Special Report

웹 취약점과 해킹 매커니즘#9 CSRF(Cross-Site Request Forgery)

■ 개요

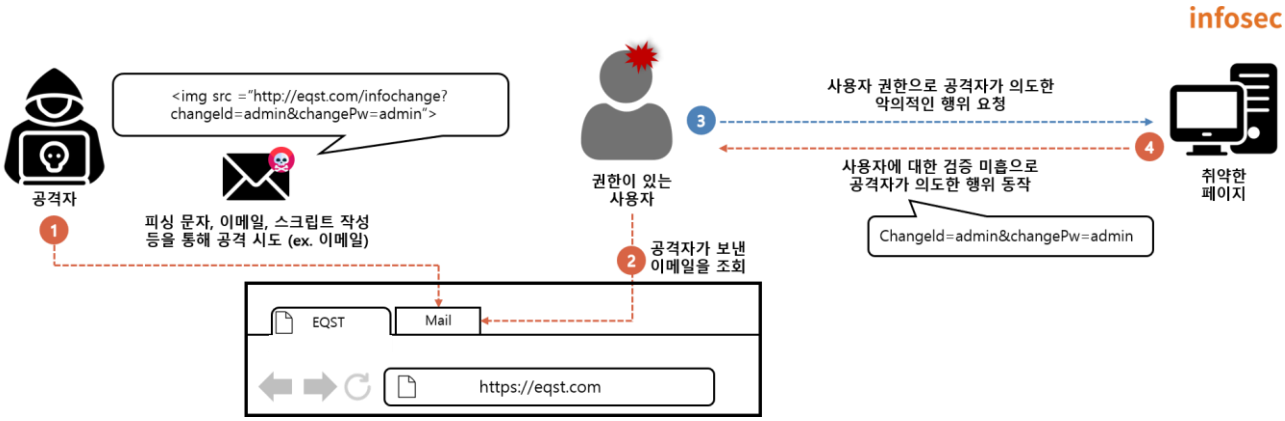
CSRF(Cross-Site Request Forgery)는 웹 취약점 공격의 하나로, 공격자가 타 사용자의 권한을 이용해 자신이 의도한 동작을 서버에 요청하게끔 유도하는 방식이다. 서버에 사용자 요청에 대한 적절한 검증 절차가 없을 때, 정상적인 요청과 조작된 요청을 구분하지 못할 경우 발생한다. 특히, 공격당한 사용자의 권한을 그대로 사용한다는 점에서 사용자의 권한 수준에 따라 피해 범위가 달라질 수 있다.

이번 Special Report 에서는 서버가 사용자의 정상적인 요청과 조작된 요청을 구분하지 못해 발생하는 취약점인 CSRF 의 개념, 동작 원리, XSS(Cross-Site Scripting)와의 비교, 보안 대책과 우회 방법에 대한 내용을 소개한다.

■ CSRF 동작 원리

CSRF 공격에 성공하기 위해서는 공격 대상 페이지의 로직을 분석하여 파라미터의 전송 형태와 각 파라미터의 기능을 알아야 한다. 이후 공격자는 피해자가 서버로 HTTP 요청을 보내도록 유도하는 스크립트를 작성하여, 피해자가 해당 스크립트를 읽었을 때 공격이 발생하도록 한다. 피해자는 자신의 권한으로 서버에 공격자가 의도한 악의적인 요청을 보내게 되는데, 이 때 사용자에 대한 검증 로직이 미흡할 경우 공격이 성공하게 된다. 공격 대상의 권한에 따라 공격자가 의도한 행위가 수행되기 때문에 공격 대상이 관리자일 경우, 피해를 가중시킬 수 있다.

