End-to-End Arguments in System Design

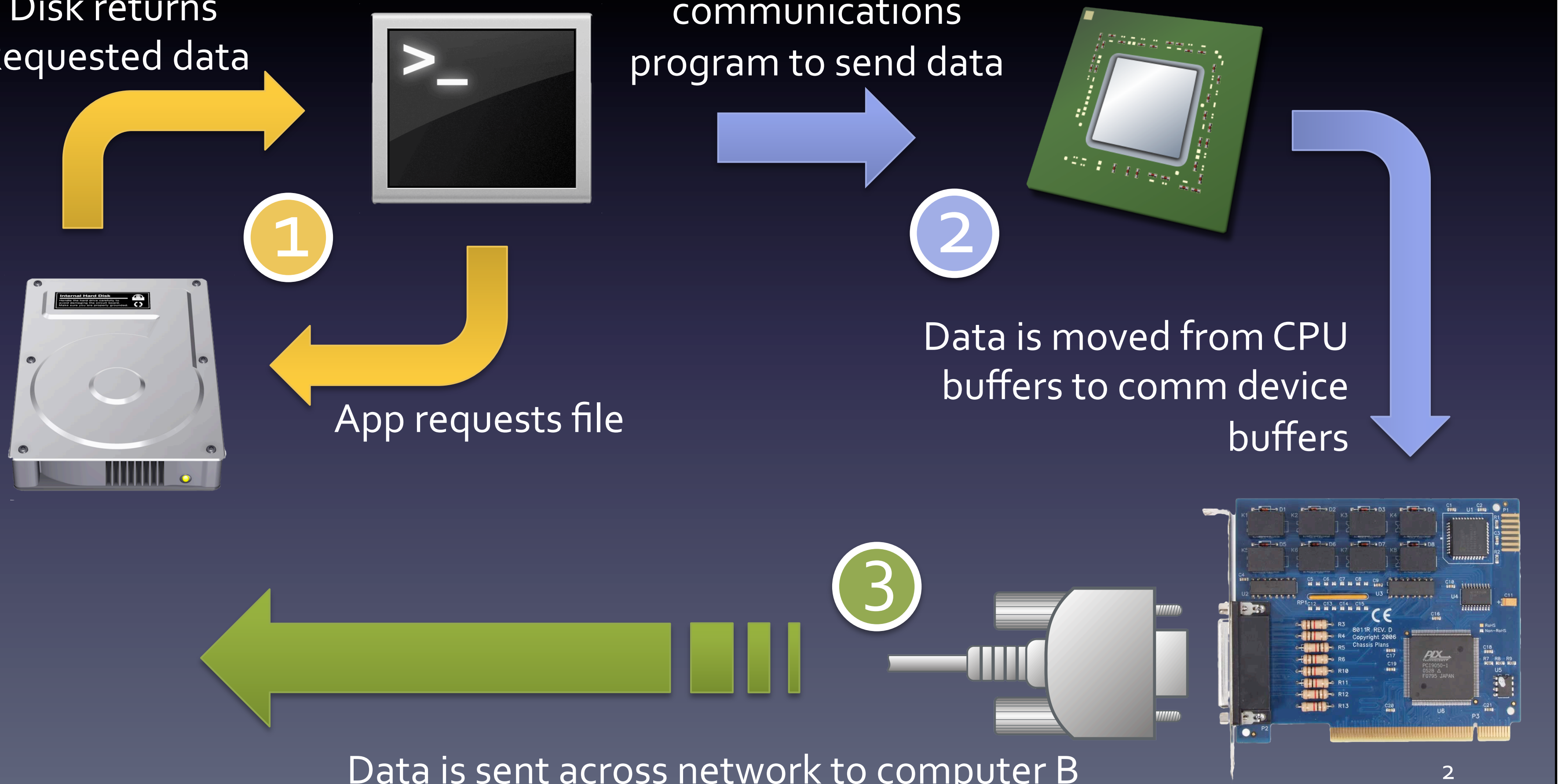
J. H. Saltzer, D. P. Reed, and D. D. Clark

ACM Transactions on Computer Systems, Vol. 2, No. 4, Nov. 1984

E2EA

- A set of best practices when designing a system
- At any given level in a system, only implement functionality which can be effectively utilized by all higher levels in the system

Careful File Transfer (Computer A)



