<https://www.canonical.com>

Network Configuration

Ubuntu ships with a number of graphical utilities to configure your network devices. This document is geared toward server administrators and will focus on managing your network on the command line.

Ethernet Interfaces

Ethernet interfaces are identified by the system using predictable network interface names. These names can appear as *eno1* or *enp0s25*. However, in some cases an interface may still use the kernel *eth#* style of naming.

Identify Ethernet Interfaces

To quickly identify all available Ethernet interfaces, you can use the ip command as shown below.

```
ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s25: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 00:16:3e:e2:52:42 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.102.66.200/24 brd 10.102.66.255 scope global dynamic eth0
       valid_lft 3257sec preferred_lft 3257sec
    inet6 fe80::216:3eff:fee2:5242/64 scope link
       valid_lft forever preferred_lft forever
```

Another application that can help identify all network interfaces available to your system is the lshw command. This command provides greater details around the hardware capabilities of specific adapters. In the example below, lshw shows a single Ethernet interface with the logical name of *eth0* along with bus information, driver details and all supported capabilities.

```
sudo lshw -class network
  *-network
       description: Ethernet interface
       product: MT26448 [ConnectX EN 10GigE, PCIe 2.0 5GT/s]
       vendor: Mellanox Technologies
       physical id: 0
       bus info: pci@0004:01:00.0
       logical name: eth4
       version: b0
       serial: e4:1d:2d:67:83:56
       slot: U78CB.001.WZS09KB-P1-C6-T1
       size: 10Gbit/s
       capacity: 10Gbit/s
       width: 64 bits
       clock: 33MHz
       capabilities: pm vpd msix pciexpress bus_master cap_list ethernet physical fibre 10000bt-fd
       configuration: autonegotiation=off broadcast=yes driver=mlx4_en driverversion=4.0-0 duplex=full firmware=2.9.1326 ip=192.168.1.1 latency=0 link=yes multicast=yes port=fibre speed=10Gbit/s
       resources: iomemory:24000-23fff irq:481 memory:3fe200000000-3fe2000fffff memory:240000000000-240007ffffff
```

Ethernet Interface Logical Names

Interface logical names can also be configured via a netplan configuration. If you would like control which interface receives a particular logical name use the *match* and *set-name* keys. The match key is used to find an adapter based on some criteria like MAC address, driver, etc. Then the set-name key can be used to change the device to the desired logial name.

```
network:
  version: 2
  renderer: networkd
  ethernets:
    eth_lan0:
      dhcp4: true
      match:
        macaddress: 00:11:22:33:44:55
      set-name: eth_lan0
```

Ethernet Interface Settings

ethtool is a program that displays and changes Ethernet card settings such as auto-negotiation, port speed, duplex mode, and Wake-on-LAN. The following is an example of how to view supported features and configured settings of an Ethernet interface.

```
sudo ethtool eth4
Settings for eth4:
    Supported ports: [ FIBRE ]
    Supported link modes:   10000baseT/Full
    Supported pause frame use: No
    Supports auto-negotiation: No
    Supported FEC modes: Not reported
    Advertised link modes:  10000baseT/Full
    Advertised pause frame use: No
    Advertised auto-negotiation: No
    Advertised FEC modes: Not reported
    Speed: 10000Mb/s
    Duplex: Full
    Port: FIBRE
    PHYAD: 0
    Transceiver: internal
    Auto-negotiation: off
    Supports Wake-on: d
    Wake-on: d
    Current message level: 0x00000014 (20)
                   link ifdown
    Link detected: yes
```

IP Addressing

The following section describes the process of configuring your systems IP address and default gateway needed for communicating on a local area network and the Internet.

Temporary IP Address Assignment

For temporary network configurations, you can use the ip command which is also found on most other GNU/Linux operating systems. The ip command allows you to configure settings which take effect immediately, however they are not persistent and will be lost after a reboot.

