Managing the Kernel - Loadable Kernel Modules

Linux typically modules have the extension .ko ("kernel object") since version 2.6 (previous versions used the .o extension). Other OS's - kernel loadable module (kld) in FreeBSD; kernel extension (kext) in macOS; kernel extension module in AIX; kernel-mode driver in Windows NT. Linux modules are generally in subdirectories of **/lib/modules/** named for kernel version. Numbering format same as Linux versioning- major.minor.patch Odd numbers for minor version are developmental. /lib/modules/*.ko files i.e, bridge.ko for network support

After **/etc/initab** specifies the default runlevel, it kicks off **/etc/rc.sysinit** to load modules
/etc/modprobe.conf -- associates/aliases drivers to devices such as eth0
"alias eth0 natsemi" says use natsemi driver/module for eth0
/etc/modprobe.d/modprobe.conf.dist -- large set of standard autoloaded aliases
/etc/modprobe.d/blacklist and **blacklist-compat** -- aliases that are not loaded

modprobe <modulename> - loads/adds modules AND auto-adds their dependencies
-v verbose; -r remove; -a add; -l list all modules; -t [dir] list modules in directory
insmod - to insert 3rd party drivers/modules; common error: dependency error: "unresolved symbol…." inserts only specified module -not it's dependencies. **modprobe** is preferable
-e make persistent; -f force; -L prevent simultaneous loading of the same module; -o specify optional module name
modinfo <modulename> - param fields have variables and such used in the system calls. -V version, -n name -a author -d description -p parameters
depmod - is run at startup right before **modprobe** to give it dependency info- creates **/lib/modules/modules.dep**
lsmod - lists loaded modules. Indicates size, number of dependencies and what it is used by
modprobe -l - will do sort of the same, with dependencies
cat /proc/modules/ - all modules currently loaded- subdirectories for /pcmcia, /net, /arch, /fs, /drivers
/sys/module also has sub-directories that contains information about each kernel module installed

rmmod remove modules from the kernel but it does not check for dependencies
modprobe -r <modulename> - removes a module from the kernel after checking for dependencies

udev is a device manager that manages the automatic detection and kernel module loading for both coldplug and hotplug devices; in charge of the **/dev** virtual file system to dynamically creates device files as devices are added and removed. When providing new hardware like a USB key, udev wakes up, intitializes the new HW with kernel so kernel can load proper modules automatically.

udevadm monitor
Upon plugging in a usb stick, devices and bus messages (truncated below), see module listed for fat and vfat

```
[root@rhelserver ~]# udevadm monitor
monitor will print the received events for:
UDEV - the event which udev sends out after rule processing
KERNEL - the kernel uevent

KERNEL[69484.521158] add      /devices/pci0000:00/0000:00:11.0/0000:02:03.0/usb1/1-1 (usb)
KERNEL[69484.554385] add      /devices/pci0000:00/0000:00:11.0/0000:02:03.0/usb1/1-1/1-1:1.0 (usb)
UDEV  [69484.564011] add      /devices/pci0000:00/0000:00:11.0/0000:02:03.0/usb1/1-1 (usb)
KERNEL[69484.583125] add      /module/usb_storage (module)
UDEV  [69484.608070] add      /module/usb_storage (module)
KERNEL[69484.619712] add      /devices/pci0000:00/0000:00:11.0/0000:02:03.0/usb1/1-1/1-1:1.0/host3 (scsi)
UDEV  [69484.619737] add      /devices/pci0000:00/0000:00:11.0/0000:02:03.0/usb1/1-1/1-1:1.0 (usb)
KERNEL[69484.619745] add      /devices/pci0000:00/0000:00:11.0/0000:02:03.0/usb1/1-1/1-1:1.0/host3/scsi_host/host3 (scsi_host)
KERNEL[69484.619752] add      /bus/usb/drivers/usb-storage (drivers)
UDEV  [69484.621580] add      /bus/usb/drivers/usb-storage (drivers)
UDEV  [69484.622757] add      /devices/pci0000:00/0000:00:11.0/0000:02:03.0/usb1/1-1/1-1:1.0/host3 (scsi)
UDEV  [69484.623611] add      /devices/pci0000:00/0000:00:11.0/0000:02:03.0/usb1/1-1/1-1:1.0/host3/scsi_host/host3 (scsi_host)
KERNEL[69486.685029] add      /module/fat (module)
KERNEL[69486.685064] add      /kernel/slab/fat_cache (slab)
KERNEL[69486.685076] add      /kernel/slab/fat_inode_cache (slab)
UDEV  [69486.687754] add      /module/fat (module)
UDEV  [69486.687778] add      /kernel/slab/fat_cache (slab)
UDEV  [69486.687785] add      /kernel/slab/fat_inode_cache (slab)
KERNEL[69486.689217] add      /module/vfat (module)
UDEV  [69486.690176] add      /module/vfat (module)
```

