**达内集团训练营**

**2018年11月23号**

**Day01**

存储器

内存：也称内部存储器，用于暂时存放CPU中程序运算数据,以及与硬盘等外部存储器交换数据

　　1) RAM：可读可写易更改

　　2) ROM：可读不可写不易更改

外存

优盘

硬盘

硬盘: 是主要的电脑存储媒介

容量(500GB、1T、2TB)

转速

类型

尺寸

主板: 集成了计算机的主要电路系统,提供一系列芯片及扩展槽和其他连接配件

光驱(光盘驱动器): 用来读取光盘介质的驱动器设备

显卡、声卡、网卡: 处理图像音频、网络数据的专用设备,通常由主板集成,有特殊需求也可以独立配备;

主要指标:

显存大小(1G、2G、4G) 显口类型

网卡速率(百兆、千兆)、双网卡,无线网卡

机箱及电源

机箱用来固定主板......

Windows起源与发展

操作系统: 一套特殊的计算机软件,为其他应用软件提供访问计算机硬件的服务,提供有效的...

常见的操作系统

UNIX: 苹果手机

Linux: 安卓手机

Windows:

Mac OS

什么是虚拟机

虚拟: 由高科技技术实现的仿实物或伪实物的技术

虚拟机的运行架构

寄居架构

虚拟机作为....

安装在虚拟机中的操作系统称为客户

虚拟机平台产品

RedHat

KVM

RHEV平台

Vmware

VMware workstation

VMware vSphere

克隆虚拟机前先关闭虚拟机

快照: 可以保存虚拟机在特定时刻的状态和数据

状态包括虚拟机的电源状态(例如: 打开电源,关闭电源,挂起);

数据包包括组成虚拟机的所有文件,包括磁盘,内存和其他设备(例如虚拟网络接口卡);

克隆和快照的区别:

|  |  |  |
| --- | --- | --- |
|  | 快照 | 克隆 |
| 创建时间 | 不限 | 虚拟机关机时才可以 |
| 创建数量 | 不限 | 不限 |
| 占用磁盘空间 | 由创建的数量决定 | 较小,由创建的数量决定,完整克隆较大 |
| 用途 | 保存虚拟机某一时刻状态 | 分发创建的虚拟机 |
| 是否独立 | 不能脱离原始虚拟机独立运行 | 链接克隆:部分脱离  完整克隆:完全脱离 |

链接克隆类似于快捷方式

用户账号概述

登录名和密码控制

1) 每个用户账号包含唯一的登录名和对应的密码

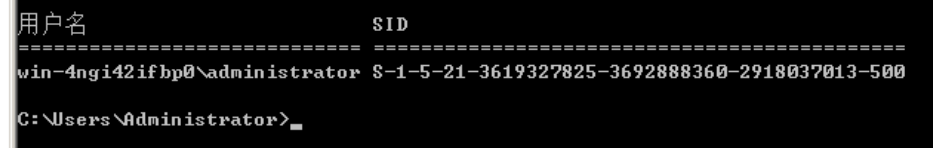
2) 不同的用户身份拥有不同的权限

系统内核根据SID识别不同用户

1) 每个用户账号拥有唯一的安全标示符(SID)

2) 用户权限通过用户的SID来记录

SID格式实例:



(在windows终端中输入: whoami /user)

windows新建用户账号

复杂度要求: 密码总必须包含以下4类字符串中的三类字符:

1) 英文大写字母 (A-Z)

2) 英文小写字母 (a-z)

3) 10个基本数字 (0-9)

4) 特殊符号 [(!@#%...)](mailto:(!@#%...))

长度要求: 最小密码长度为8位

分区:

毛坯------> 打隔断------> 装修------> 入住

硬盘------> 分区--------> 格式化-------> 读/写

格式化:

windows文件系统: 1) NTFS 2) FAT32

Linux文件系统: 1) ext4 2) xfs

windows磁盘分类

基本磁盘

传统磁盘,默认是基本磁盘

磁盘的大小固定扩展不方便,数量受限

动态磁盘

支持多种特殊的卷

比如实现卷的大小动态调整,磁盘的高可用等

windows基本磁盘分区类型

计算机中存放信息的主要的存储设备就是硬盘,但是硬盘不能直接使用,必须对硬盘进行分割,分割成的一块一块的硬盘区域就是磁盘分区

盘符是windows系统对于磁盘存储的标识符,一般使用26个英文字符加上一个冒号:来标识

MBR分区类型

在一个MBR分区表类型的硬盘中最多只能存放4个主分区,如果一个硬盘上需要超过4个以上的磁盘分快的话,那么就要使用扩展分区了;如果使用扩展分区,那么一个物理硬盘上最多只能3个主分区和一个扩展分区;扩展分区不能直接使用,它必须经过第二次分割成为一个一个的逻辑分区,然后才可以使用;一个扩展分区中的逻辑分区可以任意多个

windows动态磁盘

动态磁盘是磁盘的一种属性,而我们常用的磁盘属性是基本磁盘,在动态磁盘上没有分区的磁盘概念它以”卷”命名,卷和分区差距很大,同一个分区只能存在一个物理磁盘上,而同一个卷可以跨越多达32个物理磁盘

使用动态磁盘的好处:

1) 可扩展性

2) 高读写性能

3) 高可靠性

基本磁盘转换为动态磁盘

动态卷建立在动态磁盘之上

右击基本磁盘---->转换到动态磁盘

注意事项:

1) Administrations或者Backup Operators组成员

2) 转换之前关闭所有正在运行的程序

3) 转换为动态磁盘后,原主分区与逻辑分区自动变成简单卷,整个磁盘内不再有基本卷

4) 除非删除所有卷,否则动态磁盘不能转换为基本磁盘

跨区卷

动态磁盘是磁盘的一种属性,而我们常用的磁盘属性是基本磁盘

什么是跨区卷?

由多个位于不同磁盘的未分配空间组成;

每一个成员卷的容量大小可以不同

新建跨区卷

需要至少两块动态磁盘

容量是组成跨区卷的所有磁盘空间总和

带区卷

什么是带区卷?

1) 由多个位于不同磁盘的未分配空间组成

2) 每一个成员卷的容量大小是相同的

新建带区卷

1) 以带区形式在两个或多个磁盘上存储数据的动态卷

2) 数据被均匀以带区形式跨磁盘交替分配

3) 需要至少两快动态磁盘

4) 容量是组成带区卷所有磁盘空间总和

######################################

Github

1 来历

SVN: VisualSVN

软连接------符号链接

快捷方式-----文件

硬链接----狡兔三窟,每一个都等价,兔子=数据

git使用步骤：

1.创建仓库

2.克隆到本地

3.将需要的文件放到git目录下

4.配置:

# git config --globle user.email 账号

· 同步

# git pull

5.如果目录下有新的文件

# git add -A .

6.提交更改

# git commit

输入备注

7.上传

# git push

账号

密码

#######################################

**Day02**

未列出--->

用户名：root

密码：Taren1

day02 初识Linux

1、 Linux是一个操作系统！！

操作系统=内核+外围的应用软件

2、三大操作系统：

Unix

Linux：redhat、centos......

Windows

开源：开放源代码

top500.org

3、2017年11月13日，全球超级计算机500强榜单公布，

中国超级计算机"神威·太湖之光"和"天河二号"

连续第四次分列冠亚军

第三：瑞士“代恩特峰”

第四：日本“晓光”

第五：美国“泰坦”

4、Linux的诞生

Linux之父，Linus Torwalds

1991年10月，发布0.02版（第一个公开版）内核

1994年03月，发布1.0版内核

标准读音：“哩呐科斯”

5、如何使用硬盘？

硬盘==>分区==>格式化===>读/写文档

毛坯==>打隔断==>装修==>入住

6、存储器

内存

RAM（随机访问存储器） 可读可写（也就是可修改）易丢失

ROM（只读存储器）可读不可写不易丢失

外存（永久）：硬盘

7、传统的MBR分区模式

3个主分区+1个扩展分区（n个逻辑分区）

注意事项

MBR分区表只能记录4个主分区，编号范围1-4

当有必要使用4个以上分区时，需要建立扩展分区（占主分区编号），扩展分区最多只能有1个

扩展分区作为一个中间性质的容器，

从其中再划分出的分区空间即为逻辑分区

8、Linux里面一切皆文件

9、计算机：/（根目录）

/dev

/dev/sda5/nsd11

10、

hd，表示IDE设备

sd，表示SCSI设备

/dev/hda2

/设备所在目录/【[hd|sd]磁盘序号\*分区序号】

/dev/sdb1

11、Linux文件系统

EXT4，第四代扩展文件系统，RHEL6系列默认

XFS，高级日志文件系统，RHEL7系列默认

SWAP，交换空间（虚拟内存）

12、安装

注：光盘--->系统

光盘->/iso/rhel-server-7.4-x86\_64-dvd.iso(镜像)

root 超级管理员

#########################################

环境

# rht-vmctl reset server

# rht-vmctl reset desktop

# rht-vmctl reset classroom

克隆虚拟机

# clone-vm7

# Enter VM number:1

你的KVM会出现：rh7\_node01

localhost login: root

PassWord:【123456】

[root@localhost~]# init 5 进入图形模式

点击“未列出”

--->用户名：root 密码：123456

Tab键的运用

列出当前系统的主机名称

[root@localhost ~]# hostname

localhost.localdomain

临时修改主机名

[root@svr7 桌面]# hostname tarena

关闭当前终端，重新开启。主机名称会改变。

[root@tarena 桌面]#

永久修改主机名

[root@svr7 桌面]# hostnamectl set-hostname erha.tedu.cn

列出已激活的网卡连接信息

[root@svr7 桌面]# ifconfig

命令行的格式：

命令字 [选项] [参数]

选项:

短选项: -l

复合选项: -lh

长选项: --help

命令行提示标识的含义

[当前用户@主机名 工作目录]#

##查看当前用户

[root@xiaohua ~]# whoami

root

查看当前工作目录

pwd — Print Working Directory

用途：查看当前工作目录

[root@xiaohua ~]# pwd

/root

[root@xiaohua ~]# cd /boot

[root@xiaohua boot]# pwd

/boot

创建用户

[root@xiaohua ~]# useradd lisi

[root@xiaohua ~]# id lisi

切换用户

[root@xiaohua ~]# su - lisi

查看当前工作目录

[lisi@xiaohua ~]$ pwd

/home/lisi

登出，退出到root的登录状态

[lisi@xiaohua ~]$ exit

[root@xiaohua ~]# pwd

/root

~：代表的是家目录

#：代表当前登录的是root

$：代表当前登录的是普通用户

/home:普通用户的家目录

/root：超级管理员root的家目录

#########################################

**Day03**

快捷键:

ctrl + shift + “+”: 将终端字符变大

ctrl + “-”: 将终端字符变小

ctrl + u: 删除光标前面的内容

ctrl + k: 删除光标后面的内容

ctrl + l(或clear): 清空整个屏幕

ctrl + c: 终止当前程序

ctrl + w: 往回删除一个单词(以空格为界)

ctrl +shift + t: 以标签形式打开多个终端

alt + “数字”: 切换终端

ctrl + shift + w: 关闭当前终端

ctrl + shift + d: 关闭终端 (保留有命令行的页面,有命令行的终端关闭不了)

history: 查看历史操作命令

history -c: 清空历史操作命令

调用历史命令: !”命令编号”

检索敲过的命令: ctrl + r

添加快捷键

[root@Blizzard ~]# which gnome-terminal

/usr/bin/gnome-terminal

复制粘贴/usr/bin/gnome-terminal--->点击关机按钮--->点击小扳手--->(设备)---->键盘--->出现的界面拉到最后--->你会看到一个小”+”,并点击它---->出现添加自定义快捷键的页面

名称: 终端

命令: /usr/bin/gnome-terminal

快捷键: ctrl + “E”

点击添加

ls ----- list (多参数命令)

用途: 用来显示目录里面的内容

格式: ls [选项] [目标文件位置]

-l: 长格式显示,显示属性信息

-h: 以人类可以读取的方式显示

-d: 只显示目录本身

-R: 递归显示

黑色: 文件

蓝色: 目录

红色: 压缩文件

绿色: 可执行文件

浅蓝色: 链接

man 在线帮助手册

/: 代表即将进行搜索

/-h: 搜索的内容为”-h”

按”N”: 向上匹配

按”n”: 向下匹配

按”q”: 退出

# man man

# man ls

# man pwd

转换成英文: # LANG=en\_US

转换成中文: # LANG=zh\_CN.UTF-8

cd ---- change directory

用途: 切换工作目录

格式: cd [目标文件位置]

. 当前路径

.. 父目录(上一层)

- 返回

绝对路径: 这种方式以根目录”/”为起始;使用绝对路径可以非常准备的表示一个目标(或)文件

相对路径: 表示以当前工作目录为起点,在开头不使用”/”符号,使用时更加简短、便捷

创建目录

mkdir ----- mkdir directory

格式: mkdir [路径/]目录名

例: mkdir yaya

如果想递归创建目录,需要加参数-p

例: # mkdir -p a/b/c/d

[root@Blizzard ~]# mkdir -p f/{s,k}/p

[root@Blizzard ~]# ls -R f/

创建空白文件或更改时间戳

touch

格式: touch [路径]文件名

touch a.txt

touch {c,b}.txt

查看文件内容

cat

# cat /etc/passwd

复制

cp ---- copy

格式: cp [选项]... 原文件... 目标路径

-p: 完全相同的复制

-r: 递归复制(复制目录时用)

[root@Blizzard yaya]# cp /etc/passwd .

[root@Blizzard yaya]# cp /etc/passwd mima #把passwd复制到当前路径,并命名为mima

[root@Blizzard yaya]# cp -r a f/ # 把目录a复制到目录f下

删除

rm ------ remove

格式: rm [/路径/]文件名

-r: 递归 (删除目录时用)

-f: 强制

移动: 相当于windows中的剪切,移动的过程也可以实现重命名

mv

检查红帽发行信息

[root@svr7 桌面]# cat /etc/redhat-release

Red Hat Enterprise Linux Server release 7.2 (Maipo)

列出内核版本

[root@svr7 桌面]# uname -r

3.10.0-327.el7.x86\_64

#########################################

认识vim编辑器

1）

vim编辑器工作模式

命令模式

输入模式

末行模式

2）光标间的跳转(命令模式下)

光标行间跳转

gg：跳转到文件的行首

G：跳转到文件的行末尾

3）

命令模式下

dd: 删除

2dd：删除两行

u：撤销

yy：复制

2yy：复制两行

p：粘贴

x或delete: 删除单个字符

4）保存退出

:wq 或 :x 保存

:q! 强制退出

:w 写入

:q 退出

5）字符串替换(命令模式下)

（1）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”

（2）区域内替换

:n,m s/old/new/g 替换第n-m行所有的“old”

:%s /old/new/g 替换文件内所有的“old”

6）

命令模式下

:set number # 显示行号

:set nonumber # 不显示行号

:set tabstop=4 # 设置tab键为4个空格

1. vimtutor ------vim教程工具

######################################

**Day04**

vim编辑器: vi编辑器的增强版本

# vi /etc/fstab

# vim /etc/fstab

vim的工作模式

三种模式及切换

a、i、o等键

输入模式

命令模式

末行模式

esc

esc

: 键

|  |  |  |
| --- | --- | --- |
| **操作类型** | **按键指令** | **用途** |
| 移动光标 | ↑、↓、←、→ | 上、下、左、右 |
| 翻页 | PageUp键、PageDown键 | 向上/下翻动一整页内容 |
| 光标行内跳转 | Home键、End键 | 跳转至行首、行尾 |
| 光标行间跳转 | gg或1G | 跳转到文件的首行 |
| 10G | 到全文的第10 行 |
| G | 跳转到文件的末尾行 |

光标跳转

复制、粘贴、删除

|  |  |  |
| --- | --- | --- |
| **操作类型** | **按键指令** | **用途** |
| 复制 | yy 、 #yy | 复制光标处的一行、#行 |
| 粘贴 | p | 粘贴到光标处之后 |
| 删除 | x 或Delete键 | 删除光标处的单个字符 |
| d^、 d$ | 删除到行首、删除到行尾 |
| dd、 #dd | 删除光标处的一行、#行 |

查找字符串

|  |  |  |
| --- | --- | --- |
| **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 | 跳至下一个/上一个找到的字符串 |

:nohl 取消高亮

补充: 在命令模式下大写的C,可以删除光标之后,并且进入输入模式

TCP/IP协议

TCP/IP族是获最广泛支持的通信协议集合,包括大量的Internet应用中的标准协议,支持跨网络架构、跨操作系统平台的通信

主机与主机之间通信的三个要素

1) IP地址------->门牌号

2) 子网掩码------>小区

3) IP路由-------->高德地图

什么是IP地址?

作用: 用来标识一个节点的网络地址

地址组成(点分十进制):

一共32位二进制位

转换为4个十进制数,以”.”隔开

MAC: 物理地址 -------->身份证号

IP: 逻辑地址 ---------->手机号

IP地址版本

ipv6

ipv4

IP地址分类:

适用于一般计算机网络

A类 1-126 子网掩码 255.0.0.0

B类 128-191 子网掩码 255.255.0.0

C类 192-223 子网掩码 255.255.255.0

组播及科研专用

D类: 224-239 组播

E类: 240-254 科研

子网掩码

A类: 255.0.0.0

网络位+主机位+主机位+主机位

B类: 255.2550.0

网络位+网络位+主机位+主机位

C类: 255.255.255.0

网络位+网络位+网络位+主机位

192.168.1.1

网.网.网.主

192.168.1.0 ------>网络ID

网路ID(网段)的由来

11000000. 10101000. 00000001. 00000001 ----IP地址(192.168.1.1)

11111111. 11111111. 11111111. 00000000 --子网掩码(255.255.255.0)

11000000. 10101000. 00000001. 00000000 ----网络ID(192.168.1.0)

网络ID=IP地址 \* 子网掩码

网络ID的算法: IP地址与子网掩码转换成二进制并相与

何为同一个网段?

网络ID相同即为同一个网段

例: 172.16.31.8和172.16.30.1为同一个网段

0 ---[1-254]

255 ---广播

127 ----本地回环地址

IP地址的分类:

根据适用范围区分

公有地址: 可以在互联网合法使用,需要向NIC付费申请

私有地址: 预留给企业内部使用,无需付费

回环地址: 测试本机TCP/IP专用(127.0.0.1~127.0.0.254)

私有IP地址范围

A类: 10.0.0.1~10.255.255.254

B类: 172.16.0.1~172.31.255.254

C类: 192.168.0.1~192.168.255.254

什么是网关?

一个网络连接到另一个网络的”关口”; 通常是一台路由器,或者防火墙/接入服务器; 实质上是一个网络通向其他网络的IP地址

windows查看IP参数

按win + R --->运行: cmd ----打开命令窗口

执行ipconfig /all 查看详细IP配置

测试网络连通性

基本用法: ping 目标主机地址

网络排错常见思路:

1) 先ping回环地址127.0.0.1 # 检查TCP/IP驱动

2) 再ping本网段其他主机 # 检查内网通信

3) 再ping网关 # 检查出口路由

4) 再ping其他网段的其他主机 # 检查远程连通性

windows系统:

1) ping IP地址

2) ping 主机名

3) ping 主机名 -4

######################################

**Day05**

什么是计算机网络

硬件方面: 通过线缆将网络设备和计算机连接起来

软件方面: 操作系统,应用软件,应用程序通过线路互联

实现资源共享,信息传递

计算机网络的功能:

1) 数据通信

2) 资源共享

3) 增加可靠性

4) 提高系统处理能力

WAN与LAN

广域网(Wide-Area Network)

范围: 几十到几千千米

作用: 用于连接远距离的计算机网络

典型应用: Internet

局域网(Local-Area Network)

范围: 1KM左右

作用: 用于连接较短距离内的计算机

典型应用: 企业网,校园网

网络设备及拓扑图

网络设备生产厂商: Cisco, 华为

网络拓扑结构

线缆连接计算机和网络设备的布局

1) 点对点

2) 总线型

3) 环形

4) 星型及扩展的星型

5) 网状

点对点拓扑结构

两台设备之间有一条单独的连接

专用的广域网中电路连接的两台路由器

总线型拓扑结构

一根线缆连接所有的设备

主线缆必须以终结器结束,吸收末端信号

早期局域网拓扑

使用同轴线缆

环形拓扑结构

所有设备共享一条线缆,数据沿一个方向传输

令牌环网络

星型拓扑结构

优点: 易于实现; 易于网络拓展; 易于故障排查;

缺点: 中心节点压力大;组网成本较高

网状拓扑结构:

一个节点与其他多个节点相连

提供冗余性和容错性

可靠性高

组网成本高

线缆

有线介质

光纤: 一种传输光能的波导介质,一般由纤芯和包皮组成

双绞线: 由许多绝缘的对线交叉组成的能够数据传输的线(使用广泛,价格便宜),相互缠绕,减少对临近线对的电气干扰

双绞线分类(按屏蔽层分类)

非屏蔽双绞线(UTP)

屏蔽双绞线(STP)

线缆的连接

T568B标准：白橙，橙，白绿，蓝，白蓝，绿，白棕，棕

T568A标准：白绿，绿，白橙，蓝，白蓝，橙，白棕，棕

|  |  |  |
| --- | --- | --- |
| 管脚号 | 用途 | 颜色 |
| 1 | 发送+ | 白色和绿色 |
| 2 | 发送- | 绿色 |
| 3 | 接收+ | 白色和橘黄色 |
| 4 | 不被使用 | 蓝色 |
| 5 | 不被使用 | 白色和蓝色 |
| 6 | 接收- | 橘黄色 |
| 7 | 不被使用 | 白色和棕色 |
| 8 | 不被使用 | 棕色 |
| T568A标准中RJ-45连接器的管脚号和颜色编码 | | |

直通线: 同一根网线的两端使用同样的线序,即网线的两端都使用568A或568B的是直连线(实际运用中一般都使用568B);直通线用于不同设备的相连,例如电脑连交换机、路由器连交换机

交叉线: 同一根网线的两端使用不同的线序,即网线的两端,一端用568A,一端用568B的是交叉线;交叉线用于同设备连接,例如电脑连电脑、交换机连交换机;

特例: 路由器与PC: 交叉线; 交换机与交换机: 交叉线或直通线

**Cisco交换机的命令行**

用户模式

Switch>

特权模式(一般用于查看配置信息)

Switch>enable

Switch#

全局配置模式(所做的配置对整个设备生效)

Switch#configure terminal

Switch(config)#

Switch(config)#interface f0/1 (interface fastEthernet 0/1)

Switch(config-if)#

模式间的转换

退到上一个模式: exit

退到特权模式: end、 <Ctrl+z>

运行出错时按Ctrl+shift+”6”可以退出

配置主机名

Switch>

Switch>enable

Switch#

Switch#configure terminal

Switch(config)#hostname tedu tedu: 主机名

tedu(config)#

查看交换机配置

tedu# show running-config

tedu# show run

显示版本信息

tedu# show version

配置enable明文口令

tedu>enable

tedu#configure terminal

tedu(config)#enable password 123

取消明文密码:

tedu(config)#no enable password

配置enable加密口令

tedu>enable

tedu#configure terminal

tedu(config)#enable secret 456

取消加密口令:

tedu(config)#no enable secret

配置Console口令

tedu>enable

tedu#configure terminal

tedu(config)#line console 0

tedu(config-line)#password 789

tedu(config-line)#login

tedu(config-line)#end

交换机有一个控制端口(console),其编号为0,通常利用该端口进行本地登录,以实现对交换机的配置和管理,为安全起见,应为该端口的登录设置密码

旗标

配置旗标不仅仅是为了耍酷;可以让任何登录到路由器的人显示你想告诉的信息

#banner motd x......x

例子:

tedu>enable

tedu#configure terminal

tedu(config)#banner motd #

“你想告诉的信息”#

tedu(config)#end

tedu#exit

“你想告诉的信息”

######################################

云计算基础

云计算运维

数据库管理

云架构管理

Python开发

**第一阶段 ----牛犇**

**I-Day01**

第一阶段课程内容共19天

云计算系统管理、 云计算应用管理、 系统&服务管理进阶

一 教学环境

真机: Linux操作系统 用户名: root 密码: Taren1 (符合四分之三原则)

虚拟机软件: 利用一款软件,虚拟众多的硬件

win2008虚拟机: 密码: Taren1

什么是云计算: 由众多的服务器组成

服务器: 能够为其他计算机提供服务的更高级的电脑;

机架式, 塔式, 机柜式, 刀片式

典型服务模式

Ｃ/S, client/server架构

由服务器提供资源或某种功能

客户机使用资源或功能

TCP/IP协议及配置

TCP/IP是最广泛支持的通信协议集合

包括大量Internet应用中的标准协议

支持跨网络架构,跨操作系统平台的通信

主机与主机之间通信的三要素:

IP地址(IP address), 子网掩码(subnet mask), IP路由(IP router)

IP地址: 唯一标识网络中的一台主机; 由32个二进制组成,用点分隔4部分,最后利用4个十进制标示

二进制: 由0和1组成

IPv6: 由128个二进制组成

为虚拟机win2008配置IP地址: 192.168.1.100

操作步骤:

右击桌面网络---->属性----->更改适配器设置

双击”本地连接”------>属性

双击”Internet协议版本4(TCP/IP)”

配置完成后,单机”确定”完成

查看IP地址

操作步骤:

右击桌面网络---->属性----->更改适配器设置

双击”本地连接”------>详细信息

通过命令查看IP地址: ipconfig

图形命令行终端: 开始------>运行------>cmd 回车

开始----->运行 = windows+r

exit: 关闭图形命令行终端

IP地址的分类

适用于一般计算机网络

A类 1-127 网+主+主+主

B类 128-191 网+网+主+主

C类 192-223 网+网+网+主

组播及科研专用

D类: 224-239 组播

E类: 240-254 科研

IP地址: 网络位+主机位

座机号:区号+号码

010-12345678 北京

0312-7357777 保定

区号: 标识一个区域 号码: 标识区域中一个座机

010-12345678=在北京区域,有一个编号为12345678的座机

网络位: 标识一个网络(网段)

192.168.1.1=在192.168.1网络中,有一个编号为1的主机

192.168.2.10=在192.168.2网络中,有一个编号为10的主机

子网掩码: 用来为计算机标识该IP地址的网络位和主机位;用二进制的0标示主机位,用二进制的1标示网络位

IP地址的表示方式: 192.168.1.1 255.255.255.0=192.168.1.1/24

测试通信:

1 准备第二台机器,利用虚拟机克隆实现

2虚拟机克隆

1) 关闭win2008虚拟机

2) 进行克隆,双击桌面”虚拟系统管理器”----->右击win2008---->克隆

3 两台机器均开机,密码为Taren1

4 win2008-clone修改桌面背景:

控制面板---->显示----->更改桌面背景

5 配置win2008-clone IP地址192.168.1.200/24

6 利用ping命令测试通信

命令格式: ping 对方IP地址

ping命令是双向的,有来有回

Windows防火墙,默认拒绝ping操作,

控制面板---->Windows防火墙--->打开或关闭防火墙--->关闭防火墙

IP地址配置方式:

1 手工配置

2 自动获取IP地址,在网络中搭建DHCP服务器(自动分配IP地址)

网关: 从一个网络连接到另一个网络的”关口”

作用: 解决不同网络之间通信

DNS服务器: 域名服务器, 将在浏览器中输入的域名解析为对应的IP地址

#####################################

**I-Day02**

Linux系统简介

Linux是一种操作系统,互联网领域幕后的老大

操作系统: 一大堆软件的集合,让计算机正常的工作

UNIX生辰: 1970-1-1

Linux的诞生

Linux之父，Linus Torwalds

1991年10月，发布0.02版（第一个公开版）内核

1994年03月，发布1.0版内核

标准读音：“哩呐科斯”

内核: 调配所有的硬件

用户----->内核----->调配所有硬件

版本号: 主版本.次版本.修订号

发行版的名称/版本由发行方决定

Red Hat Enterprise Linux 5/6/7

Suse Linux Enterprise 12

Debian Linux 7.8

Ubuntu Linux 14.10/15.04

Linux是一套公开发布的完整Linux系统; Linux+ 各种外围软件(应用软件)

Red Hat系列版本

1) 红帽Linux企业版

简称RHEL(Red Hat Enterprise Linux)

http://www.redhat.com/

1. Fedora Core社区版

http://fedoraproject.org/

3) CentOS 社 区企业操作系统

<http://www.centos.org/>

Ctrl+alt: 鼠标回到真机

**安装RHEL7系统**

在虚拟机安装RHEL7系统

1 建立新的虚拟机

如何使用硬盘?

一块硬盘的”艺术”之旅

物理硬盘---->分区规划---->格式化---->读/写文档

毛坯楼层---->打隔断------->装修------->入驻

格式化: 赋予空间文件系统的过程

文件系统: 存放数据的规则

Windows文件系统: NTFS FAT

Linux文件系统: ext4 (RHEL6) xfs (RHEL7)

swap(交换文件系统,也叫虚拟内存): 缓解内存不足

Linux目录结构: 树型结构

/ 根目录: Linux系统的起点(所有数据都是在此目录下)

路径: /abc/test/1.txt (只有开头第一个’/’是根目录,后面的都是分隔符)

Linux分区表示方式

/dev: 存放设备相关的数据 (硬盘,分区,键盘,鼠标,显示器......)

hd: 表示IDE设备

sd: 表示SCSI设备

/dev/hda /dev/hdb /dev/hdc /dev/hdd

/dev/sda /dev/sdb /dev/sdc /dev/sdd

/dev/sda1: 表示SCSI设备,第一块,一个分区

获取命令行界面

虚拟控制台切换(Ctrl + alt + Fn 组合键) n: 是数字

tty1: 图形桌面

tty2~tty6: 字符控制台

图形命令行终端:

右键”打开终端”

应用程序--->系统工具--->终端

图形命令快捷键:

ctrl + shift + ‘+’: 放大

ctrl + ‘-’: 放小

ctrl + c: 结束正在运行的命令

命令行提示符标识的含义:

[当前用户@主机名 当前所在位置]$; 若当前用户是root,则最后一个字符为# ;否则,最后一个字符为$

**命令**

显示/修改主机名

hostname # 查看主机名

hostname 主机名 # 临时修改主机名

关闭当前终端,开启新的终端查看新的提示符 (重新开启一个新的终端才会在终端上显示新的主机名)

查看及切换目录

pwd---Print Working Directory

用途: 查看当前工作目录

cd----Change Directory

用途: 切换工作目录

格式: cd [目标文件夹位置]

[root@localhost ~]# cd /etc/pki/CA

[root@localhost CA]# cd ../..

[root@localhost etc]# cd ..

[root@localhost /]#

ls----list (多参数命令)

格式: ls [选项].. ..[目录或文件目录] 蓝色: 目录 黑色: 文本文件

-l: 显示详细信息

显示文本文件内容: cat

格式: cat 文件路径

[root@localhost /]# cat /root/anaconda-ks.cfg

[root@localhost /]# cat /etc/redhat-release # 显示系统版本

Red Hat Enterprise Linux Server release 7.4 (Maipo)

查看CPU处理器信息

lscpu

[root@room9pc01 ~]# lscpu | head -13 | tail -1

型号名称： Intel(R) Core(TM) i5-6500 CPU @ 3.20GHz

检查内存大小,空闲情况

cat /proc/meminfo

[root@room9pc01 ~]# cat /proc/meminfo | head -1

MemTotal: 16146728 kB

命令完成格式: 命令字 选项 参数1 参数2 .. ..

列出内核版本:

uname -r

[root@localhost /]# uname -r

3.10.0-693.el7.x86\_64

列出网络信息

ifconfig

ifconfig 网卡 IP地址 # 临时设置IP地址

eth0: 第一张网卡

lo: 回环接口 127.0.0.1------永远代表本机

virbr0: 虚拟网卡

关机: poweroff

重启: reboot

创建目录 (多参数命令)

mkdir-----make Directory

格式: mkdir [/路径/]目录名

创建新空白文件 (多参数命令)

touch

格式: touch [/路径/]文件名...

绝对路径与相对路径:

绝对路径: 以根目录开始的路径

相对路径: 不以根目录开始的路径 (以当前路径参照)

查看大文本内容操作

less分屏阅读工具

格式: less [选项] 文件名.....

优势: 支持前后翻页

基本交互操作

1) 按/键向后查找

2) 按上下键进行滚动

3) 按q退出

head tail 命令(无数字时,默认输出10行)

格式: head -n 数字 文件名 = head -数字 文件名

tail -n 数字 文件名 = tail -数字 文件名

grep工具

用途: 输出包含指定字符串的行

格式: grep [选项]... ‘查找条件’ 目标文件

[root@localhost ~]# grep root /etc/passwd

root:x:0:0:root:/root:/bin/bash

operator:x:11:0:operator:/root:/sbin/nologin

返回上一层目录

cd ..

[root@localhost ~]# cd /etc/pki/CA

[root@localhost CA]# cd ../..

[root@localhost etc]# cd ..

[root@localhost /]#

vim文本编辑器

三个模式: 命令模式 插入模式(编辑模式、输入模式) 末行模式

自带创建文件功能

三种模式及切换

a、i、o等键

输入模式

命令模式

末行模式

esc

esc

: 键

末行模式: wq 保存并退出

末行模式: q! 强制不保存退出

[root@localhost /]# vim /opt/1.txt

青春无悔，不忆过往，展望未来

[root@localhost /]# cat /opt/1.txt

青春无悔，不忆过往，展望未来

#########################################

**复习:**

1 Linux目录结构中”/”与”/dev”主要存放的内容?

/根目录: 所有数据都存放在此目录下(Linux系统的起点)

/dev: 设备相关的数据

2 Linux磁盘的表示,第三块SCSI磁盘的第二个分区?

/dev/sdc2

3 Linux常见的文件系统都有哪些?

ext4 xfs swap

4 Linux查看文本文件内容命令是?

cat less

5 Linux查看目录内容是?

ls

6 Linux查看主机的命令是?

hostname

7 Linux中查看IP地址的命令是?

ifconfig

8 Linux切换到/dev目录的命令是?

cd /dev

9 Linux显示当前位置的命令是?

pwd

10 Linux中显示/root目录详细属性怎么实现?

ls -l /root

11 Linux中显示当前主机的内核版本的命令?

uname -r

12 Linux查看CPU信息命令是?

lscpu

13 Linux查看内存信息命令是?

cat /proc/meminfo

14 Linux中查看/boot目录如何操作?

ls /boot

15 Linux中查看/etc/passwd文件前两行如何操作?

head -2 /etc/passwd

16 Linux中查看/etc/passwd中包含root的行,如何操作?

grep root /etc/passwd

**#########################################**

**I-Day03**

**命令行基础**

Linux命令: 用来实现某一类功能的指令或程序,

Linux执行命令,需要找到命令的执行程序;

命令的执行依赖于解释器(默认的解释器: /bin/bash), 绿色: 可以执行的程序

用户操作---->解释器(翻译官)---->内核---->调配所有硬件

解释器放在/etc/shells

[root@localhost ~]# cat /etc/shells

/bin/sh

/bin/bash #默认的解释器

/sbin/nologin #禁止用户登录系统

/usr/bin/sh

/usr/bin/bash

/usr/sbin/nologin

/bin/tcsh

/bin/csh

Linux命令分类:

内部命令: 属于解释器的一部分

外部命令: 解释器之外的其他程序

命令的一般格式:

基本用法: 命令字 [选项].. .. [参数1] [参数2].. ..

选项: 调控命令的执行方式

短选项 多个短选项(复合选项) 长选项

参数: 命令的操作对象,如目录或文件

快速编辑技巧

tab键自动补全: 可补全命令字,选项,参数,文件路径,软件名,服务名

快捷键:

esc + . 或 alt + . : 粘贴上一个命令的参数

ctrl + l 或者 clear : 清屏

ctrl + c : 废弃当前编辑的命令行(结束正在运行的指令)

ctrl + u : 清空至行首(不包含光标处)

ctrl + w : 往回删一个单词(以空格界定)

ctrl + shift + t : 新开一个终端

**mount挂载操作:** 让目录成为设备的访问点

安装软件

1) 连网下载

2) 从安装系统的光盘中获得众多的软件包

Windows访问光盘内容:

光盘文件------>光驱设备----->CD驱动器(双击) (访问点)

Linux访问光盘内容:

光盘文件------>光驱设备(/dev/cdrom)----->访问点(目录)

青色: 代表快捷方式

在Linux访问设备内容必须通过访问点,

1 图形方面将光盘文件放入虚拟光驱设备

2 命令行查看光驱设备

root@localhost ~]# ls -l /dev/cdrom

lrwxrwxrwx. 1 root root 3 12月 3 10:45 /dev/cdrom -> sr0

3 进行挂载,指定访问点(挂载点)

使用mount命令

格式: mount 设备路径 挂载点目录

[root@localhost ~]# mount /dev/cdrom /dvd

mount: /dev/sr0 写保护，将以只读方式挂载

4 卸载: umount

格式: umount 挂载点目录

**总结:**

1) Linux访问设备内容需要通过访问点

2) Linux访问点就是一个目录

3) 通过mount挂载命令让目录成为设备的访问点

**注意事项:**

1) 卸载: 当前路径不要是挂载点目录

[root@localhost dvd]# umount /dvd/

umount: /dvd：目标忙。

(有些情况下通过 lsof(8) 或 fuser(1) 可以

找到有关使用该设备的进程的有用信息)

2) Linux允许一个设备有多个挂载点,不允许一个目录有多个设备挂载

[root@localhost ~]# mount /dev/cdrom /test

mount: /dev/sr0 写保护，将以只读方式挂载

[root@localhost ~]# mount /dev/cdrom /test

mount: /dev/sr0 写保护，将以只读方式挂载

3) 建议不要挂载根目录下已存在的目录,最好是自己创建的目录

**目录和文件管理**

使用pwd和cd

pwd------Print Working Directory

用途：查看当前工作目录

cd -----change directory

用途: 切换工作路径

格式: cd [目标文件夹位置]

. 表示当前路径

.. 表示父目录

~ 代表用户家目录 (把存放用户个性化信息的目录)

cd ~用户名: 去往用户的家目录 # cd ~zhangsan

Linux管理员root家目录: /root

存放所有普通用户家目录: /home

[root@localhost ~]# cd ~lisi

[root@localhost lisi]# pwd

/home/lisi

ls 列出文档及属性

格式: ls [选项] ... [目录或文件名]

常用命令选项

-l : 以长格式显示,显示详细信息

-A : 包括名称以.开头的隐藏文档

-d : 显示目录本身(而不是内容)的属性

-h : 提供易读的容量单位(K、M等)

使用通配符: 针对不确定的文档名称,以特殊字符表示

\* : 任意多个任意字符

? : 单个字符

[a-z] : 多个字符或连续范围中的一个,若无则忽略(只能识别0-9)

{x,y,z} : 多组不同的字符串,全匹配

例子: 显示/dev/tty20至/dev/tty30

法1 : [root@localhost ~]# ls /dev/tty{2[0-9],30}

法2 : [root@localhost ~]# ls /dev/tty{2?,30}

别名的定义

查看已设置的别名: alias [别名名称]

定义新的别名: alias 别名名称=’实际执行的命令行’

取消已设置的别名: unalias [别名名称]

别名的优先级大于基本命令

创建目录

mkdir ----- mkdir directory

格式: mkdir [-p] [路径/]目录名...

[root@localhost ~]# mkdir -p /opt/a/b/c/d/e

ls -R

[root@localhost ~]# ls -R /opt

递归: 逐层展开显示

删除: rm (多参数命令)

格式: rm [选项] 文件或目录

常用命令选项

-r 、-f : 递归删除(含目录)、强制删除 (-f的优先级优于-i)

移动: mv (多参数命令)

格式: mv [选项]... 原文件... 目标路径

重命名: 路径不变的移动

复制: cp (-i的优先级优于-f)

格式: cp [选项]... 原文件... 目标路径

常用选项: -r,递归,复制目录时必须要有此选项

-p: 完全相同的复制

复制出现重名:

临时取消别名: \cp -r /boot/ /opt/

复制支持两个以上的参数,永远把最后一个参数作为目标,其他所有参数做为原文件;

复制与一个点(.)连用: 将原文件复制到当前路径下

复制时可以重新命名: 目标路径的新文档

[root@studnt ~]# rm -rf /opt/\*

[root@studnt ~]# cp -r /home/ /opt/test/

#将/home复制到/opt重新命名为test目录

[root@studnt ~]# cp -r /home/ /opt/test/

#将/home复制到/opt/test目录下

#########################################

**复习:**

1 Linux中命令的执行依赖于解释器,默认的解释器是哪个程序?

/bin/bash

2 如何结束当前正在运行的命令,快捷键是?

ctrl + c

3 切换上一级目录的操作命令? 去往用户root的家目录?

cd .. cd ~root或cd /root

4 ls常用选项有哪些? 不用写作用

-l -A -h -d

5 常见的通配符都有哪些? 不用写作用

\* ? {} []

6 如何定义一个别名? 如何查看所有的别名? 如何删除一个别名?

alias 别名=’实际执行的命令’ alias unalias 别名

7 移动的命令是?

mv

8 强制删除一个文件夹以及其内容的命令是?

rm -rf

9 cp命令在使用时,原是一个目录时需要加什么选项?

-r

10 vim编辑器有几种工作模式? 分别是什么?

三种: 命令模式, 插入模式, 末行模式

11 在vim末行模式中,输入什么可以不保存退出?

q!

12 在使用ls命令完成以下任务,找出/etc/目录下以re开头.conf结尾的文件

ls /etc/re\*.conf

#########################################

**I-Day04**

教学环境的介绍

每个学员机上有三台预先设置好的虚拟机

server---------作为练习用服务器

desktop-------作为练习用客户机

classroom------提供网关/DNS/软件素材等资源(优先重置开启)

Linux系统管理员:

要求具备从事Linux行业的初级/入门级技能

－侧重于单个服务器的配置和管理

－可以对Linux主机进行基础的管理工作

－比如创建用户、设置权限、管理磁盘、文档备份与恢复、管理系统任务、配置网络地址、安装软件包、访问其他服务等

Linux系统工程师:

要求具备从事Linux行业的中级技能

－侧重于多个服务器的应用部署及管理

－既能对Linux主机进行基础管理工作,还可以配置web、邮件、文件等服务器,并实现安全运行

－比如SELinux、防火墙、各种网站部署、网络磁盘、资源共享、多网卡聚合、用户环境定制、基础Shell脚本、基础数据库运维等

真机使用rht-vmctl辅助工具

控制教学用虚拟机

格式: rht-vmctl 控制指令 虚拟机名

常用控制指令: reset(还原) poweroff(强制断电) start(开机)

# rht-vmctl reset classroom # 先重置资源服务器

# rht-vmctl reset server # 再重置答题虚拟机

# rht-vmctl reset desktop

虚拟机server与虚拟机desktop的root密码为redhat

虚拟机server0:

1 查看系统版本: RHEL7.0

2 查看系统主机名: server0.example.com

3 查看系统eth0的IP地址: 172.25.0.11/24 子网划分:节约IPv4地址

虚拟机desktop0:

1 查看系统版本: RHEL7.0

2 查看系统主机名: desktop0.example.com

3 查看系统eth0的IP地址: 172.25.0.10/24

真机远程管理虚拟机:

1 测试真机与虚拟机是否可以通信

[root@room9pc01 ~]# ping 172.25.0.11

[root@room9pc01 ~]# ping 172.25.0.10

2 执行ssh进行远程管理虚拟机

格式: ssh root@对方的IP地址

# ssh root@172.25.0.11

# ssh root@172.25.0.10

3 -X(大写): 在远程管理时开启对方的图形程序

# ssh -X [root@172.25.0.10](mailto:root@172.25.0.10)

为真机设置永久的别名:

1 修改别名的配置文件(/root/.bashrc)----系统级配置文件

[root@room9pc01 ~]# vim /root/.bashrc

alias gos='ssh -X root@172.25.0.11'

alias god='ssh -X root@172.25.0.10'

注意: 开启新的终端才会生效

**软件包管理:**

一 零散软件包管理:

获得软件包

虚拟机classroom: 搭建web服务,提供RHEL7.0光盘所有软件包

http: 超文本传输协议

访问测试: firefox classroom.example.com

下载软件包:

使用wget下载工具: wget 软件包的URL网址 #默认下载到当前路径

-O(大写): 下载另存为

1 在真机的浏览器中访问classroom.example.com , 在Package中找到vsftpd包 (ctrl + f 查找vsftpd)

2 在虚拟机server0进行下载

[root@server0~]#wget <http://classroom.example.com/content/rhel7.0/x86_64/dvd/Packages/vsftpd-3.0.2-9.el7.x86_64.rpm>

[root@server0 ~]# ls | grep vsftpd

vsftpd-3.0.2-9.el7.x86\_64.rpm

使用rpm命令管理软件

RPM Package Manager ,RPM包管理器

rpm -q 软件名... #查询本机是否安装此软件

rpm -ivh 软件名-版本信息.rpm... #安装软件包

(ivh------install,显示详细信息,显示进度条)

rpm -e # 卸载程序

rpm -ql 软件名---------查看已安装软件的清单

[root@server0 ~]# rpm -ivh vsftpd-3.0.2-9.el7.x86\_64.rpm

[root@server0 ~]# rpm -q vsftpd

vsftpd-3.0.2-9.el7.x86\_64

了解内容: 导入红帽签名

[root@server0~]#wget <http://classroom.example.com/content/rhel7.0/x86_64/dvd/RPM-GPG-KEY-redhat-release>

[root@server0 ~]# ls | grep RPM

RPM-GPG-KEY-redhat-release

[root@server0 ~]# rpm --import RPM-GPG-KEY-redhat-release

这样再安装就不会有警告标志出来

**升级内核:**

1 下载新的内核软件包

[root@server0~]#wget http://classroom.example.com/content/rhel7.0/x86\_64/errata/Packages/kernel-3.10.0-123.1.2.el7.x86\_64.rpm

[root@server0 ~]# ls | grep kernel

kernel-3.10.0-123.1.2.el7.x86\_64.rpm

2 安装新的内核软件包

[root@server0 ~]# uname -r

3.10.0-123.el7.x86\_64

[root@server0 ~]# rpm -ivh kernel-3.10.0-123.1.2.el7.x86\_64.rpm

[root@server0 ~]# reboot # 重启系统刷新

Connection to 172.25.0.11 closed by remote host.

Connection to 172.25.0.11 closed.

[root@room9pc01 ~]# gos

Last login: Tue Dec 4 10:40:46 2018 from 172.25.0.250

[root@server0 ~]# uname -r

3.10.0-123.1.2.el7.x86\_64

常见安装软件包提示:

1 软件包已经安装

2 软件包的依赖关系

**yum软件仓库**, 自动解决依赖关系安装软件

服务: 为客户端自动解决依赖关系安装软件

服务端: 虚拟机classroom

1 众多的软件包

2 仓库数据文件(软件清单)

3 具备光盘所有的内容

4 搭建FTP服务或web服务,共享光盘所有内容

客户端: 虚拟机server

书写yum客户端配置文件, 指定服务端位置

配置文件要求: /etc/yum.repos.d/\*.repo ------------**记住**

在此目录下错误的文件会影响正确的文件

重点字段: [] name baseurl enabled gpgcheck ------**记住**

[root@server0 ~]# rm -rf /etc/yum.repos.d/\* #防止有其它错误文件而影响整个文件

[root@server0 ~]# vim /etc/yum.repos.d/dvd.repo

[rehl7] #仓库标识

name=RHEL7.0 #仓库描述信息

baseurl=http://classroom.example.com/content/rhel7.0/x86\_64/

dvd/ #指定服务端位置

enabled=1 #启用本文件

gpgcheck=0 # 不检测红帽认证

[root@server0 ~]# yum repolist #列出仓库信息

利用yum安装软件包

格式: yum [-y] install 软件名(软件包名)

[root@server0 ~]# yum install httpd

[root@server0 ~]# yum install sssd

[root@server0 ~]# yum -y install gcc

[root@server0 ~]# yum -y install system-config-kickstart

利用yum卸载软件包

yum [-y] remove 软件名 #建议卸载的时候不要’-y’

清除yum缓存

yum clean all

**配置网络**

一 配置永久的主机名

法1: 配置文件: /etc/hostname (默认不存在)

[root@server0 ~]# vim /etc/hostname

nsd1811.cn

[root@server0 ~]# hostname

nsd1811.cn

[root@server0 ~]# exit

登出

Connection to 172.25.0.11 closed.

[root@room9pc01 ~]# gos

Last login: Tue Dec 4 15:28:55 2018 from 172.25.0.250

[root@nsd1811 ~]#

法2: 命令行 格式: hostnamectl set-hostname 主机名

[root@server0 ~]# hostnamectl set-hostname haha

[root@server0 ~]# hostname

haha

[root@server0 ~]# exit

登出

Connection to 172.25.0.11 closed.

