1. 기사 내용

제목	모질라, CSRF 공격 막기 위해 파이어폭스 60 강화
기사 날짜	2018년 4월 30일
스크랩 담당자	손우규
요약	• 동일 사이트 쿠키 값 확인해 사이트 간 요청 위조 공격 막을 수 있어 • 스트릭트와 랙스 모드 통해 사용자가 조절 가능 CSRF 공격이란 악성 행위자가 인증이 된 사용자를 특별하게 조작된 웹페이지로 들어오게 유도함으로써 웹사이트 상에서 인증이 된 사용자인 것처럼 행동하며, 승인 되지 않는 행위들을 하는 것이다. 웹사이트로 들어가는 모든 요청에는 쿠키가 포함되어 있고, 많은 웹사이트들이 인증을 위해 이 쿠키들을 활용한다는 걸 악용한 공격이다. "현재의 구조를 보완하려면 동일 사이트 쿠키 값이 필요합니다. 지금 들어온 요청 속 쿠키가 원래의 그 동일한 사이트에서 온 쿠키인지 웹 애플리케이션이 브라우저를 통해 확인하는 것이죠." 모질라의 보인 팀이 블로그를 통해 설명한 내용이다. "URL 주소창에 있는 주소와 요청이 발생한 URL이 다를 경우, 해당 요청에는 쿠키가 포함되지 않을 겁니다." 스트릭트 모드의 경우 사용자가 외부 사이트를 브라우징 하다가 인바운드 링크를 클릭했을 때 활성화된 세션이 발생한 상태라고 하더라도 인증되지 않은 것으로 처리한다. 쿠키들이 요청에 포함되지 않을 것이기 때문이다. 하지만 랙스 모드의 경우 사용자가 외부 웹사이트들을 돌아다니고 링크를 쫓아다녀도 쿠키를 함께 전송해준다. 도메인 간 하위요청들(예를 들어 이미지나 프레임에 대한 요청)의 경우는 쿠키가 전송되지 않는다. 랙스 모드는 스트릭트 모드와 호환이 되지 않는 애플리케이션을 위해 준비된 것이라고 한다.
출처	http://www.boannews.com/media/view.asp?idx=68935&kind=1&search=title&find=CSRF

2. 용어

용어	설명
CSRF(Cross-site request forgery)	사이트 간 요청 위조(또는 크로스 사이트 요청 위조, 영어: Cross-site request forgery, CSRF, XSRF)는 웹사이트 취약점 공격의 하나로, 사용자가 자신의 의지와는 무관하게 공격자가 의도한 행위(수정, 삭제, 등록 등)를 특정 웹사이트에 요청하게 하는 공격을 말한다.
Strict Mode	ECMAScript 5 부터 사용 가능한 새로운 기능이며, 자바스크립트 코드를 좀 더 엄격한 환경에서 실행할 수 있도록 합니다. Strict mode는 안전하지 않은 액션이 발생하는 것을 방지하며 예외를 발생시킵니다.

3. 관련 기술(CSRF)

3.1 개요

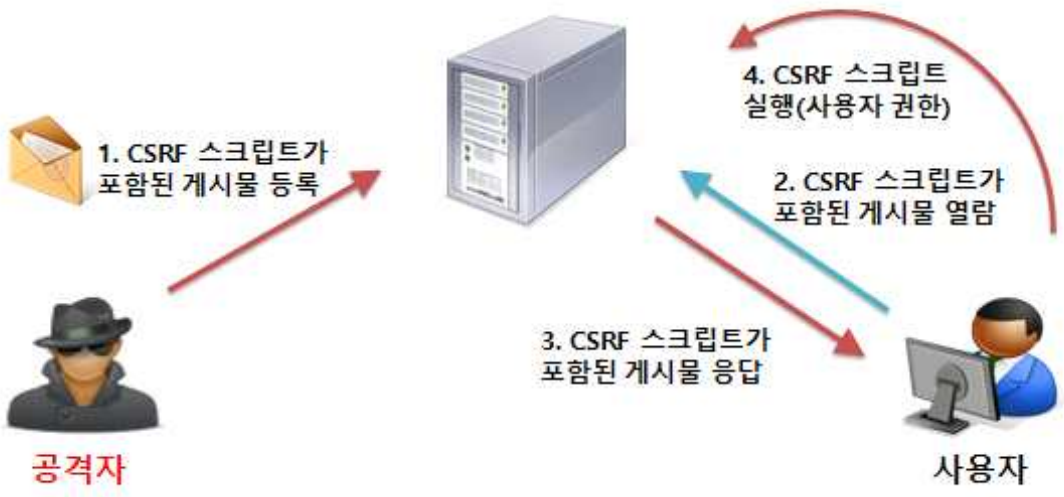
사이트 간 요청 위조(또는 크로스 사이트 요청 위조, 영어: Cross-site request forgery, CSRF, XSRF)는 웹사이트 취약점 공격의 하나로, 사용자가 자신의 의지와는 무관하게 공격자가 의도한 행위(수정, 삭제, 등록 등)를 특정 웹사이트에 요청하게 하는 공격을 말한다.

