By: Kamil Chaudhry, Asad Khan, Hamza Mehmood

The Agenda

- CSC347 Review
- Tools & Filters
- Advanced Use Cases
- Practical Demonstrations
- Real-Life Examples

First, A Quick Story…

Content Warning (Sensitive Topics)

Source: "The School Shootings Were Fake. The Terror was Real" (very interesting stuff, read the whole thing if you have time)


POLICE
POLICE STATION
POLICE

Brad Dennis

- Private investigator hired by twitch streamers
- Uses detective skills to join a private telegram channel
- Joins a WoW guild with member named Nazgul (AKA Torswats)
- Notices Nazgul’s telegram switched from swatting twitch streamers to schools

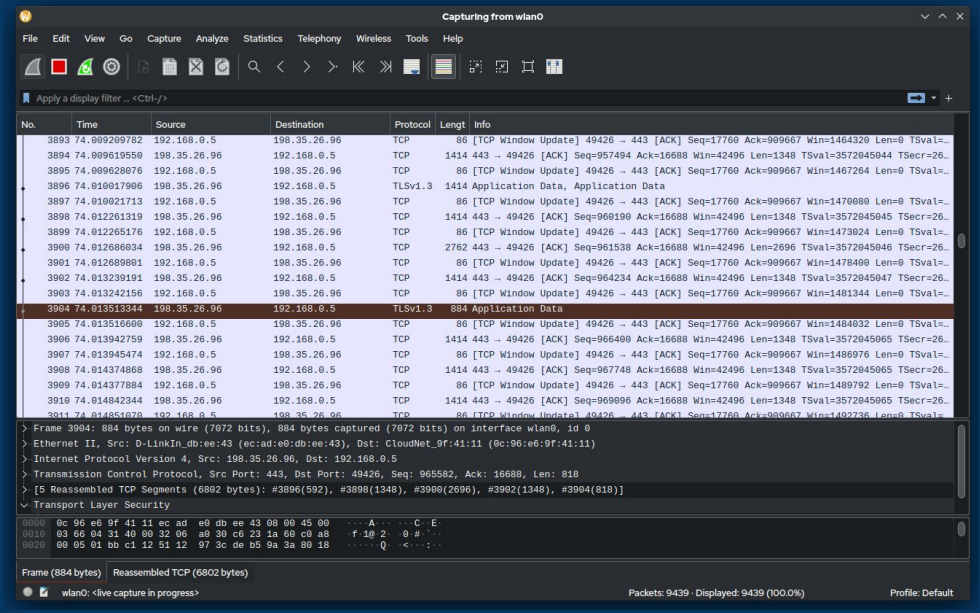
Nazgul's Arrest

- Brad convinced Nazgul to add him as a friend on Tox, tempting him with a business opportunity
- Tox uses a two-way IP to IP tunnel
- Brad starts a wireshark capture as he communicates with Nazgul
- Successfully creates a packet capture file containing Nazgul's IP address
- Provided the FBI with the .pcap file containing the IP in a USB drive
- Nazgul eventually arrested and pleaded guilty

Introduction to Wireshark

Introduction to Wireshark

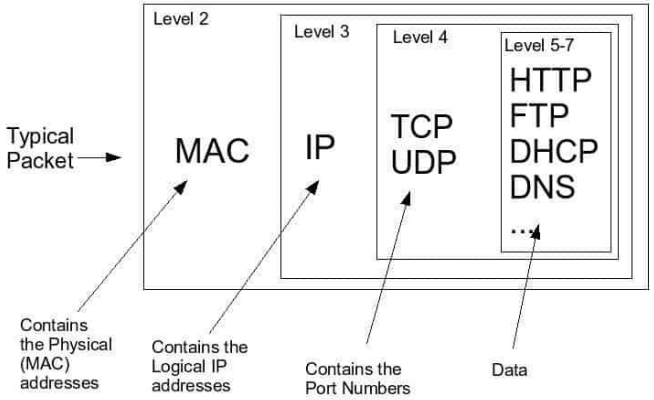
- Open-Source Network Packet/Protocol Analyzer
  - Captures packets and displays data in detail
  - Written in C, C++ and Lua
- Invented by Gerald Combs in 1998
  - Originally named “Ethereal”
  - Later renamed “Wireshark” in 2006
- Used for various purposes
  - Troubleshoot Network Issues
  - Examine Security Issues/Vulnerabilities
  - Debug Protocol Implementations

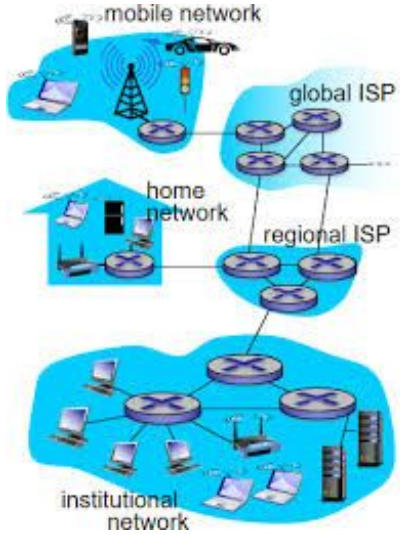


Example of a Wireshark GUI

Introduction to Networks

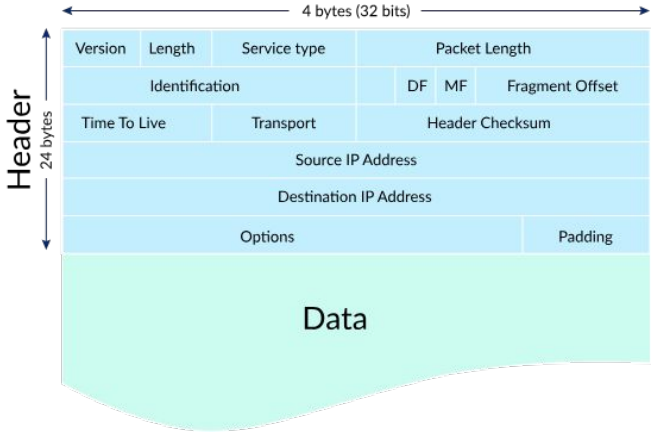
- A system of interconnected devices for communication and data sharing.
- Each device on a network is identified by it IP address
- DNS servers map domain names to IP addresses
- Devices talk to each other by sending packets over the network

