NAT, IPv6, & UDP

CS640, 2015-03-03

Announcements
● Assignment #3 released

Overview
● Network Address Translation (NAT)
● IPv6
● Transport layer
● User Datagram Protocol (UDP)

Network Address Translation (NAT)
● Hacky solution to the IPv4 address exhaustion problem
● Assign private IP addresses to hosts within a network
  ○ Private IPs only used within a network
  ○ Same private IPs may be used for hosts in another network
  ○ Reserved ranges: 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16
● NAT translates private IP into public IP when traffic leaves the network
  ○ Public IPs are globally unique within the Internet
  ○ Temporarily map one private IP to one public IP
    ■ Limits the number of hosts in the network that can talk to hosts in the Internet to the number of public IPs
  ○ Temporarily map private IP + port number to public IP + port number
    ■ Number of process within the network that talk to processes in the Internet is limited by number of public IPs x number of ports
    ■ Most common NAT approach
  ○ Need to recalculate packet checksums when you change IPs
● NAT translates public IP (& port) back to private IP (& port) for packets in the other direction
● Port forwarding -- configure NAT to always forward packets with a specific destination port to a specific host

IPv6 Addresses
● **Why do we need IPv6? Why can’t we just use NAT?**
● Address space
  ○ IPv4 has 32-bit addresses => 4.3 billion addresses
    ■ Over 7.1 billion people on earth (growing at 345K people per day)
    ■ Estimate there will be 20 billion devices connected to the Internet by 2016
  ○ IPv6 has 128-bit address => 3.4 x 10^38 addresses
    ■ 6 x 10^22 address for each square foot of the earth’s surface
    ■ Entire IPv4 address space for every star in the universe
  ○ IPv4-compatible IPv6
    ■ Zero-extend a 32-bit IPv4 address to 128 bits (96-bits of 0s + 32-bits of IPv4)
    ■ Used for dual-stack machines that speak both IPv4 and IPv6
IPV4-MAPPED IPV6
- Prefix 32-bit IPv4 address with 2 bytes of all ones, then zero-extend to 128 bits (80-bits of 0s, 16-bits of 1s, and 32-bits of IPv4 address)
- Used for machines that aren’t IPv6 compatible

ADDRESS NOTATION
- Write each 16-bit piece in hexadecimal, separated by colons
  - E.g., 47CD:1234:4422:AC02:0022:1234:A456:0124
- Skip writing one long sequence of 0s
  - E.g., 47CD:0000:0000:0000:0000:0000:A456:0124 → 47CD::A456:0124
- Write IPv4-compatible IPv6
  - E.g., 0000:0000:0000:0000:0000:0000:8069:0E7A → ::128.105.14.122
- Write IPv4-mapped IPv6

IPv6 FEATURES
- HEADER
  - Version -- 6; same position as in IPv4
  - Class -- used for quality of service (QoS) similar to DSCP/ToS in IPv4
  - Flow label -- also used for QoS
  - Payload length -- length of packet in bytes, excluding IPv6 header
  - Next header -- identifier for next type of header, either transport protocol (TCP, UDP, etc.) or special header (replaces IP options)
  - TTL -- maximum number of hops to traverse; same as IPv4

**What are key differences you notice from the IPv4 header?**
- No fragmentation fields -- put in special header
- Options included as a special header
- No separate measurement of header length (not needed due to options put in a special header that follows)
- Header is double the size (40 bytes vs. 20 bytes)
  - Not bad, given that number of bits for addresses quadrupled
IPv4 to IPv6 transition

- Need incremental deployment plan
- World IPv6 day on June 6, 2012
  - April 2014: 13% of ASes in North America advertise IPv6 prefixes; 20% in Asia Pacific
  - March 3, 2014
    - 14% of Alexa Top 1000 sites accessible via IPv6
    - UW-Madison has 6% of hosts reachable via IPv6
- Dual stack
  - Routers and hosts run both IPv4 and IPv6 and process packets based on version field
  - IPv6 address assigned to node could be IPv4 mapped to IPv6 or a completely different address
- Tunnels
  - To send an IPv6 packet over IPv4-only portion of the network, encapsulate IPv6 packet in IPv4 packet
    - I.e., take IPv6 header and all following headers and payload and add Ethernet and IPv4 header beforehand
    - Routers only look at Ethernet header and first IP header, except routers which transition from IPv6 to IPv4 portions of network (and vice versa)
  - Vice versa to send IPv4 packet over IPv6-only portion of the network
Transport Layer

- Where are we in the stack?

```
+----------------+    +----------------+    +----------------+    +----------------+
| Application    |    | Transport      |    | Network         |    | Link            |
|                |    |                 |    |                 |    | Physical        |
```

- Network layer is best effort and focused on hop-by-hop communication
- **What undesirable things might the network layer do?**
  - Drop messages
  - Reorder messages
  - Deliver duplicate copies of a given message
  - Limit messages to some finite size
  - Deliver messages after an arbitrarily long delay
- Need end-to-end communication channel that provides some guarantees
  - Reliable delivery -- guaranteed message delivery
  - In-order delivery -- messages delivered in the same order they were sent
  - Non-replicated delivery -- at most one copy of each message delivered
  - Flow control -- rate of sending can be controlled by receiver (e.g., if receiver cannot process data as fast as it is arriving)
  - Congestion control -- rate of sending can be adjusted to avoid network overload and reduce loss due to overload
- Recall, use ports for multiplexing/demultiplexing between network layer and application layer
  - Server “listens” for connections on well known port numbers
    - HTTP -- 80
    - HTTPS (TLS/SSL) -- 443
    - SSH -- 22
    - DNS -- 53
  - Client picks a random unused port
- Transport protocols
  - User Datagram Protocol (UDP)
  - Transmission Control Protocol (TCP) -- most widely used; many
  - Real-Time Transport Protocol (RTP)
- Transport layer mechanisms happen at end-hosts -- routers and switches only deal with network layer and below
User Datagram Protocol (UDP)

- Minimalist transport protocol
  - Connectionless -- no explicit exchange of packets to establish an end-to-end communication channel
  - Multiplexes/demultiplexes messages between network and applications
  - Checksum -- for bit-level reliability; checksum is optional
- UDP header

<table>
<thead>
<tr>
<th>0</th>
<th>16</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Port</td>
<td>Destination Port</td>
<td>Checksum</td>
</tr>
<tr>
<td>Checksum</td>
<td>Length</td>
<td>Payload</td>
</tr>
</tbody>
</table>

- Checksum computed over UDP header, payload, and pseudo header
  - Pseudo header includes: 1-byte of 0s, protocol (from IP header), length (again), and source/destination IP address (from IP header)

**What are some benefits of UDP?**

- Simple
  - To send: divide message into chunks, add UDP header to each chunk, pass to the network layer
  - To receive: lookup which process is using the port listed in the destination field, put the UDP payload in a buffer for that process
- Flexible -- no restrictions on how much data you send and when (i.e., no flow control or congestion control)

**What are some drawbacks of UDP?**

- Does not improve delivery guarantees -- packets may still be dropped, re-ordered, or duplicated
  - Could build some of these capabilities into application layer atop UDP
- No network awareness -- can continue to push packets into the network and cause more congestion

Applications using UDP

- Domain Name Service (DNS)
- Voice over IP (VoIP)