교육기관 홈페이지 취약점 심층점검 가이드

2019

교육부 사이버안전센터
Ministry of Education, Cyber Security Center

목 차

I. 배경 및 구성 ····· **3**

II. 입력값 검증 부재 ····· **4**

1. Injection ····· 4
2. 크로스 사이트 스크립트(XSS) ····· 16
3. 크로스 사이트 리퀘스트 변조(CSRF) ····· 20
4. 검증되지 않은 리다이렉트 및 포워드 ····· 26
5. 파일 업로드 ····· 30

III. 취약한 접근 통제 ····· **38**

6. 관리자 페이지 노출 ····· 38
7. 경로추적 및 파일 다운로드 ····· 43
8. 자동화 공격 ····· 49

IV. 취약한 인증 ····· **53**

9. URL/파라미터 변조 ····· 53
10. 불충분한 세션 관리 ····· 57
11. 쿠키 변조 ····· 61
12. 디폴트/취약한 계정 사용 ····· 64

V. 민감한 데이터 노출 ····· **67**

13. 부적절한 에러메시지 노출 ····· 67
14. 소스코드 내 중요 정보 노출 ····· 71
15. 중요 정보 비 암호화 통신 ····· 73

VI. 잘못된 보안 구성 ····· **75**

16. 디렉터리 인덱싱 ····· 75
17. 불필요한 Method 지원 ····· 79
18. 취약한 파일 존재 ····· 82
19. 히든필드 조작 ····· 85
20. 알려진 취약점이 있는 구성요소 사용 ····· 88

VII. 부록

1. 프록시 툴 사용 가이드 ····· 91
2. 홈페이지 취약점 자가점검시스템 사용자 가이드 ····· 99

교육기관 홈페이지 취약점 심층점검 가이드
관리요령 및 활용방안

- 본 자료는 교육부 사이버안전센터의 허가 없이 복제 · 복사하거나 개인 블로그 또는 개인 홈페이지 등 정보통신망에 유통되지 않도록 주의하여 주시고, 외부로 반출되지 않도록 보안 관리에 만전을 기해주시기 바랍니다.
- 교육부 정보보안기본지침 제51조(서버보안) 2항에 의거하여 홈페이지 등 자체 정보통신망을 대상으로 매년 정기적으로 보안 취약성 점검을 실시하여야 합니다.
- 이에 따라, 기관의 정보보안 담당자는 본 가이드를 활용하여 기관의 홈페이지 보안 취약점을 점검하고 보완 조치하여 주시기 바랍니다.
- 교육부 사이버 안전센터에서는 홈페이지 보안취약점 자가 점검시스템(cyber.ecsc.go.kr)을 운영하여 교육(행정)기관의 홈페이지를 점검 할 수 있도록 지원하고 있습니다.
- 예제로 삽입된 그림은 임의 수정하여 실제와 다를 수 있습니다.

I 배경 및 구성

우리 사회는 모든 주체가 사이버 공간을 향유하는 정보화 시대를 거쳐, 사물인터넷, 인공지능과 같은 신기술이 적용되는 4차 산업혁명 시대로 진입하고 있습니다. 이에 따라 신기술을 악용하여 사이버 공격 또한 지능화되고 신종위협을 끊임없이 발생하면서 위협 대응의 어려움은 지속적으로 증가하고 있습니다.

특히 홈페이지 보안 취약점을 악용한 해킹을 통해 정보시스템 파괴, 개인정보 유출, 홈페이지 위·변조 등의 피해를 발생시켜 정보시스템을 운영하는 기관의 대외 신뢰 하락과 금전적인 손실을 끼치고 있습니다.

이에 따라, 각 기관의 정보보안 담당자는 홈페이지 및 웹서버에서 발생하는 보안취약점에 대한 점검과 대응방안에 대해 숙지하고 미리 제거해 홈페이지 서비스의 안전성과 신뢰성을 확보하는 것이 매우 중요합니다.

본 가이드는 홈페이지 보안과 관련된 다양한 서적을 참고하여 교육(행정)기관 홈페이지 관리자가 홈페이지 해킹 등의 사고 예방을 위해 수행하여야 할 보안 취약점 점검 항목과 대응 방안을 담고자 노력하였습니다.

아울러, 교육부 사이버안전센터에서는 홈페이지 보안취약점 자가 점검시스템(cyber.ecsc.go.kr)을 운영하여 교육(행정)기관 담당자가 홈페이지 보안취약점을 스스로 점검하고 보완 조치할 수 있도록 지원하고 있습니다. 자세한 사항은 부록을 참고하시기 바랍니다.

정보보안 담당자께서는 본 가이드를 활용하여 수시로 소관 기관의 홈페이지 보안 취약점을 점검하고 보완하여 국민들이 안전하게 믿고 신뢰할 수 있는 정보시스템을 구축 운영하여 주시기 바랍니다.

Ⅱ 입력 값 검증 부재

개요
프로그램 입력 값에 대한 검증 누락 또는 부적절한 검증, 데이터의 잘못된 형식지정, 일관되지 않은 언어셋(Character Set) 사용 등으로 인해 발생되는 보안 약점으로 SQL 삽입, 크로스사이트 스크립트(XSS) 등의 공격을 유발할 수 있음

취약점 항목			
1	Injection	2	크로스 사이트 스크립트(XSS)
3	크로스 사이트 리퀘스트 변조(CSRF)	4	검증되지 않은 리다이렉트 및 포워드
5	파일 업로드		

1. Injection

분류	취약점 항목	위험도
입력 값 검증 부재	Injection	상

애플리케이션에서 서버로 전달되는 명령, 쿼리, 스크립트 등의 값을 변조하여 비정상적인 방법으로 시스템에 접근하는 공격 기법

1.1 SQL Injection

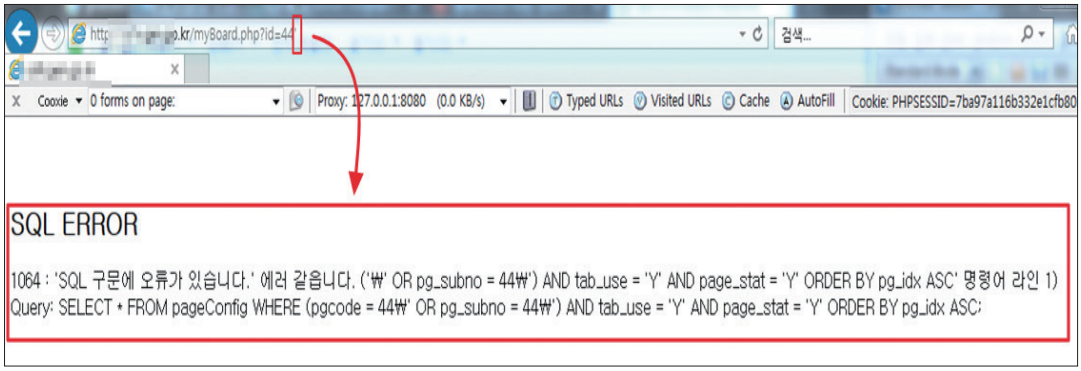
설 명	데이터베이스(DB)와 연동된 웹 애플리케이션에서 SQL 질의문에 대한 필터링이 제대로 이루어지지 않을 경우 공격자가 입력이 가능한 폼(웹 브라우저 주소 입력창 또는 로그인 폼 등)에 조작된 질의문을 삽입하여 웹 서버 데이터베이스 정보를 열람 또는 조작을 할 수 있는 취약점
점검목적	대화형 웹 사이트에 비정상적인 사용자 입력 값 허용을 차단하여 악의적인 데이터베이스 접근 및 조작을 방지하기 위함
점검내용	1) SQL 쿼리를 전달하여 DB 에러페이지가 반환되는지 확인 2) 로그인 창에 참이 되는 SQL 쿼리를 전달하여 로그인이 되는지 확인 3) 임의의 SQL 참, 거짓 쿼리에 따라 반환되는 페이지가 다른지 확인

▶ 점검 방법

1) 사용자 입력이 가능한 폼(URL파라미터, 검색 입력 폼 등)에 특수문자나 임의의 SQL 쿼리를 전달하여 DB 에러페이지가 반환되는지 여부 확인

점검 예)

http://www.점검사이트.es.kr/bbs/view.asp?Name=Notice&bbs=09"
http://www.점검사이트.es.kr/bbs/view.asp?bbs=01"&page=09



2) 로그인 입력 폼(아이디 혹은 패스워드)에 참이 되는 SQL 쿼리를 전달하여 정상 로그인 가능 여부 확인

점검 예)

아이디 : ' or 1=1#
비밀번호 : ' or 1=1#

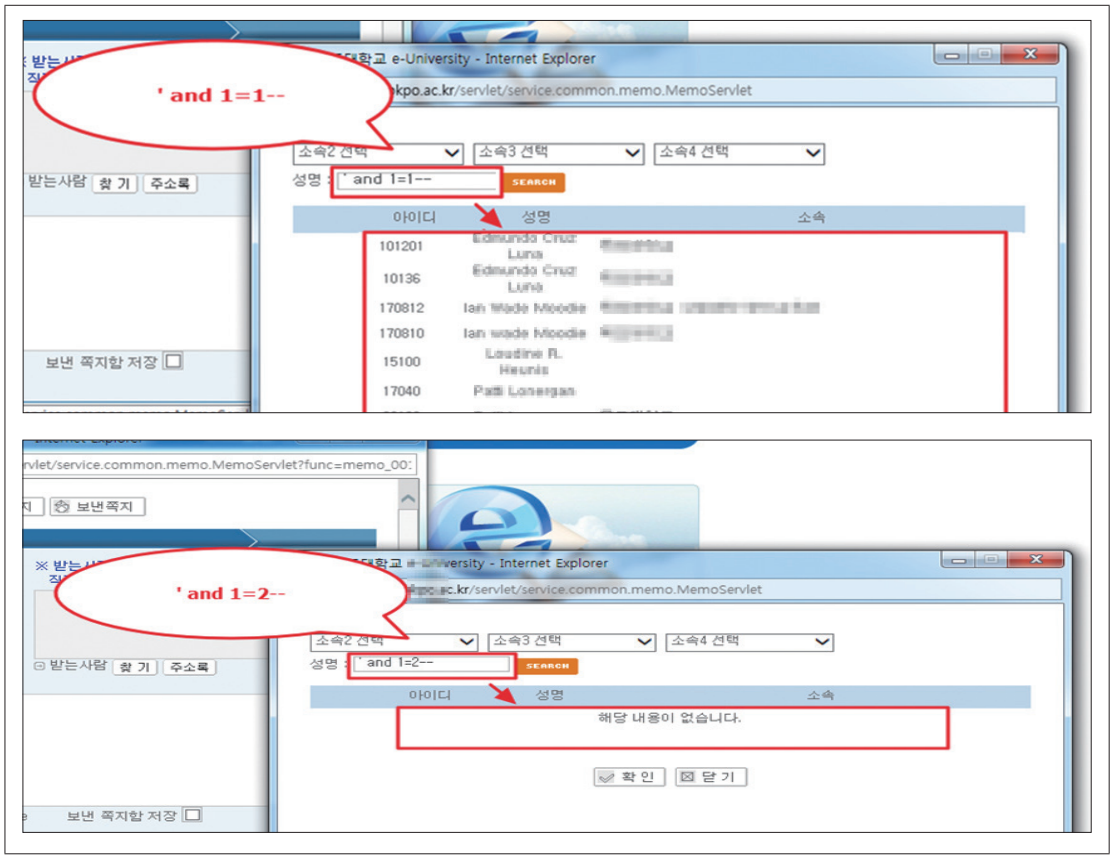
※ 다음과 같은 문자열을 이용하여 점검 가능

SQL 쿼리 구문 예시			
'or 1=1;--	or 1=1--	') or ('a'='a	+ or 1=1--
'' or 1=1--	' or 'a'='a	sql' or 1=1--	"
" or 1=1--	" or "a"="a	sql" or 1=1--	'

3) 사용자 입력이 가능한 폼(URL파라미터, 검색 입력 폼 등)에 임의의 SQL 참, 거짓 쿼리에 따라 반환되는 페이지가 다른지 확인

점검 예)

참 쿼리 : ' and 1=1--
거짓 쿼리 : ' and 1=2--



▶ 대응 방법

1) 웹 서버 내에서의 조치
 가) 웹 서버의 오류 정보가 사용자에게 노출되지 않도록 조치
 나) 웹 애플리케이션과 연동되는 데이터베이스의 접근 권한을 최소화
 다) 사용자 입력 폼(로그인 폼, 검색 폼, URL 등)을 대상으로 특수문자 필터링 규칙 적용
2) 홈페이지 개발 보안 조치
 가) 사용자로부터 입력되는 입력 값에 대한 검증과 예외처리
  ① ID, PASSWORD, 게시판, 제목, 본문, 검색창, 주소검색창 등의 모든 입력란에 악용 가능한 특수문자(SQL문에서 활용되는 특수문자 등)를 입력하지 못하도록 웹 서버의 소스코드를 수정
  ② 입력 값에 정의된 문자 길이를 검증하여 SQL문이 추가 삽입되지 않도록 구현
  ③ 파라미터가 숫자인 경우 isnumeric과 같은 함수를 이용하여 검증하며, 문자인 경우 정규표현식을 이용하여 특수문자를 치환
  ※ 특히, SQL문에서 활용되는 문자(', ", ;, --, # 등)는 반드시 치환

필터링 대상					
'	"	--	#	(	)
=	*/	/*	+	<	>
user_tables	user_table_columns		table_name	column_name	syscolumns
union	select	insert	drop	update	and
or	if	join	substrig	from	where
declare	substr	openrowset	xp_	sysobjects	%

나) Dynamic SQL 구문 사용을 지양하며 사용이 불가피할 경우, 파라미터 문자열 검사 필수 적용

안전한 코드의 예 JAVA

#1 문자열 유효성 검증 로직

```
1: public static String makeQuery(String str) {
2:     String result = "";
3:     if(str != null) {
4:             result = chkNull(replace(str, "'", ""));
5:             result = chkNull(replace(str, ";", ""));
6:             result = chkNull(replace(str, "--", ""));
7:             result = chkNull(replace(str, "|", ""));
8:             result = chkNull(replace(str, ":", ""));
9:             result = chkNull(replace(str, "+", ""));
10:            result = chkNull(replace(str, "₩", ""));
11:            result = chkNull(replace(str, "/", ""));
12:            result = chkNull(replace(str.toLowerCase(), "select", ""));
13:            result = chkNull(replace(str.toLowerCase(), "update", ""));
14:            result = chkNull(replace(str.toLowerCase(), "delete", ""));
15:            result = chkNull(replace(str.toLowerCase(), "insert", ""));
16:            result = chkNull(replace(str.toLowerCase(), "where", ""));
17:            result = chkNull(replace(str.toLowerCase(), "from", ""));
18:            result = "'"+result+"'";
19:            }
20:            return result;
21: }
22: public static String chkNull(string str) {
23:     if (str == null)
24:             return "";
25:     else
26:             return str;
27: }
```

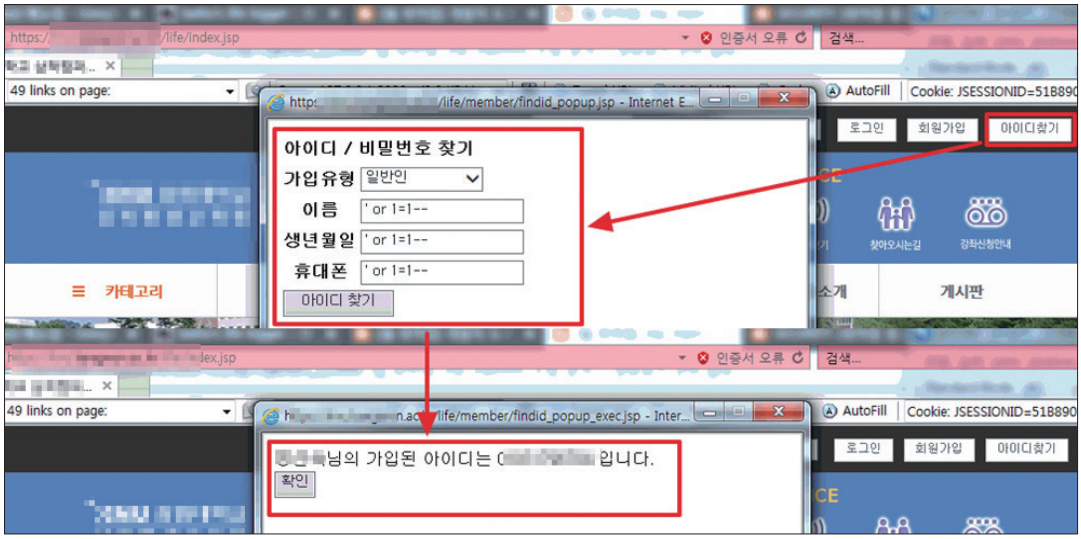
#2 Dynamic SQL 구문 사용 금지

```
1: try{
2:     String tableName = props.getProperty("jdbc.tableName");
3:     String name = props.getProperty("jdbc.name")
4:     String query = "SELECT * FROM ? WHERE Name = ?";
5:     stmt = con.perpareStatement(query);
6:     stmt.setString(1, tableName);
7:     stmt.setString(2, name);
8:     rs = stmt.executeQuery();
9:     ......
10: }
11: catch (SQLException sqle) { }
12: finally { }
```

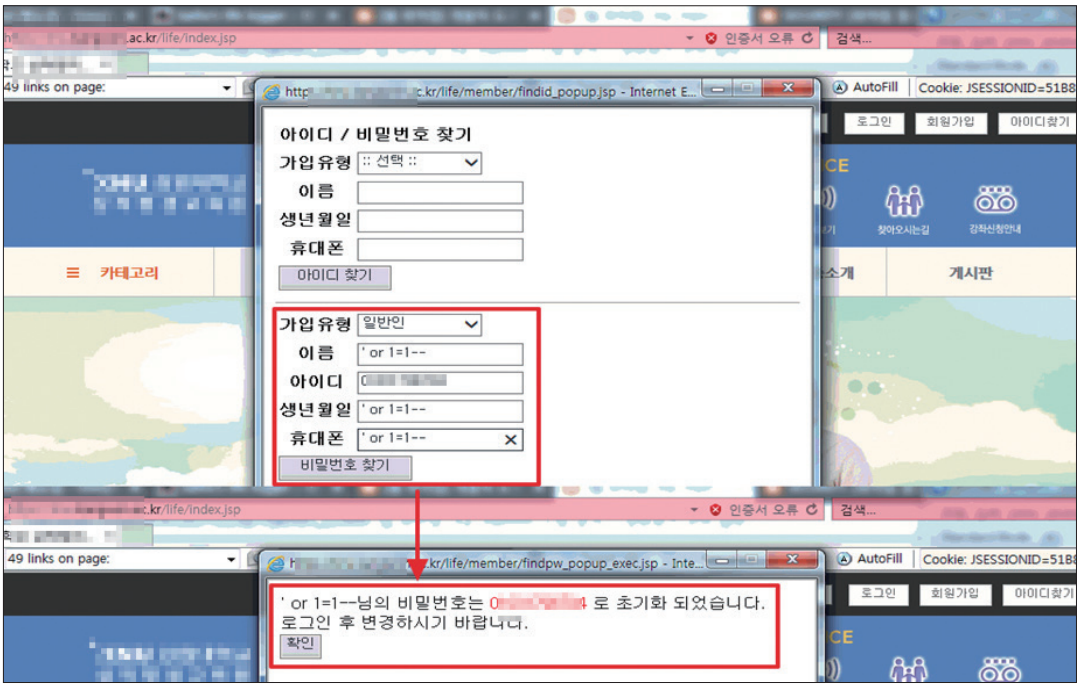
▶ 사례

- OO대학의 아이디 찾기 시 사용자 입력 폼에 SQL쿼리(' or 1=1--)를 입력하여 임의의 사용자 아이디 확인

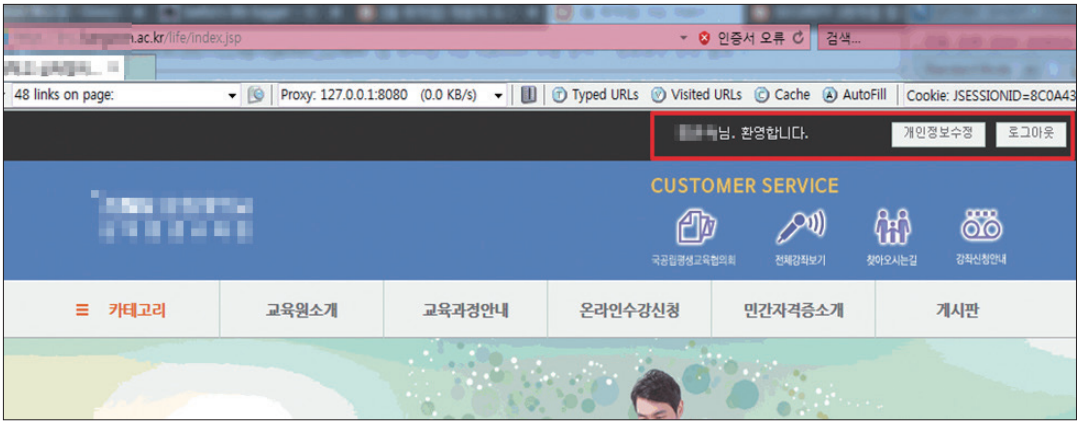
  ※ 데이터베이스(DB)의 사용자 테이블에서 첫 번째 레코드에 저장되어 있는 사용자 정보가 노출 됨



- 동일한 방식으로 비밀번호 찾기 시 아이디와 똑같은 비밀번호로 초기화 되는 것을 확인



- 확인 된 아이디/비밀번호로 정상 로그인 가능

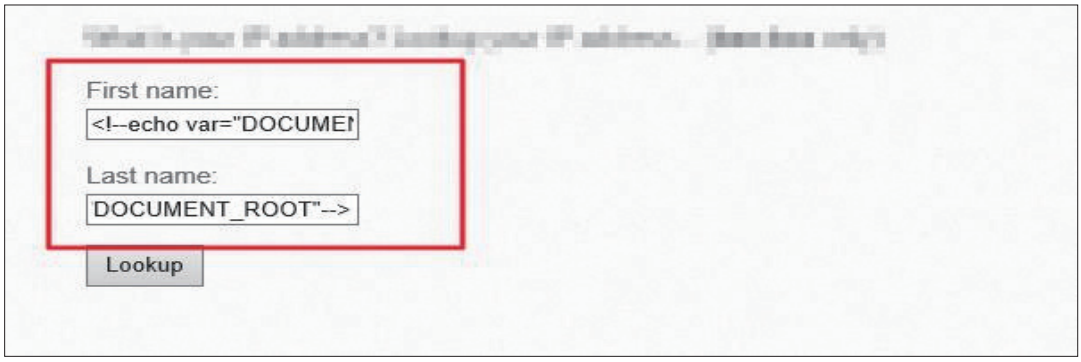


1.2 SSI Injection

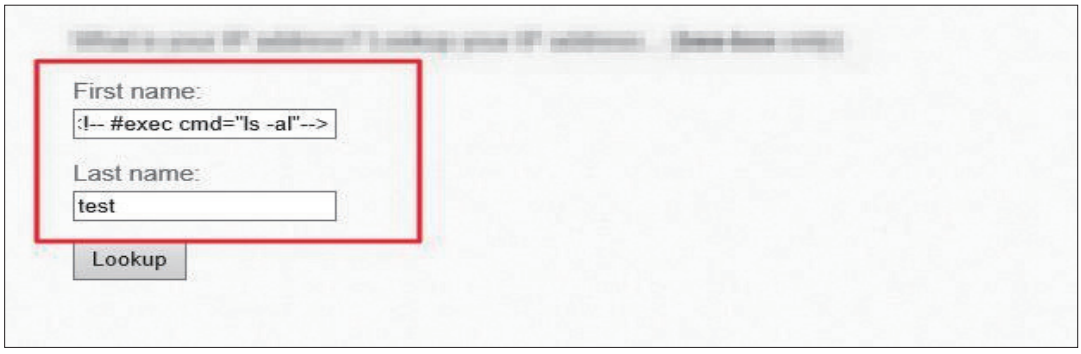
설 명	HTML 문서 내 입력받은 변수 값을 서버 측에서 처리할 때 부적절한 명령문이 포함 및 실행되어 서버의 데이터가 유출될 수 있는 취약점 *SSI(Server Side Includes) : CGI 프로그램이나 다른 동적인 기술로 페이지 전체를 만들어 서비스하지 않고도 HTML 페이지에 동적으로 생성한 내용을 추가할 수 있도록 웹 서버가 사용자에게 페이지를 제공하기 전에 구문을 해석하도록 지시하는 역할을 함
점검목적	적절한 입력 값 검증 절차를 마련하여 악의적인 파일을 include 시키지 못하도록 하여 불법적인 데이터 접근을 차단하기 위함
점검내용	1) SSI 명령어를 전달하여 실행되는지 확인

▶ 점검 방법

1) 입력 폼에 〈!--echo var="DOCUMENT_ROOT"--〉를 삽입하여 전송 시 반환되는 페이지에 사이트의 홈 디렉터리가 표시되는지 확인



2) 입력 폼에 〈!--exec cmd="ls -al"--〉를 삽입하여 전송 시 반환되는 페이지에 디렉터리의 파일 리스트가 표시되는지 확인



▶ 대응 방법

1) 홈페이지 개발 보안 조치
가) 사용자 입력으로 사용 가능한 문자들을 정해놓음
나) 정해진 문자들을 제외한 나머지 모든 문자들을 필터링
다) 필터링 해야 하는 대상은 GET 질의 문자열, POST 데이터, 쿠키, URL, 그리고 일반적으로 브라우저와 웹 서버가 주고받는 모든 데이터를 포함하며, 아래는 특수문자에 대한 Entity 형태를 표시한 것임

변경 전	<	>	"	(	)	#	&
변경 후	<	>	"	&#40	&#41	&#35	&amp

1.3 LDAP Injection

설 명	웹 애플리케이션에서 검증되지 않은 입력 값을 사용해서 동적으로 생성된 LDAP 구문에 의해 악의적인 LDAP 명령이 실행되어 개인정보 유출, LDAP 트리의 수정 및 삭제 등이 가능해지는 취약점 *LDAP(Lightweight Directory Access Protocol) : 조직이나 기업에서 내부 자원(파일, 장치 등) 등의 위치를 찾고 조작할 수 있게 하는 소프트웨어 프로토콜
점검목적	취약한 시스템에 신뢰할 수 없는 LDAP 코드 삽입 공격을 통한 비인가자의 악의적인 행위를 차단하기 위함
점검내용	1) 변조된 LDAP 쿼리를 전달하여 실행되는지 확인

▶ 점검 방법

1) 웹사이트의 사용자 인수 값을 입력받는 애플리케이션(폼필드, URL 등)에 변조된 LDAP 쿼리를 전송하여 실행되는지 확인

▶ 대응 방법

1) 홈페이지 개발 보안 조치
가) 사용자 입력 값을 White List로 지정하여 영문(a-z, A-Z)과 숫자(0-9)만을 허용
나) DN(Distinguished Name:디렉터리 항목을 고유하게 식별하는 이름)과 필터에 사용되는 사용자 입력 값에는 특수문자가 포함되지 않도록 특수문자 제거
다) 특수문자를 사용해야 하는 경우 특수문자(DN에 사용되는 특수문자는 '\', 필터에 사용되는 특수문자는 =, +, 〈, 〉, #, ;, \등)에 대해서는 실행 명령이 아닌 일반문자로 인식되도록 처리

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

1.4 XPath Injection

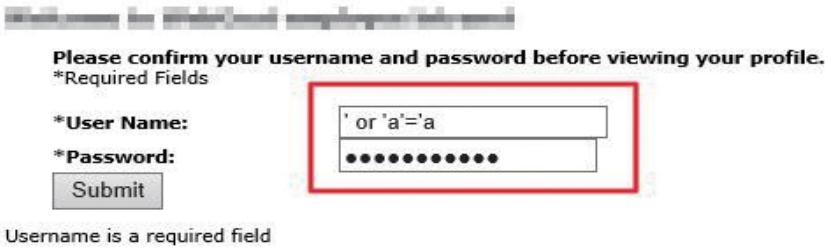
설 명	데이터베이스와 연동된 웹 애플리케이션에서 XPath(XML Path Language) 및 XQuery 질의문에 대한 필터링이 제대로 이루어지지 않을 경우 공격자가 입력이 가능한 폼(웹 브라우저 주소입력창 또는 로그인 폼 등)에 조작된 질의문을 삽입하여 인증 우회를 통해 XML 문서로부터 인가되지 않은 데이터를 열람 할 수 있는 취약점
점검목적	XPath 쿼리에 대한 적절한 필터링을 적용하여 웹사이트의 로직 손상 및 특정 데이터 추출을 차단하기 위함
점검내용	1) 로그인 창에 참이 되는 XPath 쿼리를 전달하여 로그인이 되는지 확인 2) 임의의 XPath 참, 거짓 쿼리에 따라 반환되는 페이지가 다른지 확인

▶ 점검 방법

1) 로그인 입력 폼(아이디 혹은 패스워드)에 참이 되는 XPath 쿼리를 전달하여 정상 로그인 가능 여부 확인

점검 예)

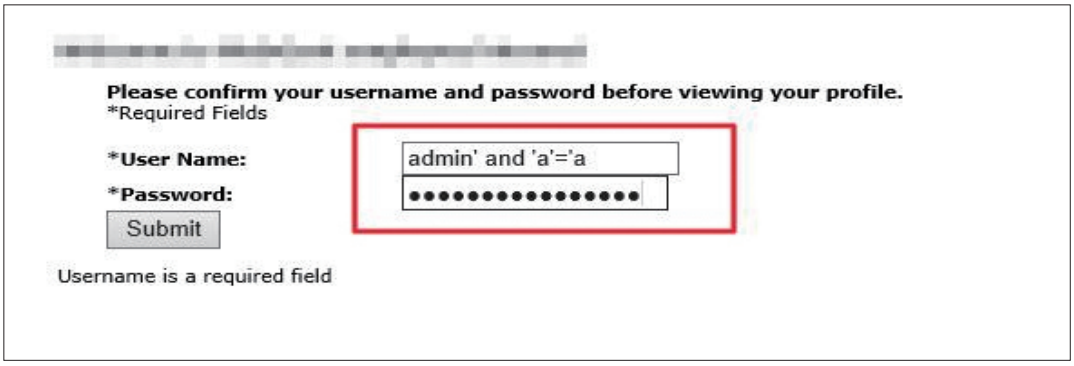
아이디 : ' or 'a'='a
비밀번호 : ' or 'a'='a



2) 사용자 입력이 가능한 폼(URL 파라미터, 검색 입력 폼 등)에 임의의 참, 거짓 쿼리에 따라 반환되는 페이지가 다른지 확인

점검 예 : ID, PW가 admin 일 때)

참 쿼리 : [ID: admin' and 'a'='a, PW: admin' and 'a'='a]
거짓 쿼리 : [ID: admin' and 'a'='a, PW: admin' and 'a'='b]



※ 다음과 같은 질의문을 이용하여 점검 가능

XPath 구문 예)

' or count(parent::*[position()=1])=0 or 'a'='b
' or count(parent::*[position()=1])>0 or 'a'='b
1 or count(parent::*[position()=1])=0
1 or count(parent::*[position()=1])>0

▶ 대응 방법

1) 웹 방화벽에서의 조치

가)모든 사용자 입력 폼(로그인 폼, 검색 폼, URL 등)을 대상으로 특수문자, 특수 구문 필터링 규칙 적용

필터링 대상				
'	,	=	(	)
[	]	/	:	*

2) 홈페이지 개발 보안 조치

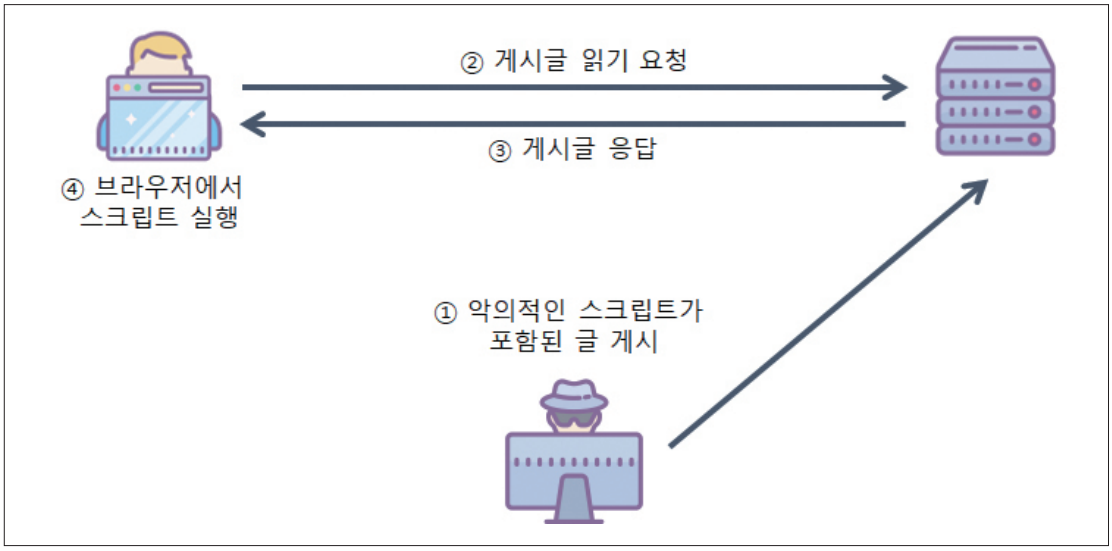
가) 서버로 전달되는 사용자 입력 값 검증 시 XPath 인젝션에 주로 사용되는 문자(', ", [등)들을 리스트로 지정하여 해당하는 문자가 존재할 경우 질의문 실행이 불가능하도록 설정

나) URLDecoder 클래스에 존재하는 decode 메소드를 통해 URL 인코딩이 적용된 사용자 입력 값을 디코딩함으로써 우회공격을 차단

다) XML 조회를 수행하는 쿼리문 작성 시 외부 입력 값이 쿼리문의 구조를 바꿀 수 없는 API(예. Java API - XQuery)를 사용

2. 크로스 사이트 스크립트(XSS)

분류	취약점 항목	위험도
입력 값 검증 부재	XSS	상



설 명	사용자 입력 값을 받는 웹 사이트의 게시판, URL 등에 악의적인 스크립트(자바스크립트, VB스크립트 등)를 삽입하여 게시글이나 이메일을 읽는 사용자의 쿠키(세션)를 도용하거나 악성코드를 유포할 수 있는 취약점
점검목적	웹 페이지 내 크로스 사이트 스크립팅 취약점을 제거하여 악성 스크립트의 실행을 차단하기 위함
점검내용	1) 웹 페이지 입력 폼에 스크립트를 저장하여 실행되는지 확인(Stored) 2) 웹 페이지 입력 값 변수에 스크립트를 전달하여 실행되는지 확인(Reflected)

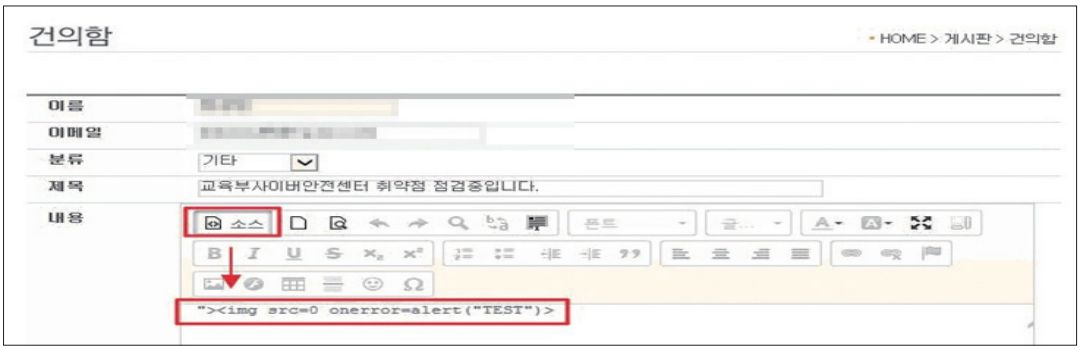
▶ 점검 방법

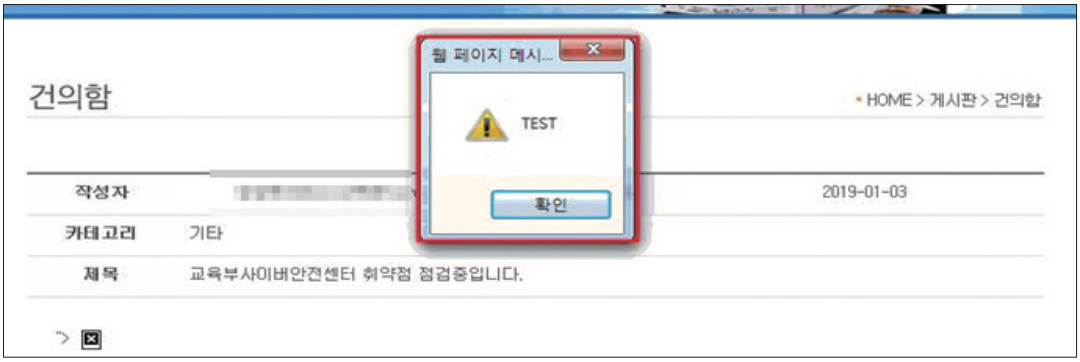
1) 사용자 입력이 가능한 폼(회원정보, 게시판, 댓글, 자료실 등)에 스크립트를 입력 후 저장하여 실행되는지 여부 확인

※ HTML 편집 기능이 있는 경우 기능 활성화 후 추가 확인 필요

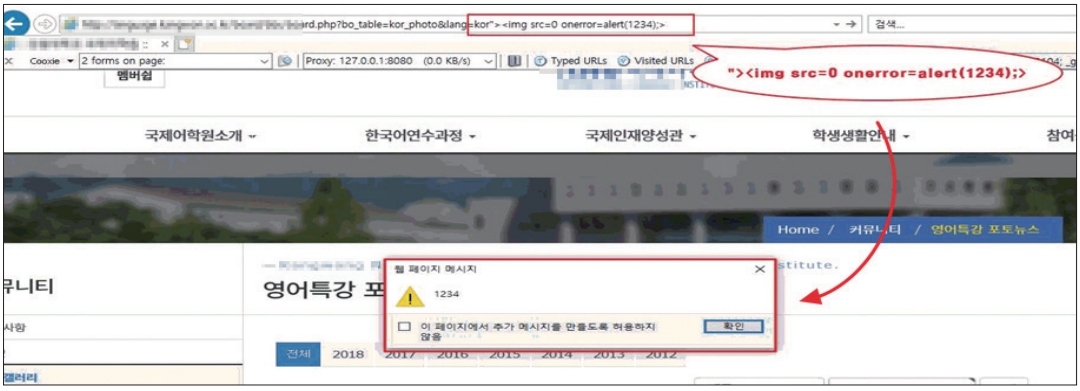
점검 예)

```
"><img src=0 onerror=alert("TEST")>
```





2) 사용자 입력이 가능한 폼(URL파라미터, 검색 입력 폼 등)에 스크립트를 입력하여 실행되는지 여부 확인



※ 다음과 같은 문자열을 이용하여 점검 가능

스크립트 구문 예시	
";alert(1111);" <script>alert(1111)</script><br">><script>alert(1111)</script><br"/><script>alert(1111)</script><br';</script><script>alert(1111)</script>	"><img src=0 onerror=alert(1111);> %22%3D%3Daalertlert(111)%2B%22 ;<sscriptcript>aalertlert(123)<sscriptcript><br">><iframe src=http://test.net><br">><embed src="http://test.net"></embed>

