Middleboxes and NFV

15-441 Spring 2019

Profs Peter Steenkiste & **Justine Sherry**

Thanks to Scott Shenker, Sylvia Ratnasamay, Peter Steenkiste, and Srini Seshan for slides.

Carnegie Mellon University

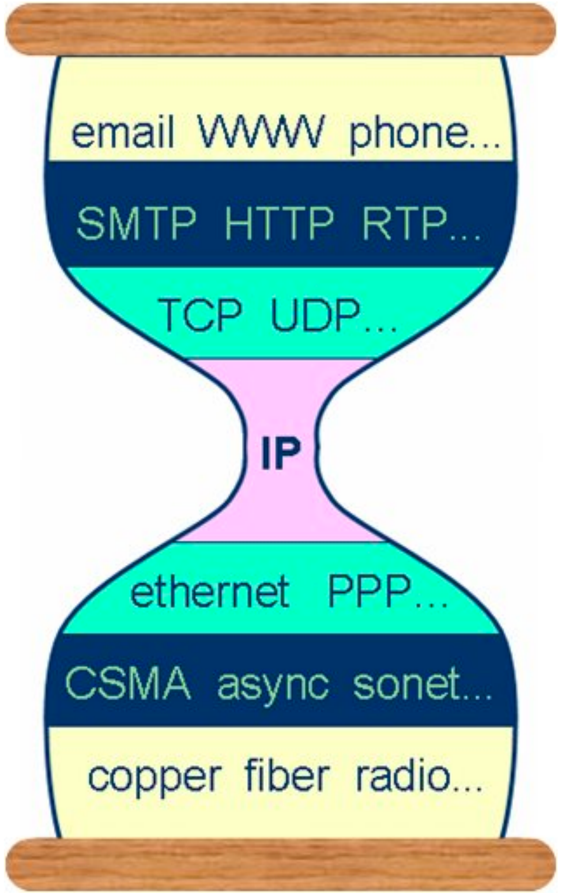
I’ve missed you!

Please TA for Me!

How was Tyler? What did you learn? Did anything surprise you?

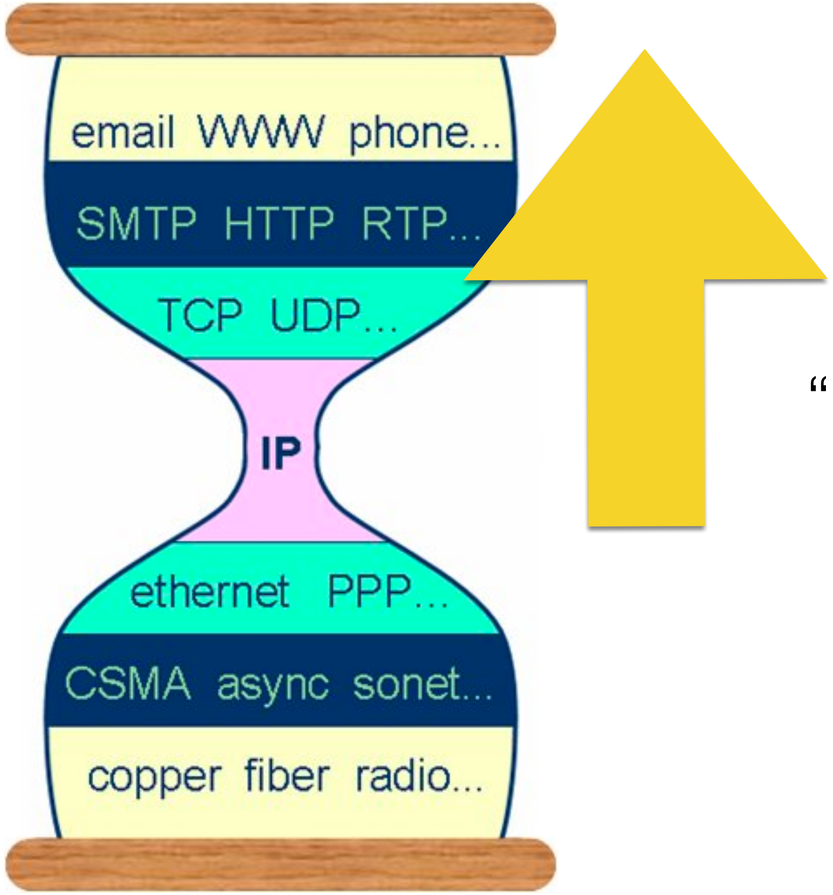
Last Week I went to a Castle in Germany

You are here.



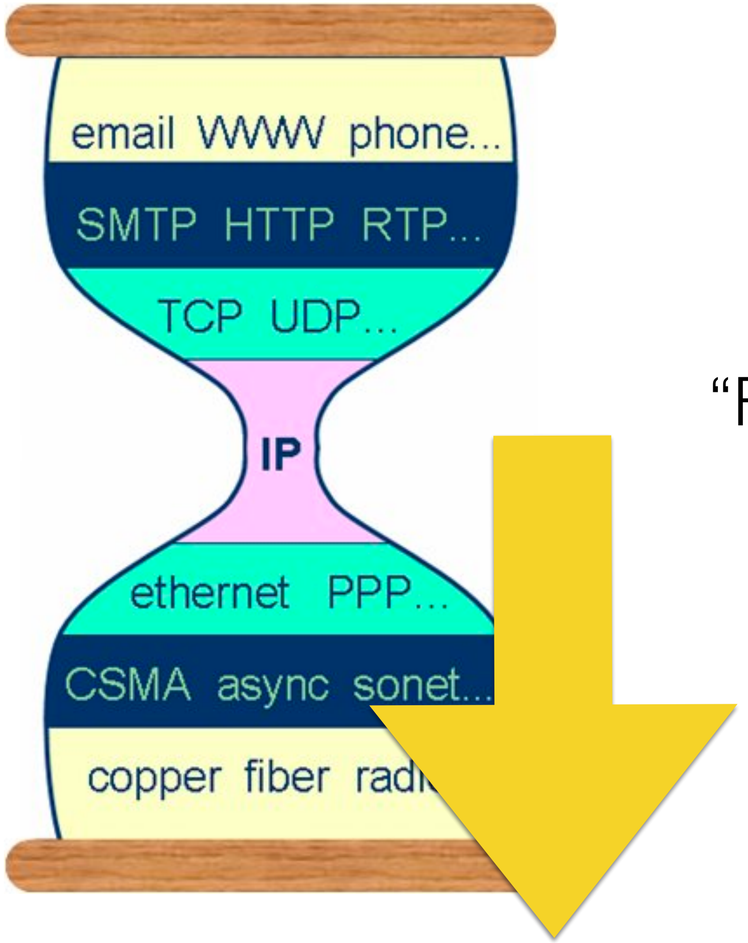
Application Layer

You are here.



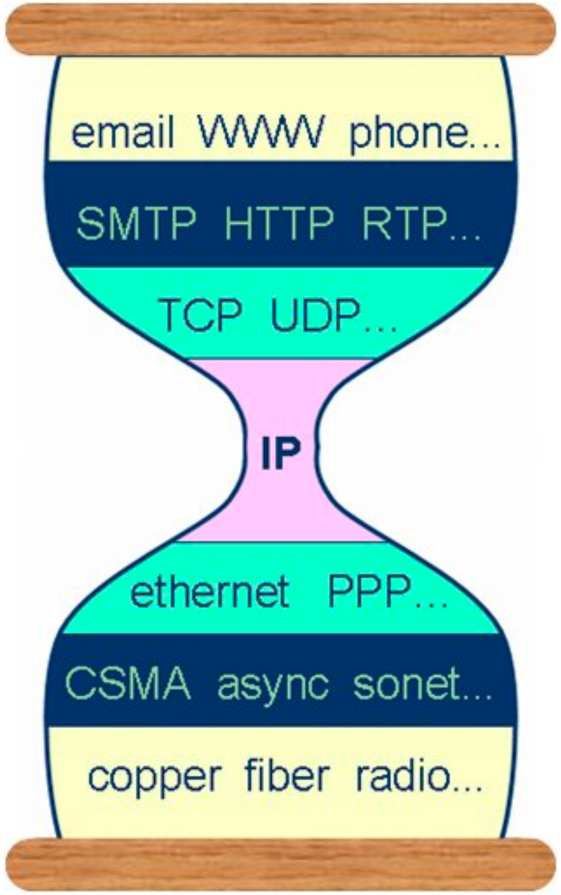
"From packets up to applications"

Next week++



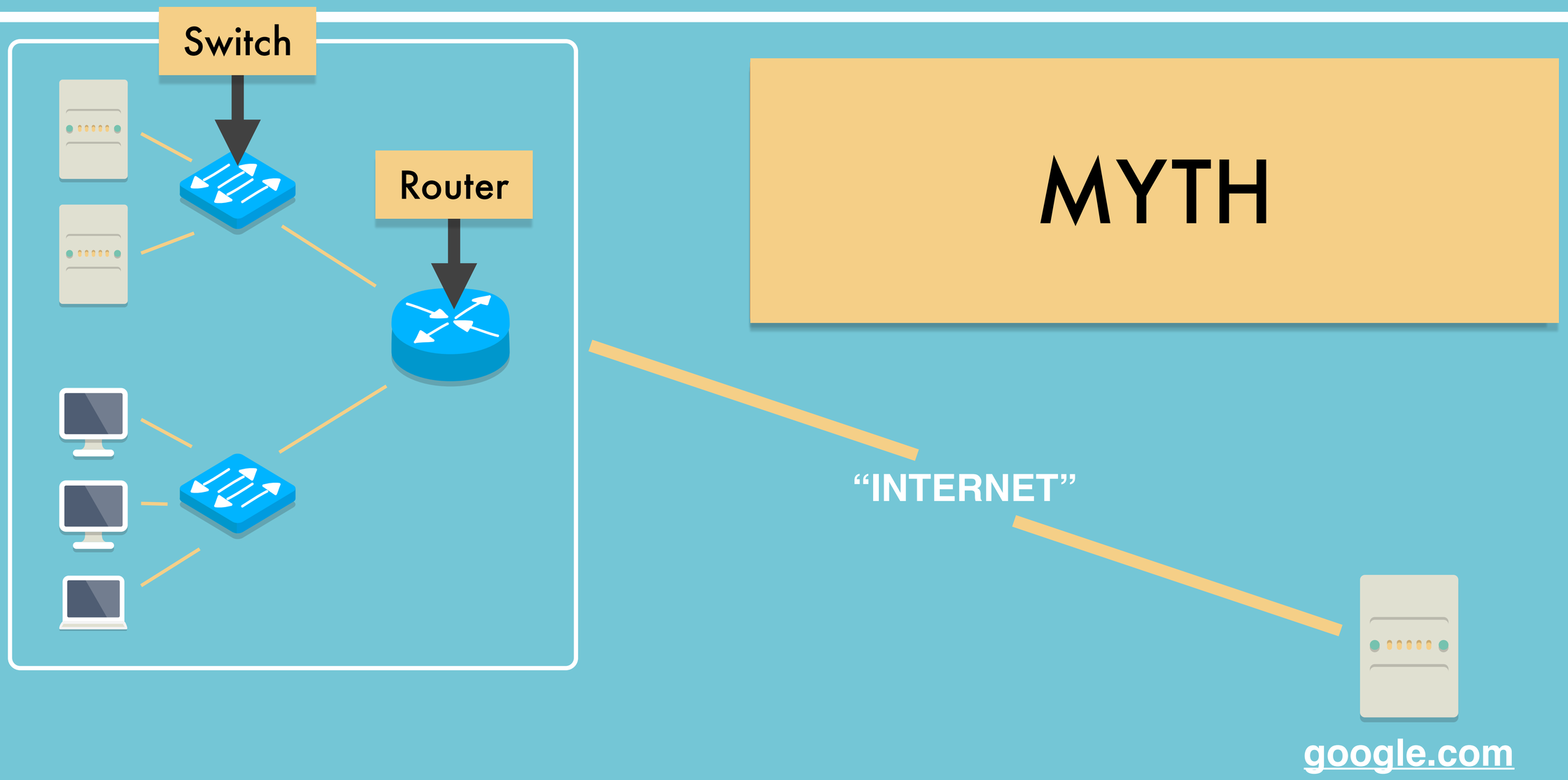
"From packets down to bits and signals"

This week…



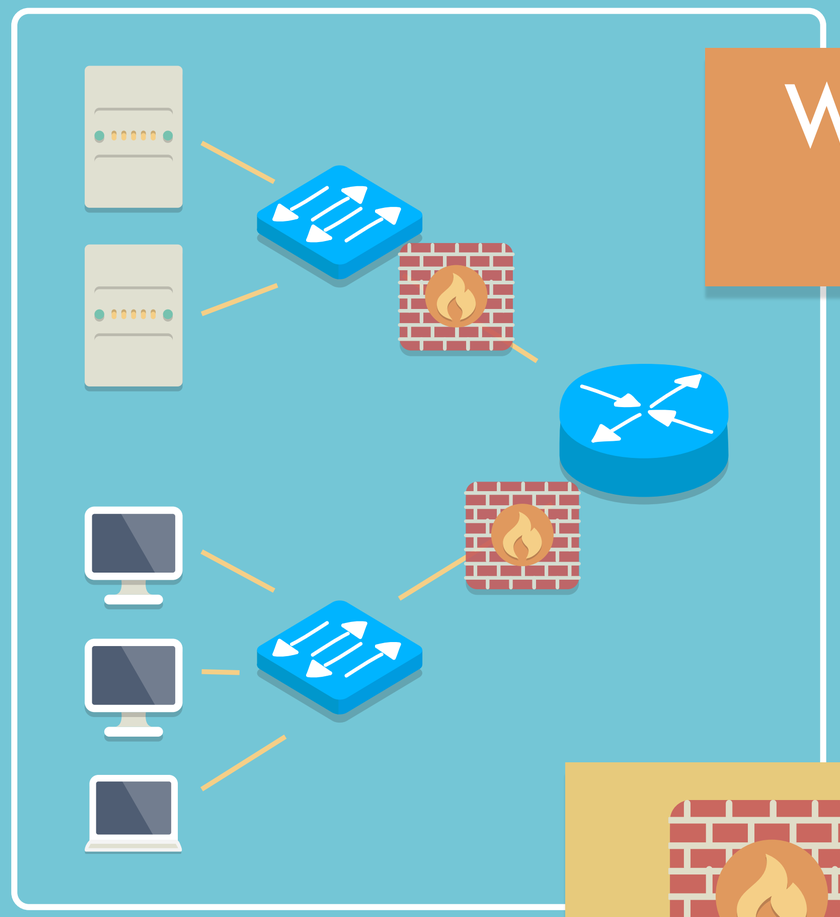
Breaking the model a little bit…

Enterprise Networks



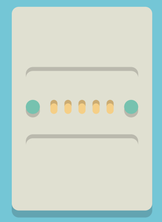
Enterprise Networks

We want to block traffic from senders known to be dangerous.



