GNU/Linux Controlling And Troubleshooting the Red Hat Enterprise Linux Boot Process

GNU/Linux
Linux 启动与排故

现代计算机系统的硬件和软件组合是一个非常复杂的组合 . 当系统从断电状态到正常运行系统直接到登陆界面 (GUI/CLI) 过程中，系统大量的在让硬件和软件共同协同工作 .

当 Linux 启动时， 如果出现故障，需要通过引导过程来判断故障是如何出现的，从而通过确定引导过程中的断点，而解决故障 .

GNU/Linux
Linux 启动与排故

本课程中，将描述基于 x86_64 硬件系统上如何引导软件及硬件，从而将 RHEL7/CentOS7 成功的引导启动 . 虚拟机安装的系统与真实硬件环境大致相同 .

GNU/Linux
Linux 启动与排故

1. P.O.S.T(加电自检程序): 无论现代的 UEFI 或老式的 BIOS, 通过加电自检来检测当前系统的硬件信息 .

GNU/Linux
Linux 启动与排故

UEFI: 统一的可扩展固件接口” (Unified Extensible Firmware Interface)，是一种详细描述类型接口的标准。

这种接口用于操作系统自动从预启动的操作环境，加载到一种操作系统上。可扩展固件接口（Extensible Firmware Interface，EFI）是 Intel 为 PC 固件的体系结构、接口和服务提出的建议标准。其主要目的是为了提供一组在 OS 加载之前（启动前）在所有平台上一致的、正确指定的启动服务，被看做是有近 20 多年历史的 BIOS 的继任者。

GNU/Linux
Linux 启动与排故

BIOS:Basic Input/Output System ，翻成中文是“基本输入 / 输出系统”

是一种所谓的“固件”，负责在开机时做硬件启动和检测等工作，并且担任操作系统控制硬件时的中介角色。

因为硬件发展迅速，传统式（ Legacy ） BIOS 成为进步的包袱，现在已发展出最新的 UEFI （ Unified Extensible Firmware Interface ）可扩展固件接口，相比传统 BIOS 的来说，未来将是一个“没有特定 BIOS” 的电脑时

GNU/Linux
Linux 启动与排故

UEFI 与 BIOS 区别在于：

1. 编码 99% 都是由 C 语言完成；
2. 一改之前的中断、硬件端口操作的方法，而采用了 Driver/protocol 的新方式；
3. 将不支持 X86 实模式，而直接采用 Flat mode（也就是不能用 DOS 了，现在有些 EFI 或 UEFI 能用是因为做了兼容，但实际上这部分不属于 UEFI 的定义了）；
4. 输出也不再是单纯的二进制 code，改为 Removable Binary Drivers；

GNU/Linux
Linux 启动与排故

UEFI 与 BIOS 的区别：

5. OS 启动不再是调用 Int19 ，而是直接利用 protocol/device Path ；
6. 对于第三方的开发，前者基本上做不到，除非参与 BIOS 的设计，但是还要受到 ROM 的大小限制，而后者就便利多了。
7. 弥补 BIOS 对新硬件的支持不足的毛病。

GNU/Linux
Linux 启动与排故

2. 在 BIOS 中对可引导的设备进行查询，启动一个 MBR(UEFI 启动固件)

3. 激活 MBR 的引导程序，加载 RHEL 启动引导控制程序 :OSLoader(GRUB2)

4. OSLoader 加载其相关配置，并显示相关的配置菜单来引导用户选择相关操作，从而引导 RHEL7/CentOS7 的启动 (/etc/grub.d,/etc/default/grub,/boot/grub2/grub.cfg)

GNU/Linux
Linux 启动与排故

5. 在用户选择后 (或 timeout 后),GRUB2 将加载内核及 initramfs 至内存中 .initramfs 属于一个 img 的虚拟磁盘 . 其是一个 GZIP 的 cpio 归档文件 .

initramfs 包含了动态的内核模块，初始化脚本及非常多的硬件驱动等，在 RHEL7 中 initramfs 自身即包含了一个完整的可用系统 .

配置文件 :/etc/dracut.conf

GNU/Linux
Linux 启动与排故

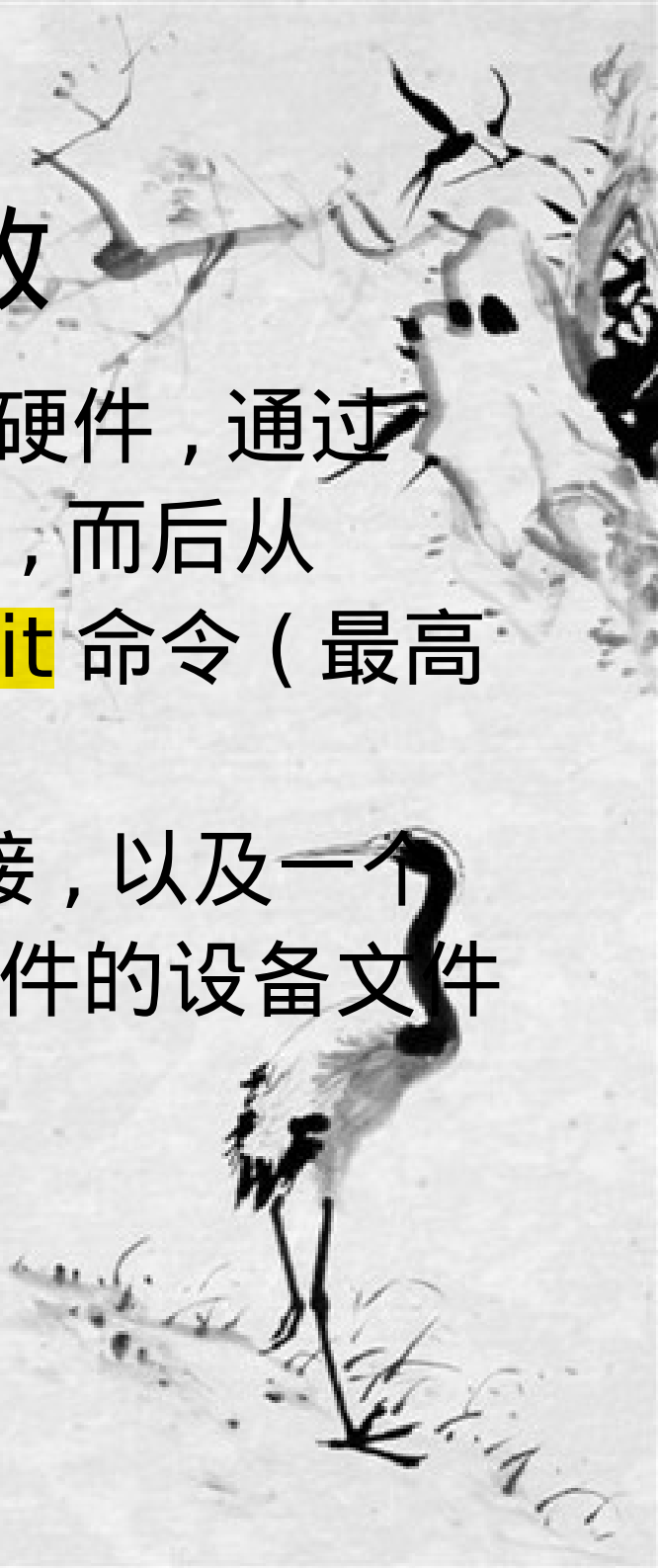
6.GRUB2 将控制系统切换到 kernel, 通过对 GRUB2 的控制可以添加各种 kernel 的选项 . 并将这些选项同时传递至内存中 , 影响 kernel 及 initramfs 的运行 . (/etc/grub.d,/etc/default/grub,/boot/grub2/grub.cfg)

GNU/Linux
Linux 启动与排故

7. kernel 在启动后将初始化所有的硬件，通过 initramfs 找到硬件相关的驱动程序，而后从 initramfs 中执行 PID 1 的 /sbin/init 命令（最高进程）. 在 RHEL 中 init 属于 /lib/systemd/systemd 的软连接，以及一个 udev 进程来自动建立已经存在的硬件的设备文件

配置方式

init = 命令行参数



GNU/Linux
Linux 启动与排故

8. systemd(init) 进程从 initramfs 中执行 initrd.target 系统初始化服务中所有可用单元，来引导 RHEL7/CentOS7 的启动 .

