Network/Link Forwarding; Course Summary; Alternative Designs

Y. Richard Yang

http://zoo.cs.yale.edu/classes/cs433/

12/7/2018

Outline

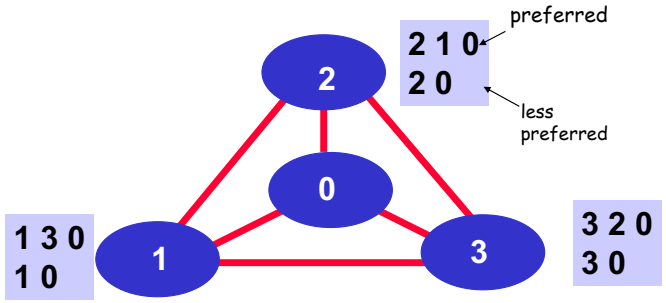
- Admin and recap
- Network/link-layer forwarding
- Course summary

Admin

- Assignments 3 and 4 grades to be released by Friday
- Exam 2: 7:00-8:30 pm Dec. 11 at AKW 200 and AKW 300
- Remaining instructor office hours:
  - Thursday (today): 2:30 – 3:30 pm
  - Friday (Dec. 7): 1:00 – 2:00 pm
  - Sunday (Dec. 9): 3:00 – 4:00 pm
  - Tuesday (Dec. 11): 2:30-3:30 pm

Recap: Policy Routing

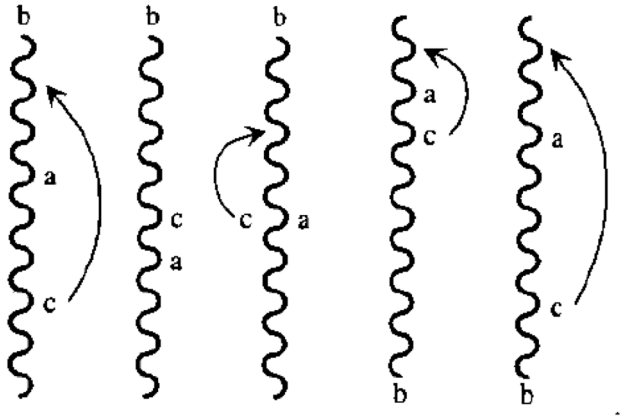
- Policy routing allows autonomy but can lead to instability



- General aggregation has limitations: Arrow's Theorem if aggregation satisfies transitivity, unanimity and IIA

The Extremal Lemma

- Let choice b be chosen arbitrarily. Assume that every voter puts b at the very top or the very bottom of his ranking. Then constitution must as well (even if half voters put b at the top and half at the bottom)
- Proof: by contradiction.
  - Assume there exist a and c such that constitution has a >= b; b >= c.
  - We can move c above a w/o changing a-b or b-c votes



Step 1: Existence of Pivotal Voter

- Let choice b be chosen arbitrarily. There exists a voter n* = n(b) who is extremely pivotal for b in the sense that by changing his vote at some profile, he can move b from the very bottom to the very top in the social ranking.

- Proof:
  - Consider an extreme profile where b is at the bottom of each voter.
  - Consider voter from 1 to n, and we move b from bottom to top one-by-one.
  - The first voter whose change causes b to move to the top is n*

Step 2: n*=n(b) is dictator of any pair ac not involving b

- Proof
  - Consider a from ac pair. We show that if a $>_{n^*}$ c, then society has a > c
  - Let profile before n* moves b to top as profile I
  - Let profile after n* moves b to top as profile II
  - Construct profile III from II by letting n* move a above b; all others can arrange ac as they want, but leave b in extreme position

Profile I

b	b	.	.	.
.	.	.	.	.
.	.	.	.	.
.	.	b	b	b

constitution: b bottom

Profile II

b	b	b	.	.
.	.	.	.	.
.	.	.	.	.
.	.	.	b	b

constitution : b top

1	2	n*	.	N

Profile III

b	b	a	.	.
.	.	b	.	.
.	.	.	.	.
.	.	c	b	b

constitution : a > b since ab same as I

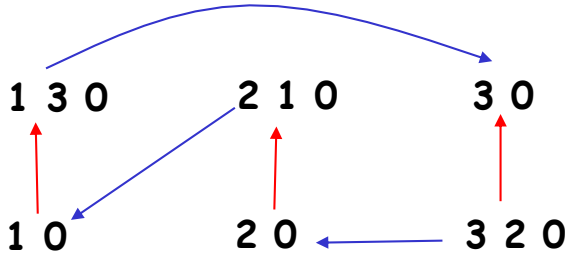
b > c since bc same as II

Step 3: n* is dictator for every pair ab

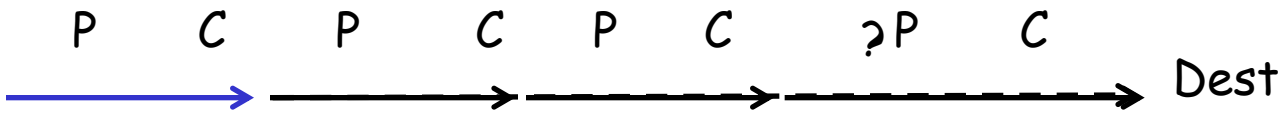
- Consider c not equal to a or b
- There exists n(c) who is a dictator of any pair not involving c, such as the pair ab, i.e.,
  - For any profile, if $a >_{n(c)} b$, $a > b$ for society
- n(c) must be n*
  - Assume not.
    - Consider Profile I and Profile II.
    - Since n(c) is not n*, n(c) ranking of ab does not change in Profile I and Profile II.
    - When n* changes ab ranking between Profile I and Profile II, the constitution ranking of ab changes.
    - Contradiction.

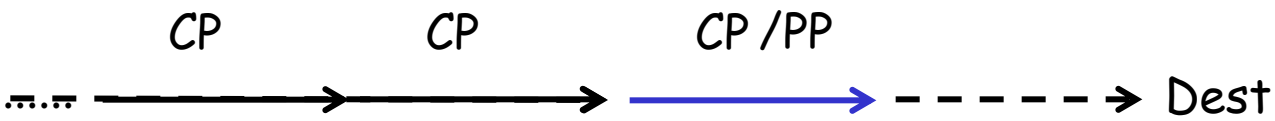
Recap: Policy Routing

- Local aggregation has tools such as p-graph to help with understanding and analysis



- Current Internet economics enforces stability





Outline

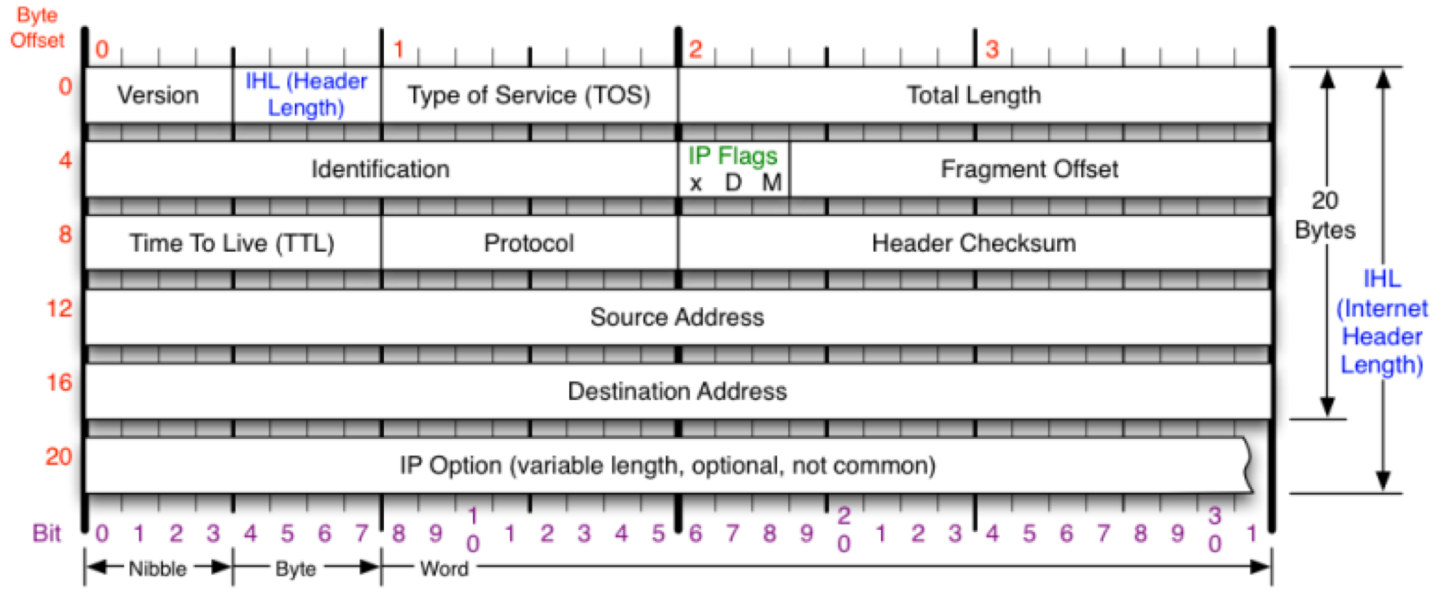
- Admin and recap
- Network and link forwarding: put it together

Network Control Plane vs Forwarding Plane

- Control plane
  - Configure devices
    - Allocate IP address (DHCP, or static)
    - Setup/compute routing table
  - Report errors (e.g., ICMP)
  - Not per packet
    - hence called slow path
- Forwarding plane
  - Forward packet to destination
  - Handle each data packet
    - hence called fast path

Discussion: Basic service API of network forwarding?

Exercise: Actions at a Router



Version

Version of IP Protocol. 4 and 6 are valid. This diagram represents version 4 structure only.

Header Length

Number of 32-bit words in TCP header, minimum value of 5. Multiply by 4 to get byte count.

Protocol

IP Protocol ID. Including (but not limited to):

1 ICMP	17 UDP	57 SKIP
2 IGMP	47 GRE	88 EIGRP
6 TCP	50 ESP	89 OSPF
9 IGRP	51 AH	115 L2TP

Total Length

Total length of IP datagram, or IP fragment if fragmented. Measured in Bytes.

Fragment Offset

Fragment offset from start of IP datagram. Measured in 8 byte (2 words, 64 bits) increments. If IP datagram is fragmented, fragment size (Total Length) must be a multiple of 8 bytes.

Header Checksum

Checksum of entire IP header

IP Flags

x D M

x 0x80 reserved (evil bit)
D 0x40 Do Not Fragment
M 0x20 More Fragments follow

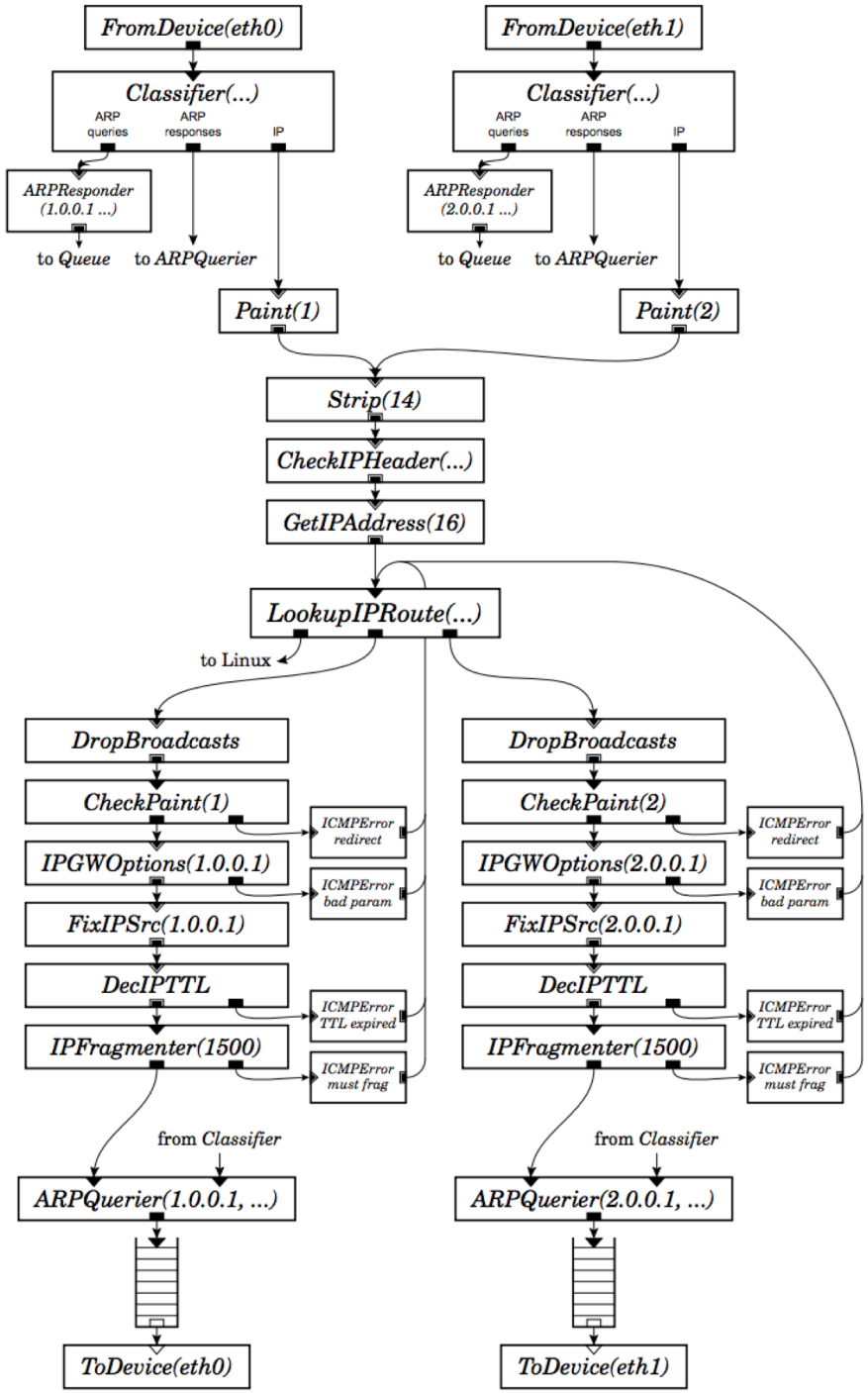
RFC 791

Please refer to RFC 791 for the complete Internet Protocol (IP) Specification.

Source: https://nmap.org/book/tcpip-ref.html

Router Actions

- Routing
  - Look up
  - Prevent loops (TTL--)
  - Notify host better routers (ICMP redirect)
- Handle IPGW options
  - (e.g., record routes)
- Handle fragmentation
- Error checking and reporting using ICMP
- Scheduling (e.g., linux tc)



Network Forwarding Plane Key Data Structure

- Routing Table (also called Forwarding Information Base, FIB)

- Exercise:
  - Zoo FIB
    - netstat -rn
  - Mac FIB
    - netstat -rn

End Host FIB

Zoo:

```
[yry3@cicada ~]$ netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         128.36.232.1    0.0.0.0         UG        0 0          0 enp0s25
128.36.232.0    0.0.0.0         255.255.255.0   U         0 0          0 enp0s25
```

MAC OS:

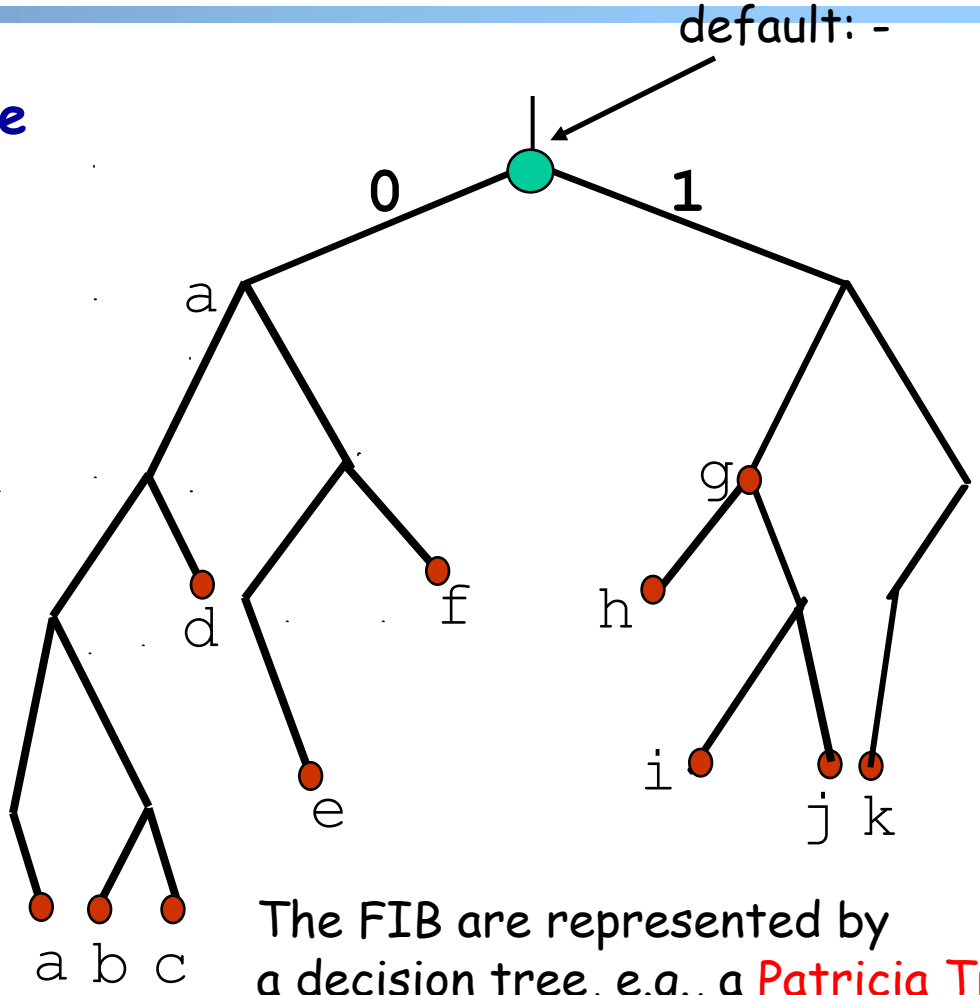
```
Routing tables

Internet:
Destination        Gateway            Flags        Refs      Use   Netif Expire
default            172.27.16.1        UGSc         1470        0     en0
127                127.0.0.1          UCS             1        0     lo0
127.0.0.1          127.0.0.1          UH              4     3788     lo0
169.254            link#4             UCS           106        0     en0
169.254.1.229      link#4             UHLSW           1        0     en0
169.254.5.209      f0:99:bf:1e:6f:de  UHLSW           1        0     en0    989
169.254.8.254      link#4             UHLSW           1        0     en0
169.254.11.96      0:cd:fe:75:59:75   UHLSW           1        0     en0   1009
169.254.13.89      64:9a:be:af:34:53  UHLSW           1        0     en0   1145
169.254.16.49      link#4             UHLSW           1        0     en0
169.254.19.58      link#4             UHLSW           1        0     en0
169.254.19.82      link#4             UHLSW           1        0     en0
169.254.21.198     link#4             UHLSW           1        0     en0
169.254.22.67      0:23:12:12:bc:39   UHLSW           1        0     en0     31
169.254.23.4       link#4             UHLSW           1        0     en0
...
```

Discussion

- How is the FIB of a device computed?
- What might be a reasonable data structure to look up an FIB efficiently?

