https://brown-csci1660.github.io

CS1660: Intro to Computer Systems Security
Spring 2026

Lecture 22: Network Security II

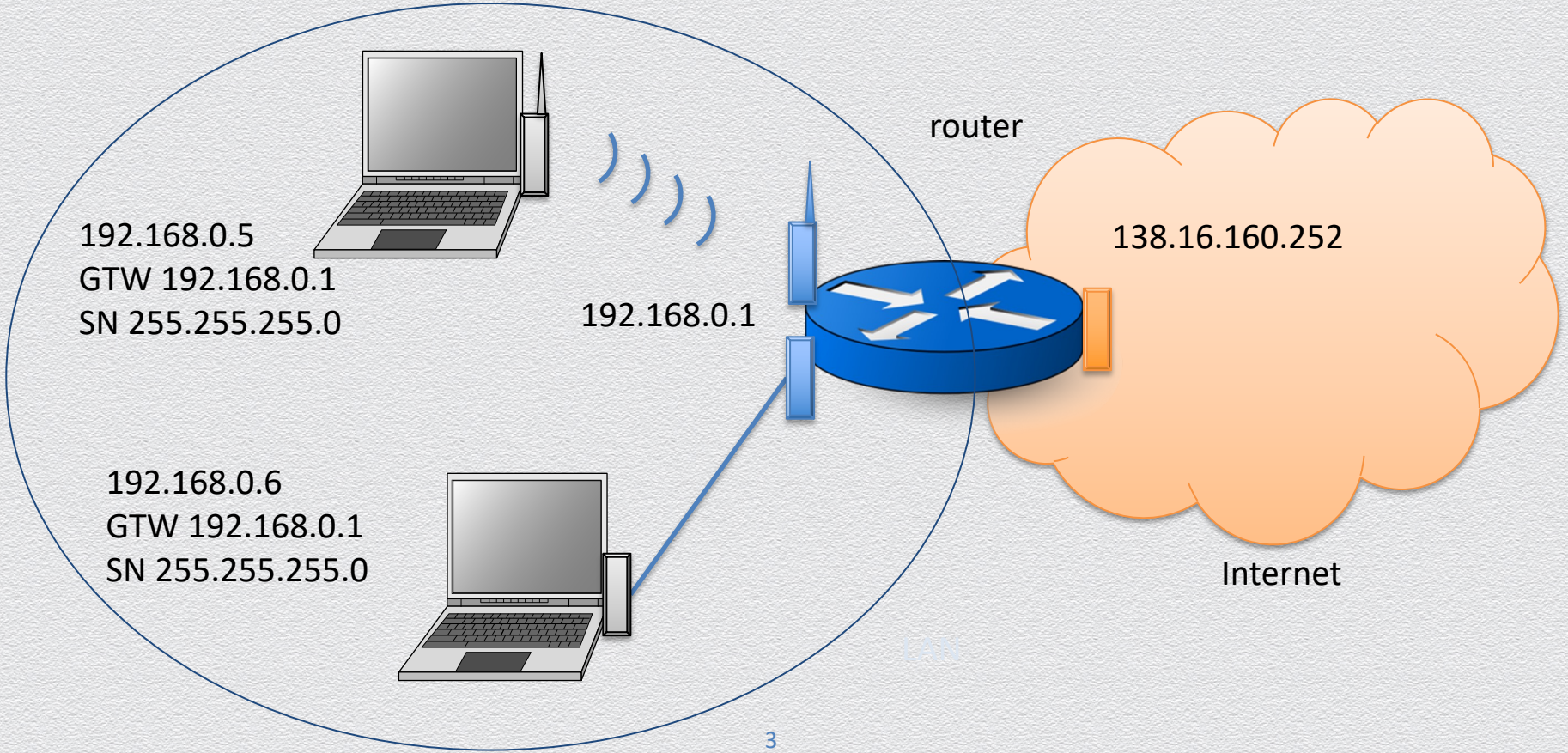
Instructor: **Nikos Triandopoulos**

April 23, 2026

BROWN

22.1 Network layer

From LAN to the Internet



Obtaining Host IP Addresses

Dynamic Host Configuration Protocol (DHCP)

- Networks are free to assign addresses within block to hosts
- Tedious and error-prone
  - E.g., laptop going from CIT to library to coffee shop
- Idea
  - Client asks network for IP on connection

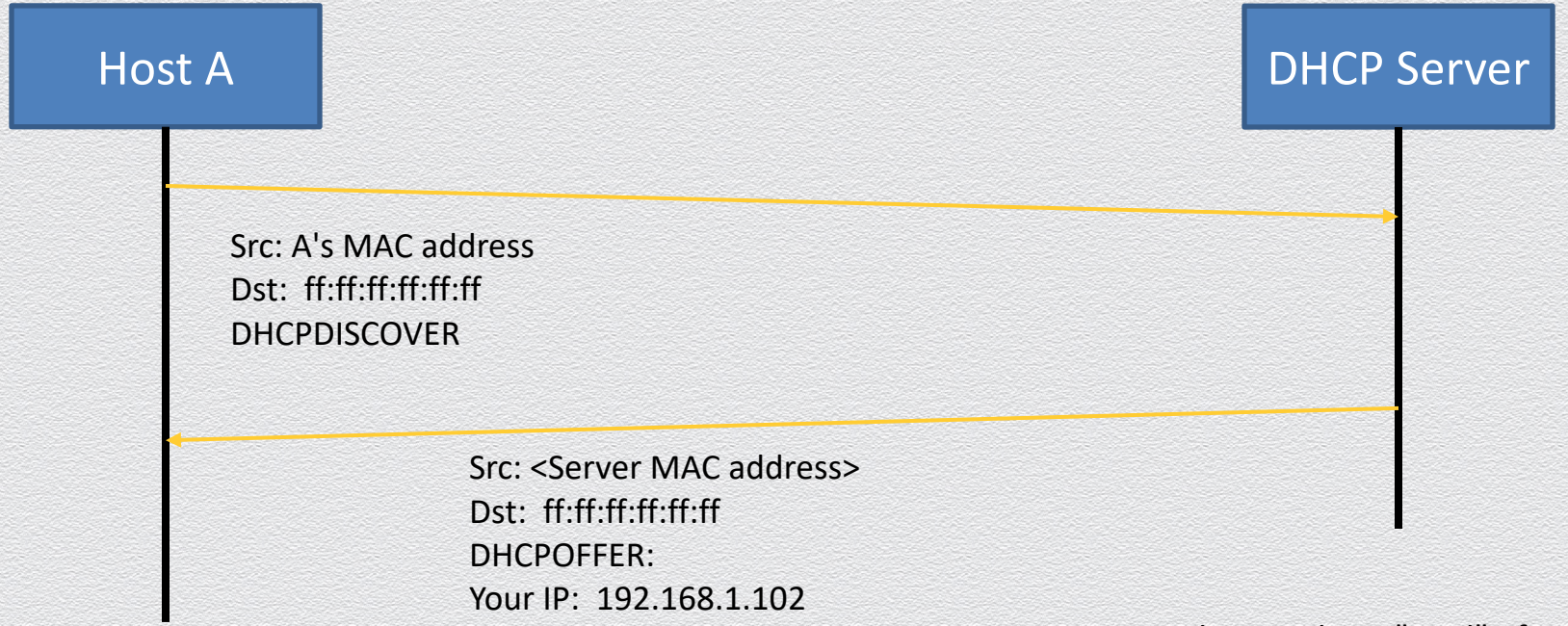
But how? How to send packets with no IP address?

Broadcast traffic

Special MAC address: ff:ff:ff:ff:ff:ff

- Forwarded to all hosts on network
- Used for link-layer protocols, particularly for finding IP addresses (DHCP, ARP)
- Each IP subnet also has a broadcast address, usually last IP (e.g., 192.168.1.255)

Start of DHCP

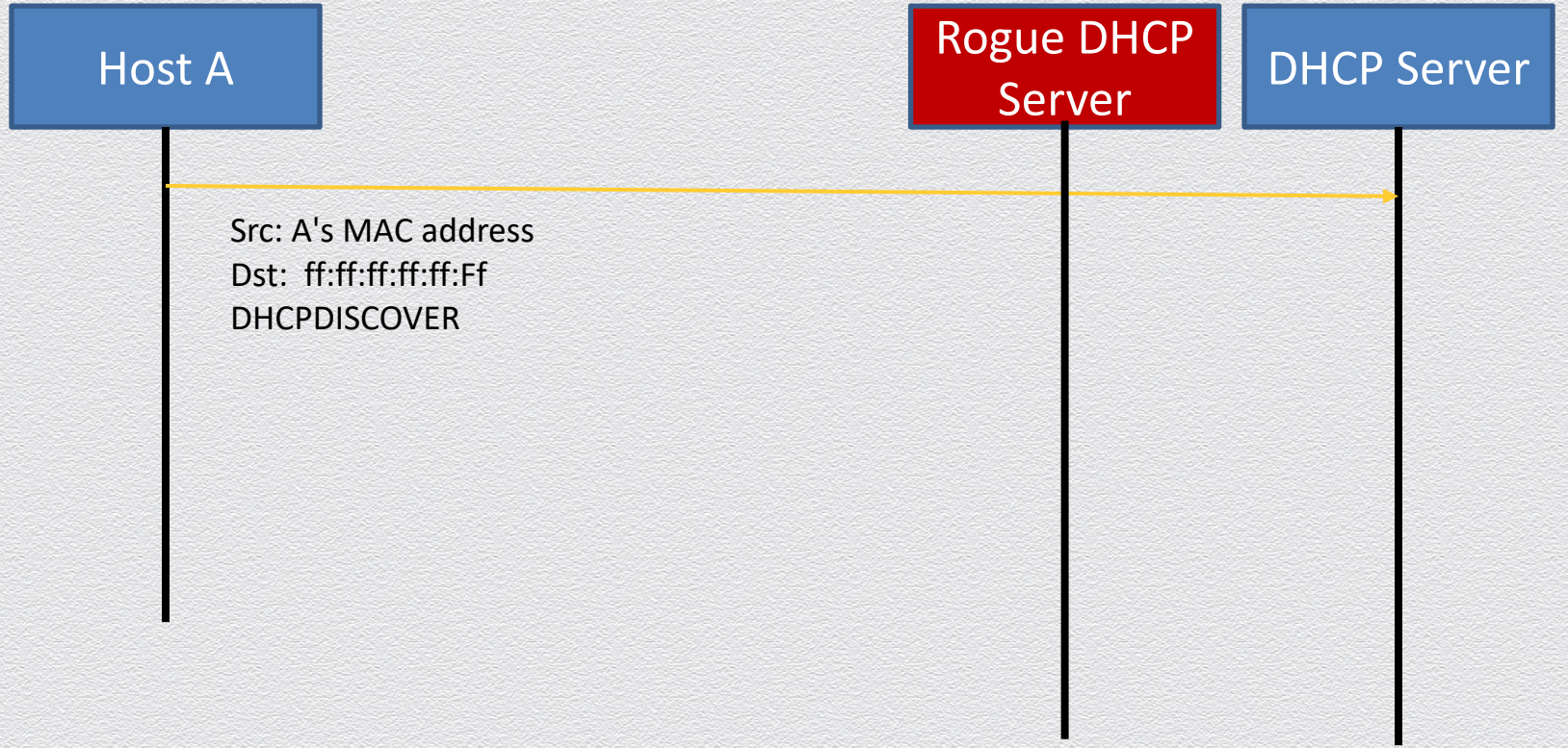


Problems with DHCP?

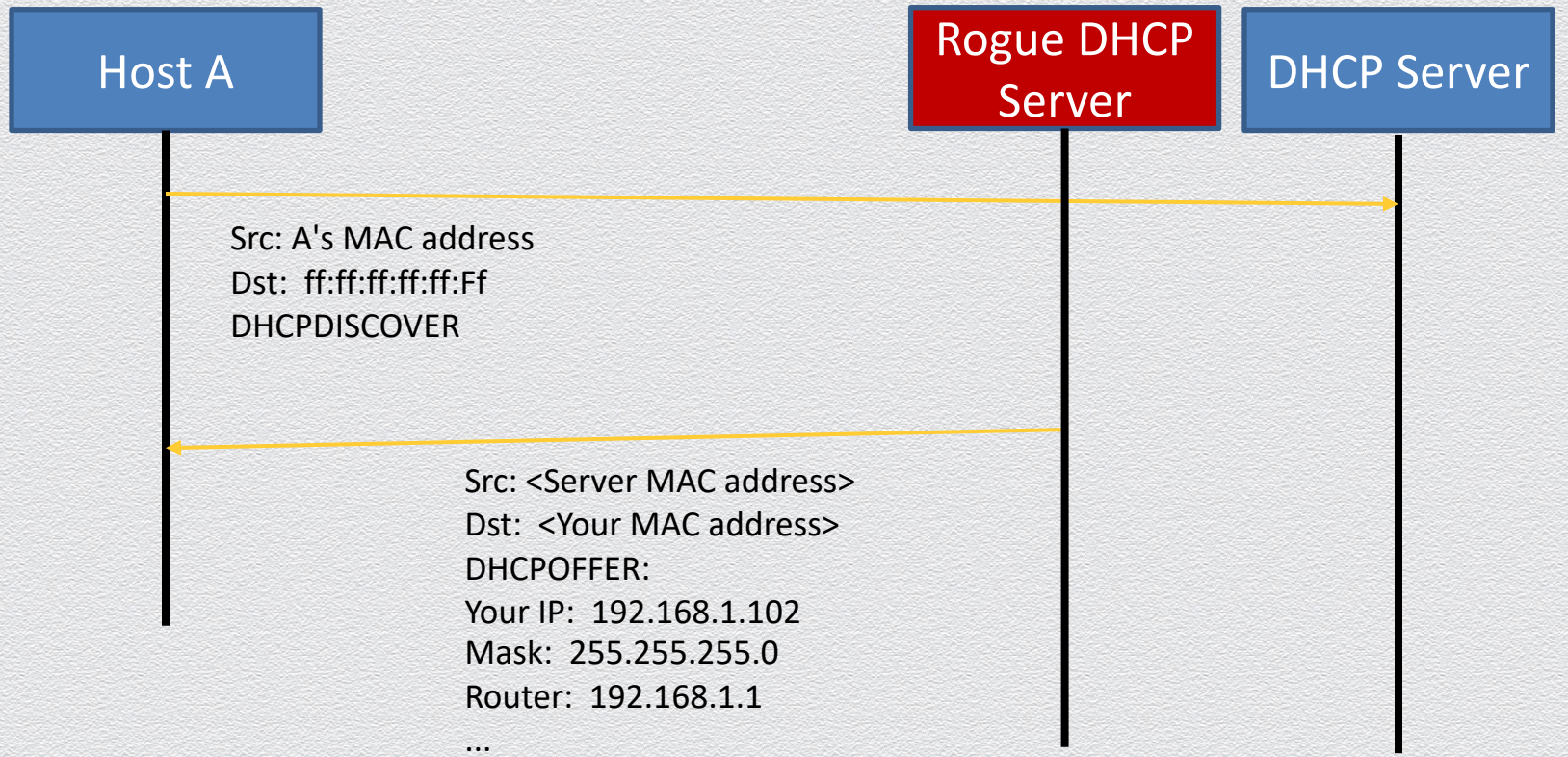
What happens if a random host decides to be a DHCP server?

- Race condition!
- If an attacker can make an offer more quickly than the server, then it can assign a host's IP settings
- Would be detected by the real DHCP server, though (why?)

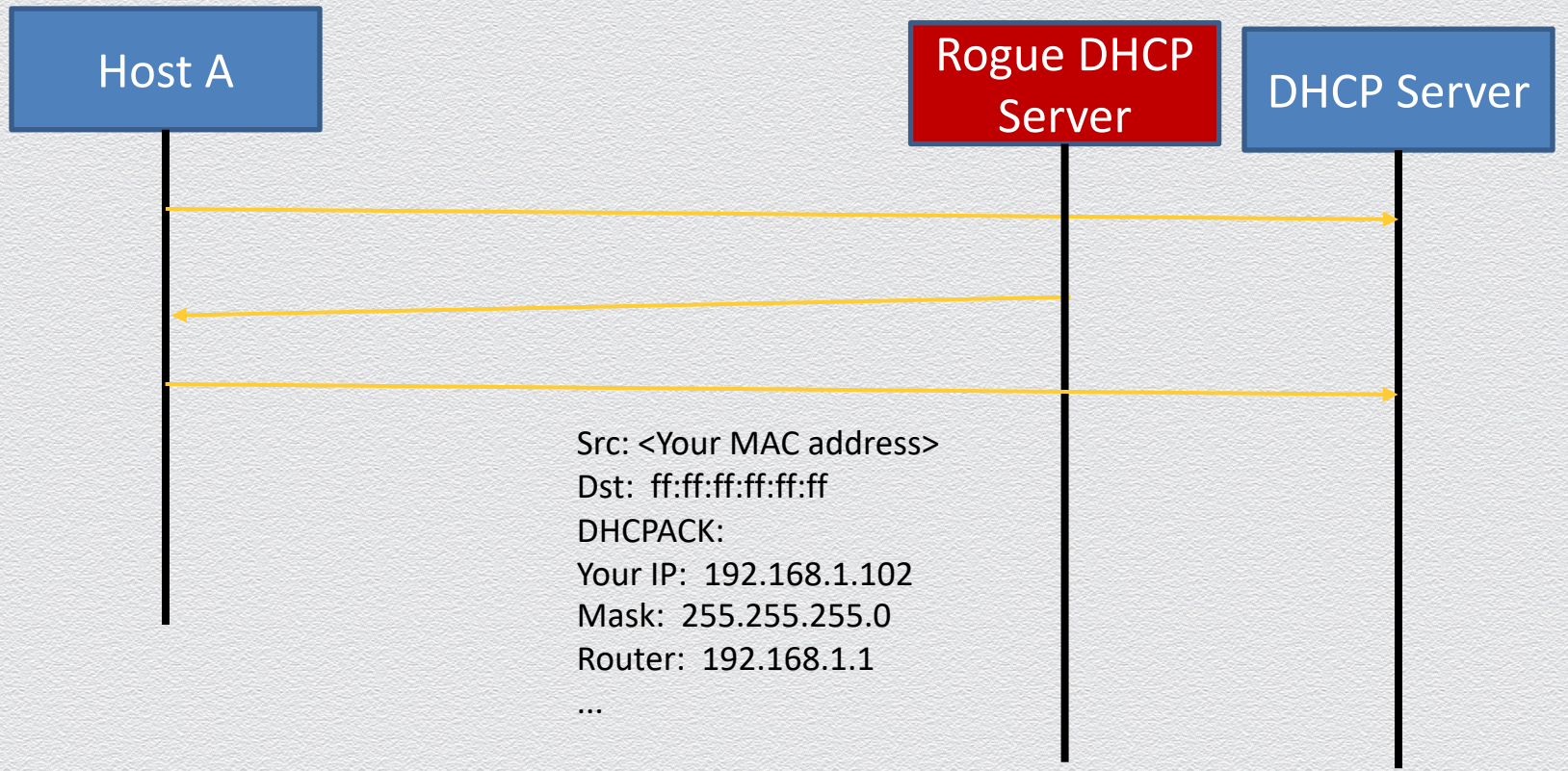
DHCP Spoofing



DHCP Spoofing



DHCP Spoofing



How to defend?

Initial DHCP messages are broadcast, so real server will see the rogue server's response

Can detect the attack!

Why use broadcast?

