

HW1

b09902004 資工三 郭懷元

UDP Packets

Wireshark packet capture showing DNS traffic on wlan0. The packet list shows a series of DNS queries and responses. The packet details pane shows the structure of a DNS query packet. The packet bytes pane shows the raw hex and ASCII data.

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|-------------|---------------|---------------|----------|--------|--|
| 5 | 0.184113716 | 10.131.225.46 | 140.112.254.4 | DNS | 83 | Standard query 0x9832 A dailymix-images.scdn.co |
| 6 | 0.184137471 | 10.131.225.46 | 140.112.254.4 | DNS | 83 | Standard query 0xdb36 AAAA dailymix-images.scdn.co |
| 7 | 0.184508400 | 10.131.225.46 | 140.112.254.4 | DNS | 83 | Standard query 0xea9 A api-partner.spotify.com |
| 8 | 0.184520924 | 10.131.225.46 | 140.112.254.4 | DNS | 83 | Standard query 0x8af AAAA api-partner.spotify.com |
| 9 | 0.186915363 | 140.112.254.4 | 10.131.225.46 | DNS | 192 | Standard query response 0x8af AAAA api-partner.spotify.com CNAME p... |
| 10 | 0.186915638 | 140.112.254.4 | 10.131.225.46 | DNS | 180 | Standard query response 0xea9 A api-partner.spotify.com CNAME part... |
| 12 | 0.193067243 | 10.131.225.46 | 140.112.254.4 | DNS | 82 | Standard query 0x872 A newjams-images.scdn.co |
| 13 | 0.193086908 | 10.131.225.46 | 140.112.254.4 | DNS | 82 | Standard query 0xba8e AAAA newjams-images.scdn.co |
| 15 | 0.193796610 | 140.112.254.4 | 10.131.225.46 | DNS | 142 | Standard query response 0x9832 A dailymix-images.scdn.co CNAME scdn... |
| 16 | 0.196081733 | 140.112.254.4 | 10.131.225.46 | DNS | 277 | Standard query response 0x872 A newjams-images.scdn.co CNAME scdn... |
| 23 | 0.228221292 | 140.112.254.4 | 10.131.225.46 | DNS | 289 | Standard query response 0xba8e AAAA newjams-images.scdn.co CNAME sc... |
| 61 | 0.220608310 | 140.112.254.4 | 10.131.225.46 | DNS | 154 | Standard query response 0xdb36 AAAA dailymix-images.scdn.co CNAME s... |

> Frame 5: 83 bytes on wire (664 bits), 83 bytes captured (664 bits) on interface wlan0, id 0

> Ethernet II, Src: IntelCor_cb:72:35 (bc:17:b8:cb:72:35), Dst: ArubaaHe_04:42:60 (00:1a:1e:04:42:60)

> Internet Protocol Version 4, Src: 10.131.225.46, Dst: 140.112.254.4

> User Datagram Protocol, Src Port: 42964, Dst Port: 53

> Domain Name System (query)

0000 00 1a 1e 04 42 60 bc 17 b8 cb 72 35 08 00 45 00 ...B...r5..E

0010 00 45 5c 45 40 00 40 11 68 3c 0a 83 e1 2e 8c 70 ...EVE@.h<...p

0020 fe 04 a7 d4 00 35 00 31 76 69 98 32 01 00 00 015!v!2...

0030 00 00 00 00 00 0f 64 61 69 6c 79 6d 69 78 2ddailymix-

0040 69 6d 61 67 65 73 04 73 63 64 6e 02 63 6f 00 00 images s cdn co...

0050 01 00 01 ...

Domain Name System: Protocol Packets: 2347 · Displayed: 40 (1.7%) · Dropped: 0 (0.0%) Profile: Default

- Server address: 140.112.254.4
- Service: DNS (Domain Name System)
- Port number used: 53 on server side, 42964 on client side

TCP Packets

The image shows a Wireshark packet capture of an SSH session. The top pane displays a list of packets, with packet 23 selected. The middle pane shows the details of packet 23, which is an SSH Protocol packet. The bottom pane shows the raw packet data in hexadecimal and ASCII.

