Network Security

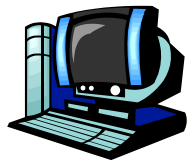
Spring 2025

TCP 3-way handshake

TCP SYN

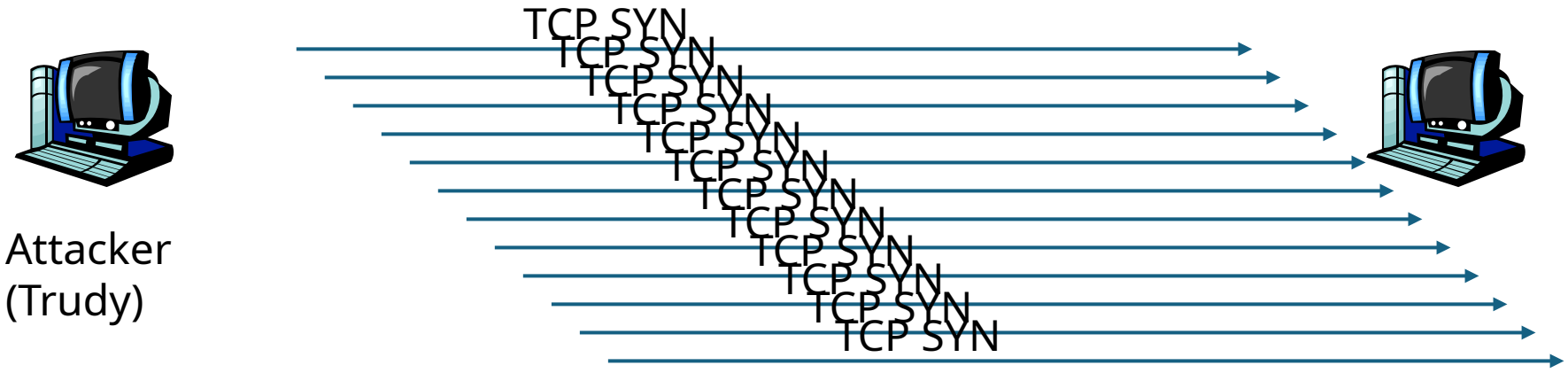
TCP SYN/ACK

TCP ACK



When host receives a TCP SYN
- Open sockets
- Allocating resources (CPU/
  to prepare for the connect

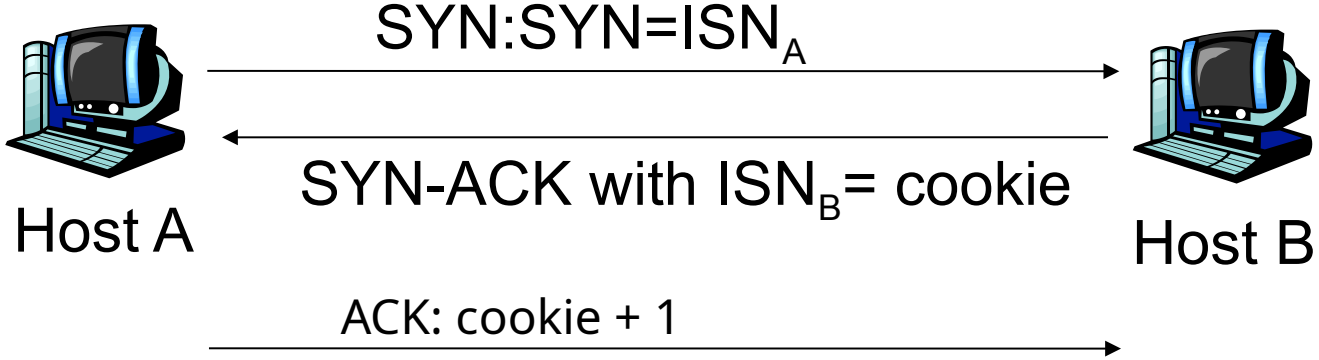
Task 1a. SYN Flood Attack



Size of a TCP SYN Packet: 40 bytes (IP header=20b + TCP header 20b)

TCP SYN Flood attacks are very easy to perform, and a normal computer has 40 available sockets

Task 1b: mitigate a SYN Flood attack: SYN Cookies



With SYN Cookie protections are on:
When Host B received a TCP SYN

- It does NOT allocate resources yet
- It sends a SYN Cookie

Only when host B received the SYN cookie back
Does it allocate a socket for the connection

- 726 Broadway, 8th Floor – ITS
- 1 Metrotech Center, 22nd Floor
- noc-its18-arin@nyu.edu
- noc-cosi-arin@nyu.edu
- +1-212-998-34
- +1-212-998-344431

- 216.165.0.0/17
- AS12 (128.122.0.8)

- Engineering.nyu.edu -> hosted by AWS
- Brightspace.nyu.edu -> nyu.Brightspace.com -> AWS
- Cyber.nyu.edu -> AWS
- Albert.nyu.edu -> 216.165.62.30
- Hosting.nyu.edu -> DigitalOcean
- Stream.nyu.edu -> kltura.com -> cloudflare.net