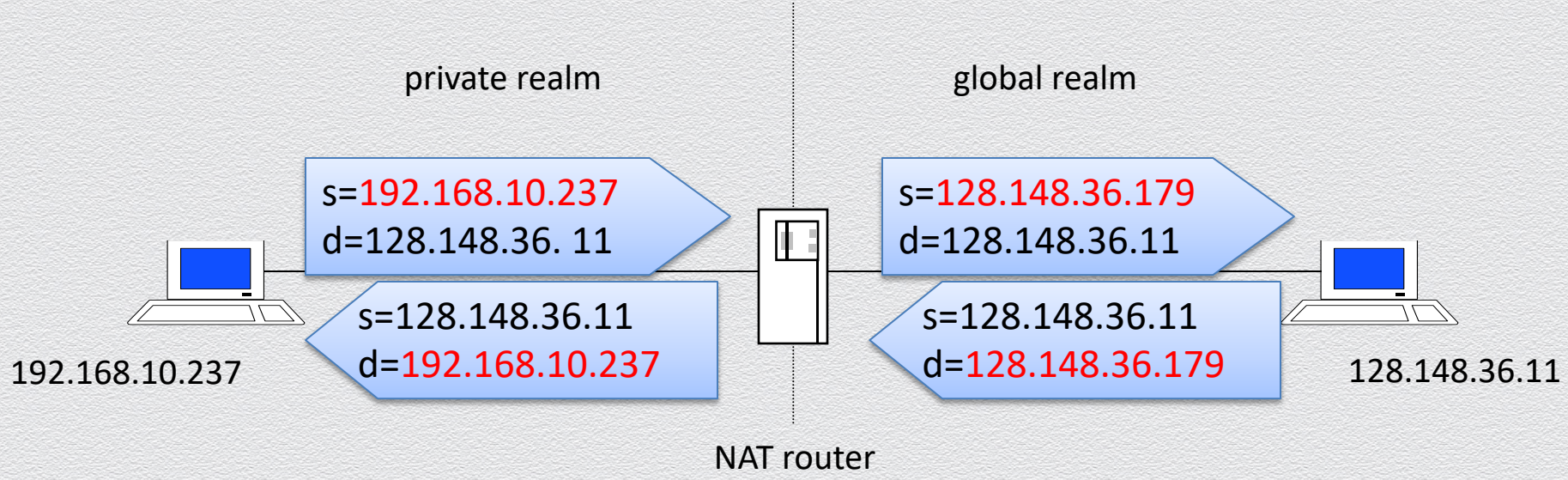
Allows multiple, redundant DHCP servers without extra coordination

Network Address Translation

- Introduced in the early 90s to alleviate IPv4 address space congestion
- Relies on translating addresses in an internal network, to an external address that is used for communication to and from the outside world
- NAT is usually implemented by placing a router in between the internal private network and the public network
- Saves IP address space since not every terminal needs a globally unique IP address, only an organizationally unique one
- While NAT should really be transparent to all high-level services, this is sadly not true because a lot of high-level communication uses things on IP

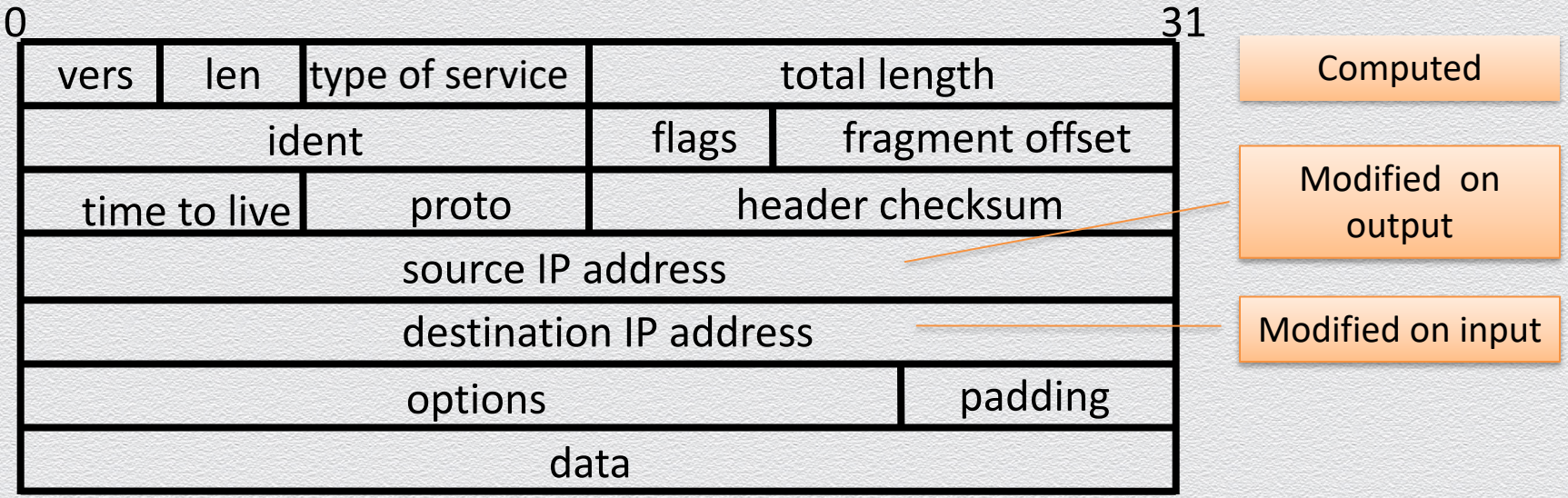
Translation

- Router has a pool of private addresses 192.168.10.0/24



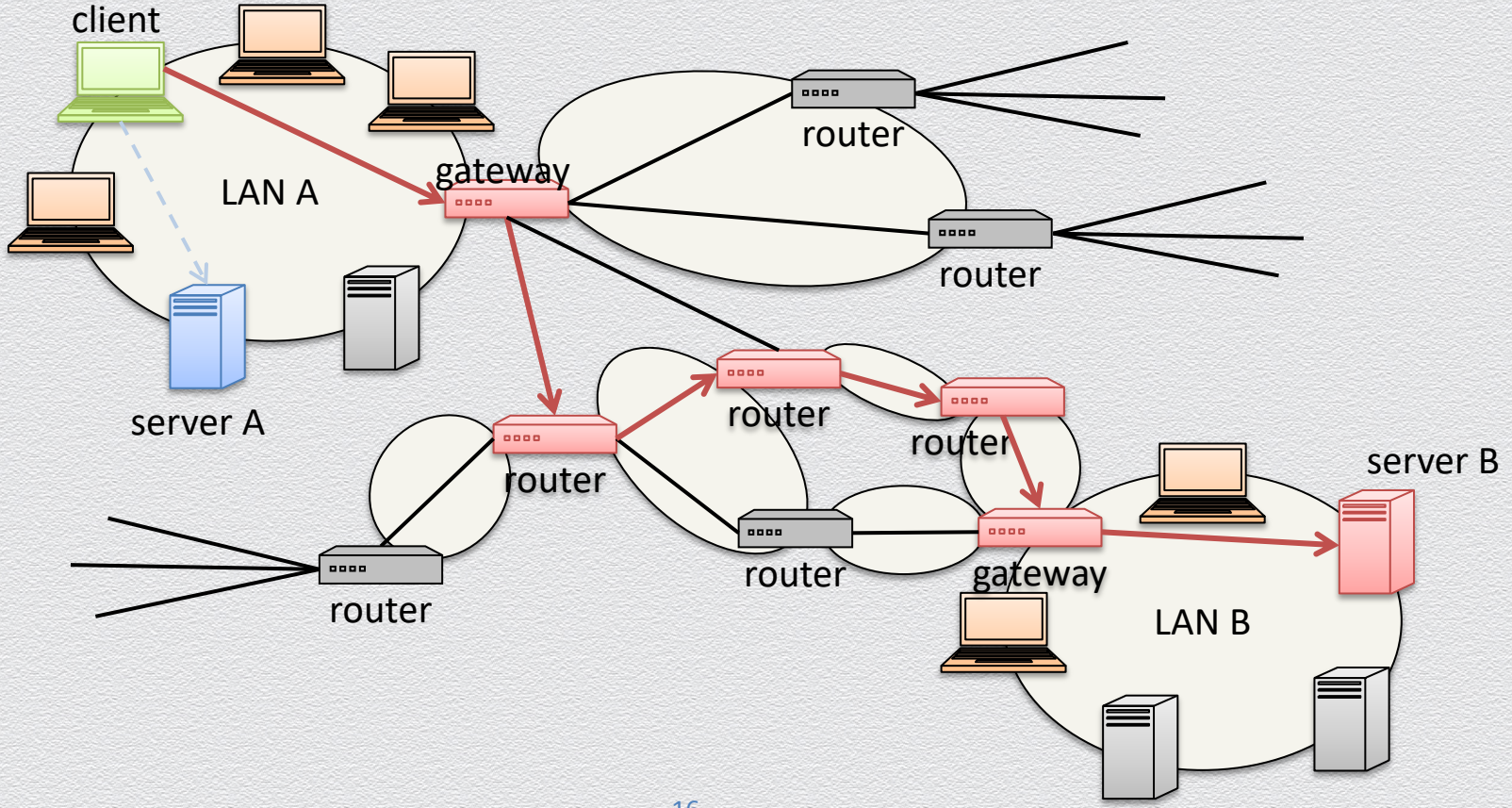
To check external IP: e.g. https://whatismyipaddress.com/

IP packet modifications

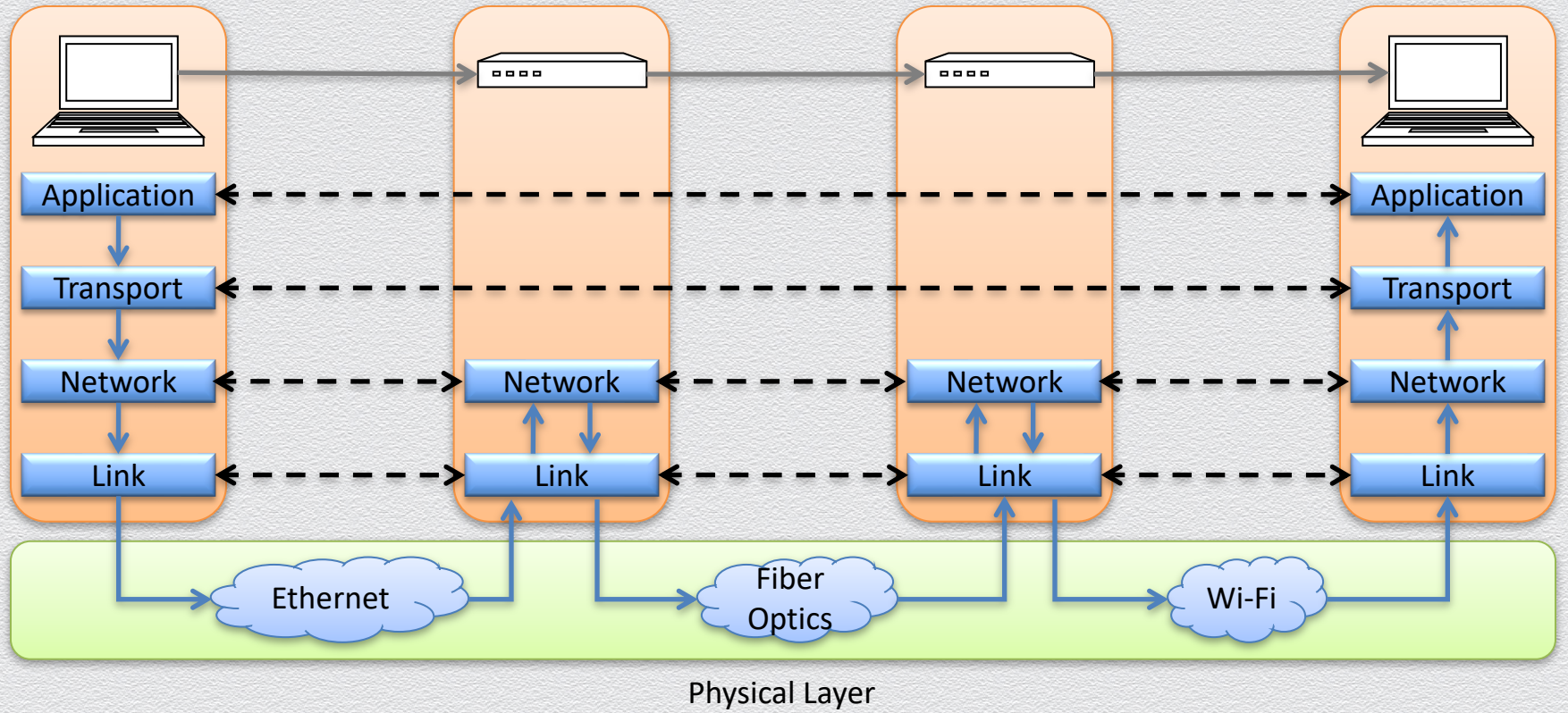


22.2 Routing

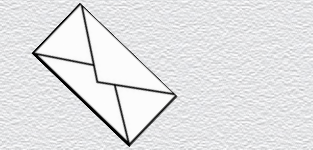
Routing


client
LAN A
gateway
router
router
server A
router
router
router
router
router
gateway
server B
LAN B

Internet Layers: How your computer talks to a website



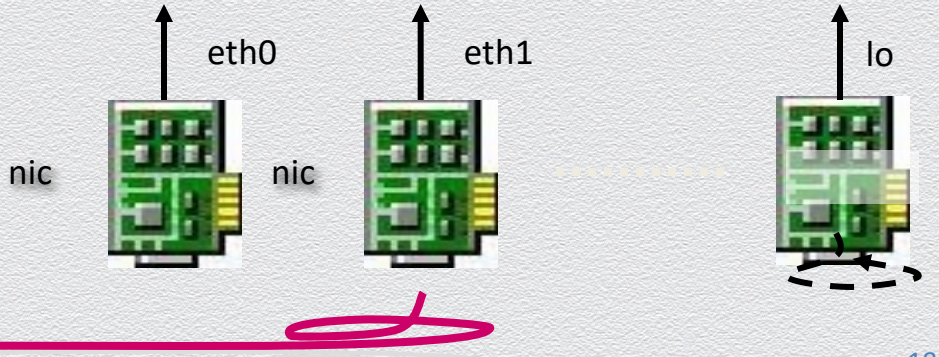
Router



yes

no

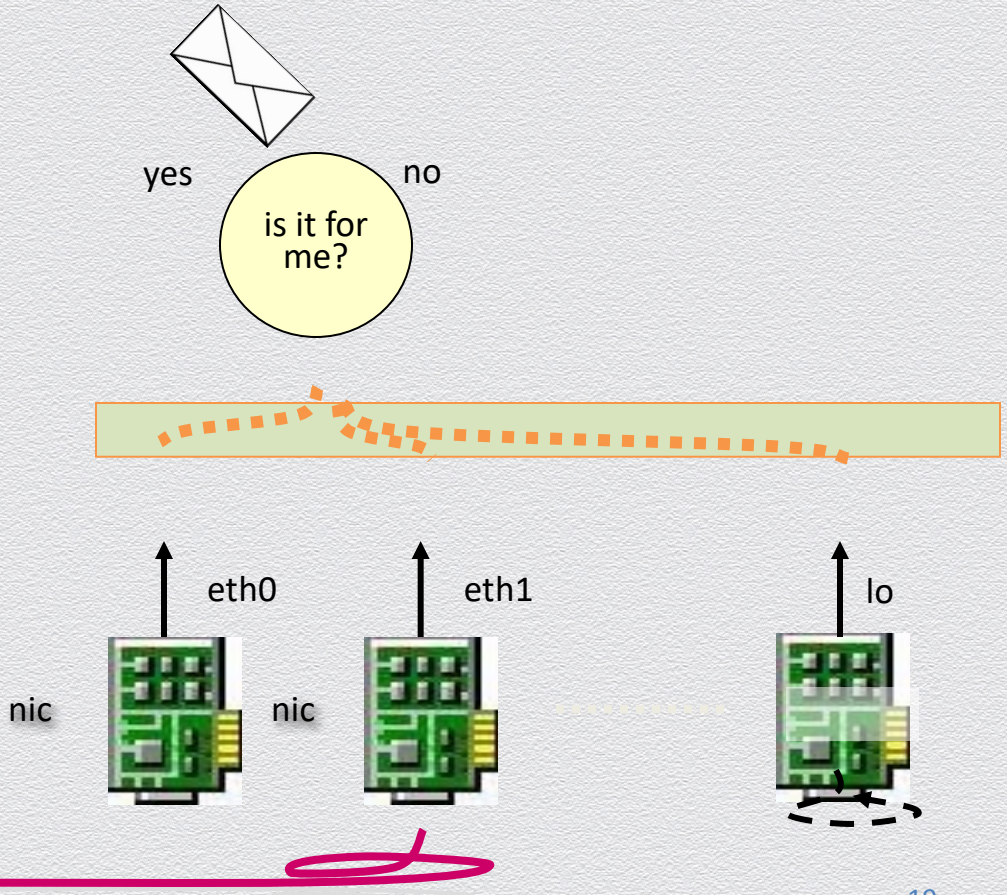
is it for me?



The IP layer decides which interface an outgoing packet has to be forwarded to

- Regular hosts have at least two interfaces, NIC and loopback

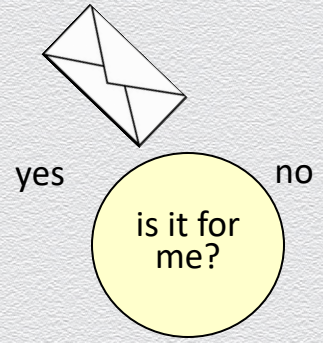
Router (cont.)



A router (gateway, intermediate-system):

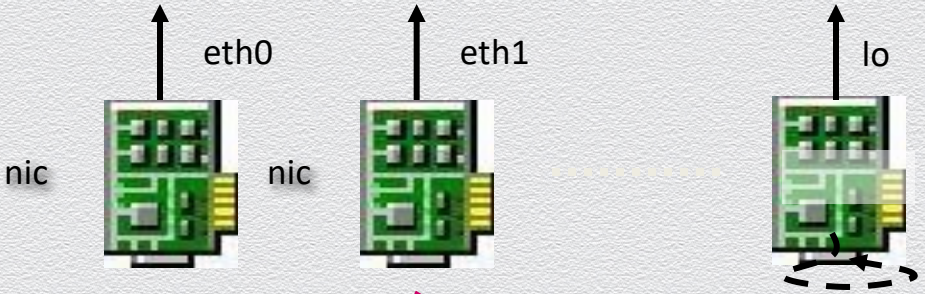
- has more than one network interface card
- feeds incoming ip packets (that are not for the router itself) back in the routing process
  - this operation is called relaying or forwarding

Routing table

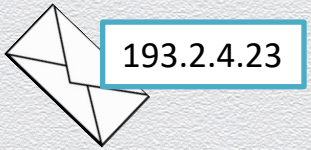


routing table

network	nmask	nexthop	int
200.3.24.0	255.255.255.0	12.0.0.4	eth1
193.2.0.0	255.255.248.0	11.0.0.2	eth0
100.4.5.0	255.240.0.0	11.0.0.3	eth0
0.0.0.0	0.0.0.0	11.0.0.2	eth0



Routing table (cont.)



routing table

network	nmask	nexthop	int
200.3.24.0	255.255.255.0	12.0.0.4	eth1
193.2.0.0	255.255.248.0	11.0.0.2	eth0
100.16.0.0	255.240.0.0	11.0.0.3	eth0
0.0.0.0	0.0.0.0	11.0.0.2	eth0

network	nmask
1100 1000.0000 0011.0001 1000.0000 0000	1111 1111.1111 1111.1111 1111.0000 0000
1100 0001.0000 0010.0000 0000.0000 0000	1111 1111.1111 1111.1111 1000.0000 0000
0110 0100.0001 0000.0000 0000.0000 0000	1111 1111.1111 0000.0000 0000.0000 0000
0000 0000.0000 0000.0000 0000.0000 0000	0000 0000.0000 0000.0000 0000.0000 0000

How to update the routing tables?

Which are the main features that we need?

