shell脚本

认识shell脚本:

Vim报错解决:

{

产生交换文件(缓存文件)

删除缓存文件 路径下后缀名.swp文件

}

shell脚本基础:

脚本:一个可以执行的文件,运行可以实现某种功能

文件颜色:绿色:表示shell脚本

{

vim 路径(后跟要创建的脚本文件) #创建shell脚本

chmod +x /脚本的绝对路径 #增加文件运行权限

脚本的绝对路径 #绝对路径运行脚本

}

规范shell脚本的一般组成

#! 环境声明(以下代码)

# 注释文本

可执行代码

重定向基础:

{

>:覆盖重定向,只收集前面命令的正确输出

2>: 只收集前面命令的错误输出

&>: 正确与错误的输出都收集

}

都可以把输出记录到

脚本符号运用:

{

单引号＇＇:取消符号的特殊意义

反撇号``:适用于小型脚本 #将命令的输出直接参与下一次命令的运行

$():适用于大型脚本书写 #将命令的输出直接参与下一次命令的运行

}

黑洞设备:dev/null

变量:为了增强脚本适应环境的变化以及需求的变化所使用的(会变化的量)

{ (以不变的名称,存储可以变化的值)

格式:(变量名=存储的值,例:a=1000)

{

$[变量名] #调用变量

}

传递参数 :(由命令行传参到)

{

交互式传递参数:(交互式命令行传递参数给脚本的代码)

{

read

-p ‘屏幕提示信息’ #

1. 产生交互
2. 记录用户在键盘上的所有输入
3. 将记录的内容交由变量存储

}

}

定义/赋值变量:

{

设置变量的注意事项:

{

1.等号两边不能有空格

}

基本格式:

{

1. 引用变量值:$变量名
2. 查看变量值:echo $变量名/echo ${变量名}(其后可与其他参数拼接)

}

}

变量的分类:

{

环境变量:变量名大写 由系统定义并且赋值完成

{

USER=当前登录的用户身份

}

位置变量:有系统定义并且赋值完成(非交互式:在执行脚本时提供的命令行参数)

{

表示$n,n的序号

}

预定义变量:由系统定义

{

$# : 已加载的位置变量的个数 ,判断用户是否输入命令行参数

$? : 程序退出后的状态值,0表示正常(真),其他值为异常(假)

{

}

}

自定义变量:用户自助设置,修改及使用

}

}

条件测试及选择:

{

- [ 测试表达式 ] #每一部分之间都要有空格

常见的:

{

-e:存在即为真

-d:存在并且必须为目录,才为真

-f:存在并且必须为文件,才为真

}

比较整数大小

{

-gt:大于

-ge:大于等于

-eq:等于

-ne:不等于

-lt:小于

-le:小于等于

}

字符串比对  
 {

==:一致为真

!=:不一致为真

}

判断命令:(if选择)

{

If单分支处理

If双分支处理

{

If [条件测试1];then

命令序列xx

elif [条件测试2];then

命令序列zz

else

命令序列yy

fi

}

}

列表式循环场景(for循环)

{

给定一批对象,反复执行类似操作

for循环:

{

for 变量名 in 值列表

do

要执行的命令行

done

}

列表值

{

}

造数工具:制造连续范围的数字 [{起始值..结束值}]

延时命令:sleep [多少秒] #单位为秒

}

echo [‘参数’] >&2 #将输出变成错误输出

exit ($?的输出 [数字]) #(脚本退出返回的状态值)

}

搭建ftp服务器

搭建FTP服务

{

FTP服务:传输数据

FTP协议:文件传输协议

实现FTP服务软件:vsftpd

实现:

{

1.安装vsftpd软件(yum -y install vsftpd)

2.重启程序(systemctl restart vsftpd)

}

用户登录路径，

local\_root 针对系统用户 local\_root=/var/ftp/

锁定用户到各自目录为其根目录 chroot\_local\_user=YES

针对匿名用户anon\_root=/var/www/html allow\_writeable\_chroot=YES 用户配置目录 user\_config\_dir=/etc/vsftpd/userconfig

