BOOTP and DHCP

Shivkumar Kalyanaraman
Rensselaer Polytechnic Institute
shivkuma@ecse.rpi.edu
http://www.ecse.rpi.edu/Homepages/shivkuma

Overview

- Bootstrapping (Diskless workstations)
- BOOTP
- Dynamic address allocation
- DHCP
- Ref: Chap 16, Doug Comer’s TCP/IP book

Bootstrapping

- Computer loads a simple boot program. The boot program loads operating system.
- On diskless machine, the computer needs to know the network address of the os file
- It needs to know its own IP address, subnet mask, IP address of default router, IP address of DNS server
- It only knows its h/w address.
Configuration

- Different nodes have different parameters
- Configuration = Setting the parameters
- Key parameters for IP hosts:
  - IP Address
  - Default router address
  - Subnet mask
  - Name
  - DNS server addresses

Method 1: Reverse ARP (RARP)

- What is the IP addr of a hardware address?
- Need RARP server to respond.
- Once IP address is gotten, it does a “tftp” to get its boot image.
- Design of the RARP server complex unlike ARP which is integrated into TCP/IP host implementations
  - Needs to maintain table for multiple hosts (/etc/ethers)
  - Kernel does not process/parse files => RARP is a user process. But does not run over IP.

RARP (contd)

- RARP cannot use IP
- Needs unique Ethernet frame type (0x8035)
- Works through a filter like BPF or nit_if/nit_pf streams modules (fig: A.1, A.2)
- Multiple RARP servers needed for reliability, but unlike ARP where only one reply is sent, each RARP sends a unicast reply => additional traffic
- Possibility of collision between RARP replies
- RARP servers cannot be consolidated since RARP requests are broadcasts => router cannot forward (relay) RARP requests
Key RARP limitations

- RARP is a user process but works over link layer directly =>
  - RARP server system dependent
  - Needs to interface with link layer driver directly => separate filters and direct access to hardware needed
- Returns only IP address
  - Booting and configuration params not returned even though there is space in packet
  - Host needs ICMP and TFTP to complete booting
- Can’t relay RARP requests to a central server.
  - Need RARP server per broadcast domain

Method 2: BOOTP

- Runs over UDP/IP! Issues w/ using UDP/IP:
  - IP software can broadcast (to 255.255.255.255) even if local IP address unknown => client broadcasts BOOTP request
  - Port number 67 for server and 68 for client (not an ephemeral port)
  - Delivers BOOTP reply to BOOTP client and not other UDP apps when reply is broadcast
  - Does not wake up other servers during broadcast reply

BOOTP (contd)

- BOOTP requests/replies sent w/ DF bit set.
- Server can send reply via broadcast or unicast:
  - For unicast reply, BOO TP server knows the IP address, but the link layer address is not in the ARP cache
  - Note that the server cannot send an ARP message because client does not know its IP address
  - Server can use ioctl(8) {or arp -s} to set the value of the cache => can do this only if it has permission
BOOTP features (contd)

- Else send broadcast reply
- Reply: IP Address, Boot Server IP address, Default Router, Boot file name, subnet mask
- More information, but still only a single packet exchange
- Client gets boot image using TFTP => booting still a 2-step process

Advantages of using UDP/IP:
- Bootstrapping can occur across a router via a relaying mechanism
- BOOTP uses checksum provided by UDP
- Multiple requests/replies
  - Process the first one
  - Client uses a transaction ID field to sort out replies
- Clients responsible for reliability =>
  - Uses timeout, retransmission & exponential backoff
  - Random initial timeout (between 0 & 4s): simultaneous reboot after power restoration.

BOOTP Message Format

<table>
<thead>
<tr>
<th>Operation</th>
<th>H/W Type</th>
<th>H/W Length</th>
<th>Hops</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>31b</td>
<td>16 B</td>
<td>64 B</td>
</tr>
</tbody>
</table>

- Transaction Identifier
- Seconds elapsed
- Unused
- Client IP Address
- Your IP Address
- Server IP Address
- Router IP Address
- Client H/W address
- Server Host Name
- Bootfile Name
- Vendor Specific Area
BOOTP Message (Cont)

- Operation: 1 = Request, 2 = Reply
- H/w type: 1 = Ethernet
- H/w Address Length
- Hops: Initialized to zero. Incremented by BOOTP relays (routers)

- Please tell me my address
- My client needs an address

- Your address is ...
- Your client’s address is ...

