Communications Network Design
lecture 03

Matthew Roughan
<matthew.roughan@adelaide.edu.au>

Discipline of Applied Mathematics
School of Mathematical Sciences
University of Adelaide

October 27, 2011

Internet Design Principles

Most of this course is about mathematical optimization of networks. However, the term design covers a lot more than just deciding the capacity of links, etc. We need to know some of these more general principles

- so you understand basic terminology
- so you can appreciate what parts of a network are under your control and what forms "constraints"

This lecture covers Internet Design Principles and TCP/IP

Lecture goals/outline

- start to understand how computer networks work
- general principles behind computer networks
- TCP/IP
- general reference [1]

http://www.warriorsofthe.net/

Copyright 2002 Gunilla Elam, Tomas Stephanson, Niklas Hanberger.

The “Internet”

What is the Internet?

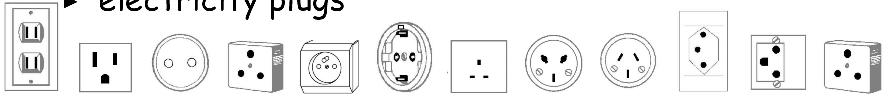
- physical infrastructure
- architecture
- protocols
- software
- services/applications
- operational practices
- standards

All of the above!

Standards

Why do we need standards

- electricity plugs



- plugs are standardized, but only within a country
- the “Internet” is an international network
  - need standards between countries
  - everyone has to agree on one “plug”
- instead of plugs we standardize **protocols**
  - still need plugs, but these are “physical layer”
  - a protocol is a more general concept

Network Standards Bodies

- ISOC (Internet Society)
  - IESG (Internet Engineering Steering Group)
    - **IETF** (Internet Engineering Task Force)
  - IAB (Internet Architecture Board)
    - IRSG (Internet Research Steering Group)
      - IRTF (Internet Research Task Force)
  - ICANN (Internet Corp. for Assigned Names and Numbers) and IANA (Internet Assigned Numbers Authority)

  `http://www.ietf.org/rfc/rfc3160.txt`
  `http://www.acm.org/ubiquity/views/v6i5_simoneli.html`
- W3C (WWW standards)
- IEEE (Inst. of Electrical and Electronic Engineers)
  - e.g. IEEE 802.3 (Ethernet)
- CCITT, ITU-T (International Telecommunications Union)
- ANSI, OSI (Open System Interconnection), OEOSC, ...

IETF

> 2.3 Dress Code
> Since attendees must wear their name tags, they must also wear shirts or blouses. Pants or skirts are also highly recommended.

Request for Comments: 3160
S. Harris, 2001

- http://www.ietf.org/
- informal standards body
  - membership is open to all interested **individuals**
  - few hard and fast rules
- publishes **RFCs** (Request For Comments)

RFCs

- **standards**: RFC 791: IP, RFC 793: TCP
- **best practice**: RFC 1818: Best Current Practices, RFC1918: Address Allocation for Private Internets
- **experimental**: RFC 2498: IPPM Metrics for Measuring Connectivity
- **informational**: RFC 3160: The Tao of IETF ... RFC 2151: A Primer On Internet and TCP/IP Tools and Utilities
- **humour**: RFC 1149: Standard for the transmission of IP datagrams on avian carriers
- **poetry**: RFC 1121: Act One - The Poems

http://www.ietf.org/rfc.html
http://www.rfc-editor.org/

Some Important RFCs

- RFC 791: Internet Protocol (IP)
  Updated in RFC 1391
- RFC 793: Transmission Control Protocol (TCP)
  Updated in RFC 3168
- RFC 1123: Requirements for Internet Hosts —
  Application and Support
  Updated by RFC1349, RFC2181
- RFC 2328: OSPF Version 2
- RFC 1771: A Border Gateway Protocol 4 (BGP-4)
- RFC 1772: Application of the Border Gateway
  Protocol in the Internet

Do I expect you to read all of this? The short answer is that you can pass the course without doing some side reading. However, if you want to do well, you will need to read outside of the course. I have some suggestions in places, but using your own initiative is always a good idea.