▶ 대응 방법

1) 웹 서버 내에서의 조치

가) 웹 서버에서 입력 값에 정의된 문자 길이를 검증하여 자바스크립트 등의 명령이 삽입되지 않도록 수정

나) 웹 서버의 검증·치환 등의 과정은 서버 사이드 스크립트(Server Side Script)에서 구현하여 검증·치환기능의 우회를 차단 (검증·치환 등의 기능을 자바스크립트로 구현할 경우 우회 가능)

다) 웹 서버에서 HTML 형식의 입력이 불가피할 경우만 XSS 공격에 주로 사용되는 Tag입력을 차단

라) 웹 서버는 사용자 입력 폼(로그인 폼, 검색 폼, URL 등)을 대상으로 특수문자, 특수구문 필터링 규칙 적용

마) 웹 서버의 취약점 조치를 완료한 후 위 과정을 다시 수행하여 XSS 취약점의 추가 존재여부를 재점검

필터링 대상				
<	>	<	>	onrowsinserted
vbscript	eval	onmousewheel	oncopy	onfocusin
expression	innerHTML	ondataavailable	oncut	onfocusout
applet	charset	onafterprint	onclick	onhelp
meta	document	onafterupdate	onchange	onkeydown
xml	string	onmousedown	onbeforecut	onkeypress
blink	create	onbeforeactivate	ondblclick	onkeyup
link	append	onbeforecopy	ondeactivate	onrowsdelete
style	binding	ondatasetchanged	ondrag	onload
script	alert	onbeforedeactivate	ondragend	onlosecapture
embed	msgbox	onbeforeeditfocus	ondragenter	onbounce
object	refresh	onbeforepaste	ondragleave	onmouseenter
iframe	cookie	onbeforeprint	ondragover	onmouseleave
frame	javascript	onbeforeunload	ondragstart	onbefore
frameset	void	onbeforeupdate	ondrop	onmouseout
ilayer	href	onpropertychange	onerror	onmouseover
layer	onpaste	ondatasetcomplete	onerrorupdate	onmouseup
bgsound	onstart	oncellchange	onfilterchange	onresizeend
title	onresize	onlayoutcomplete	onfinish	onabort
base	onrowexit	onmousemove	onfocus	onmoveend
onreset	onselect	oncontextmenu	onresizestart	onmovestart
onmove	onblur	oncontrolselect	onunload	onrowenter
onstop	onactivae	onreadystatechange	onselectstart	onsubmit
		onselectionchange		

2) 홈페이지 개발 보안 조치

가) 홈페이지 소스코드는 사용자가 입력한 문자열에서 [〈 , 〉 , & , “ , ”] 등을 replace등의 문자 변환 함수(혹은 Method)를 사용하여 [&lt, &gt, &amp, &quot] 로 치환

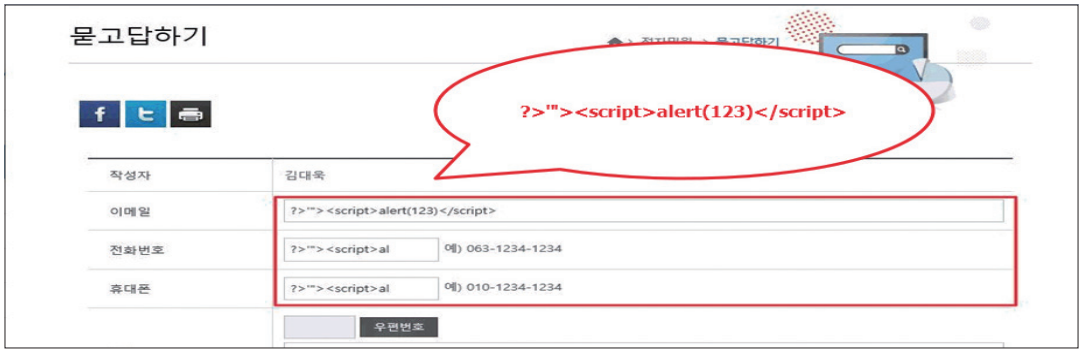
나) 홈페이지 게시판 등에서 HTML 태그 허용 시 HTML 태그의 리스트(White List)를 선정한 후, 해당 태그만 허용하는 방식 적용

안전한 코드의 예 JAVA

```
1: <%
2: String subject = request.getParameter("subject_BOX");
3:        subject = subject.replaceAll("<", "&lt");
4:        subject = subject.replaceAll(">", "&gt");
5:        .... ....
6: %>
```

▶ 사례

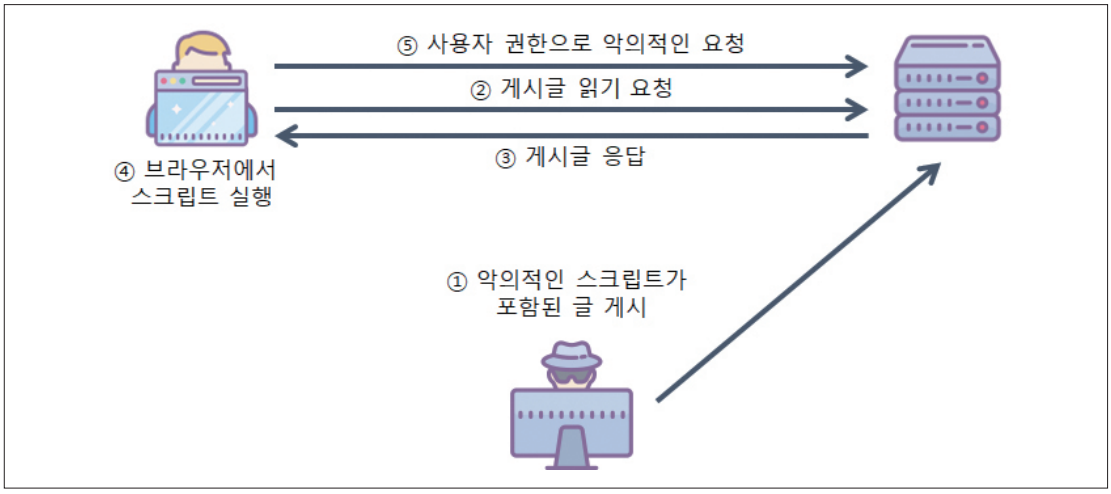
- OO대학의 묻고답하기 게시판에서 이메일, 전화번호, 휴대폰 입력 폼에 스크립트 구문 삽입 후 저장



- 저장된 게시글 열람 시 스크립트문 정상 실행 확인

3. 크로스 사이트 리퀘스트 변조(CSRF)

분류	취약점 항목	위험도
입력 값 검증 부재	CSRF	상

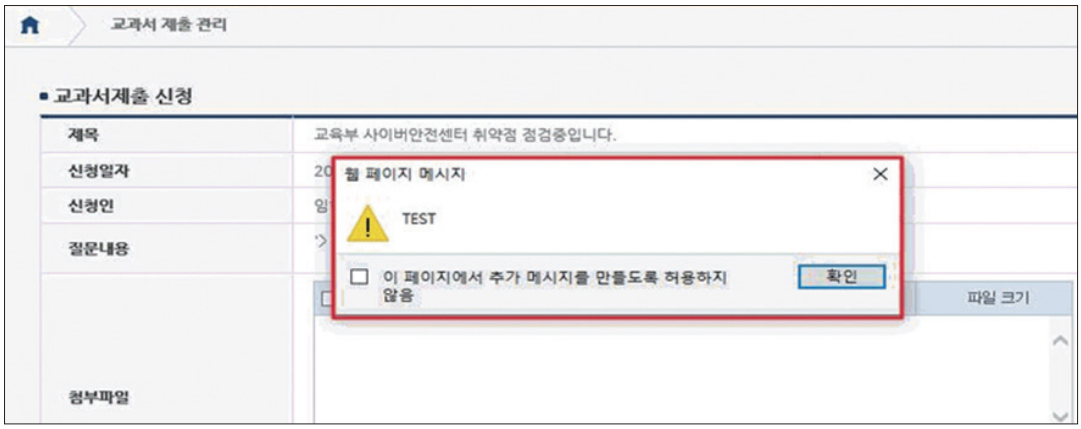


설 명	사이트 간 요청 위조(또는 크로스 사이트 요청 위조 Cross-site request forgery)는 사용자의 신뢰(인증) 정보 내에서 사용자의 요청을 변조함으로써 해당 사용자의 권한으로 악의적인 공격을 수행할 수 있는 취약점으로 사용자 의지와는 무관하게 공격자가 의도한 행위(수정, 삭제, 등록 등)를 특정 웹사이트에 요청하게 하는 공격기법
점검목적	사용자 입력 값에 대한 적절한 필터링 및 인증에 대한 유효성을 검증하여 신뢰(인증) 정보 내의 요청에 대한 변조 방지
점검내용	1) XSS 취약점 존재 여부 확인 2) 링크 클릭 시 의도하지 않은 요청이 수행되는지 여부 확인

▶ 점검 방법

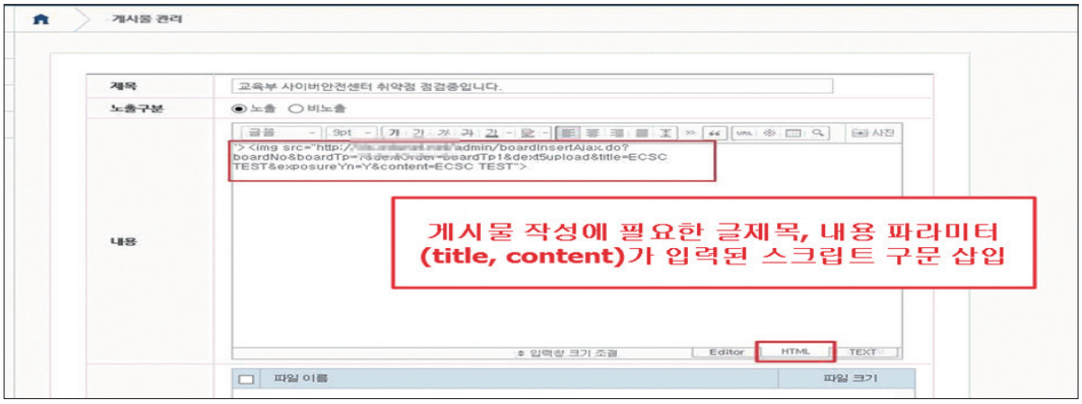
1) XSS 취약점 존재 여부 확인

가) 게시물(게시글, 답글, 수정 등) 작성 시 스크립트 삽입 후 실행 확인



나) XSS 취약점 존재 시 게시물 작성, 로그아웃, 계정 비밀번호 변경 등의 기능을 수행하는 요청 값을 스크립트로 삽입 후 저장

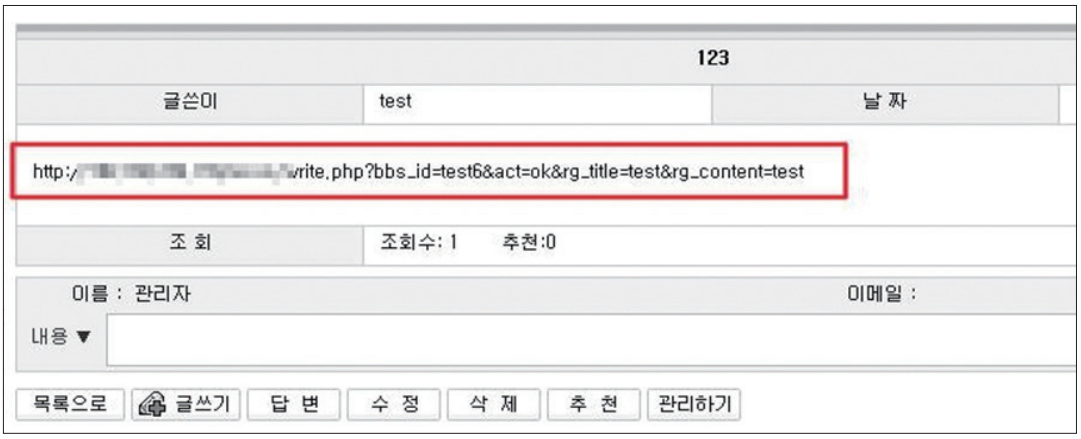
※ 특정 기능을 수행하기 위한 요청 값은 프록시(혹은 네트워크 캡쳐 프로그램)를 통해 확인 가능



다) 스크립트가 포함된 게시물을 열람하였을 경우, 열람한 사용자의 권한으로 웹 서버에 전송된 데이터가 정상적으로 수행되는지 점검

2) 서버요청에 필요한 인자가 포함된 링크 클릭 시 사용자의 권한의 요청이 정상 수행 되는지 여부 확인

가) 특정 기능(글 관리, 계정 관리 등)을 사용하기 위해 변조한 인수가 포함된 링크 업로드



나) 타 사용자(혹은 관리자)의 계정으로 해당 링크 클릭 시 해당 요청이 정상적으로 수행되는지 점검

▶ 대응 방법

1) 홈페이지 개발보안 조치

가) 사용자로 하여금 조작된 요청을 전송하지 않도록 사용자가 입력하는 값에 대한 검증 로직 설계

나) 정상적인 요청과 비정상적인 요청을 구분할 수 있도록 Form/URL에서 임의의 토큰을 추가하고 이 토큰을 검증하도록 구현

다) 정상적인 경로를 통한 요청과 비정상적인 경로를 통한 요청을 구분하기 위해 Referer 값 검증 로직 구현

라) HTML이나 자바스크립트에 해당되는 태그 사용을 사전에 제한하고 서버 단에서 사용자 입력 값에 대한 필터링 구현

마) HTML Editor 사용으로 인한 상기사항 조치 불가 시, 서버 사이드 서블릿/HTML Editor/DAO(Data Access Object) 영역에서 조치하도록 설계

바) XSS 취약점과 마찬가지로 사용자 입력 값에 대한 적절한 필터링 및 인증에 대한 유효성 검증을 통해 방지 가능

안전한 코드의 예 JAVA

#1 로그인 성공 시 로그인 처리와 함께 토큰을 생성하여 세션에 저장

```
// 무작위 문자열 생성을 위한 변수
char [] initRandomChar = {'A', 'B', 'C', 'D', 'F', 'G', 'H', 'I', 'J',
                          'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S',
                          'N', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', '0',
                          '1', '2', '3', '4', '5', '6', '7', '8', '9'};
// 랜덤 함수를 이용하여 무작위 문자열 생성
StringBuffer sb = new StringBuffer();
for ( int n=0; n<6; n++) {
    int index = (int) (initRandomChar.length * Math.random());
    sb.append(initRandomChar[index]);
}
// 토큰 생성
String CSRF_Token = sb.toString();
// 세션에 토큰 내용 저장
session.setAttribute("Token", CSRF_Token);
```

#2 웹 페이지 호출 시 URL을 통해서 토큰 전송

```
<input type="hidden" name="Token" value="<%=session.getAttribute("Token")%>">
```

#3 URL을 통해서 전송된 토큰과 세션에 저장된 토큰을 비교

```
// 서버로 전달된 파라미터를 변수에 저장
String token = request.getParament("Token");
// 서버로 전달된 토큰과 세션에 저장된 토큰이 다를 경우
```

```
4:  if ( !token.equals(session.getAttribute("Token")) ) {
5:     ... ...   // 에러 메시지 출력 (비정상적인 요청)
6:  }
```

▶ 사례

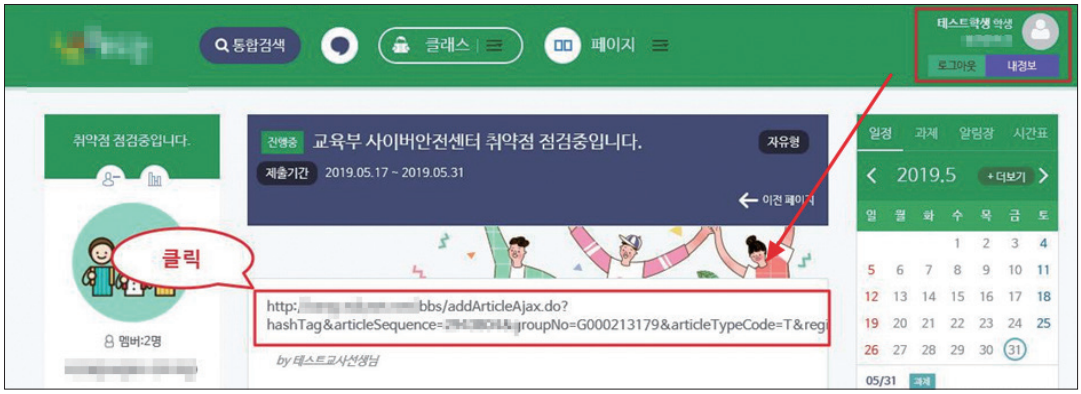
- OO기관 게시판에서 선생님 계정으로 글 작성 시도

- 프록시를 통해 글 작성 시 요청하는 인자들 확인 및 복사

```
GET
http[redacted]bbs/addArticleAjax.do?hashTag&articleSeque[redacted]&articleTypeCode=T&regi
sterDeviceCode&registrantMemberNo&attachCode&quizStr&bbsCategory&sendYn=N&fileAttach=N&imageAttach=N&albumName
&bbsOptionCode=N&content=test&poll_input&radio2=oc&radio3=1&openingConfigCode=P HTTP/1.1
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Accept: application/json, text/javascript, */*; q=0.01
X-Requested-With: XMLHttpRequest
Referer: [redacted]lass/G000213179/index.do
Accept-Language: ko-KR
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
Content-Length: 263
Host: [redacted]
Proxy-Connection: Keep-Alive
```

- 복사한 링크를 과제방 게시판에 작성 후 저장

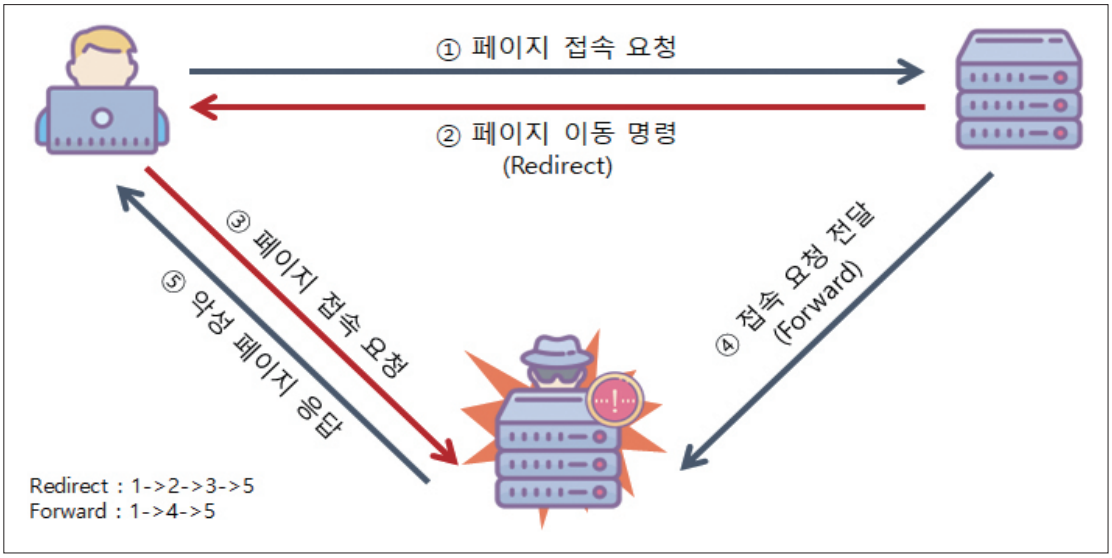
- 학생 계정으로 접속 후 게시판에 작성된 링크 클릭 시도



- 게시판 리스트에 의도하지 않은 게시글이 학생 계정으로 작성된 것을 확인

4. 검증되지 않은 리다이렉트 및 포워드

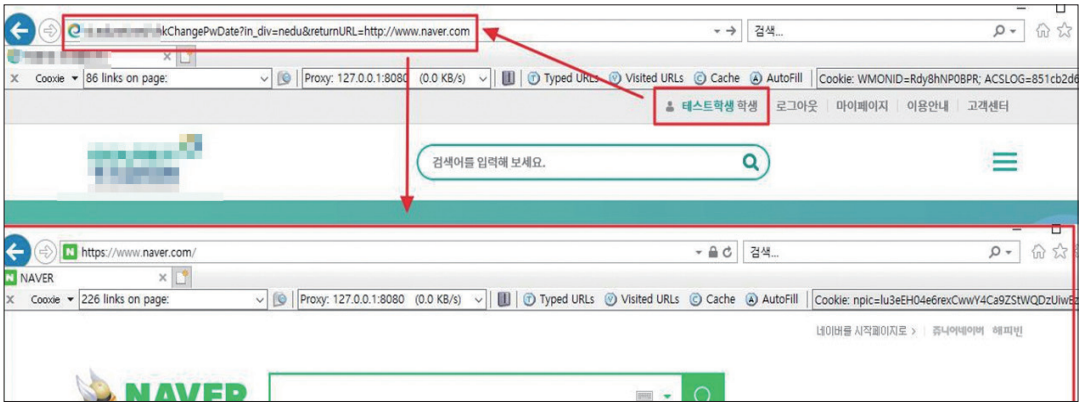
분류	취약점 항목	위험도
입력 값 검증 부재	검증되지 않은 리다이렉트 및 포워드	상



설 명	웹 애플리케이션에 접속한 사용자를 다른 페이지로 이동 시키는 경우, 이동되는 목적지에 대한 검증부재 시 피싱(Phishing), 악성코드 사이트 등의 접속 및 인가되지 않는 페이지로 접근이 가능한 취약점 *리다이렉트(Redirect) : 클라이언트에서 다른 페이지로 변경이 발생 *포워드(Forward) : 서버 단 자체에서 페이지 변경이 발생
점검목적	모든 웹 페이지에 대해 승인되지 않은 파라미터로 리다이렉트 및 포워드가 발생하는 것을 차단하여 피싱, 악성코드 사이트로의 연결을 방지하기 위함
점검내용	1) 페이지 이동을 위한 변수를 사용하는지 여부 확인

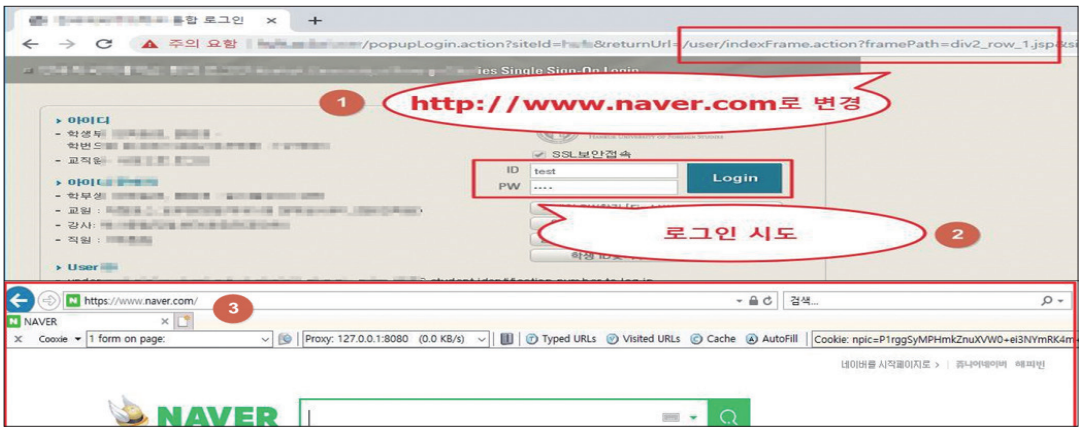
▶ 점검 방법

1) URL 파라미터에 페이지 이동을 위한 변수를 입력받는지 확인 후 (신뢰할 수 없는) 주소를 입력하여 리다이렉트 되는지 확인



2) 로그인 시 (신뢰할 수 없는) 사이트로의 이동이 되는지 여부 확인

가) 로그인 페이지에서 정상 로그인 후 페이지 이동을 위해 파라미터를 사용하는 경우, (신뢰할 수 없는) 사이트 주소 입력 후 로그인 시도



▶ 대응 방법

1) 홈페이지 개발 보안 조치

가) 리다이렉트 및 포워드 발생을 일으키는 파라미터 사용을 지양

나) 외부입력 값이 페이지이동(리다이렉트 또는 포워드)을 위한 URL로 사용되어야 하는 경우, 허용된 목적지(White-List)로만 이동할 수 있도록 설계

다) 페이지 이동을 허용하는 URL목록을 소스코드에 하드코딩 하거나, 설정파일(XML, properties)에 저장하여 허용된 URL로만 이동할 수 있도록 구현

안전한 코드의 예 JAVA

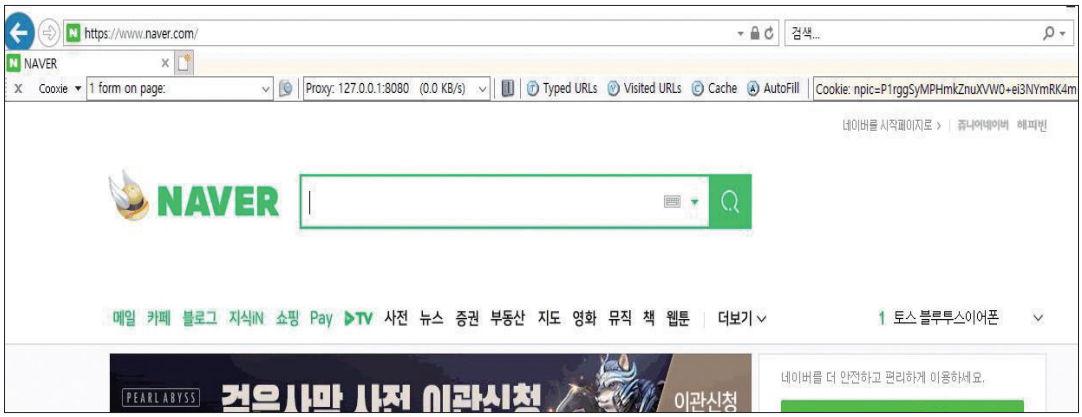
```
protected void doGet (HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

// 다른 페이지로 이동을 허용할 URL 리스트를 만듦
String allowURL[] = { "http://url1.com", "http://url2.com", "http://url3.com" }

// 입력받은 url은 미리 정해진 URL의 order로 받음
String nurl = request.getParameter("nurl");

try {
    int n = Integer.parseInt(nurl);
    if (n >= 0 && n < 3) {
        response.sendRedirect(allowURL[n]);
    }
} catch (NumberFormatException nfe) {
    // 입력 값이 숫자가 아닐 경우 에러 처리
}
}
```

▶ 사례

- OO기관 로그인 페이지에서 로그인 시도

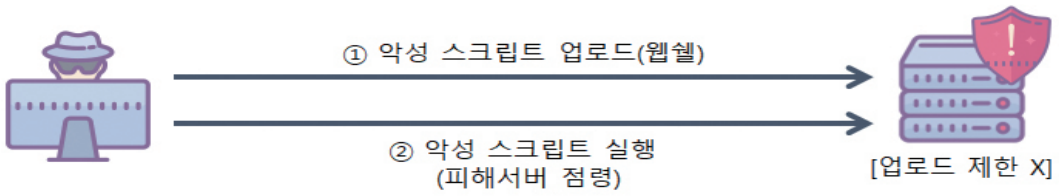
- 로그인 시 요청 파라미터에 페이지 이동을 위한 변수 ‘returnURL’ 확인 후 (신뢰할 수 없는) 사이트 주소를 입력하여 요청

- 아무런 검증 없이 (신뢰할 수 없는) 사이트로 리다이렉트 되는 것을 확인



5. 파일 업로드

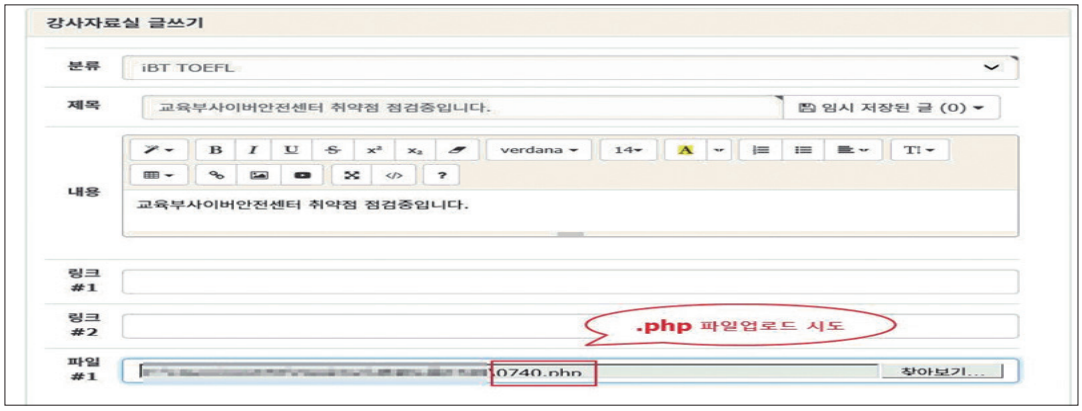
분류	취약점 항목	위험도
입력 값 검증 부재	파일 업로드	상



설 명	대부분의 홈페이지에 포함되어 있는 게시판 등과 같이 파일을 첨부할 수 있는 기능을 가진 웹페이지에 존재하며, 서버 측에서 실행될 수 있는 스크립트 파일(asp, jsp, php 파일 등)이 업로드 가능하고, 이 파일을 공격자가 웹을 통해 직접 실행시킬 수 있는 경우 시스템 내부명령어를 실행하거나 외부와 연결하여 시스템 제어가 가능한 취약점
점검목적	업로드 되는 파일의 확장자에 대한 적절성 여부를 검증하는 로직을 통해 공격자가 조작된 Server Side Script 파일 업로드 방지 및 서버 상에 저장된 경로를 유추하여 해당 파일 실행을 불가능하게 하기 위함
점검내용	1) Server Side Script 확장자(jsp, asp, php 등) 파일 업로드 가능 여부 확인 2) 업로드 된 Server Side Script 파일의 실행 가능 여부 확인

▶ 점검 방법

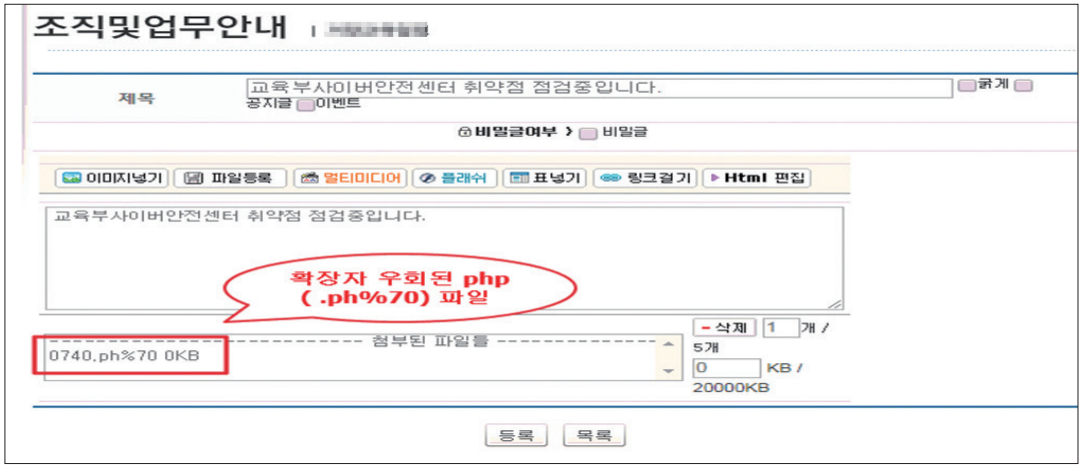
1) 파일 업로드(첨부)가 가능한 게시판에 Server Side Script 파일(asp, php, php3, jsp, cgi 등) 업로드가 가능한지 여부 확인
※ 운영 중인 서버의 개발언어에 맞는 Server Side Script를 업로드 함

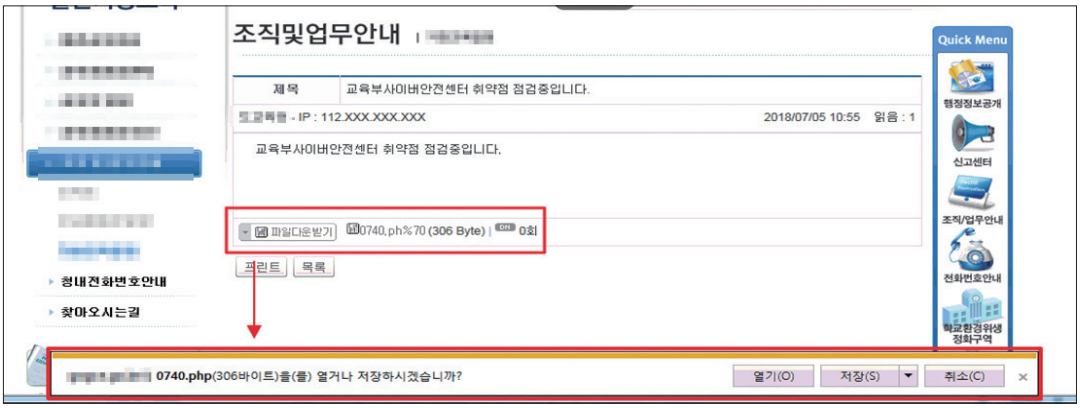


2) 파일 업로드 시 다양한 우회방법을 적용하여 업로드 가능 여부 확인

가) 확장자 우회(.js%70, .jsp;.gif, %22js%20, jpg.jsp, jsp.%00.jpg 등) 업로드

※ 단, 업로드 차단 기능이 Client Side Script(자바스크립트, VB스크립트)로 구현되었을 경우 기능 수정 후 업로드 시도



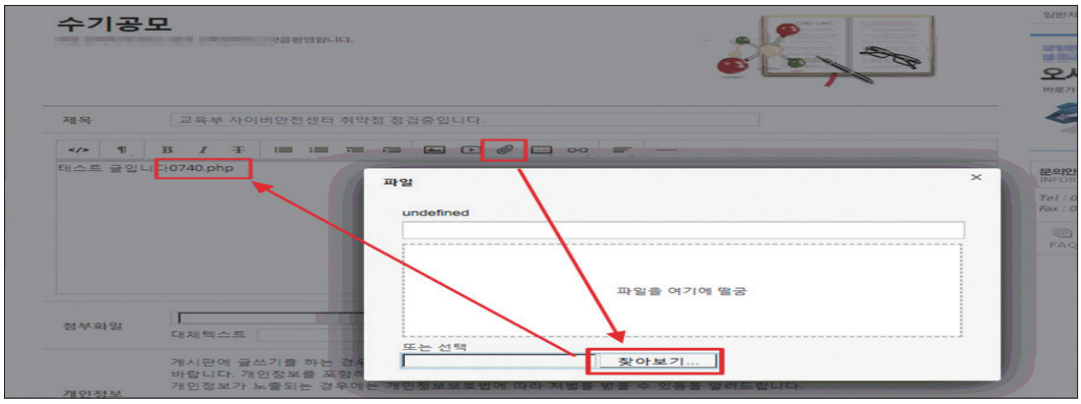


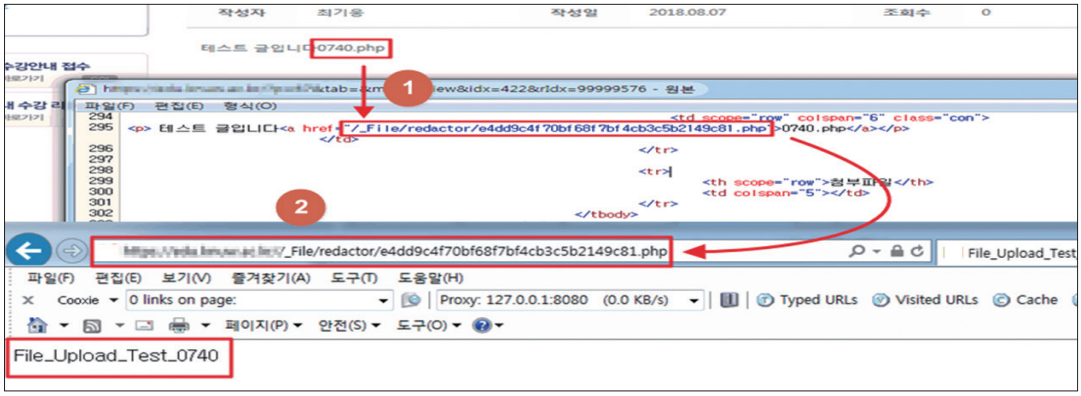
나) MIME-TYPE 우회 업로드

※ Content-Type을 허용 가능한 이미지 형식 MIME-TYPE(image/jpg, image/png 등)으로 변경

3) 업로드 된 Server Side Script 파일의 경로를 확인하고 해당 경로로 접속 시 파일이 실행되는지 점검

※ 파일 업로드 후 소스코드(혹은 속성정보 등) 내 포함된 링크를 통한 경로 확인 가능





우회방법	업로드 파일명	설명
URL 인코딩	test.js%70	URL 인코딩을 이용하여 확장자 필터링 우회
대소문자 혼합	tets.JsP	대소문자 혼합을 통해 우회
NULL 문자 추가	test.jsp%00.jpg	NULL(%00)문자를 추가해 확장자 필터링 우회
공백 문자 추가	test.jsp%20.jpg	공백 문자(%20)를 추가해 확장자 필터링 우회
특수문자 추가	test.jsp;jpg	IIS 6.0의 취약점인 ; 문자로 필터링 우회

[업로드 취약점 점검용 파일명 예시]

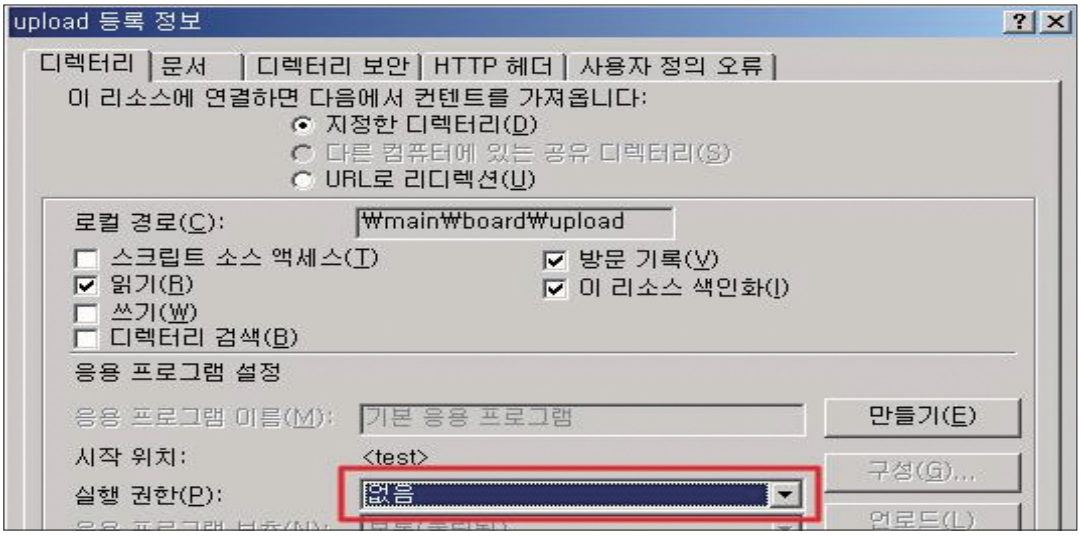
▶ 대응 방법

1) 웹 서버 내에서의 조치

가) 웹 서버 설정을 변경하여 업로드 된 해당 파일의 실행권한을 차단

① IIS 웹 서버 조치 방법

[제어판] → [관리도구] → [인터넷 서비스 관리자] → 업로드 폴더 선택 → [속성] 클릭 → 마우스 오른쪽 버튼 클릭 → [등록 정보] → [실행권한] → [없음] 선택



