Day 01 开学典礼，云计算介绍，TCP/IP协议及其配置

云计算介绍

· 云中的计算机，提供云服务

服务器架构

1. 服务器：为其他计算机提供服务的高级主机

- 机架式

- 塔式

- 机柜式

- 刀片式

1. 典型服务模式

- Client/Server(C/S)架构：由服务器提供资源和功能，客户机使用资源和功能

TCP/IP协议及配置

1. TCP/IP协议：统一准则；
2. 主机通信三要素：

-IP地址 ip address

-子网掩码 subnet mask

-IP路由 ip router

1. IP地址：网络中标识节点的逻辑地址
2. 特点：

· 唯一标识

· 长

· 二进制

1. 组成（点分十进制）

-32个二进制

-表示为4个十进制数，用.隔开

1. 分类：

· A类:1-126 网 主 主 主

· B类:128-191 网 网 主 主

· C类:192-223 网 网 网 主

· D类:224-239 组播用

· E类:240-254 科研用

· ip 127.0.0.1代表本机

1. 给企业的私有IP划分有：

· 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

1. 子网掩码：标识IP地址的主机位和网络位，1表示网络位，0表示主机位

· A类:255.0.0.0

· B类:255.255.0.0

· C类:255.255.255.0

『192.168.1.1 255.255.255.0』= 192.168.1.1/24 #24个网络位

DHCP协议：自动配置ip地址

1. 网关：

· 从一个网络连接到另一个网络的“关口”（不同的网络之间的通信）

1. DNS域名解析服务器：将域名解析为IP地址
2. 在Window系统的cmd命令行界面中：

· 查看ip的命令：ipconfig 若要查看更多信息，则使用ipconfig /all

· 测试通信的命令：ping

Day 02 Linux系统简介 安装Rhel7系统 Rhel7基本操作

1. Linux系统简介
2. Unix诞生（Linux时间的顶点）：1970-1-1；
3. Linux的诞生：
4. Linux之父：Linus Torwalds

· 1991.10发布0.02版内核；

· 1994.3发布1.0版内核；

内核作用：调配所有的硬件

内核版本号：主版本.次版本.修订号

1. 一套公开发布的完整的Linux系统=Linux内核+应用程序
2. Linux系统的应用：

· 基于Linux系统的企业服务器

· 嵌入式系统

· 高性能大型运算

安装RHEL7系统

1. 硬盘的使用：

· 物理硬盘 ---> 分区规划 ---> 格式化 ---> 读/写

· -格式化：赋予空间文件系统，决定数据在空间中如何存储（排列规则）

· Windows文件系统：FAT，NTFS

· Linux文件系统： RHEL6 ext4

RHEL7 xfs

SWAP 交换空间（虚拟内存）

· 格式化的基本作用：

· 定义向磁盘介质上存储文档的方法和数据结构，以及读取文档的规则；

· 不同类型的文件系统，其存储/读取方式不一样；

· 格式化操作就是建立新的文件系统；

1. 鼠标回到真机：ctrl+alt
2. Linux管理员用户：root
3. 使用虚拟机软件

· 通过软件来模拟真实计算机的一类软件程序

· 一台物理机可运行多个虚拟机

· 每个虚拟机提供一个相对独立的操作系统环境

Linux目录结构

1. 最顶层为根目录(/)：所有数据都在此目录下（Linux系统的起点）
2. /dev：设备所在目录;
3. hd表示IDE设备;sd表示SCSI设备;vd表示虚拟机硬盘;
4. /dev/sda5

· a:磁盘号ls

· 5:分区号

命令行基本操作

1. 获取命令行界面
2. 虚拟控制台切换(ctrl+alt+Fx)

· tty1-tty2:图形桌面

· tty3-tty6:字符控制台

·右键“在终端中打开”

·应用程序 ---> 工具 ---> 终端

1. 伪字符终端

·提示符：

管理员:[用户名@主机名 当前目录]#

用户:[用户名@主机名 当前目录]$

~:用户的家(Home)目录

/root:管理员的家目录

/home:用户的家目录

1. 命令行的一般格式：

· 命令字 [选项]... [参数1] [参数2]...

· 命令

1. pwd - Print Working Directory：查看当前工作目录
2. cd - Change Directory：切换工作目录

· 格式：cd [目标路径]

· 路径：分为 绝对路径 和 相对路径

· 在相对路径中：

- . ：当前目录

- .. ：上一级目录

- cd .. 返回上一级

1. ls - List：列出当前所在目录内容

· 格式：ls [选项]... [目录或文件名]...

· 支持多个参数

· 选项： -l：显示目录内容的详细信息

1. cat：查看文本文件内容

· 选项： -n 显示同时标注行号

· 格式：cat /root/1.txt

· cat /proc/meminfo：列出内存信息

· 查看系统版本

· cat /etc/redhat-release

1. clear & Ctrl+L：清屏
2. uname -r：列出内核版本

[root@localhost ~]# uname -r

3.10.0-693.el7.x86\_64

1. cp - copy：拷贝

· 选项： -r 拷贝整个目录

1. mv - move：移动（剪切）

· 改文件名也使用mv

1. less：分屏浏览文本

· 查看大文件

· 全文查找 “/”+查找的内容

· 按q退出浏览模式

1. head -(数字n) 文件名 看头n行（默认10）
2. tail -(数字n) 文件名 看尾n行（默认10）
3. man：查看命令帮助信息

· 格式：man 命令

1. lscpu：列出cpu信息
2. cat /proc/meminfo：查看内存信息
3. hostname：临时查看/修改主机名

· 查看： [root@xxd ~]# hostname

xxd

· 修改： [root@xxd ~]# hostname localhost

1. ifconfig：列出已激活的网卡连接信息

· ifconfig eth0 192.168.1.1 临时配置ip

1. ping：测试网络连通性

· Ctrl+C：结束正在运行的命令

1. poweroff：关机
2. reboot：重启
3. mkdir - Make Directory：创建目录
4. touch

· 用途：创建空文件

· 格式：touch 文件名

1. grep：

· 用途：输出包含指定字符串的行

· 格式：grep [选项] '查找内容' 目标文件

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

或者[root@localhost 桌面]# grep 'root' /etc/passwd

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

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

1. Linux系统中：

文本色：文件

蓝色： 目录

1. /etc/redhat-release：储存本机系统的具体版本信息
2. 技巧：<Tab> 补全键：补全路径和命令

连续两下Tab键：列出可能的所有命令;

Day 03 命令行基础、目录和文件管理、教学环境介绍

1. 命令行基础
2. 如何编写命令行

i.什么是命令

- 用来实现某一类功能的指令或程序

- 命令的执行依赖于解释器（例如：/bin/bash）

1.绿色：可以执行的程序

青色：快捷方式

2. 提示“bash:firefox：未找到命令...”的原因

1) 命令有误

2) 该命令没有安装

3. poweroff ==> 解释器 ==> 内核 ==> 硬件

4. 解释器：Shell;

Linux下默认的解释器程序：/bin/bash

ii. 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

iii. 命令行的一般格式

·命令字 [选项]... [参数1][参数2]...

· 选项：调控命令的方式

-短选项：-l、-A、-c、-d...

-多个短选项 --> 复合选项：-lh、-lA...

-长选项：--help...

iv. 快速编辑技巧

·快捷键：

- ctrl + l ：清空整个屏幕

- ctrl + u ：清空至行首

- ctrl + w ：往回删除一个单词

- ctrl + c ：废弃当前编辑的命令行

- Esc + . ：粘贴上一个命令的参数

b)mount挂载操作

· 挂载/装载：

- 将光盘/U盘/分区/网络存储等设备装到某个Linux目录

- 各种命令工具通过访问Linux目录来操作这些设备

· 访问光盘的内容：

Windows：

光盘 => 光驱设备 => CD驱动器(访问点)

Linux：

光盘 => 光驱设备 => 访问点(手动配置)

挂载点 = 访问点 <==> 目录

1. 图形界面下将光盘放入光驱设备

2. 查看光驱设备

[root@localhost ~]# ls /dev/cdrom

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

lrwxrwxrwx. 1 root root 3 2月 1 10:34 /dev/cdrom -> sr0

3. 访问光驱设备，可以提供设备的访问点

· mount命令：挂载

[root@localhost ~]# mkdir /dvd

[root@localhost ~]# ls /dvd

[root@localhost ~]# ls /dev/cdrom

/dev/cdrom

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

mount: /dev/sr0 写保护，将以只读方式挂载

[root@localhost ~]# ls /dvd

（光盘文件）

[root@localhost ~]# ls /dvd/Packages/

（包文件爱你哟~）

4. 卸载已挂载的设备或分区

· umount命令：卸载

· 卸载和重装示例：

[root@localhost ~]# umount /dvd/

[root@localhost ~]# ls /dvd/

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

mount: /dev/sr0 写保护，将以只读方式挂载

[root@localhost ~]# ls /mnt/

（光盘文件）

· 在Linux中访问设备资源内容，必须通过目录作为访问点进行访问

5. 注意事项：

· 进行挂载时，挂载点目录必须要存在

· 进行挂载时，挂载点目录不要是根目录以下已经存在的目录（建议）

· 卸载时，确认当前没有任何人在挂载点中，否则出现“目标忙”提示

· 同一设备可以在两个目录上同时挂载

二、目录和文件管理

a)查看及切换目录

1.使用pwd、cd

- cd ：

· 用途：切换工作目录

· 格式：cd [目标文件夹位置]

· /home：存放所有普通用户的家目录

· ~user表示user的家目录

- useradd ：创建新用户

2.ls - List

- 格式：ls [选项]... [目录或文件名]

· 常用命令选项

- -l ：以长格式显示

- -A ：包括以.开头的隐藏文档

- -d ：显示目录本身的属性

- -h ：提供易读的容量(K、M等)

b)新建文档

1.命令的别名：简化复杂命令的执行

· alias 别名 = '真正执行的命令'

· unalias 删除别名

2.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

- {a,min,xy}：多组不同的字符串，全匹配

· 请利用通配符显示/dev/目录下tty20至tty30?

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

3.创建目录

· mkdir - Make Directory

- 格式：mkdir [-p] [/路径/]目录名...

mkdir -p ：创建多层的目录，当父目录没有的时候创建父目录

ls -R 递归显示：目录本身的内容 所有子目录的内容 依次展开

4. vim文本编辑器

- 格式：vim [[/目录/]文件名]

- 若目标文件不存在，则新建空文件并编辑

- 若目标文件不存在，则打开此文件并编辑

· 命令模式下：

- 按i进入输入模式编辑，按Esc回到命令模式

- 按':'进入末行模式，按Esc回到命令模式

· 输入 :wq 保存并退出

· 输入 :q! 不保存强制退出

c)复制/删除/移动

1. rm —— Remove

- 格式：rm [选项]... 文件或目录...

· 常用命令选项

-r：递归删除（含目录）

-f：强制删除

· rm -rf 爆破警告

2.mv —— 移动/改名

- 格式： mv [选项]... 原文件... 目标路径

- 功能：

· 移动并改名

· 重命名：路径不变的移动

3.cp —— Copy

- 格式： cp [选项]... 原文件... 目标路径

· 常用命令选项

-r： 递归，复制目录时必须有此选项

-p：保持源文件的权限、修改时间等属性不变

· 当拷贝出现两个以上参数时，永远将最后一个参数作为目标，其他作为源

· cp支持通配符

· cp与.连用

· cp支持重命名

三、教学环境：

a)每个学员机上有三台虚拟机：

- server --- 服务器

- desktop --- 客户机

- classroom --- 提供网关/DNS/软件素材等资源

首先开启classroom，然后再看其server和desktop

还原指令：物理机：vim /user/local/bin/rht-vmctl

然后rht-vmctl reset classroom

rht-vmctl reset server

rht-vmctl reset desktop

b) 利用别名方便每一次还原三台虚拟机

#每次开机root都会读取的文件/root/.bashrc

alias c='rht-vmctl reset classroom'

alias d='rht-vmctl reset desktop'

alias s='rht-vmctl reset server'

开启一个新的终端验证

Day 04 软件包管理 配置网络 文本/文件查找

一、远程管理：

a)Linux远程管理：ssh 对方用户名@服务器IP地址

server虚拟机 ：IP地址 172.25.0.11 主机名：server0.example.com

desktop虚拟机 ：IP地址 172.25.0.10 主机名：desktop0.example.com

[root@room9pc01 ~]# ssh -X root@172.25.0.11

-X（大写）：在远程管理时，可以在本机运行对端的图形程序

ctrl + shift + t ：开启一个新的终端

b) RPM软件包的管理

· RPM Package Manager，RPM包管理器

- rpm -q 软件名... #检测是否有安装程序

- rpm -ivh 软件名-版本信息.rpm... #安装软件包

- rpm -e 软件名... #卸载软件包

[root@room9pc01 bin]# rpm -q vsftpd

1.从光盘内容安装到系统中，首先关闭虚拟机，添加光驱设备

2.查看是否具备光驱设备

[root@server0 ~]# ls /dev/cdrom

/dev/cdrom

3.挂载光驱设备

[root@server0 ~]# mkdir /dvd

[root@server0 ~]# mount /dev/cdrom /dvd/

mount: /dev/sr0 写保护，将以只读方式挂载

[root@server0 ~]# ls /dvd/

[root@server0 ~]# ls /dvd/Packages/vsftpd-3.0.2-22.el7.x86\_64.rpm

/dvd/Packages/vsftpd-3.0.2-22.el7.x86\_64.rpm

4.安装包

4.1.一般安装

[root@server0 ~]# rpm -q vsftpd

未安装软件包 vsftpd

[root@server0 ~]# rpm -ivh /dvd/Packages/vsftpd-3.0.2-22.el7.x86\_64.rpm

警告：/dvd/Packages/vsftpd-3.0.2-22.el7.x86\_64.rpm: 头V3 RSA/SHA256 Signature, 密钥 ID fd431d51: NOKEY

准备中... ################################# [100%]

正在升级/安装...

1:vsftpd-3.0.2-22.el7 ################################# [100%]

[root@server0 ~]# rpm -q vsftpd

vsftpd-3.0.2-22.el7.x86\_64

· 警告原因：系统不能识别红帽认证信息

· 导入红帽签名信息：

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

· 导入后重装：

[root@server0 ~]# rpm -ivh /dvd/Packages/vsftpd-3.0.2-22.el7.x86\_64.rpm

准备中... ################################# [100%]

正在升级/安装...

1:vsftpd-3.0.2-22.el7 ################################# [100%]

5.卸载包

[root@server0 ~]# rpm -e vsftpd

[root@server0 ~]# rpm -q vsftpd

未安装软件包 vsftpd

c) Yum软件包仓库

· 自动解决依赖关系

· Yellowdog Update Manager，黄狗升级管理器

- yum repolist 列仓库

- yum list [软件名]... 列软件

- yum clean all 清缓存

- yum [-y] install 软件名... 安装软件

- yum [-y] remove 软件名... 卸载软件

· Yum服务端：为客户端自动解决依赖关系，安装软件包 classroom.example.com

搭建完Web服务，共享了光盘所有的内容

验证：http://classroom.example.com

1. 指定yum软件源：

- 服务端（软件仓库）

集中提供软件安装包，并提供依赖支持

- 客户端（yum命令及配置）

· yum相关文件位置

- /etc/yum.conf

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

[root@server0 ~]# rm -rf /etc/yum.repos.d/\*

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

#用vim编辑器修改repo文件信息

[rhel7] #仓库标识

name=rhel 7.0 #仓库描述信息

baseurl=http://classroom.example.com/content/rhel7.0/x86\_64/dvd/ #仓库路径

enabled=1 #本文件是否生效，1为Y，0为N

gpgcheck=0 #是否检测红帽签名，1为Y，0为N

[root@server0 ~]# yum repolist #列出仓库信息，检测是否能发现Yum服务端

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

[root@server0 ~]# yum -y install sssd #安装sssd软件

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

· 真机搭建Yum仓库：

· 真机 服务端：将CentOS 7.4光盘内容显示在系统中：

[root@room9pc01 ~]# mkdir /dvd

[root@room9pc01 ~]# mount /iso/rhel-server-7.4-x86\_64-dvd.iso /dvd

mount: /dev/loop1 写保护，将以只读方式挂载

[root@room9pc01 ~]# ls /dvd/

多媒体：包组

yum groups list [hidden] #查看列表

yum groups install 多媒体 #安装

二、文本查找

a)grep过滤操作

· 根据字符串提取文本行

- grep [选项] '指定字符串'

· 选项：

· -v ：取反匹配

· -i ：忽略大小写

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

[root@server0 ~]# grep -i 'root' /etc/passwd #忽略大小写

[root@server0 ~]# grep -v 'root' /etc/passwd #不包含root的行

· 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

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

[root@server0 ~]# grep '^root' /etc/passwd

[root@server0 ~]# grep 'root$' /etc/passwd

[root@server0 ~]# grep 'bash$' /etc/passwd

b) 重定向输出

· 将命令行的正常输出保存到文件

- 覆盖式：命令行 > 文件

- 追加式：命令行 >> 文件

c) echo 'xxx' 将字符串输出到终端

三、wget 命令行下载工具

a)[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

四、网络参数的配置

1.设置永久主机名：修改配置文件 /etc/hostname

2.配置永久ip地址，子网掩码，网关地址

[root@Gay ~]# cat /etc/sysconfig/network-scripts/ifcfg-eth0

3.nmcli连接管理

a) 查看识别的网卡的名字

[root@Gay ~]# nmcli connection show

b) 配置ip地址 子网掩码 网关地址

[root@Gay ~]# nmcli connection modify 'System eth0'

连接网络 修改 修改网卡的名称

ipv4.method manual/auto

修改ipv4的方法 手动/自动 模式选择

ipv4.addresses '172.25.0.100/24 172.25.0.254'

配置ipv4的地址 ip + 掩码 + 网关

connection.autoconnect yes

每次开机自动启用

c) 激活配置：

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

4.配置永久DNS服务器地址：将网站域名解析为对应的IP地址

a)修改配置文件：/etc/resolv.conf

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

[root@Gay ~]# cat /etc/resolv.conf

nameserver 172.25.254.254

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

Server: 172.25.254.254

Address: 172.25.254.254#53

Name: server0.example.com

Address: 172.25.0.11

四、find查找文档位置

1.根据预设的条件递归查找对应的文件

- find [目录] [条件1] [-a|-o] [条件2]...

- 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"

-size +|-文件大小(k,M,G)

-user 用户名

-maxdepth n 查找深度

-mtime +10 十天之前

-mtime -10 最近十天之内

[root@Gay ~]# ls /boot/grub/menu.lst

[root@Gay ~]# ll /boot/grub/menu.lst f

[root@Gay ~]# find /boot/ -type d

[root@Gay ~]# find /etc/ -name 'passwd'

[root@Gay ~]# find /etc/ -name '\*.conf'

[root@Gay ~]# find /etc/ -name '\*tab'

[root@Gay ~]# touch nsd01.txt nsd02.txt

[root@Gay ~]# mkdir nsd1801

[root@Gay ~]# find /root/ -name 'nsd\*'

/root/nsd01.txt

/root/nsd02.txt

/root/nsd1801

[root@Gay ~]# find /root/ -name 'nsd\*' -type d

/root/nsd1801

[root@Gay ~]# find /root/ -name 'nsd\*' -type f

/root/nsd01.txt

/root/nsd02.txt

2.find结果处理

- 使用find命令的 -exec操作

- find.. .. -exec处理命令{} \;

- 优势：以{}代表所有结果，逐个处理，遇 \;结束

[root@Gay ~]# find /boot/ -size +10M

/boot/initramfs-0-rescue-946cb0e817ea4adb916183df8c4fc817.img

/boot/initramfs-3.10.0-123.el7.x86\_64.img

/boot/initramfs-3.10.0-123.1.2.el7.x86\_64.img

[root@Gay ~]# find /boot/ -size +10M -exec cp {} /opt/ \;

[root@Gay ~]# ls /opt/

initramfs-0-rescue-946cb0e817ea4adb916183df8c4fc817.img

initramfs-3.10.0-123.1.2.el7.x86\_64.img

initramfs-3.10.0-123.el7.x86\_64.img

rh

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

/boot/vmlinuz-3.10.0-123.el7.x86\_64

/boot/vmlinuz-0-rescue-946cb0e817ea4adb916183df8c4fc817

/boot/vmlinuz-3.10.0-123.1.2.el7.x86\_64

[root@Gay ~]# find /boot/ -name "vm\*" -exec cp {} /opt/ \;

[root@Gay ~]# ls /opt/

initramfs-0-rescue-946cb0e817ea4adb916183df8c4fc817.img

initramfs-3.10.0-123.1.2.el7.x86\_64.img

initramfs-3.10.0-123.el7.x86\_64.img

rh

vmlinuz-0-rescue-946cb0e817ea4adb916183df8c4fc817

vmlinuz-3.10.0-123.1.2.el7.x86\_64

vmlinuz-3.10.0-123.el7.x86\_64

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

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

/root/nsd01.txt

/root/nsd02.txt

[root@Gay ~]# find /root/ -name "nsd\*" -type f -exec cp {} /mnt/ \;

[root@Gay ~]# ls /mnt/

nsd01.txt nsd02.txt

Day 05 管理用户组 tar备份与恢复 NTP时间同步 cron计划任务

一、管理用户和组

1. 用户帐号：a)可以登录系统 b)实现访问控制（不同用户不同权限）

组帐号：方便对用户帐号进行管理 权限方面

唯一标识：UID GID

· 管理（root）：UID永远是0

· 组帐号：基本组（私有组） 附加组（公共组 从属组）

· Linux中，一个用户至少属于一个组

2.添加用户

用户基本信息存放在/etc/passwd文件（户口本）

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

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

用户名:密码占位符号:UID:基本GID:用户描述信息:家目录位置:默认的解释器

· 使用useradd命令名

· 常用命令选

- useradd [选项]... 用户项

- 用户uid、-d家目录路径、-s登陆为Shell（指定登陆的解释器程序）、-G附加组

- /sbin/nologin：不能登陆操作系统

- id 用户名 查看用户基本信息，如用户不存在，显示未查找到该用户

- groupadd 创建组

实例：

[root@server ~]# useradd nsd01

[root@server ~]# grep 'nsd01' /etc/passwd

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

[root@server ~]# useradd -u 2333 nsd02

[root@server ~]# grep 'nsd02' /etc/passwd

nsd02:x:2333:2333::/home/nsd02:/bin/bash

[root@server ~]# useradd -d /opt/nsd03 nsd03

[root@server ~]# grep 'nsd03' /etc/passwd

nsd03:x:2334:2334::/opt/nsd03:/bin/bash

[root@server ~]# useradd -s /sbin/nologin nsd04

[root@server ~]# grep 'nsd04' /etc/passwd

nsd04:x:2335:2335::/home/nsd04:/sbin/nologin

[root@server ~]# groupadd terena #创建组terena

[root@server ~]# useradd -G terena nsd05

[root@server ~]# id nsd05

uid=2336(nsd05) gid=2337(nsd05) 组=2337(nsd05),2336(terena)

3.更改密码

· 使用passwd命令

- passwd [用户名] #交互式设置密码

- echo '密码' | passwd --stdin 用户名 #非交互式设置密码

· su命令：可以实现命令行临时切换身份

· 管道命令符"|"：将前面命令的输出作为后面命令的输入

[root@server ~]# head -12 /etc/passwd | tail -5

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

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

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

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

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

4.修改用户属性

· 使用usermod命令

- usermod [选项]... 用户名

· 常用命令选项

- -u用户id、-d家目录路径、-s登陆Shell

- -g基本组

- -G附加组

5.删除用户

· 使用userdel命令

- userdel [-r] 用户名 #包括家目录一并删除

6.密码信息的存放

/etc/shadow

7.添加组

组基本信息存放在/etc/group文件中

[root@server ~]# head -1 /etc/group

root:x:0:

组名:组密码占位符:组ID:本组成员列表

· 使用groupadd命令

- groupadd [-g 组ID] 组名

· 使用gpasswd命令

- gpasswd -a 用户名 组名 #添加成员

- gpasswd -d 用户名 组名 #删除成员

8.修改组属性

· 使用groupmod命令

- groupmod [-g 组ID] [-n 新组名] 组名

9. 删除组

· 使用groupdel命令

- groupdel 组名

二、tar备份与恢复

1.归档和压缩

· 归档的含义

- 将许多零散的文件整理成一个文件

- 文件总大小基本不变

· 压缩的含义

- 案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

Windows：归档、压缩一气呵成

Linux：先归档，再压缩

Linux独有的压缩工具：gzip bzip2 xz

2.tar工具的常用选项

· tar集成备份工具

- -c：创建归档

- -x：释放归档

- -f：指定归档文件名称

- -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

- -P：保持归档内文件的绝对路径

注意：所有的操作都有f选项，且必须放在所有选项的最后

tar 选项 tar包名字 被归档的文件1 被归档的文件2...

root@server0 opt]# tar -cPf file01.tar /etc/passwd /home/ /boot/

[root@server0 opt]# ls

file01.tar rh #归档结果

[root@server0 opt]# gzip file01.tar

file01.tar.gz rh #打包结果

[root@server0 opt]# tar -czPf file02.tar.gz /etc/passwd /home/ /boot/

/etc/passwd #一步到胃

[root@server0 opt]# ls

file01.tar.gz file02.tar.gz rh #打包结果

3.tar解包

[root@server0 opt]# tar -xf file01.tar.gz

tar: 从成员名中删除开头的“/”

[root@server0 opt]# ls

boot etc file01.tar.gz file02.tar.gz home rh

指定解压路径的选项 -C

[root@server0 opt]# tar -xf file01.tar.gz -C /root/opt

案例：

[root@server0 opt]# tar -cjPf /root/backup.tar.bz2 /usr/local/

[root@server0 opt]# ls /root/backup.tar.bz2

/root/backup.tar.bz2

三、NTP时间同步

1. 查看日期和时间

[root@server0 opt]# date

2018年 02月 05日 星期一 16:08:54 CST

· 修改时间

[root@server0 opt]# date -s "年-月-日 时-分-秒"

2.NTP时间同步

· NTP网络时间协议

- 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

服务端：Linux系统上的软件，服务端软件 classroom.example.com

客户端：客户端软件 server.example.com

a)安装chrony客户端软件

[root@server0 opt]# yum -y install chrony

[root@server0 opt]# rpm -q chrony

chrony-1.29.1-1.el7.x86\_64

b) 配置chrony客户端软件

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

server classroom.example.com iburst

c)重启chrony客户端软件服务

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

[root@server0 opt]# systemctl enable chronyd #设置开机自启程序

d: daemon：守护进程

d)验证：

[root@server0 opt]# date -s "1997-5-20 20:20"

1997年 05月 20日 星期二 20:20:00 CST

[root@server0 opt]# systemctl restart chronyd

[root@server0 opt]# date

2018年 02月 05日 星期一 16:44:08 CST

四、cron计划任务

1. cron任务概述

· 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

· 系统服务：crond

· 日志文件：/var/log/crond

2.如何编写crontab任务记录

· 配置格式可参考/etc/crontab文件

时间 执行任务

- 分 时 日 月 周 任务命令行（绝对路径）

\* \* \* \* \*

# "\*" 代表任意，"\*/n"代表频率，"-"代表连续时间，","代表不连续时间，24小时制

· 使用crontab命令

- 编辑：crontab -e [-u 用户名]

· grep 中匹配空行

- '^$' 代表空行

- grep -v '^$' 目录

Day 06 权限和归属 使用LDAP认证 家目录漫游

一、权限和归属

1.基本权限

a) 访问方式（权限）

- 读取：允许查看内容-read

- 写入：允许修改内容-write

- 可执行：允许运行和切换-execute

· 对于文本文件：

r:cat less head tail

w:vim 能够保存

x:Shell脚本

b) 权限适用对象（归属）

- 所有者：拥有此文件/目录的用户-user

- 所属组：拥有此文件/目录的组-group

- 其他用户：除所有者、所属组以外的用户-other

2.查看权限

a)命令ls -l

b)开头含义：

· 以 - 开头：文本文件

· 以d开头：目录

· 以l开头：快捷方式（链接）

c) 设置基本权限

· 使用chmod命令

- chmod [-R] 归属关系+-=权限类别 文档

[-R]：递归设置

d) 如何判断Linux中用户的权限

· 查看用户对于该文档 所属的身份 匹配即停止

· 查看相应身份的权限

常见的提示信息：权限不足

Permission denied

目录的r权限：能够ls浏览此目录内容

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

目录的x权限：能够cd切换到此目录

e) 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...

3.附加权限（特殊权限）

a) Set GID

· 附加在属组的x位上

- 属组的权限标识会变为s（s覆盖执行权限x，分大小写区别原执行权限）

- 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

- 继承父目录所属组身份

b) 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（禁止操作别人文档）

c) Set UID

· 附加再属主的x位上

- 属主的执行标识会变成s

- 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

4.acl访问控制列表

a)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

- 无法实现更精细的控制

· 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

b) 命令设置：

· 使用getfacl、setfacl

- setfacl [-R] -m u:用户名:权限类别 文档... #添加策略

- setfacl [-R] -m g:组名:权限类别 文档...

