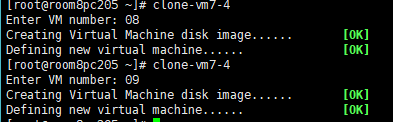
**基础环境搭建**

1. 创建两台虚拟机

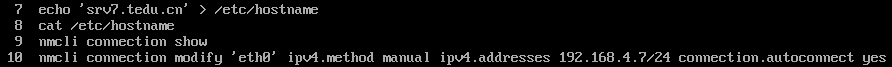




1. 将两台虚拟机，修改名字，一个为A，一个为B
2. 利用root登录，密码123456，登录两台虚拟系统
3. 设置虚拟机A

IP地址：192.168.4.7/24

主机名：svr7.tedu.cn



1. 设置虚拟机B

IP地址：192.168.4.207/24

主机名：pc207.tedu.cn



1. 进行远程进行管理
2. yum仓库的设置

服务端：真机搭建web服务，共享光盘所有内容

1. 安装httpd软件

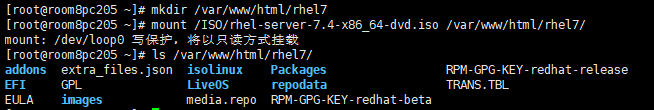
rpm -q httpd

1. 启动httpd服务，设置开机启动

systemctl status httpd

sytemctl is-enabled httpd检测是否开机自启

1. 创建文件夹，挂载，检测

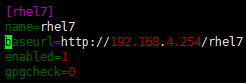


（4）开机自动

客户端：虚拟机A 虚拟机B

1. 修改配置文件

vim /etc/yum.repos.d/rhel7.repo



**发布网络yum源**

准备Yum源

yum仓库特点

作为yum源需要准备的内容

大量的.rpm软件包文件

针对这些软件包的repodata/仓库文档

repodata/仓库档案数据

filelists.xml.gz //软件包的文件安装清单

primary.xml.gz //软件包的基本/主要信息

other.xml.gz //软件包的其他信息

repomd.xml //提供.xml.gz下载和校验信息

使用RHEL7光盘库

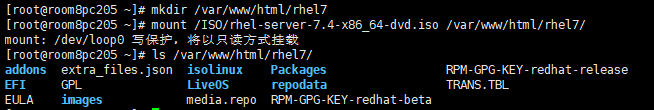
RHEL7光盘目录已经预先配置为yum源

packages/ //存放.rpm软件包

repodata/ //仓库档案

isolinux/ //光盘引导数据

RPM-GPG-KEY-redhat-release //签名校验密钥





使用第三方RPM包建库（具有非光盘的软件包）

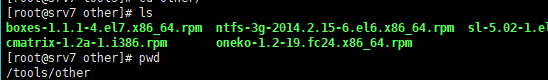
有众多软件包

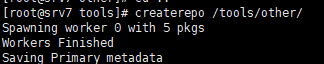
仓库数据文件（软件包列表）

需要额外创建repodata/档案

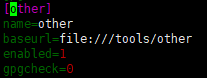
将搜集的.rpm包文件集中到指定目录

针对此目录文件使用createrepo工具创建仓库档案



建立仓库档案

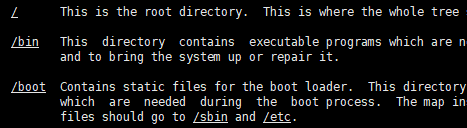
确认结果

指定新的yum仓库，file://指本地文件

**扩展的几个应用**

补充应用

目录结构：man hier



|  |  |
| --- | --- |
| 目录路径 | 主要用途 |
| /boot | 存放系统引导必须的文件，包含内核、启动配置 |
| /bin、/sbin | 存放各种命令程序 |
| /dev | 存放硬盘、键盘、鼠标、光驱等各种设备文件 |
| /etc | 存放linux系统及各种程序的配置文件 |
| /root、/home/用户名 | 分别是管理员root、普通用户的默认家目录 |
| /var | 存放日志文件、邮箱目录等经常变化的文件 |
| /proc | 存放内存中的映射数据，不占用磁盘 |
| /tmp | 存放系统运行过程中使用的一些临时文件 |

