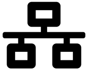
Network Basics

TCP/IP Networking

Richard Berger, 2021

Network Range



Local Area Network (LAN)

- cable based
- limited area
- Ethernet

Wireless Local Area Network (WLAN)

- wireless
- limited area
- WiFi

Wide Area Network (WAN)

- spans large geographical distances
- Dial-Up, DSL, Cable, Fiber, 4G, 5G

Network Layers

Basic Reference Model for Open Systems Interconnect (OSI model)

- A conceptional model that characterizes and standardizes how network communication works

7	Application Layer
6	Presentation Layer
5	Session Layer
4	Transport Layer
3	Network Layer
2	Data Link Layer
1	Physical Layer

Application Layer

7 Application Layer
6 Presentation Layer
5 Session Layer
4 Transport Layer
3 Network Layer
2 Data Link Layer
1 Physical Layer

- Applications use high-level network protocols such as HTTP, HTTPS, FTP, SMTP, IMAP, SSH, Telnet to talk to servers
- think of these protocols as something like a *language* to speak to a server

Application Layer Protocols

SSH

- Remote terminal

HTTP/S

- Firefox
- Chrome
- Safari

IMAP, SMTP

- Outlook
- Thunderbird
- Apple Mail

Example: HTTP GET request

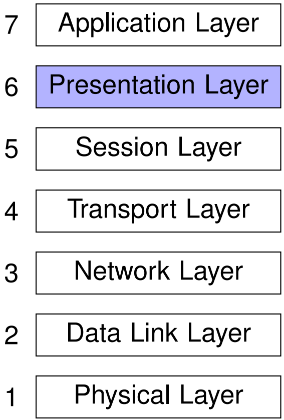
HTTP Client Request

```
GET /index.html HTTP/2
Host: www.example.com
```

HTTP Server Response

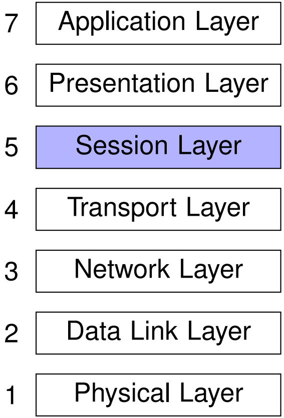
```
HTTP/2 200 OK
date: Mon, 18 Jan 2021 16:30:13 GMT
content-type: text/html; charset=UTF-8
content-encoding: en
server: Apache
content-length: 92

<html>
<head>
  <title>Example</title>
</head>
<body>
  <p>Hello World!</p>
</body>
</html>
```

Presentation Layer



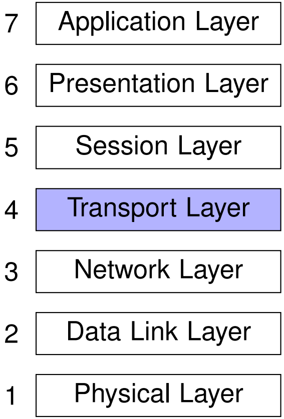
- the presentation layer is responsible for transforming data between application and the lower layers
- data in the application layer may use a different data encoding than the lower layers. E.g. text could be **encoded** or **decoded** into different formats, such as ASCII to UTF8 or vice versa.
- data could be **compressed** or **decompressed**
- data could be **encrypted** or **decrypted**

Session Layer



- communication partners will sometimes want to know about each other over a longer period of time, even if connection is interrupted
- another need for sessions is authentication and authorization
- for some network applications you will have to first authenticate to gain access to a resource, such as files on a network folder
- the session is what ensures you don’t have to authenticate every time you make a new request

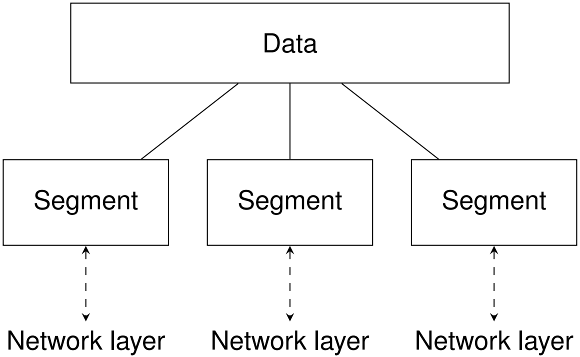
Transport Layer



- no matter what application protocol is being used, or how data is encoded, compressed, encrypted, or whether the application keeps track of a session or not, all data goes through the **transport layer**
- the transport layer is what allows multiple applications to use one network connection simultanously
- the most commonly known transport protocols are
  - User Datagram Protocol (UDP)
  - Transmission Control Protocol (TCP)

Transport Layer: Segmentation

- the transport layer splits the data it should send into TCP segments or UDP datagrams and send or receives them via the network layer



Transport Layer: Port Ranges

- The sharing of a network connection is done by providing multiple ports **per network connections**
- UDP and TCP each provides **65535 ports** per network connection
- Ports are split up three ranges

0-1023: Reserved for privileged services and well-known services, managed by Internet Assigned Numbers Authority (IANA)

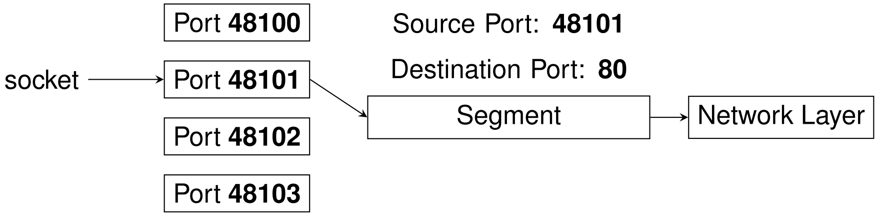
1024-49151: User or registered ports

49152-65535: Dynamic (ephemeral) or private ports

Port	Protocol
20	File Transfer Protocol (FTP) Data Transfer
22	Secure Shell (SSH) Secure Login
25	Simple Mail Transfer Protocol (SMTP) E-mail routing
53	Domain Name System (DNS) service
80	Hypertext Transfer Protocol (HTTP)
123	Network Time Protocol (NTP)
443	HTTP Secure (HTTPS) HTTP

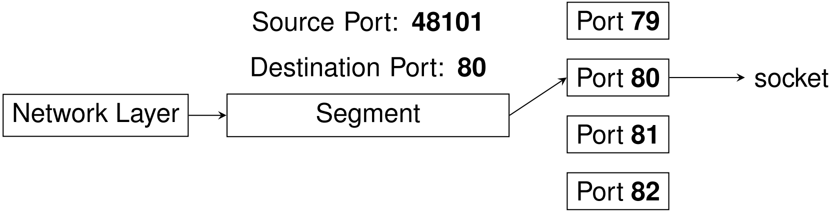
Sender

- applications use ports by creating a **socket** that binds to them
- the source port and destination port do not have to be the same
- servers usually use a fixed port, determined by what service they provide
- clients use any port that is available
- source and destination port are added to the datagram/segment header
- the segment/datagram is then forwarded to the network layer



Receiver

- datagrams/segments received by the network layer are forwarded to the transport layer
- the transport layer forwards the reassembled data to the destination port
- a server application could be listening to this port, which is again a socket bound to that port for a specific network connection using a specific transport protocol.



Transport Protocols

UDP

TCP

User Datagram Protocol (UDP)

connectionless

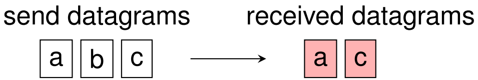
- no need to first establish communication with other side
- send data immediately

light-weight

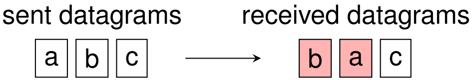
- no connection overhead
- UDP header is 8 bytes ($<$ 20 bytes for TCP)
- 16bit checksum
  - corrupt datagrams are dropped
  - no retransmission!

