

ESTR 3102

Red Hat 9

Helen Chan

SHB 118

hwchan@cse.cuhk.edu.hk

Office Hours: Fri 10am-12pm, or by appointment

Agenda

- Motivation
- Red Hat 9 Installation
- Kernel Compilation
- Adding System Call (for kernel v.2.4.x)

Motivation

Red Hat 9?









- Released in March 2003
 - Default kernel version 2.4.20-8
 - Assignment 1 requires hacking in kernel 2.4.x
 - Compiling old kernels on platforms with new kernels and GCC can be complicated ...
- Use an old platform 😊

Red Hat 9 Installation

Step 1: Obtain Disk Images

- Obtain the Red Hat 9 installation disk images
- <ftp://ftp.cuhk.edu.hk/pub/Linux/redhat/redhat-9/iso/i386>
- Download the images **shrike-i386-disc*.iso**
 - Disc 3 is optional if you follow our installation guide 😊

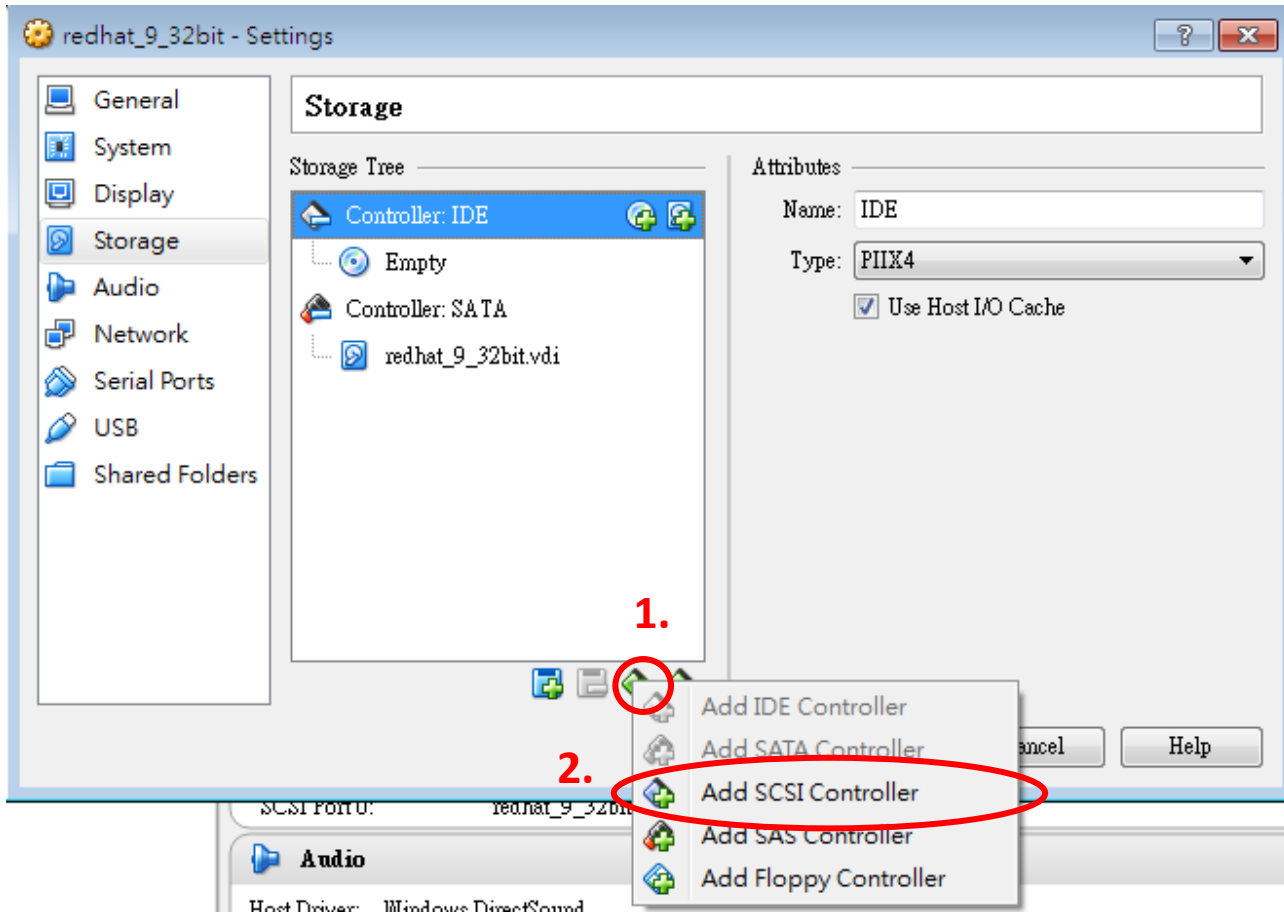
Index of /pub/Linux/redhat/redhat-9/iso/i386/

