NAME

dhclient - Dynamic Host Configuration Protocol (DHCP) client

SYNOPSIS

dhclient [**-bdqu**] [**-c** *file*] [**-l** *file*] [**-p** *file*] *interface*

DESCRIPTION

The **dhclient** utility provides a means for configuring network interfaces using DHCP, BOOTP, or if these protocols fail, by statically assigning an address.

The name of the network interface that **dhclient** should attempt to configure must be specified on the command line.

The options are as follows:

-b Forces **dhclient** to immediately move to the background.

-c *file* Specify an alternate location, *file*, for the configuration file.

-d Forces **dhclient** to always run as a foreground process. By default, **dhclient** runs in the foreground until it has configured the interface, and then will revert to running in the background.

-l *file* Specify an alternate location, *file*, for the leases file.

-p *file* Specify an alternate location for the PID file. The default is */var/run/dhclient/dhclient.interface.pid*.

-q Forces **dhclient** to be less verbose on startup.

-u Forces **dhclient** to reject leases with unknown options in them. The default behaviour is to accept such lease offers.

The DHCP protocol allows a host to contact a central server which maintains a list of IP addresses which may be assigned on one or more subnets. A DHCP client may request an address from this pool, and then use it on a temporary basis for communication on the network. The DHCP protocol also provides a mechanism whereby a client can learn important details about the network to which it is attached, such as the location of a default router, the location of a name server, and so on.

On startup, **dhclient** reads */etc/dhclient.conf* for configuration instructions. It then gets a list of all the

network interfaces that are configured in the current system. It then attempts to configure each interface with DHCP.

In order to keep track of leases across system reboots and server restarts, **dhclient** keeps a list of leases it has been assigned in the */var/db/dhclient.leases.IFNAME* file. *IFNAME* represents the network interface of the DHCP client (e.g., em0), one for each interface. On startup, after reading the dhclient.conf(5) file, **dhclient** reads the leases file to refresh its memory about what leases it has been assigned.

Old leases are kept around in case the DHCP server is unavailable when **dhclient** is first invoked (generally during the initial system boot process). In that event, old leases from the *dhclient.leases.IFNAME* file which have not yet expired are tested, and if they are determined to be valid, they are used until either they expire or the DHCP server becomes available.

A mobile host which may sometimes need to access a network on which no DHCP server exists may be preloaded with a lease for a fixed address on that network. When all attempts to contact a DHCP server have failed, **dhclient** will try to validate the static lease, and if it succeeds, it will use that lease until it is restarted.

A mobile host may also travel to some networks on which DHCP is not available but BOOTP is. In that case, it may be advantageous to arrange with the network administrator for an entry on the BOOTP database, so that the host can boot quickly on that network rather than cycling through the list of old leases.

NOTES

You must have the Berkeley Packet Filter (BPF) configured in your kernel. The **dhclient** utility requires at least one */dev/bpf** device for each broadcast network interface that is attached to your system. See bpf(4) for more information.

FILES

/etc/dhclient.conf	DHCP client configuration file
/var/db/dhclient.leases.IFNAME	database of acquired leases

SEE ALSO

dhclient.conf(5), dhclient.leases(5), dhclient-script(8)

AUTHORS

The **dhclient** utility was written by Ted Lemon <*mellon@fugue.com*> and Elliot Poger <*elliot@poger.com*>.

The current implementation was reworked by Henning Brauer <*henning@openbsd.org*>.

BUGS

The **dhclient** utility uses capsicum(4) to sandbox the main process. If the requisite kernel support is not available, the main process will attempt to run in a chroot(2) sandbox instead. This will fail if the process is jailed or the *kern.chroot_allow_open_directories* sysctl is set to 0.