}

**互联网常见的协议**:

{

1. http:超文本传输协议 #默认端口:80
2. https:安全的超文本传输协议 #默认端口:443
3. ftp:文件传输协议 #默认端口:21
4. Tftp:简单的文件传输协议 #默认端口:69
5. DNS:域名解析协议 #默认端口:53
6. telnet:远程管理协议{交换机|路由器(明文传输)} #默认端口:23
7. ssh:远程管理协议{服务器(加密传输)} #默认端口:22
8. SMTP:邮件传输协议(用户发邮件) #默认端口:25
9. pop3:邮件传输协议(用户收邮件) #默认端口:110
10. Snmp:简单的网络管理协议 #默认端口:161

}

}

**防火墙策略管理**

{

作用:隔离 (众多的策略,允许外访,严格控制内访)

分类:硬件防火墙|软件防火墙

Firewall服务基础

{

系统服务:firewalld

管理工具:

{

-cmd[非交互式命令行界面]

-config[交互式 图形界面]

}

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:仅允许访问主机的ssh dhcp ping等少数几个服务

-trusted:允许任何访问

-block:阻塞任何来访请求(明确拒绝,有回应客户端)

-drop:丢弃任何来访的数据包(直接丢弃,没有回应.节省服务端资源)

}

根据数据包的源IP地址

数据包:源IP地址,目标IP地址,数据

防火墙判定原则:

{

1.查看数据包的源IP地址,然后查看自己所有的区域,哪个区域中有该源IP地址的规则,则进入该区域

2.进入默认区域(public)

}

防火墙默认区域的修改:

{

firewall-cmd --set-default-zone=[安全区域] #修改默认区域

firewall-cmd --get-default-zone #查看默认区域

}

防火墙区域添加规则:

{

firewall-cmd --zone=public --list-all #查看区域规则

firewall-cmd --zone=public --add-service=[协议名称] #添加协议

}

防火墙规则的永久设置:

{

永久(permanent)

跟在cmd后边 : firewall-cmd [--permanent] --zone=public ...

重新加载防火墙配置文件 : firewall-cmd --reload

}

单独拒绝访问

{

将拒绝的IP地址写入block:

--zone=block --add-source=[拒绝的IP地址] #拒绝某IP访问

--zone=block --list-all #查看区域规则

}

删除规则:

{

--zone=[区域] --remove-[source|service]=[协议或IP地址]

--zone=[区域] --reload #重新加载规则

}

}

端口:(程序或协议或服务的编号)

利用root可以改变端口,而且一个程序可以具备多个端口

监听端口:

对客户端（个人机器）所操作的一种信息记录

实现本机的端口映射

{

本地应用的端口重定向(端口1--> 端口2)

--zone=[区域] --add-forward-port=[port=(原端口):proto=tcp:toport=(映射端口)]

}

}

交换分区

准备交换分区:(虚拟内存)

{

缓解真实物理内存的压力

相当于虚拟内存

-当物理内存不够用是.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

配置方法:

{

1.利用fdisk进行分区,划分两个主分区分别为2G

2.格式化为交换文件系统

{

Mkswap [分区路径] #专业格式化交换文件系统

Blkid [分区路径] #查看文件系统类型

}

1. 启用与停用交换分区(把交换分区添加到虚拟内存中)

{

Swapon [分区路径] #启用交换分区

Swapon -s #查看交换空间的成员信息

Swapoff [分区路径] #停用交换分区

}

1. 开机自动启用交换分区

{

vim /etc/fstab #编辑配置文件

{

[分区路径] swap swap defaults 0 0

}

swapon -a #专门检测/dev/fstab交换分区

}

}

}

补充内容07-09

触发挂载

{

准备:划分一个2G的主分区

{

fdisk [磁盘路径] #划分分区

partprobe #刷新分区表

mkfs.xfs [分区路径] #格式化文件系统

}

1.安装触发挂载的软件包

]# yum -y install autofs

2.重启autofs

]# systemctl restart autofs

]# systemctl enable autofs

]# systemctl status autofs #查看程序是否运行

1. 完成对[分区路径]触发挂载的配置

主配置文件:定义监控目录/etc/auto.master

格式:监控目录(必须手动创建) 挂载配置文件路径

挂载配置文件:

}

**共享存储**

iscsi共享存储:

{

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的构成:

{

backstore,后端存储

对应到服务器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

}

磁盘组名称规范:

{

网际iscsi接口 默认端口为3260

ISCSIQualified Name 名称规范(iqn)

Iqn.yyyy-mm.倒序域名:自定义标识

用来识别targt磁盘组,也用来识别客户机身份

}

配置方法:

{

1. 修改防火墙默认区域
   1. ]#firewall-cmd --set-default-zone=trusted #设置默认区域
   2. ]#firewall-cmd --get-default-zone #查看默认区域
2. 划分一个5G的主分区
   1. ]#fdisk /dev/vdb #划分分区
   2. ]#lsblk #查看分区
3. 虚拟机server服务端

准备:

{

安装软件包targetcli

]#yum -y install targetcli

运行targetcli命令进行配置

]#targetcli

/>ls #查看相关配置

}

1. 建立后端存储backstore

a) backstores/block create name=nsd01 dev=/dev/vdb1

b) 后端存储 /模块化 创建 存储名称 后端存储路径

1. 建立target磁盘组
   1. />iscsi create {iqn.201\*-0\*.com.example:[自定义名称]}
   2. 创建磁盘组 名称规定[iqn.年-月.倒序域名:自定义名称]
2. 进行lun关联

iscsi/{磁盘组}/tpg1/luns create /backstores/block/[存储名称]

ls #查看相关配置

1. 设置访问控制,客户端声称的名字,符合iqn名称规范

iscsi/{存储名称}/tpg1/acls create [访问主机的名称(符合iqn)]

1. 设置监听的IP地址

iscsi/{存储名称}/tpg1/portals create

6.重启服务(重启程序)

]#systemctl restart target #重启服务

]#systemctl enable target #设置开启自启

1. 虚拟机desktop客户端
2. 安装软件包(客户端软件)
   1. 所需软件包:iscsi-initiator-utils
   2. yum仓库可以补全软件名:

前提1:当前系统没有安装此软件

前提2.必须具备yum缓存(yum repolist可以直接生成缓存)

]#yum repolist #

]#rpm -q iscsi-initiator-utils

1. 设置客户端生成的名字
   1. vim /etc/iscsi/initiatorname.iscsi
      1. initiatorname=iqn.2019-07.com.example:desktop
2. 客户端刷新声称名字的服务
   1. ]#systemctl restart iscsid #第一种方法
   2. ]#systemctl daemon-reload #报错后第二种方法

]#systemctl restart iscsid

1. 发现iscsi共享存储
   1. iscsiadm --mode discoverydb --type sendtargets --portal [IP地址] --discover
   2. iscsiadm -m discovery -t st -p [服务器地址(:端口)]
2. 连接(加载)iscsi共享磁盘

]#systemctl restart iscsi 重启iscsi服务

]#systemctl enable iscsi #设置iscsi为开机启动

]#lsblk #查看磁盘空间

}

}

**数据库**

数据库服务基础:

{

什么是数据库:存放数据的仓库

数据库中有众多的库,在每个库中有众多的表格

常见的关系型数据库管理系统:

{

-微软的SQL server

-IBM的DB2

-甲骨文的Oracle ,MySQL

-社会开源版 MariaDB

}

database:数据库

部署mariadb数据服务器

RHEL7中的MariaDB相关包

-mariadc-server :提供服务端有关的系统程序

安装步骤:

{

1. yum -y install mariadb-server #安装数据库软件
2. systemctl restart mariadb #重启数据库软件
3. systemvtl enable mariadb #数据库软件设置为开机自启

}

部署方法:

{

进入mariadb : mysql #进入数据库配置系统(没有密码时登录)

查看库:[(none)]>show databases; #查看数据库内所含库

新建库:[(none)]>create database [库名]; #新建库

删除库:[(none)]>drop database [库名]; #删除(丢弃)库

切换到库:[(none)]>use [库名]; #切换到一个库里

退出数据库:exit #退出数据库

}

在Linux,命令行为数据库管理员设置密码

{

数据库管理员为root

针对数据库有最高权限用户

存放在mysql库user表格中

Linux系统管理员为root

针对系统有最高权限用户

存放在/etc/passwd目录中

为数据库账号修改密码:

]#mysqladmin [-u用户名] [-p旧密码(没有可不填)] password ‘新密码’

用户登录数据库:

]#mysql -u [用户名] -p[密码] 例:mysql -u root -p123 #非交互式登录

]#mysql -u [用户名] -p

Enter password: #交互式登录

}

表格:

{

增(insert)

{

insert [表名] values(‘序号’,‘第二列内容’,‘第三列内容’...)

