Advanced Unix System Administration

Lecture 14
November 3, 2008

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

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

Network Address Translation

- Parts of the IPv4 address space are designated non-public
- We can alleviate the IPv4 address crunch if we find a way to route traffic to and from these hosts
- NAT is a clever hack to do this
  - In its simplest form, just map public IPs to private ones one-to-one with some mangling
  - Not useful for the conserving IPs application

Network Address Translation

- Port translation
  - We need some way of keeping track of who sent the outbound traffic, if we're to route the replies correctly
  - Solution:
    - Mangle the source port
    - Keep track of the source ports corresponding to each client connection, and route traffic accordingly
  - This limits the number of outbound connections per client, but that's usually not an issue

Network Address Translation

- Problems with NAT
  - Applications (FTP) frequently include IPs and port numbers in their protocols, so we need to mangle these too
  - Incoming connections can't be handled, so direct connection protocols have a hard time
- Philosophical objections to NAT
  - Hosts behind NAT on the Internet aren't really full peers anymore
  - Only delays the inevitable

The Domain Name System

- People aren't very good at remembering numbers
  - And they're definitely no good at remembering IPv6 addresses!
- Classic solution: the hosts file
  - Domains maintain hosts files, which are distributed and synchronized via FTP
  - Simple, but absolutely does not scale
- DNS provides a way of providing names in a scalable, distributed way

The Domain Name System

- Structure of DNS
  - Hierarchical system, each part of hierarchy separated by dots
  - DNS servers are delegated authority over parts of the DNS zone by servers closer to the root of the tree
  - “13” “root” DNS servers store the delegations for the lowest level
- Recursive name resolution
  - Start at root, inquire for record at each level until we get what we want