- setfacl [-R] -x u:用户名 文档... #删除指定策略

- setfacl [-R] -b 文档... #清空策略

二、使用LDAP认证

1.什么是LDAP？

· 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的信息，存储方式类似于DNS分层结构

- 提供的信息包括：用户名、密码、通信录、主机名映射记录、……

· 为一组客户机提供可登陆的用户帐号

- 本地用户：/etc/passwd /etc/shadow

- 网络用户：用户名、密码信息存储在LDAP服务端

服务端：classroom.example.com

客户端：指定LDAP服务端位置（域名）

a)安装一个客户端sssd软件

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

b) authconfig-gtk图形的配置

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

c) authconfig-gtk 启动

选择LPAD

dc=example,dc=com

classroom.example.com

TLS加密，证书加密

systemctl restart sssd

systemctl enable sssd

三、家目录漫游

1.共享服务

· NFS共享：网络文件系统

- 由NFS服务器将指定的文件夹共享给客户机

- 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

- 类似于EXT4、XFS等类型，只不过资源在网上

· 客户端：server0.example.com

1)查看共享classroom.example.com

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

Export list for classroom.example.com:

/home/guests 172.25.0.0/255.255.0.0

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

[root@server0 ~]# mount 172.25.254.254:/home/guests /home/guests

[root@server0 ~]# su - ldapuser0

四、autofs工具配置

1.装包

yum -y install autofs

2.修改主配置文件

vim /etc/auto.master

/home/guests /etc/auto.guests #设置触发式挂载

:wq

3.修改次级配置

\* -rw classroom.example.com:/home/guests/& (&:和前一个\*内容相同)

:wq

4.重启服务

systemctl restart autofs

systemctl enable autofs

Day 01 分区规划及使用 LVM逻辑卷

一、分区规划及使用

1.硬盘分区管理

· 每个扇区，512字节

· 识别硬盘 => 分区规划 => 格式化 => 挂载使用

· 格式化 -> 文件系统的作用：数据在空间中的排列规则

· lsblk 查看本机识别的硬盘

[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

2. 分区规划

· MBR/msdos分区模式（主引导记录分区方案）

- 1~4个主分区，或者0~3个主分区+1个扩展分区(n个逻辑分区)

- 最大支持容量为2.2TB的硬盘

- 扩展分区不能格式化

- 分区类型：主分区 扩展分区 逻辑分区

- 一个硬盘最多只能有4个主分区

· 使用fdisk分区工具 - 交互式管理工具

- 查看分区表

fdisk -l /dev/sda

- 修改硬盘分区表

fdisk 硬盘设备

· 常用交互指令

m 列出指令帮助

p 查看现有分区表

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

3.分区格式化

· 常用的格式化工具

- 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

- 查看分区设备信息

blkid 设备路径

4.挂载使用

df -h 查看挂载分区信息

· 挂载vdb1分区

[root@server0 ~]# mkdir /part1

[root@server0 ~]# mount /dev/vdb1 /part1/

[root@server0 ~]# df -h

文件系统 容量 已用 可用 已用% 挂载点

/dev/vda1 10G 3.0G 7.0G 31% /

devtmpfs 906M 0 906M 0% /dev

tmpfs 921M 80K 921M 1% /dev/shm

tmpfs 921M 17M 904M 2% /run

tmpfs 921M 0 921M 0% /sys/fs/cgroup

/dev/vdb1 2.0G 33M 2.0G 2% /part1

/dev/vdb2 2.0G 33M 2.0G 2% /part2

5.综合分区

· 最多4个主分区，如果需要创建更多的分区，则将一个主分区换成扩展分区，在扩展分区上创建逻辑分区

· 刷新命令

- partprobe

· 使用磁盘空间：

1.查看识别的磁盘 lsblk

2.划分分区 fdisk

3.刷新分区 partprobe

4.格式化分区 mkfs.格式 设备分区

5.挂载使用 mount

6.实现开机自动挂载 /etc/fstab

· 实现开机自动挂载

- 配置文件/etc/fstab的记录格式

设备路径 挂载点 文件系统类型 参数 备份标记 检测顺序

/dev/vdb1 /part1 xfs defults 0 0

/dev/vdb2 /part2 xfs defults 0 0

二、LVM逻辑卷

1.作用：

a)整合分散的空间

b)逻辑卷空间可以扩大

c)支持线上操作（不需要umount）

2.创建逻辑卷：

a)创建物理卷

b)创建卷组（可以同时创建物理卷）

[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

PV VG Fmt Attr PSize PFree

/dev/vdc1 systemvg lvm2 a-- 10.00g 10.00g

/dev/vdc2 systemvg lvm2 a-- 10.00g 10.00g

[root@server0 ~]# vgs #查看卷组信息

VG #PV #LV #SN Attr VSize VFree

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

c) 从卷组中划分逻辑卷

· 命令格式lvcreate -L 大小 -n逻辑卷名字 卷组名字

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

Logical volume "mylv" created

[root@server0 ~]# lvs

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

mylv systemvg -wi-a----- 16.00g

d) 使用逻辑卷

[root@server0 ~]# mkfs.ext4 /dev/systemvg/mylv

[root@server0 ~]# mkdir /test

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

/dev/systemvg/mylv /test ext4 defaults 0 0

[root@server0 ~]# mount -a

[root@server0 ~]# df -h

文件系统 容量 已用 可用 已用% 挂载点

/dev/vda1 10G 3.0G 7.0G 31% /

devtmpfs 906M 0 906M 0% /dev

tmpfs 921M 80K 921M 1% /dev/shm

tmpfs 921M 17M 904M 2% /run

tmpfs 921M 0 921M 0% /sys/fs/cgroup

/dev/vdb1 2.0G 33M 2.0G 2% /part1

/dev/vdb2 2.0G 33M 2.0G 2% /part2

/dev/mapper/systemvg-mylv 16G 45M 15G 1% /test

2.扩展逻辑卷

a)卷组有足够的空间

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

Extending logical volume mylv to 18.00 GiB

Logical volume mylv successfully resized

[root@server0 ~]# lvs

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

mylv systemvg -wi-ao---- 18.00g

[root@server0 ~]# vgs

VG #PV #LV #SN Attr VSize VFree

systemvg 2 1 0 wz--n- 19.99g 1.99g

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

/dev/mapper/systemvg-mylv 16G 45M 15G 1% /test

· 扩展空间后，还需要扩展文件系统的大小

1)扩展文件系统的大小

ext4:resize2fs

xfs:xfs\_growfs

[root@server0 ~]# resize2fs /dev/systemvg/mylv #刷新文件系统

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

/dev/mapper/systemvg-mylv 18G 44M 17G 1% /test

b)卷组没有足够的空间

扩展逻辑卷的空间

3.逻辑卷也可以缩减

a)缩减文件系统

b)缩减空间

4.卷组划分空间的单位：PE

PE的大小可以更改，vgchange -s命令

5.删除逻辑卷

删除逻辑卷(lv) --> 删除卷组 --> 删除物理卷

Day 02 Shell脚本基础 使用变量 条件测试及选择 列表式循环

一、Shell脚本基础

1.脚本：

· 一个可以运行的文本文件，运行后可实现某种功能

· 提前设计可执行语句，用来完成特定的任务的文件

- 解释型程序

- 顺序、批量执行

· 将命令都写入文本文件，然后赋予文本文件执行权限（命令的堆积）

· 非交互式

- 需要提前设计、智能化难度大

- 批量执行、效率高

- 方便在后台静默执行

· 规范：

- #! 环境声明（声明下列可执行代码用什么程序翻译）

- # 注释文本

- 可执行代码

2.简单脚本技巧

· 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

· 免交互处理

- 脚本一般在后台执行，要尽量减少人工交互的语句

· 脚本输出信息的处理

- 记录有价值的信息（>> /var/log/foo.log）

- 屏蔽无价值的、干扰性的信息:黑洞设备(垃圾场w)（&> /dev/null）

- 关于重定向输出：

> ：只收集前面命令的正确输出，将其写入文本文件

2> ：只收集前面命令的错误输出，将其写入文本文件

&> ：收集前面命令的正确和错误输出，将其写入文本文件

- 自定义输出：echo '文本字符串'

3.为了增加脚本适应多变的环境、多变的需求，以及为了方便性，我们要使用变量

- 变量：会变化的量，以不变的名称，存储可以变化的值

- 变量的定义：变量命=变量值

· 为了降低脚本使用的复杂性，使用交互式

- read：产生交互的方式，将用户从键盘的输入记录作为输出赋值给一个变量

read -p '请输入你要创建的用户名:' user

· 设置变量时的注意事项

- 若指定的变量名已存在，相当于为此变量重新赋值

- 等号两边不能有空格

- 变量名由字母/数字/下划线组成，区分大小写

- 变量名不能以数字开头，不要使用关键字和特殊字符

· 基本格式

- 引用变量值：$变量名

- 查看变量值：echo $变量名;和常量混用：echo ${变量名}常量

· 补充：

$[] ：运算

' ' ：让所有的特殊字符变成普通字符

$() ：将命令的输出结果作为参数，与``作用相同

3. 变量的种类

· 环境变量

- 常见的环境变量

PWD、PATH、USER、LOGNAME

SHELL、HOME

USER:当前登陆的用户名

- 环境变量区分字母大小写

· 位置变量

- 在执行脚本时提供的命令行参数

· 预定义变量

$# 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，其他值异常

二、条件测试及选择

1.条件测试

· 常用的测试选项

-e ：文档存在为真

-d ：存在并且为目录，才为真

-f ：存在并且为文本文件，才为真

-r ：存在并且有读取权限，才为真

-w ：存在并且有写入权限，才为真

-x ：存在并且有执行权限，才为真

· 比较整数（带e字母的都有 等于 二字）

-gt ：大于

-ge ：大于等于

-eq ：等于

-ne ：不等于

-lt ：小于

-le ：小于等于

· 字符串对比

==:相等为真

!=:不相等为真

2.if选择结构

· 双分支：当条件满足/不满足时，分别作xx、yy处理

· 多分支处理：

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

else

命令序列zz

fi

经典脚本之成绩：

用户输入考试成绩

大于等于85 输出 优秀

大于等于70 输出 良好

大于等于60 输出 合格

以上条件均不满足 输出 再牛的肖邦，也弹不出哥的悲伤

三、列表式循环

1.适用于反复工作的场景

2.for循环处理

· for 变量名 in 值列表

do

扎针 抽血

done

Day 03 系统安全保护 配置用户环境 配置高级连接 防火墙策略管理

一、系统安全保护

1.SELinux概述

· Security-Enhanced Linux

- 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

- 集成到Linux内核（2.6及其以上）中运行

- 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的运行模式

- enforcing(强制)、permissive(宽松)

|

|重启

|reboot

V

- disable(彻底禁用)

查看SELinux状态：getenforce

· 切换运行模式

- 临时切换：setenforce 1|0

- 固定配置：/etc/selinux/config 文件

2.防火墙策略管理

· 作用：隔离

- 硬件防火墙

- 软件防火墙

· 搭建web

- 服务端：server0.example.com

1)安装服务端软件 httpd(完善，均衡) Nginx(并发量高，自动化运维) Tomcat(Java)

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

2)启动httpd服务，设置为开机自启

[root@server0 ~]# systemctl restart httpd

[root@server0 ~]# systemctl enable httpd

3)写自己的主页

默认网页文件路径：/var/www/html

默认网页文件的名字：index.html

默认书写网页内容的语言：html

- 客户端：desktop0.example.com

· 搭建ftp

- 服务端：server0.example.com

1) 安装服务端软件

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

2) 启动vsftpd服务，设置为开机自启

[root@server0 ~]# systemctl restart vsftpd

[root@server0 ~]# systemctl enable vsftpd

3) 默认共享路径：/var/ftp

- 客户端：desktop0.example.com

· 防火墙：firewalld服务基础

3.RHEL7的防火墙体系

a) 系统服务：firewalld

b) 管理工具：firewall-cmd、firewall-config

c) 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等少数几个服务

- trusted：允许任何访问

- block：阻塞任何来访请求

- drop：丢弃任何来访的数据包

- ......

· 配置规则的位置

- 运行时 (runtime)

- 永久配置选项 (-permanent)

· 匹配规则：原则 匹配即停止

· 防火墙判定进入哪一个区域的规则：

1) 查看客户端请求数据包中，源ip地址，查看自己所有的区域，哪一个区域有该源ip地址进入哪一个区域

2) 如果规则1不符合则进入默认区域

· 预设安全区域

- public：仅允许访问本机的sshd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，客户端会接收到拒绝信息）

- drop：丢弃任何来访的数据包（直接丢弃，节省服务器的资源）

d) 防火墙对源ip地址的控制

· 方式1-宽松模式：默认区域未trusted，将想要拒绝的源ip地址放入到block或drop中

· 方式2-严格模式：默认区域未block或drop，将想要允许的源ip地址放入到trusted中

二、端口：数字 编号，用来标识进程或程序

1.默认端口号

· ftp：21

· ssh:22

· Telnet:23

· SMTP:25

· DNS:53

· DHCP:UDP 67/68

· tftp:69

· http：80

· NTP:123

· https:443

2.端口转发

· 利用防火墙进行对指定端口的转发

# firewall-cmd --permanent --zone=public

--add-forward-port=port=5423:proto=tcp:toport=80

三、配置高级连接

1.配置高级连接（聚合连接 网卡绑定 链路聚合） 参考 man teamd.conf 全文查找/example 按n跳转匹配项

· 热备份（activebackupo）连接冗余（活跃状态 备份状态）

eth1 eth2

team0虚拟网卡

2.制作网卡绑定

a)制作虚拟网卡team0

# nmcli connection add type team

autoconnect yes con-name team0 ifname team0

config '{"runner": {"name": "activebackup"}}'

# nmcli connection 添加 类型为 team 的设备

每次开机自动启用 配置文件命名为 team0 网卡显示的名字为 team0

team0网卡内部成员工作模式为 '{"runner": {"name": "activebackup"}}' （热备）

# ifconfig

2. 为 team0添加成员

# nmcli connection add type team-slave con-name team0-1 ifname eth1 master team0

# nmcli connection add type team-slave con-name team0-2 ifname eth2 master team0

# 添加 类型为 team-slave 的设备 配置文件命名为 team0-1 网卡为eth1

主设备为team0

3.配置team0的ip地址与激活

# nmcli connection modify team0 ipv4.method manual

ipv4.addresses 192.168.1.1/24 connection.autoconnect yes

# nmcli connection up team0 #激活team0网卡

# nmcli connection up team0-1 #激活team0-1成员

# nmcli connection up team0-2 #激活team0-2成员

如果错了：

1.删除

[root@server0 ~]# nmcli connection delete team0

[root@server0 ~]# nmcli connection delete team0-1

[root@server0 ~]# nmcli connection delete team0-2

2.重新配置

专用于team测试查看的命令

[root@server0 ~]# teamdctl team0 state #查看team0信息

[root@server0 ~]# ifconfig eth1 down #禁用eth1网卡

[root@server0 ~]# teamdctl team0 state #查看team0信息

四、IPv6地址

1.IPv4地址：32个二进制 点分隔4个部分 十进制表示

2.IPv6地址：128个二进制 冒号分隔8个部分 十六进制表示

==============================================================

Day 04 配置SMB共享 配置NFS共享

一、配置永久主机名：server0.example.com

/etc/hostname

二、配置静态ip地址

nmcli connection modify 'System eth0' ipv4.method manual ipv4.addresses '172.25.0.11/24 172.25.0.254' connection.autoconnect yes

nmcli connection up 'System eth0'

三、配置DNS服务器地址：172.25.254.254

将域名解析成IP地址

echo 'nameserver 172.25.254.254' > /etc/resolv.conf

cat /etc/resolv.conf

nslookup classroom.example.com

四、搭建yum服务端

echo '[rhel7]

> name=rhel 7.0

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

> enabled=1

> gpgcheck=0' > /etc/yum.repos.d/dvd.repo

五、配置SMB共享

1.设置双端防火墙默认区域为trusted

firewall-cmd --set-default-zone=trusted

2.服务名：smb

包名：samba

3.配置SMB共享 跨平台的共享 Windows与Linux

4.Samba服务基础

· 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

服务端：server0.example.com

1) 安装软件包：samba

yum -y install samba

2)建立samba共享帐号

useradd harry

useradd kenji

useradd chihiro

echo redhat | passwd --stdin harry

echo redhat | passwd --stdin kenji

echo redhat | passwd --stdin chihiro

3)管理共享帐号

· Samba用户——专用来访问共享文件夹的用户

- 采用独立设置的密码

- 但需要提前建立同名系统用户

· 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

· 示例：

pdbedit -a harry #将本地用户harry设置为Samba共享帐号

pdbedit -a kenji

pdbedit -a chihiro

交互设置密码为123

pdbedit -L #显示本地有哪些Samba共享帐号

harry:1001:

chihiro:1003:

kenji:1002:

· 配置文件及参数

修改/etc/samba/smb.conf

[自定共享名]

path = 文件夹绝对路径

[global]

workgroup = STAFF

[common]

path = /common

4) 重启smb服务

systemctl restart smb

systemctl enable smb

5)设置SELinux

布尔值——功能的开关

· 查看服务状态(加-P永久实现)

getsebool -a | grep samba

setsebool samba\_export\_all\_ro on

客户端：desktop0.example.com

1)所需软件包：samba-client

yum -y install samba-client

2) 列出共享资源

- smbclient -L 服务器地址

3) 连接到共享文件夹

- smbclient -U harry //server0/common

5.采用更加方便科学的访问方式(挂载访问)

· 客户机上操作：

1)安装软件包

yum -y install cifs-utils

2)挂载

mount -o user=harry,pass=123 //172.25.0.11/common /common

3)查看挂载结果

ls /common

4)客户端实现开机自动挂载 /etc/fstab

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

#\_netdev：标识本设备为网络设备(先启动网络服务具备ip地址等网络参数后，再进行挂载)

umount /common #卸载已挂载目录

df -h #查看卸载是否成功

mount -a #挂载

df -h #检验挂载结果

6. 实现读写的samba共享

· 服务端：server0.example.com

1) 修改配置文件，设置新的共享

vim /etc/samba/smb.conf

[devops]

path = /devops

write list = chihiro

2) 重建相应的目录

mkdir /devops

echo abc > /devops/abc.txt

ls /devops

3) 重启smb服务，设置开机自启

systemctl restart smb

systemctl enable smb

4) 设置本地权限

setfacl u:chihiro:rwx /devops

getfacl /devops

5) 修改SELinux功能开关

getsebool -a | grep samba

setsebool samba\_export\_all\_rw on

6) 修改防火墙设置

· 客户端：desktop0.example.com

1)vim /etc/fstab

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

2)systemctl restart smb

7.多用户(multiuser)的samba共享,专为普通用户设计

· SMB客户端的multiuser挂载技术

- 管理员只需要作一次挂载

- 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

7.1 客户端操作：

· 修改/etc/fstab配置文件，添加参数

- multiuser，提供客户端多个用户身份的区分支持

- sec=ntlmssp，提供NT局域网管理安全支持

vim /etc/fstab

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

umount /devops

mount -a

df -h

su - student

cifscreds add -u chihiro 172.25.0.11

Password:

ls /mnt/dev

touch /mnt/dev/haha.txt

exit

六、配置NFS共享

1.NFS共享概述

· 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

2. 搭建NFS

· 服务端：server0.example.com

rpm -q nfs-utils

1）修改/etc/exports

- 文件夹路径 客户机地址（权限） 客户机地址（权限）

使用exportfs可以重载更新过的配置

- exportfs -r

vim /etc/exports

/public \*(ro)

2）创建共享目录

mkdir /public

cp /etc/passwd /public/passwd.txt

systemctl restart nfs-server

systemctl enable nfs-server

· 客户端：

mkdir /public

showmount -e 172.25.0.11 #查看有哪些nfs共享

vim /etc/fstab #实现开机自动挂载

172.25.0.11:/public /public nfs defaults,\_netdev 0 0

mount -a

df -h

七、分区规划

1.查看识别的磁盘 lsblk

2.划分分区

fdisk /dev/vdb

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 2G 0 part

├─vdb2 253:18 0 2G 0 part

└─vdb3 253:19 0 2G 0 part

3.将第一个主分区格式化为xfs文件系统，开机自动挂载到/mnt/mypart

# mkfs.xfs /dev/vdb1

# blkid /dev/vdb1 //查看文件系统格式

# mkdir /mnt/mypart

# echo '/dev/vdb1 /mnt/mypart xfs defaults 0 0' >> /etc/fstab

# tail -1 /etc/fstab

# mount -a

# df -h | tail -1

4.将第二个主分区和第三个主分区组成卷组systemvg

vgcreate systemvg /dev/vdb[2-3]

5.划分逻辑卷lvtest，大小为3G，格式化为ext4格式，挂载到/mnt/mylv

lvcreate -L 3G -n lvtest systemvg

mkfs.ext4 /dev/systemvg/lvtest

blkid /dev/systemvg/lvtest

mkdir /mnt/mylv

mount /dev/systemvg/lvtest

df -h | tail -1

6.逻辑卷lvtest扩大到6G

1)通过fdisk分出4G的逻辑分区vdb5

2)partprobe

3)vgextend /dev/vdb5

4)lvextend -L 6G /dev/systemvg/lvtest

5)resize2fs /dev/systemvg/lvtest #如果是xfs格式，则用xfs\_growfs命令刷新文件系统

Day 05 iSCSI技术应用 数据库服务基础 管理表数据

远程端笔记下载方法：

yum -y install git

git clone http://github.com/redhatedu/course //完整下载

cd course

git

pull //更新下载（必须在course目录下执行）

git checkout 文件名 //下载某个文件

一、iSCSI共享(磁盘共享)internet scsi

1.iSCSI磁盘的工作模式，Internet SCSI，网际SCSI接口

- 一种基于C/S架构的虚拟磁盘技术

- 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（需要的三个概念）

- backstore，后端存储

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

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

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

· target的共享名有要求(要符合iqn规范)

iqn.年-月.反转域名:任意字串

如：

iqn.2018-2.com.example:jpg

环境前提:

修改两台虚拟机的防火墙：

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

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

实验步骤：

a) 在server0上准备一个磁盘分区(vdb1)3G

b) 安装软件包，修改配置,重启服务

# yum -y install targetcli

# targetcli

/> ls

/> cd /

//建立后端存储

/> /backstore/block create (name=)nsd (dev=)/dev/vdb1

/\*\*

\* /> /iscsi create iqn.8102-12.com.example:server0

\* 创建一个iscsi共享（target磁盘组,客户端访问目标)

\* 磁盘组共享命名遵循iqn原则(qin.yyyy-mm.倒序域名:自定义标识)

\*\*/

/> /iscsi create iqn.2018-02.com.example

//创建逻辑单元，把共享名和后端的设备通过lun关联在一起

/> /iscsi/iqn.2018-02.com.example.com/tgp1/luns create /backstores/block/back\_store

/\*\*

\* 创建一个访问的口令:iqn.2018-02.com.example:desktop0，以后仅知道口令的客户端才可以访问共享

\*设置访问控制,设置客户端访问需要声明的名称

\*\*/

/> /iscsi/iqn.2018-02.com.example:data/tpg1/acls create iqn.2018-02.com.example:desktop0

//以后客户端访问本机（172.25.0.11）的3260端口(默认端口)可以访问到共享

/> /iscsi/iqn.2018-02.com.example:data/tpg1/portals create 172.25.0.11

//重启服务

# systemctl restart target

# systemctl enable target

c) 客户端访问共享

//查看硬盘设备

# 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

//实验环境可以跳过，生产环境需要检查iscsi是否安装

# yum -y install iscsi

//修改配置文件,指定口令,重启服务

# vim /etc/iscsi/initiatorname.iscsi

# systemctl daemon-reload

# systemctl restart iscsid

//查看、复制命令格式

# man iscsiadm

关键词:example

//挂载172.25.0.11服务器上的共享：

# iscsiadm --mode discoverydb --type sendtargets --portal 172.25.0.11 --discover

172.25.0.11:3260,1 iqn.2018-02.com.example:data

# iscsiadm --mode node --targetname iqn.8102-07.com.example:server0 --portal 172.25.0.11:3260 --login

//如果挂载提示authentication,口令不对

# systemctl restart iscsid

# lsblk

NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT

sda 8:0 0 3G 0 disk

vda 253:0 0 10G 0 disk

└─vda1 253:1 0 10G 0 part /

vdb 253:16 0 10G 0 disk

//可以看到sda硬盘成功挂载

二、数据库服务基础

1.DB(database)数据库：存放数据的地方

存放形式： 数据表，数据列，表头

DBMS（数据库管理系统）是具体的产品：excel,access,mysql,sql server,oracle

2.MySQL AB ==> Sun MySQL ==> Oralce MySQL

MariaDB == MySQL

RHEL7(内置的数据库是MariaDB)

在server0操作

- mariadb-server:提供服务端有关的系统程序,默认端口3306

# yum -y install mariadb-server mariadb

//mariadb-server服务器

//mariadb客户端(为mariadb-server前置)

//使用客户端软件连接数据库服务器做增、删、改、查操作

# systemctl start mariadb

# systemctl enable mariadb

mysql

//查看有哪些目录

MariaDB [(none)]> show databases;

//进入数据库

MariaDB [(none)]> use mysql

//查看有哪些数据表

MariaDB [mysql]> show tables;

//查看数据表里的数据

MariaDB [mysql]> select \* from user;

//退出数据库

MariaDB [mysql]> exit

/\*为数据库管理员设置密码\*/

· 数据库管理员root:数据库MariaDB最高权限用户,存放在mysql库中user表下

· 系统管理员root:Linux系统最高权限用户,存放在/etc/passwd文件中

· 格式:# mysqladmin [-u用户名] [-p旧密码] password '新密码'

# mysqladmin -uroot password '123456' //给mariadb数据库的管理员root修改密码，密码为123456

//直接登陆数据库会失败

# mysql

/\*交互式设置密码\*/

# mysql -u root -p

Enter password:

//使用用户与密码登陆

# mysql -uroot -p123456

//创建数据库nb

MariaDB [(none)]> create database nb;

//切换至数据库nb

MariaDB [(none)]> use nb;

· 禁止监听,只服务于本机

# vim /etc/my.cnf

[mysql]

skip-network

# systemctl restart mariadb

MariaDB [(nb)]> create table info

-> (id int, //第一列是id，数据类型为int型

-> name varchar(10), //第二列是name，数据类型为字符串，10位

-> password varchar(20)); //第三列是password，数据类型为字符串，20位

//创建一个表，表名info

MariaDB [(nb)]> describe info;

//查看创建的表

MariaDB [nb]> insert into info values

-> (1,'tom','123456');

//向info中插入数据

MariaDB [nb]> insert into info values

-> (1,'tom','123456'),

-> (2,'dachui','654321'),

-> (3,'jerry','123456'),

-> (4,'lucy','123456');

//插入多条数据的格式

MariaDB [nb]> select \* from info;

· 跨库查询

MariaDB [nb]> select \* from mysql.user;

//查看info的所有数据

MariaDB [nb]> drop table nb;

//删除名为nb的表

3.数据库的备份与还原

# mysqldump -uroot (-p123456) nb > my.bak

//把名称为nb的数据库备份到my.bak

# mysql -uroot (-p123456) nb < my.bak

//使用my.bak将名为nb的数据库还原

4.用户的创建

- 默认mariadb数据库仅有一个帐号（root)

- 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'

· 创建新的账户

//授予账户tom对nb数据库下所有表的所有权限

//all指所有权限

//可以写具体的指令[select,insert]

//tom@'localhost'（%表示任意） 表示tom只能从本机登陆数据库

//identified by设置密码

# mysql -uroot -p123456

MariaDB [nb]> grant all on nb.\*

-> to tom@'localhost'

-> identified by '123456';

5.表记录的操作（表数据的操作）:增删查改

//查看表中所有数据

MariaDB [nb]> select \* from info;

//查看info表中name与password信息

MariaDB [nb]> select name,password from info;

//查看info表中满足password为123456的条件的信息

MariaDB [nb]> select \* from info where password='123456';

//查看表结构

MariaDB [mysql]> desc user[表名称];

//更新info表

MariaDB [nb]> update info set password = 'mmp'

-> where name='tom';

//更新name为Lily的行的密码为123

//删除info表中id为3的数据

MariaDB [nb]> delete from info where id=3

//查看info表中的数据

//数据符合密码为123456且id等于1

MariaDB [nb]> select \* from info

-> where password='123456' and id=1;

//多表查询

MariaDB [nb]> SELECT count(\*)

-> FROM base,location

-> WHERE base.name="Barbara" AND

-> location.city="Sunnyvale" AND

-> base.id=location.id;

6.禁止空密码root用户访问数据库

MariaDB [none]> use mysql;

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

MariaDB [mysql]> delete from user where password='';

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

MariaDB [mysql]> flush privileges;

MariaDB [mysql]> exit

# mysql -u root -h server0.example.com

ERROR 1130 (HY000): Host 'server0.example.com' is not allowed to connect to this MariaDB server

· 登陆不成功为正确

========================================================

Day 06 HTTP服务基础 网页内容访问 部署动态网站

一.HTTP服务基础

· 基于B/S(Browser/Server)架构的网页服务

- 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

- Listen:监听地址:端口(80)

- ServerName:本站点注册的DNS名称(空缺)

- DocumentRoot:网页根目录(/var/www/html)

- DirectoryIndex:起始页/首页文件名(index.html)

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

实验一:搭建基本的WEB服务器

虚拟机Server0:

1.安装httpd(Apache)软件包

# systemctl install -y httpd

2.启动httpd服务,设置开机自启动

# systemctl start httpd

# systemctl enable httpd

3.书写页面

# vim var/www/html/index.html

# cat /var/www/html/index.html

4.虚拟机Desktop0访问测试

# firefox 172.25.0.11

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

DNS服务器:将域名解析为IP地址

classroom.example.com上的域名

server0.example.com

www0.example.com

webapp0.example.com

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

实验二:配置网站的DNS域名

ServerName:本站点注册的DNS名称(空缺)

虚拟机Server0:

1.修改配置文件:

路径: /etc/httpd/conf/httpd.conf

95行,注释去掉

ServerName server0.example.com:80

# systemctl restart httpd

虚拟机Desktop0:

# firefox server0.example.com

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

实验三:修改网页文件存放路径

DocumentRoot:网页根目录(var/www/html)

虚拟机Server0

1.修改配置文件

路径: /etc/httpd/conf/httpd.conf

2.创建路径及网页文件

# mkdir /var/www/myweb

# echo '<h1>Wo Shi MyWeb' > /var/www/myweb/index.html

3.重启服务

# systemctl restart httpd

4.访问测试

# firefox server0.example.com

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

server0.example.com

服务端:/var/www/myweb/pub

=====================================================

二.虚拟Web主机

1.作用:由同一台服务器提供多个不同的Web站点

·区分方式(构建方式)

- 基于域名的虚拟主机

- 基于端口的虚拟主机

- 基于IP地址的虚拟主机

配置文件:

/etc/httpd/conf/httpd.conf //主配置文件

/etc/httpd/conf.d/\*.conf //子配置文件

IP:端口

<VirtualHost \*:80>

ServerName DNS名称

DocumentRoot 网页根目录

</VirtualHost>

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

实验四:搭建基于域名的虚拟Web主机

虚拟机Server0:

1.创建子配置文件

# vim /etc/httpd/conf.d/dc.conf

<VirtualHost \*:80>

ServerName www0.example.com

DocumentRoot /var/www/nsd01

</VirtualHost>

<VirtualHost \*:80>

ServerName webapp0.example.com

DocumentRoot /var/www/nsd02

</VirtualHost>

2.创建网页根目录

# mkdir /var/www/nsd01 /var/www/nsd02

# echo '<h1>wo shi nsd01' > /var/www/nsd01/index.html

# echo '<h1>wo shi nsd02' > /var/www/nsd02/index.html

3.重启服务

# systemctl restart httpd

注意:

一旦使用虚拟Web主机,主配置文件中ServerName与DocumentRoot失效,所有站点都需要使用虚拟Web主机来实现

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

配置网页内容访问

· 使用<Directory>配置区段

- 每个文件夹自动继承父目录的访问控制

- 除非针对子目录有明确设置

· 格式:

