Channel Capacity

- Some number of symbols per second (baud rate). Each symbol does not necessarily correspond to a bit.
- Nyquist: symbol rate = 2H sym/sec
  - H bandwidth
- Shannon: data rate = H log (1+S/N) b/s
  - S is signal power, N is noise power
Some Comm Theory

• So, with Nyquist, we cannot hope to send binary data even over a noiseless 3-kHz channel at more than 6000 b/sec:
  - \( 2H = 2(3000) = 6000 \text{ b/sec} \)

• With Shannon, bit rate over an analog phone line is limited to about 30kb/s [assuming 30dB S/N ratio]:
  - \( H \log(1+S/N) = 3000 \log(1 + 1000) = 30 \text{kb/s} \)

Transmission Time

• Trans delay = (M bits) / (R b/s) = M/R sec
• Prop delay = D sec
• Tx Time = D + M/R sec
Latency

- Slower channels “stretch out” bits in time:
  - A bit on a 1Mb/s link is 1 µsec wide
  - A bit on a 10Mb/s link is 0.1 µsec wide

- Total Latency = tx time + queue
  - Transmit time = { last slide }
  - Queue delay = { depends! }
High Speed Links

- Large R -> small Tx Time (M/R)
- Ex: OC-3 (D = 10ms, R = 155Mb/s)
  - Tx Time = \(0.010 + \frac{(1024\times8)}{(155\times1024\times1024)}\) = 0.01005 sec
  - = 10.05 ms (D >> M/R)

Total (one way) Latency

- Propagation Delay (D) = distance/speed-of-light
- Transmission delay = (M / R)
- Queueing delay (Q) (using statistical multiplexing) depends on utilization
- Total Latency = D + (M/R) + Q
**Beware of Overheads**

---

**Measuring Latencies (1)**

```
prompt> ping localhost
ping localhost (127.0.0.1) from default, 56 data bytes, 0 iter
64 bytes from 127.0.0.1: icmp_seq=1, 0.855 ms,
64 bytes from 127.0.0.1: icmp_seq=2, 0.482 ms,
64 bytes from 127.0.0.1: icmp_seq=3, 0.62 ms,
64 bytes from 127.0.0.1: icmp_seq=4, 0.596 ms,
--- localhost:default PING Statistics---
--- 4 packets transmitted, 4 packets received, 0% packet loss
round-trip (ms) min/avg/max = 0.482/0.855/0.855
```

```
prompt> ping cs.ucla.edu
ping cs.ucla.edu (131.179.128.13) from default, 56 data bytes, 0 iter
64 bytes from 131.179.128.13: icmp_seq=1, 23.858 ms,
64 bytes from 131.179.128.13: icmp_seq=2, 23.748 ms,
64 bytes from 131.179.128.13: icmp_seq=3, 23.059 ms,
64 bytes from 131.179.128.13: icmp_seq=4, 21.153 ms,
--- cs.ucla.edu:default PING Statistics---
--- 4 packets transmitted, 4 packets received, 0% packet loss
round-trip (ms) min/avg/max = 21.153/23.748/23.858
```
Measuring Latencies (2)

```
prompt> date
Wed Feb 12 01:08:55 PST 1997
prompt> ping -n 5 cs.cmu.edu
ping cs.cmu.edu (192.30.222.172) from default, 56 data bytes, 5 iter
64 bytes from 192.30.222.172: icmp_seq=1: 12350 ms.
64 bytes from 192.30.222.172: icmp_seq=2: 12350 ms.
64 bytes from 192.30.222.172: icmp_seq=3: 12350 ms.
64 bytes from 192.30.222.172: icmp_seq=4: 12350 ms.
64 bytes from 192.30.222.172: icmp_seq=5: 12350 ms.
--- cs.cmu.edu default Ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip (ms) min/avg/max = 12350/12350/12350
```

```
prompt> ping -n 10 www.ucl.ac.uk
ping rs6-sur-8.ucl-0.bcc.ac.uk (144.82.100.19) from default, 56 data bytes, 10 iter
64 bytes from 144.82.100.19: icmp_seq=1: 2612 ms.
64 bytes from 144.82.100.19: icmp_seq=2: 2612 ms.
64 bytes from 144.82.100.19: icmp_seq=3: 2612 ms.
64 bytes from 144.82.100.19: icmp_seq=4: 2612 ms.
64 bytes from 144.82.100.19: icmp_seq=5: 1550 ms.
64 bytes from 144.82.100.19: icmp_seq=6: 1550 ms.
64 bytes from 144.82.100.19: icmp_seq=7: 1550 ms.
64 bytes from 144.82.100.19: icmp_seq=8: 1550 ms.
64 bytes from 144.82.100.19: icmp_seq=9: 276 ms.
64 bytes from 144.82.100.19: icmp_seq=10: 276 ms.
--- rs6-sur-8.ucl-0.bcc.ac.uk default Ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip (ms) min/avg/max = 276/1100/1550
```