② Apache 웹 서버 조치 방법

아파치 웹 서버의 설정 파일인 httpd.conf 파일에서 Options의 'IncludesNoExec' 지시어 추가

```
<Directory "업로드를 금지할 디렉터리">
        AddType application/x-httpd-php3-source .php3 .php .phps .ph
    .cgi .jsp .inc .htm .html shtml
      Options IncludesNoExec

</Directory>
```

2) 홈페이지 개발 보안 조치

가) 우회 기법을 통한 악의적 파일 업로드를 차단

① 파일 업로드 가능 여부를 검증하는 기능을 서버 사이드 스크립트(Server Side Script)로 구현하여 우회 기법을 통한 업로드 공격을 사전에 차단

② 자바스크립트로 필터링 기능을 구현할 경우 사용자가 임의로 수정 및 삭제 할 수 있으므로 차단기능을 우회 가능

③ 파일 업로드 필터링 방식은 White-List 방식(업로드 가능한 확장자만 허용)을 이용하여 확장자 변경 등의 우회 기법을 차단

나) 파일이 업로드 되는 디렉터리(위치 및 파일명)가 사용자에게 노출되지 않도록 조치

① 파일명과 확장자를 외부사용자가 추측할 수 없는 문자열로 변경하여 저장하고 실제 파일명은 데이터베이스에 보관하는 등 정보를 이원화하여 운영

다) 저장 경로는 'web document root' 밖에 위치시켜 웹을 통한 직접 접근 차단

라) 파일 실행여부를 설정할 수 있는 경우 실행 속성을 제거

안전한 코드의 예 JAVA

```
1: MultipartHttpServletRequest mRequest = (MultipartHttpServletRequest) request;
2: String next = (String) mRequest.getFileNames().next();
3: MultipartFile file = mRequest.getFile(next);
4: if ( file == null ) return ;
5:
6: // 업로드 파일 크기 제한
7: int size = file.getSize();
8: if ( size > MAX_FILE_SIZE ) throw new ServletException("Error");
9:
10: String fileName = file.getOriginalFilename().toLowerCase();
11:
12: // 화이트리스트 방식으로 업로드 파일의 확장자를 체크
13: if ( fileName != null )
14: {
15:     if(fileName.endsWith("*.doc") || fileName.endsWith("*.hwp")
        || fileName.endsWith("*.pdf") || fileName.endsWith("*.xls"))
16:     {
17:       ....
18:     }
19:     else throw new ServletExeption("Error");
20: }
21:
22: // 업로드 파일의 디렉터리는 'web document root' 밖에 위치
23: File uploadDir = new File("/app/webapp/data/upload/notice");
24: String uploadFilePath = uploadDir.getAbsolutePath()+"/" + fileName;
```

확장자 필터링 목록 예시

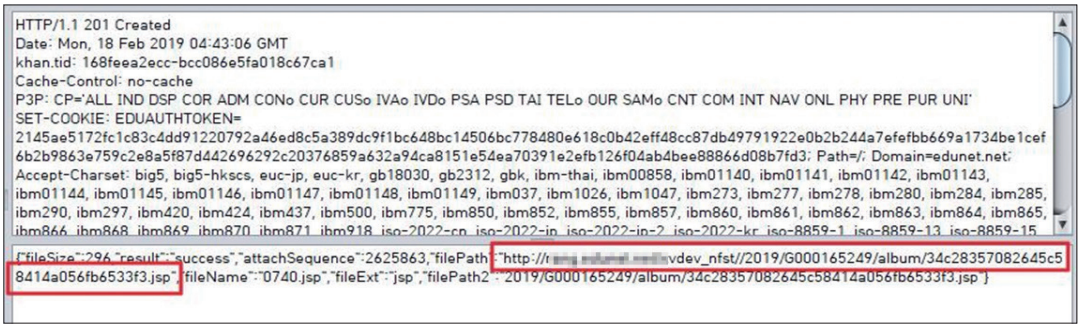
html, htm, php, php2, php3, php4, php5, phtml, pwml, inc, asp, aspx. pscx, sjp, cfm, cfc, pl, bat, exe, com, dll, vbs, js, reg, cgi, asis, sh, shtml ,shtm, phtm 등

▶ 사례

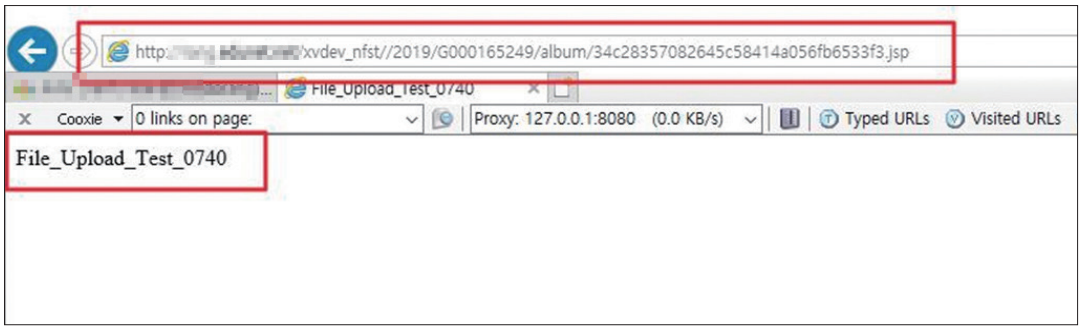
- OO기관의 사진등록 시 확장자 우회(.jsp%00.jpg)를 통한 파일 업로드 시도

- 파일 업로드 요청 값에서 파일명의 확장자를 .jsp로 변경 후 요청

- 파일 업로드 응답 값에서 파일이 업로드 된 경로(filePath) 정보 확인 가능



- 확인된 경로로 접속 시 Server Side Script 파일 실행 가능

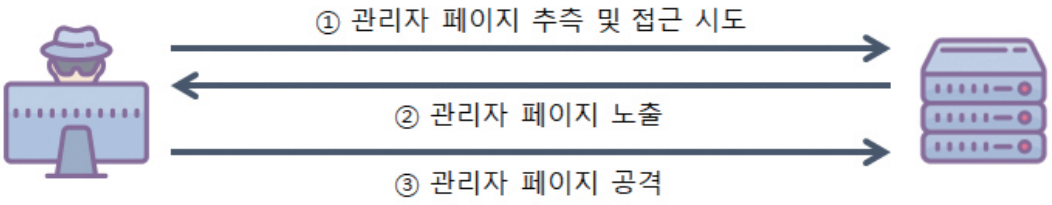


Ⅲ 취약한 접근 통제

개요			
인증된 사용자가 수행할 수 있는 작업에 대한 제한이 제대로 적용되지 않을 경우 발생되는 보안 약점으로 공격자는 이러한 결함을 악용하여 다른 사용자의 계정에 액세스하거나, 중요한 파일을 읽고, 데이터를 수정하거나 접근 권한을 변경하는 것이 가능함			
취약점 항목			
6	관리자 페이지 노출	7	경로추적 및 파일 다운로드
8	자동화 공격		

6. 관리자 페이지 노출

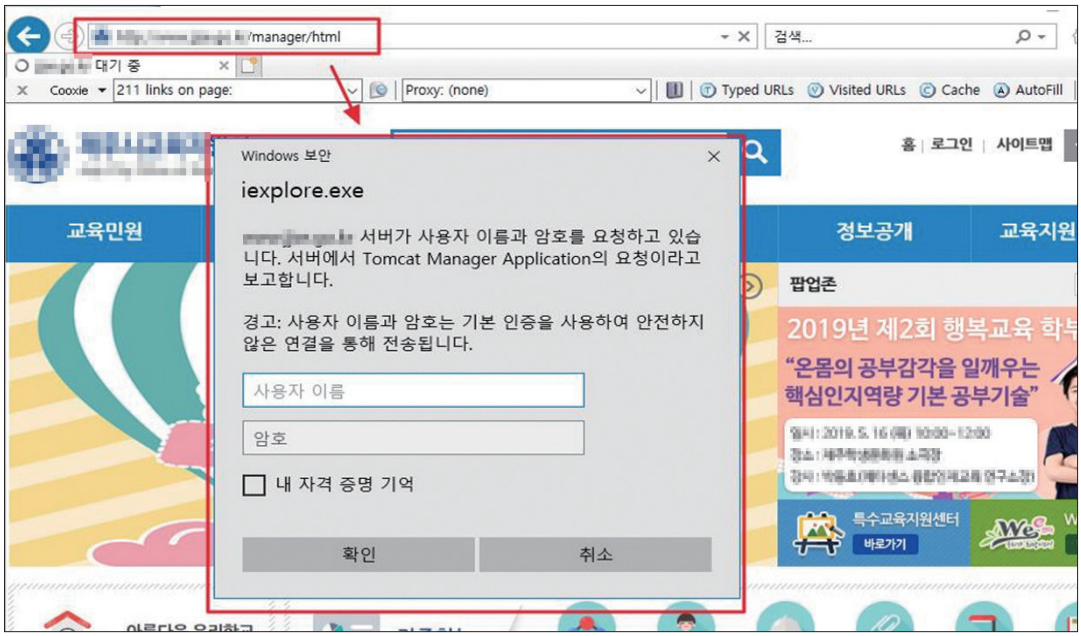
분류	취약점 항목	위험도
취약한 접근 통제	관리자 페이지 노출	중



설 명	웹 애플리케이션의 전반적인 기능 설정 및 회원 관리를 할 수 있는 관리자 페이지가 추측 가능한 형태로 구성되어 있을 경우 공격자가 관리자 페이지에 쉽게 접근을 할 수 있으며 무차별 대입 공격, SQL 인젝션을 통하여 관리자 권한을 획득할 수 있는 취약점
점검목적	관리자 페이지 URL을 유추하기 어려운 이름으로 설정하고 접근을 제어하여 비인가자의 관리자 권한 획득을 방지하기 위함
점검내용	1) 유추하기 쉬운 URL로 관리자 페이지 및 메뉴 접근의 가능 여부 점검 2) 구글 고급 검색을 통한 관리자 페이지 노출 확인

▶ 점검 방법

1) 직접 접근을 통한 점검
가) 관리자 페이지 위치를 알지 못할 경우 일반적으로 많이 사용하는 관리자 페이지 명을 입력하여 관리자 페이지가 존재하는지 점검



※ 일반적으로 많이 사용하는 관리자 페이지 명

관리자 페이지 주소 예시	
http://admin.ecsc.es.kr http://www.ecsc.es.kr/admin/ http://www.ecsc.es.kr/manager/ http://www.ecsc.es.kr/master/ http://www.ecsc.es.kr/webmaster/	http://www.ecsc.es.kr/system/ http://www.ecsc.es.kr/adm/ http://www.ecsc.es.kr/administrator/ http://www.ecsc.es.kr/masterpage/ http://www.ecsc.es.kr/super/

2) 구글 고급 검색을 통한 관리자 페이지 노출 확인

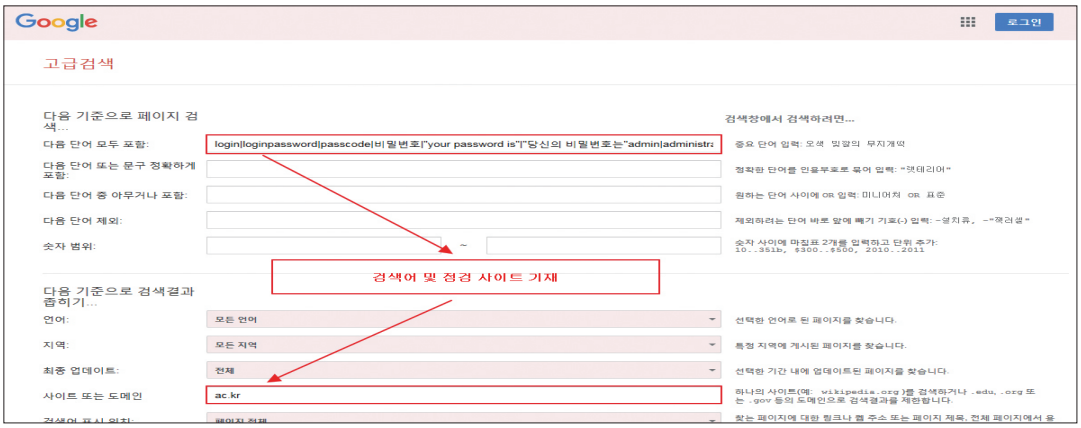
가) 구글(www.google.co.kr) 사이트에 접속 후 고급 검색으로 이동

나) 도메인 설정에 해당 웹 서버 주소를 입력하고 검색어 입력란에는 다음의 검색어를 각각 입력하여 ID/PW 및 관리자 웹 서버(관리자 로그인 페이지 등) 노출 페이지를 검색

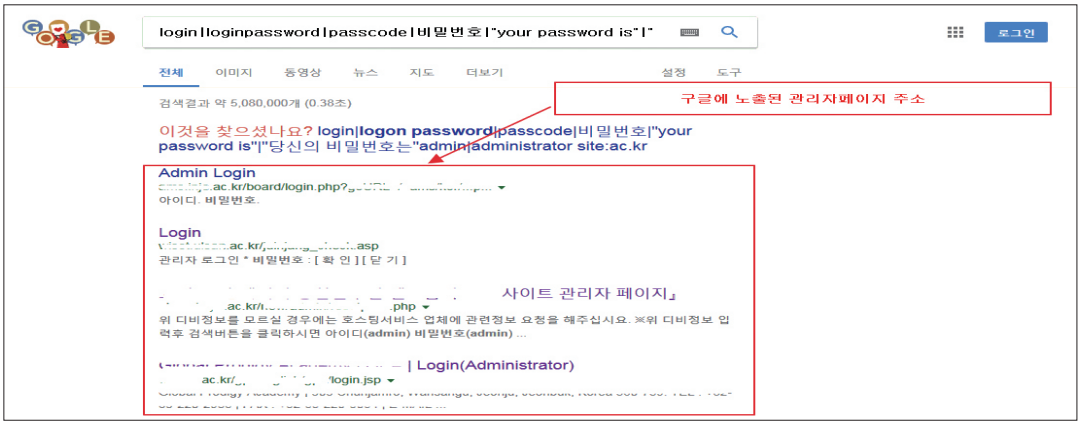
비밀번호 검색 예)

login|logon
password|passcode|비밀번호|"your password is"|"당신의 비밀번호는"
admin|administrator

※ 검색어는 공백(빈칸)이 포함되지 않아야 하며, 공백을 포함하기 위해서는 인용부호(")를 이용
(예. "your password is")



다) 검색 결과에서 해당 사이트의 관리자페이지가 노출 되었는지 확인



▶ 대응 방법

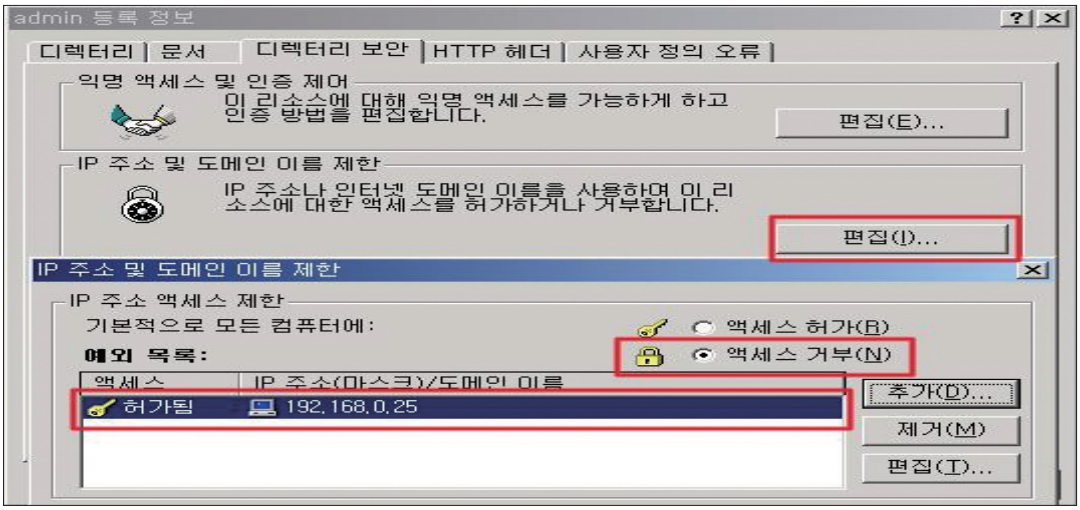
1) 웹 서버 내에서의 조치

가) 홈페이지 관리자 페이지는 관리용으로 지정된 디렉터리에만 보관하여 운영

나) 홈페이지 관리자 페이지에 임의의 사용자가 접근할 수 없도록 접근권한을 설정하여, 접근할 수 있는 권한을 가진 단말기만 접근 가능 하도록 설정

① IIS 웹 서버 조치 방법

[제어판] → [관리도구] → [인터넷 서비스 관리자] → 관리자 디렉터리 선택 후 마우스 오른쪽 버튼 클릭 → [등록 정보] → [디렉터리보안] → [IP주소 및 도메인 이름제한] → [편집] → '액세스 거부(N)' 체크 후 관리자 IP만 허용



② Apache 웹 서버 조치 방법

아파치 웹 서버의 설정 파일인 httpd.conf 파일에서 관리자 디렉터리에 'Deny', 'Allow' 지시자를 이용하여 접근 제한 설정
예) /usr/local/www/admin 폴더를 192.168.10.10만 허용하고 모두 차단할 경우

```
<Directory "/usr/local/www/admin/">
  Order allow,deny
  Deny from all
  Allow from 192.168.10.10
</Directory>
```

③ Tomcat 조치 방법

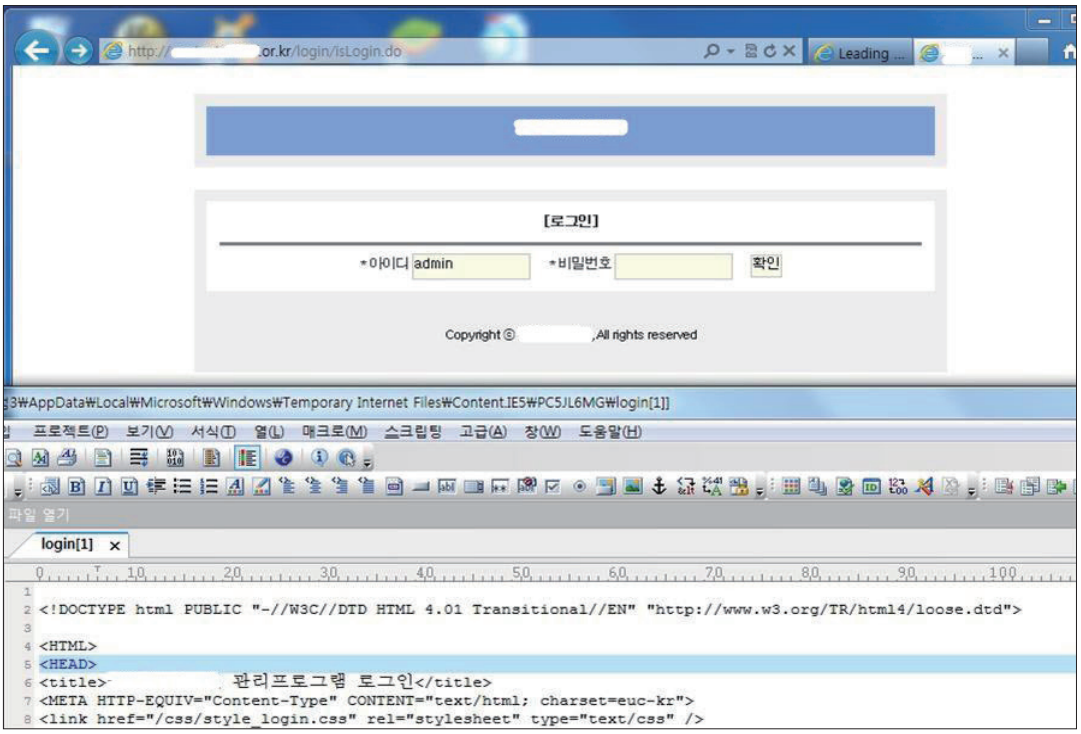
별도 관리자 페이지 디렉터리 접근 제한을 위하여 admin.xml 파일 등 별도의 설정 파일 생성 후 다음과 같은 설정을 함

```
<Context path="/admin" docBase="../webapps/admin" debug="0"
    privileged="true">
    <Valve className="org.apache.catalina.valves.RemoteAddrValve"
                allow="127.0.0.1"/>

</Context>
```

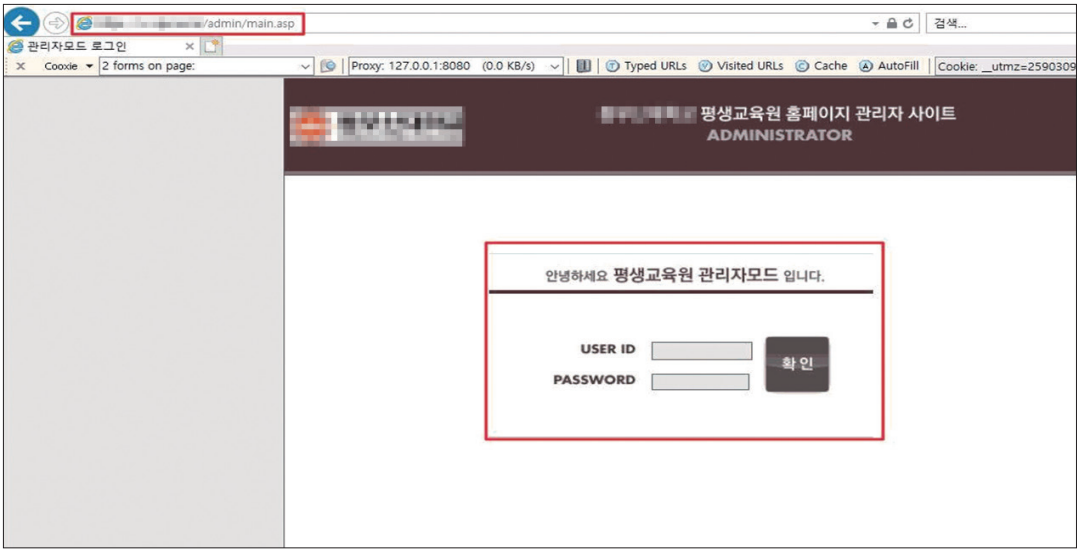
2) 구글 검색기에 노출된 경우의 조치
가) 구글에 노출된 홈페이지 관리자 페이지 정보의 캐시 삭제를 요청
나) 웹 서버에 노출 방지 표준(인터넷검색엔진배제표준)을 이용하여 개인정보가 포함된 주소를 지정하는 robots.txt 파일을 만들어 가상서버의 최상단 폴더에 저장하거나 해당 페이지의 HTML 안에 메타태그를 입력

▶ 사례

- OO기관의 경우 관리자 외의 IP에서 관리자페이지로 접근이 가능하며, 소스보기로 관리프로그램 로그인 페이지 확인

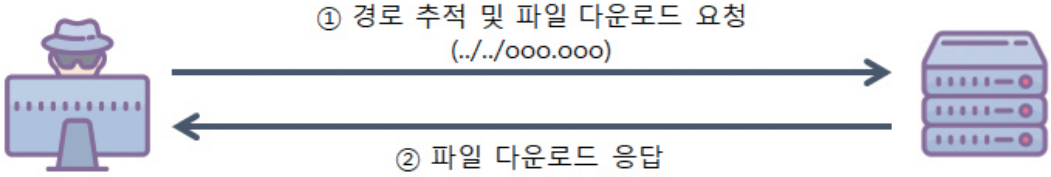


- OO대학은 유추하기 쉬운 URL을 사용하여 관리자페이지의 직접 접근이 가능



7. 경로추적 및 파일 다운로드

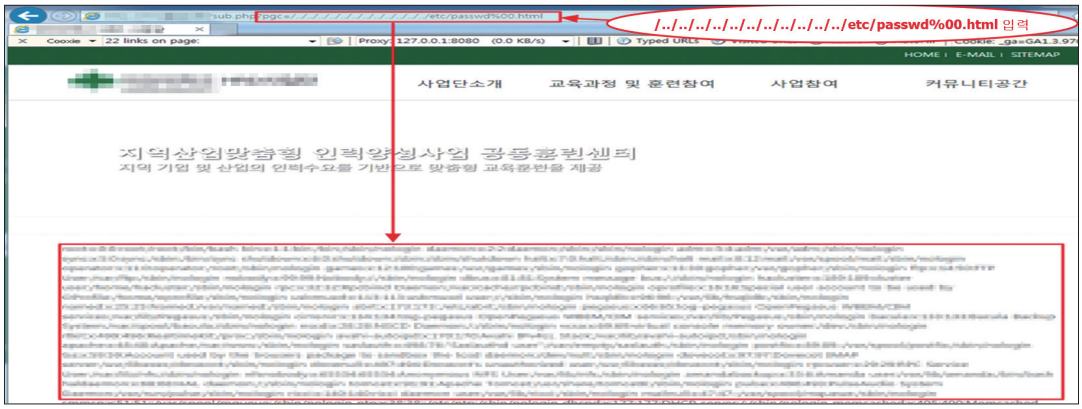
분류	취약점 항목	위험도
취약한 접근 통제	경로추적 및 파일 다운로드	상



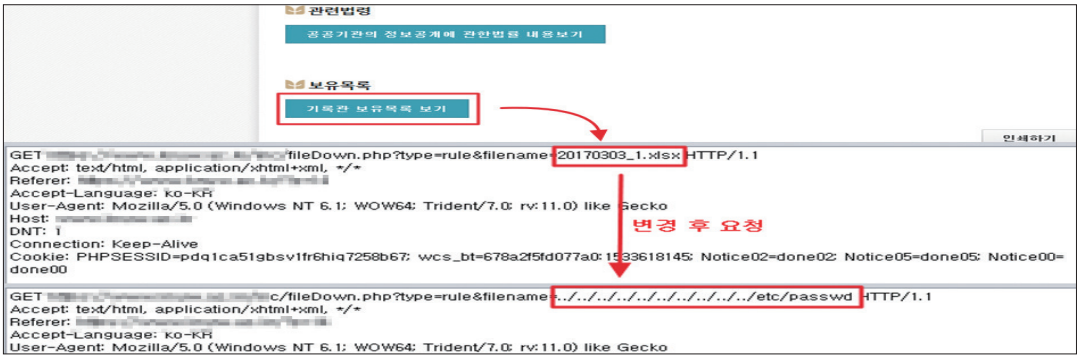
설 명	경로추적을 통해 인증되지 않은 사용자가 시스템에 접근하여 중요한 파일을 읽거나 권한 없는 기능 등을 수행할 수 있으며, 파일 다운로드 기능이 존재하는 웹 애플리케이션에서 파일 다운로드 시 파일의 경로 및 파일명을 파라미터로 받아 처리하는 경우, 파일에 대한 접근 권한이 설정되어 있지 않아 공격자가 파라미터를 조작하여 중요파일(환경설정, 웹 소스코드, 데이터베이스 등)을 다운로드 받을 수 있는 취약점
점검목적	경로추적 및 파일 다운로드를 통한 중요 파일 접근을 제한하여 허용되지 않은 파일을 열람하거나 다운로드받는 것을 방지하기 위함
점검내용	1) 경로 추적을 통한 중요 파일 열람 가능 여부 확인 2) 파일다운로드 기능을 이용한 중요 파일 다운로드 가능 여부 확인

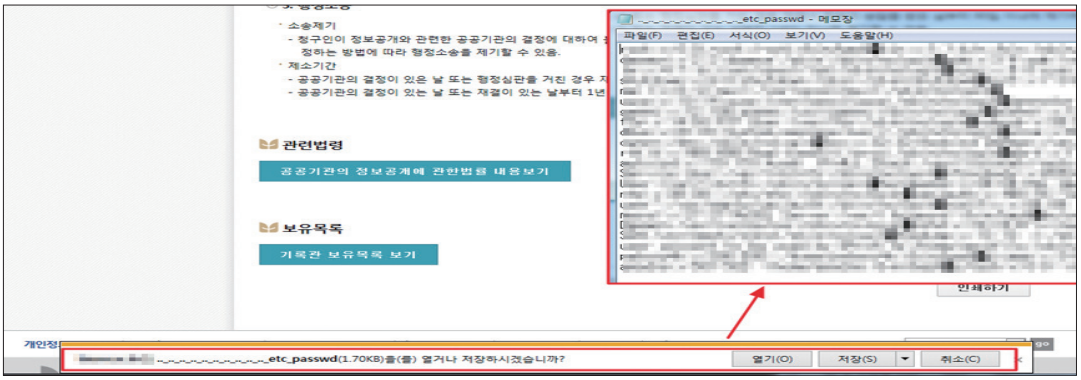
▶ 점검 방법

1) 서버의 데이터를 웹 브라우저에 로드하는 경우 데이터를 지정하는 파라미터에 ../../(경로추적)을 통한 중요파일 접근 시 해당 경로의 파일 내용이 웹 브라우저에 표시되는지 확인



2) 파일 다운로드 기능이 있는 페이지(게시판 등)에서 파일 경로 및 파일명을 변경하여 해당 파일이 다운로드 되는지 확인

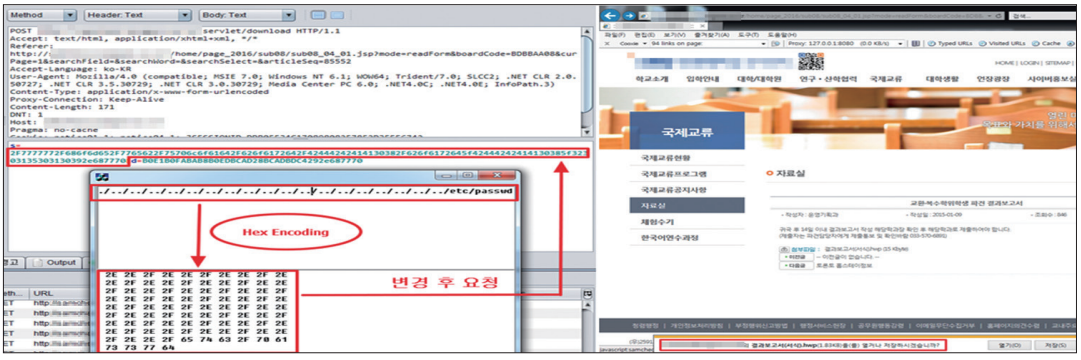




※ 다운로드 요청 시 파일경로 및 파일명을 인코딩(hex, URL 등)하여 파일다운로드 가능 여부 추가 점검 필요

인코딩 예)

URL인코딩 - .(%2e), /(%2f), ₩(%5c)
16bit 유니코드인코딩 - .(%u002e), /(%u2215), ₩(%u2216)
더블URL인코딩 - .(%252e), /(%252f), ₩(%255c)
경로치환 - ...//, ...₩₩, ...₩/, .../₩



파일경로명	설 명
/etc/passwd	UNIX 시스템 사용자 계정 정보
/etc/shadow	UNIX 시스템 사용자 패스워드 정보(패스워드는 암호화되어 있음)
/etc/hosts	UNIX 시스템 인접 시스템들 도메인 이름 및 IP정보
/etc/hosts.allow /etc/hosts.deny	UNIX 시스템 TCP/IP 상의 접근제어 설정 정보
/etc/fstab /etc/vfstab /etc/inet/vfstab	UNIX 시스템 디스크 설정 정보

[다운로드 점검 필요 주요 시스템 파일]

파일경로명	설 명
~./sh_history ~./bash_history	UNIX 시스템 사용자 쉘 명령어 실행 이력 기록
/var/log/dmesg	UNIX 시스템 부팅 로그 기록
/var/log/messages /var/log/messages.1 ...	UNIX 시스템 로그 기록
.mysql_history	사용자의 MySQL 데이터베이스 작업 이력 기록

[시스템 사용자 정보 수집용 파일]

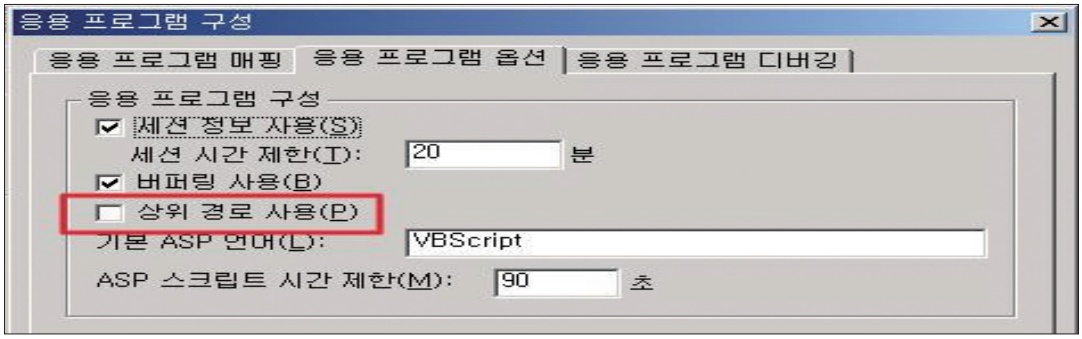
▶ 대응 방법

1) 웹 서버 내에서의 조치

가) 파일 내용을 웹 브라우저에 표시 할 수 있는 디렉터리를 특정 디렉터리로 한정하고 이 외의 다른 디렉터리에서는 파일 내용을 표시할 수 없도록 ../ 등의 상위 경로 접근이 제한되도록 설정

① IIS 웹 서버 조치 방법

[제어판] → [관리도구] → [인터넷 서비스 관리자] → 웹 사이트 선택 후 마우스 오른쪽 버튼 클릭 → [등록 정보] → [홈 디렉터리] → [구성] 버튼 선택 → [응용 프로그램 옵션] → '상위 경로 사용(P)' 체크 해제



나) 웹 사이트에서 접근하려는 파일이 있는 디렉터리에 chroot 환경을 적용해서 경로 추적 공격을 최소화

※ chroot 환경 : chroot 디렉터리는 해당 디렉터리가 루트처럼 다뤄짐. chroot 파일 시스템은 대부분의 유닉스를 기반으로 한 플랫폼에서 지원이 가능하고, 윈도우 플랫폼에서는 적절한 시작 디렉터리를 새로운 논리 드라이브로 만들어 웹 사이트에서 해당 드라이브를 통하여 접근하게 함
예) 웹 사이트의 최상위 폴더를 웹 사이트 Root 폴더로 제한

다) PHP언어로 개발된 서버의 경우 php.ini 에서 magic_quotes_gpc를 On으로 설정하여 '. \ 와 . /' 값 입력 시 치환되도록 설정
라) 파일 다운로드의 절대 경로 설정 및 DocBase의 상위경로 또는, 타 드라이브로 설정을 변경
마) 다운로드 경로 정보를 자바스크립트나 js 소스에서 확인 가능하지 않게 제한하며, 웹 서버 서블릿 내부 또는 별도의 설정 파일에서 관리

2) 홈페이지 개발 보안 조치
가) 외부 입력 값에 대해 파일 경로를 변경할 수 있는 문자열 "..", "/", "\"에 대해 필터링
나) 다운로드 파일 이름을 데이터베이스에 저장하고 파일 다운로드 시 파일 경로와 파일명이 아닌 키 값을 설정하여 다운로드 되도록 구현
다) 다운로드를 제공하는 페이지의 유효 세션 체크 로직 필수 적용

안전한 코드의 예 JAVA

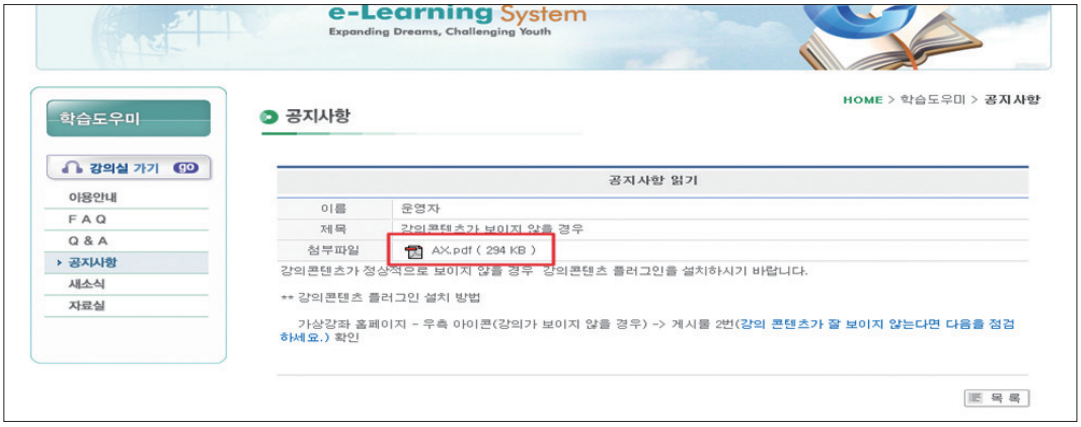
```
1:  String filename = response.getParameter("filename");
2:  String filepathname = UPLOAD_PATH + filename;
3:
4:  // 경로 추적 체크
5:  if(filename.equalsIgnoreCase(".") || filename.equalsIgnoreCase("/")
6:    || filename.equalsIgnoreCase("₩"))
7:  return 0;
8:
9:  // 파일 전송 루틴
10:  response.setContentType("application/unknown; charset=euc-kr");
11: response.setHeader("Contetn-Disposition","attachment;filename=" + filename +
";");
12: response.setHeader("Content-Transfer-Encoding:" , "base64");
13:
14: try {
15: BufferenputStraem in = new BufferedInputStream(new
FileInputStream(filepathname));
16:  ....
17: } catch (Exception e) {
18:   // 에러 체크 [파일 존재 유무 등]
19: }
```

※ 필터링 대상

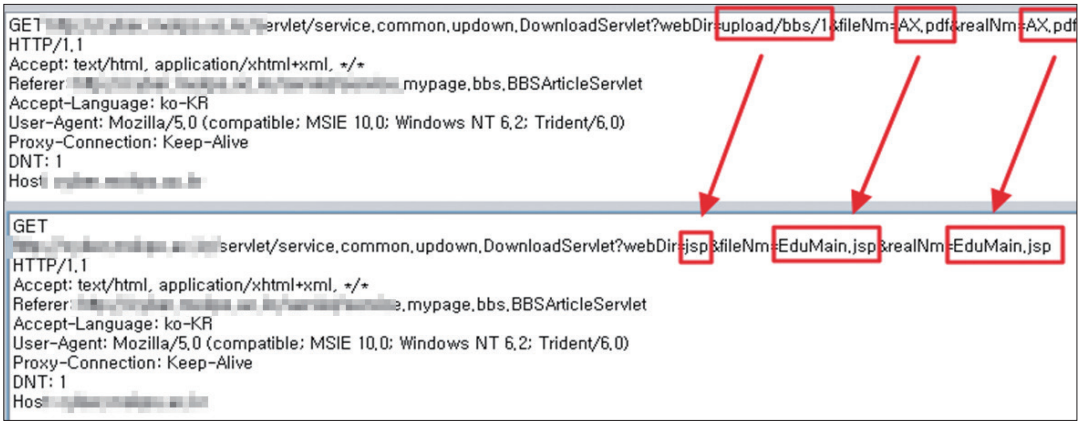
문자	설명
.	Path Traversal 가능성의 확인
/	특정 Path의 접근 가능성을 확인
₩	운영환경에 따른 Path 접근 확인
%	UTF 인코딩 파라미터