这其中包括了切换到实际的“ /” 硬盘分区及挂载

配置文件 :/etc/fstab

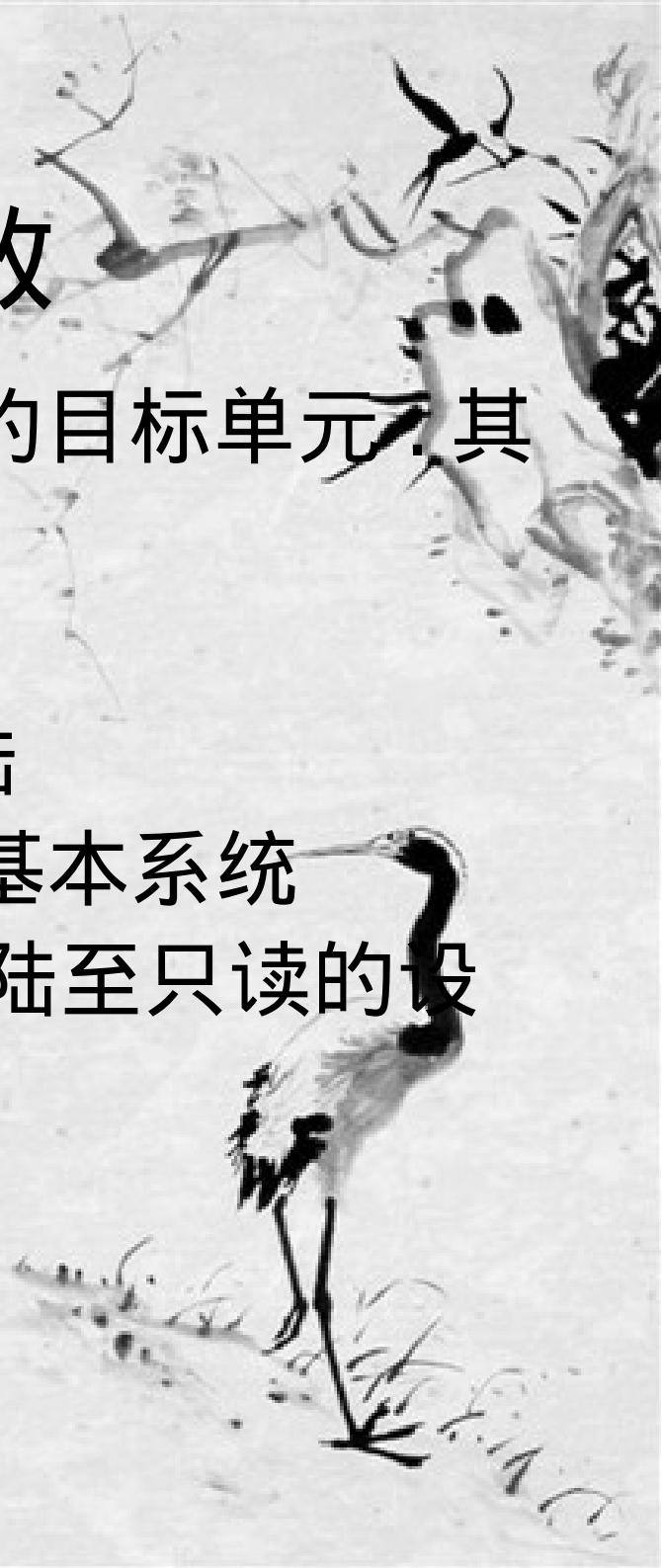
GNU/Linux
Linux 启动与排故

9. 由 initramfs 建立的内存的根区
分 /sysroot(物理 '/' 分区) 在成功挂载之后，将切换到此根分区上， 并将 systemd 重新执行安装至真实的根分区中

10. systemd 开始查找所有的服务，开始执行（停止）相关的服务，以符合该服务的配置并解决相关的依赖问题。

配置文件 :/etc/systemd/system/default.target

GNU/Linux
Linux 启动与排故



systemd 的目标是一组 systemd 的目标单元，其中最重要的如下：

graphical.target: 基于图形的登陆

multi-user.target: 基本文本的登陆

rescue.target: 使用单用户登陆至基本系统

emergency.target: 使用单用户登陆至只读的设备系统上，而非 initramfs 上

GNU/Linux
Linux 启动与排故

1. 查看指定的 target 所需要的关联服务或 target

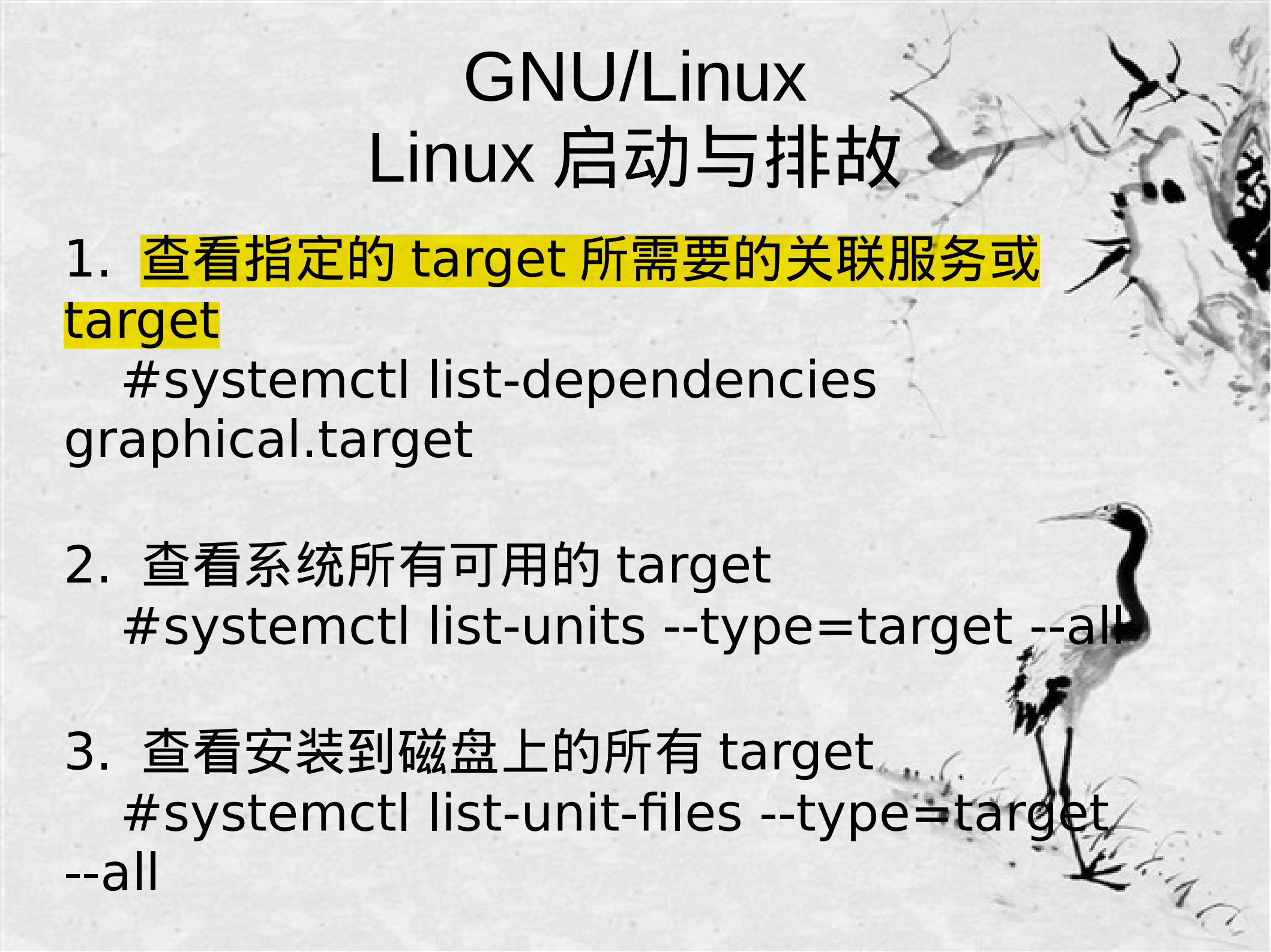
#systemctl list-dependencies graphical.target

