CSci 4271W
Development of Secure Software Systems
Day 14: Networking and security: what can go wrong?

Stephen McCamant (he/him)

University of Minnesota, Computer Science & Engineering

Based in large part on slides originally by Prof. Nick Hopper

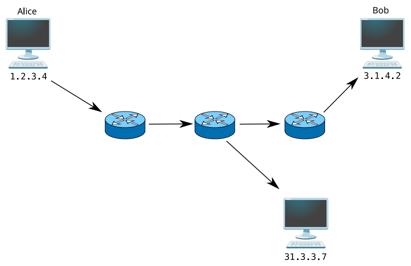
Networks and software

What could go wrong as a result of the following program?

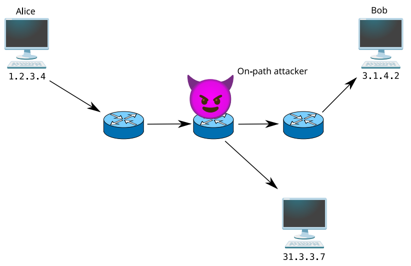
```
import urllib.request
with urllib.request.urlopen("http://neverssl.com/") as f:
  for line in f: print(line)
```

The application request is encoded into a transport connection divided over a sequence of datagrams that are sent from host to host in a series of frames. . .

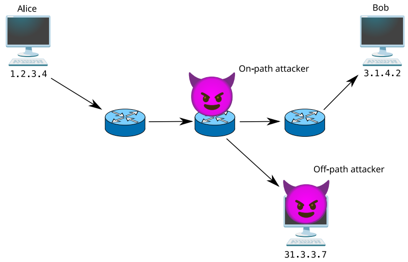
Network Attackers



Network Attackers

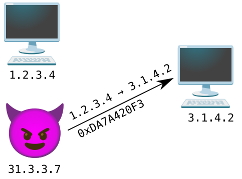


Network Attackers



Spoofing

Networks generally cannot enforce a correct sender address.



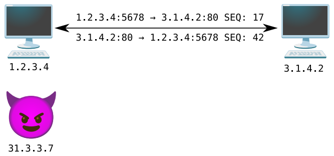
Protocols generally try to defeat this with weak secrets (port, identifier)

Not effective against on-path attackers.

Spoofing of ARP, routing, and DNS can also give off-path attacks.

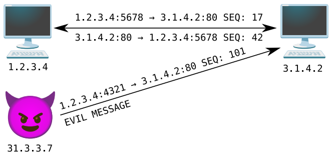
Example: TCP

A TCP connection has two (16-bit) port numbers and two (32-bit) sequence numbers: client and server


1.2.3.4:5678 → 3.1.4.2:80 SEQ: 17
3.1.4.2:80 → 1.2.3.4:5678 SEQ: 42
1.2.3.4
3.1.4.2
31.3.3.7

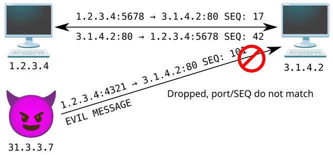
Example: TCP

A TCP connection has two (16-bit) port numbers and two (32-bit) sequence numbers: client and server


1.2.3.4:5678 → 3.1.4.2:80 SEQ: 17
3.1.4.2:80 → 1.2.3.4:5678 SEQ: 42
1.2.3.4
3.1.4.2
1.2.3.4:4321 → 3.1.4.2:80 SEQ: 101
EVIL MESSAGE
31.3.3.7

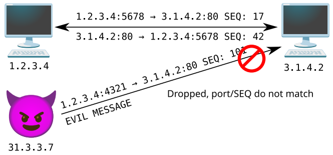
Example: TCP

A TCP connection has two (16-bit) port numbers and two (32-bit) sequence numbers: client and server


1.2.3.4:5678 → 3.1.4.2:80 SEQ: 17
3.1.4.2:80 → 1.2.3.4:5678 SEQ: 42
1.2.3.4
3.1.4.2
1.2.3.4:4321 → 3.1.4.2:80 SEQ: 101
EVIL MESSAGE
Dropped, port/SEQ do not match
31.3.3.7

Example: TCP

A TCP connection has two (16-bit) port numbers and two (32-bit) sequence numbers: client and server

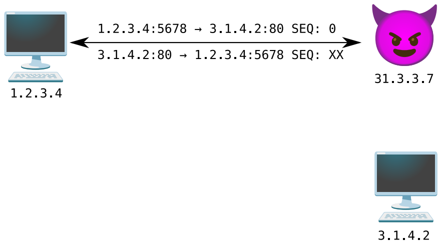

1.2.3.4:5678 → 3.1.4.2:80 SEQ: 17
3.1.4.2:80 → 1.2.3.4:5678 SEQ: 42
1.2.3.4
3.1.4.2
1.2.3.4:4321 → 3.1.4.2:80 SEQ: 101
EVIL MESSAGE
Dropped, port/SEQ do not match
31.3.3.7

Old problem: client ports assigned sequentially, and initial sequence number (ISN) was 0