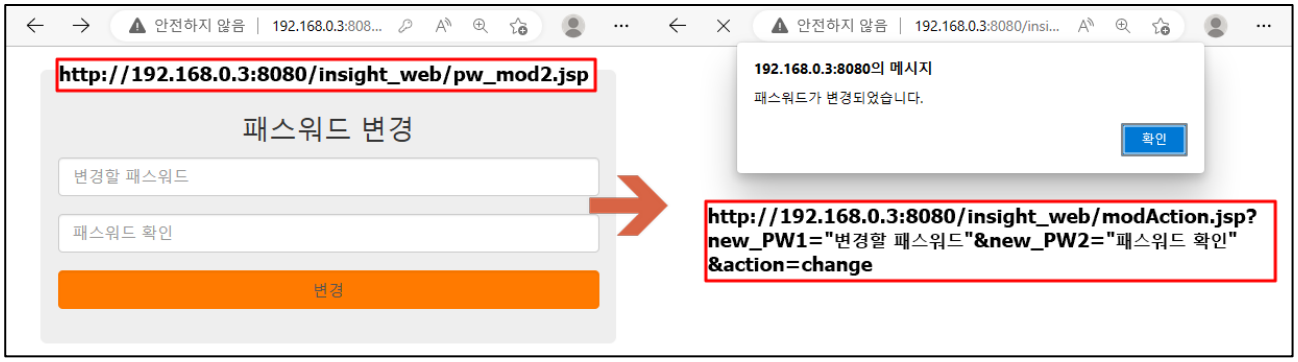
다음은 CSRF 동작 원리를 나타낸 그림이다.



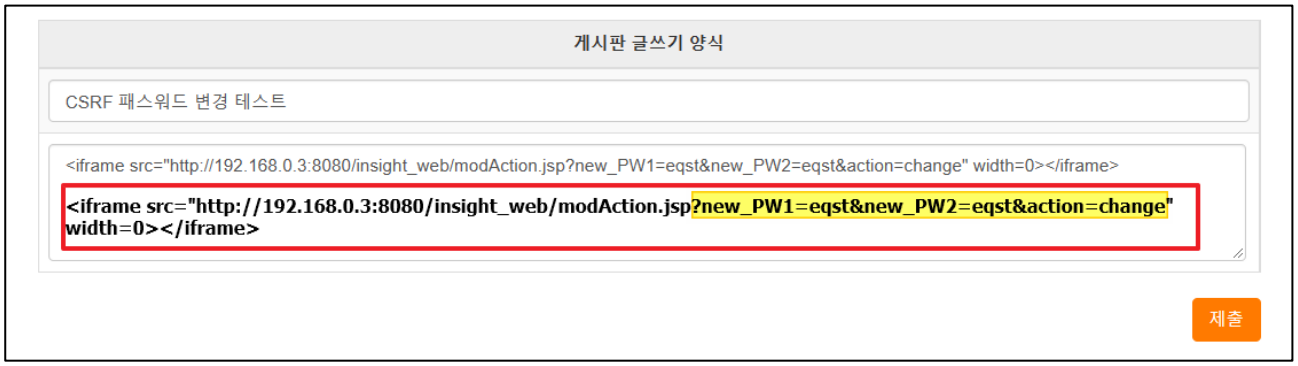
공격자는 CSRF 공격을 통해 공격 대상의 권한으로 게시글을 작성하거나 계정 정보 변경, 권한 상승 등이 가능하며 권한을 도용하여 피싱 메일 전송, 금융 거래 등의 특정 행위를 진행할 수 있다.

■ CSRF 공격 예시

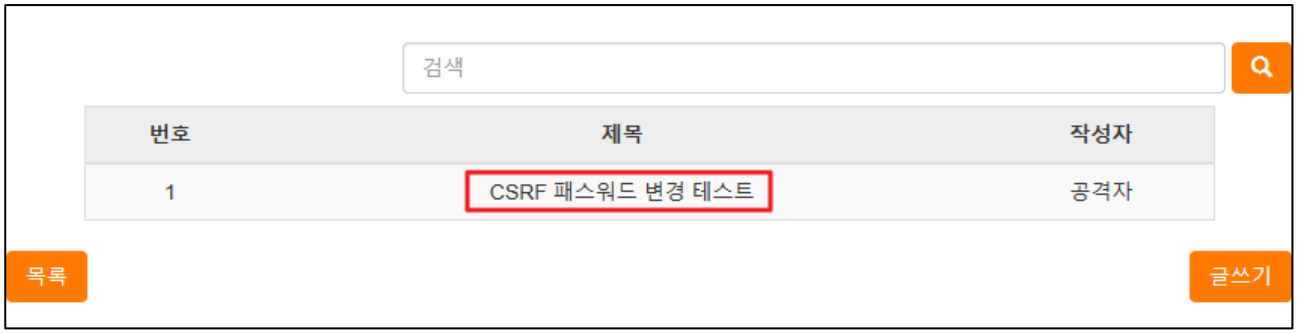
step 1) 패스워드 변경 페이지에서 패스워드 변경 요청을 보내면 아래와 같은 파라미터를 전송하여 패스워드가 변경된다.



