第 7 讲　Linux 系统配置与管理

王晓庆

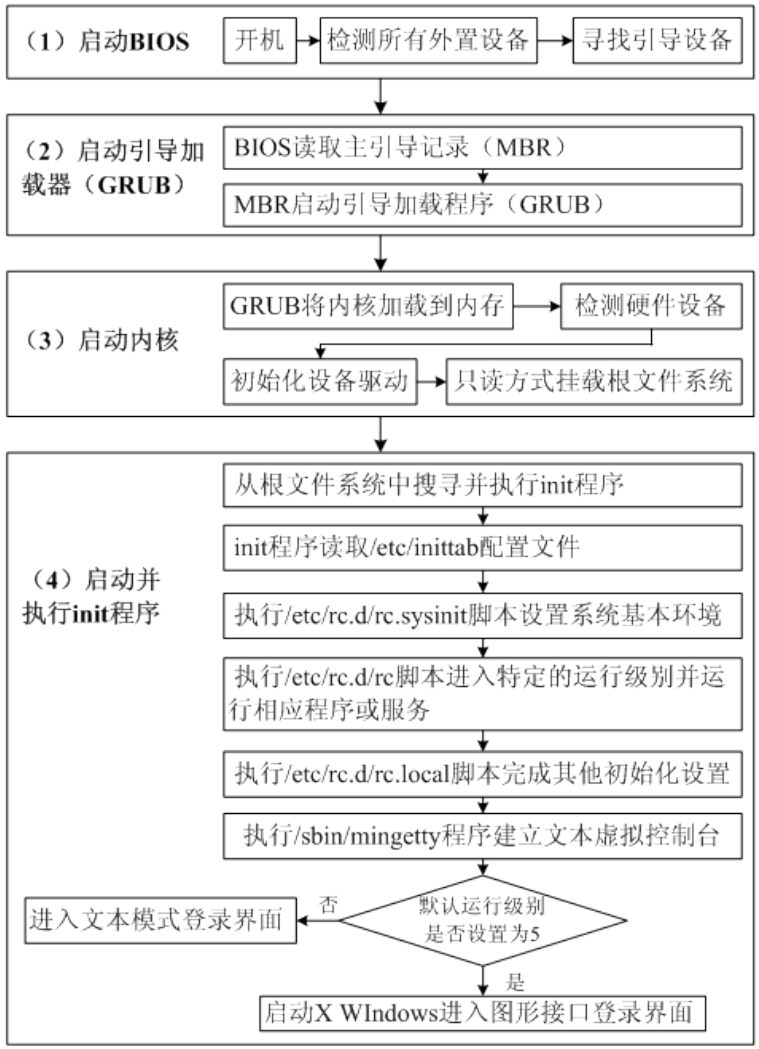
wangxiaoqing@outlook.com

May 8, 2016

Outline

Linux 启动过程

- Linux 启动步骤
  1. 启动 BIOS
  2. 启动引导
  3. 启动内核
  4. 启动并执行 init 程序



引导加载程序 grub(1)

- /boot/grub/grub.conf(符号链接：/etc/grub.conf)

```
default=0 # 默认启动的操作系统编号
timeout=5 # 等待用户选择的时间（秒）
# 引导菜单背景图案
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu # 隐藏引导菜单
title CentOS (2.6.18-398.e15) # 引导项标题
        root (hd0,0)                # 内核所在分区
        # 内核文件及内核启动参数
        kernel /vmlinuz-2.6.18-398.e15 ro root=LABEL=/
        # 内存磁盘镜像文件
        initrd /initrd-2.6.18-398.e15.img
```

引导加载程序 grub(2)

- 动态修改 grub 引导参数
  - a # 添加引导参数
  - e # 编辑 grub 菜单
  - c # 进入 grub 命令行模式
  - 例：忘记 root 口令解决办法
    在 root 行后加空格和 1(或 single)
- 设置 grub 密码
  - 明文密码
    password 明文密码
  - 加密密码
    1. grub-md5-crypt 生成加密密码
    2. password –md5 加密密码

Linux 运行级别

级别	说明	进入该级别所执行的任务	可登录用户	网络功能
0	关机。不要将默认运行级别设置为此级别	关闭所有可登录的虚拟控制台，强迫用户注销系统；结束所有启动的服务；卸载所有文件系统；停用所有外围设备	无	
1	单用户模式。以root身份开启一个虚拟控制台，主要用于管理员维护系统	关闭所有可登录的虚拟控制台；关闭网络；关闭大部分服务与应用程序	仅root登录，无需口令	不支持
2	多用户模式，不支持NFS。除了不启用网络功能，与Runlevel 3相同	启动网络；启动大部分网络服务，开启所有控制台	仅允许本机用户登录	支持
3	完整多用户模式。允许所有用户登录，拥有完整的功能，但是以文字模式进入系统	开启可登录的虚拟控制台，启用本地和网络用户；开启网络连接；启动所需网络服务	本机与网络用户	支持
4	保留。用户可自定义环境	如果未定义，进入该级别，将保持系统原有状态	本机与网络用户	支持
5	X11图形模式。与Runlevel 3功能一样，拥有完整功能，以图形界面模式进入系统	执行与Runlevel 3相同任务；启动X Windows System	本机与网络用户	支持
6	重启。不要将默认运行级别设置为此级别	执行与Runlevel 1相同任务；关闭系统之后通知BIOS重启系统	无	

