CS 352
User Datagram Protocol

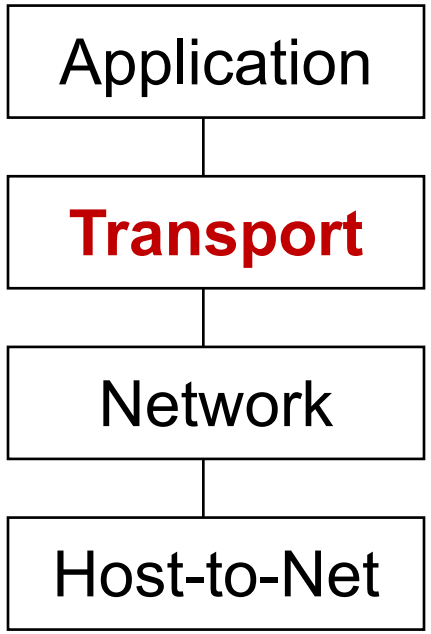
CS 352, Lecture 8.1

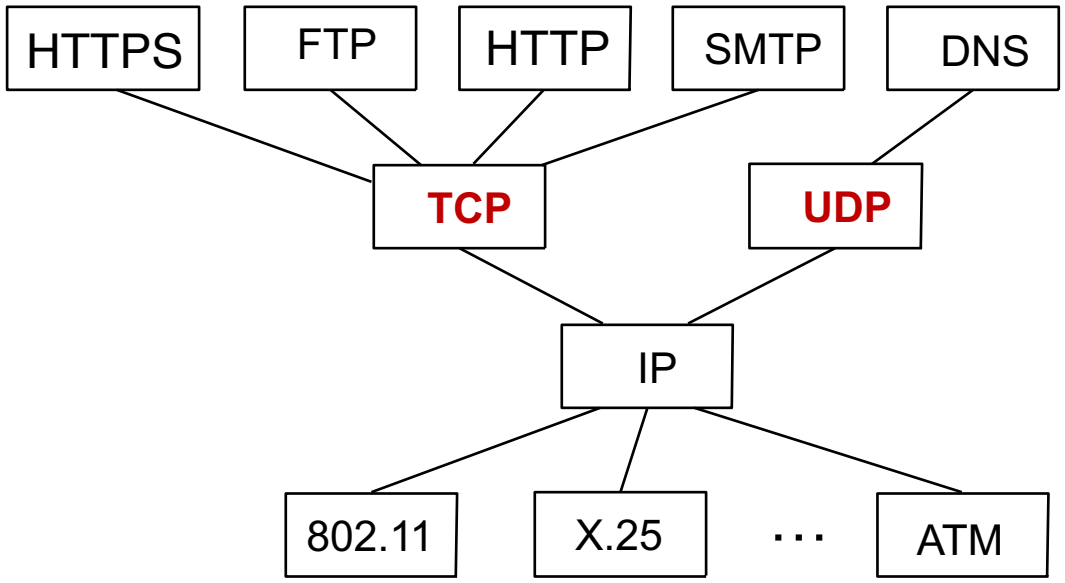
http://www.cs.rutgers.edu/~sn624/352

Srinivas Narayana

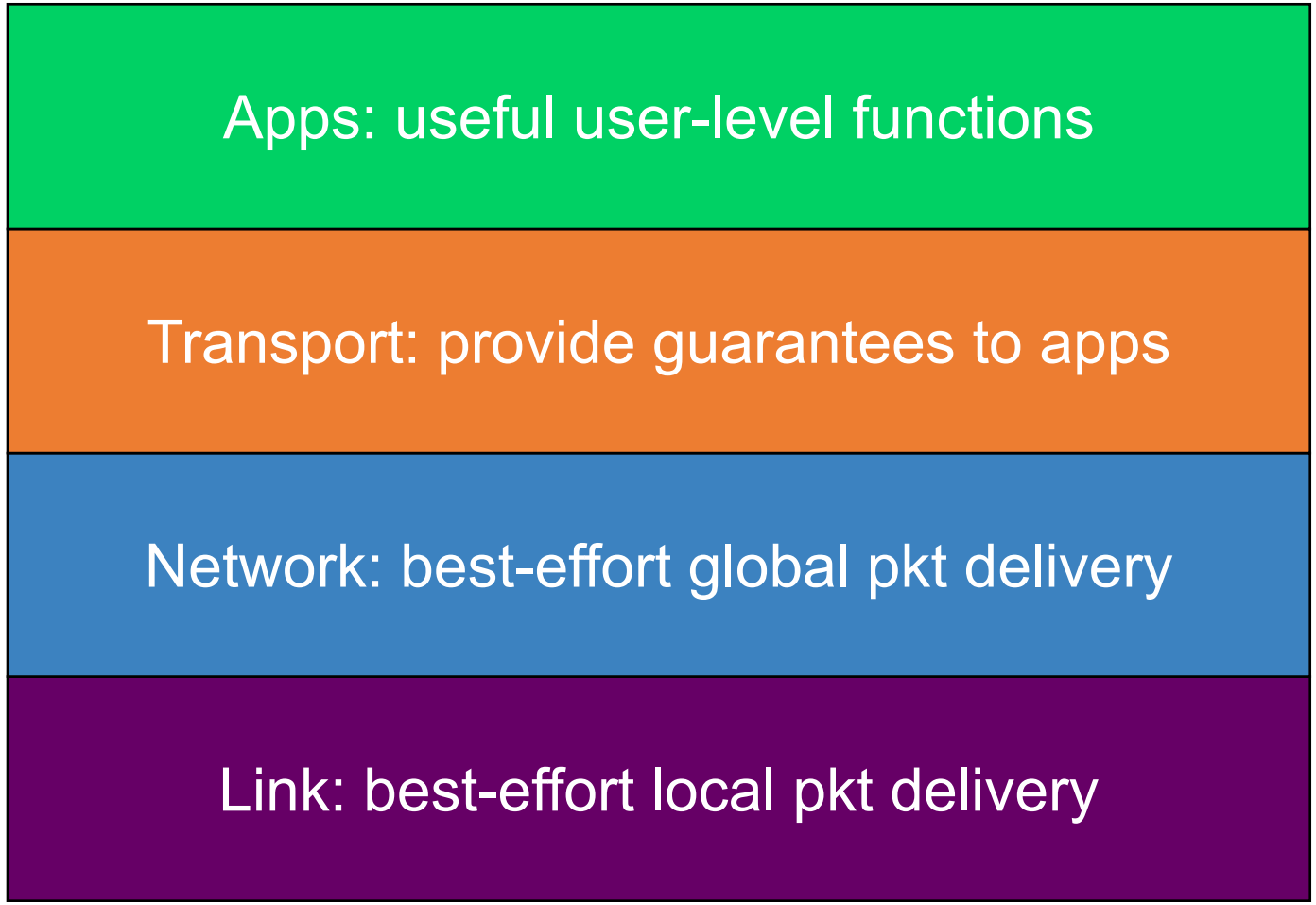