2. 查看系统所有可用的 target

#systemctl list-units --type=target --all

3. 查看安装到磁盘上的所有 target

#systemctl list-unit-files --type=target --all



GNU/Linux
Linux 启动与排故

4. 临时切换某个运行级别

   #systemctl isolate multi-user.target

5. 查看默认运行级别

   #systemctl get-default

6. 设置默认运行级别

   #systemctl set-default graphical.target

   #systemctl get-default

GNU/Linux
Linux 启动与排故

7. 在 GRUB2 中切换启动级别（为 kenrel 添加引导参数）

1)reboot Linux

2) 看到 GRUB2 的启动菜单

3) 停止 timeout 计数

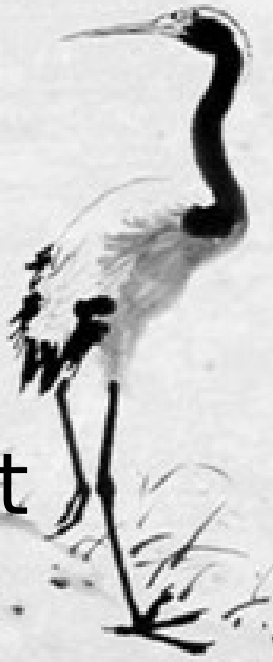
4) 选择一个所需要启动 OS

5) 按 e 编辑此 OS 启动菜单

6) 找到 linux16 字段

7) 添加 systemd.unit=rescue.target

8) 用 ^x 启动此次更改



GNU/Linux
Linux 启动与排故

8. 修改遗忘的 root 的密码

1) 看到 GRUB2 的启动菜单

3) 停止 timeout 计数

4) 选择一个所需要启动 OS

5) 按 e 编辑此 OS 启动菜单

6) 找到 linux16 字段中 ro 字段，将 ro 字段改为 rw init=/sysroot/bin/sh

或

在最尾部直接加

rd.break //*ramdisk=rd

GNU/Linux
Linux 启动与排故

8. 修改遗忘的 root 的密码

7) 用 ^x 启动此次更改

8）切换到 /sysroot(即物理 '/' 分区)

#mount -o remount,rw /sysroot

#chroot /sysroot /bin/bash

9）修改 root 密码

GNU/Linux
Linux 启动与排故

8. 修改遗忘的 root 的密码

10) 如果开启了 SELinux 需要在 '/' 分区上创建 .autorelabel 文件

#touch /.autorelabel

用来使 selinux 的所有的关联标签发生改变，以接受新的 ROOT 密码

11) 执行 'exit' 退出 chroot

12) 执行 'reboot' 重新启动 RHEL/CentOS7

GNU/Linux
Linux 启动与排故

如果有启动失败的信息，可以使用 journald 的 journalctl 来查看信息

1) 建立 journal 的目录

#mkdir -p -m 2775 /var/log/journal

2) 设置权限

#chown .systemd-jounal /var/log/journal

3) 重新加载 jouranl(USR1 信号)

#killall -USR1 systemd-journald

GNU/Linux
Linux 启动与排故

4) 查看上次启动的 ERR 信息 (-b:boot,-1: 上一次 ,-p: 指定日志级别)

#journalctl -b -1 -p ERR

5) 查看本次启动的 ERR 信息

#journalctl -b -0 -p ERR

GNU/Linux
Linux 启动与排故

当系统出现问题时，可以通过 debug 方式来根据错误信息 . 当使用 debug 方式时系统会建立一个 TTY9, 这个 shell 将使用 root 身份来跟踪 / 查看系统启动情况