Name	Size	Date Modified
 [parent directory]		
 MD5SUM	575 B	4/18/07 12:00:00 AM
 shrike-SRPMS-disc1.iso	608 MB	3/14/03 12:00:00 AM
 shrike-SRPMS-disc2.iso	645 MB	3/14/03 12:00:00 AM
 shrike-SRPMS-disc3.iso	425 MB	3/14/03 12:00:00 AM
 shrike-i386-disc1.iso	638 MB	9/4/03 12:00:00 AM
 shrike-i386-disc2.iso	646 MB	3/14/03 12:00:00 AM
 shrike-i386-disc3.iso	485 MB	3/14/03 12:00:00 AM

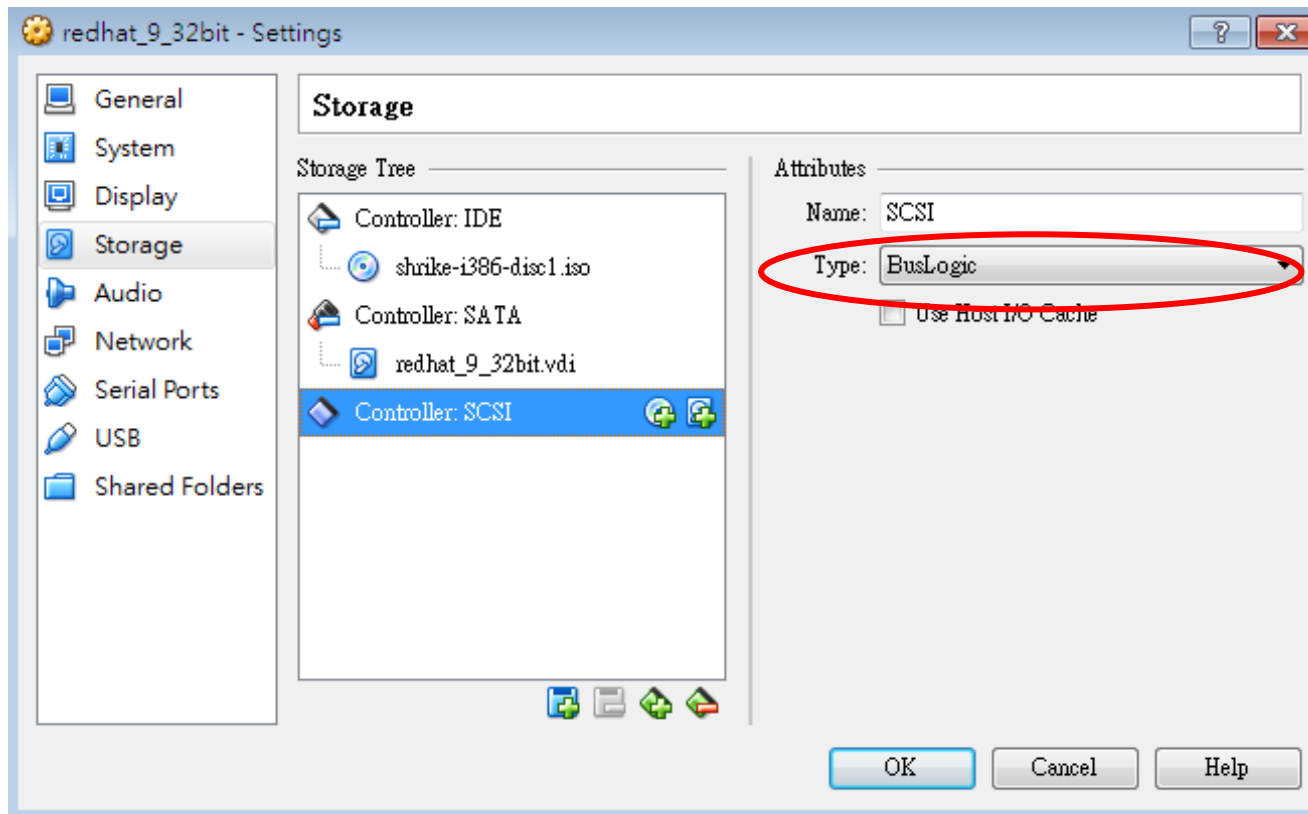
Step 2: Create the VM

(Virtual Box)

