Network (in)security

AIS23

Hacked from the original slides by
Søren Debois and Marco Carbone

Review

Photo by Michael Petrila on Unsplash

You must provide feedback.

Otherwise, we'll just keep doing it wrong.

Review

- About this course
- Security models, assumptions, goals
  Confidentiality, **I**ntegrity, **A**vailability
- Security principles

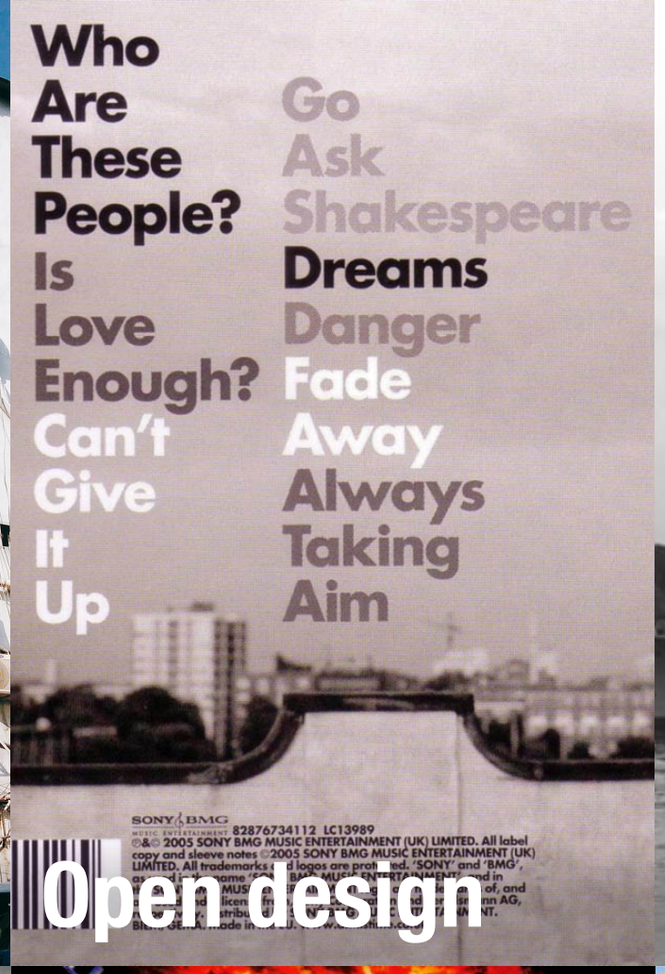
Security goals

- **C**onfidentiality
  “Prevent unauthorised access to information.”

- **I**ntegrity
  “Prevent unauthorised altering of information.”

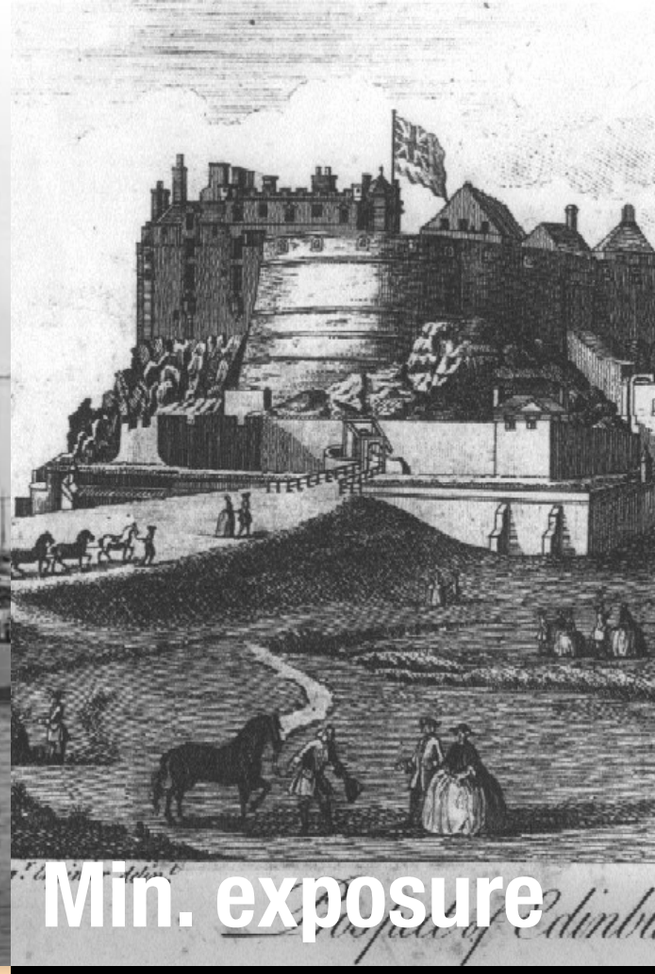
- **A**vailability
  “Ensure the availability of the system for authorised uses.”

Simplicity



Open design

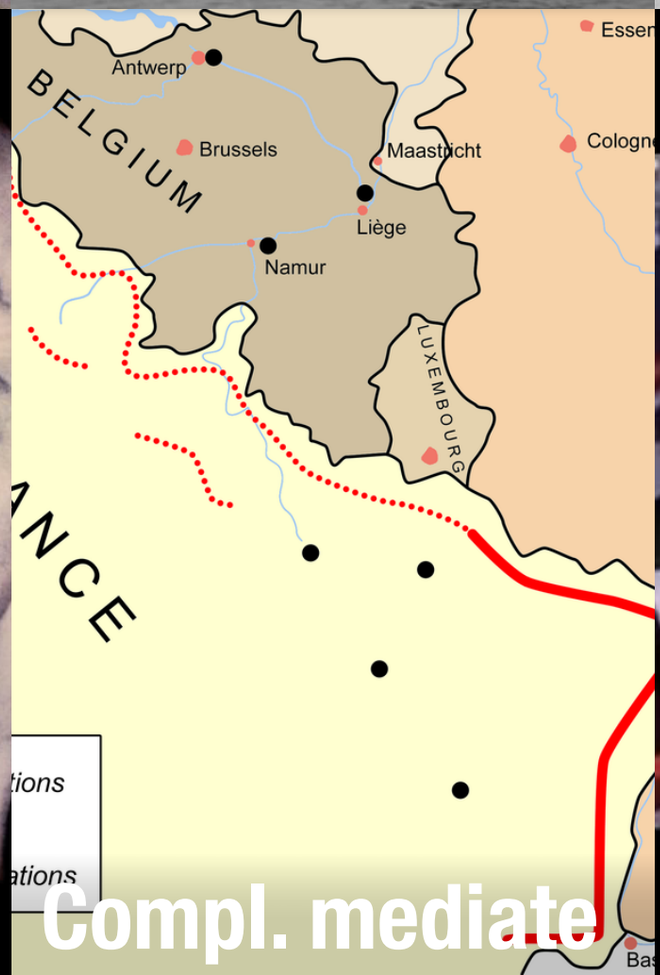
Compartment.



Min. exposure

Least privilege

Failsafe default



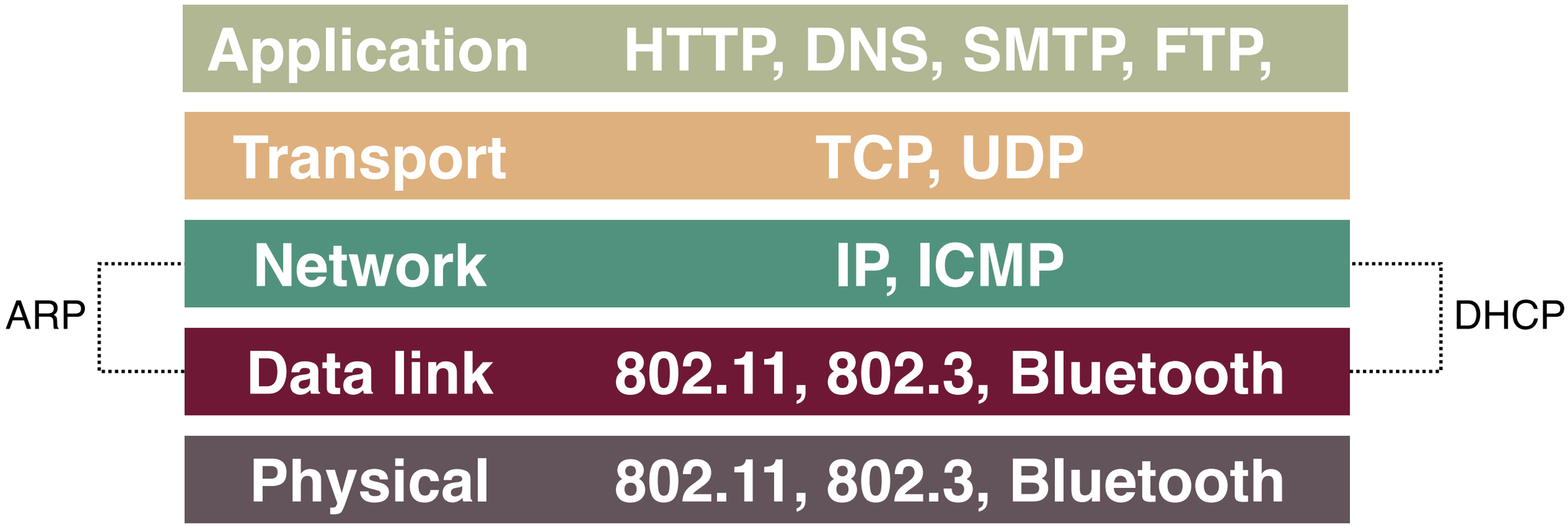
Compl. mediate

No single failure

Introduction

Plan

- Foundations of Networking
- Attacks on the network stack
- Denial of service attacks



