CHAPTER

Dynamic Host Configuration Protocol 18

What You Will Learn

In this chapter, you will learn how IP addresses are assigned in modern IP networks. You will learn how the Dynamic Host Configuration Protocol (DHCP) and related protocols, such as BOOTP, combine to allow IP addresses to be assigned to devices dynamically instead of by hand.

You will learn how users often struggle to find printers and servers whose IP addresses "jump around," and you will learn means of dealing with this issue.

When TCP/IP first became popular, configuration was never trivial and often complex. Whereas many clients needed only a handful of parameters, servers often required long lists of values. Operating systems had quickly outgrown single floppies, and most hosts now needed hard drives just to boot themselves into existence. Routers were in a class by themselves, especially when they connected more than two subnets—and in the days of expensive memory and secondary storage (hard drives), routers usually needed to load not only their configuration from a special server, but often their entire operating systems.

A once-popular movement to "diskless workstations" hyped devices that put all of their value into hefty processors while dispensing with expensive (and failure-prone) hard drives altogether. Semiconductor memory was not only prohibitively expensive in adequate quantities but universally volatile, meaning that the content did not carry over a power failure if shut down. How could routers and diskless workstations find the software and configuration information they needed when they were initially powered on?

RFC 951 addressed this situation by defining BOOTP, the bootstrap protocol, to find servers offering the software and configuration files routers and other devices needed on the subnet. The basic functions were extended in RFC 1542, which described relay agents that could be used to find BOOTP servers almost anywhere on a network. BOOTP did a good job at router software loading, but the configuration part (notably the IP addresses) assigned by the device's physical address had to be laboriously maintained by the BOOTP server administrator.

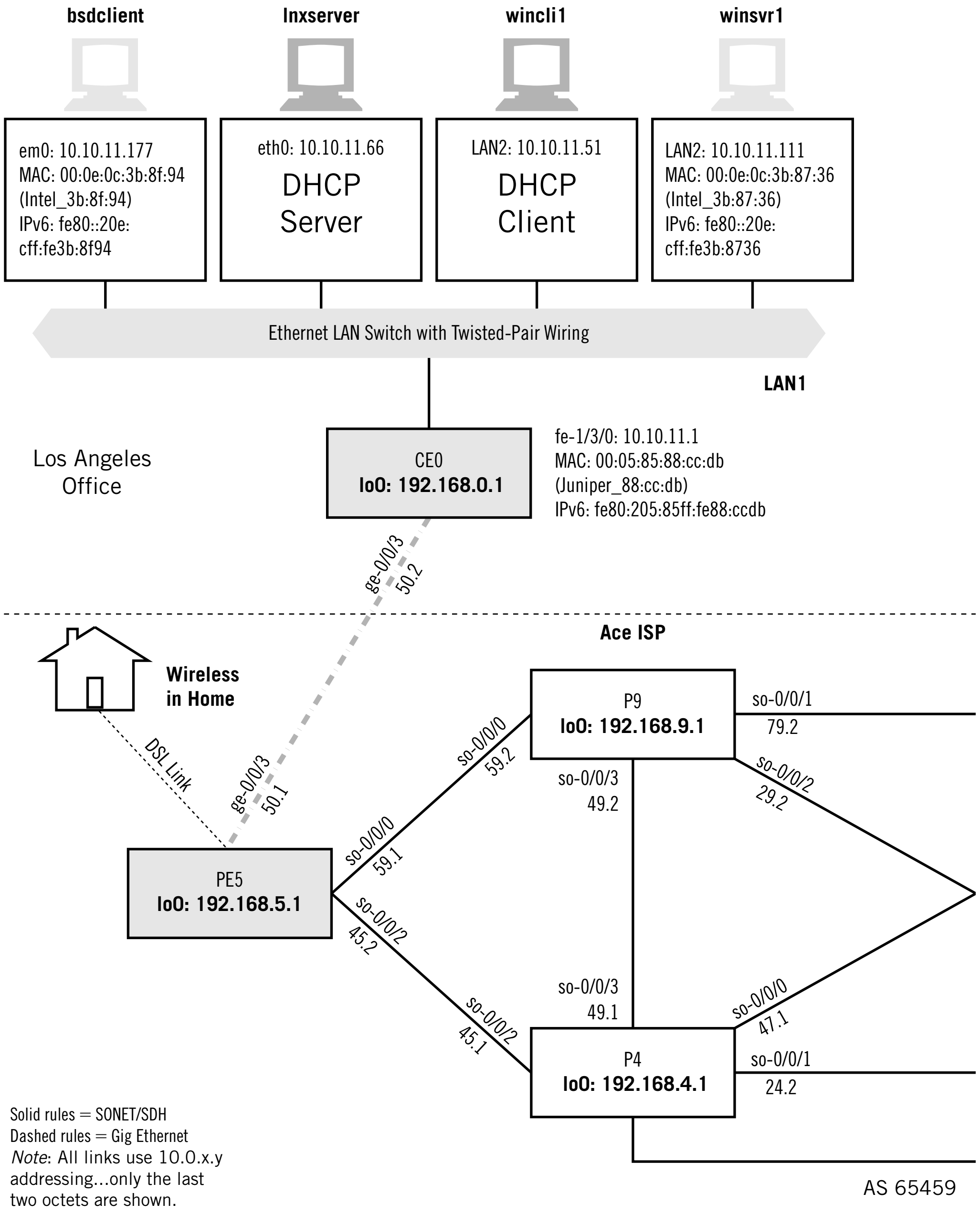
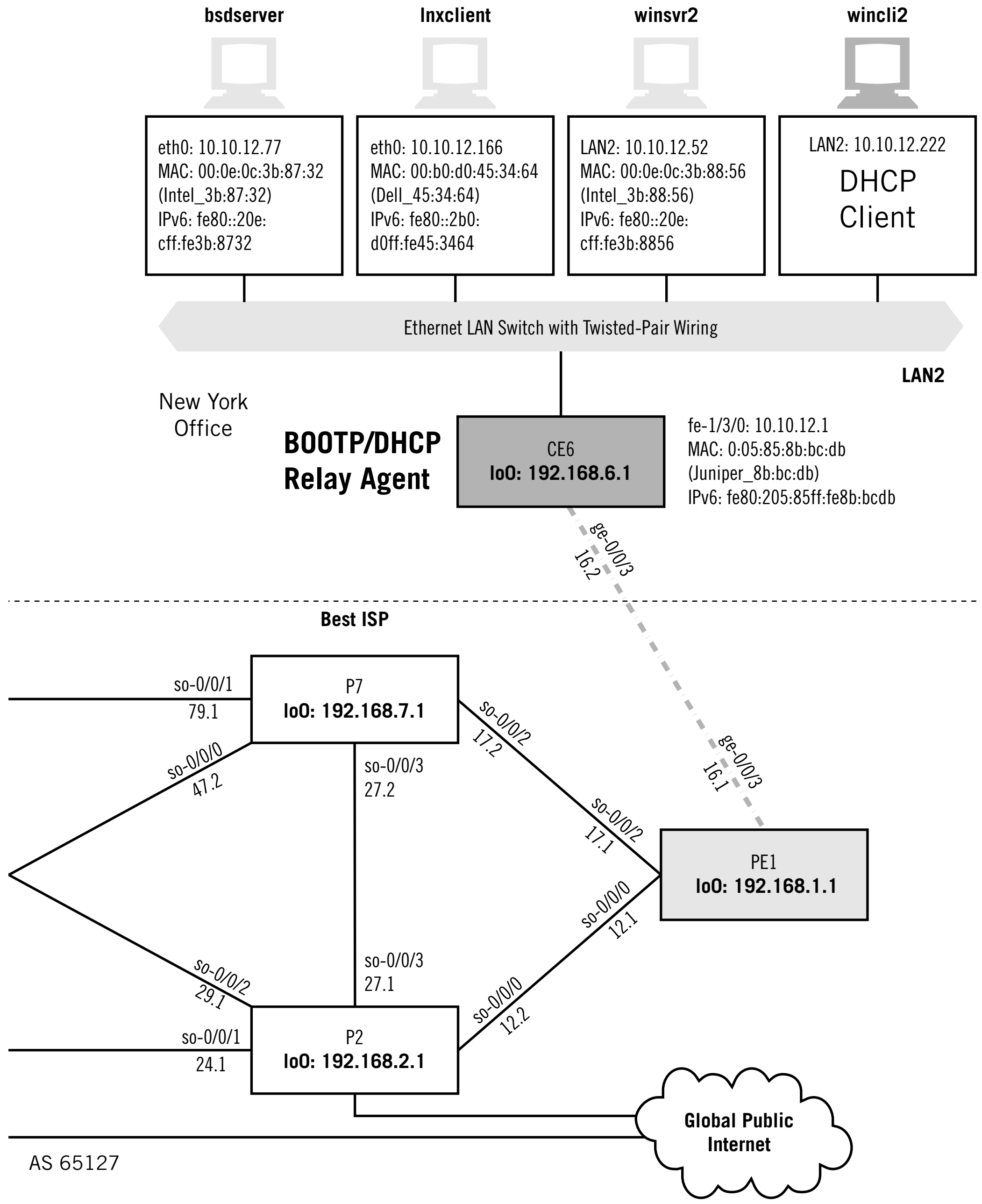