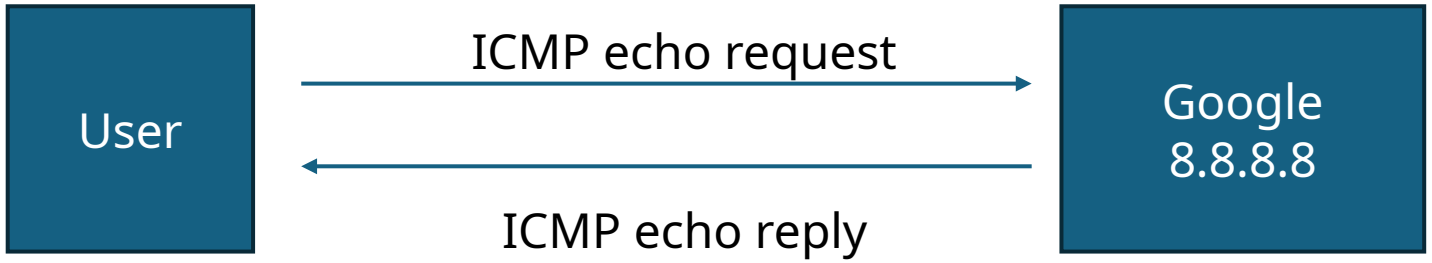
- Mail server: pphosted.com

DNS Record Types

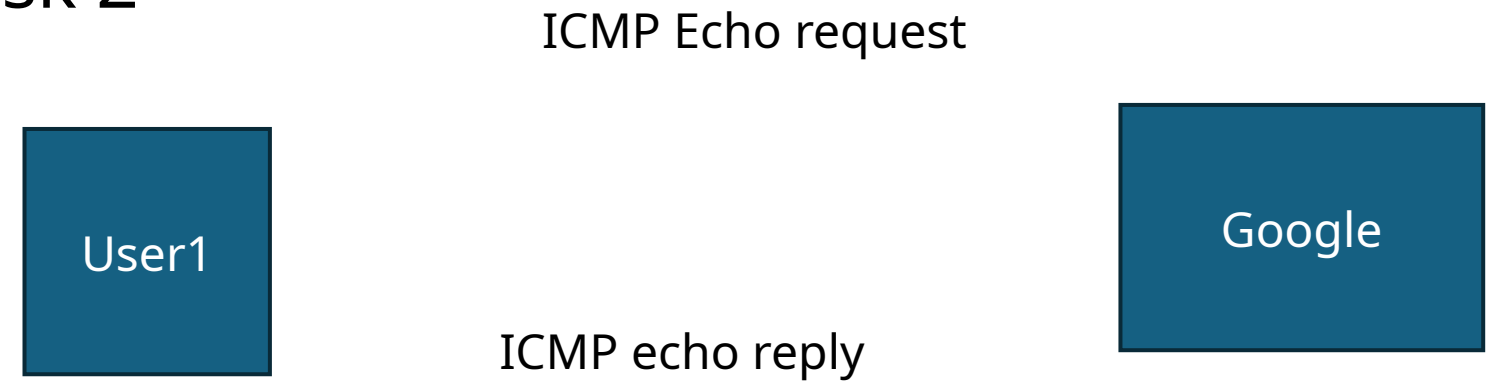
- A records – IPv4 IP addres
- AAAA – IPv6 IP address
- TXT – “free text”, often used for email security (SPF)
- CNAME – similar to alias
- NS – DNS servers for the domain
- MX – mail server

