comment renommer les interfaces de réseau eth0-sur-centos

In this article, we will learn about the Linux interface naming. It is written for RHEL-based Linux distributions like AlmaLinux, CentOS and Rocky Linux. You will learn how the network interface on Linux named eth(X), ens(X), eno(X), etc.

Firstly, we will learn about the naming concept of a network interface on the Linux system, including the classic naming scheme, biosdevname, and system udev policy.

1. Traditional Naming Convention

The Traditional Naming Convention is the default traditional interface naming scheme that provided by Linux kernel driver. As we know, the Traditional Naming Convention is such as eth(n) for Ethernet device, wlan(n) for wireless LAN device, and usb(n) for the usbnet.

In the traditional naming convention, all network hardware will be handled as the same by the kernel driver. It's good enough to tell us about the type of connection, but not much for the nature hardware you're used to.

2. Biosdevname

The Biosdevname is a project that started by Dell to provide consistent naming of network devices. The Biosdevname is a udev utility helper that provided a consistent naming mechanism for network devices based on the physical location of the hardware as suggested by the system BIOS.

Below is the default policy of naming scheme provided by biosdevname:

- For the onboard network devices, will get named as "**em<port>[_<virtual instance>]**"
- For add-ons NICs devices, will get named as "**p<slot>p<port>[_<virtual instance>]"**

More details:

- **em** = onboard net devices
- **<slot>** = the respective PCI slot
- **<port>** = Port number of NICs (if have multi-port NICs)
- **<virtual instance>** = the SRIOV and/or NPAR instance index
- The '**p**' on add-on NIC is stand as pci slot and port respectly

3. Systemd Udev Rules

Nowadays, some popular Linux distros such as Ubuntu, CentOS, and Debian are using the Systemd as a system and service manager. And since version 197, systemd has been added to the native naming policies and scheme for network interfaces that similar to Biosdevname.

Below is 5 policies of network interface naming for Systemd based system:

1. For onboard network devices, it will be named such as 'eno1', 'eno2', etc.
2. For incorporating Firmware/BIOS provided PCI Express hotplug slot index numbers, named as 'ens1'. This will be applied if the first policy is failed. Mostly used by the virtual machine such as VMWare and KVM.
3. Names incorporating physical location of the connector of the hardware (example: enp2s0), are applied if applicable, else falling directly back to scheme 5 in all other cases.
4. Names incorporating interface's MAC address (example: enx78e7d1ea46da), is not used by default but is available if the user chooses.
5. The traditional unpredictable kernel naming scheme is used if all other methods fail (example: eth0).

Now we've learned about the Linux interface naming convention.

Next, we will show you how to change the network interfaces name to the traditional 'et0', eth1, etc. And there're two ways to do it, change the network interface names through the 'biosdevname' and through the 'systemd udev rules'.

1. Change Interface Name to eth0 by Disabling Biosdevname

As we know at the top, the biosdevname is an udev utility created by Dell for naming network interfaces as suggested by the system BIOS. And by default, the biosdevname package is installed on the Linux system.

To disable the biosdevname on the Linux system, you will need to disable it on the boot level by editing the grub configuration on your system.

Before going any further, let's check current interfaces on the system using the following command.

```
ifconfig -a
ip addr
```

And below is the result.

```
[root@hakase-centos8 ~]#
[root@hakase-centos8 ~]# ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        ether 08:00:27:4d:79:ed  txqueuelen 1000  (Ethernet)
        RX packets 1  bytes 590 (590.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3  bytes 462 (462.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.119  netmask 255.255.255.0  broadcast 192.168.56.255
        ether 08:00:27:9d:82:32  txqueuelen 1000  (Ethernet)
        RX packets 195  bytes 19041 (18.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 116  bytes 17215 (16.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@hakase-centos8 ~]#
```

As a result' we've two network interfaces 'ens3' and 'ens8'.

Next, we will rename both interfaces to 'eth0' and 'eth1' on our CentOS 8 system by editing the grub configuration and network configuration for each interface.

- Setup Grub2

Go to the '/etc/sysconfig/' directory and edit the 'grub' configuration using vim editor.

```
cd /etc/sysconfig/
vim grub
```

On the 'GRUB_CMDLINE_LINUX' configuration, add the following configuration for disabling the biosdevname and the net.ifnames kernel parameter.

```
net.ifnames=0 biosdevname=0
```

Save and close.

```
GRUB_TIMEOUT=1
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="no_timer_check crashkernel=auto resume=/dev/mapper/cl_centos8-swap rd.lvm.lv=cl_centos8/root rd.lvm.lv=cl_centos8/swap net.ifnames=0 biosdevname=0 rhgb quiet"
GRUB_DISABLE_RECOVERY="true"
GRUB_ENABLE_BLSCFG=true
~
~
~
~
~
```

After that, generate a new grub configuration using the 'grub2-mkconfig' command as below.

```
sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

And you will get the result as below.

