Advanced Computer Networks

Networking Basics

Lin Wang

Period 2, Fall 2021

VU VRIJE UNIVERSITEIT AMSTERDAM

Course outline

Warm-up

- Introduction (history, principles)
- **Networking basics**
- Networking data structures and algorithms
- Network transport

Data centers

- Data center networking
- Data center transport

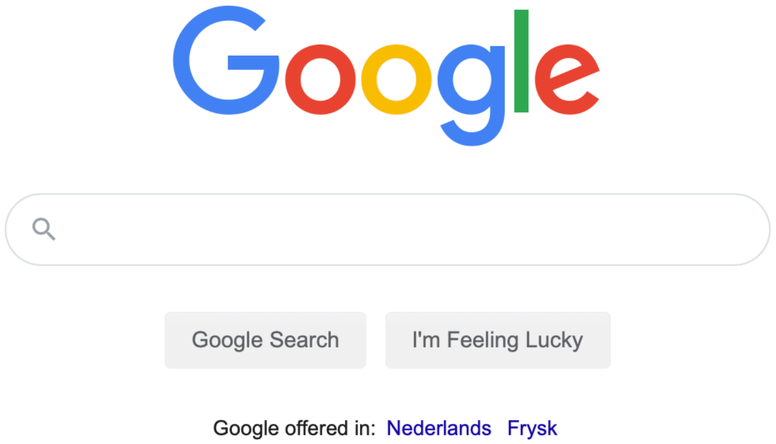
Programmability

- Software defined networking
- Network automation
- Network function virtualization
- Programmable data plane

Application

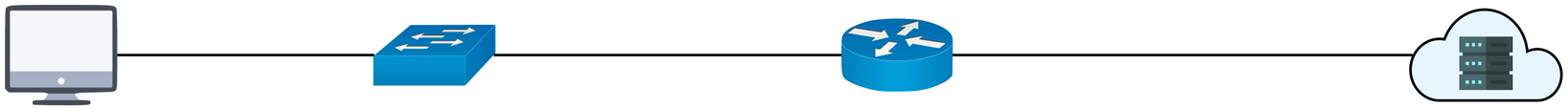
- Network monitoring
- In-network computing
- Machine learning for networking

Learning objectives



What happens under the hood when you visit https://www.google.com?

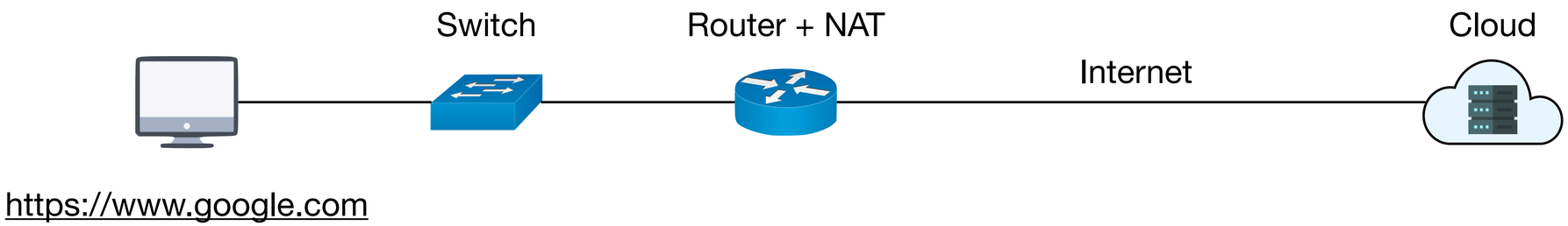
A simplified networking scenario



https://www.google.com

What networking concepts are involved?

A simplified networking scenario



Key networking concepts: DNS, Socket, TCP, IP routing, Ethernet, ARP, NAT

Domain name system (DNS)

From google.com to 142.251.36.14

Domain name system (DNS)

If you want to mail someone

- You need to get their address first

What about the Internet?

- If you need to reach Google, you need their IP
- Does anyone know Google's IP?

Problem

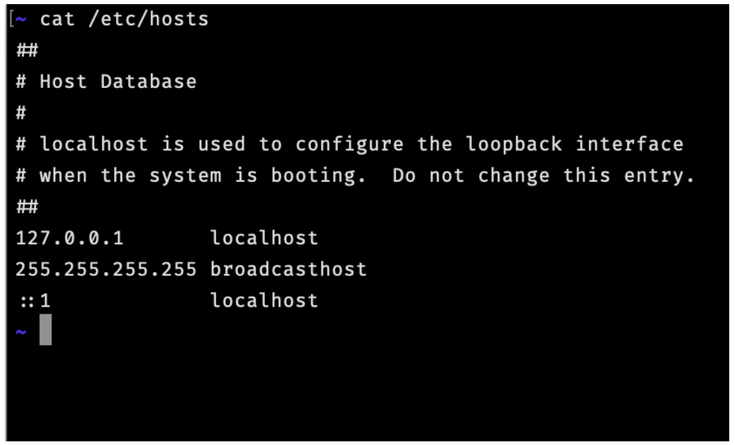
- People cannot remember IP addresses
- Need human readable names that map to IPs

```
[~ dig google.com                                                                ]

; <<>> DiG 9.10.6 <<>> google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21646
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 9

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.             194     IN      A       142.251.36.14

;; AUTHORITY SECTION:
google.com.             140319  IN      NS      ns2.google.com.
google.com.             140319  IN      NS      ns1.google.com.
google.com.             140319  IN      NS      ns4.google.com.
google.com.             140319  IN      NS      ns3.google.com.

;; ADDITIONAL SECTION:
ns1.google.com.         160187  IN      A       216.239.32.10
ns2.google.com.         152334  IN      A       216.239.34.10
ns3.google.com.         152334  IN      A       216.239.36.10
ns4.google.com.         152334  IN      A       216.239.38.10
ns1.google.com.         152334  IN      AAAA    2001:4860:4802:32::a
ns2.google.com.         152334  IN      AAAA    2001:4860:4802:34::a
ns3.google.com.         152334  IN      AAAA    2001:4860:4802:36::a
ns4.google.com.         152334  IN      AAAA    2001:4860:4802:38::a

;; Query time: 3 msec
;; SERVER: 130.37.236.48#53(130.37.236.48)
;; WHEN: Sun Oct 31 17:31:17 CET 2021
;; MSG SIZE  rcvd: 303

~
```

DNS history

Before 1983 (the advent of DNS), all mappings were in a single file

- `/etc/hosts` on Linux
- `C:\\Windows\System32\drivers\etc\hosts` on Windows



Centralized, manual system

- Changes were submitted to SRI (Stanford Research Institute) via email
- End hosts periodically FTP new copies of the `hosts` file
- Administrators could pick names at their discretion
- Any name was allowed: `alices_server_at_vrije_universiteit_amsterdam`

Not scalable Hard to enforce uniqueness Consistency issue

DNS overview

Distributed database

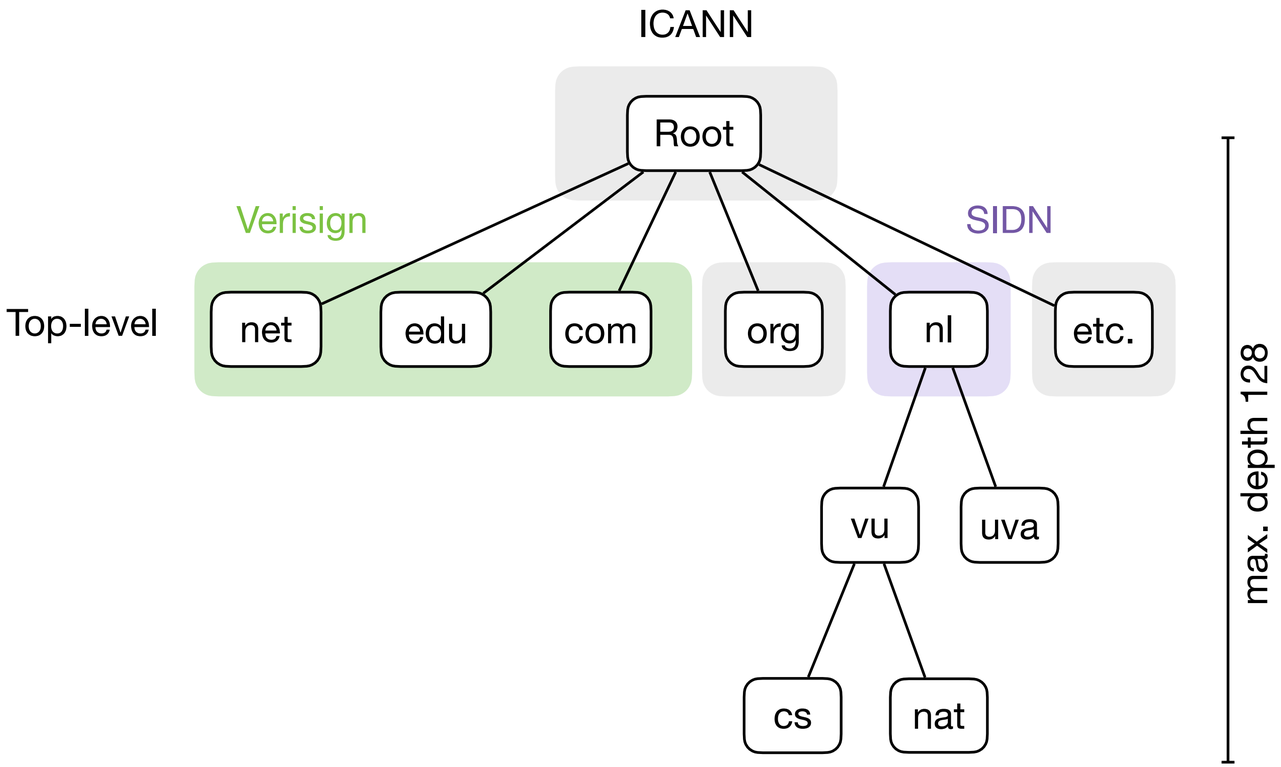
- No centralization → scalability

Simple client/server architecture

- UDP port 53, some implementations also use TCP

Hierarchical namespace

- As opposed to original, flat namespace
- E.g., `.com` → `google.com` → `mail.google.com`



Tree is divided into zones and each zone has an administrator, with a DNS server (maybe replicated)

Root name server

Responsible for the root zone file

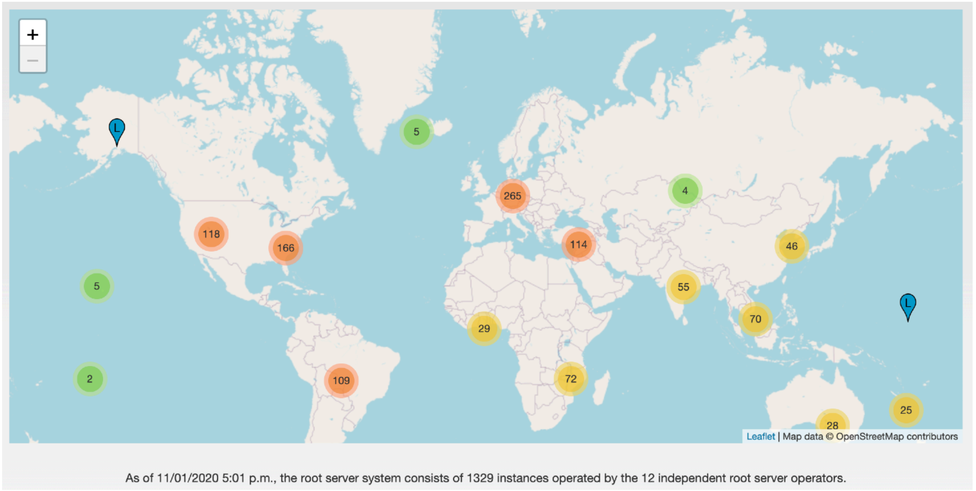
- Lists the top-level domains (TLDs) and who controls them
- ~ 2MB file size

Administrated by International Corporation for Assigned Names and Numbers (ICANN)

- 13 root servers, labeled A → M
- All are anycasted, i.e., they are globally replicated

Contacted when names cannot be resolved locally

- In practice, most systems cache this information

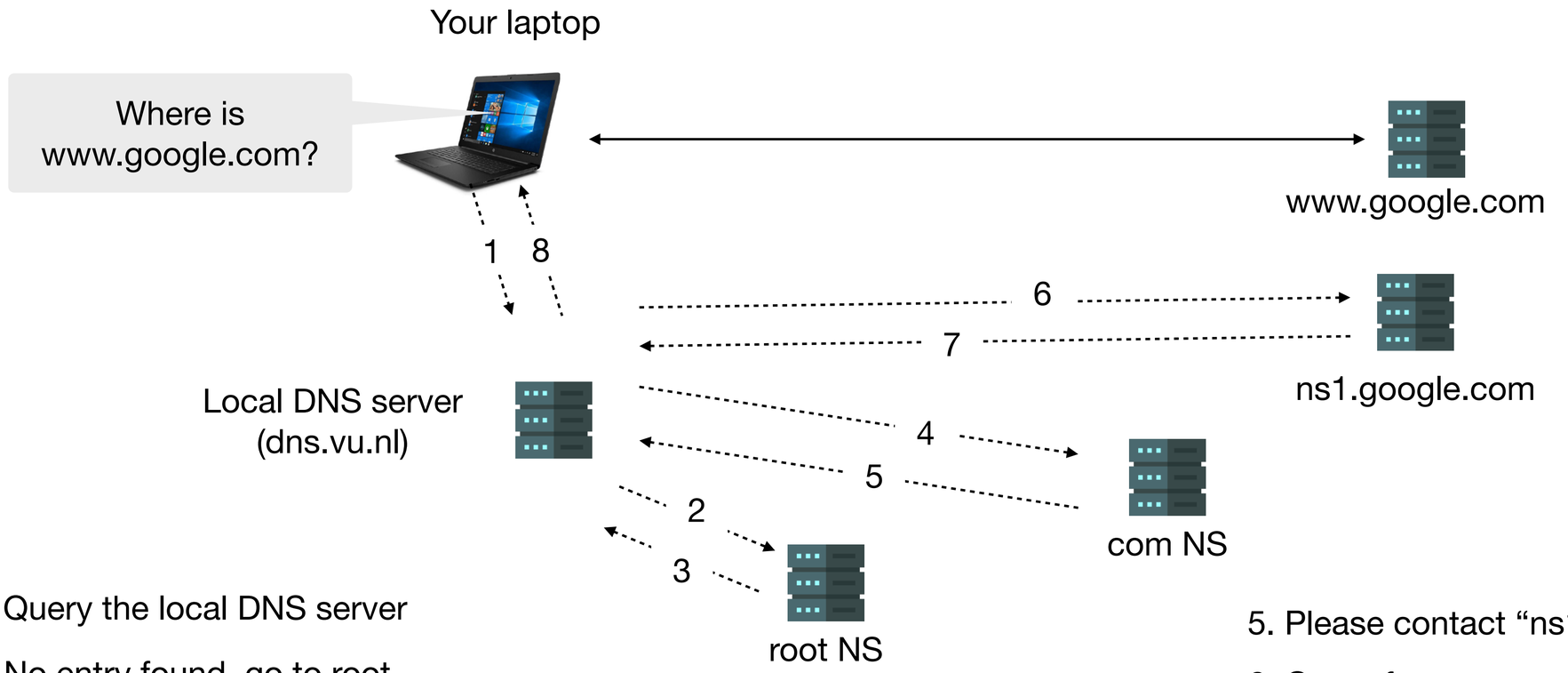


https://root-servers.org

How does a URL get resolved to an IP address?

