Overview

- Administrative Services
  - Internet Standardization Process
  - Basic Internet Protocols
  - DHCP & Stateless Address Configuration
  - Telnet
  - Traceroute + Ping
Main question

- User invokes an operation in a networked enabled application
  - Examples
    - Sends an email
    - Retrieves an email
    - Requests HTML page
    - Invokes a Web service
    - RMI call
  - Question
    - Which messages are emitted at network interface?
Each distinct version of an Internet standards-related specification
- Published as part of the "Request for Comments" series

RFCs are official publication channel
- Since 1969
- Publication responsibility of the RFC Editor
  - Under direction of IAB (Internet Architecture Board)

Standards Process itself is RFC 2026
- Formatting conventions RFC 1543
Internet Standards Process

- First posted as an Internet-Draft
  - Published for informal review and comment

- Proposed Standard
  - Generally stable
  - Significant community review

- Draft Standard
  - At least two independent and interoperable implementations with different code bases

- Internet Standard
  - Significant implementation
  - Successful operational experience
  - STDs in addition to RFCs
Internet Standards

- See RFC 3700
- IP+ICMP+IGMP
  - STD 5 (RFC 791+792+919+922+950)
- UDP
  - STD 6 (RFC 768)
- TCP
  - STD 7 (RFC 793)
ISO OSI Model

- Idealized protocol stack
  - Implementations look different (usually)
- Each upper level protocol builds on the next lower
- ISO = International Standards Organization
- OSI = Open Systems Interconnection

<table>
<thead>
<tr>
<th>Protocol Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application protocol</td>
</tr>
<tr>
<td>Presentation protocol</td>
</tr>
<tr>
<td>Session protocol</td>
</tr>
<tr>
<td>Transport protocol</td>
</tr>
<tr>
<td>Network protocol</td>
</tr>
<tr>
<td>Data link protocol</td>
</tr>
<tr>
<td>Physical protocol</td>
</tr>
</tbody>
</table>
Application Protocols

Host A

Application (client)

Application protocol
Presentation protocol
Session protocol
Transport protocol
Network protocol
Data link protocol
Physical protocol

Host B

Application (server)

Application protocol
Presentation protocol
Session protocol
Transport protocol
Network protocol
Data link protocol
Physical protocol

Physical Network
Message structure

- Message header
  - Message id
  - Message length (header length)
  - Checksum
  - Source and destination address
  - Options
  - ...
- Payload
Headers & Layers

- Encapsulation of messages
  - Message from Layer n+1
  - Forms payload of message in Layer n
  - Header for message Layer n added

- Effect of encapsulation
  - Headers for all messages contained in the final message
802.3 (Ethernet) Frames

- 48 bit addresses
- Max. 1500 Bytes of payload!
- Frames
- All hosts listen on frames for their address
  - Frame is picked when address is found
- Unique address for each node (MAC address)

<table>
<thead>
<tr>
<th>Destination address</th>
<th>Source address</th>
<th>Type</th>
<th>Data (Payload)</th>
<th>checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 bytes</td>
<td>6 bytes</td>
<td>2 bytes</td>
<td>46 - 1500 bytes</td>
<td>4 bytes</td>
</tr>
</tbody>
</table>
Internet Protocol (IP)

- IPv4
  - STD 5 (RFC 791)
- IPv6
  - RFC 2460
  - Draft Standard
IP

- Virtual Network
- Routing
- Connectionless (datagram)
  - Not required to connect to recipient
- Transmission over several networks
- Unreliable
  - Order undetermined
  - Loss of packets
IPv4

- **Addressing**
  - IP address: 32bits (network id, host id)
- **Max packet size 64kB**
- **Fragmentation and reassembly**
  - Data Link Layer Frames usually smaller
- **Time to live**
  - Number of hops
**IPv4 Header**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-bit version</td>
<td>4-bit header length</td>
</tr>
<tr>
<td>8-bit type of service (TOS)</td>
<td>16-bit total length (in bytes)</td>
</tr>
<tr>
<td>16-bit packet id</td>
<td>3-bit flags</td>
</tr>
<tr>
<td>8-bit time to live (TTL)</td>
<td>13-bit fragment offset</td>
</tr>
<tr>
<td>8-bit higher level protocol (TCP,UDP,...)</td>
<td>16-bit header checksum</td>
</tr>
<tr>
<td>32 bit - source IP address</td>
<td></td>
</tr>
<tr>
<td>32 bit - destination IP address</td>
<td></td>
</tr>
<tr>
<td>0 - 40 bytes</td>
<td>Options (if any)</td>
</tr>
</tbody>
</table>

