# 改进Linux磁盘加密性能

## 环境配置

### 安装ubuntu虚拟机

account： kerneltest

password : 123

### 安装ssh

1. 在刚装好的ubuntu系统上执行以更新依赖配置

```
sudo apt-get update
```

2. 安装open-ssh服务

```
sudo apt-get install openssh-server
```

3. 查看是否启动ssh服务

```
netstat-tlp
```

4. 远程电脑连接ssh

```
ssh -l kerneltest ip地址
```

### 常用命令

- 查看linux 所有版本信息 lsb_release -a

### 关键词

dm-crypt ， device mapper

参考网址

https://www.kernel.org/doc/

# Device Mapper 机制

Device Mapper 是 Linux2.6 内核中支持逻辑卷管理的通用设备映射机制，实现用于存储资源管理的块设备驱动提供了一个高度模块化的内核架构。

**图1 Device Mapper 的内核体系架构**

内核中通过一个个模块化的 target driver 插件实现对 IO 请求的过滤或者重新定向等工作，当前已经实现的 target driver 插件包括软 raid、软加密、逻辑卷条带、多路径、镜像、快照等。

Device mapper 进一步体现了在Linux内核设计中策略和机制分离的原则，将所有与策略相关的工作放到用户空间完成，内核中主要提供完成这些策略所需要的机制。

Device mapper 用户空间负责配置具体的策略和控制逻辑，逻辑设备和哪些物理设备建立映射，怎么建立映射关系，具体过滤和重定向 IO 请求的工作由内核中相关代码完成。

因此整个 device mapper 机制由两部分组成 —— 内核空间的 device mapper 驱动、用户空间的 device mapper 库以及它提供的 dmsetup 工具。

# 内核部分

相关内核源码在 driver/md/目录中，其代码可以分为实现 device mapper 内核中基本架构的文件和实现具体映射工作的 target driver 插件文件两部分

**重要概念**

Device mapper 在内核中作为一个块设备被驱动的，它包含三个重要的对象概念—— mapped device 、映射表、target device 。

- Mapped device : 一个逻辑抽象，理解成为内核向外提供的逻辑设备，通过映射表描述的映射关系和 target device 建立映射。
- 映射表 : 从 Mapped device 到一个 target device 的映射表由一个多元组表示，该多元组由表示 mapped device 逻辑的起始地址、范围和表示在 target device所在物理设备的地址偏移量以及 target 类型等变量组成
- Target device : mapped device 所映射的物理空间段，对mapped device 所表示的逻辑设备来说，就是该逻辑设备映射到的一个物理设备

Device mapper 中这三个对象和 target drive 插件一起构成了一个可迭代的设备树。在该树形结构中顶层根节点是最终作为逻辑设备向外提供的 mapped device，叶子节点是 target device 所表示的底层物理设备。

最小的设备树由单个 mapped device 和 target device 组成。每个target device都是被mapped device 独占的，只能被一个 mapped device使用。 一个mapped device 可以映射到一个或者多个 target device 上，而一个 mapped device 又可以作为它上层 mapped device 的 target device 被使用，该层次在理论上可以在 device mapper架构下无限迭代下去。

图2 中我们可以看到 mapped device1 通过映射表和 a、b、c 三个 target device 建立了映射关系，而 target device a 又是通过 mapped device 2 演化过来，mapped device 2 通过映射表和 target device d 建立映射关系

# 用户空间部分

Device mapper在用户空间相对简单，主要包括device mapper库和dmsetup工具。Device mapper 库就是对ioctl、用户空间创建删除device mapper逻辑设备所需必要操作的封装，dmsetup是一个提供给用户直接可用的创建删除device mapper设备的命令行工具。

用户空间主要负责如下工作：

1. 发现每个 mapped device 相关的 target device
2. 根据配置信息创建映射表
3. 将用户空间构建好的映射表传入内行人，然内行人构建该 mapped device 对应的dm_table结构；
4. 保存当前的映射信息，以便未来重新构建