#systectml enable debug-shell.service

或

#systemctl start debug-shell.service

#reboot

GNU/Linux
Linux 启动与排故

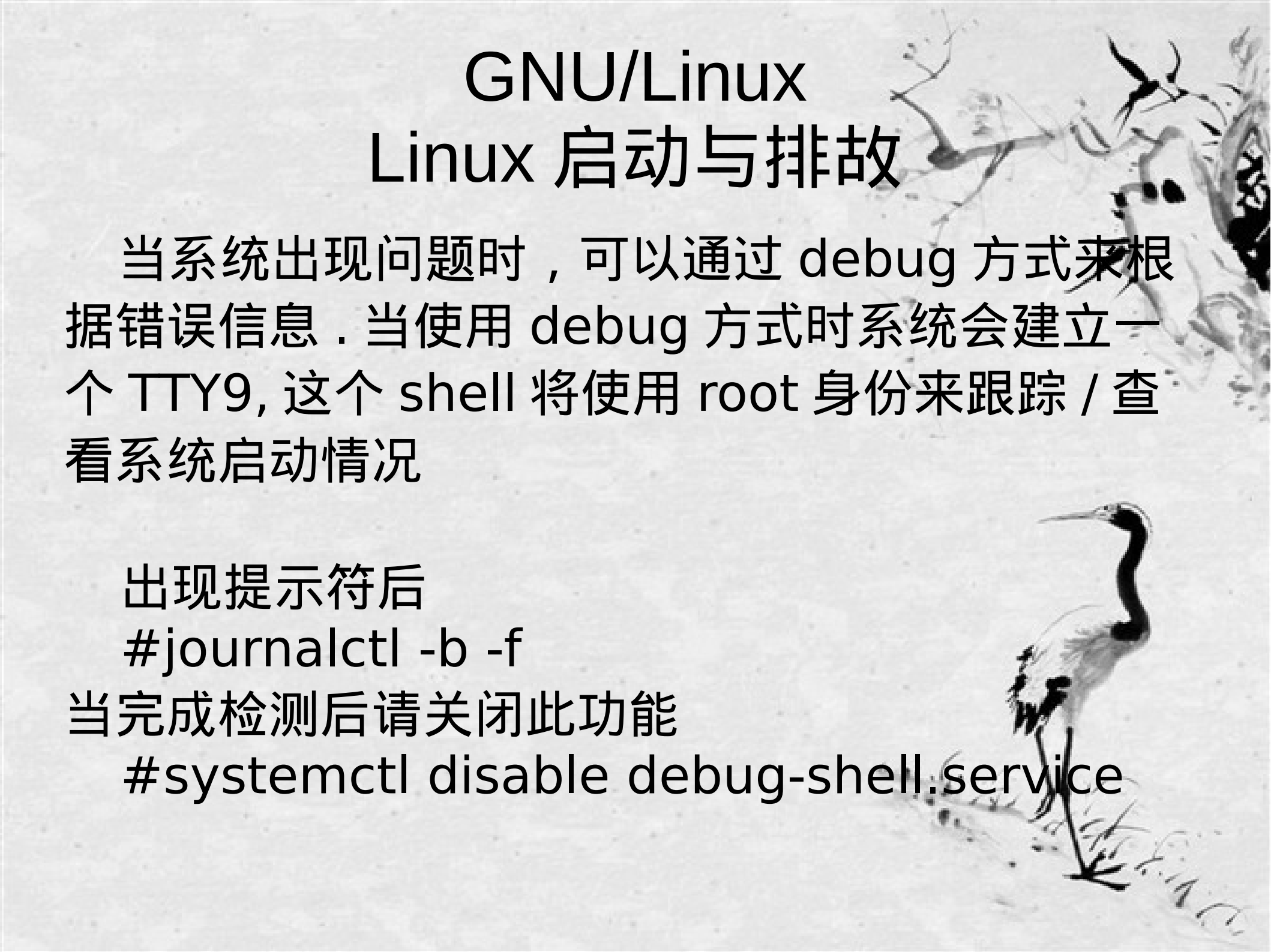
当系统出现问题时，可以通过 debug 方式来根据错误信息 . 当使用 debug 方式时系统会建立一个 TTY9, 这个 shell 将使用 root 身份来跟踪 / 查看系统启动情况

出现提示符后

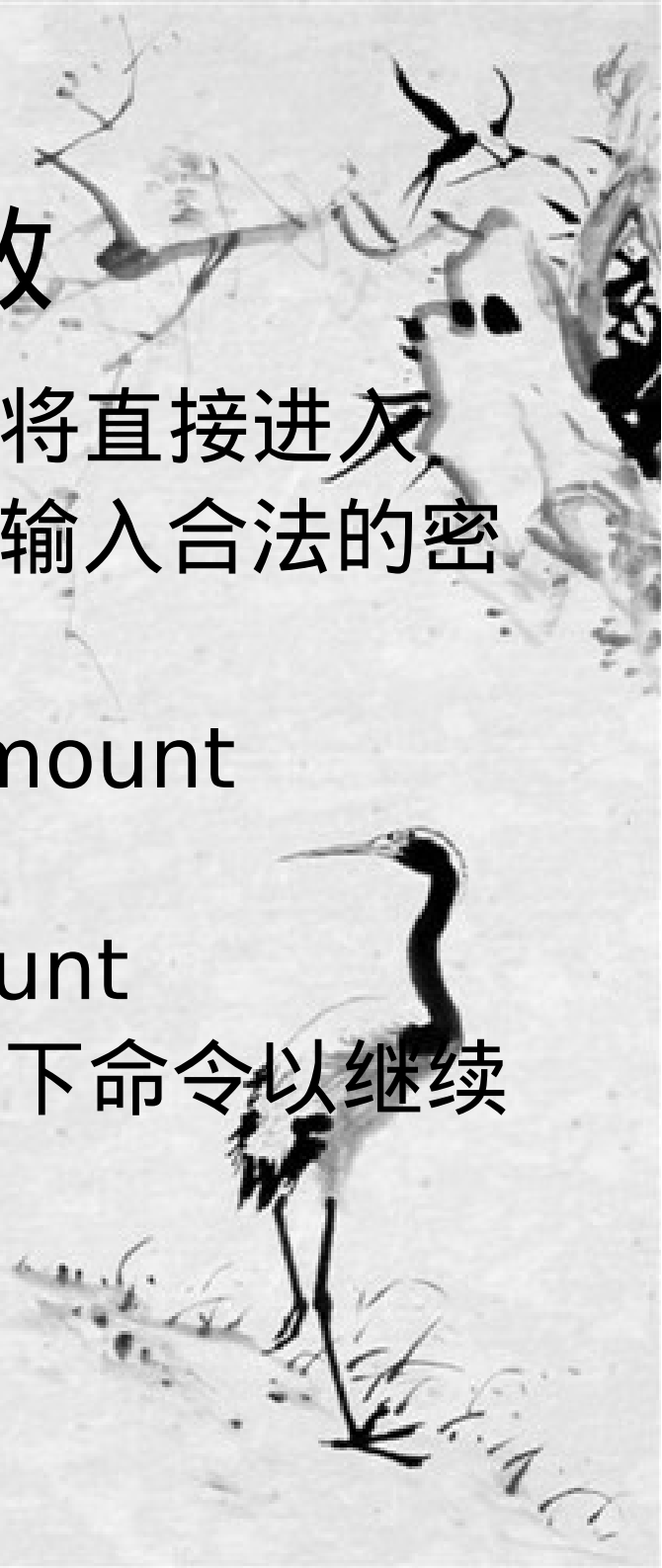
#journalctl -b -f

当完成检测后请关闭此功能

#systemctl disable debug-shell.service



GNU/Linux
Linux 启动与排故



当系统挂载出现问题，在启动时，将直接进入 emergency 模式，此时管理员可以输入合法的密码，来查看 / 修复问题的挂载

#systemctl status -l 挂载点 .mount

如

#systemctl status -l boot.mount

修改完成之后，确认正确，执行如下命令以继续启动系统

#systemctl daemon-reload

GNU/Linux
Linux 启动与排故

常见错误与修复方法

问题	处理方式
损坏的文件系统	systemd 会尝试使用 fsck 进行修复 . 如果无法修复 , 则提示用户从紧急模式运行 fsck 手工进行修复
UUID 不在 /etc/fstab	UUID 或设备文件名 / 卷标写错在 /etc/fstab 中 , 需修改
挂载点等不存在或错误	挂载点 / 文件系统 等错误 , 需修改 /etc/fstab
选项错误	defaults 错误等 , 修改 /etc/fstab