“INTERNET”

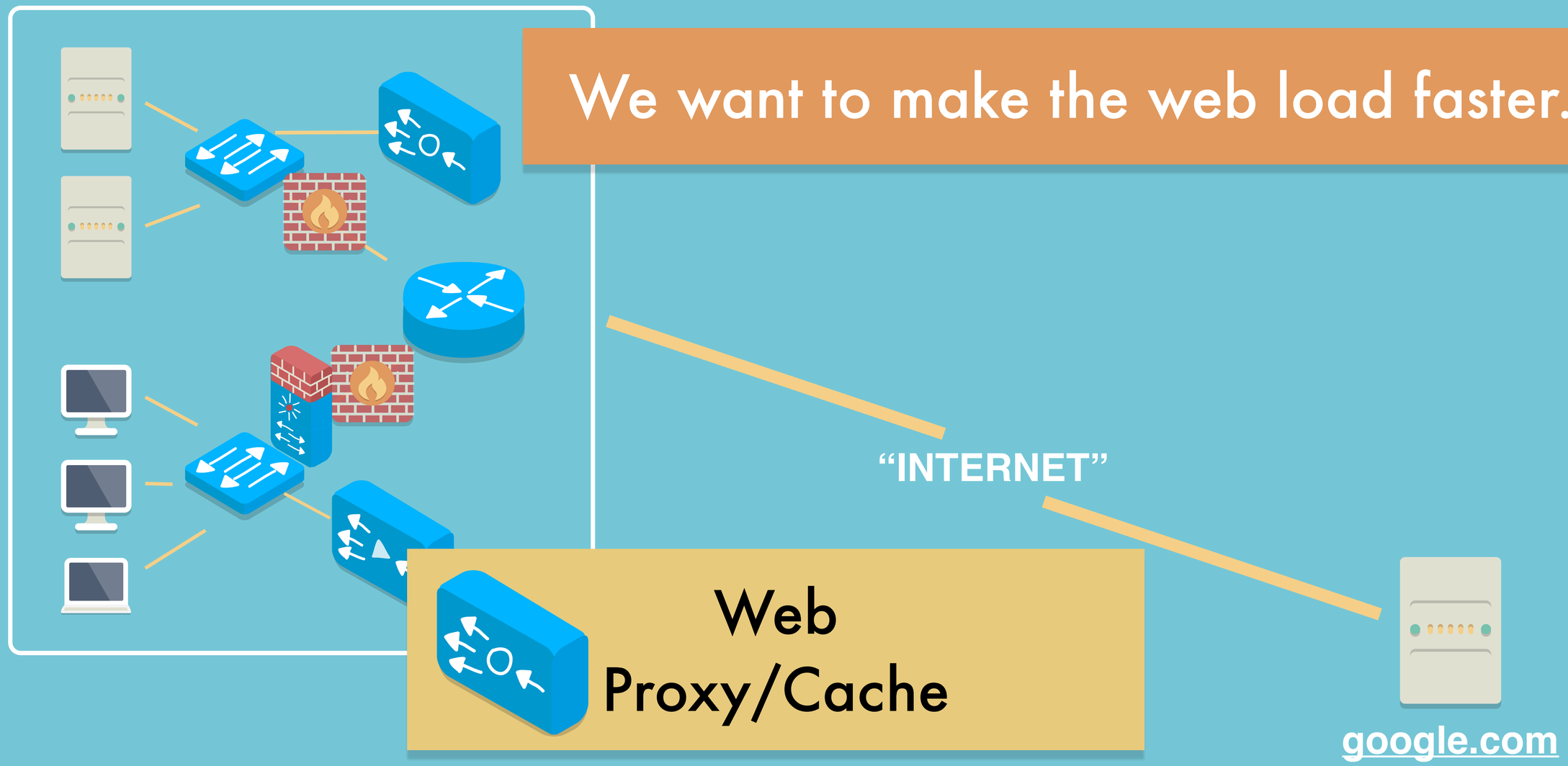
Firewall



google.com

Enterprise Networks

We want to make the web load faster.



Example: Web Proxy

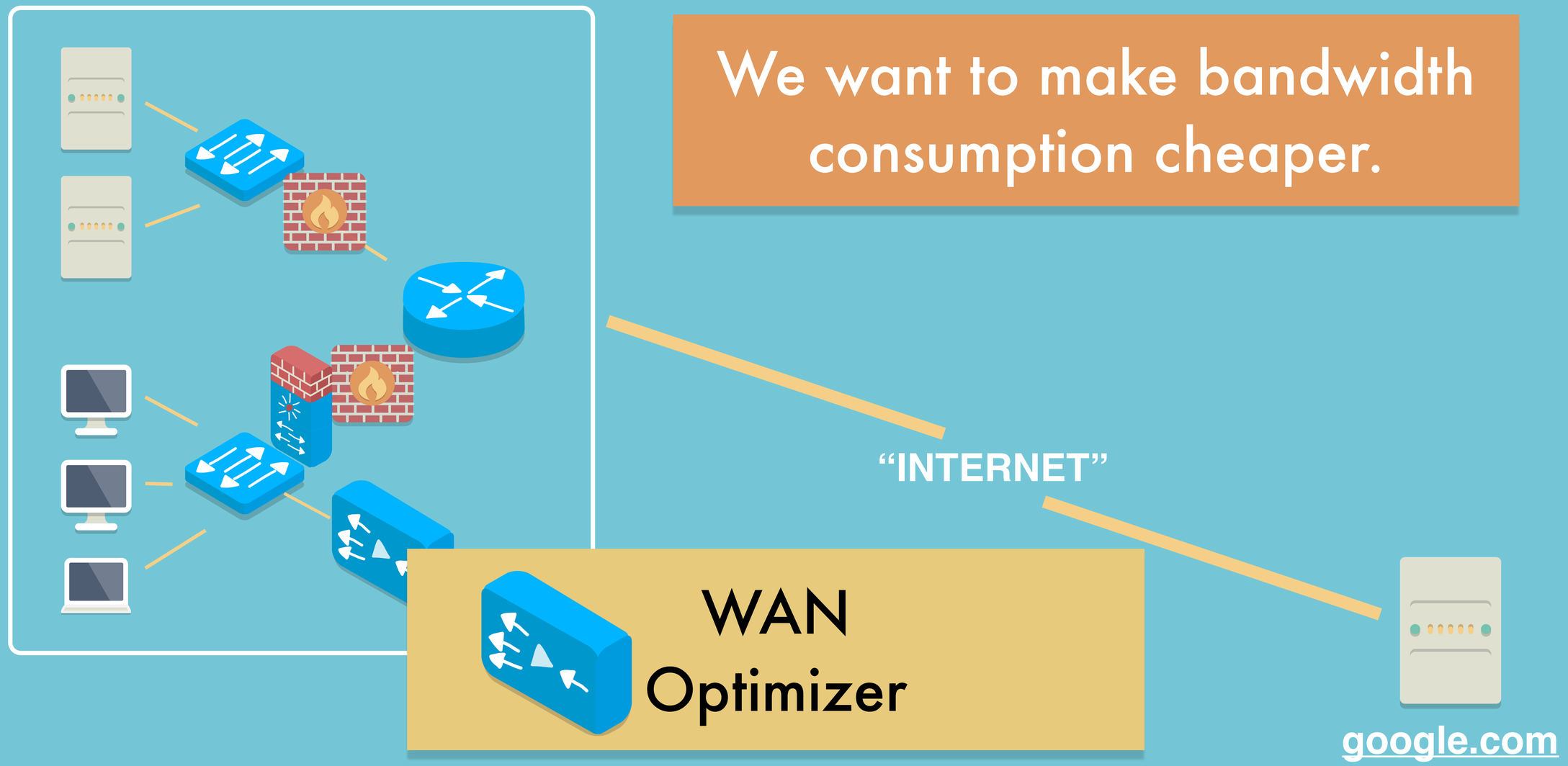
Intercepts HTTP connections and **caches** frequently accessed content.

Maintains dual connections — one to client, one to server!

- If client requests content in cache, serve locally rather than sending request to server.
- If client requests blocked content, deny the request.
- Recall: **forward** and **reverse** proxies (Lecture two weeks ago).

Enterprise Networks

We want to make bandwidth consumption cheaper.



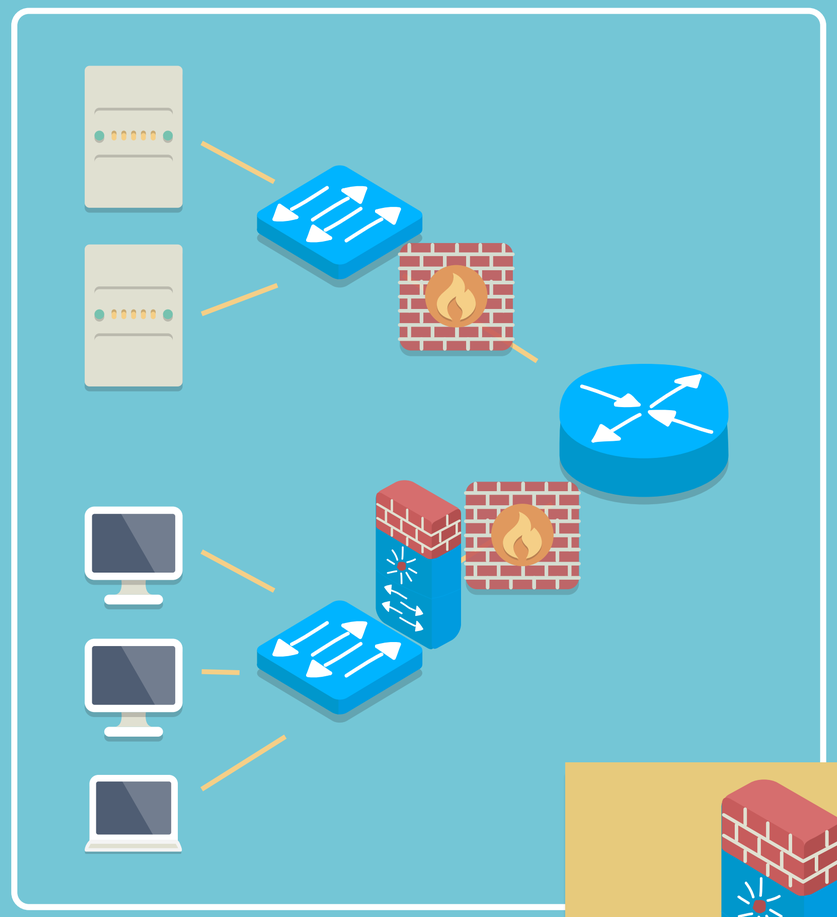
Example: WAN Optimizer

Compresses data so that it uses less bandwidth.

Sits at gateway between enterprise and Internet.

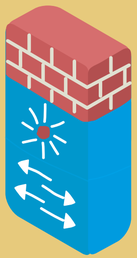
- Outgoing traffic to other sites of the same company is compressed.
- Incoming traffic is uncompressed.
- (Think gzip!)

Enterprise Networks

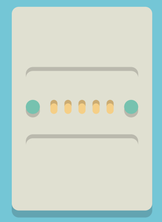


We want to detect and prevent attacks in web traffic and email.

"INTERNET"



Intrusion Detection



google.com

Example: Intrusion Prevention System

Detects anomalous or known-dangerous traffic and **blocks those connections.**

For each connection:

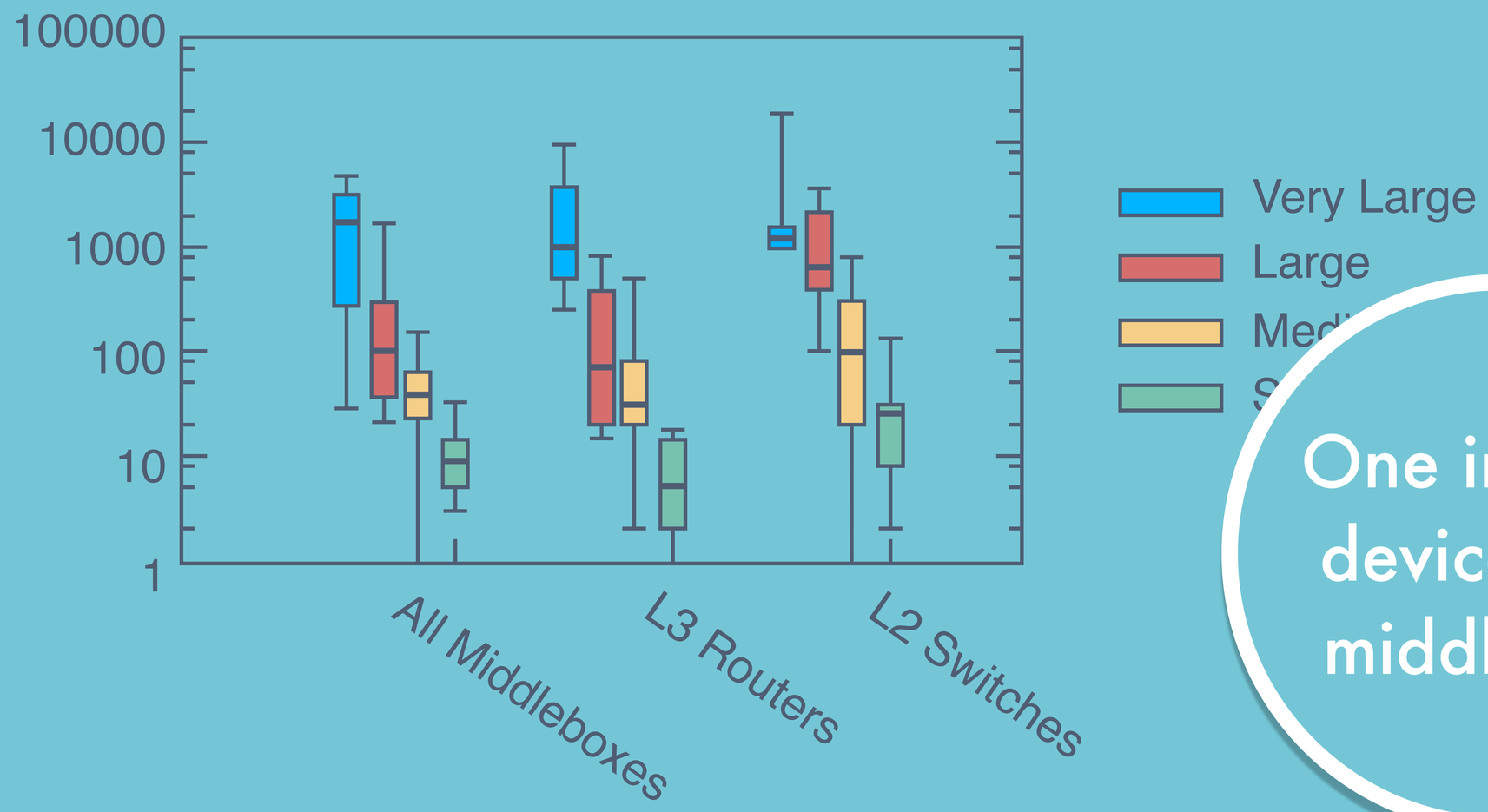
- Looks at port numbers, IP addresses and compares against blacklists.
- Reconstructs connection by stream and scans for malicious terms.
- Logs protocol, IP addresses, time of connection, etc.

…there are a lot of them!

- Network Address Translator
- Evolved Packet Gateway (EPC) Gateways
- Exfiltration Detection
- Forward and Reverse Proxies
- Firewalls
- Transcoders
- Intrusion Detection
- WAN Optimization
- Protocol Accelerators
- IPv4/IPv6 translators…

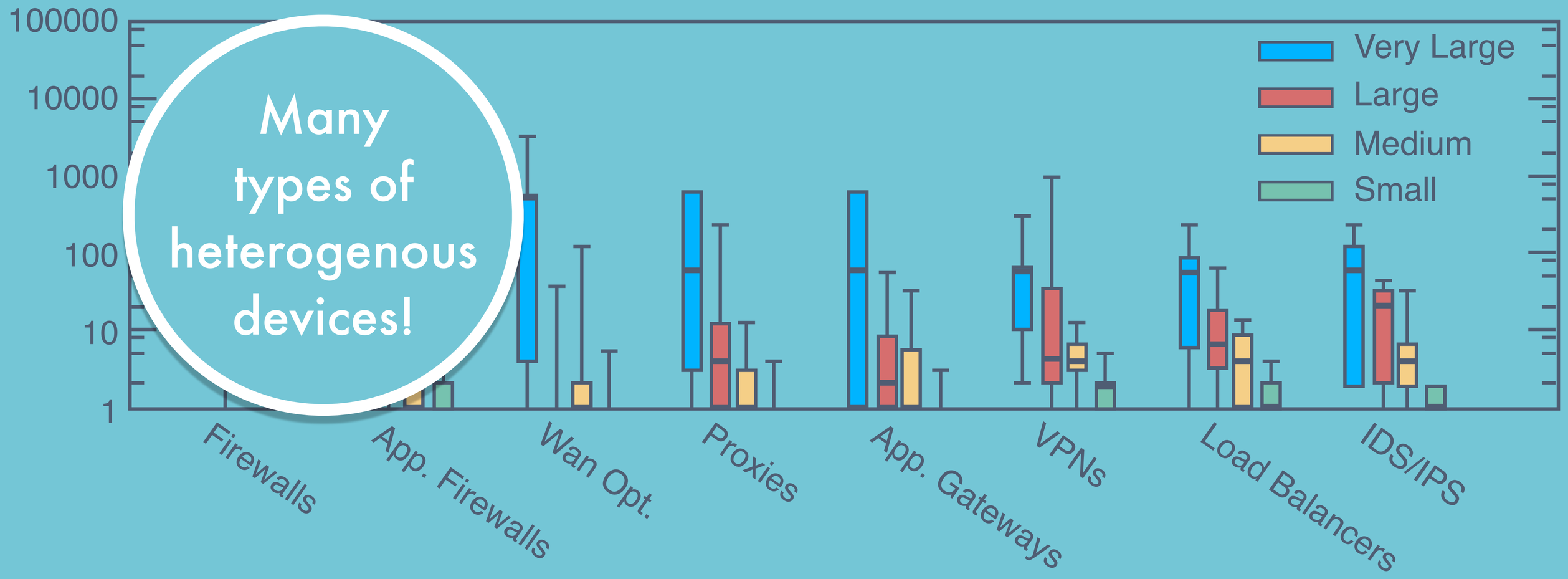
Don’t even try to memorize all of these, just learn the ones from the previous slides ;-)

Very widely deployed…



One in three devices is a middlebox!

…in great heterogeneity!



