Packet Analysis Using Wireshark

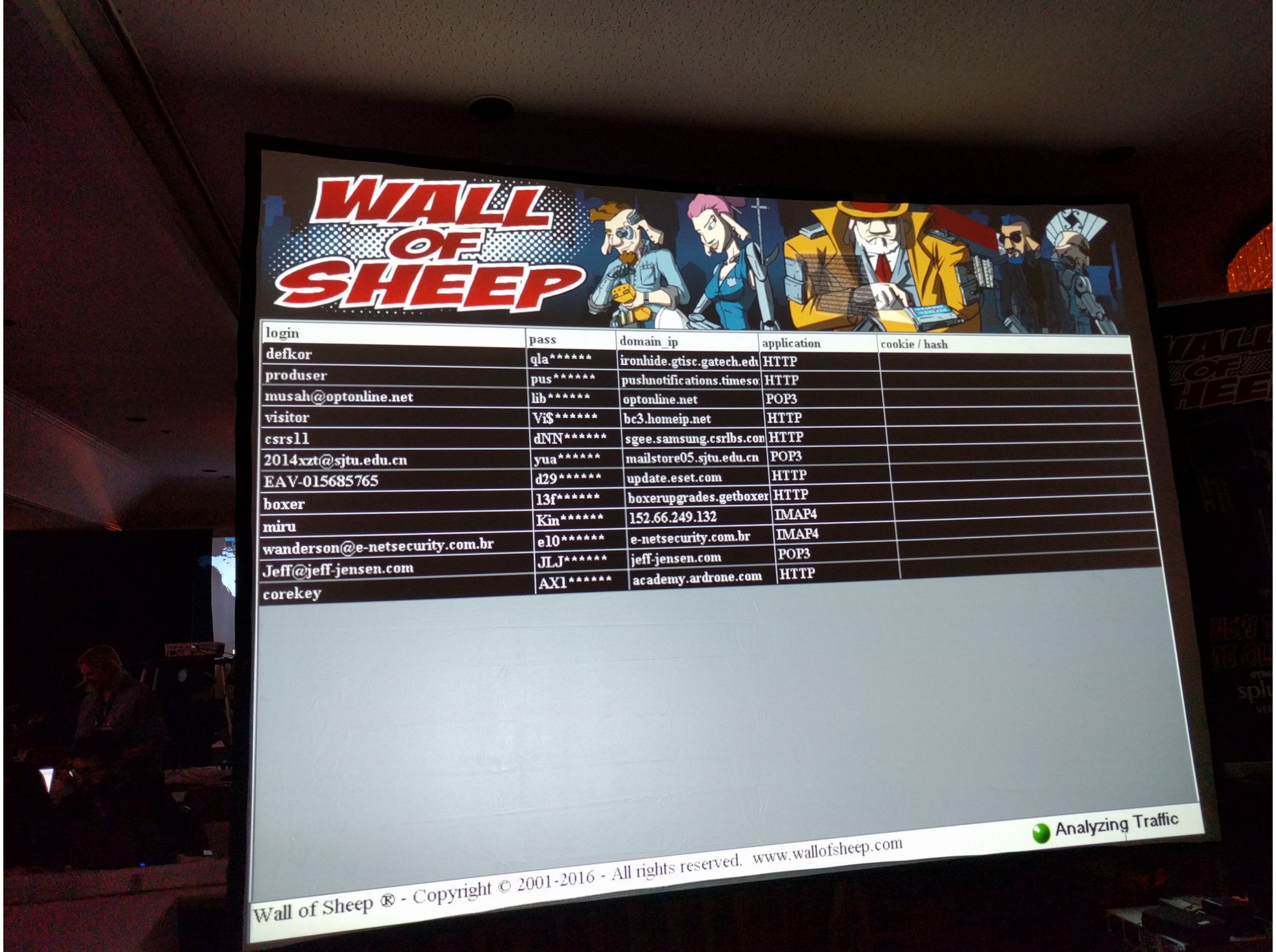
Ming Chow (ming@wallofsheep.com)

Wall of Sheep and the Packet Hacking Village

Twitter: @wallofsheep, @0xmchow

About the Wall of Sheep and the Packet Hacking Village

- Our mission: security awareness
- How we accomplish our mission: interactive demonstrations, unconventional methods
- Our team: all volunteers



login	pass	domain_ip	application	cookie / hash
defkor	qla******	ironhide.gtisc.gatech.edu	HTTP	
produser	pus******	pushnotifications.timeso	HTTP	
musah@optonline.net	lib******	optonline.net	POP3	
visitor	Vi$******	bc3.homeip.net	HTTP	
csrs11	dNN******	sgee.samsung.csrlbs.con	HTTP	
2014xzt@sjtu.edu.cn	yua******	mailstore05.sjtu.edu.cn	POP3	
EAV-015685765	d29******	update.eset.com	HTTP	
boxer	13f******	boxerupgrades.getboxer	HTTP	
miru	Kin******	152.66.249.132	IMAP4	
wanderson@e-netsecurity.com.br	e10******	e-netsecurity.com.br	IMAP4	
Jeff@jeff-jensen.com	JLJ******	jeff-jensen.com	POP3	
corekey	AX1******	academy.ardrone.com	HTTP	

