**《数据库与脚本技术》大纲**

**课程简介**

1. **课程简介：学习数据库应用技术的意义和价值**

**数据库应用技术**是云架构方向很重要的课程。

日常的Linux系统运维工作中必不可少的要做mysql数据库方面的维护工作，因此作为一名合适的云架构工程师要掌握mysql的维护知识，以胜任公司数据库方面的工作。

同时日常运维一个重要的工具就是shell脚本，如果不会写shell脚本，那么你就不算一个合格的运维人员。目前很多单位在招聘Linux系统运维工程师时，shell脚本的编写是必考的项目。有的单位甚至用shell脚本的编写能力来衡量这个Linux系统运维工程师的经验是否丰富。shell脚本写的好，那么你的求职路就会轻松的多。

本课程案例均从企业实际应用中转化而来，学完本课程可达到企业数据库的日常维护和自动化运维。如：mysql的安全配置、备份恢复及复制和自动化部署、自动化监控、自动化备份、自动化文件处理等。

1. **知识目标**
2. 掌握mysql数据库的安装
3. 掌握mysql数据库的表的管理
4. 掌握mysql数据库安全管理
5. 掌握mysql数据库备份与恢复
6. 掌握mysql数据库主从复制与读写分享的配置
7. 掌握shell脚本设计的基本语法结构和代码规范.
8. 掌握变量的使用及规范
9. 掌握条件判断语句、循环语句的使用
10. 掌握函数及数组的使用
11. 掌握正则表达式
12. 掌握3大文本处理工具：grep、sed、awk
13. 掌握计划任务的编写
14. 了解信号捕捉及进程管理
15. 掌握expect实现自动交互
16. 掌握邮件报警
17. **能力目标**
18. 速度目标：使用60分种完成mysql数据库的日常管理
    1. 库表的建立
    2. 日志管理
    3. 备份与恢复
19. 速度目标：使用30分种完成mysql安全配置
    1. 删除匿名账号
    2. 给mysql root用户设置口令
    3. 只授予普通用户必须的权限
    4. 除root 外，任何用户不应有mysql 库user 表的存取权限
20. 速度目标：使用180分种完成mysql高级应用
    1. 主从复制
    2. 读写分离
21. 速度目标：用180分钟完成项目—类nagios监控脚本
    1. 监控cpu使用状态
    2. 监控硬盘使用状态
    3. 监控内存使用状态
    4. 监控用户登录数量
    5. 监控端口3306状态
    6. 监控主从复制状态
22. 速度目标：用120分钟完成项目—自动安装和部署lnmp
    1. 自动化安装nginx
    2. 自动安装mysql
    3. 自动化部署php
    4. 自动化部署lnmp
23. 速度目标：用60分钟完成项目—自动化备份脚本
    1. 定时完全备份mysql数据库
    2. 定时增量备份apache日志
24. 速度目标：用60分钟完成项目—日志分析
    1. Nginx日志切割脚本
    2. 分析apache日志，提取攻击ip列表
25. 速度目标：用60分钟完成项目—apache主动防御脚本
    1. 定时分析apache访问日志，截取攻击ip列表
    2. 发现攻击ip，主动加入iptables规则
26. 思路目标

使学生具备良好的数据库操作能力与shell自动化编程思路，在实际工作中能够使用实际问题—程序算法分析—分步转换代码—整合代码来解决问题。

1. **课程教学和学习设计**
2. 教学特色：

每日课程内容都会通过大量案例来解析相关知识

1. 本课程学习规划：
   1. 预习：单元均有相应的预习案例。
   2. 学习：每单元均有对问题的小节讨论、知识要点综述和核心知识原理讲解、相关案例
   3. 复习：
   4. 检测：每单元都设有课后练习题，以强化学生对知识的记忆
2. **参考资料**
3. **课程目录**

[第一单元 数据库概述 5](#_Toc452536148)

[第二单元 mysql数据库中表的管理 12](#_Toc452536149)

[第三单元 mysql数据库的高级管理 15](#_Toc452536150)

[第四单元 mysql数据库的主从复制集群应用 28](#_Toc452536151)

[第五单元 高性能mysql读写分离的实现 33](#_Toc452536152)

[第六单元 mysql优化与常见故障排查 40](#_Toc452536153)

[第七单元 shell脚本概述 49](#_Toc452536154)

[第八单元 shell脚本判断语句 57](#_Toc452536155)

[第九单元 shell脚本循环语句 71](#_Toc452536156)

[第十单元 正则表达式 78](#_Toc452536157)

[第十一单元 awk文本处理工具 86](#_Toc452536158)

[第十二单元 运用expect实现自动交互 89](#_Toc452536159)

[第十三单元 Linux作业调度机制 92](#_Toc452536160)

[第十四单元 联鑫科技公司脚本应用案例 98](#_Toc452536161)

[第十五单元 中联科技公司网站数据分析案例 106](#_Toc452536162)

[第十六单元 新晨科技公司邮件报警案例 112](#_Toc452536163)

[第十七单元 海云捷讯公司ssh批量部署服务案例 117](#_Toc452536164)

[第十八单元 瑞友科技公司apache虚拟主机的自动创建案例 127](#_Toc452536165)

[第十九单元 网融众信公司全自动安装tomcat案例 131](#_Toc452536166)

[第二十单元 网际天娇公司iptables白名单与黑名单案例 140](#_Toc452536167)

1. **课程具体单元**
2. 数据库概述
   1. **数据库简介**
      1. **什么是数据库**
      2. **关系型数据库**
      3. **MYSQL、ORACLE、SQL SERVER功能和应用场合**
      4. **数据库服务器、数据库、表与记录的关系**
      5. **掌握数据库存储引擎种类**
   2. **Linux平台下安装与配置mysql**
      1. **rpm安装**
      2. **源码包编译参数与安装**
      3. **管理mysql服务**
      4. **设置初始密码**
   3. **mysql数据库的管理**
      1. **登陆mysql**
      2. **掌握mysql数据库中库的建立与删除**
      3. **掌握mysql数据库存储引擎**
      4. **掌握mysql数据库中表的字段类型**
      5. 灵活运用mysql数据库中表的约束

【考核要求】

1. **掌握什么是数据库?**

数据库就是一个存储数据的仓库。

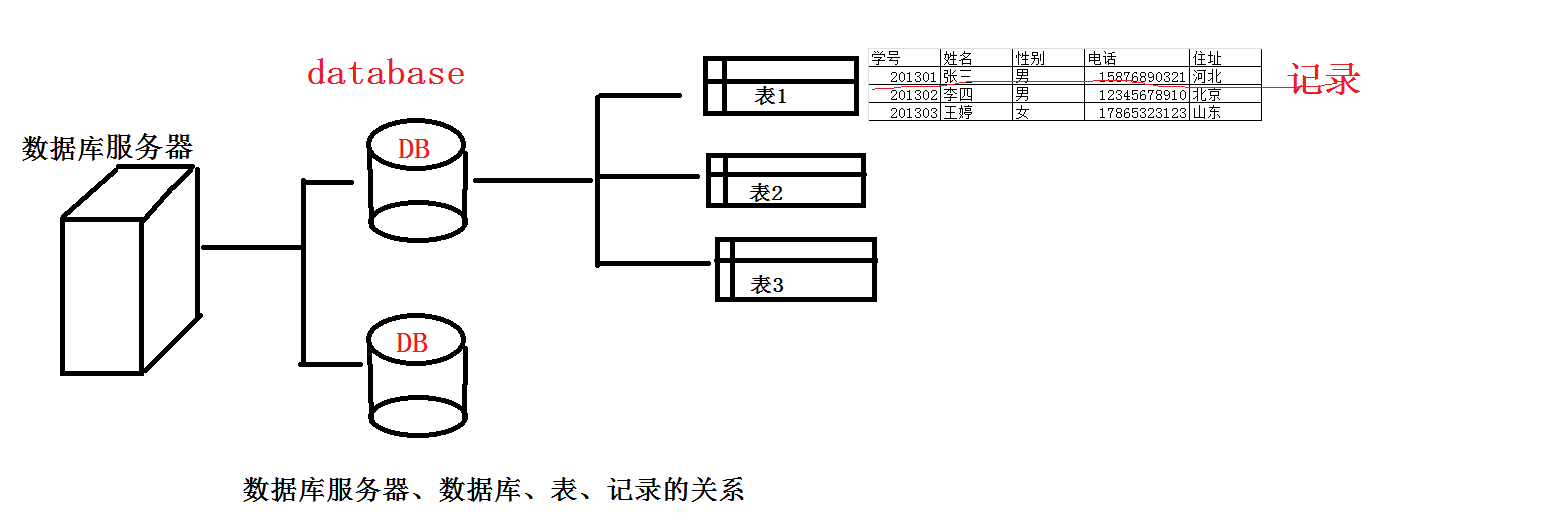
1. **了解MYSQL、ORACLE、SQL SERVER功能和应用场合**

mysql主要用于大型门户，例如搜狗、新浪等，它主要的优势就是开放源代码，因为开放源代码这个数据库是免费的，它现在是甲骨文公司的产品。

oracle主要用于银行、铁路、飞机场等。该数据库功能强大，软件费用高。也是甲骨文公司的产品。

sql server是微软公司的产品，主要应用于大中型企业，如联想、方正等。

1. **了解数据库服务器、数据库、表与记录的关系**



1. **掌握windows平台上安装与配置mysql（参照上课资料中的“mysql5最新版下载及安装.doc”）**
2. **掌握windows平台上管理mysql服务的方法**
   1. net start mysql 开启mysql服务
   2. net stop mysql 关闭mysql服务
3. **掌握linux平台上安装与配置mysql**
   1. rpm安装
      1. rpm –ivh mysql-\*.rpm
      2. rpm –ivh mysql-server-\*.rpm
   2. 源码包安装
      1. yum安装依赖包

yum -y install gcc gcc-c++ cmake ncurses-devel

* + 1. 解压缩tar包

# tar -xvf mysql-5.5.36.tar.gz

# cd mysql-5.5.36

* + 1. 编译安装

#cmake -DCMAKE\_INSTALL\_PREFIX=/usr/local/mysql \

-DDEFAULT\_CHARSET=utf8 \

-DDEFAULT\_COLLATION=utf8\_general\_ci \

-DWITH\_EXTRA\_CHARSETS:STRING=all \

-DWITH\_INNOBASE\_STORAGE\_ENGINE=1 \

-DWITH\_READLINE=1 \

-DENABLED\_LOCAL\_INFILE=1 \

-DMYSQL\_USER=mysql \

-DMYSQL\_TCP\_PORT=3306

#make && make install

常用编译参数说明：

-DCMAKE\_INSTALL\_PREFIX= 指向mysql安装目录，官方推荐设为/usr

-DINSTALL\_SBINDIR=sbin 指向可执行文件目录（prefix/sbin）

-DMYSQL\_DATADIR=/var/lib/mysql 指向mysql数据文件目录（/var/lib/mysql）

-DSYSCONFDIR=/etc/mysql 指向mysql配置文件目录（/etc/mysql）

-DINSTALL\_PLUGINDIR=lib/mysql/plugin 指向插件目录（prefix/lib/mysql/plugin）

-DINSTALL\_LIBDIR=lib/mysql 指向对象代码库目录（prefix/lib/mysql）

-DINSTALL\_INCLUDEDIR=include/mysql 指向头文件目录（prefix/include/mysql）

-DWITH\_<ENGINE>\_STORAGE\_ENGINE=1启用某个引擎的支持，如：

-DWITH\_INNOBASE\_STORAGE\_ENGINE=1

-DWITH\_ARCHIVE\_STORAGE\_ENGINE=1

-DWITH\_BLACKHOLE\_STORAGE\_ENGINE=1

-DWITHOUT\_<ENGINE>\_STORAGE\_ENGINE=1若想禁用某个引擎的支持，如：

-DWITHOUT\_EXAMPLE\_STORAGE\_ENGINE=1

-DWITHOUT\_FEDERATED\_STORAGE\_ENGINE=1

-DWITHOUT\_PARTITION\_STORAGE\_ENGINE=1

-DWITH\_READLINE=1 启用readline库支持（提供可编辑的命令行）

-DWITH\_SSL=system 启用ssl库支持（安全套接层）

-DWITH\_ZLIB=system 启用libz库支持（zib、gzib相关）

-DMYSQL\_TCP\_PORT=3306 指定TCP端口为3306

-DMYSQL\_UNIX\_ADDR=/tmp/mysqld.sock 指定mysql.sock路径

-DENABLED\_LOCAL\_INFILE=1 启用本地数据导入支持

-DEXTRA\_CHARSETS=all 或者DWITH\_EXTRA\_CHARSETS:STRING=all启用额外的字符集类型（默认为all）

-DDEFAULT\_CHARSET=utf8 指定默认的字符集为utf8

-DDEFAULT\_COLLATION=utf8\_general\_ci 设定默认排序规则（utf8\_general\_ci快速/utf8\_unicode\_ci准确）

-DMYSQL\_USER=mysql 指定mysql用户(默认为mysql)

-DWITH\_DEBUG=0 禁用debug（默认为禁用）

* 1. 源码包安装时要进行初始化并启动服务
  2. Rpm包安装时管理服务的方式

service mysqld start|stop|restart

* 1. 掌握设置初始密码

mysqladmin –uroot password 密码

1. **掌握登陆mysql**

mysql –uroot –p密码

1. **掌握显示所有数据库**：show databases;
2. **掌握mysql数据库的创建**：
   1. 直接创建数据库： create database 数据库名；

例子：create database wg；

* 1. 如果不存在则创建数据库: create database if not exists 数据库名；

例子：create database if not exists wg；

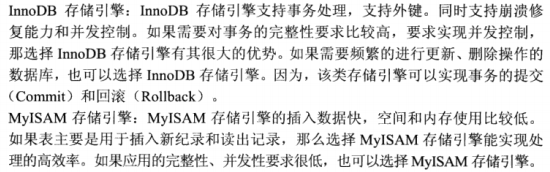
1. **掌握mysql数据库的删除**：
   1. 直接删除数据库的方法： drop database 数据库名；

例子：drop database wg；

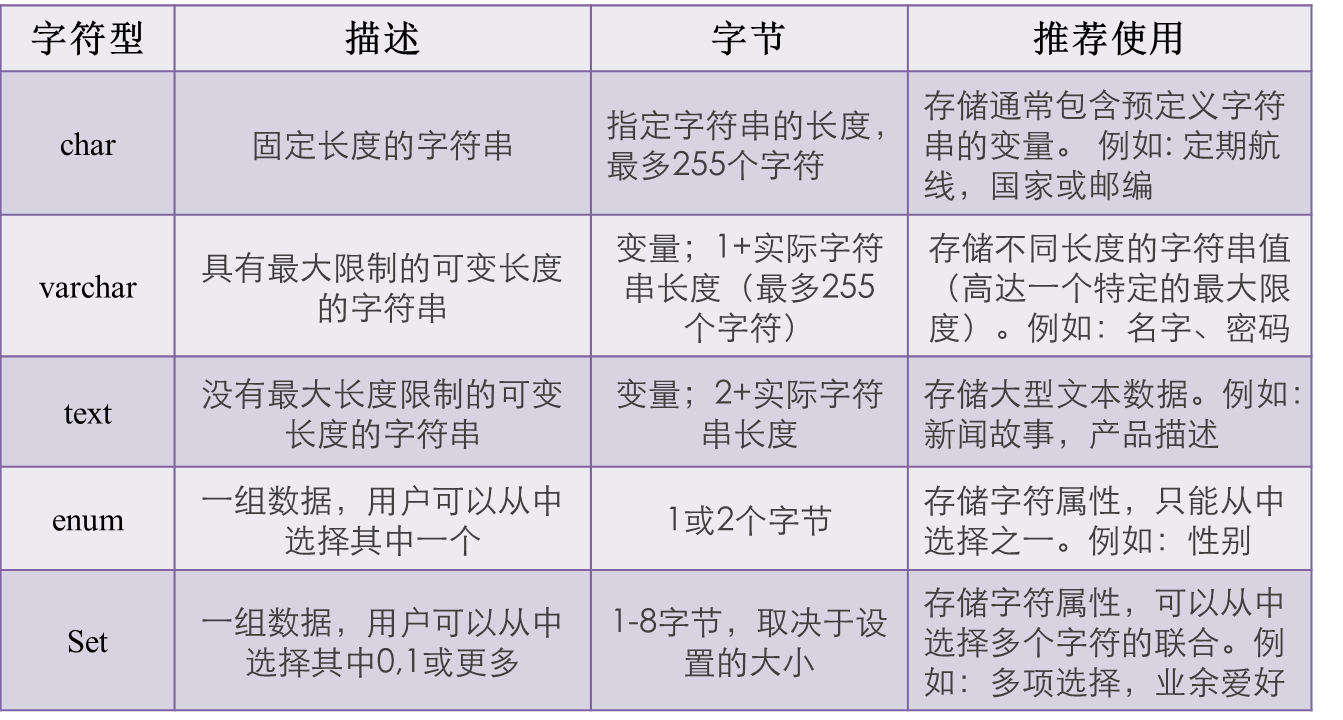
* 1. 如果存在则删除数据库的方法: drop database if exists 数据库名；

例子：drop database if exists wg；

1. **掌握数据库存储引擎**
   1. 掌握存储引擎：存储引擎就是指表的类型。数据库的存储引擎决定了表在计算机中的存储方式。
   2. 了解mysql支持的存储引擎包括：MyISAM、MEMORY、InnoDB、ARCHIVE、MRG\_MYISAM等。其中，InnoDB为默认存储引擎。最长使用的两种存储引擎是MyISAM和InnoDB。



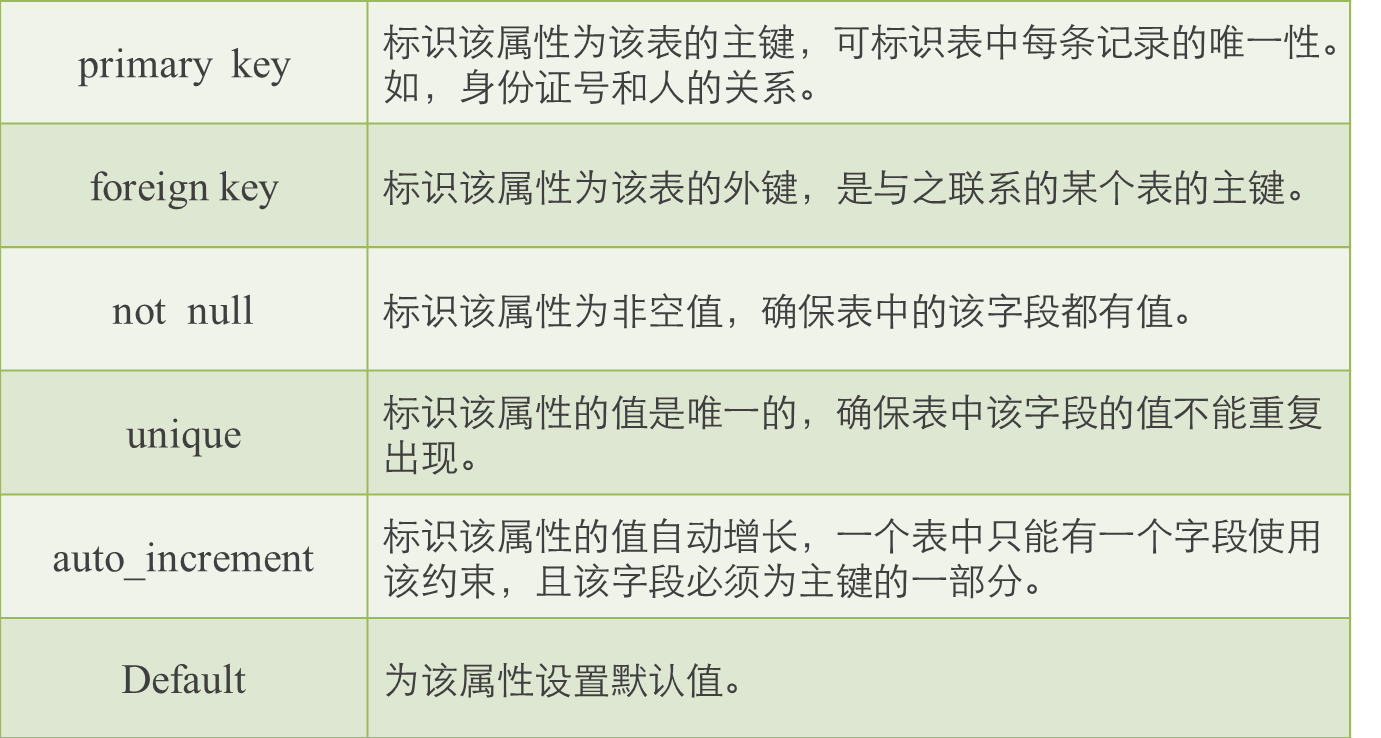
1. **掌握打开数据库：**use 数据库名;
2. **掌握字段类型**

****

****

****

1. **掌握字段约束**



1. mysql数据库中表的管理
   1. mysql数据库中表的创建、删除
      1. create table命令的使用
      2. drop table命令的使用
   2. mysql数据库中表的表结构的修改
      1. mysql表中字段的添加、修改、删除
   3. mysql数据库中表的重命名和表的复制
      1. alter命令的使用
      2. “create table…like…”的使用
   4. mysql表中数据的插入、删除和修改
      1. insert和insert into…select…语句
      2. select语句
      3. delete语句
      4. update语句
   5. mysql数据库中查询语句的使用
      1. select语句的基本结构
      2. select语句的条件顺序
   6. select语句的查询
      1. 运用条件查询
      2. 函数的使用
      3. 多表查询

【考核要求】

1. **掌握创建表的格式：**

create table 表名(

字段名1 类型 [字段约束],

字段名2 类型 [字段约束],

…

);

1. **掌握删除表：**
   1. drop table 表名;
   2. drop table if exists 表名;
2. **掌握表结构的修改**
   1. alter table 表名 add 字段名 字段类型 [after 列名]; //增加一列（字段）
   2. alter table 表名 drop 字段名; //删除一列（字段）
   3. alter table 表名 modify字段名 新字段类型; //修改字段类型
   4. alter table 表名 change 旧字段名 新字段名 字段类型; //修改字段名
3. **掌握表的重命名**
   1. alter table 表名 rename [to] 新表名；

例子：alter table student rename to stu;//将student表重新命名为stu

* 1. rename table 旧表名 to 新表名;

例子：rename table stu to student; //将stu表重命名student表

1. **掌握表的复制**
   1. create table 新表名 like 旧表名;//只复制表结构不复制数据
   2. create table 新表名 select \* from 旧表名;//表结构和数据一起复制
2. **掌握表中记录的增删改查，**对应的指令分别为：insert into、delete、update、select。
   1. 掌握insert和insert into…select…语句
      1. insert into 表名 values(字段值1,字段值2,字段值3,..); //全部列插入的方式
      2. insert into 表名(字段名1,字段名2,...) values(字段值1,字段值2,..) ; //部分列插入的方式
      3. insert into 表名 select 字段 from 表名2;//将查询结果插入的方式
   2. 了解select语句

select 查询字段 from 表名 where 条件;