Linux 运行级别

- 查看当前运行级别

  `runlevel`

- 切换到不同的运行级别
  - 启动时: 通过 grub 将运行级别作为参数传给内核
  - 启动后:init 或 telinit
- 设置默认运行级别
  - /etc/inittab 文件中的参数 id 设置
  - 若未设置默认运行级别, 启动过程中将提示用户输入一个运行级别

init 进程的工作过程

- init 启动后，根据/etc/inittab 文件的设置一次执行下列程序：
  1. /etc/rc.d/rc.sysinit：初始化 Linux 系统的环境

  主要工作包括启动 udev 与 SELinux 系统，设置内核参数、系统时间、交换空间、主机名、检查并挂载必要但文件系统，初始化硬件设备，启用软磁盘阵列与 LVM，卸载 initrd，重设磁盘参数等。
  1. /etc/rc.d/rc：建立并初始化运行级别环境

  不同的运行级别可提供不同的服务，通过/etc/rc.d/rc 来启动或停止特定运行级别中的服务。
  1. /etc/rc.d/rc.local：完成定制的初始化计划

  可在开机过程中执行某些初始化工作，执行该文件的前提是要进入的运行级别启用 local 服务，因为 local 服务的功能就是执行/etc/rc.d/rc.local
- 管理员可以定制/etc/inittab 来建立所需的系统运行环境

/etc/inittab 文件

- /etc/inittab 文件格式

```
cat /etc/inittab
man 5 inittab
id:runlevels:action:process
id          # 不超过 4 个字符的标识符
runlevels   # 运行级别列表，决定当前行对哪些运行级别起作用
action      # 进程执行方式
  sysinit   # 只要系统引导就开始运行
  respawn   # 表示进程结束后重新启动该进程
  wait      # 表示进程运行一次，init 需要等待其运行结束
  initdefault # 定义默认运行级别
process     # 具体运行的进程
```

系统启动过程故障排除顺序

1. 确定引导加载程序 GRUB 是否有问题
2. 检查是否正确载入 kernel 内核
3. 检查根目录是否挂载成功，如果不成功，应检查/sbin/init、/etc/inittab 以及/boot/grub/grub.con 配置文件的设置是否有错误，另外还要检查根目录是否损坏
4. 如果/etc/rc.d/rc.sysinit 执行不成功，则有可能是/bin/bash 文件损毁或是/etc/fstab 配置有问题
5. 检查/etc/rc.d/rc 以及/etc/rc.d/rc?.d(? 表示运行级别) 是否有问题

利用单用户模式修复系统 (1)

- 单用户模式类型
  1. runlevel 1

  执行 init 程序之后接着执行/etc/rc.sysinit 程序以初始化系统，然后再执行/etc/rc1.d/目录下的所有程序
  1. runlevel S

  执行 init 程序之后仅执行/etc/rc.sysinit 程序以初始化系统
  1. runlevel emergency

  执行 init 程序之后仅执行/etc/rc.sysinit 程序中某些必要的程序，绕过 rc.sysinit.sulogin，并不全部执行。

利用单用户模式修复系统 (2)

- 进入单用户模式
  - 在 GRUB 引导菜单按 a 或 e 键, 根据需要在内核参数项上加上“1”或“single”, 以进入单用户模式
  - 在/etc/inittab 配置文件将默认运行级别设置为 1 可在下次启动时进入单用户模式
  - 一个正在运行的 Linux 系统可以通过执行命令 init 1 切换至单用户模式
  - 启动至出现欢迎界面时，按下 I 键，系统会逐一询问是否启动每项服务，这时就处于 runlevel S 模式
  - 启动过程中如果挂载程序出现错误，会提示输入 root 用户密码，输入密码后，系统将直接进入 runlevel emergency 模式，完成修改后，按 Ctrl-d 将重启系统。

利用单用户模式修复系统 (3)

- 单用户模式修复的问题
  - Runlevel 1 执行至/etc/rc1.d/目录下的所有程序就结束，可以用来解决运行更高级别 (2~5) 时所发生的错误
  - Runlevel S 仅执行到 rc.sysinit 为止，除了可以解决 Runlevel 1 可以解决的问题之外，还可用来修复 Runlevel 1 发生的错误
  - RunLevel emergency 除了 Runlevel S 可以解决的错误之外，还可以解决 rc.sysinit 发生的错误

注意

进入 runlevel emergency 时根文件系统处于只读状态，无法直接修改 linux 系统文件。假设/etc/fstab 文件出了问题，启动时系统将进入 emergency 模式，这时需执行 mount -o remount,rw /重新挂载根文件系统，以便对/etc/fstab 进行修改。

使用 Linux 救援模式

- 救援模式 (rescue mode)
  - 当根目录所在的文件系统或 grub 损坏时无法启动内核或执行 init 进程，此时不能用单用户模式，而只能通过救援模式来修复 linux 系统故障。救援模式提供了从系统硬盘以外的设备 (光盘、U 盘等) 引导一个小型 linux 环境的能力，引导成功后再对硬盘上的错误进行修改和恢复。
- 进入救援模式的几种方式
  1. 从 CentOS 安装光盘的第 1 张盘引导系统
  2. 从 boot.iso 映像制作的引导光盘引导系统
  3. 从 bookdisk.img 映像制作的安装引导盘引导系统

