文本操作篇

极客时间

扫码试看/订阅极客时间《Linux实战技能100讲》视频课程

正则表达式与文本搜索

正则表达式与文本搜索

- 元字符
- 扩展元字符
- 文件的查找命令 find
- 文本内容的过滤（查找） grep

正则表达式的匹配方式

- 字符串 Do one thing at a time, and do well.
- 匹配字符 an

元字符

- . 匹配除换行符外的任意单个字符
- * 匹配任意一个跟在它前面的字符
- [ ] 匹配方括号中的字符类中的任意一个
- ^ 匹配开头
- $ 匹配结尾
- \ 转义后面的特殊字符

扩展元字符

- + 匹配前面的正则表达式至少出现一次
- ? 匹配前面的正则表达式出现零次或一次
- | 匹配它前面或后面的正则表达式

文件查找命令

- 文件查找命令 find
  - find 路径 查找条件 [补充条件]

文本内容的查找

- 文本内容查找命令 grep
  - grep 选项 文本文件1 [... 文本文件n]
  - 常用选项
    - -i 忽略大小写
    - -r 递归读取每一个目录下的所有文件

行编辑器 sed 和 AWK 介绍

行编辑器介绍

- Vim 和 sed、AWK 的区别
- sed 的基本用法演示
- AWK 的基本用法演示

Vim 和 sed、AWK 的区别

- 交互式与非交互式
- 文件操作模式与行操作模式

sed 基本用法

sed 一般用于对文本内容做替换

```
sed ‘/user1/s/user1/u1/’ /etc/passwd
```

AWK 基本用法

AWK 一般用于对文本内容进行统计、按需要的格式进行输出

cut　命令：　cut -d : -f 1 /etc/passwd

AWK 命令：　awk -F: '/wd$/{print $1}' /etc/passwd

sed 的替换命令

sed 的替换命令

- sed 的模式空间
- 替换命令 s

sed 的模式空间

sed 的基本工作方式是：

- 将文件以行为单位读取到内存（模式空间）
- 使用sed的每个脚本对该行进行操作
- 处理完成后输出该行

替换命令 s

sed 的替换命令 s：

- sed 's/old/new/' filename
- sed -e 's/old/new/' -e 's/old/new/' filename ...
- sed -i 's/old/new/' 's/old/new/' filename ...

使用正则表达式

带正则表达式的替换命令 s：

- sed ’s/正则表达式/new/’ filename
- sed -r ’s/扩展正则表达式/new/’ filename

sed 的替换命令加强版

sed 的替换命令加强版

- 全局替换
- 标志位
- 寻址
- 分组
- sed 脚本文件

全局替换

- s/old/new/g
  - g 为全局替换，用于替换所有出现的次数
  - / 如果和正则匹配的内容冲突可以使用其他符号，如：
    - s@old@new@g

标志位

s/old/new/标志位

- 数字，第几次出现才进行替换
- g，每次出现都进行替换
- p 打印模式空间的内容
  - sed -n ‘script’ filename 阻止默认输出
- w file 将模式空间的内容写入到文件

寻址

默认对每行进行操作，增加寻址后对匹配的行进行操作

- /正则表达式/s/old/new/g
- 行号s/old/new/g
  - 行号可以是具体的行，也可以是最后一行 $ 符号
- 可以使用两个寻址符号，也可以混合使用行号和正则地址