Disk returns Requested data

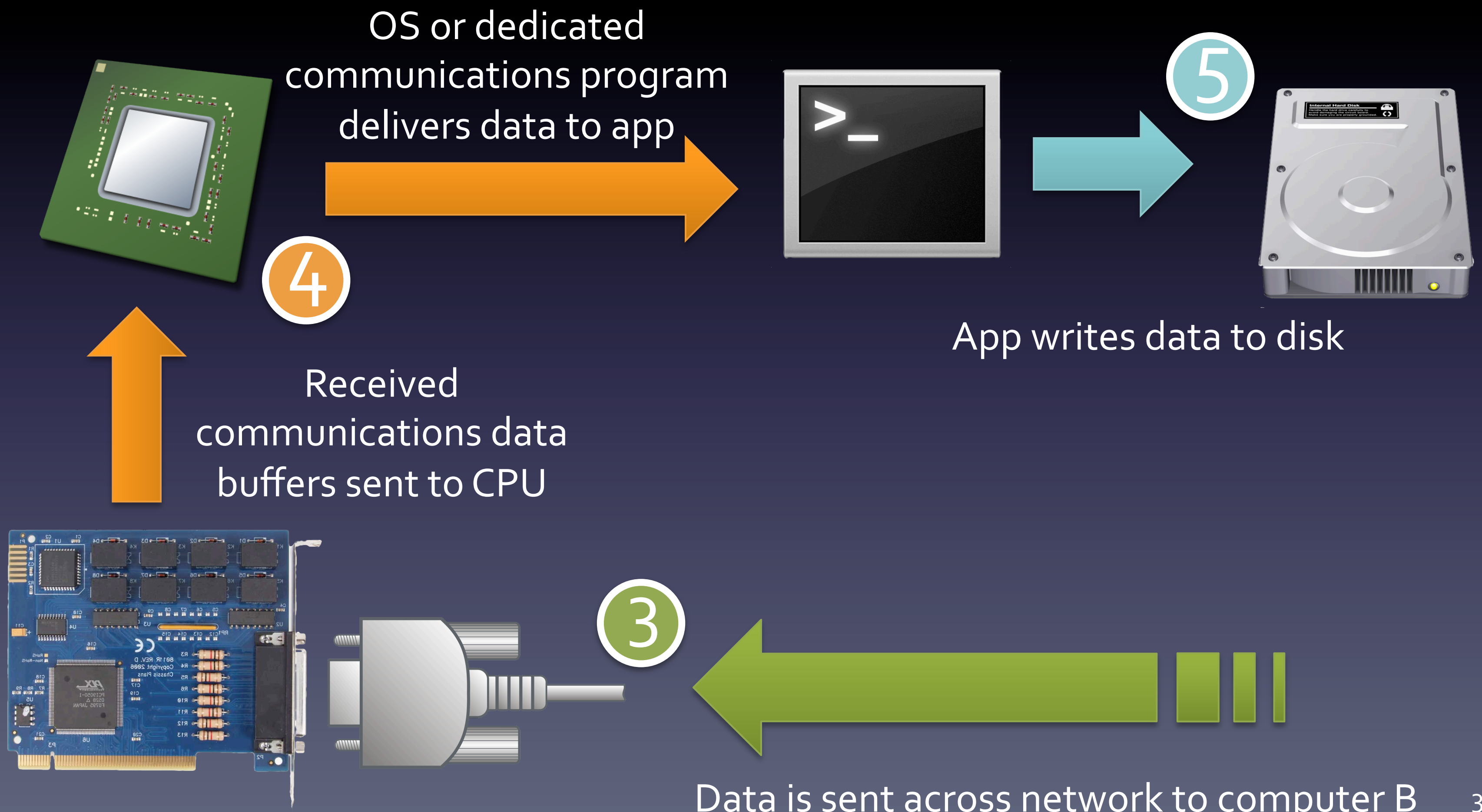
App requests file

App asks OS or communications program to send data

Data is moved from CPU buffers to comm device buffers

Data is sent across network to computer B

Careful File Transfer (Computer B)



