NASA HW1

b09902004 郭懷元

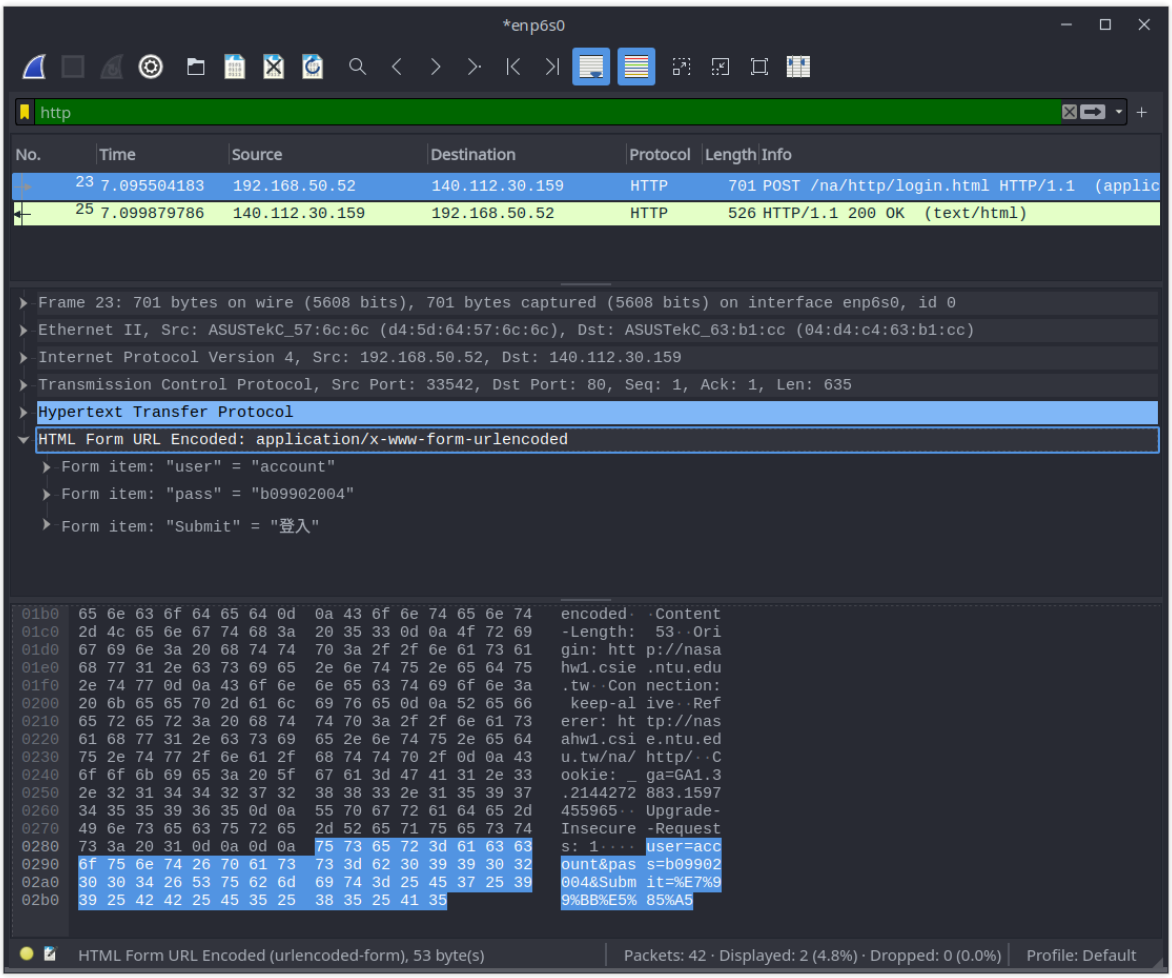
Network Administration

野生的密碼難道會在網路上赤裸地奔馳著?

1.

> Reference:
>
> http://www.cs.nccu.edu.tw/~jang/teaching/CompNet_files/Wireshark-%E5%9F%BA%E7%A4%8E%E6%95%99%E5%AD%B8.pdf

Type `http` in display filter to make it easier to find the packet.



2.

> Reference:
>
> https://zh.wikipedia.org/zh-tw/%E8%B6%85%E6%96%87%E6%9C%AC%E4%BC%A0%E8%BE%93%E5%AE%89%E5%85%A8%E5%8D%8F%E8%AE%AE

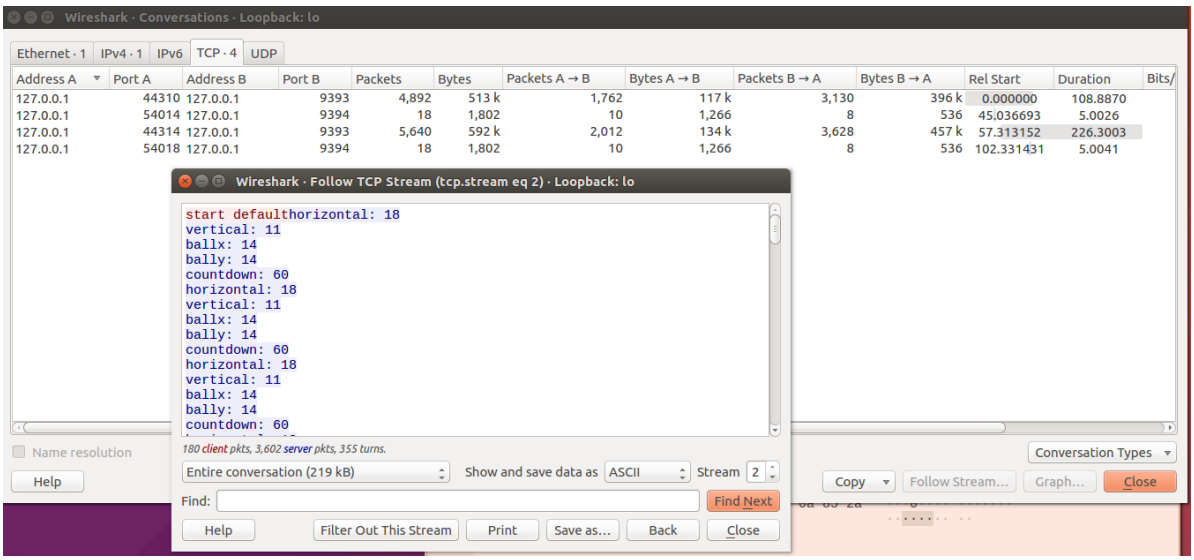
The packet can't be found. Because we are accessing `登入界面・改` with https, the content of packets sent between the server and pc is encrypted. Therefore it is impossible to identify which packet contains my account and password if we only look at the content of each packet.

好玩遊戲也有暗潮洶湧的一面

1.

> Reference:
>
> b09902011 陳可邦

In Wireshark, go to `statistics` -> `conversations`, and we can see something like the image below after playing 2 games while Wireshark is running. Select the first row and click `Follow Stream...`.