# dm-crypt 多功能Linux磁盘加密工具

## 任务

改进dm-crypt使用cipher的习惯，改每次512字节为更多

## 关键提示

把一个512-byebuffer改为使用scatter-gatherlist (sg)

## dm-crypt/cryptsetup 解释

dm-crypt 是 Linux 内核提供的一个磁盘加密功能，而 cryptsetup 是一个命令行的前端（通过它来操作 dm-crypt）

dm-crypt 在 Linux Kernel 2.6 的早期版本就被整合到内核中，至今有10多年了。

## dm-crypt 的功能和特色

### 多种加密格式

1. LUKS (Linux Unified Key Setup)

   最常用的一种模式

2. Plain

   适用于技术大佬。目前不关心这个模式

3. loop-AES

   一款比较陈旧的 Linux 磁盘加密工具。dm-crypt 对它提供了支持。应该不会用的模式

4. TCRYPT

   在 crypsetup 的 1.6.0版本后开始提供对 TrueCrypt 加密盘的支持 。

### 无需额外安装软件

dm-crypt 早已被整合到 Linux Kernel 中，因此无需额外安装。

它的命令行前端（cryptsetup），大部分主流发行版都会内置 cryptsetup 的软件包

### 与 LVM 关系

LVM(Logical Volume Manager) 是 Linux 内核提供的另一个很有用的工具。创建分区，合并分区，添加分区。

LVM 和 dm-crypt 都是基于 Linux 内核的 device mapper 机制。

## 预备知识

介绍一下操作以备用

- 如何对硬盘分区 fdisk
- 如何创建文件系统 mkfs.ext4 等
- 如何挂载/卸载文件系统 mount umount
- 如何显示已挂载的文件系统 df

## cryptsetup 命令行概述

note that : 需要使用管理员权限(sudo/su) ，来运行 cryptsetup 相关命令。

### 查看版本

使用命令 cryptsetup --version ,如果需要下载命令 sudo apt install crypsetup-bin 因为 dm-crypt/cryptsetup 的某些新功能只有新版本才提供，如果使用需要确定版本号是否支持。

### 查看新能指标

使用命令 cryptsetup benchmark 查看dm-crypt/crypsetup 针对不同"加密算法" 和 "散列算法的性能"

### 创建加密盘

dm-crypt/cryptsetup 只能用来打开 TrueCrypt 或 VeraCrypt 的加密盘，但无法创建。

创建（格式化）LUKS 加密盘的命令

```
cryptsetup 参数 luksFormate 物理设备或逻辑设备
```

运行该命令后，首先警告你，格式化会导致原有数据被覆盖。如果你确实要格式化，需要输入[大写] YES 确认，然后提示输入两次密码

# 加密

根据密钥类型不同将现代密码技术分为两类：一类是对称加密（秘密钥匙加密）系统，另一类是公开密钥加密（非对称加密）系统。

## 对称密码体制

对称密码体制是一种传统密码体制，也称为私钥密码体制。在对称加密系统中，加密和解密采用相同的密钥。因为加解密密钥相同，需要通信的双方必须选择和保存他们共同的密钥，各方必须信任对方不会将密钥泄密出去，这样就可以实现数据的机密性和完整性。对于具有n个用户的网络，需要n(n-1)/2个密钥，在用户群不是很大的情况下，对称加密系统是有效的。但是对于大型网络，当用户群很大，分布很广时，密钥的分配和保存就成了问题。

在对称密钥密码体系中，也许对不同的信息使用不同的密钥，但都面临密钥管理的难题。由于每对通讯方都必须使用异于他组的密钥，当网络成员的数量增加时，密钥数量成二次方增加。更尴尬的难题是：当安全的通道不存在于双方时，如何建立一个共有的密钥以利安全的通讯？如果有通道可以安全地建立密钥，何不使用现有的通道。这个矛盾是长年以来密码学无法在真实世界应用的阻碍。

