답안지

과정명	오픈소스 기반 보안 취약점 분석 실무자 양성	담당교사	홍제준	월차	
과목명	SW개발 보안구축	훈련생 이름	임서규	평가일자	
평가 방법	문제해결 시나리오				

답안

SW보안설계 구현 계획 및 테스트

-목차-

1. Server Platform

현재 WebHack사이트가 이용하고 있는 서버 플랫폼에 대한 설명입니다.

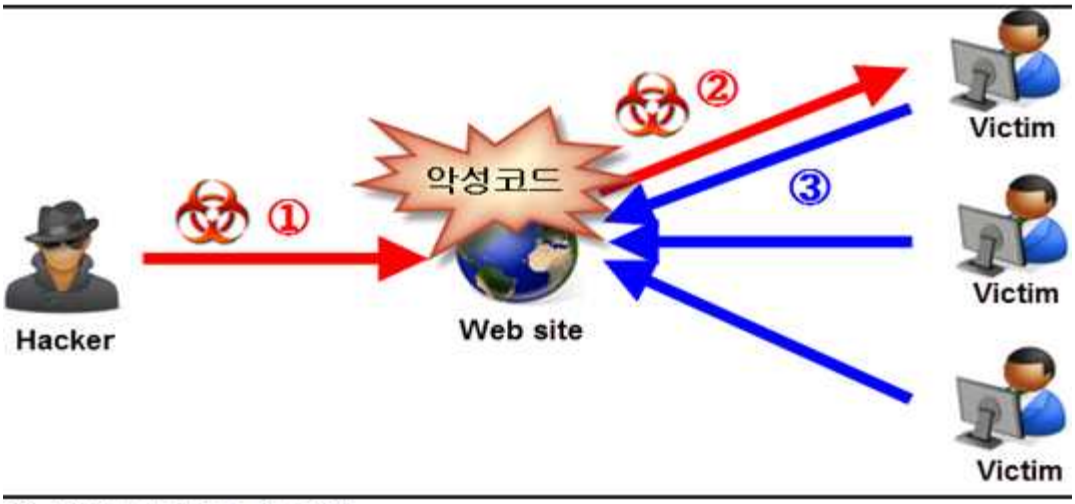
SQL Server 2000 - Copyright(c) 1988-2000 Microsoft Corporation Standard Edition on Windows NT 5.0 (Build 2195: service pack 4)

빌드 번호 또는 버전	버전 설명
8.00.194 (Intel X86) (Build 2195: service pack 4)	SQL Server 2000 Analysis Services RTM

Microsoft® SQL Server™ 또는 MS_SQL은 마이크로소프트 사에서 개발한 상업용 데이터베이스로서 대용량의 데이터 집합을 체계적으로 구성해 놓은 시스템입니다. 사용자 또는 시스템이 상호 공유가 가능하고 사용할 수 있게 되는 데이터베이스를 관리해 주는 시스템인 DBMS(Database Management System)입니다. 오라클 MY_SQL과는 다르게 window server에서만 구동이 되며 C#과는 높은 호환성을 자랑하는 DBMS입니다. 비공개 소스이므로 폐쇄적인 정책 특징을 가지고 있으며 비교적 중소기업에서 주로 이용하는 DBMS입니다.

2. XSS / CSRF

▶ XSS (Cross—Site Scripting)