Foundations of Networking

World Wide Web

An information system on the Internet which allows documents to be connected to other documents by hypertext links, enabling the user to search for information by moving from one document to another.

from Wikipedia

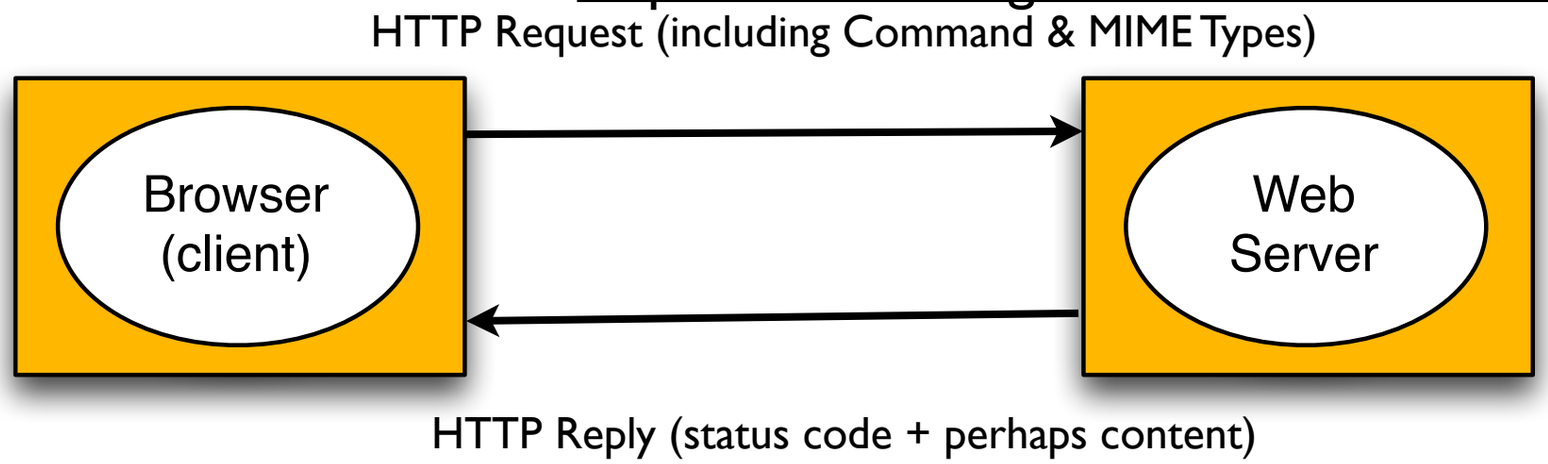
World **W**ide **W**eb

1989->

Open, extensible standards (URL, HTTP, HTML, XML, ..)

URL Example: http://servername[:port][/pathName][?query][#fragment]

http://www.w3.org/Protocols/rfc2616/rfc2616.html




```
<!DOCTYPE HTML PUBLIC "-//
W3C//DTD HTML 3.2//EN">
<html>
   <head>
      <title>Welcome!</title>
   </head>
   <body>
      <h1>Hello World!</h1>
      <p> This is a webpage.
   </body>

</html>
```

What can the resource be ?

- a static file (possibly linking to resources)
- a program, dynamically generating a file
- a static/dynamically generated file containing code

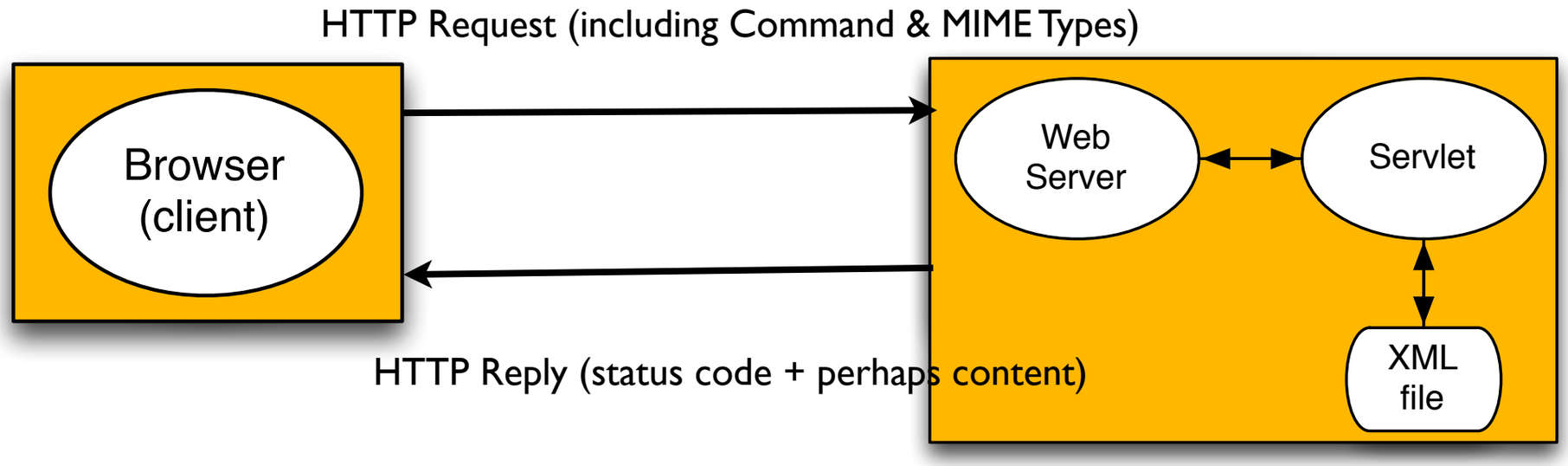
Example

World **W**ide **W**eb

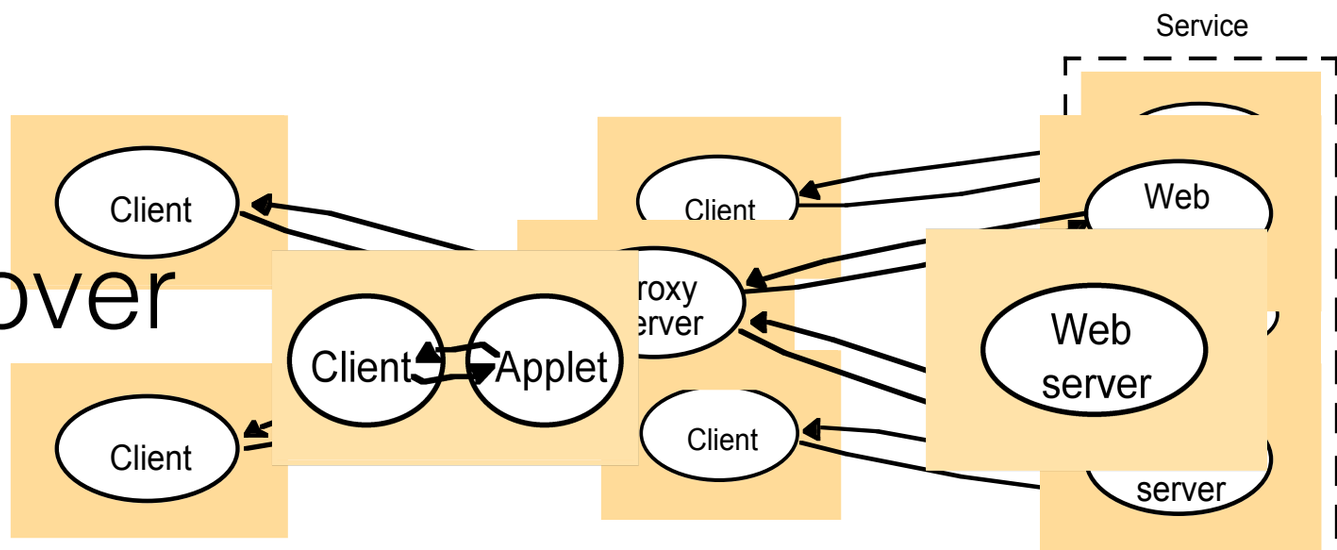
1989->

Open, extensible standards (URL, HTTP, HTML, XML, ..)

URL Example: http://servername[:port][/pathName][?query][#fragment]



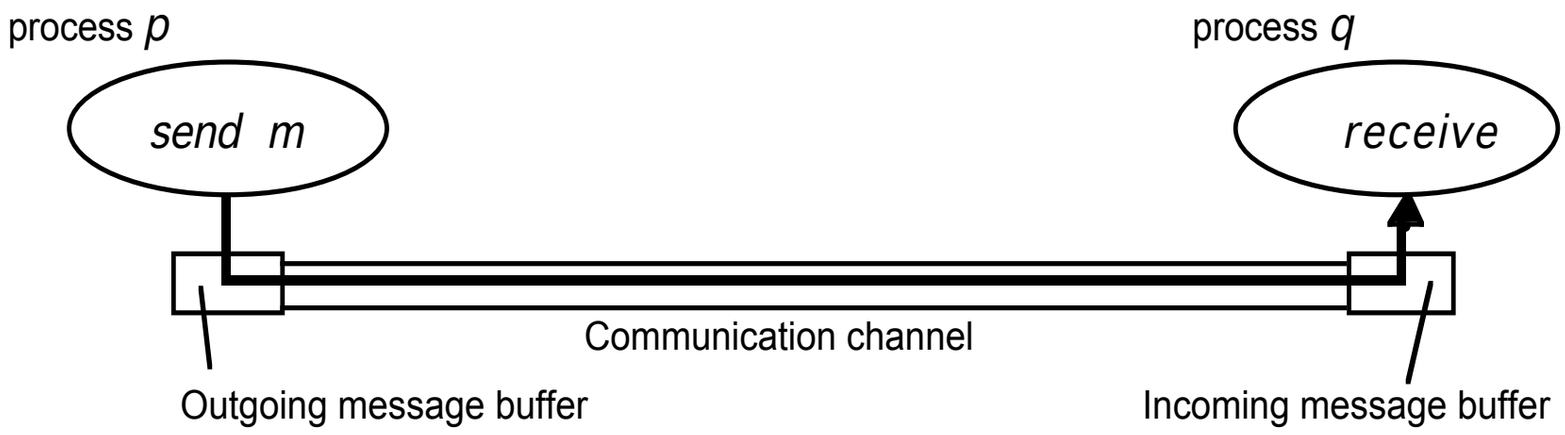
- How it is transferred over the network