FIB Look Up: Software

#	prefix	interface
a)	00001	
b)	00010	
c)	00011	
d)	001	
e)	0101	
f)	011	
g)	10	
h)	100	
i)	1010	
j)	1011	
k)	1100	

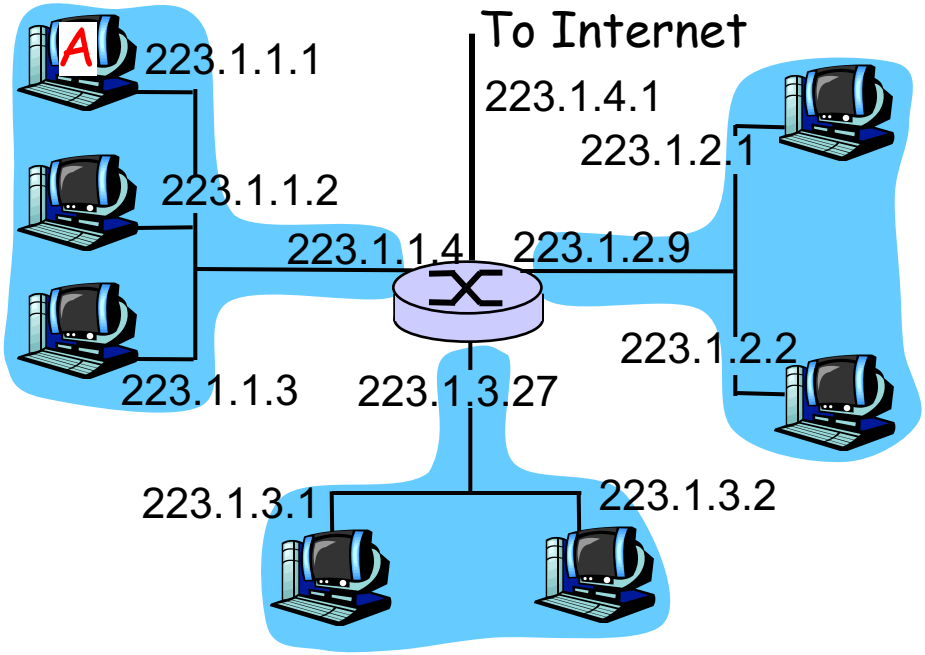


The FIB are represented by a decision tree, e.g., a Patricia Trie to look for the longest match of the destination address

Forwarding: Example 1

	src	dst	
misc fields	223.1.1.1	223.1.1.3	data

- Setting: Host A network layer receives a packet from above (transport layer).
- Action:
  - Host A looks up destination in FIB



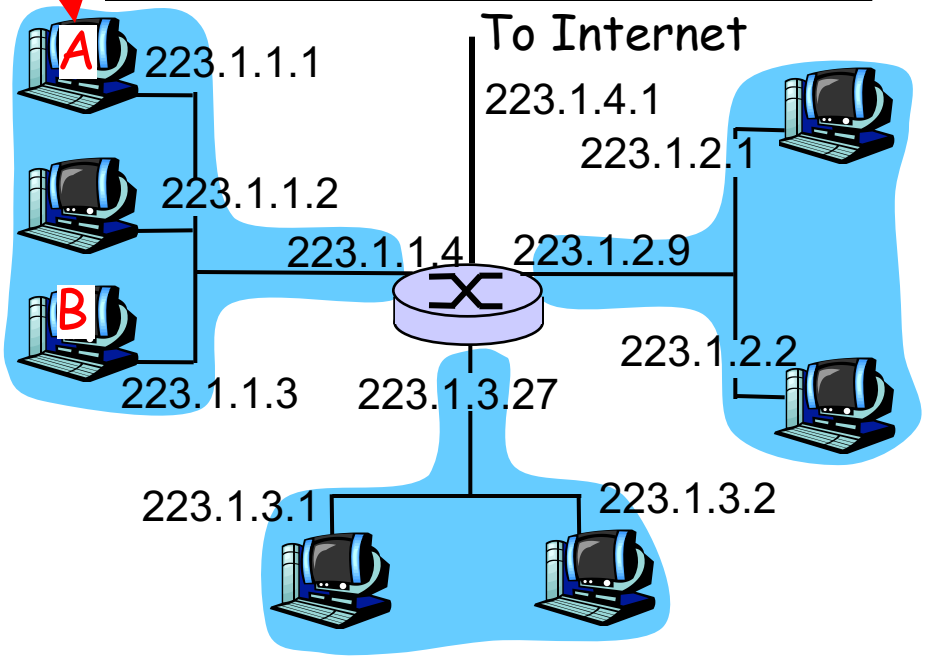
Forwarding: Example 1

misc fields	src 223.1.1.1	dst 223.1.1.3	data

- Setting: Host A network layer receives a packet from above (transport layer).
- Action:
  - Host A looks up destination in FIB
  - FIB best match is same subnet (no next router or 0.0.0.0 shown in Linux)
    - Hand packet to link layer to send inside a link-layer frame

FIB of A

Dest. Net.	next router	Nhops
223.1.1/24	0.0.0.0	1
223.1.2/24	223.1.1.4	2
223.1.3/24	223.1.1.4	2
0.0.0.0/0	223.1.1.4	-



Discussion

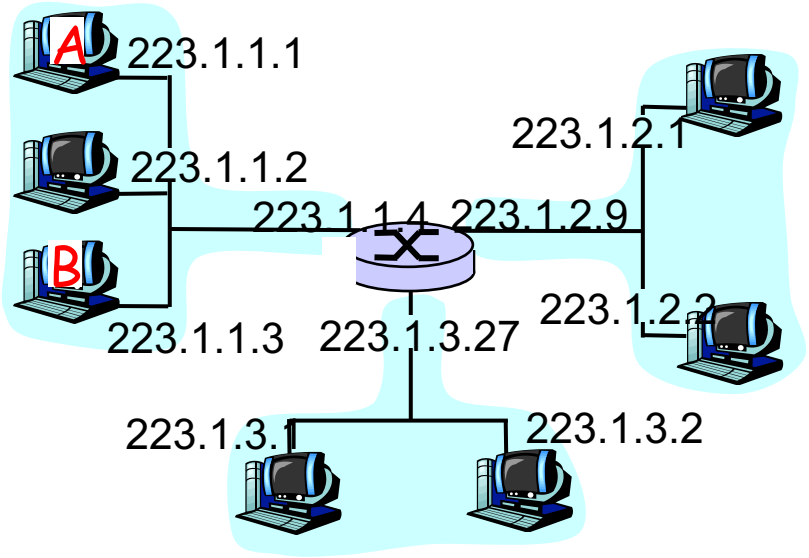
- What may the basic link-layer service API look like?

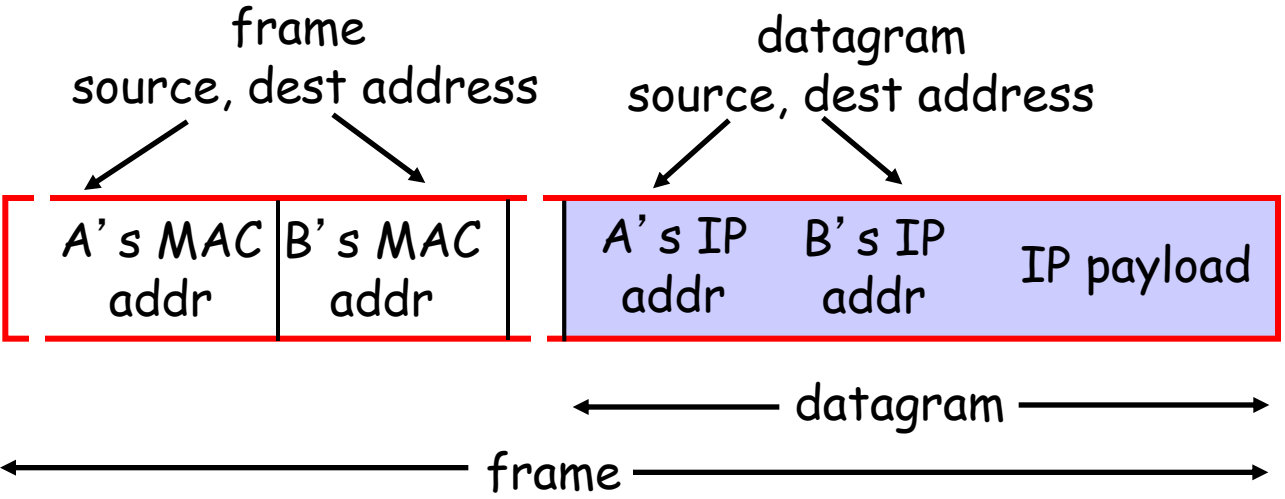
Comparison of Network (IP) address and Link Address

- Network-layer (IP) address supports global routing
  - be globally unique (if no NAT)
  - has features to improve routing system scalability
    - allows address aggregation (CIDR)/hierarchy
    - indicates where an host/a set of aggregated hosts is attached on the global Internet graph
    - bound to location, called locator
- Link-layer (MAC) supports local forwarding
  - does not need to be globally unique, locally unique is enough, but an assignment may ensure uniqueness
  - typically no need for address aggregation
  - location independent, flat address
  - bound to device, called identifier

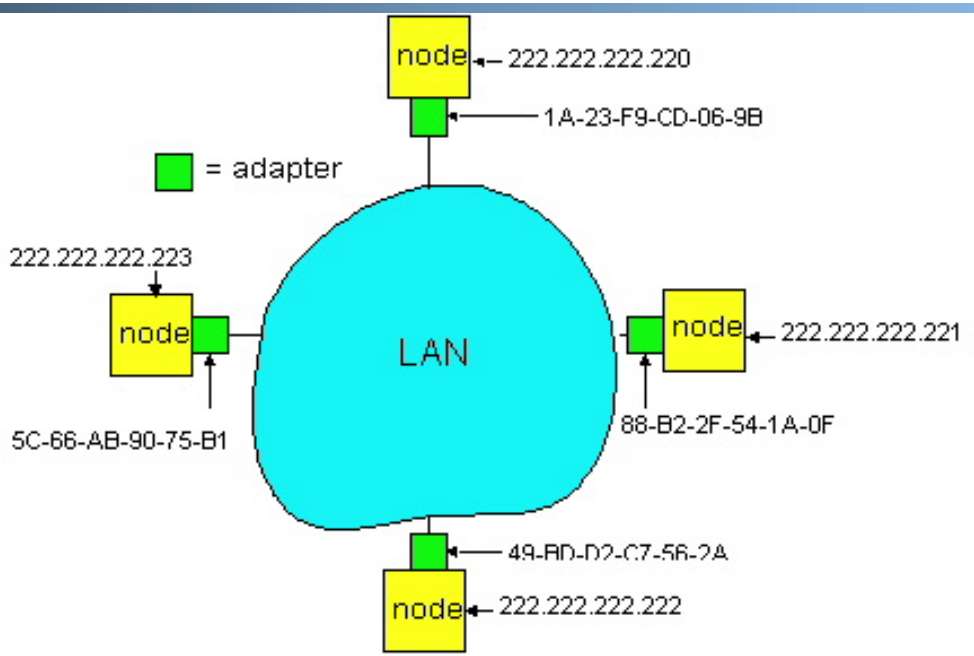
Network Address -> Link Address

Host A needs to find the link-layer (MAC) address of B to construct the link layer frame (which protocol?):





Recall: Address Resolution Table and Address Resolution Protocol (ARP)



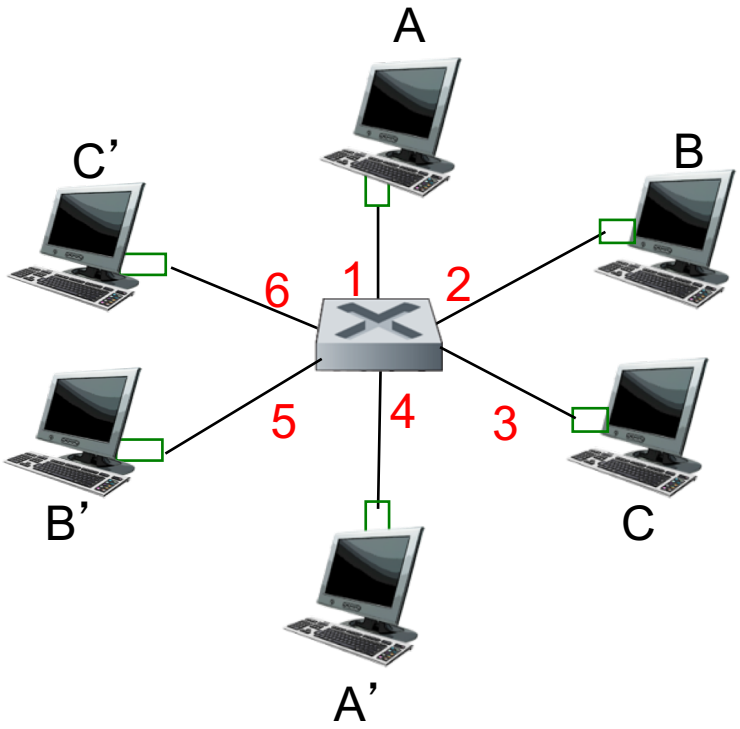
- Each IP node (Host, Router) on LAN has ARP table
- ARP Table: IP/MAC address mappings for some LAN nodes

  < IP address; MAC address; TTL>
  - TTL (Time To Live): time after which address mapping will be forgotten (typically 20 min)
- ARP table built by the ARP protocol
  - A plug-and-play protocol
  - A broadcast protocol

```
[yry3@cicada yry3]$ /sbin/arp
Address                  HWtype  HWaddress           Flags Mask            Iface
zoo-gatew.cs.yale.edu    ether   AA:00:04:00:20:D4   C                     eth0
artemis.zoo.cs.yale.edu  ether   00:06:5B:3F:6E:21   C                     eth0
lab.zoo.cs.yale.edu      ether   00:B0:D0:F3:C7:A5   C                     eth0
```

Modern Ethernet and Naïve Link Forwarding

- Modern wired Ethernet networks build link layer using non-broadcast (switched) Ethernet switches
- When switch S receives ARP query from A on B's MAC
  - switch S forwards query to all other interfaces
- When switch S receives ARP reply from B
  - switch forwards reply to all other interfaces
- When switch S receives data frame from A to B
  - switch S forwards data to all other interfaces

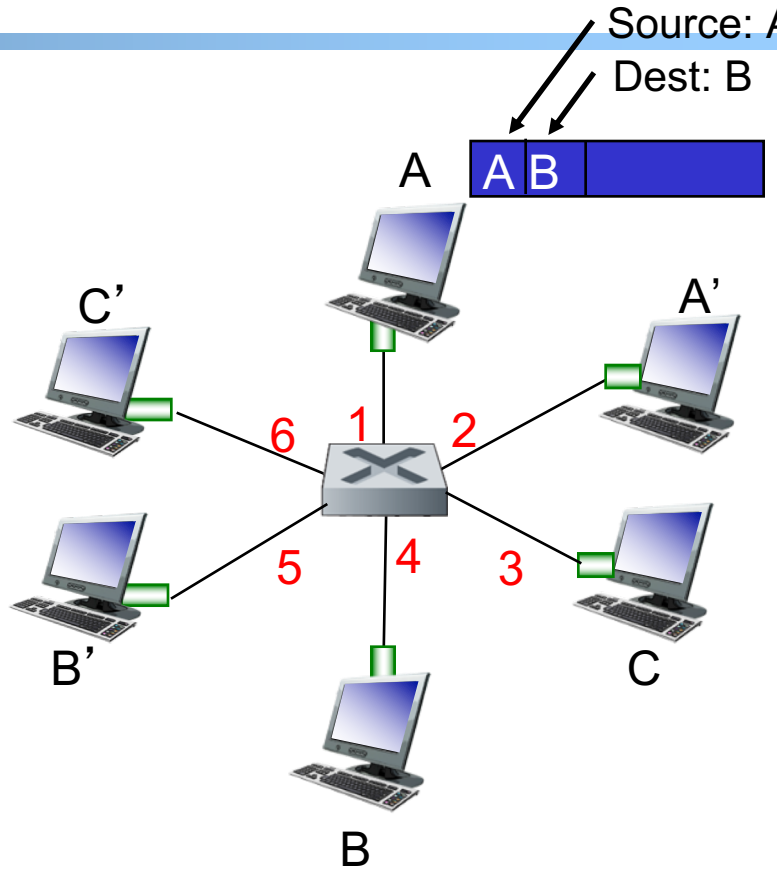


switch with six interfaces (1,2,3,4,5,6)

Q: Can we do better but still achieve plug-and-play?

Link Forwarding Key Idea: Self Learning

- Link-layer switch *learns* which hosts can be reached through which interfaces through bypassing frames
  - upon receiving a frame, switch "learns" location of sender: incoming interface
  - records sender/location pair in switch table



MAC addr	interface	TTL
A	*1*	*60*

Switch table (initially empty)

Modern Link-Layer Forwarding: Self-Learning + Flooding

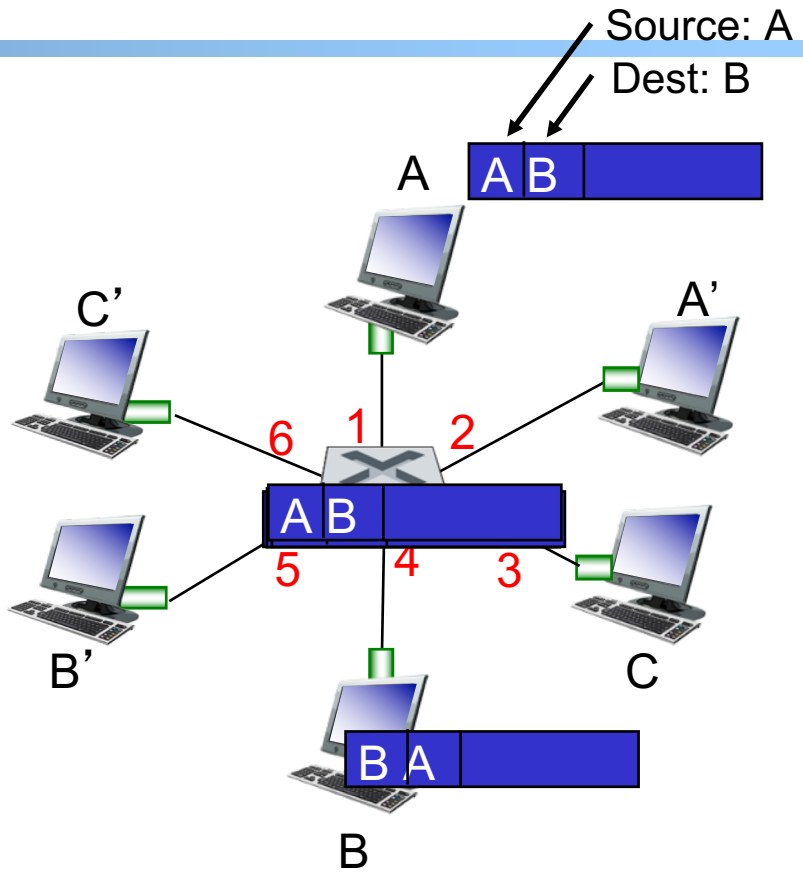
when frame received at link-layer switch:

1. record incoming link, MAC address of sending host
2. index switch table using MAC destination address
3. if (entry found for destination) {

```
    if (destination on interface from which frame arrived)
        drop frame
    else
        forward frame on interface indicated by entry
}
else
    flood  /* forward on all interfaces except
              arriving interface */
```

Example: Flooding, Learning for Data Frames

- frame destination, B, location unknown: *flood*
- destination A location known: *selectively send on just one link*

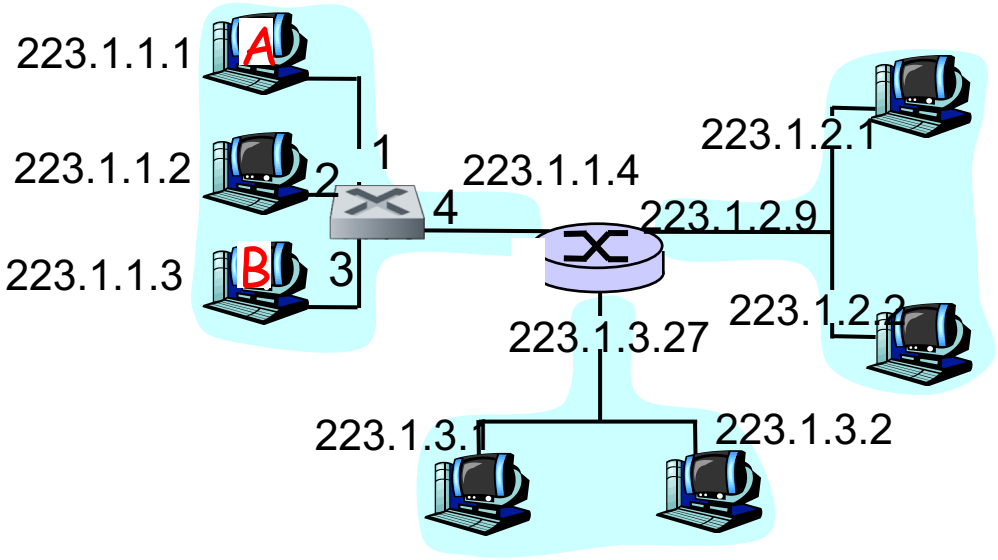


MAC addr	interface	TTL
A	*1*	*60*
B	*4*	*60*

switch table (initially empty)

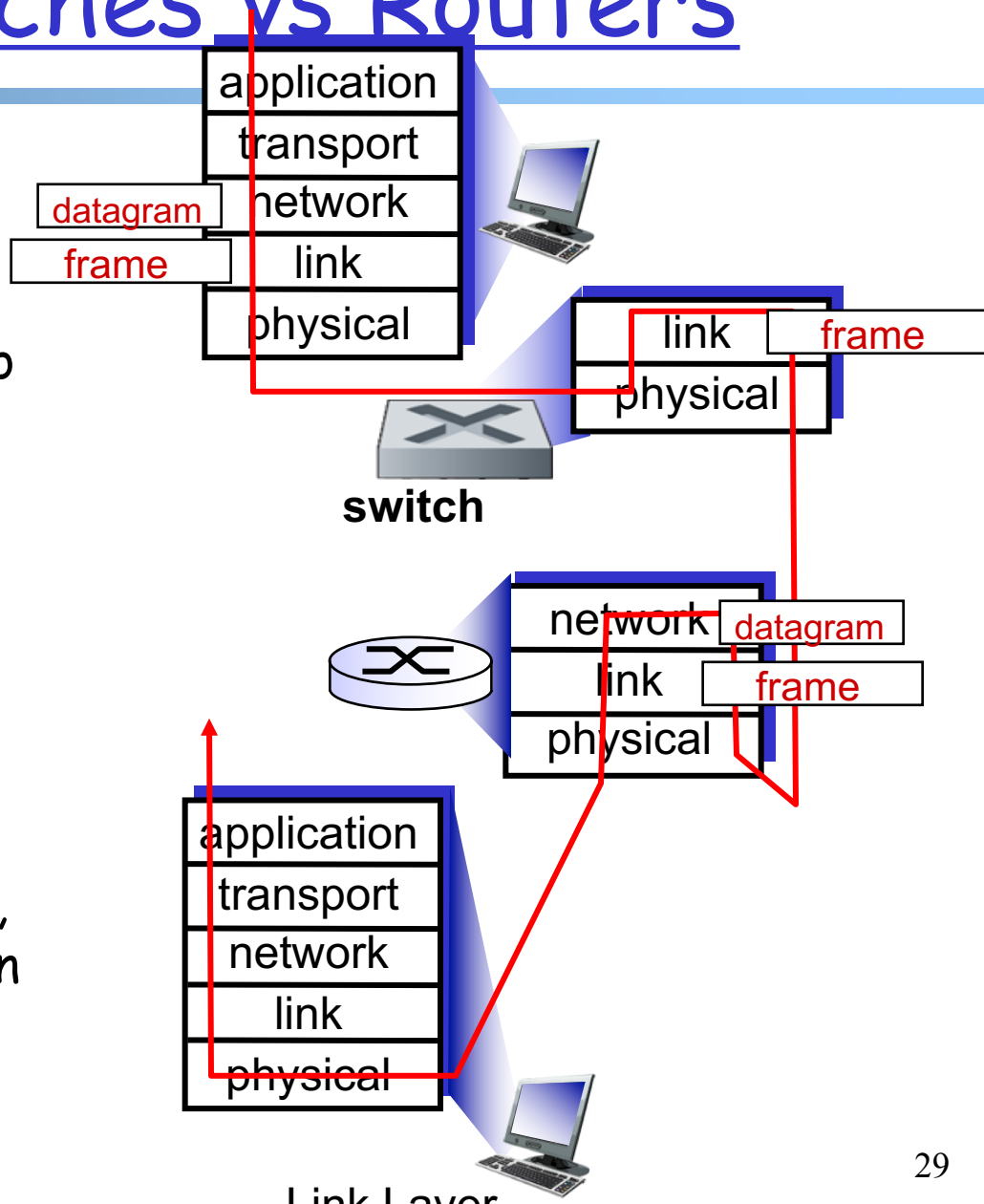
Exercise

- Host A -> Host B Flooding + Learning from initial ARP to actual A-> B data



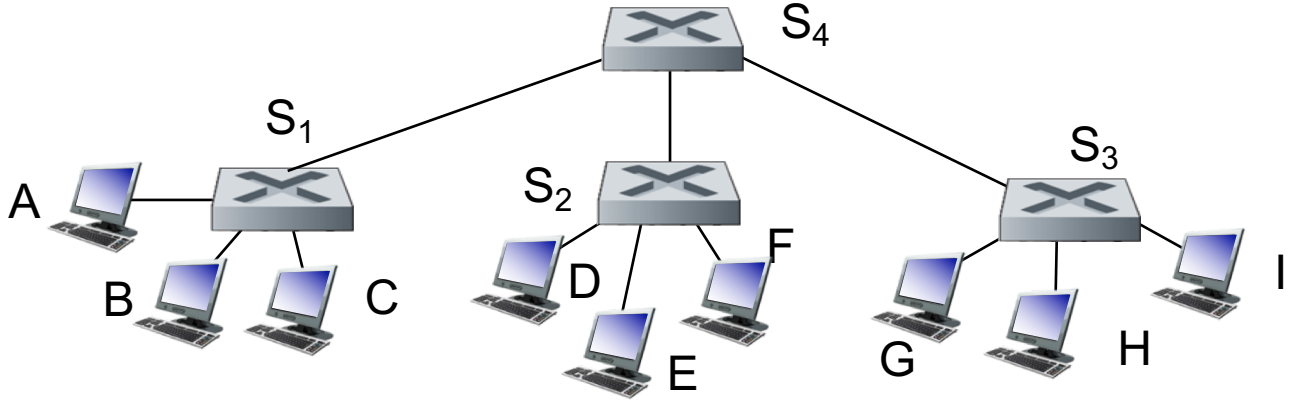
Summary: Switches vs Routers

- Common
  - Both are store and forward
  - Both have a look table to provide direction
- Differences
  - Routers examine network-layer headers while switches examine link-layer headers
  - Routers compute tables using routing algorithms, while switches depend on flooding and self-learning



Interconnecting Switches

- A layer-2 domain may need multiple switches



- Exercise: Assume A sends to G. Will the flooding+learning alg work for the above network?
- Q: For what kind of layer-2 networks will the flooding+learning alg. work?
- Q: If a network is not the right kind, what may you do?

