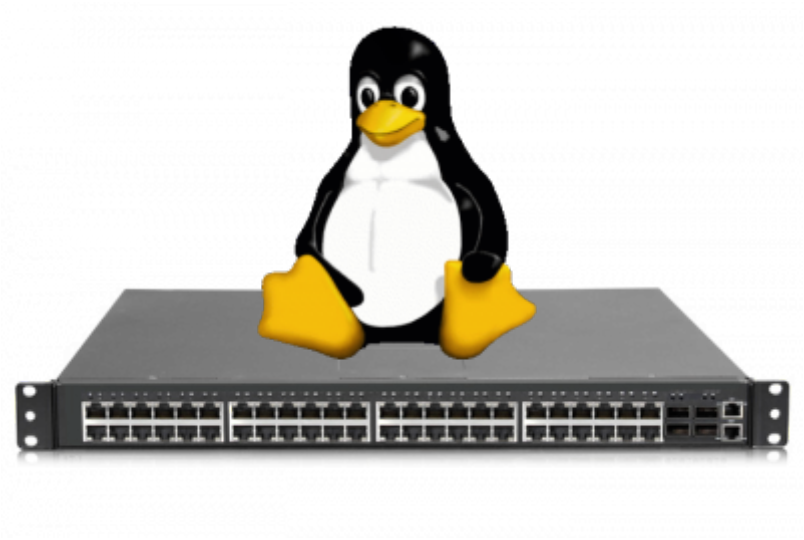
Netzwerk Konfiguration unter Debian / Ubuntu



Netzkonfiguration ab Ubuntu 18.04 / Debian X

Ubuntu 17.10 and later uses **Netplan** as the default network management tool. The previous Ubuntu versions were using ifconfig and its configuration file `/etc/network/interfaces` to configure the network. Netplan configuration files are written in YAML syntax with a `.yaml` file extension. To configure a network interface with Netplan, you need to create a YAML description for the interface, and Netplan will generate the required configuration files for the chosen renderer tool. Netplan supports two renderers, NetworkManager and Systemd-networkd. NetworkManager is mostly used on Desktop machines, while the Systemd-networkd is used on servers without a GUI.

Configuring Static IP address

On Ubuntu 20.04, the system identifies network interfaces using ‘predictable network interface names’.

The first step toward setting up a static IP address is identifying the name of the ethernet interface you want to configure. To do so, use the ip link command, as shown below:

```
# ip link
```

The command prints a list of all the available network interfaces. In this example, the name of the interface is `ens3`:

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode
DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
```

```
2: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
mode DEFAULT group default qlen 1000
    link/ether 08:00:27:6c:13:63 brd ff:ff:ff:ff:ff:ff
```

Netplan configuration files are stored in the `/etc/netplan` directory. You'll probably find one or more YAML files in this directory. The name of the file may differ from setup to setup. Usually, the file is named either `01-netcfg.yaml`, `50-cloud-init.yaml`, or `NN_interfaceName.yaml`, but in your system it may be different.

If your Ubuntu cloud instance is provisioned with cloud-init, you'll need to disable it. To do so create the following file:

```
# vim /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg
```

```
network: {config: disabled}
```

To assign a static IP address on the network interface, open the YAML configuration file with your text editor :

```
# vim /etc/netplan/01-netcfg.yaml
```

```
network:
  version: 2
  renderer: networkd
  ethernets:
    ens3:
      dhcp4: yes
```

Before changing the configuration, let's explain the code in a short.

Each Netplan Yaml file starts with the network key that has at least two required elements. The first required element is the version of the network configuration format, and the second one is the device type. The device type can be ethernets, bonds, bridges, or vlans.

The configuration above also has a line that shows the renderer type. Out of the box, if you installed Ubuntu in server mode, the renderer is configured to use networkd as the back end. Under the device's type (`ethernets`), you can specify one or more network interfaces. In this example, we have only one interface ens3 that is configured to obtain IP addressing from a DHCP server `dhcp4: yes`.

To assign a static IP address to `ens3` interface, edit the file as follows:

- Set DHCP to `dhcp4: no`.
- Specify the static IP address. Under `addresses:` you can add one or more IPv4 or IPv6 IP addresses that will be assigned to the network interface.
- Specify the gateway.
- Under `nameservers`, set the IP addresses of the nameservers.

```
network:
  version: 2
  renderer: networkd
  ethernets:
    ens3:
      dhcp4: no
      addresses:
        - 192.168.121.221/24
      gateway4: 192.168.121.1
      nameservers:
          addresses: [8.8.8.8, 1.1.1.1]
```

When editing Yaml files, make sure you follow the YAML code indent standards. If the syntax is not correct, the changes will not be applied.

Once done, save the file and apply the changes by running the following command:

```
# netplan apply
```

Verify the changes by typing:

```
# ip addr show dev ens3
```

```
2: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group
default qlen 1000
    link/ether 08:00:27:6c:13:63 brd ff:ff:ff:ff:ff:ff
    inet 192.168.121.221/24 brd 192.168.121.255 scope global dynamic ens3
       valid_lft 3575sec preferred_lft 3575sec
    inet6 fe80::5054:ff:feb0:f500/64 scope link
       valid_lft forever preferred_lft forever
```

That’s it! You have assigned a static IP to your Ubuntu server.

Statische IP - Beispiel Netzkonfiguration bis Ubuntu 16.04 / Debian X

Dies ist eine Beispielkonfiguration, in der zwei Interfaces **eth0** und **eth1** **mit statischer IP** und Angaben definiert wurden. Man kann nach diesem Beispiel noch beliebig weitere Interfaces erstellen.