通过安装光盘进入 Linux 救援模式

1. 从光盘引导系统，出现 boot: 提示符后，输入 linux rescue 命令
2. 提示救援环境是图寻找硬盘中的 linux 系统，并将其挂载到/mnt/sysimage 目录，需要选择如何处理（需要修改硬盘文件时选择 Continue；仅需读取硬盘文件而不修改时选择 Read-Only；如果要手动挂载文件系统则选择 Skip 直接跳过寻找并挂载硬盘的步骤。）
3. 按回车键后，系统提供一个 shell 给管理员使用
4. 进入救援模式后，正在运行的系统来自光盘，根分区也是光盘的/，硬盘分区全部被挂载到/mnt/sysimage 目录，有些管理工具必须在硬盘环境中执行，这时需要利用 chroot /mnt/sysimage 命令将程序运行时的根目录改为硬盘根目录，完成修复后，执行 exit 退出 chroot 环境，因为在 chroot 环境中读不到光盘文件。

实例：修复损坏的主引导记录

- 模拟损坏的主引导记录

```
dd if=/dev/zero of=/dev/sda bs=446 count=1
```

- 修复步骤
  1. 用光盘引导系统，输入 linux rescue 进入救援模式
  2. 改变根目录环境：chroot /mnt/sysinit
  3. 修复主引导记录：grub-install /dev/sda
  4. 执行 exit 退出 chroot 环境，再执行 exit 退出救援模式
  5. 重新从硬盘引导系统

内核组件

- 内核镜像文件
  - 内核通常以镜像文件 (Image File) 形式存储在 Linux 系统中
- 内核模块
  - Linux 内核的功能可以编译到内核镜像文件中 (静态模块), 或者单独成为内核模块, 以在系统运行期间动态地加载或者卸除模块 (动态模块)
- initrd 镜像文件
  - Linux 系统将部分模块制作成初始化内存磁盘 (initrd) 镜像文件, initrd 文件是在系统引导过程中挂载的一个临时根文件系统, 包含了引导过程中需要的模块和程序, 可用来挂载实际的根文件系统。

与内核有关的 rpm 软件包

- kernel: 包括内核、内核模块和必要文件
- kerenl-PAE: 适用于物理内存超过 4GB 的内核软件包
- kernel-xen: 适用于 Xen 虚拟化系统的内核软件包
- kernel-doc:Linux 内核文档 (内核源码的说明文件)
- kernel-utils: 提供 Linux 内核的管理工具, 以及与内核有关的服务
- kernel-TYPE-devel: 编译不同类型的内核 (内核镜像文件和模块) 所需的文件,TYPE 表示内核类型, 如 PAE

内核模块

- 可以在编译内核时选择将某些功能编译成为模块
- 内核模块可使用户不用重新编译内核就可动态地启用或者停用某一项功能
- 内核模块位于目录/lib/modules/‘uname -r‘/kernel
  - arch：有关硬件平台
  - crypto：加密算法
  - drivers：硬件设备驱动程序
  - fs：有关文件系统
  - lib：各种模块所需用到的链接库
  - net：有关网络
  - sound：声卡驱动
- 模块文件的扩展名为.ko，文件名部分就是模块名称

管理内核模块

- 查看已加载的内核模块

```
lsmod | grep md4
```

- 查看内核模块信息

```
modinfo [-0][-F 字段名] 模块名...
modinfo ext3
modinfo -F depends ext3
```

- 手动加载模块

```
insmod [模块文件名] [参数=值...]
insmod /lib/modules/`uname -r`/kernel/crypto/md4.ko
```

- 手动卸载模块

```
rmmod [-f] [-w] [-s] [-v] [模块名]
rmmod md4
```

处理模块间的依赖关系

- 使用 modprobe 命令可以自动加载或卸载所有必须用到的模块，modprobe 从/lib/modules/‘uname -r‘/modules.dep 文件读取模块的依赖关系。

```
modprobe [-C 配置文件] [模块名] [参数=值...]
modprobe md4 # 默认配置文件为/etc/modprobe.conf
```

- 使用 modprobe 卸载模块只需使用-r 选项

```
modprobe -r [模块名...]
```

- 使用选项-l 显示符合条件的模块文件路径

```
modprobe -l [-t dirname] [wildcard]
modprobe -l tcp*
```

内核模块配置文件

- /etc/modprobe.conf 用于配置各种内核模块的值，主要功能是设置模块的默认参数，指定加载或卸载模块时要执行的任务，以及设置模块别名。
  - alias：定义模块别名
  - options：设置模块默认参数
  - install：定义使用 insmod 或 modprobe 加载模块时执行的命令
  - remove：定义使用 delmod 或 modprobe 卸载模块时执行的命令
  - include：用于嵌入其他的内核模块配置文件

配置内核参数 (1)

- 编辑/proc 目录中的内核参数文件
  - /proc/sys 中的每一子目录存储重要的内核参数，这些目录中的每一个文件实际上是对应某一个内核参数，可称为内核参数文件，修改这些文件的内容就可以修改内核的功能

