부를 NOT를 사용하여 제한, 단 세이프모드일 때만 가능하다. 아래는 exec 실행을 금지하는 예.
===============
disable_functions = exec
```

(5) 파일 업로드 방지

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

- 자료실이나 게시판 어플리케이션을 작성할 때 PHP 실행파일은 올리지 못하도록 막는다. 올리지 못하는 파일을 검사하는 것보다는 올릴 수 있는 파일을 검사하는 방법이 효과적이다.
- 파일의 확장자 검사를 할 때, “Strcmp(확장자, “hp3”):” 검사하면 pHp3나 phP3는 구별하지 못한다. 따라서 “Strcasecmp()”같은 대소문자를 구별하지 않고 비교하는 함수를 사용해야 한다.
- 파일의 확장자 검사를할 때, “.” 기준으로 파일명과 확장자를 구별하는 경우가 많다. 파일명의 앞에서 부터 “.” 검사할 경우, 만약 “File.zip.php3” 파일을 올린다면 zip파일로 인식하고 파일은 업로드 된다. 따라서 다음과 같이 프로그래밍 하도록 한다.

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

(6) superglobal array

URL과 폼, 쿠키의 값은 $_GET와 $_POST, $_COOKIE 등 superglobal array[2]를 통해접근한다.

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

(7) 해쉬 – 파라미터 조작 검증

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

일단 데이터와 해쉬값을 수신한후, 데이터를 재해쉬하여 수신한 해쉬값이 이전값과 동일한지를 확인한다.

```
// sending the cookie
$secret_word = 'gargamel';
$id = 123745323;
$hash = md5($secret_word.$id);
setcookie('id',$id.'–'.$hash);
// receiving and verifying the cookie
list($cookie_id,$cookie_hash) = explode('–',$_COOKIE['id']);
if (md5($secret_word.$cookie_id) == $cookie_hash) {
$id = $cookie_id;}
else {
die('Invalid cookie.');
}
```

[2] 4.1.0 버전 이후 PHP는 웹 서버, 환경설정, 사용자 입력과 관련된 미리 선언된 배열 변수 집합을 추가적으로 제공한다. 여기에 속하는 변수들은 전역변수이며, 이런 변수를 superglobal array(슈퍼전역변수)라고 부른다.

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

(8) 접근제어 모듈 – PEAR

자체 개발한 접근제한 모듈보다는 PEAR(PHP Extension and Application Repository)[3] 모듈을 사용한다. "Auth" 사용자를 위한 cookie 기반 인증 방식이고, "Auth_HTTP"는 브라우저 기반의 인증 방식이다.

(9) 안전한 세션 관리

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

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

[3] http://pear.php.net/package-info.php?package=Auth
http://pear.php.net/package-info.php?package=Auth_HTTP

며, 웹 어플리케이션의 보안성을 고려하여 설계하도록 한다.

