Chapter 3 Transport Layer

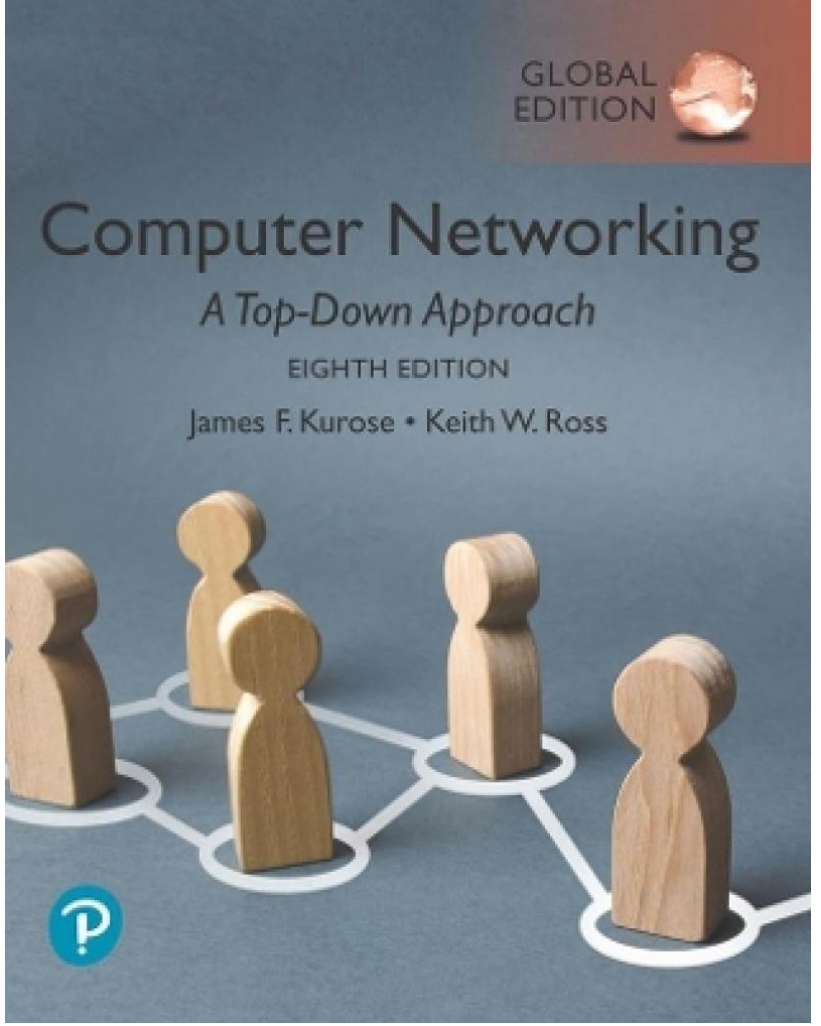
A note on the use of these PowerPoint slides:

We're making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you see the animations; and can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent a *lot* of work on our part. In return for use, we only ask the following:

- If you use these slides (e.g., in a class) that you mention their source (after all, we'd like people to use our book!)
- If you post any -lides on a www site, that you note that they are adapted from (or perhaps identical to) our slides and note our copyright of this material.

For a revision history, see the slide note for this page.