Recursive DNS query

Each layer may apply caching (1-72 hours) to improve efficiency



1. Query the local DNS server
2. No entry found, go to root
3. Please contact “com”
4. Query for www.google.com
5. Please contact “ns1.google.com”
6. Query for www.google.com
7. www.google.com → 172.217.20.110
8. 172.217.20.110 is the answer

DNS types

Query	Name: www.cs.vu.nl Type: A (or AAAA)
Resp.	Name: www.cs.vu.nl Value: 130.37.164.171

DNS resolution (AAAA for IPv6)

Query	Name: foo.cs.vu.nl Type: CNAME
Resp.	Name: foo.cs.vu.nl Value: bar.cs.vu.nl

Look for alias (canonical hostname)

Query	Name: cs.vu.nl Type: NS
Resp.	Name: cs.vu.nl Value: 130.37.164.1

Query for DNS server responsible for the partial name

Query	Name: cs.vu.nl Type: MX
Resp.	Name: cs.vu.nl Value: mail.cs.vu.nl

Look for the mail server

The importance of DNS

Without DNS…

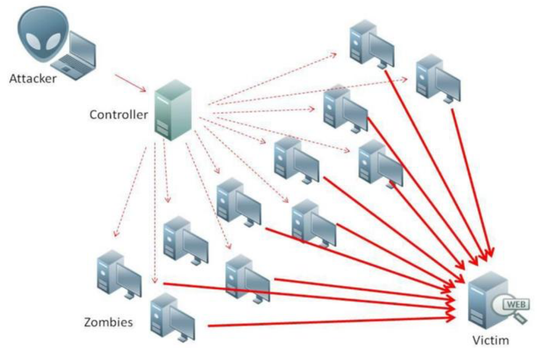
- How could you get to any websites?

You are your mail server

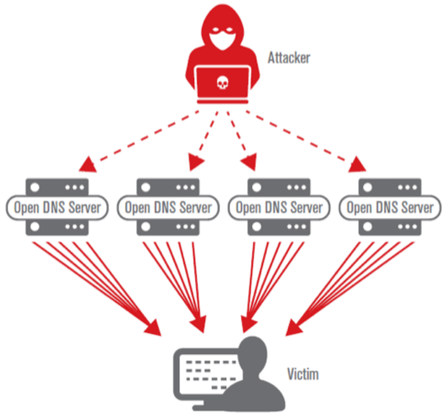
- When you sign up for websites, you use your email address
- What if someone hijacks the DNS for your mail server?

DNS is the root of trust for the web

- When a user types www.ing.nl, they expect to be taken to their bank's website
- What if the DNS record is compromised?



Distributed Denial of Service (DDoS)

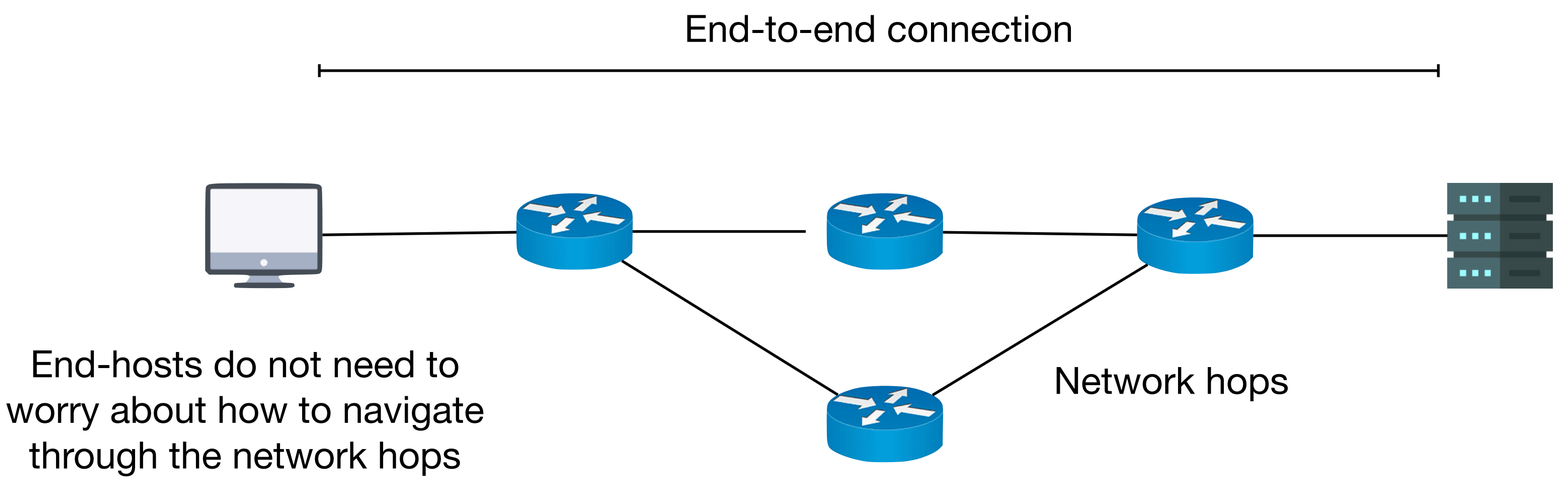


DNS amplification attack

Socket and TCP

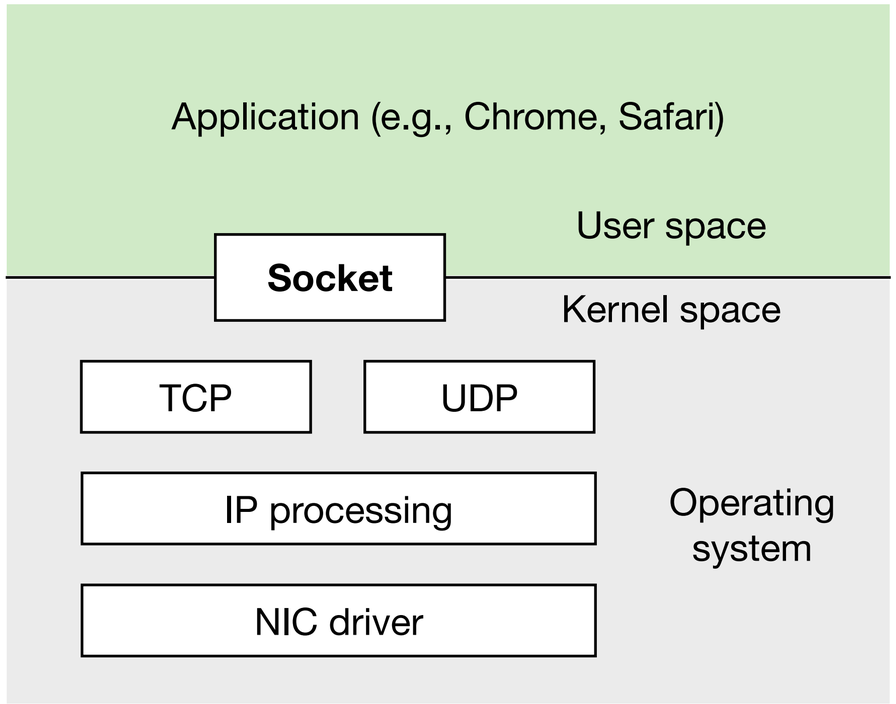
Establishing a “connection”

What is a connection?



Application
Transport
Network
Link
Physical

Making a connection through the socket interface



Network Interface Controller (NIC)

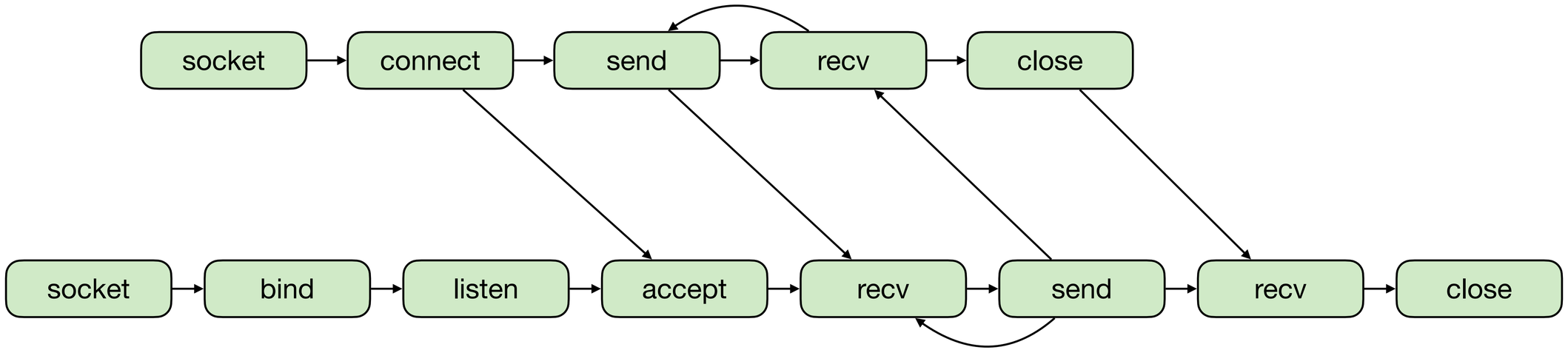
Socket represents the **communication endpoint.** It is an abstraction for user applications to access network functionalities implemented in the OS kernel.

Berkeley sockets

The de-facto socket implementation in Unix-like systems

- Also known as BSD sockets or POSIX sockets

Have you ever implemented a client-server chat program?

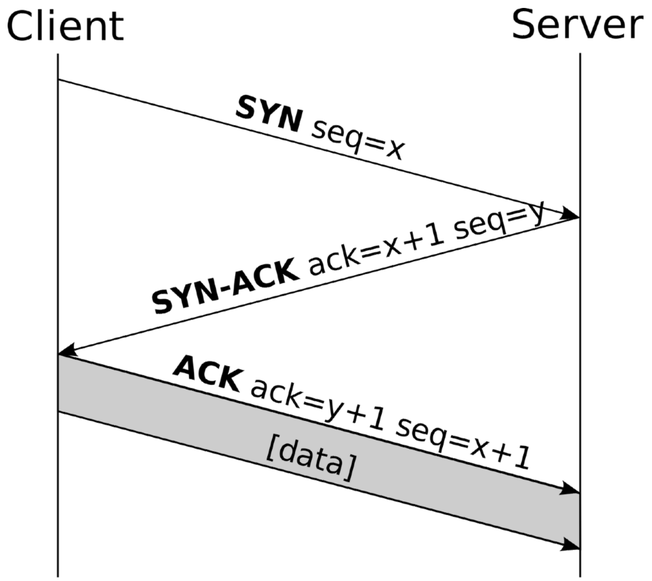


https://man7.org/linux/man-pages/man2/socket.2.html

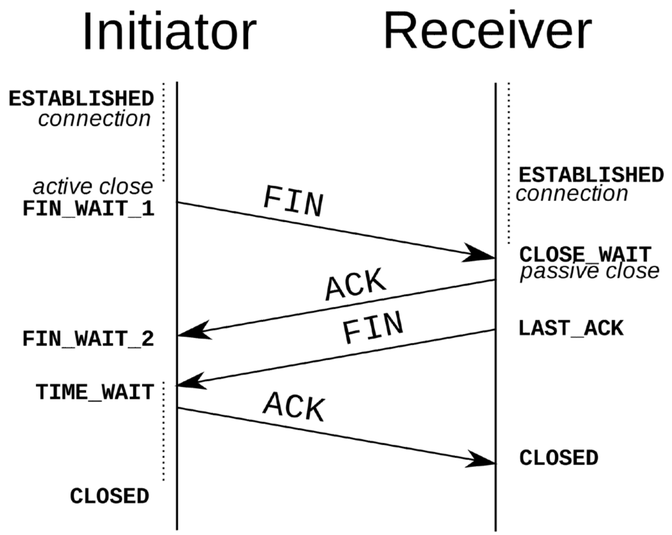
Transmission control protocol (TCP)

RFC 793

A connection-oriented transport-layer protocol



TCP connection estimation



TCP connection termination

What functionalities do TCP provide?

TCP functionalities

Reliable delivery

- Integrity check
- Packet retransmission upon losses
- Packet reordering

Flow and congestion control

- Flow control: the receiver is not overrun by the sender
- Congestion control: the network is not overrun by the sender

How are these functionalities achieved by TCP?

TCP segment header format

TCP segment header

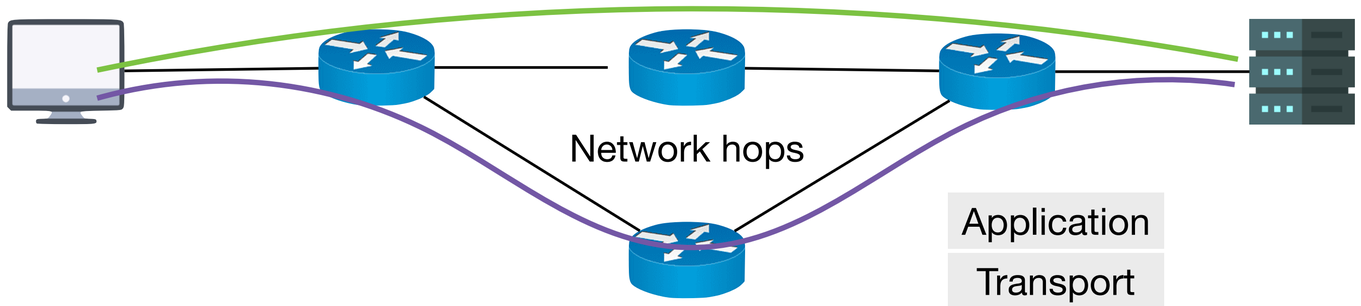
Offsets	Octet	0								1								2								3							
Octet	Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0	0	Source port																Destination port															
4	32	Sequence number																															
8	64	Acknowledgment number (if ACK set)																															
12	96	Data offset				Reserved 0 0 0				NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size														
16	128	Checksum																	Urgent pointer (if URG set)														
20	160	Options (if *data offset* > 5. Padded at the end with "0" bytes if necessary.)																															
⋮	⋮																																
60	480																																

IP routing

Finding the path for the connection

Network routing

Finding a path to interconnect the source and the destination

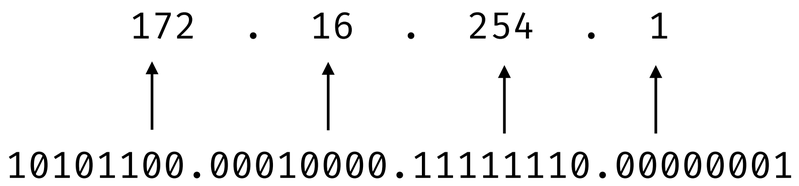


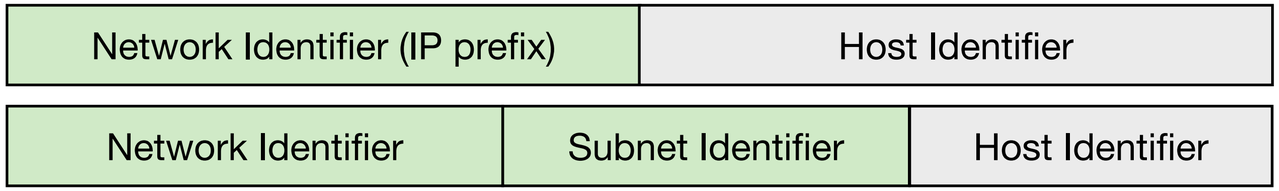
Application
Transport
Network
Link
Physical

