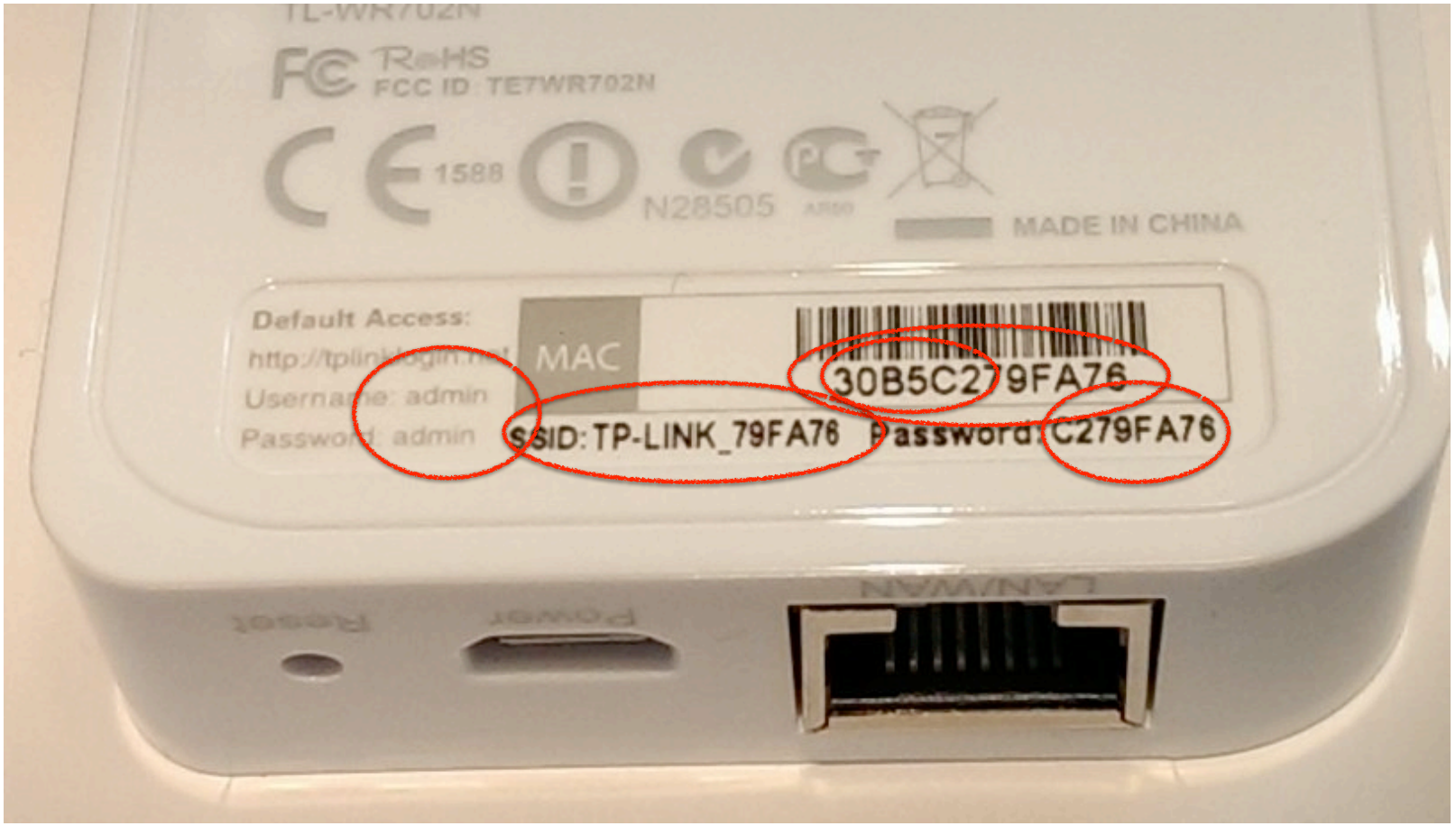
Network #3: TCP/IP

Spot the Zero Day:
TPLink Miniature Wireless Router

Spot the Zero Day:
TPLink Miniature Wireless Router



Nick's Apology...

- I'm really going to try to slow down
  - I'm also really going to try to reduce the "story factor" and check my ego
- Many thanks for the feedback!
  - And a beg: Don't wait for us to request feedback to give it!
  - When I'm going too fast or otherwise being a bad professor, PLEASE TELL ME
    - You're all smart, if you want anonymity in feedback you can
    - But be smarter: I want students to feel comfortable in telling me my screwups!

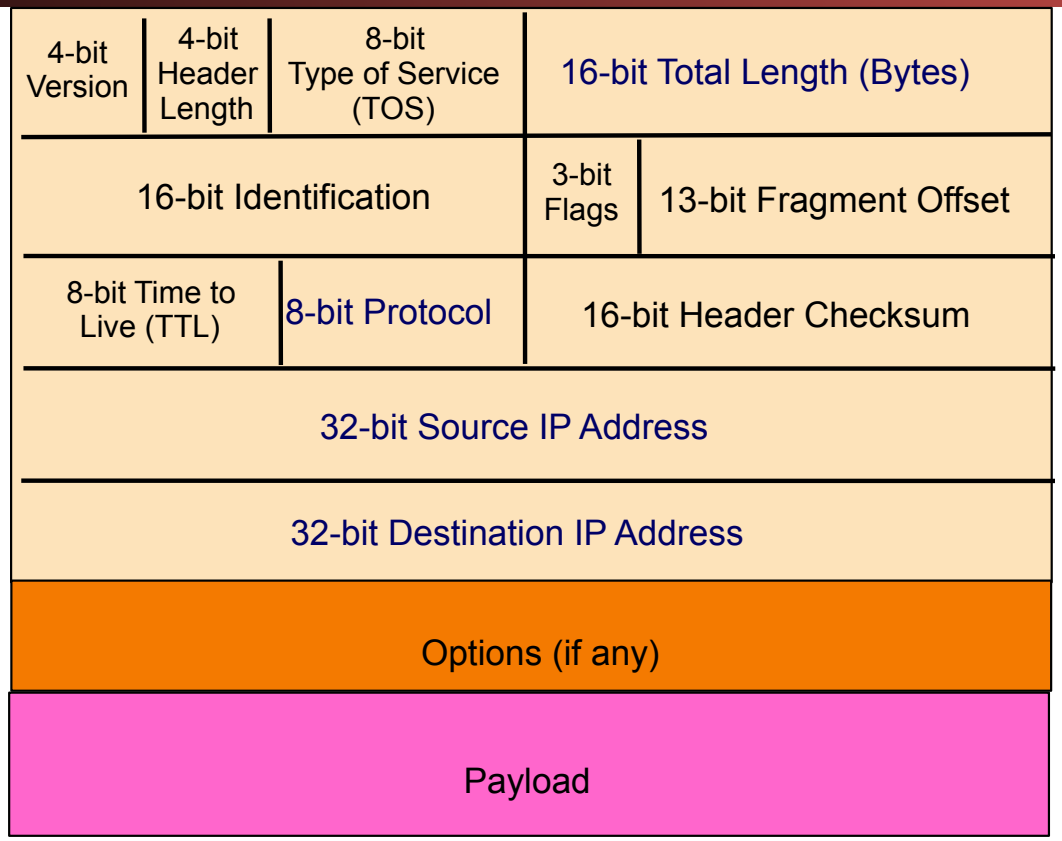
Review: VERY key topics

- Network is layered
- Wired/Wireless Network: addressed by Ethernet MAC
  - Broadcast or switched networks
  - WiFi encryption handshake
  - ARP/DHCP configuration
- Packet injection attacks
  - When the attacker sees a request...
- DNS
  - Distributed database, hierarchical trust
  - Attacks: Old-school cache poisoning, blind injection poisoning, race condition attacks (race once vs race-until-win)

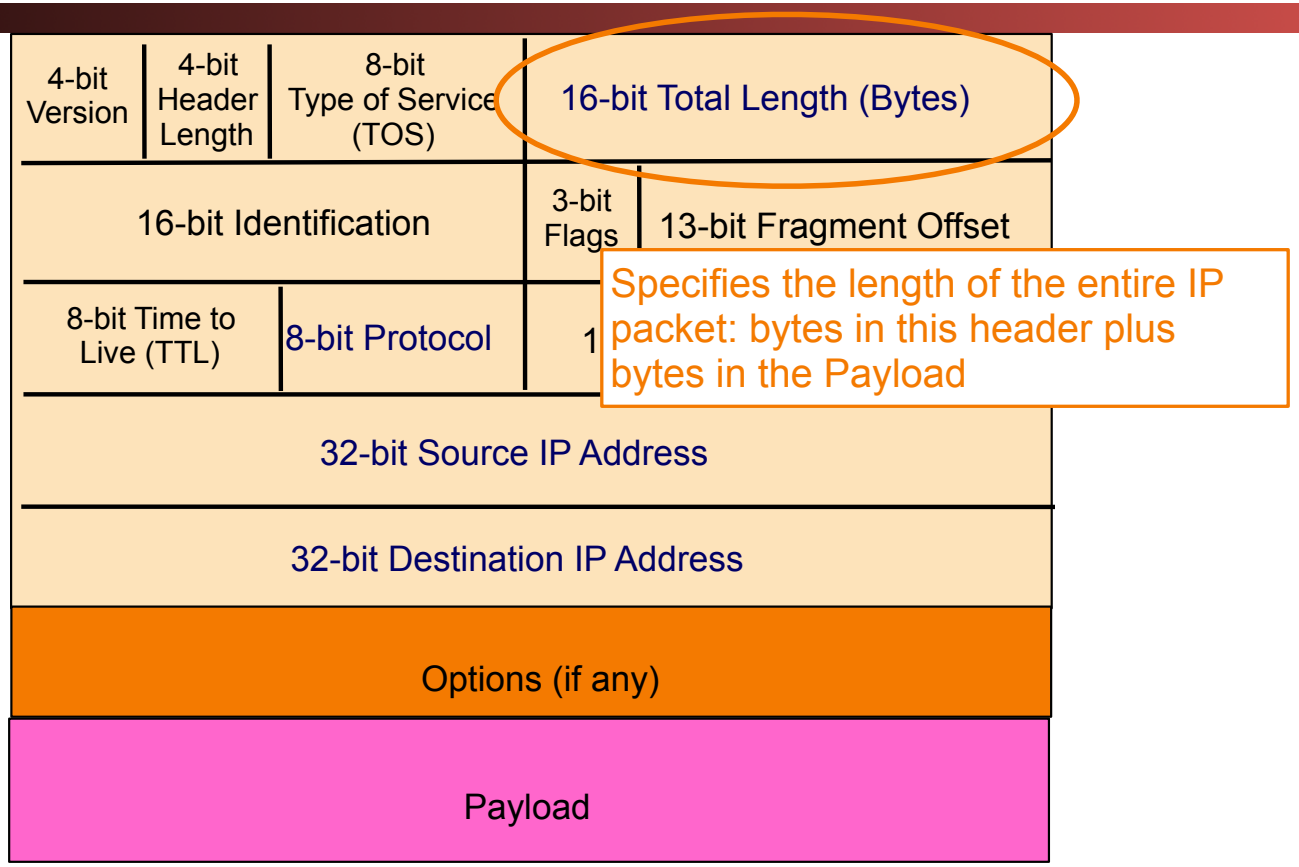
Today:
The Internet

- How the Internet routes IP packets
  - Distributed trust through Autonomous Systems
- How TCP works
- Denial of Service Attacks
- (If time) the Firewall #1

