사외비

문서번호	20220302-V3.0.0

Web/API

개발보안 Guideline

2022. 03. 02

목 차

II. 세부항목 설명 4

1. 입/출력 값 검증 부재 4

1-1. XSS / CSRF 공격 가능성 4

1-2. 삽입(Injection) 공격 가능성 22

2. 취약한 파일처리 37

2-1. 악성코드파일 업로드 37

2-2. 중요 정보 파일 다운로드 가능성 43

3. 취약한 인증 및 세션 관리 47

4-1. 쿠키(Cookie) 및 웹 스토리지(Web Storage) 조작 가능성 47

4-2. 인증(세션 및 토큰) 값 안전성 설정 여부 54

4-3. 접근제어 우회 가능성 확인 62

4-4. 비인증 상태로 중요 page접근 가능성 68

4-5. 일반계정 권한 상승 가능성 68

표 목 차

[표] 2. XSS가 가능한 특수문자 목록 5
[표] 3. XSS Keyword 필터링 문자 목록 8
[표] 4. SQL Injection 필터링 문자 목록 23
[표] 5. 필터링 수행해야 하는 경우 24
[표] 6. 삭제해야 하는 프로시저 24
[표] 7. Command Injection 필터링 문자 목록 26
[표] 8. XPATH Injection 필터링 문자 목록 27
[표] 9. LDAP Injection 필터링 문자 목록 27
[표] 10. SSI Injection 필터링 특수문자 목록 28
[표] 11. 업로드 파일 필터링 문자 목록 38
[표] 12. 다운로드 파일 필터링 문자 목록 44

Ⅱ. 세부항목 설명

1. 입/출력 값 검증 부재

웹 애플리케이션은 어떻게 반응해야 하는지 결정하기 위해 입력 값을 주로 HTTP 요청으로 받아들입니다. 공격자는 URL, 쿼리 문자열, HTTP 헤더, 쿠키, HTML 폼 인자, HTML 히든(Hidden) 필드 등 모든 HTTP 요청을 변조할 수 있으며 이를 통해 사이트의 보안 메커니즘의 우회를 시도합니다. 흔히 발생하는 공격은 URL 강제접속, 악성스크립트 삽입, 버퍼 오버플로우, 포맷스트링, SQL/Command Injection, 히든(Hidden) 필드 조작, 악성코드 파일 업로드, 파일 다운로드 시도 등의 방법이 있습니다.

1-1. XSS / CSRF 공격 가능성

구분	내용
주요내용	XSS / CSRF 취약점은 공격자가 웹 애플리케이션을 사용해서 다른 최종 사용자에게 악성코드를 보내 사용자 의도와 상관없이 특정 행위를 발생시키는 취약점입니다. 악성코드는 JavaScript, VBScript, ActiveX, Flash 등 클라이언트 웹 브라우저에서 실행되는 코드들로 구성되어 있습니다. 최종 사용자들은 악성코드가 포함된 웹 사이트의 링크를 클릭하거나 이메일에 포함된 내용을 읽거나 BBS에 게시된 게시물을 클릭하는 것만으로도 사용자의 환경 설정사항을 변경하거나, 쿠키(cookie)를 가로채어 사용자 세션을 도용하거나 악성코드 유포 및 잘못된 광고물 게시 등에 악용될 수 있습니다. 또한, XML에서 CDATA를 사용하여 XML에서 파싱되지 말아야 될 특수문자를 삽입한 뒤 악성코드를 보내 사용자의 쿠키/세션 탈취 및 CSRF 공격등이 가능합니다. DOM 환경일 경우 사용자가 입력한 악성코드가 DOM 생성의 일부로 실행되어 서버와 관계없이 브라우저에서 발생하며, 악성 URL로의 리다이렉트, 사용자 인증 정보 탈취 등의 공격으로 이어질 수 있습니다.
대응방안	해당 취약점을 예방할 수 있는 최선의 방안은 모든 코드들을 상세히 검증하는 것입니다. 즉, 헤더, 쿠키, 질의문, 폼필드와 숨겨진 필드등과 같은 모든 파라미터들을 엄격한 규칙에 의해서 검증합니다. 또한 다음과 같이 왼쪽의 필드의 입력 데이터를 오른쪽의 필드로 변환해서 필터링 해야 합니다. ※ [표] 2 참조

From	To(숫자표현)	To(문자표현)
<	<	<
>	>	>
(	(	-
)	)	-
#	#	-
&	&	&
'	'	-
"	"	"

[표] 2. XSS가 가능한 특수문자 목록

HTML에 있는 모든 메타 캐릭터의 제거가 힘들 경우 허용하는 문자들(e.g. [A-Za-z0-9])만 사용되도록 하고 이외의 것은 배제하는 형태인 positive 필터링 모드를 사용하는 것을 권고 드립니다.
또한, 사이트 내 태그 사용이 불가피 한 경우 아래 [표] 3의 Keyword를 사용하지 못하도록 필터링하여 XSS 구문 공격에 대한 최소한의 대응을 해야 합니다.

※ 패스워드 등록/변경 또는 로그인시 패스워드 정책에 따라 특수문자 사용이 필요한 경우 필터링 적용에 대한 검토가 필요합니다.

※ Keyword 필터링은 대/소문자 구분 없이 적용하여야 하며 일괄 적용 시 오류가 발생할 수 있으니 서비스 및 운영상의 영향도를 확인하시어 적용해야 합니다.

XSS Keyword 필터링 문자열		
<script>	javascript	%3Cscript
JaVaScRiPt	ScRiPt%20%0a%0d	ja%0Av%0Aa%0As%0Ac%0Aript
script	vbscript	binding
allowscriptaccess	expression	applet
meta	xml	blink
link	style	embed
object	iframe	frame

frameset	background	layer
ilayer	bgsound	title
base	eval	innerHTML
charset	refresh	string
void	create	append
%3Ealert	alert	msgbox
document	cookie	href
nabort	@import	+ADw
+AD4	aim:	%0da=eval
xmlns:html	http-equiv=refresh	http-equiv="refresh"
xmlns:html=	<htmlxmln	list-style-image
x-scriptlet	echo(	0%0d%0a%00
moz-binding	res://	#exec
%u0	&#x	fromcharcode
firefoxurl	<br size=	wvs-xss
acunetix_wvs	lowsrc	dynsrc
behavior	activexobject	microsoft.xmlhttp
clsid:cafeefac-dec7-0000-0000-abcdeffedcba	application/npruntime-scriptable-plugin	deploymenttoolkit
onactivae	onafterprint	onafterupdate
onbefore	onbeforeactivate	onbeforecopy
onbeforecut	onbeforedeactivate	onbeforeeditfocus
onbeforepaste	onbeforeprint	onbeforeunload
onbeforeupdate	onblur	onbounce
oncellchange	onchange	onclick
oncontextmenu	oncontrolselect	oncopy
oncut	ondataavailable	ondatasetchanged
ondatasetcomplete	ondblclick	ondeactivate
ondrag	ondragend	ondragenter
ondragleave	ondragover	ondragstart
ondrop	onerror	onerrorupdate
onfilterchange	onfinish	onfocus

onfocusin	onfocusout	onhelp	
onkeydown	onkeypress	onkeyup	
onlayoutcomplete	onload	onlosecapture	
onmousedown	onmouseenter	onmouseleave	
onmousemove	onmouseout	onmouseover	
onmouseup	onmousewheel	onmove	
onmoveend	onmovestart	onpaste	
onpropertychange	onreadystatechange	onreset	
onresize	onresizeend	onresizestart	
onrowenter	onrowexit	onrowsdelete	
onrowsinserted	onscroll	onselect	
onselectionchange	onselectstart	onstart	
onstop	onsubmit	onunload	
₩";	onMessage	onRowDelete	
;//	onOffline	onRowInserted	
FSCommand	onOnline	onSeek	
onAbort	onOutOfSync	onStorage	
onActivate	onPause	onSyncRestored	
onBegin	onPopState	onTimeError	
onDragDrop	onProgress	onTrackChange	
onEnd	onRedo	onUndo	
onHashChange	onRepeat	onURLFlip	
onInput	onResume	seekSegmentTime	
onMediaComplete	onReverse	onRowsEnter	
onMediaError	java.lang.Runtime	getRuntime	
onwheel	onsearch	onpagehide	
onpageshow	oninvalid	oncanplay	
oncanplaythrough	oncuechange	ondurationchange	
onloadedmetadata	onloadstart	onseeked	
onplay	onplaying	onratechange	
onseeking	onstalled	onsuspend	
ontimeupdate	onvolumechange	onwaiting	
onemptied	ontoggle	video	
audio	details	onended	

onloadeddata	onloadedmetadata	

[표] 3. XSS Keyword 필터링 문자 목록

서비스 운영 시 특정 태그에 대한 사용이 필요한 경우 OWASP에서 제공하는 AntiSamy를 이용하여 허용할 태그 및 속성, CSS 값을 직접 정의하고 정의되지 않은 이외의 값은 필터링 처리를 하여 사용하도록 합니다.
해당 필터링은 antisamy.xml 파일의 정책에 기반하여 동작합니다.

※ AntiSamy는 JAVA에 한정되어 있으며, 필수파일은 최신버전을 유지해 주시기 바랍니다.

https://owasp.org/www-project-antisamy/
https://github.com/nahsra/antisamy
- JAVA적용 필수파일-
  1) Policy File: antisamy.xsd, antisamy.xml
  2) Library File: antisamy.jar, xercesImpl.jar, batik.jar, nekohtm.jar

추가적으로 서버가 생성하는 set-cookie에 **httponly 옵션**이 있다면 javascript의 document.cookie 메소드를 통해 쿠키정보를 브라우저로 획득할 수 없기 때문에 쿠키 생성시 **httponly 옵션**을 설정 해야 합니다.

※ httponly 옵션 설정

언어	설정방법
ASP .NET	**// Framework 2.0 이후** 세션 발급하는 코드 페이지 상단에 아래 내용 추가 HttpCookie cookie = new HttpCookie("LastVisit", DateTime.Now.ToString()); **cookie.HttpOnly = true;** cookie.Name = " cookie "; Response.AppendCookie(cookie); Response.Write(cookie.Name); **// MVC 모델 사용** web.config파일에 아래 내용 추가 <system.web> <httpCookies httpOnlyCookies="true" requireSSL="true" />