<Directory />

AllowOverride none

Require all denied

</Directory>

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

实验五: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

虚拟机Server0

1.查看http://server0.example.cpm 的DocumentRooot目录

# vim /etc/httpd/conf.d/nsd01.conf

2.创建目录,建立网页文件

# mkdir /var/www/myweb/private

# echo '<h1>wo shi private' > /var/www/myweb/private/index.html

# cat /var/www/myweb/private/index.html

<h1>wo shi private

3.本机及虚拟机desktop0分别测试

# firefox server0.example.com/private

# curl server0.example.com/private/

4.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

# vim /etc/httpd/conf.d/tc.conf

<Directory /var/www/myweb/private>

Require ip 172.25.0.11

</Directory>

# systemctl restart httpd

5.本机及虚拟机Desktop0分别测试

# firefox server0.example.com/private

# curl server0.example.com/private/

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

使用自定义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目录

# mkdir /webroot

# echo '<h1>wo shi webroot' > /webroot/index.html

# cat /webroot/index.html

<h1>wo shi webroot

2.修改虚拟Web主机配置文件

/etc/httpd/conf.d/dc.conf

<VirtrualHost \*:80>

ServerName server0.example.com

DocumentRoot /webroot

</VirtualHost>

3.修改访问控制配置文件

/etc/httpd/conf.d/tc.conf

<Directory /webroot>

Require all granted

</Directory>

4.重启httpd服务

5.SELinux策略,安全上下文值(路径的进入及配置文件读取)

· 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

//查看安全上下文

# ls -Zd /var/www

# ls -Zd /webroot

# chcon -R --reference=/var/www /webroot

# ls -Zd /webroot

# chcon -t public\_content\_t /var/ftp/log2.tar

=======================================================

三.部署动态网站

· 静态页面

- 服务端原始页面

- 无数据处理

· 动态页面

- php

- wsgi

- jsp

虚拟机Server0

1.部署python页面

# cd /var/www/nsd02

# wget http://classroom.example.com/pub/materials/webinfo.wsgi

# cat webinfi.wsgi

2.方便用户的访问,页面跳转

# vim /etc/httpd/conf.d/nsd01.conf

<VirtualHost \*:80>

ServerName webapp0.example.com

DocumentRoot /var/www/nsd02/webinfo.wsgi

Alias / /var/www/nsd02/webinfo.wsgi

#当客户端访问网页站根目录时,实现页面跳转,将webinfo.wsgi呈现

</VirtualHost>

3.重启服务验证

# system restart httpd

# firefox webapp0.example.com

4.安装python脚本语言翻译软件,负责解析python页面的代码0

# yum install -y mod\_wsgi

5.修改虚拟Web主机配置文件,实现httpd进行转义和翻译

<VirtualHost \*:80>

ServerName webapp0.example.com

DocumentRoot /var/www/nsd02/

WSGIScriptAlias / /var/www/nsd02/webinfo.wsgi

</VirtualHost>

6.重启httpd服务验证

# systemctl restart httpd

# firefox webapp0.example.com

# curl webapp0.example.com

· UNIX时间戳:从1970-1-1 0:0:0算起,到达现在所经历q的秒数

7.修改虚拟主机侦听在端口8909

Listen 8909

<VirtualHost \*:8909>

ServerName webapp0.example.com

DocumentRoot /var/www/nsd02/

WSGIScriptAlias / /var/www/nsd02/webinfo.wsgi

</VirtualHost>

8.SELinux策略,非默认端口的开放

# semanage port -l

# semanage port -a -t http\_port\_t -p tcp 8909

-a:添加 -t:类型 -p:协议

# systemctl restart httpd

# firefox webapp0.example.com

# curl webapp0.example.com

· 端口优先级大于域名

==================================================================

Day 07 综合串讲 综合练习

一、HTTPS网站加密

HTTP是明文协议，网络中传输的任何数据都是明文，包括用户和密码，如果有人抓包，所有数据都可以获得

加密算法：

对称算法（AES,DES）

非对称算法（RSA,DSA）

信息摘要（md5,sha512,sha265）

对称加密（加密和解密是一把钥匙）适合单机加密

非对称密码（加密和解密不是一把钥匙）：

公钥和私钥

信息摘要：

# md5sum 文件名

数据完整性校验（检查数据是否被人修改过）

虚拟机Server0

1.部署网站证书

# cd /etc/pki/tls/certs/

# wget http://classroom.example.com/pub/tls/certs/server0.crt

# ls

2.部署根证书

# wget http://classroom.example.com/pub/example-ca.crt

# ls

3.部署私钥

# cd /etc/pki/tls/private

# wget http://classroom.example.com/pub/tls/private/server0.key

# ls

4.安装软件包mod\_ssl,提供安全支持

# yum install -y mod\_ssl

5.修改配置文件

# vim /etc/httpd/conf.d/ssl.conf

:set nu //添加行号

59 DocumentRoot "/var/www/html"

60 ServerName www0.example.com:443

//指定网站证书的位置

100 SSLCertificateFile /etc/pki/tls/certs/server0.crt

//指定私钥的位置

107 SSLCertificateKeyFile /etc/pki/tls/private/server0.key

//指定根证书的位置

122 SSLCACertificateFile /etc/pki/tls/certs/example-ca.crt

6.重启httpd服务

7.验证

# firefox https://www0.example.com

我了解风险 --> 添加例外 --> 确认添加例外

自己新建加密格式

<VirtualHost \*:443>

sslengine on

SSLCaCertificateFile /

SSLCertificateFile /

SSLCertificateKeyFile /

</VirtualHost>

=====================================================

二.基础邮件服务

1.种类

· 本域邮件

· 外域邮件

· 电子邮件服务器的基本功能

- 为用户用户提供电子邮件存储空间(用户名@域名)

- 处理用户发出的邮件(SMTP) ---- 传递给收件服务器

- 处理用户收到的邮件(pop3 IMAP) ---- 投递到邮箱

· mail命令发信/收信

//发信

- mail -s '标题' -r 发件人 收件人[@收件域]

//收信

- mail [-u]

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

虚拟机Server0

1.安装postfix包

# yum -y install postfix

# rpm -q postfix

2.修改配置文件

# vim /etc/postfix/main.cf

:set nu //添加行号

99 myorigin = server0.example.com //默认补全的域名后缀

116 inet\_interfaces = all //允许所有人使用邮件服务

164 mydestination = server0.example.com //判断为本域邮件

3.重启服务

# systemctl restart postfix

4.创建本地用户进行收发邮件

# useradd yg //杨过

# useradd gg //姑姑

&1 //查看编号为1

&q //退出

===================================================

三.分区工具fdisk,parted

· fdisk分区只能分4个主分区

· fdisk分区每个分区最大只能是2T

· parted可以分超过4个主分区（128），分区大小可以大于2T

· parted分区工具

- parted分区首先要选择分区的类型msdos,gpt模式

- msdos就是fdisk使用的模式

# parted /dev/vdb

(parted) mktable gpt //指定分区模式

(parted) print //查看分区表信息

(parted) mkpart //划分新的分区

分区名称？ [ ]? nsd //分区名称

文件系统类型？ [ext2]? ext4 //分区文件系统,未实装

起始点？ 0

结束点？ 2G

警告: The resulting partition is not properly aligned for best performance. //需要引导空间

忽略/Ignore/放弃/Cancel? Ignore //选择忽略

(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 nsd

(parted) unit GB //选择显示容量单位为GB

(parted) mkpart

分区名称？ [ ]?

文件系统类型？ [ext2]? ext4

起始点？ 2G

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 nsd

2 2.00GB 4.00GB 2.00GB

(parted) quit

# 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

=====================================================

四.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

- 交换文件:以文件模拟的设备充当的交换空间

1.创建交换空间

1)前提:要有空闲的分区

2)格式化文件系统

# mkswap /dev/vdb1

3)启用交换空间

# swapon /dev/vdb1

4)查看启用的交换空间

# swapon -s

5)停用交换空间

# swapoff /dev/vdb1

· 交换空间开机自动挂载

# vim /etc/fstab

/dev/vdb1 swap swap defaults 0 0

/dev/vdb2 swap swap defaults 0 0

# swapon -a

# swapon -s

===============================================================

Day 01 扩展的几个应用 发布网络YUM源 vim编辑技巧 源码编译安装 systemctl控制

一.补充应用

1.man hier

· 主要用途

/boot 存放系统引导必须的文件,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

======================================================

2..搭建教学环境

· 采用真机和虚拟机架构

1.真机: 搭建FTP服务,共享光盘所有内容

1)安装vsftpd软件

# yum -y install vsftpd

# rpm -q vsftpd

2)查看服务启动

# systemctl status vsftpd

# systemctl restart vsftpd

# systemctl enable vsftpd

3)共享光盘所有内容

· 默认共享:/var/ftp

· 服务端: 1.众多的软件包 2.仓库清单文件 3.共享的服务

# vim /etc/fstab

/var/lib/libvirt/images/iso/rhel-server-7.4-x86\_64-dvd.iso /var/ftp/rhel7 iso9660 defaults 0 0

# ls /var/ftp/rhel7

利用真机FTP服务,共享rhel6及CentOs7光盘内容

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

· 虚拟机

· 利用root进行登录,密码为123456

虚拟机A:

1.主机名:

# ehco 'svr7.tedu.cn' > hostname

# hostname svr7.tedu.cn

# exit

# login

2.配置ip地址:192.168.4.7/24

3.配置Yum仓库(以真机作为源)

虚拟机B:

1.主机名:pc207.tedu.cn //同上

2.配置ip地址:192.168.4.207/24

3.配置Yum仓库(以真机作为源)

# clone-vm7

· 补充内容:本地数据传递给网络中其他主机

# scp /etc/yum.repos.d/rhel7.repo root@192.168.4.207:/etc/yum.repos.d/

==============================================================

3.权限的数值表示

· 权限的数值化

- 基本权限: r=4,w=2,x=1

- 附加权限: SUID=4,SGID=2,Sticky Bit=1

# mkdir /nsd01

# ll -d /nsd01

# chmod 700 /nsd01

# ll -d /nsd01

# chmod 007 /nsd01

# ll-d /nsd01

# chmod 750 /nsd01

# ll -d /nsd01

· 附加权限

SUID，SGID，Sticky

SUID仅能对程序有效

/usr/bim/passwd(root root)程序属于root用户

以普通用户去执行有SUID的命令，执行时会获得root权限

# chmod 7777 文件 --> (rwsrwsrwt)

======================================================

4.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

# vim /etc/profile

HISTSIZE=1000 //默认记录1000条

5.统计文件的占用空间

- du [选项]... [目录或文件]...

- -s:只统计每个参数所占用的总空间大小

- -h:提供易读容量单位(K、M等)

6.date查看计算机时间

# date

# date +%F 日期

# date +%R 时间

# date +%Y%m%d 年，月，日year,month,day

# date +%Y:%m:%d 年，月，日

# date +%H:%M:%S 小时，分，秒

# date -s "年-月-日 时:分:秒" //修改计算机时间

# date -s "11:01" //仅修改时间

# date -s "2088-12-1 12:12"

======================================================

7.软链接和硬链接

· ln命令（给文件或目录创建快捷方式，链接）

· 软链接（符号链接）

- 软链接 --> 原始文档 --> i节点 --> 文档数据

软链接不占用空间，但是源文件删除，链接失效

# ln -s 源 目标

# ln -s /root/文档/xxd/ /root/桌面/xxd

· 硬链接

- 硬链接 --> i节点 --> 文档数据

硬链接不占用空间，源文件可以删除，链接依然能用

硬链接与原始文件必须在同一分区下

- i节点(编号) : 标识硬盘存储区域

# ln 源 目标

8.查看帮助的方法

# 命令 -h

# 命令 --help

可用 | grep 筛选有效信息

# man 命令

对man帮助时可以/搜索

- 5 配置文件的帮助信息

· man 5 passwd

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

9.zip归档工具,跨平台的压缩归档工具

linux压缩常用格式：gzip,bz2

windows压缩：rar,zip

· 归档+压缩操作

# zip -r my.zip /var/log //把/var/log目录压缩

· 释放归档+压缩

# unzip my.zip //把my.zip解压到当前

# unzip my.zip -d /tmp //把my.zip解压到/tmp

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

10.vim编辑技巧

· 命令模式：

yy,#yy 复制一行,#行

p,P 粘贴到下一行,上一行

dd,#dd 剪切一行,#行(多用于删除)

hjkl 左下上右

gg 到第一行

G 到最后一行

4G 移动光标到第4行（同 :4 ）

x(DELETE) 删除光标当前的一个字符

^(HOME) 光标到行首

$(END) 光标到行尾

d^ 从光标处删除到行首

d$ 从光标处删除到行尾

C 从光标处删除到行尾并且进入插入模式

u 撤销一步

Ctrl+r 取消撤销

ZZ 保存并退出

· 末行模式：

:4 移动到光标第4行

:w 保存（不退出）

:wq 保存退出

:q! 不保存退出

:r 读取其他文件 :r /etc/passwd

:w 另存为i/root/tmp.txt :w /root/tmp.txt

:s/旧/新 旧内容替换为新内容，仅替换当前行的第一个内容

:s/旧/新/g 替换当前行所有内容

:1,10 s/旧/新/g 替换1-10行所有的内容

:%s/旧/新/g 替换所有的内容

:set nu 显示行号

:set nonu 不显示行号

:set ai 启用自动缩进

:set noai 关闭自动缩进

=======================================================

11.源码包安装软件

a)编译安装的优势

· 二进制包可以安装软件[rpm,exe,msi,deb]

· 源码 ---> 编译 ---> 二进制

· Linux的软件多数都免费，开源

· 二进制包装软的缺点：

（不愿意花时间去封装二进制）

有些开源软件会封装成二进制，但时间会很久

· 源码编译安装：

- 主要优点：

- 获得软件的最新版，及时修复bug

- 软件功能可按需选择/定制，有更多软件可供选择

- 源码包适用各种平台

b)从ftp下载

# wget ftp://172.25.0.250/share/inotify-tools-3.13.tar.gz

· 登陆server0使用源码安装软件

# tar -xf inotify-tools-3.13.tar.gz

//解压

# cd inotify-tools-3.13/

//计算机的系统软件一般都是用c语言写的

//如QQ，office,windows,linux,画图，播放器

java,php,python,shell,C,汇编

# ./configure 检查你的计算机环境

# yum -y install gcc

//gcc是linux里面的一个C语言的解释器

#./configure 检查环境，不报错没有Error

# make

//用gcc解释器把源码转换为二进制

# make install

//把编译好的二进制程序安装到你的计算机

# inotifywait 这个能<tab>出来（说明成功）

//备注：执行./configure --prefix=/路径

//configure可以通过prefix参数，指定安装路径

//如果没有指定prefix，则一般默认在usr/local/

inotifywait -mrq /root/

再开一个终端，在/root目录做一些操作

# touch /root/tmp.txt

# echo "1" > 1.txt

# rm -f 1.txt

=====================================================

11.systemctl命令

# systemctl -t service

//列出启动的服务

# systemctl -t service --all

//列出所有的服务，包括没成功的

# systemctl stop 服务名称

//当前关闭，重启无效

# systemctl disable服务名称

//永久关闭

# systemctl enable 服务名称

//开启自启

# systemctl start 服务名称

//当前立刻启动

# systemctl restart 服务名称

//重启服务

# systemctl status 服务名称

//查看某个服务的状态

=======================================================

二.发布网络Yum源

1.yum排错

1) 格式是否正确

# cat /etc/yum.repos.d/rhel7.repo

2) 查看是否有光盘内容

# curl ftp://192.168.4.254/rhel7/

没有: 1.是否ping通 2.真机vsftpd服务是否开启

3.ls /var/ftp/rhel7 --> 没有挂载 --> /etc/fstab

4.真机防火墙是否关闭

2.传输tool包到虚拟机svr7

[root@svr7 ~]# mkdir /usr/xxd

[root@room9pc01 ~]# scp 桌面/tools.tar.gz root@192.168.4.7:/usr/xxd

[root@svr7 ~]# ls /usr/xxd

tools.tar.gz

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

虚拟机svr7:

· 创建自定义yum源:

1.解包

# tar -xf /usr/xxd/tools.tar.gz -C /usr/xxd/

# ls /usr/xxd/tools/other/

2.生成仓库数据文件

# createrepo /usr/xxd/tools/other/

3.修改配置文件

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

[myrpm]

name=my rpm

baseurl=file:///usr/xxd/tools/other

enable=1

gpgcheck=0

4.验证

# yum repolist

# rpm -q sl //查看是否安装

# rpm -ql sl //查看安装了什么

# oneko & //在后台运行

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

Day 02 DNS服务基础 特殊解析 DNS子域授权 缓存DNS

一、i权限：加上i权限，所有用户包括root都不能修改或删除文件

# lsattr 文档

//查看是否具有特殊权限

# chattr +i

# chattr -i

//添加/去除特殊权限

二、利用root进入虚拟机 密码123456

虚拟机A

1.配置eth0 永久静态ip地址：192.168.4.7/24

2.配置永久主机名：svr7.tedu.cn

# hostname svr7.tedu.cn

# echo 'svr7.tedu.cn' > /etc/hostname

虚拟机B

1.配置eth0 永久静态ip地址：192.168。4.207

2.配置永久主机名：pc207.tedu.cn

# hostname pc207.tedu.cn

# echo 'pc207.tedu.cn' > /etc/hostname

三、在真机上配置远程管理的别名，进行远程管理（/root/.bashrc）

alias goa='ssh -X root@192.168.4.7'

alias gob='ssh -X root@192.168.4.207'

四、为虚拟机A与虚拟机B 搭建Yum仓库

1.光盘搭本地yum

2.真机ftp/http搭网络yum

3.客户端配置yum

[rhel]

name=rhel7

baseurl=ftp://rhel7

enabled=1

gpgcheck=0

五、检查防火墙状态和SELinux状态

===================================================

六、DNS服务基础

1.DNS解析的作用

· 为什么要DNS系统 -- 域名比IP好记

· 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

2.DNS域名管理

· IANA,互联网数字分配机构

· CNNIC,中国互联网络信息中心

3.域名体系

　a)所有域名,都要以'.'来结尾

根： 根.

一级域名：.cn .kr .hk .tw .jp .us .com .net .org ...

二级域名：.com.cn .edu.cn .net.cn .gov.cn .mil.cn ...

三级域名：.dawai.com.cn .nb.com.cn .haxi.com.cn .dc.com.cn

完整主机名：www.dawai.com.cn ftp.dawai.com.cn tts.dawai.com.cn

· FQDN完全合格主机名

- 站点名.域名后缀

- 站点名. .. .. .二级域.一级域

· 域名代理/注册/购买服务商

- 新网

- 万网

- 中国互联

4.BIND域名服务

· BIND

- 伯克利 Internet 域名服务

· 主配置文件:/etc/named.conf 　　//设置本机负责解析的域名

· 地址库配置文件:/var/named/ 　　//所有的完整的主机名与ip对应关系

－系统服务：named

－默认端口：TCP/UDP 53

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

一．搭建基本的DNS服务

· 虚拟机A:

1.安装软件包

yum -y install bind-chroot bind

//域名服务包

bind

//提供虚拟根支持(牢笼政策)

bind-chroot

2.修改主配置文件

· 养成备份的习惯

# cp /etc/named.conf /root/named.bak

· 配置文件字段

//以下语句未定义时,默认为any

listen-on port 53 { any; };

allow-query { any; };

options {

directory "/var/named"; //指定地址库文件存放路径

};

zone "tedu.cn" IN { //指定本DNS服务器负责解析的域名

type master; //指定本机为权威DNS服务器(主DNS)

file "tedu.cn.zone"; //指定地址库文件的名字

};

3.建立地址库文件/var/named/tedu.cn.zone

//保证named用户对地址库文件有读权限

# cp -p /var/named/named.localhost /var/named/tedu.cn.zone

# ll /var/named/tedu.cn.zone

-rw-r-----. 1 root named 152 6月 21 2007 /var/named/tedu.cn.zone

# vim /var/named/tedu.cn.zone

tedu.cn. NS svr7

svr7 A 192.168.4.7

www A 1.2.3.4

ftp A 5.6.7.8

NS 字段前:本机解析的域名 字段后:本机域名

A字段前:域名 字段后:ip

AAAA -- ipv6

· 第一个A字段:本机 字段前:本机域名 字段后:本机ip

· 从第二个A字段开始 解析其他主机(DNS主要功能)

tips:

· 地址库文件所有域名都要以'.'结尾(解析域名和本机域名)

· 没有以'.'结尾则默认补全以地址库

· A:正向解析记录

4.重启named服务

· 虚拟机B:验证

1.指定DNS服务器

# echo nameserver 192.168.4.7 > /etc/resolv.conf

# nslookup www.tedu.cn

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

二.多区域的DNS服务器

· 虚拟机A:

1.修改主配置文件

# vim /etc/named.conf

· 添加第二个解析域名

zone "qq.com" IN {

type master;

file "qq.com.zone";

};

2.添加地址库文件

# cp -p /var/named/named.localhost /var/named/qq.com.zone

# vim /var/named/qq.com.zone

qq.com. NS svr7

svr7 A 192.168.4.7

www A 1.2.3.4

ftp A 3.3.3.3

3.重启named服务

# systemctl restart named

· 虚拟机B:验证

# nslookup www.qq.com

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

三.特殊的解析记录

1.DNS负载均衡(解析结果的轮询)

　qq.com. NS svr7

　svr7 A 192.168.4.7

　www A 1.2.3.1

　www A 1.2.3.2

　www A 1.2.3.3

　www A 1.2.3.4

　ftp A 3.3.3.3

2.泛域名解析

qq.com. NS svr7

　svr7 A 192.168.4.7

　www A 1.2.3.1

　www A 1.2.3.2

www A 1.2.3.3

　www A 1.2.3.4

ftp A 3.3.3.3

\* A 11.22.33.44

qq.com. A 6.6.6.6

3.有规律的泛域名解析

　pc1.qq.com --> 192.168.10.1

　pc2.qq.com --> 192.168.10.2

　pc3.qq.com --> 192.168.10.3

　......

pc50.qq.com --> 192.168.10.50

DNS服务内置的变量：$GENERATE 可以产生连续的数字

$GENERATE 1-50 pc$ A 192.168.10.$

4.解析记录的别名

· 虚拟机A:

# vim /var/named/qq.com.zone

　　tts　　CNAME　　ftp

# tts解析结果与ftp相同

# systemctl restart named

· 虚拟机B:

# nslookup tts.qq.com

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

四.DNS子域授权

1.父域与子域

· 父域:www.tedu.cn

· 子域:www.bj.tedu.cn www.sz.tedu.cn www.zz.tedu.cn

虚拟机A负责解析父域tedu.cn

虚拟机B负责解析父域bj.tedu.cn

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

虚拟机B：

1.装包 bind-chroot bind

2.修改配置文件 vim /etc/named.vonf

3.建立地址库文件　cp -p /var/named/named.localhost /var/named/xxx

4.重启服务　systemctl restart named

虚拟机Ａ：

1.子域授权

# vim /var/named/tedu.cn.zone

tedu.cn. NS svr7

bj.tedu.cn. NS pc207

svr7 A 192.168.4.7

pc207 A 192.168.4.207

www A 1.2.3.4

ftp A 5.6.7.8

2.重启服务

虚拟机Ｂ:验证

# nslookup www.bj.tedu.cn

Server: 192.168.4.7

Address: 192.168.4.7#53

Non-authoritative answer: //非权威解答

Name: www.bj.tedu.cn

Address: 1.2.3.4

· 递归解析：首选DNS服务器，跑到相应其他DNS服务器上，询问最终将结果带回来的过

程（客户端与首选DNS服务器的交互）

options {

directory "/var/named";

recursion no

};

· 迭代解析：首选DNS服务器与其他DNS服务器之间的交互

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

五．主机映射文件/etc/hosts(为本机提供DNS域名解析)

# vim /etc/hosts

192.168.4.110 www.sina.com

# ping www.sina.com

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

总结：客户端DNS解析域名过程

　1.客户端查询/etc/hosts

　2.查询/etc/resolv.conf 查看DNS服务器

　3.DNS服务器进行反馈(递归查询与迭代查询)

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

六．缓存DNS,加速解析效率

1.真机挂载光盘

# mkdir /dvd

# mount /ISO/CentOS-7-x86\_64-DVD-1708.iso /dvd

2.真机书写配置文件

# cd /etc/yum.repos.d/

# mkdir repos

# mv \*repo repos/

# vim dvd.repo

[dvd]

name=CentOS 7.4

baseurl=file:///dvd

enabled=1

gpgcheck=0

# yum repolist

# yum -y install bind-chroot bind

3.搭建缓存DNS服务器

　a)改配置

　# vim /etc/named.conf

　options {

　 directory "/var/named";

　 forwarders { 176.19.0.26; }; //转发给达内DNS服务器

　};

　b)启服务

　# systemctl restart named

====================================================

七.管理运行级别（运行模式）

RHEL5、RHEL6 //切换运行级别的命令 init

0：关机

1：单用户模式（破解密码、修复系统）

2：字符模式（不支持网络）

3：字符模式（支持网络）

4：无定义

5：图形模式

6：重启

=====================================================

RHEL7 运行模式

multi-user.target 字符模式（支持网络）

graphical.target 图形模式

临时切换

# systemctl isolate multi-user.target

# systemctl isolate graphical.target

永久改变默认的运行模式

# systemctl get-default

//查看当前默认的运行模式

# systemctl set-default graphical.target

//修改默认的运行模式

=============================================

Day 03 Split分离解析 RAID磁盘阵列 进程管理 日志管理

一.回顾

1.常见的DNS服务器有哪些

主DNS 一级域名DNS 二级域名DNS 三级域名DNS 根域DNS

2.常见的DNS资源解析记录

NS A PTR(反向解析) CNAME

=============================================================

二.Split分离解析(视图解析)

1.概述

· 当受到客户机的DNS查询请求的时候

- 能够区分客户机的来源地址

- 能够为不同类别的客户机提供不同的解析结果(IP地址)

- 判断客户端来源,不同客户端解析同一个域名得到不同解析结果

- 意义:让客户端访问网络中最近的服务器

· 典型适用场景:CDN提供的内容分发服务

· 注意点:

1.客户端分类合理,所有客户端都要匹配分类

2.分类匹配由上到下,依次匹配,匹配即停止

1）同一个区域(sina.com)在多个视图内分别定义，其他地址库文件相互独立，从而实现解析结果的分离

2）定义view视图后，不允许在view以外出现zone配置

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

· 环境及需求

- 权威DNS:svr7.tedu.cn 192.168.4.7

- 负责区域:sina.com

- svr7记录分离解析:以www.sina.com为例

客户机来自 解析结果

192.168.4.207 --> 192.168.4.100

其他地址 --> 1.2.3.4

· 分离解析

-修改配置文件

view "nsd"{

match-clients { 192.168.4.207;.. ..; }; //客户端源IP地址

zone "sina.com" IN {

type master;

file "sina.com.zone";

};

};

view "other"{

match-clients { any; };

zone "sina.com" IN {

type master;

file "sina.com.other";

};

};

-建立地址库文件sina.com.zone和sina.com.other

-重启服务

· 对客户机ip抽象封装

· 多区域的分离解析

acl myip{ 192.168.4.8; 192.168.4.123; 192.168.4.66; 192.168.4.58 };

view "nsd" {

match-clients { myip };

zone

...

};

==================================================================

三.RAID磁盘阵列

· 廉价冗余磁盘阵列

- 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

- 阵列的价值：提升I/O效率、硬件级别的数据冗余

- 不同RAID级别的功能、特性各不相同

· RAID 0 条带模式

- 同一个文档分散存放在不同磁盘

- 并行写入以提高效率

- 至少需要2块磁盘

· RAID 1 镜像模式

- 一个文档复制成多份，分别写入不同磁盘

- 多份拷贝提高可靠性，效率无提升

· RAID 0+1/RAID 1+0

- 整合RAID 0、RAID 1的优势

- 并行存取提高效率、镜像写入提高可靠性

- 至少4块磁盘

· RAID 5，高性价比模式

- 相当于RAID0和RAID1的折中方案

- 需要至少3块磁盘的容量来存放校验数据

· RAID 6，高性价比/可靠模式

- 相当于扩展的RAID5阵列，提供2份独立校验方案

- 需要至少4块磁盘的容量来存放校验数据

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:由操作系统来管理阵列

- 主板 --> 磁盘 --> 操作系统 --> RAID软件 --> 数据

==============================================================

四.进程管理

程序:静态的代码,占用硬盘空间

进程:动态执行的代码,占用内存和CPU

父进程 子进程 --- 树型结构

进程的唯一编号:PID 越小越优先运行

1.查看进程树

· pstree

-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 - 所有进程的父进程

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

# pstree

# pstree lisi

# pstree -p lisi

# pstree -ap lisi

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

· ps—— Processes Snapshot

-格式:ps [选项] ...

· 常用命令选项

-aux:显示当前终端所有进程(a),当前用户在所有终端下的进程(x),以用户格式输出(u)

-elf:显示系统内所有进程(e),以长格式输出(l)信息,包括最完整的信息(f)

· ps aux 操作

- 列出正在运行的所有进程

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

- 列出正在运行的所有进程

- 可以看到进程的父进程PID

PPID：父进程的PID号

PRI/NI：进程优先级，数值越小优先级越高

· wc -l统计行数

统计进程数:

# ps -aux | wc -l

# ps -aux | cat -n | tail -1

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

2.进程动态排名

# top [-d 刷新秒数] [-U 用户名]

· top交互操作指令

- ?：查看帮助

- P、M：根据%CPU、%MEM降序排列

- T：根据进程消耗的TIME降序排列

- k：杀死指定的进程

- q：退出top程序

3.进程检索

# pgrep [选项] ... 查询条件

· 支持模糊查询

· 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

-t：检索指定终端的进程

-x：精确匹配完整的进程名

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

# pgrep - log

# pgrep -lU lisi -t pts/1

# pgrep -lU lisi -t pts/2

//pts/1 <--> 终端号

=====================================================

4.控制进程

· 进程的前后台调度

· 前台启动

- 输入正常命令行，运行期间占用当前终端

· 后台启动

- 在命令行末尾添加"&"符号，不占用当前终端

· 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

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

# sleep 1000 & //将进程放入后台运行

# sleep 800

ctrl+z //将进程暂停放入后台

# bg 2 //将后台编号为2的挂起进程恢复运行

# jobs //查看后台进程信息

[1]- 运行中 sleep 1000 &

[2]+ 运行中 sleep 800 &

# fg 2 //将后台编号为2的进程恢复前台运行

sleep 800

==============================================================

5.进程处决

· 干掉进程的不同方法

- Ctrl+c组合键，中断当前命令

- kill [-9] PID...、kill [-9] %后台任务编号

- killall [-9] 进程名...