Wall of Sheep ® - Copyright © 2001-2016 - All rights reserved. www.wallofsheep.com

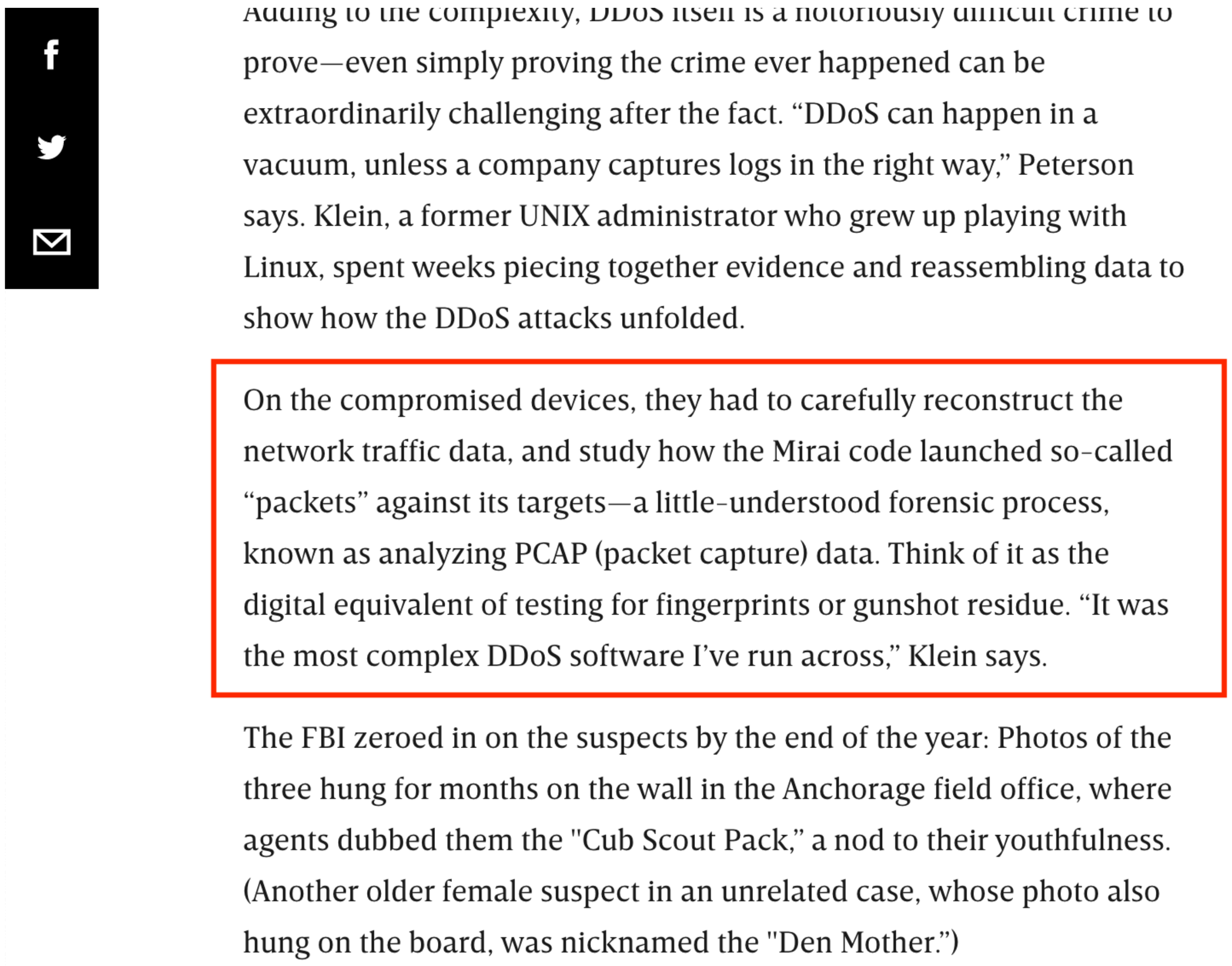
What is *Packet* Analysis?

