Operating Systems Learning Environment with VMware

Akalanka Mailewa and Jayantha Herath
Department of Computer Science
St. Cloud State University
St. Cloud, Minnesota 56301
dmakalanka@gmail.com
herath@stcloudstate.edu

Abstract

There is both local and global need for high-quality trained system programmers with the ability to learn in a short period of time and stay current with the recent advances and tools available in the cyber infrastructure. Computing programs can help to meet this demand by redesigning their operating systems courses and allowing more students to succeed at the entry level. Authors have developed a VMware based operating systems course to enhance the computing curriculum. Students download tools available in cyber infrastructure to install and use several virtual operating systems and make changes to internals. Both the public and private sectors will benefit with the greater number of highly-skilled trained system programmers. To assess and evaluate the changes made authors distributed the course modules among our students. This paper describes authors' experience in developing the project based active operating systems learning environment.

1 Introduction

We identified many deficiencies in teaching a traditional lecture based operating systems class and the importance of transforming the course into a project based learning environment. The lecture based environment is relatively passive. An effective OS class could provide links to connect other courses such as computer architecture, digital design, distributed systems, security, networking, parallel computing in the curriculum, and importantly the needs of industry. Successful application of active learning with rapidly changing tools available in the cyber infrastructure in the learning process is a way to remove the deficiencies. The objectives of this course are to introduce operating systems fundamentals using VMWARE, provide the ability to design algorithms to solve operating systems problems, implement those, test and document programs. Student outcomes at the end of the course are shown below:

1. Install two or more operating systems
2. Write programs that interact with the operating system through the use of system calls
3. Write communication and synchronization programs
4. Knowledge of tools such as *make* to support code generation
5. Develop an understanding of the challenges in modifying and building very large scale system of programs
6. Design and implement applications that apply two or more scheduling algorithms.
7. Communicate both technical and non-technical aspects of students' work in formal and informal settings.

First section of this paper discusses theoretical background of Linux Kernel, system calls, openSUSE operating system and the VMware player. Second section shows how to compile the Linux kernel, step by step. It explores several aspects of Linux kernel compilation. It shows in detail with the aids of screen shots of actual processes how to compile the latest available Linux kernel on openSUSE v12.3 operating system which is run on top of the VMware player. Finally it will discuss difficulties encountered during the process, brief conclusion and future works.

1.1 Linux kernel

The Linux kernel is the operating system kernel used by the Linux family. It is a prominent example of free and open source software [1]. The Linux kernel is released under the GNU General Public License version 2 (GPLv2) (plus some firmware images with various non-free licenses), and is developed by contributors worldwide. Day-to-day development discussions take place on the Linux kernel mailing list [1]. The Linux kernel was initially conceived and created in 1991 by Linus Torvalds. Linux rapidly accumulated developers and users who adapted code from other free software projects for use with the new operating system. The Linux kernel has received contributions from thousands of programmers. All Linux distributions released have been based upon the Linux kernel [1].

Linux is a monolithic kernel. Device drivers and kernel extensions run in kernel space (ring 0 in many CPU architectures), with full access to the hardware, although some exceptions run in user space, for example file systems based on FUSE. The graphics system most people use with Linux doesn't run in the kernel, in contrast to that found in Microsoft Windows. Unlike standard monolithic kernels, device drivers are easily configured as modules, and loaded or unloaded while running the system. Also unlike standard monolithic kernels, device drivers can be pre-empted under certain conditions. This latter feature was added to handle hardware interrupts correctly, and to improve support for symmetric multiprocessing [1].

The hardware is also incorporated into the file hierarchy. Device drivers interface to user applications via an entry in the /dev and/or /sys directories. Process information as well is mapped to the file system through the /proc directory [1].

1.2 System Call

A system call requests a service from an operating system's kernel. This may include hardware related services (e.g. accessing the hard disk), creating and executing new processes, and communicating with integral kernel services (like scheduling). System call provides an essential interface between a process and the operating system [2].

The design of the microprocessor architecture practically on all modern systems (except some embedded systems) involves a security model (such as the rings model) which specifies multiple privilege levels under which software may be executed; for instance, a program is usually limited to its own address space so that it cannot access or modify other running programs or the operating system itself, and a program is usually prevented from directly manipulating hardware devices (e.g. the frame buffer or network devices) [2].

However, many normal applications obviously need access to these components, so system calls are made available by the operating system to provide well-defined, safe implementations for such operations. The operating system executes at the highest level of privilege, and allows applications to request services via system calls, which are often executed via interrupts; an interrupt automatically puts the CPU into some required privilege level, and then passes control to the kernel, which determines whether the calling program should be granted the requested service. If the service is granted, the kernel executes a specific set of instructions over which the calling program has no direct control, returns the privilege level to that of the calling program, and then returns control to the calling program [2].

1.3 openSUSE operating system

openSUSE is a general purpose operating system built on top of the Linux kernel, developed by the community-supported openSUSE Project and sponsored by SUSE and a number of other companies. After Novell acquired SUSE Linux in January 2004, Novell decided to release the SUSE Linux Professional product as a 100% open source project. In 2011 The Attachmate Group acquired Novell and split Novell and SUSE into two autonomous subsidiary companies. SUSE offers products and services around SUSE Linux Enterprise their commercial offering that is based on openSUSE Linux [3]. The initial release of the community project was a beta version of SUSE Linux 10.0, and as of March 13, 2013 the current stable release is openSUSE 12.3 [3].

1.4 VMware Player

VMware Player is a virtualization software package supplied free of charge by VMware Inc., a company which was formerly a division of and whose majority shareholder remains EMC Corporation. VMware Player can run existing virtual appliances and create its own virtual machines (which require an operating system to be installed to be functional). It uses the same virtualization core as VMware Workstation, a similar program with more features, but not free of charge. VMware Player is available for personal non-commercial use, or for distribution or other use by written agreement. No support is provided by VMWare, but there is an active community website for discussing and resolving issues [4].

VMware claims the Player offers better graphics, faster performance, and tighter integration for running Windows XP under Windows Vista or Windows 7 than Microsoft's Windows XP Mode running on Windows Virtual PC, which is free of charge for all purposes [4].

Versions earlier than 3 of VMware Player were unable to create virtual machines (VMs), which had to be created by an application with the capability, or created manually by statements stored in a text file with extension ".vmx"; later versions can create VMs. The features of Workstation not available in Player are "developer-centric features such as Teams, multiple Snapshots and Clones, and Virtual Rights Management features for end-point security", and support by VMWare. Player allows a complete virtual machine to be copied at any time by copying a directory; while not a fully featured snapshot facility, this allows a copy of a machine in a particular state to be stored, and reverted to later if desired [4].

VMware Player is also supplied with the VMware Workstation distribution, for use in installations where not all client users are licensed to use the full VMware Workstation. In an environment where some machines without VMware Workstation licences run VMWare Player, a virtual machine created by Workstation can be distributed to computers running Player without paying for additional Workstation licenses if not used commercially [4].

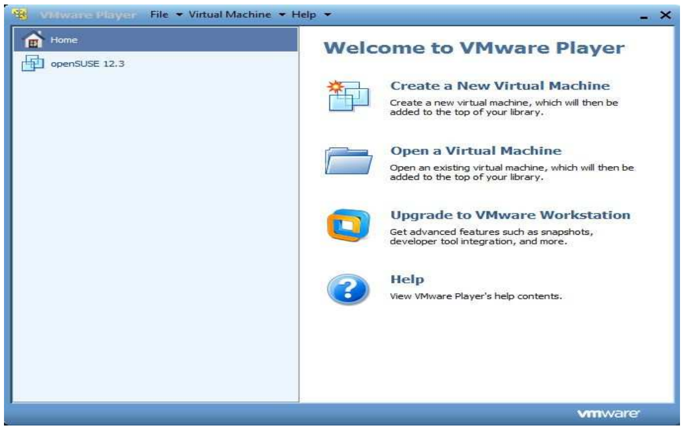
2 Compilation of the Linux Kernel

openSUSE is not Linux, it is a distribution based on the Linux kernel. The Linux kernel is a complex program which provides the underlying services to the rest of a Linux distribution. But it is easy to add new features or improvements to it as, unlike commercial operating systems like Windows or MacOS, the source code is freely available. It is common practice with a Linux based operating system to recompile the kernel from source and much effort has been put into making this a relatively user-friendly experience [5].