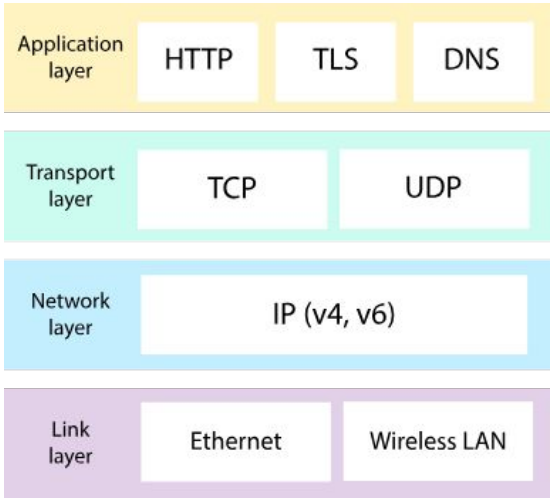




Protocols

- Packets contain routing information along with the data
- Packets always follow protocols
- Routing information encapsulating the data (payload)
- Each layer has their own protocols
    - Used to forward or interpret the packet at the receiving end

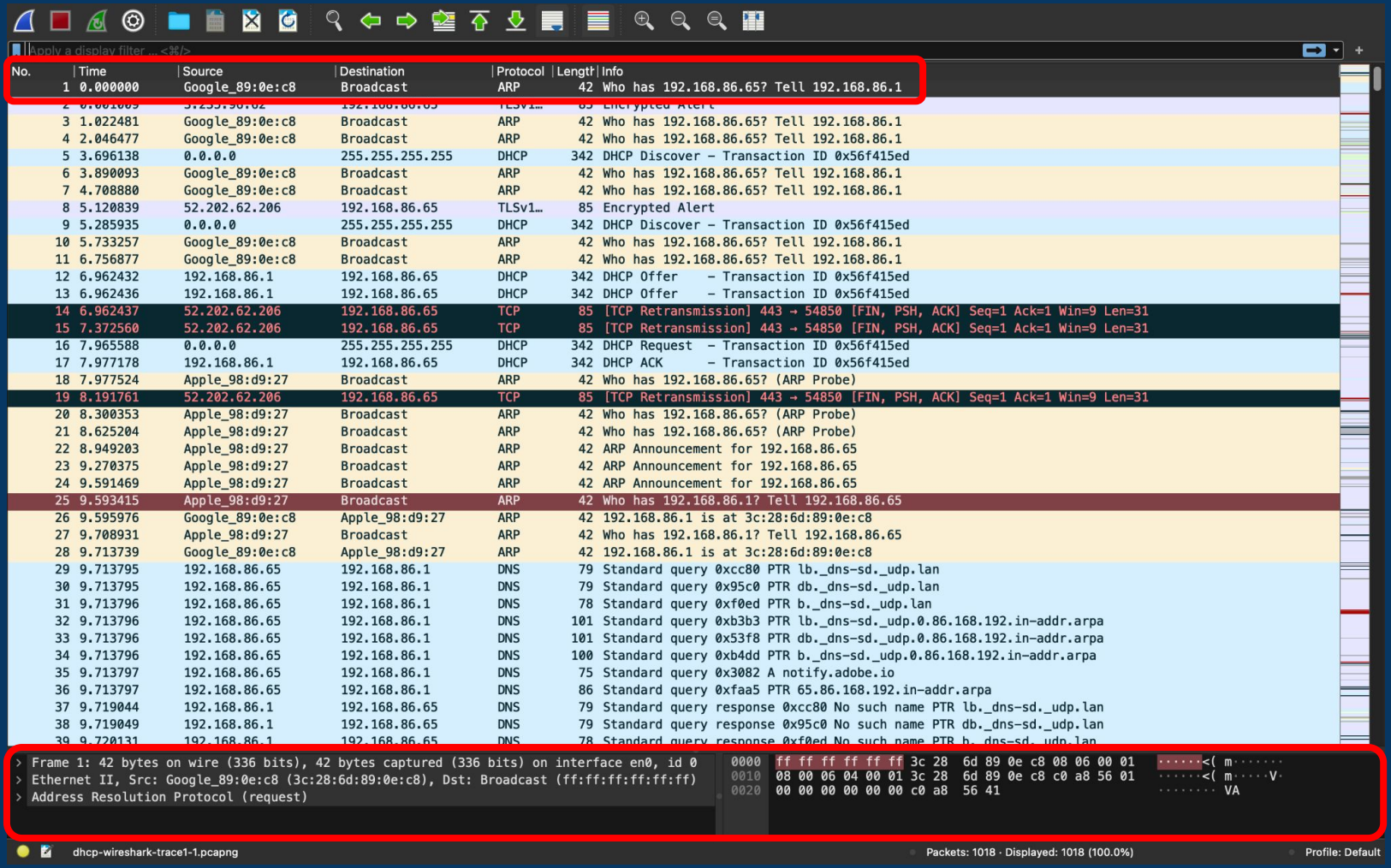




Wireshark Usage

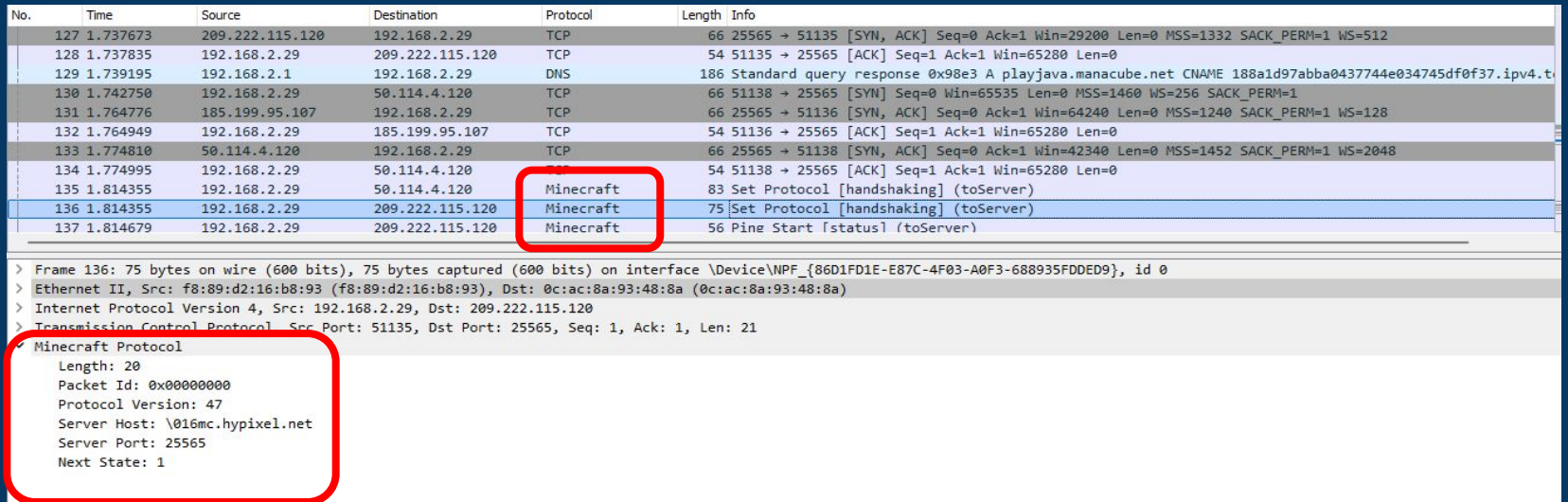
Installation

- Available on: https://wireshark.com/download.html
- Managed by Wireshark Foundation
- Open-Source
- Cross-Platform:
  - Linux
  - MacOS
  - Windows
- Can be built directly from source as well