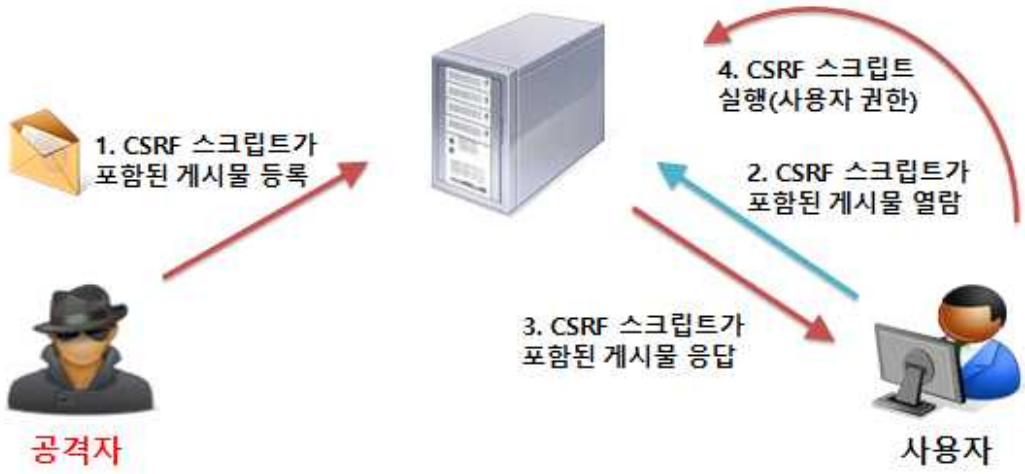


① **악성 스크립트 삽입**
② **악성 스크립트에 의한** Action
③ **게시물 접근**

SQL injection과 함께 웹 상에서 가장 기초적인 취약점 공격 방법의 일종으로, 악의적인 사용자가 공격하려는 사이트에 스크립트를 넣는 기법을 말합니다. 공격에 성공하면 사이트에 접속한 사용자(client)는 삽입된 코드를 실행하게 되며, 보통 의도치 않은 행동을 수행시키거나 쿠키나 세션 토큰 등의 민감한 정보를 탈취합니다. 공격대상은 서버가 아닌 클라이언트입니다.
크로스 사이트 스크립팅 이란 이름답게, 자바스크립트를 사용하여 공격하는 경우가 많습니다. 공격 방법이 단순하고 가장 기초적이지만, 많은 웹 사이트들이 XSS에 대한 방어 조치를 해두지 않아 공격을 받는 경우가 많습니다. 여러 사용자가 접근 가능한 게시판 등에 코드를 삽입하는 경우도 많으며, 경우에 따라서는 메일과 같은 매체를 통해서도 전파됩니다.
주로 CSRF를 하기 위해서 사용되기 때문에 종종 CSRF와 혼동되는 경우가 있으나, XSS는 자바스크립트를 실행시키는 것이고, CSRF는 특정한 행동을 시키는 것이므로 다릅니다.

■ 파급 효과
* 쿠키 / 세션 ID 정보 탈취
* 시스템 관리자 권한 탈취
* 악성 코드 다운로드

▶ CSRF (Cross-Site Request Forgery)



사이트 간 요청 위조 또는 크로스 사이트 요청 위조를 의미하며 XSS(사이트 간 스크립팅)를 이용한 공격이 사용자가 특정 웹 사이트를 신용하는 점을 이용한 것이라면, CSRF는 특정 웹 사이트가 사용자의 웹 브라우저를 신용하는 상태를 노린 것 입니다. 사용자가 자신의 의지와 무관하게 공격자가 의도한 행동을 하여 특정 웹페이지를 보안에 취약하게 한다거나 수정, 삭제 등의 작업을 하게 만드는 공격방법을 의미합니다. 2008년에 발생한 옥션의 개인정보 유출사건에서도 관리자 계정을 탈취하는데 이방법이 사용되었으며 공격의 난이도가 높지 않아 널리 사용되는 방법 중 하나입니다.

■ 파급 효과

* 사용자 권한 도용
* 사용자 정보 변경

3. XSS / CSRF 취약점 발생원인 및 시현

1) XSS

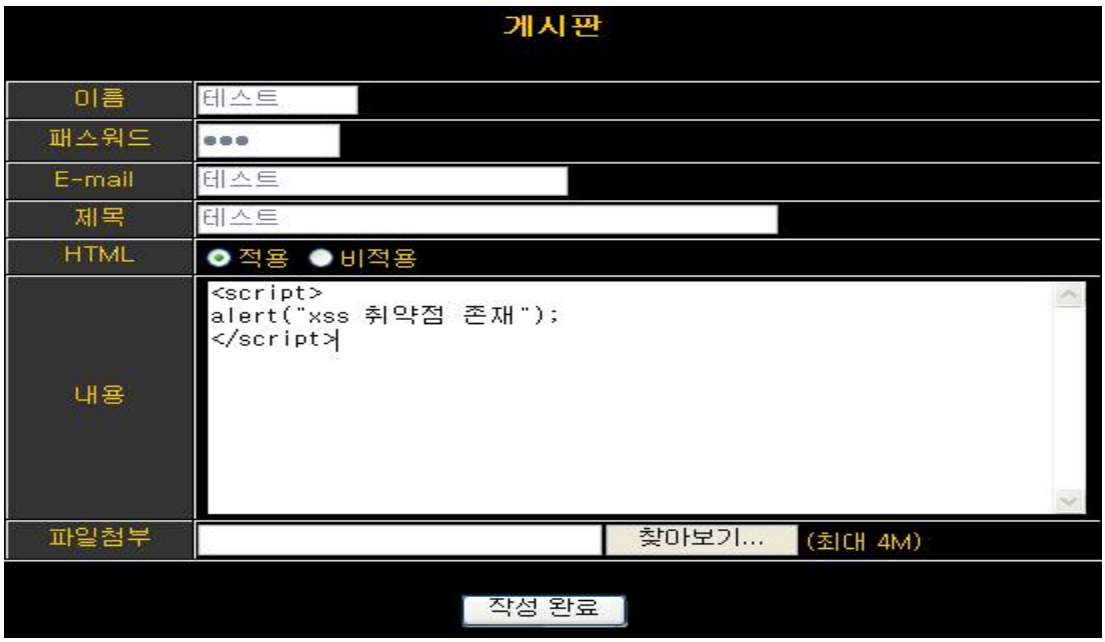
※취약점 발생원인

페이지에 악의적인 스크립트를 포함시켜 웹 페이지를 열람하는 접속자의 권한으로 부적절한 스크립트가 수행되어 정보유출 등의 공격을 유발할 수 있는 취약점이 있습니다. 열람하는 경로는 다양하지만 대표적인 사례로는 게시판이 있겠습니다.