- **IPv4 Header Format**
  - 0 - 40 bytes
  - Options (if any)
  - 32 bit - destination IP address
  - 32 bit - source IP address
  - 16-bit header checksum
  - 13-bit fragment offset
  - 3-bit flags
  - 8-bit higher level protocol (TCP,UDP,...)
  - 8-bit time to live (TTL)
  - 16-bit total length (in bytes)
  - 4-bit header length
  - 4-bit version
**IPv4 Addresses**

- **Numeric:** 128.131.172.25
- **network id and host id**
- **3 unicast classes A-C, 1 multicast D**

<table>
<thead>
<tr>
<th>Class</th>
<th>NetID Bits</th>
<th>HostID Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0 7 bit</td>
<td>24 bits</td>
</tr>
<tr>
<td>B</td>
<td>1 0 14 bit</td>
<td>16 bit</td>
</tr>
<tr>
<td>C</td>
<td>1 1 21 bit</td>
<td>8 bit</td>
</tr>
<tr>
<td>D</td>
<td>1 1 1 28 bit - Multicast group ID</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>1 1 1 27 bit - reserved for future use</td>
<td></td>
</tr>
</tbody>
</table>
Subnet Addressing

- Only small number of networks possible
  - ~2,000,000
- Interpret IP address considered as 3 parts
  - Host-ID split in Subnet-ID and Host-ID

Class B

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>0</th>
<th>14 bits - netid</th>
<th>8 bits - subnetid</th>
<th>8 bits - hostid</th>
</tr>
</thead>
</table>

- Subnet Mask
  - Hosts need to know how many bits for subnet
  - 32-bit value with bits set in Network id & Subnet id field
  - Example
    - Explicit: 128.131.172.25  255.255.255.0
    - Prefixlength: 128.131.172.25 /24 (number how many bits are set) (11111111.11111111.11111111.00000000 = 255.255.255.0)
IPv6

- **Large addressing scheme**
  - 128 bit addresses

- **Next header field**
  - Realizes linked list of headers
  - Last field refers to protocol type (TCP, UDP, …)

- **Extension headers**
  - Hop-by-Hop Options
  - Routing
    - lists Intermediate nodes to be visited
  - Fragment
    - For sending a packet larger than the path MTU
  - Destination Options
  - Authentication
  - Encapsulating Security Payload

- **Support for Jumbograms (RFC 2675)**
  - Payload larger than 64kB
### IPv6 Header

<table>
<thead>
<tr>
<th>4-bit version</th>
<th>8-bit traffic class</th>
<th>20-bit flow label</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>16-bit payload length + extension headers (in bytes)</td>
<td>8-bit next header (next protocol)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8-bit hop limit (TTL)</td>
</tr>
</tbody>
</table>

**40 bytes**

- 128 bit - source IP address
- 128 bit - destination IP address
- Extension Headers
IPv6 Addresses

- 128bit
- Written as 8 hex-numbers
  - Ex: 2001:0db8:0000:1347:0000:0000:0000:0001
  - Leading zeros may be omitted
    - 2001:db8:0:1347:0:0:0:1
  - One sequence of 0s replaced by ::
    - 2001:db8:0:1347::1
  - ::1 is loopback
- Last 64 bits are Interface ID
- First 64 bits Global Routing Prefix and Subnet ID
  - Global Routing Prefix provided by Internet Service Provider
Transmission Control Protocol
TCP

- Multiple TCP endpoints – Ports
  - 1-65535
  - Like Post Office Boxes

- Connection-oriented
  - Virtual Circuit
  - Special Flags

- Flow control
  - Transmission speed reduction if one side is too slow
### TCP Header

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>16 bit - Source port</td>
<td>Port number from which the data is sent</td>
</tr>
<tr>
<td>16 bit - Destination port</td>
<td>Port number to which the data is sent</td>
</tr>
<tr>
<td>32 bit - Sequence number</td>
<td>Unique sequence number for each sent segment</td>
</tr>
<tr>
<td>32 bit – Acknowledgement number</td>
<td>Acknowledgement for the expected sequence number</td>
</tr>
<tr>
<td>4 bit – Header length</td>
<td>Length of the header in 32-bit units</td>
</tr>
<tr>
<td>6 bit - reserved</td>
<td>Reserved bits for future use</td>
</tr>
<tr>
<td>6 bit - Flags</td>
<td>Flags for control purposes</td>
</tr>
<tr>
<td>16 bit – Window</td>
<td>Size of the receive buffer</td>
</tr>
<tr>
<td>16 bit – Checksum</td>
<td>Checksum for data integrity</td>
</tr>
<tr>
<td>16 bit – Urgent pointer</td>
<td>Pointer to urgent data</td>
</tr>
<tr>
<td>Options (optional)</td>
<td>Additional options (if any)</td>
</tr>
</tbody>
</table>
TCP Communication