- Global reachability
- Dynamic & automatic updates
- Fast converging time

Different Routing protocols are available

- Static and manual routing table update is possible but usually not practical

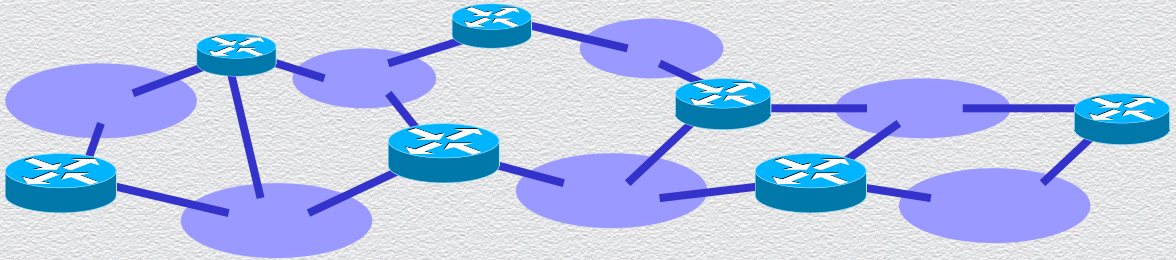
Routing protocols

They fall into two main cathegories:

- link-state routing protocols
  - approach: talk about your neighbors to everyone
  - each router reconstructs the whole network graph and computes a shortest path tree to all destinations
  - examples: IS-IS, Open Shortest Path First (OSPF)
- distance-vector routing protocols
  - approach: talk about everyone with your neighbors
  - update your routing information based on what you hear
  - examples: Routing Information Protocol (RIP), BGP

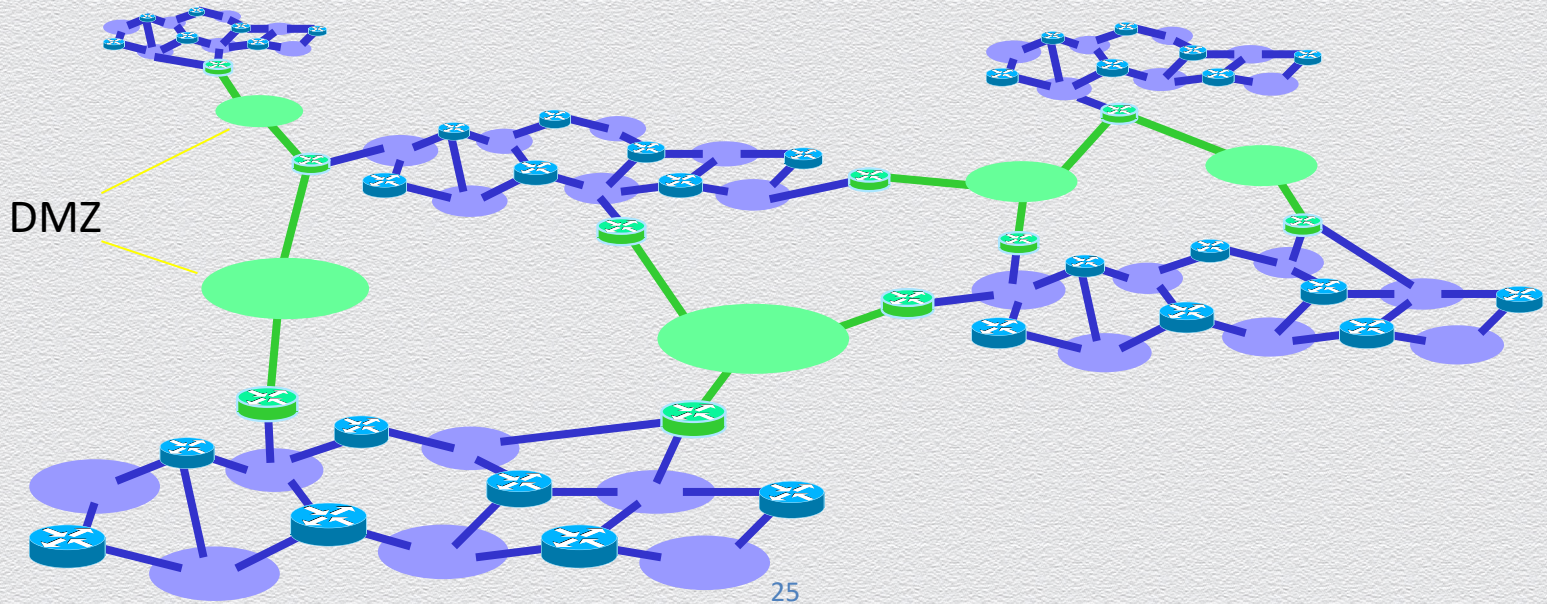
Why interd-omain routing?

- Each organization is a collection of routers and LANs under a single administration
- A routing algorithm may be chosen to automatically update the routing tables



Why inter-domain routing?

- When several organizations join to form the Internet, they have to set up links between them
- The added LANs are called demarkation zones



What about routing tables?

In order to have global connectivity:

- each router must have a routing entry (possibly the default one) that matches the destination address of the packet
- this should be true for packets to be delivered locally as well as for packets to be delivered to remote LANs

Border Gateway Protocol (BGP)

The routing protocol that makes the Internet work

- A path vector protocol (similar to a distance vector)

Used by

- customers connected to an Internet Service Provider (ISP) or several ISPs
- transit providers
- ISPs that exchange traffic at an Internet eXchange Point (IXP) or Neutral Access Point (NAP)
- customers with very large networks

Autonomous System

Autonomous systems (ASes) are the cornerstones of BGP

- Used to uniquely identify networks with a common routing policy
- Usually under single ownership, trust and administrative control

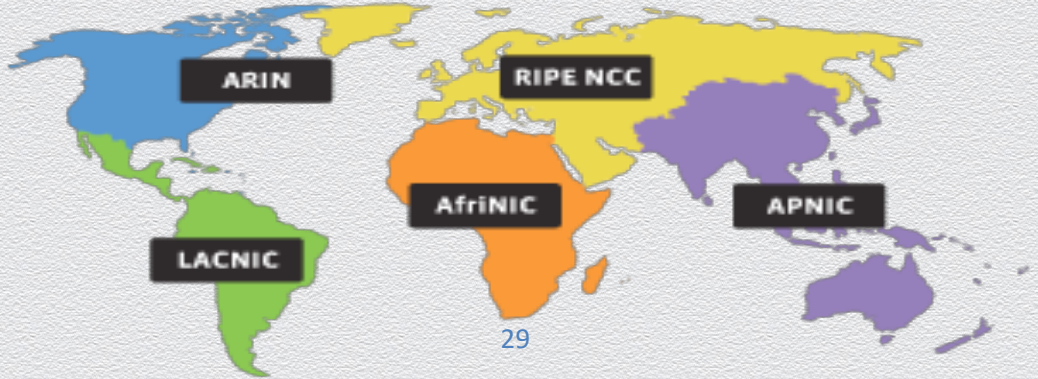
Each AS is identified by an autonomous system number (ASN)

- 32 bit integer
- two ranges
  - 0-65535 (original 16-bit range)
  - 65536-4294967295 (32-bit range - RFC4893)

Autonomous System Number

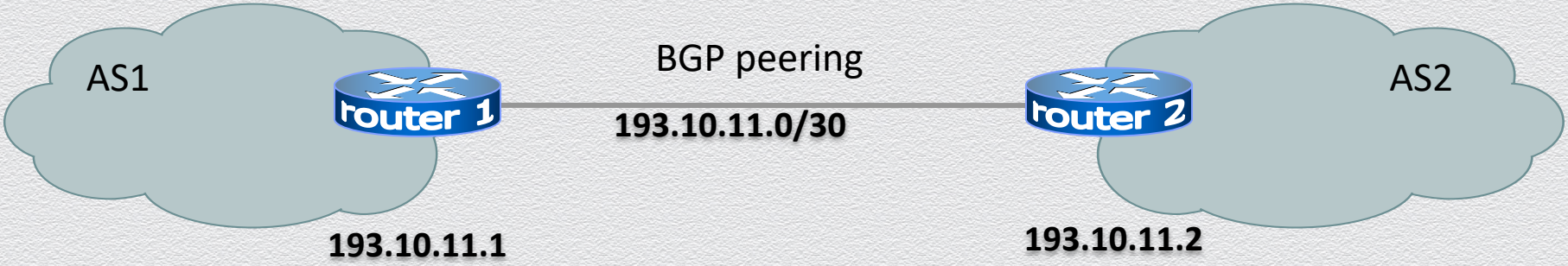
- you may ask an ASN to:
  - global ASN - to your regional internet registry (RIR): ripe, arin, apnic, etc.
  - private ASN - to your upstream ISP
- see also:

  www.iana.org/assignments/as-numbers



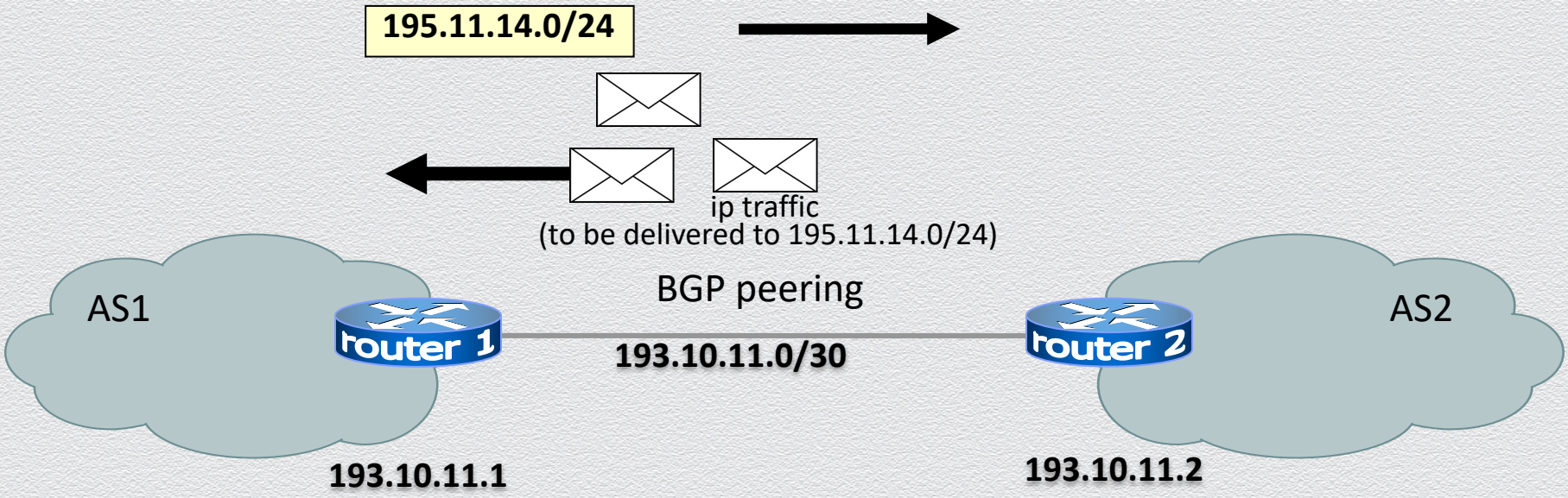
BGP peering

- BGP allows routers to exchange information only if a peering session is up
- a BGP peering is the TCP connection
  (port 179) over which routing information will be exchanged



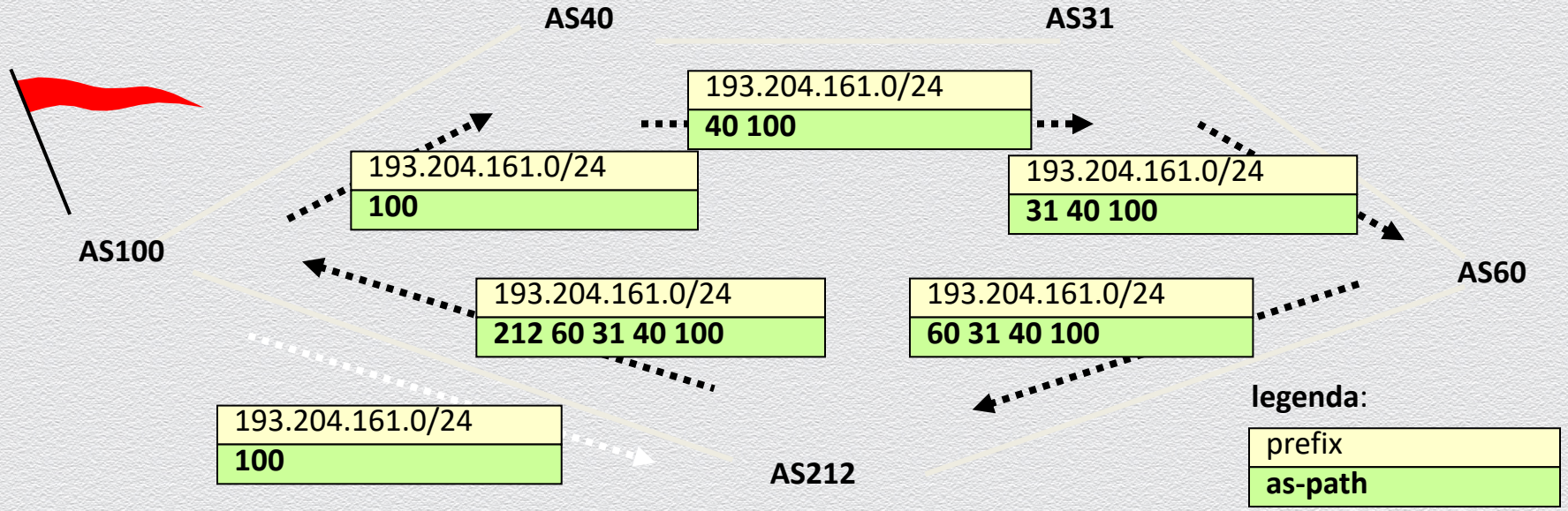
Announcements and traffic flows

- BGP allows a router to offer connectivity to another router
- “offering connectivity” means “promising the delivery to a specific destination”

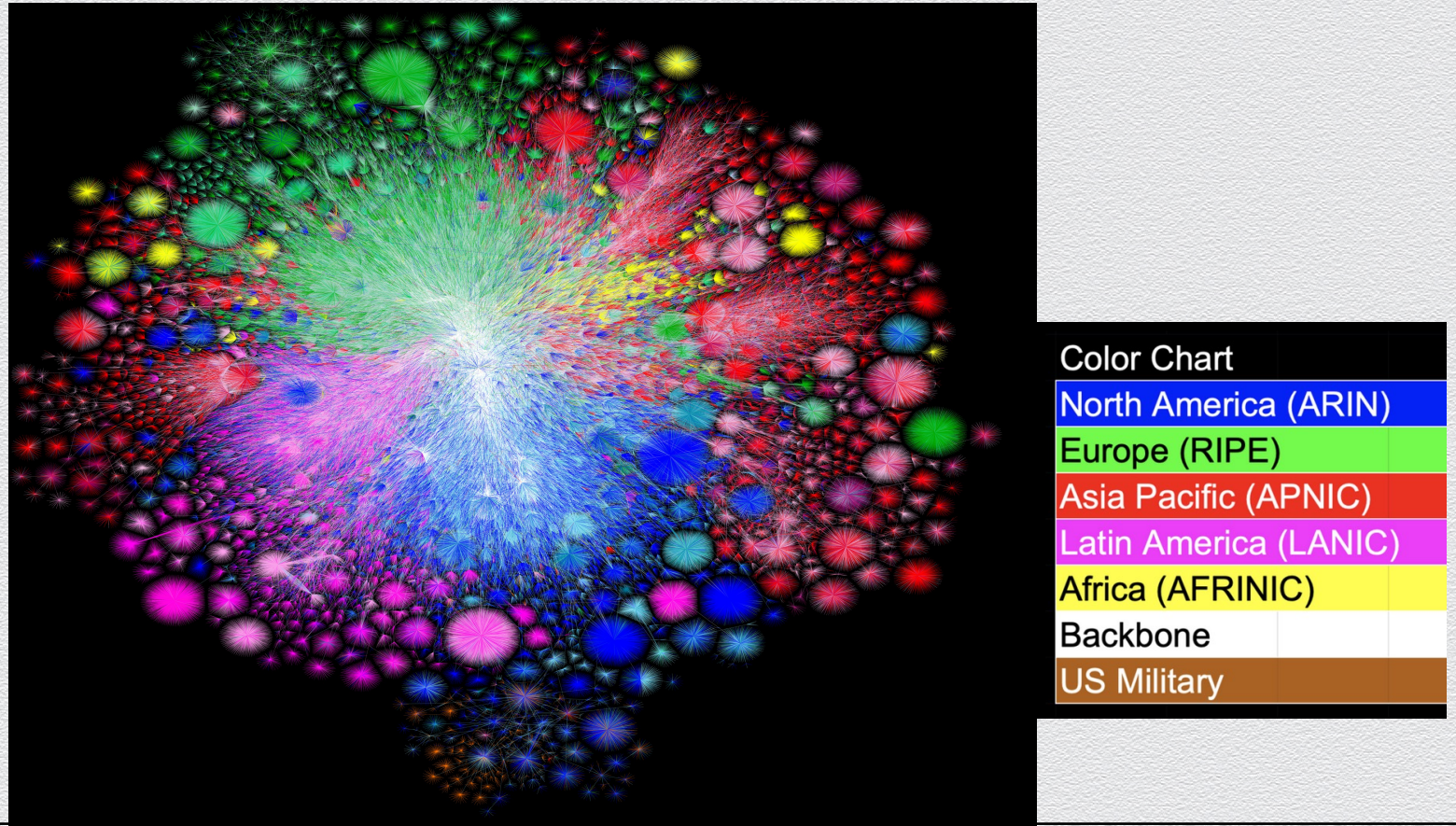


Attributes: AS-path

- BGP allows a router to offer connectivity to another router
- “offering connectivity” means “promising the delivery to a specific destination”



Map of the Internet (2021 via BGP, OPTE project)



BGP vulnerabilities

- In the original version BGP has no security mechanisms:
  - No encryption: Eavesdropping
  - No timestamp: Replaying
  - No signature: Hijacking
  - Selective dropping
- Possible attacks:
  - Injecting false information into the global routing database
  - Reroute traffic to perform a Man-in-the-Middle (MITM) attack
  - Trying to create a Denial of Service (DoS) like a black hole in the network

A big incident

- February 2008
  - Pakistan Telecom (PT) would like to block YouTube access from Pakistan
  - PT falsely informed that through this company there was the most directed way to reach Youtube
- Soon over 2/3 of the Internet was unable to reach YouTube for a couple of hours
- A routing problem…

22.3 Transport Layer

The Transport Layer

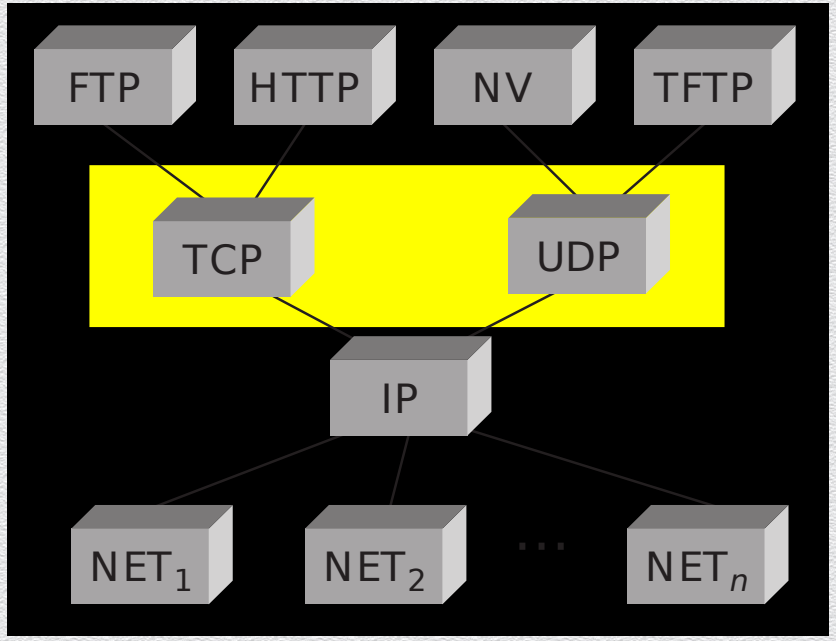
Network layer: moving data between hosts