Apply a display filter ... <⌘/>
No. Time Source Destination Protocol Length Info
1 0.000000 Google_89:0e:c8 Broadcast ARP 42 Who has 192.168.86.65? Tell 192.168.86.1
3 1.022481 Google_89:0e:c8 Broadcast ARP 42 Who has 192.168.86.65? Tell 192.168.86.1
4 2.046477 Google_89:0e:c8 Broadcast ARP 42 Who has 192.168.86.65? Tell 192.168.86.1
5 3.696138 0.0.0.0 255.255.255.255 DHCP 342 DHCP Discover - Transaction ID 0x56f415ed
6 3.890093 Google_89:0e:c8 Broadcast ARP 42 Who has 192.168.86.65? Tell 192.168.86.1
7 4.708880 Google_89:0e:c8 Broadcast ARP 42 Who has 192.168.86.65? Tell 192.168.86.1
8 5.120839 52.202.62.206 192.168.86.65 TLSv1… 85 Encrypted Alert
9 5.285935 0.0.0.0 255.255.255.255 DHCP 342 DHCP Discover - Transaction ID 0x56f415ed
10 5.733257 Google_89:0e:c8 Broadcast ARP 42 Who has 192.168.86.65? Tell 192.168.86.1
11 6.756877 Google_89:0e:c8 Broadcast ARP 42 Who has 192.168.86.65? Tell 192.168.86.1
12 6.962432 192.168.86.1 192.168.86.65 DHCP 342 DHCP Offer - Transaction ID 0x56f415ed
13 6.962436 192.168.86.1 192.168.86.65 DHCP 342 DHCP Offer - Transaction ID 0x56f415ed
14 6.962437 52.202.62.206 192.168.86.65 TCP 85 [TCP Retransmission] 443 → 54850 [FIN, PSH, ACK] Seq=1 Ack=1 Win=9 Len=31
15 7.372560 52.202.62.206 192.168.86.65 TCP 85 [TCP Retransmission] 443 → 54850 [FIN, PSH, ACK] Seq=1 Ack=1 Win=9 Len=31
16 7.965588 0.0.0.0 255.255.255.255 DHCP 342 DHCP Request - Transaction ID 0x56f415ed
17 7.977178 192.168.86.1 192.168.86.65 DHCP 342 DHCP ACK - Transaction ID 0x56f415ed
18 7.977524 Apple_98:d9:27 Broadcast ARP 42 Who has 192.168.86.65? (ARP Probe)
19 8.191761 52.202.62.206 192.168.86.65 TCP 85 [TCP Retransmission] 443 → 54850 [FIN, PSH, ACK] Seq=1 Ack=1 Win=9 Len=31
20 8.300353 Apple_98:d9:27 Broadcast ARP 42 Who has 192.168.86.65? (ARP Probe)
21 8.625204 Apple_98:d9:27 Broadcast ARP 42 Who has 192.168.86.65? (ARP Probe)
22 8.949203 Apple_98:d9:27 Broadcast ARP 42 ARP Announcement for 192.168.86.65
23 9.270375 Apple_98:d9:27 Broadcast ARP 42 ARP Announcement for 192.168.86.65
24 9.591469 Apple_98:d9:27 Broadcast ARP 42 ARP Announcement for 192.168.86.65
25 9.593415 Apple_98:d9:27 Broadcast ARP 42 Who has 192.168.86.1? Tell 192.168.86.65
26 9.595976 Google_89:0e:c8 Apple_98:d9:27 ARP 42 192.168.86.1 is at 3c:28:6d:89:0e:c8
27 9.708931 Apple_98:d9:27 Broadcast ARP 42 Who has 192.168.86.1? Tell 192.168.86.65
28 9.713739 Google_89:0e:c8 Apple_98:d9:27 ARP 42 192.168.86.1 is at 3c:28:6d:89:0e:c8
29 9.713795 192.168.86.65 192.168.86.1 DNS 79 Standard query 0xcc80 PTR lb._dns-sd._udp.lan
30 9.713795 192.168.86.65 192.168.86.1 DNS 79 Standard query 0x95c0 PTR db._dns-sd._udp.lan
31 9.713796 192.168.86.65 192.168.86.1 DNS 78 Standard query 0xf0ed PTR b._dns-sd._udp.lan
32 9.713796 192.168.86.65 192.168.86.1 DNS 101 Standard query 0xb3b3 PTR lb._dns-sd._udp.0.86.168.192.in-addr.arpa
33 9.713796 192.168.86.65 192.168.86.1 DNS 101 Standard query 0x53f8 PTR db._dns-sd._udp.0.86.168.192.in-addr.arpa
34 9.713796 192.168.86.65 192.168.86.1 DNS 100 Standard query 0xb4dd PTR b._dns-sd._udp.0.86.168.192.in-addr.arpa
35 9.713797 192.168.86.65 192.168.86.1 DNS 75 Standard query 0x3082 A notify.adobe.io
36 9.713797 192.168.86.65 192.168.86.1 DNS 86 Standard query 0xfaa5 PTR 65.86.168.192.in-addr.arpa
37 9.719044 192.168.86.1 192.168.86.65 DNS 79 Standard query response 0xcc80 No such name PTR lb._dns-sd._udp.lan
38 9.719049 192.168.86.1 192.168.86.65 DNS 79 Standard query response 0x95c0 No such name PTR db._dns-sd._udp.lan
Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface en0, id 0
Ethernet II, Src: Google_89:0e:c8 (3c:28:6d:89:0e:c8), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Address Resolution Protocol (request)
0000 ff ff ff ff ff ff 3c 28 6d 89 0e c8 08 06 00 01
0010 08 00 06 04 00 01 3c 28 6d 89 0e c8 c0 a8 56 01
0020 00 00 00 00 00 00 c0 a8 56 41
dhcp-wireshark-trace1-1.pcapng
Packets: 1018 · Displayed: 1018 (100.0%)
Profile: Default

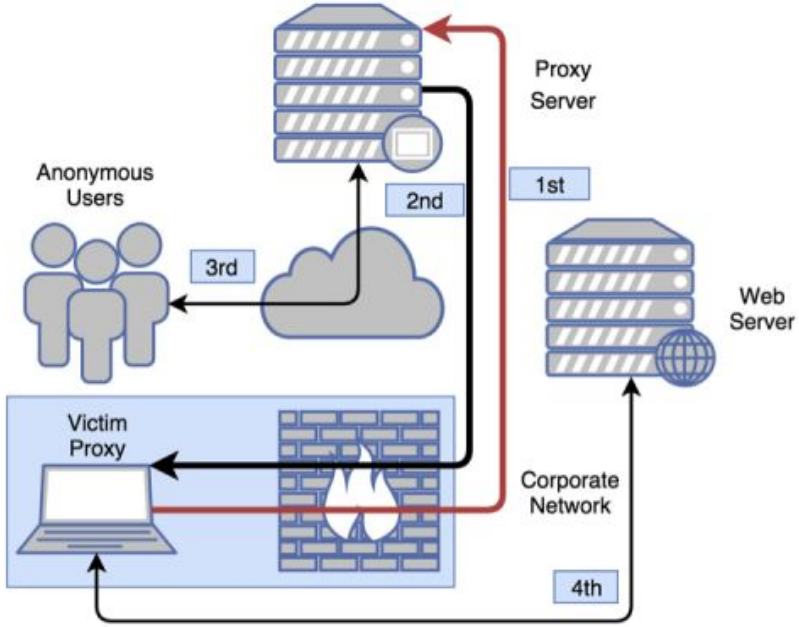
Plugins

- Wireshark supports custom open-source plugins
- Plugins add extra features for dissecting protocols and interfacing with third-party tools
- Wireshark has a vast community of developers
- Almost always a solution for what you need