其中 \* 代表所有字段

* 1. 掌握delete语句

delete from 表名; //清空表

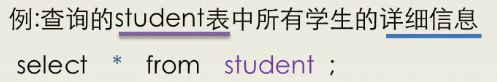
delete from 表名 where 条件;//删除满足指定条件的记录

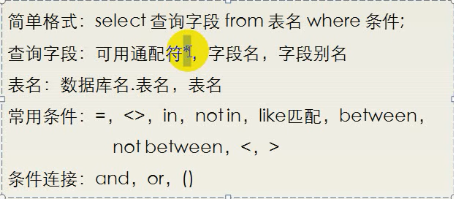
* 1. 掌握update语句

update 表名 set 字段名='新的值' [where 条件];

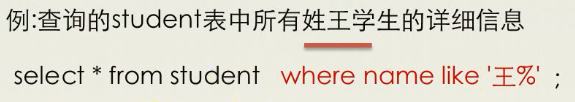
1. **掌握select语句的基本结构**

select 查询字段

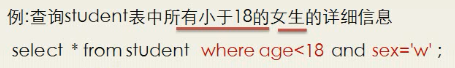
from 表名

where 条件

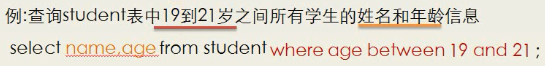
Like 模糊匹配

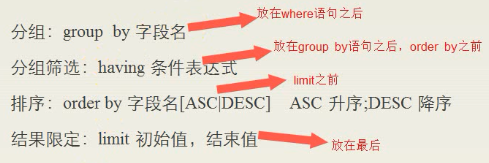


and 表示和 or 表示或



Between 表示在什么什么之间





group by 为按...分组，

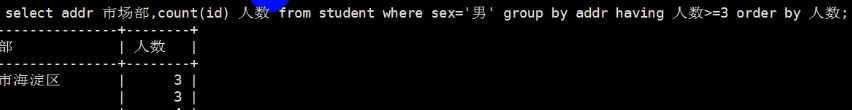


（关键字是having，用来对分组的结果进行筛选）

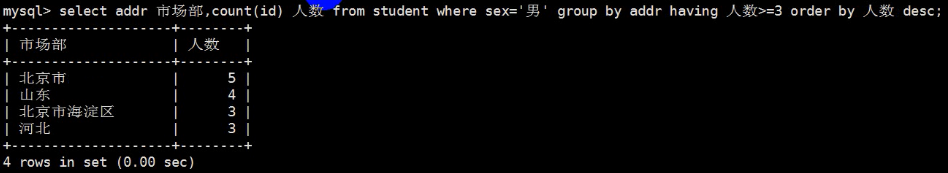
统计每个市场部的男生大于三个的有几个



order by 为按...排序 （默认是按升序排序）



降序就在后面加**desc**



limit用来限定结果，n为数字，表示筛选前n个，当limit后有两个参数时，如 limit 5,6 表示跳过前5个限定6个（放在最后）



* 1. 掌握常用的函数

max（）最大值

min（） 最小值

sum（）求和

avg（）平均值

count（）统计个数

1. mysql数据库的高级管理
   1. mysql密码的修改与恢复
      1. mysql数据库密码的修改
      2. 忘记mysql密码后的恢复
   2. mysql数据库的备份与恢复
      1. 备份的概念与分类
         1. 概念
         2. 分类
      2. 备份工具：mysqldump、mydumper
      3. 使用mysqldump工具备份与恢复
         1. 全量备份
         2. 数据库与表的恢复
      4. Mydumper工具介绍与使用
      5. mysql大数据库的备份思路
   3. mysql数据库的安全配置
      1. mysql数据库用户的添加
      2. mysql数据库用户的权限设置
   4. mysql日志管理
      1. 日志类型
      2. 错误日志
      3. 二进制日志
         1. 二进制日志
         2. 增量备份与恢复
      4. 查询日志

a) 通用查询日志

b) 慢查询日志

**【考核要求】**

1. **掌握密码的修改**
   1. 方法一：

update user set password=password('新密码') where host='主地址' and user='用户名';

flush privileges;//刷新权限

* 1. 方法二：

set password for 用户名@'主机地址' =password('新密码'); //注这种方法无需刷新权限

1. **了解忘记密码的恢复步骤**
   1. 打开cmd
   2. 停止mysql服务 net stop mysql
   3. 跳过授权列表 mysqld --skip-grant-tables
   4. 再次打开cmd，输入mysql
   5. 使用update方法修改密码
   6. 关掉所有窗口，再次打开cmd以新的密码进入mysql
2. **掌握数据库的备份**
   1. 备份的概念：为防止文件、数据丢失或[损坏](http://baike.baidu.com/view/860403.htm" \t "_blank)等可能出现的意外情况，将[电子计算机](http://baike.baidu.com/view/6373.htm" \t "_blank)[存储设备](http://baike.baidu.com/view/4403591.htm" \t "_blank)中的数据复制到[磁带](http://baike.baidu.com/view/52964.htm" \t "_blank)等大容量存储设备中,从而在原文中独立出来单独贮存的程序或文件副本; 如果[系统](http://baike.baidu.com/view/25302.htm" \t "_blank)的硬件或存储媒体发生故障，“备份”工具可以帮助您保护数据免受意外的损失。
   2. 一般的备份可分为：1、系统备份：指的是用户操作系统因磁盘损伤或损坏，计算机病毒或人为误删除等原因造成的系统文件丢失，从而造成计算机操作系统不能正常引导，因此使用系统备份，将操作系统事先贮存起来，用于故障后的后备支援。

2、数据备份：指的是用户将数据包括文件，数据库，应用程序等贮存起来，用于[数据恢复](http://baike.baidu.com/view/185060.htm" \t "_blank)时使用。

* 1. 备份更专业地可分为：
     1. 全量备份：完全备份就是指对某一个时间点上的所有数据或应用进行的一个完全拷贝
     2. 增量备份：增量备份是指在一次全备份或上一次增量备份后，以后每次的备份只需备份与前一次相比增加和者被修改的文件
     3. 差异备份：差异备份是指在一次全备份后到进行差异备份的这段时间内，对那些增加或者修改文件的备份
  2. 对mysql数据库而言，就是数据备份，可分为全量备份与增量备份
  3. 备份工具有：mysqldump、xtrabackup、mydumper
  4. mysqldump是mysql自带的一个备份工具，完成的是全量备份
     1. 备份所有数据库

mysqldump -uroot –p密码 --all-databases>盘符:\路径\文件名.sql

* + 1. 备份某个（某些）指定的数据库

mysqldump -uroot –p密码 --database 数据库名>盘符:\路径\文件名.sql

* + 1. 备份指定数据库的某个表

mysqldump -uroot –p密码 数据库名 表名>盘符:\路径\文件名.sql

* + 1. 常用选项：

--all-databases, -A： 备份所有数据库  
--database, -B： 用于备份多个数据库，如果没有该选项，mysqldump 把第一个名字参数作为数据库名，后面的作为表名。使用该选项，mysqldum把每个名字都当作为数据库名。

--force, -f：即使发现sql错误，仍然继续备份  
--host=host\_name, -h host\_name：备份主机名，默认为localhost  
--no-data, -d：只导出表结构  
--password[=password], -p[password] 密码  
--port=port\_num, -P port\_num：制定TCP/IP连接时的端口号  
--quick, -q：快速导出  
--tables：覆盖 --databases or -B选项，后面所跟参数被视作表名  
--user=user\_name, -u user\_name：用户名  
--xml, -X：导出为xml文件

* + 1. 数据恢复

mysql -u root –p密码 库或表 < 备份文件

或者登录mysql

source 备份盘符:\路径\文件名.sql

* 1. mydumper的介绍与使用举例

mydumper是一个使用C语言编写的多线程导出导入工具,并且能够保证多个表之间的一致性

下载源码包进行安装

#yum install -y gcc gcc-c++ glib2-devel mysql-devel zlib-devel pcre-devel cmake

#tar zxvf mydumper-\*.tar.gz –C /usr/src

#cd /usr/src/mydumper-\*

#cmake && make && make install

mydumper常用选项：

-u "用户"  
-p "密码"  
-t "指定并行数,默认是4"  
-B "指定DB"  
-T "指定表"  
-c "压缩"  
--less-locking "尽量减少锁表锁定时间(针对InnoDB)"  
-o "指定目录"

mydumper备份举例：

mydumper -u root -p 'xxxxxxxx' -t 4 -B test -T test -c --less-locking -o /home/mysql/

mydumper数据恢复：

myloader -u root -p 'xxxxx' -B test -d /home/mysql/

1. 了解大数据库备份思路

在数据库设计阶段就做好设计，如果可以预先估计数据很大，则可以使用数据切分，使得每个数据库的数据都比较小，这样方便备份与恢复。如果没有做好设计，单位数据库很大，那可以适当使用效率高的备份工具进行备份，如上面提到的Xtrabackup、 mydumper等

1. 掌握用户的添加

create user 用户名@'地址' identified by'密码';

1. **了解用户的权限**

权限可以分为四个层级：全局级别（\*.\*）、数据库级别(数据库名.\*)、表级别(数据库名.表名)、列级别( 权限（列） 数据库名.表名)。

全局级别的权限存放在mysql.user表中，数据库级别的权限存放在mysql.db或者mysql.host，表级别的权限存放在mysql.tables\_priv中，列级别的权限存放在mysql.columns\_priv中。

1. **掌握为用户授权**

grant 权限列表 on 数据库名.表名 to 用户@'地址';

1. **掌握在授权的同时创建用户**

grant 权限列表 on 数据库名.表名 to 用户@'地址' identified by'密码';

1. **了解收回权限**

revoke 权限列表 on 数据库名.表名 from 用户@'地址';

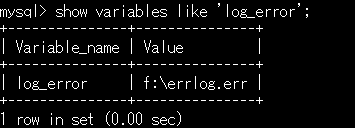
1. **掌握MySql日志的管理**
   1. 掌握mysql日志类型

错误日志：记录启动、运行或停止mysqld时出现的问题。

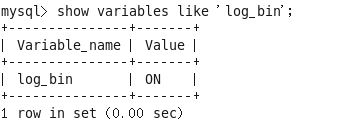
二进制日志： 该日志文件会以二进制形式记录数据库的各种操作，但是不记录查询语句。

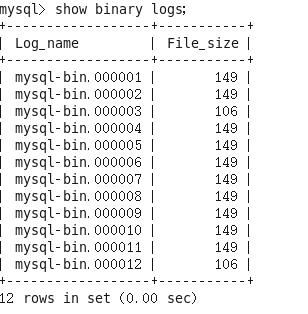
查询日志：该日志分为两类：通用查询日志和慢查询日志。前者记录建立的客户端连接和执行的语句。 后者记录所有执行时间超过long\_query\_time秒的所有查询或不使用索引的查询。

* 1. 掌握错误日志的参数:log-error
  2. 掌握获取错误日志的方法

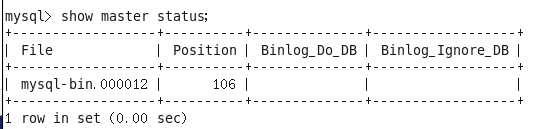


* 1. 掌握查看错误日志：使用文本工具查看
  2. 掌握二进制日志的参数：log-bin
  3. 掌握查看二进制日志是否开启

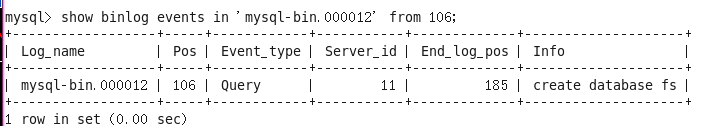
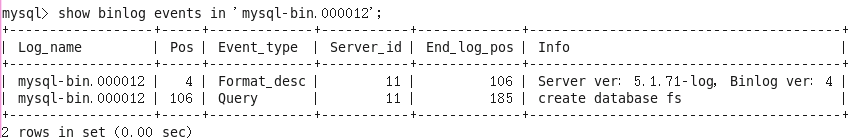


掌握查看二进制日志文件名

* 1. 掌握查看正在使用哪一个日志



* 1. 查看二进制文件内容



* 1. 了解停止二进制日志的两种方法

一种方法是删除[mysqld]组里的log-bin条目，重启mysql服务

另一种方法是：

set sql\_log\_bin=0 //暂停二进制日志

set sql\_log\_bin=1 //启用二进制日志

* 1. 了解删除二进制日志

1. 删除所有二进制日志，然后重新生成一个从000001开始的文件

reset master;

1. 删除编号小于number的所有二进制日志文件

purge master logs to filename.number；;

1. 删除指定时间之前的二进制日志

purge master logs before 'yyyy-mm-dd hh:mm:ss';

* 1. 掌握mysqlbinlog命令查看二进制日志

mysqlbinlog命令的常用选项

--start-postion 开始位置

--stop-postion 结束位置

--start-datetime 'yyyy-mm-dd hh:mm:ss' 开始时间

--stop-datetime 'yyyy-mm-dd hh:mm:ss' 结束时间

mysqlbinlog --no-defaults /var/lib/mysql/mysql-bin.000001 --start-position=106 --stop-position=272

1. **掌握使用二进制日志还原数据**

在备份mysql数据库时，将相关的二进制日志备份进行备份就是mysql的增量备份，恢复时可以二进制日志进行增量恢复

* 1. 通过时间还原

mysqlbinlog --start-datetime=“时间” 日志文件路径 | mysql -u用户 -p //从规定的起始时间还原到现在

mysqlbinlog --stop-datetime=“时间” 日志文件路径 | mysql -u用户 -p //从最开始还原到规定的结束时间

mysqlbinlog --start-datetime=“时间” --stop-datetime=“时间” 日志文件路径 | mysql -u用户 -p //从规定的起始时间还原到规定的结束时间

* 1. 通过位置还原

mysqlbinlog --start-position=“位置” 日志文件路径 | mysql -u 用户 -p //从规定的起始位置还原到现在

mysqlbinlog --stop-position=“位置” 日志文件路径 | mysql -u 用户 -p //从最开始还原到规定的结束位置

mysqlbinlog --start-position=“位置” --stop-position=“位置” 日 志文件路径 | mysql -u用户 -p //从规定的起始位置还原到规定的结束位置

1. **掌握通用查询日志的参数：log**
2. **掌握慢查询日志的参数：**log-slow-queries和设置时间的参数：long\_query\_time

课外阅读：Xtrabackup工具介绍与使用

* 1. 了解Xtrabackup的介绍与使用举例

Xtrabackup是由percona提供的mysql数据库备份工具, 有两个主要的工具：xtrabackup、innobackupex，备份过程快速、可靠，备份过程不会打断正在执行的事务，能够基于压缩等功能节约磁盘空间和流量，自动实现备份检验，还原速度快，备份可在线备份，但是恢复要关闭服务器，恢复后再启动.

可到<http://www.percona.com/downloads/>下载安装

#rpm -ivh percona-xtrabackup-\*.rhel6.x86\_64.rpm

Xtrabackup常用选项：

--defaults-file：指定my.cnf参数文件的位置

--apply-log：通过回滚未提交的事务及同步已经提交的事务至数据文件使数据文件处于一致性状态。

--copy-back：做数据恢复时将备份数据文件拷贝到MySQL服务器的datadir

--remote-host=HOSTNAME： 通过ssh将备份数据存储到进程服务器上

--stream=[tar]：备份文件输出格式, 可同时使用--tmpdir指定临时目录来解决临时文件放在/tmp是过大的问题

--tmpdir=DIRECTORY：当有指定--remote-host or --stream时, 事务日志临时存储的目录, 默认采用MySQL配置文件中所指定的临时目录tmpdir

--redo-only --apply-log：强制备份日志时只redo,跳过rollback,用在增量备份

--databases=LIST：列出需要备份的databases，默认所有库

Xtrabackup应用案例

* + - 1. 完全备份与恢复：  
         # 指定备份某一数据库   
         # innobackupex --user=root --password=123456 --defaults-file=/etc/my.cnf --port=3306 --databases=a /tmp/backup/ >/tmp/backup/innoback.log 2>&1   
         # 备份所有的数据库   
         # innobackupex --user=root --password=kongzhong --defaults-file=/etc/my.cnf --port=3306  /tmp/backup/ >/tmp/backup/innoback.log 2>&1

# innobackupex --apply-log /tmp/backup/2014-01-11\_14-46-21/ //备份完成后，应用日志，使备份的数据保持一致

# innobackupex --copy-back /tmp/backup/2014-01-11\_14-46-21/  //完全恢复，恢复前要清空数据目录

* + - 1. 增量备份与恢复

基于全备的增量备份

# innobackupex --user=root --password=123456 --defaults-file=/etc/my.cnf --port=3306 --incremental --incremental-basedir=/tmp/backup/全备的文件名/ /tmp/backup/

增量备份的增量备份

# innobackupex --user=root --password=123456 --defaults-file=/etc/my.cnf --port=3306 --incremental --incremental-basedir=/tmp/backup/增量备份的文件名/ /tmp/backup/

# innobackupex  --apply-log /tmp/backup/2014-01-11\_15-37-31/

# innobackupex  --apply-log /tmp/backup/2014-01-11\_15-37-31/ --incremental-dir=/tmp/backup/2014-01-11\_15-45-06/

# innobackupex  --apply-log /tmp/backup/2014-01-11\_15-37-31/ --incremental-dir=/tmp/backup/2014-01-11\_15-46-33/

//备份完成后，应用日志，使备份的数据保持一致，将两次增量备份数据合并到全备，恢复时只用全备就可以了

增量恢复

# innobackupex --copy-back /tmp/backup/2014-01-11\_15-37-31/

# chown -R mysql:mysql /var/lib/mysql/  //改变数据目录权限

1. mysql数据库的主从复制集群应用
   1. mysql主从复制
      1. mysql主从复制概述
      2. Mysql主从复制工作原理
   2. mysql主从复制配置
      1. 在mysql主服务器（master）上的配置
      2. 在mysql从服务器（Slave）上的配置
      3. 主从复制测试
   3. Mysql多主多从架构分析

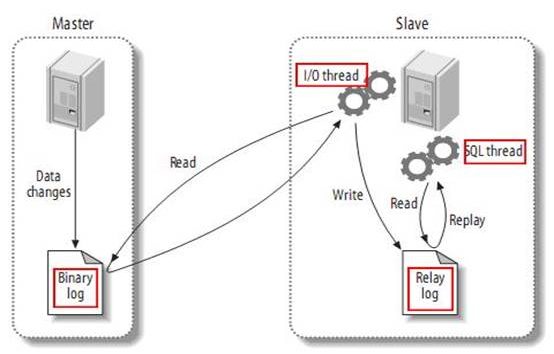
【考核要求】

1. **掌握主从复制的优点**

健壮性:主服务器出现故障时,可以切到从服务器作为备份

速度快:更新操作在主服务器端,查询操作在从服务器端,可以加快用户的响应时间

1. **掌握主从复制的过程**



1. **掌握主服务器的配置过程**

1）.配置主服务器：

在安装目录下找到my.ini配置文件，在文件结尾写入：（Linux下的配置文件为/etc/my.cnf）

server-id=1

log-bin=mysql-bin #启用二进制日志文件

2). 重启mysql服务（Linux下为service mysqld restart）

net stop mysql;

net start mysql;

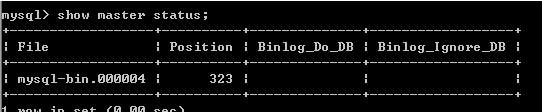
3). 在主服务器上建立账号并授权

grant replication slave on \*.\* to backup@‘%’ identified by‘backup’;

flush privileges;

4). 查看主服务器状态

show master status; #记录下日志文件名和位置，备用



1. **掌握从服务器的配置过程**

1). 配置从服务器：

在安装目录下找到my.ini配置文件，在文件结尾写入：（Linux下的配置文件为/etc/my.cnf）

server-id=2

relay-log=mysql-relay #启用中继日志文件

2). 重启mysql服务（Linux下为service mysqld restart）

net stop mysql;

net start mysql;

3). 关闭从服务器的复制功能

stop slave;

4). 配置从服务器

change master to master\_host='172.18.3.73',

master\_user='backup',

master\_password='backup',

master\_log\_file='mysql-bin.000004',

master\_log\_pos=323;

5). 开启从服务器的复制功能

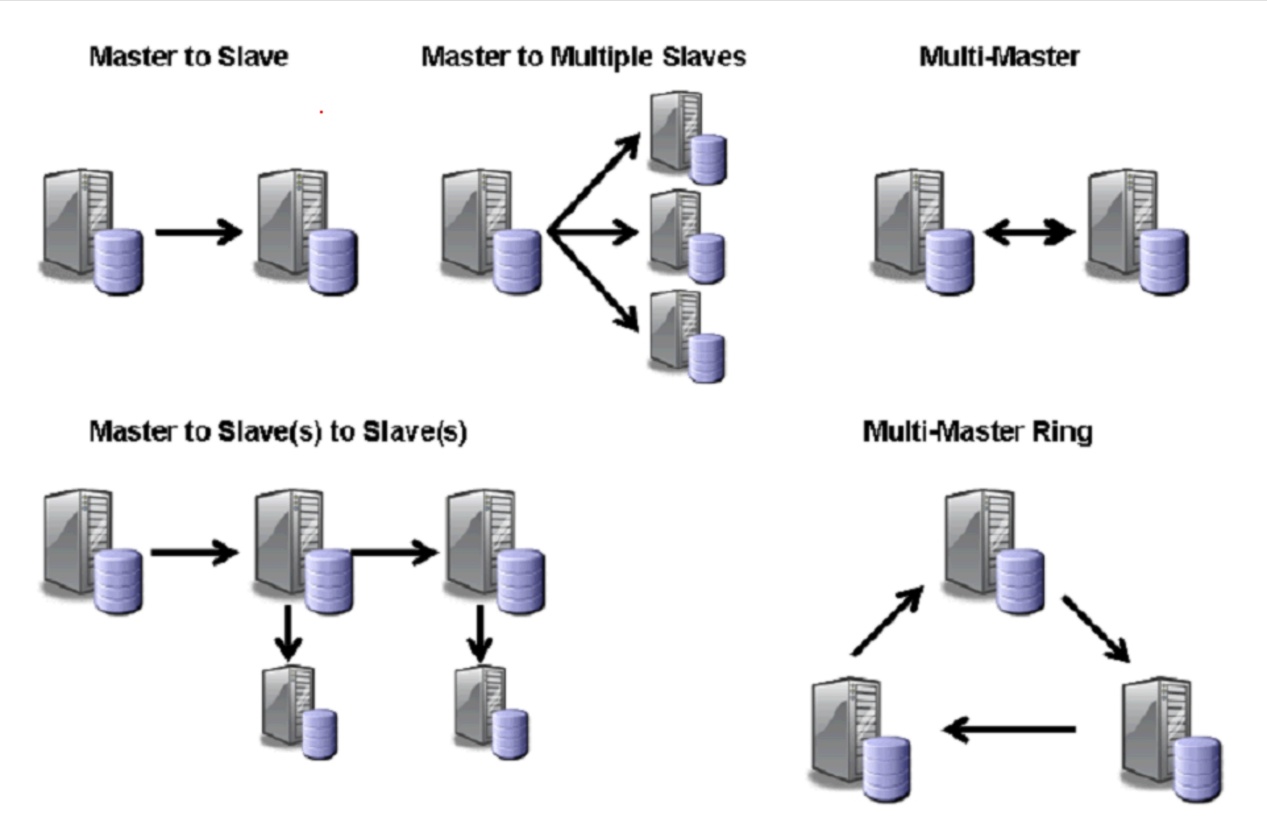
start slave;

