**第十七章 Linux系统启动原理及故障排除**

**本节所讲内容：**

**17.1 centos6系统启动过程及相关配置文件**

**17.2 centos7系统启动过程及相关配置文件**

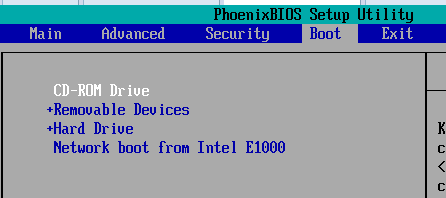
**17.3 实战-加密grub防止黑客通过单用户系统破解root密码**

**17.4 实战-通过liveCD进入救援模式-重装grub修复损坏的系统**

# 17.1 centos6系统启动过程及相关配置文件

## 17.1.1 centos6系统启动过程

**1. 加载 BIOS 的硬件信息，跟据设定取得第一个可开机引导设置，如：光驱，硬盘，网络，USB； 如果是硬盘为第一引导。**

****

**2. 读取硬盘中 MBR 的 boot Loader 就是 grub引导**

**GRUB（GRand Unified Bootloader简称“GRUB”）是一个来自GNU项目的多操作系统启动程序。**

**MBR的硬盘的0柱面、0磁头、1扇区称为主引导扇区（也叫主引导记录MBR）。它由三个部分组成，主引导程序、硬盘分区表DPT（Disk Partition table）和硬盘有效标志（55AA）。**

**互动：为什么MBR分区表，只能分4个主分区？**

**注：磁盘默认一个扇区大小为：512字节。MBR由以下3部分组成：**

**第一部分是：主引导程序（boot loader）占446个字节。主引导程序，它负责从活动分区中装载，并运行系统引导程序。**

**第二部分是Partition table区（分区表），即DPT，占64个字节，硬盘中分区有多少以及每一分区的大小都记在其中。每个分区表项长16个字节，16\*4=64字节。为分区项1、分区项2、分区项3、分区项4。64字节只存4个分区表。**

**第三部分是MBR有效标识位，占2个字节，固定为55AA。如果这个标志位0xAA55，就认为这个是MBR。**

**所以：16\*4+446+2=512**

**3. 依据 boot loader 的设定，到引导分区加载 Kernel ，Kernel 会开始侦测硬件并加载驱劢程序；**

**4. 在硬件驱动成功后，Kernel 会主动执行 init 程序，而 init 会取得 run-level 信息；**

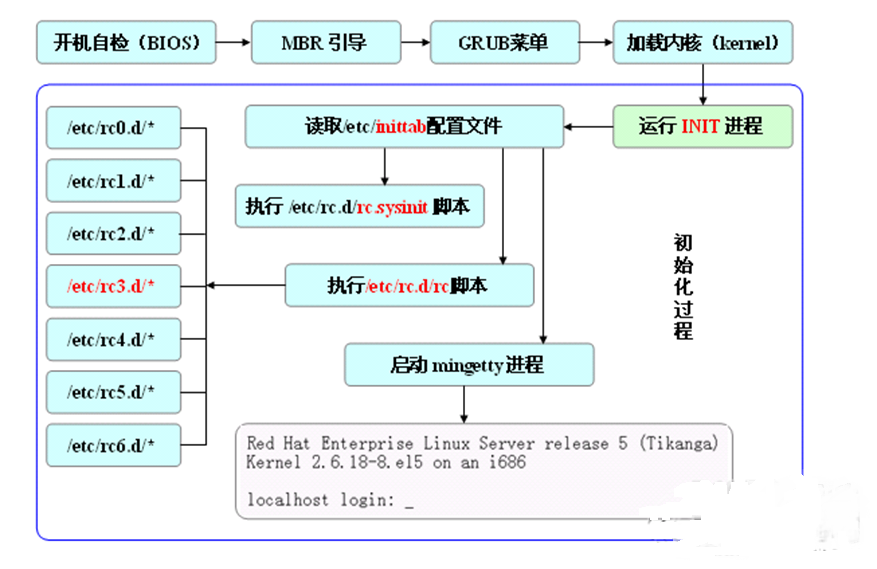
**5. init 执行 /etc/rc.d/rc.sysinit 文件来准备软件执行的作业环境 (如网络、时区等)；**

**6. init 执行 run-level 下各个服务并启动 (script 方式)；**

**7. init 执行开机后自动运行脚本 /etc/rc.d/rc.local 文件；**

**8. init 执行虚拟终端机控制程序 mingetty 来启动 login 程序，最后就等待用户登入啦；**

**如图：**

****

## 17.1.2 centos6启动相关的配置文件

**[root@xuegod64 Desktop]# vim /boot/grub/grub.conf**

**default=0 设定默认启动菜单项，当系统中有多个内核时，0表示默认加载第1个，1表示第2个内核**

**timeout=5 菜单项等待选项时间为5s**

**splashimage=(hd0,0)/grub/splash.xpm.gz 指明菜单背景图片路径为**

**hiddenmenu 隐藏菜单**

**title CentOS (2.6.32-358.6.1.el6.x86\_64) 定义菜单项**

**root (hd0,0) grub查找stage2及kernel文件所在设备分区，grub的根**

**kernel /vmlinuz-2.6.32-358.6.1.el6.x86\_64 ro root=/dev/vg\_have/lv\_root rd\_NO\_LUKS LANG=en\_US.UTF-8 SYSFONT=latarcyrheb-sun16 crashkernel=auto rhgb quiet 启动的内核**