Emails to NYU

ICMP echo request/reply



Lab 2 Task 2



Lab 2 Task 3 – write your own traceroute program

SEED machine

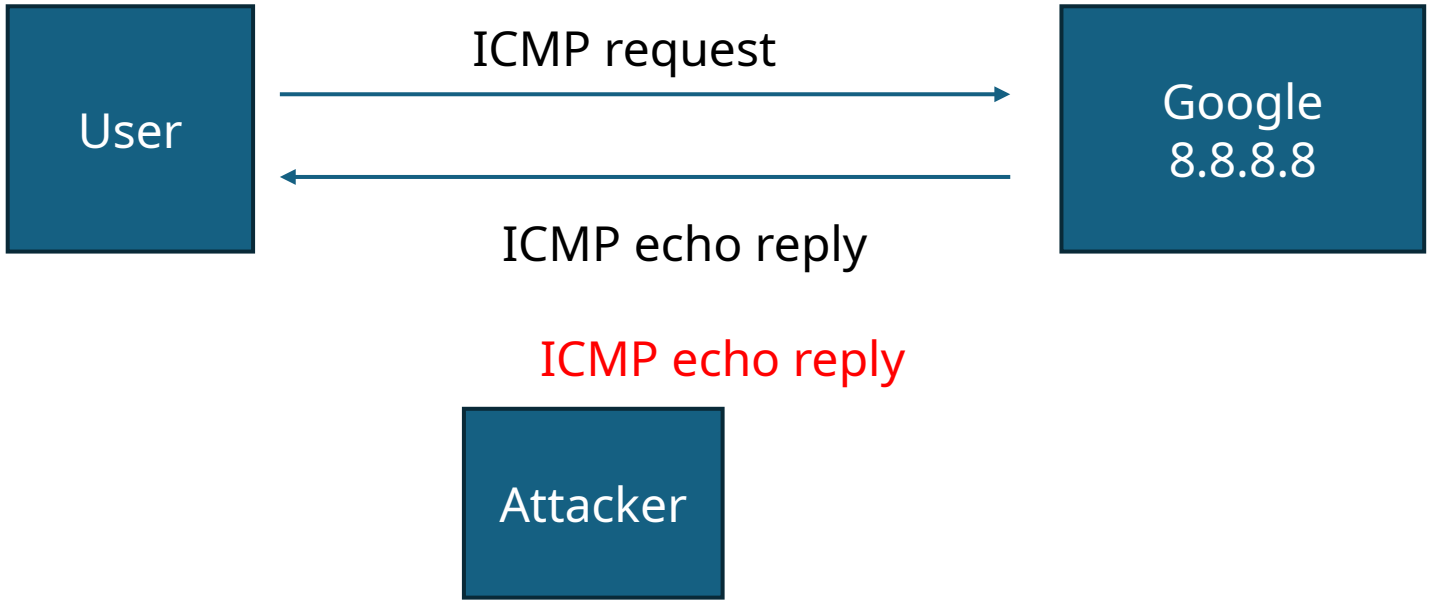
Google DNS Server
8.8.8.8

1. Icmp ping TTL 1
2. Icmp ping TLL 2
3. ..
4. ...

Task 3, the sample code does not handle when there is no reply properly (program will hang)

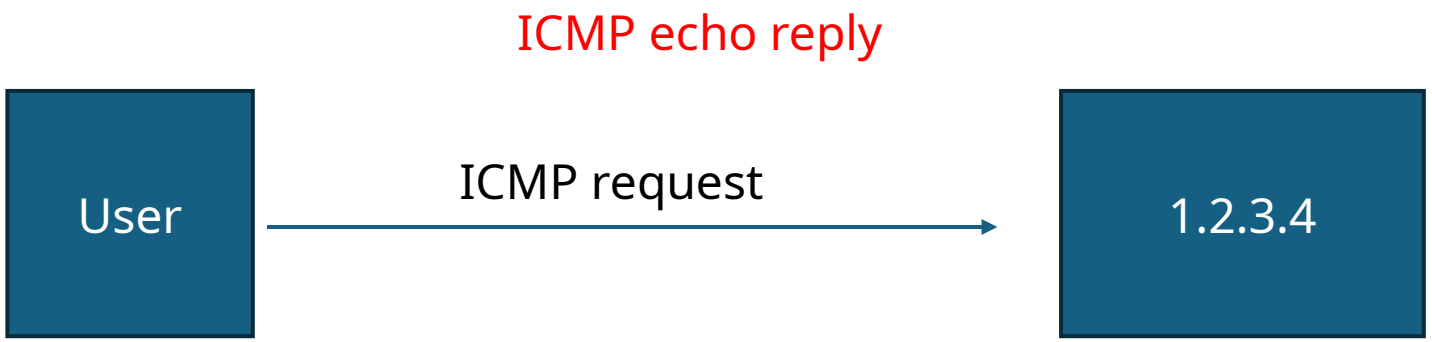
Lab 2 Task 4

Part 3



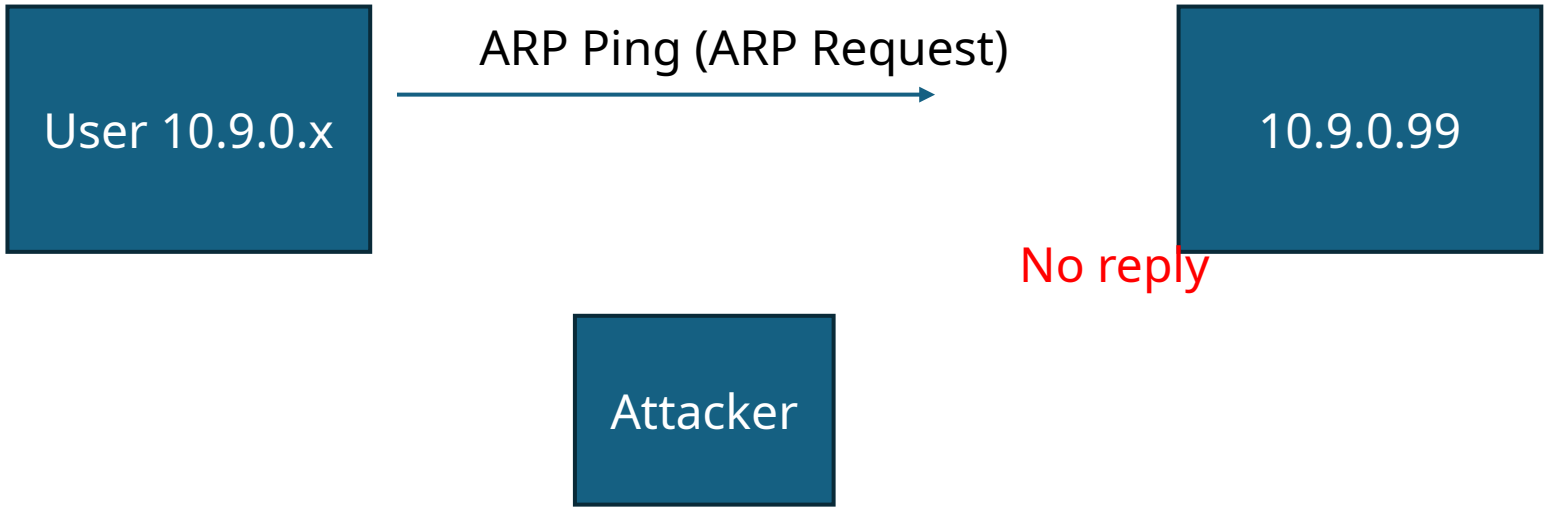
When the user tries to ping 8.8.8.8, the user will receive two icmp echo replies:
1. is from the real google DN
2. Is from the attacker (that's you)