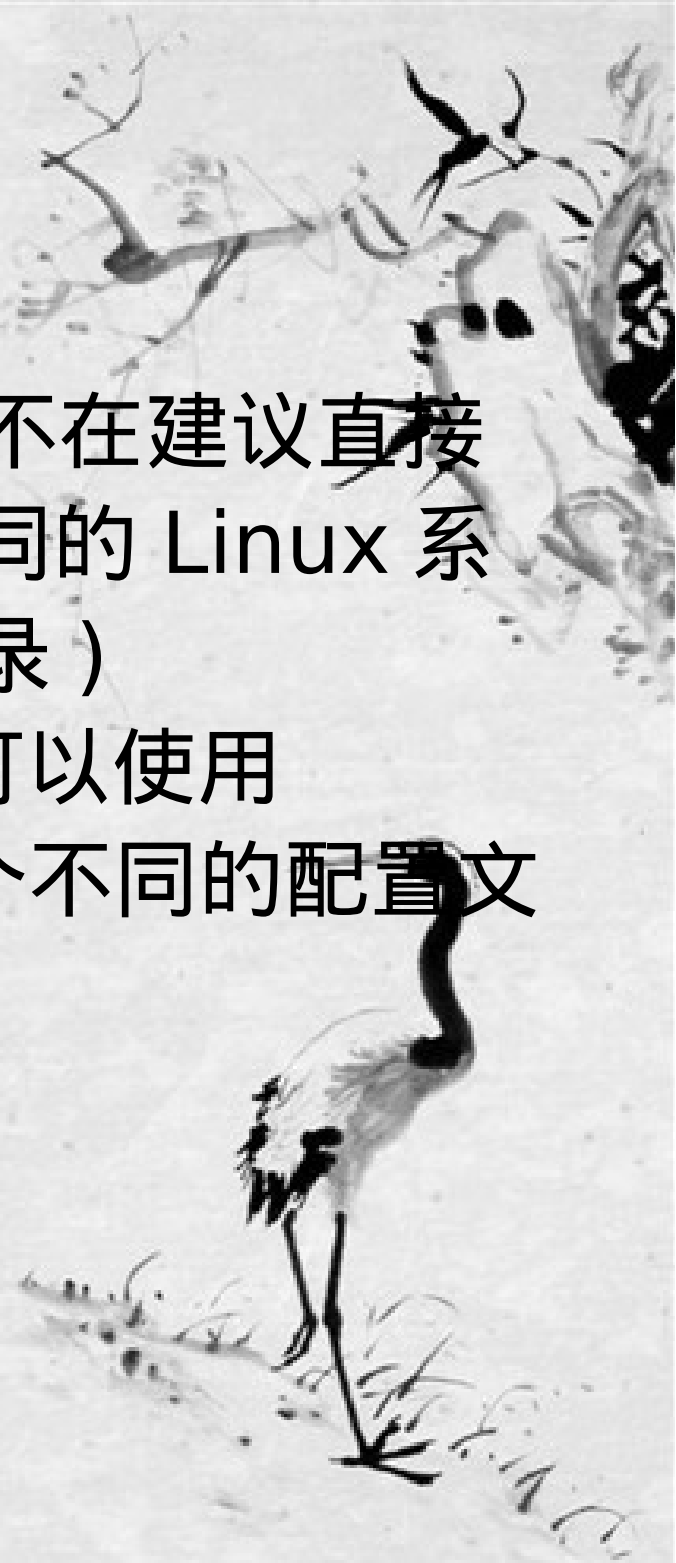
GNU/Linux
GRUB2

在 RHEL/CentOS7 系统下负责 OSLoader 的程序是 GRUB2(GRand Unified Bootloader: 盛大统一的 BootLoader)

GRUB2 可以同时支持 BIOS/UEFI 系统，并且可以在任何的现代硬件上完美的工作。

其扩展性及高度自定义性得到了用户青睐，成为了大部分 Linux 的首选 / 默认的 BootLoader

GNU/Linux
GRUB2



与 grub 的第 1 代不同 ,GRUB2 不在建议直接配置 /boot/grub2/grub.cfg(在不同的 Linux 系统中 grub2 目录有可能是 grub 目录)

如果需要对 GRUB2 进行配置，可以使用 grub2-mkconifg 命令，来生成一个不同的配置文件，并将该配置安装相关的 kernel

GNU/Linux
GRUB2

Grub2-mkconfig 生成 grub.cfg 文件实际通过了如下配置完成此结果

1. /etc/default/grub
2. /etc/grub.d/*

可通过
#grub2-mkconfig | less
查看过程

GNU/Linux
GRUB2

GRUB2 的 grub.cfg 的生成

1)BIOS 类型

#grub2-mkconfig -o /boot/grub2/grub.cfg

或

#grub-mkconfig > /boot/grub2/grub.cfg

2)UEFI 类型

#grub2-mkconfig -o /boot/efi/EFI/redhat/grub.cfg

GNU/Linux
GRUB2

打开新生成的 grub.cfg
//* (boot/grub2 或 /etc/efi/EFI/redhat)
#cd grub2 路径

#vim grub.cfg

GNU/Linux
GRUB2

添加 Windows 启动项

1. 操作系统安装步骤

1) 安装 Windows

2）安装 RHEL/CentOS

2. vim /etc/grub.d/40_custom

GNU/Linux
GRUB2

2. vim /etc/grub.d/40_custom

```
#!/bin/sh
    menuentry 'Windows Server 2008' {
        insmod part_msdos  ← 支持 msdos 类型
        insmod ntfs  ← 支持 ntfs 文件系统
        insmod ntldr   ← 支持 WIN 的 osloader 程序
        set root=(hd0,1) ← 指定 windows 分区
        chainloader +1 ←grub 将引导交付给 ntldr 执行 (windows 所在分区的第一扇区 )
        boot ← 启动
```

GNU/Linux
GRUB2

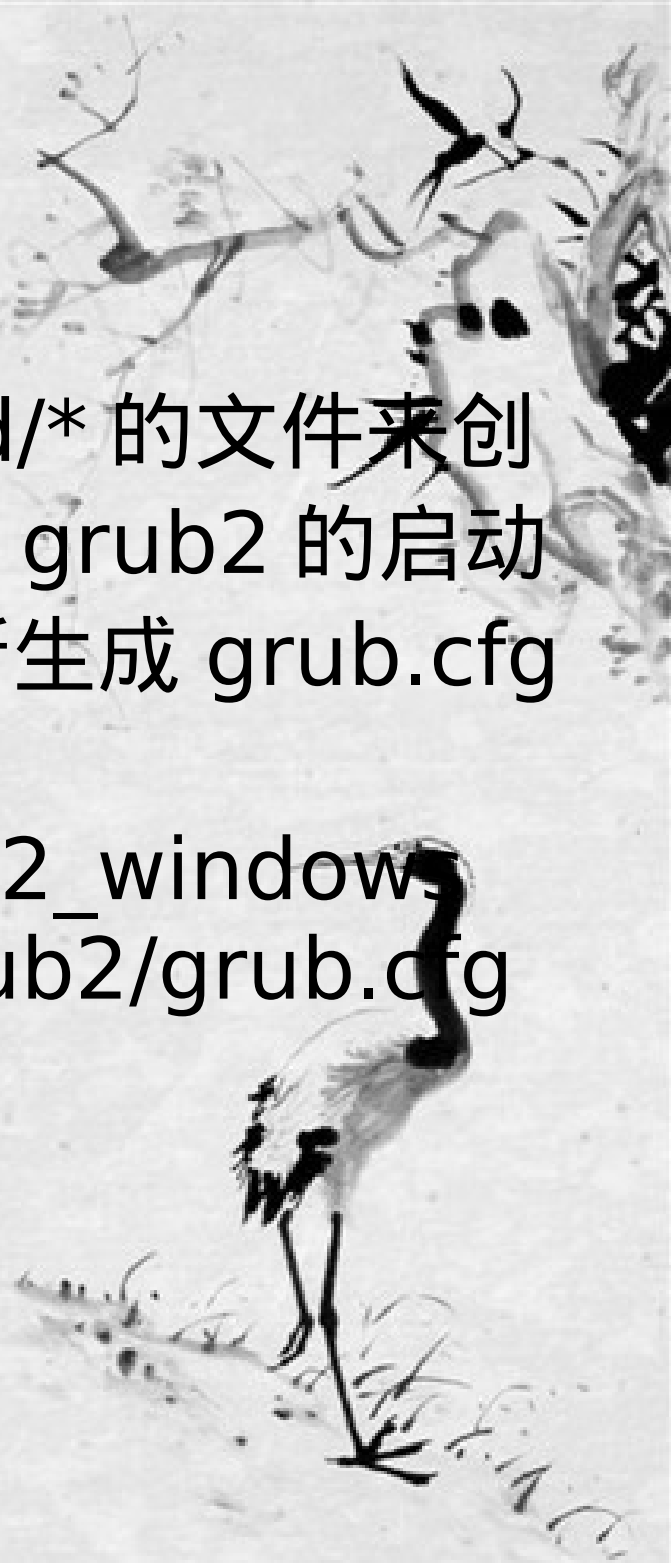
3. 生成新的 grub.cfg

```
#grub2-mkconfig  > /boot/grub2/grub.cfg
```

