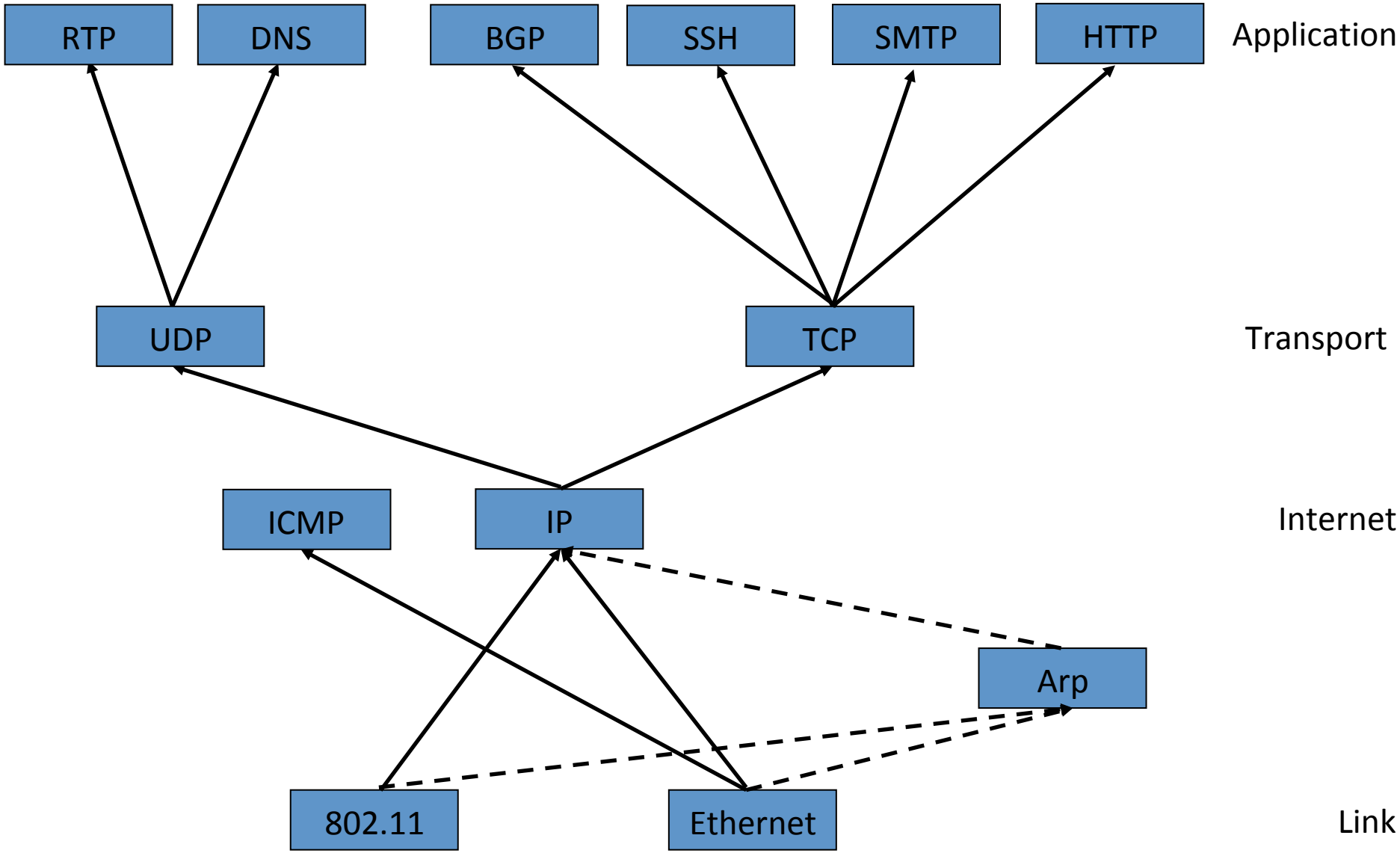
Lecture 15 – Networking Attacks

Michael Bailey
University of Illinois
ECE 422/CS 461 – Spring 2018


RTP
DNS
BGP
SSH
SMTP
HTTP
Application
UDP
TCP
Transport
ICMP
IP
Internet
Arp
802.11
Ethernet
Link

Hacking the network

- Attacks at all layers of the network
  - Data-link layer
  - Network layer
  - Transport layer
  - Application layer
- Attacks against all of the properties we care about
  - Availability
  - Integrity
  - Confidentiality

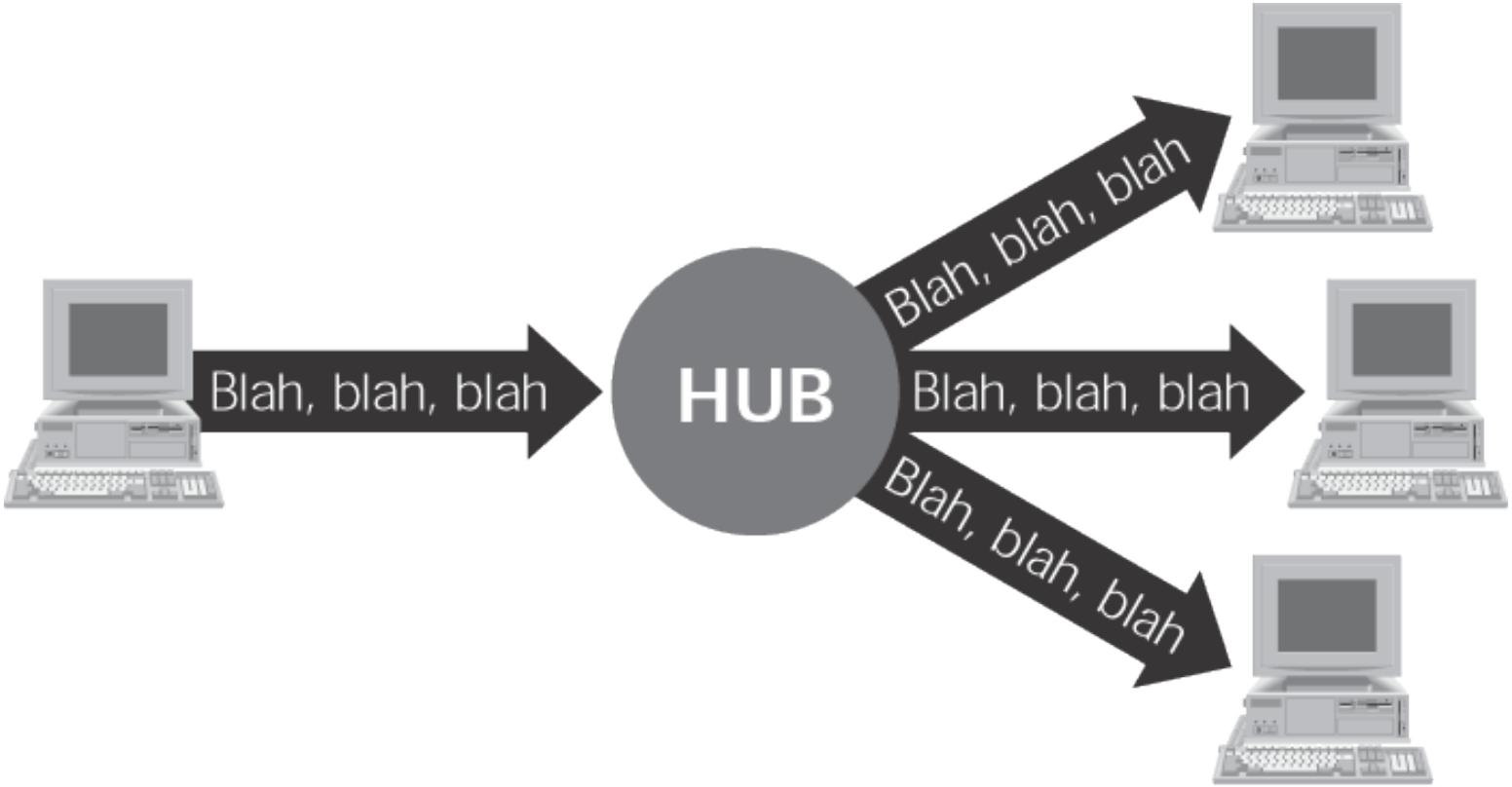
Why?

- Unencrypted transmission
  - Eavesdropping possible at any intermediate host during forwarding
- No source authentication
  - Sender can spoof source address, making it difficult to trace packet back to attacker
- No integrity checking
  - Entire packet, header and payload, can be modified while en route to destination, enabling content forgeries, redirections, and man-in-the-middle attacks
- No bandwidth constraints
  - Large number of packets can be injected into network to launch a denial-of-service attack

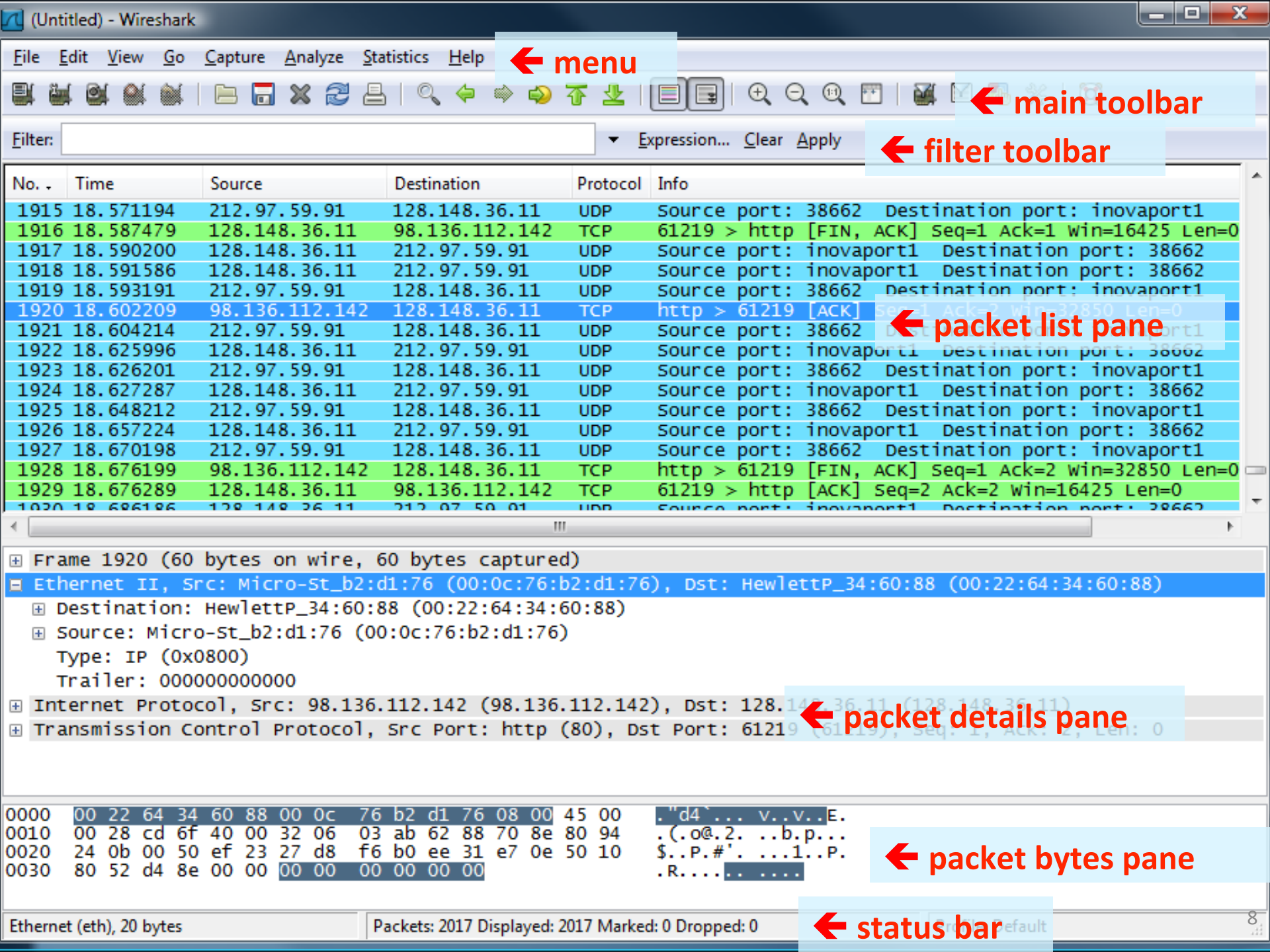
Link Layer Attacks

Sniffing and attacks against confidentiality

- Gathering packets from the local network
  - Passive (wired network with a hub or a wireless network)
  - Active (wired network built with a switch)
- Traffic is not encrypted

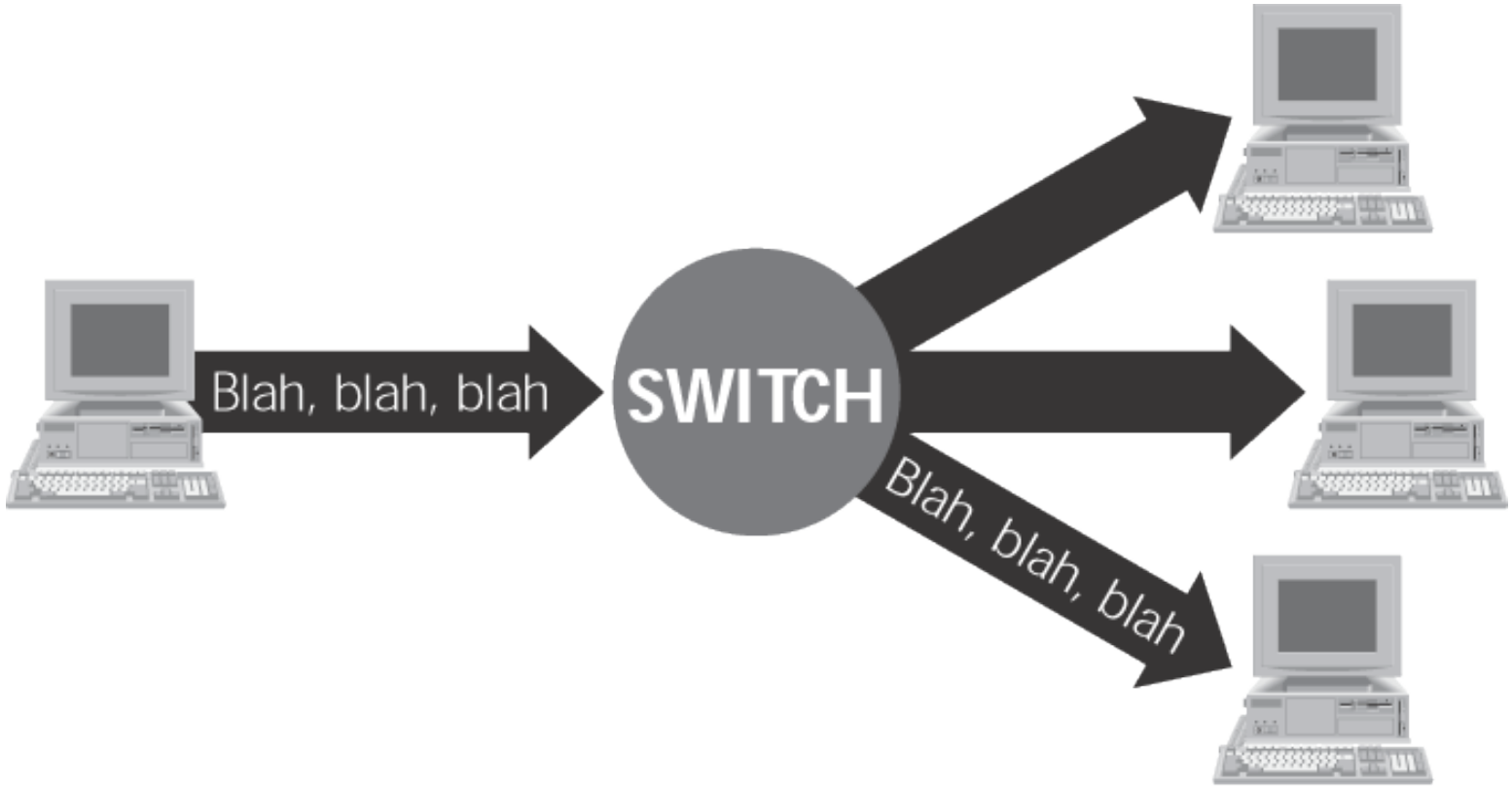


BROADCAST ETHERNET


(Untitled) - Wireshark
File Edit View Go Capture Analyze Statistics Help
menu
main toolbar
Filter:
Expression... Clear Apply
filter toolbar
No. Time Source Destination Protocol Info
1915 18.571194 212.97.59.91 128.148.36.11 UDP Source port: 38662 Destination port: inovaport1
1916 18.587479 128.148.36.11 98.136.112.142 TCP 61219 > http [FIN, ACK] Seq=1 Ack=1 Win=16425 Len=0
1917 18.590200 128.148.36.11 212.97.59.91 UDP Source port: inovaport1 Destination port: 38662
1918 18.591586 128.148.36.11 212.97.59.91 UDP Source port: inovaport1 Destination port: 38662
1919 18.593191 212.97.59.91 128.148.36.11 UDP Source port: 38662
1920 18.602209 98.136.112.142 128.148.36.11 TCP http > 61219 [ACK]
1921 18.604214 212.97.59.91 128.148.36.11 UDP Source port: 38662
1922 18.625996 128.148.36.11 212.97.59.91 UDP Source port: inovaport1
1923 18.626201 212.97.59.91 128.148.36.11 UDP Source port: 38662 Destination port: inovaport1
1924 18.627287 128.148.36.11 212.97.59.91 UDP Source port: inovaport1 Destination port: 38662
1925 18.648212 212.97.59.91 128.148.36.11 UDP Source port: 38662 Destination port: inovaport1
1926 18.657224 128.148.36.11 212.97.59.91 UDP Source port: inovaport1 Destination port: 38662
1927 18.670198 212.97.59.91 128.148.36.11 UDP Source port: 38662 Destination port: inovaport1
1928 18.676199 98.136.112.142 128.148.36.11 TCP http > 61219 [FIN, ACK] Seq=1 Ack=2 Win=32850 Len=0
1929 18.676289 128.148.36.11 98.136.112.142 TCP 61219 > http [ACK] Seq=2 Ack=2 Win=16425 Len=0
packet list pane
Frame 1920 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: Micro-St_b2:d1:76 (00:0c:76:b2:d1:76), Dst: HewlettP_34:60:88 (00:22:64:34:60:88)
Destination: HewlettP_34:60:88 (00:22:64:34:60:88)
Source: Micro-St_b2:d1:76 (00:0c:76:b2:d1:76)
Type: IP (0x0800)
Trailer: 000000000000
Internet Protocol, Src: 98.136.112.142 (98.136.112.142), Dst: 128.
Transmission Control Protocol, Src Port: http (80), Dst Port: 6121
packet details pane
0000 00 22 64 34 60 88 00 0c 76 b2 d1 76 08 00 45 00 ."d4`... v..v..E.
0010 00 28 cd 6f 40 00 32 06 03 ab 62 88 70 8e 80 94 .(.o@.2. ..b.p...
0020 24 0b 00 50 ef 23 27 d8 f6 b0 ee 31 e7 0e 50 10 $..P.#'. ...1..P.
0030 80 52 d4 8e 00 00 00 00 00 00 00 00 .R...... ....
packet bytes pane
Ethernet (eth), 20 bytes
Packets: 2017 Displayed: 2017 Marked: 0 Dropped: 0
status bar

Google?