IP Packet Structure

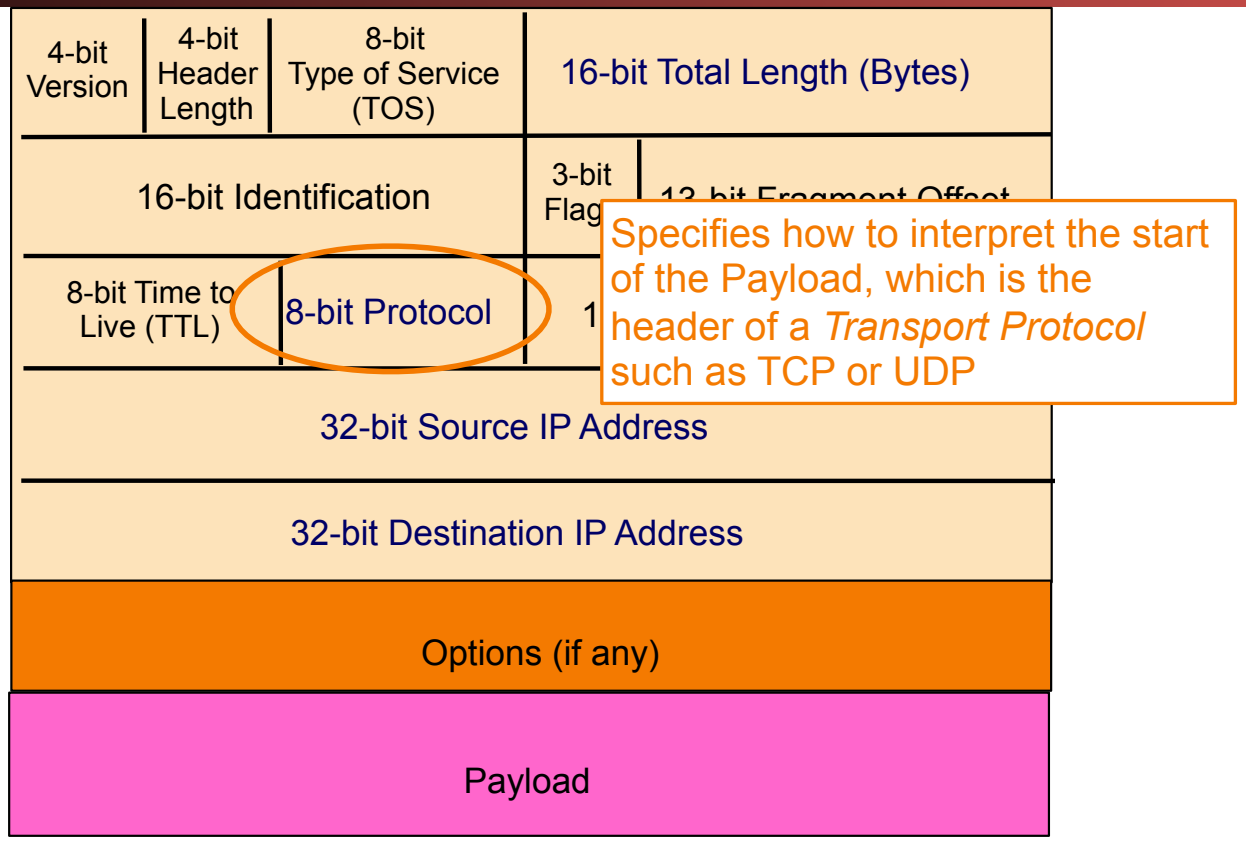


IP Packet Structure

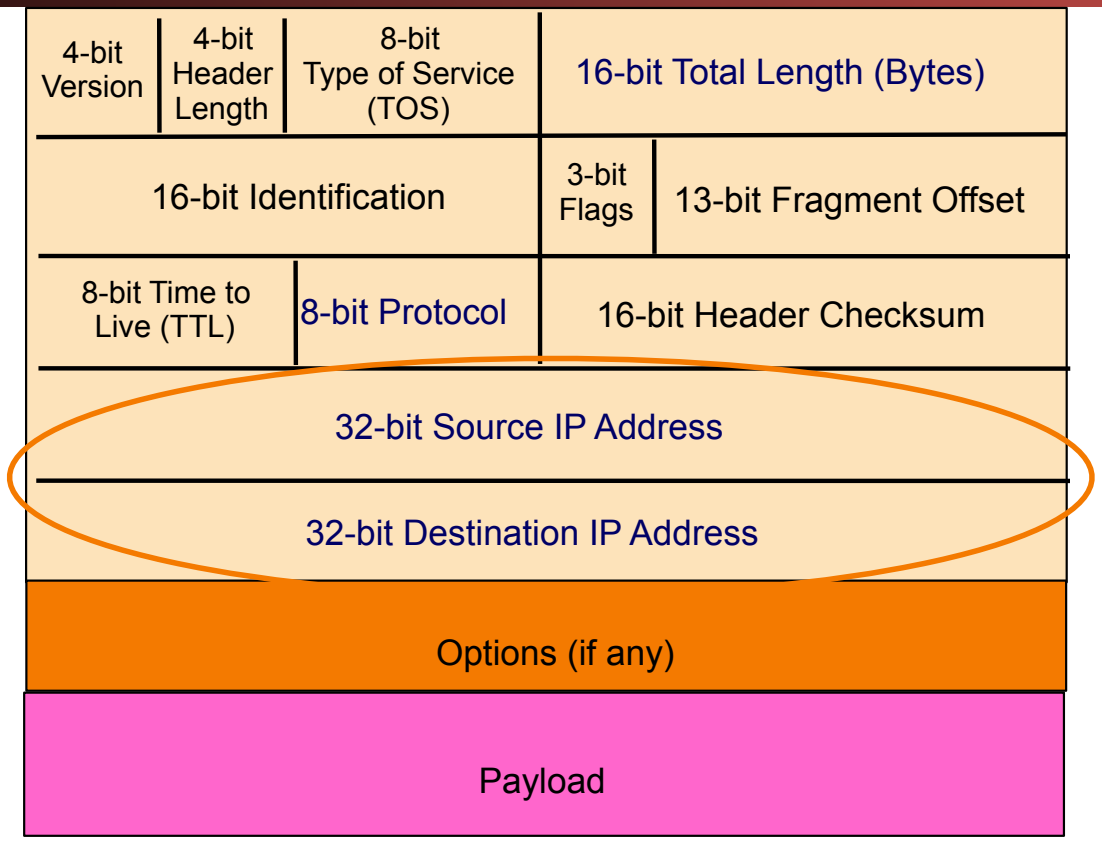


Specifies the length of the entire IP packet: bytes in this header plus bytes in the Payload

