AWD 线下攻防生存之道（Web 方向）

0x00 前言

本 Chat 将从实战的角度去讲解 AWD 比赛。AWD 攻防简单来说就是在一个线下的局域网中相互厮杀。守护自己服务，攻击甚至干掉其他人的服务。

0x01 简介

AWD 是 Attack with defence 的缩写，即要求在比赛的过程中攻守兼备，不可放弃任何一方。

如果只注重攻击而不重视防御，那么将会导致每轮你的得分几乎等于你的减分，排名很难有突破性的上升。

相反，如果只注重防御。那么前期排名可能会在一个中间的位置。但是，一但有人刷出一个你没有防御到的漏洞，同样也是凉凉。

0x02 比赛形式

俗话说，知己知彼方能百战不殆。所以正式开始之前一定要了解 AWD 比赛的常见形式。

1. 比赛开始前会给每支队伍分配 SSH 账号，比赛开始用该账号登录服务器进行维护（多为 Linux 服务器）。
2. 在服务器的某处有一个 flag 文件，默认没有权限修改，一般在根目录下。
3. 主办方每隔一定时间，进行一轮刷新。一轮内一支队伍的 flag 只能被提交一次，flag 一旦被拿走将扣除该队的分数。
4. 主办方会对服务进行 check，一般的判断标准就是服务是否还存在。

扣除的积分由获取 flag 的队伍均分。

0x03 团队分工

比赛中比较建议的分工是两个人进攻，一个人负责运维。接下来将从进攻和防御的角度来谈谈如何完成一场 CTF 比赛。但是该方式只是建议，比赛过程中应结合团队的具体能

力进行合理分工。

防御

1. 比赛开始的一瞬间，登录 SSH 修改密码

Windows：

> PuTTY： https://the.earth.li/~sgtatham/putty/0.70/w64/putty.exe

Linux：

```
ssh  -p:端口 用户名@ip地址
```

2. 上通防御 WAF、文件监控脚本流量、流量混淆脚本

(1) WAF

WAF 的原理是通过过滤一些参数，来抵御大部分的攻击。挂 WAF 的原则是将其挂在尽可能多的被包含的 .php 文件中。比如 config.php。

打开某个文件在文件的头部添加

```
<?php
    include "waf.php";
 ?>
```

(2) 文件监控

文件监控脚本的功能是监控指定时间内修改的文件，将其删除。
在其监控的目录下直接运行。

(3) 流量混淆

在作者刚刚参加 CTF 比赛时，就有大佬告诉我：如果自己不会攻击的话，就去分析流量看别人是如何打你的。或者说，为了防止其他人偷窃你“辛苦的劳动成果”，此时流量混淆十分有必要。

这里安利一个链接：

> https://bbs.ichunqiu.com/thread-46072-1-1.html

3. 仔细查看文件

仔细查看 Web 目录下的文件，看看是否有一些可疑文件，比如一句话木马或者 .bak 文件等。

4. 根据队友的反馈，编写批量提交 flag 的脚本。同时根据队友反馈出来的漏洞进行及时的修补漏洞。

注意事项：

- 修补的时候不要随意的删除文件，很容易被 check。有些时候如果 WAF 过滤得严，也容易被 check。
- 有些比赛中可能有多台靶机（Web、二进制）可以根据情况，每人负责一台，各自进攻和防御。

进攻

进攻可以说比赛排名的关键，防御做得好的话，可以将排名维持在中等，此时比赛的排名就完全取决于进攻的人获取的分数。

1. 重要重要重要

下载源码、下载源码、下载源码。

如果一不小心被人删了源码，可以自己上传文件而不浪费重置的机会。而且比赛完成以后自己也可以搭建环境进行研究。

此时建议负责进攻的一人下载源码，可以利用 Filezilla 连接 FTP 进行下载。

> https://download.filezilla-project.org/client/FileZilla_3.39.0_win64-setup_bundled.exe

2. 权限维持

在 AWD 比赛中，为了照顾比较菜的选手，一般都会有预留后门或者 Web 目录弱口令。此时可用 D 盾进行扫描源码，一般在几轮之后大部分队伍便会修复。

此时如果有一个漏洞的话，想到的不应该是刷 flag，而应该是维持权限。你比别人多维持一轮的权限，也就意味着你能比别人多刷一轮的分。

权限维持一般采用两种方法：**不死马和反弹端口。**

- 不死马：不死马的原理是，将程序写入内存中，无限执行。网上的不死马有很多，笔者比较菜。杀死不死马的方法也有很多。
  - ps auxww | grep 1.php
  - 生成一个同名文件
- 反弹端口：一般利用 NC 反弹端口，代码网上有好多，后期会分享。代码上传成功以后，访问执行。然后本地执行 `nc -lp 9999` 。