Part 1



When the user tries to ping 1.2.3.4
There is no response (probably host doesn't exist)

Part 2


User 10.9.0.x
ARP Ping (ARP Request)
10.9.0.99
No reply
Attacker
(sniffing ICMP echo request)

TCP Port scanning (1)

Normal process for
Port 443 is open

TCP SYN dport 443

TCP SYN/ACK

TCP ACK

Port 443 is closed
Gets a TCP RST

TCP SYN dport 443

TCP RST

TCP port scanning (2)

P destination unreachable
etworking error (route not
und, host does not exist)
ejected" by a Firewall

TCP SYN dport 443

ICMP destination unreachable

ponse
working error (route not
nd, host does not exist)
p" by a Firewall

TCP SYN dport 443

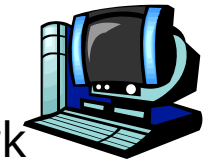
(no response)

REJECT or DROP

- REJECT
  - Firewall is being nice, is sending a ICMP destination unreachable
- DROP
  - The packet is ”being sent to the bit bucket”
  - Firewall is not responding at all

Nmap SYN Scan vs. Connect Scan

N Scan
Nmap has elevated privilege
Can write directly to the network



TCP SYN dport 443

(nmap listens for a response)

TCP SYN dport 443

TCP SYN/ACK

TCP ACK

Connect Scan
map does not have elevated privileges
ust use OS API (sockets) to open/close ports

TCP FIN

TCP FIN/ACK

TCP ACK

Nmap ACK Scan – only works on stateless firewalls

Flags: SYN/ACK

Flags: SYN/ACK

FW

TCP port 443

RST

ttacker tries to
onnect to port 443
CP SYN will get
ocked

When an attacker tries to connect to host in port 443,
It will be blocked by the FW, because it only allows outgoing
Connects to port 443, not incoming connections

Attacker connect to port 443
Using TCP ”ACK” flag only

When the attacker tries to connect to port 443,
Using TCP “ACK” packet, it will be allowed through the firew

Attackers pivot to other hosts

UDP Scanning

open
request, get a response

UDP DNS port 53

DNS Response

ed
nation unreachable is received

UDP DNS port 53

ICMP Destination Unreachable

*

esponse
rewall is blocking port
nproperly formatted request
MP Destination unreachable limit
- Limited: 1/s in Linux or 2/s in Winodws

UDP DNS port 53

(no response)

* don’t get confused with TCP ICMP unreachable

ICMP Destination Unreachable

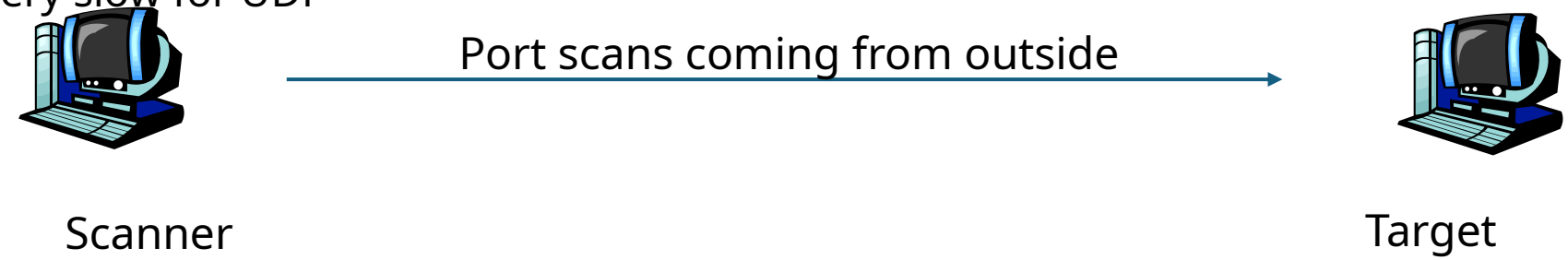
- ICMP Destination Unreachable are limited
  - Windows: 2 times / sec
  - Linux 1 time / sec
  - If you sent two UDP packets to two closed ports at the same time, only the first one will response. The 2nd response is DROPed

Victim

Trudy sends a broadcast request

Remote Scanning vs. Agent-based scanning

- Remote scanning
  - Traditional scanning method
  - It is very slow for UDP



- Agent-based scan
  - Must better and faster
  - Can list all ports/services quickly
  - Cost is approximately 10x more

SYN Cookies

TCP SYN
SEQ: ISN A (random number)
ACK: 0 (literal “0”, b/c there’s no ACK flag set)

TCP SYN/ACK
SEQ: ISN B (SYN cookie)
ACK: A+1

TCP ACK
SEQ: A+1
ACK: B+1 (SYN_cookie + 1)