※취약점 사례

현재 Guest Board에 XSS 취약점이 있다고 보고되어 있으며, 이에 따라 글쓰기 기능이 있는 게시판의 HTML편집기 기능을 이용하여 스크립트 코드의 삽입이 가능한지 테스트합니다.

#취약점 시현



경고 창을 띄우는 스크립트 구문을 작성하여 게시판에 글을 등록합니다. 작성완료 후에 경고창이 뜰 경우 XSS에 대한 취약점 있다고 판단합니다. 다음은 작성 완료 후의 모습입니다.

글을 게시한 후에 스크립트 태그 사용에 대한 "오류"나 "경고" 메시지가 발생하며 입력한 정보가 등록되지 않는다면 취약점이 없는 것으로 판단됩니다.

내용보기

작성완료를 끝내자 바로 경고 창이 뜨는 것을 확인 하였습니다. 이 경고 창을 확인함으로서 게시판의 글쓰기 내용 기능에 스크립트 구문이 작동 한다는 것을 확인하게 됩니다.
스크립트기능을 수행 할 수 있다면 사용자 쿠키 값을 획득하여 인증우회에 사용하거나, 악성코드 삽입이나 페이지에 링크를 삽입하여 악성프로그램을 전송할 수 있습니다.
이러한 스크립트가 실행되는 것을 보안하려면 스크립트 기능 자체를 억제하기위해 해당 페이지의 소스를 수정해야 할 것입니다.
다음은 CSRF에 대한 취약점입니다.

2) CSRF

※취약점 발생원인

1. 개별 링크 폼이 사용자 별로 예측 가능한 토큰을 사용할 경우.
예측 불가능한 토큰이 있다면 공격자는 요청메시지를 변조 할 수 없고, 예측 가능한 토큰이 있다면 요청 메시지를 변조할 수 있습니다.

2. 인증이나 세션, 쿠키 등 모든 웹 사이트에서 인증 된 사용자가 보내는 데이터는 정상적인 경로를 통한 파라미터 요청으로 판단.
정상적인 요청과 비정상적인 요청을 구분하지 못합니다.

※취약점 사례

공격자가 올린 게시 글을 열람하였을 때 피해자의 정보가 수정될 수 잇도록 하거나 관리자의 권한을 얻도록 스크립트를 작성하는 것이 대표적인 사례입니다. 또는 사회 공학적 해킹기법을 이용하여 form형태로 서버에 전송하는 공격이 있으며 사용자가 클릭을 한번 해주어야 되는 특징을 가진 one-click 형태의 기법입니다. 다른 유형으로는 zero-click형태로서 클릭을 하지 않아도 되지만 GET방식으로 요청이 되는 경우 사용할 수 있습니다.여기서 GET 메소드 방식이라는 것은 XSS 즉, 스크립트가 실행이 되는 경우에 해당됩니다. 이두가지 방식을 활용하여 client의 계정을 통해 서버로 공격이 가능해집니다.
현재 보고된 대로 Change Information 페이지에서 CSRF에 취약하다는 내용을 받았으며 이에 따라 취약점을 시현하겠습니다.

#취약점 시현

앞서 했던 취약점 분석에 있어 XSS에 대한 취약점이 있던 것을 확인 하였습니다.
따라서 GET메소드 방식으로 URL변조를 통하여 회원정보를 수정할 수 있습니다.
현재 로그인 되어 있는 계정의 정보는 아래와 같습니다.

닉네임 : sukyu0919
이메일 : sukyu0919@hanmail.net

다음 시나리오에 따라 아래와 같은 내용으로 회원 정보를 수정합니다.
닉네임 : hack
이메일 : sukyu0919@hack.net

```
<html>
<head>
<title>회원 정보 변경</title>
<meta http-equiv="Content-Type" content="text/vbscript; charset=euc-kr">
</head>

<body>

sukyu0919님의 회원 정보 변경 페이지 입니다.<p>

<form method="get" action="member_modify_ok.asp">
    <input type="hidden" name="exec" value="modify">
    닉네임 : <input type="text" disable="disable" name="nickname" value=sukyu0919><br>
    이메일 : <input type="text" disable="disable" name="email" value=sukyu0919@hanmail.net><br>
    <input type="submit" value="변경하기">

</form>
```

먼저 Change Information 페이지의 소스인 member_modify.asp이며 CSS(Client Side Script)입니다.

위 소스를 보면 회원 정보를 변경 할 수 있는 파라미터를 유추 할 수 있습니다.

