|  |  |
| --- | --- |
| 产品名称Product name | 密级Confidentiality level |
|  | 机密 Confidential |
| 产品版本Product version | Total pages 共页 |
|  |

Centos7\_3.18.27介绍

|  |  |  |  |
| --- | --- | --- | --- |
| Prepared by  拟制 | 郭永强 00352 | Date  日期 | 2016-4-21 |
| Reviewed by  评审人 |  | Date  日期 | yyyy-mm-dd |
| Approved by  批准 |  | Date  日期 | yyyy-mm-dd |



Zhejiang Uniview Technologies Co., Ltd.

浙江宇视科技有限公司

All rights reserved

版权所有 侵权必究

（REP01T06 V2.10 IPD-CMM V5.0 / for internal use only）

（REP01T06 V2.10/ IPD-CMM V5.0 / 仅供内部使用）

Revision Record 修订记录

| Date  日期 | Revision Version  修订 版本 | CR ID / Defect ID CR号 | Sec No.  修改 章节 | Change Description  修改描述 | Author  作者 |
| --- | --- | --- | --- | --- | --- |
| 2016/4/21 | 1.00 |  |  | initial 初稿完成 | 郭永强 00352 |
|  |  |  |  |  |  |
|  |  |  |
|  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |
|  |  |  |

Catalog 目 录

[1 Introduction 5](#_Toc450227310)

[1.1 Centos7基本介绍 5](#_Toc450227311)

[1.1.1 grub介绍 5](#_Toc450227312)

[1.1.2 分区 5](#_Toc450227313)

[1.1.3 服务 5](#_Toc450227314)

[1.1.4 常用命令 5](#_Toc450227315)

[1.1.5 计算机改名 6](#_Toc450227316)

[1.1.6 防火墙 6](#_Toc450227317)

[1.1.7 yum包安装 6](#_Toc450227318)

[1.1.8 启动流程 6](#_Toc450227319)

[1.2 工具介绍 8](#_Toc450227320)

[1.3 内核安装 8](#_Toc450227321)

[1.3.1 通过存储本机安装内核 8](#_Toc450227322)

[1.3.1 编译服务器安装后拷贝到存储上 8](#_Toc450227323)

[1.3.2 模块安装 9](#_Toc450227324)

[1.3.3 单独编译内核 10](#_Toc450227325)

[1.4 内存文件系统 10](#_Toc450227326)

[1.4.1 内存文件系统生成方式 10](#_Toc450227327)

[1.5 串口升级小系统 11](#_Toc450227328)

[1.6 unistor平台代码修改 13](#_Toc450227329)

[1.7 GUI升级修改 13](#_Toc450227330)

[1.8 kdump小系统修改 15](#_Toc450227331)

Table List 表目录

**未找到目录项。**

Figure List 图目录

图1 Centos7启动流程 7

图2 模块目录结构 9

图3 内存文件系统目录 11

图4 Makefile编译顺序 13

# Introduction

## Centos7基本介绍

### grub介绍

采用grub2 ，目录:/boot/grub2/grub.cfg, grub.cfg中的变量通过修改grubenv即可。

[root@localhost ~]# cat /boot/grub2/grubenv

# GRUB Environment Block

saved\_entry=0

pnpacpi\_switch=on

crash\_start=48

crash\_size=160

v3\_nvram\_start=360

v3\_nvram\_size=10

cache\_start=7478

cache\_size=1738

kdump\_start=2880

kdump\_size=4

nvram\_start=2884

nvram\_size=64

### 分区

GPT: 一个硬盘最多可分配128个主分区

### 服务

/usr/lib/systemd/

/usr/lib/systemd/system

系统启动的第一个进程（进程号1）是systemd，代替了原来的init

### 常用命令

systemctl status|start|stop|restart|reload UNIT

systemctl enable|disable UNIT

systemctl mask UNIT  完全disable,使unit不被手动启动或开机启动

systemctl unmask UNIT

systemctl list-unit-files [--type=service] (注：static，表示这个服务不能单独启动，而是由其它服务调用带动启动)

