Announcements

Guest Lecturer on 9/19: Noel Chiappa.
BE THERE!
Lectures 1-4 are available.
Everybody, please send email (ji+ir@cs.columbia.edu) telling me if you’re taking or auditing the class, and if you’re a CVN student.
BSD Routing Code

- Cribbed from Wright & Stevens, TCP/IP Illustrated, Volume 2.
- Strictly speaking, it is mostly the forwarding code.
- Code taken from FreeBSD 4.6-STABLE.

- The forwarding code lives in the kernel.
- There is an API to modify the forwarding tables.
  - User commands (route(8) and arp(8)) use it to change the forwarding table.
  - Routing daemons (routed(8), gated(8)) use it to reflect routing table changes into the forwarding table.
What's in the Routing Table

```
bsdi# netstat -r -f inet
Routing tables
Internet:
Destination  Gateway         Flags  Refs  Use  Netif  Expire
default      140.252.13.33  UGSc   3    61299 dc0
127          127.0.0.1      UGRSc  0     0     lo0
127.0.0.1    127.0.0.1      UH     3    51653 lo0
128.32.33.5  140.252.13.33  UGHS   2    16    dc0
140.252.13.32/27  link#1  UC     0     0     dc0
140.252.13.33  8:0:20:3:f6:42  UHLW  11    12    dc0  1143
140.252.13.34  0:0:c0:c2:9b:26  UHLW  3     1    dc0  432
140.252.13.65  140.252.13.66  UH     1     1    sl0
224          link#1        UC     0     0     dc0
224.0.0.1    link#1        UHLW   0     5    dc0  223
```

Ethernet, 140.252.13.32/27

To internet
Routing Messages

• Socket of type PF_ROUTE:
  – socket(PF_ROUTE, SOCK_RAW, AF_INET);
• Change the forwarding table by sending messages with sendmsg(2).
• Get notified of changes (through cloning, ICMP redirects, other routing daemons running) with recvmsg(2).
• The route(8) command uses it.
• Routing daemons (routed, gated) use it.

• Read the manual for route(4) and route(8).
• Read the source in /usr/src/sys/net/ and /usr/src/sys/netinet/
Forwarding Table Requirements

• Information:

<table>
<thead>
<tr>
<th>Key</th>
<th>Mask</th>
<th>Next hop router</th>
</tr>
</thead>
<tbody>
<tr>
<td>135.207.4.0</td>
<td>255.255.255.0</td>
<td>135.207.25.36</td>
</tr>
<tr>
<td>127.0.0.0</td>
<td>255.0.0.0</td>
<td>reject</td>
</tr>
<tr>
<td>default</td>
<td>255.0.0.0</td>
<td>135.207.31.1</td>
</tr>
</tbody>
</table>

• Operations:
  – Lookup, matching “longest prefix”.
  – Insert.
  – Delete

• Fast and compact:
  – In the kernel.
  – Lookups affect forwarding performance.
Multiprotocol Forwarding Table

- Need to support multiple protocol families.
- struct sockaddr: generic structure to store addresses.
- Examples: struct sockaddr_in, struct sockaddr_in6.
- One forwarding table maintained per address family.

- Forwarding table stored as a Patricia tree:
  - Practical Algorithm To Retrieve Information Coded In Alphanumeric.
  - A Patricia tree is a Trie where successive nodes with one child have been collapsed into one.
struct sockaddr_in

In /usr/include/netinet/in.h

typedef u_int_32 in_addr_t;

struct in_addr {
    in_addr_t s_addr;
};

struct sockaddr_in {
    u_char sin_len; /* 16 */
    u_char sin_family; /* AF_INET == 2 */
    u_short sin_port;
    struct in_addr sin_addr;
    char sin_zero[8];
};
struct sockaddr_in6

In /usr/include/netinet6/in6.h

struct in6_addr {
    ...
};

struct sockaddr_in6 {
    u_int_8    sin6_len;    /* 28 */
    u_int_8    sin6_family; /* AF_INET6 == 28 */
    u_int_16   sin6_port;
    u_int_32   sin6_flowinfo;
    struct     in6_addr   sin6_addr;
    u_int_32   sin6_scope_id;
};
FIBs in Hardware

- Commercial routers implement FIBs in hardware.
- Ternary CAMs (CAM=Content-Addressable Memory).
  - Key, mask, result.
  - Low density.
  - Low manufacturing volumes.
  - Expensive!
Packet classifiers

- FIBs are a special case of packet classifier.
- Many applications need to do similar lookups:
  - Firewalls.
  - Traffic directors (layer-4 switches (AITFOTL)).
  - DiffServ-aware routers.
- Active research area:
  - Papers in recent SIGCOMMs; look in http://www.acm.org/sigcomm/.