FIGURE 18.1

DHCP devices and configuration on the Illustrated Network showing the host used as DHCP relay agent.


bsdserver
lnxclient
winsvr2
wincli2
eth0: 10.10.12.77
MAC: 00:0e:0c:3b:87:32
(Intel_3b:87:32)
IPv6: fe80::20e:cff:fe3b:8732
eth0: 10.10.12.166
MAC: 00:b0:d0:45:34:64
(Dell_45:34:64)
IPv6: fe80::2b0:d0ff:fe45:3464
LAN2: 10.10.12.52
MAC: 00:0e:0c:3b:88:56
(Intel_3b:88:56)
IPv6: fe80::20e:cff:fe3b:8856
LAN2: 10.10.12.222
DHCP Client
Ethernet LAN Switch with Twisted-Pair Wiring
LAN2
New York Office
BOOTP/DHCP Relay Agent
CE6
lo0: 192.168.6.1
fe-1/3/0: 10.10.12.1
MAC: 0:05:85:8b:bc:db
(Juniper_8b:bc:db)
IPv6: fe80:205:85ff:fe8b:bcdb
ge-0/0/3
16.2
Best ISP
so-0/0/1
79.1
P7
lo0: 192.168.7.1
so-0/0/2
17.2
ge-0/0/3
16.1
so-0/0/0
47.2
so-0/0/3
27.2
so-0/0/2
17.1
PE1
lo0: 192.168.1.1
so-0/0/0
12.1
so-0/0/2
29.1
so-0/0/3
27.1
so-0/0/0
12.2
so-0/0/1
24.1
P2
lo0: 192.168.2.1
Global Public Internet
AS 65127

So, BOOTP was updated and clarified in RFC 2131 to become DHCP, which automated the IP address assignment process, making the entire system more friendly and useful for host configuration. RFC 2132 described all parameters that could be used with BOOTP and DHCP. The real value offered by DHCP over BOOTP was the ability to release an address. Dynamically assigned BOOTP devices received an address that had no upper bound on how long they could use it.

DHCP AND ADDRESSING

So far, we've used static address assignment on all of the hosts on the Illustrated Network. This is a common enough practice: Lab network testing is often hard enough without worrying about address leases expiring, host addresses changing, and cluttering up the LAN with DHCP chatter. But the point here is to dynamically assign the host addresses on the Illustrated Network (we'll leave the routers alone), so that's what we'll do for this chapter. We'll use the equipment as configured in Figure 18.1. Note that for these application-level chapters we can go back to two ISPs and routing domains.

We'll use IPv4 only and set up our Linux server (`lnxserver`) as a DHCP server for the IP address ranges on both LAN1 and LAN2. First, we'll configure Windows XP on the same LAN to find its address using the DHCP server. Naturally, as with multicast this won't help the hosts on LAN2 find the DHCP server. So, we'll configure LAN2 router CE6 as a BOOTP and DHCP relay agent by sending DHCP messages to the Linux DHCP server and sending back the replies. Finally, we'll configure the Windows XP client on LAN2 to use dynamic IP address assignment and to make sure the entire configuration works.

Once again, it must be pointed out that this network exists solely for this book. In a real situation, no one would really make clients in Los Angeles rely on a DHCP server across the country (although it would certainly work). Considering the amount of information that would be exposed, it would at least be carried over some sort of encrypted path.

DHCP Server Configuration

Linux-based DHCP servers run `/usr/sbin/dhcpd`, the DHCP daemon, using parameters found in the `/etc/dhcpd.conf` file. The configuration guide bundled with the most common DHCP implementation, from the Internet Software Consortium (ISC), is 36 pages long and gives all sorts of options that are not needed for basic configurations.

There are even freeware implementations of DHCP servers for Windows XP. These feature the expected point-and-click GUI setup interface, and are just as useful as their Unix-based cousins.

The following is a fairly minimal configuration file for a DHCP server. Note that we can assign the default router address as an option for the subnet. If this option is not present, users will have to enter their default "gateway" information manually.

```
[root@lnxserver admin]# /cat  /etc/dhcpd.conf
# dhcpd.conf
#
# global options
ddns-update-style interim;
default-lease-time 600;
max-lease-time 7200;

subnet 10.10.11.0 netmask 255.255.255.0 {
   range 10.10.11.200 10.10.11.210;
   option routers 10.10.11.1;
}
subnet 10.10.12.0 netmask 255.255.255.0 {
   range 10.10.12.210 10.10.12.220;
   option routers 10.10.12.1;
}
```

Although we are not using DHCP to dynamically update DNS entries, and we don't even have a DNS server on the LAN yet, the ISC implementation insists on having a line in the configuration referencing dynamic DNS update "style." And although a lot of TCP/IP references mention DHCP's "unhelpful" error messages, we found the error messages when we tried to start dhcpd with a missing semicolon (;) or a missing ddns-update-style line to be `explicit` and `welcome`.

By the way, this lack of DNS is one reason many hands-on Internet services workshops start with DNS first. But there is no requirement for this, as the order of the chapters in this book illustrates.

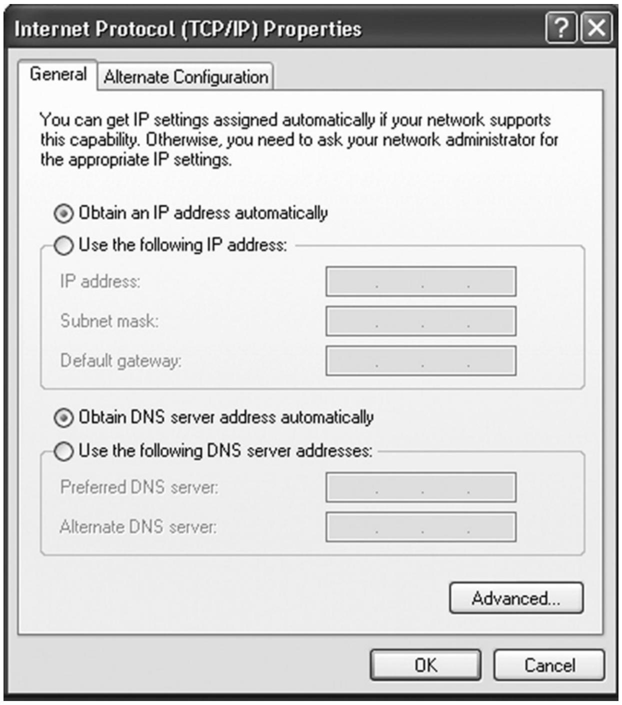
But what DNS name should be associated with a DHCP address? Typically, a generic name such as *dhcp1.example.com* is associated with the DHCP address. However, this is not appropriate for servers, and only barely tolerable for clients, which usually have more informative names in DNS. And generally, you don't want to hand out changing IP addresses to routers, servers, or the DHCP server itself.

Ordinarily, we would include an option line for the DNS server's names, but we haven't configured those yet on the network. Options can be global or applied to only a subset of the network, a nice feature. We'd also usually have a host entry for our servers so that they would get the same IPv4 addresses every time. For testing, it's common to override the default lease time and maximum lease time (which are fairly high) for which a host can ask to use the address. We've made them 10 minutes and an hour, respectively, here.

The most important lines are those that establish the address pool for hosts on LAN1 (`10.10.11.0`) that ask for an IPv4 address. This information is set in the subnet and range lines. We've made the range different from any of the IPv4 addresses used before, just so it's easy to see if Windows XP is really picking up the DHCP address.

We've also set up an address pool for LAN2 (`10.10.12.0`), just to save time. We haven't configured the LAN2 router as a DHCP relay agent yet, but we will.

Setting up a DHCP client is much easier than setting up the server. Windows XP, for example, makes it very easy to reconfigure a PC to obtain an IPv4 address (including the default router) from the network's DHCP server (as shown in Figure 18.2).


Internet Protocol (TCP/IP) Properties

General | Alternate Configuration

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

[✓ Obtain an IP address automatically]
Use the following IP address:
IP address: . . .
Subnet mask: . . .
Default gateway: . . .

[✓ Obtain DNS server address automatically]
Use the following DNS server addresses:
Preferred DNS server: . . .
Alternate DNS server: . . .

FIGURE 18.2

Configuring Windows to use DHCP, as is commonly done. Note that the IP address and DNS server to be used are assigned.