- pkill 关键字... //模糊查询查杀(包含即死)

· 强制踢出一个用户

killall -9 -u 用户名 //杀死该用户开启的所有进程（用户由登陆变成未登陆）

书写防火墙或SELinux禁止用户登录

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

# sleep 900 &

[3] 5905

# jobs -l

[3]+ 5905 运行中 sleep 900 &

# kill 5905 //根据PID杀死进程

# jobs

[3]+ 已终止 sleep 900

# sleep 1000 &

[1] 5928

# killall -9 sleep //根据进程名,强制杀死进程

[1]+ 已杀死 sleep 1000

===============================================================

五、日志管理

1.日志的功能

· 系统和程序的日记本

- 记录系统,程序运行中发生的各种事件

- 通过查看日志,了解及排除故障

- 信息安全控制的"依据"

2.查看文本日志消息

· 通过分析工具

- tail、tailf、less、grep等文本浏览/检索命令

- awk、sed等格式化过滤工具

· 专用分析工具

- Webmin系统管理套件

- Webalizer、AWStats等日志统计套件

· 路径

/var/log/messages //记录内核消息,各种服务的公共消息

/var/log/dmesg //记录系统启动过程的各种消息

/var/log/cron //记录与cron计划任务相关的消息

/var/log/maillog //记录邮件收发相关的消息

/var/log/secure //记录与访问限制相关的安全消息

· 由系统服务rsyslog统一记录/管理

- 日志消息采集用文本格式

- 主要记录时间发生的时间,主机,进程,内容

3.用户登陆分析

· users、who、w命令

- 查看已登陆的用户信息，详细度不同

· last、lastb命令

- 查看最近登陆成功/失败的用户信息

4.日志消息的优先级

· Linux内核定义的事件紧急程度

- 分为0~7共8种优先级别

- 其数值越小，表示对应事件越紧急/重要

5.使用journalctl工具

· 提取由sysremd-journal服务收集的日志

- 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"

=================================================================

Day 04 系统&服务管理进阶 批量装机环境 配置PXE引导 kickstart自动应答

补充:虚拟机命令行管理

# virt-manager //启用图形虚拟系统管理器

# virsh list //列出正在运行的虚拟机

# virsh list --all //列出所有虚拟机

# clone-vm7 //产生一个新的虚拟机

# virsh list --all

# virsh start xxx //开启虚拟机

# virsh list --all

# virsh console xxx //直接控制虚拟机,不需要IP

退出终端管理模式:Ctrl + ]

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

一.网络批量装机环境

1.部署DHCP服务器

1)DHCP概述及原理

· Dynamic Host Configuration Protocol

-动态主机配置协议,由IETF(Internet 网络工程师人物小组)组织指定,用来简化主机地址分配管理

· 主要分配以下入网参数

- IP地址/子网掩码/广播地址

- 默认网关地址,DNS服务器地址

· DHCP工作分配原理(广播进行)

- 一个网络中只能有一个DHCP服务器

- 判断机制:先到先得

· 地址分配的四次会话

- DISCOVERY --> OFFER --> REQUIST --> ACK

客户机广播 路由反馈

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

一.配置dhcpd地址分配服务

虚拟机svr7:

1.安装dhcp服务包

# yum -y install dhcp

2.修改配置文件/etc/dhcp/dhcpd.conf

末行模式输入 :r /user/share/doc/dhcp\*/dhcpd.conf.example

subnet 192.168.4.0 netmask 255.255.255.0 { #分配的网段

range 192.168.4.100 192.168.4.200; #分配的IP地址范围

option domain-name-servers 192.168.4.7; #分配的DNS地址

option routers 192.168.4.254; #分配的网关地址

default-lease-time 600; #IP地址默认租期

max-lease-time 7200; #IP地址最大租期

}

3.重启服务

# systemctl restart dhcpd

2.网络装机概述

1)网络装机的优势

-规模化:同时装配多台主机

-自动化:装系统,配置等各种服务

-远程实现:不需要光盘,U盘等物理安装介质

2)什么是PXE网络

·PXE,Pre-boot eXecution Environment

-预启动执行环境,在操作系统之前运行

-可用于远程安装

·工作模式

-PXE client集成在网卡的启动芯片中

-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

3)PXE组件及过程分析

·需要哪些服务组件

-DHCP服务,分配IP,定位引导程序

-TFTP服务,提供引导程序下载 //TFTP:简单的文件传输协议,端口69

-HTTP服务(或FTP/NFS),提供yum安装源

·简略过程示意图

next-server

-------------->

客户端 DHCP <----------------- TFTP

｜ <-------------- 众多的引导文件 |

｜ IP地址 |

------<-----------------<----------------<-------

rpm包

图vim版

next-server

-------------------->

客户端 DHCP <----------------- TFTP

| <-------------------- 众多的引导文件 |

| IP地址 |

-------<-----------------------<-----------------------------<----------

rpm包

图txt版

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

1.修改配置文件

# vim /etc/dhcp/dhcpd.conf

subnet 192.168.4.0 netmask 255.255.255.0 {

range 192.168.4.100 192.168.4.200;

option domain-name-servers 192.168.4.7;

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

filename "pxelinux.0"; #指定网卡引导文件名称

}

2.重启dhcpd服务

# systemctl restart dhcpd

pxelinix.0:网卡引导文件(网络装机说明书)二进制文件

安装一个软件可以自动产生默认叫pxelinux.0

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

二.搭建TFTP服务

简单的文件传输协议 端口69

tftp默认共享数据路径 /var/lib/tftpboot

1.安装tftp-server包

# yum -y install tftp-server

2.启动tftp服务

# systemctl restart tftp

# ls /var/lib/tftpboot

3.部署pxelinux.0文件

# yum provides \*/pxelinux.0 //查询仓库中哪个软件包产生该文件

# yum -y install syslinux-4.05-13.el7.x86\_64

# rpm -ql syslinux | grep pxelinux.0 //查询软件包安装清单

# cp /usr/share/syslinux/pxelinux.0 /var/lib/tftpboot/

# ls /var/lib/tftpboot

4.部署菜单文件

pxelinux ---> /var/lib/tftpbppt/pxelinux.cfg/default

1)创建菜单目录

# mkdir /var/lib/tftpboot/pxelinux.cfg

# ls /var/lib/tftpboot/

pxelinux.0 pxelinux.cfg

2)从光盘拷贝菜单配置文件

# scp /var/ftp/rhel7/isolinux/isolinux.cfg root@192.168.4.7:/var/lib/tftpboot/pxelinux.cfg/default

· 为了修改方便,给管理员加上写权限

# chmod u+w /var/lib/tftpboot/pxelinux.cfg/default

# ll /var/lib/tftpboot/pxelinux.cfg/default

-rw-r--r--. 1 root root 3166 7月 24 14:25 /var/lib/tftpboot/pxelinux.cfg/default

5.部署引导文件(启动内核)

vesamenu.c32 图形模块,呈现背景图片与颜色等

vmlinuz 启动内核

initrd.img 驱动程序

splash.png 黑色背景图片

# scp /var/ftp/rhel7/isolinux/vesamenu.c32 /var/ftp/rhel7/isolinux/vmlinuz /var/ftp/rhel7/isolinux/initrd.img /var/ftp/rhel7/isolinux/splash.png root@192.168.4.7:/var/lib/tftpboot/

# ls /var/lib/tftpboot/

initrd.img pxelinux.0 pxelinux.cfg splash.png vesamenu.c32 vmlinuz

6.修改菜单文件内容

# vim /var/lib/tftpboot/pxelinux.cfg/default

1 default vesamenu.c32 #默认加载图形模块

2 timeout 600 #默认读秒

11 menu title NSD1806 PXE SERVER #修改标题

61 label linux

62 menu label ^Install RHEL7.4

menu default #读秒结束默认选择

63 kernel vmlinuz

64 append initrd=initrd.img

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

初步测试:

1.新建虚拟机 选择PXE网络引导安装

2.网络类型选择private1

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

三.构建httpd服务,利用web共享众多的rpm包

1.安装软件包

# yum -y install httpd

# systemctl restart httpd

2.建立共享路径

# mkdir /var/www/html/rhel7

# vim /etc/exports //真机通过nfs共享给svr7

/var/ftp/rhel7 192.168.4.0/24(ro)

# mount 192.168.4.254:/var/ftp/rhel7 /var/www/html/rhel7/

# ls /var/www/html/rhel7

3.测试

# firefox 192.168.4.7/rhel7

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

四.部署无人值守安装,生成应答文件

1.图形生成应答文件工具system-config-kickstart

# yum -y install system-config-kickstart

2.运行工具system-config-kickstart

· 检查"软件包选择"是否可以选择

与本机Yum仓库标识有关: [development]

3.查看应答文件

# ls /root/ks.cfg

# vim /root/ks.cfg

4.利用web共享ks应答文件,传递给客户端

# cp /root/ks.cfg /var/www/html

# ls /var/www/html

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

五.修改菜单文件,指定ks应答文件

# vim /var/lib/tftpboot/pxelinux.cfg/default

label linux

menu label ^Install RHEL7.4

menu default

kernel vmlinuz

append initrd=initrd.img ks=http://192.168.4.7/ks.cfg

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

总结:

DHCP --> IP地址,next-server,filename

tftp --> pxelinux.0

pxelinux.0 --> /var/lib/tftpboot/pxelinux.cfg/default

default --> vesamenu.c32\splash.png\vmlinuz\inittrd.img

ks="http://192.168.4.7/rhel7"

================================================

Day 05 rsync同步操作 inotify实时同步 Cobbler装机平台

环境准备:

·检查Yum是否可用

虚拟机A:

# yum clean all //清空Yum缓存

# yum repolist

虚拟机B:同上

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

一.rsync同步操作

1.命令用法

- rsync [选项...] 源目录 目标目录

· 同步与复制的差异

- 复制:完全拷贝源到目标

- 同步:增量拷贝,只传输变化过的数据

· 同步控制

- 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

-v:显示详细操作信息

-z:传输过程中启用压缩/解压

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

· 虚拟机svr7本机同步

# mkdir /nsd /test

# cp /etc/passwd /nsd/

# touch /nsd/1.txt

# rsync -avz /nsd/ /test/

sending incremental file list

./

1.txt

passwd

sent 995 bytes received 53 bytes 2096.00 bytes/sec

total size is 2201 speedup is 2.10

# touch /nsd/2.txt

# rsync -avz /nsd/ /test/

sending incremental file list

./

2.txt

sent 112 bytes received 34 bytes 292.00 bytes/sec

total size is 2201 speedup is 15.08

# rsync -avz --delete /nsd/ /test/ //同步并删除目标多余文档

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

· rsync+SSH远程同步

与远程的SSH目录保持同步,格式类似scp

- 下行:rsync [...] user@host:远程目录 本地目录

- 上行:rsync [...] 本地目录 user@host:远程目录

虚拟机A

# rsync -avz --delete /opt/ root@192.168.4.207:/opt/

虚拟机B

# ls /opt

虚拟机A

# touch /opt/{1..5}.txt

# rsync -avz --delete /opt/ root@192.168.4.207:/opt/

虚拟机B

# ls /opt

二.inotify实时同步

1.实时远程同步

· 无密码验证

a)生成公钥和私钥

# ssh-keygen //一路回车

/root/.ssh/known\_hosts //记录ssh远程客户端标识文件

·公钥

·私钥

b)传递公钥到指定机器

# ssh-copy-id root@192.168.4.207

c)测试免验证登录

# rsync -avz --delete /opt/ root@192.168.4.207:/opt/

2.目录内容监控

a)安装inotify-tools控制工具可调用此机制实现监控

· 标准的源码,编译安装

rpm包:yum,rpm -ivh

源码包:

· 安装流程

源码包 --> 开发工具(gcc与make) --> 可执行的程序 --> 运行安装

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

虚拟机svr7:

1.安装gcc与make

# yum -y install gcc make

2.tar进行解包

# tar -xf /root/xxd/tools/inotify-tools-3.13.tar.gz -C /

# ls /

# cd /inotify-tools-3.13

# ls

3.'./configure'配置,指定安装目录/功能模块等选项

检测系统是否安装gcc

--prefix=路径 #指定安装目录

# cd /inotify-tools-3.13

# ./configure

4.make编译,生成可执行的二进制程序文件

# cd /inotify-tools-3.13

# make

5.make install 安装,将编译好的文件复制到安装目录

# make install

# which inotifywait //查看是否具备该程序

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

· 主要优点

- 获得软件的最新版,及时修复bug

- 软件功能可按需选择/定制,有更多软件可供选择

- 源码包适用各种平台

====================================================================

三.Cobbler装机平台

1.环境准备

·安装CentOS 7虚拟机:

1.图形方式进行安装

2.内存2G

3.磁盘大小:至少50G以上

4.网络类型private1

5.分区选择自动分区

6.软件包选择"带GUI的服务器"

7.设置root密码 创建普通用户lisi

8.将CentOS放入光驱设备,搭建本地Yum仓库

9.配置IP地址:192.168.4.168/24

10.配置主机名:Cobbler.tedu.cn

11.设置防火墙默认区域为trusted

12.当前及永久设置SELinux状态为Disabled

13.利用scp将真机的Cobbler.zip包传到虚拟机192.168.4.168的/root/目录下

2.基本用法

·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

-e,指定监视的midify,move,create,delete,attrib等事件类别

3.书写Shell脚本

循环:for循环:适合次数固定的事件

while循环:适合死循环的事件

while [条件]

do

循环执行语句

done

# vim /root/rsync.sh

#!/bin/bash

while inotifywait -rqq /opt/

do

rsync -az --delete /opt/ root@192.168.4.207:/opt/

done &

# chmod +x /root/rsync.sh

4.搭建Cobbler装机平台

Cobbler概述软件,管理dhcp,tftp,web服务

自由地导入镜像与ks应答文件

1)解压Cobbler.zip包

# unzip /root/Cobbler.zip -d /

# cd /Cobbler/

# ls

# unzip /Cobbler/cobbler.zip -d /opt

# ls /opt/cobbler

5.Cobbler简介

· 基本概念

-Cobbler是一款快速的网络系统部署工具

-集中管理所需服务,如DHCP,DNS,TFTP,Web

-Cobbler内部集成了一个镜像版本仓库

-Cobbler内部集成了一个ks应答文件仓库

-Cobbler还提供了包括yum源管理,Web界面管理,API接口,电源管理功能

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

一.安装cobbler主程序,工具包等

# yum -y install /opt/cobbler/\*.rpm

1.安装软件 cobbler cobbler-web dhcp tftp-server pykickstart httpd

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

# vim /etc/cobbler/settings

next\_server: 192.168.4.168 #设置下一个服务器还为本机

server: 192.168.4.168 #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:匹配即停止

1.硬盘启动 2.光驱设备 3.U盘 4.网络引导

1.网络引导 2.光驱设备 3.U盘 4.硬盘启动

3.配置cobbler的dhcp

# vim /etc/cobbler/dhcp.template

末行模式输入,将192.168.1默认网段改为192.168.4网段

:%s/192.168.1/192.168.4/g

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

# tar -tf /Cobbler/cobbler\_boot.tar.gz #查看包里内容

# tar -xPf /Cobbler/cobbler\_boot.tar.gz #绝对路径释放

# ls /var/lib/cobbler/loaders/

5.启动相关服务

# systemctl restart cobblerd

# systemctl enable cobblerd

# systemctl restart httpd

# systemctl enable httpd

# systemctl restart tftp

# systemctl enable tftp

# systemctl restart rsyncd #同步服务

# systemctl enable rsyncd

6.同步刷新cobbler配置

# cobbler sync

...

\*\*\* TASK COMPLETE \*\*\*

# firefox https://192.168.4.168/cobbler\_web

用户名:cobbler

密码:cobbler

7.Cobbler应用

cobbler import --path=挂载点 --name=导入系统命名(随意起)导入镜像安装

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

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/

[root@cobbler ~]# cobbler list

修改kickstart文件：

[root@cobbler ~]# cobbler profile edit --name=CentOS7.4-A --kickstart=/var/lib/cobbler/kickstarts/自定义.cfg

[root@cobbler ~]# cobbler profile report

[root@cobbler ~]# cobbler sync #同步配置

===================================================

Day 06 日志管理 systemctl服务管理 PATH变量应用

一.日志管理

1.日志的功能

· 系统和程序的日记本

- 记录系统,程序运行中发生的各种事件

- 通过查看日志,了解及排除故障

- 信息安全控制的"依据"

2.查看文本日志消息

· 通过分析工具

- tail、tailf(实时跟踪日志消息)、less、grep等文本浏览/检索命令

- awk、sed等格式化过滤工具

· 专用分析工具

- Webmin系统管理套件

- Webalizer、AWStats等日志统计套件

· 路径

/var/log/messages //记录内核消息,各种服务的公共消息

/var/log/dmesg //记录系统启动过程的各种消息

/var/log/cron //记录与cron计划任务相关的消息

/var/log/maillog //记录邮件收发相关的消息

/var/log/secure //记录与访问限制相关的安全消息

· 由系统服务rsyslog统一记录/管理

- 日志消息采集用文本格式

- 主要记录时间发生的时间,主机,进程,内容

3.用户登陆分析

· users、who、w命令

- 查看已登陆的用户信息，详细度不同

· last、lastb命令

- 查看最近登陆成功/失败的用户信息

# users

# who

# w

# last #查看登陆成功的用户信息

# lasttb #查看登录失败的用户信息

4.日志消息的优先级

· Linux内核定义的事件紧急程度

- 分为0~7共8种优先级别

- 其数值越小，表示对应事件越紧急/重要

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

5.使用journalctl工具

· 提取由sysremd-journal服务收集的日志

- 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"

# systemctl restart httpd

# journalctl -u httpd

=============================================================

二.systemctl服务管理

· 一个更高效的系统&服务管理器

- 开机服务并行启动,各系统服务间的精确依赖

- 配置目录: /etc/systemd/system/

- 服务目录: /lib/systemd/system/

- 主要管理工具:systemctl

· 服务的管理

systemvtl start 服务名 #启动服务

systemvtl stop 服务名 #停止服务

systemvtl restart 服务名 #重启服务

systemvtl ststus 服务名 #查看服务当前状态

systemvtl enable 服务名 #开机自启

systemvtl disable 服务名 #开机不自启

systemvtl is-enabled 服务名 #查看服务自启状态

· 运行模式的管理(运行级别)

RHEL5,RHEL6:

0：关机

1：单用户模式（破解密码、修复系统）

2：字符模式（不支持网络）

3：字符模式（支持网络）

4：无定义

5：图形模式

6：重启

0

开机启动 v

6

- 切换运行级别的命令 init

# init 3 切换到字符界面

# init 5 切换到图形界面

RHEL7: 运行模式

systemctl -t --all

multi-user.target :字符模式

graphical.target:图形模式

· 修改当前运行模式(重启失效)

systemctl isolate multi-user.target

systemctl isolate grphical.target

· 修改默认运行模式(重启有效)

systemctl set-default graphical.target

systemctl set-default multi-user.target

· unit配置单元

· 不同的unit决定了一组相关的启动任务

- service:后台独立服务

- socket:套接字,类似于xinetd管理的临时服务

- target:一套配置单元的组合,类似于传统"运行级别"

- device:对应udev规则标记的某个设备

- mount,automount:挂载点,触发挂载点

=========================================================

三.Cobbler装机平台(见05.txt)

=========================================================

四.PATH变量应用:提供命令搜寻路径

系统环境变量:由系统定义完成且赋值完成

# echo $PATH

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

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

[root@svr7 ~]# vim /usr/local/bin/hello.sh

[root@svr7 ~]# chmod +x /usr/local/bin/hello.sh

[root@svr7 ~]# hello.sh

hello world!

=============================================

Day 01 计算机网络概述 网络设备及拓扑 OSI参考模型 交换机命令基础

一.什么是计算机网络

1.什么是计算机网络

- 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,应用程序通过通信线路相连

- 实现资源共享,信息传递

2.计算机网络的功能

· 数据通信

· 资源共享

· 增加可靠性

· 提高系统处理能力

3.计算机网络发展阶段

· 60年代

-分组交换

· 70-80年代

-TCP/IP

· 90年代后

-Web技术

4.网络标准

· 标准化组织

-ISO(国际标准化组织)

-ANSI(美国国家标准化局)

-ITU-T(国际电信联盟-电信标准部)

-IEEE(电气和电子工程师学会)

5.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

二.网络设备及拓扑

1.网络设备生产厂商

· 网络设备生产厂商

- Cisco(思科)

- 华为

2.网络拓扑结构

· 线缆连接计算机和网络设备的布局

- 点对点

- 星型及扩展的星型

- 网状

1)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

2)星型拓扑

-优点

易于实现

易于网络扩展

易于故障排查

-缺点

中心节点压力大

组网成本较高

3)网状拓扑

-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

三.OSI参考模型

1.网络分层

-数据以电子信号的形式穿越介质到达正确的计算机,然后转换成最初的形式,以便接收者能够阅读

-为了降低网络设计的复杂性,将协议进行了分层设计

2.OSI的七层框架

应用层 网络服务与最终用户的一个借口

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

3.TCP/IP五层模型

应用层 HTTP FTP TFTP SMTP SNMP DNS

传输层 TCP UDP

网络层 ICMP IGMP IP ARP(IP->MAC的解析) RARP(反解析)

数据链路层 由底层网络定义的协议

物理层

4.协议

· 什么是协议

- 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"

- 描述网络通信中语言规范的规则称为协议

5.协议数据单元

物理层 网卡 比特流 电信号

数据链路层 交换机 数据帧 MAC头部,IP头部,TCP头部,上层数据

网络层 路由器 数据包 IP头部,TCP头部,上层数据

传输层 防火墙 数据段 TCP头部,上层数据

应用层 计算机 上层数据

四.物理层

1.以太网接口

· RJ - 45

-RJ使描述公用电信网络的接口,常用的有RJ-11和RJ-45(水晶头)

· 光纤接口

-用以稳定地但并不是永久地连接两根或多根光纤的无源组件

FC圆形带螺纹光纤接头

ST卡接式圆形光纤接头

SC方形光纤接头

LC窄体方形光纤接头

MT-RJ收发一体的方形光纤接头

· 双绞线

-双绞线TP是目前使用最广,价格相对便宜的一种传输介质

-由两根绝缘铜导线相互缠绕组成,以减少对邻近线的电气干扰

-由若干对双绞线构成的电缆被成为双绞线电缆

· 双绞线的标准

-非屏蔽双绞线UTP和屏蔽双绞线STP

-速率

cat5 100Mbps

cat5e 100Mbps

cat6 1000Mbps(1Gbps)

cat7 10000Mbps(10Gbps)

-网线长度一般不能超过150m

· 线缆的连接:

-T568A:白绿 绿 白橙 蓝 白蓝 橙 白棕 棕

-T568B:白橙 橙 白绿 蓝 白蓝 绿 白棕 棕

-直通线和交叉线

· 同类型设备用交叉,不同类型用直通

· 交换机之间用直通,路由器当主机处理

-全反线:连接设备console口

2.网络接口卡(网卡)

- 连接计算机和网络硬件

- 有一个唯一的网络节点地址

- 按照速率可分为10/100M,100/1000M自适应网卡

- 按照扩展类型可分为USB网卡,PCI网卡

- 按照提供的线缆接口类型可分为RJ-45接口网卡,光纤网卡等

· 中继器

-放大信号

-延长网络传输距离

五.Packet Tracer软件

六.交换机命令行模式

1.用户模式

Switch>用户模式

2.特权模式(一般用于查看配置信息)

Switch>enable

Switch#特权模式

3.全局配置模式(所做的配置对整个设备生效)

Switch#configure terminal

Switch(config)#全局配置模式

4.接口模式

Switch(config)#interface fastEhernet 0/1

Switch(config-if)#接口模式

5.配置主机名

>enable

#conf t

()#hostname S1

()#exit

#show running-config #查看配置信息

6.配置enable明文口令

全局配置模式：enable password 123

7.保存交换机的配置

#copy running-config startup-config

或

#write

8.恢复设备出厂默认值

特权：erase startup-config

9.重启：reload

七.设备配置的准备工作

1.空闲一段时间后，重回初始界面的问题

switch(config)#line con 0

switch(config-line)#exec-timeout 0 0

2.禁用DNS查询

switch(config)#no ip domain-lookup

3.配置输出日志同步

Switch(config)#line console 0

Switch(config-line)#logging synchronous

=========================================================

Day 02 数据链路层

1.以太网

1)以太网MAC地址(48位,用16进制表示)

- 用来识别一个以太网上的某个单独的设备或一组设备

24比特供应商标识 24比特供应商对网卡的唯一编号

第八位对于目的地址:

0-物理地址(单播地址)

1-逻辑地址(组播地址)

2)单播 1对1

组播 1对多

广播 1对所有

3)帧格式

目的地址 6字节

源地址 6字节

类型/长度 2字节

数据 46-1500字节

帧校验序列 4字节

2.以太交换机

1)什么是交换机

· 交换机是用来连接局域网的主要设备

-交换机能够根据以太网帧中目标地址智能转发数据,工作在数据链路层

2)交换机的转发原理

- 初始状态

- MAC地址学习

- 广播未知数据帧

- 接受方回应

·学习 ·广播 ·转发 ·更新

3)查看MAC地址表

#show mac-address-table

3.广播域

1)VLAN

·虚拟局域网是物理设备上连接的不受物理位置限制的用户的一个逻辑组

·为什么引入VLAN

-交换机的所有端口默认属于同一个广播域

-随着接入设备的增多,网络中广播增多,降低了网络效率

-为了分割广播域,引入了VLAN

·VLAN的作用

-广播控制

-安全性

-带宽利用

-延迟

·基于端口划分的静态VLAN

-静态VLAN的配置

-创建VLAN

-将端口加入到相应VLAN

-验证

·在全局模式下创建VLAN

()#vlan vlan-id

()#vlan vlan-name

·查看VLAN配置

#show vlan brief

#show vlan id $id

·删除已创建的VLAN

()#no vlan 5

·将端口加入VLAN

()#interface f0/2

()#switchport access vlan 2

·批量修改接口属性

()#interface range f0/1-2

()#switchport access vlan 2

4.Trunk

1)用于多广播域多交换机之间作为中继链路

·配置Trunk

()#interface 0/24

()#switchport mod trunk

Administrative Mode 动态自动(默认)

Operation Mode 正在使用的模式

-这样的链路中两台交换机之间的广播域vlan要对应

·ISL帧格式:前ISL头26字节,后CRC4字节

·ISL和802.1Q标记之间的异同

-相同点

·都是显式标记,即帧被显式标记了VLAN的信息

-不同点

·IEEE 802.1Q是公有的标记方式,ISL是Cisco私有的

·ISL采用外部标记的方法,802.1Q采用内部标记的方法

·ISL标记的长度为30字节,802.1Q标记长度为4字节

5.配置以太通道

1)概述:

·也称为以太端口捆绑,端口聚集或以太链路聚集

·以太通道为交换机提供了端口捆绑技术,允许两个交换机之间通过两个或多个端口并行连接,同时传输数据,以提供更高的带宽

2)配置以太网通道

()#interface range fastEthernet0/10-11

()#channel-group 1 mode on

Creating a port-channel interface Port-channel 1

3)查看以太网通道的配置

#show etherchannel summary

=========================================================

Day 03 网络层

1.网络层的功能

· 定义了基于IP协议的逻辑地址

· 连接不同的媒介类型

· 选择数据通过网络的最佳路径

2.路由概述

· 将数据包从一个网络发送到另一个网络

- 需要依靠路由器完成

- 路由器只关心网络的状态,决定最佳路径

3.路由器工作:

-识别数据包的目标IP地址

-识别数据包的源IP地址(用于策略路由)

-在路由表中发现可能的路径

-在路由表中选择到达目标最好的路径

-维护和检查路由信息

· 根据路由表选择最佳路径

-每个路由器都维护一张路由表,这是路由器转发数据包的关键

-每条路由表记录指明了:到达某个子网或主机应从路由器的那个物理端口发送,通过此端口可达到该路径的下一个路由器的地址

4.如何获得路由表

· 静态,缺省路由

-由管理员在路由器上手动指定

-适合分支机构,家居办公等小型网络

· 动态路由

-根据网络拓扑或流量变化,由路由器通过路由协议自动设置

-适合ISP服务商,广域网,园区网等大型网络

5.静态路由

· 主要特点

-由管理员手动配置,为单向条目

-通信双方的边缘路由器都需要指定,否则会导致数据包有去无回

· 配置静态路由

()#in g0/0

//设置接口ip

()#ip address 192.168.1.254 255.255.255.0

//启用接口

()#no shutdown

· 查看路由表

#show ip route

C直连路由

S静态路由

S\*默认路由

· 删除接口ip

()#no ip address

· 配置静态路由

()#ip route 目标网络ID 子网掩码 下一跳

6.缺省路由

· 什么是缺省路由

-缺省路由是一种特殊的静态路由,对于末梢网络的主机来说,也被称为"默认网关",一般缺省路由也只应用在末梢网络,中继网络使用缺省路由会导致丢包

-缺省路由的.目标网络为0.0.0.0 0.0.0.0 ,可匹配任何目标地址

-只有当从路由表中找不到任何明确匹配的路由条目时,才会使用缺省路由

7.三层交换机

·概述:

· 使用三层交换技术实现VLAN间通信

· 三层交换=二层交换+三层转发

·虚接口

·在三层交换机上配置的VLAN接口为虚接口

·使用SVI(交换虚拟端口)实现VLAN间路由

· 启用三层交换机的路由功能

()#ip routing

· 配置虚接口的IP

()#interface vlan $id

()#ip address $ip $netmask

()#no shutdown

· 配置路由接口

()#no switchport

· 在三层交换机上配置Trunk并指定接口封装为802.1Q

()#interface f0/24

()#switchport trunk encapsulation dot1q

()#switchport mode trunk

8.动态路由

· 动态路由

-基于某种路由协议实现

· 动态路由的特点

-减少了管理任务

-占用了网络带宽

·OSPF

-邻居列表

-链路状态数据库

-路由表

·OSPF区域

-为了适应大型网络,OSPF在网络内部划分多个区域

-每个OSPF路由器只维护所在区域的完整链路状态信息

·区域ID

-区域ID可以表示成一个IP或十进制数字

·配置OSPF

启动OSPF路由进程

()#router ospf $id

指定OSPF协议运行的网络地址和所在区域

()#network $ip 反掩码 area $areaId

重启OSPF

#clear ip ospf process

===================================================

Day 04 传输层

一.传输层作用

· 网络层提供点到点的连接

· 传输层提供端到端的连接

二.传输层的协议

· TCP

-传输控制协议

-可靠的,面向连接的协议

-传输效率低

· UDP

-用户数据报协议

-不可靠的,无连接的服务

-传输效率高

三.TCP协议

1.TCP的封装格式

· SYN 请求标记

· ACK 确定标记

2.TCP连接过程:三次握手

客户端:SYN seq=100,ctl=SYN

服务端:SYN,ACK seq=300,ack=101,ctl=SYN,ACK

客户端:ACK seq=101,ack=301,ctl=ACK

3.TCP的四次断开

客户端:发送 FIN,请求断开连接(FIN=1,ACK=1)

服务端:发送 ACK(ACK=1)

服务端:发送 FIN,请求断开连接(FIN=1,ACK=1)

客户端:发送ACK(ACK=1)

