Securing the system and boot process: Including encrypting systems, Logical Volume Management (LVM), trusted execution environment, secure enclave, Supervisor Mode Access Prevention, Intel Management Engine (ME) and AMD Platform Security Processor (PSP)

Adam Boult (www.bou.lt)

April 30, 2025

Contents

Preface

This is a live document, and is full of gaps, mistakes, typos etc.

Part I

Encryption and early user space

Chapter 1

Encrypting a partition not used in the boot process

1.1 Introduction

1.1.1 dm-crypt

Short for Device Manager enCRYPTion.

/etc/crypttab

encrypt a partition. will prompt for password

1.1.2 Linux Unified Key Setup (LUKS)

```
cryptsetup luksFormat /dev/<device>
cryptsetup luksFormat /dev/sda1
```

can check with:

```
cryptsetup open /dev/sda1 name
```

Creates in

```
/dev/mapper/name
```

Can close:

```
cryptsetup close name
```

When open can eg format it:

```
mkfs -t ext4 /dev/mapper/name
```

1.1.3 Sort

Public Key Cryptography Standards (PKCS)#11 tokens Fast IDentity Online 2 (FIDO2) tokens

dm-verity dm-integrity

/proc/crypto

/etc/initcpio/hooks/openswap

Chapter 2

Encrypting the swap partition

2.1 Introduction

2.1.1 Introduction

encrypting swap: if don't need suspend to disk, can use a random password each time. otherwise can't do that.

Chapter 3

Early user space and encrypting the root partition

3.1 Introduction

3.1.1 Introduction

Doing this encrypts the root partition.

The boot partition must remain unencrypted for this.

unified kernel image (UKI) + can be loaded by UEFI + contains all of: uefi stub loader (eg systemd-stub); kernel command line; microcode; initramfs image; kernel image; splash screen

/etc/mkinitcpio.conf /etc/mkinitcpio.d/

/boot/initramfs-

dracut (context of initrd)

initramfs + can be used to make logical volumes root. also used for encrpyted file systems as discussed later.

logical volume manager (LVM)

dm-crypt is kernel thing + cryptsetup and cryptmount are frontends for user space + dm-crypt encrypts a loop file? different to full disk encryption?

page on early user space/initramfs + after kernel is loaded, before init process is started + loads a temporary root file system + initrd and initramfs are alternative implementation * initrd 1. mounts /dev/ram 2. runs /linuxrc 3. when done prep is done so runs /sbin/init * initramfs 1. newer + used for logical partitions and encrypted drives

mkinitcpio on the early user space

done after loading kernel, before init process + used for eg encrpying swap disk and allowing reload on boot. + but not for encrpyting root drive right? if encrypted file system, how can we load the kernel? main root partition must be unencrpted during boot right?

* encrypting boot partition using GRUB * plain dm-crypt * dm-crypt + LUKS 1. LUKS2 encrpyed root partition (/), unencrypted (/boot) partition * approaches using LVM: 1. LUKS on LVM 2. LVM on LUKS

Part II

Logical volume management (LVM) and encrypting the boot partition

Chapter 4

Logical Volume Management (LVM)

4.1 Introduction

4.1.1 Introduction

device mappers map physical block devices to virtual block devices

if doing this (or encrpytion of root as later) need to have separate boot partition. already need this for uefi with GPT

Part III

Securing the boot partition

Chapter 5

Getting UEFI Secure Boot working with LVM

5.1 Introduction

5.1.1 Introduction

Chapter 6

Trusted Platform Module (TPM)

6.1 Introduction

6.1.1 Introduction

Part IV

Redundant array of independent disks (RAID) in software

Chapter 7

Software RAID

7.1 Introduction

7.1.1 Introduction

+ software raid + raid levels etc. i think this is somewhere already?

7.1.2 Software RAID

Multiple Devices ADMin (mdadm).

Take physical disk drives, create logical disk drives.

7.1.3 Striping

A single file is spread over multiple disks.

Can be done at bit/byte/block level.

7.1.4 Mirroring

Same data on multiple drives

7.1.5 Parity

Used for error detection.

In protocol for sending/saving we say that all bits must be even (eg 1100) (or odd)

For given bit of info, we add parity bit to guarantee that bit is indeed even/odd. 1100 becomes 11000; 1000 becomes 10000

Cannot correct errors, just detect them. only detects if odd number of errors.

Parity can be stored on dedicate disk, distributed.

alternative to parity bit: hamming code

7.1.6 RAID levels

RAID 0: Uses striping across disks, but no redunency. allows for improved read/write times. if any drive fails, all fail RAID 1: Data written identically to two drives. read times increased, as with raid 0, due to mirroring. writing is slower. no parity or striping RAID 2: bit level striping and hamming code. rarely used RAID 3: rarely used. byte level stripping. Dedicated parity disk RAID 4: dedicated parity disk RAID 5: block level striping, distributed parity RIAD 6: double distributed parity

7.1.7 Other RAID

snapraid unraid raid 10 hot spare

Part V

ZFS, BTRFS and BcacheFS

Chapter 8

ZFS

8.1 Introduction

8.1.1 Introduction

zfs pools, vdevs

8.1.2 RAID-Z

RAID-Z: Under ZFC, similar to RAID 5

Chapter 9

BTRFS

9.1 Introduction

9.1.1 Introduction

Chapter 10

BcacheFS

10.1 Introduction

10.1.1 Introduction

Part VI

Non-SATA storage: USB including usbutils (lsusb list usb devices)

Part VII

Non-SATA storage: USB including pciutils (lspci list pci devices)

Part VIII

Non-SATA storage: USB including usbutils (lsusb)

Part IX

More on Pluggable Authentication Modules (PAM)

Chapter 11

Encrpytion and hardware authentication with Pluggable Authentication Modules (PAM)

11.1 Introduction

11.1.1 Introduction