4. 安装 grub2

```
#grub2-install /dev/sda
```

GNU/Linux
GRUB2



已知 GRUB2 是通过 /etc/grub.d/* 的文件来创建 grub.cfg 配置的。如果打算修改 grub2 的启动顺序，只要将文件的序号改变并重新生成 grub.cfg 即可，如：

```
#cp /etc/grub.d/40_custom 02_windows
#grub2-mkconfig > /boot/grub2/grub.cfg
#grub2-install /dev/sda
```

GNU/Linux
GRUB2

修复 GRUB2

1. LiveCD 修复

   1) 进入 LiveCD 系统

   2) 获取 root 权限

   3) 将错误的 Linux 系统挂载上

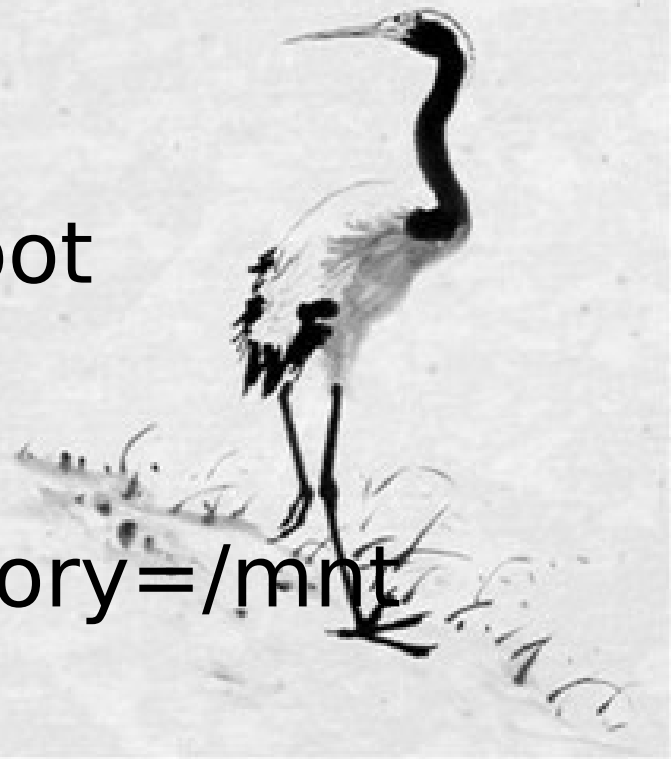
      #mount /dev/sdb1 /mnt/

      #mount /dev/sdb2 /mnt/boot

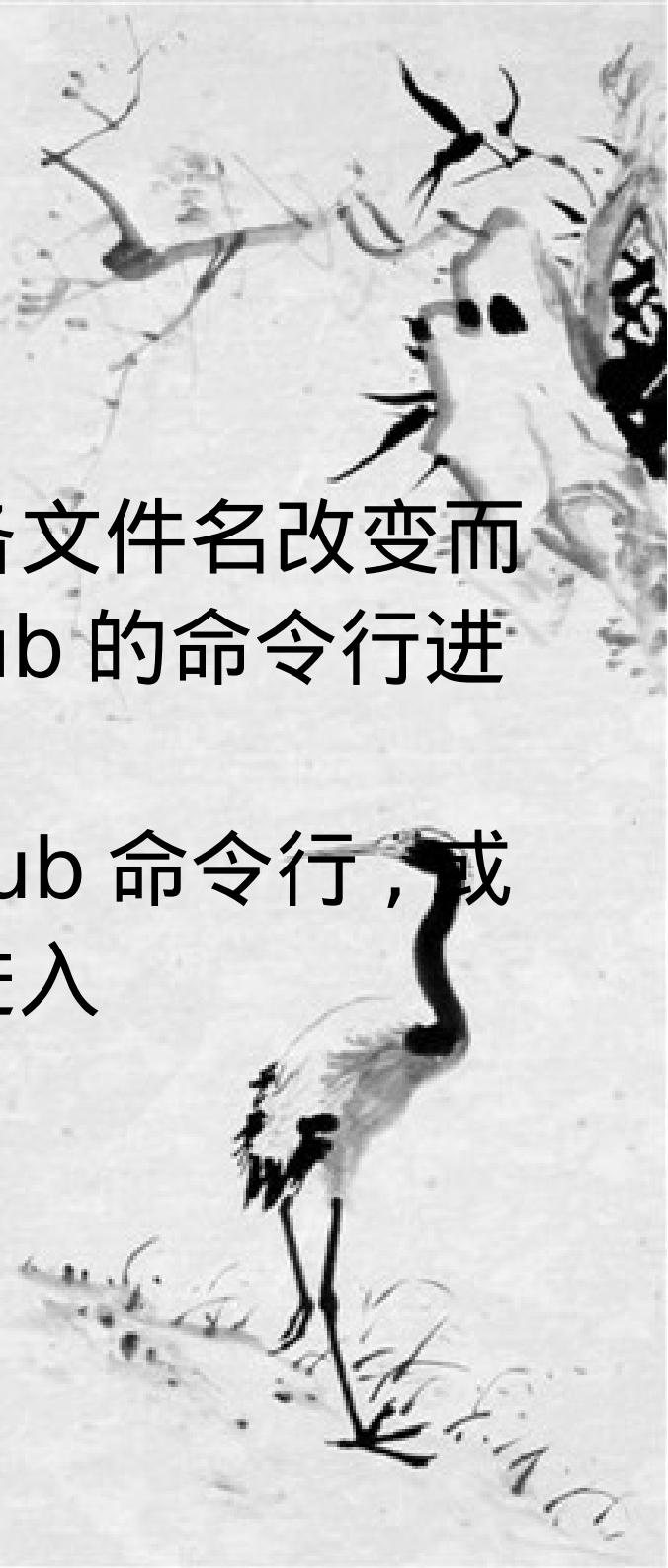
      #mount 其他分区

   4) 修复 grub

      #grub2-install --root-directory=/mnt /dev/sdb



GNU/Linux
GRUB2



修复 GRUB2

2. 因分区编号改变导致 UUID/ 设备文件名改变而产生 GRUB2 无法启动，可通过 grub 的命令行进行修复

1）在 grub 菜单中按 'c' 进入 grub 命令行，或 grub2 无法找到 grub.cfg 将直接进入

2) 找回 grub.cfg

GNU/Linux
GRUB2

2) 找回 grub.cfg
grub>ls ← 列出当前系统的分区
grub> ls (hd0,msdos1) ← 查看此分区信息
grub> ls (hd0,msdos1)/ ← 查看此分区的文件结构
grub> set root=(hd0,msdos1) ← 将此分区设置为 grub2 的启动分区
grub>set prefix=(hd0,msdos1)/grub2/
//* 如果 boot 不属于独立分区而是在 '/' 分区上则
//*grub>set
prefix=(hd0,msdos1)/boot/grub2