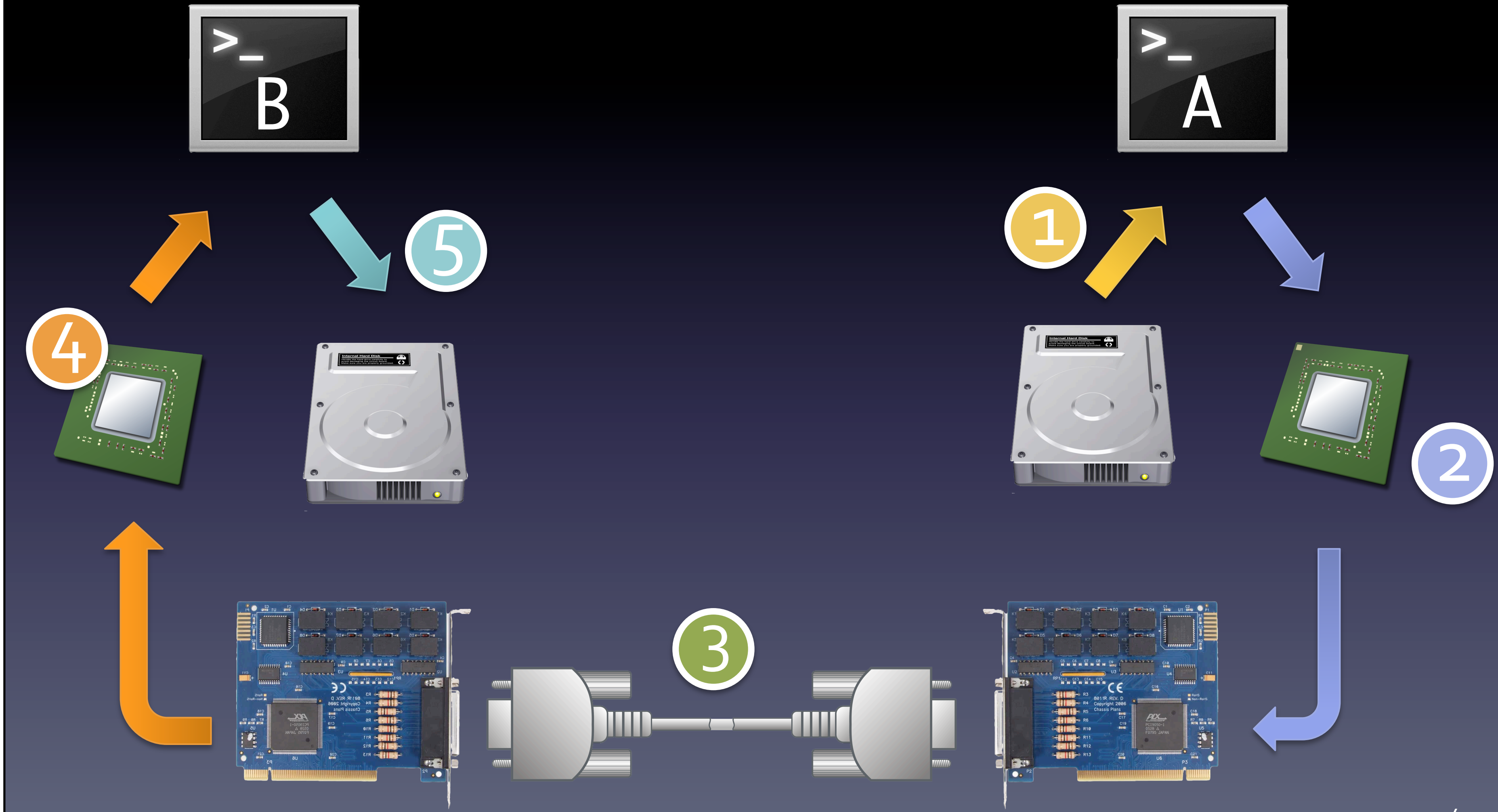
OS or dedicated communications program delivers data to app

App writes data to disk

Received communications data buffers sent to CPU

Data is sent across network to computer B

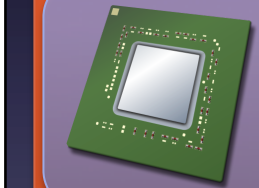
Where can things go wrong?



Where can things go wrong?

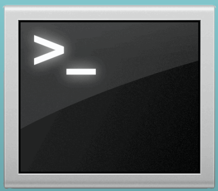
An entire system could crash during the transfer

Hardware faults cause data to be read or written incorrectly

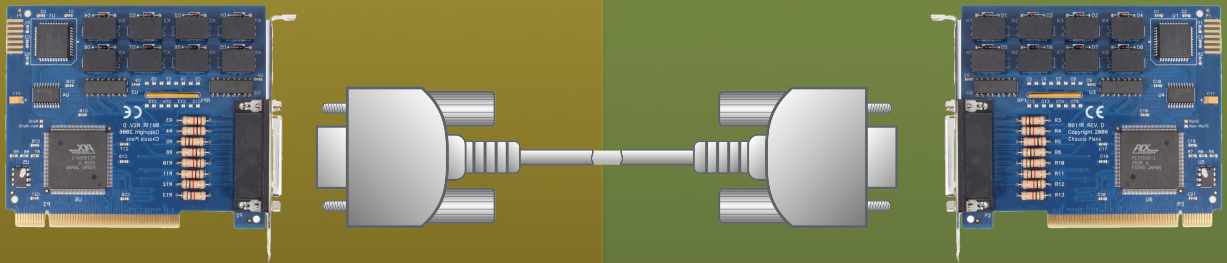


Transient errors in the CPU or RAM subsystems could cause buffers to be corrupted