Now let's run the DHCP server on `lnxserver` and see what address the Windows XP host `wincli1` is assigned.

```
C:\Documents and Settings\Owner>ipconfig
Windows IP Configuration
Ethernet adapter Local Area Connection:

     Connection-specific DNS Suffix . :
     IP Address . . . . . . . . : 10.10.11.200
     Subnet Mask . . . . . . . . : 255.255.255.0
     Default Gateway . . . . . . : 10.10.11.1
```

As expected, the address assigned is within the range specified, and is the first address in that range.

Router Relay Agent Configuration

The configuration stanza to make a Juniper Network router a DHCP relay agent is under the BOOTP hierarchy level. This makes sense because DHCP relay agents are all BOOTP relay agents as well. We'll talk more about BOOTP later in this chapter.

The router can act as a relay agent globally or for a group of interfaces. This just makes the CE6 router into a DHCP relay agent for the LAN2 interface. There is no need to do anything for LAN1 on the network because the DHCP server handles all of those hosts locally.

```
set forwarding-options helpers bootp description "DHCP relay agent for
   lnxserver on LAN1";
set forwarding-options helpers bootp server 10.10.11.66;
set forwarding-options helpers bootp interface fe-1/3/0;
```

That's all there is to it. As long as there's a way to reach network `10.10.11/24` from LAN2 and a way to get back to `10.10.12/24` from CE0, DHCP messages should have no problem crossing the network like any other packets.

Getting Addresses on LAN2

Without a relay agent running on the LAN2 router, we can fire up `wincli2` all we want and it will never receive an IP address from a DHCP server. One is not present on LAN2, and the router will not route DHCP messages unless told to.

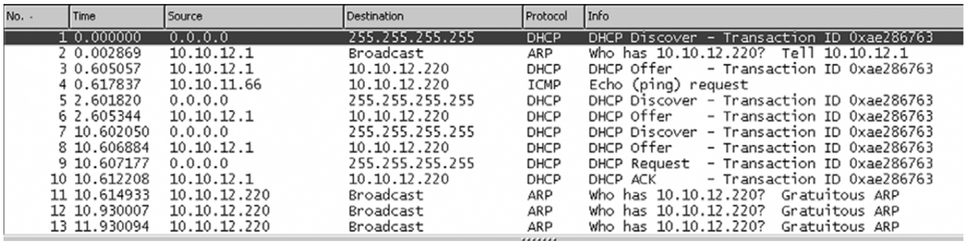
Now that we have the relay agent running, we can check the IPv4 address on `wincli2`. Note that the lowest IP address in the range is not always the first one handed out by the DHCP server. In this case, the host asks for its "old" address of `10.10.12.222`, and the server attempts to assign the closest address it has to that one.

```
C:\Documents and Settings\Owner>ipconfig
Windows IP Configuration
Ethernet adapter Local Area Connection:

      Connection-specific DNS Suffix . :
      IP Address . . . . . . . . : 10.10.12.220
      Subnet Mask . . . . . . . . : 255.255.255.0
      Default Gateway . . . . . . : 10.10.12.1
```

DHCP is such an important part of LANs and the Internet today that a closer look at the functioning of DHCP through a router relay agent is a good idea. The complete sequence of events, captured on `wincli2` as it received its DHCP address, is shown in Figure 18.3.

We'll talk about DHCP messages and sequences in detail later in this chapter. Note that the sequence starts with `wincli2` sending a broadcast DHCP discover message onto LAN2 with the "unknown" source address of `0.0.0.0`. The host asks for its "old" address, `10.10.12.222`. The router, acting as relay agent, forwards the request to the DHCP server (`10.10.11.66`, `lnxserver`) on LAN1, which replies to the relay agent and wants to assign address `10.10.12.220` to `wincli2`. The relay agent sends an ARP (No. 2) to see if anyone on LAN2 already has `10.10.12.220` (it could have been assigned statically). The relay agent then offers the host this IP address (No. 3), and the DHCP server itself (No. 4) sends a ping to check on `10.10.12.220` itself (note that there is no reply to the ping from `wincli2`).



```
No. -  Time        Source        Destination        Protocol  Info
    1  0.000000    0.0.0.0       255.255.255.255    DHCP      DHCP Discover - Transaction ID 0xae286763
    2  0.002869    10.10.12.1    Broadcast          ARP       Who has 10.10.12.220?  Tell 10.10.12.1
    3  0.605057    10.10.12.1    10.10.12.220       DHCP      DHCP Offer    - Transaction ID 0xae286763
    4  0.617837    10.10.11.66   10.10.12.220       ICMP      Echo (ping) request
    5  2.601820    0.0.0.0       255.255.255.255    DHCP      DHCP Discover - Transaction ID 0xae286763
    6  2.605344    10.10.12.1    10.10.12.220       DHCP      DHCP Offer    - Transaction ID 0xae286763
    7  10.602050   0.0.0.0       255.255.255.255    DHCP      DHCP Discover - Transaction ID 0xae286763
    8  10.606884   10.10.12.1    10.10.12.220       DHCP      DHCP Offer    - Transaction ID 0xae286763
    9  10.607177   0.0.0.0       255.255.255.255    DHCP      DHCP Request  - Transaction ID 0xae286763
   10  10.612208   10.10.12.1    10.10.12.220       DHCP      DHCP ACK      - Transaction ID 0xae286763
   11  10.614933   10.10.12.220  Broadcast          ARP       Who has 10.10.12.220?  Gratuitous ARP
   12  10.930007   10.10.12.220  Broadcast          ARP       Who has 10.10.12.220?  Gratuitous ARP
   13  11.930094   10.10.12.220  Broadcast          ARP       Who has 10.10.12.220?  Gratuitous ARP
```

FIGURE 18.3

DHCP messages sent through a router relay agent. Note the use of broadcast and the "unknown" source IP address.

It takes a while for the host to gather the information about possible multiple DHCP servers, and there are two pairs of repeated DHCP discover messages from "`0.0.0.0`" and DHCP offers from the relay agent (Nos. 5–8). In each exchange, the host asks for its old IP address (`10.10.12.222`) in the DHCP discover message, and the relay agent assigns `10.10.12.220` in the DHCP offer message.

Finally, wincli2 accepts the DHCP information and assigned address, and sends a DHCP request message (No. 9) for configuration information for `10.10.12.220`, but it is still using the `0.0.0.0` address. The relay agent replies with a DHCP acknowledgement (No. 10), which basically contains the same information as before.

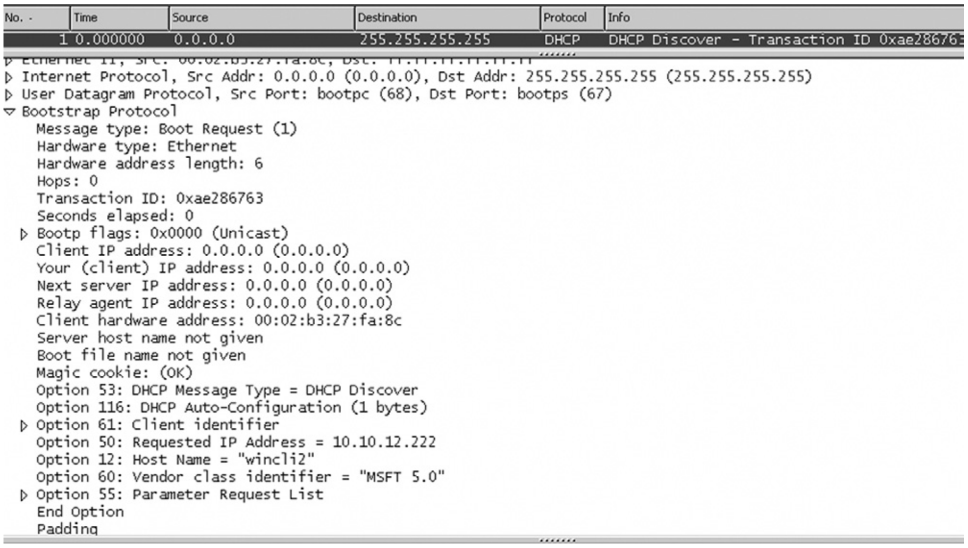
The sequence ends with a series of gratuitous ARPs to the relay agent (Nos. 11–13) for address `10.10.12.220`, the host's new address (see the source IP address field). This tells the DHCP relay agent that everything has worked out. The details of one of the DHCP discover messages sent by the host (all of them are essentially the same) are shown in Figure 18.4.

The details of one of the DHCP offer messages sent by the relay agent on behalf of the DHCP server (all of these are essentially the same too) are shown in Figure 18.5.