systemctl list-units [--type=service]  [--all]

systemctl --failed --type=service 查看失败的服务

systemctl list-dependencies UNIT 依赖的服务

systemctl reboot|poweroff|halt

systemctl get-default 获取默认的启动等级

systemctl  set-defalut muti-user.target 设置默认的启动等级

systemctl --type=service 只查类型是service的Unit

systemctl status xxx.service -l  查看更详细信息

systemctl is-enabled  xxx.service 是否启动该服务

systemd-analyze blame 查看每个服务详细的运行时间

### 计算机改名

现在是/etc/hostname

以前是/etc/sysocnfig/network

### 防火墙

默认firewalld,目前已禁用

### yum包安装

常用的rpm包都存在于Centos7自带的ISO中，CentOS-7-x86\_64-Everything-1503-01.iso，采用本地建立yum源安装rpm包即可。需要注意的是yum –y remove命令会将依赖该rpm包的其它rpm全部卸载掉，因此对于被卸载的rpm包需要一一确认。从而确认哪些rpm包需要重新安装。

yum –y install xxx

yum –y remove xxx

### 启动流程



Centos7启动流程

注：可以到官网上下载 systemd-208-20.el7.src源码包学习整个systemd流程；

## 工具介绍

* *dracut /boot/initramfs-$(uname -r).img $(uname -r)* 重新生成新的内存文件系统命令；
* *build\_kdump\_initrd.sh* 对内存文件系统、串口升级小系统、kudmp小系统解压、压缩脚本；
* *build\_kernel.sh*  编译内核脚本；
* strip\_ko.sh 对/lib/modules/$(uname -r)下的ko进行strip动作；
* *grub2-set-default 0* 从新内核启动；
* *grub2-mkconfig –o /boot/grub2/grub.cfg* 重新生成新的grub文件；

## 内核安装

* 目前Centos7.2自带的内核版本是3.10.0，若要在Centos7.1上安装3.18.27的内核版本，则需要下载3.18.27的源码包进行编译安装。网站上提供的安装包格式如下：

linux-3.18.27.tar.xz；

* 3.10.0->3.18.27的安装不同于3.10.0->修改后的3.10.0，除了内核img文件需要修改之外、内存文件系统、串口升级小系统、kdump小系统、/lib/modules/对应的模块都需要修改替换。下面重点说明3.10.0->3.18.27内核如何安装。

### 通过存储本机安装内核

* 步骤：
* 将linux-3.18.27.tar.xz 拷贝到/usr/src/kernels/目录下；
* 进行解压 tar –xvf linux-3.18.27.tar.xz；
* 编辑make menuconfig或者从SVN上拷贝config文件到/usr/src/kernels/3.18.27/.config
* 采用工具将平台修改的内核文件替换原有的文件并编译(将内核与模块全部编译出来)sh build\_kernel.sh build\_all 3.18.27/ linux\_kernel\_3.18.27/
* 执行make modules\_install 安装模块
* sh strip\_ko.sh 对/lib/modules/3.18.27/目录下的ko执行strip动作；
* 执行 make install 主要工作如下：

安装内核文件 vmlinuz-3.18.27

生成内存文件系统

更行grub.cfg文件

* grub2-set-default 0 从新内核启动
* 按照*《3.18.27内核裁剪》*裁剪不用的模块，主要是针对/lib/modules/3.18.27目录下的模块执行，可以写成一个脚本执行；直接安装后的/lib/modules/3.18.27下的ko模块大小约1.6G，经strip后变成100MB，经裁剪后变成47MB；
* 内核安装完毕；