Wireshark Usage

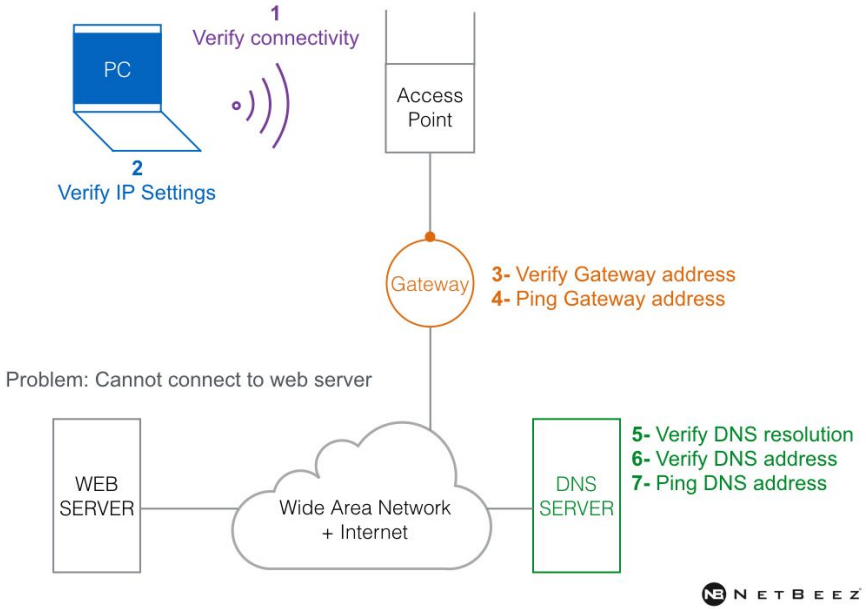
- Network Troubleshooting
  - Identifying slow responses, dropped packets, or bandwidth hogs
  - Analyzing DNS responses to see if requests are timing out
- Incident Response
  - Identifying infected hosts and suspicious traffic patterns
- Malware Analysis
  - Observing communication patterns
    - C2 traffic
    - Suspicious domains



Example Case

Identifying Users on a Network

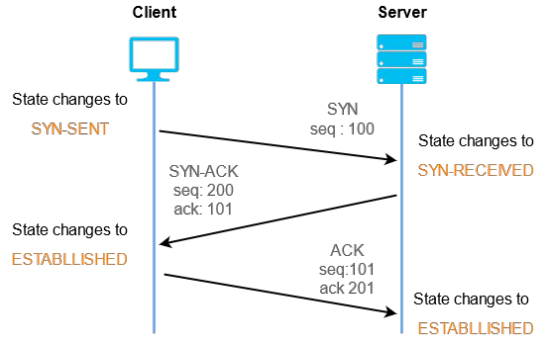
- Users within an organization's network can become compromised from a malicious attack
- Requires a review of packet capture (.pcap) data to analyze network traffic generated by the affected host
- Wireshark can be used to identify host and user data in .pcap files
- Packet capture data sourced from Palo Alto Networks (Unit 42)



Malicious Uses

Man in the Middle Attacks

- 3-way handshake an ISN is established
- Wireshark to detect connection
- Collect ISN then authentication
- Forge request on user or server behalf
- Works in cases where servers are hosting unencrypted traffic
    - i.e HTTP

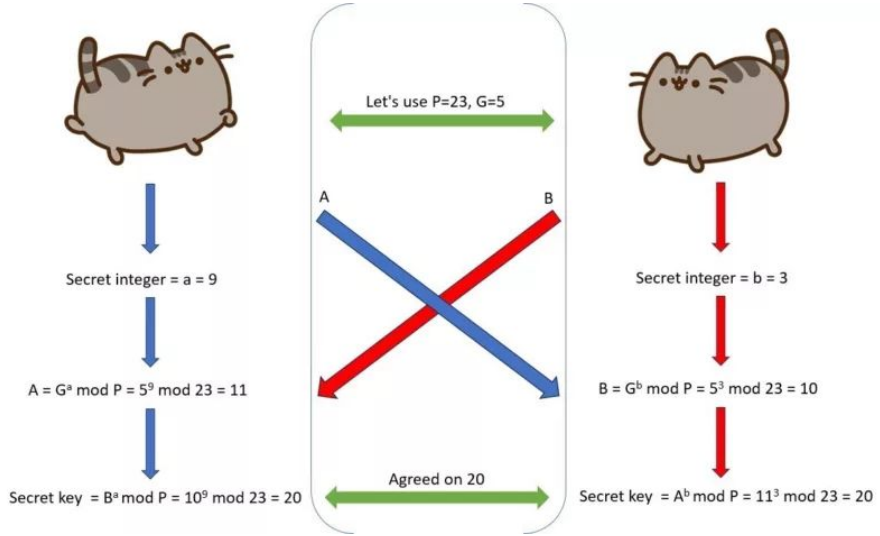


MITM Reassembling Files

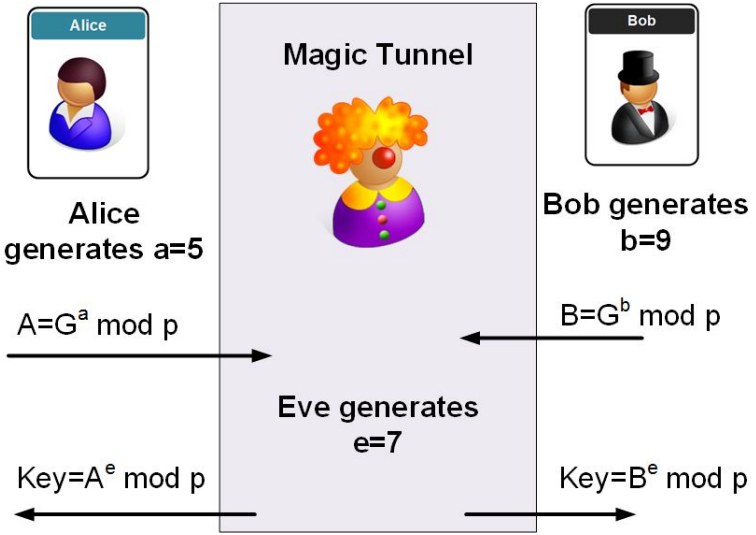
- Files being uploaded over insecure protocol
- Can be reassembled after being intercepted
- Sensitive files can potentially be used as blackmail

