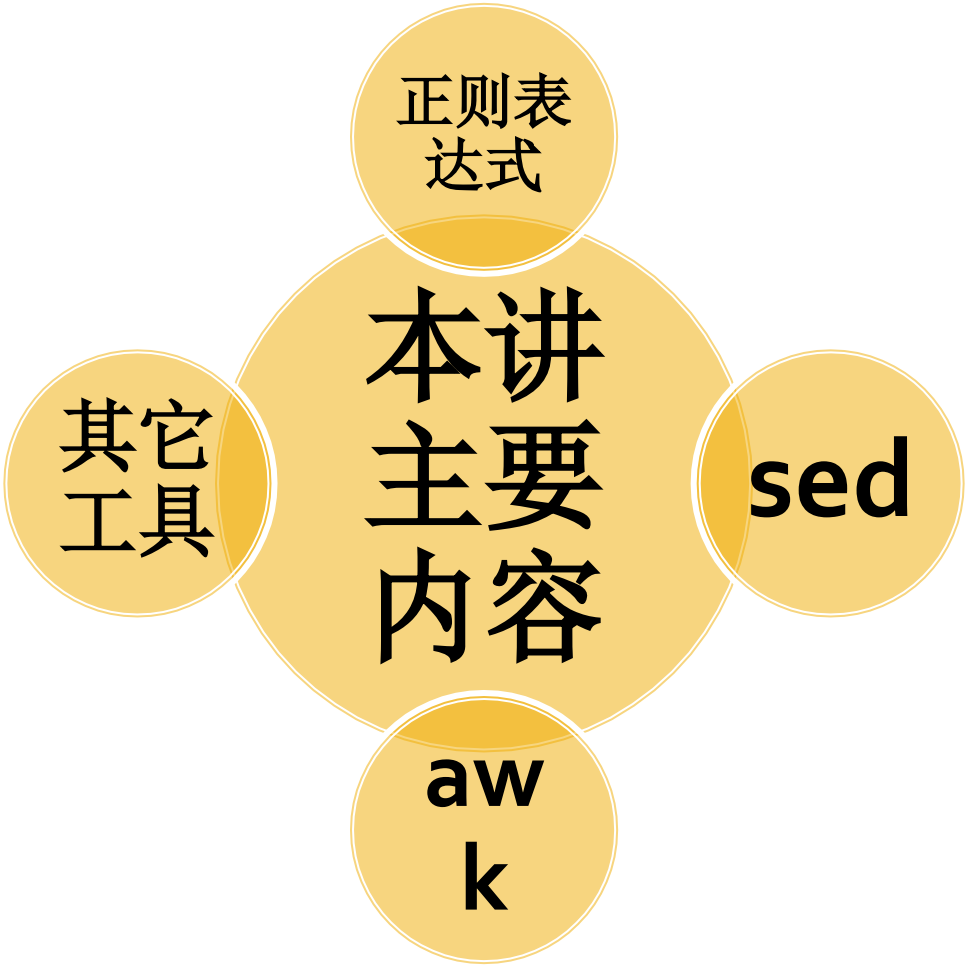
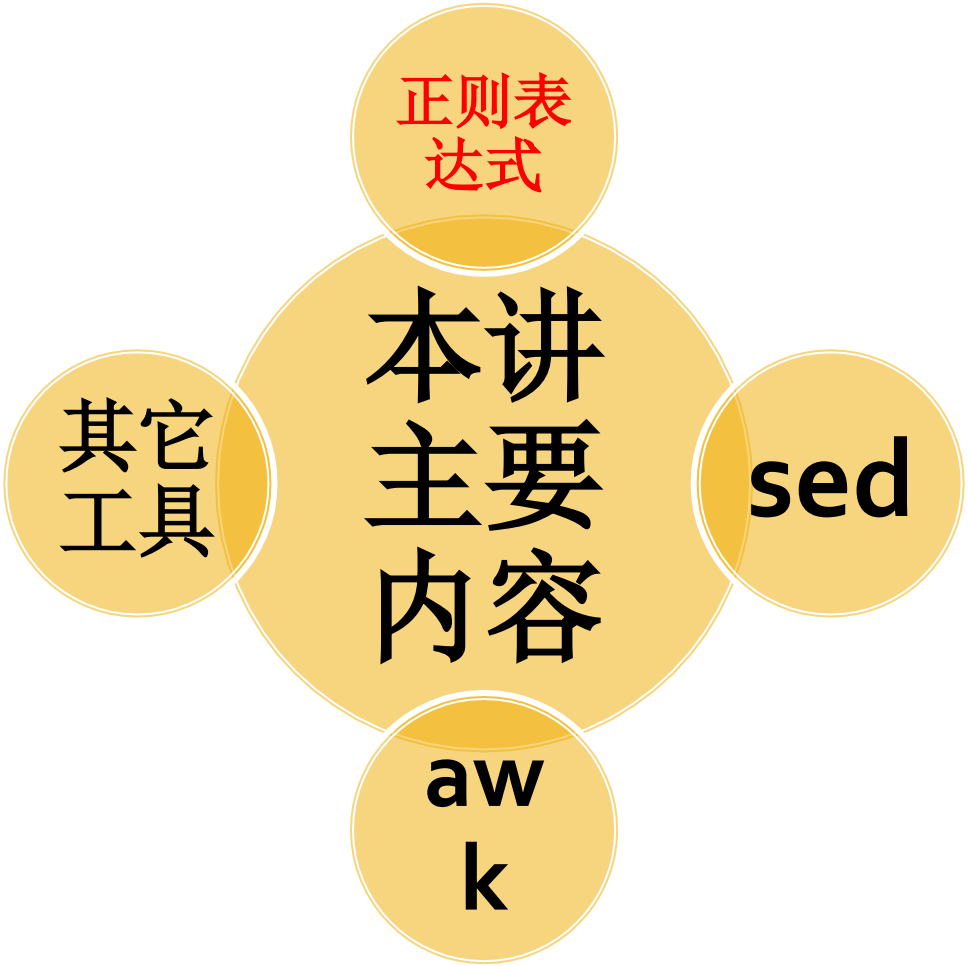
王晓庆
wxqfree@163.com