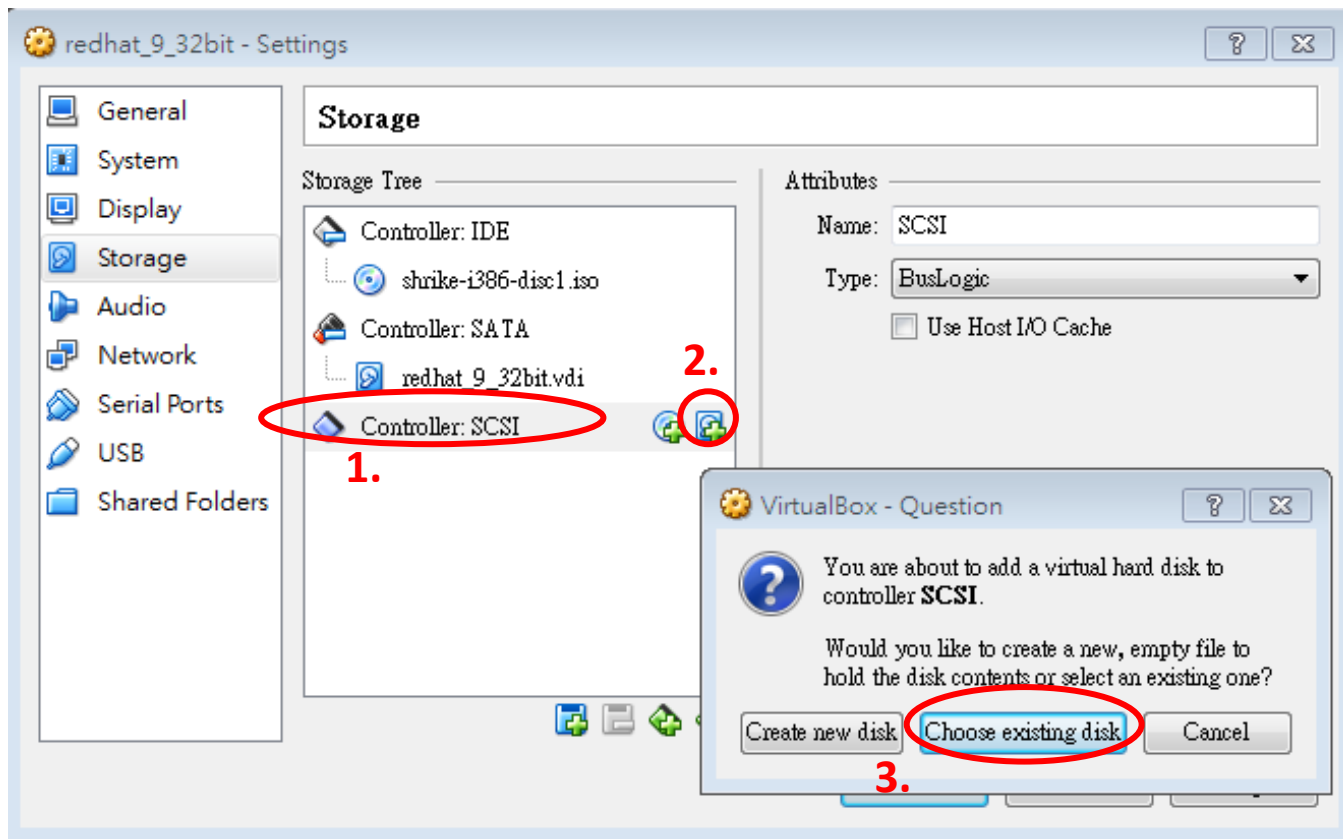
1. “New” a VM
 - OS: “Linux” and “Red Hat (32bit)”
2. Change the **disk controller** for hard disk from **“SATA” to “SCSI”**
 - a) Add a “SCSI” controller
 - b) Change the “SCSI” controller type to **“BusLogic”**
 - c) Connect the existing disk to the “SCSI” controller
 - d) Remove the “SATA” controller



