

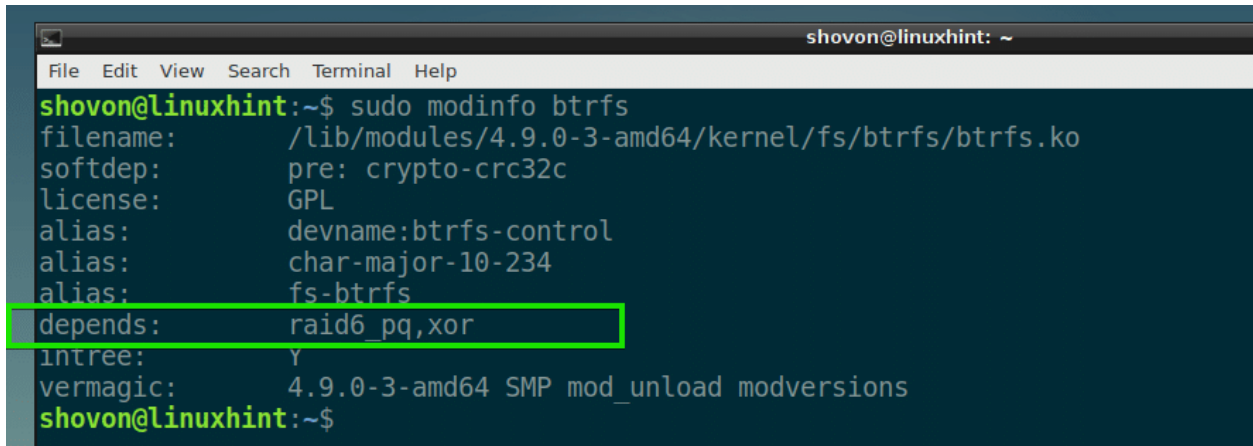
DEVICE DRIVERS PRACTISE ASSIGNMENT - 1

CS20B1057

A S V Dhanush

1. **modinfo** : modinfo command in Linux system is used to display the information about a Linux Kernel module. This command extracts the information from the Linux kernel modules given on the command line. If the module name is not a file name, then the /lib/modules/kernel-version directory is searched by default. modinfo can understand modules from any of the Linux Kernel architecture.

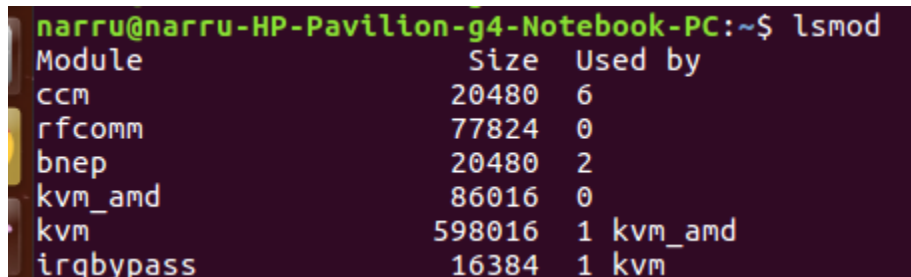
Example: modinfo bluetooth



```
shovon@linuxhint: ~  
File Edit View Search Terminal Help  
shovon@linuxhint:~$ sudo modinfo btrfs  
filename:      /lib/modules/4.9.0-3-amd64/kernel/fs/btrfs/btrfs.ko  
softdep:       pre: crypto-crc32c  
license:       GPL  
alias:         devname:btrfs-control  
alias:         char-major-10-234  
alias:         fs-btrfs  
depends:        raid6_pq,xor  
intree:        Y  
vermagic:      4.9.0-3-amd64 SMP mod_unload modversions  
shovon@linuxhint:~$
```

2. **lsmod** : lsmod command is used to display the status of modules in the Linux kernel. It results in a list of loaded modules. Lsmod is a trivial program which nicely formats the contents of the /proc/modules, showing what kernel modules are currently loaded.

Example: lsmod



```
narru@narru-HP-Pavilion-g4-Notebook-PC:~$ lsmod  
Module              Size  Used by  
ccm                  20480  6  
rfcomm               77824  0  
bnep                 20480  2  
kvm_aml              86016  0  
kvm                  598016  1 kvm_aml  
irqbypass           16384  1 kvm
```

3. **insmod** : **insmod** command in Linux systems is used to insert modules into the kernel.

LKMs(Loadable Kernel Modules) are usually used to add support for new hardware (as device drivers) and/or filesystems, or for adding system calls. This command here inserts the kernel object file (.ko) into the kernel

with/without arguments, along with a few additional options.

Example: insmod -V

```
mukkes@mex:~$ insmod -h
Usage:
  insmod [options] filename [args]
Options:
  -V, --version      show version
  -h, --help         show this help
mukkes@mex:~$
```

4. **modprobe** : command to add and remove modules from the Linux Kernel.

Example: modprobe [-c]

```
jamal ~
$ sudo modprobe module2
jamal ~
$ dmesg
[83010.207242] module1_init: In init
[83010.209745] module2_init: In init
[83010.209747] myadd: Adding 3 with 5      Result:8
[83010.209747] module2_init: Add:8
jamal ~
```

5. **kernel.h** : The linux/kernel. h header which gets used for module builds is the header which is part of the kernel source. When modules are built in the kernel source tree, that's the version which is used.

6. **device. h** : The Device Header File <device. h> contains the following sections that are device specific: Interrupt Number Definition provides interrupt numbers (IRQn) for all

exceptions and interrupts of the device. Configuration of the Processor and Core Peripherals reflect the features of the device.

7. **slab.h** : page_slab - Converts from first struct page to slab.

8. **rmmod**: rmmod command in Linux system is used to remove a module from the kernel. Most of the users still use modprobe with the -r option instead of using rmmod.

Example : rmmod -v bluetooth

```
algoscale@algoscale-Lenovo-ideapad-330-15IK8:~$ sudo rmmod -f bluetooth
rmmod: ERROR: ../libkmod/libkmod-module.c:793 kmod_module_remove_module() could not remove 'bluetooth': Resource temporarily unavailable
rmmod: ERROR: could not remove module bluetooth: Resource temporarily unavailable
```

9. **depmod** : depmod(Dependency Modules) command is used to generate a list of dependency

description of kernel modules and its associated map files. This analyzes the kernel

modules in the directory /lib/modules/kernel-release and creates a “Makefile”-like dependency file named modules.

```
depmod [ -a ] [ -b basedir ] [ -e ] [ -F System.map ] [ -n ] [ -v ] [ version ] [ -A ]
      [-n] [-v] [-A] [-P prefix] [-w] [version]

depmod [-e] [-E Module.symvers] [-F System.map] [-m] [-n] [-v] [-P prefix]
      [-w] [version] [filename...]
```

10. **init.h** : module_exit() include/linux/init. h. This macro defines the function to be called at

module removal time (or never, in the case of the file compiled into the kernel). It will only be called if the module usage count has reached zero.

11. **kdev_t.h** :Some programs want their definitions of MAJOR and MINOR and MKDEV from the kernel sources. These must be the externally visible ones.

12. **fs.h**: This file has definitions for some important file table structures

13. **uaccess.h** : Checks if a pointer to a block of memory in user space is valid

14. **__init** : The __init keyword tells the linker to place the code in a dedicated section into the kernel object file. This section is known in advance to the kernel, and freed when the module is loaded and the init function finished.

The __init function is the first most process of the kernel operating system