### 编译服务器安装后拷贝到存储上

* 步骤：
* 将linux-3.18.27.tar.xz 拷贝到/usr/src/kernels/目录下；
* 进行解压 tar –xvf linux-3.18.27.tar.xz；
* 编辑make menuconfig或者从SVN上拷贝config文件到/usr/src/kernels/3.18.27/.config
* 采用工具将平台修改的内核文件替换原有的文件并编译(将内核与模块全部编译出来)sh build\_kernel.sh build\_all 3.18.27/ linux\_kernel\_3.18.27/
* 执行make modules\_install 安装模块
* sh strip\_ko.sh 对/lib/modules/3.18.27/目录下的ko执行strip动作；
* 执行 make install 主要工作如下：

安装内核文件 vmlinuz-3.18.27

生成内存文件系统

更新grub.cfg文件

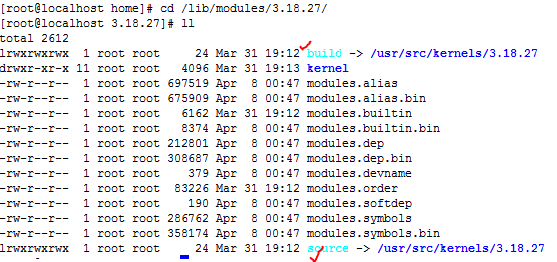
将内核文件、内存文件系统、/lib/modules/3.18.27拷贝到存储服务器上；

* 在存储设备上按照*《3.18.27内核裁剪》*裁剪不用的模块，主要是针对/lib/modules/ 3.18.27目录下的模块执行，可以写成一个脚本执行；
* 删除/boot/grub2/grub.cfg；
* grub2-mkconfig –o /boot/grub2/grub.cfg 重新生成新的grub.cfg文件；
* 参考SVN上的grub.cfg文件修改grub.cfg文件；
* 将配置文件放到/boot目录下，版本号命名以便查看，如config-3.18.27；
* 内核安装完毕；

### 模块安装

#### make modules\_install

执行make modules\_install命令后，会将编译好的ko模块安装在/lib/modules/ $(uname -r) 下，若原来已经安装了一个版本的内核如：3.10.0-229.el7.x86\_64，后面又需要安装一个3.18.27的内核，那么执行make modules\_install命令之后，在/lib/modules/会自动生成一个3.18.27的文件夹。文件夹中的内容如下：



模块目录结构

说明：对于上面build、source软连接部分，可以删除（对于编译环境千万别删/usr/src/kernels/3.18.27），因为平台代码完全依赖于这个内核目录进行编译。需要强调的是：平台的Makefile中存在两种调用内核路径，如下，通常情况下两个路径指向是一样的，建议采用第一种方案：

KERNEL\_DIR := /usr/src/kernels/$(uname -r)

KERNEL\_DIR := /lib/modules/$(uname -r)/build

* 什么时候上面两个目录的内容不一样
* 在/usr/src/kernels/$(uname -r)目录下编译，但是没有安装；
* 在别的目录下编译并执行了make modules\_install；
* 因此，若安装内核及模块，建议将内核源码存放在 /usr/src/kernels目录下进行解压编译安装；

### 单独编译内核

如果内核文件需要修改，比如要导出符号、增加函数等，需要重新编译内核即可。

* 步骤：
* 将linux-3.18.27.tar.xz 拷贝到/home/xxx/目录下(目录无强制要求)；
* 进行解压 tar –xvf linux-3.18.27.tar.xz；
* 编辑make menuconfig或者从SVN上拷贝config文件到/home/xxx/3.18.27/.config
* 采用工具将平台修改的内核文件替换原有的文件并编译(只编译内核)

sh build\_kernel.sh build 3.18.27/ linux\_kernel\_3.18.27/

* 将编译后的内核文件bzImage拷贝到存储设备/boot目录下，并更名为grub需要的名字，如vmlinuz-3.18.27；
* 将/home/xxx/3.18.27/Module.symvers拷贝到/usr/src/kernels/3.18.27/下，替换原来的额符号表；
* 结束；