Thanks, and enjoy! JFK/KWR

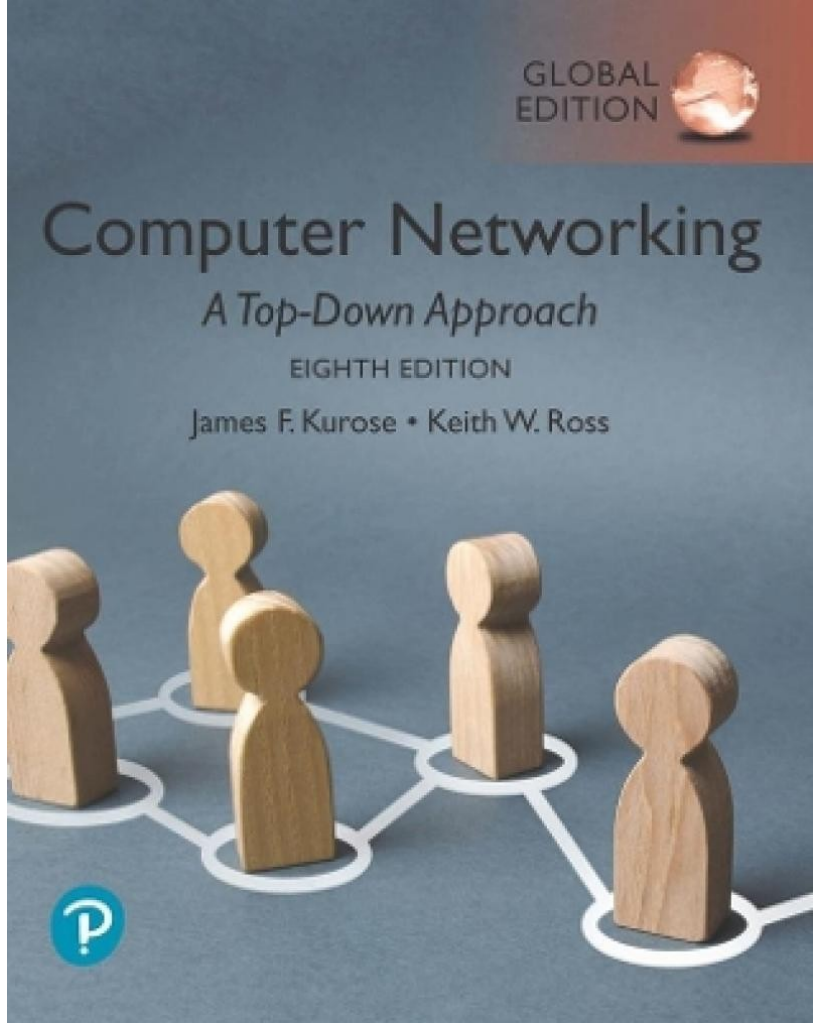


Computer Networking: A Top-Down Approach

8th edition
Jim Kurose, Keith Ross
Pearson, 2020

Transport layer: roadm

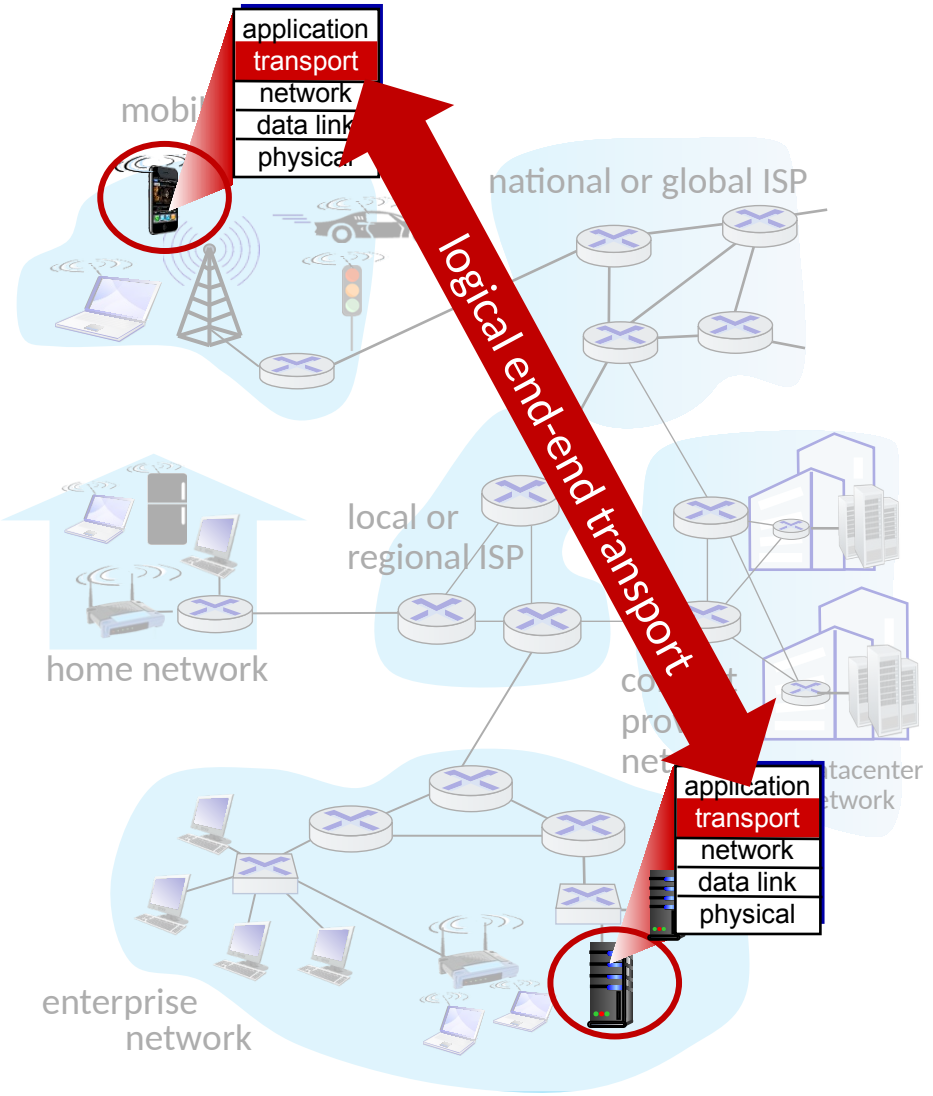
- 3.1 Transport-layer services
- 3.2 Port numbers
- 3.3 Connectionless transport: UDP
  - UDP socket programming
- 3.4 Principles of reliable data transfer
- 3.5 Connection-oriented transport: TCP
- ~~3.6 Principles of congestion control~~
- 3.7 TCP congestion control
- 3.8 QUIC: Quick UDP Internet Connections



3.1 Transport layer

- Transport layer protocols
  - End-to-end protocols
  - Implemented in the communicating end nodes, *not* in the network itself
- provide *logical communication* between *application processes* running on different hosts

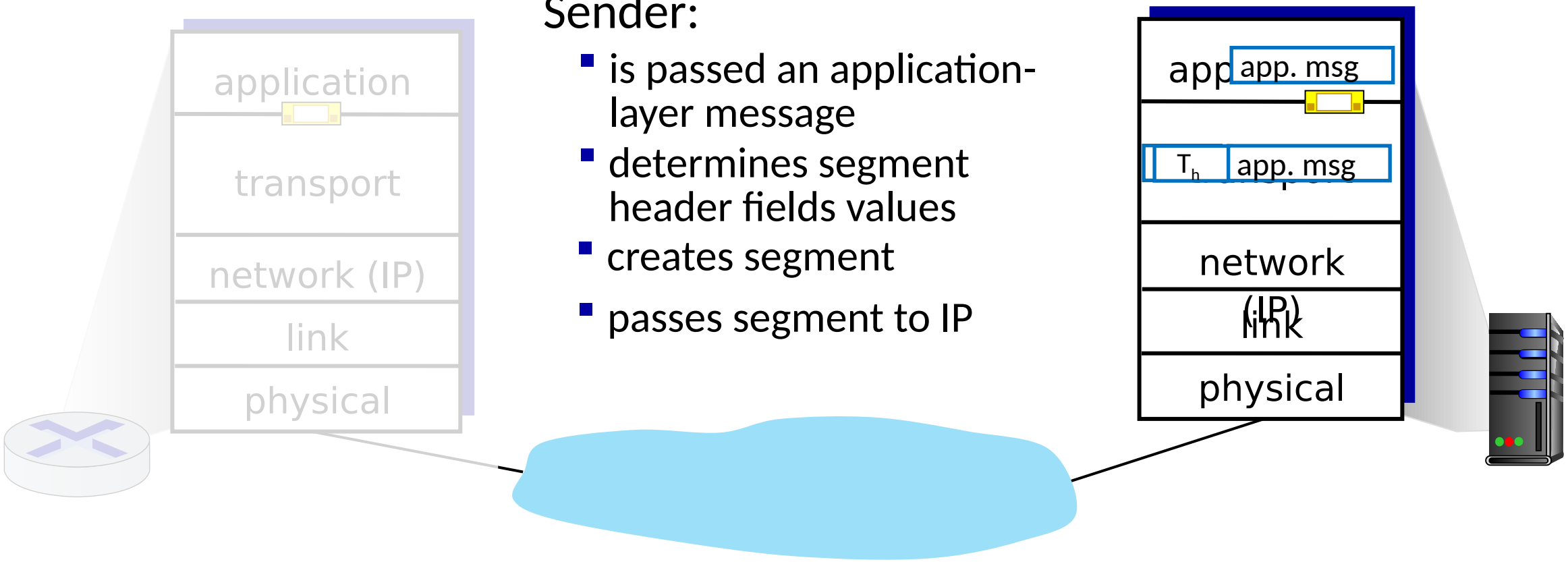
- two transport protocols
  - TCP, UDP



Transport vs. network layer services and protocols

- network layer: logical communication between *hosts*
  - IP addressing
- transport layer: logical communication between *processes*
  - relies on the network layer

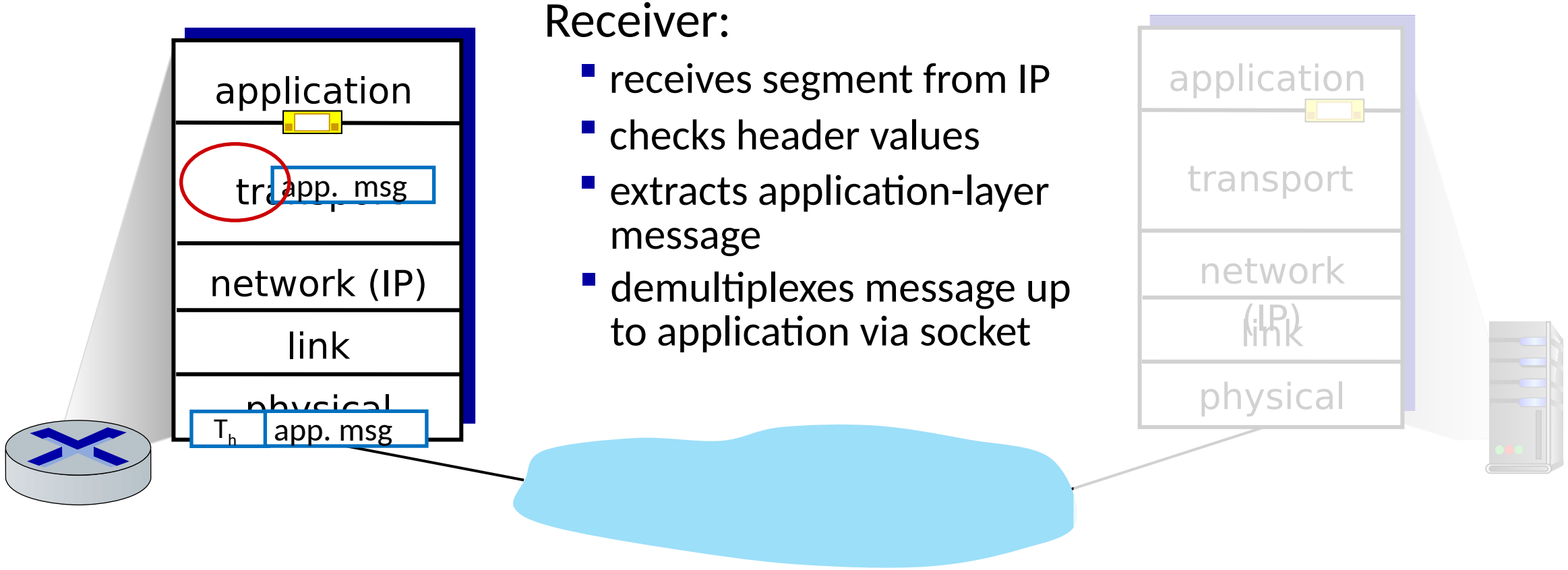
Transport Layer Actions



Sender:

- is passed an application-layer message
- determines segment header fields values
- creates segment
- passes segment to IP

Transport Layer Actions



Receiver:

- receives segment from IP
- checks header values
- extracts application-layer message
- demultiplexes message up to application via socket

Comparison of UDP and TCP

UDP:

- Port numbers
- Integrity check
- Connectionless data transmission
- No data segmentation
- Not reliable data transfer

TCP:

- Port numbers
- Integrity check
- Connection-oriented data transmission
- Data segmentation
- Reliable data transfer
  1. flow control
  2. congestion control

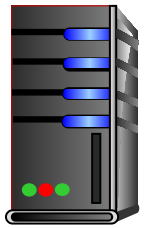
services not available:

delay guarantees and bandwidth guarantees

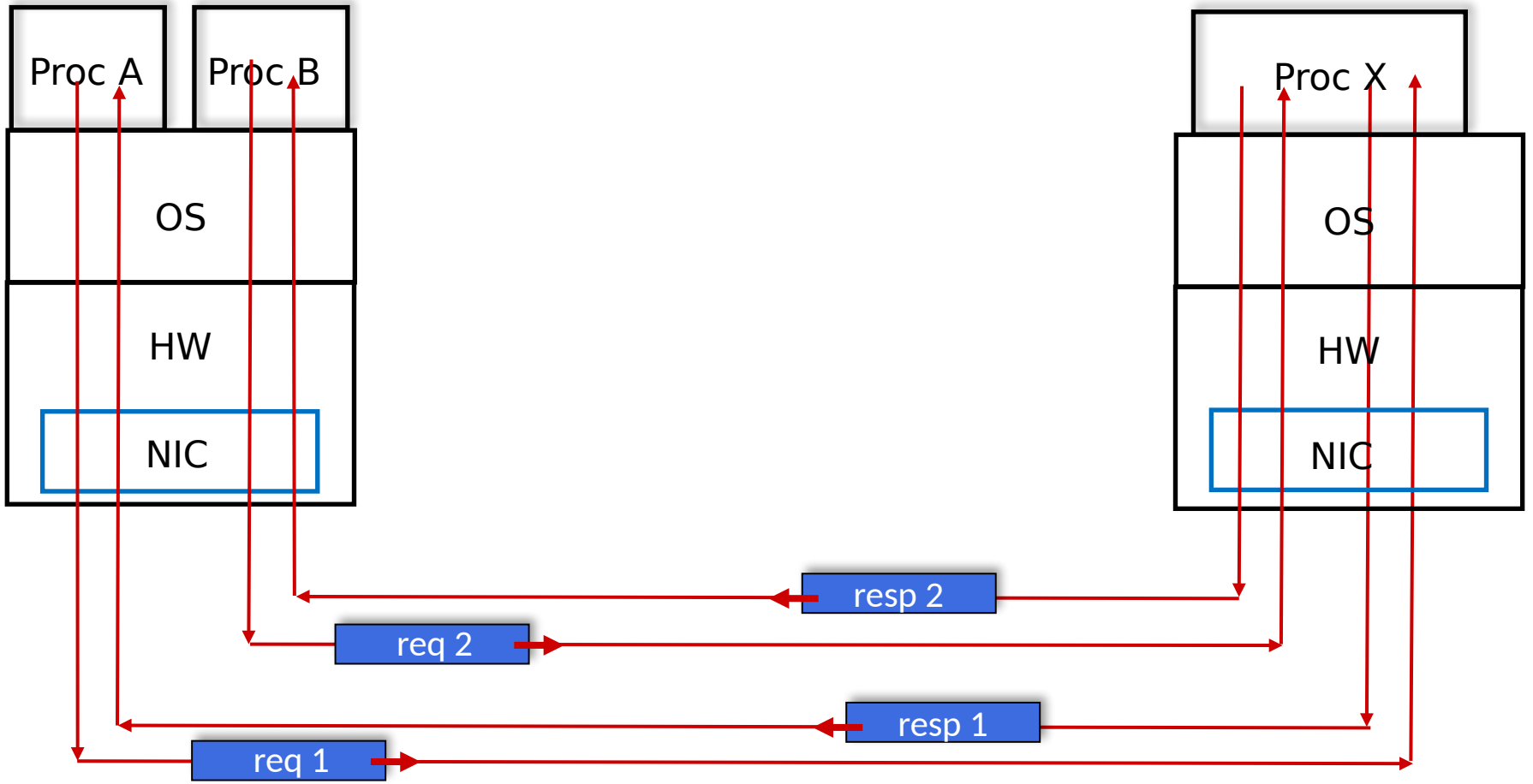
Chapter 3: roadmap

1. Transport-layer services
2. Port numbers
3. Connectionless transport: UDP
   - UDP socket programming
4. Principles of reliable data transfer
5. Connection-oriented transport: TCP
6. ~~Principles of congestion control~~
7. TCP congestion control
8. QUIC: Quick UDP Internet Connections

IP address: 1.2.3.4

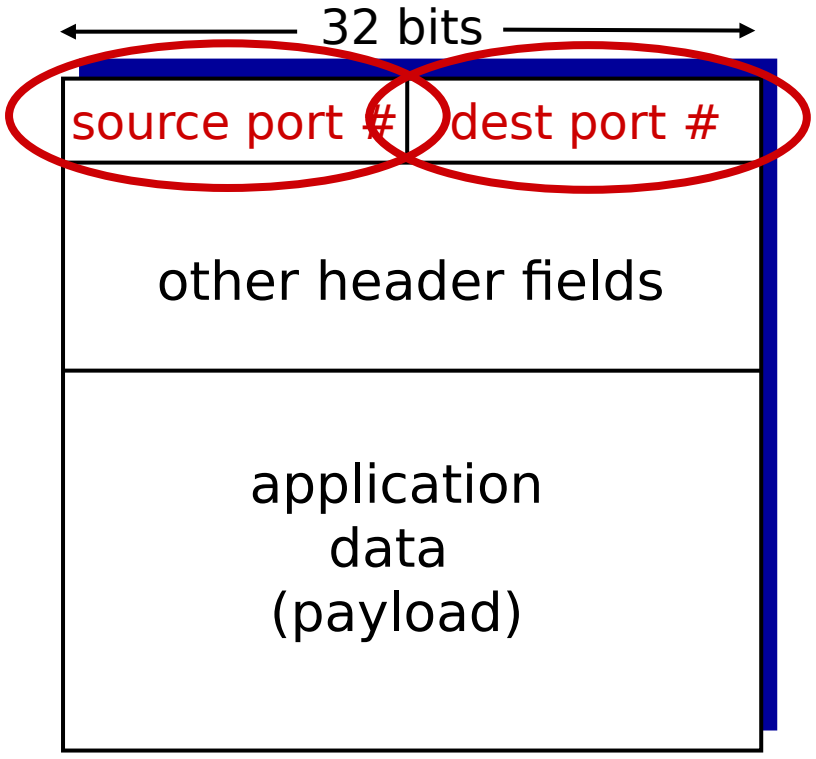


IP address: 3.3.3.3



Port numbers

1. IP addresses identify hosts in the network
2. Port numbers is a method to identify network applications processes
3. Implemented as 16-bit fields in TCP and UDP segments
   - Source port number
   - Destination port number
4. IP packets contain
   - source and target IP addresses
   - an encapsulated UDP or TCP packet (with port numbers)
5. ☾ IP addresses and port numbers are associated
   - Source IP address + Source port number
   - Destination IP address + Destination port number