```
php.ini
===============
// session.save_path 설정
===============
session.save_path= /tmp/session
```

(10) Cross-site scripting (XSS) 대책

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

PHP에서는 신뢰할 수 없는 데이터에 대한 필터링 툴이 제공되고 있다.

- htmlspecialchars()는 특수문자를 HTML 엔티티 형태로 변환한다. 선택적인 두번째 인자 quote_style은 작은 따옴표와 큰 따옴표를 어떻게 사용할지를 말해준다. 기본 모드인 ENT_COMPAT 은 단지 큰 따옴표만 변환하고 작은 따옴표는 그대로 내버려 둔다. ENT_QUOTES 가 지정되면, 작은 따옴표와 큰 따옴표 모두를 번역하며, ENT_NOQUOTES는 작은 따옴표와 큰 따옴표 모두를 번역하지 않는다.
- strtr()는 원하는 문자를 필터링 한다. HTML 엔티티로 변경하는 것은 XSS 공격을 피할 수 있는 보호대책이 된다.

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

- strip_tags()는 스트링에서 HTML과 PHP 태그를 제거한다.
- utf8_decode()는 유니코드 UTF-8 인코딩 스트링내의 ISO-8859-1 문자를 1바이트의 ASCII 문자로 변경한다. 때때로 XSS 공격자는 그들의 공격 내용을 유니코드 인코딩으로 숨기려는 시도를 하기 때문에 uft8_decode()로 악성코드를 발견할 수 있게 된다.

(11) 패치

PHP는 런타임 메모리 할당이나 포인터의 개념이 없기 때문에 C 코드와 같은 버퍼 오버플로우는 없다. 그러나 PHP에서 제공하는 함수자체 또는 확장 버전에서 발생할 수 있는 버퍼 오버플로우를 피하기 위해서는 PHP 모듈에 대한 패치가 중요하다. 따라서 보안 메일링 리스트[4]에 가입하여 공지되는 패치와 보안권고문을 참고하도록 한다.

(12) 명령어 주입 공격에 대한 방어

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

현재 웹 페이지에 쉘을 통해 명령어를 입력 받고 있다면, 우선은 정말로 그러한 인터페이스가 필요한지에 대해 재 점검한다. 만약 외부의 프로세스를 실행시키기 위한 인수 등을 사용자로부터 입력받아야 하는 경우라면, 입력값을 escapeshellcmd()와 escapeshellarg()을 사용해 점검 하도록 한다.

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

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

[4] PHP Mailing Lists: http://www.php.net/mailing-lists.php

(13) 에러 처리

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

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

(14) 암호화

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

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

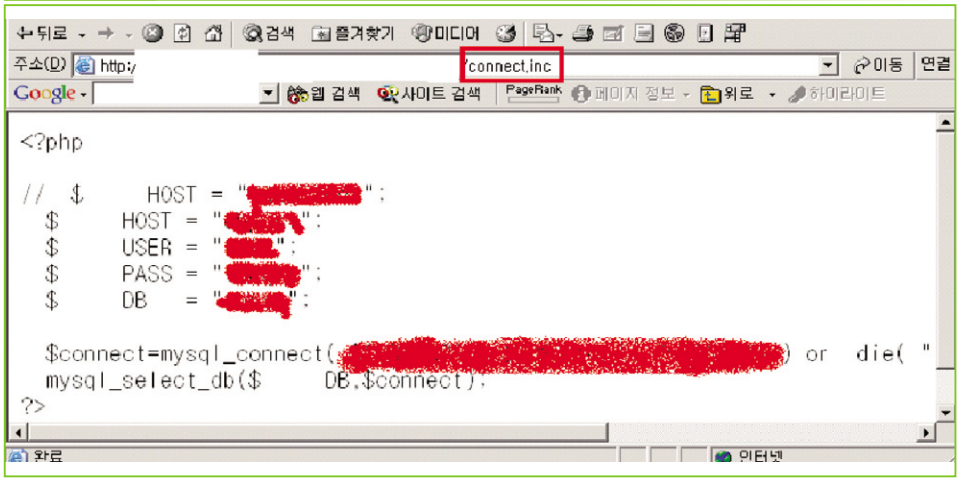
(15) SSL 설정

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

(16) 데이터베이스의 분리

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

[그림 1] 데이터베이스 접속 계정 정보 노출 예



(17) 환경 설정 파일

PHP의 환경 설정 파일인 두개의 php.ini 샘플 파일(php.ini-dist와 php.inirecommended) 중 php.ini-recommended를 사용하도록 한다.

나. ASP.NET 언어보안

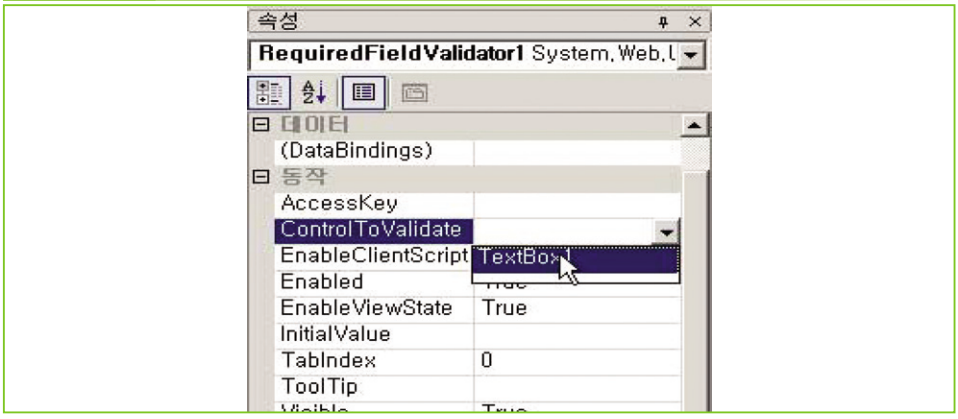
(1) 입력값의 유효성 검사

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

- string form field 입력값은 정규표현(regular expression)을 사용해서 유효성 검사를 한다(예를 들어, RegularExpressValidator 컨트롤을 사용한다)
- 정규 HTML 컨트롤, query strings, cookies, form field 등의 입력값에 대해 Regex 클래스를 사용하여 입력값을 검사한다.
- 데이터가 입력되는 곳에는 RequiredFieldValidator 컨트롤을 사용하도록 한다.
- 서버 컨트롤의 범위 체크는 RangeValidator 컨트롤을 사용한다.
- free form에 대한 입력 부분에서는 보안통제를 우회하는 악성 데이터의 입력에 주의해야 한다.

- 사용자가 입력한 값을 화면에 다시 보여줄 경우, HtmlEncode와 UrlEncode로 인코딩 한다.
- 지정된 가상 경로 혹은상대 경로를 서버의 실제 디렉토리에 매핑하는 MapPath는 다른 어플리케이션을 매핑할 경우, 적절한 위치에서 사용을 제한하도록 해야 한다.
- ASP.NET version 1.1 validateRequest 옵션은 활성화 한다.
- URLScan을 웹 서버에 설치하여 정기적으로 점검한다.
- HttpOnly cookie 옵션은 XSS 공격을 방어할 수 있다(IE6.1 이후에 적용).

[그림 2] 유효성 검사 예



(2) 인증

- 사이트를 공개 영역과 인증을 요구하는 제한영역으로 구별하여 설계한다.