**initrd /initramfs-2.6.32-358.6.1.el6.x86\_64.img 内核匹配的ramfs文件**

**修改系统启动级别：**

**[root@xuegod64 Desktop]# vim /etc/inittab**

**# Default runlevel. The runlevels used are:**

**# 0 - halt (Do NOT set initdefault to this)**

**# 1 - Single user mode**

**# 2 - Multiuser, without NFS (The same as 3, if you do not have networking)**

**# 3 - Full multiuser mode**

**# 4 - unused**

**# 5 - X11**

**# 6 - reboot (Do NOT set initdefault to this)**

**#**

**id:3:initdefault: #这里决定系统启动的级别**

**/etc/rc.d/rc.sysinit shell脚本 作用：系统初始化： 像：主机名 和/etc/fstab 都在这里指定了，完成了包括mount分区 激活swap 加载modules等重要的工作.**

**启动对应级别下的服务如： init 3 级别**

**/etc/rc.d/rc3.d/（这里的程序/服务S开头的全部开机执行；K开头的表示开机不执行，表明了关机时顺序）**

**rcn.d (n为1到6) 是对应于不同的runlevel下起不同的服务. 这些目录下都是一些符号连接, 连接到/etc/rc.d/init.d下的一些文件.以S开头的表示要启动, 以K开头的不启动.   
第一个字母后面的数值是一个优先级.**

**[root@xuegod63 ~]# ll /etc/rc.d/rc3.d/ | grep network**

**lrwxrwxrwx. 1 root root 17 Dec 18 2012 S10network -> ../init.d/network #表示network是第10个启动的服务。 所以init是顺序启动系统，需要一个一个服务启动成功，再执行下一步操作，启动系统比较慢。而centos7中的systemd可以并行启动多个服务，启动比较快。**

