CSCI 2461, Computer Networking 3 - Linux

Week 07 (Rev. 0)

Wednesday, February 21, 2018

Intro

Due Homework

Weeks 1-6

Late class 1-5 submissions were due last week; I hope you were able to catch up as we are moving forward now. If not, sign up for a 1-on-1. You should be booting from your USB drive now (plug it in and boot, now)

- How did your collaboration go in the Github discussions?
- How did the peer script review on GitHub go?
- Were you able to build your bootable Debian drive?
- Did you get git installed on your bootable drive and are you using it for your homework?

Reading Review

How were the readings, anybody deploy QEMU?

- Installing Debian from Scratch
- QEMU Bootstrap
- Cross Debootstrap

Opening

Class Plan

Our class today is going to start with reviewing `dmesg` as we walk through Chapter 5 "How the Linux Kernel boots" then we are going to make sure your USBs with Persistence boot and look at Chapter 6 "How User Space Starts." Then we're going to apply your bootable USB and explore the physical layer of networking.

- Review Chapter 5 "How the Linux Kernel Boots"
- Lab: Grub Exercise (5.5, pg98)
- Finish Chapter 5
- Lab: Boot your Linux USBs with Persistence
- Review Chapter 6 "How User Space Starts"
- Networking Lab: Physical Layer

1-on-1s

During the class lab time, if you haven't had a 1-on-1 with me regarding your class progress or have questions, please clearly block write your name in the second sign-in book.

Equipment & Lab

It may go without saying, but please do not walk off with the equipment I bring to class and please return it in the same way it was provided to you. This includes wrapping the cables up neatly and tying them up, and returning your workstation to an operating state.

Get Ready

Setup your workstation for class.

- Boot into Linux
- Wean yourself from the VM
- Open your book to Chapter 5

Ch 5: Linux Kernel

Open Ch5 and terminal

Open "How Linux Works" to Chapter 5

- Open a root terminal, it should start with `#`
- `su - root`
- or `sudo bash` (this is bad)
- then run `dmesg | less` and start reviewing the output

Startup Process

Identify aspects of each of these

1. CPU inspection
2. Memory inspection
3. Device bus discovery
4. Device discovery
5. Networking
6. Root filesystem mount
7. User space start

Kernel Log Sources

Other kernel log sources

- `less /var/log/kern.log`
- `less /var/log/messages`

Boot Parameters

Analyze the Kernel Boot Parameters

- `cat /proc/cmdline`

Before the Kernel

1. The PC BIOS or firmware initializes the hardware and searches its boot-order storage devices for boot code.
2. Upon finding the boot code, the BIOS/firmware loads and executes it. This is where GRUB begins.
3. The GRUB core loads.
4. The core initializes. At this point, GRUB can now access disks and filesystems.
5. GRUB identifies its boot partition and loads a configuration there (/boot/grub/grub.cfg)

Before the Kernel (cont)

6. GRUB gives the user a chance to change the configuration.
7. After a timeout or user action, GRUB executes the configuration in `grub.cfg`
8. GRUB may load additional modules in the boot partition.
9. GRUB executes a boot command to load and execute the kernel

Boot Loaders

What is GRUB? What is LILO?

- Interaction between BIOS/UEFI and the direct hardware
- The last direct iteraction before the Kernel

Lab: GRUB Exercise (5.5)

Open your book to Section 5.5 (Page 98), and reboot your USB or Virtual Machine and look at the bootloader options. Move the selector up and down (arrow keys) to interrupt the boot and stop the counter.