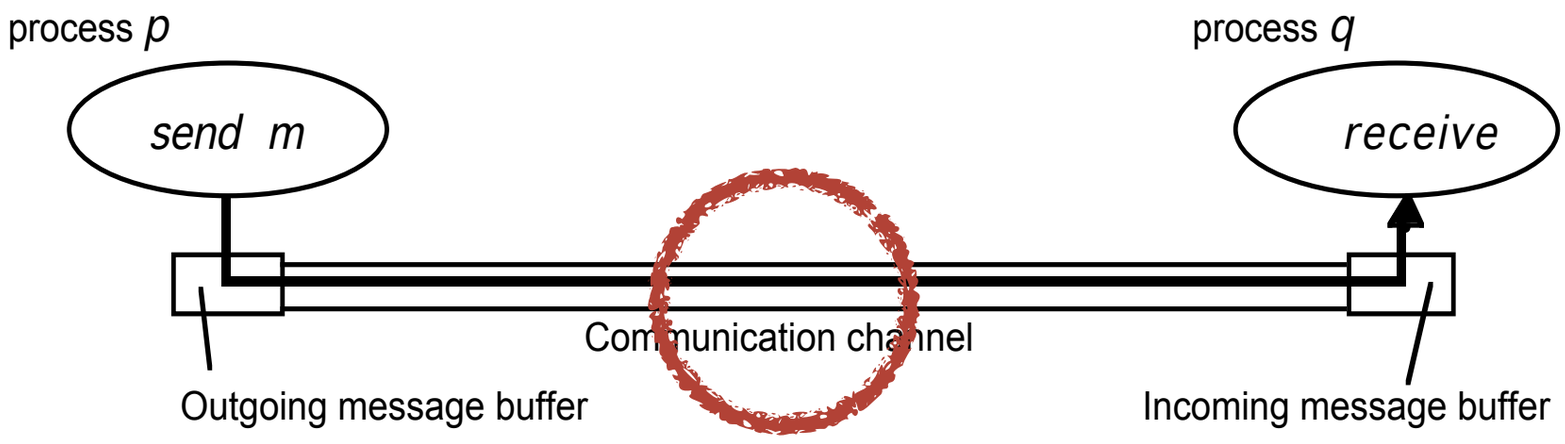
Terminology (i)

- *Processes* run on *hosts*.
- Processes send and receive *messages* on *communication channels.*
- Processes and communication channels are both subject to *failures.*
- Channel failures may yield *dropped, lost, re-ordered* or *duplicated* messages.

Basic model


process *p*
process *q*
send m
receive
Communication channel
Outgoing message buffer
Incoming message buffer

Basic model: Failures



Channel may exhibit omission & byzantine failures

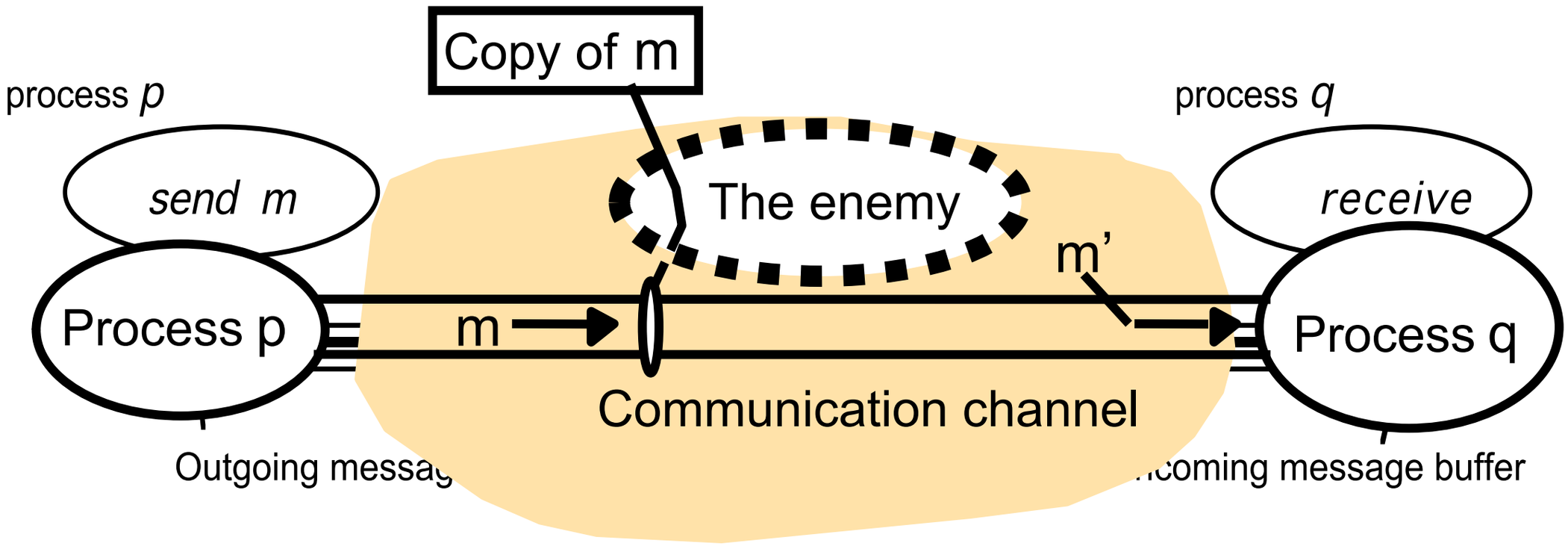
Channel failure handling

- Checksums
  (vs. message corruption)

- Acknowledgments, timeouts, re-transmissions
  (vs. lost messages)

- Sequence numbers
  (vs. duplicated/re-ordered messages)

Fundamental Models



- Interaction model. Performance (Latency, bandwidth, jitter).
- Failures.
- Security. Identification, authentication, confidentiality, integrity, denial of service

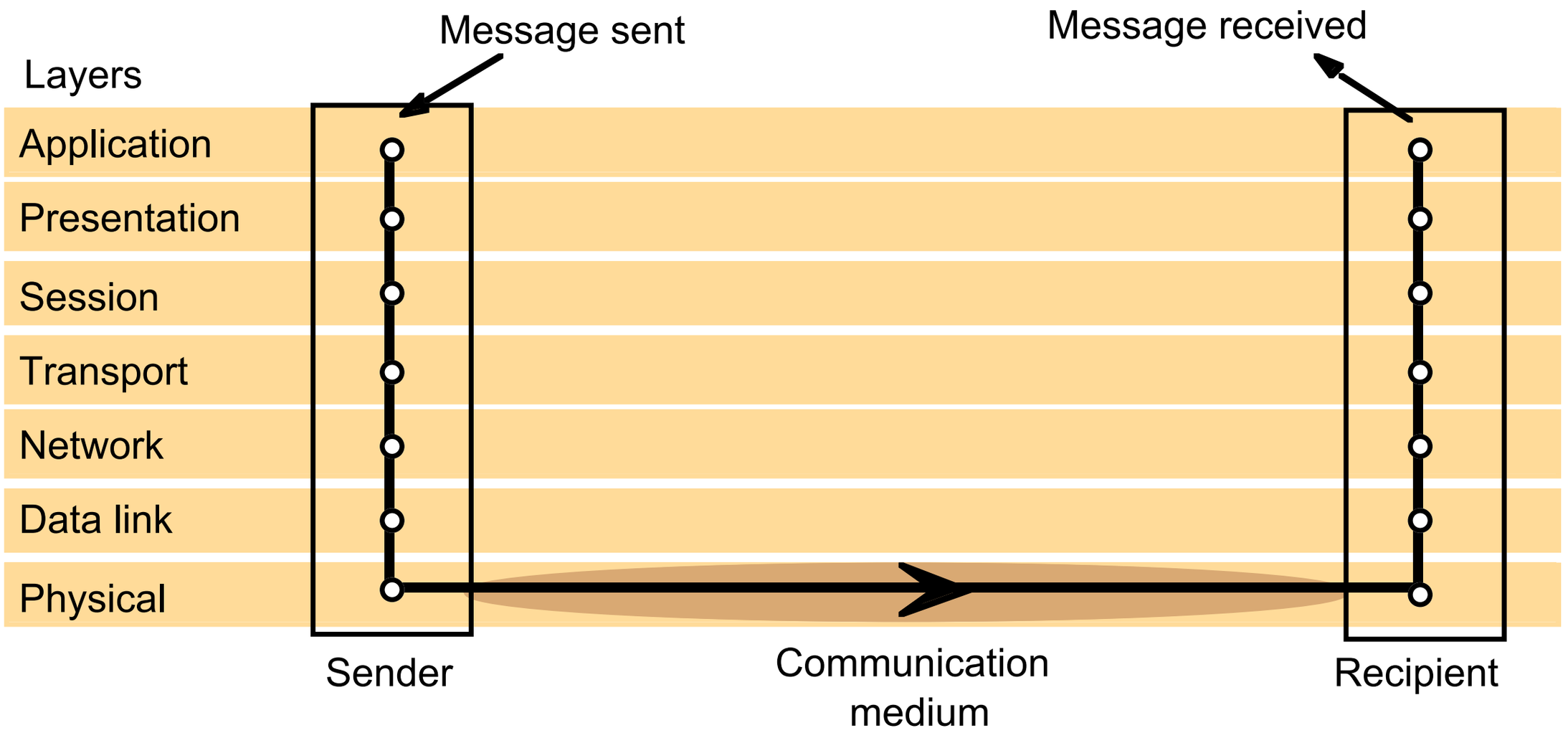
Adversary capabilities (Dolev-Yao model)

