Networks and Interprocess Communication

Johan Montelius

KTH

HT15

Requirements

- Performance
- Scalability
- Reliability
- Security
- Mobility
- Quality of Service
- Multicasting

Types of networks

- WAN - Wide Area Networks
- MAN - Metropolitan Area Networks
- LAN - Local Area Networks
- PAN - Personal Area Networks

latency

Transfer rate:

What is the rate at which we can send data?

performance

- Latency - how long time does it take to send an empty message?
- Transfer rate - what is the rate at which we can send data?

latency

Why does it take time to send a message?

- distance - speed of signal (light)
- access - granting of resource
- routing - processing in nodes

fast as ..

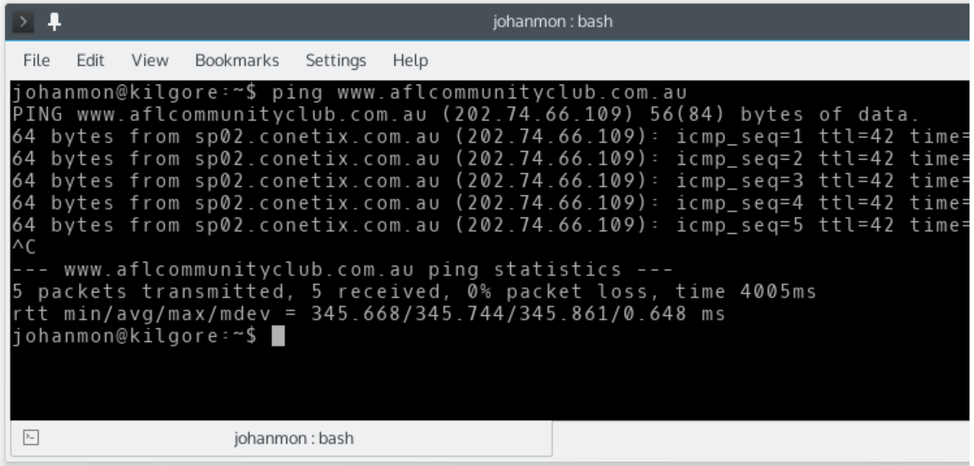
What is the speed of light?

Distance in ms:

- Stockholm - Hamburg
- Stockholm - NYC
- Stockholm - Melbourne

Routers. swithes and fiber optics adds to this so Melbourne is aprx 300 ms away.

ping



```
johanmon@kilgore:~$ ping www.aflcommunityclub.com.au
PING www.aflcommunityclub.com.au (202.74.66.109) 56(84) bytes of data.
64 bytes from sp02.conetix.com.au (202.74.66.109): icmp_seq=1 ttl=42 time=
64 bytes from sp02.conetix.com.au (202.74.66.109): icmp_seq=2 ttl=42 time=
64 bytes from sp02.conetix.com.au (202.74.66.109): icmp_seq=3 ttl=42 time=
64 bytes from sp02.conetix.com.au (202.74.66.109): icmp_seq=4 ttl=42 time=
64 bytes from sp02.conetix.com.au (202.74.66.109): icmp_seq=5 ttl=42 time=
^C
--- www.aflcommunityclub.com.au ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4005ms
rtt min/avg/max/mdev = 345.668/345.744/345.861/0.648 ms
johanmon@kilgore:~$
```

Using ICMP packages might give a better value, UDP might be slower.

latency in different networks

- LAN/WLAN - local area networks (Ethernet/WiFi)
- WAN - wide area networks (IP routed)
- Mobile networks
- Satellite (geo-stationary)

message size

How does latency vary with the size of the messages?

transfer rate

The rate at which we can send data (does not mean that it has arrived).

What is the transfer rate of:

- ADSL
- Ethernet
- 802.11
- 3G/4G

Is this shared with others?

overhead

- medium access: 802.11 - RTS, CTS
- error handling: detection, forward error correction, ARQ
- header: MAC header, IP header, TCP ...
- flow control: TCP window

what's in it for me?

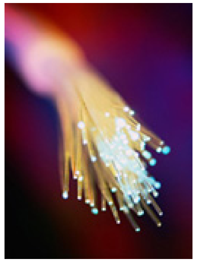
The application layer transfer rate is much lower than the physical layer bit rate.

How does the application layer latency differ from the network layer latency?