检查从服务器的复制功能状态

show slave status; #检查画横线的两行是否变为yes



1. **了解多主从从的架构**

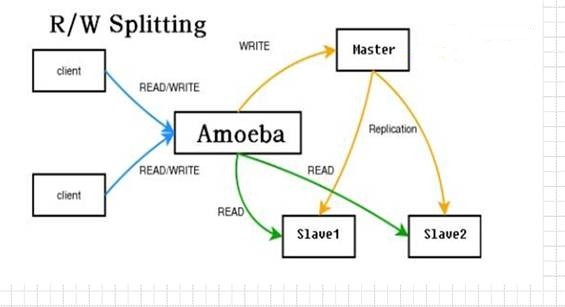
除了一主多从的架构之后，还有双主双从或者多主多从，较常见的复制架构有“一主一从”、“一主多从”、“双主”、“多级复制”和“多主环形机构”等，见下图：

1. 高性能mysql读写分离的实现
   1. mysql读写分离
      1. mysql读写分离概述
      2. mysql读写分离原理
   2. mysql读写分离配置
      1. mysql读写分离的服务器配置
      2. mysql读写分离测试
      3. mysql数据切分

【考核要求】

1. **理解读写分离的原理**

Amoeba(变形虫)项目，该开源框架于2008发布一款Amoeba for mysql软件，该软件致力于mysql的分布式数据库前端代理层，主要的作用是应用服务访问mysql服务器时充当SQL路由功能，并具有负载均衡、高可用性、SQL过滤、读写分离、可路由相关SQL的到目标数据库、可并发请求多台数据库全并结果的作用。通过Amoeba能够完成多数据源的高可用、负载均衡、数据切片的功能，目前Amoeba已在很多企业的生产线上面使用；其版本可在官网进行下载。其工作原理图如下：



1. **掌握读写分离的配置过程**

安装java环境，此处安装是jdk-6u14-linux-x64.bin，因为amoeba推荐使用 jdk1.5或 1.6版本，高版本不建议使用

chmod +x /usr/local/jdk-6u14-linux-x64.bin

/usr/local/jdk-6u14-linux-x64.bin #根据提示按回车建完成即可

mv /usr/local/jdk1.6.0\_14 /usr/local/jdk1.6

vi /etc/profile  #编辑/etc/profile文件来修改系统环境变量

export JAVA\_HOME=/usr/local/jdk1.6

export CLASSPATH=$CLASSPATH:$JAVA\_HOME/lib:$JAVA\_HOME/jre/lib

export PATH=$JAVA\_HOME/lib:$JAVA\_HOME/jre/bin:$PATH:$HOME/bin

export AMOEBA\_HOME=/usr/local/amoeba/

export PATH=$PATH:$AMOEBA\_HOME/bin



source /etc/profile  #使环境变量立即生效

java -version

http://note.youdao.com/yws/public/resource/f328c12ab408bc87744c28d580258951/DE00FF6472534DED844CBFEF1D217410

如上图java环境已配置成功

1. **安装并配置amoeba软件**

mkdir /usr/local/amoeba

tar zxf /root/amoeba-mysql-binary-2.1.0-RC5.tar.gz –C /usr/local/amoeba

chmod –R 755 /usr/local/amoeba/

/usr/local/amoeba/bin/amoeba

1. **配置amoeba读写分离**

所有的mysql数据库都要开放给amoeba访问，在主和从数据库上执行以下命令，ip地址应该为amoeba的ip地址，如：

grant all on \*.\* to test@’106.120.190.%’ identified by ‘123.com’;

vim /usr/local/amoeba/conf/amoeba.xml

……

<!-- Amoeba 账号，密码 -->

       <property name="user">amoeba</property>   //30行

      <property name="password">123456</property>

……

 <!-- 默认数据库，主数据库 -->

        <property name="defaultPool">master</property>   //115行

        <!-- 写数据库 -->

        <property name="writePool">master</property>

        <!-- 读数据库，dbServer.xml 中配置的 虚拟数据库，数据库池 -->

        <property name="readPool">slaves</property>

        <property name="needParse">true</property>

……

vim /usr/local/amoeba/conf/dbServers.xml

……

<!-- mysql port -->

     <property name="port">3306</property>

        <!-- mysql schema -->

        <property name="schema">test</property>

        <!-- mysql user -->

        <property name="user">test</property>

        <!--  mysql password -->

     <property name="password">123.com</property>

……

 <!-- Master ，Slave1， Slave2 的独立部分，也就只有 IP 了 -->

    <dbServer name="master"  parent="abstractServer">  //45行

    <factoryConfig>

    <!-- mysql ip -->

    <property name="ipAddress">106.120.190.233</property> //Master的IP

    </factoryConfig>

    </dbServer>

    <dbServer name="slave1"  parent="abstractServer">     <factoryConfig>

    <!-- mysql ip -->

    <property name="ipAddress">106.120.190.236</property> //Slave1的IP

    </factoryConfig>   //如果有多台slave，则可再配置

    </dbServer>

    <dbServer name="slave2"  parent="abstractServer">

    <factoryConfig>

    <!-- mysql ip -->

    <property name="ipAddress">106.120.190.228</property>  //Slave2的IP

    </factoryConfig>

    </dbServer>

……

<!-- 数据库池，虚拟服务器，实现读取的负载均衡 -->

  <dbServer name="slaves" virtual="true">

  <poolConfig class="com.meidusa.amoeba.server.MultipleServerPool">

  <!-- Load balancing strategy: 1=ROUNDROBIN , 2=WEIGHTBASED , 3=HA-->

  <property name="loadbalance">1</property>

  <!-- Separated by commas,such as: server1,server2,server1 -->

  <property name="poolNames">slave1</property>  //单台slave的配置格式，多台的配置格式为：<property name="poolNames">slave1,slave2</property>

 </poolConfig>

 </dbServer>

启动amoeba软件

/usr/local/amoeba/bin/amoeba

1. **测试ameoba**

测试的时候和我们平时使用一样，amoeba-mysql对我们应用透明，就是个mysql的代理

在客户端安装mysql客户端，登录mysql使用如下命令（用户名密码和上面配置要一致）

mysql –u amoeba –p123456 –h amoeba的ip –P 8066

登录上去后，为了测试读和写必须，先把mysql的主从复制停掉，才能更清楚地看出读写的服务器是哪台，在从执行用stop slave;登录到amoeba上，然后执行写和读操作，查看写的是哪台服务器，读的是哪台服务器，实验结果显示：写只在主上进行，读在从上进行，但是可通过配置使得主从都可读，比率是1:1

测试步骤：

1.在客户机上安装mysql

yum –y install mysql

mysql –uroot –p123456 –h amoeba的ip地址 –P 8806

2.在master上建一个测试用库，如db\_test，并建测试用表，如wg2:

mysql>create database db\_test;

mysql>create table a\_test(id int(10),addr varchar(20));

3.在从服务器上:

mysql>stop slave;

4.然后在主服务器上：

mysql>use db\_test;

mysql>insert into a\_test values(‘1’,’master’);

5.在从服务器上：

mysql> use db\_test;

mysql>insert into a\_test values(‘2’,‘slave’);

6.在客户端上查询数据：

mysql>select \* from db\_test.a\_test;

7.在客户端上：

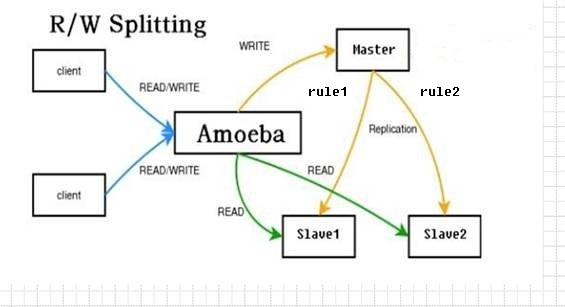
insert into db\_test.a\_test values(‘1’,’mrite’);

在主从上分别查询看结果:

select \* from db\_test.a\_test;

1. **了解使用amoeba进行数据切分**

根据数据的特点，将不同特性的数据纪录存放到不同的数据库服务器中就是数据切分，可以设置规则匹配数据特性以实现数据切分，其实，数据切分就是根据设置规则将数据分别存放到不同的数据库，以减轻数据库服务器的压力和减小单个数据库的大小，可通过设置amoeba的rule.xml文件中的切片规则实现。原理如图：



1. mysql优化与常见故障排查
   1. mysql优化
      1. 为什么要优化
      2. 硬件优化
      3. 网络优化
   2. mysql配置优化
      1. mysql参数说明
      2. mysql参数优化
   3. mysql常见故障排查
      1. 故障排查思路
      2. 故障排查案例
         1. 启动故障
         2. 连接故障
         3. 字符乱码
         4. 其他常见故障解决思路

【考核要求】

1. **了解为什么要进行数据库的优化**

优化，就是让数据库发挥更好的性能。一般情况下,数据库的优化指的就是查询性能的优化,让数据库对查询的响应尽可能的快.

仅对数据库系统本身而言,影响到查询性能的因素，包括数据库参数设置(其实就是通过参数控制数据库系统的内存,i/o,缓存，备份等一些管理性的东西),索引,分区,sql语句.分区则主要是针对大数据量的情况下，它分散了数据文件的分布，减少磁盘竞争，使效率得到提升。

1. **掌握硬件优化的措施**

在谈到基于硬件来进行[数据库](http://lib.csdn.net/base/14" \t "_blank" \o "MySQL知识库)性能瓶颈分析的时候，常被大家误解为简单的使用更为强劲的主机或者存储来替换现有的设备。我们在谈论基于硬件进行优化的时候，不能仅仅将数据库使用的硬件划分为主机和存储两部分，而是需要进一步对硬件进行更细的分解，就分为：cpu、内存、磁盘、RAID卡、存储设备等方面。

最容易出现性能瓶颈的地方主要会出现在以下几个方面：

1. IO资源方面瓶颈，主要表现在服务器 iowait 很高，系统响应较慢，数据库中经常会存在大量执行状态的 session。

优化措施：

增加内存，加大可缓存的数据量

改善底层存储设备的 IO 能力

1. CPU资源方面瓶颈，主要表现在服务器CPU利用率中 usr 所占比例很高，iowait却很小

优化措施：

将运算尽可能从数据库端迁移到应用端，降低数据库主机的计算量

提升CPU处理能力

要么增加 CPU 数目（如果支持），要么换CPU更强劲的主机

1. **掌握网络优化的措施**

一般来说应用与数据库之间的网络交互所需的资源并不是非常大，但是在分布式的集群环境中，各个数据库节点之间的网络环境经常会称为系统的瓶颈。

优化措施：

廉价一点的解决方案是通过万兆交换机来替换现在常用的千兆交换机，来提升网络处理能力降低网络延时。不过这个方案主要提升的是吞吐量方面，对于延时方面的提升可能并不一定能满足某些要求非常高的场景。这时候就该考虑使用更为昂贵但也更高效的方案：用 Infiniband 替换普通交换机来极大的降低网络方面所带来的数据交换延时。

Infiniband**：**

InfiniBand架构是一种支持多并发链接的“转换线缆”技术，在这种技术中，每种链接都可以达到2.5 Gbps的运行速度。这种架构在一个链接的时候速度是500 MB/秒，四个链接的时候速度是2 GB/秒，12个链接的时候速度可以达到6 GB /秒。

1. **掌握mysql常用配置参数与优化配置**
2. 普通参数设置

**back\_log：**设置MySQL能处理的连接数量，就是TCP/IP连接的侦听队列的大小。默认数值是50，可适当放大；

**interactive\_timeout：**服务器在关闭一个交互连接前等待活动的秒数，默认数值是28800

**max\_connections：**允许同时连接的客户的数量。增加该值能增加mysqld的文件描述符的数量。这个数字应该增加，否则，将经常看到 Too many connections 错误。 默认数值是100，可改为1024

**key\_buffer\_size：**用于设置索引块的缓冲区大小，增加它可更好处理的索引(对所有读和多重写)，但是太大时，系统将开始变慢。默认数值是8388600(8M)，可根据服务器内存的大小设置更大的值。

**table\_cache：**为所有线程打开表的数量。增加该值能增加mysqld要求的文件描述符的数量。MySQL对每个唯一打开的表需要2个文件描述符。默认数值是64，可适当增加，如512。

**wait\_timeout：**服务器在关闭一个非交互连接之前等待活动的秒数。 默认数值是28800。

1. InnoDB设置

**innodb\_buffer\_pool\_size :** 默认值为 128M. 这是最主要的优化选项,指定 InnoDB 使用多少内存来加载数据和索引，针对专用MySQL服务器,建议指定为物理内存的 50-80%这个范围. 例如,拥有64GB物理内存的机器,缓存池应该设置为50GB左右.   
 如果将该值设置得更大可能会存在风险,比如没有足够的空闲内存留给操作系统和依赖文件系统缓存的某些MySQL子系统(subsystem),包括二进制日志(binary logs),InnoDB事务日志(transaction logs)等.

**innodb\_log\_file\_size：**默认值是48M. 有很高写入吞吐量的系统需要增加该值. 将此值设置为4G以下是很安全的. 不能设置太大，日志文件太大的缺点是增加了崩溃时所需的修复时间。

1. 主从复制(Replication)参数优化

**log-bin**：启用二进制日志. 默认情况下二进制日志不是事故安全的(not crash safe), 建议大多数用户应该以稳定性为目标. 在这种情况下,还需要启用: sync\_binlog=1, sync\_relay\_log=1, relay-log-info-repository=TABLE and master-info-repository=TABLE.

sync\_relay\_log：这个参数和sync\_binlog是一样的，当设置为1时，slave的I/O线程每次接收到master发送过来的binlog日志都要写入系统缓冲区，然后刷入relay log中继日志里，这样是最安全的，因为在崩溃的时候，你最多会丢失一个事务，但会造成磁盘的大量I/O。当设置为0时，并不是马上就刷入中继日志里，而是由操作系统决定何时来写入，虽然安全性降低了，但减少了大量的磁盘I/O操作

**expire-logs-days**：默认旧日志会一直保留. 推荐设置为 1-10 天. 保存更长的时间并没有太多用处,因为从备份中恢复会快得多.

**server-id**：在一个主从复制体系(replication topology )中的所有服务器都必须设置唯一的 server-id.

**binlog\_format=ROW**：修改为基于行的复制，可以通过减少资源锁定提高性能.

其他配置(Misc)

**character-set-server=utf8mb4、collation-server=utf8mb4\_general\_ci** ：utf8编码对新应用来说是更好的默认选项.

**skip-name-resolve**：禁用反向域名解析. DNS解析在某些系统上可能有点慢，不稳定,所以如果不需要基于主机名的解析,建议避免这种解析.

1. **掌握常见故障的排查方法**
2. 了解故障排查思路

在mysql的数据库中，故障可分为三类，硬件故障，网络故障，mysql服务本身的故障。

遇到故障，首先应该仔细分析提示信息，就是报警信息，一般都能给出故障的问题所在；如果报警信息比较模糊，则可分析相应日志文件，包括mysql本身的日志文件，还有操作系统的日志文件，细心分析，就能定位问题所在，从而找出解决问题。

1. 掌握常见故障排查方法
   * + 1. 启动故障

报警：

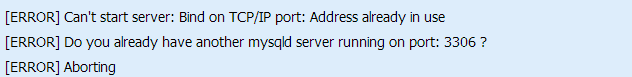


原因分析：MySQL的$datadir目录，及其下属目录、文件权限属性置不正确，导致MySQL无法正常读写文件，无法启动

解决办法：chown –R mysql:mysql 数据目录

属主：属组

报警：



原因分析：已有其他mysqld实例启动，且占用了相同端口

解决办法：修改port选项，指定其他未用端口

报警：



原因分析：安装后没有初始化数据库造成

解决办法：根据实际情况使用mysql\_install\_db工具进行数据库的初始化

* + - 1. 连接故障：

报警：

ERROR 1045 (28000): Access denied for user 'usera'@'localhost' (using password:YES)

ERROR 1045 (28000): Access denied for user 'usera'@'localhost' (using password:NO)

原因分析：客户端远程访问的用户账号并未创建； 用户账号存在，但未对其所在的客户端的IP进行远程访问授权允许；密码不正确

解决方法：对第一种情况，使用grant语句对相应用户进行授权

grant all privileges on \*.\* to '用户名'@'ip' identified by '密码' with grant option;

对后一种情况，使用grant设置相应用户能在所有ip上远程连接mysql：

GRANT ALL PRIVILEGES ON \*.\* TO '用户名'@'%' IDENTIFIED BY '密码' WITH GRANT OPTION;

执行完grant语句要flush priviges;

报警：

ERROR 2003 (HY000): Can't connect to MySQL server on '192.168.8.88' (10065)

原因分析：防火墙阻挡了连接

解决方法：设置iptables规则或者关闭防火墙

报警：

ERROR 2003 (HY000): Can't connect to MySQL server on 'hostxxxxx' (10061)

原因分析：数据库没有启动

解决办法：启动数据库即可

报警：

ERROR 2002 (HY000): Can't connect to local MySQL server server through socket '/var/lib/mysql/mysql.sock'(111)

原因分析：mysqld的mysql.sock没在相应的位置

解决办法：

/etc/rc.d/init.d/mysql stop

chown -R mysql:msyql /var/lib/mysql

vim /etc/my.cnf

[mysqld]  
  datadir=/usr/local/mysql/data  
  socket=/var/lib/mysql/mysql.sock  
[mysql.server]  
  user=mysql  
  basedir=/usr/local/mysql  
[client]  
  socket=/var/lib/mysql/mysql.sock

启动数据库

* + - 1. 字符乱码故障

中止MySQL服务（bin/mysqladmin -u root shutdown -p）

修改/etc/下的my.cnf，保存并关闭

在[client] 在下面添加  
default-character-set=utf8 默认字符集为utf8  
在[mysqld]下添加  
default-character-set=utf8 默认字符集为utf8  
init\_connect='SET NAMES utf8' （设定连接mysql[数据库](http://www.111cn.net/list-55/" \t "_blank)时使用utf8编码，以让mysql数据库为utf8运行）

启动MySQL服务（bin/mysqld\_safe &），重新查看编码显示

show variables like 'character%';

* + - 1. 其他常见故障解决思路

1）too many connections错误的解决步骤：

查看当前mysql允许的最大连接数:  
mysql>show variables like ‘max\_connections’;  
（可以通过show  variables like "%max%"，可以查看到所有的最大值限制信息）

查看当前的链接数:  
mysql> show processlist;  
增加允许的最大链接数:  
mysql>set global max\_connections=200;

2）Host '127.0.0.1' is blocked because of many connection errors.错误解决访求：

执行mysqladmin flush-hosts来解除锁定

1. shell脚本概述
   1. shell概念
   2. shell脚本结构
      1. 脚本开头
      2. 注释
      3. 命令行的书写规则
   3. 变量
      1. 系统变量
      2. 用户自定义变量
      3. 特殊变量
      4. read从键盘读入内容
   4. cut、sort、uniq、wc命令详解
   5. find文件查找工具案例分析
      1. find工具详解
      2. find案例实战

【考核要求】

1. **了解什么是shell**

shell是指“提供使用者使用界面”的软件（命令解析器）。它类似于DOS下的command和后来的cmd.exe。它接收用户命令，然后调用相应的应用程序。

1. **掌握什么是shell脚本**

Shell脚本与Windows/Dos下的批处理相似，也就是将各类命令预先放入到一个文件中，方便一次性执行的一个程序文件，主要是方便管理员进行设置或者管理用的。但是它比Windows下的批处理更强大。

1. **掌握shell脚本语法**
   1. 开头：#!/bin/sh或#!/bin/bash

符号#!用来告诉系统它后面的参数是用来执行该文件的程序

* 1. 注释:

以#开头的行表示注释

* 1. 命令行的书写规则：

一行一条命令

若一行多个命令，用分号（;）分割

长命令可以使用反斜线字符（\）

1. **掌握变量的使用**
   1. 了解系统变量

|  |  |
| --- | --- |
| 系统定义的变量 | 意义 |
| BASH=/bin/bash | Bash Shell 名称 |
| BASH\_VERSION=4.1.2(1) | Bash 版本 |
| HOME=/home/linuxtechi | 用户家目录 |
| LOGNAME=LinuxTechi | 当前登录用户的名字 |
| OSTYPE=Linux | 操作系统类型 |
| PATH=/usr/bin:/sbin:/bin:/usr/sbin | 可执行文件搜索路径 |
| PWD=/home/linuxtechi | 当前工作目录 |
| SHELL=/bin/bash | Shell 名称 |
| USERNAME=linuxtechi | 当前登录的用户名 |

1. **掌握用户自定义变量**
   * 1. 在shell 脚本中，所有的变量都由字符串组成，且不需要对变量进行声明。
     2. 用户变量可以是任何不超过20个的字母、数字或者下划线字符的文本字符串（变量只能以字母或下划线开头）。用户变量是大小写敏感的，因此，变量Var1和变量var1是不同的变量。
   1. 特殊变量

|  |  |
| --- | --- |
| 特殊变量 | 含义 |
| $$ | Shell本身的PID（ProcessID） |
| $! | Shell最后运行的后台Process的PID |
| $? | 最后运行的命令的结束代码（返回值） |
| $- | 使用Set命令设定的Flag一览 |
| $\* | 所有参数列表。如"$\*"用「"」括起来的情况、以"$1 $2 … $n"的形式输出所有参数。 |
| $@ | 所有参数列表。如"$@"用「"」括起来的情况、以"$1" "$2" … "$n" 的形式输出所有参数。 |
| $# | 添加到Shell的参数个数 |
| $0 | Shell本身的文件名 |
| $1～$n | 添加到Shell的各参数值。$1是第1参数、$2是第2参数…。 |

* 1. 变量的赋值

变量的赋值使用“=”

例子:

x=6

a="welcome to beijing"

单引号：所有转移符全部关闭，完整的反应括号中的内容

双引号：部分转义符关闭，但某些则保留(如：$ )

反引号：反引号内荣作为一个系统命令并执行

* 1. 变量的调用

在变量名前面加一个$符号

* 1. read从键盘读变量

read -p “提示信息” 变量名

例子：

read –p “请输入你的用户名” username

1. **掌握cut功能：从一行中切割部分内容**
   1. 选项：

-d：设置分隔符

-f：指定分割后的域

-c：按字符分割

* 1. 实例：

cut -c1 /etc/grub.conf

显示/etc/grub.conf里第1个字符

cut -c1-20 /etc/grub.conf

显示/etc/grub.conf里第1个到第20个字符

cut -d“:” -f1-3 /etc/passwd

显示/etc/passwd里，以冒号分隔的第1列到第3列

cut -d“:” -f1，3 /etc/passwd

显示/etc/passwd里，以冒号分隔的第1列和第3列

1. **掌握sort功能：排序**
   1. 选项：

-r：逆序

-t：设置分隔符

-n：使用纯数字排序

+a: 从分隔后的第a域开始 -b:到分隔后的第b个域结束 注：分隔后的列是从0开始计数

-u：重复行仅显示一次

-k: 按照第几个域排序

* 1. 实例：

cat /etc/passwd | sort

查看/etc/passwd里的内容并且正向排序

cat /etc/passwd | sort -t: +2n -2n

查看/etc/passwd里的内容，并且以冒号分隔，第3列（uid）按数字正向排序

cat /etc/passwd | sort -t: +2nr -2n

查看/etc/passwd里的内容，并且以冒号分隔，第3列（uid）按数字反向排序

last | sort -u

查看以前登录信息，并合并相同行

cat passwd|sort -t: -k3 -n

查看passwd文件按照冒号分隔开的第三个域排序

1. **掌握uniq功能：过滤重复行**

uniq命令可以去除排序过的文件中的重复行，因此uniq经常和sort合用。也就是说，为了使uniq起作用，所有的重复行必须是相邻的。

* 1. 选项：

-d：仅显示重复行

-u：仅显示非重复行

-c：显示行连续重复的次数

* 1. 实例：

last | cut -d“ ” -f1 | uniq –u

列出以前用户登录的信息的，以空格分隔的第1列，并且只显示不重复行

last | cut -d“ ” -f1 | uniq –d

列出以前用户登录的信息的，以空格分隔的第1列，并且只显示有重复数据行，每种重复行只显示一行

last | cut -d“ ” -f1 | uniq –c

列出以前用户登录的信息的，以空格分隔的第1列，并且打印每一重复行出现次数

1. **掌握wc功能：计数**
   1. 选项：

-l : 多少行

-m : 多少字符

-w : 多少字（空格，制表符，换行符隔开的）

* 1. 实例：

cat /etc/passwd | wc -l

查看/etc/passwd里有多少行

1. **掌握find命令的用法**
2. find命令选项基本格式

find 目录 -选项 动作[-print -exec -ok ...]