- The adversary has complete control of the network. He may:
  - intercept messages
  - replay messages
  - transform message
  - insert messages
  - delete messages

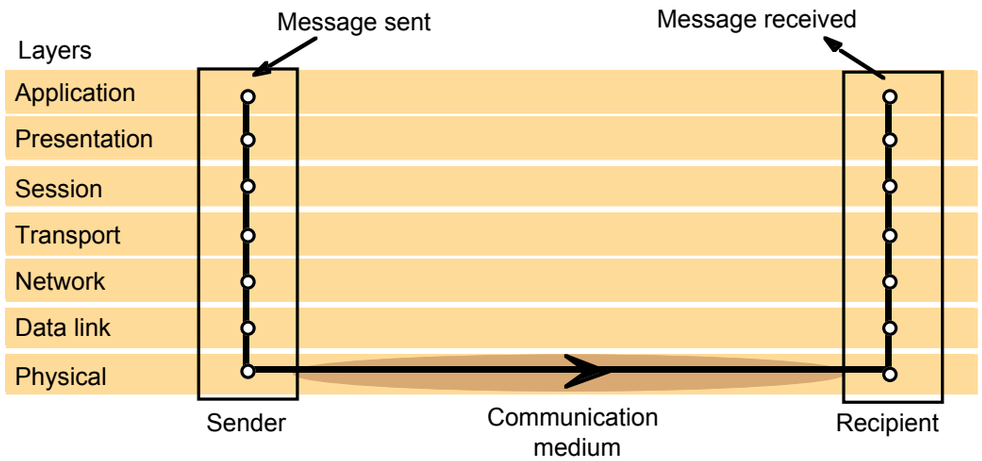
Adversary incapabilities

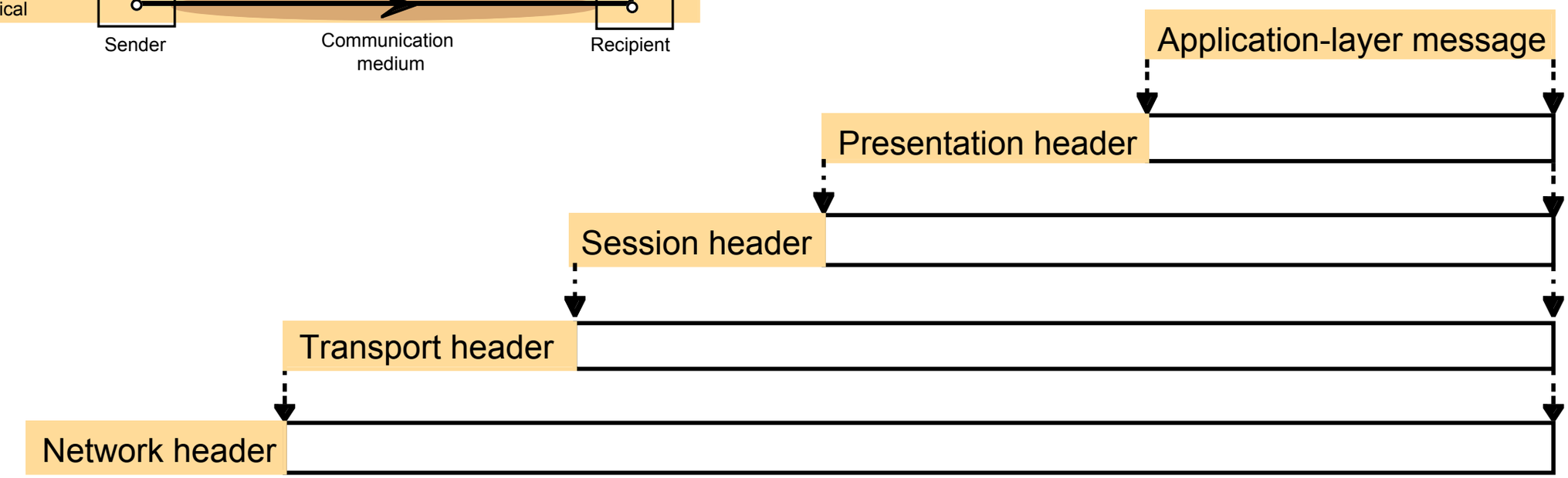
- The adversary cannot guess our secrets.

Protocol layers

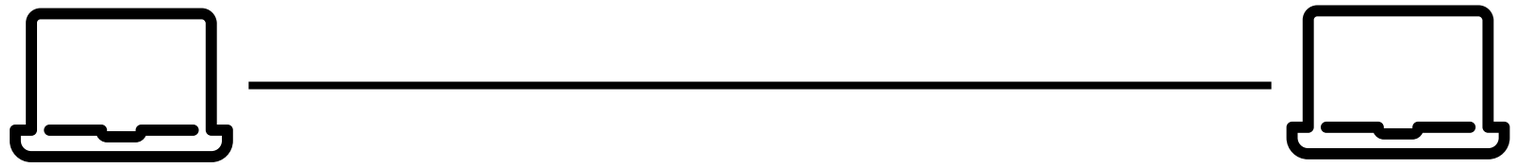


Protocol layers headers





Physical layer



The physical layer

- Responsibility:
  Transmission of binary data across a physical link

- Usually broadcast
  (e.g., IEEE 802.3 Ethernet)

- Interconnection via hub

- Usually provides no guarantees.

The physical layer

- Responsibility:
  Transmission of binary data across a physical link

- **Usually broadcast**
  (e.g., IEEE 802.3 Ethernet)

- **Usually provides no guarantees**

Let's break it!

Adversary capabilities (Dolev-Yao model)

- The adversary has complete control of the network. He may:
  - intercept messages
  - replay messages
  - transform message
  - insert messages
  - delete messages

Attacks on the physical layer

- **Eavesdropping** (confidentiality)
  Frames are broadcast;
  I can see them even if they aren't for me.

- **Tampering** (integrity)
  You won't detect my change

- **Denial-of-service** (availability)
  If I put enough noise on the line, you won't send or receive any messages.

- **Message injection**
  I can put arbitrary messages on the wire

Data-link layer

The data-link layer

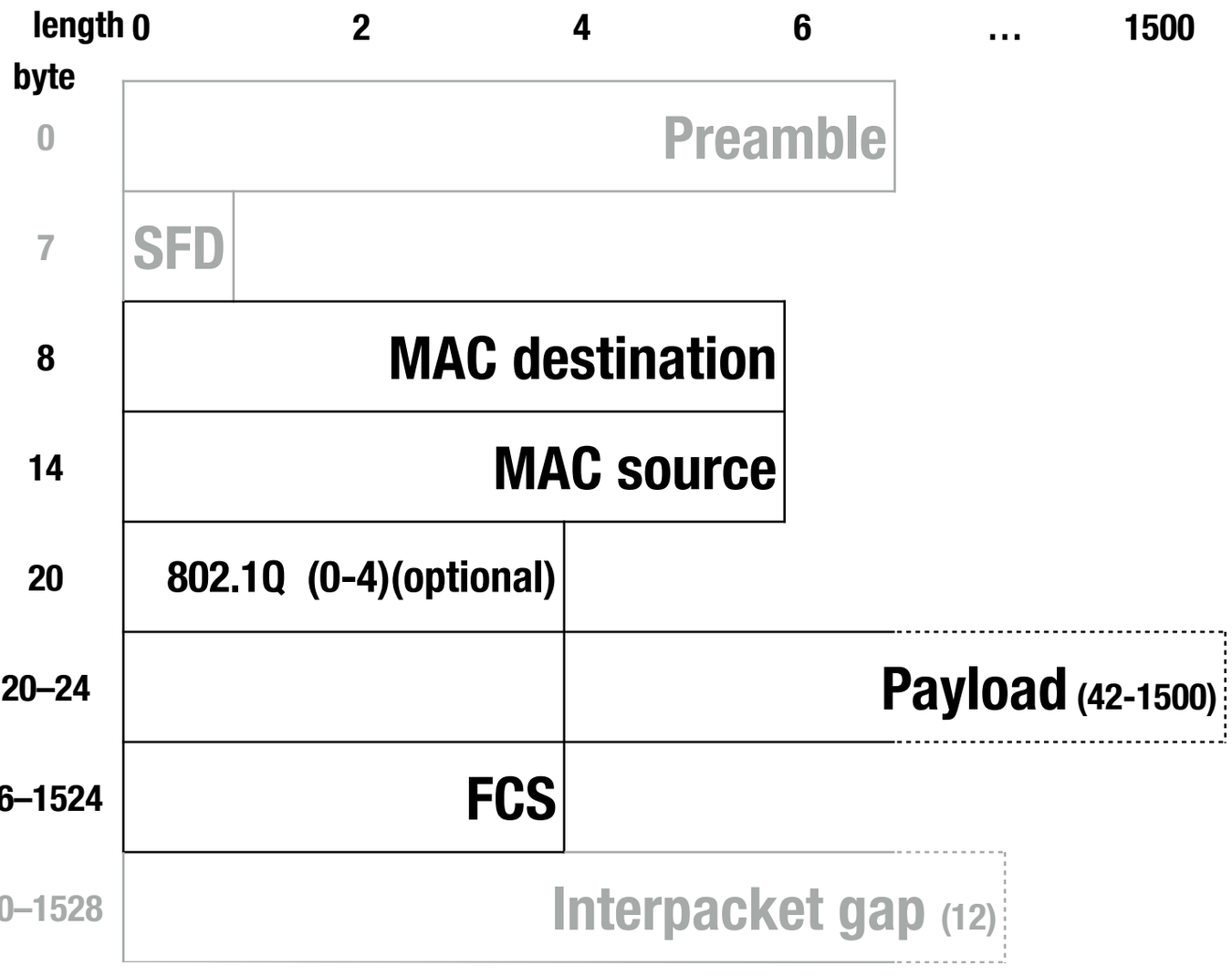
- Responsibility:
  Transmission of packets between hosts connected by a physical link