There are three reasons for a recompile. Firstly, you may have some hardware that is so new that there's no kernel module for it in on your distribution. Secondly, you may have come across some kind of bug which is fixed in a revision of the operating system. Lastly, you may have some new software which requires a newer version of the operating system [5].

2.1 Download and install the VMware player

Once you are done with the installation you can download openSUSE v12.3 image and load it into your VMware player by using “Open Virtual Machine” link, see Figure 1.



Create a New Virtual Machine
Create a new virtual machine, which will then be added to the top of your library.

Open a Virtual Machine
Open an existing virtual machine, which will then be added to the top of your library.

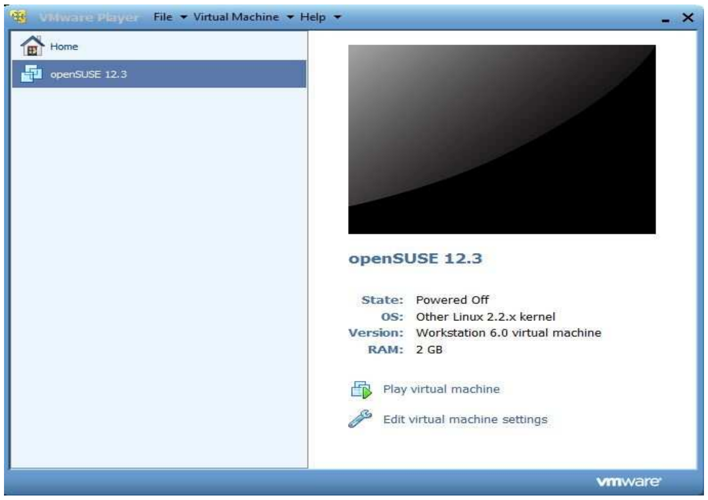
Upgrade to VMware Workstation
Get advanced features such as snapshots, developer tool integration, and more.

Help
View VMware Player's help contents.

Figure 1: VMware player installed and ready to use with openSUSE v12.3 O/S

2.2 Boot the openSUSE O/S on top of VMware Player

Before booting the O/S if you want you can change the current selected virtual machine settings by using “Edit virtual machine settings” link, see Figure 2.



openSUSE 12.3

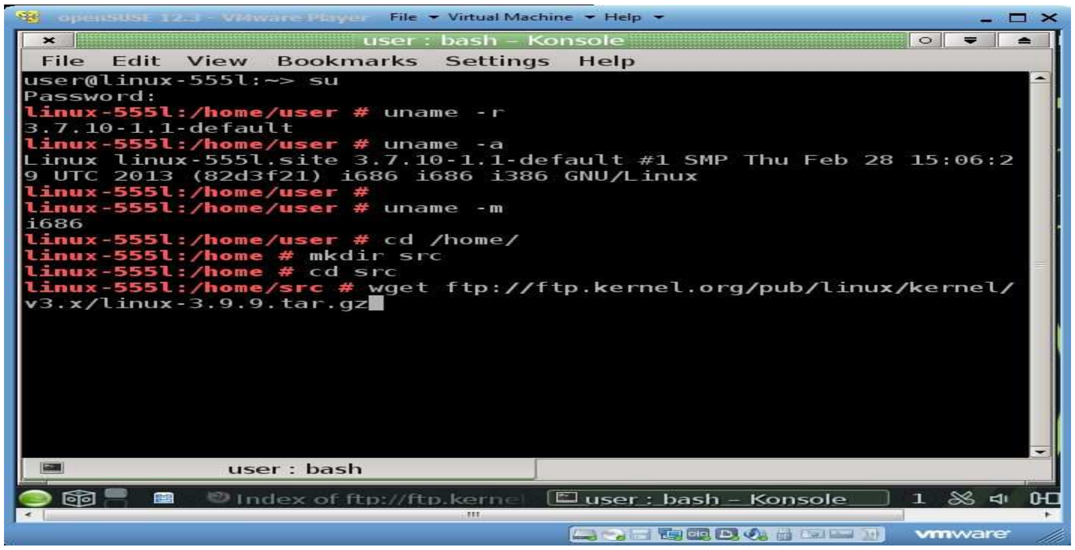
State: Powered Off
OS: Other Linux 2.2.x kernel
Version: Workstation 6.0 virtual machine
RAM: 2 GB

Figure 2: Select O/S on VMware player and boot the openSUSE v12.3 O/S

2.3 Check the current Linux kernel version after booting the O/S

Check your current working directory (user@linux-555l:~>). In order to install new kernel and access some root privileges you need to have super user rights. To obtain super user rights use “su” command followed by password, see Figure 3.

“uname” (short for *unix name*) is a software program in Unix and Unix-like computer operating systems that prints the name, version and other details about the current machine and the operating system running on it [6].

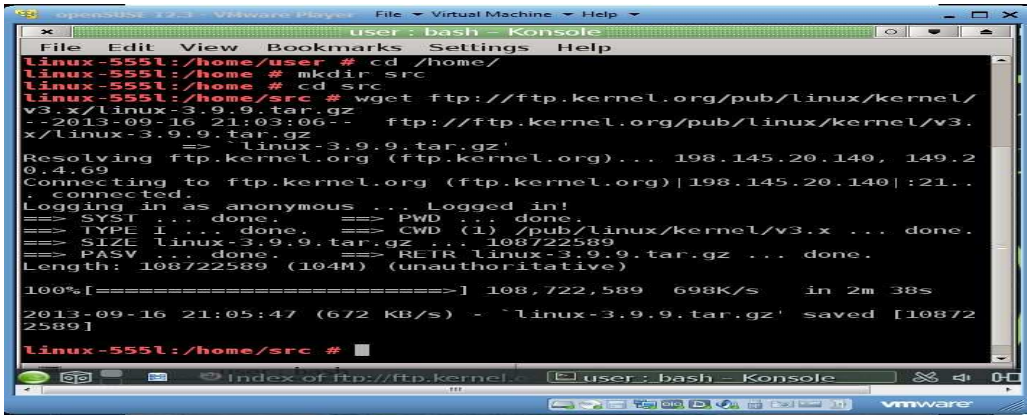

openSUSE 12.3 - VMware Player
File
Virtual Machine
Help
user : bash – Konsole
File Edit View Bookmarks Settings Help
user : bash
Index of ftp://ftp.kernel
user : bash – Konsole
vmware

```
user@linux-555l:~> su
Password:
linux-555l:/home/user # uname -r
3.7.10-1.1-default
linux-555l:/home/user # uname -a
Linux linux-555l.site 3.7.10-1.1-default #1 SMP Thu Feb 28 15:06:2
9 UTC 2013 (82d3f21) i686 i686 i386 GNU/Linux
linux-555l:/home/user #
linux-555l:/home/user # uname -m
i686
linux-555l:/home/user # cd /home/
linux-555l:/home # mkdir src
linux-555l:/home # cd src
linux-555l:/home/src # wget ftp://ftp.kernel.org/pub/linux/kernel/
v3.x/linux-3.9.9.tar.gz
```