权限的数值表示

权限的数值化

基本权限：r=4,w=2,x=1

附加权限：SUID=4，SGID=2，sticky bit=1

采用数值形式设置权限

chmod -R nnn 文档

chmod -R xnnn 文档

历史命令

管理/调用曾经执行过的命令

history：查看历史命令列表

history -c：清空历史命令

!n：执行命令历史中的第n条命令

!str：执行最近一次以str开头的历史命令

调整历史命令的数量





实用小命令工具

du，统计文件的占用空间

du [选项]... [目录或文件]...

-s：只统计每个参数所占用的总空间大小

-h：提供易读容量单位（K、M等）



wc统计命令

wc -l：只统计行数

在etc底下以conf结尾的文件有多少个

date，查看/调整系统日期时间

date +%F、date +%R

date +”%Y-%m-%d %H:%M:%S”

date -s “yyyy-mm-dd HH:MM:SS”





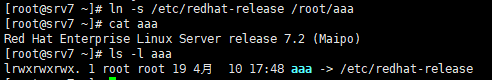


ln（软连接、硬链接）

ln，创建软连接（若原始文件或目录被删除，连接文件将失效，软连接可存放在不同分区/文件系统）（对原始文件做的快捷方式）

软连接-》原始文件-》i节点（编号，识别硬盘的存储区域）-》文档数据

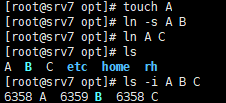
ln -s 原始文件或目录 软连接文件



ln，创建硬链接（若原始文件被删除，连接文件仍可用，硬链接与原始文件必须在同一分区/文件系统）（对i节点做的快捷方式）

硬链接-》i节点-》文档数据

ln 原始文件 硬链接文件



获取命令帮助

查看命令的简要说明

对于内部命令，使用“help 命令名”查看

对于外部命令，通常会提供“--help”帮助选项

使用man手册

一种集中的手册页文档机制

大多数程序/配置文件/库函数都提供man手册页

man，格式化手册阅读工具

按↑↓pgup pgon键滚动及翻页

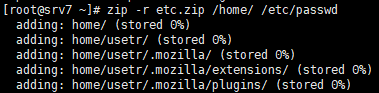
按q退出

按/键向后查找关键字（n、N切换）

zip归档工具（跨平台压缩工具）

制作.zip压缩包

zip -r 备份文件.zip 被归档的文档...



释放.zip压缩包

释放归档+解压操作

unzip 备份文件.zip [-d 目标文件夹]



**VIM编辑技巧**

命令操作模式

光标跳转

|  |  |  |
| --- | --- | --- |
| 操作类型 | 按键指令 | 用途 |
| 移动光标 | ↑↓←→ | 上、下、左、右 |
| 光标行内跳转 | home键或^、数字0 | 跳转至行首 |
| end键或”$”键 | 跳转至行尾 |
| 全文翻页 | pgup键、pgdn键 | 向上翻页、向下翻页 |
| 光标行间跳转 | 1G或gg | 跳转至文件的首行 |
| G | 跳转至文件的末尾行 |

复制/粘贴/删除

|  |  |  |
| --- | --- | --- |
| 操作类型 | 按键指令 | 用途 |
| 复制 | yy、#yy | 复制到光标处的一行、#行 |
| 粘贴 | p、P | 粘贴到光标处之后、之前 |
| 删除 | x或delete | 删除光标处的单个字符 |
| dd、#dd | 删除光标处的一行、#行（或剪切） |
| d^ | 从光标处之前删除至行首 |
| d$ | 从光标处删除至行尾 |

