Internet Protocols by Example

Internet 101 revision in 90 minutes

Philipp S. Tiesel – philipp@inet.tu-berlin.de
Slides credits to Anja Feldmann
Internet Protocols by Example

What happens when accessing a Web page:

- **Application Layer**
  - DNS, HTTP, TLS...

- **Transport Layer**
  - UDP, TCP

- **Network Layer**
  - IPv4 (+DHCP +ARP)
  - IPv6 (+auto configuration +neighbor discovery)

➢ See [NPA](#) material for references
DNS Example

- Host at cis.poly.edu wants IP address for gaia.cs.umass.edu
The HTTP Protocol

HTTP: Hypertext transfer protocol

- Application layer protocol for the Web
- Client/server model
  - *Client*: browser that requests, receives, "displays" web objects
  - *Server*: Web server sends objects in response to requests

- Also used as part of many other application layer protocol
UDP: User Datagram Protocol

- “Bare bones” Internet transport protocol
- “Best effort” service, UDP segments may be:
  - Lost
  - Delivered out of order to application
- Connectionless:
  - No handshaking between UDP sender, receiver
  - Each UDP segment handled independently of others

TCP/UDP segment format:

<table>
<thead>
<tr>
<th>source port #</th>
<th>dest port #</th>
</tr>
</thead>
<tbody>
<tr>
<td>other header fields</td>
<td></td>
</tr>
<tr>
<td>application data (message)</td>
<td></td>
</tr>
<tr>
<td>TCP/UDP segment format</td>
<td></td>
</tr>
</tbody>
</table>
TCP: Overview

- **Point-to-point:**
  - One sender, one receiver

- **Reliable, in-order byte stream:**
  - No “message boundaries”

- **Pipelined:**
  - TCP congestion and flow control set window size

- **Full duplex data:**
  - Bi-directional data flow in one connection

- **MSS: maximum segment size**

- **Connection-oriented:**
  - Handshaking (exchange of control msgs) init’s sender, receiver state before data exchange

- **Flow controlled:**
  - Sender will not overwhelm receiver

- **Congestion controlled:**
  - Sender will not overwhelm the network

RFCs: 793, 1122, 1323, 2018, 2581
## TCP Segment Structure

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source port #</td>
<td>Source port number</td>
</tr>
<tr>
<td>dest port #</td>
<td>Destination port number</td>
</tr>
<tr>
<td>sequence number</td>
<td>Sequence number</td>
</tr>
<tr>
<td>acknowledgement number</td>
<td>Acknowledgement number</td>
</tr>
<tr>
<td>head len</td>
<td>Header length</td>
</tr>
<tr>
<td>not used</td>
<td>Not used</td>
</tr>
<tr>
<td>URG</td>
<td>URG: urgent data (generally not used)</td>
</tr>
<tr>
<td>ACK</td>
<td>ACK: ACK # valid</td>
</tr>
<tr>
<td>PSH</td>
<td>PSH: push data now (generally not used)</td>
</tr>
<tr>
<td>RST, SYN, FIN</td>
<td>RST, SYN, FIN: connection estab (setup, teardown commands)</td>
</tr>
<tr>
<td>rcvr window size</td>
<td>rcvr window size</td>
</tr>
<tr>
<td>checksum</td>
<td>Checksum</td>
</tr>
<tr>
<td>ptr urgent data</td>
<td>ptr urgent data</td>
</tr>
<tr>
<td>Options (variable length)</td>
<td>Options (variable length)</td>
</tr>
<tr>
<td>application data (variable length)</td>
<td>application data (variable length)</td>
</tr>
</tbody>
</table>

**Internet checksum (as in UDP)**

**Counting by bytes of data (not segments!):**

**# bytes rcvr willing to accept:**
TCP state machine
TCP Congestion Control

- End-end control (no network assistance)
- TCP throughput limited by rcvr window (flow control)
- Transmission rate limited by congestion window size, $cwnd$, over segments:

  - $w$ segments, each with MSS bytes sent in one RTT
IPv4 datagram format

- IP protocol version number (32 bits)
- Header length (bytes)
- "Type" of data
- 16-bit identifier
- Flgs
- Fragment length
- Time to live
- Upper layer
- Internet checksum
- 32 bit source IP address
- 32 bit destination IP address
- Options (if any)
- Data (variable length, typically a TCP or UDP segment)
- Total datagram length (bytes)
- For fragmentation/reassembly

- Max number remaining hops (decremented at each router)
- Upper layer protocol to deliver payload to

E.g. timestamp, record route taken, specify list of routers to visit.
IPv6 datagram format

- IP protocol version number
- "Type" of data (bytes)
- Payload length (variable length, typically a TCP or UDP segment)
- Upper layer protocol to deliver payload to or type of IPv6 extension header

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP version</td>
<td>8 bits</td>
</tr>
<tr>
<td>Traffic class</td>
<td>8 bits, defines the type of traffic</td>
</tr>
<tr>
<td>Flow label</td>
<td>20 bits, identifies related traffic</td>
</tr>
<tr>
<td>Payload length</td>
<td>16 bits, length of the payload in bytes</td>
</tr>
<tr>
<td>Next header</td>
<td>16 bits, number of the next protocol</td>
</tr>
<tr>
<td>Hop limit</td>
<td>16 bits, max number of remaining hops</td>
</tr>
<tr>
<td>Source IP</td>
<td>128 bits, identifies the sender of the datagram</td>
</tr>
<tr>
<td>Destination IP</td>
<td>128 bits, identifies the receiver of the datagram</td>
</tr>
<tr>
<td>Data</td>
<td>Variable length, typically a TCP or UDP segment</td>
</tr>
</tbody>
</table>

Identifier for related traffic
Max number remaining hops (decremented at each router)
CIDR: Classless InterDomain Routing

- Subnet portion of address of arbitrary length
- Address format: a.b.c.d/x, where x is # bits in subnet portion of address

```
11001000  00010111  00010000  00000000
```

200.23.16.0/23
How does a host get an IP addresses?

- Hard-coded by system admin
- DHCP / DHCPv6
  - Dynamic Host Configuration Protocol
    - Request the address from a server
- IPv6 SLAAC
  - Stateless Address Auto-Configuration
    - Router advertise the IPv6 prefix
    - Hosts add an Interface Identifier as Host-Part