dnsmasq

dnsmasq (http://www.thekelleys.org.uk/dnsmasq/doc.html) provides a local **DNS server**, a **DHCP server** with support for **DHCPv6** and **PXE**, and a **TFTP server**. It is designed to be lightweight and have a small footprint, suitable for resource constrained routers and firewalls. dnsmasq can also be configured to cache DNS queries for improved DNS lookup speeds to previously visited sites.

Related articles

BIND

DNSCrypt

DNSSEC

Pdnsd

unbound

Contents

Installation

Install the `dnsmasq (https://www.archlinux.org/packages/?name=dnsmasq)` package.

Configuration

To configure dnsmasq, you need to edit `/etc/dnsmasq.conf`. The file contains extensive comments explaining its options. For all available options see `dnsmasq(8) (https://jlk.fjfi.cvut.cz/arch/manpages/man/dnsmasq.8)`.

Warning: dnsmasq by default enables its DNS server. If you do not require it, you need to explicitly disable it by setting DNS port to `0`:

/etc/dnsmasq.conf

```
port=0
```

Tip: To check configuration file(s) syntax, execute:

```
$ dnsmasq --test
```

Start the daemon

Start/enable `dnsmasq.service`.

To see if dnsmasq started properly, check the system's journal:

```
$ journalctl -u dnsmasq.service
```

The network will also need to be restarted so the DHCP client can create a new `/etc/resolv.conf`.

DNS cache setup

To set up dnsmasq as a DNS caching daemon on a single computer edit `/etc/dnsmasq.conf` and uncomment the `listen-address` directive, adding in the localhost IP address:

```
listen-address=127.0.0.1
```

To use this computer to listen on its LAN IP address for other computers on the network:

```
listen-address=192.168.1.1    # Example IP
```

It is recommended that you use a static LAN IP in this case.

Multiple ip address settings:

```
listen-address=127.0.0.1,192.168.1.1
```

DNS addresses file

After configuring dnsmasq, the DHCP client will need to prepend the localhost address to the known DNS addresses in `/etc/resolv.conf`. This causes all queries to be sent to dnsmasq before trying to resolve them with an external DNS. After the DHCP client is configured, the network will need to be restarted for changes to take effect.

resolv.conf

One option is a pure `resolv.conf` configuration. To do this, just make the first nameserver in `/etc/resolv.conf` point to localhost:

/etc/resolv.conf

```
nameserver 127.0.0.1
# External nameservers
...
```

Now DNS queries will be resolved first with dnsmasq, only checking external servers if dnsmasq cannot resolve the query. **dhcpcd (https://www.archlinux.org/packages/?name=dhcpcd)**, unfortunately, tends to overwrite `/etc/resolv.conf` by default, so if you use DHCP it is a good idea to protect `/etc/resolv.conf`. To do this, append `nohook resolv.conf` to the dhcpcd config file:

/etc/dhcpcd.conf

```
...
nohook resolv.conf
```

It is also possible to write protect your resolv.conf:

```
# chattr +i /etc/resolv.conf
```

More than three nameservers

A limitation in the way Linux handles DNS queries is that there can only be a maximum of three nameservers used in `resolv.conf`. As a workaround, you can make localhost the only nameserver in `resolv.conf`, and then create a separate `resolv-file` for your external nameservers. First, create a new resolv file for dnsmasq:

```
/etc/resolv.dnsmasq.conf
```

```
# Google's nameservers, for example
nameserver 8.8.8.8
nameserver 8.8.4.4
```

And then edit `/etc/dnsmasq.conf` to use your new resolv file:

```
/etc/dnsmasq.conf
```

```
...
resolv-file=/etc/resolv.dnsmasq.conf
...
```

dhcpcd

dhcpcd has the ability to prepend or append nameservers to `/etc/resolv.conf` by creating (or editing) the `/etc/resolv.conf.head` and `/etc/resolv.conf.tail` files respectively:

```
echo "nameserver 127.0.0.1" > /etc/resolv.conf.head
```

dhclient

For **dhclient (https://www.archlinux.org/packages/?name=dhclient)**, uncomment in `/etc/dhclient.conf`:

```
prepend domain-name-servers 127.0.0.1;
```

NetworkManager

NetworkManager has a plugin to enable DNS using dnsmasq. The advantages of this setup is that DNS lookups will be cached, shortening resolve times, and DNS lookups of VPN hosts will be routed to the relevant VPN's DNS servers (especially useful if you are connected to more than one VPN).

