# CS 3640: Introduction to Networks and Their Applications

Fall 2023, Lecture 2: The end-to-end principle

Instructor: Rishab Nithyanand
Teaching Assistant: Manisha Keim

# You should have…

- Started working on Assignment 1 (due 8/31)

# Today's class

1. Recap

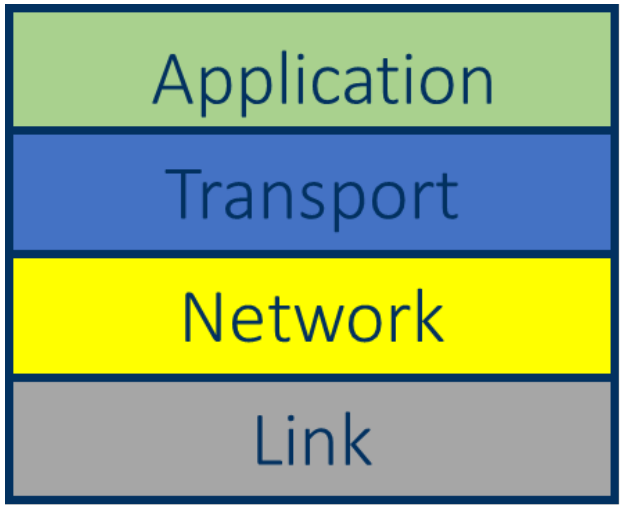
2. Building a picture of the Internet

3. The end-to-end principle

# Recap

- What is the Internet? How do the design goals of the Internet differ from a network?
- What are the four layers of the 4-layer model? What do they do?
- Why is reliability not built into the network layer?

# Recap: Layering




Provide protocols for applications to use (HTTP, email, etc.)

Provide (reliable) end-to-end delivery

Provide best-effort global delivery

Put bits on the medium

**Each layer only provides a service to the layer above it.**

# Today's class

1. Recap

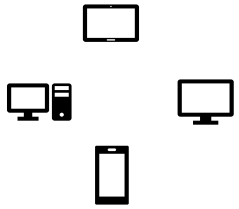
2. Building a picture of the Internet

3. The end-to-end principle

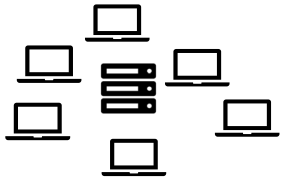
# Building a picture of the Internet: End-hosts

- End-hosts are the devices using the Internet.
  - They exist on the “edge” of the Internet. They “host” Internet apps and services. 23 Billion end-hosts today!

- Two types of end-hosts: Servers and clients.
  - In the context of an app, the “server” provides the service and the “client” uses it.



Home network



Enterprise network

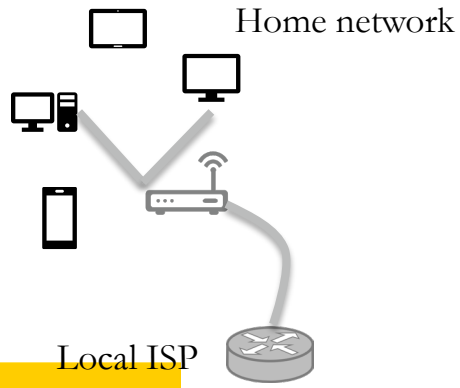
Mobile network

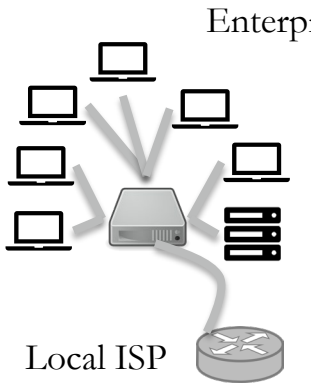
- **Discuss: What layers should be implemented on end-hosts?**

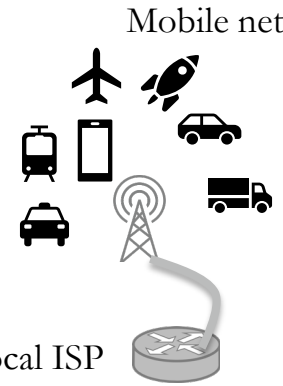
# Building a picture of the Internet: Access Networks

- Access networks connect end -hosts to the Internet infrastructure.
  - This is what your Internet Service Provider (ISP) does. They may provide cable, DSL, dial-up, or satellite access networks.
  - Your modem facilitates access to this network via coax cables, radio, optic fiber, or other wide-area network transmission media.
  - You can connect to your modem using ethernet, Wi-Fi, or other local-area network transmission media.