[root@room9pc01 ~]# gos

Last login: Sat Dec 15 16:21:05 2018 from 172.25.0.250

[root@haha ~]# cat /etc/hostname

haha

二 配置永久的IP地址、子网掩码、网关地址

网卡配置文件: /etc/sysconfig/network-scripts/ifcfg-eth0-----**记住**

法1 直接修改ifcfg-eth0配置文件

法2 利用命令方式配置



1) 查看命令识别的网卡名

[root@server0 ~]# nmcli connection show

1. 配置IP地址,子网掩码,网关地址

# nmcli connection 修改 ‘网卡名’ ipv4.方法 手工配置 ipv4.地址 ‘IP地址/子网掩码 网关地址’ 每次开机自动启用本网卡所有配置

[root@server0 ~]# nmcli connection modify 'System eth0' ipv4.method manual ipv4.addresses '172.25.0.110/24 172.25.0.254' connection.autoconnect yes

3) 激活配置

[root@server0 ~]# nmcli connection up ‘System eth0’

三 永久配置DNS服务器地址(在配置网络参数时,最后配置DNS服务器地址)

配置文件: /etc/resolv.conf-----**记住**

[root@server0 ~]# vim /etc/resolv.conf

nameserver 172.25.254.254

测试DNS解析

格式: nslookup 域名

[root@server0 ~]# nslookup server0.example.com

Server: 172.25.254.254

Address: 172.25.254.254#53

Name: server0.example.com

Address: 172.25.0.11

[root@server0 ~]# nslookup desktop0.example.com

#########################################

**I-Day05**

重定向: 将前面命令的输出,当做内容写入到文本文件中

> : 覆盖重定向

>> : 追加重定向

[root@server0 ~]# uname -r > /opt/1.txt

[root@server0 ~]# cat /opt/1.txt

3.10.0-123.el7.x86\_64

[root@server0 ~]# hostname > /opt/1.txt

[root@server0 ~]# cat /opt/1.txt

server0.example.com

[root@server0 ~]# uname -r >> /opt/1.txt

[root@server0 ~]# cat /opt/1.txt

server0.example.com

3.10.0-123.el7.x86\_64

echo (经常与重定向一起用,编写简单的配置文件或者shell脚本)

格式: echo 文本内容

[root@server0 ~]# echo 世界很美好,生活很快乐 > /opt/1.txt

[root@server0~]#echo nameserver 172.25.254.254 > /etc/resolv.conf

计算器: bc (没有四舍五入 5/2=2)

常见运算: 加减乘除(+ - \* /) 求模运算(取余数运算) %

quit: 退出

date : 查看日期时间

date +%Y : 查看年份

date +%m : 查看月份

date +%d : 查看几号

date +%Y%m%d : 查看年月日

date +%Y-%m-%d : 查看年月日

date +%F : 查看年月日

修改时间: -s “年-月-日 时:分:秒” (单引号和双引号都可以)

[root@server0 ~]# date +%Y

2018

[root@server0 ~]# date +%m

12

[root@server0 ~]# date +%d

05

[root@server0 ~]# date +%Y%m%d

20181205

[root@server0 ~]# date +%Y-%m-%d

2018-12-05

[root@server0 ~]# date +%F

2018-12-05

管道操作: | (shift + \) 可以连接无数条命令,但也有不支持的命令

作用: 将前面命令的输出结果,传递到后面命令,作为后面命令的参数

例子: 显示文件/etc/passwd 的8到12行并显示行号

[root@server0 ~]# head -12 /etc/passwd | tail -5 | cat -n

1 halt:x:7:0:halt:/sbin:/sbin/halt

2 mail:x:8:12:mail:/var/spool/mail:/sbin/nologin

3 operator:x:11:0:operator:/root:/sbin/nologin

4 games:x:12:100:games:/usr/games:/sbin/nologin

5 ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin

**管理用户和组**

唯一标识用户: UID , 从0开始的阿拉伯数字

管理员root的UID为: 0

Linux的用户至少属于一个组,一个用户可以加入不同的组

组: 方便对用户的管理,方便对权限管理

唯一标识组: GID , 从0开始的阿拉伯数字

组: 基本组 附加组(从属组)

基本组: 由系统创建,与用户同名,由系统将用户加入该组

附加组: 由管理员创建该组,随意命名,由管理员将用户加入该组

添加用户

使用useradd 命令

格式: useradd [选项]... 用户名

常用选项:

-u 用户id

-d 家目录路径

-s 登录shell(登录解释器程序)

-g 基本组

-G 附加组

用户基本信息放在/etc/passwd文件 ------系统级配置文件**(记住)**

[root@server0 ~]# head -1 /etc/passwd

root:x:0:0:root:/root:/bin/bash

用户名:密码占位符:UID:基本组的GID值:用户描述信息:用户的家目录:解释器

[root@server0 ~]# useradd nsd01

[root@server0 ~]# id nsd01 #查看用户基本信息

uid=1001(nsd01) gid=1001(nsd01) 组=1001(nsd01)

[root@server0 ~]# grep nsd /etc/passwd

nsd01:x:1001:1001::/home/nsd01:/bin/bash

[root@server0 ~]# useradd -u 1500 nsd02

[root@server0 ~]# id nsd02

uid=1500(nsd02) gid=1500(nsd02) 组=1500(nsd02)

[root@server0 ~]# useradd -d /mnt/nsd03 nsd03

[root@server0 ~]# grep nsd03 /etc/passwd

nsd03:x:1501:1501::/mnt/nsd03:/bin/bash

[root@server0 ~]# ls /mnt

nsd03

添加组 /sbin/nologin: 禁止用户登录系统

格式: groupadd 组名

[root@server0 ~]# groupadd tarena

[root@server0 ~]# useradd -G tarena nsd08

[root@server0 ~]# id nsd08

uid=1502(nsd08) gid=1503(nsd08) 组=1503(nsd08),1502(tarena)

[root@server0 ~]# useradd -s /sbin/nologin nsd09

[root@server0 ~]# grep nsd09 /etc/passwd

nsd09:x:1503:1504::/home/nsd09:/sbin/nologin

设置登录密码: 使用passwd命令

用户密码信息存放在/etc/shadow文件 ------**记住**

[root@server0 ~]# head -1 /etc/shadow

root:$6$UiGI4Tc2$htsXYn5cJnOqv3P1VLcUSgfjDu2pL5yiJBuua6foZAHdwqeuLHfYUfS/vBn27Wjvoel8EJgtdsMjyquqvKAmf1:16261:0:99999:7:::

用户名:密码加密字符串:上一次修改密码时间:密码最短使用期限:密码最长使用期限:提前七天警告:用户永远不过期(账户永远有效)

上一次修改密码的时间 : 1970-1-1~2018-12-1

管理员root用户修改密码: passwd 用户名

普通用户修改自己密码: passwd

[root@server0 ~]# passwd nsd01

更改用户 nsd01 的密码 。

新的 密码： #输入新的密码

无效的密码： 密码少于 8 个字符

重新输入新的 密码： #重新输入新密码

passwd：所有的身份验证令牌已经成功更新。

[root@server0 ~]# su - nsd01 #在命令行临时切换用户身份: su - 用户名

[nsd01@server0 ~]$ passwd

Changing password for user nsd01.

Changing password for nsd01.

(current) UNIX password: #输入旧密码

New password: #输入新密码

Retype new password: #重新输入新密码

passwd: all authentication tokens updated successfully.

非交互式设置密码(只适用于管理员)

echo ‘密码’ | passwd --stdin 用户名

[root@server0 ~]# echo 123456 | passwd --stdin nsd01

更改用户 nsd01 的密码 。

passwd：所有的身份验证令牌已经成功更新。

**修改用户属性** -----也可以通过vim /etc/passwd修改

使用usermod 命令

格式: usermod [选项] ... 用户名

常用选项

-u 用户id

-d 家目录路径

-s 登录shell

-g 基本组

-G 附加组

[root@server0 ~]# useradd nsd11

[root@server0 ~]# grep nsd11 /etc/passwd

nsd11:x:1504:1505::/home/nsd11:/bin/bash

[root@server0 ~]# usermod -u 1600 -d /opt/nsd11 -s /sbin/nologin -G tarena nsd11

[root@server0 ~]# grep nsd11 /etc/passwd

nsd11:x:1600:1505::/opt/nsd11:/sbin/nologin

[root@server0 ~]# id nsd11

uid=1600(nsd11) gid=1505(nsd11) 组=1505(nsd11),1502(tarena)

检查用户ID信息

使用id命令

格式: id 用户名

删除用户

使用userdel命令

格式: userdel [-r] 用户名

-r : 删除用户信息,并且删除用户的家目录 建议一般不要加’-r’

**管理组账号**

组基本信息存放在/etc/group文件-------**记住**

[root@server0 ~]# groupadd adminuser

[root@server0 ~]# grep adminuser /etc/group

adminuser:x:1506:

组名:组的密码占位符:组的GID:组的成员列表

添加组

使用groupadd命令

格式: groupadd [-g 组ID] 组名

[root@server0 ~]# useradd harry

[root@server0 ~]# useradd natasha

[root@server0 ~]# useradd kenji

[root@server0 ~]# useradd jack

管理组成员

组密码信息放在/etc/gshadow文件

使用gpasswd命令

格式: gpasswd -a 用户名 组名

gpasswd -d 用户名 组名

[root@server0 ~]# gpasswd -a kenji adminuser

正在将用户“kenji”加入到“adminuser”组中

[root@server0 ~]# grep adminuser /etc/group

adminuser:x:1506:kenji

[root@server0 ~]# id kenji

uid=1603(kenji) gid=1603(kenji) 组=1603(kenji),1506(adminuser)

[root@server0 ~]# gpasswd -a natasha adminuser

正在将用户“natasha”加入到“adminuser”组中

[root@server0 ~]# grep adminuser /etc/group

adminuser:x:1506:kenji,natasha

[root@server0 ~]# gpasswd -d kenji adminuser

正在将用户“kenji”从“adminuser”组中删除

[root@server0 ~]# grep adminuser /etc/group

adminuser:x:1506:natasha

[root@server0 ~]# gpasswd -a jack adminuser

正在将用户“jack”加入到“adminuser”组中

[root@server0 ~]# grep adminuser /etc/group

adminuser:x:1506:natasha,jack

[root@server0 ~]# gpasswd -d natasha adminuser

正在将用户“natasha”从“adminuser”组中删除

[root@server0 ~]# grep adminuser /etc/group

adminuser:x:1506:jack

修改组属性

使用groupmod命令

格式: groupmod [-g 组ID] [-n 新组名] 组名

删除组-------删除了组,但是组里的用户不会删除的; 基本组是删除不了的

使用groupdel 命令

格式: groupdel 组名

当Linux创建一个用户时 /etc/passwd /etc/shadow /etc/group /etc/gshadow四个文件内容都会改变

**tar备份与恢复**

压缩和归档

归档的含义: 将许多零散的文件整理为一个文件;文件总的大小基本不变

压缩的含义: 按某种算法减小文件所占用空间的大小;恢复时按对应的逆向算法解压

Linux独有的压缩格式:

gzip ------->.gz 速度快,但是效果差

bzip2 ------->.bz2 中庸

xz ------->.xz 速度慢,但效果好

制作压缩包tar命令格式:

tar 选项 压缩包的名字 被压缩的源文档(文件+目录)....

常用选项:

-c: 创建归档

-x: 释放归档

-f: 指定归档文件名称(指定tar包的名称),必须有且必须在所有选项的最后

-z、-j、-J : 调用.gz .bz2 .xz 格式的工具进行处理

-t: 显示归档中的文件清单

-C(大写): 指定释放的路径

-P(大写): 保持归档内文件的绝对路径

[root@server0 ~]# rm -rf /opt/\*

[root@server0 ~]# ls /opt

[root@server0 ~]# tar -zcf /opt/abc.tar.gz /etc/passwd /etc/shadow /etc/group /etc/gshadow

tar: 从成员名中删除开头的“/”

[root@server0 ~]# tar -Jcf /opt/file.tar.xz /etc/fstab /etc/hosts

tar: 从成员名中删除开头的“/”

[root@server0 ~]# tar -jcf /opt/test.tar.bz2 /home /etc/passwd

tar: 从成员名中删除开头的“/”

[root@server0 ~]# ls /opt

abc.tar.gz file.tar.xz test.tar.bz2

释放压缩包tar命令格式:

tar 选项 压缩包的名字 -C 释放的路径

[root@server0 ~]# rm -rf /mnt/\*

[root@server0 ~]# tar -xf /opt/abc.tar.gz -C /mnt

[root@server0 ~]# tar -xf /opt/file.tar.xz -C /mnt

[root@server0 ~]# tar -xf /opt/test.tar.bz2 -C /mnt

[root@server0 ~]# ls /mnt/

etc home

显示tar包数据内容

[root@server0 ~]# tar -tf /opt/file.tar.xz

etc/fstab

etc/hosts

[root@server0 ~]# tar -tf /opt/abc.tar.gz

etc/passwd

etc/shadow

etc/group

etc/gshadow

[root@server0 ~]# tar -tf /opt/test.tar.bz2

**NTP时间同步**-----Network Time Protocol

NTP服务器为客户机提供标准时间

NTP客户机需要与NTP服务器保持沟通

NTP服务器: 虚拟机classroom

NTP客户机: 虚拟机server

1 安装chrony软件与NTP服务器沟通

[root@server0 ~]# rpm -q chrony #默认已经安装

chrony-1.29.1-1.el7.x86\_64

2 修改主配置文件/etc/chrony.conf , 指定NTP服务器位置

以#开头的为注释行

[root@server0 ~]# vim /etc/chrony.conf

server classroom.example.com iburst #指定服务端位置

3 重启程序(服务),刷新配置 d:daemon 守护神

[root@server0 ~]# systemctl restart chronyd #重启服务

[root@server0 ~]# systemctl enable chronyd #设置服务,开机自启

4 验证时间同步

[root@server0 ~]# date

2018年 12月 05日 星期三 17:14:39 CST

[root@server0 ~]# date -s '2008-12-5 12:12:12'

2008年 12月 05日 星期五 12:12:12 CST

[root@server0 ~]# systemctl restart chronyd

[root@server0 ~]# date

2018年 12月 05日 星期三 17:16:27 CST

[root@server0 ~]# timedatectl #查看开机是否自启

[root@server0 ~]# timedatectl set-ntp true #设置开机自启

[root@server0 ~]# timedatectl set-ntp false #设置开机不自启

#########################################

**复习:**

1 创建用户的命令是什么? 常用的选项都有哪些?

useradd -G -d -u -s

2 非交互式lisi设置密码123,请写出该命令

echo 123 | passwd --stdin lisi

3 请写出存放用户基本信息的配置文件

/etc/passwd

4 请写出禁止用户登录系统的解释器程序

/sbin/nologin

5 将harry用户加入tedu组,请写出该命令

gpasswd -a harry tedu

6 请写出组账号基本信息配置文件

/etc/group

7 将/usr/local进行归档及压缩为/root/file.tar.gz请写出该命令

tar -czf /root/file.tar.gz /usr/local

8 将/root/file.tar.gz进行解压缩到/opt目录下,写出该命令

tar -xf /root/file.tar.gz -C /opt

9 请写出重启chronyd服务的命令

systemctl restart chronyd

10 请写出设置chronyd服务为开机自启的命令

systemctl enable chronyd

11 指定DNS服务器地址,配置文件路径

/etc/resolv.conf

12 yum客户端配置文件放在什么路径下? 具体都有哪些字段?

/etc/yum.repos.d/\*.repo

[] name baseurl enabled gpgcheck

13 设置永久别名配置文件,请写出该文件路径?

/root/.bashrc

#########################################

**I-Day06**

**权限和归属**

**基本权限**

基本权限的类别:

访问方式(权限)

读取:允许查看内容-----read r

写入:允许修改内容----write w

可执行:允许运行和切换 -----execute x

对于文本文件

r权限: cat less tail head 查看内容

w权限 : vim > >>

x权限 : shell脚本

对于目录

r权限: 能够ls浏览此目录内容

w权限: 能够执行rm/mv/cp/mkdir/touch/等更改目录内容的操作,对

目录本身不具备修改的权限

x权限: 能够cd切换到此目录

权限适用对象(归属)

所有者: 拥有此文件/目录的用户 -user u

所属组: 拥有此文件/目录的组 -group g

其他用户: 除所有者、所属组以外的用户 -other o



查看权限

使用ls -l 命令

格式: ls -ld 文件或目录

[root@server0 ~]# ls -l /etc/passwd

代表权限,以rwx排序,没有的就是-,分别是所有者,所属组,其他用户权限

所有者 所属组

-rw-r--r--. 1 root root 2005 7月 11 2014 /etc/passwd

[root@server0 ~]# ls -ld /etc

drwxr-xr-x. 134 root root 8192 12月 6 08:38 /etc

[root@server0 ~]# ls -l /etc/rc.local

lrwxrwxrwx. 1 root root 13 5月 7 2014 /etc/rc.local -> rc.d/rc.local

[root@server0 ~]# ls -l /etc/shadow

----------. 1 root root 1148 7月 11 2014 /etc/shadow

[root@server0 ~]# ls -ld /tmp

drwxrwxrwt. 14 root root 4096 12月 6 10:06 /tmp

比较特殊,需要记住

以 - 开头: 文本文件

以d开头: 目录

以l开头: 快捷方式

设置基本权限

使用chmod命令

格式: chmod [-R] 归属关系+-=权限类别 文档... 立即生效

[-R] : 递归修改权限

[root@server0 ~]# mkdir /nsd01

[root@server0 ~]# ls -ld /nsd01

drwxr-xr-x. 2 root root 6 12月 6 10:14 /nsd01

[root@server0 ~]# chmod u-w /nsd01

[root@server0 ~]# ls -ld /nsd01

dr-xr-xr-x. 2 root root 6 12月 6 10:14 /nsd01

[root@server0 ~]# chmod o=--- /nsd01

[root@server0 ~]# ls -ld /nsd01

dr-xr-x---. 2 root root 6 12月 6 10:14 /nsd01

[root@server0 ~]# chmod u=rwx,g=rx,o=rx /nsd01

[root@server0 ~]# ls -ld /nsd01

drwxr-xr-x. 2 root root 6 12月 6 10:14 /nsd01

[root@server0 ~]# chmod ugo=rwx /nsd01

[root@server0 ~]# ls -ld /nsd01

drwxrwxrwx. 2 root root 6 12月 6 10:14 /nsd01

Linux如何判断用户具备的权限

1 判断用户对于文档属于什么身份 所有者>所属组>其他人 (匹配及停止)

2 查看文档相应角色的权限 权限不足: permission denied

1 王老师可以进 1 所有人不可以进

2 所有人不可以进 2 王老师可以进

(王老师可以进) (王老师不可以进)

**练习:** 以root用户新建/nsddir/目录,在此目录下新建readme.txt文件,并进一步完成下列操作

1. 使用户lisi能够在此目录下创建子目录

chmod o+w /nsddir/

1. 使用户lisi不能够在此目录下创建子目录

chmod o-w /nsddir/

1. 使用户lisi能够修改readme.txt文件内容

chmod o+w /nsddir/readme.txt

1. 调整此目录的权限,使所有用户都不能cd进入此目录

chmod u-x,g-x,o-x /nsddir

1. 为此目录及其下所有文档设置权限rwxr-x---

chmod -R u=rwx,g=rx,0=--- /nsddir

**设置文档归属**

使用chown命令

chown [-R] 属主 文档...

chown [-R] :属组 文档...

chown [-R] 属主:属组 文档...

-R: 递归修改

[root@server0 ~]# mkdir /nsd03

[root@server0 ~]# ls -ld /nsd03

drwxr-xr-x. 2 root root 6 12月 6 11:30 /nsd03

[root@server0 ~]# groupadd tarena

[root@server0 ~]# chown lisi:tarena /nsd03

[root@server0 ~]# ls -ld /nsd03

drwxr-xr-x. 2 lisi tarena 6 12月 6 11:30 /nsd03

[root@server0 ~]# useradd zhangsan

[root@server0 ~]# chown zhangsan /nsd03 #修改所有者

[root@server0 ~]# ls -ld /nsd03

drwxr-xr-x. 2 zhangsan tarena 6 12月 6 11:30 /nsd03

[root@server0 ~]# chown :root /nsd03 #修改所属组

[root@server0 ~]# ls -ld /nsd03

drwxr-xr-x. 2 zhangsan root 6 12月 6 11:30 /nsd03

[root@server0 ~]# chown root /nsd03

[root@server0 ~]# ls -ld /nsd03

drwxr-xr-x. 2 root root 6 12月 6 11:30 /nsd03

**练习:** 利用root用户新建/nsd06目录，并进一步完成下列操作

1）将属主设为gelin01，属组设为tarena组

[root@server0 /]# useradd gelin01

[root@server0 /]# useradd gelin02

[root@server0 /]# groupadd tarena

[root@server0 /]# chown gelin01:tarena /nsd06

2）使用户gelin01对此目录具有rwx权限; 除属主与属组之外的人，对此目录无任何权限

[root@server0 /]# chmod o=--- /nsd06

3）使用户gelin02能进入、查看此目录

[root@server0 /]# gpasswd -a gelin02 tarena

4）将gelin01加入tarena组, 将nsd06目录的权限设为rw-r-x--- ; 再测试gelin01用户能否进入此目录

[root@server0 /]# chmod u=rw,g=rx /nsd06

[root@server0 ~]# su - gelin01

[gelin01@server0 ~]$ cd /nsd06

-bash: cd: /nsd06: Permission denied #匹配及停止

**附加权限**(特殊权限)

Set GID

1) 附加在所属组的x 位上

2) 属组的权限标识会变为 s(S)

3) 适用于目录,Set GID可以使目录下新增的文档自动设置与父目录相同的属组,自动继承所属组身份

4) 传递所属组身份(让子文档自动继承父目录的所属组身份)

设置附加权限

使用chmod 命令

格式: chmod g+s 文档...

[root@server0 ~]# mkdir /nsd07

[root@server0 ~]# chown :tarena /nsd07 #修改所属组身份

[root@server0 ~]# ls -ld /nsd07

drwxr-xr-x. 2 root tarena 6 12月 6 14:18 /nsd07

[root@server0 ~]# mkdir /nsd07/abc01

[root@server0 ~]# ls -ld /nsd07/abc01

drwxr-xr-x. 2 root root 6 12月 6 14:19 /nsd07/abc01

[root@server0 ~]# chmod g+s /nsd07/ #设置Set GID权限

[root@server0 ~]# ls -ld /nsd07/

drwxr-sr-x. 3 root tarena 18 12月 6 14:19 /nsd07/

[root@server0 ~]# mkdir /nsd07/abc02

[root@server0 ~]# ls -ld /nsd07/abc02

drwxr-sr-x. 2 root tarena 6 12月 6 14:20 /nsd07/abc02

[root@server0 ~]# mkdir /nsd07/abc02/test

[root@server0 ~]# ls -ld /nsd07/abc02/test

drwxr-sr-x. 2 root tarena 6 12月 6 14:20 /nsd07/abc02/test

**acl访问控制列表** 立即生效

acl策略的作用:

文档归属的局限性

任何人只属于三种角色:属主,属组,其他人

无法实现更精确的控制

acl访问策略

能够对个别用户,个别组设置独立的权限

大多数挂载的EXT3/4、XFS文件系统默认已支持

使用getfacl setfacl 命令

# getfacl 文档... # 查看acl策略

# setfacl [-R] -m u:用户名:权限类别 文档...

# setfacl [-R] -m g:组名:权限类别 文档...

# setfacl [-R] -x u:用户名 文档... # 删除指定acl

# setfacl [-R] -b 文档... # 删除所有acl策略

-R: 递归

ACL设置黑名单 (实际生产中基本不会用)

[root@server0 /]# mkdir /public

[root@server0 /]# chmod ugo=rwx /public

[root@server0 /]# ls -ld /public

drwxrwxrwx. 2 root root 6 12月 6 15:51 /public

[root@server0 /]# setfacl -m u:lisi:--- /public

[root@server0 /]# getfacl /public

**总结**: 请实现lisi用户可以查看/etc/shadow文件内容,您有几种方法?

四种:

1 利用其他人: chmod o+r /etc/shadow

2 利用所属组,将lisi加入文件所属组中: gpasswd -a lisi 所属组

chmod g+r /etc/shadow

3 利用所有者,将lisi设置为文件的所有者: chown lisi /etc/shadow

chmod u+r /etc/shadow

4 利用ACL: setfacl -m u:lisi:r /etc/shadow

RHCSA(红帽认证的管理员)

RHCE(红帽认证的工程师)

RHCA(红帽认证的架构师)

**使用LDAP认证**

LDAP:Lightweight Directory Access Protocol-----轻量级目录访问协议

由服务器来集中存储并向客户端提供信息,存储方式类似于DNS分成结构;

提供的信息包括: 用户名、密码、通信录、主机名映射记录......

用户的集中管理

LDAP用户: 网络用户,由网络中LDAP服务器统一提供用户信息

LDAP服务器: 虚拟机classroom.example.com

LDAP客户端: 虚拟机server

1 安装软件sssd , 客户端与LDAP服务器沟通

[root@server0 /]# yum -y install sssd

2 图形工具authconfig-gtk进行配置

[root@server0 /]# yum -y install authconfig-gtk

[root@server0 /]# authconfig-gtk

选择LDAP

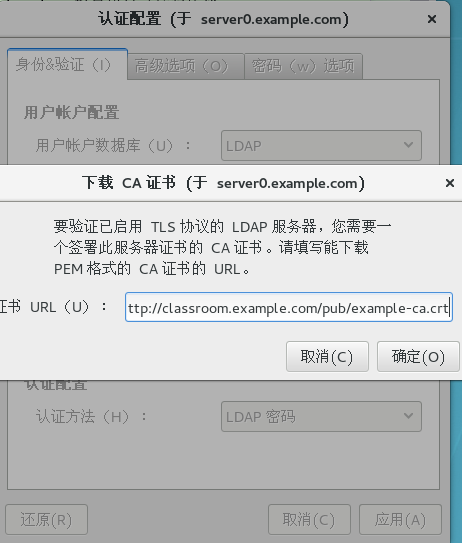
dc=example,dc=com #指定服务端域名

classroom.example.com #指定服务器主机名

勾选TLS加密

使用证书加密: <http://classroom.example.com/pub/example-ca.crt>

选择LDAP密码



3 重启sssd服务

[root@server0 /]# systemctl restart sssd #重启服务

[root@server0 /]# systemctl enable sssd #设置开机自启服务

4 验证

[root@server0 /]# grep ldapuser /etc/passwd

[root@server0 /]# id ldapuser0

uid=1700(ldapuser0) gid=1700(ldapuser0) 组=1700(ldapuser0)

[root@server0 /]# su - ldapuser0 #引出家目录漫游

su: 警告：无法更改到 /home/guests/ldapuser0 目录: 没有那个文件或目录

mkdir: cannot create directory '/home/guests': Permission denied

-bash-4.2$

**家目录漫游**

将classroom上用户的家目录, 出现在虚拟机server

NFS共享: Network File System,网络文件系统

1) 由NFS服务器将指定的文件夹共享给客户机

2) 客户机将此共享目录mount到本地目录,访问此共享资源就像访问本地目

录一样方便;

3) 类似于EXT4、XFS等类型,只不过资源在网上

NFS服务器: 虚拟机classroom

客户端: 虚拟机server 访问共享

查看NFS资源

格式: showmount -e [服务器地址]

[root@server0 /]# showmount -e classroom.example.com

Export list for classroom.example.com:

/home/guests 172.25.0.0/255.255.0.0

挂载NFS共享目录

mount 服务器地址:文件夹路径 挂载点

[root@server0 /]# mkdir /mydir

[root@server0 /]# mount classroom.example.com:/home/guests /mydir

[root@server0 /]# su - ldapuser0

上一次登录：四 12月 6 18:00:27 CST 2018pts/0 上

su: 警告：无法更改到 /home/guests/ldapuser0 目录: 没有那个文件或目录

mkdir: cannot create directory '/home/guests': Permission denied

-bash-4.2

[root@server0 /]# mkdir /home/guests

[root@server0 /]# mount classroom.example.com:/home/guests /home/guests

[root@server0 /]# su - ldapuser0

上一次登录：四 12月 6 18:03:06 CST 2018pts/0 上

#####################################

**复习:**

1 Linux中权限分类,主要分为哪几种?

三种 基本权限 附加权限 ACL访问控制列表

2 基本权限的分类? 归属关系分为哪几种?

r w x 所有者 所属组 其他人

3 Linux中如何查看一个文件或目录的权限

ls -l 文件 ls -ld 目录

4 修改权限的命令是什么? 修改归属关系的命令是什么?

chmod chown

5 一位用户能否切换到一个目录里面,只和什么权限有关?

x 执行权限

6 一位用户对目录具备w权限,该用户对目录进行哪些操作?

对目录的内容可以修改/删除/创建等操作

7 Linux中SGID权限用什么符号表示? 出现的位置? 有什么作用?

s(S) 出现在所属组的x位上 继承父目录的所属组身份(传递所属组身份)

8 ACL权限的作用?

能够对个别用户,个别组设置独立的权限

9 设置ACL权限的命令是什么? 常用的选项有哪些

setfacl -m -x -b -R

10 查看ACL权限的命令?

getfacl 文档

11 查看/boot文件夹的全部内容,以长格式显示、带易读单位, 请写出该命令

ls -lhA /boot

12 为lisi设置权限,可以读取/etc/shadow文件内容,你有几种办法,请写出

四种:

1 利用其他人: chmod o+r /etc/shadow

2 利用所属组,将lisi加入文件所属组中: gpasswd -a lisi 所属组

chmod g+r /etc/shadow

3 利用所有者,将lisi设置为文件的所有者: chown lisi /etc/shadow

chmod u+r /etc/shadow

4 利用ACL: setfacl -m u:lisi:r /etc/shadow

#########################################

**I-Day07**

**附加权限(续)**

Set UID

附加在属主的x位上

1) 属主的权限标识会变成s(S)

2) 适用于可执行文件,Set UID可以让使用者具有文件属主的身份及部分权限

3) 传递所有者身份

使用chmod 命令

格式: chmod u+s 文档...

[root@server0 ~]# cp /usr/bin/mkdir /usr/bin/hahadir

[root@server0 ~]# /usr/bin/mkdir /opt/nsd02

[root@server0 ~]# ls /opt/

nsd02 rh

[root@server0 ~]# /usr/bin/hahadir /opt/nsd03

[root@server0 ~]# ls /opt

nsd02 nsd03 rh

[root@server0 ~]# chmod u+s /usr/bin/hahadir

[root@server0 ~]# ls -l /usr/bin/hahadir

-rwsr-xr-x. 1 root root 79712 12月 7 09:44 /usr/bin/hahadir

[root@server0 ~]# su - student

[student@server0 ~]$ /usr/bin/mkdir test01

[student@server0 ~]$ ls -l

total 0

drwxrwxr-x. 2 student student 6 Dec 7 09:47 test01

[student@server0 ~]$ /usr/bin/hahadir test02 #相当于是root在创建

[student@server0 ~]$ ls -l

total 0

drwxrwxr-x. 2 student student 6 Dec 7 09:47 test01

drwxrwxr-x. 2 root student 6 Dec 7 09:48 test02

[student@server0 ~]$ /usr/bin/hahadir /opt/a.txt #相当于是root在创建

[student@server0 ~]$ ls /opt/

a.txt nsd02 nsd03 rh

[student@server0 ~]$ /usr/bin/mkdir /opt/b.txt

/usr/bin/mkdir: cannot create directory ‘/opt/b.txt’: Permission denied

Sticky Bit

附加在其他人的x位上

其他人的权限标识会变为t

适用于开放w权限的目录,可以阻止用户滥用w写入权限(禁止操作别人的

文档)

使用chmod 命令

格式: chmod o+t 文档...

[root@server0 ~]# useradd zhangsan

[root@server0 ~]# useradd lisi

[root@server0 ~]# useradd dc

[root@server0 ~]# mkdir /home/public

[root@server0 ~]# chmod ugo=rwx /home/public

[root@server0 ~]# su - lisi

[lisi@server0 ~]$ cd /home/public

[lisi@server0 public]$ touch lisi.txt

[root@server0 ~]# su - zhangsan

[zhangsan@server0 ~]$ cd /home/public/

[zhangsan@server0 public]$ touch zhangsan.txt

[root@server0 ~]# su - dc

[dc@server0 ~]$ cd /home/public/

[dc@server0 public]$ touch dc.txt

[dc@server0 public]$ ls

dc.txt lisi.txt zhangsan.txt

[root@server0 ~]# chmod o+t /home/public

[root@server0 ~]# ls -ld /home/public

drwxrwxrwt. 2 root root 38 12月 7 10:25 /home/public

[lisi@server0 public]$ rm -rf dc.txt

rm: cannot remove ‘dc.txt’: Operation not permitted

**cron计划任务**

用途: 按照设置的时间间隔为用户反复执行某一项固定的系统任务

软件包: cronie、 crontabs #默认已经安装

系统服务: crond #默认随机自启

日志文件: /var/log/cron

如何编写crontab任务记录

分 时 日 月 周 任务命令行(命令写绝对路径)

|  |  |
| --- | --- |
| 执行周期 | 配置说明 |
| 分钟 | 从0到59之间的整数 |
| 小时 | 从0到23之间的整数 |
| 日期 | 从0到31之间的整数 |
| 月份 | 从1到12之间的整数 |
| 星期 | 0~7之间的整数,0或7代表星期日 |

\* : 匹配范围内任意时间

, : 分隔多个不连续的时间点

- : 指定连续时间范围

/n : 指定时间频率,每n ..

30 8 \* \* \*

30 \*/2 \* \* 7

0 8 \* \* 1-5

30 8 1 \* 3 : 每月的一号和每周三,都会执行(“或”关系)

0 8 1 \* 1 #每月1号会被执行,每周一也会执行

0 \*/2 \* \* \* #每2个小时执行一次

使用crontab命令

编辑: crontab -e [-u 用户名]

查看: crontab -l [-u 用户名]

清除: crontab -r [-u 用户名] #不常用,会把用户创建的所有crontab计划任务全删掉

配置文件: /var/spool/cron------->vim /var/spool/cron/用户名

例子: 每分钟记录当前的系统时间,并写入到/opt/time.txt

[root@server0 ~]# crontab -e -u root

\* \* \* \* \* date >> /opt/time.txt

[root@server0 ~]# cat /var/spool/cron/root #生成任务文件

\* \* \* \* \* date >> /opt/time.txt

[root@server0 ~]# ls /var/spool/cron/

root

[root@server0 ~]# cat /opt/time.txt

Fri Dec 7 11:26:01 CST 2018

[root@server0 ~]# cat /opt/time.txt

Fri Dec 7 11:26:01 CST 2018

Fri Dec 7 11:27:01 CST 2018

**文本/文件查找**

**查找文本内容**

根据字符串模式提取文本行

1) grep [选项] ‘匹配模式’ 文本文件...

2) 命令行 | grep [选项] ‘匹配模式’

常用命令选项

-i : 忽略大小写

-v : 取反匹配

常用的匹配模式:

word: 包含字符串word

^word: 以字符串word开头

word$: 以字符串word结尾

^$: 表示空行

显示文件有效信息(去除以#号开头的注释,去除空行)

[root@server0 ~]# grep -v ^# /etc/login.defs | grep -v ^$

[root@server0 ~]# grep -v ^$ /etc/login.defs | grep -v ^#

例子: 将/etc/login.defs文件有效信息,写入到/opt/log.txt

[root@server0 ~]# grep -v ^# /etc/login.defs | grep -v ^$ > /opt/log.txt

[root@server0 ~]# cat /opt/log.txt

**查找文件(**find只能查硬盘上的数据,内存上的数据查找不了**)**

根据预设的条件递归查找相对应的文件

格式: find [目录] [条件1] [-a|-o] [条件2]......

a: and o: or 不写默认是-a

常用条件表示:

-type 类型 (f文本文件、d目录、 l快捷方式)

-name “文档名称” 文档名称用双引号标起来

-size +|-文件大小(k小写 M大写 G大写) +(大于) -(小于)

-user 用户名

[root@server0 ~]# find /boot/ -type l # 查找是快捷方式

/boot/grub/menu.lst

[root@server0 ~]# ls -l /boot/grub/menu.lst #验证

lrwxrwxrwx. 1 root root 9 5月 7 2014 /boot/grub/menu.lst -> grub.conf

[root@server0 ~]# find /boot/ -type d #查找是目录

[root@server0 ~]# find /boot/ -type f #查找是文本文件

[root@server0 ~]# find /boot/ -name "vm\*"

[root@server0 ~]# find /etc/ -name "passwd"

[root@server0 ~]# mkdir /root/nsd01

[root@server0 ~]# mkdir /root/nsd02

[root@server0 ~]# touch /root/nsd1811.txt

[root@server0 ~]# find /root/ -name "nsd\*"

/root/nsd01

/root/nsd02

/root/nsd1811.txt

[root@server0 ~]# find /root/ -name "nsd\*" -type f

/root/nsd1811.txt

[root@server0 ~]# find /root/ -name "nsd\*" -type d

/root/nsd01

/root/nsd02

[root@server0 ~]# find /root/ -name "nsd\*" -o -type d

[root@server0 ~]# find /boot/ -size -10M #查找小于10M

[root@server0 ~]# find /boot/ -size +300k #查找大于300k

[root@server0 ~]# find / -user student #查找文档的所有者为student用户

根据名称查找,忽略大小写

-iname

[root@server0 ~]# find /etc/ -iname "PAsswd"

[root@server0 ~]# find /etc/ -iname "passwd"

根据所属组

-group

[root@server0 ~]# find / -group student

限制目录查找的深度(最大层数)

-maxdepth

[root@server0 ~]# find /etc/ -maxdepth 1 -name "\*.conf"

[root@server0 ~]# find /etc/ -maxdepth 3 -name "\*.conf"

根据文件修改时间

-mtime 所有的时间都是过去时间

+10 : 查找10天之前创建或修改的文档

-10 : 查找最近10天之内创建或修改的文档

[root@server0 ~]# find /var/log/ -mtime +90

[root@server0 ~]# find /var/log/ -mtime -2

使用find命令的-exec操作

格式: find .. .. -exec 处理命令{} \;

优势: 以{}代替每一个结果,逐个处理,遇’\;’结束

案例:使用find命令完成以下任务

1、找出所有用户student拥有的文件

2、把他们拷贝到/root/findfiles/文件夹中

[root@server0 ~]# mkdir /root/findfiles

[root@server0 ~]# find / -user student -type f -exec cp {} /root/findfiles \;

[root@server0 ~]# ls -A /root/findfiles

authorized\_keys .bash\_profile lastnotification

.bash\_history .bashrc monitors.xml

.bash\_logout gnome-initial-setup-done student

########################################

**I-Day08**

**分区规划及使用**

一个扇区默认大小512字节

一块硬盘的”艺术”之旅

识别硬盘----->分区规划----->格式化----->挂载使用

毛坯楼层----->打隔断------->装修-------->入驻

1. 识别硬盘 服务器转速: 15000转

vd : 虚拟化硬盘

[root@server0 ~]# lsblk

NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT

vda 253:0 0 10G 0 disk

└─vda1 253:1 0 10G 0 part /

vdb 253:16 0 10G 0 disk

1. 分区规划

分区模式: MBR GPT(后面学)

MBR/msdos分区模式: 主引导记录模式;

分区类型: 主分区、扩展分区、逻辑分区

1) 1~4个主分区, 或者3个主分区+1个扩展分区(n个逻辑分区)

2) 最大支持容量为2.2TB的磁盘

3) 扩展分区不能格式化,不能存储数据

使用交互式fdisk分区工具

查看分区表

格式: fdisk -l /dev/vdb

修改硬盘分区表

格式: fdisk 硬盘设备

常用交互指令:

m: 列出指令帮助

p: 查看现有的分区表

n: 新建分区

d: 删除分区

q: 放弃更改并退出

w: 保存更改并退出

[root@server0 ~]# fdisk /dev/vdb

n 创建新的分区--->回车--->回车--->回车--->在last结束时 +2G

p 查看分区表

n 创建新的分区--->回车--->回车--->回车--->在last结束时 +1G

d 删除分区

w 保存并退出

[root@server0 ~]# ls /dev/vdb[1-2]

/dev/vdb1 /dev/vdb2

1. 格式化文件系统(存储数据的规则)

mkfs工具集

mkfs.ext3 分区设备路径

mkfs.ext4 分区设备路径

mkfs.xfs 分区设备路径

mkfs.vfat -F 32 分区设备路径

选项 -f : 强制格式化

[root@server0 ~]# mkfs.ext4 /dev/vdb1 #格式化文件系统为ext4

[root@server0 ~]# mkfs.xfs /dev/vdb2

[root@server0 ~]# blkid /dev/vdb2 #查看文件系统类型

/dev/vdb2: UUID="bb648c3e-48c4-4826-8cb2-00b29a6ec9e5" TYPE="xfs"

[root@server0 ~]# blkid /dev/vdb1

/dev/vdb1: UUID="faafe433-5da8-4ed0-9ee3-051fb9301de0" TYPE="ext4"

1. 挂载使用

[root@server0 ~]# mkdir /mypart1 /mypart2

[root@server0 ~]# mount /dev/vdb1 /mypart1

[root@server0 ~]# mount /dev/vdb2 /mypart2

[root@server0 ~]# df -h | tail -2 #显示正在挂载设备的使用情况

/dev/vdb1 2.0G 6.0M 1.8G 1% /mypart1

/dev/vdb2 1014M 33M 982M 4% /mypart2

五、开机自动挂载

配置文件: /etc/fstab 的记录格式 -------系统级配置文件

设备路径 挂载点 类型 参数 备份标记 检测顺序 -------**记住**

补充vim技巧: 命令模式进入插入模式, 按o可以另起一行进入插入模式

[root@server0 ~]# vim /etc/fstab

/dev/vdb1 /mypart1 ext4 defaults 0 0

/dev/vdb2 /mypart2 xfs defaults 0 0

[root@server0 ~]# umount /mypart1 /mypart2

[root@server0 ~]# mount -a

检测/etc/fstab开机自动挂载配置文件,格式是否正确

检测/etc/fstab中,书写完成,但没有挂载的设备,进行挂载

[root@server0 ~]# df -h | tail -2

/dev/vdb1 2.0G 6.0M 1.8G 1% /mypart1

/dev/vdb2 1014M 33M 982M 4% /mypart2

1. 综合分区

最终有3个主分区,分别为2G、1G、2G

创建扩展分区---->两个逻辑分区,分别为1G、1G

[root@server0 ~]# fdisk /dev/vdb

p 查看分区表

n 创建主分区--->回车--->回车--->回车---->在last结束时 +2G

p 查看分区表

n 创建扩展分区

--->回车--->起始回车--->结束回车 将所有空间给扩展分区

p 查看分区表

n 创建逻辑分区--->起始回车--->结束 +1G

n 创建逻辑分区--->起始回车--->结束 +1G

p 查看分区表

w 保存并退出

[root@server0 ~]# partprobe #刷新分区表信息,识别新的分区

[root@server0 ~]# lsblk | tail -6

**总结:**

1 查看本机的硬盘设备 lsblk

2 划分新的分区 fdisk

3 刷新新的分区表 partprobe

4 格式化 mkfs.ext4 mkfs.xfs blkid

5 挂载使用

手工挂载: mount

开机自动挂载 /etc/fstab

设备/dev/sda5标示含义?

第一块SCSI设备,第五个分区=第一块SCSI设备,第1个逻辑分区

新添加一块80G分区

1 关闭虚拟机server

[root@server0 ~]# poweroff

Connection to 172.25.0.11 closed by remote host.

Connection to 172.25.0.11 closed.

2 利用图形添加80G硬盘

[root@room9pc01 ~]# gos

[root@server0 ~]# lsblk | tail -1

vdc 253:32 0 80G 0 disk

3 综合分区:

划分3个主分区,分别为10G

划分扩展分区,在扩展分区之上,划分2个10G的逻辑分区

[root@server0 ~]# fdisk /dev/vdc

[root@server0 ~]# ls /dev/vdc[1-6]

/dev/vdc1 /dev/vdc2 /dev/vdc3 /dev/vdc4 /dev/vdc5 /dev/vdc6

**LVM逻辑卷**

作用: 1 整合分散的空间; 2 空间可以按照需求扩大

ps: 没有内容的分散分区才可以整合,有存储数据的分散分区不能整合

LVM工作方式

在”分区---->格式化”中间增加的一个逻辑层

零散空闲存储----------------------整合的虚拟磁盘------------------虚拟的分区

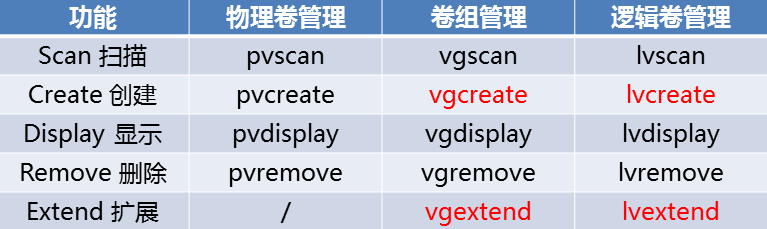
逻辑卷Logical Volume

卷组Volume Group

物理卷 Physical Volume

将众多的物理卷(PV),组建成卷组(VG) ,从卷组中划分出逻辑卷(LV)

LVM管理工具集



1. **创建逻辑卷**

1 创建卷组

格式: vgcreate 卷组名 空闲分区...

[root@server0 ~]# vgcreate systemvg /dev/vdc[1-2]

Physical volume "/dev/vdc1" successfully created

Physical volume "/dev/vdc2" successfully created

Volume group "systemvg" successfully created

[root@server0 ~]# pvs #查看物理卷信息

[root@server0 ~]# vgs #查看卷组信息

2 创建逻辑卷

格式: lvcreate -L 大小 -n 逻辑卷名称 基于卷组名

[root@server0 ~]# lvcreate -L 16G -n vo systemvg

Logical volume "vo" created

[root@server0 ~]# lvs #查看逻辑卷

3 使用逻辑卷

[root@server0 ~]# ls /dev/systemvg/vo

/dev/systemvg/vo

[root@server0 ~]# ls -l /dev/systemvg/vo

lrwxrwxrwx. 1 root root 7 12月 8 15:44 /dev/systemvg/vo -> ../dm-0

[root@server0 ~]# mkfs.xfs /dev/systemvg/vo

[root@server0 ~]# vim etc/fstab

/dev/systemvg/vo /mylv xfs defaults 0 0

后期会讲，为啥两者不一样

[root@server0 ~]# mount -a

[root@server0 ~]# mkdir /mylv

[root@server0 ~]# df -h | tail -1

/dev/mapper/systemvg-vo 16G 33M 16G 1% /mylv

[root@server0 ~]# ls -l /dev/mapper/systemvg-vo

lrwxrwxrwx. 1 root root 7 12月 8 15:55 /dev/mapper/systemvg-vo -> ../dm-0

1. **逻辑卷的扩展**-----支持线上操作

1 卷组有足够的剩余空间

格式: lvextend -L 新大小 /dev/卷组名/逻辑卷名

A: 扩展逻辑卷的空间

[root@server0 ~]# lvs

LV VG Attr LSize Pool Origin Data% Move Log Cpy%Sync Convert

vo systemvg -wi-ao---- 16.00g

[root@server0 ~]# vgs

VG #PV #LV #SN Attr VSize VFree

systemvg 2 1 0 wz--n- 19.99g 3.99g

[root@server0 ~]# lvextend -L 18G /dev/systemvg/vo

Extending logical volume vo to 18.00 GiB

Logical volume vo successfully resized

[root@server0 ~]# lvs

LV VG Attr LSize Pool Origin Data% Move Log Cpy%Sync Convert

空间大小

vo systemvg -wi-ao---- 18.00g

文件系统大小

[root@server0 ~]# df -h | tail -1

/dev/mapper/systemvg-vo 16G 33M 16G 1% /mylv

B:文件系统的扩展

xfs\_growfs: 扩展xfs文件系统

resize2fs: 扩展ext4文件系统-------**记住**

[root@server0 ~]# xfs\_growfs /dev/systemvg/vo

[root@server0 ~]# df -h | tail -1

/dev/mapper/systemvg-vo 18G 33M 18G 1% /mylv

2 卷组没有足够的剩余空间

A 扩展卷组

[root@server0 ~]# vgextend systemvg /dev/vdc3

[root@server0 ~]# vgs

VG #PV #LV #SN Attr VSize VFree

systemvg 3 1 0 wz--n- 29.99g 11.99g

B 扩展逻辑卷

[root@server0 ~]# lvextend -L 25G /dev/systemvg/vo

C 扩展文件系统

[root@server0 ~]# xfs\_growfs /dev/systemvg/vo

[root@server0 ~]# df -h | tail -1

/dev/mapper/systemvg-vo 25G 33M 25G 1% /mylv

了解内容:逻辑卷空间可以进行缩小

ext4文件系统支持缩小; xfs文件系统不支持缩小

卷组划分空间的单位: 默认4M ,称为PE

[root@server0 ~]# vgdisplay systemvg | head -16 | tail -1 #显示卷组详细信息

PE Size 4.00 MiB

创建卷组的时候设置PE大小

格式: vgcreate -s PE大小 卷组名 空闲分区...

创建逻辑卷的时候指定PE个数

格式: lvcreate -l PE个数 -n 逻辑卷名 卷组名

卷组已存在,修改卷组的PE大小

格式: vgchange -s PE大小 卷组名

题目:请创出一个逻辑卷为lvtest01 ,大小为250M

[root@server0 ~]# vgchange -s 1M systemvg

[root@server0 ~]# lvcreate -L 250M -n lvtest02 systemvg

[root@server0 ~]# lvs

LV VG Attr LSize Pool Origin Data% Move Log Cpy%Sync Convert

lvtest02 systemvg -wi-a----- 250.00m

vo systemvg -wi-ao---- 25.00g

[root@server0 ~]# vgdisplay systemvg | head -16 | tail -1

PE Size 1.00 MiB

了解: 逻辑卷的删除 (卷组没了,逻辑卷上的数据直接没有)

删除的顺序: 首先删除逻辑卷,再删除卷组,最后删除物理卷(可以不删除物理卷)

1 先卸载逻辑卷

[root@server0 ~]# lvremove /dev/systemvg/vo

Logical volume systemvg/vo contains a filesystem in use.

[root@server0 ~]# umount /dev/systemvg/vo

[root@server0 ~]# lvremove /dev/systemvg/vo

Do you really want to remove active logical volume vo? [y/n]: y

Logical volume "vo" successfully removed

[root@server0 ~]# lvremove lvtest02

2 删除卷组(没有任何一个逻辑卷基于此卷组创建)

[root@server0 ~]# vgremove /dev/systemvg

#########################################

**复习:**

1 划分分区的命令是什么?

fdisk

2 一般在磁盘中一个扇区为多少字节?

512字节

3 Linux中新硬盘经历哪些步骤才能存储文档?

识别硬盘----->分区----->格式化----->挂载存储数据

4 MBR常见的分区类型有哪三种?

主分区 扩展分区 逻辑分区

5 刷新新的分区表命令是什么?

partprobe

6 格式化ext4文件系统的命令是? 格式化xfs文件系统的命令?

mkfs.ext4 mkfs.xfs

7 查看分区的文件系统类型的命令是什么?

blkid

8 如何在开机的状态下: 检测/etc/fstab文件是否书写正确

mount -a

9 查看正在挂载的分区,使用情况的命令是什么?

df -h

10 开机挂载配置文件六个字段分别是什么?