- 보안과 비보안 폴더로 구별된 사이트에서는 절대경로의 URL을 사용하고, 주요 페이지로 이동할 경우 사용자 인증을 다시 한번 하도록 설계한다.
- credentials나 authentication cookie의 경우, SSL을 사용하거나 암호화 하고 무결성 체크(protection="ALL)를 하도록 한다.
- slidingExpiration 속성은 “false"로 설정하고, SSL로 보호되지 않는 authentication cookie는 타임아웃을 설정하도록 한다.
- form authentication cookie는 requireSSL 속성과 Secure cookie property를 사용해서 HTTPS 연결로 제한하도록 한다.
- application cookie는 유일한 이름과 경로의 조합으로 되어 있어야 한다.

- 패스워드는 사용자 공간에 저장하지 않는다. 대신 패스워드에 대한 힌트를 저장하도록 한다.
- 강력한 패스워드 정책을 세운다.
- 〈credentials〉 구문은 formauthentication을 위한 〈forms〉구문 안에 사용하지 않도록 한다.

(3) 권한

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

(4) 민감한 데이터 관리

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

(5) 세션 관리

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

(6) 파라미터 조작 방지

- view state는 MAC(message authentication code)[5]으로 보호되어야 한다.
- 서버의 주요 정보가 query string에 포함되려면 query string의 해쉬 정보로 대신해야 한다.
- 모든 입력 파라미터에 대해 유효성 검사를 수행한다.

(7) 에러 처리

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

(8) 환경 설정

- 설정파일에 대한 수정시도는 HttpForbiddedHandler를 사용해 막도록 한다.
- ASP.NET는 least-privilaged account로 운영하도록 한다.
- 개인신상정보 등은 Aspnet_setreg.exe에 의해 〈processModel〉상에서 암호화되어야 한다.
- machine-wide 정책을 설정하기 위해서, Web.config 세팅을 Machine.config 내 allowOveride="false"로 잠근다.

[5] MAC(message authentication code): 메시지 인증 코드. ①컴퓨터 보안에서 메시지의 내용, 작성자, 발신처 등 속성의 정당성을 검증하기 위해 메시지와 함께 전송되는 어떤 값 또는 부분으로 암호 기법에서, 하나의 인증 알고리듬으로 데이터를 처리해서 생성된 수 또는 값을 말함. 디지털 서명 부호라고도 함.

[표 1] Web.config파일의 설정

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

구 문	설 정 권 고
〈identity〉	가장ID(Impersonation identities)[6]는 Aspnet_setreg.exe를 사용해서 레지스트리 값에서 암호화 되어야 한다. 〈identity impersonate="true" userName="registry:HKLM\SOFTWARE\YourApp\identity\ASPNETSETREG, userName" password="registry:HKLM\SOFTWARE\YourApp\identity\ASPNET_SETREG,password"/〉
〈authorization〉	Role name의 현재 포맷을 검사한다.
〈machinseKey〉	만약 다수의 ASP.NET 어플리케이션이 같은 웹 서버에서 운영될 때, IsolateApps 옵션을 세팅하여 각 어플리케이션의 비밀키를 생성하도록 해야 한다. 〈machineKey validationKey="AutoGenerate, IsolateApps" decryptionKey="AutoGenerate, IsolateApps" validation="SHA1" /〉
〈sessionState〉	① 만약 mode=StateServer라면, credential은 Aspnet_setreg.exe로 암호화 하여 레지스트리에 저장한다. ② 만약 mode=SQLServer라면, windows 인증은 state store database에 연결할 때 사용되고, credential은 Aspnet_setreg.exe.를 사용해서 암호화하여 레지스트리에 저장한다.
〈httpHandler〉	사용안된 파일 타입은 파라미터 조작 공격을 막기 위해서 HttpForbiddenHandler에 매핑 시킨다. 〈add verb="*" path="*.rem" type="System.Web.HttpForbiddenHandler"/〉
〈webServices〉	사용하지 않는 프로토콜은 비활성화 시킨다. 자동으로 만들어지는 WSDL(Web services description language)는 비활성화 시킨다(선택사항임)

[6] '가장(impersonation)' 의미는 말 그대로 계정을 흉내 낸다는 뜻이다. 가장이 필요한 어플리케이션은 대개 서버측 어플리케이션으로서 클라이언트가 서버에 대한 권한이 있는지를 손쉽게 검사하기 위해 사용되는 방법이다.

다. JSP 언어보안

(1) 패치

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

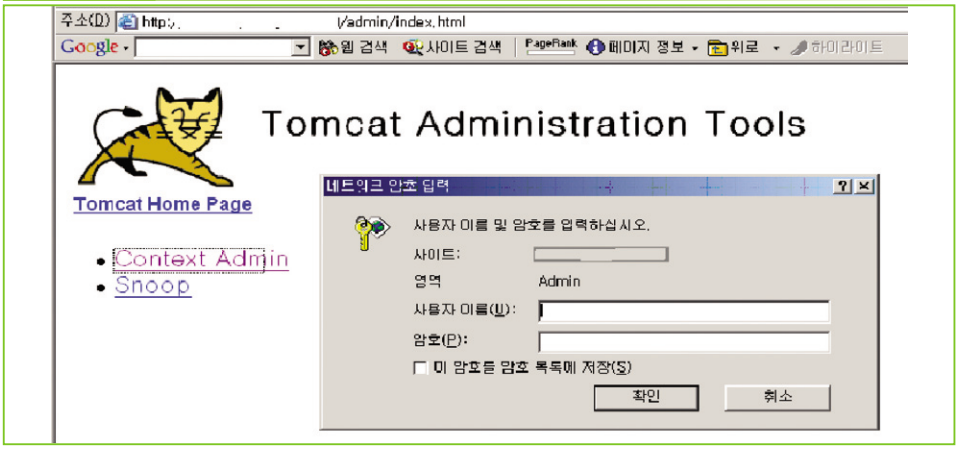
만약 아파치+Tomcat+JSP 환경의 웹 서버라면, 아파치에서 다음과 같이 설정하도록 하거나, Tomcat 모듈의 보안 패치를 최신으로 유지하도록 한다.