Google maps
Street View
google.com/streetview
STOP
HERE ON
RED



SWITCHED ETHERNET

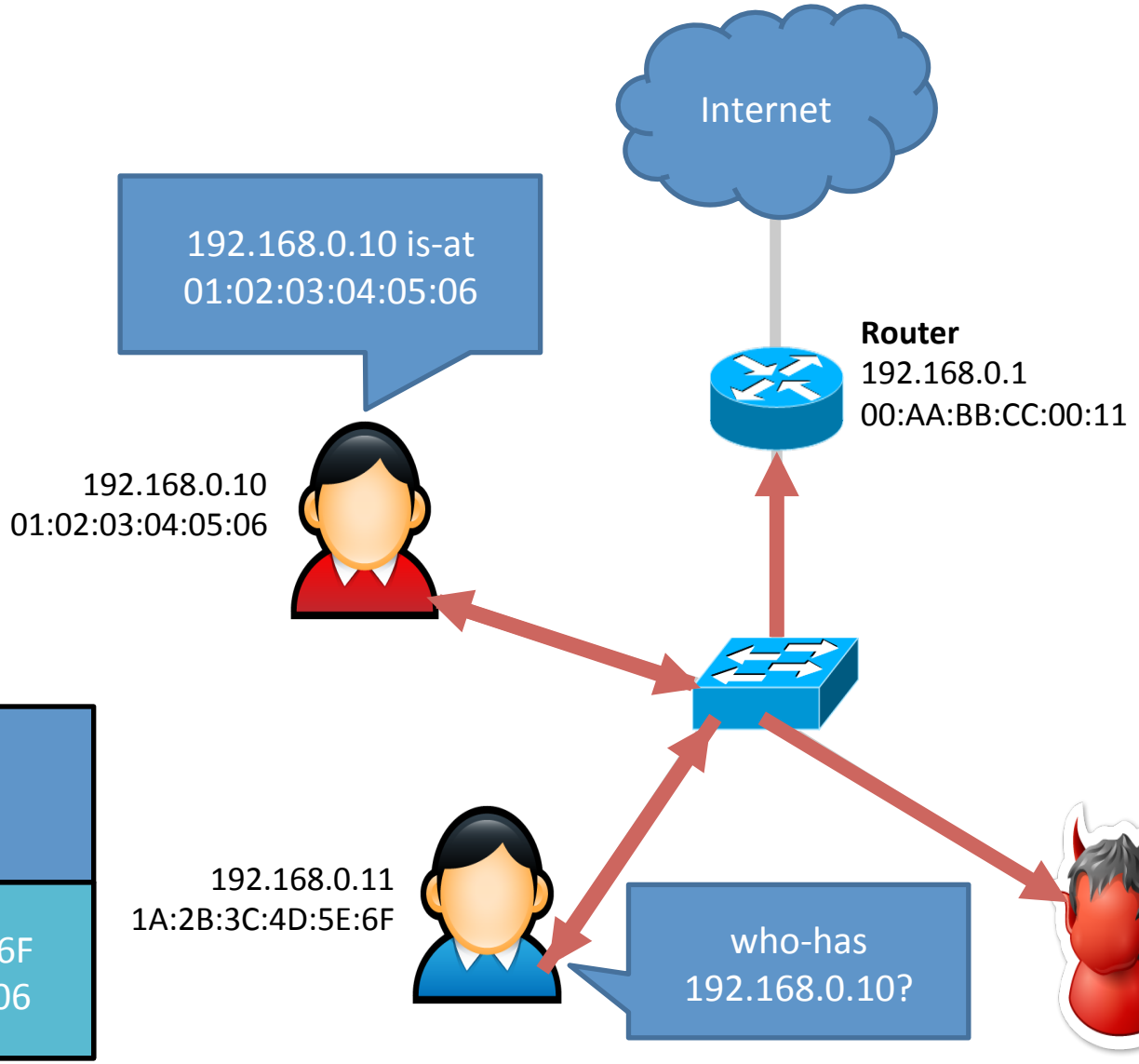
Active Sniffing

- Fool the switch into sending the packets to the sniffer
  - MAC Flooding
    - Send a flood of traffic with random MAC addresses
    - Fill up the switch's memory
    - Switches will then forward packets to all links on the switch
      - Dsniff program Macof
  - ARP spoofing
    - Send fake ARP replies to change the victim's ARP table
      - Dsniff program Arpspoof
    - Attacker configures his or her system to forward any traffic it receives to the router.
    - Any traffic from the target machine is sent to the attacker's machine before being transferred to the local network.

Address Resolution Protocol

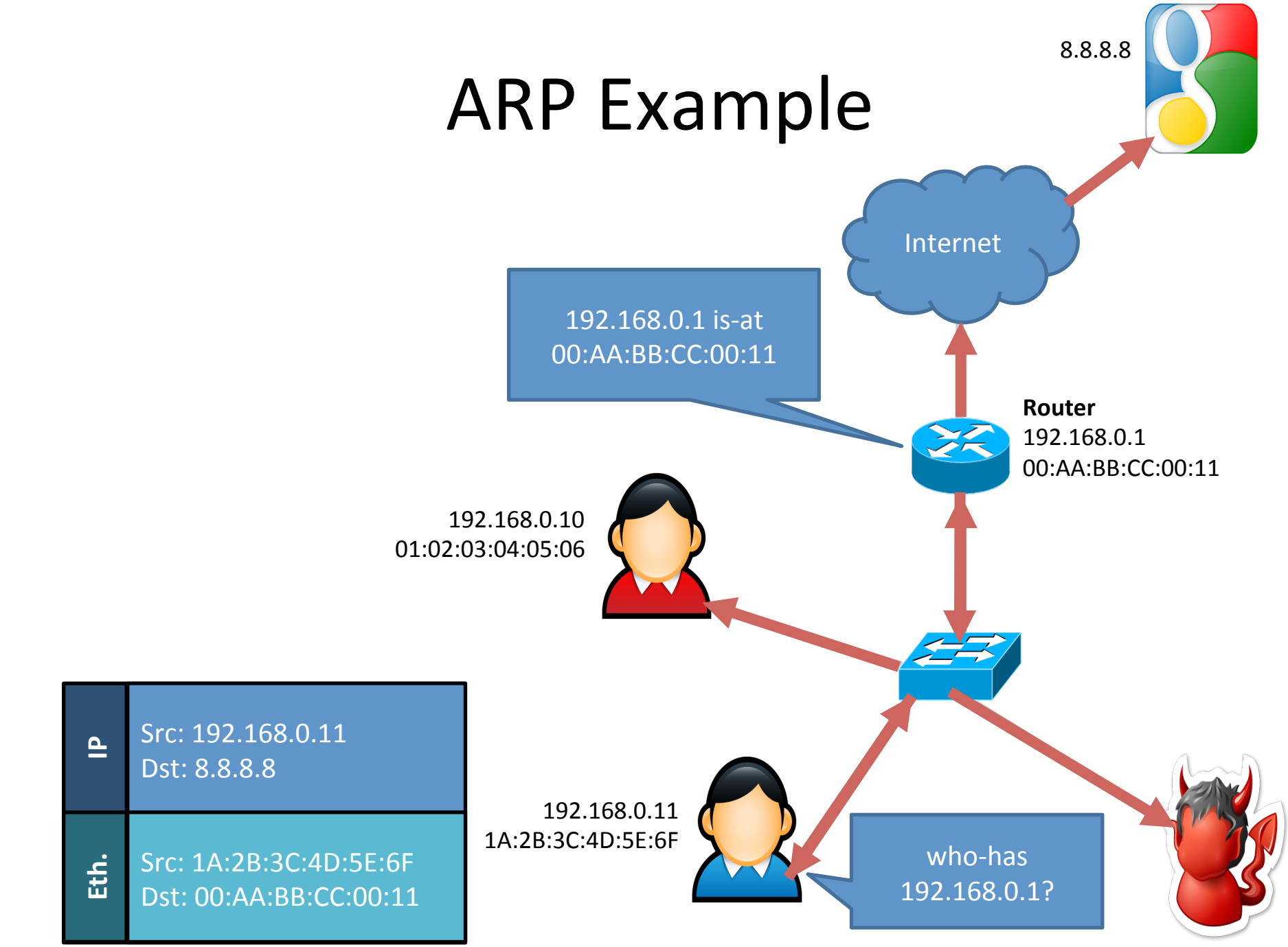
- A mapping is needed between IP and physical (MAC) address
- Dynamically established using the Address Resolution Protocol (ARP)
  - Broadcast protocol implemented at the link layer
  - Considered to be a layer 2.5 protocol
  - Used by Ethernet, 802.11, many other link layer protocols with IPv4
- Message types:
  - ***who-has*** requests – "Who has IP 192.168.0.2?"
  - ***is-at*** replies – "00:01:2D:00:5F:CC has IP 192.168.0.2"
- Problem:
  - No binding between ARP messages and sender identity
  - In other words, **no authentication**

ARP Example



IP	Src: 192.168.0.11 Dst: 192.168.0.10
Eth.	Src: 1A:2B:3C:4D:5E:6F Dst: 01:02:03:04:05:06

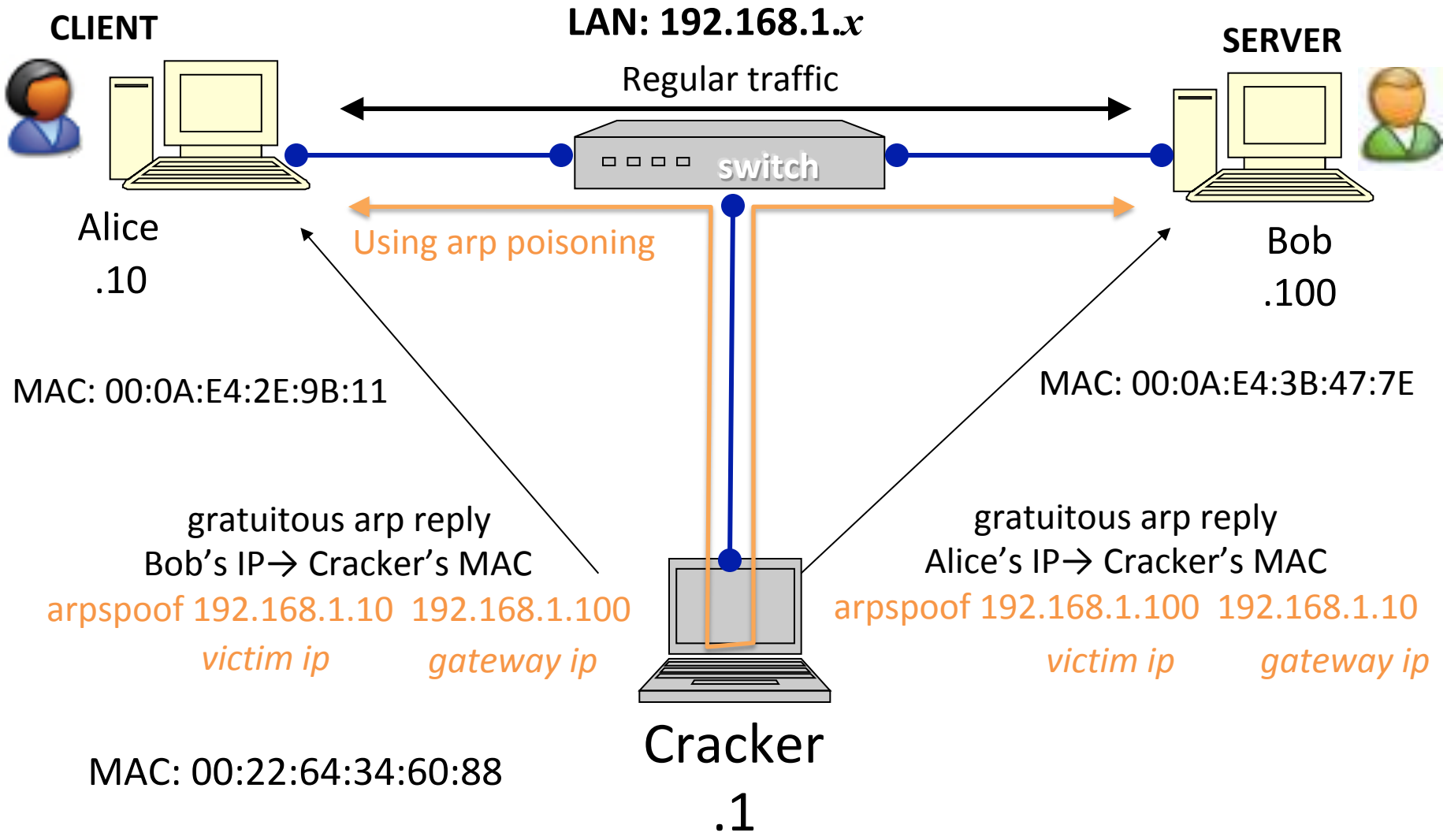
ARP Example


8.8.8.8
Internet
192.168.0.1 is-at
00:AA:BB:CC:00:11
Router
192.168.0.1
00:AA:BB:CC:00:11
192.168.0.10
01:02:03:04:05:06
192.168.0.11
1A:2B:3C:4D:5E:6F
who-has
192.168.0.1?
IP
Src: 192.168.0.11
Dst: 8.8.8.8
Eth.
Src: 1A:2B:3C:4D:5E:6F
Dst: 00:AA:BB:CC:00:11

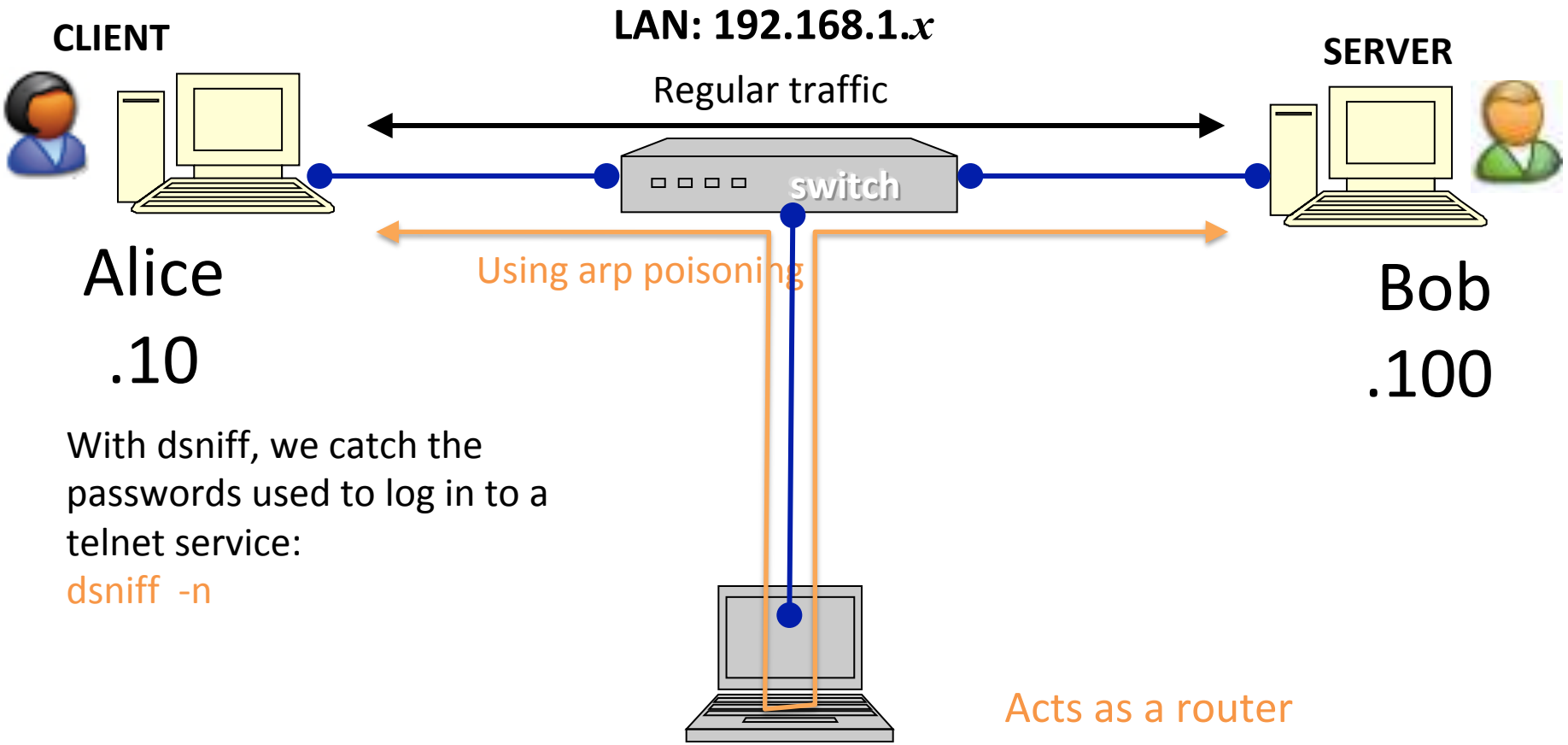
MAC modification/Spoofing

- Finding arp mapping on your network
  - Arp –a
- Change the MAC on a Host (Cloning)
  - Linux (ifconfig)
  - Windows Network Settings
- Creating Link Layer Packets (Spoofing)
  - libnet (API)
  - Linkcat (tool – netcat for link layer)

