Advanced Computer Networks
  263-3501-00

Introduction

Patrick Stuedi

Spring Semester 2014
Information about the Course

• Website
  – http://www.systems.ethz.ch/courses/spring2014/acn
• The language of the course is English
• The exam will be in English
• About me:
  – 2008-2010 at Microsoft Research, Silicon Valley
  – Since 2011 at IBM Research
Goals of this course

1. “Fill in the gaps” in the basic *Operating Systems and Networking* course at ETHZ (and similar courses elsewhere).

2. Cover in depth two hot growth areas in computer networking.
Course overview

Wireless networking technologies: first half of this course

HTTP, RPC, DNS, TCP, UDP, IP, BGP, OSPF, Ethernet, MPLS, PoS

Datacenter networking: second half of this course

covered in basic ETH “Operating Systems and Networks” course

Slides adapted from Prof. Roscoe
Assumptions

You should know about:

• The Ethernet MAC protocol
• Packet switching and circuit switching
• IP inter-domain and intra-domain routing
• TCP reliability mechanisms and congestion control
• HTTP, RPC and other application-layer protocols

Slides adapted from Prof. Roscoe
Course Outline

1. General principles of network design
   – Review of basic concepts from earlier course(s)
   – Design principles and arguments

2. Wireless and mobile networking
   – Basic MAC and PHY principles
   – Bluetooth, Wifi, GSM, 3G, 4G
   – Mobility and Cloud services

3. Datacenter and high-performance networking
   – Supercomputer interconnects, datacenter topologies
   – Infiniband, RDMA, etc.
   – L7 switching, load balancing, OpenFlow, network virtualization
   – Virtual machine networking, IOV, soft switches

Slides adapted from Prof. Roscoe
Course Material

• No good text book that covers all the material
  – Many research papers referenced during the lectures (will go up on the website)

• Supplementary text books (second half)
  – Mobile Communications 2nd edition, Jochen Schiller, Addison Wesley
  – Wireless Communications & Networks, William Stallings
  – Wireless Communications and Networking, Vijay Garg

Slides adapted from Prof. Roscoe
Lab Assignments

• Your assistant: Qin Yin

• Wireless assignments:
  – Mostly pen-and-paper exercises

• Datacenter assignments:
  – More programming-oriented
  – Please plan accordingly!
Fundamental Challenges for Networking

• Speed-of-light
• A pervasive global network
• Disparate parties must work together
• Efficient/cheap operation
• Enormous dynamic range
  – “no such thing as typical”
• Handling failure of components
• Malicious intent
• Rapid growth/evolution

Slides adapted from Prof. Roscoe
Review: network performance

What is network performance?
Two fundamental measures:

1. Bandwidth
   - Roughly: bits transferred in unit time
   - Not quite the Electrical Engineering definition
   - Also known as **Throughput**

2. Latency
   - Time for 1 message to traverse the network
   - Half the **Round Trip Time** (RTT)
   - Also known as **Delay**
Time to transfer an object

(Ignoring queuing delay, processing, etc.)
Time to transfer an object

(*Ignoring* queuing delay, processing, etc.)

e.g. digital photograph
Time to transfer an object

(*Ignoring* queuing delay, processing, etc.)

---

e.g. email message

---

Slides adapted from Prof. Roscoe
Time to transfer an object

(Ignoring queuing delay, processing, etc.)

![Graph showing perceived latency vs. round-trip time for different object sizes and transfer speeds.]

e.g. typing a character

Slides adapted from Prof. Roscoe
Example: request 1MB file over a 1Gb/s link, with 200ms RTT.

- What’s the *throughput*?

  \[
  \text{Throughput} = \frac{\text{Transfer size}}{\text{Transfer time}}
  \]

- What’s the transfer time?

  \[
  \text{Transfer time} = \text{RTT} + \frac{\text{Transfer size}}{\text{Bandwidth}}
  \]
Example: request 1MB file over a 1Gb/s link, with 200ms RTT.

• What’s the *throughput*?

\[
\text{Throughput} = \frac{\text{Transfer size}}{\text{Transfer time}}
\]

• What’s the transfer time?

\[
\text{Transfer time} = \text{RTT} + \frac{\text{Transfer size}}{\text{Bandwidth}}
\]

Request + first byte delay

Slides adapted from Prof. Roscoe
Example: request 1MB file over a 1Gb/s link, with 200ms RTT.

- **Transfer time** =
  \[
  0.2s + 8\text{Mb}/1\text{Gbs} = 0.208s
  \]

- **So throughput** =
  \[
  8\text{Mb} / 0.208s = \sim 38.5\text{Mb/s}
  \]
Example: request 1MB file over a 1Gb/s link, with 200ms RTT.

- Transfer time =

  \[0.2s + \frac{8\text{Mb}}{1\text{Gbs}} = 0.208\text{s}\]

- So throughput =

  \[\frac{8\text{Mb}}{0.208\text{s}} \approx 38.5\text{Mb/s}\]
What’s gone wrong here?