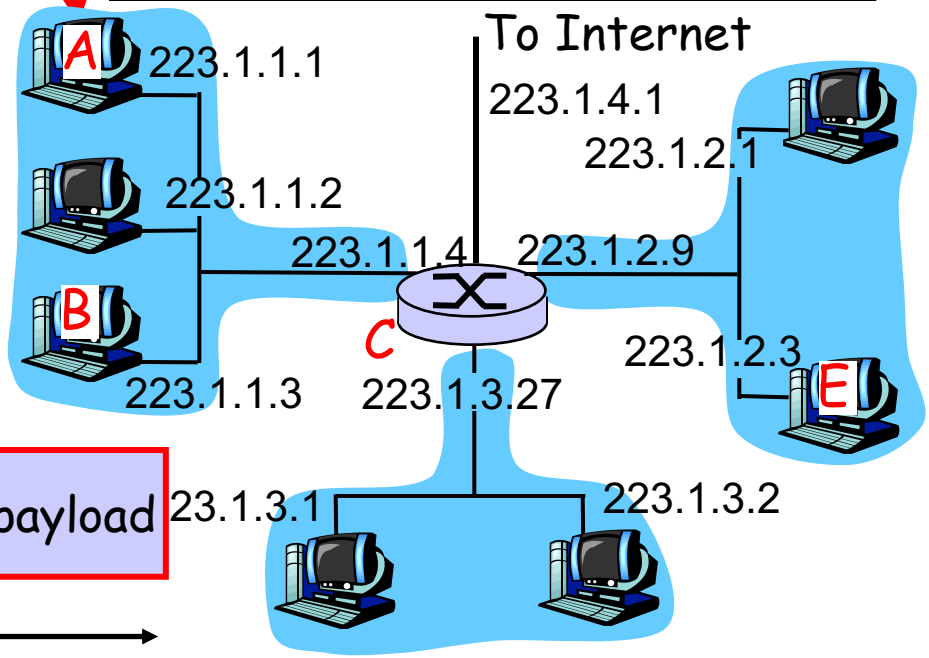
Forward Example 2 (Different Networks): A-> E

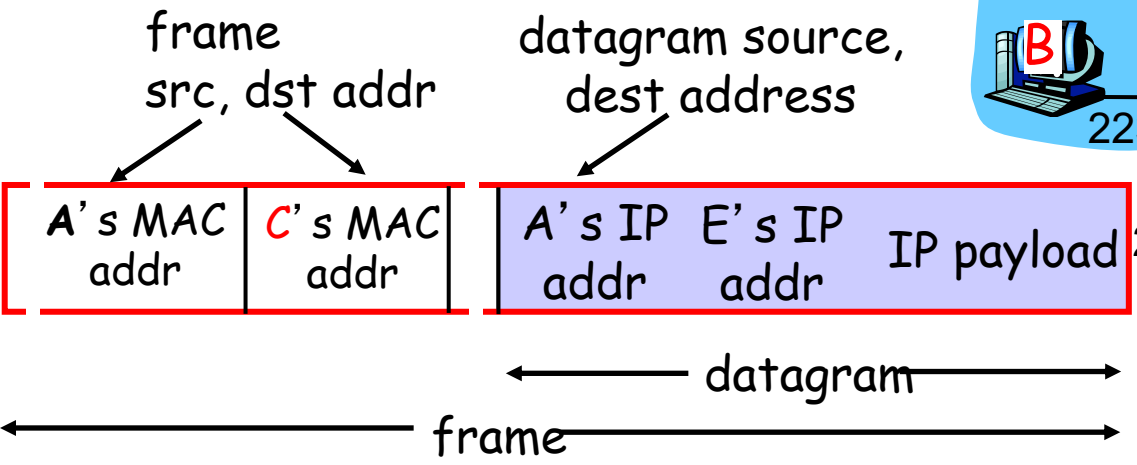
misc fields	223.1.1.1	223.1.2.3	data

- Setting: Host A network layer receives a packet above.
- Action:
  - Host A looks up destination in FIB
  - Bestmatch has next router (223.1.1.4)
    - Hand datagram to link layer to send inside a link-layer frame

FIB of A

Dest. Net.	next router	Nhops
223.1.1/24		1
223.1.2/24	223.1.1.4	2
223.1.3/24	223.1.1.4	2
0.0.0.0/0	223.1.1.4	-





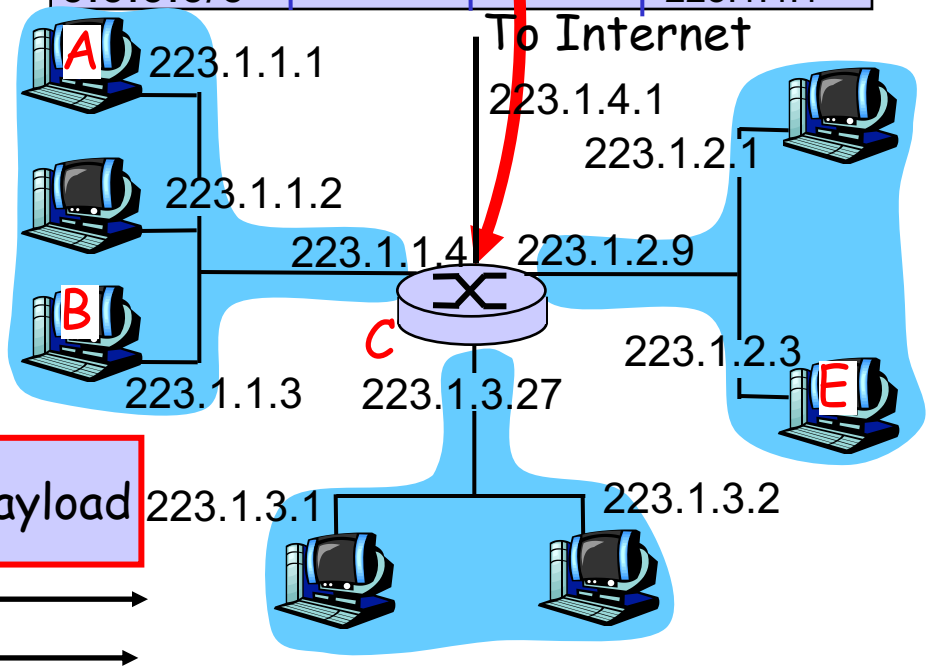
Putting it Together: Example 2 (Different Networks): A-> E

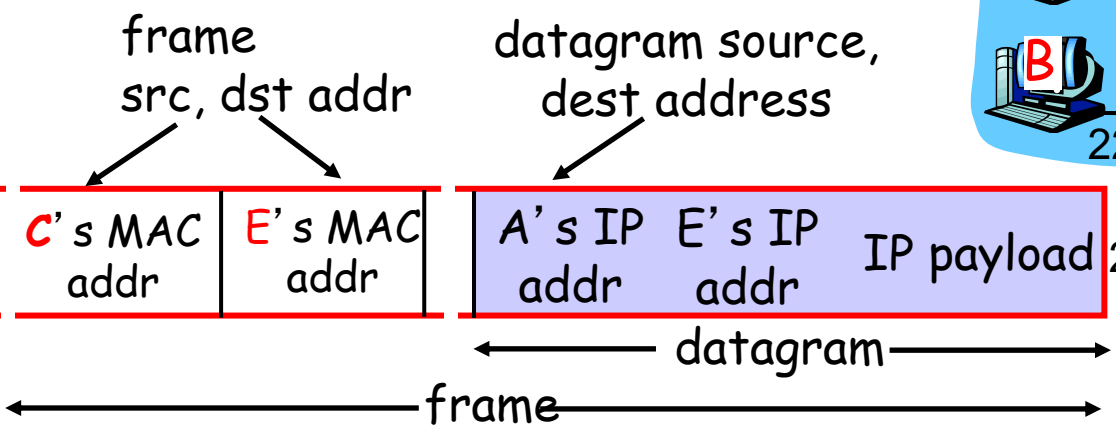
misc fields	223.1.1.1	223.1.2.3	data

- Setting: Packet above arrives at Router C's network layer.
- Action:
  - Router C looks up destination in FIB
  - Hand datagram to link layer to send inside a link-layer frame

FIB of C

Dest. Net	router	Nhops	interface
223.1.1/24	-	1	223.1.1.4
223.1.2/24	-	1	223.1.2.9
223.1.3/24	-	1	223.1.3.27
0.0.0.0/0	-	-	223.1.4.1





datagram arrives at 223.1.2.3!! (hooray!)

Summary of Network Layer

- We have covered the basics of the network layer, including both network and link layer forwarding

- There are multiple other topics that we did not cover
  - Multicast/anycast
  - QoS
  - slides as backup
    just in case you need
    reading in the break

Outline

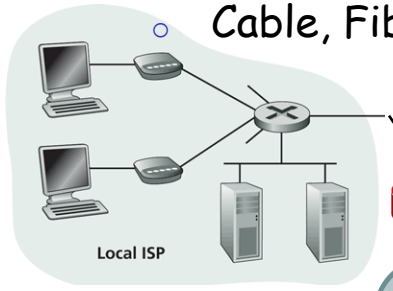
- Admin and recap
- Network/Link forwarding
- Course summary

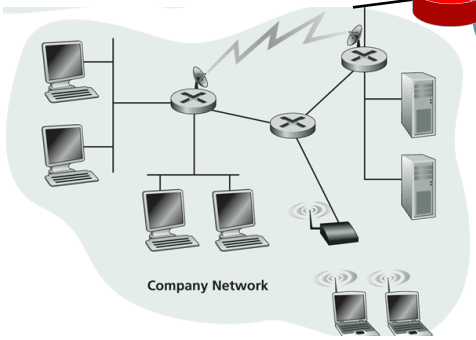
Course Topics Summary

- The Internet is a general-purpose, large-scale, distributed system providing networking services
- Major design features and underlying design/analysis principles
  - packet switching/statistical multiplexing
    - time-reversibility, queueing theory
  - layered architecture, hour-glass architecture
    - end-to-end principle, logical communications vs physical communications
  - high-performance server designs
    - multi-thread, non-blocking, operational analysis
  - distributed resource allocation
    - sliding window self clocking, AIMD adaptation, [Ethernet exp backoff]
    - axiom-based design (NBS); allocation decomposition through duality
  - decentralized (social-techno) architecture
    - DNS (hierarchy delegation), interdomain routing (peer-to-peer integration through policy routing), Arrow's Theorem
  - distributed routing systems
    - from distributed computing to state synchronization, monotonicity, local condition to enforce global condition, global invariants as analysis techniques
  - tradeoff between theoretical impossibility and practice
    - consensus impossibility and TCP close

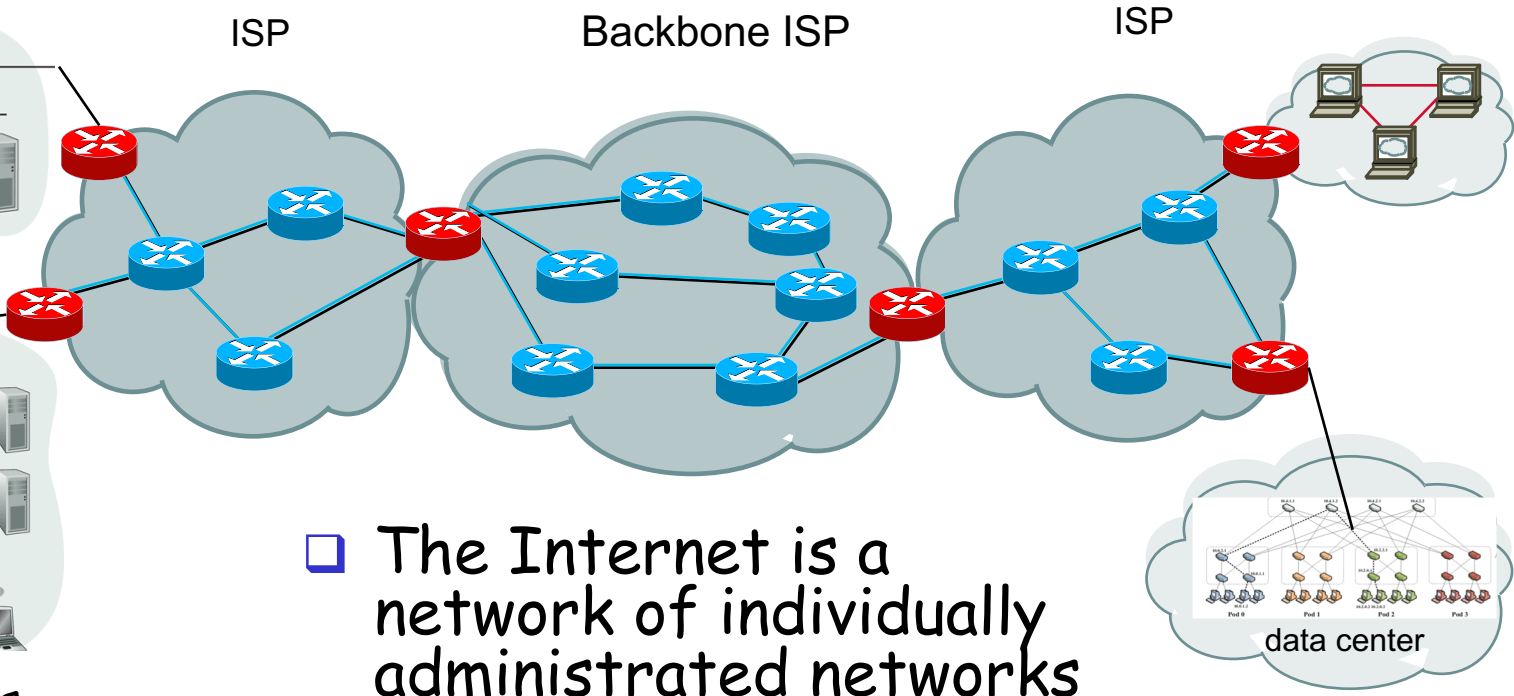
First-Day Class: Internet Physical Infrastructure

Residential access
- Cable, Fiber, DSL, Wireless





Campus access, e.g.,
- Ethernet
- Wireless



- The Internet is a network of individually administrated networks

#hosts? #autonomous systems?

#interdomain destinations announced?

https://www.google.com/loon/how/

First-Day Class: What is a Network Protocol?

- A **network protocol** defines the format and the order of messages exchanged between two or more communicating entities, as well as the actions taken on the transmission and/or receipt of a message or other events.

Protocols that we have reasonably covered?

The 433/533 Protocols Collection

- SMTP, FTP, GridFTP, HTTP, Onion Routing, Chord,
- DNS (mDNS, DNS-SD)
- UDP, TCP
- RIP, EIGRP, OSPF, BGP
- IP, ICMP, ARP, DHCP
- Ethernet

First-Day Class: General Complexity

- Complexity in highly organized systems arises primarily from design strategies intended to create robustness to uncertainty in their environments and component parts.
  - Scalability is robustness to changes to the size and complexity of a system as a whole.
  - Evolvability is robustness of lineages to large changes on various (usually long) time scales.
  - Reliability is robustness to component failures.
  - Efficiency is robustness to resource scarcity.
  - Modularity is robustness to component rearrangements *and autonomy*.

Examples of increased complexity?

First-Day Class: Evolution

- Driven by Technology, Infrastructure, Applications (usage), and Understanding:
  - technology
    - e.g., wireless/optical/quantum communication technologies and device miniaturization (sensors)
  - infrastructure
    - e.g., cloud computing vs local computing vs edge computing
  - applications (usage)
    - e.g., critical infrastructure, cloud computing, mobile computing, content distribution, game, tele presence, sensing
  - understanding
    - e.g., resource sharing principle, routing principles, mechanism design, optimal stochastic control (randomized access)
- Complexity comes from evolution.
- Don't be afraid to challenge the foundation and redesign!

Have a great winter break!

Backup Slides

Getting Faster
(Technology Driven)

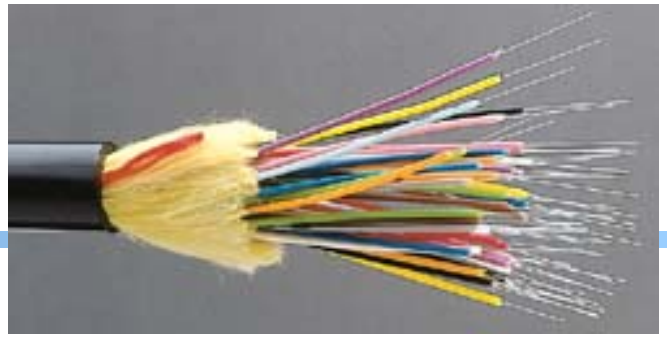
Basic Theory: Channel Capacity

- The maximum number of bits that can be transmitted per second (bps) by a physical media is:

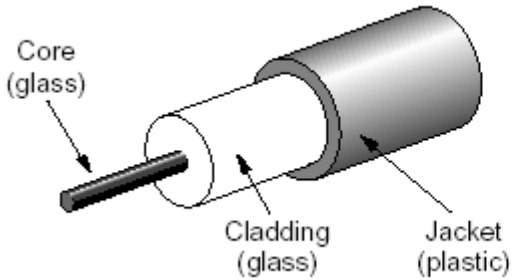
$$W \log_2 (1 + \frac{S}{N})$$

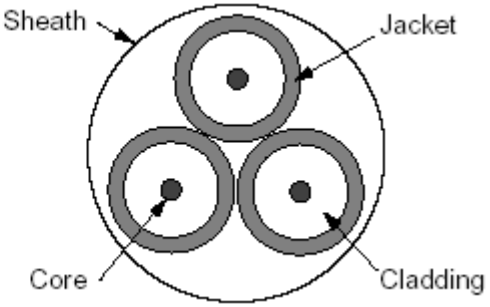
where W is the frequency range, S/N is the signal noise ratio. We assume Gaussian noise.

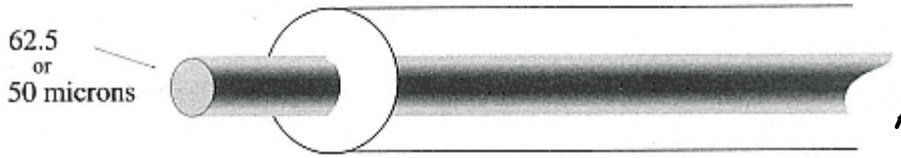
The Wire: Fiber



- A look at a fiber

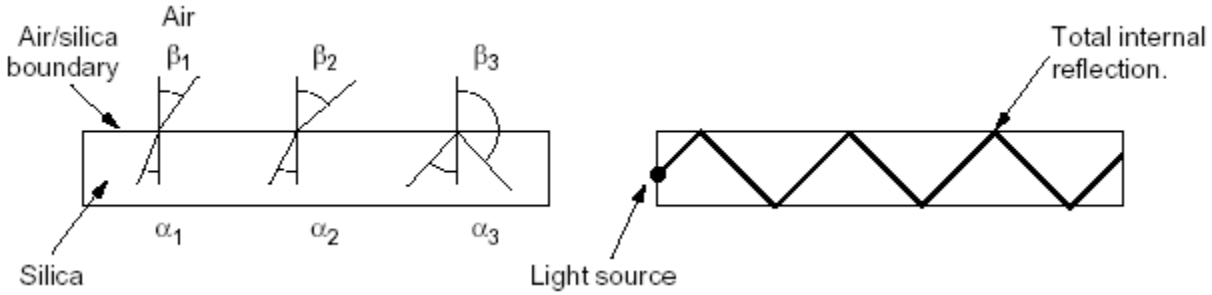






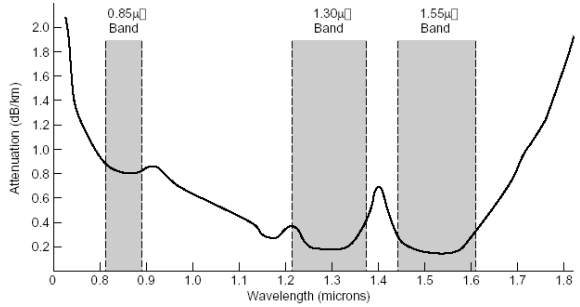
A graded index fiber

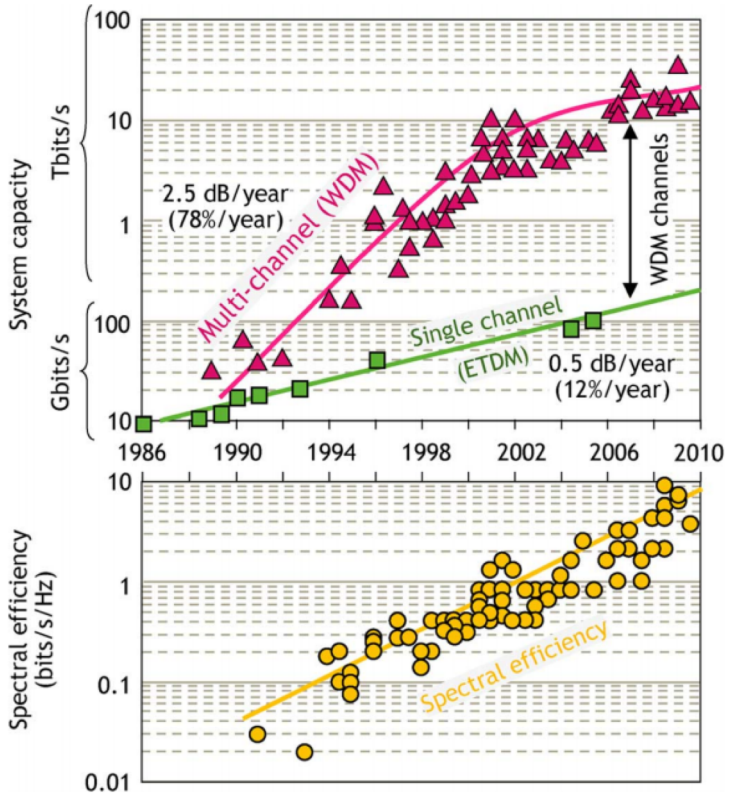
- How it works?