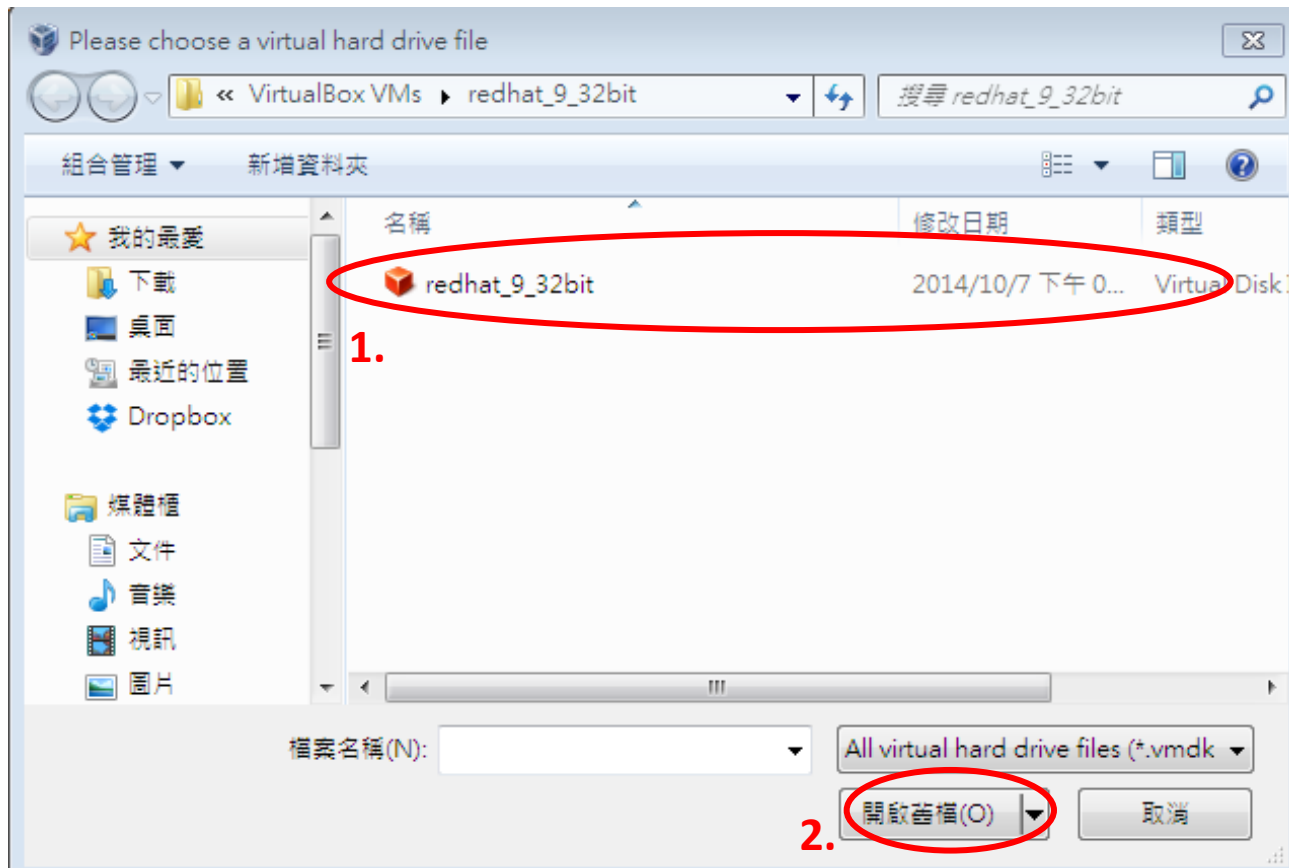
Add a SCSI controller for the Red Hat VM