Internet Design Principles

- **packet switching not circuit switching:** Don't reserve bandwidth for a connection.
- **robustness principle:** Be liberal in what you accept, and conservative in what you send [2, 3].
- **layered model:** with a thin waist.
- **end-to-end principle:** Smart terminals, dumb network [4, 5].
- **distributed control:** as compared to centralized, or decentralized [5].
- **deployment issues:** scale, incremental deployment, heterogeneity [3].
- **general issues:** simplicity, modularity, performance [3].

There are also design guidlines in section 3 of ftp://ftp.isi.edu/in-notes/rfc1958.txt, but more like coding style. e.g.

3.7 In many cases it is better to adopt an almost complete solution now, rather than to wait until a perfect solution can be found.

3.10 Be parsimonious with unsolicited packets

3.11 Circular dependencies must be avoided.

3.12 Objects should be self decribing

3.14 And perhaps most important: Nothing gets standardised until there are multiple instances of running code.

Section 5 of ftp://ftp.isi.edu/in-notes/rfc1958.txt is really about policy: e.g. patents, etc.

Packets vs circuits

- Bell-heads vs Net-heads
  - Bell-heads from old Bell system (AT&T included)
  - Net-heads: new generation, who grew up on the ARPANET/Internet
- Bell-heads believe you need a dedicated circuit
  - like a phone line (but higher speed)
  - said the ARPANET would never work
- Net-heads think circuits are a waste of time
  - poor use of resources when traffic is bursty [6].
  - invented the ARPANET/Internet
- this is a theological debate

`http://www.wired.com/wired/archive/4.10/atm.html`

Packets vs circuits

- **Circuit switching:** logical equivalent of a phone line connects two (or more) people.
  - allows **network** to control everything (in theory)
  - allows explicit QoS
    - ⋆ needs careful design and admission control
  - prime example is ATM
- **Packet switching:** no logical circuit (though there is still an analogue of a connection). Packets of data are individually switched.
  - **network** doesn't do much (in theory)
  - hard to do QoS, but network is simpler
  - prime example is IP

Packets vs circuits

Doesn’t have to be one or the other

- we’ll see layering later
  - people may run ciruit switched on one layer, and packet switched on another.
  - classic example is IP over ATM
- MPLS creates virtual circuits between end-points
  - connections are not between end-users though
  - allows **multiplexing** of traffic inside a connection
  - multiplexed traffic is less bursty

Packets vs circuits

ROSENCRANTZ AND ETHERNET by Vint Cerf [7]

All the world’s a net! And all the data in it merely packets
come to store-and-forward in the queues a while and then are
heard no more. ’Tis a network waiting to be switched!

To switch or not to switch? That is the question. Whether
’tis wiser in the net to suffer the store and forward of
stochastic networks or to raise up circuits against a sea
of packets and, by dedication, serve them.

To net, to switch. To switch, perchance to slip!
Aye, there’s the rub. For in that choice of switch,
what loops may lurk, when we have shuffled through
this Banyan net? Puzzles the will, initiates symposia,
stirs endless debate and gives rise to uncontrolled
flights of poetry beyond recompense!

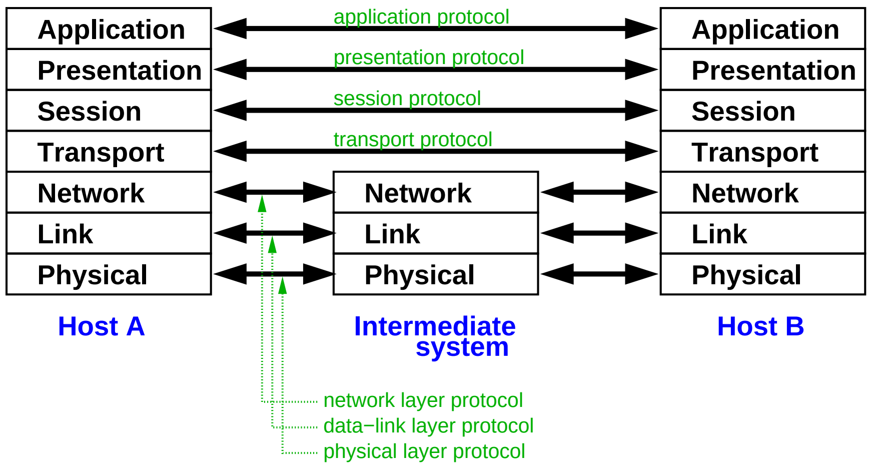
Robustness principle

Be liberal in what you accept, and conservative in what you send [2].