GNU/Linux
GRUB2

2) 找回 grub.cfg
grub>insmod /boot/grub2/i386-pc/normal.mod
grub>normal

//* 此时 grub.cfg 将出现在显示器上

GNU/Linux
GRUB2

3) 修改错误的 grub.cfg

grub>insmod /grub2/i386-pc/linux.mod

grub>set root=hd0,msdos1

grub>linux16 /vmlinuz-kernelID root=/dev/sda3

grub>initrd16 /initramfs- 内核版本号 .img

grub>boot

GNU/Linux
GRUB2

4) 修改 grub.cfg

进入系统后修改 grub.cfg 以确保可以正常启动

#vim /boot/grub2/grub.cfg

#grub2-install /dev/sda

GNU/Linux
GRUB2

为 grub 菜单加密 (明文)

1) 修改 /etc/grub.d/00_header, 在最低部加入

cat <<EOF

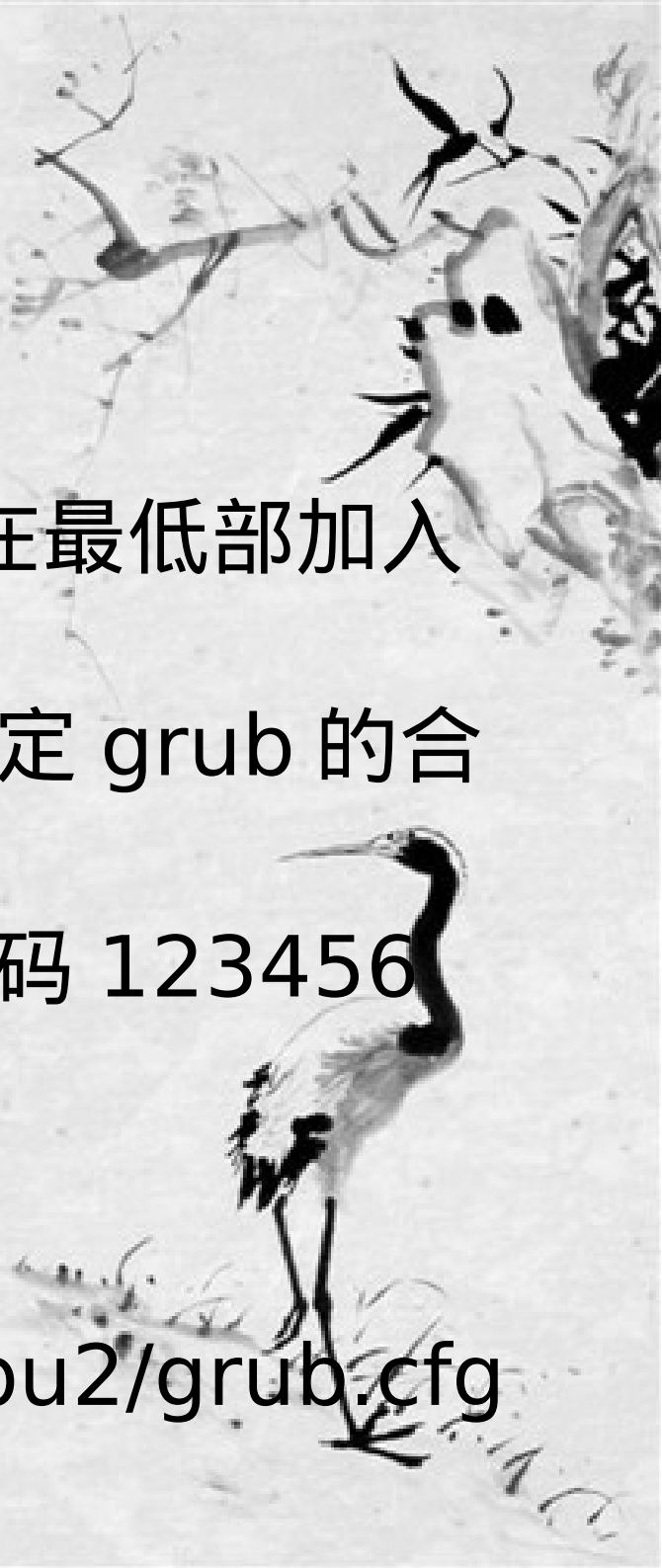
set superusers=“snow” ← 设定 grub 的合法用户

password snow 123456 ← 密码 123456

EOF

2) 重新生成 grub.cfg

#grub2-mkconfig > /boot/grbu2/grub.cfg



GNU/Linux
GRUB2

为 grub 菜单加密

1) 生成密码

#grub2-mkpasswd-pbkdf2

Enter password: 输入密码

Reenter password: 再次输入密码

2) 将所产生的密码行复制

GNU/Linux
GRUB2

为 grub 菜单加密

3) 修改 /etc/grub.d/00_header, 在最低部加入

cat <<EOF

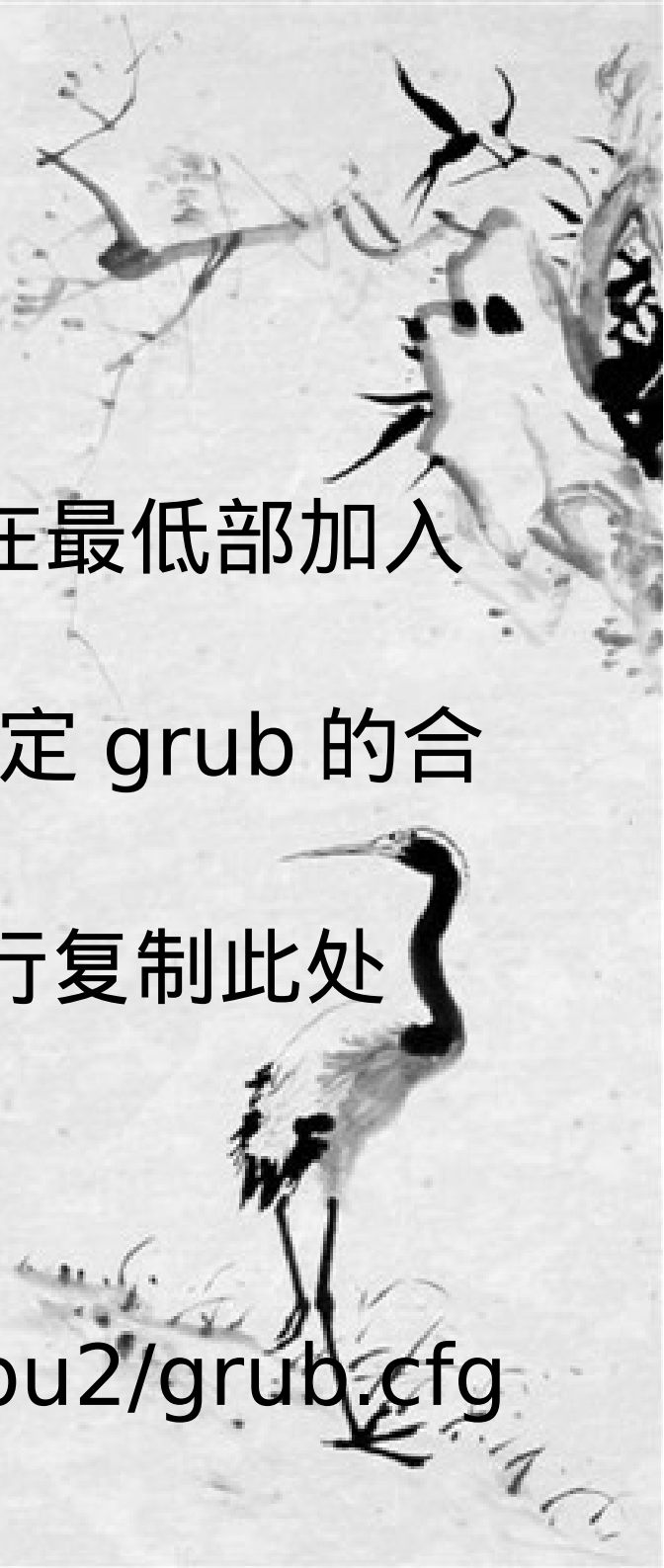
set superusers=“snow” ← 设定 grub 的合法用户

password_pbkdf2 snow 密码行复制此处

EOF

4) 重新生成 grub.cfg

#grub2-mkconfig > /boot/grbu2/grub.cfg



GNU/Linux
GRUB2

防止 grub2 更新而丢失口令 (grub2 更新将覆盖 00_header)

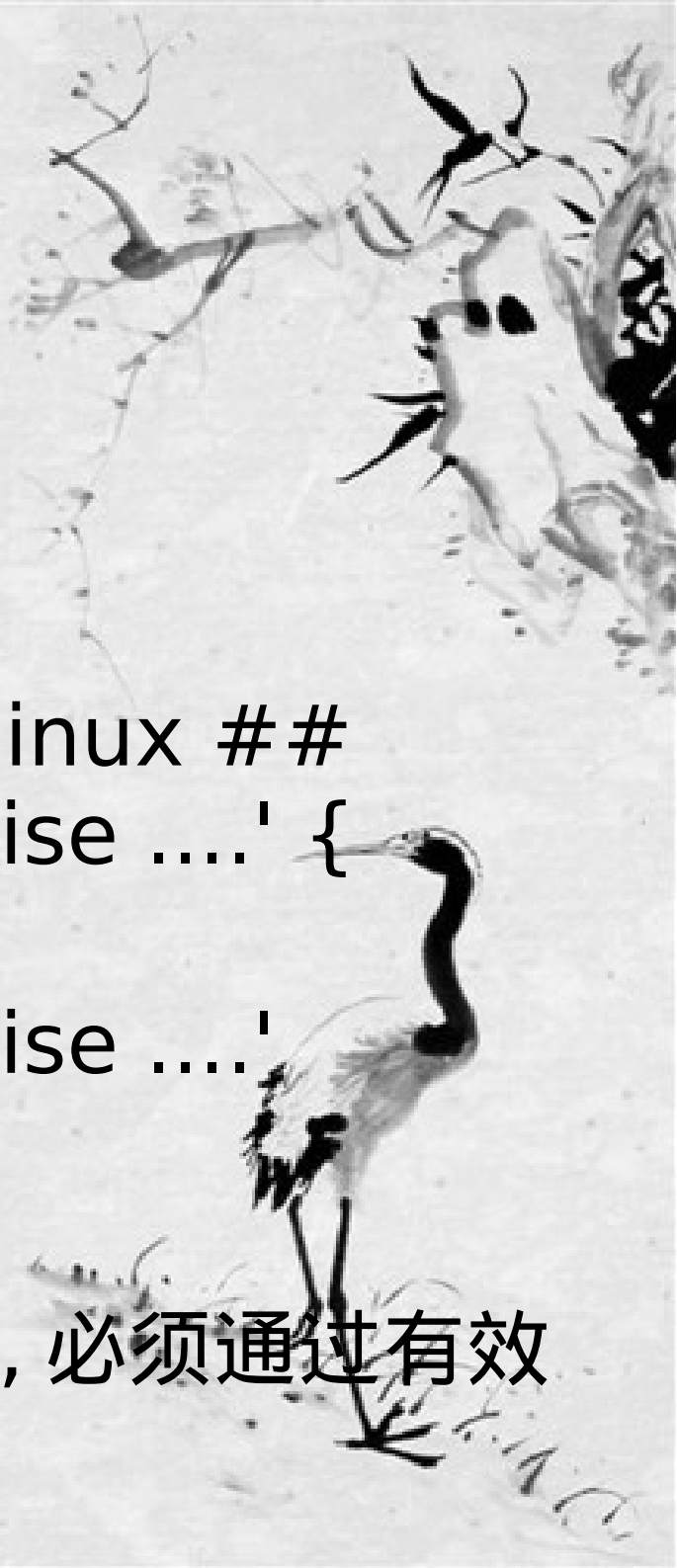
1) 自定义文件 uesrs 且级别作为 01

```