- Press `e` and compare the GRUB options
- Lookup the individual items and guess at what they do.
- Exit out of that window and at the `grub>` prompt type `ls`
- type `ls -l` at the `grub>` prompt
- `echo $root`

GRUB Configuration

Reboot into your USB or VM

- Open a terminal window
- `cd /boot; ls; cd grub`
- `less grub.cfg`

grub-mkconfig

Open the manual page for grub-mkconfig

- `man grub-mkconfig`
- `grub-mkconfig -o /boot/grub/grub.cfg`

grub-install

Open the manual page for grub-install

- `man grub-mkconfig`
- `grub-install /dev/sda`

Chainloading (5.7, pg 106)

Chain loading Windows, addition to `grub.cfg`

‘ menuentry “Windows” {

```
insmod chain
insmod ntfs
set root=(hd0,3)
chainloader +1
```

} ‘

Networking: Physical

Lab Time

Hands-on Lab

- How to correctly unplug and plug in an ethernet cable
- Physical and Data Link Layers
- Router/Switch/Firewall
- Wireless Uplink

Ch 6: User Space

init

1. init
2. Essential low-level services such as udevd and syslogd
3. Network configuration
4. Mid to high-level services (cron, printing)
5. Login prompts, GUIs, and other high-level applications

init saga

There are four main init daemons:

- sysvinit "System Five Initalization" (UNIX philosophy)
- systemd (new defacto)
- Upstart (Debian abandoned, ChromeOS, often optional)
- OpenRC (Gentoo, Alpine Distributions)

Sys V Runlevels

Open a terminal and run

```
who -r
```

What is your runlevel?

What is your init?

Find your init system

```
ls /usr/lib/systemd /etc/systemd
```

```
ls /etc/init
```

```
ls /etc/inittab
```

systemd

systemd initialization steps:

1. systemd loads conf
2. systemd loads default.target.
3. systemd determines dependencies (default)
4. systemd runs dependencies
5. systemd reacts to system events (uevents)

systemctl list-units

Unit types: Service units, Mount units, Targets (Groups)

In your expanded terminal, run:

```
sudo systemctl list-units
```

systemd config

systemd configuration location:

```
sudo systemctl -p UnitPath show
```

systemctl list-dependencies

Open a terminal and expand the screen and run:

```
sudo systemctl list-dependencies
```

systemd-analyze critical-chain

Identifying boot slowness, failed processes and warnings:

```
sudo systemd-analyze critical-chain
```

udev

Dynamic device management

```
man systemd-udevd.service
```

```
man udev
```

Reading Unit Files

Unit files are equivalent to .ini files in Microsoftese

```
cd /usr/lib/systemd/; ls
```

```
cd system; ls | less
```

Service status

What output does this give you?

```
systemctl status sshd.service
```

Enabling services

Enable sshd.service:

```
systemctl enable sshd.service
```

What output did it give you? Did it make a link?

journalctl

Check journal/log entries for sshd.service:

```
journalctl _SYSTEMD_UNIT=sshd.service
```

Look at the rest of the logs:

```
journalctl -xfe
```

shutdown (6.7)

Shutting down a system is an invocations of init followed by a halt (-h).

```
shutdown -h now
```

is (mostly) equivalent to

```
init 0
```

reboot

Rebooting is similar

`reboot` runs `shutdown -r now`

vs

`init 6`

init ramfs (initrd)

As you saw in the boot process, each system loads an initrd file that loads the prerequisits for a system booting

```
$ mkdir /tmp/initrd
```

```
$ cd /tmp/initrd
```

```
$ zcat /boot/initrd.img[tab] | cpio -i
--no-absolute-filenames
```

Single User Mode

Single user mode is `init 1` and includes minimal recovery services. In most cases it is preferable to use a bootable CD/USB to access the system.

When recovering from a separate device, you'll use the `chroot` method to load libraries and interact with the system as if it were live.

Homework

Reading

- Read and apply Chapter 6 “How User Space Starts”
- Read and write a min 500 word response to An Industry Guide to Becoming a Software Engineer
- Read, review, and run peer scripts from GitHub in your VM.
- What is the Devuan Debian Fork?

Lab

- Customize your USB and document how you did it, using Markdown formatting. Post this to your week7 repository.
- Write a script that determines your current run level, and returns the value. Post the executable script to your week7 repository as week7-runlevel.sh
- Write a unit file for a service, load it, unload it, and return the status. Post this unit file to your week7 repository with the proper naming of service files.
- Collaborate with your peers on the assignments.

Tschuss!