- **Discuss: What layers should be implemented on access network infrastructure?**

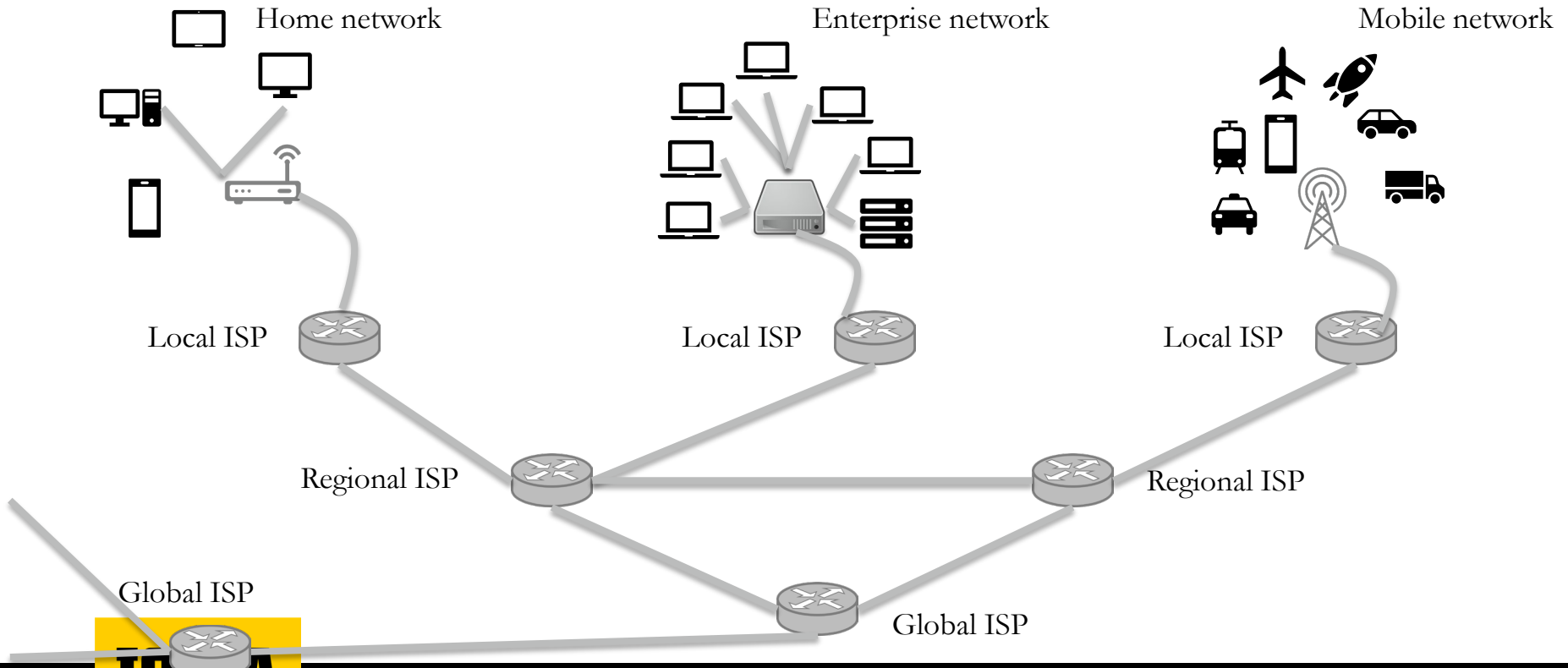


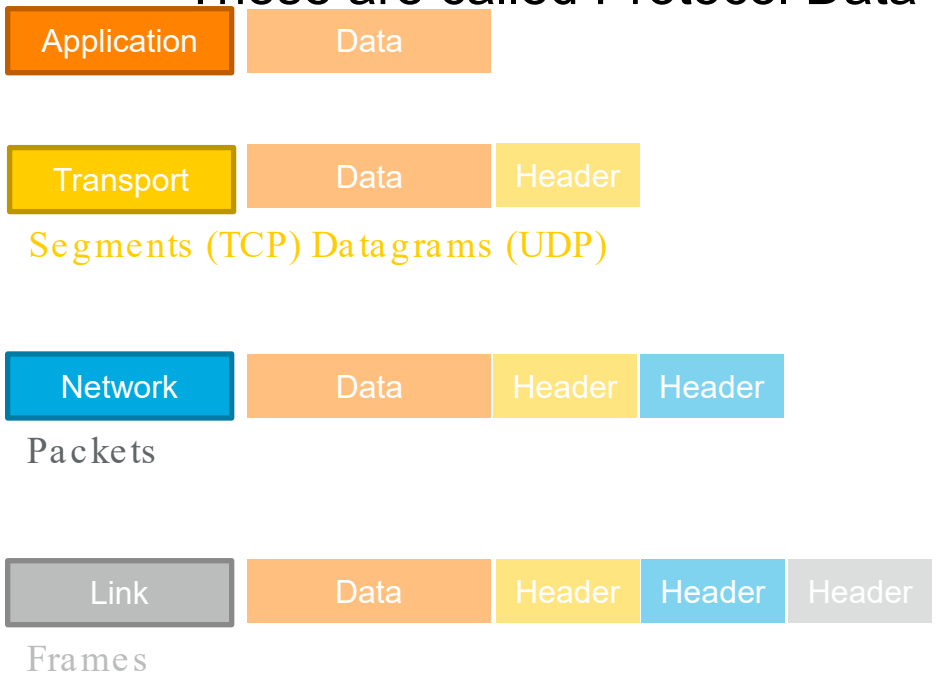
# Building a picture of the Internet: The network core

- This is the core infrastructure of the Internet
  - The core is where your local ISP connects to other “larger” ISPs. The larger ISPs do the same. Eventually, we end up with a large hierarchical network of networks – the Internet.
- **Discuss: What layers should be implemented on the network core?**



# Building a picture of the Internet: Encapsulation

- Data is broken into units at each layer of the Internet model
  - These are called Protocol Data Units (PDUs)




Which app (i.e., port) is the sender/receiver?
Information that enables check for corrupted data.
Multiple segments/datagrams (+headers) if over 64K
Information to help application layer correctly re-assemble data (if TCP).

