Internet Design Principles and Architecture

Venkat Padmanabhan
Microsoft Research
2 April 2001
Lecture Outline

• A brief history of the Internet
• How is the Internet different from the telephone network (and why)?
• Design goals and principles
• End-to-end (E2E) argument
• Rethinking the principles

• Overview of projects
A Brief History of the Internet

Two motivations
• sharing of expensive computing resources
• robust communication infrastructure

Timeline
• 1961: packet switching invented
• 1969: ARPAnet born
• Early 70s: TCP/IP designed
• 1983: NCP → TCP/IP transition
• 1986: NSFNET backbone created
• 1995: full transition to commercial Internet
POTS versus Internet

Many similarities

• Large scale (1B+ phones, 100M+ hosts)
• Global coverage
• Hierarchical addressing
• Link heterogeneity
POTS versus Internet

Main difference
• single-function versus multi-function which leads to several others
• circuit switching versus packet switching
• intelligence in the core versus in the edges
POTS

Several factors simplify engineering
• one service model: voice
• constant bit rate (64 Kbps)
• traffic engineering using models of voice calls
• flow control and error recovery by humans
• congestion control via busy signals
but
• tight latency constraint (100-200 ms)
Internet

Several complicating factors

• multitude of applications
• bursty traffic
• no universal model for traffic engineering
• protocols need to do flow control, congestion control, and loss recovery

but

• less stringent requirements ("best effort")
Internet Architecture Goals (Clark88)

Main goal: multiplexed utilization of existing interconnected networks

Secondary goals
• survivability in the face of failure
• support for multiple types of service
• support for a variety of networks
• distributed management of resources
• cost effectiveness
• easy addition of new hosts
• accounting of resource usage
Design Decisions

- Packet switching as basis for multiplexing
- Store-and-forward gateways as basis for interconnection
- "Fate-sharing" model for reliability
- Layering
- Minimal assumption about the network
Packet Switching

- Invented by Baran, Kleinrock et al. in early 60s
- Radical departure from circuit switched model
- Far more efficient for data communication since it is bursty (peak >> average)
- Key idea: statistical multiplexing
  - share on demand
  - based on statistics of offered load rather than a fixed offered load
Statistical Multiplexing: Example

• One user sends at 1 Mbps and is 90% idle
  - 10 Mbps channel; 10 users if statically allocated

• For 35 users, prob(>10 active) = 0.17%
Statistical Multiplexing

• Occasional oversubscription
  - need for buffering inside the network
  - need for loss recovery
  - need for congestion control

• How much statistical multiplexing is there in the Internet?
Network Interconnection

• Minimal assumptions made about the individual networks
  - ability to transport a datagram of a certain minimum size
• Networks interconnected by a layer of gateways
• IP is the common glue
• Intelligent end-points do the rest
  - scalable
  - flexible
Layering

• Need abstractions to handle complexity
• A protocol layer
  - implements a fixed set of functions
  - exposes a well-defined interface to other layers
• Good design principle but not always ideal for implementation
OSI Reference Model

- Seven Layers

<table>
<thead>
<tr>
<th>Layer</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>- application-dependent</td>
</tr>
<tr>
<td>Presentation</td>
<td>- Encode/decode messages</td>
</tr>
<tr>
<td>Session</td>
<td>- Manage connections</td>
</tr>
<tr>
<td>Transport</td>
<td>- Reliability, congestion control</td>
</tr>
<tr>
<td>Network</td>
<td>- Routing</td>
</tr>
<tr>
<td>Link</td>
<td>- Framing, multiple access</td>
</tr>
<tr>
<td>Physical</td>
<td>- Symbol coding, modulation</td>
</tr>
</tbody>
</table>

- Their functions:
Internet Protocol Stacks

Model

Application
Transport
Network
Link

Protocols

Many (HTTP, SMTP)
TCP / UDP
IP
Many (Ethernet, ...)

Many
Supporting Multiple Types of Service

- Original Internet protocols (Cerf & Kahn 74)
  - TCPIP was one protocol
  - addressing + “virtual circuit” service
- But this was sub-optimal
  - service abstraction not suitable for all applications (e.g., packet voice)
- Layering to the rescue
  - TCP and IP were split up into separate protocols
  - UDP created to provide datagram service
Survivability

- Continued operation in the face of network and gateway failures
- Only failure on top of transport layer is total partition
- There has *hard* state that must be protected
- “Fate-sharing” model: OK to lose state information if entity also lost
- So hard state stored in hosts, not switches
- End-to-end principle generalizes this idea
End-to-End Principle (Saltzer, Reed, Clark 84)

- Articulation of conventional wisdom in system design
- An argument against low-level function implementation if completeness and correctness require participation of endpoints
- Low-level function implementation may sometimes be warranted as a performance enhancement
  - soft state versus hard state
- Layering is a consequence of the E2E principle
Careful File Transfer

- **Goal:** reliable transmission of a file
- **Threats**
  - corruption/loss at endpoints
  - corruption/loss within the network
  - host crash
- **Error recovery within the network would be**
  - inefficient: not needed for all applications
  - incomplete: doesn’t address end-point failure
- **Preferred approach:** end-to-end check and retry
E2E and Wireless Links

• Wireless links tend to be error prone
• E2E retransmissions may be expensive
• So link-level ARQ is commonly used
• E2E still needed no matter how good link-level ARQ is
• Danger: competing retransmissions
Other Examples

- Secure data transmission
  - WEP a bad idea?
- Duplicate message suppression
  - sequence numbers
- FIFO message delivery
  - TCP fast retransmission allows some slack
- More?
E2E and Thin Clients

Proxies to integrate thin clients into the Internet

Pros:
- content transformation
- performance benefit (e.g., caching)

Cons:
- end-to-end transformation may be better
  - Web page author is one “end”
- encryption might shut out proxies
Internet Architecture Goals Revisited

- survivability in the face of failure
- support for multiple types of service
- support for a variety of networks
- distributed management of resources
- cost effectiveness
- easy addition of new hosts
- accountability of resource usage
What if the priorities were different?

- **Accountability of resources**
  - billing database attached to each router
  - digitally signed packets
- **Easy attachment of hosts**
  - dumb hosts and smart switches
  - DHCP: greater dependence on the network
- **Low cost**
  - uniform networks
Rethinking Layering
(Clark, Tennenhouse 90)

• Separation between layers may not be as clean as we would like

• Basic problem: layering may not be the most effective modularity for implementation

• Example #1: data manipulation functions
  - copying data, buffering, encryption, formatting
  - inefficient to do these in different layers

• Example #2: lost and mis-ordered data
  - presentation formatting can happen in parallel with loss recovery
Two New Principles

• Application-level Framing (ALF)
  - lower layers should deal with data units that the application specifies (ADUs)
  - is P-HTTP a good idea?

• Integrated Layer Processing (ILP)
  - there may be ordering constraints
  - need well-designed ADUs

• Used in actual applications
  - example: scalable reliable multicast
Structure of the Internet

You at work → Large corporation

Backbone service provider

You at home

“Consumer” ISP

Large corporation

“Consumer” ISP

Small corporation

Peering point

Peering point
Practical Issues

• Interconnection
  - competitors are forced to cooperate
• Client-provider versus peer-peer
  - who is providing more value?
• Settlements
Recap

• Key features of the Internet
  - multi-function
  - heterogeneous networks
  - intelligence at the edges
• End-to-end principle
  - fate sharing
• Internet’s design is a reflection of its priorities
Next Lecture


• Brush up on routing basics