Network layer address: example IPv4

RFC7020

Private addresses: `10.0.0.0/8`, `172.16.0.0/12`, `192.168.0.0/16`



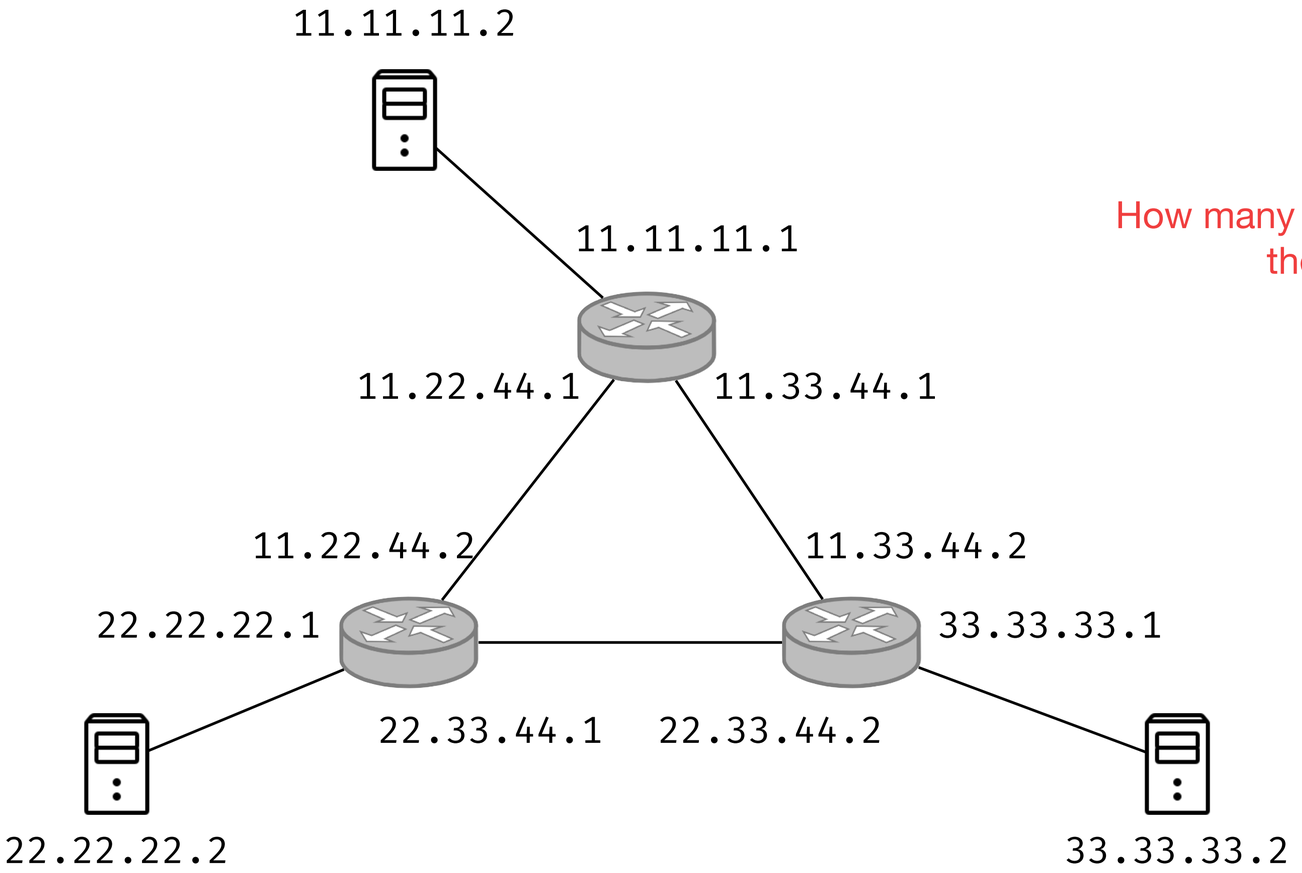


Classless Inter-Domain Routing (CIDR) notation: `10.0.0.1/24`

Subnet mask notation: `255.255.255.0`

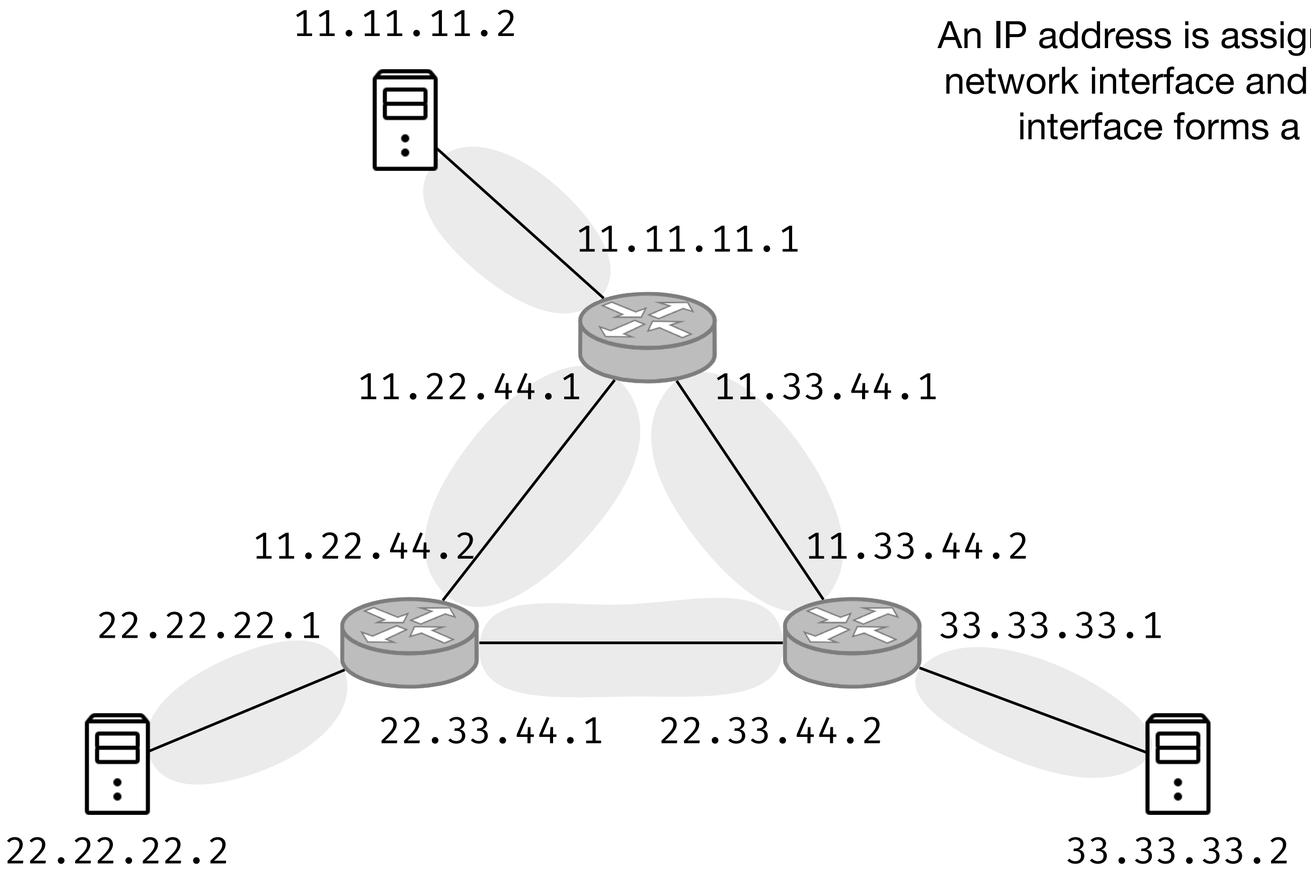
Who do we assign IP addresses to? A host? switch? router? or...

Routers interconnecting subnets



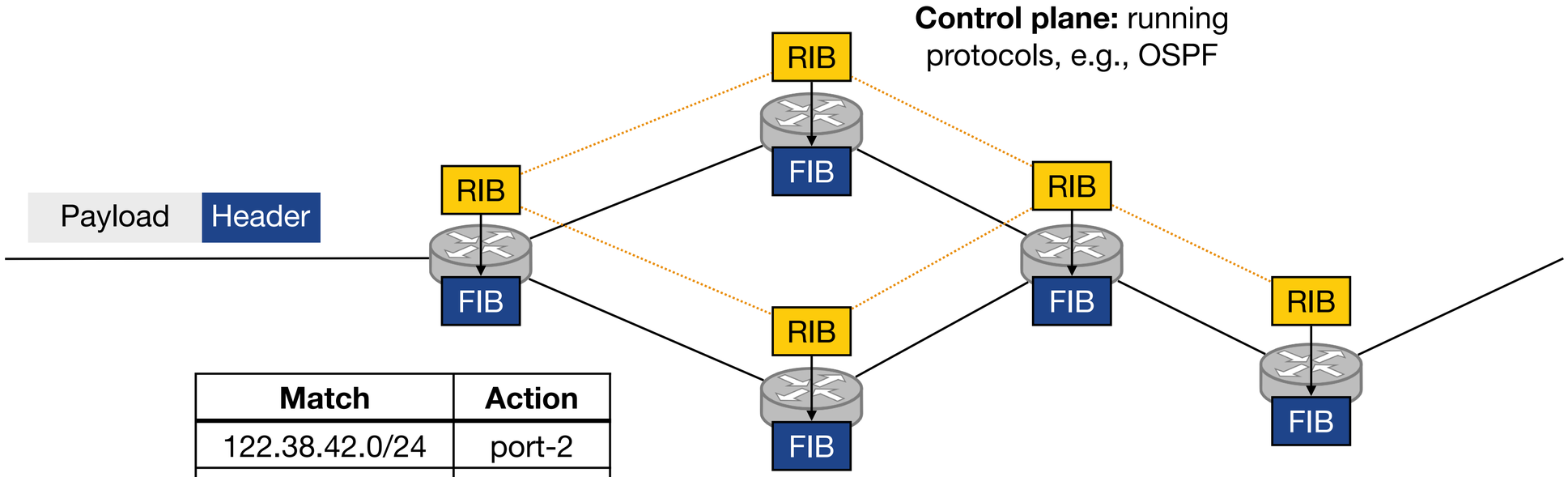
How many subnets are there in the network?

Routers interconnecting subnets



An IP address is assigned to every network interface and each router interface forms a subnet.

IP routing



Control plane: running protocols, e.g., OSPF

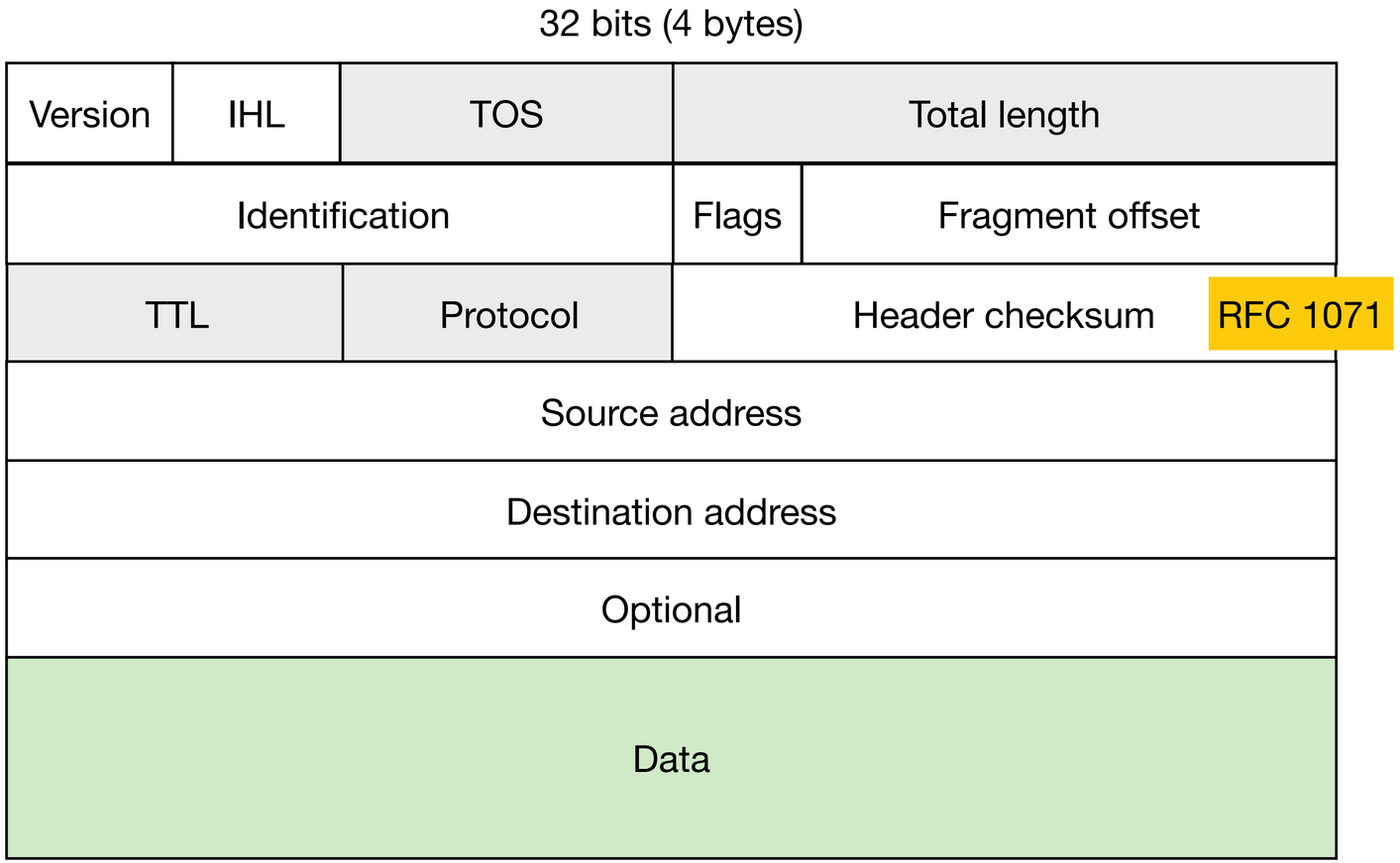
Match	Action
122.38.42.0/24	port-2
116.16.0.0/16	port-1
139.70.8.0/24	drop

Data plane: packet forwarding with the match-action model

RIB: routing information base, or routing table
FIB: forwarding information base

IPv4 packet format

RFC 791



TOS: type of service, two bits used for Explicit Congestion Notification RFC 3168

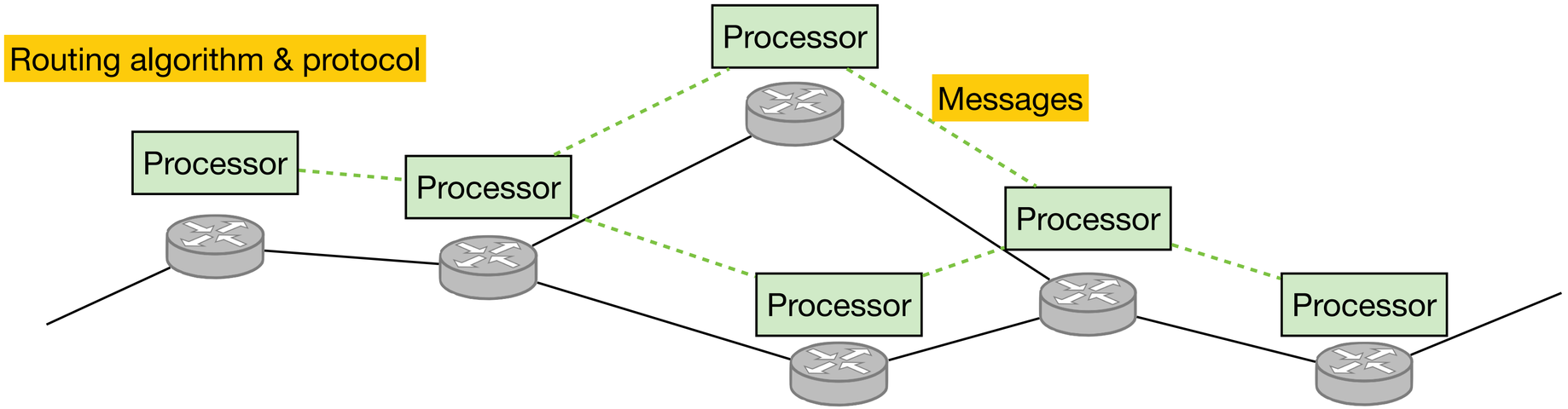
Total length: max. 65535 bytes, typically bounded by Ethernet MTU (1500 bytes)

TTL: decreased by one when passing a router, packet dropped by the router when it reaches 0

Protocol: transport layer protocol (6 for TCP, 17 for UDP)

How to generate forwarding tables?