MITM with DHKE

Normal Exchange

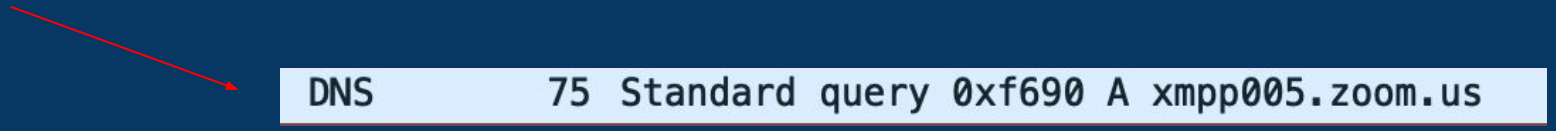


Man in the Middle



DNS Spoofing

- DNS requests plain text



- Victim sends DNS request for some address www.example.com
- Attacker is using wireshark and filtering for dns requests and finds one for the website whose replica they have created
- Attacker forges a response for a site replica that they are hosting on another ip address
- Victim puts sensitive data into the replica – their data is collected by attacker

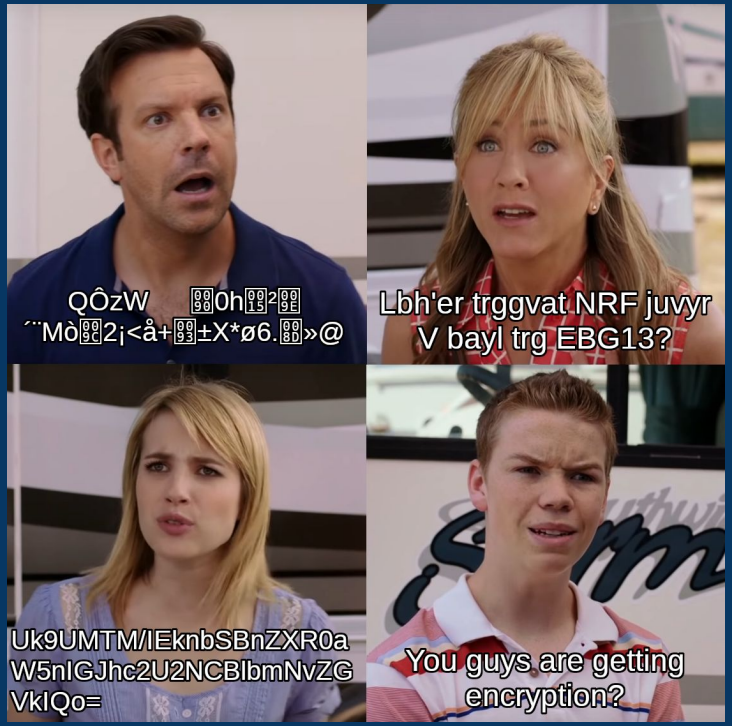
Scapy

- An interactive packet manipulation tool written in Python
- Similar to Wireshark, can be used to sniff and decode packets
- It can also be used to send packets
- Doesn’t have a nice GUI like Wireshark
- In the lab you will be using the scapy python library to forge a DNS response

