systemd

Predictable Network Interface Names

Starting with v197 systemd/udev will automatically assign predictable, stable network interface names for all local Ethernet, WLAN and WWAN interfaces. This is a departure from the traditional interface naming scheme (`eth0` , `eth1` , `wlan0` , …), but should fix real problems.

Why?

The classic naming scheme for network interfaces applied by the kernel is to simply assign names beginning with `eth0` , `eth1` , … to all interfaces as they are probed by the drivers. As the driver probing is generally not predictable for modern technology this means that as soon as multiple network interfaces are available the assignment of the names `eth0` , `eth1` and so on is generally not fixed anymore and it might very well happen that `eth0` on one boot ends up being `eth1` on the next. This can have serious security implications, for example in firewall rules which are coded for certain naming schemes, and which are hence very sensitive to unpredictable changing names.

To fix this problem multiple solutions have been proposed and implemented. For a longer time udev shipped support for assigning permanent `ethX` names to certain interfaces based on their MAC addresses. This turned out to have a multitude of problems, among them: this required a writable root directory which is generally not available; the statelessness of the system is lost as booting an OS image on a system will result in changed configuration of the image; on many systems MAC addresses are not actually fixed, such as on a lot of embedded hardware and particularly on all kinds of virtualization solutions. The biggest of all however is that the userspace components trying to assign the interface name raced against the kernel assigning new names from the same `ethX` namespace, a race condition with all kinds of weird effects, among them that assignment of names sometimes failed. As a result support for this has been removed from systemd/udev a while back.

Another solution that has been implemented is `biosdevname` which tries to find fixed slot topology information in certain firmware interfaces and uses them to assign fixed names to interfaces which incorporate their physical location on the mainboard. In a way this naming scheme is similar to what is already done natively in udev for various device nodes via `/dev/*/by-path/` symlinks. In many cases, biosdevname departs from the low-level kernel device identification schemes that udev generally uses for these symlinks, and instead invents its own enumeration schemes.

Finally, many distributions support renaming interfaces to user-chosen names (think: `internet0`, `dmz0`, …) keyed off their MAC addresses or physical locations as part of their networking scripts. This is a very good choice but does have the problem that it implies that the user is willing and capable of choosing and assigning these names.

We believe it is a good default choice to generalize the scheme pioneered by `biosdevname`. Assigning fixed names based on firmware/topology/location information has the big advantage that the names are fully automatic, fully predictable, that they stay fixed even if hardware is added or removed (i.e. no reenumeration takes place) and that broken hardware can be replaced seamlessly. That said, they admittedly are sometimes harder to read than the `eth0` or `wlan0` everybody is used to. Example: `enp5s0`

What precisely has changed in v197?

With systemd 197 we have added native support for a number of different naming policies into systemd/udevd proper and made a scheme similar to biosdevname's (but generally more powerful, and closer to kernel-internal device identification schemes) the default. The following different naming schemes for network interfaces are now supported by udev natively:

1. Names incorporating Firmware/BIOS provided index numbers for on-board devices (example: `eno1`)
2. Names incorporating Firmware/BIOS provided PCI Express hotplug slot index numbers (example: `ens1`)
3. Names incorporating physical/geographical location of the connector of the hardware (example: `enp2s0`)
4. Names incorporating the interfaces's MAC address (example: `enx78e7d1ea46da`)
5. Classic, unpredictable kernel-native ethX naming (example: `eth0`)

By default, systemd v197 will now name interfaces following policy 1) if that information from the firmware is applicable and available, falling back to 2) if that information from the firmware is applicable and available, falling back to 3) if applicable, falling back to 5) in all other cases. Policy 4) is not used by default, but is available if the user chooses so.

This combined policy is only applied as last resort. That means, if the system has biosdevname installed, it will take precedence. If the user has added udev rules which change the name of the kernel devices these will take precedence too. Also, any distribution specific naming schemes generally take precedence.

Come again, what good does this do?

With this new scheme you now get:

- Stable interface names across reboots
- Stable interface names even when hardware is added or removed, i.e. no re-enumeration takes place (to the level the firmware permits this)
- Stable interface names when kernels or drivers are updated/changed
- Stable interface names even if you have to replace broken ethernet cards by new ones
- The names are automatically determined without user configuration, they just work
- The interface names are fully predictable, i.e. just by looking at lspci you can figure out what the interface is going to be called
- Fully stateless operation, changing the hardware configuration will not result in changes in `/etc`
- Compatibility with read-only root
- The network interface naming now follows more closely the scheme used for aliasing block device nodes and other device nodes in `/dev` via symlinks
- Applicability to both x86 and non-x86 machines
- The same on all distributions that adopted systemd/udev
- It's easy to opt out of the scheme (see below)

Does this have any drawbacks? Yes, it does. Previously it was practically guaranteed that hosts equipped with a single ethernet card only had a single `eth0` interface. With this new scheme in place, an administrator now has to check first what the local interface name is before they can invoke commands on it, where previously they had a good chance that `eth0` was the right name.

I don't like this, how do I disable this?

You basically have three options:

1. You disable the assignment of fixed names, so that the unpredictable kernel names are used again. For this, simply mask udev's .link file for the default policy: `ln -s /dev/null /etc/systemd/network/99-default.link`
2. You create your own manual naming scheme, for example by naming your interfaces `internet0`, `dmz0` or `lan0`. For that create your own `.link` files in `/etc/systemd/network/`, that choose an explicit name or a better naming scheme for one, some, or all of your interfaces. See **systemd.link(5)** for more information.
3. You pass the `net.ifnames=0` on the kernel command line

How does the new naming scheme look like, precisely?

That's documented in detail the **systemd.net-naming-scheme(7)** man page. Please refer to this in case you are wondering how to decode the new interface names.

Website source