

第一章 CentOS 6.4 系统的安装

一、实训目的

熟悉 CentOS 系统的安装步骤；熟悉 Linux 操作系统的分区；熟悉系统引导程序的安全与相关参数。

二、安装步骤

启动时，安装程序显示一个欢迎界面和引导提示，如图 1-1 所示。



图 1-1 安装程序启动界面

输入 Esc 键后，引导提示为你提供第一个与 Anaconda 安装程序对话的机会。通常人们只需要按回车键，或等待提示超时，之后安装程序将继续按默认设置运行。如果在安装程序引导提示中加上如表 1-1 所示的引导参数，将修改 Anaconda 的默认行为。

表 1-1 引导 Anaconda 时常用的参数

参数	作用
text	强制安装程序使用文本模式进行安装
resolution=M×N	强制图形 X 服务器使用屏幕分辨率 M×N，其中 M、N 为数字，例如 1024×768
lowres	设置分辨率为 640×480
askmethod	询问进入安装程序第二阶段的方法（如果不使用 CD 盘安装）
ks=params	使用 Kickstart 执行脚本化安装。将在后面的章节中详细讲解 Kickstart 的安装

续表

参数	作用
mediacheck	在进行安装前对 CD 中的文件进行完整性检测
noprobe	不要自动寻找硬件
rescue	不执行安装，而是使用 Anaconda 运行救援 Shell 进行系统修复

当给出引导参数时，引导程序命令行的第一个标记一定是 `linux` 这个词。后面的标记可以从前面的列表中选择。另外，如果是安装程序无法识别的标记，就将其传给 Linux 内核作为内核引导参数。

在下面的例子中，用户选择了正确的引导参数来以文本模式启动安装程序，并在开始安装前进行介质检查。

```
boot: linux text mediacheck
```