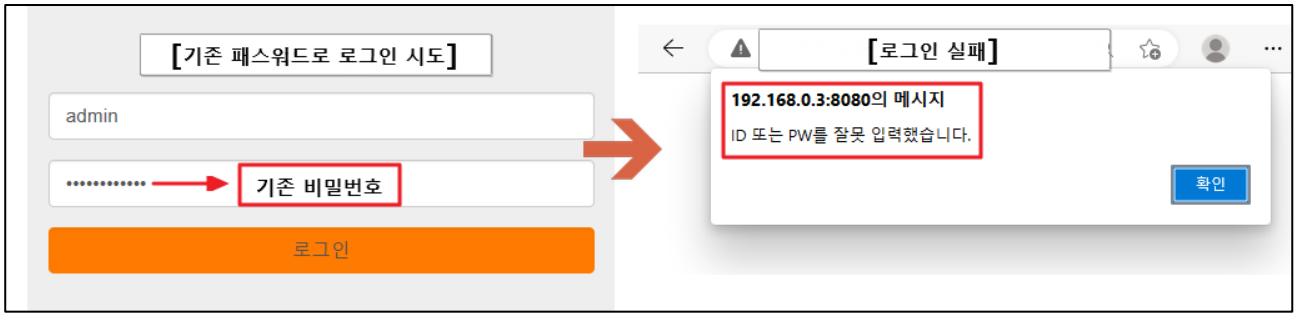
step 2) 공격자는 자신이 원하는 패스워드(eqst)로 변경을 유도하는 CSRF 스크립트를 작성하여 게시글을 등록한다.



step 3) 피해자는 공격자가 등록한 CSRF 스크립트가 삽입된 게시글에 접근한다. 피해자 측에서 패스워드 변경 스크립트가 동작하여 피해자의 패스워드가 변경된다.



step 4) 피해자(admin)가 기존 패스워드로 로그인을 시도하면 로그인에 실패하는 것을 확인할 수 있다. 또한 공격자는 피해자의 아이디와 자신이 변경한 패스워드를 통해 피해자 계정으로 로그인할 수 있다.



■ XSS vs CSRF

CSRF 는 공격 시 악의적인 스크립트를 사용한다는 점에서 XSS 과 연관 지어 언급되는 경우가 많지만, CSRF 와 XSS 는 다른 취약점 특성을 갖고 있다는 점에서 주의가 필요하다.

XSS 와 CSRF 의 가장 큰 차이점은 공격 대상에 있다. XSS 의 경우 불특정 다수의 클라이언트를 대상으로 공격을 진행하지만, CSRF 는 서버를 대상으로 공격을 진행한다. 즉 공격자에 의해 삽입된 스크립트가 서버에서 실행되는지, 클라이언트 측에서 실행되는지에 따라 구별할 수 있다.

공격 목적에 있어서도 차이가 있다. XSS 는 클라이언트 측에서 동작하여 클라이언트(사용자)의 정보 탈취에 목적이 있지만, CSRF 는 공격 대상의 권한을 통해 공격자가 원하는 요청을 보내는 공격이므로 공격자의 의도대로 서버가 동작하게끔 유도한다.

다음의 표는 XSS 와 CSRF 를 비교한 내용이다.

	XSS	CSRF
발생 원인	클라이언트가 웹 서버를 신용하여 발생	웹 서버가 클라이언트를 신용하여 발생
공격 대상	클라이언트	서버
공격 목적	쿠키/세션 탈취, 악성 사이트로의 이동 등	권한 도용, 권한 상승 등 공격자가 원하는 행위 수행
공격 행위	악의적인 스크립트 작성하여 페이지에 포함시킨 후 실행 유도	서버에서 제공하는 기능을 페이지에 포함시킨 후 실행 유도

■ 보안 대책 및 우회 방법

CSRF 공격이 가능한 이유는 요청에 대한 서버의 적절한 검증 절차가 없기 때문이다. 이에 서버로 전달되는 요청이 정상적인 요청인지, 공격자에 의한 비정상적인 요청인지에 대한 검증 로직이 중요하다.