Figure 3: Current Linux Kernel version and more details after booting the O/S

2.4 Download the latest available kernel and unzip it

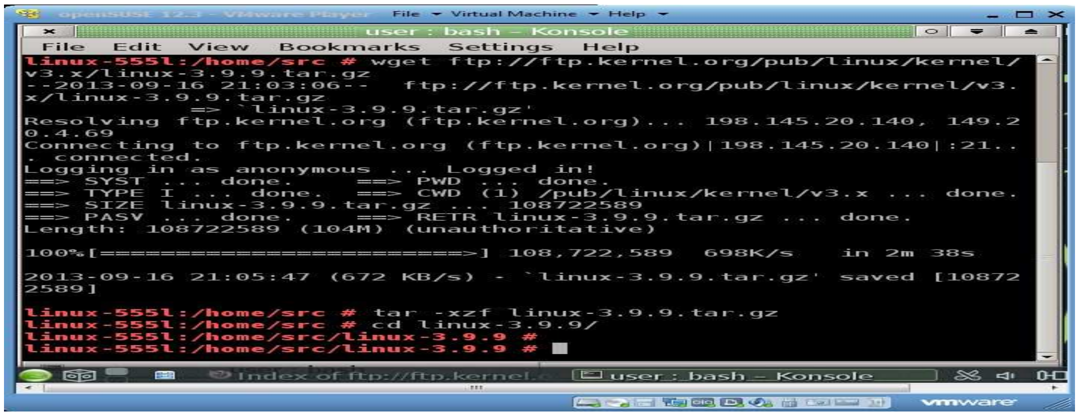
Check your current working directory (linux-555l:/home/user#). Change the directory to /home and crate new directory “src” using “mkdir” command, inside /home directory. Change the current directory to linux-555l://home/src#. Using “wget” command downloads the latest available kernel to “src” directory; see Figure 4 and Figure 5.


openSUSE 12.3 - VMware Player
File
Virtual Machine
Help
user : bash – Konsole
File Edit View Bookmarks Settings Help
Index of ftp://ftp.kernel.
user : bash – Konsole
vmware

```
linux-555l:/home/user # cd /home/
linux-555l:/home # mkdir src
linux-555l:/home # cd src
linux-555l:/home/src # wget ftp://ftp.kernel.org/pub/linux/kernel/
v3.x/linux-3.9.9.tar.gz
--2013-09-16 21:03:06--  ftp://ftp.kernel.org/pub/linux/kernel/v3.
x/linux-3.9.9.tar.gz
           => `linux-3.9.9.tar.gz'
Resolving ftp.kernel.org (ftp.kernel.org)... 198.145.20.140, 149.2
0.4.69
Connecting to ftp.kernel.org (ftp.kernel.org)|198.145.20.140|:21..
. connected.
Logging in as anonymous ... Logged in!
==> SYST ... done.    ==> PWD ... done.
==> TYPE I ... done.  ==> CWD (1) /pub/linux/kernel/v3.x ... done.
==> SIZE linux-3.9.9.tar.gz ... 108722589
==> PASV ... done.    ==> RETR linux-3.9.9.tar.gz ... done.
Length: 108722589 (104M) (unauthoritative)

100%[===========================>] 108,722,589  698K/s   in 2m 38s

2013-09-16 21:05:47 (672 KB/s) - `linux-3.9.9.tar.gz' saved [10872
2589]

linux-555l:/home/src # 
```

Figure 4: Download the latest available kernel

Unzip the kernel and change the current working directory to “linux-3.9.9”, see Figure 5.


openSUSE 12.3 - VMware Player
File Virtual Machine Help
user : bash – Konsole
File Edit View Bookmarks Settings Help

```
linux-555l:/home/src # wget ftp://ftp.kernel.org/pub/linux/kernel/
v3.x/linux-3.9.9.tar.gz
--2013-09-16 21:03:06--  ftp://ftp.kernel.org/pub/linux/kernel/v3.
x/linux-3.9.9.tar.gz
           => `linux-3.9.9.tar.gz'
Resolving ftp.kernel.org (ftp.kernel.org)... 198.145.20.140, 149.2
0.4.69
Connecting to ftp.kernel.org (ftp.kernel.org)|198.145.20.140|:21..
. connected.
Logging in as anonymous ... Logged in!
==> SYST ... done.    ==> PWD ... done.
==> TYPE I ... done.  ==> CWD (1) /pub/linux/kernel/v3.x ... done.
==> SIZE linux-3.9.9.tar.gz ... 108722589
==> PASV ... done.    ==> RETR linux-3.9.9.tar.gz ... done.
Length: 108722589 (104M) (unauthoritative)

100%[=========================>] 108,722,589  698K/s   in 2m 38s

2013-09-16 21:05:47 (672 KB/s) - `linux-3.9.9.tar.gz' saved [10872
2589]

linux-555l:/home/src # tar -xzf linux-3.9.9.tar.gz
linux-555l:/home/src # cd linux-3.9.9/
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 #
```

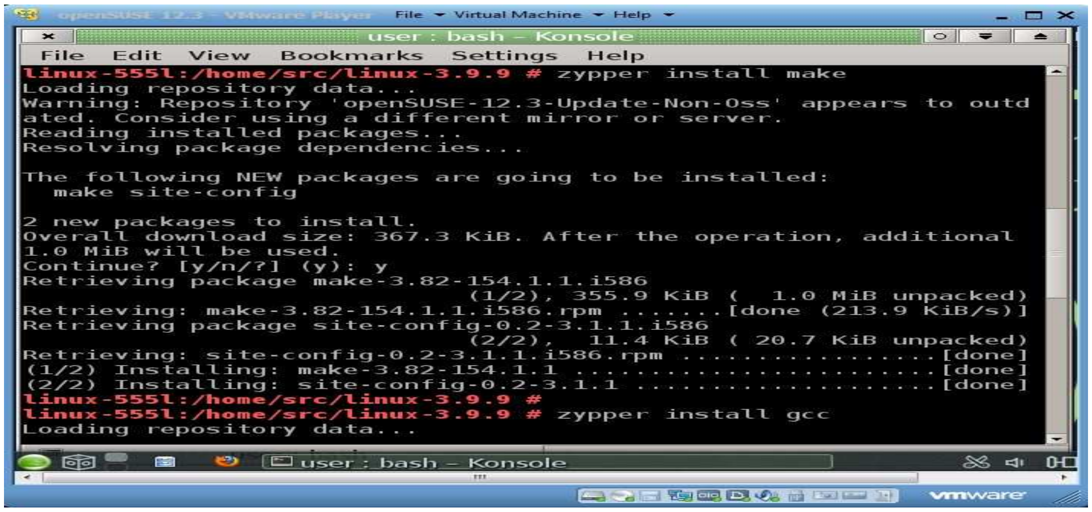
Figure 5: Unzip the latest available kernel

2.5 Install some prerequisites in order to compile the kernel

Install “Make”, “GCC” and “ncurses-devel” tools using “zypper install” command, see Figure 6.

In software development, Make is a utility that automatically builds executable programs and libraries from source code by reading files called makefiles which specify how to derive the target program. Though integrated development environments and language-specific compiler features can also be used to manage a build process, Make remains widely used, especially in UNIX [7].

The GNU Compiler Collection (GCC) is a compiler system produced by the GNU Project supporting various programming languages. GCC is a key component of the GNU tool chain. The Free Software Foundation (FSF) distributes GCC under the GNU General Public License (GNU GPL). GCC has played an important role in the growth of free software, as both a tool and an example [8].