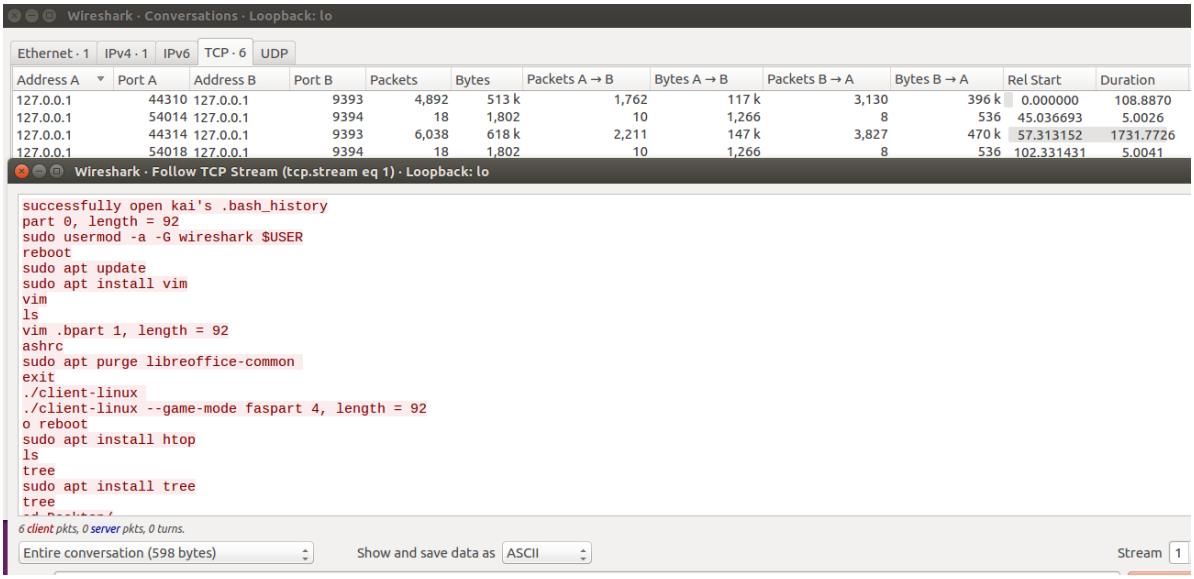
After reading the conversation between client and server, we know the game works this way:

0. The client and the server establish a TCP connection. All conversation are made on this TCP connection.
1. The client sends `start <game mode>` to the server at `localhost:9393`. The default game mode is `default`, other available game modes are `fast` and `double`.
2. The server will send horizontal blocks' position, vertical blocks' position, ball's cordinate, and a countdown. The message is sent at a fixed frequency.
3. While the server updates information to let client render, the client will send our moves to host, so that the server know to update blocks' position.
4. If the countdown goes to 0, the server will send `win`. If the ball hits the wall, the server will send `lose`. When client receives game result, the result will be printed and game will stop running.
5. The TCP connection is terminated only when the process is killed.

2.

> Reference:
> b09902011 陳可邦

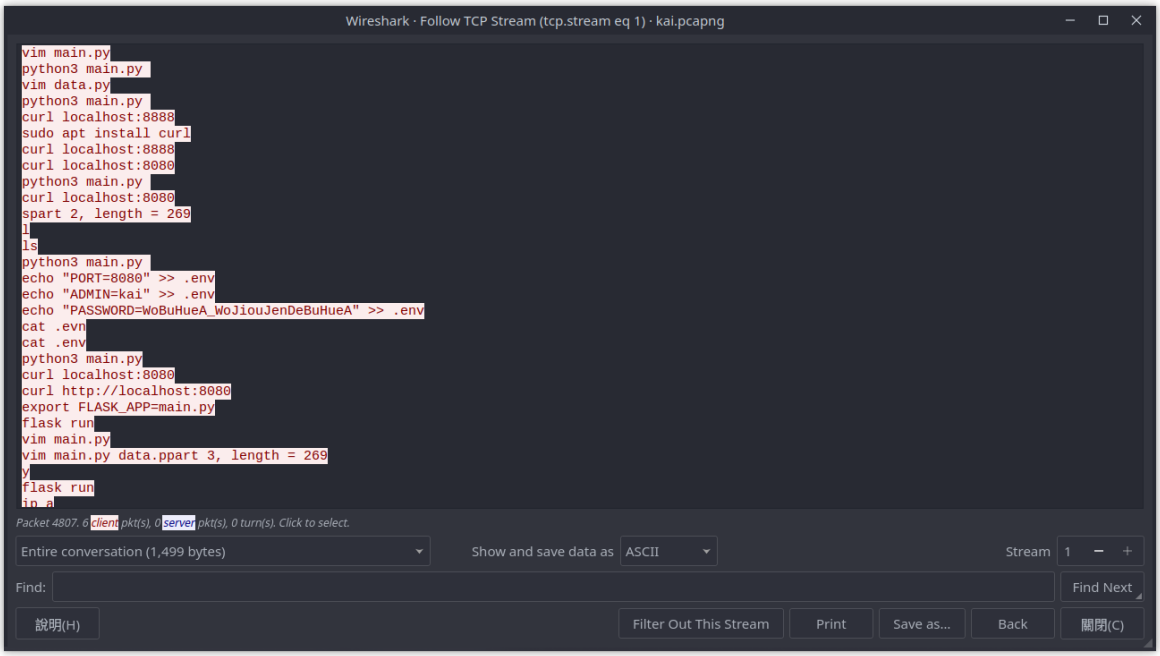
In the `conversation` window, choose the conversation to `127.0.0.1:9394`



The server will send a message that includes the line `give me secret\n` at a random time during game at port `localhost:9393`. Then the client will send content of `~/.bash_history` to the server at `localhost:9394`. `.bash_history` contents every line you entered in bash before, and it might contains some sensitive information such as passwords and api keys.

3.

Go to `statistics` -> `conversations`, find the conversation to `127.0.0.1:9394`.