- Solves addressing:
  Media Access Control (MAC) addresses

- Solves (partly) reliability:
  Checksums (e.g., IEEE 802.3 use of CRC)

- Performance gains by using **switches** rather than hubs

802.3: Ethernet (frame)



- MAC destination, source:
  “To”, “From”
  (May be broadcast)

- Payload (actual contents) up to 1500 bytes

- Frame Check Sequence:
  32-bit Cyclic Redundancy Check checksum
  Detects error bursts < 32 bit

- Check failed => Frame dropped

802.3: Ethernet (frame)

- MAC destination, source: “To”, “From” (May be broadcast)

- Payload (actual contents) up to 1500 bytes

- Frame Check Sequence: 32-bit Cyclic Redundancy

Let’s break it!

Attacks on the data link layer

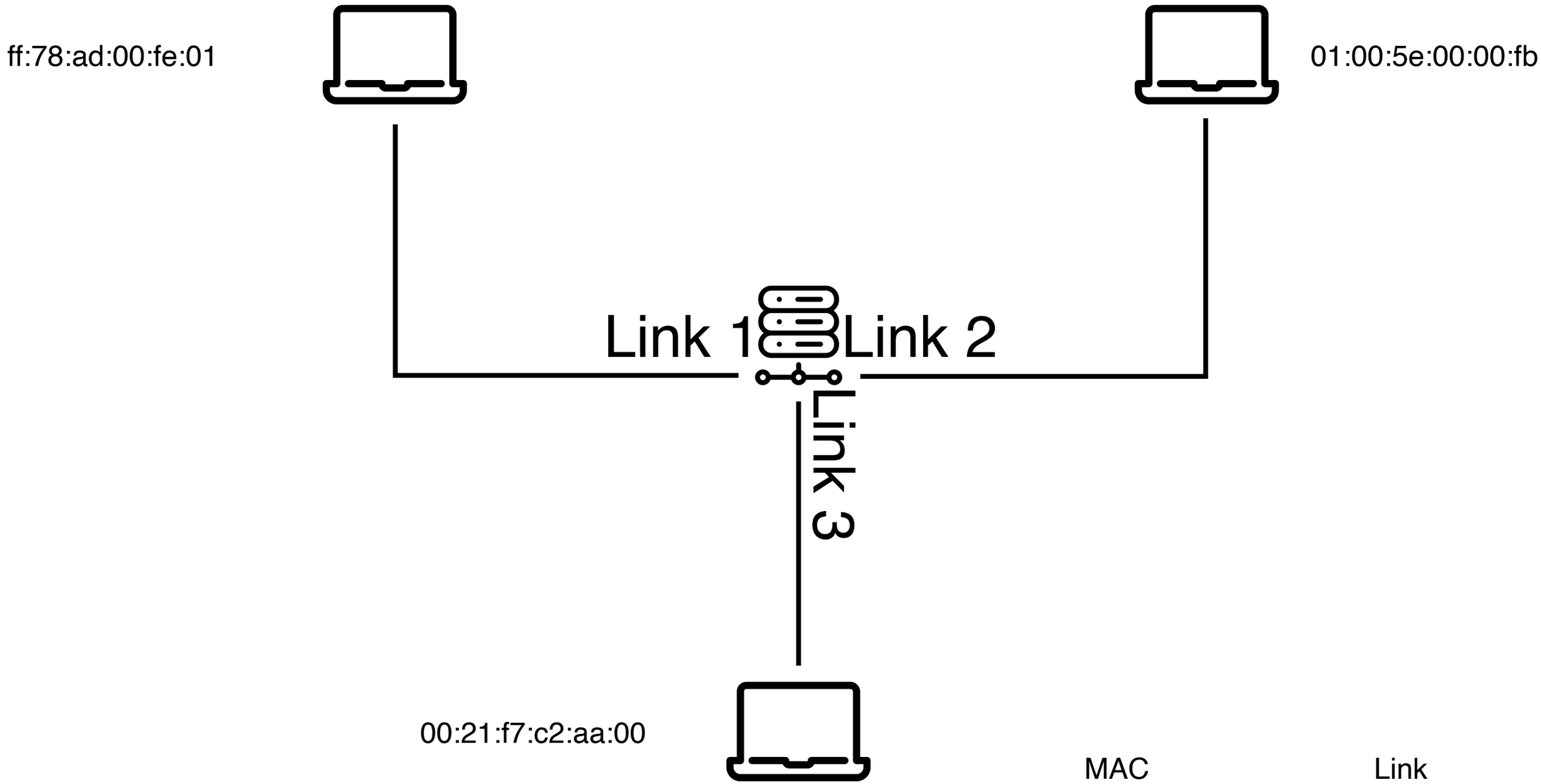
- **Tampering** (integrity)
  CRC is cryptographically weak;
  you won't detect my change
  (We'll do this in the Crypto-lecture.)

- **Message injection/MAC spoofing**
  I can put arbitrary messages on the wire

- **Eavesdropping** (confidentiality)
  Frames are broadcast;
  I can see them even if they aren't for me.
  Switches: MAC flooding

MAC flooding

- Transmit enough fake frames with new src addresses that the switches' table contains no actual addresses
- Switch must now broadcast all frames



MAC	Link
ff:78:ad:00:fe:01	1
01:00:5e:00:00:fb	2

Network layer

Network layer

- IP protocol
  (IPv4)

- Hosts identified by IP addresses

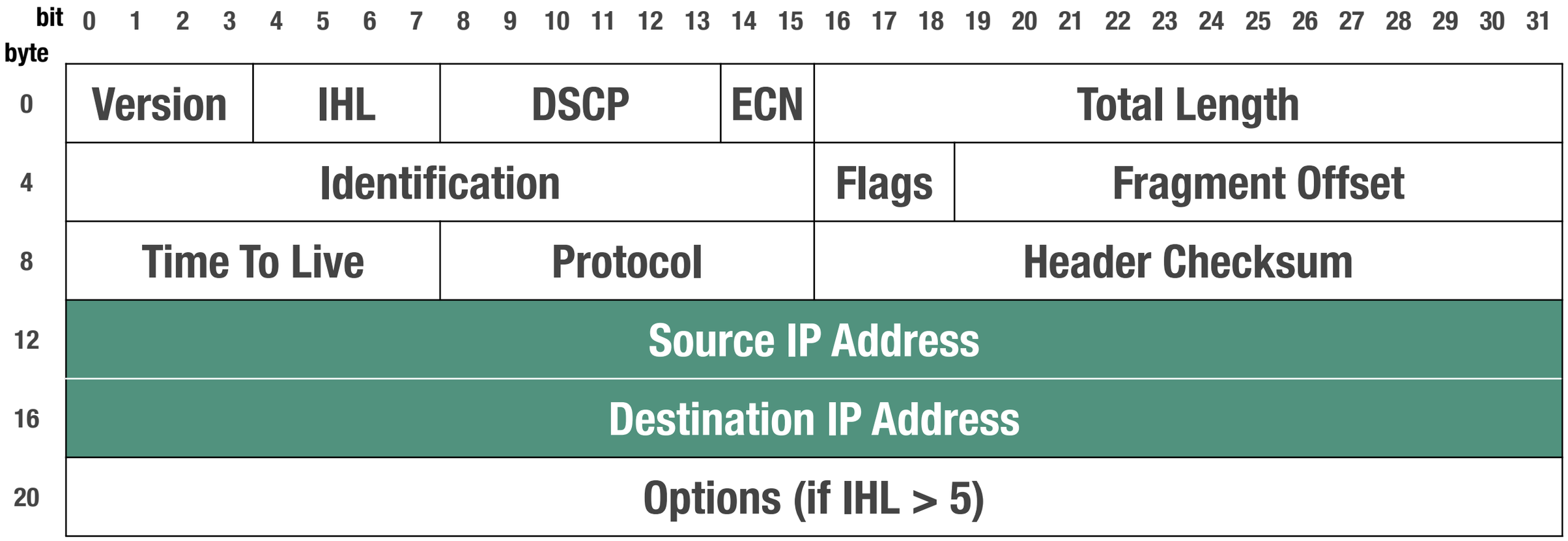
- Best-effort (unreliable) delivery

- May introduce packet duplication, out-of-order delivery

IP Operations

- Next-hop routing
- BGP, ARP, DHCP
- MTU (v4 only), Fragmentation

IPv4 Header


bit 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
byte
0
Version
IHL
DSCP
ECN
Total Length
4
Identification
Flags
Fragment Offset
8
Time To Live
Protocol
Header Checksum
12
Source IP Address
16
Destination IP Address
20
Options (if IHL > 5)

IP Operations

- Next-hop routing
- BGP, ARP, DHCP
- MTU (v4 only), Fragmentation

Let’s break it!