메소드 : get

회원정보변경 페이지 경로 : member_modify_ok.asp

닉네임 정보 파라미터 : nickname 값: sukyu0919

이메일 정보 파라미터 : email 값: sukyu0919@hanmail.net

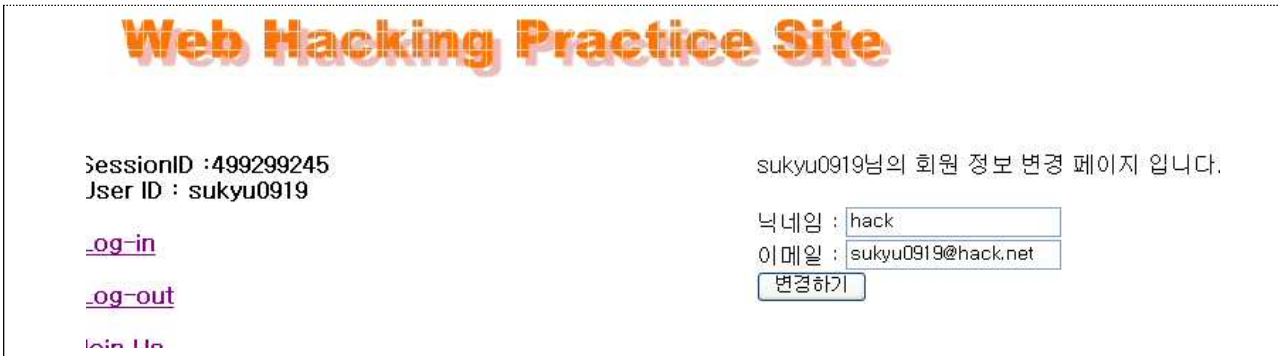
위에 파라미터 명칭에 따라 value(값)을 수정하여 위 그림처럼 URL주소를 변조 후 적용합니다.

http://172.168.189.132/member/member_modify_ok.asp?exec=modify&nickname=hack&email=sukyu0919@hack.net

nickname = hack

email = sukyu0919@hack.net

회원 정보가 변경이 되었다고 창이 뜹니다.



다시 홈페이지로 돌아가면 닉네임과 이메일이 변경된 것을 알 수 있습니다.
다음은 XSS와 CSRF 취약점에 대한 해결 방안 및 구현이 있겠습니다.

5. XSS 취약점 해결방안 및 구현

XSS 취약점을 근본적으로 제거하기 위해서는 스크립트 등 해킹에 사용될 수 있는 코딩에 사용되는 입력 및 출력 값에 대해서 검증하고 무효화시켜야 합니다. 입력 값에 대한 유효성 검사는 데이터가 입력되기 전에 가능하면, 입력 데이터에 대한 길이, 문자, 형식 및 사업적 규칙 유효성을 검사해야 합니다.

< XSS 취약점 해결방안 >

1. replace를 이용한 특수문자 치환 기법

XSS 공격은 기본적으로 script 태그를 사용하기 때문에 XSS 공격을 차단하기 위해 태그 문자(<, >) 등 위험한 문자 입력 시 문자 참조(HTML entity)로 필터링하고, 서버에서 브라우저로 전송 시 문자를 인코딩하는 것입니다. HTML 문자 참조란 ASCII 문자를 동일한 의미의 HTML 문자로 변경하는 과정입니다. 예를 들어, 문자 “<”는 동일한 의미의 HTML “<”로 변경한다는 것입니다. HTML 엔터티는 대부분의 인터프리터(브라우저)에서 특수한 의미를 가지지 않으며, 단순한 문자로 처리됩니다. 이렇게 인코딩하면 사용자는 <script>가 <script>로 보이지만 HTML 문서에서는 <script>로 나타나서 브라우저에서 일반 문자로 인식하고 스크립트로 해석되어 실행되지는 않습니다.

ASCII 문자	참조 문자	ASCII 문자	참조 문자
&	&amp;	"	&quot;
<	&lt;	'	&#x27;
>	&gt;	/	&#x2F;
(	&#40;	)	&#41;

[그림 13] 위험 문자 인코딩

위의 표는 HTML 문서에서 악성 스크립트에 포함되어 브라우저에서 실행될 수 있는 문자와 대체 문자를 정리한 것입니다. 악성 스크립트는 많은 HTML 태그 안에 포함을 할 수 있으므로 반드시 표에 있는 위험 문자의 경우 출력 값을 이스케이핑 해야 합니다.

2. 입력값 제한
사용자의 입력값을 제한하여 스크립트를 삽입하지 못하도록 해야합니다.
예를 들어 http://172.168.189.132/board?page=1 과 같은 URL에서 사용자의 입력값이 page가 화면에 바로 뿌려지는데 이때 page의 값을 숫자만으로 허용한다면 그 외의 입력값에는 출력되지 않기 때문에 대응이 가능해 집니다.
또는 게시판 작성 란의 이름, 제목 등 스크립트가 들어갈 수 있는 부분에도 필드의 문자열 길이 값을 조정하여 방지 할 수 있습니다.

3. 보안 라이브러리 사용
AntiXSS : MS에서 개발한 XSS 예방 라이브러리
OWASP ESAPI : OWASP에서 오픈소스로 개발한 라이브러리
라이브러리를 사용하여 직접적으로 xss를 예방합니다.