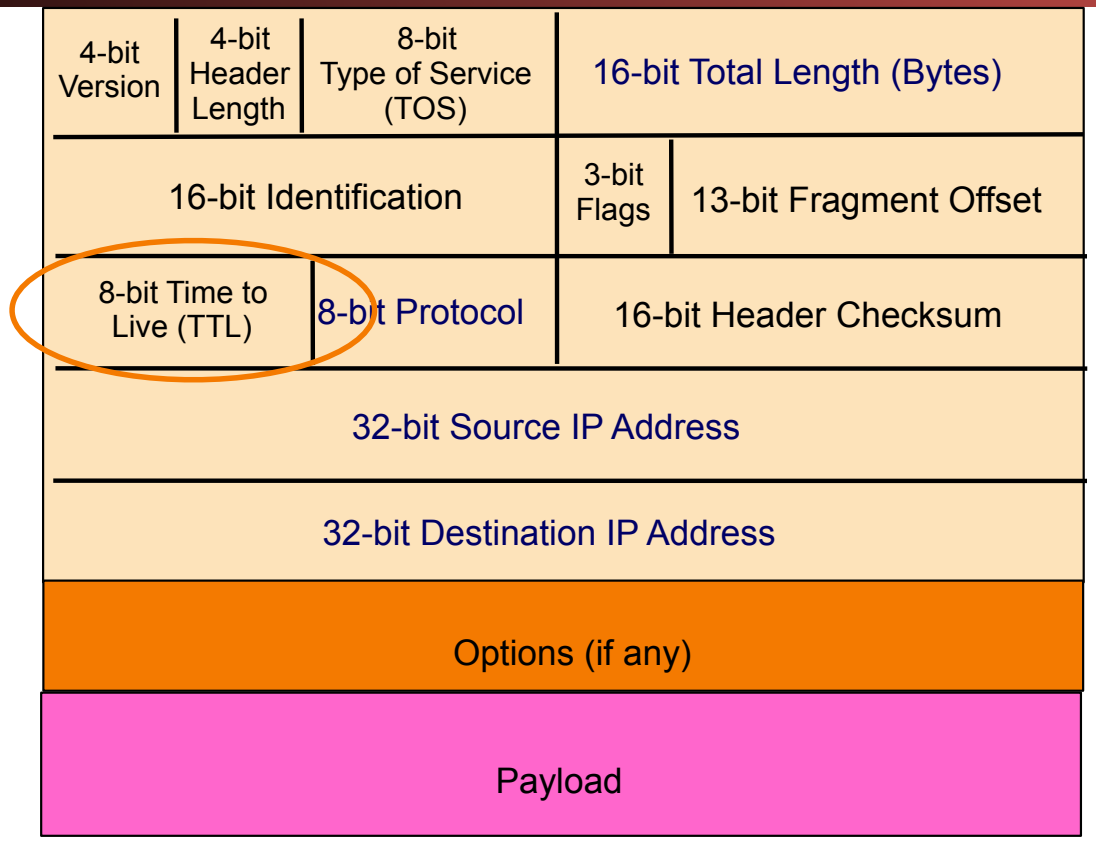
IP Packet Structure



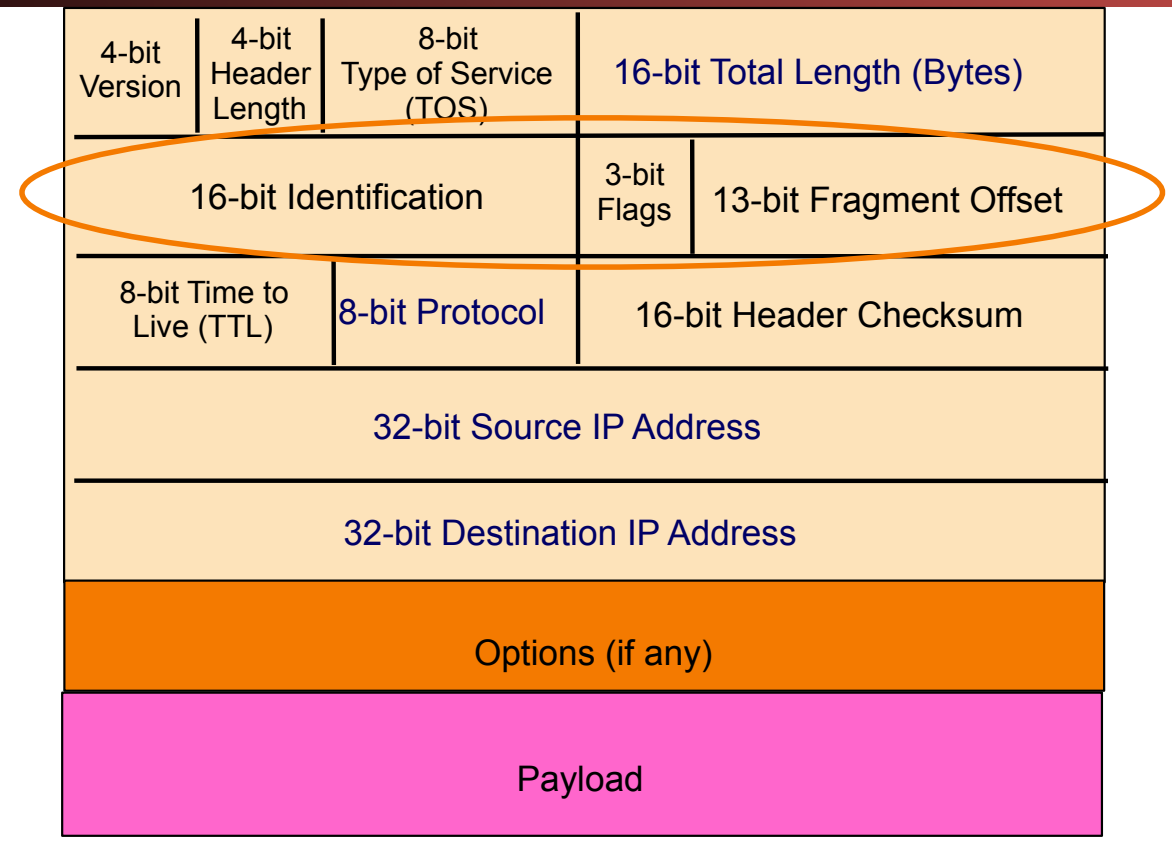
IP Packet Structure



IP Packet Structure



IP Packet Structure

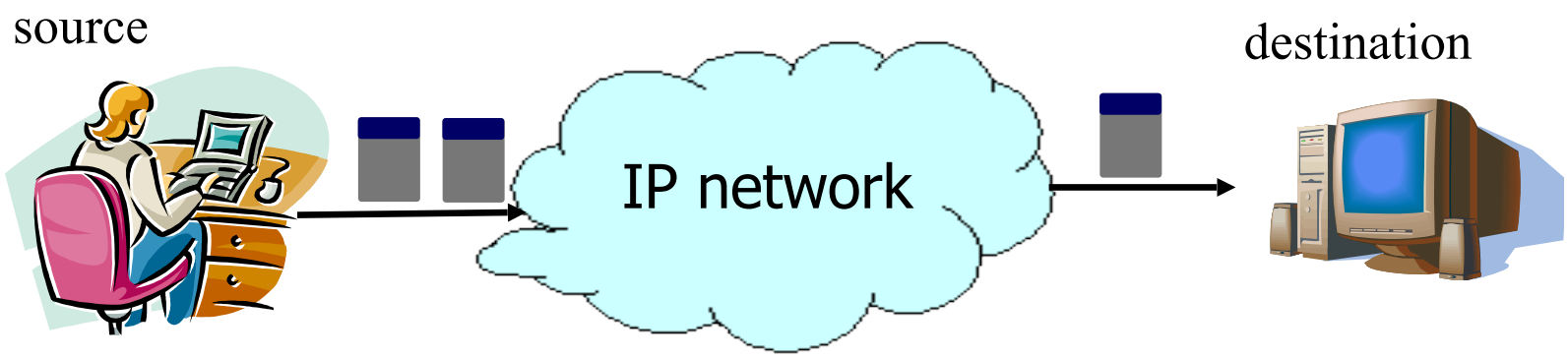


IP Packet Header (Continued)

- Two IP addresses
  - Source IP address (32 bits)
  - Destination IP address (32 bits)
- Destination address
  - Unique identifier/locator for the receiving host
  - Allows each node to make forwarding decisions
- Source address
  - Unique identifier/locator for the sending host
  - Recipient can decide whether to accept packet
  - Enables recipient to send a reply back to source

IP: “Best Effort ” Packet Delivery

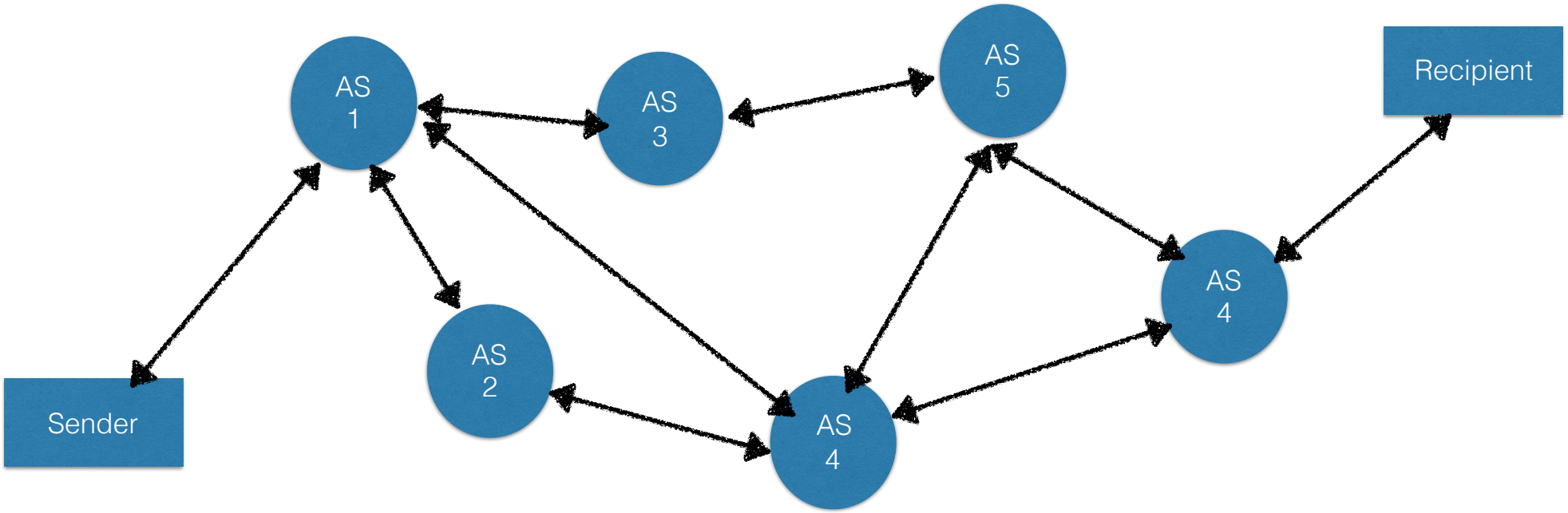
- Routers inspect destination address, locate “next hop” in forwarding table
  - Address = ~unique identifier/locator for the receiving host
- Only provides a “*I’ll give it a try*” delivery service:
  - Packets may be lost
  - Packets may be corrupted
  - Packets may be delivered out of order



IP Routing: Autonomous Systems

- Your system sends IP packets to the gateway...
  - But what happens after that?
- Within a given network its routed internally
- But the key is the Internet is a network-of-networks
  - Each "autonomous system" (AS) handles its own internal routing
  - The AS knows the next AS to forward a packet to
- Primary protocol for communicating in between ASs is BGP