1. find命令基本选项与使用方法
   * 1. -name 按照文件名查找文件。
     2. 通配符：\*：任意长度的字符串 ?:任意一个字符 [ ]：取值范围内的一个字符 [0-9]数字 [a-zA-Z]字母
     3. -perm 按照文件权限来查找文件。
     4. -user 按照文件属主来查找文件。
     5. -group 按照文件所属的组来查找文件。
     6. -nogroup 查找无有效所属组的文件，即该文件所属的组在/etc/group中不存在。
     7. -nouser 查找无有效属主的文件，即该文件的属主在/etc/passwd中不存在。
     8. -newer file1 ! newer file2 查找更改时间比文件file1新但比文件file2旧的文件。
     9. -type 查找某一类型的文件，诸如：
        + 1. d - 目录。
          2. l - 符号链接文件。
          3. f - 普通文件
     10. -size n：[c] 查找文件长度为n块的文件，带有c时表示文件长度以字节计。
     11. -mtime -n +n 按文档更改时间来查找文档，-n指n天以内，+n指n天以前（modify）
     12. -atime -n +n 按文档访问时间来查（access）
     13. -ctime -n +n 按文档创建时间来查找文档，-n指n天以内，+n指n天以前 （create）
     14. amin cmin mmin 分别是按照访问 创建 更改文件的分钟计时
     15. -maxdepth n 查找的最大目录深度 层数
     16. -mindepth n 查找的最小目录深度 层数
2. find命令选项的动作，与实现
   * 1. -print： 将匹配的文件输出到标准输出。
     2. -fprint 文件名：将匹配的文件输出到文件。
     3. -exec：对匹配的文件执行该参数所给出的命令。相应命令的形式为'command' { } \;，注意{ }和\；之间的空格。
     4. -ok： 和-exec的作用相同，只不过以一种更为安全的模式来执行该参数所给出的命令，在执行每一个命令之前，都会给出提示，让用户来确定是否执行。
3. find案例一：查找当前目录下所有名称以.sh结尾的文件，判断是否具备执行权限

#!/bin/bash

find . -name "\*.sh" -fprint file

for filename in `cat file`

do

if [ -x $filename ];then

echo "$filename 具有执行权限"

else

echo "$filename 不具有执行权限"

fi

done

1. find案例二：在/logs目录中查找更改时间在5日以前的文件并删除它们：

$ find logs -type f -mtime +5 -exec rm {} \;

1. find案例三：在当前目录中查找所有文件名以.conf结尾、更改时间在5日以上的文件，并删除它们，在删除之前先给出提示。

$ find . -name "\*.conf" -mtime +5 -ok rm {} \;

< rm ... ./conf/httpd.conf > ? n

按y键删除文件，按n键不删除。

1. xargs命令的作用是将参数列表转换成小块分段传递给其他命令，以避免参数列表过长的问题。

案例：查找系统中的每一个普通文件，然后使用xargs命令来测试它们分别属于哪类文件

find . -type f -print | xargs file

1. shell脚本判断语句
   1. if单分支
      1. if单分支脚本举例
      2. if单分支语法分析
      3. if语句判断文件存在与否实例
   2. if双分支
      1. if双分支脚本举例
      2. if双分支语法分析
      3. if双分支应用实例
2. 判断并进行用户切换实例
3. 判断系统服务运行状态实例
4. 判断系统登录用户数实例
5. 文件备份实例
   1. if多分支语句
      1. if多分支脚本举例
      2. if多分支语法分析
      3. if多分支httpd服务应用实例
   2. case语句
      1. case脚本举例
      2. case语句语法分析
      3. case语句应用实例
6. 服务启动脚本
7. 文件类型判断

【考核要求】

1. **掌握表达式测试包括字符串测试、整数测试、文件测试及逻辑测试**
2. **掌握字符串测试**
   1. 格式：

[ str1 opr str2 ] //注意方括号左右两边的空格

[ opr str ]

opr取值：

相等：= //判断两个字符串是否相等

不等：!= //判断两个字符串是否不等

空串：-z //判断字符串长度是否为0，即判断字符串是否为空，为空返回真，非空返回假

非空串：-n //判断字符串长度是否不为0，即判断字符串是否非空，为空返回假，非空返回真

* 1. 例子：

name=tom; [ -z $name ]; echo $?

name2=andy; [ $name = $name2 ]; echo $?

1. **掌握整数测试**
2. 格式：

test "num1" opr "num2"

[ "num1" opr "num2" ]

opr 取值：

相等：-eq

不等：-ne

大于：-gt

小于：-lt 【l是字母L的小写】

小于等于：-le

大于等于：-ge

整数测试也可使用let命令或双括号(( ))，相关操作符为：== ，!=，>，>=，<，<=

(let 和双圆括号中可以使用算数表达式，而中括号不能，let和双圆括号中，操作符两边可以不留空格)

1. 例子：

x=1;[ $x -eq 1 ];echo $?

x=1; let "$x==1";echo $?

x=1;(($x+1>=2));echo $?

1. **掌握文件测试**
2. 格式：[ opr file ]

opr取值：

目录： -d

普通文件： -f

链接： -L

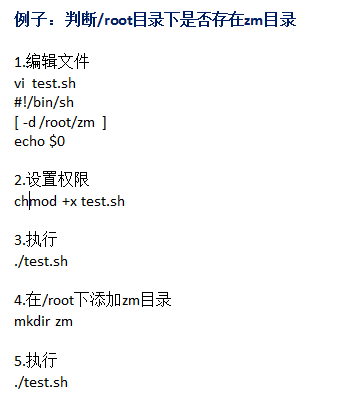
可读： -r

可写： -w

可执行： -x

文件非空： -s

1. 例子：



1. **掌握逻辑测试**
2. 格式

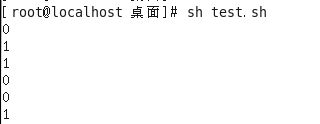
逻辑与： -a 格式： [ condition1 -a condition2 ]

逻辑或： -o 格式： [ condition1 -o condition2 ]

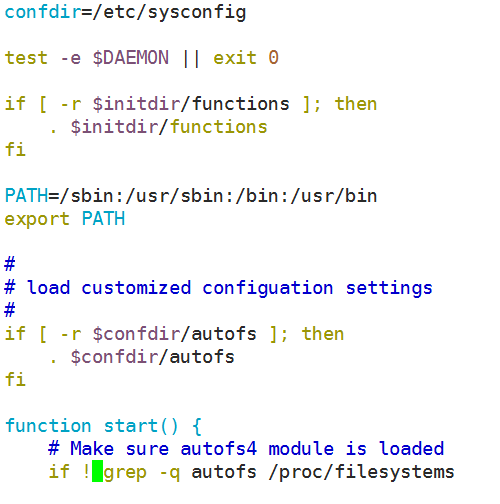
逻辑否： ! 格式： [ ! condition ]

1. 例子：





1. **掌握if语句的应用范围：**主要包括对字符串的判断、对整数的判断、对文件的判断和逻辑判断等。
2. **掌握if条件语句**
3. if单分支举例：

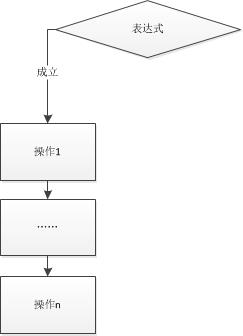


单分支if语句格式:

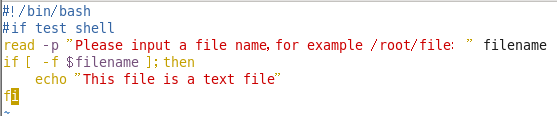
if [ 表达式 ]; then

操作

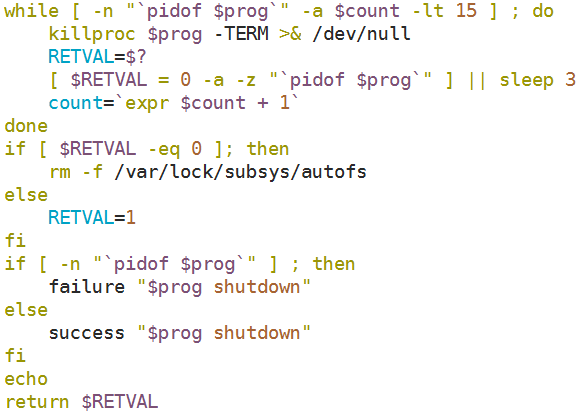
fi



if单分支实例：



1. if双分支举例：



双分支if语句格式：

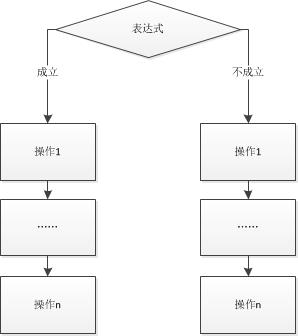
if [ 表达式 ]; then

操作1

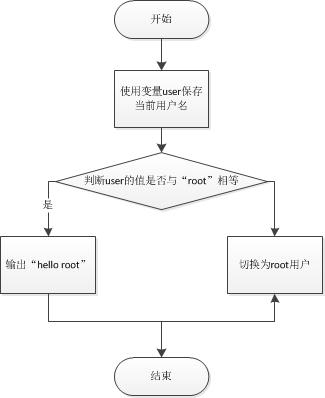
else

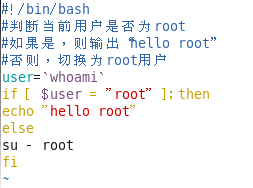
操作2

fi

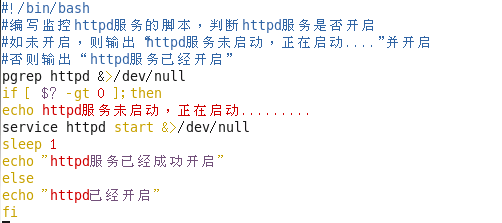


if双分支实例一：

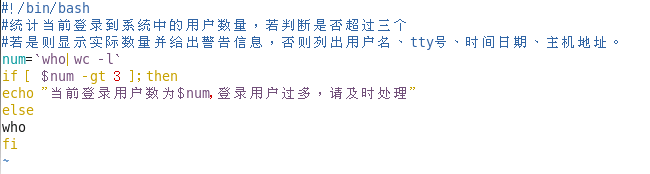




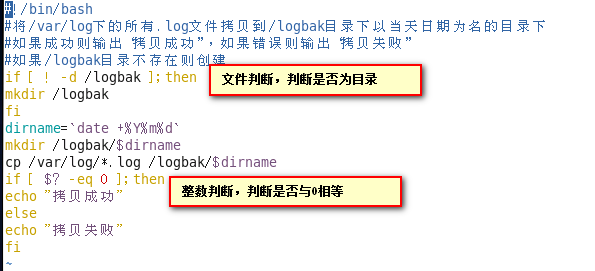
if双分支实例二：



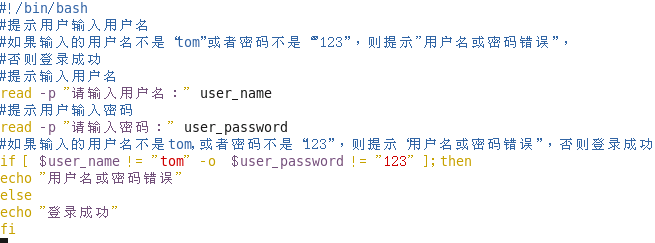
if双分支实例三：



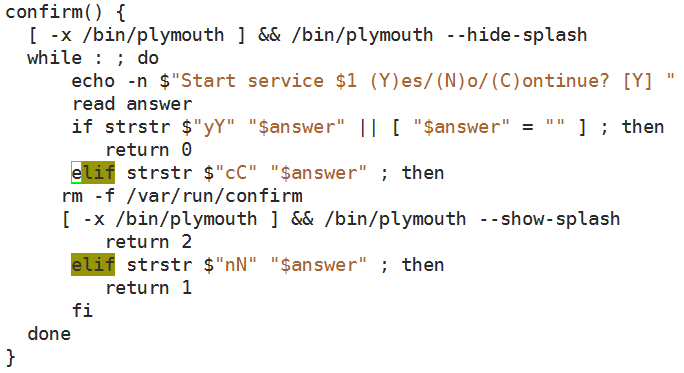
if双分支实例四：



if双分支实例五：



多分支if语句举例



if [ 表达式 ]; then

操作1

elif [ 表达式 ]；then

操作1

else

操作2

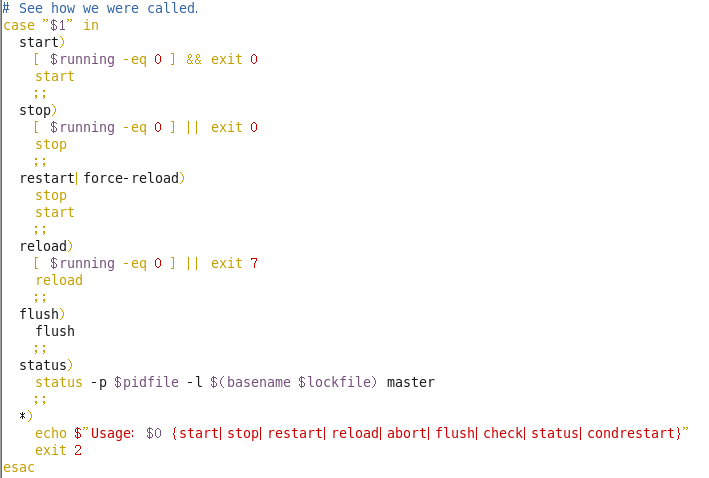
fi



if多分支语句案例：



1. 语法规范
   1. if与[之间要有空格
   2. [ ]与判断条件之间也必须有空格
   3. ]与；之间不能有空格
   4. 以fi结尾，表示if语句的结束
2. **掌握case语句**
3. case语句举例：



case语句格式：

case 表达式 in

值1|值2)

操作 ;;

值3|值4)

操作 ;;

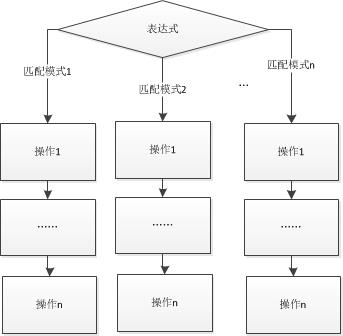
值5|值6)

操作 ;;

\*)

操作;;

esac



1. case语句的语法规范
2. 表达式expr按顺序匹配每个模式，一旦匹配成功，则执行该模式后面的命令块，然后退出case语句
3. 如果没有找到匹配的模式，则执行默认值“\*）”后的命令块， “\*）”可以没有
4. 模式中可以包含通配符或“|”,如果多个模式对应同一个命令块，则使用“|”将各个模式分开
5. 每个模式必须以右圆括号”）”结束
6. 每个命令块必须以双分号（;;）结束，双分号可独占一行，也可放在最后一个命令的后面
7. 以esac结尾，表示case语句的结束
8. **case语句案例**
9. 掌握case语句案例一：编写服务的启动脚本



1. 掌握case语句实例二：



1. **掌握if语句的嵌套使用**
2. 格式

if [ 表达式 ]; then

操作1

else

if [ 表达式1 ]; then

操作2

elif [ 表达式2 ]；then

操作3

else

操作4

fi

fi

或

if [ 表达式 ]; then

if [ 表达式1 ]; then

操作1

elif [ 表达式2 ]；then

操作2

else

操作3

fi

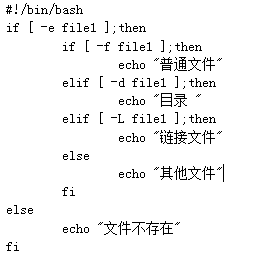
else

操作4

fi

1. if语句嵌套实例

if语句的嵌套实例一：判断file1文件是否存在，再进一步判断是普通文件，目录，链接文件或是其他，显示文件类型判断的结果



掌握if语句的嵌套实例二：



1. shell脚本循环语句
   1. for循环语句
      1. for循环语句语法分析
      2. for循环实例应用

输出指定序列

批量用户处理

* 1. while循环语句
     1. while循环语句语法分析
     2. 使用while循环产生序列实例
  2. until语句
     1. until语句语法分析
     2. 使用until产生序列实例
  3. break与continue语句
     1. break语法分析与实例
     2. continue语法分析与实例
  4. shell数组
     1. 数组的定义
     2. 数组特性的引用
  5. Shell函数
     1. Shell函数的定义
     2. Shell函数的调用

【考核要求】

1. **掌握shell中常见的循环语句：**for语句，while语句和until语句。
2. **掌握seq命令：**用于产生从某个数到另外一个数之间的所有整数

例子：

seq 1 10

结果是1 2 3 4 5 6 7 8 9 10

1. **掌握for语句**
   1. 格式1：

for 变量 in 列表

do

操作

done

* 1. 格式2：

for (( 初始化表达式； 条件表达式； 更新循环变量表达式 ))

do

循环语句

done

* 1. 例子:循环输出1到10

for i in `seq 1 10`

do

echo $i

done

* 1. 例子：循环输出1到10

for (( i=1; i<=10; i++ ))

do

echo $i

done

* 1. 例子：创建user01、user02 ...user10共10个用户

#!/bin/bash

for ((i=1;i<=10;i=i+1))

do

echo "正在创建第$i个用户"

useradd user$i

done

1. **掌握while语句**
   1. 格式：

while 表达式

do

操作

done

注意：只有表达式为真，do和done之间的语句才会执行，表达式为假时，结束循环（即条件成立就一直执行循环）

* 1. 例子：循环输出1到10的数字

#!/bin/bash

myvar=1

while [ $myvar -le 10 ]

do

echo $myvar

myvar=$(( $myvar + 1 ))

done

1. **掌握until语句**
   1. 格式：

until 表达式

do

操作

done

注意：重复do和done之间的操作，直到表达式成立为止（即只要条件成立就停止执行循环）

* 1. 例子：循环输出1到10的数字

#!/bin/bash

myvar=1

until [ $myvar -gt 10 ]

do

echo $myvar

myvar=$(( $myvar + 1 ))

done

1. **掌握break语句和continue语句**
   1. 可以使用关键字“break”来跳出循环，可以使用关键字“continue”来不执行当前循环，然后跳到下一次循环继续执行
   2. 例子1：创建文件file1，file2，file4，file5

#!/bin/bash

for i in `seq 1 5`

do

if [ $i –eq 3 ];then

continue

else

touch file$i

fi

done

* 1. 例子2: 创建文件file1，file2

#!/bin/bash

for i in `seq 1 5`

do

if [ $i –eq 3 ];then

continue

else

touch file$i

fi

done

1. **掌握数组中元素、索引、下标**
   1. 例子：

已知，array=( 1 3 9 19 43 56)

其中数组名为array

数组中的元素分别为 1、3、9、19、43、56

array[0]、array[1]分别表示第1个元素、第2个元素，其中方括号中的0、1表示下标或索引

* 1. 掌握数组的赋值

方法一：

数组名=(值1 值2 值3…)

例子

array=(tom jack rose)空格隔开的字符串依次赋值给数组每个元素

array=([0]=tom [1]=jack [2]=rose [6]=natasha)

方法二：（下标可以不连续）

array[0]="tom"

array[1]="jack"

array[2]="rose"

array[6]="natasha" 注：7个元素0-6，中间未赋值的为空

* 1. 掌握数组的调用

${数组名[下标值]}

* 1. 掌握在数组中可以使用\*或@符号来代替下标，此时\*或@为通配符，所以可以用array[\*]或array[@]数组中的所有元素
  2. 掌握数组中元素个数：${#array[\*]}或者${#array[@]} 注：是非空元素个数
  3. 掌握数组中单个元素的长度：${#array[n]}，其中n为元素下标
  4. 掌握数组使用实例：找出一组数据中的最大数，这组数据用数组保存。

#!/bin/bash

array=(1 2 3 4 5 6 7 8 9 0 11 10 19 18)

max=${array[0]}

index=${#array[\*]}

for i in `seq 1 $index`

do

if [ $max -lt $[array[$i]] ];then

max=${array[$i]}

fi

done

echo $max

1. **shell函数**
2. 掌握什么是函数：

将完成特定功能的代码封装，并命名，即函数，在使用时再进行调用

1. 函数的定义格式：

function 函数名( )

{

函数体

}

或

函数名( )

{

函数体

}

说明：可以不带function 函数名()定义，也可以直接函数名()定义

1. 函数调用方式为:

函数名 参数列表 参数存放在$1-$9)

1. 函数的使用实例:

编写一个求和函数,求两数之和

#!/bin/sh

add()

{

a=$1

b=$2

z=`expr $a + $b `

echo "The sum is $z"

}

add 8 9 17

说明：给函数传递参数: 类比于脚本的位置参数

1. 函数的返回值可以是：
   1. 函数执行返回值，是执行函数体的终端输出内容。

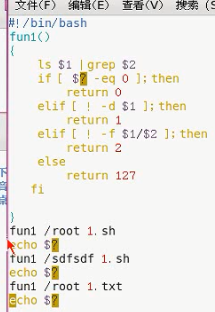
例子：将函数的输出结果（例如echo的输出），赋值给变量 例a=`sum 10 20`

* 1. 函数执行状态返回值，函数体内最后一条命令执行成功与否，即 $?，只能体现函数内最后执行那条命令成功与否。
  2. 自定义函数返回值: 用return定义返回值，return后的值会赋值给$?