Transport layer: Abstraction for getting data data to different *applications* on a host

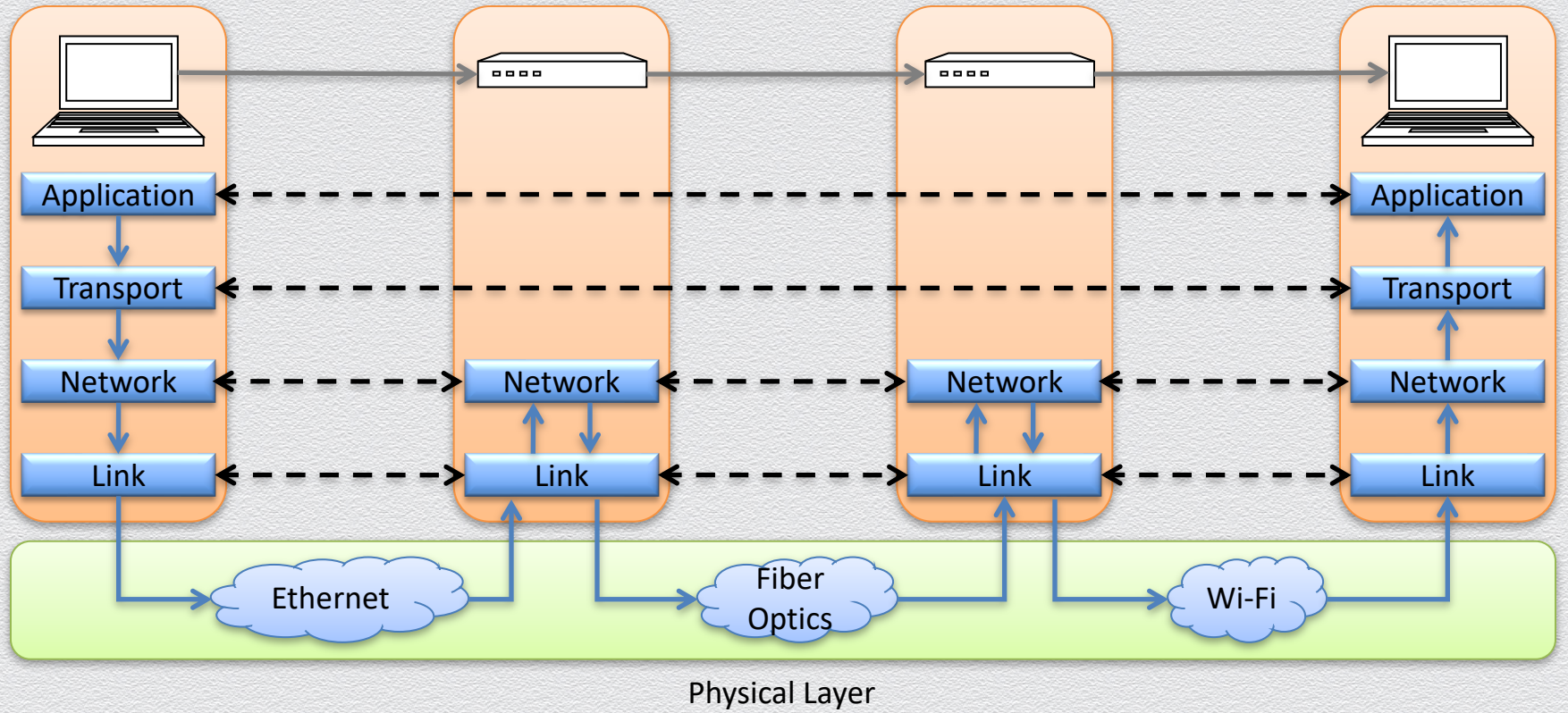
- Multiplexing multiple connections at the same IP with port numbers
- Series of packets => stream of data/messages
- May provide: reliable data delivery

Two key protocols: TCP, UDP

Transport Layer



Internet Layers: How your computer talks to a website



What's a port number?

- 16-bit unsigned number, 0-65535
- Ports define a communication *endpoint*, usually a process/service on a host
- OS keeps track of which ports map to which applications

Port numbering

- port < 1024: "Well known port numbers"
- port >= 20000: "ephemeral ports", for general app. use

Some common ports

Port	Service
20, 21	File Transfer Protocol (FTP)
22	Secure Shell (SSH)
23	Telnet (pre-SSH remote login)
25	SMTP (Email)
53	Domain Name System (DNS)
67, 68	DHCP
80	HTTP (Web traffic)
443	HTTPS (Secure HTTP over TLS)

How ports work

Two modes:

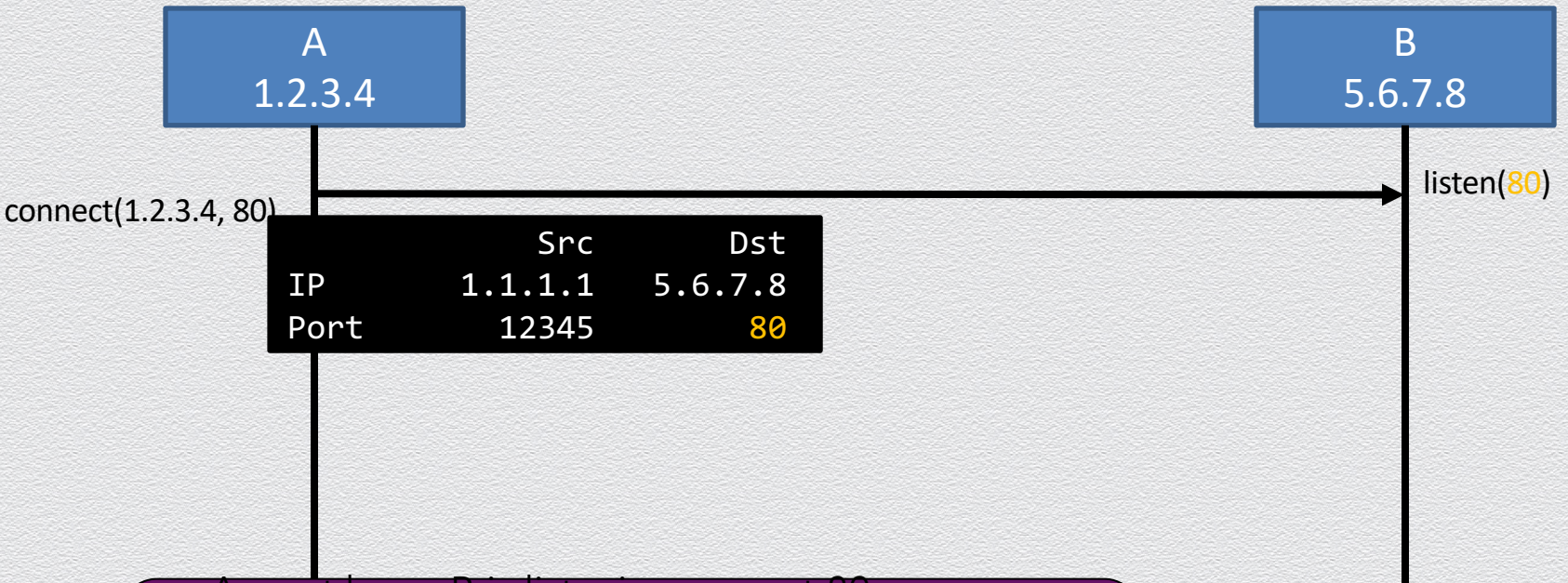
- Applications "listen on" or "bind to" a port to wait for new connections

  => Example: webserver listens on port 80 or 443

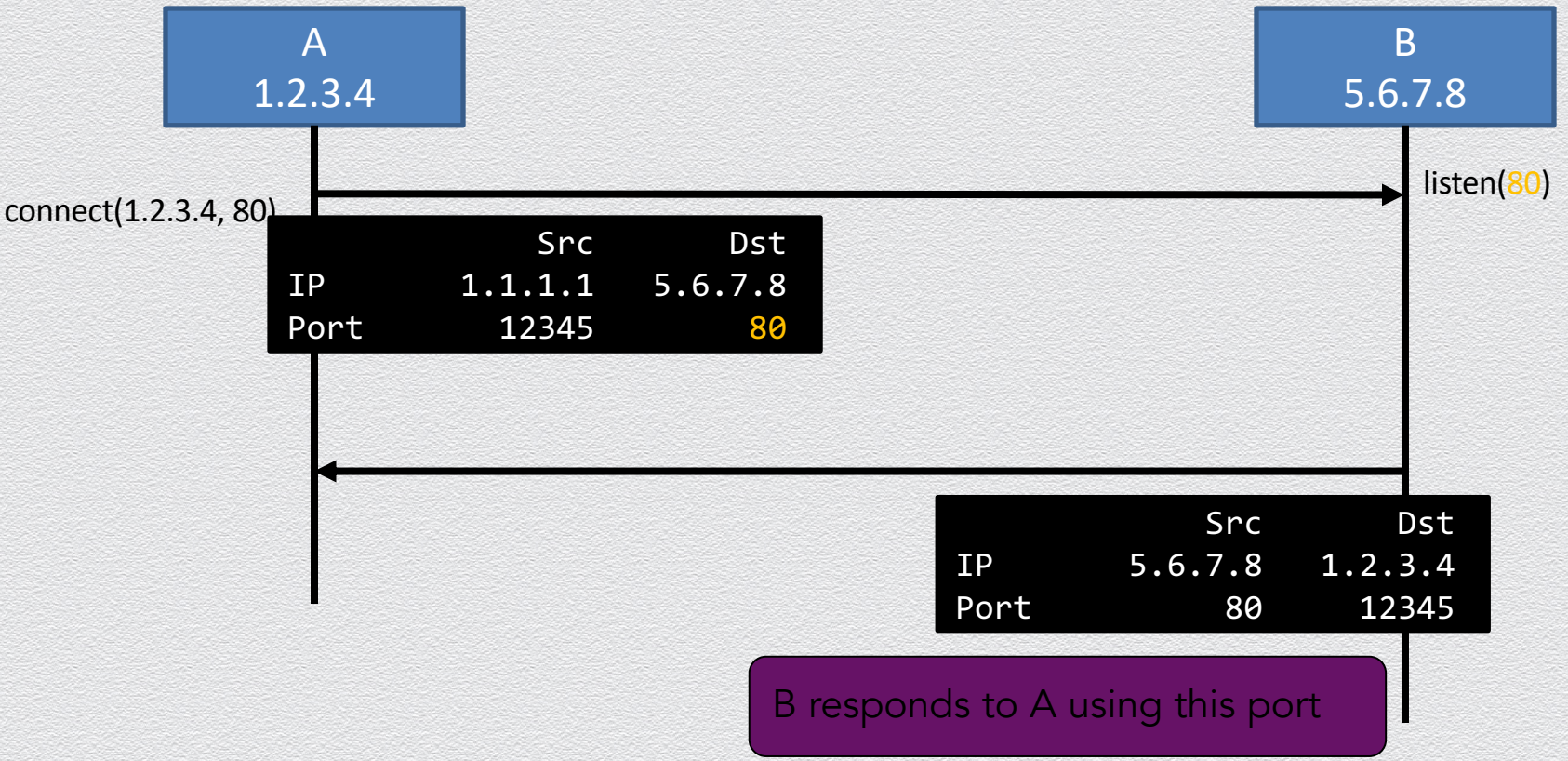
- Hosts make connections to a particular IP and port

  => Example: client connects to <webserver IP>, port 80 or 443

  (eg. 1.2.3.4:80)



- A must know B is listening on port 80
  => "well known numbers"!
- When connecting, A's OS picks random source port (eg. 12345), used for its side of connection


A
1.2.3.4
B
5.6.7.8
listen(80)
connect(1.2.3.4, 80)
Src
Dst
IP
1.1.1.1
5.6.7.8
Port
12345
80
Src
Dst
IP
5.6.7.8
1.2.3.4
Port
80
12345
B responds to A using this port

Sockets

OS keeps track of which application uses which port

Two types:

- Listening ports
- Connections between two hosts (src/dst port)

Socket: OS abstraction for a network connection, like a file descriptor

Table maps: port => socket

Netstat

```
deemer@vesta ~/Development % netstat -an
Active Internet connections (including servers)
Proto Recv-Q Send-Q  Local Address          Foreign Address        (state)
tcp4       0      0  10.3.146.161.51094     104.16.248.249.443     ESTABLISHED
tcp4       0      0  10.3.146.161.51076     172.66.43.67.443       ESTABLISHED
tcp6       0      0  2620:6e:6000:900.51074 2606:4700:3108::.443   ESTABLISHED
tcp4       0      0  10.3.146.161.51065     35.82.230.35.443       ESTABLISHED
tcp4       0      0  10.3.146.161.51055     162.159.136.234.443    ESTABLISHED
tcp4       0      0  10.3.146.161.51038     17.57.147.5.5223       ESTABLISHED
tcp6       0      0  *.22                   *.*                    LISTEN
tcp4       0      0  *.51036                *.*                    LISTEN
tcp4       0      0  127.0.0.1.9999         *.*                    LISTEN
```

netstat –an: Show all connections

netstat –lnp: Show listening ports + applications using them (as root)

Macosx: lsof -nP -iTCP -sTCP:LISTEN

Why do we care?

Ports define what services are exposed to the network

- Open port: can send data to application (reconnaissance, attacks, ...)

- OS and network hardware can monitor port numbers
  - Make decisions on how to filter/monitor traffic

Transport Layer

- The transport layer supports one or more of the following features
  A. Reliable data transfer (resending of dropped packets)
  B. In-order delivery of segments of file or media stream
  C. Congestion control (request longer/shorter segments)
  D. Ability to distinguish multiple applications on same host via ports (16-bit numbers)
- The main transport layer protocols are
  - UDP (supports B, D)
  - TCP (supports A, B, C, D)

User Datagram Protocol (UDP)

- Stateless, unreliable transport-layer protocol
- Can distinguish multiple concurrent applications on a single host
- No delivery guarantees or acknowledgments
  - Efficient
  - Suitable for audio/video streaming and voice calls
  - Unsuitable for file transmission and text messaging

Transmission Control Protocol (TCP)

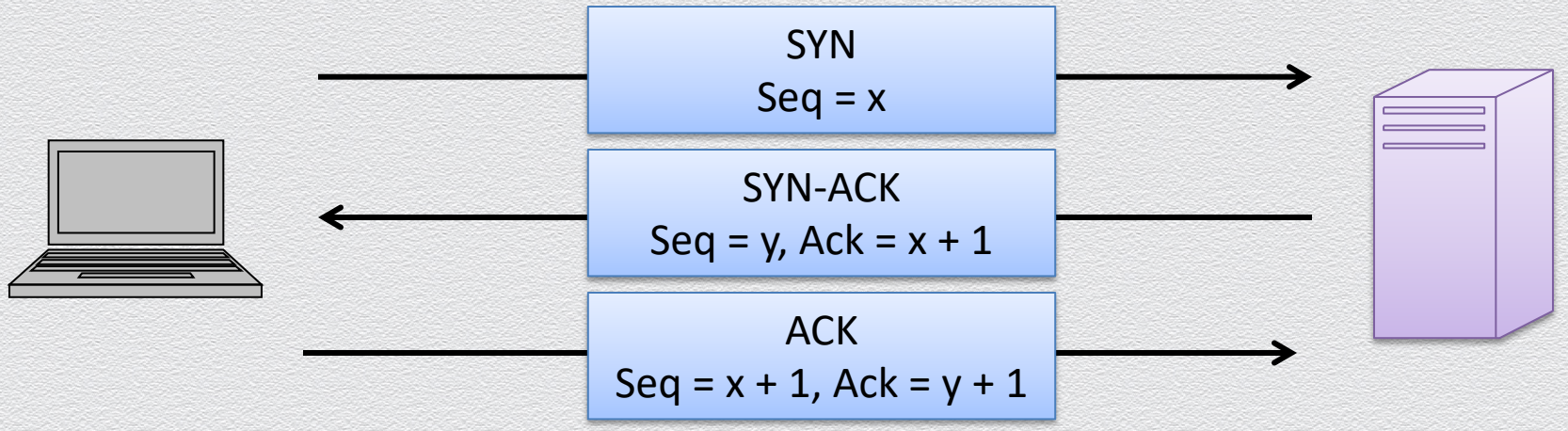
- Stateful protocol for reliable data transfer, in-order delivery of messages and ability to distinguish multiple applications on same host
  - HTTP and SSH are built on top of TCP
- TCP packages a data stream it into segments transported by IP
  - Order maintained by marking each packet with sequence number
  - Every time TCP receives a packet, it sends out an ACK to indicate successful receipt of the packet
- TCP generally checks data transmitted by comparing a checksum of the data with a checksum encoded in the packet

TCP Packet Format

Bit Offset	0-3	4-7	8-15	16-18	19-31
0	Source Port			Destination Port	
32	Sequence Number				
64	Acknowledgment Number				
96	Offset	Reserved	Flags	Window Size	
128	Checksum			Urgent Pointer	
160	Options				
>= 160	Payload				

Establishing TCP Connections

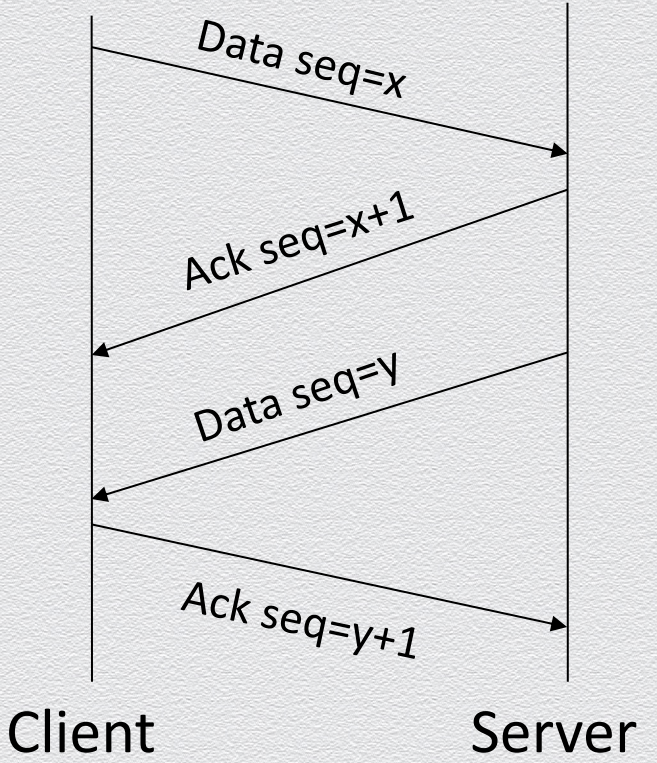
- TCP connections are established through a three-way handshake
- The server generally is a passive listener, waiting for a connection request
- The client requests a connection by sending out a SYN packet
- The server responds by sending a SYN/ACK packet, acknowledging the connection
- The client responds by sending an ACK to the server, thus establishing connection

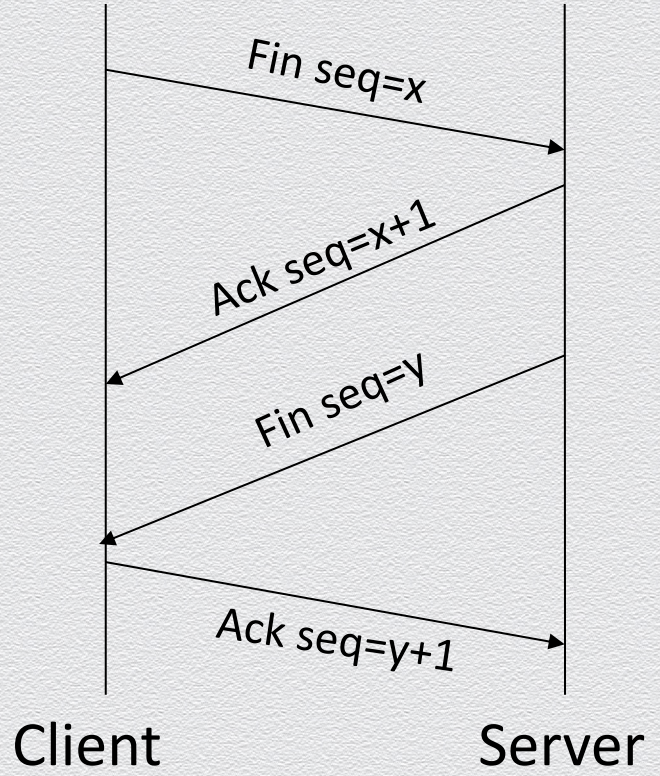


TCP Data Transfer

- The three way handshake initializes sequence numbers for the request and response data streams
- The TCP header includes a 16 bit checksum of the payload and parts of the header, including source and destination
- Acknowledgment or lack thereof is used by TCP to keep track of network congestion and control flow
- TCP connections are cleanly terminated with a 4-way handshake
  - The client which wishes to terminate the connection sends a FIN message to the other client
  - The other client responds by sending an ACK
  - The other client sends a FIN
  - The original client now sends an ACK, and the connection is terminated

TCP Data Transfer and Teardown





Port scanning

What can we learn if we just start connecting to well-known ports?