**例：**

**[root@xuegod63 rc3.d]# vim /etc/init.d/network**

**#! /bin/bash**

**#**

**# network Bring up/down networking**

**#**

**# chkconfig: 2345 10 90  
看有chkconfig的那一行, 2345表示在runlevel 2 3 4 5下被启动, 10是为此服务的启动顺序, 90为关机时，关闭此服务的顺序。**

**[root@xuegod63 ~]# chkconfig --list | grep network**

**network 0:off 1:off 2:on 3:on 4:on 5:on 6:off**

**[root@xuegod63 ~]# ll /etc/rc.d/rc3.d/ | grep network**

**lrwxrwxrwx. 1 root root 17 Dec 18 2012 S10network -> ../init.d/network #开机顺序**

**[root@xuegod63 ~]# chkconfig network off**

**[root@xuegod63 ~]# ll /etc/rc.d/rc3.d/ | grep network**

**lrwxrwxrwx 1 root root 17 May 23 21:17 K90network -> ../init.d/network #只显示k90关机顺序了**

**[root@xuegod64 rc3.d]# chkconfig --list network**

**network 0:off 1:off 2:off 3:off 4:off 5:off 6:off**

**所有服务都运行成功后，设置开机自动执行某个命令： /etc/rc.local**

**[root@xuegod64 rc3.d]# vim /etc/rc.local**

**[root@xuegod64 rc3.d]# ll !$**

**ll /etc/rc.local**

**lrwxrwxrwx. 1 root root 13 Dec 18 2012 /etc/rc.local -> rc.d/rc.local**

**[root@xuegod64 rc3.d]# ll /etc/rc.d/rc.local**

**-rwxr-xr-x. 1 root root 240 Feb 5 21:17 /etc/rc.d/rc.local**

**运行mingetty命令，打开tty1-6**

**[root@xuegod64 rc3.d]# ps -axu | grep ming**

**Warning: bad syntax, perhaps a bogus '-'? See /usr/share/doc/procps-3.2.8/FAQ**

**root 2346 0.0 0.0 4116 548 tty2 Ss+ 20:55 0:00 /sbin/mingetty /dev/tty2**

**root 2348 0.0 0.0 4116 548 tty3 Ss+ 20:55 0:00 /sbin/mingetty /dev/tty3**

**root 2350 0.0 0.0 4116 544 tty4 Ss+ 20:55 0:00 /sbin/mingetty /dev/tty4**

**root 2352 0.0 0.0 4116 544 tty5 Ss+ 20:55 0:00 /sbin/mingetty /dev/tty5**

**root 2354 0.0 0.0 4116 544 tty6 Ss+ 20:55 0:00 /sbin/mingetty /dev/tty6**

**[root@xuegod63 ~]# runlevel #查看系统启动级别**

**N 5**

**[root@xuegod63 ~]# init 3**

**[root@xuegod63 ~]# runlevel**

**5 3 #由5启动级别进入3级别**

**[root@xuegod63 ~]# init 5**

**[root@xuegod63 ~]# runlevel**

**3 5 #由3启动级别进入5级别**

# 17.2 centos7系统启动过程及相关配置文件

## 17.2.1 centos7系统启动过程

**CentOS7引导顺序**

**1. UEFi或BIOS初始化，运行POST开机自检**

**2. 选择启动设备**

**3. 引导装载程序, centos7是grub2**

**4. 加载装载程序的配置文件：/etc/grub.d/ /etc/default/grub /boot/grub2/grub.cfg**

**5. 加载内核选项**

**6. 加载initramfs初始化伪文件系统**

**7. 内核初始化，centos7使用systemd代替init**

**8. 执行initrd.target所有单元，包括挂载/etc/fstab**

**9. 从initramfs根文件系统切换到磁盘根目录**

**10. systemd执行默认target配置，配置文件/etc/systemd/system/default.target**

**11. systemd执行sysinit.target初始化系统及basic.target准备操作系统**

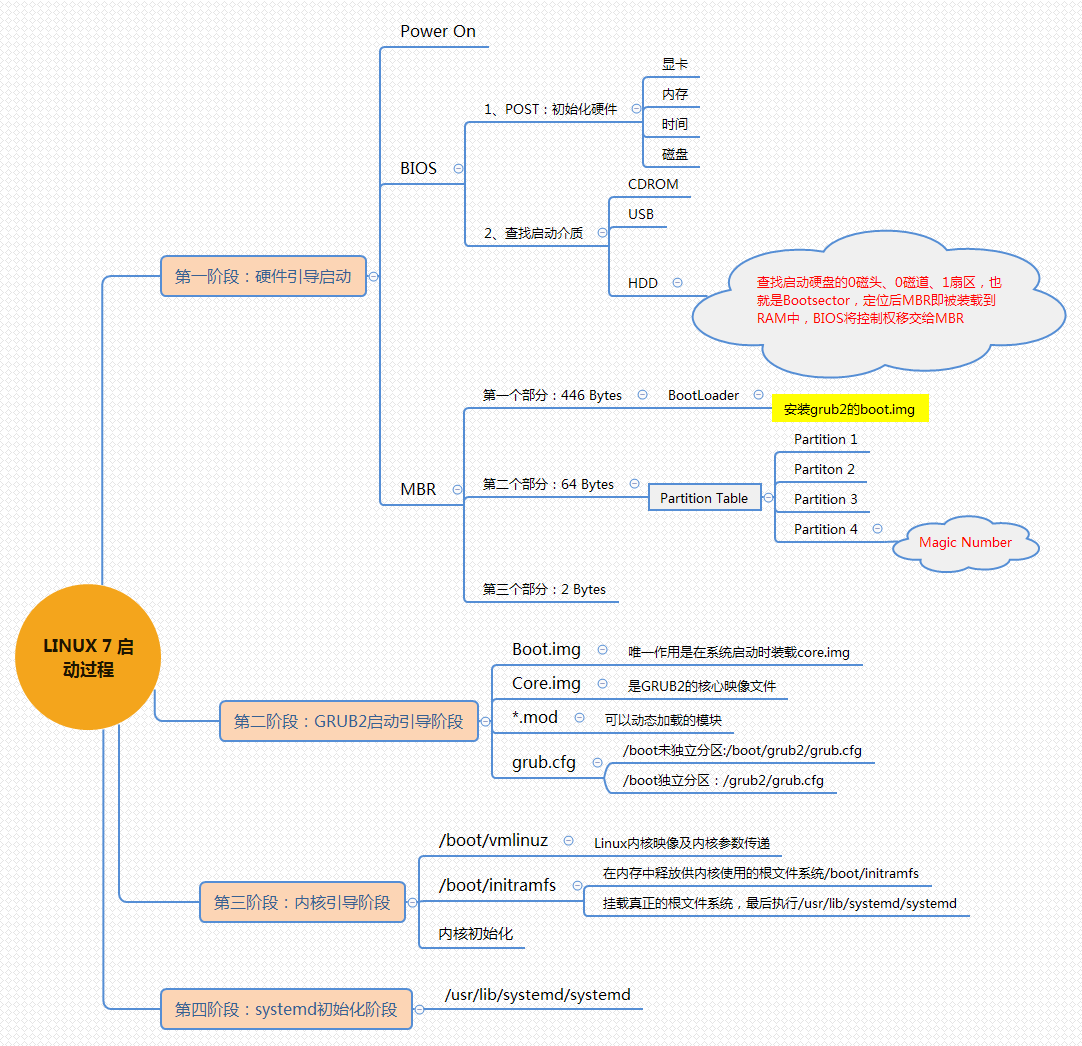
**12. systemd启动multi-user.target下的本机与服务器服务**

**13. systemd执行multi-user.target下的/etc/rc.d/rc.local**

**14. Systemd执行multi-user.target下的getty.target及登录服务**

**15. systemd执行graphical需要的服务**

**centos7启动过程：**

****

**[root@xuegod63 logs]# find /boot/ -name \*img\***

**/boot/grub2/i386-pc/core.img**

**/boot/grub2/i386-pc/boot.img**

## 17.2.2 Systemd运行原理-了解一下

**Systemd概述：systemd即为system daemon [ˈdi:mən]** **守护进程，是linux下的一种init软件,开发目标是提供更优秀的框架以表示系统服务间的依赖关系，并依此实现系统初始化时服务的并行启动，同时达到降低Shell的系统开销的效果，最终代替现在常用的System V与BSD风格init程序。**

**与多数发行版使用的System V风格init相比，systemd采用了以下新技术： (1) 采用Socket激活式与总线激活式服务，以提高相互依赖的各服务的并行运行性能； (2) 用Cgroups代替PID来追踪进程，以此即使是两次fork之后生成的守护进程也不会脱离systemd的控制。**

**unit对象：unit表示不同类型的systemd对象，通过配置文件进行标识和配置；文件中主要包含了系统服务、监听socket、保存的系统快照以及其它与init相关的信息**

**Systemd配置文件：**

**• /usr/lib/systemd/system/ #这个目录存储每个服务的启动脚本，类似于之前的/etc/init.d/**

**• /run/systemd/system/ #系统执行过程中所产生的服务脚本，比上面目录优先运行**

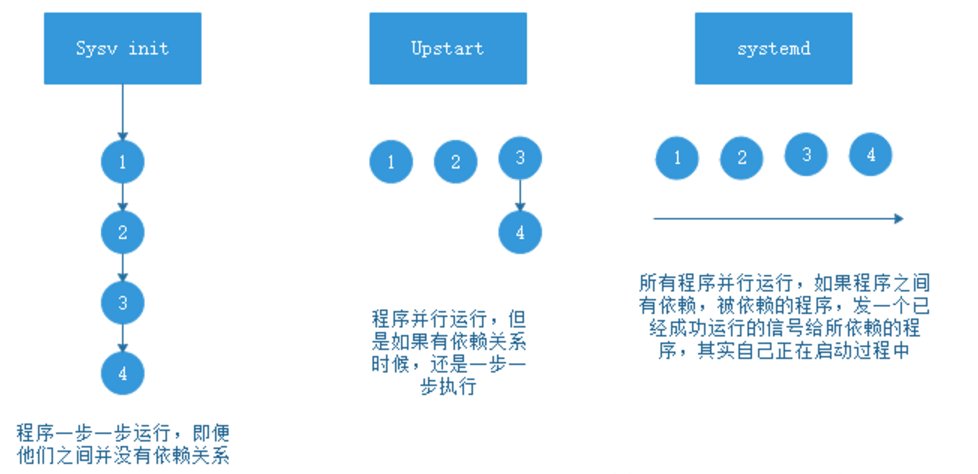
**• /etc/systemd/system/ #管理员建立的执行脚本，类似于/etc/rc.d/rcN.d/Sxx类的功能，比上面目录优先运行**