Using DHCP on a Network

As we have seen, what DHCP brings to TCP/IP for the first time is a measure of mobility. With the proper DHCP servers available, a user could unplug a host from one Ethernet LAN subnet, move it across the country, plug it into another subnet, expect the configuration data to be loaded properly, and become productive on the new subnet immediately.

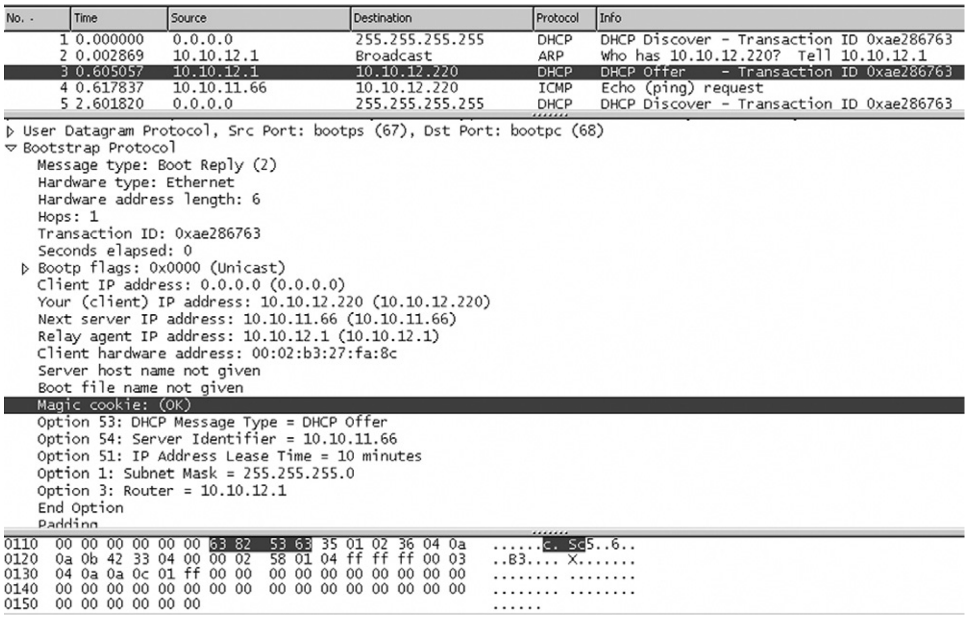
Once ISPs began offering dial-up Internet access to the general public with home PCs, the benefits of DHCP became instantly obvious. Suppose an ISP had a pool of 254 IPv4 addresses, that is, what used to be a Class C address. But the ISP also has 300 customers. Obviously, 254 IP addresses cannot be statically assigned to 300 hosts. However, all of them cannot be on-line at the same time because the ISP has only 200 dial-in modem ports (a situation that was not uncommon before the Web took over the planet). So, DHCP quickly became the means of choice in assigning IP addresses dynamically to a pool of users.



```
No. -    Time        Source         Destination       Protocol  Info
      1 0.000000     0.0.0.0        255.255.255.255   DHCP      DHCP Discover - Transaction ID 0xae28676
▷ Ethernet II, Src: 00:02:b3:27:fa:8c, Dst: ff:ff:ff:ff:ff:ff
▷ Internet Protocol, Src Addr: 0.0.0.0 (0.0.0.0), Dst Addr: 255.255.255.255 (255.255.255.255)
▷ User Datagram Protocol, Src Port: bootpc (68), Dst Port: bootps (67)
▽ Bootstrap Protocol
    Message type: Boot Request (1)
    Hardware type: Ethernet
    Hardware address length: 6
    Hops: 0
    Transaction ID: 0xae286763
    Seconds elapsed: 0
  ▷ Bootp flags: 0x0000 (Unicast)
    Client IP address: 0.0.0.0 (0.0.0.0)
    Your (client) IP address: 0.0.0.0 (0.0.0.0)
    Next server IP address: 0.0.0.0 (0.0.0.0)
    Relay agent IP address: 0.0.0.0 (0.0.0.0)
    Client hardware address: 00:02:b3:27:fa:8c
    Server host name not given
    Boot file name not given
    Magic cookie: (OK)
    Option 53: DHCP Message Type = DHCP Discover
    Option 116: DHCP Auto-Configuration (1 bytes)
  ▷ Option 61: Client identifier
    Option 50: Requested IP Address = 10.10.12.222
    Option 12: Host Name = "wincli2"
    Option 60: Vendor class identifier = "MSFT 5.0"
  ▷ Option 55: Parameter Request List
    End Option
    Padding
```

FIGURE 18.4

The DHCP discover message details. Note the use of the bootstrap protocol (BOOTP) and the numerous options.



```
No. -    Time        Source         Destination       Protocol  Info
      1 0.000000     0.0.0.0        255.255.255.255   DHCP      DHCP Discover - Transaction ID 0xae286763
      2 0.002869     10.10.12.1     Broadcast         ARP       Who has 10.10.12.220?  Tell 10.10.12.1
      3 0.605057     10.10.12.1     10.10.12.220      DHCP      DHCP Offer    - Transaction ID 0xae286763
      4 0.617837     10.10.11.66    10.10.12.220      ICMP      Echo (ping) request
      5 2.601820     0.0.0.0        255.255.255.255   DHCP      DHCP Discover - Transaction ID 0xae286763
▷ User Datagram Protocol, Src Port: bootps (67), Dst Port: bootpc (68)
▽ Bootstrap Protocol
    Message type: Boot Reply (2)
    Hardware type: Ethernet
    Hardware address length: 6
    Hops: 1
    Transaction ID: 0xae286763
    Seconds elapsed: 0
  ▷ Bootp flags: 0x0000 (Unicast)
    Client IP address: 0.0.0.0 (0.0.0.0)
    Your (client) IP address: 10.10.12.220 (10.10.12.220)
    Next server IP address: 10.10.11.66 (10.10.11.66)
    Relay agent IP address: 10.10.12.1 (10.10.12.1)
    Client hardware address: 00:02:b3:27:fa:8c
    Server host name not given
    Boot file name not given
    Magic cookie: (OK)
    Option 53: DHCP Message Type = DHCP Offer
    Option 54: Server Identifier = 10.10.11.66
    Option 51: IP Address Lease Time = 10 minutes
    Option 1: Subnet Mask = 255.255.255.0
    Option 3: Router = 10.10.12.1
    End Option
    Padding
0110  00 00 00 00 00 00 63 82  53 63 35 01 02 36 04 0a   ......c. Sc5..6..
0120  0a 0b 42 33 04 00 00 02  58 01 04 ff ff ff 00 03   ..B3.... X.......
0130  04 0a 0a 0c 01 ff 00 00  00 00 00 00 00 00 00 00   ........ ........
0140  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0150  00 00 00 00 00 00                                  ......
```

FIGURE 18.5

The DHCP offer message details, showing the use of the “magic cookie.”

Organizations that employed proxy servers to protect their Internet users (or limit Internet users) could do the same thing, and often did. In fact, any time the pool of potential users exceeds the number of IP addresses available, DHCP is a potential solution.

The heavy use of changing IP addresses among ISPs was one major reason ISPs refused to support servers on the customer's premises (asymmetric traffic loads, especially over always-on but asymmetrical DSL links, was the other one). Servers were typically included in DNS, to make them easy to remember, and this required a high degree of stability of IP addresses because changes had to propagate literally around the world. Naturally, dynamic server addresses, changing rapidly, challenged DNS procedures and capabilities. Servers could get static IP addresses, if they could be found, and running one server process like a Web server on an otherwise all-client host made the box into a server. The simplest thing for an ISP to do was to ban servers on the customer's premises, unless extra fees for DNS "maintenance" were paid (in truth, there was little maintenance the ISP had to do except initially). Officially, home servers were "not supported"; since ISPs had little way of making sure that a server was present this essentially meant, "If you call and try to open a trouble ticket on it, we won't listen."

When DHCP is configured on a client in many operating systems, it usually isn't even required to name it. Just check off or click on "obtain an IP address automatically" and you're in business.

BOOTP still exists, and some devices still use BOOTP alone. BOOTP is often combined with the Trivial File Transfer Protocol (TFTP), defined in RFC 1350 (RFCs 2347, 2348, and 2349 all discuss TFTP options). And the best way to understand why DHCP works the way it does is to begin with BOOTP.

BOOTP

Diskless workstations were expected to have only basic IP, UDP, and TFTP capabilities at start-up, although of course they needed Ethernet and rudimentary operating system functions as well. The original vision for BOOTP was to have the process complete in three steps.

The BOOTP client broadcast a request for information from port UDP 68 to a boot server listening on port 67. (BOOTP uses well-known ports for both client and server because server replies can be broadcast, but typically are not.)

The boot server returned the client's IP address and, as an option, the location of a file to be downloaded (presumably, the rest of the client's software was in this file). The client used TFTP and the boot server listening on UDP port 69 to download the software.