- Can discover things about the network
- Can learn about vulnerabilities

Large-scale port scanning

- Can reveal lots of open/insecure systems!
- Examples:
  - shodan.io
  - VNC roulette
  - Open webcam viewers..
  - …
- Also: penetration testing/vulnerability scanning

Disclaimer

- Network scanning is easy to detect
- Unless you are the owner of the network, it’s seen as malicious activity
- If you scan the whole Internet, the whole Internet will get mad at you (unless done very politely)

Question (2)

Eve is once again up to no good. She decides to modify the payload of a TCP packet that Alice sends to Bob by randomly flipping a bit. Would Bob be able to detect this?

A. Yes, since most likely the checksum will not match

B. Yes, since the packet will be totally corrupted

C. No, since there are no security features in TCP

D. No, since it is computationally infeasible

Answer (2)

Eve is once again up to no good. She decides to modify the payload of a TCP packet that Alice sends to Bob by randomly flipping a bit. Would Bob be able to detect this?

A. **Yes, since most likely the checksum will not match**

B. Yes, since the packet will be totally corrupted

C. No, since there are no security features in TCP

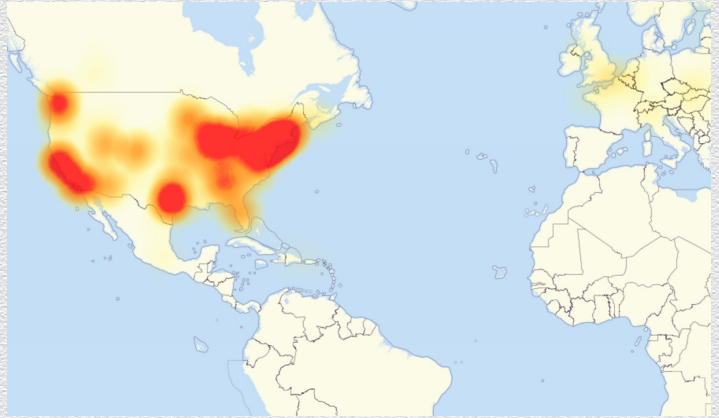
D. No, since it is computationally infeasible

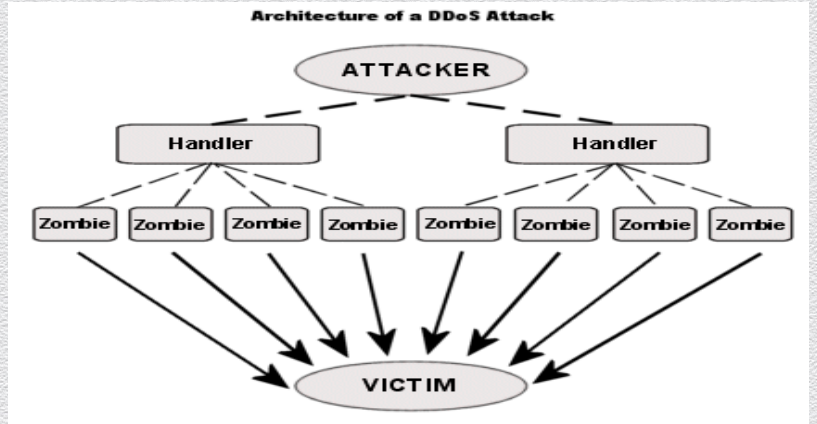
23.4 DNS security

It’s unfair! – I had no class but couldn’t watch my Netflix series!

On October 21, 2016, a large-scale cyber was launched

- it affected globally the entire Internet but particularly hit U.S. east coast
- during most of the day, no one could access a long list of major Internet platforms and services, e.g., Netflix, CNN, Airbnb, PayPal, Zillow, …
- this was a **Distributed Denial-of-Service (DDoS)** attack





DoS: A threat (mainly) against availability

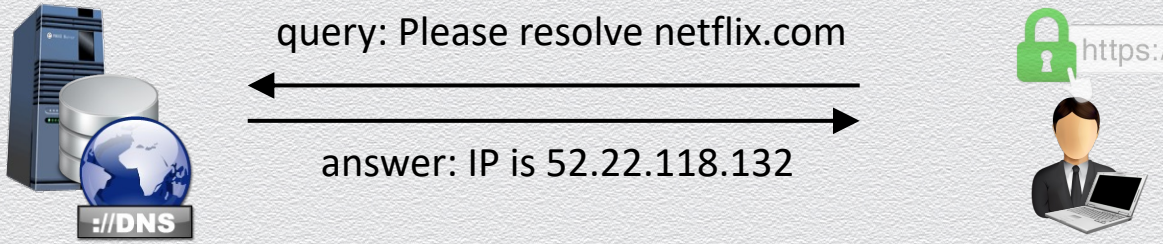
Which main security property does a Denial-of-Service (DoS) attack attempt to defeat?

- availability; a user is denied access to authorized services or data
  - availability is concerned with preserving authorized access to assets
  - a DoS attack aims against this property; its name itself implies its main goal
- integrity & confidentiality; services or data are modified or accessed by an unauthorized user
  - elements of a DoS attack may include breaching the integrity or confidentiality of a system
  - but the end goal is disruption of a service or data flow; not the manipulation, fabrication or interception of data and services

The Domain Name Service (DNS) protocol

Resolving domain names to IP addresses

- when you type a URL in your Web browser, its IP address must be found
  - e.g., domain name “netflix.com” has IP address “52.22.118.132”
  - larger websites have multiple IP responses for redundancy to distributing load
- at the heart of Internet addressing is a protocol called DNS
  - a database translating Internet names to addresses



DNS name resolution is a critical asset – a target itself!

What main security properties must be preserved in such an important service?

- all properties in CIA triad are relevant!
- resolving domain names to IP addresses is a service that
  - must critically be available during all times – availability
    - or else your browser does not know how to connect to Netflix…
  - must critically be trustworthy – integrity
    - or else connections to malicious sites may occur (e.g., DNS-spoofing attacks)
  - must also protect database entries that are not queried – confidentiality
    - or else an attacker may find out about the structure of a target organization (e.g., zone-enumeration attacks)

Recursive name resolution: hierarchical search

Search is performed recursively and hierarchically across different type of DNS resolvers

- application-level (e.g., Web browser), OS-level (e.g., stub resolver): locally managed
- recursive DNS servers: query other resolvers and cache recent results

DNS entries:
<netflix.com, 52.22.118.132>

primary

subset of cached queried entries
(or information of other resolvers)

secondary

netflix.com

52.22.118.132
(or "non-existent")

locally cached IP addresses
(at Web browser and OS)

Recursive name resolution: hierarchical search

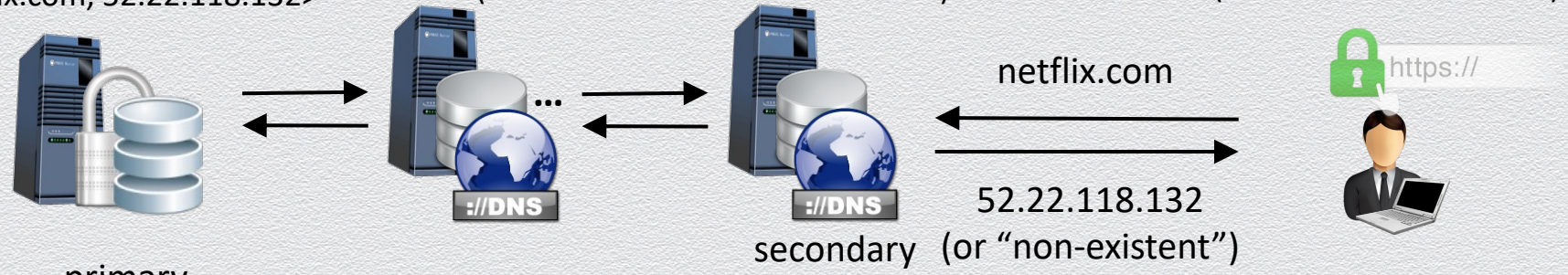
Search is performed recursively and hierarchically across different type of DNS resolvers

- application-level (e.g., Web browser), OS-level (e.g., stub resolver): locally managed
- recursive DNS servers: query other resolvers and cache recent results
- root name servers: refer to appropriate TLD (top-level domain) server
- TLD servers: control TLD zones such as .com, .org, .net, etc.

DNS entries:
<netflix.com, 52.22.118.132>

subset of cached queried entries
(or information of other resolvers)

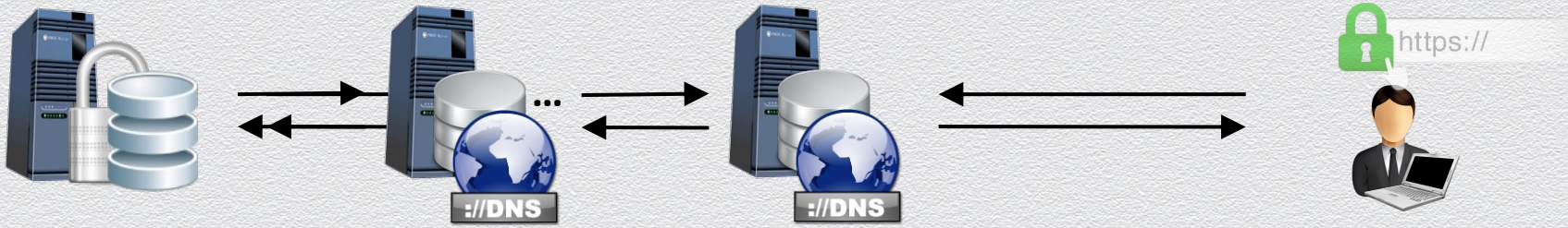
locally cached IP addresses
(at Web browser and OS)



Recursive name resolution: flexibility

Infrastructure allows for different configurations

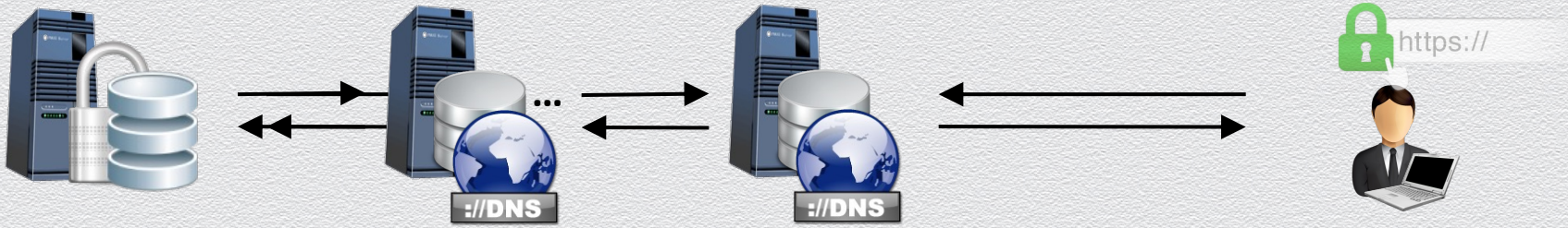
- authoritative-only servers: answer queries on zones they are responsible for
  - fast resolution, no forwarding, no cache
- caching / forwarding DNS servers: answer queries on any public domain name
  - recursive search / request forwarding, caching for speed, first-hop resolvers
- master / slaves DNS servers: authoritative servers replicating DNS data of their domains
- public / private DNS servers: control access to protected resources within an organization



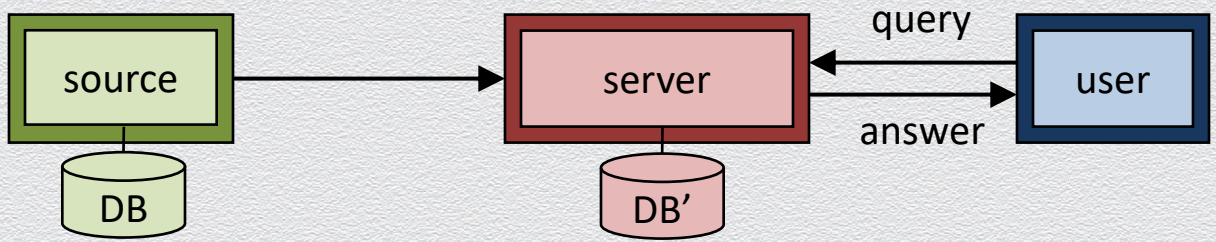
Recursive name resolution: benefits

Why DNS uses non-authoritative name servers (that is, recursive resolution)?

- for more scalability & locality
  - high query loads can saturate the response capacity of primary servers
  - secondary do not have to store large volumes of DNS entries
  - cached recently queried domain names speed up searches due to locality of queries
- for added security / locality / scalability alone – not quite
  - e.g., non-authoritative name servers are untrusted and thus possibly compromised

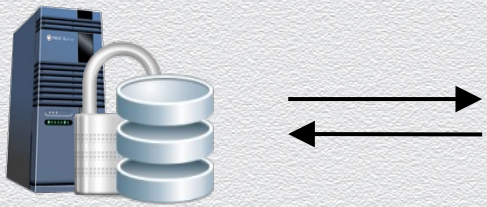


DNS as a (distributed) database-as-a-service



DNS entries:
<netflix.com, 52.22.118.132>

subset of cached queried entries
(or information of other resolvers)



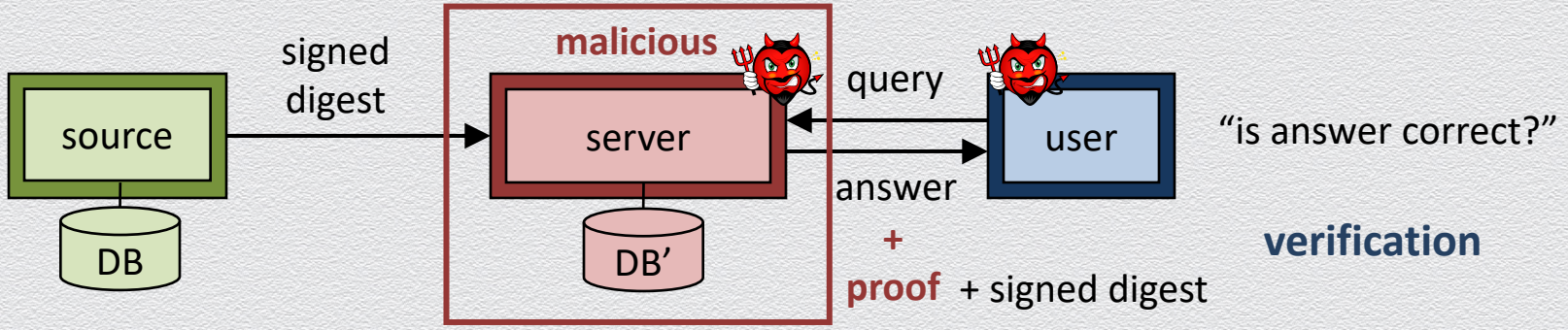
"primary"
name server

"secondary"
name server

please resolve netflix.com

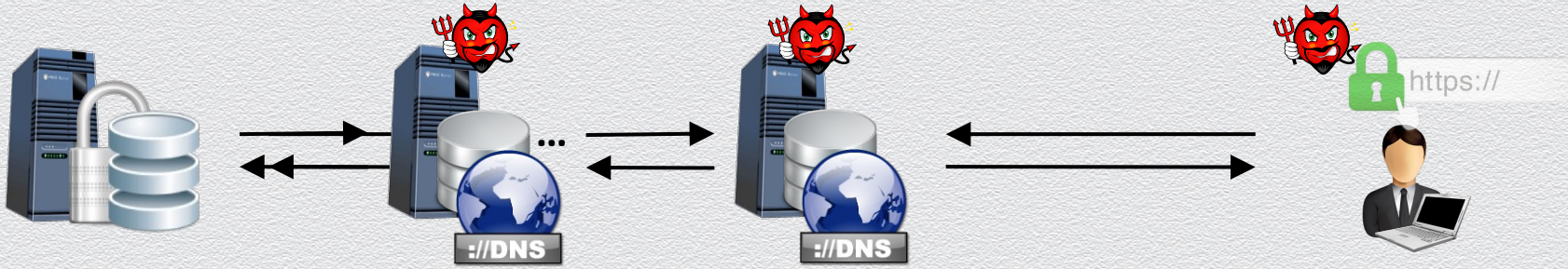
IP is 52.22.118.132
(or "aWa2j3netflix.com
is a non-existent domain")

A critical asset prone to attacks



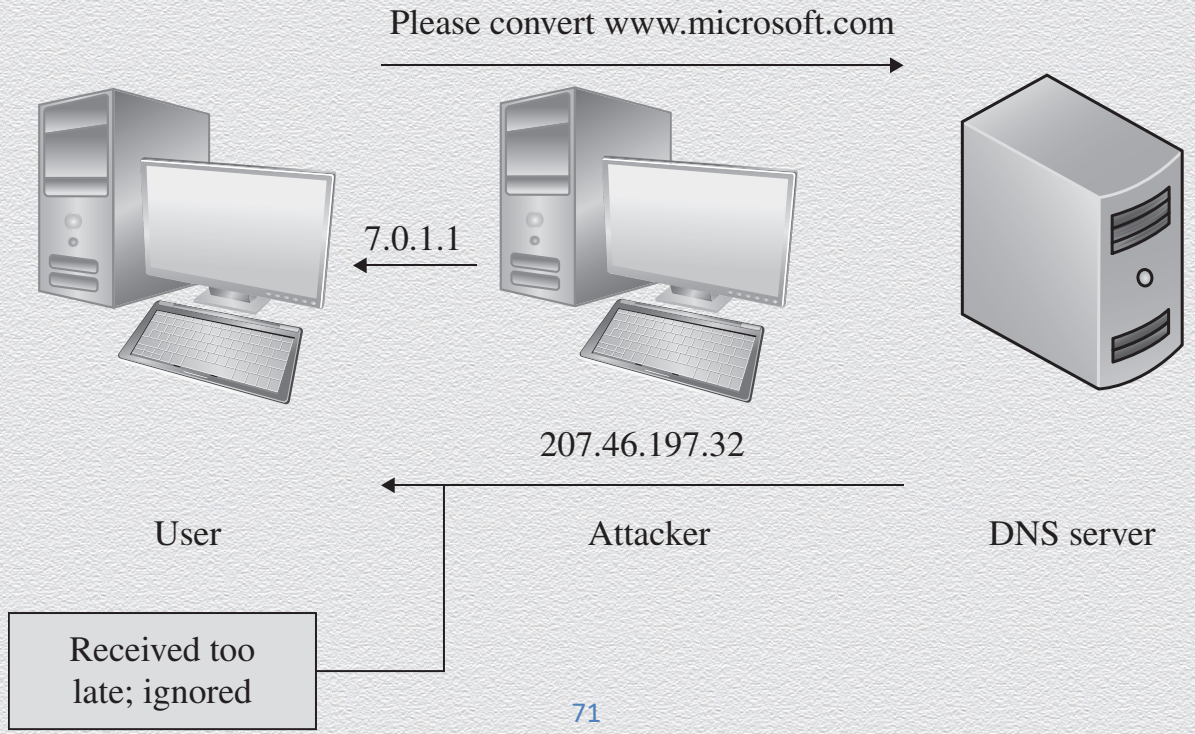
integrity

availability / confidentiality



DNS spoofing (or cache poisoning)