- Transaction ID: used to match responses with requests
- Seconds = Number of seconds since the client started to boot
- If a client knows its IP address, it places it in the Client IP address
- If server address/name fields are non-zero in the request, only the indicated host can answer the request
- Your IP Address: Clients IP address returned by the server

- Boot File name: Generic name like “unix” in the request. Full name in response.
- Vendor specific area: Misnomer. Also used for general purpose info.
- Magic cookie: First 4 octets = 99.130.83.99
- Type-length-value: (Contd on next slide)

<table>
<thead>
<tr>
<th>Item</th>
<th>Code</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Padding</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>Subnet mask</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Time of Day</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>End</td>
<td>255</td>
<td>-</td>
</tr>
</tbody>
</table>
Contents of Vendor-Specific Area

<table>
<thead>
<tr>
<th>Item</th>
<th>Code</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Routers</td>
<td>3</td>
<td>4n</td>
</tr>
<tr>
<td>Time Server</td>
<td>4</td>
<td>4n</td>
</tr>
<tr>
<td>IEN116 Server</td>
<td>5</td>
<td>4n</td>
</tr>
<tr>
<td>Domain server</td>
<td>6</td>
<td>4n</td>
</tr>
<tr>
<td>Log server</td>
<td>7</td>
<td>4n</td>
</tr>
<tr>
<td>Quote server</td>
<td>8</td>
<td>4n</td>
</tr>
<tr>
<td>LPR servers</td>
<td>9</td>
<td>4n</td>
</tr>
<tr>
<td>Impress servers</td>
<td>10</td>
<td>4n</td>
</tr>
<tr>
<td>RLP Server</td>
<td>11</td>
<td>4n</td>
</tr>
<tr>
<td>Host name</td>
<td>12</td>
<td>4n</td>
</tr>
<tr>
<td>Boot size</td>
<td>13</td>
<td>2</td>
</tr>
<tr>
<td>Reserved</td>
<td>128-254</td>
<td>-</td>
</tr>
</tbody>
</table>

Method 3: DHCP

- BOOTP limitation: cannot dynamically assign IP address
- Dynamic Host Configuration Protocol (DHCP)
  - Dynamic Host Configuration Protocol (DHCP) + Dynamic allocation of IP addresses => compatible with BOOTP. No new fields in header.
  - Addresses can be leased for a period. Reallocated to the same or other nodes after lease expiry.
  - Non-mobile computers can get a permanent address.

DHCP Message Format

<table>
<thead>
<tr>
<th>Operation</th>
<th>H/W Type</th>
<th>H/W Length</th>
<th>Hops</th>
<th>Transaction Identifier</th>
<th>Seconds elapsed</th>
<th>Flags</th>
<th>Client IP Address</th>
<th>Your IP Address</th>
<th>Server IP Address</th>
<th>Router IP Address</th>
<th>Client H/W address</th>
<th>16 B</th>
<th>Server Host Name</th>
<th>64 B</th>
<th>Bootfile Name</th>
<th>128 B</th>
<th>Options (Variable)</th>
</tr>
</thead>
</table>
DHCP Message Format

- Slightly modified version of BOOTP message
- A DHCP server can be programmed to answer BOOTP requests
- BOOTP’s Unused field renamed to Flags
- Only one bit of 16-bit Flags has been defined
  - Left-most flag bit = 1 ⇒ Servers, please reply using IP broadcast address
  - Servers by default send hardware unicast response
- Vendor specific field renamed to options
- Size increased to 312 bytes (from 64 bytes)
- Option type 53 specifies the "type of the message"

DHCP (contd)

<table>
<thead>
<tr>
<th>Type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DHCP Discover</td>
</tr>
<tr>
<td>2</td>
<td>DHCP Offer</td>
</tr>
<tr>
<td>3</td>
<td>DHCP Request</td>
</tr>
<tr>
<td>4</td>
<td>DHCP Decline</td>
</tr>
<tr>
<td>5</td>
<td>DHCP Ack</td>
</tr>
<tr>
<td>6</td>
<td>DHCP Nack</td>
</tr>
<tr>
<td>7</td>
<td>DHCP Release</td>
</tr>
</tbody>
</table>

- "Option overload"
- Server Host name and boot file name when unused for their original purpose could be used to code more options

DHCP State Diagram
DHCP states

- Boots => INITIALIZE state
- DHCPDISCOVER: broadcast request to servers => SELECT state
- DHCPOFFER (from server) => remain in SELECT
- DHCPREQUEST => select one of the offers and notify server (goto REQUEST state) about the lease
- DHCPACK => server Ok's request to lease => go to the BOUND state
- Renewal: after 50% of lease go to RENEW state
- Rebind: after 87.5% of time, if server has not responded, try again and go to REBIND.
- If server NACKs or lease expires, or client sends DHCPRELEASE, go to INITIALIZE, else come back to BOUND state

DHCP: Current Issues

- Interaction with DNS
- Should the names also be dynamically leased?
- Should the names be registered on DNS?
- How to work with a directory service (given a fixed name, find a temporary IP address)?
- Currently there are no protocols for dynamic DNS updates.

Summary

- RARP allows finding an IP address
- BOOTP allows default router, subnet mask, DNS
- DHCP allows dynamic allocation
- DHCP is backward compatible with BOOTP
Initialization: RFCs