TCP/UDP segment format

Port numbers

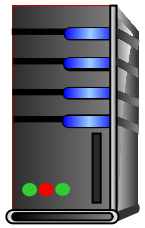
Three categories:

1. Well-known port numbers: 0 – 1023
   - Reserved for widely used protocols and services, e.g. HTTP, HTTPS, SMTP, etc.
2. Registered port numbers: 1024 – 49151
   - Used by specific applications and services.
3. Dynamic/private ports: 49152 – 65535
   - Temporarily assigned for communication between clients and servers.

Some well-known port numbers

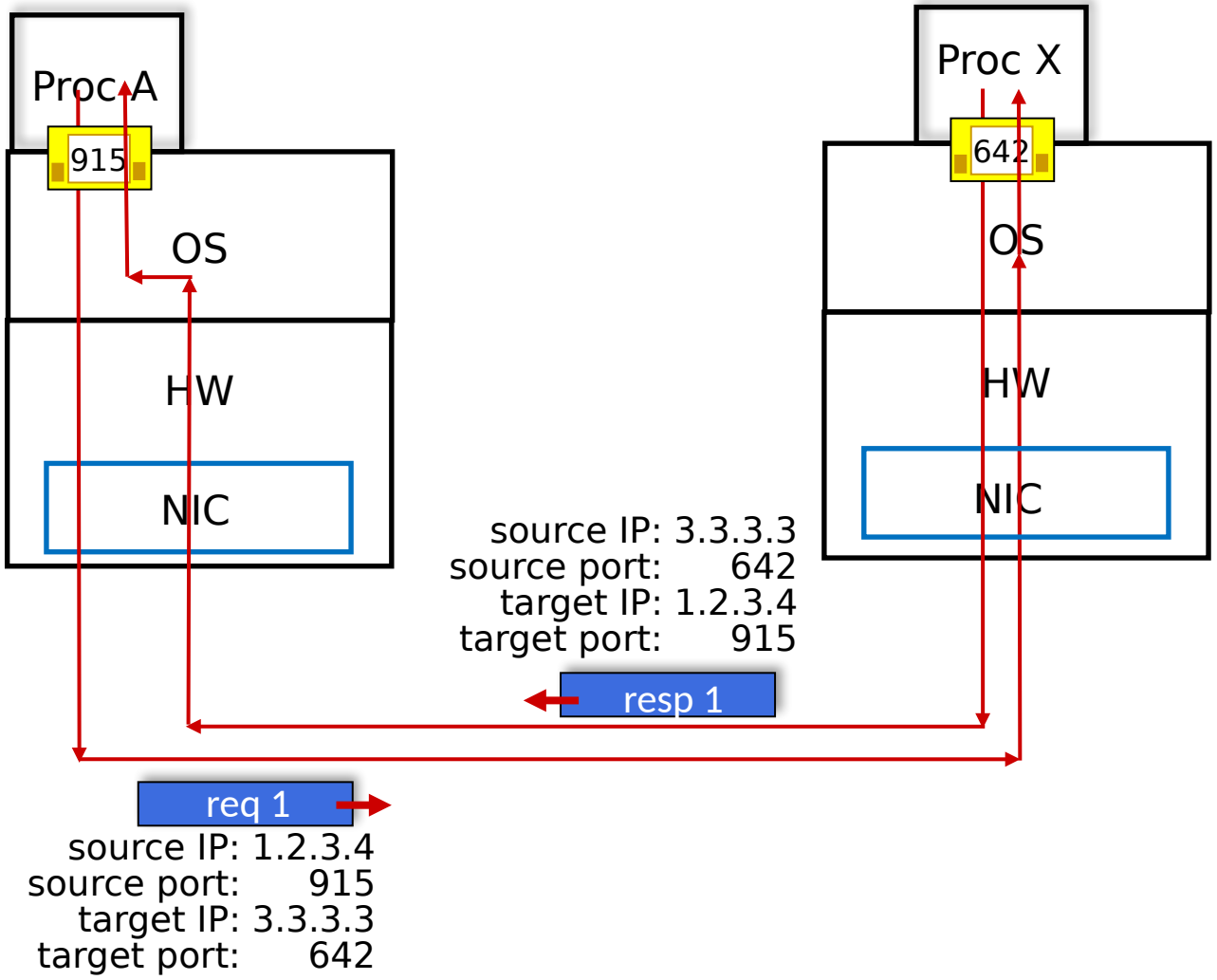
Application protocol	Port number	Protocol
DNS	53	UDP
HTTP (plaintext)	80	TCP
HTTPS (TLS/SSL)	443	TCP
SMTP (plaintext)	25	TCP
SMTP (TLS/SSL)	587, 465	TCP
POP3 (plaintext)	110	TCP
POP3 (TLS/SSL)	995	TCP
IMAP (plaintext)	143	TCP
IMAP (TLS/SSL)	993	TCP