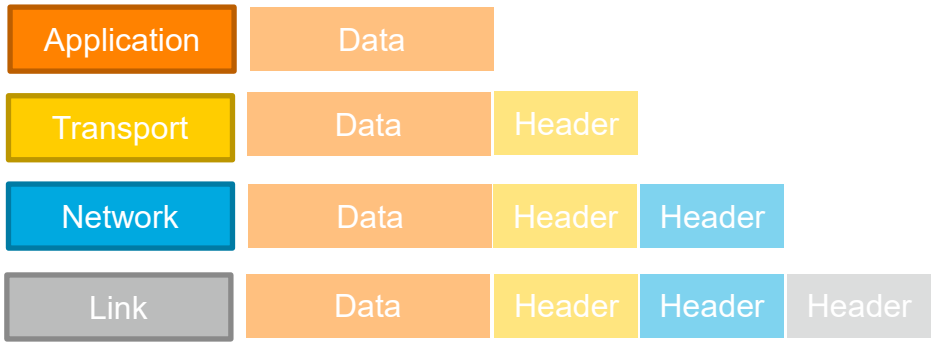
Which computer (i.e., IP address) is the sender/receiver?
How long before we destroy this packet (TTL)?
Information that enables check for corrupted data.
Multiple packets (+headers) if over 64K
Information to help transport layer correctly re-assemble data.

Which hardware interface (i.e., MAC address) is the sender/receiver?
Information that enables check for corrupted data.
Size limit decided by transmission medium.
Multiple frames (+headers) if over size limit.

Any guesses for why 64K is the size limit of a packet?

# Building a picture of the Internet: Packets

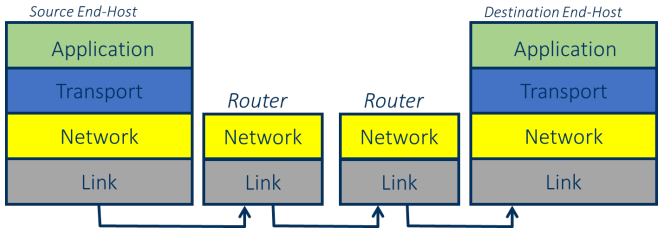
- Segments, datagrams, packets, and frames are units of data.
  - Segments and datagrams reside un-encapsulated only on the end hosts.
  - Frames only exist on a link.
  - What makes packets special?
    - They are seen un-encapsulated at every router between a source and destination because they exist at the network layer.




Segments (TCP)/Datagrams (UDP)