图 1-1 中显示的选项为：

第一项为安装或升级已经存在的系统。

第二项为安装带有基本的显卡驱动的系统。

第三项为紧急救援模式启动系统。

第四项为从本地启动系统。

第五项为内存测试。

选择第一项后，安装程序显示欢迎界面，如图 1-2 所示。



图 1-2 安装程序欢迎界面

安装程序首先会要求用户选择合适的语言，如图 1-3 所示。在后面的安装过程中会一直使用选中的语言，并将其设为已安装系统的默认语言。

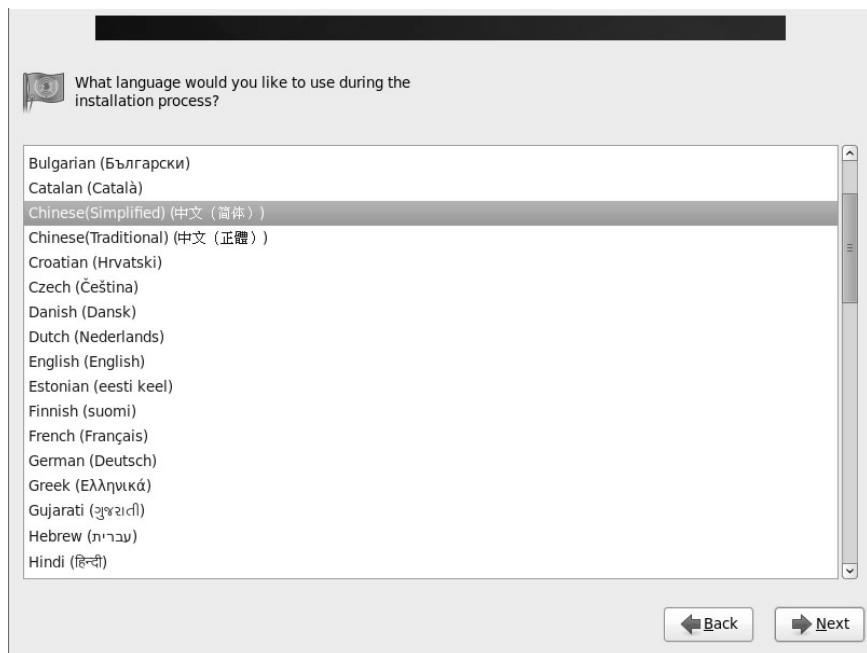


图 1-3 语言选择界面

接下来安装程序会询问键盘布局，如图 1-4 所示。选中的键盘布局将在后面的安装过程以及安装好的系统中保留。



图 1-4 键盘布局选择界面

之后是选择安装的存储类型，由于本书不讨论更复杂的非本地存储安装方式，因此只要选择默认的基本存储设备即可，如图 1-5 所示。



图 1-5 选择磁盘设备

单击“下一步”按钮进入如图 1-6 所示的界面，在界面中可以直接输入要安装系统的主机名，也可以选择使用默认值，在安装完毕后再通过后续配置或者其他方式（例如 DHCP）来自自动获取这个系统的主机名。单击“配置网络”按钮就可以进入机器的联网配置信息界面。CentOS 6.4 系统引入了和之前版本不同的 NetworkManager 工具界面来管理安装程序的网络配置。



图 1-6 主机名配置界面

单击“配置网络”按钮后，就可以在弹出的“网络连接”选项卡中选择需要配置的网卡，并且选择编辑指定的网卡，进入具体网络参数输入的界面，如图 1-7 所示。



图 1-7 网卡 IP 地址设置界面

单击“下一步”按钮，进入如图 1-8 所示的指定时区信息的界面。定义时区的条目中包括本地时间协议，比如夏令时。另外，Linux 允许 BIOS 时钟设定为本地时间，也可以设定为全球时间（UTC）。后者是更方便的方法，特别是对需要经常调整时区的手提电脑用户。当改变时区时，BIOS 根本不用调整，只要提供可用来解析时区的信息就可以了。



