Full credit is given to the above companies including the OS that this PDF file was generated!

Rocky Enterprise Linux 9.2 Manual Pages on command 'modprobe.conf.5'

$ man modprobe.conf.5

MODPROBE.D(5)   modprobe.d   MODPROBE.D(5)

NAME

　　modprobe.d - Configuration directory for modprobe

SYNOPSIS

　　/lib/modprobe.d/*.conf

　　/etc/modprobe.d/*.conf

　　/run/modprobe.d/*.conf

DESCRIPTION

　　Because the modprobe command can add or remove more than one module,

　　due to modules having dependencies, we need a method of specifying what

　　options are to be used with those modules. All files underneath the

　　/etc/modprobe.d directory which end with the .conf extension specify

　　those options as required. They can also be used to create convenient

　　aliases: alternate names for a module, or they can override the normal

　　modprobe behavior altogether for those with special requirements (such

　　as inserting more than one module).

　　Note that module and alias names (like other module names) can have -

　　or _ in them: both are interchangeable throughout all the module

commands as underscore conversion happens automatically.

The format of files under modprobe.d is simple: one command per line,
with blank lines and lines starting with '#' ignored (useful for adding
comments). A '\' at the end of a line causes it to continue on the next
line, which makes the file a bit neater.

COMMANDS

alias wildcard modulename

This allows you to give alternate names for a module. For example:
"alias my-mod really_long_modulename" means you can use "modprobe
my-mod" instead of "modprobe really_long_modulename". You can also
use shell-style wildcards, so "alias my-mod*
really_long_modulename" means that "modprobe my-mod-something" has
the same effect. You can't have aliases to other aliases (that way
lies madness), but aliases can have options, which will be added to
any other options.

Note that modules can also contain their own aliases, which you can
see using modinfo. These aliases are used as a last resort (ie. if
there is no real module, install, remove, or alias command in the
configuration).

blacklist modulename

Modules can contain their own aliases: usually these are aliases
describing the devices they support, such as "pci:123...". These
"internal" aliases can be overridden by normal "alias" keywords,
but there are cases where two or more modules both support the same
devices, or a module invalidly claims to support a device that it
does not: the blacklist keyword indicates that all of that
particular module's internal aliases are to be ignored.

install modulename command...

This command instructs modprobe to run your command instead of
inserting the module in the kernel as normal. The command can be
any shell command: this allows you to do any kind of complex
processing you might wish. For example, if the module "fred" works
better with the module "barney" already installed (but it doesn't

depend on it, so modprobe won't automatically load it), you could say "install fred /sbin/modprobe barney; /sbin/modprobe --ignore-install fred", which would do what you wanted. Note the --ignore-install, which stops the second modprobe from running the same install command again. See also remove below.

The long term future of this command as a solution to the problem of providing additional module dependencies is not assured and it is intended to replace this command with a warning about its eventual removal or deprecation at some point in a future release. Its use complicates the automated determination of module dependencies by distribution utilities, such as mkinitrd (because these now need to somehow interpret what the install commands might be doing. In a perfect world, modules would provide all dependency information without the use of this command and work is underway to implement soft dependency support within the Linux kernel.

If you use the string "$CMDLINE_OPTS" in the command, it will be replaced by any options specified on the modprobe command line. This can be useful because users expect "modprobe fred opt=1" to pass the "opt=1" arg to the module, even if there's an install command in the configuration file. So our above example becomes "install fred /sbin/modprobe barney; /sbin/modprobe --ignore-install fred $CMDLINE_OPTS"

options modulename option...

This command allows you to add options to the module modulename (which might be an alias) every time it is inserted into the kernel: whether directly (using modprobe modulename) or because the module being inserted depends on this module.

All options are added together: they can come from an option for the module itself, for an alias, and on the command line.

remove modulename command...

This is similar to the install command above, except it is invoked when "modprobe -r" is run.

softdep modulename pre: modules... post: modules...

The softdep command allows you to specify soft, or optional, module dependencies. modulename can be used without these optional modules installed, but usually with some features missing. For example, a driver for a storage HBA might require another module be loaded in order to use management features.

pre-deps and post-deps modules are lists of names and/or aliases of other modules that modprobe will attempt to install (or remove) in order before and after the main module given in the modulename argument.

Example: Assume "softdep c pre: a b post: d e" is provided in the configuration. Running "modprobe c" is now equivalent to "modprobe a b c d e" without the softdep. Flags such as --use-blacklist are applied to all the specified modules, while module parameters only apply to module c.

Note: if there are install or remove commands with the same modulename argument, softdep takes precedence.

COMPATIBILITY

A future version of kmod will come with a strong warning to avoid use of the install as explained above. This will happen once support for soft dependencies in the kernel is complete. That support will complement the existing softdep support within this utility by providing such dependencies directly within the modules.

COPYRIGHT

This manual page originally Copyright 2004, Rusty Russell, IBM Corporation. Maintained by Jon Masters and others.

SEE ALSO

modprobe(8), modules.dep(5)

AUTHORS

Jon Masters <jcm@jonmasters.org>

Developer

Robby Workman <rworkman@slackware.com>

Developer

Lucas De Marchi <lucas.de.marchi@gmail.com>

kmod   12/27/2020   MODPROBE.D(5)