```
〈LocationMatch "/(%3f|\?)\.jsp"〉
AllowOverride None
Deny from all
〈/LocationMatch〉
```

(2) admin 디렉토리 관리

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

[그림 3] Tomcat 관리자 페이지 노출 예



(3) 디렉토리 구분

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

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

만약 같은 디렉토리를 써야 할 경우라면, 다음과 같이 설정한다.

◆ 아파치 환경에서 JSP 디렉토리를 링크하는 방법

```
〉ln .s JSP디렉토리 아파치 디렉토리
```

◆ 아파치 환경에서 디렉토리를 인식 시키도록 설정하는 방법

```
ScriptAlias /jsp/ “/usr/local/jsp/dics/”
```

(4) 디렉토리 리스팅

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

◆ Tomcat의 경우 web.xml

```
〈init-param〉
        〈param-name〉listings〈/param-name〉
        〈param-value〉false〈/param-value〉
〈/init-param〉
```

◆ Resin 의 경우 resin.conf

```
〈directory-servlet〉false〈/directory-servlet〉
```

3. 홈페이지 개발 시 보안사항

가. 접근통제 보안 프로그래밍

웹 관리자 메뉴의 접근을 특정 네트워크 대역으로 제한하여, IP Address 까지도 인증요소로 체크하도록 웹 관리자 사용자인터페이스를 개발하고, 관리자 인증 후 접속할 수 있는 페이지의 경우 해당 페이지 주소를 직접 입력하여 들어가지 못하도록 관리자 페이지 각각에 대하여 관리자 인증을 위한 세션관리를 해야 한다.

- 접근 통제 정책을 구현하고 있는 코드는 구조화, 모듈화가 되어 있어야 한다.
- 접근제어가 필요한 모든 페이지에 통제수단(로그인 체크 및 권한 체크)을 구현해야 한다. 특히, 하나의 프로세스가 여러 개의 페이지 또는 모듈로 이루어져 있을 때 권한 체크가 누락되는 경우를 방지하기 위해서 공통 모듈을 사용하는 것을 권장한다.
- 인증 과정을 처리하는 부분에 Client Side Script(Javascript, VBScript 등)을 사용하면 사용자가 임의로 수정할 수 있으므로 Server Side Script (PHP, ASP, JSP 등)를 통하여 인증 및 필터링 과정이 수행되어야 한다.

(1) 취약한 프로그래밍 예

```
〈HTML〉
〈HEAD〉〈TITLE〉 관리자 페이지 〈/TITLE〉
〈SCRIPT language="JavaScript “〉
function getCookie(name)
var cname = name +"=";
var dc = document.cookie;

if(dc.length 〉 0)
begin = dc.indexOf(cname);
if(begin != −1)
begin += cname.length;
end = dc.indexOf(";", begin);
```

```
if(end == -1) end = dc.length;
retrun unescape(dc.substring(begin, end));

return null;

function getValue(element)
var value = getCookie(element.name);
if(value != null) element.value = value;

</SCRIPT>
</HEAD>
<BODY>
<SCRIPT language="JavaScript ">
var auth;
auth = getCookie("logged_in");

if(auth != 1) // 인증 성공 쿠키가 없을경우 Main Page로 이동
window.location = "http://victim.com/login.html";

</SCRIPT>
```

(2) 안전한 프로그래밍 예

1) 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 성공했을경우 logged_in 에 1의 값을 셋팅
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
%>
```

2) PHP

```
<?PHP
@session_start(); //세션 데이터를 초기화
if(!myfunc_userauth($userid, $userpw) || $_SERVER["REMOTE_ADDR"] !=
"10.10.1.1")
//DB에서 사용자 인증을 처리, 관리자 IP인지 확인
print "인증 실패";
LogSave(userid, user_ip, 0) // 접속에 실패한 ID 및 IP 기록
exit; //인증 실패시 종료

//인증에 성공한 경우 처리 해야 되는 부분
if (!session_is_registered("logged_in"))

$logged_in = 1; //인증에 성공했을경우 logged_in 에 1의 값을 셋팅
$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 ...
```

3) 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 처리 해야 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
...
```

나. 파라미터 설정값

(1) 취약한 프로그래밍 예

1) ASP

```
<%
strSize = Request.QueryString("font_size") // 사용자로부터 폰트의 크기 입력

Response.Write "<HTML><TITLE>사용자 입력값 검증</TITLE></HEAD>"
Response.Write "<BODY>"
```

```
Response.Write "<FONT size=" & strSize & ">글자 크기 조절</FONT>"

' ... 중략 ...
```

2) PHP

```
<?PHP
include "./inc/dbconn.inc"; // DB 연결 헤더
include $language . "/head.html"; // 각 국가 언어별 HTML 출력

$conn = mysql_connect($SERVER, $USER, $PASSWD);
$query = "select count(*) from main_tbl";

// ... 중략 ...
```

3) JSP

```
<%@ page contentType="text/html;charset=euc-kr" %>
<%@ page import="java.util.* " %>

<HTML><HEAD><TITLE> 사이트 접속 불가 </TITLE>
<META HTTP-EQUIV="Refresh" CONTENT="10;URL=http://victim.com/bye.html">
</HEAD>
<BODY>
<%
out.print("지금 사용하고 계신 ");
out.print(request.getHeader("USER-AGENT"));
out.print(" 브라우져로는 사이트 접속이 불가능 합니다.");
%>
</BODY>
</HTML>
```

(2) 안전한 프로그래밍 예

1) ASP

```
<%
Size = Request.QueryString("font_size") // 사용자로부터 폰트의 크기 입력

Size = CInt(Size) // 입력되는 값을 정수로 형 변환

Response.Write "<HTML><TITLE>사용자 입력값 검증</TITLE></HEAD>"
Response.Write "<BODY>"
Response.Write "<FONT size=" & Size ">글자 크기 조절</FONT>"