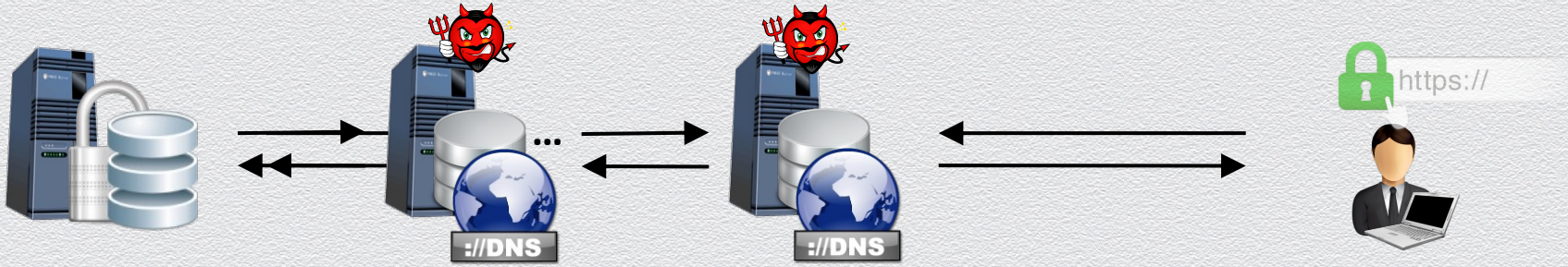
The attacker acts as the DNS server in order to redirect the user to malicious sites



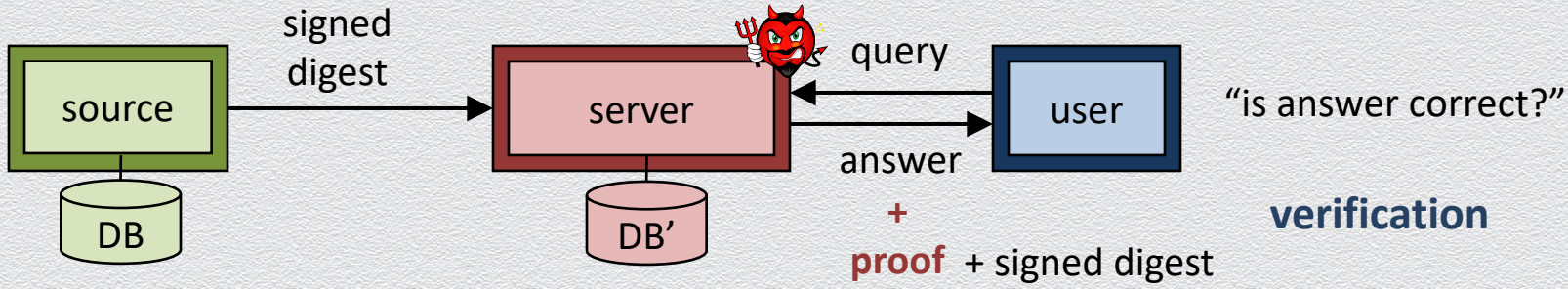
DNSSEC & NSEC

Security extension of DNS protocol to protect integrity of DNS data

- correct resolution, origin authentication, authenticated denial of existence
- specifications made by Internet Engineering Task Force (IETF) via RFCs
  - an RFC (request for comments) is a suggested solution under peer review
- challenges: backward-compatible, simplicity, confidentiality, who signs
  - NSEC (next secure record): extension that provides proofs of denial of existence



DNSSEC & NSEC: core idea



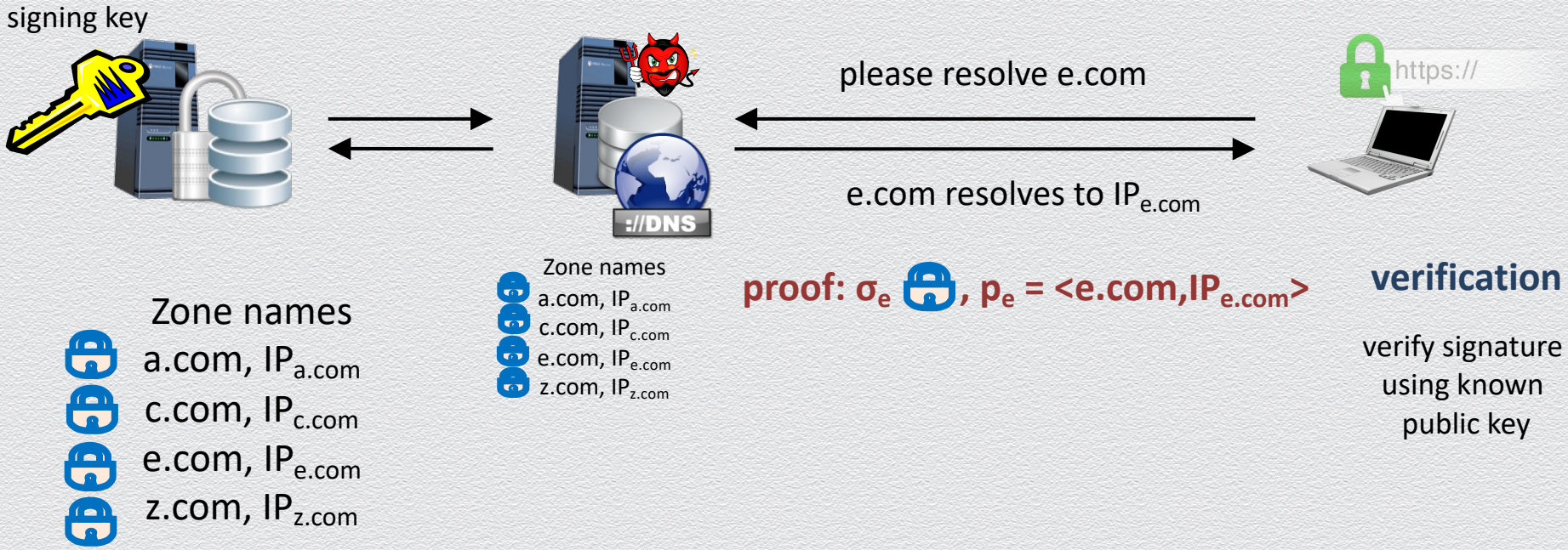
DNSSEC protocol: each DNS entry is pre-signed by primary name server

NSEC protocol:

- domain names are lexicographically ordered and then each pair of neighboring existing domain names is pre-signed by the primary name server
- non-existing names, e.g., aWa2j3netflix.com are proved by providing this pair "containing" missed query name, e.g., <awa.com, awb.com>

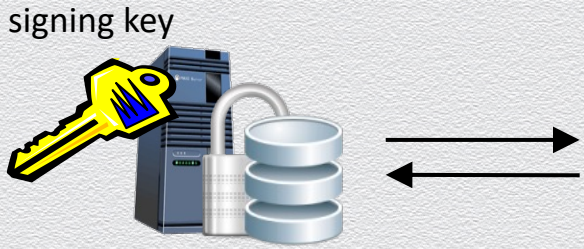
DNSSEC: example

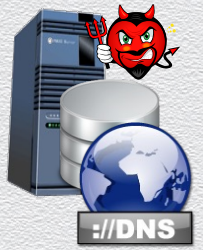
Each entry <domain name, IP address> in the database is individually signed by a primary DNS server and uploaded to secondary DNS servers in signed form



NSEC: example

Additionally, pairs of consecutive (in alphabetical order) domain names are individually signed by a primary DNS server and uploaded to secondary DNS servers in signed form





please resolve b.com

domain name b.com doesn't exist

Zone names

a.com
c.com
e.com
z.com
a.com

σ_1 σ_2 σ_3 σ_4

proof: σ_1, p_1 = <a.com, c.com>

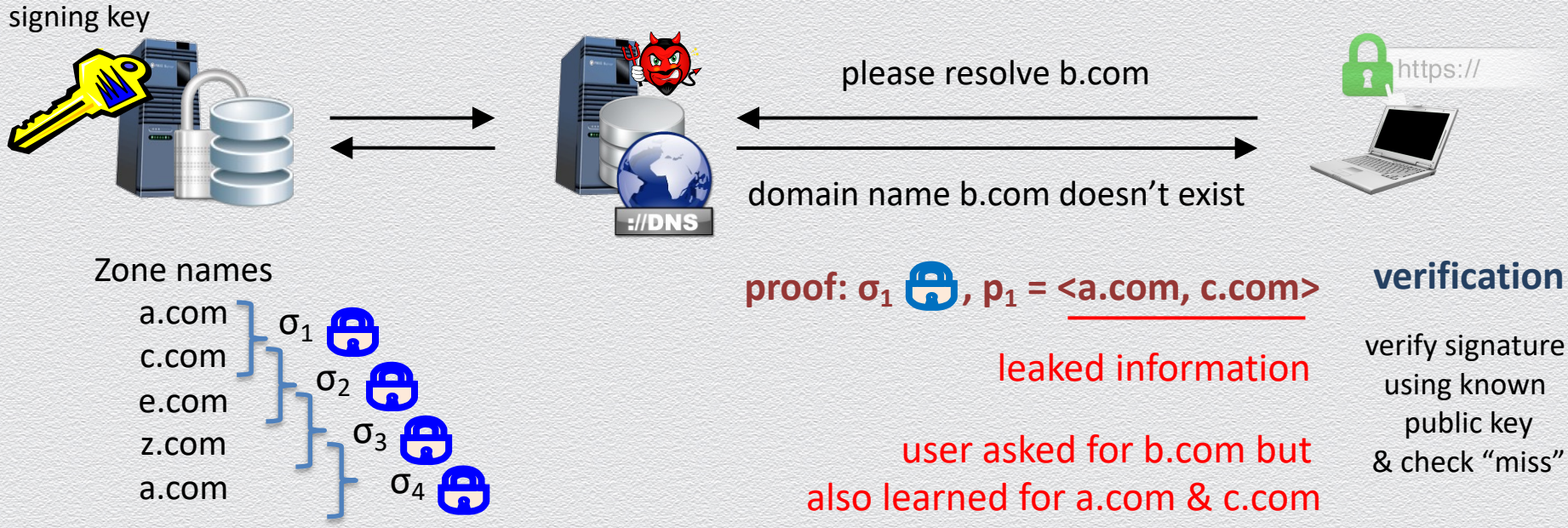
verification

verify signature using known public key & check "miss"

22.4.2 NSEC vulnerability: Protocols NSEC3 & NSEC5

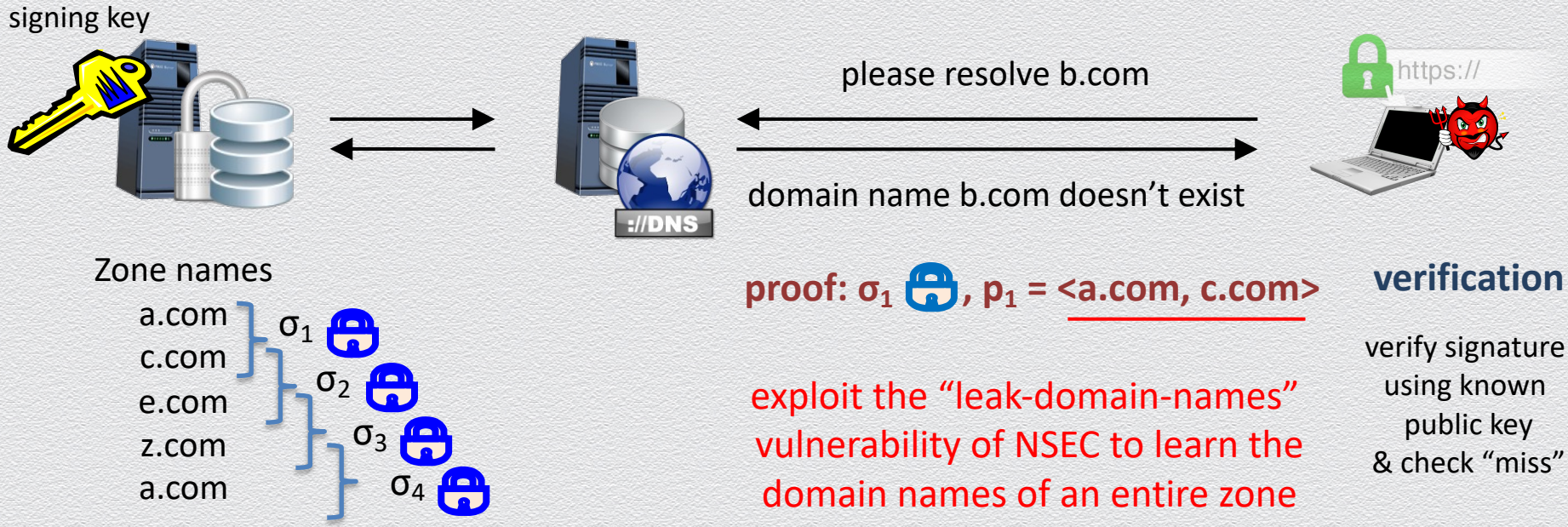
The problem

Proofs of non-existing names leak information about other unknown domain names



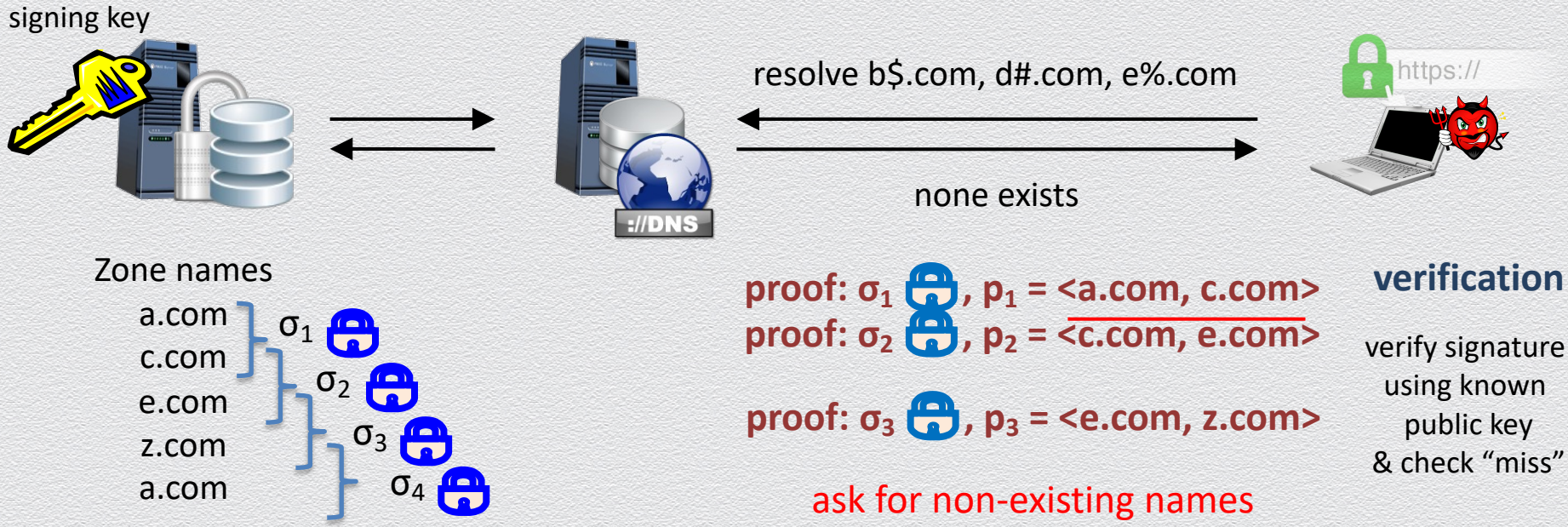
Zone enumeration attack: Main idea

An attacker can simply act as a “querier” to learn target organization’s network structure!



Zone enumeration attack: Example

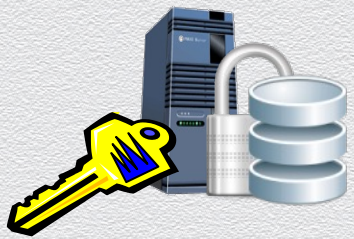
An attacker can simply act as a “querier” to learn target organization’s network structure!



Zone enumeration attack: Result

An attacker can simply act as a “querier” to learn target organization’s network structure!

NSEC3: NSEC in the hash domain



please resolve b.com

b.com is a non-existent domain

proof: σ_3 , p_3 = <dde45,zrit5>

Zone names

a.com
c.com
e.com
z.com

hash h →

a1bb5
23ced
zrit5
dde45

sort →

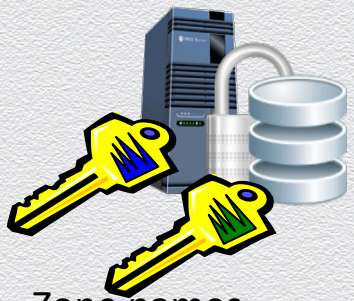
23ced
a1bb5
dde45
zrit5
23ced

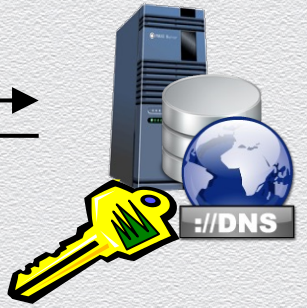
σ_1
σ_2
σ_3
σ_4

asked for b.com but
learned h(e.com) & h(z.com)

h(b.com) = ntwo4
e.g., h is SHA-256

NSEC5: A secure solution





please resolve b.com

b.com is a non-existent domain

proof: σ_3 , p_3 = <dde45,zrit5>

RSA-signature of f(b.com)

Zone names

a.com
c.com
e.com
z.com

hash h’ →

a1bb5
23ced
zrit5
dde45

sort →

23ced
a1bb5
dde45
zrit5
23ced

σ_1
σ_2
σ_3
σ_4

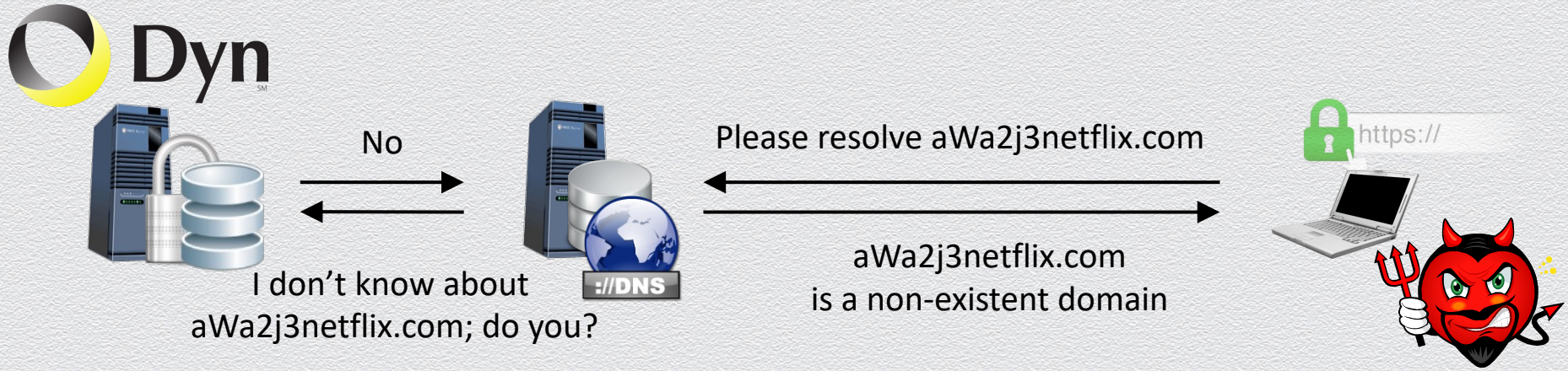
asked for b.com but
learned h’(e.com) & h’(z.com)

h’(b.com) = ntwo4
h: as in NSEC3
f: “message transformation” hash

h’(x) = h(RSA-Sign( , f(x)))