• File is too small?
• Round-trip time is too high?
• You can’t reduce the latency
• Protocol buffers aren’t large enough
• Adding bandwidth won’t really help.
Bandwidth-Delay product

• Example: Latency = 200ms, Bandwidth = 40Gb/s
  – ⇒ “channel memory” = 8Gb, or 1 gigabyte

• What the sender can send before receiver sees anything
  – Or must send to keep the pipe full...

Slides adapted from Prof. Roscoe
TCP window size

• Recall TCP keeps data around for transmission!
  – E.g. consider sliding window of $w$ bytes
  – TCP moves $w$ bytes every RTT (needs to wait for ACK)
    $\Rightarrow$ throughput $= w / \text{RTT}$

• What’s the optimal window size $w$?
  – Need to keep the pipe completely full
    $\Rightarrow w = \text{RTT} \times \text{pipe bandwidth}$

• Example: 10Gb/s, 200ms RTT
  – Gives: $w \sim 200\text{MB}$ per connection…

• Protocol limits:
  – TCP window size without scaling $\leq 64\text{kB}$
  – TCP window size with RFC1323 scaling $\leq 1\text{GB}$
What’s gone wrong here?

• File is too small?
• Round-trip time is too high?
• Protocol buffers aren’t large enough
• Adding bandwidth won’t really help.
• You can’t reduce the latency.

• **Moral 1**: effectively using high bandwidth-delay product networks is hard
• **Moral 2**: as bandwidth increases, latency is more important for performance.

Slides adapted from Prof. Roscoe
What are these high-BDP networks?

• Where is there high bandwidth?
  – Datacenters! 40Gbps Ethernet → 100Gbps Ethernet soon

• Where is there high delay?
  – Between datacenters: 200ms not unusual
  – Wireless protocols: many reasons, as we will see
Protocol design impacts performance

• Protocols which require many RTTs don’t work well in the wide area.

• Example: Opening a network folder in Windows 2000
  – About 80 request/response pairs on average
  – 200ms RTT (e.g. London-Redmond)
  – ⇒ more than 16 seconds delay

• Upgrading your network will not help!
  – (and didn’t...)
Review: application requirements
Application requirements

• Some applications can’t use all the network

• Example: constant-bit-rate 320kbs MP3 streams

• **Utility function**: measure of how useful each network resource (bandwidth, latency, etc.) is to an application
Application utility functions

“Utility”

Bandwidth

Infinitely large file transfer

Slides adapted from Prof. Roscoe
Application utility functions

“Utility”

Bandwidth

Real file transfer

Slides adapted from Prof. Roscoe
Application utility functions

“Utility”

Bandwidth

Constant bit rate stream
(e.g. audio, video)

Slides adapted from Prof. Roscoe
Application utility functions

“Utility”

Bandwidth

Network-coded variable-bit-rate multimedia

Slides adapted from Prof. Roscoe
Application requirements: Burstiness

- Many applications are “bursty”
  - Required network bandwidth varies over time

- Challenge: describe and analyze bursty sources
  - Token buckets, leaky buckets, queuing theory...
Application requirements: Jitter

- **Variance** in end-to-end latency (or RTT)
- **Example**: voice (telephony)
- **How long to wait to play a received packet?**
  - Too long: hard to have a conversation (150ms limit)
  - Too short: some packets arrive too late (lose audio)
Application requirements: Loss

• Networks do, sometimes, lose packets.
• Loss is complex:
  – Packet loss rate, Bit-error rate
• Losses and errors must be detected:
  – Codes, sequence numbers, checksums
• Handled by:
  – Error-correction
  – Retransmission

Slides adapted from Prof. Roscoe
## Application requirements: Loss

<table>
<thead>
<tr>
<th>Distance from server to user</th>
<th>Network latency</th>
<th>Typical packet loss</th>
<th>Throughput (quality)</th>
<th>4GB DVD download time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local: &lt;160 km</td>
<td>1.6ms</td>
<td>0.6%</td>
<td>44Mb/s (HDTV)</td>
<td>12 min.</td>
</tr>
<tr>
<td>Regional: 800-1600km</td>
<td>16ms</td>
<td>0.7%</td>
<td>4Mbs (not quite DVD)</td>
<td>2.2 hrs.</td>
</tr>
<tr>
<td>Cross-continent ~4800km</td>
<td>48ms</td>
<td>1.0%</td>
<td>1Mbs (not quite TV)</td>
<td>8.2 hrs.</td>
</tr>
<tr>
<td>Multi-continent ~10000km</td>
<td>96ms</td>
<td>1.4%</td>
<td>0.4Mbs (poor)</td>
<td>20 hrs.</td>
</tr>
</tbody>
</table>

Network design principles
Principles

• Naming
• Layering and modularity
• The Internet Hourglass
• The End-to-End argument
• Postel’s Law: be conservative in what you send, liberal in what you accept
• Soft state vs. Hard state
• Fate-sharing
Naming

Naming at many layers

- HTTP URLs
- Domain names (FQDNs)
- Socket descriptors
- TCP port numbers
- IP addresses
- Protocol identifiers
- MAC addresses
- etc...
What is being named?