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|-------------|---------------|---------------|----------|--------|--|
| 23 | 7.641476930 | 10.131.225.46 | 140.112.30.36 | SSHv2 | 87 | Client: Protocol (SSH-2.0-OpenSSH_9.0) |
| 25 | 7.651446743 | 140.112.30.36 | 10.131.225.46 | SSHv2 | 87 | Server: Protocol (SSH-2.0-OpenSSH_9.0) |
| 27 | 7.652231660 | 10.131.225.46 | 140.112.30.36 | SSHv2 | 1570 | Client: Key Exchange Init |
| 28 | 7.653543800 | 140.112.30.36 | 10.131.225.46 | SSHv2 | 1146 | Server: Key Exchange Init |
| 31 | 7.774848859 | 10.131.225.46 | 140.112.30.36 | SSHv2 | 1274 | Client: Diffie-Hellman Key Exchange Init |
| 32 | 7.801832807 | 140.112.30.36 | 10.131.225.46 | SSHv2 | 1630 | Server: Diffie-Hellman Key Exchange Reply, New Keys, Encrypted pack... |
| 34 | 9.089944584 | 10.131.225.46 | 140.112.30.36 | SSHv2 | 82 | Client: New Keys |
| 36 | 9.141939888 | 10.131.225.46 | 140.112.30.36 | SSHv2 | 110 | Client: Encrypted packet (len=44) |
| 38 | 9.143941294 | 140.112.30.36 | 10.131.225.46 | SSHv2 | 110 | Server: Encrypted packet (len=44) |
| 40 | 9.144197794 | 10.131.225.46 | 140.112.30.36 | SSHv2 | 134 | Client: Encrypted packet (len=68) |
| 41 | 9.153372795 | 140.112.30.36 | 10.131.225.46 | SSHv2 | 118 | Server: Encrypted packet (len=52) |
| 42 | 9.153657740 | 10.131.225.46 | 140.112.30.36 | SSHv2 | 556 | Client: Encrypted packet (len=500) |

Frame 23: 87 bytes on wire (696 bits), 87 bytes captured (696 bits) on interface wlan0, id 0
> Ethernet II, Src: IntelCor_cb:72:35 (bc:17:b8:cb:72:35), Dst: ArubaaHe_04:42:60 (00:1a:1e:04:42:60)
> Internet Protocol Version 4, Src: 10.131.225.46, Dst: 140.112.30.36
> Transmission Control Protocol, Src Port: 49810, Dst Port: 22, Seq: 1, Ack: 1, Len: 21
> SSH Protocol

```
0000  00 1a 1e 04 42 60 bc 17 b8 cb 72 35 08 00 45 48  .B...r5..EH
0010  00 49 d8 d4 40 00 40 06 cb 4c 0a 83 e1 2e 8c 70  .I-@...L...p
0020  1e 24 c2 92 00 16 04 49 8c b6 80 34 b0 cb 80 18  .$. ....I...4....
0030  01 f6 96 81 00 00 01 01 08 0a c4 29 06 be 1b e0  .....-)...
0040  09 f1 53 53 48 2d 32 2e 30 2d 4f 70 65 6e 53 53  .-SSH-2. 0-OpenSSH
0050  48 5f 39 2e 30 0d 0a  H_9.0..
```

- SSH server: linux5.csie.ntu.edu.tw at 140.112.30.36
- SSH server port: 22
- My machine used private IP address in the tcp packets, because the subnet 10.0.0.0/8 is reserved for private network in IPv4, and my source ip address is 10.131.225.46