		MVC모델로 입력데이터를 출력할 경우, HTMLEncode 처리가 되어 출력이 되기 때문에 스크립트 구문이 실행되지 않지만, @Html.Raw()를 사용할 경우는 입력 값 검증을 수행해야 합니다.
	PHP	php.ini 파일에 **session.cookie_httponly = True** 추가 및 **allow_url_include = off** 설정
	JAVA	WEB-INF/web.xml 파일에 아래 내용 추가 <session-config> <cookie-config> **<http-only>true</http-only>** </cookie-config> <session-config>
	Node.js	쿠키 설정 페이지에 아래 내용 추가 **response.writeHead(200, {'Set-Cookie':['HttpOnly=HttpOnly; HttpOnly']});** 또는, app.use 설정 내 쿠키 설정 시 아래 내용 추가 **Cookie: { httpOnly: true }**
샘플소스	**ASP.NET sample source code** **// XSS Keyword 필터링** Public string XSSFilter(string strString) string keywords = "<script>,javascript,%3Cscript,JaVaScRiPt,ScRiPt%20%0a%0d,ja%0Av%0Aa%0As%0Ac%0Aript,script,vbscript,binding,allowscriptaccess,expression,applet,meta,xml,blink,link,style,embed,object,iframe,frame,frameset,background,layer,ilayer,bgsound,title,base,eval,innerHTML,charset,refresh,string,void,create,append,%3Ealert,alert,msgbox,document,cookie,href,nabort,@import,+ADw,+AD4,aim:,%0da=eval,xmlns:html,http-equiv=refresh,http-equiv="refresh",xmlns:html=,<htmlxmln,list-style-image,x-scriptlet,echo(,0%0d%0a%00,moz-binding,res://,#exec,%u0,&#x,fromcharcode,firefoxurl,<br size=,wvs-xss,acunetix_wvs,lowsrc,dynsrc,behavior,activexobject,microsoft.xmlhttp,clsid:cafeefac-dec7-0000-0000-abcdeffedcba,application/npruntime-scriptable-	

	plugin,deploymenttoolkit,onactivae,onafterprint,onafterupdate,onbefore,onbeforeactivate,onbeforecopy,onbeforecut,onbeforedeactivate,onbeforeeditfocus,onbeforepaste,onbeforeprint,onbeforeunload,onbeforeupdate,onblur,onbounce,oncellchange,onchange,onclick,oncontextmenu,oncontrolselect,oncopy,oncut,ondataavailable,ondatasetchanged,ondatasetcomplete,ondblclick,ondeactivate,ondrag,ondragend,ondragenter,ondragleave,ondragover,ondragstart,ondrop,onerror,onerrorupdate,onfilterchange,onfinish,onfocus,onfocusin,onfocusout,onhelp,onkeydown,onkeypress,onkeyup,onlayoutcomplete,onload,onlosecapture,onmousedown,onmouseenter,onmouseleave,onmousemove,onmouseout,onmouseover,onmouseup,onmousewheel,onmove,onmoveend,onmovestart,onpaste,onpropertychange,onreadystatechange,onreset,onresize,onresizeend,onresizestart,onrowenter,onrowexit,onrowsdelete,onrowsinserted,onscroll,onselect,onselectionchange,onselectstart,onstart,onstop,onsubmit,onunload,₩";,onMessage,onRowDelete,;//,onOffline,onRowInserted,FSCommand,onOnline,onSeek,onAbort,onOutOfSync,onStorage,onActivate,onPause,onSyncRestored,onBegin,onPopState,onTimeError,onDragDrop,onProgress,onTrackChange,onEnd,onRedo,onUndo,onHashChange,onRepeat,onURLFlip,onInput,onResume,seekSegmentTime,onMediaComplete,onReverse,onRowsEnter,onMediaError,java.lang.Runtime,getRuntime,onwheel,onpageshow,oncanplaythrough,onloadedmetadata,onplay,onseeking,ontimeupdate,onemptied,audio,onsearch,oninvalid,oncuechange,onloadstart,onplaying,onstalled,onvolumechange,ontoggle,details,onpagehide,oncanplay,ondurationchange,onseeked,onratechange,onsuspend,onwating,video"
	string[] arrKeywords = keywords.ToLower().Split(',');
	for (int i = 0; i < arrKeywords.Length; i++){
	strString = strString.ToLower().Replace(arrKeywords[i], "_" + arrKeywords[i] + "_");
	}
	return strString;
	}

```
==============================================

// XSS 특수 문자(태그) 필터링
public string clearXSS(string strString)
{
    strString = strString.Replace( "<", "&lt;" );
    strString = strString.Replace( ">", "&gt;" );
    strString = strString.Replace( "(", "&#40;" );
    strString = strString.Replace( ")", "&#41;" );
    strString = strString.Replace( "#", "&#35;" );
    strString = strString.Replace( "&", "&#38;" );
    strString = strString.Replace( "'", "&#39;" );
    strString = strString.Replace( "₩"", "&quot;" );

    return strString;
}

==============================================

// XSS 특수 문자 HTML Encoding
public string clearXSS(string strString)
{
    // AntiXSS 라이브러리를 사용하여 HTML Encoding 처리
    string encString = AntiXss.HtmlEncode(strString);

    // 추가 특수기호 Encoding 처리
    encString = encString.Replace( "(", "&#40;" );
    encString = encString.Replace( ")", "&#41;" );

    return encString;
}
```

ASP sample source code

```
// XSS Keyword 필터링
<%
Function XSSFilter(strString)
  filstr="<script>,javascript,%3Cscript,JaVaScRiPt,ScRiPt%20%0a%0d,ja%0Av%0Aa%0As%0Ac%0Aript,script,vbscript,binding,allowscriptaccess,expression,applet,meta,xml,blink,link,style,embed,object,iframe,frame,frameset,background,layer,ilayer,bgsound,title,base,eval,innerHTML,charset,refresh,string,void,create,append,%3Ealert,alert,msgbox,document,cookie,href,nabort,@import,+ADw,+AD4,aim:,%0da=eval,xmlns:html,http-equiv=refresh,http-equiv="refresh",xmlns:html=,<htmlxmln,list-style-image,x-scriptlet,echo(,0%0d%0a%00,moz-binding,res://,#exec,%u0,&#x,fromcharcode,firefoxurl,<br size=,wvs-xss,acunetix_wvs,lowsrc,dynsrc,behavior,activexobject,microsoft.xmlhttp,clsid:cafeefac-dec7-0000-0000-abcdeffedcba,application/npruntime-scriptable-plugin,deploymenttoolkit,onactivae,onafterprint,onafterupdate,onbefore,onbeforeactivate,onbeforecopy,onbeforecut,onbeforedeactivate,onbeforeeditfocus,onbeforepaste,onbeforeprint,onbeforeunload,onbeforeupdate,onblur,onbounce,oncellchange,onchange,onclick,oncontextmenu,oncontrolselect,oncopy,oncut,ondataavailable,ondatasetchanged,ondatasetcomplete,ondblclick,ondeactivate,ondrag,ondragend,ondragenter,ondragleave,ondragover,ondragstart,ondrop,onerror,onerrorupdate,onfilterchange,onfinish,onfocus,onfocusin,onfocusout,onhelp,onkeydown,onkeypress,onkeyup,onlayoutcomplete,onload,onlosecapture,onmousedown,onmouseenter,onmouseleave,onmousemove,onmouseout,onmouseover,onmouseup,onmousewheel,onmove,onmoveend,onmovestart,onpaste,onpropertychange,onreadystatechange,onreset,onresize,onresizeend,onresizestart,onrowenter,onrowexit,onrowsdelete,onrowsinserted,onscroll,onselect,onselectionchange,onselectstart,onstart,onstop,onsubmit,onunload,₩";,onMessage,onRowDelete,;//,onOffline,onRowInserted,FSCommand,onOnline,onSeek,onAbort,onOutOfSync,onStorage,onActivate,onPause,onSyncRestored,onBegin,onPopState,onTimeError,onDragDrop,onProgress,onTrackChange,onEnd,onRedo,onUndo,onHashChange,onRepeat,onURLFlip,onInput,onResume,seekSegmentTime,onMediaComplete,onReverse,onRowsEnter,onMediaError,java.lang.Runtime,getRuntime,onwheel,onpageshow,oncanplaythrough,onloadedmetadata,onplay,ons
```

```
eeking,ontimeupdate,onemptied,audio,onsearch,oninvalid,oncuechange,onl
oadstart,onplaying,onstalled,onvolumechange,ontoggle,details,onpagehide
,oncanplay,ondurationchange,onseeked,onratechange,onsuspend,onwating
,video"

  filstr = Lcase(Replace(filstr," ",""))

  if(filstr <>"") Then
    fillist = split(filstr,",")
    for each f in fillist
      strString = Replace(Lcase(strString), f, "_"&f&"_")
    next
  End if
  XSSFilter = strString
End Function
// XSS 특수 문자 필터링
Function clearXSS(strString)
  avatag="p,br"   '허용할 태그 리스트(white list)

  strString = replace(strString, "<", "& l t ;")
  strString = replace(strString, ">", "& g t ;")
  strString = replace(strString, "₩0", "")

  '허용할 태그를 지정한 경우
  if (avatag <> "") Then
    taglist = split(avatag, ",")
    '허용할 태그 원상태로 변환
    for each p in taglist
      strString = replace(strString, "& l t ;"&p&" ", "<"&p&" ", 1, -1, 1)
      strString = replace(strString, "& l t ;"&p&"& g t ;", "<"&p&">", 1, -
1, 1)
      strString = replace(strString, " "&p&"& g t ;", " "&p&">", 1, -1, 1)
      strString = replace(strString, " "&p&"&gt", " "&p&">/", 1, -1, 1)
      strString = replace(strString, "& l t ;/"&p&" ", "</"&p&" ", 1, -1, 1)
    next
```

```
  End If
  clearXSS = strString
End Function
%>
```

PHP sample source code

```
// XSS Keyword 필터링
<?php
function XSSfilter($str) {
  $filstr="<script>,javascript,%3Cscript,JaVaScRiPt,ScRiPt%20%0a%0d,ja%0Av%0Aa%0As%0Ac%0Aript,script,vbscript,binding,allowscriptaccess,expression,applet,meta,xml,blink,link,style,embed,object,iframe,frame,frameset,background,layer,ilayer,bgsound,title,base,eval,innerHTML,charset,refresh,string,void,create,append,%3Ealert,alert,msgbox,document,cookie,href,nabort,@import,+ADw,+AD4,aim:,%0da=eval,xmlns:html,http-equiv=refresh,http-equiv="refresh",xmlns:html=,<htmlxmln,list-style-image,x-scriptlet,echo(,0%0d%0a%00,moz-binding,res://,#exec,%u0,&#x,fromcharcode,firefoxurl,<br size=,wvs-xss,acunetix_wvs,lowsrc,dynsrc,behavior,activexobject,microsoft.xmlhttp,clsid:cafeefac-dec7-0000-0000-abcdeffedcba,application/npruntime-scriptable-plugin,deploymenttoolkit,onactivae,onafterprint,onafterupdate,onbefore,onbeforeactivate,onbeforecopy,onbeforecut,onbeforedeactivate,onbeforeeditfocus,onbeforepaste,onbeforeprint,onbeforeunload,onbeforeupdate,onblur,onbounce,oncellchange,onchange,onclick,oncontextmenu,oncontrolselect,oncopy,oncut,ondataavailable,ondatasetchanged,ondatasetcomplete,ondblclick,ondeactivate,ondrag,ondragend,ondragenter,ondragleave,ondragover,ondragstart,ondrop,onerror,onerrorupdate,onfilterchange,onfinish,onfocus,onfocusin,onfocusout,onhelp,onkeydown,onkeypress,onkeyup,onlayoutcomplete,onload,onlosecapture,onmousedown,onmouseenter,onmouseleave,onmousemove,onmouseout,onmouseover,onmouseup,onmousewheel,onmove,onmoveend,onmovestart,onpaste,onpropertychange,onreadystatechange,onreset,onresize,onresizeend,onresizestart,onrowenter,onrowexit,onrowsdelete,onrowsinserted,onscroll,onselect,onselectionchange,onselectstart,onstart,onstop,onsubmit,onunload,₩";,onMessage,onRowDelete,;//,onOffline,onRowInserted,FSCommand,onOnline,onSeek,onAbort,onOutOfSync,onSto
```

```
rage,onActivate,onPause,onSyncRestored,onBegin,onPopState,onTimeError,
onDragDrop,onProgress,onTrackChange,onEnd,onRedo,onUndo,onHashCh
ange,onRepeat,onURLFlip,onInput,onResume,seekSegmentTime,onMediaC
omplete,onReverse,onRowsEnter,onMediaError,java.lang.Runtime,getRunti
me,onwheel,onpageshow,oncanplaythrough,onloadedmetadata,onplay,ons
eeking,ontimeupdate,onemptied,audio,onsearch,oninvalid,oncuechange,onl
oadstart,onplaying,onstalled,onvolumechange,ontoggle,details,onpagehide
,oncanplay,ondurationchange,onseeked,onratechange,onsuspend,onwating
,video";   //필터링 할 문자열

   $str = strtolower ($str);
   $filstr = str_replace(" ","",strtolower($filstr));
      if ($filstr != "") {
         $otag = explode (",", $filstr);
         for ($i = 0;$i < count($otag);$i++) {
            $str = str_replace($otag[$i], "_".$otag[$i]."_", $str);
         }
      }
   return $str;
}

//XSS 특수 문자(태그) 필터링
function clearXSS($str){
   $avatag = "p,br"; //허용할 태그 리스트(화이트 리스트)
   $str = str_replace(array("<",">"),array("&lt;","&gt;"), $str);

   //허용할 태그를 지정한 경우
   if ($avatag != "") {
      $otag = explode (",", $avatag);

      //허용할 태그 원상태로 변환
      for ($i = 0;$i < count($otag);$i++) {
         $str = str_replace("&lt;".$otag[$i]." ", "<".$otag[$i]." ", $str);
         $str = str_replace("&lt;".$otag[$i]."&gt;", "<".$otag[$i].">", $str);
         $str = str_replace(" "+$otag[$i]."&gt;", " ".$otag[$i].">", $str);
```

```
        $str = str_replace($otag[$i]."/&gt;", $otag[$i]."/>", $str);
        $str = str_replace("&lt;/".$otag[$i], "</".$otag[$i], $str);
      }
      return $str;
  }
}
?>
```