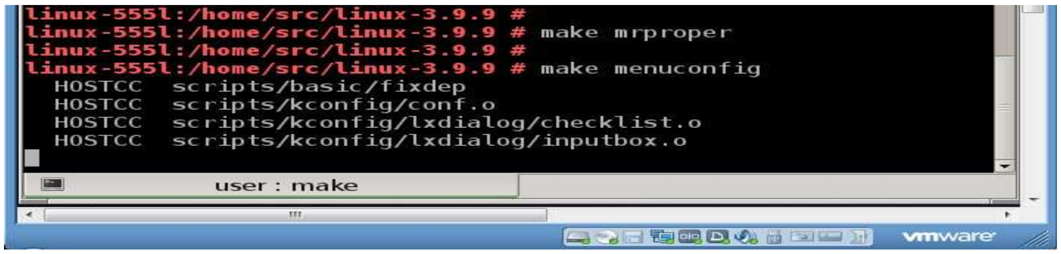

openSUSE 12.3 - VMware Player
File Virtual Machine Help
user : bash – Konsole
File Edit View Bookmarks Settings Help

```
linux-555l:/home/src/linux-3.9.9 # zypper install make
Loading repository data...
Warning: Repository 'openSUSE-12.3-Update-Non-Oss' appears to outd
ated. Consider using a different mirror or server.
Reading installed packages...
Resolving package dependencies...

The following NEW packages are going to be installed:
  make site-config

2 new packages to install.
Overall download size: 367.3 KiB. After the operation, additional
1.0 MiB will be used.
Continue? [y/n/?] (y): y
Retrieving package make-3.82-154.1.1.i586
                             (1/2), 355.9 KiB (  1.0 MiB unpacked)
Retrieving: make-3.82-154.1.1.i586.rpm .......[done (213.9 KiB/s)]
Retrieving package site-config-0.2-3.1.1.i586
                             (2/2),  11.4 KiB ( 20.7 KiB unpacked)
Retrieving: site-config-0.2-3.1.1.i586.rpm .................[done]
(1/2) Installing: make-3.82-154.1.1 ........................[done]
(2/2) Installing: site-config-0.2-3.1.1 ....................[done]
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 # zypper install gcc
Loading repository data...
```

Figure 6: Installation of “make” and “gcc”

2.6 Generate proper cleanup

Run "make mrproper" to make sure you have no stale .o files and dependencies lying around. This will also delete ".config" file that has the current kernel configuration options. If you want to keep the current configuration, one hack is to rename ".config" before runing "make mrproper" and rename the configuration file back to ".config". Here I do not want to backup current configuration file and I am going to create a new configuration file using “make menuconfig” command, see Figure 7.



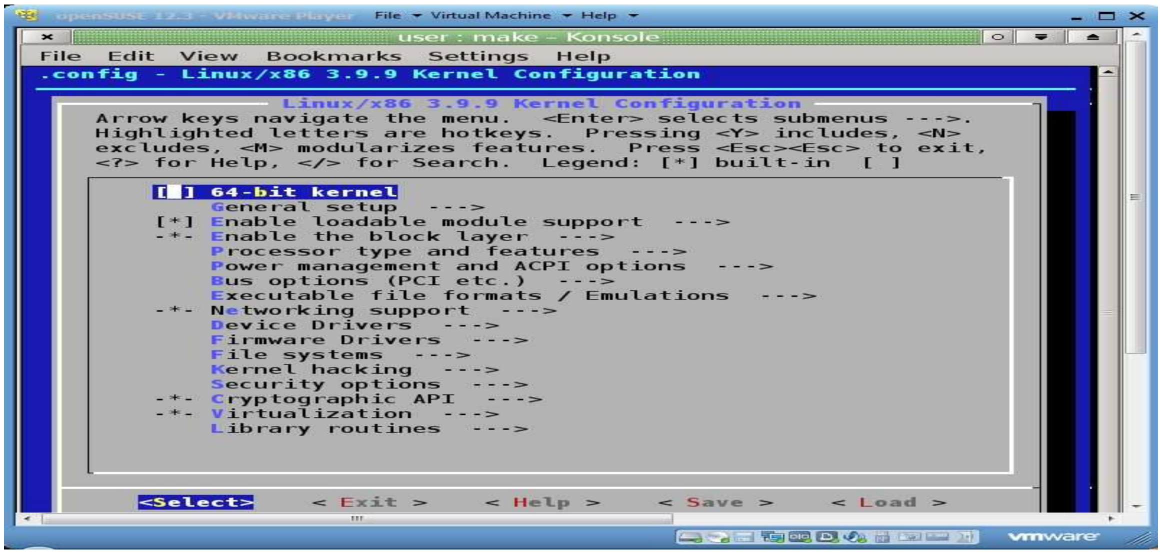
```
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 # make mrproper
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 # make menuconfig
  HOSTCC  scripts/basic/fixdep
  HOSTCC  scripts/kconfig/conf.o
  HOSTCC  scripts/kconfig/lxdialog/checklist.o
  HOSTCC  scripts/kconfig/lxdialog/inputbox.o
```

Figure 7: Generate proper cleanup and starting menu configuration

2.7 Configure the kernel

Run "make menuconfig" to configure the basic kernel. Do not skip this step even if you are only upgrading one minor version. If you do not want to change the current fossil client default kernel configuration, just "save" and "exit", see Figure 8.

“make menuconfig” is one of five similar tools that can configure the Linux kernel source, a necessary early step needed to compile the source code. make menuconfig, with a menu-driven user-interface, allows the user to choose the features of the Linux kernel (and other options) that will be compiled. It is normally invoked using the command make menuconfig, menuconfig is a target in the Linux kernel Makefile [10].



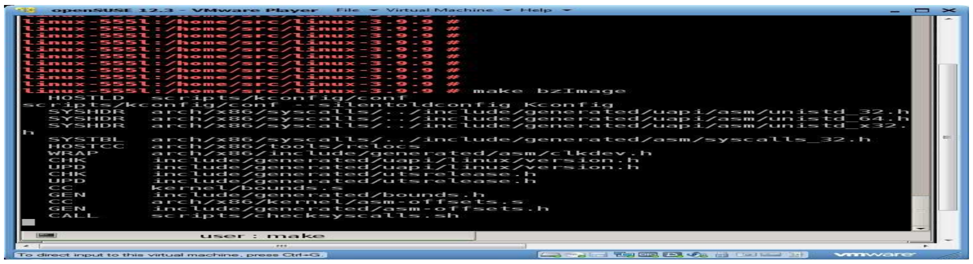
openSUSE 12.3 - VMware Player
File
Virtual Machine
Help
user : make - Konsole
File Edit View Bookmarks Settings Help
vmware

```
.config - Linux/x86 3.9.9 Kernel Configuration
                 Linux/x86 3.9.9 Kernel Configuration
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N>
  excludes, <M> modularizes features.  Press <Esc><Esc> to exit,
  <?> for Help, </> for Search.  Legend: [*] built-in  [ ]

        [ ] 64-bit kernel
            General setup  --->
        [*] Enable loadable module support  --->
        -*- Enable the block layer  --->
            Processor type and features  --->
            Power management and ACPI options  --->
            Bus options (PCI etc.)  --->
            Executable file formats / Emulations  --->
        -*- Networking support  --->
            Device Drivers  --->
            Firmware Drivers  --->
            File systems  --->
            Kernel hacking  --->
            Security options  --->
        -*- Cryptographic API  --->
        -*- Virtualization  --->
            Library routines  --->

    <Select>    < Exit >    < Help >    < Save >    < Load >
```

Figure 8: Menu configuration

2.8 Build the kernel

To build the kernel run “make bzImage” command and it will take about 45 minutes, see Figure 9.