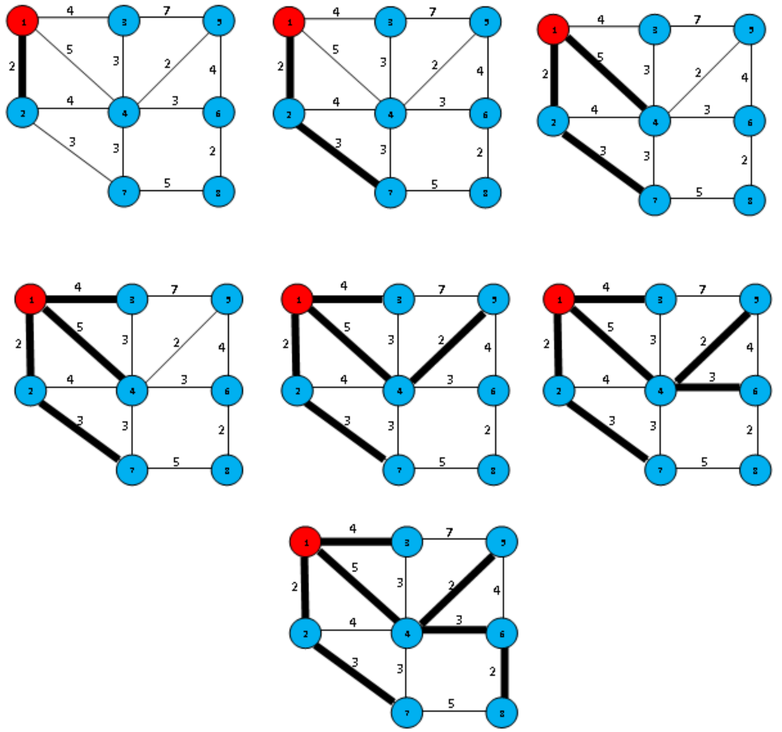
Control plane: modern routers employ a **distributed protocol** to exchange messages and compute shortest paths to other routers to generate the forwarding table: OSPF (link state), BGP (distance vector)



Routing protocol: intra-domain

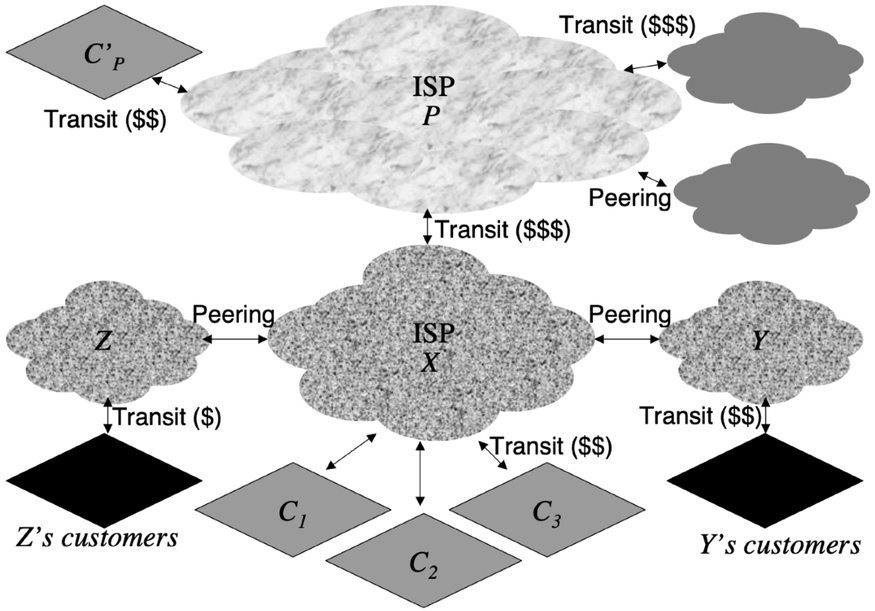
Open Shortest Path First (OSPF):

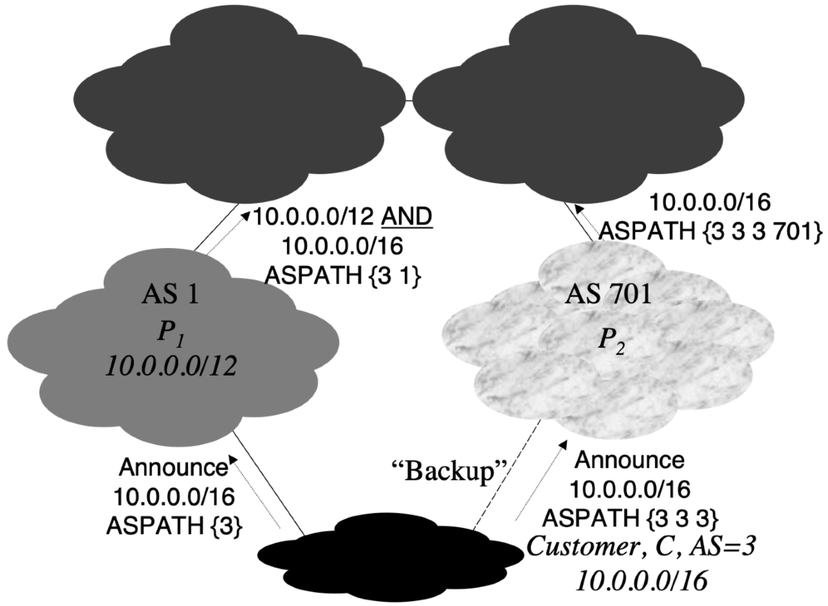
- Routers exchange link-state messages to learn the topology
- Each router runs the **Dijkstra's algorithm** to computer the shortest paths to other routers
- Each router generates the forwarding table entries based on the shortest paths



Routing protocol: inter-domain (BGP)

Announcing routes to peer ASes

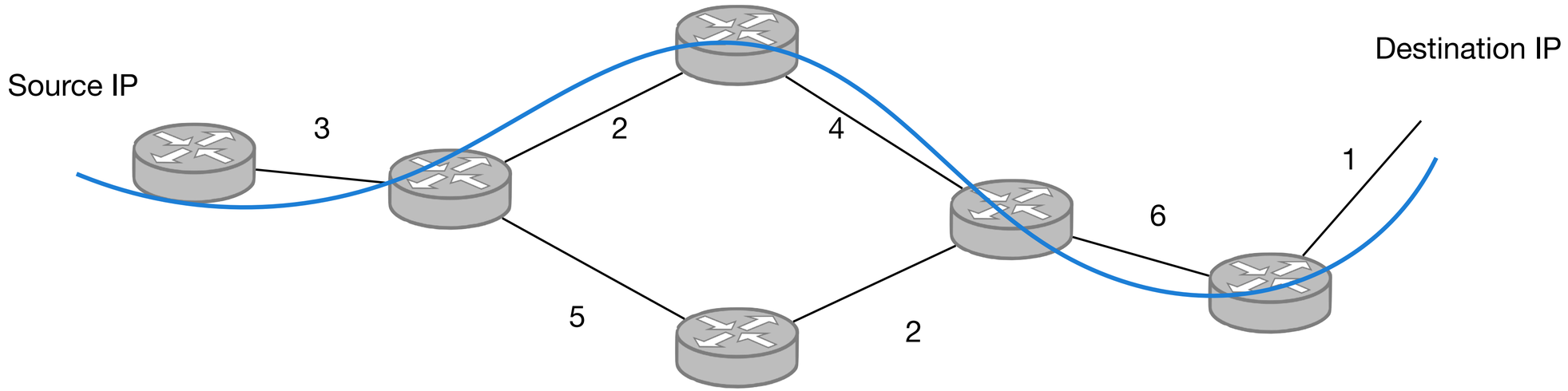




Traffic engineering

RFC 3272 RFC 2702

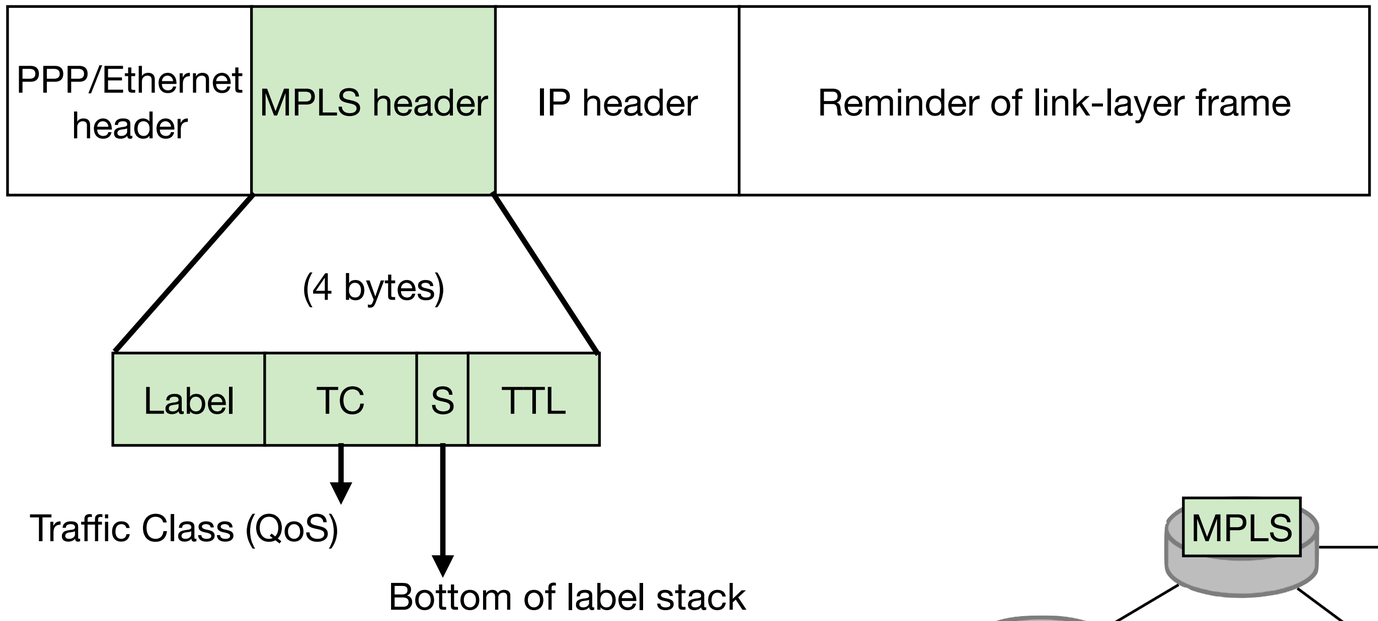
The aspect of network engineering that deals with the issue of **performance evaluation and performance optimization** of operational IP networks. Traffic engineering encompasses the application of technology and scientific principles to the measurement, characterization, modeling, and control of Internet traffic.

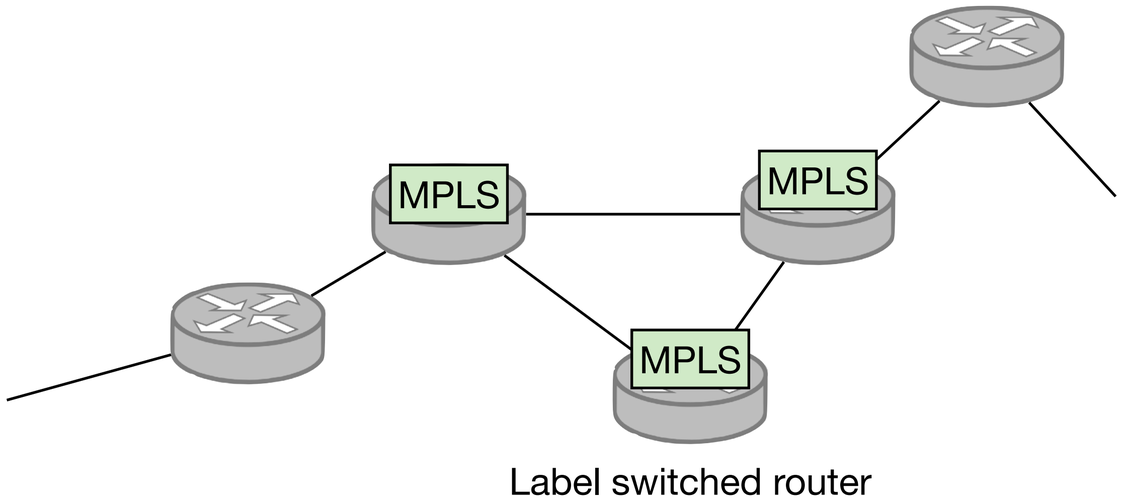


What performance issues can you foresee in network routing?