JSP sample source code

// XSS Keyword 필터링

```
<%
public String xssFilter(String str) {
  String filstr =
"<script>,javascript,%3Cscript,JaVaScRiPt,ScRiPt%20%0a%0d,ja%0Av%0Aa
%0As%0Ac%0Aript,script,vbscript,binding,allowscriptaccess,expression,appl
et,meta,xml,blink,link,style,embed,object,iframe,frame,frameset,background,
layer,ilayer,bgsound,title,base,eval,innerHTML,charset,refresh,string,void,cre
ate,append,%3Ealert,alert,msgbox,document,cookie,href,nabort,@import,+
ADw,+AD4,aim:,%0da=eval,xmlns:html,http-equiv=refresh,http-
equiv="refresh",xmlns:html=,<htmlxmln,list-style-image,x-
scriptlet,echo(,0%0d%0a%00,moz-
binding,res://,#exec,%u0,&#x,fromcharcode,firefoxurl,<br size=,wvs-
xss,acunetix_wvs,lowsrc,dynsrc,behavior,activexobject,microsoft.xmlhttp,clsi
d:cafeefac-dec7-0000-0000-abcdeffedcba,application/npruntime-
scriptable-
plugin,deploymenttoolkit,onactivae,onafterprint,onafterupdate,onbefore,on
beforeactivate,onbeforecopy,onbeforecut,onbeforedeactivate,onbeforeeditf
ocus,onbeforepaste,onbeforeprint,onbeforeunload,onbeforeupdate,onblur,
onbounce,oncellchange,onchange,onclick,oncontextmenu,oncontrolselect,
oncopy,oncut,ondataavailable,ondatasetchanged,ondatasetcomplete,ondbl
click,ondeactivate,ondrag,ondragend,ondragenter,ondragleave,ondragover,
ondragstart,ondrop,onerror,onerrorupdate,onfilterchange,onfinish,onfocus,
onfocusin,onfocusout,onhelp,onkeydown,onkeypress,onkeyup,onlayoutco
mplete,onload,onlosecapture,onmousedown,onmouseenter,onmouseleave,
onmousemove,onmouseout,onmouseover,onmouseup,onmousewheel,onm
ove,onmoveend,onmovestart,onpaste,onpropertychange,onreadystatechan
```

```
ge,onreset,onresize,onresizeend,onresizestart,onrowenter,onrowexit,onrows
delete,onrowsinserted,onscroll,onselect,onselectionchange,onselectstart,on
start,onstop,onsubmit,onunload,₩";,onMessage,onRowDelete,;//,onOffline,
onRowInserted,FSCommand,onOnline,onSeek,onAbort,onOutOfSync,onSto
rage,onActivate,onPause,onSyncRestored,onBegin,onPopState,onTimeError,
onDragDrop,onProgress,onTrackChange,onEnd,onRedo,onUndo,onHashCh
ange,onRepeat,onURLFlip,onInput,onResume,seekSegmentTime,onMediaC
omplete,onReverse,onRowsEnter,onMediaError,java.lang.Runtime,getRunti
me,onwheel,onpageshow,oncanplaythrough,onloadedmetadata,onplay,ons
eeking,ontimeupdate,onemptied,audio,onsearch,oninvalid,oncuechange,onl
oadstart,onplaying,onstalled,onvolumechange,ontoggle,details,onpagehide
,oncanplay,ondurationchange,onseeked,onratechange,onsuspend,onwating
,video"; //필터링 할 문자열

   filstr = filstr.replaceAll(" ","");
   if (!str.equals("")) {
      String [] st = filstr.split(",");
      for( int x = 0; x < st.length; x++ ) {
         str = str.replaceAll("(?i)"+st[x], "_"+st[x]+"_");
      }
   }
   return str;
}

//XSS 특수 문자 필터링
public String clearXSS(String str) {
   String avatag = "p,br"; //허용할 태그 리스트(화이트 리스트)
      str = str.replaceAll("<","&lt;");
      str = str.replaceAll(">","&gt;");
      str = str.replaceAll("₩0"," ");

      //허용할 태그를 지정할 경우
      if (!avatag.equals("")) {
         avatag.replaceAll(" ","");
         String st[] = avatag.split(",");
```

```
        //허용할 태그 원상태로 변환
        for( int i = 0; i < st.length; i++ ) {
          str = str.replaceAll("&lt;"+st[i]+" ", "<"+st[i]+" ");
          str = str.replaceAll("&lt;"+st[i]+"&gt;", "<"+st[i]+">");
          str = str.replaceAll(" "+st[i]+"&gt;", " "+st[i]+">");
          str = str.replaceAll(st[i]+"/&gt;", st[i]+"/>");
          str = str.replaceAll("&lt;/"+st[i], "</"+st[i]);
        }
      }
    return str;
}
%>
```

// AntiSamy 필터링

/* WEB-INF/web.xml 필터 선언 */

```
<filter>
    <filter-name>AntiXssFilter</filter-name>
    <filter-class>test.antisamy.AntiXssFilter</filter-class>
 </filter>
 <filter-mapping>
    <filter-name>AntiXssFilter</filter-name>
    <url-pattern>/test/*</url-pattern>
</filter-mapping>
```

/* antisamy-1.5.8.xml 사용/불가능 태그 정의 */

```
        ..................................................................
<!-- Tags related to JavaScript -->
<tag name="script" action="remove"/>
<tag name="noscript" action="remove"/>

<!-- Frame & related tags -->
<tag name="iframe" action="remove"/>
<tag name="frameset" action="remove"/>
<tag name="frame" action="remove"/>
```

```
    <tag name="noframes" action="remove"/>
        <!-- CSS related tags -->
    <tag name="style" action="remove"/>

      <!-- All reasonable formatting tags -->
    <tag name="p" action="validate">
        <attribute name="align"/>
    </tag>
    <tag name="div" action="validate"/>
    <tag name="i" action="validate"/>
    <tag name="b" action="validate"/>
    <tag name="em" action="validate"/>
    <tag name="blockquote" action="validate"/>
    <tag name="tt" action="validate"/>
    <tag name="strong" action="validate"/>
```

/* AntiXssFilter.java 정책 적용 */

```
import org.owasp.validator.html.*;   // Import AntiSamy

public AntiXssFilter() {
   String POLICY_FILE = "antisamy-1.5.8.xml"; // Policy file 경로
Policy policy = Policy.getInstance(POLICY_FILE);   // Policy 객체생성
   AntiSamy as = new AntiSamy(policy);   // AntiSamy 객체생성
}

private String filterString(AntiSamy as, String AntiParam) {
   CleanResults cr = as.scan(AntiParam,AntiSamy.DOM);   // Scan
AntiParam
return cr.getCleanHTML();
}
```

node.js sample source code

```
// XSS Keyword 필터링
module.exports.cleanXss = function(str){
  const keyword = "<script>,javascript,%3Cscript,JaVaScRiPt,
  ScRiPt%20%0a%0d,ja%0Av%0Aa%0As%0Ac%0Aript,script,vbscript,binding,allowscriptaccess,expression,applet,meta,xml,blink,link,style,embed,object,iframe,frame,frameset,background,layer,ilayer,bgsound,title,base,eval,innerHTML,charset,refresh,string,void,create,append,%3Ealert,alert,msgbox,document,cookie,href,nabort,@import,+ADw,+AD4,aim:,%0da=eval,xmlns:html,http-equiv=refresh,http-equiv="refresh",xmlns:html=,<htmlxmln,list-style-image,x-scriptlet,echo(,0%0d%0a%00,moz-binding,res://,#exec, %u0, &#x,fromcharcode,firefoxurl,<br size=,wvs-xss,acunetix_wvs,lowsrc,dynsrc,behavior,activexobject,microsoft.xmlhttp,clsid:cafeefac-dec7-0000-0000-abcdeffedcba,application/npruntime-scriptable-plugin,deploymenttoolkit,onactivae,onafterprint,onafterupdate,onbefore,onbeforeactivate,onbeforecopy,onbeforecut,onbeforedeactivate,onbeforeeditfocus,onbeforepaste,onbeforeprint,onbeforeunload,onbeforeupdate,onblur,onbounce,oncellchange,onchange,onclick,oncontextmenu,oncontrolselect,oncopy,oncut,ondataavailable,ondatasetchanged,ondatasetcomplete,ondblclick,ondeactivate,ondrag,ondragend,ondragenter,ondragleave,ondragover,ondragstart,ondrop,onerror,onerrorupdate,onfilterchange,onfinish,onfocus,onfocusin,onfocusout,onhelp,onkeydown,onkeypress,onkeyup,onlayoutcomplete,onload,onlosecapture,onmousedown,onmouseenter,onmouseleave,onmousemove,onmouseout,onmouseover,onmouseup,onmousewheel,onmove,onmoveend,onmovestart,onpaste,onpropertychange,onreadystatechange,onreset,onresize,onresizeend,onresizestart,onrowenter,onrowexit,onrowsdelete,onrowsinserted,onscroll,onselect,onselectionchange,onselectstart,onstart,onstop,onsubmit,onunload,₩";,onMessage,onRowDelete,;//,onOffline,onRowInserted,FSCommand,onOnline,onSeek,onAbort,onOutOfSync,onStorage,onActivate,onPause,onSyncRestored,onBegin,onPopState,onTimeError,onDragDrop,onProgress,onTrackChange,onEnd,onRedo,onUndo,onHashChange,onRepeat,onURLFlip,onInput,onResume,seekSegmentTime,onMediaComplete,onReverse,onRowsEnter,onMediaError,java.lang.Runtime,getRuntime,onwheel,onpageshow,oncanplaythrough,onloadedmetadata,onplay,onseeking,ontimeupdate,onemptied,audio,onsearch,oninvalid,oncuechange,onl
```

```
oadstart,onplaying,onstalled,onvolumechange,ontoggle,details,onpagehide
,oncanplay,ondurationchange,onseeked,onratechange,onsuspend,onwating
,video,onended,onloadeddata,onloadedmetadata,onwaiting ";
  const spkeyword = keyword.split(",");
  str = str.replace(/(₩s*)/g, "");
  let arr = "";
  for(let i=0; i < spkeyword.length; i++){
    arr= new RegExp(spkeyword[i], "gi");
    str = str.replace(arr,"_");
  }
  return str;
}
//XSS 특수 문자 필터링
module.exports.tagFilter = function(str) {
  str = str.replace(/</g,"&lt;");
  str = str.replace(/>/g,"&gt;");
  str = str.replace(/₩(/g,"&#40;");
  str = str.replace(/₩)/g,"&#41;");
  str = str.replace(/&/g,"&#38;");
  str = str.replace(/#/g,"&#35;");
  str = str.replace(/₩"/g,"&quot;");
  str = str.replace(/₩'/g,"&#39;");
  return str;
}
```