    ```
    echo "server1.abc.com" >/proc/sys/kernel/hostname
    ```

  - 修改内核参数属于临时修改，在关闭系统时就会丢失，可以通过修改/etc/rc.d/rc.local 文件指定要设置哪些内核参数

    ```
    echo 'echo "server1.abc.com" >\
    /proc/sys/kernel/hostname' >>/etc/rc.d/rc.local
    ```

配置内核参数 (2)

- 使用 sysctl 配置内核参数
  - sysctl 定义的内核参数名称

    /proc/sys/kernel/hostname 对应内核参数 kernel.hostname
  - 查看内核参数

```
sysctl -a                  # 查看所有内核参数
sysctl kernel.hostname     # 查看指定内核参数
```

  - 使用 sysctl 临时修改内核参数 (立即生效)

```
sysctl -w kernel.hostname=server1
```

  - 通过 sysctl 配置文件永久性配置内核参数

```
vim /etc/sysctl.conf       # 默认内核参数配置文件
kernel.hostname=server1    # 添加一行
sysctl -p                  # 使配置立即生效
```

rpm 软件包管理 (1)

- rpm 是由 Red Hat 公司提出的一种软件包管理标准，可用于软件的安装、查询、更新、升级、校验、卸载等。是目前应用比较广泛的软件包格式之一。
- 另一个应用比较广泛的软件包格式是来自 Debian Linux 提出的 deb 软件包管理标准，其功能与 rpm 类似。
- rpm 软件包名称规范

  `name(名字)-version(版本)-release(发行).arch(架构).rpm`
  例：`wget-1.11.4-3.el5_8.2.i386.rpm`

rpm 软件包管理 (2)

- 安装

```
mount /dev/cdrom /media/cdrom
cd /media/cdrom/CentOS
rpm -ivh zsh-4.2.6-9.el5.i386.rpm
rpm -ivh --replacepkgs zsh-4.2.6-9.el5.i386.rpm # 重新安
```

- 升级

```
rpm -Uvh zsh-4.2.6-9.el5.i386.rpm
# 会自动删除旧版软件包
# 即使以前未安装过该软件包也会完成软件包的安装
# 会自动把旧版本的配置文件备份
rpm -Uvh --oldpackage zsh-older.rpm # 降级安装
```

- 卸载

```
rpm -e zsh-4.2.6-9.el5.i386.rpm
```

rpm 软件包管理 (3)

- 查询

```
rpm -qa          # 打印所有已安装软件包列表
rpm -q zsh       # 查询软件包是否安装
rpm -qa | grep zsh   # 同上
rpm -qi zsh      # 查看软件包信息
rpm -ql zsh      # 查看软件包文件列表
rpm -qd zsh      # 查看软件包文档列表
rpm -qc zsh      # 查看软件包配置文件列表
rpm -qs zsh      # 查看软件包文件状态
rpm -qf /etc/inittab # 查询文件所属软件包
# 上述带-q 选项的命令加上-p 选项可查询待安装 rpm 包信息
rpm -qip /media/cdrom/CentOS/w3m-0.5.1-18.el5.i386.rpm
```

rpm 软件包管理 (4)

- 校验

  #rpm 校验通常用于两种情况：
  #1. 软件包本来运行良好，现在却出问题了，需要找出问题所在
  #2. 安全性考虑，检查是否有关键文件被修改了

  ```
  rpm -V pam
  ```

- rpm 校验标记

标记	相关属性
S	大小
M	模式 (权限)
5	md5 校验和
D	设备号不匹配
L	符号链接状态
U	所有者
G	所属组
T	修改时间

rpm 软件包管理 (5)

- 软件包签名验证
  - CentOS 公钥可以从以下任何一个地方获得：
    - http://mirror.centos.org/centos/
    - 安装光盘根目录
    - /etc/pki/rpm-gpg/目录
  - 导入 CentOS 公钥

    ```
    rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-*
    ```
  - 查看已导入的公钥列表

    ```
    rpm -qa | grep gpg-pubkey
    ```
  - 手动检查软件包签名

    ```
    rpm --checksig evince-0.6.0-17.el5.i386.rpm
    ```

rpm 软件包管理 (6)

- 高级查询

```
# 查找系统已安装的 10 个最大软件包
rpm -qa --queryformat "%10{size} %{name}\n" \
| sort -rn | head
rpm -q --requires samba # 查看 samba 软件包的必要条件
rpm -q --provides samba # 查看 samba 软件包提供的内容
rpm -q --scripts samba  # 查看 samba 软件包相关脚本
                        # 相关脚本分为 4 类：
                        # 安装前/安装后/卸载前/卸载后
rpm -qa --last | head   # 查看最新安装的 rpm 软件包
```

rpm 软件包管理 (7)

- **从 rpm 包中提取文件**

```
rpm2cpio 包全名 | cpio -idv . 文件绝对路径
#rpm2cpio 将 rpm 包转换为 cpio 格式
#cpio 用于创建软件档案文件和从档案文件中提取文件
-i #copy-in 模式，还原/提取
-d # 还原时自动新建目录
-v # 显示还原过程
```

