



**每日报告总结**

|  |  |
| --- | --- |
| **学习课程：** | shell脚本编程 |
| **学 院：** | 数学与信息工程学院 |
| **班 级：** | 2020级网络2班 |
| **学 号：** | 2020531214 |
| **姓 名：** | 肖文杰 |
| **实习企业：** | 达内集团股份有限公司 |
| **学习地点：** | 线上 |
| **学习方式：** | 校内 |
| **学习时间：** | 2023年2月20日至2023年3月2日 |

|  |  |  |
| --- | --- | --- |
| **学习报告** | | |
| 学习日期（第1天） | | 2022年2月 20日 |
| 目的 | 初学Shell编程 | |
| 内容 | **语言的种类：**  语言的类型：   * 编译型语言：C 、C++、go必须把源代码编译（./configure、make、make install）成目标程序，然后执行（不依赖语言环境），看不到源代码（反编译）。例如：rsync、nfs、nginx、php、linux等。优先是执行效率高，但是比较复杂。 * 解释型语言：有一个专门的解释器读取代码，然后解释执行，因此比较依赖环境。而且能够看到源代码，不保密。例如：python、perl、php、bash、shell等，虽然性能比不上编译型语言，但是目前CPU性能过剩，我们也没有那么高的要求，因此不足为虑   **为什么要学习shell编程**   * Shell简本语言是实习呢linux/unix系统管理及自动化运维所必备的重要工具； * Linux/unix系统的底层及基础应用软件的核心大部分涉及shell脚本的内容； * 每一个合格的linux系统管理员或运维工程师，都需要熟练的编写shell脚本语言，并能够阅读系统及各类软件附带的shell脚本内容。   学好shell编程的目标：  30道企业面试shell题  http://blog.51cto.com/oldboy/1867160  **学好shell编程所）的基础知识**   * 能够熟练使用vim编辑器，熟悉ssh终端及“.vimrc”等的配置。 * 有一定的linux命令基础，至少需要掌握80个以上linux常用命令，并且能够熟练使用他们 * 要熟练掌握linux正则表达式及三剑客（grep、sed、awk） * 熟悉常见的linux网络服务部署、优化、日志分析及排错。   **如何学好shell编程**   * 学好shell编程的核心：多练-多思考-再练-在思考，坚持如此循环即可 * 新手大忌：不可拿来主义，可以模仿，但是要自己嚼烂了再吃下去，否则会闹肚子的 * 格言：你觉得你会了并不一定会了，你认为对的并不一定是对的 * 勤动手，自行完成学习笔记和代码的书写。通过每一个小目标培养自己的兴趣以及成就感。   **什么是shell脚本**  Shell是一个命令解释器，他在操作系统的最外层，负责直接与用户对话，把用户的输入解释给操作，并处理各种各样的操作系统的输出结果，输出屏幕返回给用户。  查看shell的目录：echo $SHELL    **Shell脚本三要素**  命令语言  变量  逻辑判断  **Shell脚本的建立**   * 脚本放在统一的目录中   mkdir –p /server/scripts/   * 推荐使用vim编辑器编辑脚本 * 以.sh为扩展名 * 规范的shell脚本第一行会指出有哪个程序（解释器）来执行脚本中的内容   #！/bin/bash  或  #！/bin/sh  开头的“#！”称为幻数，在执行shell脚本的时候，内核会根据“#！”后的解释器来确定用哪个程序解释脚本中的内容。注意：这一行必须在每个脚本顶端的第一行，如果不是第一行则为脚本注释行。  脚本注释：   * 在shell脚本中，跟在#后面的内容表示注释。注释部分不会被执行，仅给人看。注释可以自成一行，也可以跟在命令后面，与命令同行。要养成写注释的习惯，方便自己与他人 * 最好不用中文注释，因为在不同字符集的系统会出现乱码。   **添加版权信息**  [root@rainjin ~]# pwd /root  [root@rainjin ~]# cat .vimrc autocmd BufNewFile \*.py,\*.cc,\*.sh,\*.java exec ":call SetTitle()"  func SetTitle() if expand("%:e") == 'sh' call setline(1,"#!/bin/bash") call setline(2, "##############################################################") call setline(3, "# File Name: ".expand("%")) call setline(4, "# Version: V1.0") call setline(5, "# Author: Rain Jin") call setline(6, "# Blog Site: http://www.cnblogs.com/rainjin/") call setline(7, "# Created Time : ".strftime("%F %T")) call setline(8, "# Environment: CentOS 7.2 Kernal 3.10.0") call setline(9, "##############################################################") call setline(10, "") endif endfunc    自动添加版权信息要求：   1. .vim==rc 文件配置正确 2. 创建新的文件 3. .sh结尾   **Shell脚本的执行**    **变量**  **什么是变量**    **变量命名规范**    **变量分类**    **全局变量**      说明：  前两个配置文件针对全局变量生效。后两个配置文件对当前变量生效  环境变量小结：    **普通变量**    变量与引号：    普通变量命名规则：    **特殊变量—位置变量**      位置变量$0  [root@lb03 scripts]# vim oldboy.sh  [root@lb03 scripts]# tail -1 oldboy.sh  echo "执行脚本名称" $0  [root@lb03 scripts]# sh oldboy.sh  执行脚本名称 oldboy.sh  [root@lb03 scripts]# sh /server/scripts/oldboy.sh  执行脚本名称 /server/scripts/oldboy.sh  [root@lb03 scripts]# cp oldboy.sh oldgirl.sh  [root@lb03 scripts]# sh oldgirl.sh  执行脚本名称 oldgirl.sh  位置变量$1 $2 ...  [root@lb03 scripts]# vim oldboy.sh  [root@lb03 scripts]# tail -4 oldboy.sh  echo "执行脚本名称" $0  echo "第1个参数" $1  echo "第2个参数" $2  echo "第3个参数" $3  [root@lb03 scripts]# sh oldboy.sh  执行脚本名称 oldboy.sh  第1个参数  第2个参数  第3个参数  [root@lb03 scripts]# sh oldboy.sh arg1  执行脚本名称 oldboy.sh  第1个参数 arg1  第2个参数  第3个参数  [root@lb03 scripts]# sh oldboy.sh arg1 arg2 arg3  执行脚本名称 oldboy.sh  第1个参数 arg1  第2个参数 arg2  第3个参数 arg3  [root@lb03 scripts]# sh oldboy.sh arg1 arg2 arg3 arg4  执行脚本名称 oldboy.sh  第1个参数 arg1  第2个参数 arg2  第3个参数 arg3  $@与$\*的区别：  [root@lb03 scripts]# set -- "I am" handsome boy..  [root@lb03 scripts]# echo $1  I am  [root@lb03 scripts]# echo $0  bash  [root@lb03 scripts]# echo $3  boy..  [root@lb03 scripts]# echo "$\*"  I am handsome boy..  [root@lb03 scripts]# echo "$@"  I am handsome boy..  [root@lb03 scripts]# for i in $\*;do echo $i;done;  I  am  handsome  boy..  [root@lb03 scripts]# for i in "$\*";do echo $i;done;  I am handsome boy..  [root@lb03 scripts]# for i in $@;do echo $i;done;  I  am  handsome  boy..  [root@lb03 scripts]# for i in "$@";do echo $i;done;  I am  handsome  boy.. | |
| 实习总结 | 今天我们学习了Shell脚本的基础语法  第一:shell脚本的编写用vim来编写,#!/bin/bash是告诉shell我们要用bash解释器这一行是必须的  第二:变量的声明和使用  第三:我们还学习了变量的分类,有全局变量,环境变量和特殊变量等等 | |

|  |  |  |
| --- | --- | --- |
| **学习报告** | | |
| 学习日期（第2天） | | 2023年2月 21 日 |
| 目的 | 正则表达式 | |
| 内容 | **特殊符号**   1. > 输出重定向 2. >> 追加输出重定向 3. < 输入重定向 4. << 追加输入重定向   cat 向一个文件追加多行。  cat >>/oldboy/alex.txt<<EOF  Love me, love my dog.  Love Li, love li's dog.  EOF  表示位置的   1. .点 当前目录 2. ..点点 当前目录的上以及目录 3. ~ 当前目录的家目录   特殊符号   1. # 注释 root用户的命令提示符 2. $ 取变量的内容（命令行）取某一列 普通用户的命令提示符 3. ！查找最近一次使用过的命令然后执行 find排除/取反 awk取反vim强制 4. | 管道 5. ；分割多条命令   引号系列   1. `` $() 先执行命令 2. ‘’所见即所得 3. “”解析特殊符号   简单判断   1. && 前一个命令执行成功，执行后面的命令   Ifdown eth0 && ifup eth0   1. || 前一个命令执行失败，执行后面的命令   **通配符**  1.什么作用？  方便大家查找出文件的（文件名）。  Linux大部分命令都支持。  \*.log  \*.txt  2.常见的通配符  1. \* 所有 任何东西  以.txt结尾的 \*.txt  以.log结尾的\*.log  系统中以ls开头的文件  find / -type f -name “ls\*”  找出系统中文件名包含oldboy的文件  Find -type f -name “\*oldboy\*”  2.{} 生成序列  echo {1..10}  echo {10..1}  echo {01..10}  echo {01..100}  echo {a..c}  echo {a..z}  echo {A..Z}  [root@oldboyedu42-lnb ~]# echo stu{01..10}  stu01 stu02 stu03 stu04 stu05 stu06 stu07 stu08 stu09 stu10  [root@oldboyedu42-lnb ~]# echo 20{01..10}  2001 2002 2003 2004 2005 2006 2007 2008 2009 2010  #通过{}  [root@oldboyedu42-lnb ~]# echo {a..z}  a b c d e f g h i j k l m n o p q r s t u v w x y z  [root@oldboyedu42-lnb ~]# echo {a c f}  {a c f}  [root@oldboyedu42-lnb ~]# echo {a,c,f}  a c f  [root@oldboyedu42-lnb ~]# echo A{B,C}  AB AC  [root@oldboyedu42-lnb ~]# echo A{,C}  A AC  [root@oldboyedu42-lnb ~]# echo oldboy.txt{,.bak}  oldboy.txt oldboy.txt.bak  [root@oldboyedu42-lnb ~]# touch oldboy.txt  [root@oldboyedu42-lnb ~]# cp oldboy.txt{,.bak}  cp: overwrite `oldboy.txt.bak'? y  [root@oldboyedu42-lnb ~]# ls -l oldboy.txt\*  [root@oldboyedu42-lnb ~]# ls -l oldboy.txt\*  -rw-r--r-- 3 root root 0 Nov 11 23:50 oldboy.txt  -rw-r--r--. 