1-2. 삽입(Injection) 공격 가능성

구분	내용
주요내용	해당 항목은 SQL Injection 및 Command Injection 등의 웹 애플리케이션 또는 응용프로그램에서 발생할 수 있는 모든 삽입(Injection) 공격 취약점을 설명합니다. **- SQL Injection** 웹 애플리케이션이 Database로 사용자가 입력 가능한 변수를 전달할 때 해당 변수 값의 체크 없이 Database로 전달하여 발생하는 문제입니다. 특정 Database Query를 이용하여 인증을 위한 SQL Query를 비정상적으로 만들어, 아이디나 패스워드가 맞지 않음에도 불구하고 정상적으로 인증한 것처럼 우회할 수 있으며 Database Query시에 웹 애플리케이션에서의 정상적인 Query에 공격자가 원하는 Query를 더하여 특정 테이블의 값을 질의할 수 있으므로 결과적으로 Database내 모든 정보를 유출시키거나 변조할 수 있습니다. **- Command Injection** 시스템 명령어를 호출하는 파라미터의 인자 값을 그대로 시스템 명령에 사용하는 경우 공격자는 " **;** " 또는 " **|** " 연산자를 통해 임의의 시스템 명령어를 삽입하여 의도하지 않은 시스템 명령어를 실행시켜 권한 변경, 시스템 동작, 운영 등 악영향을 미칠 수 있습니다. 또한, **개발 시 취약한 함수[ex. eval(), function(), exec() 등]를 사용할 경우** 임의의 시스템 명령어를 삽입하여 시스템 권한을 획득하거나 중요정보파일이 유출될 가능성이 있습니다. **- XPATH Injection** XPATH는 XML 문서의 노드로부터 정보를 질의 하기 위해 이용되며 SQL Injection 공격과 유사하게, XPATH에 공격자가 비정상적인 코드를 삽입하여 XML 문서에 저장된 사용자 인증을 우회하거나 정보 추출이 가능한 취약점입니다. **- LDAP Injection** 입력 데이터를 LDAP 쿼리로 수행하고 해당 값에 대한 적절한 필터링 및 유효성 검증을 하지 않아, 공격자가 구문을 변조하여 LDAP 트리의 수정/삭제, 개인정보 획득 및 임의 명령 실행 등을 가능하게 하는 취약점입니

다.

- XML eXternal Entities (XXE)

웹 애플리케이션 내 XML 포맷 형식으로 브라우저와 서버 간의 데이터를 주고 받을 때 사용자로부터 전달된 파라미터 값(사용자 지정 XML 엔티티)을 별도 검증 없이 사용할 경우 공격자가 XML DTD(Document Type Definition) 특성을 이용해 대상 서버 내 파일 조회, 공유, 포트 스캔, 원격 코드 실행 등의 공격을 할 수 있습니다.

XXE 예제

```
<!DOCTYPE security [<!ENTITY xxe SYSTEM "file:///etc/passwd">]>
<node>&xxe;</node>
```

- SSI (Server Side Includes) Injection

SSI는 CGI 프로그램이나 다른 동적인 기술을 통해 페이지 전체를 만들지 않고, 일부 동적으로 생성한 내용을 추가하기 위해 만들어진 기능으로 HTML 문서 내 입력 받은 변수 값을 서버 측에서 처리하는 경우에 악의적인 코드를 삽입하여 임의의 명령을 실행시키거나 중요정보를 탈취할 수 있는 취약점입니다. 주로 웹 서비스 내 SSI를 사용할 경우 .shtml 확장자의 형태로 사용됩니다.

대응방안

- SQL Injection (Query 유효성 검증)

쿼리문 위조/변조를 통해 잘못 해석함에서 발생하는 문제입니다. 이를 해결하기 위해서는 쿼리문 생성시에 입력되는 모든 값에 대한 유효성 검사를 수행해야 합니다. ‘, ‘’ 문자를 ‘’, ‘’’’로 변경해 주거나 공백으로 처리해야 합니다. 또한 숫자는 숫자만 입력되게 처리해야 합니다.

※ [표] 4,5 참조

필터링 문자		
'	)	union
"	sp_	--
=	xp_	#
<	select	;
>	drop	cmdshell
(	₩	%

[표] 4. SQL Injection 필터링 문자 목록

※ 서비스 및 운영상 영향도를 고려하시어 검토 후 저장 프로시저를 삭제하여 주시기 바랍니다.

다음과 같은 경우에 모든 문자열 안에서 싱글 쿼트, 더블 쿼트, 슬래쉬, 백슬래쉬, 세미 콜론, NULL 같은 확장된 문자, 캐리지 리턴, 뉴라인 등과 같은 문자를 필터링 합니다.

필터링을 수행해야 하는 경우
사용자로부터의 입력
URL에 포함되어 있는 파라미터
쿠키 또는 세션에 포함된 값

[표] 5. 필터링 수행해야 하는 경우

※ 패스워드 등록/변경 또는 로그인시 패스워드 정책에 따라 특수문자 사용이 필요한 경우 필터링 적용에 대한 검토가 필요합니다.

사용하는 값이 숫자라면 SQL 문으로 사용하기 전에 그것을 정수로 변환하거나 또는 정수인지를 확인하기 위해서 ISNUMERIC을 사용할 수도 있습니다.

MS_SQL의 경우 사용하지 않는 Master Table 내의 아래 프로시저들을 삭제해야만 합니다.

삭제해야 할 프로시저
xp_cmdshell
xp_stratmail
xp_sendmail
xp_grantlogin
sp_makewebtask
xp_regread
xp_regwrite

[표] 6. 삭제해야 하는 프로시저

웹 애플리케이션 언어로 Java를 사용한다면 PreparedStatement 클래스를

사용하여야 하며 매개변수 값은 setString 등과 같은 setXXX 함수를 사용하여 처리하는 것이 바람직합니다.

iBatis/MyBatis의 Data Map을 사용하면 SQL문에서 동적 매개변수를 지정할 수 있으며 일반적으로 # 형태로 연결하면 PreparedStatement 형태로 쿼리문이 전달되기 때문에 SQL Injection의 공격대응이 가능하나, $ 형태로 연결할 경우 사용자 입력 값을 쿼리에 그대로 대입시키기 때문에 부주의하게 사용하면 인젝션 취약점에 노출될 수 있습니다.

WHERE 절에서의 LIKE 검색 시 #을 이용하면 PreparedStatement 형식으로 변환되며 '%?%'가 되어 오류를 발생하기 때문에 LIKE 검색 구문 내에서 $를 이용하는 경우가 있습니다. 이러한 경우 역시 $에 공격 구문을 넣어 SQL Injection 공격이 가능합니다. LIKE 구문에서는 아래와 같이 각 Database에 맞게 SQL 와일드 문자를 포함시킬 수 있습니다.
※ 아래 샘플소스 (iBatis/MyBatis sample source code) 참조

웹 애플리케이션과 DB 연동 시에는 DB 계정에 웹 애플리케이션 구동을 위한 최소한의 권한만 부여하여 전체 데이터가 취약점에 노출되는 것을 방지할 수 있습니다.

- Command Injection (Command 유효성 검증)
웹 인터페이스를 통해 서버 내부로 시스템 명령어를 전달시키지 않도록 애플리케이션을 구성하고 외부에서 전달되는 값을 검증 없이 시스템 내부 명령어로 사용하지 않아야 합니다.
입력에 따라 명령어 직접 호출이 필요한 경우에는 명령어 호출 시 필요한 값들을 미리 지정해 놓고 입력에 따라 선택하여 사용해야 하며 입력된 값들에 대한 검증을 수행해야 합니다. ※ [표] 7 참조

필터링 문자
&
|
;
'
"
[

]
=

[표] 7. Command Injection 필터링 문자 목록

참고)

& - 윈도우 명령어 해석기에서 첫 번째 명령이 성공했을 경우만 두 번째 명령어를 실행

| - 첫 번째 명령어가 성공하는지에 상관없이 두 번째 명령어를 실행

` - 쉘 해석기가 명령어를 해석하다 역 작은따옴표(`) 내에 포함된 명령어를 만나면 (예, `ls -al`) 기존 명령어를 계속 실행하기 전에 역 작은따옴표로 둘러싸인 명령어를 먼저 실행

eval() 함수는 JSON 형태의 문자열을 자바스크립트 객체로 변환하는 역할을 하며 변환 시 보안 검사를 하지 않으므로, JSON문자열을 개체로 변환하는 JSON.parse() 함수로 대체하여 사용하거나, 입력 값에 대한 유효성 검사가 필요합니다.

예제)

```
<script type="text/javascript">
  var jsonStr = '{"no":1, "name":"홍길동"}';

  // eval("var obj = ("+ jsonStr +")" );

  // JSON.parse 함수 사용
  var obj = JSON.parse(jsonStr);

  document.write(obj.no + ', ');
  document.write(obj.name);
</script>
```

JSON 파서 사용 시 js를 다운받아 추가해주어야 합니다.

http://www.json.org/js.html

- XPATH Injection (XPATH Query 유효성 검증)

SQL Injection과 마찬가지로 XPATH 쿼리에 입력되는 모든 값에 대한 유효성 검사를 수행해야 합니다. ※ [표] 8 참조

필터링 문자
'
=
[
]
(
)
,
*
:
/

[표] 8. XPATH Injection 필터링 문자 목록

- LDAP Injection (LDAP Query 유효성 검증)

LDAP 쿼리에 입력되는 값을 화이트리스트로 지정하여 허용된 값만 사용할 수 있도록 하고 쿼리에 영향을 주는 특수문자에 대해 유효성 검증을 해야 합니다. ※ 아래 표 참조

필터링 문자			
'	"	--	#
(	=	*/	/*
+	<	>	user_tables
user_table_columns		table_name	Syscolumns
column_name		union	select
insert	drop	update	and
or	If	join	substring
from	where	declare	substr
openrowset	xp_	sysobject	%
*	;	&	|

[표] 9. LDAP Injection 필터링 문자 목록

- XML eXternal Entities (XXE)

가능한 XML 보다 안전한 JSON 형태의 파일 포맷으로 서버와 통신하시

기 권고 드리며 사용자로부터 전달되는 입력 값에 대해 서버 측에서 유효성 검사(화이트리스트)를 설정해 악의적 공격에 대해 필터링 하시기 바랍니다.
또한, XML eXternal Entities (XXE) 공격에 대응하기 위해서는, 통신 구간 내 사용되는 XML 파서에 외부 엔티티 및 DTD 사용을 금지해야 합니다. 외부 엔티티 금지 설정 방법은 XML 파서에 따라 다르므로, 문서 속성 지정이 불가능할 경우 명시적으로 expand_entities(0)을 사용하여 확장 엔티티를 해제할 수 있습니다.

SOAP를 사용하는 경우에는 SOAP 1.2버전 이상으로 업그레이드 해야합니다.

기타 언어 및 라이브러리 별 XXE 대응방안은 해당 링크(XML External Entity Prevention Cheat Sheet)를 참고 바랍니다.

- SSI (Server Side Includes) Injection
웹 애플리케이션에 전달되는 사용자 입력 값이 HTML 태그로 적용되지 않도록 아래 특수문자 표를 참조해 사용자 입력값에 대한 유효성 검사를 해야하며 PHP의 경우 htmlspecialchars와 같은 함수를 사용하여 일반문자로 인식되도록 설정하시기 바랍니다.