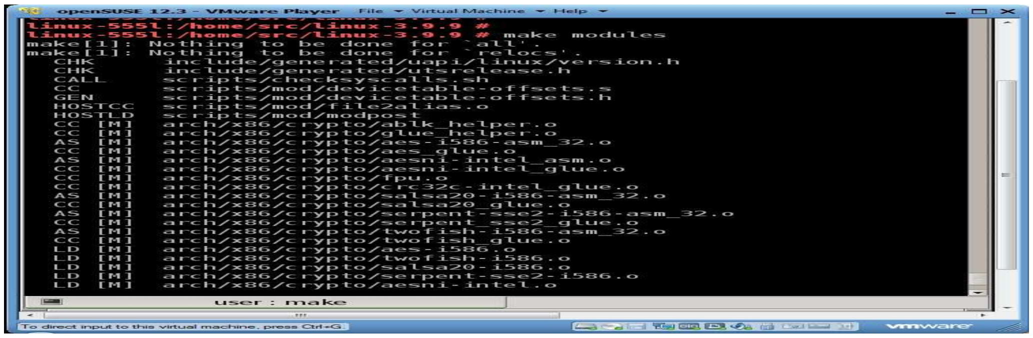
openSUSE 12.3 - VMware Player File Virtual Machine Help
user : make
To direct input to this virtual machine, press Ctrl+G.
vmware

```
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 # make bzImage
  HOSTLD  scripts/kconfig/conf
scripts/kconfig/conf --silentoldconfig Kconfig
  SYSHDR  arch/x86/syscalls/../include/generated/uapi/asm/unistd_32.h
  SYSHDR  arch/x86/syscalls/../include/generated/uapi/asm/unistd_64.h
  SYSHDR  arch/x86/syscalls/../include/generated/uapi/asm/unistd_x32.
h
  SYSTBL  arch/x86/syscalls/../include/generated/asm/syscalls_32.h
  HOSTCC  arch/x86/tools/relocs
  WRAP    arch/x86/include/generated/asm/clkdev.h
  CHK     include/generated/uapi/linux/version.h
  UPD     include/generated/uapi/linux/version.h
  CHK     include/generated/utsrelease.h
  UPD     include/generated/utsrelease.h
  CC      kernel/bounds.s
  GEN     include/generated/bounds.h
  CC      arch/x86/kernel/asm-offsets.s
  GEN     include/generated/asm-offsets.h
  CALL    scripts/checksyscalls.sh
```

Figure 9: Building the Linux kernel

2.9 Build the kernel modules

To build the kernel modules run “make modules” command and it will take about 2 hours, see Figure 10.



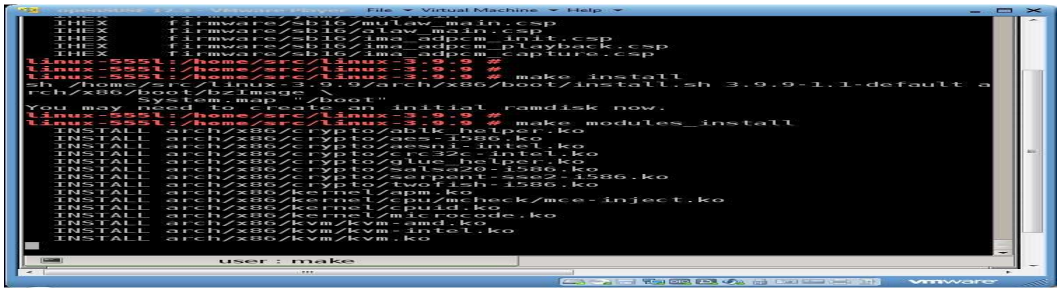
openSUSE 12.3 - VMware Player File Virtual Machine Help
user : make
To direct input to this virtual machine, press Ctrl+G.
vmware

```
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 # make modules
make[1]: Nothing to be done for `all'.
make[1]: Nothing to be done for `relocs'.
  CHK     include/generated/uapi/linux/version.h
  CHK     include/generated/utsrelease.h
  CALL    scripts/checksyscalls.sh
  CC      scripts/mod/devicetable-offsets.s
  GEN     scripts/mod/devicetable-offsets.h
  HOSTCC  scripts/mod/file2alias.o
  HOSTLD  scripts/mod/modpost
  CC [M]  arch/x86/crypto/ablk_helper.o
  CC [M]  arch/x86/crypto/glue_helper.o
  AS [M]  arch/x86/crypto/aes-i586-asm_32.o
  CC [M]  arch/x86/crypto/aes_glue.o
  AS [M]  arch/x86/crypto/aesni-intel_asm.o
  CC [M]  arch/x86/crypto/aesni-intel_glue.o
  CC [M]  arch/x86/crypto/fpu.o
  CC [M]  arch/x86/crypto/crc32c-intel_glue.o
  AS [M]  arch/x86/crypto/salsa20-i586-asm_32.o
  CC [M]  arch/x86/crypto/salsa20_glue.o
  AS [M]  arch/x86/crypto/serpent-sse2-i586-asm_32.o
  CC [M]  arch/x86/crypto/serpent_sse2_glue.o
  AS [M]  arch/x86/crypto/twofish-i586-asm_32.o
  CC [M]  arch/x86/crypto/twofish_glue.o
  LD [M]  arch/x86/crypto/aes-i586.o
  LD [M]  arch/x86/crypto/twofish-i586.o
  LD [M]  arch/x86/crypto/salsa20-i586.o
  LD [M]  arch/x86/crypto/serpent-sse2-i586.o
  LD [M]  arch/x86/crypto/aesni-intel.o
```

Figure 10: Building the Linux kernel modules

2.10 Install the kernel and kernel modules

To install the kernel run “make install” command and it will take about 15 minutes. To install kernel modules run “make modules_install” command and it will take about 1 hour, see Figure 11.



```
  IHEX    firmware/sb16/mulaw_main.csp
  IHEX    firmware/sb16/alaw_main.csp
  IHEX    firmware/sb16/ima_adpcm_init.csp
  IHEX    firmware/sb16/ima_adpcm_playback.csp
  IHEX    firmware/sb16/ima_adpcm_capture.csp
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 # make install
sh /home/src/linux-3.9.9/arch/x86/boot/install.sh 3.9.9-1.1-default a
rch/x86/boot/bzImage \
        System.map "/boot"
You may need to create an initial ramdisk now.
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 # make modules_install
  INSTALL arch/x86/crypto/ablk_helper.ko
  INSTALL arch/x86/crypto/aes-i586.ko
  INSTALL arch/x86/crypto/aesni-intel.ko
  INSTALL arch/x86/crypto/crc32c-intel.ko
  INSTALL arch/x86/crypto/glue_helper.ko
  INSTALL arch/x86/crypto/salsa20-i586.ko
  INSTALL arch/x86/crypto/serpent-sse2-i586.ko
  INSTALL arch/x86/crypto/twofish-i586.ko
  INSTALL arch/x86/kernel/apm.ko
  INSTALL arch/x86/kernel/cpu/mcheck/mce-inject.ko
  INSTALL arch/x86/kernel/cpuid.ko
  INSTALL arch/x86/kernel/microcode.ko
  INSTALL arch/x86/kvm/kvm-amd.ko
  INSTALL arch/x86/kvm/kvm-intel.ko
  INSTALL arch/x86/kvm/kvm.ko
```

Figure 11: Install the kernel and Install kernel modules

2.11 Copy the kernel and the system map and create initial RAMDisk

In Linux, the System.map file is a symbol table used by the kernel. A symbol table is a look-up between symbol names and their addresses in memory. A symbol name may be the name of a variable or the name of a function. The System.map is required when the address of a symbol name, or the symbol name of an address, is needed. It is especially useful for debugging "kernel panics" and "kernel oopses". The kernel does the address-to-name translation itself when CONFIG_KALLSYMS is enabled so that tools like ksymoops are not required [12].