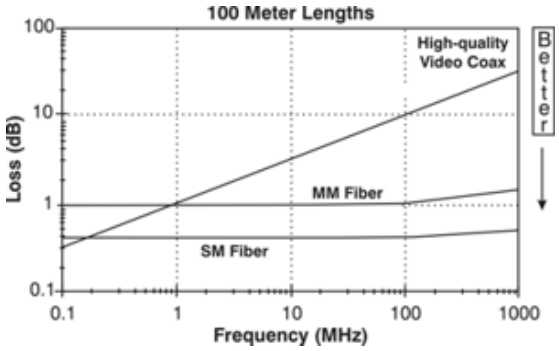
The Wire: Fiber

- Wide spectrum (large W) at low loss (high S/N): ~0.3db/km (c.f. copper ~190db/km @100Mhz), 30-100km without repeater



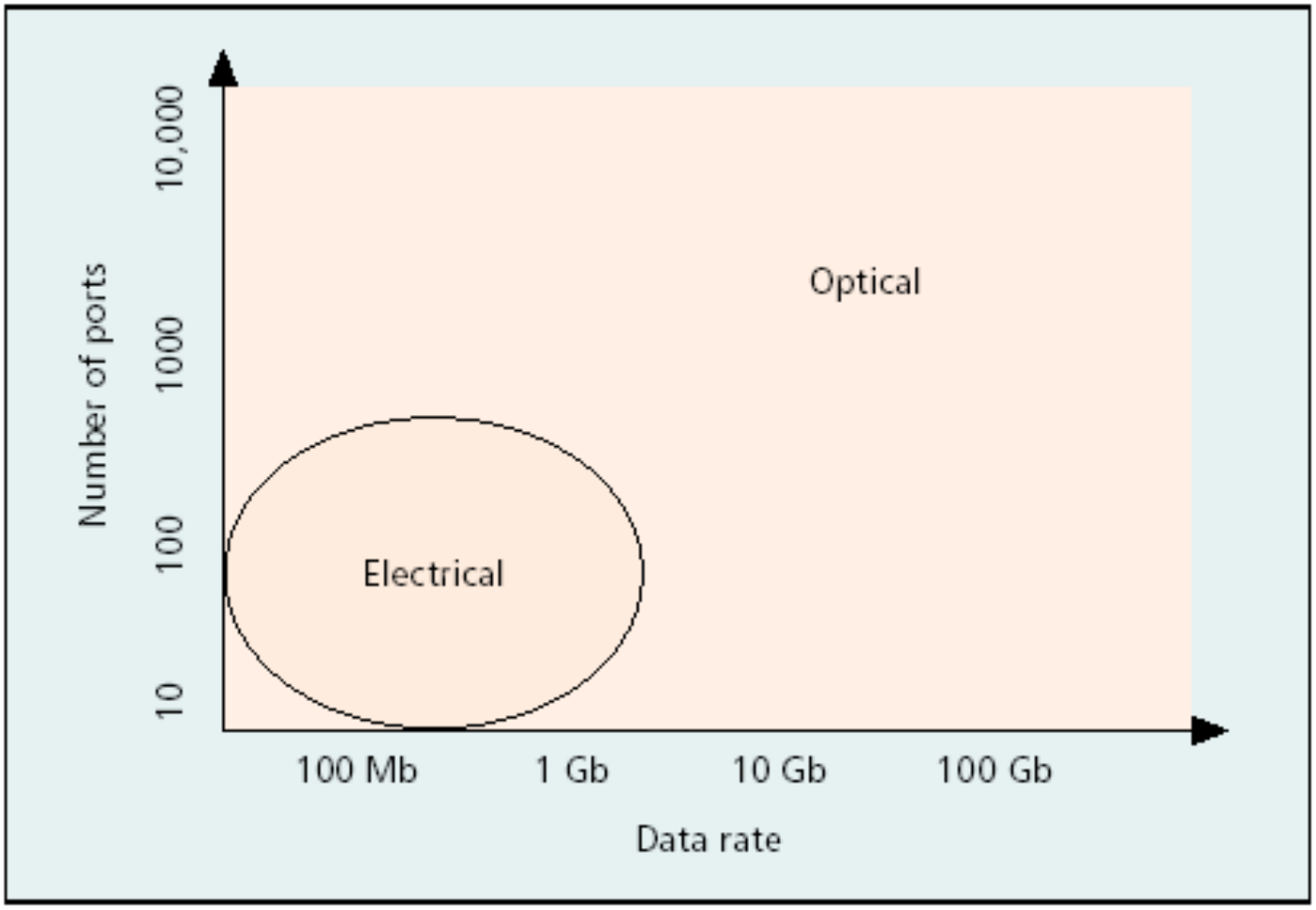


- Lightweight: 33 tons of copper to transmit the same amount of information carried by $\frac{1}{4}$ pound of optical fiber



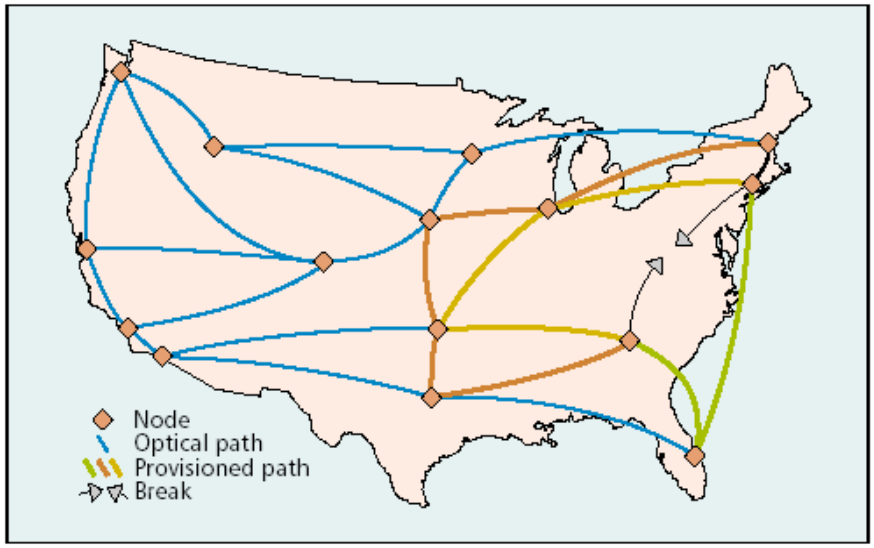
Capacity Limits of Optical Fiber Networks
http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5420239

Advantages of Fibers

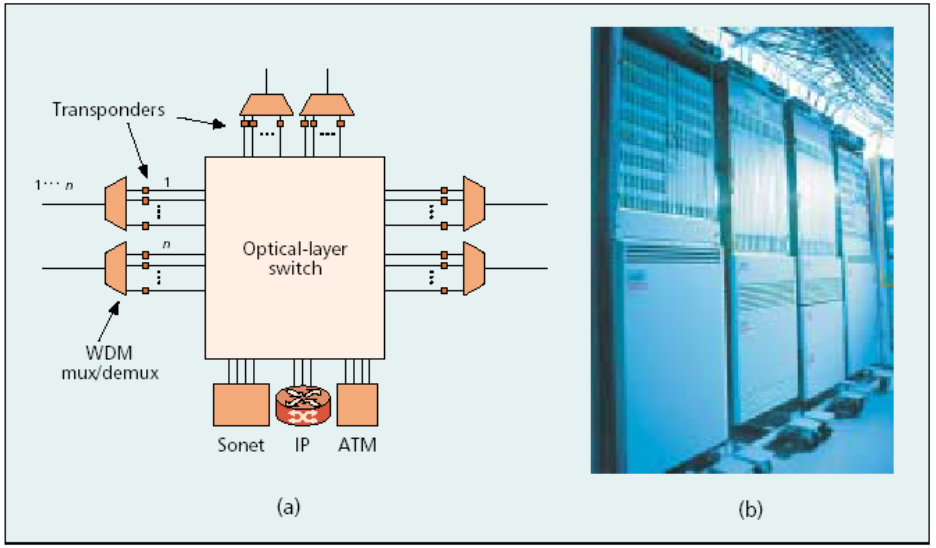


How to Do Switching?

- Optical-Electrical-Optical
- Optical switch: optical micro-electro-mechanical systems (MEMS)



Optical path

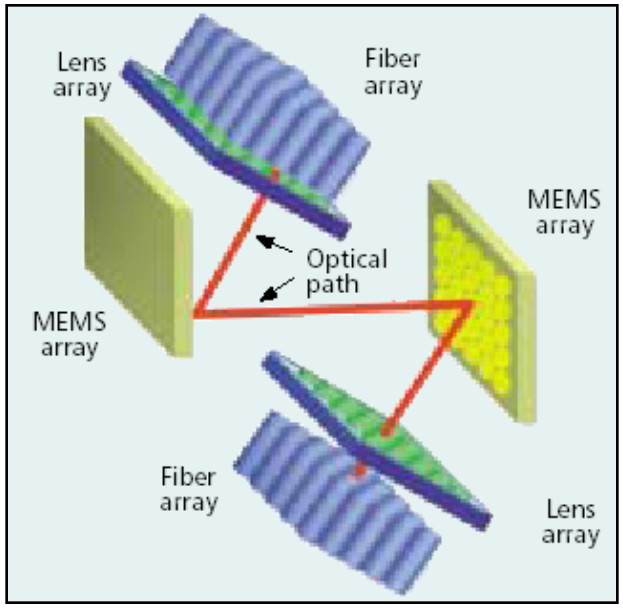


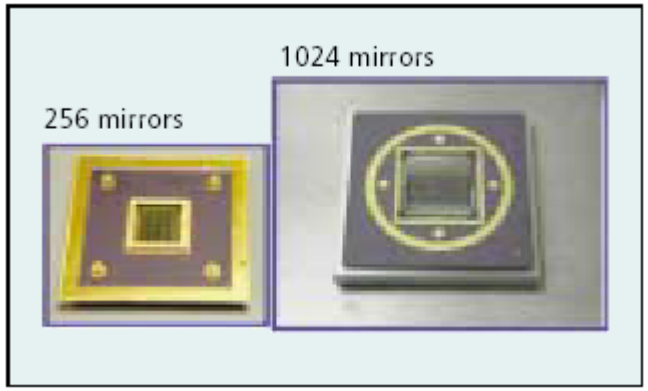
One optical switch

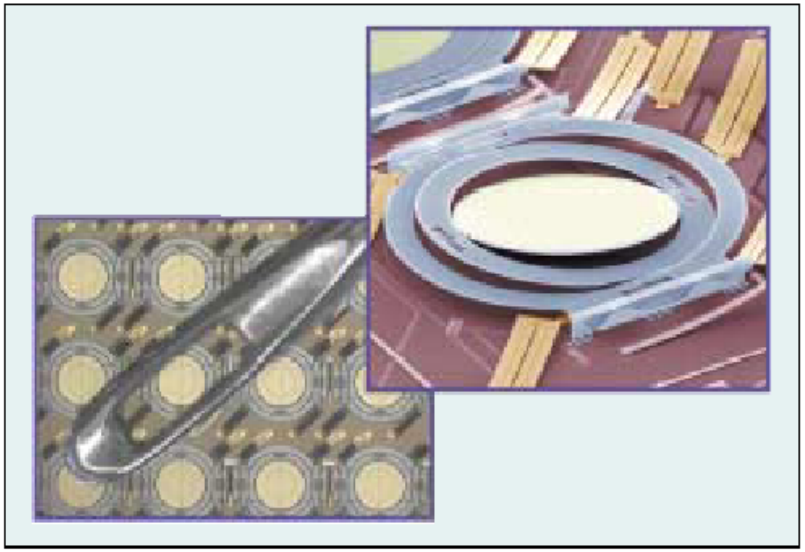
http://www.qwest.com/largebusiness/enterprisesolutions/networkMaps/preloader.swf

Example: MEMS Optical Switch

- Using mirrors, e.g. Lambda Router







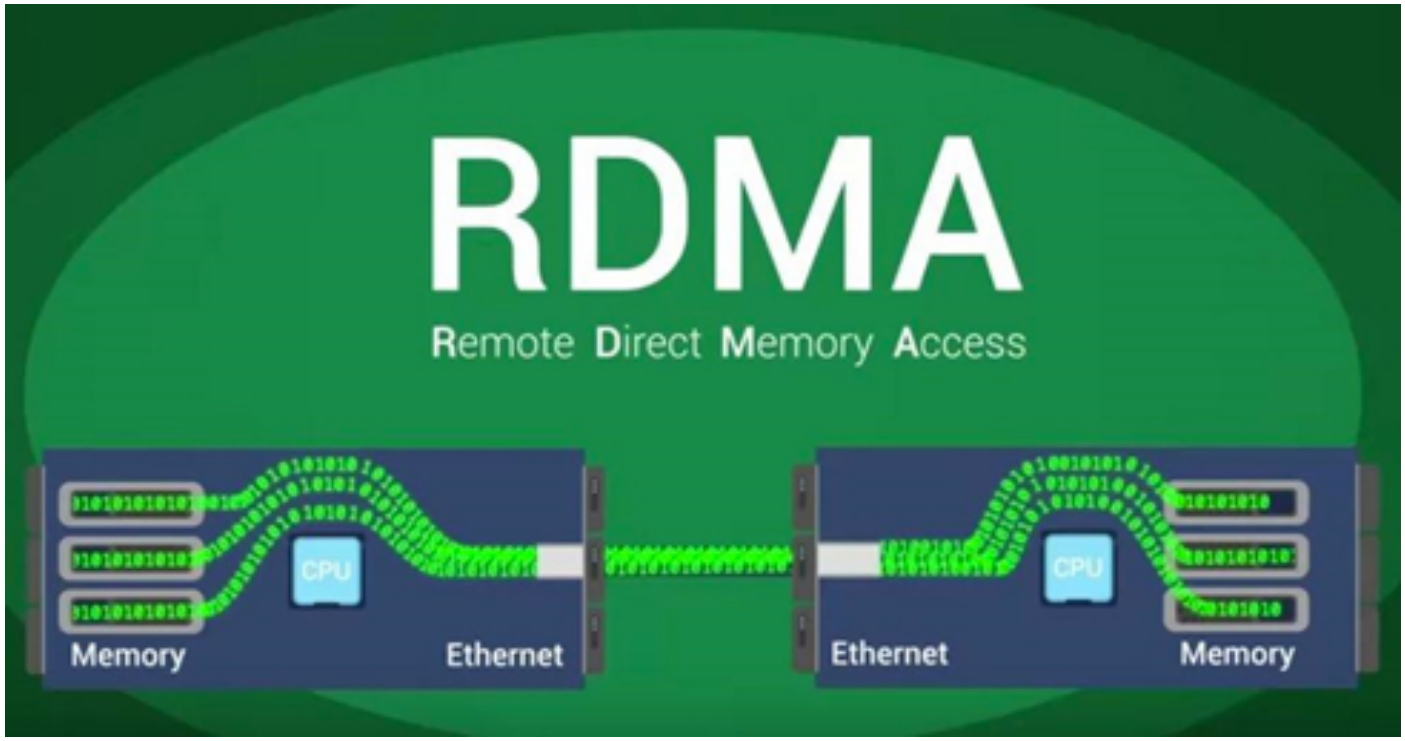
Two images of MEMS-based OXC mirrors used in the Lucent LambdaRouter. The image in the upper right is a single mirror, and an array of mirrors is shown in the lower left. An eye of a needle is shown for comparison on the array.

Implications

- Fine-grained switching may not be feasible
- What is the architecture of optical networks: packet switching, circuit switching, or others?

Getting Faster
(Technology Driven)

End Host: RDMA



Existing socket API (Assignment 4) typically involves memory copy but this is slow. RDMA wants to remove the memory copy. Will your API still work?

More Reliable
(Usage Driven)

Is the Internet Reliable?

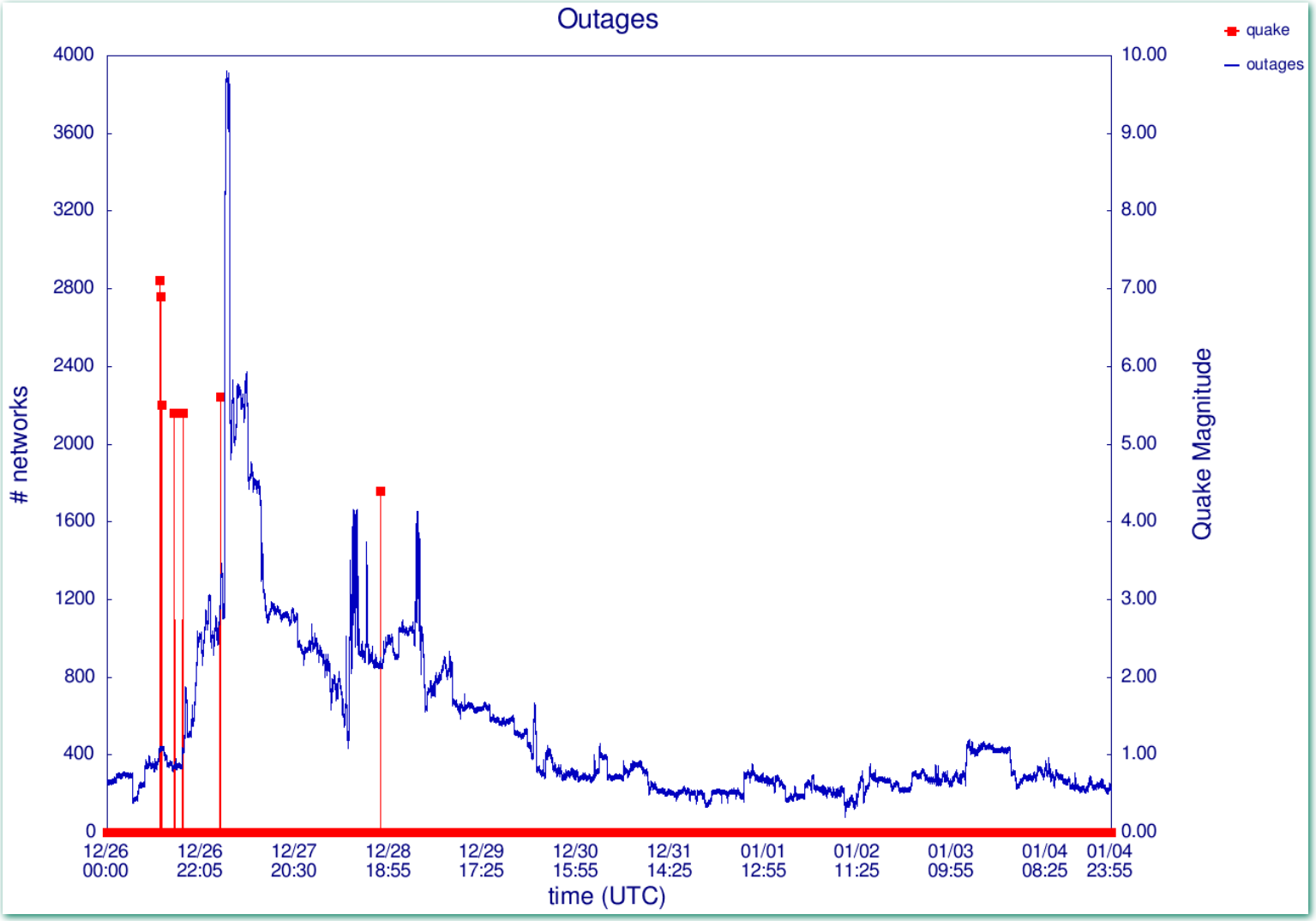
- More people are cutting the wire to depend only on the Internet--- critical infrastructure

- Does the Internet infrastructure achieve the target reliability objective of a highly reliable system (99.999%)?