- Client connects to server
  - Sends TCP (command) segment with
    - SYN flag on, ACK flag off
    - SequenceNr = x

- Server responds
  - Sends TCP (command) segment with
    - SYN flag on, ACK flag on
    - SequenceNr = y, AckNr = x+1

- Client sends data to server
  - Sends TCP segment with
    - SYN flag off, ACK flag on
    - SequenceNr = x+1, AckNr = y+1

- ...
User Datagram Protocol

- Transmitted within IP protocols
- Multiple UDP endpoints – Ports
  - 1-65535
- Connection-less
<table>
<thead>
<tr>
<th>16 bit - Source port</th>
<th>16 bit - Destination port</th>
</tr>
</thead>
<tbody>
<tr>
<td>16 bit - Length</td>
<td>16 bit - checksum</td>
</tr>
</tbody>
</table>

8 bytes
Internet Control Message Protocol - ICMP

- Transmitted within IP protocols
- IP's Response & Error mechanism
- ICMP error message
  - Types
    - Network unreachable
    - Host unreachable
    - Port unreachable
    - ...
- ICMP query messages
  - Eg. Echo request, Echo reply
  - ...


### ICMP Error Message

<table>
<thead>
<tr>
<th></th>
<th>8 bit - Type</th>
<th>8 bit - Code</th>
<th>16 bit - Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>32 bit –</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Message Data</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>20-60 bytes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Original</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Header</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>8 byte –</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Original</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Request Example / 1

- Via HTTP (HyperText Transfer Protocol)
  - more details in some weeks
- In Pseudocode (Java-like):
  ```java
  Socket s =
  new Socket("www.tuwien.ac.at", 80);
  s.send("GET / HTTP/1.0");
  ```
Request Example / 2

- TCP socket
  - Server listens on particular port
    - 80 in our example, standard port for HTTP
  - Client connects to the server host with its own client port
    - Free port is chosen

- Socket Pair
  - Server IP address + Port
  - Client IP address + Port
Request Example / 3

Problem
- IP needs IP destination address
- What is the IP address of "www.tuwien.ac.at"

Solution
1. Already cached by client
2. Domain Name System
   - Sends other messages!
3. HOSTS / HOSTS.TXT
How is IP packet delivered?
- IP makes only sense to IP layers
- Data link layer protocols own addressing
  - In same subnet
    - Requires MAC address in destination field
  - Other subnet via Routers
How is MAC address of another host found?

- Address Resolution Protocol (ARP)
- ARP cache
  - Hosts may fill cache when they see frames
Address Resolution Protocol

ARP
- Provides a mapping between two different forms of addresses

Ethernet
- RFC 826
- 32-bit IP and 48-bit ethernet
- Ethernet specific protocol

Exists in every TCP/IP implementation
- Automatically without intervention of Administrator
Reverse Address Resolution Protocol

- **RARP**
  - Maps Hardware Addresses to IP
  - RFC 903

- **Original task**
  - Obtain IP address on booting
    - Only IP address
  - Today replaced by DHCP
Dynamic Host Configuration Protocol (DHCP)

- RFC 2131
- Passing configuration information to hosts
  - On TCP networks
- Based on BOOTP (Bootstrap) (RFC 951)
  - DHCP allows transmission of larger options
- UDP as transport protocol
  - DHCP server port 67, DHCP client port 68
DHCP Goals

- Delivery of host-specific configuration parameters
  - from a DHCP server to a host
  - key-value pairs stored at server

- Allocation of network addresses to host
  - Eg. Client requests use of an IP address
DHCP Address assignment

- **Automatic assignment**
  - Permanent IP address to a client

- **Dynamic allocation**
  - Assignment of IP address for a limited time
  - Reassigning free IP addresses
DHCP Client-Server Protocol

- Assumption
  - client does not know its IP address!