RUTGERS
UNIVERSITY | NEW BRUNSWICK

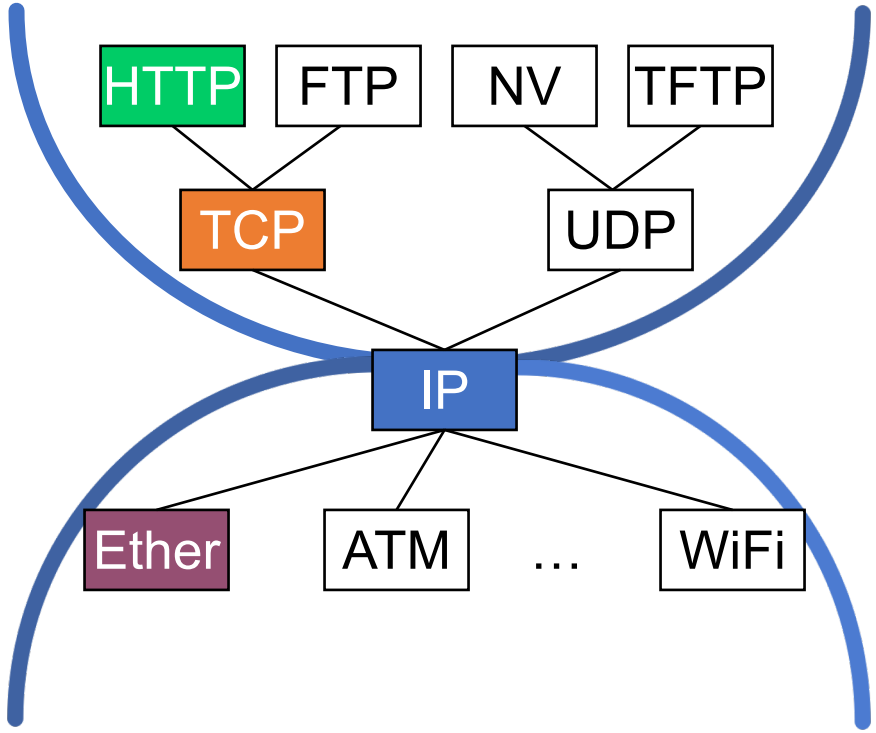
Transport


Application
Transport
Network
Host-to-Net


HTTPS
FTP
HTTP
SMTP
DNS
TCP
UDP
IP
802.11
X.25
...
ATM