C（大写）：从光标处删除到行尾，并且进入插入模式

查找/撤销/保存

|  |  |  |
| --- | --- | --- |
| 操作类型 | 按键指令 | 用途 |
| 文本查找 | /word | 向后查找字符串“word” |
| n\N | 跳至后/前一个结果 |
| 撤销编辑 | u | 撤销最近的一次操作（可按多次） |
| U | 撤销对当前行的所有修改 |
| ctrl+r | 取消前一次撤销操作（反撤销） |
| 保存退出 | ZZ | 保存修改并退出 |

末行模式操作

保存/退出/文件操作

|  |  |  |
| --- | --- | --- |
| 操作类型 | 设置指令 | 用途 |
| 存盘及退出 | :w | 保存当前文件 |
| :q! | 放弃已有更改后强制退出 |
| :wq或:x | 保存已有修改后退出 |
| 文件操作 | :w /root/newfile | 另存为其他文件 |
| :r /etc/filesystems | 读入其他文件内容 |

字符串替换

|  |  |  |
| --- | --- | --- |
| 操作类型 | 设置指令 | 用途 |
| 行内替换 | :s/old/new | 替换当前行第一个“old” |
| :s/old/new/g | 替换当前行所有的“old” |
| 区域内替换 | :n,m s/old/new/g | 替换n-m行所有的“old” |
| :%s/old/new/g | 替换文件内所有的“old” |

开关参数设置

|  |  |  |
| --- | --- | --- |
| 操作类型 | 设置指令 | 用途 |
| 编辑器设置 | :set nu|nonu | 显示/不显示行号 |
| :set ai|noai | 启用/关闭自动缩进 |

**源码编译安装**

编译安装介绍

源码编译安装的优势

主要优点：

获得软件的最新版

软件功能可按需选择/定制，有更多软件可供选择

源码包适用于各种平台

准备编译环境

开源软件多使用C/C++开发

需要gcc、gcc-c++、make等编译工具





基本实现过程

1. 下载源码包（tar包，释放源代码至指定目录）
2. ./configure配置，指定安装目录/功能模块等选项

作用1：检测是否安装gcc

作用2：指定安装的功能和位置（只是记住）

1. make编译，生成可执行的二进制程序文件
2. make install安装，将编译好的文件复制到安装目录
3. 测试及应用/维护软件

**systemctl控制**

**开启服务：systemctl start 服务名**

**停止服务：systemctl stop 服务名**

**重启服务：systemctl restart 服务名**

**设置开机自启：systemctl enable 服务名**

**设置开机不自启：systemctl disable 服务名**

**查看当前是不是开机自启：systemctl is-enabled 服务名**

systemd介绍

init程序的作用

linux系统和服务管理器

是内核引导之后加载的第一个初始化进程（PID=1）

负责掌控整个Linux的运行/服务资源组合

传统的init程序风格

system v：顺序加载，RHEL5系列采用

upstart：时间触发,rhel6系列采用

systemd

一个更高效的系统&服务管理器

开机服务并行启动，各系统服务间的精确依赖

配置目录：/etc/systemd/system

服务目录：/lib/systemd/system

主要管理工具：systemctl

unit配置单元

不同的unit决定了一组相关的启动任务

service：后台独立服务

socket：套接字，类似于xinetd管理的临时任务

target：一套配置单元的组合，类似于传统“运行级别”

device：对应udev规则标记的某个设备

mount、automount：挂载点、触发挂载点

列出服务

列出活动的系统服务：systemctl -t service

列出所有的系统服务（包含不活动的）：systemctl -t service --all

管理运行级别

切换级别

列出可用运行级别systemctl -t target

切换到文本/图形模式

systemctl isolate multi-user.target

systemctl isolate graphical.target

设置默认级别

查看默认级别systemctl get-default

设置默认级别systemctl set-default multi-user.target

systemctl set-default graphical.target