Project 3: Encrypted Block Device

Courtney Bonn, Isaac Chan

Group #39

Abstract

I. Design Plan

Our plan for project 3 is to first get a basic ramdisk block driver up and running on the VM before we begin any encryption. We first read through the chapter on block drivers in Linux Device Drivers which provided a simple block driver called *sbull*. Because this driver was based off an older version of the Linux Kernel, we did a bit more research for ramdisk drivers that might be more recent.

This research led us to a comment on the book which led to a more recent version of the ramdisk driver. In a post by Pat Patterson, we found the ramdisk driver that would work with kernel 2.6.31. Because we are using kernel 3.19, we still weren't sure if this would work. In the comments of the post, which can be found at http://blog.superpat.com/2010/05/04/a-simple-block-dri there was a recent comment by Sarge that updated one line of the code to make it work for kernels 3.15 and later.

With this base code, we were able to compile the kernel and boot the VM. Once in the VM, we were able to run a series of commands that loaded the ramdisk module, made a filesystem, mount the module, and then unmount and remove the module. The commands were:

1) scp (ONID Username)@os2.engr.oregonstate.edu:/scratch/fall2017/39/linux-ramdisk/drivers/block/ramd.ko .
2) insmod ramd.ko
3) fdisk /dev/sbd0
4) Command: n
5) Partition type: p
6) Partition number: 1
7) First sector: [press enter for default value]
8) Last sector: [press enter for default value]
9) Command: w
10) mkfs /dev/sbd0p1
11) mount /dev/sbd0p1 /mnt
12) echo Hi >/mnt/file1
13) cat /mnt/file1
14) ls -l /mnt (just to view the file was created)
15) umount /mnt
16) rmmod ramd.ko

Now that we were able to successfully run an unencrypted ramdisk, our next step was to begin the encryption section.

Once the encryption step was done, we were able to check if the encryption was successful. After creating the *file1* file, we can check in the raw device if there's anything that matches the *file1* string with *grep -a 'Hi' /sbd0p1* which should return nothing if it's being encrypted and decrypted correctly. After this we can unmount and remove the module as before.

II. Version Control Log

III. Work Log

Date	Time	Person	Event
November 1, 2017	9:20am	Courtney	Started HW3 LaTeX file
November 2, 2017	7:15pm	Courtney and Isaac	Begin researching Ram Disks
	8:30pm	Courtney	Set up basic block driver
	9:30pm	Isaac	Building and compiling ramd.c
November 3, 2017	9:00am	Courtney	Successfully boot VM and loaded the Ramd.c module, made the filesystem, mounted it, verified it worked, unmounted and removed module
November 5, 2017	1:30pm	Courtney	Began researching Crypto API, focusing on how to use module parameters for the key
November 8, 2017	2:00pm	Isaac	Start implementing crypto functionality
November 11, 2017	5:45pm	Courtney	Began testing crypto code
	7:00pm	Isaac	Running crypto code
	8:45pm	Isaac	Finished crypto code
	8:45pm	Isaac	Wrapping up assignment documentation and start generating patch file

IV. Write Up

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

   The main point was to implement a block device in our virtual machine. The block device encrypted when read was called and decrypted when write was called. It replaced the existing RAM disk driver.

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

   We started with an block device without encryption and ensured that worked before attempting encryption. When beginning encryption, we looked into the Crypto API for methods we could utilize. Essentially the algorithm is as follows:

   a) Initiate the Crypto API
   b) Set the cipher key to either the default or the supplied argument
   c) Encrypt and decrypt one byte at a time, as reads and writes are called
   d) Free the cipher when finished

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

   Initially we had the block device print out each byte that was encrypted/decrpyted, but it proved to be too much to

handle. We decided to create a file in the filesystem, then search for the string in the raw device. If it wasn't found, it means that the string is being encrypted correctly.

4) What did you learn?

   We learned a lot about block devices in Linux and how they handle IO. We also learned how to use the Crypto API.

V. Run Instructions

To apply the patch:

1) cd $linux_src/drivers/block
2) patch > (dir where this is)/linux.patch

Then source the environment and start qemu. Then the block device can be run as described above.