// ... 중략 ...
```

2) PHP

```
<?PHP
@require_once "./inc/dbconn.inc"; // DB 연결 헤더
$default_lang = "korea"; // 기본값 설정

if(!file_exists($language."/head.html")) // 파일이 존재하는지 체크
if(eregi(":\/\/", $language)) $language = $default_lang; // URL이 포함되는지 체크
else // 파일이 없는 경우 기본값 설정
$language = $default_lang;

@require_once $language . "/head.html"; // 각 국가 언어별 HTML 출력

$conn = @mysql_connect($SERVER, $USER, $PASSWD);
$query = "select count(*) from main_tbl";

// ... 중략 ...
```

3) JSP

```
<%@ page contentType="text/html;charset=euc-kr" %>
<%@ page import="java.util.* " %>

<HTML><HEAD><TITLE> 사이트 접속 불가 </TITLE>
<META HTTP-EQUIV="Refresh" CONTENT="10;URL=http://victim.com/bye.html">
</HEAD>
<BODY>
<%
String user_agent = request.getHeader("USER-AGENT");
// HTTP HEADER 중 USER_AGENT를 변경 하여 크로스사이트 스크립트 공격하는
것을 차단
user_agent = user_agent.replaceAll("<","&lt;"); // HTML tag가 있을 경우 제거
user_agent = user_agent.replaceAll(">","&gt;");
out.print("지금 사용하고 계신 ");
out.print(user_agent);
out.print(" 브라우져로는 사이트 접속이 불가능 합니다.");

%>
</BODY>
</HTML>
```

다. 세션설정

쿠키 저장 시 타인이 임의로 쿠키를 읽어 들일 수 없도록 도메인과 경로 지정에 유의해야 하며, 서버 측에서 유효성 여부를 확인할 수 있는 대책을 강구한다. 예를 들어 브라우저에 저장된 쿠키에만 의존하는 쿠키방식보다는 서버 측에 일부 정보를 저장하여 상호 대조할 수 있는 세션(Session) 방식으로 대체하고, 세션방식의 경우도 서버 측에 사용자의 IP정보 등을 함께 저장하여 유효성 여부를 확인하는 방식을 권한다.

Session 방식은 접속자 별로 세션을 생성하여 사용자의 정보를 각각 저장 할 수 있는 오브젝트로써 페이지의 접근을 허가하거나 금지할 때 또는 사용자별로 정보를 저장할 때 많이 사용된다. 클라이언트의 자원을 사용하는 쿠키와는 달리 세션은 서버 쪽의 자원을 차지하고 있으므로 보안을 고려하여 세션방식을 채택하는 것이 바람직 하다.

(1) 취약한 프로그래밍 예

1) ASP

```
// login_ok.asp 사용자 인증 처리를 하는 스크립트
<%
 // 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 처리 해야 되는 부분
    Response.Cookies("logged_in") = 1
 // 인증에 성공했을경우 logged_in 에 1의 값을 셋팅
    Response.Cookies("userid") = userid
  End If
  ...
%>
```

```
user_menu.asp // 사용자 검증이 필요한 페이지
<%
  IF Request.Cookies("logged_in") = 1 Then
    Response.write "허가된 사용자 입니다."
  Else
    Response.write "허가되지 않은 사용자 입니다."
  End If
%>
```

2) PHP

```
//login_ok.php // 사용자 인증 처리를 하는 스크립트
<?PHP
// 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 실패시 종료

// 인증에 성공한 경우 처리 해야 되는 부분
setcookie("logged_in", "1"); // 인증에 성공했을경우 logged_in 에 1의 값을 셋팅
setcookie("userid", $userid);
...
?>

//user_menu.php // 사용자 검증이 필요한 페이지
<?PHP
if($_COOKIE["logged_in"] == 1)
echo "인증 성공: " . $_COOKIE["userid"];

?>
```

3) JSP

```
<%@ page contentType="text/html;charset=euc-kr" %>
<%@ page import="java.util.*" %>
<%@ page import="java.sql.* " %>
// login_ok.jsp// 사용자 로그인 처리를 하는 스크립트
<%
// 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 처리 해야 되는 부분
Cookie cookie1 = new Cookie("logged_in", "1");
response.addCookie(cookie1); // 인증에 성공했을경우 logged_in 에 1의 값을 셋팅
Cookie cookie2 = new Cookie("userid", userid);
response.addCookie(cookie2);
...
%>

//user_menu.jsp //사용자 검증이 필요한 페이지
<%
Cookie[] cookies = request.getCookies();
for(int i=0; i< cookies.length; i++)
Cookie thisCookie = cookie[i];
if(thisCookie.getName.equals("logged_in"))
String logged_in = thisCookie.getValue();
if(thisCookie.getName.equals("userid"))
String userid = thisCookie.getValue();

if(logged_in.equals("1"))
out.println("인증 성공: " + userid);

%>
```

(2) 안전한 프로그래밍 예

1) ASP

```
// login_ok.asp 사용자 인증 처리를 하는 스크립트
<%
// 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 처리 해야 되는 부분

If Session("logged_in") <> 1 Then
Session("logged_in") = 1 // 인증에 성공했을경우 logged_in 에 1의 값을 셋팅
Session("userid") = userid
Session("user_ip") = Request.Servervariables("REMOTE_ADDR")
End If
End If
...
%>

// user_menu.asp 사용자 검증이 필요한 페이지
<%
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
%>
```

2) PHP

```
//login_ok.php // 사용자 인증 처리를 하는 스크립트
<?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 실패시 종료

// 인증에 성공한 경우 처리 해야 되는 부분
if (!session_is_registered("logged_in"))

$logged_in = 1; // 인증에 성공했을경우 logged_in 에 1의 값을 셋팅
$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

...
?>

//user_menu.php // 사용자 검증이 필요한 페이지
<?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;