Packets

Frames



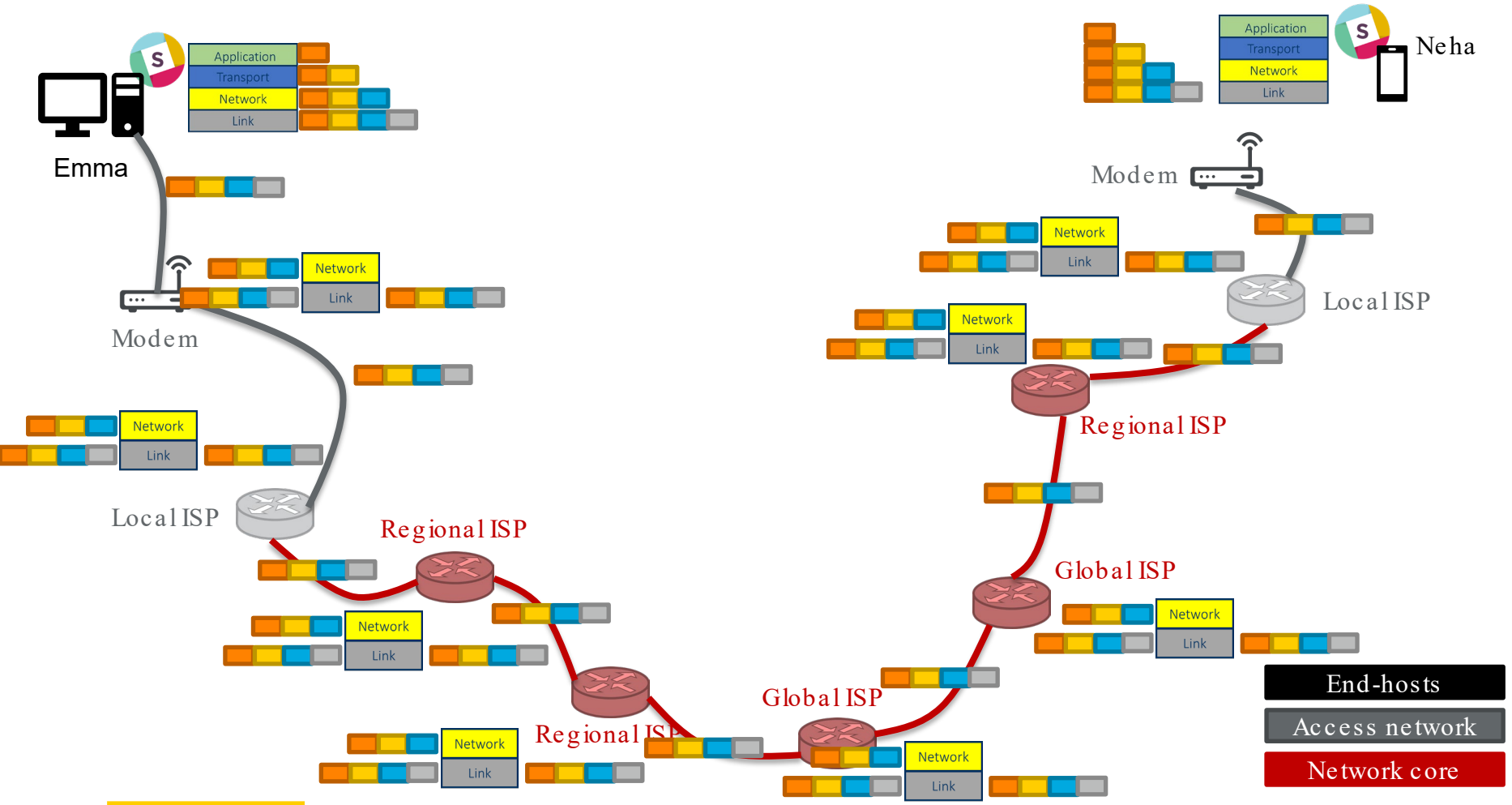

IOWA

# Building a picture of the Internet: Packets

- A packet is a unit of data on the network layer.
  - Think of each packet as a piece of mail.
  - It has control information such as: source IP address, destination IP address, and a “time to live” (TTL). This control information is stored in its 20 byte “header”.
    - Mail analogy: To/From addresses and postage stamps on the envelope.
  - The payload of a packet has the segment/datagram to be read by the recipient.
    - At intermediate routers, the header is examined to identify the next router the link layer should send the packet to.
      - Mail analogy: Intermediate postal centers forward the envelope to the next postal center.
    - When a packet reaches its destination, the payload is delivered to the transport layer above it.
      - Mail analogy: Mail is processed by its receiver after delivery.

In 72ms a packet goes from NYC to Sydney. In its lifetime it travels under the oceans, through war-torn lands, dictatorships, and democracies.