1 root root 0 Nov 11 23:50 oldboy.txt.bak  -rw-r--r-- 3 root root 0 Nov 11 23:50 oldboy.txt-hard  [root@oldboyedu42-lnb ~]#  [root@oldboyedu42-lnb ~]# #cp oldboy.txt{,.bak}  [root@oldboyedu42-lnb ~]# echo oldboy.txt{,.bak}  oldboy.txt oldboy.txt.bak  [root@oldboyedu42-lnb ~]# echo A{,C}  A AC    3.通配符小结  1）查找文件  2）\* {}  **正则表达式 RE regular expression**  通过符号匹配查找出各种文字。  正则表达式通过特殊符号 ^ $ [] . \* 表示各种各样的文字。  方便我们处理文本（日志）。  2.谁可以使用正则  三剑客正则(grep sed awk )  python java    3.正则表达式与通配符区别  正则---在文件中进行过滤（查找文件内容） 三剑客支持  通配符-找出文件（文件名） 大部分命令都可以使用    4.使用正则注意事项:  1) 正则默认是按照行为单位处理。  2) 一定要注意不要使用中文符号。  5.正则分类  基础正则: ^ $ . \* []    扩展正则: + | () {} ?  6.环境准备  /oldboy/re.txt  I am oldboy teacher!  I teach linux.  I like badminton ball ,billiard ball and chinese chess!  my blog is http://oldboy.blog.51cto.com  our size is http://blog.oldboyedu.com  my qq is 49000448  not 4900000448.  my god ,i am not oldbey,but OLDBOY!  7.基础正则-BRE basic regular expression  1) ^ ^m 表示以....开头的行  [root@oldboyedu42-lnb oldboy]# grep '^m' re.txt  my blog is http://oldboy.blog.51cto.com  my qq is 49000448  my god ,i am not oldbey,but OLDBOY!  2) $ m$ 表示以....结尾的行  [root@oldboyedu42-lnb oldboy]# cat -A re.txt  I am oldboy teacher!$  I teach linux.$  $  I like badminton ball ,billiard ball and chinese chess!$  my blog is http://oldboy.blog.51cto.com $ #注意此处结尾是空格不是m  $  our size is http://blog.oldboyedu.com $ #注意此处结尾是空格不是m  $  my qq is 49000448$  $  not 4900000448.$  my god ,i am not oldbey,but OLDBOY!$  #删除结尾的空格  [root@oldboyedu42-lnb oldboy]# vim re.txt    I am oldboy teacher!  I teach linux.  I like badminton ball ,billiard ball and chinese chess!  my blog is http://oldboy.blog.51cto.com  our size is http://blog.oldboyedu.com  my qq is 49000448  not 4900000448.  my god ,i am not oldbey,but OLDBOY!    #检查是否修改成功  [root@oldboyedu42-lnb oldboy]# cat -A re.txt  I am oldboy teacher!$  I teach linux.$  $  I like badminton ball ,billiard ball and chinese chess!$  my blog is http://oldboy.blog.51cto.com$  $  our size is http://blog.oldboyedu.com$  $  my qq is 49000448$  $  not 4900000448.$  my god ,i am not oldbey,but OLDBOY!$  [root@oldboyedu42-lnb oldboy]#  [root@oldboyedu42-lnb oldboy]# grep 'm$' re.txt  my blog is http://oldboy.blog.51cto.com  our size is http://blog.oldboyedu.com  cat -A #显示出文件中所有的符号 $这一行的结尾  [root@oldboyedu42-lnb oldboy]# grep ' ' re.txt  I am oldboy teacher!  I teach linux.  I like badminton ball ,billiard ball and chinese chess!  my blog is http://oldboy.blog.51cto.com  our size is http://blog.oldboyedu.com  my qq is 49000448  not 4900000448.  my god ,i am not oldbey,but OLDBOY!  3）^$ 空行 这一行里面什么都没有  [root@oldboyedu42-lnb oldboy]# grep -n '^$' re.txt  3:  6:  8:  10:    练习题:排查文件中的空行  [root@oldboyedu42-lnb oldboy]# grep -v '^$' re.txt  I am oldboy teacher!  I teach linux.  I like badminton ball ,billiard ball and chinese chess!  my blog is http://oldboy.blog.51cto.com  our size is http://blog.oldboyedu.com  my qq is 49000448  not 4900000448.  my god ,i am not oldbey,but OLDBOY!    4）. 任意一个字符 不会匹配空行  #grep -o '.' re.txt  #-o 显示grep的执行过程，grep每一次找出什么  grep -o '.' re.txt  #-o 显示grep的执行过程，grep每一次找出什么  #-o 的结果中 每一行表示 grep每次找出什么  [root@oldboyedu42-lnb oldboy]# grep '0' re.txt  my qq is 49000448  not 4900000448.  [root@oldboyedu42-lnb oldboy]# grep -o '0' re.txt  0  0  0  0  0  0  0  0  [root@oldboyedu42-lnb oldboy]# grep '00' re.txt  my qq is 49000448  not 4900000448.  [root@oldboyedu42-lnb oldboy]# grep -o '00' re.txt  00  00  00  5） \ 撬棍 转义字符 去掉符号特殊含义 脱掉马甲，打回原形    显示出文件中 以.结尾的行？    [root@oldboyedu42-lnb oldboy]# grep '\.$' re.txt  I teach linux.  not 4900000448.    撬棍系列 转义字符系列  \n ====== 回车      6）\* 前一个字符连续出现0次或0次以上    [root@oldboyedu42-lnb oldboy]# grep '0\*' re.txt  I am oldboy teacher!  I teach linux.  I like badminton ball ,billiard ball and chinese chess!  my blog is http://oldboy.blog.51cto.com  our size is http://blog.oldboyedu.com  my qq is 49000448  not 4900000448.  my god ,i am not oldbey,but OLDBOY!  [root@oldboyedu42-lnb oldboy]#  [root@oldboyedu42-lnb oldboy]# grep '0\*' re.txt -o  000  00000  #连续出现  #正则表示连续出现的时候，会尽可能的匹配（吃）更多的符号 贪婪性  [root@oldboyedu42-lnb oldboy]# #0\* 表示0连续出现0次  [root@oldboyedu42-lnb oldboy]# #0\* 表示0连续出现1次及1次以上  [root@oldboyedu42-lnb oldboy]# #0  [root@oldboyedu42-lnb oldboy]# #000  [root@oldboyedu42-lnb oldboy]# #0000000  [root@oldboyedu42-lnb oldboy]# #0\* 表示0连续出现0次  [root@oldboyedu42-lnb oldboy]# #'0\*' 只出现0次的时候 ====== ''  [root@oldboyedu42-lnb oldboy]# #会把整个文件的内容都显示出来  [root@oldboyedu42-lnb oldboy]# grep '' re.txt  I am oldboy teacher!  I teach linux.  I like badminton ball ,billiard ball and chinese chess!  my blog is http://oldboy.blog.51cto.com  our size is http://blog.oldboyedu.com  my qq is 49000448  not 4900000448.  my god ,i am not oldbey,but OLDBOY!    小结正则之\*：  1]] 贪婪性  2]] 连续出现  7）.\* 所有 任何符号    #在正则中表示连续出现 表示所有 贪婪性  找出文件中以字母m开头的行 并且 以m结尾的行  [root@oldboyedu42-lnb oldboy]# grep '^m' re.txt  my blog is http://oldboy.blog.51cto.com  my qq is 49000448  my god ,i am not oldbey,but OLDBOY!  [root@oldboyedu42-lnb oldboy]# grep '^m' re.txt |grep 'm$'  my blog is http://oldboy.blog.51cto.com  [root@oldboyedu42-lnb oldboy]# grep '^m.\*m$' re.txt  my blog is http://oldboy.blog.51cto.com  [root@oldboyedu42-lnb oldboy]# grep '^.