Applied Networks & Security

TCP/IP Protocol Suite

http://condor.depaul.edu/~jkrystof/it263/

John Kristoff
jtk@depaul.edu
ARP overview

• datalink to network layer address mapping
  • e.g. 0000.1234.abcd <--- 192.0.2.1
• Hosts and routers build ARP table/cache
  • ARP entries associated with a local interface
  • Timers used to age old table entries
• Potential security problems with ARP
  • No authentication, can lead to impersonation
Typical ARP process...

Sender
- Send L2 broadcast
- Fill in known target IP

Receiver
- Fill in missing fields
- Learn sender's IP/MAC
- Reply directly to sender
IP review

• IP provides just enough *connected-ness*
  • Global addressing
  • Hop-by-hop routing
• IP over everything
  • Ethernet, ATM, X.25, fiber, etc.
• Minimizes network state
• Unreliable datagram forwarding
TCP key features

• Sequencing
• Byte-stream delivery
• Connection-oriented
• Reliability
• Flow-control
• Congestion avoidance
TCP feature summary

Provides a completely reliable (no data duplication or loss), connection-oriented, full-duplex byte stream transport service that allows two application programs to form a connection, send data in either direction simultaneously and then terminate the connection.
Apparent contradiction

- IP offers best effort (unreliable) delivery
- TCP uses IP
- TCP provides completely reliable transfer
- How is this possible?
Achieving reliability

• Reliable connection start-up
• Reliable data transfer
  • Sender starts a timer
  • Receiver sends ACK when data arrives
  • Sender retransmits if timer expires before ACK is returned
• Reliable connection shutdown
Reliability illustrated
When do you retransmit?

- The time for an ACK to return depends on:
  - Distance between endpoints (propagation delay)
  - Network traffic conditions (congestion)
  - End system conditions (CPU, buffers)
- Packets can be lost, damaged or fragmented
- Network traffic conditions can change rapidly
Solving retransmission problem

- Keep running average of round trip time (RTT)
- Current average determines retransmission timer
- This is known as adaptive retransmission
- This is key to TCP's success
- How does each RTT sample affect the average?
  - What weight do you give each sample?
  - Higher weight means timer changes quickly
  - Lower weight means timer changes slowly
Adaptive retransmission illustrated
Flow control

• Match the sending rate with allowable receiver rate
• TCP uses a sliding window
  • Receiver advertises available buffer space
  • Also known as the window
  • Sender can transmit a full window without receiving an ACK for that transmitted data
• Ideally the window size allows pipe to remain full
Window size advertisement

• Each ACK carries receiver's current window size
  • Called the window advertisement
  • If zero, window is closed, no data can be sent
• Interpretation of window advertisement:
  • Receiver: I can accept \( X \) octets or less unless I tell you otherwise
Window size illustrated

Sender Events
- send data octets 1-1000
- send data octets 1001-2000
- send data octets 2001-2500
- receive ack for 1000
- receive ack for 2000
- receive ack for 2500
- send data octets 2501-3500
- send data octets 3501-4500
- receive ack for 3500
- receive ack for 4500
- receive ack for 4500

Receiver Events
- advertise window=2500
- ack up to 1000, window=1500
- ack up to 2000, window=500
- ack up to 2500, window=0
- application reads 2000 octets
- ack up to 2500, window=2000
- ack up to 3500, window=1000
- ack up to 4500, window=0
- application reads 1000 octets
- ack up to 4500, window=1000
Window size: another picture

(a) 

(b) 

(c)
Byte stream sequencing

- Each segment carries a sequence number
- Sequencing helps ensure in order delivery
- TCP sequence numbers are fixed at 32 bits
  - Byte stream is not limited to $2^{32}$ bytes
  - Sequence number space can wrap
- Each side has an initial sequence number (ISN)
  - Exchanged during connection establishment
- Receiver ACKs cumulative octets (bytes)
TCP segment illustrated

| bit 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
|------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
|      | Source Port | Destination Port |
|      | Sequence Number |
|      | Acknowledgement Number |
| HLEN | Reserved |
|      | URG ACK PSH RST SYN FIN |
|      | Window |
|      | Checksum |
|      | Urgent Pointer |
|      | Options (if any) |
|      | Padding |
|      | Data |
|      | ... |
Application multiplexing

- OS independent identifier for a network process
- Each application assigned a unique 16-bit integer
  - Called a port number
- Server applications
  - Use standard, well-known port numbers
  - Usually low numbered port numbers
- Clients
  - Obtain unused number from protocol software
  - Usually uses high numbered port numbers
TCP connection start-up

• The three-way handshake used
• Servers use a passive open
  • Application sits waiting on an open port
• Clients use an active open
  • Application requests a connection to server
• Initial sequence number (ISN) exchange is the primary goal
• Other parameters/options can also be exchanged
  • e.g. Window scale, maximum segment size, etc.
3-way handshake illustrated

Time

Host A
- Send SYN seq=x
- Receive SYN + ACK
- Send ACK y+1

Host B
- Receive SYN
- Send SYN seq=y, ACK x+1
- Receive ACK
Connection shutdown illustrated
Congestion principles

- Flow control
  - Matching the sending and receiving rates
- Congestion control
  - Active response to network overload conditions
  - End hosts cannot control congestion per se
  - Network devices (routers) do this
- Congestion avoidance
  - Cautionary response to presumed conditions
  - TCP does this
TCP congestion control

• Recall sliding window (advertised window)
  • Receiver based control of sending rate
• Congestion window is sender based control
• Sender transmits min(cwnd, advertised window)
  • This value is the *transmission window*
• TCP sender infers network conditions and adjusts
TCP retransmission

- TCP starts timer after sending a segment
- If ACK returns, reset timer
- If time-out occurs, retransmit and increase timer
  - This is a back-off process
- Can't retransmit forever, need some upper bound
- Eventually TCP would give up
  - Maximum time-out must be at least 60 seconds
UDP message format

<table>
<thead>
<tr>
<th>bit 0-3</th>
<th>bit 4-7</th>
<th>bit 8-15</th>
<th>bit 16-23</th>
<th>bit 24-31</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Port</td>
<td>Destination Port</td>
<td>Length</td>
<td>Checksum</td>
<td>Data</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>