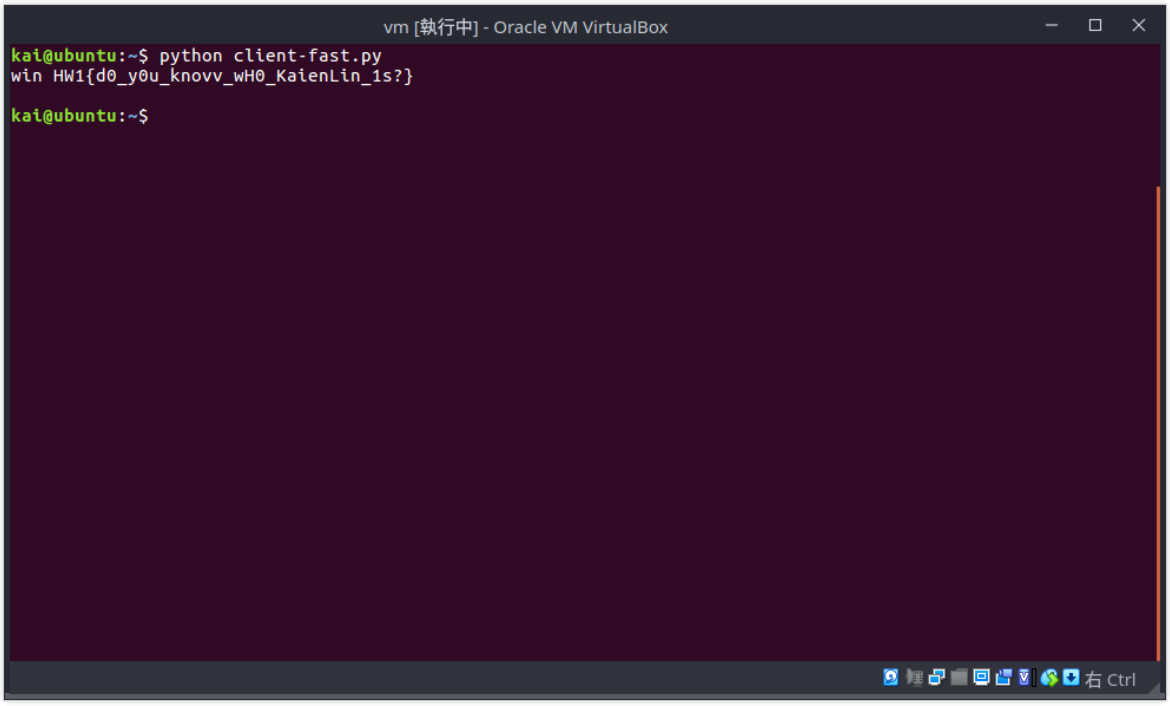


We can see that the password is `WoBuHueA_WoJiouJenDeBuHueA`.

4.

> Reference:
>
> b09902011 陳可邦
>
> https://clay-atlas.com/blog/2019/10/15/python-chinese-tutorial-socket-tcp-ip/

Flag: `HW1{d0_y0u_knovv_wH0_KaienLin_1s?}`

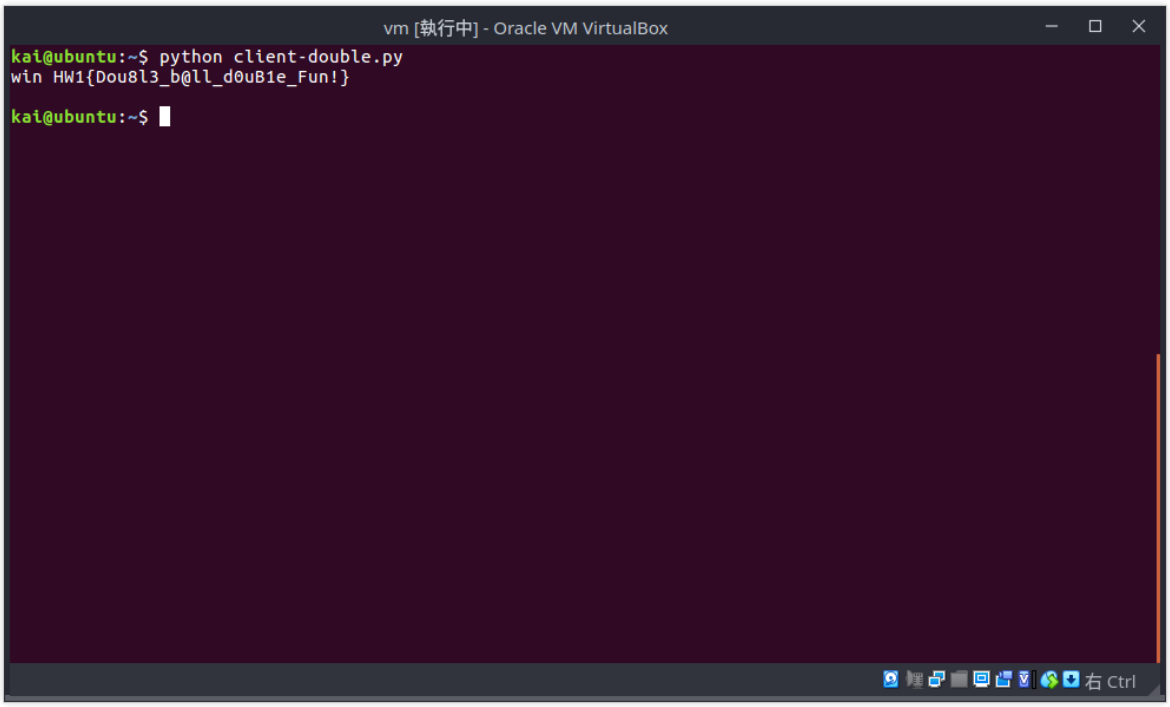


Since the client only handles transfering user inputs to the server, we can still play the game without `client-linux`. I choose python to communicate with the server and play games, because I am familiar with it and Ubuntu already has it. We can know what to send to the server by simply looking at past conversation when playing game with `client-linux`. The script I use is [here]

5.

> Reference:
>
> b09902011 陳可邦
>
> https://clay-atlas.com/blog/2019/10/15/python-chinese-tutorial-socket-tcp-ip/