Perspective

- 911 Phone service (1993 NRIC report +)
  - 29 minutes per year per line
  - 99.994% availability
- Std. Phone service (various sources)
  - 53+ minutes per line per year
  - 99.99+% availability
- *…what about the Internet?*
  - *Various studies: about 99.5%*
  - *Need to reduce down time by 500 times to achieve five nines; 50 times to match phone service*

Unreachable Networks: 10 days



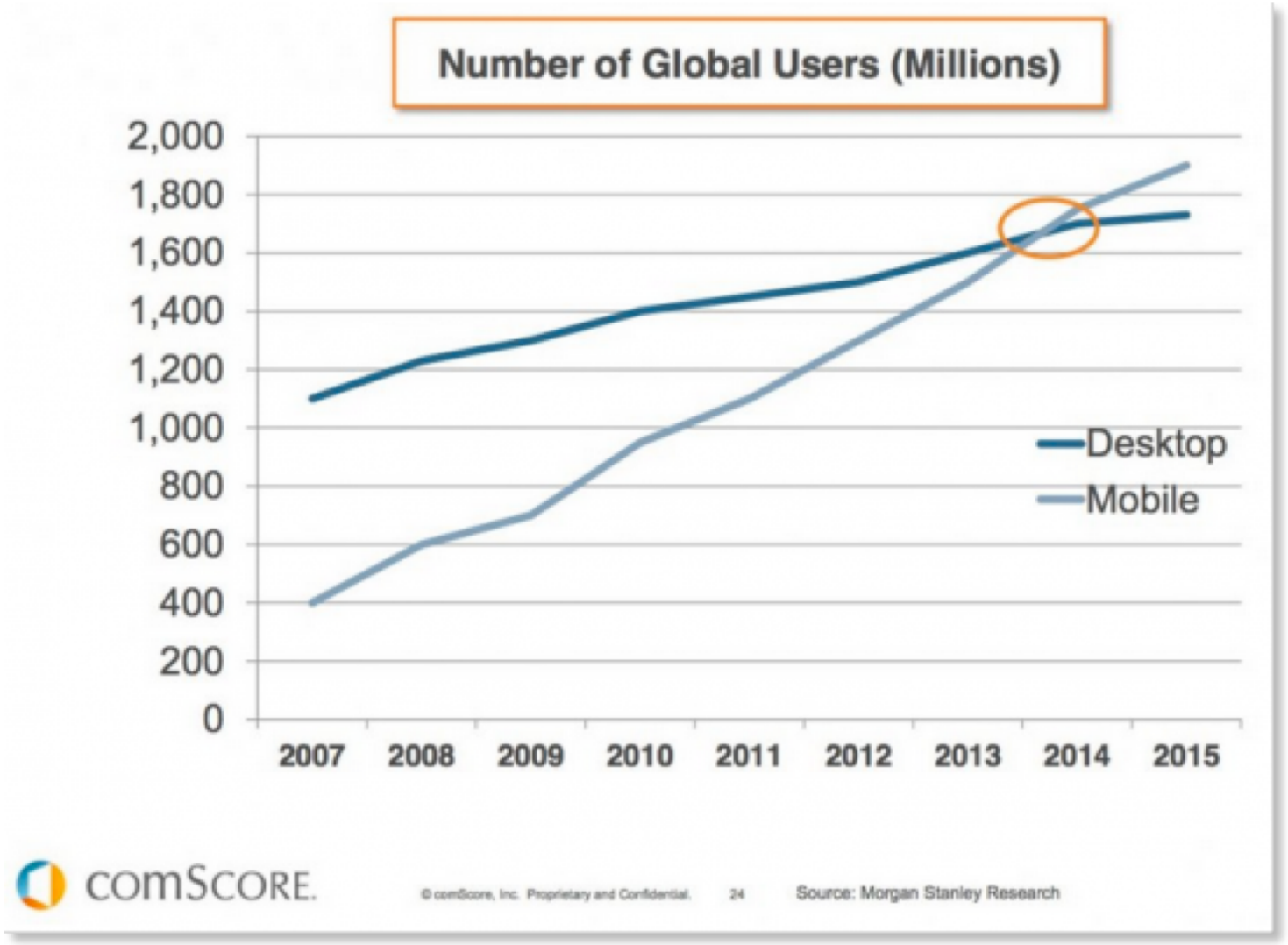
Internet Disaster Recovery Response

- Why slow response?
  - the cable repairing is slow: not until 21 days after quake
  - BGP is not designed to create business relationship

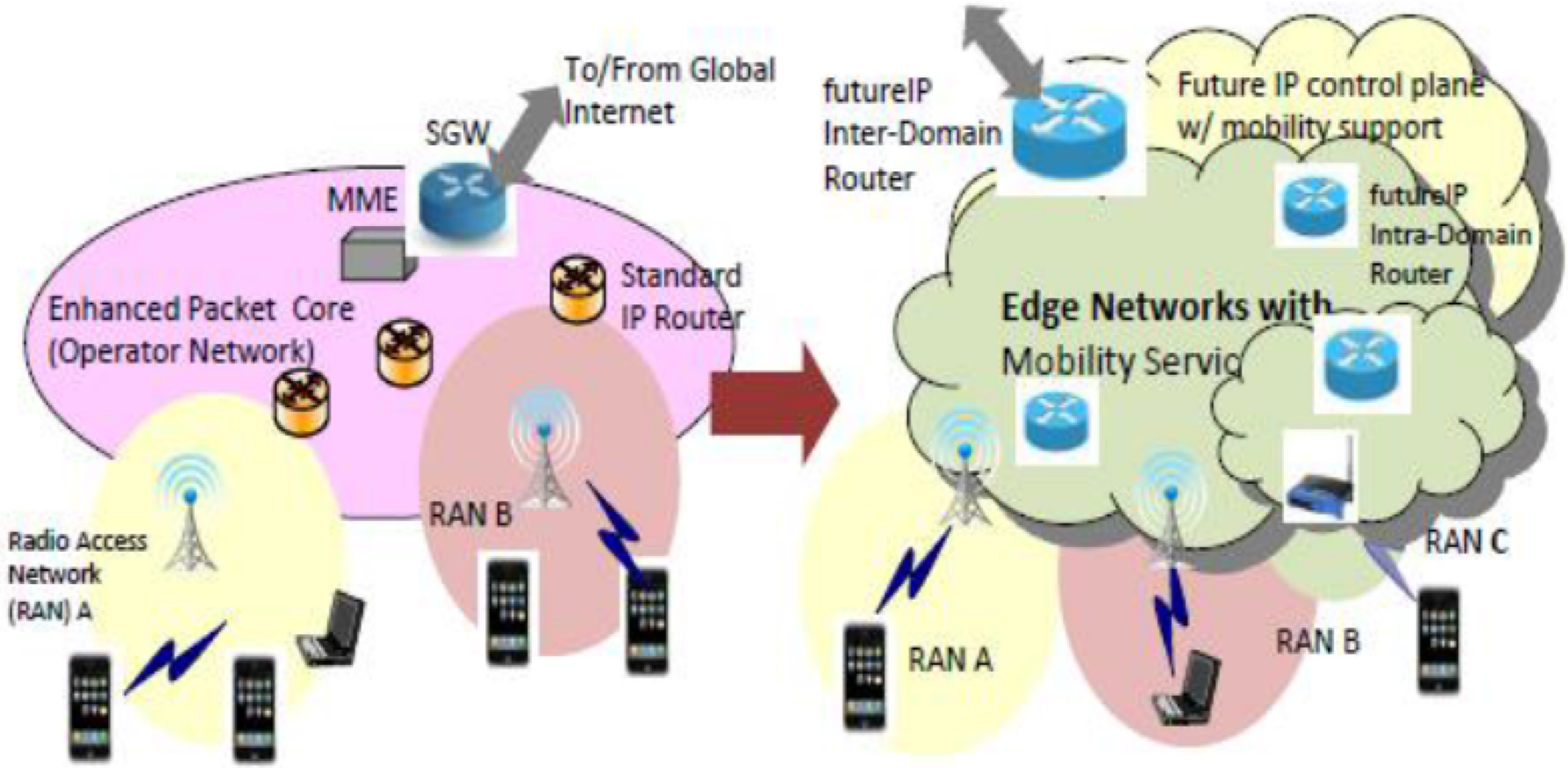
- Issue to think about: How to substantially improve the reliability of the Internet both under typical failures and extreme failures (attacks)?

Mobility First
(Usage Driven)

Shift of Computing Devices



Network Architecture Convergence

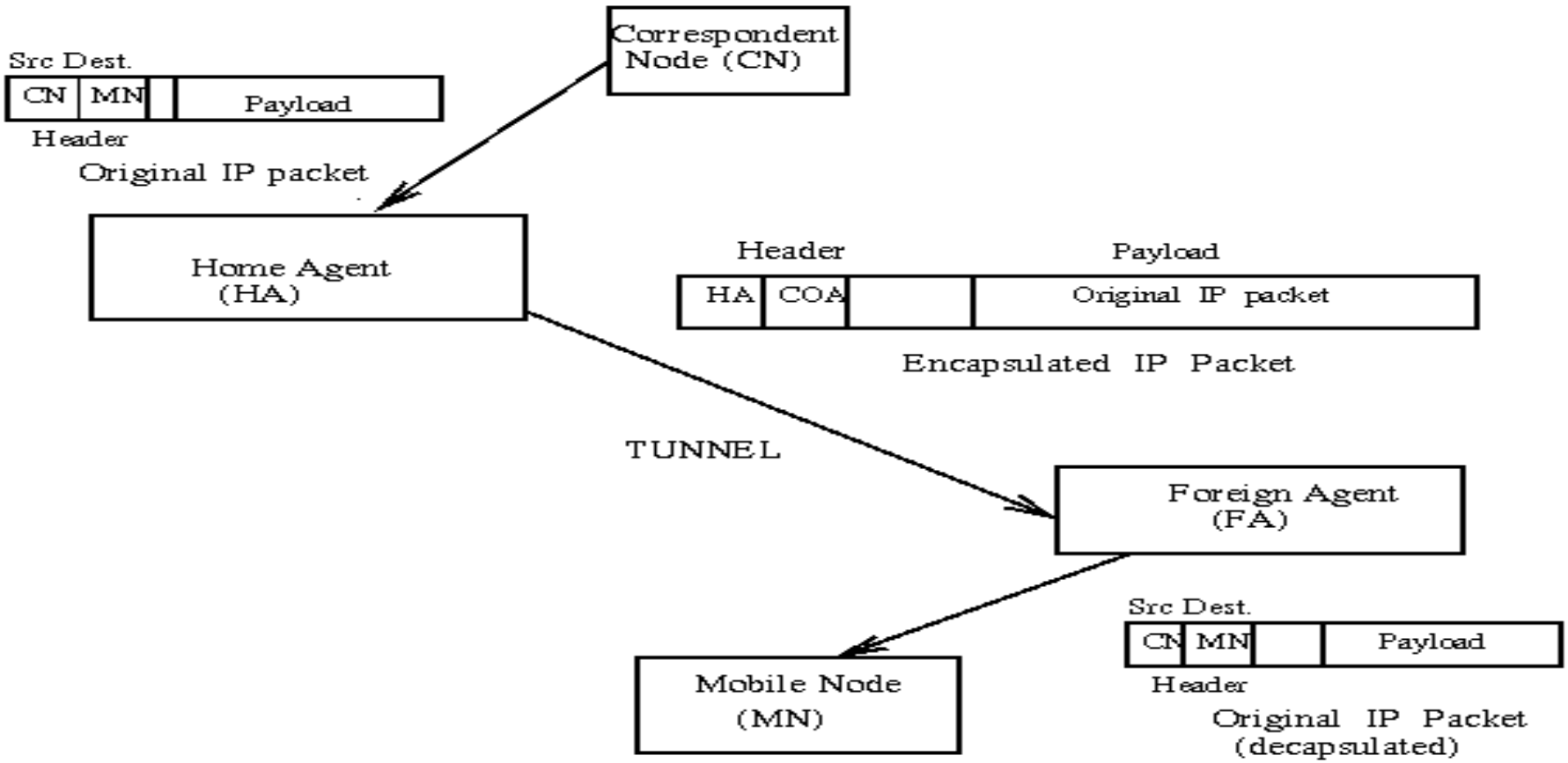


Internet+Cellular

Unified Mobile Internet

Q: Issues of the Internet under mobility?

Current Design: Mobile IP



Problems with Mobile IP

- Suboptimal "triangle" routing
  - What if MN is in same subnetwork as the node to which it is communicating and HA is on the other side of the world?
    - It would be nice if we could directly route packets
  - Solution: Let the CN know the COA of MN
    - Then the CN can create its own tunnel to MN
    - CN must be equipped with software to enable it to learn the COA
    - Initiated by HA who notifies CN via "binding update"
    - Binding table can become stale

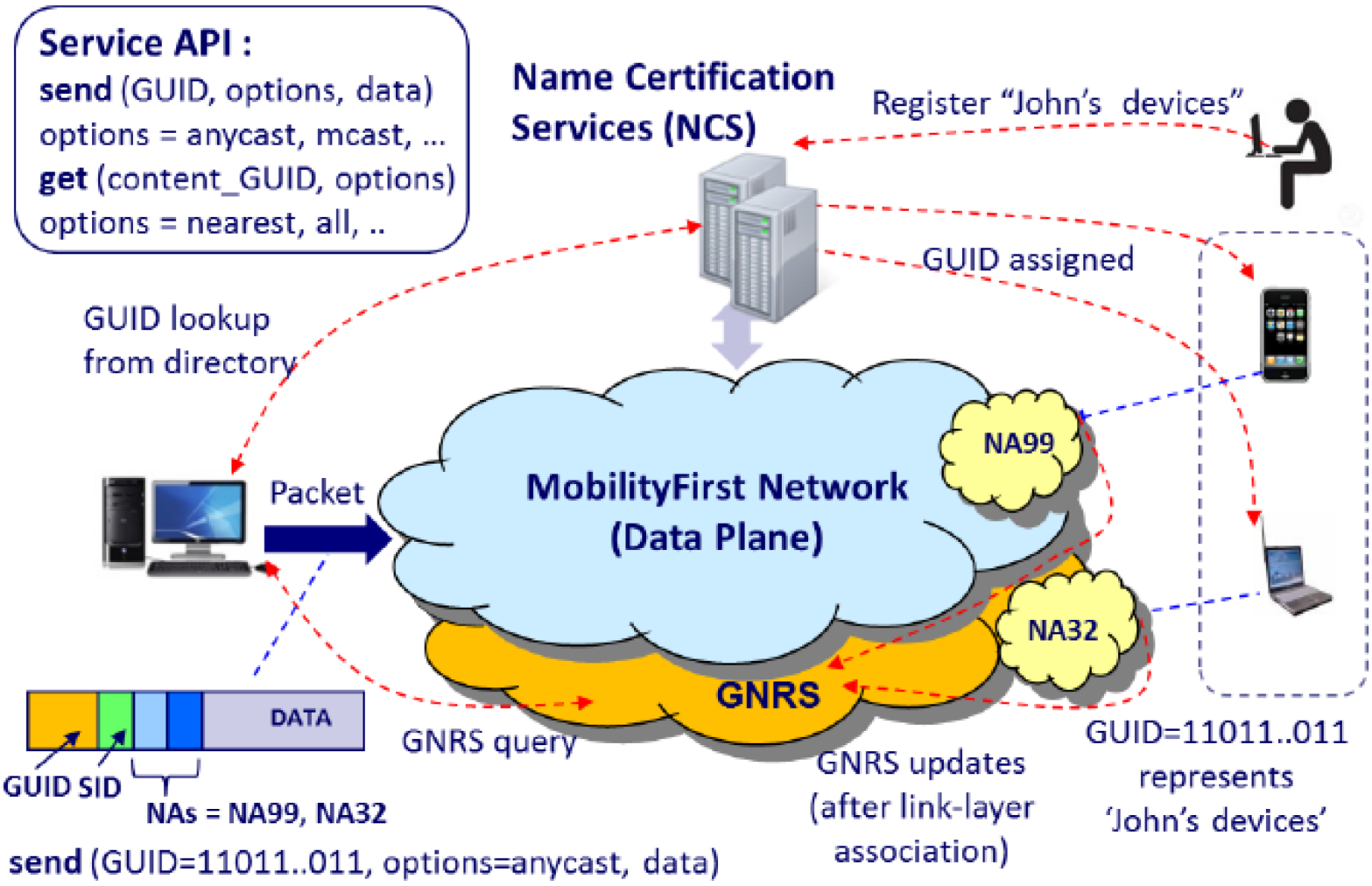
Problems with Mobile IP

- Single HA model is fragile
  - Possible solution – have multiple HA
- Frequent reports to HA if MN is moving
  - Possible solution – support of FA clustering
- Security
  - Connection hijacking, snooping…
- Many issues

An Extreme Design [MobilityFirst]

- Separation of names and locators: Each host/service has a flat GUID (global unique ID), e.g., a secure hash of public key
- A global name resolution service to find a set of network addresses (NAs)
- Routing using both names and network addresses, where the GUID name is considered to be the authoritative packet header

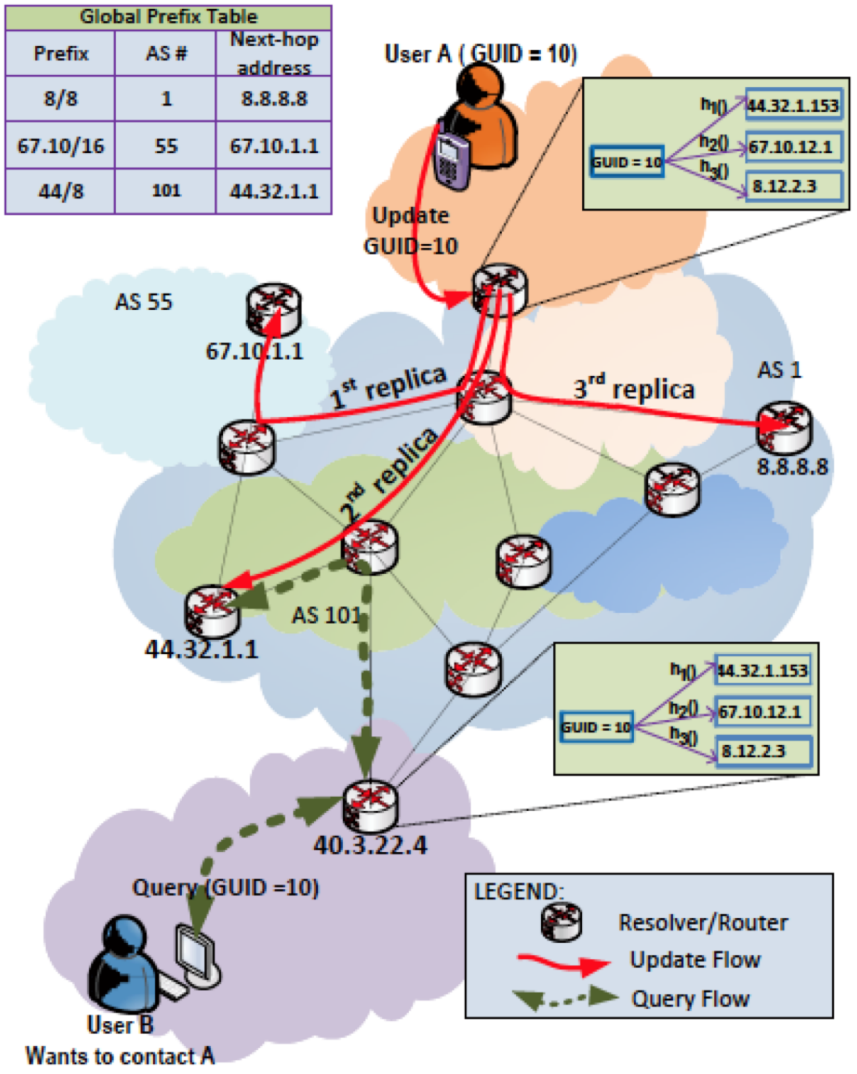
A More Extreme Design [MobilityFirst]



Key Issue: Global Name (ID) Lookup

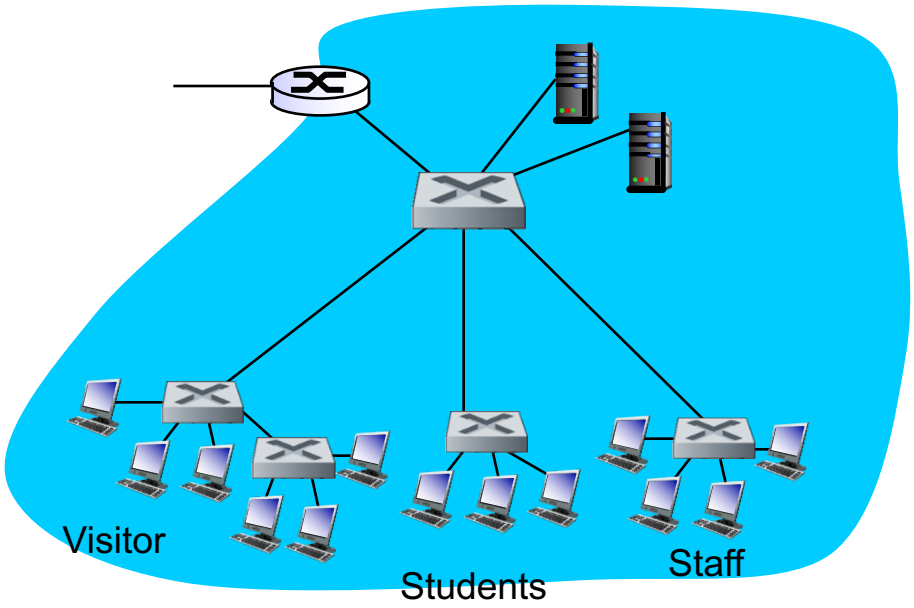
- Keep BGP routing
- Each host hashes its GUID to an IP address, and the AS announcing the IP range hosts the lookup table

Do you buy this design?



