CPSC 457: Principles of Operating Systems

Tutorial 01: TR 11:00 - 11:50 MS 119
Tutorial 02: TR 16:00 - 16:50 MS 176

February 7, 2013
Announcements

Homework 1 is due tomorrow!!!
Homework 2 is released it is due March 15th
Base Kernel Code is the part of the kernel that is compiled and built into the kernel image. This is what is booted during the system start up. LKMs are code that is added to the kernel while it is running. i.e. they are not part of the boot image. Once installed LKMs are a part of the kernel and run at the kernel level. They are NOT user level programs and therefore can easily break a system.
Why LKM\text{s}?

- Can modify kernel without the need to rebuild base kernel image.
- Reduces size of base kernel image.
- Saves memory
- Debugging and Maintenance
- No performance penalty
Why NOT LKMs?

- Not loaded until after system boots
- Risk of an important module being unloaded
What LKM Are Used For

- Device Drivers
- Filesystem Drivers
- System Calls (add new ones or override existing)
- Network Drivers
- TTY line Disciplines
- Executable Interpreters
LKM Commands

- **insmod** (insert an LKM into the kernel)
- **rmmod** (remove an LKM from the kernel)
- **depmod** (determine interdependencies)
- **kerneld** (kerneld daemon program)
- **ksyms** (display symbols exported by kernel)
- **lsmod** (list currently loaded LKMs)
- **modinfo** (display contents of .modinfo section)
- **modprobe** (insert or remove LKM intelligently)
LKM Compilation Process

- Kernel 2.4 - the loader would link the object file to the running kernel

- Kernel 2.6 and later - kernel does the linking
  - object file (.o) is created
  - modpost uses this to create C source with additional sections (.mod.c)
  - compile .mod.c to create .mod.o
  - link .mod.o with original .o to create module (.ko) file
Questions?