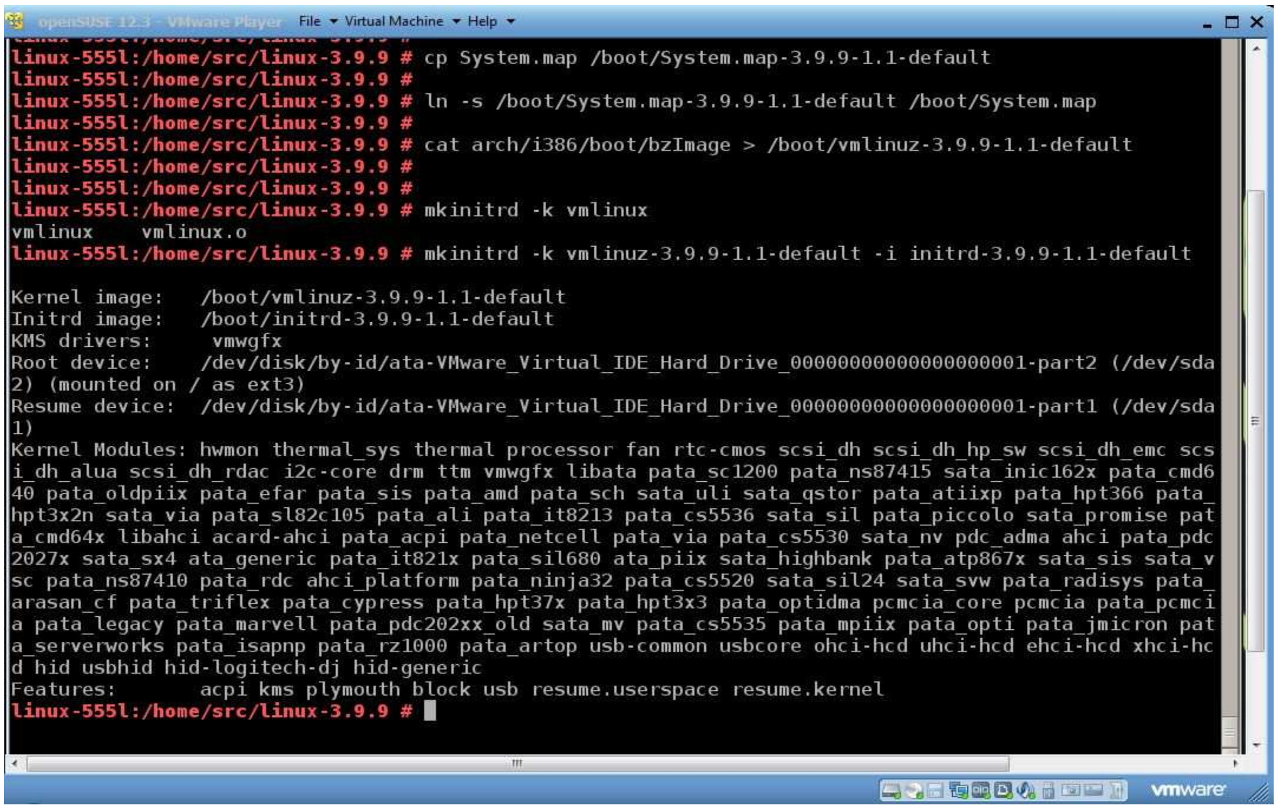
In computing, "initrd" (initial ramdisk) is a scheme for loading a temporary root file system into memory in the boot process of the Linux kernel. initrd and initramfs refer to two different methods of achieving this. Both are commonly used to make preparations before the real root file system can be mounted [13].

After installing Linux kernel and kernel modules, copy new kernel and system map as follows, see Figure 12.

- cp System.map /boot/System.map-3.9.9-1.1-default
- ln -s /boot/ System.map-3.9.9-1.1-default /boot/System.map

After that create initial RAMDisk as follows, see Figure 12.

- cat arch /i386/boot/bzImage> /boot/vmlinuz-3.9.9-1.1-default
- mkinitrd –k vmlinuz-3.9.9-1.1-default –i initrd-3.9.9-1.1-default



```
linux-555l:/home/src/linux-3.9.9 # cp System.map /boot/System.map-3.9.9-1.1-default
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 # ln -s /boot/System.map-3.9.9-1.1-default /boot/System.map
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 # cat arch/i386/boot/bzImage > /boot/vmlinuz-3.9.9-1.1-default
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 #
linux-555l:/home/src/linux-3.9.9 # mkinitrd -k vmlinux
vmlinux    vmlinux.o
linux-555l:/home/src/linux-3.9.9 # mkinitrd -k vmlinuz-3.9.9-1.1-default -i initrd-3.9.9-1.1-default

Kernel image:   /boot/vmlinuz-3.9.9-1.1-default
Initrd image:   /boot/initrd-3.9.9-1.1-default
KMS drivers:     vmwgfx
Root device:    /dev/disk/by-id/ata-VMware_Virtual_IDE_Hard_Drive_00000000000000000001-part2 (/dev/sda
2) (mounted on / as ext3)
Resume device:  /dev/disk/by-id/ata-VMware_Virtual_IDE_Hard_Drive_00000000000000000001-part1 (/dev/sda
1)
Kernel Modules: hwmon thermal_sys thermal processor fan rtc-cmos scsi_dh scsi_dh_hp_sw scsi_dh_emc scs
i_dh_alua scsi_dh_rdac i2c-core drm ttm vmwgfx libata pata_sc1200 pata_ns87415 sata_inic162x pata_cmd6
40 pata_oldpiix pata_efar pata_sis pata_amd pata_sch sata_uli sata_qstor pata_atiixp pata_hpt366 pata_
hpt3x2n sata_via pata_sl82c105 pata_ali pata_it8213 pata_cs5536 sata_sil pata_piccolo sata_promise pat
a_cmd64x libahci acard-ahci pata_acpi pata_netcell pata_via pata_cs5530 sata_nv pdc_adma ahci pata_pdc
2027x sata_sx4 ata_generic pata_it821x pata_sil680 ata_piix sata_highbank pata_atp867x sata_sis sata_v
sc pata_ns87410 pata_rdc ahci_platform pata_ninja32 pata_cs5520 sata_sil24 sata_svw pata_radisys pata_
arasan_cf pata_triflex pata_cypress pata_hpt37x pata_hpt3x3 pata_optidma pcmcia_core pcmcia pata_pcmci
a pata_legacy pata_marvell pata_pdc202xx_old sata_mv pata_cs5535 pata_mpiix pata_opti pata_jmicron pat
a_serverworks pata_isapnp pata_rz1000 pata_artop usb-common usbcore ohci-hcd uhci-hcd ehci-hcd xhci-hc
d hid usbhid hid-logitech-dj hid-generic
Features:       acpi kms plymouth block usb resume.userspace resume.kernel
linux-555l:/home/src/linux-3.9.9 #
```

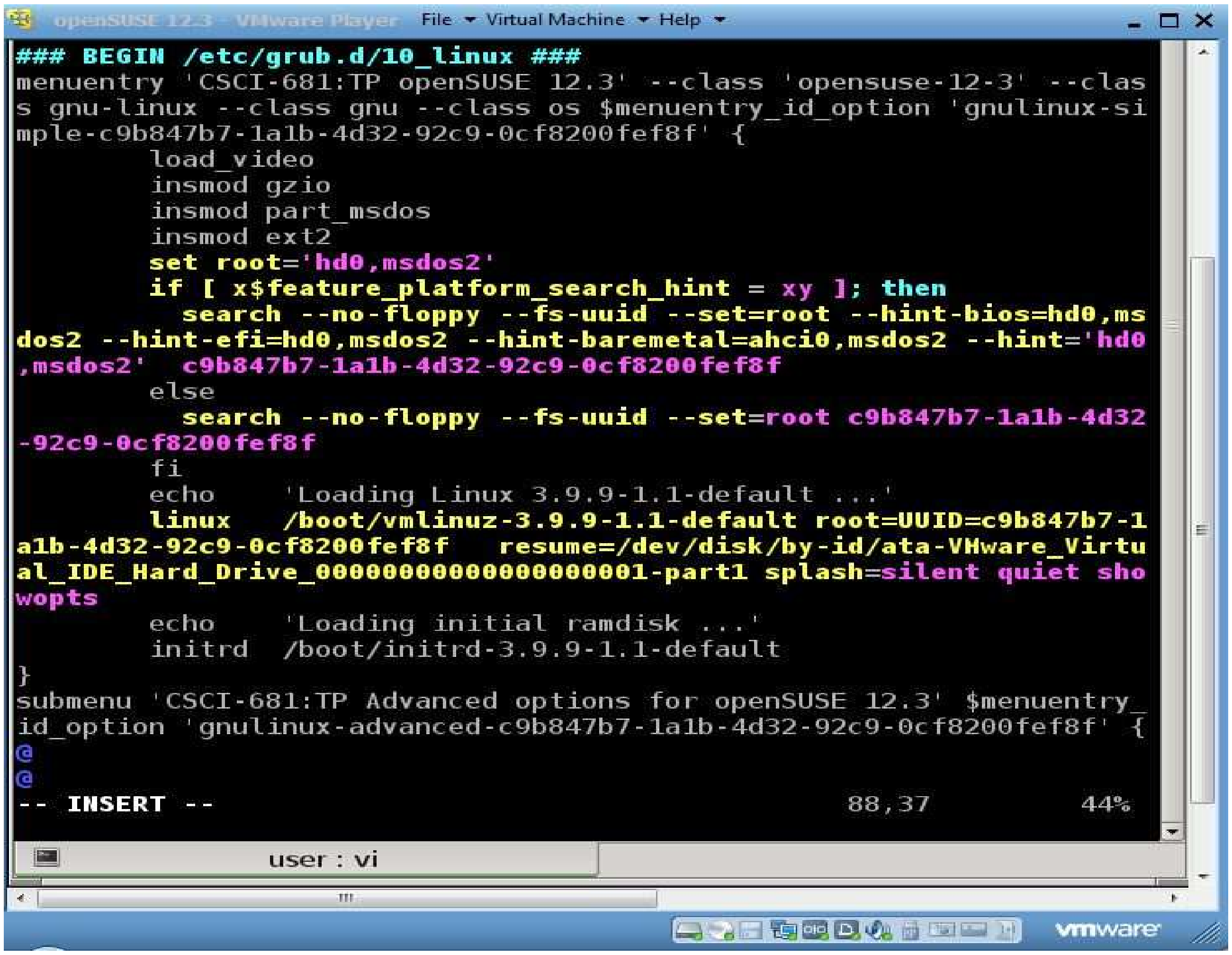
Figure 12: Copy the kernel and the system map and create initial RAMDisk