Multiprotocol label switching (MPLS)

RFC 3031 RFC 3032



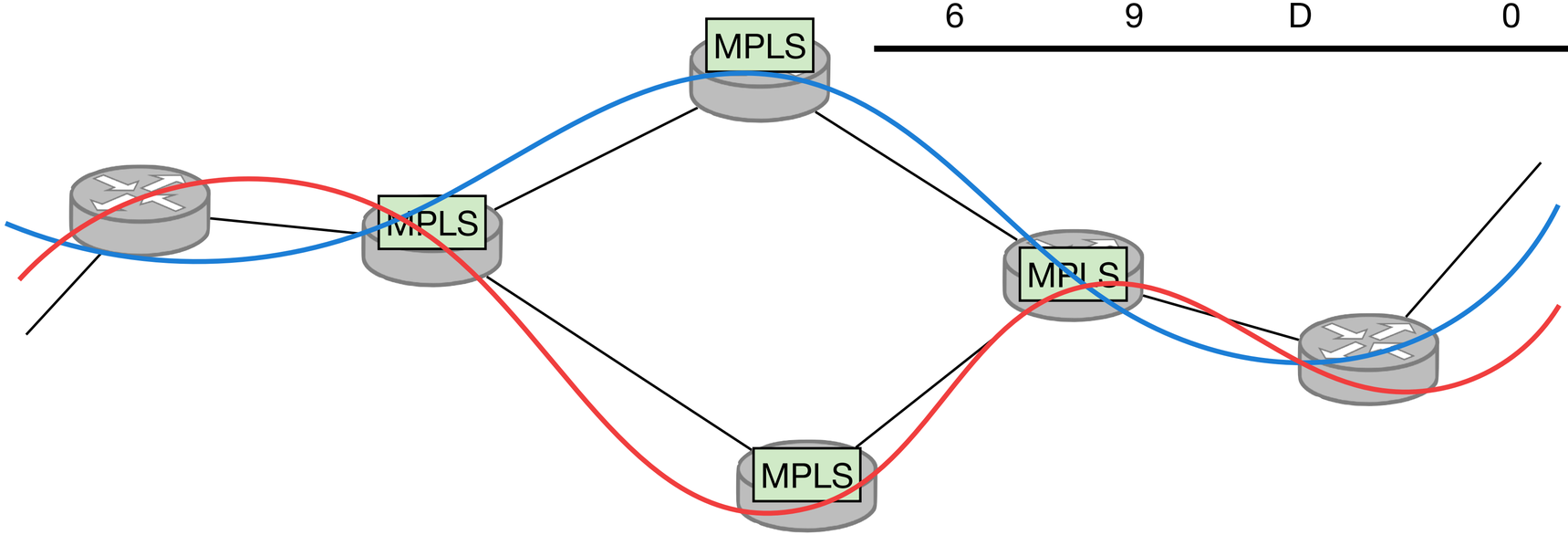


Traffic engineering with MPLS

RFC 2702 RFC 3272 RFC 3346

Even for the same source-destination (IP) pair, multiple paths can be set up for forwarding the traffic. By carefully assigning the labels, we can control how the traffic is shipped on the network links - traffic engineering

In-label	Out-label	Dest	Out interface
10	12	A	1
6	9	D	0

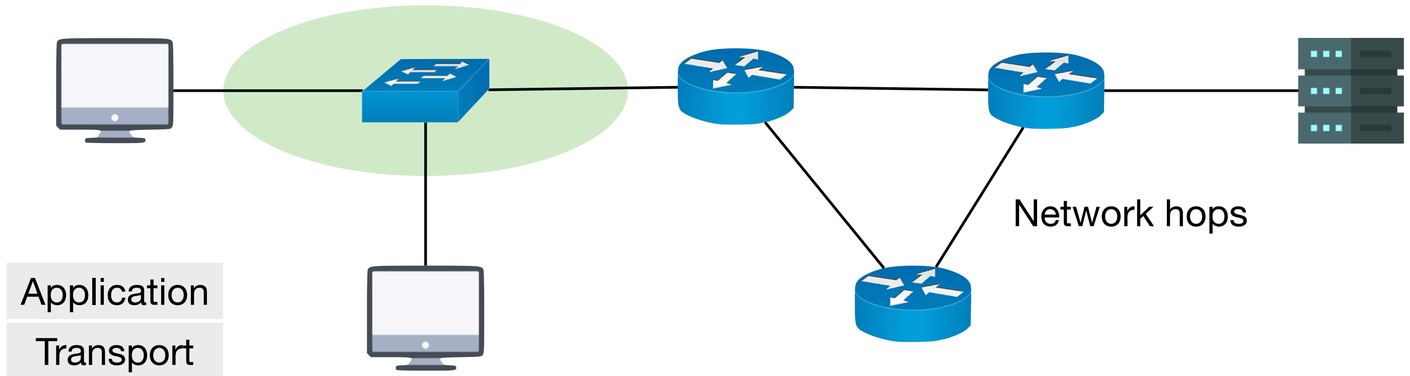


Ethernet and ARP

Sending packets within the local network

Link-layer forwarding

Finding where to forward the packet within the local network



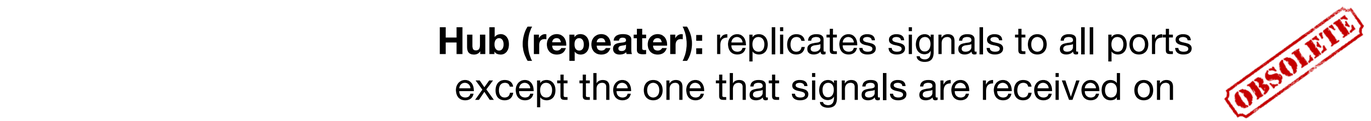
Application
Transport
Network
Link
Physical

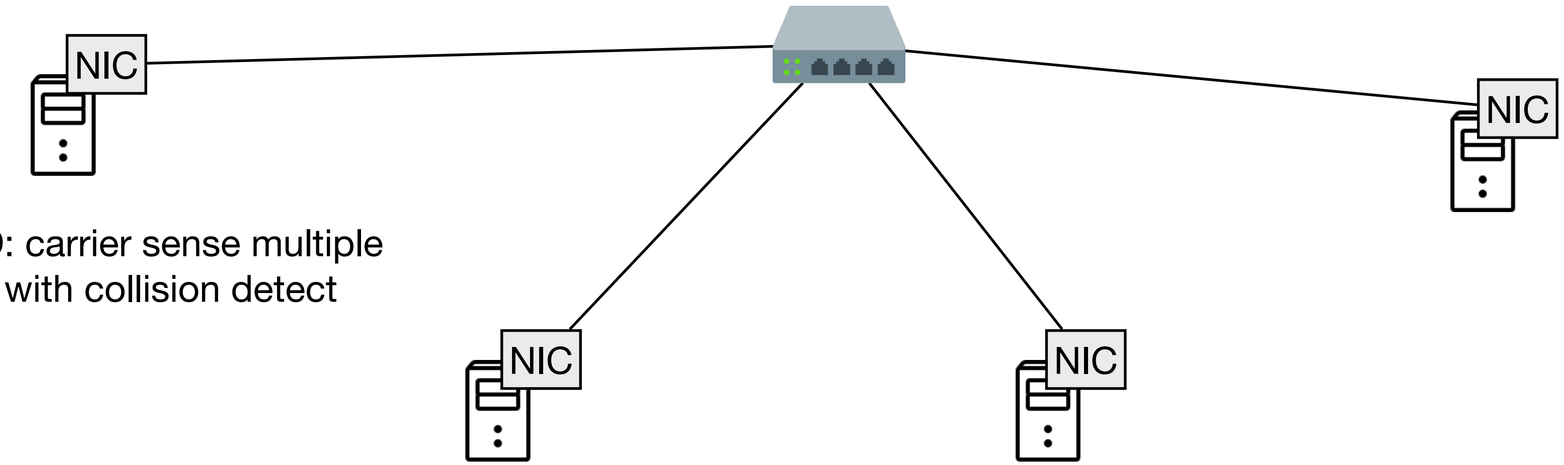
Ethernet

IEEE 802.3

A family of networking technologies commonly used in Local Area Networks (LAN) and other networks

Hub (repeater): replicates signals to all ports except the one that signals are received on





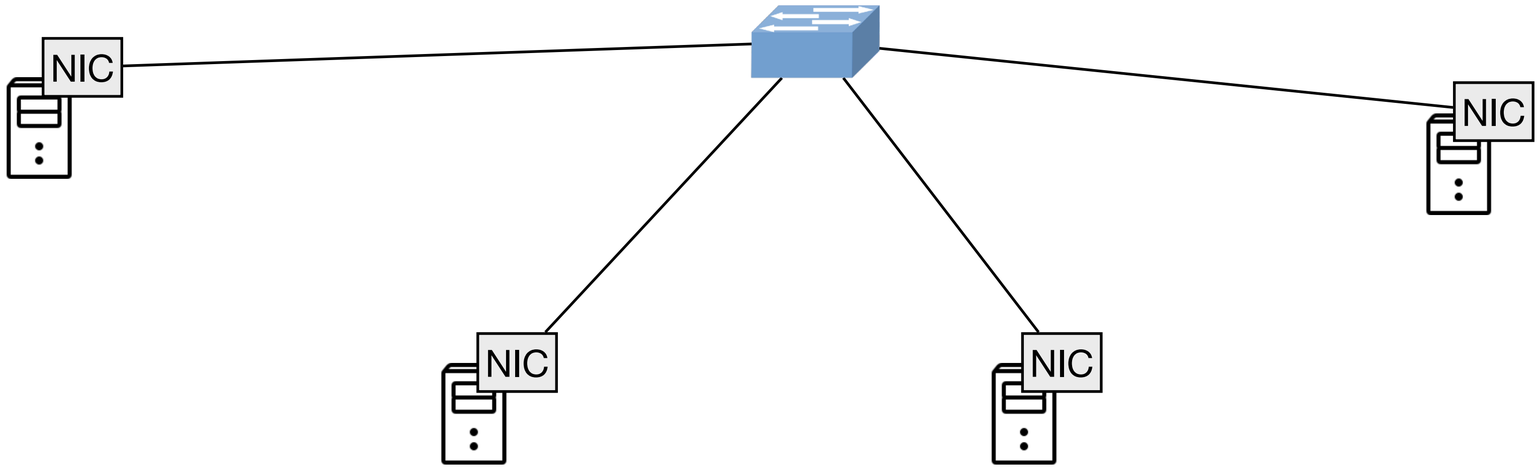
CSMA/CD: carrier sense multiple access with collision detect

Switched Ethernet

Different Ethernet segments are interconnected with switches (that work on the link layer)

Switch: creates Ethernet segments and forwards frames between segments based on the MAC address

Switches typically do not need to run CSMA/CD, why?

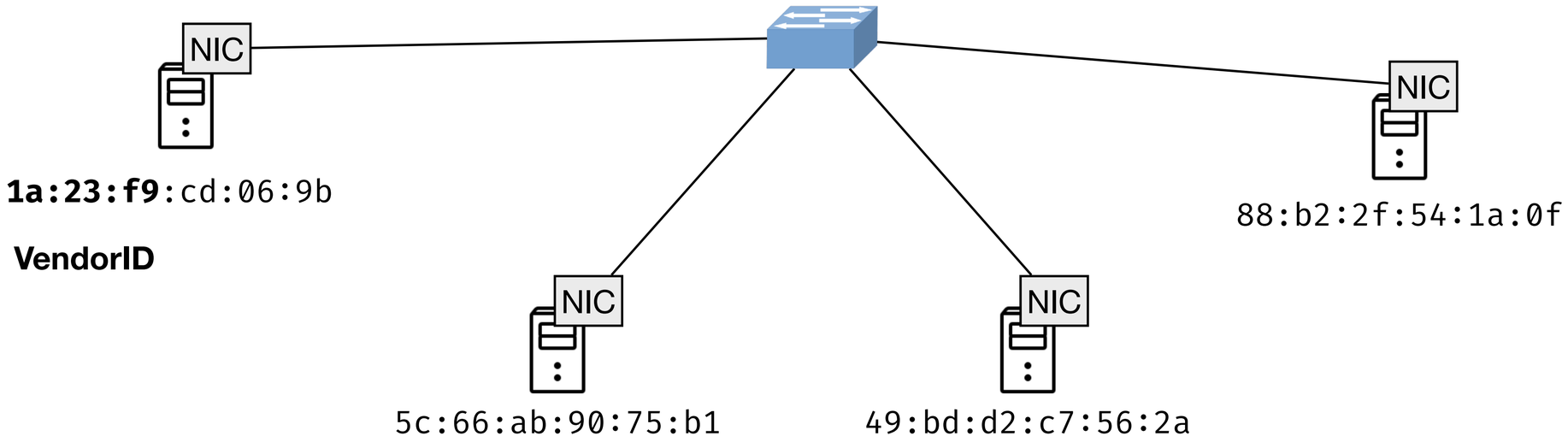


Ethernet MAC address

6-byte long, unique among all network adapters, managed by IEEE

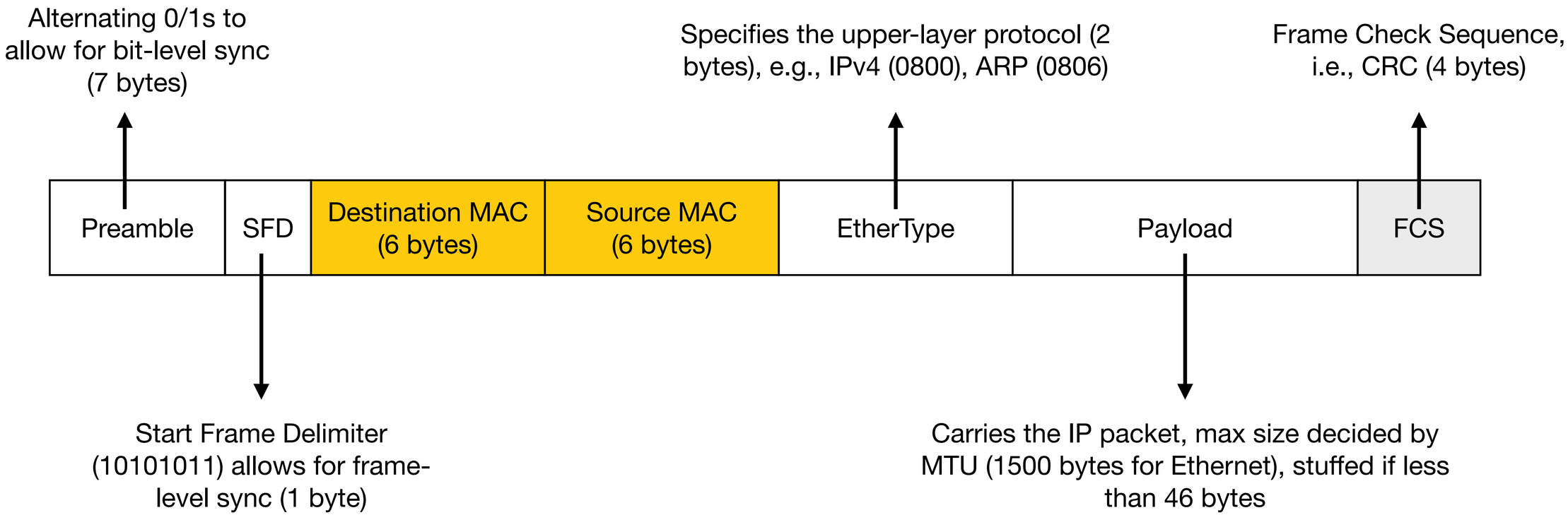
Broadcast MAC address
`ff:ff:ff:ff:ff:ff`

Do switches need MAC addresses? Why?