Modularity through layering





UDP: User Datagram Protocol [RFC 768]

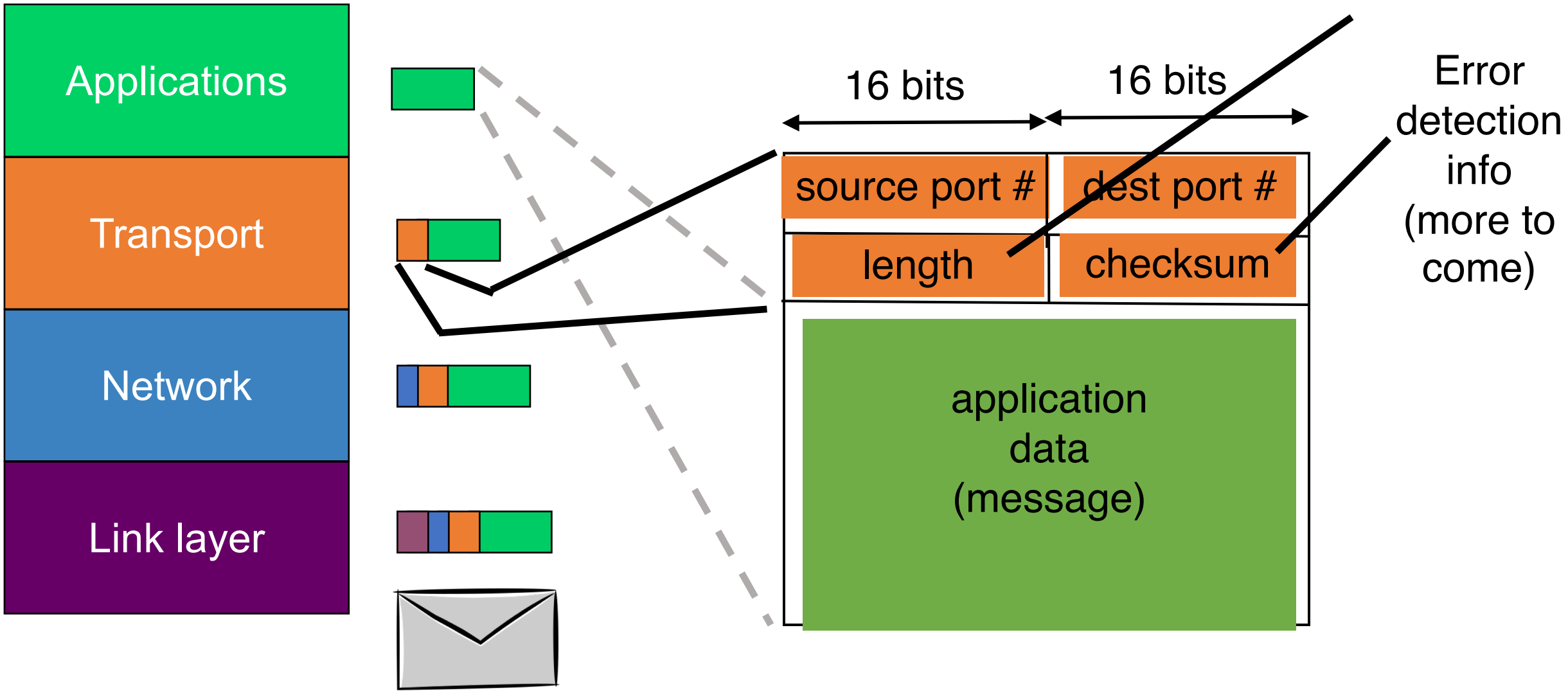
- Best effort service. UDP segments may be:
  - Lost
  - Delivered out of order to app
- UDP is connectionless
  - Each UDP segment handled independently of others (i.e. no "memory" across packets)
- Suitable for one-off req/resp
  - E.g., DNS uses UDP
- Also for loss-tolerant delay-sensitive apps, e.g., video calling