4.TCP的应用

端口 协议 说　　明

21 FTP FTP服务器所开放的控制端口

23 TELNET 用于远程登录，可以远程控制管理目标计算机

25 SMTP SMTP服务器开放的端口，用于发送邮件

80 HTTP 超文本传输协议

53 DNS 域名服务，当用户输入网站的名称后，由DNS负责将它解析成IP地址，这个过程中用到的端口号是53

四.UDP协议

1.UDP的流控和差错控制

· UDP缺乏可靠机制

· UDP只有校验和来提供差错控制

-需要上层协议来提供差错控制:如TFTP协议

2.UDP的应用

端口 协议 说明

69 TFTP 简单文件传输协议

123 NTP 网络时间协议

53 DNS 域名服务

五.访问控制列表概述

1.访问控制列表(ACL)

- 读取第三层,第四层头部信息

- 根据预先定义好的规则对数据进行过滤

2.访问控制列表的类型

· 标准访问控制列表

- 基于源IP地址过滤数据包

- 标准访问控制列表的表号是1~99

· 扩展访问控制列表

- 基于源IP地址,目的IP地址,指定协议,端口来过滤数据包

-扩展访问控制列表的访问控制列表号是100~199

3.标准访问控制列表的配置

· 创建ACL

()#access-list $number { permit | deny } source [ source-wildcard ]

· 实例

//允许192.168.1.0/24网段的流量通过

Router(config)# access-list 1 permit 192.168.1.0 0.0.0.255

//允许192.168.2.2主机的流量通过

Router(config)# access-list 1 permit 192.168.2.2 0.0.0.0

· 隐含的拒绝语句(拒绝所有人)

Router(config)# access-list 1 deny 0.0.0.0 255.255.255.255

· 关键字

-host

-any

· 将ACL应用于接口

Router(config-if)# ip access-group $number {in|out}

· 删除ACL

Router(config-if)# no ip access-group $number {in |out}

4.扩展访问控制列表的配置

· 创建ACL

Router(config)# access-list $number { permit | deny } protocol { source source-wildcard destination destination-wildcard } [ operator operan ]

六.NAT概述

1.NAT作用

· NAT

-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外部网络上

· NAT的优点

-节省公有合法IP地址

-处理地址重叠

-安全性

· NAT的缺点

-延迟增大

-配置和维护的复杂性

2.NAT实现方式

· 静态转换

· 端口多路复用

3.静态转换

· IP地址的对应关系是一对一的,而且是不变的,借助静态转换,能实现外部网络对内部网络中某些特设定服务器的转换

· 静态NAT配置步骤

-接口IP地址配置

-决定需要转换的主机地址

-决定采用什么公有地址

-在内部和外部端口上启用NAT

()#ip nat inside source static $localIP $globalIP

· 将内网地址192.168.1.1静态转换为合法的外部地址100.0.0.2以便访问外网

· 设置外部端口的IP地址

Router(config)#interface g0/1

Router(config-if)#ip address 100.0.0.1 255.0.0.0

Router(config-if)#no shut

· 设置内部端口的IP地址

Router(config)#interface g0/0

Router(config-if)#ip address 192.168.1.254 255.255.255.0

Router(config-if)#no shut

· 建立静态地址转换

Router(config)#ip nat inside source static 192.168.1.1 100.0.0.2

· 在内部和外部端口上启用NAT

Router(config)#interface g0/1

Router(config-if)#ip nat outside

Router(config)#interface g0/0

4.NAT端口映射

· 建立NAT端口映射

· 配置实例(只发布服务器的80端口)

Router(config)#ip nat inside source static tcp 192.168.1.6 80 61.159.62.133 80

Router(config-if)#ip nat inside

5.端口多路复用(PAT)

· ACL定义内部IP地址

Router(config)#access-list 1 permit 192.168.1.0 0.0.0.255

· 设置复用动态IP地址转换

//配置端口多路复用使企业内网192.168.1.0/24复用g0/1端口的ip,实现外部网络的访问

Router(config)#ip nat inside source list 1 interface g 0/1 overload

· PAT只适用于内网访问外网,无法架设服务

===================================================================

Day 05 应用层

一.STP生成树算法

1.广播风暴的产生

• 交换机工作原理

– 根据MAC地址表转发数据帧,如果地址未知,则广播

– 如果交换机接收到广播帧也会向所有端口发送

• 当网络中存在物理环路,会产生广播风暴

2.STP概述

• STP简介

– STP - Spanning Tree Protocol(生成树协议)

逻辑上断开环路,防止广播风暴的产生

当线路故障,阻塞接口被激活,恢复通信,起备份线路的作用

3.选择根网桥

• 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

网桥优先级 2字节

网桥的MAC地址 6字节 取值范围:0 ~ 65535

缺省值:32768

二.STP配置

1.PVST+的配置命令

• 启用生成树命令

Switch(config)#spanning-tree vlan $vlanList

• 指定根网桥

Switch(config)#spanning-tree vlan $vlanList priority Bridge-priority

Switch(config)#spanning-tree vlan $vlanList root { primary | secondary }

• 查看生成树的配置

Switch#show spanning-tree

• 查看某个VLAN的生成树详细信息

Switch#show spanning-tree vlan vlan-id

N.HSRP概述

n.HSRP的相关概念

1.热备份路由选择协议

-HSRP(Hot Standby Routing Protocol)

-是Cisco私有协议(VRRP公有协议了解一下)

2.HSRP组成员

·两台物理路由器(一台活跃一台备份)

-活跃路由器

-备份路由器

-虚拟路由器

-其他路由器

3.HSRP原理

一台活跃,一台热备,虚拟桥接,来回切换

4.HSRP配置

• 配置为HSRP的成员

Switch(config-if)#standby group-number ip $vlanIP

• 配置HSRP的优先级

Switch(config-if)#standby group-number priority priority-value

• 查看HSRP摘要信息

Switch(config-if)#show standby brief

• MS1配置

MS1(config)#interface vlan 1

MS1(config-if)#ip address 192.168.1.252 255.255.255.0

MS1(config-if)#standby 1 ip 192.168.1.254

MS1(config-if)#standby 1 priority 105

• HSRP端口跟踪

– 跟踪端口不可用时,HSRP优先级降低

– 活跃路由器可以根据线路情况自动调整

• HSRP占先权

– 优先级高的路由器重新获得转发权,恢复成为活跃路由器

– HSRP占先权配置

()#standby group-number perrmpt

===================================================

Day06综合项目完整实验步骤

二层交换机

分别创建VLAN10、20、30、40

sw1将f0/5接口加入vlan10

Switch(config)#interface fastEthernet 0/5

Switch(config-if)#switchport access vlan 10

sw2将f0/5接口加入vlan20

Switch(config)#interface fastEthernet 0/5

Switch(config-if)#switchport access vlan 20

sw3将f0/5接口加入vlan30

Switch(config)#interface fastEthernet 0/5

Switch(config-if)#switchport access vlan 30

sw4将f0/5接口加入vlan40

Switch(config)#interface fastEthernet 0/5

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

Switch(config)#interface range f0/1-2

Switch(config-if-range)#channel-group 1 mode on

Switch(config)#interface range f0/3-4

Switch(config-if-range)#channel-group 1 mode on

查看以太通道汇总信息

Switch#show etherchannel summary

依次进入所有二层交换机的以太通道接口，配置中继链路

Switch(config)#interface port-channel 1

Switch(config-if)#switchport mode trunk

Switch(config)#interface port-channel 2

Switch(config-if)#switchport mode trunk

==========================================================

三层交换机

每台设备分别创建VLAN10、20、30、40

1-2口捆绑为通道1

3-4口捆绑为通道2

5-6口捆绑为通道3

7-8口捆绑为通道4

9-10口捆绑为通道5

依次进入三层交换机的4个通道接口，配置中继链路（两台三层交换机配置相同）

Switch(config)#interface port-channel 1

Switch(config-if)# switchport trunk encapsulation dot1q

Switch(config-if)#switchport mode trunk

Switch(config)#interface port-channel 2

Switch(config-if)# switchport trunk encapsulation dot1q

Switch(config-if)#switchport mode trunk

Switch(config)#interface port-channel 3

Switch(config-if)# switchport trunk encapsulation dot1q

Switch(config-if)#switchport mode trunk

Switch(config)#interface port-channel 4

Switch(config-if)# switchport trunk encapsulation dot1q

Switch(config-if)#switchport mode trunk

Switch(config)#interface port-channel 5

Switch(config-if)# switchport trunk encapsulation dot1q

Switch(config-if)#switchport mode trunk

配置三层交换机vlan10、20、30、40的ip地址

Switch(config)#interface vlan 10

Switch(config-if)#ip address 192.168.10.252 255.255.255.0

Switch(config)#interface vlan 20

Switch(config-if)#ip address 192.168.20.252 255.255.255.0

Switch(config)#interface vlan 30

Switch(config-if)#ip address 192.168.30.252 255.255.255.0

Switch(config)#interface vlan 40

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

Switch(config)#interface vlan 10

Switch(config-if)#ip address 192.168.10.253 255.255.255.0

Switch(config)#interface vlan 20

Switch(config-if)#ip address 192.168.20.253 255.255.255.0

Switch(config)#interface vlan 30

Switch(config-if)#ip address 192.168.30.253 255.255.255.0

Switch(config)#interface vlan 40

Switch(config-if)#ip address 192.168.40.253 255.255.255.0

===========================================================

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

Switch(config)#spanning-tree vlan 10 root primary

Switch(config)#spanning-tree vlan 20 root primary

Switch(config)#spanning-tree vlan 30 root secondary

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

Switch(config)#spanning-tree vlan 30 root primary

Switch(config)#spanning-tree vlan 40 root primary

Switch(config)#spanning-tree vlan 10 root secondary

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

Switch(config)#interface vlan 10

Switch(config-if)#standby 10 ip 192.168.10.254

Switch(config-if)#standby 10 priority 105

Switch(config-if)#standby 10 preempt

Switch(config)#interface vlan 20

Switch(config-if)#standby 20 ip 192.168.20.254

Switch(config-if)#standby 20 priority 105

Switch(config-if)#standby 20 preempt

Switch(config)#interface vlan 30

Switch(config-if)#standby 30 ip 192.168.30.254

Switch(config)#interface vlan 40

Switch(config-if)#standby 40 ip 192.168.40.254

查看热备份状态

Switch#show standby brief

MS2配置HSRP 使其成为vlan30、40的活跃路由器 vlan10、20的备份路由器

Switch(config)#interface vlan 30

Switch(config-if)#standby 30 ip 192.168.30.254

Switch(config-if)#standby 30 priority 105

Switch(config-if)#standby 30 preempt

Switch(config)#interface vlan 40

Switch(config-if)#standby 40 ip 192.168.40.254

Switch(config-if)#standby 40 priority 105

Switch(config-if)#standby 40 preempt

Switch(config)#interface vlan 10

Switch(config-if)#standby 10 ip 192.168.10.254

Switch(config)#interface vlan 20

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

Switch(config)#ip routing

然后测试目前网络是否可以达成全网互通。

============================================================

按图为路由器与三层交换机相连的接口配置ip

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

Switch(config)#router ospf 1

Switch(config-router)#network 192.168.10.0 0.0.0.255 area 0

Switch(config-router)#network 192.168.20.0 0.0.0.255 area 0

Switch(config-router)#network 192.168.30.0 0.0.0.255 area 0

Switch(config-router)#network 192.168.40.0 0.0.0.255 area 0

Switch(config-router)#network 192.168.50.0 0.0.0.255 area 0

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

Switch(config)#router ospf 1

Switch(config-router)#network 192.168.10.0 0.0.0.255 area 0

Switch(config-router)#network 192.168.20.0 0.0.0.255 area 0

Switch(config-router)#network 192.168.30.0 0.0.0.255 area 0

Switch(config-router)#network 192.168.40.0 0.0.0.255 area 0

Switch(config-router)#network 192.168.70.0 0.0.0.255 area 0

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

Router(config)#router ospf 1

Router(config-router)#network 192.168.50.0 0.0.0.255 area 0

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

Router(config)#router ospf 1

Router(config-router)#network 192.168.60.0 0.0.0.255 area 0

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

show ip route

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

Router(config)#ip nat inside source static 192.168.40.1 100.0.0.3

Router(config)#in g0/2

Router(config-if)#ip nat outside

Router(config-if)#in range g0/0-1

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）

Router(config)#ip route 0.0.0.0 0.0.0.0 100.0.0.10

Router(config)#router ospf 1

Router(config-router)#default-information originate

三层交换机如果看不到从路由器学习来的O\*默认路由就去检查路由器G0/2地址是否配置？

验证从外网可以访问内网的web服务。

===========================================================

Day 01 Shell概述 编写及执行脚本 Shell变量 总结和答疑

一.Shell环境及特性

1.什么是shell

· 在Linux内核与用户之间的解释器程序

-通常指/bin/bash

-负责向内核翻译及传达用户/程序指令

-相当于操作系统的"外壳"

·shell:概念的统称

·bash:具体的形式

# cat /etc/shells

/bin/sh

/bin/bash

/sbin/nologin

/usr/bin/sh

/usr/bin/bash

/usr/sbin/nologin

/bin/tcsh

/bin/csh

# yum -y install ksh

# cat /etc/shells

/bin/sh

/bin/bash

/sbin/nologin

/usr/bin/sh

/usr/bin/bash

/usr/sbin/nologin

/bin/tcsh

/bin/csh

/bin/ksh

# ksh

·切换到ksh解释器, 按exit退出

2.Shell的使用方式 一阶段回顾

· 交互式 --命令行

-人工干预,智能化程度高

-逐条解释执行,效率低

· 非交互式 --脚本

-需要提前设计,智能化难度大

-批量执行,效率高

-方便在后台静默运行

· 计划任务

# at 21:00

at >ls

at >cd /

at >ls

· 快捷键:

ctrl+s:挂起,冻结终端

ctrl+q:解除挂起/冻结终端

ctrl+d:结束输入

· 历史记录文件配置:

# vim /etc/profile

...

HISTSIZE=1000

history | wc -l

1000

1031 grep 'zhangsan' /etc/passwd

//两种调用方法

!grep

!1031

· 清除历史命令

# history -c //清空自己的历史命令

> ~/.bash\_history //清空记录文件

· 别名

1)查看别名

# alias

alias cp='cp -i'

alias egrep='egrep --color=auto'

alias fgrep='fgrep --color=auto'

alias grep='grep --color=auto'

alias l.='ls -d .\* --color=auto'

alias ll='ls -l --color=auto'

alias ls='ls --color=auto'

alias mv='mv -i'

alias rm='rm -i'

alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'

tip:别名设置一般存放在用户的.bashrc内

2)取消别名

# unalias

· 标准输入/输出/错误输出

标准输入(stdin),描述号为0;

标准输出(stdout),描述号为1;

保准错误(stderr),描述号为2;

1)重定向标准输出

- 使用'>'将命令执行的正常输出重定向到文件

- '>' 先清空再覆盖

- '>>' 追加重定向

2)重定向标准错误

- 使用'2>'将错误信息重定向到文件

- 使用'2>>'将错误信息追加重定向到文件

3)全部重定向

- 使用'&>'将所有信息重定向到文件

4)分别重定向

//将正确的信息重定向到a.txt,错误信息重定向到b.txt

# ls a.txt nb.txt >a.txt 2>b.txt

# cat a.txt b.txt

a.txt

ls: 无法访问nb.txt: 没有那个文件或目录

· 编写一个脚本,建立user1-user200

for i in {1..200}

do

useradd user$i 2>> a.log

echo "12345" | passwd --stdin user$i > /dev/null

done

3.Shell脚本格式

· 一个规范的Shell脚本构成包括：

- 脚本声明（需要的解释器、作者信息等）

- 注释信息（步骤、思路、用途、变量含义等）

- 可执行语句（操作代码）

4.Shell脚本运行方式

· 给权限 -> 绝对路径

· bash/sh 脚本名 - - 开子进程

· source 脚本名 - - 不开子进程

example:

# bash sleep.sh

# pstree | grep bash

|-sshd-+-sshd---bash---bash---sleep

| `-sshd---bash-+-grep

# source sleep.sh

# pstree | grep bash

|-sshd-+-sshd---bash---sleep

| `-sshd---bash-+-grep

# history | wc -l

11

# vim /etc/profile

HISTSIZE=10

# bash /etc/profile

# history | wc -l

14

# source /etc/profile

# history | wc -l

10

5.变量

1)建立变量

test=11

2)取消变量

unset test

3)变量与常量的区分

echo ${test}RMB

4)系统环境变量

大写

env:查看所有环境变量

5)$0 - 脚本自身

$1-n - 脚本第n个参数

$\* - 脚本所有参数

$$ - 脚本pid

$? - 判断脚本(命令)是否执行成功

6)符号运用

引号: 将内容划分为一个整体参数

- 单引号:内部符号全部判定为字符串

- 双引号:内部特殊符号照常执行,其他判定为字符串

双撇: 执行内部语句

7)read用法

# read str 由用户输入,赋值给变量str

//通过[-p]选项给出提升

# read -p "请输入一个整数:"i

8)stty终端显示控制

· 将回显功能关闭(stty -echo)

· 将回显功能恢复(stty echo)

9)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 yy //发布已定义的变量

# export XX="1234" //发布新变量

====================================================

Day 02 Shell中的数值运算 条件测试操作 使用if选择结构

一.Shell中的数值运算

1.整数运算工具

1)使用expr命令

· 乘法运算应用'\\*'转义,避免被认做通配符

[root@svr5 ~]# X=1234 //定义变量X

[root@svr5 ~]# expr $X + 78 //加法

1312

[root@svr5 ~]# expr $X - 78 //减法

1156

[root@svr5 ~]# expr $X \\* 78 //乘法，操作符应添加\转义

96252

[root@svr5 ~]# expr $X / 78 //除法，仅保留整除结果

15

[root@svr5 ~]# expr $X % 78 //求模

64

2)使用$[ ]或$(())表达式

[root@svr5 ~]# X=1234

[root@svr5 ~]# echo $[X+78]

1312

[root@svr5 ~]# echo $[X-78]

1156

[root@svr5 ~]# echo $[X\*78]

96252

[root@svr5 ~]# echo $[X/78]

15

[root@svr5 ~]# echo $[X%78]

64

3)使用let命令

expr或$[ ]、$(())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=1234，在执行let运算后的值会变更；另外，let运算操作并不显示结果，但是可以结合echo命令来查看：

[root@svr5 ~]# X=1234

[root@svr5 ~]# let y=X+22

[root@svr5 ~]# echo $y

1256

[root@svr5 ~]# let X++; echo $X # X++(X=X+1)

[root@svr5 ~]# let X--; echo $X # X--(X=X-1)

[root@svr5 ~]# let X+=78 ; echo $X # X+=78(X=X+78)

[root@svr5 ~]# let X-=78 ; echo $X # X-=78(X=X-78)

[root@svr5 ~]# let X\*=78 ; echo $X # X\*=78(X=X\*78)

[root@svr5 ~]# let X/=78 ; echo $X # X/=78(X=X/78)

[root@svr5 ~]# let X%=78 ; echo $X # X%=78(X=X%78)

4)bc交互式运算

[root@svr5 ~]# bc

12.34+56.78 //加法

69.12

12.34-56.78 //减法

-44.44

12.34\*56.78 //乘法

700.66

12.34/56.78 //除法

0

quit //退出交互计算器

5)bc非交互运算

[root@svr5 ~]# echo 'scale=4;12.34+5.678' | bc

18.018

[root@svr5 ~]# echo 'scale=4;12.34\*5.678' | bc

70.0665

[root@svr5 ~]# echo 'scale=4;12.34/5.678' | bc

2.1733

2.条件测试

· 子串判断 == != -z !-z

· 数字判断

-eq equal

-ne not equal

-gt greater than

-ge greater or equal

-lt less than

-le less or equal

· 文件或目录判断

-e exists 是否存在

-f file 是否为文件且存在

-d directory 是否为目录且存在

-r read 是否可读

-w write 是否可写

-x execute 是否可执行

1)'==' 比较两个字符串是否相同

2）!= 比较两个字符串是否不相同

3）一行执行多条命令的情况

# 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 //思考？

4) -z 检查变量的值是否未设置（空值）

[root@svr5 ~]# var1="nb" ; var2=""

[root@svr5 ~]# [ -z "$var1" ] && echo "空值" || echo "非空值"

非空值

[root@svr5 ~]# [ -z $var2 ] && echo "空值" || echo "非空值"

空值 //变量var2已设置，但无任何值，视为空

[root@svr5 ~]# [ ! -z $var1 ] //测试var1是否为非空

还有一个-n可以测试变量是否不为空（相当于! -z）。

===============================================================

Day 03 循环结构 case语句 、 函数及中断控制

一.for循环

1.不能代入变量的格式

for i in {1..100}

ex:

for i in {1..10}

do

echo "$i"

done

2.可以代入变量的格式

for i in `seq n`

for i in `ls \*.txt`

ex:

for i in `cat /root/user.txt

do

useradd $i

echo 123456 | passwd –stdin $i

done

· seq函数技巧: -s

3.c函数格式

for((i=0;i<10;i++))

ex:

for ((i=1;i<=5;i++))

do

echoi $i

done

二.while循环

1.死循环一般格式

ex:

while :

do

echo "hello world"

done

三.case分支编写脚本

case 变量 in

模式1)

命令序列;;

模式2)

命令序列;;

\*)

默认命令续写

esac

四.函数

1.格式

1)function 函数名 {

命令序列

....

}

2)函数名() {

命令序列

....

}

2.函数调用

# mycd(){

mkdir /test

cd /test

}

#mycd

3.编写mycolor.sh脚本

1）任务需求及思路分析

用户在执行时提供2个整数参数，这个可以通过位置变量$1、$2读入。

调用函数时，将用户提供的两个参数传递给函数处理。

颜色输出的命令:echo -e "\033[32mOK\033[0m"。

3X为字体颜色，4X为背景颜色。

2）根据实现思路编写脚本文件

[root@svr5 ~]# vim mycolor.sh

#!/bin/bash

cecho() {

echo –e "\033[$1m$2\033[0m"

}

cecho 32 OK

cecho 33 OK

cecho 34 OK

cecho 35 OK

[root@svr5 ~]# chmod +x mycolor.sh

4.Shell版本的fork炸弹(一键死机)

#!/bin/bash

.(){

.|.&

}

.

5.中断与退出

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

===========================================================

Day 04 字符串处理 扩展的脚本技巧 正则表达式

总结:

1.变量相关

echo ${变量}

echo ${变量::}

echo ${变量//}

echo ${变量#}

echo ${变量%}

2.正则

^开始

$结尾

[]集合,取集合中任意单个字符

[^]对集合取反

.任意单个字符

\*前一个字符任意次,包括0

\{n,m\}前一个字符n到m次

\{n\}前一个字符n次

\{n,\}前一个字符n次以上

\(\)保留

一.字符串截取及切割

1.字符串截取的三种方法

· ${变量名:起始位置:长度}

ex:

[root@svr5 ~]# phone="13788768897"

[root@svr5 ~]# echo ${#phone}

11 //包括11个字符

[root@svr5 ~]# echo ${phone:0:6}

137887

[root@svr5 ~]# echo ${phone:1:6}

378876

· expr substr "$变量名" 起始位置 长度

ex:

[root@svr5 ~]# echo $phone

13788768897

[root@svr5 ~]# expr substr "$phone" 1 6

137887

[root@svr5 ~]# expr substr "$phone" 9 3

897

· echo $变量名 | cut -b 起始位置-结束位置

ex:

[root@svr5 ~]# echo $phone

13788768897

[root@svr5 ~]# echo $phone | cut -b 1-6

137887

[root@svr5 ~]# echo $phone | cut -b 8-

8897

[root@svr5 ~]# echo $phone | cut -b 9

8

[root@svr5 ~]# echo $phone | cut -b 3,5,8

788

[root@svr5 ~]# vim rand.sh

#!/bin/bash

x=abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789

//所有密码的可能性是26+26+10=62（0-61是62个数字）

pass=''

for i in {1..8}

do

num=$[RANDOM%62]

tmp=${x:num:1}

pass=${pass}$tmp

done

echo $pass

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

2.子串替换的两种方法

· 只替换第一个匹配结果:${变量名/old/new}

[root@svr5 ~]# echo ${phone/8/X}

137X8768897

· 替换全部匹配结果:${变量//old/new}

[root@svr5 ~]# echo ${phone//8/X}

137XX76XX97

3.字符串掐头去尾

· 从左向右,最短匹配删除:${变量名#\*关键词}

[root@svr5 ~]# echo ${A#\*:}

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

· 从左向右,最长匹配删除:${变量名##\*关键词}

[root@svr5 ~]# echo $A //确认变量A的值

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

[root@svr5 ~]# echo ${A##\*:}

/bin/bash

· 从右向左,最短匹配删除:${变量名%关键词\*}

[root@svr5 ~]# echo ${A%:\*}

root:x:0:0:root:/root

· 从右向左,最长匹配删除:${变量名%%关键词\*}

[root@svr5 ~]# echo ${A%%:\*}

root

[root@svr5 rendir]# vim renfile.sh

#!/bin/bash

for i in `ls \*.doc` #注意这里有反引号

do

mv $i ${i%.\*}.txt

done

[root@svr5 ~]# chmod +x renfile.sh

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

额外:seq用法补充

#!/bin/bash

read -p '请输入累加最大值:' x

sum=0

x=${x:-1}

seq -s + $x | bc

seq -s - :

以-s后的符号将值分隔

seq -s + 10 等价于 1+2+3+4+5+6+7+8+9+10

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

4.字符串初值的处理

1）只取值，${var:-word}

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。

变量值已存在的情况：

[root@svr5 ~]# XX=11

[root@svr5 ~]# echo $XX //查看原变量值

11

[root@svr5 ~]# echo ${XX:-123} //因XX已存在，输出变量XX的值

11

· 变量值不存在的情况：

[root@svr5 ~]# echo ${YY:-123} //因YY不存在，输出“123”

123

# cat /root/test.sh

#!/bin/bash

read -p "请输入用户名:" user

[ -z $user ] && exit //如果无用户名，则脚本退出

read -p "请输入用户名:" pass

pass=${pass:-123456} //如果用户没有输入密码，则默认密码为123456

useradd $user

echo "$pass" | passwd --stdin $user

二.expect工具

1.安装expect工具

[root@svr5 ~]# yum -y install expect

2.写脚本

#!/bin/bash

expect << EOF

spawn ssh -o StrictHostKeyChecking=no 172.25.0.10

expect "password" {send "redhat\n"}

expect "#" {send "touch /abc.txt\n"}

expect "#" {send "exit\r"}

EOF

echo

三.正则表达式

1.基本正则表

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,\} 匹配前一个字符n次以上

\(\) 保留

2.扩展正则

正则符号 描述

+ 最少匹配一次

? 最多匹配一次

{n,m} 匹配n到m次

() 组合为整体,保留

| 或者

\b 单词边界

3.应用案例

1)使用grep

# grep '^r' /etc/passwd

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

rpc:x:32:32:Portmapper RPC user:/:/sbin/nologin

rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin

2)若使用grep -E或egrep命令，可支持扩展正则匹配模式

# grep -E '^root|^daemon' /etc/passwd

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

daemon:x:2:2:daemon:/sbin:/sbin/nologin

==============================================================

Day 05 sed基本用法 sed文本块处理 、 sed高级应用

一.sed工具

1.sed工具选项

-n 屏蔽默认输出

-r 让sed支持扩展正则

-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

sed -n '1p' /etc/hosts //输出第一行

sed -n 'p' a.txt //输出所有行

sed -n '4p' a.txt //输出第四行

sed -n '4,7p' a.txt //输出4~7行

sed -n '4,+10p' a.txt //输出第4行和后面10行

sed -n '/^bin/p' a.txt //输出以bin开头的行

sed -n '$=' a.txt //输出总共的行数

sed '3,5d' a.txt //删除第3-5行

sed '/xml/d' a.txt //删除包含xml的行

sed '/xml/!d' a.txt //删除不含xml的行

sed '/^install/d' a.txt //删除以install开头的行

sed '$d' a.txt //删除文件最后一行

sed '/^$/d' a.txt //删除空行

sed 's/2017/xxyy/' a.txt //替换第一个

sed 's/2017/xxyy/2' a.txt //替换前两个

sed 's/2017/xxyy/g' a.txt //替换全部

sed 's#/bin/bash#/sbin/sh#' passwd //替换/bin/bash为/sbin/sh

sed -i '1~2d' a.txt //删除文件中的奇数行

sed '4,7s/^/#/' a.txt //加注释

sed 's/^#an/an/' a.txt //去注释

· 将文件中每行的第一个、倒数第1个字符互换

思路:每行文本拆分为“第1个字符”、“中间的所有字符”、“倒数第1个字符”三个部分，然后通过替换操作重排顺序为“3-2-1”

sed -r 's/^(.)(.\*)(.)$/\3\2\1/' nssw.txt

sed [选项] '条件指令' 文件

-r

-n

-i

条件: 行号,/正则/,没有条件

指令: p,d,s,a,i,c

i:在指定的行之前插入文本 insert

a:在指定的行之后追加文本 appendDay 06 awk工具

一.使用awk提取文本

1.awk工具概述

· awk编程语言/数据处理引擎

- 创造者: Aho,Weinberger,Kernighan

- 基于模式匹配检查输入文本,逐行处理并输出

- 通常用在Shell脚本中,获取指定的数据

- 单独用时,可对文本数据做统计

2.主要用法

- 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

其中,print是最常用的编辑指令,若由多条编辑指令,可用分号分隔;

Awk过滤数据时仅支持打印某一列;

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;

awk常用内置变量：

$0 文本当前行的全部内容

$1 文本的第1列

$2 文件的第2列

$3 文件的第3列，依此类推

NR 文件当前行的行号

NF 文件当前行的列数（有几列）

· /var/log/secure日志文件存放他人对本机的远程记录

例子:

# cat test01.txt

hello world

ni hao ma

# awk '{print $1,$3}' test01.txt

hello

ni ma

# df -h | awk '{print $4,$6}'

可用 挂载点

7.0G /

# awk -F: '{print $1}' /etc/passwd

root

.. ..

# awk -F"[:/]" '{print $1,$10}' /etc/passwd

root bash

.. ..

# awk '{print $0}' test01.txt

hello world

ni hao ma

# awk '{print NR,NF}' test01.txt

1 2

2 3

# awk '{print $NF}' test01.txt

world

ma

# awk -F: '{print $1"的解释器是:"$7}' /etc/passwd

root的解释器是:/bin/bash

.. ..