Defending Against Snooping & Attacks

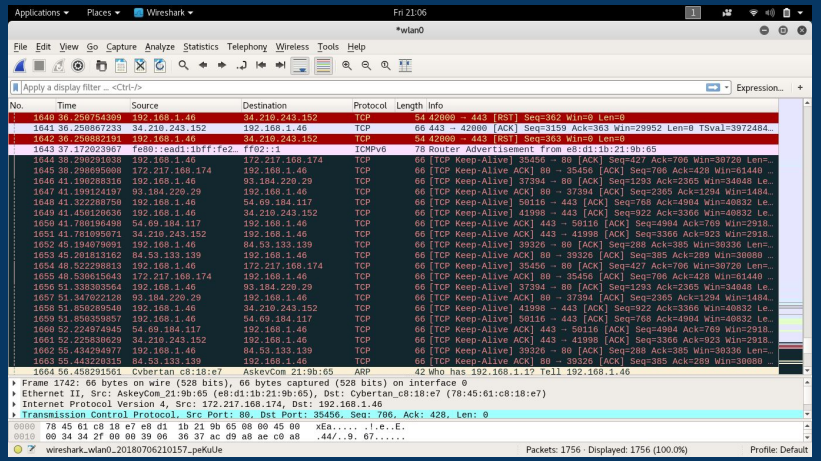
Encryption

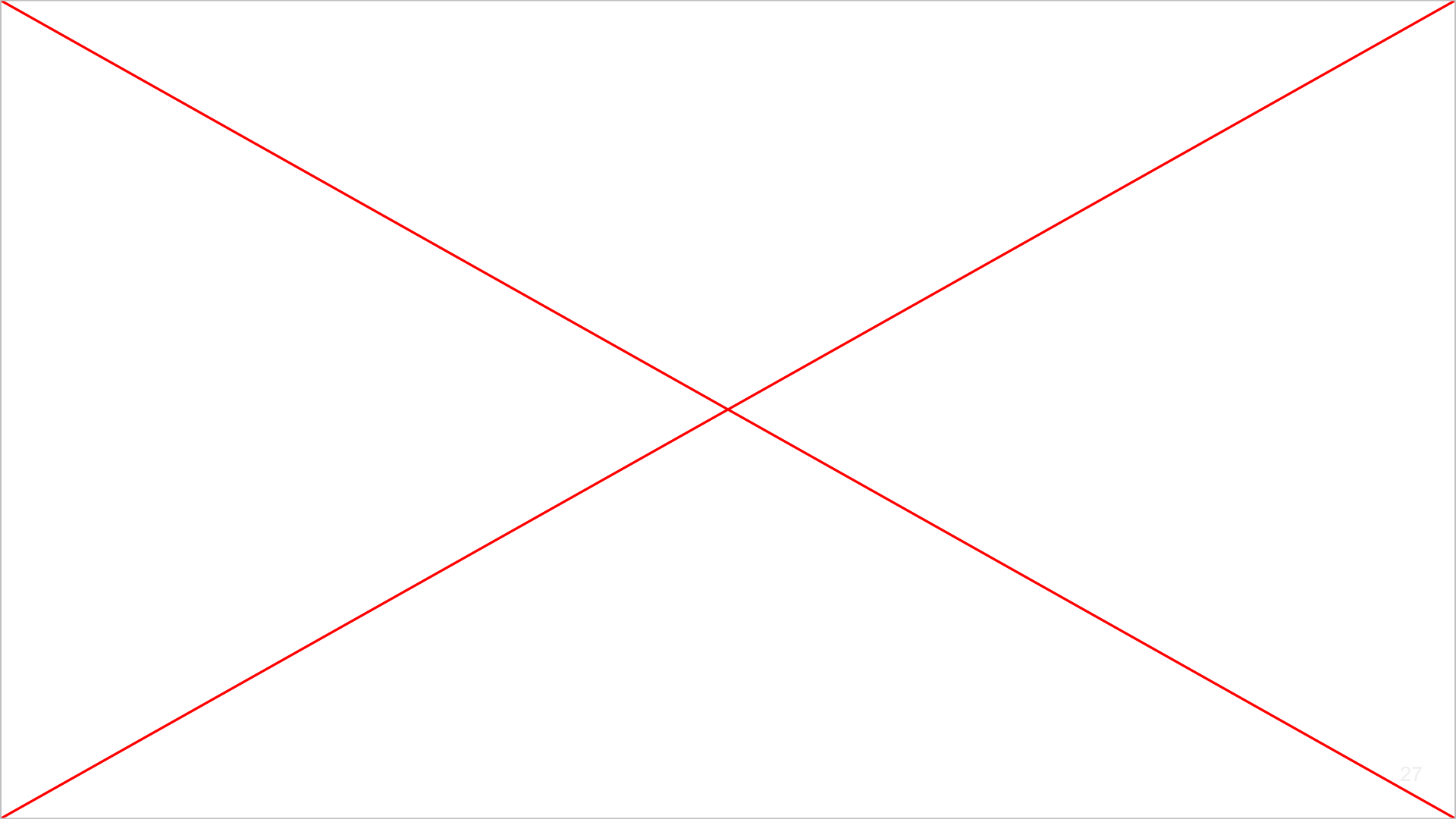
- Use encrypted or safe versions of services
  - HTTPS over HTTP
  - SSH over TELNET
  - SFTP over FTP
  - SMTPS over SMTP