- Look at and understand network traffic
- Also known as analyzing packets, also known as network traffic analysis, also known as packet sniffing, also known as protocol analysis, also known as packet tracing

Why Packet Analysis?

- Troubleshoot networking issues
- Record communications (e.g., email, voice, chat)
- Record and analyze web traffic
- Reconstruct images and other data transmitted on network
- Catch usernames and passwords, personal information, and other sensitive information that were sent insecurely, in plaintext

Source: https://www.wired.com/story/mirai-botnet-minecraft-scam-brought-down-the-internet/



Adding to the complexity, DDoS itself is a notoriously difficult crime to prove—even simply proving the crime ever happened can be extraordinarily challenging after the fact. “DDoS can happen in a vacuum, unless a company captures logs in the right way,” Peterson says. Klein, a former UNIX administrator who grew up playing with Linux, spent weeks piecing together evidence and reassembling data to show how the DDoS attacks unfolded.

On the compromised devices, they had to carefully reconstruct the network traffic data, and study how the Mirai code launched so-called “packets” against its targets—a little-understood forensic process, known as analyzing PCAP (packet capture) data. Think of it as the digital equivalent of testing for fingerprints or gunshot residue. “It was the most complex DDoS software I’ve run across,” Klein says.

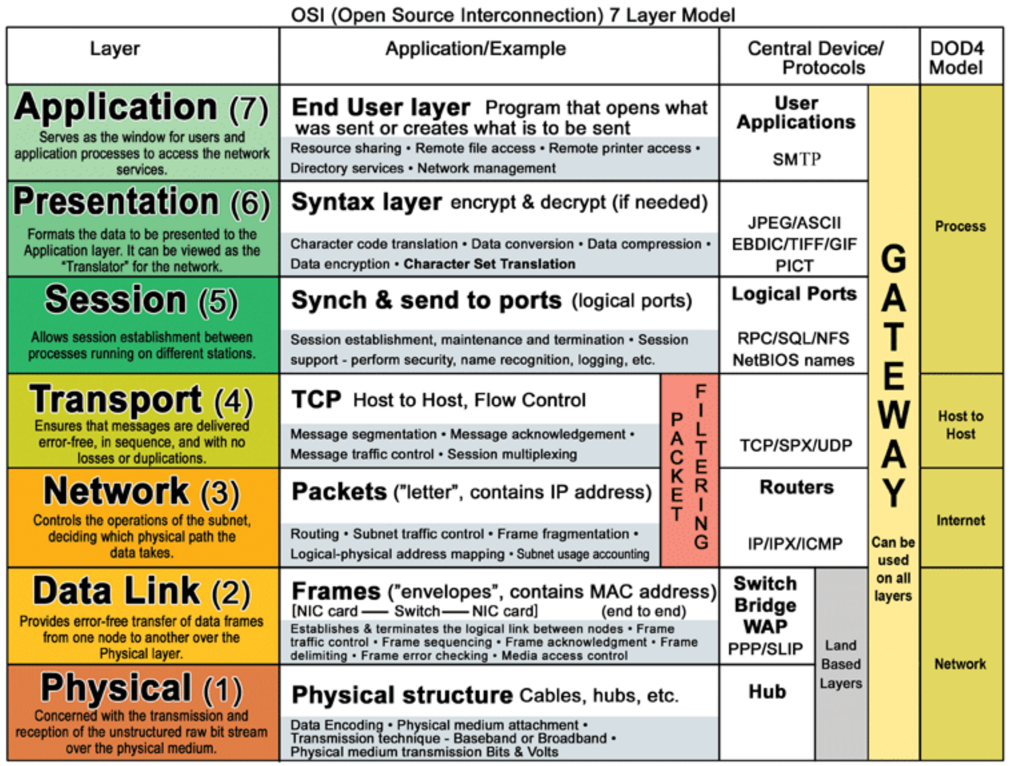
The FBI zeroed in on the suspects by the end of the year: Photos of the three hung for months on the wall in the Anchorage field office, where agents dubbed them the "Cub Scout Pack," a nod to their youthfulness. (Another older female suspect in an unrelated case, whose photo also hung on the board, was nicknamed the "Den Mother.")

What is a Packet?

- A unit of data
- A data stream (e.g., video, a web page) is comprised of many packets
- In general, a single packet contains the following information:
  - Source and destination IP addresses and ports
  - MAC address
  - Time To Live (TTL)
  - Protocol (e.g., TCP, UDP, IMCP)
  - Payload
- A packet encapsulates all layers of the *Open Systems Interconnection (OSI) model*

What is the OSI Model?

