sysctl.d

- Description
- Configuration Format
- Examples
- See Also

systemd

sysctl.d

5

sysctl.d

Configure kernel parameters at boot

`/etc/sysctl.d/*.conf`, `/run/sysctl.d/*.conf`, `/usr/local/lib/sysctl.d/*.conf`, `/usr/lib/sysctl.d/*.conf`

```
key.name.under.proc.sys = some value
key/name/under/proc/sys = some value
key/middle.part.with.dots/foo = 123
key.middle/part/with/dots.foo = 123
-key.that.will.not.fail = value
key.pattern.*.with.glob = whatever
-key.pattern.excluded.with.glob
key.pattern.overridden.with.glob = custom
```

Description

At boot, `systemd-sysctl.service(8)` reads configuration files from the above directories to configure `sysctl(8)` kernel parameters.

Configuration Format

The configuration files contain a list of variable assignments, separated by newlines. Empty lines and lines whose first non-whitespace character is `#` or `;` are ignored.

Note that either `/` or `.` may be used as separators within sysctl variable names. If the first separator is a slash, remaining slashes and dots are left intact. If the first separator is a dot, dots and slashes are interchanged. `kernel.domainname=foo` and `kernel/domainname=foo` are equivalent and will cause `foo` to be written to `/proc/sys/kernel/domainname`. Either `net.ipv4.conf.enp3s0/200.forwarding` or `net/ipv4/conf/enp3s0.200/forwarding` may be used to refer to `/proc/sys/net/ipv4/conf/enp3s0.200/forwarding`. A glob `glob(7)` pattern may be used to write the same value to all matching keys. Keys for which an explicit pattern exists will be excluded from any glob matching. In addition, a key may be explicitly excluded from being set by any matching glob patterns by specifying the key name prefixed with a `-` character and not followed by `=`, see SYNOPSIS.

Any access permission errors and attempts to write variables not present on the local system are logged at debug level and do not cause the service to fail. Other types of errors when setting variables are logged with higher priority and cause the service to return failure at the end (after processing other variables). As an exception, if a variable assignment is prefixed with a single `-` character, failure to set the variable for any reason will be logged at debug level and will not cause the service to fail.

The settings configured with `sysctl.d` files will be applied early on boot. The network interface-specific options will also be applied individually for each network interface as it shows up in the system. (More specifically, `net.ipv4.conf.*`, `net.ipv6.conf.*`, `net.ipv4.neigh.*` and `net.ipv6.neigh.*`).

Many sysctl parameters only become available when certain kernel modules are loaded. Modules are usually loaded on demand, e.g. when certain hardware is plugged in or network brought up. This means that `systemd-sysctl.service(8)` which runs during early boot will not configure such parameters if they become available after it has run. To set such parameters, it is recommended to add an `udev(7)` rule to set those parameters when they become available. Alternatively, a slightly simpler and less efficient option is to add the module to `modules-load.d(5)`, causing it to be loaded statically before sysctl settings are applied (see example below).

Examples

Set kernel YP domain name

`/etc/sysctl.d/domain-name.conf`:

```
kernel.domainname=example.com
```

Apply settings available only when a certain module is loaded (method one)

`/etc/udev/rules.d/99-bridge.rules`:

```
ACTION=="add", SUBSYSTEM=="module", KERNEL=="br_netfilter", \
      RUN+="/usr/lib/systemd/systemd-sysctl --prefix=/net/bridge"
```

`/etc/sysctl.d/bridge.conf`:

```
net.bridge.bridge-nf-call-ip6tables = 0
net.bridge.bridge-nf-call-iptables = 0
net.bridge.bridge-nf-call-arptables = 0
```

This method applies settings when the module is loaded. Please note that, unless the `br_netfilter` module is loaded, bridged packets will not be filtered by Netfilter (starting with kernel 3.18), so simply not loading the module is sufficient to avoid filtering.

Apply settings available only when a certain module is loaded (method two)

`/etc/modules-load.d/bridge.conf`:

```
br_netfilter
```

`/etc/sysctl.d/bridge.conf`:

```
net.bridge.bridge-nf-call-ip6tables = 0
net.bridge.bridge-nf-call-iptables = 0
net.bridge.bridge-nf-call-arptables = 0
```

This method forces the module to be always loaded. Please note that, unless the `br_netfilter` module is loaded, bridged packets will not be filtered by Netfilter (starting with kernel 3.18), so simply not loading the module is sufficient to avoid filtering.

Set network routing properties for all interfaces

`/etc/sysctl.d/20-rp_filter.conf`:

```
net.ipv4.conf.default.rp_filter = 2
net.ipv4.conf.*.rp_filter = 2
-net.ipv4.conf.all.rp_filter
net.ipv4.conf.hub0.rp_filter = 1
```

The `rp_filter` key will be set to "2" for all interfaces, except "hub0". We set `net.ipv4.conf.default.rp_filter` first, so any interfaces which are added *later* will get this value (this also covers any interfaces detected while we're running). The glob matches any interfaces which were detected *earlier*. The glob will also match `net.ipv4.conf.all.rp_filter`, which we do not want to set at all, so it is explicitly excluded. And "hub0" is excluded from the glob because it has an explicit setting.

See Also

`systemd(1)`, `systemd-sysctl.service(8)`, `systemd-delta(1)`, `sysctl(8)`, `sysctl.conf(5)`, `modprobe(8)`

□□□ #2
□ □□□ □□□ 2026-10-31 14:58:47 UTC
□ □□□ □□□ 2026-10-31 14:58:48 UTC