TCP & UDP Comparison

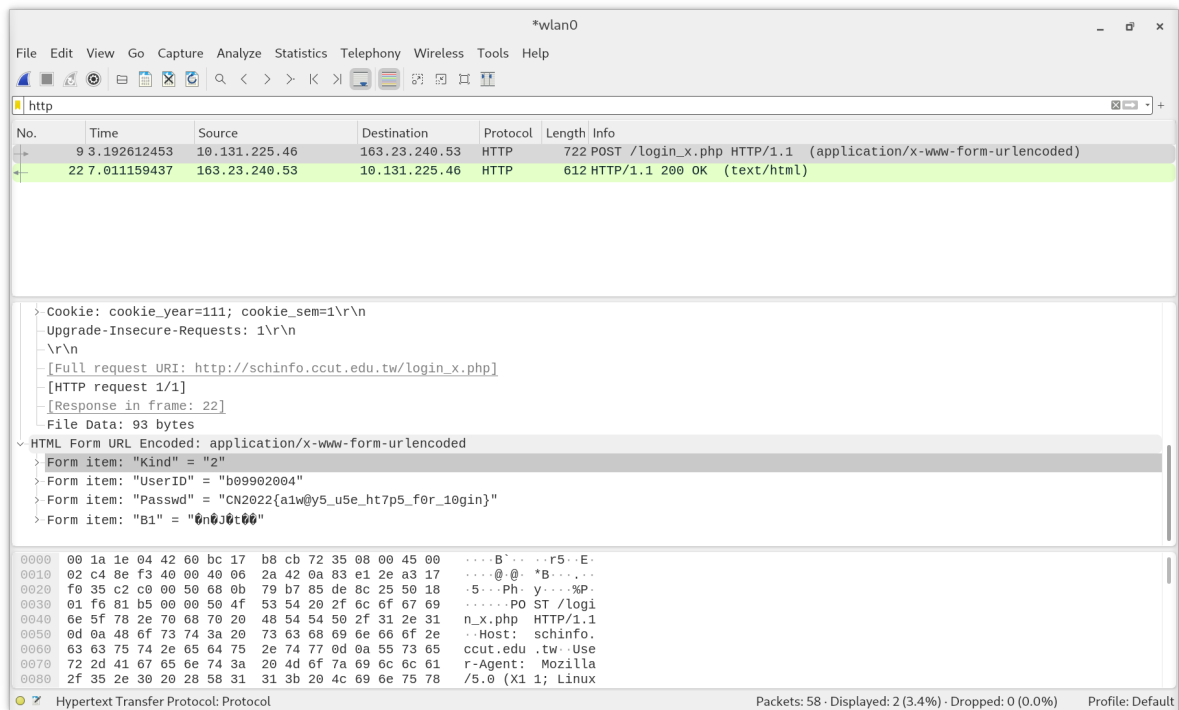
Same fields

- Source port and destination port
- Checksum

Different fields

- Header Length (only in TCP)
- Sequence number (only in TCP)
- Acknowledgement number (only in TCP)

Plaintext Password in Packets

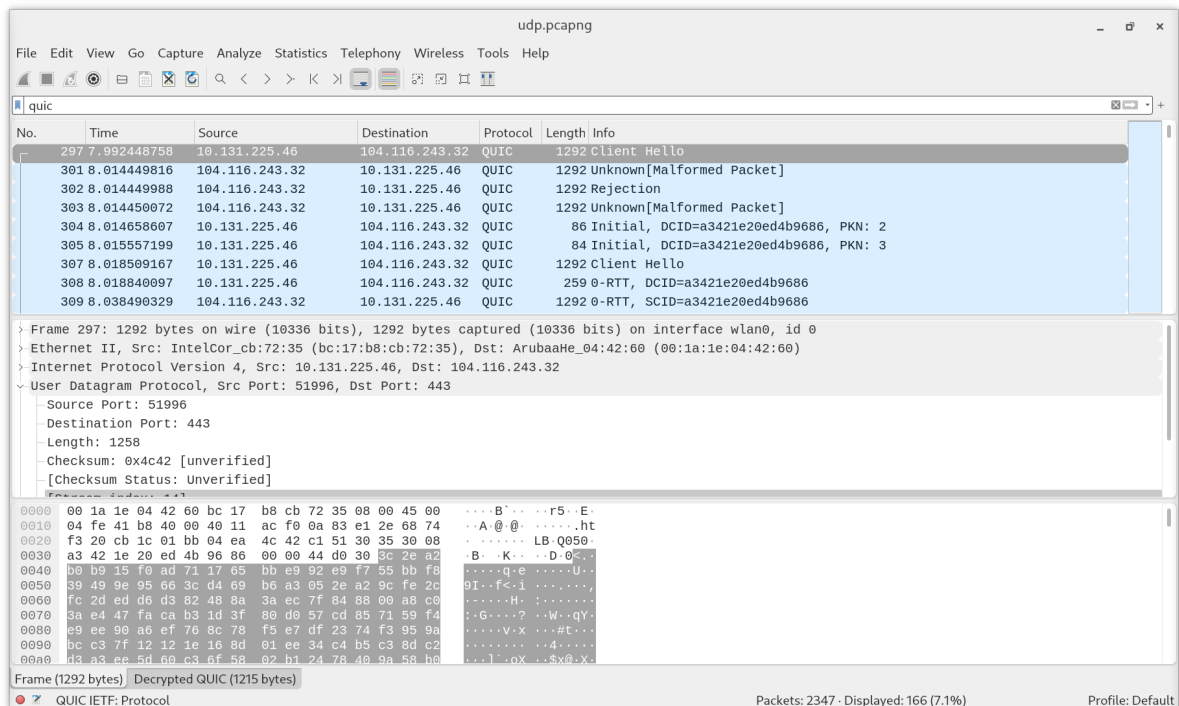


- Website: [中州科技大學校園網路資訊系統](#)
- If an attacker can capture packets going through their campus's router, then any user of this website that is connected to the router would have their passwords leaked.

Other Discoveries

Refs:

- <https://en.wikipedia.org/wiki/QUIC>
- <https://en.wikipedia.org/wiki/HTTP/3>



I found a protocol called "QUIC" that I hadn't heard of. It's a relatively new transport layer protocol running on top of UDP. It aims to improve performance of web apps that uses TCP now.

QUIC is used in the newly proposed HTTP/3. Previous versions of HTTP uses TCP as the transport layer protocol.