▶ 사례

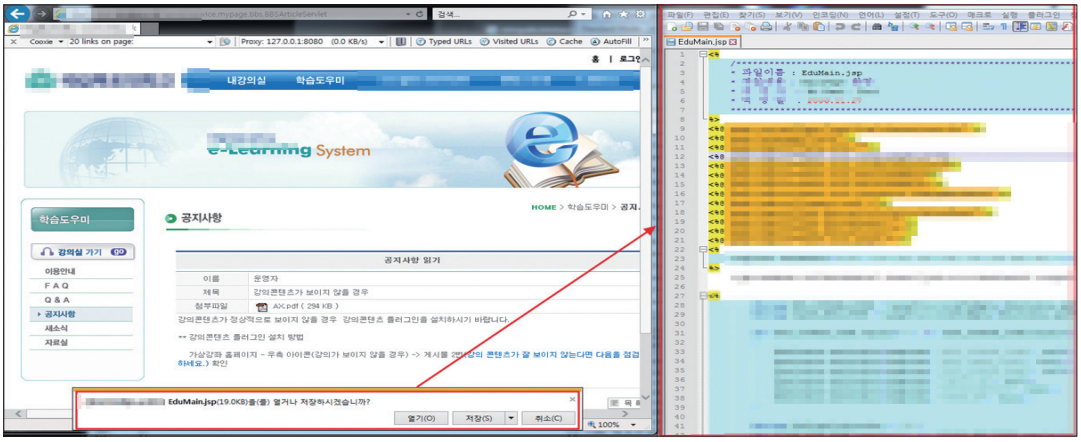
- OO대학 게시판 내 첨부파일 다운로드 시도



- 파일경로와 파일명을 변경하여 요청 시도

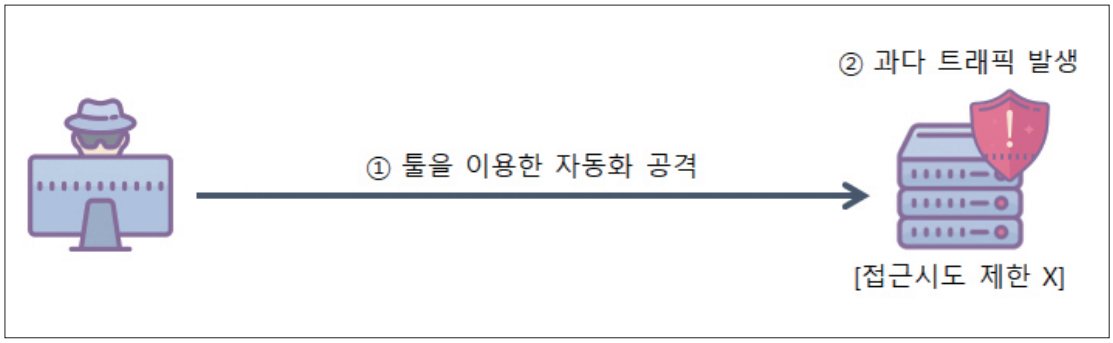


- 서버 내 중요파일(소스코드 파일) 다운로드 가능



8. 자동화 공격

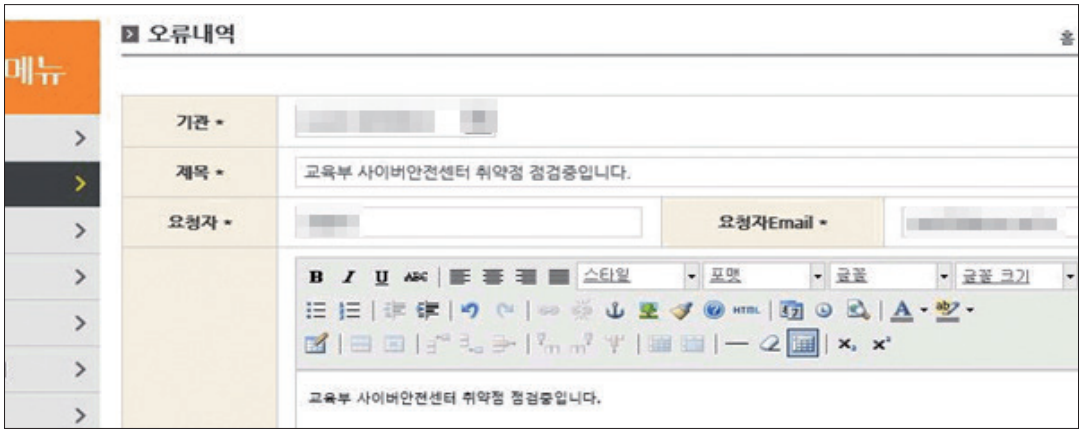
분류	취약점 항목	위험도
취약한 접근 통제	자동화 공격	중



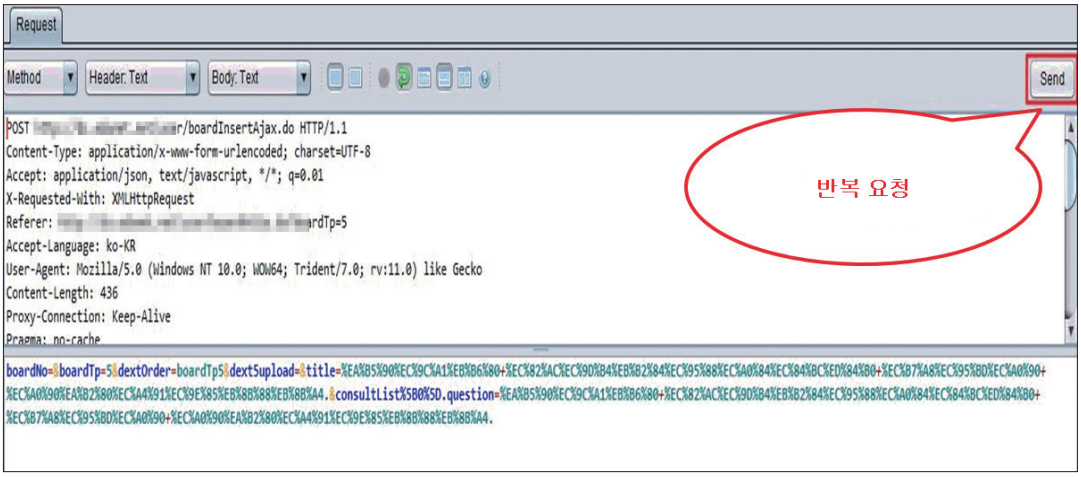
설 명	애플리케이션 운영 시 특정 프로세스에 대한 접근시도 횟수 제한을 설정하지 않을 경우 공격자가 자동화 툴 및 봇을 활용하여 1분에 수백 번의 접근을 시도할 수 있으며 짧은 시간동안 특정 프로세스가 반복 실행되어 시스템 성능에 영향을 미칠 수 있는 취약점
점검목적	웹 애플리케이션에 구현된 기능의 적절성에 대한 검증 로직을 구현하여 자동화 공격 및 무차별 대입 공격을 방지하기 위함
점검내용	1) 자동화된 공격으로 인한 특정 프로세스 반복 실행 가능 여부 점검

▶ 점검 방법

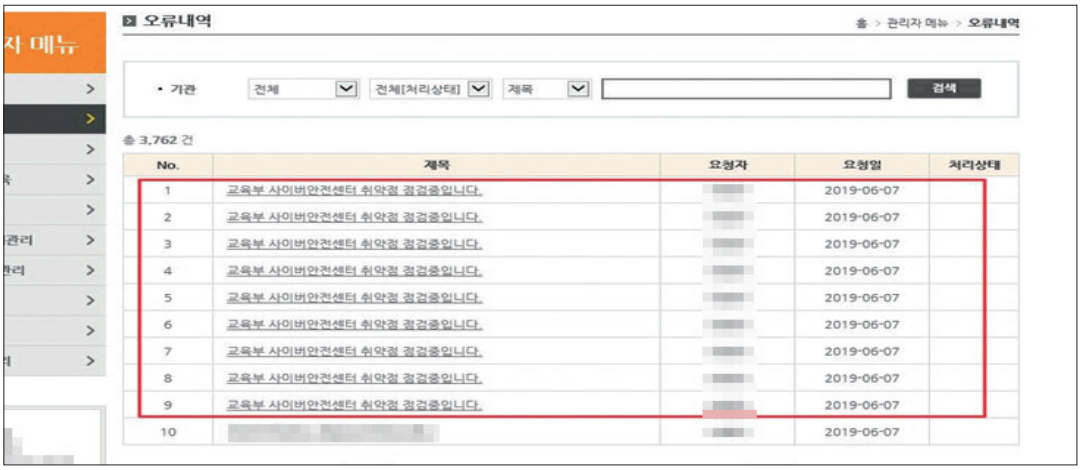
1) 로그인 시도, 데이터 등록 및 메일 발송 등의 프로세스 반복 실행 가능 여부 확인
가) 게시글 작성 시도



나) 프록시 툴의 Repeater(또는 Resend) 기능을 이용하여 글쓰기 트래픽을 반복 전송



다) 글 등록 프로세스 시간, 횟수 제한 없이 다량의 게시물이 작성되는지 확인



▶ 대응 방법

1) 보안 장비에서 조치

가) 시스템 과부하를 방지하기 위해 패킷량을 모니터링 할 수 있는 시스템(IDS/IPS)을 구축하여 다량의 패킷이 유입될 경우 해당 접속을 차단

2) 홈페이지 개발 보안 조치

가) 특정 시간 내 동일 프로세스가 반복 실행되지 않도록 시간제한 설정

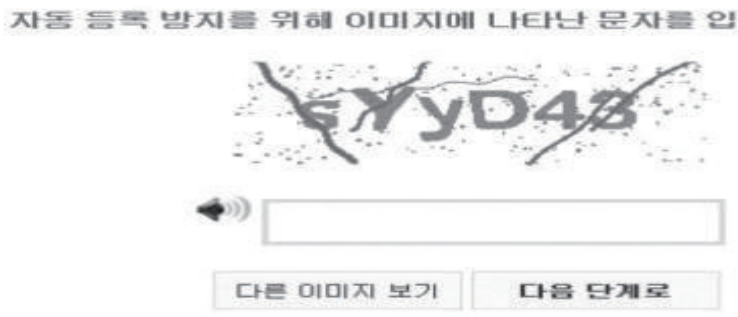
나) 웹 애플리케이션 로그인 관련 테이블에 로그인 시도 횟수를 저장하는 컬럼을 추가하여 로그인 시도가 있을 때마다 횟수를 증가시키고, 일정 횟수 이상이 되면 자동화 공격으로 인식하여 로그인을 할 수 없도록 차단

안전한 코드의 예 JAVA

```
1:  // JAVA PreparedStatement 객체 생성
2: PreparedStatement stmt = connection.prepareStatement("SELECT login_count
FROM users WHERE userid=?);
3:  // setString 메소드를 통한 바인딩 질의 사용
4:  stmt.setString(1, user_id);
5:  ResultSet rs = stmt.executeQuery();
6:  // 로그인 시도 횟수가 5회 이상인지 확인
7:  if ( rs.getInt(1) > 5 ) {
8:      // 로그인이 불가능 하도록 설정(계정 잠금)
9:  }
10:  else { ... }     // 정상적인 로그인 절차
```

다) 게시물 등록, 메일 발송 등의 기능에서 사용자의 요청이 일회성이 될 수 있도록 캡챠(이미지를 이용하여 확인 값을 표시하고 사용자가 값을 입력하여 인증) 등을 이용

※ 캡챠(CAPTCHA): 자동화된 컴퓨터와 사람을 판별하기 위한 기술의 일종

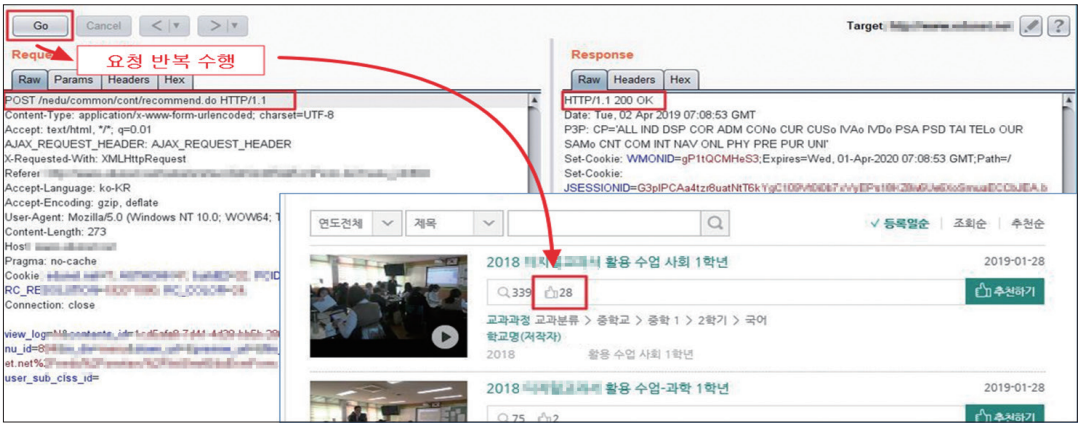


안전한 코드의 예 JAVA

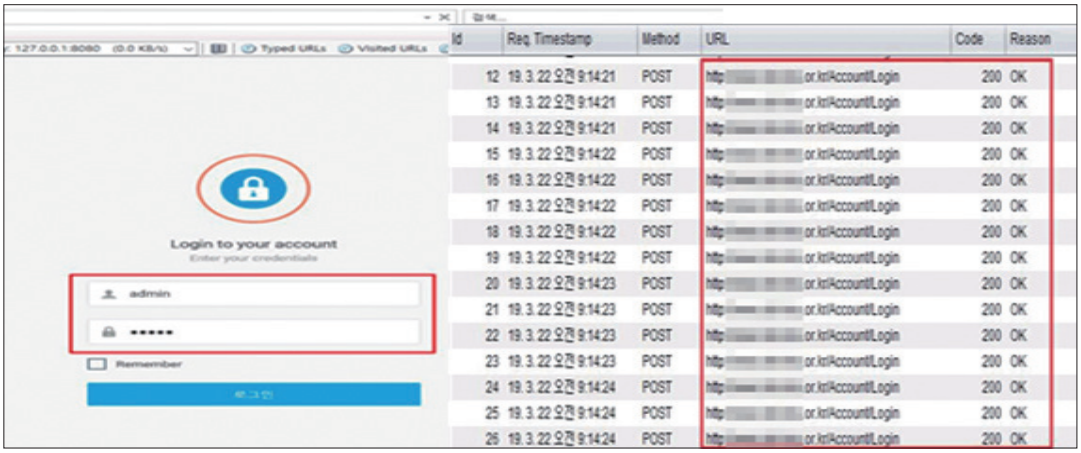
```
1: // 캡챠 사용을 위한 라이브러리
2: <%@ page import = "nl.captcha.Captcha" %>
3: <%
4: // 캡챠 생성
5: Captcha captcha = (Captcha)session.getAttribute(Captcha.NAME);
6: request.setCharacterEncoding("UTF-8");
7: String answer = request.getParameter("answer");
8: // 캡챠 내용 검증
9: if ( captcha.isCorrect(answer) ) {
10:     // 정상적인 프로세스 처리
11: }
12: else { ... } // 에러 메시지 출력
13: %>
```

▶ 사례

- OO기관은 익명 사용자가 동일한 게시물을 시간, 횟수 제한 없이 추천하기 가능



- OO대학은 로그인 요청 프로세스 시간, 횟수 제약 없이 반복 실행 가능



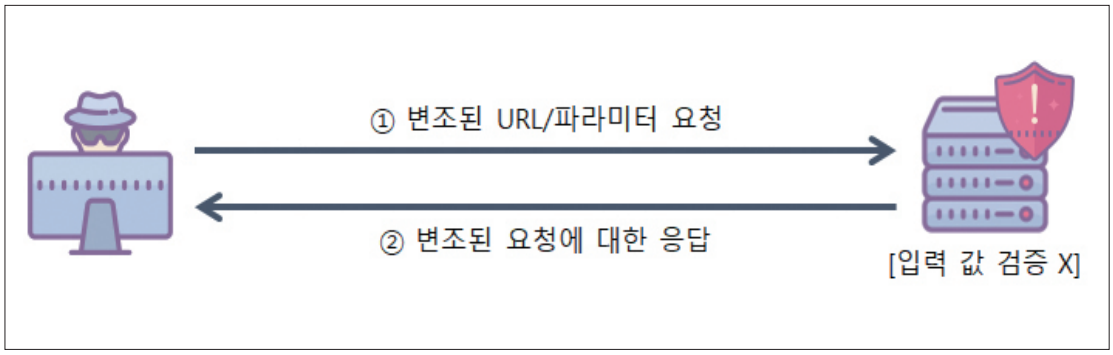
Ⅳ 취약한 인증

개요
인증 및 세션 관리와 관련된 애플리케이션 기능의 구현 상 결함에 의해 발생되는 보안 약점으로 암호, 키, 세션 토큰 등이 노출되어 공격자에 의해 악용될 경우 일시적 또는 영구적으로 다른 사용자의 권한을 탈취하는 것이 가능함

취약점 항목			
9	URL/파라미터 변조	10	불충분한 세션 관리
11	쿠키 변조	12	디폴트/취약한 계정사용

9. URL/파라미터 변조

분류	취약점 항목	위험도
취약한 인증	URL/파라미터 변조	상



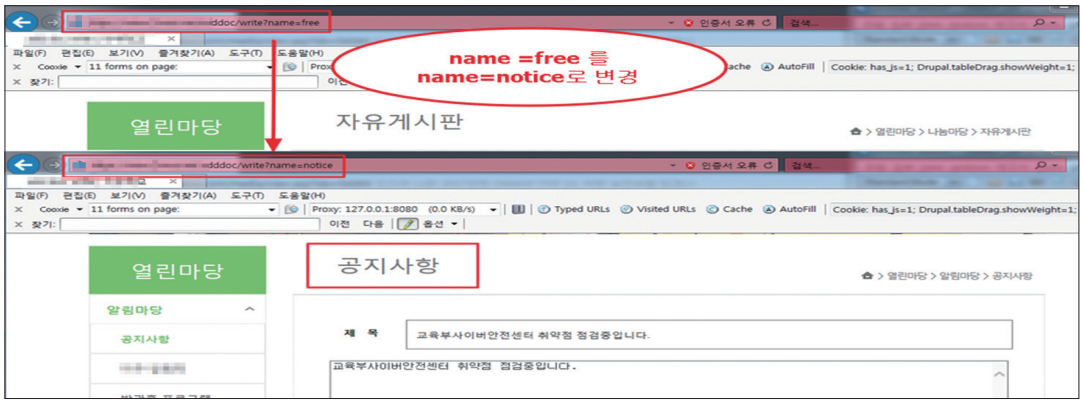
설 명	웹 서버에 전송되는 모든 HTTP 요청 값(URL 파라미터, Form 변수 등)을 조작하여 접근 권한이 없는 정보를 조회, 변경하고 인가 받지 않은 기밀정보를 유출하거나 악용 할 수 있는 취약점
점검목적	파라미터를 변조하여 인가 받지 않은 기밀정보를 유출하거나 악용하는 것을 방지하기 위함
점검내용	1) 파라미터 변조를 통한 권한 상승이 가능한지 여부 확인

▶ 점검 방법

1) URL의 파라미터 값을 조작하여 권한이 없는 게시판에 글쓰기(수정, 삭제 등) 가능 확인

점검 예)

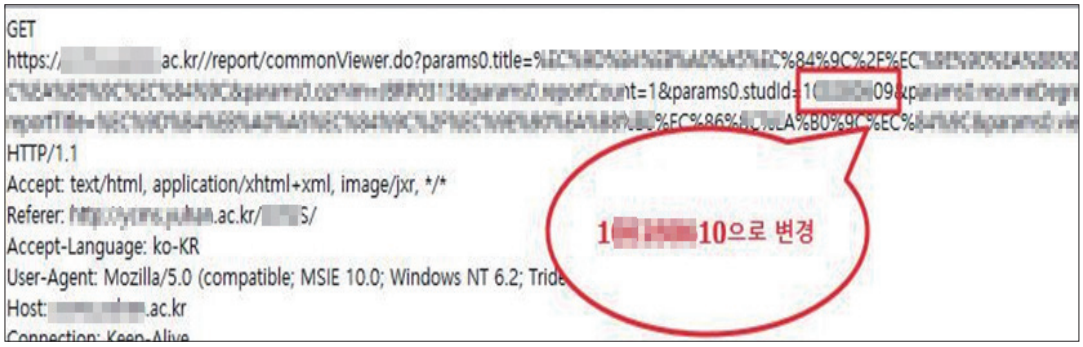
글 쓰기 : board.php?mode=list에서 mode값을 write, add, insert 등으로 변경
글 수정 : board.php?mode=view&idx=1에서 mode값을 update, modify 등으로 변경
글 삭제 : board.php?mode=delete&idx=1에서 idx값을 다른 게시물 번호로 변경

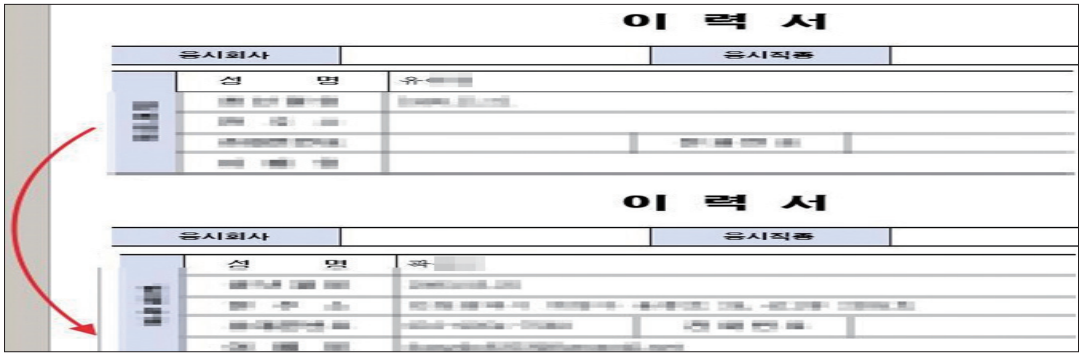


2) URL의 파라미터 값을 조작하여 권한이 없는 게시물(비밀글, 개인정보 등) 열람 가능 확인

가) 본인 권한으로 한정된 기능 사용 시도

나) 파라미터에 포함된 본인 인증 값(학번, ID등)을 변조하여 전송 후 결과 확인





▶ 대응 방법

1) 홈페이지 개발 보안 조치

가) 홈페이지 중 중요한 정보가 있는 페이지(계좌이체 등)는 재 인증 적용

나) 안전하다고 확인된 라이브러리나 프레임워크 (OpenSSL이나 ESAPI의 보안 기능 등)를 사용

다) 상태정보나 민감한 데이터 특히, 사용자 세션정보와 같은 중요한 정보는 서버에 저장하고 보안확인 절차도 서버에서 실행

라) 인증이 필요한 모든 페이지에 대해 유효 세션임을 확인하는 프로세스 및 주요 정보 페이지에 접근 요청자의 권한 검증 로직을 적용

마) 사용자의 권한에 따른 ACL(Access Control List) 관리

바) 응용프로그램이 제공하는 정보와 기능을 역할에 따라 배분함으로써 공격자에게 노출되는 공격노출면(attack surface) 최소화

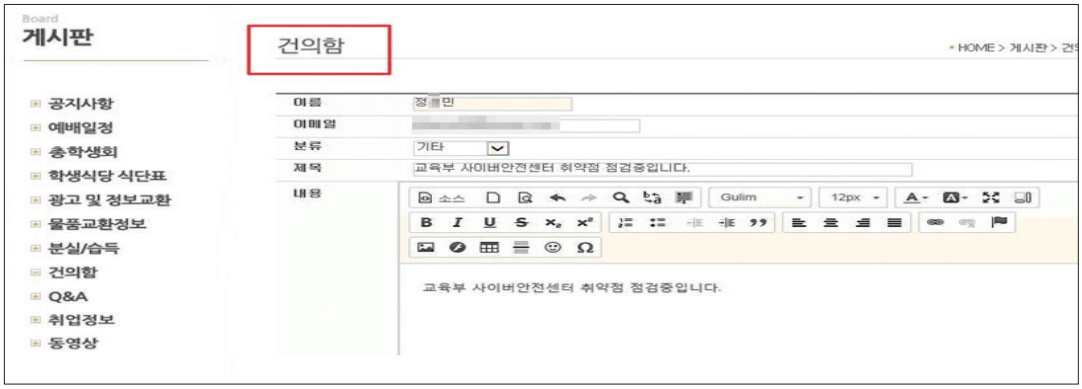
사) 사용자 권한, 인증 여부 등 보안결정에 사용하는 값은 사용자 입력 값을 사용하지 않고 서버 내부의 값을 활용하며, 사용자 입력에 의존해야하는 값을 제외하고는 반드시 서버가 보유하고 있는 정보를 이용하여 처리하도록 코딩

안전한 코드의 예 JAVA

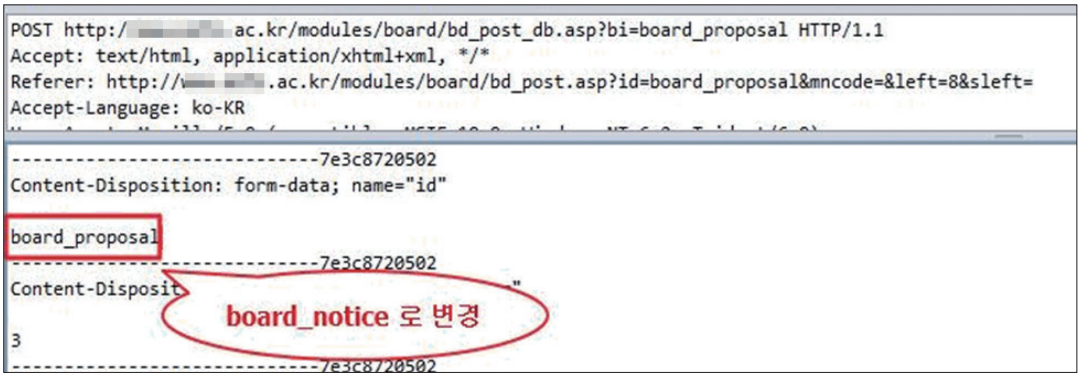
```
// 사용자의 세션 정보를 반환
HttpSession session = request.getSession(true);
// 세션에 존재하는 userID 정보를 반환
String sessionID = session.getAttribute("userID");
// 서버로 전달된 파라미터를 변수에 저장
String userID = request.getParameter("userID");
// 사용자 입력 값과 세션 값을 비교
if ( sessionID.equals(userID) ) {
  // 정상적인 게시글 등록
}
else { ... } // 에러 메시지 출력 (파라미터 변조 탐지)
```

▶ 사례

- OO대학의 글 작성 권한이 있는 게시판에 글 작성 시도



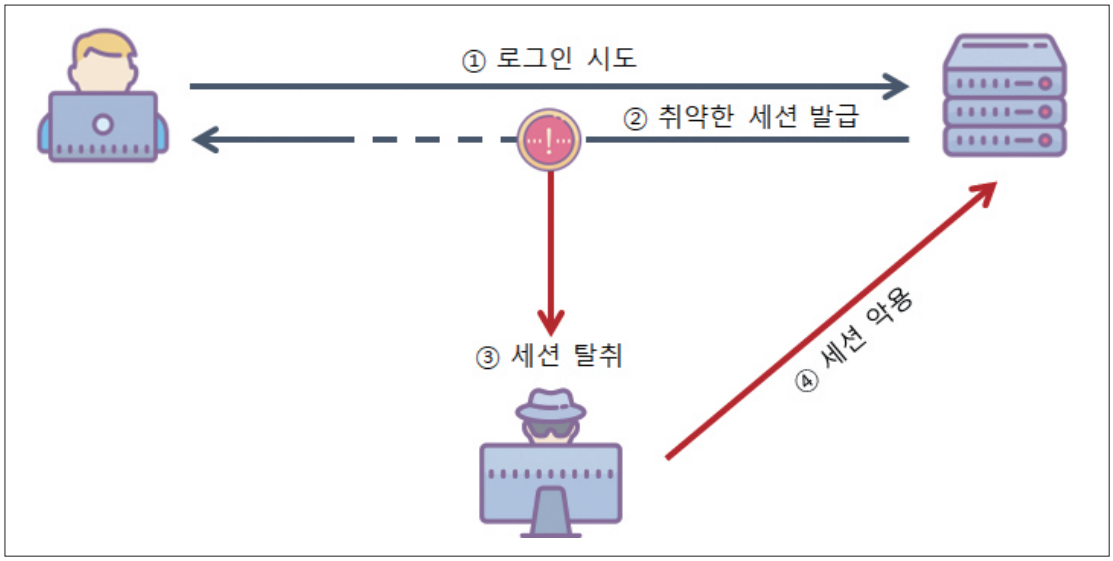
- 게시판 id 값을 공지사항 게시판 id 값으로 변경 후 요청



- 글 작성 권한이 없는 게시판에 게시물 업로드 가능

10. 불충분한 세션 관리

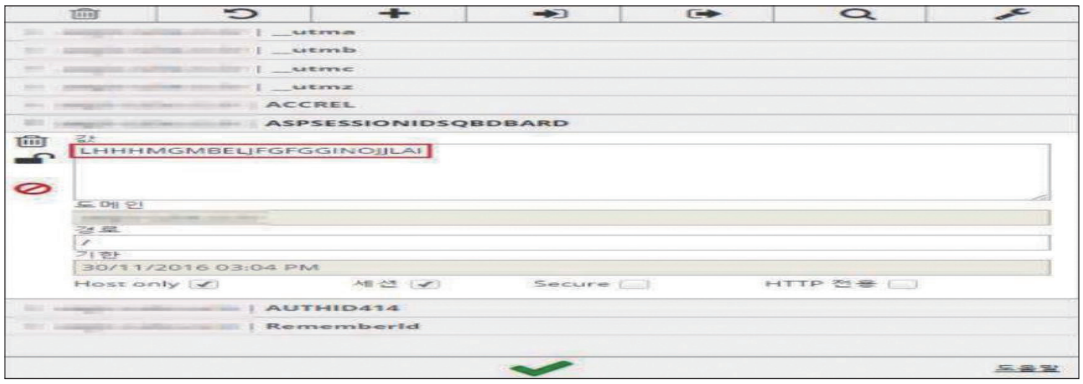
분류	취약점 항목	위험도
취약한 인증	불충분한 세션 관리	상



설 명	웹 애플리케이션에서 사용자가 로그인할 경우 매번 같은 세션ID를 발급하거나, 세션 타임아웃을 너무 길게 설정하였을 경우 공격자가 만료되지 않은 세션을 재사용하여 해당 사용자의 권한을 탈취할 수 있는 취약점 *세션(Session) : 일정 시간동안 같은 사용자(브라우저)로부터 들어오는 일련의 요구를 하나의 상태로 보고 그 상태를 일정하게 유지 시키는 기술
점검목적	로그인 할 때 마다 예측 불가능한 새로운 세션 ID를 발행하고 세션 타임아웃을 설정하여 공격자가 만료되지 않은 세션을 악용하는 것을 방지하기 위함
점검내용	1) 사용자 로그인 시 일정하게 고정된 세션 ID값을 발행하는지 여부 점검 2) 세션의 만료 시간 설정 여부 점검 3) 로그아웃 후 세션 폐기를 정상적으로 수행하는지 여부 점검

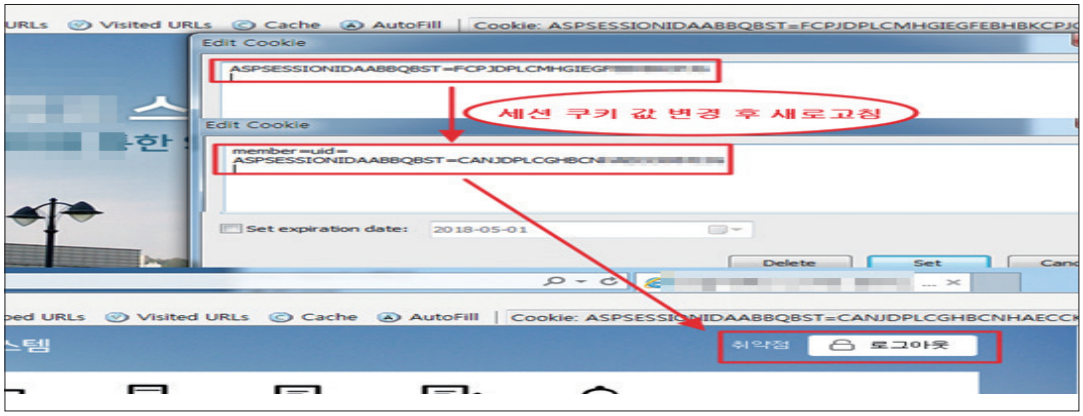
▶ 점검 방법

1) 로그인을 여러 번 시도하여 동일한 세션 ID(혹은 추측 가능한 세션) 값을 발행하는지 여부 확인



2) 인증을 완료 한 후 로그아웃하지 않고 일정시간이 경과했을 때, 세션 유지 여부 확인

3) 로그아웃 후 로그아웃 전에 수집해 둔 세션을 통한 재 로그인이 가능한지 확인.



▶ 대응 방법

1) 웹 서버 내에서의 조치

가) 세션 연결 후 일정 시간동안 반응이 없을 경우 세션이 끊어지도록 설정

① Tomcat 조치 방법

web.xml 파일에서 <session-config> 태그를 사용하여 타임아웃을 지정
web.xml, Weblogic.xml 중 한 곳에만 설정
(두 곳 모두 설정 시 우선순위에 의해 web.xml의 설정이 적용됨)

- Web.xml : "분"단위

```
<session-config>
   <session-timeout>10</session-timeout>
</session-config>
```

- Weblogic.xml: "초"단위

```
<session-descriptor>
   <timeout-secs>600</timeout-secs>
</session-descriptor>
또는,
<session-param>
   <param-name>TimeoutSecs</param-name>
   <param-value>600</param-value>
</session-param>
```

2) 홈페이지 개발 보안 조치

가) 로그인 할 때마다 예측 불가능한 새로운 세션 ID를 발급받도록 설계하고 기존 세션 ID는 파기

나) 세션 종료 시간 설정 또는 자동 로그아웃 기능 구현(세션 종료 시간은 사이트 특성에 따라 달라질 수 있으므로 사이트의 특성에 맞게 시간 설정)

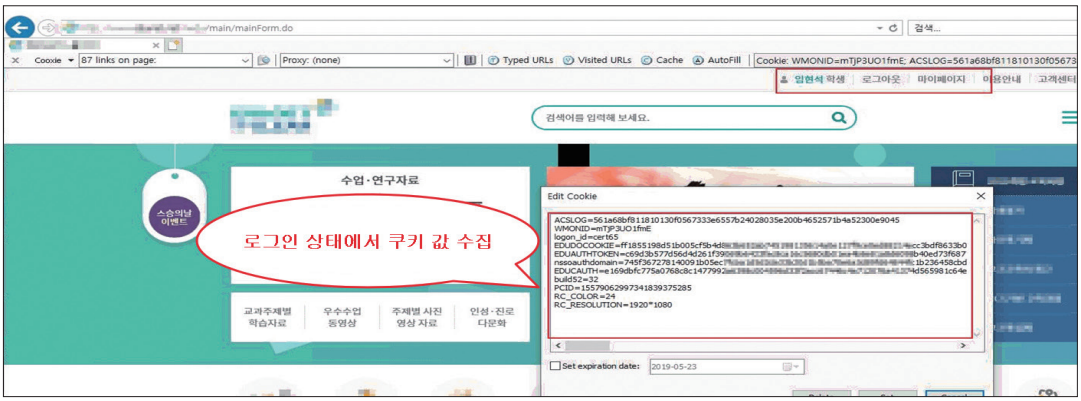
다) 단순 조합 보다는 상용 웹 서버나 웹 애플리케이션 플랫폼에서 제공하는 세션 ID를 사용하고, 가능하다면 맞춤형 세션 관리 체계를 권고

안전한 코드의 예 JAVA

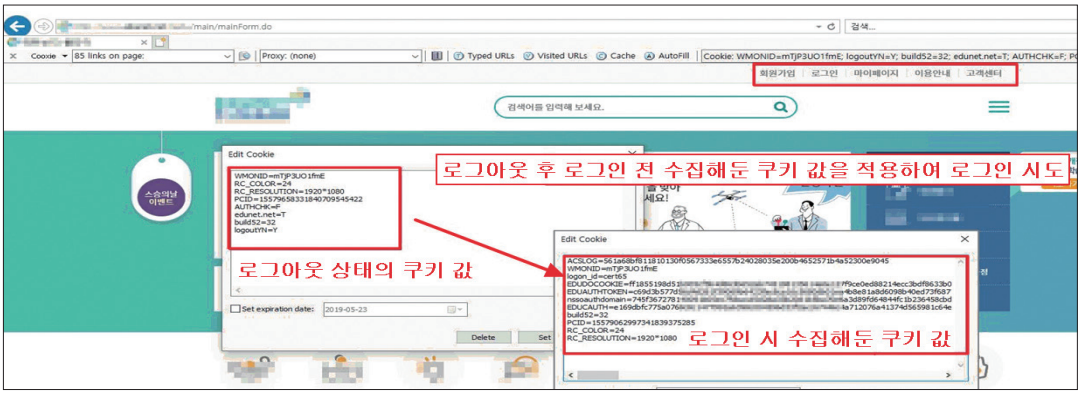
```
1: // Session의 유지 시간 Setting
2: String strTime = Param.getPropertyFromXML("SessionPersistenceTime");
3: if (strTime == null) {
4: session.setMaxInactiveInterval(600);
5: } else {
6: session.setMaxInactiveInterval((new Integer(strTime)).intValue()); // 초 단위
7: }
```