4. web.xml 설정
JAVA기반 웹 어플리케이션의 경우 web.xml에 필터를 선언하여 모든 파라미터가 해당 필터를 거치도록 할 수 있습니다. 서버나 DB에 저장된 데이터를 사용자의 웹페이지에 뿌려줄 때에도 필터링 합니다.

다음에서 위의 4가지 취약점 대응 기법 중 1번의 해당사항을 구현 하겠습니다.

1. replace를 이용한 특수문자 치환 기법

Server side에서 필터링 코드를 추가하여 보안하는 방법입니다.

이것은 게시판에서 글쓰기 작성 란에 스크립트가 실행되는 것을 방지하기 위함입니다.

먼저 스크립트를 작성하여 게시판에 글이 올라오는지 확인합니다. 경고 창이 뜰 경우 스크립트가 실행이 된 것을 확인 할 수 있습니다.

내용보기

스크립트가 실행이 되는 것을 확인합니다.

board_write[1] - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
<html>
<head>
<title>글 작성</title>
<meta http-equiv="Content-Type" content="text/html; charset=ks_c_5601-1987">


<script language=javascript>
<!--
function write_ok()
{
                if(document.write_form.name.value == "")
                {
                        alert("이름을 입력해주세요.~!")
                        document.write_form.name.focus();
                        return;
                }
                if(document.write_form.password.value == "")
                {
                        alert("비밀번호를 입력해주세요.~!")
                        document.write_form.password1.focus();
                        return;
                }
                if(document.write_form.subject.value == "")
                {
                        alert("제목을 입력해주세요.~!")
                        document.write_form.subject.focus();
                        return;
                }
                if (document.write_form.content.value == "") {
                        alert("내용을 입력해주세요.~!");
                        document.write_form.content.focus();
                        return;
                }
                if (document.write_form.attachFile.value == "*.asp") {
                        alert("asp 파일은 업로드 할 수 없습니다.");
                        document.write_form.content.focus();
                        return;
                }

        document.write_form.action="board_write_ok.asp";
        document.write_form.submit();
}
```

Client side스크립트인 게시판 글 작성(board_write.asp)의 소스입니다.

submit() 하기 이전에, 내용을 입력받는 함수가 document.write_form.content.value임을 알 수 있으며 action을 취하게 되면 board_write_ok.asp로 넘어가는 것을 알 수 있습니다.이 소스를 수정하여 태그가 실행되지 않도록 할 수 있지만, Server side에서 수정을 해야 효율적으로 보안 할 수 있습니다.

windows server 2000으로 접속하여 board폴더에서 게시판에 관련된 asp파일을 볼 수 있습니다.

board_write_ok.asp - 메모장
파일(F) 편집(E) 서식(O) 도움말(H)

```
<!--#include file="../dbconn.inc"-->
<%
        Dim DirectoryPath
        DirectoryPath = Server.MapPath("upload") '파일이 저장될 로컬폴더 경로

        Dim abc, oFile
        Set abc = Server.CreateObject("ABCUpload4.XForm")

        Dim strName, strPassword, strEmail, strSubject, strContent, bTag

        s_Name          = abc.item("name")
        s_Password  = abc.item("password")
        s_Email    = abc.item("email")
        s_Subject       = abc.item("subject")
        s_Content       = abc.item("content")
        b_Tag            = abc.item("tag")

        s_Subject       = replace(s_Subject,"'","''")
        s_Content       = replace(s_Content,"'","''")

        abc.AbsolutePath = True
        Set oFile = abc("attachFile")(1)