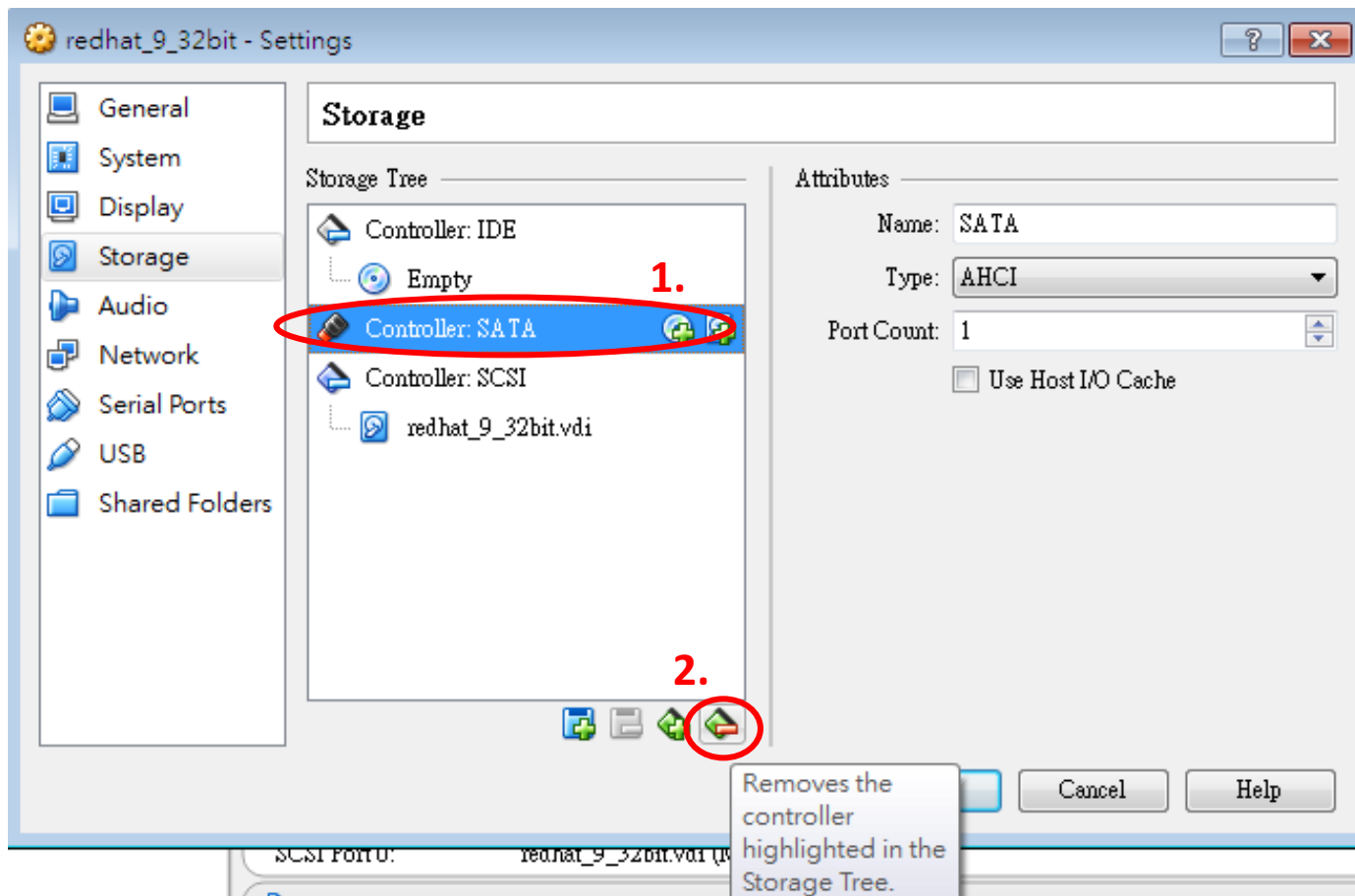
Change the type of “SCSI” controller to “BusLogic”



Connect the existing disk to the SCSI controller



Connect the existing disk to the SCSI controller
(By default, under the folder
“*[Home directory]/VirtualBox VMs/[Name of VM]/*”)