## 内存文件系统

内存文件系统 /boot/initramfs-3.18.27.img，目的是通过在内存中仿真成一个根目录，加载启动过程中必须的内核驱动模块。因此该img文件中必须包含文件系统及系统盘设备驱动。我们目前用的系统盘通常包括：CF/MSATA/SATA/SSD，因此其对应的驱动必须存在。

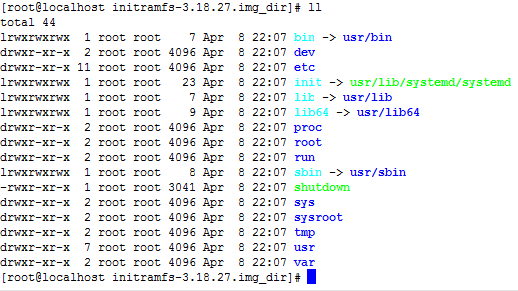
### 内存文件系统生成方式

* 前面介绍了两种内核安装的方式：通过本机make install或者从服务器上拷贝的方式将内存文件系统拷贝到存储/boot/目录下即可。也可以通过命令dracut生成，该命令生成的原理：根据系统现有的硬件将必要的驱动打包起来。实际上内存文件系统需要的驱动非常少。文件系统 及系统盘（CF/MSATA/SSD/SATA）需要的驱动。网卡驱动、I2C驱动都可以在后面真正的系统中加载。

通过命令dracut生成内存文件系统的方法：

*dracut* /boot/initramfs-$(uname -r).img $(uname -r)

* 生成内存文件系统后，若想进一步去优化，则需要利用工具 build\_kdump\_initrd.sh 进行解压、修改、压缩：sh *build\_kdump\_initrd.sh* initramfs-3.18.27.img release



内存文件系统目录

* 裁剪驱动模块
* 删除不必要的ko，最后保留ko如下：

[root@localhost initramfs-3.18.27.img\_dir]# find ./ -name "\*.ko"

./usr/lib/modules/3.18.27/kernel/drivers/scsi/sr\_mod.ko

./usr/lib/modules/3.18.27/kernel/drivers/scsi/scsi\_transport\_sas.ko

./usr/lib/modules/3.18.27/kernel/drivers/scsi/sd\_mod.ko

./usr/lib/modules/3.18.27/kernel/drivers/ata/ata\_piix.ko

./usr/lib/modules/3.18.27/kernel/drivers/cdrom/cdrom.ko

./usr/lib/modules/3.18.27/kernel/fs/ext4/ext4.ko

./usr/lib/modules/3.18.27/kernel/fs/jbd2/jbd2.ko

./usr/lib/modules/3.18.27/kernel/fs/mbcache.ko

* libata libahci ahci等驱动都编译到内核，考虑到后续只编译内核，而不需要每次都重新制作三个文件系统。

CONFIG\_ATA=y

CONFIG\_ATA\_ACPI=y

CONFIG\_SATA\_AHCI\_PLATFORM=y

* 内存文件系统中的ko要经过strip,如果/lib/modules/下的ko已经strip过，然后才通过命令*dracut*生成内存文件系统，则无需理会；

## 串口升级小系统

* 目的

构造一个虚拟文件系统，将ISO拷贝到内存中解压、格式化分区并进行安装、升级DEU/SAS/BIOS/CPLD。该小系统需要网络、需要文件系统、需要常用的工具、及ISO升级过程中的rpm包。如果直接拿现有的内存文件系统作为串口升级小系统是行不通的。因为内存文件系统中很多ISO升级依赖的工具、库都缺失。目前的方案是将busybox小系统编译到内存文件系统中，最终生成串口升级小系统。方法如下：

