Wireshark and Password Cracking

Discord - Need assistance?

If you are stuck, please use our discord server for assistance - not the TryHackMe chat. There are lots of people at the ready to help you.

Even if you're not stuck, we'd love to see you on the TryHackMe discord!

Our discord server link: https://discord.gg/zGdzUad

Wireshark

Given the right permissions, anyone can load a program such as Wireshark and start capturing network traffic. Without going into low-level detail about how a frame is formed (a network packet formed created using the OSI model layers), you can easily filter through a network capture file and view what data your computer has been sending and receiving.

You can sniff your own network traffic and see your own data sent and received, or you can sniff traffic on a switch or hub (where data is sent to to be routed to other devices) and reveal what everyone has been looking at.

Without packet data being encrypted you could see all network requests and responses, along with its data; you could see what websites people have been visiting, users personal information (credentials, bank account data).. Anything.. Providing the data is not encrypted of course. Protocols such as telnet and http will transfer data in plaintext, which means you can extract human-readable data out of it.

Below is a table of terms used throughout this document.

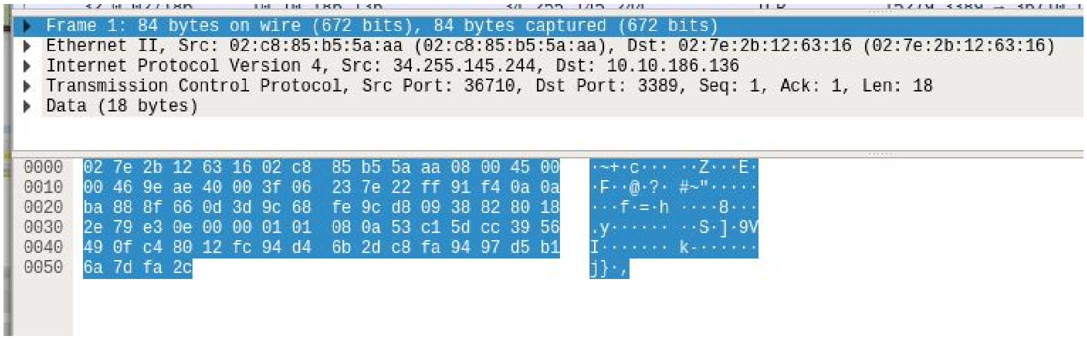
Term	Description
Packet	A packet consists of control information and user data, which is also known as the payload.
Protocol	A network protocol is a set of rules followed by the network. An example of a protocol is HTTP, explained in challenge 1.
Port	A network port is a number that identifies one side of a connection between two computers. Computers use port numbers to determine to which process or application a message should be delivered. For example, SSH uses port 22 to communicate, Telnet uses port 23 and FTP (file transfer protocol) uses port 21.

We will use Wireshark, a free and open-source packet analyzer tool to review a pcap file (pcaps are the file extension for network files, commonly associated with Wireshark).

To open a pcap file, on Wireshark click File -> open. This will load the network capture, at a first glance you will see the following columns:

- No - This is the packet number
- Time - This is the time the packet was captured from when the ‘sniffing’ took place.
- Source - The IP address of the source device
- Destination - The IP address of the destination device
- Protocol - The protocol of the packet
- Length - The length of the frame
- Info - Common information.

When you select a packet, you will see the following:



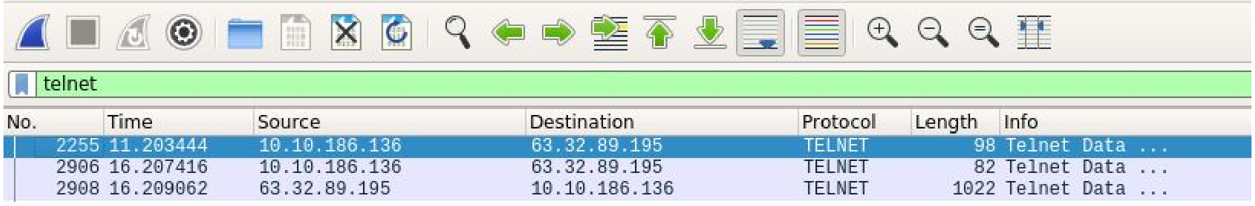
```
▶ Frame 1: 84 bytes on wire (672 bits), 84 bytes captured (672 bits)
▶ Ethernet II, Src: 02:c8:85:b5:5a:aa (02:c8:85:b5:5a:aa), Dst: 02:7e:2b:12:63:16 (02:7e:2b:12:63:16)
▶ Internet Protocol Version 4, Src: 34.255.145.244, Dst: 10.10.186.136
▶ Transmission Control Protocol, Src Port: 36710, Dst Port: 3389, Seq: 1, Ack: 1, Len: 18
▶ Data (18 bytes)

0000  02 7e 2b 12 63 16 02 c8  85 b5 5a aa 08 00 45 00   ·~+·c··· ··Z···E·
0010  00 46 9e ae 40 00 3f 06  23 7e 22 ff 91 f4 0a 0a   ·F··@·?· #~"·····
0020  ba 88 8f 66 0d 3d 9c 68  fe 9c d8 09 38 82 80 18   ···f·=·h ····8···
0030  2e 79 e3 0e 00 00 01 01  08 0a 53 c1 5d cc 39 56   .y······ ··S·]·9V
0040  49 0f c4 80 12 fc 94 d4  6b 2d c8 fa 94 97 d5 b1   I······· k-······
0050  6a 7d fa 2c                                        j}·,
```