- “A conceptual framework that describes the functions of a networking or telecommunication system.”
- 7 layers
- Each layer is abstracted from the other
- Sources:
    - https://www.networkworld.com/article/3239677/lan-wan/the-osi-model-explained-how-to-understand-and-remember-the-7-layer-network-model.html
    - https://buildingautomationmonthly.com/what-is-the-osi-model/



OSI (Open Source Interconnection) 7 Layer Model

Layer	Application/Example		Central Device/ Protocols			DOD4 Model
Application (7) Serves as the window for users and application processes to access the network services.	**End User layer** Program that opens what was sent or creates what is to be sent Resource sharing • Remote file access • Remote printer access • Directory services • Network management		**User Applications** SMTP		**GATEWAY** **Can be used on all layers**	**Process**
Presentation (6) Formats the data to be presented to the Application layer. It can be viewed as the "Translator" for the network.	**Syntax layer** encrypt & decrypt (if needed) Character code translation • Data conversion • Data compression • Data encryption • **Character Set Translation**		JPEG/ASCII EBDIC/TIFF/GIF PICT			
Session (5) Allows session establishment between processes running on different stations.	**Synch & send to ports** (logical ports) Session establishment, maintenance and termination • Session support - perform security, name recognition, logging, etc.		**Logical Ports** RPC/SQL/NFS NetBIOS names			
Transport (4) Ensures that messages are delivered error-free, in sequence, and with no losses or duplications.	**TCP** Host to Host, Flow Control Message segmentation • Message acknowledgement • Message traffic control • Session multiplexing	PACKET FILTERING	TCP/SPX/UDP			**Host to Host**
Network (3) Controls the operations of the subnet, deciding which physical path the data takes.	**Packets** ("letter", contains IP address) Routing • Subnet traffic control • Frame fragmentation • Logical-physical address mapping • Subnet usage accounting		**Routers** IP/IPX/ICMP			**Internet**
Data Link (2) Provides error-free transfer of data frames from one node to another over the Physical layer.	**Frames** ("envelopes", contains MAC address) [NIC card —— Switch —— NIC card] (end to end) Establishes & terminates the logical link between nodes • Frame traffic control • Frame sequencing • Frame acknowledgment • Frame delimiting • Frame error checking • Media access control		**Switch Bridge WAP** PPP/SLIP	Land Based Layers		**Network**
Physical (1) Concerned with the transmission and reception of the unstructured raw bit stream over the physical medium.	**Physical structure** Cables, hubs, etc. Data Encoding • Physical medium attachment • Transmission technique - Baseband or Broadband • Physical medium transmission Bits & Volts		**Hub**			

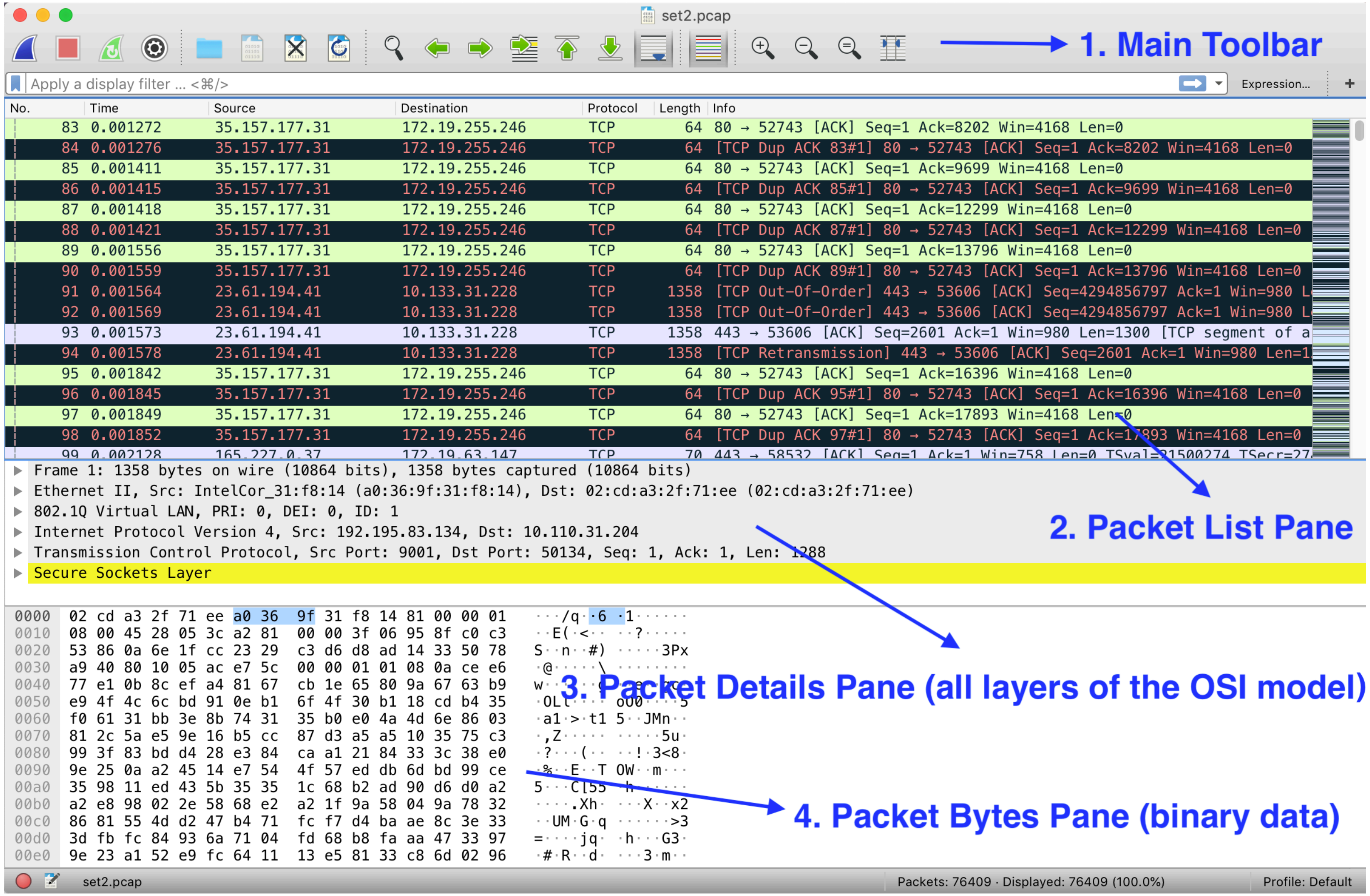
What is a PCAP File?