Measuring Latencies (3)

```
prompt> ping -n 10 repo.dit.co.jp
ping repo.dit.co.jp (133.159.1) from default, 56 data bytes, 10 iter
64 bytes from 133.159.1: icmp_seq=1: 261 ms.
64 bytes from 133.159.1: icmp_seq=2: 261 ms.
64 bytes from 133.159.1: icmp_seq=3: 261 ms.
64 bytes from 133.159.1: icmp_seq=4: 261 ms.
64 bytes from 133.159.1: icmp_seq=5: 261 ms.
64 bytes from 133.159.1: icmp_seq=6: 261 ms.
64 bytes from 133.159.1: icmp_seq=7: 261 ms.
64 bytes from 133.159.1: icmp_seq=8: 261 ms.
64 bytes from 133.159.1: icmp_seq=9: 261 ms.
64 bytes from 133.159.1: icmp_seq=10: 261 ms.
--- repo.dit.co.jp default Ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip (ms) min/avg/max = 261/261/261
```

```
prompt> ping -n 20 saathi.next.net.in
ping saathi.next.net.in (144.16.1.2) from default, 56 data bytes, 20 iter
64 bytes from 144.16.1.2: icmp_seq=1: 261 ms.
64 bytes from 144.16.1.2: icmp_seq=2: 261 ms.
64 bytes from 144.16.1.2: icmp_seq=3: 261 ms.
64 bytes from 144.16.1.2: icmp_seq=4: 261 ms.
64 bytes from 144.16.1.2: icmp_seq=5: 261 ms.
64 bytes from 144.16.1.2: icmp_seq=6: 261 ms.
64 bytes from 144.16.1.2: icmp_seq=7: 261 ms.
64 bytes from 144.16.1.2: icmp_seq=8: 261 ms.
64 bytes from 144.16.1.2: icmp_seq=9: 261 ms.
64 bytes from 144.16.1.2: icmp_seq=10: 261 ms.
--- saathi.next.net.in default Ping statistics ---
20 packets transmitted, 10 packets received, 60% packet loss
round-trip (ms) min/avg/max = 261/261/261
```

What Happens on the Web?

- Click on a link (http://foo.bar.com/xx)
- Conversion from name to address
- Open connection to remote machine
- Pass arguments to process
- Retrieve contents from server
- Display locally

So, What does this Require?

- Name mapping service (DNS)
- Addressing/routing (IP)
- Reliable delivery (TCP)
- Representation of content (HTTP)
- Local display (application)
Naming Computers

• Need a way to locate services; easier for humans than numbers

• Flat Name Space:
  – every computer has unstructured name
  – must coordinate not to stomp on each other
  – examples: ucbvax, sdcsvax, sri-nic
  – didn’t scale very well

Hierarchical Naming

• First real growth problem of Internet
  – rule of thumb: things break if they grow 2 orders of magnitude (5-7 years in today’s Internet!)
  – Common Idea: hierarchies scale well

• Divide up space into “Domains”
  – examples: EDU, COM, MIL, ORG, NET
Benefits of Naming Hierarchy

- much better scaling
- decentralized administration
- redundant databases
- recursive, can subdivide each subdivision

URLs: New Names

- Relatively New Name Format on Internet
  - popularized by web browsers
  - proto://host-name:port/arg1/arg2/arg3/...
  - http://www.cs.berkeley.edu/~kfall
  - gopher://gopher.colorado.edu
  - telnet://blueskies.spri.umich.edu:3000
A Problem with HTTP

- In version 1.0 of HTTP, the host name is not passed to the web server
- What about “web hosting” multiple sites?
- Utilizes more IP addresses than necessary!