图 1-8 选择时区信息界面

安装程序接下来询问根用户（管理账户）的密码，如图 1-9 所示。

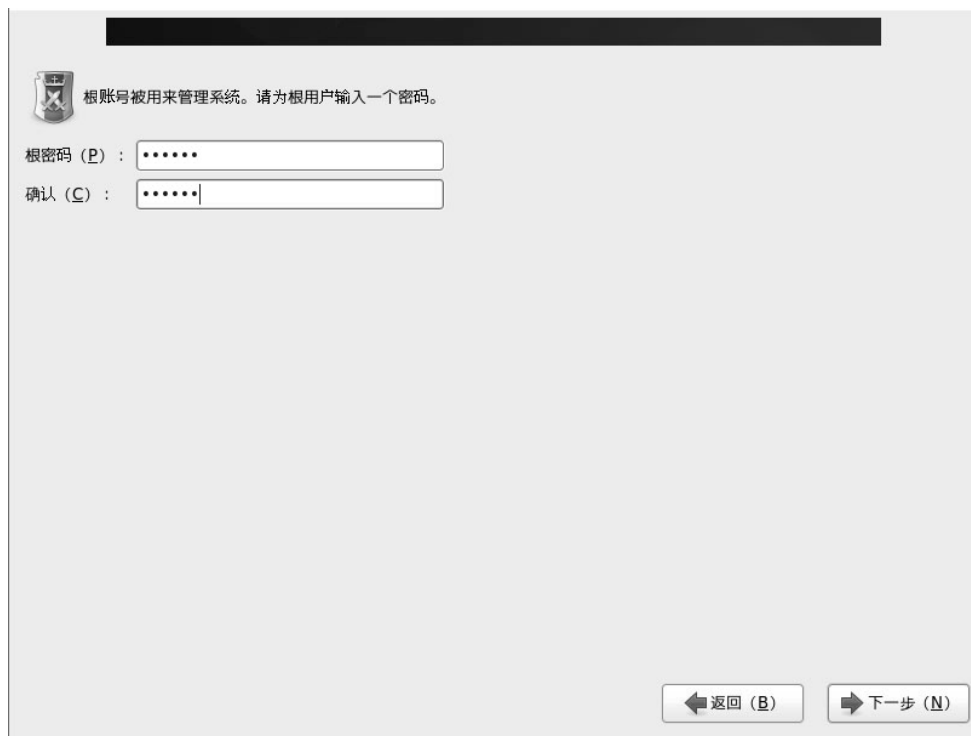


图 1-9 根密码设置界面

注意：选择一个合适的密码并不是一件容易的事。下面是几条比较好的密码选择策略。

- (1) 密码最小长度应在 8 位以上。
- (2) 密码应包含大写字母、小写字母和特殊符号（如 \$、#、%、& 等）。
- (3) 密码不包含个人信息。
- (4) 密码应容易记忆。

和其他操作系统不同，Linux 使用了 UNIX 将所有磁盘分区合并为一个统一的目录树结构的模式。它将不同分区的根目录绑定到一个统一目录树的特定目录中，而不是将磁盘分区指定为“C:”或者“D:”。将分区的根目录与其他目录绑定的动作就是挂载的过程，目标目录就是分区的挂载点。

当执行安装时，安装程序需要了解要创建多少个分区、分区应该有多大以及在哪里挂载分区。另外，由于 Linux 支持几种不同的文件系统来格式化分区，因此安装程序需要了解分区使用的文件系统是什么。

分区可能是安装 Linux 系统过程中最复杂的部分，并且一旦系统安装成功，它也是相对比较容易更改的部分。在有关分区的第一个界面中，安装程序询问采用哪种磁盘分区方案，如图 1-10 所示。

如果不选择左下角的“查看并修改分区布局”复选框，Anaconda 就会按照预设的方式部署在磁盘上部署分区以及逻辑卷。默认分区效果如图 1-11 所示。



图 1-10 Anaconda 分区选择界面



图 1-11 默认自动分区界面

安装程序在这一界面的顶部显示了所有探测到的磁盘列表及当前分区方案。在底部有一个表格列出了所有磁盘及其分区的详细资料。在当前界面中可创建新的分区，或者编辑、删除已存在的分区。如果要创建新分区，安装程序会打开一个二级对话框来收集有关新分区的信息，如图 1-12 所示。



图 1-12 添加新的分区界面

当创建新分区的时候要提供以下 3 项重要信息。

(1) 用这个分区来做什么？

为了正确初始化分区，安装程序必须了解分区的用途。“文件系统类型”的弹出菜单可提供下列选项。

1) 文件系统：大多数分区是用来保存文件，并可用文件系统进行格式化。安装程序可使用 ext2、ext3、ext4 或者 vfat 文件系统来格式化分区。在 red hat CentOS 6.4 中，使用 ext4 文件系统作为默认的文件系统。

2) swap（交换分区）：该分区用来存储当系统物理内存不足时，临时存放当前不需要的内存页。

3) RAID 或者逻辑卷管理（Logical Volume Management，简称 LVM）：分区将被用来构成更复杂的结构，这种结构可提供更快的速度（通过并行化）、更强的弹性（通过冗余）和更多的灵活性（通过间接作用）。

(2) 分区应该有多大？

为了要了解需要分配的空间大小，Anaconda 需要知道分区所需要的大小（以 MB 为单位）。用下面的一个选项可以限定分区的大小。

1) 固定大小：分区应该和指定的大小完全相符。

2) 有限分区：指定的分区大小应理解为分区大小的下限，并可指定分区大小的上限。得到的分区将至少为指定的大小，但是（根据其他分区操作）可使用未分配的磁盘空间，最多可达指定的最大值。

3) 非限定分区：指定的大小应理解为分区大小的下限，但得到的分区应该可使用特定驱动器上所有未分配的磁盘空间。

(3) 应该在哪里挂载分区？

假设已经用文件系统将分区进行了格式化，那么就一定要指定一个挂载点来将分区并入目录结构。

1) 删除分区：删除分区的过程很简单。选中列表中的分区，单击“删除”按钮将分区删除即可。

2) 添加分区：选中图 1-11 中硬盘驱动器列表中的硬盘分区或者卷组，然后单击“创建”按钮，出现图 1-12 中“生成存储”界面，使用默认选项，单击“创建”按钮，出现图 1-12 中“添加分区”界面，修改相应的参数后，单击“确定”按钮，这样就创建了一个分区。