User Datagram Protocol (UDP)

- dropped datagrams are **not detected**, every datagram is sent **once**



- UDP does not garantee the order of datagrams



- there is no congestion control $\rightarrow$ more dropped datagrams on congested connections

Transmission Control Protocol (TCP)

connection-based

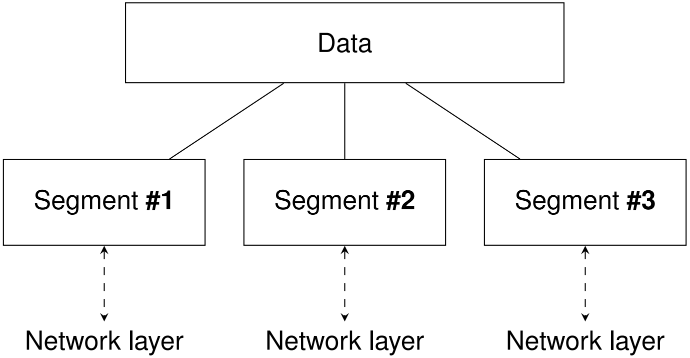
- establish and synchronize communication before sending data
- data can not be sent until connection is established

robust

- adds overhead, but more reliable
- larger header (20 bytes) than UDP (8 bytes)
- same 16bit checksum
- all segments have a 32bit **sequence number**

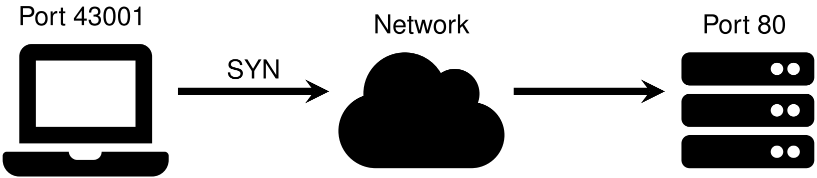
TCP Segments

- TCP can **reconstruct** the correct order of segments
- and **detect dropped** segments
- this allows to **retransmit segments** if needed



Establishing a TCP Connection

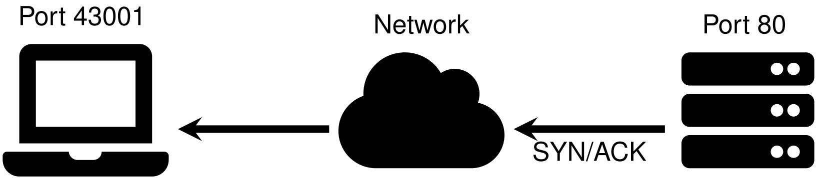
- before data is transmitted, the two hosts perform a so-called **three-way handshake** to initiate a connection



1. The client sends that it wants to establish a connection (SYN) and that it will use the attached sequence number

Establishing a TCP Connection

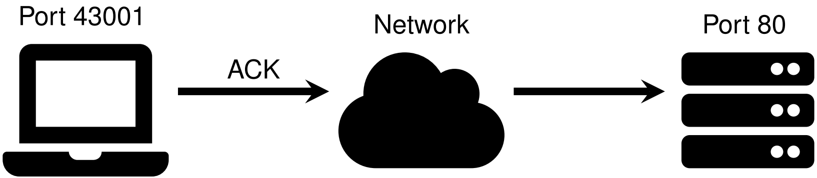
- before data is transmitted, the two hosts perform a so-called **three-way handshake** to initiate a connection



1. The client sends that it wants to establish a connection (SYN) and that it will use the attached sequence number
2. The server will respond with SYN and ACK bits set, returning its own sequence number

Establishing a TCP Connection

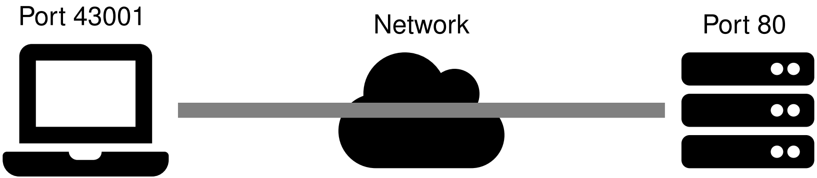
- before data is transmitted, the two hosts perform a so-called **three-way handshake** to initiate a connection



1. The client sends that it wants to establish a connection (SYN) and that it will use the attached sequence number
2. The server will respond with SYN and ACK bits set, returning its own sequence number
3. Finally the client acknowledges with ACK with its next sequence number, which establishes the connection

Establishing a TCP Connection

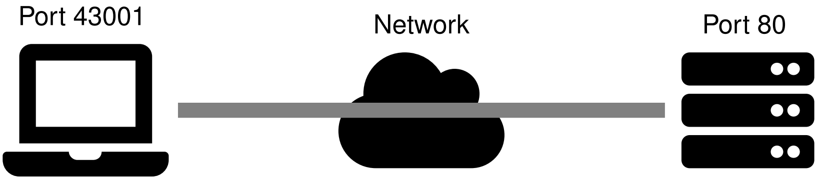
- before data is transmitted, the two hosts perform a so-called **three-way handshake** to initiate a connection



1. The client sends that it wants to establish a connection (SYN) and that it will use the attached sequence number
2. The server will respond with SYN and ACK bits set, returning its own sequence number
3. Finally the client acknowledges with ACK with its next sequence number, which establishes the connection

Establishing a TCP Connection

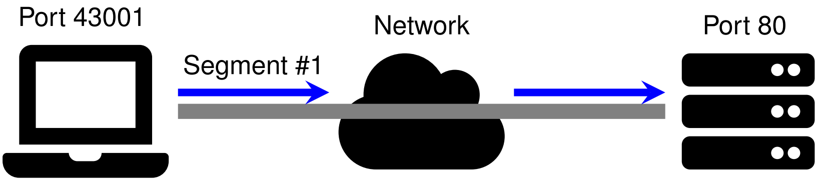
- before data is transmitted, the two hosts perform a so-called **three-way handshake** to initiate a connection



1. The client sends that it wants to establish a connection (SYN) and that it will use the attached sequence number
2. The server will respond with SYN and ACK bits set, returning its own sequence number
3. Finally the client acknowledges with ACK with its next sequence number, which establishes the connection

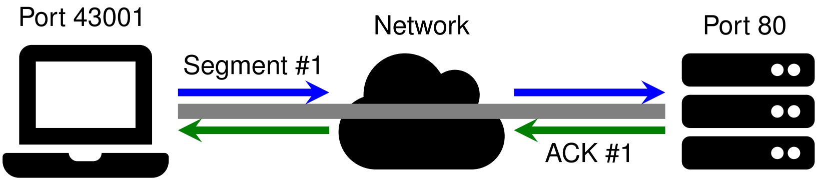
- something similar happens for closing the connection

Receiving TCP Segments



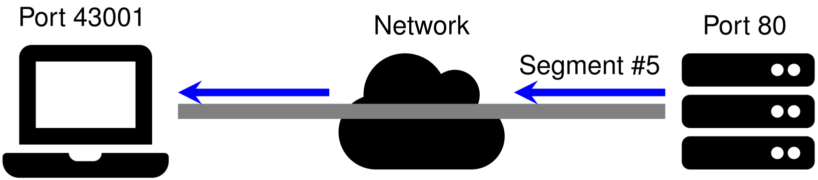
- any segment that is transmitted must be **acknowledged** by the other side
- if the sender does not receive an acknowledgment after a certain time, it assumes the segment was lost
- $\Rightarrow$ segment will be sent again
- the same happens if the segment checksum is invalid

Receiving TCP Segments



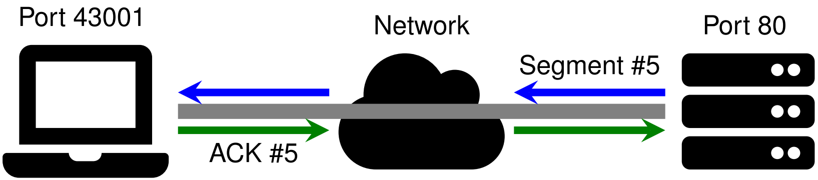
- any segment that is transmitted must be **acknowledged** by the other side
- if the sender does not receive an acknowledgment after a certain time, it assumes the segment was lost
- $\Rightarrow$ segment will be sent again
- the same happens if the segment checksum is invalid

Receiving TCP Segments



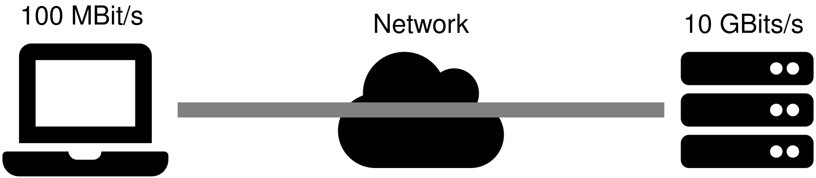
- any segment that is transmitted must be **acknowledged** by the other side
- if the sender does not receive an acknowledgment after a certain time, it assumes the segment was lost
- $\Rightarrow$ segment will be sent again
- the same happens if the segment checksum is invalid