- if some-else's software screws up, don't let this mess your system up (liberal in what you accept)
  - e.g. TCP connection termination
- don't cause other systems problems (conservative in what you send)
  - e.g. congestion control

Layered protocols: OSI model

OSI model breaks functionality into layers called a **protocol stack**



Application
Presentation
Session
Transport
Network
Link
Physical
Host A

application protocol
presentation protocol
session protocol
transport protocol

Network
Link
Physical
Intermediate system

Application
Presentation
Session
Transport
Network
Link
Physical
Host B

network layer protocol
data−link layer protocol
physical layer protocol

Additional references:

```
http://www.geocities.com/SiliconValley/Monitor/3131/ne/osimodel.html
http://www2.rad.com/networks/1994/osi/layers.htm
http://www.webopedia.com/quick_ref/OSI_Layers.asp
http://en.wikipedia.org/wiki/OSI_model
```

Layered protocols: OSI model

- Somewhat like subroutines in programming
  - Each layer provides **services** (functions) to higher layers
  - Function call **interface** hides details of how the service is provided
  - e.g. network layer asks link layer to transport a packet across a link, without any network details
  - the **interface** is well defined
- Benefits
  - reduction in complexity
  - reuse of functionality
    - ⋆ may be many applications on one session layer
- Communications between peers using **protocols**

Encapsulation

Lower layers deal with higher layer by

- treat information from higher layer as “black box”.
  - don’t look inside data
  - just treat as bunch of bits
- allowed operations on the data
  - just break data into blocks
  - **encapsulate** the blocks, by adding
    - ⋆ headers (e.g. addresses)
    - ⋆ trailers
- when passing back to higher
  - layers strip headers
  - join blocks back together

Layer 1: Physical layer

Function: Transmission of raw bit stream between devices.
Services: Physical connection, Binary modulation, frequency, ...
Issues: # pins/wires, duplex, serial/parallel, modulation, ...
Media:

- copper wire: e.g. coax, twisted pair (CAT-3/CAT-5), RS-232, USB, firewire
- lasers (fibre optics)
- lasers (free air)
- microwave, RF, satellite, ...
- infra-red
- carrier pigeons (RFC 1149) ;-)

Layer 2: Data-link layer

Function: provide reliable transport of information between a pair of adjacent nodes.
Services: creates frames/packets, error control, flow control
Issues: Medium Access Control (MAC), headers/trailers, ...
Examples:

- Ethernet
- Token-ring
- IEEE 802.11 (Wi-Fi)
- FDDI (Fiber Distributed Data Interface)
- ATM (Asynchronous Transfer Mode) (also layer 3)
- POS (Packet over SONET)
- PPP (Point to Point Protocol)

Layer 3: Network layer ***

Function: forwarding packets from end-to-end
Services: packet forwarding, some congestion control
Issues: determining what routing to use
Examples:

- **IPv4 (Internet Protocol version 4)**
- IPv6 (Internet Protocol version 6)
- ARP (Address Resolution Protocol)
- ATM (Asynchronous Transfer Mode) (also layer 2)
- Routing protocols (e.g. OSPF, IS-IS, RIP, EIGRP)

*** — this is the bit we care about most here!

Layer 4: Transport layer

Function: reliable end-to-end transport of data
Services: multiplexing, end-to-end error and flow control
Issues: congestion control algorithm
Examples:

- TCP (Transmission Control Protocol)
- UDP (User Datagram Protocol)
- SCTP (Stream Control Transmission Protocol)
- RTP (Real-time Transport Protocol)

Layer 5: Session layer

Function: combine logically connected transmissions
Services: group several connections into a session
Issues: what to use it for?
Examples:

- NFS = Network File System
- SMB = Server Message Block

Layer 6: Presentation layer

Function: specific regularly requested functions.
Services: encryption, compression, ascii<->unicode, ...
Issues: want to do compression before encryption, but compression may be done by a lower layer.
Examples:

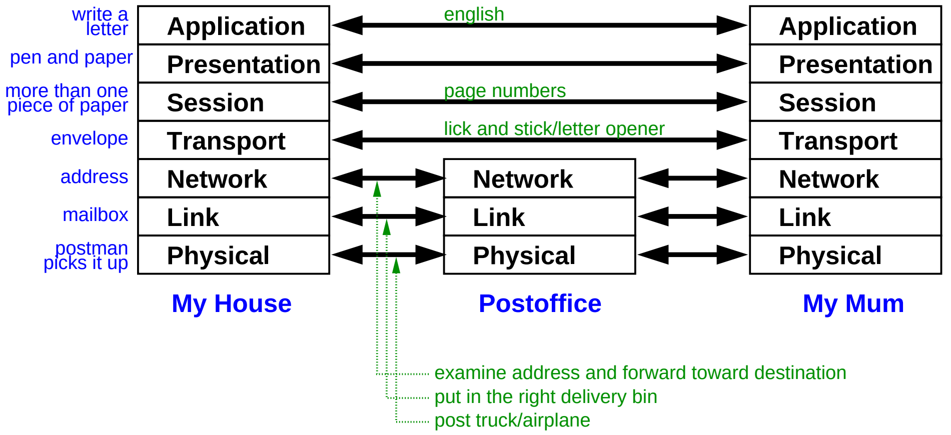
- SSL (Secure Sockets Layer) (at a stretch)

Layer 7: Application layer

- E-mail (POP, IMAP, SMTP)
- File transfer (FTP — File Transfer Protocol)
- Remote terminal (Telnet, SSH, ...)
- WWW (HTTP — Hyper-Text Transfer Protocol)
- File sharing (Gnutella, Napster, Kazaa, ...)
- Video conferences
- Newsgroups
- NTP (Network Time Protocol)
- VoIP (Voice over IP)
- Games (Quake, MMORP, ...)
- RFC 2324: Hyper Text Coffee Pot Control Protocol (HTCPCP/1.0)

Post office analogy

We could describe snail-mail using OSI model
e.g. sending mail to my mum.



write a letter
pen and paper
more than one piece of paper
envelope
address
mailbox
postman picks it up

Application
Presentation
Session
Transport
Network
Link
Physical

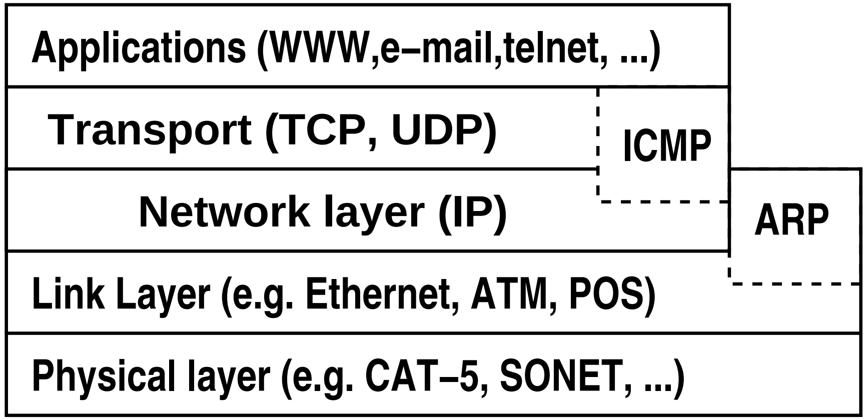
english
page numbers
lick and stick/letter opener

My House
Postoffice
My Mum

examine address and forward toward destination
put in the right delivery bin
post truck/airplane

Another example “James Bond Meets The OSI Model” is given at
`http://www.lewistech.com/rlewis/Resources/JamesBondOSI2.aspx`

TCP/IP has 5 “layers”



Applications (WWW,e−mail,telnet, ...)

Transport (TCP, UDP)

