Networking

CS 4410
Operating Systems

[R. Agarwal, L. Alvisi, A. Bracy, M. George, Kurose, Ross, E. Sirer, R. Van Renesse]
Introduction
Application Layer
Transport Layer
Link Layer
Network Layer
Routing
Basic Network Abstraction

• A process can create “endpoints”
• Each endpoint has a unique address
• A message is a byte array
• Processes can:
  • receive messages on endpoints
  • send messages to endpoints
Network “protocol”

Agreement between processes about the content of messages

**Syntax:** Layout of bits, bytes, fields, etc.
- message format

**Semantics:** what fields, messages mean

**Example:**
- HTTP “get” requests and responses
Network Layering

Network abstraction is usually *layered*

- Like Object Oriented-style inheritance
- Also like the hw/sw stack

Proposed 7-Layer ISO/OSI reference model (1970’s)  
Actual 5-Layer Internet Protocol Stack
# OSI Layers

<table>
<thead>
<tr>
<th>Layer</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Application</strong></td>
<td>Network-aware applications, clients &amp; servers</td>
</tr>
<tr>
<td><strong>Presentation</strong></td>
<td>Translation between network and application formats (e.g., RPC packages, sockets)</td>
</tr>
<tr>
<td><strong>Session</strong></td>
<td>Connection management</td>
</tr>
<tr>
<td><strong>Transport</strong></td>
<td>Data transfer, reliability, packetization, retransmission. Lets multiple apps share 1 network connection</td>
</tr>
<tr>
<td><strong>Network</strong></td>
<td>Path determination across multiple network segments, routing, logical addressing.</td>
</tr>
<tr>
<td><strong>Link</strong></td>
<td>Decides whose turn it is to talk, finds physical device on network.</td>
</tr>
<tr>
<td><strong>Physical</strong></td>
<td>Exchanges bits on the media (electrical, optical, etc.)</td>
</tr>
</tbody>
</table>
## Internet Protocol Stack

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
<th>Protocols</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>exchanges <strong>messages</strong></td>
<td>HTTP, FTP, DNS</td>
</tr>
<tr>
<td>Transport</td>
<td>Transports messages; exchanges <strong>segments</strong></td>
<td>TCP, UDP</td>
</tr>
<tr>
<td>Network</td>
<td>Transports segments; exchanges <strong>datagrams</strong></td>
<td>IP, ICMP (ping)</td>
</tr>
<tr>
<td>Link</td>
<td>Transports datagrams; exchanges <strong>frames</strong></td>
<td>Ethernet, WiFi</td>
</tr>
<tr>
<td>Physical</td>
<td>Transports frames; exchanges <strong>bits</strong></td>
<td>wires, signal encoding</td>
</tr>
</tbody>
</table>
Who does what?

<table>
<thead>
<tr>
<th>Layer</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>HTTP, FTP, DNS (these(^\ast) are usually in libraries)</td>
</tr>
<tr>
<td>Transport</td>
<td>TCP, UDP</td>
</tr>
<tr>
<td>Network</td>
<td>IP, ICMP (ping)</td>
</tr>
<tr>
<td>Link</td>
<td>Ethernet, WiFi</td>
</tr>
<tr>
<td>Physical</td>
<td>wires, signal encoding</td>
</tr>
</tbody>
</table>

(Hard to draw firm lines here)

- Each host has 1+ Network Interface Cards (NIC)
- Attaches into host’s system buses
- Combination of hardware, software, firmware
Layers support **Modularity**

Each layer:
- relies on services from layer below
- exports services to layer above

Can identify the relationship between distinct pieces of complex system.

Interfaces between layers:
- Hide implementation details
- Ease maintenance, updates
  - change of implementation of layer’s service transparent to rest of system
Internet, The Big Picture

How about an analogy?

Routers

Endpoints
Application Layer

FROM: Tim
107 Hoy Rd
Ithaca

TO: Kim
100 Baker Dr
Stockholm

Analogy by Shubham Dubey on Quora
Transport Layer

FROM: Tim
107 Hoy Rd
Ithaca

TO: Kim
100 Baker Dr
Stockholm

Ithaca Sorting Office

FROM: Tim
107 Hoy Rd
Ithaca

TO: Kim
100 Baker Dr
Stockholm

Ithaca Postman

Stockholm Postman

SPEEDY DELIVERY!
TO: Stockholm Sorting Office

FROM: Tim
107 Hoy Rd
Ithaca

TO: Kim
100 Baker Dr
Stockholm

FROM: Tim
107 Hoy Rd
Ithaca

TO: Kim
100 Baker Dr
Stockholm

FROM: Tim
107 Hoy Rd
Ithaca

TO: Kim
100 Baker Dr
Stockholm

FROM: Tim
107 Hoy Rd
Ithaca

TO: Kim
100 Baker Dr
Stockholm

FROM: Tim
107 Hoy Rd
Ithaca

TO: Kim
100 Baker Dr
Stockholm
The Big Picture

Application

Transport

Network

Data Link

Physical

messages

segments

Ports (http: 80, DNS: 53, Telnet: 23)

datagrams

IP addresses (192.168.100.254)
or packets

frames

MAC Addresses (00:12:F4:AB:0C:82)

bits
The Big Picture

Application -> messages -> Transport

Transport <- segments <- Network

Network <- datagrams <- Data Link

Data Link <- frames <- Physical

Physical <-> bits <-> Physical

Router1 <-> Network <-> Router2

Network <-> Data Link <-> Network

Data Link <-> frames <-> Data Link

Physical <-> bits <-> Physical

Router1 <-> Network <-> Router2

Network <-> Data Link <-> Network

Data Link <-> frames <-> Data Link

Physical <-> bits <-> Physical

Router1 <-> Network <-> Router2

Network <-> Data Link <-> Network

Data Link <-> frames <-> Data Link

Physical <-> bits <-> Physical
Encapsulation

**source**
- message
- segment
- datagram

**destination**
- application
- transport
- network
- link
- physical

**Headers**
- src & dst ports + ...
- src & dest IP addr + ...
- src & dest MAC addr + ...

**router**
End-to-End Argument

- Occam’s Razor for Internet architecture
- Application-specific properties are best provided by the applications, **not the network**
  - Guaranteed, or ordered, packet delivery, duplicate suppression, security, etc.

- Internet performs the simplest packet routing and delivery service it can
  - Packets are sent on a best-effort basis
  - Higher-level applications do the rest
End-to-End Example

Should the network guarantee packet delivery?

Consider: a file transfer program (read file from disk, send it, receiver reads packets & writes them to disk)

• Q: If network guarantees delivery, wouldn’t applications be simpler? (no retransmissions!)
• A: no, still need to check that file was written to remote disk intact

A check is necessary if nodes can fail.

→ Applications need to be written to perform their own retransmits

Why burden the network with properties that can, and must, be implemented at the periphery?
The Missing Layers

**Presentation**
translation between network & application formats (e.g., RPC packages, sockets).
Allows communicating applications to interpret the meaning of data exchanged:
• data conversion
• character code translation
• compression
• encryption

**Session**
synchronization of data exchange:
• supports checkpointing and recovery schemes
• establish, manage, and tear down connections

Need these services?
Put them in your application.