# ifconfig eth0 | awk '/RX p/{print "入站网卡流量:"$5}'

入站网卡流量:422209

# df -h | awk '/\/$/{print $4}'

7.0G

# awk '/[AF][ca][ci]/{print $11}' /var/log/secure

172.25.0.250 //成功的

172.25.0.10 //成功的

172.25.0.10 //失败的

二.格式化输出

1.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。

awk [选项] '[条件]{指令}' 文件

awk [选项] 'BEGIN{指令}{指令}END{指令}'

· BEGIN{}行前处理.读取文件内容前执行,指令执行1次,一般用于初始化

· {}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,一般用于总结输出

/正则/ 全行匹配(包含即可)

$1~/正则/ 第一列匹配

$1!~/正则 取反

2.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)

数字/字符精确匹配

$3>=1000

$3<1000

$1=="root"

&& ||

# awk -F: '$3>1000&&$3<1003' /etc/passswd

# awk -F: '$3>1000||$3<10' /etc/passwd

对

# awk -F: '$3>1000||$3<1003' /etc/passwd

# awk -F: '$3>1000&&$3<10' /etc/passwd

错

3.分支结构

awk [选项] '条件{指令}' 文件

awk [选项] '{if($3<=1000){x++}}' 文件

案例:

# awk 'BEGIN{A=24;print A\*2}'

48

# awk 'BEGIN{print x+1}' //x可以不定义,直接用

1

# awk 'BEGIN{print 3.2+3.5}'

6.7

三.awk中数组的使用

# awk -F: 'BEGIN{print "用户名""\t""UID""\t""家目录"}{print $1" "$3" "$6}' /etc/passwd | column -t

用户名 UID 家目录

root 0 /root

# awk '/bash$/{print}' /etc/passwd

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

student:x:1000:1000:Student User:/home/student:/bin/bash

# awk '/bash$/' /etc/passwd //两者等价

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

student:x:1000:1000:Student User:/home/student:/bin/bash

# awk -F: '/^(root|adm)/{print $1,$3}' /etc/passwd //正则

root 0

adm 3

输出第1列包含root:

# awk -F: '$1~/root/' /etc/passwd

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

# awk -F: '$7!~/nologin$/{print $1,$7}' /etc/passwd

root /bin/bash

sync /bin/sync

shutdown /sbin/shutdown

halt /sbin/halt

student /bin/bash

# awk -F: 'NR==3{print $1}' /etc/passwd

awk -f: 'NR==3{print $1}' /etc/passwd

# awk -F: '$3>=1000{print $1}' /etc/passwd

nfsnobody

student

# awk -F: '$1=="root"' /etc/passwd

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

# awk -F: -v x=$i '$1==x' /etc/shadow

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

=======================================================

c:替换指定的行

sed '2c xxx' a.txt

sed 'r /etc/hosts' a.txt //读取hosts中的内容到a.txt中

tips:a.txt一定要有内容,不能是空白文件,哪怕一个空格也可以

2.sed复制剪切

· 基本动作

-H: 模式空间 ---- 追加 ----> 保持空间

-h: 模式空间 ---- 覆盖 ----> 保持空间

-G: 保持空间 ---- 追加 ----> 模式空间

-g: 保持空间 ---- 覆盖 ----> 模式空间

==========================================================

Day 06 awk工具

一.使用awk提取文本

1.awk工具概述

· awk编程语言/数据处理引擎

- 创造者: Aho,Weinberger,Kernighan

- 基于模式匹配检查输入文本,逐行处理并输出

- 通常用在Shell脚本中,获取指定的数据

- 单独用时,可对文本数据做统计

2.主要用法

- 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

其中,print是最常用的编辑指令,若由多条编辑指令,可用分号分隔;

Awk过滤数据时仅支持打印某一列;

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;

awk常用内置变量：

$0 文本当前行的全部内容

$1 文本的第1列

$2 文件的第2列

$3 文件的第3列，依此类推

NR 文件当前行的行号

NF 文件当前行的列数（有几列）

· /var/log/secure日志文件存放他人对本机的远程记录

例子:

# cat test01.txt

hello world

ni hao ma

# awk '{print $1,$3}' test01.txt

hello

ni ma

# df -h | awk '{print $4,$6}'

可用 挂载点

7.0G /

# awk -F: '{print $1}' /etc/passwd

root

.. ..

# awk -F"[:/]" '{print $1,$10}' /etc/passwd

root bash

.. ..

# awk '{print $0}' test01.txt

hello world

ni hao ma

# awk '{print NR,NF}' test01.txt

1 2

2 3

# awk '{print $NF}' test01.txt

world

ma

# awk -F: '{print $1"的解释器是:"$7}' /etc/passwd

root的解释器是:/bin/bash

.. ..

# ifconfig eth0 | awk '/RX p/{print "入站网卡流量:"$5}'

入站网卡流量:422209

# df -h | awk '/\/$/{print $4}'

7.0G

# awk '/[AF][ca][ci]/{print $11}' /var/log/secure

172.25.0.250 //成功的

172.25.0.10 //成功的

172.25.0.10 //失败的

二.格式化输出

1.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。

awk [选项] '[条件]{指令}' 文件

awk [选项] 'BEGIN{指令}{指令}END{指令}'

· BEGIN{}行前处理.读取文件内容前执行,指令执行1次,一般用于初始化

· {}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,一般用于总结输出

/正则/ 全行匹配(包含即可)

$1~/正则/ 第一列匹配

$1!~/正则 取反

2.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)

数字/字符精确匹配

$3>=1000

$3<1000

$1=="root"

&& ||

# awk -F: '$3>1000&&$3<1003' /etc/passswd

# awk -F: '$3>1000||$3<10' /etc/passwd

对

# awk -F: '$3>1000||$3<1003' /etc/passwd

# awk -F: '$3>1000&&$3<10' /etc/passwd

错

3.分支结构

awk [选项] '条件{指令}' 文件

awk [选项] '{if($3<=1000){x++}}' 文件

案例:

# awk 'BEGIN{A=24;print A\*2}'

48

# awk 'BEGIN{print x+1}' //x可以不定义,直接用

1

# awk 'BEGIN{print 3.2+3.5}'

6.7

三.awk中数组的使用

# awk -F: 'BEGIN{print "用户名""\t""UID""\t""家目录"}{print $1" "$3" "$6}' /etc/passwd | column -t

用户名 UID 家目录

root 0 /root

# awk '/bash$/{print}' /etc/passwd

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

student:x:1000:1000:Student User:/home/student:/bin/bash

# awk '/bash$/' /etc/passwd //两者等价

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

student:x:1000:1000:Student User:/home/student:/bin/bash

# awk -F: '/^(root|adm)/{print $1,$3}' /etc/passwd //正则

root 0

adm 3

输出第1列包含root:

# awk -F: '$1~/root/' /etc/passwd

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

# awk -F: '$7!~/nologin$/{print $1,$7}' /etc/passwd

root /bin/bash

sync /bin/sync

shutdown /sbin/shutdown

halt /sbin/halt

student /bin/bash

# awk -F: 'NR==3{print $1}' /etc/passwd

awk -f: 'NR==3{print $1}' /etc/passwd

# awk -F: '$3>=1000{print $1}' /etc/passwd

nfsnobody

student

# awk -F: '$1=="root"' /etc/passwd

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

# awk -F: -v x=$i '$1==x' /etc/shadow

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

===========================================================

Day 01 Nginx安装与升级 Nginx服务器 Nginx虚拟主机 、 HTTPS加密网站

一.Nginx安装与升级

1.Web服务器对比

· Unix和Linux平台下

- Apache,Nginx,Tengine,Lighttpd //php

- Apache Tomcat,IBM WebSphere,Redhat Jboss,Oracle weblogic //Java

· Windows平台下

- 微软公司的IIS

· Nginx

- 俄罗斯人,轻量级http服务器

- 是一个高性能的HTTP和反向代理服务器,同时也是一个IMAP/POP3/SMTP 代理服务器

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

nginx服务安装

# yum -y install gcc pcre-devel openssl-devel //安装依赖包

# useradd -s /sbin/nologin nginx

# tar -xf nginx-1.10.3.tar.gz

# cd nginx-1.10.3/

# ./configure \

> --prefix=/usr/local/nginx

> --user=nginx

> --group=nginx

> --with-http\_ssl\_module

# make && make install

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：

-a显示所有端口的信息

-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）

# netstat -antulp | grep nginx

tcp 0 0 0.0.0.0:80 0.0.0.0:\* LISTEN 6471/nginx: master

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

2.nginx升级

· 模块安装 -> 仅安装sbin下主文件

kill pid

killall name

二.nginx虚拟主机与网站加密

1.配置文件

/usr/local/nginx/conf/nginx.conf

2.虚拟主机配置

http{

server{

listen 80;

server\_name localhost;

root html;

index index.html;

}

}

3.网站加密

1)修改配置

http{

server{

listen 80;

server\_name localhost;

auth\_basic "Input Password:";

auth\_basic\_user\_file "/usr/local/nginx/pass";

root html;

index index.html;

}

}

# yum install httpd-tools -y //安装加密工具包

# htpass -c /usr/local/nginx/pass tom

New password:

Re-type new password:

Adding password for user dc

# nginx -s reload

· 验证:

[root@room9pc01 ~]# firefox 192.168.4.5

4.vim批量修改

ctrl+v 多选,相当于图形txt中按住shift

5.SSL虚拟主机

· 该站点通过https访问,通过私钥、证书对该站点所有数据加密

- 源码安装Nginx时必须使用--with-http\_ssl\_module参数，启用加密模块，对于需要进行SSL加密处理的站点添加ssl相关指令（设置网站需要的私钥和证书）。

· 专业用语解释: 公钥(证书)

· 加密算法一般分为对称算法、非对称算法、信息摘要。

- 对称算法有：AES、DES，主要应用在单机数据加密。

- 非对称算法有：RSA、DSA，主要应用在网络数据加密。

- 信息摘要：MD5、sha256，主要应用在数据完整性校验、数据秒传等。

# cd /usr/local/nginx/conf

# openssl genrsa > cert.key //生成私钥

# openssl req -new -x509 -key cert.key > cert.pem //生成证书

# vim /usr/local/nginx/conf/nginx.conf

… …

server {

listen 443 ssl;

server\_name www.c.com;

ssl\_certificate cert.pem; #这里是证书文件

ssl\_certificate\_key cert.key; #这里是私钥文件

ssl\_session\_cache shared:SSL:1m;

ssl\_session\_timeout 5m;

ssl\_ciphers HIGH:!aNULL:!MD5;

ssl\_prefer\_server\_ciphers on;

location / {

root html;

index index.html index.htm;

}

}

=================================================================

Day 02 部署LNMP Nginx+FastCGI 、 Nginx高级技术

一.部署LNMP(Linux,Nginx,Mariadb,PHP)

1.静态页面与动态页面

· 静态页面:无数据交互,只有页面文件(页面文件包括页面文字,图像,视频等素材),通过css(页面效果文件)可以实现一些简单或复杂的特效,但因为没有数据交互,也还是属于静态的范畴

· 动态页面:有数据交互,会根据用户的操作实现功能交互(javascript,php等),或执行某个脚本实现固定功能

2.环境准备

1)装包

· nginx

· mariadb、mariadb-server、mariadb-devel

· php、php-fpm、php-mysql

2)启服务

· nginx

· mariadb

· php-fpm

3.动静分离

· nginx实现: 如果用户访问的是静态页面,则自己直接找到页面,直接返回

· 如果用户访问的是动态php页面,则转发给9000端口,解释后再返回

· localtion匹配用户的地址栏

location / {

allow all;

}

location /test {

allow all;

deny 1.1.1.1;

}

location /abc {

deny all;

}

http://www.a.com

http://www.a.com/test/

http://www.a.com/abc/

http://www.a.com/qq/ //未定义,同'/'处理

location ~ \.php$ {

root html;

fastcgi\_pass 127.0.0.1:9000; #将请求转发给本机9000端口，PHP解释器

fastcgi\_index index.php;

#fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

include fastcgi.conf;

}

4.php-fpm配置文件

· FastCGI的内存消耗问题，一个PHP-FPM解释器将消耗约25M的内存。

# vim /etc/php-fpm.d/www.conf

[www]

listen = 127.0.0.1:9000 //PHP端口号

pm.max\_children = 32 //最大进程数量

pm.start\_servers = 15 //最小进程数量

pm.min\_spare\_servers = 5 //最少需要几个空闲着的进程

pm.max\_spare\_servers = 32 //最多允许几个进程处于空闲状态

Nginx的默认访问日志文件为/usr/local/nginx/logs/access.log

Nginx的默认错误日志文件为/usr/local/nginx/logs/error.log

PHP默认错误日志文件为/var/log/php-fpm/www-error.log

二.地址重写

1.关于Nginx服务器的地址重写，主要用到的配置参数是rewrite：

· rewrite regex(正则) replacement flag

· rewrite 旧地址 新地址 [选项]

# vim /usr/local/nginx/conf/nginx.conf

.. ..

server {

listen 80;

server\_name localhost;

location / {

root html;

index index.html index.htm;

rewrite /a.html /b.html;

}

}

# vim /usr/local/nginx/conf/nginx.conf

.. ..

server {

listen 80;

server\_name localhost;

location / {

root html;

index index.html index.htm;

rewrite /a.html /b.html redirect;

}

}

· redirect:地址栏随重写而变化

# vim /usr/local/nginx/conf/nginx.conf

.. ..

server {

listen 80;

server\_name localhost;

rewrite ^/(.\*)$ http://www.tmooc.cn/$1;

location / {

root html;

index index.html index.htm;

# rewrite /a.html /b.html redirect;

}

}

2.地址重写格式总结

rewrite 旧地址 新地址 [选项];

last 不再读其他rewrite

break 不再读其他语句，结束请求

redirect

//网络蜘蛛用(网络蜘蛛:百度,搜狗等)

redirect 临时重定向

permament 永久重定向

SEO SEM

==================================================================

Day 03 Nginx调度器 Nginx常见问题

一.Nginx反向代理

1.Nginx调度算法

- 轮询(默认的):逐一循环调度

- weight:指定轮询几率,权重值和访问比率成正比

- ip\_hash:根据客户端IP分配固定的后端服务器

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

#使用upstream定义后端服务器集群，集群名称任意(如webserver)

#使用server定义集群中的具体服务器和端口

upstream webserver {

ip\_hash; //相同客户访问相同服务器

server 192.168.2.100:80 max\_fails=2 fail\_timeout=30;

server 192.168.2.200:80;

server 192.168.2.101 down;

}

.. ..

server {

listen 80;

server\_name localhost;

location / {

#通过proxy\_pass将用户的请求转发给webserver集群

proxy\_pass http://webserver;

}

}

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

2.支持4层TCP/UDP代理的Nginx代理

1)部署nginx服务器

编译安装--with-stream

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

· 配置

# vim /usr/local/nginx/conf/nginx.conf

stream {

upstream backend {

server 192.168.2.100:22;

server 192.168.2.200:22;

}

server {

listen 12345;

proxy\_connect\_timeout 1s;

proxy\_timeout 3s;

proxy\_pass backend;

}

}

http {

.. ..

}

· 验证

# ssh root@192.168.4.5 -p 12345

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

二.Nginx常见问题处理

1.压力测试与服务器优化

· 压力测试

# ab -n 2000 -c 2000 http://192.168.4.5/

报错: socket: Too many open files

# vim /usr/local/nginx/conf/nginx.conf

worker\_processes 1; #与CPU核心数量有关

events {

worker\_connections 66666; #每个worker最大并发量

}

# ulimit -a

# ulimit -Hn 100000

# ulimit -Sn 100000

# vim /etc/security/limits.conf

http {

client\_header\_buffer\_size 1k; //默认请求包头信息的缓存

large\_client\_header\_buffers 4 4k; //大请求包头部信息的缓存个数与容量

.. ..

}

===================================================

2.浏览器本地静态缓存数据

· 服务器缓存设置

location ~\* \.(jpg|jpeg|gif|png|css|js|ico|xml)$ {

expires 30d; //定义客户端缓存时间为30天

}

· 报错页面设置

# vim /usr/local/nginx/conf/nginx.conf

.. ..

error\_page 404 /404.html; //自定义错误页面

.. ..

# vim /usr/local/nginx/html/404.html

====================================================

3.查看服务器状态信息

· 编译安装时使用--with-http\_stub\_status\_module开启状态页面模块

# ./configure \

> --with-http\_ssl\_module //开启SSL加密功能

> --with-stream //开启TCP/UDP代理模块

> --with-http\_stub\_status\_module //开启status状态页面

· 修改配置文件

# vim /usr/local/nginx/conf/nginx.conf

location /status {

stub\_status on;

}

· 验证

# curl http://192.168.4.5/status

Active connections: 1

server accepts handled requests

10 10 3

Reading: 0 Writing: 1 Waiting: 0

Active connections：当前活动的连接数量。

Accepts：已经接受客户端的连接总数量。

Handled：已经处理客户端的连接总数量（一般与accepts一致，除非服务器限制了连接数量）。

Requests：客户端发送的请求数量。

Reading：当前服务器正在读取客户端请求头的数量。

Writing：当前服务器正在写响应信息的数量。

Waiting：当前多少客户端在等待服务器的响应。

4.对页面进行压缩处理

1）修改Nginx配置文件

[root@proxy ~]# cat /usr/local/nginx/conf/nginx.conf

http {

.. ..

gzip on; //开启压缩

gzip\_min\_length 1000; //小文件不压缩

gzip\_comp\_level 4; //压缩比率

gzip\_types text/plain text/css application/json application/x-javascript text/xml application/xml application/xml+rss text/javascript;

//对特定文件压缩，类型参考mime.types

.. ..

}

5.服务器内存缓存

1）如果需要处理大量静态文件，可以将文件缓存在内存，下次访问会更快。

http {

open\_file\_cache max=2000 inactive=20s;

open\_file\_cache\_valid 60s;

open\_file\_cache\_min\_uses 5;

open\_file\_cache\_errors off;

//设置服务器最大缓存2000个文件句柄，关闭20秒内无请求的文件句柄

//文件句柄的有效时间是60秒，60秒后过期

//只有访问次数超过5次会被缓存

}

==================================================================

Day 04 memcached原理 部署memcached 、 Session共享

一.构建memcached服务

1.传统Web架构的问题

· mysql,marriadb: 数据库 --> 数据表

· 许多Web应用都将数据保存到RDBMS中,应用服务器从中读取数据并在浏览器中显示

· 随着数据量的增大,访问的集中,就会出现RDBMS的负担加重,数据库响应恶化,网站显示延迟等重大影响

· memcached:分布式缓存数据库,数据以key-value(键值对)形式存放

- 用来集中缓存数据库查询结果,减少数据库访问次数,以提高动态Web应用的响应速度

- 官方网站:http://memcached.org/

2.rhel7中yum源安装的服务配置文件目录

/usr/lib/systemd/system

与 systemctl xxx ooo.service 服务启动相关

3.memcached数据库操作

· 在启动memcached服务时,使用-m指定其占用的内存容量大小,-c用来限制memcached服务的最大连接数

· 通过telnet工具连接

# yum -y install telnet

# telnet 192.168.4.5 11211

add name 0 180 10 //变量不存在则添加

set name 0 180 10 //添加或替换变量

replace name 0 180 10 //替换

get name //读取变量

append name 0 180 10 //向变量中追加数据

delete name //删除变量

stats //查看状态

flush\_all //清空所有

提示：0表示不压缩，180为数据缓存时间，10为需要存储的数据字节数量。

4.PHP实现session共享

· 在前面的基础上:

# yum -y install php-pecl-memcache

# vim /etc/php-fpm.d/www.conf //修改该配置文件的两个参数

//文件的最后2行

修改前效果如下:

php\_value[session.save\_handler] = files

php\_value[session.save\_path] = /var/lib/php/session

//原始文件，默认定义Sessoin会话信息本地计算机（默认在/var/lib/php/session）

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

修改后效果如下:

php\_value[session.save\_handler] = memcache

php\_value[session.save\_path] = "tcp://192.168.2.5:11211"

//定义Session信息存储在公共的memcached服务器上，主机参数中为memcache（没有d）

//通过path参数定义公共的memcached服务器在哪（服务器的IP和端口）

[root@web1 ~]# systemctl restart php-fpm

=================================================================

Day 05 Tomcat服务器 Tomcat应用案例 、 Varnish代理服务器

一.安装部署Tomcat服务器

1.部署环境

· 使用RPM安装JDK环境

# yum –y install java-1.8.0-openjdk //安装JDK

# yum -y install java-1.8.0-openjdk-headless //安装JDK

# java -version //查看JAVA版本

· 安装Tomcat（apache-tomcat-8.0.30.tar.gz软件包，在lnmp\_soft中有提供）

# tar -xf apache-tomcat-8.0.30.tar.gz

# mv apache-tomcat-8.0.30 /usr/local/tomcat

# ls /usr/local/tomcat

# /usr/local/tomcat/bin/startup.sh //启动服务

· 验证

# ss -antulp | grep java //查看java监听的端口,正确的应该有3个端口

8080

8009

8005 用来关闭Tomcat的端口

二.使用Tomcat部署虚拟主机

1.修改server.xml配置文件，创建两个域名的虚拟主机，修改如下两个参数块：

# cat /usr/local/tomcat/conf/server.xml

<Host name="www.a.com" appBase="a" unpackWARS="true" autoDeploy="true">

</Host>

<Host name="www.b.com" appBase="b" unpackWARS="true" autoDeploy="true">

</Host>

2.修改www.b.com网站的首页目录为base

# vim /usr/local/tomcat/conf/server.xml

.. ..

<Host name="www.a.com" appBase="a" unpackWARS="true" autoDeploy="true">

</Host>

<Host name="www.b.com" appBase="b" unpackWARS="true" autoDeploy="true">

<Context path="" docBase="base" reloadable="true"/>

</Host>

# mkdir /usr/local/tomcat/b/base

# echo "BASE" > /usr/local/tomcat/b/base/index.html

# /usr/local/tomcat/bin/shutdown.sh

# /usr/local/tomcat/bin/startup.sh

3.跳转 : 当用户访问http://www.a.com/test打开/var/www/html目录下的页面

# vim /usr/local/tomcat/conf/server.xml

.. ..

<Host name="www.a.com" appBase="a" unpackWARS="true" autoDeploy="true">

<Context path="/test" docBase="/var/www/html/" />

</Host>

<Host name="www.b.com" appBase="b" unpackWARS="true" autoDeploy="true">

<Context path="" docBase="base" />

</Host>

# echo "Test" > /var/www/html/index.html

# /usr/local/tomcat/bin/shutdown.sh

# /usr/local/tomcat/bin/startup.sh

验证:

# firefox http://www.a.com:8080/test

//返回/var/www/html/index.html的内容

//注意，访问的端口为8080

4.配置Tomcat支持SSL加密网站

· 创建加密用的私钥和证书文件

# keytool -genkeypair -alias tomcat -keyalg RSA -keystore /usr/local/tomcat/keystore

//-genkeypair 生成密钥对

//-alias tomcat 密钥别名

//-keyalg RSA 定义密钥算法为RSA算法

//-keystore 定义密钥文件存储在:/usr/local/tomcat/keystore

# vim /usr/local/tomcat/conf/server.xml

<Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol"

maxThreads="150" SSLEnabled="true" scheme="https" secure="true"

keystoreFile="/usr/local/tomcat/keystore" keystorePass="123456" clientAuth="false" sslProtocol="TLS" />

//备注，默认这段Connector被注释掉了，打开注释，添加密钥信息即可

5.

二.Varnish

\* 代理服务器概念:

· proxy,代理人,经纪人,中介

· 歹徒用户处理事务

- 能提供正常情况下不便或无法访问的资源

· 应用场景:

- Web访问加速(正向/反向)

- IP伪装,"翻墙"

1.使用Varnish加速Web

· nginx [代理]

· varnish [代理]+缓存

· DNS分离解析

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

1)构建web服务器并创建测试页面:

httpd或nginx

2)部署Varnish缓存服务器(192.168.4.5)

· 编译安装Varnish

//安装依赖包

# yum -y install gcc readline-devel

# yum -y install ncurses-devel

# yum -y install pcre-devel

# yum -y install \

python-docutils-0.11-0.2.20130715svn7687.el7.noarch.rpm

//安装varnish

# useradd -s /sbin/nologin varnish //创建账户

# tar -xf varnish-5.2.1.tar.gz

# cd varnish-5.2.1

# ./configure

# make && make install

3)复制启动脚本及配置文件

· 注意: 这里的etc不是根下的etc

[root@proxy varnish-5.2.1]# cp etc/example.vcl /usr/local/etc/default.vcl

# vim /usr/local/etc/default.vcl

backend default {

.host = "192.168.2.100";

.port = "80";

}

3)其他操作

· 查看varnish日志

# varnishlog //varnish详细日志

# varnishncsa //简洁访问日志

· 更新缓存数据，在后台web服务器更新页面内容后，用户访问代理服务器看到的还是之前的数据，说明缓存中的数据过期了需要更新（默认也会自动更新，但非实时更新）

# varnishadm

varnish> ban req.url ~ .\*

//清空缓存数据，支持正则表达式

========================================================================

Day 06 版本控制 SVN基础 实战案例 、 RPM打包

一.Subversion基本操作(SVN)

1.什么是Subversion

· Subversion是一个自由/开源的版本控制系统

- 在SVN管理下,文件和目录可以超越时空

- SVN允许你数据回复到早期版本或者检查数据修改的历史

- 允许你和别人协作文档并跟踪所做的修改

运维工作中:

shell代码,配置文件等同步

2.Repository--仓库(repo)

· 客户端

- 命令行

- 图形

· 通信方式

- 本地访问

- SVN服务器

- Web服务

· 版本库

- 版本库是版本控制的核心

- 任意数量客户端

- 客户端通过写数据库分享代码

· SVN特点

- 记录每一次改变

3.安装SVN服务器

1)yum安装

# yum -y install subversion

2)创建版本库

# mkdir /var/svn

# svnadmin create /var/svn/project

# ls /var/svn/project/

conf/ db/ format hooks/ locks/ README.txt

3)本地导入初始化数据

# cd /usr/lib/systemd/system

$svn import . file:///var/svn/project/ -m "Init Data"

4)修改配置文件,创建账户和密码

# vim /var/svn/project/conf/svnserve.conf

· 注: 后面的几行前面不能有空格

//19行,匿名权限

anon-access = none

//20行,有效账户可写

auth-access = write

//27行,密码文件

password-db = passwd

//34行,ACL访问控制列表

authz-db = authz

# vim /var/svn/project/conf/passwd

.. ..

[users]

//用户名和密码

harry = 123456

tom = 123456

# cat /var/svn/project/conf/authz

//定义ACL访问控制

[/]

harry = rw 用户对项目根路径可读可写

tom = rw

5)启动服务

# svnserve -d -r /var/svn/project

# netstat -ntulp | grep svnserve

tcp 0 0 0.0.0.0:3690 0.0.0.0:\* LISTEN 4043/svnserve

备注：启动服务也可以使用svnserve -d启动，但客户端访问时需要指定绝对路径（svn://服务器IP/var/svn/project）

4.客户端测试(192.168.2.200)

1)将服务器上的代码下载到本地

# cd /tmp

# svn --username harry --password 123456 \

co svn://192.168.2.100/ code

//建立本地副本,从服务器192.168.2.100上co下载代码到本地code目录

//用户名harry,密码123456

# cd /tmp/code

# ls

# vim user.slice //挑选任意文件修改内容

# svn ci -m "modify user" //将本地修改的数据同步到服务器

# svn update //将服务器上新的数据同步到本地

# svn info svn://192.168.2.100 //查看版本仓库基本信息

# svn log svn://192.168.2.100 //查看版本仓库的日志

# echo "test" > test.sh //本地新建一个文件

# svn ci -m "new file" //提交失败,该文件不被svn管理

# svn add test.sh //将文件或目录加入版本控制

# svn ci -m "new file" //再次提交,成功

# svn mkdir subdir //创建子目录

# svn rm timers.target //使用svn删除文件

# svn ci -m "xxx" //提交一次代码

# svn mkdir

# vim umount.target //任意修改本地的一个文件

# svn diff //查看所有文件的差异

# svn diff umount.target //仅查看某一个文件的差异

# svn cat svn://192.168.2.100/reboot.target //查看服务器文件的内容

//删除文件所有内容,但未提交

# sed -i 'd' tmp.mount

//还原tmp.mount文件

# svn revert tmp.mount

//任意删除若干文件

# rm -rf \*.target

//还原

# svn update

//修改本地副本中的代码文件

# sed -i '1a #test###' tuned.service

//提交代码

# svn ci -m "xxx"

//将文件从版本7还原到版本2

# svn merge -r7:2 tuned.service

二.使用Subversion协同工作

1.多人协同工作

web1:

# cd /tmp

# svn --username tom --password 123456 \

> co svn://192.168.2.100/ code

web2:

# cd /tmp

# svn --username harry --password 123456 \

> co svn://192.168.2.100/ code

1)harry和tom修改不同的文件

[root@web1 mycode]# sed -i "3a ###harry modify#####" tmp.mount

[root@web1 mycode]# svn ci -m "has modified"

[root@web2 mycode]# sed -i "3a ###tom modify#####" umount.target

[root@web2 mycode]# svn ci -m "has modified"

[root@web2 mycode]# svn update

[root@web1 mycode]# svn update

2)harry和tom修改相同文件的不同行

[root@srv5 ~]# cd harry

[root@web1 mycode]# sed -i "3a ###harry modify#####" user.slice

[root@web1 mycode]# svn ci -m "modified"

[root@web2 mycode]# sed -i "6a ###tom modify#####" user.slice

[root@web2 mycode]# svn ci -m "modified" //提交失败

Sending svnserve

Transmitting file data .svn: Commit failed (details follow):

svn: File '/user.slice' is out of date（过期）

[root@web2 mycode]# svn update //提示失败后，先更新再提交即可

[root@web2 mycode]# svn ci -m "modified" //提交成功

Sending user.slice

Transmitting file data .

3)harry和tom修改相同文件的相同行

[root@web1 mycode]# sed -i '1c [UNIT]' tuned.service

[root@web1 mycode]# svn ci -m "modified"

[root@web2 mycode]# sed -i '1c [unit]' tuned.service

[root@web2 mycode]# svn ci -m "modified"

Sending tuned.service

Transmitting file data .svn: Commit failed (details follow):

svn: File '/tuned.service' is out of date(过期)

[root@web2 mycode]# svn update //出现冲突，需要解决

Conflict discovered in 'tuned.service'.

Select: (p) postpone, (df) diff-full, (e) edit,

(mc) mine-conflict, (tc) theirs-conflict,

(s) show all options:p //选择先标记p，随后解决

[root@web2 mycode]# ls

tuned.service tuned.service.mine tuned.service.r10 tuned.service.r9

[root@web2 mycode]# mv tuned.service.mine tuned.service

[root@web2 mycode]# rm -rf tuned.service.r10 tuned.service.r9

[root@web2 mycode]# svn ci -m "modified" //解决冲突

2.使用dump指令备份版本库数据

# svnadmin dump /var/svn/project > project.bak //备份

\* Dumped revision 0.

\* Dumped revision 1.

\* Dumped revision 2.

\* Dumped revision 3.

\* Dumped revision 4.

\* Dumped revision 5.

\* Dumped revision 6.

\* Dumped revision 7.

\* Dumped revision 8.

\* Dumped revision 9.

\* Dumped revision 10.

\* Dumped revision 11.