示例

```
rpm -qf /bin/cat          # 查询 cat 属于哪个软件包
mv /bin/cat /tmp          # 假装误删除 cat 命令
cd /mnt/cdrom/CentOS
rpm2cpio coreutils-5.97-34.el5_8.1.i386.rpm \|
cpio -idv ./bin/cat       # 提取 cat 命令至当前目录
cp ./bin/cat /bin/cat     # 恢复 cat 命令至系统目录
```

rpm 软件包管理 (8)

- rpm 包的依赖性
  - 树形依赖：a –> b –> c(可按 c –> b –> a 顺序安装解决)
  - 环形依赖：a –> b –> c –> a(可同时安装 a、b、c 解决)
  - 模块依赖：www.rpmfind.net(可查询模块所属软件包)

```
ldd /bin/cat # 查看 cat 命令所需的动态链接库
```

rpm 包依赖性示例

```
cd /mnt/cdrom/CentOS
rpm -ivh wireshark-gnome-1.0.15-6.el5_10.i386.rpm
```

yum 包管理 (1)

- yum(Yellowdog Update, Modified) 是由 Duke 大学开发的包管理器，它能自动解决安装 rpm 软件包时遇到的依赖问题。
  - 软件仓库：一个预先准备好的目录或网站，包含了软件包和索引文件。yum 可以在仓库中自动定位并获取 rpm 软件包，只需一个命令就可以更新系统中的所有软件，也可以根据指定搜索条件来查找安装新的软件。
  - 配置 yum 仓库：在使用 yum 安装软件之前，应先配置 yum 指定使用的仓库。定义 yum 仓库的配置文件存放在/etc/yum.repos.d 目录下，文件名格式为 *.repo。
    - 默认/etc/yum.repos.d 目录内已经定义好了几个软件仓库。

yum 包管理 (2)

- 配置本地 yum 源的步骤
  - 建立 yum 软件仓库
    1. 安装 createrepo 软件包
    2. 将所有 rpm 软件包复制到一个目录 (如/mnt/myrepo)
    3. 执行 createrepo /mnt/myrepo 创建软件仓库
  - 配置 yum 软件仓库
    - 在/etc/yum.repos.d 目录下建立 myrepo.repo 配置文件

```
[myrepo]
name=my local repo
baseurl=file:///mnt/myrepo
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
```

  - 清理 yum 缓存并导入签名

```
yum clean all
rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
```

yum 包管理 (3)

- yum 软件包管理

```
yum list                  # 列出所有已安装和可用的软件包
yum list bash             # 列出 bash 软件包
yum list available        # 列出仓库中所有可用的软件包
yum list updates          # 列出仓库中比已安装包更新的软件包
yum list installed        # 列出已安装的软件包
yum list recent           # 列出新加入仓库的软件包
yum search httpd          # 搜索 httpd 相关软件包
yum -y install pkgs       # 安装软件包
yum -y update [pkgs]      # 升级软件包，未给包名则升级整个系统
yum remove pkgs           # 删除软件包及所有依赖于该包的软件包
yum info pkgs             # 查询软件包信息
yum grouplist             # 查询软件包组
yum groupinstall "grpname" # 安装软件包组
yum groupremove "grpname"  # 卸载软件包组
```

yum 包管理 (4)

- 使用本地光盘 yum 源

```
mount /dev/cdrom /mnt/cdrom
cd /etc/yum.repos.d/
mkdir bak; mv *.repo bak; mv bak/CentOS-Media.repo .
vim CentOS-Media.repo
[c5-media]
name=CentOS-$releasever - Media
baseurl=file:///mnt/cdrom
#         file:///media/cdrom/
#         file:///media/cdrecorder/
gpgcheck=1
enabled=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-5
```

注意：注释符 # 必须位于行首！

源码包管理 (1)

- 源码包
  - 源码包大多以 tar.gz 和 tar.bz2 文件格式打包，里面包含源代码及相关文件
  - 源码包需要编译安葬，要求系统已安装开发工具和开发库
    - 编译软件时可根据提示安装所需的开发工具和开发库, 有时还要用源码包编译安装所依赖的包
  - 通过 rpm 安装的包的相关信息在/var/lib/rpm 中，可以通过 rpm 命令进行同一管理，而源码包则无法通过 rpm 命令进行管理
  - 通过 rpm 安装的包都是安装在默认位置，但源码包则可以安装在指定位置

源码包管理 (2)

- rpm 包的默认安装位置

/etc	配置文件安装目录
/usr/bin	可执行文件安装目录
/usr/lib	库文件安装目录
/usr/share/doc	软件文档安装目录
/usr/share/man	帮助手册安装目录

- 源码包一般安装位置
  *usr/local/*软件名

源码包管理 (3)

- rpm 包安装的服务可以使用系统服务管理命令 (service) 进行管理，例如 rpm 包安装的 apache 的启动方法是：

```
/etc/rc.d/init.d/httpd start
service httpd start
```

  - 类似的命令有 checkconfig、ntsysv 等

- 源码包安装的服务不能被上述服务管理命令管理，因为没有安装到默认路径中，所以只能用绝对路径进行服务的管理

```
/usr/local/apache2/bin/apachectl start
```

源码包管理 (4)