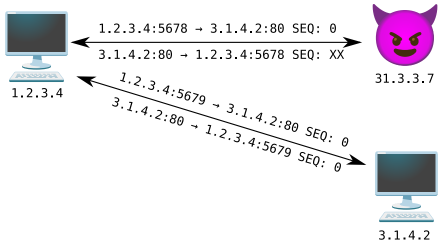
(Old) TCP attack

1.2.3.4
31.3.3.7
3.1.4.2

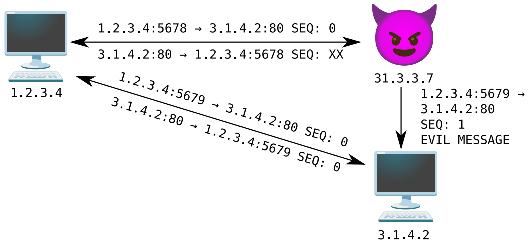
(Old) TCP attack


1.2.3.4:5678 → 3.1.4.2:80 SEQ: 0
3.1.4.2:80 → 1.2.3.4:5678 SEQ: XX
1.2.3.4
31.3.3.7
3.1.4.2

(Old) TCP attack

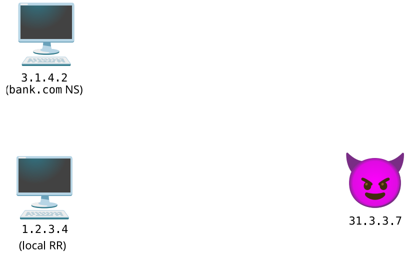

1.2.3.4:5678 → 3.1.4.2:80 SEQ: 0
3.1.4.2:80 → 1.2.3.4:5678 SEQ: XX
1.2.3.4
31.3.3.7
1.2.3.4:5679 → 3.1.4.2:80 SEQ: 0
3.1.4.2:80 → 1.2.3.4:5679 SEQ: 0
3.1.4.2

(Old) TCP attack


1.2.3.4:5678 → 3.1.4.2:80 SEQ: 0
3.1.4.2:80 → 1.2.3.4:5678 SEQ: XX
1.2.3.4
31.3.3.7
1.2.3.4:5679 → 3.1.4.2:80 SEQ: 1 EVIL MESSAGE
1.2.3.4:5679 → 3.1.4.2:80 SEQ: 0
3.1.4.2:80 → 1.2.3.4:5679 SEQ: 0
3.1.4.2

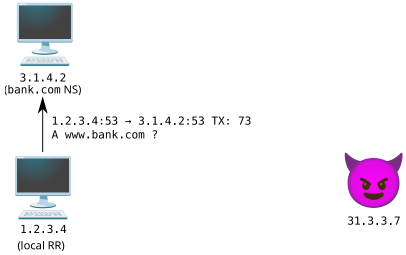
Example: DNS

A DNS query has a 16-bit transaction ID



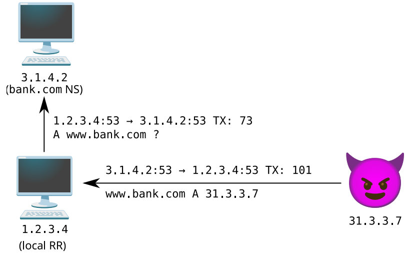
Example: DNS

A DNS query has a 16-bit transaction ID



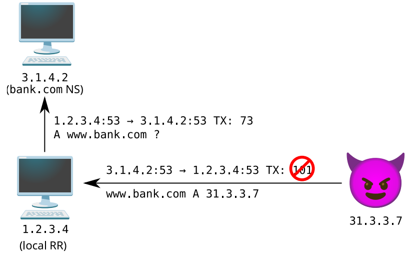
Example: DNS

A DNS query has a 16-bit transaction ID



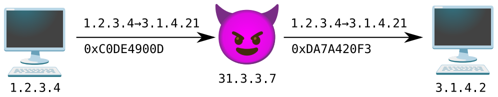
Example: DNS

A DNS query has a 16-bit transaction ID



Tampering

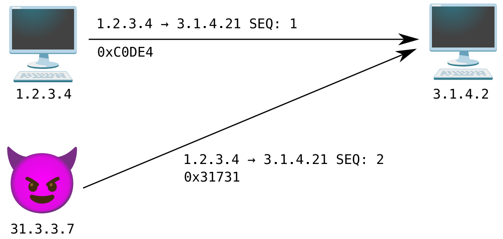
On-path attackers can modify packets



“Checksums” are easy to compute and modify; cryptography is needed to defend against this attack.