IP (v4) Addresses

- Every interface has at least 1 IP address
- IP addresses are 32-bit numbers (4.3 billion of them!)
- Divided into parts: (network prefix, host number)
- Classical structure use net/subnet/host partitioning where hosts on same subnet share net and subnet number
Expressing Addresses

• 4 decimal numbers, called “dotted quad”
• Each (decimal) number is one byte
• Example: 128.32.25.12
• Can generally be used in place of names
• Classically, parts of “Classes”

IP Address Classes (historical)

<table>
<thead>
<tr>
<th>Class</th>
<th>Network</th>
<th>Host</th>
<th>Highest Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>31</td>
<td>127.255.255.255</td>
</tr>
<tr>
<td>B</td>
<td>10</td>
<td>24</td>
<td>191.255.255.255</td>
</tr>
<tr>
<td>C</td>
<td>110</td>
<td>24</td>
<td>223.255.255.255</td>
</tr>
<tr>
<td>D</td>
<td>1110</td>
<td>10</td>
<td>239.255.255.255</td>
</tr>
</tbody>
</table>
Special IP Addresses

<table>
<thead>
<tr>
<th>32 bits</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>all zero</td>
<td>This host (during boot)</td>
</tr>
<tr>
<td></td>
<td>Default route (in tables)</td>
</tr>
<tr>
<td>all one</td>
<td>Local net broadcast</td>
</tr>
<tr>
<td>Network</td>
<td>Directed broadcast</td>
</tr>
<tr>
<td>01111111</td>
<td>Net 0, often 1</td>
</tr>
<tr>
<td></td>
<td>Loopback</td>
</tr>
</tbody>
</table>

Example Assignments
Subnet Addressing

- Historical, but terminology is consistent and still used
- Allows one site to have multiple subnetworks of their main network. Practical result: multiple segments.
- Subnetting scheme is a **local** decision
- Requires a “subnet mask”

Subnet Structure

- Idea is to steal host bits and use them for numbering subnets
- Rest of Internet only sees classes (or their aggregates--- later)
- Mask indicates which bits are network/subnet part, and which are host part
Subnet Example

• 128.32.25.12 is a “Class B” address
• 16 bits of network, 16 bits of host
• Locally, want a thousand “subnets”
• So, need 10 bits to indicate subnet
• Use a subnet mask of (16+10=26) bits

Subnet Example (cont)

• 26 bit mask: 0xffffffffc0 or simply “/26”
• So, 128.32.25.12/26 is:
  – 10000000 00100000 00011001 00001100
  & 11111111 11111111 11111111 11000000
• Subnet 100 of net 128.32, host 12
Subnet Partitioning (ex cont)

- 128.32.0.0/26 gives $2^{(26-16)} = 1024$ subnets of $2^{(32-26)}-2 = 62$ hosts each
- First usable address: 128.32.0.1 (see RFC1812, page 48)
- Last usable address: 128.32.255.254
- Any address with all “1” bits in host part is a (subnet) broadcast

128.32.25.12/26 is:
- 10000000 00100000 00011001 00001100

128.32.0.65/26 is:
- 10000000 00100000 00000000 01000011

128.32.255.190/26 is:
- 10000000 00100000 11111111 10111110
Common Subnet?

- Is 128.32.25.12 and 128.32.25.85 on the same subnet using a /26 mask?
- 128.32.25.12 is: $\text{10000000 00100000 00011001 00001100}$
- 128.32.25.85 is: $\text{10000000 00100000 00011001 01010101}$
- Prefixes differ, so not on same subnet (need router to reach)

Classless Inter-domain Routing (CIDR)

- About 1993, remove strict classes from architecture
- Generalized notion of “network prefix”
- Requires “longest prefix” match routing
- Subsumes and generalizes subnetting
- (will discuss when we cover IP routing)