**Morning：**

1. 字符串截取及切割（不会影响原值）

${#arr} ‘查看$arr字符串的长度

1. 子串截取三种用法：

-${变量名:起始位置:长度} ，起始位从0开始

例如： echo $arr ‘显示abcdefg

echo ${arr:1:3} ‘显示bcd

echo ${arr::4} ‘显示abcd

1. 使用 expr substr

-expr substr "$变量名" 起始位置 长度 ，起始位从1开始

例如： expr substr “$arr” 1 6 ‘显示abcdef

1. 使用cut分割工具

-echo $变量名 | cut -b 起始位置-结束位置

关于位置： p1-p2代表连续；

-p2 代表从头截取到p2；

p1- 代表从p1截取到尾；

p1,p2,p3,p4代表不连续，按顺序显示；

例如： echo $arr | cut -b 3-5 ‘显示cde

echo $arr | cut -b 1,3,5 ‘显示ace

Tips：arr=$a$b$c ‘字符串合并

1. 字符串的替换（不会影响原值）
2. 只替换第1个字串

-${变量名/old/new}

例如： echo $arr ‘显示abcddcba

echo ${arr/d/3} ‘显示abc3dcba

1. 替换全部字串

-${变量名//old/new}

例如： echo ${arr//d/3} ‘显示abc33cba

1. 字符串的匹配删除（不会影响原值）

1）从左向右，最短匹配删除（掐头）

-${变量名#\*关键词}

例如： echo $arr ’显示root:x:0:0:root:/root:/bin/bash

echo ${arr#\*:} ‘显示x:0:0:root:/root:/bin/bash

2）从左向右，最长匹配删除

-${变量名##\*关键词}

例如： echo ${arr##\*:} ‘显示/bin/bash

1. 从右向左，最短匹配删除

-${变量名%关键词\*}

例如： echo ${arr%:\*} ‘显示root:x:0:0:root:/root

4）从右向左，最长匹配删除

-${变量名%%关键词\*}

例如： echo ${arr%%:\*} ‘显示root

1. 字符串初值的处理

-${var:-word}

例如：echo ${arr:-123} ‘若$arr有值，原值输出；否则初始化输出

Tips：不会自动赋值给变量，需要arr=${arr:-123}

**Afternoon：**

1. expect预期交互

1）mail的知识储备：

-mail -s test student #交互式，以.结尾

-echo “hello” | mail -s test student #非交互式，管道

-mail -s input root < /root/test.txt #从文件导入（有依赖）

-mail -s input root << EOF #EOF .... EOF（无依赖）

This is a test mail...

EOF

1. expect预期交互格式：

必须事先yum -y install expect

只能对spawn所指定的操作下进行预期交互

expect << EOF

spawn ssh -o StrictHostKeyChecking=no 172.25.0.10

expect "password" { send "redhat\n"} ‘关键字、命令

expect "#" { send "touch /123.txt\n"}

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

expect ‘默认不执行

EOF

Tips： 1）expect脚本的最后一行默认不执行

2）如果不希望ssh时出现yes/no的提示，远程时使用如下选项:

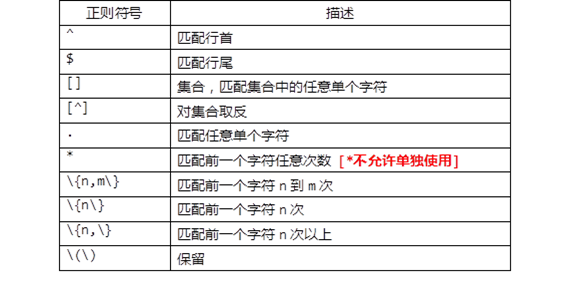
ssh -o StrictHostKeyChecking=no server0

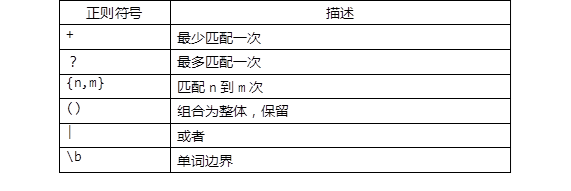
3）expect等待超时时间默认为10s，通过set timeout -1取消， 或者set timeout 20 设为20s

4）spawn 程序，下面的指令必须是建立在该进程的，否则不执行或者直接报错

1. 正则表达式

-过滤、匹配数据。通用表达式





基本正则：

1. grep -v(取反) -i(忽略大小写) ，^$代表空行
2. 使用正则表达式时最好使用””

3）[a-dm9]=[abcdm9]，单个字符匹配即可

4）[^abc]，对集合取反，只能把^放在前面

5）正则.匹配任意单个字符，相当于命令行?

6）aa\*，匹配前一个字符出现任意次数（包括0次）

7）.\*，相当于命令行的\*

扩展正则：

1）+/？是\*号的分支，对匹配次数作了控制

2）{n,m}与\{n,m\}是一样意思

3）egrep “(ab){1,3}” 1.txt 查找ababab

4）egrep “\bthe\b” 1.txt ‘查找the单词

egrep “the\b” 1.txt ‘the结尾

5）()组合并保留复制一份，应用于sed ‘s/(beijing)/ni hao \1/’ a.txt

Tips： 基本正则兼容性强，几乎所有软件都支持，书写麻烦

扩展正则兼容性较差，grep不支持扩展正则，书写简单

egrep，或者grep -E支持扩展