# The exciting life of a packet!




IOWA

# Today's class

1. Recap

2. Building a picture of the Internet

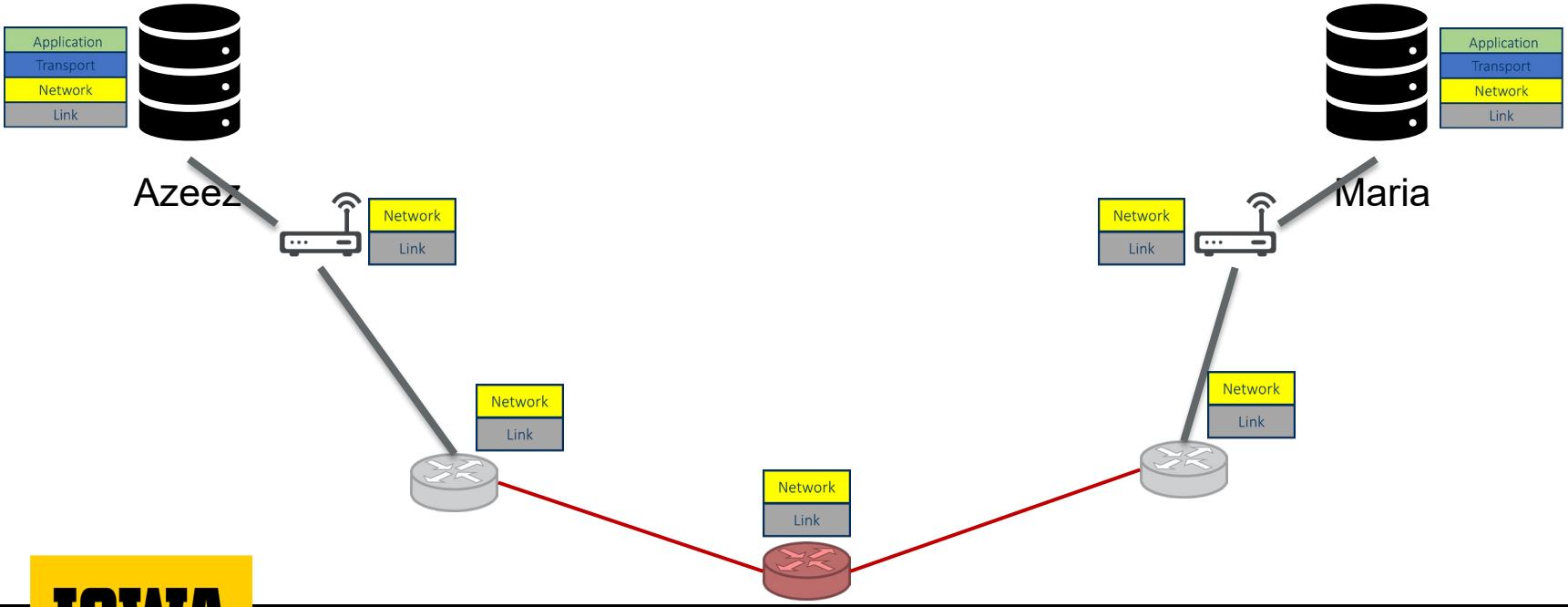
3. The end-to-end principle

# The three design principles of the Internet

- Layering
  - What modules does the Internet need?
- End-to-end
- Fate-sharing

# Let's design a file transfer app

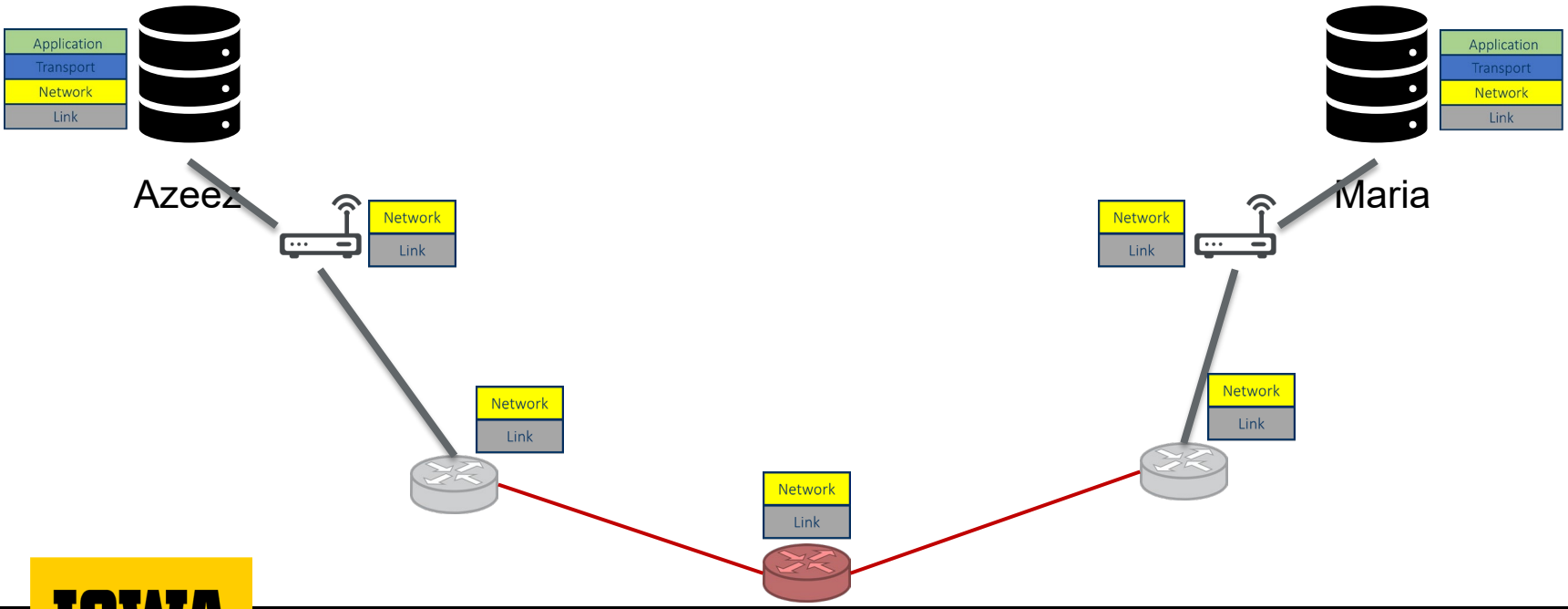
- Scenario: Azeez has to transfer large files to Maria. They need reliability guarantees. How should they implement it?
  - Errors may come from hosts or the network (packets can get corrupted or lost at each node).
  - At the network layer (per hop) or transport layer (per file)? Why?



# Let's design a file transfer app

- Solution 1: Make each hop reliable.