Receiving TCP Segments



- any segment that is transmitted must be **acknowledged** by the other side
- if the sender does not receive an acknowledgment after a certain time, it assumes the segment was lost
- $\Rightarrow$ segment will be sent again
- the same happens if the segment checksum is invalid

TCP Flow Control



- if one side is sending data at a rate higher than what the receiver can handle, or if there is congestion on the network, packets will be dropped
- with TCP the connection can slow down or increase the transmission rate to adjust to connectivity and traffic congestion

Network Layer

- 7 Application Layer
- 6 Presentation Layer
- 5 Session Layer
- 4 Transport Layer
- 3 Network Layer
- 2 Data Link Layer
- 1 Physical Layer

- the network layer is responsible of transmitting data from one host to another, even if they are located in different networks
- protocols create and work with **packets**

Network protocols provide

Network protocols provide

logical addressing

every host has a unique
address in its network

Network protocols provide

logical addressing

every host has a unique address in its network

routing

packets from one network can redirected to another

Network protocols provide

logical addressing

every host has a unique address in its network

routing

packets from one network can redirected to another

path determination

determine the best path/route to reach a destination

Internet Protocol - Version 4 (IPv4)

Internet Protocol - Version 4 (IPv4)

- **IPv4 addresses** are 32-bit (4 byte), usually expressed as a dotted quad, e.g. 192.168.1.10, each number being from 0-255 (one byte)

192 . 168 . 1 . 10

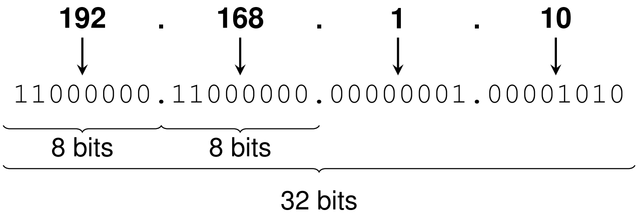
↓ ↓ ↓ ↓

`11000000.11000000.00000001.00001010`

32 bits

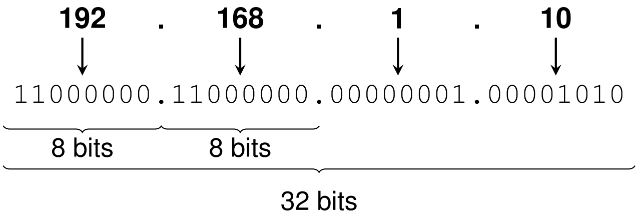
Internet Protocol - Version 4 (IPv4)

- **IPv4 addresses** are 32-bit (4 byte), usually expressed as a dotted quad, e.g. 192.168.1.10, each number being from 0-255 (one byte)



Internet Protocol - Version 4 (IPv4)

- **IPv4 addresses** are 32-bit (4 byte), usually expressed as a dotted quad, e.g. 192.168.1.10, each number being from 0-255 (one byte)



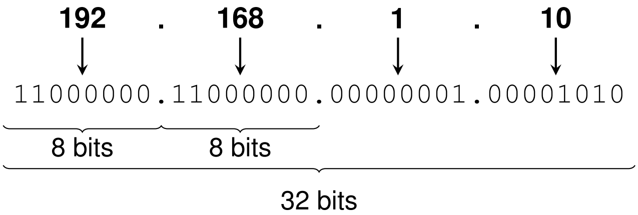
- IP addresses consist of two parts

  network number: uniquely identifies the network

  host identifier: a unique number within the network for a host

Internet Protocol - Version 4 (IPv4)

- **IPv4 addresses** are 32-bit (4 byte), usually expressed as a dotted quad, e.g. 192.168.1.10, each number being from 0-255 (one byte)



- IP addresses consist of two parts

  network number: uniquely identifies the network

  host identifier: a unique number within the network for a host

- the number of bits used for each part is determined by the **subnet mask**

Subnet Mask

Specified either using as dotted quad (32bit) or using a **prefix**, which is simply the number of bits allocated to the network portion.

IP Address

192 . 168 . 1 . 10

↓ ↓ ↓ ↓

11000000.11000000.00000001.00001010

Subnet Mask

255 . 255 . 255 . 0

↓ ↓ ↓ ↓

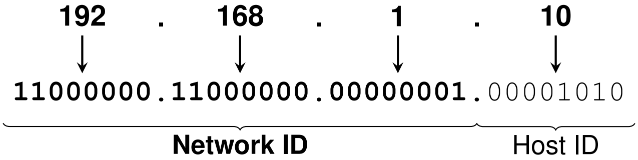
11111111.11111111.11111111.00000000

Prefix = 24 bits

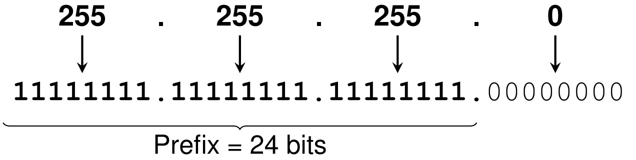
Subnet Mask

Specified either using as dotted quad (32bit) or using a **prefix**, which is simply the number of bits allocated to the network portion.

IP Address



Subnet Mask



Class-less Inter Domain Routing (CIDR) Notation

- the IP and subnet mask can be expressed using class-less inter domain routing (CIDR) notation, which simply adds the prefix after the IP address

Dotted Quad for 24bit netmask

IP: 192.168.1.1

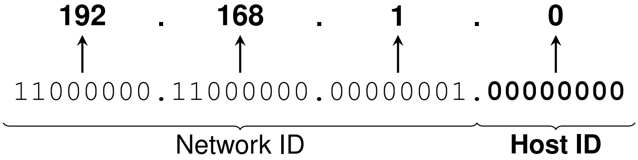
Netmask: 255.255.255.0

CIDR notation

192.168.1.1/24

Specifying an IP Network

- a common way to specify a network is to set the host identifier to 0 and write the IP address in CIDR notation, e.g., 192.168.1.0/24



Dotted Quad for 24bit netmask

IP: 192.168.1.0

Netmask: 255.255.255.0

CIDR notation

192.168.1.0/24

IP Network Range

IP Address

192 . **168** . **1** . **0**

`11000000.11000000.00000001.`**`00000000`**

Network ID: `11000000.11000000.00000001`

Host ID: `00000000`

$2^8 = 256$

possible values

Subnet Mask

255 . **255** . **255** . **0**

`11111111.11111111.11111111.`**`00000000`**

Prefix = 24 bits

8 bits

IP Network Range

IP Address

192 . **168** . **1** . **0**

11000000.11000000.00000001.**00000000**

Network ID | **Host ID**

$2^8 = 256$ possible values

Subnet Mask

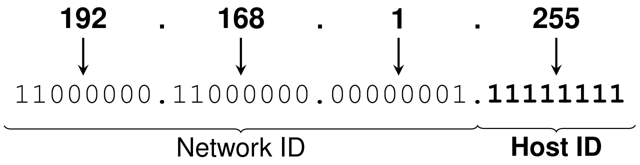
255 . **255** . **255** . **0**

11111111.11111111.11111111.**00000000**

Prefix = 24 bits | **8 bits**

Broadcast Address

- the broadcast address of a network is a special address that can be used to send a packet to all hosts of a network
- it is obtained by setting the host ID to all 1s. For the 192.168.1.0/24 network, the broadcast address is 192.168.1.255.



IPv4 Address Space

- In total IPv4 provides 2^{32} unique IP addresses

4,294,967,296 IP addresses

- Internet Assigned Number Authority (IANA) manages the IP address space **globally**
- IANA delegates blocks of IPs to **five regional** internet registries (RIR)
- Each RIR controls the assigment to local internet registries (LIR)
- LIRs provide IPs and IP ranges to their customers
- e.g., your internet service provider gives you an IP to access the internet

Special IPs

127.0.0.1 Loopback address, commonly known as localhost

0.0.0.0 - 0.255.255.255 : Unused but reserved addresses. 0.0.0.0 is used to specify ANY IP address of a host when opening a socket, which means it will listen to all network interfaces.