说明return # 范围0-255，进一步判断函数功能实现与否

1. 函数实例：





1. 正则表达式
   1. 正则表达式
      1. 正则表达式的概念
      2. 基础正则表达式
      3. 扩展正则表达式
   2. grep与egrep文本处理工具详解
      1. grep与egrep命令应用场景
      2. grep与egrep命令详解与举例
   3. sed文件处理工具剖析
      1. sed应用场景
      2. sed工具详解
      3. sed应用案例

【考核要求】

1. **正则表达式**
2. 了解什么是正则表达式：

正则表达式由普通字符和元字符组成的表达式, 通过此模式，用来匹配特定的内容。

普通字符包括大小写的字母和数字。

元字符，是一些特殊符号，这些字符并不表示字面上的含义, 而会被解释为一些特定的含义.

正则表达式区分大小写。

正则表达式主要用于简化管理员的工作负担。

1. 掌握正则表达式分类：

基本正则表达式：Basic REGEXP

扩展正则表达式：Extended REGEXP

1. 掌握元字符：
   * 1. 匹配字符：

. ：匹配任意单个字符

[ ]：匹配指定范围内的任意单个字符

[^ ]：匹配指定范围i外的任意单个字符（出现在[ ]中的^代表排除）

例：^[0-9] 匹配行首字符为数字的

[^ ] ^出现在[]内 代表排除“非” 举例：[^0-9] 排除纯数字 [^a-z] 排除纯字母

[0-9]： 数字

[a-z]：小写字母

[A-Z]：大写字母

* + 1. 匹配次数：

\*：匹配其前面的一个字符任意次，包括0次 则.\*：匹配任意长度的任意字符

\?：需加\转义字符使用，匹配其前面的字符1次或0次 可有可无

\｛m，n\｝ ：注意花括号外要有转义字符，防止被shell解释，匹配其前面字符至少m次，至多n次

* + 1. 匹配位置：锚指定正则表达式所要匹配的文本在文本行中所处的位置。

^：锚定行首，此^字符后面的内容必须出现在行首

$：锚定行尾

^$：空行

\< :锚定词首，其后面的任意字符必须作为单词首部出现，单词：不出现特殊符号的字符组合

\> :锚定词尾，其后面的任意字符必须作为单词尾部出现

\b :匹配单词的边界

* + 1. 分组：

\（\）:将内容分组，作为整体

实例：m\（ab\）\* 将ab作为整体，m后ab可以出现0或任意次，小括号目的是为了以后引用此小括号处匹配到的内容

\1: 引用第一个小括号所包括的所有内容

\2: 引用第二个小括号所包括的所有内容

\3: 引用第三个小括号所包括的所有内容

1. 了解扩展正则表达式：
   1. 字符匹配：

. [ ] [^ ]

* 1. 次数匹配：

\*

？：不需要转义字符 \

+ ：匹配其前面的字符至少一次

* 1. 位置锚定：

^、$、\<、\>

* 1. 分组：

（）：不需要转义字符 \ ，分组

\1 \2 \3

* 1. |：或者

a|b

例：C|cat 匹配C或者cat

（C|c）at 匹配Cat或者cat

1. **grep与egrep文本处理工具详解**
2. 了解grep与egrep定义与应用场景

根据模式，搜索文本，并将符合模式的文本行显示出来，实现文本查找，文本处理，所以在shell脚本编程或者系统管理中，只要涉及到查找符合一定条件的文本行时即可使用。

1. 掌握grep语法格式

grep [option] pattern file

grep 选项 模式 文件

1. 掌握grep命令常用选项：

-i:忽略大小写

-v:反向查找，显示没有匹配模式的行

-c: 仅显示匹配行数，不显示匹配内容

-n: 显示行号

-o：只显示被模式匹配的字符串

-r：递归，读取目录下所有文件进行匹配

-l: 不显示输出的匹配结果，仅显示匹配的文件名

--color :加颜色

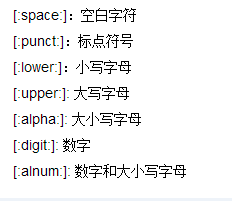
1. 掌握文件名通配符：

\*：任意长度的任意字符

？：任意单个字符

[ ]：指定范围内

[^ ]：指定范围外



1. 实例

实例一：

grep 'a.\{1,3\}b' test.txt 匹配ab之间1到3个任意字符 .和｛｝ 组合限定指定范围的任意字符串

实例二：

[root@localhost 15]# grep '\<mail\>' passwd

Mail

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

实例三：

行首至少一个空白字符的行

grep -E “^[[:space:]]+” /boot/grub/grub.conf

｛m，n｝：不需要转义字符 \

1. egrep的用法与grep基本一样，与grep –E等同，后面都是使用扩展正则表达式

实例：

egrep “^[[:space:]]+” /boot/grub/grub.conf

egrep '(ro+t).\*\1' passwd

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

1. **sed文件处理工具剖析**
2. 了解sed简介与应用场景

sed 是一种在线编辑器，它一次处理一行内容。处理时，把当前处理的行存储在临时缓冲区中，称为“模式空间”（pattern space），接着用sed命令处理缓冲区中的内容，处理完成后，把缓冲区的内容送往屏幕。接着处理下一行，这样不断重复，直到文件末尾。文件内容并没有改变，除非你使用重定向存储输出。Sed主要用来自动编辑一个或多个文件；简化对文件的反复操作；编写转换程序等。

经常用于在对文件中符合要求文本进行免交互的操作，如打印、删除、字符串替换、配置文件选项的批量自动修改等。

1. 掌握定址

可以通过定址来定位你所希望编辑的行，该地址用数字构成，用逗号分隔的两个行数表示以这两行为起止的行的范围（包括行数表示的那两行）。如1，3表示1，2，3行，美元符号($)表示最后一行。也可通过模式来确定，或行号和模式的结合

例子：

x x为行号，如1

x,y 表范围，如2,5表示2到5行

/pattern/ 匹配到此模式的行

/pattern/,/pattern/ 包含两个模式的行

x,/pattern/ 通过行号或模式定位

$ 最后一行

1. 掌握sed调用

选项：-i修改文本内容 -n取消默认输出 -e承接多个筛选条件

方法一:

sed [options] 'addresscommand' file(s)

方法二:

sed [options] -f scriptfile file(s)

1. 掌握sed常用命令：

p:打印

d:删除

s:替换（字符串被替换） g:全局模式，所有匹配的字符串都被替换（不加g默认只替换每行匹配的第一个） p与-n选项结合，实现只打印被替换行

y:变换字符（一一对应）

i:插入（匹配行前）

a:追加（匹配行后）

c:修改（匹配行被修改）

r:从文件读入

w:写入文件

q:找到第一个匹配就退出

1. sed 使用实例：
2. p:打印即显示

sed -n -e '/root/p' /etc/passwd 显示包括root的行

sed -n -e '1,10p' /etc/passwd 显示1到10行

sed -n -e '/^root/，$p' /etc/passwd 显示root开头行到最后一行

sed -n -e '/^root/，/^tom/p' /etc/passwd 显示root开头行到tom开头行

1. d:删除

sed -e '/bash/d' /etc/passwd 删除带有bash的行

sed -e '/^#/d' /etc/squid/squid.conf 将注释行删除

sed -e '/^#/d' -e '/^$/d'/etc/squid/squid.conf 将注释行和空行删除

1. a:追加

sed '/^root/a\hello' /etc/passwd 在行首为root的行后面换行追加hello

sed '2a/tom' name.txt 在name.txt第2行追加tom

sed '2，3a/tom' name.txt 分别在name.txt第2和3行追加tom

4）y:变换（一一对应）

sed –r 'y/abcdefg/ABCDEFG/' /etc/passwd 将小写字母abcdefg变换为大写字母ABCDEFG

1. s:替换

sed '/bash/s/root/shrek' /etc/passwd 有bash的行将root替换成shrek

sed '1，10s/bash/nologin' /etc/passwd 文件1到10行将bash替换成nologin

由于sed命令不需要交互，帮助自动更改配置文件

sed -n '/PasswordAuthentication/s/yes/no/p' /etc/ssh/sshd\_config

注：&代表正则表达式匹配到的值

sed –e 's/^root/a&a/' passwd

sed -r 's/r[ob][oa]t/ha&ah/g' passwd

1. -r：支持扩展正则表达式

sed 's/ro\([oa]\)t/ha\1ah/g' passwd 或 sed -r 's/ro([oa])t/ha\1ah/g' passwd

注：\1代表第一次（）中匹配的值，\2代表第二次（）中匹配的值，以此类推

sed -r 's/r([ob])([oa])t/ha\1\2ah/g' /etc/passwd

1. w:写入文件

sed -n '/root/w output' /etc/passwd ( 将/etc/passwd中带有root的行写入output文件 )

1. r:从文件读入

sed –n '/root/r input' /etc/passwd (将/etc/passwd中带有root的行后读入input文件内 )

1. q:找到第一个匹配就退出

sed –n '/root/q ' /etc/passwd ( 在/etc/passwd中匹配到第一个root就退出 )

1. awk文本处理工具
   1. awk高级应用
      1. awk命令详解
      2. awk内置变量
      3. awk工具BEGIN与END
      4. awk应用举例
   2. awk流程控制语句
      1. if-else语句
      2. while语句
      3. for循环语句

【考核要求】

1. **awk高级应用**
2. 了解awk是一个强大的文本分析工具，相对于grep的查找，sed的编辑，awk在其对数据分析并生成报告时，显得尤为强大。简单来说awk就是把文件逐行的读入，以空格为默认分隔符将每行切片，切开的部分再进行各种分析处理
3. 掌握常用选项：

-F 指定域分隔符（默认为空格或tab键）

-f 指定从脚本文件中读取awk命令

1. 掌握语法格式：

awk [ -F 分隔符 ] ' /模式/{操作} ' 文件名

所以说awk脚本是由模式和操作组成的。

常见的操作为：print、printf

1. 掌握工作流程：

awk工作流程是这样的：读入有'\n'换行符分割的一条记录，然后将记录按指定的域分隔符划分域，$0则表示所有域,$1表示第一个域,$n表示第n个域。

1. 掌握print：

例子：

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

1. 了解printf：格式化输出

例子：

awk -F: '{printf "%-15s %i\n",$1,$3}' /etc/passwd

awk -F: '{printf "%5d\n",$3}' /etc/passwd

1. **掌握BEGIN和END**

BEGIN和END部分在awk中都仅执行一次且有各自的用途。

BEGIN用程序一开始时, 改变awk分隔数据行的方式、设定变量的起始值、印出一行title、不需要读入任何数据行，而END用来打印结尾信息或者用来输出统计信息等。

1. **掌握awk实例：**
   1. 例子1：只显示/etc/passwd的账户名

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

* 1. 例子2：搜索/etc/passwd有root关键字的所有行，并显示对应的shell

awk -F: '/root/{print $7}' /etc/passwd

* 1. 例子3：如果只是显示/etc/passwd的账户和账户对应的shell,而账户与shell之间以逗号分割,而且在开头一行添加列名name,shell,在最后一行添加"blue,/bin/nosh"。

cat /etc/passwd |awk -F ':' 'BEGIN {print "name,shell"} {print $1","$7} END {print "blue,/bin/nosh"}'

1. **awk流程控制语句**
2. 掌握内置变量

FS 设置输入域分隔符，等价于命令行 -F选项

NF 浏览记录的域的个数

NR 已读的记录数

OFS 输出域分隔符

ORS 输出记录分隔符

1. **掌握awk中的流程控制语句-条件语句**
   1. if-else语句
      1. 格式：

if (condition) {then-body}

else

{ else-body }

* + 1. 实例：显示用户信息配置文件中用户id大于等于500的(普通用户)用户数量

awk -F: -v sum=0 '{if ($3>=500) sum++}END{print sum}' /etc/passwd

1. **掌握awk中的流程控制语句-循环语句**
2. while语句

格式:

while (condition){statement1; statment2; ...}

1. 实例：显示用户信息配置文件中$1到$3的字段

awk -F: '{i=1;while (i<=3) {print $i;i++}}' /etc/passwd

* 1. do …while语句
     1. 格式

do {statement1, statement2, ...} while (condition)

* + 1. 实例：显示$1到$3的字段

awk -F: '{i=1;do {print $i;i++}while(i<=3)}' /etc/passwd

1. **for语句**
   * 1. 格式

for ( variable assignment; condition; iteration process) { statement1, statement2, ...}

* + 1. 实例：显示用户信息配置文件中$1到$3字段

awk -F: '{for(i=1;i<=3;i++) print $i}' /etc/passwd

1. 运用expect实现自动交互
   1. expect语言
      1. expect语言介绍
      2. expect企业应用场景
      3. expect语法
   2. 应用expect实现ssh自动登录
      1. 实现思路与流程分析
      2. 脚本编程测试
   3. 使用expcet与scp实现自动文件传输
      1. 实现思路与流程分析
      2. 脚本编程测试
   4. 实现ftp自动文件传输
      1. 实现思路与流程分析
      2. 脚本编程测试

【考核要求】

1. **了解expect简介与应用场景**

EXPECT是一个用来实现自动交互功能的软件套件（EXPECT is a software suite for automating interactive tools），用于提供自动交互的工具。

设想一个场景：比如你要用ssh或者 telnet登录服务器的时候，命令行下提示需要输入用户和密码等。如果叫你用shell脚本写自动登录脚本的话是不是有点困难呢？ 你会说在shell中执行ssh命令的话，他会一直停在那里等你密码啊，这个要手工来做才行吧？

Expect就是为处理这类情况而设计的工具，简单来说它可以控制、处理输入，输出流，然后提供自动填写数据等需要用户交互式输入的数据的地方实现自动化处理。Expect就是为了处理“自动交互”的工具。

Expect 支持自定义的脚本，所以你通过编写自定义脚本就可以实现你想要的自动填写数据功能了。

1. **expect的工作流程**

expect的工作流程可以理解为，spawn启动进程---->expect期待关键字---->send向进进程发送字符--->退出结束。

1. **掌握except语法**
   1. spawn

spawn命令是expect的初始命令，它用于启动一个进程，之后所有expect操作都在这个进程中进行，如果没有spawn语句，整个expect就无法执行了，spawn使用方法如下：

spawn ssh [root@192.168.2.5](mailto:root@192.168.2.5)

// 在spawn命令后面，直接加上要启动的进程、命令等信息

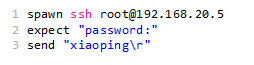
* 1. expect

expect 表达式 动作 表达式 动作。。。。。。。

expect 命令用于等候一个相匹配内容的输出，一旦匹配上就执行expect后面的动作或命令，如

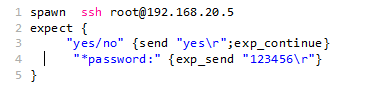


从上面的例子可以看出。expect是依附与spawn命令的，当执行ssh命令后，expect就匹配命令执行后的关键字： password:，如果匹配到关键字就会执行后面包含在{}括号中的exp\_send动作，匹配以及动作可以放在二行，这样就不需要使用{}括号了，就像下面这样，实际完成的功能与上面是一样的



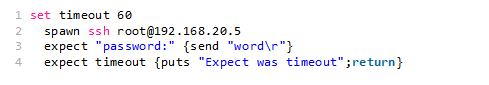
* 1. exp\_send和send

exp\_send命令或send命令是expect中的动作命令，使用方法如下：



* 1. timeout

timeout 是expect中的一个重要变量，它是一个全局的时间控制开关，你可以通过为这个变量赋值来规定整个expectc操作的时间，注意这个变量是服务与expect全局的，它不会纠缠某一条命令，即使命令没有任何错误，到时间依然会激活这个变量。使用方法如下



timeout 变量中，设置为0表示立即超时，-1则表示永久不超时。

* 1. expect eof

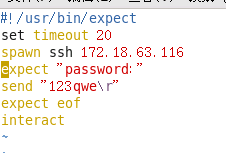
与spawn对应表示捕获终端输出信息终止

1. 案例一：使用expect实现ssh自动登录

实现思路：

按照一般ssh登录的步骤，可得到实现思路：执行ssh命令远程登录ssh服务器🡪等待ssh服务器端返回输入用户名与密码的界面🡪输入用户名与密码实现登录

编程实现：

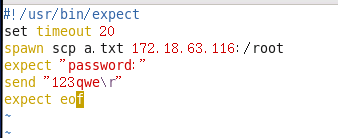


1. 案例二：使用expect与scp远程传输文件

思路分析：

启动scp命令🡪服务端返回输入用户名与密码界面🡪发送用户名密码完成登录

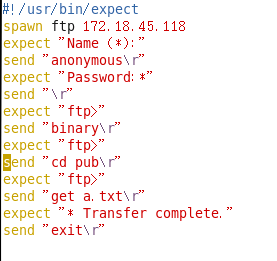
编程实现：



1. 案例三：使用expect自动操作ftp

实现思路：启动命令🡪服务器返回输入用户名密码界面🡪发送用户名密码完成登录🡪服务器返回命令操作界面🡪发送操作命令🡪退出

编程实现：



1. Linux作业调度机制
   1. Linux计划任务at命令
      1. at应用场景
      2. at命令选项解读
      3. at命令应用举例
   2. Linux计划任务crond命令
      1. crond应用场景
      2. crond命令常用选项
      3. crond应用举例
   3. Linux进程控制
      1. 前台进程与后台进程
      2. 信号的概念解读
      3. Linux进程的控制
   4. Linux作业调度
      1. jobs命令
      2. fg、bg、&、nohup

【考核要求】

1. **Linux计划任务at命令**
2. 掌握at定时任务应用场景：在未来的某个时间点执行一次某个任务，如果在企业管理中，有一个任务要在以后的一个时间里运行一次，那就使用at命令设置任务执行时间与执行的任务便可，如果任务比较复杂，可将命令写成脚本并在at命令中调用
3. 掌握编写at定时任务
4. at 时间 回车

at>命令

at>Ctrl+D结束

1. 指定时间支持多种形式：

绝对时间：HH：MM DD.MM.YY MM/DD/YY

相对时间：now+3m tomorrow +3days 单位：minutes hours days weeks

模糊时间：noon midnight teatime(下午四点)

1. 掌握查看at作业列表：

命令：at -l或atq

作业号 执行时间 队列（将不同作业放入不同队列分类） 用户

1. 掌握删除at作业：

命令：at -d或atrm 作业号

例：at -d 2

1. **Linux计划任务crond命令**
2. 了解crond服务：是一个不间断运行的服务（确保crond服务开启）
3. 应用场景：crond服务可用于周期性执行指定任务，比如有一个任务在每周五晚上2点执行，可使用这个服务进行设置
4. 掌握编写cron计划任务的格式

格式：分 时 天 月 周 用户 任务

时间的通配举例：

\*：对应的所有有效取值

3 \*　\* \* \* 每小时的第3分钟

3 \*　\* \* 7 每个周日的每小时的第3分钟

3 12　\* \* \* 每天的12点3分

3 12　\* \* 5 每周五的12点3分

3 12　6 \* \* 每月6号的12点3分

3 12　6 7 \* 每年7月6号的12点3分

，：离散时间点取值

10 ，40 \* \* \* \* 每小时的第10分钟和第40分钟

10 ，40 \* \* \* 2，5 每周二和周五的每小时的第10分钟和第40分钟

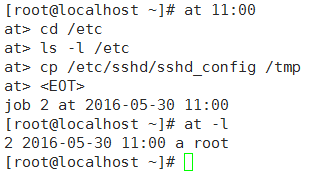
-：连续时间点

10 02 \* \* \* 1-5 每周一到五的2点10分

/：对应取值范围每多久一次

\*/3 \* \* \* \* 每3分钟

at 应用举例



1. 掌握管理cron计划任务
2. 编辑

crontab –e

1. 查看

crontab -l

1. 移除所有（移除一条：编辑中修改）

crontab –r

1. 管理其他用户的计划任务

crontab -u username

1. crond应用举例：

59 23 \* \* \* /etc/init.d/httpd restart //表示每晚的23:59重启apache

45 4 1 \* \* /etc/init.d/httpd restart //表示每月1日的4:45分重启apache

45 4 1,10,22 \* \* /etc/init.d/httpd restart //表示每月1、10、22日的4 : 45重启apache

10 1 \* \* 6,0 /usr/local/apache/bin/apachectl restart //表示每周六、周日的1 : 10重启apache

0,30 18-23 \* \* \* /usr/local/apache/bin/apachectl restart //表示在每天18 : 00至23 : 00之间每隔30分钟重启apache

\* \*/1 \* \* \* /usr/local/apache/bin/apachectl restart //每一小时重启apache

\* 23-7/1 \* \* \* /usr/local/apache/bin/apachectl restart //晚上11点到早上7点之间，每隔一小时重启apache

1. **Linux进程控制**
2. 了解信号是进程之间传递信息的方式
3. 掌握生成信号
   1. 中断进程

使用ctrl+c组合键可以生产SIGINT信号比如用sleep命令测试

[root@wzp ~]# sleep 100

如果我不使用组合键那么控制台就无法进行输入了一直运行该sleep程序，所以通过这方法可以终止进程。

* 1. 暂停进程

有些进程想暂停而不是终止它可以使用ctrl+z组合键生产SIGTSTP信号

[root@wzp ~]# sleep 100

[1]+ Stopped sleep 100

看到没有，如果是暂停进程会有log信息显示stopped的。

如上可以看到中括号里面有一个1数值就是shell分配的作业编号，第一个启动的进程分配作业编号1，第二个启动的进程分配作业编号2，依此类推，如果shell会话中存在停止的作业，退出shell时会发出警告信号的

