Renaming Network Interfaces With udev (Can I get my eth0, eth1, ... back again)

Ubuntu

I normally use Ubuntu (currently 22.04 LTS), and networking is an area where different distros use significanly different methods to configure things. So this presentation has the specifics for current Ubuntu versions – in other distros, the configuration files may be in different places, or even work slightly differently.

History

In the beginning, there was eth0, soon followed by its sibling, eth1. And every boot, they were the same ports. Blessed be the kernel, that arranged things like this. And for a long time, the network interfaces worked, and their humans were happy as they understood which port was which, and they were always called eth0, eth1, ...

But the humans wanted more – they provided the kernel with multi-core CPUs! And for a while, things still worked just the same. But the humans were not satisfied – their new multi-core CPUs were only using one core at boot time! So they commanded the kernel to use all the cores, even at boot time. But there were strange happenings! The ethernet ports often switched around, and then there was no Internet!

After much scratching of heads and deep thinking, the humans understood – the kernel was running its drivers in parallel, and the cores were racing each other to finish initialising the hardware! Which port finished first, got the first name and was crowned eth0!

This was a disaster, so the humans quickly put udev to work and gave the Ethernet ports specifc names, and the networking worked again!

The Problem

Modern Linux distros now use udev to name their Ethernet interfaces using “Predictable Naming”, so they get names like enp5s0 or enp7s1, created from the PCI addresses. These names are hard to remember, and different on each PC. If, like me, you frequently use the interface names on multiple different systems, you find they are longer to type and you have to look them up (and maybe copy and paste them) each time you use them. And, despite assurances to the contrary, they can change occasionally if you change the hardware. They **will** change if you upgrade the motherboard. The old eth<x> names were much easier to use – the main interface on each PC was always eth0.

The Reason

To disable Predictable Naming, you can just add “net.ifnames=0” to the kernel command line by adding:

```
GRUB_CMDLINE_LINUX=”net.ifnames=0”
```

in */etc/default/grub*, and run *update-grub*. But then you get the original problem that Predictable Naming was invented to fix – if you have more than one Ethernet interface, on each boot their drivers race each other and the first one gets to be eth0. So the names of the interfaces will swap around from one boot to the next. However, if you only have one Ethernet interface, the net.ifnames=0 option works well.

The Fix - udev

With multiple Ethernet interfaces, the only way to ensure that the interfaces always get the same name is to use udev to rename them in a way that always matches the same interface with the same name. This is what the Predictable Naming udev rules do automatically. But the names used are awful. So using your own udev rules to get the names you want is the best option. There are also other options, such as netplan, which will automatically generate udev rules for renaming interfaces.

How udev Works

Udev is a daemon that is run early in the boot process. It receives messages telling it about the hardware that has been discovered, and uses its rules to tell it what actions to take (if any) for each of those messages. In Ubuntu, udev is systemd-udevd, an integral part of systemd (its source code is in the systemd source package). Udev handles messages about all sorts of hardware, and also software events. Today, I am only talking about the networking events, and specifically the Ethernet interface events.

Ethernet device events

When an Ethernet device is detected by the kernel, the matching device driver gets started, and if it is able to attach to the Ethernet device, it will generate an “add” event to say a new Ethernet device has been added. There are also “remove” and “change” events that can be generated, but the “add” event is where the name for an Ethernet interface gets assigned. To manually assign a name on an “add” event, you use udev rules like this:

```
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*",
ATTR{address}=="3c:7c:3f:1f:c0:5c",  NAME="lan0"

SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*",
ATTR{address}=="3c:7c:3f:1f:c0:5b",  NAME="lan1"
```

udev Rule Files

Such rules are put in files under */etc/udev/rules.d*, which is where user created rules go. The rules installed from packages are found in */lib/udev/rules.d*, and the run-time created rules (such as the ones generated by netplan) are put in */run/udev/rules.d*. Where a file has the same name in these different directories, the user created file is used and any others are ignored, so you can override package provided rules by using the same file name. The rule files in all the directories are merged and are used from the merged list in collating sequence order. So rule files normally are named with a two digit prefix to provide the correct order. Lower numbers will be matched to event messages before higher numbers.

Breakdown Of An Ethernet Rename Rule

Udev rules have two parts. The first part is used to match event messages, and has clauses that use the == or other matching operators. The second part is the actions, which use the = operator. So for an Ethernet renaming rule, the SUBSYSTEM==, ACTION==, DRIVERS== and ATTR{}== clauses are for matching, and the NAME= clause is an action.

SUBSYSTEM=="net" matches the network events.

ACTION=="add" matches and “add” event.

DRIVERS=="?"* matches any driver.

ATTR{address}=="3c:7c:3f:1f:c0:5c" matches that MAC address.

Netplan

In recent Ubuntu versions, you can also use netplan to generate udev rules. On my g7 laptop, I have the file */etc/netplan/01-networkmanager-all.yaml* with this in it:

```
network:
  version: 2
  renderer: NetworkManager
  ethernets:
    enp8s0:
      dhcp4: true
      dhcp6: true
      match:
        macaddress: a4:bb:6d:88:5b:60
      set-name: eth0
```

The netplan generator then creates (among other things), a */run/udev/rules.d/99-netplan-enp8s0.rules* file containing this:

```
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*",
ATTR{address}=="a4:bb:6d:88:5b:60", NAME="eth0"
```

Debugging udev Rules

To see all the debug output for the udev net rules, create a file */etc/udev/rules.d/00-debug-net.rules*, containing this line:

```
SUBSYSTEM=="net", OPTIONS="log_level=debug"
```

then you will see debug output in the udev journal:

```
journalctl -eu systemd-udevd
```

and also in */var/log/syslog*.

To turn off the debug output, delete that file or add a # at the start of the line to turn it into a comment.

The Problem With The eth<x> Names

The code that udev calls to action a NAME= clause for network devices will generate an error if the name is invalid. One reason for it being invalid is if that name is already in use. This is the reason that it is not possible to rename multiple Ethernet interfaces to eth<x> names successfully. The kernel (or maybe even the BIOS) automatically gives an eth<x> name to each Ethernet interface as it is created. So if there are two interfaces, the names eth0 and eth1 will be already in use when a udev rule attempts to change an Ethernet device to either eth0 or eth1, and the rename will fail. However, if the rename just happens to match the existing name, it will succeed.

So if your two Ethernet interfaces just happen to have been correctly named to eth0 and eth1 by the kernel, your udev rules renaming them to those names will work. But if they are reversed on any particular boot, the renames will fail. So with multiple Ethernet interfaces, you can not rename to eth<x> names. If the kernel code that applies the eth<x> names automatically was able to be optioned off, or was removed, it would be possible to rename to eth<x>. But until then, you have to use different names. I use lan<x>, as those are the names used by my first network capable operating system, OS/2. So I can easily remember to use lan<x> instead of eth<x>.

When You Have Changed The Interface Names

If you have succeeded in changing your network interface names, remember that there is configuration attached to those names, so you will need to update those config files. For NetworkManager, look for files under */etc/NetworkManager/system-connections* and replace all the old interface names in the files with the new ones. Look also at */etc/netplan* and */etc/network/interfaces* if you have used other ways to configure your network settings.

Summary

- If you have only one Ethernet interface, there are two options to get it named as eth0: use net.ifnames=0, or a udev rename rule.
- If you have multiple Ethernet interfaces, use some other name such as lan<x>.