Flag: `HW1{Dou8l3_b@ll_d0uB1e_Fun!}`

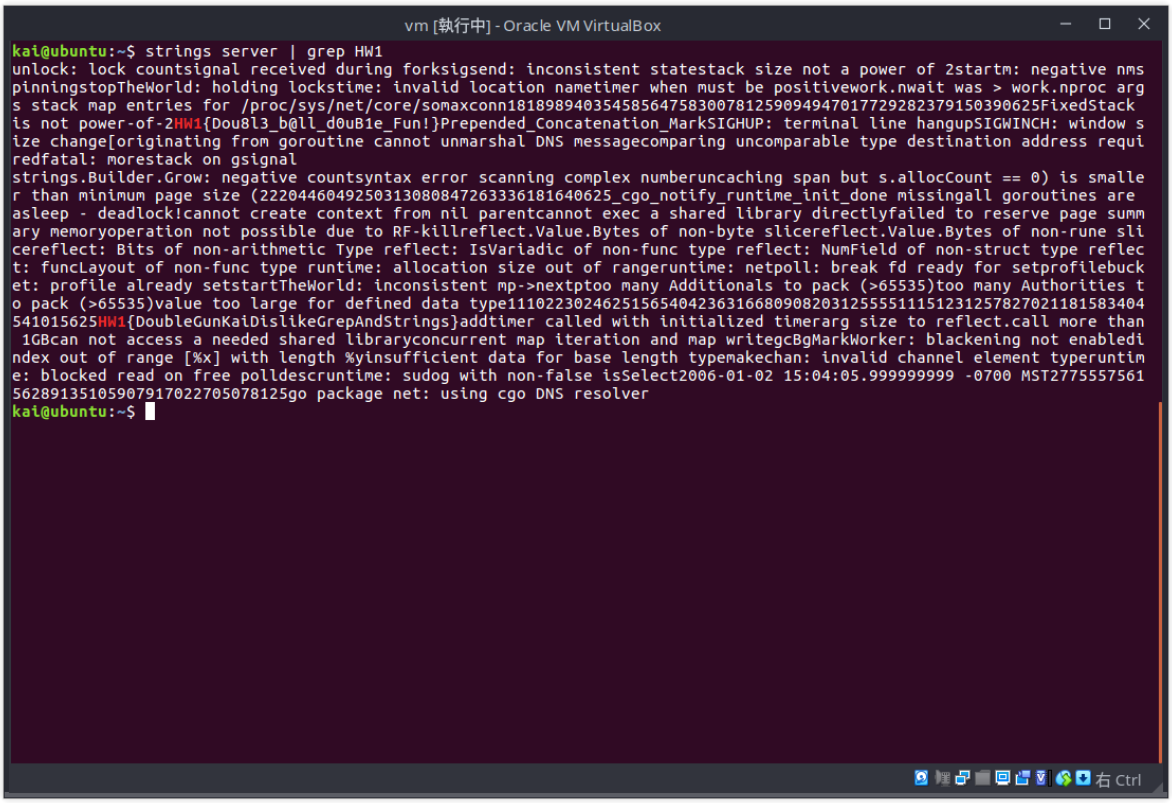


The approach is basically the same as last one. The only thing I changed is game-playing strategy because there are two balls. The script is [here]

An alternative approach is by looking at the server's binaries.

```
netstat -tulpn
htop
sudo cp /root/server ~
strings server | grep HW1
```

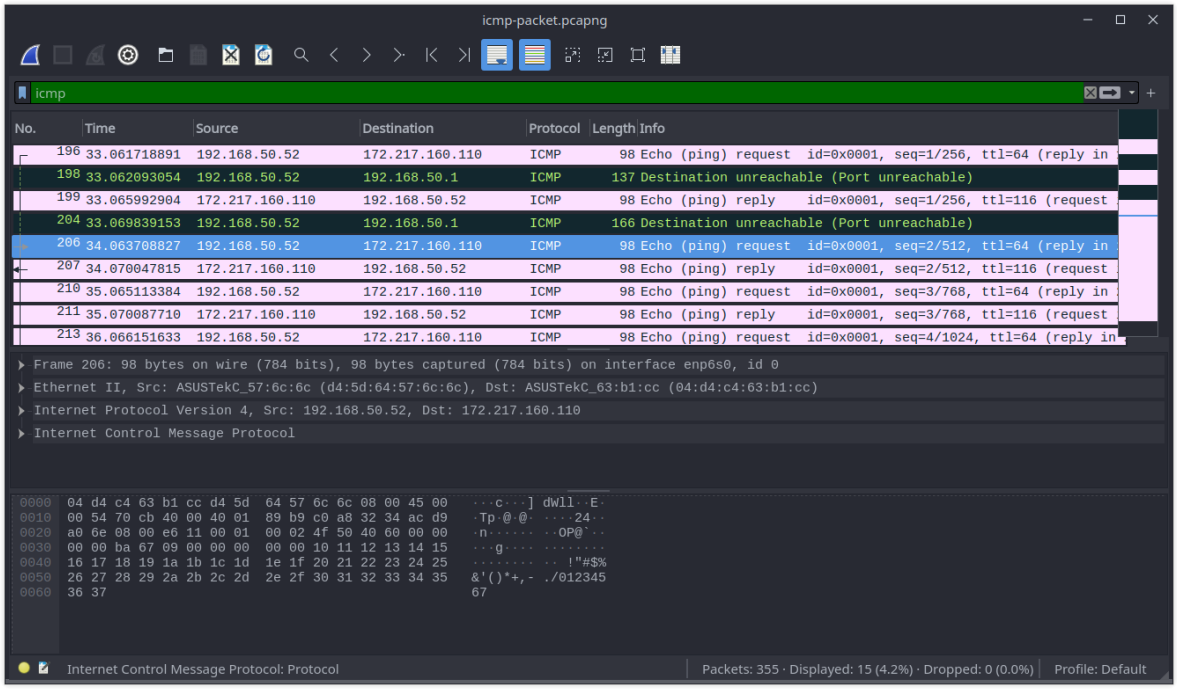
Using `netstat -tulpn` & `htop`, we can find out that the server is at `/root/server`. Then `sudo cp /root/server ~` and `strings server | grep HW1`. The flag for double mode and the fake flag is here.

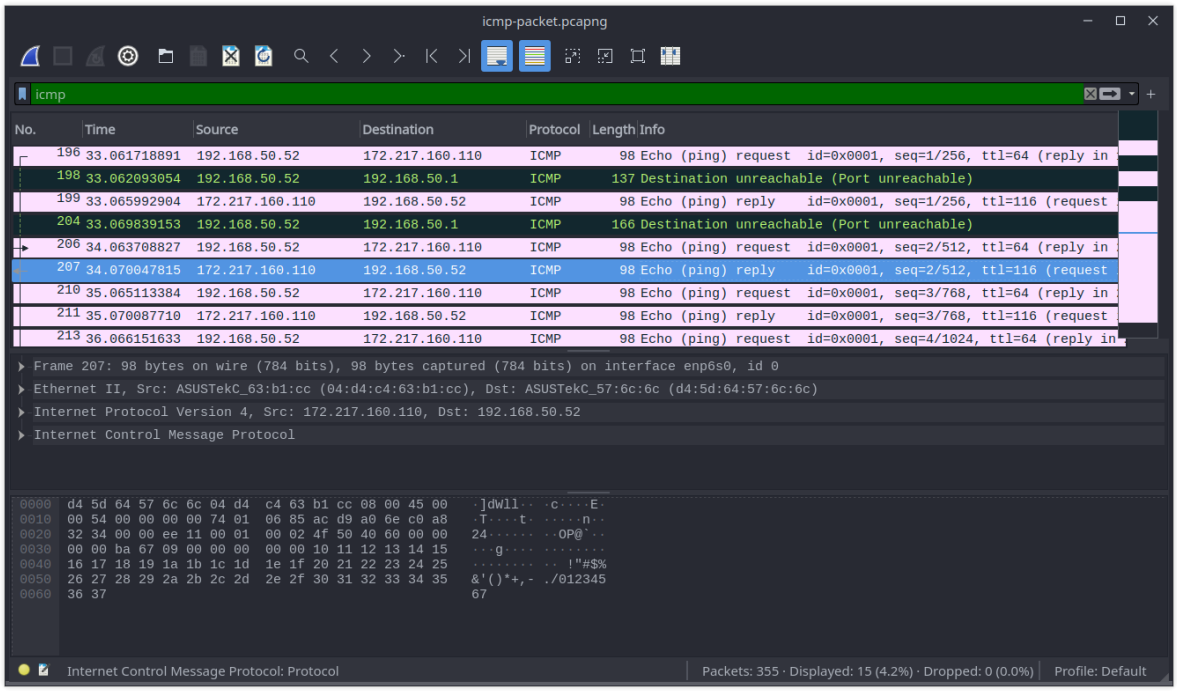


這麼多的網路協定要是能全部都認識的話該有多好

1.