网络操作系统

第05讲 正则表达式与文本处理



第05讲 正则表达式与文本处理



什么是正则表达式？

- 是一种用来描述文本模式的特殊语法
- 由若干普通字符和/或元字符组成
- 通常用于查找符合特定模式的文本

- 支持正则表达式的常用工具
  - grep、sed、awk、vi、...

简单表达式

- 简单表达式：不包含元字符的正则表达式
  - grep bash /etc/passwd
- 当正则表达式中包含空白符时，需要用引号(单引号或双引号)括起来
  - grep 'lo inet' /etc/network/interfaces

基本正则表达式元字符

- .
- []、[^]
- *
- ^
- $

匹配除换行符外的单个任意字符

- **$ grep dev/.d /etc/fstab**
  - 注意: 其功能与文件名扩展中的?字符相同
- **$ grep dev\.d /etc/fstab**

[], [^]

- 匹配或不匹配其中包含的任意单个字符
  - **$ grep dev/[cf]d /etc/fstab**
  - **$ grep dev/[^cf]d /etc/fstab**
  - **$ grep dev/[.]d /etc/fstab**

*

- 匹配前一正则表达式项的o次或多次出现
  - cp /usr/share/doc/ed/copyrigh .
  - grep ‘t*y’ copyright
  - grep ‘tt*y’ copyright
  - grep tt*y copyright
  - touch tty
  - grep tt*y copyright
  - echo grep tt*y copyright

^

行首

- grep ‘^#’ /etc/login.defs
- grep -v ‘^#’ /etc/login.defs
- grep ‘^[^#]’ /etc/login.defs

$

- 行尾
  - grep 'yes$' /etc/login.defs
  - grep '[0-9][0-9]*$' /etc/login.defs
  - grep -v '^$' copyright
  - grep '..*' copyright

练习

- 找出copyright中以空格开头的行
- 找出copyright中以句点结尾的行
- 找出/usr/include/stdio.h中以tab键开头的行
- 找出/etc/adduser.conf中以#开头以.结尾的行

扩展正则表达式

- 扩展正则表达式(ERE)元字符
  - ? + | { } ( )
- 以上字符在基本正则表达式(BRE)中是普通字符，但前面加上\后就能成为元字符
  - grep 'BRE' files
  - grep -E 'ERE' files
  - egrep 'ERE' files

扩展正则表达式元字符

+	重复前项1次以上
?	重复前项0次或1次
s1|s2	s1或s2
()	组合
x{m}	x重复m次
x{m,}	x重复m次以上
x{m,n}	x重复m至n次

字符类(Character Class)