ARP Spoofing



Catch telnet password



With dsniff, we catch the passwords used to log in to a telnet service:

dsniff -n

Network layer hacks

IP Spoofing

- IP Spoofing is an attempt by an intruder to send packets from one IP address that appear to originate at another
- If the server thinks it is receiving messages from the real source after authenticating a session, it could inadvertently behave maliciously
- There are two basic forms of IP Spoofing
    - Blind Spoofing
        - Attack from any source
    - Non-Blind Spoofing
        - Attack from the same subnet

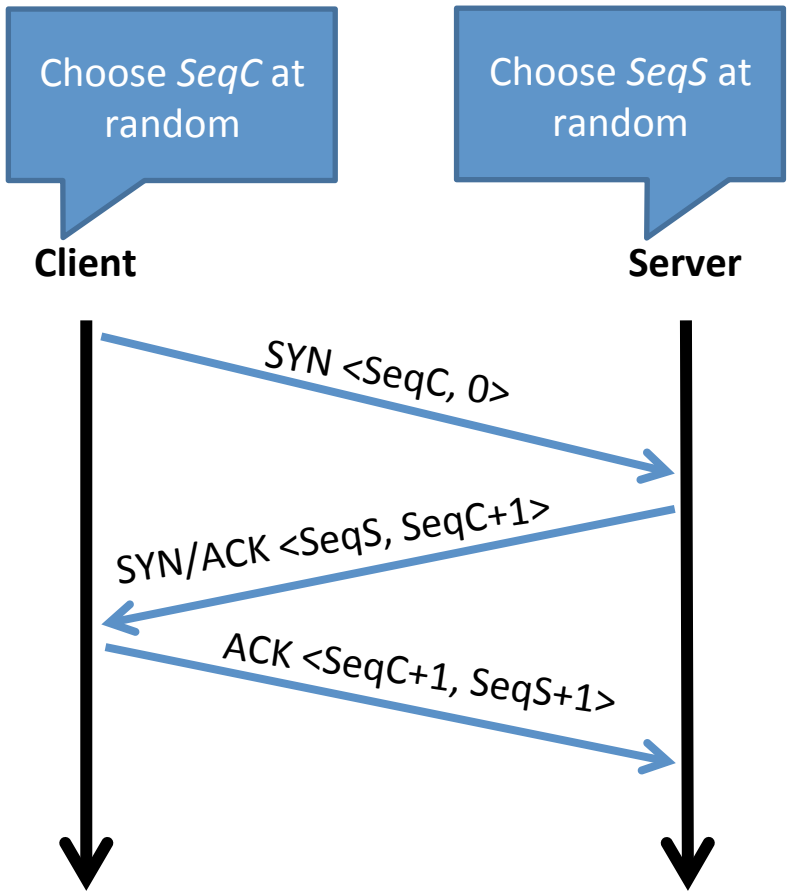
IP Spoofing

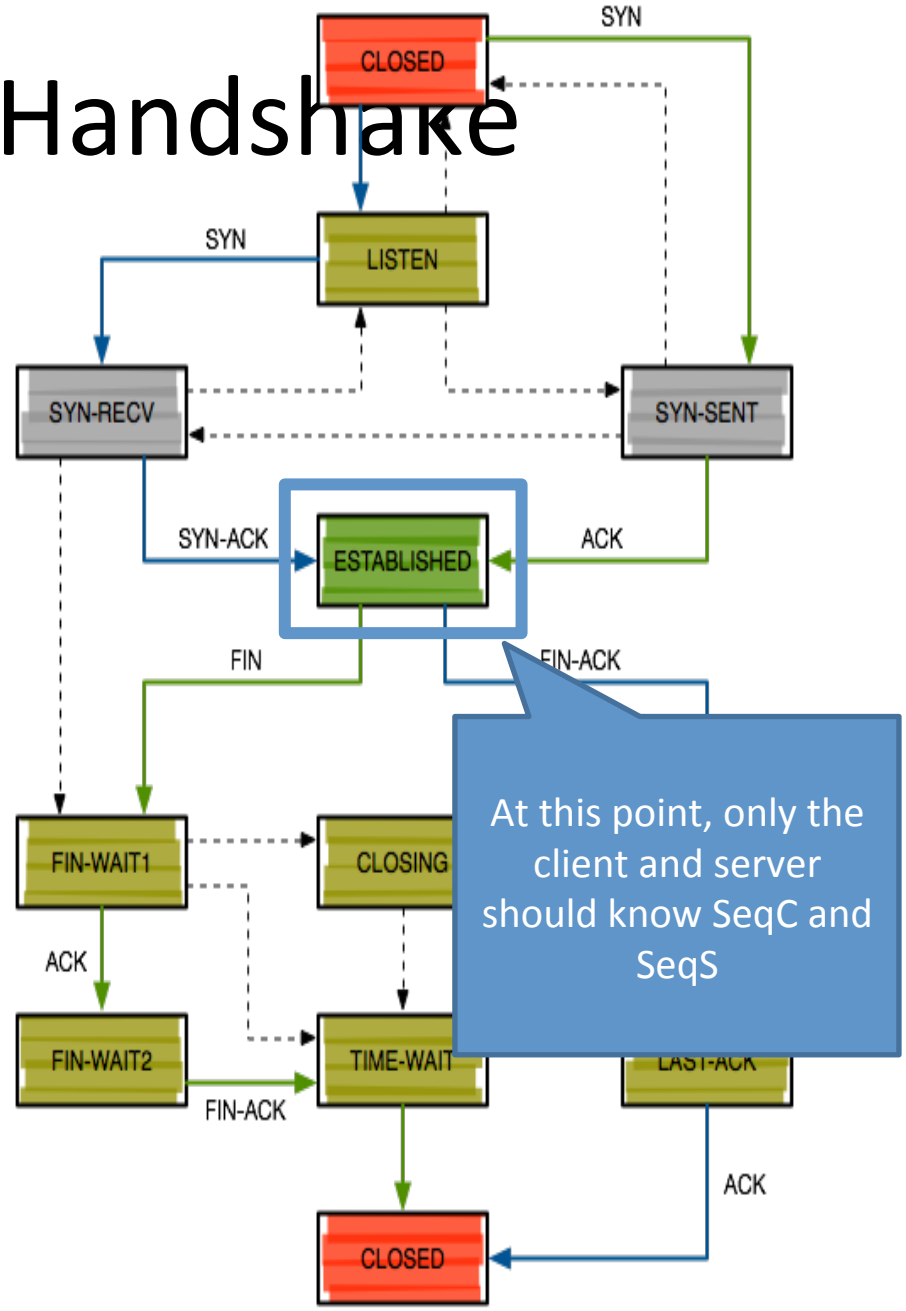
- Attacks made possible by IP spoofing include
  - Denial of Service (DOS)
  - Session Hijacking
  - Man in the Middle
- To take over a TCP stream, sequence and acknowledgement numbers must be sniffed or predicted.

TCP Connection Setup and Flags

- Why do we need connection setup?
  - To establish state on both hosts
  - Most important state: sequence numbers
    - Count the number of bytes that have been sent
    - Initial value chosen at random
  - Random initial sequence numbers are a form of authentication
- Important TCP flags (1 bit each)
  - SYN – synchronization, used for connection setup
  - ACK – acknowledge received data
  - FIN – finish, used to tear down connection
  - RST – unrecoverable error, immediately terminate the connection

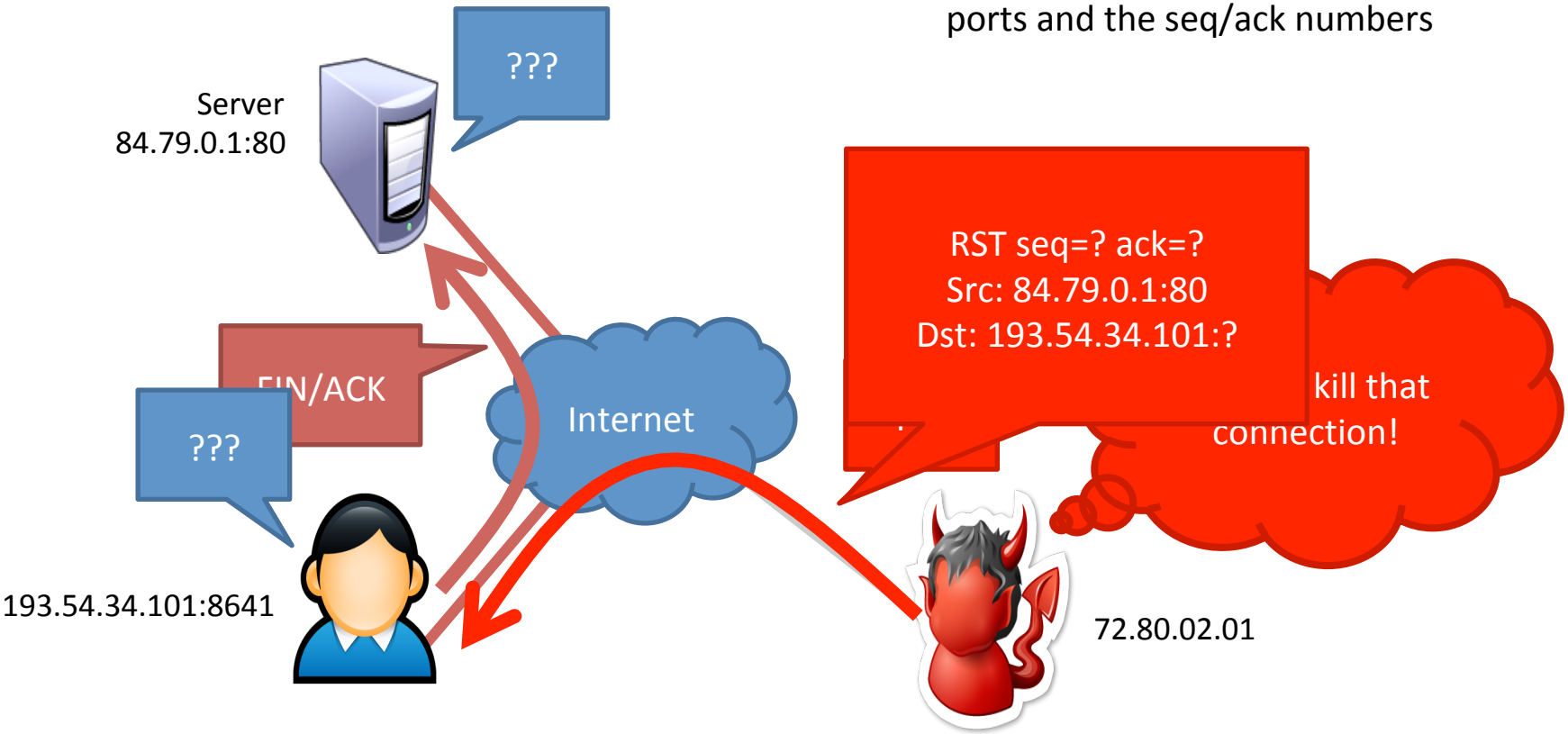
Three Way Handshake





Attack Strategies

- Send a FIN packet?
  - Triggers a multi-packet, graceful shutdown
- Send a RST packet?
  - Only works if you can guess both ports and the seq/ack numbers

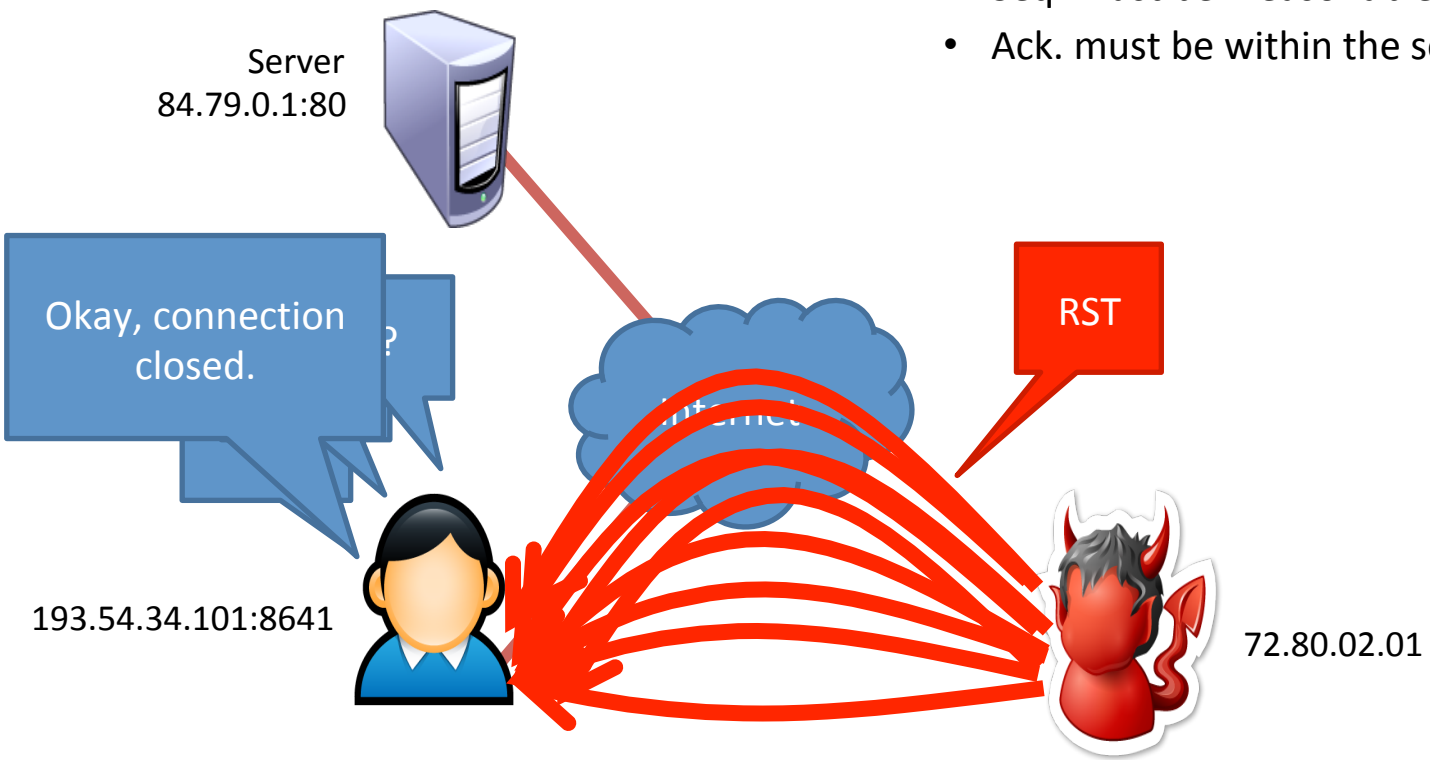


Guessing Sequence Numbers

- The security of TCP connections relies on the randomness of the initial sequence numbers (ISNs)
  - If an attacker knows the sequence numbers, they can spoof packets
- Problem: many OSs used to have low-entropy ISN generators
  - Typically seeded by the current time
  - RFC793 – “increase the ISN by 1 every 4 microseconds”
  - Windows NT 4.0 – *ISN = ms * 10 % 2^32*
- Attacker can measure the victim’s ISN generators
  - NTP query
  - Repeatedly open connections to known services like HTTP, FTP, echo, etc.