**注意： 对于新创建的unit文件，或者修改了的unit文件，要通知systemd重载此配置文件,而后可以选择重启**

**[root@xuegod63 ~]# systemctl daemon-reload**

**总结：centos5-6-7 3个系统版本启动过程：**

**CentOS 5: SysV init ； CentOS 6: Upstart ；CentOS 7: Systemd**

****

## 17.2.3 管理系统服务

**命令： systemctl COMMAND name.service**

| **-** | **centOS6** | **CentOS7** |
| --- | --- | --- |
| **启动** | **service name start** | **systemctl start name.service** |
| **停止** | **service name stop** | **systemctl stop name.service** |
| **重启** | **service name restart** | **systemctl restart name.service** |
| **状态** | **service name status** | **systemctl status name.service** |
| **重载或重启服务(先加载，再启动)** | **-** | **systemctl reload-or-restart name.service** |

**chkconfig命令的对应关系**

| **-** | **centOS6** | **CentOS7** |
| --- | --- | --- |
| **设定某服务开机自启** | **chkconfig name on** | **systemctl enable name.service** |
| **设定某服务开机禁止启动** | **chkconfig name off** | **systemctl disable name.service** |
| **查看所有服务的开机自启状态** | **chkconfig --list** | **systemctl list-unit-files --type service** |
| **用来列出该服务在哪些运行级别下启用和禁用** | **chkconfig sshd –list** | **ls /etc/systemd/system/\*.wants/sshd.service** |
| **查看服务是否开机自启** | **-** | **systemctl is-enabled name.service** |

**服务状态**

**[root@xuegod63 ~]# systemctl list-unit-files #显示状态**

**• loaded:Unit配置文件已处理**

**• active(running):一次或多次持续处理的运行**

**• active(exited):成功完成一次性的配置**

**• active(waiting):运行中，等待一个事件**

**• inactive:不运行**

**• enabled:开机启动**

**• disabled:开机不启动**

**• static:开机不启动，但可被另一个启用的服务激活**

## 17.2.4 运行级别

**centos6下Linux运行级别0-6的各自含义**

**0： 关机模式**

**1：单用户模式 ，用于破解root密码**

**2：无网络，支持的多用户模式**

**3：有网络支持的多用户模式（一般叫字符界面，工作中最长使用的模式）**

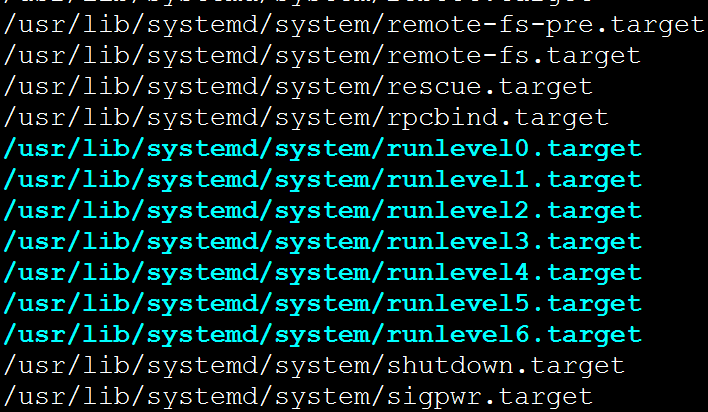
**4：保留，未使用**

**5：有网络支持，支持图形界面，支持的多用户模式（图形界面）**

**6：重新引导系统，及重启**

**可以在不同级别下，设置服务是否随系统启动运行。在centOS7上运行级别的含义已经和之前不同了，已由.target来代替运行级别，我们可以称target为目标态，我们可以通过target定制更符合我们工作运行环境。**

**[root@xuegod63 ~]# ls /usr/lib/systemd/system/\*.target #查看我们的机器上有多少个target**



**[root@xuegod63 ~]# ll /usr/lib/systemd/system/\*.target | grep runlevel**

**lrwxrwxrwx. 1 root root 15 9月 19 2017 /usr/lib/systemd/system/runlevel0.target -> poweroff.target**

**lrwxrwxrwx. 1 root root 13 9月 19 2017 /usr/lib/systemd/system/runlevel1.target -> rescue.target**

**lrwxrwxrwx. 1 root root 17 9月 19 2017 /usr/lib/systemd/system/runlevel2.target -> multi-user.target**

**lrwxrwxrwx. 1 root root 17 9月 19 2017 /usr/lib/systemd/system/runlevel3.target -> multi-user.target**

**lrwxrwxrwx. 1 root root 17 9月 19 2017 /usr/lib/systemd/system/runlevel4.target -> multi-user.target**

**lrwxrwxrwx. 1 root root 16 9月 19 2017 /usr/lib/systemd/system/runlevel5.target -> graphical.target**

**lrwxrwxrwx. 1 root root 13 9月 19 2017 /usr/lib/systemd/system/runlevel6.target -> reboot.target**

**注： 发现在runlevel2-4 都是调用multi-user.target这个unit。所以在centos7上runlevel2-4是一个意思**

**[root@xuegod63 ~]# systemctl list-unit-files --type target #查看所有target的状态**

**[root@xuegod63 ~]# systemctl list-dependencies runlevel3.target #查看3级别Unit 的所有依赖。Unit 之间存在依赖关系：A 依赖于 B，就意味着 Systemd 在启动 A 的时候，同时会去启动 B。也可以理解也3运行级别下都开启哪些服务**

**在centOS7上所谓的目标态，其实就是由各种指定的服务和基础target组合而成的。**

**总结：centos6和7运行级别的变化**

**6 7**

**init systemd**

**Traditional runlevel | New target name Symbolically linked to...**

**Runlevel 0 | runlevel0.target -> poweroff.target**

**Runlevel 1 | runlevel1.target -> rescue.target**

**Runlevel 2 | runlevel2.target -> multi-user.target**

**Runlevel 3 | runlevel3.target -> multi-user.target**

**Runlevel 4 | runlevel4.target -> multi-user.target**

**Runlevel 5 | runlevel5.target -> graphical.target**

**Runlevel 6 | runlevel6.target -> reboot.target**

**Init 0 🡪 systemctl poweroff 关机**

**Init 1 🡪 systemctl isolate rescue.target 单用户**

**Init 3 🡪 systemctl isolate multi-user.target 字符界面**

**Init 5 🡪 systemctl isolate graphical.target 图形化**

**Init 6 🡪 systemctl reboot 重启**

## 17.2.5 运行级别的切换

**1、在centOS6上，我们切换级别使用init，在centOS7上虽然也能使用，但是调用的不再是原来的程序了。centos7使用systemctl isolate name.target来切换target。 # isolate [ˈaɪsəleɪt] 分离，隔离**