Ethernet frame structure

IEEE 802.3

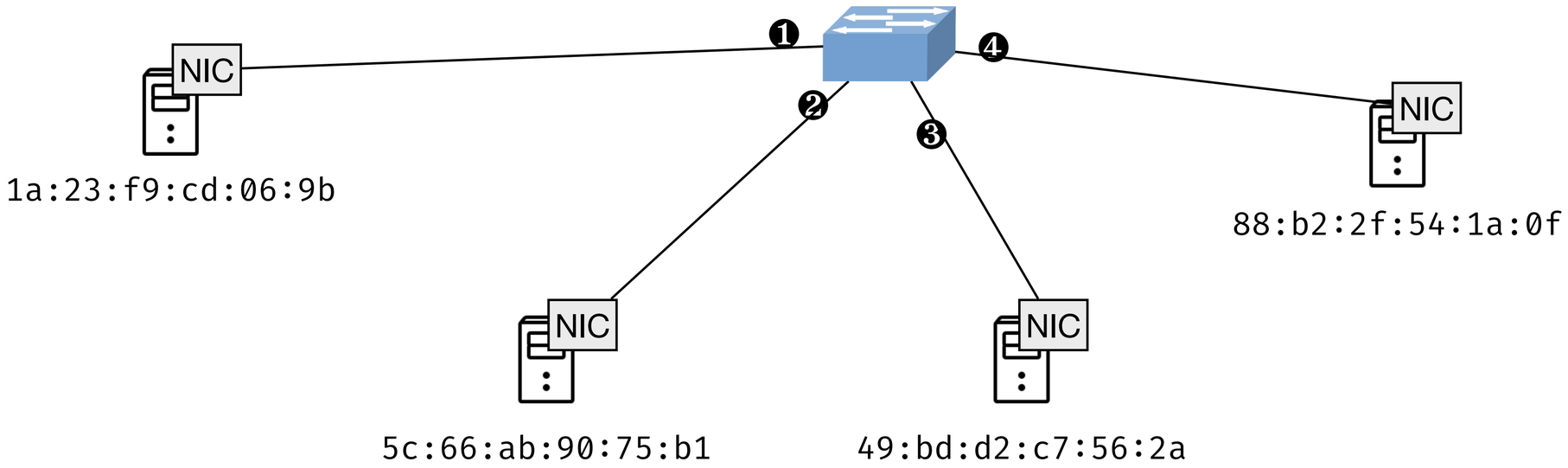


Link layer switches

Switches forward/broadcast/drop frames based on a switch table (a.k.a. forwarding table) and operate transparently to the hosts, i.e., no need for MAC addresses on them

MAC	Interface	Time
88:b2:2f:54:1a:0f	4	9:32
5c:66:ab:90:75:b1	2	9:34

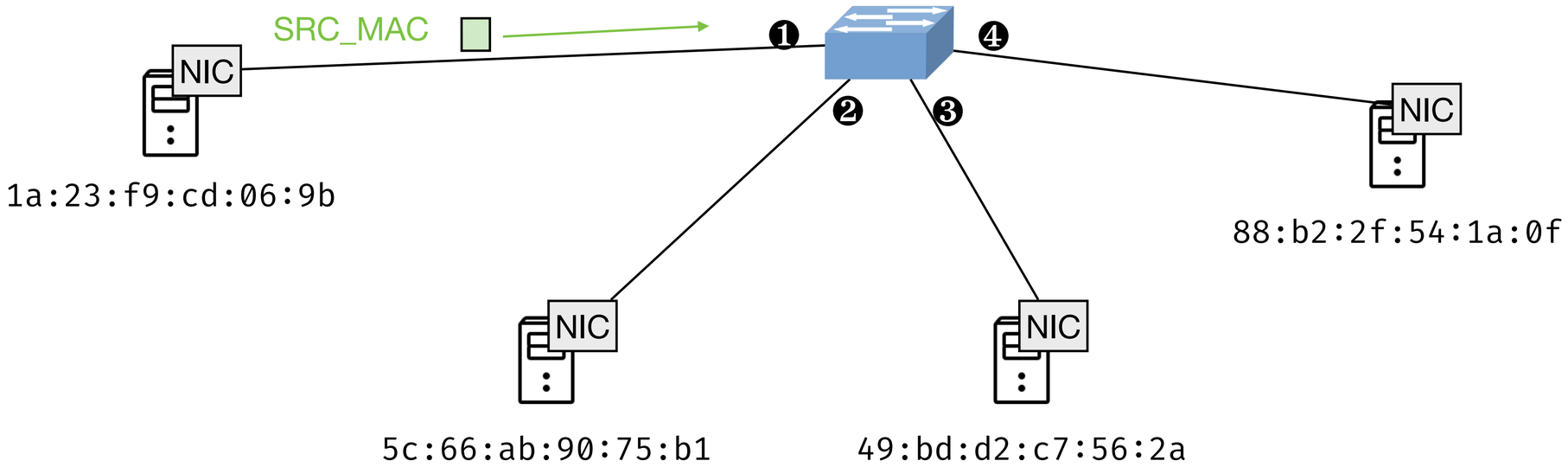
How to configure the forwarding table?



Self learning

Learn new MAC-interface mappings through incoming frames

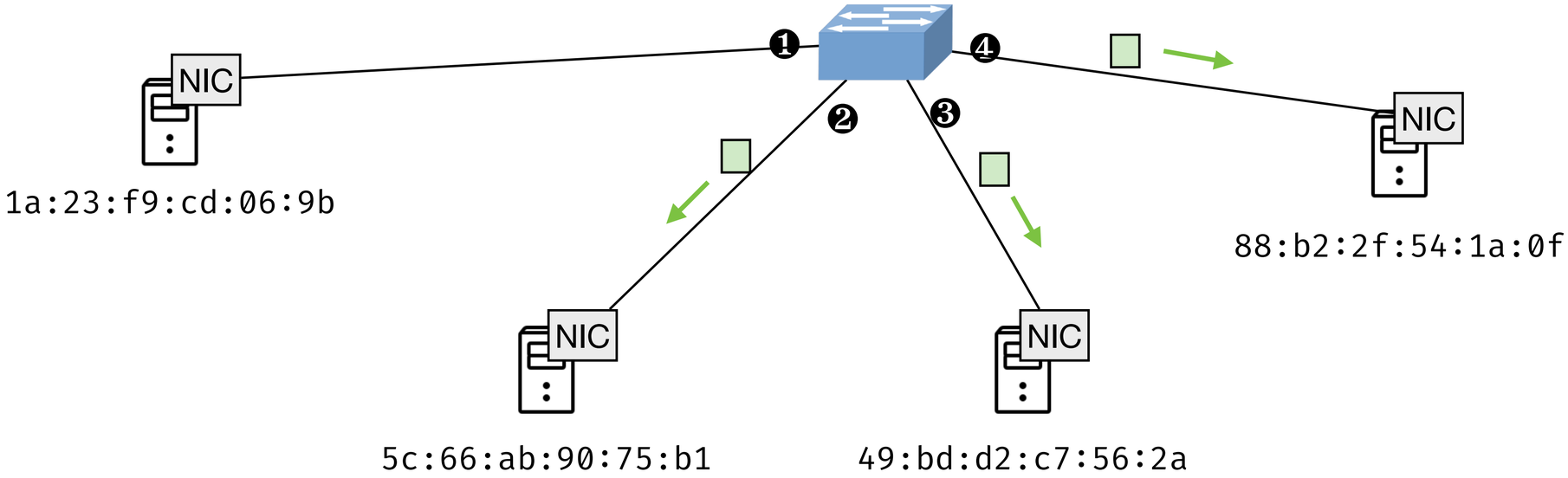
MAC	Interface	Time
88:b2:2f:54:1a:0f	4	9:32
5c:66:ab:90:75:b1	2	9:34
1a:23:f9:cd:06:9b	1	10:00



Self learning

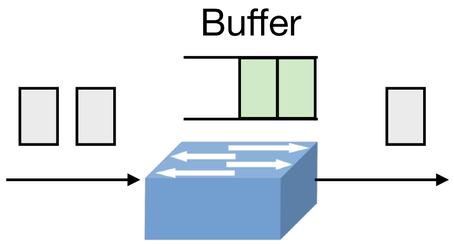
Broadcast the new frame with **unknown destination MAC** on all interfaces
but the one that has received the frame

MAC	Interface	Time
88:b2:2f:54:1a:0f	4	9:32
5c:66:ab:90:75:b1	2	9:34
1a:23:f9:cd:06:9b	1	10:00



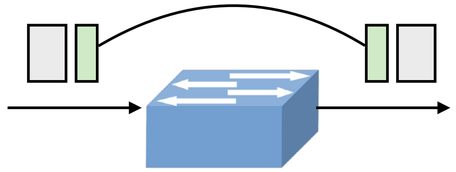
Store-and-forward vs. cut-through

Store-and-forward



Packets are received in full, buffered, and forwarded onto the output link.

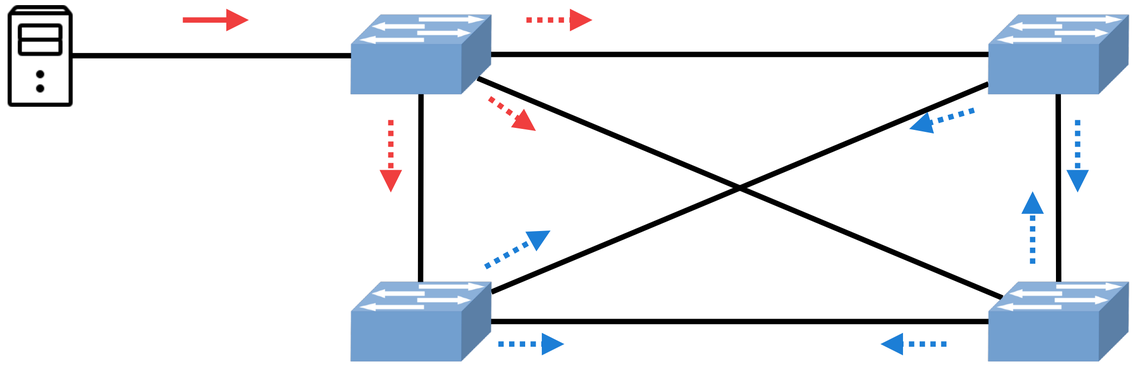
Cut-through



Once lookup is done, packet receiving and sending happen at the same time.

What are the pros and cons of each approach?

Problem #1: when flooding meets loops



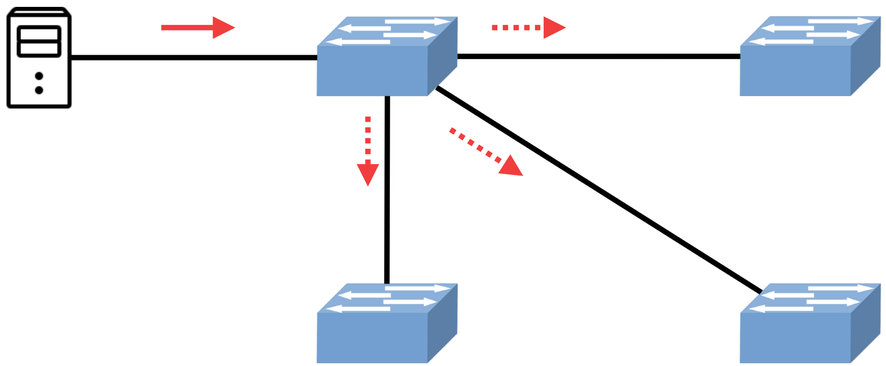
Each frame leads to the creation of at least two new frames.
Exponential increase, with no TTL to remove looping frames...

Redundancy without loops

Solution

- Reduce the network to one logical spanning tree
- Upon failure, automatically rebuild a spanning tree

In practice, switches run a distributed spanning tree protocol (STP)

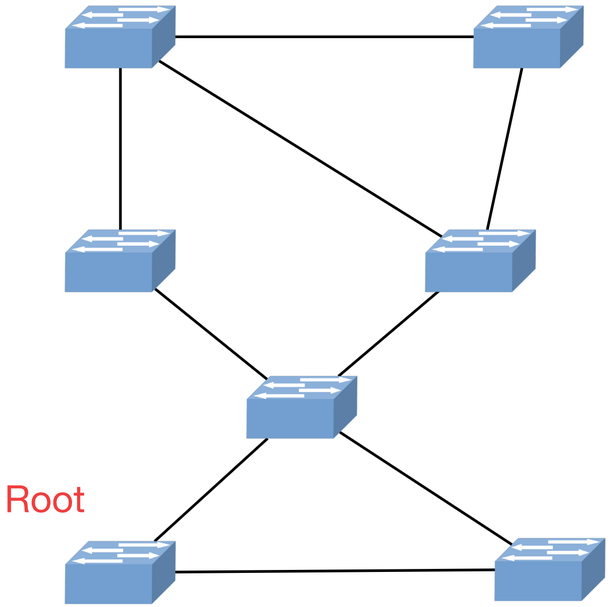


Algorhyme

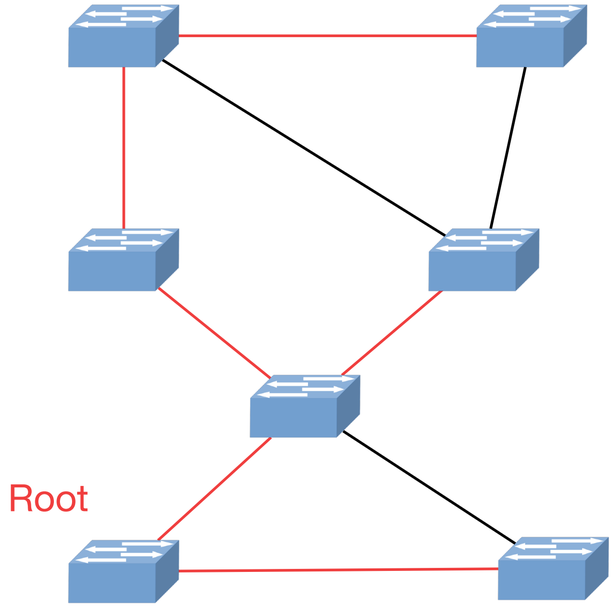
I think that I shall never see a graph more
lovely than a tree.
A tree whose crucial property is loop-free
connectivity.
A tree that must be sure to span so
packets can reach every LAN.
First, the root must be selected.
By ID, it is elected.
Least-cost paths from root are traced.
In the tree, these paths are placed.
A mesh is made by folks like me, then
bridges find a spanning tree.