设备 挂载点 文件系统 参数 备份标记 检测顺序

#########################################

**I-Day09**

shell脚本基础

脚本: 可以执行的一个文件,实现某种功能

|  |  |
| --- | --- |
| 常见的脚本语言 | Bash shell |
| Python/Perl/Ruby |
| JSP/PHP/ASP/CGI |
| JavaScipt |

shell脚本: 提前设计可执行语句,用来完成特定任务的文件

1) 解释型程序

2) 顺序、批量执行

案例: 编写一个面世问候/root/hello.sh脚本

显示出一段话”Hello World”

[root@server0 ~]# vim /root/hello.sh

echo Hello World

[root@server0 ~]# /root/hello.sh #默认vim编写的文件没有执行权限

-bash: /root/hello.sh: 权限不够

[root@server0 ~]# chmod +x /root/hello.sh

[root@server0 ~]# /root/hello.sh #以绝对路径执行

Hello World

规范shell脚本的一般组成

1) #! 环境声明 (默认是/bin/bash) python是/usr/bin/python

2) # 注释文本

3) 可执行代码

案例: 编写一个能输出系统信息/root/sysinfo.sh脚本

1) 输出当前红帽系统的版本信息

2) 输出当前使用的内核版本

3) 输出当前系统的主机名

[root@server0 ~]# vim /root/sysinfo.sh

#! /bin/bash

uname -r

cat /etc/redhat-release

hostname

[root@server0 ~]# chmod ugo+x /root/sysinfo.sh

[root@server0 ~]# /root/sysinfo.sh

3.10.0-123.el7.x86\_64

Red Hat Enterprise Linux Server release 7.0 (Maipo)

server0.example.com

管道传递

使用 ’|’ 管道操作

将前一条命令的标准输出交给后一条命令处理

重定向输出

> : 只收集前面命令的正确输出

2> : 只收集前面命令的错误输出

&> : 收集前面命令的正确输出与错误输出

**黑洞设备: /dev/null**

案例: 书写一个创建用户的脚本,并且设置密码

[root@server0 ~]# vim /root/user.sh

#! /bin/bash

useradd nsd01

echo 123 | passwd --stdin nsd01

[root@server0 ~]# chmod ugo+x /root/user.sh

[root@server0 ~]# /root/user.sh

更改用户 nsd01 的密码 。

passwd：所有的身份验证令牌已经成功更新。

[root@server0 ~]# vim /root/user.sh

#! /bin/bash

useradd nsd01 &> /dev/null

echo 用户nsd01创建成功

echo 123 | passwd --stdin nsd01 &> /dev/null

echo 用户nsd01密码设置成功

[root@server0 ~]# /root/user.sh

用户nsd01创建成功

用户nsd01密码设置成功

为了增加脚本的灵活度,适应多变的环境,使用变量

变量:会变化的量,类似于容器,以不变的名称存储可以变化的值

定义变量: 变量名=存放的值

[root@server0 ~]# vim /root/user.sh

#! /bin/bash

a=nsd03

useradd $a &> /dev/null

echo 用户$a创建成功

echo 123 | passwd --stdin $a &> /dev/null

echo 用户$a密码设置成功

[root@server0 ~]# /root/user.sh

用户nsd03创建成功

用户nsd03密码设置成功

定义/赋值变量

设置变量时的注意事项

1) 若指定的变量已存在,相当于为此变量重新赋值

2) 等号两边不要有空格

3) 变量名只能由字母/数字/下划线组成,区分大小写

4) 变量名不能以数字开头,不要使用关键字和特殊字符

查看/引用变量

基本格式:

引用变量值: $变量名

查看变量值: echo $变量名、 echo ${变量名}

[root@server0 ~]# a=10

[root@server0 ~]# echo $a

10

[root@server0 ~]# echo ${a}

10

[root@server0 ~]# a=rhel

[root@server0 ~]# echo $a7 #未定义的变量为空值

[root@server0 ~]# echo ${a}7

rhel7

**补充:**

单引号’’ : 取消特殊字符的意义,原样输出

反撇号`` 或$() : 将命令的输出,作为参数直接参与其他命令运行

[root@server0 ~]# rm -rf /opt/\*

[root@server0 ~]# mkdir /opt/nsd-`date +%F`

[root@server0 ~]# ls /opt

nsd-2018-12-10

[root@server0 ~]# mkdir /opt/abc-$(date +%F)

[root@server0 ~]# ls /opt

abc-2018-12-10 nsd-2018-12-10

[root@server0 ~]# mkdir /opt/`hostname`-$(date +%F)

[root@server0 ~]# ls /opt

abc-2018-12-10 nsd-2018-12-10 server0.example.com-2018-12-10

根据变量的用途不同区分

|  |  |
| --- | --- |
| **类型** | **说明** |
| 环境变量 | 变量名一般都大写,用来设置用户/系统环境 |
| 位置变量 | bash内置,存储执行脚本时提供的命令行参数 |
| 预定义变量 | bash内置,可直接调用的特殊值,不能直接修改 |
| 自定义变量 | 用户自主设置,修改及使用 |

**环境变量**: 变量名一般都大写,由系统定义完成,并且赋值完成

常见的环境变量: PWD、PATH、USER、LOGNAME、SHELL、HOME

USER=当前系统登录的用户名

PATH=储存命令程序的搜寻的路径

[root@server0 ~]# echo $USER

root

[root@server0 ~]# su - student

默认没有这个路径,需要自己创建

[student@server0 ~]$ echo $USER

Student

[root@server0 ~]# echo $PATH

/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin

[root@server0 ~]# vim /opt/hello.sh

#! /bin/bash

echo hello

[root@server0 ~]# chmod +x /opt/hello.sh

[root@server0 ~]# /opt/hello.sh

hello

[root@server0 ~]# cp /opt/hello.sh /usr/bin/

[root@server0 ~]# hello.sh

hello

[root@server0 ~]# which hello.sh

/usr/bin/hello.sh

**位置变量**: bash内置，存储执行脚本时提供的命令行参数；由系统定义完成,由系统赋值完成

表示为$n, n为序号

$1、$2、$3....${10}、${11}...

[root@server0 ~]# vim /root/user.sh

#! /bin/bash

useradd $1 &> /dev/null

echo 用户$1创建成功

echo 123 | passwd --stdin $1 &> /dev/null

echo 用户$1密码设置成功

[root@server0 ~]# /root/user.sh natasha

用户natasha创建成功

用户natasha密码设置成功

[root@server0 ~]# vim /root/1.sh

cat -n $1 | head -$2

[root@server0 ~]# chmod ugo+x /root/1.sh

[root@server0 ~]# /root/1.sh /etc/passwd 1

1 root:x:0:0:root:/root:/bin/bash

为了降低脚本的使用难度,提高客户的体验度,采用交互式传递参数方法

格式: read -p ‘屏幕提示信息’

作用: 1 产生交互

2 自动记录用户在键盘上的输入

3 自动将用户在键盘上的输入,赋值给一个变量储存

[root@server0 ~]# vim /root/user.sh

#! /bin/bash

read -p '请输入您要创建的用户名:' a

useradd $a &> /dev/null

echo 用户$a创建成功

echo 123 | passwd --stdin $a &> /dev/null

echo 用户$a密码设置成功

[root@server0 ~]# /root/user.sh

请输入您要创建的用户名:harry

用户harry创建成功

用户harry密码设置成功

**预定义变量**: bash内置,可直接调用的特殊值,不能直接修改,由系统定义完成,由系统赋值完成

|  |  |
| --- | --- |
| **变量名** | **含义** |
| $# | 已加载的位置变量的个数 |
| $\* | 所有位置变量的值 |
| $? | 程序退出后的状态值, 0表示正常,其他置异常 |

[root@server0 ~]# vim /root/2.sh

#! /bin/bash

echo $1

echo $2

echo $3

echo 您一共输入了$#个位置变量

echo 它们的值分别为$\*

[root@server0 ~]# chmod ugo+x /root/2.sh

[root@server0 ~]# /root/2.sh dc tc dz tz

dc

tc

dz

您一共输入了4个位置变量

它们的值分别为dc tc dz tz

**自定义变量**: 用户自主设置、修改及使用

**条件测试及选择**

几种常用的测试方式

正常的命令行

test -选项 参数...

[ 测试表达式 ] #每一部分之间都要有空格

检查文件状态

-e: 文档存在为真

-d: 文档存在,并且为目录才为真

-f: 文档存在,并且为文件才为真

-r: 文档存在,且对文档具备读取权限为真

-w: 文档存在,且对文档具备写权限为真

-x: 文档存在,且对文档具备执行权限为真

[root@server0 ~]# [ -e /etc ]

[root@server0 ~]# echo $?

0

[root@server0 ~]# [ -e /abc ]

[root@server0 ~]# echo $?

1

[root@server0 ~]# [ -f /etc/passwd ]

[root@server0 ~]# echo $?

0

[root@server0 ~]# [ -f /etc ]

[root@server0 ~]# echo $?

1

[root@server0 ~]# [ -d /etc ]

[root@server0 ~]# echo $?

0

比较整数大小(带e的都有等于二字, g是大于, l是小于)

-gt: 大于

-ge: 大于等于

-eq: 等于

-ne: 不等于

-lt: 小于

-le: 小于等于

[root@server0 ~]# [ 1 -eq 1 ]

[root@server0 ~]# echo $?

0

[root@server0 ~]# [ 1 -gt 2 ]

[root@server0 ~]# echo $?

1

[root@server0 ~]# [ 1 -ge 1 ]

[root@server0 ~]# echo $?

0

[root@server0 ~]# [ 10 -lt 11 ]

[root@server0 ~]# echo $?

0

字符串比对

== : 相等一致为真

!= : 不相等为真

[root@server0 ~]# [ root == root ]

[root@server0 ~]# echo $?

0

[root@server0 ~]# [ $USER == student ]

[root@server0 ~]# echo $?

1

[root@server0 ~]# [ $USER != student ]

[root@server0 ~]# echo $?

0

**if单分支处理**

单分支: 当条件满足时,作xx处理

**if双分支处理**

双分支: 当条件满足/不满足时,分别作xx,yy处理

if [ 条件测试 ] ; then

命令序列xx

else

命令序列yy

fi

[root@server0 ~]# vim /root/if01.sh

#! /bin/bash

if [ $1 -eq 1 ];then

echo hello

else

echo hi

fi

[root@server0 ~]# chmod ugo+x /root/if01.sh

[root@server0 ~]# /root/if01.sh 1

hello

[root@server0 ~]# /root/if01.sh 2

hi

案例: 判断IP地址的脚本

利用read获取用户输入的IP地址,进行判断:

如果本机与用户输入的IP地址可以通信,则输出可以通信;

如果本机与用户输入的IP地址可不以通信,则输出无法通信;

[root@server0 ~]# vim /root/if02.sh

#! /bin/bash

read -p '请输入您要测试的IP地址:' IP

ping -c 2 $IP &> /dev/null

if [ $? -eq 0 ]; then

echo $IP 可以通信

else

echo $IP 无法通信

fi

[root@server0 ~]# chmod +x /root/if02.sh

[root@server0 ~]# /root/if02.sh

请输入您要测试的IP地址127.0.0.1

127.0.0.1 可以通信

[root@server0 ~]# /root/if02.sh

请输入您要测试的IP地址:192.168.0.1

192.168.0.1 无法通信

**if多分支处理**

多分支: 当条件1满足时,作xx处理;否则继续检查条件2,若成立则作yy处理;否则,作zz处理

if [ 条件测试1 ] ; then

命令序列xx

elif [ 条件测试2 ] ; then

命令序列yy

elif [ 条件测试3 ] ; then

命令序列aa

elif [ 条件测试4 ] ; then

命令序列ss

........

else

命令序列zz

fi

案例: 书写考试成绩脚本

利用read获取用户的输入成绩

如果成绩大于等于90,则输出优秀

如果成绩大于等于80,则输出良好

如果成绩大于等于70,则输出一般

如果成绩大于等于60,则输出合格

以上条件均不满足,则输出’再牛的肖邦,也弹不出哥的悲伤’

[root@server0 ~]# vim /root/num.sh

#! /bin/bash

read -p '请输入您的成绩:' num

if [ $num -ge 90 ];then

echo 优秀

elif [ $num -ge 80 ];then

echo 良好

elif [ $num -ge 70 ];then

echo 一般

elif [ $num -ge 60 ];then

echo 合格

else

echo '再牛的肖邦,也弹不出哥的悲伤'

fi

[root@server0 ~]# chmod +x /root/num.sh

[root@server0 ~]# /root/num.sh

请输入您的成绩:98

优秀

[root@server0 ~]# /root/num.sh

请输入您的成绩:58

再牛的肖邦,也弹不出哥的悲伤

**列表式循环(for循环):**可以重复执行一个操作

根据变量不同的取值,重复执行xx处理

for 变量名 in 值列表

do

命令序列

done

[root@server0 ~]# vim /root/for01.sh

#! /bin/bash

for a in zhangsan lisi wangwu dz tc

do

useradd $a &> /dev/null

echo $a用户创建成功

done

[root@server0 ~]# chmod +x /root/for01.sh

[root@server0 ~]# /root/for01.sh

zhangsan用户创建成功

lisi用户创建成功

wangwu用户创建成功

dz用户创建成功

tc用户创建成功

for循环中,变量的值可以不参与循环执行的动作

造数工具: {起始点..结束点} {1..50}生成1到50 之间所有的数字

[root@server0 ~]# vim /root/for02.sh

#! /bin/bash

for a in {1..15}

do

echo hello world

done

[root@server0 ~]# chmod +x /root/for02.sh

[root@server0 ~]# /root/for02.sh

**案例3：Shell脚本**

为系统 server0 书写脚本/root/user.sh

运行脚本，可以判断用户输入的用户是否存在

如果存在，输出用户基本信息（id 用户名）

如果用户，不存在则创建用户，并输出用户创建成功

**法1:** [root@server0 ~]# vim /root/user.sh

#! /bin/bash

read -p '请输入您要测试的用户名:' a

id $a &> /dev/null

if [ $? -eq 0 ];then

echo 用户存在

id $a

else

useradd $a

echo 用户创建成功

fi

[root@server0 ~]# chmod +x /root/user.sh

[root@server0 ~]# /root/user.sh

请输入您要测试的用户名:lisi

用户创建成功

[root@server0 ~]# /root/user.sh

请输入您要测试的用户名:lisi

用户存在

uid=1001(lisi) gid=1001(lisi) 组=1001(lisi)

**法2:** [root@server0 ~]# vim /root/user.sh

#! /bin/bash

read -p '请输入您要测试的用户名:' a

useradd $a &> /dev/null

if [ $? -eq 0 ];then

echo 用户创建成功

else

echo 用户存在

id $a

fi

[root@server0 ~]# chmod +x /root/user.sh

[root@server0 ~]# /root/user.sh

请输入您要测试的用户名:tc

用户创建成功

[root@server0 ~]# /root/user.sh

请输入您要测试的用户名:tc

用户存在

uid=1004(tc) gid=1004(tc) 组=1004(tc)

**案例4: 编写一个判断脚本**

在server0上创建/root/foo.sh脚本

1) 当运行/root/foo.sh redhat,输出fedora

2) 当运行/root/foo.sh fedora, 输出redhat

3) 当没有任何参数或者参数不是redhat或者fedora时,其错误输出产生以

下信息: /root/foo.sh redhat|fedora

[root@server0 ~]# vim /root/foo.sh

#! /bin/bash

if [ $# -eq 0 ];then #匹配用户没有输入参数

echo '/root/foo.sh redhat|fedora' >&2 **>&2: 转变为错误输出**

exit 3 # 程序运行退出状态值

elif [ $1 == redhat ];then

echo fedora

elif [ $1 == fedora ];then

echo redhat

else

echo '/root/foo.sh redhat|fedora' >&2

exit 4 # 程序运行退出状态值

fi

[root@server0 ~]# chmod +x /root/foo.sh

[root@server0 ~]# /root/foo.sh

/root/foo.sh redhat|fedora

[root@server0 ~]# /root/foo.sh jkl

/root/foo.sh redhat|fedora

[root@server0 ~]# /root/foo.sh redhat

fedora

[root@server0 ~]# /root/foo.sh fedora

redhat

**案例5: 编写一个批量添加用户脚本**

在server0上创建/root/batchusers脚本

1. 此脚本要求提供用户名列表文件作为参数
2. 如果没有提供参数,此脚本应该给出提示 Usage:/root/batchusers,退出并返回相应值
3. 如果提供一个不存在的文件,此脚本应该给出提示 Input file not found ,退出并返回相应值
4. 新用户的登录shell为/bin/false ,无需设置密码
5. 用户列表测试文件: <http://classroom>/pub/materials/userlist

[root@server0 ~]# wget http://classroom.example.com/pub/materials/userlist

[root@server0 ~]# vim /root/batchusers

#! /bin/bash

if [ $# -eq 0 ];then

echo 'Usage:/root/batchusers' >&2 #将输出变为错误输出

exit 2

elif [ -f $1 ];then #判断命令行参数是否为文件

for a in `cat $1` #将命令的输出作为列表

do

useradd -s /bin/false $a &> /dev/null

echo $a创建成功

done

else

echo 'Input file not found' >&2

exit 5

fi

[root@server0 ~]# chmod +x /root/batchusers

[root@server0 ~]# /root/batchusers userlist

duanwu创建成功

zhongqiu创建成功

zhsan创建成功

lisi创建成功

[root@server0 ~]# /root/batchusers

Usage:/root/batchusers

[root@server0 ~]# /root/batchusers saa

Input file not found

#########################################

**I-Day10**

**系统安全保护**

SELinux安全机制

SELinux概述(Security-Enhanced Linux)

1) 美国NSA国家安全局主导开发,一套增强Linux系统安全的**强制**访问控制体系;

2) 集成到Linux内核(2.6及以上)中运行;

3) RHEL7基于SELinux体系针对用户、进程、目录和文件提供了预设的保护策略,以及管理工具

SELinux运行模式的切换

SELinux的运行模式:

enforcing (强制)

permissive (宽松)

disabled (彻底禁用)

任何模式变成disabled(彻底禁用),或者disabled变成其他模式,都需要经历重启系统

切换运行模式

临时切换: setenfore 1|0 1----强制模式 0----宽松模式

固定配置: /etc/selinux/config 文件**-----不会立即生效,需要重启**

查看当前模式: getenfore

虚拟机server0

[root@server0 ~]# vim /etc/selinux/config #修改固定配置文件

SELINUX=permissive

[root@server0 ~]# setenforce 0 #修改当前模式为permissive

[root@server0 ~]# getenforce #查看selinux当前模式

Permissive

虚拟机desktop0

[root@desktop0 ~]# vim /etc/selinux/config

SELINUX=permissive

[root@desktop0 ~]# setenforce 0

[root@desktop0 ~]# getenforce

Permissive

**防火墙策略基本使用**

虚拟机server0

一 搭建基本web服务

1 安装httpd软件

[root@server0 ~]# yum -y install httpd

2 重启httpd服务

[root@server0 ~]# systemctl restart httpd

[root@server0 ~]# systemctl enable httpd

3 本机访问测试

[root@server0 ~]# firefox 172.25.0.11

4 书写一个页面文件

默认页面文件存放路径: /var/www/html

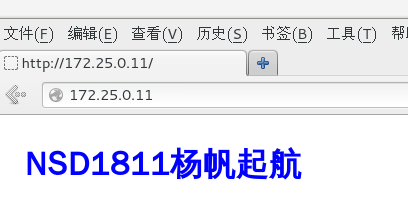
默认网页文件名称: index.html

[root@server0 ~]# vim /var/www/html/index.html

<marquee><font color=blue><h1>NSD1811杨帆起航

滚动 字体颜色为蓝色 最大标题字体

[root@server0 ~]# firefox 172.25.0.11



二 搭建基本FTP服务(文件传输协议)

1 安装vsftpd软件

[root@server0 ~]# yum -y install vsftpd

2 重启vsftpd服务

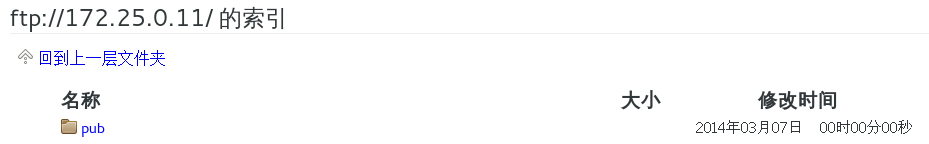
[root@server0 ~]# systemctl restart vsftpd

[root@server0 ~]# systemctl enable vsftpd

3 本机访问测试

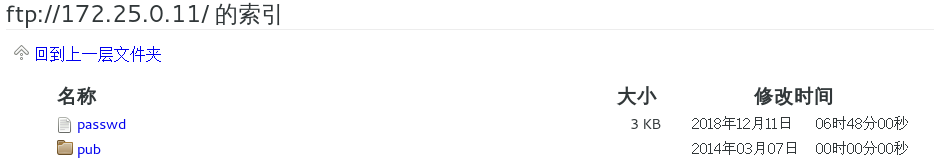
默认共享数据的路径: /var/ftp/

[root@server0 ~]# firefox <ftp://172.25.0.11>



[root@server0 ~]# cp /etc/passwd /var/ftp/

[root@server0 ~]# firefox ftp://172.25.0.11



**防火墙策略管理**

作用: 隔离过滤

规则: 允许出站,过滤入站

硬件防火墙:

软件防火墙: 360防火墙 金山毒霸防火墙......

**firewalld服务基础**

RHEL7的防火墙体系

系统服务: firewalld

管理工具: firewall-cmd(命令工具)、 firewall-config(图形工具)

预设安全区域  
 根据所在的网络场所区分,预设保护规则集

public: 仅允许访问本机的sshd、DHCP、ping等少数几个服务

trusted: 允许任何访问

block: 阻塞任何来访请求(明确拒绝,会给回应)

drop: 丢弃任何来访的数据包(没有回应,直接丢弃,节省资源)

数据包: 源IP地址、目标IP地址、数据 目标端口

防火墙判定的规则:

1) 查看客户端请求中源IP地址,然后查阅自己哪个区域有该IP地址的规则,则进入该区域

2) 进入默认区域public; root用户可以修改默认区域

查看防火墙规则列表

列表查看操作

firewall-cmd --list-all [--zone=区域名]

firewall-cmd --list-all-zones

firewall-cmd --get-zones

firewall-cmd --get-services

firewall-cmd --get-default-zone

虚拟机server0

[root@server0 ~]# firewall-cmd --get-default-zone #查看默认区域

public

虚拟机desktop0

[root@desktop0 ~]# ping 172.25.0.11 #可以通信

虚拟机server0

[root@server0 ~]# firewall-cmd --set-default-zone=block #修改默认防火墙区域

success

[root@server0 ~]# firewall-cmd --get-default-zone

block

虚拟机desktop0

[root@desktop0 ~]# ping 172.25.0.11 #不可以通信,有回应

PING 172.25.0.11 (172.25.0.11) 56(84) bytes of data.

From 172.25.0.11 icmp\_seq=1 Destination Host Prohibited

虚拟机server0

[root@server0 ~]# firewall-cmd --set-default-zone=drop #修改默认防火墙区域

success

[root@server0 ~]# firewall-cmd --get-default-zone

drop

虚拟机desktop0

[root@desktop0 ~]# ping 172.25.0.11 #不可以通信,没有回应

PING 172.25.0.11 (172.25.0.11) 56(84) bytes of data.

**再public区域添加服务或协议**

虚拟机server0

[root@server0 ~]# firewall-cmd --set-default-zone=public

success

[root@server0 ~]# firewall-cmd --get-default-zone

public

[root@server0 ~]# firewall-cmd --list-all --zone=public | head -4 | tail -1

services: dhcpv6-client ssh

虚拟机desktop0

[root@desktop0 ~]# firefox 172.25.0.11 #无法访问

[root@desktop0 ~]# firefox <ftp://172.25.0.11> #无法访问

虚拟机server0

[root@server0 ~]# firewall-cmd --zone=public --add-service=ftp

[root@server0 ~]# firewall-cmd --zone=public --add-service=http

[root@server0 ~]# firewall-cmd --list-all --zone=public | head -4 | tail -1

services: dhcpv6-client ftp http ssh

虚拟机desktop0

[root@desktop0 ~]# firefox 172.25.0.11 # 访问成功

[root@desktop0 ~]# firefox <ftp://172.25.0.11> #访问成功

**常见互联网应用协议:**-----------------**记住**

http: 超文本传输协议 默认端口: 80

https: 安全超文本传输协议 默认端口: 443

ftp: 文件传输协议 默认端口: 21

tftp: 简单文件传输协议 默认端口: 69

DNS: 域名解析协议 默认端口: 53

telnet: 远程管理协议 默认端口: 23

SMTP: 邮件协议(用户发邮件) 默认端口: 25

pop3: 邮件协议(用户收邮件) 默认端口: 110

SNMP: 简单网络管理协议 默认端口: 161

**永久在public区域中添加协议**

配置规则的位置

运行时 (--runtime)

永久 (--permanent) #需要重新加载或者重启,实则改的是配置文件里的内容

[root@server0 ~]# firewall-cmd --reload #重新加载防火墙所有规则

success

[root@server0 ~]# firewall-cmd --zone=public --list-all | head -4 | tail -1

services: dhcpv6-client ssh

[root@server0 ~]# firewall-cmd --permanent --zone=public --add-service=http

success

[root@server0 ~]# firewall-cmd --reload #重新加载防火墙所有规则

success

[root@server0 ~]# firewall-cmd --zone=public --list-all | head -4 | tail -1

services: dhcpv6-client http ssh

了解内容: 单独拒绝虚拟机desktop0-----将虚拟机desktop0的IP地址放入block区域中

虚拟机server:设置防火墙规则

[root@server0 ~]# firewall-cmd --permanent --zone=block --add-source=172.25.0.10

success

[root@server0 ~]# firewall-cmd --reload

success

[root@server0 ~]# firewall-cmd --list-all --zone=block | head -3 | tail -1

sources: 172.25.0.10

虚拟机desktop0

[root@desktop0 ~]# firefox 172.25.0.11 #刷新浏览器访问,失败

[root@desktop0 ~]# ping 172.25.0.11 #失败

[root@room9pc01 ~]# firefox 172.25.0.11 #真机访问成功

删除规则:

[root@server0 ~]# firewall-cmd --permanent --zone=block --remove-source=172.25.0.10

[root@server0 ~]# firewall-cmd --reload

临时删除:

[root@server0 ~]# firewall-cmd --zone=public --remove-service=http

永久删除:

[root@server0 ~]# firewall-cmd --permanent --zone=public --remove-service=http

**实现本机的端口转发(映射)**

端口: 协议或服务或程序的编号,起到标识的作用; root可以改变端口;一个程序可以具备多个端口,(在主机标识协议或服务或程序)

案例: 本地应用的端口重定向(端口1---->端口2)

从客户机访问端口5423的请求,自动映射到本机端口80

比如,访问以下两个地址可以看到相同的页面:

[http://172.25.0.11:5423/](http://server0.example.com:5423/)

[http://1](http://server0.example.com:5423/)72.25.0.11

虚拟机desktop0

[root@desktop0 ~]# firefox 172.25.0.11:5423 #访问失败

虚拟机server0

[root@server0 ~]# firewall-cmd --permanent --zone=public --add-forward-port=port=5423:proto=tcp:toport=80 -----**记住**

success

[root@server0 ~]# firewall-cmd --reload

success

[root@server0 ~]# firewall-cmd --zone=public --list-all | head -7 | tail -1

forward-ports: port=5423:proto=tcp:toport=80:toaddr=

虚拟机desktop0

[root@desktop0 ~]# firefox 172.25.0.11:5423 #访问成功

#########################################

**I-Day11**

环境准备：设置两台虚拟机防火墙默认区域为trusted

[root@server0 ~]# firewall-cmd --set-default-zone=trusted

[root@desktop0 ~]# firewall-cmd --set-default-zone=trusted

**配置SMB共享**

Samba服务基础: 跨平台的共享(Windows与Linux)

Samba软件项目

用途: 为客户机提供共享使用的文件夹

协议: SMB(TCP 139)、CIFS(TCP 445)

所需软件包: samba

系统服务: smb

管理共享账号

Samba用户: 专用来访问共享文件夹的用户

1) 采用独立设置的密码

2) 但需要提前建立同名的系统用户(可以不设密码)

必须建立系统用户,登录系统的密码可以设置,也可以不设置

必须建立系统用户,可以不用登录系统(useradd -s /sbin/nologin dc)

使用pdbedit 管理工具

添加用户: pdbedit -a 用户名

查询用户: pdbedit -L [用户名]

删除用户: pdbedit -x 用户名

**虚拟机server0:**

1 安装软件samba

[root@server0 ~]# yum -y install samba

2 建立Samba共享账号

[root@server0 ~]# useradd -s /sbin/nologin harry

[root@server0 ~]# useradd -s /sbin/nologin kenji

[root@server0 ~]# useradd -s /sbin/nologin chihiro

[root@server0 ~]# pdbedit -a harry

[root@server0 ~]# pdbedit -a kenji #添加为Samba共享账号

[root@server0 ~]# pdbedit -a chihiro

[root@server0 ~]# pdbedit -L #查看Samba共享账号

harry:1001:

chihiro:1003:

kenji:1002:

3 创建共享目录

[root@server0 ~]# mkdir /common

[root@server0 ~]# echo 123 > /common/1.txt

4 修改配置文件/etc/samba/smb.conf

**[自定共享名]**

**path = 文件夹绝对路径**

public = no|yes 默认no,与windows的guest来宾用户有关

browseable = yes|no 默认yes,与隐藏共享有关

read only = yes|no 默认yes

**write list = 用户1.... 默认无**

valid users = 用户1.... 默认所有samba共享账号

hosts allow = 客户机地址....

hosts deny = 客户机地址....

[root@server0 ~]# vim /etc/samba/smb.conf

workgroup = STAFF #此服务器必须是STAFF工作组的一个成员

命令模式下,按大写的G到全文的最后

[common] #共享名

path = /common #共享的实际路径

5 重启smb服务,设置为开机自启

[root@server0 ~]# systemctl restart smb

[root@server0 ~]# systemctl enable smb

6 修改selinux策略[**布尔值(服务功能开和关)**、安全上下文、非默认端口开放]

**需要加-P(大写)选项才能实现永久设置,但是会消耗内存**

[root@server0 ~]# getsebool -a | grep samba | head -5 | tail -2 #查看布尔值

samba\_export\_all\_ro --> off

samba\_export\_all\_rw --> off

[root@server0 ~]# setsebool samba\_export\_all\_ro on #修改布尔值,开启只读访问

[root@server0 ~]# getsebool -a | grep samba | head -5 | tail -2

samba\_export\_all\_ro --> on

samba\_export\_all\_rw --> off

**虚拟机desktop0:**

1 安装客户端软件samba-client (客户端软件: 访问服务端服务)

[root@desktop0 ~]# yum -y install samba-client

2 使用客户端软件提供的命令,访问服务端共享

列出共享资源:

格式: smbclient -L 服务器地址

连接到共享文件夹:

格式: smbclient -U 用户名 //服务器地址/**共享名**

[root@desktop0 ~]# smbclient -L 172.25.0.11

Enter root's password: # 直接回车,不用输密码

Anonymous login successful # 以匿名用户进行登录

[root@desktop0 ~]# smbclient -U harry //172.25.0.11/common

Enter harry's password: #输入密码(123)

Domain=[STAFF] OS=[Unix] Server=[Samba 4.1.1]

smb: \> ls #如果没有开启selinux布尔值策略服务,是不能访问的

. D 0 Wed Dec 12 10:04:13 2018

.. D 0 Wed Dec 12 10:04:03 2018

1.txt N 4 Wed Dec 12 10:04:13 2018

40913 blocks of size 262144. 28589 blocks available

**客户端更加方便的访问:**

1 安装cifs-utils,支持cifs协议及相应文件系统

[root@desktop0 ~]# yum -y install cifs-utils

2 进行手动挂载

[root@desktop0 ~]# mkdir /mnt/nsd

[root@desktop0 ~]# mount **-o username=harry,passwd**=123 //172.25.0.11/common /mnt/nsd

[root@desktop0 ~]# mount **-o user=harry,pass**=123 //172.25.0.11/common /mnt/nsd

[root@desktop0 ~]# df -h | tail -1 #查看samba共享挂载情况

//172.25.0.11/common 10G 3.1G 7.0G 31% /mnt/nsd

[root@desktop0 ~]# ls /mnt/nsd

1.txt

3 开机自动挂载: 修改配置文件/etc/fstab

设备路劲 挂载点 文件系统类型 参数 备份标记 检测顺序

\_netdev : 声明网络设备;在挂载该设备时,需要配置好本机IP地址及网络服务正常,才能挂载本设备

[root@desktop0 ~]# vim /etc/fstab

//172.25.0.11/common /mnt/nsd cifs defaults,user=harry,pass=123,\_netdev 0 0

[root@desktop0 ~]# umount /mnt/nsd #卸载前面的手动挂载

[root@desktop0 ~]# mount -a

[root@desktop0 ~]# df -h | tail -1

//172.25.0.11/common 10G 3.1G 7.0G 31% /mnt/nsd

**读写Samba共享**

**虚拟机server0**

1 创建共享的目录

[root@server0 ~]# mkdir /devops

[root@server0 ~]# echo abc > /devops/a.txt

2 修改Samba配置文件/etc/samba/smb.conf

[root@server0 ~]# vim /etc/samba/smb.conf

[devops] #共享名

path = /devops #共享的实际路径

write list = chihiro #允许chihiro用户可以读写

3 重启smb服务

[root@server0 ~]# systemctl restart smb

4 修改selinux策略(布尔值),开放读写权限

[root@server0 ~]# getsebool -a | grep samba | head -5 | tail -2

samba\_export\_all\_ro --> on

samba\_export\_all\_rw --> off

[root@server0 ~]# setsebool samba\_export\_all\_rw on

[root@server0 ~]# getsebool -a | grep samba | head -5 | tail -2

samba\_export\_all\_ro --> on

samba\_export\_all\_rw --> on

5 设置本地目录权限

[root@server0 ~]# setfacl -m u:chihiro:rwx /devops

[root@server0 ~]# getfacl /devops | head -5 | tail -1

getfacl: Removing leading '/' from absolute path names

user:chihiro:rwx

**客户端虚拟机desktop0**

1 开机自动挂载

[root@desktop0 ~]# mkdir /mnt/dev

[root@desktop0 ~]# vim /etc/fstab

//172.25.0.11/devops /mnt/dev cifs defaults,user=chihiro,pass=123,\_netdev 0 0

[root@desktop0 ~]# mount -a

[root@desktop0 ~]# df -h | tail -1

//172.25.0.11/devops 10G 3.1G 7.0G 31% /mnt/dev

[root@desktop0 ~]# cat /mnt/dev/a.txt

abc

[root@desktop0 ~]# touch /mnt/dev/b.txt

[root@desktop0 ~]# ls /mnt/dev/

a.txt b.txt

影响客户端访问服务端的因素--------**记住**

1) 防火墙

2) 安全selinux

3) 服务本身的访问控制

4) 本地目录的权限

必须全部满足才可以,一个不满足就不能访问(一票否定权)

案例: 配置smb文件夹共享

在server0上发布共享文件夹,要求如下:

1) 发布目录/public,共享名为class

2) 用户kenji对共享class可读写权限,密码123

**虚拟机server0:**

[root@server0 ~]# mkdir /public

[root@server0 ~]# vim /etc/samba/smb.conf

[class]

path = /public

write list = kenji

root@server0 ~]# systemctl restart smb

[root@server0 ~]# chmod o+w /public

**虚拟机desktop0:**

[root@desktop0 ~]# vim /etc/fstab

//172.25.0.11/class /mnt/abc cifs defaults,user=kenji,pass=123,\_netdev 0 0

[root@desktop0 ~]# mkdir /mnt/abc

[root@desktop0 ~]# mount -a

[root@desktop0 ~]# touch /mnt/abc/1.txt

[root@desktop0 ~]# ls /mnt/abc/

1.txt

**总结Samba共享的搭建**

服务端:

1) 设置防火墙默认区域为trusted

2) 安装软件包samba

3) 创建共享账号 pdbedit

4) 创建共享目录

5) 修改配置文件/etc/samba/smb.conf 发布共享, 指明可写用户是谁

(write list = 可写用户)

6) 重启服务

7) 开放selinux读写布尔值

8) 设置本地目录的权限

客户端:

1) 设置防火墙默认区域为trusted

2) 安装软件cifs-utils

3) 挂载访问

4) 实现开机自动挂载

**multiuser机制(了解内容)**

SMB客户端的multiuser挂载技术

1) 管理员只需要作一次挂载

2) 客户端在访问挂载点时,若需要不同权限,可以临时切换为新的共享用户

(无需重新挂载)

实现方式

1) 挂载SMB共享时启用multiuser支持

2) 使用cifscreds临时切换身份

mount.cifs的挂载参数

1) multiuser 提供对客户端多个用户身份的区分支持

2) sec=ntlmssp 提供NT局域网管理安全支持

[root@desktop0 ~]# vim /etc/fstab

//172.25.0.11/devops /mnt/dev cifs defaults,user=chihiro,pass=123,\_netdev,multiuser,sec=ntlmssp 0 0

[root@desktop0 ~]# umount /mnt/dev

[root@desktop0 ~]# mount -a

[root@desktop0 ~]# df -h

[root@desktop0 ~]# useradd lisi

[root@desktop0 ~]# su - lisi

[lisi@desktop0 ~]$ ls /mnt/dev/

ls: cannot access /mnt/dev/: Permission denied

使用cifscreds提交新的用户凭据并测试

格式: cifscreds add -u 共享用户名 服务器地址

[lisi@desktop0 ~]$ cifscreds add -u chihiro 172.25.0.11

Password:

[lisi@desktop0 ~]$ ls /mnt/dev/

a.txt b.txt

[lisi@desktop0 ~]$ touch /mnt/dev/c.txt

[lisi@desktop0 ~]$ ls /mnt/dev/

a.txt b.txt c.txt

**配置NFS共享: Linux与Linux平台之间的共享**

Network File System,网络文件系统

用途: 为客户机提供共享使用的文件夹

协议: NFS(TCP/UDP 2049) 、RPC(TCP/UDP 111)

所需软件包: nfs-utils

系统服务: nfs -server

**虚拟机server0:**

1 安装nfs-utils软件 #默认已安装

[root@server0 ~]# yum -y install nfs-utils

2 创建共享目录

[root@server0 ~]# mkdir /nfs

[root@server0 ~]# echo haha > /nfs/1.txt

3 修改配置文件/etc/exports发布共享

文件夹路径 客户机地址(权限) 客户机地址(权限)...

[root@server0 ~]# vim /etc/exports

/nfs \*(ro)

4 重启服务nfs-server

[root@server0 ~]# systemctl restart nfs-server

[root@server0 ~]# systemctl enable nfs-server

**客户端虚拟机desktop0:**

1 完成开机自动挂载/etc/fstab

服务器地址:文件夹路径 挂载点 nfs defaults,\_netdev 0 0

[root@desktop0 ~]# vim /etc/fstab

172.25.0.11:/nfs /mnt/nfsmount nfs defaults,\_netdev 0 0

2 进行mount -a 检测

[root@desktop0 ~]# mkdir /mnt/nfsmount

[root@desktop0 ~]# mount -a

[root@desktop0 ~]# df -h | tail -1

172.25.0.11:/nfs 10G 3.1G 7.0G 31% /mnt/nfsmount

[root@desktop0 ~]# cat /mnt/nfsmount/1.txt

haha

用户个性化配置文件

影响指定用户的bash解释环境

~/.bashrc, 每次开启bash终端时生效

全局环境配置

影响所有用户的bash解释环境

/etc/bashrc,每次开启bash终端时生效

[root@server0 ~]# vim /root/.bashrc

alias hello='echo hello'

[root@server0 ~]# vim /home/student/.bashrc

alias hi='echo hi'

[root@server0 ~]# vim /etc/bashrc

alias abc='echo abc'

[root@server0 ~]# hello #需要重新gos

hello

[root@server0 ~]# abc

abc

[root@server0 ~]# hi

bash: hi: 未找到命令..

[root@server0 ~]# su - student

[student@server0 ~]$ abc

abc

[student@server0 ~]$ hello

bash: hello: command not found...

[student@server0 ~]$ hi

hi

**配置高级连接**

配置ipv6地址

ipv6地址表示:

1) 128个二进制位,冒号分隔的十六进制数

2) 每段内连续的前置0可省略、连续的多个 : 可简化为::

例如: 2003:ac18:0000:0000:0000:0000:0000:0305 = 2003:ac18::305

[root@server0 ~]# nmcli connection modify 'System eth0' ipv6.method manual ipv6.addresses '2003:ac18::305/64' connection.autoconnect yes

[root@server0 ~]# nmcli connection up 'System eth0'

[root@server0 ~]# ping6 2003:ac18::305

**复习:**

1 Linux中黑洞设备是什么?

/dev/null

2 Linux中正确信息与错误信息信息都收集的重定向是什么?

&>

3 下列变量名中有效的shell变量名是(C)

A: -2-time B: \_2$3

C: trust\_no\_1 D: 2004file

4 vim编辑器有几种工作模式,分别是?

三种 命令模式 插入模式 末行模式

5 开机挂载配置文件六个字段分别为什么?

/etc/fstab

设备路径 挂载点 文件系统类型 参数 备份标记 检测顺序

6 搭建smb服务,需要安装什么软件?

samba

7 配置nfs共享,配置文件在哪里?

/etc/exports

8 yum客户端配置文件放在什么路径下? 具体都有哪些字段?

/etc/yum.repos.d/\*.repo

[] name baseurl enabled gpgcheck

#########################################

**I-Day12**

环境准备：设置两台虚拟机防火墙默认区域为trusted

[root@server0 ~]# firewall-cmd --set-default-zone=trusted

[root@desktop0 ~]# firewall-cmd --set-default-zone=trusted

**iscsi网络磁盘 默认端口：3260**

Internet SCSI,网际SCSI接口

1) 一种基于C/S架构的虚拟磁盘技术

2) 服务器提供磁盘空间,客户机连接并当成本地磁盘使用

iscsi磁盘的构成

backstore-----后端存储 /dev/vdb1 大冰箱

对应到服务端提供实际存储空间的设备,**需要起一个管理名称**

target-----磁盘组 木质箱子

是客户端的访问目标,作为一个框架,由多个lun组成

lun-----逻辑单元 将大冰箱+纸质箱子, 放入木质箱子(是一个动作)

每一个lun需要关联到某一个后端存储设备,在客户端会视为一块虚拟硬盘

举例: 大冰箱------>纸制的包装------>木质包装

**虚拟机server0:**

一 划分一个主分区5G

[root@server0 ~]# fdisk /dev/vdb

[root@server0 ~]# lsblk | tail -1

└─vdb1 253:17 0 5G 0 part

二 安装软件targetcli

[root@server0 ~]# yum -y install targetcli

三 配置共享存储

[root@server0 ~]# targetcli #进入共享iscsi磁盘配置模式,会自动写入配置文件,因为直接修改配置文件过于复杂

1 创建后端存储,为其设置名字为nsd (购买大冰箱,进行纸质的包装)

/> backstores/block create dev=/dev/vdb1 name=nsd

2 创建targetcli磁盘组 (打造木质的箱子)

ISCSI Qualified Name 名称规范 (iqn)

1) iqn.yyyy-mm.倒序域名:自定义标识

2) 用来识别 target 磁盘组,也用来识别客户机身份

/> iscsi/ create iqn.2016-02.com.example:server0

3 lun关联 (将后端存储放入target磁盘组)

/> iscsi/iqn.2016-02.com.example:server0/tpg1/luns create /backstores/block/nsd #可以tab键

4 配置ACL访问控制

设置客户端访问时声称的名字,该名字必须符合iqn命名规范

/> iscsi/iqn.2016-02.com.example:server0/tpg1/acls create iqn.2018-12.com.example:abc

5 启用网络接口及端口号

/> iscsi/iqn.2016-02.com.example:server0/tpg1/portals create 172.25.0.11 #可以按tab键

/> exit #保存并退出

四 重启服务target

[root@server0 ~]# systemctl restart target

[root@server0 ~]# systemctl enable target

**虚拟机desktop0**

1 安装软件包iscsi-initiator-utils

yum仓库具备补全仓库名功能

1) yum要生成缓存

2) 当前系统没有安装该软件包

[root@server0 ~]# rpm -q iscsi-initiator-utils 默认已经装了64位的

iscsi-initiator-utils-6.2.0.873-21.el7.x86\_64

server0 ~]# yum -y install iscsi-initiator-utils.i686 #32位的,不用装

2 修改配置文件/etc/iscsi/initiatorname.iscsi,指定客户端声称的名字

[root@desktop0 ~]# vim /etc/iscsi/initiatorname.iscsi #tab键

InitiatorName=iqn.2018-12.com.example:abc

3 重启iscsid服务,刷新客户端声称的名字

[root@desktop0 ~]# systemctl restart iscsid #有些时候不会报错,看人品

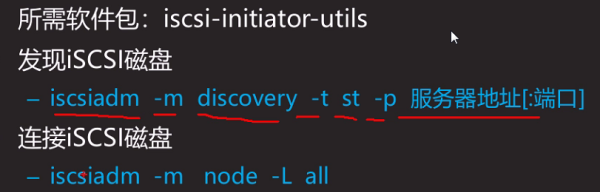
Warning: Unit file of iscsid.service changed on

disk, 'systemctl daemon-reload' recommended.

[root@desktop0 ~]# systemctl daemon-reload

[root@desktop0 ~]# systemctl restart iscsid

4 加载(发现)服务端共享存储



Linux查看命令帮助的方法: man iscsiadm

全文查找/example

[root@desktop0 ~]# iscsiadm --mode discoverydb --type sendtargets --portal 172.25.0.11 --discover

5 重启客户端服务,识别加载共享存储

[root@desktop0 ~]# lsblk

[root@desktop0 ~]# systemctl restart iscsi

[root@desktop0 ~]# systemctl enable iscsi

[root@desktop0 ~]# lsblk | head -2

NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT

sda 8:0 0 5G 0 disk

[root@desktop0 ~]#

**总结:**

服务端:

1 设置防火墙默认区域为trusted

2 安装软件包targetcli

3 运行targetcli命令执行配置

a 创建后端存储

b 创建磁盘组

c 进行关联lun

d 配置ACL访问控制,客户端声称的名字

e 启用网络接口,启用端口

4 重启服务target

客户端:

1 设置防火墙默认区域为trusted

2 安装软件包iscsi-initiator-utils

3 修改配置文件,指定客户端声称的名字

4 重启iscsid刷新客户端声称的名字

5 执行命令,发现服务端共享存储

6 重启客户端服务iscsi,识别加载共享存储

**数据库服务基础**

DB: database

数据库: 存放数据的仓库,一批数据的集合,主流的数据库多用来存放关系型表格数据

关系型数据: 以二维表格记录大量实体的属性信息

DBMS: 数据库管理系统

在数据库中会有很多的库,在每一个库中会有很多的表格

常见的关系型数据库管理系统:

1) 微软的SQL Server

2) IBM的DB2

3) 甲骨文的Oracle、MySQL

4) 社区开源版 MariaDB

(MariaDB和MySQL是同一个作者写的,相当于是同一个软件,就是名字不一样)

一 部署mariadb数据库服务器, **默认端口: 3306**

RHEL7中的MariaDB相关包

1) mariadb-server: 提供服务端有关的系统程序

2) mariadb: 提供客户端及管理工具

[root@server0 ~]# yum -y install mariadb-server

[root@server0 ~]# systemctl restart mariadb

[root@server0 ~]# systemctl enable mariadb

二 数据库的基本操作

访问mariadb数据库: mysql [-u用户名] [-p[密码]]

数据库的增删查

MariaDB [(none)]> 交互指令

列出数据库: show databases;

使用/选择数据库: use 数据库名;

列出库里有哪些表: show tables;

创建数据库: create database 数据库名;

删除数据库: drop database 数据库名;

[root@server0 ~]# mysql #进入数据库

MariaDB [(none)]> create database nsd1811; #创建nsd1811库

MariaDB [(none)]> show databases; #显示所有的库

三 为数据库管理员设置密码

数据库管理员: 账户名: root 用户信息存放在mysql库中user表中

系统管理员: 账户名: root 用户信息存放在/etc/passwd

格式: mysqladmin [-u 用户名] [-p[旧密码]] password ‘新密码’

[root@server0 ~]# mysqladmin -u root password '123'

[root@server0 ~]# mysql -u root -p #交互式登录

注意没有空格

Enter password:

[root@server0 ~]# mysql -u root -p123 #非交互式登录

禁止监听网络,只服务于本机

[root@server0 ~]# vim /etc/my.cnf # 数据库主配置文件

[mysqld]

skip-networking # 跳过网络监听

.. ..

[root@server0 ~]# systemctl restart mariadb

数据库的导出和导入操作

导出/备份数据库

mysqldump [-u 用户名] [-p[密码]] 数据库名 > 备份文件.sql

导入/恢复到数据库

mysql [-u 用户名] [-p[密码]] 数据库名 < 备份文件.sql

[root@server0 ~]# wget http://classroom.example.com/pub/materials/users.sql

[root@server0 ~]# mysql -u root -p123 nsd1811 < users.sql

[root@server0 ~]# mysql -u root -p123

MariaDB [(none)]> use nsd1811; #进入nsd811库

MariaDB [nsd1811]> show tables; #查看库中所有表格

**数据表的增删查**

创建新的数据表

create table 表名(字段1名 类型(长度), 字段2名 类型(长度), .. ..);

查看表结构

describe 表名; 或者desc 表名;

删除数据表

drop table 表名;

**表格: 表记录 表字段**

基本操作: 增(insert) 删(delete) 改(update) 查(select)

insert插入表记录

insert into [数据库.]表名 values(值1,值2, .. ..);

select查询表记录

select \* from [数据库.]表名;

select 字段1,字段2,... from [数据库.]表名;

select .. .. where 字段1=’值’ and|or 字段2=’值’;

update修改表记录

update [数据库名.]表名 set 字段名=新值 where 条件语句;

delete删除表记录

delete from [数据库.]表名 where条件语句;

多表组合查询

统计查询结果数量: select count(\*) .. ..;

关联查询: .. ..from 表1,表2 where 表1.字段1=’值’ and 表2.字段2=’值’ and 表1.字段1=表2.字段2;

MariaDB [nsd1811]> show tables; #查看当前所有库表格

MariaDB [nsd1811]> select \* from base; #查看所有表字段信息

MariaDB [nsd1811]> use mysql; #进入mysql库

MariaDB [mysql]> select \* from nsd1811.base;

有条件的查询:

select 表字段 from [库名.]表名 where 表字段=’值’;

MariaDB [nsd1811]> select \* from base where password='123';

MariaDB [nsd1811]> select \* from base where name='james';

查看表结构: desc 表名;

MariaDB [nsd1811]> use mysql;

MariaDB [mysql]> desc user;

MariaDB [mysql]> select user,host,password from user;

**用户授权设置**

格式: grant 权限列表 on 数据库名.表名 to 用户名@客户机地址 identified by ‘密码’

例子: 除了root用户,数据库nsd1811只能被用户lisi查询, 此用户密码为123

[root@server0 ~]# mysql -u root -p123

MariaDB [(none)]> grant select on nsd1811.\* to lisi@localhost identified by '123'; # 当lisi从本地localhost登录后,输入密码123将会获得nsd811库中所有表的查询权限

MariaDB [(none)]> select user,host,password from mysql.user;

MariaDB [(none)]> exit

[root@server0 ~]# mysql -u lisi -p123

MariaDB [(none)]> select \* from nsd1811.base;

案例: 使用数据库查询

1 在系统server0上使用数据库nsd1811, 并使用相应的SQL查询以回答下列问题:

1. 密码是solicitous的人的名字?

[root@server0 ~]# mysql -u root -p123

MariaDB [(none)]> use nsd1811;

MariaDB [nsd1811]> select \* from base where password='solicitous';

2) 有多少人的姓名是Barbara同时居住在sunnyvale?