**例1：在centos6/7下切换到字符界面：**

**[root@xuegod63 ~]# init 3 #切换到字符界面**

**[root@xuegod63 ~]# init 5 #切换到图形界面**

**例2：centos7切换到字符界面**

**[root@xuegod63 ~]# systemctl isolate multi-user.target**

**或：**

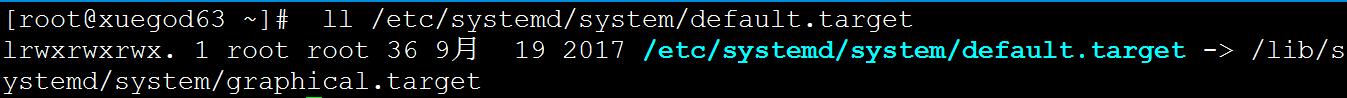
**[root@xuegod63 ~]# systemctl isolate runlevel3.target**

**2、centos7设置默认系统默认启动级别**

**systemctl set-default name.target来修改我们的目标态。**

**我们看一下我们的默认目标态究竟为何物。**

**[root@xuegod63 ~]# ll /etc/systemd/system/default.target**



**注：它其实就是创建了一个软链接到指定的target上去了**

**例1：默认系统启动使用3级别字符界面**

**[root@xuegod63 ~]# systemctl set-default multi-user.target**

**Removed symlink /etc/systemd/system/default.target.**

**Created symlink from /etc/systemd/system/default.target to /usr/lib/systemd/system/multi-user.target.**

**[root@xuegod63 ~]# ll /etc/systemd/system/default.target #查看链接**

**lrwxrwxrwx 1 root root 41 5月 23 19:08 /etc/systemd/system/default.target -> /usr/lib/systemd/system/multi-user.target**