22.4.2 The Dyn DDoS attack

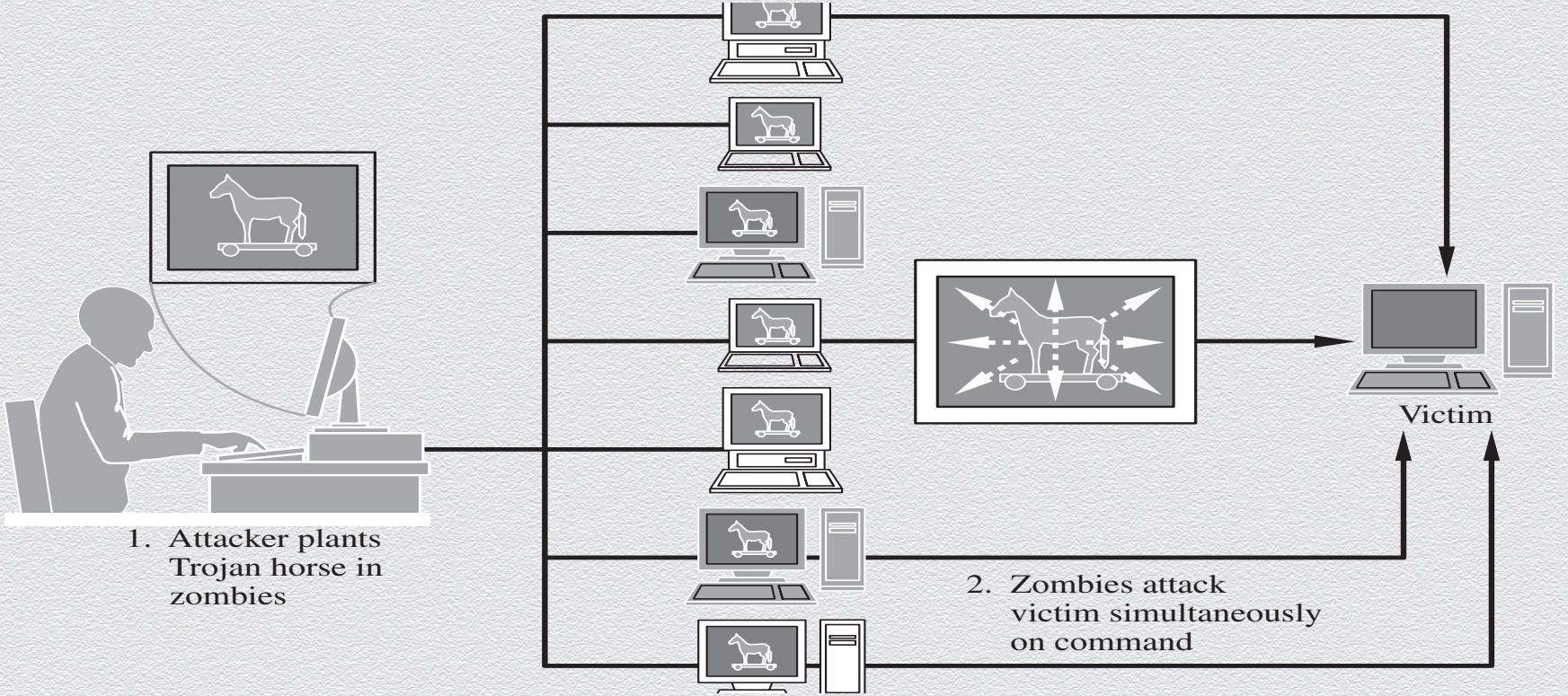
Core idea of attack: Saturate Dyn's primary servers



Attack:

- from a compromised machine ask for domain names that do not exist
- query is forwarded to fewer primary Dyn servers, i.e., defeating benefits of distribution
- ask **A LOT** of such queries to bring down the Dyn DNS service!

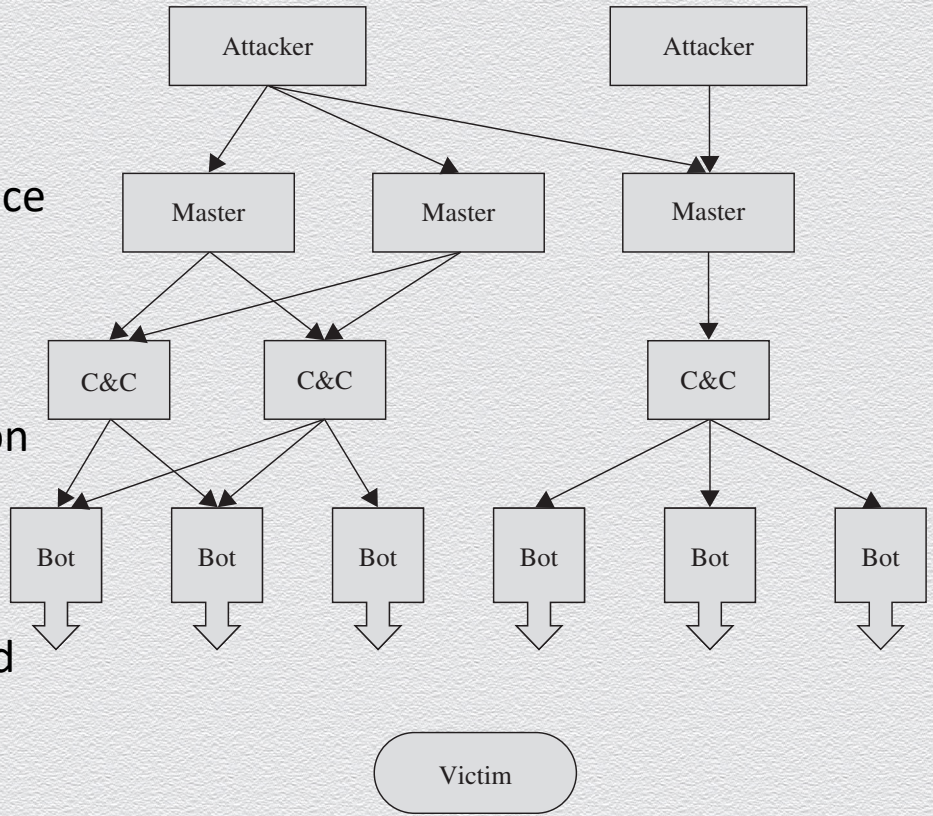
Distributed Denial of Service (DDoS)



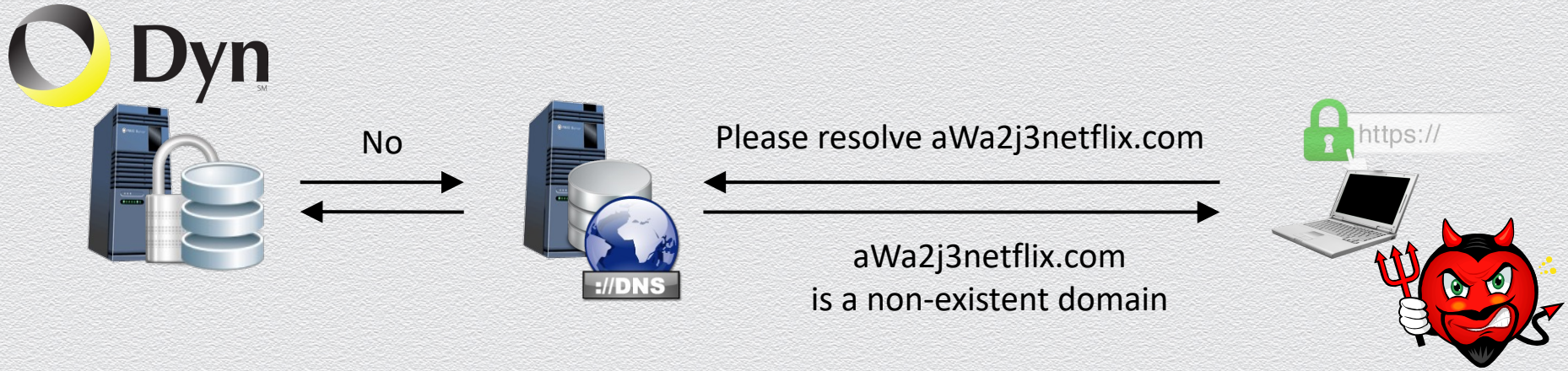
Botnets

Networks of machines running malicious code under remote control

- massive: scale to million of bots
  - comprise main tool for DDoS attacks
- stealth: remain undetected & difficult to trace
  - do little harm to the host machines
    - users won't likely remove malware
  - multiple-level attacker Vs. bots separation
- resilient: have redundant components
  - even if one master or C&C node is taken down, connectivity is maintained

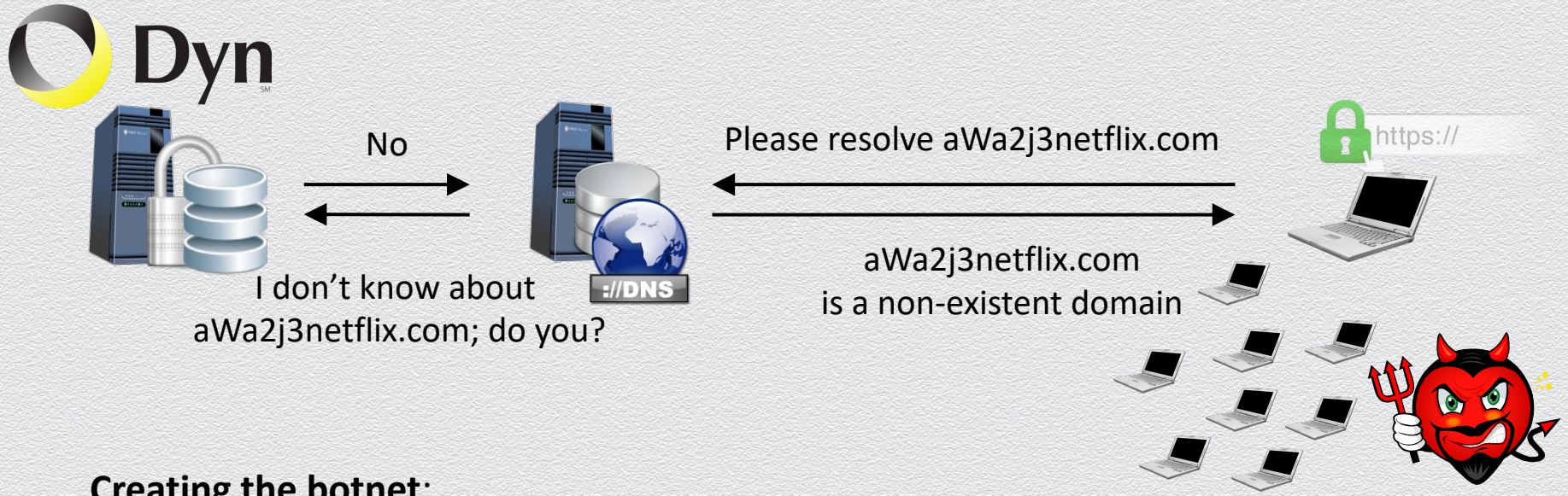


Why botnets are often behind DoS attacks?



- to avoid effective countermeasures and increase "attack" traffic
  - if the high-volume "attack" traffic comes from few devices, they can be filtered out by blocking their connections to the Dyn servers
  - by employing a large botnet of millions of devices the attacker inflicts a larger, more devastating "attack" traffic against the victim Dyn servers

Recruiting an army: Internet of Things (IoT)

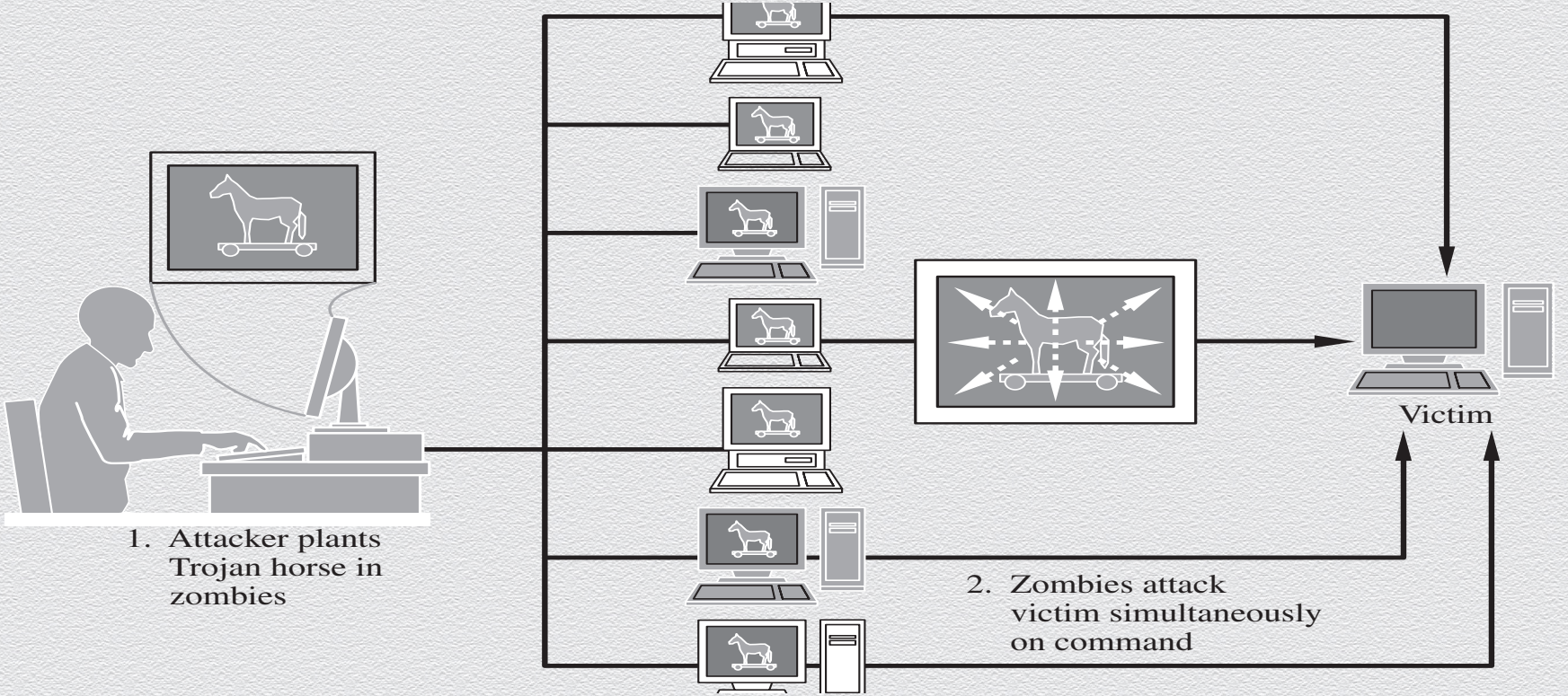


Creating the botnet:

- compromise easy targets: IoT "thin" devices, e.g., printers, cameras, home routers, …
- how? find a vulnerability on these devices...
  - all such devices used an OS with a static, hard-wired, thus known, admin password...!

22.5 DDoS attacks

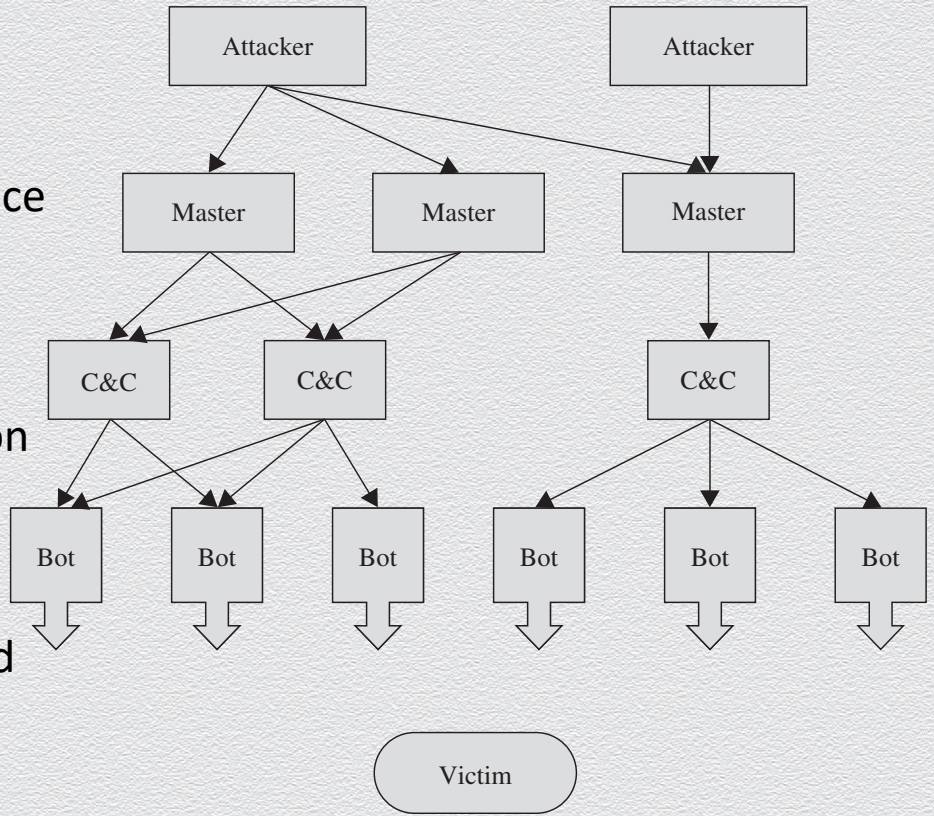
Recall: Distributed Denial of Service (DDoS)



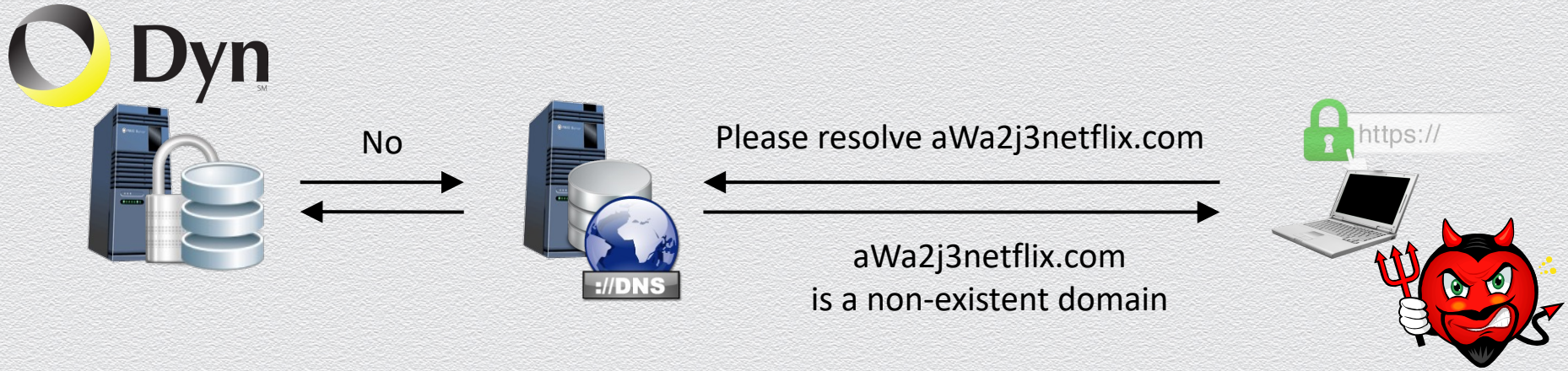
Recall: Botnets

Networks of machines running malicious code under remote control

- massive: scale to million of bots
  - comprise main tool for DDoS attacks
- stealth: remain undetected & difficult to trace
  - do little harm to the host machines
    - users won't likely remove malware
  - multiple-level attacker Vs. bots separation
- resilient: have redundant components
  - even if one master or C&C node is taken down, connectivity is maintained

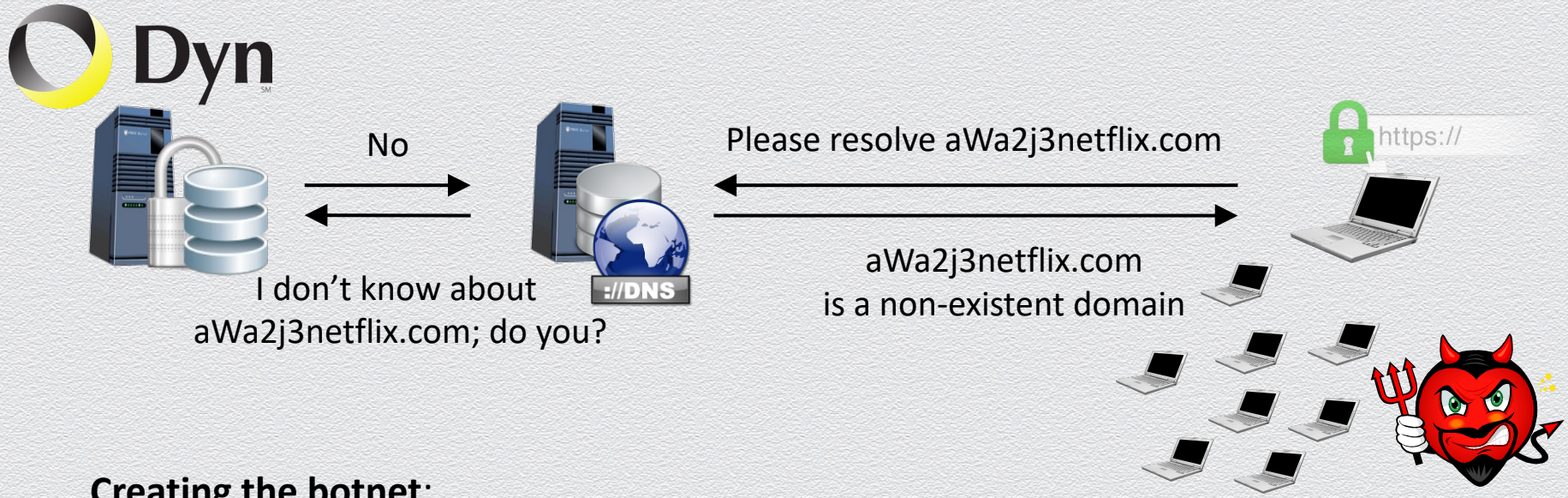


Recall: Why botnets are often behind DoS attacks?