When SYN Cookies are enabled
Upon receipt of a TCP SYN
The host will generate a SYN Coo

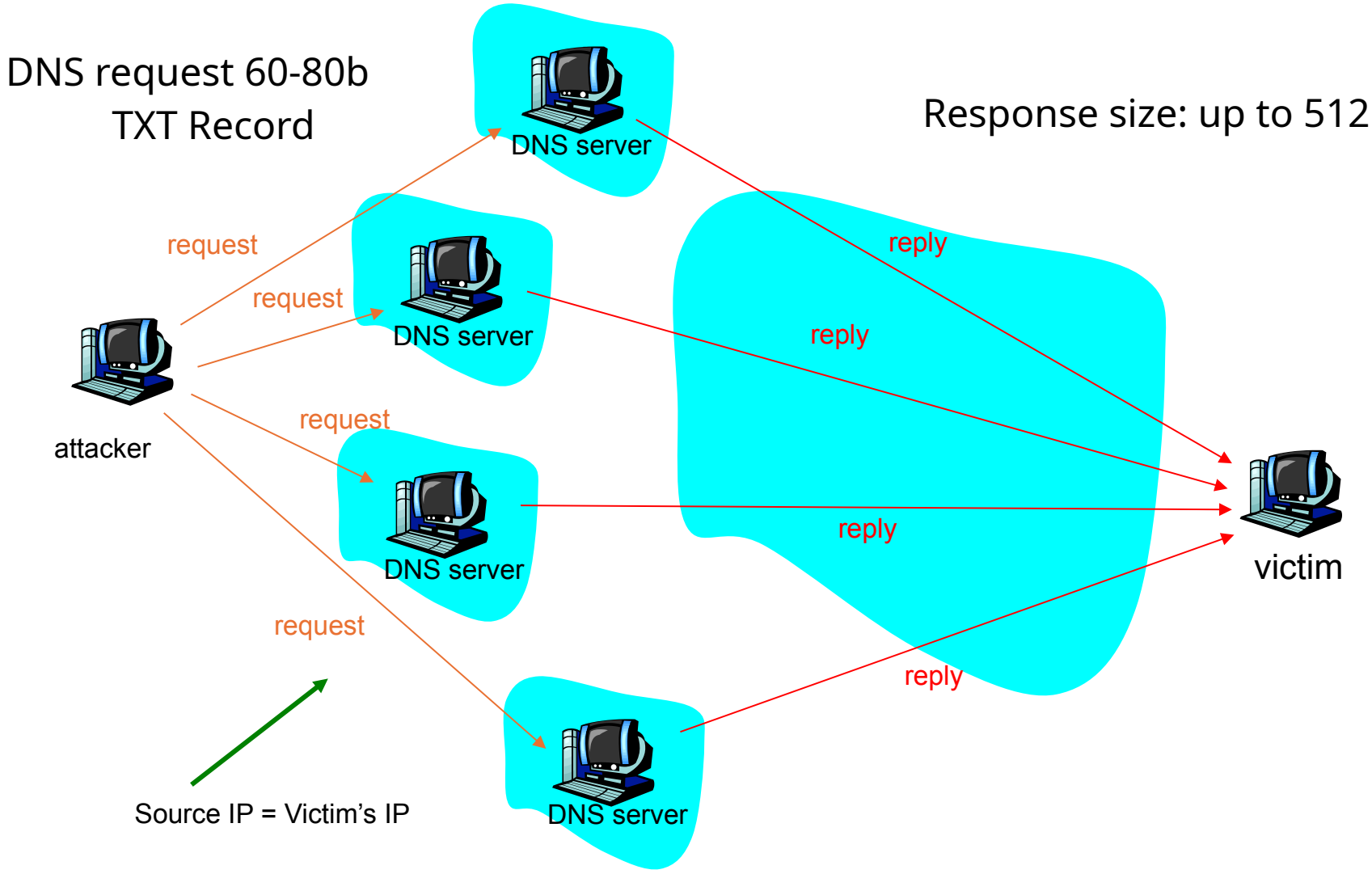
SYN_cookie =
md5(IPs,ports,slow_time,magic)

All the info is forgotten.

If a SYN_cookie is returned in the TCP ACK packet

Host will recalculate the SYN_cookie, and see if it matches the SYN_cookie from the TCP ACK
Only if the correct SYN cookie is