2.12 Edit the GRUB

GNU GRUB (GNU GRand Unified Bootloader) is a boot loader package from the GNU Project. GRUB is the reference implementation of the Free Software Foundation's Multiboot Specification, which provides a user the choice to boot one of multiple operating systems installed on a computer or select a specific kernel configuration available on a particular operating system's partitions [14].

GNU GRUB was developed from a package called the Grand Unified Bootloader (a play on Grand Unified Theory). It is predominantly used for Unix-like systems. The GNU operating system uses GNU GRUB as its boot loader, as do most Linux distributions. The Solaris operating system has used GRUB as its boot loader on x86 systems, starting with the Solaris 10 1/06 release [14].

In order to edit the GRUB use “vi /boot/grub2/grub.cfg” command and change the current O/S name to “CSCI-681: TP openSUSE 12.3” and make other required changes and save the file, see Figure 13.

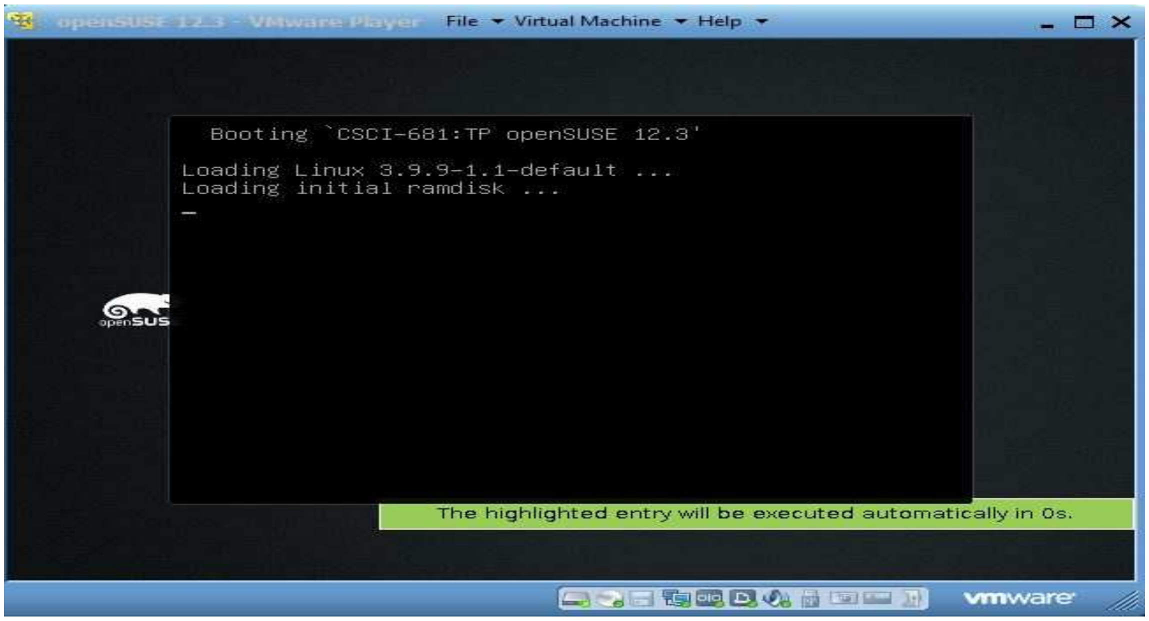


```
### BEGIN /etc/grub.d/10_linux ###
menuentry 'CSCI-681:TP openSUSE 12.3' --class 'opensuse-12-3' --clas
s gnu-linux --class gnu --class os $menuentry_id_option 'gnulinux-si
mple-c9b847b7-1a1b-4d32-92c9-0cf8200fef8f' {
        load_video
        insmod gzio
        insmod part_msdos
        insmod ext2
        set root='hd0,msdos2'
        if [ x$feature_platform_search_hint = xy ]; then
          search --no-floppy --fs-uuid --set=root --hint-bios=hd0,ms
dos2 --hint-efi=hd0,msdos2 --hint-baremetal=ahci0,msdos2 --hint='hd0
,msdos2'  c9b847b7-1a1b-4d32-92c9-0cf8200fef8f
        else
          search --no-floppy --fs-uuid --set=root c9b847b7-1a1b-4d32
-92c9-0cf8200fef8f
        fi
        echo    'Loading Linux 3.9.9-1.1-default ...'
        linux   /boot/vmlinuz-3.9.9-1.1-default root=UUID=c9b847b7-1
a1b-4d32-92c9-0cf8200fef8f   resume=/dev/disk/by-id/ata-VMware_Virtu
al_IDE_Hard_Drive_00000000000000000001-part1 splash=silent quiet sho
wopts
        echo    'Loading initial ramdisk ...'
        initrd  /boot/initrd-3.9.9-1.1-default
}
submenu 'CSCI-681:TP Advanced options for openSUSE 12.3' $menuentry_
id_option 'gnulinux-advanced-c9b847b7-1a1b-4d32-92c9-0cf8200fef8f' {
@
@
-- INSERT --                                      88,37         44%
```