Why are UDP's guarantees even okay?

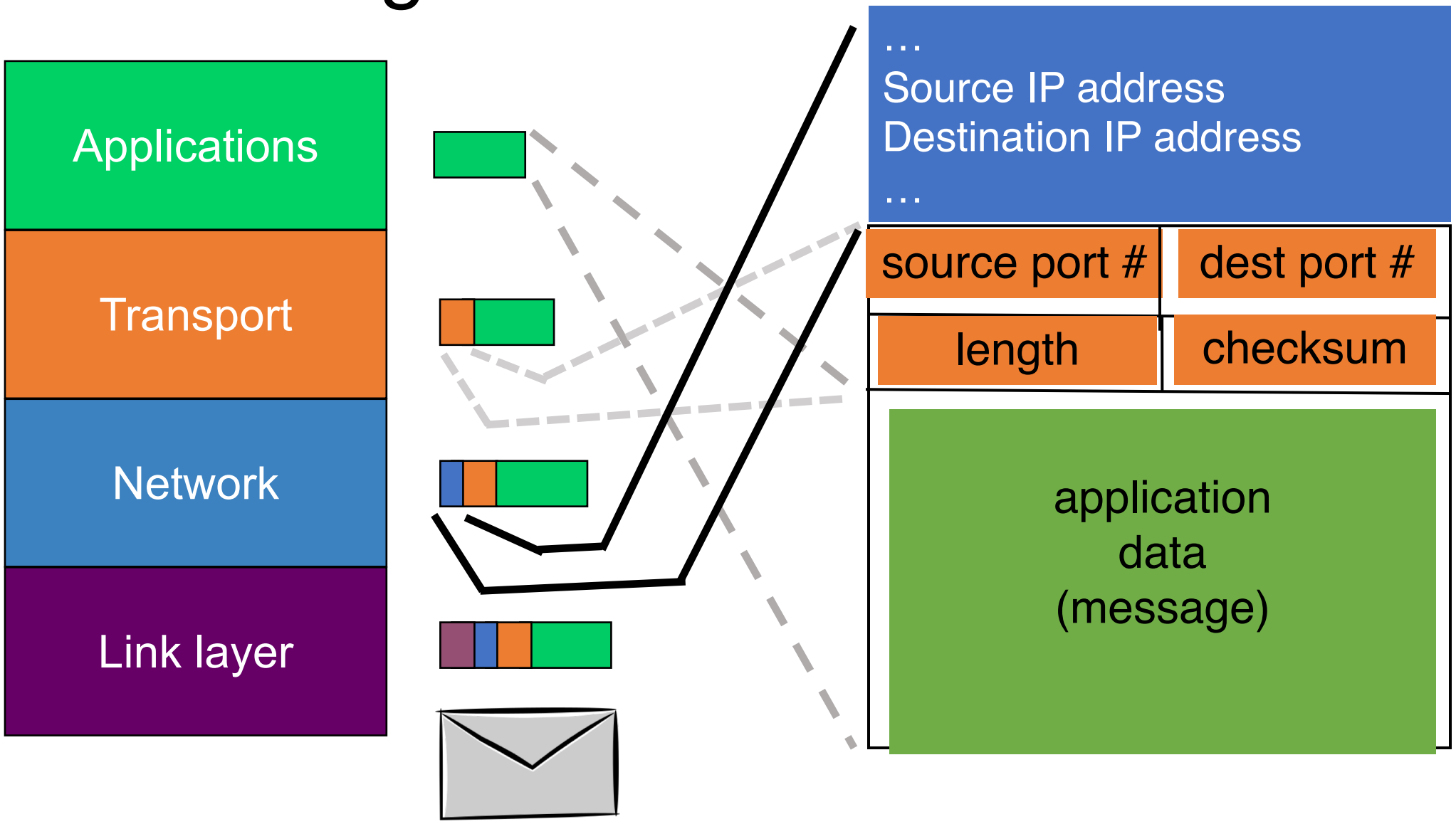
Simple & low overhead compared to TCP:

- No delays due to connection establishment
  - UDP can send data immediately
- No memory for connection state at sender & receiver
- Small segment header
- UDP can blast away data as fast as desired
  - UDP has no "congestion control"

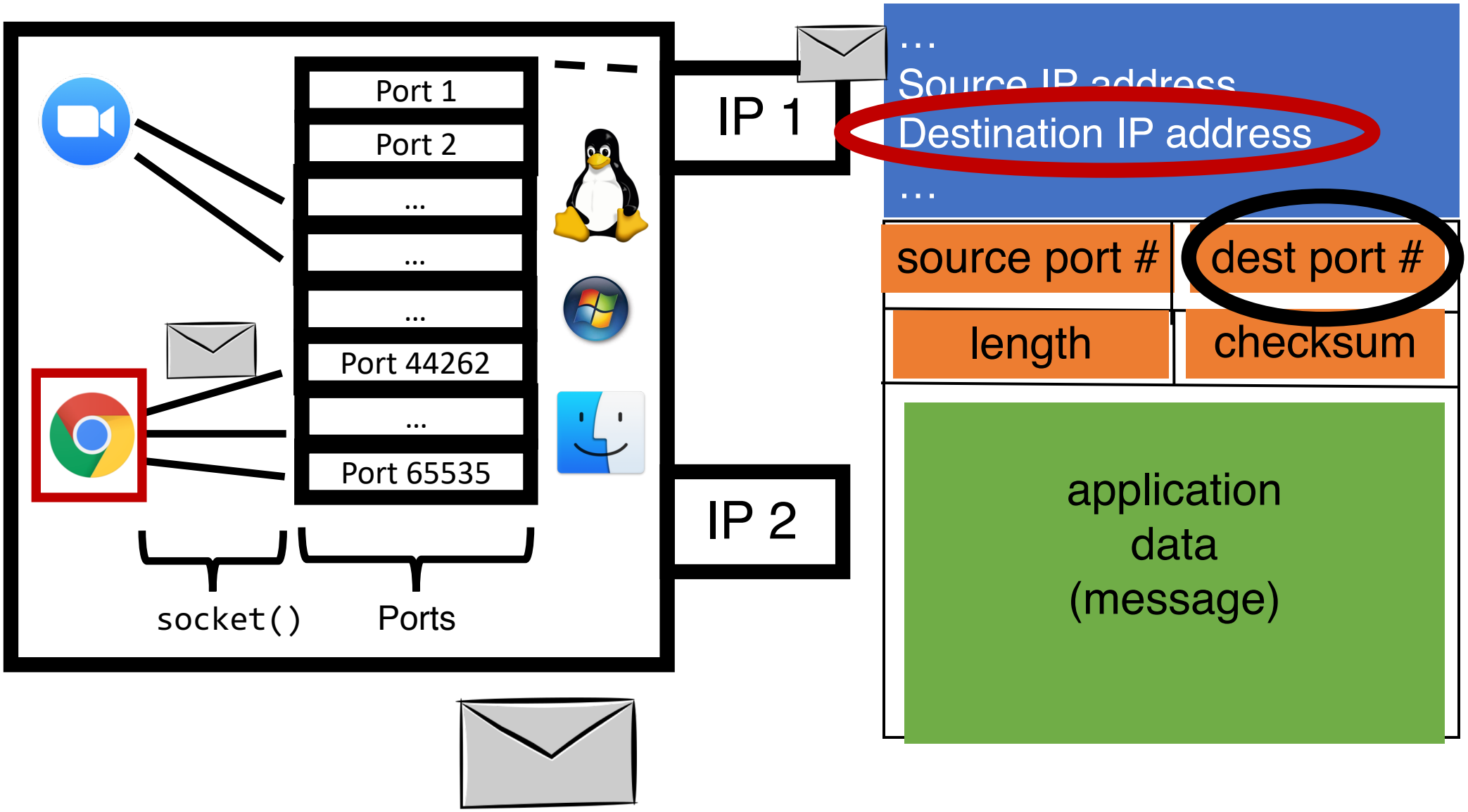
UDP segment structure



UDP segment structure



Review: UDP demultiplexing



UDP packets

- A small demo

CS 352
UDP: Error Detection

CS 352, Lecture 8.2

http://www.cs.rutgers.edu/~sn624/352

Srinivas Narayana

RUTGERS
UNIVERSITY | NEW BRUNSWICK

UDP: Best Effort Service

- Simple and low overhead transport: connectionless
- Data may be lost
- Data may be corrupted along the way (e.g., 1 -> 0)
- Data may be reordered
- However, simple error detection is possible.

UDP Error Detection

- Key idea: have sender compute a function over the data
  - Store the result in the packet
  - Receiver can check the function's value in received packet

- An analogy: you're sending a package of goodies and want your recipient to know if goodies were leaked along the way

- Your idea: weigh the package; stamp the weight on the package
  - Have the recipient weigh the package and cross-check the weight with the stamped value

Error detection function

- Function must be easy to compute
- Function must capture the likely changes to the packet
  - If the packet was corrupted through these likely changes, the function value must change
- Function must be easy to verify

- UDP uses a function called a checksum
  - Very common idea: used in multiple parts of networks and computer systems

UDP Checksum

Sender:

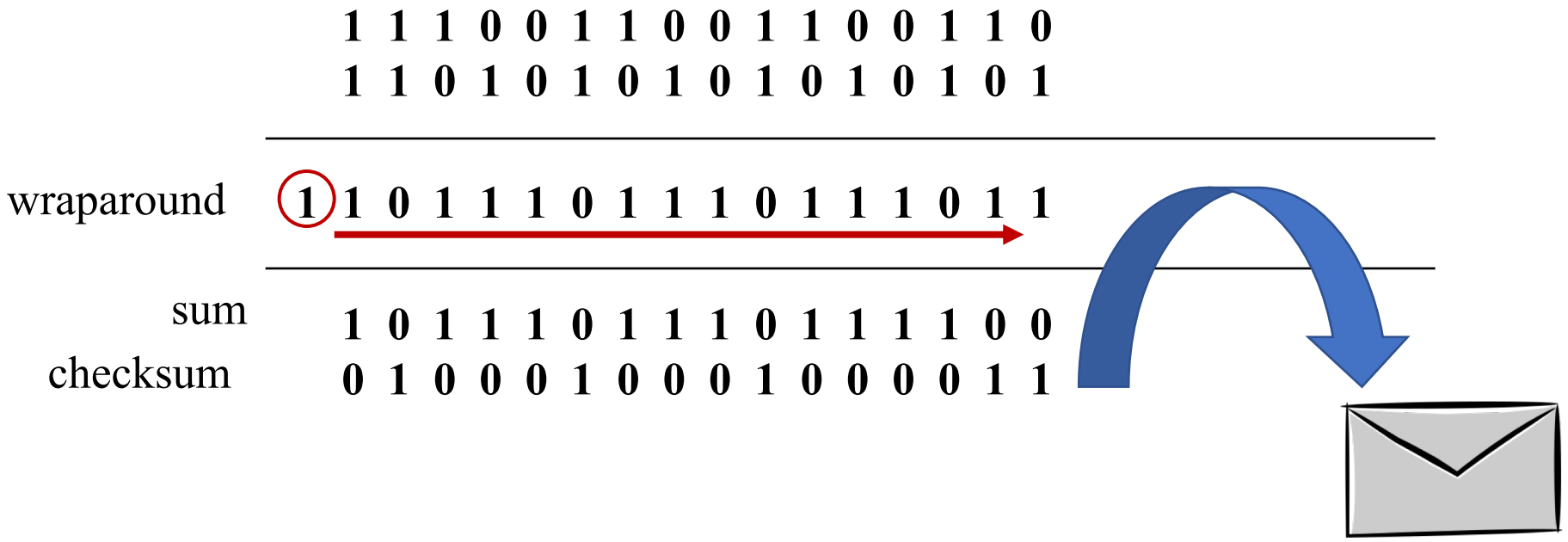
- treat segment contents as sequence of 16-bit integers
- checksum: addition (1's complement sum) of segment contents
- sender puts checksum value into UDP checksum field

Receiver:

- compute a checksum of the received segment, including the checksum in packet itself
- check if the resulting (computed) checksum is 0
- NO – an error is detected
- YES – *assume* no error

Computing 1's complement sum

- Very similar to regular (unsigned) binary addition.
- However, when adding numbers, a carryout from the most significant bit needs to be added to the result
- Example: add two 16-bit integers



From the UDP specification (RFC 768)

- Checksum is the 16-bit one's complement of the one's complement sum of a pseudo header of information from the IP header, the UDP header, and the data, padded with zero octets at the end (if necessary) to make a multiple of two octets.

- The pseudo header conceptually prefixed to the UDP header contains the source address, the destination address, the protocol, and the UDP length.

Some observations on checksums

- Checksums don't detect all bit errors
  - Consider (x, y) vs. (x – 1, y + 1) as adjacent 16-bit values in packet
  - Analogy: you can't assume the package hasn't been meddled with if its weight matches the one on the stamp. More smarts needed for that. ☺
  - But it's a lightweight method that works well in many cases

- Checksums are part of the packet; they can get corrupted too
  - The receiver will just declare an error if it finds an error
  - However, checksums don't enable the receiver to detect where the error lies or correct the error(s)
  - Checksum is an error detection mechanism; not a correction mechanism.

Some observations on checksums

- Checksums are insufficient for reliable data delivery
  - If a packet is lost, so is its checksum
- UDP and TCP use the same checksum function
  - TCP also uses the lightweight error detection capability
  - However, TCP has more mature mechanisms for generally reliable data delivery (lots more to come on this)

Playing with checksums

- A small demo

Summary of UDP

- UDP is a thin shim around network layer's best-effort delivery
  - One-off request/response messages
  - Lightweight transport for loss-tolerant delay-sensitive applications

- Provides basic multiplexing/demultiplexing for application
- No reliability, performance, or ordering guarantees
- Can do basic error detection (bit flips) using checksums
  - Error detection is necessary to deliver data reliably but it is insufficient