DNS Amplification/Reflection attack

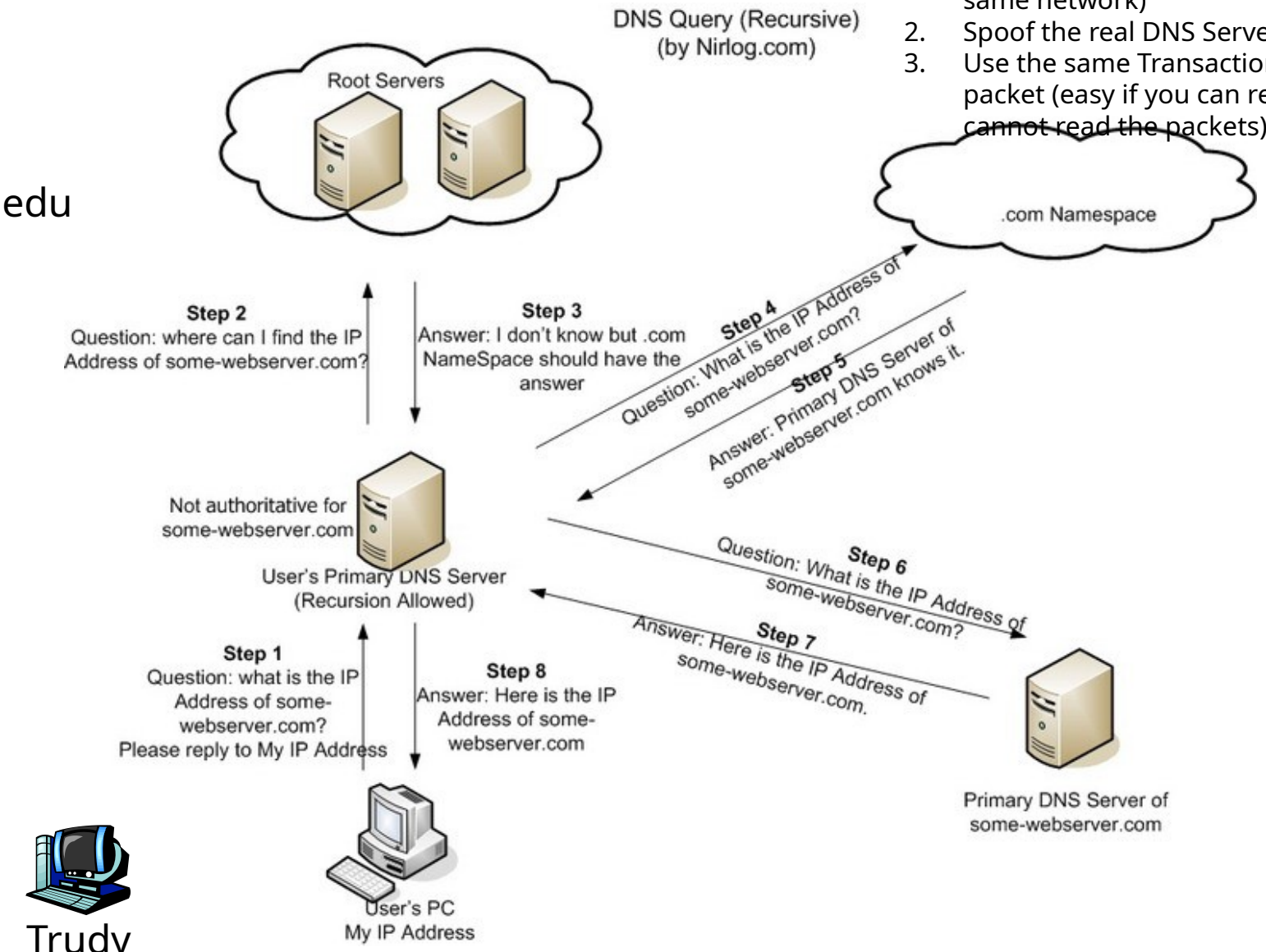


Interlude: How DNS Works

For the attacker to perform a DNS cache poisoning, the attacker must:

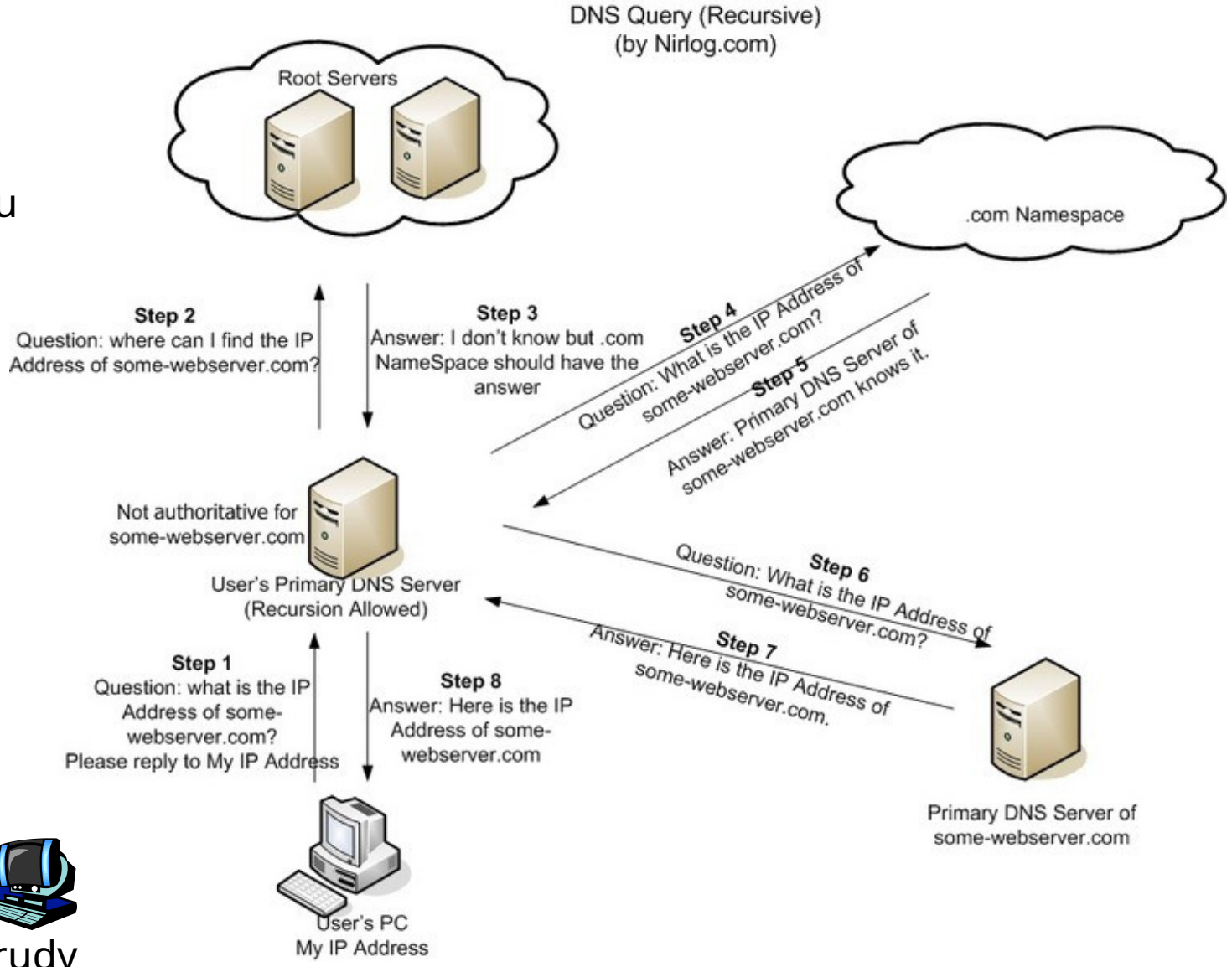
1. Must respond back faster than the real DNS server (very easy if on the same network, hard if not on the same network)
2. Spoof the real DNS Server's IP address (very easy)
3. Use the same Transaction ID & port# as the DNS packet (easy if you can read the packets, hard if you cannot read the packets)