Private Address Space

The Internet Assigned Numbers Authority (IANA) has reserved the following three blocks of the IP address space for private internets:

- 10.0.0.0 - 10.255.255.255 (10.0.0.0/8)
- 172.16.0.0 - 172.31.255.255 (172.16.0.0/12)
- 192.168.0.0 - 192.168.255.255 (192.168.0.0/16)

Addresses out of these ranges can not be routed on the public Internet.

- Use these ranges to create smaller private subnets by using a bigger prefix / subnet mask

Subnets

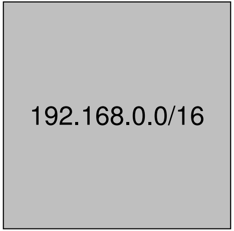
192.168.0.0/16: 1 Network, 2^{16} = 65536 IPs

192 . **168** . **0** . **0**

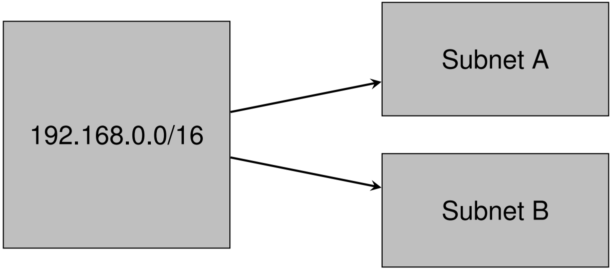
11000000.11000000.00000000.00000000

Network ID (16 bits) | Host ID (16 bits)

IP Range: 192.168.0.0 - 192.168.255.255



Subnets



Subnets

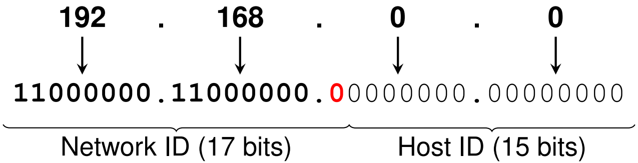
2 subnets out of 192.168.0.0/16 block

- we add one more bit to the network ID

Subnets

2 subnets out of 192.168.0.0/16 block

- we add one more bit to the network ID
- **192.168.0.0/17**: Range: 192.168.0.0 - 192.168.127.255



Subnets

2 subnets out of 192.168.0.0/16 block

- we add one more bit to the network ID
- **192.168.0.0/17**: Range: 192.168.0.0 - 192.168.127.255
- **192.168.128.0/17**: Range: 192.168.128.0 - 192.168.255.255

192 . 168 . 0 . 0

↓ ↓ ↓ ↓

11000000.11000000.00000000.00000000

Network ID (17 bits) | Host ID (15 bits)

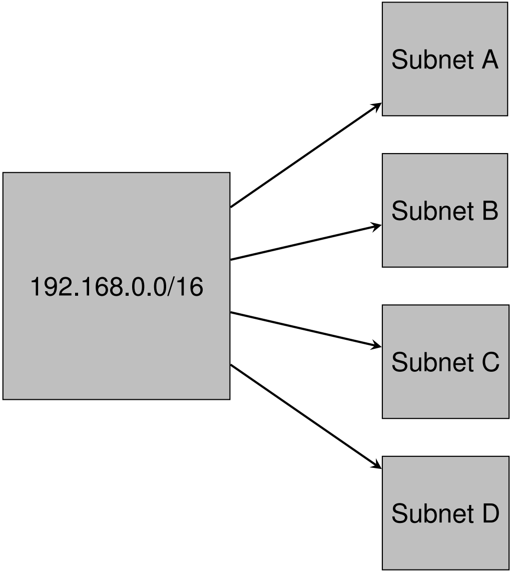
192 . 168 . 128 . 0

↓ ↓ ↓ ↓

11000000.11000000.10000000.00000000

Network ID (17 bits) | Host ID (15 bits)

Subnets



Subnets

4 subnets out of 192.168.0.0/16 block

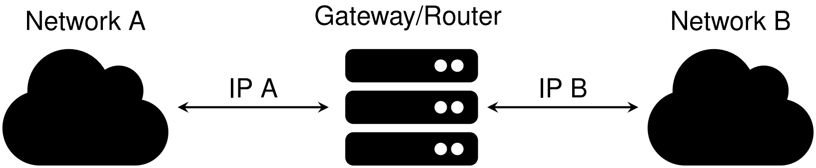
- we add two more bits to subnet mask and therefore the network ID
- **192.168.0.0/18**: Range: 192.168.0.0 - 192.168.63.255
- **192.168.64.0/18**: Range: 192.168.64.0 - 192.168.127.255
- **192.168.128.0/18**: Range: 192.168.128.0 - 192.168.191.255
- **192.168.192.0/18**: Range: 192.168.192.0 - 192.168.255.255

192	.	168	.		.	0

```
11000000.11000000.00000000.00000000
                  01000000
                  10000000
                  11000000
```

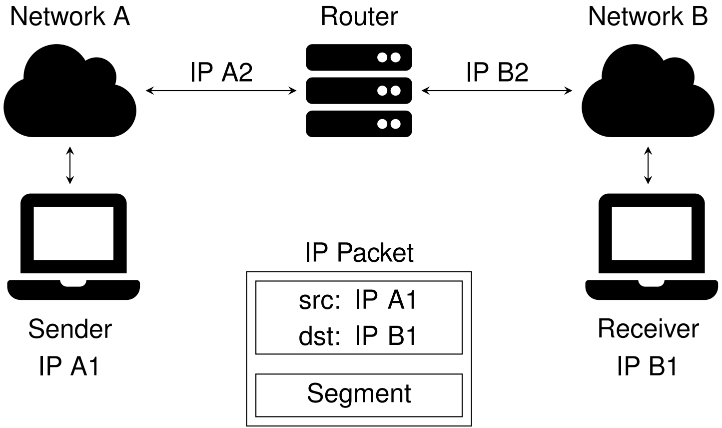
Router / Gateway

- packets move from one network to another via special network hosts called **routers** or **gateways**
- a router is connected to at least two local networks and can forward IP packets from one network to the other
- each router has mulitple IP addresses, one for each network



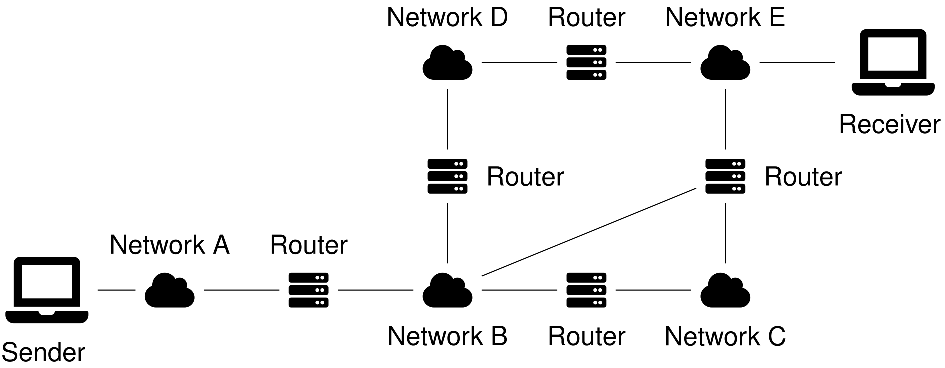
Routing

- IP packets are routed based on rules defined in **routing tables**
- each system has its own routing table
- a **route** determines the next hop of IP packets, based on their destination address
- the **default route**, if set, can forward IP packets to a special router, called the **default gateway**, if no other route is available (e.g., to get to the outside world, the internet)



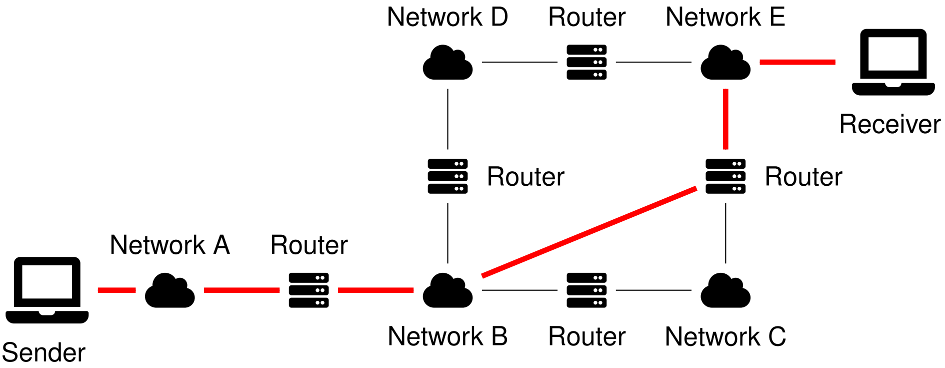
Path Determination

- static routes can define which router should be used to forward a packet
- used to ensure best path to a target network is used
- dynamic protocols: e.g., Open Shortest Path First (OSPF)



