Topic 2 – Architecture and Philosophy

- Abstraction
- Layering
- Layers and Communications
- Entities and Peers
- What is a protocol?
- Protocol Standardization
- The architects process
  - How to break system into modules
  - Where modules are implemented
  - Where is state stored
- Internet Philosophy and Tensions

1

1

TRIGGER WARNING

- Philosophy,
- Bad Analogies, and
- RANTS verging
  on POLEMIC

Will follow.....

2

Abstraction Concept

A mechanism for breaking down a problem

what not *how*
- eg Specification *versus* implementation
- eg Modules in programs

Allows replacement of implementations without affecting system behavior

Vertical versus *Horizontal*

"Vertical" what happens in a box "How does it attach to the network?"

"Horizontal" the communications paths running through the system

Hint: paths are built ("layered") on top of other paths

3

3

Computer System Modularity

Partition system into modules & abstractions:
- Well-defined interfaces give flexibility
  - *Hides* implementation - can be freely changed
  - Extend functionality of system by adding new modules

- E.g., libraries encapsulating set of functionality

- E.g., programming language + compiler abstracts away how the particular CPU works …

4

4

Computer System Modularity (cnt'd)

- Well-defined interfaces hide information
  - Isolate assumptions
  - Present high-level abstractions

- **But can impair performance!**

- Ease of implementation vs worse performance

5

5

Network System Modularity
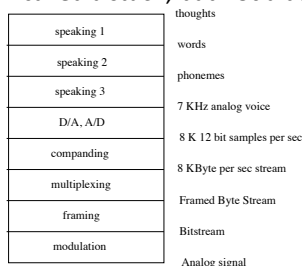
Like software modularity, but:
- Implementation is distributed across many machines (routers and hosts)
- Must decide:
  - How to break system into modules
    - **Layering**
  - Where modules are implemented
    - **End-to-End Principle**
  - Where state is stored
    - **Fate-sharing**

6

6

Layering Concept

- A restricted form of abstraction: system functions are divided into layers, one built upon another
- Often called a *stack*; but **not** a data structure!

	thoughts
speaking 1	words
speaking 2	phonemes
speaking 3	7 KHz analog voice
D/A, A/D	8 K 12 bit samples per sec
companding	8 KByte per sec stream
multiplexing	Framed Byte Stream
framing	Bitstream
modulation	Analog signal

7

Layers and Communications

- Interaction only between adjacent layers
- *layer n* uses services provided by *layer n-1*
- *layer n* provides service to *layer n+1*
- Bottom layer is physical media
- Top layer is application

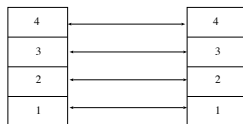| |
|---|
| n + 1 layer |
| n layer |
| n - 1 layer |
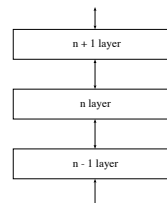
8

Entities and Peers

Entity – a *thing* (an independent existence)

Entities *interact* with the layers above and below

Entities *communicate* with *peer* entities

- same level but different place (eg different person, different box, different host)

Communications between peers is supported by entities at the lower layers

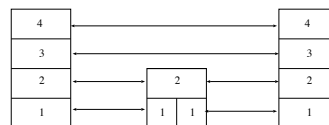| 4 | 4 |
3	3
2	2
1	1

9

Entities and Peers

Entities usually do something useful

- Encryption – Error correction – Reliable Delivery
- Nothing at all is also reasonable

Not all communications is end-to-end

Examples for things in the middle

- IP Router – Mobile Phone Cell Tower
- Person translating French to English

4			4
3			3
2	2		2
1	1	1	1

10

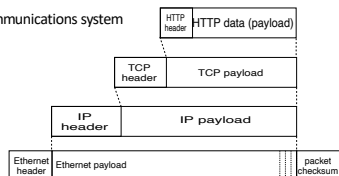
Layering and Embedding

In Computer Networks we often see higher-layer information embedded within lower-layer information
- Such embedding can be considered a form of layering
- Higher layer information is generated by stripping off headers and trailers of the current layer
- eg an IP entity only looks at the IP headers
    BUT embedding is not the only form of layering