}

删(drop库,delete表中)

改(update)

查(select)

{

1. ]#mysql -u root -p[密码] #登录到数据库
2. ]>use [库名]; #切换到库
3. ]>show tables; #查看库所含的表格
4. ]>select [表字段] from [表名]; #查看表所有字段及所有记录内容
5. ]>select [表字段] [from] [库名.(绝对路径)]表名 where [选项=＇字段＇];

查询表结构:desc 表名; #适用于大型表格

Count

多项查询:

select [\*] from [表名,表名] where 选项=条件 and 选项=条件 ......;

}

表记录(表格的内容) 表字段(表示有纵列)

}

导入与恢复到数据库

{

1. Mysql [-u用户名] [-p密码] 数据库名 < 备份文件.sql
2. ]#wget 备份文件地址 #下载数据库备份文件
3. ]#mysql -u root -p[密码] #登录到数据库
4. ]>show tables; #查看库所含的表格

｝

数据库的授权:

{

用户授权设置 ]> #交互式操作

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’

当用户从客户机地址登录,输入密码‘#’,将会获得(#)库中所有表格的查询权限

}

禁止空密码用户访问mariadb数据库:

{

1. 查看密码为空的用户

]>select user,host,password from user where password=‘';

1. 删除空密码的记录

]>delete from user where password='';

1. 刷新user表记录

]> flush privileges;

1. 验证登录失败:

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

}

}

**NFS服务**

配置NFS共享:

{

NFS共享概述:

{

用途:为客户机提供共享使用的文件夹

协议:NFS(2049端口,TCP/UDP),RPC(111端口,TCP/UDP)

}

服务端构建NFS服务:

{

1. 安装软件包:
   1. 安装软件包:yum -y install nfs-utils
   2. 检测安装包是否安装成功:rpm -q nfs-utils
2. 配置共享:

{

exports配置文件解析;

修改 /etc/exprots

-文件夹路径 客户机地址[可为\*](权限) 客户机地址[可为\*](权限)..

exportfs -s #重新加载更新过的nfs配置

}

重启服务(重启程序)

]#systemctl restart nfs-server

}

客户端配置服务(访问共享文件夹):

{

1. 挂载共享目录:
   1. ]#mkdir /mnt/a /mnt/b #创建挂载点
   2. ]#showmount -e [服务端IP地址] #查看服务端共享内容
   3. ]#mount [服务端IP地址]:[共享路径] [挂载点] #挂载服务端
2. 开机自动挂载:
   1. \_netdev:声明网络设备

指明挂载本设备需要有网络参数再进行挂载

]#vim /etc/fstab

[服务端IP]:[共享路径] [挂载路径] defaults,\_netdev 0 0

]#mount -a #刷新挂载点

]#df -ah #显示所有挂载项

}

}

**Web服务**

HTTP服务基础

{

Web的基本概念:(提供网页内容)

实现web服务的软件 : httpd Nginx Tomcat

实现web通信的协议 http(超文本传输协议)

基于B/S架构的网页服务

服务端提供网页

浏览器下载并下载网页

HTML,超文本标记语言,

HTTP,超文本传输协议.(80端口)

搭建web服务:

{

软件包:httpd(Apache组织) #专门实现Web服务的软件

使用:

{

1. 安装httpd软件[yum -y install httpd]
2. 重启程序(重启服务)[systemctl restart httpd]
3. 本机测试访问[firefox 172.25.0.11]
4. 书写自己的页面文件

{

默认存放文件路径:/var/www/html

默认首页文件的名称:index.html

a)

}

}

}

主配置文件配置字段:

{

-Listen:监听地址:端口(80)

-ServerName:本站点注册的DNS名称(空缺)

-Doc:网页文件根目录(/var/www/html)

新建路径:mkdir /var/www/myweb

echo ‘<h1>我 是 myweb’ > /var/www/myweb/myuweb.html

更改路径:vim /etc/httpd/conf/httpd.conf #进入配置文件

/Doc -> DocumentRoot “/var/www/myweb”#查询跟目录的位置并更改

重启服务:]#systemctl restart httpd

测试更改:firefox 172.25.0.11

-DirectoryIndex:起始页/首页文件名(index.html)

虚拟机classroom:

DNS服务器的解析:将域名解析为IP地址

nslookup [域名] #解析域名的IP地址

只有3个 :

server0.example.com-----------172.25.0.11

www0.example.com--------------172.25.0.11

webapp0.example.com-----------172.25.0.11

}

网络路径与实际服务路径

客户端:firefox [http://172.25.0.11:80(浏览器默认添加http://和:80)](http://172.25.0.11:80(浏览器默认添加http:/和:80)) -->

服务端172.25.0.11 -->httpd 80 --> DocumentRoot /var/www/html --> index.html

虚拟Web主机:

{

由同一台服务器提供多个web站点

区分方式:

-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

修改配置文件

{

配置文件路径:

1. -/etc/httpd/conf/httpd.conf #主配置文件
2. -/etc/httpd/conf.d/\*.conf #调用配置文件

为每个虚拟站点添加配置

{

<VirtualHost \*:端口> #开头声明

Servername 此站点的DNS名称

DocumentRoot 此站点的网页根目录

</VirtualHost> #结尾

}

}

虚拟web主机的优先级:由上到下依次匹配,匹配即停止

一旦使用虚拟web主机功能,所有的网站都必须利用虚拟主机实现

}

网页访问权限:

{

Web访问控制,针对存放网页文件的目录

子目录默认继承父目录的Web访问控制

<Directory />

Require all denied #拒绝所有人访问

</Directory>

<Directory "/var/www">

Require all granted #允许所有人访问

</Directory>

案例:自定义Web的网页文件根目录

1.实现server0.example.com访问网页文件根目录修改为/webroot

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

<VirtualHost \*:80>

ServerName server0.example.com

DocumentRoot /webroot

</VirtualHost>

]# mkdir /webroot

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

2.修改访问控制

]# vim /etc/httpd/conf/httpd.conf

]# vim /etc/httpd/conf.d/nsd02.conf

<Directory "/webroot"> #针对/webroot路径

Require all granted #允许所有人访问

</Directory>

]# systemctl restart httpd #重启web服务

3.SELinux策略

]# setenforce 0 #临时更改SELinux策略

]# getenforce #查看当前SELinux策略

Permissive

]# vim /etc/selinux/config #永久更改SELinux策略

SELINUX=permissive #更改配置文件

客户机地址限制:

{

使用<Directory>配置区段

每个文件夹自动继承其父目录的ACL访问权限

除非针对子目录有明确设置

禁止任何客户机访问:Require all denied

允许任何客户机访问:Require all granted

仅允许部分客户机访问:Require ip IP地址或网段...

}

}

部署动态网站:

{

静态网站概述:

服务端原始页面 = 浏览器访问到的页面

-由web服务软件处理所有请求

-文本(txt/html),图片(jpg/png)等静态资源

动态网站概述:

服务端原始页面 ≠ 浏览器访问到的页面

-由web服务软件接受请求,动态程序转后端模块处理

-PHP网页,Python网页,JSP网页......

配置(部署测试页webinfo.wsgi,调整首页跳转)

{

部署Python页面文件

]#cat /etc/httpd/conf.d/abc01.conf

]#cd /var/www/abc03/

]#wget <http://......webinfo.wsgi>

页面跳转(页面别名,地址重写)方便用户的访问

当用户访问 [域名] 将 [动态页面文件]页面进行呈现

修改配置文件:

vim 配置文件路径(例:vim /etc/httpd/conf.d/abc01.conf)

格式: alias 网络路径 实际文件

例: alias / /var/www/abx03/webinfo.wsgi

翻译Python页面代码

安装软件:(httpd,mod\_wsgi)

]#Yum -y install mod\_wsgi

编辑配置文件:

WsgiScriptalias / /var/www.abc03/webinfo.wsgi

重启服务:

]#systemctl restart httpd

}

修改虚拟web主机的端口:

修改配置文件:Listen [监听端口号]

<VirtualHost \*:[监听端口号]>

}

}

**安全的web服务**

数字证书基础

{

PKI公钥基础设施:

{

Public key Infrastructure公钥基本设施

-公钥:主要用来加密数据

-私钥:主要用来解密数据(与相应的公钥匹配)

-数字证书:证明拥有者的合法性/权威性

(单位名称,有效期,公钥,颁发机构及签名)

-Certificate authority 数字证书授权中心:

负责证书的申请/审核颁发/鉴定撤销等管理工作

}

补充:vim末行模式 输入set nu可以显示行号

Vim命令模式 输入G(大写) 切换到全文的末行

部署方法:

{

1.部署网站证书:

{

证书路径: /etc/pki/tls/certs/

下载网站证书: wget [网络路径\*.crt]

}

1. 部署网站的根证书:(工商局的信息)

根证书路径:/etc/pki/tls/certs/

下载网站跟证书: wget [网络路径example-ca.crt]

1. 部署解密数据的私钥:

私钥存放路径:

下载私钥文件:wget [网络路径\*.key]

1. 安装可以进行加密支持的软件mod\_ssl

]#yum -y install mod\_ssl

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

59 DocumentRoot "/var/www/abc02"

60 ServerName www0.example.com:443

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

#指定网站证书

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

#指定解密秘钥

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

#指定网站根证书

]#systemctl restart httpd #重启服务(程序)

5.我以了解可能的风险==>添加信任==>确定安全例外

}

}

**补充07-12**

Samba共享服务:(Windows与Linux跨平台的共享)

{

运用smb(用户验证通信协议)cifs(samba独有文件系统)

程序名(服务名):smb

使用共享服务:

{

1. 安装samba服务:

虚拟机server:]#yum -y install samba

1. 创建samba共享账号:

建立在本地用户的基础上

采用独立的密码,与用户登录系统的密码不同

]#useradd [用户名] #创建用户

]#pdbedit -a [用户名] #添加为samba账号

<Pdbedit的用法:

-a [用户名] #添加用户

-L #查看所有的samba账号

-x [用户名] #删除samba用户>

1. 创建共享目录及配置共享:

服务端(server):

创建共享的目录 : ]#mkdir [共享的目录]

创建文件测试 : ]#echo haha > [共享目录]/1.txt

修改配置文件 : ]#vim /etc/samba/smb.conf

全文末行:[nsd] #设置共享名

Path = [共享的目录] #设置实际共享的路径

重启服务(程序) : ]#systemctl restart smb

1. 修改SELinux策略:布尔值策略(服务功能的开关)

]#getsebool -a | grep samba #查看关于samba布尔值策略

]#setsebool [策略名] on #打开允许samba的策略

]#getsebool -a | grep samba #查看关于samba布尔值策略

客户端(desktop)

安装软件包,支持cifs文件系统 ]#yum -y install cifs-utils

进行挂载访问:

]#mkdir /mnt/smb #创建挂载点

]#mount -o user=smb01,pass=123 //172.25.0.11/nsd /mnt/smb

格式:mount -o user=[用户],pass=[密码] //[服务器IP地址]/[共享名] [挂载点]

开机自动挂载/etc/fstab

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

}

}

**系统安全保护**

SELinux策略保护:

{

SELinux

{

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及以上)中运行

RHEL7基于SELinux体系针对用户，进程

SELinux的运行模式:

{

enforcing(强制)

permissive(宽松)

disabled(彻底禁用)

任何模式切换成disabled模式,都需要经历重启

切换运行模式:

{

临时切换:setenforce 1|0 (0表示宽松模式,1表示强制模式)

固定配置:/etc/selinux/config (需要重启后生效)

{

SELinux=permissive

}

Getenforce 查看当前SELinux的状态

}

}

影响制定用户的bash解释环境

~/.bashrc,每次开启bash终端时生效

影响所有用户的bash解释环境

/etc/bashrc,每次开启bash终端时生效

修改SELinux策略:上下文值(打标签) 布尔值策略 非默认端口

{

查看策略文件:

semanage fcontext -l | {grep [查询的内容]}(可不加,查询所有策略)

修改策略文件:

参照标准目录,重设新目录的属性

chcon [-R(递归修改)] --refsrence=模拟目录 新目录

查找SELinux允许的所有端口

semanage port -l | [grep]

添加SELinux端口

semanage port -a -t httpd\_port\_t -p tcp 8909

添加用法:[-a 创建][-t类型][-p 协议] 端口

}

SELinux常用参数:

{

-a [创建] -t [类型] -p [协议]

-l [查看] fcontext -l [查看策略文件] port -l [查看端口]

}

}

}

**补充07-13**

{

永久设置IP地址 子网掩码 网关地址

网卡配置文件的位置 : /etc/sysconfig/network-scripts/ifcfg-[网卡名]

1.查看识别的网卡名:

]#nmcli connection show

2.非交互式:

]#nmcli connection modify “网卡名” ipv4.method manual/auto ipv4.address ‘IP地址’ connection.autoconnect yes