Do it manually: unplug USB, then
lsmod | grep fat - shows vfat module still loaded. Don't rely on **udev** to unload what it activated
modprobe -r vfat --removes modules and dependencies no longer needed
modprobe vfat -- loads the module and dependencies (which udev is expected to accomplish automatically)
---
udevadm controls **systemd-udevd**, requests kernel events, manages event queue, and simple debugging.
udevadm [**info** | **trigger** | **settle** | **monitor**] <options> AND **udevadm control** <command>
The **/etc/udev/rules.d/** - directory allows naming devices when they are connected

Changing Parameters of Kernel Modules (Devices)

modinfo cdrom - reports params. These can be changed, but only by unloading the module with **modprobe -r cdrom**, then **modprobe cdrom lockdoor=0** (for example, to turn off the lockdoor boolean param)

```
[root@rhelserver ~]# lsmod | grep cdrom
cdrom                  42556  1 sr_mod
[root@rhelserver ~]# modinfo cdrom
filename:       /lib/modules/3.10.0-121.el7.x86_64/kernel/drivers/cdrom/cdrom.ko
license:        GPL
srcversion:     B5F2D59440347DFFB175E71
depends:
intree:         Y
vermagic:       3.10.0-121.el7.x86_64 SMP mod_unload modversions
signer:         Red Hat Enterprise Linux kernel signing key
sig_key:        42:49:68:9E:EF:C7:7E:95:88:0B:13:DF:E4:67:EB:1B:7A:91:D1:08
sig_hashalgo:   sha256
parm:           debug:bool
parm:           autoclose:bool
parm:           autoeject:bool
parm:           lockdoor:bool
parm:           check_media_type:bool
parm:           mrw_format_restart:bool
```

It used to be modifying modprobe.conf could make changes but in RHEL7 it changed
These are default settings for kernel modules, from the associated rpm packages:
/lib/modprobe.d/dist-alsa.conf and **/lib/modprobe.d/dist-blacklist.conf**
You don't want to edit them.

Instead, edit files in **/etc/modprobe.d/** By default it is empty- it is the place to put custom conf files.
(see **man 5 modprobe.d** - "options" section*). In this directory create/ **vim cdrom.conf** and add:
options cdrom lockdoor=0

Generally you need to restart to see the effects and truly reload- reloading the module isn't enough.
For some modules you can look in **/sys/module/**, find a directory for the module, and see a file called parameters (or something), but it is up to the programmers to provide this kind of file. For cdrom it isn't. Looking in **dmesg | grep <modulename>** might help find something about when the module was initialized.

```
[root@rhelserver module]# cd cdrom/
[root@rhelserver cdrom]# ls
coresize  holders  initsize  initstate  notes  refcnt  sections  srcversion  taint  uevent
[root@rhelserver cdrom]# dmesg | grep cdrom
[    2.738573] cdrom: Uniform CD-ROM driver Revision: 3.20
[root@rhelserver cdrom]# dmesg | grep -A5 cdrom
[    2.738573] cdrom: Uniform CD-ROM driver Revision: 3.20
[    2.739174] sr 1:0:0:0: Attached scsi CD-ROM sr0
[    2.777003] usb 2-2: New USB device found, idVendor=0e0f, idProduct=0002
[    2.777006] usb 2-2: New USB device strings: Mfr=0, Product=1, SerialNumber=0
[    2.777008] usb 2-2: Product: VMware Virtual USB Hub
[    2.782794] hub 2-2:1.0: USB hub found
```

*man page refers to locations **/etc/modprobe.d/*.conf**, **/lib/modprobe.d/*.conf**, and **/run/modprobe.d/*.conf**

Fields in /etc/modprobe.d/modprobe.conf (or **/etc/modprobe.conf**)

alias {wildcard} {module name}	Specify an alternate name for a module with a long name.
include {file name}	Add configuration files to a module.
options {module name} {option}	Options to be added to each module before insertion into the kernel.
install {module name} {command}	Run the command specified without inserting the module into the kernel.

Don't confuse with shared library files! (aren't kernel modules)

The **/usr/lib** and **/lib** directories are the default system library file locations where the system libraries are kept.
Contains routines, which are used by various applications; loaded into memory when executable that links to them is loaded. They are then shared with other applications.
When added, new library file details are passed on to **/etc/ld.so.conf** (default system library info)
Running **ldconfig** updates changes in that file and loads the shared libraries from locations specified by **/etc/ld.so.cache**.

ldconfig -f <config-file>
-C <cache-file> where library updates will be stored
-v view details of library file, rebuilds cache
-p - show shared library cache
-n /<location> update the library file info in the specified location instead of the default

ldd -v <program-name> - List dynamic dependencies of executable files or shared objects.
LD_LIBRARY_PATH environment variable