3) 编辑已存在的分区：选中图 1-11 中硬盘驱动器列表中的硬盘分区或者卷组，然后单击“编辑”按钮，出现如图 1-13 所示的界面，根据要求修改相应的参数后，单击“确定”按钮完成分区的编辑。



图 1-13 编辑分区界面

选定分区方案后，Anaconda 会询问系统引导程序的配置信息，如图 1-14 所示。引导程序是一个小型、底层可执行文件，它可将控制权从 BIOS 转移给选择的操作系统。对于许多操作系统来说，引导程序是一个用户不了解的概念。因为 Linux 有与其他操作系统共存的先例，因此 Linux 引导程序通常很灵活，并容易配置。

有时候，当配置双引导（dual-boot）系统（在启动时在可能安装的多个操作系统中进行选择的引导系统）时，管理员希望保留一个已存在的引导程序。这是个例外，通常的规则是安装程序应该按默认设置将 GRUB 引导程序安装到磁盘主引导记录中（Master Boot Record，简称 MBR），使用 GRUB 作为默认的操作系统引导程序。

因为 GRUB 引导程序非常灵活，所以当用户使用它来引导系统进入维护模式时，可能会带来安全隐患。无论何时，在考虑安全问题时，应该用密码保护引导程序。

安装程序接下来会让用户决定安装什么样的软件。在 CentOS 6.4 中，软件包分组被切分得比之前更加细小，因此也可以把系统安装得更加精确。正常来说，作为一个企业级的服务器系统，用户在服务器上应该只安装自己需要的和必要的系统以及应用软件。这样做主要是出于安全方面的考虑。假设用户安装一台 Linux 服务器的目的是用于万维网服务器（http 服务器），那么就不应该安装并且打开 Samba 文件共享服务器或者 VNC 远程桌面连接服务。

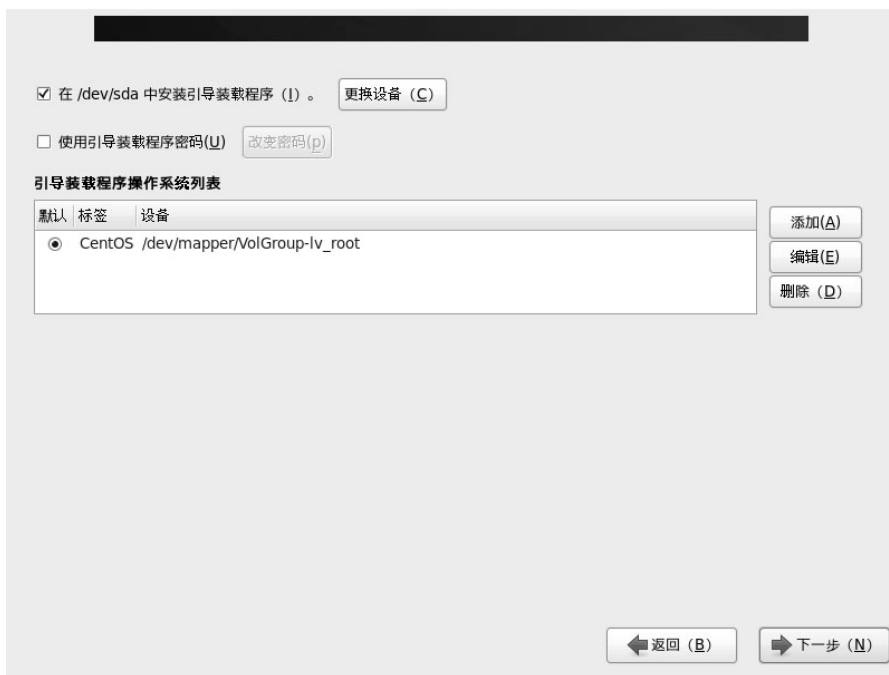


图 1-14 安装程序引导程序界面

因为这些不需要但是却运行着的服务器可能会由于它们本身的安全问题导致需要正常运行的 HTTP 服务受到影响,另外也增加了服务器维护的时间以及复杂度。软件选择的界面如图 1-15 所示。