RARP, discussed in Chapter 5, provides the IP address that goes with a physical address (such as the MAC address). RARP provides an IP address to a diskless client, but *only* an IP address. And RARP broadcasts never pass through a router, whereas BOOTP requests, in proper configurations, will (this requires a relay agent, as in DHCP).

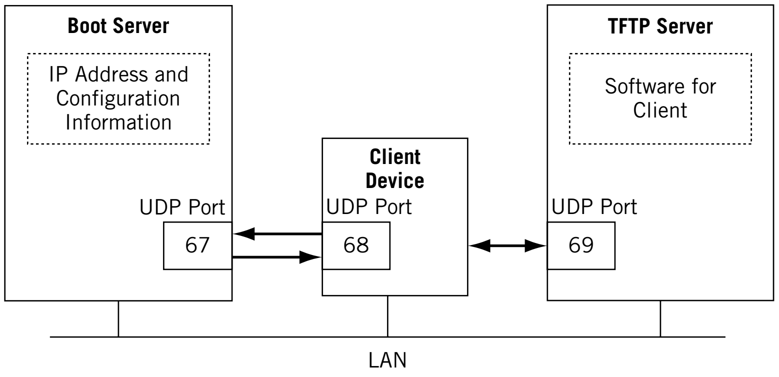


FIGURE 18.6

BOOTP and TFTP servers, showing the ports used by the servers and client.

BOOTP Implementation

Diskless workstations never became a popular line, and most users saw them as a return to the "bad old days" of "dumb terminals" and considered a full-blooded PC on the desktop as a sign of status. And soon enough the cost differential for diskless devices as opposed to full-fledged workstations or desktops shrunk to zero and then went negative. Applications for devices with no local storage still exist, but there is no cost benefit associated with them.

Once almost all PCs began to ship with minimal hard disks it became more common to split the boot server functions between two separate servers. The boot server still listened on UDP port 67 for client broadcast requests sent on port 68, and this was usually all PCs needed. But for truly diskless devices one or more TFTP servers provided the files needed for further operation, usually separated by type. This arrangement is shown in Figure 18.6.

BOOTP was very flexible. Clients could start with some or no information, accept any boot server or pick a particular one, and use no file (a default) or a specific download file.

BOOTP Messages

All BOOTP requests and replies are sent as 300-byte UDP messages. These are shown in Figure 18.7. Fields shown in bold must be filled in for a BOOTP request, and those in italic represent optional information supplied by the client.

Opcode—This byte is set to 1 for a request and 2 for a reply.

Hardware Type—This byte is set to 1 for Ethernet, and uses the same values as the hardware type field in an ARP message.

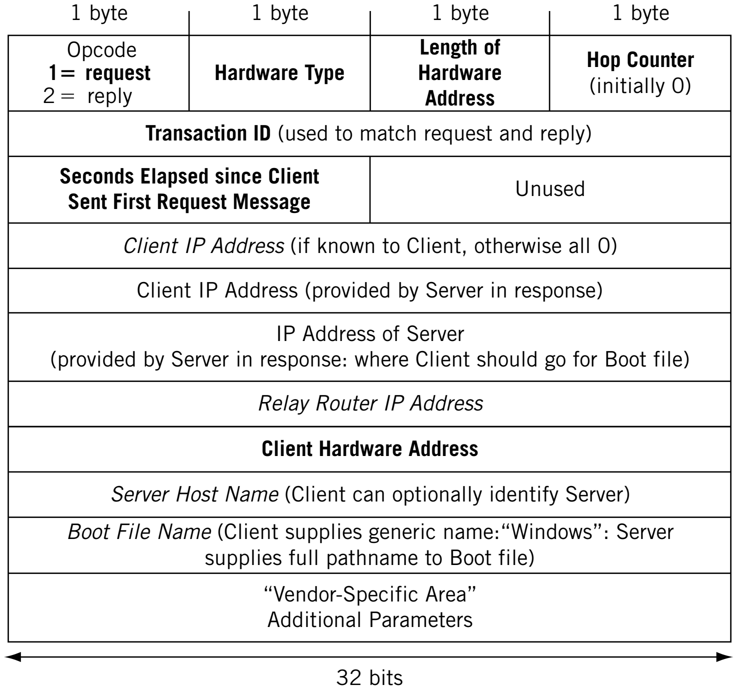

1 byte
1 byte
1 byte
1 byte
Opcode
1 = request
2 = reply
Hardware Type
Length of Hardware Address
Hop Counter (initially 0)
Transaction ID (used to match request and reply)
Seconds Elapsed since Client Sent First Request Message
Unused
Client IP Address (if known to Client, otherwise all 0)
Client IP Address (provided by Server in response)
IP Address of Server
(provided by Server in response: where Client should go for Boot file)
Relay Router IP Address
Client Hardware Address
Server Host Name (Client can optionally identify Server)
Boot File Name (Client supplies generic name: "Windows": Server supplies full pathname to Boot file)
"Vendor-Specific Area"
Additional Parameters
32 bits

FIGURE 18.7

Request.

Hardware Address Length—This byte is set to 6 for Ethernet.

Hop Counter—The client sets this to 0, but a proxy BOOTP server (or *relay agent*, described later) can use this field when the BOOTP message is sent beyond the local Ethernet.

Transaction ID—A random 4-byte number chosen by the client and used to match replies to their requests. Multiple servers can reply, and only the first is accepted by the client.

Seconds Elapsed—A 2-byte field set by the client to the amount of time since the bootstrap process began. It starts at 0 and gradually increases if the request is not answered. A secondary server can monitor this value, and if it gets too high will assume the primary BOOTP server is down and reply to the client.

Client IP Address—Set to all 0 bits unless the client knows its IP address, in which case it is placed here.

"Your" Client IP Address—If the previous field is 0, the server supplies the client's IP address in this field.

Server IP Address—Filled in by the server.

Relay Router IP Address—If a BOOTP relay agent is used, the router fills in the address of the port the request was received on. This allows the server to reply to the proper relay agent.

Client Hardware Address—The same 16-byte address is in the frame source address, but the BOOTP process has no easy access to this information (which is three layers away) so it is placed here.

Server Hostname—The server optionally can use these 64 bytes (null terminated) to identify itself to the client.

Boot File Name—The server optionally can use these 128 bytes (null terminated) to identify the path to and the name of the boot file.

Vendor-Specific Area—These 64 bytes are used for BOOTP extensions, defined in RFC 1533.

BOOTP Relay Agents

BOOTP requests are broadcast, and broadcasts will not be forwarded through a router. Yet maintaining BOOTP servers on all subnets, which are often quite small, can be burdensome in many organizations. So, BOOTP allows the use of relay agents, which can be hosts but are usually routers having the added capability to forward BOOTP requests to a centrally located server.

The router BOOTP relay agent is allowed to broadcast the request onto other subnets, using the hop count to control endless looping, but it is more common for the relay agent to maintain a list of the IP addresses of one or more boot servers to which to forward the requests. The way it all fits together is shown in Figure 18.8.

The relay agent receiving a BOOTP broadcast checks the Relay Router field. If it is set to 0, the relay agent inserts the port's IP address (if the field is non-zero, another relay router has already processed this request). The BOOTP server will use the address to reply to the proper relay agent.

The relay agent can send the request to one or more preconfigured BOOTP servers. The relay agent usually replaces the broadcast IP address with the BOOTP server's destination address.

BOOTP "Vendor-Specific Area" Options

The fields in the BOOTP request and reply do not cover a lot of things client hosts often need to know to function properly. For example, how is the subnet mask and default router address conveyed to the client?

RFC 1533 kept the vendor-specific purpose of the field but added several optional functions that can be used to supply needed information to a client. The "magic cookie"

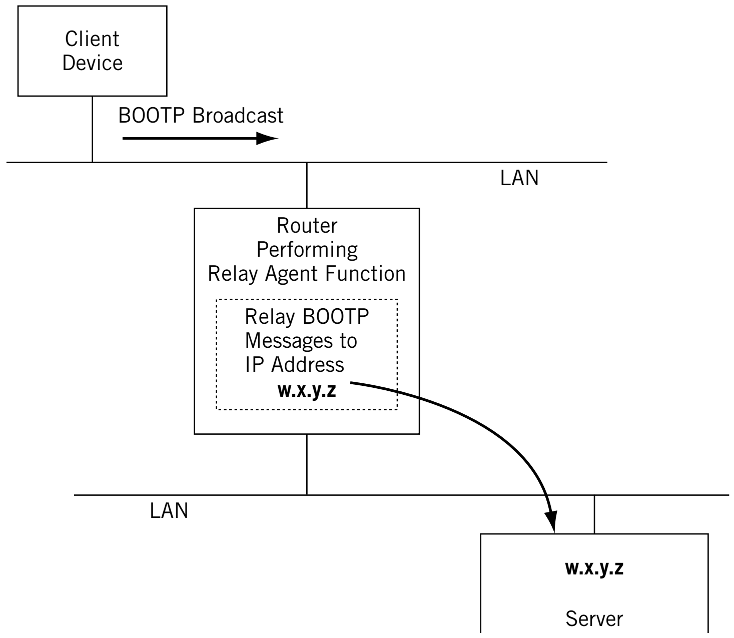

Client Device
BOOTP Broadcast
LAN
Router Performing Relay Agent Function
Relay BOOTP Messages to IP Address w.x.y.z
LAN
w.x.y.z
Server

FIGURE 18.8

BOOTP relay agent (router), showing how the relay agent forwards broadcast BOOTP messages to a unicast IP address.