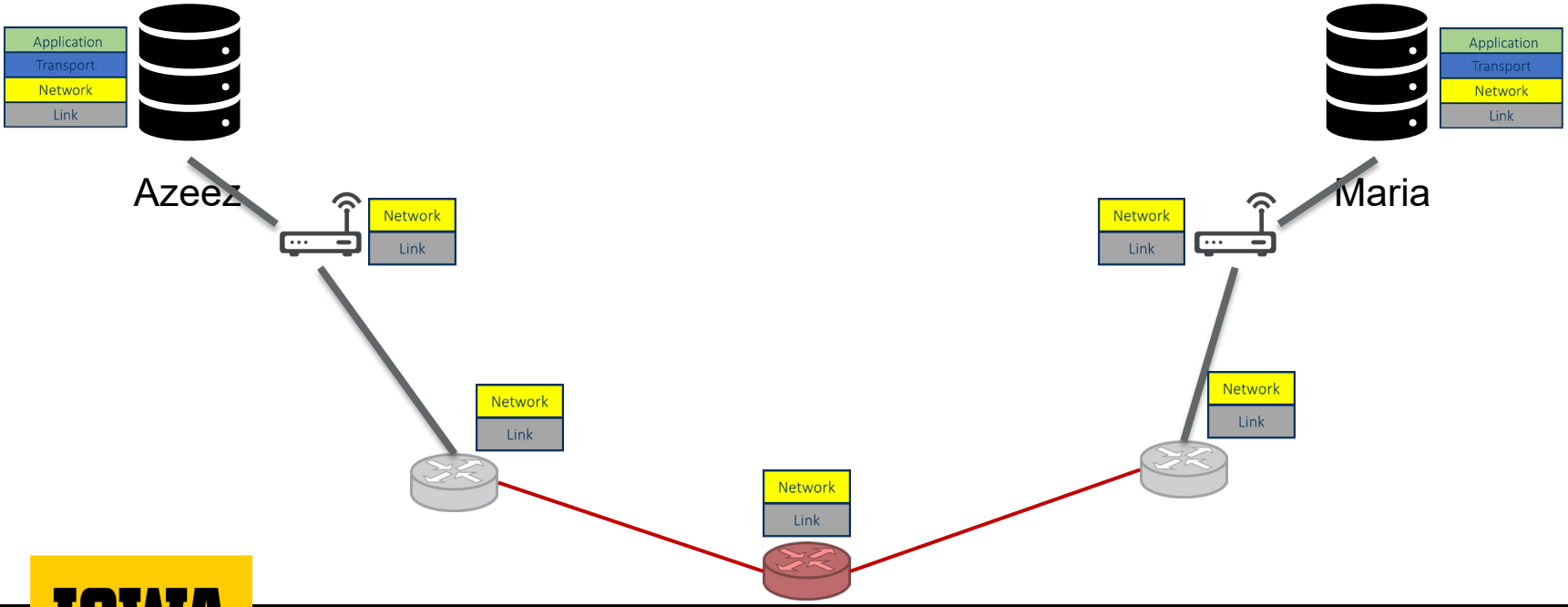
**Discuss: Good idea?**



# Let's design a file transfer app

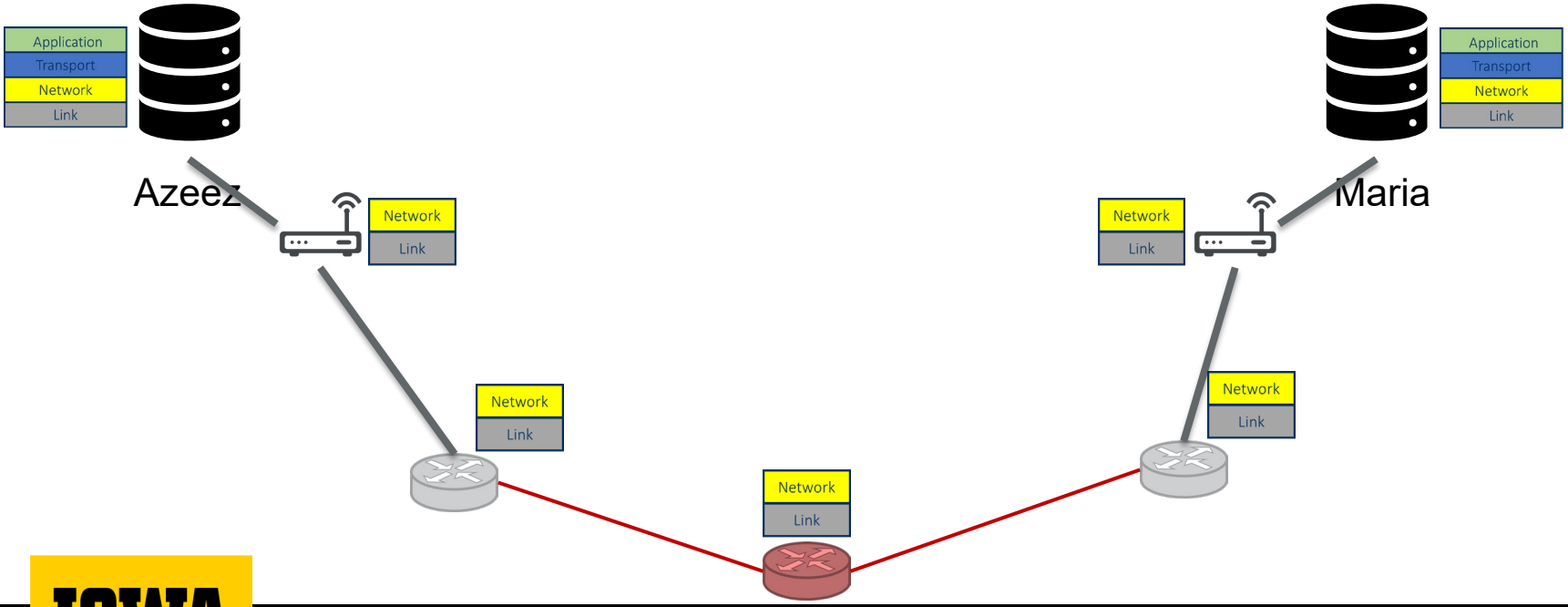
- Solution 2: Let hops be unreliable. Do a final check at the end-host and try the transfer again if required.