From	To
<	<
>	>
'	'
"	"
#	#
!	!
=	=

[표] 10. SSI Injection 필터링 특수문자 목록

샘플소스

ASP.NET sample source code

```
// SqlCommand.Parameters 속성을 사용한 바인딩 처리
SqlCommand cmd = new SqlCommand();
SqlDataAdapter da = new SqlDataAdapter();
```

```
cmd.CommandText = string.Format("select * from Guideline.dbo.Country
Where Country=@Country");
cmd.Parameters.Add("@Country", SqlDbType.NVarChar);
cmd.Parameters["@Country"].Value = country;
cmd.CommandType = CommandType.Text;

da.SelectCommand = cmd;
da.Fill(ds, "Country");


return ds;


==============================================
// AntiSQLi 사용
SqlParameterizedQuery pQuery = new SqlParameterizedQuery();
pQuery.LoadQueryText("select * from Guideline.dbo.Country Where
Country = {0}", country);
pQuery.Connection = con;

da.SelectCommand = pQuery.Command;
da.Fill(ds, "Country");


return ds;


==============================================
//HttpModule을 통한 특수문자 필터링 선처리
* ASP.NET 2.0 이상
namespace SqlFilter{
  public class InjectionFiltering : IHttpModule{
  // 필터링 할 특수기호 및 문자열 추가
  public static string[] blackList = {"'","₩"", "=", "<", ">", "(", ")", "@@",
  "@", "/*", "*/", "select", "drop", "union", "–", "#", ";"};
  ...
```

```
void beginRequest(object send, EventArgs e){
  HttpRequest Request = (sender as HttpApplication).Context.Request;
  foreach (string key in Request.QueryString){
    Filtering(Request.QueryString[key]);
  }
  //필터링 수행
  private void Filtering(string param){
  for (int i = 0; i < blackList.Length; i++){
    //입력값을 소문자로 변환 뒤 blackList 문자열과 비교
    if ((param.ToLower().IndexOf(blackList[i],
      StringComparison.OrdinalIgnoreCase) > -1)){
      HttpContext.Current.Response.Redirect("~/Error.aspx");
    }
 ..
}
```

//HttpModule 설정

//IIS 6을 포함한 이전 버전 혹은 IIS 7 클래식 모드

```
<httpModules>
  ...
  <add name="InjectionFiltering" type="SqlFilter.InjectionFiltering"/>
  ...
</httpModules>
```

//IIS 7 통합 모드

```
<modules>
  ...
  <add name="InjectionFiltering" type="SqlFilter.InjectionFiltering"
    preCondition="managedHandler"/>
  ...
</modules>
```

2) XML eXternal Entities (XXE)

(.NET 4.0 이전)

```
  XmlTextReader reader = new XmlTextReader(stream);
  reader.ProhibitDtd = true;
(.NET 4.0-.NET 4.5.2)
  XmlTextReader reader = new XmlTextReader(stream);
reader.DtdProcessing = DtdProcessing.Prohibit;
```

ASP sample source code

```
' SQL, Command Injection 특수문자 필터링
<%
Function sqlFilter(search)
  'SQL Injection 특수문자 필수 필터링 문자 리스트
  strSearch = Array("'", "--", "#", ">", "<", "=", "*/", "/*", "+", "₩", "%", "(", ")", ";", "@", ",", "|", "&", "[", "]", "=")
  strdata = search
  cnt = 0
  For each str in strSearch  '필터링 인덱스를 배열 크기와 맞춰준다.
    strdata = replaceAll(strdata, str,"")
    cnt = cnt + 1
  Next
  sqlFilter = strdata
End Function
%>
```

PHP sample source code

```
// SQL, Command Injection 특수문자 필터링
<?
//SQL 입력값 문자열 필터
//$str = 입력 문자열
function sqlfilter($str) {
 if(preg_match("/[₩%₩'₩;₩"₩=₩-₩-₩#₩/*]+/", $str)) {
```

```
    return "error";
}
else {
$strdata = "₩'&₩&₩"&₩&(&)&#&>&<&=&*/&/*&+&₩&%&;&|&--
&@&=&[&]&,";
  $search = explode("&",$strdata);
  for($i=0;$i < count($search);$i++){
    $str = str_replace($search[$i], "", $str);
  }
  return $str;
}
?>
```

2) XML eXternal Entities (XXE)

libxml_disable_entity_loader(true);

JSP sample source code

```
// SQL, Command Injection 특수문자 필터링
public String sqlFilter(String str) {
        String Arr[] = {"'", "₩"", "--", ",", "₩₩(", "₩₩)", "#", ">", "<", "=",
"₩₩*/", "/₩₩*", "₩₩+", "%", ",", "@", ";", "₩₩₩₩", "|", "[", "]", "&"};
        for(int i=0; i< Arr.length; i++)
        {
                str = str.replaceAll(Arr[i],"");
        }
        return str;
}

//SQL 질의
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"));← set 함수를 사용하여
매개변수를 바인딩 해야 SQL Injection을 피할 수 있음.
pstmt.setString(2, request.getParameter("password"));
rs = pstmt.executeQuery();
```

iBatis/MyBatis sample source code

```
<statement id="insertProduct" parameter="java.lang.Integer">
select * from PRODUCT where PRD_ID =#value#← 반드시 #매개변수
사용해야 함
</statement>
```

MySQL

(iBatis의 경우)

SELECT * FROM tbl WHERE column LIKE **concat('%', #value#, '%')**

(MyBatis의 경우)

SELECT * FROM tbl WHERE column LIKE **concat('%', #{value}, '%')**

Oracle

(iBatis의 경우)

SELECT * FROM tbl WHERE column LIKE **'%' || #value# || '%'**

(MyBatis의 경우)

SELECT * FROM tbl WHERE column LIKE **'%' || #{value} || '%'**

Sysbase / MSSQL

(iBatis의 경우)

SELECT * FROM tbl WHERE column LIKE **'%' + #value# + '%'**

(MyBatis의 경우)

SELECT * FROM tbl WHERE column LIKE **'%' + #{value} + '%'**

예제

```
<statement id="getProduct" resultMap="get-product-result">
select * from PRODUCT where $preferredOrder$ ←$매개변수가 있으면 SQL Injection에 취약
</statement>
```

node.js sample source code

```
// SQL Command Injection 특수문자 필터링
  function sqlFilter(str) {
  array = [/"/g,/'/g,/#/g,/₩)/g,/₩(/g,/=/g,/</g,/>/g,/--/g,/;/g,/&/g,/₩|/g,/₩[/g,/₩]/g]
  for(var i=0; i<array.length;i++) {

  str=str.replace(array[i],"")
                    }
      return str
   }
   app.post('/:id/delete', function(req, res){
   var id = req.params.id;
   var sql = 'DELETE FROM test WHERE id=?';
   conn.query(sql, [id], function(err, result){
              if(err){ res.send('Error');}
              res.end(); });
});
```

JAVA sample source code

1) XML eXternal Entities (XXE)

```
.........
DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
String FEATURE = null;
FEATURE = "http://apache.org/xml/features/disallow-doctype-decl";
factory.setFeature(FEATURE, true);

DocumentBuilder builder = factory.newDocumentBuilder();
Document doc = builder.parse(new File("XML PATH"));
.........
```

2) SSI Injection

```
public class SSI_Injection {

public String getParameterValues(String parameter) {
        String values = parameter;
        String FilteredValues = "";
        if (values==null) {
            return null;
        }
        FilteredValues = Filter_SSI(values);
        return FilteredValues;
    }

    private String Filter_SSI(String value)
    {
        value = value.replaceAll("<", "&lt;").replaceAll(">", "&gt;");
        value = value.replaceAll("!", "&#33;").replaceAll("#", "&#35;");
        value = value.replaceAll("=", "").replace("₩"", "&#61;");
        value = value.replaceAll("--", "");
        value = value.replaceAll("exec", "");
        value = value.replaceAll("include", "");
        return value;
```

```
    }

    public static void main(String[] args) {
        String User_Input_Data = "공격 Payload";
        String Filter_Value = "";
        SSI_Injection ssi_injection = new SSI_Injection();
        Filter_Value = ssi_injection.getParameterValues(User_Input_Data);
        System.out.println(Filter_Value);
        }
}
```

2. 취약한 파일처리

취약한 파일처리 항목은 특정 서버 스크립트 파일을 업로드 하거나 악성코드 파일을 연동(include)하여 원격에서 임의의 명령을 웹 서버에서 실행시키거나, 정상적인 파일다운 경로를 조작하여 애플리케이션 소스파일을 내려 받거나 서버운영상 생성되는 로그파일 등을 내려 받을 수 있는 취약점입니다.
일반적으로 java로 작성된 웹 애플리케이션의 경우 시스템 관리자 권한으로 실행되고 있으므로 이 공격이 성공하면 공격자에게 시스템 관리자 권한을 제공하게 되어 심각한 위험에 빠질 수 있습니다.

2-1. 악성코드파일 업로드

구분	내용
주요내용	악성코드파일 업로드 가능성 취약점은 웹 서버의 잘못된 설정을 이용하여 웹 서버 내에서 악성코드를 실행하는 취약점입니다. 이 취약점은 웹 서버에서 실행 가능한 서버 스크립트 파일을 업로드 하고 실행하여 악성코드 파일을 연동(include)하여 원격에서 임의의 명령을 실행 시켜 웹 서버에 대한 시스템 권한을 획득하거나 중요정보파일이 유출될 가능성이 있습니다. IIS 6.0 버전에서는 특수기호(;)를 이용해 jpg 확장자를 가정한 악성 코드 파일을 업로드 하면 실행이 가능한 문제점이 있습니다. 예제 test1.asp;test.jpg, test2.php;test.jpg, test3.jsp;test.jpg IIS 7.5 버전 이하에서 에디터 및 게시판을 이용해 디렉터리 생성이 가능한 경우 디렉터리 생성 후 해당 폴더에 악성코드파일 업로드가 가능하고 실행 될 수 있습니다. IIS서버에서 ASP 구동시 사용되는 asp.dll 자체의 문제점으로 확장자 ".asp"에 대한 MIME 타입 구분이 없어 발생됩니다.(파일, 디렉터리) 예제 1.asp 이름으로 디렉터리 생성 후 1.jpg<확장자만 이미지로 바꾼 WebShell파일> 파일 업로드

	또한 웹 서버에 실행이 가능한 악성코드 파일이(aspx, asp, php, jsp, html, cgi, perl, sh, exe등) 업로드 되어 웹 애플리케이션을 통해 파일이 실행되지는 않지만 다른 취약점의 연계공격에 이용될 수 있습니다.
대응방안	Application Server에서 실행이 가능한 악성 코드파일의(aspx, asp, php, jsp, html, cgi, perl, sh, exe등) 업로드를 차단하기 위해 수용 가능한 파일의 확장자외 모든 파일의 업로드를 금지해야 하며 Server Side Language(PHP, ASP, JSP 등)에서 제한하도록 설정해야 합니다. 예제 이미지 파일: (JPG, GIF, BMP등 수용 가능한 확장자만 허용) 문서 파일: (xlsx, pdf, pptx, docx등 수용 가능한 확장자만 허용) 수용 가능한 파일의 확장자 검증 시 대소문자 구분 없이 비교하는 함수를 사용해야 하며, 이중 확장자(ex: file.jpg.jsp 등)나 확장자 끝에 "."를 사용하는 등의 trick에 의해서 확장자 검사가 우회되지 않도록 주의 깊게 처리해야 합니다. 또한 업로드 되는 파일 명에는 "../", ";", "%0" 등과 같은 특수문자를 사용할 수 없도록 Server Side Language(PHP, ASP, JSP 등)에서 제한해야 하며 업로드 파일 크기를 제한(파일 사이즈가 너무 크거나 작은 파일에 대한 제한조치)해야 합니다. [표] 9 참조 [표] 11. 업로드 파일 필터링 문자 목록 웹 서버를 통한 업로드 경로는 웹 애플리케이션 하위 폴더가 아닌 다른 폴더 나 별도의 드라이브 또는 별도의 물리적 서버에 분리하여 저장하고 외부로부터 업로드 디렉터리에 URL 접근이 불가능 하도록 해야 합니다. 부득이하게 웹 애플리케이션 하위 폴더에 저장할 경우는 웹 애플리케이

필터링 문자
../
./
..₩
.₩
%
;

션 폴더와 구분이 가능하도록 폴더 구조를 생성하고 업로드 폴더 위치를 서버 관리자에게 전달하여 악성 코드 파일의 실행을 원천적으로 막을 수 있도록 해야 합니다.

Web 서버에서 처리해야 할 소스를 WAS 서버에서 처리하도록 구성하는 것은 보안 위반이며 이렇게 구성되어 있다면 빠른 시간 내에 서버관리자와 협의하여 구성을 변경하여야 합니다.