\*m$' re.txt  my blog is http://oldboy.blog.51cto.com  our size is http://blog.oldboyedu.com  [root@oldboyedu42-lnb oldboy]# grep 'm$' re.txt  my blog is http://oldboy.blog.51cto.com  our size is http://blog.oldboyedu.com  [root@oldboyedu-42mvp ~]# grep '.\*m' re.txt  8）[] [abc] 表示一个整体，a或b或c任意一个字符    [root@oldboyedu42-lnb oldboy]# grep '[abc]' re.txt  I am oldboy teacher!  I teach linux.  I like badminton ball ,billiard ball and chinese chess!  my blog is http://oldboy.blog.51cto.com  our size is http://blog.oldboyedu.com  my god ,i am not oldbey,but OLDBOY!  grep '[abc]' re.txt -o      [root@oldboyedu42-lnb oldboy]# grep '[a-z]' re.txt  I am oldboy teacher!  I teach linux.  I like badminton ball ,billiard ball and chinese chess!  my blog is http://oldboy.blog.51cto.com  our size is http://blog.oldboyedu.com  my qq is 49000448  not 4900000448.  my god ,i am not oldbey,but OLDBOY!  [root@oldboyedu42-lnb oldboy]# grep '[A-Z]' re.txt  I am oldboy teacher!  I teach linux.  I like badminton ball ,billiard ball and chinese chess!  my god ,i am not oldbey,but OLDBOY!  [root@oldboyedu42-lnb oldboy]# grep '[a-zA-Z]' re.txt  I am oldboy teacher!  I teach linux.  I like badminton ball ,billiard ball and chinese chess!  my blog is http://oldboy.blog.51cto.com  our size is http://blog.oldboyedu.com  my qq is 49000448  not 4900000448.  my god ,i am not oldbey,but OLDBOY!    #找出文件中以m或n开头的行  [root@oldboyedu42-lnb oldboy]# #第1个里程碑-m或n  [root@oldboyedu42-lnb oldboy]# grep '[mn]' re.txt  I am oldboy teacher!  I teach linux.  I like badminton ball ,billiard ball and chinese chess!  my blog is http://oldboy.blog.51cto.com  our size is http://blog.oldboyedu.com  my qq is 49000448  not 4900000448.  my god ,i am not oldbey,but OLDBOY!  [root@oldboyedu42-lnb oldboy]# #第2个里程碑-以m或n开头的行  [root@oldboyedu42-lnb oldboy]# grep '^[mn]' re.txt  my blog is http://oldboy.blog.51cto.com  my qq is 49000448  not 4900000448.  my god ,i am not oldbey,but OLDBOY!    grep '[a-z]' re.txt  grep '[A-Z]' re.txt  grep '[0-9]' re.txt | |
| 实  习  总 结 | 正则表达式语法：了解正则表达式的基本语法，包括元字符、字符类、重复符、定位符等等。这是掌握正则表达式的基础，也是学习 shell 脚本正则表达式的基础。  在 shell 脚本中使用正则表达式：学习如何在 shell 脚本中使用正则表达式来匹配文本、替换文本等操作。掌握 shell 脚本中的正则表达式语法和用法。  常用正则表达式实例：学习正则表达式在实际应用中的常见用法，比如匹配数字、字母、空格、邮箱地址、URL等等。 | |

|  |  |  |
| --- | --- | --- |
| **学习报告** | | |
| 学习日期（第3天） | | 2023年2月 22 日 |
| 目的 | If条件判断 | |
| 内容 | 在shell脚本中，条件判断语句可以用于判断一个条件是否成立，并根据判断结果执行不同的操作。常用的条件判断语句包括if语句、case语句和test命令。  if语句的语法格式为：  if condition then  command1  command2 ...  else  command3  command4 ...  fi  其中，condition为判断条件，如果成立则执行then后面的命令，否则执行else后面的命令。  case语句的语法格式为：  case variable in  pattern1) command1 ;;  pattern2) command2 ;;  pattern3) command3 ;;  \*) default\_command ;;  esac  其中，variable为需要判断的变量，pattern1、pattern2、pattern3等为匹配模式，如果variable与某个模式匹配，则执行对应的命令，如果都不匹配，则执行default\_command。  test命令可以用于判断某个条件是否成立，其语法格式为：  test condition  或者简写为：  [ condition ]  其中，condition为需要判断的条件，如果成立则返回0，否则返回1。  除了以上的条件判断语句，还可以使用逻辑运算符（&&、||、!）和比较运算符（-eq、-ne、-gt、-lt、-ge、-le）进行更复杂的条件判断。  检查文件是否存在  if [ -f /path/to/file ]; then  echo "File exists"  else  echo "File does not exist"  fi  检查文件是否可写  if [ -w /path/to/file ]; then  echo "File is writable"  else  echo "File is not writable"  fi  检查目录是否存在  if [ -d /path/to/directory ]; then  echo "Directory exists"  else  echo "Directory does not exist"  fi  检查字符串是否为空  if [ -z "$string" ]; then  echo "String is empty"  else  echo "String is not empty"  fi  检查两个数值是否相等  if [ "$num1" -eq "$num2" ]; then  echo "Numbers are equal"  else  echo "Numbers are not equal"  fi | |
| 实  习  总 结 | 1.学习if条件判断的基本语法  2.学习case多重判断的基本语法  3.学习了test语句的判断 | |
| **学习报告** | | |
| 学习日期（第4天） | | 2023年2月 23 日 |
| 目的 | 用户管理 | |
| 内容 | **用户管理**  **命令解释器(shell)**  /bin/bash 用户默认的  /sbin/nologin 用户是虚拟用户 无法登陆系统  #与用户有关的目录  /etc/skel/ <新用户>的家目录的模板  添加新用户的时候 系统会把目录中的文件 复制到新用户家里  [root@oldboyedu42-lnb ~]# cat /etc/skel/readme.txt  爱生活,远离rm  [root@oldboyedu42-lnb ~]# useradd liyy  [root@oldboyedu42-lnb ~]# su - liyy  [liyy@oldboyedu42-lnb ~]$ ll -a  total 24  drwx------ 2 liyy liyy 4096 Nov 13 09:12 .  drwxr-xr-x. 5 root root 4096 Nov 13 09:12 ..  -rw-r--r-- 1 liyy liyy 18 Mar 23 2017 .bash\_logout  -rw-r--r-- 1 liyy liyy 176 Mar 23 2017 .bash\_profile  -rw-r--r-- 1 liyy liyy 124 Mar 23 2017 .bashrc  -rw-r--r-- 1 liyy liyy 19 Nov 13 09:11 readme.txt  [liyy@oldboyedu42-lnb ~]$ ll  total 4  -rw-r--r-- 1 liyy liyy 19 Nov 13 09:11 readme.txt  [liyy@oldboyedu42-lnb ~]$ cat readme.txt  爱生活,远离rm      [root@oldboyedu42-lnb ~]# ls -la /etc/skel/  total 20  drwxr-xr-x. 2 root root 4096 Oct 16 12:38 .  drwxr-xr-x. 78 root root 4096 Nov 8 11:34 ..  -rw-r--r--. 1 root root 18 Mar 23 2017 .bash\_logout  -rw-r--r--. 1 root root 176 Mar 23 2017 .bash\_profile #/etc/profile 用户自己的环境变量和别名  -rw-r--r--. 1 root root 124 Mar 23 2017 .bashrc #/etc/bashrc 别名 这个用户生效  **#切换到某个用户，提示 -bash-4.1$**  ###模拟  [alex@oldboyedu42-lnb ~]$ whoami  alex  [alex@oldboyedu42-lnb ~]$ rm ~/.\* -fr  command bny . .. .bash\_history .bash\_logout .bash\_profile .bashrc -fr  [alex@oldboyedu42-lnb ~]$ \rm ~/.\* -fr  rm: cannot remove directory: `.'  rm: cannot remove directory: `..'    ###原因  与用户环境变量有关的文件被删除    ###解决  从哪里复制一份 /etc/skel /home/oldboy  切换到alex用户下面  [root@oldboyedu42-lnb ~]# su - alex  -bash-4.1$  -bash-4.1$ cp /etc/skel/.bash\* ~  重新登陆用户  **#与用户有关的命令**  ##useradd  添加一个用户alex999指定uid为888 禁止用户登录系统 不创建家目录  添加一个uid是888的虚拟用户alex999    [root@oldboyedu42-lnb ~]# useradd -u 888 -s /sbin/nologin -M alex888  [root@oldboyedu42-lnb ~]# id alex888  uid=888(alex888) gid=888(alex888) groups=888(alex888)  [root@oldboyedu42-lnb ~]# grep alex888 /etc/passwd  alex888:x:888:888::/home/alex888:/sbin/nologin  [root@oldboyedu42-lnb ~]#  [root@oldboyedu42-lnb ~]# su - alex888  su: warning: cannot change directory to /home/alex888: No such file or directory  This account is currently not available.  [root@oldboyedu42-lnb ~]# ls /home/alex888  ls: cannot access /home/alex888: No such file or directory  **系统安全**  1.最小化  2.禁止root远程登录 更改系统端口号  3.隐藏文件属性 chattr lsattr  4.给重要的文件或命令 做一个指纹 定时任务+md5sum定时检查  **给文件创建指纹**  #文件的内容变化 === 指纹变化  [root@oldboyedu42-lnb ~]# md5sum oldboy.txt  bdb4cba31f30cdee67dc74a609bd285f oldboy.txt  [root@oldboyedu42-lnb ~]# echo oldboyedu.com >>oldboy.txt  [root@oldboyedu42-lnb ~]# md5sum oldboy.txt  892899f82caab30dda848c54668c9c77 oldboy.txt    #2.如何快速的对比指纹  ## 1)把你的指纹记录下来  [root@oldboyedu42-lnb ~]# md5sum oldboy.txt  892899f82caab30dda848c54668c9c77 oldboy.txt  [root@oldboyedu42-lnb ~]# md5sum oldboy.txt >>zhiwen.log  [root@oldboyedu42-lnb ~]# cat zhiwen.log  892899f82caab30dda848c54668c9c77 oldboy.txt  ## 2)如何进行对比  [root@oldboyedu42-lnb ~]# md5sum -c zhiwen.log  oldboy.txt: OK  [root@oldboyedu42-lnb ~]# >oldboy.txt  [root@oldboyedu42-lnb ~]# md5sum -c zhiwen.log  oldboy.txt: FAILED  md5sum: WARNING: 1 of 1 computed checksum did NOT match  把/e*tc/passwd /*etc/shadow /etc/group /etc/gshadow 指纹放在 /tmp/zhiwen.log中  并进行指纹对比。  [root@oldboyedu42-lnb ~]# ###1.创建指纹列表  [root@oldboyedu42-lnb ~]# md5sum /etc/passwd /etc/shadow /etc/group /etc/gshadow  657d3be9ba430cd969353d188d856355 /etc/passwd  c66ff31b39a816af3826ed73838c7413 /etc/shadow  3618108e39a370e4f9df7c0be59a5a90 /etc/group  075e1b5ee7849f3f096a585272afd6b7 /etc/gshadow  [root@oldboyedu42-lnb ~]# md5sum /etc/passwd /etc/shadow /etc/group /etc/gshadow >/tmp/zhiwen.log  [root@oldboyedu42-lnb ~]#  [root@oldboyedu42-lnb ~]# cat /tmp/zhiwen.log  657d3be9ba430cd969353d188d856355 /etc/passwd  c66ff31b39a816af3826ed73838c7413 /etc/shadow  3618108e39a370e4f9df7c0be59a5a90 /etc/group  075e1b5ee7849f3f096a585272afd6b7 /etc/gshadow  [root@oldboyedu42-lnb ~]# #2.根据指纹列表对比文件内容是否变化  [root@oldboyedu42-lnb ~]# md5sum --check /tmp/zhiwen.log  /etc/passwd: OK  /etc/shadow: OK  /etc/group: OK  /etc/gshadow: OK  [root@oldboyedu42-lnb ~]# md5sum -c /tmp/zhiwen.log  /etc/passwd: OK  /etc/shadow: OK  /etc/group: OK  /etc/gshadow: OK  #什么时候需要做指纹  命令或配置文件  C:\Users\Administrator\Desktop\day22-用户管理 .png | |
| 实  习  总 结 | 1. 添加用户：使用“useradd”命令来添加新用户。 2. 删除用户：使用“userdel”命令来删除用户。 3. 修改用户密码：使用“passwd”命令来修改用户密码。 4. 查看用户信息：使用“id”命令来查看用户ID和组ID，使用“whoami”命令来查看当前用户的用户名。 5. 修改用户信息：使用“usermod”命令来修改用户信息，例如修改用户的用户名、主目录、shell等。 6. 检查用户是否存在：使用“id”命令来检查用户是否存在。 7. 批量添加用户：使用循环来批量添加用户。 | |

|  |  |  |
| --- | --- | --- |
| **学习报告** | | |
| 学习日期（第5天） | | 2023年2月 24日 |
| 目的 | For循环 | |
| 内容 | **shell-for循环**  for 变量 in 列表  do  命令  done  for name in 1 2 3 4 5  do  echo $name  done    for name in {a..z}  do  echo $name  done  显示最近十天的日期，格式为2017-11-11    2017-11-01  2017-11-02  2017-11-03  2017-11-04  2017-11-05  2017-11-06  2017-11-07  方法1  2017-11-01 2017-11-02 2017-11-03 2017-11-04 2017-11-05 2017-11-06 2017-11-07  for time in 2017-11-{01..10}  do  echo $time  done  方法2  for time in {1..10}  do  date -d "-${time}day" +%F  done  [root@oldboyedu42-lnb ~]# week=7  [root@oldboyedu42-lnb ~]# echo $week  7  [root@oldboyedu42-lnb ~]# echo $weekday  [root@oldboyedu42-lnb ~]# echo $week\_day  [root@oldboyedu42-lnb ~]# #金庸新著  [root@oldboyedu42-lnb ~]# #金庸新著  [root@oldboyedu42-lnb ~]# #金庸 新著  [root@oldboyedu42-lnb ~]# #金庸新 著  [root@oldboyedu42-lnb ~]#  [root@oldboyedu42-lnb ~]# echo ${week}day  7day  [root@oldboyedu42-lnb ~]# echo ${week}\_day  7day  **优化系统的开机启动项目**  1.必须要开机自启动  crond 定时任务服务  network 网络服务  sshd 远程连接服务  rsyslog 记录系统日志服务  sysstat 收集系统状态  2.关闭除了这5个服务之外  chkconfig abrt-ccpp off  chkconfig abrtd off  chkconfig acpid off  chkconfig atd off  chkconfig auditd off  chkconfig blk-availability off  chkconfig cpuspeed off  chkconfig haldaemon off  chkconfig ip6tables off  chkconfig iptables off  chkconfig irqbalance off  chkconfig kdump off  chkconfig lvm2-monitor off  chkconfig mdmonitor off  chkconfig messagebus off  chkconfig netfs off  chkconfig oldboyd off  chkconfig rdma off  chkconfig udev-post off  3.通过循环完成  for name in 服务名字的列表  do  命令  done  4.除了这5个服务之外 服务名字的列表  [root@oldboyedu42-lnb ~]# chkconfig |egrep -v "crond|network|sshd|rsyslog|sysstat"|awk '{print $1}'  abrt-ccpp  abrtd  acpid  atd  auditd  blk-availability  cpuspeed  haldaemon  ip6tables  iptables  irqbalance  kdump  lvm2-monitor  mdmonitor  messagebus  netconsole  netfs  nfs-rdma  ntpd  ntpdate  oldboyd  postfix  psacct  quota\_nld  rdisc  rdma  restorecond  rngd  saslauthd  smartd  svnserve  udev-post  5.使用循环-不让他执行    for name in 服务名字的列表  do  echo chkconfig $name off  done    for name in $(chkconfig |egrep -v "crond|network|sshd|rsyslog|sysstat"|awk '{print $1}')  do  echo chkconfig $name off  done  6.使用循环  for name in $(chkconfig |egrep -v "crond|network|sshd|rsyslog|sysstat"|awk '{print $1}')  do  chkconfig $name off  done  7.检查  [root@oldboyedu42-lnb ~]# chkconfig |grep 3:on  crond 0:off 1:off 2:on 3:on 4:on 5:on 6:off  network 0:off 1:off 2:on 3:on 4:on 5:on 6:off  rsyslog 0:off 1:off 2:on 3:on 4:on 5:on 6:off  sshd 0:off 1:off 2:on 3:on 4:on 5:on 6:off  sysstat 0:off 1:on 2:on 3:on 4:on 5:on 6:off  **Awk**  **awk课前准备**  [root@oldboyedu42-lnb ~]# mkdir -p /server/files  [root@oldboyedu42-lnb ~]# cd /server/files  [root@oldboyedu42-lnb files]# pwd  /server/files  [root@oldboyedu42-lnb files]# yum install lrzsz -y  [root@oldboyedu42-lnb files]# rpm -qa lrzsz  lrzsz-0.12.20-27.1.el6.x86\_64  上传 02-【awk练习文件】secure-20161219 01-【awk练习文件】access 文件到/server/files 里面  windows上传到linux里面 rz  Linux文件下载到windows sz 文件名  **awk执行过程**  '找谁{干啥}'  '条件{命令}'  [root@oldboyedu42-lnb files]# awk -F: 'NR==1{print $1,$3}' /etc/passwd  root 0  2）找谁{干啥} awk里面如何过滤  1# 符号 大于 小于 等于  NR==1  NR>1  NR>=1  NR<=10  NR<10  **2# 正则表达式作为条件\*\*\*\*\***  mkdir -p /server/files/  cat >>/server/files/reg.txt<<EOF  Zhang Dandan 41117397 :250:100:175  Zhang Xiaoyu 390320151 :155:90:201  Meng Feixue 80042789 :250:60:50  Wu Waiwai 70271111 :250:80:75  Liu Bingbing 41117483 :250:100:175  Wang Xiaoai 3515064655 :50:95:135  Zi Gege 1986787350 :250:168:200  Li Youjiu 918391635 :175:75:300  Lao Nanhai 918391635 :250:100:175  EOF  ~ 匹配  !