다음의 표는 CSRF 보안 대책과 우회 방법, 보안 강도에 대한 내용이다.

보안 대책	구분	설명	보안 강도
XSS Filtering	보안 기법	조작된 요청 생성에 사용되는 ?, &, <, > 등의 특수문자를 필터링하여 사이트에서 스크립트가 실행되지 않도록 한다. *XSS Filtering 에 대한 내용은 웹 취약점과 해킹 매커니즘#7, #8 에서 확인할 수 있다.	**미흡**
	우회 방법	Filtering 규칙이 미흡할 경우 필터링 우회 가능성이 존재한다. 또한 외부 사이트에서 실행되어 요청을 보내는 경우 공격이 발생할 수 있다.	
GET/POST 구분	보안 기법	GET 메소드를 활용한 URL 공격을 하지 못하도록 POST 메소드를 사용한 요청만 허용하도록 한다.	**미흡**
	우회 방법	POST 요청을 보내는 스크립트를 작성하여 우회가 가능하다.	
사용자 요청 검증	보안 기법	Submit Button 등을 이용하여 정상 사용자가 보내는 요청인지 검증하는 수단을 만든다.	**일부 미흡**
	우회 방법	공격자가 검증 값을 포함한 요청을 보내도록 스크립트를 작성할 시 우회가 가능할 수 있다.	
보안 토큰 사용	보안 기법	중요 기능 동작 시 랜덤으로 발행되는 보안 토큰을 발행해 토큰 값 일치 여부에 따라 요청이 동작하도록 구현한다.	**일부 미흡**
	우회 방법	토큰 발급 페이지에 접근한 후 토큰 값을 획득해 요청을 보내면 정상 요청으로 판단하여 우회가 가능하다,	
추가 인증	보안 기법	중요 기능에 대해 2 차, 3 차의 추가 인증을 필수적으로 수행하도록 구현한다.	**양호**
	설명	비밀번호 입력, 휴대폰 인증(문자, ARS), Captcha 등으로 동작 행위자만 수행할 수 있는 인증을 추가로 구현한다. 이는 보안성이 뛰어나지만 사용자로부터 매번 인증을 수행하게 하여 사용자의 편의성이 떨어진다.	

각 보안 대책별 상세 내용과 우회 방안은 다음과 같다.

1) XSS Filtering – (미흡)

- 보안 대책

공격자에 의해 악의적인 요청 생성에 사용되는 ?, &, 〈, 〉 등의 특수문자 필터링한다. 특수 문자 필터링의 예시는 아래와 같다.

*XSS Filtering 에 대한 내용은 웹 취약점과 해킹 매커니즘#7, #8 에서 확인할 수 있다.

특수문자	<	>	'	"	(	)
Entity	<	>	'	"	(	)

- 우회 기법

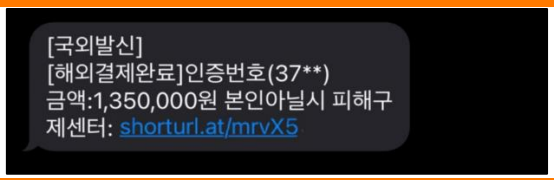
XSS 취약점을 제거하는 것은 CSRF 의 공격 범위를 줄여주는 역할만 하기 때문에 안전한 보안 대책은 아니다. 두 취약점은 전혀 별개의 취약점으로 해당 사이트에서 XSS 공격이 불가능해도 CSRF 취약점이 존재할 수 있다. 공격자는 피싱 문자, 피싱 이메일, 악의적인 스크립트가 포함된 게시글 작성을 통해 공격을 시도한다. 이에 대한 전제조건은 피해자가 로그인 등을 통해 사이트에 대한 권한을 획득하고 유지한 상태여야만 한다.