Dim strFileName, FileSize, FileType, strFileWholePath
```

Server side스크립트인 board_write_ok.asp의 소스입니다.
위를 보시면 s_Content의 게시판 내용을 담는 함수를 볼 수 있습니다.

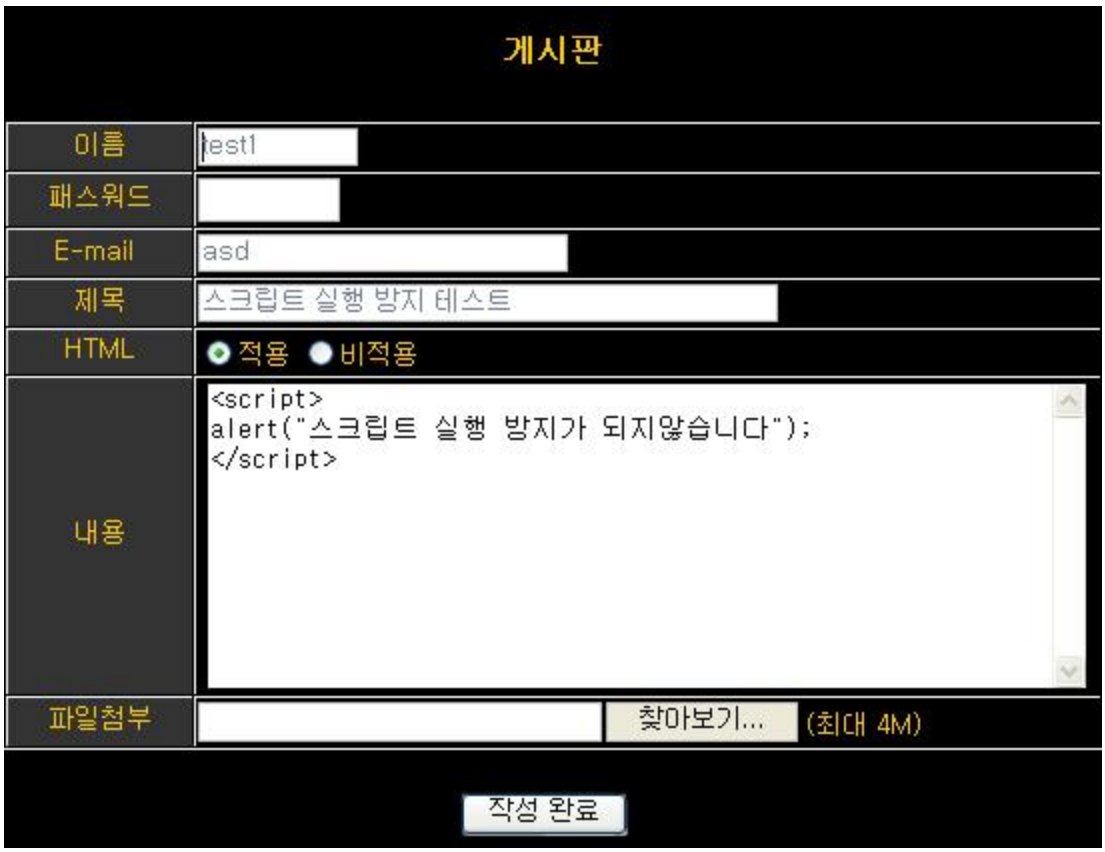
```
        Dim strName, strPassword, strEmail, strSubject, strContent, bTag

        s_Name          = abc.item("name")
        s_Password  = abc.item("password")
        s_Email    = abc.item("email")
        s_Subject       = abc.item("subject")
        s_Content       = abc.item("content")
        b_Tag            = abc.item("tag")

        s_Subject       = replace(s_Subject,"'","''")
        s_Content       = replace(s_Content,"'","''")
        s_Content       = replace(s_Content,"<","&lt;")
        s_Content       = replace(s_Content,">","&gt;")
```

replace 메소드를 이용하여 “<”, “>”와 같이 HTMl 태그 실행에 사용되는 문자열을 치환하여 공격자가 삽입한 악의적인 스크립트가 실행되지 않도록 설정합니다.

```
s_Subject       = replace(s_Subject,"'","''")
s_Content       = replace(s_Content,"'","''")
s_Content       = replace(s_Content,"<","&lt;")
s_Content       = replace(s_Content,">","&gt;")
s_Content       = replace(s_Content,"&lt;p&gt;","<p>")
```

부득이하게 사용해야할 HTMl 태그가 존재한다면 밑에 위와 같이 replace메소드를 사용하여 치환된 문자열을 다시 원래 태그로 치환함으로써 HTMl태그가 실행 가능하도록 설정하는 것을 권장합니다.
<p>를 사용해야한다면 위와 같이 다시 치환함으로서 사용할 수 있습니다.

설정을 끝내고 보안설정을 테스트하기 위해 WebHack 게시판으로 이동합니다.



코드 수정을 끝내고 다시 게시판에 접속하여 <script>를 입력합니다.
게시판 내용은 <script>구문이 들어갈 경우 방지가 되지 않는다는 내용의 경고창이 뜨도록 합니다.

내용보기

이름	test1	등록일	2018-08-28 오후 5:57:00
Email	asd	조회	1
제목	스크립트 실행 방지 테스트		
내용	<script> alert("스크립트 실행 방지가 되지않습니다"); </script>		
첨부			

삭제하기 목록으로

글 작성을 마치고 게시글을 열어보니 <script>가 실행되지 않고 필터링이 된 것을 확인 하였습니다.