Path Determination

- static routes can define which router should be used to forward a packet
- used to ensure best path to a target network is used
- dynamic protocols: e.g., Open Shortest Path First (OSPF)



Data Link Layer

7 Application Layer

6 Presentation Layer

5 Session Layer

4 Transport Layer

3 Network Layer

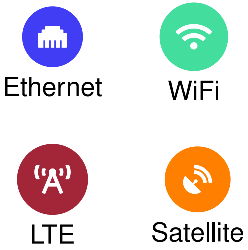
2 Data Link Layer

1 Physical Layer

Data Link Layer

7 Application Layer
6 Presentation Layer
5 Session Layer
4 Transport Layer
3 Network Layer
2 Data Link Layer
1 Physical Layer

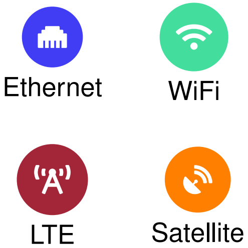
- concerned with putting data on and getting data from **media**



Data Link Layer

7	Application Layer
6	Presentation Layer
5	Session Layer
4	Transport Layer
3	Network Layer
2	**Data Link Layer**
1	Physical Layer

- concerned with putting data on and getting data from **media**



- uses **physical addressing**
- each host on the same medium has a unique physical address

Data Link Layer: Ethernet

- hosts communicate by exchanging **frames**

Data Link Layer: Ethernet

- hosts communicate by exchanging **frames**
- every host is uniquely identified by its media access control (MAC) address

Data Link Layer: Ethernet

- hosts communicate by exchanging **frames**
- every host is uniquely identified by its media access control (MAC) address
- A MAC is 48bit and typically represented as a hexadecimal string, such as

  `02:42:5f : d2:10:ab`

Data Link Layer: Ethernet

- hosts communicate by exchanging **frames**
- every host is uniquely identified by its media access control (MAC) address
- A MAC is 48bit and typically represented as a hexadecimal string, such as

  `02:42:5f : d2:10:ab`

- The MAC address is hard-coded into the Network Interface Controller (NIC), your network device

Data Link Layer: Ethernet

- hosts communicate by exchanging **frames**
- every host is uniquely identified by its media access control (MAC) address
- A MAC is 48bit and typically represented as a hexadecimal string, such as

  `02:42:5f : d2:10:ab`

- The MAC address is hard-coded into the Network Interface Controller (NIC), your network device
- modern NICs allow changing the MAC programmatically

Data Link Layer: Ethernet

- hosts communicate by exchanging **frames**
- every host is uniquely identified by its media access control (MAC) address
- A MAC is 48bit and typically represented as a hexadecimal string, such as

  `02:42:5f` : `d2:10:ab`

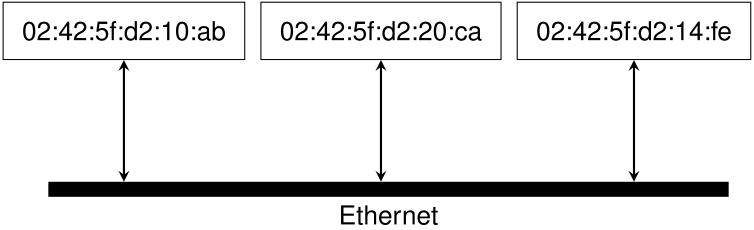
  OUI (02:42:5f) — host part (d2:10:ab)

- The MAC address is hard-coded into the Network Interface Controller (NIC), your network device
- modern NICs allow changing the MAC programmatically

OUI: Organizationally Unique Idendifier (24 bits)

Host: Unique Host Identifier by vendor (24 bits)

Ethernet



- conceptually a single-bus where each host connects to
- in modern Ethernet networks each device connects to a **switch**
- every host can send a frame to any other host
- Ethernet also supports broadcasting a frame to every hosts in a network
- Broadcasts are addressed to the special MAC address `ff:ff:ff:ff:ff:ff`
- For this reason an Ethernet network is also sometimes called *broadcast domain*

Address Resolution Protocol (ARP)

- NICs use MACs for addressing

Address Resolution Protocol (ARP)

- NICs use MACs for addressing
- TCP/IP applications uses IP addresses

Address Resolution Protocol (ARP)

- NICs use MACs for addressing
- TCP/IP applications uses IP addresses
- The Address Resolution Protocol (ARP) is used to bridge the gap and translate MACs to IP addresses

Address Resolution Protocol (ARP)

- NICs use MACs for addressing
- TCP/IP applications uses IP addresses
- The Address Resolution Protocol (ARP) is used to bridge the gap and translate MACs to IP addresses
- ARP assumes all hosts in the same subnet are on the same local network

Address Resolution Protocol (ARP)

- NICs use MACs for addressing
- TCP/IP applications uses IP addresses
- The Address Resolution Protocol (ARP) is used to bridge the gap and translate MACs to IP addresses
- ARP assumes all hosts in the same subnet are on the same local network
- $\Rightarrow$ only hosts in the same subnet can communicate

Address Resolution Protocol (ARP)

- NICs use MACs for addressing
- TCP/IP applications uses IP addresses
- The Address Resolution Protocol (ARP) is used to bridge the gap and translate MACs to IP addresses
- ARP assumes all hosts in the same subnet are on the same local network
- $\Rightarrow$ only hosts in the same subnet can communicate
- an IP packet addressed outside of the subnet, will instead be sent to the default gateway. So ARP will determine the MAC of the gateway IP instead

Address Resolution Protocol (ARP)

ARP Request

To: everybody (ff:ff:ff:ff:ff:ff)
I'm looking for IP: 192.168.0.10
Signed: MAC 02:42:5f:d2:10:ab

- ARP will broadcast and ask who owns a given IP

Address Resolution Protocol (ARP)

ARP Request

To: everybody (ff:ff:ff:ff:ff:ff)
I'm looking for IP: 192.168.0.10
Signed: MAC 02:42:5f:d2:10:ab

- ARP will broadcast and ask who owns a given IP

ARP Response

To: 02:42:5f:d2:10:ab
I have IP: 192.168.0.10
Signed: MAC 02:42:5f:d2:20:ca

- The ARP response contains the MAC address of the host who owns it, which is then used for future communication

Manual ARP requests on Linux

```
[root@master ~]# arping -c 3 -I em3 192.168.1.1
ARPING 192.168.1.1 from 192.168.0.1 em3
Unicast reply from 192.168.1.1 [84:7B:EB:D9:35:2A]  0.705ms
Unicast reply from 192.168.1.1 [84:7B:EB:D9:35:2A]  0.728ms
Unicast reply from 192.168.1.1 [84:7B:EB:D9:35:2A]  0.701ms
Sent 3 probes (1 broadcast(s))
Received 3 response(s)
```

ARP Cache on Linux

- to reduce the number of ARP requests operating systems maintain an ARP cache that contains the mapping of IP addresses to MACs

```
[root@master ~]# arp -n
Address                  HWtype  HWaddress           Iface
192.168.20.153           ether   7c:d3:0a:c7:36:a4   em1
192.168.19.11            ether   84:7b:eb:f4:fd:5c   em1
192.168.20.56            ether   7c:d3:0a:c7:29:f6   em1
192.168.0.80             ether   24:8a:07:51:a7:82   em3
...
```

Ethernet Frames

- When a NIC receives a frame, it checks the following:

Ethernet Frames

- When a NIC receives a frame, it checks the following:
  - is the frame for my MAC address?

Ethernet Frames

- When a NIC receives a frame, it checks the following:
  - is the frame for my MAC address?
  - is the frame being broadcast?