- PCAP stands for “packet capture”
- `.pcap`: The common file extension for packet captures and is commonly used in many applications such as *Wireshark*
- A 100 MB PCAP file contains tens of thousands of packets

What is Wireshark?

- Graphical and extensive packet analyzer
- Open source and free
- Platform independent (Windows, macOS, and Linux versions available)
- Features include filtering, reconstructing conversations, reconstructing files based on packets
- Website: https://www.wireshark.org/

The Wireshark User Interface



set2.pcap

Apply a display filter ... <⌘/>

Expression...

```
No.   Time      Source           Destination      Protocol Length Info
   83 0.001272  35.157.177.31    172.19.255.246   TCP        64  80 → 52743 [ACK] Seq=1 Ack=8202 Win=4168 Len=0
   84 0.001276  35.157.177.31    172.19.255.246   TCP        64  [TCP Dup ACK 83#1] 80 → 52743 [ACK] Seq=1 Ack=8202 Win=4168 Len=0
   85 0.001411  35.157.177.31    172.19.255.246   TCP        64  80 → 52743 [ACK] Seq=1 Ack=9699 Win=4168 Len=0
   86 0.001415  35.157.177.31    172.19.255.246   TCP        64  [TCP Dup ACK 85#1] 80 → 52743 [ACK] Seq=1 Ack=9699 Win=4168 Len=0
   87 0.001418  35.157.177.31    172.19.255.246   TCP        64  80 → 52743 [ACK] Seq=1 Ack=12299 Win=4168 Len=0
   88 0.001421  35.157.177.31    172.19.255.246   TCP        64  [TCP Dup ACK 87#1] 80 → 52743 [ACK] Seq=1 Ack=12299 Win=4168 Len=0
   89 0.001556  35.157.177.31    172.19.255.246   TCP        64  80 → 52743 [ACK] Seq=1 Ack=13796 Win=4168 Len=0
   90 0.001559  35.157.177.31    172.19.255.246   TCP        64  [TCP Dup ACK 89#1] 80 → 52743 [ACK] Seq=1 Ack=13796 Win=4168 Len=0
   91 0.001564  23.61.194.41     10.133.31.228    TCP      1358  [TCP Out-Of-Order] 443 → 53606 [ACK] Seq=4294856797 Ack=1 Win=980 L
   92 0.001569  23.61.194.41     10.133.31.228    TCP      1358  [TCP Out-Of-Order] 443 → 53606 [ACK] Seq=4294856797 Ack=1 Win=980 L
   93 0.001573  23.61.194.41     10.133.31.228    TCP      1358  443 → 53606 [ACK] Seq=2601 Ack=1 Win=980 Len=1300 [TCP segment of a
   94 0.001578  23.61.194.41     10.133.31.228    TCP      1358  [TCP Retransmission] 443 → 53606 [ACK] Seq=2601 Ack=1 Win=980 Len=1