```
    <td align="center" width="10%"><font size=2>내용</font></td>
      <td colspan="4" style="padding:15px;"><font size=2>&lt;script&gt;
alert("스크립트 실행 방지가 되지않습니다");
&lt;/script&gt;
```

위에서 앞서 설명 드렸듯이 소스 문서를 열어보니 <script>가 <script>로 치환된 것을 볼수 있습니다. 이러한 필터링 방식을 이용하여 스크립트 실행을 방지 할 수 있겠습니다.

6. CSRF 취약점 해결방안 및 구현

CSRF라고 불리우는 cross-site 요청 변조는 웹사이트에 악성 스크립트를 입력하고 해당 사이트를 방문하는 사용자가 스크립트에 의해 특정 행동으로 서버에게 요청을 하게 만드는 것입니다.

<CSRF 취약점 해결방안>

1. Referer 검사 및 확인

요청을 받는 웹 어플리케이션에서 Referer 파라미터 값을 검사하여 요청을 발생시킨 페이지가 정상적인 페이지 여부를 검사하여 정상적이지 않는 경우 차단합니다.

2. XSS 취약점 확인

공격원인은 XSS와 같음으로 GET방식으로 요청이 되는 경우, 즉 스크립트 실행이 되는 경우 공격이 원활함으로 XSS취약점을 확인합니다.

3. 보안 토큰 생성

사용자의 행동이 발생하기 전, 현재 사용자의 세션과 그 세션이 취할 액션을 검증할 토큰을 포함하여 전송 후, 요청 발생시 서버에서는 액션 토큰과 사용자의 세션 토큰을 비교하여 올바른 사용자가 취하는 액션인지 검증합니다. 쉽게 말해 action이 필요한 모든 서비스 페이지에 보안토큰을 넣어 확인 하는 것입니다. 현재 이 방법은 안전한 방법으로 널리 알려져 있습니다.

4. 정보 변경시 CAPTCHA를 이용하여 재인증 요구

CAPTCHAR는 HIP(Human Interaction Proof) 기술의 일종으로, 어떠한 사용자가 실제 사람인지 컴퓨터 프로그램인지를 구별하기 위해 사용되는 방법입니다. 사람은 구별할 수 있지만 컴퓨터는 구별하기 힘들게 의도적으로 비틀거나 덧칠한 그림을 주고 그 그림에 쓰여 있는 내용을 물어보는 방법이 자주 사용되며, 이것은 기존의 텍스트와 이미지를 일그러뜨린 형태로 변형한 후 인식 대상이 변형된 이미지로부터 기존 이미지를 도출해 낼 수 있는지를 확인하는 방식의 테스트입니다. 컴퓨터 프로그램이 변형시킨 이미지는 사람이 쉽게 인식 할 수 있지만 컴퓨터 프로그램은 변형된 이미지를 인식하지 못하므로 테스트를 통과하지 못한다면 테스트 대상이 사람이 아님을 판정할 수 있고 흔히 웹 사이트 회원가입이나 회원정보 변경시 인증을 요구할 때 쓰입니다.

5. HTTPS 사용

공격자가 파라미터 값을 확인 할 수 없도록 암호화된 프로토콜인 HTTPS를 사용합니다.

위의 5가지 방안 중 1번의 대응 방안으로 구현하겠습니다.

1. Referer 검사 및 확인

위에서 취약점을 시현했듯이 Change Information 페이지에서 URL의 파라미터값을 조작하여 회원정보를 변경 하는 공격을 했습니다. 이에 따라 회원정보가 CSRF기법을 통한 공격을 방어하도록 서버사이드 스크립트인 member_modify_ok.asp의 코드를 수정합니다.Referer 헤더의 문자열 길이를 확인하는 방법으로는 CSRF공격을 방어할수 없으므로 그 안의 내용이 정상적인지 확인하는 방법으로 방어해야 합니다. 따라서 사용자에 따라 Referer의 파라미터값이 바뀌는 경우를 고려하여 CSRF공격을 방어합니다.

member_modify_ok.asp - 메모장

파일(F) 편집(E) 서식(O) 도움말(H)

```
<!--#include file="../dbconn.inc"-->
<html>
<head>
<title>회원 정보 변경</title>
<meta http-equiv="Content-Type" content="text/html; charset=euc-kr">
<%
StringRefer = Request.ServerVariables("HTTP_REFERER")
user_id = session("user_id")
exec = request("exec")
nickname = request("nickname")
email = request("email")