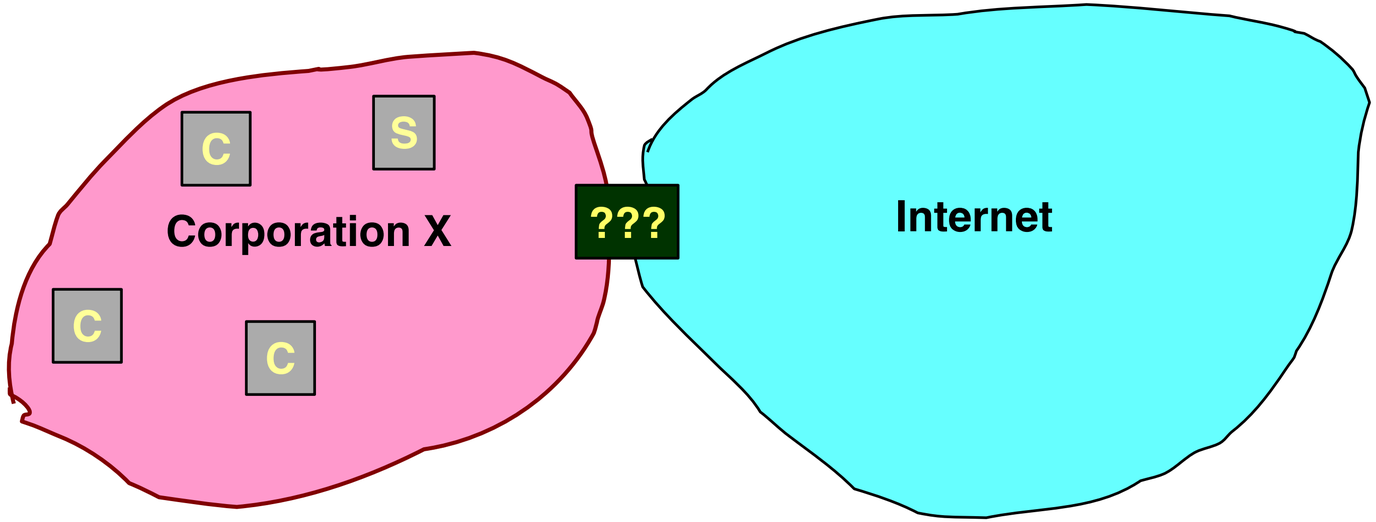
Many types of heterogenous devices!

A middlebox that you have in your house: Network Address Translator

How have we made it so far with IPv4, though?

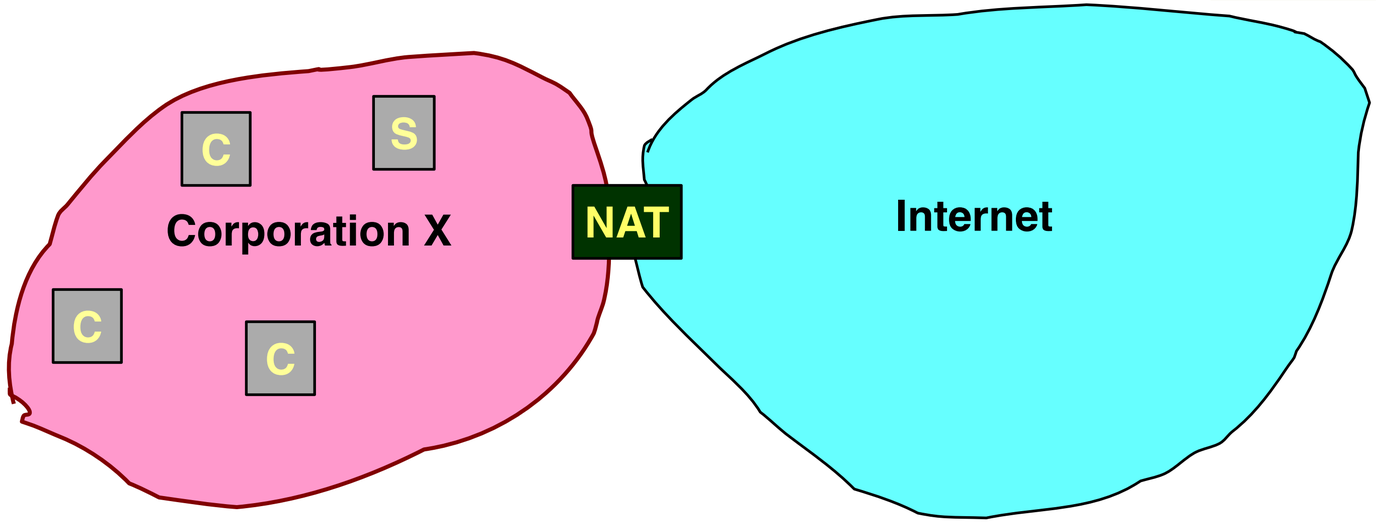
- Original IP Model: Every host has unique IP address
- This has very attractive properties …
  - Any host can communicate with any other host
  - Any host can act as a server
    - Just need to know host ID and port number
- … but the system is open – complicates security
  - Any host can attack any other host
  - It is easy to forge packets
    - Use invalid source address
- … and it places pressure on the address space
  - Every host requires “public” IP address

Challenges When Connecting to Public Internet



- Not enough IP addresses for every host in organization
  - Increasingly hard to get large address blocks
- Security
  - Don't want every machine in organization known to outside world
  - Want to control or monitor traffic in / out of organization

But not All Hosts are Equal!

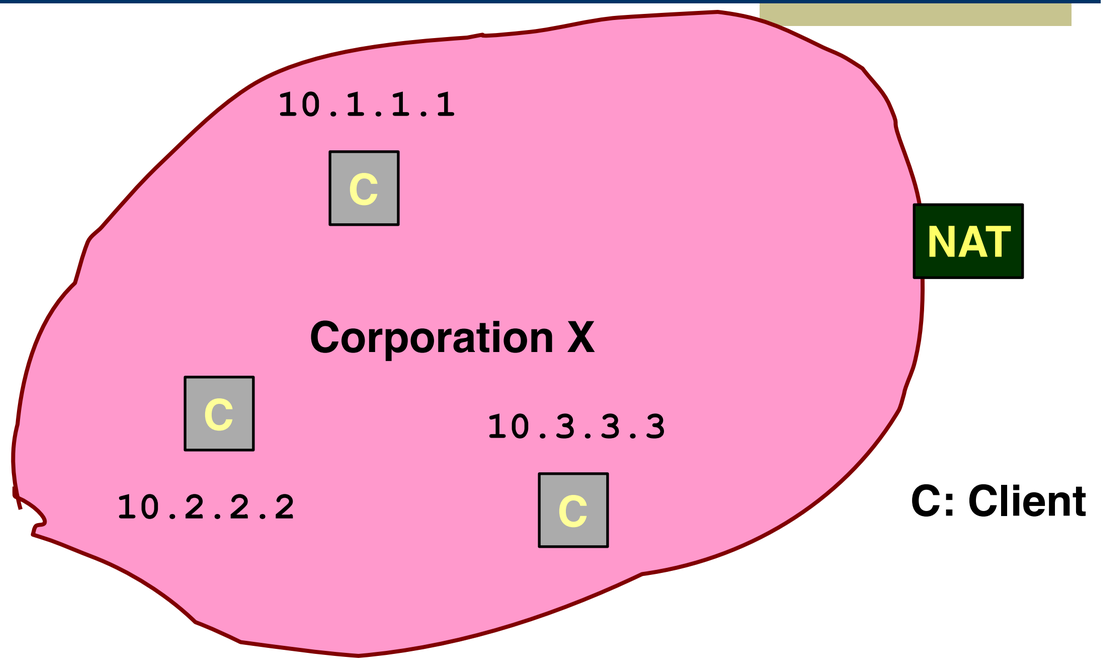


- Most machines within organization are used by individuals
  - For most applications, they act as clients
- Only a small number of machines act as servers for the entire organization
  - E.g., mail server, web, ..
  - All traffic to outside passes through firewall

(Most) machines within organization do not need public IP addresses!

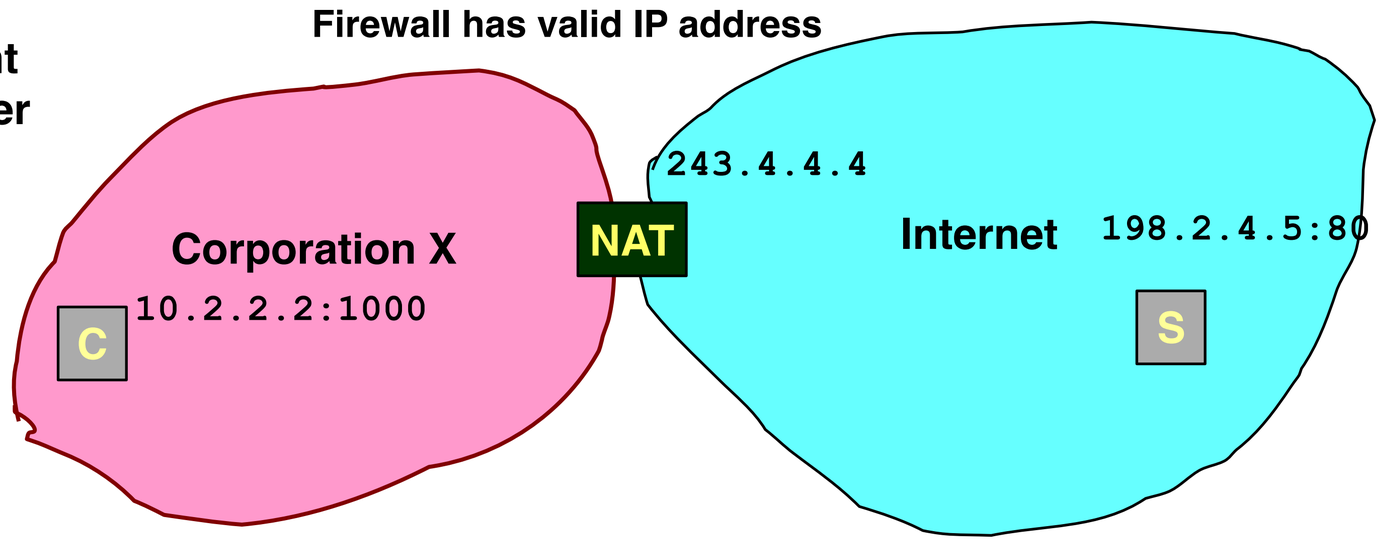
Reducing Address Use: Network Address Translation

- Within organization: assign each host a private IP address
  - IP addresses blocks 10/8 & 192.168/16 are set aside for this
  - Route within organization by IP protocol
  - Can do subnetting, ..



- The NAT translates between public and private IP addresses as packets travel to/from the public Internet
  - It does not let any packets from internal nodes “escape”
  - Outside world does not need to know about internal addresses

NAT: Opening Client Connection

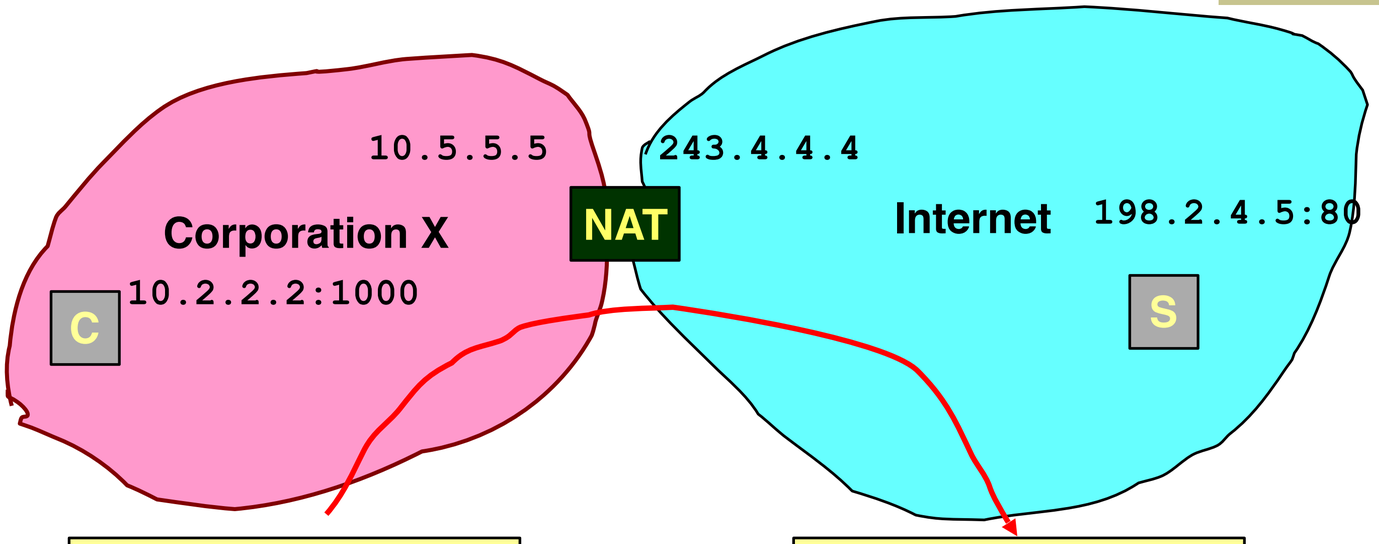


- Client 10.2.2.2 wants to connect to server 198.2.4.5:80
  - OS assigns ephemeral port (1000)
- Connection request intercepted by firewall
  - Maps client to port of firewall (5000)
  - Creates NAT table entry

Int Addr	Int Port	NAT Port
10.2.2.2	1000	5000

NAT: Client Request

C: Client
S: Server



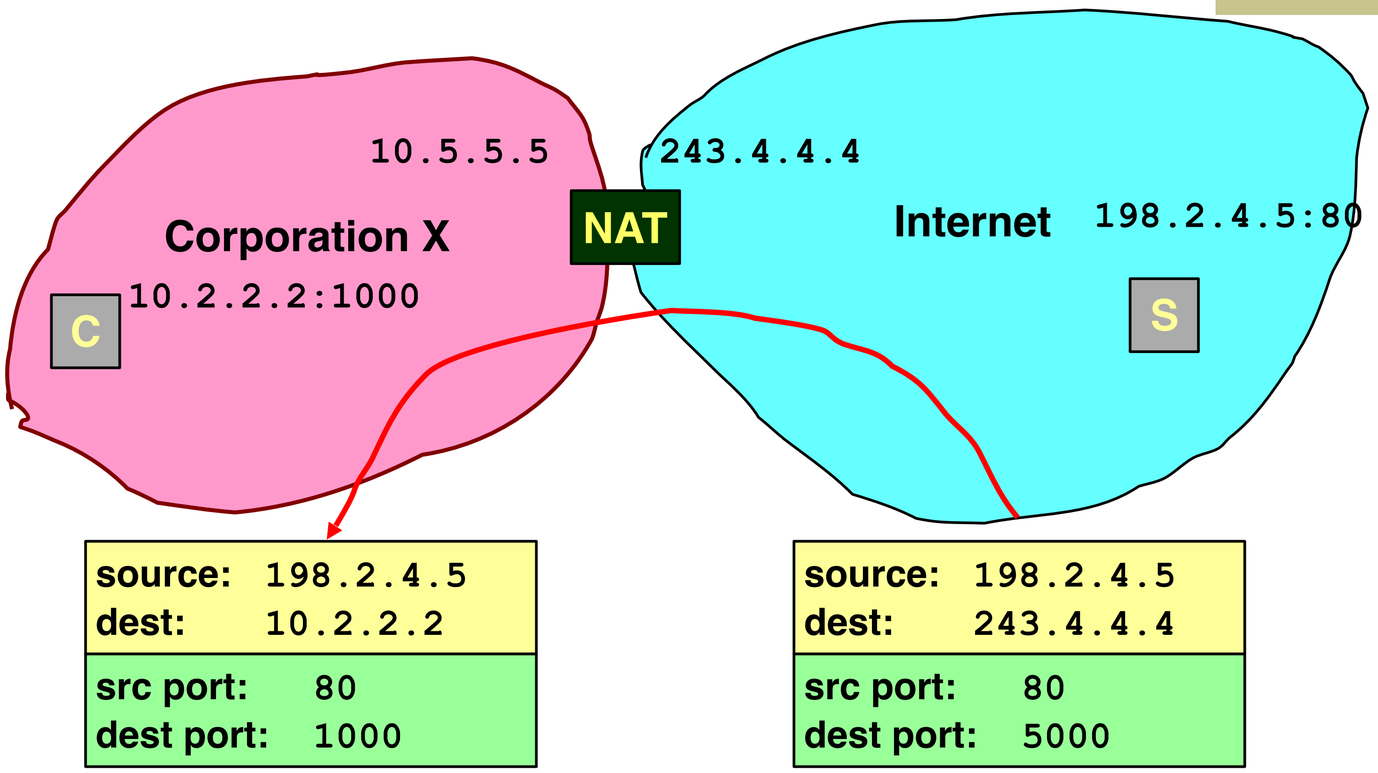
source:	10.2.2.2
dest:	198.2.4.5
src port:	1000
dest port:	80

source:	243.4.4.4
dest:	198.2.4.5
src port:	5000
dest port:	80

Int Addr	Int Port	NAT Port
10.2.2.2	1000	5000

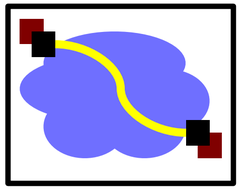
NAT: Server Response

C: Client
S: Server



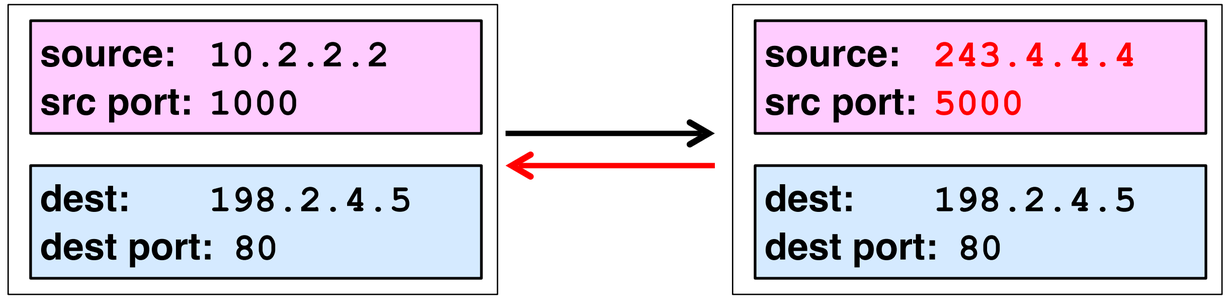
Int Addr	Int Port	NAT Port
10.2.2.2	1000	5000