Include 되는 파일명에 대한 입력 값 검증이 필요하며 입력되는 값은 상대경로 및 외부경로에 대한 제한을 해야 합니다.
업로드 된 파일이 서버상에서 실행이 불가능하도록 디렉터리의 실행권한을 삭제하고, 에디터 사용시 디렉터리 생성기능을 제거해야 합니다.

Apache를 사용하는 경우 AddLanguage, AddCharset 옵션의 활성화를 금지하거나 두 옵션에 등록된 확장자를 필터링 해야 합니다.

샘플소스

ASP.NET sample source code

```
 public bool checkUploadFile(string strFile)
{
  // 허용가능한 파일일 경우 true 리턴
  bool bResult = false;

  //업로드 가능한 확장자
  string[] allowExt = new string[] { "jpg","jpeg","png" };

  // 특수문자 필터링
  if(strFile.IndexOf("..") != -1 || strFile.IndexOf("./") != -1 ||
strFile.IndexOf(".₩₩") != -1 || strFile.IndexOf("%") != -1 ||
strFile.IndexOf(";") != -1 || strFile.IndexOf("₩0") != -1)
  {
    bResult = false; // 특수문자 포함된 경우 false 리턴
  } else {
    // 확장자 검증
    int tempPoint = strFile.LastIndexOf(".");
    if(tempPoint > -1)
```

```
        {
            string fileExt = strFile.Substring(tempPoint + 1);

            if (fileExt != null && !fileExt.Trim().Equals(""))
            {
                // 소문자로 변환하여 확장자 검증
                if(Array.IndexOf(allowExt, fileExt.ToLower()) != -1)
                    bResult = true; // 확장자 검증 성공 시 true로 변경
            }
        }
    }
    return bResult;
}
```

ASP sample source code

```
' 업로드 필터링 함수
<%
Function Check_Ext(filename)
    Dim Exts, FileStartName, FileEndName
    Exts="jpg,gif,bmp" '허용할 확장자
    If instr(filename, "₩0") or instr(filename, ";") or instr(filename, "..") or
instr(filename, ";") or instr(filename, "./") or instr(filename, ".₩") Then
        response.write "출력데이타 : "&"error"&"<br>"
        Response.End
    End If

    FileEndName = Mid(filename, InstrRev(filename, ".")+1)

     '허용할 확장자 체크
    if Exts <> "" Then
        ok_file = split(Exts,",")
        for each p in ok_file
            if instr(FileEndName,p)>0 then
                Check_st = 1
            End If
        next
```

```
  End If

  if Check_st >0 Then
    Check_Ext = "ok"
  Else
    Check_Ext = "error"
  End If
End Function
%>
```

PHP sample source code

```
//업로드 필터링 함수
<?
function checkext($filename) {
  $exts = "jpg,gif,bmp"; //허용할 확장자 리스트
  $file_ext = substr(strrchr($filename, '.'), 1);
  if(eregi("₩₩0",$filename) || eregi("₩;",$filename)|| eregi("₩.₩.",$filename)
|| eregi("₩.₩/",$filename)) { return "error"; }

  if ($exts != "") {
    $check = 0;
    $ext_check=explode(",",$exts);
    for($i=0;$i < count($ext_check);$i++) {
      if($file_ext == $ext_check[$i]) { $check = 1;}
    }
    if($check) { return "ok"; }
    else { return "error"; }
  }
}
?>
```

JSP sample source code

```
// 업로드 필터링 함수
<%
public String checkext(String fileName) {
```

```
        String chkExt = "false";
        String Exts = "jpg,gif,bmp"; //허용할 확장자

        if ((fileName.indexOf("₩0") > -1) || (fileName.indexOf(";") > -1) ||
(fileName.indexOf("./") > -1) || (fileName.indexOf(".₩₩") > -1))
        {
                chkExt = "false";
        }else{
                String file_ext =
fileName.substring(fileName.lastIndexOf('.') + 1);
                if (!Exts.equals("")) {
                        Exts = Exts.replaceAll(" ","");
                        String compStr[] = Exts.split(",");
                        for (int i = 0;i < compStr.length; i++) {
                                if
(file_ext.equalsIgnoreCase(compStr[i])) {
                                        chkExt = "true";
                                }
                        }
                }else{
                        chkExt = "true";
                }
        }
        return chkExt;
}
%>
```

2-2. 중요 정보 파일 다운로드 가능성

구분	내용
주요내용	대부분의 웹 사이트들은 파일을 직접 링크시켜 다운로드 하기보다는 일반적으로 독립적인 파일 다운로드 애플리케이션을 작성, 이를 이용하여 파일 다운로드를 시도합니다. 파일 다운로드 애플리케이션의 작동 방식은 웹 사이트 별로 다양하지만 특정 디렉터리 내에 있는 디렉터리나 파일에 접근하기 위해 외부 입력 값으로 접근 경로명을 구성할 경우 지정된 디렉터리 외 경로로 접근할 수 있는 특수문자("..", "/" 등)를 경로명에 포함하여 시스템 내의 파일을 다운로드 할 수 있습니다. 또한 다운로드 할 파일의 내용을 읽어 들이는 함수들은 일반적으로 파일의 마지막을 null byte로 규정하고 null byte가 나올 때까지 파일을 읽어 들이는 기능을 가지는데 입력 값으로 null byte를 포함하는 값을 전송했을 경우 정상적으로 파일 읽기를 종료하고 사용자의 브라우저에 해당 파일의 내용을 표시하게 됩니다. 예제 http://victim.com/download.jsp?**path=/upload/**&**filename=victim.zip** **※ 파일 다운로드 기능이 구현된 웹 애플리케이션에서 발생하는 취약점**
대응방안	별도의 다운로드 애플리케이션을 사용해야 할 경우 웹 애플리케이션(DocBase) 하위 폴더가 아닌 상위 경로 폴더 또는 타 드라이브에서 제공되어야 하며 **파일이 위치하는 경로를 변수로써 사용하지 말고** 해당 파일의 키 값을 변수로 사용하여 다운로드를 제공해야 합니다. 파일이 위치하는 경로가 다양한 경우에는 경로 맵핑 변수를 활용하여 서버에서 위치를 맵핑하여 처리해야 하며 파일 업로드가 많고 소스 위치가 다양할 경우에는 필터 기능을 활용하여 처리하는 것을 권고합니다. **※ 필터 기능관련 지원이 필요한 경우 IT보안 담당자를 통해 서버관리자에게 요청하여 지원 받을 수 있습니다.** 파일 다운로드 모듈의 경로를 처리하는 파라미터 변수에 대해 Server Side Language(PHP, ASP, JSP 등)에서 아래 표의 특수문자(path traversal 문자열)를 포함하는 파라미터 변수일 경우 에러를 발생시켜 프로세스를

중단하도록 처리해야 합니다.

path traversal 문자
../
./
..₩
.₩
%
;

[표] 12. 다운로드 파일 필터링 문자 목록

또한, 요청 사용자의 권한, 유효세션에 대한 검증 및 다운로드 하려는 파일의 정보를 최소한으로 제공하거나 인지 불가하도록 구현해야 합니다.

예제

http://www.test.com/Download.jspx?**filePath=UlBcMjAxMFwwM**&**fileName=MjAwOeuFhCTshKD**

http://www.test.com/web/filedownload.cmd?**fileId=231996be-6a32-48cb-abbc-6f3d35fcc901**

샘플소스

ASP.NET sample source code

```
public bool checkDownloadFile( string strFile )
{
  // 허용가능한 파일일 경우 True 리턴
  bool bResult = true;

  if ( strFile.IndexOf("..") != -1     || strFile.IndexOf("/") != -1
    || strFile.IndexOf("₩₩") != -1 || strFile.IndexOf("%") != -1
    || strFile.IndexOf(";") != -1 )
  {
    bResult = false;
  }

  return bResult;
}
```

ASP sample source code

```
' 다운로드 필터링 함수
<%
  Function Check_Down(fileName)
     If instr(filename, "..") or instr(filename, "./") or instr(filename, ".₩")
     or instr(filename, "..₩") or instr(filename, "../") or instr(filename, "%")
     or instr(filename, ";")
       Then
        Check_Down = "error"
     Else
        Check_Down = "ok"
     End If
  End Function
%>
```

PHP sample source code

```
// 다운로드 필터링 함수 (PHP 5.3 이전버전)
<?
  function check_Down($filename)
     {
     if(eregi("₩.₩.",$filename) ||eregi("₩.₩/",$filename)) ||
       eregi("₩..₩",$filename)) || eregi("₩..₩/",$filename))
       ||eregi("%",$filename))   {
        return "error";
     }
     else {
        return "ok";
     }
  }
?>

// 다운로드 필터링 함수 (PHP 5.3 이후버전)
<?
  function check_Down($filename)
```

```
    {
    if(preg_match("/₩.₩//",$filename) ||
      preg_match("/₩.₩₩₩/",$filename) || preg_match("/%/",$filename))
      {
        return "error";
    }
    else {
        return "ok";
    }
  }
?>
```

JSP sample source code

```
// 다운로드 필터링 함수
<%
public String checkpath(String fileName) {
  if((fileName.indexOf("..")) != -1 || (fileName.indexOf("./")) != -1 ||
  (fileName.indexOf(".₩")) != -1|| (fileName.indexOf("../")) != -1 ||
  (fileName.indexOf("..₩")) != -1 || (fileName.indexOf("%")) != -1) {
          return "false";
          }
          return "true";
}
%>
```

3. 취약한 인증 및 세션 관리

인터넷 쇼핑몰의 발전과 더불어 고객의 상태 정보를 유지하기 위한 수단의 하나로 Cookie가 등장하였으며 이후 보안성이 우수해진 Session이 등장하여 정당한 사용자임을 증명하는 보편적인 방법이 되었습니다. 그러나 모든 웹 페이지에서 정당한 사용자임을 판단하는 로직을 구현하지 않을 경우 인증을 우회하여 불법적으로 웹 페이지에 접속하고 경우에 따라 중요한 정보가 유출되는 사고가 발생할 수 있습니다.

4-1. 쿠키(Cookie) 및 웹 스토리지(Web Storage) 조작 가능성

구분	내용
주요내용	쿠키(Cookie)는 클라이언트와 서버간의 정보 유지를 위해서 사용하는 기술로 HTTP 프로토콜의 단점을 보완하고자 만든 기술입니다. 하지만, 별도의 암호화나 안전성을 확보하지 못해, 네트웍 패킷을 캡쳐하거나 간단한 툴을 사용하는 것만으로도 해당 값을 알아낼 수 있습니다. 이와 비슷한 형태의 웹 스토리지(Web Storage)는 HTML 5에서 제공되는 기능(API) 중 하나로 사용자 로컬 (Client-Side) 내 데이터를 저장/관리하는 기술이며, 저장되는 데이터는 두 개의 저장소 (Local Storage 또는 Session Storage) 내에 Key-Value 형태로 값을 저장/사용합니다. 쿠키와 비슷한 목적으로 사용되지만 다른 특징들이 존재합니다. 이처럼 사용자 측에 생성되는 쿠키 및 웹스토리지에 인증정보, 권한 및 중요 데이터를 저장하여 해당 값만으로 인증을 하는 경우 쿠키 또는 웹 스토리지의 값을 조작하여 일반 사용자가 다른 사용자 또는 관리자로 도용할 수 있으며 권한상승 및 인증을 우회할 수 있습니다. **- 웹 스토리지 (Web Storage)** 1) HTTP 요청 시 쿠키(Cookie)와는 달리 서버에게 데이터를 전달하지 않으며, 쿠키(Cookie)보다 더 많은 데이터를 웹 스토리지 객체에 저장할 수 있습니다. 2) 웹 스토리지 객체는 기본적으로 SOP(Same Origin Policy)에 의해 보호받기 때문에 프로토콜과 서브 도메인이 다르면 데이터에 접근할 수 없습니다. 3) 웹 스토리지 객체 조작은 자바스크립트(JavaScript)를 사용해야 합니