Fake engineering.nyu.edu



Interlude: How DNS Works

Fake engineering.nyu.edu

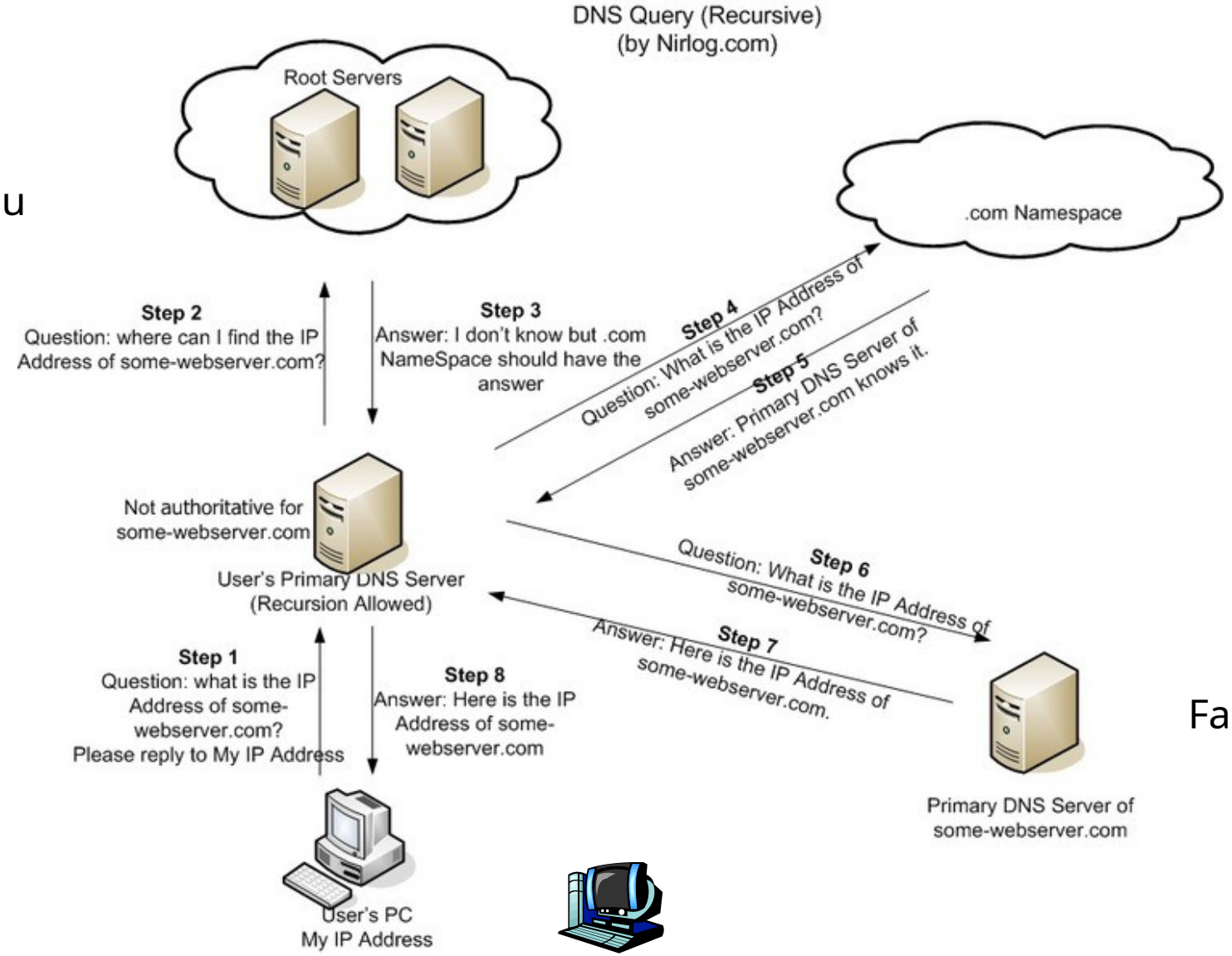


Trudy

Atta

Interlude: How DNS Works

Fake engineering.nyu.edu



Fake NYU.edu NS server

Another victim

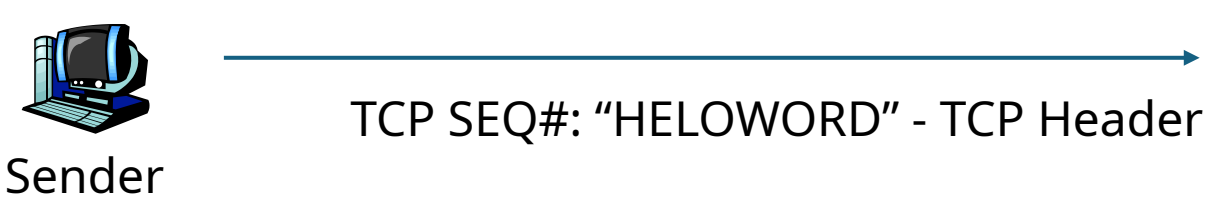
Trudy

Cover_tcp

- 1. IP ID method – random 16-bit number in the IP header, used for reassembling fragmented packets.