▶ 사례

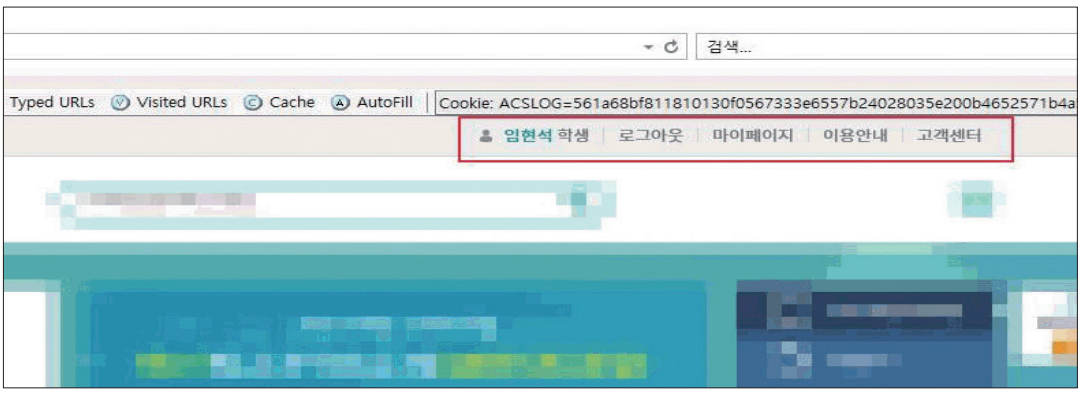
- 로그인 상태에서 세션 값이 포함된 쿠키 값 복사



- 로그아웃 후 로그인 전에 수집해둔 쿠키 값을 적용

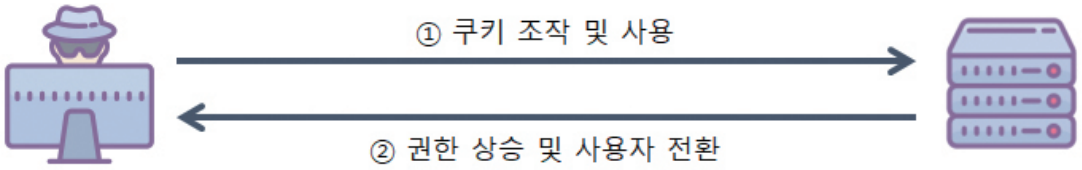


- 쿠키 값이 적용된 상태에서 새로 고침 시 정상 로그인되는 것을 확인됨



11. 쿠키 변조

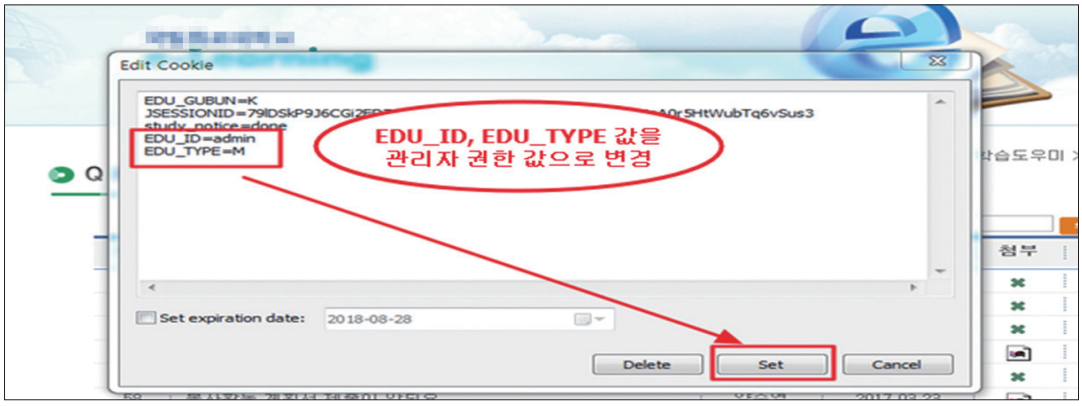
분류	취약점 항목	위험도
취약한 인증	쿠키 변조	상



설 명	사용자 인증 방식중 하나인 쿠키를 변조하여 다른 사용자로 전환하거나 권한 상승이 가능한 취약점 *쿠키(cookie) : 인터넷 사용자가 웹사이트를 방문할 경우 그 사이트가 사용하고 있는 서버에서 인터넷 사용자의 컴퓨터에 설치하는 기록 정보 파일
점검목적	쿠키를 사용하는 경우 안전한 알고리즘으로 암호화하여 쿠키 값 변조를 통한 다른 사용자로의 위장 및 권한 변경을 방지하기 위함
점검내용	1) 쿠키 값(인증을 위한 ID, 권한을 위한 구분자 등) 변조를 통한 권한 상승 가능 여부 점검

▶ 점검 방법

1) 쿠키 값(인증을 위한 ID, 권한을 위한 구분자 등) 변조를 통한 권한 상승 가능 여부 점검

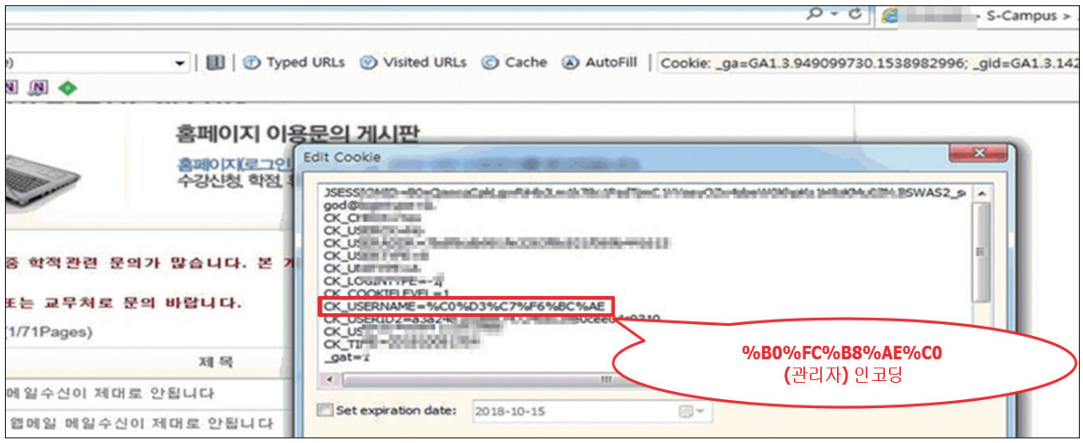


▶ 대응 방법

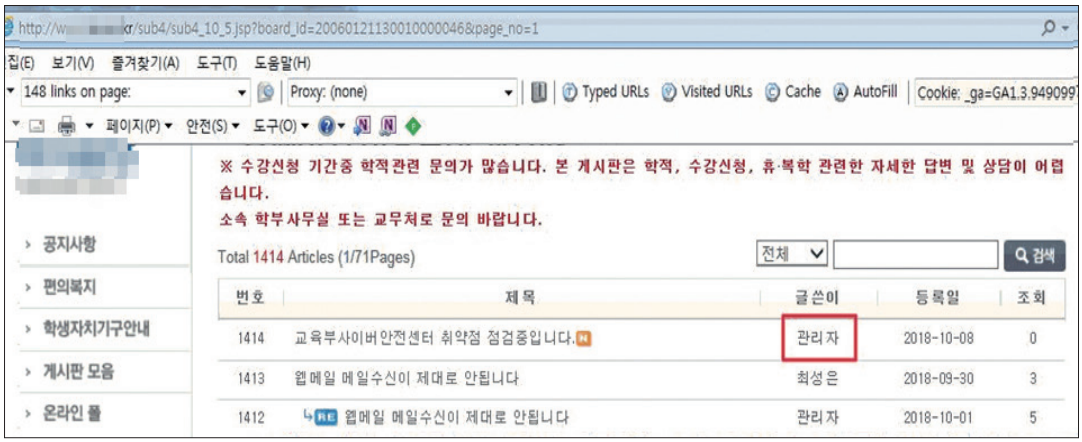
1) 홈페이지 개발 보안 조치
가) 홈페이지는 사용자 인증 등 중요기능 구현 시 가급적이면 Cookie 대신 Session 방식 사용
나) 홈페이지의 사용자 인증 등 중요기능 구현 시 Cookie(또는 Session) 방식 활용 시 안전한 알고리즘(SEED, AES 등)을 사용
다) 세션정보와 같은 중요한 정보는 서버에 저장하고 보안확인 절차도 서버에서 실행

▶ 사례

- OO대학의 쿠키 정보에서 사용자 이름으로 보여지는 값을 변조 후 적용

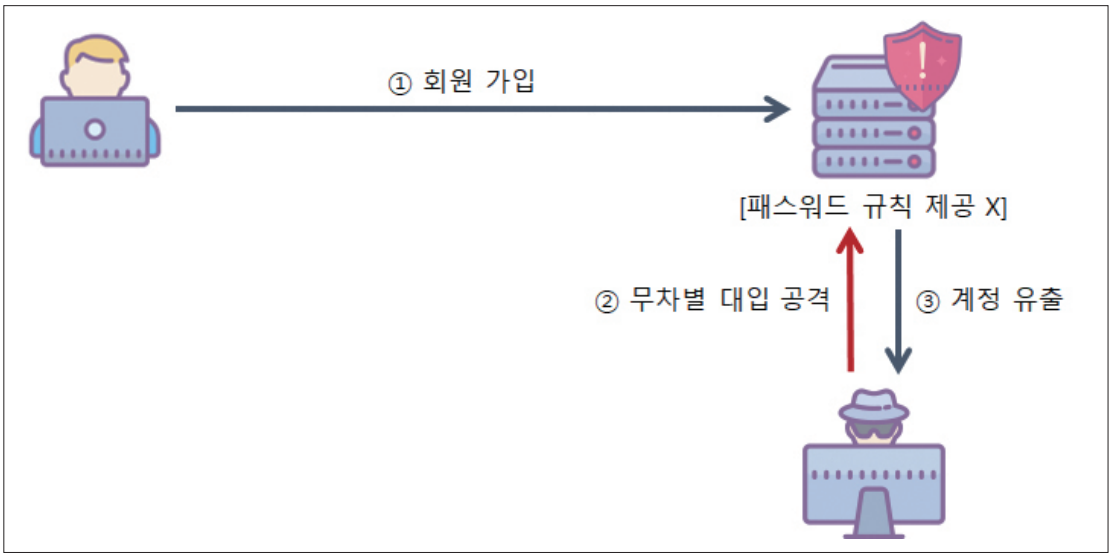


- 글 작성 시 쿠키 값에서 변경한 사용자 이름으로 글 등록 되는 것을 확인



12. 디폴트/취약한 계정사용

분류	취약점 항목	위험도
취약한 인증	디폴트/취약한 계정사용	상



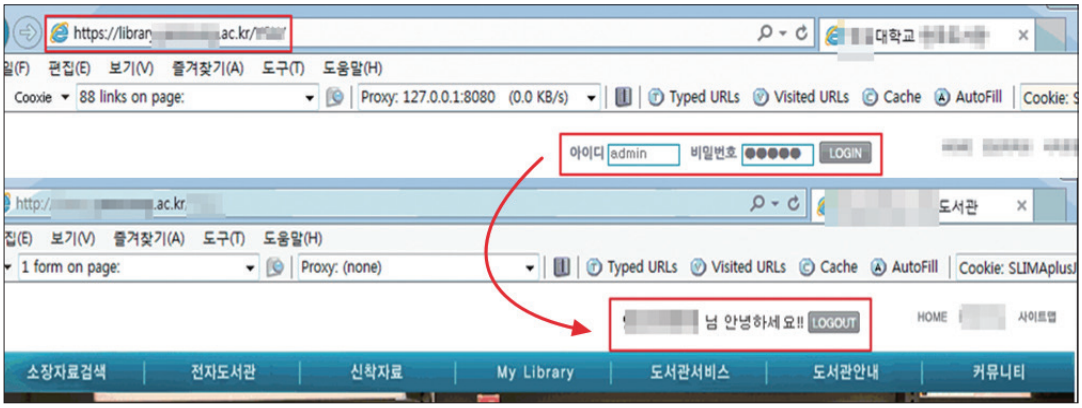
설 명	회원가입 시 취약한 패스워드로 회원 가입이 가능한 경우나 개발용으로 사용된 테스트 계정, 웹 애플리케이션 서버의 default 계정 등이 무차별 대입 공격을 통해 패스워드가 누출 될 수 있는 취약점
점검목적	default 계정사용 및 유추 가능한 취약한 계정 사용을 제한하여 계정 및 패스워드 추측공격을 방지하기 위함
점검내용	1) 테스트 계정, default 계정 등 취약한 계정 사용 여부 확인

▶ 점검 방법

1) 로그인 시 취약한 계정(추측 가능한 계정)으로 로그인 가능 여부 확인

ID/PW 점검 예)

master, webmaster, admin, administrator, root, manager, test, masterweb 등



2) 웹 애플리케이션 서버에서 default 계정 사용 여부 확인

▶ 대응 방법

1) 홈페이지 개발 보안 조치

가) 사용자가 취약한 패스워드를 사용할 수 없도록 패스워드 생성규칙을 강제 할 수 있는 로직을 적용

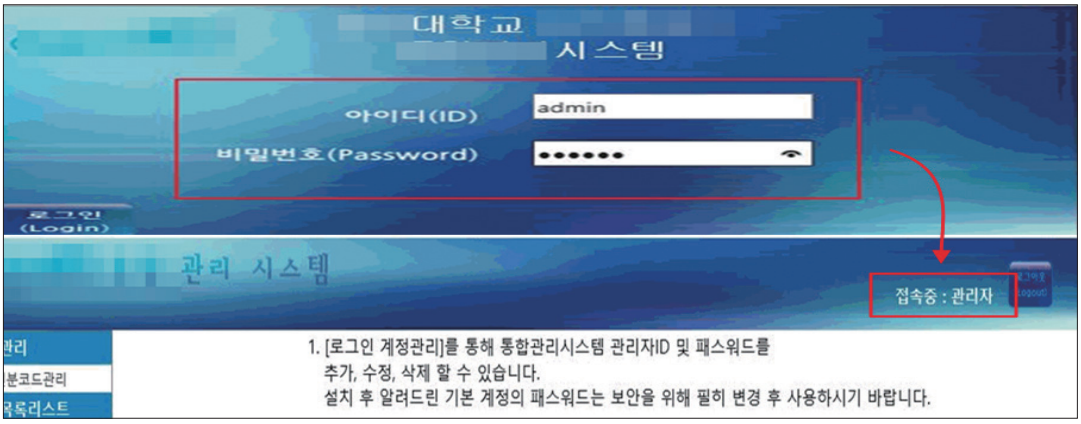
분류	내용
패스워드 생성규칙	- 세가지 종류 이상의 문자구성으로 8자리 이상의 길이 - 두가지 종류 이상의 문자구성으로 10자리 이상의 길이
패스워드 생성 금지규칙	- 간단한 문자(영어단어 포함)나 숫자의 연속사용은 금지 - 키보드 상에서 일련화 된 배열을 따르는 패스워드 선택 금지 - 사전에 있는 단어, 이를 거꾸로 철자화한 단어 사용 금지 - 생일, 전화번호, 개인정보 및 아이디와 비슷한 추측하기 쉬운 비밀번호 사용 금지 - 이전에 사용한 패스워드는 재사용 금지 - 계정 잠금 정책 설정 ex)로그인 5회 실패 시 30분동안 사용중지

나) 개발용으로 사용된 계정은 삭제 하거나 안전한 패스워드로 생성

다) 디폴트 계정 존재 시 패스워드 정보 변경

▶ 사례

- OO 대학의 솔루션 관리자 페이지에서 디폴트 계정 정보로 정상 로그인 가능

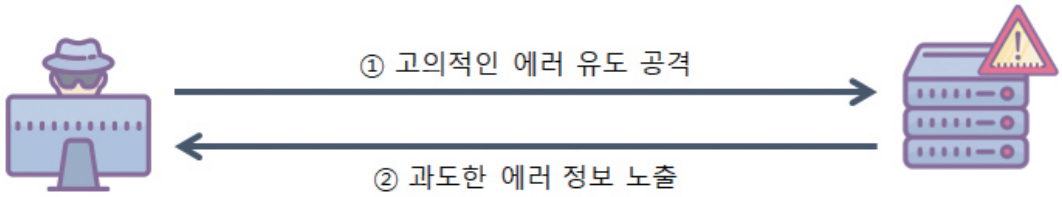


V 민감한 데이터 노출

개요			
민감한 정보가 제대로 보호되지 않을 경우 에러메시지, 소스코드 혹은 데이터 송수신 과정에서 개인식별정보, 계정정보 등이 노출 될 수 있으며, 이는 공격자의 악성 행위를 위한 정보로 활용 될 수 있음			
취약점 항목			
13	부적절한 에러메시지 노출	14	소스코드 내 중요 정보 노출
15	중요 정보 비 암호화 통신		

13. 부적절한 에러메시지 노출

분류	취약점 항목	위험도
민감한 데이터 노출	부적절한 에러메시지 노출	중



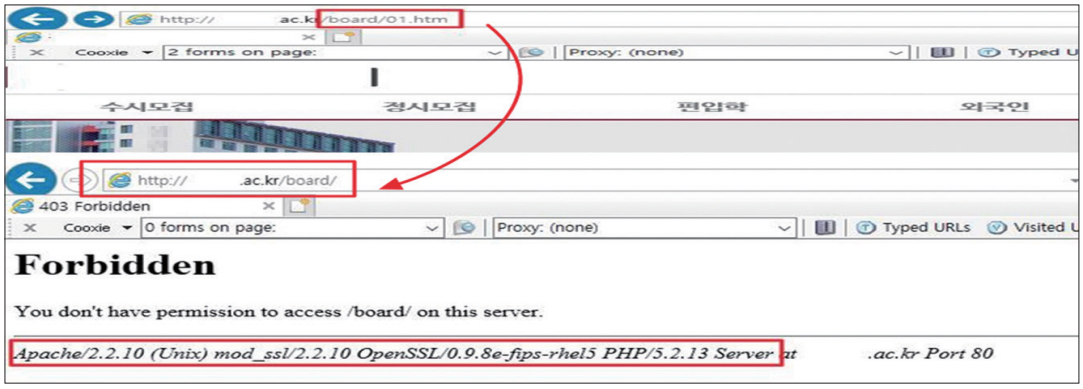
설 명	사용자의 실수 또는 고의적인 입력 데이터에 대해 애플리케이션은 시스템 에러를 보이거나 특정 페이지로 이동하는 등 여러 형태의 반응을 보이게 되는데 이 때 서버 데이터 정보 등 공격자에게 잠재적인 취약점을 알려 주는 요인이 노출되는 취약점
점검목적	에러 상황에서 적절한 에러 메시지가 노출되도록 하여 2차 공격에 활용될 수 있는 불필요한 정보 노출을 차단하기 위함
점검내용	1) URL에 웹 서버 디렉터리명을 입력하여 에러페이지 확인 2) 로그인 페이지에서 로그인 실패 메시지 확인

▶ 점검 방법

1) URL에 웹 서버 디렉터리명을 입력하여 에러페이지 확인

점검 예)

실제 경로) http://www.점검사이트.ac.kr/board/01.htm
요청 경로) http://www.점검사이트.ac.kr/board/



2) 로그인 페이지에서 로그인 실패 메시지 확인

점검 예)

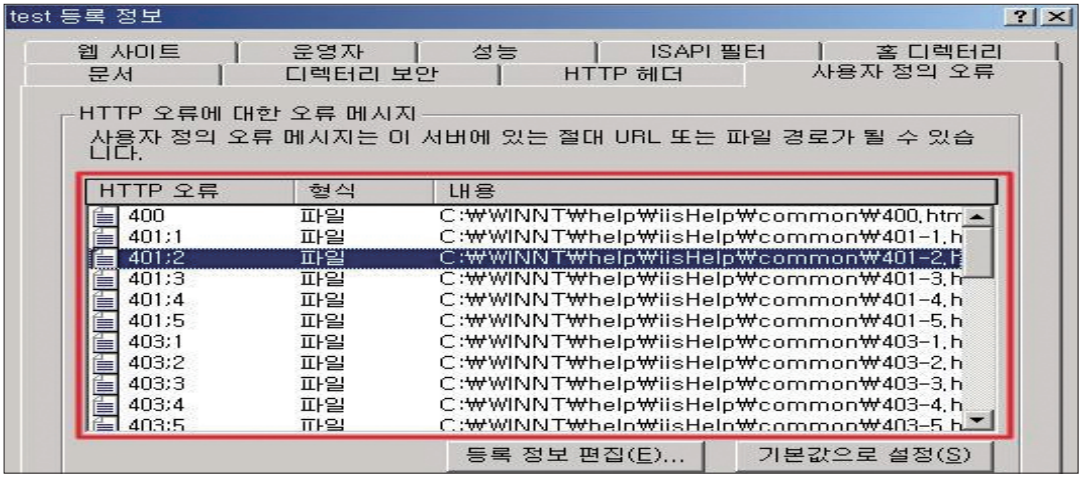
존재하지 않는 계정의 패워스드를 틀렸을 경우, "아이디가 존재하지 않습니다."
존재하는 게정의 패스워드를 틀렸을 경우, "패스워드가 일치하지 않습니다."

▶ 대응 방법

1) 웹 서버 내에서의 조치
가) 에러 발생 시 에러페이지를 별도 제작하여 리다이렉트 하거나 메인페이지로 리다이렉트 되도록 설정

① IIS 웹 서버 조치 방법

[제어판] → [관리도구] → [인터넷 서비스 관리자] → 웹 사이트 선택 후 마우스 오른쪽 버튼 클릭 → [등록 정보] → [사용자 정의 오류] → 상태 코드별 응답 설정



② Apache 웹 서버 조치 방법

아파치 웹 서버의 설정 파일인 httpd.conf 파일에서 "ErrorDocument"를 찾아 수정

```
ErrorDocument 503 "제작한 에러페이지 경로"
ErrorDocument 500 "제작한 에러페이지 경로"
ErrorDocument 404 "제작한 에러페이지 경로"
ErrorDocument 403 "제작한 에러페이지 경로"
```

③ Tomcat 조치 방법

web.xml 파일에서 <error-page> 태그를 사용하여 에러페이지 설정

```
<error-page>
<error-code>404</error-code>
<location>제작한 에러페이지 경로</location>
</error-page>
<error-page>
<error-code>500</error-code>
<location>제작한 에러페이지 경로</location>
</error-page>
<error-page>
<exception-type>java.lang.Throwable</exception-type>
<location>제작한 에러페이지 경로</location>
</error-page>
```

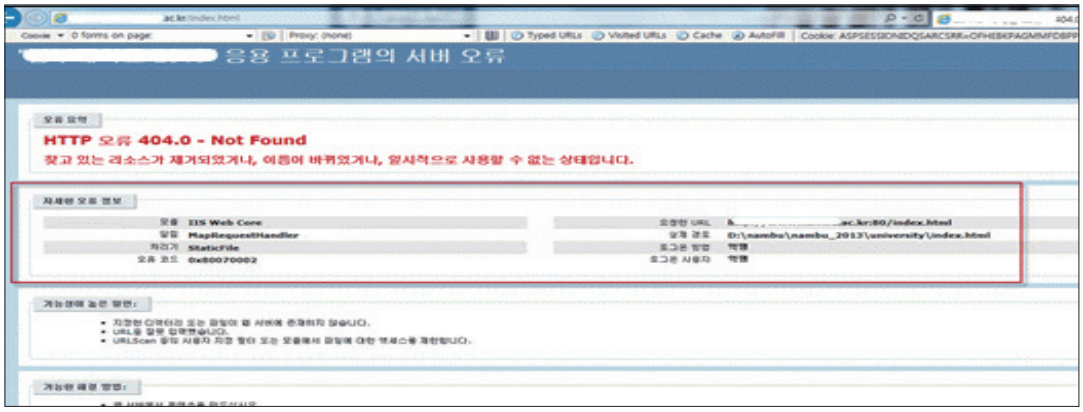
2) 홈페이지 개발 보안 조치가) 로그인 시 아이디나 패스워드가 틀린 경우 통일된 메시지 출력

메시지 출력 예시)

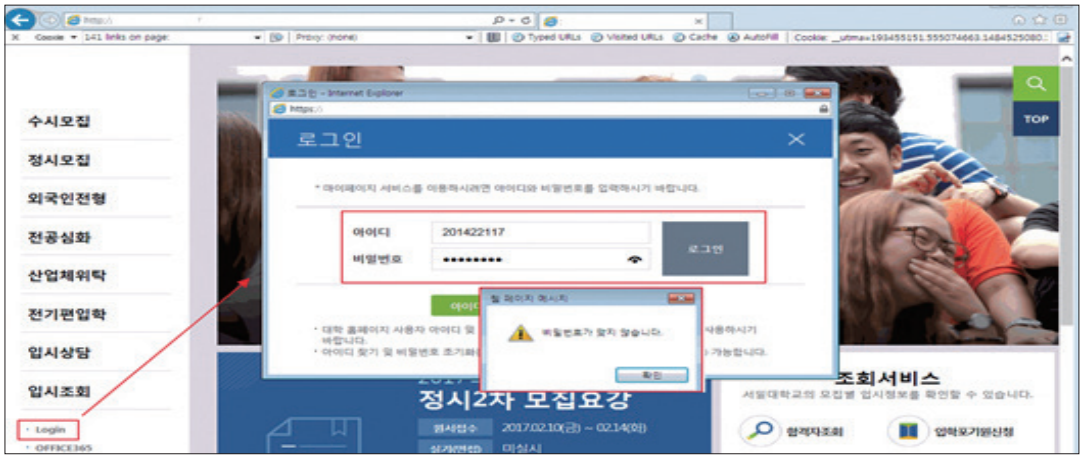
"아이디가 틀렸습니다.", "패스워드가 틀렸습니다." (X)
"로그인에 실패했습니다.", "아이디 혹은 비밀번호가 틀렸습니다." (O)

▶ 사례

- OO대학은 특정 경로 요청 시 서버 정보 및 경로 정보가 노출됨

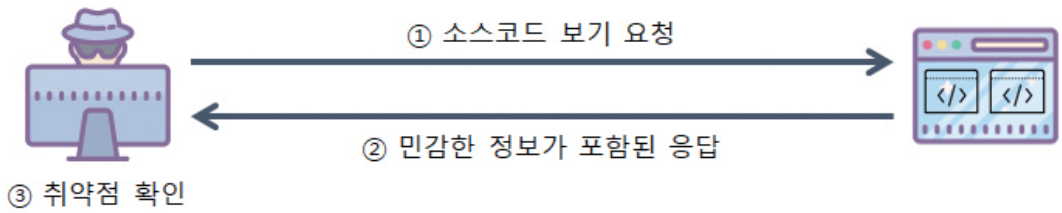


- OO대학은 로그인 시 계정 존재 유무를 추측할 수 있는 실패 메시지 출력 확인



14. 소스코드 내 중요 정보 노출

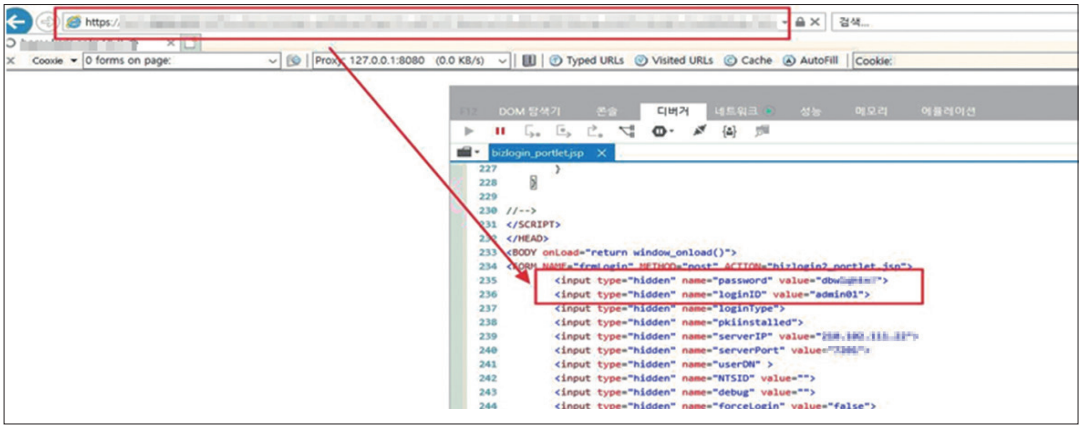
분류	취약점 항목	위험도
민감한 데이터 노출	소스코드 내 중요 정보 노출	상



설 명	소스코드에 민감한 정보(개인 정보, 시스템 정보 등)이 포함되어 있는 경우, 외부 공격자에 의해 패스워드 등 보안 관련정보가 노출될 수 있는 취약점
점검목적	소스코드를 통한 민감한 정보 유출로 인한 추가 공격을 하지 못하도록 방지하기 위함
점검내용	1) 소스코드 내에 중요 정보(개인 정보, 인증정보, DB 접속 정보 등) 노출 여부 확인

▶ 점검 방법

1) 웹 페이지에서 소스보기를 통해 민감한 정보 존재여부 확인



▶ 대응 방법

1) 홈페이지 개발 보안 조치

가) 홈페이지 소스코드에는 디버깅 목적으로 주로 ID, 패스워드, 시스템 관련 정보 등 보안관련 정보가 남지 않도록 개발완료 후 제거 필요

나) 프로그램 코드 내부에 패스워드는 암호화하여 별도의 파일에 저장하여 사용하고, SW 설치 시 사용하는 디폴트 패스워드, 키 등을 사용하는 대신 “최초-로그인” 모드를 두어 사용자가 직접 패스워드나 키를 입력하도록 설계

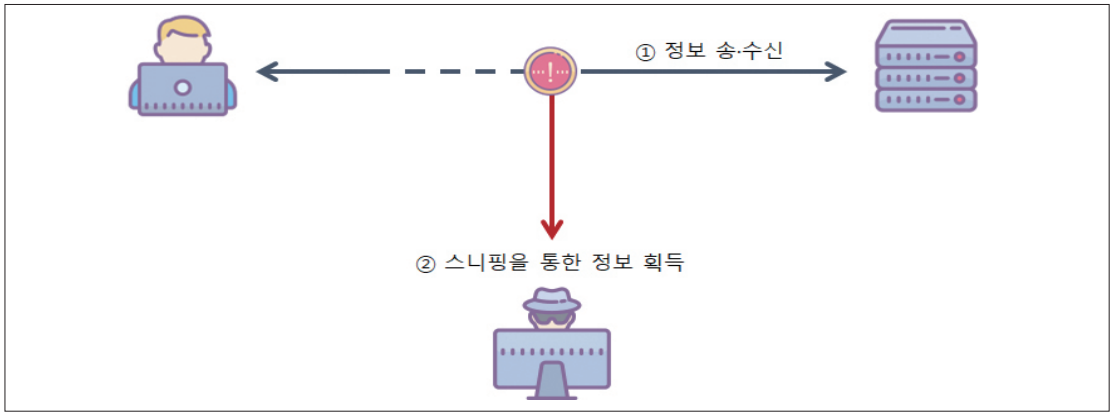
▶ 사례

- OO대학의 소스코드 파일에서 악용할 수 있는 관리자 권한 함수 다수 노출 확인

```
view.jsp
654   }
655   function moves( board , pk){
656     url = "cate.jsp?board="+board + "&idx="+pk;
657     w = 380
658     h = 400
659     l = ( screen.width - w ) /2
660     t = ( screen.height - h) /2
661     str = "width="+w+", height="+h+", top="+t+",left="+l + ", resizable=1, scrollbars=1"
662     window.open( url , "CATE" , str )
663   }
664   function RealMoves( board , pk){
665     url ="move_ok.jsp?menu_board=&board=&old_board=[board]&page=&idx="+pk
666     location.href= url
667   }
668   function movesfirst( board , pk){
669       url = "prime.jsp?board="+board + "&idx="+pk;
670     w = 380
671     h = 400
672     l = ( screen.width - w ) /2
673     t = ( screen.height - h) /2
674     str = "width="+w+", height="+h+", top="+t+",left="+l + ", resizable=1, scrollbars=1"
675     window.open( url , "PRIME" , str )
676   }
677   function imsi_del( board , pk){
678     if ( confirm("삭제하시겠습니까?")){
679         url = "imsi_del_ok.jsp?board="+board + "&idx="+pk;
680       w = 380
681       h = 300
682       l = ( screen.width - w ) /2
```

15. 중요 정보 비 암호화 통신

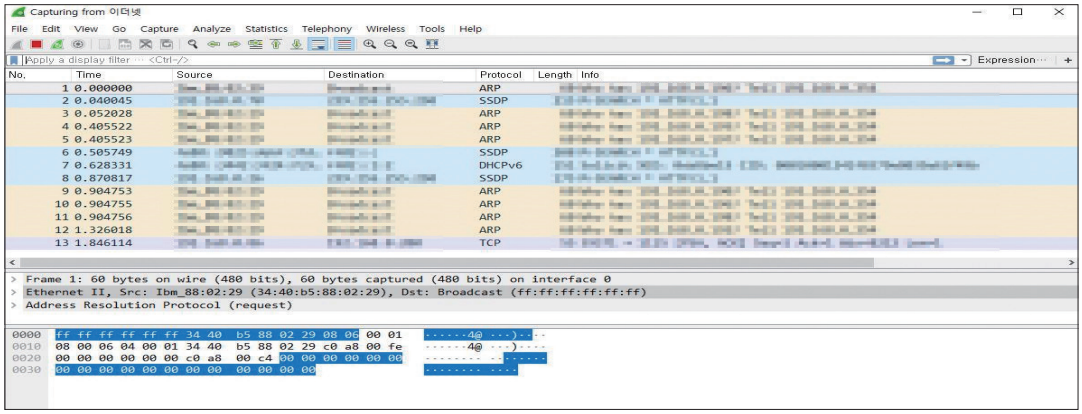
분류	취약점 항목	위험도
민감한 데이터 노출	중요 정보 비 암호화 통신	상



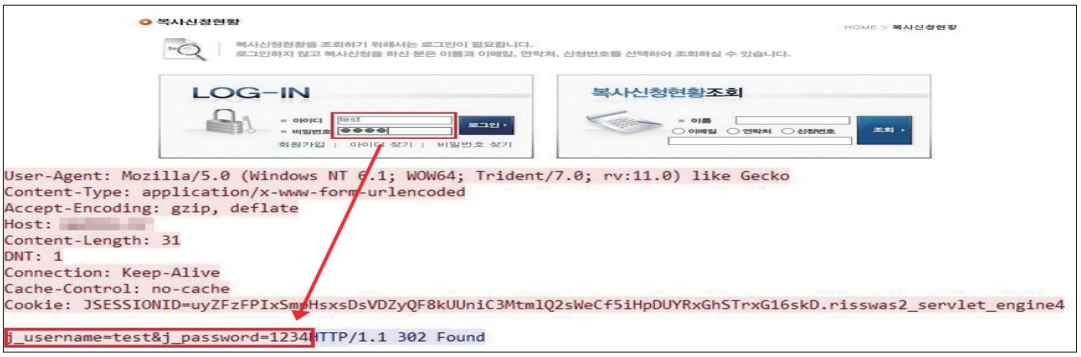
설 명	민감한 데이터를 평문으로 통신채널을 통해서 송수신 할 경우, 통신채널 스니핑(sniffing)을 통해 인가되지 않은 사용자에게 민감한 데이터가 노출될 수 있는 취약점
점검목적	서버와 클라이언트 간 통신 시 데이터가 평문으로 전송되어 정보가 유출되는 위험을 방지하기 위함
점검내용	1) 중요 정보 송수신 페이지가 암호화 통신(https, 데이터 암호화 등)을 하는지 확인

▶ 점검 방법

1) 네트워크 패킷 모니터링 프로그램을 이용하여 네트워크 트래픽을 확인
※ 공개용 패킷 모니터링 프로그램 : Wireshark(http://www.wireshark.org)



가) 중요 정보 송수신 페이지(로그인, 개인정보 수정 등)에서 정보 송수신 시도 시 네트워크 패킷 모니터링 프로그램에서 중요 정보가 평문으로 전송되지 확인

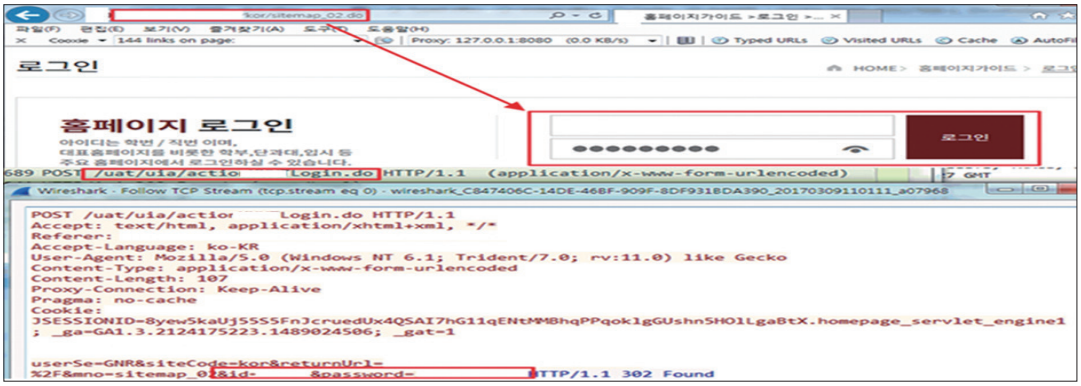


▶ 대응 방법

1) 웹 서버 내에서의 조치
 가) 웹 서버는 전자서명인증서, SSL(Secure Socket Layer)을 이용하여 사용자 식별 및 DATA 전송 시 암호화 통신으로 데이터 전송의 안전성을 확보
 나) 조치 완료 후 인증과정 등의 주요 정보 노출 여부를 재점검
2) 홈페이지 개발 보안 조치
 가) 홈페이지는 중요정보와 관련된 민감한 데이터(개인정보, 비밀번호 등) 전송 시 통신채널(또는 전송데이터) 암호화 적용
 나) 암호알고리즘 적용 시 IT보안인증 사무국이 안전성을 확인한 검증필 암호모듈 사용

▶ 사례

- OO대학은 로그인 과정에서 사용자와 서버간의 통신 정보가 암호화되지 않아 감청을 통해 사용자 정보 획득이 가능

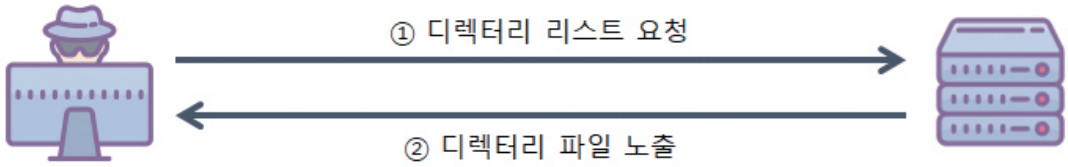


Ⅵ 잘못된 보안 구성

개요			
기본적으로 탑재되어야 하는 보안구성이 제대로 적용 되어 있지 않거나, 애플리케이션에서 사용되는 소프트웨어가 최신 상태를 유지하지 않을 경우 발생할 수 있는 보안 약점			
취약점 항목			
16	디렉터리 인덱싱	17	불필요한 Method 지원
18	취약한 파일 존재	19	히든필드 조작 취약점
20	알려진 취약점이 있는 구성요소 사용		