?>
```

3) JSP

```
<%@ page contentType="text/html;charset=euc-kr" %>
<%@ page import="java.util.*" %>
<%@ page import="java.sql.* " %>
// login_ok.jsp// 사용자 로그인 처리를 하는 스크립트
<%
// 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 처리 해야 되는 부분
session.putValue('logged_in',"1");
session.putValue('userid',userid);
session.putValue('user_ip',request.getRemoteAddr());
...
%>

//user_menu.jsp // 사용자 검증이 필요한 페이지
<%
//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.";

%>
```

라. 입력값 검증 처리(XSS)

사용자 입력으로 사용 가능한 문자들을 정해놓고, 그 문자들을 제외한 나머지 모든 문자들을 필터링 한다. 필터링 해야 하는 대상은 GET 질의 문자열, POST 데이터, 쿠키, URL, 그리고 일반적으로 브라우저와 웹 서버가 주고받는 모든 데이터를 포함한다.

아래는 특수문자에 대한 Entity 형태를 표시한 것이다.

[표 2] 특수문자 변경

변경 전	〈	〉	(	)	#	&
변경 후	&lt:	&gt:	&#40	&#41	&#35	&#38

- 게시판에서 HTML 포맷을 사용할 수 없도록 설정한다.
- 필요한 경우 모든 HTML을 사용하지 못하게 설정 후 필요한 HTML tag만 쓸 수 있도록 설정한다.

(1) 취약한 프로그래밍 예

1) ASP

```
<%
Set objDBConn = Server.CreateObject("ADODB.Connection") // 게시물 읽기
Set objRs = Server.CreateObject("ADODB.RecordSet")
objDBConn.Open "board", "user", "passwd"

query = "SELECT id, name, memo FROM board_tbl WHERE id=1"
objRs.Open query, objDBConn

memo = objRs("memo")
Response.write "게시물 내용--" & memo & "<BR>" // DB에서 게시판의 내용 출력
```

2) PHP

```
$query = "SELECT id, name, memo FROM board_tbl WHERE id=1"; // 게시물 읽기
$result = mysql_query($query, $connect);

while($row = mysql_fetch_array($result))
$name = $row[name];
$memo = $row[memo];
echo "게시물 내용-" . $memo . "<BR>\n"; // DB에서 게시판의 내용을 출력
```

3) JSP

```
Connection conn = DriverManager.getConnection(DB_URL, DB_USER,
DB_PASSWORD);
Statement stmt = conn.createStatement();
ResultSet rs=null;

String query = "SELECT memo FROM board_tbl WHERE id=1";
rs = stmt.executeQuery(query);
String memo = rs.getString(1);
out.print("게시물 내용-" + memo + "<BR>");
```

(2) 안전한 프로그래밍 예

1) ASP

```
If use_html Then // HTML tag를 사용하게 할 경우 부분 허용
memo = Server.HTMLEncode(memo) // HTML tag를 모두 제거

// 허용할 HTML tag만 변경
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>"
```

2) PHP

```
$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() 사용시 일부 한글이 깨어지는 현상이 발생 할 수 있음

$memo = str_replace("<", "&lt;", $memo);
$memo = str_replace(">", "&gt;", $memo);


echo "게시물 내용–" . $memo . "<BR>₩n";
```

3) JSP

```
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>");
```

마. 버퍼오버플로우

버퍼 오버플로우는 C언어에서 안전하지 않은 함수를 사용함으로 인해 발생되는 경우가 많다.

strcpy(), strcat(), sprintf(). vsprintf(), gets()와 같은 함수는 경계 값 체크를 하지 않으므로 strncpy(), strncat(), snprintf(), fget()과 같은 함수로 대체해야 한다. 또한 scanf 계열의 함수들은 위험하므로 최대한 입력받을 수 있는 스트링의 길이를 제한하여야 한다. realpath()나 getopt()과 같은 함수도 최소한의 PATH_MAX 바이트 길이를 정해주는 getwd() 함수를 사용하는 것이 안전하다.

(1) 취약한 프로그래밍 예

1) strcpy() 함수의 대체

```
void func(char *str) {
    char buffer[256];
    strcpy(buffer, str);
    return;
}
```

2) strcat() 함수의 대체

```
void func(char *str) {
    char buffer[256];
    strcat(buffer, str);
    return;
}
```

3) sprintf() 함수의 대체

```
void func(char *str) {
    char buffer[256];
    sprintf(buffer, "%s", str);
    return;
}
```

4) gets() 함수의 대체

```
void func(char *str) {
char buffer[256];
gets(buffer);
return;
}
```

5) scanf(), sscanf(), fscanf(), 함수의 대체

```
void func() {
    char buffer[256];
    int num;
    num = fscanf(stdio, "%s", buffer); char buffer[256];
    return;
}
```

6) C로 개발한 프로그램

```
#include <stdio.h>
#include <string.h>
#define FILENAME "/usr/local/apache/cgi-bin/counter.dat"

int main() {
    FILE *fp;
    int counter=0;
    char envc[255], *env;

    env = getenv("HTTP_USER_AGENT");

    printf("Content-Type: text/html \n\n");

    if (!env)
                    exit(1);

    strcpy(envc, env);
    strtok(envc," ");

    if((fp=fopen(FILENAME,"rt")) == NULL ) exit(1);
    fscanf(fp,"%d",&counter);
    fclose(fp);
    printf("<FONT size=2><B>VISIT</B>: %d / <B>BROWSER</B>:
    %s</FONT>\n",counter, envc);
```