Latency and transfere rate

Stockholm to Gothenburg - 400 km, best possible data communication layer?

100 m^3 or five million BlueRay 50Gbyte disks, delivered in 6 h, two trucks every day



10 Gbit/s

Communication layers

- application: the end product
- presentation: encoding of information, serialization, marshaling
- session: security, authentication, initialization
- transport: messages, streams, reliability, flow control
- network: addressing of nodes in a network, routing, switching
- data link: point to point deliver of frames, medium access, link control
- physical layer: bits to analog signals, electrical, optical, radio ...

Internet stack

- HTTP, FTP, SMTP
- TCP, UDP, SCTP, ICMP
- IP, ARP
- Ethernet, WiFi, ..

what if

What would the world look like ...

.. if we only had Ethernet?

routing

Two approaches:

- Distance vector: send routing table to neighbors, RIP, BGP
- Link state: tell everyone about your direct links, OSPF

Pros and cons?

IP addresses

What is the structure of an IP address?

How would you allocate IP addresses to make routing easier?

What is actually happening?

UDP and TCP

One word that that describes the difference between UDP and TCP.

UDP and TCP

Introduces two communication abstractions:

- UDP: datagram
- TCP: stream

- Gives us port numbers to address processes on a node.
- About hundred other protocols defined using IP. (ICMP, IGMP, RSVP, SCTP...)
- More protocols defined on top of UDP and TCP.

UDP

- A datagram abstraction, independent messages, limited in size.
- Low cost, no set up or tear down phase.
- No acknowledgment.

TCP

- A duplex stream abstraction.
- Reliability, lost or erroneous packets are retransmitted.
- Flow control, to prevent the sender from flooding the receiver.
- Congestion friendly, slows down if a router is choked.

UDP and TCP

- UDP: small size messages, build your own streams
- TCP: large size messages, flow control of a steam of messages

Can you trust TCP delivery?

sockets

Sockets is the programmers abstraction of the network layer:

- datagram sockets for messages (UDP)
- stream sockets for duplex byte streams (TCP)

stream socket

- server
  - Create a *listen socket* attached to a port (could be in several steps: create, bind, listen)
  - Accept incoming request and create a *communication socket*, this is the socket used for reading/writing.
- client
  - Create a *communication socket* and connect to a server given a specified port.
  - Read and write from socket.

A server in Erlang

```
init(Port) ->
 case gen_tcp:listen(Port, [..]) of
     {ok, Listen} ->
         handler(Listen),
         gen_tcp:close(Listen);
     {error, Error} ->
         error
 end.
```

```
handler(Listen) ->
 case gen_tcp:accept(Liste
    {ok, Client} ->
        request(Client),
        handler(Listen);
     {error, Error} ->
         error
 end.
```

A server in Erlang

```
request(Client) ->
  case gen_tcp:recv(Client, 0) of
     {ok, Request} ->
        Response = reply(Request),
        gen_tcp:send(Client, Response);
     {error, Error} ->
        error
    end,
    gen_tcp:close(Client).
```

```
respons(Request) ->
     :
    generat and return
 a  byte seqence
```

datagram socket

- server
  - Create *message socket* and bind to port.
  - Read an incoming message (message contains source address and port).
- client
  - Create *message socket* with source port.
  - Create message and give destination address and port.
  - Send message.

marshaling of data

How do we transform internal datstructure into sequencing of bytes?

- Language dependent: Java serialization, Erlang external term format
- Independent: XML, Google Protocol Buffer, ASN.1
  - message format defined by specification: XML Schema, .proto, ...
  - specification is used by compiler to generate encoder and decoder

example

ANS.1 specification

```
FooProtocol DEFINITIONS ::= BEGIN
    FooQuestion ::= SEQUENCE {
        trackingNumber INTEGER,
        question       IA5String}
    FooAnswer ::= SEQUENCE {
        questionNumber INTEGER,
        answer         BOOLEAN}
END
```

C data structures

```
struct foo_question {
    int   tracking_number;
    char  question[128];
}

foo = {5, "Anybody there?"};
```

```
 30 13 02 01 05 16 0e 41 6e 79 62 6f 64 79 20 74 68 65 7
```

summary

The application layer should in a perfect world be independent of underlying layers.

The world is not perfect.

Understanding underlying network characteristics is essential when developing distributed applications.