RIPE ncc

DHCP introduction

Geert Jan de Groot

RIPE NCC

<GeertJan.deGroot@ripe.net>

RIPE ncc

Overview

- What
- Why
- How
- RARP/BOOTP
- DHCP
- Server setup examples
- Client setup examples

RIPE ncc

What

Dynamic Host Configuration is assignment of network parameters (IPnum, netmask, default, ...) from the network instead of local storage (disk, EEPROM)

RIPE ncc

Why?

- Network numbers are more likely to change these days than in the past (CIDR, changing provider requires renumbering)
- Network manager wants to be in control of network parameters for all machines on the net.
  Having these parameters on local inaccessible storage is no good!
- Experienced sysadmins use `rdist` etc but that's no good for DOS, Windows, MacOS, Win95, ...

Types of allocation

Manual allocation
Fixed relation between MAC address and IP number

Automatic allocation
New MAC address gets IP number assigned automatically (permanent)

Dynamic allocation
IP numbers are leased for limited time

How

- RARP
  Reverse-ARP protocol
- BOOTP
  BOOT Protocol
- DHCP
  Dynamic Host Configuration Protocol

RARP (Reverse ARP)

- ARP = IP address -> MAC address
  RARP = MAC address -> IP address
- RARP response gives only IP number, nothing else (no netmask, default, hostname, ...)
- Non forwardable across IP router (thus RARP server must be directly attached to network)
- Manual allocation only (DRARP exists, but...)
- Used by SUN (together with rpc.bootparamd & friends)

BOOTP (BOOT Protocol)

- Uses MAC address as searchkey
- BOOTP response includes IPnum, netmask, default route, bootfile, swapfile, log server, print server, time server, ...
- Uses IP, thus forwardable across IP router (thus BOOTP server may be on different net as requests can be relayed, e.g. Cisco's helper-address)
- Mainly used for manual allocation, address lease not possible
- Two families of protocols: CMU and RFC1084
- Packet size limitation: 'vendor-specific area' is only 64 bytes

BOOTP packet size limitation

A bootp packet has:

```
1       Request/Response
1       Hardware address type
1       Hardware address length
1       Request hop count
4       Transaction ID
2       Seconds since boot
2       Unused
4       Client IP address (filled by client) *
4       Client IP address (filled by server)
4       BOOTP server address *
4       Gateway IP address
16      Client hardware address
64      Server hostname
128     Boot filename
64      Vendor-specific data
```

Many ‘interesting options’ must fit in 64 bytes;
this field is usually too small

* filled in by client if known

DHCP (Dynamic Host Configuration Protocol)

Evolved from BOOTP, with additions:

- Allows address lease for limited time
  (thus allows all of Manual, Automatic
  and Dynamic allocation)
  Includes renewal mechanism
- Fancy negotiation mechanism
- Packet size limitation solved:
  seldom used BOOTP fields used if 64 bytes too small

RFC1541, draft-ietf-dhc-dhcp-XX.txt

DHCP flow

Sample flow:

```
Client   sends DHCPDISCOVER to find DHCP server
Server   sends DHCPOFFER
Client   sends DHCPREQUEST to selected DHCP server
Server   sends DHCPACK with committed addresses,
         reserves addresses etc
Client   notes timeout values and starts operating

At time t1:
Client   sends DHCPREQUEST to extend lease
Server   sends DHCPACK to extend lease
Client   notes renewed timeout values and continues

At shutdown:
Client   sends DHCPDECLINE when shut down
```

More situations are possible; see RFC.

Implementions

- WIDE
  client + server, freeware
  `ftp://sh.wide.ad.jp/WIDE/free-ware/dhcp/*`
- ISC
  currently server only, freeware
  `ftp://ftp.vix.com/pub/dhcp/*`
- WIN95
  client only
- PC/NFS
  client only
  (server included in other SUN packages)
- ... many more!

Server setup (using WIDE code)

```
# define different master entries for each subnet. . .
subnet30::rout=133.4.30.1:dht1=500:dht2=850:\
        :!snmk=255.255.255.224:brda=133.4.30.31:
```

- dht1: renew lease after 50% of lease time
- dht2: rebind after 85% of lease expired

Server setup (manual allocation)

```
# define different master entries for each subnet. . .
subnet30::rout=133.4.30.1:dht1=500:dht2=850:\
        :!snmk=255.255.255.224:brda=133.4.30.31:

# entries for manual allocation (DHCP, BOOTP)
3001:    :ipad=133.4.30.1:tblc=subnet30:\
         :clid="1:0x0080c75fdc03":
3002:    :ipad=133.4.30.2:tblc=subnet30:\
         :clid="1:0x08004600e5d5":
```

- clid = client ethernet address
- 3001, 3002 are just labels

Server setup (automatic allocation)

```
# define different master entries for each subnet. . .
subnet30::rout=133.4.30.1:dht1=500:dht2=850:\
        :!snmk=255.255.255.224:brda=133.4.30.31:

# entries for automatic allocation (DHCP)
3005:    :maxl=0xffffffff:ipad=133.4.30.5:\
         :tblc=subnet30:
3006:    :maxl=0xffffffff:ipad=133.4.30.6:\
         :tblc=subnet30:
```

- maxl=0xffffffff indicates infinite lease (permanent address)

Server setup (dynamic allocation)

```
# define different master entries for each subnet. . .
subnet30::rout=133.4.30.1:dht1=500:dht2=850:\
        :!snmk=255.255.255.224:brda=133.4.30.31:

# entries for dynamic allocation (DHCP)
3003:    :ipad=133.4.30.3:dfll=3600:maxl=7200:\
         :tblc=subnet30:
3004:    :ipad=133.4.30.4:dfll=3600:maxl=7200:\
         :tblc=subnet30:
```

- dfll is default lease time in seconds
- maxl is maximal lease time server will accept

RIPE ncc

Client setup

dhcpc ef0

That's all there is to it!