IP Operations

- **Next-hop routing**
- **BGP, ARP, DHCP**
- **MTU (v4 only), Fragmentation**

Let’s break it!

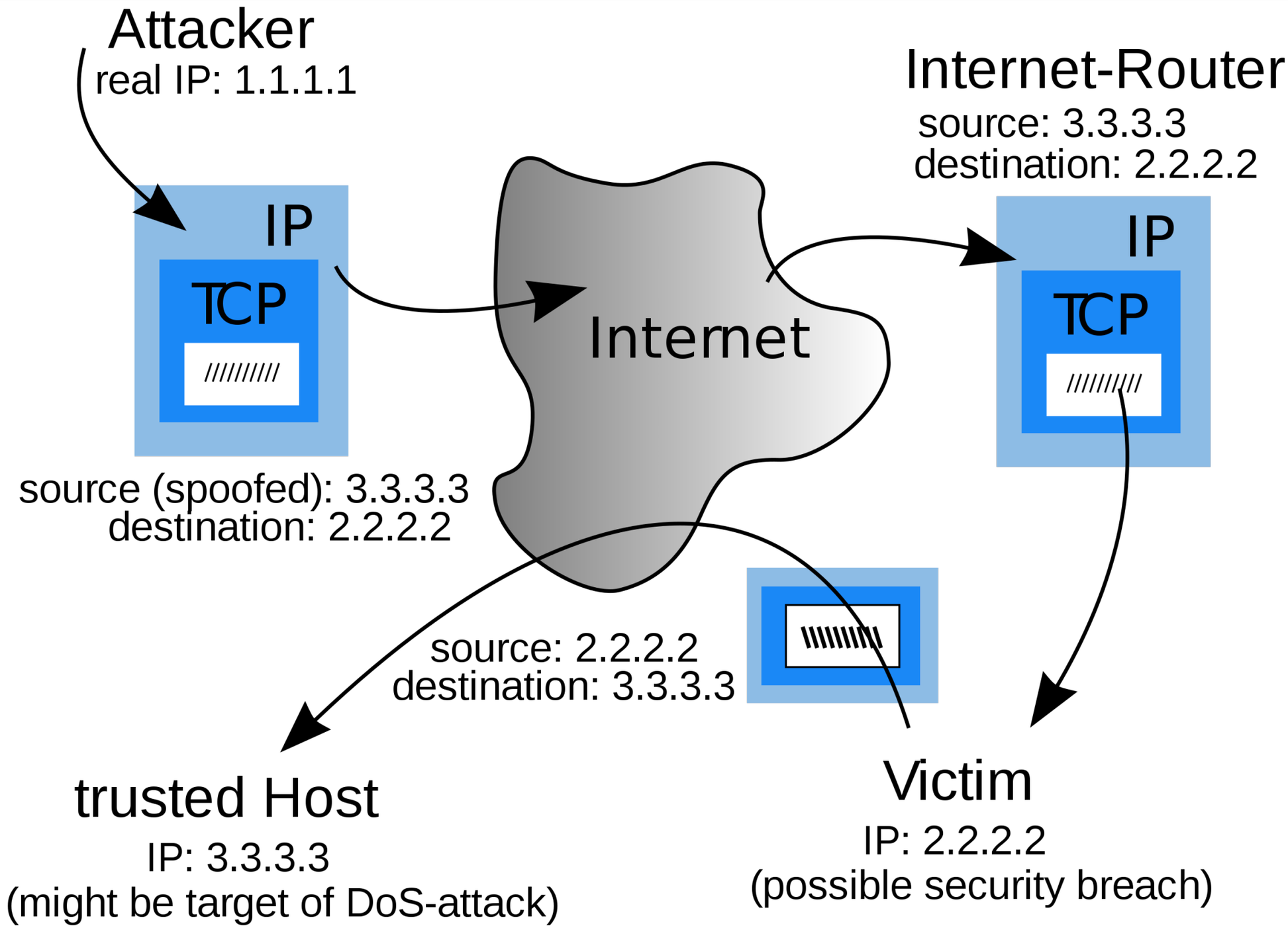
Spoofing

- **ARP Cache Poisoning** (?)
  (aka ARP spoofing, ARP Poison Routing)
  Spoof ARP “is” packets redirecting traffic for other IP to my machine.

- **IP Spoofing**

- **DHCP Starvation**

IP Spoofing


Attacker
real IP: 1.1.1.1
IP
TCP
source (spoofed): 3.3.3.3
destination: 2.2.2.2
Internet
Internet-Router
source: 3.3.3.3
destination: 2.2.2.2
IP
TCP
source: 2.2.2.2
destination: 3.3.3.3
trusted Host
IP: 3.3.3.3
(might be target of DoS-attack)
Victim
IP: 2.2.2.2
(possible security breach)

Local denial-of-service attacks

- ARP Cache poisoning
- DHCP Starvation

Remote denial-of-service attacks

- Ping flooding
- IP fragmentation attack
- Distribute the attack from many attacking machines for maximum effect

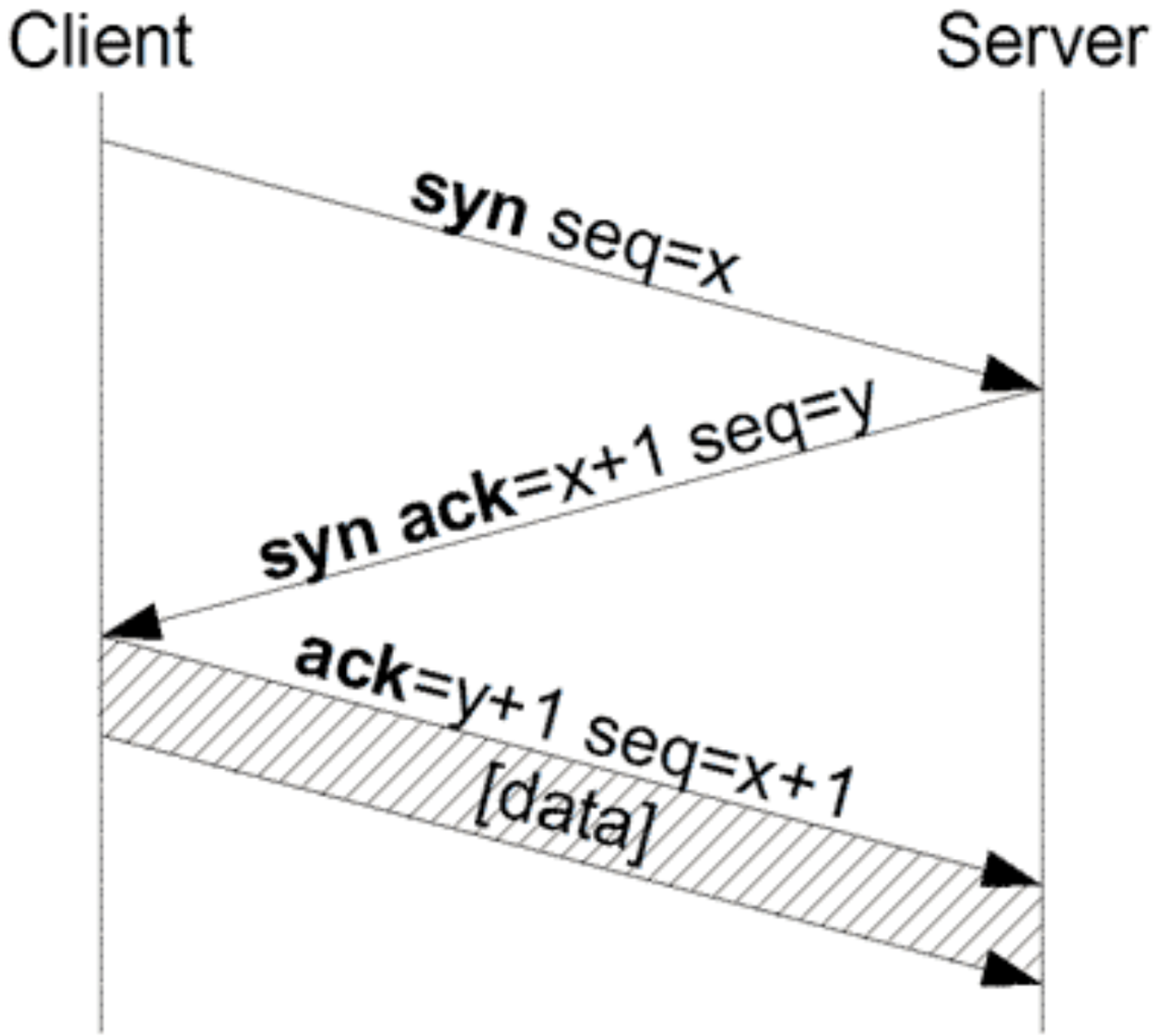
Transport layer

TCP

- Connection-oriented, reliable, streaming protocol.
- Achieved by message/acknowledgment sequence numbers, timeouts.
- Protocol specified as a fairly complex state machine
- Also: Flow control, congestion control

Connection setup

- 3-way handshake
- A TCP connection is identified by all of (ip1, port1, ip2, port2).
- … so, a web-server at 130.226.133.47:80 can have multiple connections at that single port.



bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
byte																																
0	Source port																Destination port														
4	Sequence number																															
8	Acknowledgment number (on ACK)																															
12	Data offset				Reserved			NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window size															
16	Checksum																Urgent pointer (on URG)															
20	Options …																															
…																																

URG out-of-band receive

ACK acknowledgment significant

PSH do not buffer

SYN synchronise sequence number

RST drop connection

FIN last packet

bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
byte																																
0	Source port																Destination port														
4	Sequence number																															
8	Acknowledgment number (on ACK)																															
12	Data offset				Reserved			NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window size															
16	Checksum																Urgent pointer (on URG)															
20	Options …																															
…																																

URG out-of-band receive

SYN synchronise sequence number

Let’s break it!