```
    if((fp=fopen(FILENAME,"wt")) == NULL ) exit(1);
    fprintf(fp,"%d\n",counter+1);
    fclose(fp);

    return 0;
}
```

(2) 안전한 프로그래밍 예

1) strcpy() 함수의 대체

```
void func(char *str) {
    char buffer[256];
    strncpy(buffer, str, sizeof(buffer)-1);
    buffer[sizeof(buffer)-1] = 0;
    return;
}
```

2) strcat() 함수의 대체

```
void func(char *str) {
    char buffer[256];
    strncat(buffer, str, sizeof(buffer)-1);
    return;
}
```

3) sprintf() 함수의 대체

```
void func(char *str) {
    char buffer[256];
    if(snprintf(target, sizeof(target)-1, "%s", string) > sizeof(target)-1)
    /*....*/
    return;
}
```

4) gets() 함수의 대체

```
void func(char *str) {
    char buffer[256];
    fgets(buffer, sizeof(buffer)-1, stdin);
    return;
}
```

5) scanf(), sscanf(), fscanf(), 함수의 대체

```
void func() {
    char buffer[256];
    int num;
    num = fscanf(stdio, "%255s", buffer);
    return;
}
```

6) C로 개발한 프로그램

```
#include <stdio.h>
#include <string.h>
#define FILENAME "/usr/local/apache/cgi-bin/counter.dat"

int main() {
    FILE *fp;
    int counter=0;
    char envc[255], *env;

    env = getenv("HTTP_USER_AGENT");
    printf("Content-Type: text/html \n\n");

    if (!env)
                    exit(1);

    strncpy(envc, env, sizeof(envc)-1);
    strtok(envc," ");

    if((fp=fopen(FILENAME,"rt")) == NULL ) exit(1);
    fscanf(fp,"%d",&counter);
    fclose(fp);
    printf("<FONT size=2><B>VISIT</B>: %d / <B>BROWSER</B>:
    %s</FONT>\n",counter, envc);

    if((fp=fopen(FILENAME,"wt")) == NULL ) exit(1);
    fprintf(fp,"%d\n",counter+1);
    fclose(fp);

    return 0;
}
```

바. 쿼리 명령어 처리(SQL Injection)

- 사용자로부터 입력받은 변수로 SQL 쿼리 구문을 생성하는 CGI는 입력받은 변수를 체크하거나 변경하는 로직을 포함하고 있어야 한다.
- 입력받은 변수와 데이터 베이스 필드의 데이터형을 일치 시켜야 하고, 사용 중인 SQL 구문을 변경시킬 수 있는 특수문자가 포함되어 있는지 체크해야 한다.
- 검색 부분과 같이 클라이언트로부터 생성된 SQL 구문을 받는 부분이 있다면 이를 제거해야 한다.

(1) 취약한 프로그래밍 예

1) ASP

```
prodId = Request.QueryString("productId")

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
```

2) PHP

```
$query = "SELECT id, password, username FROM user_table WHERE id='$id'";
// 사용자로부터 입력받은 id 값을 사용자 table에서 조회
$result = OCIParse($conn, $query);
if (!OCIExecute($result))
echo "⟨META http-equiv=\"refresh\" content=\"0;URL=http://victim.com\"⟩"; // 메인 페이지로
redirect

OCIFetchInto($result, &$rows);
... 중략 ...
```

3) JSP

```
String sql = "SELECT * FROM user_table" + " WHERE id = " + response.getParameter("id")
+ " AND password = " + response.getParameter("password");

Class.forName("org.gjt.mm.mysql.Driver");
conn = DriverManager.getConnection(DB_URL, DB_USER, DB_PASSWORD);

stmt = conn.createStatement();
rs = stmt.executeQuery(query);

while(rs.next())
```

(2) 안전한 프로그래밍 예

1) ASP

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

2) PHP

```
$query = sprintf("SELECT id,password,username FROM user_table WHERE
id='%s';",addslashes($id));
// id변수를 문자형으로 받고, id변수의 특수문자를 일반문자로 변환한다.

// @ 로 php 에러 메시지를 막는다.
$result = @OCIParse($conn, $query);
if (!@OCIExecute($result))
error("SQL 구문 에러");
exit;

@OCIFetchInto($result,&$rows);
... 중략 ...
```

3) 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();
```

사. 첨부파일 설정(파일 업로드)

- 첨부 파일에 대한 검사는 반드시 Server Side Script에서 구현해야 한다.
- 첨부파일을 체크하여 특정 종류의 파일들만 첨부 가능하도록 하고 에러코드를 삽입하거나 첨부 파일을 처리하는 파일 업로드 프로그램(php, php3, cgi, html, jsp 등)에서 모든 실행 가능한 파일은 첨부할 수 없도록 한다
- 프로그램에서 필터링을 할 경우 단순히 파일이름 기준으로 점검하지 말고 확장자명에 대하여 검사하되 대소문자를 모두 검사하도록 한다.
- 너무 작거나 큰 파일을 처리하는 로직을 포함해야 하고, 임시 디렉토리에서 업로드 된 파일을 지우거나 다른 곳으로 이동시켜야 한다. 또한 폼에서 어떠한 파일도 선택되지 않았다면, 파일 업로드에 사용되는 변수를 초기화 시켜주어야 한다.
- 웹 서버 엔진 설정 시 업로드 된 디렉토리의 Server Side Script 언어의 실행 권한을 제거하고 업로드 된 파일이름을 임의로 변경하여 저장하는 것도 안전한 방법이다.