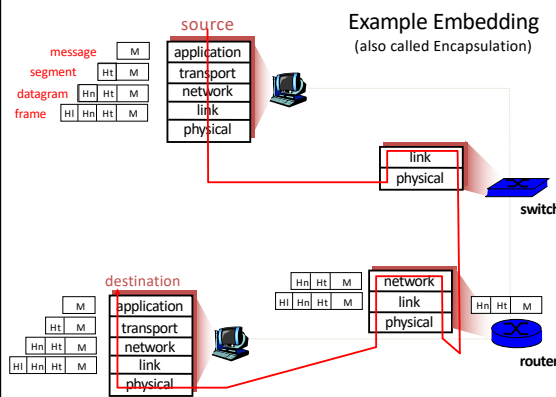Layering is to help understand a communications system
NOT
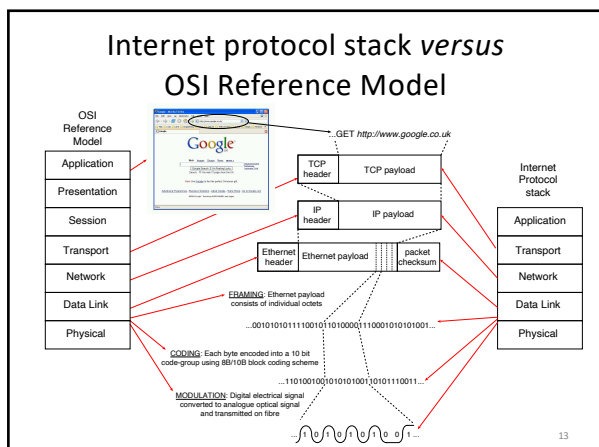determine implementation strategy

HTTP header	HTTP data (payload)

TCP header	TCP payload

IP header	IP payload

Ethernet header	Ethernet payload	packet checksum

11

Example Embedding
(also called Encapsulation)

source

message — M
segment — Ht M
datagram — Hn Ht M
frame — Hl Hn Ht M

application
transport
network
link
physical

link
physical

switch

network
link
physical

router

destination

application
transport
network
link
physical

M
Ht M
Hn Ht M
Hl Hn Ht M

Hn Ht M
Hl Hn Ht M

Hn Ht M

12

Internet protocol stack *versus* OSI Reference Model

13

ISO/OSI reference model

- **presentation:** allow applications to interpret meaning of data, e.g., encryption, compression, machine-specific conventions
- *session:* synchronization, checkpointing, recovery of data exchange
- Internet stack "missing" these layers!
  - these services, *if needed,* must be implemented in application

application
presentation
session
transport
network
link
physical

14

What is a protocol?

human protocols:
- "what's the time?"
- "I have a question"
- introductions

… specific msgs sent
… specific actions taken when msgs received, or other events

network protocols:
- machines rather than humans
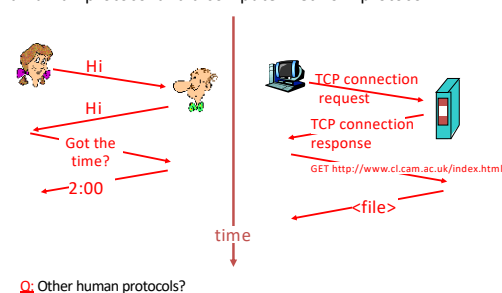- all communication activity in Internet governed by protocols

protocols define format, order of msgs sent and received among network entities, and actions taken on msg transmission, receipt

15

What is a protocol?

a human protocol and a computer network protocol:

Hi
Hi
Got the time?
2:00

TCP connection request
TCP connection response
GET http://www.cl.cam.ac.uk/index.html
<file>

time

Q: Other human protocols?