	다. 4) 로컬 스토리지 (Local Storage)의 데이터는 사용자 혹은 웹 애플리케이션에 의해서 삭제될 수 있지만, 일반적으로 브라우저를 종료해도 데이터들은 삭제되지 않습니다. 5) 세션 스토리지 (Session Storage)는 기본적으로 Local Storage와 유사하지만 브라우저 혹은 탭을 종료할 경우에 데이터가 삭제 됩니다.
대응방안	Client Side Cookie 및 Web Storage는 그 구조상 다양한 취약점에 노출될 수 있으므로 가능한 웹 서버에서 제공되는 Server Side Session을 사용하는 것이 바람직합니다. 사용자의 인증 또는 권한 체크 및 인증이나 기타 반드시 저장해야 할 정보를 위해 쿠키 또는 웹 스토리지 내 값을 사용해야 한다면 사용자가 임의적으로 정상적인 형태의 값을 추측 및 변조할 수 없도록 안전한 알고리즘을 사용하여 암호화 하거나 가능하면 패킷 전체에 암호화를 지원하여 데이터 안전성을 확보해야 합니다. 또한 쿠키 또는 웹 스토리지 내 저장된 값만을 가지고 사용자를 식별하는 경우 타인의 쿠키 값이 도용된 상태라면 해당 사용자의 로그인 상태를 유지할 수 있게 되며, 특히 사용자정보 수정 또는 유료 결제와 같은 중요 프로세스에 대해 불법적으로 수행이 가능할 수 있습니다. 따라서 개인의 고유 정보에 침해될 위협이 있는 중요 프로세스에는 기본적인 로그인 인증 이 외에 추가적으로 본인 확인을 거치는 등(SMS 인증, 패스워드 재인증 등)의 2차 인증 정책을 마련하는 것을 권고합니다. 이 외에 쿠키 생성 시 secure 옵션을 설정하여 HTTPS 통신을 사용할 경우에만 쿠키가 전송될 수 있도록 하며, 이용자의 중요 개인 정보나 이용되지 않는 불필요한 쿠키정보는 저장되지 않도록 해야 합니다. 특히 이용자의 쿠키 유출을 막기 위해 XSS 취약점에 대한 보안이 적용되어야 합니다. 예제 - PHP php.ini 파일을 아래와 같이 설정 **session.cookie_secure=true**

	- JAVA web.xml 파일을 아래와 같이 설정 **<session-config>** **<cookie-config>** **<secure>true</secure>** **</cookie-config>** **</session-config>**

샘플소스	**ASP.NET sample source code**
	//Web API Controller 사용 시 AuthorizeAttribute 필터를 사용한 웹 사용자 권한 검증
	//App_Start₩FilterConfig.cs 파일 생성 혹은 수정
	public static void RegisterGlobalFilters(GlobalFilterCollection filters){
	…
	filters.Add(new System.Web.Http.AuthorizeAttribute());
	…
	}
	//Controller 파일
	//권한이 없는 경우
	[AllowAnonymous]
	public ActionResult Index(){
	return View();
	}
	//특정 User에게만 허용 시
	[Authorize(Users="Hong,Kim")]
	public ActionResult List(){
	return View(db.Contacts.ToList());
	}

```
//특정 권한에게만 허용 시
[Authorize(Roles="Admin")]
[HttpPost, ActionName("Delete")]
  public ActionResult Delete(int id){
    User user = db.Users.Find(id);
    db.Users.Remove(user);
    db.SaveChanges();
    return RedirectToAction("UserList");
}
```

ASP sample source code

```
<%
UserGrade = "admin" '해당 웹페이지의 통제 레벨 설정 admin 의 경우
IP Address, 사용자인증 모두 체크

'세션 체크 정상적으로 로그인 했는지 조사
If UserGrade <> Session("user_grade") Then
 Response.Write "<script>alert('관리자 레벨: 접근 제한');history.go(-
 1);</script>"
        Response.End
Else
      '사용자 권한이 관리자일 경우
      If UserGrade = "admin" Then
            '접근한 사용자 IP Address 와 세션에 설정된 IP
Address 에 포함되는지 조사
        If (instr(Request.Servervariables("REMOTE_ADDR"),
 Session("access_ip"))) Then
                '로그인 하였는지 여부 조사
                If Session("logged_in") = "" Then
                      Response.Write
"<script>alert('로그인: 접근 제한');history.go(-1);</script>"
                End If
                Response.End
```

```
                        Else
                                Response.Write "<script>alert('IP: 접근 제한');history.go(-1);</script>"
                                Response.End
                        End If

            '권한이 사용자일 경우
            ElseIf UserGrade = "user" Then
                        If Session("user_id") = "" Then
                                Response.Write "<script>alert('USER 레벨: 접근 제한');history.go(-1);</script>"
                                Response.End
                        End If
            End If
End If
%>
```

PHP sample source code

```
<?
//이 페이지를 각 폴더 별로 넣고 접근을 통제할 웹 페이지 초기 부분에 모두 include 해야 한다.
$UserGrade = "admin";

@session_start();

//세션 체크 정상적으로 로그인 했는지 조사
if ($UserGrade != $_SESSION["user_grade"]) {
            print "관리자 레벨: 접근 제한";
            exit;
}
else {
            //사용자 권한이 관리자일 경우
            if ($UserGrade == "admin") {
                        //접근한 사용자 IP Address 와 세션에 설정된 IP Address 에 포함되는지 조사
```

```
                        if (eregi($_SESSION["access_ip"],
$_SERVER["REMOTE_ADDR"] ) ==   false) {
                                print "IP: 접근 제한";
                                exit;
                        }
                        else {
                                //로그인 하였는지 여부 조사
                                if (isset($_SESSION["logged_id"])) {
                                        print "로그인: 접근 제한";
                                }
                                exit;
                        }
                }

                //권한이 사용자일 경우
                else if ($UserGrade == "user") {
                        if ($_SESSION["user_id"] == "") {
                                print "USER 레벨: 접근 제한";
                        }
                        exit;
                }
}
?>
```

JSP sample source code

```
<%@ page language="java" import="java.sql.*,java.util.*"
contentType="text/html;charset=KSC5601" %>
<%@ page
import="java.util.*,java.lang.*,java.io.*,java.sql.*,java.text.*" %>
<%
HttpSession sess = request.getSession(true);
String access_ip = request.getRemoteAddr();
String admin_ip = (String)sess.getValue("admin_ip");
String UserGrade = "admin";

if (UserGrade.equals(sess.getValue("user_grade"))) {
```

```
            if (UserGrade.equals("admin")) {
                        if (access_ip.indexOf(admin_ip) != -1) {
                                    if (sess.getValue("logged_in").equals("")) {
                                                out.println("권한 없음");
                                                return;
                                    }
                        }
                        else {
                                    out.println("IP: 권한 없음");
                                    return;
                        }
            }

            else if (UserGrade.equals("user")) {
                        if (sess.getValue("user_id").equals("")) {
                                    out.println("권한 없음");
                                    return;
                        }
            }
}

else {
            out.println("권한 없음");
            return;
}
%>
```

4-2. 인증(세션 및 토큰) 값 안전성 설정 여부

구분	내용
주요내용	해당 항목은 웹 애플리케이션 및 응용프로그램 이용 시 사용되는 사용자, 단말 등의 인증(세션 및 토큰) 값에 대한 안전성 설정 및 적용 여부를 확인하는 취약점입니다. **- 복잡도** 사용자 혹은 단말 인증을 위한 토큰 사용 시 해당 인증 값에 대한 복잡도(암호화, 생성패턴, 길이) 설정이 미흡할 경우 공격자로 하여금 해당 인증 값을 통해 타 사용자의 정보 유추등의 추가 공격의 활용 가능성이 존재합니다. **- 만료시간(타임아웃) 설정** 로그인 또는 사용이 완료된 인증 값의 경우 일정시간(권고: 30분)이 지난 이후에도 사이트 접근이 가능한 경우에 문제가 될 수 있습니다. **- 미파기** 서비스 내 사용자가 로그아웃 등과 같은 형태로 사용자 인증을 해지하는 경우 서버 내에 적절한 파기가 이루어지지 않으면 공격자로부터의 악의적 사용에 노출될수 있습니다. **- 사용횟수 제한** 단말 인증 또는 결제 시 사용되는 인증 토큰과 같이 지속적인 사용이 아닌 일회성 또는 단발적 사용의 인증 값의 경우 사용횟수가 지정되지 않을 경우 공격자로부터 하여금 지속적인 인증 시도의 문제가 발생할수 있습니다. **- 동일 세션 ID 발급 및 재사용** 로그인 시 마다 동일한 세션 ID가 생성되어 사용되는 경우 세션 탈취를 통해 공격자로부터의 악의적 사용에 노출될수 있습니다. **- 중복 로그인** 하나의 사용자 또는 관리자 계정이 서로 다른 IP주소의 클라이언트에서 동일한 세션 정보를 이용해 사이트 동시 접속이 가능한 경우 해당 계정을 통해 계정 도용 및 중요정보 탈취 등의 악의적인 공격이 가능할 수

	있습니다.
대응방안	사용자의 안전한 인증(세션 및 토큰) 값을 사용하기 위해서는 아래와 같이 적용하여 사용해야합니다. **- 복잡도** 사용자 인증 토큰의 경우 사용 시 공격자로 하여금 사용자의 인증 정보를 추측하지 못하도록 안전한 암호 알고리즘을 통해 암호화해 사용해야하며 사용 이후 재사용 시에는 다른 형태의 패턴으로 사용해야합니다. **- 만료시간(타임아웃) 설정** 서비스 내 만료시간(타임아웃) 설정을 통해 **일정시간(권고: 30분)** 동안 움직임이 없을 경우 자동 로그아웃 등과 같이 해당 인증값이 재사용 되지 않도록 파기되도록 설정해야 합니다. **- 미파기** 세션 및 토큰의 경우 로그아웃 및 브라우저 종료 등의 사용자 인증 값 사용을 했을 경우 세션과 토큰을 파기하여 재사용을 금지해야 합니다. **- 사용횟수 제한** 단말 인증 또는 결제 시 사용되는 인증 토큰과 같이 지속적인 사용이 아닌 일회성 또는 단발적 사용의 인증 값의 경우 사용횟수를 제한해 재사용이 되지 않도록 해야합니다. **- 동일 세션 ID 발급 및 재사용** 세션 ID는 로그인 시 마다 새로운 **세션 ID를 발급**받도록 해야 합니다. **- 중복 로그인** 하나의 계정으로 동시 로그인이 불가능 하도록 **"고유한 값(Key)"을** 세션 값에 포함하여 서버 측에서 검증하도록 해야 합니다. 예제 Key = Hash(TimeStamp | IP Address | User ID | Token) 등 **※ 외부서비스 인증의 경우 세션 값에 적용 시 NAT IP로 인한 사용자**

	접속오류가 발생할 수 있으니, 서비스 및 운영상의 영향도를 확인하시어 고유한 값을 설정하여 적용해야 합니다.