TCP RST Attack

- Server port is typically known, client port must be guessed
- Older OSs accepted a wide range of plausible seq/ack numbers with RST
- Modern OSs are more conservative
  - Seq. must be “reasonable”
  - Ack. must be within the sender’s window



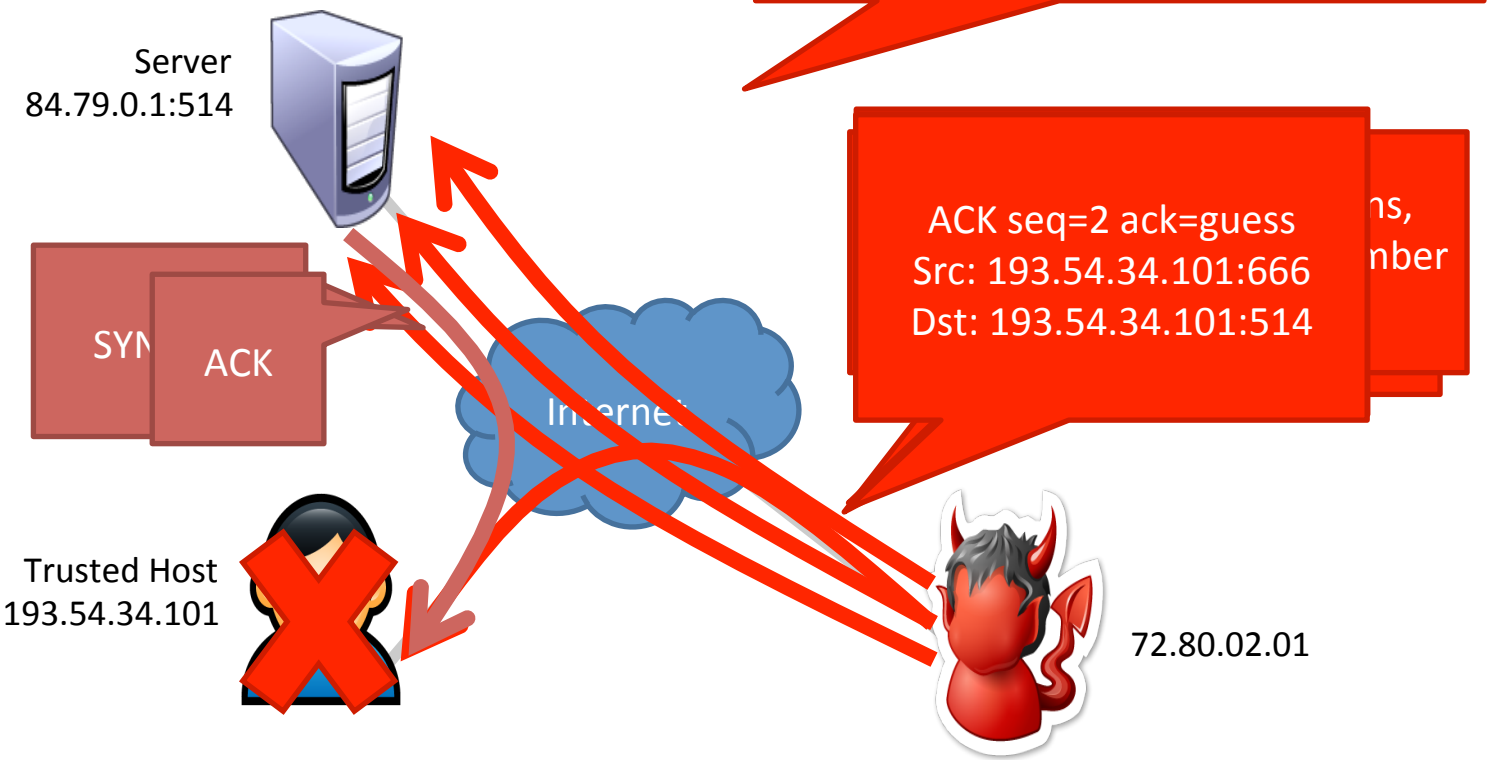
TCP Connection Hijacking

- RST attacks enable DoS, but not packet injection
- Attackers can hijack TCP connections by:
  1. Silencing one participant (*A*)
  2. Sending spoofed packets to the other participant (*B*)
- If *B* accepts a spoofed packet, the connection becomes desynchronized
- Why is it useful to silence one participant?
  - *A* may RST the connection if they observe a desynchronization or unsolicited packets

rsh Connection Hijacking Example

- Remote shell (*rsh*) was the predecessor to *ssh*
- Typical … figured list of "
- Attacke … a trusted host

Finish login, redirect stdout/stderr to /dev/null, create a new account for the attacker, etc.



Modern TCP

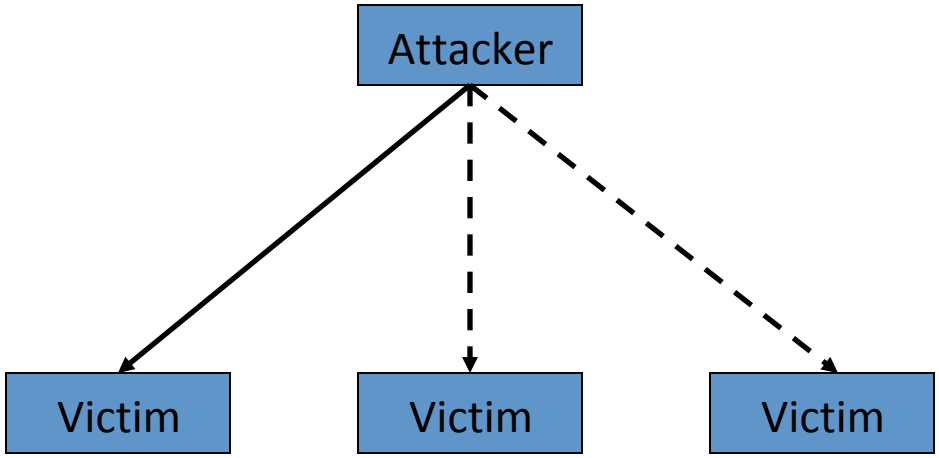
- Modern OSs choose ISNs purely at random
  - This makes off-path hijacking attacks extremely difficult
- Does this mean TCP is now secure from spoofing and hijacking?
  - No.
  - On-path attackers still see everything, no guessing required
  - May drop, modify, or inject packets at will
- This is why we need IPSEC, TLS, etc.

DDoS Attacks

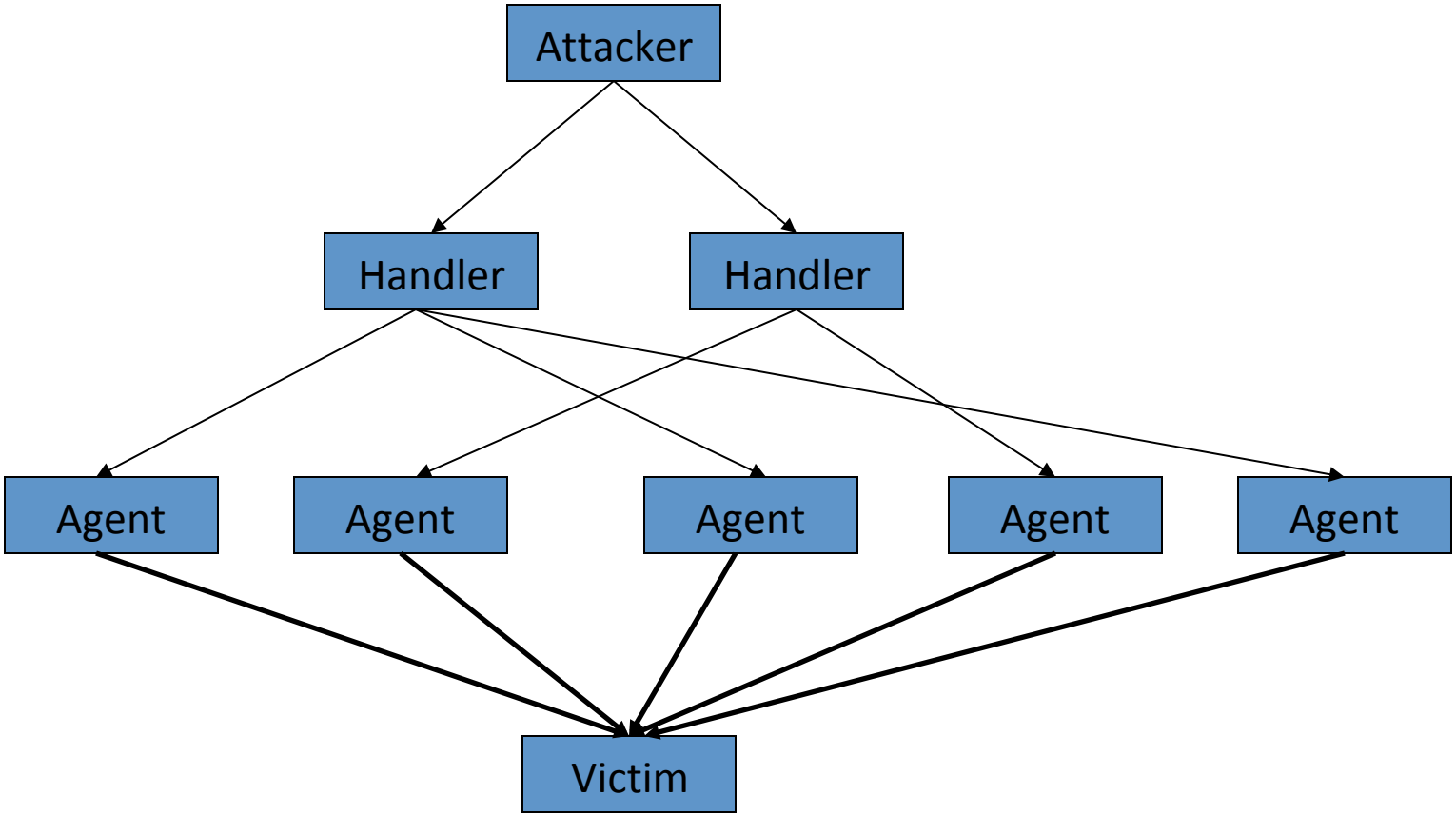
Attacks against Availability: Denial of Service attacks

- An attempt to consume finite resources, exploit weaknesses in software design or implementation, or exploit lack of infrastructure capacity
- Effects the availability and utility of computing and network resources

Simple DoS



Distributed DoS



DoS History

- Locally-induced crash
  - exploit operating system or server software bug
- Local resource consumption
  - fork() bomb, fill disks, deep directory nesting
- Deny service to individual hosts
  - force crash or outage of critical services
- Remotely-induced crash
  - “magic” packets – ping of death, teardrop
- Remote resource consumption
  - syslog, SYN, fragment flood, UDP storm

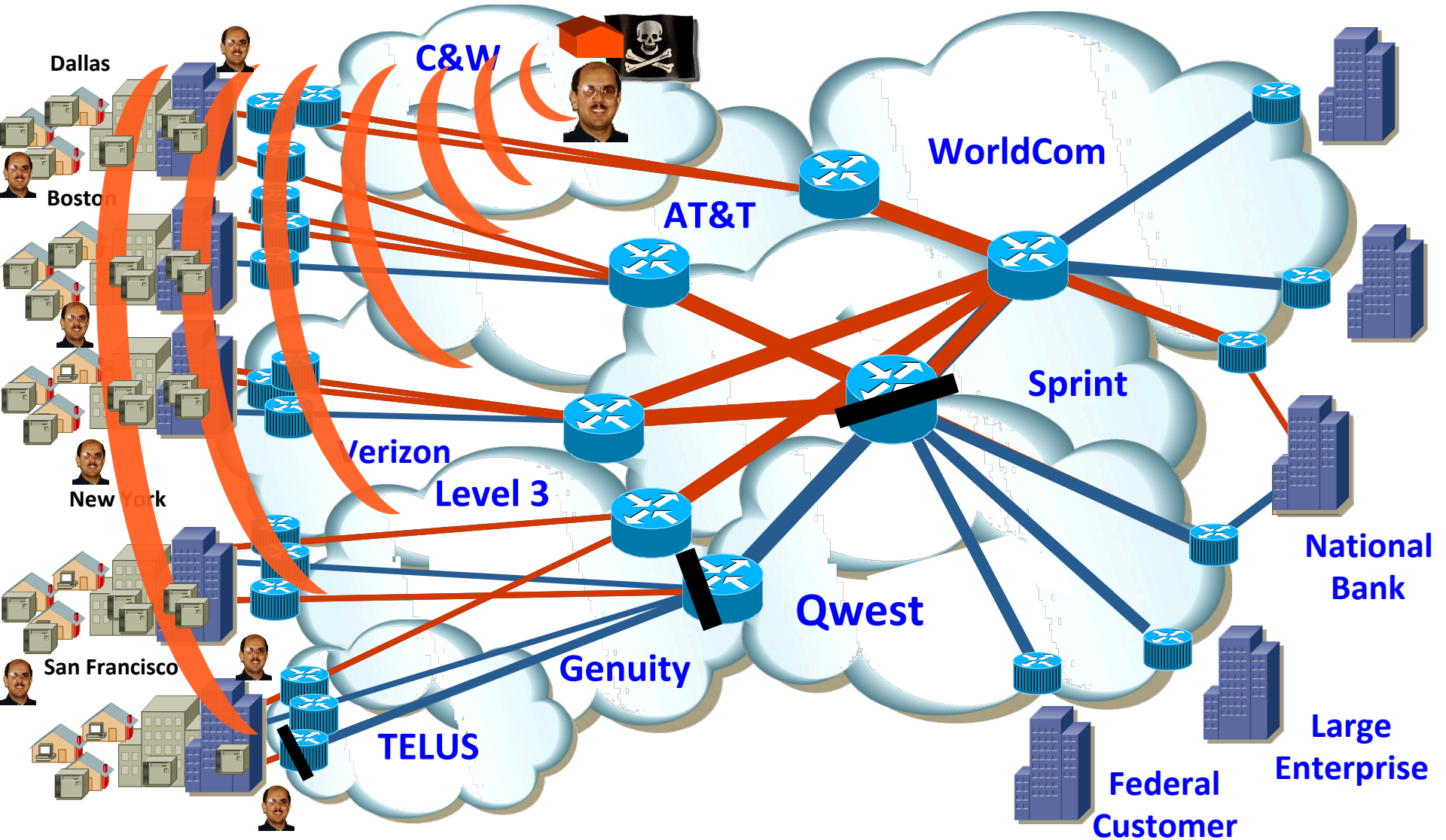
DoS History (cont.)

- Deny service to an entire network
  - target vulnerable links or critical network infrastructure / information
- Remotely-induced network outage
  - attacks against routers, DNS servers
  - redirected routes – forged routing information
- Remote network congestion
  - forged directed broadcasts – smurf, fraggle
  - remote control of compromised hosts (“zombies”) for coordinated flooding – DDoS

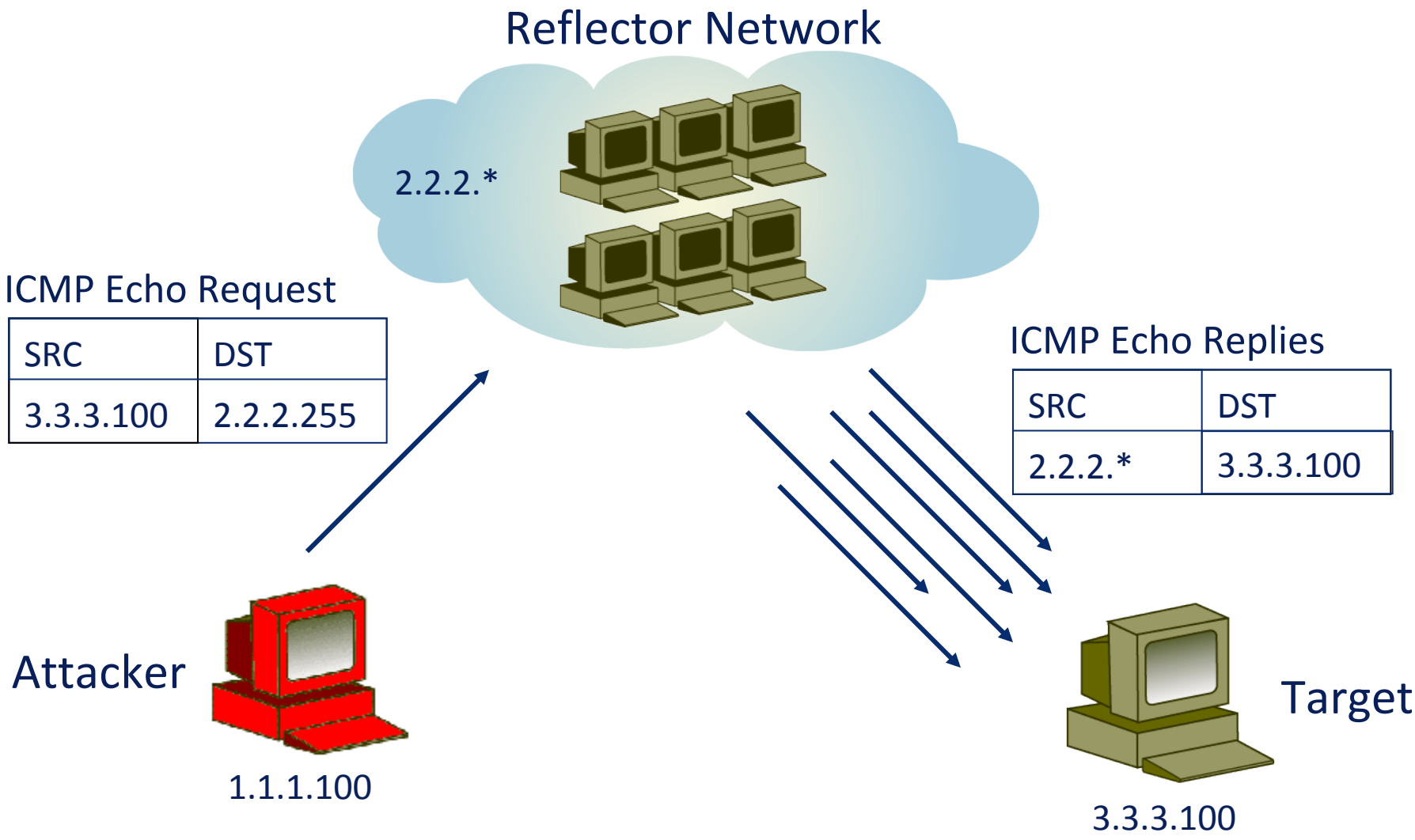
Timeline of a DDoS attack

- A large set of machines are compromised
- Attacker identifies exploitable hosts with scanners, or other techniques
- Attacker accesses the system with automated remote exploits, sniffers, password cracking, worms, trojans
- Attacker installs attack tools
- Attacker remotely instructs compromised machines to attack target