16

Protocol Standardization

- All hosts must follow same protocol
  - Very small modifications can make a big difference
  - Or prevent it from working altogether
- This is why we have standards
  - Can have multiple implementations of protocol
- Internet Engineering Task Force (IETF)
  - Based on working groups that focus on specific issues
  - Produces "Request For Comments" (RFCs)
  - IETF Web site is ***http://www.ietf.org***
  - RFCs archived at ***http://www.rfc-editor.org***
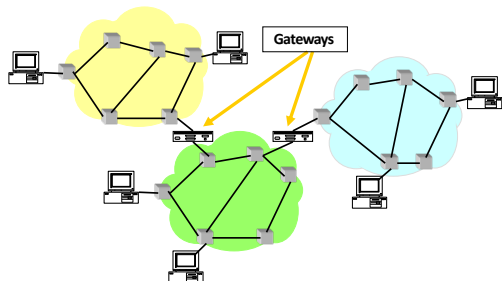
17

So many Standards Problem

- Many different packet-switching networks
- Each with its own Protocol
- Only nodes on the same network could communicate
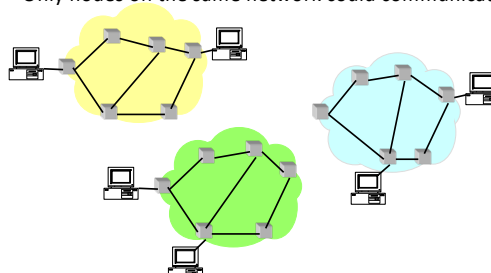
18

INTERnet Solution

Gateways

19

19

Internet Design Goals (Clark '88)

- **Connect existing networks**
- Robust in face of failures
- Support multiple types of delivery services
- Accommodate a variety of networks
- Allow distributed management
- Easy host attachment
- Cost effective
- Allow resource accountability
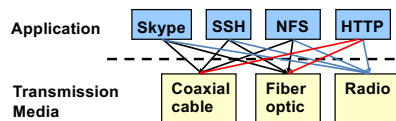
20

20

Real Goals

Internet Motto

We reject kings , presidents, and voting. We believe in rough consensus and running code." – David Clark

- **Build something that works!**
- Connect existing networks
- Robust in face of failures
- Support multiple types of delivery services
- Accommodate a variety of networks
- Allow distributed management
- Easy host attachment
- Cost effective
- ~~Allow resource accountability~~

21

21

A Multitude of Apps Problem

Application   Skype SSH NFS HTTP

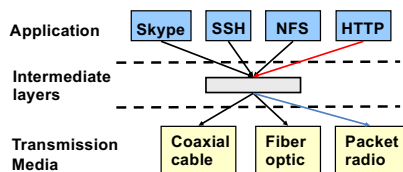**Transmission Media**   Coaxial cable Fiber optic Radio

- Re-implement every application for every technology?
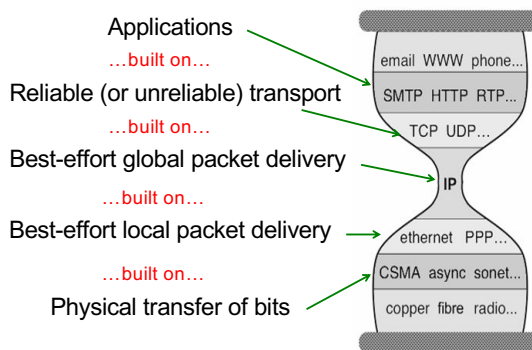- No! But how does the Internet design avoid this?

22

22

Solution: Intermediate Layers

- Introduce intermediate layers that provide set of abstractions for various network functionality and technologies
  - A new app/media implemented only once
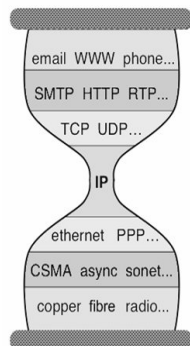 - Variation on "add another level of indirection"

Application   Skype SSH NFS HTTP

Intermediate layers

Transmission Media   Coaxial cable Fiber optic Packet radio

23

23

In the context of the Internet

Applications
…built on…
email WWW phone...