IPv4 address of 99.130.83.99 is used to signal clients that there is useful information in this area.

Each item begins with a 1-byte Tag (for example, Tag = 1 is for the subnet mask) and Length (subnet mask = 4 bytes) field. Tag = 0 is used to pad items to a 32-bit boundary, and Tag = 255 is used pad out the end of the list.

Once a client has used BOOTP to obtain an IP address, subnet mask, and default router address, it is ready to begin the software download phase if needed. The TFTP protocol is used for this process.

TRIVIAL FILE TRANSFER PROTOCOL

Many books discuss TFTP in the context of full FTP. But TFTP is best understood in the context of the BOOTP environment. In particular, TFTP differs greatly from usual FTP operation (FTP is discussed in Chapter 20). In contrast to full FTP, TFTP

- Uses UDP port 69
- Uses uniformly sized 512-byte blocks of data, except for the last (If the file is a multiple of 512 bytes, a final, empty block signals end-of-file.)

- Numbers blocks starting from 1
- Acknowledges every block
- Uses no authentication

Today, of course, the lack of authentication means that use of TFTP requires special considerations. And it still makes more sense to use Trivial File Transfer Protocol for BOOTP software downloads because in many cases the client and server are on the same low-error-rate LAN.

Once a client knows where to go and what to get, a TFTP transaction starts with a read request (RRQ) to download a file or write request (WRQ), used if the client is going to save information back onto the TFTP server. The requests are sent to UDP port 69 on the server, and a dynamic port is used on the client.

The server does not use port 69 throughout the process, but identifies a server port to use for the rest of the procedure. Data transfer proceeds through an exchange of sequenced data blocks and answering ACKs, one-for-one, echoing the data block number. Any non–full-data block ends the exchange.

The default block size can be changed using the options at the end of the read or write request. A size of 1468 (a 1500-byte Ethernet frame minus the 20 IP, 8 UDP, and 4 TFTP header bytes) is common. Other options include a resend timeout value (UDP has none of its own) and the total size of the file to be transferred. This value is offered in the client write request, but is set to 0 in a read request and sent by the server in response. A client is allowed to abort the transfer if the file size the server wants to transfer is too large.

TFTP Messages

TFTP really only has requests (RQ), data blocks (DATA), and ACKs, but these are employed to yield a total of six message types.

- Read request (RRQ)
- Write request (WRQ)
- Data block (DATA)
- Acknowledgment (ACK)
- Error (ACK)
- Option acknowledgment (OACK)

The six operation codes are used in the Trivial File Transfer Protocol header, shown in Figure 18.9.

The fields in RRQ and WRQ can vary in size and are thus delimited with all-0 bytes. Oddly, there are no codes for the modes or for the strings `netascii` and `octet` (there was also a `mail` mode initially).

TFTP Download

TFTP lives up to its name. A simple TFTP transfer is shown in Figure 18.10. In the figure, it is assumed that no options are used.

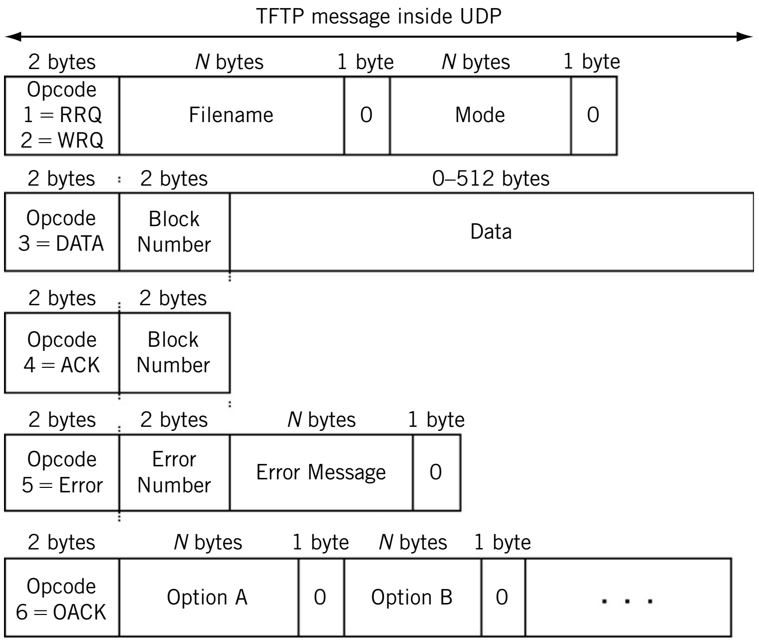


FIGURE 18.9

The six TFTP messages. Note that the content is extremely variable depending on opcode.

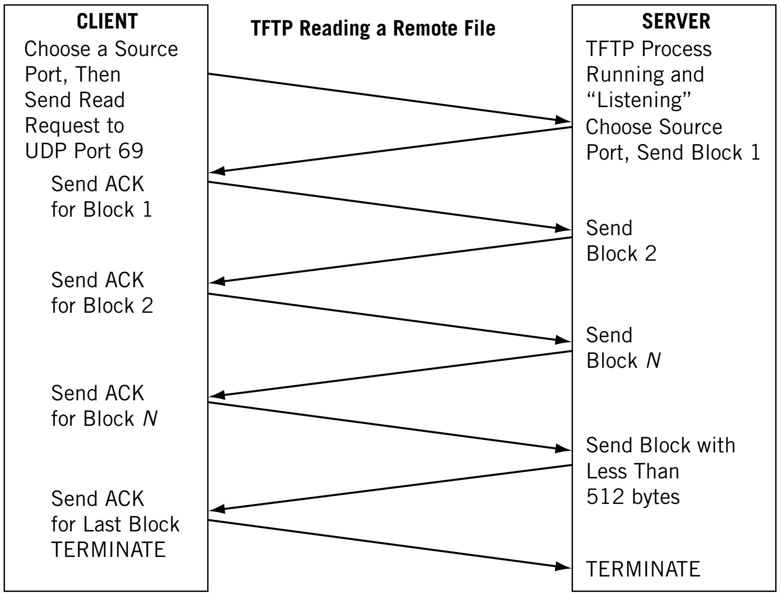

CLIENT
Choose a Source Port, Then Send Read Request to UDP Port 69
Send ACK for Block 1
Send ACK for Block 2
Send ACK for Block N
Send ACK for Last Block TERMINATE
TFTP Reading a Remote File
SERVER
TFTP Process Running and "Listening" Choose Source Port, Send Block 1
Send Block 2
Send Block N
Send Block with Less Than 512 bytes
TERMINATE

FIGURE 18.10

TFTP file transfer. Compared to full FTP, this exchange is very simple.

DHCP

It might seem odd to spend so much time in a chapter on DHCP discussing BOOTP and TFTP. But much of what DHCP does and the way it accomplishes its functions is similar to the operation of these two earlier protocols. DHCP involves a more complex exchange of messages between client and server, but the intention was always that servers could provide both BOOTP and DHCP functions with a minimum of recoding.

DHCP was referenced in BOOTP RFCs 1533 and 1534, but as an "extension" of BOOTP capabilities. Currently, RFC 2131 describes DHCP and distinguishes it from BOOTP. Not only does a DHCP server allocate addresses to clients, but it also maintains parameters for individual clients and entire client groups, greatly enhancing the efficiency of the entire system. In general, DHCP is designed to:

- Be a mechanism. No "policy" or ideas about IP address allocation schemes are assumed by DHCP. However, DHCP can be the mechanism on which such policies are built.
- Do away with manual configuration. A user should always be able to simply plug their devices into the network and work. (The requirement to configure DHCP, if not the default, is beyond DHCP's control.)
- Handle many subnets from one server. DHCP employs the BOOTP relay agent concept, mostly implemented in routers, for this purpose.
- Allow multiple servers. For redundancy and reliability, clients and servers must be able to deal with more than one DHCP server.
- Coexist with statically addressed hosts. As mentioned, dynamically addressed servers are a challenge for DNS and the user in general. DHCP must allow these hosts to function properly.
- Support BOOTP. DHCP can use BOOTP relay agents and must be able to service BOOTP clients.
- Guarantee unique addresses. No address can ever be assigned to two clients at the same time.
- Retain client information. The servers must retain all client parameters in case of failures or between shutdown and start-up.