(1) 취약한 프로그래밍 예

1) ASP

```
<%
Set Up = Server.CreateObject("SiteGalaxyUpload.Form")

uploadPath = server.mappath(".") & "\upload\" // 업로드 디렉토리

Fname = Up("file1")
if Fname <> "" then // 파일 첨부가 되었으면
fileName=Mid(Fname,InstrRev(Fname,"\")+1) // 파일이름부분 추출
savePath = uploadPath & fileName

Set fso = CreateObject("Scripting.FileSystemObject")
Up("file1").SaveAs(savePath)
response.write(savePath & " 저장 완료")
else
response.write("Error")
end if

Set Up = nothing
%>
```

2) PHP

```
<?php
$uploaddir = '/var/www/uploads/';

$uploadfile = $uploaddir. $_FILES['userfile']['name'];

if(copy($_FILES['userfile']['tmp_name'], $uploadfile))
print "성공적으로 업로드 되었습니다.";
print_r($_FILES);
else
```

```
else
print "파일 업로드 실패";
print_r($_FILES);

?>
```

3) 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, new
DefaultFileRenamePolicy());
Enumeration formNames=multi.getFileNames(); // 폼의 이름 반환
String formName=(String)formNames.nextElement();
String fileName=multi.getFilesystemName(formName); // 파일의 이름 얻기

if(fileName == null)
out.print("Error");
  else
fileName=new String(fileName.getBytes("8859_1"),"euc-kr");
out.print("User Name : " + multi.getParameter("userName") + "<BR>");
out.print("Form Name : " + formName + "<BR>");
out.print("File Name : " + fileName);

  catch(Exception e)
out.print("Error");

%>
```

(2) 안전한 프로그래밍 예

1) 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

// 중복시에 파일이름부분을 변경하기 위해 분리를 한다
Farry=split(Filename,".") // .을 기준으로 분리
preFname=Farry(0) // 파일이름 앞부분
extFname=Farry(1) // 파일의 확장자

// 저장할 전체 path를 만든다, 파일이름을 구한다
Path2 = Path1 & Filename
saveFname=preFname & "." & extFname

Set fso = CreateObject("Scripting.FileSystemObject")
countNo = 0 // 파일 중복될경우 셋팅 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
%>
```

2) PHP

```
<?php
$uploaddir = '/var/www/uploads/';

//파일 사이즈가 0byte 보다 작거나 최대 업로드 사이즈보다 크면 업로드를 금지 시킨다.
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 금지 시킨다.
if (eregi("[^a-z0-9\._\-]",$_FILES['userfile']['name']))
print "파일 이름의 특수문자 체크";
exit;

//파일 확장자중 업로드를 허용할 확장자를 정의한다.
$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 사용한다. 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 한다.

$extension= strtolower($extension);
if (!( ereg($extension","hwp") || ereg($extension","pdf") || ereg($extension","jpg")) )
print "업로드 금지 파일 입니다";
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);

?>
```

3) 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"); // 한글인코딩
out.print("File Name : " + fileName);

  catch(Exception e)
```

아. 다운로드 설정

(1) 취약한 프로그래밍 예

1) 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

objStream.Open
objStream.Type = 1
```

```
objStream.LoadFromFile Server.MapPath("./upfiles/")&"\"& file // 서버 절대경로

download = objStream.Read
Response.BinaryWrite download

Set objStream = nothing // 객체 초기화
```

2) PHP

```
$dn_path = "/var/www/data/$up_dir/$dn_file_name";

// 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");
```

3) JSP

```
String UPLOAD_PATH= "/var/www/upload/";
String filename= response.getParameter("filename");
String filepathname = UPLOAD_PATH + filename;

// 파일 전송 루틴
response.setContentType("application/unknown; charset=euc-kr");
response.setHeader("Content-Disposition","attachment;filename=" + filename + ";");
response.setHeader("Content-Transfer-Encoding:" , "base64");

BufferedInputStream in = new BufferedInputStream(new
FileInputStream(filepathname));
```

(2) 안전한 프로그래밍 예

1) 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
%>
```

2) 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");
```

3) 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 유무등]
```

자. 개발 언어별 로그인 인증 프로그래밍 예

(1) ASP 예

1) login.html

```
〈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.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〉
```

2) login.asp

```
〈% 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 실패시 인증 페이지로 Redirect
Else
If Session("logged_in") 〈〉 1 Then // 인증된 사용자 인지 체크
Session("logged_in") = 1 // 인증에 성공했을경우 logged_in 에 1의 값을 셋팅
Session("user_id") = user_id // 사용자 ID 저장
Session("user_ip") = Request.Servervariables("REMOTE_ADDR") // IP 저장
End If
Response.redirect("/main.asp") // 인증 성공시 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 못했을경우
UserAuth = 0
Else
UserAuth = 1
End If

objRs.Close // DB 연결 해제
Set objRs = Nothing
objConn.Close
Set objConn = Nothing
End Function
```

3) main.asp

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

(2) PHP 예

1) login.html

```
<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>
```

2) login.php

```
<?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 부분
                header("Location: login.html");
                exit; // 인증 실패시 종료

// 인증에 성공한 경우 처리 해야 되는 부분
if (!session_is_registered("logged_in"))

$logged_in = 1; // 인증에 성공했을경우 logged_in 에 1의 값을 셋팅
```

```
$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;
?>
```

3) main.php

```
<?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;

?>
```

(3) JSP 예

1) login.html

```
<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.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〉
```

2) login.jsp

```
〈%@ 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); // Servlet 의 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 입력값 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"); // SESSION에 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 성공시 Main 페이지로 Redirect
else
response.sendRedirect("/login.html"); // 인증 실패시 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)

%>
```

3) main.jsp

```
<%@ 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 실패시 인증 페이지로 Redirect

%>
```