Virtualized
(Usage Driven)

Virtualization is not New



consider:

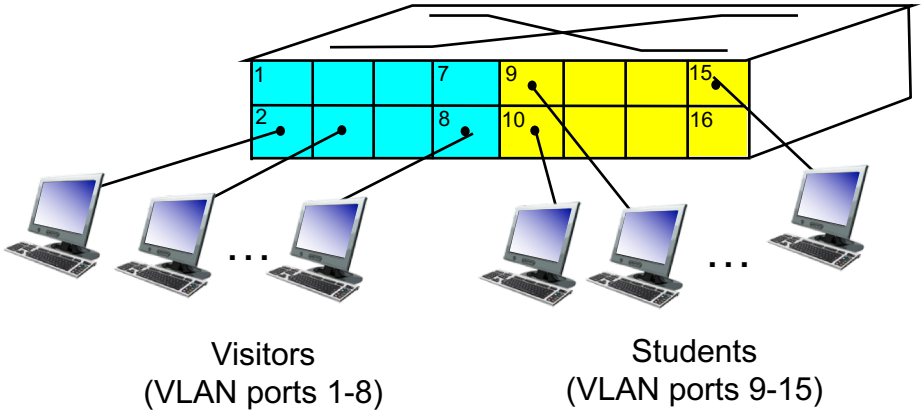
- Mixed office environment
- Per switch per group
  - expensive
- Shared switch
  - single broadcast domain
  - security/privacy, efficiency issues

VLANs

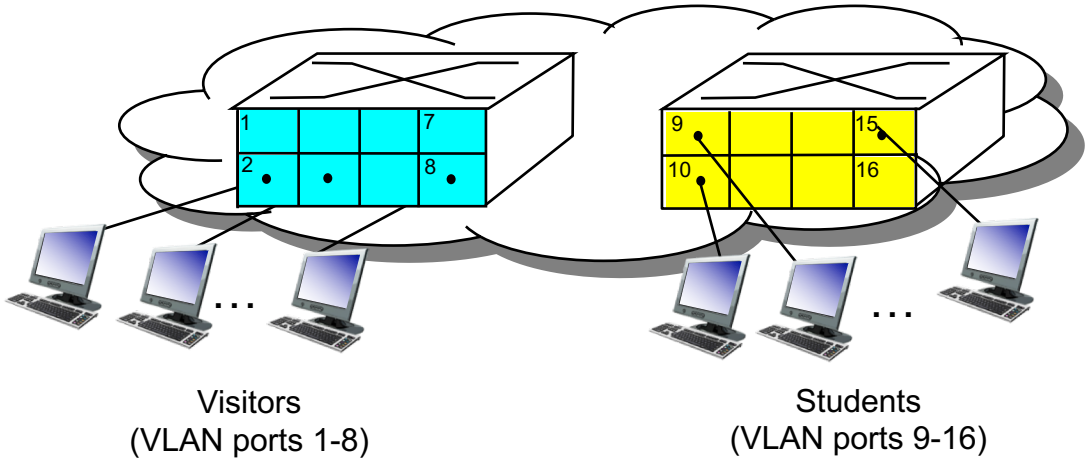
Virtual Local Area Network

switch(es) supporting VLAN capabilities can be configured to define multiple ***virtual*** LANS over single physical LAN infrastructure.

port-based VLAN: switch ports grouped (by switch management software) so that *single* physical switch ……

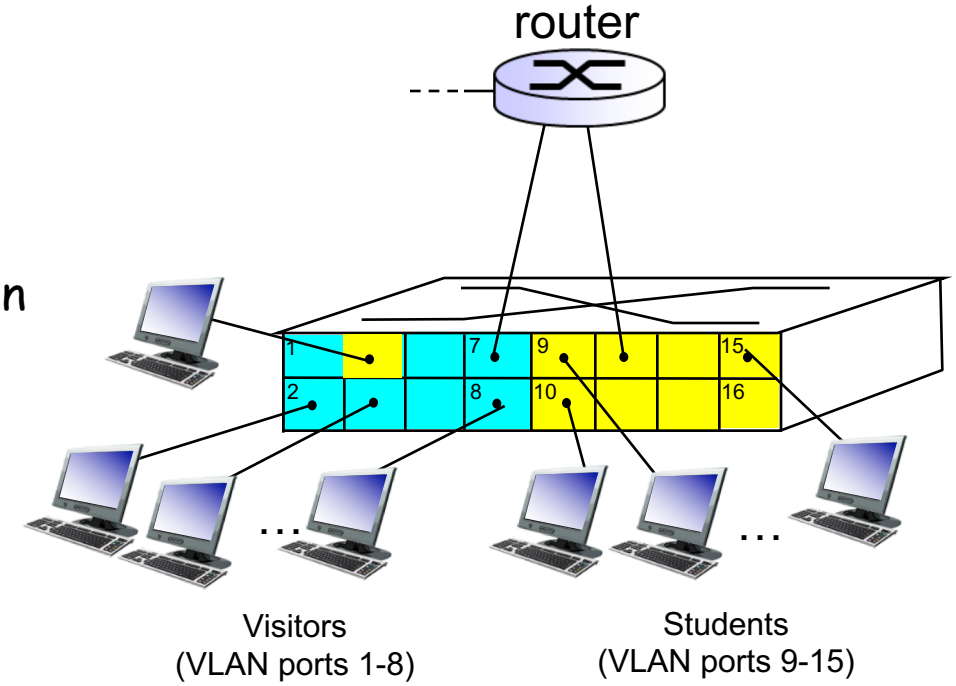


… operates as *multiple* virtual switches



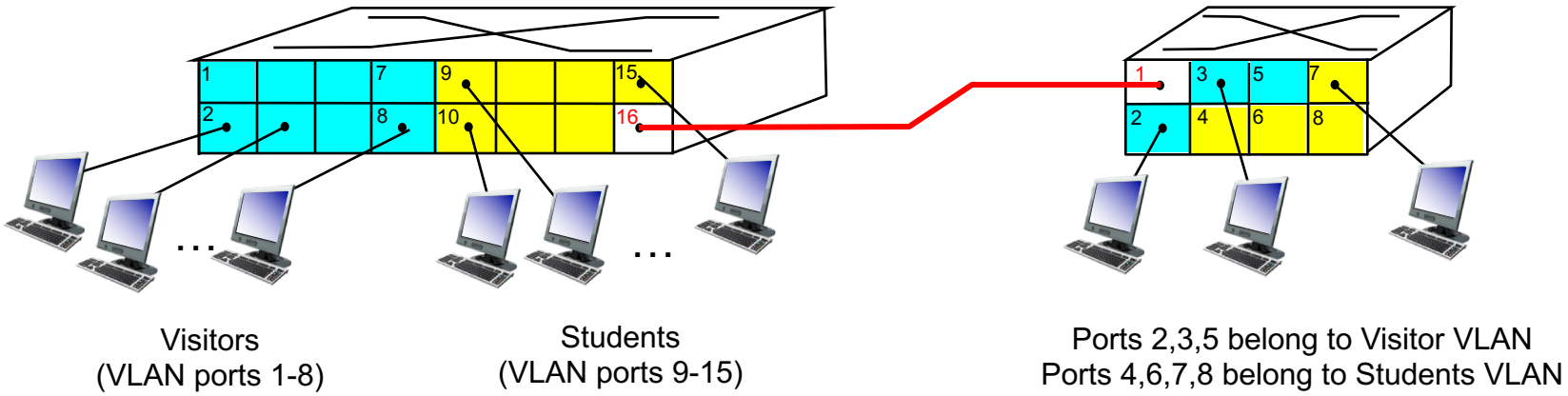
Port-based VLAN

- *traffic isolation:* frames to/from ports 1-8 can *only* reach ports 1-8
  - can also define VLAN based on MAC addresses of endpoints, rather than switch port
- *dynamic membership:* ports can be dynamically assigned among VLANs



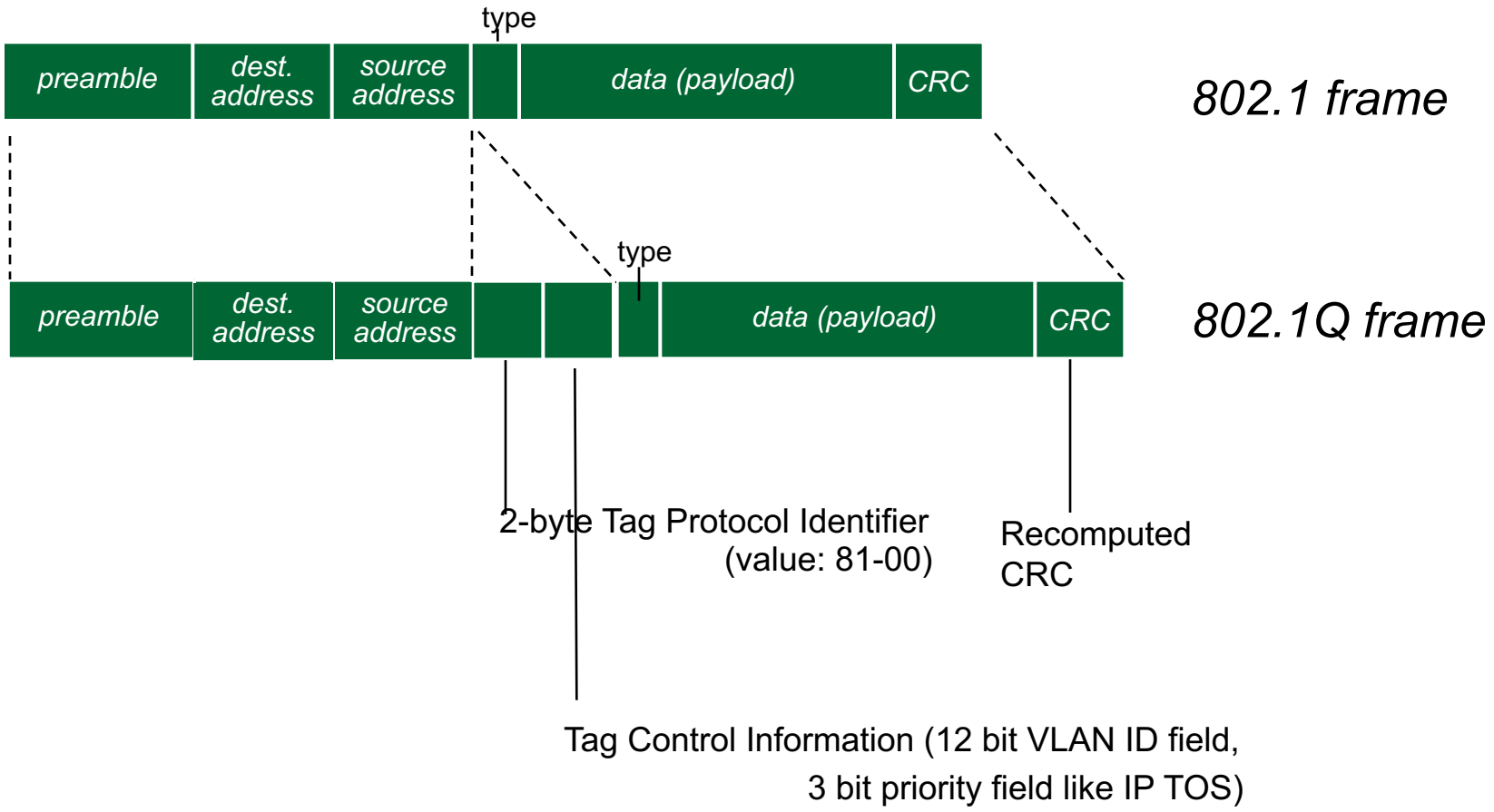
- *forwarding between VLANS:* done via routing (just as with separate switches)
  - in practice vendors sell combined switches plus routers

VLANS spanning multiple switches



- *trunk port:* carries frames between VLANS defined over multiple physical switches
  - frames forwarded within VLAN between switches can't be vanilla 802.1 frames (must carry VLAN ID info)
  - 802.1q protocol adds/removed additional header fields for frames forwarded between trunk ports

802.1Q VLAN frame format



Amazon VPC

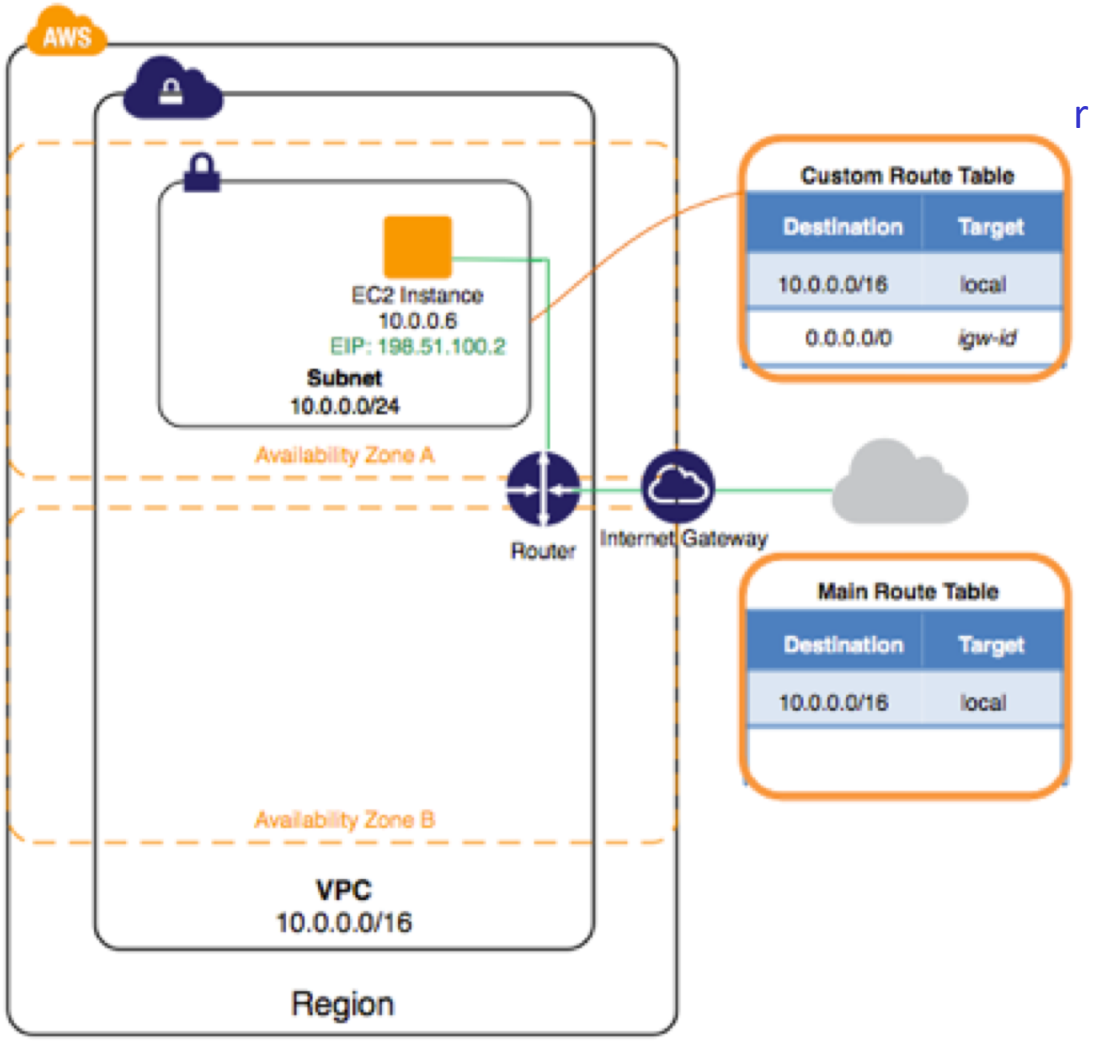
Amazon Virtual Private Cloud

Provision a logically isolated section of the Amazon Web Services (AWS) Cloud where you can launch AWS resources in a virtual network that you define

Amazon Virtual Private Cloud (Amazon VPC) lets you provision a logically isolated section of the AWS Cloud where you can launch AWS resources in a virtual network that you define. You have complete control over your virtual networking environment, including selection of your own IP address range, creation of subnets, and configuration of route tables and network gateways. You can use both IPv4 and IPv6 in your VPC for secure and easy access to resources and applications.

You can easily customize the network configuration for your Amazon VPC. For example, you can create a public-facing subnet for your web servers that has access to the Internet, and place your backend systems such as databases or application servers in a private-facing subnet with no Internet access. You can leverage multiple layers of security, including security groups and network access control lists, to help control access to Amazon EC2 instances in each subnet.

Amazon VPC



r

OpenStack Network API (http://developer.openstack.org/api-ref-networking-v2.html)

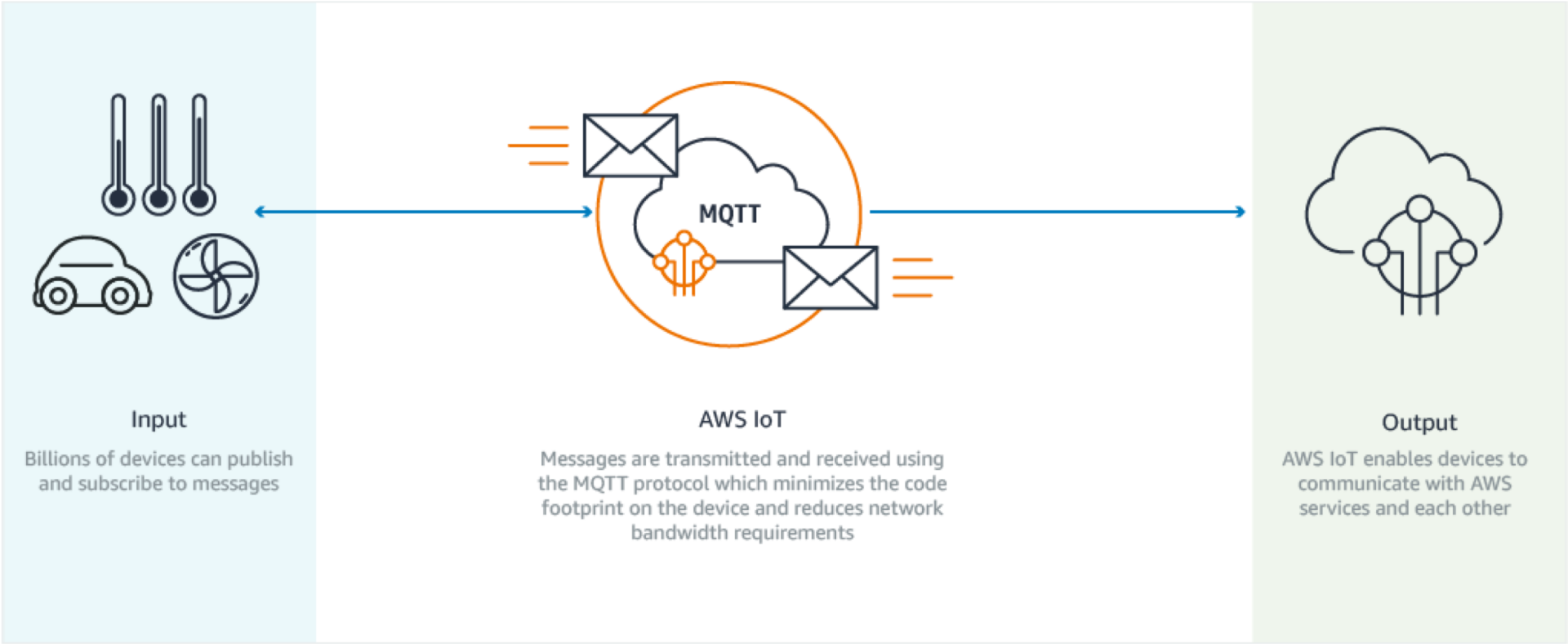
- POST /v2.0/networks
- POST /v2.0/subnets
- POST /v2.0/ports

Issues

- Substantial more complexity in network control
  - Embed virtual networks into shared physical infrastructure, with isolation and QoS
- Question to think about: Assume virtualization is the way to go, how should the Internet infrastructure look like?

Things Oriented
(Usage Driven)

Amazon IoT Core

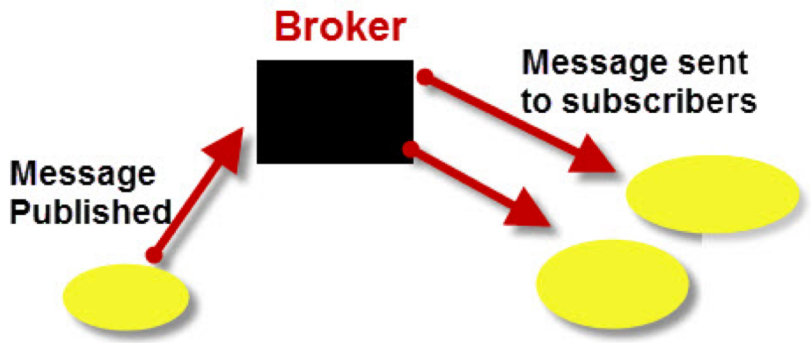


https://aws.amazon.com/iot-core/

MQTT: Message Queuing Telemetry Transport Protocol

MQTT

- TCP/IP based, pub/sub protocol using topics
- Topics format: hierarchical names, e.g.
  - house/room/main-light
- Sub w/ wildcard
  - sub house/#
    - house/room1/main-light
    - house/room1/alarm
    - house/garage/main-light
  - sub house/+/main-light
    - house/room1/main-light
    - house/garage/main-light



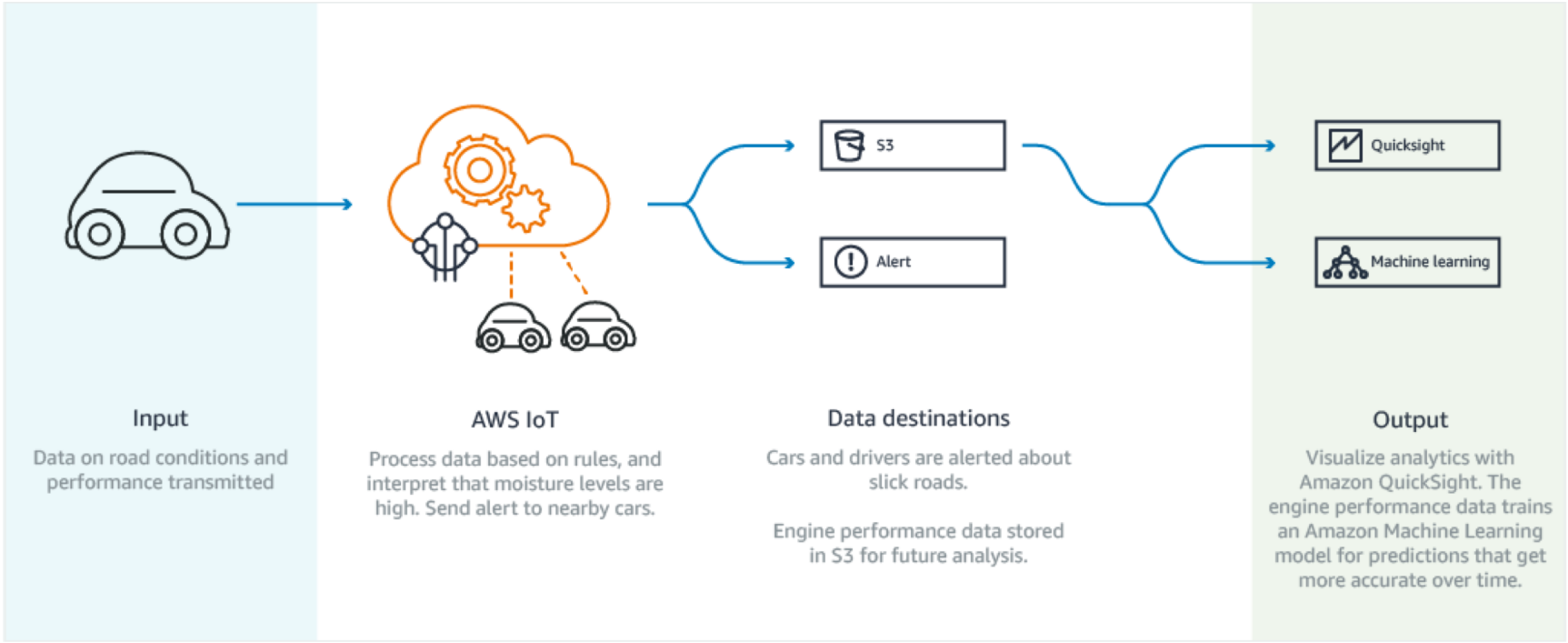
MQTT- Publish Subscribe Model

bit	7	6	5	4	3	2	1	0
byte 1	Message Type				DUP flag	QoS level		RETAIN
byte 2	Remaining Length							

Mnemonic	Enumeration	Description
Reserved	0	Reserved
CONNECT	1	Client request to connect to Server
CONNACK	2	Connect Acknowledgment
PUBLISH	3	Publish message
PUBACK	4	Publish Acknowledgment

http://public.dhe.ibm.com/software/dw/webservices/ws-mqtt/MQTT_V3.1_Protocol_Specific.pdf

Although Much Progress, Still in Early Stage

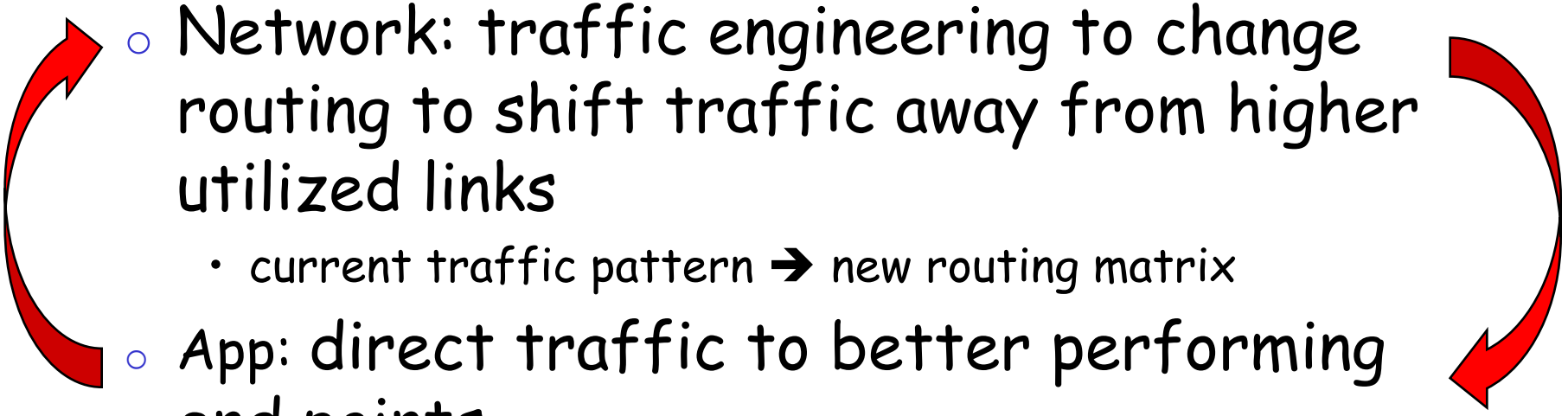


- Questions to think about: The simple topics based sub/pub model can be limited. What is the communication/programming model?