%>
```

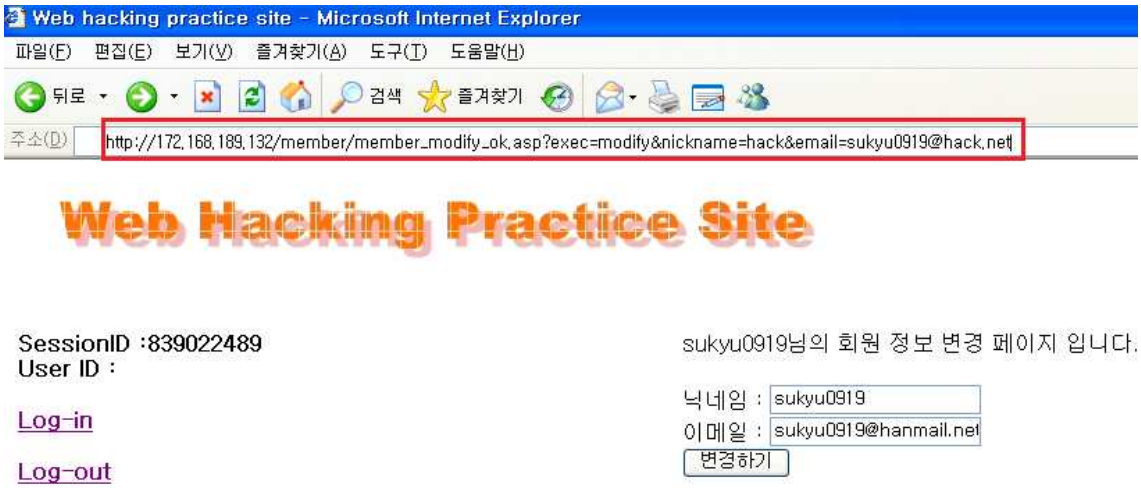
member_modify_ok.asp의 변수선언부입니다.

Referer는 어떤 링크를 타고 들어왔나를 확인 할 수 있게 해주는 내장변수입니다.

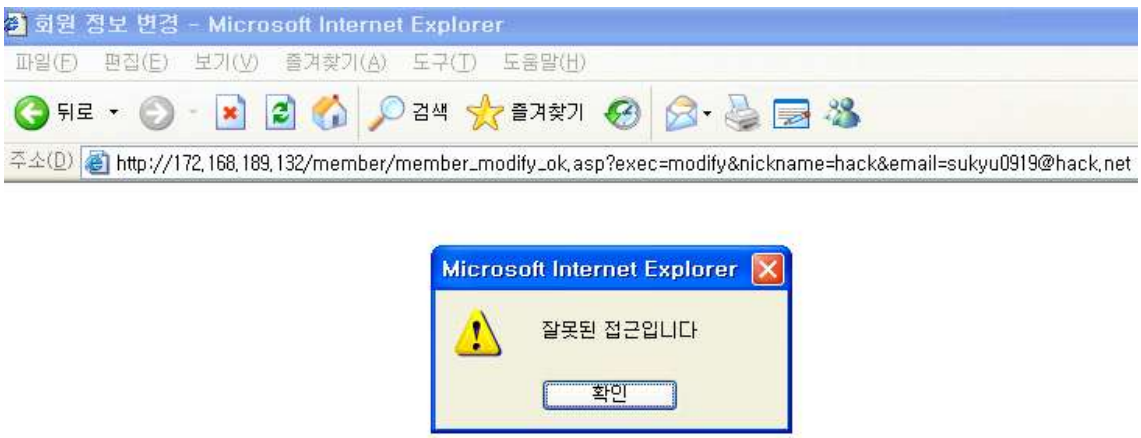
Request.ServerVariables("HTTP_REFERER")라는 Referer 헤더를 가지고 오는 변수를 선언합니다.

```
<body>
<%
        if StringRefer <> "http://172.168.189.132/member/member_modify.asp" Then
        response.write "<script>alert('잘못된 접근입니다');</script>"
        response.end
        end if

        if exec = "modify" then
                strSQL = "update member set nickname = '" + nickname + "' , email = '" + email + "' where user_id = '" + user_id +"'"
                set Rs=DBconn.execute(strSQL)
                //Rs.close
                //set Rs=nothing
                DBconn.close
                set DBconn=nothing
        end if
                response.write("회원 정보가 변경 되었습니다.<br>")

%>
```

웹 사이트의 서버관리자가 사이트 방문객이 어떤 경로로 자신의 사이트에 방문했는지 알아볼 때 유용하게 사용되는게 Referer이며 Change Iformaion의 페이지와 대조하여 직접접근을 차단할 수 있습니다. <> 같지않을 경우 response.write 출력을 통하여 경고창이 뜨고 응답을 끝내도록 수정하였습니다.



위와 같은 방법으로 URL을 통하여 회원의 닉네임과 이메일 값을 수정하여 공격을 시도합니다.



경고창이 뜨면서 실질적으로 접근이 차단되는 것을 확인합니다.

paros 툴을 이용하여 정상적으로 동작하는지 확인합니다.

```
GET http://172.168.189.132/member/member_modify_ok.asp?exec=modify&nickname=hack&email=sukyu0919@hack.net HTTP/1.0
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash, */*
Accept-Language: ko
Cookie: ASPSESSIONIDAQSAAARD=HJHHCACDBGOCJDKNDDHIPDIC
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1) Paros/3.2.13
Host: 172.168.189.132
Proxy-Connection: Keep-Alive
Content-length: 0
```

GET메소드를 인용한 URL직접접근을 확인합니다.

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/5.0
Date: Wed, 29 Aug 2018 07:45:42 GMT
Connection: Keep-Alive
Content-Length: 184
Content-Type: text/html
Cache-control: private
```

```
<html>
<head>
<title>회원 정보 변경</title>
<meta http-equiv="Content-Type" content="text/html; charset=euc-kr">

</head>

<body>
<script>alert('잘못된 접근입니다');</script>
```

접근이 통제되는 것을 확인합니다.

이는 URL 파라미터 조작으로 인한 직접접근을 차단하기 위해서 코드를 작성하였습니다.