**例2：默认系统启动使用5级别图形界面**

**[root@xuegod63 ~]# systemctl set-default graphical.target**

## 17.2.6 grub2和grub区别-了解

**在centOS6上，我们的grub文件是/boot/grub/grub.conf**

**在centOS7使用grub2，配置文件改成/boot/grub2/grub.cfg了，但是功能还是大致一样的都是用于加载内核的，不过在centOS7上设置默认启动项发生了一些变化。**

**互动：如果我们的系统中有两个内核？改变默认启动的内核顺序？**

**例1: centos7修改内核启动顺序**

**[root@xuegod63 ~]# vim /etc/default/grub**

**GRUB\_TIMEOUT=5 #开机时** **grub 默认5秒后启动内核**

**GRUB\_DISTRIBUTOR="$(sed 's, release .\*$,,g' /etc/system-release)"**

**改：GRUB\_DEFAULT=** **saved**

**为：GRUB\_DEFAULT=** **1 #这里我们改成1，0代表第一个内核，1代表第二个，以此类推。**

**UB\_DISABLE\_SUBMENU=true**

**GRUB\_TERMINAL\_OUTPUT="console"**

**GRUB\_CMDLINE\_LINUX="crashkernel=auto rhgb quiet net.ifnames=0"**

**GRUB\_DISABLE\_RECOVERY="true"**

**[root@xuegod63 ~]# grub2-mkconfig -o /boot/grub2/grub.cfg #修改完成后，并没有立即生效，使用此命令来生成grub.cfg文件，我们在下次启动的时候就会默认选择新的默认内核。**

**[root@xuegod63 ~]# uname -r #查当前系统内核**

**3.10.0-693.2.2.el7.x86\_64**

**[root@xuegod63 ~]# reboot**

**[root@xuegod63 ~]# uname -r #重启成功后， 发现加载的内核变了**

**3.10.0-693.el7.x86\_64**

**例2: centos6修改内核启动顺序-了解**

**[root@xuegod63 ~]# vim /boot/grub/grub.conf**

**改：10 default=0**

**为：10 default=1**

**[root@xuegod63 ~]# reboot**

# 17.3 实战-加密grub防止黑客通过单用户系统破解root密码

**实战场景：如何防止别人恶意通过单用户系统破解root密码，进入系统窃取数据？**

**给grub加密，不让别人通过grub进入单用户。**

## 17.3.1 基于centos6进行grub加密

**[root@xuegod63 ~]# grub-md5-crypt**

**Password: 123456**

**Retype password: 123456**

**$1$oaqo5$3d/cmTosm68jTw6o1wCu31**

**[root@localhost init]# vim /boot/grub/grub.conf**

**#boot=/dev/sda**

**default=0**

**timeout=5**

**splashimage=(hd0,0)/grub/splash.xpm.gz**

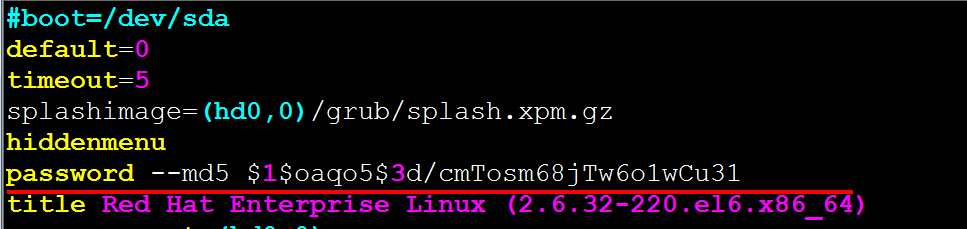
**hiddenmenu**

**password --md5 $1$oaqo5$3d/cmTosm68jTw6o1wCu31**

**title Red Hat Enterprise Linux (2.6.32-220.el6.x86\_64)**

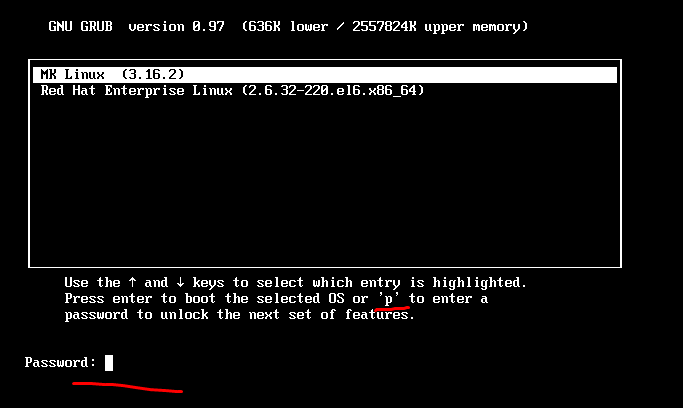
**root (hd0,0)**

**如图：**



**重启测试：**

**编辑grub时，需要按下p键，然后输入密码：123456**



## 17.3.2 基于centos7进行grub加密

**生成密码**

**[root@xuegod63 ~]# grub2-mkpasswd-pbkdf2**

**输入口令： 123456**

**Reenter password: 123456**

**PBKDF2 hash of your password is grub.pbkdf2.sha512.10000.8F355BAB512AFB7B8C990A1FEB887B8F2F3F1C54467E9B9F0535F2268E1FFC5F4E8D33F7633D7FBEC25B2039C6D8B3226A90528D4883AB9B99E391A4965D069F.DDE992693BE2C09FFEEC1149120B6B84DBAB933DE6CF7BFF718E1DDC858AB73EE32CFF45EB7F06AC45AA6792E91C4CD09E2B445FC288C47E79F537DBBABAD756**

**[root@xuegod63 ~]# vim /etc/grub.d/00\_header #在最后后面添加如下内容，注mk这个用户名可以换成自己的用户名**

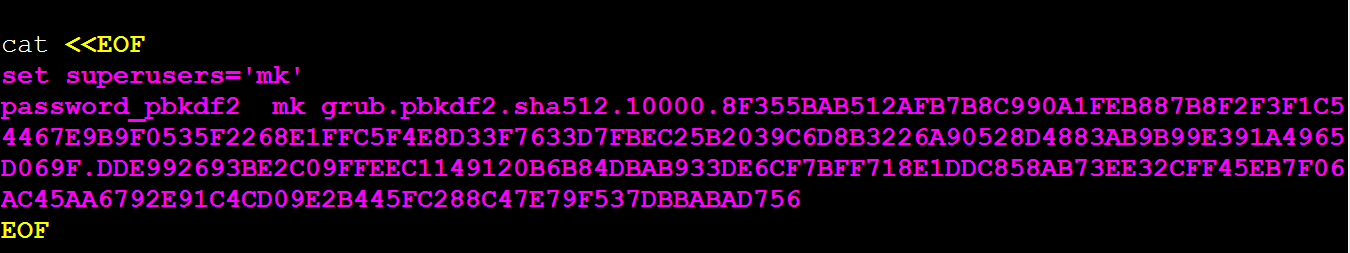
**cat <<EOF**

**set superusers='mk'**

**password\_pbkdf2 mk grub.pbkdf2.sha512.10000.8F355BAB512AFB7B8C990A1FEB887B8F2F3F1C54467E9B9F0535F2268E1FFC5F4E8D33F7633D7FBEC25B2039C6D8B3226A90528D4883AB9B99E391A4965D069F.DDE992693BE2C09FFEEC1149120B6B84DBAB933DE6CF7BFF718E1DDC858AB73EE32CFF45EB7F06AC45AA6792E91C4CD09E2B445FC288C47E79F537DBBABAD756**