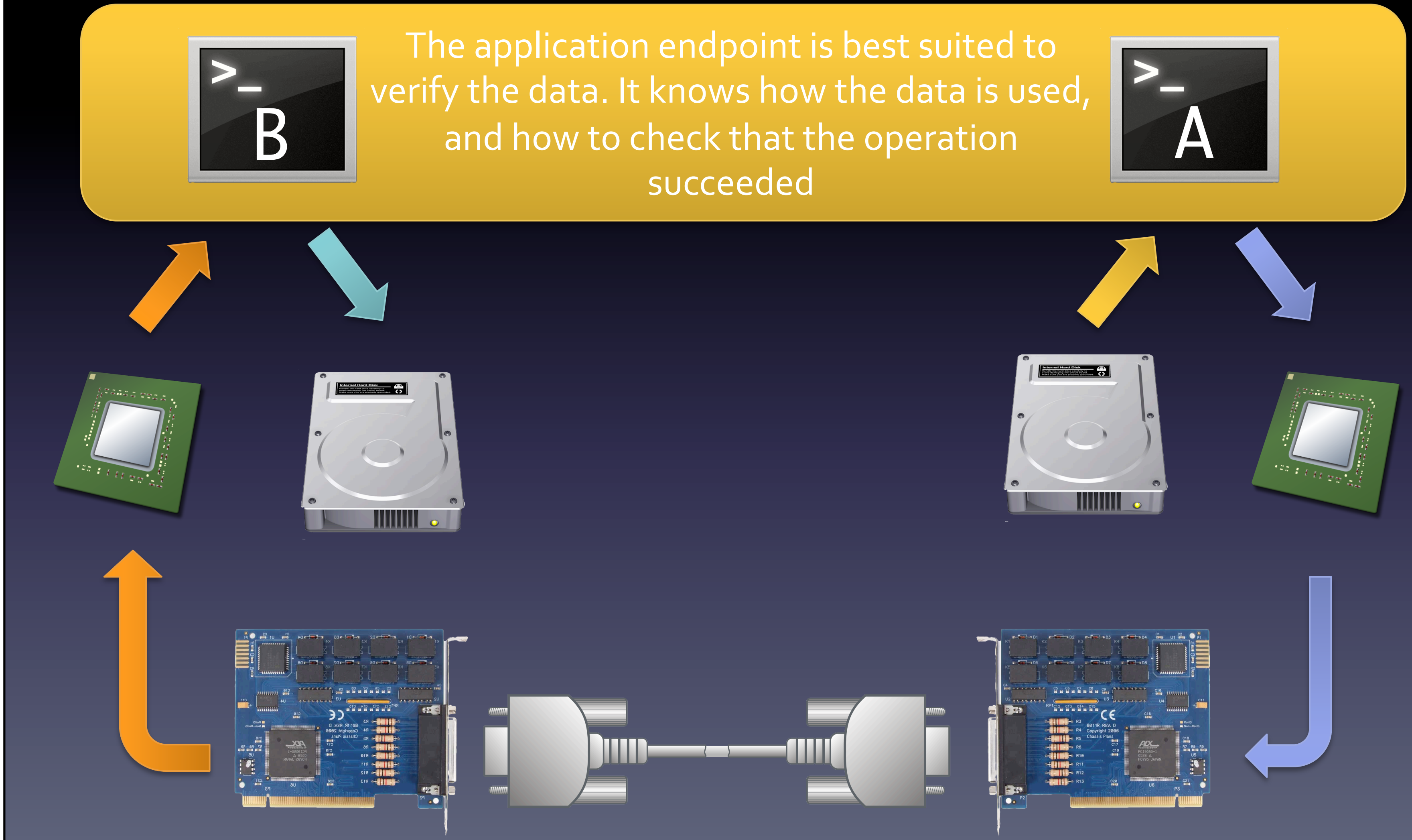
Incorrect logic or other flaws in the OS or file transfer software can corrupt data