## 非对称密码体制

非对称密码体制也叫公钥加密技术，是针对私钥密码体制的缺陷被提出来的。相对于对称密钥密码体系，最大的特点在于加密和解密使用不同的密钥。由于加密和解密是相对独立的，加密和解密会使用两把不同的密钥，加密密钥向公众公开，谁都可以使用，解密密钥只有解密人自己知道，非法使用者根据公开的加密密钥无法推算出解密密钥，故其可称为公钥密码体制。假如一个人选择并公布了他的公钥，另外任何人都可以用这一公钥来加密传送给那个人的消息。私钥是秘密保存的，只有私钥的所有者才能利用私钥对密文进行解密。公开密钥系统通常是复合式的，内含一个高效率的对称密钥算法，用以加密信息，再以公开密钥加密对称钥匙系统所使用的钥匙，以增进效率。

# 使用dm-crypt建立加密设备

要创建作为加密设备装在的文件系统，有两种选择：

- 建立一个磁盘映像，然后作为回送设备加载；
- 使用物理设备。

无论那种情况，除了在建立和捆绑回送设备外，其它操作过程都是相似的。

1. 建立回送磁盘映像

   如果你没有用来加密的物理设备，作为替换你可以使用命令 dd 来建立一个空磁盘映像，然后将该映像作为回送设备来装载。

   ```
   1 dd if=/dev/zero of=~/secret.img bs=1M count=100
   ```

   新建一个大小为100MB的磁盘映像，该映像名字为secret.img。大小由count值决定

   使用losetup命令将该映像和一个回送设备联系起来

   ```
   1 sudo losetup /dev/loop0 ~/secret.img
   ```

   现在已经得到一个虚拟块设备，其位于 /dev/loop0，并且我们可以如同使用其他设备一样使用它

2. 设置块设备

   准备好物理块设备或者虚拟块设备（像前面那样建立了回送映象，并利用device-mapper将其作为加密的逻辑卷加载），我们就可以进行块设备配置了。

   使用cryptsetup建立逻辑卷，并将其与块设备绑定

   ```
   1 sudo cryptsetup -y create myEncryptedFilesystem /dev/DEVICENAME
   ```

   其中，myEncrypteFilesystem 是新建逻辑卷。 其中最后一个参数是将作为加密的块设备。所以，根据以上建立的回送映像虚拟块设备，应当运行

   ```
   1 sudo cryptsetup -y create myEncryptedFilesystem /dev/loop0
   ```

   无论是物理设备还是虚拟块设备，程序会要求你输入逻辑卷的口令，-y 要求你输入两次该口令以确保无误。因为一旦口令错误，谁也帮不了你

为了确认逻辑卷是否已经建立，使用下列命令进行检测

```
sudo dmsetup ls
```

device-mapper会把它的虚拟设备装载到/dev/mapper下面，所以的虚拟块设备应该是/dev/mapper/myEncryptedFilesystem ，尽管用起来它和其它块设备没什么不同，实际上它却是经过透明加密的。

在虚拟设备上创建文件系统：

```
sudo mkfs.ext3 /dev/mapper/myEncryptedFilesystem
```

现在为新的虚拟块设备建立一个装载点，然后将其装载:

```
sudo mkdir /mnt/myEncrptedFilesystem
sudo mount /dev/mapper/myEncryptedFilesystem
/mnt/myEncryptedFilesystem
```

使用下列命令查看装载后的情况

```
df -h /mnt/myEncryptedFilesystem
```

```
| Filesystem                        | Size | Used | Avail | Use% | Mounted on
|
| --------------------------------- | ---- | ---- | ----- | ---- | ----------
---------------- |
| /dev/mapper/myEncryptedFilesystem | 93M  | 1.6M | 87M   | 2%   |
/mnt/myEncryptedFilesystem |
```

现在的文件系统看起来与其他文件系统无异，但实际上写到/mnt/myEncryptedFilesystem /下的所有数据，在数据写入之前都是经过透明的加密处理后才写入磁盘的，因此，从该处读取的数据都是些密文。