Homework 3

Homework 3 Writeup CS444 Spring2018 Group 37

Brandon Mei, Brian Huang

1 Implement Design

To begin the problem we gone through the trial and error by starting the basic block driver and slowly implement the encryption system through looking online resources. The crypto handles the IO requests. The final result was to include a diver mounted device with utilize the implemented block driver and looking in the encryption block device where device module need to made in order to make a device driver with the linux Crypto API. This will ensure a valid encryption files.

2 Control Log

Detail	Author	Description
455b948	Brandon Mei	Adding the crypto in init() and transfer() function
fb027ec	Brian Huang	added document and implement the sbd
f50dec6	Brian Huang	implement the searcher
31b31b	Brandon Mei	implement the patch files and update the inserter
d1500e	Brandon Mei	Update the writeup and finalize.

3 Work Log

The work log began with doing some research on what is the assignment was asking. We focus using LDD3 and searching around the web for additional information. After understanding what is exactly what was needed to be done, We got a basic block driver (Sbull.c) started to go through many trial and error by setting up the whole system. Then from building the module, we move the module to the VM through the SCP and installing the module with insmod. After that we mount with the fdisk and making a filesystem with the mkfs.ext2, and mounting it using mount. This is the list of commands that can be found below of the steps taken to get there.

- Getting 3 files from patch
- make -j4 all (Should make the .ko file)
- Qemu - Remove of -net:none
- scp to get the module (.ko file) into the VM
- insmod sdb.ko
- fdisk /dev/sbd0p1 and make a partition
- mkfs.ext2 to make the file system (mkfs.ext2 /dev/sbd0p1)
- mount /dev/sbd0p1 /test

Finally, we got everything is working. We can start with actually implementing the Crypto into the device. After, going through trial and error, the Crypto is implemented and files can be placed into the filesystem.

4 QUESTIONS

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

The main point of this assignment was to understand and develop the linux block devices, Crypto API, and modules. We thought the purpose of this assignment was to create an understanding of how to bring the modules onto the Virtual Machine in order to get establish a general foundation of importing Linux modules.

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

The way we approach his assignment is going through the problem by trial and error starting with the basic block diver with no encryption in it. After that, we implemented the encryption system through various online resource and putting it together in sections. When there was an error, we revert to previous commits. Then most of the crypto can be found in the transfer function that handles the I/O request. It is the algorithm that checks the block by block to see if it should encrypt, read or write, and decrypt. While going through the check, it either performs the encryption or decryption with the crypto_chiper_encrypt_one() or crypto_chiper_decryp_one(). Also there was some crypto instantiation in the init() function.

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

We decided to write a script that basically does all of the require steps to the solution and set up everything including unpatching, make the .ko file, begain the VM, install the module, using SCP to bring the module, partition the drive, make file systen, and mount it with the device.

4.4 What did you learn?

We learn from this assignment was on Linux devices, encrypting the blocks, and write the block drivers. Also we learned how to SCP the files into the VM. We learn the partition the dive and make the file system and mount the drives. Then learn the linux crypto API.

4.5 How should the TA evaluate your work? Provide detailed steps to prove correctness.

As the steps we took above going through learning from trial and error. The commands we used is by getting 3 files from the patch and -j4 all to make the .ko files. After that the SCP to get module of .ko file into the VM. Then insmod sbd.ko. fdisk /dev/sbd0p1 and make the partition. Mkfs.ext2 is to make the file system and finally mount /dev.sbd0p1 /test. We also added some print statements to figure what is doing so we know it was done correctly.

REFERENCES

[1] J. Debian, "qemu-doc - qemu emulator user documentation," https://manpages.debian.org/jessie/qemu-system-x86/qemu-system-x86_64.1.en.htm May 2016, (Accessed on 4/15/2018).

[2] R. Love, "Linux kernel development," Pearson Education, 2010, Accessed on 5/6/2018.