图 1-15 软件选择界面

如果指定要自定义选择软件（选中“现在自定义”单选按钮），安装程序接下来会显示软件分类选择界面，如图 1-16 所示。软件组件将软件按大项目分类，例如桌面、应用程序等分类组。

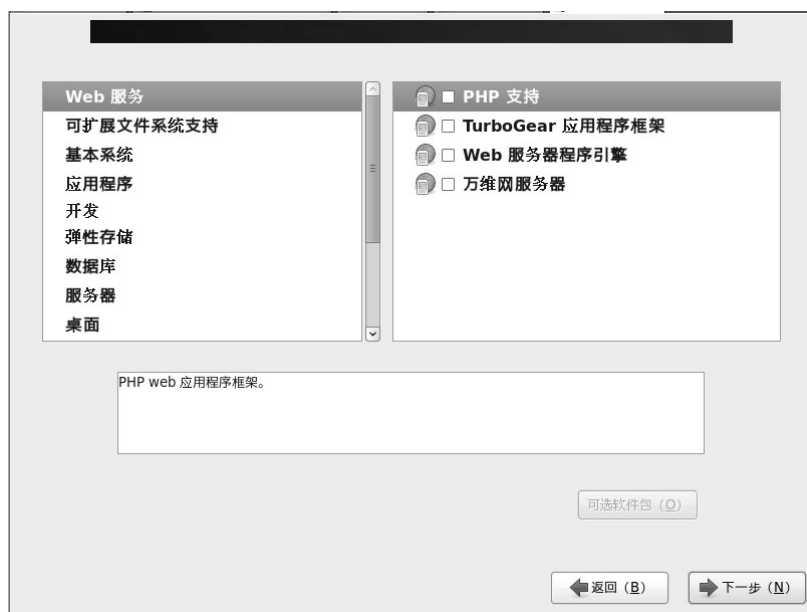


图 1-16 软件分类选择界面

软件分类组又进一步细分为小的软件组，包含多个可选软件组分类。每个软件组又分为必须安装的软件包和选择安装的软件包。如果选择浏览组件组的可选软件包，可以选定或者取消特定的软件包，如图 1-17 所示。