1. Client broadcasts message "DHCPDISCOVER" on local physical subnet
   - Client's hardware address (eg. MAC address)

2. (Multiple) Server respond DHCPOFFER messages
   - Includes client's IP address
   - Client's Lease (expiration time)

3. Client chooses one Server that sent DHCPOFFER
   - Verification of server parameter
   - Sends DHCPREQUEST message

4. Server sends DHCPACK
   - Contains configuration parameters
DHCP

- Information valid as long as lease
  - No guarantee IP address is valid any longer
- Client may send RENEW messages
  - Timer watches lease expiration
  - Gets a new lease from DHCP server
- DHCP for IPv6 (RFC 3315)
  - Different messages than DHCP for IPv6
  - More configuration options than DHCP for IPv4
    - Eg. NIS+, NTP
  - Authorization
Stateless Address Configuration

- Stateless means
  - No DHCP server required
  - No specific configuration required
- IPv6 only
- RFC 2462
- IPv6 Interface ID (64 bit)
  - Created based on 48-bit MAC address
  - Verified with routers that it is unique
- 64 bit Prefixes determined from routers
  - Global Routing Prefix & Subnet ID
Any host has a routing table

Which physical interface to use for outgoing IP datagrams

<table>
<thead>
<tr>
<th>Destination IP</th>
<th>Next Hop Router</th>
<th>Flags</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>UH (H=Host)</td>
<td>Lo0</td>
</tr>
<tr>
<td>128.131.172.25</td>
<td>128.131.172.72</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Routing / 2

- Target host is determined via
  1. Routing table has entry that matches complete destination IP
     - Send packet to this router / interface
  2. Routing table has entry that matches destination network ID
     - Send packet to this router / interface
  3. Search routing table for default entry
     1. Send packet to this router / interface
Remote Login

Agenda

- RLogin
- Telnet
- SSH
- X-Window
Remote Login

- RLogin
  - one of the first remote login tools
  - Clear-text passwords
  - Allows bypassing of passwords
    - Security Problem
Communication between
- Any host
- Any terminal

RFC 854

Network Virtual Terminal (NVT)
- Lowest common denominator terminal
- All Telnet terminals shall conform to NVT

NVT Printer

NVT Keyboard
Telnet process model

Telnet client
  Terminal driver
  TCP/IP

Telnet server
  TCP/IP
  Pseudo Terminal driver

TCP Connection
NVT Ascii
- 7-bit US variant used in most Internet protocols
  - SMTP, HTTP, FTP, ...
  - Defines allowed symbols for these protocols
- 7-bit character sent as 8-bit (high-order bit = 0)
- Allows specific symbols
  - Those with high-order bit = 1
End-of-line symbol

2-character sequence
- CR (carriage Return)
- LF (Linefeed)
- \n
Carriage Return symbol itself
- Sent as \0 (CR NUL)
Telnet / 3

- Commands
  - 0xFF (255) (= Interpretate as Command)
  - Command-byte follows
Telnet Command

- Exists on most systems
  - `telnet <host> [<port>]` (default port: 23)
  - "Internet terminal"
  - Telnet server: telnetd
  - Windows Telnet server: start via Control Panel

- Data sent in the clear
- Passwords in the clear
  - Not widely used extensions/options for encryption

- Importance of Telnet
  - Debugging Tool
  - NVT Ascii used by most Application layer protocols
Telnet Example / 1
Remote Login

telnet compaq1.infosys.tuwien.ac.at
Suse Linux release 8.1
Kernel 2.4.2
login: joe
Password:
Last login: Tue Mar 22 ... from dellpc05.
... 
-bash-3.00$
Telnet Example / 2
Debug HTTP

telnet www.tuwien.ac.at 80
Trying 128.131.172.239...
Connected to pent21.infosys.tuwien.ac.at.
Escape character is '^[].'
GET / HTTP/1.0

HTTP/1.1 200 OK
Date: Fri, 18 Mar 2005 15:51:59 GMT
Server: Apache/1.3.26 Ben-SSL/1.48 (Unix) PHP/4.1.0
Last-Modified: Tue, 15 Mar 2005 08:21:32 GMT
ETag: "109eb-1ae2-42369b0c"
Accept-Ranges: bytes
Content-Length: 6882
Connection: close
Content-Type: text/html

<!doctype html public "^-//w3c//dtd html 3.2//en">
<html lang="de">
<head>
<title>TU Wien</title>
<LINK rel="stylesheet" type=
"text/css" href="styles/homepage.css">
...
Connection to host lost.
X-Window / 1

- Graphical windows on remote hosts
- X-Client
  - End-user application run on (remote) hosts
    - Terminal
    - Editor
    - ...
  - Sends messages to client
- X-Server
  - Renders the messages at the end-users host
  - Gets input from keyboard/mouse and sends it to X-client
- Be aware: Server vs. Client
  - X Server provides rendering services to the clients
X - Window Protocol