# svnadmin create /var/svn/project2 //新建空仓库

# svnadmin load /var/svn/project2 < project.bak //还原

三.制作nginx的RPM包

· 什么是RPM--本质上是压缩包

· 制作RPM包

1.安装rpm-build软件包

# yum -y install rpm-build

· 生成rpmbuild目录结构

# rpmbuild -ba nginx.spec //会报错,没有文件或目录

# ls /root/rpmbuild //自动生成的目录结构

BUILD BUILDROOT RPMS SOURCES SPECS SRPMS

//准备工作,将源码复制到SOURCES目录

# cp nginx-1.12.2.tar.gz /root/rpmbuild/SOURCES/

//创建并修改SPEC配置文件

# vim /root/rpmbuild/SPECS/nginx.spec

Name:nginx

Version:1.12.2

Release: 10

Summary: Nginx is a web server software.

License:GPL

URL: www.test.com

Source0:nginx-1.12.2.tar.gz

#BuildRequires:

#Requires:

%description

nginx [engine x] is an HTTP and reverse proxy server.

%prep

%setup –q //自动解压源码包，并cd进入目录

%build

./configure

make %{?\_smp\_mflags}

%install

make install DESTDIR=%{buildroot}

cp /root/rpmbuild/SPECS/nginx.sh %{buildroot}/usr/local/nginx/

##注意，cp非必须操作，注意，这里是将一个脚本拷贝到安装目录，必须提前准备该文件

%files

%doc

/usr/local/nginx/\* //对哪些文件与目录打包

%changelog

2.ppmbuild创建RPM软件包

# rpmbuild -ba /root/rpmbuild/SPECS/nginx.spec

# ls /root/rpmbuild/RPMS/x86\_64/nginx-1.12.2-10.x86\_64.rpm

# rpm -qpi RPMS/x86\_64/nginx-1.12.2-10.x86\_64.rpm

# rpm -ivh RPMS/x86\_64/nginx-1.12.2-10.x86\_64.rpm

# rpm -qa |grep nginx

# /usr/local/nginx/sbin/nginx

# curl http://127.0.0.1/

三.git的使用

git

git clone

git add

git commit -m "xx"

gitp push

在本机上同步git

git clone git clone https://github.com/redhatedu/course

cd course

git pull

=========================================================================

Day 07 VPN服务器 NTP时间同步 PSSH远程工具

一.VPN服务器

1.VPN概述

· Virtual Private Network(虚拟专用网络)

- 在公用网络上建立专用私有网络,进行加密通讯

- 多用于为集团公司的各地子公司建立连接

- 连接完成后,各个地区的子公司可以像局域网一样通讯

- 在企业中由广泛应用

- 偶尔可以用于翻墙

- 目前主流的VPN技术(GRE,PPTP,L2TP+IPSec,SSL)

2.GRE模块VPN

1)启用GRE模块(client和proxy都需要操作，下面以client为例)

1.查看计算机当前加载的模块

# lsmod //显示模块列表

# lsmod | grep ip\_gre //确定是否加载了gre模块

2.加载模块ip\_gre

# modprobe ip\_gre

# lsmod | grep ip\_gre

3.查看模块信息

# modinfo ip\_gre

2)Client主机创建VPN隧道

1.创建隧道

# ip tunnel add tun0 mode gre \

> remote 201.1.2.5 local 201.1.2.10

//ip tunnel add 创建隧道(隧道名称为tun0),ip tunnel help 可以查看帮助

//mode设置隧道使用gre模式

//local后面跟本机的IP地址,remote后面是与其他主机建立隧道的对方ip地址

//ip link 查看

2.启用该隧道(类似于设置网卡ip)

# ip link show

# ip link set tun0 up //设置up

# ip link show

3.为VPN配置隧道IP地址

# ip addr add 10.10.10.10/24 peer 10.10.10.5/24 \

> dev tun0

//为隧道tun0设置本地IP地址(10.10.10.10/24)

//隧道对面的主机ip的ip隧道为10.10.10.5/24

# ip a s //查看ip地址

4.关闭防火墙和selinux

# firewall-cmd --set-default-zone=trusted

3)Proxy主机创建VPN隧道

1.同上步骤开启通道: ping 10.10.10.10 可通

2.开启路由转发,关闭防火墙

# echo "1" > /proc/sys/net/ipv4/ip\_forward

# firewall-cmd --set-default-zone=trust

3.创建PPTP VPN

· 部署VPN服务器

1)安装软件包

[root@proxy ~]# yum localinstall pptpd-1.4.0-2.el7.x86\_64.rpm

[root@proxy ~]# rpm -qc pptpd

/etc/ppp/options.pptpd

/etc/pptpd.conf

/etc/sysconfig/pptpd

2)修改配置文件

[root@proxy ~]# vim /etc/pptpd.conf

.. ..

localip 201.1.2.5 //服务器本地IP

remoteip 192.168.3.1-50 //分配给客户端的IP池

[root@proxy ~]# vim /etc/ppp/options.pptpd

require-mppe-128 //使用MPPE加密数据

ms-dns 8.8.8.8 //DNS服务器

[root@proxy ~]# vim /etc/ppp/chap-secrets //修改账户配置文件

jacob \* 123456 \*

//用户名 服务器标记 密码 客户端

[root@proxy ~]# echo "1" > /proc/sys/net/ipv4/ip\_forward //开启路由转发

3)启动服务

# systemctl start pptpd

# systemctl enable pptpd

# firewall-cmd --set-default-zone=trusted

4)翻墙设置(非必须)

root@proxy ~]# iptables -t nat -A POSTROUTING -s 192.168.3.0/24 \

> -j SNAT --to-source 201.1.2.5

· 客户端设置

启动一台Windows虚拟机，将虚拟机网卡桥接到public2，配置IP地址为201.1.2.20。

新建网络连接，输入VPN服务器账户与密码，连接VPN并测试网络连通性。

4.创建L2TP+IPSec VPN

· 部署IPSec服务

1)安装软件包

# yum -y install libreswan

2)新建IPSec密钥验证配置文件

# cat /etc/ipsec.conf //仅查看一下该主配置文件

.. ..

include /etc/ipsec.d/\*.conf //加载该目录下的所有配置文件

[root@client ~]# vim /etc/ipsec.d/myipsec.conf

//新建该文件，参考lnmp\_soft/vpn/myipsec.conf

conn IDC-PSK-NAT

rightsubnet=vhost:%priv //允许建立的VPN虚拟网络

also=IDC-PSK-noNAT

conn IDC-PSK-noNAT

authby=secret //加密认证

ike=3des-sha1;modp1024 //算法

phase2alg=aes256-sha1;modp2048 //算法

pfs=no

auto=add

keyingtries=3

rekey=no

ikelifetime=8h

keylife=3h

type=transport

left=201.1.2.200 //重要，服务器本机的外网IP

leftprotoport=17/1701

right=%any //允许任何客户端连接

rightprotoport=17/%any

3)创建IPSec预定义共享密钥

# cat /etc/ipsec.secrets //仅查看，不要修改该文件

include /etc/ipsec.d/\*.secrets

# vim /etc/ipsec.d/mypass.secrets //新建该文件

201.1.2.200 %any: PSK "randpass" //randpass为密钥

//201.1.2.200是VPN服务器的IP

4)启动IPSec服务

[root@client ~]# systemctl start ipsec

[root@client ~]# netstat -ntulp |grep pluto

udp 0 0 127.0.0.1:4500 0.0.0.0:\* 3148/pluto

udp 0 0 192.168.4.200:4500 0.0.0.0:\* 3148/pluto

udp 0 0 201.1.2.200:4500 0.0.0.0:\* 3148/pluto

udp 0 0 127.0.0.1:500 0.0.0.0:\* 3148/pluto

udp 0 0 192.168.4.200:500 0.0.0.0:\* 3148/pluto

udp 0 0 201.1.2.200:500 0.0.0.0:\* 3148/pluto

udp6 0 0 ::1:500 :::\* 3148/pluto

· 部署XL2TP服务

1)安装软件包

# yum localinstall xl2tpd-1.3.8-2.el7.x86\_64.rpm

2) 修改xl2tp配置文件（修改3个配置文件的内容)

# vim /etc/xl2tpd/xl2tpd.conf

[global]

.. ..

[lns default]

.. ..

ip range = 192.168.3.128-192.168.3.254 //分配给客户端的IP池

local ip = 201.1.2.200 //VPN服务器的IP地址

# vim /etc/ppp/options.xl2tpd //认证配置

require-mschap-v2 //添加一行，强制要求认证

#crtscts //注释或删除该行

#lock //注释或删除该行

# vim /etc/ppp/chap-secrets //修改密码文件

jacob \* 123456 \* //账户名称 服务器标记 密码 客户端IP

3)启动服务

# systemctl start xl2tpd

# ss -antulp | grep xl2tpd

udp 0 0 0.0.0.0:1701 0.0.0.0:\* 3580/xl2tpd

4）设置路由转发，防火墙

# echo "1" > /proc/sys/net/ipv4/ip\_forward

# firewall-cmd --set-default-zone=trusted

5）翻墙设置（非必需操作）

# iptables -t nat -A POSTROUTING -s 192.168.3.0/24 \

> -j SNAT --to-source 201.1.2.200

二.NTP时间同步

1.NTP协议概述

· Network Time Protocol(网络时间协议)

· 它用来同步网络中各个计算机的时间的协议

· 210.72.145.39(国家授时中心服务器IP地址)

· Stratum(分层设计)

· Stratum层的总数限制在15以内(包括15)

2.部署NTP服务

1)安装软件包

# yum -y install chrony

# rpm -qc chrony //查看配置文件列表

/etc/chrony.conf

/etc/chrony.keys

.. ..

2)修改配置文件

cat /etc/chrony.conf

.. ..

server 0.centos.pool.ntp.org iburst //server用户客户端指向上层NTP服务器

allow 192.168.4.0/24 //允许那个IP或网络访问NTP

#deny 192.168.4.1 //拒绝那个IP或网络访问NTP

local stratum 10 //设置NTP服务器的层数量

.. ..

3)启动NTP服务

# systemctl restart chronyd

# systemctl enable chronyd

3.配置客户端

1)安装软件包

# yum –y install chrony

2) 修改配置文件

# vim /etc/chrony.conf

server 192.168.4.5 iburst //设置与哪台服务器同步数据

//iburst参数设置重启服务后尽快同步时间

3) 将客户端时间修改为错误的时间

# date -s "hour:minute" //调整时间（小时：分钟）

# date //查看修改后的时间

4) 重启chrony与服务器同步时间

# systemctl restart chronyd

5) 确认时间是否已经同步

# date //多执行几次查看结果

三.pssh远程套件工具

1.作用

a)使用密码批量、多并发远程其他主机

b)使用密钥批量、多并发远程其他主机

c)批量、多并发拷贝数据到其他主机

d)批量、多并发从其他主机下载数据到本机

e)批量、多并发杀死其他主机的进程

2.准备工作

1）安装软件包

# rpm -ivh pssh-2.3.1-5.el7.noarch.rpm

2) 修改/etc/hosts本地解析文件

cat /etc/hosts

… …

192.168.2.100 host1

192.168.2.200 host2

192.168.4.100 host3

… …

3) 创建主机列表文件

# cat /root/host.txt //每行一个用户名、IP或域名

… …

root@host1

host2

host3

… …

3.使用密码批量、多并发远程其他主机

1）语法格式

# man pssh //通过man帮助查看工具选项的作用

pssh提供并发远程连接功能

-A 使用密码远程其他主机（默认使用密钥）

-i 将输出显示在屏幕

-H 设置需要连接的主机

-h 设置主机列表文件

-p 设置并发数量

-t 设置超时时间

-o dir 设置标准输出信息保存的目录

-e dir 设置错误输出信息保存的目录

-x 传递参数给ssh

4.使用密钥批量、多并发远程其他主机

1）生成密钥并发送密钥到其他主机

# ssh-keygen -N '' -f /root/.ssh/id\_rsa //非交互生成密钥文件

# ssh-copy-id host1

# ssh-copy-id host2

# ssh-copy-id host3

2)使用密钥远程其他主机

# pssh -h host.txt echo hello

3)使用密钥远程其他主机，将标准输出信息写入到/tmp目录

# pssh -h host.txt -o /tmp/ echo hello

5.批量、多并发拷贝数据到其他主机

1）语法格式

# man pscp.pssh //通过man帮助查看工具选项的作用

pscp.pssh提供并发拷贝文件功能

-r 递归拷贝目录

· 其他选项基本与pssh一致

2) 将本地的/etc/hosts拷贝到远程主机的/tmp目录下

# pscp.pssh -h host.txt /etc/hosts /tmp

3) 递归将本地的/etc目录拷贝到远程主机的/tmp目录下

# pscp.pssh -r -h host.txt /etc /tmp

6.批量、多并发从其他主机下载数据到本机

1）语法格式

# man pslurp //通过man帮助查看工具选项的作用

pslurp提供远程下载功能

· 选项与pscp.pssh基本一致

2) 将远程主机的/etc/passwd，拷贝到当前目录下，存放在对应IP下的pass文件中

# pslurp -h host.txt /etc/passwd /pass

· 注意：最后的pass是文件名

3) 将远程主机的/etc/passwd目录，拷贝到media下，存放在对应IP下的pass文件

# pslurp -h host.txt -L /media /etc/passwd /pass

7.批量、多并发杀死其他主机的进程

1）语法格式

# man pnuke //通过man帮助查看工具选项的作用

pnuke提供远程杀死进程的功能

· 选项与pssh基本一致

2) 将远程主机上的sleep进程杀死

# pnuke -h host.txt sleep

3) 将远程主机上的test相关脚本都杀死（如：test1,testtt,test2等等）

# pnuke -h host.txt test

4) 将远程主机上的test.sh脚本杀死

# pnuke -h host.txt test.sh

==============================================================================

Day 01 Linux基本防护 用户切换与提权 SSH访问控制 SELinux安全 、 SSH访问控制 SELinux安全

一.Linux基本防护

· 用户账号安全

1.设置账号有效日期

· 使用chage工具

-d 0 强制修改密码

-l 查看账户信息

-E yyyy-mm-dd 指定失效日期(-1取消) 失效的用户将无法登录

· 定义默认有效期（强制定期修改密码）

# cat /etc/login.defs

PASS\_MAX\_DAYS 99999 //密码最长有效期

PASS\_MIN\_DAYS 0 //密码最短有效期

PASS\_MIN\_LEN 5 //密码最短长度

PASS\_WARN\_AGE 7 //密码过期前几天提示警告信息

UID\_MIN 1000 //UID最小值

UID\_MAX 60000 //UID最大值

2.账号的锁定/解锁

· 使用passwd命令

-l 锁定

-u 解锁

-S 查看状态

3.修改tty登录的提示信息，隐藏系统版本(伪装登录提示)

· vim /etc/issue 本地登录

· vim /etc/issue.net 远程登录

4.文件系统安全

· 程序和服务控制

1) 禁用非必要的系统服务

- 使用systemctl,chkconfig工具

2) 关闭执行权限

# mount -o remount,rw,noexec /dev/vda1 /boot/ //临时修改,永久修改需写入fstab中

# mount | grep boot

3) 关闭更新atime

# # mount -o remount,rw,atime /dev/vda1 /boot/ //临时修改,永久修改需写入fstab中

5.锁定/解锁保护文件

· EXT3/EXT4的文件属性控制

- chattr,lsattr

· +,-,=控制方式

-i 不可变

-a 仅可追加

· 用户切换与提权

1.su切换用户身份

· su - [目标用户]

· su [-] -c "命令" [目标用户]

2.sudu提升执行权限

· 管理员预先设置执行许可

· 被授权用户有执行授权的命令,验证自己的口令

· 命令格式

# sudo 特权命令

# sudo [-u 目标用户] 特权命令

· 授权

# vim /etc/sudoers

或

# visudo

· 允许用户useradm通过sudo方式添加/删除/修改除root以外的用户账号

zhangsan ALL=(ALL) /usr/bin/systemctl

useradm ALL=(root) /usr/bin/passwd,!/usr/bin/passwd root,/usr/sbin/user\*,!/usr/sbin/user\* root

· 允许wheel组成员以特权执行所有命令

%wheel ALL=(ALL) ALL

# usermod -a -G wheel zhangsan

· 为sudo机制启用日志记录，以便跟踪sudo执行操作

# visudo

Defaults logfile="/var/log/sudo"

· 提高SSH服务安全

1.配置基本安全策略

# vim /etc/ssh/sshd\_config

.. ..

Protocol 2 //SSH协议

PermitRootLogin no //禁止root用户登录

PermitEmptyPasswords no //禁止密码为空的用户登录

UseDNS no //不解析客户机地址

LoginGraceTime 1m //登录限时

MaxAuthTries 3 //每连接最多认证次数

.. ..

# systemctl restart sshd

2.针对SSH访问采用仅允许的策略，未明确列出的用户一概拒绝登录

# vim /etc/ssh/sshd\_config

.. ..

AllowUsers zhangsan tom useradm@192.168.4.0/24 //定义账户白名单

##DenyUsers USER1 USER2 //定义账户黑名单

##DenyGroups GROUP1 GROUP2 //定义组黑名单

##AllowGroups GROUP1 GROUP2 //定义组白名单

# systemctl restart sshd

· 验证SSH访问控制，未授权的用户将拒绝登录

· SELinux安全防护

1.SELinux概述

· 什么是SELinux

- 一套强化Linux安全的扩展模块

- 美国国家安全局主导开发

· SELinux的运作机制

- 集成到Linux内核(2.6及以上)

- 操作系统可定制的策略,管理工具

· SELinux策略集

SELINUXTYPE=targeted

- 推荐,仅保护最常见/关键的网络服务,其他不限制

- 主要软件包:

selinux-policy

selinux-policy-targeted

libselinux-utils

coreutils

policycoreutuilks

2.SELinux模式控制

1）固定配置：修改/etc/selinux/config文件

· 修改kernel启动参数

- 添加selinux=0以禁用

- 添加enforcing=0 设置SELinux为允许模式

· 修改文件/etc/selinux/config

- 设置SELINUX=disabled以禁用

- 设置SELINUX=permissive宽松/允许模式

- 设置SELINUX=enforcing强制模式

==============================================================================

Day 02 加密与解密 AIDE入侵检测系统 扫描与抓包

一.加密与解密

· 加/解密概述

1.信息传递中的风险

· 电脑黑客

· 出差人员

· 离职人员

· 合作伙伴

· 商业间谍

· 高管习惯

· 开发人员

· 流程失控

2.什么是加密/解密

· 发送方 : 明文 -> 密文

· 接收方 : 密文 -> 明文

3.加密目的及方式

· 确保数据的机密性

- 对称加密:加密/解密用同一个密钥

- 非对称加密:加密/解密用不同密钥

· 保护信息的完整性

- 信息摘要:基于输入的信息生成长度较短,数位固定的散列值

4.常见的加密算法

· 对称加密

- DES

- AES

· 非对称加密

- RSA

- DSA

· Hash散列技术,用于信息摘要

- MD5

- SHA(256/512)

5.MD5完整性教研

· 使用md5sum校验工具

- 生成MD5校验值

- 与软件官方提供的校验值比对

vimdiff a b 比对a,b两个文件

· GPG加/解密工具

1.使用GPG对称加密方式保护文件

1) 确保已经安装了相关软件（默认已经安装好了）

# yum -y install gnupg2 //安装软件

# gpg --version //查看版本

gpg (GnuPG) 2.0.22

2) gpg使用对称加密算法加密数据的操作

# echo 233 > 1.txt //创建1.txt

# gpg -c 1.txt //对称加密

# cat 1.txt.gpg //显示为乱码

�ʐik�����­����n-r6\_x0010\_X�E@��2��MJ9C�L����

3) 使用gpg对加密文件进行解密操作

# gpg -d 1.txt //对称解密

# gpg -d 1.txt.gpg | cat

gpg: 钥匙环‘/root/.gnupg/secring.gpg’已建立

gpg: CAST5 加密过的数据

gpg: 以 1 个密码加密

gpg: 警告：报文未受到完整的保护

233

2.使用GPG非对称加密方式保护文件

非对称加密/解密文件时，UserA（192.168.4.100）生成私钥与公钥，并把公钥发送给UserB（192.168.4.5），UserB使用公钥加密数据，并把加密后的数据传给UserA，UserA最后使用自己的私钥解密数据

1) 接收方UserA创建自己的公钥,私钥对(192.168.4.100)

# gpg --gen-key //创建密钥对

请选择您要使用的密钥种类：

(1) RSA and RSA (default)

(2) DSA and Elgamal

(3) DSA (仅用于签名)

(4) RSA (仅用于签名)

您的选择？

RSA 密钥长度应在 1024 位与 4096 位之间。

您想要用多大的密钥尺寸？(2048)

您所要求的密钥尺寸是 2048 位

请设定这把密钥的有效期限。

0 = 密钥永不过期

<n> = 密钥在 n 天后过期

<n>w = 密钥在 n 周后过期

<n>m = 密钥在 n 月后过期

<n>y = 密钥在 n 年后过期

密钥的有效期限是？(0)

密钥永远不会过期

以上正确吗？(y/n)y

You need a user ID to identify your key; the software constructs the user ID

from the Real Name, Comment and Email Address in this form:

"Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"

真实姓名：tommy

电子邮件地址：tom@163.com

注释：abc

您选定了这个用户标识：

“tommy (abc) <tom@163.com>”

更改姓名(N)、注释(C)、电子邮件地址(E)或确定(O)/退出(Q)？o

您需要一个密码来保护您的私钥。

您不想要有密码――这个想法实在是遭透了！

不过，我仍然会照您想的去做。您任何时候都可以变更您的密码，仅需要

再次执行这个程序，并且使用“--edit-key”选项即可。

我们需要生成大量的随机字节。这个时候您可以多做些琐事(像是敲打键盘、移动

鼠标、读写硬盘之类的)，这会让随机数字发生器有更好的机会获得足够的熵数。

我们需要生成大量的随机字节。这个时候您可以多做些琐事(像是敲打键盘、移动

鼠标、读写硬盘之类的)，这会让随机数字发生器有更好的机会获得足够的熵数。

gpg: 密钥 C600BC96 被标记为绝对信任

公钥和私钥已经生成并经签名。

gpg: 正在检查信任度数据库

gpg: 需要 3 份勉强信任和 1 份完全信任，PGP 信任模型

gpg: 深度：0 有效性： 1 已签名： 0 信任度：0-，0q，0n，0m，0f，1u

pub 2048R/C600BC96 2018-08-30

密钥指纹 = 9E41 785B BBF8 3558 E71C D197 DA72 E794 C600 BC96

uid tommy (abc) <tom@163.com>

sub 2048R/3EF61A55 2018-08-30

2) UserA导出自己的公钥文件

· 用户的公钥,私钥信息分别保存在pubring.gpg和secring.gpg文件内

· 使用gpg命令结合--export选项将其中的公钥文本导出

# gpg -a --export UserA > UserA.pub //--export的作用是导出密钥，-a的作用是导出的密钥存储为ASCII格式

# scp UserA.pub 192.168.4.5:/tmp/ //将密钥传给Proxy

3) UserB使用公钥加密数据，并把加密后的数据传给UserA（在192.168.4.5操作）

# echo "I love you ." > love.txt

# gpg -e -r UserA love.txt

无论如何还是使用这把密钥吗？(y/N)y //确认使用此密钥加密文件

//-e选项是使用密钥加密数据

//-r选项后面跟的是密钥，说明使用哪个密钥对文件加密

# scp love.txt.gpg 192.168.4.100:/root //加密的数据传给UserA

4) UserA以自己的私钥解密文件（在192.168.4.100操作）

# gpg -d love.txt.gpg > love.txt

您需要输入密码，才能解开这个用户的私钥：“UserA (UserA) <UserA@tarena.com>”

2048 位的 RSA 密钥，钥匙号 9FA3AD25，建立于 2017-08-16 (主钥匙号 421C9354)

//验证私钥口令

gpg: 由 2048 位的 RSA 密钥加密，钥匙号为 9FA3AD25、生成于 2017-08-16

“UserA (UserA) <UserA@tarena.com>”

# cat love.txt //获得解密后的文件内容

I love you.

3.使用GPG的签名机制，检查数据来源的正确性

1) 在client(192.168.4.100)上，UserA为软件包创建分离式签名

# tar zcf log.tar /var/log //建立测试软件包

# gpg -b log.tar //创建分离式数字签名

# ls -lh log.tar\*

-rw-rw-r--. 1 root root 170 8月 17 21:18 log.tar

-rw-rw-r--. 1 root root 287 8月 17 21:22 log.tar.sig

# scp log.tar\* 192.168.4.5:/root //将签名文件与签名传给UserB

2) 在192.168.4.5上验证签名

# gpg --verify log.tar.sig log.tar

gpg:于2028年06月07日 星期六 23时23分23秒 CST 创建的签名，使用 RSA，钥匙号 421C9354

gpg: 完好的签名，来自于“UserA (UserA) <UserA@tarena.com>”

.. ..

二.使用AIDE做入侵检测

· 初始化系统

1.安装软件包

# yum -y install aide

2.修改配置文件

# vim /etc/aide.conf

@@define DBDIR /var/lib/aide //数据库目录

@@define LOGDIR /var/log/aide //日志目录

database\_out=file:@@{DBDIR}/aide.db.new.gz //数据库文件名

//一下内容为可以检查的项目（权限，用户，组，大小，哈希值等）

#p: permissions

#i: inode:

#n: number of links

#u: user

#g: group

#s: size

#md5: md5 checksum

#sha1: sha1 checksum

#sha256: sha256 checksum

DATAONLY = p+n+u+g+s+acl+selinux+xattrs+sha256

//以下内容设置需要对哪些数据进行入侵校验检查

//注意：为了校验的效率，这里将所有默认的校验目录与文件都注释

//仅保留/root目录，其他目录都注释掉

/root DATAONLY

#/boot NORMAL //对哪些目录进行什么校验

#/bin NORMAL

#/sbin NORMAL

#/lib NORMAL

#/lib64 NORMAL

#/opt NORMAL

#/usr NORMAL

#!/usr/src //使用[!]，设置不校验的目录

#!/usr/tmp

· 初始化数据库，入侵后检测

1.入侵前对数据进行校验，生成初始化数据库

[root@proxy ~]# aide --init

AIDE, version 0.15.1

AIDE database at /var/lib/aide/aide.db.new.gz initialized.

//生成校验数据库，数据保存在/var/lib/aide/aide.db.new.gz

# mv /var/lib/aide/aide.db.new.gz /var/lib/aide/aide.db.gz

//校验时是和不带new的文件进行比对

2.备份数据库，将数据库文件拷贝到U盘（非必须的操作）

[root@proxy ~]# cp /var/lib/aide/aide.db.new.gz /media/

3.入侵后检测

[root@proxy ~]# cd /var/lib/aide/

[root@proxy ~]# mv aide.db.new.gz aide.db.gz

[root@proxy ~]# aide --check //检查哪些数据发生了变化

三.扫描与抓包分析

· 使用NMAP扫描来获取指定主机/网段的相关信息

1.安装软件

# yum -y install nmap

//基本用法：

# nmap [扫描类型] [选项] <扫描目标 ...>

//常用的扫描类型

// -sS，TCP SYN扫描（半开）

// -sT，TCP 连接扫描（全开）

// -sU，UDP扫描

// -sP，ICMP扫描

// -A，目标系统全面分析

2.检查192.168.4.100主机是否可以ping通

# nmap -sP 192.168.4.100

Starting Nmap 6.40 ( http://nmap.org ) at 2018-06-06 21:59 CST

mass\_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers

Nmap scan report for host3 (192.168.4.100)

Host is up (0.00036s latency).

MAC Address: 52:54:00:71:07:76 (QEMU Virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.02 seconds

· 使用-n选项可以不执行DNS解析

· 可以批量检查网段(192.168.2.0/24)

3.检查目标主机所开启的TCP服务

[root@proxy ~]# nmap -sT 192.168.4.100

.. ..

PORT STATE SERVICE

21/tcp open ftp

22/tcp open ssh

25/tcp open smtp

80/tcp open http

110/tcp open pop3

111/tcp open rpcbind

143/tcp open imap

443/tcp open https

993/tcp open imaps

995/tcp open pop3s

MAC Address: 00:0C:29:74:BE:21 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.2 seconds

· 检查192.168.4.0/24网段内哪些主机开启了FTP、SSH服务

[root@proxy ~]# nmap -p 21-22 192.168.4.0/24

[root@proxy ~]# nmap -n -sT -p 22-80 192.168.2.0/24

· nmap -sT 全开扫描

· nmap -sS 半开扫描

4.检查目标主机所开启的UDP服务

[root@proxy ~]# nmap -sU 192.168.4.100 //指定-sU扫描UDP

53/udp open domain

111/udp open rpcbind

5.全面分析目标主机192.168.4.100和192.168.4.5的操作系统信息

[root@proxy ~]# nmap -A 192.168.4.100,5

· 中间人攻击

IP地址欺骗

MAC地址欺骗

IP不同(域名类似)

www.icbc.coe[955xx发短信]

dhcp(ip,netmask,网关)

\* 中间人攻击!=抓包

· 使用tcpdump分析FTP访问中的明文交换信息

1.准备vsftpd服务器

# yum -y install vsftpd

# systemctl restart vsftpd

2.启用tcpdump命令行抓包

# tcpdump

//监控选项如下：

// -i，指定监控的网络接口（默认监听第一个网卡）

// -A，转换为 ACSII 码，以方便阅读

// -w，将数据包信息保存到指定文件

// -r，从指定文件读取数据包信息

//tcpdump的过滤条件：

// 类型：host、net、port、portrange

// 方向：src、dst

// 协议：tcp、udp、ip、wlan、arp、……

// 多个条件组合：and、or、not

========================================================

Day 03 系统审计 服务安全 Linux安全之打补丁

一.系统审计

· 什么是审计

- 基于事先配置的规则生成日志,记录可能发生在系统上的事件

- 审计不会为系统提供额外的安全保护,但他会发现并记录违反安全策略的人及其对应的行为

- 审计能够记录的日志内容:

a.日期与事件,事件结果

b.触发事件的用户

c.所有认证机制的使用都可以被记录,如ssh等

d.对关键数据文件的修改行为等

· 审计的案例

- 监控文件访问

- 监控系统调用

- 记录用户运行的命令

- 审计可以监控网络访问行为

- ausearch工具,可以根据条件过滤审计日志

- aureport工具,可以生成审计报告

1.部署audit监控文件

1) 安装软件包，查看配置文件（确定审计日志的位置）

# yum -y install audit //安装软件包

# cat /etc/audit/auditd.conf //查看配置文件，确定日志位置

log\_file = /var/log/audit/audit.log //日志文件路径

# systemctl start auditd //启动服务

# systemctl enable auditd //设置开机自启

2) 配置审计规则

· 可以使用auditctl命令控制审计系统并设置规则

# auditctl -s //查询状态

# auditctl -l //查看规则

# auditctl -D //删除所有规则

· 定义临时文件系统规则：

#语法格式：auditctl -w path -p permission -k key\_name

# path为需要审计的文件或目录

# 权限可以是r,w,x,a(文件或目录的属性发生变化)

# Key\_name为可选项，方便识别哪些规则生成特定的日志项

· example:

# auditctl -w /etc/passwd -p wa -k passwd\_change

//设置规则所有对passwd文件的写、属性修改操作都会被记录审计日志

# auditctl -w /etc/selinux/ -p wa -k selinux\_change

//设置规则，监控/etc/selinux目录

# auditctl -w /usr/sbin/fdisk -p x -k disk\_partition

//设置规则，监控fdisk程序

# auditclt -w /etc/ssh/sshd\_conf -p warx -k sshd\_config

//设置规则，监控sshd\_conf文件

· 如果需要创建永久审计规则，则需要修改规则配置文件：

# vim /etc/audit/rules.d/audit.rules

-w /etc/passwd -p wa -k passwd\_changes

-w /usr/sbin/fdisk -p x -k partition\_disks

2.查看并分析日志

1）手动查看日志

查看SSH的主配置文件/etc/ssh/sshd\_conf，查看audit日志信息：

[root@proxy ~]# tailf /var/log/audit/audit.log

type=SYSCALL msg=audit(1517557590.644:229228): arch=c000003e

syscall=2 success=yes exit=3

a0=7fff71721839 a1=0 a2=1fffffffffff0000 a3=7fff717204c0

items=1 ppid=7654 pid=7808 auid=0 uid=0 gid=0 euid=0 suid=0

fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts2 ses=3 comm="cat"

exe="/usr/bin/cat"

subj=unconfined\_u:unconfined\_r:unconfined\_t:s0-s0:c0.c1023 key="sshd\_config"

.. ..