Packet Routing on the Internet



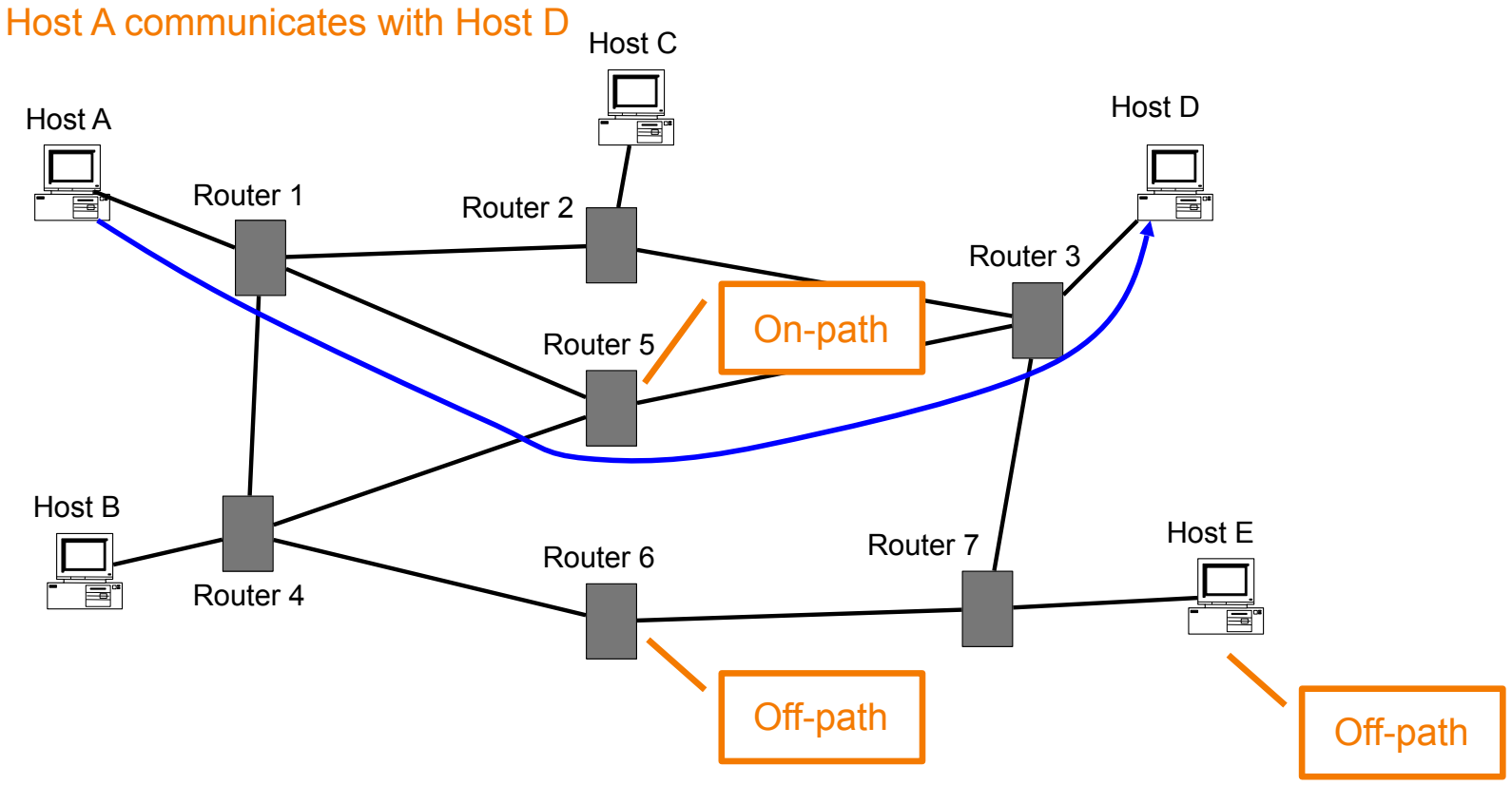
Remarks

- This is a network of networks
  - Its designed with failures in mind:
    Links can go down and the system will recover
  - But it also generally trust-based
    - A system can lie about what networks it can route to!
- Each hop decrements the TTL
  - Prevents a "routing loop" from happening
- Routing can be asymmetric
  - Since in practice networks may (slightly) override BGP, and

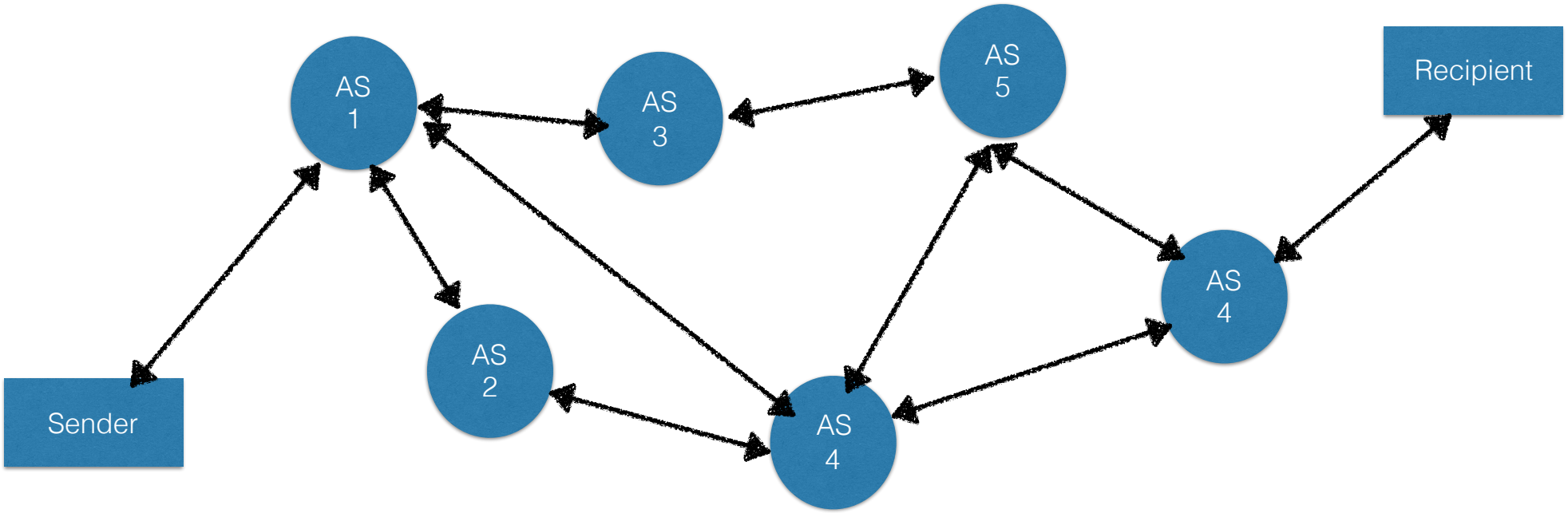
IP Spoofing
And Autonomous Systems

- The edge-AS where a user connects ***should*** restrict packet spoofing
  - Sending a packet with a different sender IP address
- But about 25% of them don't...
  - So a system can simply lie and say it comes from someplace else
- This enables blind-spoofing attacks
  - Such as the Kaminski attack on DNS
- It also enables "reflected DOS attacks"