1. 掌握trap捕捉信号的格式：

trap '命令' 信号列表

1. 掌握trap捕捉信号实例：捕捉中断信息和退出信号

#!/bin/bash

trap "echo 程序运行过程中不可中断" SIGINT

trap "echo 程序执行完毕" exit

for i in `seq 1 5`

do

if ping -c 1 172.18.7.$i&>/dev/null;then

echo 172.18.7.$i is up

else

echo 172.18.7.$i is down

fi

done

1. **Linux作业调度**
2. 了解前台进程和后台进程

前台进程需前台控制，所以只有等进程结束才能回到输入命令提示符状态。

而后台不需前台控制，启动后可以回到命令提示符状态继续执行其它进程。

1. 可以掌握使用&命令把作业放到后台执行。
   1. 格式

命令 &

* 1. 实例：

vim /etc/passwd &

用cat命令在后台查看/etc/passwd 文件

1. 掌握jobs命令

查看当前有多少在后台运行的命令

1. 掌握fg命令

将后台中的命令调至前台继续运行

* 1. 实例

fg 2

将后台作业号为2的作业调至前台继续运行

1. 掌握bg命令

将一个在后台暂停的命令，变成继续执行

* + 1. 实例

bg 3

将后台暂停的作业变成后台继续执行

1. 掌握nohup

如果正在运行一个进程，而且在退出帐户时（exit）该进程还不会结束，那么可以使用nohup命令。该命令可以在退出帐户之后继续运行相应的进程。 nohup就是不挂起的意思( no hang up)。

1. 格式

nohup command &

1. 实例

nohup sleep 300 &

1. 掌握ps -aux命令：显示系统中所运行进程的详细信息
2. 掌握kill命令
   * 1. 停止系统中的进程运行

格式

kill 进程号

示例: kill 213

* + 1. 强制停止系统中的进程运行

格式

kill -9 进程号

示例: kill -9 2135

* + 1. 结束后台作业

格式

kill %作业号

示例：kill %2

1. 联鑫科技公司脚本应用案例
   1. mysql备份脚本
      1. 备份场景说明
      2. 脚本编程思路与流程分析
      3. 备份脚本实现
   2. 文件备份脚本
      1. 场景说明
      2. 编程思路
      3. Apache、ftp文件备份脚本实现
   3. 服务状态监控脚本
      1. 场景说明
      2. 编程思路与流程分析
      3. ftp、httpd、mysql状态监控脚本编程实现
   4. 服务器运行状态监控脚本
      1. 场景说明
      2. 思路分析
      3. 脚本编程实现

【考核要求】

* + - 1. **案例一：**备份MYSQL数据库，每天凌晨12:00完全备份（自动）

1. 应用场景分析：公司使用的是mysql数据库，保存着公司里所有重要的数据，如果数据库服务或者服务器出现问题，那对整个公司将是极大的损失，作为系统工程师的你，要负责数据库的备份。手动备份工作效率低下并且繁琐，所以要编写数据库备份脚本，以实现全自动备份数据库
2. 思路流程分析：

设置crond服务，指定周期执行脚本的时间

设置变量以保存要备份的数据库的库名或者表名🡪设置保存备份文件的变量🡪判断备份目录是否存在🡪数据库备份并压缩🡪删除过时的备份文件🡪删除未压缩的备份文件

1. 脚本编程调试

crontab -e

00 00 \* \* \* sh /mysqlbackup.sh

vim /mysqlbackup.sh

#!/bin/bash

export PATH=/usr/lib64/qt-3.3/bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin

# 要备份的数据库名，多个数据库用空格分开

databases=(db1 db2 db3)

# 备份文件要保存的目录

basepath='/root/backup/mysql/'

if [ ! -d "$basepath" ]; then

mkdir -p "$basepath"

fi

# 循环databases数组

for db in ${databases[\*]}

do

# 备份数据库生成SQL文件

/usr/bin/mysqldump -uUSER -pPASSWORD --database $db > $basepath$db-$(date +%Y%m%d).sql

# 将生成的SQL文件压缩

tar zPcf $basepath$db-$(date +%Y%m%d).sql.tar.gz $basepath$db-$(date +%Y%m%d).sql

# 删除7天之前的备份数据

find $basepath -mtime +7 -name "\*.sql.tar.gz" -exec rm -rf {} \;

done

# 删除生成的SQL文件

rm -rf $basepath/\*.sql

**2. 案例二：**备份apache配置文件，FTP配置文件到/home/myback/日期目录每天零点自动备份

1. 应用场景：对于公司重要的配置文件，为了在服务出现意外时能快速响应并第一时间恢复服务，需要定时备份公司业务中重要的文件，如公司网站的服务的配置文件、网页数据文件、公司员工工作时上传的重要文件
2. 实现思路：

设置crond服务，指定脚本执行时间

设置备份目录，与系统时间一致🡪备份文件🡪压缩文件🡪删除中间备份文件

1. 编程调试

crontab -e

00 00 \* \* \* sh /myback.sh

#!/bin/bash

export PATH=/usr/lib64/qt-3.3/bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin

echo"今天的日期是"

date

dir=$(date '+%Y%m%d')

mkdir –p /home/myback/$dir

cp /etc/httpd/conf/httpd.conf /home/myback/$dir

cp /etc/vsftpd/vsftpd.conf /home/myback/$dir

tar –zcvf /home/myback/http-ftp-back-$dir.tar.gz /home/myback/$dir

cd /home/myback

rm –rf $dir

**3. 案例三：**每周日夜里2点完全备份/etc目录到/home/myback/日期目录下，其他天数给/etc目录做增量备份，用zip压缩

1. 应用场景：

公司的/etc目录都是配置文件，如目录文件出现问题，例如文件丢失，势必会对公司造成极大影响，因此有必要对该目录进行备份，但是没有必要每次进行全量备份，只要在一周的某一天进行全量备份，其他时间进行增量备份即可

1. 实现思路与流程分析：

设置crond服务，以在指定时间执行脚本

设置备份目录相关变量🡪判断当时时间是否为指定的进行全量备份的时间，如是进行全量备份并进行压缩🡪否则进行增量备份并进行压缩

crontab -e

00 02 \* \* \* sh /myback1.sh

#!/bin/bash

export PATH=/usr/lib64/qt-3.3/bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin

echo "今天的日期是"

date

dir=$(date '+%Y%m%d')

mkdir /home/myback/$dir

if [`date'+%w'` -eq 0 ];then

cp /etc /home/myback/$dir

tar zcvf /home/myback/etc-full-$dir.tar.gz /home/myback/$dir

cd /home/myback

rm –rf $dir

else

cd /home/myback/$dir

tar -g zengliang -zcvf etc-incre-$dir.tar.gz /etc

fi

1. **案例四：**判断ftp、http、mysql的状态，如不在线自动启动上述服务
2. 应用场景：公司运行着ftp、mysql、http几个服务，是公司主要的几个服务，其中http服务支撑着公司的网站业务，与http相配合的mysql数据库、与公司员工文件相关联的是ftp服务，公司的系统工程师要时刻监控以上几个服务的运行状态，如服务没有运行则重启服务
3. 实现思路与流程分析

设置crond服务

分别检测ftp、httpd、mysql服务状态🡪判断相关服务器状态，如没有运行则将服务启动

编程调试

crontab -e

00 02 \* \* \* sh /tatus-test.sh

#!/bin/bash

export PATH=/usr/lib64/qt-3.3/bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin

echo " 检测ftp服务是否开启"

service vsftpd status &>/dev/nul

if [ `echo $?`-gt 0 ];then

echo 'ftp服务没开启，现在开启'

service vsftpd start &>/dev/null

else

echo '服务已开启'

fi

echo '================================='

echo "检测http服务是否开启"

service httpd status &>/dev/nul

if [ `echo$?` -gt 0 ];then

echo 'http服务没有开启，现在开启'

service httpd start &>/dev/nul

else

echo '服务已开启'

fi

echo '================================'

echo "检测mysql服务是否开启"

service mysqld status &>/dev/nul

if [ `echo $?` -gt 0 ];then

echo 'mysql服务没有开启，现在开启'

service mysqld start &>/dev/nul

else

echo '服务已开启'

**5、案例五：**

1. 应用场景分析：

为了持续观察公司服务器每天的基本运行状况，提供方便易读的集中的日志记录数据，需要结合Shell脚本和计划任务设置，定期记录不同时间段服务器的cpu负载、内存和交换空间、磁盘使用率等各种信息，以时刻监控服务器的运行负载，方便作出响应，及时处理服务器故障

1. 实现思路与流程分析

1)、每隔15分钟记录一次当前服务器的基本运行状况包括cpu负载、内存和交换空间、磁盘空间、最近的用户登录情况等，连同当时的时间信息(以YYYY-mm-dd HH:MM的格式输出)追加保存到日志文件“/var/log/runrec/running.today”中。

2)、每天晚上22:35时将“/var/log/running.today”文件压缩备份，保存到“/var/log/runrec”目录中，文件名使用“running.record-YYYYmmdd.tar.bz2”的格式。备份完毕后删除当天的记录文件(running.today)。

1. 编程调试

#!/bin/bash

nowday=`date +%Y%m%d`

cpufuzai=`uptime`

memuse=`free -m|grep "Mem"|awk '{print $3}'`

memfree=`free -m|grep "Mem"|awk '{print $4}'`

swapuse=`free -m|grep "Swap"|awk '{print $3}'`

swapfree=`free -m|grep "Swap"|awk '{print $4}'`

echo "当前时间：$nowday"

echo "CPU负载：$cpufuzai"

echo "内存已用：$memuse"

echo "内存剩余：$memfree"

echo "swap 已用$swapuse"

echo "swap 剩余$swapfree"

echo "磁盘使用情况"

df -h

echo "最近的十次登录"

last -10

2）crontab -e计划任务

\*/15 \* \* \* \* /root/22/2.sh>>/var/log/running.today

35 22 \* \* \* tar -jcf /var/log/runrec/running.record-`date +%Y%m%d`.tar.bz2` /var/lof/runninf.today --remove-files

注：

命令：uptime

显示结果为：

10:19:04 up 257 days, 18:56, 12 users, load average: 2.10, 2.10,2.09

显示内容说明：

10:19:04 //系统当前时间

up 257 days, 18:56 //主机已运行时间,时间越大，说明你的机器越稳定。

12 user //用户连接数，是总连接数而不是用户数

load average // 系统平均负载，统计最近1，5，15分钟的系统平均负载

系统平均负载是指在特定时间间隔内运行队列中的平均进程数。

如果每个CPU内核的当前活动进程数不大于3的话，那么系统的性能是良好的。如果每个CPU内核的任务数大于5，那么这台机器的性能有严重问题。如果你的linux主机是1个双核CPU的话，当Load Average 为6的时候说明机器已经被充分使用了。

1. 中联科技公司网站数据分析案例
   1. Apache日志格式分析
      1. 企业网站ip、pv、uv概念解读
      2. Aapche日志格式说明
   2. 网站安全监控
      1. 进程监控
      2. ip监控
      3. 访问时段监控
      4. 访问页面监控
      5. 网站点击量监控
      6. 文件传输量监控
      7. 综合脚本调试

【考核要求】

1. **理解ip、pv、uv的概念**

ip值：ip可以理解为独立IP的访问用户，指1天内使用不同IP地址的用户访问网站的数量，同一IP无论访问了几个页面，独立IP数均为1。但是假如说两台机器访问而使用的是同一个IP，那么只能算是一个IP的访问。

pv值：即页面浏览量，或点击量；网站的PV值，即网站的人均浏览次数；pv通常是衡量一个网络新闻频道或网站甚至一条网络新闻的主要指标，或者简单的说，PV值就是一个访问者在24小时(0点到24点)内到底看了网站多少个页面。这里需要强调：同一个人浏览你网站同一个页面，不重复计算pv量，点100次也算1次。

PV值对于网站来说，就像电视的收视率一样，从某种程度上已成为投资者衡量商业网站表现的最重要尺度。

PV值的计算：当一个访问者访问网站的时候，记录他所访问的页面和对应的IP，然后确定这个IP今天访问了这个页面没有。如果你的网站到了23点，单纯IP有60万条的话，每个访问者平均访问了3个页面，那么PV表的记录就要有180（60×3）万条。

当然，有时还会同时考察另外一个指标，即uv（uniquevisitor），指访问某个站点或点击某条新闻的不同IP地址的人数。在同一天内，UV只记录第一次进入网站的具有独立IP的访问者，在同一天内再次访问该网站则不计数。

1. **了解apache日志格式**

apache访问日志格式：

58.61.164.141 – - [22/Feb/2010:09:51:46 +0800] “GET / HTTP/1.1″ 206 6326 ” http://www.google.cn/search?q=webdataanalysis” “Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)”

可以看到这个日志主要由以下几个部分组成：

访问主机（remotehost）：显示主机的IP地址或者已解析的域名。

标识符（Ident）：由identd或直接由浏览器返回浏览者的EMAIL或其他唯一标示，因为涉及用户邮箱等隐私信息，目前几乎所有的浏览器就取消了这项功能。

授权用户（authuser）：用于记录浏览者进行身份验证时提供的名字，如果需要身份验证或者访问密码保护的信息则这项不为空，但目前大多数网站的日志这项也都是为空的。

日期时间（date）：一般的格式形如[22/Feb/2010:09:51:46 +0800]，即[日期/月份/年份:小时:分钟:秒钟 时区]，占用的的字符位数也基本固定。

请求（request）：即在网站上通过何种方式获取了哪些信息，也是日志中较为重要的一项，主要包括以下三个部分：

请求类型（METHOD）：常见的请求类型主要包括GET/POST/HEAD这三种；

请求资源（RESOURCE）：显示的是相应资源的URL，可以是某个网页的地址，也可以是网页上调用的图片、动画、CSS等资源；

协议版本号（PROTOCOL）：显示协议及版本信息，通常是HTTP/1.1或HTTP/1.0。

状态码（status）：用于表示服务器的响应状态，通常1xx的状态码表示继续消息；2xx表示请求成功；3xx表示请求的重定向；4xx表示客户端错误；5xx表示服务器错误。

传输字节数（bytes）：即该次请求中一共传输的字节数。

来源页面（referrer）：用于表示浏览者在访问该页面之前所浏览的页面，只有从上一页面链接过来的请求才会有该项输出，如果是新开的页面则该项为空。上例中来源页面是google，即用户从google搜索的结果中点击进入。

用户代理（agent）：用于显示用户的详细信息，包括IP、OS、Bowser等。

1. **掌握apache或者nginx各种参数监控方法**
2. 查看进程数:

ps aux | grep httpd | grep -v grep | wc -l

或 pgrep httpd|wc -l

1. 各种参数监控脚本编程调试

#!/bin/bash

log="/usr/local/nginx/logs/access.log"

#访问次数最多的10个ip

echo "max--10--ip"

awk '{print $1}' $log|sort|uniq -c|sort -nr|head -10

#或者

#cat $log |awk ‘{counts[$1]+=1}; END {for(url in counts) print counts[url], url}’

#访问量最大的十个时间段

echo "max--10--time"

awk '{print $4}' $log |cut -c 14-18|sort|uniq -c|sort -nr|head -10

#查看日志中访问次数最多的前10个IP

cat $log |awk '{print $1}' |sort |uniq -c | sort -nr | head -10

#14085 121.207.252.122

#13753 218.66.36.119

#11069 220.162.237.6

#1188 59.63.158.118

#1025 ::1

#728 220.231.141.28

#查看日志中出现100次以上的IP

cat $log |cut -d ' ' -f 1 |sort |uniq -c | awk '$1 > 100{ print $0}'｜sort -nr |less

#14085 121.207.252.122

#13753 218.66.36.119

#11069 220.162.237.6

#1188 59.63.158.118

#1025 ::1

#728 220.231.141.28

#查看最近访问量最高的页面

cat $log |tail -10000|awk '{print $7}'|sort|uniq -c|sort -nr|less

#8729 /server-status?auto

#618 /

#15 /favicon.ico

#12 /manager/html

#10 \*

#9 /top/icons.gif

#8 /www.766.com/awstats.www.766.com.html

#8 /awstatsicons/other/vv.png

#8 /awstatsicons/other/vu.png

#8 /awstatsicons/other/vp.png

#查看最近访问量最高的(.png)页面

cat $log |awk '{print $7}'|grep '.png'|sort|uniq -c|sort -nr |head -n 10

#或者cat $log |awk '$7~/.png$/{print $7}'|sort|uniq -c |sort -rn

#241 /awstatsicons/other/awstats\_logo6.png

#227 /awstatsicons/clock/hr12.png

#226 /awstatsicons/other/vv.png

#226 /awstatsicons/other/vu.png

3226 /awstatsicons/clock/hr9.png

#查看日志中访问超过100次的页面

cat $log | awk '{print $7}' | sort |uniq -c | awk '$1 > 100{ print $0}' | less

#20107 /

#1027 \*

#215 /awstatsicons/browser/chrome.png

#201 /awstatsicons/browser/netscape.png

#123 /awstatsicons/browser/notavailable.png

#226 /awstatsicons/clock/hr10.png

#226 /awstatsicons/clock/hr11.png

#226 /awstatsicons/clock/hr2.png

#226 /awstatsicons/clock/hr3.png

#查看12/Nov/2009一天的点击量(clicks);

cat $log |awk '$4~/12\/Nov\/2009/'|wc -l

#0

#查看12/Aug/2009一天访问网站的独立IP有哪些；

cat $log |grep '12/Aug/2009''|sort|uniq

#194

#统计某url(/images/index/e1.gif)，一天的访问次数

cat $log |grep '12/Aug/2009'|grep '/images/index/e1.gif'|wc -l

#或者cat $log |awk '$4~/\[21\/Oct\/2014/{print $0}'|awk '$7~/png$/{print $0}'|wc –l

1. 将以上脚本输出的结果保存到不同的文件中，将进行调试及优化即可得到公司监控脚本，进行参数监控
2. **掌握网站数据监控杂例实现方法**

查看某ip218.66.36.119在干嘛，浏览过哪些页面:

cat /usr/local/nginx/logs/access\_log | grep 218.66.36.119| awk '{print $1"/t"$7}' | sort | uniq -c | sort -nr | less

243 218.66.36.119 /

210 218.66.36.119 /awstatsicons/other/awstats\_logo6.png

198 218.66.36.119 /awstatsicons/clock/hr12.png

197 218.66.36.119 /awstatsicons/other/vv.png

197 218.66.36.119 /awstatsicons/clock/hr9.png

197 218.66.36.119 /awstatsicons/clock/hr8.png

#列出传输字节超过 30 的文件：

cat /usr/local/nginx/logs/access\_log|awk ‘($10 > 30){print $7}’|sort -n|uniq -c|sort -nr|head -20

14058 /server-status?auto

8966 /

3955 /manager/html

1025 \*

214 /www.766.com/awstats.www.766.com.html

211 /awstatsicons/other/awstats\_logo6.png

199 /awstatsicons/clock/hr12.png

198 /awstatsicons/other/vv.png

198 /awstatsicons/other/vu.png

列出传输最大的几个exe文件（分析下载站的时候常用）：

cat /usr/local/nginx/logs/access.log |awk ‘$7~/.exe/ {print $7}’|sort -nr|head -20

列出传输大于200000byte(约200kb)的exe文件以及发生次数

cat /usr/local/nginx/logs/access.log |awk ‘($10 > 200000 && $7~/.exe/){print $7}’|sort |uniq -c|sort -nr|head -100

1. 新晨科技公司邮件报警案例
   1. 邮件报警工具的使用
      1. mail邮件工具
      2. mutt邮件工具
   2. 文件备份并邮件报警实例
      1. 应用场景说明
      2. 脚本实现思路与流程分析
      3. 脚本编程测试
   3. 监控系统服务并邮件报警实例
      1. 应用场景说明
      2. 脚本实现思路与流程分析
      3. 脚本编程测试
      4. 139邮箱报警举例
   4. 监控系统磁盘状态并邮件报警
      1. 应用场景说明
      2. 脚本实现思路与流程分析
      3. 脚本编程测试

【考核要求】

1. **掌握mail命令的用法**
   1. 将文件当做电子邮件的内容送出

语法：mail -s “主题”用户名@地址< 文件

例如：

mail -s “program” user < file.c

将file.c当做mail的内容，送至user，主题为program。

* 1. 传送电子邮件给本系统用户

语法：mail 用户名

* 1. 传送电子邮件至外地用户

语法： mail 用户名@接受地址

例如：

mail [test@hotmail.com](mailto:test@hotmail.com)

Subject : mail test

:

键入信文内容

: :

按下Ctrl+D 键或. 键结束正文。

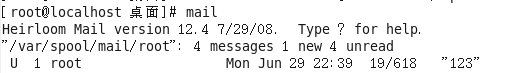
连按两次Ctrl+C键则中断工作，不送此信件。

* 1. 通过管道符号传送电子邮件

echo “hello”|mail -s “test” tom@localhost

1. **mail命令的使用**

在linux系统中输入mail，就进入了收件箱



unread 标记为未读

h|headers 显示当前的邮件列表

l|list 显示当前支持的命令列表

?|help 显示多个查看邮件列表的命令参数用法

d 删除当前邮件，指针并下移。 d 1-100 删除第1到100封邮件

z  显示刚进行收件箱时的后面二十封邮件列表

x  退出mail命令平台，并不保存之前的操作，比如删除邮件

q  退出mail

1. **掌握mutt命令**
   1. 常用选项

-s: 邮件标题，用""括上。

-a: 邮件附件。

如：< /tmp/x.txt：/tmp目录下x.txt文件里的内容作为邮件内容发送。

1. **掌握邮件报警实例**：备份系统文件
2. 应用场景：

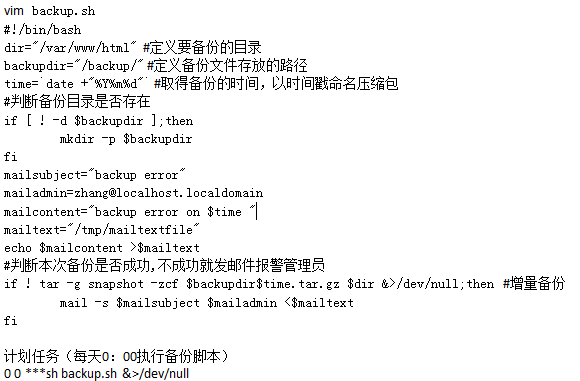
公司的网站数据每天都会有新内容，为了防止公司网站出现问题，需要将公司的网站数据进行备份，在进行全量备份的基础上，每天进行增量备份即可，如备份没有完成则发邮件进行报警，及时处理故障，确保公司数据安全，注意备份文件名附上日期。