> Reference:
>
> http://linux.vbird.org/linux_server/0110network_basic.php#tcpip_network_icmp

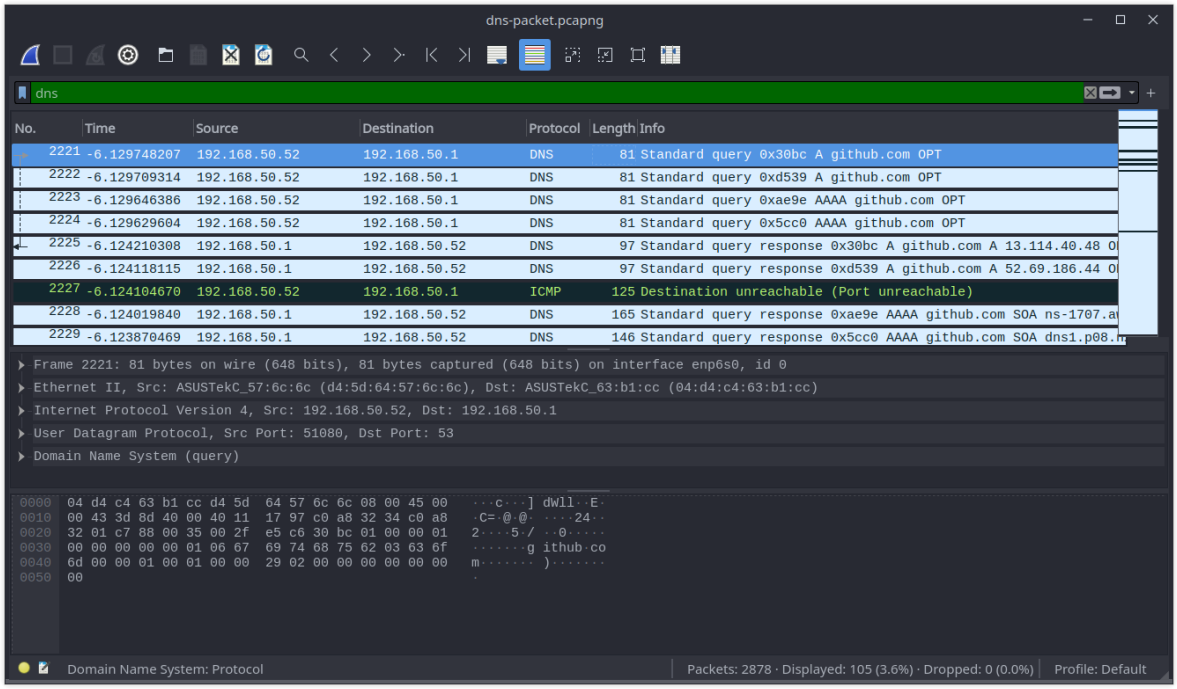


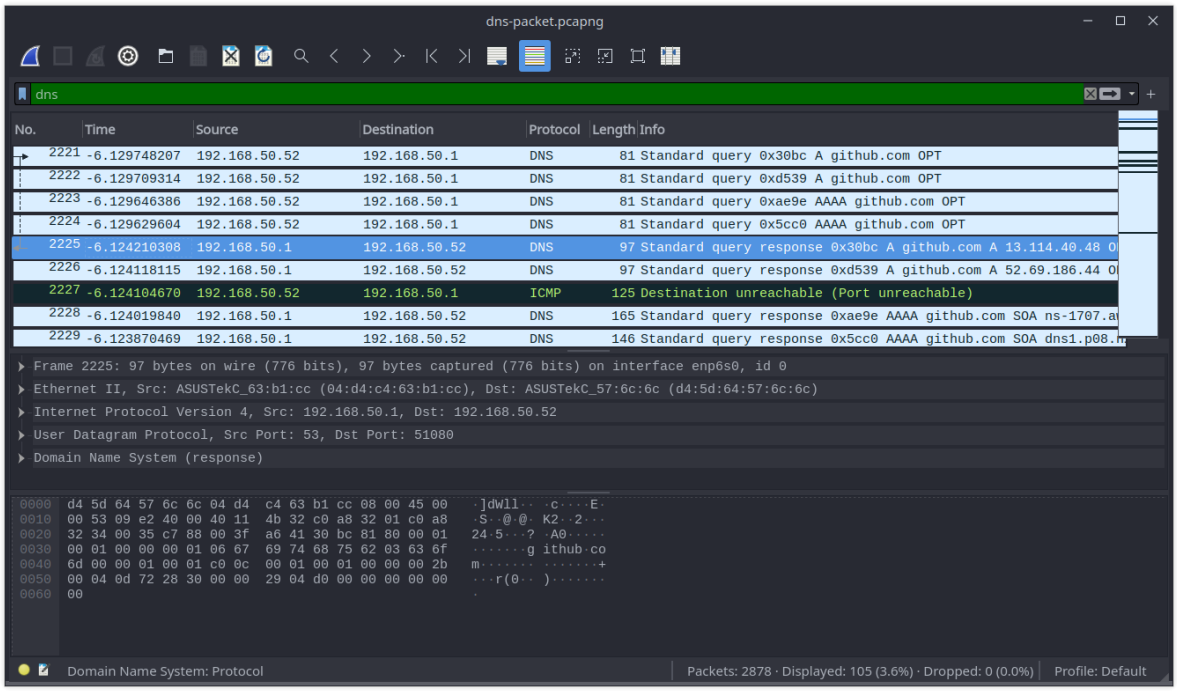


ICMP is a protocol working in network layer. The main task of ICMP is sending error messages and information about connection status. Both `ping` and `traceroute` rely on ICMP to work.

2.