예를 들어, SK 쉴더스 사이트에 보안 담당자가 사이트에 발생할 수 있는 XSS 를 모두 조치한다고 하더라도 피싱 메일이나 SMS, 혹은 XSS 취약점이 존재하는 타 사이트를 통해 접근하여 공격하는 시나리오가 가능하기 때문에 XSS 를 막는 것은 CSRF 와 전혀 상관이 없다. 다만 CSRF 공격이 XSS 와 자주 연계되는 공격이며, XSS 가 완벽하게 막혀 있다면 아래 소개될 토큰을 통한 보안 적용이 함께 이뤄지기 때문에 이를 고려하여 보안대책으로 활용할 수 있다.

2) GET/POST 구분 – (미흡)

- 보안 대책

GET 메소드 허용 시 단순 URL 로도 CSRF 공격이 가능하며, 이는 POST 메소드를 사용하여 막을 수 있다. GET 메소드의 경우 외부로부터 변수 값에 직접적으로 접근 가능하기 때문에 공격에 취약하다. 아래는 위에서 언급한 피싱을 통한 공격 시나리오 중 피싱 문자에 해당하는 경우로, 메시지 내 링크를 클릭할 시 공격자가 의도한 악의적인 행위가 실행될 수 있다.

화면	공격 스크립트
	"https://eqst.com/infochange?changeId=admin&changePw=admin" (*URL 은 shorturl 을 통해 변형되었다.)

- 우회 기법

공격자는 HTML 태그나 javascript 사용이 가능한 게시판과 같은 곳에서 POST 메소드로 요청을 보내도록 스크립트를 작성하여 공격 대상이 해당 요청을 전송하도록 유도한다. Iframe 을 사용할 경우 창 크기를 0 으로 만들어 공격자는 사용자가 알지 못하도록 요청을 숨겨 보낼 수 있다. 아래는 피해자의 계정 정보를 공격자가 원하는 계정 정보로 바꾸도록 하는 스크립트 예시이다.

```
<iframe width="0" height="0" border="0" name=" stealthframe" id="stealthframe"
style="display: none;"></iframe>
<body onload="document.csrf_form.submit();">
<form method="POST" name="csrf_form" action="https://eqst.com/infochange"
target="stealthframe">
<input type ="hidden" name="changeId" value="admin">
<input type ="hidden" name="changePw" value="admin">
</form>
```

3) 사용자 요청 검증 – (일부 미흡)

- 보안 대책

서버로 전송되는 요청이 정상적인 사용자에 의한 요청인지 확인하는 검증 로직을 추가하는 방법이다. 검증 로직은 Submit Button, Alert 창 등을 이용해, 피해자 몰래 실행되는 것이 아닌 알림창을 확인하고 실행하도록 로직을 변경하는 것이다. 이 보안대책을 적용하면 아래와 같이 로직 실행 확인 용도의 파라미터가 추가된다.

검증 값을 추가하기 전	검증 값을 추가한 후
changeId=admin& changePw=admin	changeId=admin& changePw=admin &**confirm=ok**