[root@server0 ~]# mysql -u root -p123

MariaDB [(none)]> use nsd1811;

MariaDB [nsd1811]> select count(\*) from base,location where base.name='Barbara' and location.city='sunnyvale' and base.id=location.id;

+----------+

| count(\*) |

+----------+

| 1 |

+----------+

1 row in set (0.00 sec)

案例2: 禁止空密码root用户访问mariadb数据库

[root@server0 ~]# mysql -u root -p123

MariaDB [(none)]> use mysql;

MariaDB [mysql]> delete from user where password=''; #删除密码为空的记录

MariaDB [mysql]> flush privileges; #刷新user表记录

[root@server0 ~]# mysql -u root -h server0.example.com; #登录失败

-h : 登录方式

#######################################

**I-Day13**

环境准备：设置两台虚拟机防火墙默认区域为trusted

[root@server0 ~]# firewall-cmd --set-default-zone=trusted

[root@desktop0 ~]# firewall-cmd --set-default-zone=trusted

**HTTP服务基础**

web通信基本概念:

基于B/S(Browser/Server)架构的网页服务

作用: 服务端提供网页; 浏览器下载并显示网页

Hyper Text Markup Language -----超文本标记语言(html)

Hyper Text Transfer Protocol ------超文本传输协议(http),默认端口80

web常用计算机语言: Apache(httpd)、nginx、Tomcat

一 搭建基本的web服务

虚拟机server0

1 安装软件httpd

[root@server0 ~]# yum -y install httpd

2 书写页面文件

[root@server0 ~]# echo '<h1>NSD1811 web1' > /var/www/html/index.html

3 重启httpd服务

[root@server0 ~]# systemctl restart httpd

[root@server0 ~]# systemctl enable httpd

虚拟机desktop0上验证

[root@desktop0 ~]# firefox 172.25.0.11



RHEL7中的web服务

软件包:httpd

系统服务: httpd

提供的默认配置:

Listen: 监听地址:端口(80)

ServerName: 本站点注册的DNS名称(空缺)

DocumentRoot: 设置网页文件存放的**根目录**(默认是/var/www/html)

DirectoryIndex: 起始页/首页文件名(index.html)

DNS服务器: 虚拟机classroom, 只能解析下面三种域名:

www0.example.com

server0.example.com

webapp0.example.com

测试DNS解析：nslookup 域名

[root@server0 ~]# nslookup server0.example.com

[root@server0 ~]# nslookup www0.example.com

[root@server0 ~]# nslookup webapp0.example.com

虚拟机server0

1 建立新的网页文件存放路径

[root@server0 ~]# mkdir /var/www/myweb

[root@server0 ~]# echo '<h1>wo shi myweb' > /var/www/myweb/index.html

[root@server0 ~]# cat /var/www/myweb/index.html

<h1>wo shi myweb

2 修改配置文件 /etc/httpd/conf/httpd.conf -----主配置文件

[root@server0 ~]# vim /etc/httpd/conf/httpd.conf

DocumentRoot "/var/www/html" #双引号可有可无

DocumentRoot "/var/www/myweb"

3 重启httpd服务

[root@server0 ~]# systemctl restart httpd

4 虚拟机desktop0上验证

[root@desktop0 ~]# firefox 172.25.0.11



**网络路径与服务器实际存放路径**

网络路径：在浏览器输入的路径

实际路径：服务器存放网页文件的路径

网络路径: [root@desktop0 ~]# firefox 172.25.0.11

服务器实际存放路径: DocumentRoot /var/www/myweb

网络路径: [root@desktop0 ~]# firefox 172.25.0.11/nsd

服务器实际存放路径: DocumentRoot /var/www/myweb/nsd

firefox 172.25.0.11 ----->服务端----->httpd----->DocumentRoot

----->/var/www/myweb

虚拟机server0

[root@server0 ~]# mkdir /var/www/myweb/nsd

[root@server0 ~]# echo '<h1>wo shi nsd' > /var/www/myweb/nsd/index.html

虚拟机desktop0

[root@desktop0 ~]# firefox 172.25.0.11/nsd



**虚拟web主机**

作用：一台服务器可以提供多个不同的Web站点

区分方式:

1. 基于域名的虚拟主机

2) 基于端口的虚拟主机

3) 基于IP地址的虚拟主机 (已经淘汰)

配置一个虚拟站点

配置文件路径:

1) /etc/httpd/conf/httpd.conf 主配置文件

2) /etc/httpd/conf.d/\*.conf 调用配置文件

为每个虚拟站点添加配置

<VirtualHost IP地址:端口>

ServerName 此站点的DNS名称

DocumentRoot 此站点的网页根目录

</VirtualHost>

**虚拟web主机会由上及下进行匹配,写在上面的优先级高,一旦使用了虚拟web主机功能,所有的站点都必须要使用虚拟web主机功能呈现**

虚拟机server0:

1 在新的调用配置文件中书写

[root@server0 ~]# vim /etc/httpd/conf.d/nsd01.conf

<VirtualHost \*:80> #在所有IP上80端口,启用虚拟web主机

ServerName www0.example.com #指定网站访问域名

DocumentRoot /var/www/baidu #指定存放网页文件路径

</VirtualHost>

<VirtualHost \*:80>

ServerName webapp0.example.com

DocumentRoot /var/www/qq

</VirtualHost>

<VirtualHost \*:80>

ServerName server0.example.com

DocumentRoot /var/www/myweb

</VirtualHost>

[root@server0 ~]# systemctl restart httpd

[root@server0 ~]# mkdir /var/www/qq /var/www/baidu

[root@server0 ~]# echo '<h1>百度' > /var/www/baidu/index.html

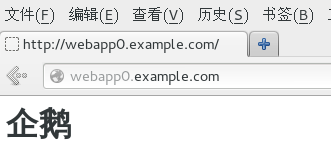
[root@server0 ~]# echo '<h1>企鹅' > /var/www/qq/index.html

虚拟机desktop0

[root@desktop0 ~]# firefox www0.example.com



[root@desktop0 ~]# firefox webapp0.example.com



[root@desktop0 ~]# firefox server0.example.com



**网页内容访问**

**配置目录访问**

文件夹权限: 针对DocumentRoot网页目录的权限控制

1) httpd 运行身份(用户/组):apache

2) 能够提取哪些网页资源

客户机地址限制

使用<Directory>配置区段

1) 每个文件夹自动**继承其父目录的ACL**访问权限

2) 除非针对子目录有明确设置

<Directory 目录的绝对路径>

.. ..

Require all denied或granted

Require ip或网段地址.. ..

</Directory>

禁止任何客户机访问

<Directory />

Require all denied

</Directory>

允许任何客户机访问

<Directory /var/www/html>

Require all granted

</Directory>

仅允许部分客户机访问

<Directory /var/www/html/private>

Require ip 172.25.0.11

</Directory>

案例3:配置网页内容访问

在 Web 网站 http://server0.example.com 的

DocumentRoot 目录下创建一个名为 private 的子目录

要求如下:

从server0上, 任何人都可以浏览private的内容,但是从其他系统不能访问这个目录的内容

1 查看DocumentRoot路径

[root@server0 ~]# cat /etc/httpd/conf.d/nsd01.conf

2 书写页面文件

[root@server0 ~]# mkdir /var/www/myweb/private

[root@server0 ~]# echo '<h1>wo shi private' > /var/www/myweb/private/index.html

3 测试验证

[root@desktop0 ~]# firefox server0.example.com/private



[root@server0 ~]# vim /etc/httpd/conf.d/nsd02.conf

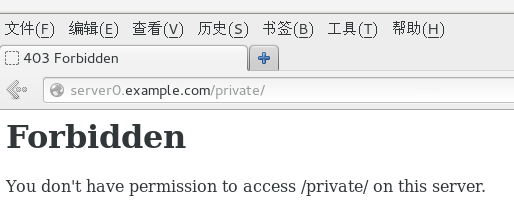
<Directory /var/www/myweb/private> #针对路径进行访问配置

require ip 172.25.0.11 #仅允许172.25.0.11

</Directory>

[root@server0 ~]# systemctl restart httpd

[root@desktop0 ~]# firefox server0.example.com/private



案例4: 使用自定web根目录

调整web站点http://server0.example.com的网页目录,要求如下:

1) 新建目录/webroot , 作为此站点新的网页文件根目录

2) 确保站点http://server0.example.com 仍然可以访问

1 新建目录/webroot,写入页面文件

[root@server0 ~]# mkdir /webroot

[root@server0 ~]# echo '<h1>wo shi webroot' > /webroot/index.html

2 配置/webroot,作为此站点新的网页文件根目录

[root@server0 ~]# vim /etc/httpd/conf.d/nsd01.conf

......

<VirtualHost \*:80>

ServerName server0.example.com

DocumentRoot /webroot #修改网页文件根目录

</VirtualHost>

[root@server0 ~]# vim /etc/httpd/conf.d/nsd02.conf

<Directory /webroot> #针对/webroot进行访问配置

require all granted #允许所有人访问

</Directory>

[root@server0 ~]# systemctl restart httpd

3 修改selinux策略(安全上下文----标签)

添加新的web目录

方式1: 参照标准目录,重设新目录的属性

chcon [-R] --reference=模板目录 新目录

方式2: 将新目录增加到预设的标准web目录范围 (后面了解)

semanage focntext -a -t httpd\_sys\_content\_t ‘新目录(/.\*)?’

[root@server0 ~]# ls -Zd /var/www #查看目录上下文值(-Z 大写)

[root@server0 ~]# ls -Zd /webroot #查看目录上下文值

[root@server0 ~]# chcon -R --reference=/var/www /webroot/

[root@server0 ~]# ls -Zd /webroot

drwxr-xr-x. root root system\_u:object\_r:httpd\_sys\_content\_t:s0 /webroot

4 验证测试

[root@desktop0 ~]# firefox server0.example.com



**部署动态网站**

LAMP: Linux Apache MariaDB/MySQL PHP/Python

LNMP: Linux Nginx MariaDB/MySQL PHP/Python

静态网站的运行

服务端的原始网页=浏览器访问到的网页

1) 由web服务软件处理所有请求

2) 文本(txt/html) 、图片(jpg/png)等静态资源

动态网站的运行

服务端的原始网页≠浏览器访问到的网页

1) 由web服务端软件接受请求,动态程序转后端模块处理

2) PHP网页、Python网页、JSP网页.....

案例: 部署并测试WSGI站点

为站点webapp0.example.com配置提供动态web内容,要求如下:

1) 此虚拟机主机侦听在端口8909

2) 测试网页从以下地址下载,不要做任何修改

http://classroom.example.com/pub/materials/webinfo.wsgi

3) 从浏览器访问http://webapp0.example.com:8909, 可接收到动态生成的web页面

4) 此站点必须能被example.com域内的所有系统访问

虚拟机server0:

部署动态web

1 为站点web0.example.com配置提供动态web内容

[root@server0 ~]# cat /etc/httpd/conf.d/nsd01.conf

[root@server0 ~]# cd /var/www/qq

[root@server0 qq]# wget http://classroom.example.com/pub/materials/webinfo.wsgi

2 方便客户端用户访问动态页面,页面跳转(页面别名/地址重写)

配置: Alias 网络路径 实际服务器本地路径

例子: Alias / /var/www/qq/webinfo.wsgi

当客户端访问网页文件根目录时，页面进行跳转到/var/www/qq/webinfo.wsgi

[root@server0 qq]# vim /etc/httpd/conf.d/nsd01.conf

.......

<VirtualHost \*:80>

ServerName webapp0.example.com

DocumentRoot /var/www/qq

Alias / /var/www/qq/webinfo.wsgi

</VirtualHost>

[root@server0 qq]# systemctl restart httpd

[root@server0 qq]# firefox webapp0.example.com #测试验证,网络页面为源代码

3 安装软件mod\_wsgi, 翻译Python页面的代码

[root@server0 qq]# yum -y install mod\_wsgi

[root@server0 qq]# vim /etc/httpd/conf.d/nsd01.conf

......

<VirtualHost \*:80>

ServerName webapp0.example.com

DocumentRoot /var/www/qq

WsgiScriptAlias / /var/www/qq/webinfo.wsgi

</VirtualHost>

[root@server0 qq]# systemctl restart httpd

[root@server0 qq]# firefox webapp0.example.com



Unix时间戳:从1970-1-1 0:0:0开始计算,到现在时间经历的秒数

4 此虚拟主机侦听在端口8909

[root@server0 qq]# vim /etc/httpd/conf.d/nsd01.conf

......

Listen 8909 #让httpd程序监听8909

<VirtualHost \*:8909> #该虚拟web主机利用8909提供

ServerName webapp0.example.com

DocumentRoot /var/www/qq

WsgiScriptAlias / /var/www/qq/webinfo.wsgi

</VirtualHost>

5 修改selinux策略(非默认端口的开放)

[root@server0 qq]# semanage port -l | grep http

http\_cache\_port\_t tcp 8080, 8118, 8123, 10001-10010

http\_cache\_port\_t udp 3130

http\_port\_t tcp 80, 81, 443, 488, 8008, 8009, 8443, 9000

pegasus\_http\_port\_t tcp 5988

pegasus\_https\_port\_t tcp 5989

[root@server0 qq]# semanage port -a -t http\_port\_t -p tcp 8909 #添加新的端口 -a : 添加 -t: 类型 -p: 协议

[root@server0 qq]# semanage port -l | grep http | head -3 | tail -1

http\_port\_t tcp 8909, 80, 81, 443, 488, 8008, 8009, 8443, 9000

[root@server0 qq]# systemctl restart httpd

[root@server0 qq]# firefox server0.example.com:8909 #访问成功



web服务:

端口的优先级最高,优于域名,也优于IP地址

虚拟Web由上及下的匹配，在最上面的优先级最高

#########################################

**补充:**

破解密码:

重启电脑,进入grub2引导界面,按e键进入编辑模式,找到Linux16所在的行,在末尾添加 rd.break console=tty0 # 注意空格,然后按ctrl+x 快捷键进入救援模式

在救援模式中输入:

mount -o remount,rw sysroot

chroot /sysroot

echo 密码 | passwd --stdin root #密码只是提示,可自定义

touch /.autorelabel

exit

exit/reboot

添加一个swap分区

在您的系统中添加一个大小为512M的swap分区:

1. 当您的系统启动时,swap分区应该可以自动挂载
2. 不要移除或者修改其他存在于您的系统中的swap分区

[root@server0 ~]# fdisk /dev/vdb

[root@server0 ~]# mkswap /dev/vdb1 #格式化文件系统swap

[root@server0 ~]# vim /etc/fstab

/dev/vdb1 swap swap defaults 0 0 #不用自己创建挂载点,默认swap

[root@server0 ~]# swapon -a #开机自动挂载

[root@server0 ~]# swapon -s #查看挂载情况

文件名 类型 大小 已用 权限

/dev/vdb1 partition 524284 0 -1

[root@server0 ~]# swapoff /dev/vdb1 #卸载swap挂载

[root@server0 ~]# swapon -s

系统 classroom.example.com 提供了一个 LDAP 验证服务。您的系统需要按照以下要求绑定到这个服务上：

1) 验证服务器的基本 DN 是：dc=example,dc=com

2) 帐户信息和验证信息都是由 LDAP 提供的

3) 连接要使用证书进行加密，证书可以在下面的链接中下载 ：<http://classroom.example.com/pub/example-ca.crt>

4) 当正确完成配置后，用户 ldapuser0 应该能够登录到您的系统中，但是没有主目录。当您完成 autofs的题目之后，才能生成主目录

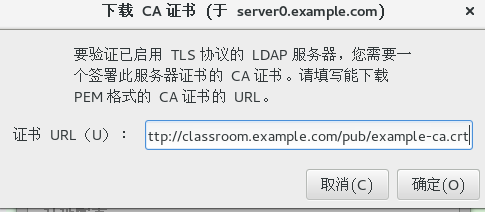
5) 用户ldapuser0的密码是password

[root@server0 ~]# yum -y install sssd

[root@server0 ~]# yum -y install authconfig-gtk

[root@server0 ~]# authconfig-gtk





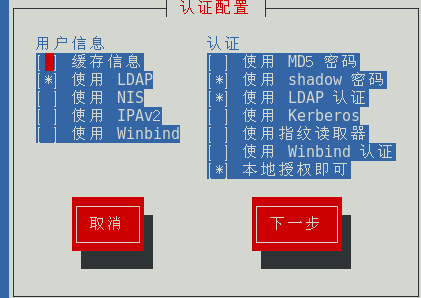
[root@server0 ~]# systemctl restart sssd

[root@server0 ~]# systemctl enable sssd

**法二**

[root@server0 ~]# yum -y install sssd

[root@server0 ~]# authconfig-tui







[root@server0 ~]# cd /etc/openldap/cacerts

[root@server0 cacerts]# wget http://classroom.example.com/pub/example-ca.crt

[root@server0 ~]# systemctl restart sssd

[root@server0 ~]# systemctl enable sssd

[root@server0 ~]# id ldapuser0

uid=1700(ldapuser0) gid=1700(ldapuser0) 组=1700(ldapuser0)

按照下述要求配置autofs用来自动挂载 LDAP 用户的主目录：

1) classroom.example.com（172.25.0.11）通过 NFS 输出 /home/guests 目录到您的系统，这个文件系统包含了用户ldapuser0的主目录，并且已经预先配置好了

2) ldapuser0用户的主目录是 classroom.example.com:/home/guests/ldapuser0

3) ldapuser0的主目录应该按需挂载到本地的/home/guests/ldapuser0 目录下

4) 用户对其主目录必须是可写的

5) ldapuser0用户的密码是password

[root@server0 ~]# yum -y install autofs

[root@server0 ~]# vim /etc/auto.master

......

/home/guests /etc/auto.guests

.......

[root@server0 ~]# vim /etc/auto.guests

ldapuser0 -rw classroom.example.com:/home/guests/ldapuser0

[root@server0 ~]# systemctl restart autofs

[root@server0 ~]# systemctl enable autofs

[root@server0 ~]# su - ldapuser0 -c pwd

/home/guests/ldapuser0

**注意**: ldap服务必须完成之后才能完成autofs服务

autofs:触发式挂载

用户:

1. 当用户访问目录时,自动挂载
2. 当用户目录闲置时,则自动卸载

**实现思路:**

1. 首先创建挂载点b目录的父目录a
2. 请autofs监控父目录a,当有人访问b目录时,自动创建b目录并挂载
3. 当b目录无人访问时(系统默认时间300秒),则自动卸载并删除b目录

1 装包 yum -y install autofs

2 配置

vim /etc/auto.master #主配置文件

/home/guests(被监控的目录) /etc/auto.guests(挂载策略文件)

vim /etc/auto.guests #挂载策略文件

ldapuser0 -rw classroom.example.com:/home/guests/ldapuser0

挂载的子目录 挂载参数 设备路径

或者

\* -rw classroom.example.com:/home/guests/&

3 启动服务

systemctl start autofs

systemctl enable autofs

4 验证

su - ldapuser0 -c pwd

**yum搭建**

[root@server0 ~]# yum-config-manager --add http://content.example.com/rhel7.0/x86\_64/dvd

[root@server0 ~]# echo 'gpgcheck=0' >> /etc/yum.repos.d/content.example.com\_rhel7.0\_x86\_64\_dvd.repo

########################################

**复习:**

1 vim编辑器命令模式,删除光标之后如何操作?

C(大写的C) 或者 d$

2 开机挂载配置文件与指定DNS服务器配置文件,请写出路径?

/etc/fstab /etc/resolv.conf

3 Samba文件系统?

cifs

4 配置nfs共享,配置文件在哪里?

/etc/exports

5 yum客户端配置文件放在什么路径下? 具体都有哪些字段?

/etc/yum.repos.d/\*.repo

[] name baseurl enabled gpgcheck

6 配置Apache的http服务时,虚拟web主机区分方式都有哪些?

基于域名 基于端口 基于IP地址

7 配置Apache的http服务,指定网页文件根目录配置字段是什么?

DocumentRoot

8 httpd服务进行虚拟web主机配置,具体格式是什么?

<VirtualHost \*:80>

ServerName

DocumentRoot

</VirtualHost>

9 请您写出您熟知的Linux命令,至少10 条以上

尽量写多点 , 命令不要过于简单

#########################################

**I-Day14**

设置防火墙默认区域

[root@server0 ~]# firewall-cmd --set-default-zone=trusted

[root@desktop0 ~]# firewall-cmd --set-default-zone=trusted

**安全web服务(https 安全的超文本传输协议 默认端口是443)**

PKI (Public Key Infrastructure) 公钥基础设施

– 公钥:主要用来加密数据

– 私钥:主要用来解密数据(与相应的公钥匹配)

– 数字证书:证明拥有者的合法性/权威性(单位名称、有效期、公钥、颁发机构及

签名、......)

– Certificate Authority,数字证书授权中心:负责证书的申请/审核/颁发/鉴定/撤销等管理工作

公钥与私钥是非对称加密

部署安全web服务器

虚拟接server0

1 安装httpd软件

[root@server0 ~]# yum -y install httpd

2 部署网站证书(营业执照)

[root@server0 ~]# cd /etc/pki/tls/certs #一定要下载到此目录下

[root@server0 certs]# wget http://classroom.example.com/pub/tls/certs/server0.crt

3 部署根证书(工商局信息)

[root@server0 ~]# cd /etc/pki/tls/certs #一定要下载到此目录下

[root@server0 certs]# wget http://classroom.example.com/pub/example-ca.crt

4 部署解密的私钥

[root@server0 certs]# cd /etc/pki/tls/private/ #一定要下载到此目录下

[root@server0 private]# wget http://classroom.example.com/pub/tls/private/server0.key

5 安装mod\_ssl软件,支持安全加密的web软件

[root@server0 private]# yum -y install mod\_ssl

[root@server0 private]# rpm -ql mod\_ssl

/etc/httpd/conf.d/ssl.conf

/etc/httpd/conf.modules.d/00-ssl.conf

/usr/lib64/httpd/modules/mod\_ssl.so

/usr/libexec/httpd-ssl-pass-dialog

/var/cache/httpd/ssl

6 修改配置文件

[root@server0 private]# vim /etc/httpd/conf.d/ssl.conf

在末行模式 :set nu #开启行号功能

......

59 DocumentRoot "/var/www/html" #去掉注释

60 ServerName www0.example.com:443 #去掉注释

......

100 SSLCertificateFile /etc/pki/tls/certs/localhost.crt

100 SSLCertificateFile /etc/pki/tls/certs/server0.crt #指定网站证书

107 SSLCertificateKeyFile /etc/pki/tls/private/localhost.key

107 SSLCertificateKeyFile /etc/pki/tls/private/server0.key #指定解密的私钥文件

122 SSLCACertificateFile /etc/pki/tls/certs/ca-bundle.crt

122 SSLCACertificateFile /etc/pki/tls/certs/example-ca.crt #指定根证书(工商局信息)

7 重启httpd服务

[root@server0 private]# systemctl restart httpd

[root@server0 private]# echo '<h1>NSD1811 Web' > /var/www/html/index.html

8 测试验证

[root@server0 private]# firefox <https://www0.example.com>



点击--->我已了解可能的风险---->添加例外安全 --->确认安全例外



**基础邮件服务**

电子邮件服务器的基本功能

1) 为用户提供电子邮箱存储空间(用户名@邮件域名)

2) 处理用户发出的邮件 —— 传递给收件服务器

3) 处理用户收到的邮件 —— 投递到邮箱

SMTP: 用户发邮件协议------默认端口25

pop3: 用户收邮件协议-------默认端口110

DNS服务器: 虚拟机classroom

虚拟机server0: 搭建邮件服务器

1 安装软件包postfix

[root@server0 ~]# yum -y install postfix #默认已经安装

2 修改主配置文件 /etc/postfix/main.cf

[root@server0 ~]# vim /etc/postfix/main.cf

99 #myorigin = $mydomain

99 myorigin = server0.example.com #收件人与发件人补全的域名后缀

116 inet\_interfaces = localhost

116 inet\_interfaces = all #本机所有网络接口都提供邮件功能

164 mydestination = $myhostname, localhost.$mydomain, localhost

164 mydestination = server0.example.com #判定收件人域名后缀 为本域邮件

3 重启服务postfix

[root@server0 ~]# systemctl restart postfix

4 创建普通用户测试邮件收发

[root@server0 ~]# useradd yg

[root@server0 ~]# useradd xln

5 使用mail命令发信/收信

mail 发信操作

格式: mail -s '邮件标题' -r 发件人 收件人

mail 收信操作

格式: mail [-u 用户名]

[root@server0 ~]# mail -s 'test01' -r yg xln

今天天气很好

. #点’.’代表提交

EOT

[root@server0 ~]# mail -u xln

Heirloom Mail version 12.5 7/5/10. Type ? for help.

"/var/mail/xln": 1 message 1 new

>N 1 yg@server0.example.c Mon Dec 17 14:18 18/584 "test01"

& 1 #输入邮件的编号,进行查看邮件内容

Message 1:

From yg@server0.example.com Mon Dec 17 14:18:12 2018

Return-Path: <yg@server0.example.com>

X-Original-To: xln

Delivered-To: xln@server0.example.com

Date: Mon, 17 Dec 2018 14:18:12 +0800

From: yg@server0.example.com

To: xln@server0.example.com

Subject: test01

User-Agent: Heirloom mailx 12.5 7/5/10

Content-Type: text/plain; charset=utf-8

Status: R

今天天气很好

6 非交互式发送邮件

[root@server0 ~]# echo 今天天气真的非常好 | mail -s 'test02' -r yg xln

[root@server0 ~]# mail -u xln

Heirloom Mail version 12.5 7/5/10. Type ? for help.

"/var/mail/xln": 2 messages 2 new

>N 1 yg@server0.example.c Mon Dec 17 14:18 18/584 "test01"

N 2 yg@server0.example.c Mon Dec 17 14:20 18/593 "test02"

& 2 #输入邮件的编号,进行查看邮件内容

Message 2:

From yg@server0.example.com Mon Dec 17 14:20:22 2018

Return-Path: <yg@server0.example.com>

X-Original-To: xln

Delivered-To: xln@server0.example.com

Date: Mon, 17 Dec 2018 14:20:22 +0800

From: yg@server0.example.com

To: xln@server0.example.com

Subject: test02

User-Agent: Heirloom mailx 12.5 7/5/10

Content-Type: text/plain; charset=utf-8

Status: R

今天天气真的非常好

**parted分区工具**

使用fdisk操作>2.2TB的磁盘时, 超出容量的磁盘将会无法识别,导致分区失败

GPT: 最多可以划分128个主分区,最大空间支持到18EB容量

1EB=1000PB

1PB=1000TB

1TB=1000GB

1Tib=1024Gib

[root@server0 ~]# parted /dev/vdb

(parted) mktable gpt #指定分区模式

(parted) mkpart #划分新的分区

分区名称？ []? primary1 #指定分区的名称,随意写

文件系统类型？ [ext2]? ext4 #分区文件系统类型,不起实际作用

起始点？ 0

结束点？ 2G

警告: The resulting partition is not properly aligned for best performance.

忽略/Ignore/放弃/Cancel? ignore #忽略硬盘存储分区表空间

(parted) print

Model: Virtio Block Device (virtblk)

Disk /dev/vdb: 10.7GB

Sector size (logical/physical): 512B/512B

Partition Table: gpt

Disk Flags:

Number Start End Size File system Name 标志

1 17.4kB 2000MB 2000MB primary1

(parted) unit GB #使用单位为GB

(parted) print

Model: Virtio Block Device (virtblk)

Disk /dev/vdb: 10.7GB

Sector size (logical/physical): 512B/512B

Partition Table: gpt

Disk Flags:

Number Start End Size File system Name 标志

1 0.00GB 2.00GB 2.00GB primary1

(parted) mkpart

分区名称？ []? primary2

文件系统类型？ [ext2]? ext4

起始点？ 2G #起始点为2G,为上一个分区的结束

结束点？ 4G

(parted) print

Model: Virtio Block Device (virtblk)

Disk /dev/vdb: 10.7GB

Sector size (logical/physical): 512B/512B

Partition Table: gpt

Disk Flags:

Number Start End Size File system Name 标志

1 0.00GB 2.00GB 2.00GB primary1

2 2.00GB 4.00GB 2.00GB primary2

(parted) quit #退出分区

信息: You may need to update /etc/fstab.

[root@server0 ~]# lsblk

NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT

vda 253:0 0 10G 0 disk

└─vda1 253:1 0 10G 0 part /

vdb 253:16 0 10G 0 disk

├─vdb1 253:17 0 1.9G 0 part

└─vdb2 253:18 0 1.9G 0 part

**什么是交换空间**

• 相当于虚拟内存

– 当物理内存不够用时,使用磁盘空间来模拟内存

– 在一定程度上缓解内存不足的问题

– 交换分区:以空闲分区充当的交换空间

1 格式化交换文件系统

[root@server0 ~]# mkswap /dev/vdb1

正在设置交换空间版本 1，大小 = 1953104 KiB

无标签，UUID=9617779e-dab2-47bf-ab83-b199bbf2264b

[root@server0 ~]# mkswap /dev/vdb2

正在设置交换空间版本 1，大小 = 1952764 KiB

无标签，UUID=b00f55e1-2cea-48a9-b995-9f82ecb079e8

[root@server0 ~]# blkid /dev/vdb1

[root@server0 ~]# blkid /dev/vdb2

2 启用交换分区

[root@server0 ~]# swapon /dev/vdb1

[root@server0 ~]# swapon /dev/vdb2

[root@server0 ~]# swapon -s #查看组成交换空间成员信息

3 停用交换分区

[root@server0 ~]# swapoff /dev/vdb1

[root@server0 ~]# swapoff /dev/vdb2

[root@server0 ~]# swapon -s

4 开机自动启用交换分区

[root@server0 ~]# vim /etc/fstab

/dev/vdb1 swap swap defaults 0 0

/dev/vdb2 swap swap defaults 0 0

[root@server0 ~]# swapon -a #专用于检测交换分区是否书写正确的指令

[root@server0 ~]# swapon -s

**配置聚合连接(链路聚合/网卡组队/网卡绑定)**

采用虚拟网卡设计,对网卡设备的备份

链路聚合的优势

• team,聚合连接(也称为链路聚合)

– 由多块网卡(team-slave)一起组建而成的虚拟网卡,即“组队”

– 作用1:轮询式(roundrobin)的流量负载均衡-------以后讲

– 作用2:热备份(activebackup)连接冗余

1 创建虚拟网卡team0 参考: man teamd.conf #全文搜索/example

[root@server0 ~]# nmcli connection add type team ifname team0 con-name team0 autoconnect yes config '{"runner": {"name": "activebackup"}}'

nmcli connection 添加 类型 team(组队) 网卡名 team0 配置文件名 team0 每次开机自动启用 配置运行模式 热备份模式

ifname=ifconfig name con-name=config-name

[root@server0 ~]# ifconfig #查看team0网卡信息

[root@server0 ~]# cat /etc/sysconfig/network-scripts/ifcfg-team0

# cat /et(tab)/sysco(tab)/netw(tab)-(tab)/ifcfg-team0

生成的网卡配置文件信息

如果有敲错，一定要删除: nmcli connection delete team0

2 添加成员

[root@server0 ~]# nmcli connection add type team-slave ifname eth1 con-name team0-1 master team0

[root@server0 ~]# nmcli connection add type team-slave ifname eth2 con-name team0-2 master team0

解析: nmcli connection 添加 类型为 team的成员 网卡为eth1 配置文件名 team0-1 主设备为team0

如果有敲错，一定要删除: nmcli connection delete team0-1

如果有敲错，一定要删除: nmcli connection delete team0-2

3 配置team0的IP地址

[root@server0 ~]# nmcli connection modify team0 ipv4.method manual ipv4.addresses 192.168.1.1/24 connection.autoconnect yes

[root@server0 ~]# nmcli connection up team0 #激活配置

[root@server0 ~]# ifconfig

专用于显示team信息命令

[root@server0 ~]# teamdctl team0 state

[root@server0 ~]# ifconfig eth1 down

[root@server0 ~]# ifconfig eth1 up

#########################################

**I-Day15**

**搭建新的教学环境**

真机上运行clone-vm7可以快速产生虚拟机(clone-vm7是达内写的shell脚本,系统本身不自带)

1 利用clone-vm7命令产生两台新的虚拟机

[root@room9pc01 ~]# clone-vm7

Enter VM number: 3

Creating Virtual Machine disk image...... [OK]

Defining new virtual machine...... [OK]

[root@room9pc01 ~]# clone-vm7

Enter VM number: 4

Creating Virtual Machine disk image...... [OK]

Defining new virtual machine...... [OK]

2 修改虚拟机的名称,修改为A与B

虚拟机A: 服务端 虚拟机B: 客户端

3 虚拟机配置 root密码为123456

虚拟机A: 服务端

1. 配置永久主机名:svr7.tedu.cn
2. 为eth0配置永久的IP地址: 192.168.4.7/24

虚拟机B: 客户端

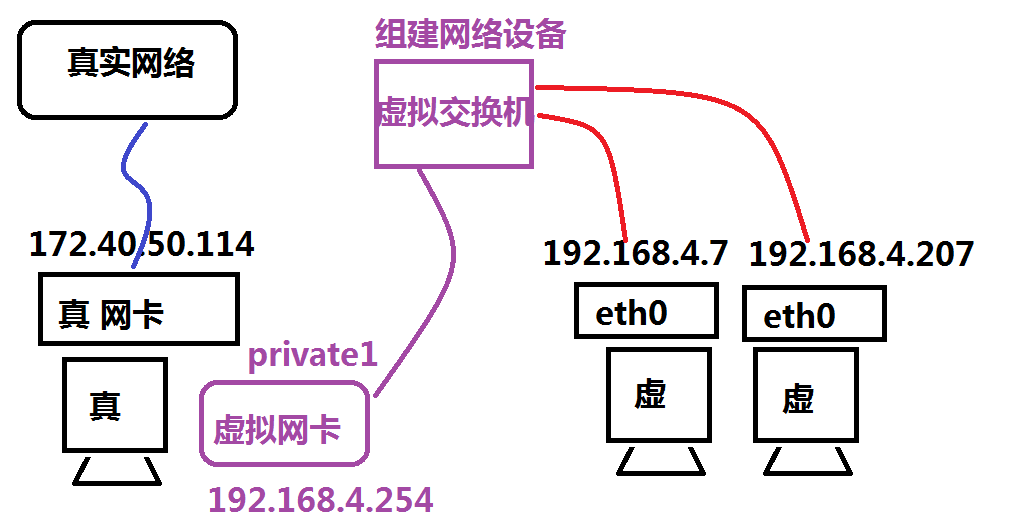
1) 配置永久主机名:pc207.tedu.cn

2) 为eth0配置永久的IP地址: 192.168.4.207/24

4 真机进行远程管理

[root@room9pc01 ~]# ping 192.168.4.7

[root@room9pc01 ~]# ping 192.168.4.207



[root@room9pc01 ~]# ifconfig private1 | head -2

private1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500

inet 192.168.4.254 netmask 255.255.255.0 broadcast 192.168.4.255

5 为真机制作永久的别名

[root@room9pc01 ~]# vim /root/.bashrc

alias goa='ssh -X root@192.168.4.7'

alias gob='ssh -X root@192.168.4.207'

6 搭建网络yum仓库

服务端: 真机搭建vsftpd服务(FTP服务),共享光盘所有内容

1) 安装软件包vsftpd

[root@room9pc01 ~]# rpm -q vsftpd #默认已经安装

vsftpd-3.0.2-22.el7.x86\_64

2) 利用FTP服务共享光盘内容,给网络中所有虚拟机 #默认已经装好

[root@room9pc01 ~]# mkdir /var/ftp/rhel7/

[root@room9pc01 ~]# ls -l /iso

lrwxrwxrwx 1 root root 27 1月 23 2018 /iso -> /var/lib/libvirt/images/iso

[root@room9pc01 ~]# ls /var/lib/libvirt/images/iso/

[root@room9pc01 ~]# vim /etc/fstab #文件内容不支持快捷方式

/var/lib/libvirt/images/iso/rhel-server-7.4-x86\_64-dvd.iso /var/ftp/rhel7 iso9660 defaults 0 0

[root@room9pc01 ~]# ls /var/ftp/rhel7/

[root@room9pc01 ~]# firefox ftp://192.168.4.254/rhel7

客户端: 所有新创建的虚拟机

1) 虚拟机A 操作:

[root@svr7 ~]# vim /etc/yum.repos.d/rhel7.repo

[rhel7]

name=rhel7.4

baseurl=ftp://192.168.4.254/rhel7

enabled=1

gpgcheck=0

[root@svr7 ~]# yum repolist

[root@svr7 ~]# yum -y install httpd

2) 虚拟机B操作: 同上虚拟机A操作

**扩展的几个应用**

**目录结构**

认识Linux的目录层次:

格式: man hier

常见一级目录的用途

|  |  |
| --- | --- |
| **目录路径** | **主要用途** |
| /boot | 存放系统引导必需的文件,包括内核、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

1) 基本权限:r = 4, w = 2, x = 1

2) 附加权限:SUID = 4, SGID = 2, Sticky Bit = 1

采用数值形式设置权限

基本权限格式: chmod [-R] nnn 文档...

附加权限格式: chmod [-R] xnnn 文档...

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **分组** | User权限 | | | Group权限 | | | Other权限 | | |
| **字符** | r | w | x | r | - | x | r | - | x |
| **数字** | 4 | 2 | 1 | 4 | 0 | 1 | 4 | 0 | 1 |
| **求和** | 7 | | | 5 | | | 5 | | |

**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

[root@svr7 ~]# vim /etc/profile #一般需要重启才能生效

HISTSIZE=1000 #默认记录1000条

[root@svr7 ~]# !cat #显示最近以cat开头的历史命令

[root@svr7 ~]# !ls #显示最近以ls开头的历史命令

**实用小目命令工具**

du, 统计文件的占用空间

格式: 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)

[root@svr7 ~]# du /boot/

0 /boot/efi/EFI/redhat

0 /boot/efi/EFI

0 /boot/efi

2400 /boot/grub2/i386-pc

3176 /boot/grub2/locale

2504 /boot/grub2/fonts

8096 /boot/grub2

130840 /boot/

[root@svr7 ~]# du -s /boot

130840 /boot

[root@svr7 ~]# du -sh /boot

128M /boot

[root@svr7 ~]# du -sh /boot /etc/pki

128M /boot

1.4M /etc/pki

**date,查看/调整系统日期时间**

date +%F 展现年月日

date +%R 展现时分

date +%T 展现时分秒

date +"%Y-%m-%d %H:%M:%S"

date -s "yyyy-mm-dd HH:MM:SS" #修改时间

[root@desktop0 opt]# date +%F

2019-01-12

[root@desktop0 opt]# date +%R

14:58

[root@desktop0 opt]# date +%T

14:58:24

**ln 创建软连接(快捷方式)**

格式: ln -s 原始文件或目录 软连接文件 #必须写绝对路径

1) 若原始文件或目录被删除,连接文件将失效

2) 软连接可存放在不同分区/文件系统,支持目录

[root@svr7 ~]# cat /etc/redhat-release

Red Hat Enterprise Linux Server release 7.4 (Maipo)

[root@svr7 ~]# ln -s /etc/redhat-release /opt/rr

[root@svr7 ~]# ls /opt/

rh rr

[root@svr7 ~]# ls -l /opt/rr

lrwxrwxrwx. 1 root root 19 12月 18 14:23 /opt/rr -> /etc/redhat-release

[root@svr7 ~]# cat /opt/rr

Red Hat Enterprise Linux Server release 7.4 (Maipo)

**ln 创建硬链接(快捷方式)**

格式: ln 原始文件 硬链接文件

1. 若原始文件被删除,连接文件仍可用

2) 硬连接与原始文件必须在同一分区/文件系统,不支持目录

[root@svr7 ~]# rm -rf /opt/\*

[root@svr7 ~]# echo 123 > /opt/A

[root@svr7 ~]# ln -s /opt/A /opt/A1 #制作软连接

[root@svr7 ~]# ln /opt/A /opt/A2 #制作硬链接

[root@svr7 ~]# ls /opt

A A1 A2

[root@svr7 ~]# cat /opt/A1

123

[root@svr7 ~]# cat /opt/A2

123

[root@svr7 ~]# rm -rf /opt/A

[root@svr7 ~]# cat /opt/A1

cat: /opt/A1: 没有那个文件或目录

[root@svr7 ~]# cat /opt/A2

123

**获取命令的帮助信息**

对于内部命令程序,使用”help 命令名” 查看

对于外部命令程序,通常 会提供“--help”帮助选项

**5: 配置文件的帮助信息**

[root@svr7 ~]# man 5 passwd

**zip归档工具(跨平台的压缩/归档格式)**

归档+压缩操作

格式: zip [-r] 备份文件.zip 被归档的文档...

释放归档+解压操作

格式: unzip 备份文件.zip [-d目标文件夹]

[root@svr7 ~]# rm -rf /opt/\*

[root@svr7 ~]# zip -r /opt/nsd01.zip /etc/passwd /boot /root/

[root@svr7 ~]# zip -r /opt/nsd02.zip /etc/fstab /etc/resolv.conf

[root@svr7 ~]# ls /opt

nsd01.zip nsd02.zip

[root@svr7 ~]# unzip /opt/nsd02.zip -d /mnt/

[root@svr7 ~]# ls /mnt/

etc

**自定义yum仓库: 将自己从互联网下载的软件包,建立仓库**

1 获取互联网下载的软件包tools.tar.gz

使用scp命令在虚拟机和物理机之间传输文件

格式: scp 源文件 root@客户机IP:路径

[root@room9pc01 ~]# scp /root/桌面/tools.tar.gz [root@192.168.4.7:/root/](mailto:root@192.168.4.7:/root/) #真机上操作

root@192.168.4.7's password:

tools.tar.gz 100% 766KB 111.3MB/s 00:00

2 虚拟机A上查看

[root@svr7 ~]# ls /root

3 解包

[root@svr7 ~]# tar -xf /root/tools.tar.gz -C /

[root@svr7 ~]# ls /

[root@svr7 ~]# ls /tools

inotify-tools-3.13.tar.gz other

[root@svr7 ~]# ls /tools/other

4 创建仓库数据文件(描述仓库所有包的数量及信息)

[root@svr7 ~]# createrepo /tools/other/ #创建仓库数据文件

[root@svr7 ~]# ls /tools/other/

5 书写客户端配置文件

[root@svr7 ~]# vim /etc/yum.repos.d/rhel7.repo

......

[others]

name=my rmp

baseurl=file:///tools/other #指定本机为yum仓库

enabled=1

gpgcheck=0

[root@svr7 ~]# yum repolist

[root@svr7 ~]# yum -y install sl

[root@svr7 ~]# yum -y install cmatrix

[root@svr7 ~]# yum -y install oneko

[root@svr7 ~]# oneko & #放入后台运行

[root@svr7 ~]# killall oneko #杀死oneko程序

**注意: 客户端是读取的yum仓库数据文件**

**vim编辑技巧**

**命令模式操作**

光标跳转



复制/粘贴/删除



查找/撤销/保存



**末行模式操作**

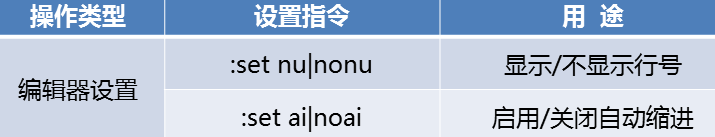
保存/退出/文件操作



字符串替换



开关参数的控制



**vim补充:**

1 vim不能创建路径

2 vim缓存文件(交换文件),不允许多个终端打开同一个文件

#########################################

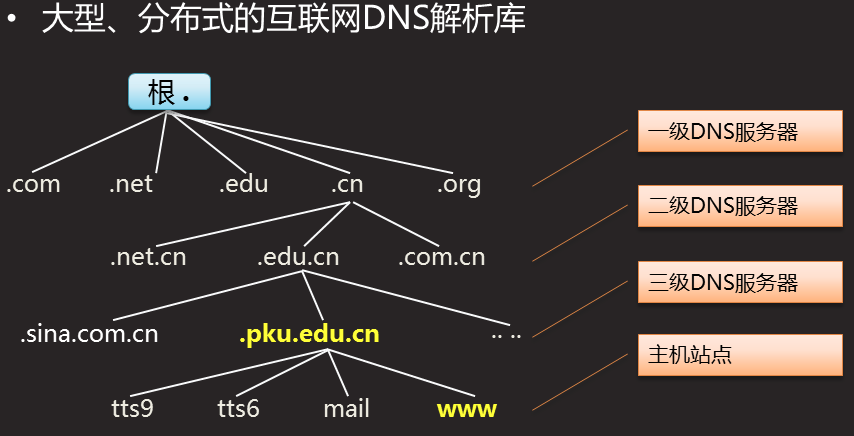
**I-Day16**

**DNS服务基础**

DNS服务器的功能

正向解析: 根据注册的域名查找其对应的IP地址

反向解析: 根据IP地址查找对应的注册域名,不常用



所有域名都必须以 点 作为结尾 www.baidu.com.

根域名: ‘.’

一级域名(顶级域名): .cn .us .hk .tw .jp .kr ........

二级域名: .com.cn .edu.cn .org.cn .net.cn .......

三级域名: .nb.com.cn .haha.com.cn .xixi.com.cn ........

完整合格的主机名: [www.nb.com.cn](http://www.bn.com.cn) ftp.nb.com.cn

完整合格的主机名: 主机头部+域名

Full Qualified Domain Name,完全合格主机名(FQDN)

**BIND服务分析**

DNS默认端口: TCP/UDP 53

系统服务: named

主配置文件: /etc/named.conf #设置本机负责解析的域名

地址库文件: /var/named/... #完整的主机名与IP地址对应关系

软件包: bind bind-chroot

虚拟机A:

1.安装软件包

[root@svr7 ~]# yum -y install bind bind-chroot

bind.x86\_64.32.9.9.4-50.el7 #域名服务包

bind-chroot.x86\_64.32.9.9.4-50.el7 #提供虚拟根支持(牢笼政策)

2 修改主配置文件

[root@svr7 ~]# cp /etc/named.conf /etc/named.bak #备份文件,防止配置的时候出错

[root@svr7 ~]# vim /etc/named.conf #修改成下面的文本内容

options {

directory "/var/named"; #指定地址库文件存放路径

};

zone "tedu.cn" IN { #指定负责解析的域名为tedu.cn

type master; #指定本机为权威服务器

file "tedu.cn.zone"; #指定地址库文件名称

};

3 建立地址库文件/var/named/tedu.cn.zone #保证named用户有读权限

[root@svr7 ~]# cd /var/named

[root@svr7 named]# cp -p named.localhost tedu.cn.zone **#保持权限复制**

[root@svr7 named]# id named

uid=25(named) gid=25(named) 组=25(named)

[root@svr7 named]# vim tedu.cn.zone #改成下面的文本内容

1) 地址库文件所有域名都必须以点结尾;

2) 不以点结尾, 则自动补全本地址库负责的域名

$TTL 1D

@ IN SOA @ rname.invalid. (

0 ; serial

1D ; refresh

两者要一样,可以换成其他名

1H ; retry

1W ; expire

3H ) ; minimum

tedu.cn. NS svr7

svr7 A 192.168.4.7

www A 192.168.4.1

ftp A 192.168.4.2

4 重启named服务

[root@svr7 named]# systemctl restart named

虚拟机B: 客户端

1 指定本机DNS服务器地址/etc/resolve.conf

[root@pc207 ~]# echo nameserver 192.168.4.7 > /etc/resolv.conf

2 测试解析

[root@pc207 ~]# nslookup www.tedu.cn

Server: 192.168.4.7

Address: 192.168.4.7#53

Name: www.tedu.cn

Address: 192.168.4.1

[root@pc207 ~]# nslookup ftp.tedu.cn

**构建多区域的DNS服务器**

1 修改主配置文件

[root@svr7 ~]# vim /etc/named.conf

options {

directory "/var/named";

};

zone "tedu.cn" IN {

type master;

file "tedu.cn.zone";

};

zone "qq.com" IN {

type master;

file "qq.com.zone";

};

2 建立地址库文件/var/named/qq.com.zone

[root@svr7 ~]# cd /var/named

[root@svr7 named]# cp -p tedu.cn.zone qq.com.zone

[root@svr7 named]# vim qq.com.zone

$TTL 1D

@ IN SOA @ rname.invalid. (

0 ; serial

1D ; refresh

1H ; retry

1W ; expire

3H ) ; minimum

qq.com. NS svr7

svr7 A 192.168.4.7

www A 1.2.3.4

3 重启named服务

[root@svr7 named]# systemctl restart named

**特殊的解析记录**

1 基于DNS的轮询 -------几乎没人用

一个域名------->多个不同的IP地址

[root@svr7 named]# vim /var/named/tedu.cn.zone

......

tedu.cn. NS svr7

svr7 A 192.168.4.7

www A 192.168.4.1

www A 192.168.4.2

www A 192.168.4.3

www A 192.168.4.4

ftp A 2.2.2.2

[root@svr7 named]# !sys

systemctl restart named

2 泛域名解析

[root@svr7 ~]# vim /var/named/tedu.cn.zone

......

tedu.cn. NS svr7

svr7 A 192.168.4.7

tedu.cn. A 10.20.30.40

\* A 11.12.13.14

[root@svr7 ~]# systemctl restart named

虚拟机B测试:

[root@pc207 ~]# nslookup tedu.cn

[root@pc207 ~]# nslookup wwww.tedu.cn

[root@pc207 ~]# nslookup xixi.tedu.cn

3 有规律的泛域名解析

pc1.tedu.cn. ------> 192.168.10.1

pc2.tedu.cn. ------> 192.168.10.2

pc3.tedu.cn. ------> 192.168.10.3

pc4.tedu.cn. ------> 192.168.10.4

........

pc50.tedu.cn ------> 192.168.10.50

内置函数: $GENERATE 起始点-结束点

[root@svr7 ~]# vim /var/named/tedu.cn.zone

......

$GENERATE 1-50 pc$ A 192.168.10.$

[root@svr7 ~]# systemctl restart named

虚拟机B测试:

[root@pc207 ~]# nslookup pc5.tedu.cn

Server: 192.168.4.7

Address: 192.168.4.7#53

Name: pc5.tedu.cn

Address: 192.168.10.5

[root@pc207 ~]# nslookup pc51.tedu.cn

Server: 192.168.4.7

Address: 192.168.4.7#53

Name: pc51.tedu.cn

Address: 11.12.13.14

4 解析记录的别名

root@svr7 /]# vim /var/named/tedu.cn.zone

......

tts CNAME svr #tts解析结果与svr一致

[root@svr7 /]# !sys

systemctl restart named

虚拟机B测试

[root@pc207 named]# nslookup tts.tedu.cn

Server: 192.168.4.7

Address: 192.168.4.7#53

tts.tedu.cn canonical name = svr7.tedu.cn.