```
# vim /etc/network/interfaces
```

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet static
address 192.168.1.23
netmask 255.255.255.0
gateway 192.168.1.1
dns-nameservers 192.168.1.1

# BlackSERVhost to BlackSERV
auto eth1
iface eth1 inet static
address 10.0.0.23
netmask 255.255.255.0
broadcast 10.0.0.255
```

- **auto ethX** sorgt dafür, dass das NIC bei jedem Neustart mit gestartet wird
- **iface ethX inet static** definiert für das NIC eine statische IP
  - **address abc.def.ghi.jkl** spezifiziert die IP
  - **netmask** die Netzmaske
  - **gateway** das Standardgateway
  - **dns-nameservers** spezifiziert die DNS-Server. Es können beliebig viele eingetragen werden, diese werden dann der Reihe nach angefragt

Nach dem eintragen, eines **neuen Interfaces**, **muss** dies bevor es verwendet werden kann, noch **aktiviert werden**. Dies geschieht z.B für das eth1 mit: → **`ifup eth1`**

Zum übernehmen der Interface Änderungen unten schauen!

Dynamische IP - Beispiel Netzkonfiguration für Ubuntu

Es soll nun nur noch ein Interface gebraucht werden (**eth0**). Dieses, muss auch noch eine DHCP Adresse bekommen. Dazu wird die Konfiguration folgendermassen abgeändert:

```
# vim /etc/network/interfaces
```

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet dhcp
#address 192.168.1.23
#netmask 255.255.255.0
#gateway 192.168.1.1
#dns-nameservers 192.168.1.1
```

- Nutzt man **iface ethX inet dhcp** versucht sich das NIC die Konfiguration über DHCP zu holen, dann darf aber nach dieser Zeile keine weiter Konfiguration für dieses NIC vorgenommen werden.

Zum übernehmen der Interface Änderungen unten schauen!

Übernehmen der neuen Netzwerk Konfiguration

Zum übernehmen / aktivieren der neuen Netzwerkkonfiguration, kann entweder das Netzwerk Interface deaktiviert und anschliessend wieder aktiviert werden; oder der Server rebootet werden.

- Deaktivieren und aktivieren des Netzwerkadapters eth0:

```
# ifdown eth0
# ifup eth0
```

- Neustarten des Systems:

```
# systemctl reboot
```

Weiteres bezüglich Netzwerk Konfiguration

DNS-Config

Neben der Konfiguration der DNS-Server in der Datei **/etc/network/interfaces** kann dies auch über die Datei **etc/resolv.conf** geschehen. Hier trägt man einen DNS-Server mit

```
nameserver ip.adresse.des.servers
```

ein. Hier ist Linux leider etwas inkonsequent! Gibt man DNS-Server in der **/etc/network/interfaces** an so muss man als key `dns-nameservers` nutzen, in der **/etc/resolv.conf** nutzt man `nameserver`

IP Forwarding

IP-Forwarding ist zuständig, dass IP-Pakete auf allen Netzwerkkarten weitergeleitet werden. Das benötigt man zum Beispiel bei einem gehosteten Subnetz bei einem ISP. Da dies ein Kernelmodul ist, muss es in `/etc/sysctl.conf` dauerhaft aktiviert werden. Hierzu muss die Zeile mit IP-Forward wie folgt angepasst werden:

```
net.ipv4.ip_forward=1
```

Möchte man dieses Modul nur temporär aktivieren macht man dies über das Terminal:

```
# sysctl net.ipv4.ip_forward=1
```

Statische Routen

Nutzt man mehrere Netzwerke ergibt sich manchmal das Problem, dass der Server die Route in ein bestimmten Netz nicht findet, da zum Beispiel der “Zugang” zu diesem Netz auf einem Server liegt, der kein Gateway ist (Beispielsweise bei OpenVPN). Um trotzdem auf diese Netze zu kommen kann man statische Routen festlegen: Dies geht temporär mit

```
sudo route add -net 10.8.0.0 netmask 255.255.255.0 gw vpn.server.i.p
```

Hierbei ist das LAN **10.8.0.0/24** das Netzwerk in das geroutet werden soll und **vpn.server.i.p** der Server, der den “physikalischen” Zugang zum Netz darstellt. Damit diese Routen persistent bleiben müssen sie in ohne `sudo` in die Datei **/etc/rc.local** eingetragen werden:

```
route add -net 10.8.0.0 netmask 255.255.255.0 gw vpn.server.i.p
```

Last update: **2021/05/28 13:43**