- 源码包安装过程
  1. 下载源码包
  2. 解压缩源码包
  3. 进入解压缩目录
  4. 查看 INSTALL 和 README 文件
  5. ./configure 软件配置与检查 (编译前准备)
     - 定义需要的功能选项 (./configure –help 查看可用选项)
     - 监测系统环境是否符合安装要求
     - 生成编译阶段所需的 Makefile 文件
  6. 编译 (make)
     - 如果编译出错，在下一次重新编译前应该使用命令 make clean 清空上次编译生成的文件
  7. 编译安装 (make install)：需要 root 权限

源码包管理 (5)

- 源码包安装示例

```
tar xjvf httpd-2.2.31.tar.bz2
cd httpd-2.2.31
vim INSTALL
./configure --prefix=/usr/local/apache2
make
make install
/usr/local/apache2/bin/apachectl start
/usr/local/apache2/bin/apachectl stop
```

- 源码包的卸载
  - 不需要卸载命令，直接删除安装目录即可，不会遗留任何垃圾文件。

```
rm -rf /usr/local/apache2
```

设备文件与设备识别号

- Linux 内核并不关心/dev 目录下的设备文件名，而是设备识别号
- 每一个设备拥有主、次两个识别号
  - 主识别号帮助操作系统查找设备驱动程序代码，区分设备种类
  - 次识别号用于区分同一类设备的不同个体，从 0 开始编号
- 设备识别号无法修改，除非修改 Linux 内核源码，并且重新编译内核

创建设备文件

- mknod
  mknod [选项] 设备文件名类型主识别号次识别号

```
mknod /dev/md1 b 9 1
```

- MAKEDEV
  根据/etc/makedev.d 目录中的配置文件创建标准设备文件

```
cd /etc/makedev.d
grep 'sdb$' *
ls /dev/sd*
MAKEDEV -x sdb #-x 仅创建 sdb, 而非以 sdb 开头的所有设备
ls /dev/sd*
```

通过 udev 自动创建设备文件

- 内核 2.6 版新增 udev 子系统，用于动态创建或删除设备文件，减少/dev 目录的文件数
- udev 自动创建设备文件的过程
  1. 内核发现安装/卸载某设备时，执行 hotplug 安装/卸载该硬件驱动程序
  2. 请求执行 udev 创建/卸载该硬件的设备文件
  3. udev 通过 libsysfs 读取 sys 文件系统，获取该硬件设备信息
  4. udev 向 namedev 查询设备的设备文件信息，如名称、权限等
  5. udev 依据上述结果自动建立该设备的设备文件
- 如果需要修改系统预先提供的配置，可以修改/etc/udev 目录中的 udev 规则配置文件
  - udev 主配置文件

    /etc/udev/udev.conf 一般不用修改
  - udev 规则文件：.rules 文件

    以两位数字开头，表示系统应用该规则的顺序。规则采用键-值对形式，一条规则由多个键值对组成，由逗号隔开。

监控硬件设备

- 内核事件信息
  - 不管是 Linux 驱动硬件设备，还是硬件设备的状态发生变化，内核都会将这些信息存储为内核事件信息。
  - dmesg 命令：可以查看存储于 dmesg 缓冲区的内容，但 dmesg 缓冲区仅能保存 16K 信息，一旦缓冲区满，新信息会覆盖最老的信息。
  - /var/log/dmesg 文件：每次引导时会被重写，保存了最近一次启动后完整的内核事件信息
- 查看/proc 相关文件

```
cat /proc/cpuinfo     # 查看 cpu 信息
cat /proc/meminfo     # 查看内存信息
cat /proc/devices     # 查看硬件设备列表
cat /proc/diskstats   # 查看磁盘状态信息
/proc/ide/            #ide 设备信息
/proc/scsi/           #scsi 设备信息
```

管理 PCI 设备和 USB 设备

- 管理 PCI 设备

```
lspci   # 查看系统安装的 PCI 设备信息
setpci [选项] 设备 操作 # 配置 PCI 设备
```

- 查看 USB 设备

```
lsusb                          # 查看 USB 设备信息
cat /proc/bus/usb/devices # 查看 USB 设备详细信息
```

性能监测概述

- 性能监测的目的：找出系统的性能瓶颈以便调整优化
- 系统性能指标
  - 响应时间：从用户发出请求到用户得到返回结果所需时间
  - 吞吐量：在给定时间段内系统完成的交易数量
- 需监测的系统资源
  - CPU
  - 内存
  - 磁盘
  - 网络
- 图形界面下的“系统监视器”，可以直观实时地查看进程、CPU、内存、网络和文件系统等方面的基本信息。但要深入分析系统性能，还需借助其他工具。

CPU 性能监测

- 安装 sysstat 软件包
- sar 命令显示 CPU 总的性能情况
  - sar 是后台进程 sadc 的前端显示工具，安装 sysstat 包后，sadc 就会自动每隔 10 分钟收集一次系统状态并将它们存储到/var/log/sa 目录中

```
sar [选项] [采样间隔] [采样次数]
sar    # 查看当天的 CPU 统计信息
sar -u 3 5 # 每 3 秒采样 1 次，连续采样 5 次
```

- mpstat 命令可查看多个 CPU 的情况

```
mpstat [-P CPU 编号 |ALL] [采样间隔] [采样次数]
mpstat
mpstat -P 0 3 5 # 观测 CPU0，每 3 秒采样 1 次，连续采样
```

内存和磁盘 I/O 性能监测

- 内存性能监测

```
free      # 查看内存使用情况
sar -r    # 查看内存统计信息
vmstat    # 查看虚拟内存统计报告
```