Distributed Denial of Service

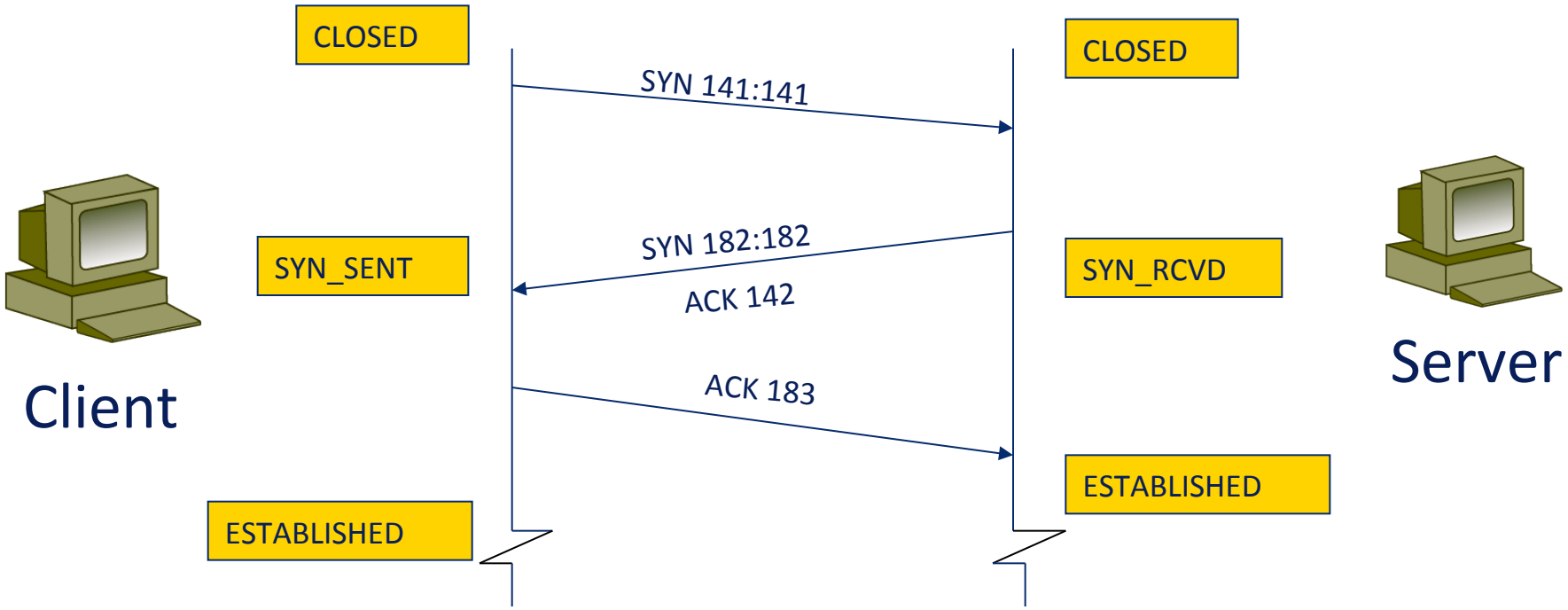


Blind spoofing and Smurf Denial of Service (DOS) Attacks

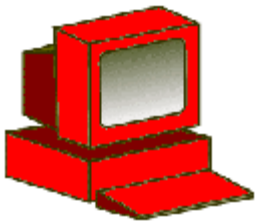


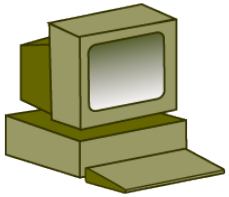
Example: TCP SYN Flood

Normal sequence for TCP connection establishment (3-way handshake)



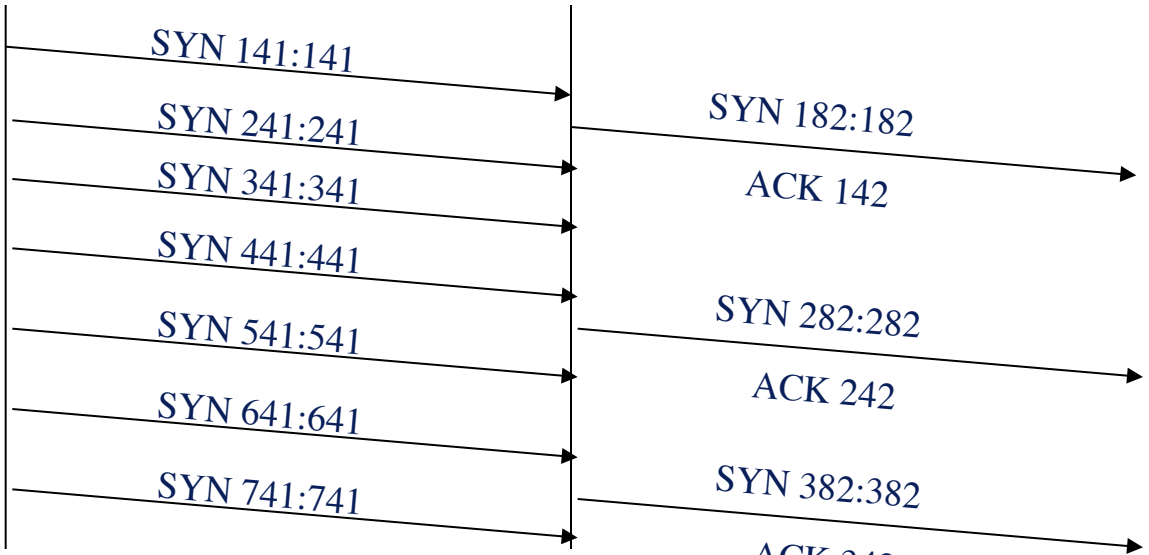
Example: TCP SYN Flood (cont.)



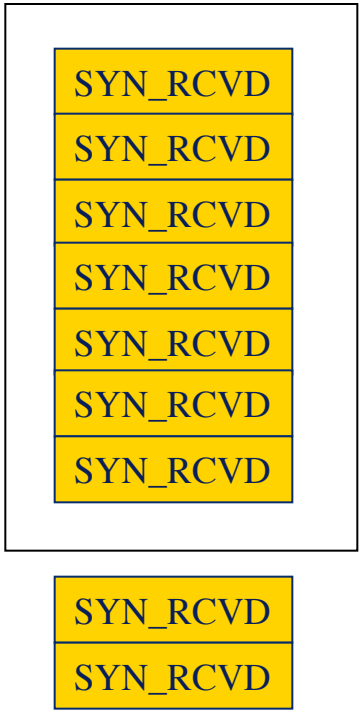


Attacker

Server



Listen Queue



DNS Attacks

DNS: Domain Name System

People: many identifiers:

- SSN, name, passport #

Internet hosts, routers:

- IP address (32 bit) - used for addressing datagrams
- "name", e.g., www.eecs.umich.edu - used by humans

Q: map between IP addresses and name ?

Domain Name System:

- *distributed database* implemented in hierarchy of many *name servers*
- *application-layer protocol* host, routers, name servers to communicate to *resolve* names (address/name translation)
  - note: core Internet function, implemented as application-layer protocol
  - complexity at network's "edge"

DNS name servers

Why not centralize DNS?

- single point of failure
- traffic volume
- distant centralized database
- Maintenance

doesn't *scale!*

Record Type	Million Queries	Percentage
A	1,220	70.4%
AAAA	206	11.9%
MX	152	8.8%
DS	69	4.0%
NS	25	1.4%
ANY	19	1.1%
TXT	18	1.0%
SOA	6	0.4%
A6	5	0.3%
SPF	4	0.3%
Other	8	0.5%
Total	1,732	

Table 8: Top 10 Resource Records Requested

- no server has all name-to-IP address mappings

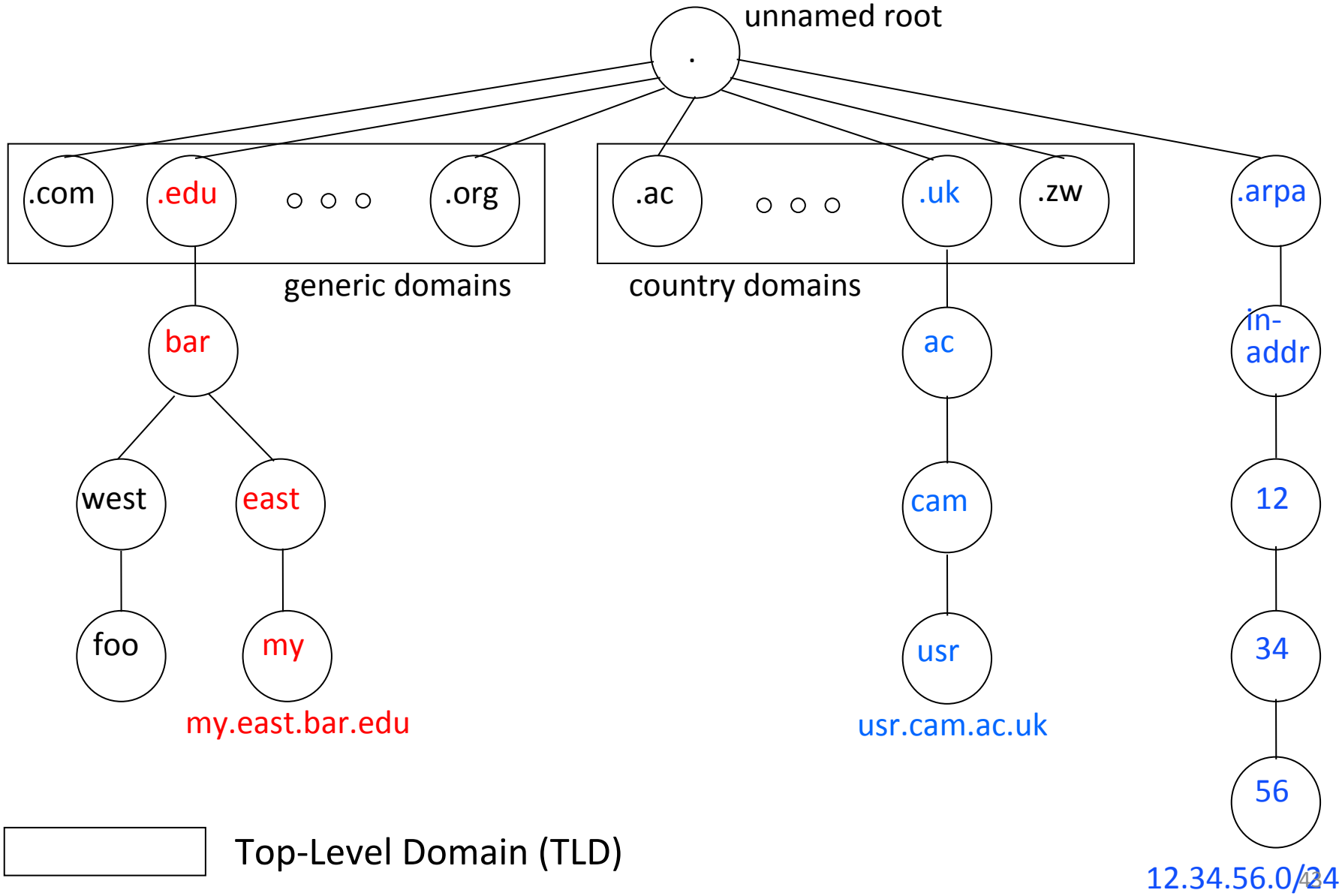
local name servers:

- each ISP, company has *local (default) name server*
- host DNS query first goes to local name server

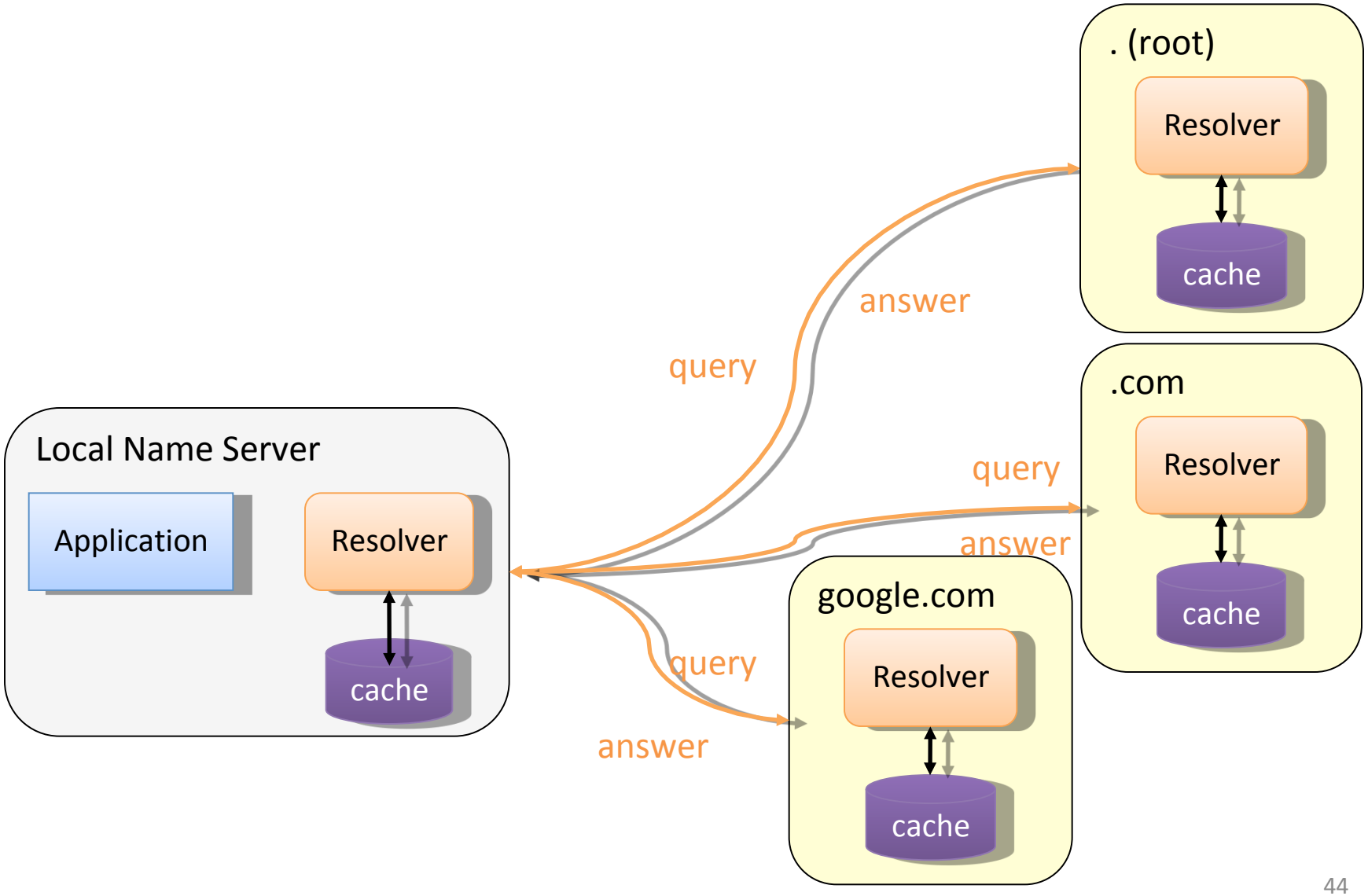
authoritative name server:

- for a host: stores that host's IP address, name
- can perform name/address translation for that host's name