#内容分析

# type为类型

# msg为(time\_stamp:ID)，时间是date +%s（1970-1-1至今的秒数）

# arch=c000003e，代表x86\_64（16进制）

# success=yes/no，事件是否成功

# a0-a3是程序调用时前4个参数，16进制编码了

# ppid父进程ID，如bash，pid进程ID，如cat命令

# auid是审核用户的id，su - test, 依然可以追踪su前的账户

# uid，gid用户与组

# tty:从哪个终端执行的命令

# comm="cat" 用户在命令行执行的指令

# exe="/bin/cat" 实际程序的路径

# key="sshd\_config" 管理员定义的策略关键字key

# type=CWD 用来记录当前工作目录

# cwd="/home/username"

# type=PATH

# ouid(owner's user id） 对象所有者id

# guid(owner's groupid） 对象所有者id

2）通过工具搜索日志

系统提供的ausearch命令可以方便的搜索特定日志，默认该程序会搜索/var/log/audit/audit.log，ausearch options -if file\_name可以指定文件名。

[root@proxy ~]# ausearch -k sshd\_config -i

//根据key搜索日志，-i选项表示以交互式方式操作

二.加固常见服务的安全

1.优化Nginx服务的安全配置

1） 删除不需要的模块

Nignx是模块化设计的软件，需要什么功能与模块以及不需要哪些模块，都可以在编译安装软件时自定义，使用--with参数可以开启某些模块，使用--without可以禁用某些模块。最小化安装永远都是对的方案！

下面是禁用某些模块的案例：

[root@proxy ~]# tar -xf nginx-1.12.tar.gz

[root@proxy ~]# cd nginx-1.12

[root@proxy nginx-1.12]# ./configure \

>--without-http\_autoindex\_module \ //禁用自动索引文件目录模块

>--without-http\_ssi\_module

[root@proxy nginx-1.12]# make

[root@proxy nginx-1.12]# make install

2）修改版本信息，并隐藏具体的版本号

默认Nginx会显示版本信息以及具体的版本号，这些信息给攻击者带来了便利性，便于他们找到具体版本的漏洞。

如果需要屏蔽版本号信息，执行如下操作，可以隐藏版本号。

[root@proxy ~]# vim /usr/local/nginx/conf/nginx.conf

… …

http{

server\_tokens off; //在http下面手动添加这么一行

… …

}

[root@proxy ~]# nginx -s reload

[root@proxy ~]# curl -I http://192.168.4.5 //查看服务器响应的头部信息

· 但服务器还是显示了使用的软件为nginx，通过如下方法可以修改该信息。

[root@proxy nginx-1.12]# vim +48 src/http/ngx\_http\_header\_filter\_module.c

//注意：vim这条命令必须在nginx-1.12源码包目录下执行！！！！！！

//该文件修改前效果如下：

static u\_char ngx\_http\_server\_string[] = "Server: nginx" CRLF;

static u\_char ngx\_http\_server\_full\_string[] = "Server: " NGINX\_VER CRLF;

static u\_char ngx\_http\_server\_build\_string[] = "Server: " NGINX\_VER\_BUILD CRLF;

//下面是我们修改后的效果：

static u\_char ngx\_http\_server\_string[] = "Server: Jacob" CRLF;

static u\_char ngx\_http\_server\_full\_string[] = "Server: Jacob" CRLF;

static u\_char ngx\_http\_server\_build\_string[] = "Server: Jacob" CRLF;

//修改完成后，再去编译安装Nignx，版本信息将不再显示为Nginx，而是Jacob

[root@proxy nginx-1.12]# ./configure

[root@proxy nginx-1.12]# make && make install

[root@proxy nginx-1.12]# killall nginx

[root@proxy nginx-1.12]# /usr/local/nginx/sbin/nginx //启动服务

[root@proxy nginx-1.12]# curl -I http://192.168.4.5 //查看版本信息验证

3） 限制并发量

DDOS攻击者会发送大量的并发连接，占用服务器资源（包括连接数、带宽等），这样会导致正常用户处于等待或无法访问服务器的状态。

Nginx提供了一个ngx\_http\_limit\_req\_module模块，可以有效降低DDOS攻击的风险，操作方法如下：

[root@proxy ~]# vim /usr/local/nginx/conf/nginx.conf

… …

http{

… …

limit\_req\_zone $binary\_remote\_addr zone=one:10m rate=1r/s;

server {

listen 80;

server\_name localhost;

limit\_req zone=one burst=5;

}

}

//备注说明：

//limit\_req\_zone语法格式如下：

//limit\_req\_zone key zone=name:size rate=rate;

//上面案例中是将客户端IP信息存储名称为one的共享内存，内存空间为10M

//1M可以存储8千个IP信息，10M可以存储8万个主机连接的状态，容量可以根据需要任意调整

//每秒中仅接受1个请求，多余的放入漏斗

//漏斗超过5个则报错

[root@proxy ~]# /usr/local/nginx/sbin/nginx -s reload

· 客户端使用ab测试软件测试效果：

[root@client ~]# ab -c 100 -n 100 http://192.168.4.5/

4） 拒绝非法的请求

通过如下设置可以让Nginx拒绝非法的请求方法：

[root@proxy ~]# vim /usr/local/nginx/conf/nginx.conf

http{

server {

listen 80;

#这里，!符号表示对正则取反，~符号是正则匹配符号

#如果用户使用非GET或POST方法访问网站，则retrun返回400的错误信息

if ($request\_method !~ ^(GET|POST)$ ) {

return 400;

}

}

}

[root@proxy ~]# /usr/local/nginx/sbin/nginx -s reload

修改服务器配置后，客户端使用不同请求方法测试：

[root@client ~]# curl -i -X GET http://192.168.4.5 //正常

[root@client ~]# curl -i -X HEAD http://192.168.4.5 //报错

5） 防止buffer溢出

当客户端连接服务器时，服务器会启用各种缓存，用来存放连接的状态信息。

如果攻击者发送大量的连接请求，而服务器不对缓存做限制的话，内存数据就有可能溢出（空间不足）。

修改Nginx配置文件，调整各种buffer参数，可以有效降低溢出风险。

[root@proxy ~]# vim /usr/local/nginx/conf/nginx.conf

http{

client\_body\_buffer\_size 1K;

client\_header\_buffer\_size 1k;

client\_max\_body\_size 1k;

large\_client\_header\_buffers 2 1k;

… …

}

[root@proxy ~]# /usr/local/nginx/sbin/nginx -s reload

2.数据库安全

1） 初始化安全脚本

安装完MariaDB或MySQL后，默认root没有密码，并且提供了一个任何人都可以操作的test测试数据库。有一个名称为mysql\_secure\_installation的脚本，该脚本可以帮助我们为root设置密码，并禁止root从远程其他主机登陆数据库，并删除测试性数据库test。

[root@proxy ~]# systemctl status mariadb

//确保服务已启动

[root@proxy ~]# mysql\_secure\_installation

//执行初始化安全脚本

2）密码安全

手动修改MariaDB或MySQL数据库密码的方法：

[root@proxy ~]# mysqladmin -uroot -predhat password 'mysql'

//修改密码，旧密码为redhat，新密码为mysql

[root@proxy ~]# mysql -uroot -pmysql

MariaDB [(none)]>set password for root@'localhost'=password('redhat');

//使用账户登录数据库，修改密码

MariaDB [(none)]> select user,host,password from mysql.user;

· 修改密码成功，而且密码在数据库中是加密的，有什么问题吗？问题是你的密码被明文记录了，下面来看看明文密码：

[root@proxy ~]# cat .bash\_history

mysqladmin -uroot -pxxx password 'redhat'

//通过命令行修改的密码，bash会自动记录历史，历史记录中记录了明文密码

[root@proxy ~]# cat .mysql\_history

set password for root@'localhost'=password('redhat');

select user,host,password from mysql.user;

flush privileges;

//通过mysql命令修改的密码，mysql也会有所有操作指令的记录，这里也记录了明文密码

管理好自己的历史，不使用明文登录，选择合适的版本5.6以后的版本，日志，行为审计（找到行为人），使用防火墙从TCP层设置ACL（禁止外网接触数据库）。

3）数据备份与还原

首先，备份数据库（注意用户名为root，密码为redhat）：

[root@proxy ~]# mysqldump -uroot -predhat mydb table > table.sql

//备份数据库中的某个数据表

[root@proxy ~]# mysqldump -uroot -predhat mydb > mydb.sql

//备份某个数据库

[root@proxy ~]# mysqldump -uroot -predhat --all-databases > all.sql

//备份所有数据库

接下来，还原数据库（注意用户名为root，密码为redhat）：

[root@proxy ~]# mysql -uroot -predhat mydb < table.sql //还原数据表

[root@proxy ~]# mysql -uroot -predhat mydb < mydb.sql //还原数据库

[root@proxy ~]# mysql -uroot -predhat < all.sql //还原所有数据库

4）数据安全

在服务器上（192.168.4.5），创建一个数据库账户：

[root@proxy ~]# mysql -uroot -predhat

//使用管理员，登陆数据库

MariaDB [(none)]> grant all on \*.\* to tom@'%' identified by '123';

//创建一个新账户tom

使用tcpdump抓包（192.168.4.5）

[root@proxy ~]# tcpdump -w log -i any src or dst port 3306

//抓取源或目标端口是3306的数据包，保存到log文件中

客户端（192.168.4.100）从远程登陆数据库服务器（192.168.4.5）

[root@client ~]# mysql -utom -p123 -h 192.168.4.5

//在192.168.4.100这台主机使用mysql命令登陆远程数据库服务器（192.168.4.5）

//用户名为tom，密码为123

MariaDB [(none)]> select \* from mysql.user;

//登陆数据库后，任意执行一条查询语句

回到服务器查看抓取的数据包

[root@proxy ~]# tcpdump -A -r log

//使用tcpdump查看之前抓取的数据包，很多数据库的数据都明文显示出来

· 可以使用SSH远程连接服务器后，再从本地登陆数据库;

· 或者也可以使用SSL对MySQL服务器进行加密，类似与HTTP+SSL一样，MySQL也支持SSL加密

3.Tomcat安全性

1） 隐藏版本信息、修改tomcat主配置文件（隐藏版本信息）

未修改版本信息前，使用命令查看服务器的版本信息

注意：proxy有192.168.2.5的IP地址，这里使用proxy作为客户端访问192.168.2.100服务器。

[root@proxy ~]# curl -I http://192.168.2.100:8080/xx

//访问不存在的页面文件，查看头部信息

[root@proxy ~]# curl -I http://192.168.2.100:8080

//访问存在的页面文件，查看头部信息

[root@proxy ~]# curl http://192.168.2.100:8080/xx

//访问不存在的页面文件，查看错误信息

· 修改tomcat配置文件，修改版本信息(在192.168.2.100操作)：

[root@web1 tomcat]# yum -y install java-1.8.0-openjdk-devel

[root@web1 tomcat]# cd /usr/local/tomcat/lib/

[root@web1 lib]# jar -xf catalina.jar

[root@web1 lib]# vim org/apache/catalina/util/ServerInfo.properties

//根据自己的需要，修改版本信息的内容

[root@web1 lib]# /usr/local/tomcat/bin/shutdown.sh //关闭服务

[root@web1 lib]# /usr/local/tomcat/bin/startup.sh //启动服务

· 修改后，客户端再次查看版本信息（在192.168.2.5操作）：

[root@proxy ~]# curl -I http://192.168.2.100:8080/xx

//访问不存在的页面文件，查看头部信息

[root@proxy ~]# curl -I http://192.168.2.100:8080

//访问存在的页面文件，查看头部信息

[root@proxy ~]# curl http://192.168.2.100:8080/xx

//访问不存在的页面文件，查看错误信息

· 再次修改tomcat服务器配置文件，修改版本信息，手动添加server参数（在192.168.2.100操作）：

[root@web1 lib]# vim /usr/local/tomcat/conf/server.xml

<Connector port="8080" protocol="HTTP/1.1"

connectionTimeout="20000" redirectPort="8443" server="jacob" />

[root@web1 lib]# /usr/local/tomcat/bin/shutdown.sh //关闭服务

[root@web1 lib]# /usr/local/tomcat/bin/startup.sh //启动服务

· 修改后,客户端再次查看版本信息（在192.168.2.5操作）：

[root@proxy ~]# curl -I http://192.168.2.100:8080/xx

//访问不存在的页面文件，查看头部信息

[root@proxy ~]# curl -I http://192.168.2.100:8080

//访问存在的页面文件，查看头部信息

[root@proxy ~]# curl http://192.168.2.100:8080/xx

//访问不存在的页面文件，查看错误信息

2）降级启动

默认tomcat使用系统高级管理员账户root启动服务，启动服务尽量使用普通用户。

[root@web1 ~]# useradd tomcat

[root@web1 ~]# chown -R tomcat:tomcat /usr/local/tomcat/

//修改tomcat目录的权限，让tomcat账户对该目录有操作权限

[root@web1 ~]# su -c /usr/local/tomcat/bin/startup.sh tomcat

//使用su命令切换为tomcat账户，以tomcat账户的身份启动tomcat服务

[root@web1 ~]# chmod +x /etc/rc.local //该文件为开机启动文件

[root@web1 ~]# vim /etc/rc.local //修改文件，添加如下内容

su -c /usr/local/tomcat/bin/startup.sh tomcat

3）删除默认的测试页面

[root@web1 ~]# rm -rf /usr/local/tomcat/webapps/\*

三.使用diff和patch工具打补丁

1.比单个文件差异

1） 编写两个版本的脚本，一个为v1版本，一个为v2版本。

[root@proxy ~]# cat test1.sh //v1版本脚本

#!/bin/bash

echo "hello wrld"

[root@proxy ~]# cat test2.sh //v2版本脚本

#!/bin/bash

echo "hello the world"

echo "test file"

2） 使用diff命令语法

使用diff命令查看不同版本文件的差异。

[root@proxy ~]# diff test1.sh test2.sh //查看文件差异

@@ -1,3 +1,3 @@

#!/bin/bash

-echo "hello world"

-echo "test"

+echo "hello the world"

+echo "test file"

[root@proxy ~]# diff -u test1.sh test2.sh //查看差异，包含头部信息

--- test1.sh 2018-02-07 22:20:02.723971251 +0800

+++ test2.sh 2018-02-07 22:20:13.358760687 +0800

@@ -1,3 +1,3 @@

#!/bin/bash

-echo "hello world"

-echo "test"

+echo "hello the world"

+echo "test file"

2.使用patch命令对单文件代码打补丁

1）准备实验环境

[root@proxy ~]# cd demo

[root@proxy demo]# vim test1.sh

#!/bin/bash

echo "hello world"

echo "test"

[root@proxy demo]# vim test2.sh

#!/bin/bash

echo "hello the world"

echo "test file"

2) 生成补丁文件

[root@proxy demo]# diff -u test1.sh test2.sh > test.patch

3）使用patch命令打补丁

在代码相同目录下为代码打补丁

[root@proxy demo]# yum -y install patch

[root@proxy demo]# patch -p0 < test.patch //打补丁

patching file test1.sh

//patch -pnum（其中num为数字，指定删除补丁文件中多少层路径前缀）

//如原始路径为/u/howard/src/blurfl/blurfl.c

//-p0则整个路径不变

//-p1则修改路径为u/howard/src/blurfl/blurfl.c

//-p4则修改路径为blurfl/blurfl.c

//-R(reverse)反向修复，-E修复后如果文件为空，则删除该文件

[root@proxy demo]# patch -RE < test.patch //还原旧版本，反向修复

==================================================================

Day 04 iptables防火墙 filter表控制 扩展匹配 nat表典型应用

一.iptables防火墙

· Linux包过滤防火墙

· RHEL7默认使用firewalld作为防火墙

· 但firewalld底层还是调用包过滤防火墙iptables

1)iptables的四个表(区分大小写) :

iptables默认有4个表: nat表(地址转换表),filter表(数据过滤表),raw表(状态跟踪表),mangle表(包标记表)

2)iptables的五个链(区分大小写) :

INPUT链(入站规则)

OUTPUT(出站规则)

FORWARD(转发规则)

PREROUTING(路由前规则)

POSTROUTING(路由后规则)

1.iptables命令的基本使用方法

# iptables [-t 表名] 选项 [链名] [条件] [-j 目标操作]

# iptables -t filter -I INPUT -p icmp -j REJECT

# iptables -t filter -I INPUT -p icmp -j ACCEPT

# iptables -I INPUT -p icmp -j REJECT

//注意事项与规律：

//可以不指定表，默认为filter表

//可以不指定链，默认为对应表的所有链

//如果没有找到匹配条件，则执行防火墙默认规则

//选项/链名/目标操作用大写字母，其余都小写

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

//目标操作：

// ACCEPT：允许通过/放行

// DROP：直接丢弃，不给出任何回应

// REJECT：拒绝通过，必要时会给出提示

// LOG：记录日志，然后传给下一条规则

iptables常用选项

添加规则:

-A 追加一条防火墙规则至链末尾位置

-I 插入一条防火墙规则至链的开头

查看规则:

-L 查看iptables 所有规则

-n 以数字形式显示地址,端口等信息

--line-numbers 查看规则时,显示规则的行号

删除规则:

-D 删除链内指定序号(或内容)的一条规则

-F 清空所有的规则

-P 为指定的链设置默认规则

# iptables -t filter -A INPUT -p tcp -j ACCEPT

//追加规则至filter表中的INPUT链的末尾，允许任何人使用TCP协议访问本机

# iptables -I INPUT -p udp -j ACCEPT

//插入规则至filter表中的INPUT链的开头，允许任何人使用UDP协议访问本机

# iptables -I INPUT 2 -p icmp -j ACCEPT

//插入规则至filter表中的INPUT链的第2行，允许任何人使用ICMP协议访问本机

· 查看iptables防火墙规则

# iptables -nL INPUT //仅查看INPUT链的规则

· 删除规则，清空所有规则

# iptables -D INPUT 3 //删除filter表中INPUT链的第3条规则

# iptables -nL INPOUT

# iptables -F //清空filter表中所有链的防火墙规则

· 设置防火墙默认规则

# iptables -t filter -P INPUT DROP

# iptables -nL

Chain INPUT (policy DROP)

iptables防火墙规则的条件

通用匹配:

协议匹配 -p 协议名称

地址匹配 -s 源地址,-d 目标地址

接口匹配 -i 接受数据的网卡,-o 发送数据的网卡

隐含匹配:

端口匹配 --sport 源端口号,--dport 目标端口号

ICMP 类型匹配 --icmp-type ICMP 类型

# iptables -I INPUT -p tcp --dport 80 -j REJECT

# iptables -I INPUT -s 192.168.2.100 -j REJECT

# iptables -I INPUT -d 192.168.2.5 -p tcp --dport 80 -j REJECT

# iptables -I INPUT -i eth0 -p tcp --dport 80 -j REJECT

# iptables -A INPUT -s 192.168.4.100 -j DROP

//丢弃192.168.4.100发给本机的所有数据包

2.禁ping的相关策略

1) 默认直接禁ping的问题

# iptables -I INPUT -p icmp -j DROP

//设置完上面的规则后，其他主机确实无法ping本机，但本机也无法ping其他主机

//当本机ping其他主机，其他主机回应也是使用icmp，对方的回应被丢弃

2) 禁止其他主机ping本机，允许本机ping其他主机

# iptables -A INPUT -p icmp \

> --icmp-type echo-request -j DROP

//仅禁止入站的ping请求，不拒绝入站的ping回应包

注意：关于ICMP的类型，可以参考help帮助，参考命令如下：

# iptables -p icmp --help

二.防火墙扩展规则

1.根据MAC地址过滤

# ip link show eth0 //查看client的MAC地址

eth0: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP mode DEFAULT qlen 1000

link/ether 52:54:00:00:00:0b brd ff:ff:ff:ff:ff:ff

[root@proxy ~]# iptables -A INPUT -p tcp --dport 22\

> -m mac --mac-source 52:54:00:00:00:0b -j DROP

//拒绝52:54:00:00:00:0b这台主机远程本机

2.基于多端口设置过滤规则

# iptables -A INPUT -p tcp \

> -m multiport --dports 20:22,25,80,110,143,16501:16800 -j ACCEPT

//一次性开启20,21,22,25,80,110,143,16501到16800所有的端口

3.根据IP地址范围设置规则

# iptables -A INPUT -p tcp --dport 22 \

> -m iprange --src-range 192.168.4.10-192.168.4.20 -j ACCEPT

三.配置SNAT实现共享上网

1.确保proxy主机开启了路由转发功能

# echo 1 > /proc/sys/net/ipv4/ip\_forward //开启路由转发

2.设置防火墙规则，实现SNAT地址转换

# iptables -t nat -A POSTROUTING \

> -s 192.168.4.0/24 -p tcp --dport 80 -j SNAT --to-source 192.168.2.5

3.登陆web主机查看日志

# tail /var/log/httpd/access\_log

.. ..

192.168.2.5 - - [12/Aug/2018:17:57:10 +0800] "GET / HTTP/1.1" 200 27 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)"

4.扩展知识，对于proxy外网IP不固定的情况可以执行下面的地址伪装，动态伪装IP。

# iptables -t nat -A POSTROUTING \

> -s 192.168.4.0/24 -p tcp --dport 80 -j MASQUERADE

最后，所有iptables规则都是临时规则，如果需要永久保留规则需要执行如下命令:

# service iptables save

===========================================================

Day 05 监控概述 、 Zabbix基础 、 Zabbix监控服务

一.监控概述

· 监控的目的

1.报告系统运行状况

- 每一部分必须同时监控

- 内容暴扣吞吐量,反应时间,使用率等

2.提前发现问题

- 进行服务器性能调整前,知道调整什么

- 找出系统的瓶颈在什么地方

· 监控的资源类别

1.公开数据

- Web,ftp,ssh,数据库等应用服务

- TCP或UDP端口

2.私有数据

- CPU,内存,磁盘,网卡流量等使用信息

- 用户,进程等运行信息

· 被监控主机需安装agent(每一台)

· agent a

· 监控软件

1.系统监控命令

· ps · ifconfig

· uptime · netstat或ss

· free · ping

· swapon -s · traceroute

· df -h · iostat

2.自动化监控系统

· Cacti

- 基于SNMP协议的监控软件,拥有强大的绘图能力

· Nagios

- 基于Agent监控,强大的状态检查与报警机制

- 插件极多,自己写监控脚本潜入到Nagios非常方便

· Zabbix

- 基于多种监控机制,支持分布式监控(树型结构)

3.部署监控服务器ZAbbix Server

1) 源码安装Zabbix Server

· 先安装相关依赖包

# yum -y install net-snmp-devel \

>curl-devel

# yum -y install \

> libevent-devel-2.0.21-4.el7.x86\_64.rpm //在lnmp-soft中

# tar -xf zabbix-3.4.4.tar.gz

# cd zabbix-3.4.4/

# ./configure --enable-server \

> --enable-proxy --enable-agent --with-mysql=/usr/bin/mysql\_config \

> --with-net-snmp --with-libcurl

// --enable-server安装部署zabbix服务器端软件

// --enable-agent安装部署zabbix被监控端软件

// --enable-proxy安装部署zabbix代理相关软件

// --with-mysql配置mysql\_config路径

// --with-net-snmp允许zabbix通过snmp协议监控其他设备

// --with-libcurl安装相关curl库文件，这样zabbix就可以通过curl连接http等服务，测试被监控主机服务的状态

# make && make install

4.初始化Zabbix,配置监控服务器

1) 创建数据库，上线Zabbix的Web页面

# mysql

mysql> create database zabbix character set utf8;

//创建数据库，支持中文字符集

mysql> grant all on zabbix.\* to zabbix@'localhost' identified by 'zabbix';

//创建可以访问数据库的账户与密码

# cd lnmp\_soft/zabbix-3.4.4/database/mysql/

# mysql -uzabbix -pzabbix zabbix < schema.sql

# mysql -uzabbix -pzabbix zabbix < images.sql

# mysql -uzabbix -pzabbix zabbix < data.sql

//刚刚创建是空数据库，zabbix源码包目录下，有提前准备好的数据

//使用mysql导入这些数据即可（注意导入顺序）

2) 上线Zabbix的Web页面

# cd lnmp\_soft/zabbix-3.4.4/frontends/php/

# cp -r \* /usr/local/nginx/html/

# chmod -R 777 /usr/local/nginx/html/\*

3) 修改Zabbix\_server配置文件，设置数据库相关参数，启动Zabbix\_server服务

# vim /usr/local/etc/zabbix\_server.conf

DBHost=localhost

//数据库主机，默认该行被注释

DBName=zabbix

//设置数据库名称

DBUser=zabbix

//设置数据库账户

DBPassword=zabbix

//设置数据库密码，默认该行被注释

LogFile=/tmp/zabbix\_server.log

//设置日志，仅查看以下即可

[root@zabbixserver ~]# useradd -s /sbin/nologin zabbix

//不创建用户无法启动服务

# zabbix\_server //启动服务

4) 确认连接状态，端口10051

# ss -ntulp | grep zabbix\_server

· 如果是因为配置文件不对，导致服务无法启动时，不要重复执行zabbix\_server,一定要先使用killall zabbix\_server关闭服务后，再重新启动一次

5) 浏览器访问Zabbix\_server服务器的Web页面

# firefox http://192.168.2.5/index.php

//第一次访问，初始化PHP页面会检查计算机环境是否满足要求，如果不满足会给出修改建议

//默认会提示PHP的配置不满足环境要求，需要修改PHP配置文件

根据错误提示，修改PHP配置文件，满足Zabbix\_server的Web环境要求,php-bcmath和php-mbstring都在lnmp\_soft目录下有提供软件包

# yum -y install php-gd php-xml

# yum install php-bcmath-5.4.16-42.el7.x86\_64.rpm

# yum install php-mbstring-5.4.16-42.el7.x86\_64.rpm

# vim /etc/php.ini

date.timezone = Asia/Shanghai //设置时区

max\_execution\_time = 300 //最大执行时间，秒

post\_max\_size = 32M //POST数据最大容量

max\_input\_time = 300 //服务器接收数据的时间限制

memory\_limit = 128M //内存容量限制

# systemctl restart php-fpm

修改完PHP配置文件后，再次使用浏览器访问服务器，则会开始初始化

数据库端口:3306

数据库密码:zabbix

· 配置完成后,使用用户admin和密码zabbix登录,设置语言环境为中文

5.配置监控客户端

· 修改Zabbix\_agent配置文件，启动Zabbix\_agent服务,使服务器监控自身

# vim /usr/local/etc/zabbix\_agentd.conf

Server=127.0.0.1,192.168.2.5 //允许哪些主机监控本机

ServerActive=127.0.0.1,192.168.2.5 //允许哪些主机通过主动模式监控本机

Hostname=zabbixserver //设置本机主机名

LogFile=/tmp/zabbix\_agentd.log //设置日志文件

UnsafeUserParameters=1 //是否允许自定义key

# zabbix\_agentd //启动监控agent

· 配置web1,web2被监控服务器

1) 源码编译安装zabbbix,修改配置

# cd /root/myTools/lnmp-soft/

# tar -xf zabbix[Tab]

# cd zabbix[Tab]

# ./configure --enable-agent

# make && make install

# vim /usr/local/etc/zabbix\_agentd.conf

Logfile=/tmp/zabbix\_agentd.log

Hostname=zabbixclient\_webx[1-2] //被监控端自己的主机名

Server=127.0.0.1,192.168.2.5 //谁可以监控本机（被动监控模式）

ActiveServer=127.0.0.1,192.168.2.5 //谁可以监控本机（主动监控模式）

EnableRemoteCommands=1 //监控异常后，是否允许服务器远程过来执行命令，如重启某个服务

UnsafeUserParameters=1 //是否允许自定义key监控

# zabbix\_agentd //启动agent服务

2) 拷贝启动脚本（非必须操作，可选做），有启动脚本可以方便管理服务，启动与关闭服务。启动脚本位于zabbix源码目录下。

# cd misc/init.d/fedora/core

# cp zabbix\_agentd /etc/init.d/

# /etc/init.d/zabbix\_agentd start

# /etc/init.d/zabbix\_agentd stop

# /etc/init.d/zabbix\_agentd status

# /etc/init.d/zabbix\_agentd restart

6.自定义Zabbix监控项目

1) 步骤:

· 创建自定义key

· 创建监控项目

· 创建监控图形

· 将监控模板关联到主机

2) 监控主机创建自定义key（在192.168.2.100操作）

# vim /usr/local/etc/zabbix\_agentd.conf

Include=/usr/local/etc/zabbix\_agentd.conf.d/ //加载配置文件目录

# cd /usr/local/etc/zabbix\_agentd.conf.d/

# vim count.line.passwd

UserParameter=count.line.passwd,wc -l /etc/passwd | awk ' {print $1} '

//自定义key语法格式:

//UserParameter=自定义key名称,命令

3) 测试自定义key是否正常工作

# killall zabbix\_agentd

# zabbix\_agentd //重启agent服务

# zabbix\_get -s 127.0.0.1 -k count.line.passwd

21

如zabbix\_get命令执行错误，提示Check access restrictions in Zabbix agent configuration，则需要检查agent配置文件是否正确：

# vim /usr/local/etc/zabbix\_agentd.conf

Server=127.0.0.1,192.168.2.5

ServerActive=127.0.0.1,192.168.2.5

7.创建监控模板

1) 添加监控模板

2) 创建应用

3) 创建监控项目item（监控项）

4) 创建图形

5) 查看监控数据图形