- 우회 기법

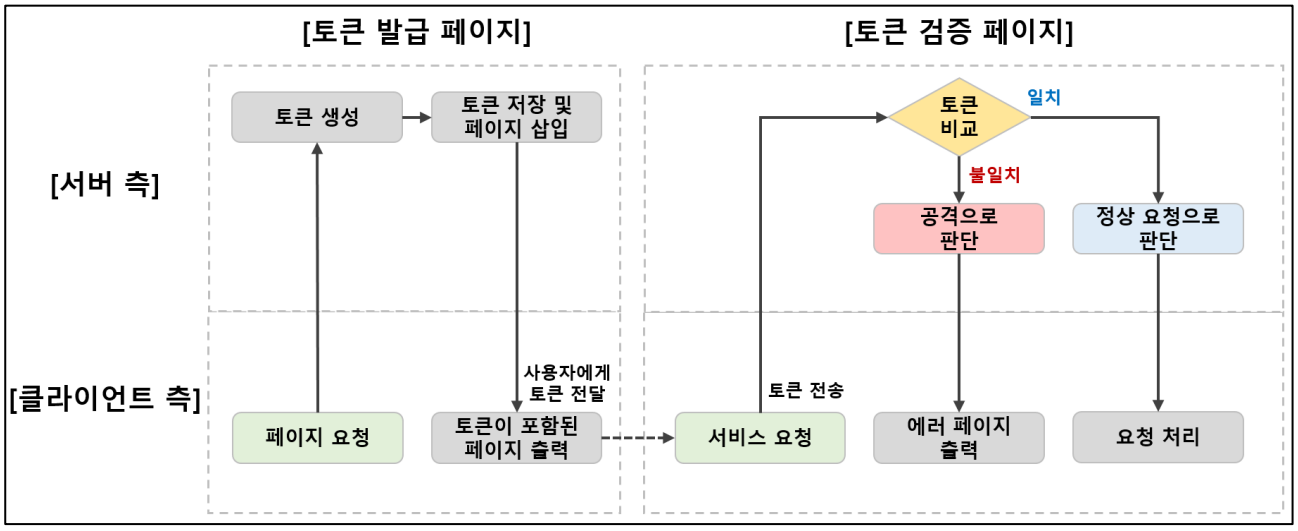
공격자 입장에서는 alert창을 확인한 이후에 파라미터명/값만 알게 되면 파라미터를 미리 설정할 수 있다. 이 값이 고정이고, Y,N,ok 등과 같은 파라미터를 쓸 경우 쉽게 유추가 가능하다. 특히 오픈소스로 개발되었거나 무료게시판을 사용하는 사이트는 공격자가 미리 구조를 파악하여 공격하기 때문에, 피해가 더 자주 발생하는 편이다.

```
<iframe width="0" height="0" border="0" name=" stealthframe" id="stealthframe"
style="display: none;"></iframe>
<body onload="document.csrf_form.submit();">
<form method="POST" name="csrf_form" action="https://eqst.com/infochange"
target="stealthframe">
<input type ="hidden" name="changeId" value="admin">
<input type ="hidden" name="changePw" value="admin">
<input type ="hidden" name="confirm" value="ok">
</form>
```

4) 보안 토큰 사용 – (일부 미흡)

- 보안 대책

사용자 요청 검증이 고정값이고 유추하기 쉬워 우회가 가능하다면, 이를 고도화한 것이 토큰 방식이다. 보안 토큰의 경우 토큰 '발급' 페이지에서 공격자가 유추할 수 없는 랜덤값을 매번 생성하고, 요청 시마다 변경되며, 토큰 '검증' 페이지에서 유효성 검사를 한다. 매번 한 번만 사용하고 만료처리를 하여 재사용을 방지하는 보안 로직이다. 예를 들어 관리자만 접근이 가능한 사용자 권한 조회 페이지와 사용자 권한 수정 페이지가 존재한다고 가정하면, 권한 조회페이지에서 토큰을 발급하고, 수정페이지에서 토큰을 검증하는 형태다. 이렇게 보안대책을 설정할 경우, 게시판에서 토큰값이 없는 요청이 갑자기 온다 하더라도 로직을 실행시키지 못하기 때문에 안전하게 사용할 수 있다.



[토큰 동작 방식]

- 우회 기법

공격자가 공격을 수행하기 위해서는 토큰 발급 페이지에 접근이 가능해야 한다. 토큰 발급 페이지에서 토큰을 발급한 후 생성된 토큰을 이용하는 스크립트를 작성해 공격을 수행한다. 토큰 획득시에 다른 사이트에서 XSS 를 통한 접근이 불가능하여 같은 사이트 내에서 발생하는 XSS 공격을 이용해 토큰을 획득해야 한다.

토큰 우회 로직은 다음과 같다.

(1) 토큰 발급 페이지에서 토큰을 생성해 변수로 받아온다.
(2) 서버로 보내는 요청에 변수로 받아온 토큰을 자동으로 제출하도록 공격 스크립트를 작성한다.
(3) 같은 사이트 내에서 XSS 공격과 연계해 스크립트를 실행시킨다.
(4) 토큰 검증 페이지에서 정상요청으로 판단되어 요청이 처리된다.

