Advanced Unix System Administration

Lecture 13
April 7, 2008

Steven Luo
<sluo+decal@OCF.Berkeley.EDU>

The Transport Layer

- Transmission Control Protocol (TCP)
  - Much more elaborate and featureful than UDP
  - Reliable, stream-oriented, connection-oriented
  - Applications send streams of data which TCP packages into packets and sends over the network
  - Correct and in-order delivery is guaranteed even on unreliable networks

The Transport Layer

- The TCP packet
  - Source port (16 bits), destination port (16 bits)
  - Sequence number (32 bits)
  - Acknowledgment number (32 bits)
  - Data offset (4 bits), gives size of header in 32-bit words, reserved field (4 bits)
  - TCP flags (8 bits): CWR, ECE, URG, ACK, PSH, RST, SYN, FIN
  - Window size (16 bits), gives number of bytes sender is willing to receive before ACK

The Transport Layer

- The TCP packet con't
  - Checksum (16 bits)
  - Urgent pointer (16 bits)
  - Options, padded to an integral multiple of 32 bits
  - Data
- TCP connections
  - Three phases of connections: establishment, data transfer, teardown

The Transport Layer

- TCP connections con't
  - Establishment (3-way handshake):
    - Client sends packet with SYN set to server
    - Server replies with SYN/ACK
    - Client sends ACK
    - Unexpected/unwanted connections rejected with RST
  - Data transfer
    - The sequence number of the packets with SYN set give initial sequence numbers (ISNs)
    - Each byte of data in the stream is given a sequence number, starting with ISN+1

The Transport Layer

- TCP connections con't
  - Data transfer con't
    - Receipt of each packet is acknowledged with an ACK with ack number set to the last byte in sequence received + 1
      - Selective packet acknowledgment is available as an option
    - Packets not acknowledged will be retransmitted; duplicates will be dropped silently
    - Number of bytes a sender will send before waiting for ACK is controlled by the window size

The Transport Layer

- TCP connections con't
  - Data transfer con't
    - TCP implementations use data such as retransmissions, ACK rates, and the like to adjust to conditions (via changing the window, slowing transmission rate, etc.)
  - Teardown
    - FIN is sent to announce that one has no more data to send
    - That half of the connection is closed when the ACK reply is received

The Transport Layer

- Application considerations
  - For short communications that may happen frequently/quickly, UDP is used
  - Longer conversations, anything that needs to happen reliably, etc. should be done over TCP
  - Stream connections that can't take the overhead or connection handling of TCP may use UDP, but this requires careful application design
  - By volume, TCP traffic dominates on the Internet

Packet Filtering and Firewalls

- At the simplest level, this is really easy to do
  - Hooks into parts of the network stack to examine attributes of packets
  - Decision to drop or allow through packet based on some simple matching rules
  - The lower the level you confine your examination to, the faster it'll be
    - This gives you less information, of course
    - Good filtering is a tradeoff between speed and flexibility

Packet Filtering and Firewalls

- State
  - Stateless packet filtering can't give you information about TCP connections
  - Having the firewall engine keep connection state allows real filtering of incoming connections
  - Once you're keeping state, other statistics such as connection rate can also be useful
  - Speed can be a problem – but you can also use state to speed up packet processing

Packet Filtering and Firewalls

- Packet mangling
  - It's not a long step towards actually changing the packets based on matched rules
  - Depending on where the hooks are, one can change the destination of the packet, its attributes, ...
- Notable implementations
  - netfilter (Linux), pf (OpenBSD and other BSD), ipfilter (portable) are quite flexible
  - Most Windows firewalls are simpler packet filters