CS444 Homework 3

Lilliana Watts, Nicholas Skinner, Yong Ping Li

May 24, 2018

Design Plan

Plans for this assignment were to research and implement an encrypted block device. For this, we will need to look into, and research drivers, encryption, and block I/O. to implement this we will create our files, and then test them against the kernel and find some form of logging to ensure we are doing everything correctly.

Writeup Questions

A. What do you think the main point of this assignment is?

• The main point of this assignment seems to be to learn how to use a poorly documented API, for this assignment, it is the Kernel Crypto API. Another addition to the main point would be teaching us the important parts of block devices, as well as cryptography, we learned about what goes into these drivers to make them function.

B. How did you personally approach the problem? Design decisions, algorithm, etc.

• Our team's approach to this project was to research as much as we could about Block devices in the linux kernel, we had used chapter 16 of the Linux Device drivers as the basis for our device driver. [1]The next step was to understand how to use the crypto that is buit into linux. After that, we then needed to combine the two prior steps into our group's device driver.

C. How did you ensure your solution was correct? Testing details, for instance.

• To test and ensure our solution had been working properly, we had
used printk statements throughout our code to ensure that we could see how it was working with both encrypted as well as unencrypted data.

Testing procedure for ensuring correctness of this problem included the following:

• Clone the kernel code to a fresh instance of a repo: git clone git://git.yoctoproject.org/linux-yocto-3.19

• Navigate into the Git repo folder: cd linux-yocto-3.19 • Checkout the appropriate version of our kernel: git checkout v3.19.2

• Apply the patch file : git apply assign3.patch

• Source the proper environment for the kernel: source /scratch/files/environment-setup-i586-poky-linux.csh

• Copy the core into the linux folder: cp /scratch/files/core-image-lsb-sdk-qemux86.ext4 .

• Copy the config file into the linux folder: cp /scratch/files/config-3.19.2-yocto-standard .config

• Go into the menuconfig, and then set sbd into a module:

make menuconfig

Go to device drivers

Enter block drivers

go to the SBD block driver

Press M

Save the new configuration, and then exit

• After setting the menuconfig, now we will make the VM: make -j4 all

• next we will start the VM with networking enabled: qemu-system-i386 -gdb tcp::5507 -S -nographic -kernel bzImage-qemux86.bin -drive file=core-image-lsb-sdk-qemux86.ext4,if=virtio -enable-kvm -usb -localtime –no-reboot –append ”root=/dev/vda rw console=ttyS0 debug”.

• Login with the root account, there is no password, so just type: root • go to the base directory: cd .. cd ..

• Copy the module into the base directory using SCP:

```
scp skinnern@os2.engr.oregonstate.edu:
/scratch/spring2018/group7_new/linux-yocto-3.19/drivers/block/sbd.ko .
```

• start up the module: insmod sbd.ko

• Create the ext2 file directory: mkfs.ext2 /dev/sbd0

• Create a directory to mount to, if it does not exist already: mkdir /mnt

• We will then mount it: mount /dev/sbd0 /mnt

• Enter some text into a file that is mounted on the dir: echo ”Test this stuff” ¿ /mnt/test

• Next, try to find that test on the device by using grep: grep -a ”Test” /dev/sbd0

• If nothing is returned for the previous step, then there was a successful encryption, if there was ne encryption that had occurred, then you would be able to find the text with using grep. You can also try any

other things that are put into the file, but as long as they do not show up in grep, we will have encryption.

D. What did you learn?

• Document, document document. Documentation is very important when creating a system, or any segment of code for that matter. We also learned that we are able to search for programs that had been created for other operating systems with similar architecture for code for general ideas on how to implement the program that we are trying to write. We also through some trial and error learned how to troubleshoot what devices and drivers are doing through debugging and printk statements.

GITHUB LOG

Detail	**Author**	**Description**
77f914a	Nicholas	Start Writeup
49f9b37	Nicholas	Update with Patch
9df8749	Nicholas	Update Writeup
cc9ed65	Nicholas	Fix Patch File

WORK LOG

Date	**Name**	**Hours**	**Description**
5/14	Nicholas	1	Setup Document
5/20	Nicholas	2	Research Block
5/21	Nicholas	4	Implementation
5/22	Nicholas	3	Fixing Encryption
5/22	Nicholas	1	Testing
5/24	Nicholas	2	Finish Writeup

REFERENCES

[1] Jonathan Corbet, Alessandro Rubini, and Greg Kroah-Hartman. *Linux Device Drivers*. O'Reilly Media, Inc., 3 edition, 2005.