~ 不匹配  [root@oldboyedu42-lnb files]# awk '第2列中有o' reg.txt  [root@oldboyedu42-lnb files]# awk '$2~/o/' reg.txt  Zhang Xiaoyu 390320151 :155:90:201  Wang Xiaoai 3515064655 :50:95:135  Li Youjiu 918391635 :175:75:300  [root@oldboyedu42-lnb files]# #第3列中以数字9开头的  [root@oldboyedu42-lnb files]# awk '$3~/^9/' reg.txt  Li Youjiu 918391635 :175:75:300  Lao Nanhai 918391635 :250:100:175  显示Xiaoyu的姓氏和ID号码  [root@oldboyedu42-lnb files]##显示Xiaoyu的姓氏和ID号码  [root@oldboyedu42-lnb files]# awk '$2~/Xiaoyu/' reg.txt  Zhang Xiaoyu 390320151 :155:90:201  [root@oldboyedu42-lnb files]# awk '$2~/Xiaoyu/{print $1,$2,$3}' reg.txt  Zhang Xiaoyu 390320151  [root@oldboyedu42-lnb files]# awk '$2~/^Xiaoyu$/{print $1,$2,$3}' reg.txt  Zhang Xiaoyu 390320151  姓氏是Zhang的人，显示他的第二次捐款金额及她的名字  [root@oldboyedu42-lnb files]# awk -F"[ :]+" '$1~/^Zhang$/{print $1,$2,$5}' reg.txt  Zhang Dandan 100  Zhang Xiaoyu 90  [root@oldboyedu42-lnb files]# awk -F"[ :]+" '$1~/^Zhang$/{print $1,$2,$(NF-1)}' reg.txt  Zhang Dandan 100  Zhang Xiaoyu 90  显示所有ID号码最后一位数字是1或5的人的全名  [root@oldboyedu42-lnb files]# awk '$3~/[15]$/' reg.txt  Zhang Xiaoyu 390320151 :155:90:201  Wu Waiwai 70271111 :250:80:75  Wang Xiaoai 3515064655 :50:95:135  Li Youjiu 918391635 :175:75:300  Lao Nanhai 918391635 :250:100:175  [root@oldboyedu42-lnb files]# awk '$3~/[15]$/{print $1,$2}' reg.txt  Zhang Xiaoyu  Wu Waiwai  Wang Xiaoai  Li Youjiu  Lao Nanhai  [root@oldboyedu42-lnb files]# awk '$3~/[15]$/{print $1,$2}' reg.txt |column -t  Zhang Xiaoyu  Wu Waiwai  Wang Xiaoai  Li Youjiu  Lao Nanhai  显示Xiaoyu的捐款.每个值时都有以$开头.如$520$200$135  's###g'  substitute sub  gsub ==== 's###g'  **#gsub(/找谁/,"替换成啥",某一列) #在哪一列中 ，把谁替换成什么。 \*\*\*\*\*\***  #  #gsub(/找谁/,"替换成啥") #这一行中 ，把谁替换成什么。  #gsub(/找谁/,"替换成啥",$0) #这一行中 ，把谁替换成什么。  #  [root@oldboyedu42-lnb files]# awk '{gsub(/:/,"$",$NF);print $NF}' reg.txt  $250$100$175  $155$90$201  $250$60$50  $250$80$75  $250$100$175  $50$95$135  $250$168$200  $175$75$300  $250$100$175  [root@oldboyedu42-lnb files]# awk '$2~/^Xiaoyu$/{gsub(/:/,"$",$NF);print $NF}' reg.txt  $155$90$201  小结：awk条件  1.正则表达式作为条件\*\*\*\*\*  2.比较作为条件  3.范围  'NR==1,NR==10'  '/^101/,/^105/'  **awk计算统计基础**  1.'END{}' 表示END{}里面的内容，会在awk读取完最后一行后执行  先计算，然后再END{}中显示最后的结果。  [root@oldboyedu42-lnb files]# awk '{print $0}END{print "end of file"}' reg.txt  Zhang Dandan 41117397 :250:100:175  Zhang Xiaoyu 390320151 :155:90:201  Meng Feixue 80042789 :250:60:50  Wu Waiwai 70271111 :250:80:75  Liu Bingbing 41117483 :250:100:175  Wang Xiaoai 3515064655 :50:95:135  Zi Gege 1986787350 :250:168:200  Li Youjiu 918391635 :175:75:300  Lao Nanhai 918391635 :250:100:175  end of file  统计下/etc/services文件中有多少空行？  #i=i+1 进行wc -l 统计次数  [root@oldboyedu42-lnb files]# awk '/^$/{i=i+1;print i}' /etc/services |tail -1  16  [root@oldboyedu42-lnb files]# awk '/^$/{i=i+1}END{print i}' /etc/services  16    #i=i+1 == i++    [root@oldboyedu42-lnb files]# seq 10 >num.txt  [root@oldboyedu42-lnb files]# cat num.txt  1  2  3  4  5  6  7  8  9  10  计算总和  [root@oldboyedu42-lnb files]# awk '{sum=sum+$0}END{print sum}' num.txt  55  总结：  1.i=i+1 i++ 进行计数 计数出现次数  2.i=i+$0 i+=$0 计算总和 累加 | |
| 实  习  总 结 | 1.本节学习了for循环的基本语法和for循环的嵌套  2.本节学习了awk命令  3.学习了正则表达式作为条件for循环  4.学习了awk计算统计基础 | |

|  |  |  |
| --- | --- | --- |
| **学习报告** | | |
| 学习日期（第6天） | | 2023年2月 25日 |
| 目的 | 文件,磁盘 | |
| 内容 | **文件、命令权限**  **#alex查看文件的内容**  ##1.改变这些日志的权限 rw----r--  ##2.给root密码  ##3.给cat 加上suid权限  ##4.？？  [alex@oldboyedu42-lnb ~]$  [alex@oldboyedu42-lnb ~]$ #reboot  [alex@oldboyedu42-lnb ~]$ ll /usr/bin/passwd  -rwsr-xr-x. 1 root root 30768 Nov 24 2015 /usr/bin/passwd  [alex@oldboyedu42-lnb ~]$  [alex@oldboyedu42-lnb ~]$ ll /usr/bin/passwd  -rwsr-xr-x. 1 root root 30768 Nov 24 2015 /usr/bin/passwd  [alex@oldboyedu42-lnb ~]$  [alex@oldboyedu42-lnb ~]$  #sudo--Linux尚方宝剑/黄马褂  #让某个用户运行某个命令的时候可以是root  **###sudo命令**  #看看我有什么 尚方宝剑  [alex@oldboyedu42-lnb ~]$ sudo -l  We trust you have received the usual lecture from the local System  Administrator. It usually boils down to these three things:  #1) Respect the privacy of others.  #2) Think before you type.  #3) With great power comes great responsibility. -spider man    Sorry, user alex may not run sudo on oldboyedu42-lnb.  抱歉， alex用户 不能在这台服务器运行sudo.  #给alex用户授权 cat  ###root用户  visudo ##编辑sudo授权  #第92行 插入  alex ALL=(ALL) /bin/cat  [root@oldboyedu42-lnb ~]# grep alex /etc/sudoers  alex ALL=(ALL) /bin/cat  ###alex用户  [alex@oldboyedu42-lnb ~]$ sudo -l  [sudo] password for alex:  Matching Defaults entries for alex on this host:  !visiblepw, always\_set\_home, env\_reset, env\_keep="COLORS DISPLAY HOSTNAME HISTSIZE INPUTRC KDEDIR  LS\_COLORS", env\_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC\_ADDRESS LC\_CTYPE", env\_keep+="LC\_COLLATE  LC\_IDENTIFICATION LC\_MEASUREMENT LC\_MESSAGES", env\_keep+="LC\_MONETARY LC\_NAME LC\_NUMERIC LC\_PAPER  LC\_TELEPHONE", env\_keep+="LC\_TIME LC\_ALL LANGUAGE LINGUAS \_XKB\_CHARSET XAUTHORITY",  secure\_path=/sbin\:/bin\:/usr/sbin\:/usr/bin  User alex may run the following commands on this host:  (ALL) /bin/cat  [alex@oldboyedu42-lnb ~]$ cat /var/log/cron  cat: /var/log/cron: Permission denied  [alex@oldboyedu42-lnb ~]$ sudo cat /var/log/cron  **#给alex用户授权一个tail命令**  ###编辑sudo的方法  #1.  