TCP sequence prediction attack

- Suppose we want to hi-jack a connection from host A to B.

- TCP Sequence numbers are sent in cleartext (eavesdropping)

- Listen to traffic from B. Kill B's end of the connection (e.g., next slide)

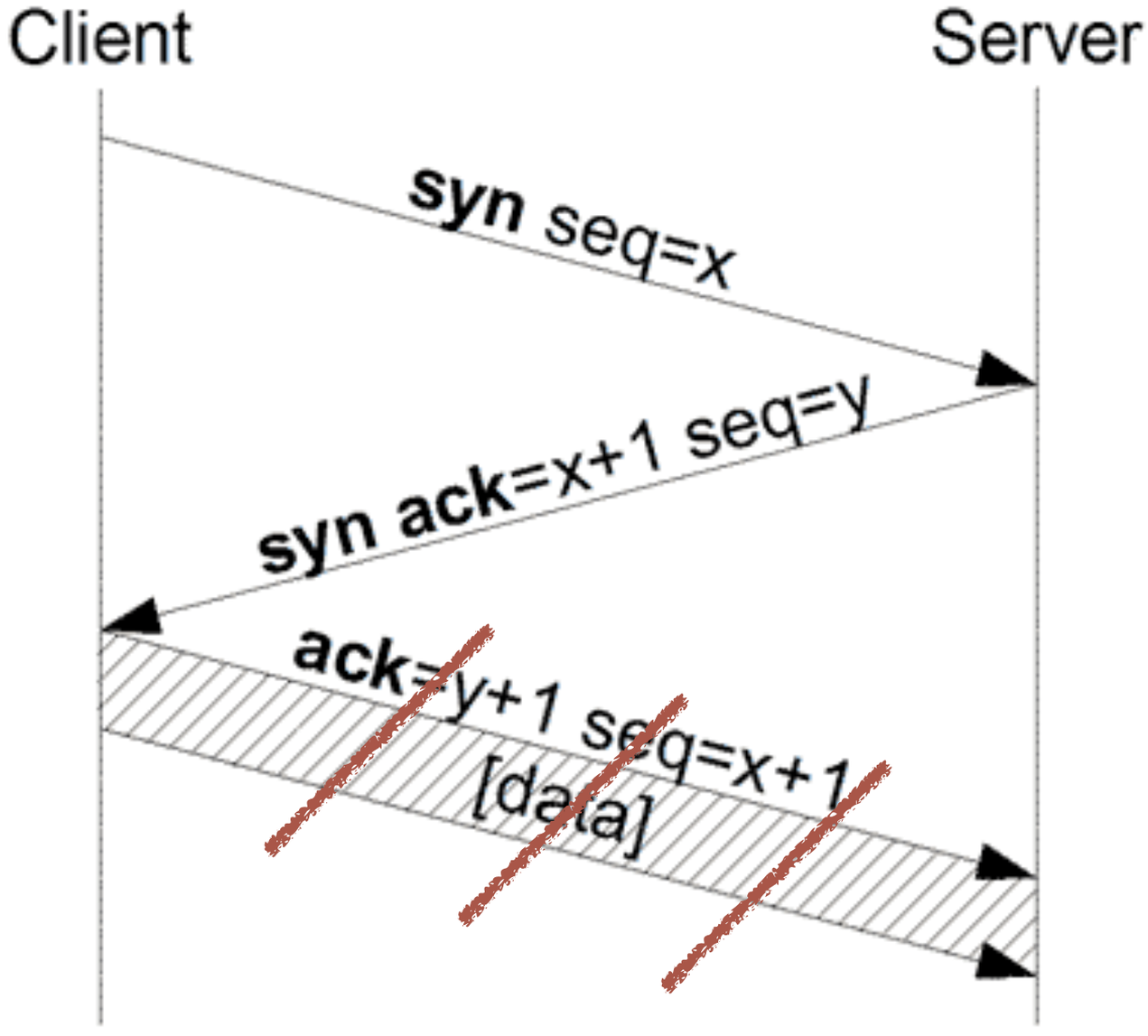
- Spoof TCP packets to A.

TCP RESET attack

- Spoof TCP packet with RST set to 1.
- Remote system should drop connection
- Bypassing IDS/Firewall may require sequence prediction.

TCP SYN flood

- Send large volumes of initial SYN message
- Very cheap
- Ties up buffers at receiving end



Application layer

Domain names

- How do I find the IP address for www.itu.dk?
- Using a UDP query to the Domain-name system
- Premise: You must know *some* nameserver

All layers

All broken

Application	HTTP, DNS, SMTP, FTP,
Transport	TCP, UDP
Network	IP, ICMP
Data link	802.11, 802.3, Bluetooth
Physical	802.11, 802.3, Bluetooth

Denial of Service

“What happens if this botnet actually takes down google.com and we lose all of our revenue?”

– Google

Consume the resources of your computer

- Network bandwidth
- System resources
- Application resources

Ping flood

Send
- max-size ping (-s 65507)
- at max rate (-f)

To overwhelm receiver.

Example on the right:
target not particularly overwhelmed.

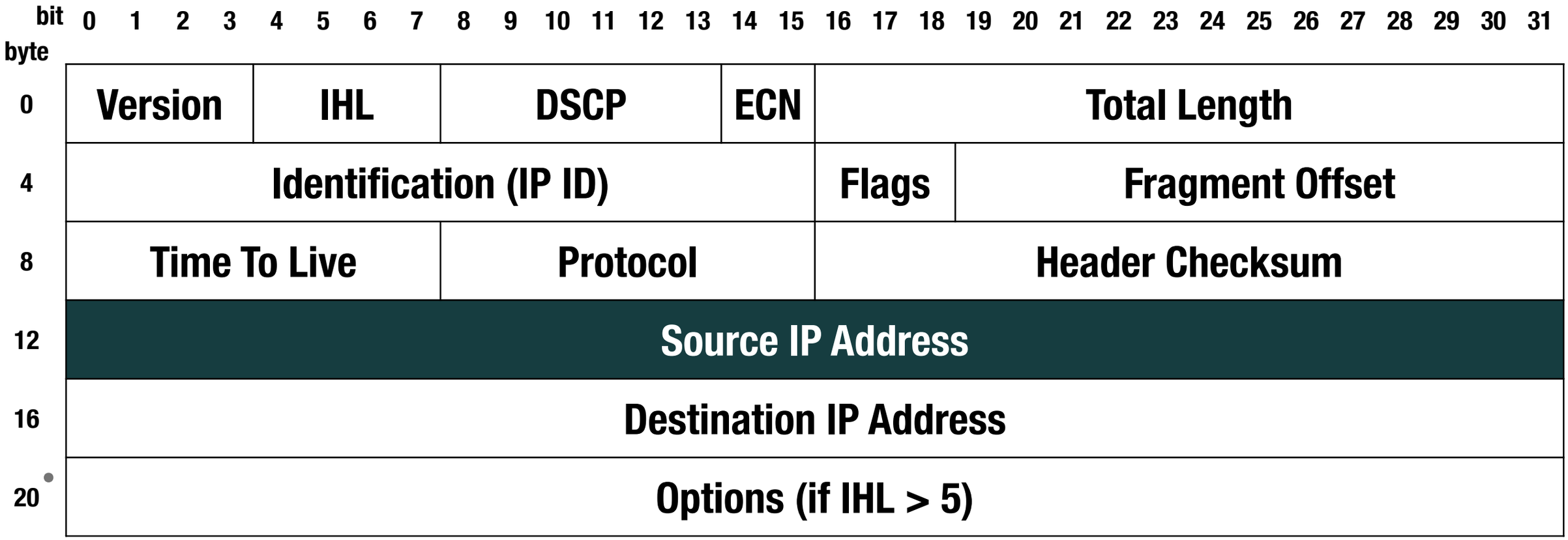
```
~ $ time ping dcr.itu.dk
PING tiger.itu.dk (130.226.142.220): 56 data bytes
64 bytes from 130.226.142.220: icmp_seq=0 ttl=60 time=1.026 ms
64 bytes from 130.226.142.220: icmp_seq=1 ttl=60 time=0.540 ms
64 bytes from 130.226.142.220: icmp_seq=2 ttl=60 time=0.697 ms
64 bytes from 130.226.142.220: icmp_seq=3 ttl=60 time=0.778 ms
64 bytes from 130.226.142.220: icmp_seq=4 ttl=60 time=0.611 ms
^C
--- tiger.itu.dk ping statistics ---
5 packets transmitted, 5 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.540/0.730/1.026/0.168 ms

real    0m4.353s
user    0m0.001s
sys 0m0.003s
~ $ time sudo ping -f -s 65507 dcr.itu.dk
PING tiger.itu.dk (130.226.142.220): 65507 data bytes
..Request timeout for icmp_seq 525
..Request timeout for icmp_seq 792
..Request timeout for icmp_seq 872
.^C
--- tiger.itu.dk ping statistics ---
1368 packets transmitted, 1364 packets received, 0.3% packet loss
round-trip min/avg/max/stddev = 3.401/3.843/10.840/0.475 ms

real    0m4.681s
user    0m0.048s
sys 0m0.180s
~ $
```

Source address spoofing

- Ping packets are returned to sender.
  The adversary wouldn't want that.