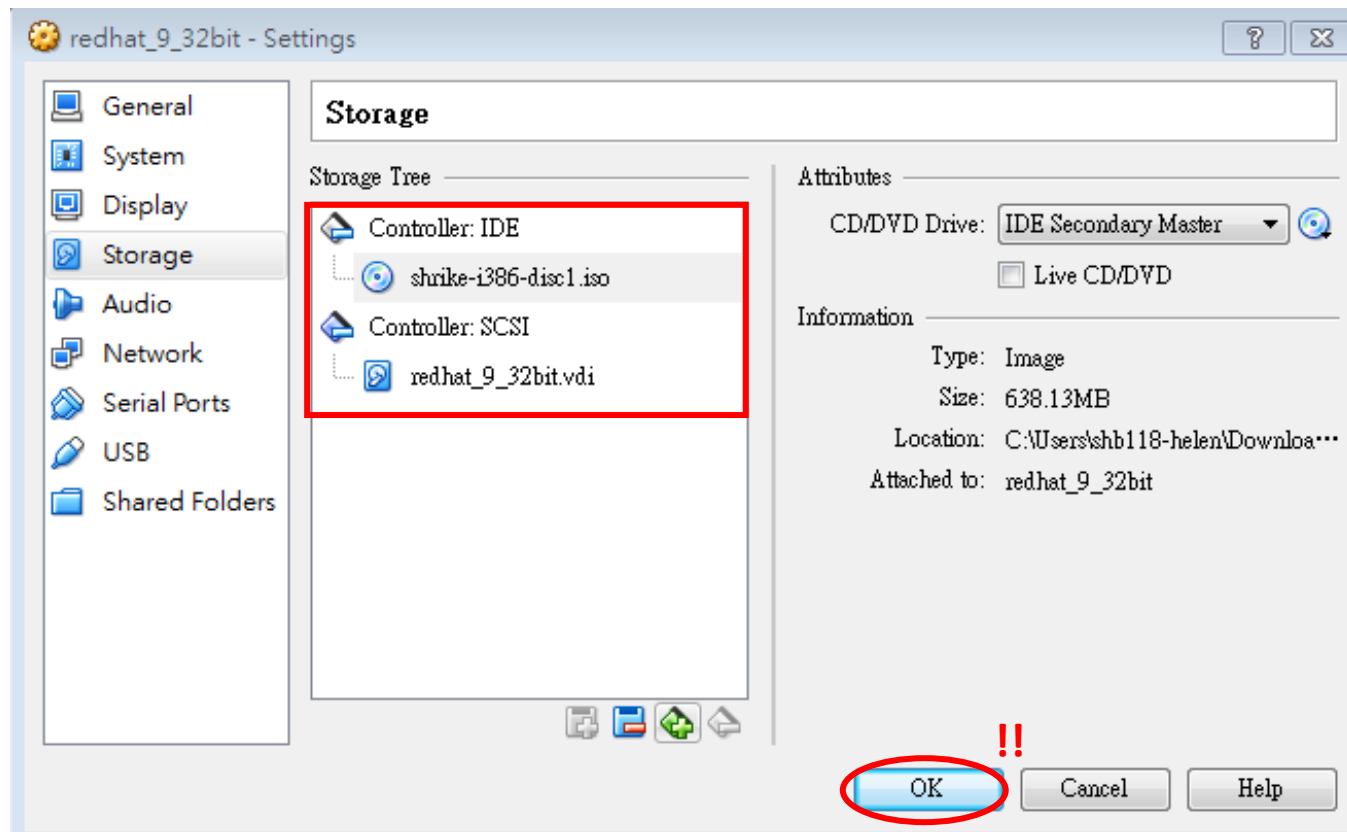


Remove the SATA controller

Step 2: Create the VM

(Virtual Box)

3. Insert the Red Hat installation CD, shrike-i386-disc1.iso
4. (Optional) Add a “Host-only Network Adaptor”
 - Settings > Network > Adaptor 2 > Enable Network Adaptor > “Host-only Adapter”
 - For host-to-VM SSH access



Final outcome (remember to click “OK”)

Step 2: Create the VM

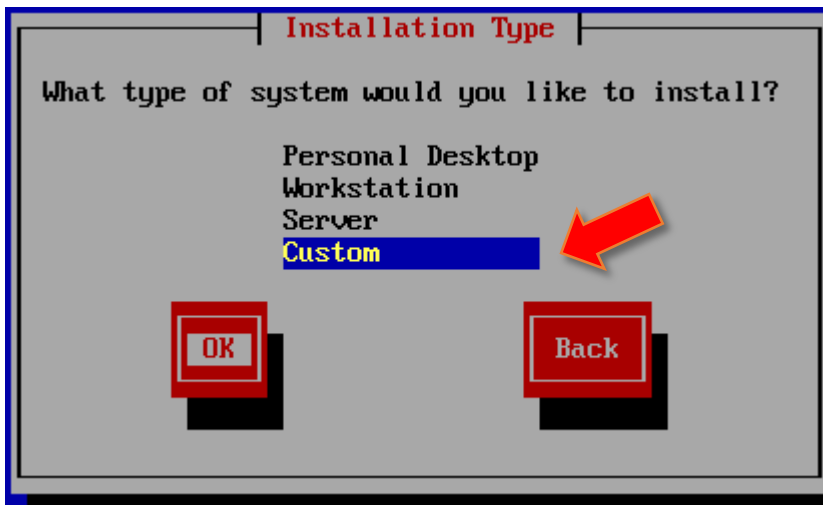
(VMWare)

1. “Create” a VM
 - OS: “**Linux**” and “**Red Hat Linux**”
2. No need to change the controller type
 - SCSI controller is used for hard disk by default
3. Insert the Red Hat Installation CD, shrike-i386-disc1.iso
4. (Optional) Add a “Host-only Network Adaptor”
 - Settings > Add > Network Adaptor > “Host-only”
 - For host-to-VM SSH access