Figure 13: Edit the GNU GRUB

2.13 Reboot the O/S

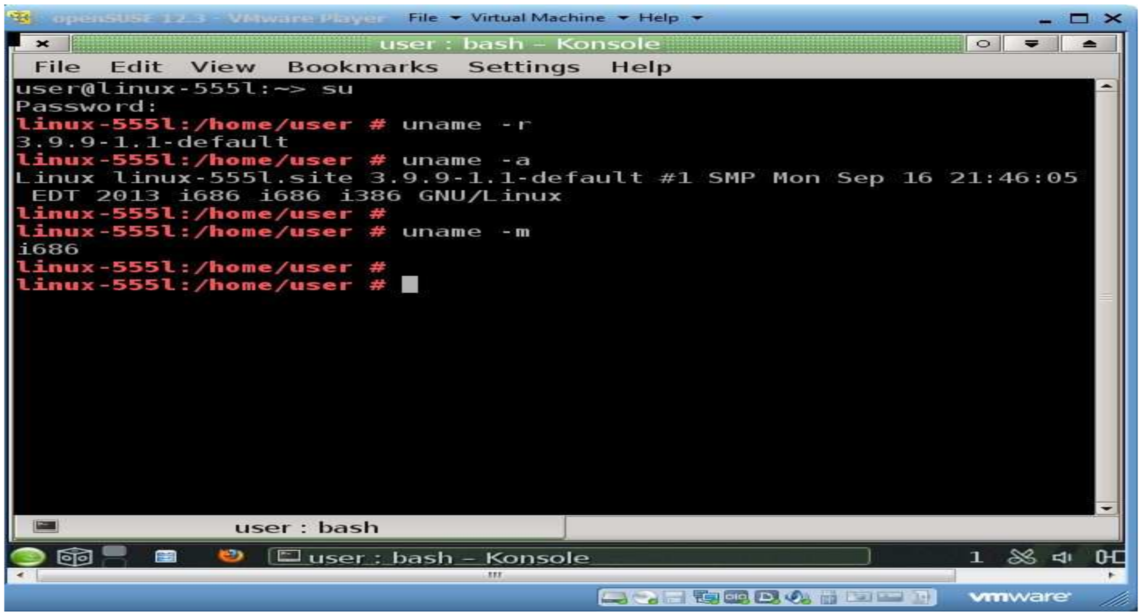
After configuring the GRUB restart the O/S to apply all new modifications to the kernel. Now the kernel compilation and installation are done. This concludes this section. Figure 14 shows Current Linux Kernel version while booting the O/S and Figure 15 shows Current Linux Kernel version and more details after booting the O/S.



```
  Booting `CSCI-681:TP openSUSE 12.3'

 Loading Linux 3.9.9-1.1-default ...
 Loading initial ramdisk ...
 _
```

The highlighted entry will be executed automatically in 0s.

Figure 14: Current Linux Kernel version while booting the O/S.



```
user@linux-555l:~> su
Password:
linux-555l:/home/user # uname -r
3.9.9-1.1-default
linux-555l:/home/user # uname -a
Linux linux-555l.site 3.9.9-1.1-default #1 SMP Mon Sep 16 21:46:05
 EDT 2013 i686 i686 i386 GNU/Linux
linux-555l:/home/user #
linux-555l:/home/user # uname -m
i686
linux-555l:/home/user #
linux-555l:/home/user #
```

Figure 15: Current Linux Kernel version and more details after booting the O/S

3 Discussions

This section presents issues encountered during the process of kernel compilation. When the technology is being updated, all the tools that we used here also are being updated. For example couple of years back in February 2012 we did the same experiment with VMware version 4, openSUSE 11.2 and Linux kernel 3.2.5. For this experiment we used VMware version 6, openSUSE 12.3 and Linux kernel 3.9.9. Within two year time period most of the things were updated and changed specially in the Linux kernel. For example previously GRUB configuration file is located at "/boot/grub/menu.lst" as menu.list, but now it is located at "/boot/grub2/grub.cfg" as grub.cfg. Also user interfaces of VMware and openSUSE are changed up to some certain point.

When compiling the kernel we came across few issues. Some of the programs that are required to execute some commands were not installed in openSUSE kernel. Particularly commands such as "make mrproper" and "make menuconfig" needed "make" and "gcc" programs (tools). Here we installed those tools manually using "zipper install" command as described in step 2.5.

Since we knew everything regarding Linux kernel compilation before start the experiment, this entire process took approximately 4 hours with Intel core i5 processor and 4 GB of RAM in hosted operating system. Also for the virtual O/S (openSUSE) we allocated 2 GB of RAM. To build the kernel it took about 45 minutes and to build the kernel modules it took about 2 hours. All the other steps took about 1 hour.

4 Conclusions

This virtual environment will help students to do so many trials before going to implement those in actual systems without doing any harm to current hosted operating system. Once students are perfectly sure with what they are doing, they can go to the real implementation without having any trouble.

Also if there is a major or minor improvement applied to current kernel and released it as new kernel version, you only have to download it from "www.kernel.org" then compile and install it by using this environment and above procedure.

This paper only discussed usage of VMware and Linux kernel compilation. We have developed several other experiments for "System Calls", "Kernel Modules and Proc Files", "Processes", "Scheduling" and "Synchronization" with the aids of screen shots for each and every step. We will plan to publish those experiments in the near future.

Here we have used free and open source tools for all the experiments. So if a particular organization or university would like to follow this learning environment for their experiments, this will be cost effective.

In the future we are planning to repeat these experiments with latest versions of VMware, openSUSE and Linux kernel once a year time intervals with all the steps illustrated above and publish a paper for each experiment. Also plan to extend these experiments with different Linux distributions such as Ubuntu, Fedora, Debian and so on.

From our initial planning input, we took actions to transform the course to a project based one and then collected results. Feedback from the action stage and results are used to improve the plan. We continuously improved the quality of instruction and actions while using the feedback received from the students, and took action to transform the course suitable for e-learning and for more planning. A course with the features outlined above could help our students develop design skills in several different languages before their graduation.

Also it is good to do performance analysis by doing the same experiment with different processors with different RAMs and measure the time taken to compile kernel and kernel module. So that we can come up with a solution: which kernel is suitable for which processor.

References

[1]. Wikipedia contributors. "Linux kernel." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 25 Mar. 2014. Web. 26 Mar. 2014.
[2]. Wikipedia contributors. "System call." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 10 Mar. 2014. Web. 26 Mar. 2014.
[3]. Wikipedia contributors. "OpenSUSE." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 22 Jan. 2014. Web. 26 Mar. 2014.
[4]. Wikipedia contributors. "VMware Player." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 15 Mar. 2014. Web. 26 Mar. 2014.
[5]. Lowe, Kwan. "Kernel rebuild guide." *Digital Hermit* (2004).
[6]. Wikipedia contributors. "Uname." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 21 Mar. 2014. Web. 26 Mar. 2014.
[7]. Wikipedia contributors. "Make (software)." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 24 Mar. 2014. Web. 26 Mar. 2014.
[8]. Wikipedia contributors. "GNU Compiler Collection." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 21 Mar. 2014. Web. 26 Mar. 2014.
[9]. Wikipedia contributors. "Ncurses." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 27 Jan. 2014. Web. 26 Mar. 2014.
[10]. Wikipedia contributors. "Menuconfig." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 2 Mar. 2014. Web. 26 Mar. 2014.
[11]. Wikipedia contributors. "Vmlinux." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 11 Mar. 2014. Web. 26 Mar. 2014.
[12]. Wikipedia contributors. "System.map." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 29 Oct. 2013. Web. 26 Mar. 2014.
[13]. Wikipedia contributors. "Initrd." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 9 Mar. 2014. Web. 26 Mar. 2014.
[14]. Wikipedia contributors. "GNU GRUB." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 23 Feb. 2014. Web. 26 Mar. 2014.

[15]. Laadan, Oren, Jason Nieh, and Nicolas Viennot. "Structured linux kernel projects for teaching operating systems concepts." *Proceedings of the 42nd ACM technical symposium on Computer science education*. ACM, 2011.
[16]. Moon, Jae Yun, and Lee Sproull. "Essence of distributed work: The case of the Linux kernel." *Distributed work* (2002): 381-404.
[17]. Nutt, Gary J. *Kernel projects for Linux*. Addison Wesley Longman, 2001.
[18]. Silberschatz, Abraham, Peter B. Galvin, and Greg Gagne. *Operating system concepts*. Vol. 8. Wiley, 2013.
[19]. Nieh, Jason, and Chris Vaill. "Experiences teaching operating systems using virtual platforms and linux." *ACM SIGCSE Bulletin*. Vol. 37. No. 1. ACM, 2005.
[20]. Hess, Rob, and Paul Paulson. "Linux kernel projects for an undergraduate operating systems course." *Proceedings of the 41st ACM technical symposium on Computer science education*. ACM, 2010.
[21]. Love, Robert. *Linux kernel development*. Pearson Education, 2010.