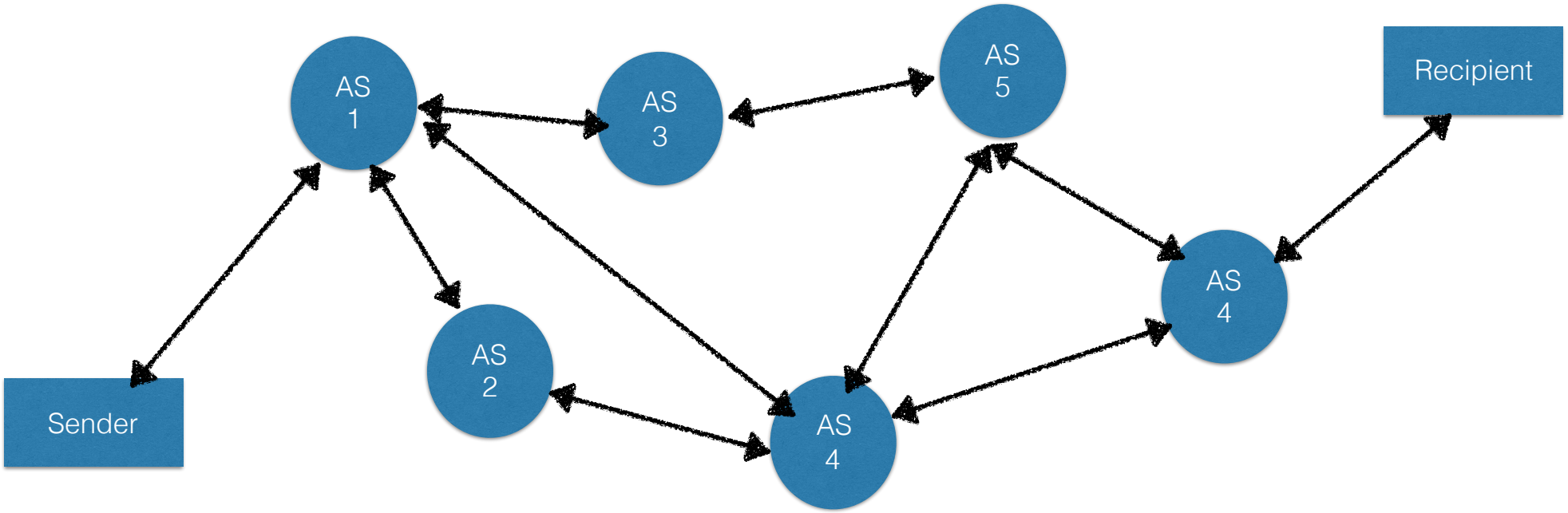
On-path Injection vs Off-path Spoofing



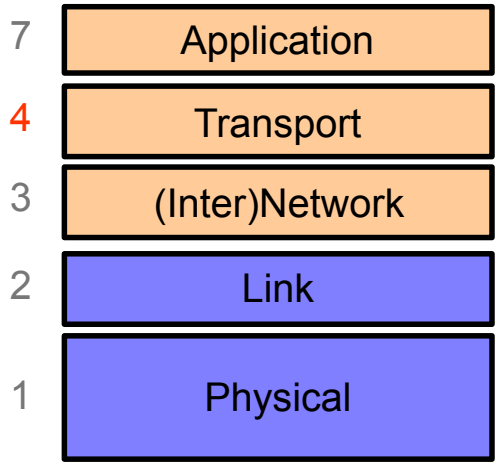
Lying in BGP

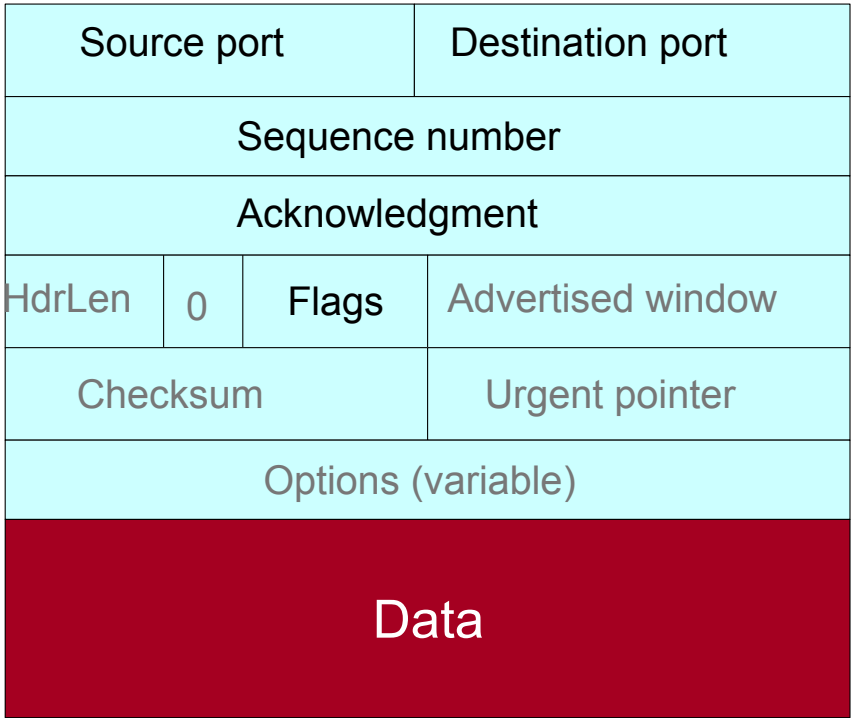


Lying in BGP



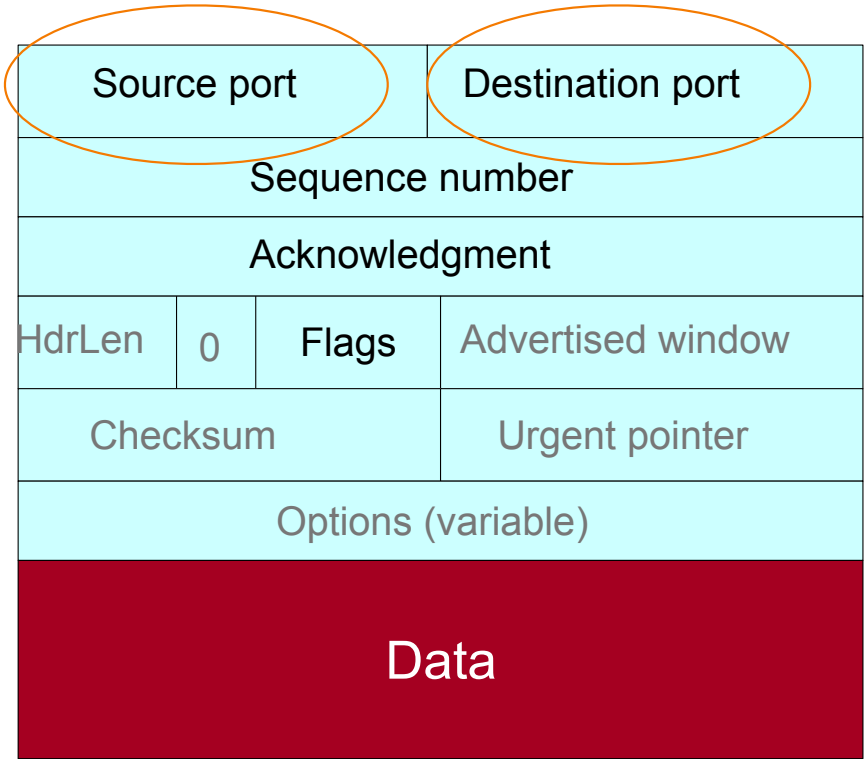
TCP


7
Application
4
Transport
3
(Inter)Network
2
Link
1
Physical


Source port
Destination port
Sequence number
Acknowledgment
HdrLen
0
Flags
Advertised window
Checksum
Urgent pointer
Options (variable)
Data

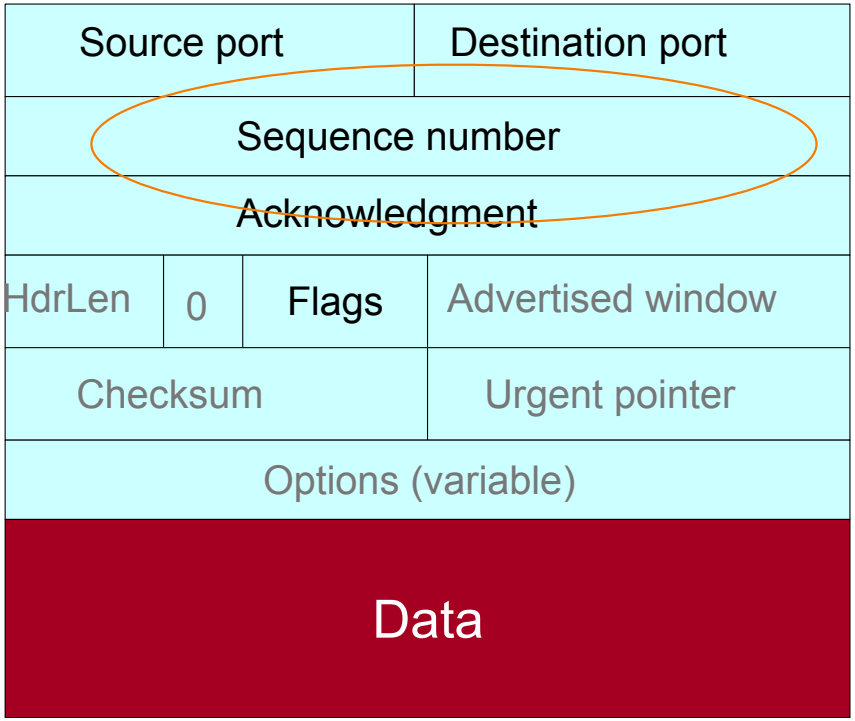
TCP

These plus IP addresses define a given connection



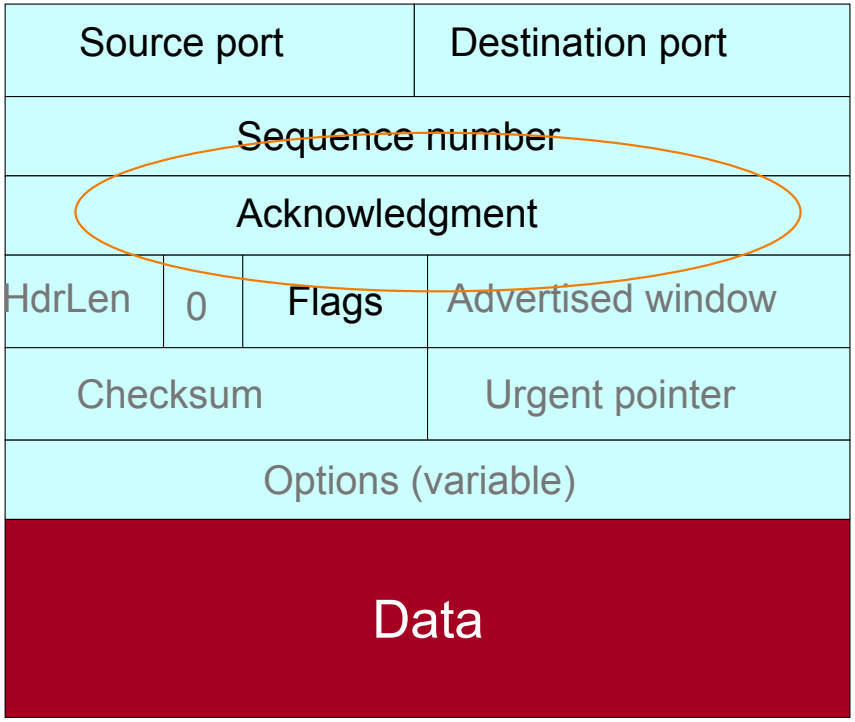
TCP

Used to order data in the connection: client program receives data ***in order***

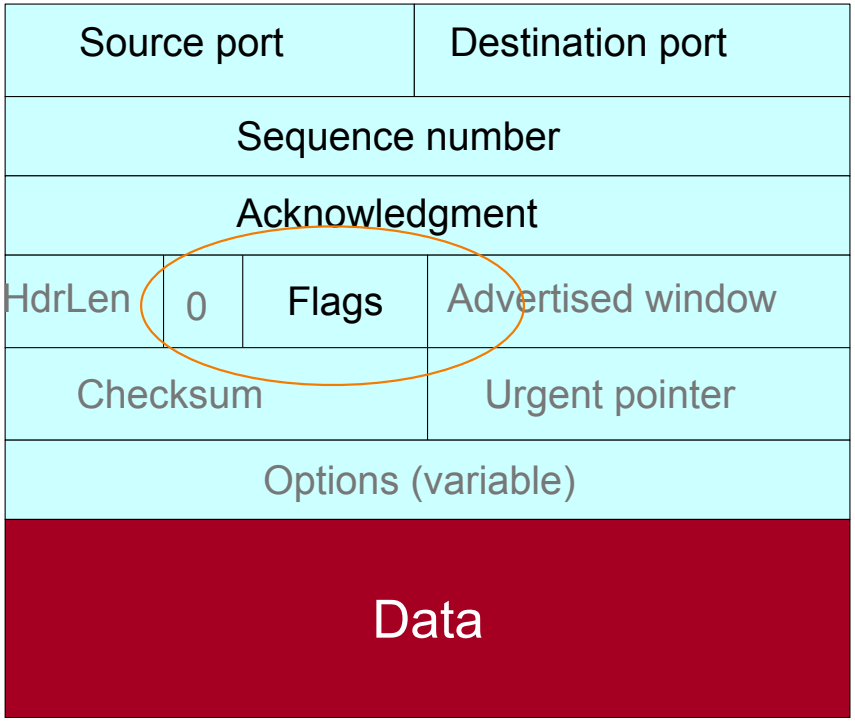


TCP

Used to say how much data has been received



TCP



Flags have different meaning:

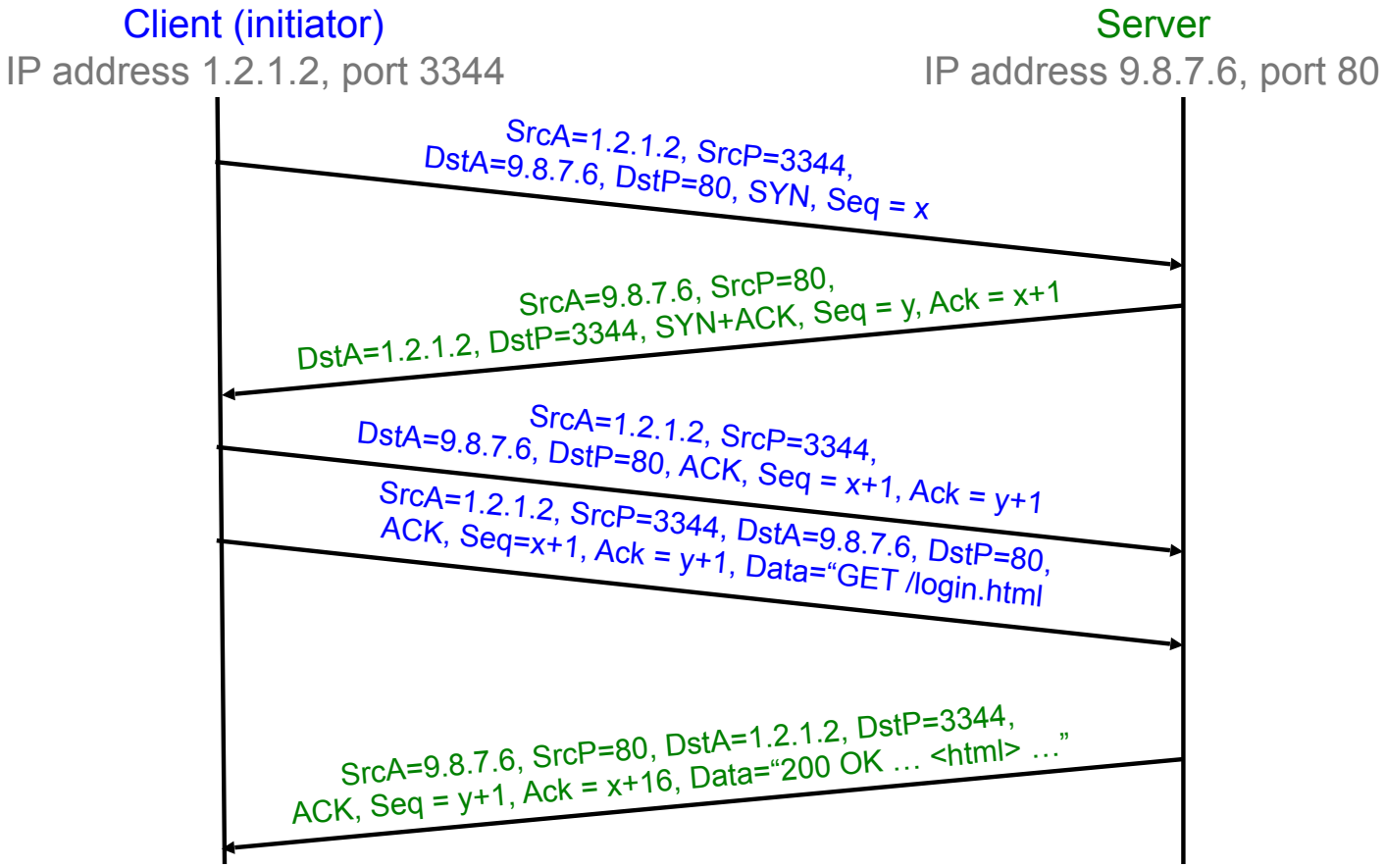
SYN: Synchronize,
used to initiate a connection