Reliable (or unreliable) transport
…built on…
SMTP HTTP RTP...

TCP UDP...

Best-effort global packet delivery
…built on…
IP

Best-effort local packet delivery
…built on…
ethernet PPP…

CSMA async sonet...

Physical transfer of bits
copper fibre radio...

24

Three Observations

- Each layer:
  - Depends on layer below
  - Supports layer above
  - Independent of others

- Multiple versions in layer
  - Interfaces differ somewhat
  - Components pick which lower-level protocol to use

- But only one IP layer
  - Unifying protocol

email WWW phone…
SMTP HTTP RTP…
TCP UDP…
IP
ethernet PPP…
CSMA async sonet…
copper fibre radio…

25

Layering Crucial to Internet's Success

- Reuse

- Hides underlying detail

- Innovation at each level can proceed in parallel

- Pursued by very different communities

email WWW phone…
SMTP HTTP RTP…
TCP UDP…
IP
ethernet PPP…
CSMA async sonet…
copper fibre radio…
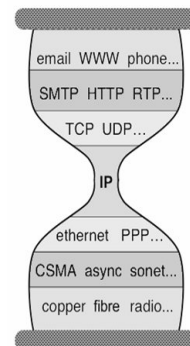
26

What are some of the drawbacks of protocols and layering?

27

Drawbacks of Layering

- Layer N may duplicate lower layer functionality
  - e.g., error recovery to retransmit lost data
- Information hiding may hurt performance
  - e.g., packet loss due to corruption vs. congestion
- Headers start to get really big
  - e.g., typical TCP+IP+Ethernet is 54 bytes
- Layer violations when the gains too great to resist
  - e.g., TCP-over-wireless
- Layer violations when network doesn't trust ends
  - e.g., firewalls

28

Placing Network Functionality

- Hugely influential paper: "End-to-End Arguments in System Design" by Saltzer, Reed, and Clark ('84)
  - articulated as the "End-to-End Principle" (E2E)

- Endless debate over what it means

- Everyone cites it as supporting their position
(regardless of the position!)

29

Basic Observation

- Some application requirements can only be correctly implemented **end-to-end**
  - reliability, security, *etc.*

- Implementing these in the network is hard
  - every step along the way must be fail proof

- Hosts
  - **Can** satisfy the requirement without network's help
  - **Will/must** do so, since they can't rely on the network

30

Example: Reliable File Transfer
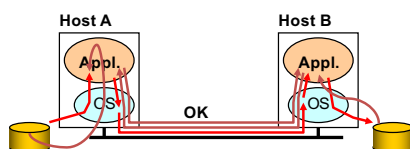
- Solution 1: make each step reliable, and string them together to make reliable end-to-end process
- Solution 2: end-to-end **check** and retry

31

31

Discussion

- Solution 1 is incomplete
  - What happens if any network element misbehaves?
  - Receiver has to do the check anyway!

- Solution 2 is complete
  - Full functionality can be entirely implemented at application layer with no need for reliability from lower layers

- Is there any need to implement reliability at lower layers?

32

32

Summary of End-to-End Principle

- Implementing functionality (e.g., reliability) in the network
  - Doesn't reduce host implementation complexity
  - Does increase network complexity
  - Probably increases delay and overhead on all applications even if they don't need the functionality (e.g. VoIP)

- However, implementing in the network can improve performance in some cases
  - e.g., consider a very lossy link

33

33

"Only-if-Sufficient" Interpretation

- Don't implement a function at the lower levels of the system unless it can be completely implemented at this level

- *Unless you can relieve the burden from hosts, don't bother*

34

34

"Only-if-Necessary" Interpretation

- Don't implement *anything* in the network that can be implemented correctly by the hosts

- Make network layer absolutely minimal
  - This E2E interpretation trumps performance issues
  - Increases flexibility, since lower layers stay **simple**