IP address: 1.2.3.4

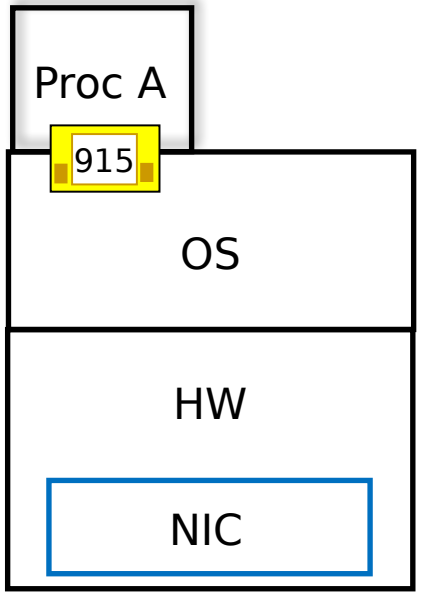


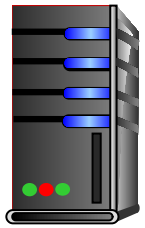
IP address: 3.3.3.3

IP address: 7.8.9.1

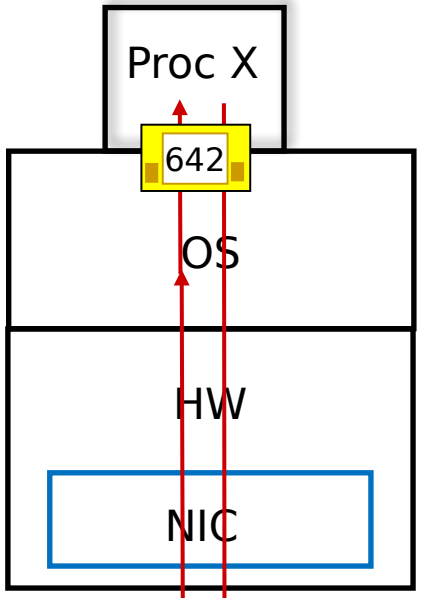


IP address: 1.2.3.4

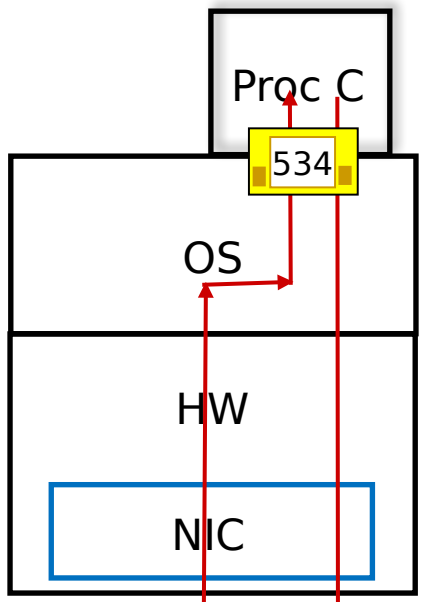




IP address: 3.3.3.3



IP address: 7.8.9.1



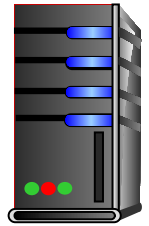
source IP : 3.3.3.3
source port: 643
target IP : 7.8.9.1
target port: 534

resp 2

req 2

source IP: 7.8.9.1
source port: 534
target IP: 3.3.3.3
target port: 643

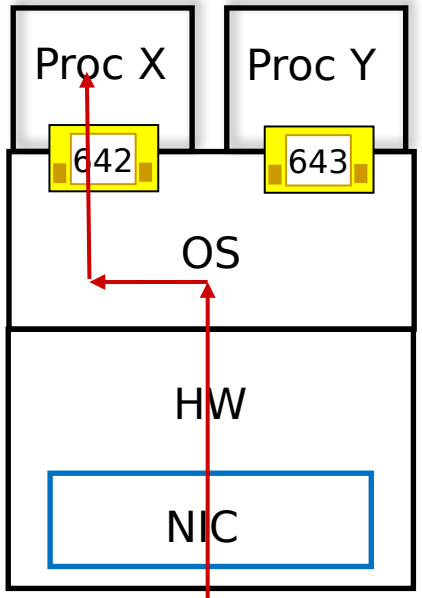
Selecting right process



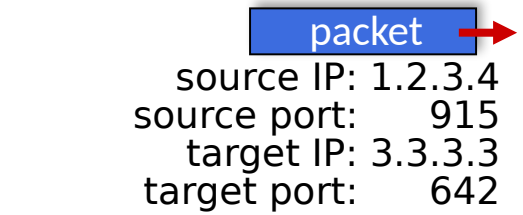
IP address: 3.3.3.3

- A host can have more than one network application processes
- Received segments must be directed to the correct process

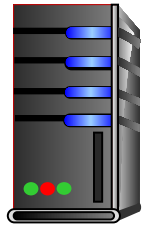
Target port number	Process ID
642	X
643	Y



- The OS determines the target process looking at the target port number

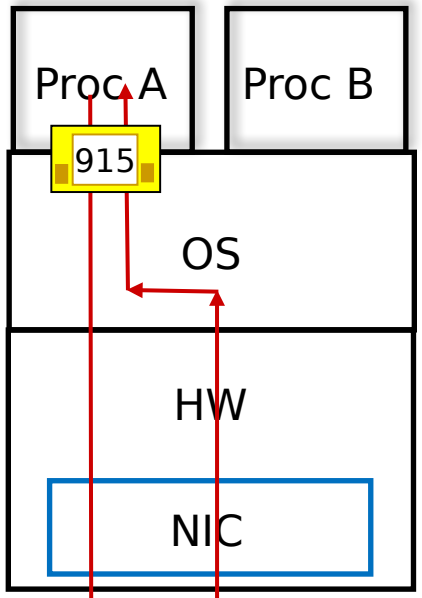


Selecting the right process



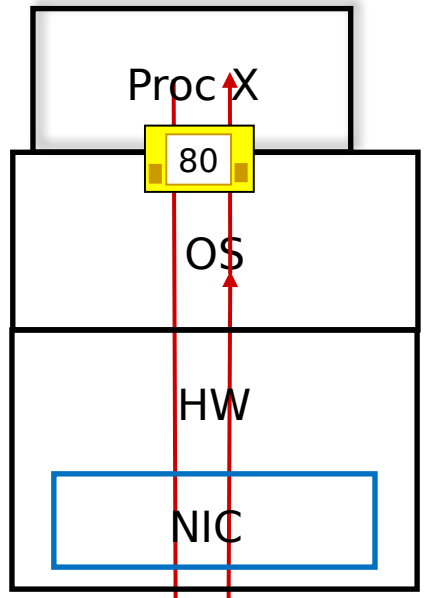
IP address: 1.2.3.4

IP address: 3.3.3.3



Target port number	Process ID
915	A

Target port number	Process ID
80	X



source IP: 3.3.3.3
source port: 80
target IP: 1.2.3.4
target port: 915

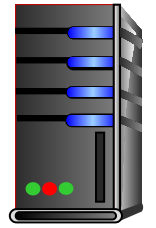
resp 1

req 1

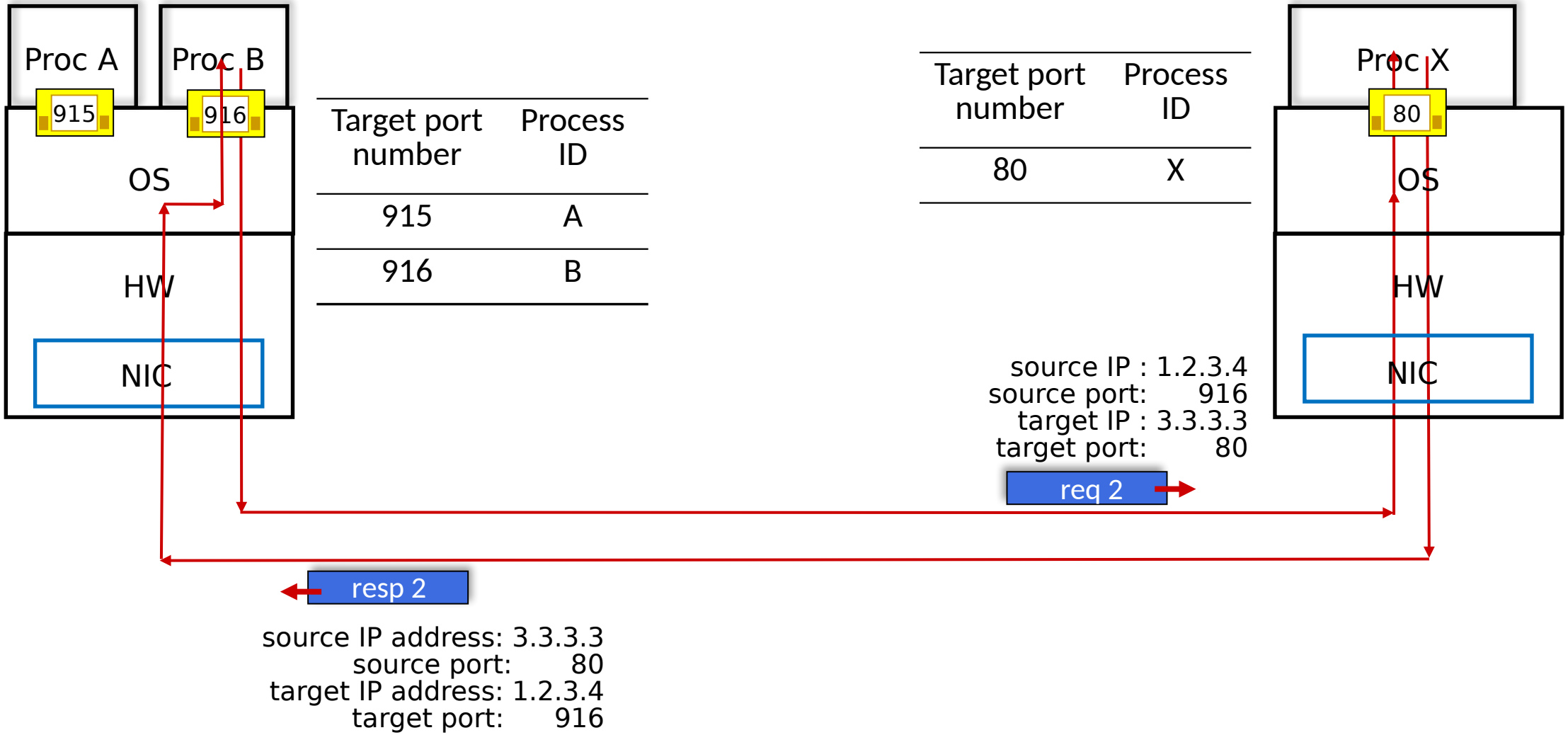
source IP: 1.2.3.4
source port: 915
target IP: 3.3.3.3
target port: 80