Name: svr7.tedu.cn

Address: 192.168.4.7

**总结: 常见的DNS资源解析记录有哪些?**

**NS: 声明DNS记录**

**A: 正向解析记录**

**CNAME: 解析记录别名**

**DNS子域授权**

父域: www.qq.com

子域: [www.bj.qq.com](http://www.bj.qq.com)

父域qq.com由虚拟机A解析

子域bj.qq.com由虚拟机B解析

虚拟机B: 构建DNS服务器

1 安装软件包

[root@pc207 ~]# yum -y install bind bind-chroot

2 修改主配置文件

[root@pc207 ~]# cp /etc/named.conf /etc/named.bak #备份,防止出错

[root@pc207 ~]# vim /etc/named.conf

options {

directory "/var/named";

};

zone "bj.qq.com" IN {

type master;

file "bj.qq.com.zone";

};

3 建立地址库文件

[root@pc207 ~]# cd /var/named

[root@pc207 named]# cp -p named.localhost bj.qq.com.zone

[root@pc207 named]# vim bj.qq.com.zone

$TTL 1D

@ IN SOA @ rname.invalid. (

0 ; serial

1D ; refresh

1H ; retry

1W ; expire

3H ) ; minimum

bj.qq.com. NS pc207

pc207 A 192.168.4.207

www A 192.168.100.1

4 重启named服务

[root@pc207 named]# systemctl restart named

5 测试解析

[root@pc207 named]# nslookup www.bj.qq.com 192.168.4.207

Server: 192.168.4.207

Address: 192.168.4.207#53

Name: www.bj.qq.com

Address: 192.168.100.1

**子域授权: 让父域的DNS服务器,可以解析子域DNS负责的域名**

虚拟机A: 修改地址库文件

[root@svr7 ~]# vim /var/named/qq.com.zone

......

qq.com. NS svr7

bj.qq.com. NS pc207

svr7 A 192.168.4.7

pc207 A 192.168.4.207

www A 1.2.3.4

[root@svr7 ~]# systemctl restart named

[root@svr7 ~]# nslookup www.bj.qq.com 192.168.4.7 #测试

Server: 192.168.4.7

Address: 192.168.4.7#53

Non-authoritative answer: #非权威解答

Name: www.bj.qq.com

Address: 192.168.100.1

**递归解析**: 将首选DNS服务器,与其他DNS服务器交互,最终将解析结果带回来的过程

**迭代解析:** 将首选DNS服务器,与其他DNS服务器交互

recursion yes|no 递归解析|禁止递归解析 (默认yes,在主配置文件中)

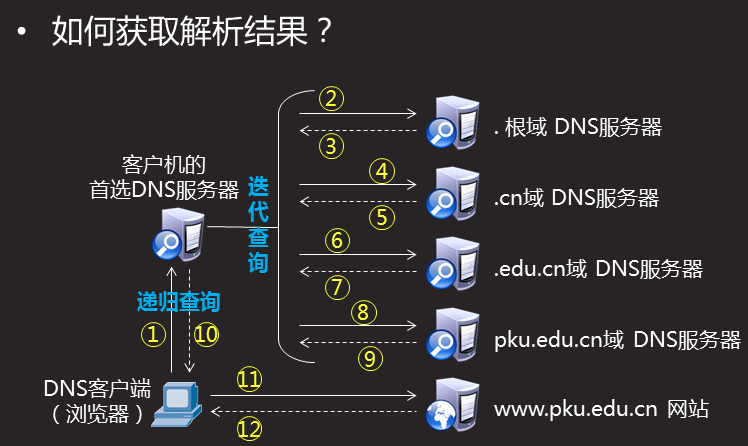
......

options {

directory "/var/named";

recursion yes; #支持递归解析:

};



**主机名映射文件:** /etc/hosts nslookup不读此文件

作用: 只为本机直接提供解析结果

[root@svr7 ~]# vim /etc/hosts

......

192.168.4.222 www.sina.com

[root@svr7 ~]# ping -c 1 www.sina.com

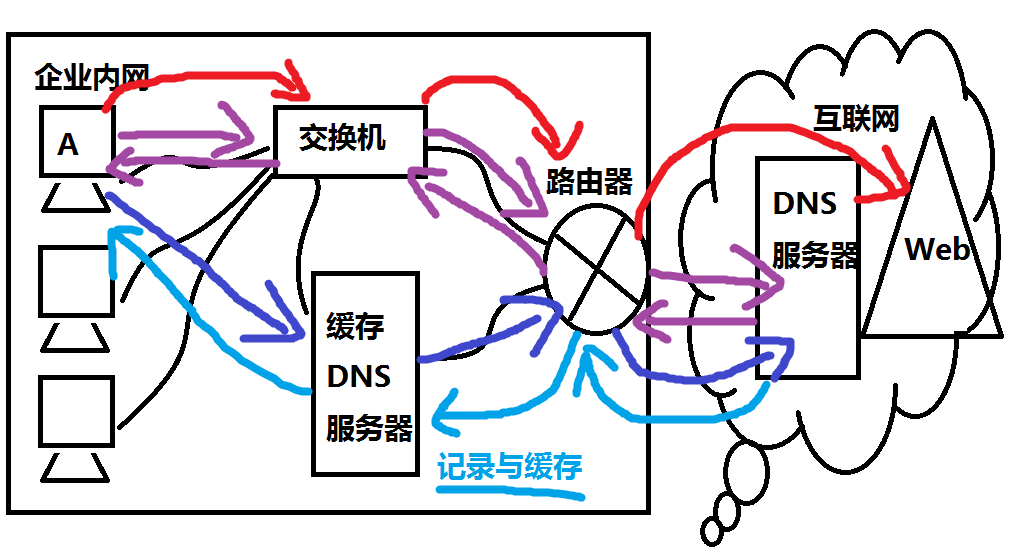
PING www.sina.com (192.168.4.222) 56(84) bytes of data.

客户端解析域名优先级(匹配及停止)

1 /etc/hosts > 2 /etc/resolv.conf

**缓存DNS服务 -----企业中应用的最多**

作用:缓存解析记录,加快解析



在真机上操作:

1 搭建yum仓库

[root@room9pc01 ~]# cat /etc/redhat-release

CentOS Linux release 7.4.1708 (Core)

[root@room9pc01 ~]# mkdir /dvd/

[root@room9pc01 ~]# mount /ISO/CentOS-7-x86\_64-DVD-1708.iso /dvd/

mount: /dev/loop2 写保护，将以只读方式挂载

[root@room9pc01 ~]# cd /etc/yum.repos.d/

[root@room9pc01 yum.repos.d]# mkdir repo

[root@room9pc01 yum.repos.d]# mv \*.repo repo

[root@room9pc01 yum.repos.d]# vim dvd.repo

[dvd]

name=Centos7.4

baseurl=file:///dvd

enabled=1

gpgcheck=0

[root@room9pc01 yum.repos.d]# yum -y install bind bind-chroot

2 查看真机DNS服务器

[root@room9pc01 yum.repos.d]# cat /etc/resolv.conf

# Generated by NetworkManager

search tedu.cn

nameserver 176.19.0.26

3 修改配置文件

[root@room9pc01 yum.repos.d]# cp /etc/named.conf /etc/named.bak #备份,防止出错

[root@room9pc01 yum.repos.d]# vim /etc/named.conf

options {

directory "/var/named";

forwarders { 176.19.0.26; }; #转发给真机DNS服务器

};

[root@room9pc01 yum.repos.d]# systemctl restart named

4 虚拟机验证

[root@svr7 ~]# nslookup www.baidu.com 192.168.4.254

Server: 192.168.4.254

Address: 192.168.4.254#53

Non-authoritative answer:

www.baidu.com canonical name = www.a.shifen.com.

Name: www.a.shifen.com

Address: 115.239.211.112

Name: www.a.shifen.com

Address: 115.239.210.27

#######################################

**I-Day17**

**Split分离解析**

Split分离解析(视图解析):

什么是分离解析

• 当收到客户机的DNS查询请求的时候

– 能够区分客户机的来源地址

– 为不同类别的客户机提供不同的解析结果(IP地址)

– 不同类别的客户机,解析同样的域名,结果不同

– 作用: 针对不同的客户机就近提供服务器 (为客户端提供最近的资源)

BIND的view视图

1) 根据源地址集合将客户机分类

2) 不同客户机获得不同结果(待遇有差别)

3) 由**上及下进行匹配**,匹配及停止

4) 分类要合理,所有的客户端都要找到自己的分类

5) 所有的zone都要在view中

view "联通" {

match-clients { 来源地址1; .. ..; };

zone "12306.cn" IN {

...... 地址库1;

};

};

view "铁通" {

match-clients { 来源地址2; .. .. ; };

zone "12306.cn" IN {

...... 地址库2;

};

};

定义view视图后,不允许在view以外出现zone配置

**案例:**

• 环境及需求

– 权威DNS:svr7.tedu.cn 192.168.4.7

– 负责区域:tedu.cn

– A记录分离解析 —— 以 www.tedu.cn 为例

客户机来自 解析结果

192.168.4.207 、192.168.7.0/24---------> 192.168.4.100

其他地址 ---------> 1.2.3.4

[root@svr7 ~]# vim /etc/named.conf

options {

directory "/var/named";

};

view "nsd" {

match-clients { 192.168.4.207; 192.168.7.0/24; };

zone "tedu.cn" IN {

type master;

file "tedu.cn.zone";

};

};

view "other" {

match-clients { any; };

zone "tedu.cn" IN {

type master;

file "tedu.cn.other";

};

};

[root@svr7 ~]# vim /var/named/tedu.cn.zone

$TTL 1D

@ IN SOA @ rname.invalid. (

0 ; serial

1D ; refresh

1H ; retry

1W ; expire

3H ) ; minimum

tedu.cn. NS svr7

svr7 A 192.168.4.7

www A 192.168.4.100

[root@svr7 ~]# vim /var/named/tedu.cn.other

$TTL 1D

@ IN SOA @ rname.invalid. (

0 ; serial

1D ; refresh

1H ; retry

1W ; expire

3H ) ; minimum

tedu.cn. NS svr7

svr7 A 192.168.4.7

www A 1.2.3.4

[root@svr7 ~]# systemctl restart named

验证:

[root@svr7 ~]# nslookup www.tedu.cn

Server: 127.0.0.1

Address: 127.0.0.1#53

Name: www.tedu.cn

Address: 1.2.3.4

[root@pc207 ~]# nslookup www.tedu.cn

Server: 192.168.4.7

Address: 192.168.4.7#53

Name: www.tedu.cn

Address: 192.168.4.100

**多区域的分离解析**

原则: 每一个view中,zone的个数与负责的域名保持一致,地址库文件不同

[root@svr7 ~]# vim /etc/named.conf

options {

directory "/var/named";

};

view "nsd" {

match-clients { 192.168.4.207; 192.168.7.0/24; };

zone "tedu.cn" IN {

type master;

file "tedu.cn.zone";

};

zone "qq.com" IN {

type master;

file "qq.com.zone"; --------->解析结果6.6.6.6

};

};

view "other" {

match-clients { any; };

zone "tedu.cn" IN {

type master;

file "tedu.cn.other";

};

zone "qq.com" IN {

type master;

file "qq.com.other"; --------->解析结果7.7.7.7

};

};

[root@svr7 ~]# cd /var/named

[root@svr7 named]# vim qq.com.zone

$TTL 1D

@ IN SOA @ rname.invalid. (

0 ; serial

1D ; refresh

1H ; retry

1W ; expire

3H ) ; minimum

qq.com. NS svr7

bj.qq.com. NS pc207

svr7 A 192.168.4.7

pc207 A 192.168.4.207

www A 6.6.6.6

[root@svr7 named]# cp -p qq.com.zone qq.com.other

[root@svr7 named]# vim qq.com.other

$TTL 1D

@ IN SOA @ rname.invalid. (

0 ; serial

1D ; refresh

1H ; retry

1W ; expire

3H ) ; minimum

qq.com. NS svr7

bj.qq.com. NS pc207

svr7 A 192.168.4.7

pc207 A 192.168.4.207

www A 7.7.7.7

[root@svr7 named]# systemctl restart named

虚拟机A验证

[root@svr7 named]# nslookup www.qq.com 192.168.4.7

Server: 192.168.4.7

Address: 192.168.4.7#53

Name: www.qq.com

Address: 7.7.7.7

虚拟机B验证

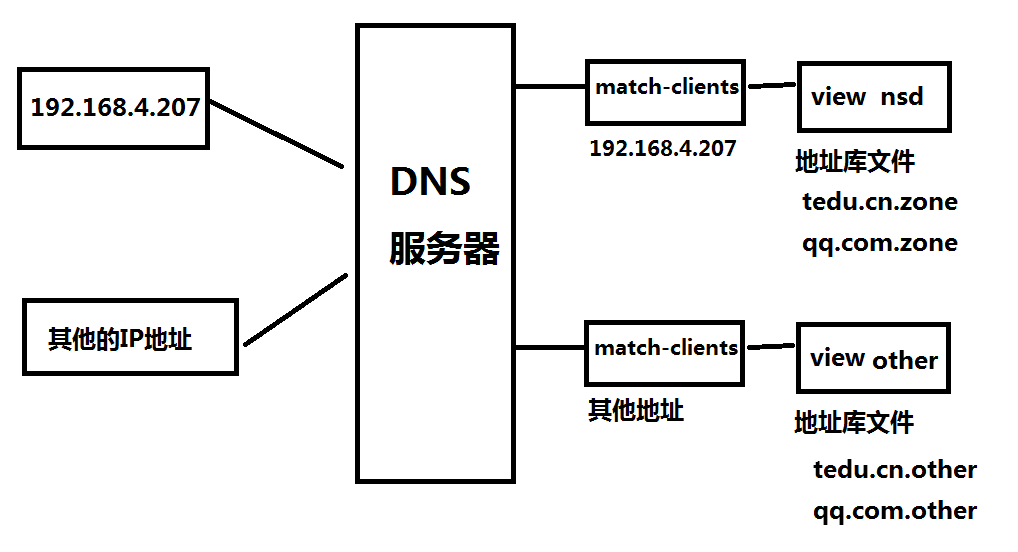
[root@pc207 ~]# nslookup www.qq.com

Server: 192.168.4.7

Address: 192.168.4.7#53

Name: www.qq.com

Address: 6.6.6.6



**了解内容: acl地址列表**

• 为大批量的客户机地址建立列表

– 调用时指定列表名即可,列表名 any 可匹配任意地址

– 根据view调用的顺序,“匹配即停止”

[root@svr7 ~]# vim /etc/named.conf

options {

directory "/var/named";

};

acl abc { 192.168.4.207; 192.168.7.0/24; };

view "nsd" {

match-clients { abc; };

zone "tedu.cn" IN {

type master;

file "tedu.cn.zone";

};

zone "qq.com" IN {

type master;

file "qq.com.zone";

};

};

view "other" {

match-clients { any; };

zone "tedu.cn" IN {

type master;

file "tedu.cn.other";

};

zone "qq.com" IN {

type master;

file "qq.com.other";

};

};

**RAID磁盘阵列**

RAID阵列概述

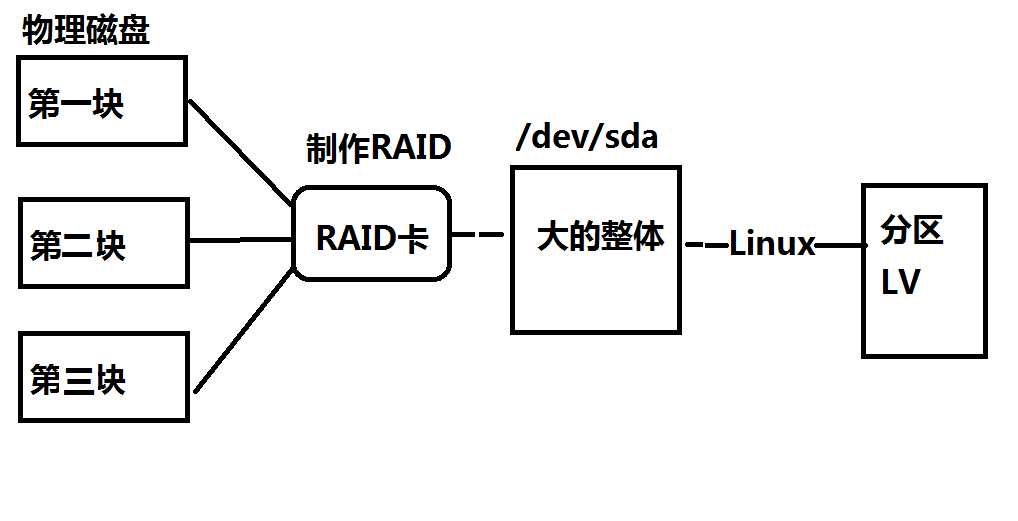
廉价冗余磁盘阵列

1) Redundant Arrays of Inexpensive Disks

2) 通过硬件/软件技术,将多个较小/低速的磁盘整合成一个大磁盘

3) 阵列的价值:提升I/O效率、硬件级别的数据冗余

4) 不同RAID级别的功能、特性各不相同



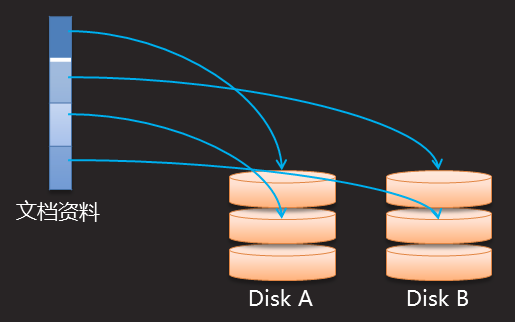
RAID 0, 条带模式

1) 同一个文档分散存放在不同磁盘

2) 并行写入以提高效率

3) 无可靠性提升

4) 至少由两块磁盘组成

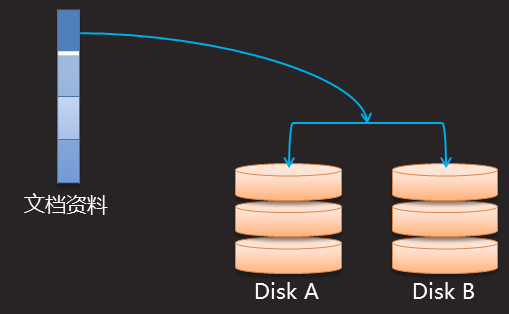


RAID 1,镜像模式

1) 一个文档复制成多份,分别写入不同磁盘

2) 多份拷贝提高可靠性,效率无提升

3) 至少需要两块磁盘组成

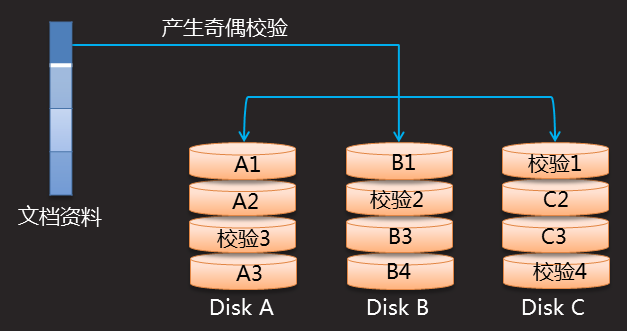


RAID5,高性价比模式

1) 相当于RAID0和RAID1的折中方案

2) 需要至少一块磁盘的容量来存放校验数据

3) 至少需要三块磁盘组成

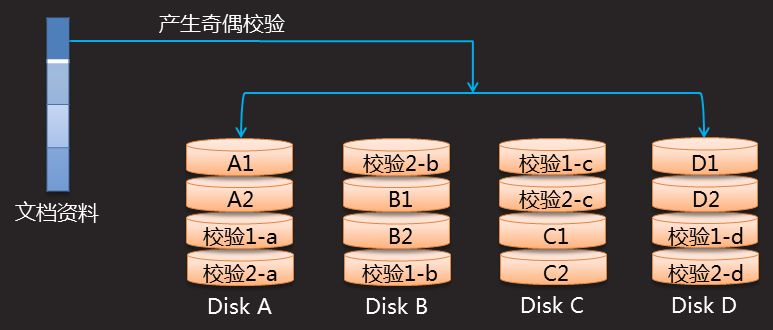


RAID6,高性价比/可靠模式

1) 相当于扩展的RAID5阵列,提供2份独立校验方案

2) 需要至少两块磁盘的容量来存放校验数据

3) 至少需要四块磁盘组成

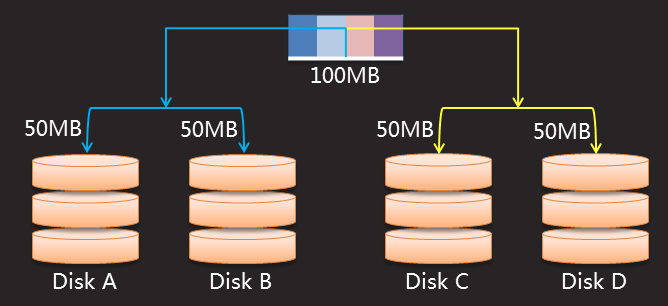


RAID 0+1/RAID 1+0(RAID10)

1) 整合RAID 0、RAID 1的优势

2) 并行存取提高效率、镜像写入提高可靠性

3) 至少需要四块磁盘组成



RAID各级别特点对比:



RAID阵列实现方式:

硬RAID: 由RAID控制卡管理阵列

主板---->阵列卡---->磁盘---->操作系统---->数据

软RAID: 有操作系统来管理阵列

主板---->磁盘---->操作系统---->RAID软件---->数据

**进程管理**程序: 静态的代码,占用磁盘空间

进程: 动态运行的代码,占用内存与CPU

父进程与子进程-------树型结构 父进程结束,所有子进程也得结束(先结束子进程,父进程在结束)

进程标识: PID

**查看进程树**

pstree (Processes Tree)

格式: pstree [选项] [PID或用户名]

常用命令选项

-a: 显示完整的命令行

-p: 列出对应PID编号

systemd : 上帝进程(所有进程的父进程), Linux初始化进程,PID永远为1

[root@svr7 ~]# su - lisi

[lisi@svr7 ~]$ vim a.txt #打开即可,停在命令模式

[root@svr7 ~]# pstree lisi

bash───vim

[root@svr7 ~]# pstree -a lisi

bash

└─vim a.txt

[root@svr7 ~]# pstree -p lisi

bash(5341)───vim(5385)

[root@svr7 ~]# pstree -pa lisi

bash,5341

└─vim,5385 a.txt

**查看进程快照**

ps -----Processes Snapshot

ps aux 操作

列出正在运行的所有进程,信息非常详细

用户 进程ID %CPU %内存 虚拟内存 固定内存 终端 状态 起始时间 CPU时间 程序指令

ps -elf 操作 (PPID----父进程的PID)

列出正在运行的所有进程,会有PPID(父进程)信息

补充命令: wc 统计

格式: wc [选项] [参数]

计算Linux一共有多少个用户?

[root@svr7 ~]# wc -l /etc/passwd

42 /etc/passwd

计算/etc/目录下有多少个.conf结尾的文件

[root@svr7 ~]# find /etc -name "\*.conf" | wc -l

435

计算/etc/目录下有多少个tab结尾的文件

[root@svr7 ~]# find /etc -name "\*tab" | wc -l

9

统计正在运行的进程有多少个

[root@svr7 ~]# ps aux | wc -l

115

[root@svr7 ~]# ps -elf | wc -l

115

[root@svr7 ~]# ps aux #显示正在运行的所有进程信息

[root@svr7 ~]# ps -elf #显示正在运行的所有进程信息,有PPID信息

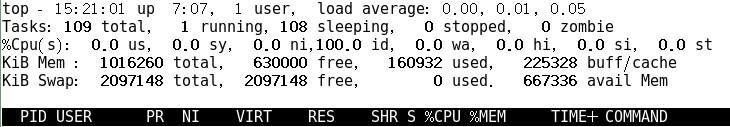
**进程动态排名**

top 交互式工具

格式: top [-d 刷新秒数] [-U 用户名] #默认5秒

按大写的P,利用CPU进行降序排列

按大写的M,利用内存使用比例,进行降序排列



**检索进程**

pgrep — Process Grep

格式: pgrep [选项]... 查询条件

常用命令选项

-l:输出进程名,而不仅仅是 PID

-U:检索指定用户的进程 # -l得在-U之前, 否则会报错

-t:检索指定终端的进程

-x:精确匹配完整的进程名

root@svr7 ~]# pgrep a

root@svr7 ~]# pgrep -l log

root@svr7 ~]# pgrep -l cron

root@svr7 ~]# pgrep -lU lisi #查询lisi用户开启的进程

root@svr7 ~]# pgrep -ap lisi

root@svr7 ~]# pstree -ap lisi

root@svr7 ~]# pgrep -lU lisi

root@svr7 ~]# who #查询当前登录的用户信息

root@svr7 ~]# pgrep -lU lisi -t pts/2 #查询pts/2终端

root@svr7 ~]# top -d 1

**控制进程**

**进程的前后台调度**

前台启动: 输入正常的命令行,运行期间占用当前终端

后台启动: 命令行末尾添加“&”符号,不占用当前终端,正在运行的放入后台

Ctrl + z 组合键: 挂起当前进程(暂停并转入后台)

jobs 命令: 查看后台任务列表

fg 命令: 将后台任务恢复到前台运行

bg 命令: 激活后台被挂起的任务

[root@svr7 ~]# sleep 800 & #将进程正在运行放入后台

[1] 6247

[root@svr7 ~]# jobs #查看后台进程信息

[1]+ 运行中 sleep 800 &

[root@svr7 ~]# sleep 700

^Z # 按ctrl+z暂停放入后台

[2]+ 已停止 sleep 700

[root@svr7 ~]# jobs

[1]- 运行中 sleep 800 &

[2]+ 已停止 sleep 700

[root@svr7 ~]# bg 2 #将后台编号为2的进程,继续运行

[2]+ sleep 700 &

[root@svr7 ~]# jobs

[1]- 运行中 sleep 800 &

[2]+ 运行中 sleep 700 &

[root@svr7 ~]# fg 2 #将后台编号为2的进程,恢复到前台进行运行

sleep 700

^C #按ctrl+c结束进程

[root@svr7 ~]# jobs

[1]+ 运行中 sleep 800 &

[root@svr7 ~]# fg 1 #将后台编号为1的进程,恢复到前台进行运行sleep 800

^C #按ctrl+c结束进程

[root@svr7 ~]# jobs

**杀死进程**

干掉进程的不同方法

1) Ctrl+c 组合键,中断当前命令程序

2) kill [-9] PID... kill [-9] %后台任务编号 -9:强制杀死进程

3) killall [-9] 进程名... -9:强制杀死进程

4) pkill 查找条件 不常用,会误杀

[root@svr7 ~]# sleep 800 &

[1] 6565

[root@svr7 ~]# sleep 700 &

[2] 6566

[root@svr7 ~]# sleep 600 &

[3] 6567

[root@svr7 ~]# jobs -l

[1] 6565 运行中 sleep 800 &

[2]- 6566 运行中 sleep 700 &

[3]+ 6567 运行中 sleep 600 &

[root@svr7 ~]#

[root@svr7 ~]# kill 6565

[root@svr7 ~]# jobs -l

[1] 6565 已终止 sleep 800

[2]- 6566 运行中 sleep 700 &

[3]+ 6567 运行中 sleep 600 &

[root@svr7 ~]# kill -9 6566

[root@svr7 ~]# jobs -l #查看后台进程信息,显示PID

[2]- 6566 已杀死 sleep 700

[3]+ 6567 运行中 sleep 600 &

[root@svr7 ~]# killall sleep

[3]+ 已终止 sleep 600

**补充: 强制踢出一个用户(杀死该用户所有进程)**

[root@svr7 ~]# killall -9 -u lisi #会把终端也杀死

**日志管理在一个网络中,只能有一台DHCP服务器**

日志的功能

系统和程序的“日记本”

1) 记录系统、程序运行中发生的各种事件

2) 通过查看日志,了解及排除故障

3) 信息安全控制的“依据”

内核及系统日志

由系统服务rsyslog统一记录/管理

1) 日志消息采用文本格式

2) 主要记录事件发生的时间、主机、进程、内容

常见的日志文件



用户日志

由登录程序负责记录/管理

1. 日志消息采用二进制格式
2. 记录登录用户的时间、来源、执行的命令等信息



**日志分析**

通用分析工具

1. tail、tailf、less、grep等文本浏览/检索命令

tailf: 实时跟踪日志

1. awk、sed等格式化过滤工具

专用分析工具------以后学

1. Webmin 系统管理套件
2. Webalizer 、AWStats等日志统计套件

**用户登录分析**

users、who、w 命令: 查看已登录的用户信息,详细度不同

last、lastb 命令: 查看最近登录成功/失败的用户信息

[root@svr7 /]# last -2 #查看最近登录成功

[root@svr7 /]# lastb -2 #查看最近登录失败

**日志消息的优先级**

Linux内核定义的事件紧急程度

1) 分为 0~7 共8种优先级别

|  |  |  |
| --- | --- | --- |
| 0 | EMERG（紧急） | 会导致主机系统不可用的情况 |
| 1 | ALERT（警告） | 必须马上采取措施解决的问题 |
| 2 | CRIT（严重 | 比较严重的情况 |
| 3 | ERR（错误） | 运行出现错误 |
| 4 | WARNING（提醒） | 可能会影响系统功能的事件 |
| 5 | NOTICE（注意） | 不会影响系统但值得注意 |
| 6 | INFO（信息 | 一般信息 |
| 7 | DEBUG（调试） | 程序或系统调试信息等 |

2) 其数值越小,表示对应事件越紧急/重要

**使用journalctl工具 只有redhat7支持**

提取由 systemd-journal 服务搜集的日志

主要包括内核/系统日志、服务日志

常见用法

journalctl | grep 关键词

**journalctl -u 服务名 [-p 优先级]**

journalctl -n 消息条数

journalctl --since="yyyy-mm-dd HH:MM:SS" --

until="yyyy-mm-dd HH:MM:SS"

[root@svr7 /]# journalctl -u httpd #显示httpd服务日志

**systemctl控制**

Linux系统和服务管理器

1) 是内核引导之后加载的第一个初始化进程(PID=1)

2) 负责掌控整个Linux的运行/服务资源组合

systemd:

一个更高效的系统&服务管理器

1. 开机服务并行启动,各系统服务间的精确依赖
2. 配置目录: /etc/systemd/system/
3. 服务目录: /lib/systemd/system/
4. 主要管理工具: systemctl

对于服务的管理

systemctl restart 服务名 #重起服务

systemctl start 服务名 #开启服务

systemctl stop 服务名 #停止服务

systemctl status 服务名 #查看服务当前的状态

systemctl enable 服务名 #设置服务开机自启动

systemctl disable 服务名 #设置服务不开机自启动

**RHEL6 运行级别** 200

0：关机 0

1：单用户模式（基本功能的实现，破解Linux密码） 50

2：多用户字符界面（不支持网络） 60

3：多用户字符界面（支持网络）服务器默认的运行级别 80

4：未定义

5：图形界面 160

6：重起 0

切换运行级别：init [0-6]

**RHEL7 运行模式**

字符模式：multi-user.target

图形模式：graphical.target

[root@svr7 /]# ls -l /lib/systemd/system/

当前直接切换到图形模式

[root@svr7 /]# systemctl isolate graphical.target

当前直接切换到字符模式

[root@svr7 /]# systemctl isolate multi-user.target

查看每次开机默认进入模式

[root@svr7 ~]# systemctl get-default

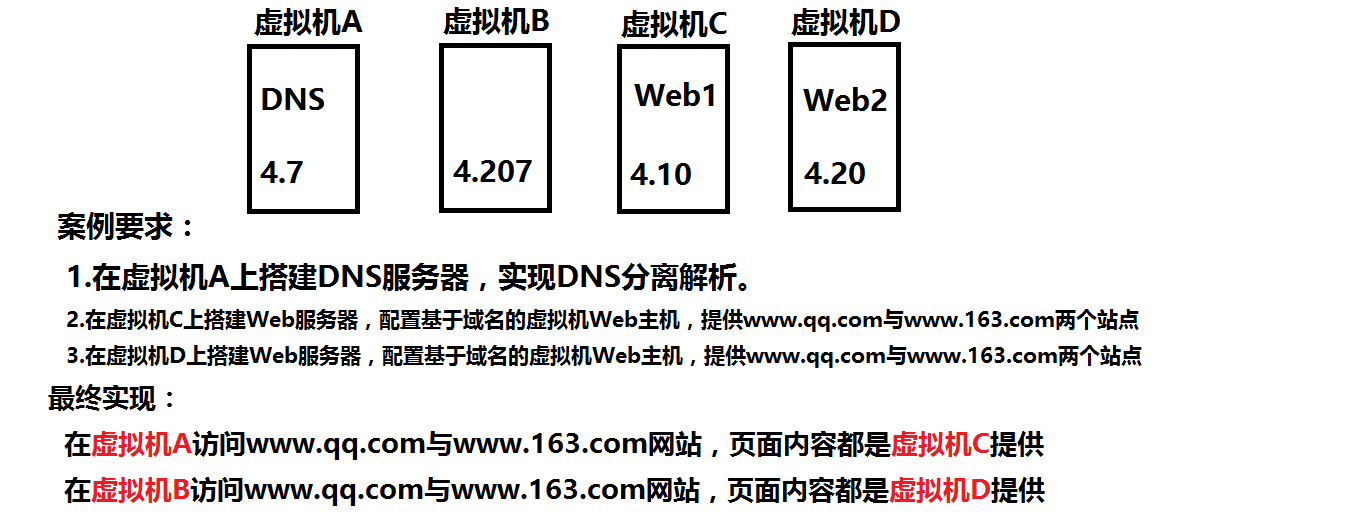
multi-user.target

设置永久策略，每次开机自动进入graphical.target

[root@svr7 ~]# systemctl set-default graphical.target

[root@svr7 ~]# reboot

########################################



#########################################

**I-Day18**

**批量装机**

**部署DHCP服务**

DHCP服务作用:

Dynamic Host Configuration Protocol

动态主机配置协议,由 IETF(Internet 网络工程师任务小组)组织制定,用来简化主机地址分配管理

主要分配以下入网参数

1) IP地址/子网掩码/广播地址

2) 默认网关地址、DNS服务器地址

3) PXE引导设置(TFTP服务器地址、引导文件名)

DHCP地址分配的四次会话(广播的形式进行,**先到先得**)

DISCOVERY ----> OFFER ----> REQUEST ---->ACK

**在一个网络中,只能有一台DHCP服务器,不然客户端IP地址会经常变**

服务端基本概念

1) 租期:允许客户机租用IP地址的时间期限,单位为秒

2) 作用域:分配给客户机的IP地址所在的网段

3) 地址池:用来动态分配的IP地址的范围

装软件包dhcp

配置文件/etc/dhcp/dhcpd.conf

起服务dhcpd

虚拟机A:

1 装软件包dhcp

[root@svr7 ~]# yum -y install dhcp

2 修改配置文件/etc/dhcp/dhcpd.conf

[root@svr7 ~]# vim /etc/dhcp/dhcpd.conf

开启行号功能,vim末行模式下: set nu

读入其他文件内容:r /usr/share/doc/dhcp\*/dhcpd.conf.example

1 #

2 # DHCP Server Configuration file.

3 # see /usr/share/doc/dhcp\*/dhcpd.conf.example

4 # see dhcpd.conf(5) man page

5 subnet 192.168.4.0 netmask 255.255.255.0 { #指定分配的网段

6 range 192.168.4.20 192.168.4.100; #指定分配的IP地址范围

7 option domain-name-servers 8.8.8.8; #指定分配DNS服务器地址

8 option routers 192.168.4.254; #指定网关地址

9 default-lease-time 600;

10 max-lease-time 7200;

11 }

3 重启服务dhcpd

[root@svr7 ~]# systemctl restart dhcpd

**网络装机**

优势:

1. 规模化: 同时装配多台主机
2. 自动化: 装系统、配置各种服务
3. 远程实现: 不需要光盘、U盘等物理安装介质

什么是PXE网络

1. 预启动执行环境,在操作系统之前运行
2. 可用于远程安装

工作模式

1. PXE client集成在网卡的启动芯片中
2. 当计算机引导时,从网卡芯片中把PXE client调入内存执行,获取PXE server配置,显示菜单,根据用户选择将远程引导程序下载到本机运行

PXE组件及过程分析

• 需要哪些服务组件?

– DHCP服务,分配IP地址、定位引导程序

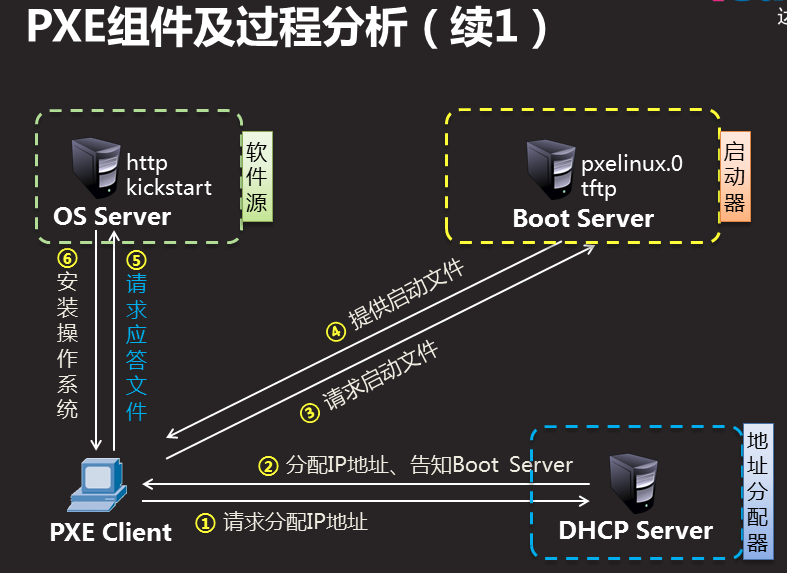
– TFTP服务,提供引导程序下载

– HTTP服务(或FTP/NFS),提供yum安装源

• 客户机应具备的条件

– 网卡芯片必须支持PXE协议

– 主板支持从网卡启动



虚拟机A:部署网络装机服务器

一 部署DHCP

1 修改配置文件

修改配置文件/etc/dhcp/dhcpd.conf

#

# DHCP Server Configuration file.

# see /usr/share/doc/dhcp\*/dhcpd.conf.example

# see dhcpd.conf(5) man page

subnet 192.168.4.0 netmask 255.255.255.0 {

range 192.168.4.20 192.168.4.100;

option domain-name-servers 8.8.8.8;

option routers 192.168.4.254;

default-lease-time 600;

max-lease-time 7200;

**next-server 192.168.4.7;** #指定下一个服务器IP地址

**filename "pxelinux.0";** #指定网卡引导文件名称

}

2 重启dhcpd服务

[root@svr7 ~]# systemctl restart dhcpd

pxelinux.0:网络装机说明书; 二进制文件; 安装一个软件产生,默认名字pxelinux.0

二 搭建tftp服务

tftp: 简单的文件传输协议,默认端口为69

默认共享路径: /var/lib/tftpboot

1 安装软件包tftp-server

[root@svr7 ~]# yum -y install tftp-server

2 重启tftp服务

[root@svr7 ~]# systemctl restart tftp

[root@svr7 ~]# systemctl enable tftp

3 部署pxelinux.0文件

[root@svr7 ~]# yum provides \*/pxelinux.0 #查询仓库中哪个包产生此文件

[root@svr7 ~]# yum -y install syslinux

[root@svr7 ~]# rpm -ql syslinux | grep pxelinux.0 #查询安装清单

[root@svr7 ~]# cp /usr/share/syslinux/pxelinux.0 /var/lib/tftpboot

pxelinux.0 ----->读取菜单文件 (/var/lib/tftpboot/pxelinux.cfg/default)

4 部署菜单文件

[root@svr7 ~]# mount /dev/cdrom /mnt

mount: /dev/sr0 写保护，将以只读方式挂载

[root@svr7 ~]# ls /mnt

addons GPL media.repo RPM-GPG-KEY-redhat-release

EFI images Packages TRANS.TBL

EULA isolinux repodata

extra\_files.json LiveOS RPM-GPG-KEY-redhat-beta

[root@svr7 ~]# mkdir /var/lib/tftpboot/pxelinux.cfg

[root@svr7 ~]# cp /mnt/isolinux/isolinux.cfg /var/lib/tftpboot/pxelinux.cfg/default

[root@svr7 ~]# ls -l /var/lib/tftpboot/pxelinux.cfg/default

-r--r--r--. 1 root root 3166 12月 21 12:02 /var/lib/tftpboot/pxelinux.cfg/default

[root@svr7 ~]# chmod u+w /var/lib/tftpboot/pxelinux.cfg/default

5 部署图形模块与背景图片

图形模块: vesamenu.c32 展现一定的颜色

背景图片: splash.png

[root@svr7 ~]# cp /mnt/isolinux/vesamenu.c32 /mnt/isolinux/splash.png /var/lib/tftpboot

[root@svr7 ~]# ls /var/lib/tftpboot

pxelinux.0 pxelinux.cfg splash.png vesamenu.c32

6 部署启动内核与启动所需的驱动程序

启动内核: vmlinuz

启动所需的驱动程序: initrd.img

[root@svr7 ~]# cp /mnt/isolinux/vmlinuz /mnt/isolinux/initrd.img /var/lib/tftpboot/

[root@svr7 ~]# ls /var/lib/tftpboot/

initrd.img pxelinux.cfg vesamenu.c32

pxelinux.0 splash.png vmlinuz

7 修改菜单文件

[root@svr7 ~]# vim /var/lib/tftpboot/pxelinux.cfg/default

1 default vesamenu.c32 #加载图形的模块

2 timeout 60 #读秒时间 1/10

......

10 menu background splash.png #背景图片

11 menu title NSD1811 PXE Server #修改标题

......

61 label linux

62 menu label Install RHEL7 #菜单显示内容

63 menu default #读秒结束后默认选择

64 kernel vmlinuz #加载内核

65 append initrd=initrd.img #加载驱动程序

三 搭建web服务,共享光盘所有的内容

1 安装httpd软件,直接启动服务

[root@svr7 ~]# yum -y install httpd

[root@svr7 ~]# systemctl restart httpd

[root@svr7 ~]# ls /var/www/html

[root@svr7 ~]# firefox 192.168.4.7 #页面为红帽测试页面

2 创建挂载目录,进行挂载光驱设备

[root@svr7 ~]# mkdir /var/www/html/rhel7

[root@svr7 ~]# mount /dev/cdrom /var/www/html/rhel7

mount: /dev/sr0 写保护，将以只读方式挂载

[root@svr7 ~]# ls /var/www/html/rhel7

[root@svr7 ~]# firefox 192.168.4.7/rhel7 #访问成功

四 部署无人值守安装, 生成应答文件

1 图形工具system-config-kickstart进行生成

[root@svr7 ~]# yum -y install system-config-kickstart

2 运行system-config-kickstart进行配置

[root@svr7 ~]# system-config-kickstart

首选查看 “软件包选择” 是否可以使用 #不可用(是正常的)

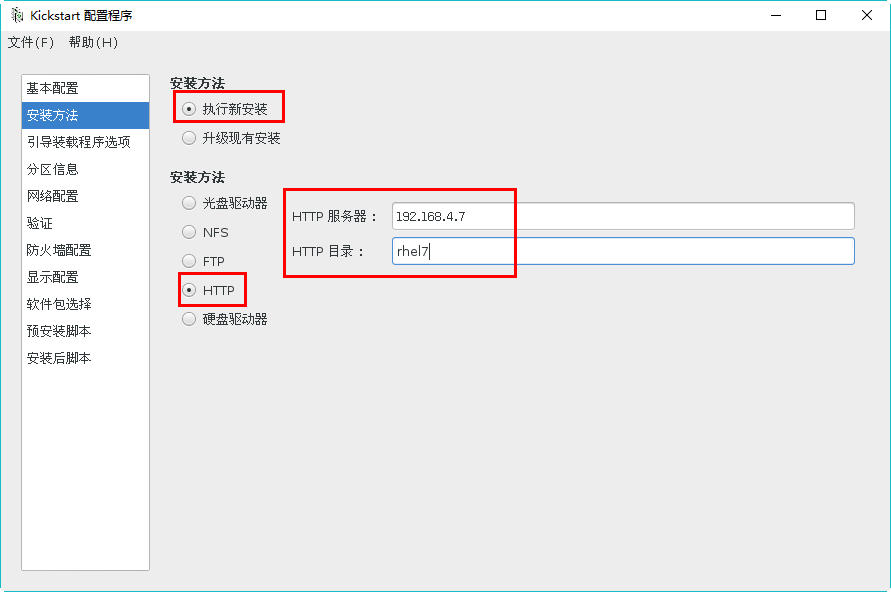
需要yum仓库的支持,必须要求光盘的仓库标识为[development]

[root@svr7 ~]# vim /etc/yum.repos.d/rhel7.repo

[development]

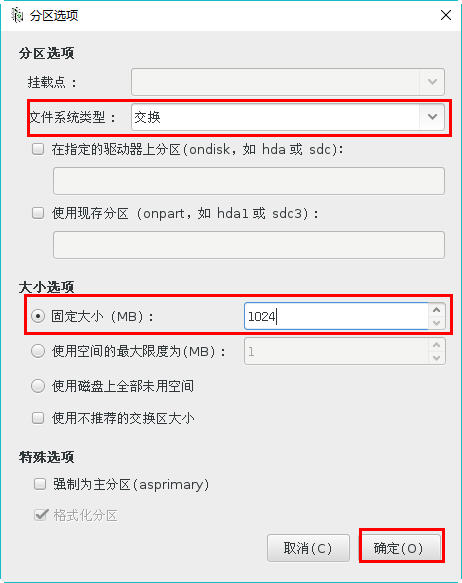
......

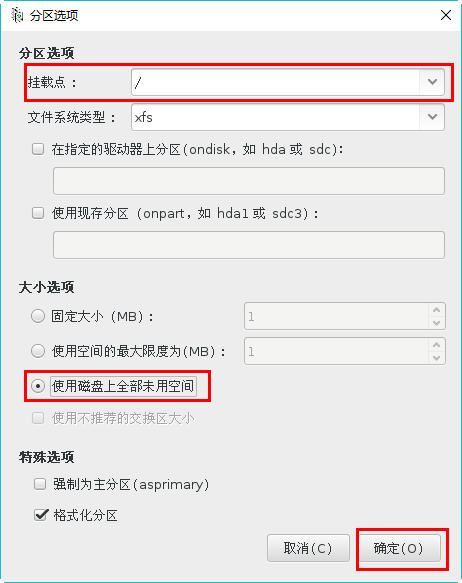




















3 查看root目录下ks文件生成

[root@svr7 ~]# ls /root/ks.cfg

/root/ks.cfg

4 利用web服务共享应答文件

[root@svr7 ~]# cp /root/ks.cfg /var/www/html/

[root@svr7 ~]# ls /var/www/html/

ks.cfg rhel7

[root@svr7 ~]# firefox 192.168.4.7/ks.cfg

5 修改菜单文件,指定ks应答文件位置

[root@svr7 ~]# vim /var/lib/tftpboot/pxelinux.cfg/default

......

label linux

menu label Install RHEL7

menu default

kernel vmlinuz

append initrd=initrd.img ks=http://192.168.4.7/ks.cfg #不能换行

**总结:**

1 DHCP---->IP地址 next-server filename

2 tftp-------> pxelinux.0

3 pxelinux.0------->菜单文件(/var/lib/tftpboot/pxelinux.cfg/default)

4 default------>vesamenu.c32 splash.png vmlnuz initrd.img

5 ks文件----->时区 密码 分区...

--url="http://192.168.4.7/rhel7"

验证:

1 新建一台虚拟机,安装方式选择PXE网络引导安装; 网络类型选择private1

课外扩展:

让客户端安装完系统后,yum仓库可用(利用ks应答文件安装后脚本实现)

方案1: 本地yum仓库

方案2: 网络yum仓库

服务端: 第一个真机FTP的yum源-------企业中比较常用

第二个选择网络装机服务器web的yum源

#########################################

**I-Day19**

**源码包编译安装**

RPM包: rpm -ivh yum -y install

源码包----->开发工具gcc与make------>可以执行的程序----->运行安装

主要优点:

1. 获得软件的最新版,及时修复bug
2. 软件功能可按需选择/定制,有更多的软件可供选择
3. 源码包适用各种平台

一 准备源码包

1 真机上操作

使用scp命令在虚拟机和物理机之间传输文件

格式: scp 源文件 root@客户机IP:路径

[root@room9pc01 ~]# scp /root/桌面/tools.tar.gz [root@192.168.4.7:/root/](mailto:root@192.168.4.7:/root/) #真机上操作

root@192.168.4.7's password:

tools.tar.gz 100% 766KB 111.3MB/s 00:00

2 虚拟机A上查看

[root@svr7 ~]# ls /root

3 解包

[root@svr7 ~]# tar -xf /root/tools.tar.gz -C /

[root@svr7 ~]# ls /tools

二 源码包编译安装

步骤1: 安装开发工具gcc与make

[root@svr7 ~]# yum -y install gcc make

步骤2: tar解包,释放源代码至指定目录

[root@svr7 ~]# tar -xf /tools/inotify-tools-3.13.tar.gz -C /opt/

[root@svr7 ~]# ls /opt/

步骤3: ./configure配置,指定安装目录/功能模块等选项,检测gcc是否安装

./ 类似于Windows的双击

--prefix: 指定安装路径

常见提示: 未安装gcc软件

[root@svr7 ~]# cd /opt/inotify-tools-3.13/ #必须在相对路径下执行

[root@svr7 inotify-tools-3.13]# ./configure

步骤4: make编译,生成可执行的二进制程序文件

[root@svr7 ~]# cd /opt/inotify-tools-3.13/

[root@svr7 ~]# make

步骤5: make install安装,将编译好的文件复制到安装目录

[root@svr7 ~]# cd /opt/inotify-tools-3.13/

[root@svr7 ~]# make install

[root@svr7 inotify-tools-3.13]# ls /usr/local/bin/inotifywait #检测程序是否生成

/usr/local/bin/inotifywait

**rsync同步操作**

命令用法

-r: 递归,包括目录/子目录及所有文件

-l: 保留符号链接文件

-p、-t:保留文件的权限、时间标记

-o、-g:保留文件的属主/属组标记

-D:保留设备文件及其他特俗文件

格式: rsync [选项...] 源目录 目标目录

同步与复制的差异

1) 复制:完全拷贝源到目标

2) 同步:增量拷贝,只传输变化过的数据

rsync操作选项

-n:测试同步过程,不做实际修改

--delete:删除目标文件夹内多余的文档

-a:归档模式,相当于-rlptgoD 必加的一个选项

-v:显示详细操作信息

-z:传输过程中启用压缩/解压

**本地同步**

1) rsync [选项...] 本地目录1 本地目录2 #同步整个文件夹

2) rsync [选项...] 本地目录1/ 本地目录2 #只同步目录下的数据

[root@svr7 ~]# mkdir /abc /dir

[root@svr7 ~]# cp /etc/fstab /etc/passwd /abc

[root@svr7 ~]# ls /abc

fstab passwd

[root@svr7 ~]# ls /dir

[root@svr7 ~]# rsync -avz /abc /dir #同步整个文件夹

[root@svr7 ~]# ls /dir

abc

[root@svr7 ~]# rm -rf /dir/\*

[root@svr7 ~]# rsync -zva /abc/ /dir #只同步目录下的数据

[root@svr7 ~]# ls /dir

fstab passwd

[root@svr7 ~]# cp /etc/shadow /abc/

[root@svr7 ~]# rsync -avz /abc/ /dir

[root@svr7 ~]# ls dir

fstab passwd shadow

[root@svr7 ~]# cp /etc/hosts /abc/

[root@svr7 ~]# touch /dir/{1..3}.txt

[root@svr7 ~]# ls /abc

fstab hosts passwd shadow

[root@svr7 ~]# ls /dir/

1.txt 2.txt 3.txt fstab passwd shadow

[root@svr7 ~]# rsync -avz --delete /abc/ /dir/

[root@svr7 ~]# ls /abc/

fstab hosts passwd shadow

[root@svr7 ~]# ls /dir/

fstab hosts passwd shadow

**rsync+SSH同步**

与远程的 SSH目录保持同步

1) 下行:rsync [...] user@host:远程目录 本地目录

2) 上行:rsync [...] 本地目录 user@host:远程目录

虚拟机svr7:

[root@svr7 ~]# rsync -avz --delete /opt/ root@192.168.4.207:/opt

虚拟机pc207:

[root@pc207 ~]# ls /opt

**两台服务器之间实时同步**

一 部署公钥私钥ssh无密码验证

1 虚拟机svr7生成公钥与私钥

[root@svr7 ~]# ssh-keygen #生成公钥与私钥,一路回车即可

[root@svr7 ~]# ls /root/.ssh

id\_rsa id\_rsa.pub known\_hosts

(known\_hosts: 记录曾经远程管理过的服务器信息)

2 将公钥进行拷贝到虚拟机pc207

[root@svr7 ~]# ssh-copy-id root@192.168.4.207

3 虚拟机pc207查看

[root@pc207 ~]# ls /root/.ssh/

authorized\_keys #公钥名字

4 虚拟机svr7验证无密码的同步

[root@svr7 ~]# rsync -avz --delete /opt/ root@192.168.4.207:/opt

二 运行一个程序,监控目录内容的变化

Linux内核的inotify机制

1. 提供事件响应式的文件系统通知机制
2. 安装inotify-tools控制工具可调用此机制实现监控

基本用法

格式: inotifywait [选项] 目标文件夹

常用命令选项

-m:持续监控(捕获一个事件后不退出)

-r:递归监控、包括子目录及文件

-q:减少屏幕输出信息

-qq: 屏幕不输出

-e:指定监视的 modify、move、create、delete、attrib(权限) 等事件类别

三 书写一个shell脚本完成实时同步

for循环适合写有次数的循环 {1..100}

while循环适合写死循环,当条件满足开始循环,条件失败停止循环

while [ 条件 ] while 系统命令 #执行命令,判断是否执行成功

do do

循环执行的操作 循环执行的操作

done done

[root@svr7 ~]# vim /root/rsync.sh

#! /bin/bash

while inotifywait -rqq /opt

do

rsync -az --delete /opt/ root@192.168.4.207:/opt/

done

[root@svr7 ~]# chmod +x /root/rsync.sh

[root@svr7 ~]# /root/rsync.sh & #放入后台运行

[1] 8794

[root@svr7 ~]# touch /opt/haha.txt

[root@svr7 ~]# ls /opt/haha.txt

/opt/haha.txt

虚拟机pc207

[root@pc207 ~]# ls /opt/haha.txt

/opt/haha.txt

虚拟机svr7

[root@svr7 ~]# jobs #查看后台运行的脚本信息

[1]+ 运行中 /root/rsync.sh &

[root@svr7 ~]# killall rsync.sh #杀死运行的脚本

[1]+ 已终止 /root/rsync.sh

**Cobbler装机平台**

cobbler装机平台,不同版本的多系统安装

基本概念:

1. Cobbler是一款快速的网络系统部署工具
2. 集中管理所需服务,如DHCP、DNS、TFTP、Web
3. Cobbler内部集成了一个镜像版本仓库
4. Cobbler内部集成了一个ks应答文件仓库
5. Cobbler提供了包括yum源管理,web界面管理,API接口,电源管理等功能

一 安装一个CentOS虚拟机

真机:

[root@room9pc01 ~]# ls /ISO

CentOS-7-x86\_64-DVD-1708.iso

虚拟机要求:

1硬盘大小50G

2 网络类型private1

3 软件包选择”带GUI的服务器”

4 分区选择”自动分区”

5 管理员root密码为123456

6创建普通用户lisi

7 将CentOS放入光驱设备,搭建本地yum仓库 #利用本地yum仓库

8 配置IP地址:192.168.4.66/24

9 配置主机名: Cobbler.tedu.cn

二 虚拟机设置

1 设置防火墙为trusted

[root@cobbler ~]# firewall-cmd --set-default-zone=trusted

success

2 当前及永久设置selinux状态为permissive

[root@cobbler ~]# setenforce 0

[root@cobbler ~]# vim /etc/selinux/config

[root@cobbler ~]# getenforce

Permissive

三 利用scp真机传递Cobbler.zip包到虚拟机192.168.4.66中

[root@room9pc01 ~]# scp /root/桌面/Cobbler.zip root@192.168.4.66:/root/

四 搭建Cobbler装机平台

Cobbler概述软件,管理DHCP, tftp, web服务,自由的导入镜像与ks应答文件

1 解压Cobbler.zip包

[root@cobbler ~]# unzip /root/Cobbler.zip -d /

[root@cobbler ~]# unzip /Cobbler/cobbler.zip -d /opt/

[root@cobbler ~]# ls /opt/cobbler/

五 安装cobbler主程序,工具包等

[root@cobbler ~]# yum -y install /opt/cobbler/\*.rpm

[root@cobbler ~]# rpm -q cobbler

cobbler-2.8.2-1.el7.x86\_64

1.安装软件 cobbler cobbler-web dhcp tftp-server pykickstart httpd tftp-server

cobbler #cobbler程序包

cobbler-web #cobbler的web服务包

pykickstart #cobbler检查kickstart语法错误

httpd #Apache web服务

dhcp #Dhcp服务

tftp-server #tftp服务

2配置cobbler

[root@cobbler ~]# vim /etc/cobbler/settings **#直接搜索修改,不是写入**

......

next \_server: 192.168.4.66 #设置下一个服务器还为本机

server: 192.168.4.66 #设置本机为cobbler服务器

manage\_dhcp: 1 #设置cobbler管理dhcp服务

pxe\_just\_once: 1 #防止客户端重复安装操作系统

......