* 步骤
* 将内存文件系统改名 mv initramfs-3.18.27.img initrd\_netrecovery-3.18.27.img
* 对内存文件系统进行解压

sh build\_kdump\_initrd.sh initrd\_netrecovery-3.18.27.img release

* 下载busybox源码包并进行解压，如:busybox-1.23.2.tar.bz2
* 在解压后的busybox目录下修改配置文件

make menuconfig 配置static编译和安装路径

[\*] Build BusyBox as a static binary (no shared libs) #配置busybox运行为静态库模式

Installation Options ("make install" behavior) ---> #修改安装路径为实际需要用到的initrd解压路径，如：



* make
* make install 自动在initrd解压目录安装busybox并建立链接
* 在initrd\_netrecovery-3.18.27.img\_dir目录下增加文件

./init

./usr/bin/getfile.sh

./usr/bin/get\_iso\_file\_size

./usr/bin/isoupdate

./usr/bin/rename\_nic

./usr/bin/rename\_pci\_nic

./usr/bin/rename\_usb\_nic

./usr/lib64/ libreadline.so.6

./usr/lib64/libncurses.so.5

./usr/sbin/udevd

./usr/lib/modules/3.18.27/kernel/drivers/net/vxlan.ko

./usr/lib/modules/3.18.27/kernel/drivers/net/mii.ko

./usr/lib/modules/3.18.27/kernel/drivers/net/ethernet/intel/e1000e/e1000e.ko

./usr/lib/modules/3.18.27/kernel/drivers/net/ethernet/intel/igb/igb.ko

./usr/lib/modules/3.18.27/kernel/drivers/net/ethernet/intel/i40e/i40e.ko

./usr/lib/modules/3.18.27/kernel/drivers/net/ethernet/intel/ixgbe/ixgbe.ko

./usr/lib/modules/3.18.27/kernel/drivers/net/usb/usbnet.ko

./usr/lib/modules/3.18.27/kernel/drivers/net/usb/smsc95xx.ko

./usr/lib/modules/3.18.27/kernel/drivers/net/mdio.ko

./usr/lib/modules/3.18.27/kernel/drivers/block/loop.ko

* 增加rpm包，这些rpm包主要用于ftp传输升级fw使用

./usr/rpms/dmidecode-2.12-5.el7.x86\_64.rpm

./usr/rpms/ expect-5.45-12.el7.x86\_64.rpm

./usr/rpms/ftp-0.17-66.el7.x86\_64.rpm

./usr/rpms/glibc-2.17-78.el7.i686.rpm

./usr/rpms/glibc-2.17-78.el7.x86\_64.rpm

./usr/rpms/ libstdc++-4.8.3-9.el7.x86\_64.rpm

./usr/rpms/localtime

./usr/rpms/services

* 由于整个串口升级小系统并不需要systemd这套流程，因此将systemd这套流程给删除掉。

rm –rf ./usr/lib/systemd

rm –rf ./etc/systemd

rm –rf ./usr/bin/systemd-cgls

rm –rf ./usr/bin/systemd-tmpfiles

rm –rf ./usr/bin/systemctl

rm –rf ./etc/conf.d/systemd.conf

* 重新打包img文件，将生成的img文件拷贝到存储上/boot目录下即可

sh build\_kdump\_initrd.sh initrd\_netrecovery-3.18.27.img build

* 串口升级小系统制作完成

## unistor平台代码修改

* 删除如下两个内核符号表

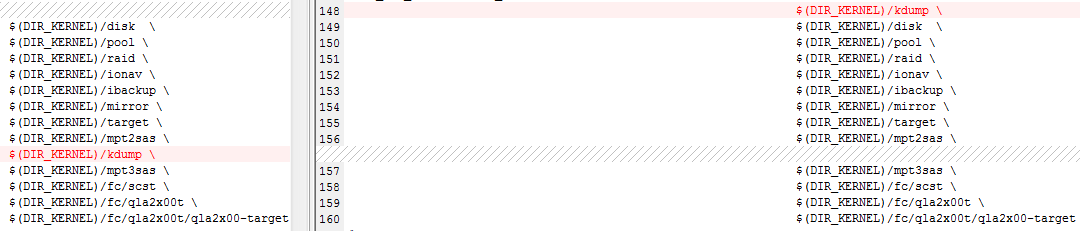
/storware/kernel/kdump/Kernel\_Module.symvers