- 磁盘 I/O 性能监测

```
sar -b    # 查看磁盘 I/O 统计信息
iostat [选项] [采样间隔] [采样次数]
iostat
```

通过 top 实现综合监控

- top

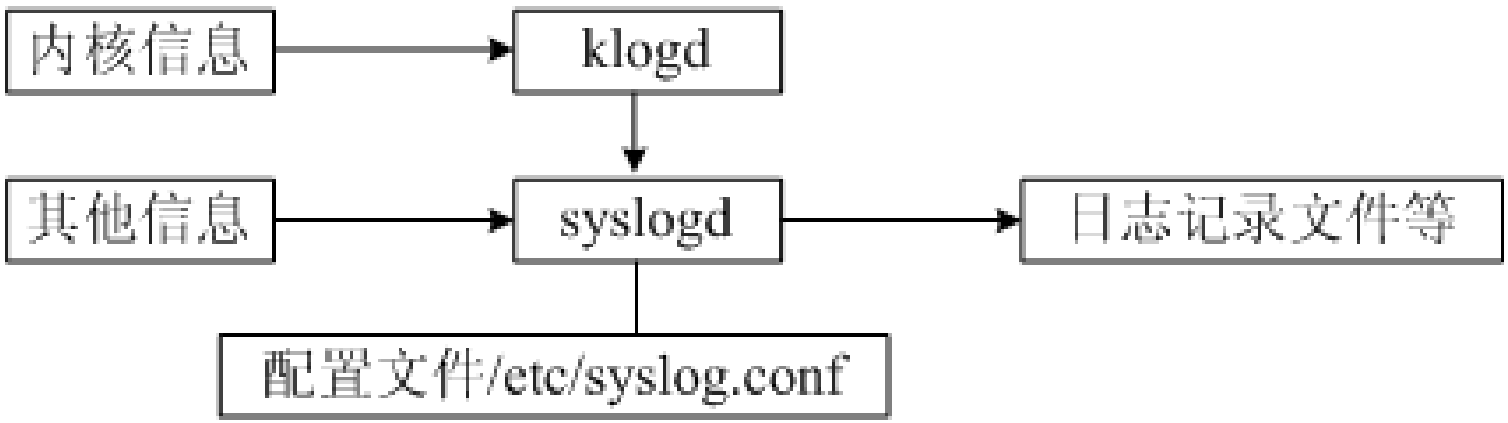
```
top
us # 用户进程占用 CPU 百分比
sy # 系统内核占用 CPU 百分比
ni # 更改过优先级的进程占 CPU 百分比
id #CPU 空闲时间百分比
wa #CPU 等待 I/O 操作时间百分比
hi #CPU 用于处理硬件中断所占时间百分比
si #CPU 用于处理软件中断所占时间百分比
st # 虚拟设备所占 CPU 时间百分比
```

```
PID(进程id) USER(用户) PR(优先级) NI(nice值)
VIRT(占用虚拟内存大小) RES(占用物理内存大小)
SHR(共享内存) S(进程状态) %CPU(占用CPU百分比)
%MEM(占用物理内存百分比) TIME+(使用CPU的时间)
COMMAND(进程命令名)
```

优化系统性能

1. 使用监测工具监视系统的活动
2. 分析得到的性能数据，找出系统的性能瓶颈
   - top 默认按使用 CPU 百分比降序排列
   - 按 M 键可按内存占用百分比降序排列
   - 按 F 键可选择按任意字段排序或显示更多字段
3. 分析造成性能降低的原因，采取相应的优化措施
   - 如果 id 值很低，说明 CPU 使用率很高，可以找出最占用 CPU 的进程
   - 如果 wa 值很高，说明系统处于高 I/O 状态，可用 iostat 进一步分析
   - 如果内存不足，可以找出最占用内存的进程

syslog 简介



常见日志文件

文件	记录内容
/var/log/cron	系统定时任务相关日志
/var/log/cups	打印日志
/var/log/dmesg	系统开机时内核自检信息，可用 dmesg 查看
/var/log/btmp	错误登录日志，用 lastb 查看
/var/log/lastlog	所有用户最后一次登录时间，用 lastlog 查看
/var/log/maillog	邮件日志
/var/log/message	系统绝大多数日志信息
/var/log/secure	验证和授权日志，记录涉及账户和密码的操作
/var/log/wtmp	所有用户登录、系统启停日志，用 last 查看
/var/log/utmp	记录当前登录用户信息，用 w、who 等命令查看

常见日志文件 (2)

- 除了系统默认的日志之外，采用 rpm 方式安装的系统服务也会默认把日志记录在/var/log/目录中（源码包安装的服务日志是在源码包指定目录中）。不过这些日志不是由 syslogd 服务来记录和管理的，而是各个服务使用自己的日志管理文档来记录自身日志。

文件	记录内容
/var/log/httpd	apache 服务的服务日志目录
/var/log/mail	邮件服务的服务日志目录
/var/log/samba	samba 服务的服务日志目录
/var/log/sssd	安全服务的服务日志目录

日志管理

- 日志格式
  1. 时间产生的时间
  2. 产生事件的服务器的主机名
  3. 产生事件的服务名或程序名
  4. 事件的具体信息
- 清空日志文件内容

```
echo >/var/log/secure
```