开机启动项: 匹配及停止 1、硬盘启动 2、光驱设备 3、U盘 4、网络引导

3 配置cobbler的dhcp

[root@cobbler ~]# vim /etc/cobbler/dhcp.template

:%s /192.168.1/192.168.4/g # 全文替换(5次替换,共4行)

4 绝对路径解压cobbler\_boot.tar.gz #众多的引导文件

[root@cobbler ~]# tar -tf /Cobbler/cobbler\_boot.tar.gz #查看包里面内容

[root@cobbler ~]# tar -xPf /Cobbler/cobbler\_boot.tar.gz **#绝对路径释放**

[root@cobbler ~]# ls /var/lib/cobbler/lgeteoaders/

5 启动相关服务

[root@cobbler ~]# systemctl restart cobblerd

[root@cobbler ~]# systemctl enable cobblerd

[root@cobbler ~]# systemctl restart httpd

[root@cobbler ~]# systemctl enable httpd

[root@cobbler ~]# systemctl restart tftp

[root@cobbler ~]# systemctl enable tftp

[root@cobbler ~]# systemctl restart rsyncd #同步服务

[root@cobbler ~]# systemctl enable rsyncd

或者

[root@Cobbler ~]# for i in cobblerd httpd tftp rsyncd

> do

> systemctl restart $i

> systemctl enable $i

> done

[root@Cobbler ~]#

6 同步刷新cobbler配置

[root@cobbler ~]# cobbler sync

......

\*\*\* TASK COMPLETE \*\*\* #看到complete说明成功,否则出错

[root@cobbler ~]# firefox <https://192.168.4.66/cobbler_web>

用户名：cobbler

密码：cobbler





**cobbler应用**

导入安装镜像数据

格式: cobbler import --path=挂载点 --name=导入系统命名(随意起)

cobbler导入的镜像放在: /var/www/cobbler/ks\_mirror

[root@Cobbler ~]# cobbler import --path=/dvd --name=CentOS7

[root@Cobbler ~]# ls /var/www/cobbler/ks\_mirror/

CentOS7 config

[root@cobbler ~]# cobbler list

安装客户端至少2G内存;安装完成机器默认root密码cobbler

cobbler list 查看有哪些系统

cobbler profile remove --name=CentOS7-x86\_64 #删除菜单信息

cobbler distro remove --name=CentOS7-x86\_64 #删除镜像信息

自定义应答文件: 开头注释行删除

[root@cobbler ~]# system-config-kickstart #生成ks文件

必须默认kickstart文件存放位置：/var/lib/cobbler/kickstarts/

修改kickstart文件：

[root@cobbler ~]# cobbler profile edit --name=CentOS7-x86\_64 --kickstart=/var/lib/cobbler/kickstarts/自定义.cfg

[root@cobbler ~]# cobbler profile report

[root@cobbler ~]# cobbler sync #同步配置

#########################################

**I-Day20**

**查漏补缺**

查询命令对应程序文件在哪: which 命令

关于默认网关的配置

如果是RHEL7.0

nmcli ..... ipv4.addresses “IPv4地址/掩码位数 默认网关”

如果是RHEL7.2及以上

nmcli ..... ipv4.gateway “默认网关”

LDAP客户端

使用authconfig-tui命令去配置, 配置过程中提示下载证书时先不要按OK, 另开终端连过去,把证书下载好以后再返回再ok

autofs配置

按题目要求只需要实现自定挂载ldapuserX一个用户; 如果配置适用于所有的用户也可以; 如果LDAP用户的家目录是通过NFSv3共享的,挂载参数需要添加v3

# vim /etc/策略文件

ldapuser0 -rw,v3 服务器地址:目录路径

NTP网络时间客户端有可能chrony包已经装好,但是/etc/chrony.conf 文件中有错误,可以先删除错误的配置文件

# rm -rf /etc/chrony.conf

再重装chrony包

# yum -y reinstall chrony

#########################################

**网络(10天) + shell脚本(7天)------王凯**

**II-Day01**

硬件方面: 线缆 网络设备 计算机

软件方面: 操作系统 应用系统

网络的作用:信息传递;资源共享;增加数据的安全;提高系统的处理性能(集群)

网络发展历史以及标志性技术

60年代: 网络的萌芽阶段 分组交换: 提高传输效率

70～80: TCP/IP 统一网络

90年代: web技术

标准化组织

ISO(国际标准化组织), IEEE(电气和电子工程师学会)

WiFi: 美国标准

wapi: 中国标准

WAN: 广域网, 几十~上百Km ISP: 网络服务供应商

LAN: 局域网,小于1KM; 需要个人或组织维护

网络设备生产厂商:

cisco(思科)

华为

路由交换设备:

cisco 2911路由器

cisco 2960交换机、3560三层交换机

拓扑结构

WAN

1. 点对点: 专用的广域网中电路连接的两台路由器

LAN

1. 星形: 易于实现,易于扩展,方便故障排查;中心节点压力大,组网成本较高
2. 网状: 一个节点与其他多个节点相连,提供冗余性和容错性,可靠性高,组网成本高

ISO颁布了OSI七层网络参考模型

OSI七层: 理论框架

TCP/IP五层: 实际使用

OSI网络参考模型（框架理论）

物理层，数据链路层，网络层，传输层，会话层，表示层，应用层

OSI的七层框架功能

|  |  |
| --- | --- |
| 应用层 | 网络服务与最终用户的一个接口 |
| 表示层 | 数据的表示、安全、压缩 |
| 会话层 | 建立、管理、中止会话 |
| 传输层 | 定义传输数据的协议端口号,以及流控和差错校验 |
| 网络层 | 进行逻辑地址(IP地址)寻址,实现不同网络之间的路径选择 |
| 数据链路层 | 建立逻辑连接、进行硬件地址寻址、差错校验等功能 |
| 物理层 | 建立、维护、断开物理连接 |

|  |  |
| --- | --- |
| **TCP/IP五层模型** | **OSI七层模型** |
| 应用层 | 应用层 |
| 表示层 |
| 会话层 |
| 传输层 | 传输层 |
| 网络层 | 网络层 |
| 数据链路层 | 数据链路层 |
| 物理层 | 物理层 |

协议:

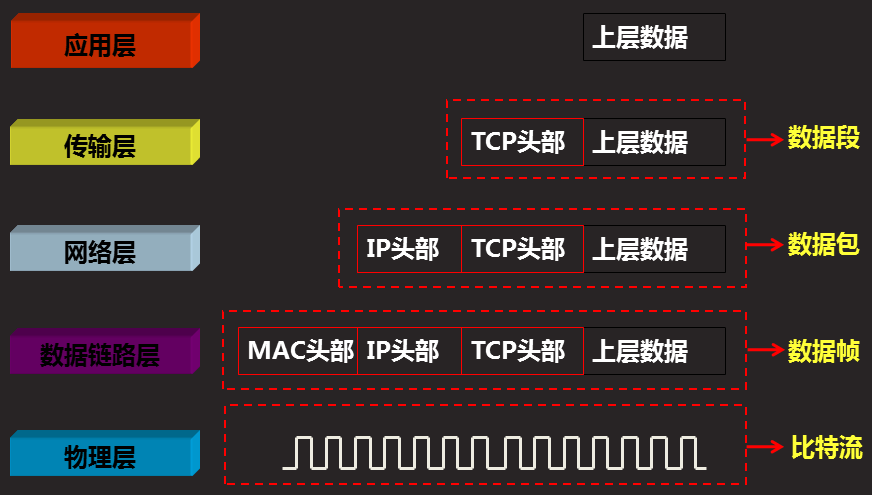
1) 为了使数据可以在网络上从源传递到目的地,网络上所有设备需要”讲”相同的”语言”

2) 描述网络通信中”语言”规范的一组规则就是协议

TCP/IP协议族的组成

|  |  |
| --- | --- |
| 应用层 | HTTP FTP TFTP SMTP SNMP DNS |
| 传输层 | TCP UDP |
| 网络层 | ICMP IGMP IP ARP RARP |
| 数据链路层 | 由底层网络定义的协议 |
| 物理层 |

协议数据单元(PDU)



设备与层的对应关系



**物理层解析**

物理层是TCP/IP模型的最底层

为数据端设备提供传送数据的通路

RJ-45: 俗称水晶头,网线

RJ-11: 电话线

有源: 网线 近距离传递 150米以内

无源: 光纤, 远距离传递 N公里以上 光纤熔接机

双绞线(网线),抗干扰

非屏蔽双绞线UTP 屏蔽双绞线STP

5类线(cat5) 100Mb

超5类(cat5e) 100Mb

6类(cat6) 1000Mb

7类(cat7) 10000Mb

双绞线制作规范

T568b标准：白橙，橙，白绿，蓝，白蓝，绿，白棕，棕

T568a标准：白绿，绿，白橙，蓝，白蓝，橙，白棕，棕

直通线: 网线端使用相同标准连接

交叉线: 网线端使用不同标准连接

全反线：以T568b为例，一边是T568b的线序，另一边是：棕，白棕，绿，白蓝，蓝，白绿，橙，白橙

连接交换机必有直通线,其他设备都用交叉线

中继器: 放大信号; 延长网络传输距离

连接设备的终端软件:

1) crt

2) xshell

win2008密码：

Taren1

交换机连接pc设备方法

1) 使用console线连接交换机console口

2) pc端连接rs232接口

3) pc端在desktop选项卡中打开终端（terminal）, 点确定。

**交换机命令行模式:** ?:相当于Linux中的双tab

**1) 用户模式:**

Switch>

**2) 特权模式**(一般用于查看配置信息)

Switch>enable

Switch#

**3) 全局配置模式**(所做的配置对整个设备生效)

Switch#configure terminal

Switch(config)#

**4) 接口模式**

Switch(config)#interface fastEthernet 0/1 0:模块号 1: 接口号

Switch(config-if)#

2）交换机工作模式的退出下：

Switch(config-if)#exit //返回上一模式

Switch(config-if)#end //返回特权模式，或者快捷键ctrl+z

命令输入错误卡死后同时按ctrl+shift+6(只能模拟器用)

交换机命令行配置: 注意命令行模式

Switch(config)# hostname s1 //修改主机名

s1# show running-config //查看设备配置(按空格翻页)

Switch(config)#enable password 123 //配置密码

Switch(config)#no enable password //删除密码

Switch#write //保存配置, 可以缩写w

Switch#copy running-config startup-config //保存配置

Switch#reload //重启

Switch#erase startup-config //清空设置,清空后重启生效, 按w/write可以救回

**配置禁用DNS查询**

Switch(config)#no ip domain-lookup //防止命令行卡死,实际工作中用

**配置输入同步**

Switch(config)#line console 0

Switch(config-line)#logging synchronous //防止系统提示信息,打断用户命令输入

**配置控制台会话时间永不超时**

Switch(config)#line console 0

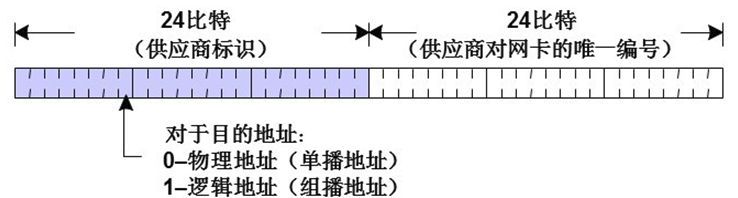
Switch(config-line)#exec-timeout 0 0 //几分几秒后锁屏

#######################################

**II-Day02**

**数据链路层解析**

MAN地址/硬件地址/物理地址: 用来识别一个以太网上的某个单独的设备或一组设备,全球唯一,设备出厂就有,不需自己配置,16进制标示





MAC地址长度48位(6个字节)，前24位代表厂商，后24位代表网卡编号，

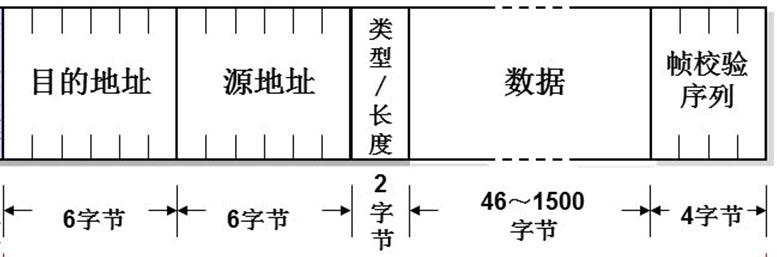
MAC地址的第8位为0时表示该MAC地址为单播地址，为1时表示组播地址。48位都为1表示广播地址

单播: 一对一的数据传递

组播: 一对多的数据传递

广播: 一对所有的数据传递

数据链路层的协议数据单元:



1字节(byte)=8位(bit)

交换机

1、交换机是用来连接局域网的主要设备，可以实现数据帧的转发

2、交换机的工作原理:

学习: 学习源mac地址

广播: 除数据来源的接口之外的所有接口

转发: 正常通信

更新: 如果超过300秒无任何数据通信,交换机会清除mac地址表记录;

如果所连接主机关机或重启,交换机会清除mac地址记录;

ipconfig /all : windows系统中查看mac地址

查看MAC地址表:

Switch#show mac-address-table

广播域: 广播域指接收同样广播消息的节点的集合;交换机的所有端口默认属于同一个广播域

划分广播域(VLAN)的好处:

1) 限制广播; 2)增加安全; 3) 提高带宽利用; 4)降低延迟

创建VLAN(虚拟局域网)的最大数量: 4096

默认交换机拥有VLAN1, 所有接口都在VLAN1中

Switch(config)#vlan 2 //创建vlan2

Switch(config-vlan)#name xiaoshou //给vlan2定义名称(不支持中文)

Switch#show vlan brief //查看vlan摘要信息

Switch(config)#in f0/3 //进入f0/3接口

Switch(config-if)#switchport access vlan 2 //把f0/3接口加入vlan2

Switch(config)#in range f0/5-6 //同时进入5、6接口

Switch(config)#no vlan vlan-id //删除已创建的vlan

中继链路trunk/干道

ISL-------Cisco私有协议

IEEE 802.1Q-------公有

access: 接入链路,承载1个vlan,仅支持一个vlan(为一个vlan服务)

truck: 中继链路, 承载多个vlan, 能够为数据帧打所属vlan的标签

为接口配置**中继链路/trunk**

Switch(config)#in f0/7

Switch(config-if)#switchport mode truck //将接口设置为truck

**以太通道/接口聚集/以太链路聚集/以太接口捆绑**

配置以太通道,提高链路速度与可靠

Switch(config)#interface range fastEthernet 0/1-2

Switch(config-if-range)#channel-group 1 mode on //将两个接口捆绑为以太通道

Switch#show etherchannel summary //查看以太通道

#####################################

**II-Day03**

IP分类

A: 1~127 255.0.0.0

B: 128~191 255.255.0.0

C: 192~223 255.255.255.0

子网掩码中1代表网络位, 0代表主机位

一个IP地址段中第一个地址叫网络ID,最后一个地址叫广播地址

交换机: 组建局域网络

路由器: 连接不同范围的主机

路由器: 网关

in g0/1 //进入路由接口

no shutdown //开启接口

no ip address //删除接口的IP地址

ip address 192.168.1.254 255.255.255.0 //配置ip地址,思科不支持/24

ICMP协议: 提供网络状态信息

ping -t ip地址 //一直ping,不停止

ping -l 数据大小 ip地址 //指定数据大小的ping

Switch#show ip route //查看路由表

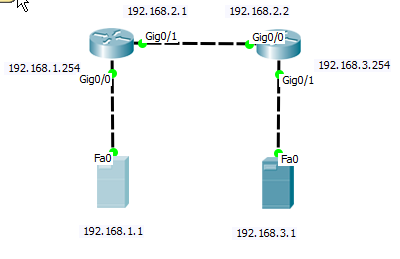
以C开头的为直连路由,产生条件是接口开启并且配置了ip地址

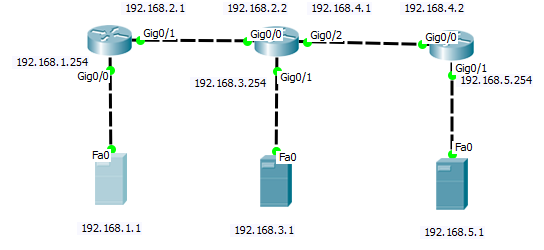
以S开头的为静态路由: 由管理员手工配置,要实现数据互通需要配置来回两个方向,否则有去无回

设置静态路由:

格式: ip route 目标IP地址网段 子网掩码 下一跳IP地址

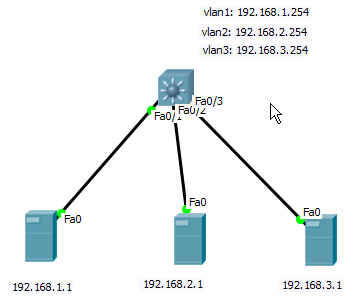
ip route 192.168.3.0 255.255.255.0 192.168.2.2





**注意:** 有错误的静态路由必须删掉,不然会有50%的丢包率

**三层交换机配置**



1, 创建vlan2; vlan3

2, 将接口加入相应vlan

3, 开启路由功能 Switch(config)#ip routing

4, 进入vlan接口(虚拟接口),配置ip地址, 作为客户机的网关

Switch(config)#interface vlan 1

Switch(config-if)#ip address 192.168.1.254 255.255.255.0

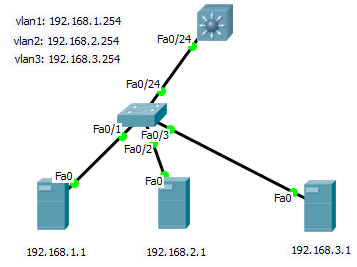
Switch(config-if)#no shutdown //开启vlan1, 只有第一个需要此操作

Switch(config)#interface vlan 2

Switch(config-if)#ip address 192.168.2.254 255.255.255.0

Switch(config)#interface vlan 3

Switch(config-if)#ip address 192.168.3.254 255.255.255.0



1. 增加二层交换机,创建vlan2 vlan3
2. 并将相应接口加入vlan

Switch(config)#in f0/2

Switch(config-if)#switchport access vlan 2

Switch(config)#in f0/3

Switch(config-if)#switchport access vlan 3

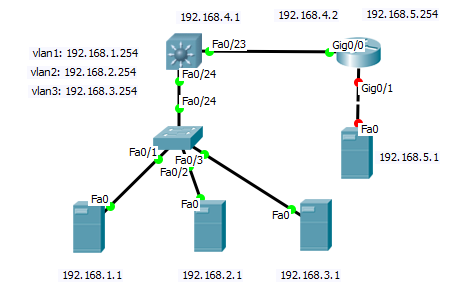
1. f0/24接口设置为中继链路

4) 在三层交换机上配置Trunk并指定接口封装为802.1q

Switch(config)#interface fastEthernet 0/24

Switch(config-if)#switchport trunk encapsulation dot1q //指定truck协议为IEEE 802.1Q

Switch(config-if)#switchport mode trunk



在三层交换机中:

Switch(config)#in f0/23

Switch(config-if)#no switchport //将交换机接口转换为路由接口

Switch(config-if)#ip address 192.168.4.1 255.255.255.0 //为转换好的接口配置ip地址

**默认路由(S\*)**: 是一种特殊的静态路由,可以达到允许设备访问任意网段的效果

在三层交换机中

Switch(config)#ip route 0.0.0.0 0.0.0.0 192.168.4.2 //配置默认路由

默认路由方向必须一致对外,不然会出现路由环路

**动态路由**: 基于某种动态路由协议的配置,可以自动学习路由条目

ospf: 动态路由协议, 开放式最短路径优先,适合大中型网络使用

宣告: 宣告自己直接连接的网段

首先删除路由器的所有静态路由,然后配置动态路由

Router(config)#router ospf 1 //启动ospf路由进程(1:进程号)

Router(config-router)#network 192.168.4.0 0.0.0.255 area 0

然后在三层交换机中配置动态路由ospf

Router(config)#router ospf 1

反掩码

指定ospf协议运行的接口和所在的区域

Router(config-router)#network 192.168.1.0 0.0.0.255 area 0

Router(config-router)#network 192.168.2.0 0.0.0.255 area 0

Router(config-router)#network 192.168.3.0 0.0.0.255 area 0

Router(config-router)#network 192.168.4.0 0.0.0.255 area 0

#########################################

**II-Day04**

网络层传输点到点的连接

传输层提供端到端的连接

传输层协议:

1、TCP:传输控制协议

1) 可靠的、面向连接的协议

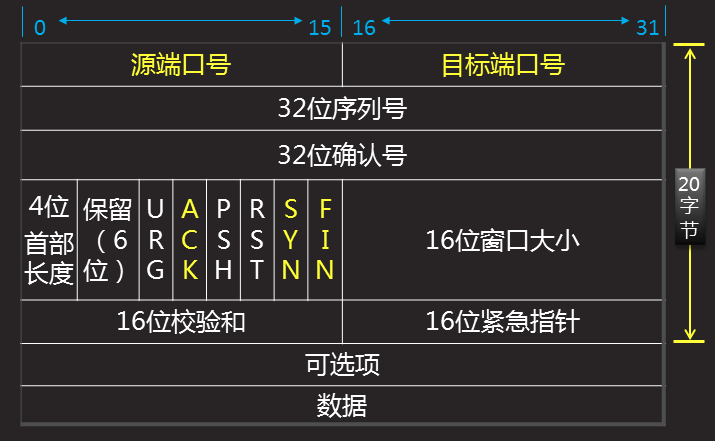
2) 传输效率低

2、UDP:用户数据报协议

1) 不可靠的、无连接的服务

2) 传输效率高

TCP的封装格式



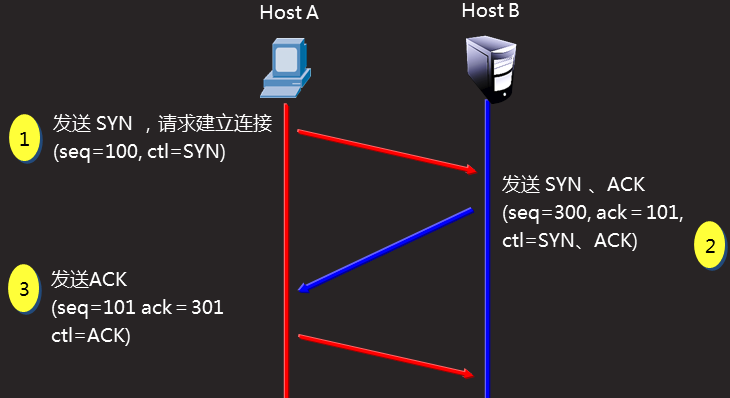
TCP标志位

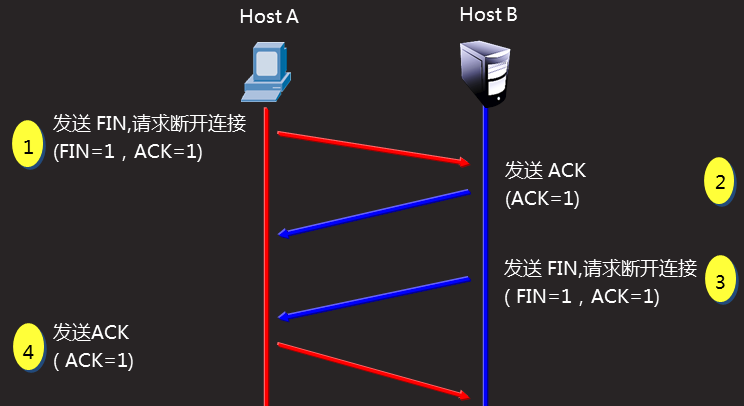
SYN: 想与对方建立连接

ACK: 确认对方信息

FIN: 想与对方断开连接

TCP三次握手、四次断开





TCP的应用:

|  |  |  |
| --- | --- | --- |
| **端口** | **协议** | **说明** |
| 21 | FTP | 文件传输协议,用于上传、下载 |
| 23 | Telnet | 用于远程登录,通过连接目标计算机的这一端口,得到验证后可以远程控制管理目标计算机 |
| 25 | SMTP | 简单邮件传输协议,用于发送邮件 |
| 53 | DNS | 域名服务,当用户输入网站的名称后,由DNS负责它解析成IP地址,这个过程中用到的端口号是53 |
| 80 | HTTP | 超文本传输协议,通过HTTP实现网络上超文本的传输 |

UDP的封装格式:



UDP的应用

|  |  |  |
| --- | --- | --- |
| **端口** | **协议** | **说明** |
| 69 | TFTP | 简单的文件传输协议 |
| 53 | DNS | 域名服务 |
| 123 | NTP | 网络时间协议 |

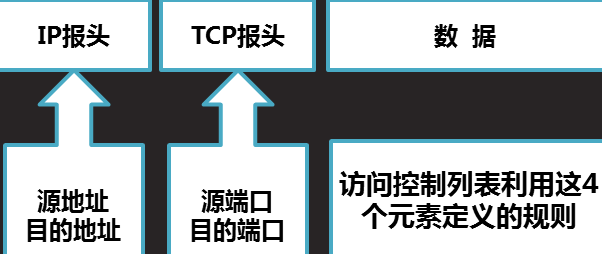
端口号范围: 0~65535 (0~1023基本被知名服务占用了)

访问控制列表作用

访问控制列表

1) 读取第三层、第四层头部信息

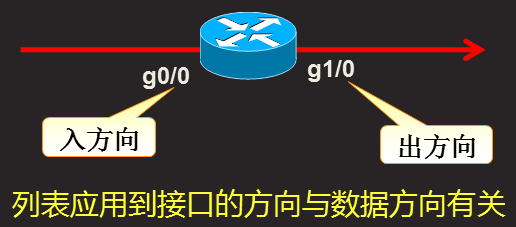
2) 根据预先定义好的规则对数据进行过滤



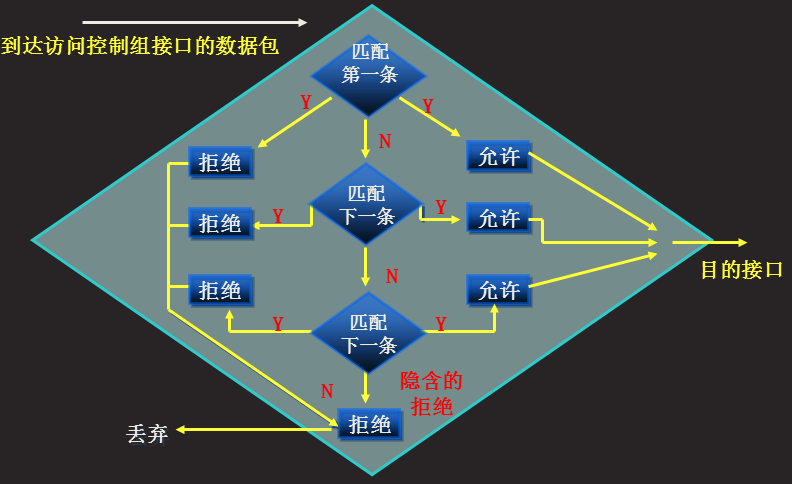
访问控制列表在接口应用的方向:

出: 已经过路由器的处理,正离开路由器接口的数据

入: 已到达路由器接口的数据包,将被路由器处理



访问控制列表的处理过程: 匹配即停止,不允许蒙混过关



**访问控制列表的类型：**

标准访问控制列表

1) 基于**源IP地址**过滤数据包

2) 标准访问控制列表的访问控制列表号是1～99

扩展访问控制列表

1) 基于源IP地址、目的IP地址、指定协议、端口等来过滤数据包

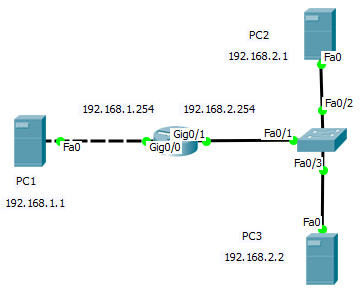
2) 扩展访问控制列表的访问控制列表号是100~199

反掩码: 0: 严格匹配 1: 不匹配

permit-----允许 deny----拒绝

案例1:标准acl的配置

禁止主机PC2与PC1通信,而允许所有其他的流量



反掩码: 严格匹配

Router(config)#access-list 1 deny 192.168.2.1 0.0.0.0 //使用标准acl,拒绝192.168.2.1通过

(Router(config)#access-list 1 deny host 192.168.2.1 )

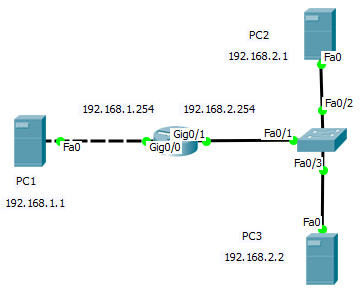
Router(config)#in g0/1 //进入接口

Router(config-if)#ip access-group 1 in //放置acl列表1

Router(config)#access-list 1 permit any //设置acl允许所有主机,因为有个隐含的拒绝

案例2:标准acl的配置

允许主机PC2与PC1互通,而禁止其他设备访问PC1



Router(config)#access-list 2 permit host 192.168.2.1 //使用标准acl允许主机192.168.2.1通过

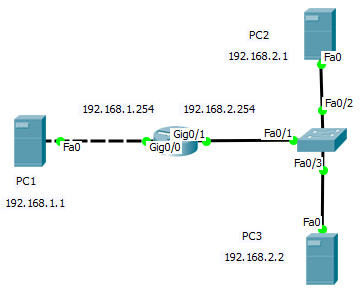
Router(config)#in g0/1 //进入接口

Router(config-if)#ip access-group 2 in //放置acl列表2

Router#show access-lists //查看acl列表

案例3:扩展acl的配置

通过配置扩展acl禁止pc2访问pc1的ftp服务,禁止pc3访问pc1的www服务,所有主机的其他服务不受限制



目标IP地址

源IP地址

Router(config)#access-list 100 deny tcp host 192.168.2.1 host 192.168.1.1 eq 21 //拒绝192.168.2.1访问192.168.1.1的ftp服务

Router(config)#in g0/1 //进入接口

Router(config-if)#ip access-group 100 in //放置acl列表100

Router(config)#access-list 100 permit ip host 192.168.2.1 host 192.168.1.1 //放行192.168.2.1访问192.168.1.1

Router(config)#access-list 100 deny tcp host 192.168.2.2 host 192.168.1.1 eq 80

Router(config)#access-list 100 permit ip host 192.168.2.2 host 192.168.1.1

简化

Router(config)#access-list 100 deny tcp host 192.168.2.1 host 192.168.1.1 eq 21

Router(config)#access-list 100 deny tcp host 192.168.2.2 host 192.168.1.1 eq 80

Router(config)#access-list 100 permit ip any any //放行所有设备对其他主机的访问

Router(config)#in g0/1 //进入接口

Router(config-if)#ip access-group 100 in

私有IP地址分类

A类: 10.0.0.0--10.255.255.255

B类: 172.16.0.0--172.31.255.255

C类: 192.168.0.0--192.168.255.255

动态公有地址-----个人

固定共有地址-----企业

**NAT**

NAT: network Address Translation 网络地址转换

作用：通过将内部网络的私有IP地址翻译成全球唯一的公网IP地址，使内部网络可以连接到互联网等外部网络上

优点：1)节省公有合法IP地址 2)处理地址重叠 3)安全性

缺点: 1)延迟增大 2)配置和维护的复杂性

实现方式:

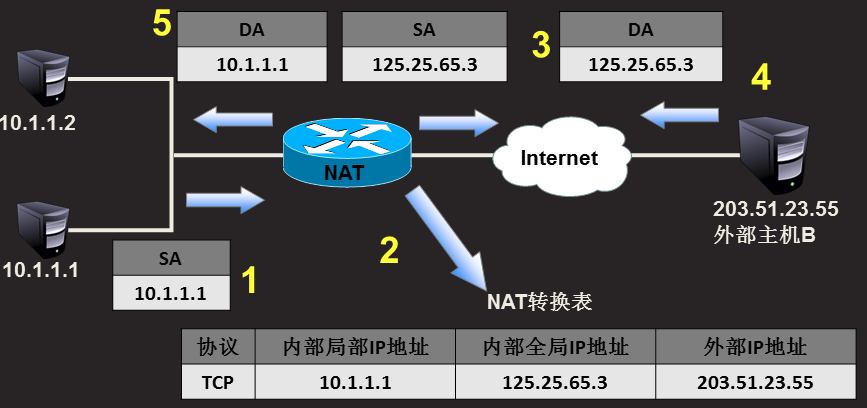
1. **静态转换SA**: 一对一,一般运用在服务器**发布服务**到公网时使用

端口映射: 可以更改对外发布服务的端口号; 限制对外发布服务的数量

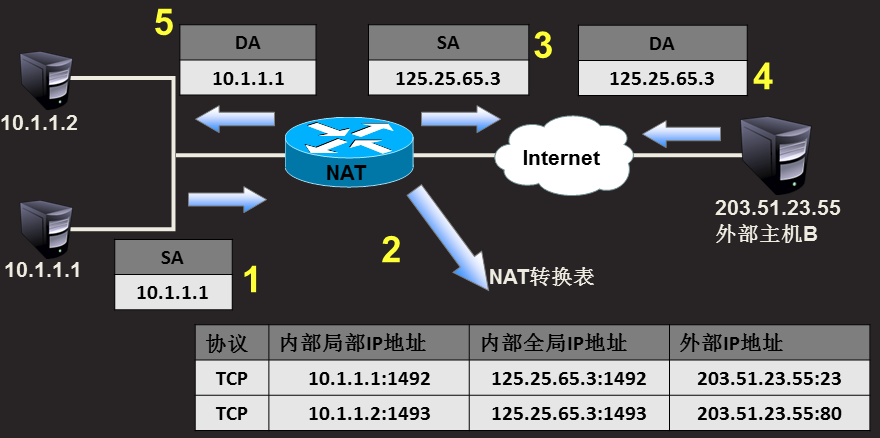
Router(config)#ip nat inside source static tcp 192.168.1.1 80 100.0.0.2 8080

2)**端口多路复用PAT**: 一对多,一般运用在员工主机**访问公网**时使用

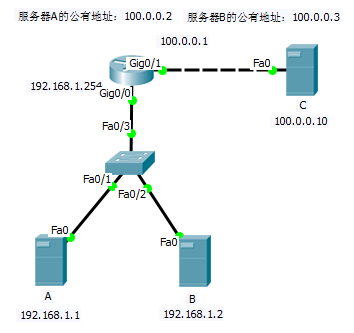
静态转换的工作过程:



PAT的工作过程:



**配置nat静态转换:**



Router(config)#ip nat inside source static 192.168.1.1 100.0.0.2 //访问外网时将192.168.1.1转化为100.0.0.2

Router(config)#interface g0/1

Router(config-if)#ip nat outside //在接口中开启nat功能,外边

Router(config)#interface g0/0

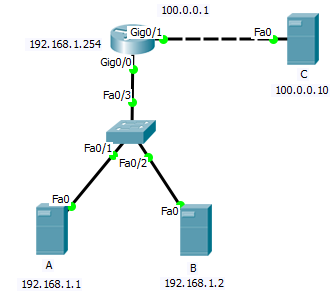
Router(config-if)#ip nat inside //在接口中开启nat功能,里边

Router(config)#ip nat inside source static 192.168.1.2 100.0.0.3

Router#debug ip nat //开启nat排错功能

Router#undebug ip nat //关闭nat排错功能

**配置端口多路复用**



Router(config)#access-list 1 permit 192.168.1.0 0.0.0.255 //定义允许访问外网的主机,名称为列表1

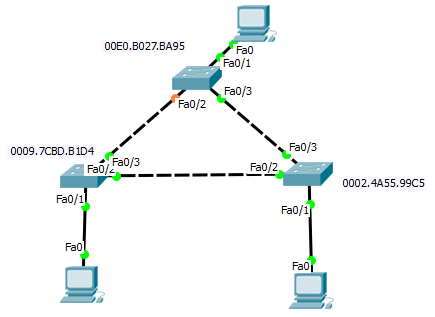
Router(config)#ip nat inside source list 1 interface g0/1 overload //使列表1中的主机可以通过g0/1接口访问外网

允许A和B可以同时访问外网,不然会依次排队

####################################################################

**II-Day05**

广播风暴



STP生成树协议: 防止广播风暴;备份线路

Switch(config)#spanning-tree vlan 1 //开启STP生成树协议(默认是开启的)

网桥ID (BID): 网桥ID是唯一的,交换机之间选择BID值最小的交换机作为网络中的根网桥,取值范围0~65535



网桥优先级: 默认值为32768 ,修改的话必须是4096的倍数(0和4096也可以), 越小越优先

阻塞接口BLK: 断开的接口

转发接口FWD: 能通信的接口

Switch#show version //查看版本信息,包含MAC地址

Switch#show spanning-tree vlan 1 //查看vlan 1的生成树详细信息

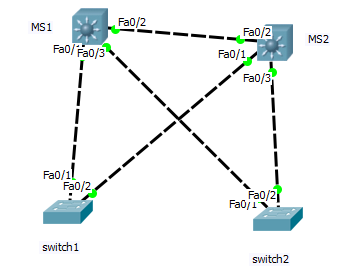
Switch(config)#spanning-tree vlan 1 priority 28672 //修改网桥优先级,用来指定根网桥

Switch(config)#spanning-tree vlan 1 root primary //直接指定根网桥

Switch(config)#spanning-tree vlan 1 root secondary //直接指定次网桥

**PVST+ :每vlan生成树-------实现网络的负载分担(负载均衡)**

案例:通过配置PVST+实现MS1负责转发VLAN1的数据，MS2负责转发VLAN2的数据



Switch(config)#vlan 2 //在所有的交换机中创建vlan2

在MS1中配置

Switch(config)#interface range fastEthernet 0/1-3

Switch(config-if-range)#switchport truck encapsulation dot1q

Switch(config-if-range)#switchport mode trunk

Switch#show spanning-tree vlan 2

在MS2中配置

Switch(config)#interface range fastEthernet 0/1-3

Switch(config-if-range)#switchport truck encapsulation dot1q

Switch(config-if-range)#switchport mode trunk

Switch#show spanning-tree vlan 2

在MS1中配置

Switch(config)#spanning-tree vlan 1 root primary //设置为vlan1的主根

Switch(config)#spanning-tree vlan 2 root secondary //设置为vlan2的次根

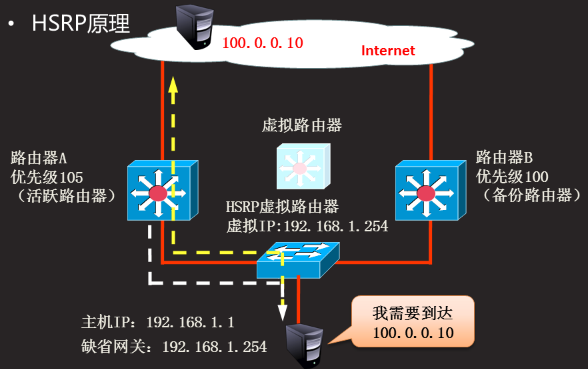
在MS2中配置

Switch(config)#spanning-tree vlan 2 root primary //设置为vlan2的主根

Switch(config)#spanning-tree vlan 1 root secondary //设置为vlan1的次根

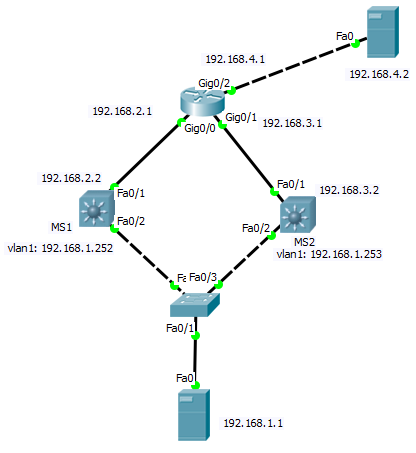
**热备份路由选择协议**

HSRP (Hot Standby Routing Protocol)



默认优先级为100,越大越优先

按照图所示拓扑结构，在三层交换机配置热备份路由协议使组内两个出口设备共享一个虚拟ip地址192.168.1.254为内网主机的网关



1. 配置接口的ip地址,vlan的ip地址

interface vlan 1

ip address 192.168.1.252 255.255.255.0

no shutdown

1. 三层交换机开启路由功能,并通过动态路由ospf宣告直连网段

MS1命令:

ip routing //开启路由功能

router ospf 1 //进入ospf

network 192.168.1.0 0.0.0.255 area 0 //宣告直连网段

network 192.168.2.0 0.0.0.255 area 0

1. 路由器开启ospf宣告直连网段

router ospf 1 //进入ospf

network 192.168.2.0 0.0.0.255 area 0 //宣告直连网段

network 192.168.3.0 0.0.0.255 area 0 //宣告直连网段

network 192.168.4.0 0.0.0.255 area 0 //宣告直连网段

4、192.168.1.1使用临时网关192.168.1.252或者192.168.1.253测试与192.168.4.2能否连通

5、配置HSRP

Switch(config)#interface vlan 1 //进入vlan

Switch(config-if)#standby 1 ip 192.168.1.254 //开启热备份功能,组号1,虚拟路由器ip是192.168.1.254

**占先权**: 当本机优先级大于其他主机时,立刻成为活跃路由器

Switch(config-if)#standby 1 priority 105 //配置HSRP的优先级为105

Switch(config-if)#Standby 1 preempt //配置占先权

Switch#show standby brief //查看HSRP摘要信息

####################################

**II-Day06**

将Windows2008文件下载到真机,或者给2008上传文件

1. 关闭真机防火墙和selinux (默认都已经关闭)
2. 设置/var/ftp/pub目录增加w权限

chmod o+w /var/ftp/pub

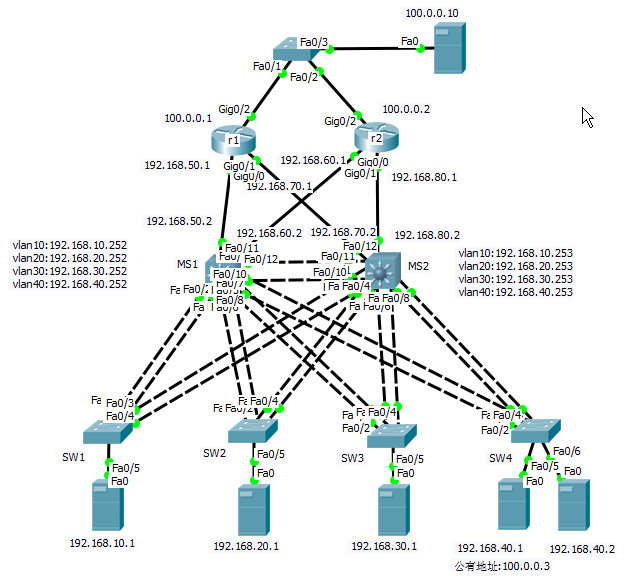
1. 修改/etc/vsftpd/vsftpd.conf

打开后按27gg跳跃到27行,将#去掉,保存退出

1. 重启ftp服务

systemctl restart vsftpd

1. 进入2008打开文件夹,在地址栏输入ftp://192.168.4.254, 进入pub目录上传或者下载所需文件(需要将win2008IP地址设置为192.168.4.0网段)



**步骤一：二层交换机配置**

1. 分别创建VLAN10、20、30、40

sw1将f0/5接口加入vlan10

1. Switch(config)#interface fastEthernet 0/5
2. Switch(config-if)#switchport access vlan 10

sw2将f0/5接口加入vlan20

1. Switch(config)#interface fastEthernet 0/5
2. Switch(config-if)#switchport access vlan 20

sw3将f0/5接口加入vlan30

1. Switch(config)#interface fastEthernet 0/5
2. Switch(config-if)#switchport access vlan 30

sw4将f0/5-6接口加入vlan40

1. Switch(config)#interface range fastEthernet 0/5-6
2. Switch(config-if)#switchport access vlan 40

每台设备捆绑以太通道，将f0/1与f0/2捆绑为通道1，f0/3与f0/4捆绑为通道2

1. Switch(config)#interface range f0/1-2
2. Switch(config-if-range)#channel-group 1 mode on
3. Switch(config)#interface range f0/3-4
4. Switch(config-if-range)#channel-group 2 mode on

5. Switch#show etherchannel summary //查看以太通道汇总信息

依次进入所有二层交换机的以太通道接口，配置中继链路

1. Switch(config)#interface port-channel 1
2. Switch(config-if)#switchport mode trunk
3. Switch(config)#interface port-channel 2
4. Switch(config-if)#switchport mode trunk

**步骤二：三层交换机基本配置**

1. 每台设备分别创建VLAN10、20、30、40
2. 1-2口捆绑为通道1
3. 3-4口捆绑为通道2
4. 5-6口捆绑为通道3
5. 7-8口捆绑为通道4
6. 9-10口捆绑为通道5

依次进入三层交换机的5个通道接口，配置中继链路（两台三层交换机配置相同）

1. Switch(config)#interface port-channel 1
2. Switch(config-if)# switchport trunk encapsulation dot1q
3. Switch(config-if)#switchport mode trunk
4. Switch(config)#interface port-channel 2
5. Switch(config-if)# switchport trunk encapsulation dot1q
6. Switch(config-if)#switchport mode trunk
7. Switch(config)#interface port-channel 3
8. Switch(config-if)# switchport trunk encapsulation dot1q
9. Switch(config-if)#switchport mode trunk
10. Switch(config)#interface port-channel 4
11. Switch(config-if)# switchport trunk encapsulation dot1q
12. Switch(config-if)#switchport mode trunk
13. Switch(config)#interface port-channel 5
14. Switch(config-if)# switchport trunk encapsulation dot1q
15. Switch(config-if)#switchport mode trunk

配置三层交换机vlan10、20、30、40的ip地址

1. Switch(config)#interface vlan 10
2. Switch(config-if)#ip address 192.168.10.252 255.255.255.0
3. Switch(config)#interface vlan 20
4. Switch(config-if)#ip address 192.168.20.252 255.255.255.0
5. Switch(config)#interface vlan 30
6. Switch(config-if)#ip address 192.168.30.252 255.255.255.0
7. Switch(config)#interface vlan 40
8. Switch(config-if)#ip address 192.168.40.252 255.255.255.0

注意：另外一台三层交换机配置的ip地址是253

1. Switch(config)#interface vlan 10
2. Switch(config-if)#ip address 192.168.10.253 255.255.255.0
3. Switch(config)#interface vlan 20
4. Switch(config-if)#ip address 192.168.20.253 255.255.255.0
5. Switch(config)#interface vlan 30
6. Switch(config-if)#ip address 192.168.30.253 255.255.255.0
7. Switch(config)#interface vlan 40
8. Switch(config-if)#ip address 192.168.40.253 255.255.255.0

