CS118 Discussion 1A, Week 9

Zengwen Yuan (zyuan [at] cs.ucla.edu)
Boelter Hall 9436, Friday 12:00—1:50 p.m.
Outline

• Project 2

• Homework solution

• Lecture review
Course Project 2: questions?

• Demo time survey! https://goo.gl/forms/LbNF2dT1GGvFPiPP2

• We want to implement byte-stream reliable data transfer
  • WND is in unit of bytes, not packets

• How to realize timeout?
  • alarm() + signal()

• Should we implement RTT estimation?
  • Depends on if you plan to realize congestion control
Homework solution

• Derivation of slotted ALOHA efficiency
• Derivation of unslotted ALOHA efficiency
• Derivation of CSMA/CD efficiency
Random access: slotted ALOHA

- Assumptions:
  - all frames same size
  - time divided into equal size slots (time to transmit 1 frame)
  - nodes start to transmit only slot beginning
  - nodes are synchronized
  - if 2 or more nodes transmit in slot, all nodes detect collision
CSMA/CD (collision detection)

- CSMA/CD: carrier sensing, deferral as in CSMA
  - collisions detected within short time
  - colliding transmissions aborted, reducing channel wastage
- collision detection:
  - easy in wired LANs: measure signal strengths, compare transmitted, received signals
  - difficult in wireless LANs: received signal strength overwhelmed by local transmission strength
ARP: address resolution protocol

• How to determine interface’s MAC address, knowing its IP address?

• ARP table: each IP node (host, router) on LAN has table
  • IP/MAC address mappings for some LAN nodes:
    • <IP address; MAC address; TTL>
ARP protocol in same LAN

- A wants to send datagram to B, but B’s MAC address not in A’s ARP table.
- A broadcasts ARP query packet, containing B's IP address (all nodes on LAN receive ARP query)
  - dest MAC address = FF-FF-FF-FF-FF-FF
- B receives ARP packet, replies to A with its (B's) MAC address
  - frame sent to A’s MAC address (unicast)
- A caches IP-to-MAC address pair in its ARP table until information becomes old (times out)
- ARP is “plug-and-play”: nodes create their ARP tables without intervention from net administrator
Three questions

• What’s the difference between IP address and MAC address?

• If we are using ARP, what should destination address, source address and frame type look like?

• If we are using IP, what should destination address, source address and frame type look like?
Ethernet

• Connectionless and unreliable protocol
  • Why doesn’t Ethernet provide reliable data transfer?
• MAC protocol: CSMA/CD + exponential backoff
  • Can we use CSMA/CD in wireless network?
• Switch-based Ethernet
  • No real broadcast channel anymore
  • Self-learning algorithm: support plug-and-play
  • **Differences between routing table, switch table and ARP table?**
MPLS: Multi-Protocol Label Switching
Routing VS. Switching

- Circuit-switch network: connection should be established before forwarding the data
  - At each hop, the circuit path is marked as a label
  - Data forwarding is based on label: $O(1)$ complexity
  - Vulnerable to link/node failures
- Packet-switched network: connectionless, packets are forwarded based on IP header
  - Longest prefix matching: $O(N)$ complexity
  - Robust to link/node failures
- Can we take advantage of both, while preventing any vulnerabilities?
Multi-Protocol Label Switching

- Idea: **switching technique** into **connectionless** network

- In IP routing table, each entry is associated with a label

- Neighboring routers exchange labels, and forms an index of next hop’s forwarding table

- When forwarding the packet, lookup the index only
  - Only the first hop performs longest prefix matching
Exchanging labels Between Routers

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Next</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1.1</td>
<td>0</td>
</tr>
<tr>
<td>10.3.3</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Next</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1.1</td>
<td>1</td>
</tr>
<tr>
<td>10.3.3</td>
<td>0</td>
</tr>
</tbody>
</table>
Exchanging labels Between Routers

Store remote labels

Allocate label for each entry

Prefix | Next | Label
--- | --- | ---
10.1.1 | 0 | 15
10.3.3 | 0 | 16

Label | Prefix | Next
--- | --- | ---
15 | 10.1.1 | 1
16 | 10.3.3 | 0
Forwarding Packets

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Next</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1.1</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>10.3.3</td>
<td>0</td>
<td>16</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Label</th>
<th>Prefix</th>
<th>Next</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>10.1.1</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>10.3.3</td>
<td>0</td>
</tr>
</tbody>
</table>

Longest Prefix Matching
Forwarding Packets

Label-based Switching
Other Benefits

• In IP table, the IP addresses are aggregated based on prefixes

• In MPLS, the IP addresses can be aggregated in more flexible ways
  • How: assign same label for IP addresses in the same category
  • Benefit: better management (e.g., offer different levels of performance using different labels)
Wireless and Mobile Network

• Wireless access: WIFI
  • CSMA/CA VS. CSMA/CD
  • RTS/CTS mechanism

• Mobility: MobileIP
  • Home network, visited network
  • Permanent address VS. care-of-address
  • Indirect (triangle) routing VS. direct routing

• Wireless and mobility are not necessarily correlated
  • Wireless without mobility?
  • Mobility without wireless?
A day in the life: scenario

- Comcast network: 68.80.0.0/13
- Google's network: 64.233.160.0/19
- School network: 68.80.2.0/24
- Web server: 64.233.169.105
- DNS server
- Browser

A user interacts with a browser on a laptop connected to the school network. The browser requests a web page from Google’s network, which is routed through the Comcast network. The request is handled by the Google web server located at 64.233.169.105.
A day in the life… connecting to the Internet

- connecting laptop needs to get its own IP address, addr of first-hop router, addr of DNS server: use **DHCP**

- **DHCP** request encapsulated in **UDP**, encapsulated in **IP**, encapsulated in **802.3 Ethernet**

- Ethernet frame **broadcast** (dest: FFFFFFFFFFFFFF) on LAN, received at router running **DHCP server**

- Ethernet **demuxed** to IP demuxed, UDP demuxed to **DHCP**
DHCP server formulates **DHCP ACK** containing client’s IP address, IP address of first-hop router for client, name & IP address of DNS server

- encapsulation at DHCP server, frame forwarded (switch learning) through LAN, demultiplexing at client
- DHCP client receives DHCP ACK reply

Client now has IP address, knows name & addr of DNS server, IP address of its first-hop router
A day in the life… ARP (before DNS, before HTTP)

- before sending *HTTP* request, need IP address of www.google.com: **DNS**
- DNS query created, encapsulated in UDP, encapsulated in IP, encapsulated in Eth. To send frame to router, need MAC address of router interface: **ARP**
- **ARP query** broadcast, received by router, which replies with **ARP reply** giving MAC address of router interface
- client now knows MAC address of first hop router, so can now send frame containing DNS query
A day in the life… using DNS

- IP datagram containing DNS query forwarded via LAN switch from client to 1st hop router

- IP datagram forwarded from campus network into Comcast network, routed (tables created by RIP, OSPF, IS-IS and/or BGP routing protocols) to DNS server

- demux’ed to DNS server

- DNS server replies to client with IP address of www.google.com
A day in the life…TCP connection carrying HTTP

- to send HTTP request, client first opens TCP socket to web server
- TCP SYN segment (step 1 in 3-way handshake) inter-domain routed to web server
- web server responds with TCP SYNACK (step 2 in 3-way handshake)
- TCP connection established!
A day in the life… HTTP request/reply

- HTTP request sent into TCP socket
- IP datagram containing HTTP request routed to www.google.com
- web server responds with HTTP reply (containing web page)
- IP datagram containing HTTP reply routed back to client
- web page finally (!!!) displayed