35

35

"Only-if-Useful" Interpretation

- If hosts can implement functionality correctly, implement it in a lower layer only as a performance enhancement
- But do so only if it does not impose burden on applications that do not require that functionality

36

36

We have some tools:

- Abstraction
- Layering
- Layers and Communications
- Entities and Peers
- Protocol as motivation
- Examples of the architects process
- Internet Philosophy and Tensions

37

37

Distributing Layers Across Network

- Layers are simple if only on a single machine
  - Just stack of modules interacting with those above/below

- But we need to implement layers across machines
  - Hosts
  - Routers (switches)

- What gets implemented where?

38

38

What Gets Implemented on Host?

- Bits arrive on wire, must make it up to application

- Therefore, all layers must exist at the host

source / destination

| M | application |
Ht	M	transport
Hn	Ht	M	network
Hl	Hn	Ht	M	link
physical
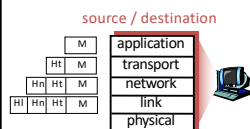
39

39

What Gets Implemented on a Router?

- Bits arrive on wire
  - Physical layer necessary

- Packets must be delivered to next-hop
  - Datalink layer necessary

- Routers participate in global delivery
  - Network layer necessary

- Routers don't support reliable delivery
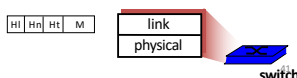 - Transport layer (and above) *__not__* supported
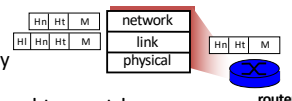
router

40

40

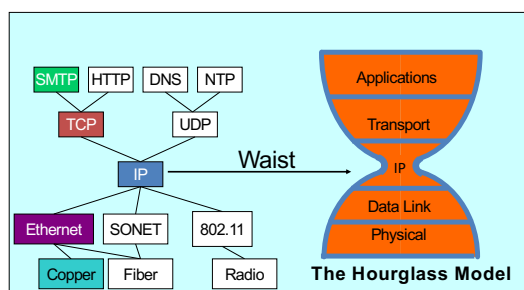## What Gets Implemented on Switches?

- Switches do what routers do, except they don't participate in global delivery, just local delivery

- They only need to support Physical and Datalink
  - Don't need to support Network layer

- Won't focus on the router/switch distinction
  - Almost all boxes support network layer these days
  - Routers have switches but switches do not have routers

switch

41

41

The Internet *Hourglass*

| SMTP | HTTP | DNS | NTP |
| TCP | | UDP |
| IP |

Waist

| Ethernet | SONET | 802.11 |
| Copper | Fiber | Radio |

Applications
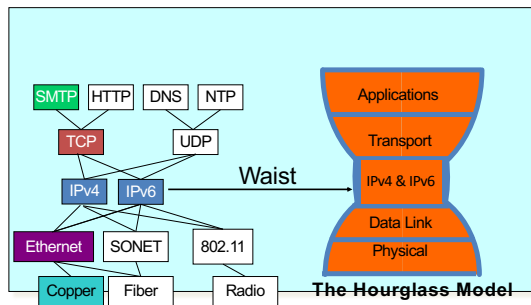Transport
IP
Data Link
Physical

The Hourglass Model

There is just one network-layer protocol, **IP**.
The "narrow waist" facilitates interoperability.

42

42

The middle-age Internet *Hourglass*

| SMTP | HTTP | DNS | NTP |

TCP   UDP

IPv4   IPv6   **Waist**

Ethernet   SONET   802.11

Copper   Fiber   Radio

The Hourglass Model

Applications

Transport

IPv4 & IPv6

Data Link

Physical

There is just ~~one~~ TWO network-layer protocol, **IP**v4 + v6
The "narrow waist" facilitates interoperability.

43

Alternative to Standardization?

- Have one implementation used by everyone

- Open-source projects
  - Which has had more impact, Linux or POSIX?

- Or just sole-sourced implementation
  - Skype, many P2P implementations, etc.

44