3.激活网卡:

]#nmcli connection up ‘网卡名’

]#ifconfig | head -5

]#route #查看网关

DNS服务器的地址:

1. 永久配置DNS服务器

修改配置文件:

]#cat /etc/resolv.conf

]#echo ‘nameserver [服务器地址]’> /etc/resolv.conf

]#cat /etc/resolv.conf

]#nslookup server0.example.com

IPv6地址的组成:

Ipv4地址表示

-32个二进制位,点分隔的十进制数

例:172.25.0.11 127.0.0.1

Ipv6地址表示

-128个二进制位,冒号分隔的十六进制数

-每段内连续的前置 0可省略,连续的多个:可简化为::

Ping6 [ipv6地址]

聚合连接:(链路聚合网卡绑定)

{

链路聚合的优势:

-由多块网卡(team-slave)一起组件而成的虚拟网卡,即‘组队’

作用:实现网卡的高可用

-作用1:轮询式(roundrobin)的流量负载均衡

-作用2:热备份(activebackup)连接

1.新建聚合网卡:

Nmcli connection add type team con-name team0 ifname team0 autoconnect yes config ‘{“runner”: {“name”: “activebackup”}}’

格式]#nmcli connection 添加 类型 组队 con-name 配置文件名 ifname 网卡名 autoconnect yes ‘{“运行方式”:{“name”: “activebackup”}}’

2.添加成员:

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

格式Nmcli connection 添加 类型 team-成员 配置文件名 team0-1 网卡名 eth1 主设备 team0

3.Team0配置IP地址:

Nmcli connection modify team0 ipv4.method manual ipv4.address ‘IP地址’ connection.autoconnect yes

1. 激活虚拟网卡:

Nmcli connection up team0

1. 查看 team0的信息

Teamdctl team0 state

}

基础邮件服务:

{

前提:修改IP地址,符合DNS解析结果

查看DNS服务器地址:> cat /etc/resolv.conf

测试解析:]#nslookup server0.example.com

电子邮件服务器的基本功能:

-为用户提供电子邮箱存储空间(用户名@域名)

-处理用户发出的邮件--传递给收件服务器

-处理用户收到的邮件--投递到邮箱

服务端构建邮件服务器

1. 安装postfix软件实现邮件服务]

Rpm -q postfix

1. 修改配置文件

]#Vim /etc/postfix/main.cf

末行模式:set nu #显示行号

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

116 inet\_interfaces = all #在本机所有网卡启用邮件功能

164 mydestination = server0.example.com #判断为本域邮件的依据

1. 重启邮件服务

]#systemctl restart postfix

1. 测试邮件服务

{

使用mail命令发信/收信

\*mail发信操作

-mail -s ‘邮件标题’ -r 发件人 收件人[@收件域]......

交互式发邮件:

]#useradd yg #创建用户

]#useradd xln #创建用户

]# mail-s ‘邮件标题’ -r yg xln #发邮件命令

拉萨的飞机客户 #邮件内容

非交互式发邮件:

]#echo ‘邮件内容’ > mail -s ‘邮件标题’ yg xln

\*mail收信操作

-mail -u 收件人

]#mail -u xln #收邮件命令

>N 1 nsdyg@server0.exampl Sat Jul 13 14:32 18/623

& 1

}

}

Parted分区工具,进行GPT分区模式

{

MBR分区模式

最多有4个主分区,划分3个主分区,1个扩展分区,n个逻辑分区

最大空间支持2.2TB

GPT分区模式:parted分区工具

最多由128个主分区

最大空间支持18EB (1EB=1024PB,1PB=1024TB)

Parted分区:

{

]#parted /dev/vdb #选择分区

d)mkrable gpt #指定分区模式gpt

d)print #输出分区表的信息

d)mkpart #划分新的分区

分区名称? []?Nsd01 #随意起名字

文件系统类型? [ext2]?Xfs #随意写不起决定性作用

起始点? 0

结束点? 2G

忽略/Ignore/放弃/Cancel? Ignore #输入i忽略

d) unit GB #使用GB作为单位显示

d) print #输出分区表的信息

d)mkpart #划分新的分区

分区名称? []?Nsd01 #随意起名字

文件系统类型? [ext2]?Xfs #随意写不起决定性作用

起始点? 2G

结束点? 4G

d)print #输出分区表的信息

}

}

}