To temporarily configure an IP address, you can use the ip command in the following manner. Modify the IP address and subnet mask to match your network requirements.

```
sudo ip addr add 10.102.66.200/24 dev enp0s25
```

The ip can then be used to set the link up or down.

```
ip link set dev enp0s25 up
ip link set dev enp0s25 down
```

To verify the IP address configuration of enp0s25, you can use the ip command in the following manner.

```
ip address show dev enp0s25
10: enp0s25: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 00:16:3e:e2:52:42 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.102.66.200/24 brd 10.102.66.255 scope global dynamic eth0
       valid_lft 2857sec preferred_lft 2857sec
    inet6 fe80::216:3eff:fee2:5242/64 scope link
       valid_lft forever preferred_lft forever6
```

To configure a default gateway, you can use the ip command in the following manner. Modify the default gateway address to match your network requirements.

```
sudo ip route add default via 10.102.66.1
```

To verify your default gateway configuration, you can use the ip command in the following manner.

```
ip route show
default via 10.102.66.1 dev eth0 proto dhcp src 10.102.66.200 metric 100
10.102.66.0/24 dev eth0 proto kernel scope link src 10.102.66.200
10.102.66.1 dev eth0 proto dhcp scope link src 10.102.66.200 metric 100
```

If you require DNS for your temporary network configuration, you can add DNS server IP addresses in the file `/etc/resolv.conf`. In general, editing `/etc/resolv.conf` directly is not recommanded, but this is a temporary and non-persistent configuration. The example below shows how to enter two DNS servers to `/etc/resolv.conf`, which should be changed to servers appropriate for your network. A more lengthy description of the proper persistent way to do DNS client configuration is in a following section.

```
nameserver 8.8.8.8
nameserver 8.8.4.4
```

If you no longer need this configuration and wish to purge all IP configuration from an interface, you can use the ip command with the flush option as shown below.

```
ip addr flush eth0
```

> **Note**
>
> Flushing the IP configuration using the ip command does not clear the contents of `/etc/resolv.conf`. You must remove or modify those entries manually, or re-boot which should also cause `/etc/resolv.conf`, which is a symlink to `/run/systemd/resolve/stub-resolv.conf`, to be re-written.

Dynamic IP Address Assignment (DHCP Client)

To configure your server to use DHCP for dynamic address assignment, create a netplan configuration in the file `/etc/netplan/99_config.yaml`. The example below assumes you are configuring your first Ethernet interface identified as *enp3s0*.

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp3s0:
      dhcp4: true
```

The configuration can then be applied using the netplan command.

```
sudo netplan apply
```

Static IP Address Assignment

To configure your system to use static address assignment, create a netplan configuration in the file `/etc/netplan/99_config.yaml`. The example below assumes you are configuring your first Ethernet interface identified as *eth0*. Change the *addresses*, *gateway4*, and *nameservers* values to meet the requirements of your network.

```
network:
  version: 2
  renderer: networkd
  ethernets:
    eth0:
      addresses:
        - 10.10.10.2/24
      gateway4: 10.10.10.1
      nameservers:
          search: [mydomain, otherdomain]
          addresses: [10.10.10.1, 1.1.1.1]
```

The configuration can then be applied using the netplan command.

```
sudo netplan apply
```

Loopback Interface

The loopback interface is identified by the system as *lo* and has a default IP address of 127.0.0.1. It can be viewed using the ip command.

```
ip address show lo
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
```

Name Resolution

Name resolution as it relates to IP networking is the process of mapping IP addresses to hostnames, making it easier to identify resources on a network. The following section will explain how to properly configure your system for name resolution using DNS and static hostname records.

DNS Client Configuration

Traditionally, the file `/etc/resolv.conf` was a static configuration file that rarely needed to be changed or automatically changed via DCHP client hooks. Systemd-resolved handles name server configuration, and it should be interacted with through the `systemd-resolve` command. Netplan configures systemd-resolved to generate a list of nameservers and domains to put in `/etc/resolv.conf`, which is a symlink:

```
/etc/resolv.conf -> ../run/systemd/resolve/stub-resolv.conf
```

To configure the resolver, add the IP addresses of the nameservers that are appropriate for your network to the netplan configuration file. You can also add an optional DNS suffix search-lists to match your network domain names. The resulting file might look like the following:

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s25:
      addresses:
        - 192.168.0.100/24
      gateway4: 192.168.0.1
      nameservers:
          search: [mydomain, otherdomain]
          addresses: [1.1.1.1, 8.8.8.8, 4.4.4.4]
```

The *search* option can also be used with multiple domain names so that DNS queries will be appended in the order in which they are entered. For example, your network may have multiple sub-domains to search; a parent domain of *example.com*, and two sub-domains, *sales.example.com* and *dev.example.com*.