Client Request Mapping



Private network:

Public Internet:



- NAT manages mapping between two four-tuples
- Mapping must be unique: one to one
- Must respect practical constraints
  - Cannot modify server IP address or port number
  - Client has limited number of IP addresses, often 1
  - Mapping client port numbers is important!
- Mapping must be consistent
  - The same for all packets in a communication session

Let’s Try It

Table Mapping

WAN	LAN
138.76.29.4-8000	10.0.0.5-5000
138.76.29.4-8001	10.0.0.6-5000
138.76.29.4-8002	10.0.10-6000
138.76.29.4-8003	10.0.1.101-6001
138.76.29.4-8004	10.0.07-7000

What if 34.26.52.42 sends to 138.76.29.7:8002?

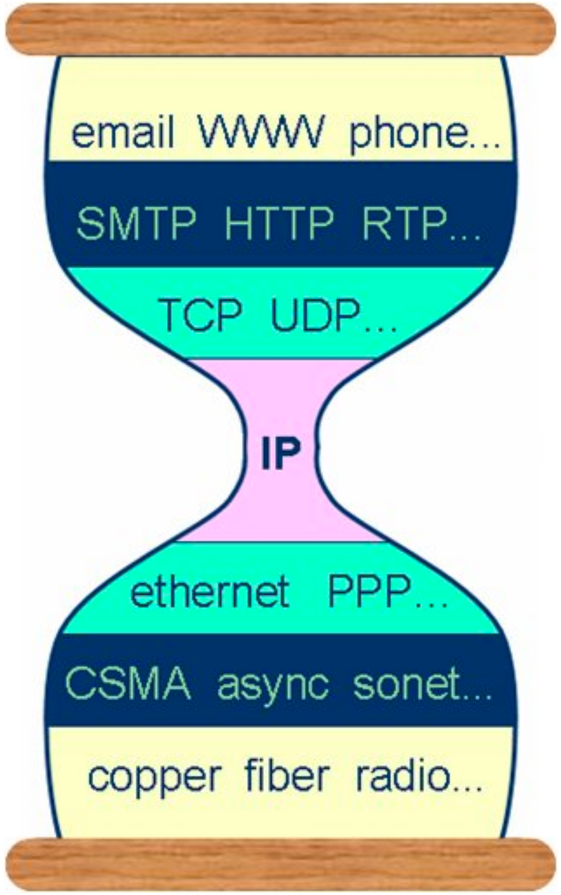
WAN	LAN
138.76.29.4-8000	10.0.0.5-5000
138.76.29.4-8001	10.0.0.6-5000
138.76.29.4-8002	10.0.10-6000
138.76.29.4-8003	10.0.1.101-6001
138.76.29.4-8004	10.0.07-7000

What if 34.26.52.42 sends to 138.76.29.7:9043?

WAN	LAN
138.76.29.4-8000	10.0.0.5-5000
138.76.29.4-8001	10.0.0.6-5000
138.76.29.4-8002	10.0.10-6000
138.76.29.4-8003	10.0.1.101-6001
138.76.29.4-8004	10.0.07-7000

Now that you know what middleboxes are

Where do middleboxes fit in the model?



In what ways are middleboxes at the application layer?

In what ways are middleboxes at the network layer?

MIDDLEBOXES ARE CONTROVERSIAL

The rest of this lecture

- The End to End Argument (aka, why middleboxes are controversial)
- Why we deploy middleboxes anyway
- Some challenges they leave us with
- A new movement called Network Functions Virtualization

The rest of this lecture

- **The End to End Argument**
- Why we deploy middleboxes anyway
- Some challenges they leave us with
- A new movement called Network Functions Virtualization

“Careful File Transfer”

At host A the file transfer program calls upon the file system to read the file from the disk, where it resides on several tracks, and the file system passes it to the file transfer program in fixed-size blocks chosen to be disk-format independent.

A

File System

Program

Network

B

File System

Program

Network

“Careful File Transfer”

At host A the file transfer program calls upon the file system to read the file from the disk, where it resides on several tracks, and the file system passes it to the file transfer program in fixed-size blocks chosen to be disk-format independent.

A

File System

Program

Network

Alternate *ACM* SIG Proceedings Paper in LaTeX Format*

[Extended Abstract]†

B

File System

Program

Network

“Careful File Transfer”

At host A the file transfer program calls upon the file system to read the file from the disk, where it resides on several tracks, and the file system passes it to the file transfer program in fixed-size blocks chosen to be disk-format independent.

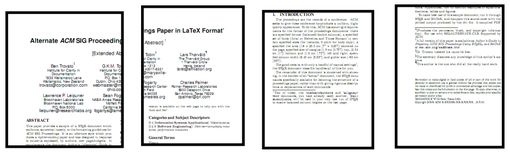
A

B

File System

Program

Network



File System

Program

Network

“Careful File Transfer”

Also at host A the file transfer program asks the data communication system to transmit the file using some communication protocol that involves splitting the data into packets. The packet size is typically different from the file block size and the disk track size.

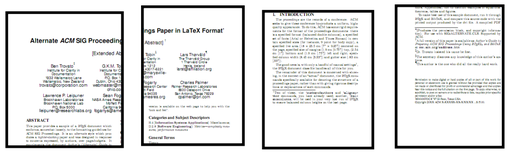
A

B

File System

Program

Network



File System

Program

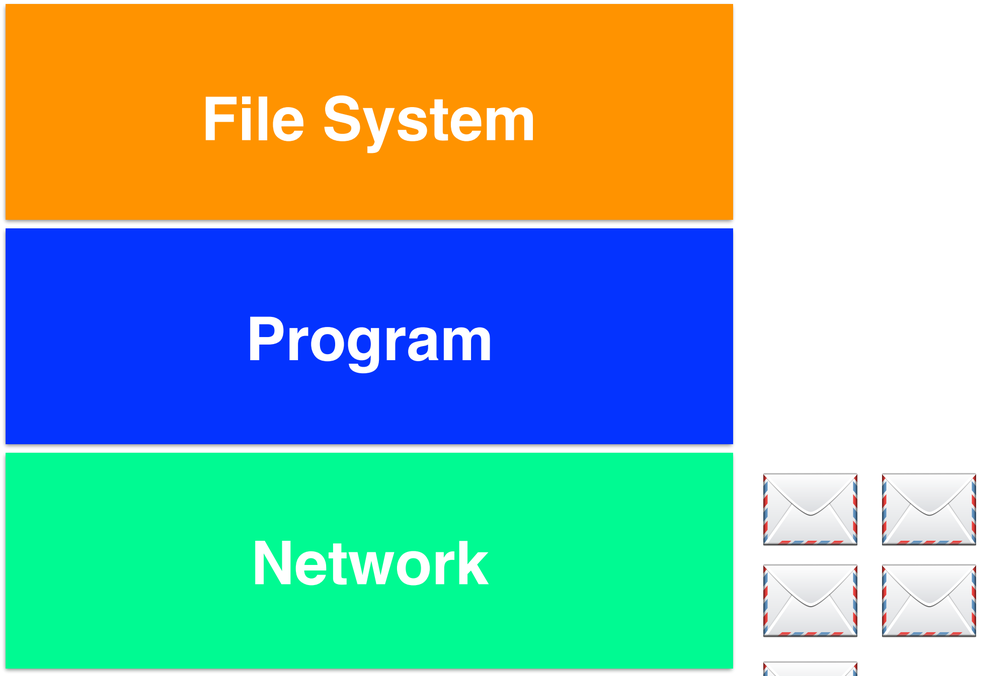
Network

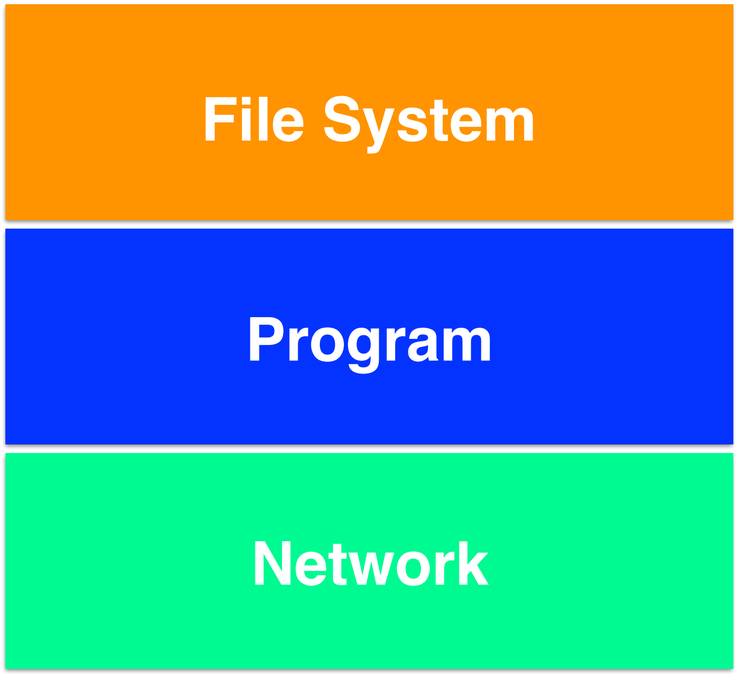
“Careful File Transfer”

Also at host A the file transfer program asks the data communication system to transmit the file using some communication protocol that involves splitting the data into packets. The packet size is typically different from the file block size and the disk track size.

A

B





“Careful File Transfer”

The data communication network moves the packets from computer A to computer B.

A

File System

Program

Network

B

File System

Program

Network

“Careful File Transfer”

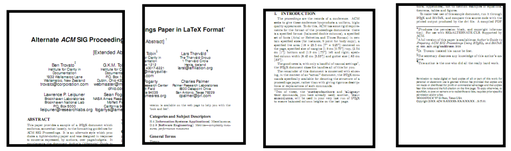
At host B a data communication program removes the packets from the data communication protocol and hands the contained data on to a second part of the file transfer application, the part that operates within host B.

A

File System

Program

Network



B

File System

Program

Network

“Careful File Transfer”

At host B, the file transfer program asks the file system to write the received data on the disk of host B.

A

File System

Program

Network

B

File System

Program

Network

What if Zeeshan later reads the file and find it is corrupted? What could have gone wrong?

The file, though originally written correctly onto the disk at host A, if read now may contain incorrect data, perhaps because of hardware faults in the disk storage system.

The software of the file system, the file transfer program, or the data communication system might make a mistake in buffering and copying the data of the file, either at host A or host B.

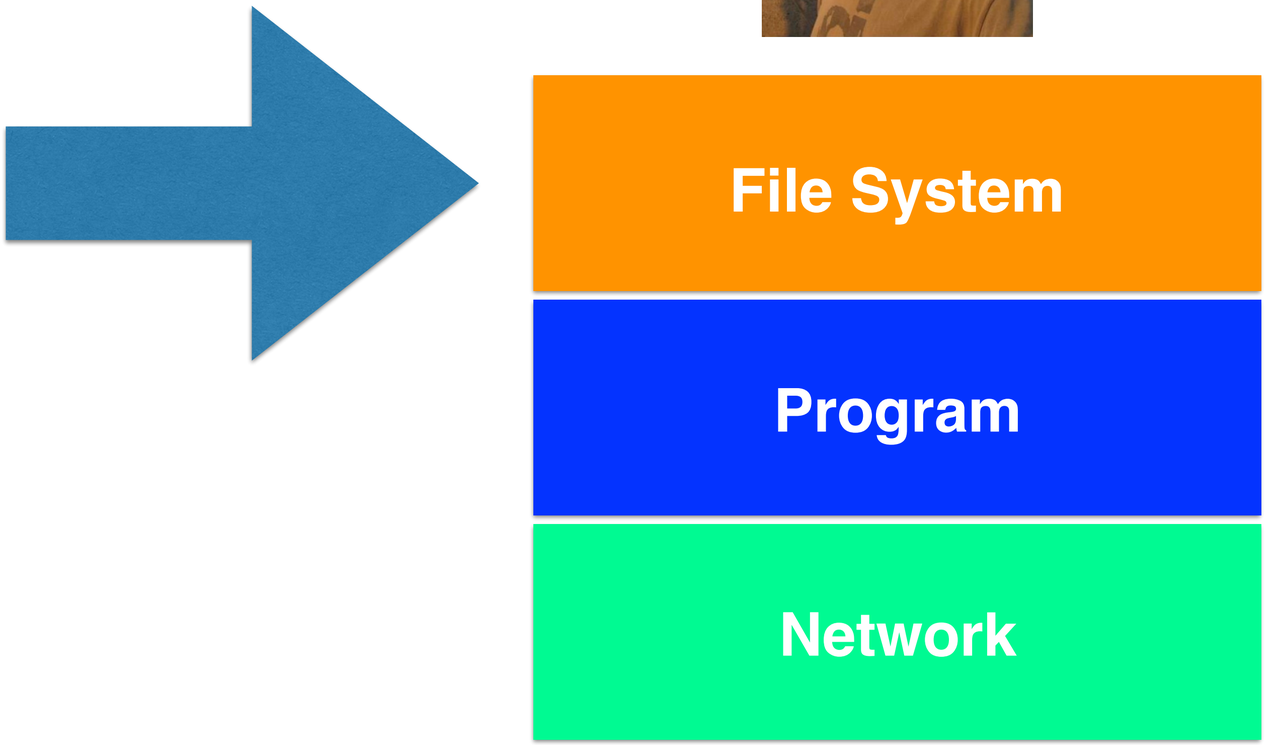
The hardware processor or its local memory might have a transient error while doing the buffering and copying, either at host A or host B.

The communication system might drop or change the bits in a packet, or lose a packet or deliver a packet more than once.