/storware/kernel/sata/Kernel\_Module.symvers

* 这样所有平台代码编译时都依赖于/usr/src/kernels/$(uname -r)/ Module.symvers
* 将整个cf目录删除，修改编译顺序，这样无需单独提供kdump下的Module.symvers

/mnt/bugfix/g00352/RAIDNT\_D051SP01/trunk/storware/kernel/cf/ Module.symvers

* 修改Makefile文件，修改编译顺序



Makefile编译顺序

* 由于2.6.32、3.10.0、3.18.27三个内核版本差异都很大，直接导致unistor平台代码很多依赖内核的接口都要发生变化，修改内容请参见*《CentOS7内核变化修改列表》*；
* 2.6.32与后面两个版本变化非常大，版本整个操作过程中分为两个阶段，最终于要基于3.18.27集成CEPH：

2.6.32->3.10.0（基于D052\_Centos7版本提供）；

3.10.0->3.18.27（基于D055\_ Centos7版本提供）；

## GUI升级修改

###### 升级小系统制作流程

1、首先从SVN上boot目录下取出img文件 initrd\_netrecovery-3.18.27.img，一般目录是

D055\_CentOS7\ISO\Modules\base\_CentOS7\file\boot

2、使用build\_kdump\_initrd.sh脚本解压，解压命令如下：



sh build\_kdump\_initrd.sh initrd\_netrecovery-3.18.27.img release

就会在当前目录下生成 initrd\_netrecovery-3.18.27.img目录，打开目录显示如下：



3、修改对应目录上的文件（或者添加或者删除），修改完成后，压缩好小系统

sh build\_kdump\_initrd.sh initramfs-3.18.27kdump.img build

这样就做好了修改后的netrecovery img小系统，然后替换掉设备上boot目录下的netrecovery小系统

###### GUI升级修改为调用升级小系统

1、Deufw\_update.c 文件中修改：

#define SWITCH\_TO\_KDUMP\_CMD "echo 4 > /sys/kdump\_update/kdump\_debug"

修改为：

#define SWITCH\_TO\_KDUMP\_CMD "/bin/switch\_to\_netrecovery.sh"

然后编译替换deufw\_update libdeufw\_update.so这两个文件

2、\trunk\storware\usr\gui\jgui\com\storware\ipsan\console\rpc\SWGlobalConstant.java

public static final int ISO\_UPDATE\_TIME\_OUT = 60; //升级时超时时间60s

修改为：

public static final int ISO\_UPDATE\_TIME\_OUT = 180; //升级时超时时间180s

然后编译GUI，并做以下动作：

a、替换sms\_install/usr/lib/\*.jar

b、执行gui\_tomcat.sh

c、执行jarsigner.sh

d、IE重新下载GUI控制台

3、switch\_to\_netrecovery.sh脚本和pre\_iso\_update 脚本

A、Switch\_to\_netrecovery.sh脚本主要是修改grubenv文件，通过GUI发送命令执行脚本，然后让其从升级小系统启动。脚本放入正式系统bin目录下 \D055\_CentOS7\ISO\Modules\driver\_CentOS7\_common\file\usr\bin

B、pre\_iso\_update脚本

这个脚本的作用就是在进入升级小系统后，修改升级小系统root用户的密码，让其为无密码模式，自动配置小系统的ip，配置完全参照正式系统的网络配置，相当于复制出来一套。然后就是挂载GUI上传iso所在分区，取得iso。挂载光盘镜像之后，运行telnetd作为telnet通信服务器。

将pre\_iso\_update脚本放入initrd\_netrecovery-3.18.27.img里面的/usr/sbin目录。

4、修改initrd\_netrecovery-3.18.27.img里面的init脚本，init脚本是升级小系统最先执行的文件，将pre\_iso\_update脚本放入里面执行。