Step 3: Install Red Hat 9

Note: Only the steps required manual setting are included

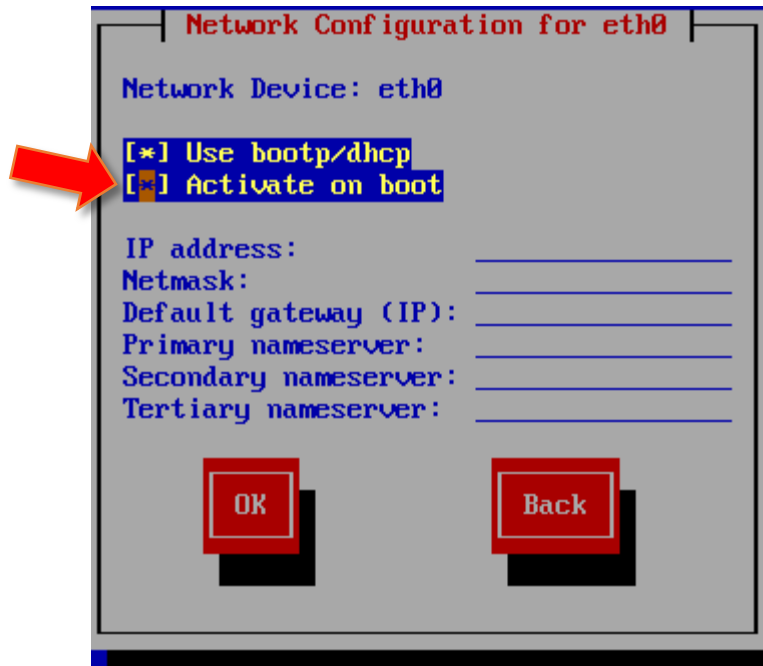
1. Start the VM
2. Type **“linux text”** and press “<Enter>”
3. Skip “testing the CD media”
4. Installation Type: **“Custom”**



Step 3: Install Red Hat 9

5. Network Card Configuration

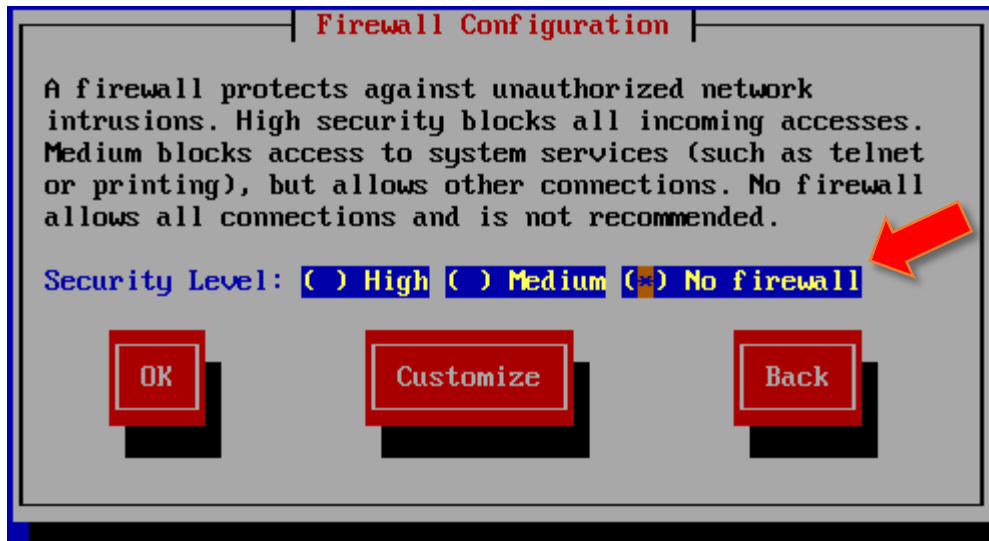
- Remember to check “Activate on boot”



Step 3: Install Red Hat 9

6. Firewall Configuration: **“No firewall”**

- If “medium” or “high” is chose, you will need to change the firewall setting in order to unblock the SSH port



Step 3: Install Red Hat 9

7. Set **root password**
8. Package Group Selection: “**Developer Tools**” and “**Editors**” are essential
9. Skip creating a diskette

Kernel Compilation

Step 1: Obtain Kernel Source Code

- From kernel.org in the [folder](#)
 - Today's Kernel: 2.4.37.10
1. Login as **root**
 2. Get the kernel source tarball
 - *# wget http://ftp.kernel.org/pub/linux/kernel/v2.4/linux-2.4.37.10.tar.gz*
 3. Decompress the source tarball
 - *# tar zxf linux-2.4.37.10.tar.gz -C /usr/src*
 - (ignore the “Unknown file type ‘g’” warning)

Step 2: Compile Kernel

1. Go to the root folder of kernel source
 - `# cd /usr/src/linux-2.4.37.10`
2. Copy the old kernel configuration
 - `# cp /boot/config-2.4.20-8 ./config`

Step 2: Compile Kernel

3. Update the configuration file

- *# yes "" | make oldconfig*

4. Compile the new kernel

- *# make dep && make*
- *# make modules modules_install*
- *# make install*

5. Reboot into new kernel

- Note: The original kernel (v2.4.20-8) is still there. Reboot into this old kernel in case the new kernel (v2.4.37-10) crushes