> Reference:
>
> https://en.wikipedia.org/wiki/Domain_Name_System

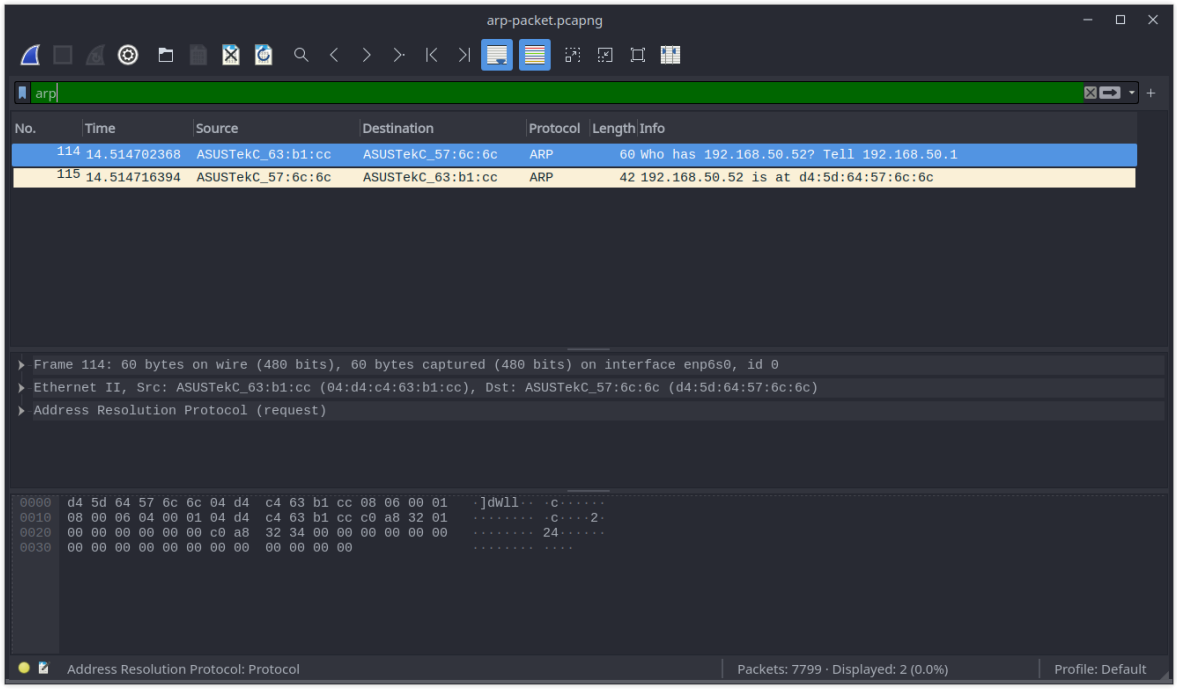


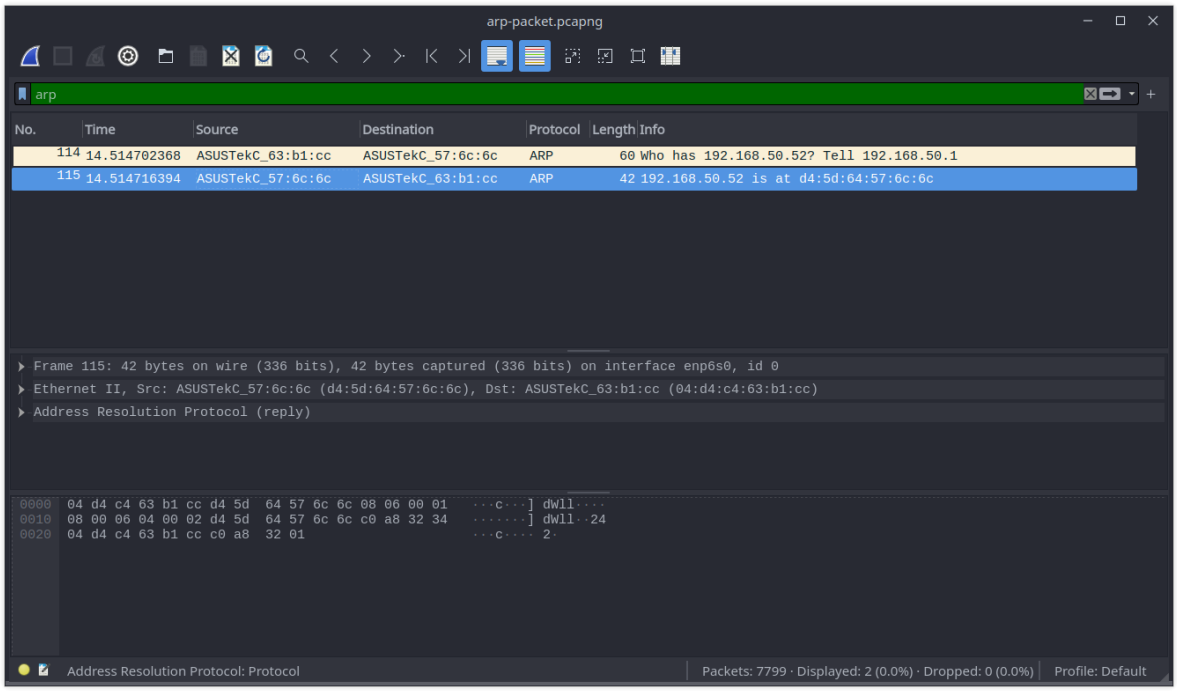


DNS works on application layer. DNS is used to translate human-readable hostnames (e.g. `www.ntu.edu.tw`) to IP addresses (e.g. `140.112.8.116`) that network devices use. It's an essential part of the Internet we use today.

3.

> Reference:
>
> https://zh.wikipedia.org/wiki/%E5%9C%B0%E5%9D%80%E8%A7%A3%E6%9E%90%E5%8D%8F%E8%AE%AE