   95 0.001842  35.157.177.31    172.19.255.246   TCP        64  80 → 52743 [ACK] Seq=1 Ack=16396 Win=4168 Len=0
   96 0.001845  35.157.177.31    172.19.255.246   TCP        64  [TCP Dup ACK 95#1] 80 → 52743 [ACK] Seq=1 Ack=16396 Win=4168 Len=0
   97 0.001849  35.157.177.31    172.19.255.246   TCP        64  80 → 52743 [ACK] Seq=1 Ack=17893 Win=4168 Len=0
   98 0.001852  35.157.177.31    172.19.255.246   TCP        64  [TCP Dup ACK 97#1] 80 → 52743 [ACK] Seq=1 Ack=17893 Win=4168 Len=0
   99 0.002128  165.227.0.37     172.19.63.147    TCP        70  443 → 58532 [ACK] Seq=1 Ack=1 Win=758 Len=0 TSval=21500274 TSecr=27
```

```
▶ Frame 1: 1358 bytes on wire (10864 bits), 1358 bytes captured (10864 bits)
▶ Ethernet II, Src: IntelCor_31:f8:14 (a0:36:9f:31:f8:14), Dst: 02:cd:a3:2f:71:ee (02:cd:a3:2f:71:ee)
▶ 802.1Q Virtual LAN, PRI: 0, DEI: 0, ID: 1
▶ Internet Protocol Version 4, Src: 192.195.83.134, Dst: 10.110.31.204
▶ Transmission Control Protocol, Src Port: 9001, Dst Port: 50134, Seq: 1, Ack: 1, Len: 1288
▶ Secure Sockets Layer
```

```
0000  02 cd a3 2f 71 ee a0 36  9f 31 f8 14 81 00 00 01   ···/q·6 ·1······
0010  08 00 45 28 05 3c a2 81  00 00 3f 06 95 8f c0 c3   ··E(·<·· ··?·····
0020  53 86 0a 6e 1f cc 23 29  c3 d6 d8 ad 14 33 50 78   S··n··#) ·····3Px
0030  a9 40 80 10 05 ac e7 5c  00 00 01 01 08 0a ce e6   ·@·····\ ········
0040  77 e1 0b 8c ef a4 81 67  cb 1e 65 80 9a 67 63 b9   w······g ··e··gc·
0050  e9 4f 4c 6c bd 91 0e b1  6f 4f 30 b1 18 cd b4 35   ·OLl···· oO0····5
0060  f0 61 31 bb 3e 8b 74 31  35 b0 e0 4a 4d 6e 86 03   ·a1·>·t1 5··JMn··
0070  81 2c 5a e5 9e 16 b5 cc  87 d3 a5 a5 10 35 75 c3   ·,Z····· ·····5u·
0080  99 3f 83 bd d4 28 e3 84  ca a1 21 84 33 3c 38 e0   ·?···(·· ··!·3<8·
0090  9e 25 0a a2 45 14 e7 54  4f 57 ed db 6d bd 99 ce   ·%··E··T OW··m···
00a0  35 98 11 ed 43 5b 35 35  1c 68 b2 ad 90 d6 d0 a2   5···C[55 ·h······
00b0  a2 e8 98 02 2e 58 68 e2  a2 1f 9a 58 04 9a 78 32   ····.Xh· ···X··x2
00c0  86 81 55 4d d2 47 b4 71  fc f7 d4 ba ae 8c 3e 33   ··UM·G·q ······>3
00d0  3d fb fc 84 93 6a 71 04  fd 68 b8 fa aa 47 33 97   =····jq· ·h···G3·
00e0  9e 23 a1 52 e9 fc 64 11  13 e5 81 33 c8 6d 02 96   ·#·R··d· ···3·m··
```

set2.pcap

Packets: 76409 · Displayed: 76409 (100.0%)