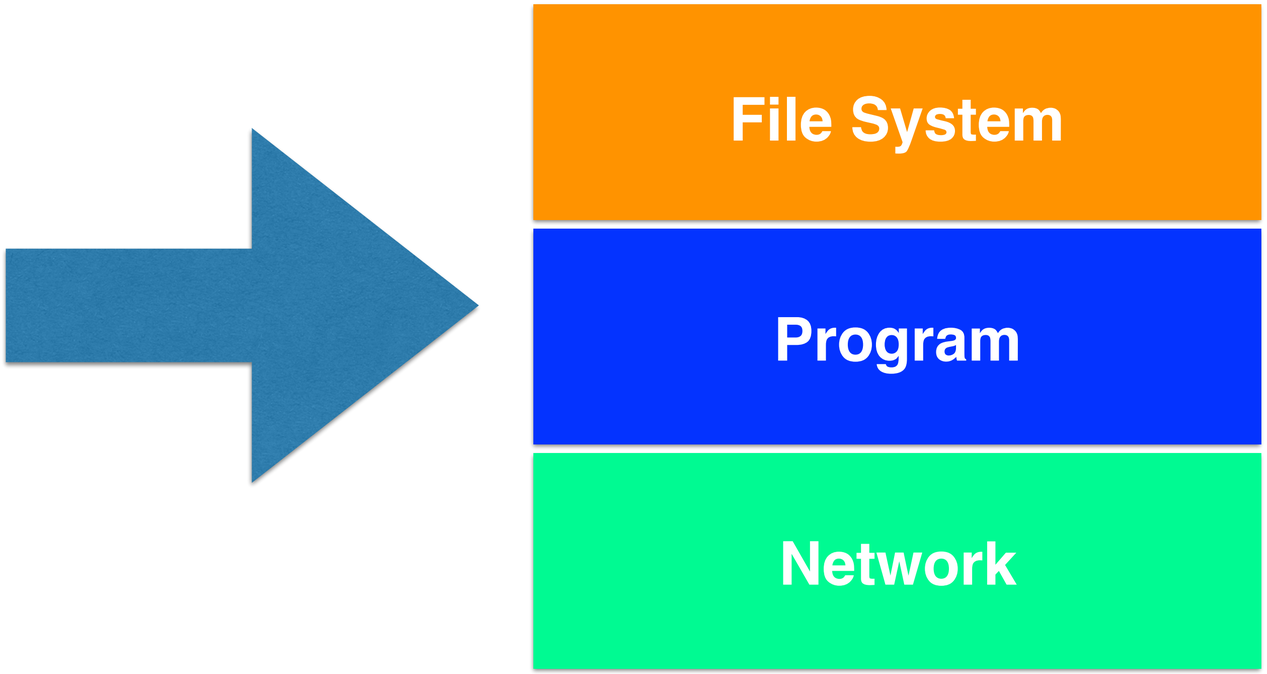
Either of the hosts may crash part way through the transaction after performing an unknown amount (perhaps all) of the transaction.

How do we re-design our system to make sure the file doesn’t get corrupted?

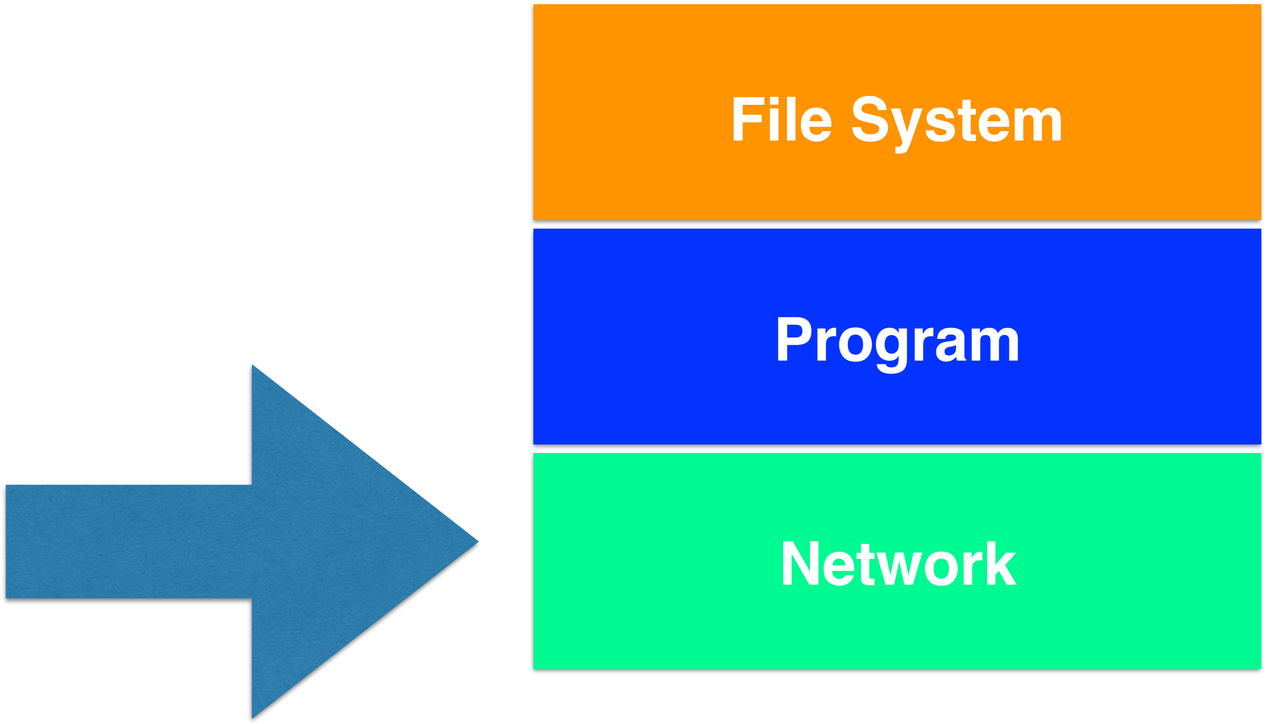
B



B



B



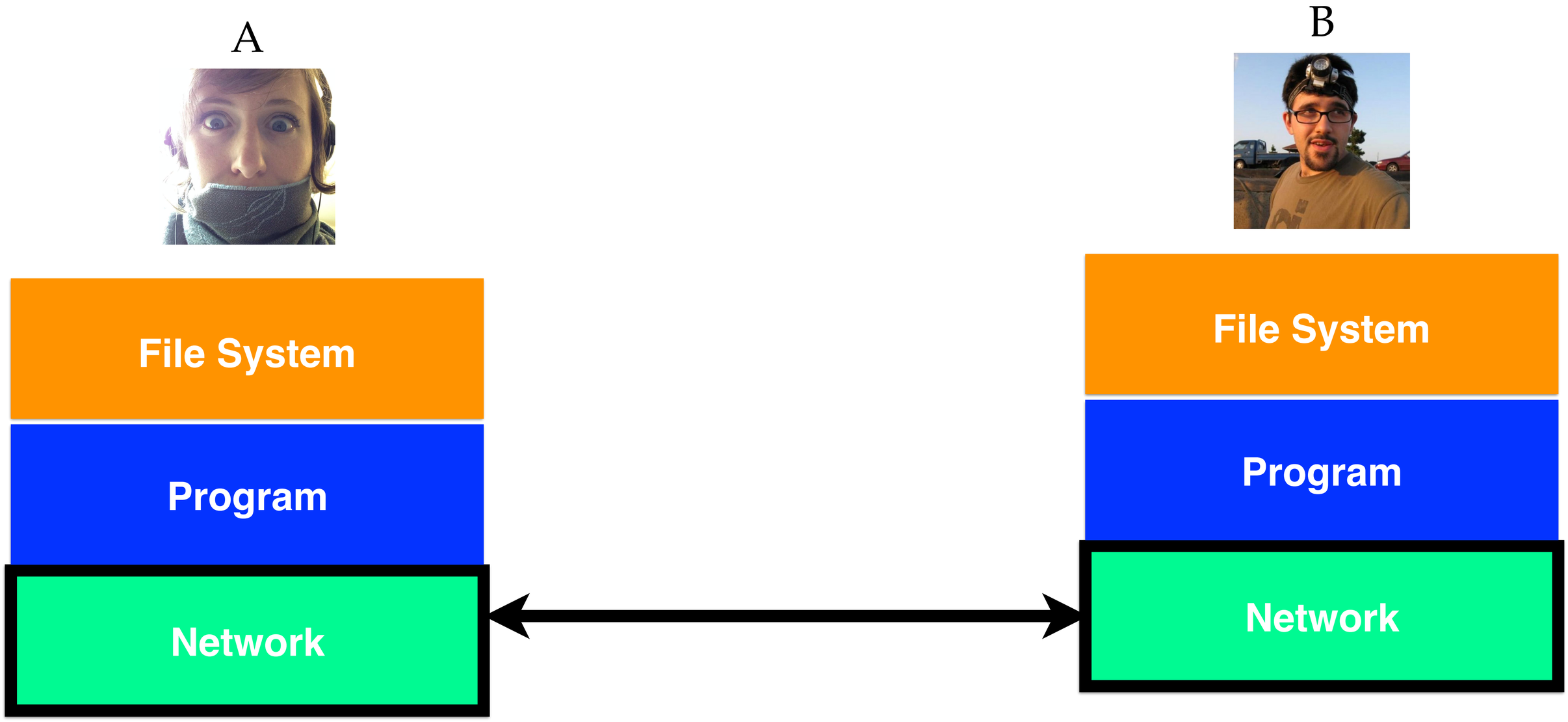
The End-to-End Argument

[If] the function in question can completely and correctly be implemented with the knowledge and help of the application standing at the endpoints of the communication system:

[Then] providing that questioned function as a feature of the communication system [or lower layer] is not possible.

[However], sometimes an incomplete version of the function provided by the communication system may be useful as a performance enhancement.

Let's say we had a perfectly reliable network



Would that solve our reliability problem?

The file, though originally written correctly onto the disk at host A, if read now may contain incorrect data, perhaps because of hardware faults in the disk storage system.

The software of the file system, the file transfer program or the data communication system might make a mistake in buffering and copying the data of the file, either at host A or host B.

The hardware processor or its local memory might have a transient error while doing the buffering and copying, either at host A or host B.

The communication system might drop or change the bits in a packet, or lose a packet or deliver a packet more than once.

Either of the hosts may crash part way through the transaction after performing an unknown amount (perhaps all) of the transaction.

Would that solve our reliability problem?

The file, though originally written correctly onto the disk at host A, if read now may contain incorrect data, perhaps because of hardware faults in the disk storage system.

The software of the file system, the file transfer program or the data communication system might make a mistake in buffering and copying the data of the file, either at host A or host B.

The hardware processor or its local memory might have a transient error while doing the buffering and copying, either at host A or host B.

The communication system might drop or change the bits in a packet, or lose a packet or deliver a packet more than once.

Either of the hosts may crash part way through the transaction after performing an unknown amount (perhaps all) of the transaction.

Well, that wasn’t very helpful…

“End to End Check and Retry”

A

File System

Program

Network

B

File System

Program

Network

“End to End Check and Retry”

Read file and its checksum from disk.
Verify file + checksum.
Send File AND Checksum.

A

File System

Program

Network

B

File System

Program

Network

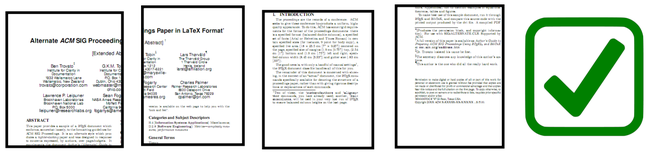
“End to End Check and Retry”

A

File System

Program

Network



B

File System

Program

Network

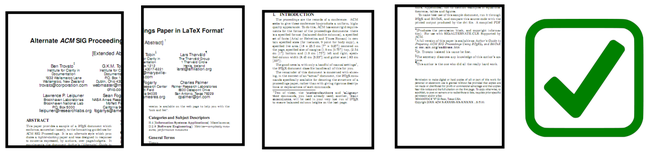
“End to End Check and Retry”

A

File System

Program

Network



B

File System

Program

Network

“Careful File Transfer”

The data communication network moves the packets from computer A to computer B.

A

File System

Program

Network

B

File System

Program

Network

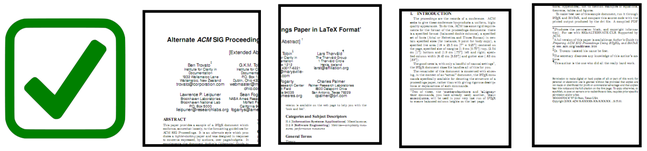
“End to End Check and Retry”

A

File System

Program

Network



B

File System

Program

Network

“End to End Check and Retry”

A

File System

Program

Network

Write file and checksum to disk.
Then read back and double-check that checksum + file verify.

Alternate *ACM* SIG Proceedings Paper in LaTeX Format

[Extended Abstract]

B

File System

Program

Network

“End to End Check and Retry”

A

File System

Program

Network

If Checksum doesn’t match?
Just ask Justine to re-send.
(ie, try all over again!)

Alternate *ACM* SIG Proceedings Paper in LaTeX Format

[Extended Abstract]

B

File System

Program

Network

Would that solve our reliability problem?

The file, though originally written correctly onto the disk at host A, if read now may contain incorrect data, perhaps because of hardware faults in the disk storage system.

The software of the file system, the file transfer program or the data communication system might make a mistake in buffering and copying the data of the file, either at host A or host B.

The hardware processor or its local memory might have a transient error while doing the buffering and copying, either at host A or host B.

The communication system might drop or change the bits in a packet, or lose a packet or deliver a packet more than once.

Either of the hosts may crash part way through the transaction after performing an unknown amount (perhaps all) of the transaction.

Lesson: If you can do it at the "higher" layer, don't bother implementing it at a lower layer.

Don't waste your time!

Avoid causing confusion.

We've already seen some examples in this class.

TCP Congestion Control?

Circuit Switched Networking?

Packet fragment reassembly?

The End-to-End Argument

One Exception!

[If] the function in [illegible]estion can completely and correctly be implemented with [illegible]knowledge and help of the application standing a[illegible] the communication system:

[Then] providi[illegible]stic[illegible]ed function as a feature of the communica[illegible]n syste[illegible] [or [illegible]er layer] is not possible.

[However], sometimes an incomplete version of the function provided by the communication system may be useful as a performance enhancement.

What if 90% of my loss really was happening at the network layer?

A

File System

Program

Network

As a performance optimization, you might want to implement it in the lower layer anyway (redundantly).