5、GUI等到一切都就绪后，将作为telnet客户端，然后尝试连接，测试连接通过后，发送命令/mnt/RecoverCD/resumesystem.sh脚本开始升级

附：

将修改完后重新压缩的 initrd\_netrecovery-3.18.27.img文件放入如下目录，替换掉已有img：

\D055\_CentOS7\ISO\Modules\base\_CentOS7\file\boot

## kdump小系统修改

###### 目的

###### Linux系统在遇到崩溃的情况下，获取内存崩溃时的内存镜像，有助于分析系统在崩溃前发生什么，分析原因并修复错误，改进系统的稳定性。我们的kdump主要是检测到以下如：切换升级，异常掉电，狗溢出，系统堆栈等，出现这类情况后，系统会将出现异常的日志打印出来，并且刷缓存写到CF卡上，主要写三个部分：nvram,V3 nvram和写缓存。刷完缓存后，系统重启。

###### kdump制作过程

1、首先从SVN上boot目录下取出img文件 initramfs-3.18.27kdump.img，一般目录是

D055\_CentOS7\ISO\Modules\base\_CentOS7\file\boot

2、使用build\_kdump\_initrd.sh脚本解压，解压命令如下：

sh build\_kdump\_initrd.sh initramfs-3.18.27kdump.img release

就会在当前目录下生成 initramfs-3.18.27kdump.img\_dir目录，打开目录显示如下：



3、修改对应目录上的文件（或者添加或者删除），修改完成后，压缩好小系统

sh build\_kdump\_initrd.sh initramfs-3.18.27kdump.img build

这样就做好了修改后的kdump img小系统，然后替换掉设备上boot目录下的kdump小系统

###### kdump小系统修改文件

增加文件

./usr/bin/ioport

在平台代码.c文件修改完后，重新编译生成新的ko文件，kdump的三个主要的ko文件

./usr/lib/modules/3.18.27/getkdump.ko

./usr/lib/modules/3.18.27/getkdumpinfo.ko

./usr/lib/modules/3.18.27/kexecparams.ko

加载fs module

./usr/lib/modules/3.18.27/kernel/fs/jbd2/jbd2.ko

./usr/lib/modules/3.18.27/kernel/fs/mbcache.ko

./usr/lib/modules/3.18.27/kernel/fs/ext4/ext4.ko

./usr/lib/modules/3.18.27/kernel/fs/isofs/isofs.ko

加载其它modules

./lib/modules/$kernel\_version/kernel/addition/sg.ko

./insmod /lib/modules/$kernel\_version/kernel/addition/shpchp.ko

./lib/modules/$kernel\_version/kernel/addition/i2c-i801.ko

./lib/modules/$kernel\_version/kernel/addition/iTCO\_wdt.ko

加载watchdog module

insmod /lib/modules/$kernel\_version/kernel/addition/iTCO\_vendor\_support.ko

新增目录usr/lib64/rsyslog

./usr/lib64/rsyslog/imjournal.so

./usr/lib64/rsyslog/imklog.so

./usr/lib64/rsyslog/imuxsock.so

./usr/lib64/rsyslog/lmnet.so

因为升级方式的改变，GUI不在切换kdump小系统升级，所以去掉所有与升级相关的一些文件和ko

去掉所有网络相关的ko

* rm –rf smsc95xx.ko
* rm –rf ptp.ko
* rm –rf pps\_core.ko
* rm –rf e1000e.ko
* rm –rf igb.ko
* rm –rf i2c-core.ko
* rm –rf i2c-algo-bit.ko
* rm –rf dca.ko
* rm –rf mdio.ko
* rm –rf ixgbe.ko
* rm –rf i40e.ko
* rm –rf bonding.ko
* 去掉升级所需要的脚本和文件
* rm –rf isoupdate
* rm –rf pre\_iso\_update
* 重新打包img文件，将生成的img文件拷贝到存储上/boot目录下即可

sh build\_kdump\_initrd.sh initramfs-3.18.27kdump.img build

* Kdump小系统制作完成