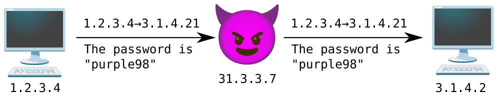
Tampering (cont’d)

Off-path attackers can potentially inject packets



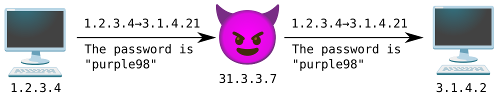
Information disclosure

On-path attackers can see contents and addresses of packets

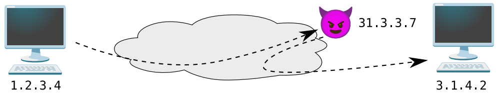


Information disclosure

On-path attackers can see contents and addresses of packets



Route spoofing can lead to disclosure to (formerly) off-path attackers



Outline

Network STI

Announcements intermission

Network DoS

Upcoming assignments

- Project 1 regular due date 11:59pm tonight on Gradescope
- Homework 4 (mostly networking) due next Tuesday 3/25
  - Material for questions 1–3 covered through today
- Midterm 2 will be a week from Thursday, 3/27

Outline

Network STI

Announcements intermission

Network DoS

Denial of service

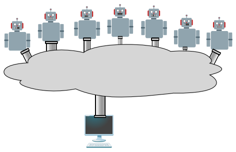
Off-path network DoS attacks generally fall into two categories:

- Distributed Denial of Service (DDoS): the "brute force" attack
- Protocol-Based DoS: attempt to deny service with as few packets as possible

DDoS

Is the brute force attack that just sends more bandwidth than victim's network can carry:

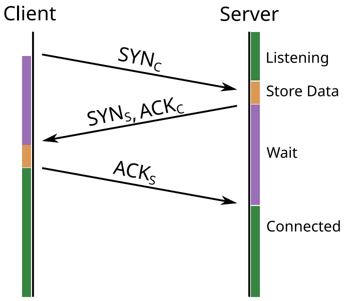
1. Acquire botnet (compromised hosts)
2. Point at `www.victim.com`
3. . . .
4. Profit!



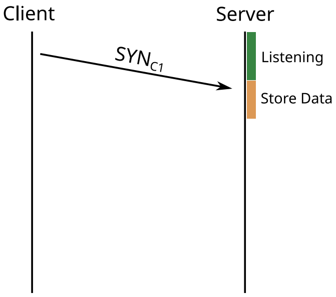
Protocol DoS attacks

- Can we convince a host to stop sending or receiving data?
- Can we prevent data from being delivered?
- Can we get other hosts to "help" with an attack?

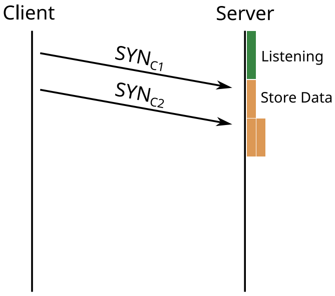
TCP handshake



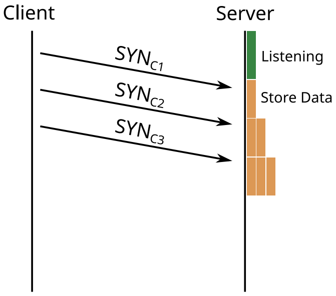
SYN flooding



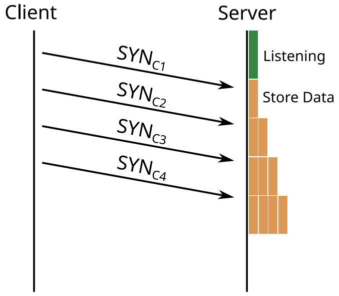
SYN flooding



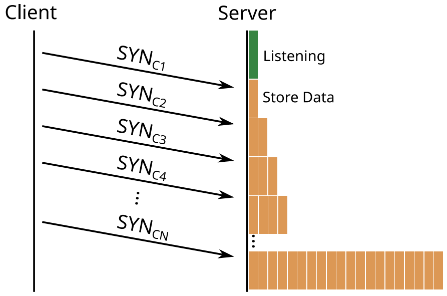
SYN flooding



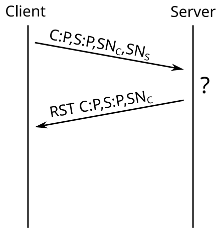
SYN flooding



SYN flooding



TCP reset



A TCP connection can end in three ways:

- FIN (orderly close)
- timeout
- RST (abrupt termination)

A client will accept a RST within one TCP window (typically 65K) of the last SN$_C$.

Predictable ISNs and large windows compound this problem.

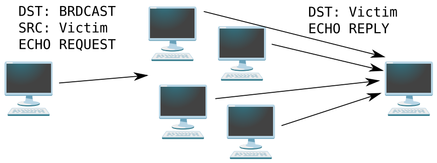
Control channel DoS

Many protocols have a narrow “control channel” that enables further communication and is a DoS target:

- In 802.11, wireless nodes send “RTS” message and listen for “CTS” before sending
- Bittorrent clients have to download a `.torrent` file before connecting to tracker or joining a swarm
- VoLTE uses a single SIP server to connect all callers
- Route spoofing can be used for DoS as well

Amplification

Amplification attacks are DoS attacks that use “reflectors” to send more data than the originator sends. Typically enabled by spoofing of victim address as source.



STRIDE leftovers

- Repudiation
  - Manipulation of packet information can create inaccurate logs
- Elevation of privilege
  - Off-path to on-path, routing and redirection attacks? Similar in allowing attack chaining, but carrying traffic isn’t a privilege
  - Remote code execution attacks? Network stacks are software, but that’s a software issue