The network subsystem could drop packets or flip bits

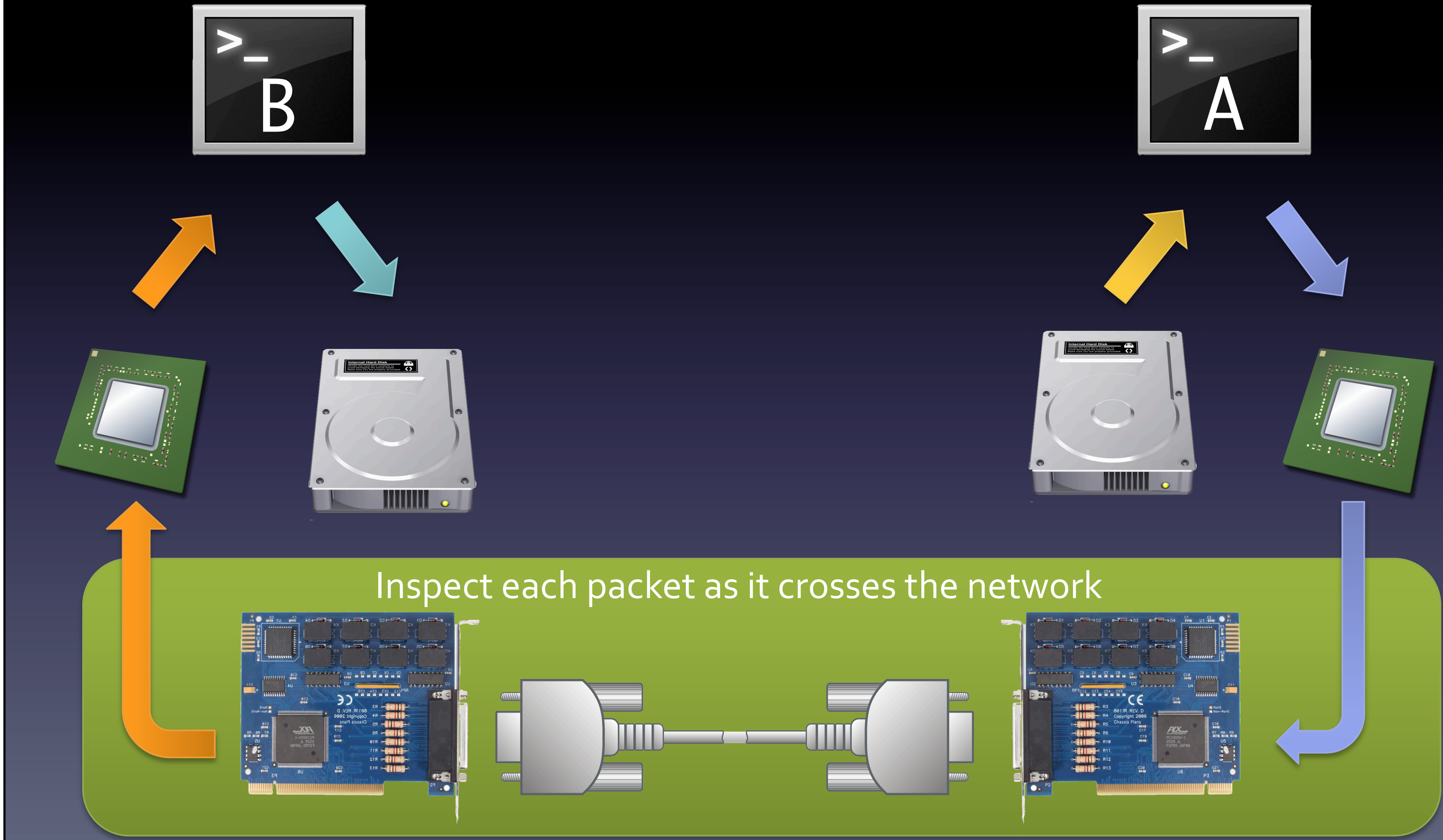


Check at the endpoints



The application endpoint is best suited to verify the data. It knows how the data is used, and how to check that the operation succeeded

Verify data in low level systems



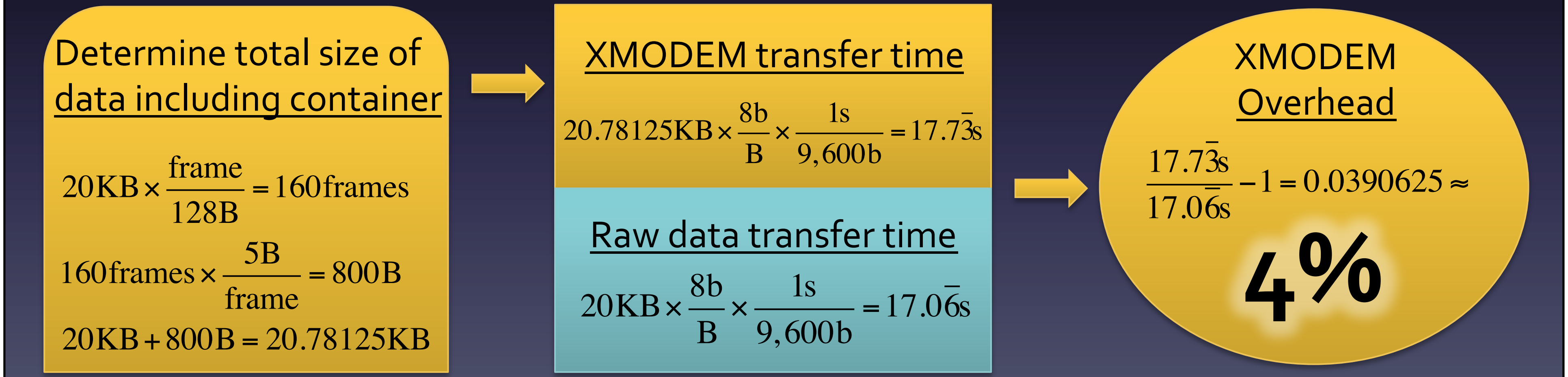
Inspect each packet as it crosses the network

Verifying each packet

- Encapsulate a 20KB file transfer using the XMODEM protocol and transfer at 9,600 bps

XMODEM Packet Structure:

SOH	Frame #	$\overline{\text{Frame \#}}$	Byte 1	Byte 2	• • •	Byte 128	CRC	CRC



Determine total size of data including container

$$20\text{KB} \times \frac{\text{frame}}{128\text{B}} = 160\,\text{frames}$$

$$160\,\text{frames} \times \frac{5\text{B}}{\text{frame}} = 800\text{B}$$

$$20\text{KB} + 800\text{B} = 20.78125\text{KB}$$

XMODEM transfer time

$$20.78125\text{KB} \times \frac{8\text{b}}{\text{B}} \times \frac{1\text{s}}{9{,}600\text{b}} = 17.7\bar{3}\text{s}$$

Raw data transfer time

$$20\text{KB} \times \frac{8\text{b}}{\text{B}} \times \frac{1\text{s}}{9{,}600\text{b}} = 17.0\bar{6}\text{s}$$