— *Radia Perlman*

STP example



Select the root



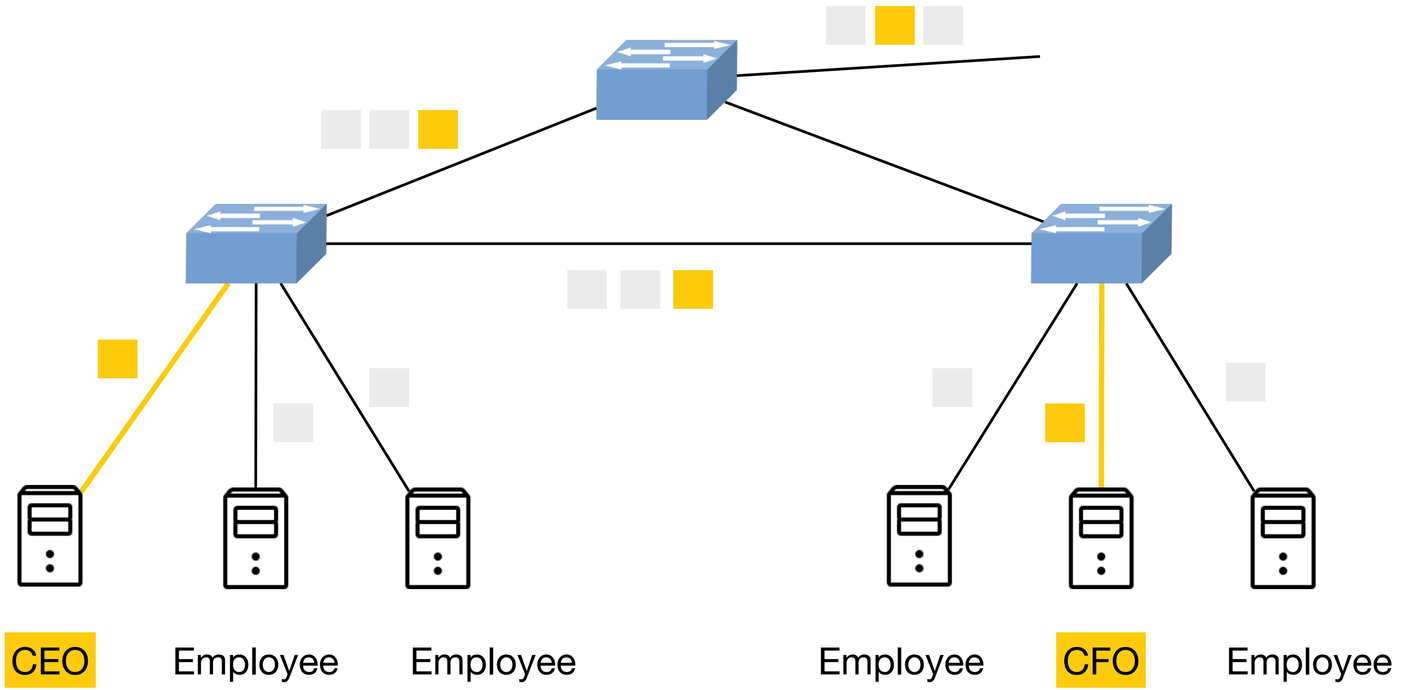
Keep shortest paths to root

To ensure robustness, the root switch keeps sending the messages. If timeout, switches claim itself to be root.

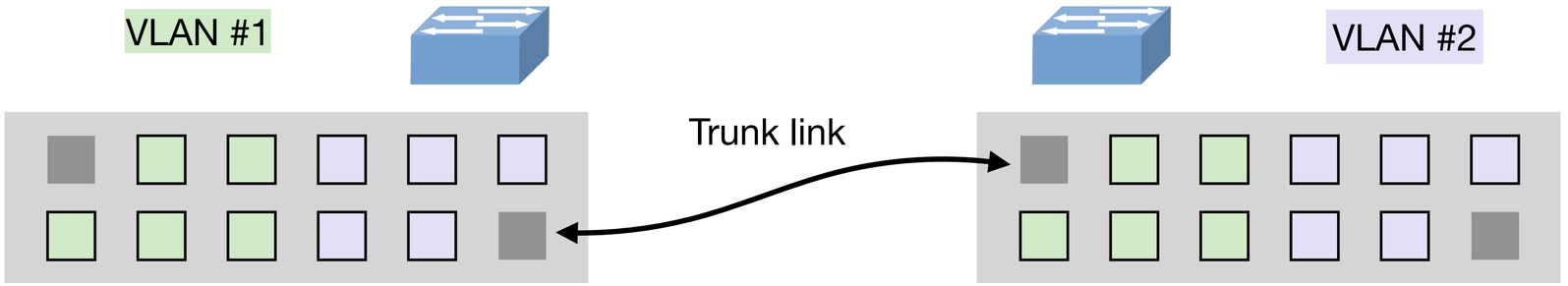
Problem #2: traffic isolation

Broadcast packets cannot be localized and can cause broadcast storm in the network

Hard user management: A user has to be connected to the a particular switch in order to isolate its traffic



VLAN

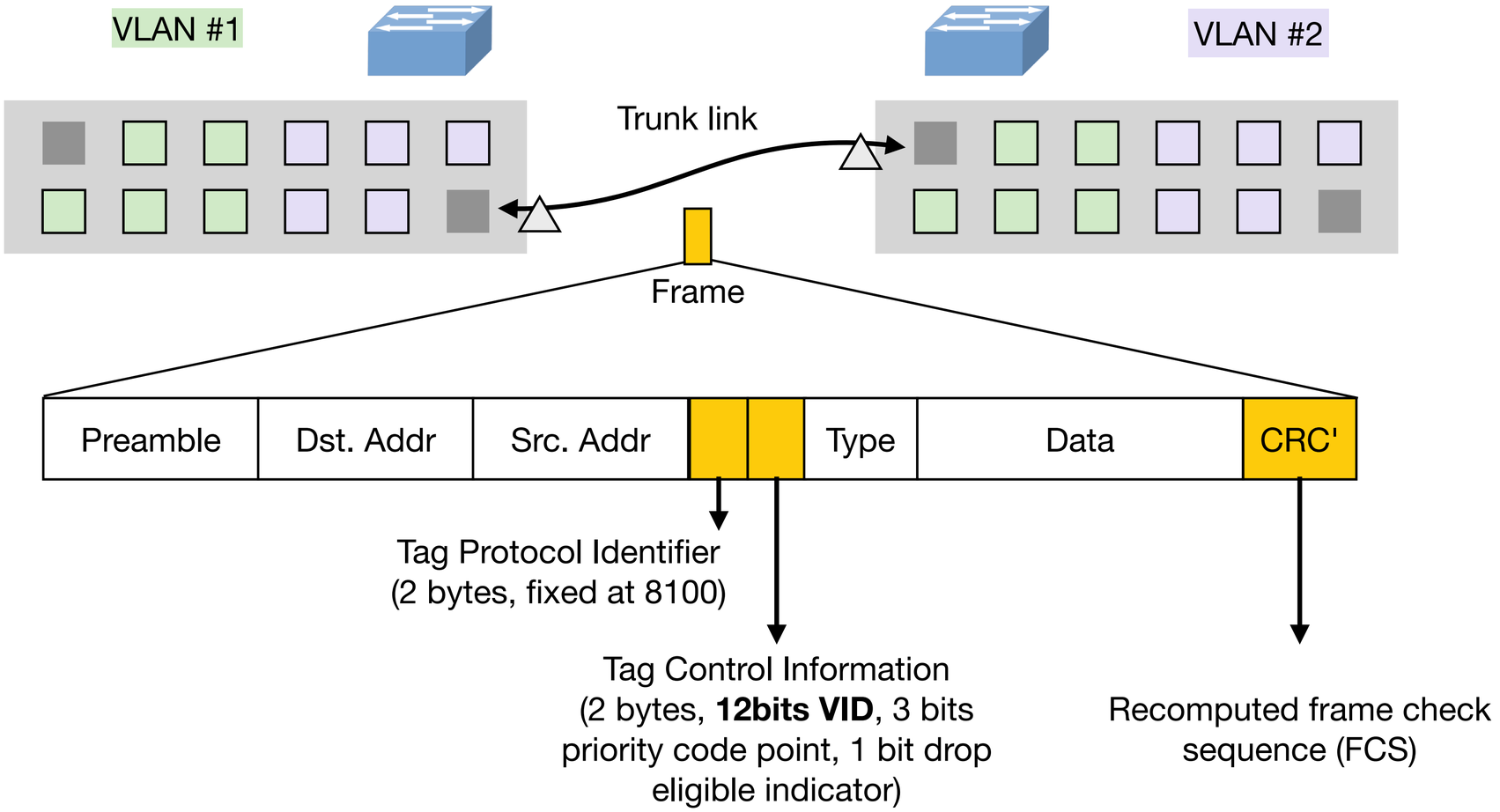


1. Network manager can partition the ports into subsets and assign them to VLANs
2. Ports in the same VLAN form a broadcast domain, while ports on different VLANs are routed through an internal router within the switch
3. Switches are connected on trunk ports that belong to all VLANs

How does a receiving switch know which VLAN a frame belongs to?

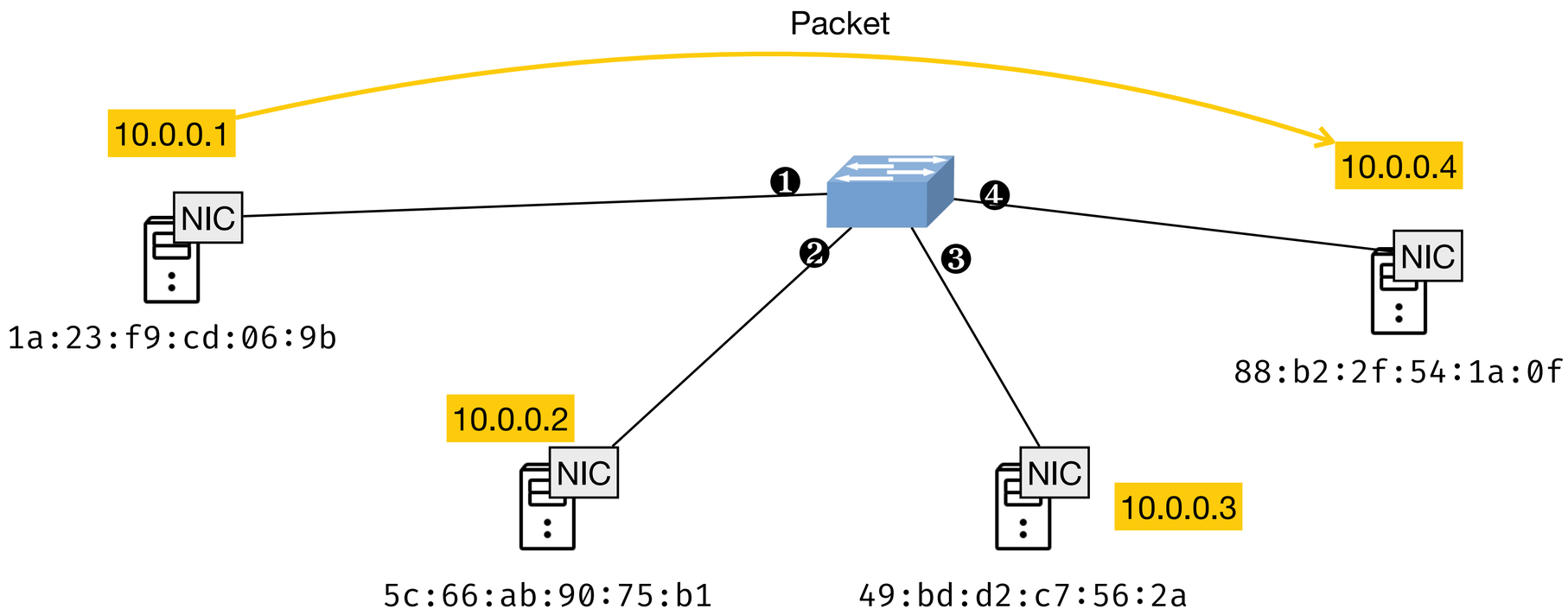
VLAN tag

IEEE 802.1Q



How to obtain the destination MAC address?

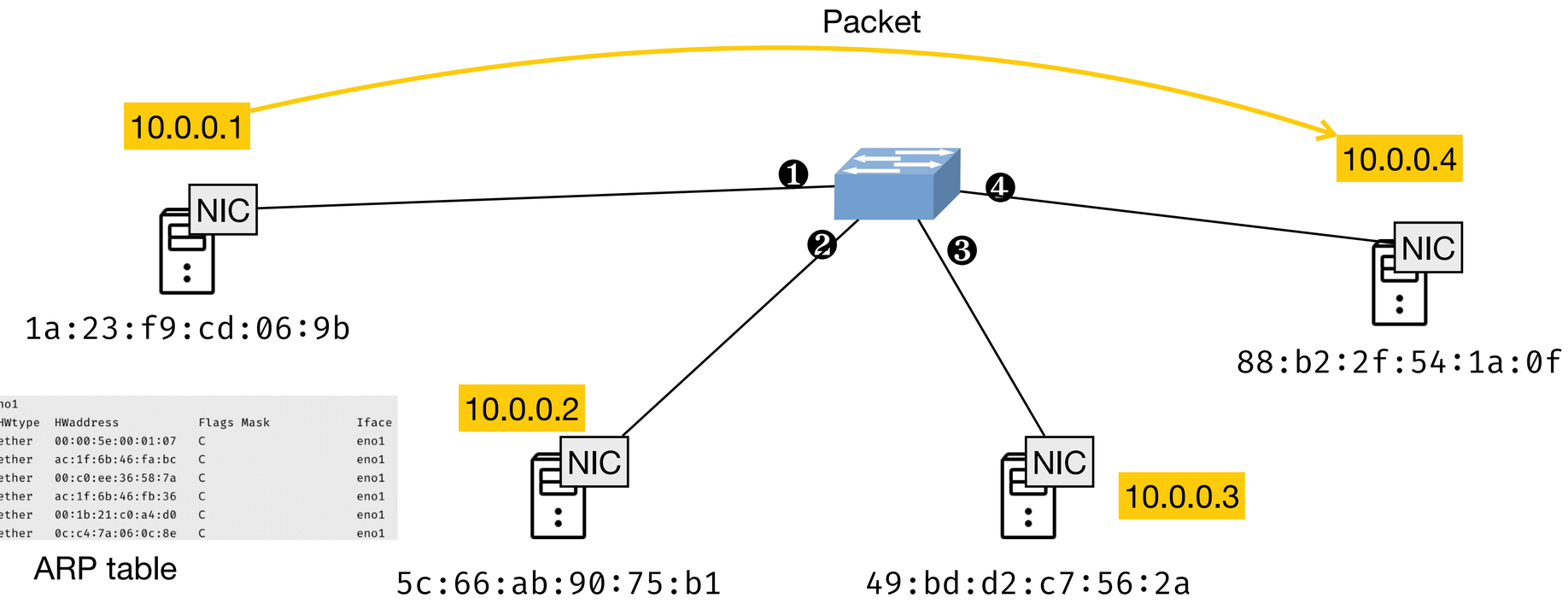
Assume we want to send a packet from `10.0.0.1` to `10.0.0.4` on the same subnet. The first step is to know where to forward the packet (or more precisely the frame containing the packet), i.e., obtaining the MAC address of the destination.



How to obtain the destination MAC address?