Ethernet Frames

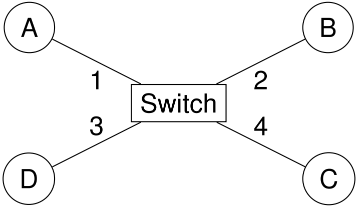
- When a NIC receives a frame, it checks the following:
  - is the frame for my MAC address?
  - is the frame being broadcast?
  - otherwise discard frame

Ethernet Frames

- When a NIC receives a frame, it checks the following:
  - is the frame for my MAC address?
  - is the frame being broadcast?
  - otherwise discard frame
- NICs can be configured in *promiscuous mode* to pass along all frames

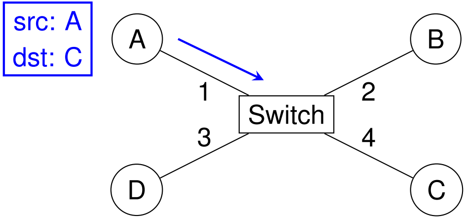
Ethernet Switches

Ethernet Switches



- a switch is a network device with many ports
- it forwards Ethernet frames to the correct port based on the MAC address
- at the beginning the switch doesn't know which port is connected to which MAC, so it broadcasts the frames to all ports
- by learning from the traffic it will create a mapping of MACs to ports and only send frames to the correct port
- this mapping is stored in a so-called *forwarding table* or *forwarding information base* (FIB)

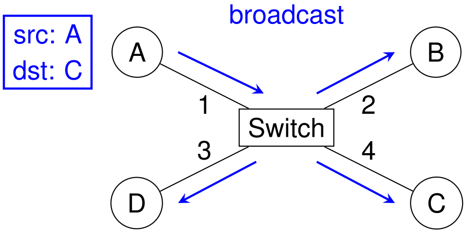
Switching: Broadcast, Unicast, and Multicast



Port	MAC
1	
2	
3	
4	

- Frame is sent with src=A and dest=C to port 1 of switch

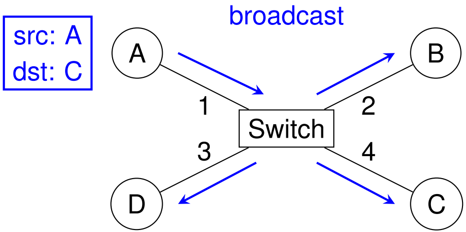
Switching: Broadcast, Unicast, and Multicast



Port	MAC
1	
2	
3	
4	

- Frame is sent with src=A and dest=C to port 1 of switch
- The switch doesn't know where to send it, so it **broadcasts** the frame on all ports

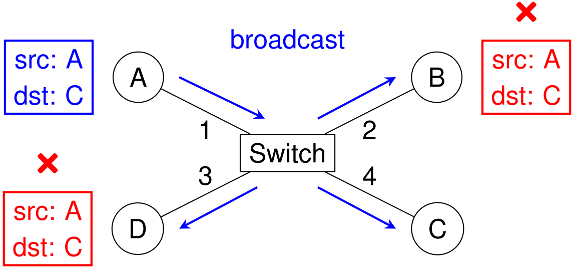
Switching: Broadcast, Unicast, and Multicast



Port	MAC
1	**A**
2	
3	
4	

- Frame is sent with src=A and dest=C to port 1 of switch
- The switch doesn't know where to send it, so it **broadcasts** the frame on all ports
- Along the way the switch learns that MAC A is sending from port 1 and stores this information in its MAC table

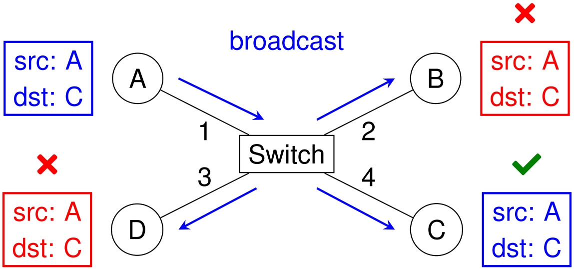
Switching: Broadcast, Unicast, and Multicast



Port	MAC
1	**A**
2	
3	
4	

- B and D will drop the frame, because the MAC isn't theirs

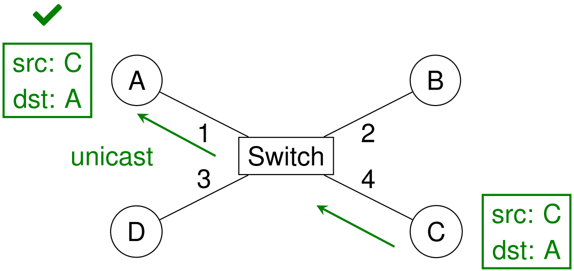
Switching: Broadcast, Unicast, and Multicast



Port	MAC
1	**A**
2	
3	
4	

- B and D will drop the frame, because the MAC isn’t theirs
- C will accept and process the frame

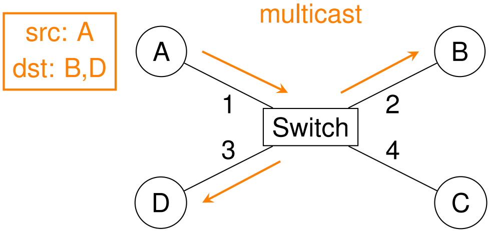
Switching: Broadcast, Unicast, and Multicast



Port	MAC
1	**A**
2	
3	
4	**C**

- When C replies to A, the switch already knows which port A is at, so it forwards the frame with **unicast** to port 1
- The switch will also learn the origin port of C and store it in its MAC table

Switching: Broadcast, Unicast, and Multicast

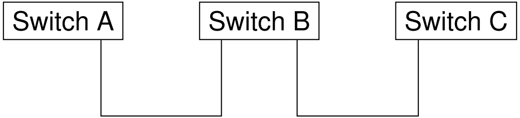


Port	MAC
1	**A**
2	**B**
3	**D**
4	**C**

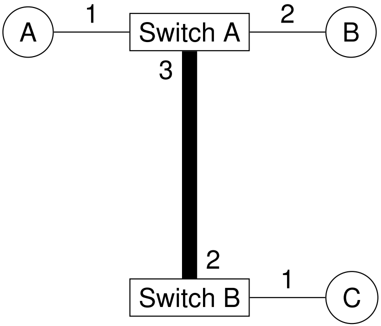
- beside unicast and broadcast, there is also **multicast**, which means an Ethernet frame is sent to multiple MACs at the same time

Connecting switches

- you can daisy-chain switches to create larger networks



Connecting switches



- Switches can learn that multiple MAC addresses are reachable from a single port