유명 경매 사이트인 옥션에서 발생한 개인정보 유출 사건에서 사용된 공격 방식 중 하나다.

사이트 간 스크립팅(XSS)을 이용한 공격이 사용자가 특정 웹사이트를 신용하는 점을 노린 것이라면, 사이트간 요청 위조는 특정 웹사이트가 사용자의 웹 브라우저를 신용하는 상태를 노린 것이다. 일단 사용자가 웹사이트에 로그인한 상태에서 사이트간 요청 위조 공격 코드가 삽입된 페이지를 열면, 공격 대상이 되는 웹사이트는 위조된 공격 명령이 믿을 수 있는 사용자로부터 발송된 것으로 판단하게 되어 공격에 노출된다.

3.2 공격 형태

3.2.1 CSRF(Cross-site request forgery) Script



1. 공격자는 CSRF 스크립트가 포함된 게시물을 등록한다.

2. 사용자는 CSRF 스크립트가 포함된 페이지의 게시물 열람을 요청한다.
3. 게시물을 읽은 사용자의 권한으로 공격자가 원하는 요청이 발생한다.
4. 공격자가 원하는 CSRF 스크립트 결과가 발생한다.

3.3 공격 원인

개별 링크와 폼이 사용자 별로 예측 가능한 토큰을 사용할 때 발생한다. 예측 불가능한 토큰이 있다면 공격자는 요청 메시지를 변조 할 수 없지만, 예측 가능한 토큰이 있다면 공격자는 요청 메시지를 변조 할 수 있다. 그러므로 상태를 변경하는 기능들을 호출하는 링크와 폼이 CSRF 공격의 가장 중요한 공격 대상이다. 그리고 인증이나 세션쿠키 등 모든 웹 사이트에서 인증된 사용자가 보내는 데이터는 정상적인 경로를 통한 파라미터 요청으로 판단한다. 즉, 정상적인 요청과 비정상적인 요청을 구분하지 못한다.

3.4 보완 방법(해결방안)

모든 입력 값들을 서버 측에서 검증해야 한다. 헤더, 쿠키, 질의문, 폼 필드, 히든 필드 등과 같은 모든 변수들을 엄격한 규칙에 의해 검증하여 HTML을 사용할 경우 태그 내에 ?, &등이 포함되지 않도록 필터링해야 한다. 가장 기본적으로 특별한 목적이 있는 경우가 아닌 이상 게시판에서 HTML을 사용하지 못하도록 하는 것이 가장 안전하다. 또한 각각의 HTTP 요청 내에 임의 토큰을 추가하여 이 토큰 값을 검증하면 대응이 가능하다.

3.4.1 서버에서 쿠키 이외의 다른 파라미터 값으로 추가 인증을 처리

중요 action을 처리할 때, 추가 인증 수단을 사용한다면 공격이 불가능하다.

3.4.2 XSS(Cross Site Script) 스크립트의 실행 방지

XSS 취약점이 존재하지 않더라도 스크립트를 실행시킬 수 있기 때문에 XSS만 막았다고 해서 CSRF를 막았다고는 할 수 없다.

3.4.3 값이 매번 바뀌는 one TIME 값 사용

인증 값을 알아내는 것이 힘들고, 사용자마다 매번 인증 값이 바뀐다. 쿠키가 아닌 다른 형태의 매번 바뀌는 인증 값을 사용한다.

3.4.4 IPS나 웹 방화벽을 사용

CSRF 스크립트는 정상적인 HTML 스크립트이기 때문에 보안 솔루션으로는 방어를 할 수 없고 중요 공격 로직을 파악하고 분석하여 안전한 웹 어플리케이션을 개발한다.

3.4.5 Referer 체크

Referer는 HTTP 헤더에 있는 정보로 해당 요청이 요청된 페이지의 정보를 가지고 있는데 해당 정보는 Paros나 Zap, fiddler같은 프로그램으로 조작이 가능하지만 방법이 간단하여 소규모 웹사이트에 주로 이용되는 방법이다.

3.4.6 GET/POST 구분

img 태그 등을 이용할 경우 GET 요청으로 들어오게 될 것이고, 반면 흔히 하듯 form을 이용해 값을 받을 경우 POST를 이용하게 되는 경우가 많기 때문이다.

4. 참고 자료

- https://ko.wikipedia.org/wiki/%EC%82%AC%EC%9D%B4%ED%8A%B8_%EA%B0%84_%EC%9A%94%EC%B2%AD_%EC%9C%84%EC%A1%B0