visudo 直接编辑  #2.  vim /etc/sudoers 编辑  然后  visudo -c  ###授权用户多个命令  查询  tail grep less more head cat vim  /bin/grep, /usr/bin/tail, /usr/bin/less, /bin/more, /usr/bin/head, /bin/cat, /usr/bin/vim  alex ALL=(ALL) /bin/grep, /usr/bin/tail, /usr/bin/less, /bin/more, /usr/bin/head, /bin/cat, /usr/bin/vim  续行符号  ###授权某个用户所有命令 并不需要输入密码  alex ALL=(ALL) NOPASSWD: ALL  总结：  1.sudo 作用  2.给某个用户授权某个命令或多个命令  3.给某个用户授权所有命令并且不需要密码  **###磁盘**  #磁盘主要参数  磁盘转数:  磁盘接口: sata sas scsi pci-e  磁盘结构: 机械硬盘 固态硬盘（ssd）  #磁盘接口  sata 容量大，价格便宜 备份，不重要的数据  sas 服务器标配 稳定 读写速度快 可以使用sata接口的硬盘  pci-e 固态硬盘的接口    SAS机械  企业默认 15K 300G 600G  10k 1.8TB  7.2k 6TB  线上/生产环境使用  数据库，存储    SATA机械  线下环境使用  备份，不重要的数据    固态硬盘 SATA/SAS/PCI-E + 机械（sata）  速度快 价格高 容量小  高并发 小容量  ###机械硬盘 固态硬盘  容量 比较大 比较小  速度 慢 快  稳定 差 好  寿命 一直可用 1万-10万读写  ##磁盘内部组成  磁头--读写数据  盘片  磁头 head  磁道 tract  以主轴为圆心，圆环区域（呼啦圈）  当磁盘旋转时，磁头若保持在一个位置上，则每个磁头都会在磁盘表面划出一个圆形轨迹，这些圆形轨迹就叫做磁道。  扇区 sector  磁盘上最小的单位默认是512字节。  柱面 cylinder  ##磁盘读写过程  #磁头读写的时候是按照柱面进行的。  ##磁盘大小计算  柱面数量\*每个柱面的大小  Disk /dev/sda: 10.7 GB, 10737418240 bytes  255 heads, 63 sectors/track, 1305 cylinders  Units = cylinders of 16065 \* 512 = 8225280 bytes | |
| 实  习  总 结 | 1.sudo用户管理  2.常用的跳板机/堡垒机  3.磁盘接口：sata sas pci-e 使用场景  4.查询服务器常见的配置和型号 dell cpu型号 几路 几核 内存 磁盘最多可以几个硬盘  5.磁头 磁道 扇区 柱面 | |

|  |  |  |
| --- | --- | --- |
| **学习报告** | | |
| 学习日期（第7天） | | 2023年2月 27 日 |
| 目的 | 磁盘分区 | |
| 内容 | **磁盘、raid**  **磁盘阵列**  磁盘阵列（Redundant Arrays of Independent Disks，RAID），有“独立磁盘构成的具有冗余能力的阵列”之意。  磁盘阵列是由很多价格较便宜的[磁盘](https://baike.baidu.com/item/%E7%A3%81%E7%9B%98)，组合成一个容量巨大的磁盘组，利用个别磁盘提供数据所产生加成效果提升整个磁盘系统效能。利用这项技术，将数据切割成许多区段，分别存放在各个硬盘上。[1]  磁盘阵列还能利用同位检查（Parity Check）的观念，在[数组](https://baike.baidu.com/item/%E6%95%B0%E7%BB%84" \t "_blank)中任意一个[硬盘故障](https://baike.baidu.com/item/%E7%A1%AC%E7%9B%98%E6%95%85%E9%9A%9C)时，仍可读出数据，在数据重构时，将数据经计算后重新置入新硬盘中。  **Riad 0 没有奇偶校验的条带**  Riad 0 RAID 0 并不是真正的RAID结构，没有[数据冗余](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%86%97%E4%BD%99" \t "_blank)，没有数据校验的磁盘陈列。实现RAID 0至少需要两块以上的硬盘，它将两块以上的硬盘合并成一块，数据连续地分割在每块盘上。 因为带宽加倍，所以读/写速度加倍， 但RAID 0在提高性能的同时，并没有提供数据保护功能，只要任何一块硬盘损坏就会丢失所有数据。因此RAID 0 不可应用于需要数据高可用性的关键领域。  RAID 0的缺点是不提供[数据冗余](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%86%97%E4%BD%99" \t "_blank)，因此一旦用户数据损坏，损坏的数据将无法得到恢复。RAID0运行时只要其中任一块[硬盘](https://baike.baidu.com/item/%E7%A1%AC%E7%9B%98" \t "_blank)出现问题就会导致整个数据的故障。一般不建议企业用户单独使用。  RAID 0具有的特点，使其特别适用于对性能要求较高，而对数据安全不太在乎的领域，如[图形工作站](https://baike.baidu.com/item/%E5%9B%BE%E5%BD%A2%E5%B7%A5%E4%BD%9C%E7%AB%99" \t "_blank)等。对于个人用户，RAID 0也是提高[硬盘存储](https://baike.baidu.com/item/%E7%A1%AC%E7%9B%98%E5%AD%98%E5%82%A8" \t "_blank)性能的绝佳选择。  **RAID 1**  RAID 1通过[磁盘](https://baike.baidu.com/item/%E7%A3%81%E7%9B%98" \t "_blank)数据镜像实现[数据冗余](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%86%97%E4%BD%99)，在成对的独立磁盘上产生互 为备份的数据。当原始数据繁忙时，可直接从镜像拷贝中读取数据，因此RAID 1可以提高读取性能。RAID 1是[磁盘阵列](https://baike.baidu.com/item/%E7%A3%81%E7%9B%98%E9%98%B5%E5%88%97" \t "_blank)中单位成本最高的，但提供了很高的数据安全性和可用性。当一个磁盘失效时，系统可以自动切换到镜像磁盘上读写，而不需要重组失效的数据。  RAID1通过硬盘数据镜像实现数据的冗余，保护数据安全，在两块盘上产生互为备份的数据，当原始数据繁忙时，可直接从镜像备份中读取数据，因此RAID1可以提供读取性能。  RAID1是硬盘中单位成本最高的，但提供了很高的数据安全性和可用性，当一个硬盘失效时，系统可以自动切换到镜像硬盘上读/写，并且不需要重组失效的数据。  **RAID 5**  RAID 5 是一种存储性能、[数据安全](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%AE%89%E5%85%A8" \t "_blank)和存储成本兼顾的存储解决方案。 RAID 5可以理解为是RAID 0和RAID 1的折中方案。RAID 5可以为系统提供[数据安全](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%AE%89%E5%85%A8" \t "_blank)保障，但保障程度要比Mirror低而磁盘空间利用率要比Mirror高。RAID 5具有和RAID 0相近似的数据读取速度，只是多了一个[奇偶校验](https://baike.baidu.com/item/%E5%A5%87%E5%81%B6%E6%A0%A1%E9%AA%8C" \t "_blank)信息，写入数据的速度比对单个[磁盘](https://baike.baidu.com/item/%E7%A3%81%E7%9B%98)进行写入操作稍慢。同时由于多个数据对应一个[奇偶校验](https://baike.baidu.com/item/%E5%A5%87%E5%81%B6%E6%A0%A1%E9%AA%8C)信息，RAID 5的磁盘空间利用率要比[RAID 1](https://baike.baidu.com/item/RAID%201" \t "_blank)高，存储成本相对较低，是目前运用较多的一种解决方案  **Riad10**  Raid 10是一个Raid 1与[Raid](https://baike.baidu.com/item/Raid" \t "_blank)0的组合体，它是利用[奇偶校验](https://baike.baidu.com/item/%E5%A5%87%E5%81%B6%E6%A0%A1%E9%AA%8C" \t "_blank)实现条带集镜像，所以它继承了Raid0的快速和Raid1的安全。我们知道，RAID 1在这里就是一个冗余的备份阵列，而RAID 0则负责数据的读写阵列。  RAID10也被称为镜象阵列条带。像RAID0一样，数据跨磁盘抽取；像RAID1一样，每个磁盘都有一个镜象磁盘, 所以RAID 10的另一种会说法是 RAID 1+0。RAID10提供100%的数据冗余，支持更大的卷尺寸，但价格也相对较高。对大多数只要求具有冗余度而不必考虑价格的应用来说，RAID10提供最好的性能。使用RAID10，可以获得更好的可靠性，因为即使两个物理驱动器发生故障，每个阵列中都有一个，数据仍然可以得到保护。RAID10需要4 + 2\*N 个磁盘驱动器（N >=0)， 而且只能使用其中一半或更小的磁盘用量, 例如 4 个 250G 的硬盘使用RAID10 阵列， 实际容量是 500G。  **磁盘分区**  #第一列  磁盘分区或设备名字 /dev/sdb #第二列  挂载点 #第三列  文件系统的类型（ext4） #第四列  挂载参数 #第五列  是否进行备份      0表示不备份 #第六列  是否进行磁盘检查  0表示不检查  1.查看磁盘  [root@sunwuce ~]# df -h  Filesystem Size Used Avail Use% Mounted on  /dev/sda3 8.8G 1.5G 6.9G 18% /  tmpfs 926M 0 926M 0% /dev/shm  /dev/sda1 190M 35M 146M 19% /boot  /dev/sdb1 94M 1.6M 88M 2% /mnt  [root@sunwuce ~]# fdisk -l |grep '/dev/sd\*'  Disk /dev/sda: 10.7 GB, 10737418240 bytes  /dev/sda1 \* 1 26 204800 83 Linux  /dev/sda2 26 124 783360 82 Linux swap / Solaris  /dev/sda3 124 1306 9496576 83 Linux  Disk /dev/sdb: 106 MB, 106954752 bytes  /dev/sdb1 2 102 103424 83 Linux  Disk /dev/sdc: 106 MB, 106954752 bytes  2.创建分区  [root@sunwuce ~]# fdisk -cu /dev/sdc  Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel  Building a new DOS disklabel with disk identifier 0x5bf64fa3.  Changes will remain in memory only, until you decide to write them.  After that, of course, the previous content won't be recoverable.  Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)  Command (m for help): n  Command action  e extended  p primary partition (1-4)  p  Partition number (1-4):  Value out of range.  