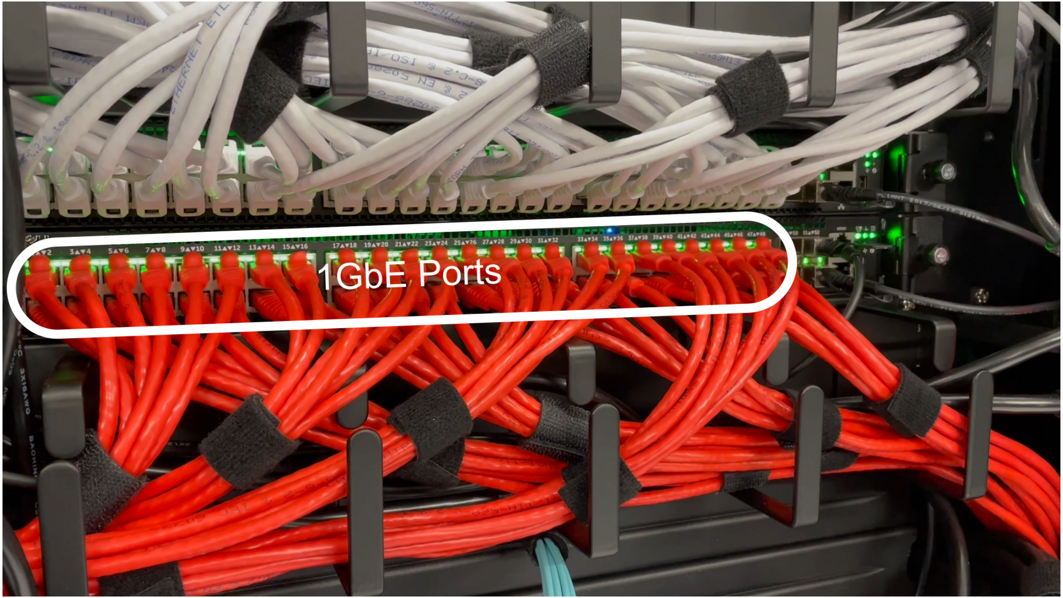
Table: Switch A

Port	MAC
1	A
2	B
3	C

Table: Switch B

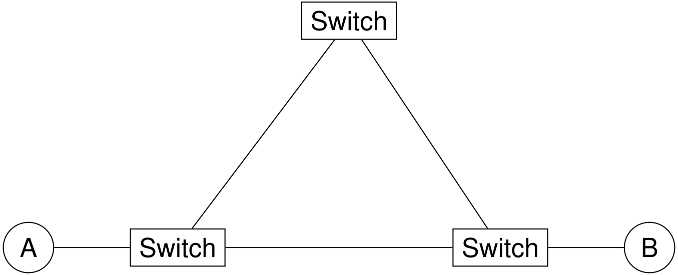
Port	MAC
1	C
2	**A,B**

Uplinks



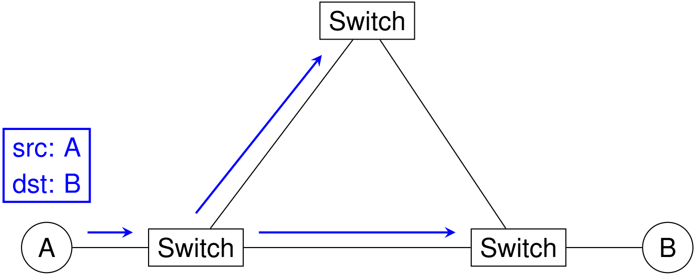
Uplinks

Switching Loops



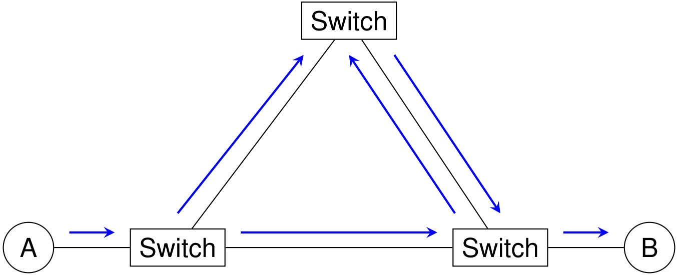
- one might want to connect switches which causes loops

Switching Loops



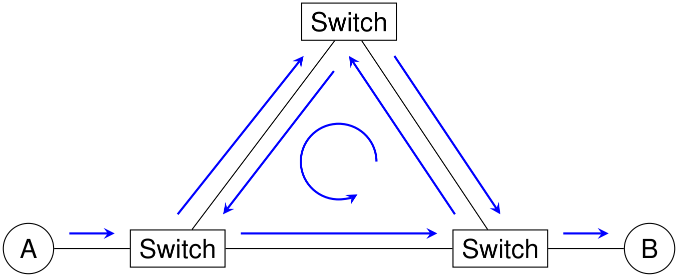
- one might want to connect switches which causes loops
- due to broadcasting, loops can cause frames to transmitted indefinitely

Switching Loops



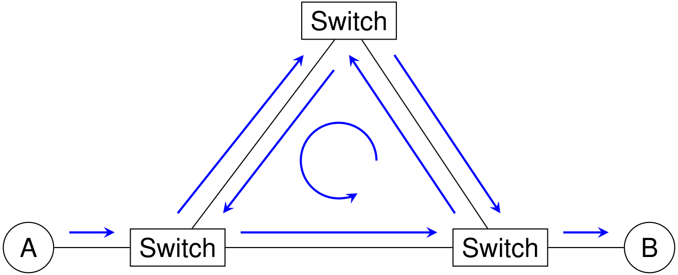
- one might want to connect switches which causes loops
- due to broadcasting, loops can cause frames to transmitted indefinitely

Switching Loops



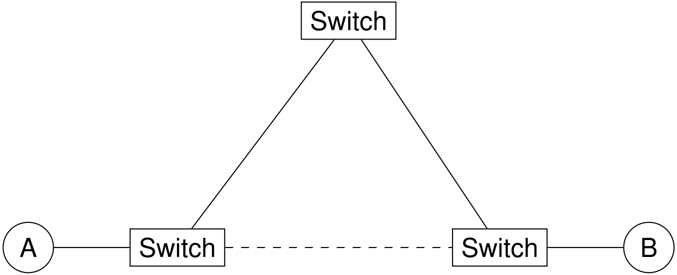
- one might want to connect switches which causes loops
- due to broadcasting, loops can cause frames to transmitted indefinitely

Switching Loops



- one might want to connect switches which causes loops
- due to broadcasting, loops can cause frames to transmitted indefinitely
- this will lead to high processor load in the switches and eventually to dropped packets, since normal traffic can no longer flow

Spanning Tree Protocol (STP)



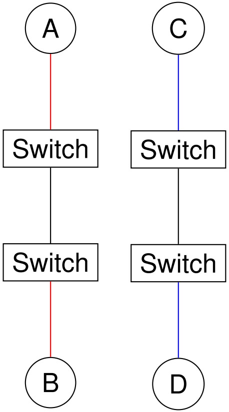
- as soon as switches are connected, they start talking with each other
- with STP they detect loops and disable links
- should a link fail, disabled links will be enabled

LAN vs VLAN

LAN vs VLAN

LAN

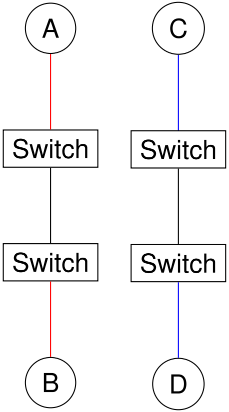
For each network you need a separate set of switches.



LAN vs VLAN

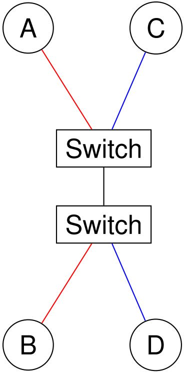
LAN

For each network you need a separate set of switches.



VLAN

You can set up multiple broadcast domains in a single set switches to create *virtual* networks.



VLAN

- allows a single switch to act like multiple ones

VLAN

- allows a single switch to act like multiple ones
- can be used to isolate traffic between switchports

VLAN

- allows a single switch to act like multiple ones
- can be used to isolate traffic between switchports
- each VLAN has a unique ID, between 1 and 4095

VLAN

- allows a single switch to act like multiple ones
- can be used to isolate traffic between switchports
- each VLAN has a unique ID, between 1 and 4095
- each switch port can be associated to a VLAN ID $\Rightarrow$ becomes an **access port**

VLAN

- allows a single switch to act like multiple ones
- can be used to isolate traffic between switchports
- each VLAN has a unique ID, between 1 and 4095
- each switch port can be associated to a VLAN ID $\Rightarrow$ becomes an **access port**
- frames coming into an access port are automatically **tagged** for a particular VLAN.

VLAN

- allows a single switch to act like multiple ones
- can be used to isolate traffic between switchports
- each VLAN has a unique ID, between 1 and 4095
- each switch port can be associated to a VLAN ID $\Rightarrow$ becomes an **access port**
- frames coming into an access port are automatically **tagged** for a particular VLAN.
- frames of one VLAN will only be forwarded to access ports of that VLAN.

VLAN

- allows a single switch to act like multiple ones
- can be used to isolate traffic between switchports
- each VLAN has a unique ID, between 1 and 4095
- each switch port can be associated to a VLAN ID $\Rightarrow$ becomes an **access port**
- frames coming into an access port are automatically **tagged** for a particular VLAN.
- frames of one VLAN will only be forwarded to access ports of that VLAN.
- ports connecting switches that serve the same VLANs must be configured to forward frames of any VLAN and must add the VLAN information to each frame. Such ports are called **trunk ports**

VLAN

- allows a single switch to act like multiple ones
- can be used to isolate traffic between switchports
- each VLAN has a unique ID, between 1 and 4095
- each switch port can be associated to a VLAN ID $\Rightarrow$ becomes an **access port**
- frames coming into an access port are automatically **tagged** for a particular VLAN.
- frames of one VLAN will only be forwarded to access ports of that VLAN.
- ports connecting switches that serve the same VLANs must be configured to forward frames of any VLAN and must add the VLAN information to each frame. Such ports are called **trunk ports**
- IEEE 802.1Q is the standard on how VLAN information is encoded in frames, which makes it work between different switch vendors.

