CSE 534
Design Philosophy
End-to-End Arguments

Vyas Sekar
Stony Brook University

Based in part on content borrowed from:
Srini Seshan, Nick McKeown
Logistics Clarifications

- Reviews to be submitted online
- Reviews to be submitted BEFORE class
- Syllabus will be updated as we go
Design Considerations

• Assigned Reading
  – [SRC84] End-to-end Arguments in System Design
  – [Cla88] Design Philosophy of the DARPA Internet Protocols

• Key motivating question
  *How to determine split of functionality*
  – Across protocol layers
  – Across network nodes
Context: David D. Clark (MIT)

• Chief Protocol Architect for the Internet from 1981.
• Continues to be a network visionary today.
• At the time of writing (1987)…
  – (Almost) no commercial Internet
  – 1 yr after Cisco’s 1st product, IETF started
  – Number of hosts reaches 10,000
  – NSFNET backbone 1 year old; 1.5Mb/s
Why did Clark write this paper?

• Difficult to deduce why protocols are what they are

• Misunderstood motivations for designs
  – E.g., datagram is a minimum assumption not a service

• Context for evolution and future designs
First order goal

- **Effective** Technique for **Multiplexed** Utilization of **Interconnected** Networks

  initially ARPANET and ARPA packet radio network

- Effective – undefined!

- Multiplexed == Packet Switching

- Interconnected == One particular gateway realization
Why not unified?

- Networks represent admin boundaries
- Existing networks?
Connecting Networks

• How to internetwork various network technologies
  – ARPANET, X.25 networks, LANs, satellite networks, packet networks, serial links...

• Many differences between networks
  – Address formats
  – Performance – bandwidth/latency
  – Packet size
  – Loss rate/pattern/handling
  – Routing
Challenge 1: Address Formats

• Map one address format to another?
  – Bad idea → many translations needed

• Provide one common format
  – Map lower level addresses to common format
Challenge 2: Different Packet Sizes

• Define a maximum packet size over all networks?
  – Either inefficient or high threshold to support

• Implement fragmentation/re-assembly
  – Who is doing fragmentation?
  – Who is doing re-assembly?
Gateway Alternatives

• Translation
  – Difficulty in dealing with different features supported by networks
  – Scales poorly with number of network types (N^2 conversions)

• Standardization
  – “IP over everything” (Design Principle 1)
  – Minimal assumptions about network
  – Hourglass design
Standardization

- Minimum set of assumptions for underlying net
  - Minimum packet size
  - Reasonable delivery odds, but not 100%
  - Some form of addressing unless point to point

- Important non-assumptions:
  - Perfect reliability
  - Broadcast, multicast
  - Priority handling of traffic
  - Internal knowledge of delays, speeds, failures, etc.

- Much engineering then only has to be done once
Second Order Goals [Clark88]

1. Survivability
   ensure communication service even in the presence of network and router failures

2. Support multiple types of services

3. Must accommodate a variety of networks

4. Allow distributed management

5. Allow host attachment with a low level of effort

6. Be cost effective

7. Allow resource accountability
Is this a laundry list of desired properties

• No, it is ordered

• Ordered for a reason!

• Military context
  ➔ Survivability
  ➔ Less worried about accounting
Survivability

• If network disrupted and reconfigured
  – Communicating entities should not care!
  – No higher-level state reconfiguration

• How to achieve such reliability?
  – Where can communication state be stored?

<table>
<thead>
<tr>
<th></th>
<th>Network</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failure handing</td>
<td>Replication</td>
<td>“Fate sharing”</td>
</tr>
<tr>
<td>Net Engineering</td>
<td>Tough</td>
<td>Simple</td>
</tr>
<tr>
<td>Switches</td>
<td>Maintain state</td>
<td>Stateless</td>
</tr>
<tr>
<td>Host trust</td>
<td>Less</td>
<td>More</td>
</tr>
</tbody>
</table>
Principle: Fate Sharing

• Lose state information for an entity if and only if the entity itself is lost.

• Examples:
  – OK to lose TCP state if one endpoint crashes
    • NOT okay to lose if an intermediate router reboots
  – Is this still true in today’s network?
    • NATs and firewalls

• Survivability compromise: Heterogeneous network → less information available to end hosts and Internet level recovery mechanisms
Principle: Soft-state

• Soft-state
  – Announce state
  – Refresh state
  – Timeout state

• Penalty for timeout – poor performance

• Robust way to identify communication flows
  – Possible mechanism to provide non-best effort service

• Helps survivability
Detour: End-to-End Argument
Principle: End-to-End Argument

• Deals with where to place functionality
  – Inside the network (in switching elements)
  – At the edges

• Argument
  – There are functions that can only be correctly implemented by the endpoints – do not try to completely implement these elsewhere
  – Guideline not a law
Example: Reliable File Transfer

- Solution 1: make each step reliable, and then concatenate them
- Solution 2: end-to-end check and retry
E2E Example: File Transfer

• Even if network guaranteed reliable delivery
  – Need to provide end-to-end checks
  – E.g., network card may malfunction
  – The receiver has to do the check anyway!

