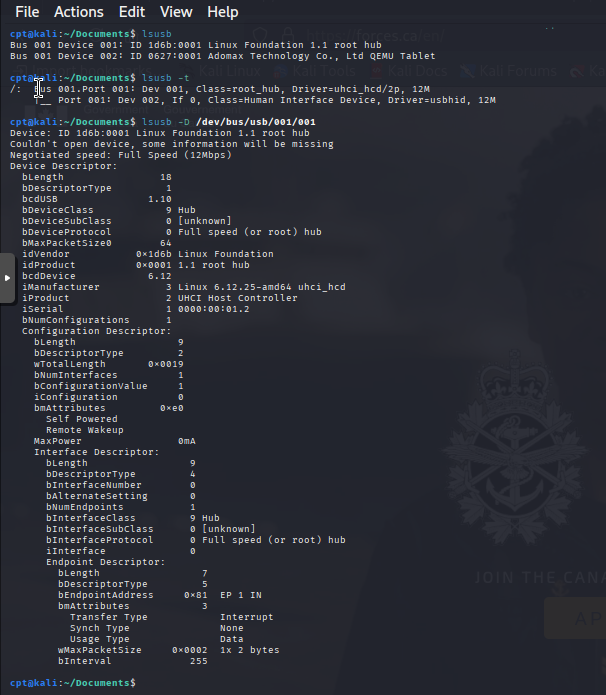
*Determine and configure hardware settings*

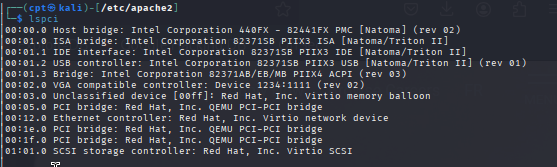
'**lsusb'**

The '**lsusb'** command in Linux is a useful utility for displaying information about USB buses and the devices connected to them. It provides a detailed view of the USB hardware connected to your system, including details such as speed, bus number, device class, and type. This command is particularly valuable for troubleshooting USB devices and understanding the hardware configuration of your Linux system.

-D (Display Specific Device File Information)



The **lspci** (list PCI) [**Linux command**](https://phoenixnap.com/kb/linux-commands) displays information about each PCI bus on your system. This includes information about the devices connected to the PCI subsystem.



**modprob/lsmod/insmod**

The Linux kernel is the core component of the Linux operating system. It manages the system’s resources and it is a bridge between your computer’s hardware and software.

The Linux kernel has a modular design. A kernel module, or often referred to as a driver, is a piece of code that extends the kernel’s functionality. Modules are either compiled as loadable modules or built into the kernel. Loadable modules can be loaded and unloaded in the running kernel on request, without the need to reboot the system.

Generally, the modules are loaded on demand by udev (device manager). However, sometimes you may need to fine-tune how the modules are loaded. For example, you may need to load a module with additional parameters or to prevent the automatic loading of a module.

lsmod is a command-line utility that displays information about the loaded Linux kernel modules.

**insmod** command in Linux systems is used to insert modules into the kernel. Linux is an Operating System that allows the user to load kernel modules on run time to extend the kernel functionalities.

Both modprobe and insmod are used to insert kernel modules in Linux, but they differ in how smart and flexible they are.

* modprobe: The Intelligent Loader

Automatically handles dependencies: If your module relies on others, modprobe loads them too.

Searches standard module paths: It looks in /lib/modules/$(uname -r)/ and uses modules.dep to resolve dependencies.

Preferred for production: It's safer and more convenient for system administrators.

* insmod: The Manual Tool

Loads a single module only: No dependency resolution.

Requires full path to the .ko file.

Used in development: Handy when testing custom modules not yet installed system-wide.

*Boot the system*

*Change runlevels*