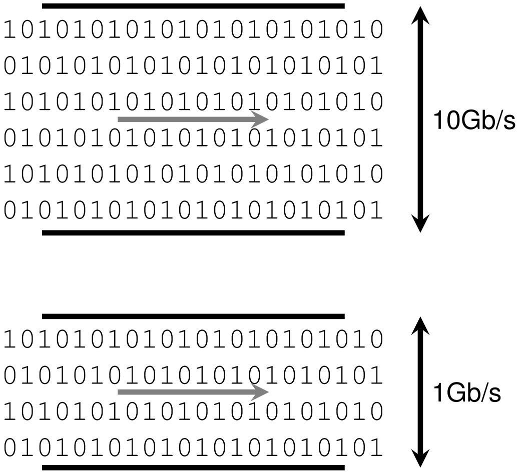
Physical Layer

7	Application Layer
6	Presentation Layer
5	Session Layer
4	Transport Layer
3	Network Layer
2	Data Link Layer
1	**Physical Layer**

- Responsible for how bits are encoded and transmitted as **electrical or optical signals or electromagnetic waves**.
- Used technology limits both the **bandwidth** and the **latency** of a network

Network Bandwidth

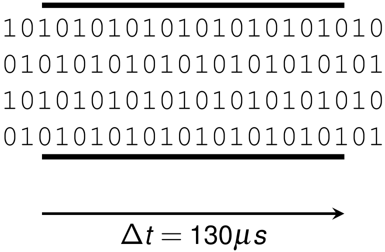
- the maximum amount of data that can be transfered across a path per second
- determines the theoretical maximum transfer rate
- Unit: bits / second



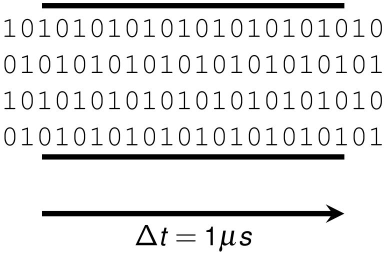
Network Bandwidth: Examples

Technology	Bandwidth
Ethernet (copper) 1000BASE-T	1 Gbit/s
Ethernet (copper) 10GBASE-T	10 Gbit/s
Ethernet (fiber) 10GBASE-SR	10 Gbit/s
Ethernet (fiber) 10GBASE-SR4	40 Gbit/s
InfiniBand EDR	100 Gbit/s
WiFi 802.11g	56 Mbit/s
WiFi 802.11n	600 Mbit/s
WiFi 802.11ac	1.3 Gbit/s
4G LTE	300 Mbit/s
5G mmWave	1.8Gbit/s

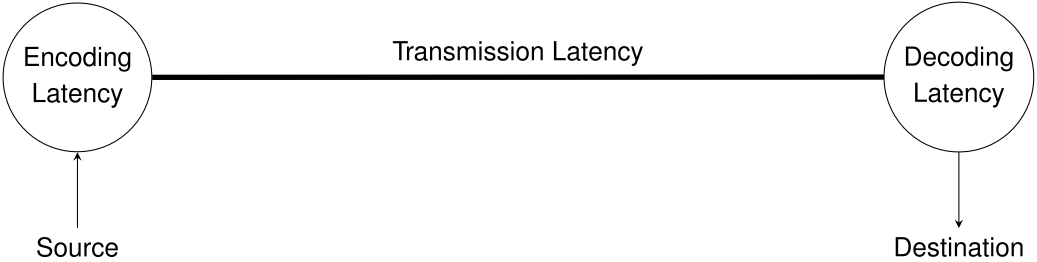
Network Latency



- how much time does it take for a transmission from start to finish

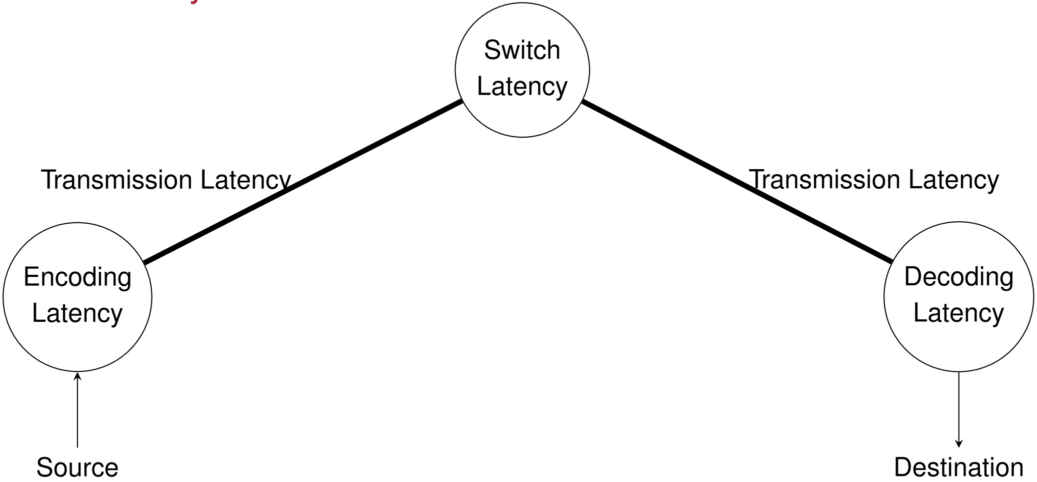


Network Latency: Point-to-Point



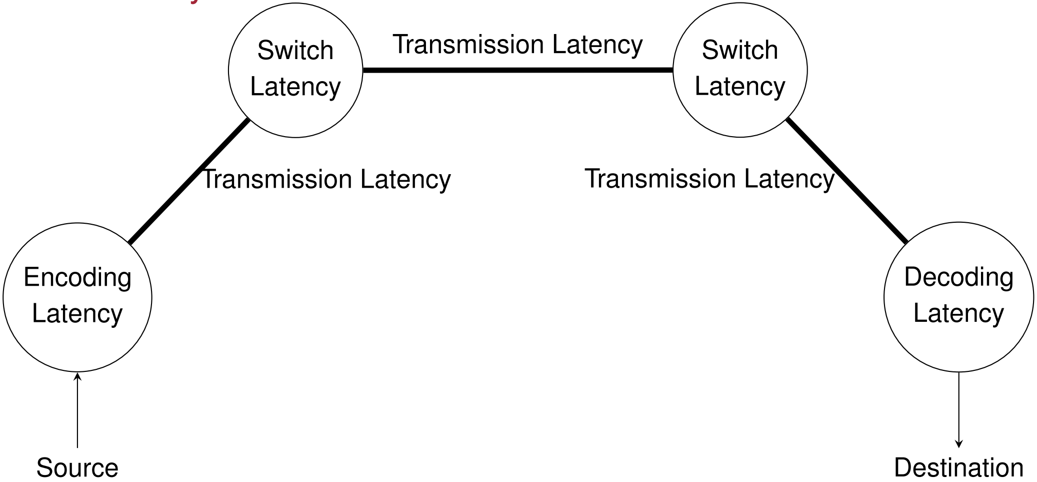
$$\text{latency} = \Delta t_{\text{encoding}} + \Delta t_{\text{transmission}} + \Delta t_{\text{decoding}}$$

Network Latency: 2 transmissions



$$\text{latency} = \Delta t_{\text{encoding}} + \sum \Delta t_{\text{transmission}} + \Delta t_{\text{switch}} + \Delta t_{\text{decoding}}$$

Network Latency: 3 transmissions



$$\text{latency} = \Delta t_{\text{encoding}} + \sum \Delta t_{\text{transmission}} + \sum \Delta t_{\text{switch}} + \Delta t_{\text{decoding}}$$

Network Layers: Summary

Basic Reference Model for Open Systems Interconnect (OSI model)

7	Application Layer	Application protocol (HTTP, FTP, SMTP)
6	Presentation Layer	compression, encryption, encoding
5	Session Layer	authentication, permissions, session restoration
4	Transport Layer	end-to-end communication (TCP, UDP)
3	Network Layer	data exchange across network boundaries (packets)
2	Data Link Layer	reliable data delivery in a LAN/WAN (frames)
1	Physical Layer	how are bits transmitted (symbols)