3. 代码审计

一些简单的比赛利用完成以后，剩下便是漏洞的发现与利用。此时便是代码审计。

安利大佬的链接：

> https://www.anquanke.com/post/id/98574

注意事项：

- 一定要备份源码
- 一定要注意权限的维持

0x04 搅屎攻略

古人说过：打得过就打，打不过就跑。在 AWD 中，大概是打得过就打，打不过就让你宕机。

宕机的可能大概有如下几种：

1. 删系统目录： `rm -rf *`

如果开始存在 SSH 弱口令并且存在 root 权限，手速够快的话可以登录删系统（最好提前写好脚本），或者得到对方 Shell 以后尝试提取。

2. 删 Web 目录

得到对方 Shell 以后可以尝试删除，可以让对方被 check。然后写一个定时任务，每隔五分钟删一次目录。

定时任务，在固定的时间间隔执行指定的系统指令或 Shell script。

```
crontab [-u user] file_name
crontab [-u user] [-e |-l| -r]
```

3. 让对方服务崩溃

原理就是不断地申请内存，直到对方服务器崩溃。

安利一段代码：

```
<?php
        set_time_limit(0);
        ignore_user_abort(true);
```

```
            while(1){
file_put_contents(randstr().'.php',file_get_content(__FILE__));
            file_get_contents("http://127.0.0.1/");
        }
        ?>
```

4. 心理战

得到一个 Shell 以后，删掉其主页，然后替队友挂一个黑页。将战火引到某一个比自己分数高的团队上去。

5. 提升权限

普通权限玩起来当然不如 root 权限好玩。可以的情况下尝试提权。
安利一个链接：

6. 封 IP

由于比赛前需要接入局域网，所以比赛前需要配置 IP 地址。因此可以猜到其他选手的 IP 地址，封掉其他选手 IP。完美解决。

在服务器上进行如下命令操作进行规则设置即可：

```
iptables -A INPUT -s ip段/网络位数 -j DROP
```

例如：禁止 192.168.11.0/24 网段访问服务器，直接在服务器上用命令就可以实现。

```
iptables -A INPUT -s 192.168.11.0/24 -j DROP （添加规则，所有来自这个网段的数据都丢弃）
/etc/rc.d/init.d/iptables save （保存规则）
service iptables restart （重启iptables服务以便生效）
```

0x05 CTFD 平台搭建

搭建 CTF 平台推荐使用 CTFD，界面简单，同是对服务器资源占用不是太高。

1. 首先更新一下系统

```
apt-get update&&apt-get upgrade &&apt-get dist-upgrade
```

2. 安装 git

```
apt-get install git
```

3. 安装 pip、setuptools

```
apt-get install python-pip python-setuptools.
```

1. 安装 Flask

```
pip install Flask
```

1. 下载 & 安装 & 运行 CTFD

```
sudo git clone https://github.com/isislab/CTFd.git
cd CTFd
sudo ./prepare.sh
sudo python serve.py
```

然后这时候访问 127.0.0.1看 是否出现页面。出现则成功。

1. 更新 pip

```
pip install --upgrade pip
```

1. 安装 Gunicorn

```
$ sudo pip install gunicorn
$sudo gunicorn --bind 0.0.0.0:4000 -w 1 "CTFd:create_app()"
```

安装完成以后可以访问公网地址 4000 访问。其实端口可以改变。

0x06 AWD 平台搭建

准备一场简陋线下的 AWD 比赛需要一台服务器、一台电口三层交换机、网线若干。

1. 服务器搭建 Ubuntu 18.04 Desktop。
2. 安装 Python 2.7 和 Docker。
3. 这里利用 GitHub 上大佬写好的 AWD 平台 https://github.com/zhl2008/awd-platform。
4. 首先讲环境 pull 下来大概 1.5G 左右（Docker pull zhl2008/web_14.04)。

5. 然后按照 GitHub 中的说明启动比赛环境即可，此处不做详细说明。
6. 交换机需要划分 VLAN，还有做好访问控制。本人是一个硬件小弟，目的是为了防止内网攻击，比如 ARP 等。

此过程不再详细叙事，组织一场比赛是一个繁杂的过程，可能会有很多的情况发生。如有兴趣可以依照此过程详细探索。

0x07 写在最后

本人也是一个小菜，如有问题多多交流。写的有不足之处，还望各位大佬海涵。