Partition number (1-4): 1  First sector (2048-208895, default 2048):  Using default value 2048  Last sector, +sectors or +size{K,M,G} (2048-208895, default 208895):  Using default value 208895  3.检查分区保存并推出  Command (m for help): p  Disk /dev/sdc: 106 MB, 106954752 bytes  64 heads, 32 sectors/track, 102 cylinders, total 208896 sectors  Units = sectors of 1 \* 512 = 512 bytes  Sector size (logical/physical): 512 bytes / 512 bytes  I/O size (minimum/optimal): 512 bytes / 512 bytes  Disk identifier: 0x5bf64fa3  Device Boot Start End Blocks Id System  /dev/sdc1 2048 208895 103424 83 Linux  Command (m for help): w  The partition table has been altered!  Calling ioctl() to re-read partition table.  Syncing disks.  4.告诉系统分区表变化了  [root@sunwuce ~]# partprobe /dev/sdc1  5.创建系统分区，关闭磁盘检查  [root@sunwuce ~]# mkfs.ext4 /dev/sdc1  mke2fs 1.41.12 (17-May-2010)  Filesystem label=  OS type: Linux  Block size=1024 (log=0)  Fragment size=1024 (log=0)  Stride=0 blocks, Stripe width=0 blocks  25896 inodes, 103424 blocks  5171 blocks (5.00%) reserved for the super user  First data block=1  Maximum filesystem blocks=67371008  13 block groups  8192 blocks per group, 8192 fragments per group  1992 inodes per group  Superblock backups stored on blocks:  8193, 24577, 40961, 57345, 73729  Writing inode tables: done  Creating journal (4096 blocks): done  Writing superblocks and filesystem accounting information: done  This filesystem will be automatically checked every 37 mounts or  180 days, whichever comes first. Use tune2fs -c or -i to override.  [root@sunwuce ~]# tune2fs -c 0 -i 0 /dev/sdc1  tune2fs 1.41.12 (17-May-2010)  Setting maximal mount count to -1  Setting interval between checks to 0 seconds  6.挂载磁盘  Mount  7.生效  7.1 mount 磁盘写入 etc/fstab  7.1 vim etc/rc.local | |
| 实  习  总 结 | 磁盘分区是将一个物理磁盘分成多个逻辑区域，以便不同的文件系统使用和管理。在shell脚本中，可以使用fdisk、parted和sfdisk等命令进行磁盘分区。  fdisk命令是一个常用的磁盘分区工具，它可以显示、创建、修改和删除磁盘分区。使用fdisk命令时，需要首先选择要分区的磁盘，然后输入分区类型、分区大小等信息。分区完成后，需要使用mkfs命令对分区进行格式化，以便文件系统可以使用。  parted命令是另一个常用的磁盘分区工具，它可以创建、删除、移动和调整磁盘分区。parted命令可以支持更多的文件系统类型，并且可以在无需重启计算机的情况下重新分配分区。  sfdisk命令是一个用于管理磁盘分区表的命令，它可以显示、创建、修改和删除磁盘分区表。sfdisk命令可以将分区表导出到文件中，并且可以在其他计算机上重新导入分区表。  在shell脚本中，可以使用这些命令来创建、修改和删除磁盘分区，以便在不同的文件系统中使用。同时，还可以使用df命令来显示已经挂载的分区和其使用情况，以便进行磁盘空间管理。 | |

|  |  |  |
| --- | --- | --- |
| **学习报告** | | |
| 学习日期（第8天） | | 2023年2月 28 日 |
| 目的 | Swap | |
| 内容 | **如何增加swap**  **方法一**  [root@sunwuce ~]# dd if=/dev/zero of=/tmp/100m bs=1M count=100  100+0 records in  100+0 records out  104857600 bytes (105 MB) copied, 0.549104 s, 191 MB/s  [root@sunwuce ~]# cat /tmp/100m  [root@sunwuce ~]# ls /tmp  100m yum.log  [root@sunwuce ~]# mkswap /tmp/100m  mkswap: /tmp/100m: warning: don't erase bootbits sectors  on whole disk. Use -f to force.  Setting up swapspace version 1, size = 102396 KiB  no label, UUID=ebacbf92-023a-4515-a379-9e44d533ec83  [root@sunwuce ~]# swapon /tmp/100m  [root@sunwuce ~]# df -h  Filesystem Size Used Avail Use% Mounted on  /dev/sda3 8.8G 1.3G 7.1G 16% /  tmpfs 491M 0 491M 0% /dev/shm  /dev/sda1 190M 35M 146M 19% /boot  [root@sunwuce ~]# cd ..  [root@sunwuce /]# df -h  Filesystem Size Used Avail Use% Mounted on  /dev/sda3 8.8G 1.3G 7.1G 16% /  tmpfs 491M 0 491M 0% /dev/shm  /dev/sda1 190M 35M 146M 19% /boot  [root@sunwuce /]# swapon -s  Filename Type Size Used Priority  /dev/sda2 partition 783356 0 -1  /tmp/100m file 102396 0 -2  [root@sunwuce /]# free -h  total used free shared buffers cached  Mem: 980M 305M 675M 224K 31M 149M  -/+ buffers/cache: 124M 855M  Swap: 864M 0B 864M  [root@sunwuce /]# swapoff /tmp/100m  [root@sunwuce /]# free -h  total used free shared buffers cached  Mem: 980M 305M 675M 224K 31M 149M  -/+ buffers/cache: 124M 855M  Swap: 764M 0B 764M  **用parted创建相当于增加swap的分区**  [root@sunwuce ~]# parted /dev/sdb  GNU Parted 2.1  Using /dev/sdb  Welcome to GNU Parted! Type 'help' to view a list of commands.  (parted) p  Error: /dev/sdb: unrecognised disk label  (parted) n  Error: /dev/sdb: unrecognised disk label  (parted) mklabel gpt  (parted) p  Model: VMware, VMware Virtual S (scsi)  Disk /dev/sdb: 107MB  Sector size (logical/physical): 512B/512B  Partition Table: gpt  Number Start End Size File system Name Flags  (parted) mkpart primary 0 10M  Warning: The resulting partition is not properly aligned for best performance.  Ignore/Cancel? i  (parted) p  Model: VMware, VMware Virtual S (scsi)  Disk /dev/sdb: 107MB  Sector size (logical/physical): 512B/512B  Partition Table: gpt  Number Start End Size File system Name Flags  1 17.4kB 10.0MB 9983kB primary  (parted) mkpart primary 10M 20M  (parted) p  Model: VMware, VMware Virtual S (scsi)  Disk /dev/sdb: 107MB  Sector size (logical/physical): 512B/512B  Partition Table: gpt  Number Start End Size File system Name Flags  1 17.4kB 10.0MB 9983kB primary  2 10.5MB 19.9MB 9437kB primary  (parted) q  Information: You may need to update /etc/fstab.  [root@sunwuce ~]# ll /dev/sdb\*  brw-rw----. 1 root disk 8, 16 Nov 16 15:40 /dev/sdb  brw-rw----. 1 root disk 8, 17 Nov 16 15:40 /dev/sdb1  brw-rw----. 1 root disk 8, 18 Nov 16 15:40 /dev/sdb2  **如何进入单用户模式**  C:\Users\Administrator\Desktop\QQ图片20171116160300.png  C:\Users\Administrator\Desktop\QQ图片20171116160309.png  C:\Users\Administrator\Desktop\QQ图片20171116160324.png  **Sed命令**  C:\Users\Administrator\Desktop\QQ图片20171116160328.png  C:\Users\Administrator\Desktop\QQ图片20171116160335.png | |
| 实  习  总 结 | Swap是一种虚拟内存技术，可将硬盘空间用作RAM的扩展。在Linux中，Swap分区是必需的，尤其是在内存不足的情况下，它可以防止系统宕机或崩溃。  sed是一个文本处理工具，可以在处理文件时实现文本的替换、删除、插入、修改等操作。它可以从标准输入读取数据，处理后输出到标准输出，也可以直接对文件进行操作。sed的工作原理是按行读取数据，对每一行进行处理并输出 | |