- Origin at MIT
- Currently at X.ORG
  - X11
- Usually on TCP (ports 6000-6063)
- Initial negotiation phase
- RPC like messages
  - CreateWindow, DestroyWindow
  - SetInputFocus
  - ClearArea
  - FillPoly
  - Bell
- X-client initiates the connection
Other graphical remoting tools

- **VNC**
  - Remote Frame Buffer protocol
    - One primitive operation
      - "put a rectangle of pixel data at a given x,y position"
    - stateless
  - Remote access to graphical user interfaces
    - X11, Windows, Mac

- **RDP – Microsoft Remote Desktop Protocol**
  - Remote administration of Windows Systems
  - Protocol not published
  - Performs better than X
Secure Shell (SSH)

- Protocol for secure
  - Remote Login
  - Other secure network services
- Strong encryption
- Server Authentication
- Integrity protection
- May provide compression (zlib, RFC1950/1951)
- Type of service negotiated
  - Public key algorithm
  - Symmetric algorithm
  - Message authentication algorithm
- RFC 4250-4256
  - Recently "Internet Proposed Standard"
Secure Shell (SSH)

- Standard methods
  - Interactive shell sessions
  - Remote execution of commands
  - Forwarding (tunneling) arbitrary TCP/IP ports
  - X11 connections

- More details
  - Later in this lecture about security protocols
Ping, Traceroute
Based on ICMP
- Sends an ICMP echo query request to a particular host
- Receives ICMP echo reply
- Identifier transmitted
  - Often sender process number (=ping process)
- Sequence number
  - Identification of the packet
  - Incremented at each send

Exists on most operating systems
Ping often blocked by firewalls
joe@mail:~$ ping localhost
PING mail (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.0 ms
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.0 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.0 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.0 ms

--- mail ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
Traceroute / 1

- Determines the route to a specified target host (via hosts and routers)
- IP header has 8-bit TTL (Time-to-live) field
  - Sender initializes this field to some value
  - Usually 64
  - To avoid endless loops
- Router detects IP datagram with TTL 0 or 1
  - Router throws away the datagram
  - Sends an ICMP message "time exceeded" to originating host
  - TTL > 1 datagram forwarded and TTL decremented by 1
- Today firewalls often block ICMP messages
Traceroute functionality (Pseudocode)

boolean hostFound = false;
int port = 30000; // no host shall have a service running this port
int ttl = 0;

while(!hostFound) {
    try {
        ttl = ttl + 1;
        sendUPD(targetHost, port, ttl)
    } catch (ICMP_TTLExceeded ttlExcptl) {
        System.out.println("Host: " + ttlExcptl.host);
    } catch (ICM_PortUnreachable pue) {
        System.out.println("Final port reached!");
        hostFound = true;
    }
}
Traceroute / 3
Traceroute example

/users/home6/e9425196 36% traceroute www.apache.org

traceroute: Warning: Multiple interfaces found; using 193.170.75.14 @ lan2
traceroute to www.apache.org (192.87.106.226), 30 hops max, 40 byte packets
1  193.170.75.254 (193.170.75.254)  1.357 ms  1.247 ms  1.251 ms
2  192.35.243.25 (192.35.243.25)  0.774 ms  0.782 ms  0.852 ms
3  defcon-in.kom.tuwien.ac.at (192.35.241.35)  0.751 ms  0.454 ms  0.451 ms
4  192.35.241.116 (192.35.241.116)  0.637 ms  0.732 ms  0.750 ms
5  193.171.13.9 (193.171.13.9)  1.440 ms  1.440 ms  1.233 ms
6  193.171.23.33 (193.171.23.33)  1.411 ms  1.748 ms  1.618 ms
7  aconet.at1.at.geant.net (62.40.103.1)  1.955 ms  1.712 ms  2.148 ms
8  at.de2.de.geant.net (62.40.96.58)  13.938 ms  14.032 ms  14.421 ms
9  de2-2.de1.de.geant.net (62.40.96.54)  13.668 ms  24.610 ms  14.290 ms
10 de.nl1.nl.geant.net (62.40.96.102)  20.278 ms  24.153 ms  20.409 ms
11 surfnet-gw.nl1.nl.geant.net (62.40.103.98)  20.475 ms  20.693 ms  20.463 ms
12 PO11-0.CR1.Amsterdam1.surf.net (145.145.166.33)  20.519 ms  20.312 ms  30.719 ms
13 PO0-0.AR5.Amsterdam1.surf.net (145.145.162.2)  20.465 ms  22.724 ms  20.615 ms
14 Te1-1.5W14.Amsterdam1.surf.net (145.145.140.158)  20.362 ms  20.828 ms  20.284 ms
15 * * * *