**Discuss: Good idea?**



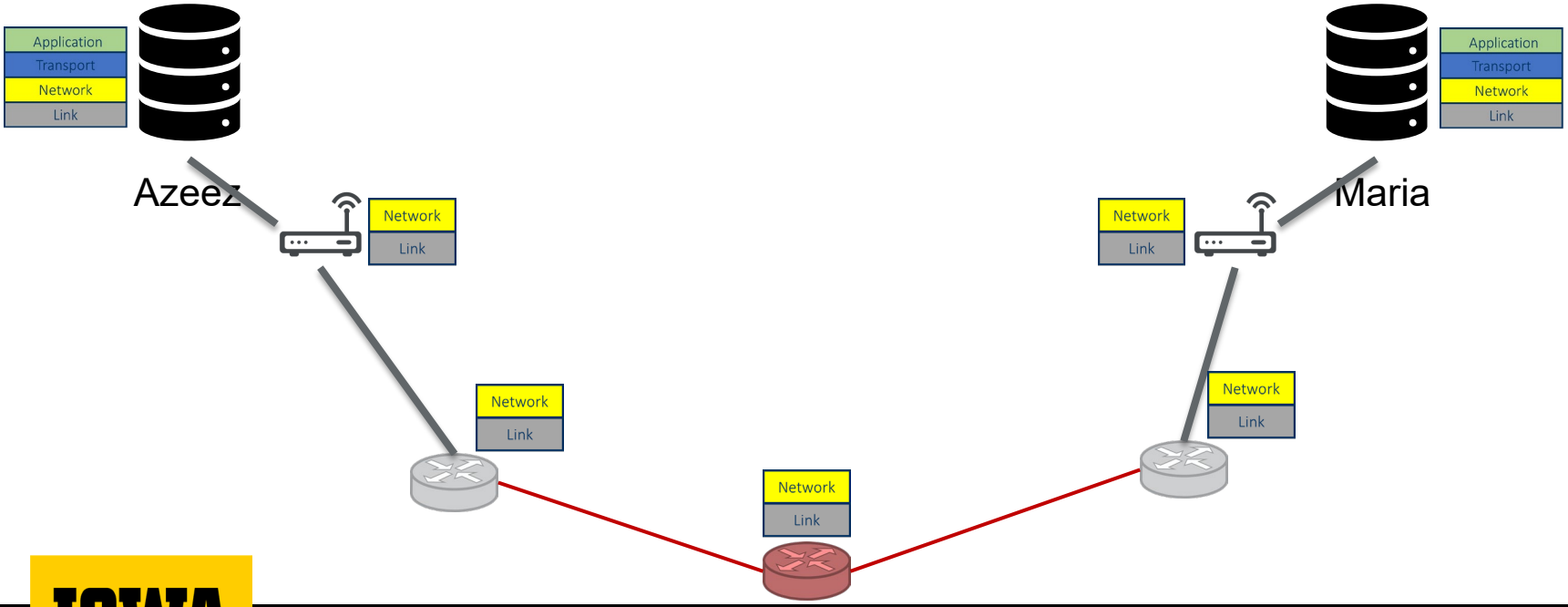
# Let’s design a file transfer app

- Solution 1: Make each hop reliable.
  - You cannot make anything on the network perfectly reliable. Requires that Maria trusts the functioning of network elements not in her control.
  - What happens if a router misbehaves? Maria will still always have to check files at her end. We introduced a redundant functionality to the network layer.



# Let's design a file transfer app

- Solution 2: Let hops be unreliable. Do a final check at the end-host and try the transfer again if required.
  - Can fail only if Maria (the end-host) herself fails. She only has to trust her own functionality.



# Layering vs. end-to-end principle

- Layering tells us how to compose functionality. But it never tells us what functionality should be implemented at which layer.
  - Example: Use HTTP over TCP over IP over Ethernet to download web pages. But where should encryption be implemented?

- The end-to-end principle gives us guidance on where functionality should be implemented.

# The end-to-end principle

- Functionality should be implemented at the lower layer if and only if:
  - It can be implemented completely
  - It can be implemented correctly
  - It is not made redundant by a higher layer
- We can make exceptions for performance optimizations.
  - As long as it does not add a burden to applications that do not need the functionality.
- This is a principle, not a rule.
  - Violations exist on the Internet today.
  - Examples: Firewalls, proxies, and NATs. (We'll get to them later in the term).

# A simplified version of the end-to-end principle

If you can implement a functionality on the end-hosts without requiring the network's help, do it.

Why?

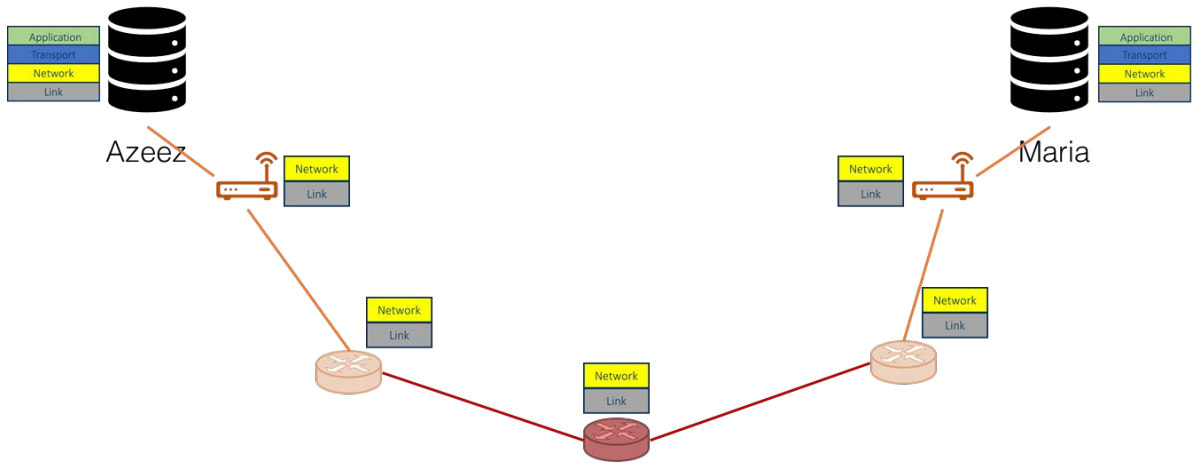
Because you can never rely on the network.