If you have multiple domains you wish to search, your configuration might look like the following:

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s25:
      addresses:
        - 192.168.0.100/24
      gateway4: 192.168.0.1
      nameservers:
          search: [example.com, sales.example.com, dev.example.com]
          addresses: [1.1.1.1, 8.8.8.8, 4.4.4.4]
```

If you try to ping a host with the name of *server1*, your system will automatically query DNS for its Fully Qualified Domain Name (FQDN) in the following order:

1. `server1.example.com`
2. `server1.sales.example.com`
3. `server1.dev.example.com`

If no matches are found, the DNS server will provide a result of *notfound* and the DNS query will fail.

Static Hostnames

Static hostnames are locally defined hostname-to-IP mappings located in the file `/etc/hosts`. Entries in the `hosts` file will have precedence over DNS by default. This means that if your system tries to resolve a hostname and it matches an entry in /etc/hosts, it will not attempt to look up the record in DNS. In some configurations, especially when Internet access is not required, servers that communicate with a limited number of resources can be conveniently set to use static hostnames instead of DNS.

The following is an example of a `hosts` file where a number of local servers have been identified by simple hostnames, aliases and their equivalent Fully Qualified Domain Names (FQDN's).

```
127.0.0.1   localhost
127.0.1.1   ubuntu-server
10.0.0.11   server1 server1.example.com vpn
10.0.0.12   server2 server2.example.com mail
10.0.0.13   server3 server3.example.com www
10.0.0.14   server4 server4.example.com file
```

Note

In the above example, notice that each of the servers have been given aliases in addition to their proper names and FQDN's. *Server1* has been mapped to the name *vpn*, *server2* is referred to as *mail*, *server3* as *www*, and *server4* as *file*.

Name Service Switch Configuration

The order in which your system selects a method of resolving hostnames to IP addresses is controlled by the Name Service Switch (NSS) configuration file `/etc/nsswitch.conf`. As mentioned in the previous section, typically static hostnames defined in the systems `/etc/hosts` file have precedence over names resolved from DNS. The following is an example of the line responsible for this order of hostname lookups in the file `/etc/nsswitch.conf`.

```
hosts:          files mdns4_minimal [NOTFOUND=return] dns mdns4
```

- **files** first tries to resolve static hostnames located in `/etc/hosts`.
- **mdns4_minimal** attempts to resolve the name using Multicast DNS.
- **[NOTFOUND=return]** means that any response of *notfound* by the preceding *mdns4_minimal* process should be treated as authoritative and that the system should not try to continue hunting for an answer.
- **dns** represents a legacy unicast DNS query.
- **mdns4** represents a Multicast DNS query.

To modify the order of the above mentioned name resolution methods, you can simply change the *hosts:* string to the value of your choosing. For example, if you prefer to use legacy Unicast DNS versus Multicast DNS, you can change the string in `/etc/nsswitch.conf` as shown below.

```
hosts:          files dns [NOTFOUND=return] mdns4_minimal mdns4
```

Bridging

Bridging multiple interfaces is a more advanced configuration, but is very useful in multiple scenarios. One scenario is setting up a bridge with multiple network interfaces, then using a firewall to filter traffic between two network segments. Another scenario is using bridge on a system with one interface to allow virtual machines direct access to the outside network. The following example covers the latter scenario.

Configure the bridge by editing your netplan configuration found in `/etc/netplan/`:

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp3s0:
      dhcp4: no
  bridges:
    br0:
      dhcp4: yes
      interfaces:
        - enp3s0
```

Note

Enter the appropriate values for your physical interface and network.

Now apply the configuration to enable the bridge:

```
sudo netplan apply
```

The new bridge interface should now be up and running. The brctl provides useful information about the state of the bridge, controls which interfaces are part of the bridge, etc. See `man brctl` for more information.

Resources

- The **Ubuntu Wiki Network page <https://help.ubuntu.com/community/Network>** has links to articles covering more advanced network configuration.
- The **netplan website <https://netplan.io>** has additional examples and documentation.
- The **netplan man page <https://manpages.ubuntu.com/manpages/focal/man5/netplan.5.html>** has more information on netplan.
- The **systemd-resolve man page <https://manpages.ubuntu.com/manpages/focal/man1/systemd-resolve.1.html>** has details on systemd-resolve command.
- The **systemd-resolved man page <https://manpages.ubuntu.com/manpages/focal/man8/systemd-resolved.8.html>** has more information on systemd-resolved service.
- For more information on *bridging* see the **netplan.io examples page <https://netplan.io/examples>** and the Linux Foundation's **Networking-Bridge <http://www.linuxfoundation.org/collaborate/workgroups/networking/bridge>** page.

Last updated 3 months ago. **Help improve this document in the forum <https://discourse.ubuntu.com/t/network-configuration/11876>** .