Each dropdown section here is a layer in the OSI model. For example, we have the whole frame, then Ethernet (Data Link Layer - Layer 2), Internet Protocol Version 4 (Network Layer - Layer 3), Transmission Control Protocol (TCP - Transport Layer - Layer 4) and the data contained in the packet. Don’t worry if this doesn’t make sense yet. This is a gentle introduction

into network analysis. Read more about the OSI model here:
https://medium.com/software-engineering-roundup/the-osi-model-87e5adf35e10

We can apply filters to Wireshark to search for specific packets. We can filter by packet length, protocol, source or destination IP and more. Lets try searching for the telnet port to see if there is any telnet protocol packets.

In the filter field, type ‘telnet’.



telnet

No.	Time	Source	Destination	Protocol	Length	Info
2255	11.203444	10.10.186.136	63.32.89.195	TELNET	98	Telnet Data ...
2906	16.207416	10.10.186.136	63.32.89.195	TELNET	82	Telnet Data ...
2908	16.209062	63.32.89.195	10.10.186.136	TELNET	1022	Telnet Data ...

You should see 3 packets. If you look in the packet data view, you should see plain text data! In our case, it looks like its Linux commands being sent from the source 10.10.186.136 to 63.32.89.195.

To follow the complete stream of data, when the two machines where communicating, right click on the first packet shown, then click follow, then click TCP Stream. You should now be able to see all commands sent to the remote machine (shown in red) and the response from that server (shown in blue)!

Whoo! You have now analysed a pcap file to identify where someone was using telnet to execute commands on a remote machine!

Password Cracking

In the previous wireshark capture, we found some data a device send over telnet. You would have seen the command **cat /etc/shadow** - On a Linux system the shadow file contains all user account details. Its broken down into the following format:

    username:$hash_algorithm$hash_salt$hash_data:other_data..

In this challenge, we’re only really interested in the users hashed password. Firstly, what is a hash?

Hashing and encrypting are **not** the same. If you encrypt something, you can decrypt it again to get the original plain text data. With a hash, it only works one way. You can turn it into another not-human readable form and it cannot be reversed. With a hash, the only way to tell the value of the hash is, taking characters, hashing them and comparing them to see if both hashes are the same.

We can try and *Crack The Hash* by taking a word list, using a hashing algorithm and hashing each word from the list, comparing it to the original. If it’s the same hash, we have the word that was originally hashed, if its different we can move onto the next word to compare.

Let’s take an example hash and crack it together! Let's say we have the following data:

testuser:**$6$/5K3q7L0$XsNMzp37s0Q8/sAX0NXtQQjsy6a2f5tvKn2ZJSGWwE8uL9JLhXKpR7.pCbu/WoZa4LXIPYe7k18Z3Nohymk5T0:18233:0:99999:7:::**

Blue shows the username, the green shows all the hash information, and the red color shows the rest of the data.

Lets separate the user and the hash:

Linux username:

testuser

Hash Information:

$6$/5K3q7L0$XsNMzp37s0Q8/sAX0NXtQQjsy6a2f5tvKn2ZJSGWwE8uL9JLhXKpR7.pCbu/WoZa4LXIPYe7k18Z3Nohymk5T0

Using the first $6, we can look up what type of hash algorithm was used. To look this up, check this page: https://hashcat.net/wiki/doku.php?id=example_hashes and view the Hash-mode. We can search for $6 and see the hash-mode for Hashcat is 1800 and the type is using sha512crypt.

Hashcat is a a very popular password cracking tool. You can download it from the official hashcat website: https://hashcat.net/hashcat/**.** Alternatively, if you’re using the virtual Kali machine on TryHackMe it will be pre-installed for you - simply open up a terminal and type ‘hashcat’.

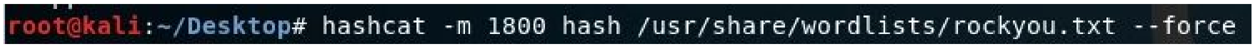
We now need a wordlist to hash and compare to the original, for this we’re going to use a password list called rockyou.txt. Download this here: https://github.com/brannondorsey/naive-hashcat/releases/download/data/rockyou.txt - again, if you’re using the virtual Kali machine, its already downloaded and can be found: /usr/share/wordlists (you will need to run gunzip /usr/share/wordlists/rockyou.txt.gz to decompress it).