图 1-17 组件自定义界面

安装程序现在开始执行所有的指定配置，创建并格式化所需分区并启动。接下来安装程序会使用进度条来监视各种软件的安装进程，如图 1-18 所示。



图 1-18 安装程序进度界面

最终会出现如图 1-19 所示的完成安装界面，提示安装完毕。单击“重新引导”按钮，以重启计算机。



图 1-19 完成安装界面

至此 CentOS 6.4 系统安装结束。

系统引导程序 GRUB 配置文件的内容如下：

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#           all kernel and initrd paths are relative to /boot/, eg.
#           root (hd0,0)
#           kernel /vmlinuz-version ro root=/dev/mapper/vg_mail-lv_root
#           initrd /initrd-[generic-]version.img
#boot=/dev/sda
default=0 //默认启动的操作系统，注：下面的系统菜单数从 0 开始
timeout=5 //设置在 5s 后启动默认的系统
splashimage=(hd0,0)/grub/splash.xpm.gz //开机画面的文件所存放的路径和文件名，这里是指用在
                                         (hd0,0)/grub/下的 splash.xpm.gz 文件作为开机画面。(hd0,0)
                                         指的是第一个硬盘的第一个分区
Hiddenmenu //隐藏启动系统名字的菜单，把它注释掉后会在开机画面中显示机器中所有的系统名
title CentOS (2.6.32-358.el6.x86_64) //title 就是一个引导标签，可以对 title 后的文字部分进行修改，
                                         使它更符合我们的使用习惯
root (hd0,0) //Grub 的根文件系统即/boot 所在分区
kernel /vmlinuz-2.6.32-358.el6.x86_64 ro root=/dev/mapper/vg_mail-lv_root rd_NO_LUKS rd_LVM_
LV=vg_mail/lv_root rd_NO_MD rd_LVM_LV=vg_mail/lv_swap crashk_ernel=auto LANG=zh_
CN.UTF-8 KEYBOARDTYPE=pc KEYTABLE=us rd_NO_DM rhgb quiet
//kernel /vmlinuz-2.6.32-358.el6.x86_64 是 Linux 内核/boot/ vmlinuz-2.6.32-358.el6.x86_64。ro 和 root
是传递给内核的参数，root 后是 Linux 的根文件系统所在分区。内核参数 ro 是 read only 的意思。
rhgb 图形方式启动，quiet 不输出启动信息
initrd /initramfs-2.6.32-358.el6.x86_64.img //即/boot/ initrd-2.6.9-55.ELsmp.img 内存空间映像，用于
                                         初始化和启动设备
```

第二章 Yum 仓库的建立与命令的使用

一、实训目的

掌握建立 Yum 源；重建 Yum 仓库和常见 Yum 命令的使用。

二、Yum 简介

Yum 是 Yellow dog Updater Modified 的简称，起初是由 yellow dog 这一发行版的开发者 Terra Soft 研发，用 python 写成，那时叫做 yup (yellow dog updater)，后经杜克大学的 Linux@Duke 开发团队进行改进，遂改此名。Yum 的宗旨是自动化地升级、安装与移除 rpm 包，收集 rpm 包的相关信息，检查依赖性并自动提示用户解决。Yum 的关键之处是要有可靠的软件仓库，它可以是 HTTP、FTP 站点，也可以是本地软件池，但必须包含 rpm 的 header，header 包括了 rpm 包的各种信息，包括描述、功能、提供的文件、依赖性等。

三、配置文件

Yum 的配置文件在 /etc 目录下，主要包含 /etc/yum.conf、/etc/yum 目录下的所有文件、/etc/yum.repos.d 目录下的所有 Yum 安装源文件。

1. Yum 的配置文件

Yum 的一切配置信息都储存在一个叫 yum.conf 的配置文件中，通常位于 /etc 目录下，这是整个 Yum 系统的重中之重，内容如下：

```
[main]
cachedir=/var/cache/yum/$basearch/$releasever
keepcache=0
debuglevel=2
logfile=/var/log/yum.log
exactarch=1
obsoletes=1
gpgcheck=1
plugins=1
installonly_limit=5
bugtracker_url=http://bugs.centos.org/set_project.php?project_id=16&ref=http://bugs.centos.org/bug_report_page.php?category=yum
distroverpkg=centos-release
# PUT YOUR REPOS HERE OR IN separate files named file.repo
# in /etc/yum.repos.d
```

下面是对 yum.conf 文件作简要的说明：

cachedir: Yum 缓存的目录，Yum 在此存储下载的 rpm 包和数据库，一般是 /var/cache/yum。

debuglevel: 除错级别，0—10，默认是 2。

logfile: Yum 的日志文件，默认是 /var/log/yum.log。

exactarch: 有两个选项 1 和 0，代表是否只升级和用户安装软件包 CPU 体系一致的包，如果设为 1，则如用户安装了一个 i386 的 rpm，则 Yum 不会用 686 的包来升级。

obsoletes: 这是一个 update 的参数，具体请参阅 yum(8)，简单的说就是相当于 upgrade，允许更新陈旧的 rpm 包。

gpgcheck: 有 1 和 0 两个选择，分别代表是否进行 gpg 校验，如果没有这一项，默认是进行 gpg 校验。

Plugins: 是否启用插件，默认 1 为允许，0 表示不允许。

installonly_limit: 网络连接错误重试的次数。

bugtracker_url: 设置上传 bug 的地址。

distroverpkg: 指定一个软件包，Yum 会根据这个包判断用户的发行版本，默认是 redhat-release，也可以是安装的任何针对自己发行版的 rpm 包。

2. Yum 源文件

Yum 源文件是指定 Yum 仓库的位置。创建 Yum 源文件/etc/yum.repos.d/CentOS-Media.repo，/etc/yum.repos.d/目录下最好只有 CentOS-Media.repo 一个文件，否则如果网络有问题就会报告找不到 Yum 源的错误，内容如下：