• Full functionality can only be entirely implemented at application layer; no need for reliability from lower layers

• Does FTP look like E2E file transfer?
  – TCP provides reliability between kernels not disks

• Is there any need to implement reliability at lower layers?
Discussion

• Yes, but only to improve performance
• If network is highly unreliable
  – Adding some level of reliability helps performance, not correctness
  – Don’t try to achieve perfect reliability!
  – Implementing a functionality at a lower level should have minimum performance impact on the applications that do not use the functionality
Other examples of E2E in paper

• Security
  – Do you want to trust the network with your keys?

• Duplicate message suppression
  – Is it a legit duplicate or not?

• FIFO delivery
Parallels to E2E argument

• Application systems – banking, airlines

• RISC architecture

• “Open operating system” – Lampson
Why E2E

Communication subsystem is frequently specified before applications that use that system are known!

Don’t try to double guess and add complexity ..
Second Order Goals [Clark88]

1. Survivability
   ensure communication service even in the presence of network and router failures

2. Support multiple types of services

3. Must accommodate a variety of networks

4. Allow distributed management

5. Allow host attachment with a low level of effort

6. Be cost effective

7. Allow resource accountability
Types of Service

• TCP vs. UDP
  – Elastic apps that need reliability: remote login or email
  – Inelastic, loss-tolerant apps: real-time voice or video
  – Others in between, or with stronger requirements
  – Biggest cause of delay variation: reliable delivery
    • Today’s net: ~100ms RTT
    • Reliable delivery can add seconds.

• Original Internet model: “TCP/IP” one layer
  – First app was remote login...
  – But then came debugging, voice, etc.
  – These differences caused the layer split, added UDP
Second Order Goals [Clark88]

1. Survivability
   ensure communication service even in the presence of network and router failures
2. Support multiple types of services
3. Must accommodate a variety of networks
4. Allow distributed management
5. Allow host attachment with a low level of effort
6. Be cost effective
7. Allow resource accountability
Minimum Assumptions of interconnected networks

1. Can transport a datagram
2. …of reasonable size
3. …with reasonable chance of delivery

Interesting comments:
Reliability and qualities of service were not built in because they would require too much change. Datagram as a building block, not as a service.
Second Order Goals [Clark88]

1. Survivability
   ensure communication service even in the presence of network and router failures
2. Support multiple types of services
3. Must accommodate a variety of networks
4. Allow distributed management
5. Allow host attachment with a low level of effort
6. Be cost effective
7. Allow resource accountability
Principle: Decentralization

- Each network owned and managed separately
- Will see this in BGP routing especially
Principle: Decentralization’

• Be conservative in what you send and liberal in what you accept
  – Unwritten rule

• Especially useful since many protocol specifications are ambiguous

• E.g. TCP will accept and ignore bogus acknowledgements
Second Order Goals [Clark88]

1. Survivability
   ensure communication service even in the presence of network and router failures
2. Support multiple types of services
3. Must accommodate a variety of networks
4. Allow distributed management
5. Allow host attachment with a low level of effort
6. Be cost effective
7. Allow resource accountability
Other discussion questions

“The most important change in the Internet...will probably be the development of a new generation of tools for management of resources...”

Q. Has this happened?
IP Design Weaknesses

• Greedy sources aren’t handled well
• Weak accounting and pricing tools
• Weak administration and management tools
• Incremental deployment difficult at times
  – Result of no centralized control
  – No more “flag” days
  – Are active networks the solution?
Changes Over Time

• Developed in simpler times
  – Common goals, consistent vision

• With success came multiple goals – examples:
  – ISPs must talk to provide connectivity but are fierce competitors
  – Privacy of users vs. government’s need to monitor
  – User’s desire to exchange files vs. copyright owners

• Must deal with the tussle between concerns in design
Summary: Internet Architecture

- Packet-switched datagram network
- IP is the “compatibility layer”
  - Hourglass architecture
  - All hosts and routers run IP
- Stateless architecture
  - no per flow state inside network
Summary: Minimalist Approach

- **Dumb network**
  - IP provide minimal functionalities to support connectivity
    - Addressing, forwarding, routing
- **Smart end system**
  - Transport layer or application performs more sophisticated functionalities
    - Flow control, error control, congestion control
- **Advantages**
  - Accommodate heterogeneous technologies (Ethernet, modem, satellite, wireless)
  - Support diverse applications (telnet, ftp, Web, X windows)
  - Decentralized network administration
New Principles?

• Design for variation in outcome
  – Allow design to be flexible to different uses/results

• Isolate tussles
  – QoS designs uses separate ToS bits instead of overloading other parts of packet like port number
  – Separate QoS decisions from application/protocol design

• Provide choice → allow all parties to make choices on interactions
  – Creates competition
  – Fear between providers helps shape the tussle