Make sure **dnsmasq (https://www.archlinux.org/packages/?name=dnsmasq)** has been installed, but has been disabled. Then, edit `/etc/NetworkManager/NetworkManager.conf` and change the `dns` in the `[main]` section:

/etc/NetworkManager/NetworkManager.conf

```
[main]
...
dns=dnsmasq
```

Now restart NetworkManager or reboot. NetworkManager will automatically start dnsmasq and add 127.0.0.1 to `/etc/resolv.conf`. The actual DNS servers can be found in `/run/NetworkManager/resolv.conf`. You can verify dnsmasq is being used by doing the same DNS lookup twice with `$ drill example.com` and verifying the server and query times.

Custom configuration

Custom configurations can be created for *dnsmasq* by creating configuration files in `/etc/NetworkManager/dnsmasq.d/`. For example, to change the size of the DNS cache (which is stored in RAM):

/etc/NetworkManager/dnsmasq.d/cache.conf

```
cache-size=1000
```

IPv6

Enabling `dnsmasq` in NetworkManager may break IPv6-only DNS lookups (i.e. `drill -6 [hostname]`) which would otherwise work. In order to resolve this, creating the following file will configure *dnsmasq* to also listen to the IPv6 loopback:

```
/etc/NetworkManager/dnsmasq.d/ipv6_listen.conf
```

```
listen-address=::1
```

In addition, `dnsmasq` also does not prioritize upstream IPv6 DNS. Unfortunately NetworkManager does not do this (**Ubuntu Bug (https://bugs.launchpad.net/ubuntu/+source/network-manager/+bug/936712)**). A workaround would be to disable IPv4 DNS in the NetworkManager config, assuming one exists

Other methods

Another option is in NetworkManagers' settings (usually by right-clicking the applet) and entering settings manually. Setting up will depending on the type of front-end used; the process usually involves right-clicking on the applet, editing (or creating) a profile, and then choosing DHCP type as 'Automatic (specify addresses).' The DNS addresses will need to be entered and are usually in this form: `127.0.0.1, DNS-server-one, ...`.

Test

To do a lookup speed test choose a website that has not been visited since dnsmasq has been started (*drill* is part of the **ldns (https://www.archlinux.org/packages/?name=ldns)** package):

```
$ drill archlinux.org | grep "Query time"
```

Running the command again will use the cached DNS IP and result in a faster lookup time if dnsmasq is setup correctly:

```
$ drill archlinux.org | grep "Query time"
;; Query time: 18 msec
```

```
$ drill archlinux.org | grep "Query time"
;; Query time: 2 msec
```

Server setup

DNS server

Adding a custom domain

It is possible to add a custom domain to hosts in your (local) network:

```
local=/home.lan/
domain=home.lan
```

In this example it is possible to ping a host/device (e.g. defined in your `/etc/hosts` file) as `hostname.home.lan`.

Uncomment `expand-hosts` to add the custom domain to hosts entries:

```
expand-hosts
```

Without this setting, you will have to add the domain to entries of `/etc/hosts`.

DHCP server

By default dnsmasq has the DHCP functionality turned off, if you want to use it you must turn it on in (`/etc/dnsmasq.conf`). Here are the important settings:

```
# Only listen to routers' LAN NIC.  Doing so opens up tcp/udp port 53 to
# localhost and udp port 67 to world:
interface=<LAN-NIC>

# dnsmasq will open tcp/udp port 53 and udp port 67 to world to help with
# dynamic interfaces (assigning dynamic ips). Dnsmasq will discard world
# requests to them, but the paranoid might like to close them and let the
# kernel handle them:
bind-interfaces

# Optionally set a domain name
domain=example.com

# Set default gateway
dhcp-option=3,192.168.1.1
```

```
# Set DNS servers to announce
dhcp-option=6,8.8.8.8,8.8.4.4

# Dynamic range of IPs to make available to LAN PC and the lease time.
# Ideally set the lease time to 5m only at first to test everything works okay before you set long-lasting records.
dhcp-range=192.168.111.50,192.168.111.100,12h

# If you'd like to have dnsmasq assign static IPs to some clients, bind the LAN computers
# NIC MAC addresses:
dhcp-host=aa:bb:cc:dd:ee:ff,192.168.111.50
dhcp-host=aa:bb:cc:ff:dd:ee,192.168.111.51
```

See **dnsmasq(8) (https://jlk.fjfi.cvut.cz/arch/manpages/man/dnsmasq.8)** for more options.