|  |  |  |
| --- | --- | --- |
| **学习报告** | | |
| 学习日期（第9天） | | 2023年3月 1 日 |
| 目的 | Shell爬虫程序编写 | |
| 内容 | url="https://www.ldxy.edu.cn/"  user\_agent="Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.3"  referer="https://www.baidu.com/"  curl -A "$user\_agent" -e "$referer" -s "$url" > page.html  grep -o '<img[^>]\*src="[^"]\*"' page.html | sed 's/<img[^>]\*src="//g' | sed 's/"$//' > images.txt  while read image\_url; do  # 生成随机休眠时间（1~3秒）  sleep\_time=$(( (RANDOM % 3) + 1 ))  echo "下载图片: $image\_url，等待 $sleep\_time 秒后继续..."  sleep "$sleep\_time"  wget -p /images -q "$image\_url"  done < images.txt  设置爬虫参数：设置要爬取的网站 URL，用户代理（User-Agent）以及来源（Referer）。  url="https://www.ldxy.edu.cn/"  user\_agent="Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.3"  referer="https://www.baidu.com/"  下载网页内容：使用 curl 命令下载网站的 HTML 内容，并将结果保存到本地的 page.html 文件中。-A 参数用于设置用户代理，-e 参数用于设置来源，-s 参数用于静默模式，即不输出下载过程中的信息。  curl -A "$user\_agent" -e "$referer" -s "$url" > page.html  提取所有图片链接：使用 grep 命令和正则表达式从 page.html 文件中提取所有的图片链接，并将结果保存到 images.txt 文件中。-o 参数用于仅输出匹配到的内容，不输出整行，这样可以方便后续的字符串处理。正则表达式 <img[^>]\*src="[^"]\*" 用于匹配所有以 <img 开头、包含 src 属性且以双引号结束的 HTML 标签。接着，使用 sed 命令去除所有匹配行中的 <img 和 " 符号，仅保留图片链接。  grep -o '<img[^>]\*src="[^"]\*"' page.html | sed 's/<img[^>]\*src="//g' | sed 's/"$//' > images.txt  下载所有图片：使用 while 循环遍历 images.txt 文件中的每个图片链接，并使用 wget 命令将图片下载到本地的 /images 目录中。-p 参数用于创建递归目录结构，-q 参数用于安静模式，即不输出下载过程中的信息。此外，为了防止下载过于频繁被网站封禁，每次下载前还会生成 1~3 秒的随机休眠时间。  while read image\_url; do  # 生成随机休眠时间（1~3秒）  sleep\_time=$(( (RANDOM % 3) + 1 ))  echo "下载图片: $image\_url，等待 $sleep\_time 秒后继续..."  sleep "$sleep\_time"  wget -p /images -q "$image\_url"  done < images.txt  清理临时文件：删除下载的网页内容文件 page.html 和提取出的图片链接文件 images.txt | |
| 实  习  总 结 | 这段代码是一个 Bash 脚本，主要用于爬取指定网站的所有图片，并将它们下载到本地的 /images 目录中。 | |

|  |  |  |
| --- | --- | --- |
| **学习报告** | | |
| 学习日期（第10天） | | 2023年3月 2日 |
| 目的 | Shell练习10题 | |
| 内容 | **1、编写hello world脚本**  #!/bin/bash  echo “hello world”  **2、通过位置变量创建 Linux 系统账户及密码**  #!/bin/bash  useradd “$1”  echo “$2” | passwd --stdin “$1”  # 通过位置变量创建 Linux 系统账户及密码    #$1 是执行脚本的第一个参数,$2 是执行脚本的第二个参数  **3、备份日志**  #!/bin/bash  # 每周 5 使用 tar 命令备份/var/log 下的所有日志文件  # vim /root/logbak.sh  # 编写备份脚本,备份后的文件名包含日期标签,防止后面的备份将前面的备份数据覆盖  # 注意 date 命令需要使用反引号括起来,反引号在键盘<tab>键上面  tar -czf log-`date +%y%m%d`.tar.gz /var/log  # crontab ‐e #编写计划任务,执行备份脚本  crontab -e  0 0 \* \* 5 sh /root/logbak.sh  **4、一键部署 LNMP(RPM 包版本)**  # 一键部署 LNMP(RPM 包版本)  # 使用 yum 安装部署 LNMP,需要提前配置好 yum 源,否则该脚本会失败  # 本脚本使用于 centos7.2 或 RHEL7.2  #!/bin/bash  yum ‐y install httpd  yum ‐y install mariadb mariadb‐devel mariadb‐server  yum ‐y install php php‐mysql    systemctl start httpd mariadb  systemctl enable httpd mariadb  **5、监控内存和磁盘容量，小于给定值时报警**    # 实时监控本机内存和硬盘剩余空间,剩余内存小于500M、根分区剩余空间小于1000M时,发送报警邮件给root管理员  # 提取根分区剩余空间  # 提取内存剩余空间  #!/bin/bash  disk\_size=$(df / | awk '///{print $4}')  mem\_size=$(free | awk '/Mem/{print $4}'  while true  do  if [ $disk\_size -le 512000 -a $mem\_size -le 1024000 ]; then  mail ‐s "Warning" root <<EOF  Insufficient resources,资源不足  EOF  fi  done  **6、猜数字游戏**  #!/bin/bash  rannum=$((${RANDOM}%100+1))  while true  do  read -p “输入一个100以内的数” inputnum  if [ ${inputnum} -gt ${rannum} ]; then  echo “猜大了”  elif [ ${inputnum} -lt ${rannum} ]; then  echo “猜小了”  else  echo “猜对了”  exit 0;  fi  done  # 脚本生成一个 100 以内的随机数,提示用户猜数字,根据用户的输入,提示用户猜对了,  # 猜小了或猜大了,直至用户猜对脚本结束。    # RANDOM 为系统自带的系统变量,值为 0‐32767的随机数  # 使用取余算法将随机数变为 1‐100 的随机数    # 使用 read 提示用户猜数字  # 使用 if 判断用户猜数字的大小关系:‐eq(等于),‐ne(不等于),‐gt(大于),‐ge(大于等于),  # ‐lt(小于),‐le(小于等于)  **7、检测本机当前用户是否为超级管理员,如果是管理员,则使用 yum 安装 vsftpd,如果不是,则提示您非管理员(使用字串对比版本)**  #!/bin/bash  if [ “$USER” == “root” ]; then  yum -y install vsftpd  else  echo “你不是超级管理员”  fi  # 检测本机当前用户是否为超级管理员,如果是管理员,则使用 yum 安装 vsftpd,如果不  # 是,则提示您非管理员(使用字串对比版本)  **8、检测本机当前用户是否为超级管理员,如果是管理员,则使用 yum 安装 vsftpd,如果不是,则提示您非管理员(使用 UID 数字对比版本)**  #!/bin/bash  if [ “$UID” == 0 ]; then  yum -y install vsftpd  else  echo “你不是超级管理员”  fi  # 检测本机当前用户是否为超级管理员,如果是管理员,则使用 yum 安装 vsftpd,如果不  # 是,则提示您非管理员(使用 UID 数字对比版本)  **9、编写脚本:提示用户输入用户名和密码,脚本自动创建相应的账户及配置密码。如果用户不输入账户名,则提示必须输入账户名并退出脚本;如果用户不输入密码,则统一使用默认的 123456 作为默认密码。**  #!/bin/bash  read -p “请输入用户名” name  if [ -z ${name} ];then  echo “必须输入用户名”  exit 2  else  stty -echo  read -p “请输入密码” PASSWD  stty echo  if [ -z ${PASSWD}]; then  useradd ${name}  echo $PASSWD | passwd --stdin $name  else  useradd ${name}  echo “123456” | passwd --stdin $name  fi  fi  # 编写脚本:提示用户输入用户名和密码,脚本自动创建相应的账户及配置密码。如果用户  # 不输入账户名,则提示必须输入账户名并退出脚本;如果用户不输入密码,则统一使用默  # 认的 123456 作为默认密码。  #使用‐z 可以判断一个变量是否为空,如果为空,提示用户必须输入账户名,并退出脚本,退出码为 2  #没有输入用户名脚本退出后,使用$?查看的返回码为 2  #使用 stty ‐echo 关闭 shell 的回显功能  #使用 stty echo 打开 shell 的回显功能  **10、输入三个数并进行升序排序**  #!/bin/bash  # 依次提示用户输入 3 个整数,脚本根据数字大小依次排序输出 3 个数字  read -p "请输入一个整数:" num1  read -p "请输入一个整数:" num2  read -p "请输入一个整数:" num3  if [ ${num1} -gt ${num2} ]; then  tmp=${num1}  num1=${num2}  num2=${tmp}  fi  if [ ${num1} -gt ${num3} ]; then  tmp=${num1}  num1=${num3}  num3=${tmp}  fi  if [ ${num2} -gt ${num3} ]; then  tmp=${num2}  num2=${num3}  num2=${tmp}  fi  echo "$num1,$num2,$num3"  # 不管谁大谁小,最后都打印 echo "$num1,$num2,$num3"  # num1 中永远存最小的值,num2 中永远存中间值,num3 永远存最大值  # 如果输入的不是这样的顺序,则改变数的存储顺序,如:可以将 num1 和 num2 的值对调 | |
| 实  习  总 结 | 本次是shell学习的最后一天了,我们完成了10道shell编程题  Shell脚本的基础语法：包括变量的声明和使用、条件语句、循环语句、函数定义和调用等。  常用的命令和工具：包括文件和目录操作、文本处理、系统管理等。比如，ls、cp、mv、rm、cat、grep、sed、awk、top、ps等。  Shell脚本的调试和错误处理：包括调试选项、错误处理、日志记录等。  Shell脚本的高级特性：包括正则表达式、数组、文件描述符、进程管理等。  Shell脚本的编写规范：包括脚本注释、代码缩进、函数命名等。  Shell脚本的实际应用：包括自动化脚本、系统管理脚本、数据处理脚本、网络脚本等。  在掌握了以上内容后，你可以编写各种Shell脚本，从而实现自动化、批处理、日志分析等任务，提高工作效率。同时，你也可以在Linux系统中进行更高效的管理和维护 | |