Selecting the right process

IP address: 1.2.3.4



IP address: 3.3.3.3



CLIENTS
in a local network (LAN)

SERVER

Network address translation (NAT)

Private network side		Public network side	
Source IP	Source Port	Target IP	NAT Port
1.0.0.1	837	3.3.3.3	267

IP address:
1.0.0.1

source IP: 1.0.0.1
source port: 837
target IP: 3.3.3.3
target port: 80

req 1

837

resp 1

source IP: 3.3.3.3
source port: 80
target IP: 1.0.0.1
target port: 837

Default gateway
IP address:
1.0.0.9

External
IP address:
4.2.1.5

source IP: 4.2.1.5
source port: 267
target IP: 3.3.3.3
target port: 80

req 1

80

resp 1

source IP: 3.3.3.3
source port: 80
target IP: 4.2.1.5
target port: 267

Target

IP address:
3.3.3.3

"Proxy", NAT

IP address:
1.0.0.2

CLIENTS
in a local network (LAN)

Network address translation (NAT)

Private network side		Public network side	
Source IP	Source Port	Target IP	NAT Port
1.0.0.1	837	3.3.3.3	267
1.0.0.2	932	3.3.3.3	413

SERVER

IP address:
1.0.0.1

837

Default gateway
IP address:
1.0.0.9

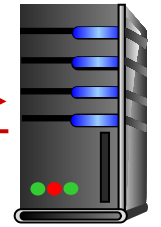
External
IP address:
4.2.1.5

source IP: 4.2.1.5
source port: 413
target IP: 3.3.3.3
target port: 80

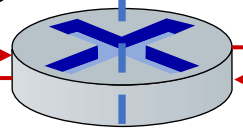
Target

IP address:
3.3.3.3

req 2



80



resp 2

source IP: 1.0.0.2
source port: 932
target IP: 3.3.3.3
target port: 80

req 2

"Proxy", NAT

source IP: 3.3.3.3
source port: 80
target IP: 4.2.1.5
target port: 413

932

resp 2

IP address:
1.0.0.2

source IP: 3.3.3.3
source port: 80
target IP: 1.0.0.2
target port: 932

Chapter 3: roadmap

1. Transport-layer services
2. Port numbers
3. Connectionless transport: UDP
   - UDP socket programming
4. Principles of reliable data transfer
5. Connection-oriented transport: TCP
6. ~~Principles of congestion control~~
7. TCP congestion control
8. QUIC: Quick UDP Internet Connections

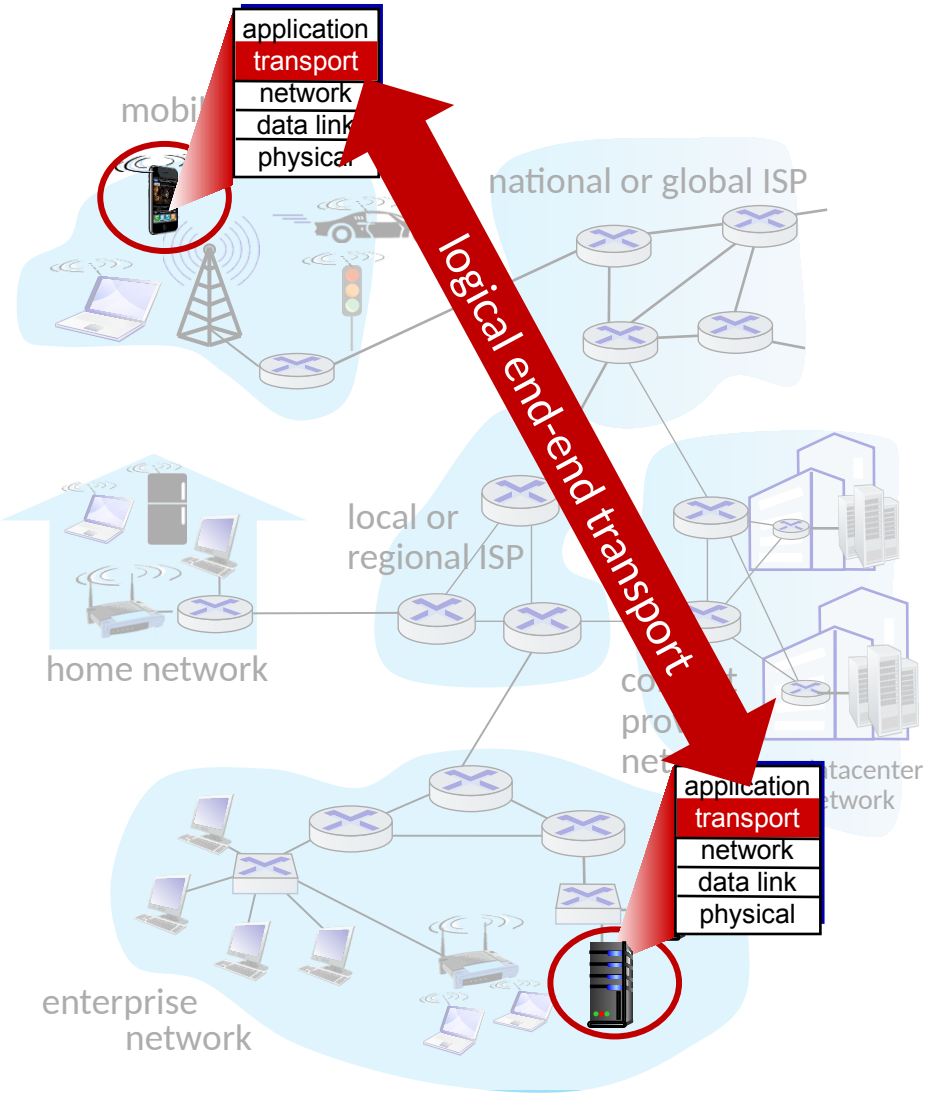
UDP: User Datagram Protocol

- Port numbers
- Integrity check
- Connectionless data transmission
- No data segmentation
- Not reliable data transfer
  - "Best-effort" IP