Example: what does an IP address actually name?

A *computer*?

Many people seem to think so, but:

– Computers have several IP addresses at a time!
– 127.0.0.1 is always “my computer”
– My WiFi-enabled phone gets a new IP address every time I walk into the coffee shop
– If I’m behind a NAT, my IP address doesn’t mean much
– See Mobile IP later...
What is being named?

Example: what does an IP address actually name?

A *network interface*?

But:

– 127.0.0.1 isn’t a real interface
– One interface can have multiple IP addresses
– A computer might respond to ARPs on multiple interfaces for the same IP address

• And as before:
  – IP addresses change, get reused, sit behind NATs
What is being named?

Example: what does an IP address actually name?

A *service*?

Surely not - IP is a network layer concept! But:

- In a datacenter many machines sit behind a single IP address
- TCP connections are *load balanced* across a web farm
- Transparent proxies intercept IP addresses of particular web sites
What is being named?

Example: what does an IP address actually name?

A service?

Surely not - IP is a network layer concept! But:

- In a datacenter many machines sit behind a single IP address
- TCP connections are load balanced across a web farm
- Transparent proxies intercept IP addresses of particular web sites

Clearly some kind of disciplined is approach is needed to make sense of all this!
Basics:
We need to name objects

Socket clientSocket = new Socket("hostname", 6789);

Create a new object
Give it a name

Slides adapted from Prof. Roscoe
Naming provides *indirection*

```java
DataOutputStream outToServer = new DataOutputStream(clientSocket.getOutputStream());
```

Could be any socket we have now
Network naming

Shoch’s terminology:

• A *name* identifies *what you want*.
• An *address* identifies *where it is*.
• A *route* identifies *a way* to get there.

Note that in this terminology, *syntax* (human readable, binary, etc.) is completely irrelevant!
Network naming

Saltzer’s criticism on Shoch’s terminology:
- Classification into name, address and route is artificial: address is just another name

Saltzer’s view:
- Many different types of network destination (objects), each of which can be named
- Bindings among them
Objects to be named

- Services and users
  - Functions that are used ("Facebook") and clients that use them ("a web browser", "a user")
- Nodes
  - Computers that can run services or user programs
- Network attachment points
  - Places where a node is attached to a network (e.g. a network interface)
- Paths
  - Connections between network attachment points, traversing forwarding nodes and communication links
Binding

• The association between a name and a value is called a *binding*.

• In most cases, the binding isn’t immediately visible
  – Most people miss it, or don’t know it exists
  – Often conflated with creating the value itself

• Sometimes bindings are *explicit*, and are objects themselves.
To talk to a service...

1. Find a node on which the service is running
   – Service name resolution

2. Find a network attachment point to which that node is connected
   – Node name location

3. Find a path from this attachment point to that attachment point
   – Routing, connection setup

Slides adapted from Prof. Roscoe
Context

• Problem: binding might not be obvious under certain circumstances

• “you”, “here”, “Ueli Maurer” are names that require a context to be useful

• Any naming scheme must have ≥ 1 context

• Context may not be stated: always look for it!
A general model of naming

- Designer creates a **naming scheme**.
  1. Name space: what names are valid?
  2. Universe of values: what values are valid?
  3. Name mapping algorithm: what is the association of names to values?

- Mapping algorithm also known as a **resolver**

- Requires a **context**
Example naming scheme: Virtual address space

- **Name space:**
  - Virtual memory addresses (e.g. 64-bit numbers)

- **Universe of values:**
  - Physical memory addresses (e.g. 64-bit numbers)

- **Mapping algorithm:**
  - Translation via a page table

- **Context:**
  - Page table root

Slides adapted from Prof. Roscoe
Single vs. multiple contexts

• IPv4 addresses:
  – E.g. 129.132.102.54
  – Single (global) context: routable from anywhere
  – Well, sort of…

• ATM virtual circuit/path identifiers
  – E.g. 43:4435
  – Local context: only valid on a particular link/port
  – Many contexts!

Slides adapted from Prof. Roscoe
The message...

Naming is complex!

A clear model of naming (such as Saltzer’s):

• Does help to understand real networks
  – What is being named? What is its context?
  – Is the “name” an identifier, or an address, or both?

• Does not correspond to real networks
  – Real networks (particularly the Internet!) confuse and conflate different naming concepts

• Motivates understanding new work
  – E.g. HIP: Host Identity Protocol

Slides adapted from Prof. Roscoe
Next week...

More principles:

- Layering and modularity
- The Internet Hourglass
- The End-to-End argument
- Postel’s Law (robustness principle)
- Soft state vs. Hard state
- Fate-sharing