```
# /etc/yum.repos.d/CentOS-Media.repo
# or for ONLY the media repo, do this:
# yum --disablerepo=* --enablerepo=c6-media [command]
```

```
[c6-media]
name=CentOS-$releasever - Media
baseurl=file:///media/CentOS/
        file:///media/cdrom/
        file:///media/cdrecorder/
gpgcheck=1
enabled=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-6
```

下面是对 yum.conf 文件作简要的说明：

[]: 是用于区别各个不同的 repository，必须有一个独一无二的名称。

name: 是对 repository 的描述，支持像 \$releasever \$basearch 这样的变量。

baseurl: 是服务器设置中最重要的部分，只有设置正确，才能获取软件。它的格式是：

```
baseurl=url://server1/path/to/repository/
        url://server2/path/to/repository/
        url://server3/path/to/repository/
```

其中 url 支持的协议有 http、ftp 和 file 三种。baseurl 后可以跟多个 url，用户可以自己改为速度比较快的镜像站，但 baseurl 只能有一个。

gpgcheck: 设置是否进行验证。

enabled: Yum 源是否生效。

gpgkey: 设置验证密钥文件的路径。

3. 重建 Yum 仓库

首先安装 createrepo 包，createrepo 是用来创建 Yum 仓库的命令，它的依赖文件有 deltarpm、

libxml2-python、python-deltarpm。命令如下：

```
#yum createrepo -y
```

接下来先将 rpm 包文件拷贝到 /var/ftp/centos6.4/Packages 目录下 (rpm 包文件都在当前目录下)，然后使用 createrepo 命令创建 yum 仓库所必须的一些信息，这些信息都存放在 /var/ftp/centos6.4/repodata/ 目录下。命令如下：

```
#bin/cp ./*.rpm /var/ftp/centos6.4/Packages
```

```
#createrepo -v /var/ftp/centos6.4
```

最后修改 Yum 源文件的 baseurl 的值，修改后的 Yum 源文件内容如下：

```
[c6-media]
name=CentOS-$releasever - Media
baseurl=file:///var/ftp/centos6.4
gpgcheck=0
enabled=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-6
```

注意：如果要将 gpgcheck 值改为 1，则要将所有的 rpm 包文件的 gpgkey 导入，用 rpm --import /path。

path 为保存 gpgkey 文件的完整路径。

四、Yum 命令的使用

1. 使用 Yum 更新软件包的命令

在查看系统中的软件是否有升级包时，会用到 yum check-update 命令；在软件出现漏洞时，对软件进行升级或者对系统内核进行升级就会用到 yum update 命令：

```
yum check-update    //列出所有可更新的软件清单
yum update          //安装所有更新软件
```

2. 使用 Yum 安装与删除软件

使用 Yum 安装和删除软件，有个前提是 Yum 安装的软件包都是 rpm 格式的。安装的命令是 yum install xxx，Yum 会查询数据库，如果有，则检查其依赖冲突关系；如果没有依赖冲突，那么下载安装；如果有，则会给出提示，询问是否要同时安装依赖，或删除冲突的包。

删除的命令是 yum remove xxx，同安装一样，Yum 也会查询数据库，给出解决依赖关系的提示。

```
yum install xxx      //用 Yum 安装软件包，xxx 为安装软件包的名称，不包括版本号和版本号
                    之后的信息
yum remove xxx       //用 Yum 删除软件包，xxx 为安装软件包的名称，不包括版本号和版本号
                    之后的信息
```

3. 使用 Yum 查询想安装的软件

常常会碰到这样的情况，想要安装一个软件，只知道它和某方面有关，但又不能确切知道它的名字。这时 Yum 的查询功能就起作用了。用户可以用 yum search keyword 这样的命令来进行搜索，比如要安装一个 Instant Messenger，但又不知到底有哪些，这时不妨用 yum search messenger 这样的指令进行搜索，Yum 会搜索所有可用 rpm 的描述，列出所有描述中和 messenger 有关的 rpm 包，于是用户可能得到 gaim、kopete 等，并从中选择。