DNS Hierarchical Name Space

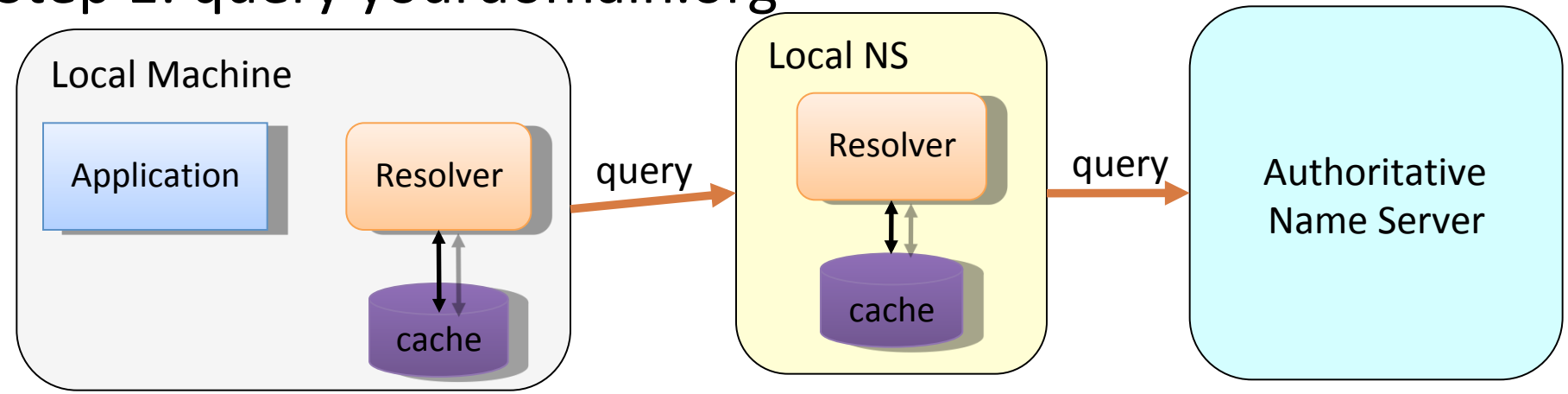


Iterative Name Resolution

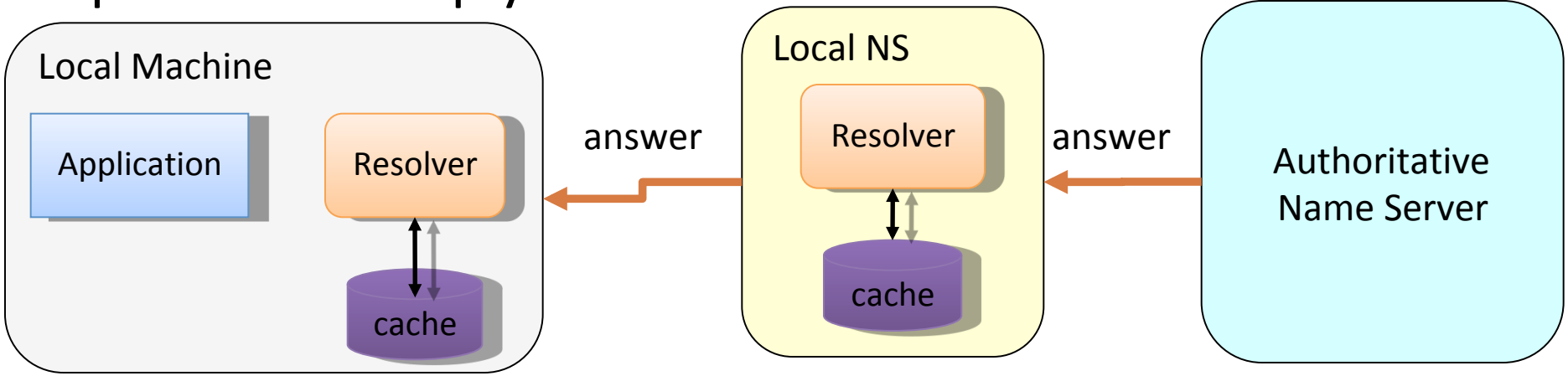


DNS Caching

Step 1: query yourdomain.org

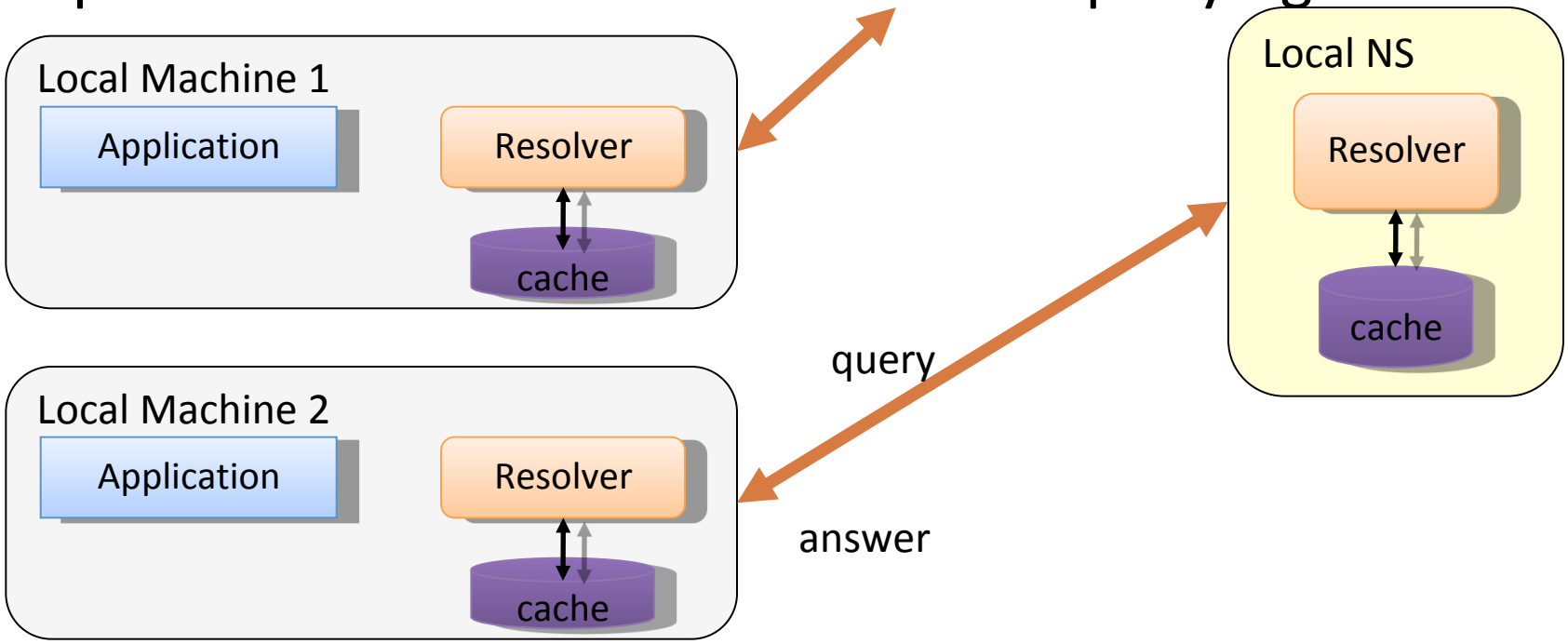


Step 2: receive reply and cache at local NS and host



DNS Caching (con'd)

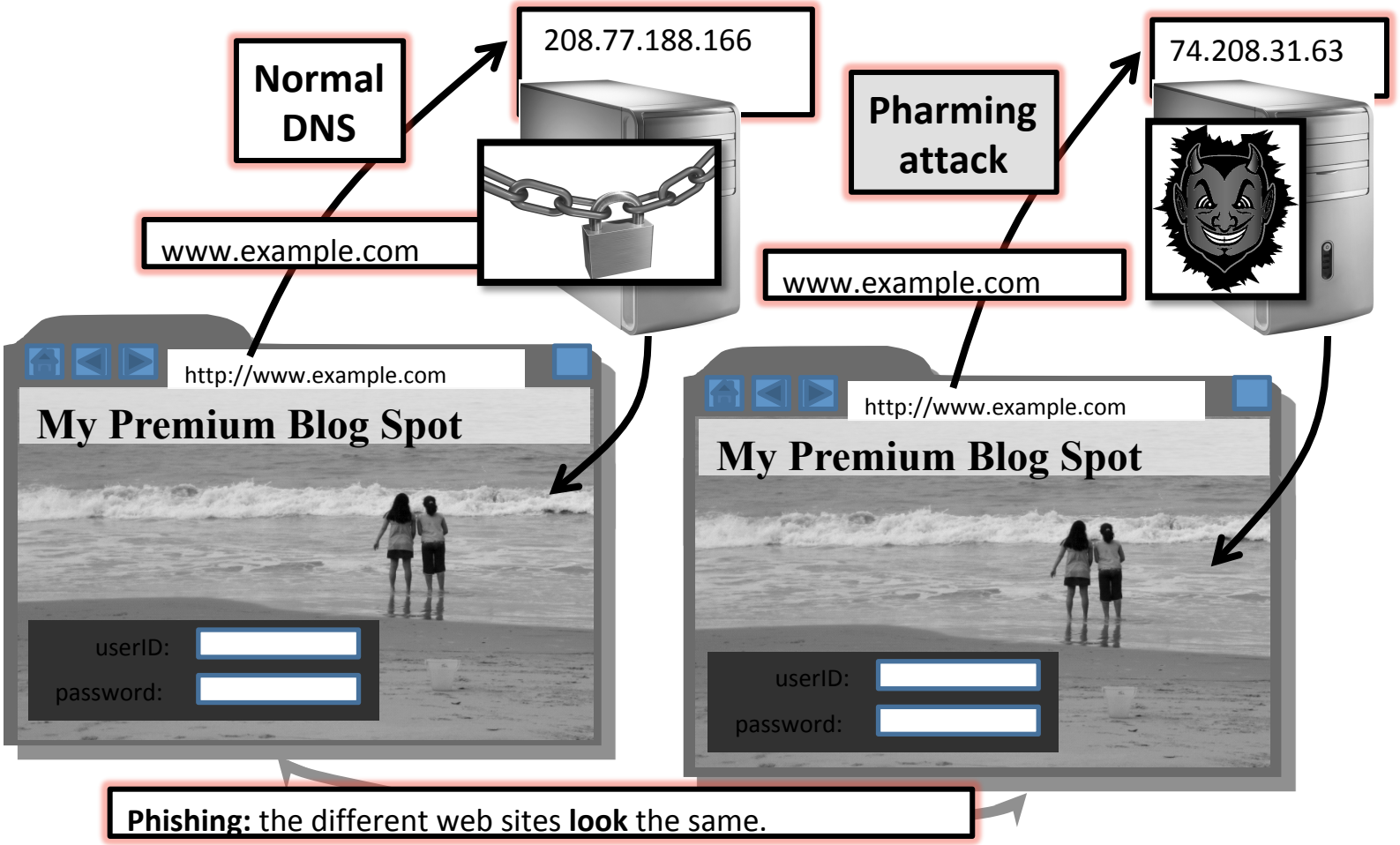
Step 3: use cached results rather than querying the ANS



Step 4: Evict cache entries upon ttl expiration

Pharming: DNS Hijacking

- Changing IP associated with a server maliciously:

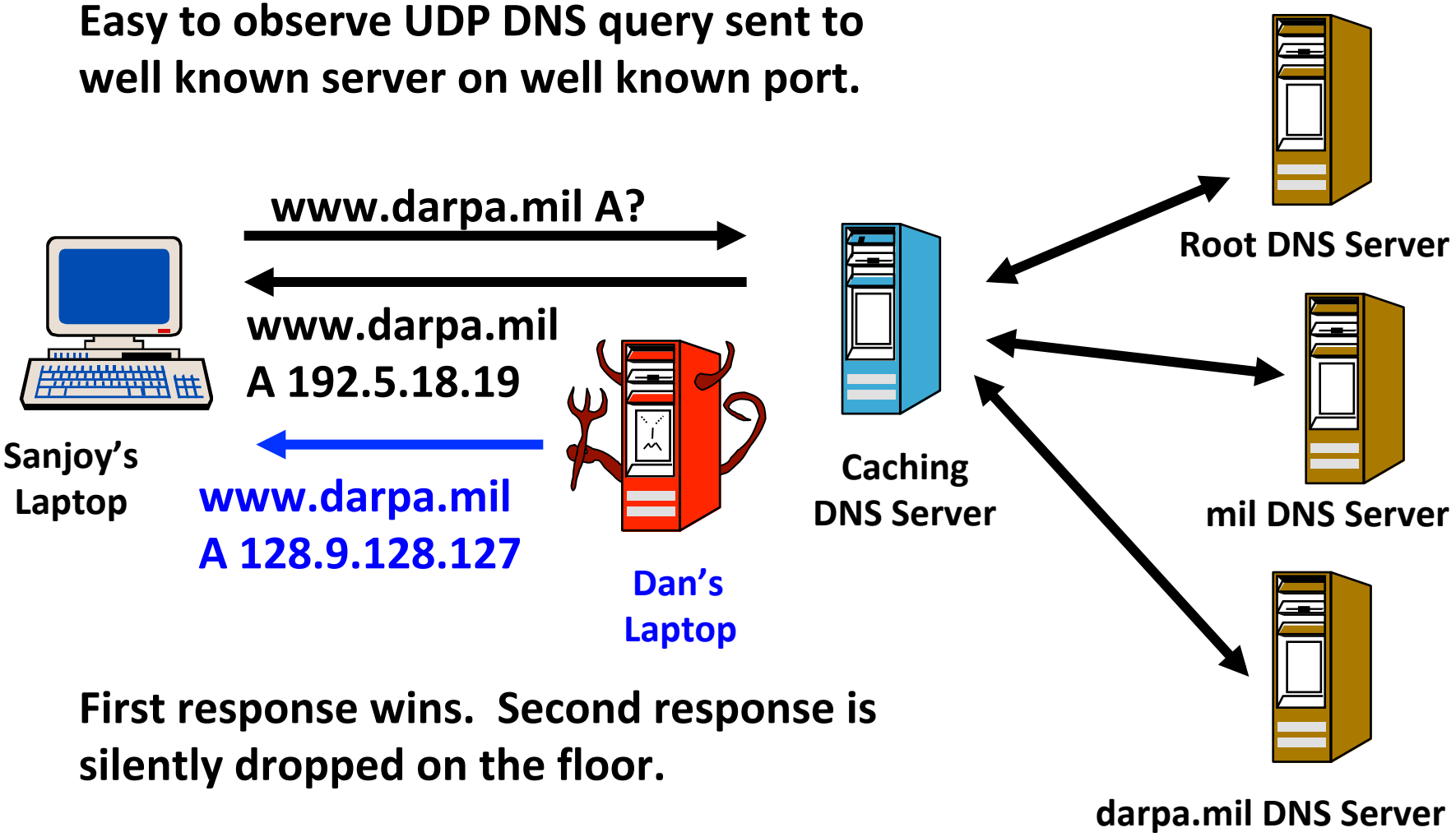


DNS spoofing

- Problem
  - No authentication of responses
  - Any DNS response is generally believed.
  - No attempt to distinguish valid data from invalid.
  - Responses can contain entries that should not be trusted but are
  - Responses are cached
  - Just one false root server could disrupt the entire DNS.
- Attacks
  - Inject bogus DNS responses
  - Attach additional bogus entries in valid DNS responses (especially for internal names)

DNS spoofing

Easy to observe UDP DNS query sent to well known server on well known port.

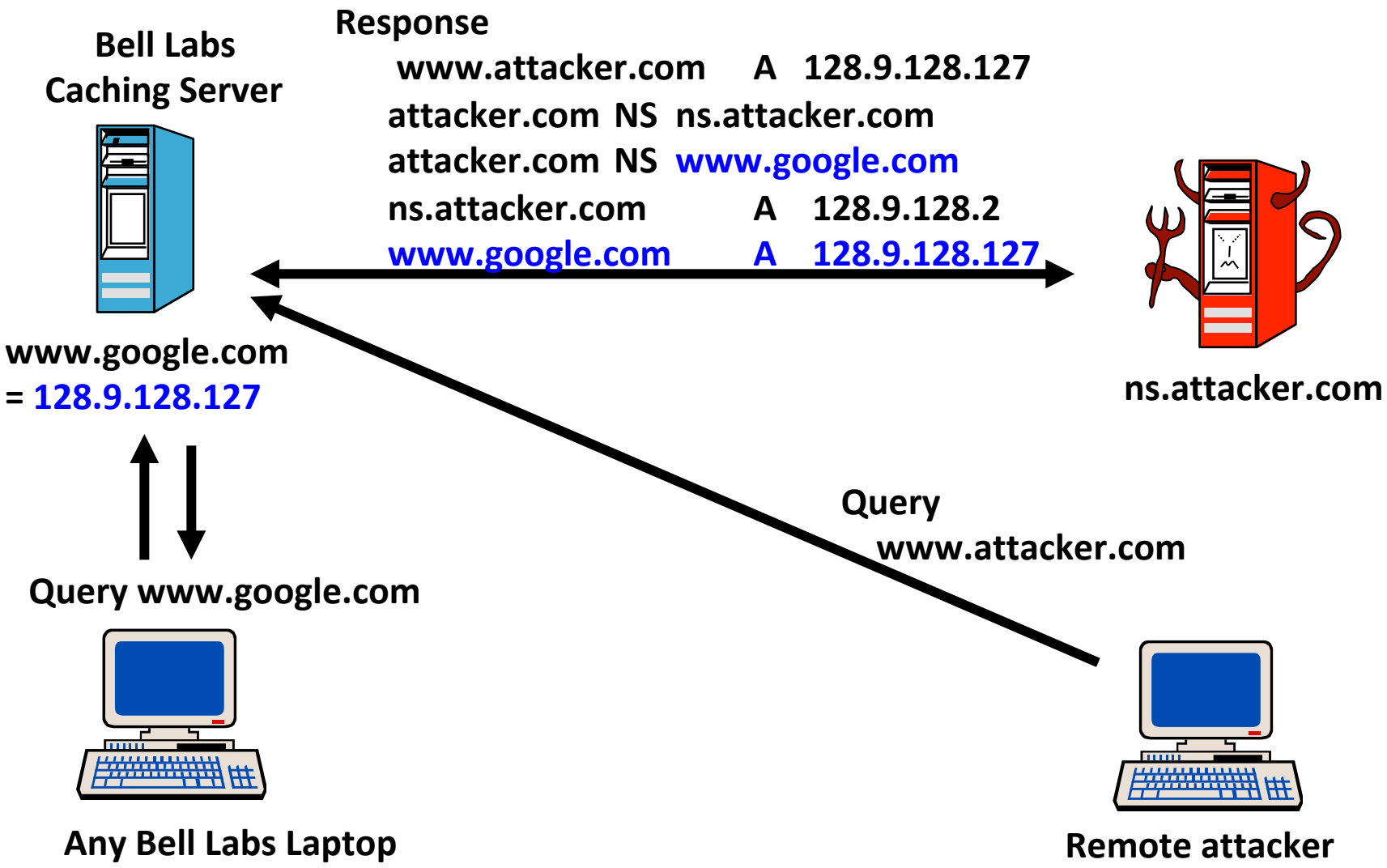


First response wins. Second response is silently dropped on the floor.