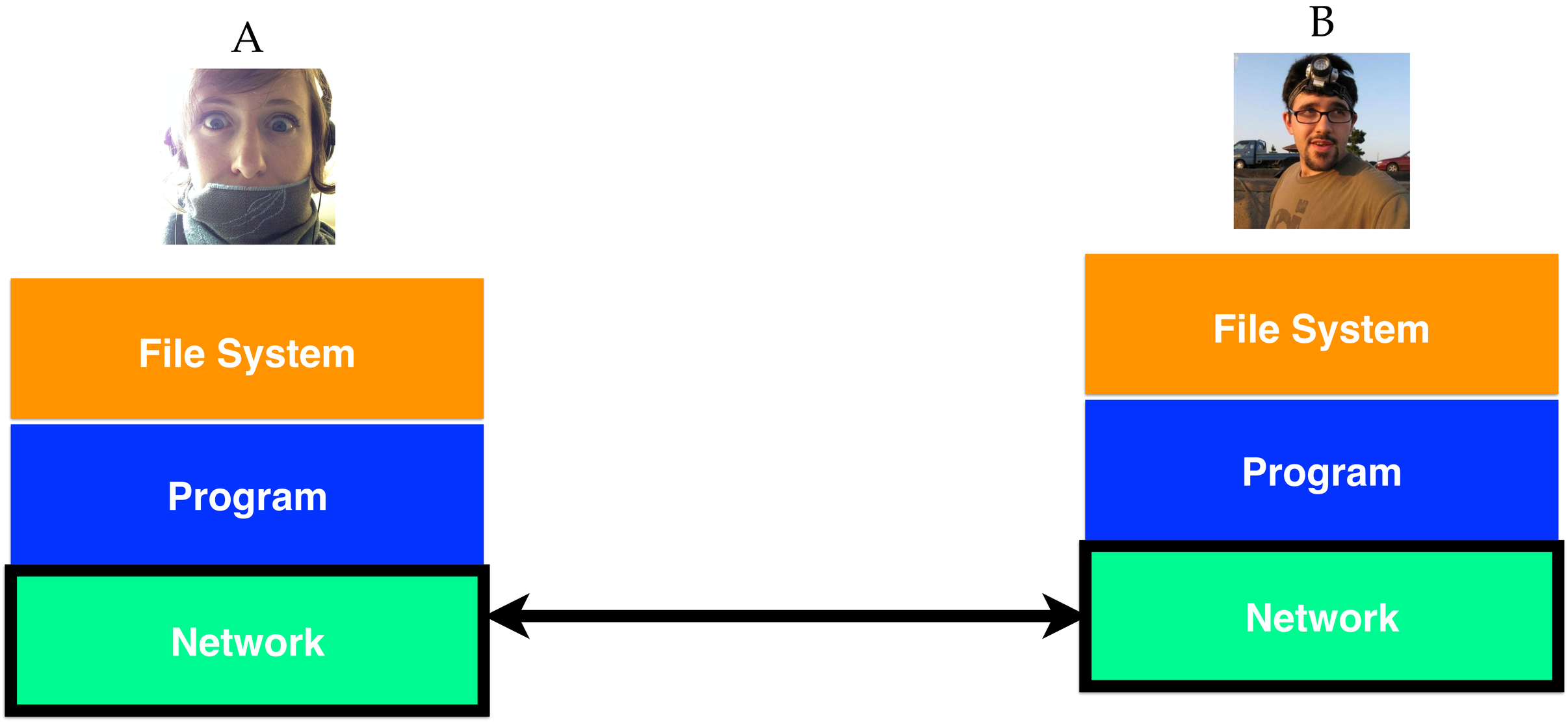
B

File System

Program

Network

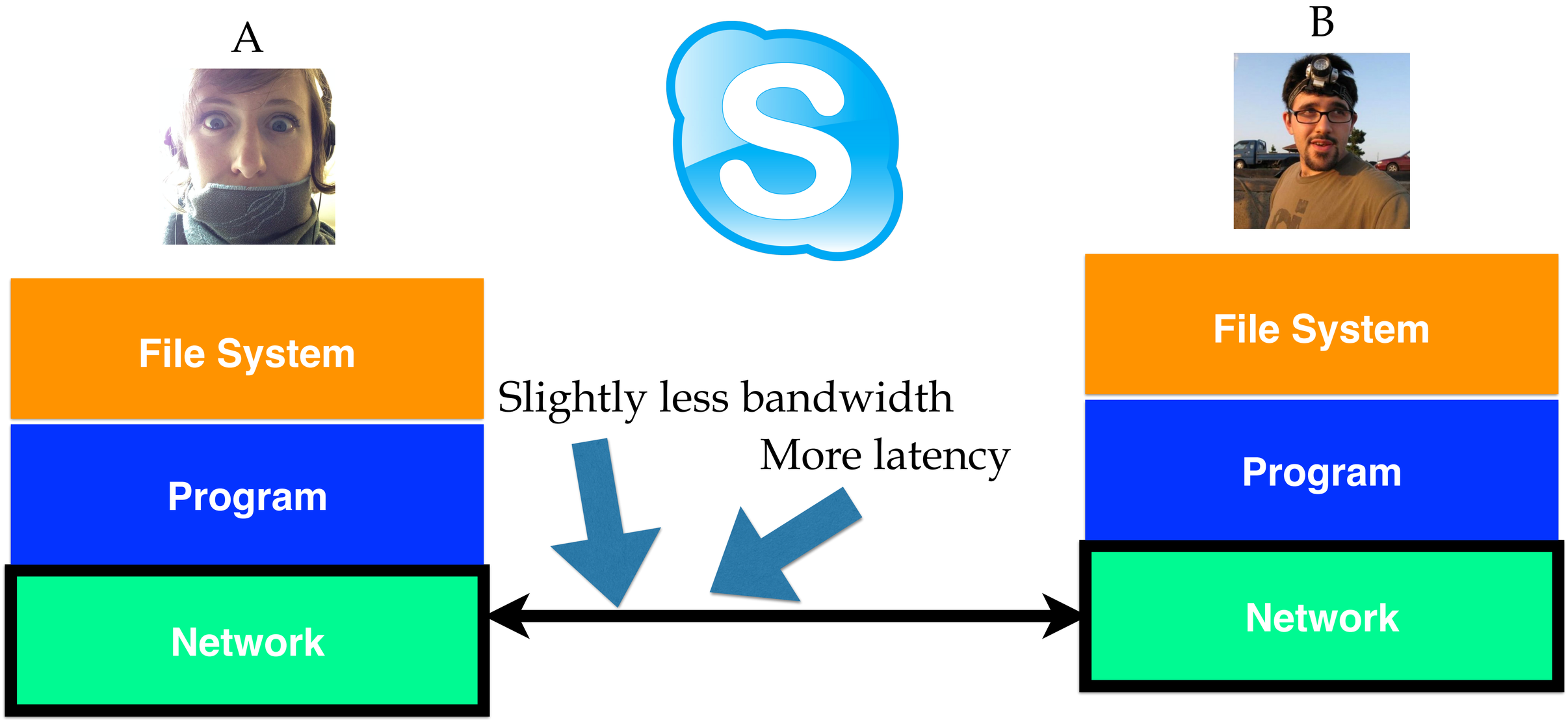
“End to End Check and Retry” + A Reliable Network



The “Strong” End-to-End Argument

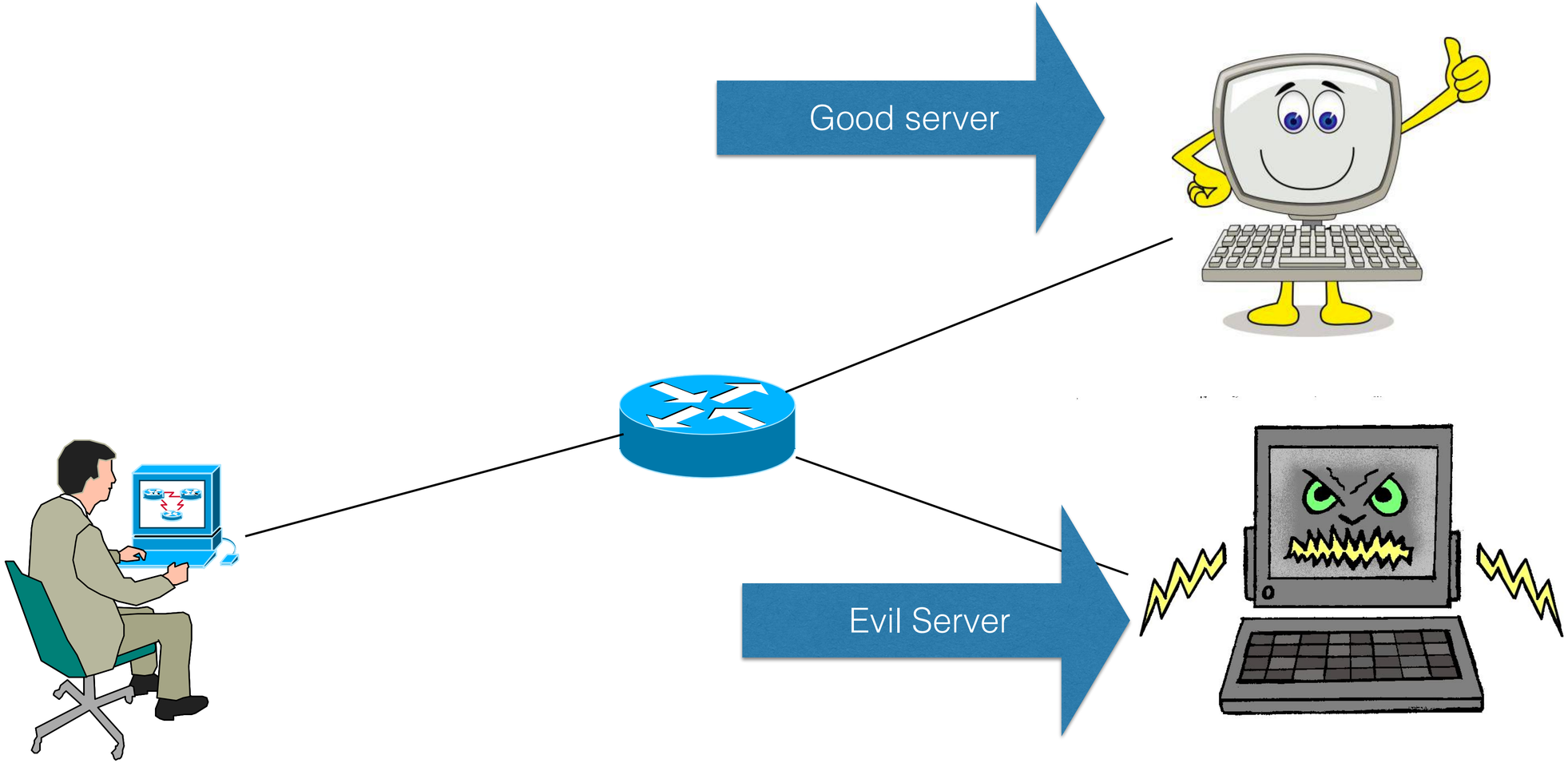
It’s not just a waste of time to put non-essential functionality at lower layers: *it’s actually harmful*.

“End to End Check and Retry” + A Reliable Network

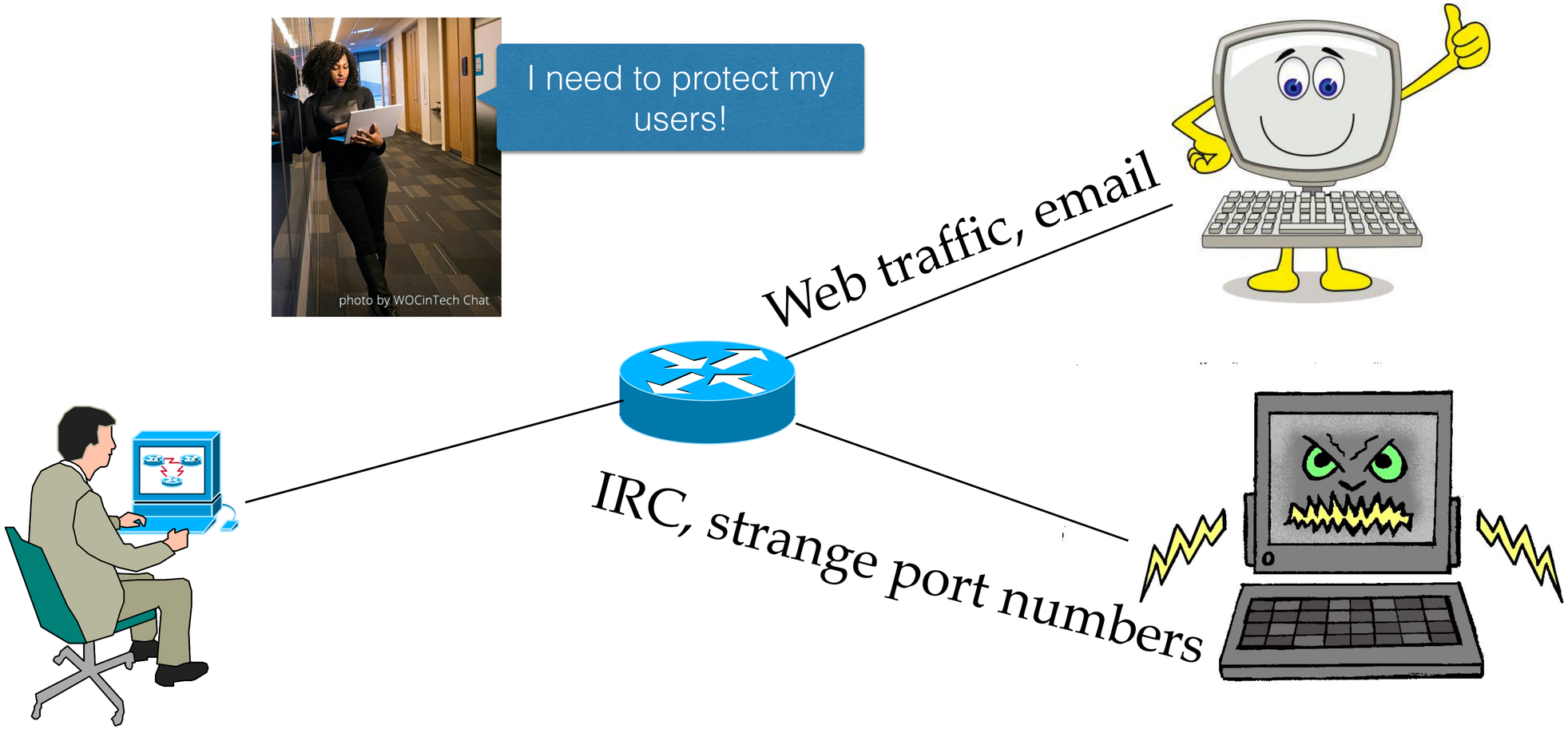


Some applications may be *constrained* by the new functionality.

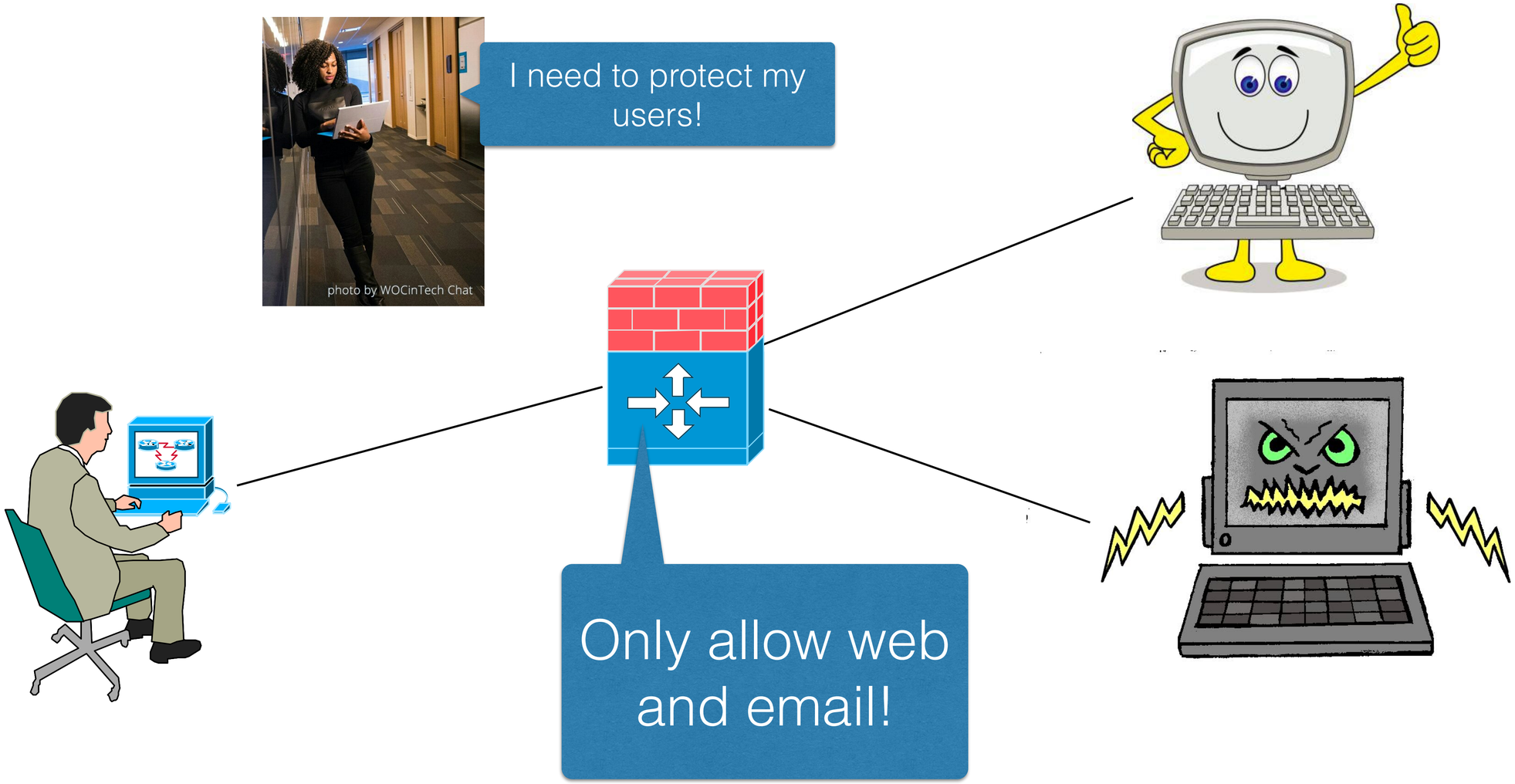
Firewalls and Intrusion Detection



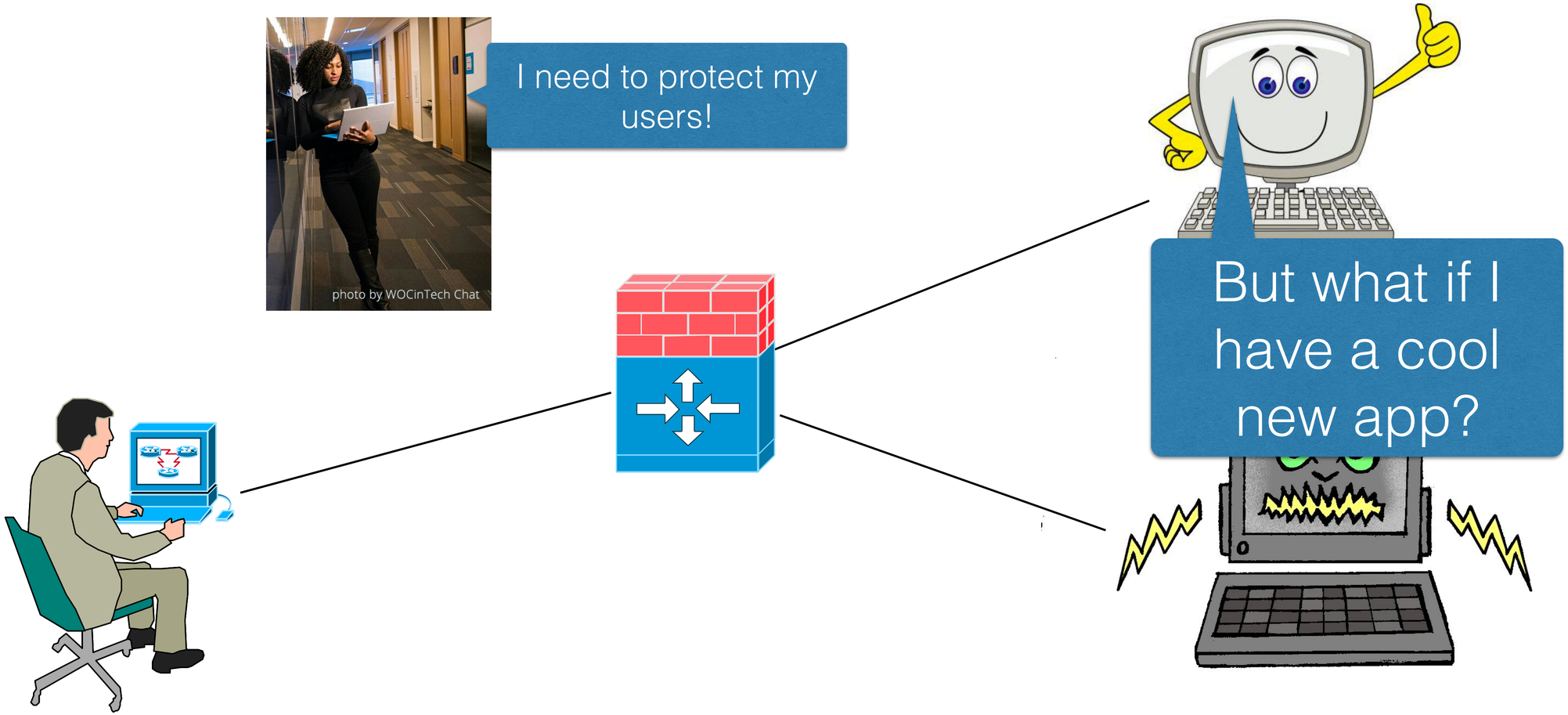
Firewalls and Intrusion Detection



Firewalls and Intrusion Detection



Firewalls and Intrusion Detection



Have you ever had your network stop you from doing something? What happened?