ICMP

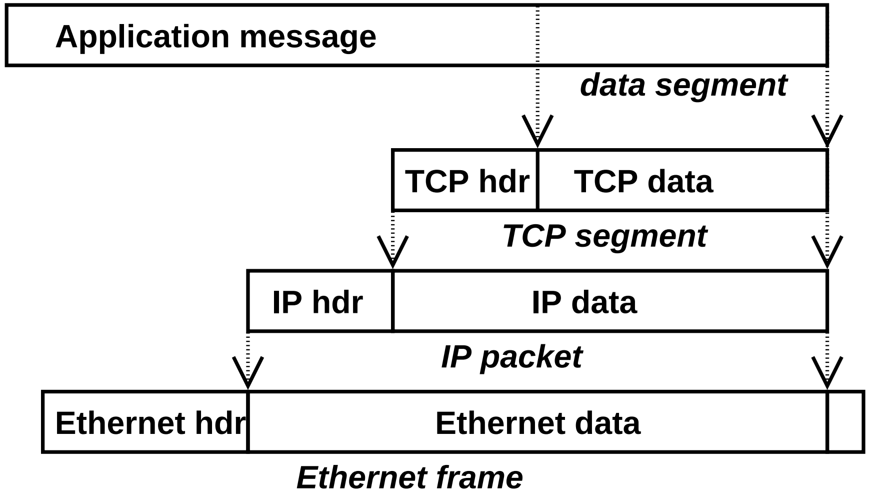
Network layer (IP)

ARP

Link Layer (e.g. Ethernet, ATM, POS)

Physical layer (e.g. CAT−5, SONET, ...)