16. 디렉터리 인덱싱

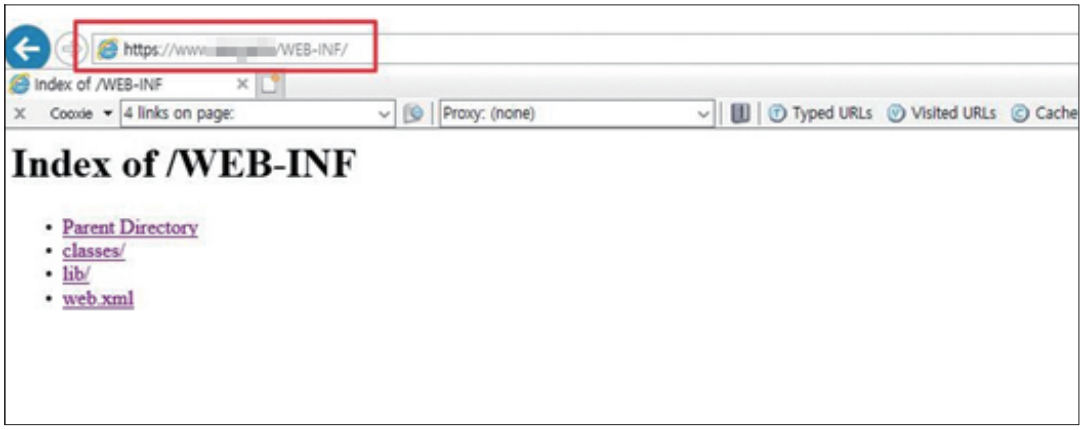
분류	취약점 항목	위험도
잘못된 보안 구성	디렉터리 인덱싱	중



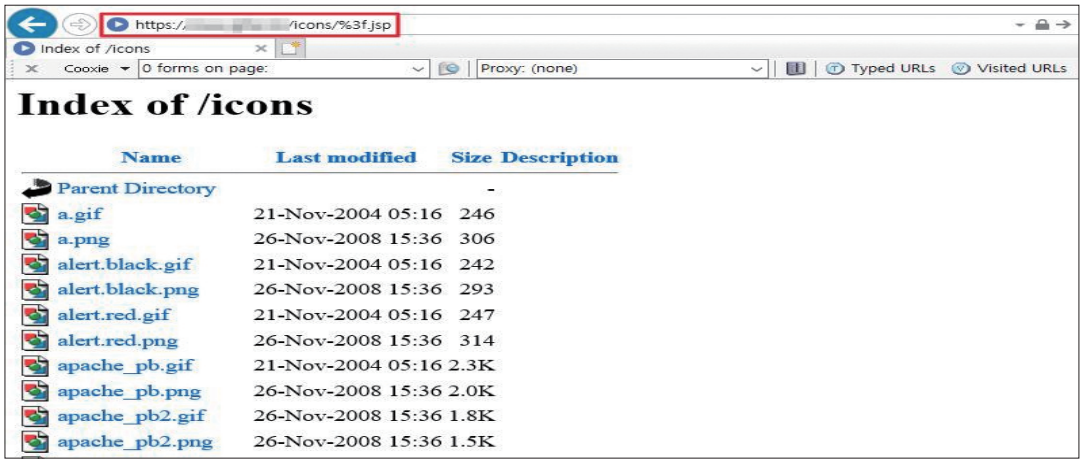
설 명	특정 디렉터리 내 파일 리스트를 노출하여 응용시스템의 구조를 외부에 허용할 수 있고, 민감한 정보 및 시스템 설정 파일 등이 노출될 경우 보안상 심각한 위험을 초래할 수 있음
점검목적	특정 디렉터리 내 불필요한 파일 정보의 노출을 차단하여 공격자에 의한 추가 공격을 방지하기 위함
점검내용	1) URL주소창에 디렉터리를 입력하여 인덱싱 여부 확인 2) 디렉터리 끝에 특수 문자열을 붙여 인덱싱 여부 확인 3) 구글 고급 검색을 통한 디렉터리 노출 확인

▶ 점검 방법

1) URL에 디렉터리 경로 입력 후 인덱싱 여부 확인



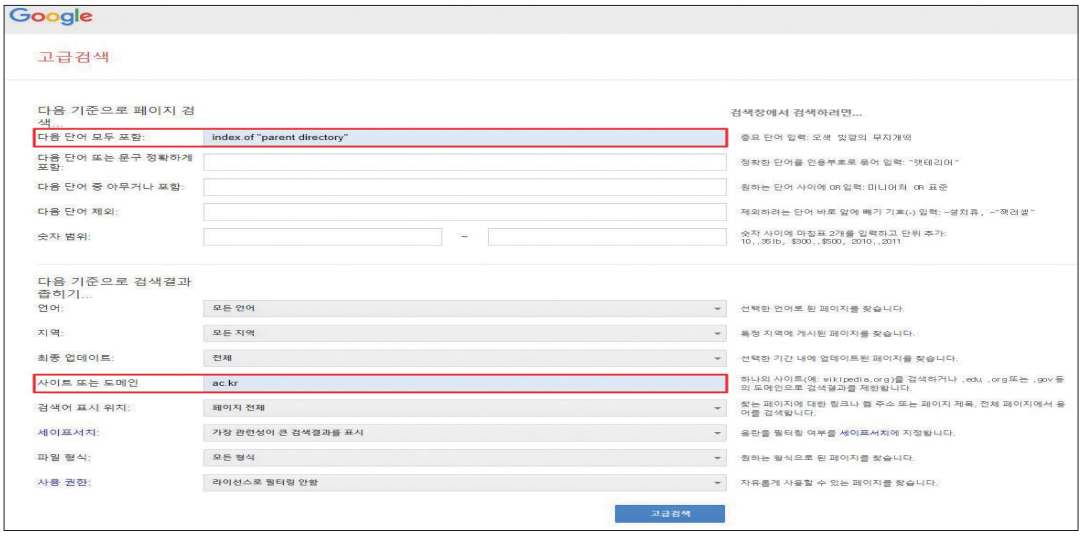
2) 디렉터리 경로 끝에 특수문자(%3f.jsp, %00 등) 문자열 삽입 후 인덱싱 여부 확인



3) 구글 고급 검색을 통한 디렉터리 노출 확인

웹 애플리케이션(웹 서버)	검색어
IIS	Parent Directory
Apache	Directory Listing
Tomcat	Directory Listing
기타	Index of

IIS 웹 서버 검색 예)
index.of "Parent Directory"



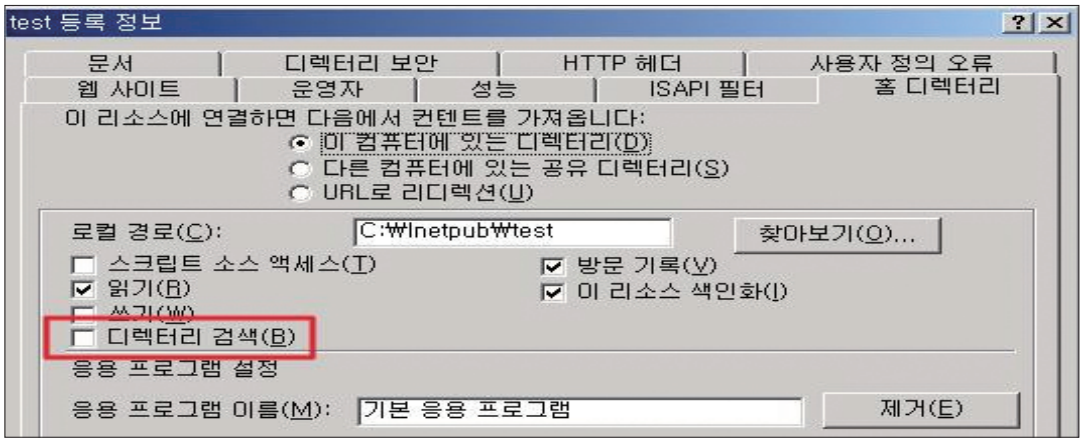
▶ 대응 방법

1) 웹 서버 내에서의 조치

가) 디렉터리 리스팅이 불가능 하도록 모든 페이지에 대해 디렉터리 리스팅 차단 설정

① IIS 웹 서버 조치 방법

[제어판] → [관리도구] → [인터넷 서비스 관리자] → 웹 사이트 선택 후 마우스 오른쪽 버튼 클릭 → [등록 정보] → [홈 디렉터리] → '디렉터리 검색(B)'의 체크 해제



② Apache 웹 서버 조치 방법

아파치 웹 서버의 설정 파일인 httpd.conf 파일에서 Options의 'Indexes' 지시어 삭제 후 저장

```
<Directory "/usr/local/apache/htdocs">
  Options Indexes FollowSymLinks Includes
</Directory>
```

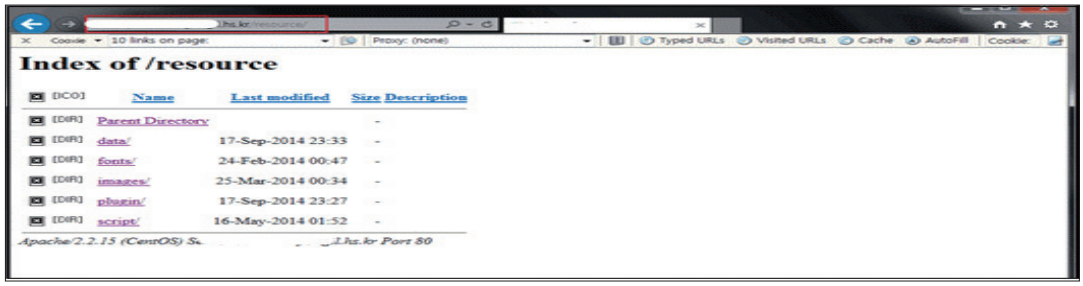
③ Tomcat 조치 방법

web.xml 파일에서 디렉터리 나열을 차단하는 지시어(listings, false)를 설정

```
<init-param>
  <param-name>listings</param-name>
  <param-value>false</param-value>
</init-param>
```

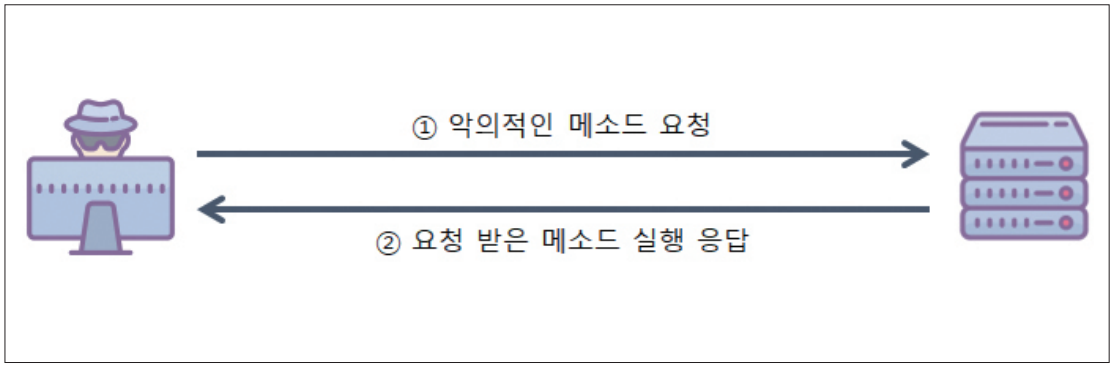
▶ 사례

- OO학교는 일부 페이지에서 디렉터리 나열 취약점이 존재하여 웹 서버 내 파일 목록 열람 가능



17. 불필요한 Method 지원

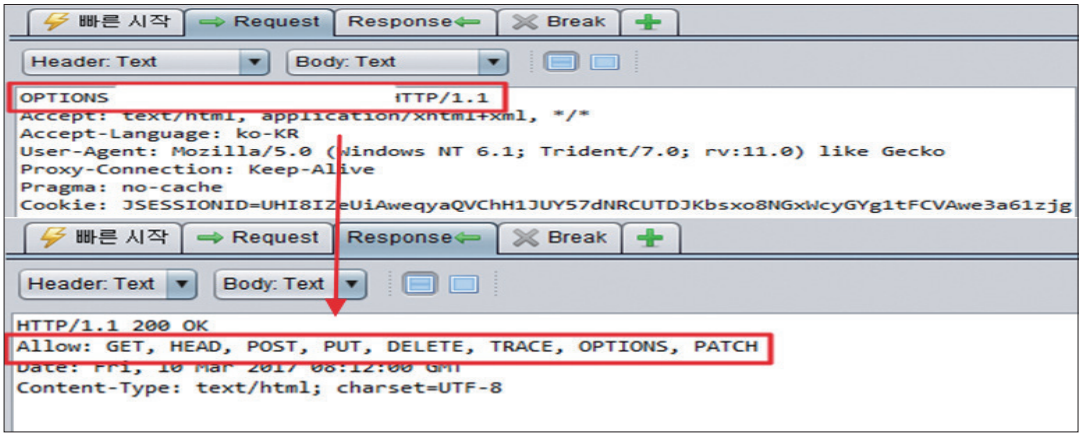
분류	취약점 항목	위험도
잘못된 보안 구성	불필요한 Method 지원	중



설 명	웹 서비스 제공 시 불필요한 Method(PUT, DELETE, OPTIONS 등) 허용으로 공격자에 의해 악성파일을 업로드 하거나 중요파일 삭제가 가능해지는 취약점
점검목적	불필요한 Method를 통한 악의적인 행위를 차단하기 위함
점검내용	1) 불필요한 Method가 활성화 되어 있는지 여부 확인

▶ 점검 방법

1) 점검 도구를 이용하여 불필요한 Method가 활성화 되어 있는지 확인



▶ 대응 방법

1) 웹 서버 내에서의 조치

가) 홈페이지 운영에 불필요한 Method(PUT, DELETE, OPTIONS 등) 비활성화

① Apache 웹 서버 조치 방법

아파치 웹 서버의 설정 파일인 httpd.conf 파일에서 허용할 메소드에 대해 'LimitExcept' 지시자를 사용하여 저장
(Limit 지시자(Black-List 방식)가 아닌 LimitExcept 지시자(White-List 방식) 사용 권고)

```
<Directory "/">
  <LimitExcept GET POST> // 허용할 메소드
    Order deny,allow
    Deny from all
  </LimitExcept>
</Directory>
```

② Tomcat 조치 방법

web.xml 파일에서 http-method 지시어에 불필요한 Method를 입력하여 비활성화
예) 모든 페이지(/*)에 대하여 PUT, DELETE, TRACE, OPTIONS Method 사용 제한

```
<security-constraint>
  <web-resource-collection>
    <web-resource-name>Protected Context</web-resource-name>
      <url-pattern>/*</url-pattern>
      <http-method>PUT</http-method>
      <http-method>DELETE</http-method>
      <http-method>TRACE</http-method>
      <http-method>OPTIONS</http-method>
    </web-resource-collection>
  <auth-constraint />
</security-constraint>
```

▶ 사례

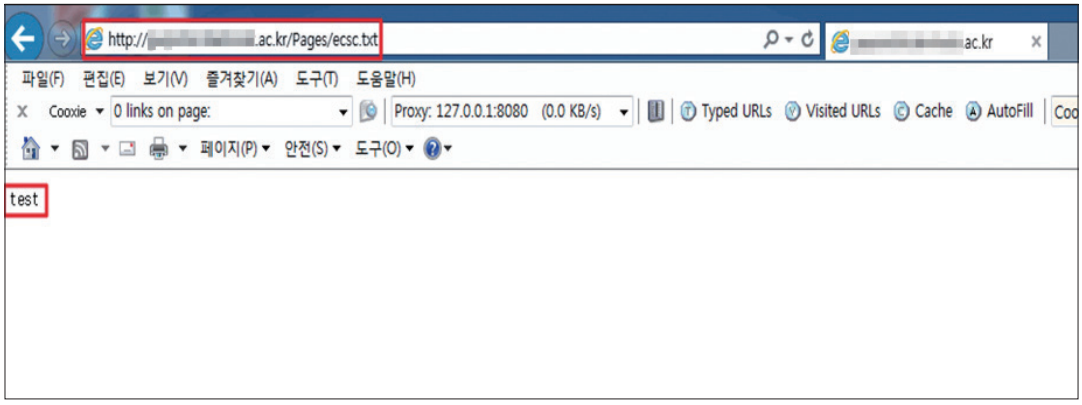
- OO대학에 불필요한 Method(OPTIONS) 허용으로 인한 Method 정보 노출 확인

```
OPTIONS http://[redacted].ac.kr/Pages/default.aspx HTTP/1.1
Accept: text/html, application/xhtml+xml, */*
Accept-Language: ko-KR
User-Agent: Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.2; Trident/6.0)
Proxy-Connection: Keep-Alive
Cookie: UTF8_Option=12092279; LoginCookie=
00594813229906975704529122993703071000504528093100500050460509311107576637585317375816724528452832662004; BF
get%5Ehttp%3A%2F%2Fgw.[redacted].ac.kr%2Fsso%2Flogout.aspx%3Flogout%3D1%24get%5Ehttp%3A%2F%2Fidisk.[redacted].ac.k

HTTP/1.1 200 OK
Cache-Control: private,max-age=0
Allow: GET, POST, OPTIONS, HEAD, MKCOL, PUT, PROPFIND, PROPPATCH, DELETE, MOVE, COPY, GETLIB, LOCK, UNLOCK
Content-Length: 0
Expires: Thu, 19 Jul 2018 04:27:47 GMT
Accept-Ranges: none
Server: Microsoft-IIS/7.5
SPRequestGuid: 27b82449-25cd-4add-a5b2-ecc288e88576
Set-Cookie: WSS_KeepSessionAuthenticated={15b6fa95-d2c6-428f-9620-52f25e77297f}; path=/
```

- PUT Method가 허용된 것을 확인 후 파일 생성 시도

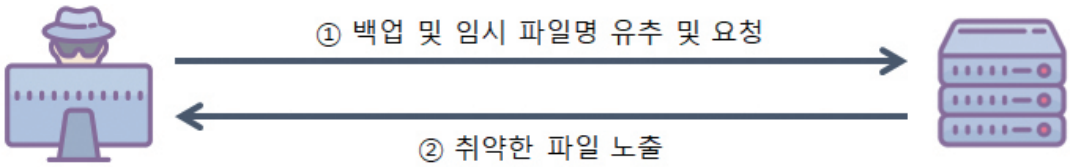
```
PUT http://[redacted].ac.kr/Pages/ecsc.txt HTTP/1.1
Accept: text/html, application/xhtml+xml, */*
Accept-Language: ko-KR
User-Agent: Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.2; Trident/6.0)
Proxy-Connection: Keep-Alive
Cookie: UTF8_Option=12092279; LoginCookie=
00594813229906975704529122993703071000504528093100500050460509311107576637585317375816724528452
get%5Ehttp%3A%2F%2Fgw.[redacted].ac.kr%2Fsso%2Flogout.aspx%3Flogout%3D1%24get%5Ehttp%3A%2F%2Fidisk.

test

HTTP/1.1 201 CREATED
Cache-Control: private,max-age=0
Content-Length: 0
Expires: Thu, 19 Jul 2018 04:29:08 GMT
Last-Modified: Fri, 03 Aug 2018 04:29:09 GMT
ETag: "{6D9062CD-6164-4EBB-AC92-96D3F350E759},1"
Location: http://[redacted].ac.kr/Pages/ecsc.txt
Server: Microsoft-IIS/7.5
SPRequestGuid: f8d51615-f6a2-4095-8549-ea12b7ab78dc
Set-Cookie: WSS_KeepSessionAuthenticated={15b6fa95-d2c6-428f-9620-52f25e77297f}; path=/
```

- 서버 내에 파일이 정상 생성된 것을 확인



18. 취약한 파일 존재

분류	취약점 항목	위험도
잘못된 보안 구성	취약한 파일 존재	중



설 명	웹 루트 하위에 내부 문서나 백업파일, 로그파일, 압축파일과 같은 파일이 존재할 경우 파일명을 유추하여 파일명을 알아내고, 직접 요청하여 해킹에 필요한 서비스 정보를 획득할 수 있는 취약점
점검목적	취약한 파일의 위치를 예측하여 파일 및 정보 획득을 방지하기 위함
점검내용	1) 웹 루트 디렉터리 내 웹 서비스에 불필요한 확장자 파일이 존재 하는지 확인 2) 각종 샘플페이지의 디렉터리 및 파일 존재여부 확인

▶ 점검 방법

1) 웹 루트 디렉터리 내 웹 서비스에 불필요한 확장자 파일이 존재 하는지 확인
※ 문서파일일 경우 내용에 개인정보 등의 주요정보 존재여부 확인 필요

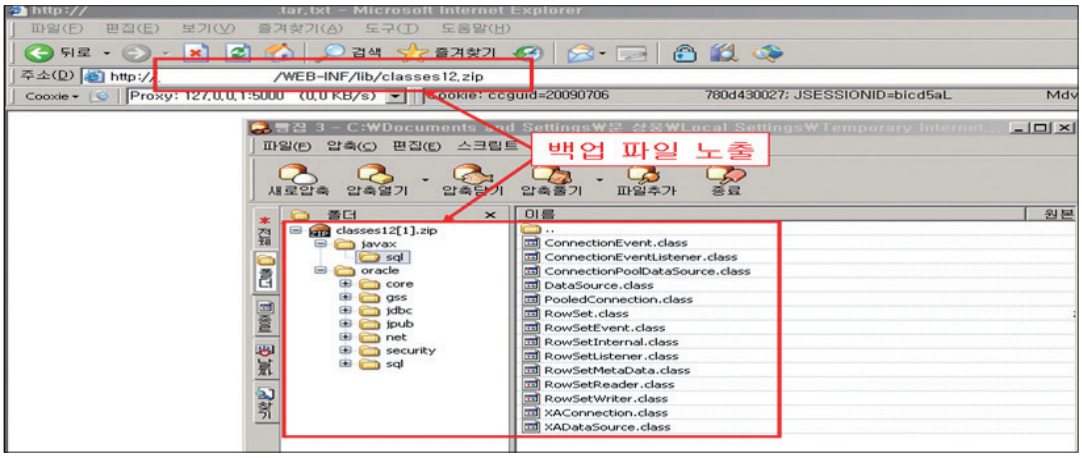
구분	검색할 파일의 형식(확장자)
압축파일	.zip, .rar, .alz, .tar, .gz, .gzip 등의 압축파일
백업파일	.bak, .org 등
로그파일	.log, .txt 등
설정파일	.sql, .ini, .bat 등
문서파일	.hwp, .doc, .xls, .ppt, .pdf 등
기타	test.*, imsi.* .tmp 등

Windows 의 검색 예)

dir /[웹 서버디렉터리] /s *.bak

Unix, Linux 의 검색 예)

find /[웹 서버디렉터리] -name ".*.bak"



※ PHP 언어로 개발한 웹 서버의 경우 아래와 같은 정보를 출력하는 웹페이지(phpinfo.php)가 존재 하는지 점검

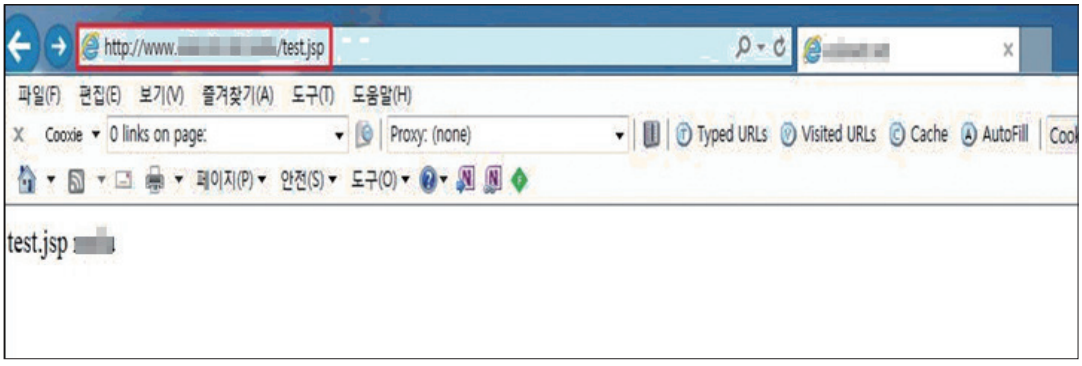
검색 예)

find /[웹 서버디렉터리] -name "phpinfo.php"

또는 모든 PHP 파일에 아래의 문자열이 포함되어있는 파일을 조회함
(단, 파일내의 문자열 검색 시 시스템에 과부하가 발생할 수 있음)

grep "phpinfo()" *php

2) 각종 샘플페이지의 디렉터리 및 파일 존재여부 확인



▶ 대응 방법

1) 웹 서버 내에서의 조치

가) 웹 서버는 개발과 운영 환경을 분리하여 운영 환경에서 소스 코드 수정 또는 테스트 목적의 임시 파일을 생성하지 않도록 함

나) 웹 서버의 디렉터리에 존재하는 기본 설치 파일, 임시 및 백업 파일을 조사하여 웹 사용자가 접근하지 못하도록 조치

구분	설치 시 생성되는 기본 파일 위치
아파치(Apache)	ServerRoot/cgi-bin/
톰켓(Tomcat)	TOMCAT_HOME/examples
웹투비(WebToB)	ServerRoot/cgi-bin/

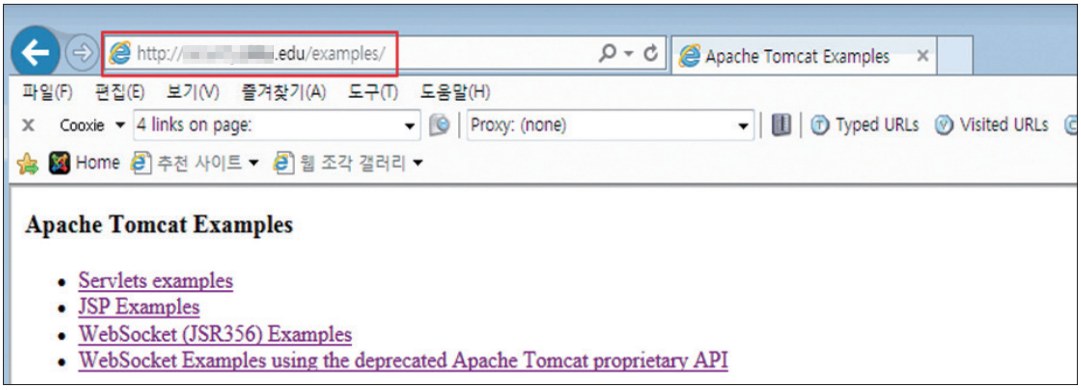
※ 일반적으로 존재하는 백업파일 유형

구분	내용
*.bak	Edit plus, Ultra Edit 등의 작업을 할 경우 기본 설정에 의해 백업파일이 생성
ws_ftp.log	WS FTML를 사용하여 애플리케이션을 업로드 한 경우 로그파일에 디렉터리 나열 구조, 숨겨진 파일들을 알아낼 수 있음
*.tar.gz	주로 웹 애플리케이션을 압축한 형태로 존재
*.zip	주로 웹 애플리케이션을 압축한 형태로 존재
파일명.날짜	main.jsp.20190101과 같은 형식의 백업파일을 사용
*.html.old	기존 파일을 백업하는 개념으로 old를 사용

다) 정기적으로 불필요 파일을 검색하여 제거함

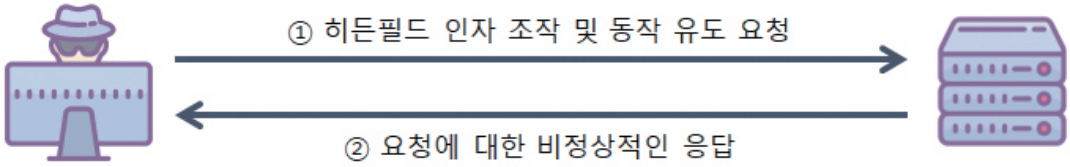
▶ 사례

- OO대학에서 Tomcat examples 페이지가 존재하는 것을 확인



19. 히든필드 조작

분류	취약점 항목	위험도
잘못된 보안 구성	히든필드 조작	중

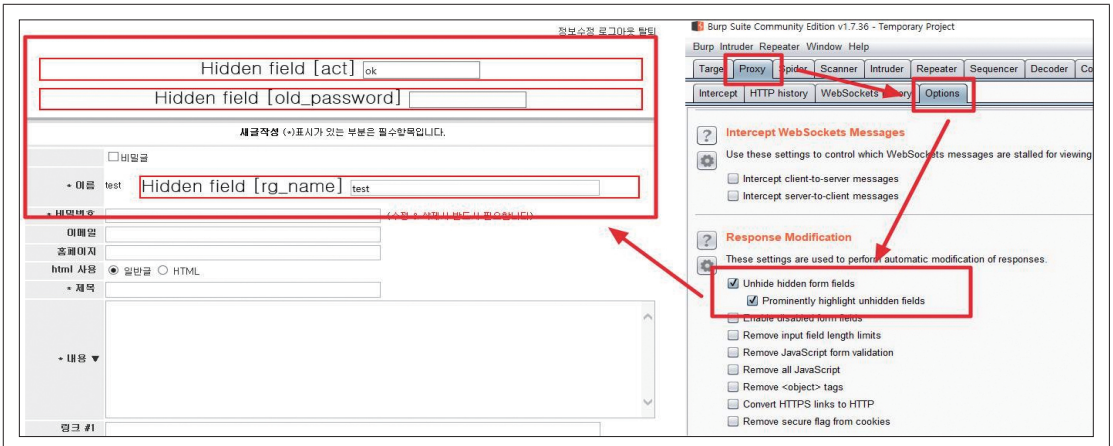


설 명	Hidden Field의 인자를 조작하거나 숨겨진 기능을 찾아 애플리케이션의 비정상적인 동작을 유도하여 승인되지 않은 페이지에 접근 가능하거나 애플리케이션 오류를 발생시켜 정보 획득 및 추가 공격을 야기할 수 있는 취약점
점검목적	Hidden Field 조작을 통한 악의적인 행위를 방지하고자 함
점검내용	1) Hidden Field의 데이터를 조작하여 값이 서버에 반영되는지 확인

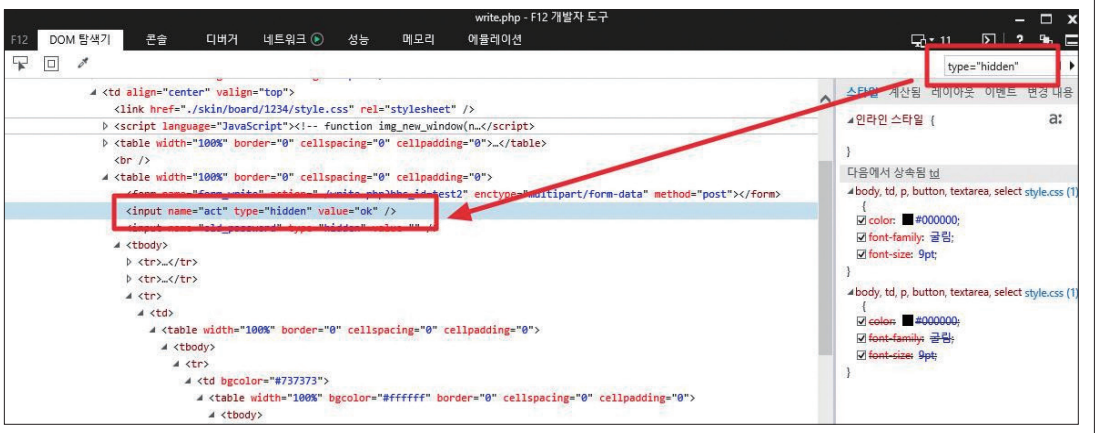
▶ 점검 방법

1) 소스코드 내 Hidden Field 존재 여부 확인
가) 프록시 툴을 사용하여 확인

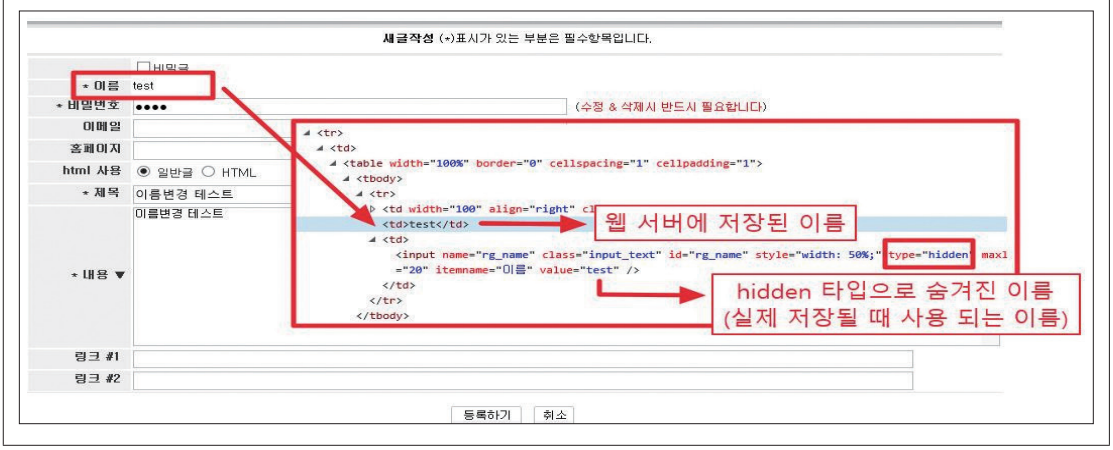
Burp Suite(프록시 툴) → [Proxy 탭] → [Options 탭] → Response Modification 옵션의 'Unhide hidden form fields'항목과 'Prominently highlight unhidden fields'항목 체크 후 페이지 새로고침

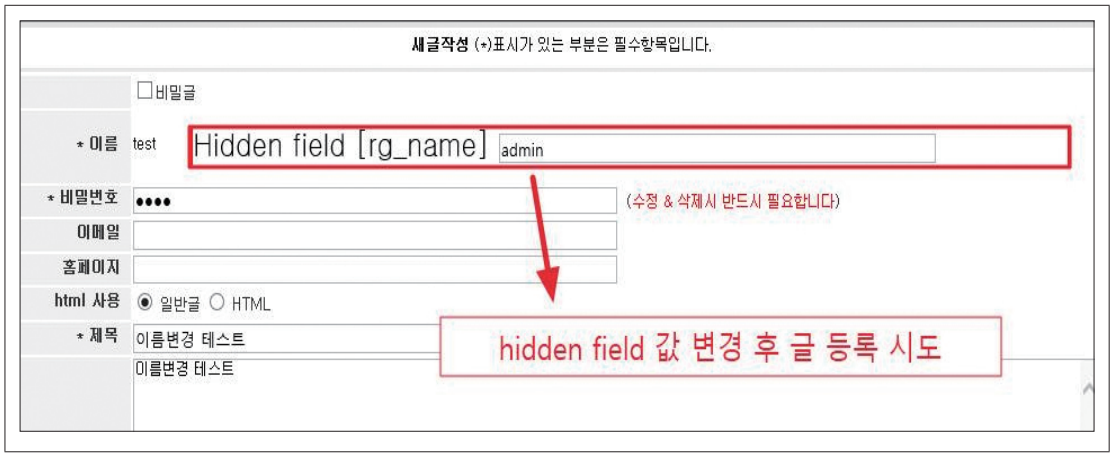


나) 소스코드에서 type="hidden" 검색을 통해 확인



2) Hidden Field의 데이터를 조작하여 값이 서버에 반영되는지 확인



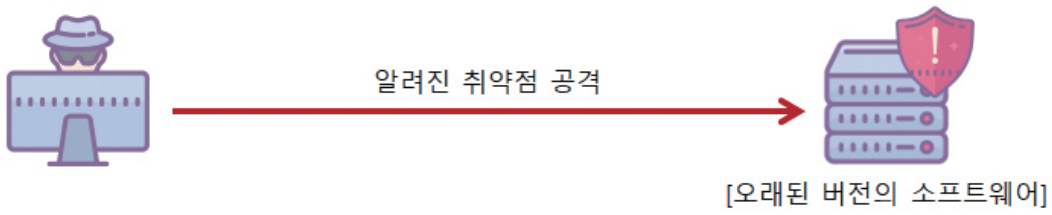


▶ 대응 방법

1) 홈페이지 개발 보안 조치
가) 중요 정보(개인정보, 패스워드 등)는 클라이언트에서 확인 가능한 소스코드 내의 Hidden Field에서의 저장을 지양하고, 저장이 불가피할 경우 해당 내용은 암호화 처리
나) 중요한 정보 및 숨겨진 기능 처리는 서버 측에서 이루어질 수 있도록 설계
다) Hidden Field 데이터에 대한 입력 값 검증 로직을 구현하여 무결성 체크

20. 알려진 취약점이 있는 구성요소 사용

분류	취약점 항목	위험도
잘못된 보안 구성	알려진 취약점이 있는 구성요소 사용	상



설 명	웹 애플리케이션에서 취약점이 알려진 라이브러리, 프레임워크, 모듈 등을 사용함으로서 발생할 수 있는 취약점
점검목적	알려진 취약점을 통한 공격을 사전에 방지하기 위함
점검내용	1) 웹 애플리케이션 및 상용 소프트웨어(공개용 웹 게시판 등)의 버전 확인

▶ 점검 방법

1) 공개용 웹 게시판 버전 확인

Unix, Linux 검색 예)
제로보드 검색 예)

```
find /[웹 서버디렉터리] -name "license.txt" -exec ls -alt {} ₩; -exec grep "배포버젼 :" {} ₩;
```

테크노트 검색 예)

```
find /[웹 서버디렉터리] -name "config.cgi" -exec ls -alt {} ₩; -exec grep "# 최종수정 배포일" {} ₩; >
find /[웹 서버디렉터리] -name "main.cgi" -exec ls -alt {} ₩; -exec grep "Update:" {} ₩;
```

브라우저를 통한 검색 예)
제로보드 검색 예)
http://홈페이지주소/bbs/license.txt
http://홈페이지주소/zb/license.txt
http://홈페이지주소/zeroboard/license.txt

테크노트 검색 예)
http://홈페이지주소/게시판 디렉터리/config.cgi
http://홈페이지주소/게시판 디렉터리/main.cgi

2) 운영 중인 웹 서버 버전 확인

Linux 검색 예)
① **Apache**
httpd –v
② **Tomcat**
chkconfig --list | grep tomcat
tomcat version | head –3

Ubuntu 검색 예)
① **Apache**
apache2 –v
② **Tomcat**
sudo /tomcat 설치경로/version.sh

Window 검색 예)
① **Apache**
cd ₩apache 설치 경로₩
httpd -v
② **Tomcat**
cd ₩tomcat 설치 경로₩
version
③ **IIS**
시작 > 실행 > regedit 입력 >
HKEY_LOCAL_MACHINE₩SOFTWARE₩Microsoft₩InetStp₩VersionString
경로의 Version 확인

▶ 대응 방법

1) 웹 서버 내에서의 조치
가) 웹 서버에는 공개용 웹 게시판(제로보드, 테크노트, 그누보드, 세팔보드 등) 사용 지양
나) 부득이하게 사용해야 할 경우 보안 취약점이 존재하지 않도록 보안 패치 또는 최신버전의 제품으로 설치하며, 정기적으로 게시판 배포 사이트에 방문하여 보안 취약점 정보를 확인

※ 게시판 재설치 시 기존 데이터가 삭제될 수 있으므로 백업 작업을 수행한 후 재설치

다) 서버정보 노출 방지

① Apache 웹 서버 조치 방법

아파치 웹 서버의 설정 파일인 httpd.conf 파일에서 ServerTokens Prod 설정

```
ServerTokens Prob
ServerSignature Off
```

② Tomcat 조치 방법

server.xml 파일에서 Connector 태그에 server="" 추가

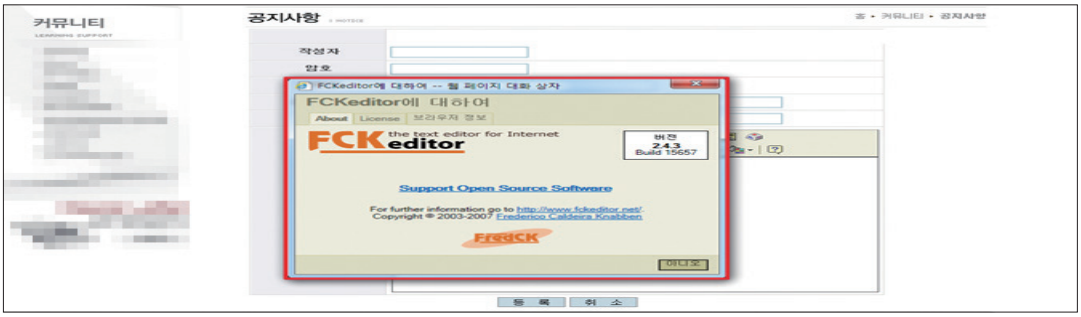
```
<Connector port="8080" protocol="HTTP/1.1"
        server="Tomcat"  // 서버 정보에 표시할 문자
        connectionTimeout="20000"
        redirectPort="8443" />
```