Adding System Call

(Materials in this section are mostly adopted from slides by Dr. Wong in previous semesters 😊)

Step 0: Choosing Text Editor

- By default, Red Hat comes with “vim” and “emacs”
- If you need “nano”,
 - Download the package:
wget http://www.cse.cuhk.edu.hk/~hwchan/nano.rpm
 - Install the package:
rpm -i nano.rpm

Step 1: Register the System Call

1. Go to the root folder of kernel source

- `# cd /usr/src/linux-2.4.37.10`

2. Add a record to the system call table in

arch/i386/kernel/entry.S [Line 406]

```
.....
ENTRY(sys_call_table)
    .long SYMBOL_NAME(sys_ni_syscall)    /* 0 - old "setup()" system call*/
    .long SYMBOL_NAME(sys_exit)
        .....
    .long SYMBOL_NAME(sys_ni_syscall)    /* sys_set_tid_address */
    .long SYMBOL_NAME(sys_hello_world)  /* 259 hello_world */

    .rept NR_syscalls-(.-sys_call_table)/4
        .long SYMBOL_NAME(sys_ni_syscall)
    .endr
```

Step 1: Register the System Call

3. Add a record to the list of system call numbers in *include/asm/unistd.h* [Line 4]

```
.....  
/*  
 * This file contains the system call numbers  
 */  
  
#define __NR_exit          1  
    .....  
#define __NR_exit_group   252  
#define __NR_hello_world  259  
  
/* user-visible error numbers are in the range -1 - -124: see .....
```

Step 2: Implement the System Call

1. Implement the system call

- Prefer any *.c file under *kernel/*
- In *kernel/sys.c* [Line 1285]

```
asmlinkage int sys_hello_world(void) {  
    printk(KERN_INFO "System call hello_world :D\n");  
    return 0;  
}
```

```
EXPORT_SYMBOL(notifier_chain_register);  
    .....  
EXPORT_SYMBOL(in_egroup_p);
```

Step 3: Recompile Kernel

1. Rebuild the kernel
 - *# make && make install*
2. Reboot

Step 4: Update Header Files

- *Note: This step only need to be performed once. Skip this step when adding new system calls in the future*

1. Back up original headers

- `# mv /usr/include/linux /usr/include/linux.origin`
- `# mv /usr/include/asm /usr/include/asm.origin`

2. Create symbolic links to new headers

- `# ln -s /usr/src/linux-2.4.37.10/include/linux \ /usr/include/linux`
- `# ln -s /usr/src/linux-2.4.37.10/include/asm \ /usr/include/asm`

Step 5: Test the System Call

1. Create a tiny test program

- Let's call it *hello.c*

```
#include <stdio.h>
#include <linux/unistd.h>
#include <errno.h>
#include <string.h>

int main (void) {
    int ret = syscall(__NR_hello_world);
    if (ret != 0) {
        printf("Error: %s\n", strerror(errno));
    }
    return 0;
}
```

Step 5: Test the System Call

2. Compile the program

- `# gcc -o hello hello.c`

3. Run the program

- `# ./hello`
- No output to console
- `# dmesg | tail`

```
.....  
System call hello_world :D
```


Appendix

Appendix – About Setup

(Virtual Box)

- Why use “SCSI” controller?
 - Red Hat 9 does not recognize SATA controllers
 - IDE controller also works, but to play safe, we follow the default configuration in VMware
- Why choose “BusLogic”?
 - Again, we follow the default configuration in VMware

Appendix – About Setup

- Why “Customize” Red Hat?
 - Get important packages installed
 - Developer Tools: Compilers like GCC and related libraries are essential for our work
 - Editors: Just a matter of choice 😊 (the tutor prefers *vim*)
 - Remove packages that are unlikely to be used, saving space and time spend on installation

Appendix – Known Problems

- For Virtual Box
 - VM never shutdown properly (CPU usage is ~100%)...

```
Unmounting file systems: [ OK ]  
Halting system...  
flushing ide devices: hdc  
Power down.
```

- Reboot is fine
- The “solution” ... force VM to close
 - Close > Power off
- Work-around
 - Use VMware Player / Workstation
 - However, snapshots are not supported in VMware Player

Appendix – Tips on GRUB

- Boot into new kernel by default
 - Modify */boot/grub/grub.conf*

```
# grub.conf generated by anaconda
.....
#boot=/dev/sda
default=0
timeout=10
splashimage=(hd0,0)/grub/splash.xpm.gz
title Red Hat Linux (2.4.37.10)           /* Record 0 */
.....
title Red Hat Linux (2.4.20-8)         /* Record 1 */
.....
```

- [!!!] Never set *timeout* to ≤ 0