分组

- 寻址可以匹配多条命令
- /regular/ { s/old/new/ ; s/old/new/ }

脚本文件

- 可以将选项保存为文件，使用-f 加载脚本文件
- sed -f sedscript filename

sed 的其他命令

sed 的其他命令

- 删除命令
- 追加、插入、更改
- 打印
- 下一行
- 读文件和写文件
- 退出命令

删除命令

- [寻址]d
  - 删除模式空间内容，改变脚本的控制流，读取新的输入行

追加插入和更改

- 追加命令 a
- 插入命令 i
- 更改命令 c

打印

- 打印命令 p

下一行

- 下一行命令 n
- 打印行号命令 =

读文件和写文件

- 读文件命令 r
- 写文件命令 w

退出命令

- 退出命令 q
- 哪个效率会更高呢？
  - sed 10q filename
  - sed -n 1,10p filename

sed 的多行模式

sed 的多行模式

- 为什要有多行模式
- 多行模式处理命令 N、D、 P

为什么要有多行模式

- 配置文件一般为单行出现
- 也有使用 XML 或 JSON 格式的配置文件，为多行出现

多行匹配命令

- N 将下一行加入到模式空间
- D 删除模式空间中的第一个字符到第一个换行符
- P 打印模式空间中的第一个字符到第一个换行符

sed 的保持空间

sed 的保持空间

- 什么是保持空间
- 保持空间命令

什么是保持空间

- 保持空间也是多行的一种操作方式
- 将内容暂存在保持空间，便于做多行处理

保持空间命令

- h 和 H 将模式空间内容存放到保持空间
- g 和 G 将保持空间内容取出到模式空间
- x 交换模式空间和保持空间内容

AWK

AWK

- AWK 和 sed 的区别
- AWK 脚本的流程控制

AWK 和 sed 的区别

- AWK 更像是脚本语言
- AWK 用于“比较规范”的文本处理，用于统计数量并输出指定字段
- 使用 sed 将不规范的文本，处理为“比较规范”的文本

AWK 脚本的流程控制

- 输入数据前例程 BEGIN{ }
- 主输入循环{ }
- 所有文件读取完成例程 END{ }

AWK 的字段引用和分离

AWK 的字段引用和分离

- 记录和字段
- 字段的引用

记录和字段

- 每行称作 AWK 的记录
- 使用空格、制表符分隔开的单词称作字段
- 可以自己指定分隔的字段

字段的引用

- awk 中使用 $1 $2 ... $n 表示每一个字段
  - awk ‘{ print $1, $2, $3}’ filename
- awk 可以使用 -F 选项改变字段分隔符
  - awk -F ‘,’ ‘{ print $1, $2, $3}’ filename
  - 分隔符可以使用正则表达式

AWK 的表达式

AWK 的表达式

- 赋值操作符
- 算数操作符
- 系统变量
- 关系操作符
- 布尔操作符

赋值操作符

- = 是最常用的赋值操作符
  - var1 = “name”
  - var2 = “hello" “world”
  - var3 = $1
- 其他赋值操作符
  - ++ - - += -= *= /= %= ^=

算数操作符

- 算数操作符
  - + - * / % ^

系统变量

- FS 和 OFS 字段分隔符，OFS 表示输出的字段分隔符
- RS 记录分隔符
- NR 和 FNR 行数
- NF 字段数量，最后一个字段内容可以用 $NF 取出

关系操作符

- 关系操作符
  - < > <= >= == != ~ !~

布尔操作符

- 布尔操作符
  - && || !

AWK 的条件和循环

AWK 的条件和循环

- 条件语句
- 循环

条件语句

- 条件语句使用 if 开头，根据表达式的结果来判断执行哪条语句

if (表达式)

awk语句1

[else

awk语句2

]

- 如果有多个语句需要执行可以使用 { } 将多个语句括起来

循环

- while 循环

  while(表达式)

  awk语句1

- do 循环

  do{

  awk 语句1

  }while(表达式)

循环

- for循环

  for(初始值；循环判断条件；累加)

    awk语句1

- 影响控制的其他语句
  - break
  - continue

AWK 的数组

AWK 的数组

- 数组的定义
- 数组的遍历
- 删除数组
- 命令行参数数组

数组的定义

- 数组： 一组有某种关联的数据（变量），通过下标依次访问
  - 数组名[下标] = 值
  - 下标可以使用数字也可以使用字符串

数组的遍历

- for(变量 in 数组名)
  - 使用 数组名 [变量] 的方式依次对每个数组的元素进行操作

删除数组

- 删除数组
  - delete 数组[下标]

命令行参数数组

- 命令行参数数组
  - ARGC
  - ARGV

AWK 的函数

AWK 的函数

- 算数函数
- 字符串函数
- 自定义函数

算数函数

- sin( ) cos( )
- int( )
- rand( ) srand( )

字符串函数

- gsub(r, s, t)
- index(s, t)
- length(s)
- match(s, r)
- split(s, a, sep)
- sub(r, s, t)
- substr(s, p, n)

自定义函数

```
function  函数名（ 参数 ）{
    awk语句
    return awk变量
}
```

极客时间

扫码试看/订阅极客时间《Linux实战技能100讲》视频课程