Intro to Networking

By Jesse and Evan

What’s an internet

- Lots of computers messaging each other using the Internet protocol suite (TCP/IP)
- Network data is called a “packet”
- TCP/IP sends these packets from a source host to a destination host
    - *Not unlike you (source) sending a lovely box (packets) to your friend (destination) through the US Postal Service (internet)
    - Source host can also be referred to as a “client”, destination host as a “server”

*yes, people still do this

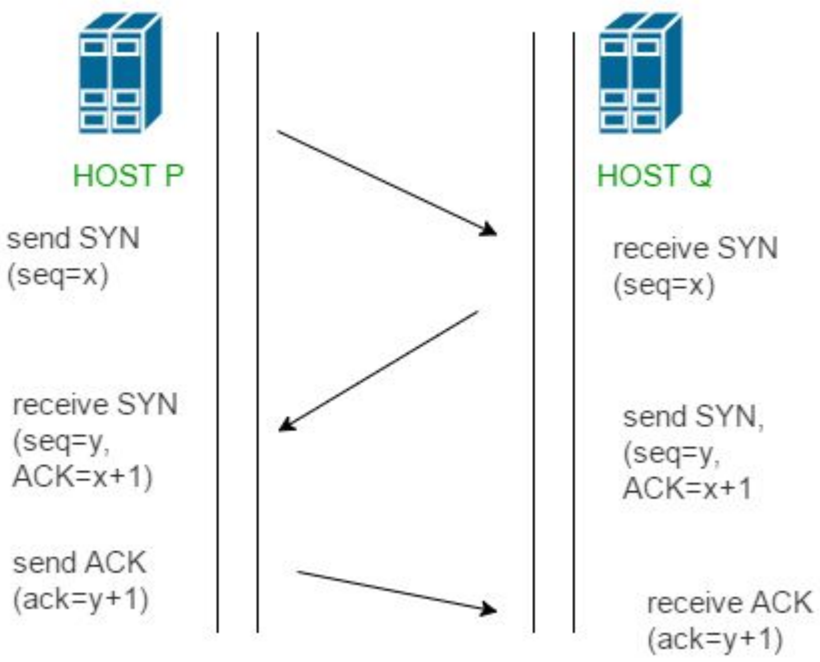
TCP/IP

- Internet Protocol (IP) - the packaging, contains:
    - Header
        - IP address of the source host and the destination host (the return address, the destination address)
    - Payload
        - Packets of data in a transport layer (the box)
        - TCP - reliable, ordered, error-checking
        - UDP - here’s some packets bye

TCP

- Reliable, ensures packets from actual source (client) get to destination (server)
  - 3-way handshake: source sends SYN, destination sends SYN-ACK, source sends ACK, yay successful connection
  - If a single step is missing connection is cut off
- Ordered: each packet is ordered, so if duplicate packets are sent/packets are missing it is known
- Congestion/flow control: doesn’t overwhelm network
- Used for anything where you NEED all data (downloads, SSH, webpages)

TCP Handshake



HOST P

HOST Q

send SYN
(seq=x)

receive SYN
(seq=x)

receive SYN
(seq=y,
ACK=x+1)

send SYN,
(seq=y,
ACK=x+1

send ACK
(ack=y+1)

receive ACK
(ack=y+1)

UDP

- The U stands for uncaring
    - Unreliable, unordered; slaps a destination with the packets
    - Does not want a response back
- Fast
- Used when you care about speed and getting *most* of the data (video games, media streaming)
- maybe ddos too

Netcat

- TCP/UDP utility knife
- Can be a client and a server
- Usage: nc [ip address] [port]
- Lots of flags but i don't know all of them
- nc -l -p [port] to open a TCP server on a port, useful for transferring files
- very cute

cheat sheet:

https://www.sans.org/security-resources/sec560/netcat_cheat_sheet_v1.pdf

Scapy

- Python-based, super flexible packet maker
- Essentially lets you create any TCP/IP packet
  - Including spoofing client ip address did somebody say ddos
- Usage:
  - RUN AS ROOT/SUDO
  - SYN: Create IP object, create TCP object, layer them and sr1()
  - If that didn’t make any sense go to link below

cheat sheet:
https://blogs.sans.org/pen-testing/files/2016/04/ScapyCheatSheet_v0.2.pdf

Downloading Scapy

clone the scapy repo at https://github.com/secdev/scapy OR:

Ubuntu/Linux: pip install scapy

Mac: can use homebrew or macports, you also need to install Python bindings

Windows: why are you using windows

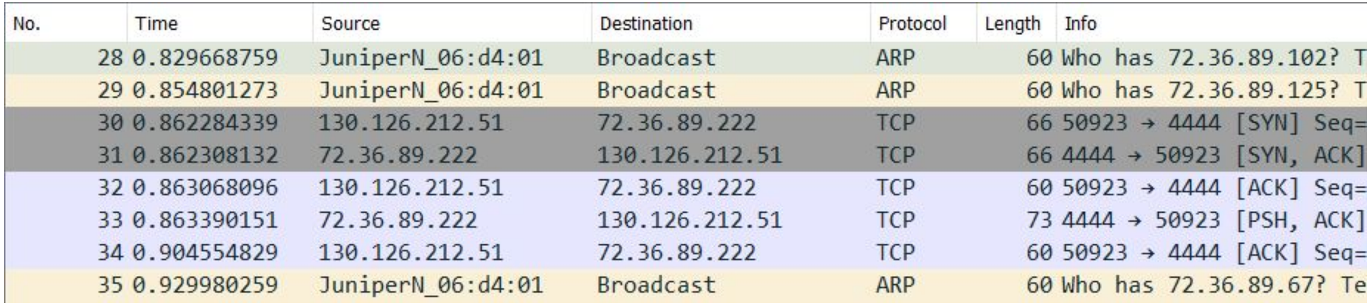
https://scapy.readthedocs.io/en/latest/installation.html

*As scapy sends out network packets locally (not through the system), your operating system might not like that, and sends out RST packets to block them. You need to tell your system to shut up and not do that. On Linux, configure iptables (iptables -A OUTPUT -p tcp --tcp-flags RST RST -j DROP), On mac/windows you might need to change firewall settings (?), not sure, should probably look it up

Wireshark

- This tool is crazy
- Captures and analyzes network traffic, stores it in a .pcap file
- Lets you know:
    - Timestamp of packet, source of packet, destination of packet, protocol of packet, size of packet, what’s in the packet, insert funny joke here
- tshark on terminal, tcpdump is similar but no pretty colors
- only for REAL hackers that speak in hex
    - Can do lots of not quite legal stuff with it

Analyzing with Wireshark



No.	Time	Source	Destination	Protocol	Length	Info
28	0.829668759	JuniperN_06:d4:01	Broadcast	ARP	60	Who has 72.36.89.102? T
29	0.854801273	JuniperN_06:d4:01	Broadcast	ARP	60	Who has 72.36.89.125? T
30	0.862284339	130.126.212.51	72.36.89.222	TCP	66	50923 → 4444 [SYN] Seq=
31	0.862308132	72.36.89.222	130.126.212.51	TCP	66	4444 → 50923 [SYN, ACK]
32	0.863068096	130.126.212.51	72.36.89.222	TCP	60	50923 → 4444 [ACK] Seq=
33	0.863390151	72.36.89.222	130.126.212.51	TCP	73	4444 → 50923 [PSH, ACK]
34	0.904554829	130.126.212.51	72.36.89.222	TCP	60	50923 → 4444 [ACK] Seq=
35	0.929980259	JuniperN_06:d4:01	Broadcast	ARP	60	Who has 72.36.89.67? Te

Source: Where the packet came from

Destination: Where the packet is going

Protocol: The layering (TCP, UDP, ARP, and more; in this case, we want TCP)

Length: Packet size

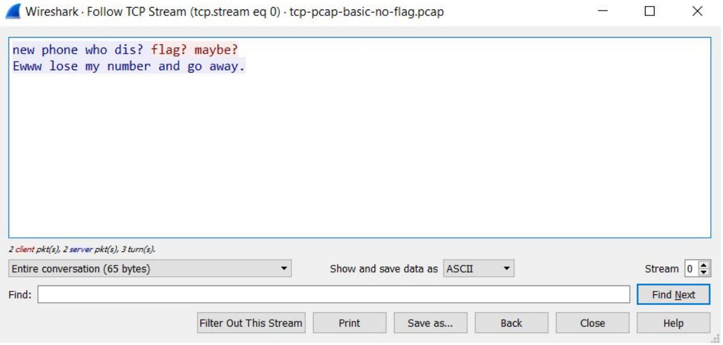
Info: What’s in the packet (ports, flags - SYN, SYN/ACK, ACK, etc., data, etc.)

“Follow” on Wireshark

Go to “Analyze” -> “Follow” -> “TCP Stream” (or Ctrl-Alt-Shift-T)

Blue are data in packets sent by server,
red are packets sent by client

Click up-down arrows next to Stream to go through all “conversations”



Wireshark · Follow TCP Stream (tcp.stream eq 0) · tcp-pcap-basic-no-flag.pcap

```
new phone who dis? flag? maybe?
Ewww lose my number and go away.
```

2 client pkt(s), 2 server pkt(s), 3 turn(s).

Entire conversation (65 bytes)

Show and save data as ASCII

Stream 0

Find:

The end

thanks

go do the challenges now

ask someone old or me if you need help