```
[root@hakase-centos8 ~]#
[root@hakase-centos8 ~]#
[root@hakase-centos8 ~]# cd /etc/sysconfig/
[root@hakase-centos8 sysconfig]#
[root@hakase-centos8 sysconfig]# vim grub
[root@hakase-centos8 sysconfig]#
[root@hakase-centos8 sysconfig]# sudo grub2-mkconfig -o /boot/grub2/grub.cfg
Generating grub configuration file ...
done
[root@hakase-centos8 sysconfig]#
```

- Edit Network Script

Next, we need to rename both network interfaces on the configuration located at the '/etc/sysconfig/network-scripts/' directory.

Go to the '/etc/sysconfig/network-scripts/'.

```
cd /etc/sysconfig/network-scripts/
```

Rename both interface name to 'ifcfg-ethX'.

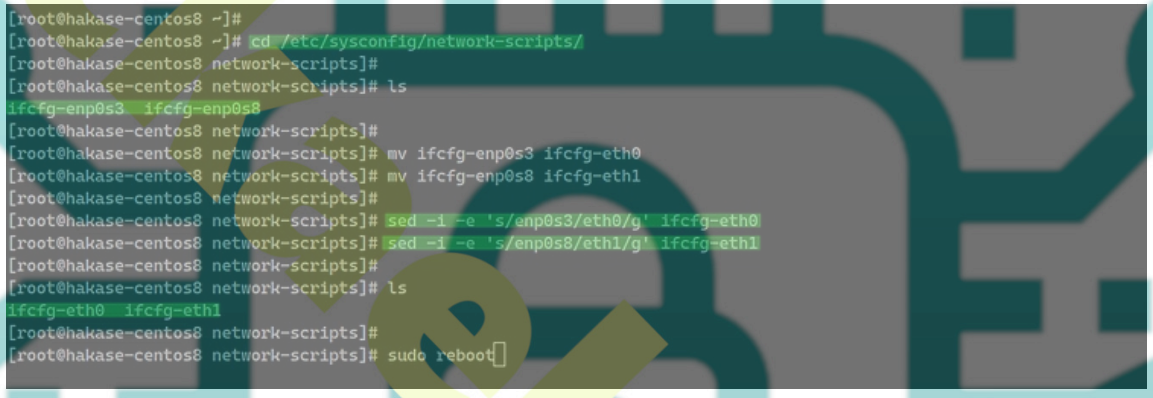
```
mv ifcfg-enp0s3 ifcfg-eth0
mv ifcfg-enp0s8 ifcfg-eth1
```

After that, change both interfaces to 'ethX' on both files using the sed command below.

```
sed -i -e 's/enp0s3/eth0/g' ifcfg-eth0
sed -i -e 's/enp0s8/eth1/g' ifcfg-eth1
```

Now restart the CentOS 8 system.

```
sudo reboot
```

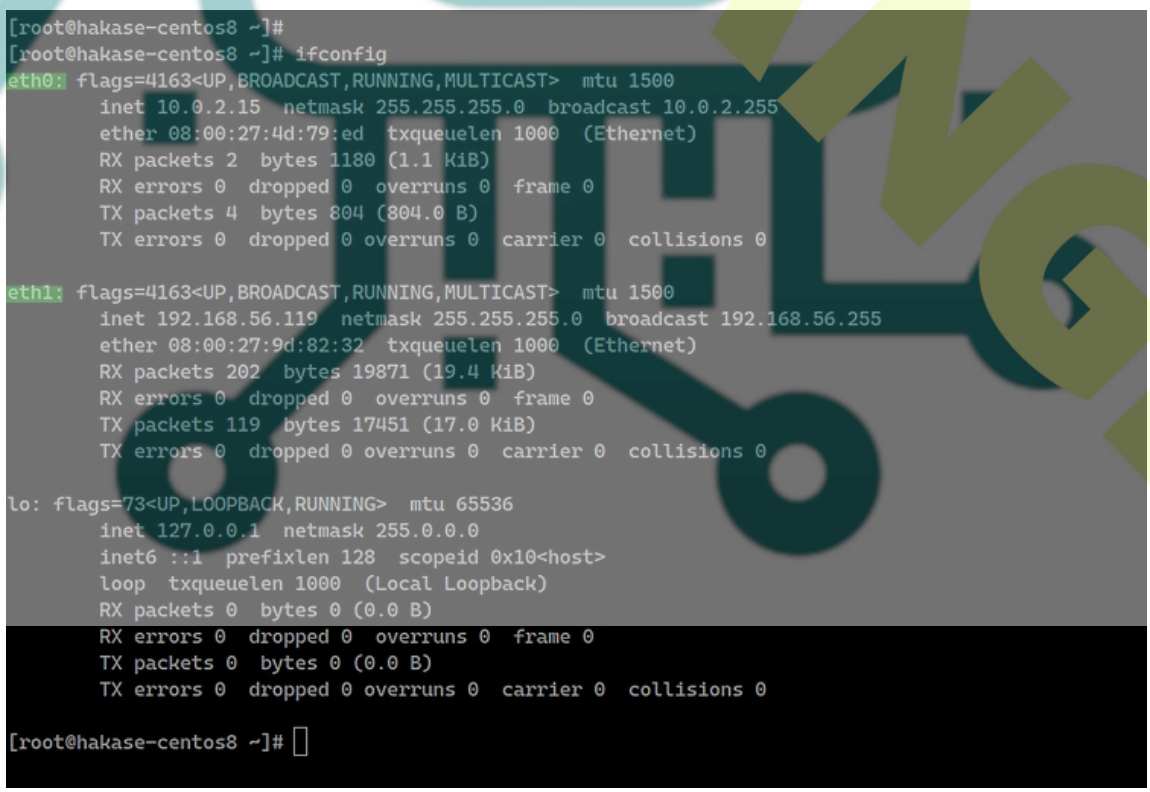


- Testing

Once you've logged in again, check available network interfaces using the following command.