DNS Cache Poisoning

- Basic idea: give DNS servers false records and get it cached
- DNS uses a 16-bit request identifier to pair queries with answers
- Cache may be poisoned when a name server:
  - Disregards identifiers
  - Has predictable ids
  - Accepts unsolicited DNS records

DNS cache poisoning



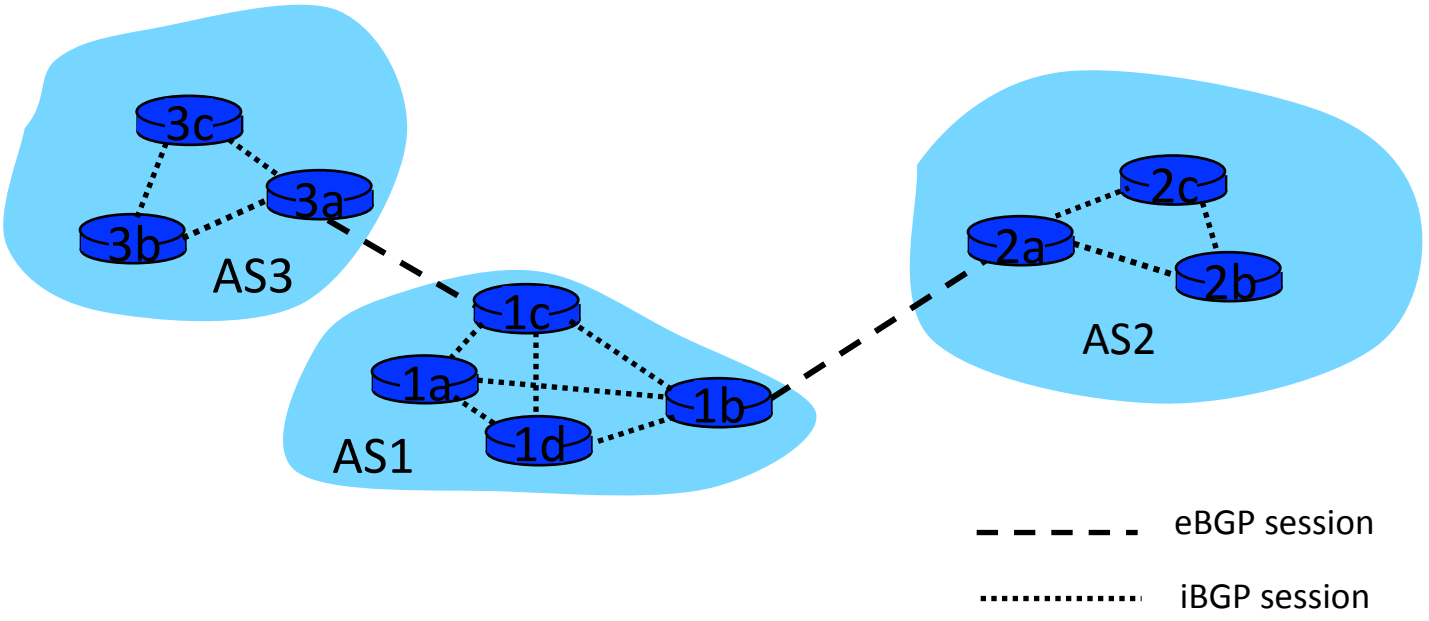
Routing Attacks

Internet inter-AS routing: BGP

- BGP (Border Gateway Protocol): *the* de facto standard
- BGP provides each AS a means to:
  1. Obtain subnet reachability information from neighboring ASs.
  2. Propagate the reachability information to all routers internal to the AS.
  3. Determine "good" routes to subnets based on reachability information and policy.
- Allows a subnet to advertise its existence to rest of the Internet: *"I am here"*

BGP basics

- Pairs of routers (BGP peers) exchange routing info over semi-permanent TCP connections: BGP sessions
- Note that BGP sessions do not correspond to physical links.
- When AS2 advertises a prefix to AS1, AS2 is *promising* it will forward any datagrams destined to that prefix towards the prefix.
  - AS2 can aggregate prefixes in its advertisement

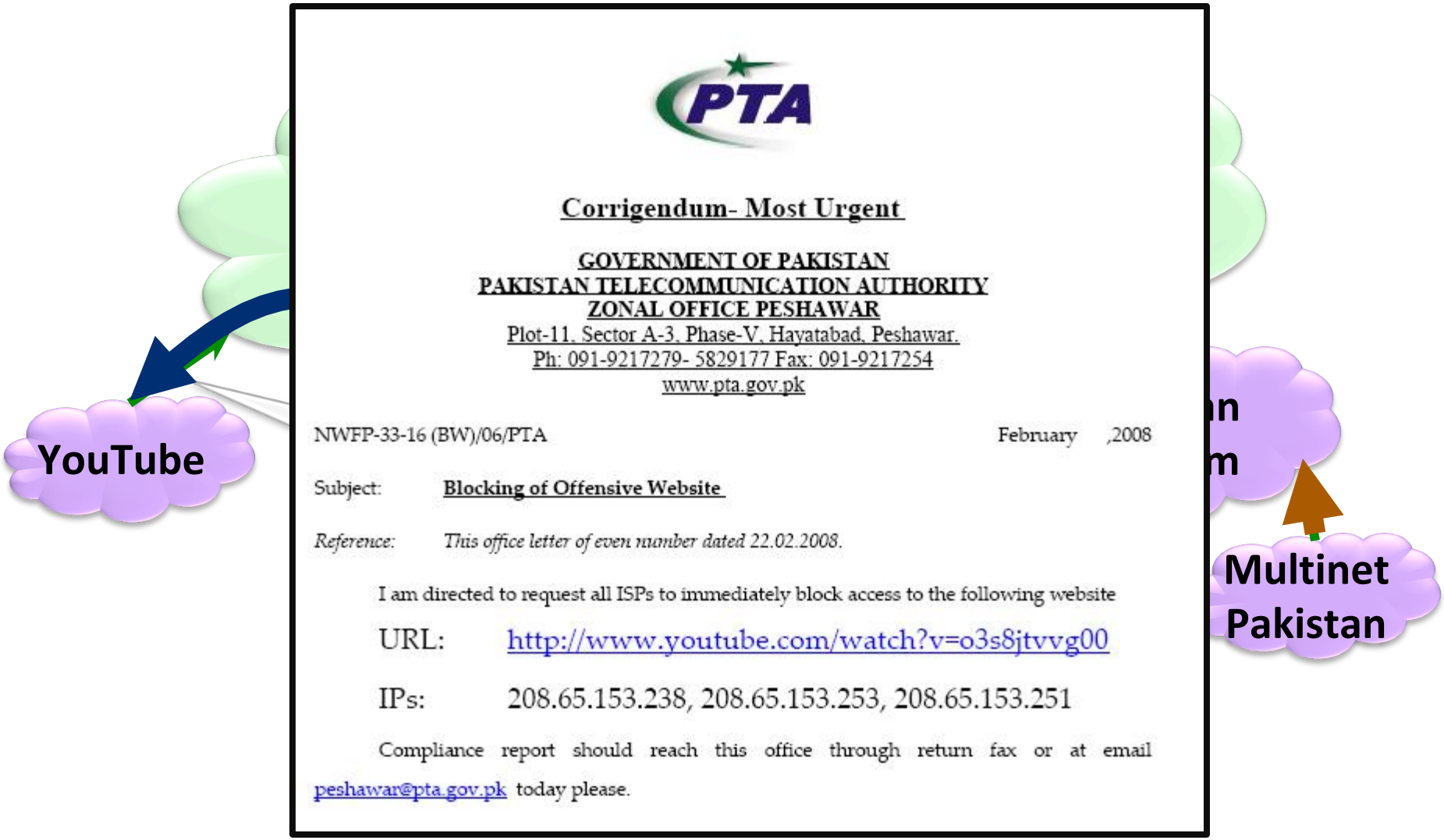


How to Hijack a Prefix

- The hijacking AS has
  - Router with eBGP session(s)
  - Configured to originate the prefix
- Getting access to the router
  - Network operator makes configuration mistake
  - Disgruntled operator launches an attack
  - Outsider breaks in to the router and reconfigures
- Getting other ASes to believe bogus route
  - Neighbor ASes not filtering the routes
  - … e.g., by allowing only expected prefixes
  - But, specifying filters on *peering* links is hard

http://www3.cs.stonybrook.edu/~phillipa/CSE390/RoutingSecurity.pptx

Pakistan Telecom: Sub-prefix hijack



Corrigendum- Most Urgent

GOVERNMENT OF PAKISTAN
PAKISTAN TELECOMMUNICATION AUTHORITY
ZONAL OFFICE PESHAWAR
Plot-11, Sector A-3, Phase-V, Hayatabad, Peshawar.
Ph: 091-9217279- 5829177 Fax: 091-9217254
www.pta.gov.pk

NWFP-33-16 (BW)/06/PTA February ,2008

Subject: **Blocking of Offensive Website**

Reference: This office letter of even number dated 22.02.2008.

I am directed to request all ISPs to immediately block access to the following website

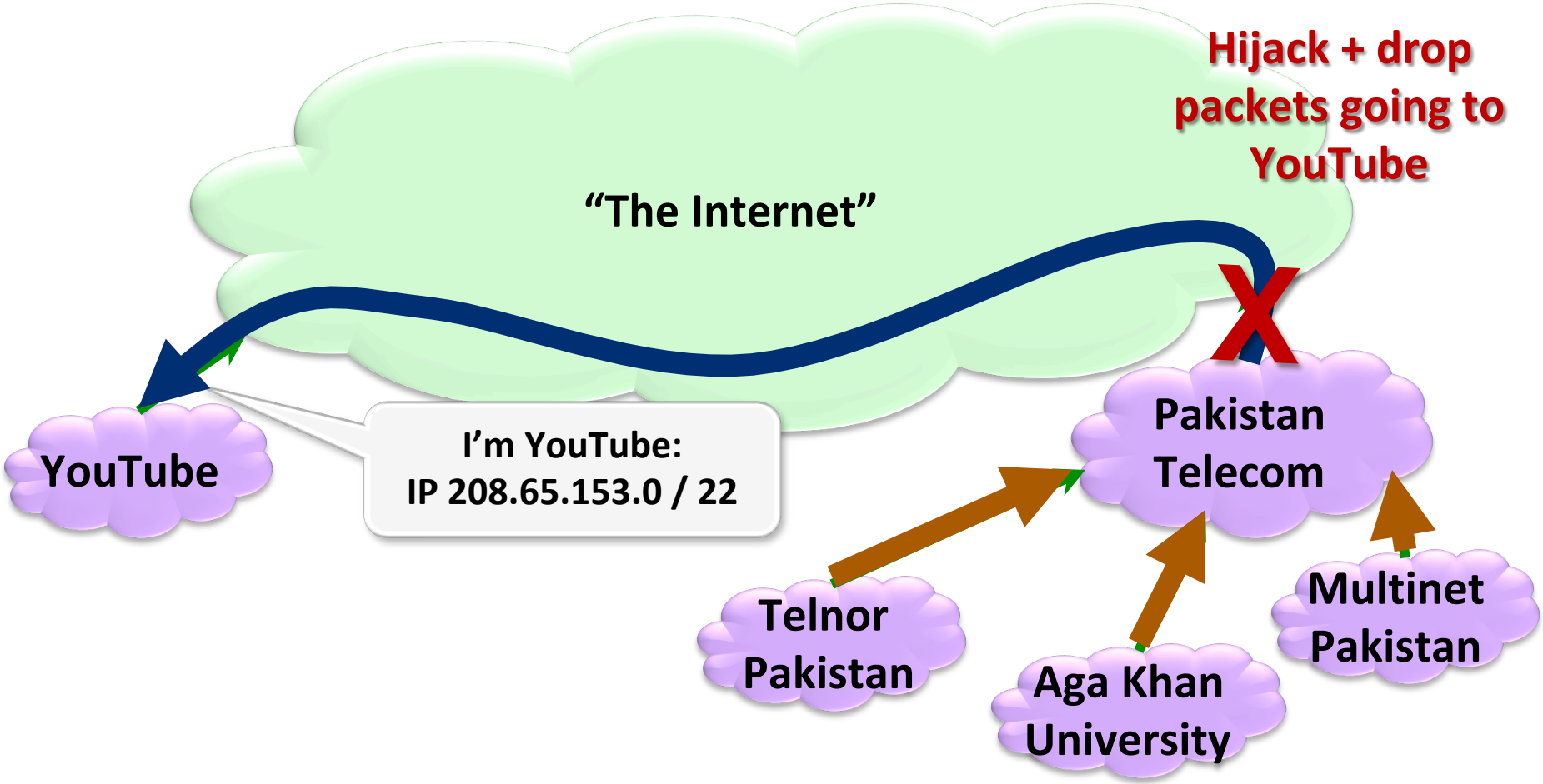
URL: http://www.youtube.com/watch?v=o3s8jtvvg00

IPs: 208.65.153.238, 208.65.153.253, 208.65.153.251

Compliance report should reach this office through return fax or at email peshawar@pta.gov.pk today please.

Pakistan Telecom: Sub-prefix hijack

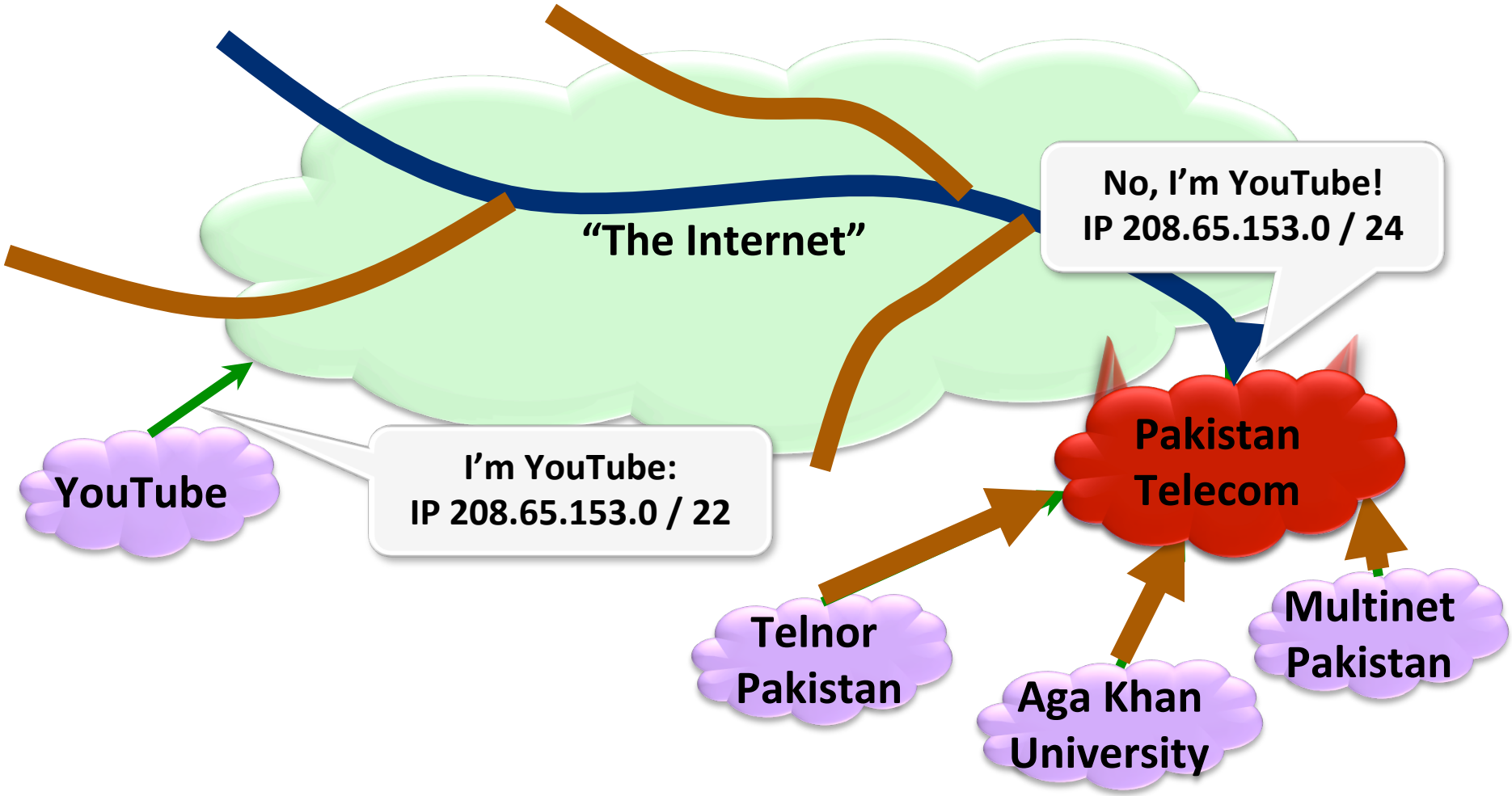
Here's what should have happened….