1. 实现思路与流程分析：

设置变量以指定要备份的网站数据目录与备份文件放置的目录🡪判断备份目录是否存在🡪设置邮件相关内容，如主题、邮件账号🡪执行增量备份，如失败发邮件

设置crond服务，定期周期执行脚本

1. 脚本编程与调试



1. **掌握邮件报警实例**：监控系统服务的运行状态
2. 应用场景：

公司运行着httpd、vsftpd、sshd等服务，为了确保公司服务运行正常，要时常监控这些服务的运行状态，出现问题就发邮件给指定用户进行报警，以及时处理故障

1. 实现思路与流程分析：

设置变量以存放要监控的服务名称🡪逐个监控服务正常刚显示服务在正运行🡪否则给指定用户发邮件报警

注：可使用ps aux|grep httpd|grep -v grep 查看httpd进程或是pgrep httpd进行各个服务运行状态；也可以使用service httpd status &>/dev/null进行服务运行状态的监控；邮件主题为warning，要求每隔10分钟运行一次脚本。

1. 编程实现测试

crontab –e

\*/10 \* \* \* \* sh /root/bin/service-status.sh

#!/bin/bash

export PATH=/usr/lib64/qt-3.3/bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin

sers="httpd vsftpd sshd"

for ser in $sers

do

service $ser status &>/dev/nul

if [ $? -eq 0 ]

then

echo "$ser is runnig"

else

for user in root jack tom

do

echo "$ser is down" |mail -s "warning" $user@localhost.localdomain

done

fi

done

1. **掌握使用139邮箱进行报警的方法**

在日常的服务监控中，可以使用139邮箱来发送服务器的报警，只要注册好139邮箱即可使用

mail -s “warning” [11111111111@139.com](mailto:11111111111@139.com) <warning-file

echo “warning”| mail –s “warning” 11111111111@139.com

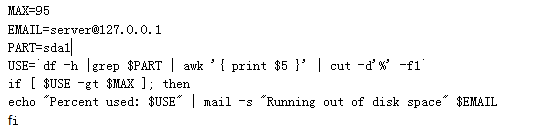
1. **掌握邮件报警实例：**监控磁盘
2. 应用场景：

公司服务器运行期间如果磁盘过满，对公司业务将会产生极大影响，可能会使公司业务停止，所以要时刻监控磁盘等系统运行状态数据，如超过一定范围则给管理员发邮件发出警告

1. 实现思路与流程分析：

设置变量以存放范围数据和邮件用户地址🡪提取磁盘使用百分比🡪如果超过指定范围，则发出警告邮件

1. 编程实现：



1. 海云捷讯公司ssh批量部署服务案例
   1. 公司批量部署
      1. 批量部署应用场景
      2. 实现思路与流程分析
   2. 使用脚本进行源码包安装apache
      1. 实现步骤分析
      2. 安装脚本编程
   3. ssh无密码登录
      1. ssh无密码登录原理
      2. ssh无密码登录服务实现
      3. 使用ssh无密码登录实现批量部署
      4. 综合脚本调试
   4. apache启动脚本
      1. 实现思路与流程分析
      2. 脚本编程测试
      3. 启动脚本的优化

【考核要求】

1. **了解批量部署的运用场景**

在企业服务器系统中，往往要在几台服务器甚至成十台服务器部署相同的服务，以形成集群，共同协调完成公司业务处理，提高业务响应能力与速度，并保证数据安全。如果手动在所有服务器上安装相同服务，那工作量就会加大及效率势必会极大低下，因此要用到批量自动部署。

1. **掌握批量部署的实现思路**

在公司中要实现全自动批量部署，目前使用最多的就是通过ssh服务的无密码登录，因此首先应该在所有的服务器上配置ssh的无密码登录服务；之后将手动安装服务的命令编写成脚本并将脚本与软件包通过scp服务远程传输到各个服务器，并在运程服务器上执行脚本即可完成指定部署

1. **掌握在单台服务器上使用脚本源码包安培apache服务**
2. 实现步骤分析：

配置yum源🡪安装编译工具🡪源码安装apache🡪

1. 编程实现：

#!/bin/bash

#配置yum源

rm -f /etc/yum.repos.d/\*

yumfile="/etc/yum.repos.d/wg.repo"

echo "[local]">$yumfile

echo "name=wg">>$yumfile

echo "baseurl=file:///media">>$yumfile

echo "gpgcheck=0">>$yumfile

echo "enabled=1">>$yumfile

mount /dev/sr0 /media

#安装编译工具

yum -y install gcc make

if rpm -q gcc &&rpm -q make ;then

echo "gcc make安装正确"

else

echo "gcc make安装有误"

exit

fi

#源码包安装apache

service httpd stop

cd /root/

tar -xvf httpd-2.2.26.tar.gz

cd /root/httpd-2.2.26

./configure --prefix=/usr/local/apache &&make &&make install

echo "shell">/usr/local/apache/htdocs/index.html

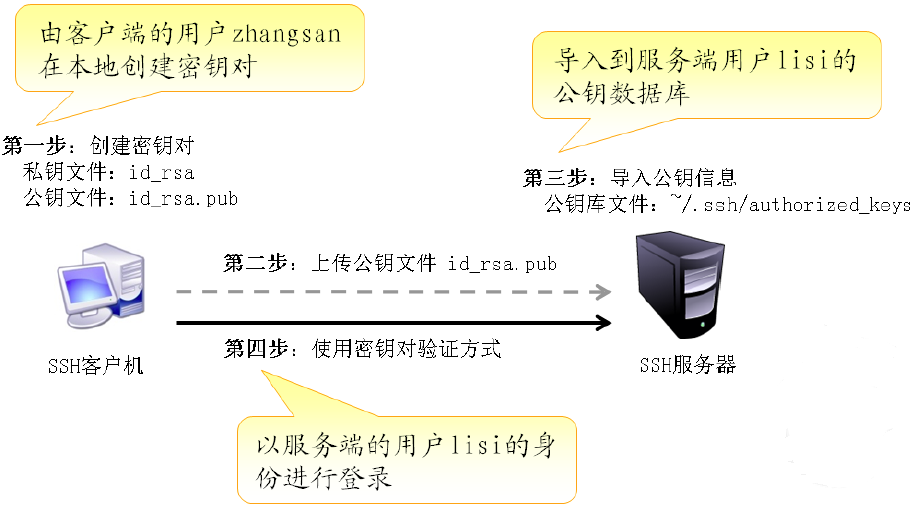
/usr/local/apache/bin/apachectl start

iptables -A INPUT -p tcp --dport 80 -j ACCEPT

#service iptables stop

netstat -tnlp|grep 80

1. **掌握通过ssh实现批量分发部署**
2. ssh无密码登录原理



1. ssh无密码登录实现步骤

**Linux客户端主机上生成密钥文件**

ssh-keygen -t rsa

选择密钥文件保存位置：【回车】默认保存在/root/.ssh/

为私钥文件设置密码：【回车】 不设置密码

再次输入密码：【回车】不设置密码

生成文件：id\_rsa 私钥 id\_rsa.pub 公钥

**将公钥上传至ssh服务器**

scp /root/.ssh/id\_rsa.pub root@172.18.209.220:/root/.ssh/authorized\_keys

**测试**

ssh root@172.18.49.100 无需验证密码直接登录

1. **ssh批量分发部署：**
2. 实现思路：

将要批量部署的服务器的ip地址保存在文件中🡪读取ip文件逐个将apache源码安装包与安装脚本传输到服务器上🡪远程执行安装脚本🡪为了方便可使用at工具进行自动安装

1. 编程实现：

vim iplist 所有需要部署服务的ip地址列表

-------------------

172.18.6.101

172.18.6.102

------------------

vim dis.sh

-------------------------

#/bin/bash

for ip in `cat iplist`

do

scp /root/httpd-2.2.26.tar.gz $ip:/root/

scp /root/httpdinstall.sh $ip:/root/

ssh $ip sh /root/httpdinstall.sh

####远程执行脚本----改写成计划任务at

#

#expect "]#" #命令提示符#或是$

#send "at now+3minutes<<EOF

#echo 'expect shixian jihuarenwu' ##注意双引号内再要有双引号用单引号代替

#EOF

#\r"

#

done

-------------------------------------

执行sh dis.sh进行测试并调试，成功后就可完成批量部署服务器

1. **掌握服务的启动脚本编程**
2. 实现思路与流程分析：

设置变量以存放apache启动命令工具🡪定义start、stop、restart、status等各个函数🡪在case语句中调用各个函数

1. 编程实现与调试

#!/bin/bash

#service apache start|stop|restart|status

lock="/usr/local/apache/httpd.lock"

start ()

{

/usr/local/apache/bin/apachectl start

echo "正在启动服务"

touch $lock

}

stop ()

{

/usr/local/apache/bin/apachectl stop

echo "正在关闭服务"

rm -f $lock

}

restart ()

{

stop

start

}

status()

{

if [ -e $lock ];then

echo "服务已启动"

else

echo "服务已停止"

fi

}

case "$1" in

"start")

start;;

"stop")

stop;;

"restart")

restart;;

"status")

status;;

\*)

echo "usage:service apache start|stop|restart|status";;

esac

1. 脚本优化

将所编写的脚本按照标准的系统启动脚本进行优化，以实现更加丰富的脚本功能

#!/bin/bash

# apache - this script starts and stops the apache daemon

# chkconfig: - 85 15

# description: Apache is an HTTP(S) server

# processname: apache

# config: /usr/local/apache/conf/apache.conf

# Source function library.

. /etc/rc.d/init.d/functions

# Source networking configuration.

. /etc/sysconfig/network

# Check that networking is up.

[ "$NETWORKING" = "no" ] && exit 0

apache="/usr/local/apache/bin/apachectl"

prog=$(basename $apache)

APACHE\_CONF\_FILE="/usr/local/apache/conf/httpd.conf"

lockfile=/var/lock/subsys/apache

start() {

[ -x $apache ] || exit 5

[ -f $APACHE\_CONF\_FILE ] || exit 6

echo -n $"Starting $prog: "

echo\_success

echo

$apache start

retval=$?

[ $retval -eq 0 ] && touch $lockfile

return $retval

}

stop() {

echo -n $"Stopping $prog: "

echo\_success

echo

$apache stop

retval=$?

[ $retval -eq 0 ] && rm -f $lockfile

return $retval

}

restart() {

stop

start

}

status()

{

if [ -e $lockfile ];then

echo "服务已启动"

else

echo "服务已停止"

fi

}

case "$1" in

start)

$1

;;

stop)

$1

;;

restart)

$1

;;

status)

$1;;

\*)

echo $"Usage: $0 {start|stop|status|restart}"

exit 2

esac

1. 瑞友科技公司apache虚拟主机的自动创建案例
   1. Apache虚拟主机的创建
      1. 手动创建虚拟主机
      2. 脚本创建虚拟主机
   2. 优化创建虚拟主机脚本
      1. 优化思路与流程分析
      2. 变量设置与优化编程
      3. 脚本调试
      4. 定义判断输入函数
      5. 监控httpd服务并使用邮件报警
      6. 综合脚本调试

【考核要求】

1. **掌握apache虚拟主机的创建过程**

公司要创建的虚拟主机，为了提高创建效率，需要编写创建虚拟主机的脚本

1. 查看apache的配置文件/etc/httpd/conf/httpd.conf，开启NameVirtualHost \*:80

sed -i 's/#\(NameVirtualHost.\*80$\)/\1/' /etc/httpd/conf/httpd.conf

1. 添加虚拟主机

cat >>/etc/httpd/conf/httpd.conf <<EOF //写虚拟主机配置文件信息追加到配置文件中

<VirtualHost 172.18.8.90:80>

DocumentRoot /var/www/wg.bw.com

ServerName wg.bw.com

</VirtualHost>

EOF

1. 重启httpd服务

/etc/init.d/httpd restart

1. **掌握脚本实现：将上述命令编写到一个脚本中即可**

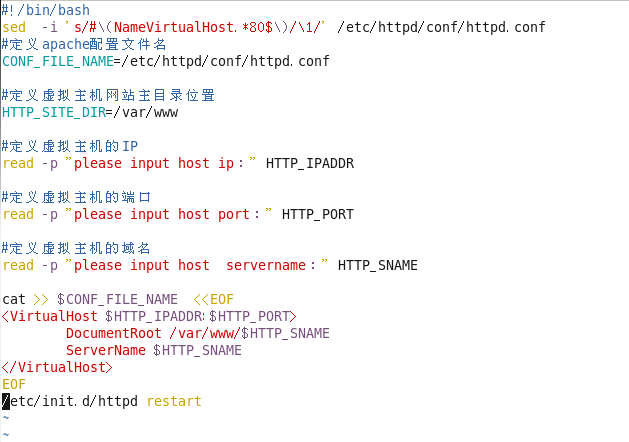


1. **脚本优化**
2. 优化需求：
   1. 创建过程交互式进行，要求用户输入创建的ip、端口及虚拟主机域名
   2. 虚拟主机的主目录在/var/www/下，目录名称为虚拟主机域名
   3. 创建过程中如果用户回车无效，要求再次输入
   4. apache若不正常则报警
3. 实现思路与流程分析：

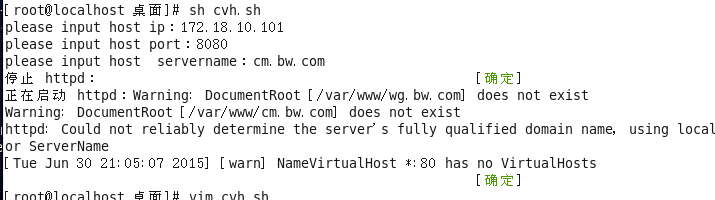
通过read –p命令进行交互式🡪定义一个函数fun\_input实现交互式输入值得判断🡪通过cat命令将虚拟主机配置内容写入文档

🡪apache服务若不正常则发送报警邮件，正文为“XX月XX日 XX点XX分 $ip apache shutdown”🡪将配置文件、虚拟主机主目录、虚拟主机的域名、ip通过变量引用

1. 编程实现：
   1. 优化一：



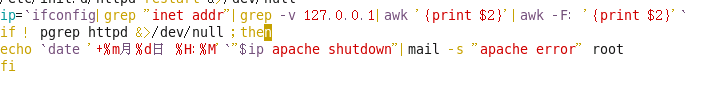
执行测试



* 1. 优化二：定义判断输入值得函数



* 1. 优化三：邮件报警



1. 网融众信公司全自动安装tomcat案例
   1. tomcat介绍
      1. tomcat企业应用场景
      2. tomcat企业架构
      3. 手动安装tomcat
   2. 自动安装tomcat
      1. 实现思路与流程分析
      2. 分步编程实现
      3. 脚本调试
      4. 使用ssh实现批量部署并邮件报警
      5. 综合脚本调试

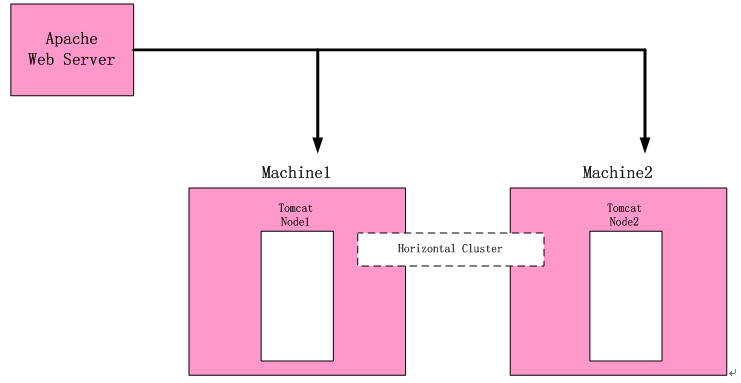
【考核要求】

1. **了解tomcat企业应用场景**

Tomcat是Apache 软件基金会（Apache Software Foundation）的Jakarta 项目中的一个核心项目，由[Apache](http://baike.baidu.com/subview/28283/5418752.htm" \t "_blank)、Sun 和其他一些公司及个人共同开发而成。由于有了Sun 的参与和支持，最新的Servlet 和JSP 规范总是能在Tomcat 中得到体现，Tomcat 5支持最新的Servlet 2.4 和JSP 2.0 规范, 因为Tomcat 技术先进、性能稳定，而且免费，因而深受Java 爱好者的喜爱并得到了部分软件开发商的认可，成为目前比较流行的Web 应用服务器。

Tomcat 服务器是一个免费的开放源代码的Web 应用服务器，属于轻量级应用[服务器](http://baike.baidu.com/view/899.htm" \t "_blank)，在中小型系统和并发访问用户不是很多的场合下被普遍使用，是开发和调试JSP 程序的首选。对于一个初学者来说，可以这样认为，当在一台机器上配置好Apache 服务器，可利用它响应[HTML](http://baike.baidu.com/view/692.htm" \t "_blank)（[标准通用标记语言](http://baike.baidu.com/view/5286041.htm" \t "_blank)下的一个应用）页面的访问请求。实际上Tomcat 部分是Apache 服务器的扩展，但它是独立运行的，所以当你运行tomcat 时，它实际上作为一个与Apache 独立的进程单独运行的。

1. **了解tomcat的企业应用架构**



1. **掌握安装tomcat步骤**
2. 下载安装对应的jdk,并配置Java环境。可到官网下载，地址：

<http://www.oracle.com/technetwork/java/javase/downloads/jdk-6u26-download-400750.html>

但是要根据所要安装的tomcat来下载安装相应的jdk环境。

将jdk解压后放到/usr/local目录下：

[root@master ~]#tar zxvf jdk-7u65-linux-x64.tar.gz

[root@master ~]#mv jdk1.7.0\_65 /usr/local

建立/usr/local/下的jdk软连接方便以后版本升级 ：

[root@master ~]# ln -s /usr/local/jdk1.7.0\_65/ /usr/local/jdk

配置环境变量：

在 /etc/profile 中加入以下内容：

JAVA\_HOME=/usr/local/jdk1.7.0\_65

JAVA\_BIN=/usr/local/jdk1.7.0\_65/bin

PATH=$PATH:$JAVA\_BIN

CLASSPATH=$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar

export JAVA\_HOME JAVA\_BIN PATH CLASSPATH

保存退出

[root@master ~]source /etc/profile

看java环境变量是否生效

[root@TESTEDW conf]# java -version

java version "1.7.0\_65"

Java(TM) SE Runtime Environment (build 1.7.0\_65-b17)

Java HotSpot(TM) 64-Bit Server VM (build 24.65-b04, mixed mode)

测试成功

1. 下载安装tomcat（http://tomcat.apache.org/）

[root@zhanglin ~]# tar -zxvf apache-tomcat-8.0.15.tar.gz

[root@zhanglin ~]# mv apache-tomcat-8.0.15/ /usr/local/tomcat8.0

[root@zhanglin ~]cd /usr/local/

[root@zhanglin local]# ln -s /usr/local/tomcat8.0/ /usr/local/tomcat

[root@zhanglin local]# cd tomcat/bin/

[root@zhanglin bin]#ls

[root@zhanglin bin]#vim catalina.sh

添加以下内容：

CATALINA\_HOME=/usr/local/tomcat8.0/

[root@zhanglin local]#chmod +x \*.sh

1. 启动tomcat服务器

[root@TESTEDW bin]# ./startup.sh

Using CATALINA\_BASE: /usr/local/tomcat8.0

Using CATALINA\_HOME: /usr/local/tomcat8.0

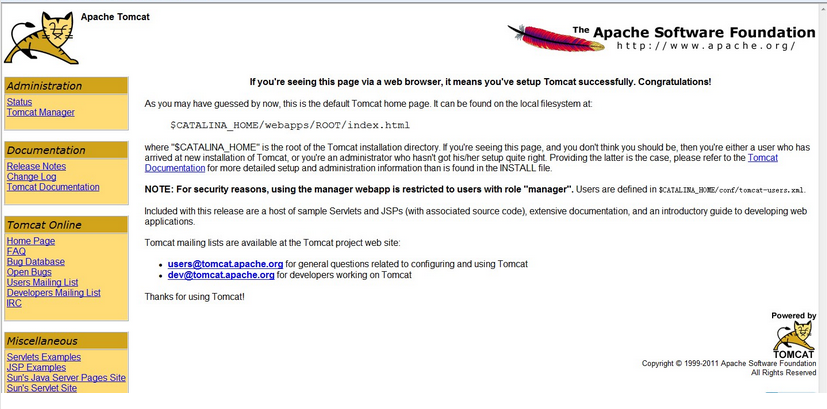
Using CATALINA\_TMPDIR: /usr/local/tomcat8.0/temp

Using JRE\_HOME: /usr/local/jdk1.7.0\_65

Using CLASSPATH: /usr/local/tomcat8.0/bin/bootstrap.jar:/usr/local/tomcat8.0/bin/tomcat-juli.jar

Tomcat started.