아래는 실제 공격에 사용되는 페이로드이다.

```
<script>
var readToken = function(){
   var doc = document.getElementById("frame1").contentDocument
   var token = doc.getElementsByName("csrf_token")[0].getAttribute("value");

   var frame2 = document.getElementById("frame2");
   frame2.src = "http://eqst.com/updateTokenAuth?loginId=admin&csrf_token=" + token;
}
</script>

<iframe id = "frame2" >
</iframe>

<iframe id = "frame1" onload="readToken()" src="http://eqst.com/authpage">
</iframe>
```

위 로직이 토큰 '발급' 페이지에서 토큰값을 받아서 토큰 '검증' 페이지 요청 시 파라미터에 자동제출 하도록 스크립트를 작성하는 것인데, 이 공격기법에는 제약조건이 하나 있다. SOP(Same-Origin-Policy, 동일 출처 정책) 설정인데, 같은 사이트 내 자원만 참조가 가능한 설정이다.

예를 들면, EQST 사이트에서 EQST 사이트 자원을 참조하거나 훔치는 것은 되지만, EQST 사이트에서 타 사이트 자원을 훔치거나 참조하는 것은 불가능하다는 의미다. 따라서 스크립트가 동작해야 하는 사이트 내에서 토큰을 훔쳐서 자동 제출할 수 있어야만 공격이

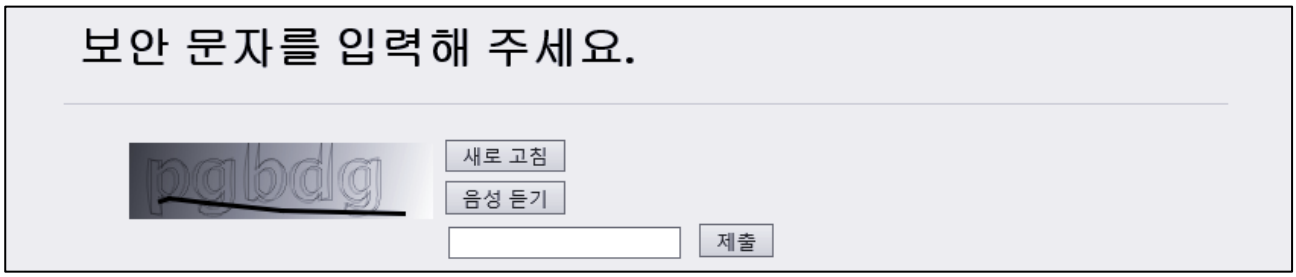
가능하다고 볼 수 있다. 다만, XSS 취약점이 존재하지 않는 사이트에서는 토큰 방식이 안전할 수 있기 때문에, XSS 와 토큰방식을 동시에 사용한다면 안전할 수 있다.

5) 추가 인증 – (양호)

- 보안 대책

지금까지 살펴본 보안 대책들은 공격자가 보안 코드를 우회하거나 검증 로직 값을 자동 제출함으로써 우회가 가능했다. 하지만 근본적으로 해커가 예측할 수 없는 값, 예를 들어 사용자 본인만 알고 있는 비밀번호나 SMS 인증 등으로 2Factor 구조 설계를 한다면 CSRF 공격은 절대로 성립될 수 없다.

따라서 2 차 인증은 사용자 본인만 알 수 있는 값으로 추가인증을 구현하거나, 사용자 본인이 수행하지 않은 요청을 방지하는 Captcha 를 통해 보안 대책을 적용할 수 있다. Captcha 는 해커가 미리 값을 설정할 수 없기 때문에 안전하다.



[Captcha 사용 예시]

■ 맺음말

CSRF 는 중요 로직에 한해서만 취약점으로 분류되고, 중요도에 따라서 보안대책을 다르게 적용할 수 있다. 예를 들어 100 원만 이체하는 경우 Token 과 XSS 를 활용하는 정도로 보안 적용이 가능하지만 100 만원을 이체하는 경우 추가 인증을 구현함으로써 중요도에 따른 대책을 적용시켜야 한다.

사용자 편의성과 보안성을 동시에 만족할 수 있다면 좋겠지만, CSRF 의 경우 그러기 쉽지 않은 탓에 어떤 보안대책을 어떻게 구성할지는 담당자와 개발부서가 협의하여 페이지마다 적절한 보안대책을 적용해야 한다.

이어지는 2 월호 Special Report 에서는 서버 측에서 이루어지는 요청을 변조하여 공격자가 의도한 서버로 임의의 요청을 하게 만드는 공격인 SSRF(Server-Side Request Forgery)에 대한 내용을 소개한다.