```
ifconfig
ip addr
```

And you will be shown the result as below.



Both network interfaces on the CentOS 8 system have been changed and now are using the traditional naming

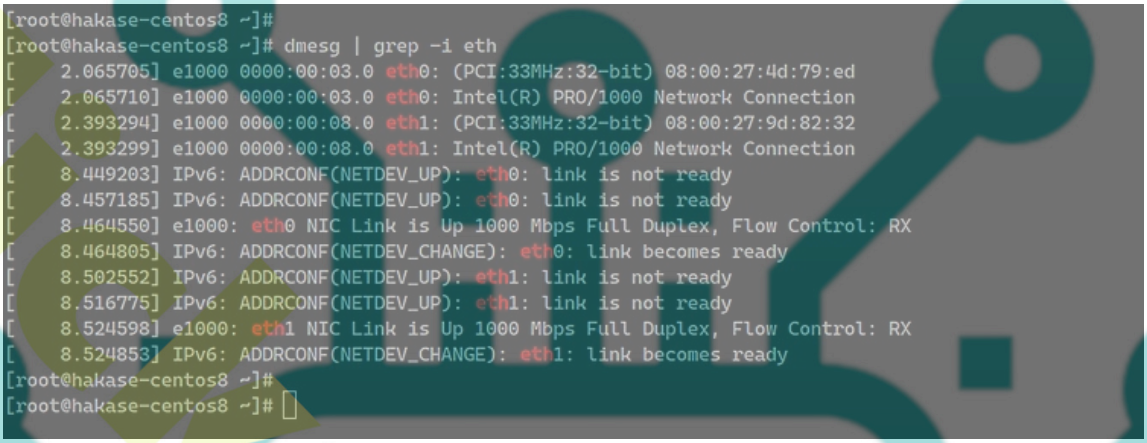
convention. The network interfaces changed by disabling the 'biosdevname' during system boot.

- Optionally

If you want to check details about the changes in the system boot, you can run the following command.

```
dmesg | grep -i eth
```

You will see the default interface name has been change to traditional name 'ethX', changed through the custom udev rules.



2. Change Interface Name to eth0 using the Udev Rules

Another way to change the network interface name to traditional naming such as eth0 etc is by creating the custom udev rules. And by using the custom udev rules, the network interfaces will be changed after the system.

- Create Custom Udev Rules

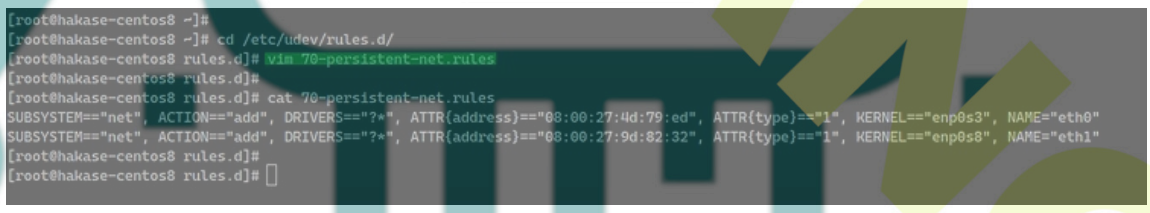
Now go to the '/etc/udev/rules.d/' directory and create the custom rule named '70-persistent-net.rules' using vim editor.