**步骤三：三层交换机高级配置**

配置生成树协议，产生负载均衡效果。

MS1配置PVST+ 使其成为vlan10、20的主根 vlan30、40的次根

1. Switch(config)#spanning-tree vlan 10 root primary
2. Switch(config)#spanning-tree vlan 20 root primary
3. Switch(config)#spanning-tree vlan 30 root secondary
4. Switch(config)#spanning-tree vlan 40 root secondary

MS2配置PVST+ 使其成为vlan30、40的主根 vlan10、20的次根

1. Switch(config)#spanning-tree vlan 30 root primary
2. Switch(config)#spanning-tree vlan 40 root primary
3. Switch(config)#spanning-tree vlan 10 root secondary
4. Switch(config)#spanning-tree vlan 20 root secondary

配置热备份路由协议，完善负载均衡效果。

MS1配置HSRP 使其成为vlan10、20的活跃路由器 vlan30、40的备份路由器

1. Switch(config)#interface vlan 10
2. Switch(config-if)#standby 10 ip 192.168.10.254
3. Switch(config-if)#standby 10 priority 105
4. Switch(config-if)#standby 10 preempt
5. Switch(config)#interface vlan 20
6. Switch(config-if)#standby 20 ip 192.168.20.254
7. Switch(config-if)#standby 20 priority 105
8. Switch(config-if)#standby 20 preempt
9. Switch(config)#interface vlan 30
10. Switch(config-if)#standby 30 ip 192.168.30.254
11. Switch(config)#interface vlan 40
12. Switch(config-if)#standby 40 ip 192.168.40.254

13. Switch#show standby brief //查看热备份状态

MS2配置HSRP 使其成为vlan30、40的活跃路由器 vlan10、20的备份路由器

1. Switch(config)#interface vlan 30
2. Switch(config-if)#standby 30 ip 192.168.30.254
3. Switch(config-if)#standby 30 priority 105
4. Switch(config-if)#standby 30 preempt
5. Switch(config)#interface vlan 40
6. Switch(config-if)#standby 40 ip 192.168.40.254
7. Switch(config-if)#standby 40 priority 105
8. Switch(config-if)#standby 40 preempt
9. Switch(config)#interface vlan 10
10. Switch(config-if)#standby 10 ip 192.168.10.254
11. Switch(config)#interface vlan 20
12. Switch(config-if)#standby 20 ip 192.168.20.254

开启两台三层交换机的路由功能，并设置每个服务器所在vlan的网关

1. Switch(config)#ip routing

然后测试目前网络是否可以达成全网互通。

**步骤四：动态路由配置**

配置动态路由协议，使所有内网互通。

在ms1中开启ospf动态路由，并宣告直连网段

1. Switch(config)#router ospf 1
2. Switch(config-router)#network 192.168.10.0 0.0.0.255 area 0
3. Switch(config-router)#network 192.168.20.0 0.0.0.255 area 0
4. Switch(config-router)#network 192.168.30.0 0.0.0.255 area 0
5. Switch(config-router)#network 192.168.40.0 0.0.0.255 area 0
6. Switch(config-router)#network 192.168.50.0 0.0.0.255 area 0
7. Switch(config-router)#network 192.168.60.0 0.0.0.255 area 0

在ms2中开启ospf动态路由，并宣告直连网段

1. Switch(config)#router ospf 1
2. Switch(config-router)#network 192.168.10.0 0.0.0.255 area 0
3. Switch(config-router)#network 192.168.20.0 0.0.0.255 area 0
4. Switch(config-router)#network 192.168.30.0 0.0.0.255 area 0
5. Switch(config-router)#network 192.168.40.0 0.0.0.255 area 0
6. Switch(config-router)#network 192.168.70.0 0.0.0.255 area 0
7. Switch(config-router)#network 192.168.80.0 0.0.0.255 area 0

在r1中开启ospf动态路由，并宣告直连网段

1. Router(config)#router ospf 1
2. Router(config-router)#network 192.168.50.0 0.0.0.255 area 0
3. Router(config-router)#network 192.168.70.0 0.0.0.255 area 0

在r2中开启ospf动态路由，并宣告直连网段

1. Router(config)#router ospf 1
2. Router(config-router)#network 192.168.60.0 0.0.0.255 area 0
3. Router(config-router)#network 192.168.80.0 0.0.0.255 area 0

查看所有三层设备路由表，应该是统一状态

1. show ip route

**步骤五：NAT配置**

配置r1与r2的nat功能，使内网服务器40.1映射到外网100.0.0.3，并在接口中开启

1. Router(config)#ip nat inside source static 192.168.40.1 100.0.0.3
2. Router(config)#in g0/2
3. Router(config-if)#ip nat outside
4. Router(config-if)#in range g0/0-1
5. Router(config-if-range)#ip nat inside

在r1与r2中配置默认路由之后，使用ospf宣告自己是默认信息源（表示自己有通往外网的默认路由）

1. Router(config)#ip route 0.0.0.0 0.0.0.0 100.0.0.10
2. Router(config)#router ospf 1
3. Router(config-router)#default-information originate

#########################################

**II-Day07**

子网划分

255.255.254.0

11111111.11111111.11111110.00000000

172.16.0.0/23

128+64+32+16+8+4+2+1=255

**搭建BBS论坛**

1 clone-vm7 //创建新虚拟机,增加内存为2G

2 获取Discuz\_X3.3\_SC\_UTF8文件到真机

3 使用root账户,密码123456登录新虚拟机, init 5进入图形化界面

4 配置IP地址192.168.4.100/24

5 搭建yum仓库,

准备镜像源,将iso镜像放入虚拟机光驱 mount /dev/cdrom /mnt

[root@localhost ~]# yum repolist //检测yum仓库

6 安装论坛所需软件包mariadb-server mariadb php php-mysql

[root@localhost ~]# yum -y install mariadb-server mariadb php php-mysql

7 开启服务

[root@localhost ~]# systemctl start mariadb.service

[root@localhost ~]# systemctl enable mariadb.service

[root@localhost ~]# systemctl start httpd

[root@localhost ~]# systemctl enable httpd

8 关闭防火墙和selinux

[root@localhost ~]# systemctl stop firewalld

[root@localhost ~]# setenforce 0

9 拷贝论坛文件到虚拟机

[root@room9pc01 ~]#scp /root/桌面/Discuz\_X3.3\_SC\_UTF8.zip [root@192.168.4.100:/](mailto:root@192.168.4.100:/)

10 解压压缩文件到/opt目录中

[root@localhost ~]# unzip /Discuz\_X3.3\_SC\_UTF8.zip -d /opt

11 拷贝论坛文件到http服务中

[root@localhost ~]# cp -rf /opt/upload/\* /var/www/html/

12 修改目录所属主

[root@localhost ~]# cd /var/www/html/

[root@localhost html]# chown -R apache config/ data/ template/ uc\_client/ uc\_server/

13 打开浏览器,输入虚拟机IP地址,安装论坛,安装过程中需要清除数据库密码,再设置论坛管理员密码

#########################################

**II-shell-Day01**

shell: 在Linux内核与用户之间的解释器程序

1. 通常指/bin/bash
2. 负责向内核翻译及传达用户/程序指令
3. 相当于操作系统的”外壳”

[root@desktop0 ~]# cat /etc/shells

/bin/sh #多数Unix默认的shell

/bin/bash #多数Linux默认使用的shell

/sbin/nologin #非登录shell

/usr/bin/sh

/usr/bin/bash

在RHEL系统中,实际上sh是bash的符号链接

/usr/sbin/nologin

/bin/tcsh

/bin/csh

[root@desktop0 ~]# yum -y install ksh

[root@desktop0 ~]# cat /etc/shells | grep ksh

/bin/ksh

/bin/bash的优点

1 支持tab键

2 历史记录

3 支持管道

4 支持别名

5 快捷键

6 重定向

|  |  |
| --- | --- |
| **快捷键** | **描述** |
| Ctrl+a | 将光标移至行首 |
| Ctrl+e | 将光标移至行尾 |
| Ctrl+c | 中止操作 |
| Ctrl+d | 一般为结束输入 |
| Ctrl+m | 回车 |
| Ctrl+u | 删除光标至行首的所有内容 |
| Ctrl+w | 删除光标前面的一个单词(空格分隔) |
| Ctrl+s | 挂起、冻结终端 |
| Ctrl+q | 解除冻结终端 |
| Alt+. | 使用前一个命令的最后一个词 |
| 方向键(上下键) | 历史命令 |
| Tab键 | 补齐命令、选项、路径与文件名  (补齐选项需要bash-completion软件包) |

[root@desktop0 ~]# vim /etc/profile

......

HISTSIZE=10 //修改历史记录条数

......

[root@desktop0 ~]# history -c //临时删除历史记录(清除历史记录缓存)

[root@desktop0 ~]# > /root/.bash\_history //清空历史记录文件

**重定向标准输入/输出/错误输出**

标准输入<（stdin），描述号为0;将文本输入来源由键盘改为指定的文件

[root@desktop0 ~]# mail -s Error root < /etc/passwd

标准输出>（stdout），描述号为1；

标准错误2>（stderr），描述号为2。

追加重定向: >> 2>>

&> : 正确与错误的都收集走

&>> :

标准的shell脚本构成:

1. 声明解释器
2. 脚本功能描述 #脚本功能:
3. 具体代码

shell脚本的执行方式:

法1: 作为”命令字”: 指定脚本文件的路径,前提是有x权限

法2: 作为”参数” :使用bash、sh来加载脚本文件,不要求x权限

法3: source来加载脚本文件,不要求x权限

案例: 要求编写三个脚本程序，分别实现以下目标：

1) 在屏幕上输出一段文字“Hello World”

2) 能够为本机快速配好Yum仓库

3) 能够为本机快速装配好vsftpd服务

[root@desktop0 ~]# vim /opt/a.sh

#! /bin/bash

echo 'hello world'

rm -rf /etc/yum.repos.d/\*.repo

echo “[rhel]

name=rhel

baseurl=http://classroom.example.com/content/rhel7.0/x86\_64/dvd/

enabled=1

gpgcheck=0” > /etc/yum.repos.d/dvd.repo

echo '当前yum可用软件数量'

yum repolist | tail -1

yum -y install vsftpd &> /dev/null

systemctl start vsftpd

systemctl enable vsftpd

echo '服务开启成功'

[root@desktop0 ~]# bash /opt/a.sh #开启新解释器执行脚本

hello world

当前yum可用软件数量

注意空格

repolist: 4305

服务开启成功

[root@desktop0 ~]# . /opt/a.sh

hello world

当前yum可用软件数量

repolist: 4,305 (source /opt/a.sh=. /opt/a.sh)

服务开启成功

[root@desktop0 ~]# source /opt/a.sh #不开启新解释器执行脚本

hello world

当前yum可用软件数量

repolist: 4,305

服务开启成功

**自定义变量**,不能使用数字开头或者特殊符号定义变量名称

若要撤销已有的变量，可使用unset命令

[root@desktop0 ~]# a=10 #自定义变量a

[root@desktop0 ~]# echo $a

10

[root@desktop0 ~]# unset a #取消定义变量a

[root@desktop0 ~]# echo $a #查看时无结果

**环境变量PWD、USER、HOME、SHELL、UID、PATH**

[root@desktop0 ~]# echo $PWD $USER $HOME $SHELL $UID

/root root /root /bin/bash 0

[root@desktop0 ~]# echo $PS1

[\u@\h \W]\$ #一级提示符

\u 用户名、\h 主机名、\W 工作目录、\$ 权限标识

[root@desktop0 ~]# echo $PS2

> #二级提示符

[root@desktop0 ~]# echo $PATH

/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin

[root@desktop0 ~]# vim /opt/c.sh

#! /bin/bash

echo hello

[root@desktop0 ~]# bash /opt/c.sh

hello

[root@desktop0 ~]# cp /opt/c.sh /usr/bin/

[root@desktop0 ~]# chmod +x /usr/bin/c.sh

[root@desktop0 ~]# c.sh

hello

**预定义变量$0、$$、$?、$#、$\***

**位置变量$1、$2、......${10}、${11}……**  
echo $0             //脚本的名称

echo $1              //第一个参数

echo $2              //第二个参数

echo $\*               //显示所有的位置变量

echo $#             //所有位置变量的个数

echo $$             //当前进程的进程号

echo $?              //上一个程序的返回状态码 0:表示正确 非0:表示错误

echo $! //上一个进程的进程号

[root@desktop0 ~]# vim /opt/test1.sh

#! /bin/bash

useradd $1

echo $2 | passwd --stdin $1

[root@desktop0 ~]# bash /opt/test1.sh dc 123

更改用户 dc 的密码 。

passwd：所有的身份验证令牌已经成功更新。

引号作用: 界定范围

单引号’’ : 取消特殊符号的作用

反撇号`` : 将命令的执行输出作为变量值

[root@desktop0 ~]# a=10

[root@desktop0 ~]# echo $a

10

[root@desktop0 ~]# echo "$a"

10

[root@desktop0 ~]# echo '$a'

$a

[root@desktop0 ~]# echo \$a

$a

[root@desktop0 ~]# stty -echo #将回显功能关闭

[root@desktop0 ~]# stty echo #将回显功能恢复

[root@desktop0 ~]# vim /opt/d.sh

#! /bin/bash

read -p "请输入用户名:" n

useradd $n

stty -echo #关闭回显

read -p "请输入密码:" p

stty echo #恢复回显

echo #恢复回显后补一个空行

echo $p | passwd --stdin $n

[root@desktop0 ~]# chmod +x /opt/d.sh

[root@desktop0 ~]# /opt/d.sh

请输入用户名:tc

请输入密码:

更改用户 tc 的密码 。

passwd：所有的身份验证令牌已经成功更新。

**全局变量**

默认情况下，自定义的变量为局部变量，只在当前Shell环境中有效，而在子Shell环境中无法直接使用。比如已定义的SCHOOL变量，当进入到sh或bash子Shell后，变量SCHOOL将处于未定义的状态

若希望定义的变量能被子进程使用，可以使用export命令将其发布为全局变量。使用export发布时，只需指定变量名（可以有多个）即可，也可以通过export命令直接设置新的全局变量

export 局部变量名[=变量值] #为局部变量添加全局属性

export -n 全局变量名 #取消指定变量的全局属性

[root@desktop0 ~]# yy="世界您好"

[root@desktop0 ~]# echo $yy

世界您好

[root@desktop0 ~]# bash

[root@desktop0 ~]# echo $yy #输出空值

[root@desktop0 ~]# exit

exit

[root@desktop0 ~]# export yy

[root@desktop0 ~]# export xx=1234

[root@desktop0 ~]# bash

[root@desktop0 ~]# echo $yy

世界您好

[root@desktop0 ~]# echo $xx

1234

[root@desktop0 ~]# exit

exit

[root@desktop0 ~]# echo $xx

1234

#########################################

**II-shell-Day02**

**shell中的数值运算**

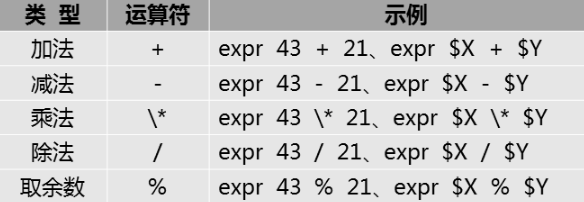
1) 使用expr、$[ ]、let等整数运算工具

2) 使用bc实现小数运算操作

**expr运算工具(不改变变量原本的值)**

格式: expr 整数1 运算符 整数2 ...

乘法操作应采用\\*转义,避免被作为shell通配符; 参与运算的整数值与运算操作符之间需要以空格分开,引用变量时必须加$符号



[root@desktop0 ~]# x=1234

[root@desktop0 ~]# expr $x + 78

1312

[root@desktop0 ~]# expr $x - 78

1156

[root@desktop0 ~]# expr $x \\* 78

96252

[root@desktop0 ~]# expr $x / 78

15

[root@desktop0 ~]# expr $x % 78

64

**$[]或$(())表达式**

格式: $[整数1运算符整数2 ...]

乘法操作\*无需转义，运算符两侧可以无空格；引用变量可省略$符号；计算结果替换表达式本身，可结合echo命令输出。

[root@desktop0 ~]# x=1234

[root@desktop0 ~]# $[x+78]

bash: 1312: 未找到命令...

[root@desktop0 ~]# echo $[x+78]

1312

[root@desktop0 ~]#

[root@desktop0 ~]# echo $[x-78]

1156

[root@desktop0 ~]# echo $[x\*78]

96252

[root@desktop0 ~]# echo $[x/78]

15

[root@desktop0 ~]# echo $[x%78]

64

**使用let命令**

expr或者$[]、$(())方式只进行运算,并不会改变变量的值;而let命令可以直接对变量值做运算再保存新的值,因此变量x=12,在执行let运算后的值会变更;另外,let运算操作并不会显示结果,但是可以结合echo命令来查看

|  |  |
| --- | --- |
| 简写表达式 | 完整表达式 |
| i++ | i=i+1 |
| i-- | i=i-1 |
| i+=2 | i=i+2 |
| i-=2 | i=i-2 |
| i\*=2 | i=i\*2 |
| i/=2 | i=i/2 |
| i%=2 | i=i%2 |

[root@desktop0 ~]# x=1234

[root@desktop0 ~]# let y=x+22

[root@desktop0 ~]# echo $y

1256

[root@desktop0 ~]# x=1234

[root@desktop0 ~]# let y=x+22

[root@desktop0 ~]# echo $y

1256

[root@desktop0 ~]# let x++;echo $x

1235

[root@desktop0 ~]# let x--;echo $x

1234

[root@desktop0 ~]# let x+=78; echo $x

1312

[root@desktop0 ~]# let x-=78;echo $x

1234

[root@desktop0 ~]# let x\*=78;echo $x

96252

[root@desktop0 ~]# let x/=78;echo $x

1234

[root@desktop0 ~]# let x%=78;echo $x

0

**使用bc实现小数运算**

1. 支持高精度的数值运算
2. 直接运行bc可进入交互式运算界面,quit退出
3. 设置scale=n可约束小数位,除此以外也受参与运算的数值的小数位影响
4. 多个表达式以分号分隔

[root@desktop0 ~]# a=12.34

[root@desktop0 ~]# echo "$a\*34.34" | bc #单表达式

423.75

[root@desktop0 ~]# echo "scale=4;$a\*12.1234;5/3" | bc #多表达式

149.6027

1.6666

[root@desktop0 ~]# echo "scale=4;12.34+5.678" | bc

18.018

**小数值的比较**

格式: echo “数值1 比较符 数值2” | bc

1. 如果表达式成立,则返回的计算结果为1,否则为0
2. 常见的比较操作: >、>=、<、<=、 ==、 !=

[root@desktop0 ~]# a=12.34;b=56.78

[root@desktop0 ~]# echo "$a<=$b" | bc

1

[root@desktop0 ~]# echo "$a>$b" | bc

0

**常见的测试操作**

1) 使用“test 表达式”或者[ 表达式 ]都可以，表达式两边至少要留一个空格。

2) 条件测试操作本身不显示出任何信息。测试的条件是否成立主要体现在命令执行后的返回状态（即 $?），所以可以在测试后查看变量$?的值来做出判断，或者结合&&、||等逻辑操作显示出结果（或作其他操作）

A && B #仅当A命令执行成功,才执行B命令

A || B #仅当A命令执行失败,才执行B命令

A ; B #执行A命令后执行B命令,两者没有逻辑关系

A && B || C

**字符串比较**

[ 操作符 字符串 ]

|  |  |
| --- | --- |
| 操作符 | 含义 |
| -z | 检查变量的值是否未设置(空值) |
| -n | 测试变量是否不为空(相当于! -z) |

[root@desktop0 ~]# [ ! -z $var1 ];echo $?

0

[root@desktop0 ~]# var1=nb;var3=""

[root@desktop0 ~]# [ -z $var1 ] && echo 空值 || echo 非空值

非空值

[root@desktop0 ~]# [ -z $var2 ] && echo 空值 || echo 非空值

空值

[ 字符串1 操作符 字符串2 ]

|  |  |
| --- | --- |
| 操作符 | 含义 |
| == | 两个字符串相同 |
| != | 两个字符串不相同 |

[root@desktop0 ~]# [ $USER == root ]

[root@desktop0 ~]# echo $?

0

**整数值比较**

[ 整数值1 操作符 整数值2 ]

|  |  |
| --- | --- |
| 操作符 | 含义 |
| -eq | 等于(equal) |
| -ne | 不等于(not equal) |
| -ge | 大于或等于(greater or equal) |
| -le | 小于或等于(lesser or equal) |
| -gt | 大于(greater than) |
| -lt | 小于(lesser than) |

**文件状态测试**

[ 操作符 文件或目录 ]

|  |  |
| --- | --- |
| 操作符 | 含义 |
| -e | 判断对象是否存在,若存在则结果为真 |
| -d | 判断对象是否为目录,是则为真 |
| -f | 判断对象是否为一般文件,是则为真 |
| -r | 判断对象是否有可读权限,是则为真 |
| -w | 判断对象是否有可写权限,是则为真 |
| -x | 判断对象是否有可执行权限,是则为真 |

**逻辑分隔操作**

格式: 命令1 操作符 命令2 ......

格式: [ 条件1 ] 操作符 [ 条件2 ]......

|  |  |
| --- | --- |
| 操作符 | 含义 |
| && | 给定条件必须都成立,整个测试结果才为真 |
| || | 只要其中一个条件成立,则整个测试结果为真 |

[root@desktop0 ~]# [ $USER == root ] && [ $PWD == /root ]

[root@desktop0 ~]# [ $? -eq 0 ] && echo yes

yes

[root@desktop0 ~]# [ -d /opt/testdir ] || mkdir /opt/testdir

[root@desktop0 ~]# ls -ld /opt/testdir

drwxr-xr-x. 2 root root 6 1月 10 19:28 /opt/testdir

案例: 每2分钟检测系统登录的账号,如果超过3个,给管理员发送报警邮件

[root@desktop0 opt]# vim /opt/a.sh

#! /bin/bash

[ `who | wc -l` -gt 3 ] && echo "有非法用户登录" | mail -s "警报" root

或者:

n=`who | wc -l`

[ $n -gt 3 ] && mail -s “警报” root < /opt/w.txt #需要编辑邮件内容

[root@desktop0 opt]# echo 有非法用户登录 > /opt/w.txt

[root@desktop0 opt]# chmod +x /opt/a.sh

[root@desktop0 opt]# crontab -e -u root

\*/2 \* \* \* \* /opt/a.sh

案例: 判断/opt中文件是否有文件ABC,如果有则显示文件已经存在,如果没有则创建文件

[root@desktop0 opt]# [ -f /opt/ABC ] && echo 文件已经存在 || touch /opt/ABC

[root@desktop0 opt]# ls | grep ABC

ABC

案例: 判断/opt中是否有mnt目录,如果没有则创建

[root@desktop0 opt]# vim /opt/c.sh

#! /bin/bash

if [ ! -d /opt/mnt ];then

mkdir /opt/mnt

fi

[root@desktop0 opt]# chmod +x /opt/c.sh

[root@desktop0 opt]# /opt/c.sh

案例: 测试与某台主机是否可以连通,如果可以连通则显示通了,否则显示不通

为了节省ping测试时间，可以只发送3个测试包（-c 3）、缩短发送测试包的间隔秒数（-i 0.2）、等待反馈的超时秒数（-W 1）

-c : ping的次数

-i : ping的间隔秒数

-W(大写) : ping反馈结果的等待秒数

[root@desktop0 opt]# vim /opt/d.sh

#! /bin/bash

ping -c 3 -i 0.2 -W 1 $1 &> /dev/null

if [ $? -eq 0 ];then

echo 通了

else

echo 不通

fi

[root@desktop0 opt]# chmod +x /opt/d.sh

[root@desktop0 opt]# /opt/d.sh 172.25.254.254

通了

[root@desktop0 opt]# /opt/d.sh 172.25.254.253

不通

案例: 猜数字游戏

[root@desktop0 opt]# vim /opt/e.sh

#! /bin/bash

a=$[RANDOM%11] #RANDOM: 系统自带的系统变量,值为0-32767随机数

read -p "请输入一个数字(0-10)" n

if [ $n -eq $a ];then

echo "猜对了"

exit

elif [ $n -gt $a ];then

echo "猜大了"

else

echo "猜小了"

fi

[root@desktop0 opt]# bash /opt/e.sh

请输入一个数字6

猜小了

#########################################

**II-shell-Day03**

在Shell脚本应用中，常见的for循环采用遍历式、列表式的执行流程，通过指定变量从值列表中循环赋值，每次复制后执行固定的一组操作。

for 变量名 in {1..5} for 变量名 in `seq 5`

do do

seq n 列出 1~n 的数;

seq n m 列出 n~m 的数;

seq n step m列出n~m,以 step为步长的数

命令序列 命令序列

done done

[root@desktop0 opt]# vim /opt/test1.sh

#! /bin/bash

for i in `seq $1`

do

echo nb

done

[root@desktop0 opt]# chmod +x /opt/test1.sh

[root@desktop0 opt]# /opt/test1.sh 100

案例: 使用for循环编写测试多台主机是否连通的脚本(真机操作,方便验证)

[root@room9pc01 ~]# vim /opt/test2.sh

#! /bin/bash

for ip in `seq 151 158`

do

ping -c 3 -i 0.2 -W 1 176.19.8.$ip &> /dev/null

[ $? -eq 0 ] && echo "176.19.8.$ip 通了" || echo "176.19.8.$ip 不通"

done

[root@room9pc01 ~]# chmod +x /opt/test2.sh

[root@room9pc01 ~]# /opt/test2.sh

[root@room9pc01 ~]# vim /opt/test9.sh

#! /bin/bash

a=0

b=0

for ip in `seq 151 158`

do

ping -c 3 -i 0.2 -W 1 176.19.8.$ip &> /dev/null

if [ $? -eq 0 ];then

echo "176.19.8.$ip 通了"

let a++

else

echo "176.19.8.$ip 不通"

let b++

fi

done

echo "$a台通 $b台不通"

[root@room9pc01 ~]# chmod +x /opt/test2.sh

[root@room9pc01 ~]# /opt/test9.sh

176.19.8.151 通了

176.19.8.152 不通

176.19.8.153 不通

176.19.8.154 通了

176.19.8.155 不通

176.19.8.156 不通

176.19.8.157 通了

176.19.8.158 通了

4台通 4台不通

案例: 获取一份员工名单,并为这些员工创建账号,并设置不同的密码

[root@desktop0 opt]# vim /opt/test3.sh

#! /bin/bash

n=`cat /opt/users | wc -l`

for i in `seq $n`

do

useradd `head -$i /opt/users | tail -1` &> /dev/null

echo `head -$i /opt/passwd | tail -1` | passwd --stdin `head -$i /opt/users | tail -1` &> /dev/null

done

[root@desktop0 opt]# chmod +x /opt/test3.sh

[root@desktop0 opt]# /opt/test3.sh

**while循环**属于条件式的执行流程，会反复判断指定的测试条件，只要条件成立即执行固定的一组操作，直到条件变化为不成立为止。所以while循环的条件一般通过变量来进行控制，在循环体内对变量值做相应改变，以便在适当的时候退出，避免陷入死循环。while循环的语法结构如下所示：

while 条件测试 while :

do do

命令序列 命令序列

done done

案例: 使用while循环写猜数游戏,直到猜对为止

[root@desktop0 opt]# vim /opt/test4.sh

#! /bin/bash

x=$[RANDOM%100]

m=0

while :

do

read -p "请输入一个[0-99]的数字:" n

let m++

if [ -z $n ];then

exit

elif [ $n -eq $x ];then

echo "猜对了"

echo "你一共猜了$m次"

exit

elif [ $n -lt $x ];then

echo "猜小了"

else

echo "猜大了"

fi

done

[root@desktop0 opt]# chmod +x /opt/test4.sh

[root@desktop0 opt]# /opt/test4.sh

请输入一个[0-99]的数字:50

猜小了

请输入一个[0-99]的数字:58

猜对了

你一共猜了2次

**case分支**属于匹配执行的方式，它针对指定的变量预先设置一个可能的取值，判断该变量的实际取值是否与预设的某一个值相匹配，如果匹配上了，就执行相应的一组操作，如果没有任何值能够匹配，就执行预先设置的默认操作。

(case分支,功能和if类似,可以理解为简化版本)

case分支的语法结构如下所示：

case $变量 in

模式1)

命令序列1 ;; #注意双分号

模式2)

命令序列2 ;; #注意双分号

.. ..

\*)

默认命令序列 #注意没有双分号

esac

[root@desktop0 opt]# vim /opt/test5.sh

#! /bin/bash

case $1 in

a|-a) # 此处’|’相当于或者的意思

echo aaa;;

b)

echo bbb;;

\*)

echo ccc

esac

[root@desktop0 opt]# chmod +x /opt/test5.sh

[root@desktop0 opt]# /opt/test5.sh a

aaa

[root@desktop0 opt]# /opt/test5.sh -a

aaa

[root@desktop0 opt]# /opt/test5.sh b

bbb

[root@desktop0 opt]# /opt/test5.sh c

ccc

案例: 编写脚本,使用 read 提示用户输入一个字符,使用 case 语句判断用户输入的是:数字、字符、还是其他符号。提示:[a-Z]代表字符,[0-9]代表数字,其他属于符号。

[root@server0 opt]# vim /opt/test5.sh

#! /bin/bash

read -p "请输入一个字符:" n

case $n in

[a-z]|[A-Z])

echo $n是字符;;

[0-9])

echo $n是数字;;

\*)

echo $n是符号

esac

[root@server0 opt]# chmod +x /opt/test5.sh

[root@server0 opt]# /opt/test5.sh

请输入一个字符:4

4是数字

[root@server0 opt]# /opt/test5.sh

请输入一个字符:t

t是字符

[root@server0 opt]# /opt/test5.sh

请输入一个字符:!

!是符号

在shell脚本中，将一些需重复使用的操作，定义为公共的语句块，即可称为**函数**。通过使用函数，可以使脚本代码更加简洁，增强易读性，提高Shell脚本的执行效率

1）函数的定义方法

格式1： 格式2：

function 函数名 { 函数名() {

命令序列 命令序列

.. .. .. ..

} }

[root@desktop0 opt]# abc (){

> mkdir $1

> cd $1

> }

[root@desktop0 opt]# type abc

abc 是函数

abc ()

{

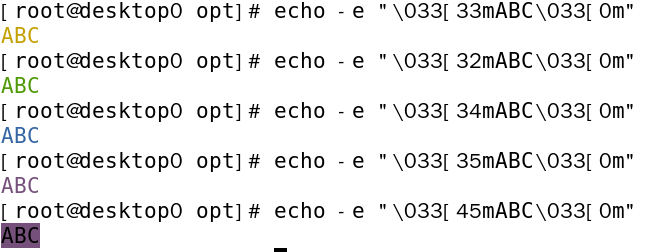
mkdir $1;

cd $1

}

[root@desktop0 opt]# abc aa

[root@desktop0 aa]#



\033[ : 固定格式----记住

3X为字体颜色，4X为背景颜色

ABC : 编辑的内容

0m : 还原颜色

[root@desktop0 opt]# vim /opt/test6.sh

#! /bin/bash

cecho(){ #使用函数

echo -e "\033[$1m$2\033[0m" #函数中执行的命令,实现彩色文字效果

}

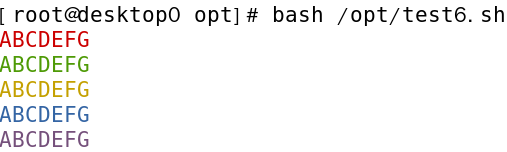
cecho 31 ABCDEFG #调用名称为cecho的函数

cecho 32 ABCDEFG

cecho 33 ABCDEFG

cecho 34 ABCDEFG

cecho 35 ABCDEFG



**shell版本的fork炸弹**

[root@desktop0 opt]# vim /opt/test7.sh

#! /bin/bash

.(){

.|.&

}

.

[root@desktop0 opt]# chmod +x /opt/test7.sh

[root@desktop0 opt]#/opt/test7.sh #死机(CPU耗尽)

**中断及退出**

通过break、continue、exit在Shell脚本中实现中断与退出的功能。

break可以结束整个循环；

continue结束本次循环，进入下一次循环；

exit结束整个脚本

[root@desktop0 opt]# vim /opt/test8.sh

#! /bin/bash

for i in {1..5}

do

echo $i

[ $i -eq 3 ] && continue

echo 123

done

echo abc

[root@desktop0 opt]# /opt/test8.sh

1

123

2

123

3

4

123

5

123

abc

[root@desktop0 opt]# vim /opt/test9.sh

#! /bin/bash

for i in {1..5}

do

echo $i

break

echo 123

done

echo abc

[root@desktop0 opt]# /opt/test9.sh

1

abc

案例: 让用户输入数字,每输入一次,都与之前的数字相加,如果输入0则退出脚本,并显示之前所有数字之和

[root@desktop0 opt]# vim /opt/test10.sh

#! /bin/bash

m=0

while :

do

read -p "请输入数字(0是退出):" n

[ $n -eq 0 ] && break || let m+=n

done

echo 总和是:$m

[root@desktop0 opt]# chmod +x /opt/test10.sh

[root@desktop0 opt]# /opt/test10.sh

请输入数字(0是退出):56

请输入数字(0是退出):10

请输入数字(0是退出):0

总和是:66

案例: 编写脚本文件，找出1-20内6的倍数，并打印她的平方值

[root@desktop0 opt]# vim /opt/test11.sh

#! /bin/bash

for i in {1..20}

do

[ $[i%6] -ne 0 ] && continue

echo $[i\*i]

done

[root@desktop0 opt]# chmod +x /opt/test11.sh

[root@desktop0 opt]# /opt/test11.sh

36

144

324

案例: 真实机编写脚本使用函数批量远程虚拟机 desktop 和 server,给两台虚拟机安装 vsftpd 软件包。启动 classroom,再启动 desktop 和 server。

[root@room9pc01 opt]# vim /opt/test12.sh

#! /bin/bash

pssh(){

ssh $1 $2

}

for i in desktop0 server0

do

pssh $i "yum -y install vsftpd &> /dev/null" &

done

[root@room9pc01 opt]# bash /opt/test12.sh

[root@desktop0 opt]# rpm -q vsftpd

vsftpd-3.0.2-9.el7.x86\_64

[root@server0 ~]# rpm -q vsftpd

vsftpd-3.0.2-9.el7.x86\_64

案例: 编写脚本将/etc/\*.conf 文件,每个文件单独使用 tar 命令备份一份,备份文件名包含时间标记,但是不备份/etc/yum.conf 文件(备注:遇/etc/yum.conf 则跳过备份),备份文件可以放在任何位置。

[root@server0 opt]# vim /opt/test8.sh

#! /bin/bash

x=0

for i in `ls /etc/\*.conf`

do

let x++

[ $i == "/etc/yum.conf" ] && continue

tar -cPjf /opt/`date +%F`-$x.tar.bz2 $i

done

[root@server0 opt]# bash /opt/test8.sh

echo -n 可以不换行输出

\n 换行符

\r 回车符

#########################################

**II-shell-Day04**

**字符串截取及切割**

最小值为0

**子串/字串截取的三种用法：**

最小值为1

1) ${变量名:起始位置:长度}

2) expr substr "$变量名" 起始位置 长度

最小值为1

3) echo $变量名 | cut -b 起始位置-结束位置

[root@desktop0 opt]# a=123456

[root@desktop0 opt]# echo ${#a} #查看变量字符长度

6

[root@desktop0 opt]# a=abcdef

[root@desktop0 opt]# echo $a

abcdef

[root@desktop0 opt]# echo ${a:0:3}

abc

[root@desktop0 opt]# echo ${a:3:3}

def

[root@desktop0 opt]# echo ${a:10:3} #空值

[root@desktop0 opt]# echo ${a:3:10}

def

[root@desktop0 opt]# echo ${a::3} #默认从0开始查找,0可以省略

abc

[root@desktop0 opt]# echo ${a:0:} #空值

[root@desktop0 opt]# echo ${a:0}

abcdef

[root@desktop0 opt]# expr substr $a 1 1 #从第一个字符开始截取一个

a

[root@desktop0 opt]# expr substr $a 2 2 #从第二个字符开始截取二个

bc

[root@desktop0 opt]# echo $a | cut -b 1-2

ab

[root@desktop0 opt]# echo $a | cut -b 1

a

[root@desktop0 opt]# echo $a | cut -b 2

b

[root@desktop0 opt]# echo $a | cut -b 2-

bcdef

[root@desktop0 opt]# echo $a | cut -b -3

abc

[root@desktop0 opt]# echo $a | cut -b 1,3

ac

[root@desktop0 opt]# echo $a | cut -b 1-3,5

abce

随机显示一个字符的脚本

[root@desktop0 opt]# vim /opt/text1.sh

#! /bin/bash

x=$[RANDOM%62]

a=abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789

echo ${a:$x:1}

#! /bin/bash

x=$[RANDOM%62+1]

a=abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789

echo $a | cut -b $x

#! /bin/bash

x=$[RANDOM%62+1]

a=abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789

expr substr $a $x 1

[root@desktop0 opt]# chmod +x /opt/text1.sh

[root@desktop0 opt]# /opt/text1.sh

l

[root@desktop0 opt]# /opt/text1.sh

z

显示八个随机字符的脚本

[root@desktop0 opt]# vim /opt/test2.sh

#! /bin/bash

a=abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789

for i in `seq 8`

do

n=$[RANDOM%62+1]

x=`echo $a | cut -b $n`

p=$p$x

done

echo $p

[root@desktop0 opt]# chmod +x /opt/test2.sh

[root@desktop0 opt]# /opt/test2.sh

mYKz7qDB

[root@desktop0 opt]# /opt/test2.sh

I9uzgfvm

案例: 创建一个用户,并配置随机8位密码的脚本

[root@desktop0 opt]# vim /opt/test3.sh

#! /bin/bash

a=abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789

for i in `seq 8`

do

n=$[RANDOM%62+1]

x=`echo $a | cut -b $n`

p=$p$x

done

useradd $1

echo $p | passwd --stdin $1 &> /dev/null

echo $1的密码是$p >> /opt/passwd.txt

[root@desktop0 opt]# chmod +x /opt/test3.sh

[root@desktop0 opt]# /opt/test3.sh dc

[root@desktop0 opt]# cat /opt/passwd.txt

dc的密码是gXuQSAZ8

**子串替换的两种用法：并不改变变量的值**

[root@desktop0 opt]# a=abcdef

[root@desktop0 opt]# echo ${a/a/x}

xbcdef

[root@desktop0 opt]# echo $a

abcdef

1) 只替换第一个匹配结果：${变量名/old/new}

2) 替换全部匹配结果：${变量名//old/new}

[root@desktop0 opt]# a=abcdef

[root@desktop0 opt]# echo ${a/a/x}

xbcdef

[root@desktop0 opt]# a=abadef

[root@desktop0 opt]# echo ${a/a/x}

xbadef

[root@desktop0 opt]# echo ${a/ab/x}

xadef

[root@desktop0 opt]# a=aaacef

[root@desktop0 opt]# echo ${a//a/x}

xxxcef

**字符串掐头去尾：不会改变变量的值**

1) 从左向右，最短匹配删除：${变量名#\*关键词}

2) 从左向右，最长匹配删除：${变量名##\*关键词}

3) 从右向左，最短匹配删除：${变量名%关键词\*}

4) 从右向左，最长匹配删除：${变量名%%关键词\*}

[root@desktop0 opt]# a=`head -1 /etc/passwd`

[root@desktop0 opt]# echo $a

root:x:0:0:root:/root:/bin/bash

[root@desktop0 opt]# echo ${a#root}

:x:0:0:root:/root:/bin/bash

[root@desktop0 opt]# echo ${a%bash}

root:x:0:0:root:/root:/bin/

[root@desktop0 opt]# echo ${a#\*:}

x:0:0:root:/root:/bin/bash

[root@desktop0 opt]# echo ${a%:\*}

root:x:0:0:root:/root

[root@desktop0 opt]# echo ${a##\*:root:}

/root:/bin/bash

案例: 利用字符串删除,编写批量修改文件后缀名的脚本

[root@desktop0 opt]# touch a{1..10}.txt

[root@desktop0 opt]# vim /opt/test4.sh

#! /bin/bash

for i in `ls /opt/\*.txt`

do

mv $i ${i%.\*}.doc

done

[root@desktop0 opt]# chmod +x /opt/test4.sh

[root@desktop0 opt]# /opt/test4.sh

[root@desktop0 opt]# vim /opt/test5.sh

#! /bin/bash

for i in `ls /opt/\*.$1`

do

mv $i ${i%.\*}.$2

done

[root@desktop0 opt]# chmod +x /opt/test5.sh

[root@desktop0 opt]# /opt/test5.sh doc txt

**变量初始值处理**

通过${var:-word}判断变量是否存在，决定变量的初始值。

若变量var已存在且非Null，则返回$var的值；否则返回字串“word”，原变量var的值不受影响。

[root@desktop0 opt]# x=11

[root@desktop0 opt]# echo $x

11

[root@desktop0 opt]# echo ${x:-123456} #因为x已存在,输出变量x的值

11

[root@desktop0 opt]# echo ${y:-123456} #因为y不存在,输出123456

123456

案例: 使用定义初值功能编写脚本,当用户没有输入新用户密码时,用123456作为密码

[root@desktop0 opt]# vim /opt/test6.sh

#! /bin/bash

read -p "请输用户名:" n

useradd $n

read -p "请输入密码:" p

echo ${p:-123456} | passwd --stdin $n

[root@desktop0 opt]# chmod +x /opt/test6.sh

[root@desktop0 opt]# /opt/test6.sh

请输用户名:haha

请输入密码:

更改用户 haha 的密码 。

passwd：所有的身份验证令牌已经成功更新。

案例: 让用户输入一个整数,可以计算出1到此数相加的结果,如果用户没有输入数字,则显示1

[root@desktop0 opt]# vim /opt/test7.sh

#! /bin/bash

read -p "请输入一个整数:" n

for i in `seq ${n:-1}`

do

let sum+=i

done

echo $sum

[root@desktop0 opt]# chmod +x /opt/test7.sh

[root@desktop0 opt]# /opt/test7.sh

请输入一个整数:100

5050

[root@desktop0 opt]# /opt/test7.sh

请输入一个整数:

1

**expect预期交互**

expect可以为交互式过程（比如FTP、SSH等登录过程）自动输送预先准备的文本或指令，而无需人工干预。触发的依据是预期会出现的特征提示文本

[root@desktop0 opt]# yum -y install expect

[root@desktop0 opt]# vim /opt/test8.sh

#! /bin/bash

expect << EOF #固定格式-----记住

spawn ssh root@172.25.0.11

expect "password:" { send "redhat\r" }

expect "#" { send "touch /tmp.txt\r" }

expect "#" { send "exit\r" } #此条命令不执行

EOF #固定格式

echo

[root@desktop0 opt]# chmod +x /opt/test8.sh

[root@desktop0 opt]# /opt/test8.sh

spawn ssh root@172.25.0.11

root@172.25.0.11's password:

Last login: Mon Jan 14 15:55:43 2019 from desktop0.example.com

[root@server0 ~]# touch /tmp.txt

[root@server0 ~]#

[root@desktop0 opt]#

**注意事项：**

expect脚本的最后一行默认不执行

如果不希望ssh时出现yes/no的提示，远程时使用如下选项:

ssh -o StrictHostKeyChecking=no server0

[root@desktop0 opt]# ssh -o StrictHostKeyChecking=no root@172.25.0.11

root@172.25.0.11's password:

**shell数组**

数组: 作用与自定义变量一样,而且使用一个数组名能定义多个值,达到精简脚本的效果

shell对变量类型的管理比较松散,变量的值默认均视为文本;用在数学运算时,自动将其转换为整数

[root@desktop0 opt]# var1=123

[root@desktop0 opt]# var2=$var1+20

[root@desktop0 opt# echo $var2

123+20 #123作为文本字串

[root@desktop0 opt]# expr $var1 + 20

143 #123作为整数值

**定义 /赋值数组**

法1: 整体赋值 格式: 数组名=(值1 值2 .. .. 值n)

法2: 为单个元素赋值 格式: 数组名[下标]=值 **下标从0开始**

**输出数组元素**

获取单个数组元素 格式: ${数组名[下标]}

获取所有数组元素 ${数组名[@]}

获取数组元素个数 ${#数组名[@]}

获取连续的多个数组元素 ${数组名[@]:起始下标:元素个数}

[root@desktop0 opt]# w=(11 22) #定义数组,放入两个值

[root@desktop0 opt]# echo ${w[0]}

11

[root@desktop0 opt]# echo ${w[1]}

22

[root@desktop0 opt]# w[2]=33 #追加数组值,下标是2

[root@desktop0 opt]# echo ${w[2]}

33

[root@desktop0 opt]# echo ${w[@]} #显示数组w中所有的值

11 22 33

[root@desktop0 opt]# echo ${#w[@]} #显示数组w中拥有的值的个数

3

[root@desktop0 opt]# echo ${w[@]:1:1} #获取数组中某个下标的值

22

**正则表达式**

使用”一串符号”来描述有共同属性的数据

egrep过滤工具(等同于grep -E,表示允许使用扩展的正则表达式)

格式1: egrep [选项] ‘正则表达式’ 文件.. ..

格式2: 前置命令 | egrep [选项] ‘正则表达式’

常用命令选项

-i : 忽略字母大小写

-v : 条件取反

-c : 统计匹配的行数

-q : 静默、无任何输出,一般用于检测(看$?返回值,如果为0,说明有匹配,否则无匹配)

-n : 显示出匹配结果所在的行号

--color : 标红显示匹配字串

**基本正则列表**

|  |  |
| --- | --- |
| **正则符号** | **描述** |
| ^ | 匹配行首 |
| $ | 匹配行尾 |
| [] | 集合,匹配集合中的任意单个字符 |
| [^] | 对集合取反 |
| . | 匹配任意单个字符 |
| \* | 匹配前一个字符任意次数(\*不允许单独使用) |
| \{n,m\} | 匹配前一个字符n到m次 |
| \{n\} | 匹配前一个字符n次 |
| \{n,\} | 匹配前一个n次以上 |
| \(\) | 保留 |

**扩展正则列表**

|  |  |
| --- | --- |
| **正则符号** | **描述** |
| + | 最少匹配一次 |
| ? | 最多匹配一次 |
| {n,m}  {n}  {n,} | 匹配n到m次  匹配前一个字符n次  匹配前一个n次以上 |
| () | 组合为整体,保留 |
| | | 或者 |
| \b | 单词边界 |

**行首尾及单字匹配**



**未定匹配次数**



**{}限定次数**

**限定表达式的匹配次数 {n} {n,m} {n,}**



**[]范围内单字匹配: 匹配指定字符集合内的任何一个字符; []内加^可取反**

|  |  |
| --- | --- |
| **示例** | **说明** |
| [alc45\_?] | 匹配a、l、c、4、5、\_、? |
| [a-z] | 匹配任意小写字母 |
| [A-Z] | 匹配任意大写字母 |
| [0-9] | 匹配任意数字 |
| [a-Z0-9] | 匹配任意字母或数字 |
| [^A-Z] | 匹配包括非大写字母的行 |
| ^[^a-z] | 匹配不以小写字母开头的行 |

**整体及边界匹配**



\ : 转义符号,可以为一些普通字符赋予特殊含义,或者将一些特殊字符变为普通字符

#########################################

**II-shell-Day05**

**sed命令解析**

sed : Stream EDitor 流式编辑器

1. 非交互,基于模式匹配过滤及修改文本
2. 逐行处理,并将结果输出到屏幕
3. 可实现对文本的输出,删除,替换,复制,剪切,导入,导出等各种操作

格式1：前置命令 | sed [选项] '条件指令'  **‘ ’记得加**

格式2：sed [选项] '条件指令' 文件.. ..