Profile: Default

1. Main Toolbar

2. Packet List Pane

3. Packet Details Pane (all layers of the OSI model)

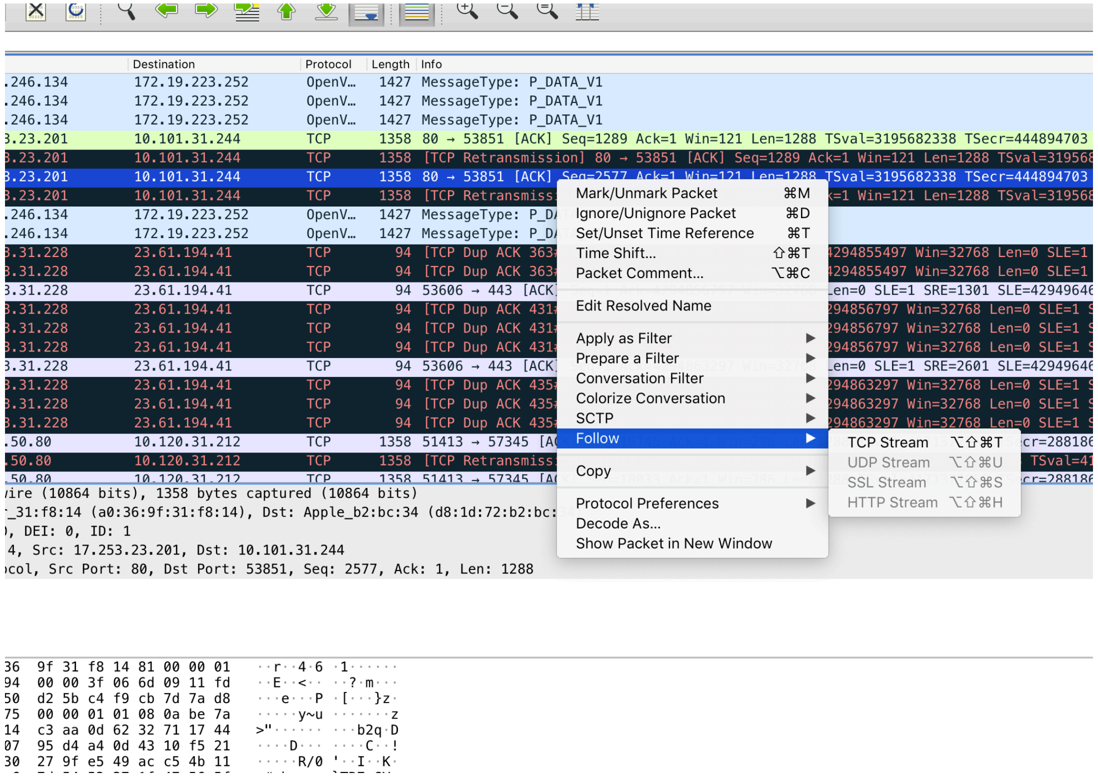
4. Packet Bytes Pane (binary data)

Exercise 1: Opening a Simple PCAP File in Wireshark

- Download: https://www.cs.tufts.edu/comp/116/simple.pcap
- Question 1: How many packets are there?
- Question 2: What networking protocol is used?
- Question 3: What is the source IP address?
- Question 4: What is the destination IP address?
- Question 5: What port number is the source using to communicate with the destination (or what port number is the destination listening on)?
- BONUS: Do you notice the “three-way handshake”?