샘플소스	**ASP.NET sample source code**
	// 로그인 체크 함수
	public ActionResult requestLogin(string id, string pw)
	{
	  ... 중략 ...
	
	  if (checkLoginUser(id, pw))
	  {
	  // 로그인 성공 시 ID, IP 를 세션에 저장
	  Session["UserID"] = id;
	  Session["LoginIP"] = Request.UserHostAddress;
	  }
	
	  ... 중략 ...
	
	  return View();
	}
	
	// 로그인 권한필요 페이지 출력 시
	public ActionResult openUserInfoPage( )
	{
	  // 로그인 사용자 여부 검증
	  if (Session["UserID"] == null
	  || String.IsNullOrEmpty(Session["UserID"].ToString()))
	  {
	  return Redirect("/Error?message=Need Login");
	  }
	
	  // 세션이 동일한 사용자에게 사용되었는지 확인
	  string requestIP = Request.UserHostAddress;
	  String loginIP = Session["LoginIP"].ToString();
	
	  if (!requestIP.Equals(loginIP))

```
  {
     return Redirect("/Error?message=Your session is duplicated");
  }

  ... 중략 ...

  return View();
}
```

ASP sample source code

```
'IP 정보 세션 저장 후 비교
<%
Session("user_ip") = Request.Servervariables("REMOTE_ADDR")

If Session("user_ip") = Request.Servervariables("REMOTE_ADDR") Then
  Response.write "허용 세션"
Else
  Response.write "중복 세션"
End If
%>
```

PHP sample source code

```
' IP 정보 세션 저장 후 비교
<?
  @session_start();
  $user_ip = $_SERVER["REMOTE_ADDR"];
  session_register("user_ip"); // 사용자 IP 를 저장

  if (eregi($_SESSION["user_ip"], $_SERVER["REMOTE_ADDR"] ) ==
false) {
            print "중복 세션";
            exit;
        }
  else {
             print "허용 세션";
             exit;
    }
```

```
?>
```

JSP sample source code

' IP + ID 정보 세션 저장

```
<%
  String ip = request.getRemoteAddr();    // IP 정보
  session.setAttribute("ip", ip);
  session.setAttribute("uid", userid);
 %>
```

' 저장 된 세션 정보 비교

```
<%
  String gs = session.getAttribute("uid");
  String bs = (String) userid;

  if(session.getAttribute("bs") !=null)
  {
    if(bs.equals(gs))
    {
      out.println("<br><h1>허용 세션</h1>");
    }
    else
    {
      out.println("중복 세션");
    }
  }
  else
   {
      out.println("<br>비 로그인 상태");
    }
%>
```

Tomcat / Unix, Windows

(조치 방법)

/[Tomcat 설치 디렉터리]/conf/web.xml 설정

	<session-config> <session-timeout>30</session-timeout> </session-config>
	WebLogic / Unix, Windows (조치 방법) **web.xml (분단위 설정)** <session-config> <session-timeout>30</session-timeout> </session-config> **weblogic.xml (초단위 설정)** <session-descriptor> <timeout-secs>1800</timeout-secs> <session-descriptor> 둘 다 설정했을 경우에는 web.xml의 설정이 우선 적용.

JEUS/ Unix, Windows

(조치 방법)

JEUS 4, 5 : WEBMain.xml의 shared설정 값에 따라 적용순서가 달라짐.

```
<session-config>
  <timeout>30</timeout>
  <shared>false</shared>
</session-config>
```

<shared>true</shared>일 경우

1. setMaxInactiveInterval
2. WEBMain.xml
3. web.xml
4. webcommon.xml

<shared>false</shared>일 경우

1. setMaxInactiveInterval
2. web.xml
3. webcommon.xml
4. WEBMain.xml

JEUS6 : JEUS6에서는 <shared>값과는 무관함.

1. setMaxInactiveInterval
2. web.xml
3. webcommon.xml
4. WEBMain.xml

각 설정법

CONTEXT/WEB-INF/web.xml

```
<session-config>
```

```
    <session-timeout>30</session-timeout>
</session-config>
```

SERVLET_ENGINE/webcommon.xml

```
<session-config>
    <session-timeout>30</session-timeout>
    </session-config>
```

SERVLET_ENGINE/WEBMain.xml

```
<web-container>
  <context-group>
      ... 중략 ...
    <session-config>
      <timeout>30</timeout>   <!-- 분단위 -->
      <shared>false</shared>
    </session-config>
      ... 중략 ...
  </context-group>
</web-container>
```

4-3. 접근제어 우회 가능성 확인

구분	내용
주요내용	접근 제어 및 권한 체크, 필요 프로그램 설치, 파일 업로드, 실명인증 등 인증, 권한, 제어, 확인 등의 과정을 처리하는 부분에서 Client Side Script(Javascript, VBScript 등)를 사용하여 제어하는 경우 사용자가 임의로 수정하여 쉽게 우회하여 접근할 수 있습니다. 또한 중요정보를 보여주는 화면에 no-cache를 설정하지 않은 경우, 로그아웃 후에도 브라우저의 뒤로 가기 기능을 통해 해당 중요정보를 열람할 수 있습니다.

대응방안	**- 인증/권한 처리 Server Side 구현**
	인증, 권한, 제어, 확인 등의 과정을 처리하는 부분에 Client Side Script(Javascript, VBScript 등)로 제어하지 않는 것이 가장 바람직합니다. 만약 확인해야 할 정보가 있다면 Client Side Script가 아닌 Server Side Language(PHP, ASP, JSP 등)에서 체크하여 리턴 값을 client로 넘어가게 해야 합니다.
	실명 인증 시 전송한 데이터와 실명 인증 후에 데이터 값을 비교하여 조작 유무를 확인하도록 설정해야 합니다.
	
	예제: 취약한 프로그래밍
	정상적인 페이지에 javascript를 삽입하는 방식으로 redirection을 처리할 경우, 사용자가 javascript를 중간에 삭제하여 정상적인 페이지의 내용을 열람할 수 있음.
	%
	(중략)
	// 인증 처리 부분 (인증이 실패했을 경우 login.asp로 redirection)
	If Session(logged_in) != 1 Then
	Response.write "<script>location.href='/login.asp';</script>"
	// javascript 를 이용해서 페이지 redirection
	End If
	(... 정상적인 페이지 내용 ...)
	%>
	
	- cache 사용 제한
	또한 회원정보 열람/ 수정, 금융결제, 상품 및 콘텐츠 구매, 핸드폰 인증

	등의 페이지에서 cache 사용을 제한해야 하며, no-cache 설정을 위해서 HTML HEAD 부분 적용 및 해당 언어별 소스를 참고하시기 바랍니다.
	
	※ 서비스 및 운영상의 영향도를 확인하시어 no-cache 값을 설정하여 적용해야 합니다.
	
	HTTP/1.0
	`<META http-equiv="Expires" content="-1">`
	`<META http-equiv="Pragma" content="no-cache">`
	`<META http-equiv="Cache-Control" content="No-Cache">`
	
	HTTP/1.1
	`<META http-equiv="Cache-Control"content="No-Cache"/>`
	
	헤더 설정 : Chrome, Edge, Firefox, IE, Opera, Safari 호환 가능
	Cache-Control: no-cache, no-store, must-revalidate
	Pragma: no-cache
	Expires: -1

샘플소스	**ASP.NET sample source code**
	// 세션을 통한 접근 권한 검증
	public ActionResult openUserInfo()
	{
	... 중략 ...
	
	// 로그인 사용자 여부 확인
	if (Session["UserID"] == null
	|| String.isNullOrEmpty(Session["UserID"].ToString()))
	{
	return Redirect("/Error?message=Need Login");
	}
	
	// 세션 내 UserID 값을 사용하여 DB 에 저장된 사용자 그룹 리턴
	string userID = Session["UserID"].ToString();
	string group = GetUserGroup(userID);

```
  if (group == null || String.isNullOrEmpty( group ))
  {
     return Redirect("/Error?message=Need Login");
  }

  // 현재 출력하려는 페이지에 대한 권한체크
  if ( !CheckPermission( group, pageCode ) )
  {
      return Redirect("/Error?message=Need Permission");
  }

  ... 중략 ...

  return View();
}
```

// no-cache 설정

```
<%
Response.Cache.AppendCacheExtension("no-store");
Response.AppendHeader("Pragma", "no-cache");
Response.AppendHeader("Expires",-1);
%>
```

ASP sample source code

// 세션을 통한 접근 권한 검증

```
<%
        '유효세션 및 접근권한 점검 예제
        '인증성공 시 세션 값 셋팅
        ... 중략 ...
        Session("sessionChk") = True
        Session("UserID") = userID
        Session("UserGrp") = userGrp
        Session("UserIP") = Request.Servervariables("REMOTE_ADDR")
        ... 중략 ...
```

```
        ' 사용자 그룹 리턴함수
        Function GetUserGroup(strUserID)
        End function
        ... 중략 ...

        ChkUserGrp = GetUserGroup(userID)  '세션userID값을 통해
        DB에 저장된 사용자 그룹 리턴
        ... 중략 ...
%>
```

// no-cache 설정

```
<%
Response.AddHeader "cache-control", "no-cache, no-store"
Response.AddHeader "pragma", "no-cache"
Response.Expires -1
%>
```

PHP sample source code

// 세션을 통한 접근 권한 검증

```
<%
// 유효세션 및 접근권한 점검 예제
... 중략 ...

        PortalSessionManager sessionMgr = (PortalSessionManager)
session.getAttribute("sessionMgr");

        if (sessionMgr == null || sessionMgr.getUserId() == null) {
        (new FailToAuthenticateCmd()).execute(request,response);
        }
        ... 중략 ...

    String usrGrp = session.getAttribute("Usrgrp") == null ?
    "" : (String)session.getAttribute("Usrgrp");

    if (!usrGrp.equals("") || !userGrp.equals(Code.getMarket())) {
```

```
            // 접근권한을 인가할 수 없음
                    (new FailToPermissionCmd()).execute(request,response);
        }
        ... 중략 ...
%>
```

// no-cache 설정

```
<%php
header('Cache-Control: no-cache, no-store');
header('Pragma: no-cache');
header('Expires: -1');
%>
```

※ PHP의 경우 php.ini 설정 파일 내 "session.cache_limiter = nocache" 속성을 통해 no-cache 설정이 가능

JSP sample source code

// 세션을 통한 접근 권한 검증

```
<%
// 유효세션 및 접근권한 점검 예제
    ... 중략 ...
        PortalSessionManager sessionMgr = (PortalSessionManager)
    session.getAttribute("sessionMgr");
        if (sessionMgr == null || sessionMgr.getUserId() == null) {
        (new FailToAuthenticateCmd()).execute(request,response);
        }
    ... 중략 ...

    String usrGrp = session.getAttribute("Usrgrp") == null ?
    "" : (String)session.getAttribute("Usrgrp");

    if (!usrGrp.equals("") || !userGrp.equals(Code.getMarket())) {
            // 접근권한을 인가할 수 없음
        (new FailToPermissionCmd()).execute(request,response);
    }
```

	... 중략 ...
	%>
	
	// no-cache 설정
	<%
	response.setHeader("Cache-Control", "no-cache, no-store");
	response.setHeader("Pragma", "no-cache");
	response.setDateHeader("Expires", -1);
	%>

4-4. 비인증 상태로 중요 page접근 가능성

구분	내용
주요내용	로그인 후에 사용하는 게시판이나 관리자 페이지는 반드시 로그인 확인을 거치도록 해야 합니다. 만약 로그인을 확인하는 모듈이 존재하는 않는 관리자 페이지나 기타 페이지가 존재한다면 로그인을 하지 않고 직접적인 접근이 가능하게 되는 문제가 발생 할 수 있습니다.
대응방안	인증이 필요한 페이지의 경우 해당 페이지 주소를 직접 입력하여 접근하지 못하도록 페이지 각각에 대하여 로그인 체크 및 권한 체크를 점검하는 모듈을 적용해야 합니다. 하나의 프로세스가 여러 개의 페이지 또는 모듈로 이루어져 있을 때 권한 체크가 누락되는 경우를 방지하기 위해서 공통 모듈을 사용하는 것을 권장합니다.
샘플소스	**쿠키 조작 가능성**의 샘플 소스 참조

4-5. 일반계정 권한 상승 가능성

구분	내용
주요내용	사용자 계정으로 로그인 된 상태에서 관리자 page나 권한검증 부재로 인해 권한이 낮은 계정으로 접근이 허가 되지 않은 상위 권한 page에 접속이 가능하거나, 동일한 권한의 사용자 계정이지만 그룹, 팀, 파트너사 등으로 구분되어 있는 경우 접근제어 부재로 인해 허가 되지 않은 기능에 접근 가능한 경우가 존재합니다. 이는 접근제어통제(ACL)가 설정이 미흡하여 발생하는 취약점입니다.
대응방안	계정 인증과 연계하여 접근제어통제(ACL)를 정책적으로 수립하여 계정 권한에 맞게 page 접근제어를 하도록 설정해야 합니다.
샘플소스	**쿠키 조작 가능성**의 샘플 소스 참조