#vim /etc/grub.d/01_users
cat  <<EOF
set superusers="snow"
password_pbkdf2 snow  密码
EOF
```

2) 增加权限

```
#chmod 755 01_users
```

GNU/Linux
GRUB2



为操作系统增加口令 1

#vim /boot/grub2/grub.cfg

找到

BEGIN /etc/grub.d/10_linux

menuentry 'Red Hat Enterprise ....' {

改为

menuentry 'Red Hat Enterprise ....'
--users snow {

即意味着，当此指定系统启动时，必须通过有效认证方可启动

GNU/Linux
GRUB2

为操作系统增加口令 2

为防止每次生成 grub.cfg 时手工配置启动验证选项，可以通过 /etc/grub.d/10_linux 进行修改，作为以后生成 grub.cfg 的默认选项

GNU/Linux
GRUB2

为操作系统增加口令 2

#vim 10_linux

找到 (RHEL7.0 中此文件的 29 行)

CLASS="--class gnu-linux --class gnu --class of --unrestricted"

在最后增加 --users " 用户名", 修改后如下

CLASS="--class gnu-linux --class gnu --class of --unrestricted --users "snow""

GNU/Linux
GRUB2

对于非本系统，可以通过 /etc/grub.d/30_os-prober 或 /etc/grub.d/ 自定义文件 (40_custom) 等进行修改

#sed -i 's/--class os/--class os --users “snow”/' /etc/grub.d/30_os-prober

GNU/Linux
GRUB2

关闭 ipv6

#vim /etc/default/grub

第 6 行加入

GRUB_CMDLINE_LINUX="ipv6.disable=1
..."

使用 ethx

#vim /etc/default/grub

第 6 行加入

GRUB_CMDLINE_LINUX="net.ifnames=0
..."

GNU/Linux
rc.local

如果打算自动加载所需要执行的程序，同时欲恢复 rc.local 文件的功能可做如下操作：

#touch /etc/rc.d/rc.local

#chmod u+x /etc/rc.d/rc.local

#systemctl start rc-local