  - services not available:

    delay guarantees

    bandwidth guarantees



UDP: User Datagram Protocol

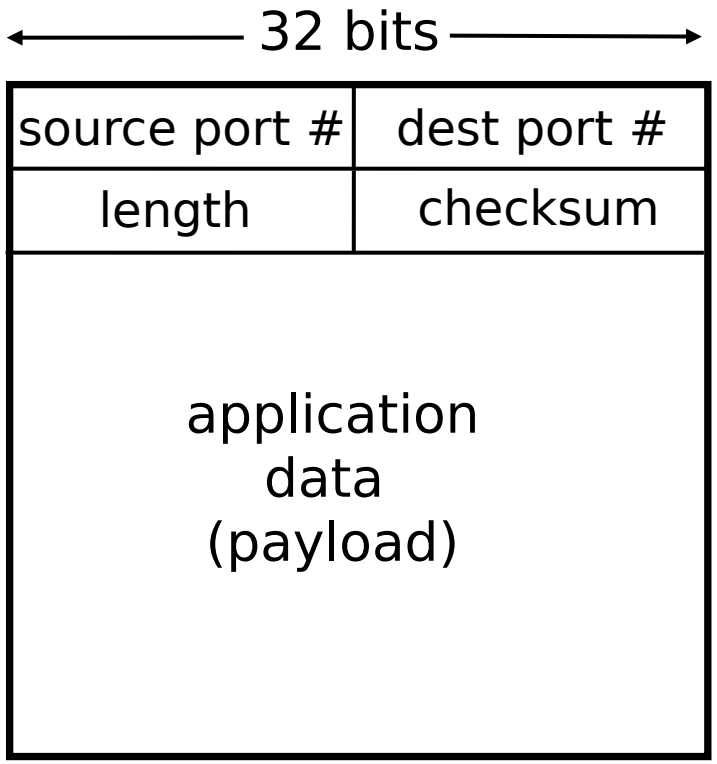
Why is there a UDP?

- no connection establishment (which can add RTT delay)
- simple: no connection state at sender, receiver
- small header size
- no congestion control
  - UDP can blast away as fast as desired!
  - can function in the face of congestion

UDP: User Datagram Protocol

- UDP use:
  - streaming multimedia apps (loss tolerant, rate sensitive)
  - DNS
  - SNMP
  - HTTP/3
- if reliable transfer needed over UDP (e.g., HTTP/3):
  - add needed reliability at application layer
  - add congestion control at application layer

UDP segment header



UDP segment format

UDP: User Datagram Protocol [RFC 768]

INTERNET STANDARD

RFC 768

J. Postel
ISI
28 August 1980

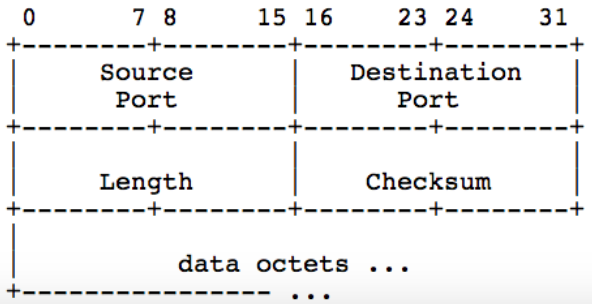
User Datagram Protocol

Introduction

This User Datagram Protocol (UDP) is defined to make available a datagram mode of packet-switched computer communication in the environment of an interconnected set of computer networks. This protocol assumes that the Internet Protocol (IP) [1] is used as the underlying protocol.

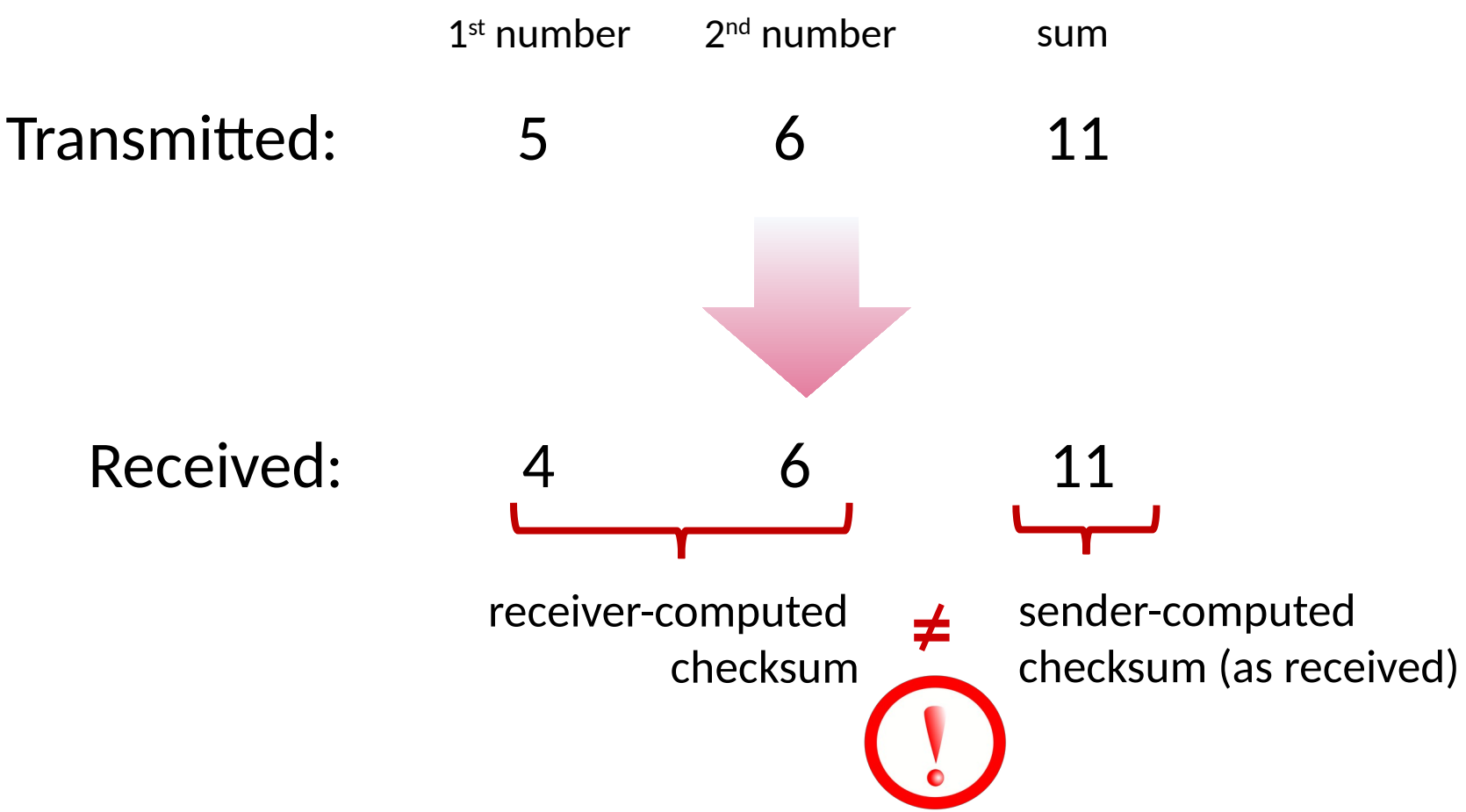
This protocol provides a procedure for application programs to send messages to other programs with a minimum of protocol mechanism. The protocol is transaction oriented, and delivery and duplicate protection are not guaranteed. Applications requiring ordered reliable delivery of streams of data should use the Transmission Control Protocol (TCP) [2].