ACK: Acknowledge,
used to indicate
acknowledgement of data

FIN: Finish,
used to indicate no more data
will be sent (but can still receive
and acknowledge data)

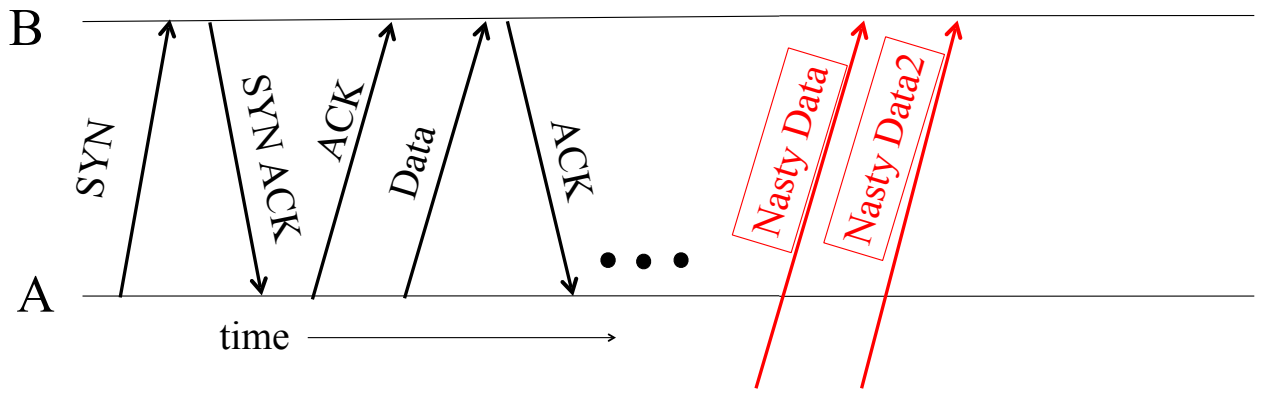
RST: Reset,
used to terminate the
connection completely

TCP Conn. Setup & Data Exchange

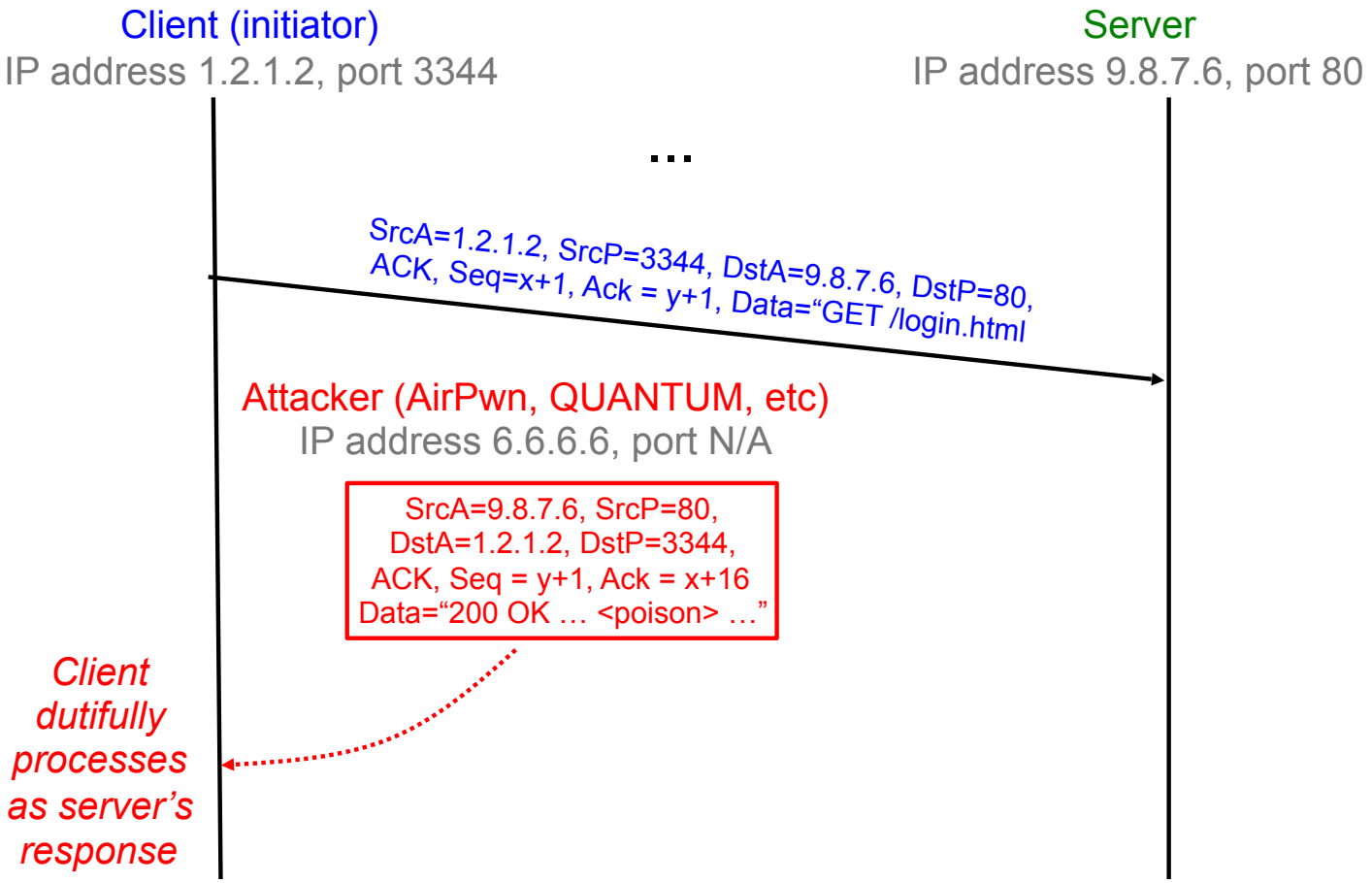


TCP Threat: Data Injection

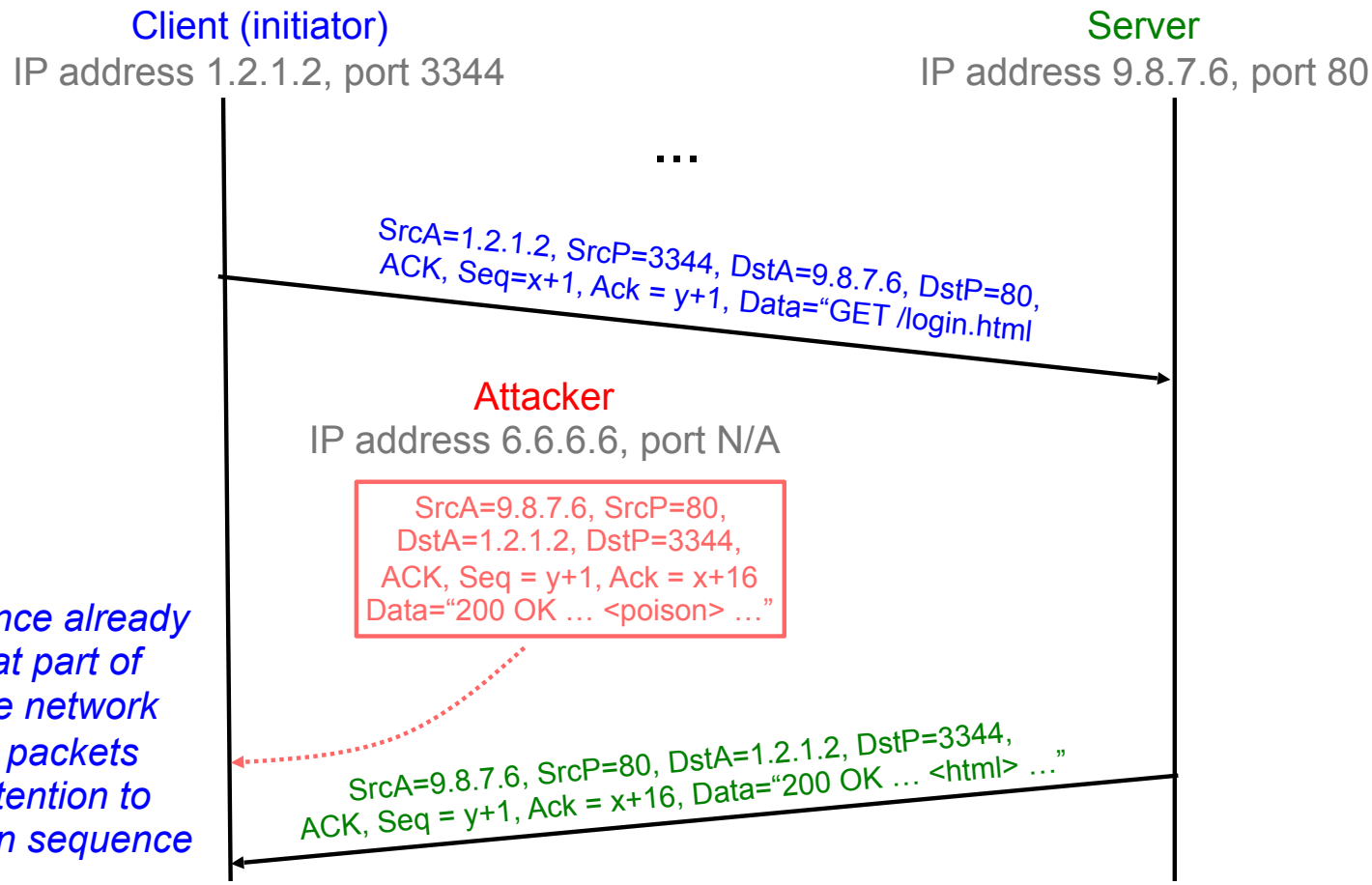
- If attacker knows ports & sequence numbers (e.g., on-path attacker), attacker can inject data into any TCP connection
  - Receiver B is *none the wiser!*
- Termed TCP connection hijacking (or "*session hijacking*")
  - A general means to take over an already-established connection!
- We are toast if an attacker can see our TCP traffic!
  - Because then they immediately know the port & sequence numbers