```
yum search            //使用 Yum 查找软件包
```

```

yum list           //列出所有可安装的软件包
yum list updates   //列出所有可更新的软件包
yum list installed  //列出所有已安装的软件包
yum list extras     //列出所有已安装但不在 Yum Repository 内的软件包

```

4. 使用 Yum 获取软件包信息

常常会碰到安装了一个包，但又不知道其用途的情况，这时用户可以用 `yum info packagename` 这个指令来获取信息。

```

yum info           //列出所有软件包的信息
yum info updates   //列出所有可更新的软件包信息
yum info installed  //列出所有已安装的软件包信息
yum info extras     //列出所有已安装但不在 Yum Repository 内的软件包信息
yum provides       //列出软件包提供哪些文件

```

5. 清除 Yum 缓存

Yum 会把下载的软件包和 header 存储在 Cache 中，而不会自动删除。如果用户觉得它们占用了磁盘空间，可以使用 `yum clean` 指令进行清除，更精确的用法是 `yum clean headers` 清除 header，`yum clean packages` 清除下载的 rpm 包，`yum clean all` 清除所有。

```

yum clean packages //清除缓存目录(/var/cache/yum)下的软件包
yum clean headers   //清除缓存目录(/var/cache/yum)下的 headers
yum clean oldheaders //清除缓存目录(/var/cache/yum)下旧的 headers
yum clean all       //清除缓存目录(/var/cache/yum)下的软件包及旧的 headers

```

6. 示例：安装 httpd 软件

```

[root@mail repodata]# yum install httpd
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
* c6-media:
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package httpd.x86_64 0:2.2.15-26.el6.centos will be installed
--> Processing Dependency: httpd-tools = 2.2.15-26.el6.centos for package: httpd-2.2.15-26.el6.centos.x86_64
--> Processing Dependency: apr-util-ldap for package: httpd-2.2.15-26.el6.centos.x86_64
--> Processing Dependency: /etc/mime.types for package: httpd-2.2.15-26.el6.centos.x86_64
--> Processing Dependency: libaprutil-1.so.0()(64bit) for package: httpd-2.2.15-26.el6.centos.x86_64
--> Processing Dependency: libapr-1.so.0()(64bit) for package: httpd-2.2.15-26.el6.centos.x86_64
--> Running transaction check
---> Package apr.x86_64 0:1.3.9-5.el6_2 will be installed
---> Package apr-util.x86_64 0:1.3.9-3.el6_0.1 will be installed
---> Package apr-util-ldap.x86_64 0:1.3.9-3.el6_0.1 will be installed
---> Package httpd-tools.x86_64 0:2.2.15-26.el6.centos will be installed
---> Package mailcap.noarch 0:2.1.31-2.el6 will be installed
--> Finished Dependency Resolution
Dependencies Resolved

```

Package	Arch	Version	Repository	Size
---------	------	---------	------------	------

Installing:

httpd	x86_64	2.2.15-26.el6.centos	c6-media	821 k
-------	--------	----------------------	----------	-------

Installing for dependencies:

apr	x86_64	1.3.9-5.el6_2	c6-media	123 k
apr-util	x86_64	1.3.9-3.el6_0.1	c6-media	87 k
apr-util-ldap	x86_64	1.3.9-3.el6_0.1	c6-media	15 k
httpd-tools	x86_64	2.2.15-26.el6.centos	c6-media	72 k
mailcap	noarch	2.1.31-2.el6	c6-media	27 k

Transaction Summary

=====
Install 6 Package(s)

Total download size: 1.1 M

Installed size: 3.6 M

Is this ok [y/N]: y //回答 y 后, 就会进行软件安装