TCP/IP Encapsulation



Application message

data segment

TCP hdr

TCP data

TCP segment

IP hdr

IP data

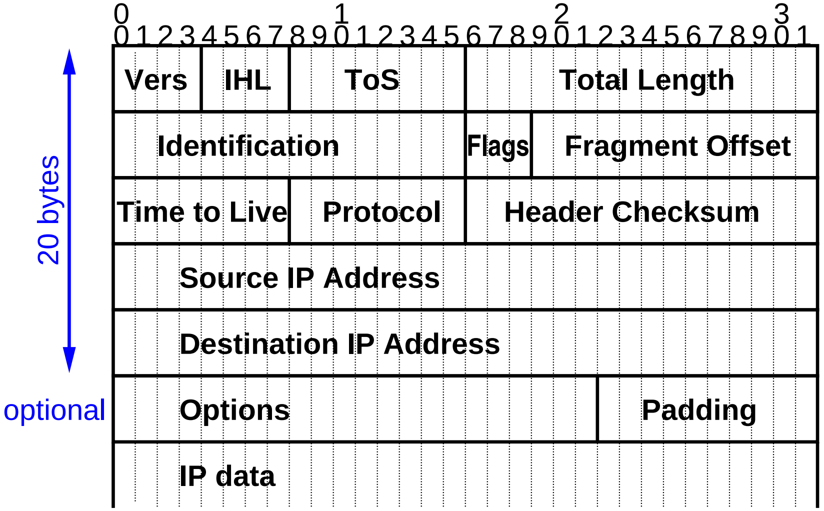
IP packet

Ethernet hdr

Ethernet data

Ethernet frame

IP header



0 1 2 3

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

Vers | IHL | ToS | Total Length

Identification | Flags | Fragment Offset

Time to Live | Protocol | Header Checksum

Source IP Address

Destination IP Address

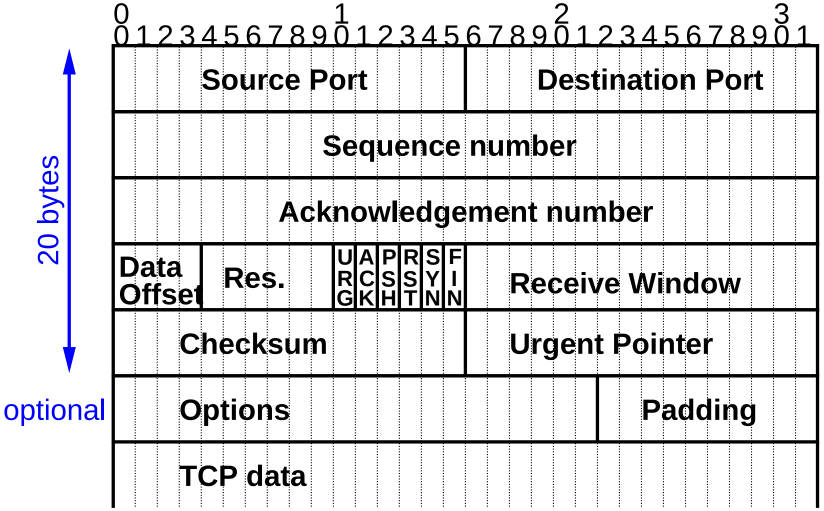
Options | Padding

IP data

20 bytes

optional

TCP header



0 1 2 3

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

Source Port | Destination Port

Sequence number

Acknowledgement number

Data Offset | Res. | URG | ACK | PSH | RST | SYN | FIN | Receive Window

Checksum | Urgent Pointer

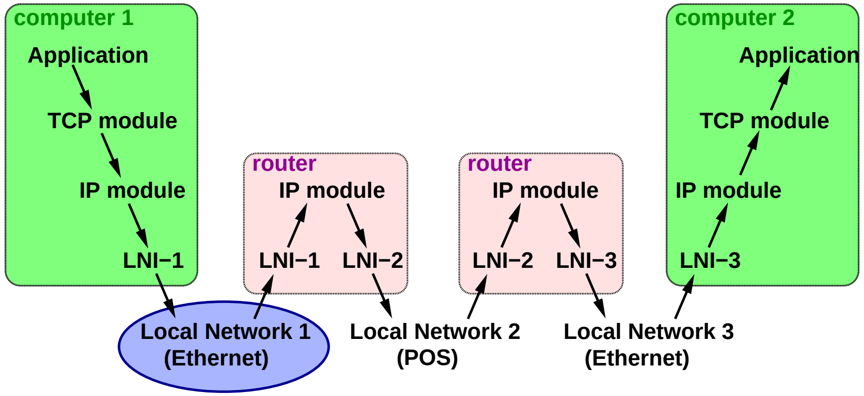
Options | Padding

TCP data

20 bytes

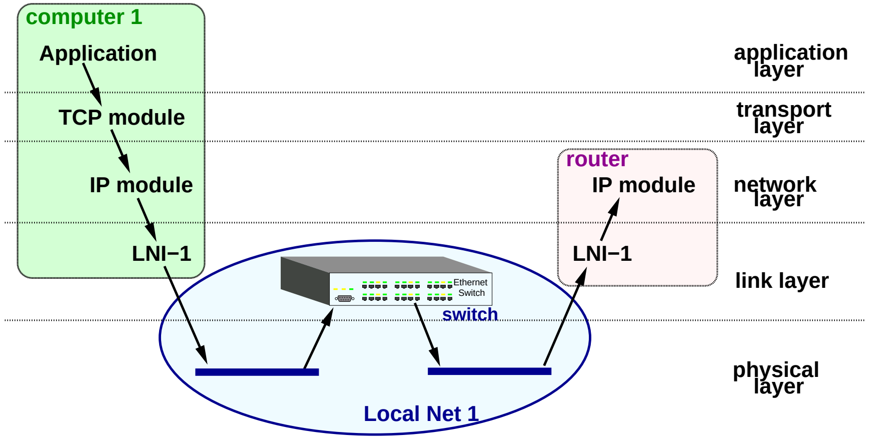
optional

TCP/IP operation



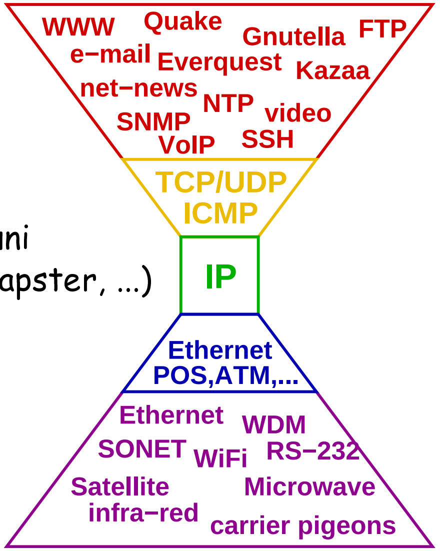
LNI = Local Network Interface

TCP/IP operation



Narrow Waist of IP: hourglass

- robustness against technological innovations
- anyone can innovate at either end
  - new applications built by uni students (e.g. netscape, napster, ...)
  - new physical/link layers
- allows huge heterogenity
- = success



Broken layering

TCP/IP layers are broken more often than not

- ICMP – uses IP, but controls its operation
- BGP is a routing protocol (IP layer), but is routed
- IP over ATM over IP over ATM over SONET
- anything involving MPLS
- often services are provided at multiple layers: error and flow control, e.g. error control in SONET (sort-of physical), link layer, IP, TCP, ...