Test

From a computer that is connected to the one with dnsmasq on it, configure it to use DHCP for automatic IP address assignment, then attempt to log into the network normally.

If you inspect the `/var/lib/misc/dnsmasq.leases` file on the server, you should be able to see the lease.

TFTP server

dnsmasq has built-in **TFTP** server.

To use it, create a directory for TFTP root (e.g. `/srv/tftp`) to put transferable files in.

For increased security it is advised to use dnsmasq's TFTP secure mode. In secure mode only files owned by the `dnsmasq` user will be served over TFTP. You will need to **chown** TFTP root and all files in it to `dnsmasq` user to use this feature.

Enable TFTP in `dnsmasq.conf`:

```
/etc/dnsmasq.conf
```

```
enable-tftp
tftp-root=/srv/tftp
tftp-secure
```

See **dnsmasq(8) (https://jlk.fjfi.cvut.cz/arch/manpages/man/dnsmasq.8)** for more options.

PXE server

PXE requires DHCP and TFTP servers, both functions can be provided by dnsmasq.

Tip: dnsmasq can add PXE booting options to a network with an already running DHCP server:

```
/etc/dnsmasq.conf
```

```
interface=enp0s0
bind-dynamic
dhcp-range=192.168.0.1,proxy
```

1. set up **#TFTP server** and **#DHCP server**
2. copy and configure a PXE compatible bootloader (e.g. **PXELINUX**) on TFTP root
3. enable PXE in `/etc/dnsmasq.conf`:

Note:

- file paths are relative to TFTP root
- if the file has a `.0` suffix, you must exclude the suffix in `pxe-service` options

To simply send one file:

```
dhcp-boot=lpxelinux.0
```

To send a file depending on client architecture:

```
pxe-service=x86PC, "PXELINUX (BIOS)", "bios/lpxelinux"
pxe-service=X86-64_EFI, "PXELINUX (EFI)", "efi64/syslinux.efi"
```

Note: In case `pxe-service` does not work (especially for UEFI-based clients), combination of `dhcp-match` and `dhcp-boot` can be used. See **RFC4578 (https://tools.ietf.org/html/rfc4578#section-2.1)** for more `client-arch` numbers for use with dhcp boot protocol.

```
dhcp-match=set:efi-x86_64,option:client-arch,7
dhcp-match=set:efi-x86_64,option:client-arch,9
dhcp-match=set:efi-x86,option:client-arch,6
dhcp-match=set:bios,option:client-arch,0
```

```
dhcp-boot=tag:efi-x86_64,"efi64/syslinux.efi"
dhcp-boot=tag:efi-x86,"efi32/syslinux.efi"
dhcp-boot=tag:bios,"bios/lpxelinux.0"
```

See **dnsmasq(8) (https://jlk.fjfi.cvut.cz/arch/manpages/man/dnsmasq.8)** for more options.

The rest is up to the **bootloader**.

Tips and tricks

Prevent OpenDNS redirecting Google queries

To prevent OpenDNS from redirecting all Google queries to their own search server, add to `/etc/dnsmasq.conf`:

```
server=/www.google.com/<ISP DNS IP>
```

Override addresses

In some cases, such as when operating a captive portal, it can be useful to resolve specific domains names to a hard-coded set of addresses. This is done with the `address` config:

```
address=/example.com/1.2.3.4
```

Furthermore, it's possible to return a specific address for all domain names that are not answered from `/etc/hosts` or DHCP by using a special wildcard:

```
address=/#/1.2.3.4
```

More than one instance

If we want two or more dnsmasq servers works per interface(s).

Static

To do this staticly, server per interface, use `interface` and `bind-interface` options. This enforce start second dnsmasq.

Dynamic

In this case we can exclude per interface and bind any others:

```
except-interface=lo
bind-dynamic
```

Note: This is default in libvirt.

See also

- **Caching Nameserver using dnsmasq, and a few other tips and tricks. (http://www.g-loaded.eu/2010/09/18/caching-nameserver-using-dnsmasq/)**

Retrieved from "https://wiki.archlinux.org/index.php?title=Dnsmasq&oldid=506962"