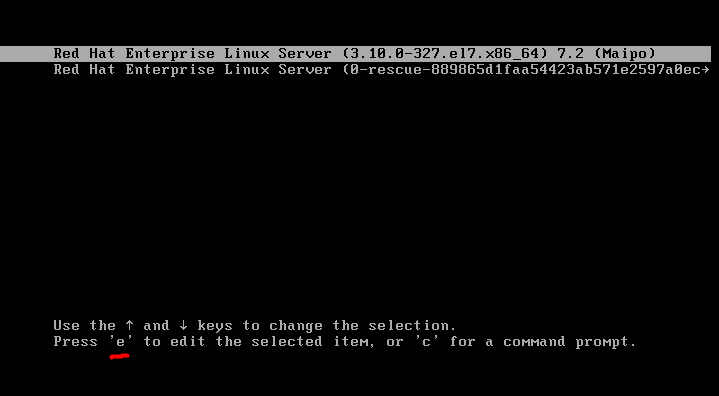
**EOF**

**如下图：**

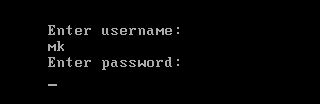


**[root@xuegod63 ~]# grub2-mkconfig -o /boot/grub2/grub.cfg #更新grub信息**

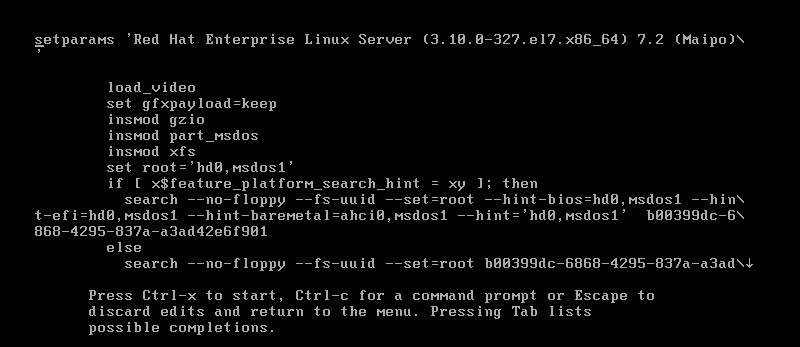
**重启验证：**

****

**输入用户名和密码**



**看到可以进入GRUB菜单，就证明你加密成功了**

****

**按ctrl-x 开始启动**

# 17.4 实战-通过liveCD进入救援模式-重装grub修复损坏的系统

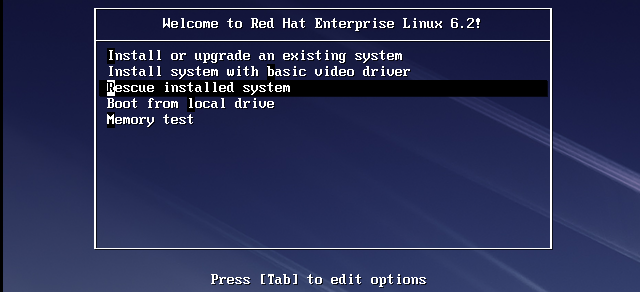
**实战：使用系统光盘进入救援模式拯救坏掉的系统**

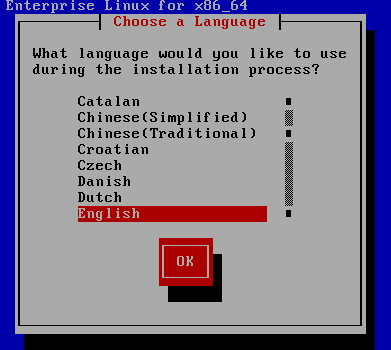
**实战场景：当系统坏了，进不去了，还需要把里面的数据复制出来，怎么办？**

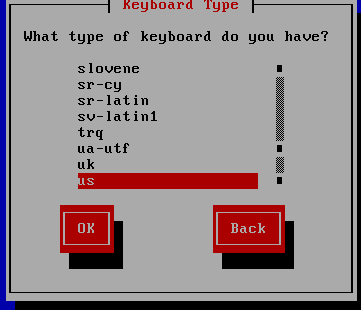
**可以进入救援模式拷贝数据**

## 17.4.1 基于6版本系统进入救援模式

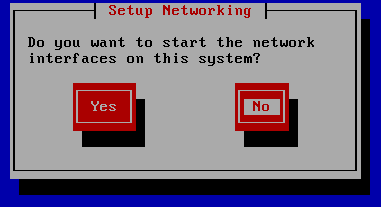
**修改BIOS启动顺序，直接以光盘引导系统**

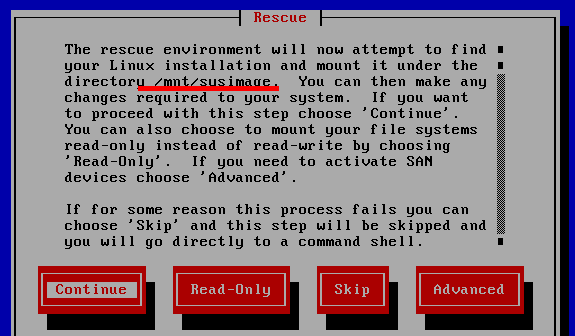


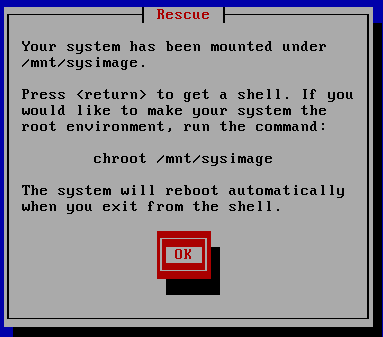






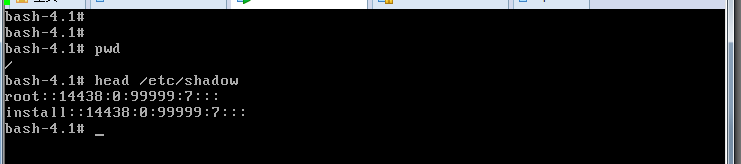






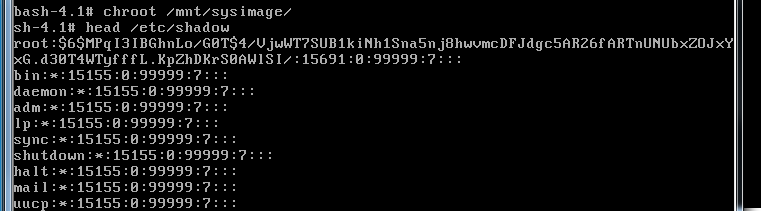






**ramfs ： 内存文件系统**

**chroot /mnt/sysimage # 切换文件系统根**



## 17.4.2 实战-当MBR引导记录损坏后-重装grub进行修复

**使用场景： 修复MBR，主要出现在安装双系统时，后安装的系统把原来系统的MBR删除了，需要修复。**

**第一步：在centOS7下破坏硬盘的前446字节：**

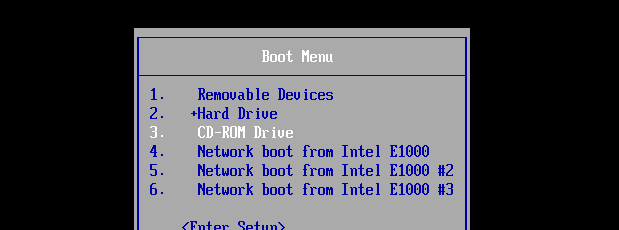
**[root@CT731 ~]#dd if=/dev/zero of=/dev/sda bs=1 count=446**