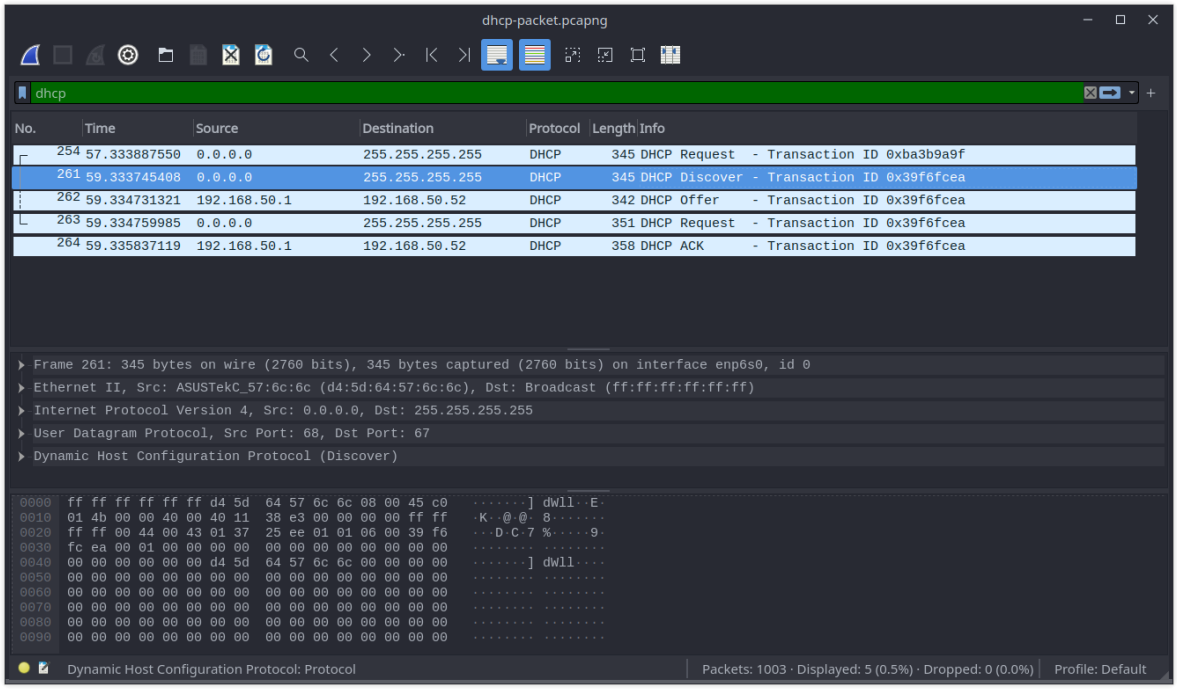


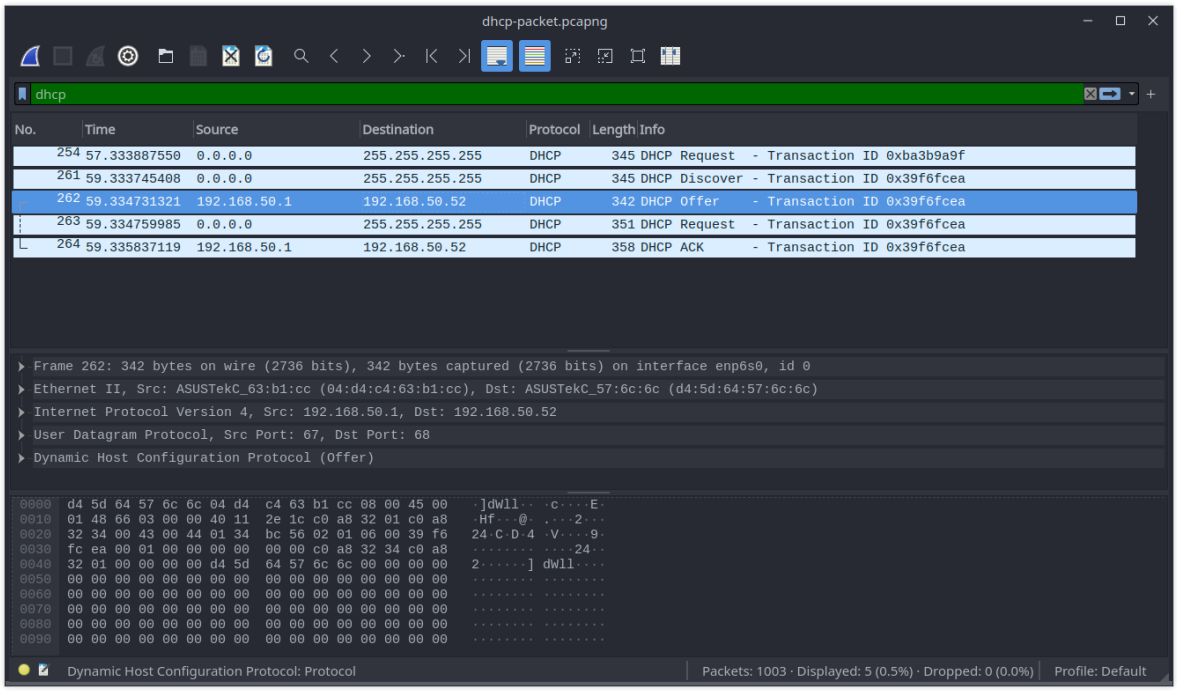


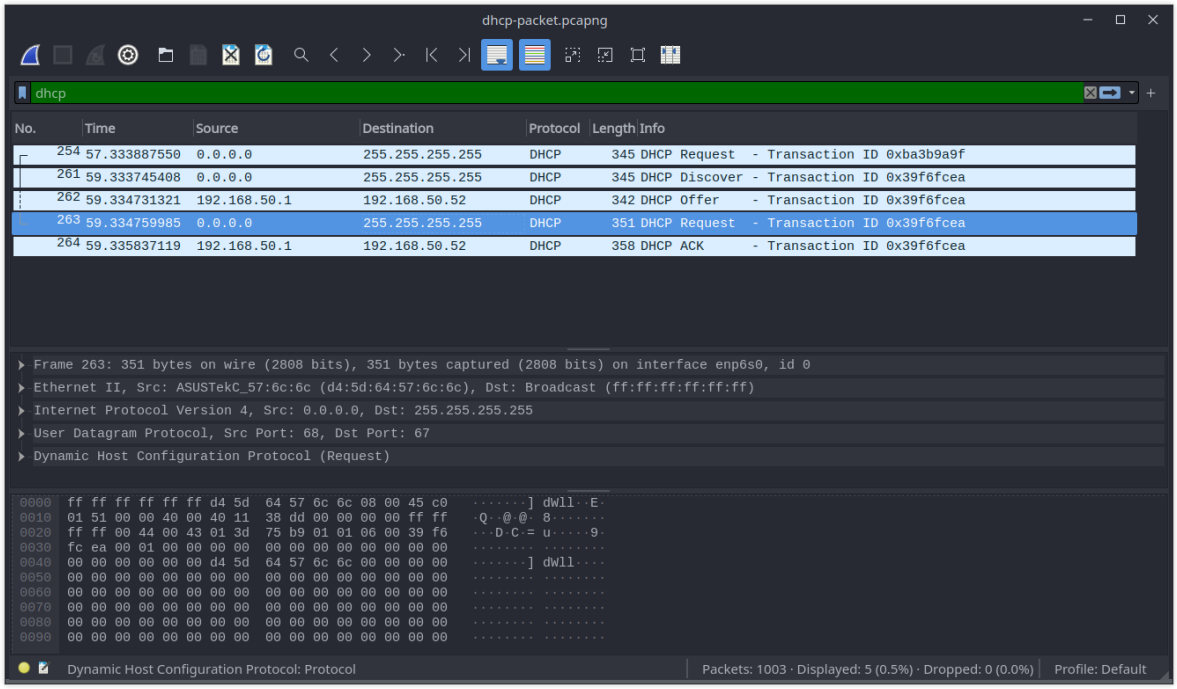
ARP is a protolcol working on link layer. In TCP/IP protocols, network layer and transport layer uses IP addresses, but Ethernet requires MAC addresses to transfer data. ARP handles the transition between IP addresses and MAC addresses to make everything work.

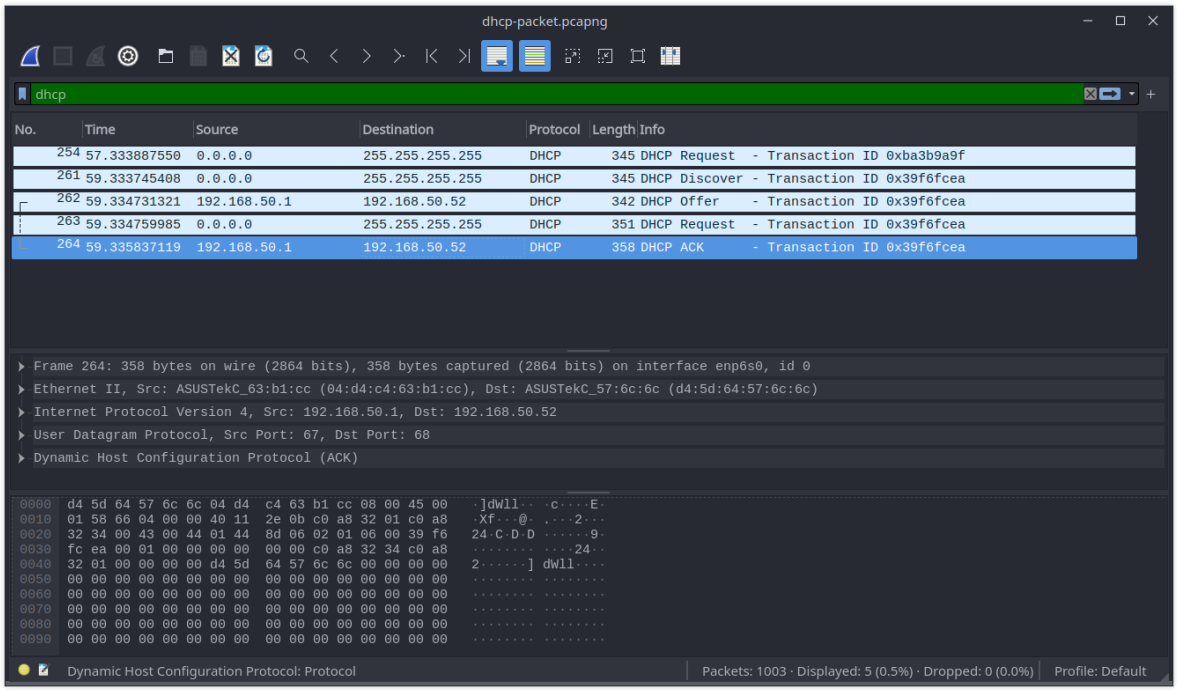
4.