라) 사용하고 있는 구성요소에 대한 취약점을 항상 체크 및 제거

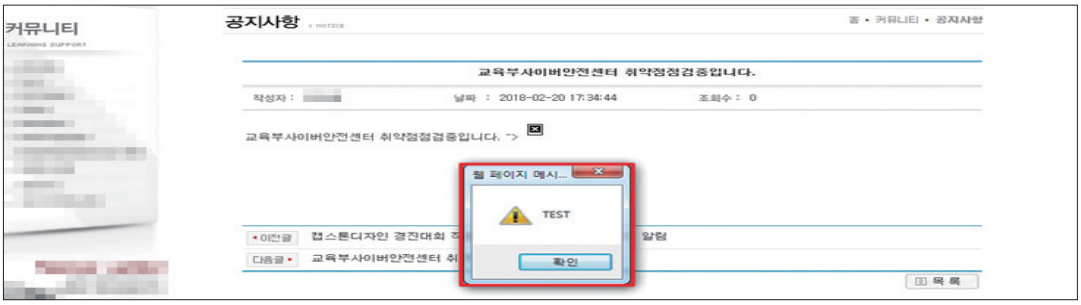
마) 취약점이 있는 구성요소를 보안 패치 또는 최신 버전으로 업데이트

▶ 사례

- OO대학은 취약한 버전의 FCKeditor 게시판을 사용하는 것을 확인



- 알려진 버전의 취약점(XSS)을 통한 공격 가능 확인



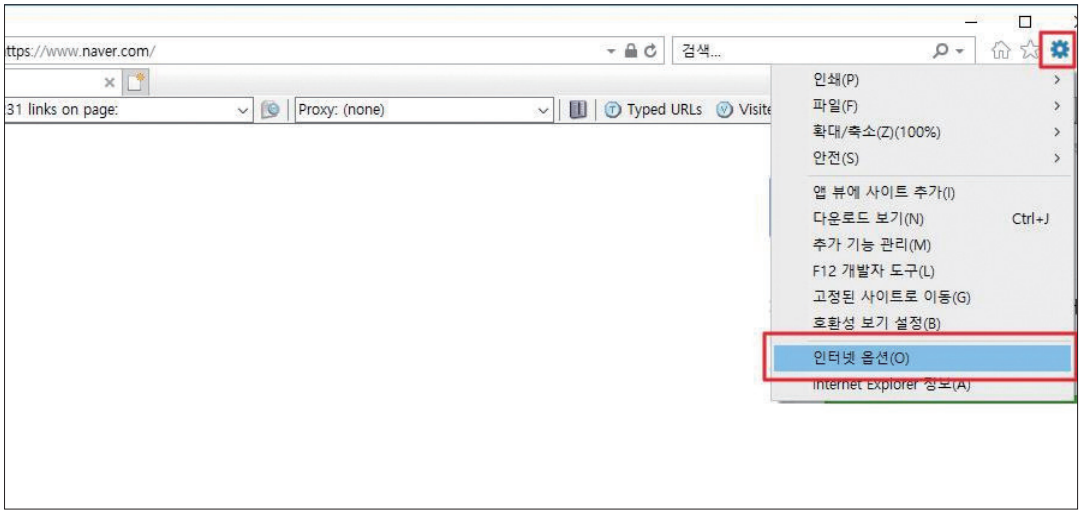
Ⅶ. 부록

1. 프록시 툴 사용 가이드

1. 인터넷 옵션 설정

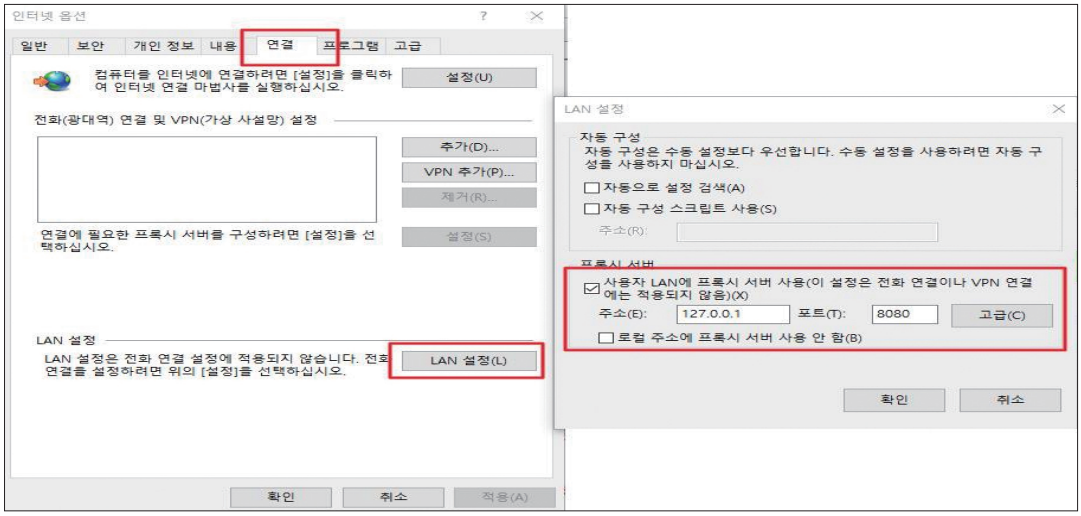
※ 프록시 툴 사용 시 인터넷 옵션 설정 후 사용

- 인터넷 브라우저(IE)에서 인터넷 옵션 선택



- [인터넷 옵션] 〉 [연결] 〉 [LAN 설정]에서 “사용자 LAN에 프록시 서버 사용” 선택 후 주소 및 포트 설정

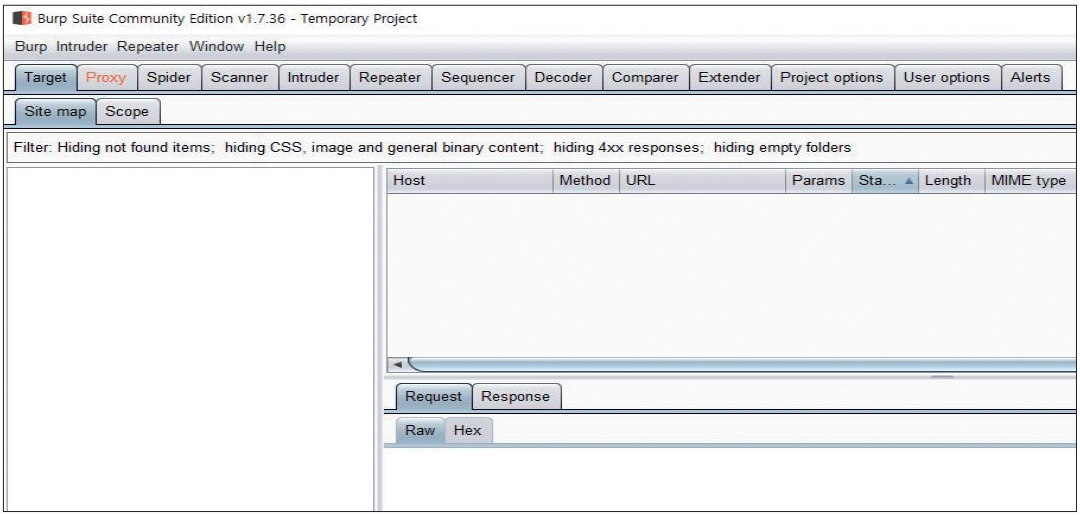
  ※ IP = 127.0.0.1(로컬 호스트), PORT = 8080



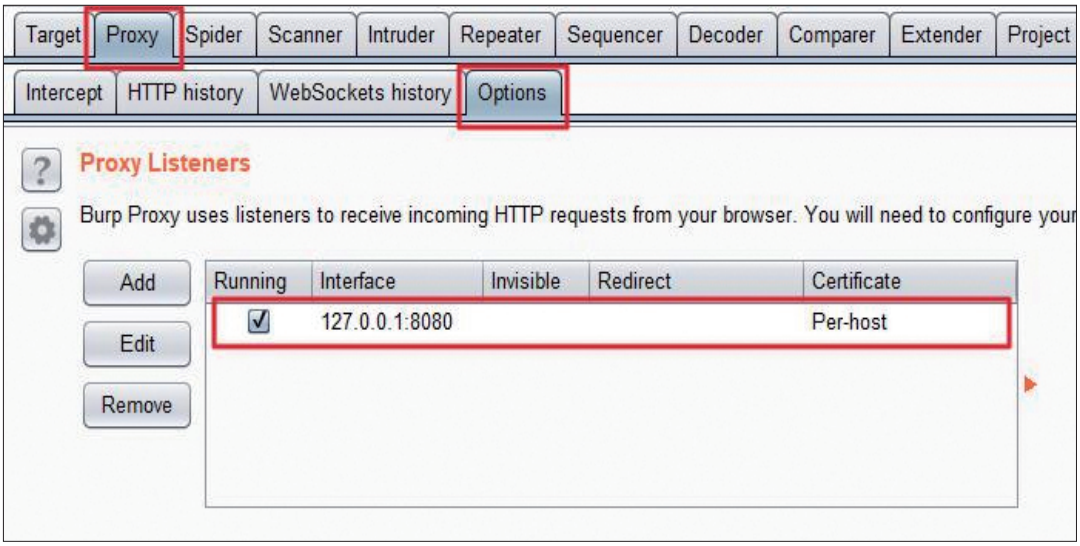
2. 프록시 툴 설정

2-1. 프록시 툴[Burp Suite]

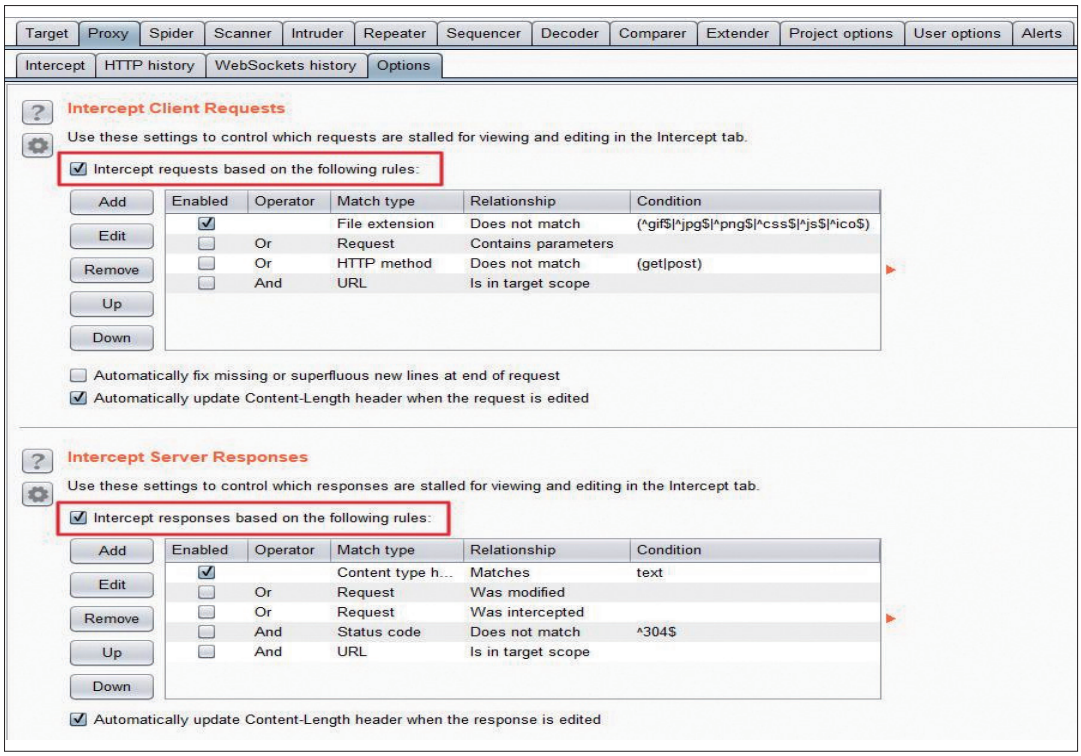
1. Burp Suite 설치 및 실행
 - Burp Suite 설치 후 프로그램 실행
  ※ Burp Suite 설치 경로 (https://portswigger.net/burp/communitydownload)



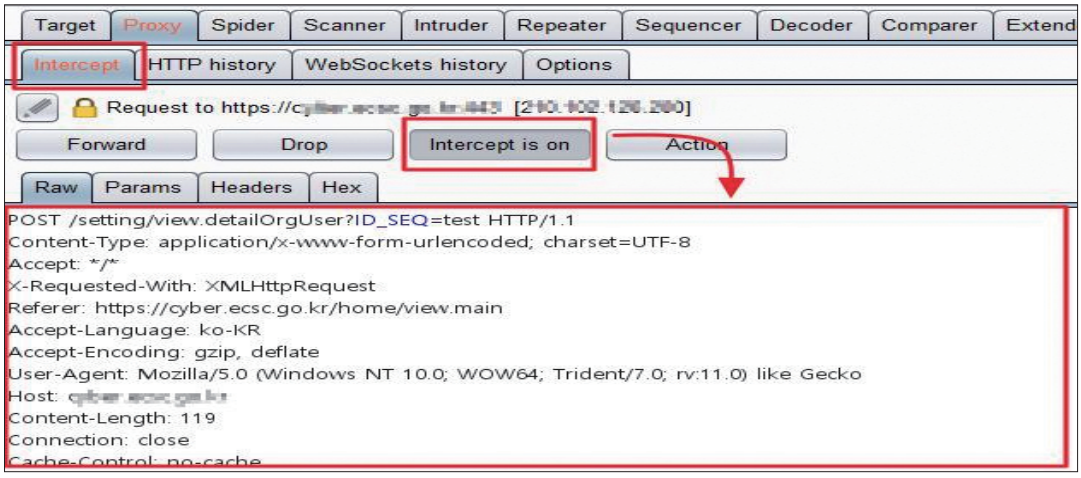
2. 프록시 옵션 설정
 - [Proxy 탭] 〉 [Options 탭] 〉 Proxy Listeners 옵션에서 인터페이스 선택(또는 추가)
  ※ 주소 및 포트는 브라우저 인터넷 옵션에서 설정한 값과 동일하게 설정
  ※ IP = 127.0.0.1(로컬 호스트), PORT = 8080



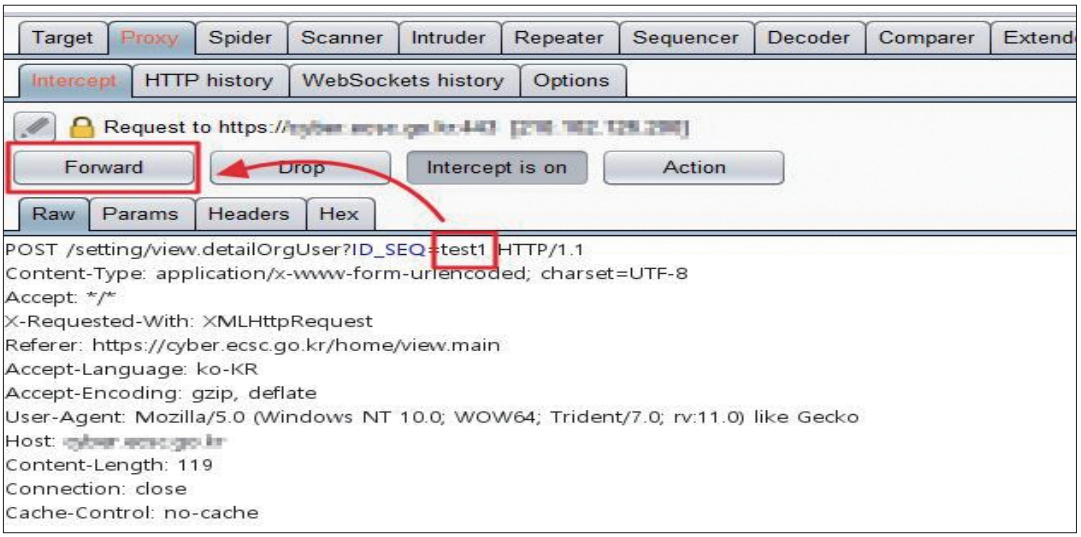
- [Proxy 탭] 〉 [Options 탭] 〉 Intercept Client Requests 옵션 및 Intercept Server Responses 옵션 선택
  ※ Intercept Client Requests = 페이지 요청 값 가로채기
  ※ Intercept Server Responses = 페이지 응답 값 가로채기



3. Burp Suite를 이용하여 데이터 가로채기
   - 프록시 툴에서 [proxy 탭] 〉 [Intercept 탭]의 'Intercept is on'으로 선택 후 페이지 요청 시 데이터 가로채기 가능

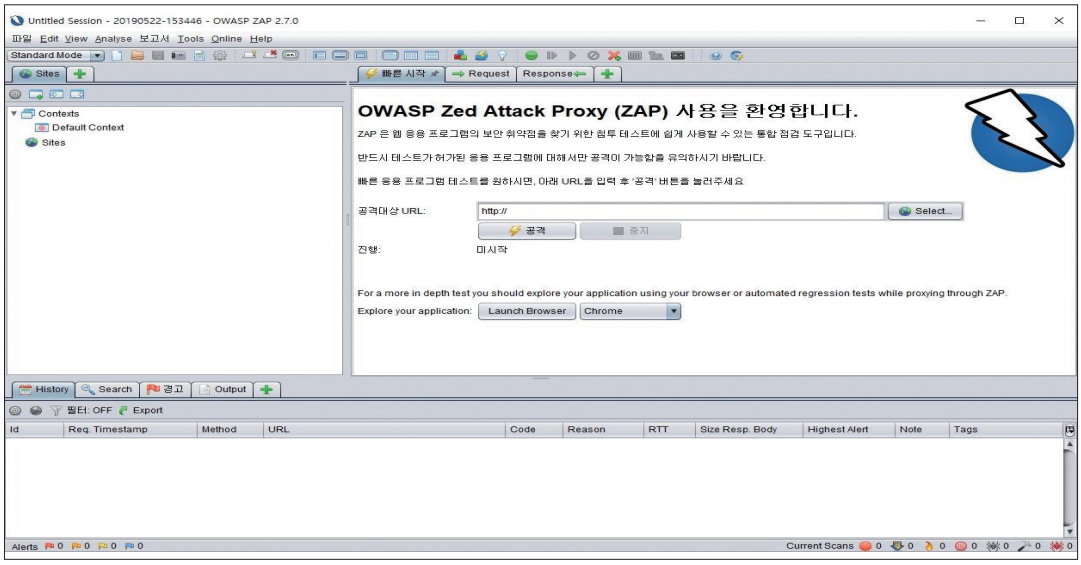


- 가로챈 데이터 변조 후 'Forward'를 통해 웹 서버에 전송
  ※ 웹 서버에서 데이터 검증을 하지 않을 경우, 공격자는 해당 방법으로 다양한 우회 공격이 가능

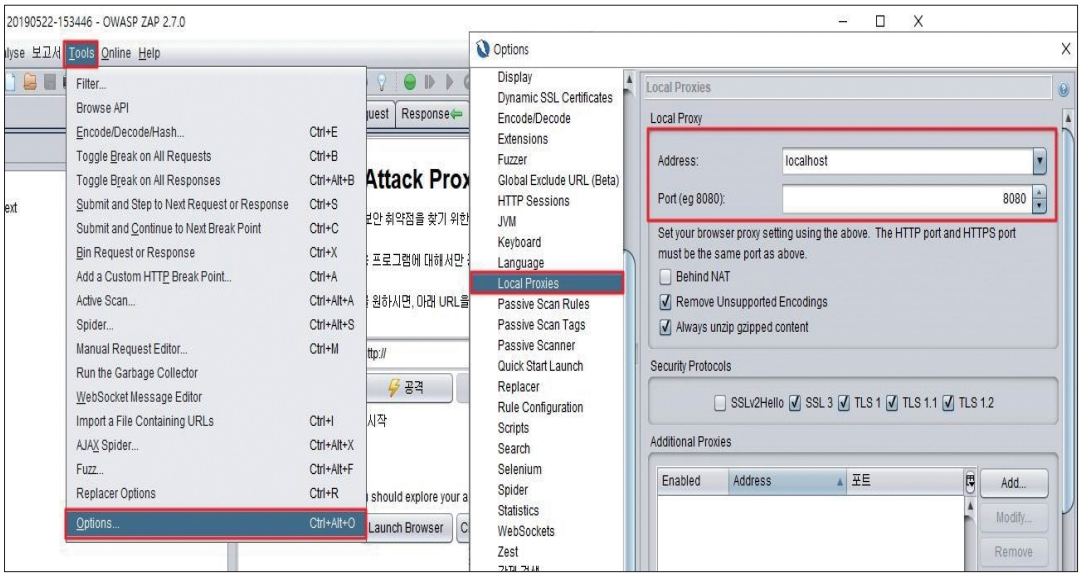


2-2. 프록시 툴[OWASP Zed Attack Proxy(ZAP)]

1. OWASP ZAP 설치 및 실행
 - OWASP ZAP 설치 후 프로그램 실행
  ※ OWASP ZAP 설치 경로
   (https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project)

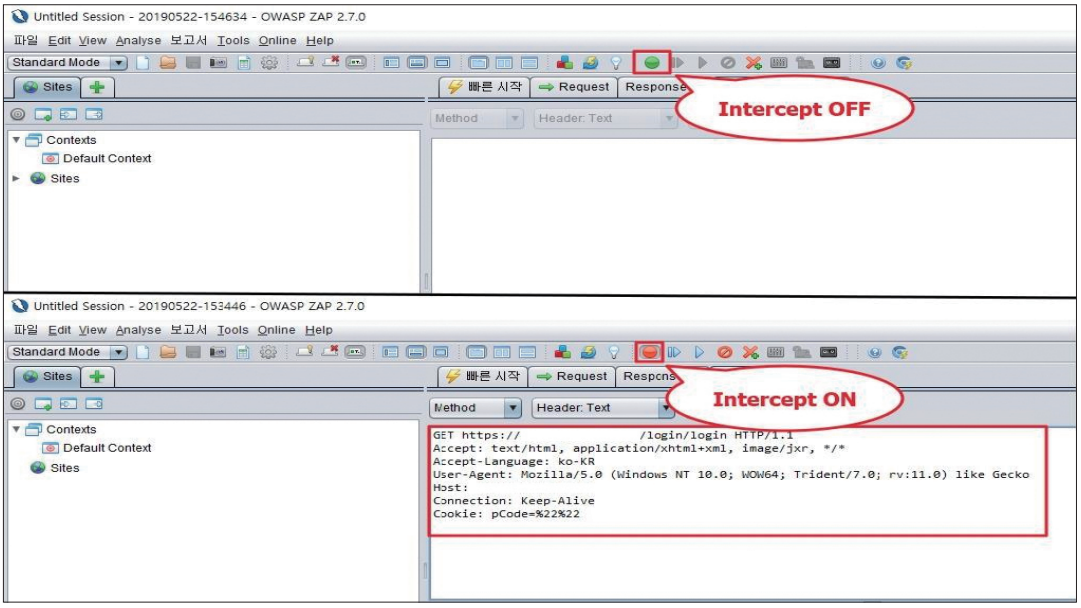


2. 프록시 옵션 설정
 - [Tool 탭] 〉 [Options] 〉 [Local Proxies] 〉 Local Proxy 옵션에서 인터페이스 설정
  ※ 주소 및 포트는 브라우저 인터넷 옵션에서 설정한 값과 동일하게 설정
  ※ IP = 127.0.0.1(로컬 호스트), PORT = 8080



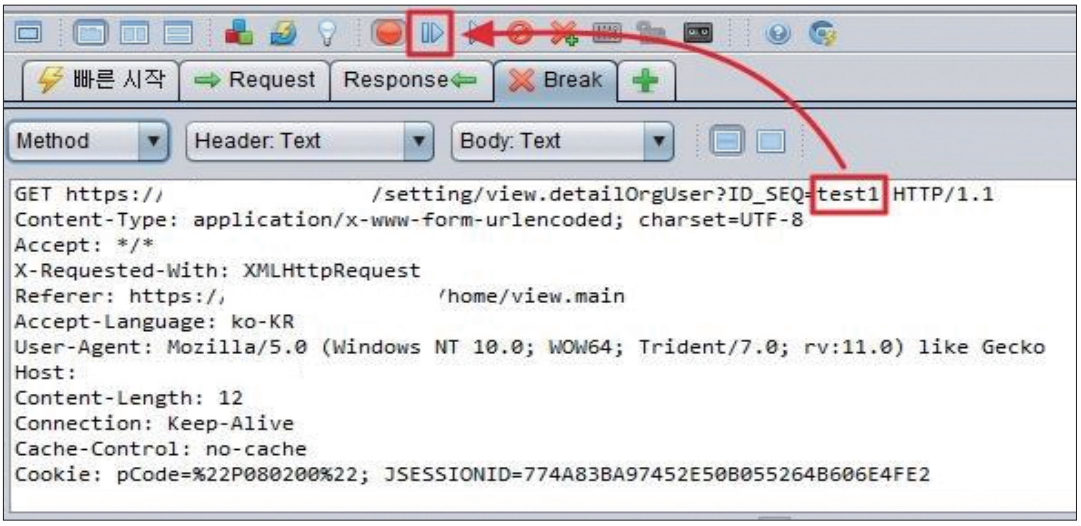
3. OWASP ZAP를 이용하여 데이터 가로채기

- 프록시 툴에서 [기능 탭]에서 Intercept 버튼을'빨간불'로 변경 후 페이지 요청 시 데이터 가로채기 가능



- 가로챈 데이터 변조 후 'Forward'를 통해 웹 서버에 전송

※ 웹 서버에서 데이터 검증을 하지 않을 경우, 공격자는 해당 방법으로 다양한 우회 공격이 가능



Ⅶ. 부록

2. 홈페이지 취약점 자가점검 시스템 사용자 가이드

1. 개요

1.1 운영 계획

- 교육사이버위협 정보공유시스템(https://cyber.ecsc.go.kr)을 통해 홈페이지 취약점 점검 시스템에 접속하여 점검
- 기관담당자가 홈페이지 취약점 점검시스템에 점검 홈페이지 등록 후 점검 시간을 예약하면 자동점검 실시

1.2 유의 사항

- 웹 취약점 점검으로 네트워크 및 시스템 부하가 발생될 수 있으므로, 웹 취약점 점검 시간에는 보안 담당자 및 시스템 담당자 대기
- 자가 점검 시 웹 방화벽, 침입방지시스템(IPS) 등 보안장비에서 차단되지 않도록 사전 예외처리 필수 ※ 진단 서버 IP 및 세부정보는 홈페이지 취약점 점검 시스템 사용자 매뉴얼 참조

2. 홈페이지 취약점 진단 시스템 점검 방법

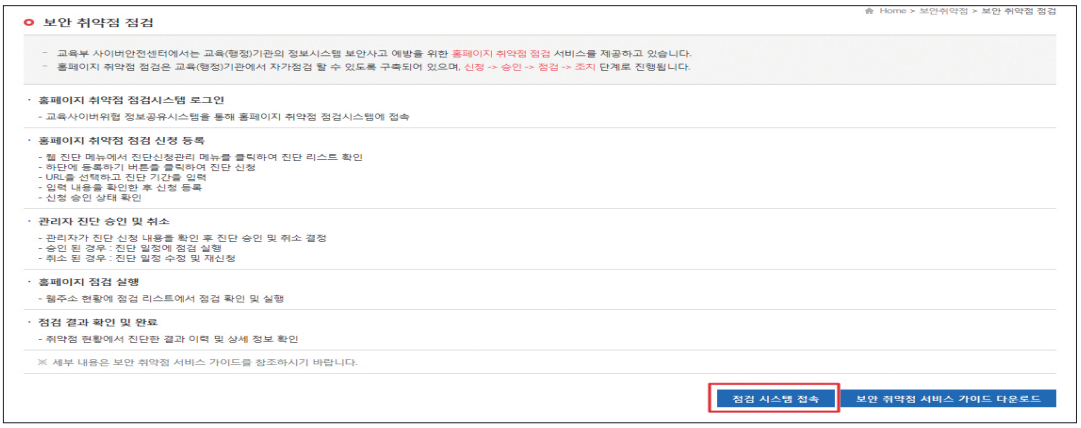
2.1 취약점 점검 시스템 접속

※ 교육사이버위협 정보공유시스템(https://cyber.ecsc.go.kr)에 로그인 후 사용 가능

- 교육사이버위협 정보공유시스템에 로그인

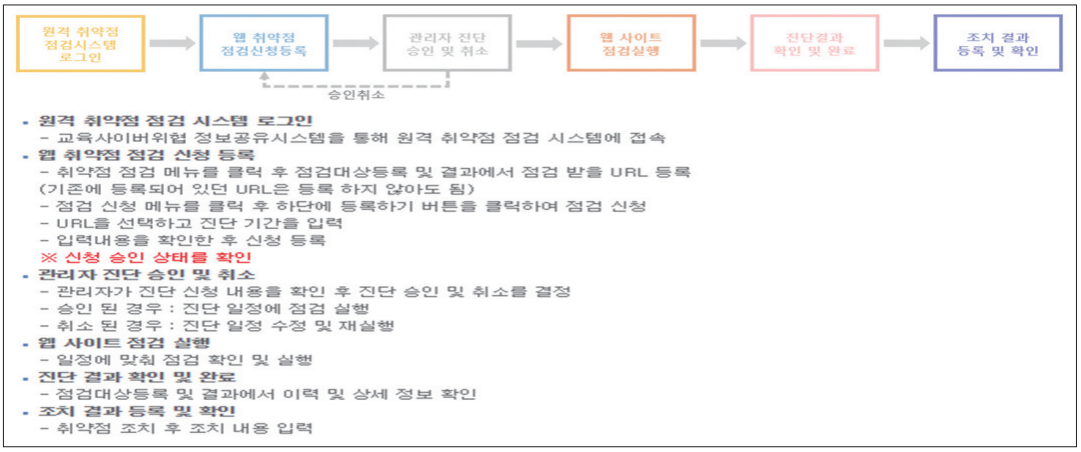
- [보안취약점] 〉 [보안 취약점 점검] 메뉴 선택

- 점검 시스템 접속 버튼 클릭

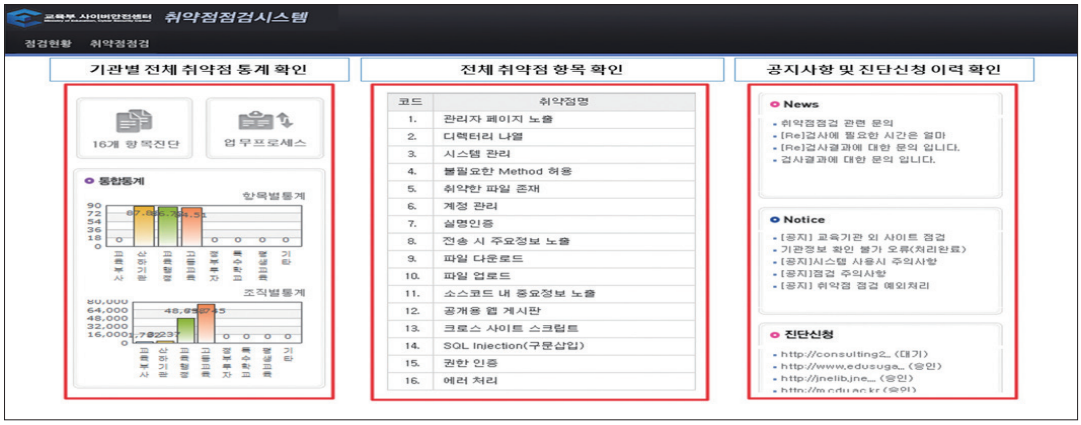


- 취약점 점검 시스템 접속

2.2 웹 취약점 진단 프로세스



2.3 점검 시스템 화면 구성



2.4 점검 현황 확인 및 취약점점검 확인

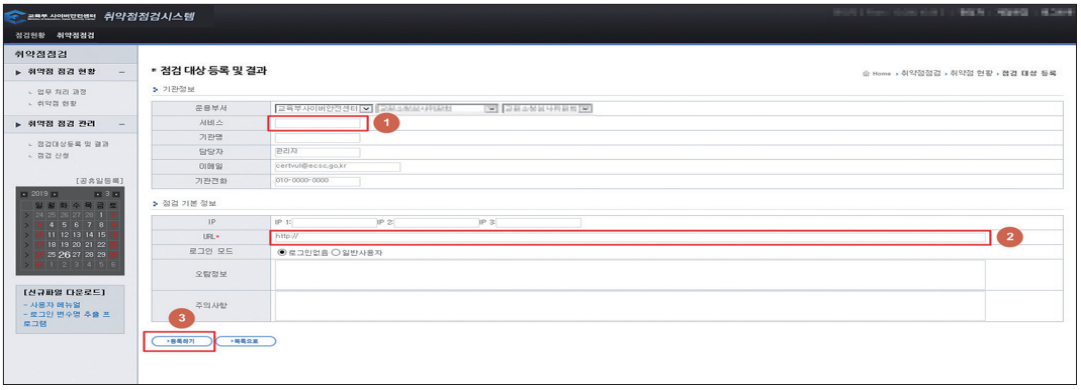
- 점검신청 현황 및 승인내역 확인

- 점검대상 등록 및 점검이력 확인

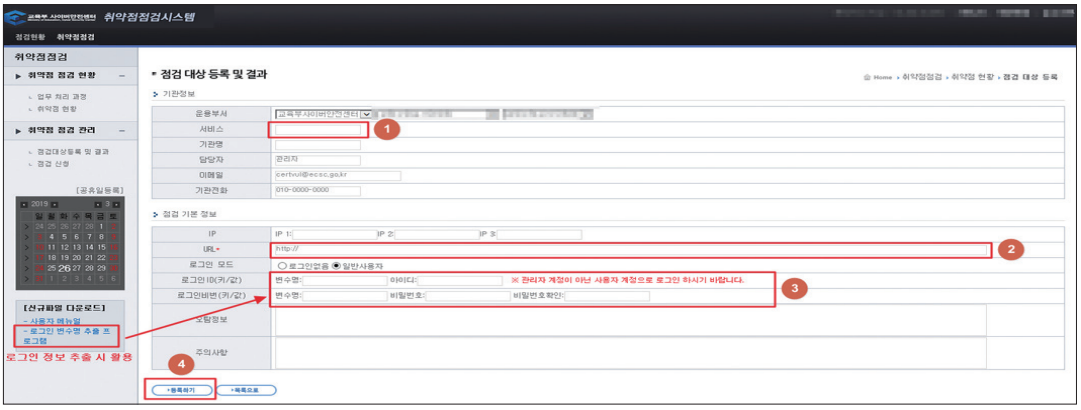
2.5 점검대상 등록 방법

- [취약점 점검] 〉 [점검대상등록 및 결과] 〉 [등록하기]

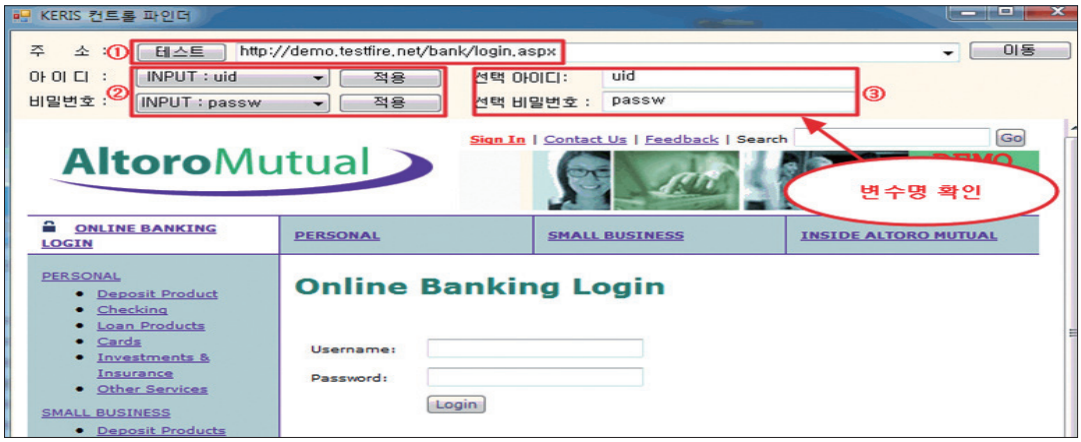
- [서비스 명 입력] 〉 [URL 입력] 〉 [등록하기]



- [서비스 명 입력] 〉 [URL 입력] 〉 [계정 정보 입력] 〉 [등록하기]
  ※ 계정 정보가 있을 시



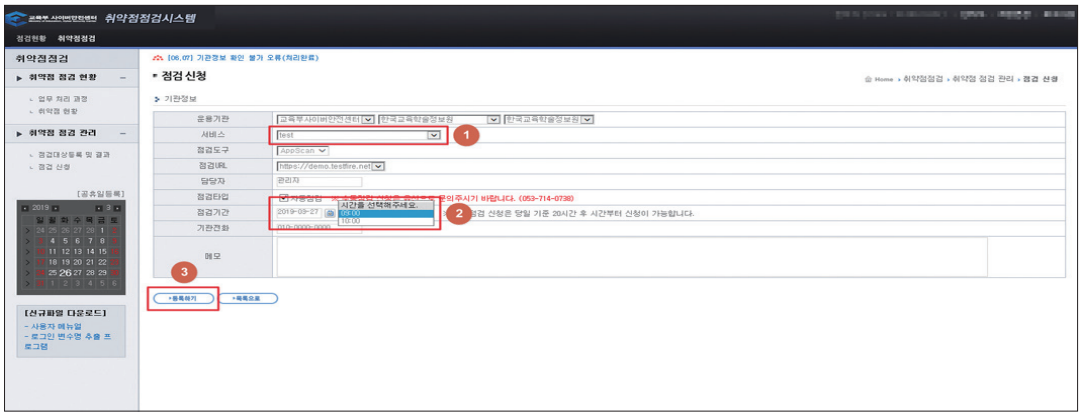
- 로그인 정보 추출 프로그램 활용법 [주소란에 대상 사이트 “로그인 페이지” 입력 후 이동] 〉 [적용된 “변수명” 확인]



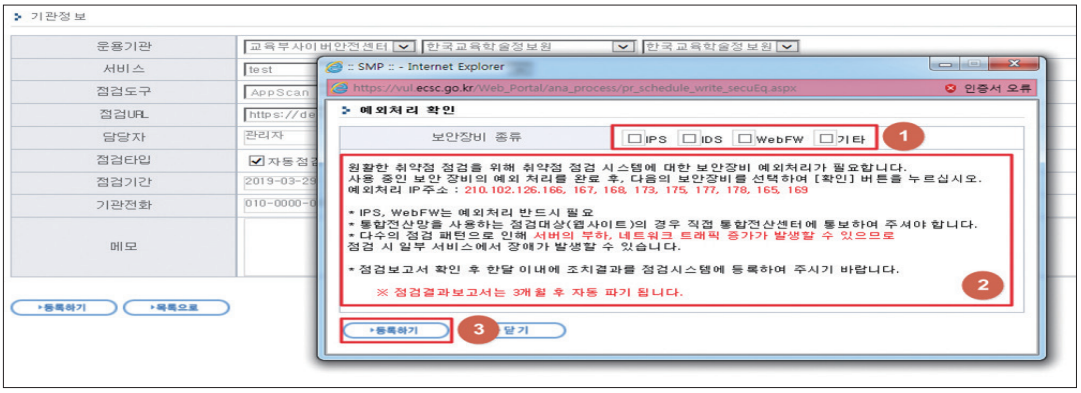
2.6 진단 신청 방법

- [취약점 점검] 〉 [점검 신청] 〉 [등록하기]