OSI standards are too complicated

End-to-end principle

Put functionality as high up the stack as possible [4].

- pushes functionality out towards the end points (logical as well as physical)
- avoid repeating functionality
- more efficient in many cases anyway
- results in

  dumb network, smart terminals
- contrast to PSTN (Telephone Network)

  smart network, dumb terminals
- also allows survival of partial network failures
  - e.g. link failure, we can reroute
  - if we avoid state in the network

Distributed control

- anything centralized is vulnerable
- don't just distribute physical infrastucture
- also distribute network control
- e.g. routing protocols
  - OSPF, IS-IS, BGP, ...
  - we will see more on these later
- not everything can be completely decentralized
  - e.g. NOC, NCC
  - still can provide redundancy

Deployment issues

- **scale**: has to work for a large range of networks (in distance, and number of hosts).
  - IP creates "networks of networks", that can span any scale: <1m — $>10,000$ km; 1 — $>10^9$ hosts; link speeds 9600 bps — 40 Gbps.
- **incremental deployment**: need to be able to deploy gradually.
  - constant change in the network
  - legacy networks won't go away
- **heterogeneity**: different technologies and applications and link speeds.
  - see layers 1-2 and 7 above.
  - link speeds covering 8 orders of magnitude.

Network scale

Geographic scale

- LAN = Local Area Network (one building)
  - Ethernet (vast majority), Token ring, Wi-Fi, ...
- CAN = Campus Area Network (one campus)
- MAN = Metropolitan Area Network (one city)
- WAN = Wide Area Network (bigger than one city)
  - the Internet (best known), Frame relay, ATM, ...

Number of routers/switches (my classification)

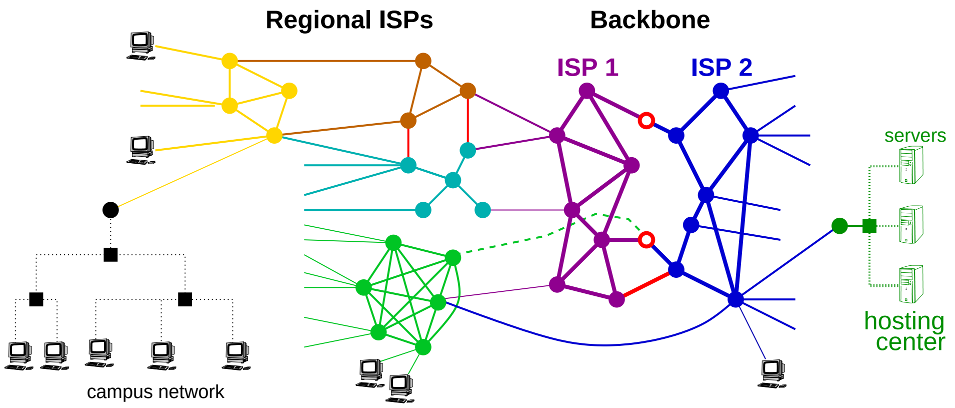
- small < 10
- medium 10-100
- large > 100

IP is not all good

Some things IP does not do well

- billing
  - circuits are easy to bill
  - packets are not
    - most Internet charging is flat rate
    - Australia is an exception
  - does this really matter? [8].
- QoS (Quality of Service)
  - e.g. network not design to provide voice quality
  - maybe it can be fixed? Lots of research.
- security (crypto doesn't fix DoS, Worms, ...)