UDP flood

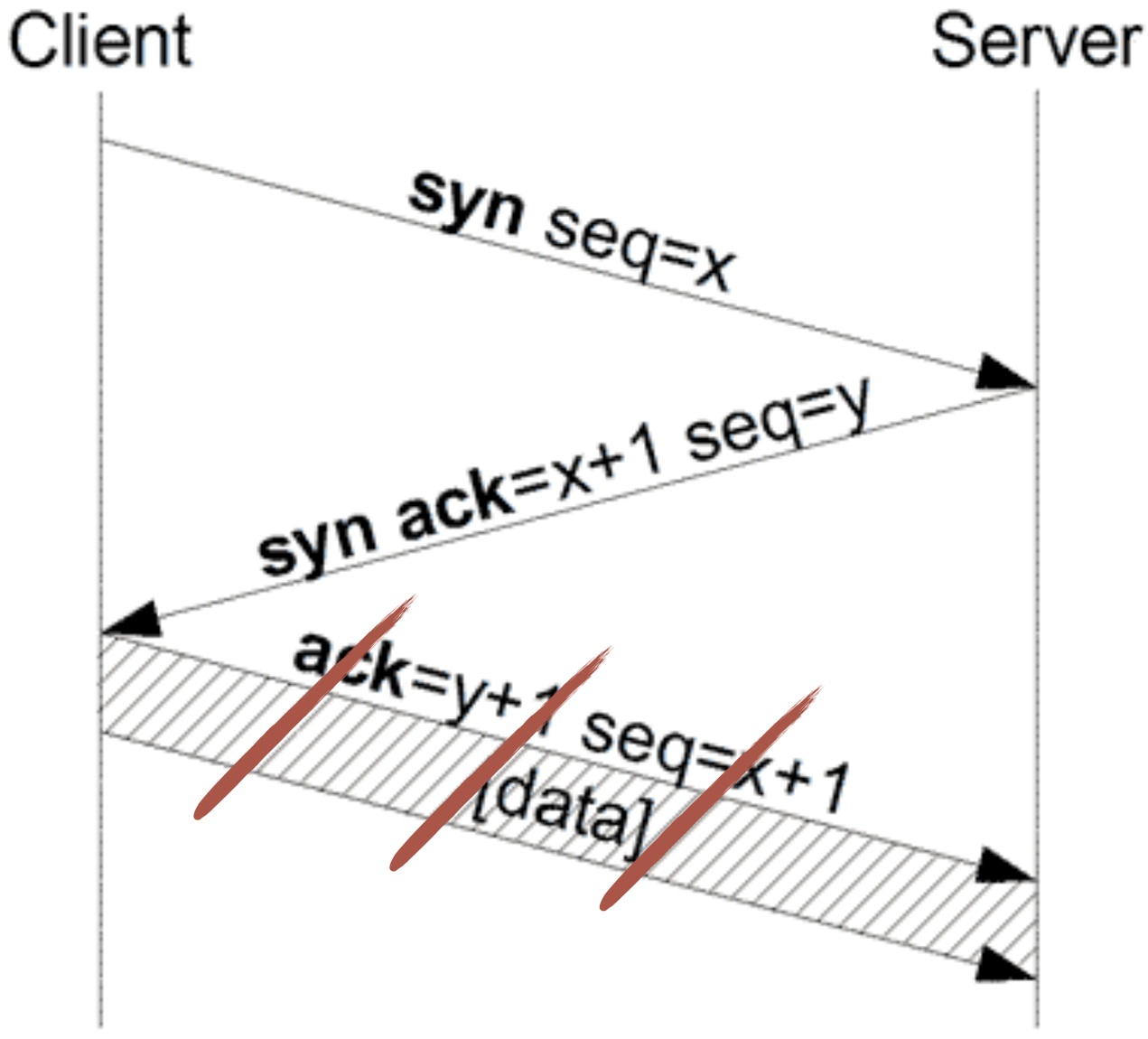
- Same idea as ping flood, but UDP
- Targets specific port
- Target returns UDP-reply or ICMP message
- Consumes both bandwidth and local resources

SYN spoofing

Leave the the 3-way handshake hanging.

Consumes buffers on the server side.

Very cheap.



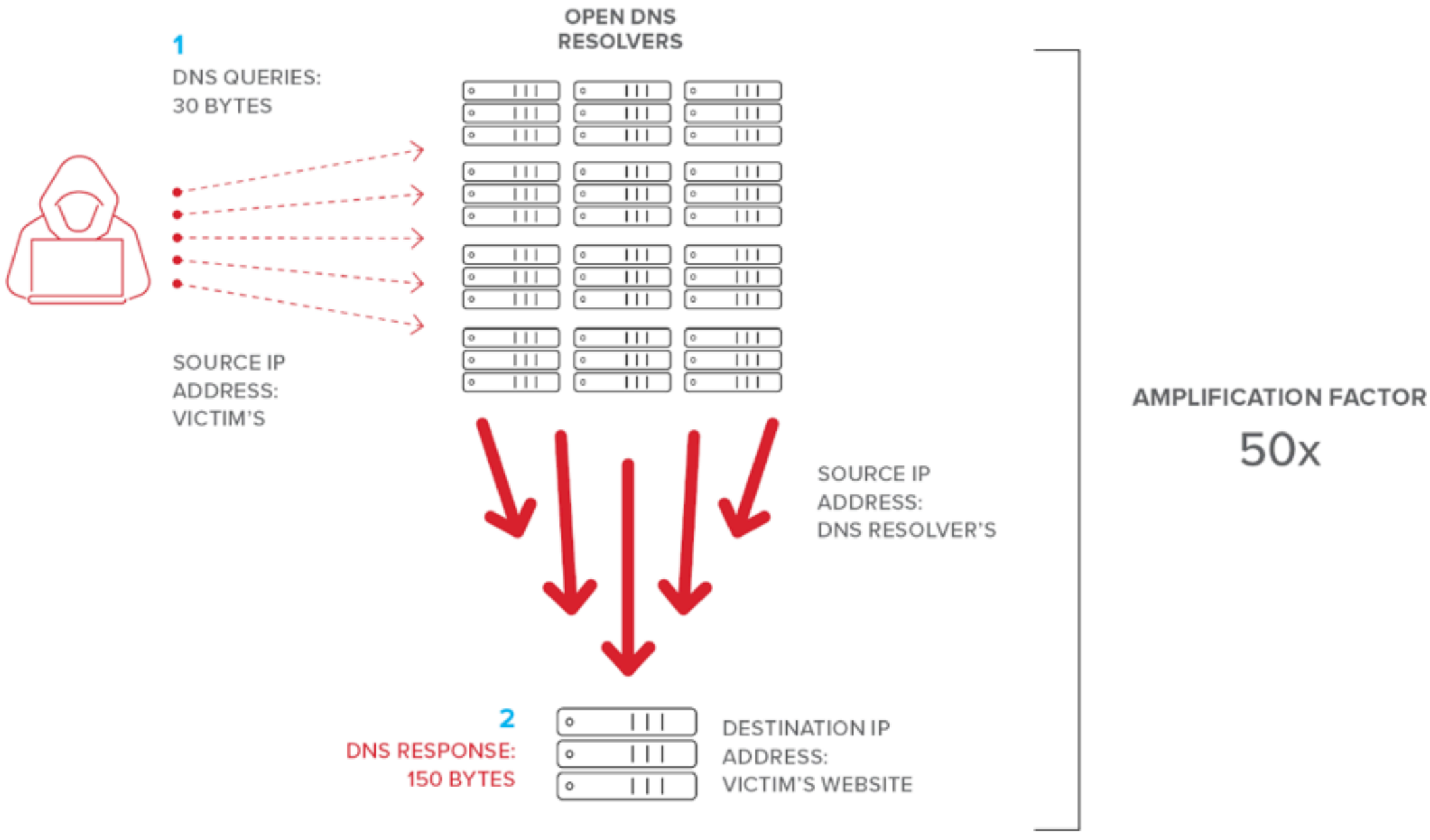
Distributed Denial of Service attacks

- Hack a lot of other devices (zombies)
- Instruct them all (botnet) to DDoS your actual target
- Arrange zombies in a hierarchy. IRC popular for command-and-control (C&C).

Amplification attacks

- Generate *more than one* reply for each spoofed request
- DNS request “any” : 30 bytes
- DNS response to “any”: 150 bytes
- Amplification factor: 50x

Amplification attacks



source: f5.com

In the real world

- 2002: 400 Mbps
- 2010: 100 Gbps
- 2013: 300 Gbps (Spamhaus)
- 2015: 600 Gbps (BBC)
- 2018: 1350 Gbps (github)

Defenses

- Cannot be prevented completely:
  Legitimate traffic; The *Slashdot* effect.

- DNS Reflection:
  Firewall may block replies to requests I didn't send

- Amplification via IP broadcast:
  Firewall blocks broadcasts from outside local network

Defenses (II)

- RFC 2827. Block IP-spoofing at earliest point—local ISP knows if originator IP is legit or not.

- SYN-cookies vs. SYN-spoofing: encode *all* necessary state in SYN-ACK sequence number(!)

- Vs. Application-resources: CAPTCHAs.

Summary

Summary

- Attacks on the network stack
- Denial of service attacks