- /etc/syslog.conf 配置文件

```
authpriv.* /var/log/secure
选择子        动作
选择子：服务名称[连接符号]日志等级
动作：日志记录位置或日志处理动作
认证相关服务.所有日志等级 记录在/var/log/secure中
```

服务名称

服务名称	含义
auth	安全和认证相关消息 (不推荐使用 authpriv 替代)
authpriv	安全和认证相关消息（私有的）
cron	系统定时任务 cron 和 at 产生的日志
daemon	和各个守护进程相关的日志
ftp	ftp 守护进程产生的日志
kern	内核而非用户进程产生的日志
local10-local7	为本地使用预留的服务
lpr	打印产生的日志
mail	邮件收发日志
news	与新闻服务器相关的日志
syslog	由 syslogd 产生的日志信息
user	用户等级类别的日志信息
uucp	uucp 子系统的日志信息

日志等级

等级名称	说明
none	不记录所有等级日志
*	记录所有等级日志
debug	一般的调试信息说明
info	基本的通知信息
notice	普通信息，但是有一定的重要性
warning	警告信息，但未影响到服务或系统的运行
err	错误信息，已经影响到服务或系统的运行
crit	临界状况信息，比 err 等级还要严重
alert	警告状态信息，比 crit 还要严重，必须立即采取行动
emerg	紧急等级信息，系统已经无法使用

连接符和日志处理方式

- 连接符

连接符	含义
.	记录大于等于指定等级的日志
.=	仅记录指定等级的日志
.!	记录除指定等级之外的日志

- 日志处理方式
  - 记录至指定日志文件：如/var/log/secure
  - 发送到系统设备文件：如/dev/lp0
  - 转发给远程主机：如 @192.168.0.210
  - 发送到指定用户所在终端：如 root,adm

日志管理和测试

- 修改日志配置文件后使配置立即生效

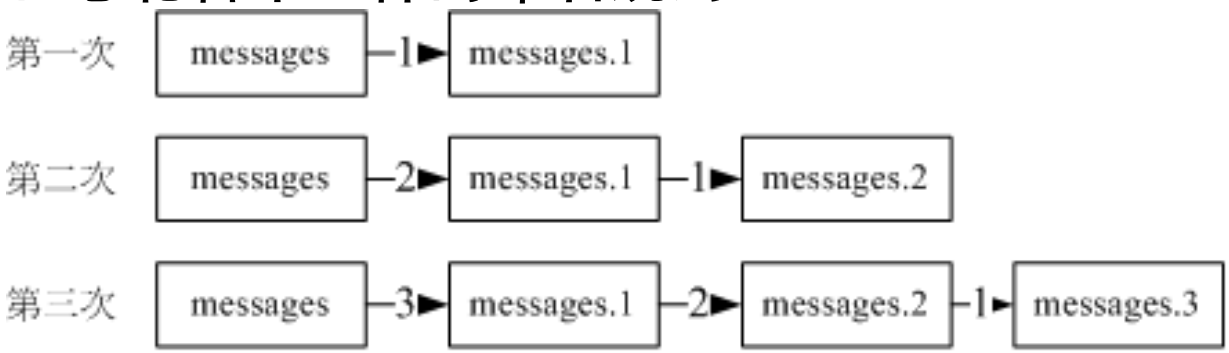
```
killall -HUP syslogd
```

- 使用 logger 工具进行测试

```
logger kern.info "kern info test" # 模拟 kern.info 消息
```

日志轮替

- 日志轮替：防止日志占用过多磁盘空间
  - 日志存储为多个文件，如每日一个
  - 限制日志文件个数，如只保留一周的日志
- 日志轮替中文件的命名规则



日志轮替配置文件/etc/logrotate.conf

参数	说明
daily	每天轮替
weekly	每周轮替
monthly	每月轮替
rotate n	保留 n 个日志文件
compress	轮替时压缩旧日志
create mode owner group	新建日志 (权限、所有者、属组)
mail address	轮替时发邮件至指定邮箱地址
missingok	忽略日志不存在的情况
notifempty	不轮替空日志
minsize 10k	按时间轮替，日志小于 10k 时不轮替
size 1M	日志按大小轮替，达到 1M 时轮替
dateext	用日期作为日志后缀，如 secure-20160606

日志轮替管理

- 用 rpm 包安装的软件包默认使用了日志轮替
- 用源码安装的软件包需要手工配置日志轮替

```
vim /etc/logrotate.conf
/usr/local/apache2/logs/access_log {
    daily
    create
    rotate 30
}
```

- logrotate [选项] 配置文件名
  - 如果没有指定选项，则按照配置文件的配置进行日志轮替
  - -v 显示日志轮替详细信息
  - -f 强制进行日志轮替

集中式日志管理

- 在作为日志客户端的计算机上设置适当的信息传送到日志服务器。此时，需在“处理方式”字段中使用 @ 指定日志服务器的计算机名称或 IP 地址

```
*.* @logserver
```

- 修改日志服务器的/etc/sysconfig/syslog 配置文件

```
SYSLOGD_OPTIONS="-m 0 -r"
```

- 重新启动日志服务器与客户端的系统日志服务

```
service syslog restart
```