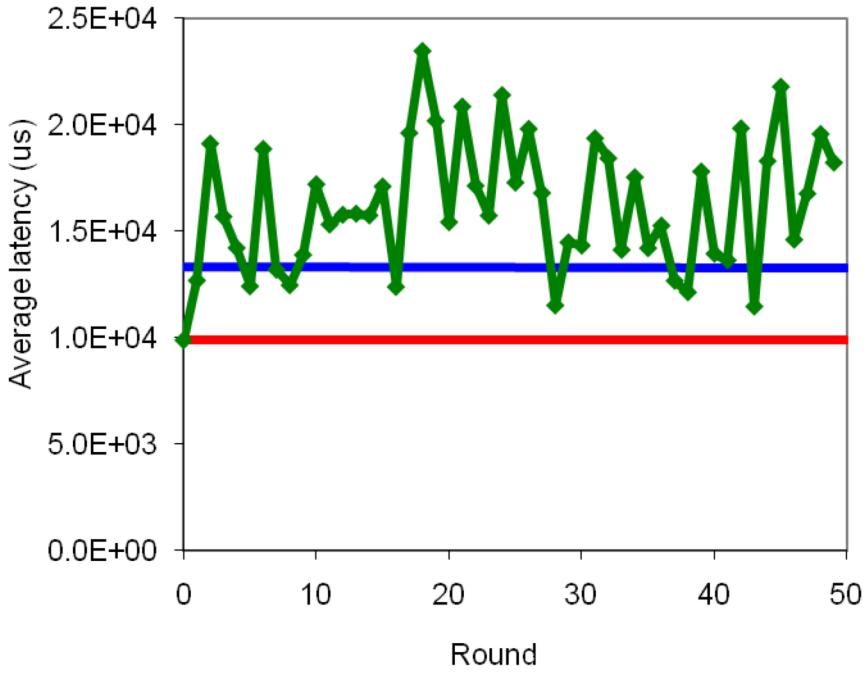
More Efficient
Net/App Integration
(Understanding/Architecture Driven)

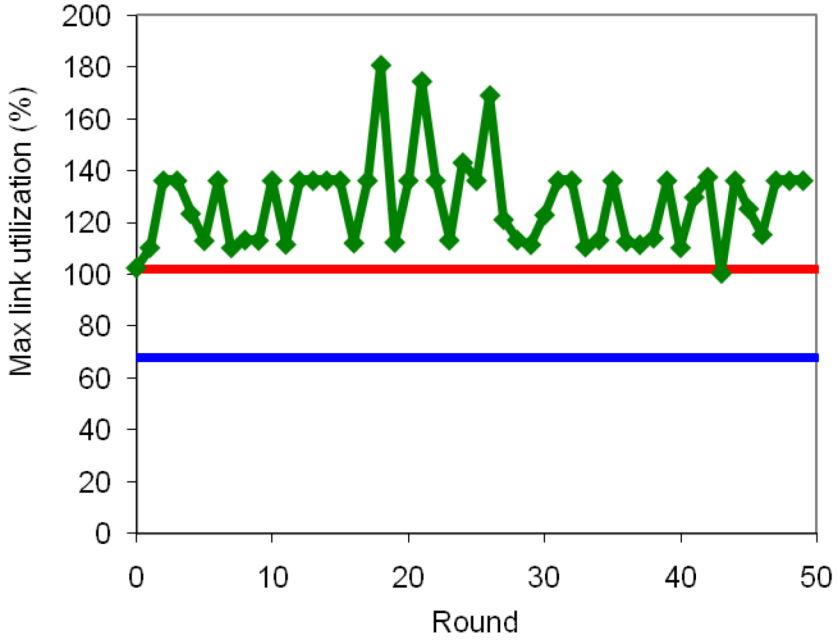
Problem: Inefficient Interactions

- Large deployment of highly adaptive, multipoint applications
- An iterative process between two sets of adaptation:
  - Network: traffic engineering to change routing to shift traffic away from higher utilized links
    - current traffic pattern ➔ new routing matrix
  - App: direct traffic to better performing end points
    - current routing matrix ➔ new traffic pattern



ISP Traffic Engineering+ App Latency Optimizer





- red: App adjust alone; fixed Net routing
- blue: Net traffic engineering adapt alone; fixed App communications

Network optimizer interacts poorly with App.

A Fundamental Problem

- Internet architectural feedback to application efficiency is limited:
  - routing (hidden)
  - rate control through coarse-grained TCP congestion feedback

- To achieve better efficiency, needs explicit communications between network resource providers and applications

Example App Objective

- For example, optimize completion time =>

  maximizing up/down link capacity usage

$$\max \sum_i \sum_{j \neq i} t_{ij}$$

$$s.t. \forall i, \sum_{j \neq i} t_{ij} \leq u_i ,$$

$$\forall i, \sum_{j \neq i} t_{ji} \leq d_i ,$$

$$\forall i \neq j, t_{ij} \geq 0$$

Example Network Objective

- ISP Objective
  - Minimize MLU

- Notations:
  - Assume K applications in the ISP's network
  - b_e: background traffic volume on link e
  - c_e: capacity of link e
  - $I_e(i,j) = 1$ if link e is on the route from i to j
  - t^k : a traffic demand matrix $\{t^k_{ij}\}$ for each pair of nodes (i,j)

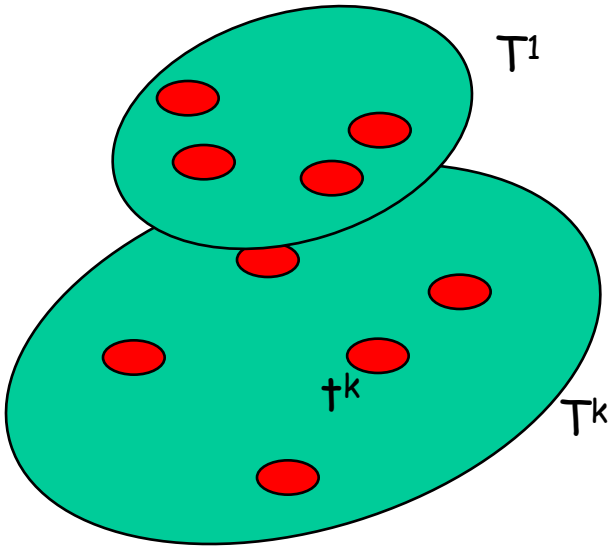
$$\min_{\forall t^k \in T^k} \max_{e \in E} (b_e + \sum_k \sum_{i \neq j} t^k_{ij} I_e(i,j)) / c_e$$

System Formulation

- Combine the objectives of net and applications

$$\min \max_{e \in E} (b_e + \sum_k \sum_{i \neq j} t_{ij}^k I_e(i,j)) / c_e$$

s.t., for any k,



$$\max \sum_i \sum_{j \neq i} t_{ij}^k$$

$$s.t. \forall i, \sum_{j \neq i} t_{ij}^k \leq u_i^k,$$

$$\forall i, \sum_{j \neq i} t_{ji}^k \leq d_i^k,$$

$$\forall i \neq j, t_{ij}^k \geq 0$$

Possible Solution

$$\min \max_{e \in E} (b_e + \sum_k \sum_{i \neq j} t_{ij}^k I_e(i,j)) / c_e$$

s.t., for any k,

$$\max \sum_i \sum_{j \neq i} t_{ij}^k$$

$$s.t. \forall i, \sum_{j \neq i} t_{ij}^k \leq u_i^k,$$

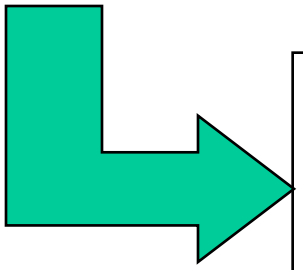
$$\forall i, \sum_{j \neq i} t_{ji}^k \leq d_i^k,$$

$$\forall i \neq j, t_{ij}^k \geq 0$$

- A straightforward approach: centralized solution
- Issues
  - Not application-agnostic
  - Not scalable

Constraints Couple Entities

$$\min_{\forall k: t^k \in T^k} \max_{e \in E} (b_e + \sum_k \sum_{i \neq j} t_{ij}^k I_e(i,j)) / c_e$$



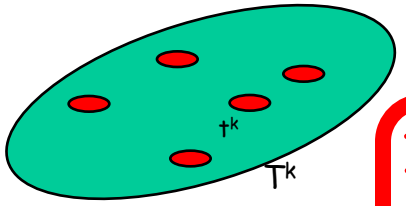
$$\min_{\forall k: t^k \in T^k} \alpha$$

$$s.t. \quad \forall e: b_e + \sum_k \sum_{i \neq j} t_{ij}^k I_e(i,j) \leq \alpha c_e$$

Constraints couple net/app together!

Objective: Decouple Net/Apps

$$\min_{\forall k: t^k \in T^k} \max_{e \in E} (b_e + \sum_k \sum_{i \neq j} t_{ij}^k I_e(i,j)) / c_e$$



Introduce p_e to decouple the constraints

$$\min_{\forall k: t^k \in T^k} \alpha$$

$$s.t. \quad \forall e: b_e + \sum_k \sum_{i \neq j} t_{ij}^k I_e(i,j) \leq \alpha c_e \quad p_e$$

MLU: Dual

- With dual variable p_e (≥ 0) for the inequality of each link e

$$D(\{p_e\}) = \min_{\alpha;\forall k:t^k \in T^k} \alpha + \sum_e p_e(b_e + \sum_k t_e^k - \alpha c_e)$$

- To make the dual finite, need

$$\sum_e p_e c_e = 1$$

MLU: Dual

- Then the dual is

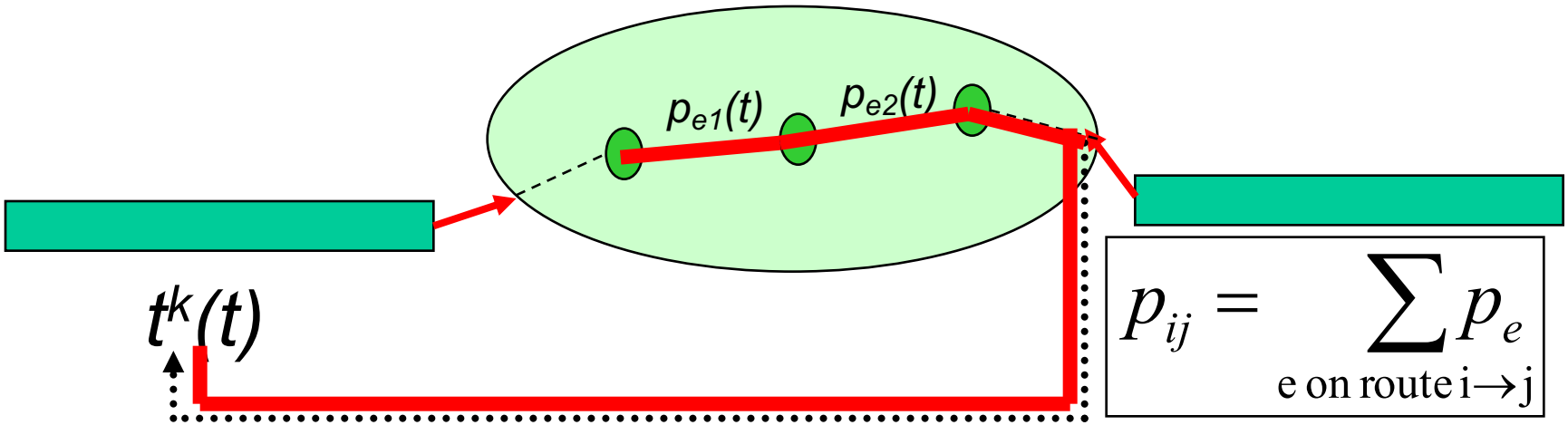
$$D(\{p_e\}) = \min_{\forall k: t^k \in T^k} \sum_e p_e (b_e + \sum_k t_e^k)$$

$$= \sum_e p_e b_e + \sum_k \min_{t^k \in T^k} \sum_{i \neq j} p_{ij} t_{ij}^k$$

where p_{ij} is the sum of p_e along the path from node i to node j

Net/App Interactions

- The interface between applications and providers is the dual variables $\{p_{ij}\}$



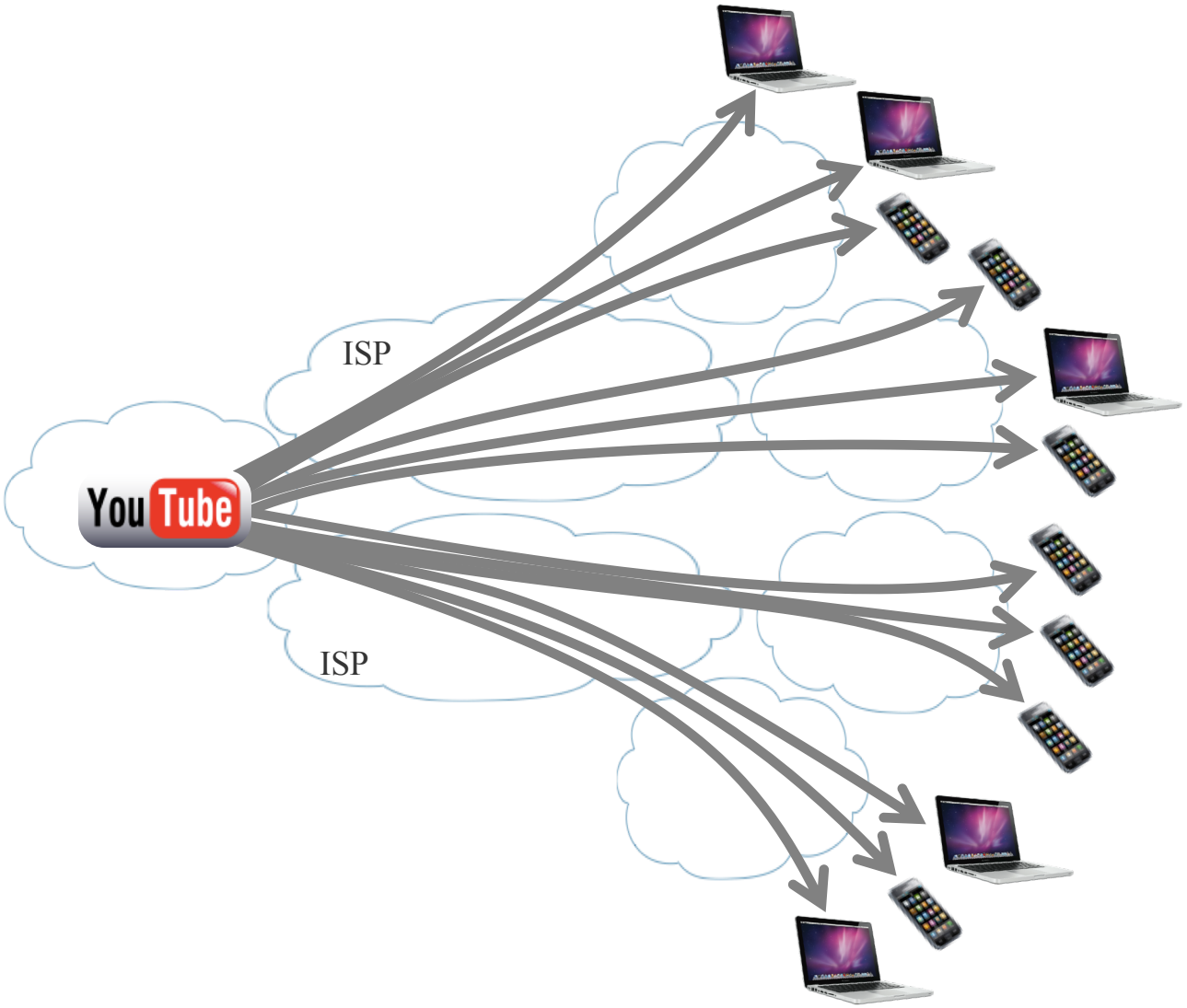
Questions to Think About

- Does the proceeding work for general applications?
- Overall, how to make OTT apps (e.g., Netflix, YouTube, facebook) more resource efficient?