```
cd /etc/udev/rules.d/
vim 70-persistent-net.rules
```

Change the mac address and the default interface 'ens3' and 'ens8' with your own and paste into it.

```
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="52:54:00:6c:a8:e6", ATTR{type}=="1", KERNEL=="ens3", NAME="eth0"
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="52:54:00:97:cf:33", ATTR{type}=="1", KERNEL=="ens8", NAME="eth1"
```

Save and close.



- Edit Network Script

Next, go to the '/etc/sysconfig/networ-scripts' directory.

```
cd /etc/sysconfig/network-scripts/
```

Rename both interface name to 'ifcfg-ethX'.

```
mv ifcfg-enp0s3 ifcfg-eth0
mv ifcfg-enp0s8 ifcfg-eth1
```

After that, change both interfaces to 'ethX' on both files using the sed command below.

```
sed -i -e 's/enp0s3/eth0/g' ifcfg-eth0
sed -i -e 's/enp0s8/eth1/g' ifcfg-eth1
```

Now reboot the CentOS 8 server.

```
sudo reboot
```

```
[root@hakase-centos8 ~]#
[root@hakase-centos8 ~]#
[root@hakase-centos8 ~]# cd /etc/sysconfig/network-scripts/
[root@hakase-centos8 network-scripts]#
[root@hakase-centos8 network-scripts]# mv ifcfg-enp0s3 ifcfg-eth0
[root@hakase-centos8 network-scripts]# mv ifcfg-enp0s8 ifcfg-eth1
[root@hakase-centos8 network-scripts]#
[root@hakase-centos8 network-scripts]# sed -i -e 's/enp0s3/eth0/g' ifcfg-eth0
[root@hakase-centos8 network-scripts]# sed -i -e 's/enp0s8/eth1/g' ifcfg-eth1
[root@hakase-centos8 network-scripts]#
[root@hakase-centos8 network-scripts]# ls
ifcfg-eth0  ifcfg-eth1
[root@hakase-centos8 network-scripts]#
[root@hakase-centos8 network-scripts]# sudo reboot
```

- Testing

Once you've logged in again, check all available network interfaces using the following commands.

```
ifconfig
ip addr
```

And you will get the new interface 'ethX' that has been assigned.

```
[root@hakase-centos8 ~]#
[root@hakase-centos8 ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        ether 08:00:27:4d:79:ed  txqueuelen 1000  (Ethernet)
        RX packets 1  bytes 590 (590.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3  bytes 462 (462.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.119  netmask 255.255.255.0  broadcast 192.168.56.255
        ether 08:00:27:9d:82:32  txqueuelen 1000  (Ethernet)
        RX packets 169  bytes 16671 (16.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 98  bytes 13183 (12.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@hakase-centos8 ~]#
```

- Optionally

If you want to check details about the changes in the system boot, you can run the following command.

```
dmesg | grep -i eth
```

You will see the 'ensX' name is changed to the traditional name 'ethX', changed through the custom udev rules.

```
[root@hakase-centos8 ~]#
[root@hakase-centos8 ~]# dmesg | grep -i eth
[    2.033474] e1000 0000:00:03.0 eth0: (PCI:33MHz:32-bit) 08:00:27:4d:79:ed
[    2.033479] e1000 0000:00:03.0 eth0: Intel(R) PRO/1000 Network Connection
[    2.359558] e1000 0000:00:08.0 eth1: (PCI:33MHz:32-bit) 08:00:27:9d:82:32
[    2.359563] e1000 0000:00:08.0 eth1: Intel(R) PRO/1000 Network Connection
[    2.362013] e1000 0000:00:08.0 enp0s8: renamed from eth1
[    2.363559] e1000 0000:00:03.0 enp0s3: renamed from eth0
[    4.827930] e1000 0000:00:08.0 eth1: renamed from enp0s8
[    4.921792] e1000 0000:00:03.0 eth0: renamed from enp0s3
[    8.077814] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
[    8.085759] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
[    8.090408] e1000: eth0 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: RX
[    8.090664] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
[    8.131927] IPv6: ADDRCONF(NETDEV_UP): eth1: link is not ready
[    8.137121] IPv6: ADDRCONF(NETDEV_UP): eth1: link is not ready
[    8.140426] e1000: eth1 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: RX
[    8.140680] IPv6: ADDRCONF(NETDEV_CHANGE): eth1: link becomes ready
[root@hakase-centos8 ~]#
[root@hakase-centos8 ~]#
```

Reference

- https://access.redhat.com/documentation/