RFC 826

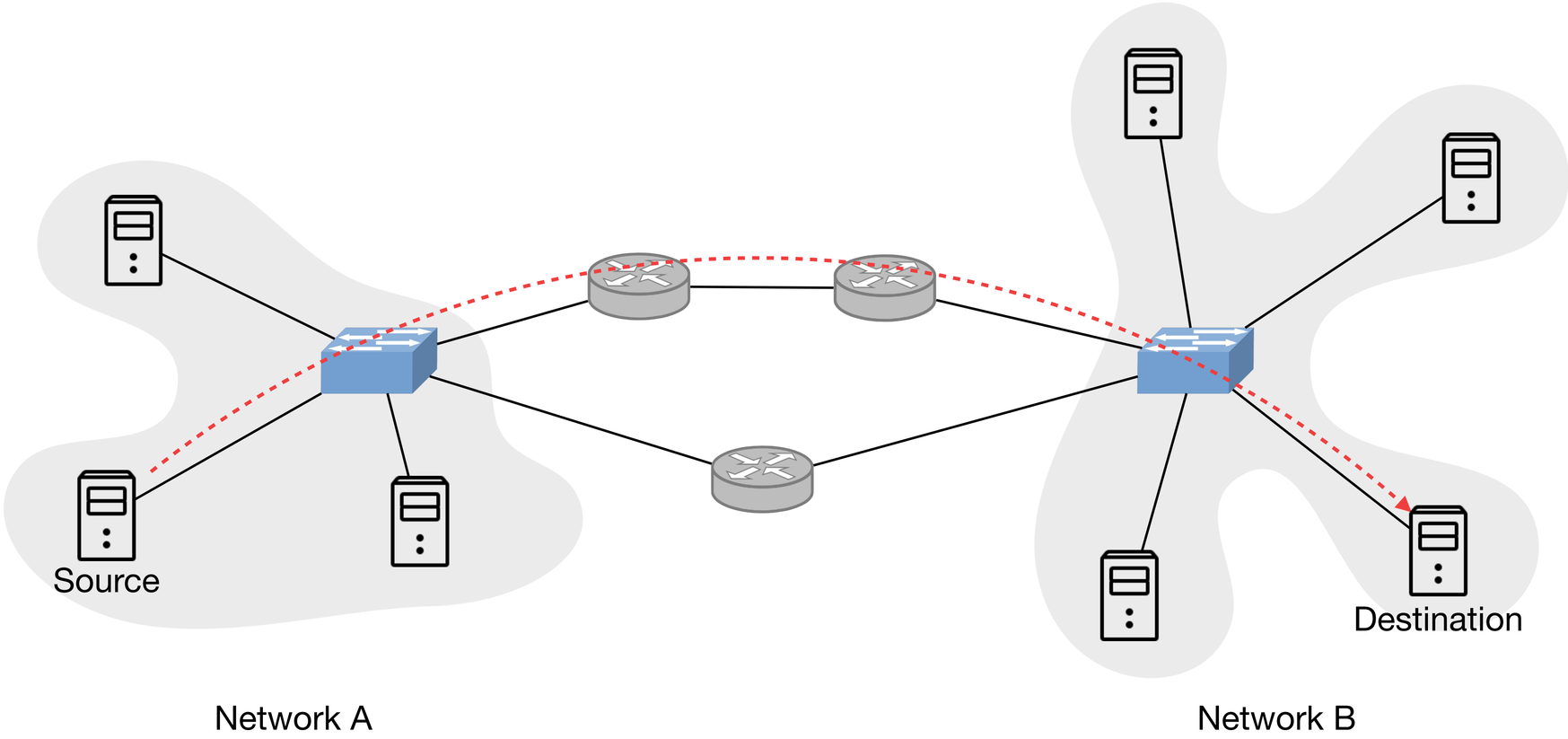
ARP query: Whoever has the IP address `10.0.0.4`, please tell me your MAC address
ARP reply: that is me, my MAC address is `88:b2:2f:54:1a:0f`




```
wang@erdos:~$ arp -n -i eno1
Address                  HWtype  HWaddress           Flags Mask            Iface
130.83.163.254           ether   00:00:5e:00:01:07   C                     eno1
130.83.163.133           ether   ac:1f:6b:46:fa:bc   C                     eno1
130.83.163.200           ether   00:c0:ee:36:58:7a   C                     eno1
130.83.163.96            ether   ac:1f:6b:46:fb:36   C                     eno1
130.83.163.253           ether   00:1b:21:c0:a4:d0   C                     eno1
130.83.163.19            ether   0c:c4:7a:06:0c:8e   C                     eno1
```

ARP table

An example



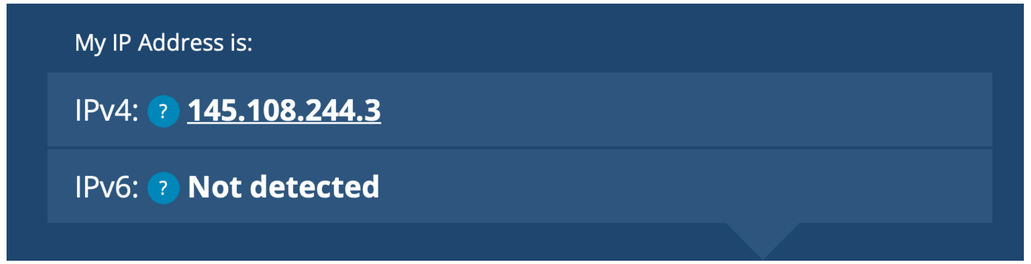
Network address translation (NAT)

Translating between internal and external network addresses

NAT example

```
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=6463<RXCSUM,TXCSUM,TSO4,TSO6,CHANNEL_IO,PARTIAL_CSUM,ZEROINVERT_CSUM>
        ether 3c:22:fb:0c:7b:b6
        inet6 fe80::87:47d2:32cd:873e%en0 prefixlen 64 secured scopeid 0x7
        inet 10.0.0.200 netmask 0xffffff00 broadcast 10.0.0.255
        nd6 options=201<PERFORMNUD,DAD>
        media: autoselect
        status: active
```

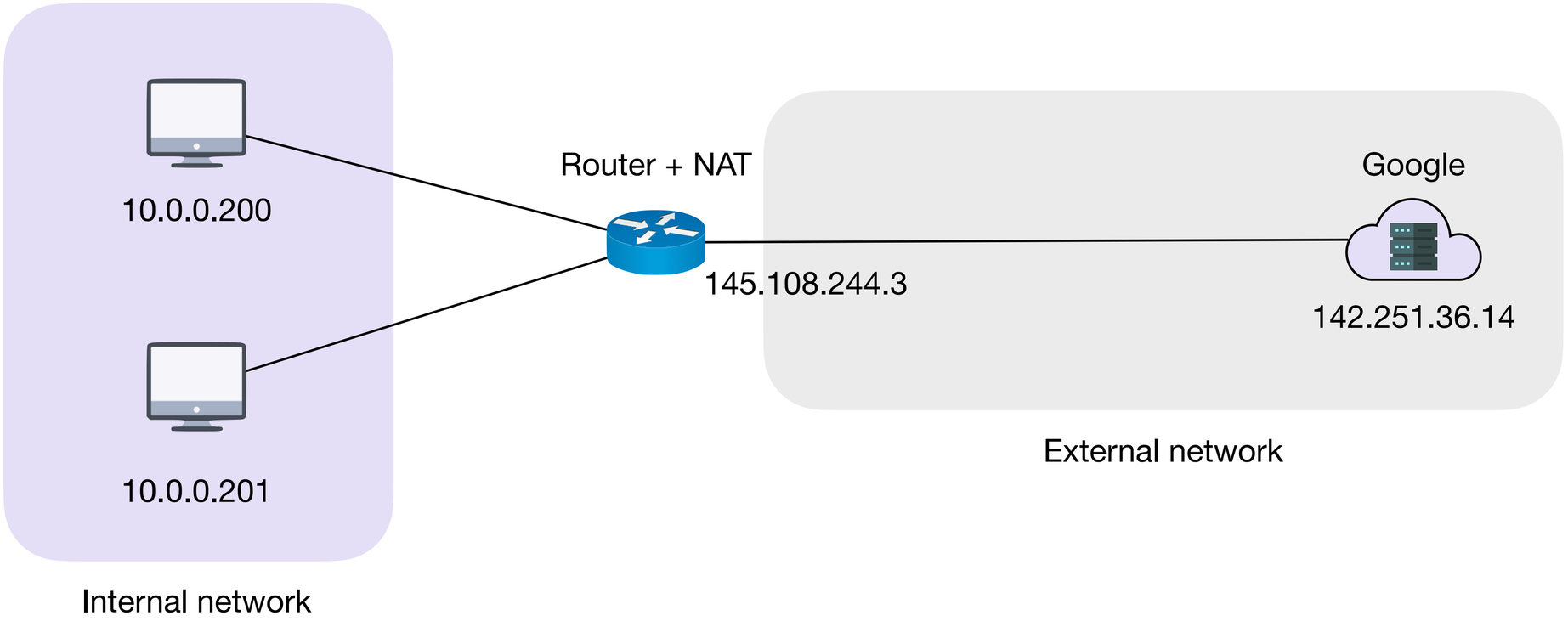
IP as you see from your computer: 10.0.0.200



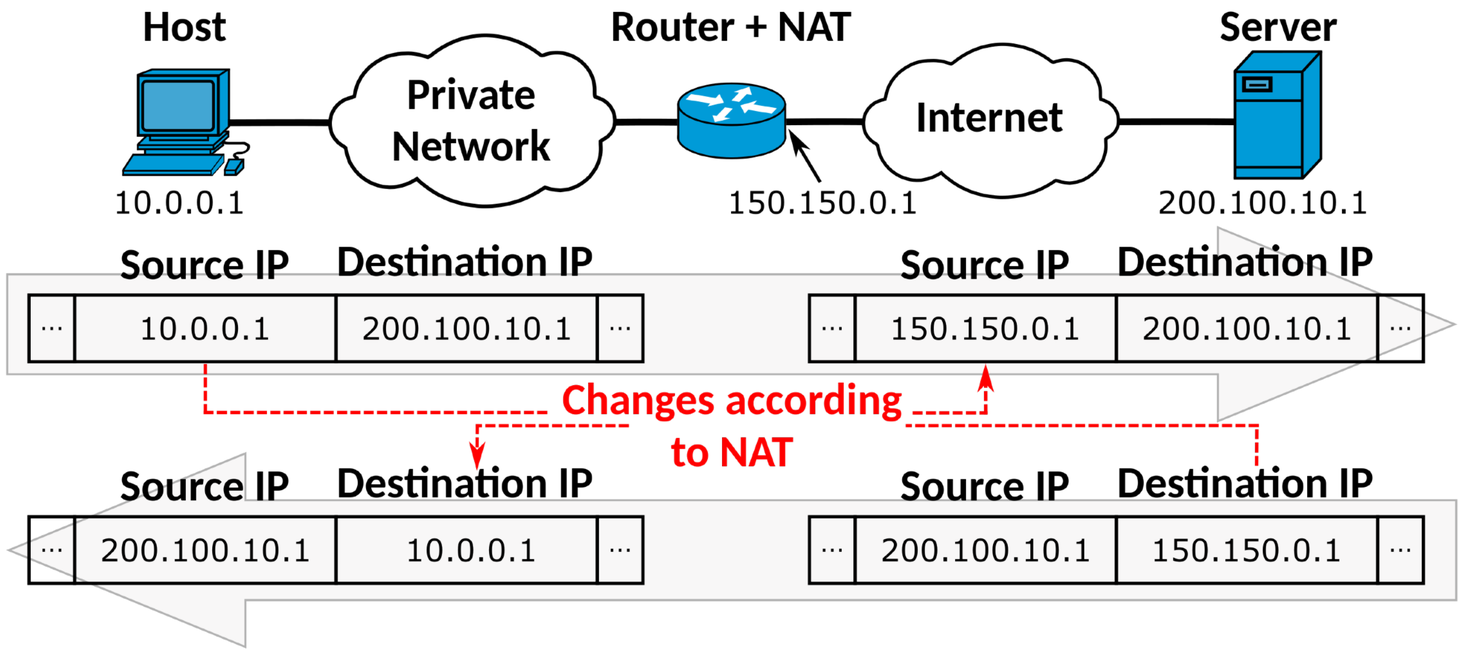
IP seen from outside: 145.108.244.3

Network address translation (NAT)

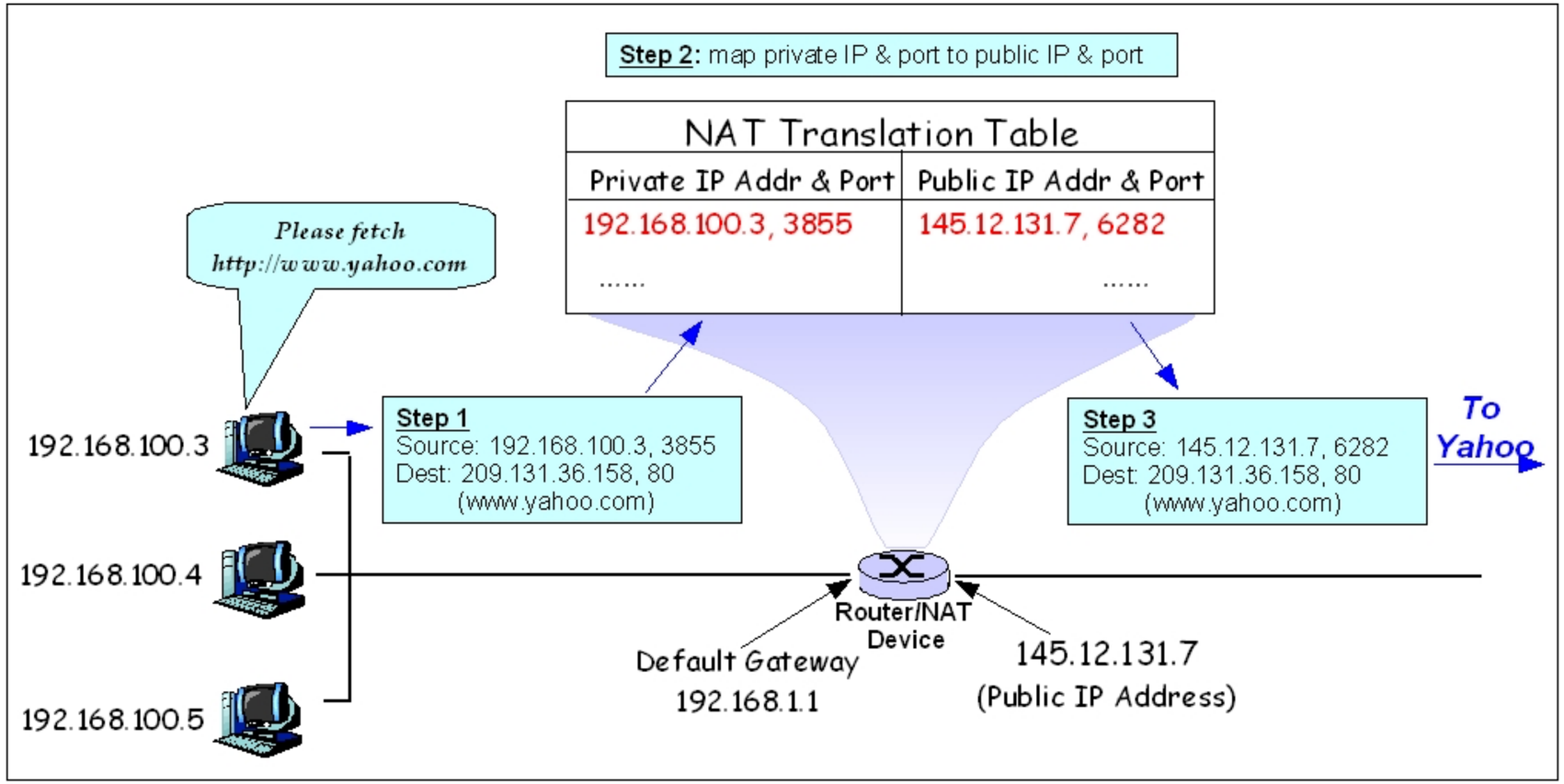
A method of mapping an IP address space into another, used for masking network changes and mitigating IPv4 address exhaustion



NAT: simplest case



NAT's working



NAT pros and cons

Mitigates IPv4 address exhaustion problem: reuse IPv4 addresses in private networks

Destination NAT for port forwarding: hiding internal servers, load balancing

Hard to establish peer-to-peer connections

Violates the end-to-end principle!

Next lecture: networking algorithms and data structures