- to avoid effective countermeasures and increase "attack" traffic
  - if the high-volume "attack" traffic comes from few devices, they can be filtered out by blocking their connections to the Dyn servers
  - by employing a large botnet of millions of devices the attacker inflicts a larger, more devastating "attack" traffic against the victim Dyn servers

Recall: Recruiting an army: Internet of Things (IoT)



Creating the botnet:

- compromise easy targets: IoT "thin" devices, e.g., printers, cameras, home routers, …
- how? find a vulnerability on these devices...
  - all such devices used an OS with a static, hard-wired, thus known, admin password...!

The Internet of Things (IoT)

Refers to Internet-connected everyday devices

- comprise a world of so-called smart devices
- examples:
  - smart appliances, such as refrigerators and dishwashers
  - smart home, such as thermostats and alarm systems
  - smart health, such as fitness monitors and insulin pumps
  - smart transportation, such as driverless cars
  - smart entertainment, such as video recorders
- potential downsides
  - loss of privacy
  - loss of control of data
  - potential for subversion
  - mistaken identification
  - uncontrolled access

Smartphones

The control hub of the IoT – important target for malware

- 2013: 143,211 distinct new forms of malware against mobile devices
- 98% targeted Android devices, far in excess of its market share
  - Android: open approach
    - unlike its competitors, does not limit the software users are allowed to install
    - thus, an easier target
  - Apple: locked-down approach
    - in contrast, only allows apps from its app store to be installed on its smartphones
    - all apps go through an approval process, which includes some security review
    - once approved, apps are signed, using a certificate approach

More generally on DoS attacks

DoS attacks are attempts to defeat a system's availability

- volumetric attacks (e.g., flooding)
  - potential weaknesses in the capacity of a computer network

insufficient capacity, overloading

- application-based attacks (e.g., routing malfunction, blocked access)
  - exhaust the resources of an application that services a particular network

blocked access

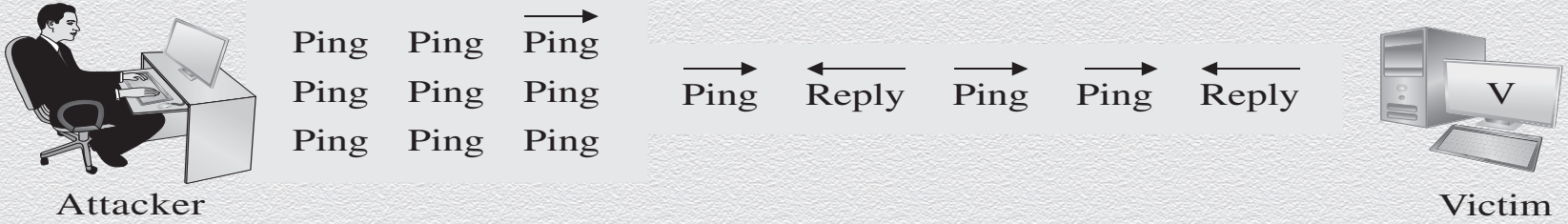
- disabled communications (e.g., access failure)
  - physically disable a communication link or disrupt a single-point of failure
- hardware or software failure (e.g., access failure)
  - failures on machines or programs (without fault-tolerance provisions)

unresponsive asset

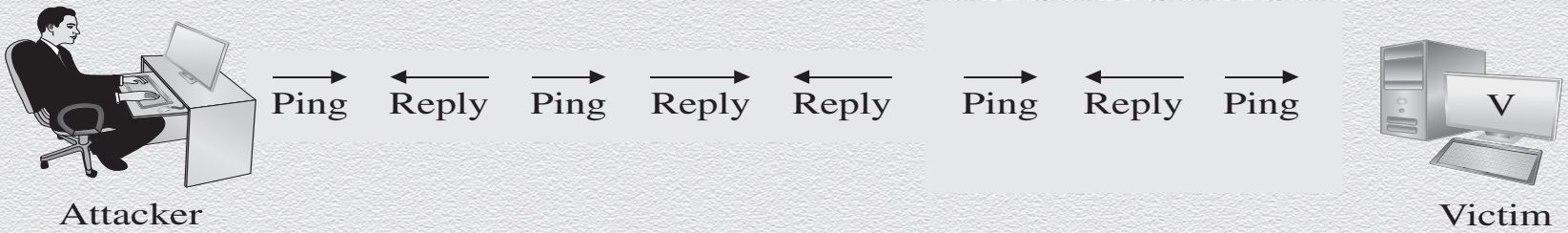
Examples

- Benign errors
  - Beth Israel Hospital system downtime, in 2002, due to mishandling of switches
- Malicious code
  - use vulnerabilities in communication protocols
  - e.g., cause uncontrolled congestion in TCP communications (Vs. UDP)
  - e.g., exploit/misuse Internet Control Message Protocols (ICMP)
    - ping (destination is reachable and functional)
    - echo (the connection between two machines is reliable)
    - destination unreachable (destination address cannot be accessed)
    - source quench (destination is saturated and source should suspend transmissions)

Ping flood (or pink of death)



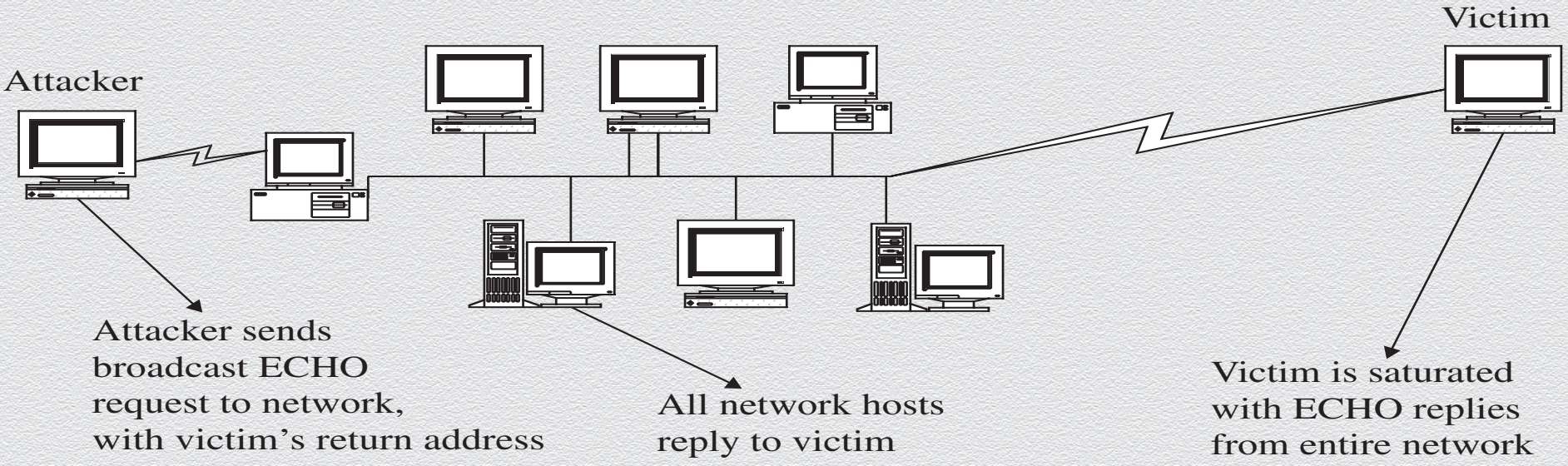
(a) Attacker has greater bandwidth



(b) Victim has greater bandwidth

Smurf Attack

Unwitting victims become accomplices



Echo-Chargen

Victim A

Chargen packet with echo bit on →

← Echoing what you just sent me

Chargen another packet with echo bit on →

← Echoing that again

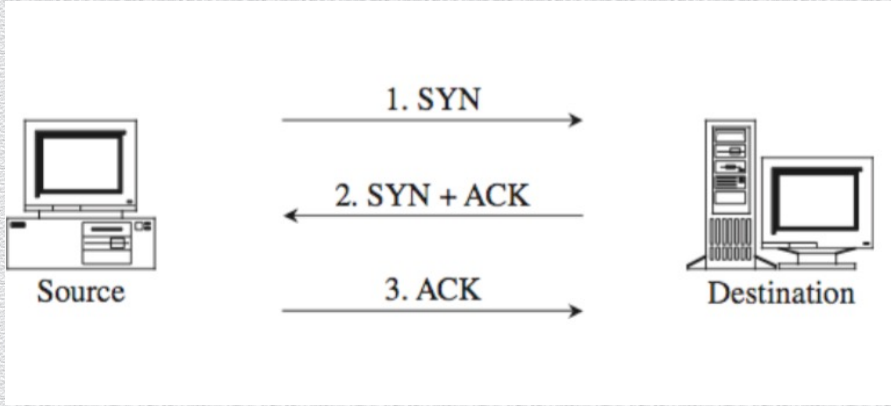
Chargen another packet with echo bit on →

Victim B

SYN flood

Three-way handshake used in TCP to establish a new session

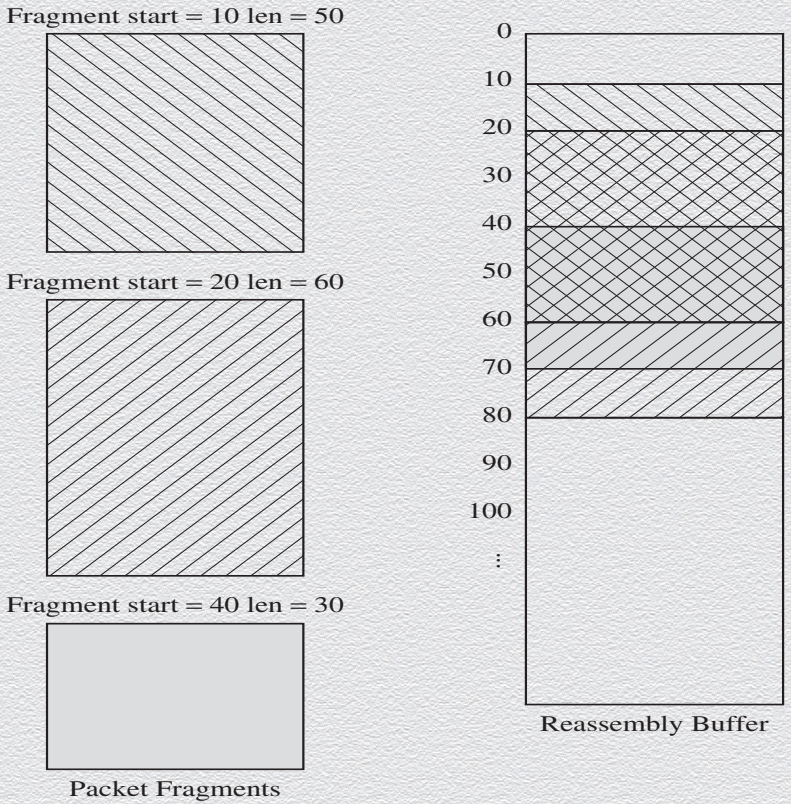
- source & destination exchange control messages to complete session creation
- destination keeps track of incomplete session-creation protocols (SYN-RECV cache)
- attacker spoofs return address of many SYN handshake messages sent to victim
- victim's cache is filled (after ~20) pending incomplete sessions and delays are created



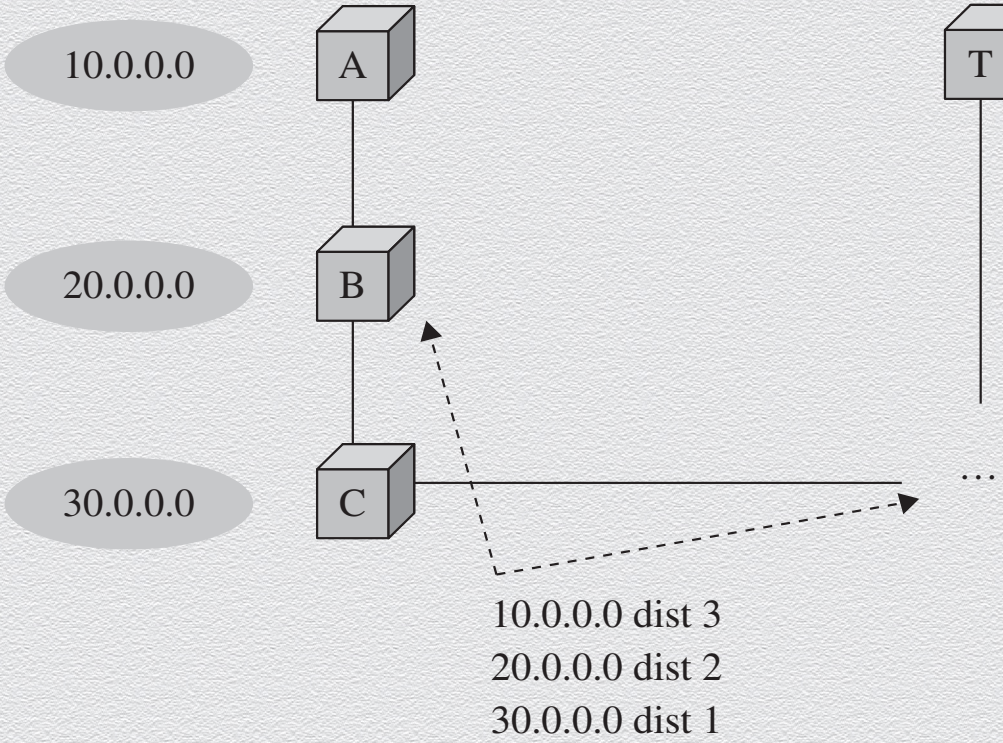
Teardrop Attack

IP datagrams allow to carry variable-length data

- each datagram specifies the length of its data payload and its offset (start-end)
- the attacker can spoof these metadata so that recipient's state remains inconsistent
- sent packets cannot possibly be reassembled, as they conflict instructions
- in extreme cases, this can cause the entire OS to lock up



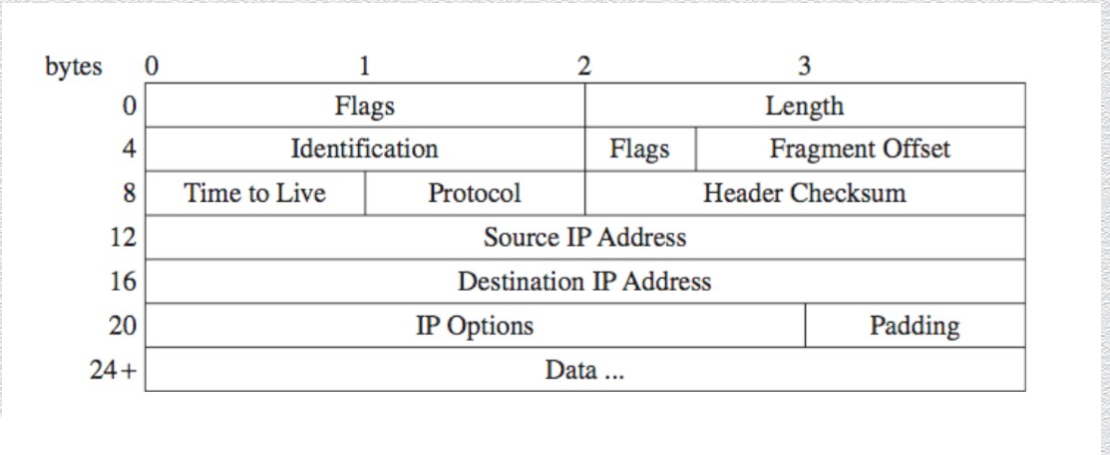
Rerouting routing

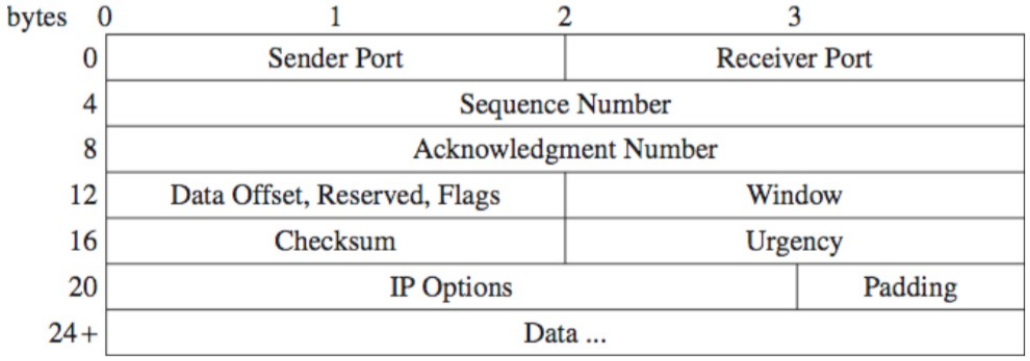


- router C advertises the routes it knows about to the routers adjacent to it
- routers rely on these advertising messages to be accurate
- when they aren't, DoS can ensue

TCP/IP headers

- headers of IP datagram and TCP packet
- IP: right
- TCP: below





Session hijacking

- an attacker is able to synchronize with a receiver while breaking synchronization with the sender and resetting the sender's connection
- the attacker continues the TCP session while the sender thinks the connection just broke off