Reconstructing a Conversation in Wireshark

1. Click on a packet (it will be highlighted in blue)
2. Right-click on packet
3. Go to “Follow”
4. Follow one of the following streams depending on protocol (TCP Stream is most common)



```
            Destination      Protocol Length Info
.246.134    172.19.223.252   OpenV…   1427 MessageType: P_DATA_V1
.246.134    172.19.223.252   OpenV…   1427 MessageType: P_DATA_V1
.246.134    172.19.223.252   OpenV…   1427 MessageType: P_DATA_V1
3.23.201    10.101.31.244    TCP      1358 80 → 53851 [ACK] Seq=1289 Ack=1 Win=121 Len=1288 TSval=3195682338 TSecr=444894703
3.23.201    10.101.31.244    TCP      1358 [TCP Retransmission] 80 → 53851 [ACK] Seq=1289 Ack=1 Win=121 Len=1288 TSval=319568
3.23.201    10.101.31.244    TCP      1358 80 → 53851 [ACK] Seq=2577 Ack=1 Win=121 Len=1288 TSval=3195682338 TSecr=444894703
3.23.201    10.101.31.244    TCP      1358 [TCP Retransmiss                        k=1 Win=121 Len=1288 TSval=319568
.246.134    172.19.223.252   OpenV…   1427 MessageType: P_D
.246.134    172.19.223.252   OpenV…   1427 MessageType: P_D
3.31.228    23.61.194.41     TCP        94 [TCP Dup ACK 363                        4294855497 Win=32768 Len=0 SLE=1
3.31.228    23.61.194.41     TCP        94 [TCP Dup ACK 363                        4294855497 Win=32768 Len=0 SLE=1
3.31.228    23.61.194.41     TCP        94 53606 → 443 [ACK                        Len=0 SLE=1 SRE=1301 SLE=4294964
3.31.228    23.61.194.41     TCP        94 [TCP Dup ACK 431                        294856797 Win=32768 Len=0 SLE=1 S
3.31.228    23.61.194.41     TCP        94 [TCP Dup ACK 431                        294856797 Win=32768 Len=0 SLE=1 S
3.31.228    23.61.194.41     TCP        94 [TCP Dup ACK 431                        294856797 Win=32768 Len=0 SLE=1 S
3.31.228    23.61.194.41     TCP        94 53606 → 443 [ACK                        Len=0 SLE=1 SRE=2601 SLE=4294964
3.31.228    23.61.194.41     TCP        94 [TCP Dup ACK 435                        294863297 Win=32768 Len=0 SLE=1 S
3.31.228    23.61.194.41     TCP        94 [TCP Dup ACK 435                        294863297 Win=32768 Len=0 SLE=1 S
3.31.228    23.61.194.41     TCP        94 [TCP Dup ACK 435                        294863297 Win=32768 Len=0 SLE=1 S
.50.80      10.120.31.212    TCP      1358 51413 → 57345 [A                                                 cr=288186
.50.80      10.120.31.212    TCP      1358 [TCP Retransmiss                                                 TSval=41
.50.80      10.120.31.212    TCP      1358 51413 → 57345 [A                                                 cr=288186
```

```
vire (10864 bits), 1358 bytes captured (10864 bits)
_31:f8:14 (a0:36:9f:31:f8:14), Dst: Apple_b2:bc:34 (d8:1d:72:b2:bc:
), DEI: 0, ID: 1
4, Src: 17.253.23.201, Dst: 10.101.31.244
ocol, Src Port: 80, Dst Port: 53851, Seq: 2577, Ack: 1, Len: 1288
```

```
36  9f 31 f8 14 81 00 00 01   ··r··4·6 ·1······
94  00 00 3f 06 6d 09 11 fd   ··E··<·· ··?·m···
50  d2 5b c4 f9 cb 7d 7a d8   ···e···P ·[···}z·
75  00 00 01 01 08 0a be 7a   ·····y~u ·······z
14  c3 aa 0d 62 32 71 17 44   >"······ ···b2q·D
07  95 d4 a4 0d 43 10 f5 21   ····D··· ····C··!
30  27 9f e5 49 ac c5 4b 11   ·····R/0 '··I··K·
```

Exercise 2: Extracting Pictures

- Download: https://www.cs.tufts.edu/comp/116/set1.pcap
- Question 1: What insecure protocol was used to transmit pictures on network?
- Question 2: How many pictures were transmitted?
- Question 3: Extract one of the pictures that was transmitted. HINT: show and save the picture as “Raw” format.

Base64

- Base64 is an *encoding* scheme
- Used to represent binary data in ASCII text format
- Base64 is not encryption. Base64 is not encryption. Base64 is not encryption. Base64 is not encryption. Base64 is not encryption. Base64 is not encryption. Base64 is not encryption. Base64 is not encryption. Base64 is not encryption. Base64 is not encryption. https://sempf.net/post/base64-is-not-encryption
- Why is this important? "In basic HTTP authentication, a request contains a header field of the form `Authorization: Basic <credentials>`, where credentials is the Base64 encoding of id and password joined by a colon." (source: https://en.wikipedia.org/wiki/Basic_access_authentication)

Exercise 3: Extracting Username:Password Pairs

- Download: https://www.cs.tufts.edu/comp/116/set3.pcap
- Question 1: What protocol was used to transmit the username:password pair (credentials)?
- Question 2: What is one username:password pair in this PCAP set? HINT: use Edit > Find Packet
- Question 3: Is the username:password pair valid? Why / why not?

Where Do You Go From Here?

- Sniff and validate passwords
- Reconstruct files (e.g., images, MP3s)
- Volunteer at the Wall of Sheep and the Packet Hacking Village
- Learn more at our Packet Inspector event
- Further develop your skills at our Packet Detective event https://www.wallofsheep.com/pages/packet-detective
- Enter Capture The Packet, a DEF CON Black Badge contest

(If time allows) Exercise 4: Extracting Username:Password Pairs

- Download: https://www.cs.tufts.edu/comp/116/set2.pcap
- This PCAP set is from the DEF CON conference. I am not responsible for the contents in this PCAP set.
- Question 1: How many packets are there in this PCAP set?
- Question 2: Find all the credentials in this PCAP set
- Question 3: Are the credentials valid?
- BONUS: Provide a list of all the domains and IP addresses in this PCAP set

Appendix: What is tshark?

- Command-line-based Wireshark
- Installed with Wireshark
- Dumps and analyzes network traffic
- Example, list the hosts (IP addresses and domains) in the PCAP file
  - `tshark -r file.pcap -q -z hosts,ipv4`