The rest of this lecture

- **The End to End Argument**
- Why we deploy middleboxes anyway
- Some challenges they leave us with
- A new movement called Network Functions Virtualization

The rest of this lecture

- The End to End Argument
- **Why we deploy middleboxes anyway**
- Some challenges they leave us with
- A new movement called Network Functions Virtualization

What do you think?

Two Reasons We Deploy Middleboxes

- (1) It's a fast, drop in way to upgrade network features.
- (2) It's a centralized point of control.

(1) A fast way to upgrade your network

- Remember address-space exhaustion?
  - IPv6 is the clean solution, but it takes a long time to upgrade because *everyone must update* their infrastructure and code.
  - The fast solution: Network Address Translators. Drop-in, no one needs to make any changes (for the most part) except network administrator.

(1) A fast way to upgrade your network

- Remember DDoS and attack traffic?
  - Many proposals exist to upgrade routers so that *receivers tell routers to start blocking certain traffic sources.* Once again… this requires upgrades to routers and hosts — lots of changes.
    - *See "IP pushback" work if you're curious*
  - The fast solution: Firewalls. Drop-in, no one needs to make any changes (for the most part) except network administrator.

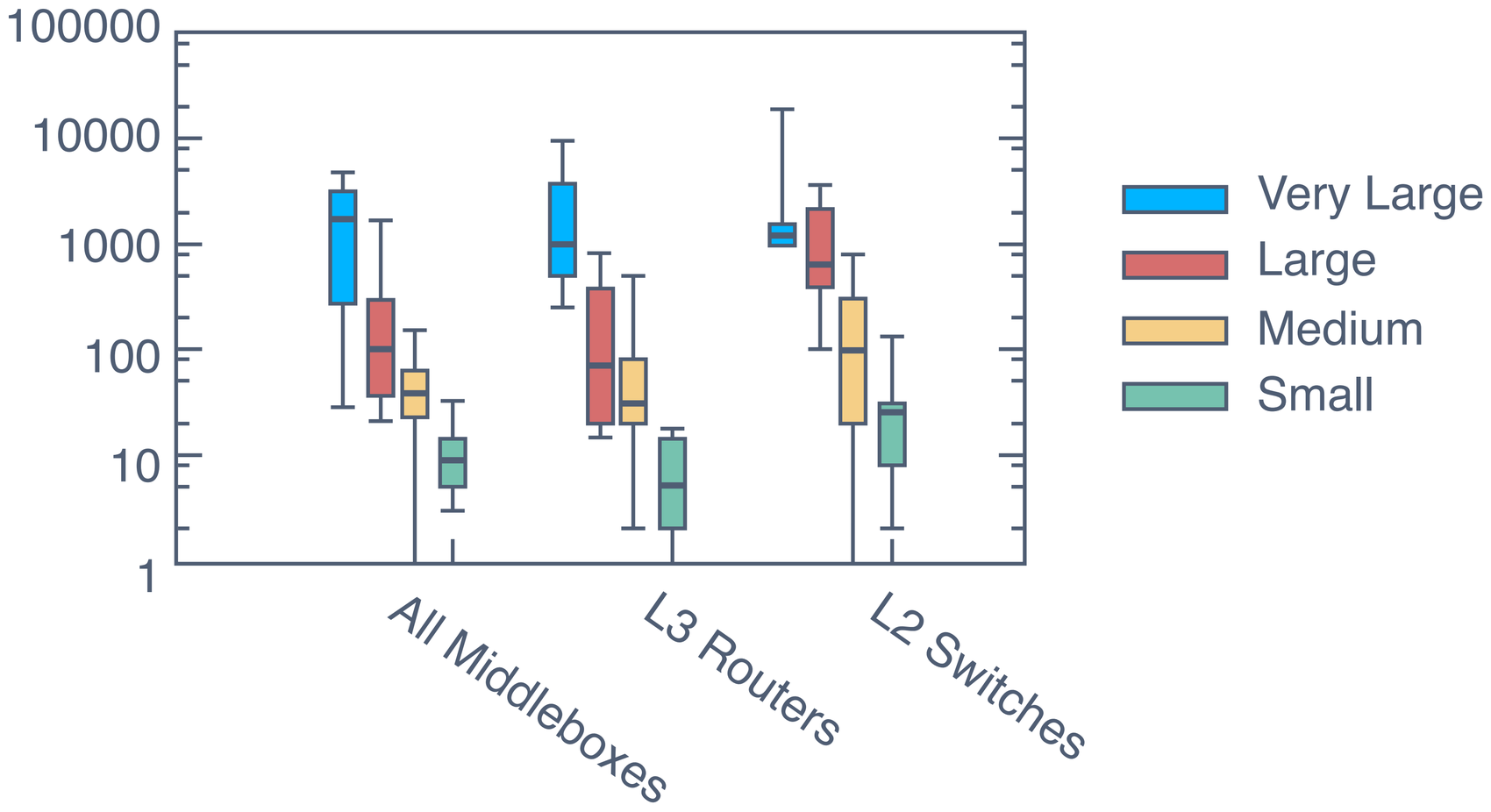
(2) A centralized point of control

- Network administrators want to *enforce policies* over how their networks are used.
  - “No one can host a botnet from within my network”: deploy and IDS
  - “All traffic is cached and compressed to save company $$ on bandwidth.”: deploy a WAN Optimizer

(2) A centralized point of control

- Network administrators want to *enforce policies* over how their networks are used, continued.
  - Note that some of these features *could be implemented by end users!* E2E would say to implement at the edge!
  - But network administrators cannot enforce what happens on end hosts: only what happens in the network.
    - Hence, middleboxes.

So, in practice, we're here:



The rest of this lecture

- The End to End Argument
- **Why we deploy middleboxes anyway**
- Some challenges they leave us with
- A new movement called Network Functions Virtualization

The rest of this lecture

- The End to End Argument
- Why we deploy middleboxes anyway
- **Some challenges they leave us with**
- A new movement called Network Functions Virtualization

Three practical challenges

- (1) Tussle
- (2) Compatibility
- (3) Complexity, Cost, and Management

Tussle

- Basically: ISPs install middleboxes and users don't always want them.
- One pressing example: **censorship**
- Middleboxes are used to filter content in many parts of the world
  - Users install VPNs or use tunnels to route through filters
    - ISPs detect VPNs and block those too…
      - Users make VPNs look like benign traffic…
        - The back and forth between users and providers is called "Tussle"

Tussle

- Other Tussles:
  - ISPs ban home users from hosting web servers
    - Users run servers over a port other than port 80
  - ISPs rate-limit BitTorrent traffic
    - BitTorrent uses “camouflaged” port numbers to make it harder to detect/classify.

Compatibility

- Middleboxes make assumptions about how protocols work. What happens when protocols change or new protocols are deployed?
  - Need to upgrade the middlebox. But many don’t.

Compatibility

- Cool story from a colleague at Google:
  - Google was testing the new QUIC protocol
  - They changed how they were using some header fields in QUIC
  - Deployed the new version of QUIC to Chrome
  - Large fractions of the Internet stopped being able to use QUIC!
  - The problem? A major middlebox vendor saw the changed ports, determined the traffic was non-standard and maybe dangerous. Blocked the traffic.

Manageability, Cost, and Complexity

- Middleboxes are custom, hardware-based devices.
  - Slow to upgrade, and expensive — $10ks
- Have to be physically wired together and configured one-by-one
  - Time consuming and confusing
- Every device has its own management interface and toolchain!

The rest of this lecture

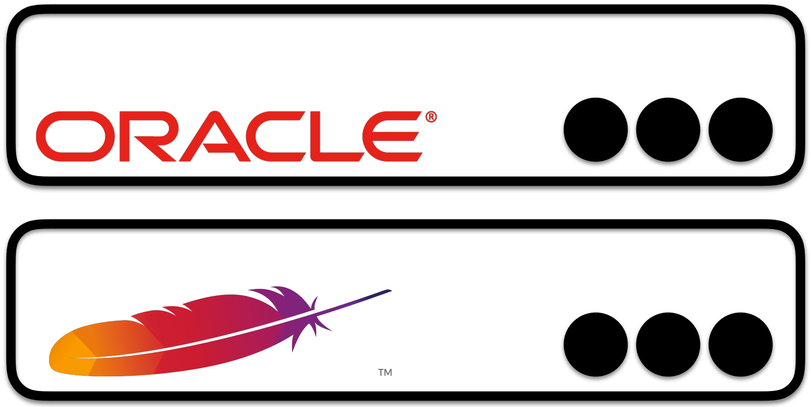
- The End to End Argument
- Why we deploy middleboxes anyway
- Some challenges they leave us with
- **A new movement called Network Functions Virtualization**
  - **(MY RESEARCH)**

Imagine cloud computing if it were deployed like middleboxes.

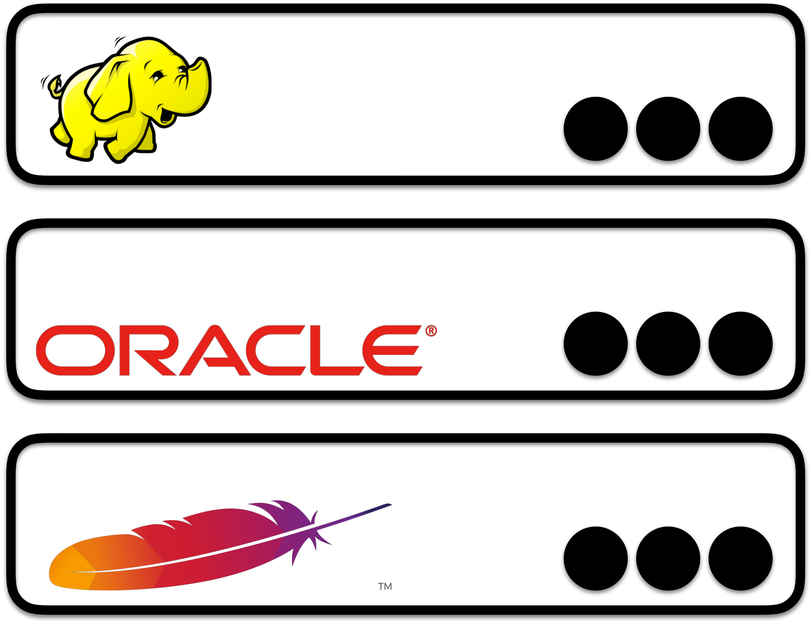
So you want to deploy a web service.

Imagine cloud computing if it were deployed like middleboxes.

So you want to deploy a web service.

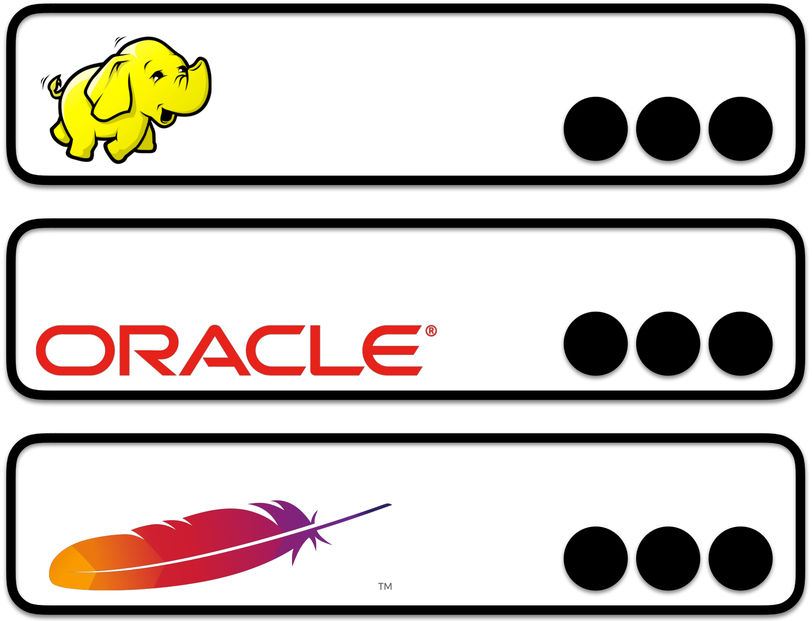


Imagine cloud computing if it were deployed like middleboxes.

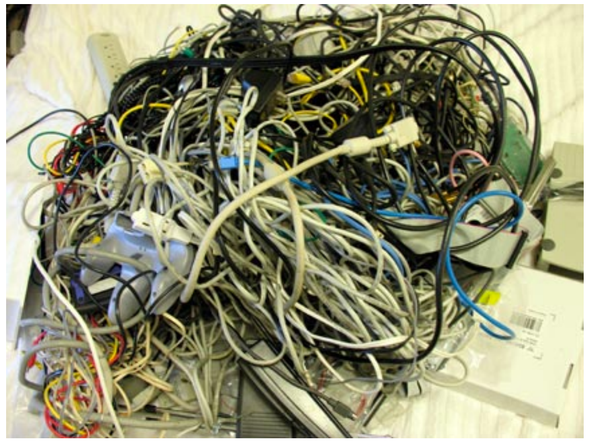


So you want to deploy a web service.

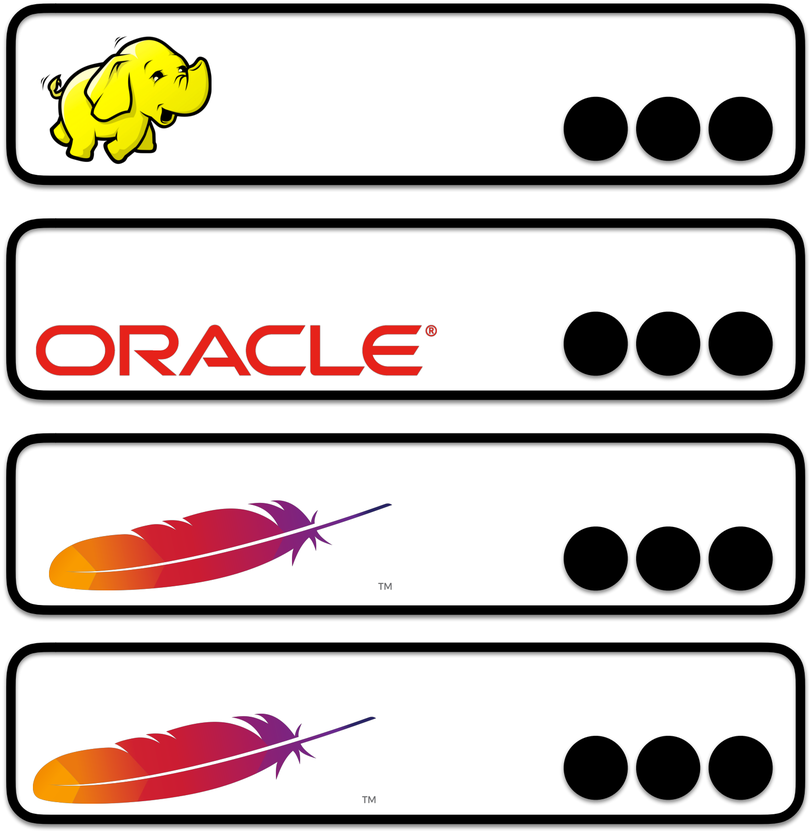
Imagine cloud computing if it were deployed like middleboxes.