Layer 3 view of the Internet



🟥 peering link

🟪🟦 backbone links

🟩 regional ISP links

┈ LAN

○ exchange point

● ● backbone routers

●●●●● other routers

■ ■ switches

Addresses

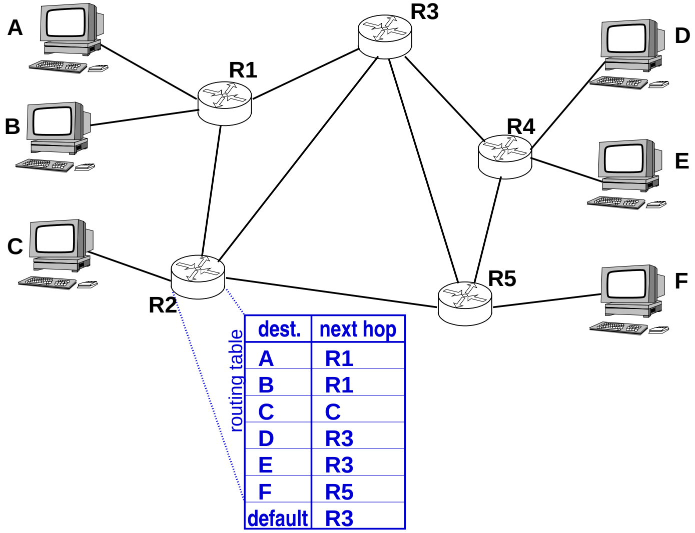
- IPv4 addresses, 32 bit, written as X.X.X.X
  - each chunk is a 8 bit number (0-255)
  - e.g. 10.1.2.255
  - subnet = group of IP addresses with a common **prefix**
    - e.g. private addresses 192.168.0.0/16
      - all address with same first **16** bits 192.168
      - 192.168.0.0 - 192.168.255.255
- IPv6 addresses, 128 bits — see [9]

Routing vs Forwarding

We need to make a distinction

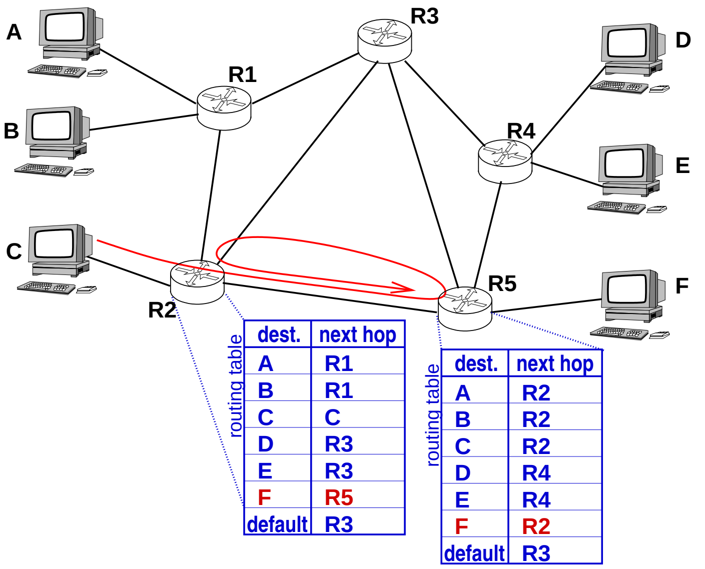
- **routing** = routing protocols build the routing tables
  - distributed routing protocol
  - build forwarding/routing/lookup table
- **forwarding** = send packets to their next hop
  - lookup destination address in forwarding table
  - forward packet

Forwarding



dest.	next hop
A	R1
B	R1
C	C
D	R3
E	R3
F	R5
default	R3

Route Loop



dest.	next hop
A	R1
B	R1
C	C
D	R3
E	R3
F	R5
default	R3

dest.	next hop
A	R2
B	R2
C	R2
D	R4
E	R4
F	R2
default	R3

References

[1] A. S. Tanenbaum, Computer Networks. Prentice-Hall, 3rd ed., 1996.

[2] R. Braden, "Requirements for Internet hosts – application and support." IETF, Request for Comments: 1123, 1989.

[3] B. Carpenter, "Architectural Principles of the Internet." IETF, Request for Comments: 1958, 1996.

[4] J. H. Saltzer, D. P. Reed, and D. D. Clark, "End-to-end arguments in system design," in Second International Conference on Distributed Computing Systems, pp. 509–512, April 1981.

[5] D. Clarke, "The Design Principles of the DARPA Internet Protocols," Computer Communication Review, vol. 25, January 1995.

[6] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of Ethernet traffic (extended version)," IEEE/ACM Transactions on Networking, vol. 2, pp. 1–15, Feb 1994.

[7] J. Postel, L. Kleinrock, V. Cerf, and B. Boehm, "Act One - The Poems." IETF, Request for Comments: 1121, 1989.

[8] A. M. Odlyzko, "The history of communications and its implications for the Internet." http://www.dtc.umn.edu/~odlyzko/doc/networks.html.

[9] S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6)." IETF, Request for Comments: 2460, 1998.