1. 在浏览器中输入<http://localhost:8080/>或者输入对应的ip地址也可以，测试出现tomcat页面则测试成功：



注：需要说明的是tomcat的默认测试页面是放在webapps下面，这个其实是在server.xml文件中配置的，如下所示：



[root@SOR\_SYS tomcat8.0]# ls

LICENSE NOTICE RELEASE-NOTES RUNNING.txt bin conf lib logs temp webapps work

1)webapps文件夹主要用于web应用程序部署，比如你可以把你的应用程序包，如war文件拷到该目录下，容器会自动部署。

2)conf文件夹下主要是放置tomcat的服务器的相关配置文件

3）修改tomcat的监听端口，有时可能需要修改访问端口

[root@master ~]# cd /usr/local/tomcat/conf

[root@master ~]# vim server.xml

<connector port="8080"protocol="HTTP/1.1"connectionTimeout="20000"   
redirectPort="8443"/>

将端口修改为没有被占用的端口即可，重启tomcat。访问<http://192.168.55.229>:新端口

1. **掌握使用脚本全自动安装tomcat的方法**

将手动安装的命令编辑到脚本中，并进行修改调试即可。

1. 编程思路与流程分析：

解包并安装jdk🡪设置环境变量🡪检测java环境是否正常🡪解包并安装tomcat🡪启动tomcat进行检测

1. 编程实现与调试(tomcat.sh)

#!/bin/bash

tar zxvf jdk-7u65-linux-x64.tar.gz

mv jdk1.7.0\_65 /usr/local

#建立/usr/local/下的jdk软连接方便以后版本升级 ：

ln -s /usr/local/jdk1.7.0\_65/ /usr/local/jdk

#配置环境变量：在 /etc/profile 中加入JAVA的环境变量：

echo 'JAVA\_HOME=/usr/local/jdk1.7.0\_65' >>/etc/profile

echo 'JAVA\_BIN=/usr/local/jdk1.7.0\_65/bin' >>/etc/profile

echo 'PATH=$PATH:$JAVA\_BIN' >>/etc/profile

echo 'CLASSPATH=$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar' >>/etc/profile

echo 'export JAVA\_HOME JAVA\_BIN PATH CLASSPATH' >>/etc/profile

source /etc/profile

#查看java环境变量是否生效

java -version & >>/dev/null

if [ $? -eq 0 ]

then

echo "jdk ok"

else

echo "warning,jdk error"

fi

#下载安装tomcat（http://tomcat.apache.org/）

tar -zxvf apache-tomcat-8.0.15.tar.gz

mv apache-tomcat-8.0.15/ /usr/local/tomcat8.0

#cd /usr/local/

ln -s /usr/local/tomcat8.0/ /usr/local/tomcat

#cd tomcat/bin/

#catalina.sh添加以下内容：

echo 'CATALINA\_HOME=/usr/local/tomcat8.0' >>/usr/local/tomcat/bin/catalina.sh

chmod +x \*.sh

#启动tomcat服务器

/usr/local/tomcat/bin/startup.sh &>>/dev/null

if [ $? -eq 0 ]

then

echo "tomcat ok"

else

echo "warning ,tomcat starte error"

fi

1. **批量部署**

根据第十七单元的ssh批量部署的方法，设置批量部署的ip地址，并将jdk与tomcat以及全自动安装脚本准备好，修改批量部署的脚本即可

vim iplist 所有需要部署服务的ip地址列表

-------------------

172.18.6.101

172.18.6.102

------------------

vim dis.sh

-------------------------

#/bin/bash

for ip in `cat iplist`

do

scp /root/apache-tomcat-8.0.15.tar.gz $ip:/root/

scp /root/ jdk-7u65-linux-x64.tar.gz $ip:/root/

scp /root/tomcat.sh $ip:/root/

ssh $ip sh /root/ tomcat.sh

done

-------------------------------------

执行sh dis.sh进行测试并调试，成功后就可完成批量部署服务器

1. 网际天娇公司iptables白名单与黑名单案例
   1. 公司安全机制介绍
      1. 防火墙概念介绍
      2. 软件防火墙
      3. 硬件防火墙
   2. iptales企业应用
      1. 应用场景说明
      2. iptables规则设置
   3. iptables白名单与黑名单
      1. 黑白名单功能说明
      2. 实现思路与流程分析
      3. 综合脚本调试

【考核要求】

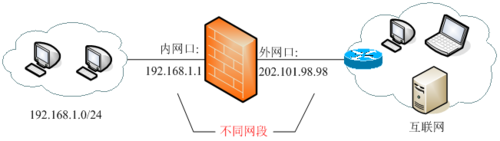
1. **了解防火墙的概念**

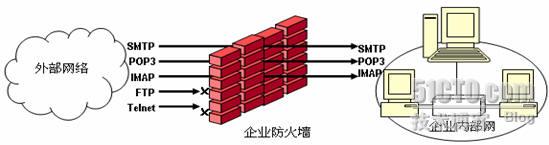
公司使用linux作为网关服务器，在网关服务器上使用iptables进行安全的控制，因为更好的保护公司业务的安全，有必要学习iptables脚本的编程，以方便高效设置iptables规则。

防火墙（Firewall），也称防护墙，是由Check Point创立者Gil Shwed于1993年发明并引入国际互联网（US5606668（A）1993-12-15）。它是一种位于内部网络与外部网络之间的网络安全系统。一项信息安全的防护系统，依照特定的规则，允许或是限制传输的数据通过。

所谓防火墙指的是一个由软件和硬件设备组合而成、在内部网和外部网之间、专用网与公共网之间的界面上构造的保护屏障.是一种获取安全性方法的形象说法，它是一种计算机硬件和软件的结合，使Internet与Intranet之间建立起一个安全网关（Security Gateway），从而保护内部网免受非法用户的侵入，防火墙主要由服务访问规则、验证工具、包过滤和应用网关4个部分组成，防火墙就是一个位于计算机和它所连接的网络之间的软件或硬件。该计算机流入流出的所有网络通信和数据包均要经过此防火墙。

在网络中，所谓“防火墙”，是指一种将内部网和公众访问网（如Internet）分开的方法，它实际上是一种隔离技术。防火墙是在两个网络通讯时执行的一种访问控制尺度，它能允许你“同意”的人和数据进入你的网络，同时将你“不同意”的人和数据拒之门外，最大限度地阻止网络中的黑客来访问你的网络。换句话说，如果不通过防火墙，公司内部的人就无法访问Internet，Internet上的人也无法和公司内部的人进行通信。





1. **了解软件防火墙**

软件防火墙单独使用软件系统来完成防火墙功能，将软件部署在系统主机上，其安全性较硬件防火墙差，同时占用系统资源，在一定程度上影响系统性能。其一般用于单机系统或是极少数的个人计算机，很少用于计算机网络中。



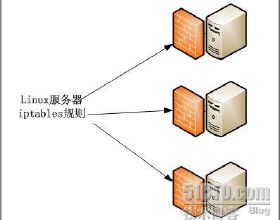
1. **了解硬件防火墙**

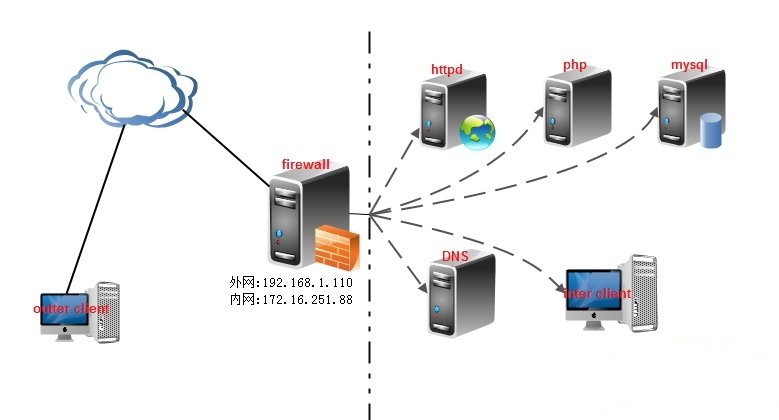
把软件防火墙嵌入在硬件中，一般的软件安全厂商所提供的硬件防火墙便是在硬件服务器厂商定制硬件，然后再把linux系统与自己的软件系统嵌入，也就是说[硬件](http://baike.baidu.com/view/25278.htm" \t "_blank)[防火墙](http://baike.baidu.com/view/3067.htm" \t "_blank)是指把防火墙程序做到[芯片](http://baike.baidu.com/view/26651.htm" \t "_blank)里面，由硬件执行这些功能，能减少[CPU](http://baike.baidu.com/view/2089.htm" \t "_blank)的负担，使[路由](http://baike.baidu.com/view/18655.htm" \t "_blank)更稳定。

1. **了解iptables企业应用**

iptables是一个软件防火墙,一般应用在网关的位置上，也可以应用在单机上，通过设置相应的规则，对流过网关或者主机的数据包进行过滤，从而确保服务器的网络安全。





1. **掌握iptables规则设置**
2. 四个表:

          filter：一般的过滤功能

          nat:用于nat功能（端口映射，地址映射等）

          mangle:用于对特定数据包的修改

          raw:有限级最高，设置raw时一般是为了不再让iptables做数据包的链接跟踪处理，提高性能

1. 五个规则链：

PREROUTING (路由前)

INPUT (数据包流入口)

FORWARD (转发)

OUTPUT(数据包出口)

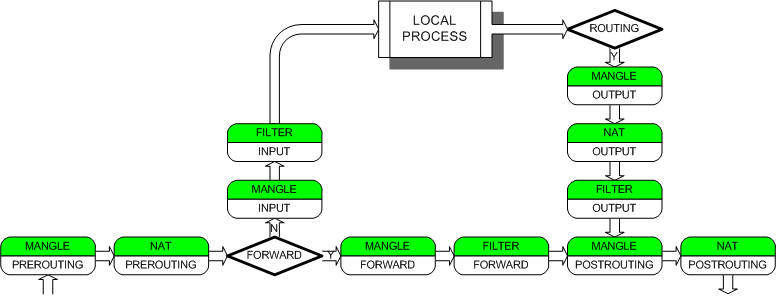
POSTROUTING（路由后）

任何一个数据包，只要经过本机，必将经过这五个链中的其中一个链。

对于filter来讲一般只能做在3个链上：INPUT，FORWARD，OUTPUT

对于nat来讲一般也只能做在3个链上：PREROUTING，OUTPUT，POSTROUTING

而mangle则是5个链都可以做：PREROUTING，INPUT，FORWARD，OUTPUT，POSTROUTING



1. iptables的基本语法格式：  
    iptables [-t 表名] 命令选项 ［链名］ ［条件匹配］ ［-j 目标动作或跳转］  
   说明：
2. 表名、链名用于指定iptables命令所操作的表和链
3. 命令选项用于指定管理iptables规则的方式（比如：插入、增加、删除、查看等
4. 条件匹配用于指定对符合什么样条件的数据包进行处理
5. 目标动作或跳转用于指定数据包的处理方式（比如允许通过、拒绝、丢弃、跳转（Jump）给其它链处理。
6. Iptables命令的管理控制选项：

-A 在指定链的末尾添加（append）一条新的规则  
-D删除（delete）指定链中的某一条规则，可以按规则序号和内容删除  
-I在指定链中插入（insert）一条新的规则，默认在第一行添加  
-R修改、替换（replace）指定链中的某一条规则，可以按规则序号和内容替换  
-L列出（list）指定链中所有的规则进行查看  
-F清空（flush）  
-N新建（new-chain）一条用户自己定义的规则链  
-X删除指定表中用户自定义的规则链（delete-chain）  
-P设置指定链的默认策略（policy）  
-n使用数字形式（numeric）显示输出结果  
-v查看规则表详细信息（verbose）的信息  
-V查看版本(version)  
-h获取帮助（help）

1. 防火墙处理数据包的四种方式：

ACCEPT 允许数据包通过  
DROP 直接丢弃数据包，不给任何回应信息  
REJECT 拒绝数据包通过，必要时会给数据发送端一个响应的信息。  
LOG在/var/log/messages文件中记录日志信息，然后将数据包传递给下一条规则

1. iptables防火墙规则的保存与恢复

iptables-save把规则保存到文件中，再由目录rc.d下的脚本（/etc/rc.d/init.d/iptables）自动装载，一般用：

iptables-save > /etc/sysconfig/iptables

也可以用：

service iptables save

它能把规则自动保存在/etc/sysconfig/iptables中。

当计算机启动时，rc.d下的脚本将用命令iptables-restore调用这个文件，从而就自动恢复了规则。

1. iptables应用举例

删除INPUT链的第一条规则  
iptables -D INPUT 1

iptables防火墙常用的策略：  
拒绝进入防火墙的所有ICMP协议数据包  
iptables -I INPUT -p icmp -j REJECT

允许防火墙转发除ICMP协议以外的所有数据包  
iptables -A FORWARD -p ! icmp -j ACCEPT  
说明：使用“！”可以将条件取反。

拒绝转发来自192.168.1.10主机的数据，允许转发来自192.168.0.0/24网段的数据  
iptables -A FORWARD -s 192.168.1.11 -j REJECT  
iptables -A FORWARD -s 192.168.0.0/24 -j ACCEPT  
说明：注意要把拒绝的放在前面不然就不起作用了啊。

丢弃从外网接口（eth1）进入防火墙本机的源地址为私网地址的数据包  
iptables -A INPUT -i eth1 -s 192.168.0.0/16 -j DROP  
iptables -A INPUT -i eth1 -s 172.16.0.0/12 -j DROP  
iptables -A INPUT -i eth1 -s 10.0.0.0/8 -j DROP

封堵网段（192.168.1.0/24），两小时后解封。  
[root@server ~]# iptables -I INPUT -s 10.20.30.0/24 -j DROP  
[root@server ~]# iptables -I FORWARD -s 10.20.30.0/24 -j DROP  
[root@server ~]# at now +2 hours  
at> iptables -D INPUT 1  
at> iptables -D FORWARD 1  
说明：这个策略咱们借助crond计划任务来完成，就再好不过了。  
[1]+  Stopped     at now +2 hours

只允许管理员从202.13.0.0/16网段使用SSH远程登录防火墙主机。  
iptables -A INPUT -p tcp --dport 22 -s 202.13.0.0/16 -j ACCEPT  
iptables -A INPUT -p tcp --dport 22 -j DROP  
说明：这个用法比较适合对设备进行远程管理时使用，比如位于分公司中的SQL服务器需要被总公司的管理员管理时。

允许本机开放从TCP端口20-1024提供的应用服务。  
iptables -A INPUT -p tcp --dport 20:1024 -j ACCEPT  
iptables -A OUTPUT -p tcp --sport 20:1024 -j ACCEPT

允许转发来自192.168.0.0/24局域网段的DNS解析请求数据包。  
iptables -A FORWARD -s 192.168.0.0/24 -p udp --dport 53 -j ACCEPT  
iptables -A FORWARD -d 192.168.0.0/24 -p udp --sport 53 -j ACCEPT

禁止其他主机ping防火墙主机，但是允许从防火墙上ping其他主机  
iptables -I INPUT -p icmp --icmp-type Echo-Request -j DROP  
iptables -I INPUT -p icmp --icmp-type Echo-Reply -j ACCEPT  
iptables -I INPUT -p icmp --icmp-type destination-Unreachable -j ACCEPT

禁止转发来自MAC地址为00：0C：29：27：55：3F的和主机的数据包  
iptables -A FORWARD -m mac --mac-source 00:0c:29:27:55:3F -j DROP  
说明：iptables中使用“-m 模块关键字”的形式调用显示匹配。咱们这里用“-m mac –mac-source”来表示数据包的源MAC地址。

允许防火墙本机对外开放TCP端口20、21、25、110以及被动模式FTP端口1250-1280  
iptables -A INPUT -p tcp -m multiport --dport 20,21,25,110,1250:1280 -j ACCEPT  
说明：这里用“-m multiport –dport”来指定目的端口及范围

禁止转发源IP地址为192.168.1.20-192.168.1.99的TCP数据包。  
iptables -A FORWARD -p tcp -m iprange --src-range 192.168.1.20-192.168.1.99 -j DROP  
说明：此处用“-m –iprange –src-range”指定IP范围。

禁止转发与正常TCP连接无关的非—syn请求数据包。  
iptables -A FORWARD -m state --state NEW -p tcp ! --syn -j DROP  
说明：“-m state”表示数据包的连接状态，“NEW”表示与任何连接无关的，新的嘛！

拒绝访问防火墙的新数据包，但允许响应连接或与已有连接相关的数据包  
iptables -A INPUT -p tcp -m state --state NEW -j DROP  
iptables -A INPUT -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT  
说明：“ESTABLISHED”表示已经响应请求或者已经建立连接的数据包，“RELATED”表示与已建立的连接有相关性的，比如FTP数据连接等。

只开放本机的web服务（80）、FTP(20、21、20450-20480)，放行外部主机发住服务器其它端口的应答数据包，将其他入站数据包均予以丢弃处理。  
iptables -I INPUT -p tcp -m multiport --dport 20,21,80 -j ACCEPT  
iptables -I INPUT -p tcp --dport 20450:20480 -j ACCEPT  
iptables -I INPUT -p tcp -m state --state ESTABLISHED -j ACCEPT  
iptables -P INPUT DROP

1. **了解iptbales黑名单与白名单的作用**

iptables黑名单就是将某些不安全的主机的ip保存在一个文件中，通过设置iptables规则将相应的主机拒绝在门外，以保护公司的服务器的安全。

白名单保存的是安全的服务器的ip地址，设置iptables规则时将这些主机设置为放行即可。

1. **掌握实现思路与流程分析**
2. 应用场景：

对于数量比较大的黑名单名或白名单而言，要手动进行iptables规则的设置，重复工作量势必很大，效率很大，因此要编写iptables规则脚本来进行设置，提高效率。

1. 编程思路与流程分析

将ip地址写入文件，一行一个🡪设置iptables相关变量🡪针对公司业务设置iptables规则🡪针对黑名单或白名单设置规则

1. 编程实现与调试

vim /root/balcklist.txt

192.168.1.1

192.168.1.2

…….

vim /root/whitelist.txt

192.168.1.3

192.168.1.4

…….

#!/bin/bash

IF="eth0"

OF=”eth1”

modprobe ipt\_MASQUERADE  
modprobe ip\_conntrack\_ftp  
modprobe ip\_nat\_ftp  
#清除规则  
/sbin/iptables -F   
/sbin/iptables -X   
/sbin/iptables -Z  
# 预定义策略，根据公司业务进行设置  
/sbin/iptables -A INPUT -s 127.0.0.1 -j ACCEPT # 允许回环接口可以被访问  
/sbin/iptables -P INPUT DROP # 默认是拒绝访问  
/sbin/iptables -P OUTPUT ACCEPT # 允许本机访问其他机器，无限制   
/sbin/iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT   
/sbin/iptables -A INPUT -p icmp -j ACCEPT # 允许ping  
#允许的本机服务  
/sbin/iptables -A INPUT -p TCP -i $IF --dport 22 -j ACCEPT # SSH服务   
/sbin/iptables -A INPUT -p TCP -i $IF --dport 3306 -j ACCEPT # mysql数据库  
/sbin/iptables -A INPUT -p TCP -i $IF --dport 80 -j ACCEPT # web服务  
# 黑名单  
for IP in $(cat /root/blacklist.txt)

do

/sbin/iptables -A INPUT -s $IP -j DROP  
done

# 信任的网络和IP，就是白名单设置  
for IP in $(cat /root/whitlist.txt)

do

/sbin/iptables -A INPUT -s $IP -j ACCEPT  
done

1. 生产iptabels脚本举例

#!/bin/sh  
#  
modprobe ipt\_MASQUERADE  
modprobe ip\_conntrack\_ftp  
modprobe ip\_nat\_ftp  
iptables -F  
iptables -t nat -F  
iptables -X  
iptables -t nat -X  
###########################**INPUT键**###################################  
  
  
iptables -P INPUT DROP  
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT  
iptables -A INPUT -p tcp -m multiport --dports 110,80,25 -j ACCEPT  
iptables -A INPUT -p tcp -s 192.168.0.0/24 --dport 139 -j ACCEPT  
#允许内网samba,smtp,pop3,连接  
iptables -A INPUT -i eth1 -p udp -m multiport --dports 53 -j ACCEPT  
#允许dns连接  
iptables -A INPUT -p tcp --dport 1723 -j ACCEPT  
iptables -A INPUT -p gre -j ACCEPT  
#允许外网vpn连接  
iptables -A INPUT -s 192.186.0.0/24 -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT  
iptables -A INPUT -i ppp0 -p tcp --syn -m connlimit --connlimit-above 15 -j DROP  
#为了防止DOS太多连接进来,那么可以允许最多15个初始连接,超过的丢弃  
iptables -A INPUT -s 192.186.0.0/24 -p tcp --syn -m connlimit --connlimit-above 15 -j DROP  
#为了防止DOS太多连接进来,那么可以允许最多15个初始连接,超过的丢弃  
iptables -A INPUT -p icmp -m limit --limit 3/s -j LOG --log-level INFO --log-prefix "ICMP packet IN: "  
iptables -A INPUT -p icmp -j DROP  
#禁止icmp通信-ping 不通  
iptables -t nat -A POSTROUTING -o ppp0 -s 192.168.0.0/24 -j MASQUERADE  
#内网转发  
iptables -N syn-flood  
iptables -A INPUT -p tcp --syn -j syn-flood  
iptables -I syn-flood -p tcp -m limit --limit 3/s --limit-burst 6 -j RETURN  
iptables -A syn-flood -j REJECT  
#防止SYN攻击 轻量  
#######################**FORWARD链**###########################  
iptables -P FORWARD DROP  
iptables -A FORWARD -p tcp -s 192.168.0.0/24 -m multiport --dports 80,110,21,25,1723 -j ACCEPT  
iptables -A FORWARD -p udp -s 192.168.0.0/24 --dport 53 -j ACCEPT  
iptables -A FORWARD -p gre -s 192.168.0.0/24 -j ACCEPT  
iptables -A FORWARD -p icmp -s 192.168.0.0/24 -j ACCEPT  
#允许 vpn客户走vpn网络连接外网  
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT  
iptables -I FORWARD -p udp --dport 53 -m string --string "tencent" -m time --timestart 8:15 --timestop 12:30 --days Mon,Tue,Wed,Thu,Fri,Sat  -j DROP  
#星期一到星期六的8:00-12:30禁止qq通信  
iptables -I FORWARD -p udp --dport 53 -m string --string "TENCENT" -m time --timestart 8:15 --timestop 12:30 --days Mon,Tue,Wed,Thu,Fri,Sat  -j DROP  
#星期一到星期六的8:00-12:30禁止qq通信  
iptables -I FORWARD -p udp --dport 53 -m string --string "tencent" -m time --timestart 13:30 --timestop 20:30 --days Mon,Tue,Wed,Thu,Fri,Sat  -j DROP  
iptables -I FORWARD -p udp --dport 53 -m string --string "TENCENT" -m time --timestart 13:30 --timestop 20:30 --days Mon,Tue,Wed,Thu,Fri,Sat  -j DROP  
#星期一到星期六的13:30-20:30禁止QQ通信  
iptables -I FORWARD -s 192.168.0.0/24 -m string --string "qq.com" -m time --timestart 8:15 --timestop 12:30 --days Mon,Tue,Wed,Thu,Fri,Sat  -j DROP  
#星期一到星期六的8:00-12:30禁止qq网页  
iptables -I FORWARD -s 192.168.0.0/24 -m string --string "qq.com" -m time --timestart 13:00 --timestop 20:30 --days Mon,Tue,Wed,Thu,Fri,Sat  -j DROP  
#星期一到星期六的13:30-20:30禁止QQ网页  
iptables -I FORWARD -s 192.168.0.0/24 -m string --string "ay2000.net" -j DROP  
iptables -I FORWARD -d 192.168.0.0/24 -m string --string "宽频影院" -j DROP  
iptables -I FORWARD -s 192.168.0.0/24 -m string --string "色情" -j DROP  
iptables -I FORWARD -p tcp --sport 80 -m string --string "广告" -j DROP  
#禁止ay2000.net，宽频影院，色情，广告网页连接 ！但中文 不是很理想  
iptables -A FORWARD -m ipp2p --edk --kazaa --bit -j DROP  
iptables -A FORWARD -p tcp -m ipp2p --ares -j DROP  
iptables -A FORWARD -p udp -m ipp2p --kazaa -j DROP  
#禁止BT连接  
iptables -A FORWARD -p tcp --syn --dport 80 -m connlimit --connlimit-above 15 --connlimit-mask 24  
#######################################################################  
sysctl -w net.ipv4.ip\_forward=1 &>/dev/null  
#打开转发  
#######################################################################  
sysctl -w net.ipv4.tcp\_syncookies=1 &>/dev/null  
#打开 syncookie （轻量级预防 DOS 攻击）  
sysctl -w net.ipv4.netfilter.ip\_conntrack\_tcp\_timeout\_established=3800 &>/dev/null  
#设置默认 TCP 连接痴呆时长为 3800 秒（此选项可以大大降低连接数）  
sysctl -w net.ipv4.ip\_conntrack\_max=300000 &>/dev/null  
#设置支持最大连接树为 30W（这个根据你的内存和 iptables 版本来，每个 connection 需要 300 多个字节）  
#######################################################################  
iptables -I INPUT -s 192.168.1.140 -j ACCEPT  
iptables -I FORWARD -s 192.168.1.140 -j ACCEPT  
#192.168.1.140是我的机子，全部放行！