First things first, lets save the Hash Information into a file (so, just copy all the green text under the *Hash Information* section above).

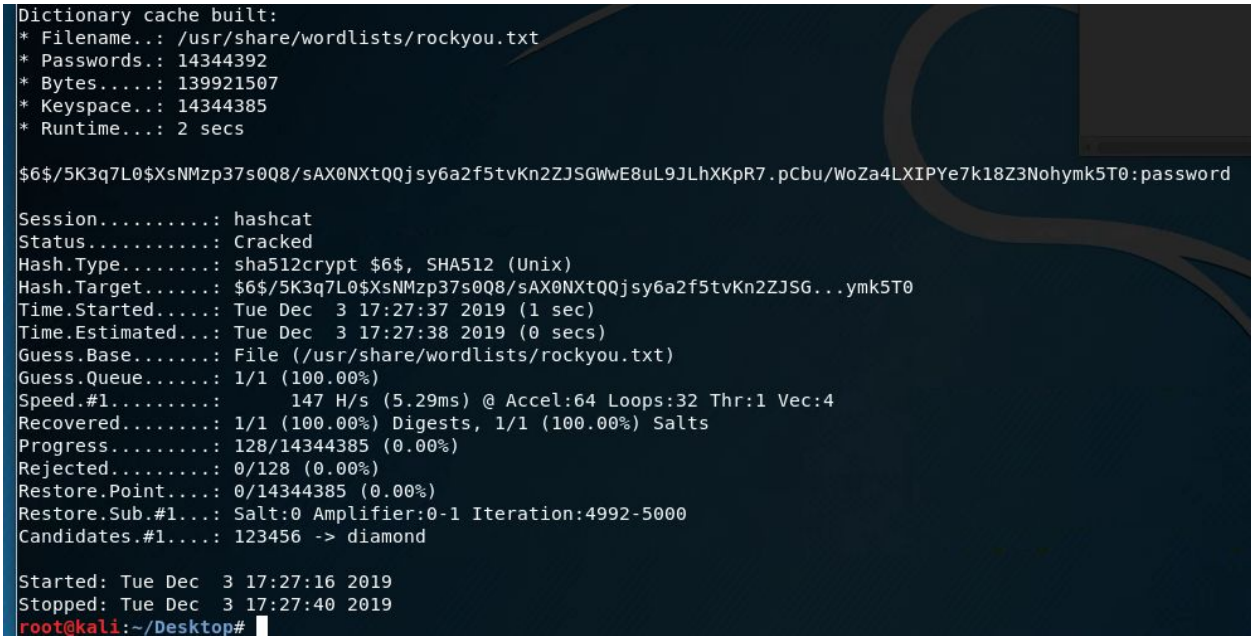
Now let’s crack this hash! We have our hash mode (1800), our hash data in a file and a wordlist (rockyou.txt).

Run the following command:

Hashcat -m 1800 <your hash file> <your list file>



```
root@kali:~/Desktop# hashcat -m 1800 hash /usr/share/wordlists/rockyou.txt --force
```



```
Dictionary cache built:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344392
* Bytes.....: 139921507
* Keyspace..: 14344385
* Runtime...: 2 secs

$6$/5K3q7L0$XsNMzp37s0Q8/sAX0NXtQQjsy6a2f5tvKn2ZJSGWwE8uL9JLhXKpR7.pCbu/WoZa4LXIPYe7k18Z3Nohymk5T0:password

Session..........: hashcat
Status...........: Cracked
Hash.Type........: sha512crypt $6$, SHA512 (Unix)
Hash.Target......: $6$/5K3q7L0$XsNMzp37s0Q8/sAX0NXtQQjsy6a2f5tvKn2ZJSG...ymk5T0
Time.Started.....: Tue Dec  3 17:27:37 2019 (1 sec)
Time.Estimated...: Tue Dec  3 17:27:38 2019 (0 secs)
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:      147 H/s (5.29ms) @ Accel:64 Loops:32 Thr:1 Vec:4
Recovered........: 1/1 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.........: 128/14344385 (0.00%)
Rejected.........: 0/128 (0.00%)
Restore.Point....: 0/14344385 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:4992-5000
Candidates.#1....: 123456 -> diamond

Started: Tue Dec  3 17:27:16 2019
Stopped: Tue Dec  3 17:27:40 2019
root@kali:~/Desktop#
```

You should eventually get the following output. We can see the hash and after the **:** we see it says password. To the testuser’s password was.. Password.

Use this principle to crack the hash you found in the network capture.

Is this realistic: network capture files are commonly used to identify attacker activity when a network hash been breached. Analysing aspects like the protocols they used, the commands they sent and even the IP addresses they use help us understand more about an attacker. While this example uses telnet, more sophisticated attackers would encrypt their traffic to prevent identification.