# Why does the end-to-end principle make sense?

- In general, implementing functionality in the network:
  - Does not reduce end-host/application complexity.
  - Increases network complexity.
  - Imposes additional costs on all applications (even those that do not require the functionality).

# When does violating the e2e principle make sense?

- Scenario: Highly unreliable networks.
  - There are six links between Azeez and Maria. Each has a fail rate of 10%
  - Probability of a packet not reaching Maria: 47%
  - Should we retransmit the entire file each time the network drops a packet?



# Impact of the e2e principle

- The end-to-end principle is why reliability is not implemented in the network.
  - It is implemented at the end-host (in the transport layer via TCP).
- Reduces the functions of the stressed network core. Allows it focus on doing a few jobs well.
  - (Funnily enough) This results in better network reliability.
- More flexibility for applications.
  - They get to pick and choose what kind of functionality they want.
  - They don’t pay a price for things they don’t need.

# Discussion

- Who benefits from the e2e principle? Which entities have an interest in violating it?

# The three design principles of the Internet

- Layering
  - What modules does the Internet need?
- End-to-end
  - What functionality should go in which modules?
- Fate-sharing

# Fate sharing

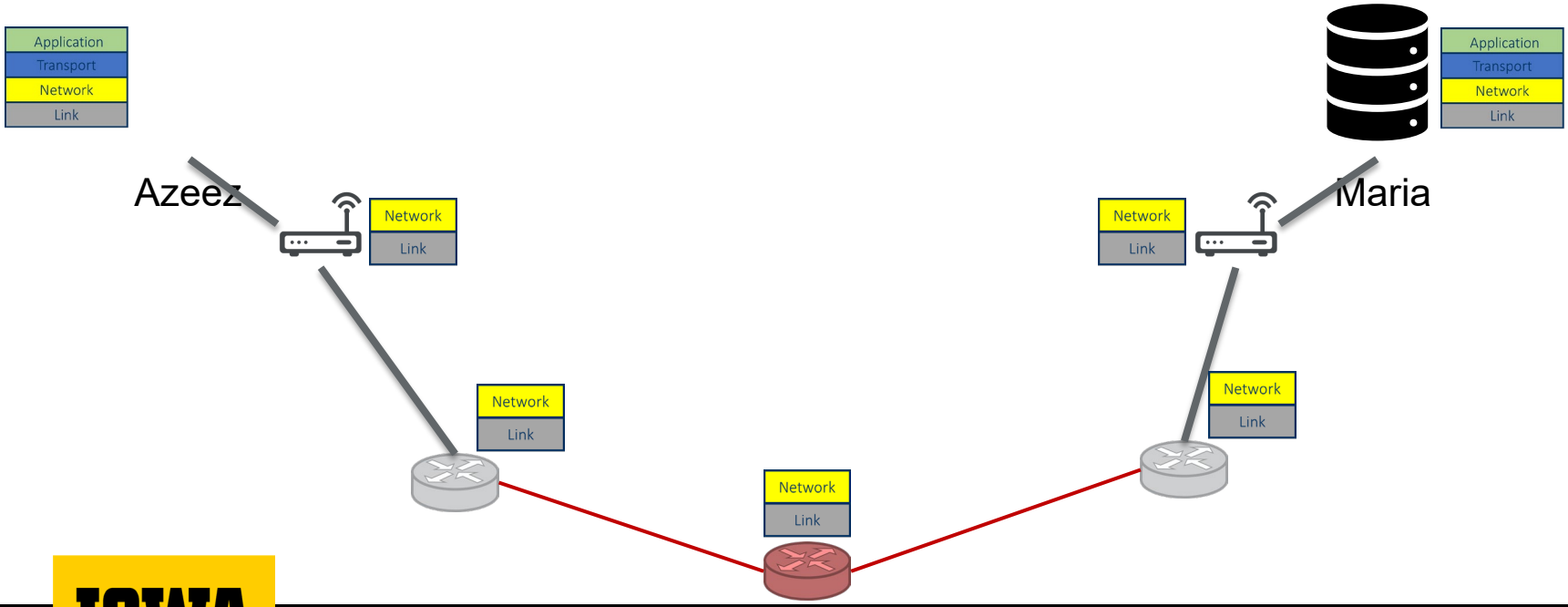
> **"A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable." – Leslie Lamport**

- Store network state only in the entities that rely on this state.
  - Critical state is lost only when the entities that depend on it fail.

We'll revisit this principle a lot when we explore how packets are routed on the Internet (the network layer).

# Fate sharing

- Solution 2: Allow hops to be unreliable. Do a check on completion and try the transfer again if required.
- Why was this the better solution?
  - The success of the transfer relied only on Maria and Azeez – the two parties who benefitted from its completion.



# The three design principles of the Internet

- Layering
  - What modules does the Internet need?
- End-to-end
  - What functionality should go in which modules?
- Fate-sharing
  - Who should store network state?

# Highly recommended reading

- David Clark. "The Design Philosophy of the DARPA Internet Protocols".

# What you should remember

- What are the building blocks of the Internet?
  - Which Internet layers do each of them use?
- What are the units of data at each layer?
  - What role do headers play at each level?
- What is the end-to-end principle?
  - What does it do that layering does not?
  - Why does it apply to most conditions?
  - When might we consider violating the principle?
  - How has it shaped the Internet?
- What is the fate-sharing principle?