So you want to deploy a web service.



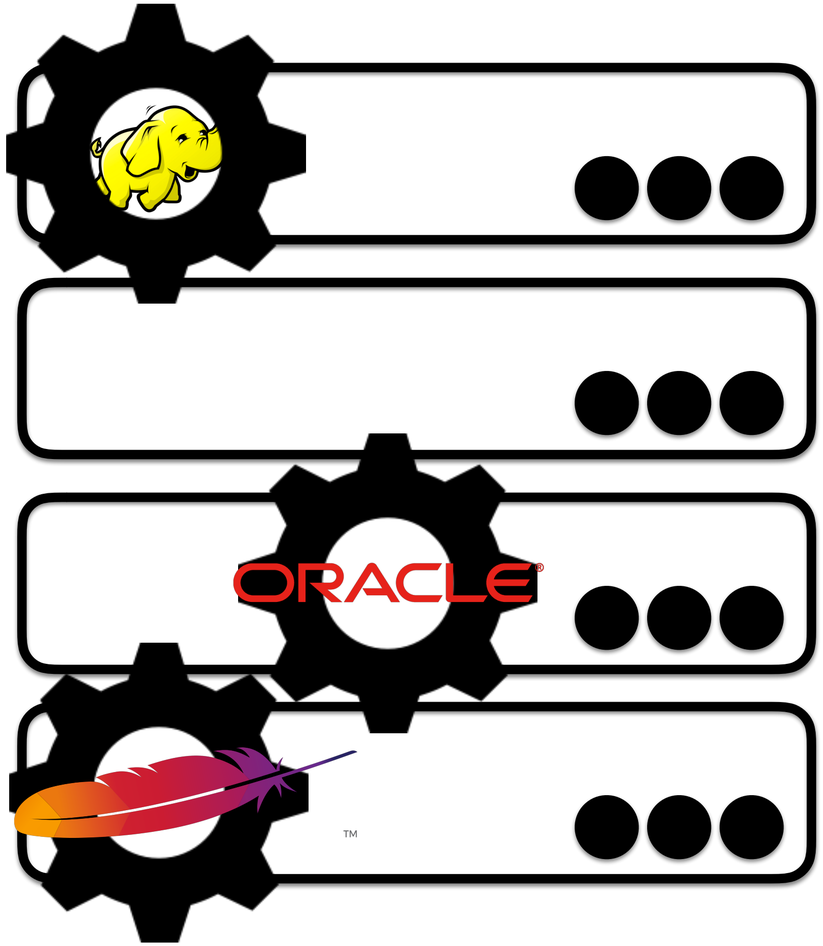
Imagine cloud computing if it were deployed like middleboxes.



So you want to deploy a web service.

This is ridiculous and not what anybody does for cloud services. But it’s what we were doing with middleboxes!

What we actually do in cloud computing.



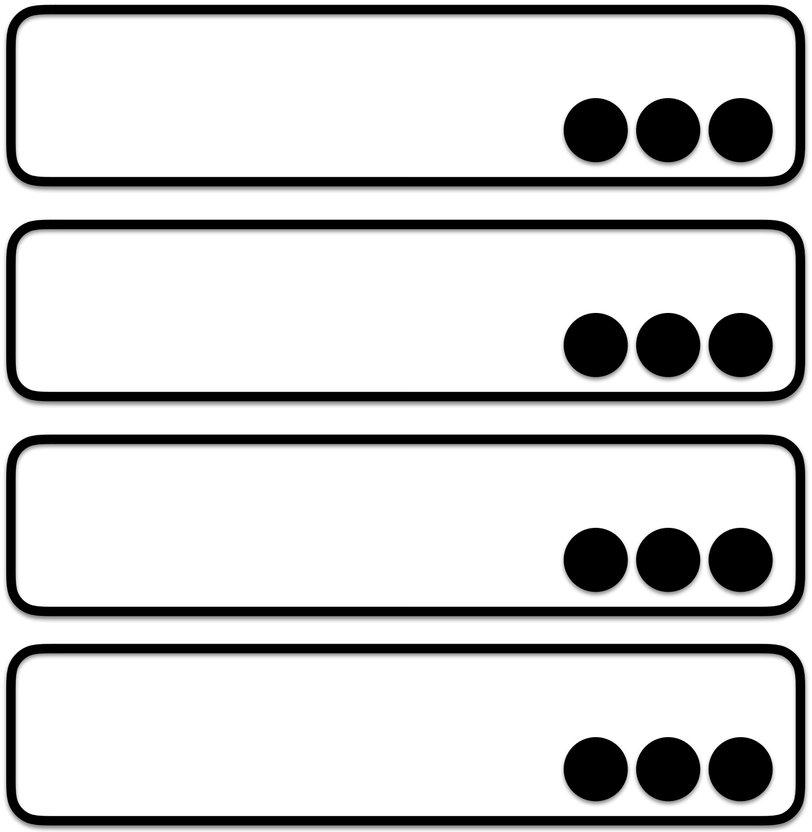
General-purpose hardware.

Services run in software.

Installation is a “click” — no cabling required.

Can re-use infrastructure for different tasks.

2012: ETSI Network Functions Virtualization



Network traffic routed through general-purpose hardware.

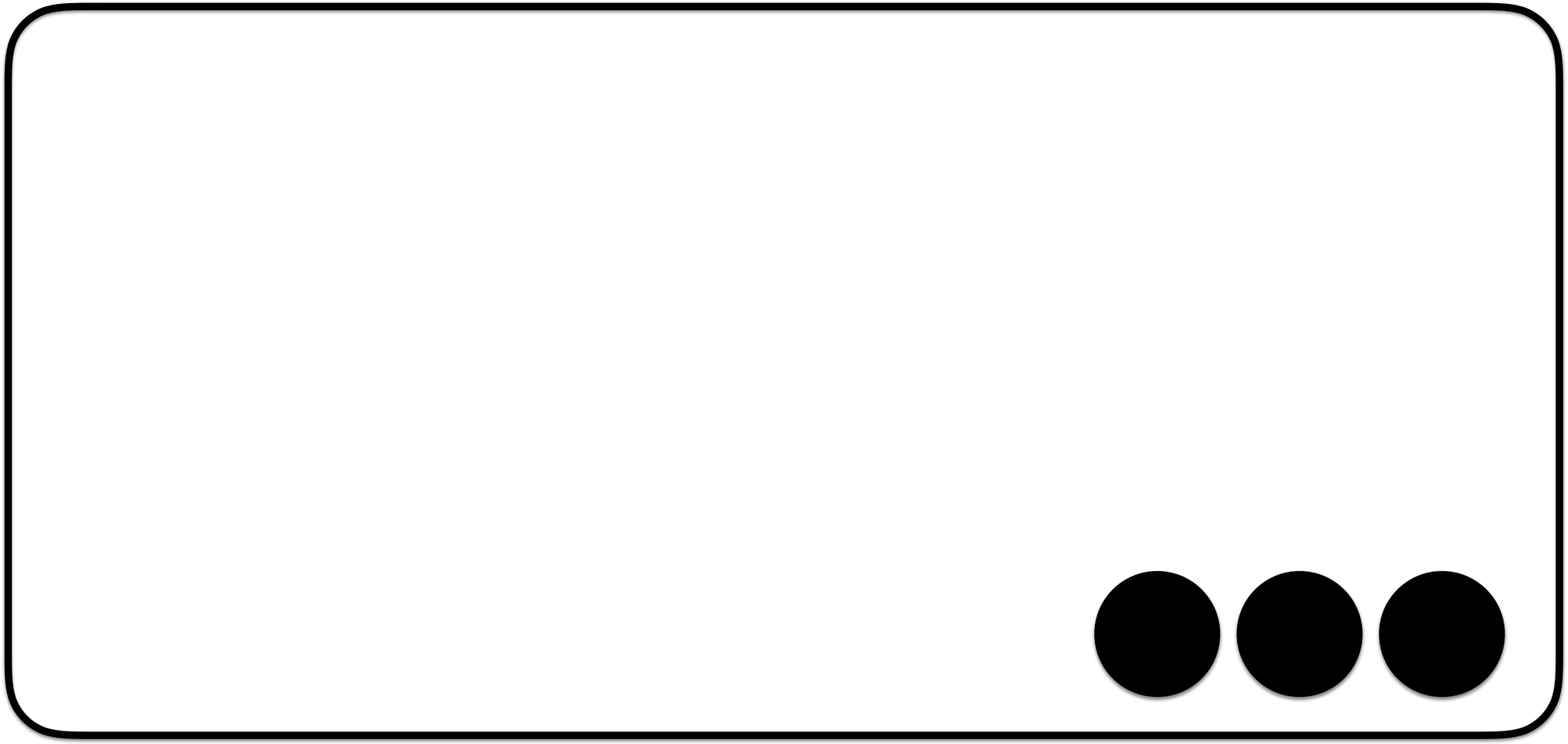
"Network Functions"

Benefits of NFV

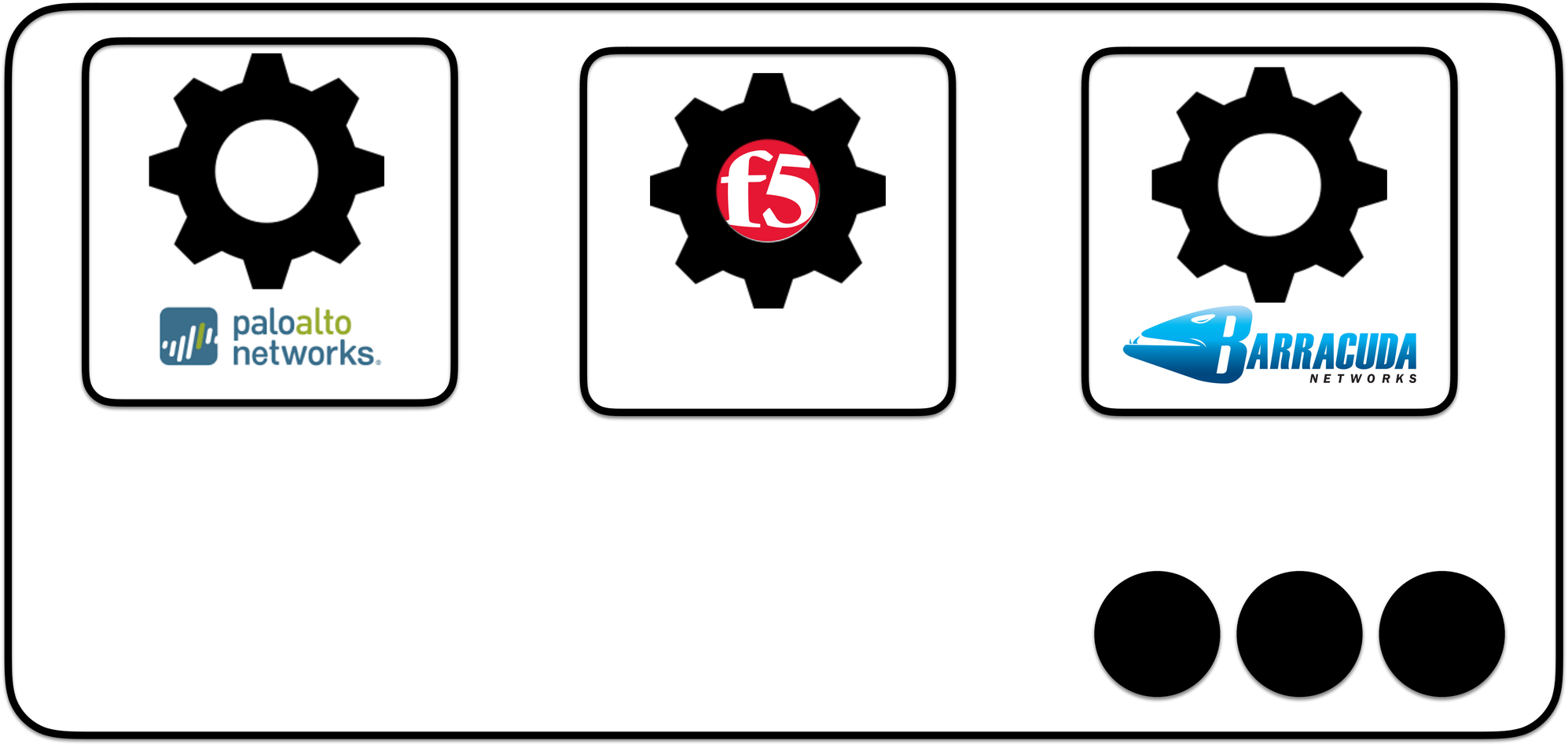
- Re-use hardware resources for many different applications
- “Scale on demand” as load changes
- Easier and more generic management tools
- Fast to upgrade and change software deployments
- Generic hardware usually -> cheaper, too!

Rough NFV System Architecture

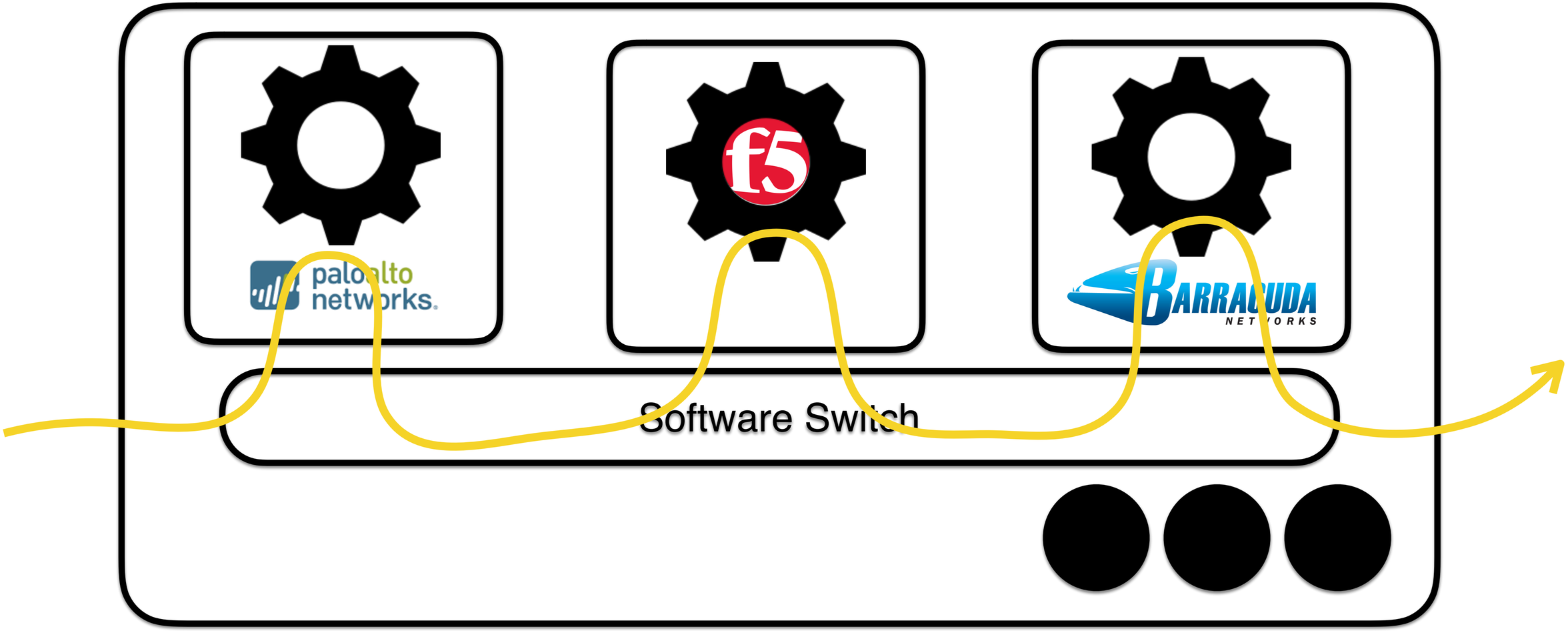
Rough NFV System Architecture



Rough NFV System Architecture



Rough NFV System Architecture



NFV is a big trend in industry right now!

NFV standardization body

Startup I worked at last year

Open Source project to develop NFV platform

Middleboxes: Summary

- Middleboxes are the de-facto way to insert new functionality into networks.
- Very widely deployed: 1/3 network devices is a middlebox
- Challenging to manage (upgrades, compatibility, complexity) and at times controversial (tussle).
- NFV is a new movement to build middleboxes in software using lessons from cloud computing.