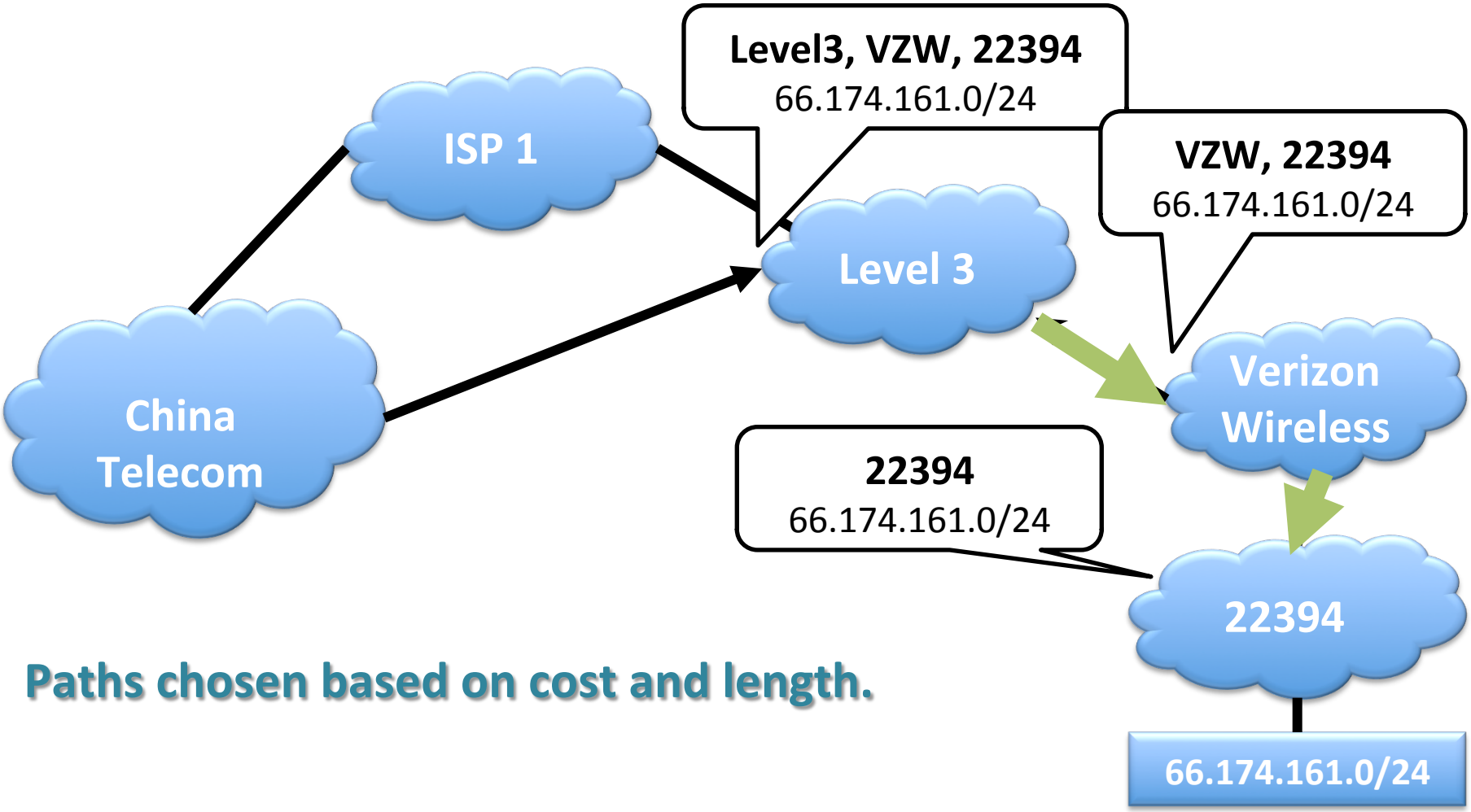
Block your own customers.

Pakistan Telecom: Sub-prefix hijack

But here's what Pakistan ended up doing…



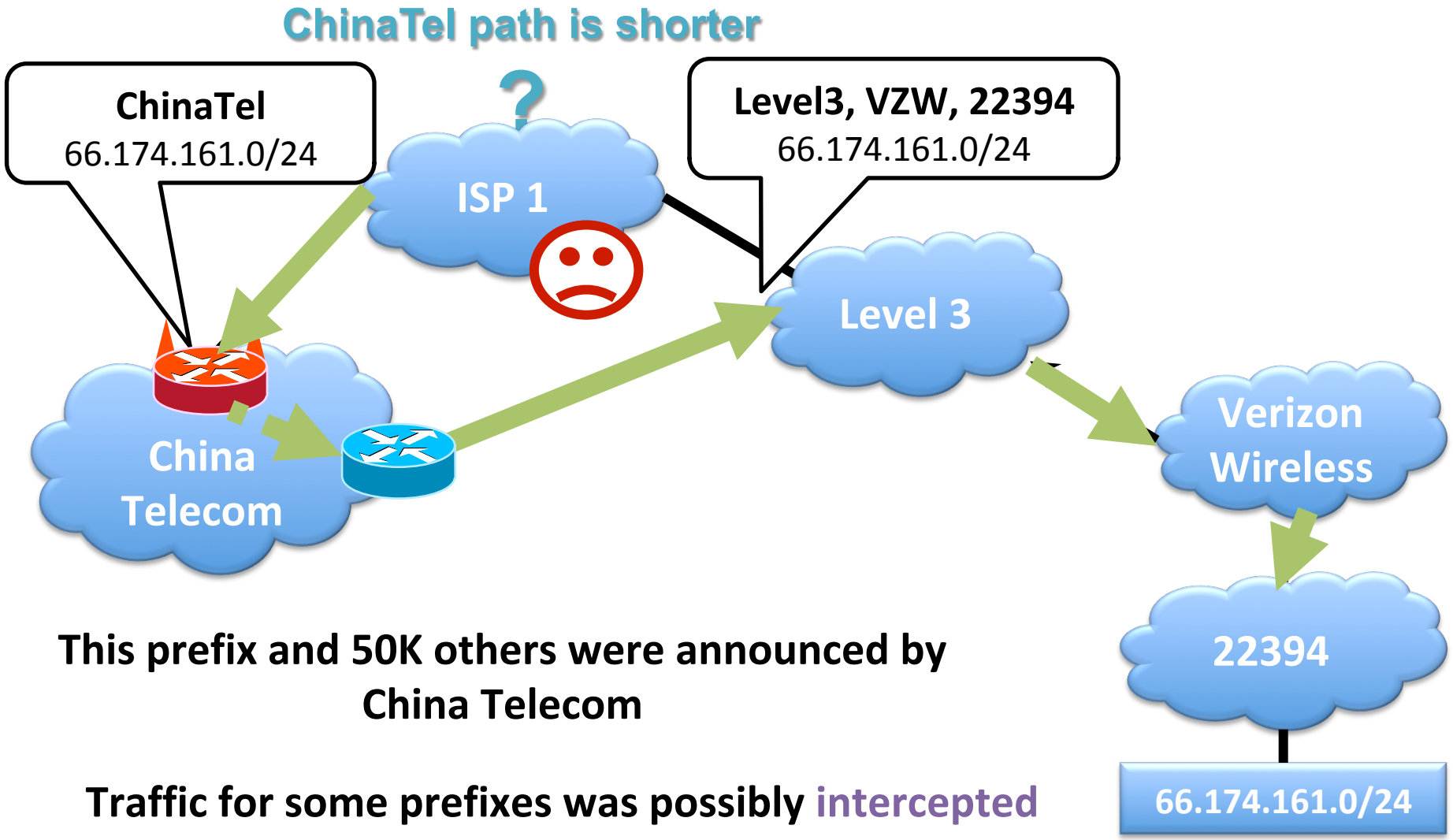
China Telecom: Interception



Paths chosen based on cost and length.

China Telecom: Interception

April 2010 : China Telecom intercepts traffic



This prefix and 50K others were announced by China Telecom

Traffic for some prefixes was possibly intercepted

Canadian Bitcoin hijack (2/3/2014)

- Bitcoin Background
- Miners solve cryptographic puzzles using their computing power
  - Once the puzzle is solved new bitcoins are created and the miner gets some reward
- Mining is generic: mining pool dictates currency
  - E.g., dogecoin, bitcoin etc.
- Miners communicate with pool server using a protocol called 'stratum'
  - Mining server can redirect user to a different server (e.g., for load balancing)

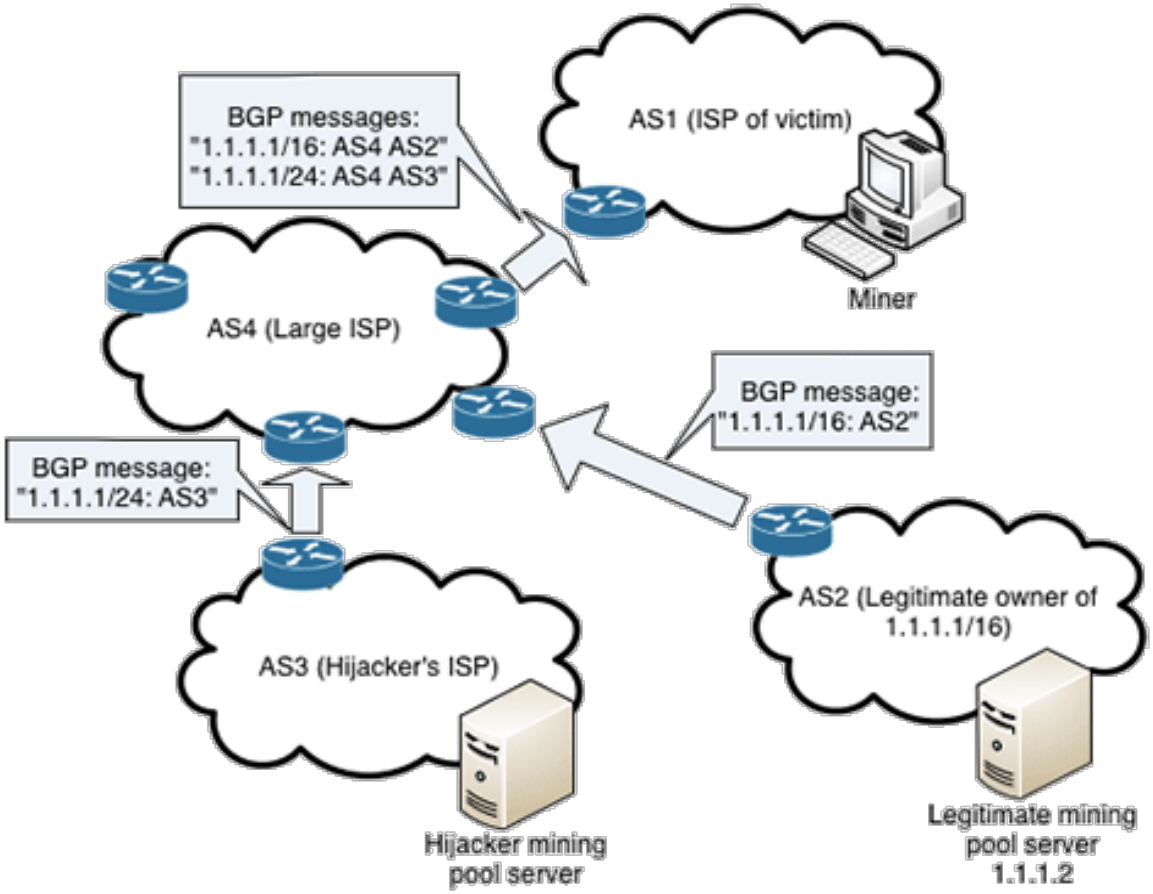
http://www.secureworks.com/cyber-threat-intelligence/threats/bgp-hijacking-for-cryptocurrency-profit/

Canadian Bitcoin hijack (2/3/2014)

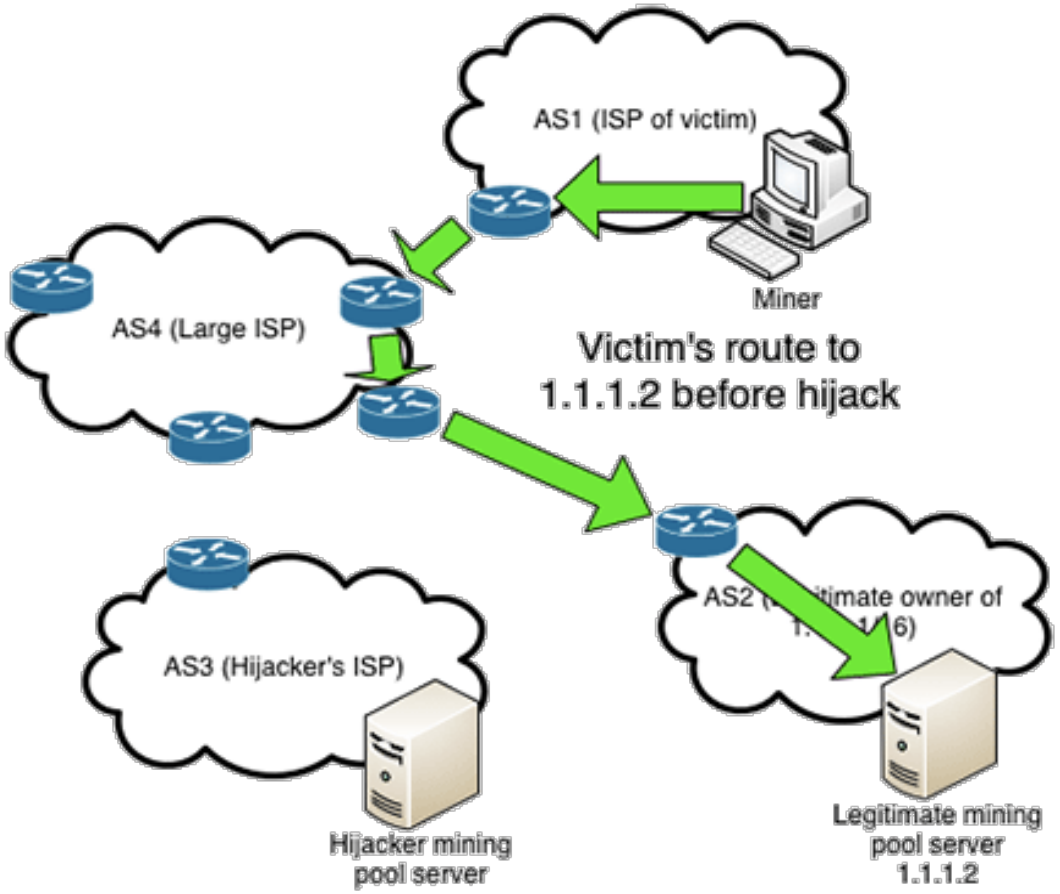
- Hijacked users got directed to a mining server IP that was under the control of the hijacker and redirects them to a malicious mining pool.
- Miners continue to receive tasks and solve puzzles but don't get compensated. ☹

http://www.secureworks.com/cyber-threat-intelligence/threats/bgp-hijacking-for-cryptocurrency-profit/

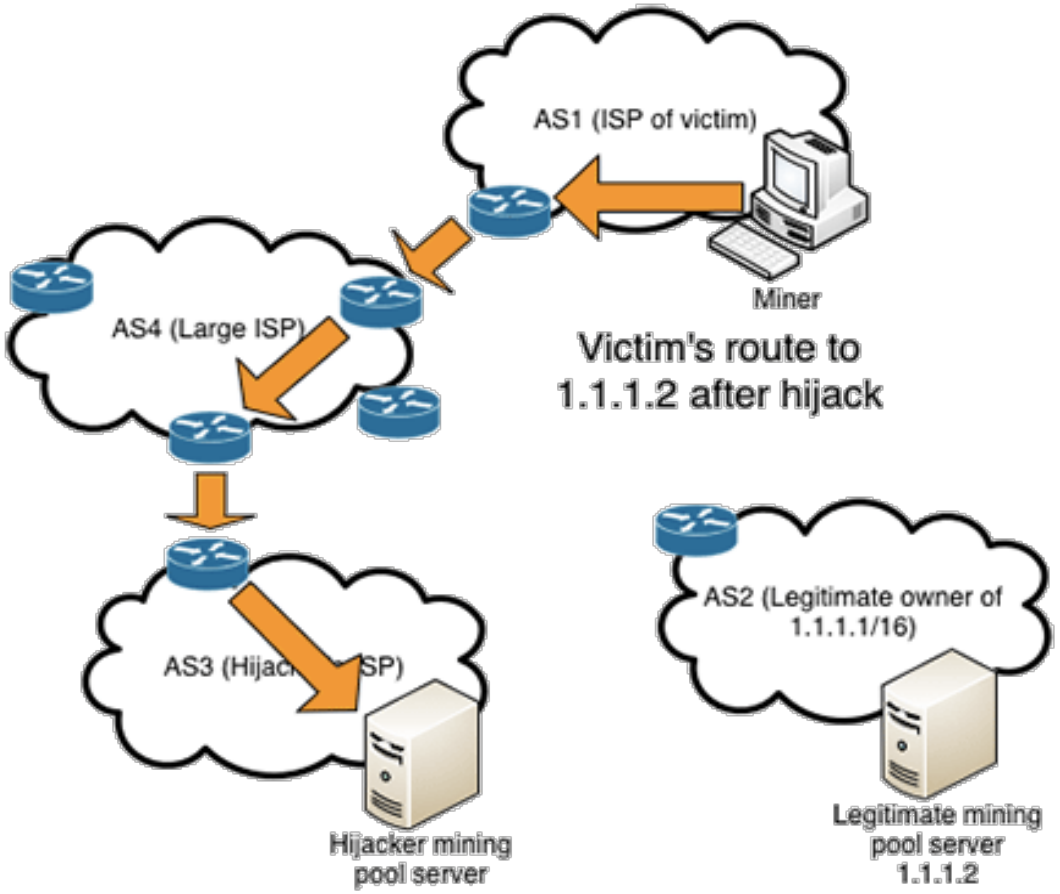
Canadian Bitcoin hijack (2/3/2014)



Canadian Bitcoin hijack (2/3/2014)



Canadian Bitcoin hijack (2/3/2014)



Canadian Bitcoin hijack (2/3/2014)