Format



UDP checksum

Goal: detect errors (*i.e.*, flipped bits) in transmitted segment



UDP checksum

Goal: detect errors (*i.e.*, flipped bits) in transmitted segment

sender:

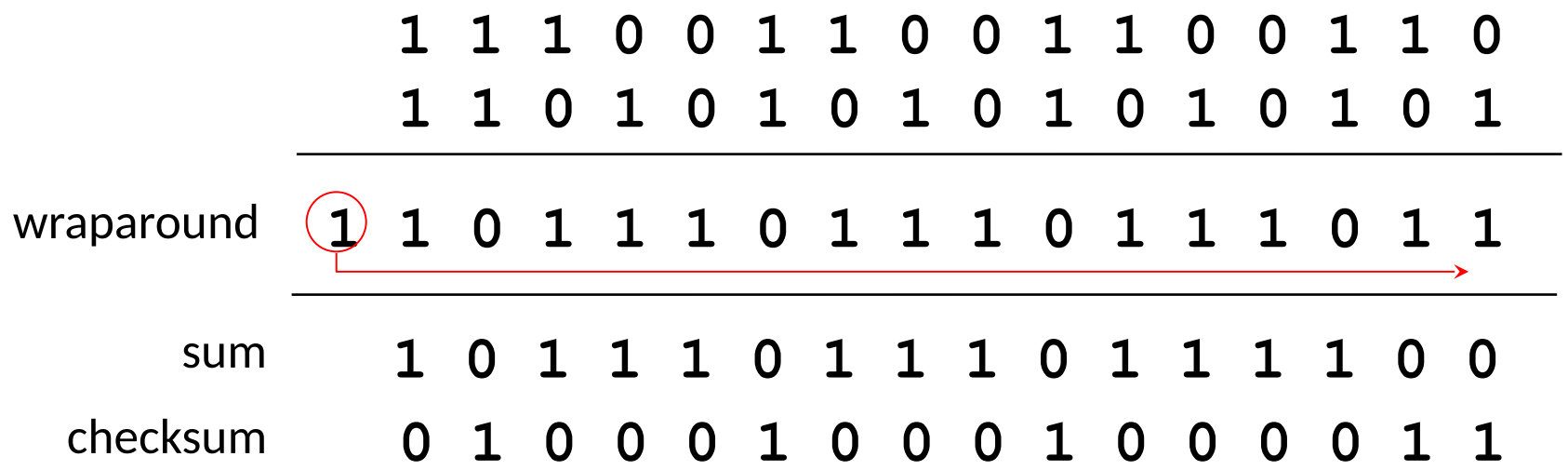
- treat contents of UDP segment (including UDP header fields and a pseudo header from IP header) as sequence of 16-bit integers
- checksum: addition (one's complement sum) of segment content
- checksum value put into UDP checksum field

receiver:

- compute checksum of received segment
- check if computed checksum equals checksum field value:
  - Not equal - error detected
  - Equal - no error detected. *But maybe errors, nonetheless?* More later ….

Internet checksum: an example

example: add two 16-bit integers

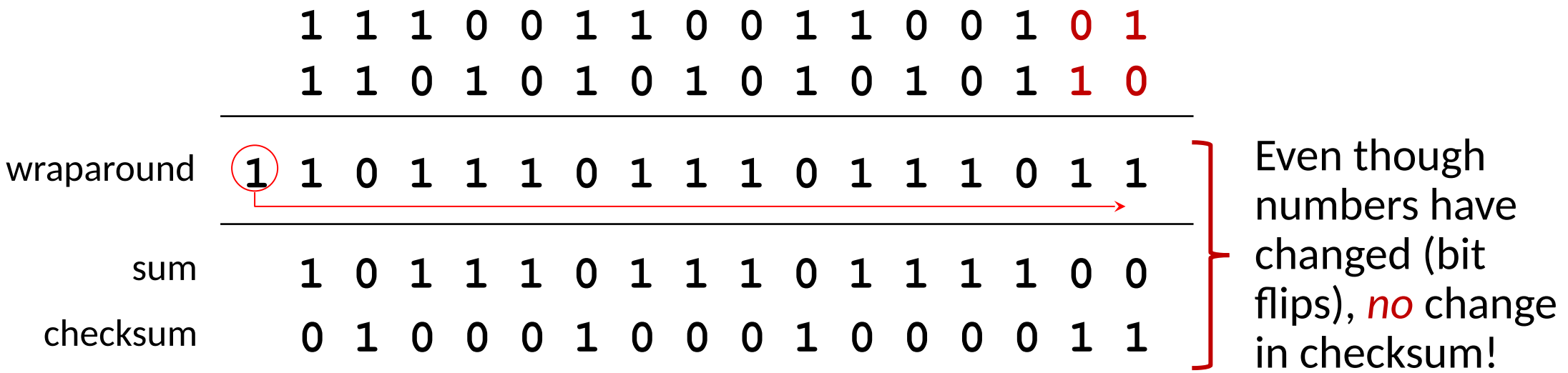


Note: when adding numbers, a carryout from the most significant bit needs to be added to the result

* Check out the online interactive exercises for more examples: http://gaia.cs.umass.edu/kurose_ross/interactive/

Internet checksum: weak protection!

example: add two 16-bit integers



IP loopback address

- The special network address, 127.0.0.1, is defined as a local loopback address
- Hosts use local loopback addresses to send messages to themselves
- In Windows, the name **localhost** is an alias for 127.0.0.1

```
C:\Users\sigurde>tracert 127.0.0.1

Tracing route to UIA5CG4081L51 [127.0.0.1]
over a maximum of 30 hops:

  1    <1 ms    <1 ms    <1 ms  UIA5CG4081L51

Trace complete.

C:\Users\sigurde>tracert localhost

Tracing route to UIA5CG4081L51 [::1]
over a maximum of 30 hops:

  1    <1 ms    <1 ms    <1 ms  UIA5CG4081L51

Trace complete.
```

Sending and receiving UDP packets using ncat

- Two command prompt windows

SENDER

```
C:\> ncat –u 127.0.0.1 8888
```

RECEIVER

```
C:\> ncat -l -u 8888
```

Active connections – TCP ports

- A TCP or UDP process *listens* on a local port

```
C:\> netstat -a

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    10.0.0.9:54747         20.238.236.234:https   ESTABLISHED
  TCP    10.0.0.9:54909         104.18.35.23:https     ESTABLISHED
  TCP    127.0.0.1:49201        UIA5CG4081L51:0        LISTENING
```

- netstat lists the names of the processes that are listening using –b
  - Requires elevated command prompt

Active connections - UDP-ports

- A network process *listens* on a local port

```
C:\> netstat -a p udp

Active Connections

  Proto  Local Address          Foreign Address        State
  UDP    0.0.0.0:5355           *:*
  UDP    0.0.0.0:8888           *:*
  UDP    0.0.0.0:50080          *:*
  UDP    0.0.0.0:52068          127.0.0.1:52067
```

Scanning for open UDP ports

```
C:\Users\sigurde>nmap -sU -p 8888 localhost
Starting Nmap 7.95 ( https://nmap.org ) at 2025-01-31 10:56 W. Europe S
e
Nmap scan report for localhost (127.0.0.1)
Host is up.
Other addresses for localhost (not scanned): ::1

PORT     STATE         SERVICE
8888/udp open|filtered ddi-udp-1

Nmap done: 1 IP address (1 host up) scanned in 2.34 seconds
```

UDP socket programming