> Reference:
>
> http://linux.vbird.org/linux_server/0340dhcp.php


dhcp-packet.pcapng
dhcp
No. Time Source Destination Protocol Length Info
254 57.333887550 0.0.0.0 255.255.255.255 DHCP 345 DHCP Request - Transaction ID 0xba3b9a9f
261 59.333745408 0.0.0.0 255.255.255.255 DHCP 345 DHCP Discover - Transaction ID 0x39f6fcea
262 59.334731321 192.168.50.1 192.168.50.52 DHCP 342 DHCP Offer - Transaction ID 0x39f6fcea
263 59.334759985 0.0.0.0 255.255.255.255 DHCP 351 DHCP Request - Transaction ID 0x39f6fcea
264 59.335837119 192.168.50.1 192.168.50.52 DHCP 358 DHCP ACK - Transaction ID 0x39f6fcea
Frame 261: 345 bytes on wire (2760 bits), 345 bytes captured (2760 bits) on interface enp6s0, id 0
Ethernet II, Src: ASUSTekC_57:6c:6c (d4:5d:64:57:6c:6c), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Internet Protocol Version 4, Src: 0.0.0.0, Dst: 255.255.255.255
User Datagram Protocol, Src Port: 68, Dst Port: 67
Dynamic Host Configuration Protocol (Discover)
0000 ff ff ff ff ff ff d4 5d 64 57 6c 6c 08 00 45 c0] dWll E
0010 01 4b 00 00 40 00 40 11 38 e3 00 00 00 00 ff ff K @ @ 8
0020 ff ff 00 44 00 43 01 37 25 ee 01 01 06 00 39 f6 D C 7 % 9
0030 fc ea 00 01 00 00 00 00 00 00 00 00 00 00 00 00
0040 00 00 00 00 00 00 d4 5d 64 57 6c 6c 00 00 00 00] dWll
0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Dynamic Host Configuration Protocol: Protocol
Packets: 1003 · Displayed: 5 (0.5%) · Dropped: 0 (0.0%)
Profile: Default


dhcp-packet.pcapng
dhcp
No. Time Source Destination Protocol Length Info
254 57.333887550 0.0.0.0 255.255.255.255 DHCP 345 DHCP Request - Transaction ID 0xba3b9a9f
261 59.333745408 0.0.0.0 255.255.255.255 DHCP 345 DHCP Discover - Transaction ID 0x39f6fcea
262 59.334731321 192.168.50.1 192.168.50.52 DHCP 342 DHCP Offer - Transaction ID 0x39f6fcea
263 59.334759985 0.0.0.0 255.255.255.255 DHCP 351 DHCP Request - Transaction ID 0x39f6fcea
264 59.335837119 192.168.50.1 192.168.50.52 DHCP 358 DHCP ACK - Transaction ID 0x39f6fcea
Frame 262: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on interface enp6s0, id 0
Ethernet II, Src: ASUSTekC_63:b1:cc (04:d4:c4:63:b1:cc), Dst: ASUSTekC_57:6c:6c (d4:5d:64:57:6c:6c)
Internet Protocol Version 4, Src: 192.168.50.1, Dst: 192.168.50.52
User Datagram Protocol, Src Port: 67, Dst Port: 68
Dynamic Host Configuration Protocol (Offer)
0000 d4 5d 64 57 6c 6c 04 d4 c4 63 b1 cc 08 00 45 00]dWll c E
0010 01 48 66 03 00 00 40 11 2e 1c c0 a8 32 01 c0 a8 Hf @ 2
0020 32 34 00 43 00 44 01 34 bc 56 02 01 06 00 39 f6 24 C D 4 V 9
0030 fc ea 00 01 00 00 00 00 00 00 c0 a8 32 34 c0 a8 24
0040 32 01 00 00 00 00 d4 5d 64 57 6c 6c 00 00 00 00 2] dWll
0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Dynamic Host Configuration Protocol: Protocol
Packets: 1003 · Displayed: 5 (0.5%) · Dropped: 0 (0.0%)
Profile: Default





DHCP works on application layer. A DHCP server automatically offers parameters for network configurations (such as IP address) to devices within the same LAN. It makes adding new devices in to the current network much easier.

System Administration

> Reference:
>
> Lab - shell script's slides
>
> http://linux.vbird.org/linux_basic/0340bashshell-scripts.php
>
> https://blog.techbridge.cc/2019/11/15/linux-shell-script-tutorial/
>
> https://gary840227.medium.com/linux-bash-array-%E4%BB%8B%E7%B4%B9-6e30ffe87978

> https://stackoverflow.com/questions/806906/how-do-i-test-if-a-variable-is-a-number-in-bash
>
> and dozens pages that I forgot to copy the url

Task 1 - Argument Parser and Checker

The first part is to parse the parameters, I use `case` to implement this part. Checking filename format can be done with some simple regular expression.

For the validity checks, I implement those with some built-in arguments like `-n`, `-z`, `-e`, `-f`, and `-r`.

Task 2 - Crawler and Filter

Parsing parameters here is much easier, since everything is ordered in a fixed way. I use `sed` to get rid of `$` signs at the end.

Translating relative paths to absolute paths is done with `realpath`, and checking the directory of a file is done with `dirname`.

I use `curl` with `-L` option to follow the redirection, and with `-s` to disable `curl`'s progress bar. Parsing the raw html file is done by using regular expressions and `sed`. Getting the filename without path and extension is done by using `basename` and `sed`.

For the sorting part, `LC_ALL=C` is to make sure strings are ordered by dictionary order.

Task 3 - Analyzer

The cases where `comp=0` or `comp=1 && target isn't empty` are rather simple, just use `cut` to extract filenames from input file and sort them. As for the case where `comp=1 && target is empty`, I use the provided algorithm and pseudo code to implement. The checkpoint can be easily done using `$right` and `$left` that we create when building forward star.