CS460 – In Class Kernel Lab

May 2, 2016

Open a Linux console. Start VirtualBox

This was update on May 1, 2016 to work with archlinux-2016.01.01-dual.iso and ArchLinuxStudent.vdi.

$ wget http://zeus.cs.pacificu.edu/chadd/ArchLinux64Bit-CS460StudentCloneBackup.ova

$ VirtualBox

Import Appliance

(login as **cs460**)

Username: cs460
Password: CS460!!pac

root password is: CS460!!pac

Wait for the Desktop to come up.

Now we are ready to work.

Make sure the date and time are correct. **Do this every time you reboot!**

[root@myhost]# date
[root@myhost]# date *MMDDhhmm[CCYY]*

For Example:
[root@myhost]# date 050116452016

Let's get the kernel:

wget -c https://www.kernel.org/pub/linux/kernel/v4.x/linux-4.3.3.tar.gz

Let's extract and configure the kernel (do this exactly once!):

```
tar xvjf linux-4.3.3.ta.gz
cd linux-4.3.3
make mrproper

zcat /proc/config.gz > .config          # get current configuration from the kernel
make oldconfig                                  # update with any new configuration options
```

Change the kernel identifier. Open the file linux-4.3.3/Makefile with the edit tool.
Change EXTRAVERSION to **-CS460-trad**

Now, we are ready to build the kernel. In the console, go to the kernel/linux-4.3.3 directory.

```
make CC="ccache gcc"  -j 6
```

In another window, start top and watch the compiler run.

```
sudo make modules_install
```

Now, we can install the kernel:

```
sudo cp -v arch/x86_64/boot/bzImage /boot/vmlinuz-4.3.3-CS460-trad
sudo cp -v System.map /boot/System.map-4.3.3-CS460-trad
sudo mkinitcpio -k 4.3.3-CS460-trad-ARCH -c /etc/mkinitcpio.conf -g /boot/initramfs-4.3.3-CS460-trad.img
sudo ln -sf /boot/System.map-4.3.3-CS460-trad /boot/System.map
```

Let's make the boot loader aware of our kernel and make a backup entry!

Edit /etc/grub.d/40_custom

```
menuentry “CS460_Linux” {
        set root=(hd0,1)
        linux /vmlinuz-4.3.3-CS460-trad root=/dev/sda3 rw
        initrd /initramfs-4.3.3-CS460-trad.img
}

menuentry “CS460_Backup” {
        set root=(hd0,1)
        linux /vmlinuz-linux root=/dev/sda3 rw
        initrd /initramfs-linux.img
}

sudo grub-mkconfig -o /boot/grub/grub.cfg  # on some distros this is update-grub
```

If you get time errors (clock skew), run:
`find . -name '*' | xargs touch`

Reboot and select Arch Linux CS460 Custom!

Run **ls -al /boot** to see your new kernel! Run **uname -a** to see the running kernel version.

If it does not restart properly, use the Machine | Reset menu option and choose the Fallback Kernel.

Adding a system call!

Let's add a simple system call that will print “HELLO WORLD” to the logs (/var/log/messages) and return a value of 42 to the user program.

Create a new file (**CS460_Syscalls_PUNetID.c**) in the directory **~cs460/kernel/linux-4.3.3/kernel**

The file must contain:

```
#include <linux/linkage.h>
#include <linux/kernel.h>
#include <linux/syscalls.h>

SYSCALL_DEFINE0(helloworld)
{
     printk(KERN_EMERG "HELLO WORLD!");
     return 42;
}
```

Edit the **Makefile** in that directory and add CS460_Syscalls_PUNetID.o to the end of the **obj-y** list.

Edit arch/x86/entry/syscalls/syscall_32.tbl

`376 i386 helloworld sys_helloworld`

Edit arch/x86/entry/syscalls/syscall_64.tbl

`325 common helloworld sys_helloworld`

Add :
Edit **~/kernel/linux-4.3.3/include/linux/syscalls.h** (this is the non-architecture specific version of the file). Add the following line:

asmlinkage long sys_helloworld(void);

before the final #endif

Edit kernel/sys_ni.c

`cond_syscall(sys_helloworld);`

Build and install the kernel as described above. Reboot into the new (Custom) kernel.

Add the file **/usr/include/CS460.h** which contains:

```
#include <linux/unistd.h>
#include <sys/syscall.h>
#define sys_helloworld 325
#define helloworld() syscall(sys_helloworld)
```

This header file exposes the helloworld system call to the user. Normally, this code would be in a shared library (such as glibc) but for simplicity we will just put it in a header file.

Write a test case. In your home directory, create the file **CS460_TestSscalls_PUNetID.c**
This file should contain:

```
#include <stdio.h>
#include <CS460.h>

int main(void)
{
    int c;
    c = helloworld();

    printf("System call returned %d\n", c);

    return 0;

}
```

gcc -o CS460_TestSyscalls_PUNetID CS460_TestSyscalls_PUNetID.c -g
./CS460_TestSyscalls_PUNetID

Run your new executable. **Be sure you have rebooted to the Custom kernel since installing the new kernel!** To see the hello world message in the logs run

$ dmesg | tail

NOTES:

You may need to resize the screen after a reboot.

QUESTIONS

Use man, Google, and your book to answer the following questions. Submit these answers as a GoogleDoc (CS460_InClass_PUNetID) by Monday, May 9, 4:45 pm. These questions are worth a homework grade.

1. Describe what SYSCALL_DEFINE0 does. Where is this defined?
2. Describe what syscall() does. Where is it defined?
3. What is sys_ni.c used for?
4. What is the file initramfs.img used for?
5. What is the System.map file used for?
6. grep helloworld System.map. What do you find? What does the sys_helloworld entry mean?
7. Reboot using the original kernel (Arch Linux) and rerun **CS460_TestSyscalls_PUNetID.** What do you see? What does dmesg | tail tell you? What is remarkable about this?

References

https://www.kernel.org/doc/Documentation/adding-syscalls.txt
https://shanetully.com/2014/04/adding-a-syscall-to-linux-3-14/
https://tssurya.wordpress.com/2014/08/19/adding-a-hello-world-system-call-to-linux-kernel-3-16-0/
https://wiki.gentoo.org/wiki/GRUB2_Quick_Start

http://www.cs.columbia.edu/~hgs/teaching/os/hw3.html
http://lxr.free-electrons.com/source/?v=3.1
http://users.sosdg.org/~qiyong/lxr/source/?v=3.x

page 93 in your book.

http://enzam.wordpress.com/2011/03/26/how-to-add-a-system-call-in-linux-kernel-ubuntu-os/
http://tldp.org/HOWTO/html_single/Implement-Sys-Call-Linux-2.6-i386/

https://wiki.archlinux.org/index.php/Kernel_Compilation_From_Source
https://wiki.archlinux.org/index.php/Arch_Linux_VirtualBox_Guest