相关说明如下：

[定址符]处理动作

[地址1,[地址2]]

条件可以是行号或者/正则/

没有条件时，默认为所有条件

指令可以是增、删、改、查等指令

默认sed会将所有输出的内容都打印出来，可以使用-n屏蔽默认输出

选项中可以使用-r选项，让sed支持扩展正则

sed命令的常用选项：

-n（屏蔽**默认输出**，默认sed会输出读取文档的全部内容）

-r（让sed支持扩展正则）,若与其他选项一起使用,应作为首个选项

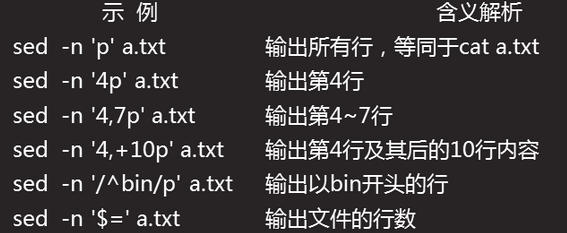
-i（sed**直接修改源文件**，默认sed只是通过内存临时修改文件，源文件无影响）

**常用动作指令**



替换操作的分隔”/”可改用其他字符,如#、&等,便于修改文件路径

**输出文本**

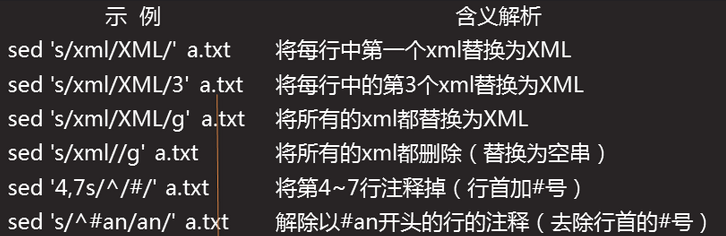


**删除文本**



此例中只作输出,不更改源文件(若需要更改,应添加选项-i)

**替换文本**



此例中只作输出,不更改源文件(若需要更改,应添加选项-i)

**p : 输出指定内容到屏幕**

[root@desktop0 opt]# sed -n '1p' /etc/passwd #显示文本中的第一行

root:x:0:0:root:/root:/bin/bash

[root@desktop0 opt]# sed -n '2p' /etc/passwd #显示文本中的第二行

bin:x:1:1:bin:/bin:/sbin/nologin

[root@desktop0 opt]# sed -n '1,3p' /etc/passwd #显示文本中的第一行到第三行

root:x:0:0:root:/root:/bin/bash

bin:x:1:1:bin:/bin:/sbin/nologin

daemon:x:2:2:daemon:/sbin:/sbin/nologin

[root@desktop0 opt]# sed -n '1p;3p' /etc/passwd #显示文本中的第一行和第三行

root:x:0:0:root:/root:/bin/bash

daemon:x:2:2:daemon:/sbin:/sbin/nologin

[root@desktop0 opt]# cat /opt/ab

aaa

bbb

ccc

ddd

eee

fff

[root@desktop0 opt]# sed -n '2,+2p' ab #显示文件中第二行以及后面的2行

bbb

ccc

ddd

[root@desktop0 opt]# sed -n '1~2p' ab #显示文件中第一行,第三行,第五行......

aaa

ccc

eee

[root@desktop0 opt]# sed -n '$=' ab #输出文件的行数

6

[root@desktop0 opt]# sed -n '/e/p' ab #正则表达式必须加’/ /’

eee

[root@desktop0 opt]# sed -n '$p' ab #输出最后一行

fff

[root@desktop0 opt]# cat /opt/xyz

aaaaaathe

thebbbbbaaaaa

aaa the bbb

ddddcccccaaa

a

[root@desktop0 opt]# sed -n '/the|bbb/p' xyz #显示空值

[root@desktop0 opt]# sed -rn '/the|bbb/p' xyz #扩展正则需要加’-r’

aaaaaathe

thebbbbbaaaaa

aaa the bbb

**d : 删除指定的内容**

[root@desktop0 opt]# sed '1d' ab #删除文本中的第一行

bbb

ccc

ddd

eee

fff

[root@desktop0 opt]# sed '1,3d' ab #删除文本中的一行到第三行

ddd

eee

fff

[root@desktop0 opt]# sed '1d;3d' ab #删除文本中的第一行和第三行

bbb

ddd

eee

fff

[root@desktop0 opt]# sed '1,+3d' ab #删除第一行和后面的三行

eee

fff

[root@desktop0 opt]# sed '1~2d' ab #删除第一行,第三行,第五行 .. ..

bbb

ddd

fff

[root@desktop0 opt]# sed '/a/!d' ab

aaa

[root@desktop0 opt]# sed '/[^a]/d' ab

aaa

[root@desktop0 opt]# sed '$d' ab #删除最后一行

aaa

bbb

ccc

ddd

eee

[root@desktop0 opt]# sed '/^$/d;/^#/d' xyz #删除空行和注释

**s : 替换**

格式 : s/旧内容/新内容/选项

替换操作的分隔“/”可改用其他字符，如#、&等，便于修改文件路径

[root@desktop0 opt]# sed 's/the/xyz/' xyz

aaaaaaxyz

xyzbbbbbaaaaa

aaa xyz bbb

ddddcccccaaa

a

[root@desktop0 opt]# sed '1s/the/xyz/' xyz

aaaaaaxyz

thebbbbbaaaaa

aaa the bbb

ddddcccccaaa

a

[root@desktop0 opt]# sed -n '1s/the/xyz/p' xyz

aaaaaaxyz

[root@desktop0 opt]# sed -n 's/^the/xyz/p' xyz

xyzbbbbbaaaaa

[root@desktop0 opt]# sed 's/a/x/g' xyz #将文档所有行的a替换成x

xxxxxxthe

thebbbbbxxxxx

xxx the bbb

ddddcccccxxx

x

[root@desktop0 opt]# sed 's/a/x/' xyz #将文档中所有行的第一个a替换成x

xaaaaathe

thebbbbbxaaaa

xaa the bbb

ddddcccccxaa

x

[root@desktop0 opt]# sed '2s/a/x/' xyz #将文档中第二行的第一个a替换成x

aaaaaathe

thebbbbbxaaaa

aaa the bbb

ddddcccccaaa

a

[root@desktop0 opt]# sed 's/a/x/2' xyz #将文档中所有行的第二个a替换成x

axaaaathe

thebbbbbaxaaa

axa the bbb

ddddcccccaxa

a

[root@desktop0 opt]# sed 's/a/x/2;s/a/x/4' xyz #会把第五个a替换成x

axaaxathe

thebbbbbaxaax

axa the bbb

ddddcccccaxa

a

[root@desktop0 opt]# sed 's/\/bin\/bash/\/bin\/sh/' /etc/passwd #转义符操作

[root@desktop0 opt]# sed 's#/bin/bash#/bin/sh#' /etc/passwd

# 可用大键盘上的0-9上面的任意一个特殊符号代替’///’

[root@desktop0 opt]# cat /opt/nssw.txt

Hello the world

ni hao ma beijing

**1) 删除文件中每行的第二个、最后一个字符**

[root@desktop0 opt]# sed 's/.//2;s/.$//' /opt/nssw.txt #替换每行第二个字符为空,替换每行最后一个字符为空

Hllo the worl

n hao ma beijin

**2) 将文件中每行的第一个、倒数第一个字符互换**

[root@desktop0 opt]# sed -r 's/^(.)(.\*)(.)$/\3\2\1/' nssw.txt #先将每行的第一个字符复制为\1, 中间的字符复制为\2, 结尾的字符复制为\3 ,然后替换顺序粘贴

dello the worlH

gi hao ma beijinn

**3）删除文件中所有的数字**

因原文件内没有数字，行首也没有空格，这里稍作做一点处理，生成一个新测试文件：

[root@desktop0 opt]# sed 's/[0-9]//' nssw.txt #首先用[0-9]匹配所有数字,然后替换为空

**4) 删除所有数字、行首空格的操作如下：**

法1: [root@desktop0 opt]#sed -r 's/[0-9]//g;s/^( )+//' nssw2.txt

法2: [root@desktop0 opt]#sed -r 's/[0-9]//g;s/^ +//' nssw2.txt #使用扩展正则要加-r,+指前面的空格可以出现1次或多次

**5）为文件中每个大写字母添加括号**

使用“（）”可实现保留功能，所以可参考下列操作解决：

[root@desktop0 opt]# sed -r 's/([A-Z])/(\1)/g' nssw.txt

123(H)el(L)o the world

ni h(A)o ma beijing

ab789

[root@desktop0 opt]# sed -r 's/([A-Z])/[\1]/g' nssw.txt #用[A-Z]找到所有大写字母,然后加小括号,相当于复制,在替换为本身,并加括号

123[H]el[L]o the world

ni h[A]o ma beijing

ab789

[root@desktop0 opt]# cat /opt/x

abcde

zzzxy

[root@desktop0 opt]# sed -r 's/(.)(.)(.)(.)(.)/\5\4\3\2\1/' /opt/x

edcba

yxzzz

案例: 编写脚本anonftp.sh，实现以下功能：

1) 通过yum安装vsftpd软件包

2) 修改vsftpd服务配置，开启匿名上传

3) 调整/var/ftp/pub目录权限，允许写入

4) 启动vsftpd服务，并设置开机自运行

[root@desktop0 opt]# vim /opt/anonftp.sh

#! /bin/bash

yum -y install vsftpd &> /dev/null

sed -i 's/^#anon\_up/anon\_up/' /etc/vsftpd/vsftpd.conf

chmod o+w /var/ftp/pub

systemctl start vsftpd

systemctl enable vsftpd

systemctl stop firewalld

setenforce 0

[root@desktop0 opt]# chmod +x /opt/anonftp.sh

[root@desktop0 opt]# /opt/anonftp.sh #运行成功,真机上可上传文件

sed [选项] '条件指令' 文件..



sed工具的多行文本处理操作： 用法与p、d、s一样

i： 在指定的行之前插入文本

a：在指定的行之后追加文本

c：替换指定的行

[root@desktop0 opt]# sed ‘i xx’ a.txt #在每一行前面插入xx

[root@desktop0 opt]# sed ‘2a xx’ a.txt #在第二行后面,追加xx

[root@desktop0 opt]# sed ‘2i xx’ a.txt #在第二行前面插入xx

[root@desktop0 opt]# sed ‘2c xx’ a.txt #将第二行替换为xx

[root@desktop0 opt]# sed -i '$a 192.168.4.5 svr5.tarena.com svr5' /etc/hosts #在最后一行插入

修改后的文本有多行时,以换行符\n分隔; 或者使用\强制换行

[root@desktop0 opt]# cat m.txt

aa

bb

cc

[root@desktop0 opt]# sed '2cxx\nyy' m.txt

aa

xx

yy

cc

[root@desktop0 opt]# sed '2cxx\

> yy' m.txt

aa

xx

yy

cc

**文件导入导出操作**

基本动作

r 动作应结合-i选项才会存入,否则只输出

w 动作以覆盖的方式另存为新文件



[root@desktop0 opt]# cat a.txt

abc

xyz

123

456

[root@desktop0 opt]# cat b.txt

aa

[root@desktop0 opt]# sed '1r b.txt' a.txt #使用读入功能,将文件b.txt中的内容,读入到文件a.txt第一行的后面

abc

aa

xyz

123

456

[root@desktop0 opt]# sed 'w b.txt' a.txt

abc

xyz

123

456

[root@desktop0 opt]# cat b.txt

abc

xyz

123

456

[root@desktop0 opt]# sed '2w 2.txt' a.txt #将文件a.txt第二行覆盖保存到文件2.txt

abc

xyz

123

456

[root@desktop0 opt]# cat 2.txt

xyz

**sed复制剪切**

基本动作

H : 模式空间 ---[追加]---->保持空间

复制

h : 模式空间 ---[覆盖]---->保持空间

G : 保持空间 ---[追加]---->模式空间

粘贴

g : 保持空间 ---[覆盖]---->模式空间

[root@desktop0 opt]# cat a.txt

abc

xyz

123

456

[root@desktop0 opt]# sed '1H;$G' a.txt

abc

xyz

123

456

abc

[root@desktop0 opt]# sed '1h;$G' a.txt

abc

xyz

123

456

abc

[root@desktop0 opt]# sed '1h;2G' a.txt #使用覆盖的复制第一行到剪切板板中,然后追加的粘贴到第二行后

abc

xyz

abc

123

456

[root@desktop0 opt]# sed '1h;2g' a.txt #使用覆盖的复制第一行到剪切板板中,然后覆盖的粘贴到第二行后

abc

abc

123

456

案例: 要求编写脚本getupwd.sh，实现以下需求：

找到使用bash作登录Shell的本地用户

列出这些用户的shadow密码记录

按每行“用户名 --> 密码记录”保存到getupwd.log

[root@desktop0 opt]# vim /opt/getupwd.sh

#! /bin/bash

A=$(sed -n '/bash$/s/:.\*//p' /etc/passwd) #提取符合条件的账号记录,并只提取账户名

for i in $A #循环赋值账号给变量i

do

pass1=$(grep $i /etc/shadow) #按照刚才找到的账户在shadow文件中继续查找对应的人

pass2=${pass1#\*:} #将含有账户密码等信息的行掐头

pass3=${pass2%%:\*} #将含有密码等信息的行去尾,得到纯粹的密码

echo "$i--->$pass3" > /opt/a.txt #按照 用户名--->密码的方式输出

done

[root@desktop0 opt]# chmod +x /opt/getupwd.sh

[root@desktop0 opt]# /opt/getupwd.sh

[root@desktop0 opt]# cat /opt/a.txt

**练习题:**

题目1：显示包含San的行  
[root@room9pc01 opt]# sed -n '/San/p' data.txt

[root@room9pc01 opt]# awk /San/ data.txt

题目2：显示以J开头的人名所在的行

[root@room9pc01 opt]# sed -n '/^J/p' data.txt

题目3：显示700结尾的行

[root@room9pc01 opt]# sed -n '/700$/p' data.txt

题目4：显示不包834的行

[root@room9pc01 opt]# sed -n '/834/!p' data.txt

题目5：显示电话号码的区号为834的行

[root@room9pc01 opt]# sed -n '/:834-/p' data.txt

题目6：显示这样的行:它包含一个大写字后跟四个小写字母,逗号,空格,和一个大写字母

[root@room9pc01 opt]# sed -rn '/[A-Z][a-z]{4}, [A-Z]/p' data.txt

[root@room9pc01 opt]# sed -n '/[A-Z][a-z]\{4\}, [A-Z]/p' data.txt

题目7：显示以K或k开头的行

[root@room9pc01 opt]# sed -n '/^K\|^k/p' data.txt

[root@room9pc01 opt]# sed -rn '/^K|^k/p' data.txt

[root@room9pc01 opt]# sed -n '/^[Kk]/p' data.txt

题目8：显示工资为六位数的行

[root@room9pc01 opt]# sed -n '/:......$/p' data.txt

[root@room9pc01 opt]# sed -rn '/:[0-9]{6}$/p' data.txt

[root@room9pc01 opt]# sed -n '/:[0-9]\{6\}$/p' data.txt

题目9：显示包括Lincoln或lincoln的行

[root@room9pc01 opt]# sed -rn '/Lincoln|lincoln/p' data.txt

[root@room9pc01 opt]# sed -n '/Lincoln\|lincoln/p' data.txt

[root@room9pc01 opt]# sed -n '/[Ll]incoln/p' data.txt

题目10：把Jon的名字改成Jonathan

[root@room9pc01 opt]# sed -n 's/Jon/Jonathan/p' data.txt

题目11：删除文件头3行内容

[root@room9pc01 opt]# sed '1,3d' data.txt

题目12：显示5-10行

[root@room9pc01 opt]# sed -n '5,10p' data.txt

题目13：把三个星\*\*\*添加到有Fred的行

[root@room9pc01 opt]# sed -rn '/Fred/s/(.)$/\1\*\*\*/p' data.txt

[root@room9pc01 opt]# sed -n /Fred/s/$/\*\*\*/p data.txt

题目14：删除空白行

[root@room9pc01 opt]# sed '/^$/d' data.txt

#########################################

**II-shell-Day06**

**awk命令解析**

awk工具概述:精确查找文本,不能修改文本

awk编程语言/数据处理引擎

1) 创造者: Aho、Weinberger、Kernighan

2) 基于模式匹配检查输入文本,逐行处理并输出

3) 通常用在shell脚本中,获取指定的数据

4) 单独使用时,可对文本数据做统计

格式1: 前置命令 | awk [选项] ‘[条件]{指令}’

格式2: awk [选项] ‘[条件]{指令}’ 文件 .. ..

**注意**:

print是最常用的编辑指令；若有多条编辑指令，可用分号分隔。

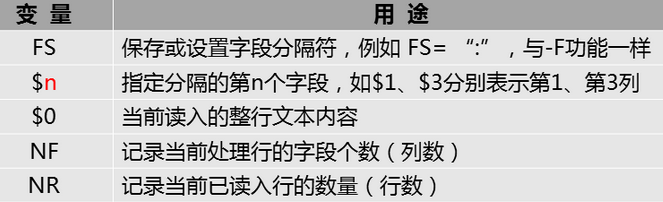
awk过滤数据时仅支持打印某一列，如第2列、第5列等。

处理文本时，若未指定分隔符，则默认将空格、制表符等作为分隔符。

常用命令选项:

-F : 指定分隔符,可省略(默认空格或tab位)

awk内置变量: 有特殊含义,可直接使用



$NF是最后一列

[root@desktop0 opt]# cat test.txt

hello the world

welcome to beijing

[root@desktop0 opt]# awk '{print$1}' test.txt #在文本中查找每行的第一列,并输出

hello

welcome

[root@desktop0 ~]# head -1 /etc/passwd | awk -F: '{print $1;print$7}'

root

/bin/bash

[root@desktop0 opt]# head -1 /etc/passwd | awk -F: '{print $1,$7}' #使用:作为分隔符,输出第一列和第7列

root /bin/bash

[root@desktop0 opt]# head -1 /etc/passwd | awk -F: '{print $1 $7}'

root/bin/bash

[root@desktop0 opt]# head -1 /etc/passwd | awk -F[:/] '{print $10}'

bash

[root@desktop0 opt]# df -h | awk '{print $4}' #使用空格作为默认的分隔符,输出第4列

[root@desktop0 opt]# head -1 /etc/passwd | awk -F: '{print NR}'

1

[root@desktop0 opt]# head -2 /etc/passwd | awk -F: '{print NR,$0}' #输出当前文本所有内容,并加上行号

1 root:x:0:0:root:/root:/bin/bash

2 bin:x:1:1:bin:/bin:/sbin/nologin

[root@desktop0 opt]# head -2 /etc/passwd | awk -F: '{print NF,$0}'

7 root:x:0:0:root:/root:/bin/bash

7 bin:x:1:1:bin:/bin:/sbin/nologin

[root@desktop0 opt]# head -2 /etc/passwd | awk -F: '{print NR,NF}' #输出当前文本每行的行号和列数

1 7

2 7

[root@desktop0 opt]# head -2 /etc/passwd | awk -F: '{print $1,"的解释器是",$7}'

root 的解释器是 /bin/bash

bin 的解释器是 /sbin/nologin

[root@desktop0 opt]# cat /etc/passwd | awk -F: '/^root/{print $1,"的 解释器是",$7}'

root 的解释器是 /bin/bash

[root@desktop0 opt]# df -h | awk '/\/$/{print "/分区的剩余容量是:",$4}'

/分区的剩余容量是: 7.1G

[root@desktop0 opt]# df -h | awk '/^\/dev\/vda1/{print "/分区的剩余容量是:",$4}'

/分区的剩余容量是: 7.1G

案例: 使用awk输出当前主机XXX网卡接收的网络数据是XXX

法1: [root@desktop0 ~]# ifconfig eth0 | grep 'RX p' -P | awk -F[\(\)] '{print "当前主机eth0网卡接收的数据是:"$2}'

当前主机eth0网卡接收的数据是:1.5 MiB

法2: [root@desktop0 ~]# ifconfig eth0 | grep RX\ p | awk -F[\(\)] '{print "当前主机eth0网卡接收的数据是:" $2}'

当前主机eth0网卡接收的数据是:1.5MiB

法3: [root@desktop0 opt]# ifconfig eth0 | awk -F[\(\)] '/RX p/{print "当前主机eth0网卡接收的数据是:"$2}'

当前主机eth0网卡接收的数据是:1.5 MiB

[root@desktop0 ~]# awk '/Failed/{print $11}' /var/log/secure #查找安全日志文件中,曾经访问本机失败的IP

172.25.0.11

**awk处理的时机**

awk会逐行处理文本，支持在处理第一行之前做一些准备工作，以及在处理完最后一行之后做一些总结性质的工作。在命令格式上分别体现如下：

awk [选项] '[条件]{指令}' 文件

**awk [选项] 'BEGIN{指令} {指令} END{指令}' 文件**

BEGIN{ } 行前处理，读取文件内容前执行，指令执行1次

{ } 逐行处理，读取文件过程中执行，指令执行n次

END{ } 行后处理，读取文件结束后执行，指令执行1次

[root@desktop0 ~]# awk 'BEGIN {A=24;print A\*2}'

48

[root@desktop0 ~]# awk -F: 'BEGIN{print x+1}/^root/{print $1}' /etc/passwd

1

root

[root@desktop0 ~]# awk 'BEGIN{x=0}/bash$/{x++}END{print x}' /etc/passwd #在逐行任务之前,先定义变量x=0,然后执行逐行任务,每找到一个以bash结尾的行,就把x++,最后结尾输出x

2

案例: 格式化输出/etc/passwd文件

要求: 格式化输出passwd文件内容时，要求第一行为列表标题，中间打印用户的名称、UID、家目录信息，最后一行提示一共已处理文本的总行数，

\t=tab键(水平制表符)

[root@desktop0 ~]# awk -F: 'BEGIN{print "User\tUID\tHome"}{print $1"\t"$3"\t"$6}END{print "Total",NR,"lines"}' /etc/passwd #按一定格式输出用户名、UID、家目录,注意\t代表tab键,属于常量,需要在双引号内

user uid home

root 0 /root

... ...

total 39 lines

**awk查找方式一: 使用正则设置条件**

[root@desktop0 ~]# awk '/^(root|adm)/' /etc/passwd

root:x:0:0:root:/root:/bin/bash

adm:x:3:4:adm:/var/adm:/sbin/nologin

[root@desktop0 ~]# awk '/^root|^adm/' /etc/passwd

root:x:0:0:root:/root:/bin/bash

adm:x:3:4:adm:/var/adm:/sbin/nologin

[root@desktop0 ~]# awk -F: '$1~/root/' /etc/passwd #查找第一列是root的行,并输出整行,输出整行时可以省略{print}

root:x:0:0:root:/root:/bin/bash

[root@desktop0 opt]# awk -F: '$7!~/nologin$/{print $1,$7}' /etc/passwd #查找第七列不是nologin的行,并输出第一列和第七列

~ : 匹配 !~ : 不匹配

**awk查找方式二: 使用数值/字符串比较设置条件**

比较符号：==(等于) !=（不等于） >（大于）

>=（大于等于） <（小于） <=（小于等于）

[root@desktop0 opt]# awk 'NR==1{print}' a.txt

[root@desktop0 opt]# awk 'NR==1;NR==2{print}' a.txt

[root@desktop0 opt]# awk -F: '$3<=10{print $1,$3}' /etc/passwd

[root@desktop0 ~]# awk -F: '$1=="root"' /etc/passwd

root:x:0:0:root:/root:/bin/bash

**awk查找方式三: 逻辑测试条件**

[root@desktop0 opt]# awk -F: '$3>10 && $3<20' /etc/passwd

[root@desktop0 opt]# awk -F: '$3>1000 || $3<10' /etc/passwd

**awk查找方式四: 数学运算**

[root@desktop0 opt]# awk 'BEGIN{x++;print x}'

1

[root@desktop0 opt]# awk 'BEGIN{x=8;print x+=2}'

10

[root@desktop0 opt]# awk 'BEGIN{x=8;x--;print x}'

7

[root@desktop0 opt]# awk 'BEGIN{print 2+3}'

5

[root@desktop0 opt]# awk 'BEGIN{print 2\*3}'

6

[root@desktop0 opt]# awk 'BEGIN{print 23%8}'

7

[root@desktop0 opt]# seq 200 | awk '$1%3==0' #查找200以内3的倍数

案例: 要求编写脚本，实现以下需求：

1) 找到使用bash作登录Shell的本地用户

2) 列出这些用户的shadow密码记录

[root@desktop0 ~]# vim /opt/test.sh

#! /bin/bash

u=`awk -F: '/bash$/{print $1}' /etc/passwd`

for i in $u

do

p=`grep "^$i:" /etc/shadow | awk -F: '{print $2}'`

echo "$i---->$p"

done

[root@desktop0 ~]# chmod +x /opt/test.sh

[root@desktop0 ~]# /opt/test.sh

**awk流程控制**

**awk过滤中的if分支结构**

**1) 单分支**

统计/etc/passwd文件中UID小于或等于1000的用户个数：

[root@desktop0 ~]# awk -F: '{if($3<=1000){x++}}END{print x}' /etc/passwd

38

统计/etc/passwd文件中登录Shell是“/bin/bash”的用户个数：

[root@desktop0 ~]# awk -F: '{if($7=="/bin/bash"){x++}}END{print x}' /etc/passwd

2

[root@desktop0 ~]# awk -F: '{if($7~/bash$/){x++}}END{print x}' /etc/passwd

2

1. **双分支**

案例: 在awk中使用if判断查找使用bash的用户数量,和不用bash的用户数量

[root@desktop0 ~]# awk -F: '{if(/bash$/){x++}else{y++}}END{print "用bash用户数量:"x"\n" "不用bash用户数量:"y}' /etc/passwd

用bash用户数量:2

不用bash用户数量:37

分别统计/etc/passwd文件中UID小于或等于1000、UID大于1000的用户个数：

[root@desktop0 ~]# awk -F: '{if($3<=1000){x++}else{y++}}END{print x,y}' /etc/passwd

39 1

**awk数组**

数组的语法格式:

数组是一个可以存储多个值的变量，具体使用的格式如下：

定义数组的格式：数组名[下标]=元素值

调用数组的格式：数组名[下标]

遍历数组的用法：for(变量 in 数组名){print 数组名[变量]}。

**注意**: awk数组的下标除了可以使用数字，也可以使用字符串，**字符串需要使用双引号**

[root@desktop0 opt]# awk 'BEGIN{a[0]=11;a[1]=88;print a[0],a[1]}'

11 88

[root@desktop0 opt]# awk 'BEGIN{a++;print a}'

1

[root@desktop0 opt]# awk 'BEGIN{a0++;print a0}'

1

[root@desktop0 opt]# awk 'BEGIN {a[0]++;print a[0]}'

1

[root@desktop0 opt]# awk 'BEGIN{a[0]=0;a[1]=11;a[2]=22;for(i in a){print i,a[i]}}'

0 0

1 11

2 22

[root@desktop0 opt]# awk 'BEGIN{a["hehe"]=1;print a["hehe"]}'

1

**awk扩展应用**

[root@desktop0 opt]# awk '{ip[$1]++}END{for(i in ip){print ip[i],i}}' /var/log/httpd/access\_log #查找web服务被访问的次数于对应的IP

[root@desktop0 opt]# awk '/Failed/{ip[$11]++}END{for(i in ip){print ip[i],i}}' /var/log/secure #查找安全日志中,访问本机失败的主机次数于对应的IP

1 172.25.0.11

利用sort对提取结果排序

-n: 按数字升序排列

-k: 针对指定的列进行排序

-r: 反向排序

练习:

grade.txt,内容如下:



**题目 1:打印整个文件**

$ awk '{print $0}' grade.txt

[root@room9pc01 opt]# awk '{print}' grade.txt

**题目 2:打印第一和第四个列**

$ awk '{print $1,$4}' grade.txt

[root@room9pc01 opt]# awk '{print $1,$4}' grade.txt

**题目 3:打印表头 Name Belt,然后显示文档第一列和第四列**

[root@room9pc01 opt]# awk 'BEGIN{print "Name\tBelt"}{print $1"\t"$4}' grade.txt

**题目 4:打印第四列包含 Brown 的行**

$ awk '$4~/Brown/ {print $0}' grade.txt

[root@room9pc01 opt]# awk 'NF==4;/Brown/{print}' grade.txt

**题目 5:打印第三列包含 48 的行**

$ awk '$3 ~/48/ {print $0}' grade.txt

[root@room9pc01 opt]# awk '$3==3;/48/' grade.txt

**题目 6:显示第三列是 48 的行**

$ awk '$3=="48" {print $0}' grade.txt

[root@room9pc01 opt]# awk '$3=="48"' grade.txt

**题目 7:显示不包含 Brown 的行**

$ awk '$0 !~ /Brown/' grade.txt

[root@room9pc01 opt]# awk '$0!~/Brown/' grade.txt

**题目 8:打印第四列不是 Brown-2 的行**

$ awk '$4 != "Brown-2" {print $0}' grade.txt

[root@room9pc01 opt]# awk '$4!~Brown-2' grade.txt

**题目 9:当第六列大于第七列是,显示:$6 大于 $7**

$ awk '$6 > $7 {print $6,"大于",$7}' grade.txt

[root@room9pc01 opt]# awk '$6>$7{print $6,"大于",$7}' grade.txt

**题目 10:当第一列的第四个字符是 a 时,显示该行**

$ awk '$1 ~/^...a/' grade.txt

[root@room9pc01 opt]# awk '/^...a/' grade.txt

**题目 11:显示包含 Yellow 或 Brown 的行**

$ awk '$0 ~/(Yellow|Brown)/' grade.txt

[root@room9pc01 opt]# awk '/Yellow|Brown/' grade.txt

**题目 12:显示文档第一列、第六列、第七列,如果第一列是 M.Tansley,则**

**先将其第六列的值减 1 后再输出。**

$ awk '{if($1=="M.Tansley") $6=$6-1; print $1,$6,$7}' grade.txt

[root@room9pc01 opt]# awk '{if($1=="M.Tansley"){$6-1}}{print $1,$6,$7}' grade.txt

#########################################

**II-shell-Day07**

案例1:

要求编写脚本实现一键部署Nginx软件（Web服务器）：

1) 一键源码安装Nginx软件

2) 脚本自动安装相关软件的依赖包

3) 脚本自动判断yum是否可用

[root@desktop0 opt]# vim /opt/test1.sh

#! /bin/bash

a=`yum repolist | awk '/^repolist/{print $2}' | sed 's/,//'`

if [ $a -eq 0 ];then

echo "yum不可用"

exit

fi

yum -y install gcc openssl-devel pcre-devel &> /dev/null

tar -xf /opt/nginx-1.12.2.tar.gz -C /opt/

cd /opt/nginx-1.12.2/

./configure

make

make install

[root@desktop0 opt]# chmod +x /opt/test1.sh

[root@desktop0 opt]# /opt/test1.sh

确认安装效果:

Nginx默认安装路径为/usr/local/nginx,该目录下会提供4个子目录，分别如下：

/usr/local/nginx/conf 配置文件目录

/usr/local/nginx/html 网站页面目录

/usr/local/nginx/logs Nginx日志目录

/usr/local/nginx/sbin 主程序目录

[root@desktop0 opt]# /usr/local/nginx/sbin/nginx                //启动服务

[root@desktop0 opt]# /usr/local/nginx/sbin/nginx    -s stop        //关闭服务

[root@desktop0 opt]# /usr/local/nginx/sbin/nginx    -V            //查看软件信息

nginx version: nginx/1.12.2

built by gcc 4.8.2 20140120 (Red Hat 4.8.2-16) (GCC)

configure arguments:

案例2:

要求编写Ngin启动脚本，要求如下：

1) 脚本支持start、stop、restart、status

2) 脚本支持报错提示

3) 脚本具有判断是否已经开启或关闭的功能

netstat命令可以查看系统中启动的端口信息，该命令常用选项如下：

-n:以数字格式显示端口号

-t:显示TCP连接的端口

-u:显示UDP连接的端口

-l:显示服务正在监听的端口信息，如httpd启动后，会一直监听80端口

-p:显示监听端口的服务名称是什么（也就是程序名称）

[root@desktop0 opt]# vim /opt/test2.sh

#! /bin/bash

case $1 in

start)

/usr/local/nginx/sbin/nginx;;

stop)

/usr/local/nginx/sbin/nginx -s stop;;

restart)

/usr/local/nginx/sbin/nginx -s stop

/usr/local/nginx/sbin/nginx;;

status)

netstat -ntulp | grep -q nginx

if [ $? -eq 0 ];then

echo "服务OK"

else

echo "服务未启动"

fi;;

\*)

echo "输入错误"

esac

[root@desktop0 opt]# chmod +x /opt/test2.sh

[root@desktop0 opt]# /opt/test2.sh

输入错误

[root@desktop0 opt]# /opt/test2.sh start

[root@desktop0 opt]# /opt/test2.sh stop

[root@desktop0 opt]# /opt/test2.sh status

服务未启动

[root@desktop0 opt]# /opt/test2.sh start

[root@desktop0 opt]# /opt/test2.sh status

服务OK

案例3:

要求编写脚本，实现计算机各个性能数据监控的功能，具体监控项目要求如下：

1) CPU负载

2) 网卡流量

3) 内存剩余容量

4) 磁盘剩余容量

5) 计算机账户数量

6) 当前登录账户数量

7) 计算机当前开启的进程数量

8) 本机已安装的软件包数量

[root@desktop0 opt]# vim /opt/test3.sh

#! /bin/bash

uptime | awk '{print "CPU 1分、5分、15分的平均负载均衡:",$8,$9,$10}'

ifconfig eth0 | awk -F[\(\)] '/RX p/{print "网卡eth0接收的数据流量是:",$2}'

ifconfig eth0 | awk -F[\(\)] '/TX p/{print "网卡eth0发送的数据流量是:",$2}'

free -h | awk '/^Mem:/{print "内存剩余容量是:",$4}'

df -h | awk '/^\/dev/{print "磁盘剩余容量:",$4}'

n=`cat /etc/passwd | wc -l`

echo "计算机账户数量是:$n"

m=`who | wc -l`

echo "当前登录的账户数量是:$m"

x=`ps aux | wc -l`

echo "计算机当前开启的进程数量是:$x"

y=`rpm -qa | wc -l`

echo "本机已安装的软件包数量是:$y"

[root@desktop0 opt]# chmod +x /opt/test3.sh

[root@desktop0 opt]# /opt/test3.sh

CPU 1分、5分、15分的平均负载均衡: 0.01, 0.03, 0.05

网卡eth0接收的数据流量是: 34.3 MiB

网卡eth0发送的数据流量是: 1.0 MiB

内存剩余容量是: 129M

磁盘剩余容量: 6.9G

计算机账户数量是:39

当前登录的账户数量是:4

计算机当前开启的进程数量是:130

本机已安装的软件包数量是:1224

案例4:

要求编写脚本，防止远程ssh暴力破解密码，具体监控项目要求如下：

1) 检测ssh登录日志，如果远程登陆账号名错误3次，则屏蔽远程主机的IP

2) 检测ssh登录日志，如果远程登陆密码错误3次，则屏蔽远程主机的IP

[root@desktop0 opt]# vim /opt/test4.sh

#! /bin/bash

awk '/Failed password/{print $11}' /var/log/secure | awk '{ip[$1]++}END{for(i in ip){print ip[i],i}}' | awk '$1>3{print $2}'

awk '/Invalid user/{print $10}' /var/log/secure | awk '{a[$1]++}END{for(x in a){print a[x],x}}' | awk '$1>3{print $2}'

[root@desktop0 opt]# chmod +x /opt/test4.sh

[root@desktop0 opt]# /opt/test4.sh

案例5:

要求编写脚本，实现带进程显示的复制脚本，具体要求如下：

1) 默认Linux的cp命令不具有进度显示

2) 我们需要自己编写脚本实现进度显示

3) 可以使用进度条的方式，或者显示百分比的方式

[root@desktop0 ~]# vim /opt/test5.sh

#!/bin/bash

jindu(){

while :

do

echo -ne '\033[43m \033[0m'

sleep 0.3

done

}

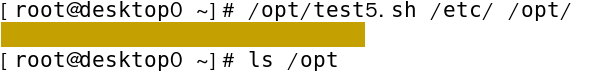
jindu &

cp -r $1 $2

echo

kill $!

[root@desktop0 ~]# chmod +x /opt/test5.sh



#########################################

[root@room9pc01 ~]# virt-manager #打开虚拟系统管理器

[root@room9pc01 ~]# virsh list #显示所有正在运行的虚拟机

[root@room9pc01 ~]# virsh shutdown rh7\_node58 #关闭虚拟机

域 rh7\_node58 被关闭

[root@room9pc01 ~]# virsh list --all #显示所有的虚拟机

Id 名称 状态

----------------------------------------------------

1 classroom running

2 server running

3 desktop running

- A 关闭

- B 关闭

- C 关闭

- D 关闭

- E 关闭

- F 关闭

- generic 关闭

- generic-2 关闭

- rh7\_node08 关闭

- win2008 关闭

[root@room9pc01 ~]# virsh start B #开启虚拟机B

域 B 已开始

[root@room9pc01 ~]# \virsh start C #开启虚拟机C

域 C 已开始

[root@room9pc01 ~]# virsh list --all

Id 名称 状态

----------------------------------------------------

1 classroom running

2 server running

3 desktop running

5 A running

6 B running

7 C running

- D 关闭

- E 关闭

- F 关闭

- generic 关闭

- generic-2 关闭

- rh7\_node08 关闭

- win2008 关闭

[root@room9pc01 ~]# virsh console rh7\_node58

(virsh console 虚拟机名)

连接到域 rh7\_node58

换码符为 ^] #直接回车

Red Hat Enterprise Linux Server 7.4 (Maipo)

Kernel 3.10.0-693.el7.x86\_64 on an x86\_64

mysql58 login: root #Ctrl + ] 退出

密码： #输入密码

Last login: Tue Feb 12 20:29:30 from 192.168.4.50

[root@mysql58 ~]#

#########################################

[root@server0 ~]# nmcli connection add type ethernet con-name eth1 ifname eth1 #添加网卡eth1

[root@server0 ~]# nmcli connection modify eth1 ipv4.method manual ipv4.addresses 172.25.0.12/24 connection.autoconnect yes

[root@server0 ~]# nmcli connection up eth1 #给网卡eth1配置IP地址

网卡配置文件路径: /etc/sysconfig/network-scripts/

#########################################

**补充:**

[root@room9pc01 ~]# for i in 10 11

> do

> ssh 172.25.0.$i yum -y install mariadb mariadb-server mariadb-devel php php-mysql

> done

[root@desktop0 ~]# rpm -q mariadb

mariadb-5.5.35-3.el7.x86\_64

[root@desktop0 ~]# rpm -q php

php-5.4.16-21.el7.x86\_64

[root@desktop0 ~]# rpm -q php-mysql

php-mysql-5.4.16-21.el7.x86\_64

[root@desktop0 ~]# rpm -q mariadb-server

mariadb-server-5.5.35-3.el7.x86\_64

[root@server0 ~]# ssh -X root@172.25.0.10 touch /opt/a.txt

[root@server0 ~]#

[root@desktop0 ~]# ls /opt/a.txt

/opt/a.txt

如何使Nginx不显示版本号?

修改配置文件/usr/local/nginx/conf/ndinx.conf, 在http{}中加入server\_tokens off; 保存并退出, 重新加载配置文件

如果是nginx+PHP,找到php-fpm配置文件(fastcgi.conf), 找到fastcgi\_param SERVER\_SOFTWARE nginx/$nginx\_version;

修改为:fastcgi\_param SERVER\_SOFTWARE nginx;

如何修改nginx的版本名字?

一般来说修改3个位置，一个是scr/core/nginx.h、另一个是src/http/ngx\_http\_header\_filter\_module.c、还有一个src/http/ngx\_http\_special\_response.c。

提示：一般修改都是在nginx编译之前修改，修改完了之后需要重新编译

[root@xnj9 nginx-1.12.2]# vim src/core/nginx.h

#define NGINX\_VERSION "1.4.7"

#define NGINX\_VER "nginx/" NGINX\_VERSION

修改为：(其实版本号也可以随意修改)

#define NGINX\_VERSION "1.4.7"

#define NGINX\_VER "jdws/" NGINX\_VERSION

其实修改上面一个大部分就会生效！！！

[root@xnj9 nginx-1.12.2]# vim src/http/ngx\_http\_header\_filter\_module.c

static char ngx\_http\_server\_string[] = "Server: nginx" CRLF;

修改为：

static char ngx\_http\_server\_string[] = "Server: jdws" CRLF;

[root@xnj9 nginx-1.12.2]# vim src/http/ngx\_http\_special\_response.c

static u\_char ngx\_http\_error\_tail[] =

"<hr><center>nginx</center>" CRLF

"</body>" CRLF

"</html>" CRLF

修改为：

static u\_char ngx\_http\_error\_tail[] =

"<hr><center>jdws</center>" CRLF

"</body>" CRLF

"</html>" CRLF

除了随意修改版本号之外，还可以隐藏版本号：

修改nginx.conf

在http{}加入

http{

server\_tokens off;

}

就可以隐藏版本号了。

**注意**：

关掉了server\_tokens之后，nginx.conf配置的东西会无效，回去寻找刚才我们第二步和第三步的配置

**ftp命令行下载文件**

[root@room9pc01 ~]# ftp 176.19.7.250

Connected to 176.19.7.250 (176.19.7.250).

220 (vsFTPd 3.0.2)

Name (176.19.7.250:root): anonymous

331 Please specify the password.

Password: #直接回车

230 Login successful.

Remote system type is UNIX.

Using binary mode to transfer files.

ftp> ls

ftp> get 训练营测试.docx #下载文件

local: 训练营测试.docx remote: 训练营测试.docx

227 Entering Passive Mode (176,19,7,250,99,155).

150 Opening BINARY mode data connection for 训练营测试.docx (17182 bytes).

226 Transfer complete.

17182 bytes received in 0.0015 secs (11469.96 Kbytes/sec)

ftp> cd RHCE #切换目录

250 Directory successfully changed.

ftp> exit #退出

[root@room9pc01 ~]# ls 训练营测试.docx

训练营测试.docx

**真机输入法故障修改**

[root@room9pc01 ~]# vim /usr/bin/wps

1 #!/bin/bash

2 export XMODIFIERS="@im=fcitx"

3 export QT\_IM\_MODULE="fcitx"

**添加网关**

[root@mysql54 ~]# systemctl stop NetworkManager

[root@mysql54 ~]# route add default gw 192.168.4.50 #添加网关

[root@mysql54 ~]# route del default gw 192.168.4.50 #删除网关

[root@mysql54 ~]# route -n #查看网关,立即生效

**终端分屏显示:**

[root@room9pc01 ~]# yum -y install screen

[root@room9pc01 ~]# screen #回车

Ctrl+a--->shift+s #上分屏

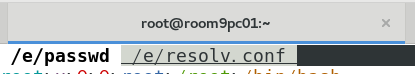
Ctrl+a--->shift+\ #左右分屏

Ctrl+a--->tab #换屏

Ctrl+a--->c #显示所在屏的终端,可以输入命令

**vim标签显示**

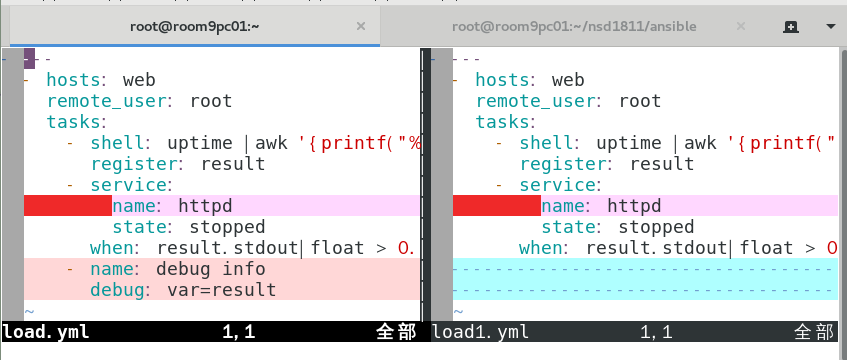
[root@room9pc01 ~]# vim -p /etc/passwd /etc/resolv.conf #g+t切换标签



[root@room9pc01 ~]# vim -O /etc/passwd /etc/resolv.conf #ctrl+w+w切换屏幕



[root@room9pc01 ~]# vimdiff load.yml load1.yml #Ctrl+w+w切换屏幕



**ubuntu系统apt-get和dpkg用法**

## 浏览器输入mirrors.ustc.edu.cn 查资料

[root@room9pc01 ~]# ssh sxk@192.168.4.55

sxk@192.168.4.55's password: #输入密码普通用户sxk的密码

sxk@ubuntu:~$ sudo su - 切换到root用户

[sudo] sxk 的密码：

root@ubuntu:~#

## **apt命令用法:**

packagename指代为软件包的名称。

apt-get update #在修改/etc/apt/sources.list或/etc/apt/preferences之后运行该命令。此外您需要定期运行这一命令以确保您的软件包列表是最新的。

apt-get install packagename #安装一个新软件包（参见下文的aptitude）

apt-get remove packagename #卸载一个已安装的软件包（保留配置文档）

apt-get remove --purge packagename #卸载一个已安装的软件包（删除配置文档）

apt-get autoremove packagename #删除包及其依赖的软件包

apt-get autoremove --purge packagname #删除包及其依赖的软件包+配置文件，比上面的要删除的彻底一点

dpkg --force-all --purge packagename #有些软件很难卸载，而且还阻止了别的软件的应用，就能够用这个，但是有点冒险。

apt-get autoclean #apt会把已装或已卸的软件都备份在硬盘上，所以假如需要空间的话，能够让这个命令来删除您已卸载掉的软件的备份。

apt-get clean #这个命令会把安装的软件的备份也删除，但是这样不会影响软件的使用。

apt-get upgrade #可以使用这条命令更新软件包，apt-get upgrade不仅可以从相同版本号的发布版中更新软件包，也可以从新版本号的发布版中更新软件包，尽管实现后一种更新的推荐命令为apt-get dist-upgrade。

在运行apt-get upgrade命令时加上-u选项很有用（即：apt-get -u upgrade)。这个选项让APT显示完整的可更新软件包列表。不加这个选项，你就只能盲目地更新。APT会下载每个软件包的最新更新版本，然后以合理的次序安装它们。注意在运行该命令前应先运行 apt-get update更新数据库，更新任何已安装的软件包。

apt-get dist-upgrade #将系统升级到新版本。

apt-cache search string #在软件包列表中搜索字符串。

dpkg -l package-name-pattern #列出任何和模式相匹配的软件包。假如您不知道软件包的全名，您能够使用“\*package-name-pattern\*”。

aptitude #周详查看已安装或可用的软件包。和apt-get类似，aptitude能够通过命令行方式调用，但仅限于某些命令——最常见的有安装和卸载命令。

由于aptitude比apt-get了解更多信息，能够说他更适合用来进行安装和卸载。

apt-cache showpkg pkgs #显示软件包信息。

apt-cache dumpavail #打印可用软件包列表。

apt-cache show pkgs #显示软件包记录，类似于dpkg –print-avail。

apt-cache pkgnames #打印软件包列表中任何软件包的名称。

dpkg -S file #这个文档属于哪个已安装软件包。

dpkg -L package #列出软件包中的任何文档。

dpkg -l #列出所有已安装的软件包。

apt-file search filename #查找包含特定文档的软件包（不一定是已安装的），这些文档的文档名中含有指定的字符串。

apt-file是个单独的软件包。您必须先使用apt-get install来安装他，然后运行apt-file update。假如apt-file search filename输出的内容太多，您能够尝试使用apt-file search filename | grep -w filename（只显示指定字符串作为完整的单词出现在其中的那些文档名）或类似方法，例如：apt-file search filename | grep /bin/（只显示位于诸如/bin或/usr/bin这些文件夹中的文档，假如您要查找的是某个特定的执行文档的话，这样做是有帮助的）。

apt-get autoclean #定期运行这个命令来清除那些已卸载的软件的.deb文档。通过这种方式，您能够释放大量的磁盘空间。假如您的需求十分迫切，能够使用apt-get clean以释放更多空间。这个命令会将已安装软件包裹的.deb文档一并删除。大多数情况下您不会再用到这些.debs文档，因此假如您为磁盘空间不足而感到焦头烂额，这个办法也许值得一试。

**dpkg的用法:**

1）安装软件

命令行：dpkg -i <.deb file name>

示例：dpkg -i avg71flm\_r28-1\_i386.deb

2）安装一个目录下面所有的软件包

命令行：dpkg -R

示例：dpkg -R /usr/local/src

3）释放软件包，但是不进行配置

命令行：dpkg –-unpack package\_file 如果和-R一起使用，参数可以是一个目录

示例：dpkg –-unpack avg71flm\_r28-1\_i386.deb

4）重新配置和释放软件包

命令行：dpkg –configure package\_file

如果和-a一起使用，将配置所有没有配置的软件包

示例：dpkg –configure avg71flm\_r28-1\_i386.deb

5）删除软件包（保留其配置信息）

命令行：dpkg -r

示例：dpkg -r avg71flm

6）替代软件包的信息

命令行：dpkg –update-avail <Packages-file>

7）合并软件包信息

dpkg –merge-avail <Packages-file>

8）从软件包里面读取软件的信息

命令行：dpkg -A package\_file

9）删除一个包（包括配置信息）

命令行：dpkg -P

10）丢失所有的Uninstall的软件包信息

命令行：dpkg –forget-old-unavail

11）删除软件包的Avaliable信息

命令行：dpkg –clear-avail

12）查找只有部分安装的软件包信息

命令行：dpkg -C

13）比较同一个包的不同版本之间的差别

命令行：dpkg –compare-versions ver1 op ver2

14）显示帮助信息

命令行：dpkg –help

15）显示dpkg的Licence

命令行：dpkg –licence (or) dpkg –license

16）显示dpkg的版本号

命令行：dpkg --version

17）建立一个deb文件

命令行：dpkg -b directory [filename]

18）显示一个Deb文件的目录

命令行：dpkg -c filename

19）显示一个Deb的说明

命令行：dpkg -I filename [control-file]

20）搜索Deb包

命令行：dpkg -l package-name-pattern

示例：dpkg -I vim

21）显示所有已经安装的Deb包，同时显示版本号以及简短说明

命令行：dpkg -l

22）报告指定包的状态信息

命令行：dpkg -s package-name

示例：dpkg -s ssh

23）显示一个包安装到系统里面的文件目录信息

命令行：dpkg -L package-Name

示例：dpkg -L apache2

24）搜索指定包里面的文件（模糊查询）

命令行：dpkg -S filename-search-pattern

25）显示包的具体信息

命令行：dpkg -p package-name

示例：dpkg -p cacti

[root@docker03 ~]# curl 220.181.57.216 -o /var/www/html/index.html #把首页文件变成百度首页

[root@docker03 ~]# curl www.baidu.com -o /var/www/html/index.html

[root@docker03 ~]# curl www.tmooc.cn -o /var/www/html/index.html

**RHEL6服务管理**

**服务**

**系统服务**

独立监听的、响应速度快、持续占用系统资源

**路径**

系统服务(脚本): /etc/init.d/\*

[root@Centos6 ~]# ls /etc/init.d #列出所有的系统服务(脚本名)

**系统服务控制**

基本方法:

格式1: service 服务名称 控制参数

格式2: /etc/init.d/服务名称 控制参数

主要控制参数

start:启动

stop:停止

status:查看服务的当前状态

restart:重新启动

reload:重新加载配置

[root@Centos6 ~]# /etc/init.d/httpd #用法提示

Usage: httpd {start|stop|restart|condrestart|try-restart|force-reload|reload|status|fullstatus|graceful|help|configtest}

[root@Centos6 ~]# /etc/init.d/httpd start #启动服务

[root@Centos6 ~]# ss -antulp | grep 80

[root@Centos6 ~]# service httpd status

httpd (pid 2394) 正在运行...

[root@Centos6 ~]# service httpd stop #停止服务

**设置服务的自启状态**

chkconfig 命令行工具

格式: chkconfig --list [服务名]

chkconfig [--level 级别列表] 服务名 on

chkconfig [--level 级别列表] 服务名 off

[root@Centos6 ~]# chkconfig --list #列出所有服务的自启状态

[root@Centos6 ~]# chkconfig iptables off

[root@Centos6 ~]# chkconfig iptables --list = chkconfig --list iptables

iptables 0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:关闭 6:关闭

[root@Centos6 ~]# chkconfig iptables on

[root@Centos6 ~]# chkconfig --list iptables

iptables 0:关闭 1:关闭 2:启用 3:启用 4:启用 5:启用 6:关闭

**RHEL7特性**

文件系统: 默认使用的文件系统为XFS, XFS是一个高性能的大文件系统,XFS支持在线调整大小, RHEL6默是ext4

**systemd**

init程序的作用

Linux系统和服务管理器

1. 是内核引导之后加载的第一个初始化进程(PID=1)

2) 负责掌控整个Linux的运行/服务资源组合

传统的 init 程序风格

1) system v: 顺序加载,RHEL6系列采用

systemd

一个更高效的系统&服务管理器

1. 开机服务并行启动,各系统服务间的精确依赖
2. 配置目录:/etc/systemd/system/
3. 服务目录:/lib/systemd/system/
4. 主要管理工具:systemctl

[root@room9pc01 ~]# ls -l /sbin/init

lrwxrwxrwx. 1 root root 22 4月 29 2018 /sbin/init -> ../lib/systemd/systemd

[root@room9pc01 ~]# systemctl -t service 列出活动的系统服务

[root@room9pc01 ~]# systemctl -t service --all #列出所有系统服务(包括不活动的)

**管理运行级别**

rhel6:

[root@Centos6 ~]# 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:5:initdefault:

[root@Centos6 ~]# init 5 #图形化模式

[root@Centos6 ~]# init 3 #字符界面模式

RHEL7:

[root@room9pc01 ~]# systemctl -t target 列出可用运行级别

[root@room9pc01 ~]# systemctl isolate multi-user.target #字符模式

[root@room9pc01 ~]# systemctl isolate graphical.target #图形模式

[root@room9pc01 ~]# runlevel #确认结果

3 5 #旧级别 当前级别

[root@room9pc01 ~]# systemctl get-default #查看默认级别

graphical.target

[root@room9pc01 ~]# systemctl set-default multi-user.target #设置默认级别

[root@room9pc01 ~]# systemctl set-default graphical.target

**RHEL6与RHEL7命令对比:**