**446+0 records in**

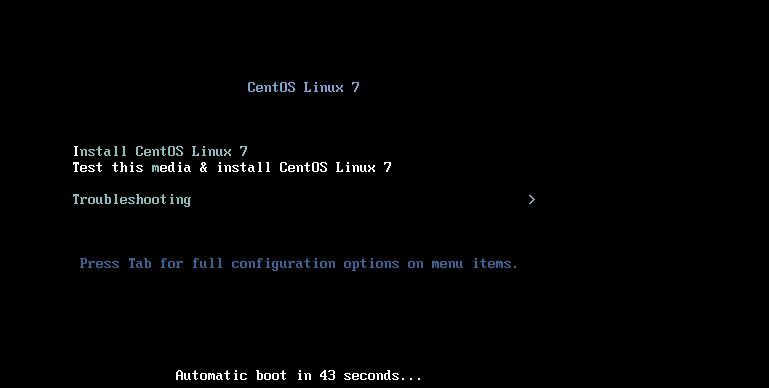
**446+0 records out**

**446 bytes (446 B) copied,0.000758682 s,588 kB/s**

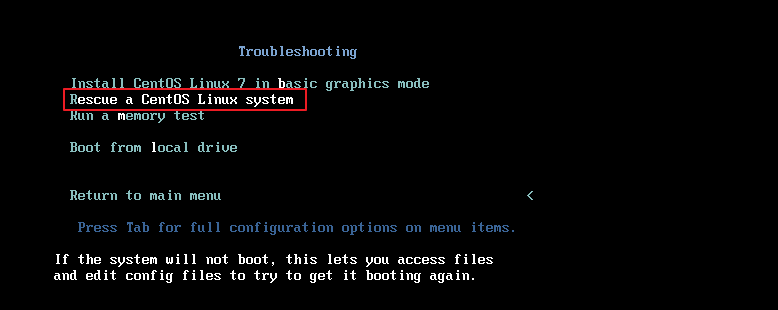
**第二步：将centos7系统光盘挂载到虚拟机光驱上，，重启计算机，修改BIOS引导顺序，让光盘启动。**

****

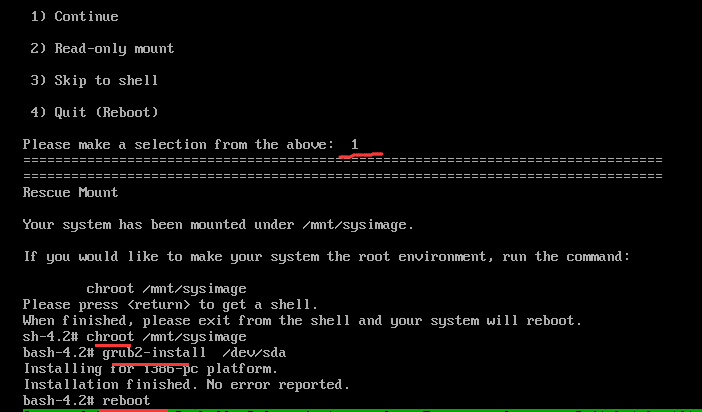
**进入启动的界面**

****

**上面有三项，我们选择第三项进入troubleshooting故障排除界面 ，进入第三项后，点击第二项，进入救援模式的centos的系统**

****

**然后我们进入如下模式，选择1，继续进行，接下来，我们就会进入到一个shell模式中，需要切换根目录，进行系统修复：**



**先退一下，再重启，修复完成**



## 17.4.3 实战-在centOS7下误删除grub文件进行修复

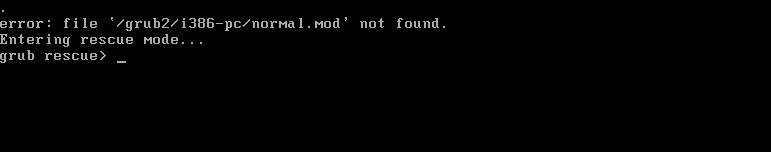
**第一步：删除grub2**

**[root@xuegod63 ~]# rm -rf /boot/grub2**

**第二步，重启计算机**

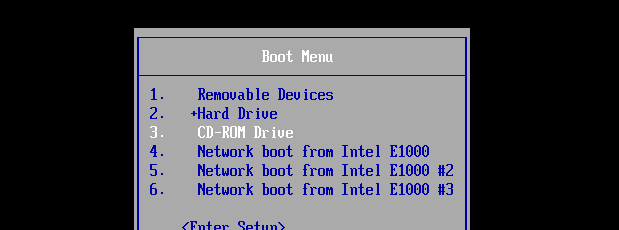
**[root@xuegod63 ~]# reboot**

**进入如下界面：**

****

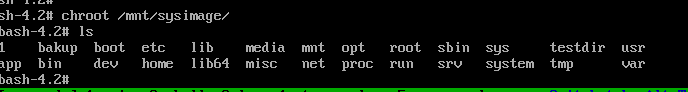
**现在开始解决grub**

**重启系统，按Esc，进入光盘救援模式，选择第三项，进入光盘救援（前提是挂载光盘）**

****

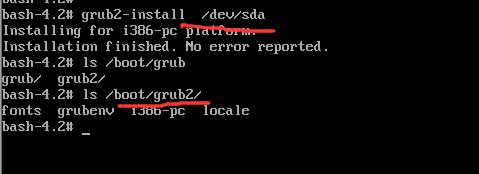
**使用live cd 进入救援模式后：**

**第一步：切根**

****

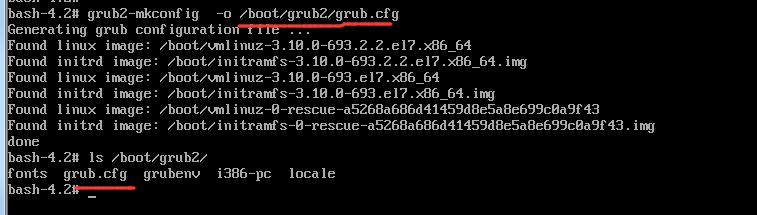
**然后执行命令**

**grub2-install**

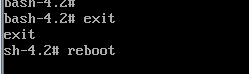


**下图中，我们可以看到在grub2文件夹中，还没有grub.cfg文件，接下来，我们需要生成配置文件：**

**进入到grub2下，**



**exit然后，重启电脑：**



**修改BIOS 引导，让硬盘做第一引导**

**注： centos系统下载链接  IMG_256http://vault.centos.org/ 大家可以在这里下载centos6相关的系统。**

**总结:**

**17.1 centos6系统启动过程及相关配置文件**

**17.2 centos7系统启动过程及相关配置文件**

**17.3 实战-加密grub防止黑客通过单用户系统破解root密码**

**17.4 实战-通过liveCD进入救援模式-重装grub修复损坏的系统**