Every IP packet has a IP ID #. The sender can choose a new IP ID per packet

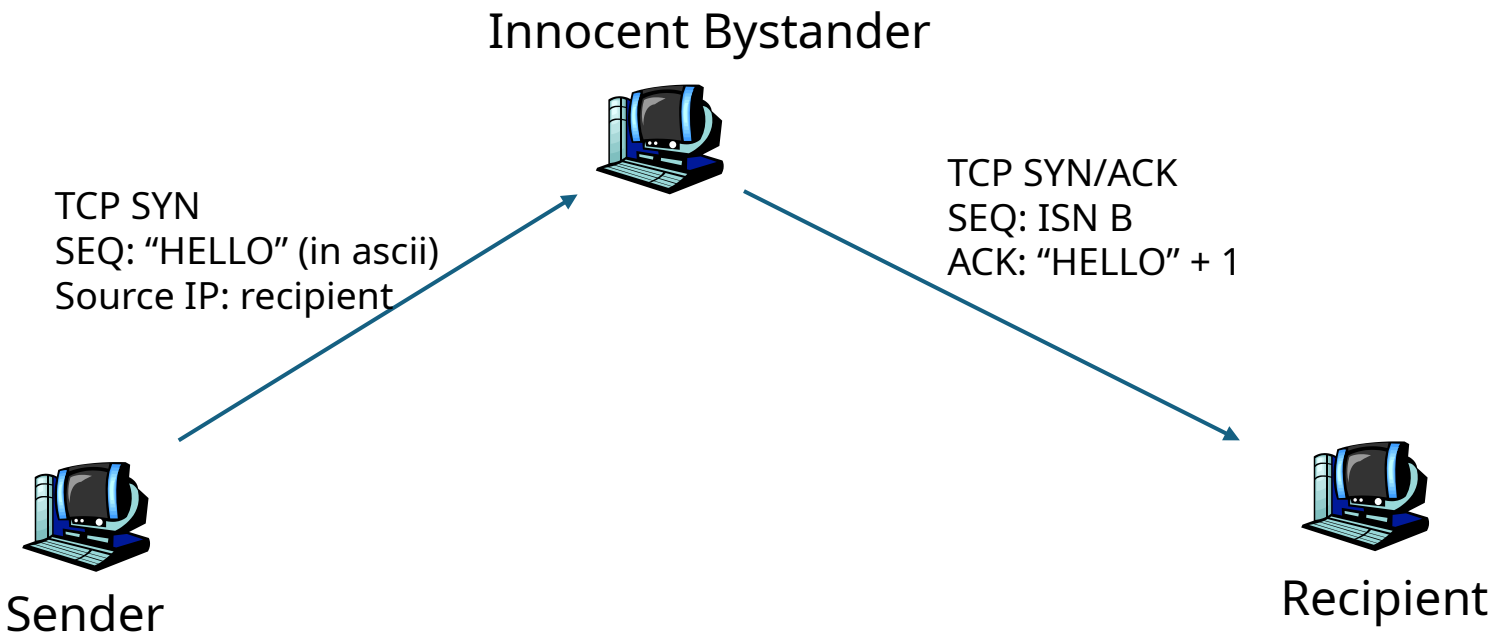
- 2. SEQ# Method (32-bits)



Every new TCP connection has a SEQ number chooses initially (called a ISN).

Cover_tcp (cont)

- 3. ACK # Method



Recipient, upon receiving “HELLO” + 1, knows that the original message was “HELLO”