If the addresses handed out by DHCP were permanent, there would be little difference between static assignment or the way that BOOTP operates. But the DHCP association between client and address is called a *binding*, or, more commonly, a *lease*. And like any lease, it must be renewed periodically or become available for assignment to a new client.

The pool of IP addresses handed out by the DHCP server is called a *scope*. A collection of scopes gathered for administrative purposes is known as a *superscope*.

DHCP Operation

The format of the DHCP message is shown in Figure 18.11, which should be compared to the BOOTP message in Figure 18.7. Many BOOTP clients have no problem interacting with DHCP servers, and that was the intent all along.

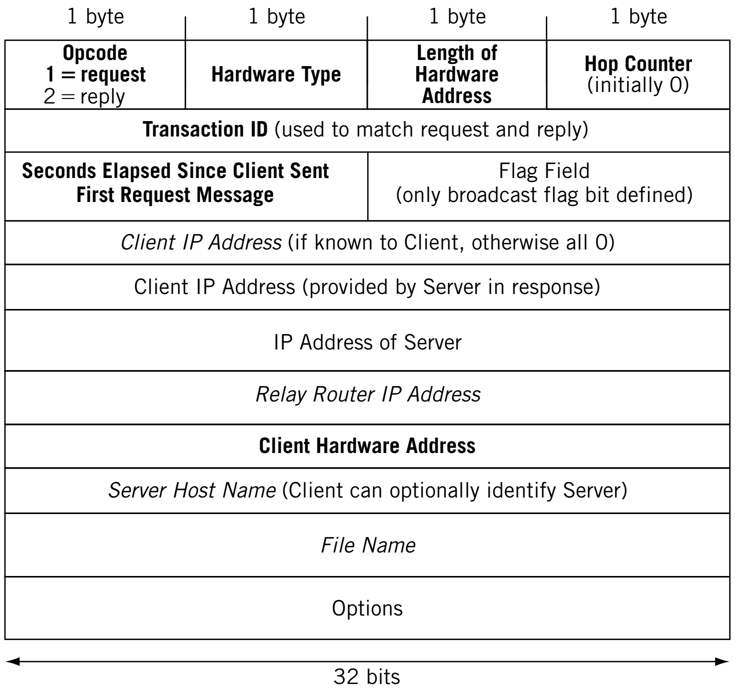

1 byte
1 byte
1 byte
1 byte
Opcode 1 = request 2 = reply
Hardware Type
Length of Hardware Address
Hop Counter (initially 0)
Transaction ID (used to match request and reply)
Seconds Elapsed Since Client Sent First Request Message
Flag Field (only broadcast flag bit defined)
Client IP Address (if known to Client, otherwise all 0)
Client IP Address (provided by Server in response)
IP Address of Server
Relay Router IP Address
Client Hardware Address
Server Host Name (Client can optionally identify Server)
File Name
Options
32 bits

FIGURE 18.11

DHCP message format, showing similarities with the BOOTP message.

The fields are the same in form and content as those for BOOTP, with a few exceptions. Opcode DHCP uses the same operation codes as BOOTP (1 = request and 2 = reply). DHCP is indicated by the use of an Option Tag value of 53. This allowed DHCP to use BOOTP relay agents transparently.

Flags—These 16 bits were unused in BOOTP. Only one flag is defined for DHCP, the rightmost bit, or BROADCAST flag. All other bits must be set to 0. A tricky issue in dynamic configuration was the fact that some clients discarded unicast packets until configuration was complete, and so the DHCP messages were rejected with their addresses! The BROADCAST bit told servers to broadcast replies to these DHCP clients.

Options—The BOOTP "vendor-specific" fields in what is now the DHCP options field, were greatly extended to become DHCP parameters. Client ID Option DHCP clients can be identified other than by hardware MAC address, as in BOOTP. Some other identifier, such as a fully qualified domain name, could be used instead. This helped if NIC cards were replaced. In practice, those cards are very reliable and this option is not used much.

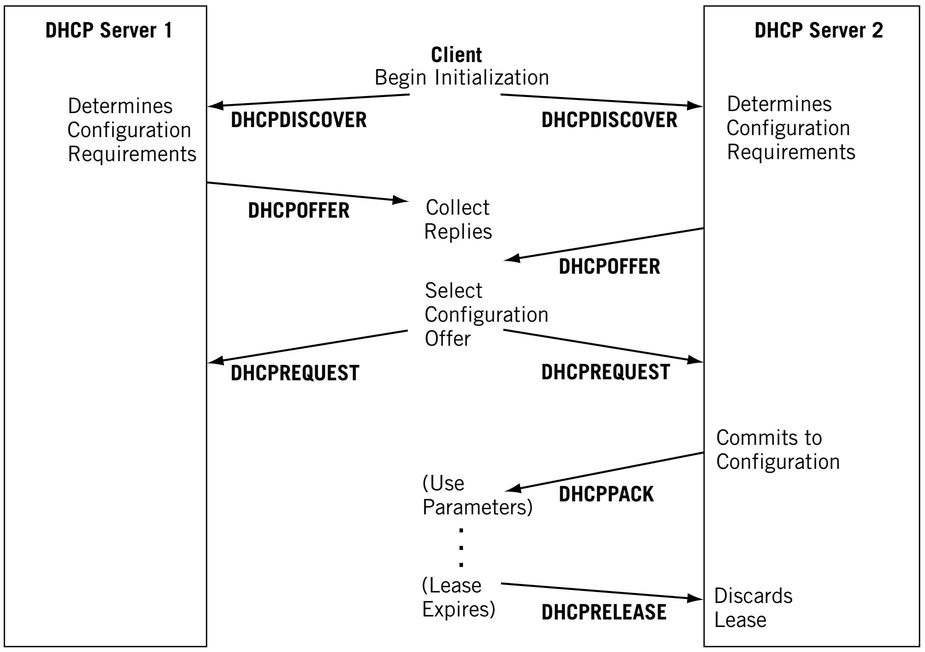


FIGURE 18.12

Typical DHCP message flow when there are two potential DHCP servers from which to choose.

The client ID option is used for several things: It provides better logging, supports dynamic DNS, and allows for hosts with more than one network interface (such as laptops with wired and wireless capability). Care must be taken that you don't produce collisions, because two hosts with the same client ID will get the same IP address.

Once a host is configured to seek out configuration information using DHCP, the message flow is straightforward—even with two "competing" DHCP servers on a LAN. The usual flow of messages is shown in Figure 18.12.

DHCP, in contrast to BOOTP, uses a complex sequence of messages between clients and servers, all tucked neatly inside the "BOOTP" options field at the end of the message. There are eight major DHCP messages types (all using either request or reply operation codes, of course).

- DHCPDISCOVER—Used by clients to discover DHCP servers, and usually includes a list of the parameters for which the client needs values, such as IP addresses, subnet mask, and default router.
- DHCPOFFER—Used by servers to offer the needed values to clients.
- DHCPREQUEST—Used by a client to request a reply from one server. The request is sent to all servers, even those not selected.
- DHCPDECLINE—Used by a client to refuse to accept one or more values from a server, usually because they are not valid for the client.

- DHCPACK—Used for server responses and to furnish the parameters to a client.
- DHCPNAK—Used by a server to refuse a client request. (Clients must start over.)
- DHCPRELEASE—Used by a client to release an IP address, returning it to the server pool.
- DHCPINFORM—Used by clients to tell servers the client has an IP address already, but needs the values for other parameters.

DHCP Message Type Options

DHCP clients can request values for more than 60 different parameters from a DHCP server. The first 49 can be used by BOOTP or DHCP, and these include the very fundamental IP subnet mask request (Tag = 1) and default router address (Tag = 3).

Options 50 through 61 are reserved for DHCP only. These are outlined in Table 18.1. Tag numbers through 127 are reserved for current and future standard options. Tags 128 through 254 are reserved for site-specific options.

Table 18.1 DHCP Parameters Shown by Tag Value

Tag	Parameter	Description
50	Requested IP address	Client asks for a specific IP address.
51	IP address lease time	Client's request or time granted by server.
52	Option overload	The Server Host Name or Boot File Name fields are carrying DHCP options to save space in the message.
53	DHCP message type	This is how the DISCOVER, OFFER, or REQUEST formats are determined.
54	DHCP server identifier	Client tells which server was accepted.
55	Parameter request list	Client's list of needed parameters.
56	Message	Used for errors. Server sends errors with DHCPNAK, and client uses DHCPDECLINE.
57	Max. DHCP message size	Largest DHCP message the client can accept.
58	Renewal time (T1)	Client will try to renew lease after this time.
59	Rebinding time (T2)	If lease renewal fails, client tries any server after this elapsed time (T2 must be greater than T1).
60	Class identifier	Vendor code describing client. Servers can reply based on this class.
61	Client identifier	Unique identifier for this client used by server to determine parameters.