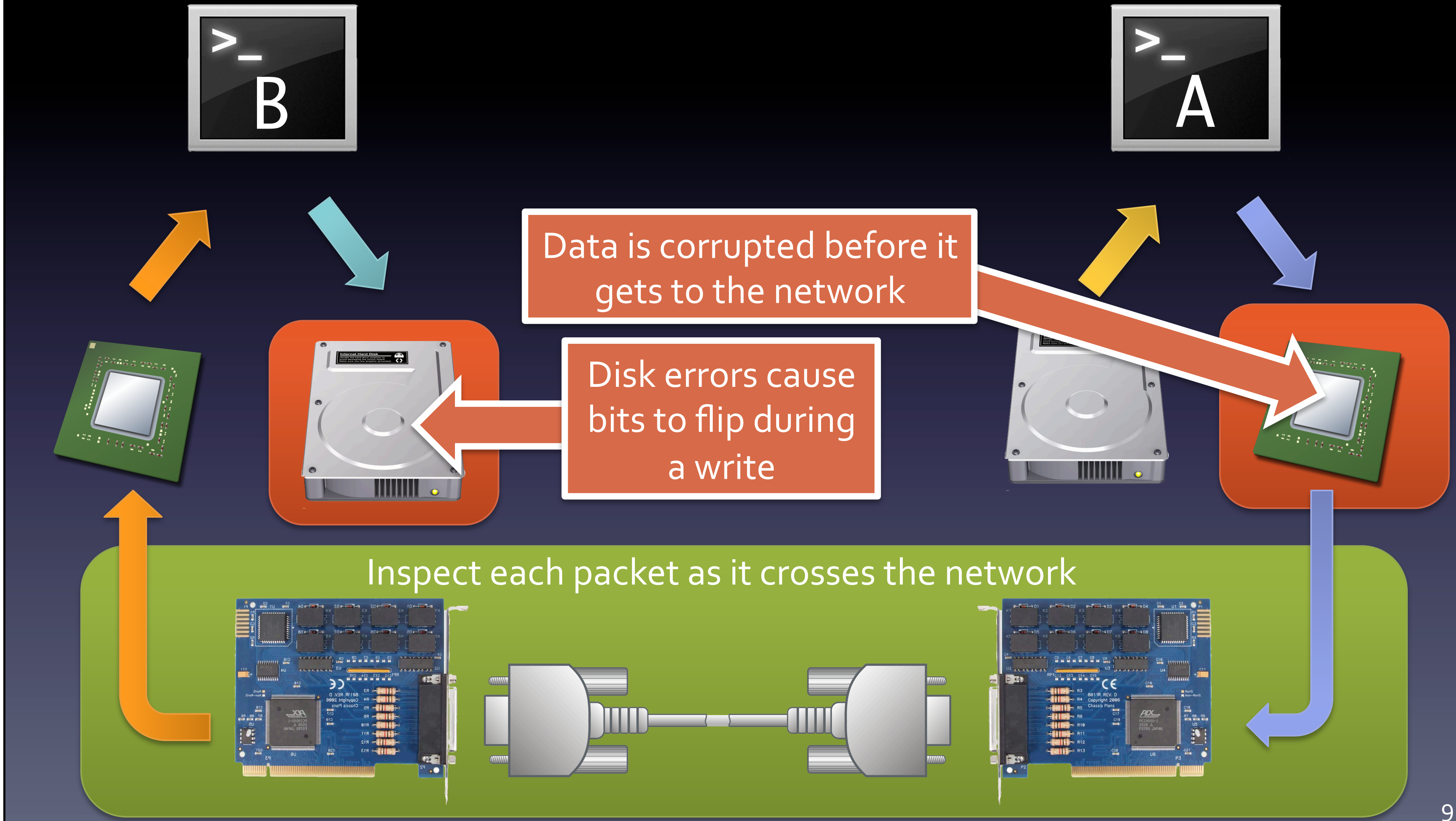
XMODEM Overhead

$$\frac{17.7\bar{3}\text{s}}{17.0\bar{6}\text{s}} - 1 = 0.0390625 \approx$$

4%

The overhead imposed by checking each packet *seems* modest…