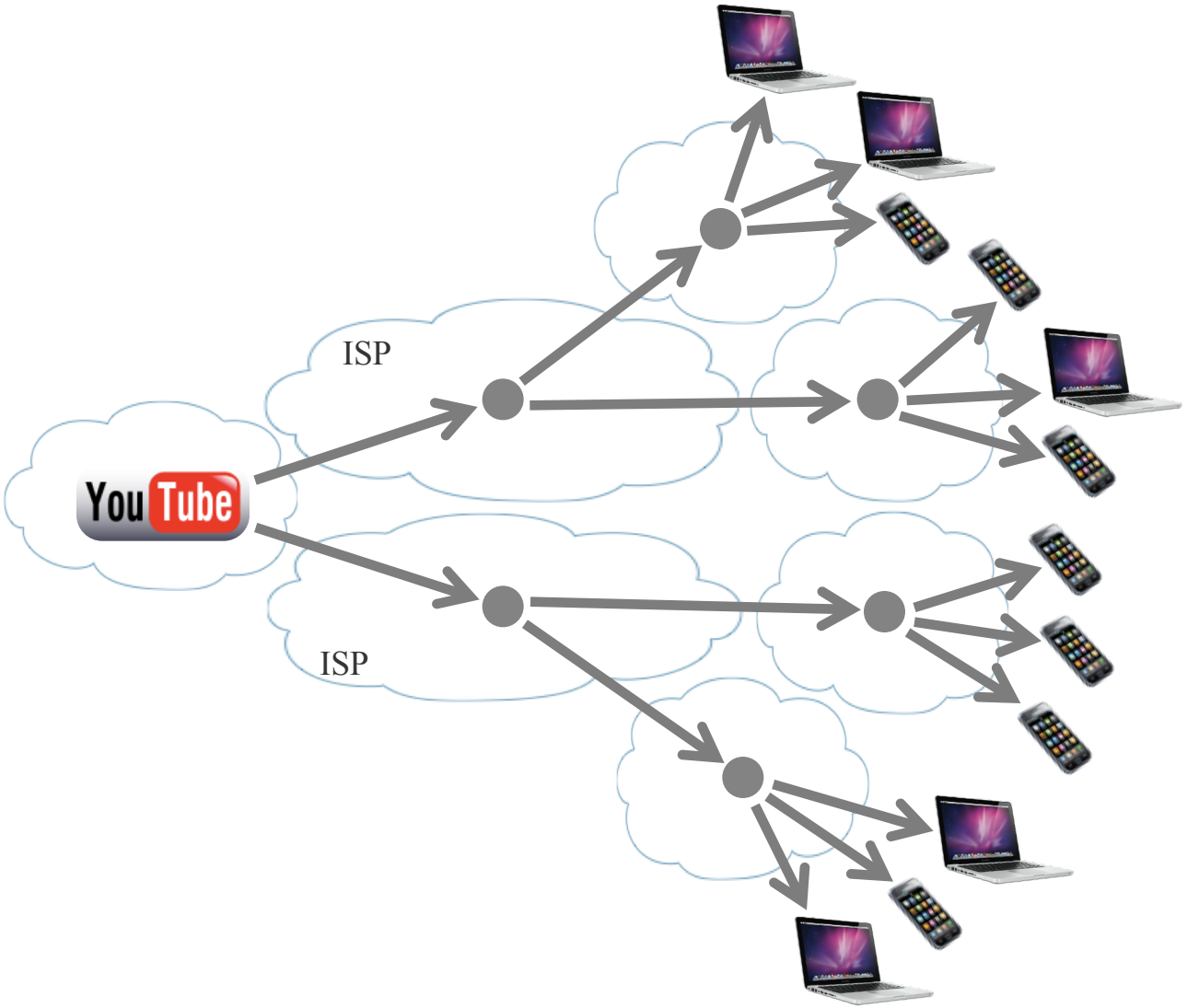
Alternative Design: Named Data Networks, AKA Content Centric

(Understanding/Architecture Driven)

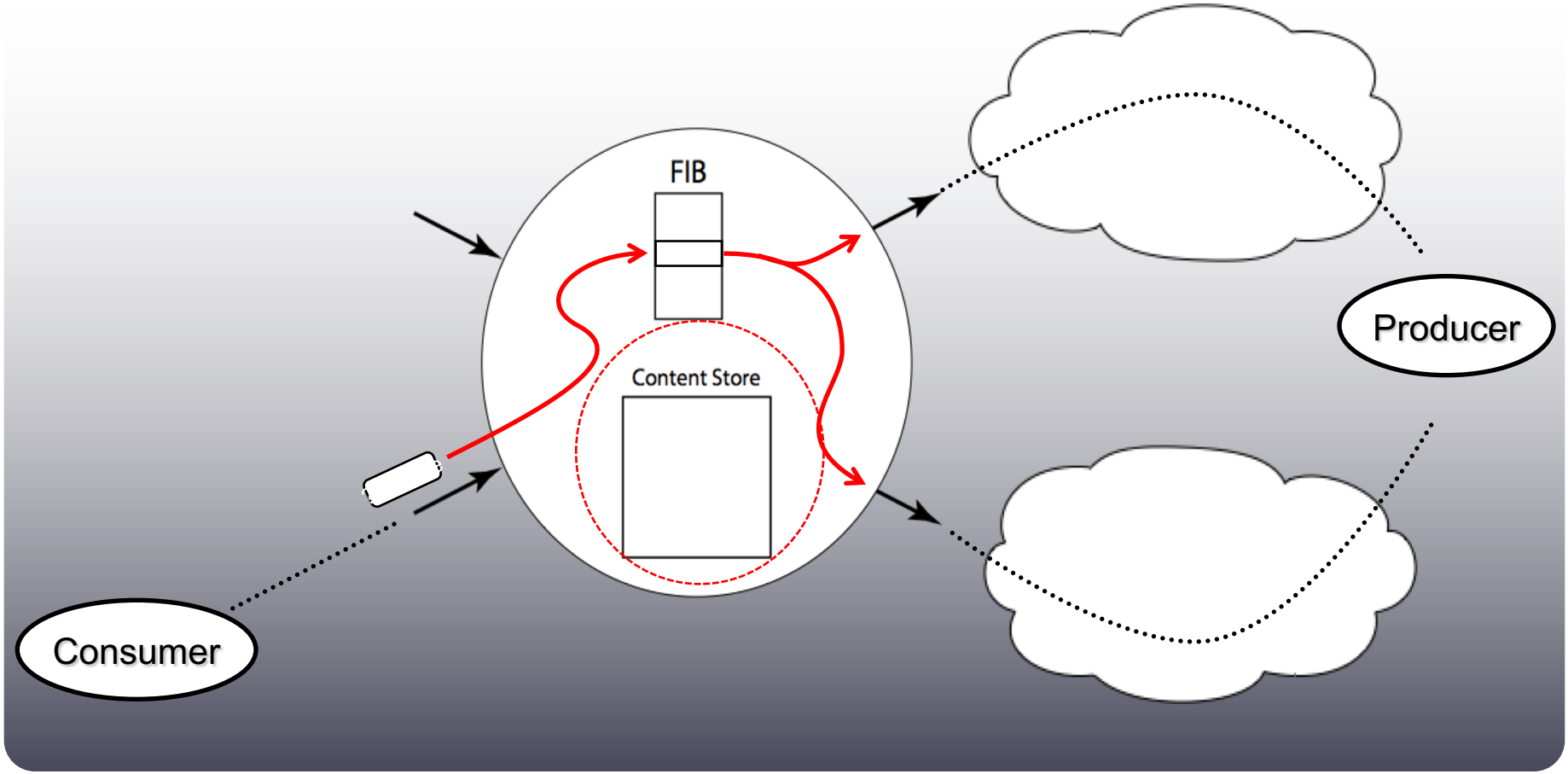
Current Internet



An Alternative

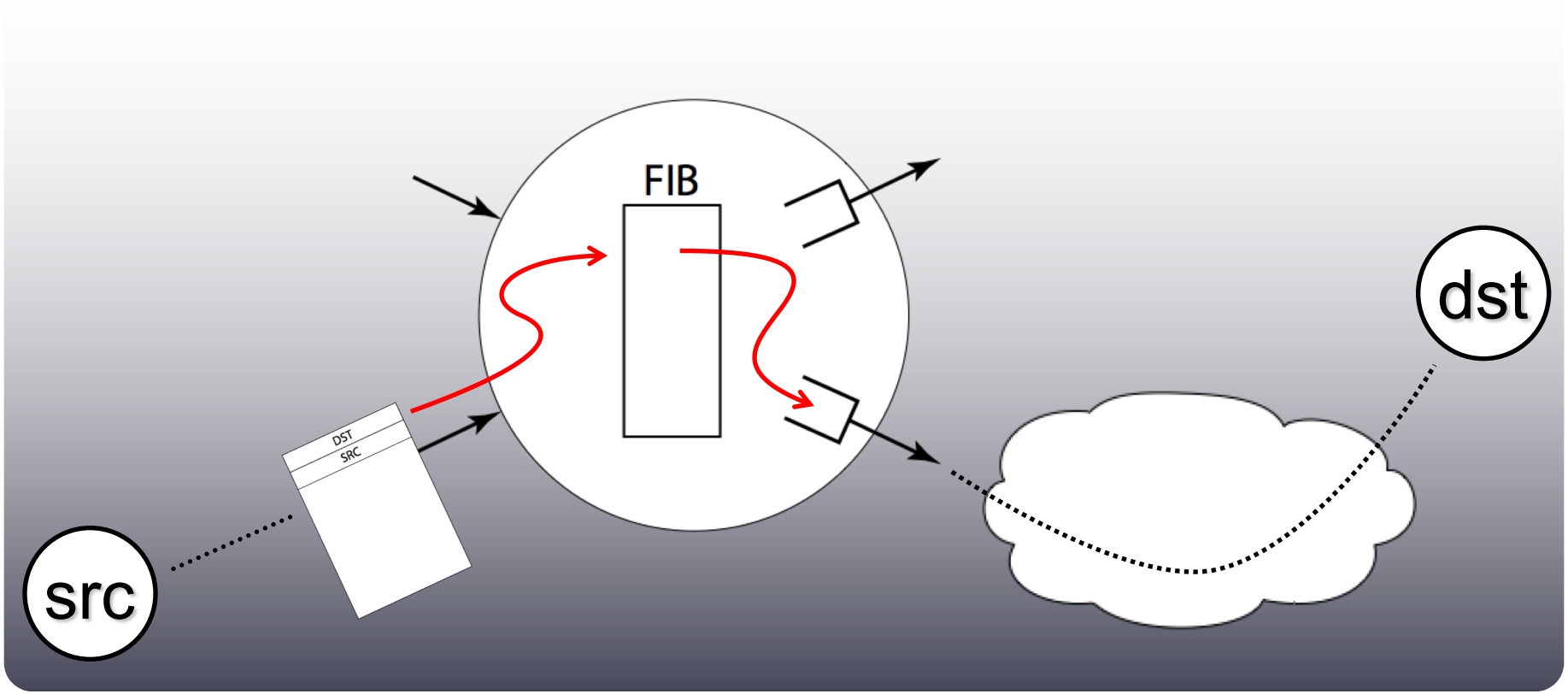


Named Data Networks (NDN)



- Key idea: routers conduct both routing and **content caching**

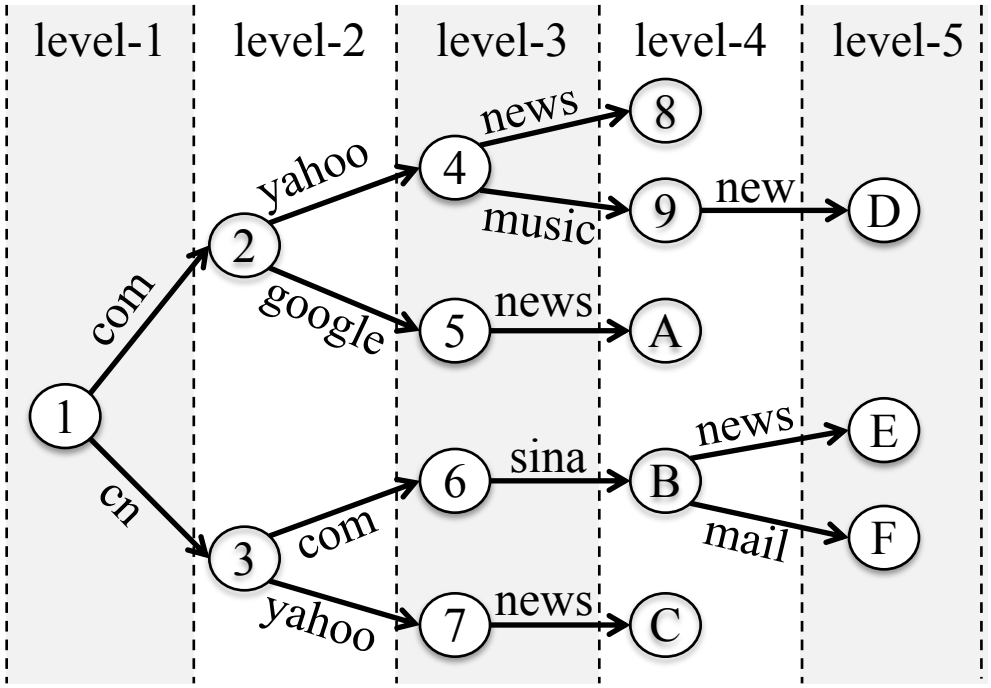
Router Today



- Content lookup: Google -> URL -> DNS -> IP
- Longest prefix match on IP address

Named Data Networks (NDN)

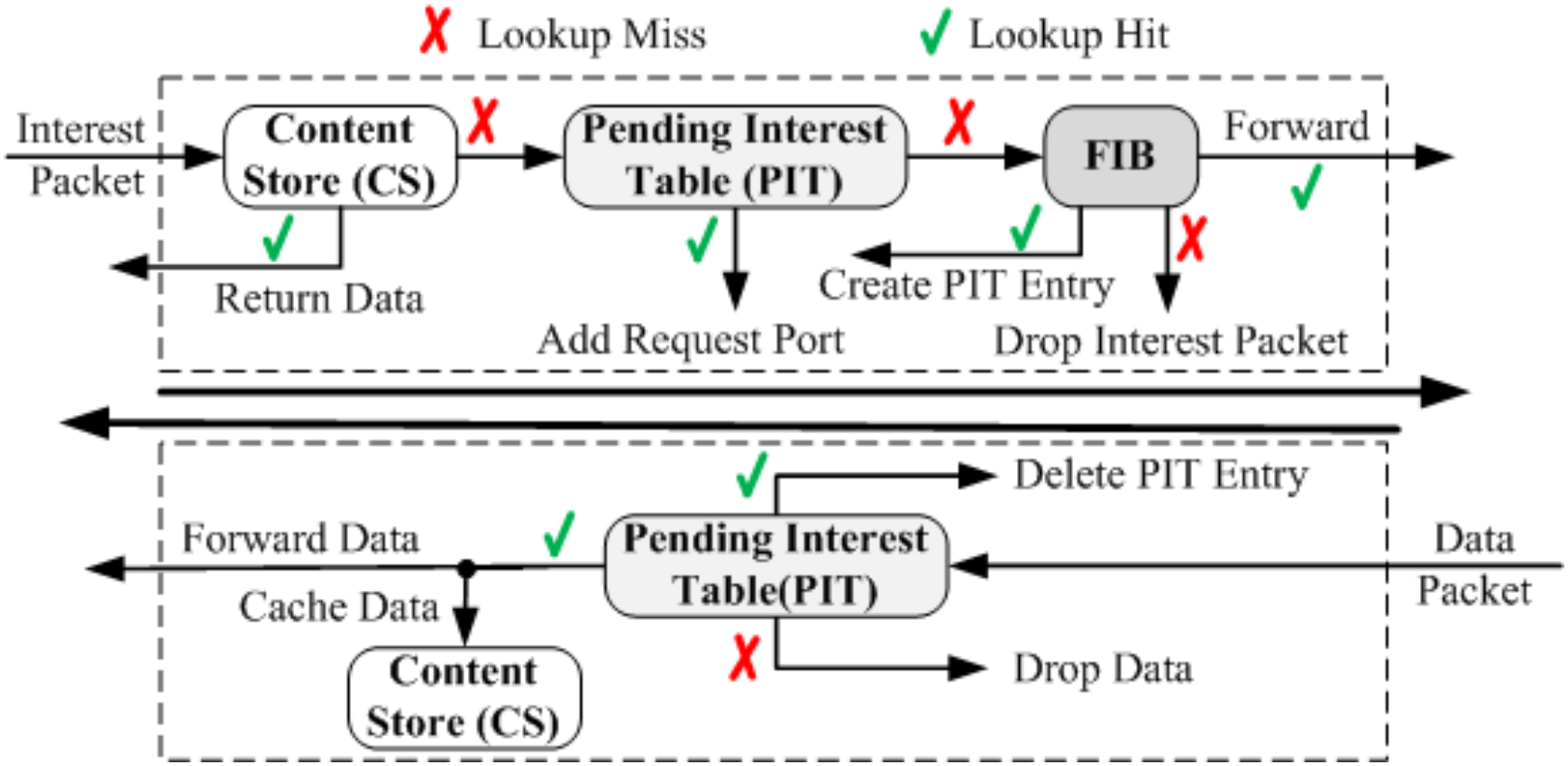
/com/yahoo/news
/com/yahoo/music/new
/com/google/news
/com/google
/cn/com/sina/news
/cn/com/sina/mail
/cn/yahoo/news



Name Prefix Trie (NPT)

Key idea 2: from routing on IP prefix to routing on Name prefix

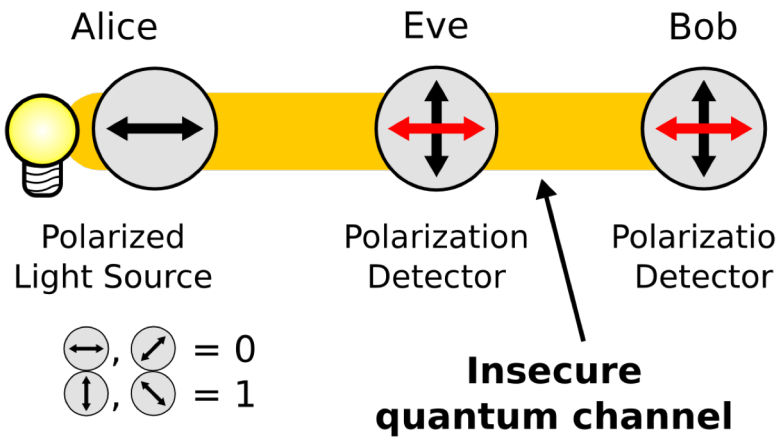
NDN Details



Do you buy this design?

More Possibilities

- More secure
  - Quantum networks
    - Key exchange through physical properties instead of computational hardness
- More economical
  - Cloud/edge integration
- ...

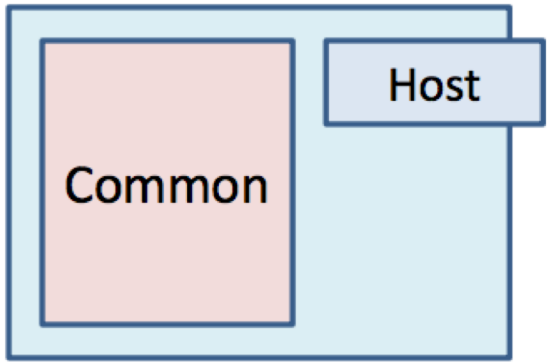


Source: https://en.wikipedia.org/wiki/Quantum_network#/media/File:BB84-network_setup.png

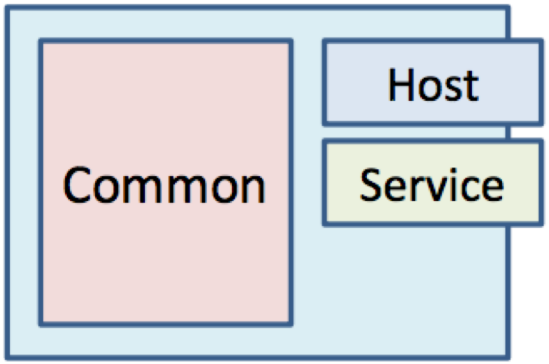
Due to quantum mechanics and the no-cloning theorem, Eve cannot eavesdrop and not being detected

Evolvability

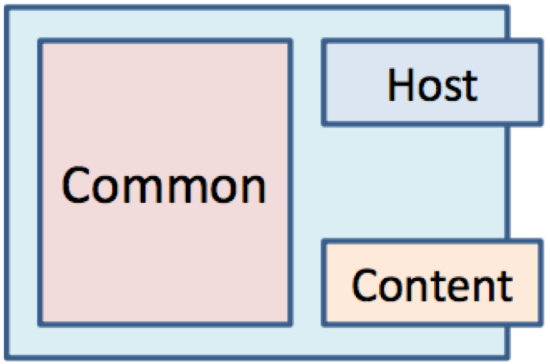
Routers w/ Evolving Capabilities



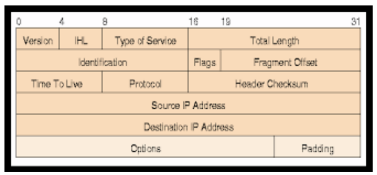
Host-only router

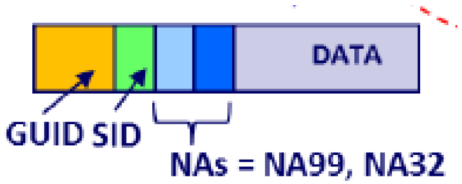


Service-enabled router

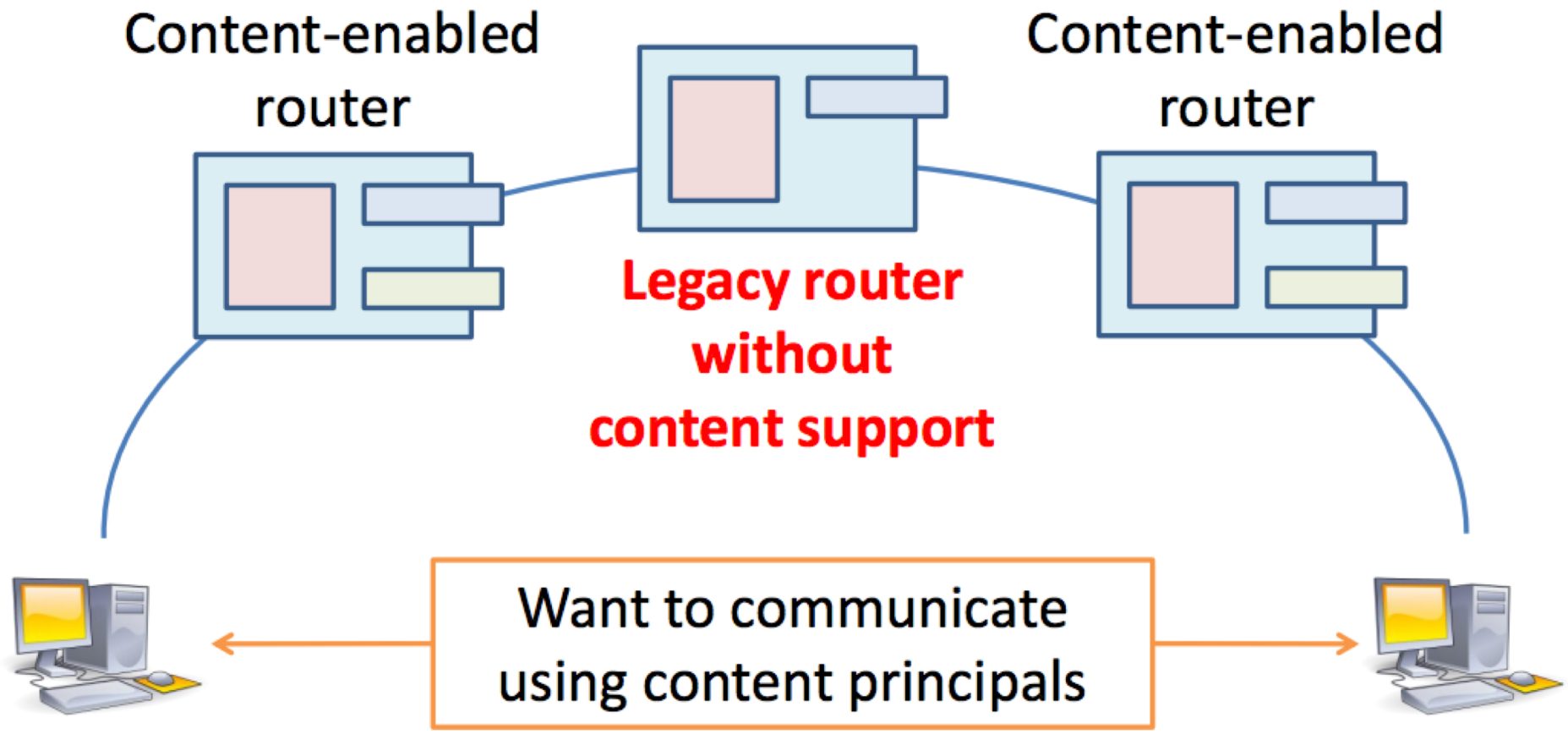


Content-enabled router





Routers w/ Evolving Capabilities



https://www.usenix.org/sites/default/files/conference/protected-files/xia-nsdi2012-public.pdf

Basic Idea: Fallback and Intent

Intent: Retrieve Content

Fallback: Contact Host ,

who understands Content request

What the network does:

- With content-enabled routers, use Content for routing
- Otherwise, use Host for routing (always succeeds)

Basic Idea: DAG Based Addresses