DHCP AND ROUTERS

DHCP takes advantage of the BOOTP relay agent concept. In fact, router configuration of DHCP can be complicated because many routers mention only BOOTP relay agents and assume administrators know they are the same.

A DHCP relay agent is usually a router, but it could also be a dual-homed host that uses a router to reach the DHCP server. A typical configuration using a router as a relay agent was shown in Figure 18.1.

The DHCP relay agent listens for broadcast BOOTP request messages and sends them to the server. The relay agent then receives replies from the DHCP server and replies to the client.

DHCPv6

We haven't done anything with DHCP in IPv6. There's a reason for that, and it has to do with the way IPv6 configures itself on a host.

A lot of what DHCP does in IPv4 can also be done with RARP and ICMP. Yet DHCP is all over the place in IPv4. IPv6 includes elaborate neighbor and router discovery protocols that allow IPv6 hosts to invent link-local IPv6 addresses and multicast groups for configuration purposes. Yet, just like IPv4 DHCP for IPv6 exists as DHCPv6. There are at least three reasons DHCPv6 continues to make sense in IPv6.

- Not all networks support the multicasts needed for IPv6 autoconfiguration, like those consisting of point-to-point links or ATM and frame relay.
- Some small IPv6 networks might not have a router, which is required for IPv6 autoconfiguration.
- Network managers might desire more control over device configuration than afforded by IPv6 autoconfiguration.

DHCPv6 will not be used on the Illustrated Network. There is no BOOTP support because it is not really needed in IPv6. In truth, a lot of DHCP parameters are superfluous in IPv6. It is enough for this chapter to point out that DHCPv6 can be triggered by options in the IPV6 Router Advertisement messages, which we first introduced in Chapter 5.

DHCPv6 and Router Advertisements

DHCPv6 and its relationship to IPv6 addressing are described in a series of RFCs, most notably RFC 3315 and 3726. DHCPv6 can provide stateless or stateful address autoconfiguration information to IPv6 hosts. Stateless address autoconfiguration is used to configure both link-local and additional non–link-local addresses through the exchange of Router Solicitation and Router Advertisement messages with routers. Stateful address autoconfiguration is used to configure non–link-local addresses through the use of a configuration protocol such as DHCP.

How does a host know which one it can use? We did not emphasize it then, but our discussion of the IPv6 Router Advertisement protocol in Chapter 7 mentioned the M and O bit flags. The Router Advertisement message can set the following:

Managed Address Configuration Flag, known as the M flag—When set to 1, this bit instructs the host to use the configuration protocol to obtain a stateful (non-link-local) address.

Other Stateful Configuration Flag, known as the O flag—When set to 1, this bit instructs the host to use the configuration protocol to obtain more configuration settings.

There can be four different situations.

1. Both M and O flags are 0. This is used when the local network has no DHCPv6 infrastructure. IPv6 hosts use Router Advertisements and other methods, such as manual configuration, to get non-link-local addresses and other settings.
2. Both M and O flags are 1. In this case, DHCPv6 is used to obtain both addresses and other configuration settings. This is known as the "DHCPv6 stateful" situation, and DHCPv6 is used to assign stateful addresses to the IPv6 hosts.
3. M flag is 0, O flag is 1. DHCPv6 is not used to provide addresses, but only other configuration settings, such as the location of DNS servers. The routers are set to advertise non-link-local prefixes from which the IPv6 hosts can configure stateless addresses. This is known as "DHCPv6 stateless" because stateful addresses are not provided.
4. M flag is 1, O flag is 0. DHCPv6 is used to provide addresses, but no other settings. This combination is allowed but unlikely, because IPv6 hosts need to know other things, such as the addresses of the DNS servers.

Because we're not using DHCPv6 on the Illustrated Network, we won't detail the DHCPv4 message formats and exchange patterns—which are different for stateful and stateless operation.

DHCPv6 Operation

All DHCP servers and relay agents are required to join the local All-DHCP-Agents multicast group, and all servers must join the local All-DHCP-Servers group. All relay agents also join the local All-DHCP-Relays group.

DHCPv6 servers and agents send to UDP port 546, and clients send to UDP port 547. There are six message types defined for DHCPv6, and one nice feature is that the operation code (or message type byte) comes first in the message instead of being buried in the old BOOTP options field (as is DHCP for IPv4).

QUESTIONS FOR READERS

Figure 18.13 shows some of the concepts discussed in this chapter and can be used to help you answer the following questions.

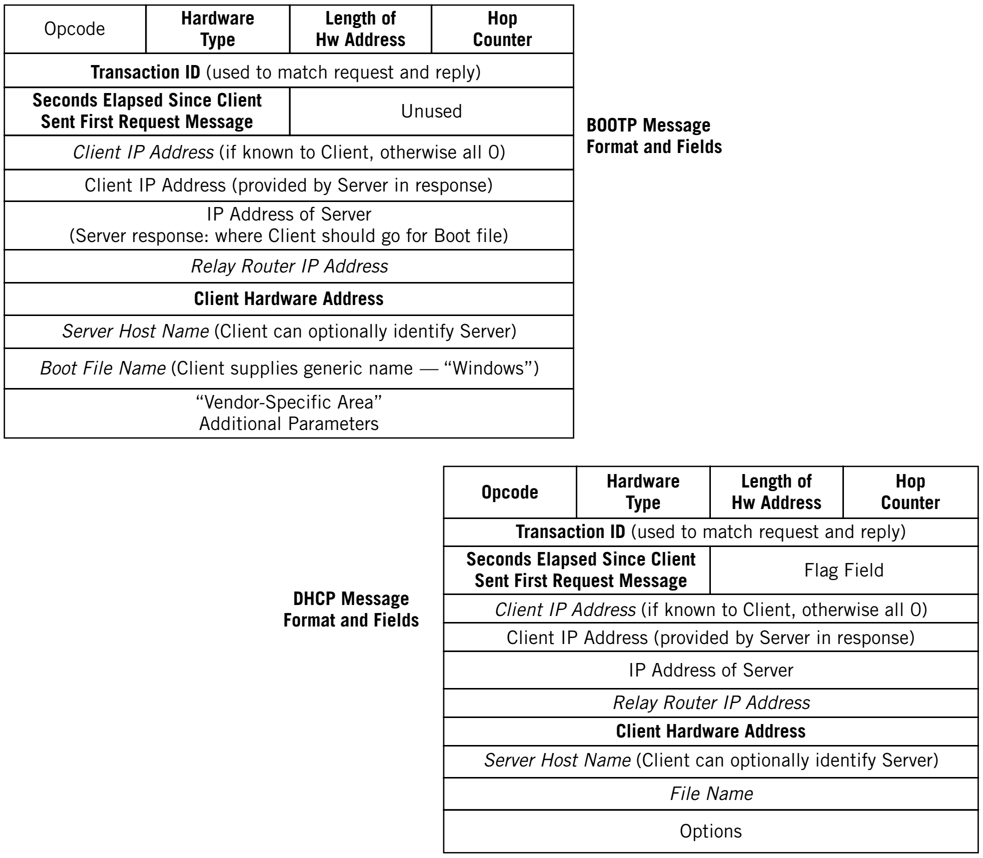

Opcode
Hardware Type
Length of Hw Address
Hop Counter
Transaction ID (used to match request and reply)
Seconds Elapsed Since Client Sent First Request Message
Unused
Client IP Address (if known to Client, otherwise all 0)
Client IP Address (provided by Server in response)
IP Address of Server (Server response: where Client should go for Boot file)
Relay Router IP Address
Client Hardware Address
Server Host Name (Client can optionally identify Server)
Boot File Name (Client supplies generic name — "Windows")
"Vendor-Specific Area" Additional Parameters
BOOTP Message Format and Fields
Opcode
Hardware Type
Length of Hw Address
Hop Counter
Transaction ID (used to match request and reply)
Seconds Elapsed Since Client Sent First Request Message
Flag Field
Client IP Address (if known to Client, otherwise all 0)
Client IP Address (provided by Server in response)
IP Address of Server
Relay Router IP Address
Client Hardware Address
Server Host Name (Client can optionally identify Server)
File Name
Options
DHCP Message Format and Fields

FIGURE 18.13

The BOOTP and DHCP messages compared.

1. The client sets the BOOTP hop count to zero initially. If that is the case, what is the hop counter used for?
2. What is the hardware type and hardware address length for Ethernet?
3. How is the relay router IP address field used?
4. What is the client ID option in DHCP?
5. What is the "magic cookie" IP address in BOOTP?