Other errors can defeat packet inspections

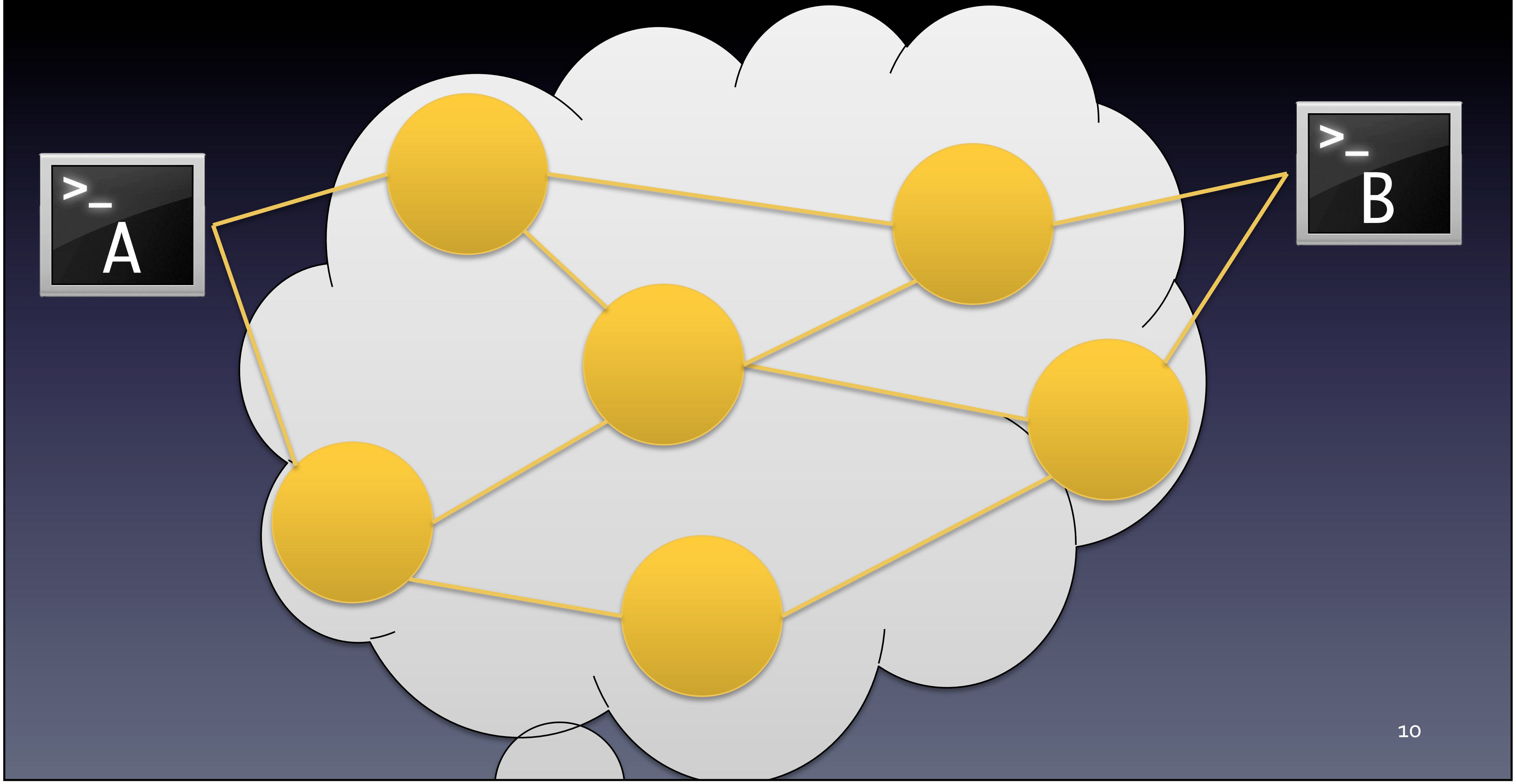


Data is corrupted before it gets to the network

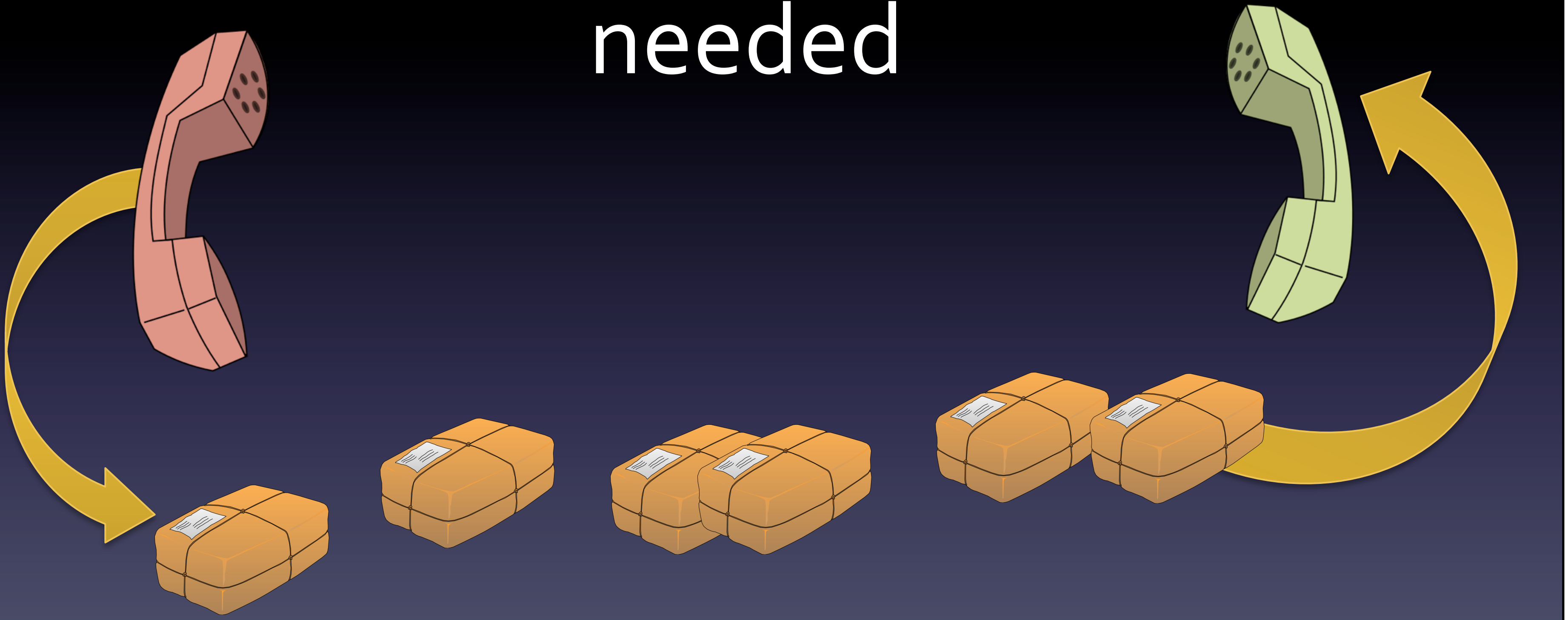
Disk errors cause bits to flip during a write