- Prevents data from being exposed to snoopers



Vigilance

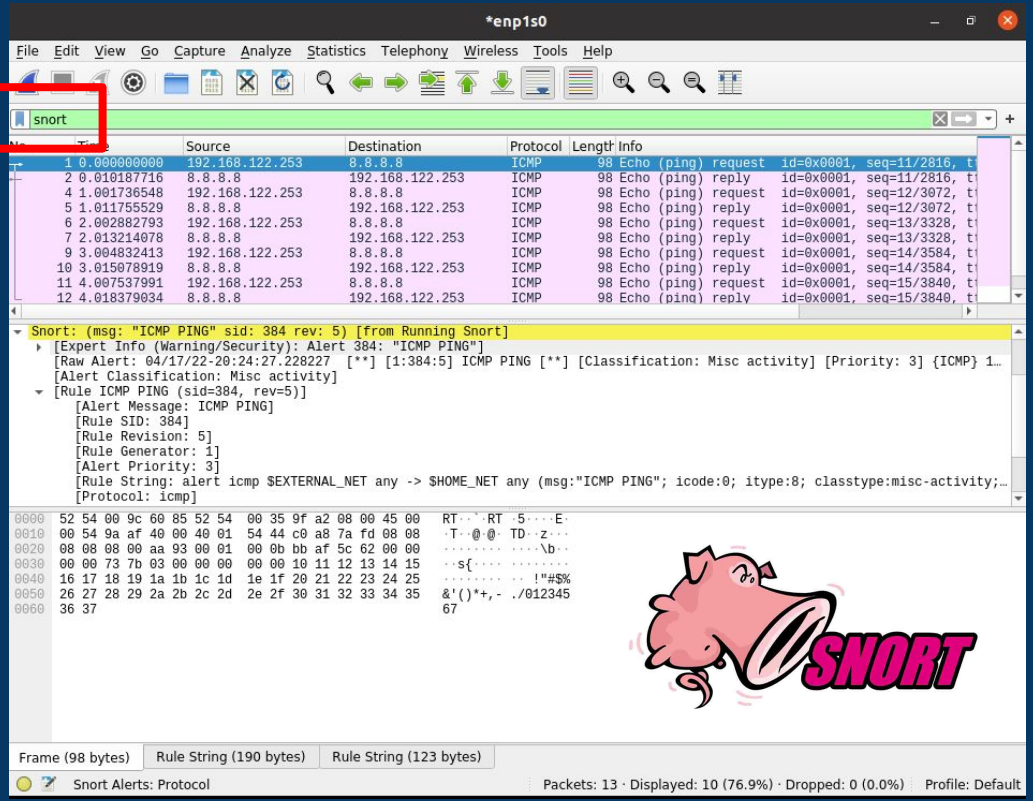
- Avoid public WiFi
- Use a trusted VPN
- Ensure URL correctness & certificates
- Ensure network speeds are within expected range
- Inspect your own network traffic to identify suspicious requests