TCP Data Injection



TCP Data Injection



Client ignores since already processed that part of bytestream: the network can duplicate packets so only pay attention to the first version in sequence

TCP Threat: Disruption aka RST injection

- The attacker can also inject RST packets instead of payloads
  - TCP clients must respect RST packets and stop all communication
    - Because its a real world error recovery mechanism
    - So "just ignore RSTs don't work"
- Who uses this?
  - China: The Great Firewall does this to TCP requests
  - A long time ago: Comcast, to block BitTorrent uploads
  - Some intrusion detection systems: To hopefully mitigate an attack in progress

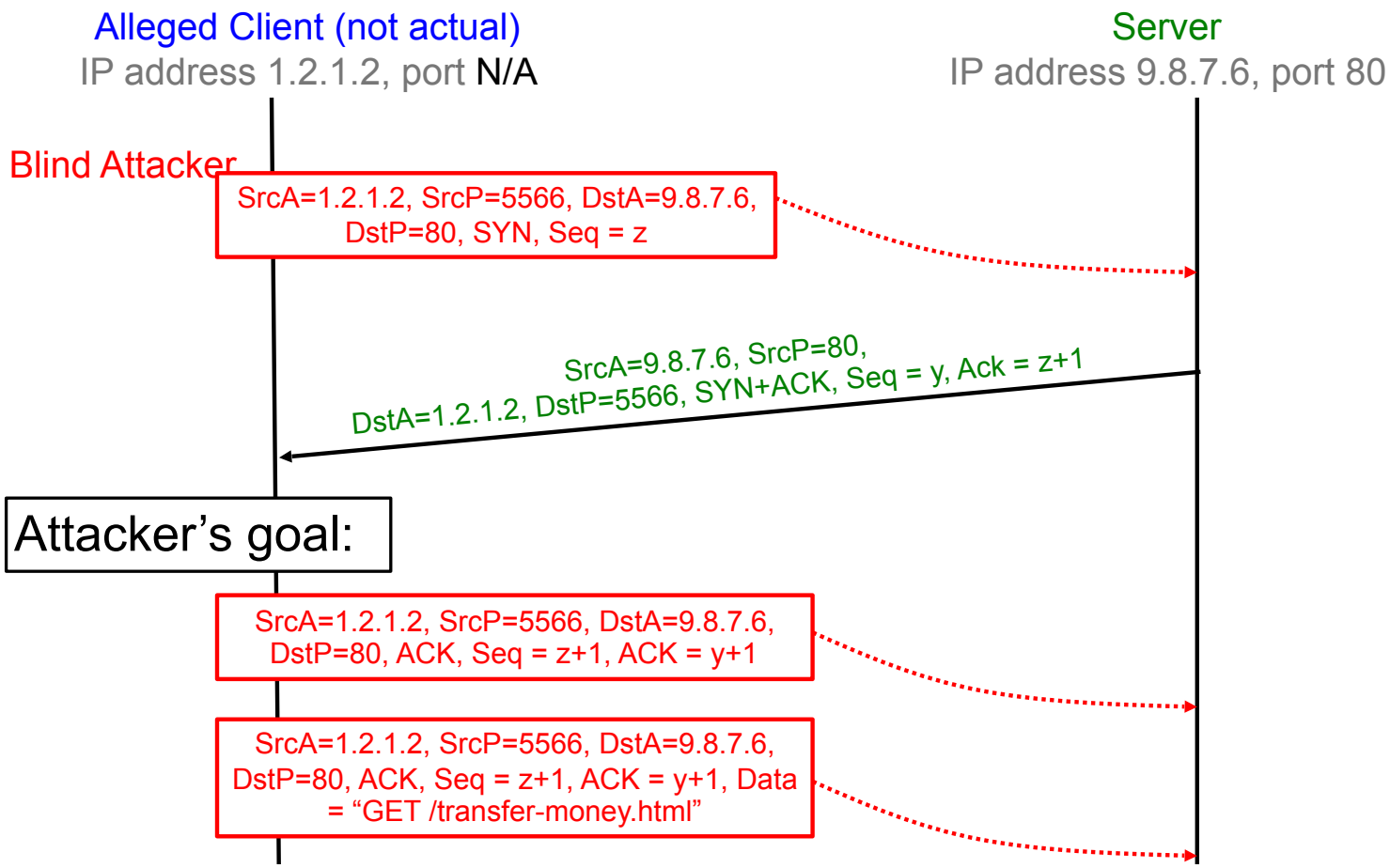
TCP Threat: Blind Hijacking

- Is it possible for an off-path attacker to inject into a TCP connection even if they can't see our traffic?
- YES: if somehow they can infer or guess the port and sequence numbers

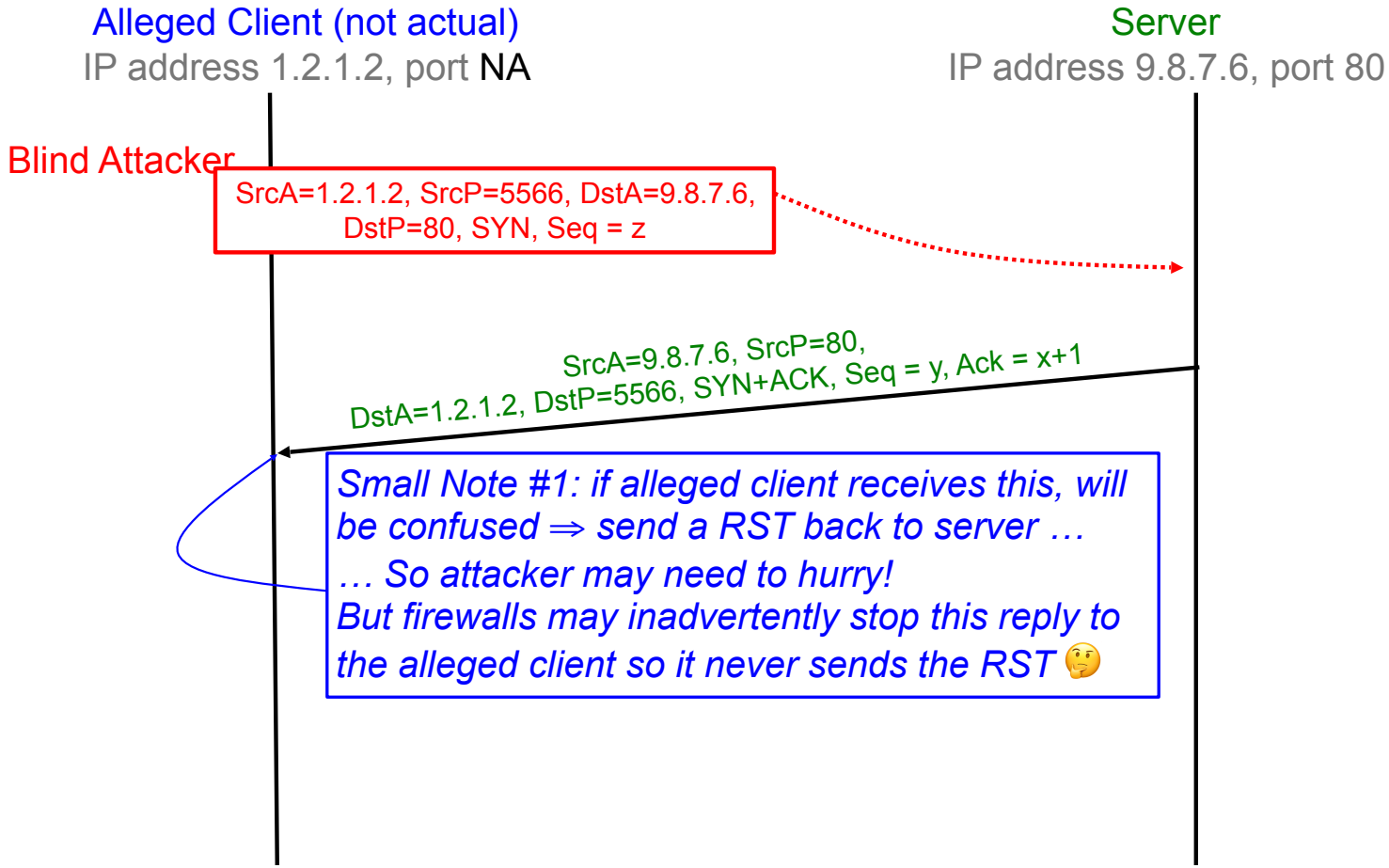
TCP Threat: Blind Spoofing

- Is it possible for an off-path attacker to create a fake TCP connection, even if they can't see responses?
- YES: if somehow they can infer or guess the TCP initial sequence numbers
- Why would an attacker want to do this?
  - Perhaps to leverage a server's trust of a given client as identified by its IP address
  - Perhaps to frame a given client so the attacker's actions during the connections can't be traced back to the attacker