Inspect each packet as it crosses the network

Checking packets at each hop



Let the application decide what error checking and recovery is needed

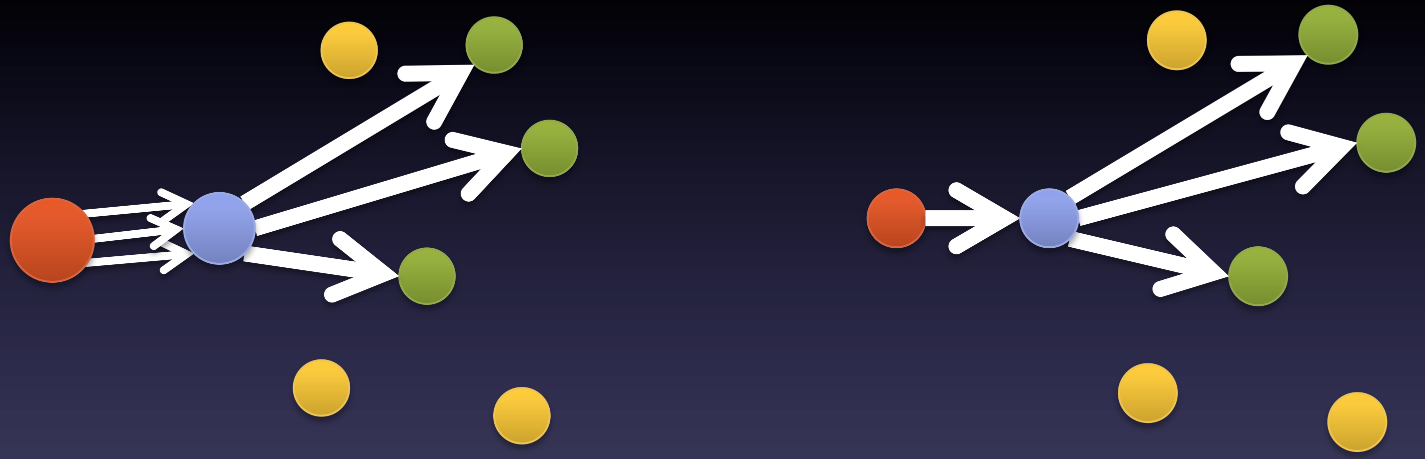


Error checking and recovery is not preferred in *every* situation

What do you think?

Multicast

- Unicast requires a dedicated message for each client
- Can be bandwidth intensive, since identical content may be sent across the network to different users



- Multicast allows users to register to receive messages from a particular source
- Allows the sender to send one message, which is duplicated at a node with multiple interested clients attached

This task cannot be performed in the ends, but clients are not forced to use multicast

Source: T Moore. **A critical review of "End-to-end arguments in system design".** *In IEEE ICC 2002*. vol 2. pp 1214-1219. August 2002. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=997043

Sorting Libraries

```
void qsort ( void * base,
             size_t num,
             size_t size,
             int ( * comparator ) ( const void *, const void * ) );
```

- The endpoint is an application which requires sorted data sets
- The sort function doesn’t have enough information about how to compare items
  - It’s a partial implementation
- The calling application provides the rest of the implementation using a comparator

The sorting library does not restrict callers to use a particular value system, yet provides a library implementation of quicksort

Questions?

Thank You

BACKUP

What are the ‘ends’

- “The ‘ends’… are the things that the property that you’re actually trying to argue about defines as the ends [of the system].”
- “That’s only slightly circular, it basically says that the E2EA is a style of argument so you bind the E2EA to the terms in a property. So a security property typically names some ‘things’ that are the subject and object and some invariant you want, and the ends are the things that you name… That’s the mathematical idea.”
- “In practice, it’s really a step outside this to map the ends. Sometimes it’s a user if you a security statement, for example, you’re basically saying that the user is represented by a principle and the principle has the following capabilities: to act, to carry out the set of actions and they communicate thru a communications system so the ends in that case are the things about which you’re reasoning.”

Source: D Reed. **End-to-End Arguments: The Internet and Beyond.** *From the 19th USENIX Security Symposium*. August 13, 2010. Washington, DC. http://www.usenix.org/multimedia/sec10reed