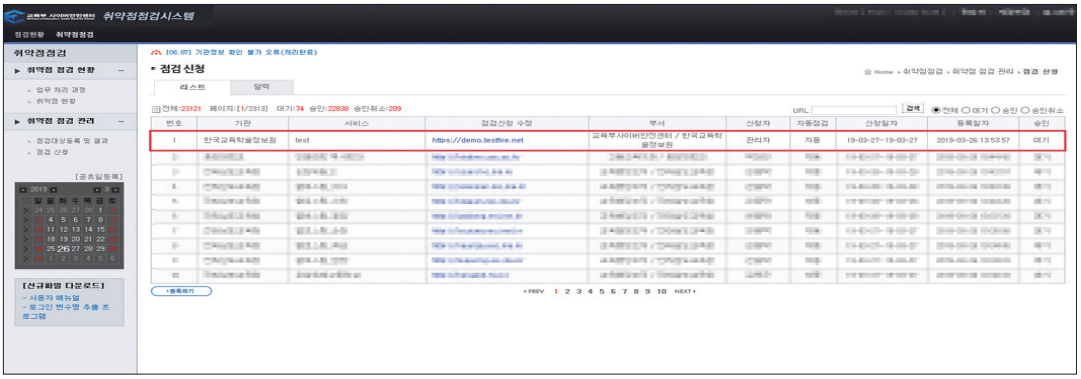
- [서비스 선택] 〉 [점검기간 선택] 〉 [등록하기]



- [예외처리 확인] 〉 [보안장비 종류 선택] 〉 [주의사항 확인] 〉 [등록하기]

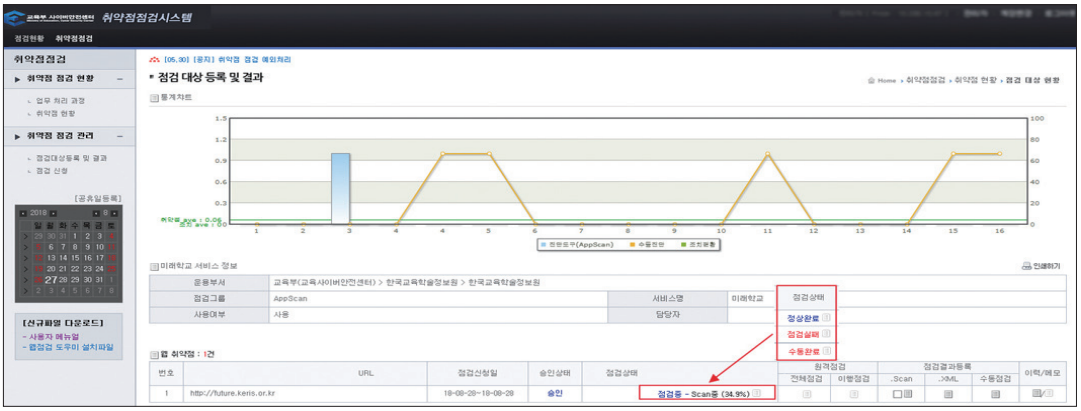


- 점검신청 리스트에서 정상 등록 여부 확인



2.7 점검 진행 상태 여부 확인

- 점검중 - 현재 정상적으로 취약점 점검이 진행 중인 상태
- 정상완료 - 정상적으로 취약점 점검을 완료한 상태
- 점검실패 - 취약점 점검이 실패한 상태(해당 사유는 전화로 확인 가능)
- 수동완료 - 정상적으로 취약점 점검은 완료하였으나 취약점이 발견되지 않은 상태(취약점 미발견으로 보고서 생성되지 않음)



2.8 웹 취약점 자가점검시스템 이용 전 유의사항

- 반드시 점검신청 전 대상 사이트가 등록되어 있어야 함
- 한 기관 당 1일 최대 5개 사이트 점검 가능
  (긴급한 경우 개수와 상관없이 점검)
- 점검 시간은 일과시간(09시 ~ 18시)만 가능
- 등록된 대상 홈페이지의 정보 수정 및 삭제 필요 시 관리자에게 문의
- 점검신청 후 날짜 수정 및 삭제는 불가함으로 관리자에 요청
- 한 서버에 여러 도메인이 있을 경우 나눠서 신청
  → 한 서버에 있는 5개 도메인을 같은 날 진행 할 경우 서버 부하 등 문제가 발생 할 수 있음
- 예외처리가 정상적으로 되지 않았을 경우 진단 불가
  → 웹 방화벽 등 장비에서 차단 된 경우 예외처리 후 다시 신청

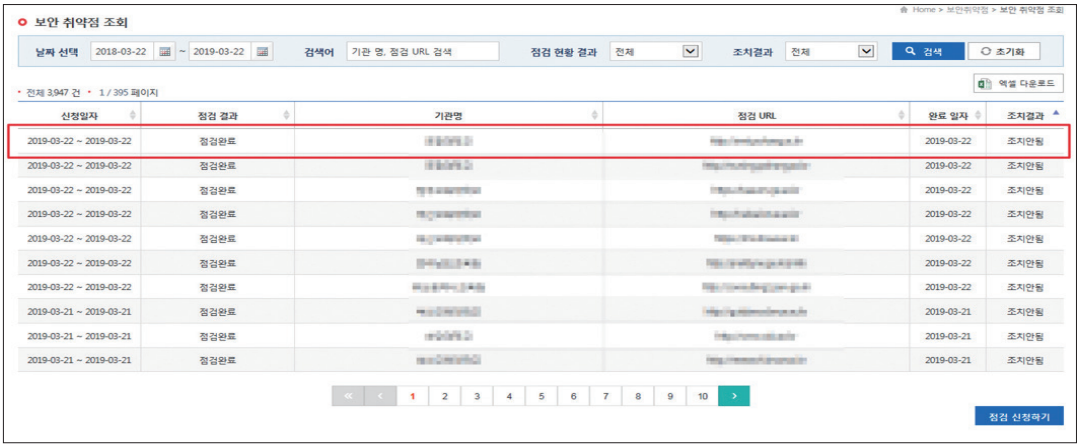
3. 점검 결과 확인

3.1 자동 점검 결과 확인 방법

- [정보공유시스템] 〉 [보안 취약점] 〉 [보안 취약점 조회]

- 점검 날짜, 기관 명, 점검 URL, 점검 현황 결과, 조치결과 방식으로 검색 후 결과 리스트 확인

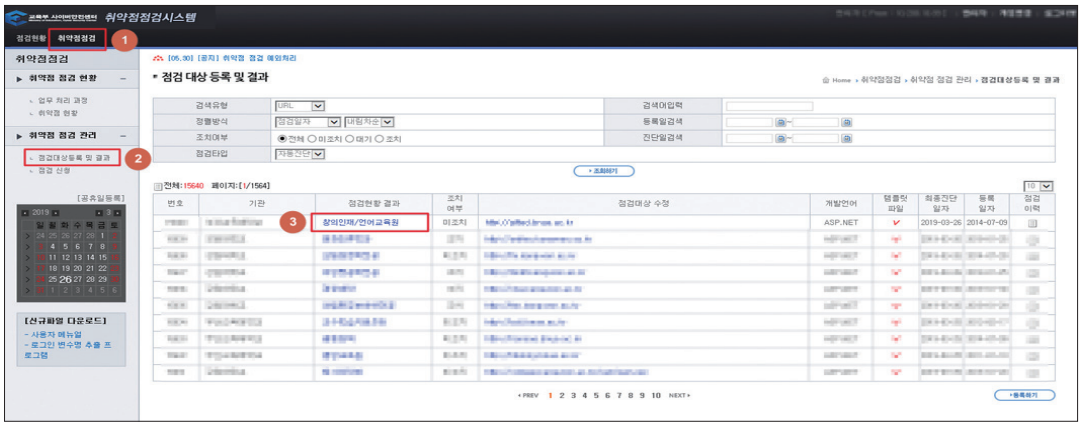
- 점검 완료된 홈페이지의 경우 리스트에서 확인 가능하며, 해당 항목 클릭 시 상세 페이지로 이동



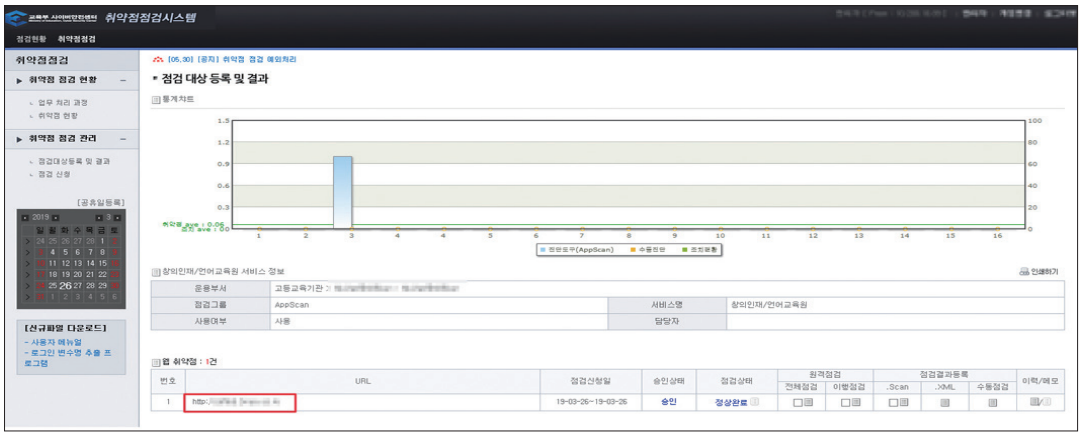
- 상세 페이지에서 점검 결과 이력 확인 및 보고서 다운로드

3.2 수동 점검 결과 확인 방법

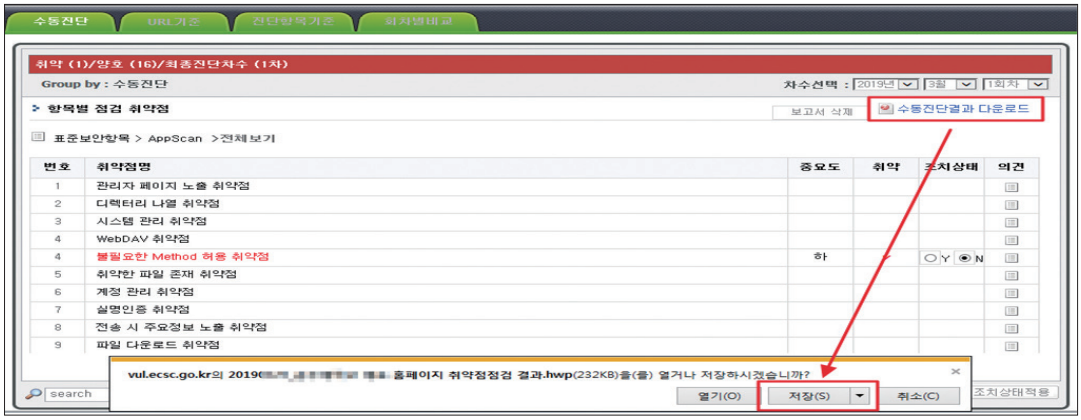
- [취약점점검시스템] 〉 [취약점 점검] 〉 [점검대상등록 및 결과] 〉 [점검현황 결과]



- 상세 페이지에서 점검 URL 클릭

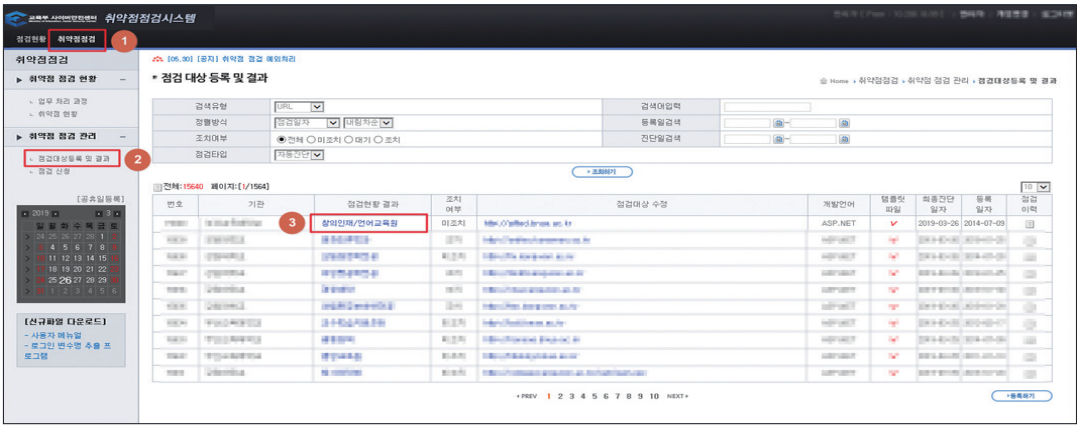


- [수동진단] 〉 [수동진단결과 다운로드] 〉 [저장]
  ※ 수동진단 - 교육부 사이버안전센터 취약점 점검팀 점검자가 직접 진단

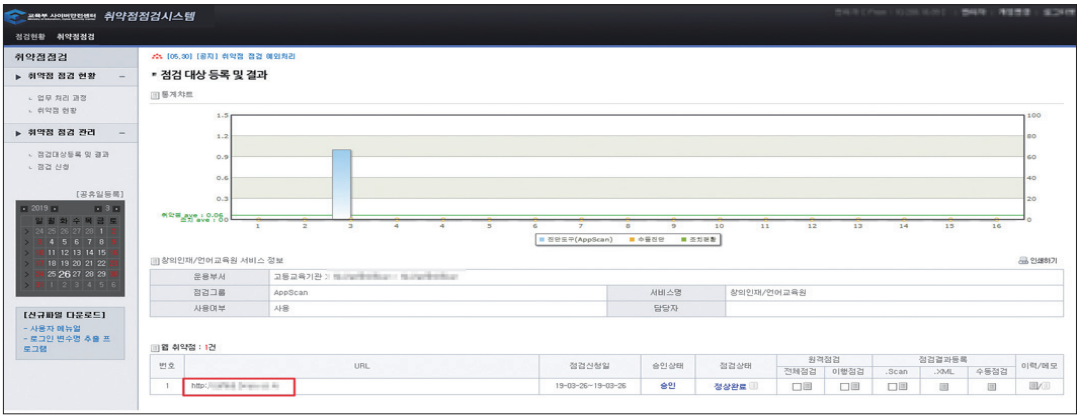


3.3 점검 결과 상세내역 확인 방법

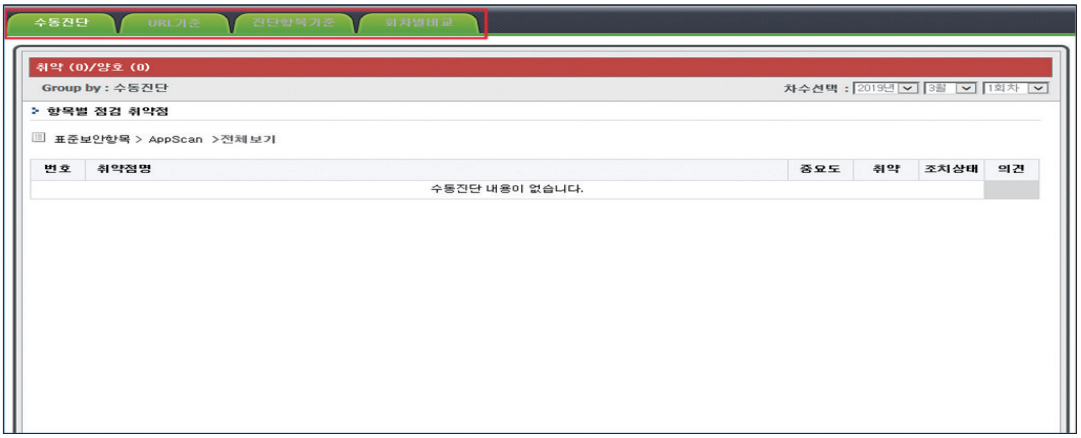
- [취약점점검시스템] 〉 [취약점 점검] 〉 [점검대상등록 및 결과] 〉 [점검현황 결과]



- 상세 페이지에서 점검 URL 클릭

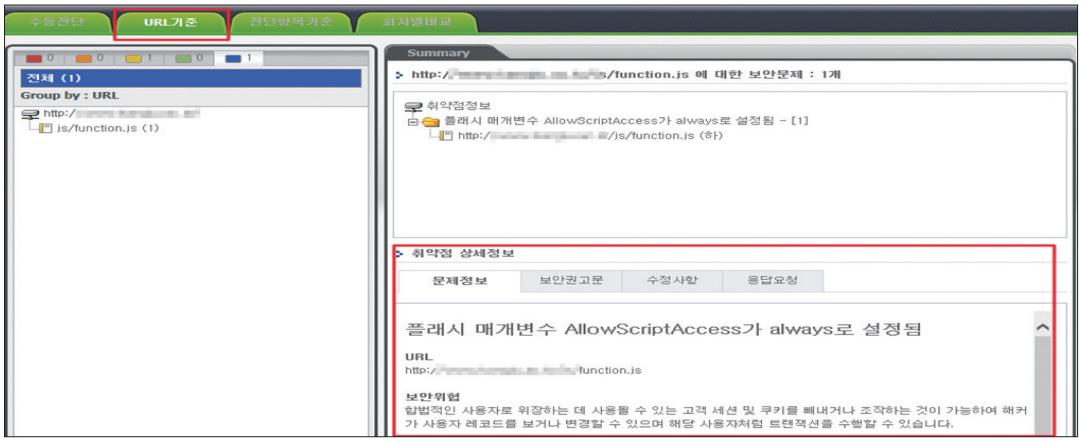


- 점검 결과는 수동진단, URL기준, 진단항목기준, 회차별비교로 구성

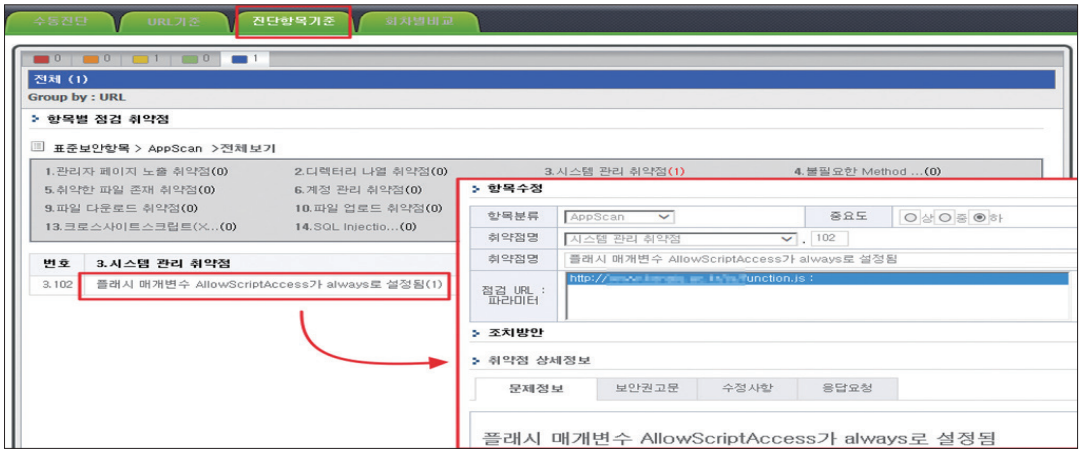


- [수동진단 결과 조회] 수동진단 결과를 발견된 취약점 항목별로 확인 가능하며, 수동진단결과 파일 다운로드 가능

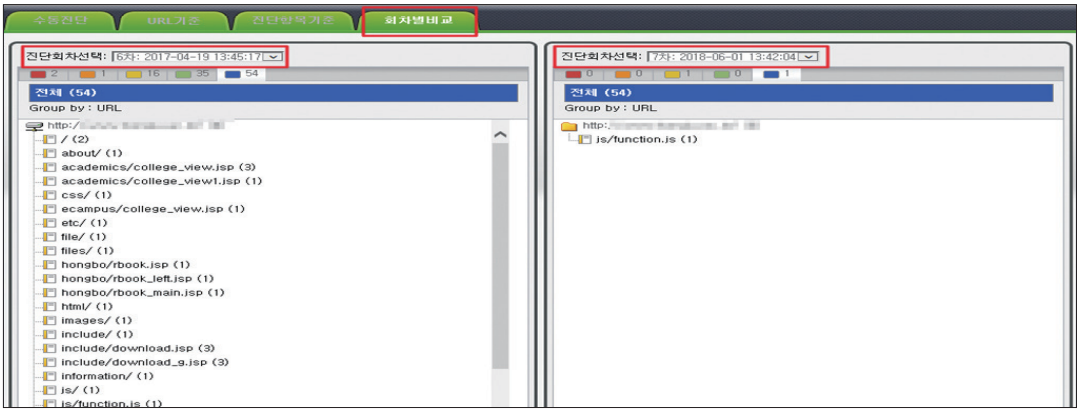
- [URL 기준 조회] 취약점이 존재하는 URL 별로 취약점 상세정보(문제정보, 보안권고문, 수정사항, 응답요청) 확인 가능



- [진단항목기준 조회] 발견된 취약점 항목별로 상세정보 확인 가능



- [회차별비교 조회] 지난 점검 회차와 비교 조회 가능

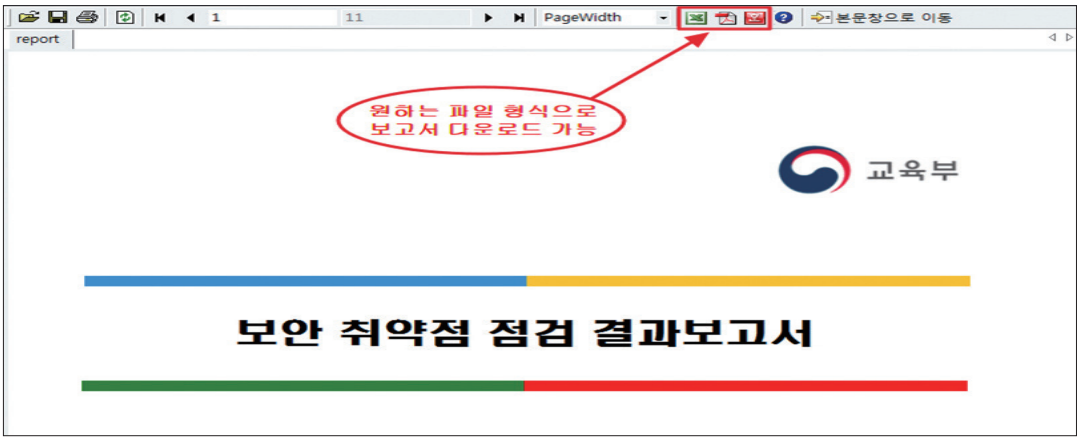


3.4 자가점검 결과 보고서 다운로드 및 상세 내역

- 개발자 보고서 : 취약점 조치를 위한 개발자용 보고서
- 요약 보고서 : 기관 관리자 및 운영자를 위한 요약 보고서

3.5 요약 보고서 상세 내역

- 요약 보고서를 원하는 형식의 파일로 선택하여 다운로드 가능



- 요약 보고서는 16개 항목으로 구분되며 취약점 명, 상세내용, 취약점 경로, 증빙화면, 조치방안으로 구성

홈페이지	http://[illegible]
취약점	시스템 관리 취약점
상세내용	○ 응용프로그램 설치 중에 생성되는 설치 파일 및 임시 파일이 존재하거나 시스템 상 설정 미비로 인해, 응용프로그램의 취약점 정보를 수집할 수 있음
취약점 경로	http://[illegible] http://[illegible]
증빙화면	Robots.txt 파일 웹 사이트 구조 노출 테스트 요청에서 AppScan이 변경한 사항 AppScan에서 이 문제를 보고한 이유 테스트 응답
조치방안	○ 응용프로그램 설치 시 생성되는 파일을 삭제 또는 권한을 변경하여 사용자 접속을 차단한다. ○ 웹 서버 소스 상에 설정이 잘못된 것은 없는지, 시스템 보안 설정이 미비한지 점검하여 해당 시스템에 가장 알맞게 설정 한다. ○ 참고자료 : 1. 교육부, 「웹 서버 및 홈페이지 취약점 점검가이드」 19페이지 참고

홈페이지	http://[illegible]
취약점	취약한 파일 존재 취약점
상세내용	○ 웹서버 개발 시 개발 · 보수 등의 이유로 임시 페이지 및 불필요한 파일 (백업파일, 로그파일 등) 존재로 인해 시스템 정보가 노출 되어 공격에 이용
취약점 경로	http://[illegible] http://[illegible]
증빙화면	애플리케이션 테스트 스크립트 발견 테스트 요청에서 AppScan이 변경한 사항 AppScan에서 이 문제를 보고한 이유 테스트 요청 / 테스트 응답
조치방안	○ 불필요한 임시 파일을 삭제 및 테스트 목적의 임시 파일을 생성하지 않도록 한다. ○ 삭제가 불가피할 경우, 임의의 사용자 또는 일반 사용자가 해당 정보를 열람하지 못하도록 권한을 설정한다. ○ 참고자료 : 1. 교육부, 「웹 서버 및 홈페이지 취약점 점검가이드」 28페이지 참고

3.6 개발자 보고서 상세 내역

- 개발자 보고서는 4단계로 구분하여 제공

개요	심각도, 유형별, 취약/비 취약 URL 비교 등의 통계 정보 제공
↓	
상세 요약	심각도별 취약점 항목, 취약한 URL 정보 제공
↓	
취약점 정보	보안 권고문과 수정안, 테스트 자료 및 증빙화면 제공
↓	
조치 방안	발견된 취약점 별 조치 방안 제공

- 상세 요약에는 취약점 이름, 보안 위협의 심각도에 따른 발견된 취약점 수량 제공

상세 요약

↓

취약점 정보

↓

조치 방안

상세 요약

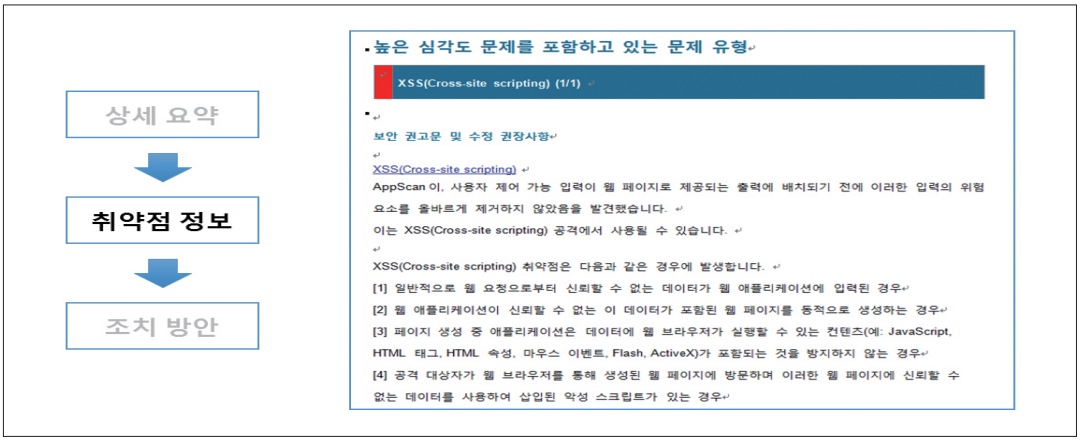
- 높은 심각도 문제

문제 유형	문제(모든 심각도)
XSS(Cross-site scripting)	10

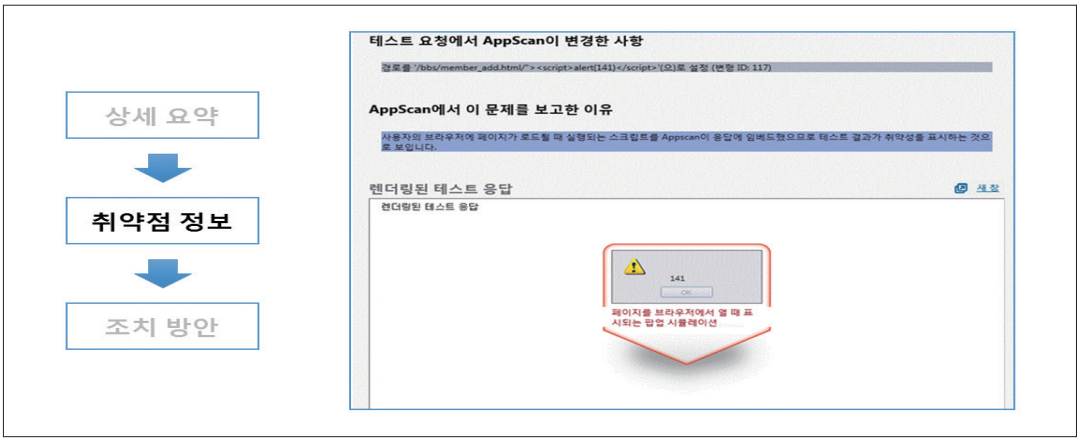
- 중간 심각도 문제

문제 유형	문제(모든 심각도)
Microsoft Windows MHTML XSS(Cross-site scripting)	3
디렉토리 목록화	2

- 취약점 정보는 취약점 항목에 대한 상세 취약성 정보 페이지를 조회할 수 있음



- 취약점 점검 시 테스트 응답에 대한 결과 증빙 화면 제공



- 변형 요청 반응을 통해 취약점 존재 유무 확인

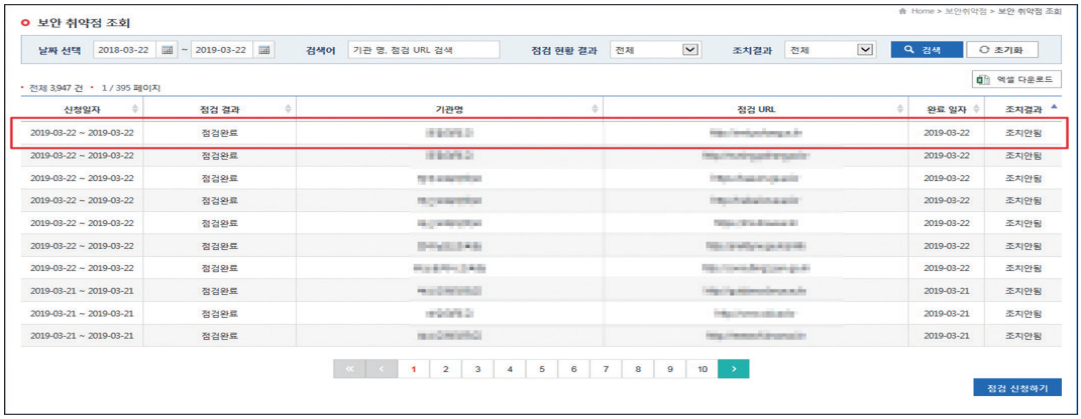
- 발견된 취약점에 대한 조치 가이드가 제공되므로 조치 시 활용

4. 조치 결과 등록

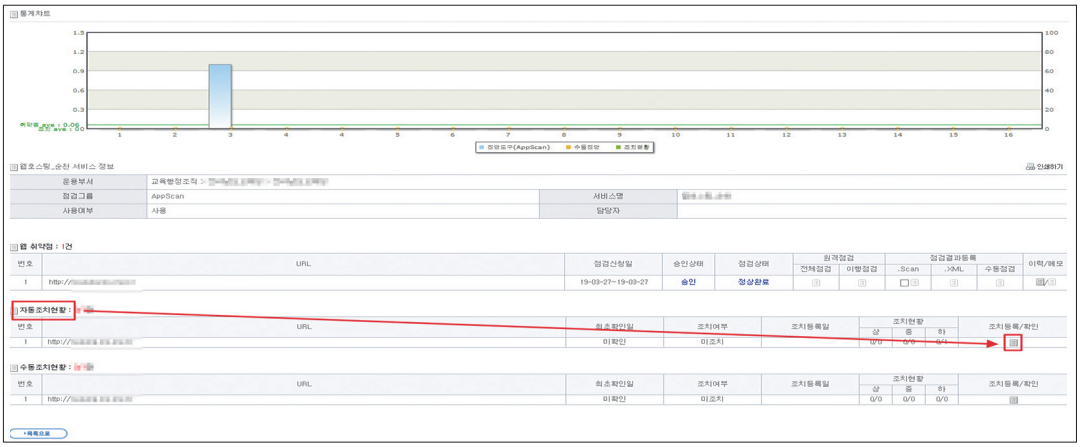
4.1 자동점검 조치 결과 등록

- [정보공유시스템] 〉 [보안 취약점] 〉 [보안 취약점 조회]

- 점검 현황 결과 [점검완료] & 조치결과 [조치안됨] 검색 후 결과 항목 선택

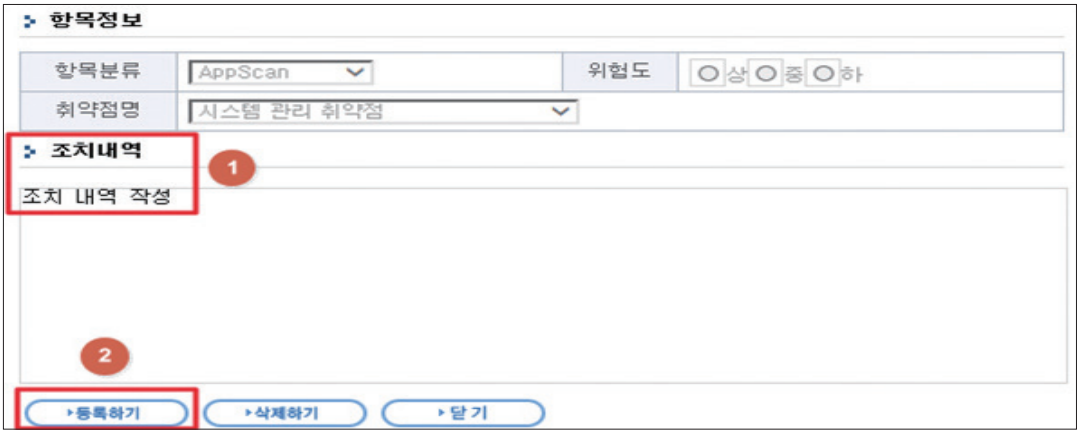


- [자동조치현황] 〉 [조치등록/확인]



- [취약점 명] 〉 [의견]

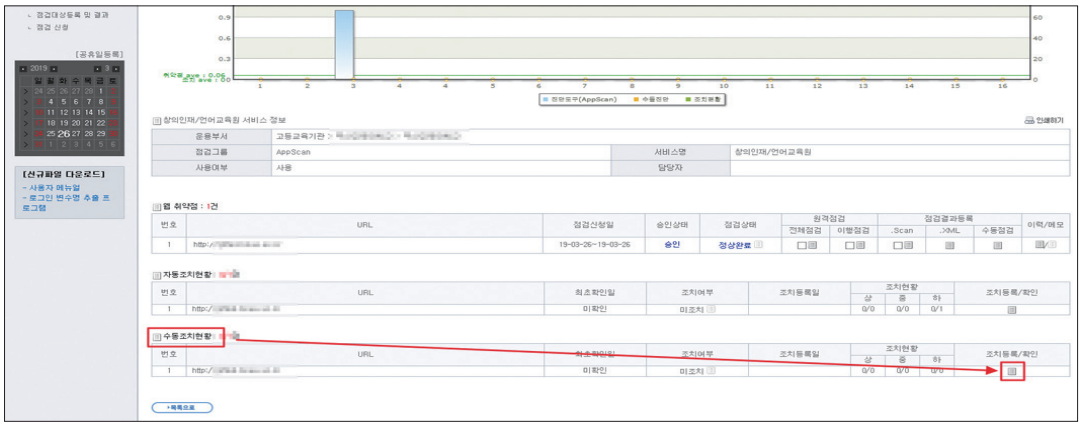
- [의견란에 조치 내역 작성] 〉 [등록하기]



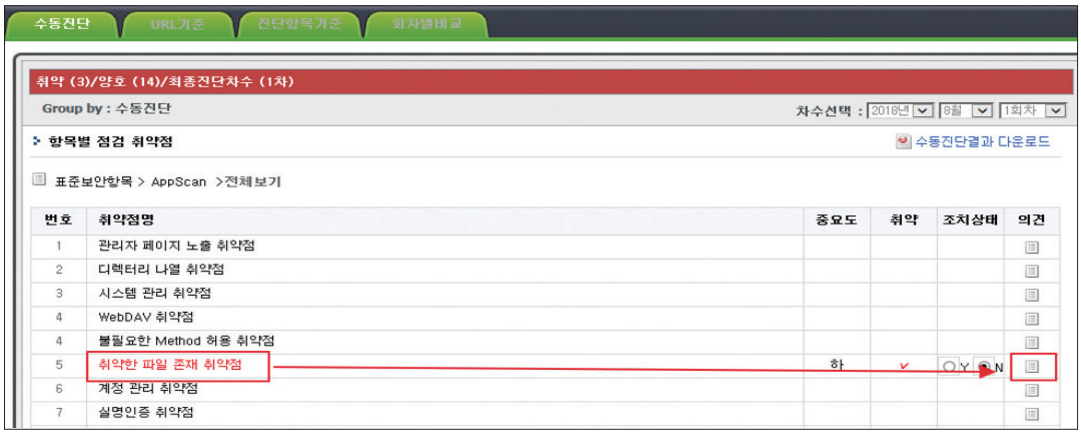
4.2 수동점검 조치 결과 등록

- [취약점 점검] 〉 [점검대상등록 및 결과] 〉 [점검현황 결과]

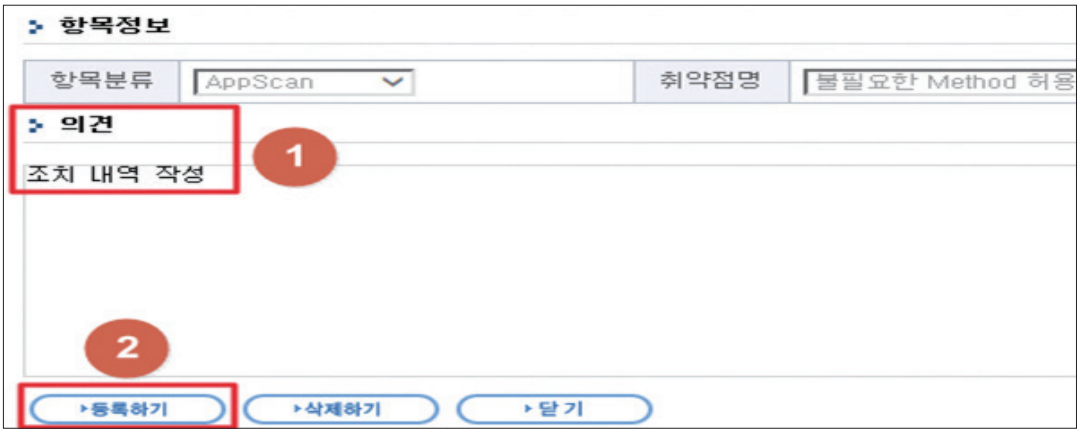
- [수동조치현황] 〉 [조치등록/확인]



- [취약점 명] 〉 [의견]



- [의견란에 조치 내역 작성] 〉 [등록하기]



참 고 자 료

[국내]

- 행정자치부 · 한국인터넷진흥원, 「전자정부 SW개발 · 운영자를 위한 소프트웨어 개발보안 가이드」, 2017
- 한국인터넷진흥원, 「홈페이지 취약점 진단 · 제거 가이드」, 2013
- 국가사이버안전센터·국가보안기술연구소, 「홈페이지 취약점 점검 매뉴얼」, 2012
- 과학기술정보통신부 · 한국인터넷진흥원,「주요정보통신기반시설 기술적 취약점 분석 · 평가 방법 상세가이드」, 2017

[국외]

- 아이콘의 출처, "All icons made by Icon8(https://icons8.com/)"

교육기관 홈페이지 취약점 심층점검 가이드

2019년 8월 인쇄
2019년 8월 발행
발행처 : 교육부 • 한국교육학술정보원
세종특별자치시 갈매로 408
정부세종청사
대구광역시 동구 동내로 64
한국교육학술정보원
Tel: (053) 714-0777

(비매품)

교육부 사이버안전센터
Ministry of Education, Cyber Security Center