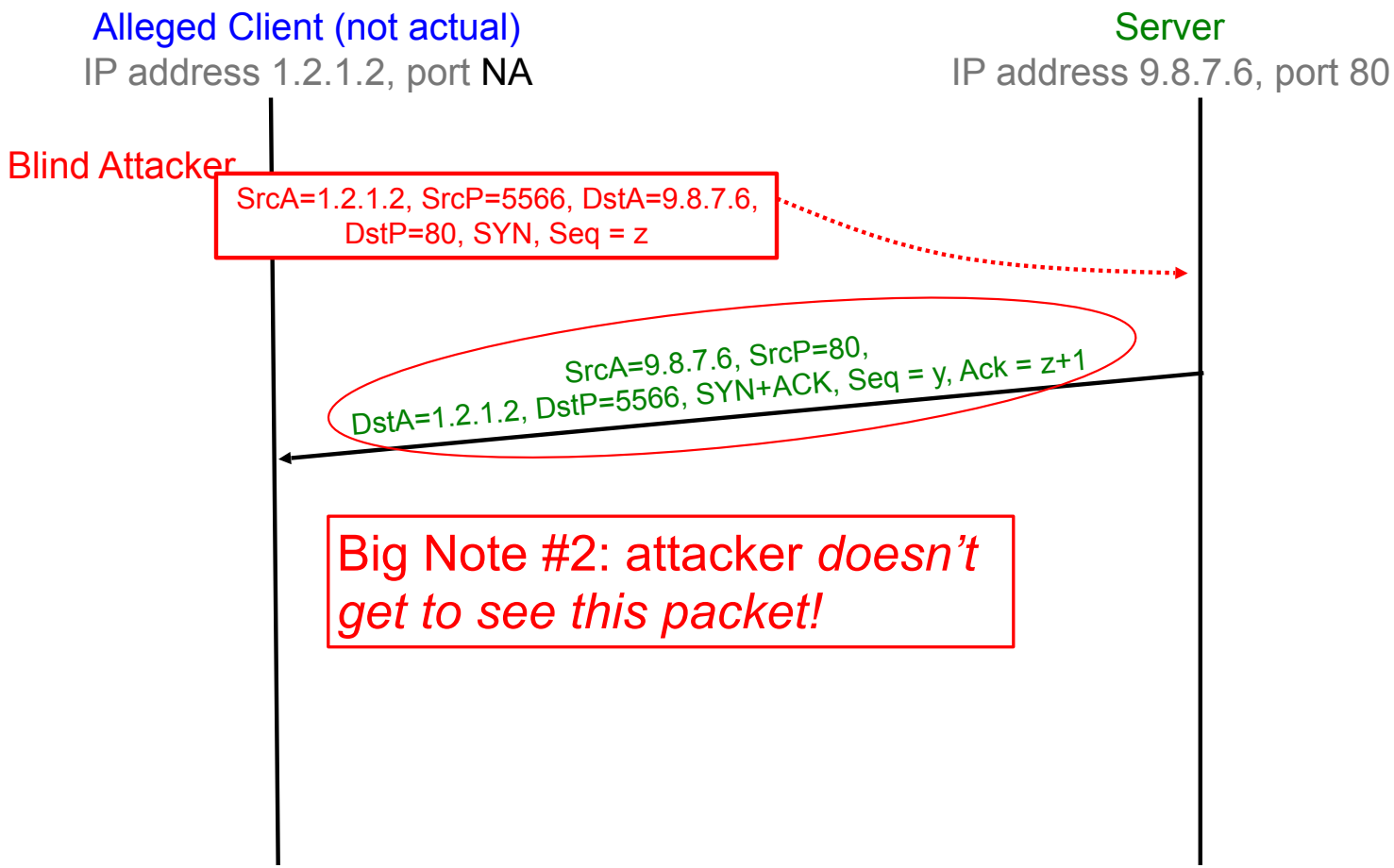
Blind Spoofing on TCP Handshake



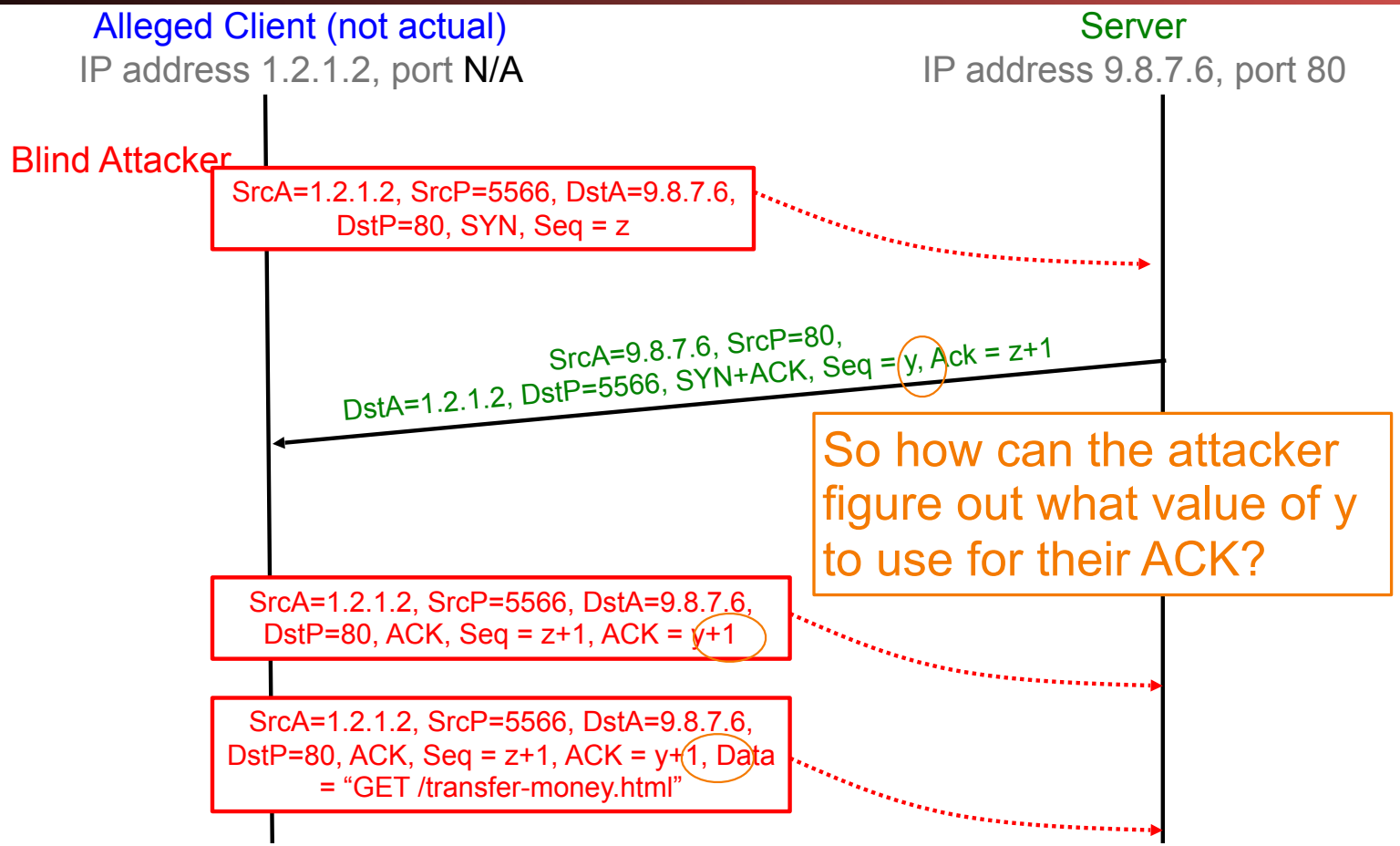
Blind Spoofing on TCP Handshake



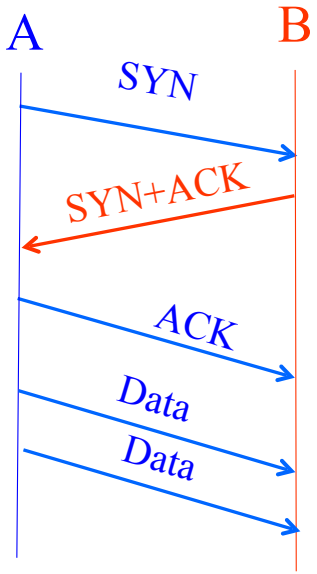
Blind Spoofing on TCP Handshake



Blind Spoofing on TCP Handshake



Reminder: Establishing a TCP Connection



Each host tells its *Initial Sequence Number* (ISN) to the other host.

(Spec says to pick based on local clock)

Hmm, any way for the attacker to know *this?*

How Do We Fix This?

Use a (Pseudo)-*Random* ISN

Sure – make a non-spoofed connection *first*, and see what server used for ISN y then!

Summary of TCP Security Issues

- An attacker who can observe your TCP connection can manipulate it:
  - Forcefully terminate by forging a RST packet
  - Inject (spoof) data into either direction by forging data packets
  - Works because they can include in their spoofed traffic the correct sequence numbers (both directions) and TCP ports
  - Remains a major threat today

Summary of TCP Security Issues

- An attacker who can observe your TCP connection can manipulate it:
  - Forcefully terminate by forging a RST packet
  - Inject (spoof) data into either direction by forging data packets
  - Works because they can include in their spoofed traffic the correct sequence numbers (both directions) and TCP ports
  - Remains a major threat today
- If attacker could predict the ISN chosen by a server, could "blind spoof" a connection to the server
  - Makes it appear that host ABC has connected, and has sent data of the attacker's choosing, when in fact it hasn't
  - Undermines any security based on trusting ABC's IP address
  - Allows attacker to "frame" ABC or otherwise avoid detection
  - Fixed (mostly) today by choosing random ISNs

But wasn't fixed completely...

- CVE-2016-5696
  - "Off-Path TCP Exploits: Global Rate Limit Considered Dangerous" Usenix Security 2016
  - https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/cao
- Key idea:
  - RFC 5961 added some global rate limits that acted as an ***information leak***:
    - Could determine if two clients were communicating on a given port
    - Could determine if you could correctly guess the sequence #s for this communication
      - Required a third host to probe this and at the same time spoof packets
  - Once you get the sequence #s, you can then inject arbitrary content into the TCP stream (d'oh)

The Bane of the Internet: The (distributed) Denial of Service Attack

- Lets say you've run afoul of a bad guy...
  - And he don't like your web page
  - He hires some other bad guy to launch a "Denial of Service" attack
- This other bad guys controls a lot of machines on the Internet
  - These days a million systems is not unheard of
- The bad guy just instructs those machines to make a ***lot*** of requests to your server...
  - Blowing it off the network with traffic

And the Firewall...

- Attackers can't attack what they can't talk to!
  - If you don't accept ***any*** communication from an attacker, you can't be exploited
- The firewall is a network device (or software filter on the end host) that restricts communication
  - Primarily just by IP/Port or network/Port
- Default deny:
  - By default, disallow any contact to this host on any port
- Default allow:
  - By default, allow any contact to this host on any port
- More when we discuss Intrusion Detection next week