Snort

- Quite difficult to always keep track of network activity
- Snort monitors network activity and uses a series of rules that help define malicious network activity
- Generates alerts in case of malicious activity
- Packets capture on snort can be forwarded to Wireshark for inspection



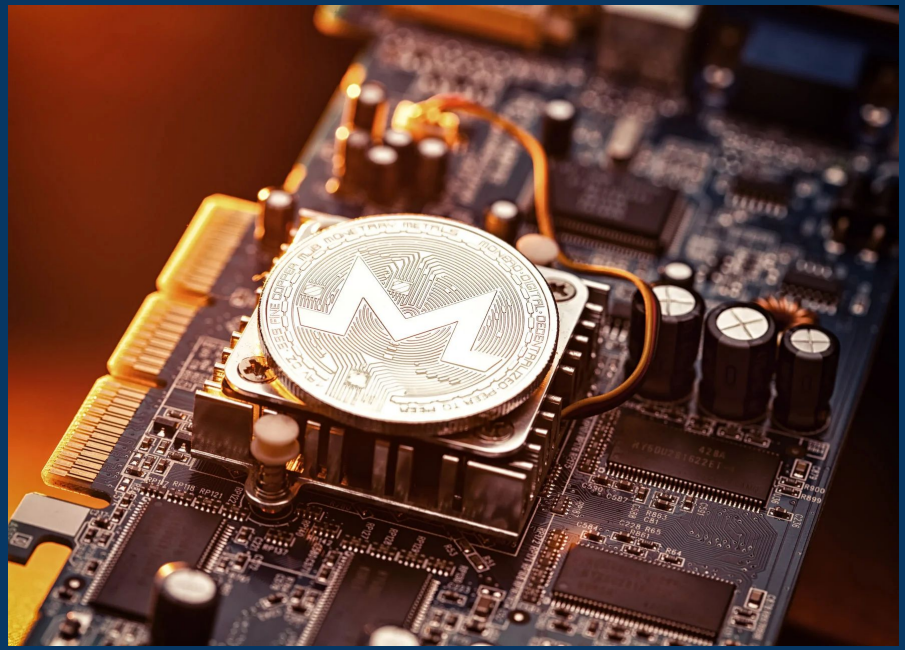
The Norman Cryptominer

The Case

- Varonis Security Research Team
- Monero crypto mining infection at a mid-sized company
- Discovery of a new cryptocurrency mining variant named “Norman”
- Infection spread to nearly all servers and workstations
- Infection caused major slowdowns and application instability

Norman

- XMRIG-based cryptocurrency mining software
  - Open-Source tool
  - Used for mining Monero
- Monero
  - Cryptocurrency w/ privacy focus
- Payloads injected into legitimate process for stealth
- Terminates the mining process upon Task Manager detection
- Uses legitimate looking processes
  - ‘wuapp.exe’

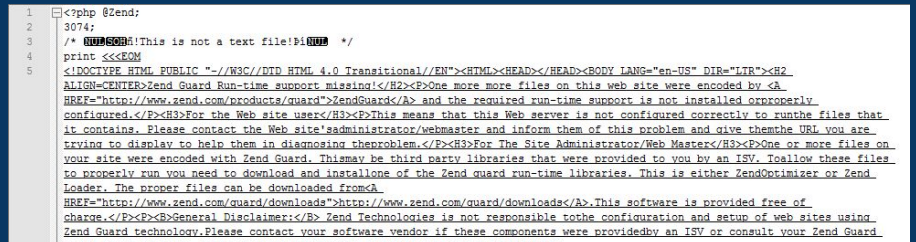


Wireshark’s Role in the Case

- Varonis Data Security Platform alerted of suspicious activity
- Abnormal web activity and correlated file behaviours flagged
- Wireshark used to monitor network communications and identify command-and-control patterns
- Application of Wireshark filters found that infected hosts were accessed by DuckDNS
    - DNS service to map dynamic IPs to static domains

Stopping the Attack

- Varonis team could see all IP addresses of the C&C servers the attackers were using with Wireshark.
- Revealed a PHP shell that continually connects to the C&C server.
- Found an XSL file being run by a known windows executable.
  - Was actually Zend Guard obfuscated PHP code.
  - Led to finding that the malware's payload is based on the execution in the XSL file.
- Allowed them to do further analysis by deciphering the XSL file.



Obfuscated XSL File

Conclusions

- Wireshark as a packet analysis tool
- Wireshark GUI & configuration
- Wireshark used for network administration and information security purposes
- Specific attacks and how to address them (Man-in-the-Middle, DNS Spoofing)
- Demonstrated how filters & analysis features can aid in detecting & mitigating threats
- Real-world cases showing Wireshark’s effectiveness in identifying and resolving network issues


WIRESHARK