- 为配合非英语的环境，POSIX标准强化其字符集范围的能力，例如[a-z]，以匹配非英文字母字符。
  - [:alnum:] 数字字母
  - [:alpha:] 字母字符
  - [:blank:] 空格与tab字符
  - [:cntrl:] 控制字符
  - [:digit:] 数字字符
  - [:graph:] 非空格字符
  - [:lower:] 小写字母字符
  - [:print:] 可显示字符
  - [:punct:] 标点符号字符
  - [:space:] 空白字符
  - [:upper:] 大写字母字符
  - [:xdigit:] 十六进制数字
- 使用时必须使用方括号表达式，如[[:alnum:]]

反斜杠字符

- 单词：由字母、数字和下划线构成的串
  - \w 匹配单词,即[[:alnum:]]的同义词
  - \W 匹配非单词,即[^[:alnum:]]的同义词
  - \< 匹配单词的开头
  - \> 匹配单词的结尾
  - \b 匹配单词边缘的空字符串
  - \B 匹配非单词边缘的空字符串

后向引用(\n)

- 其中n为1,2,…,9，引用前面的第n个被圆括号括起来的子表达式
  - egrep '\<(.)(.).\2\1\>' /etc/bash_completion
  - grep -E '\<(.)(.).\2\1\>' /etc/bash_completion
  - grep '\<\(.\)\(.\).\2\1\>' /etc/bash_completion

练习

- 查找/etc/profile中包含被单引号或双引号括起来的字符串的行
  - 提示：使用后向引用

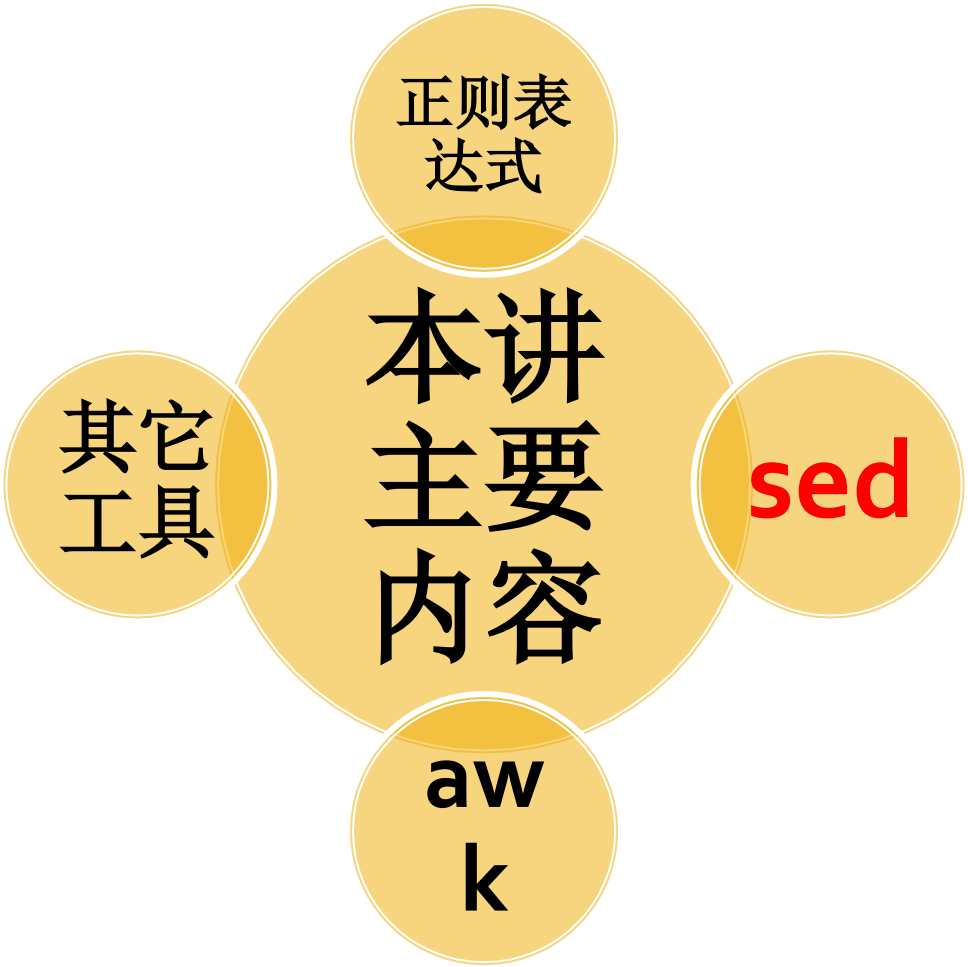
fgrep（fast grep）

- 匹配固定字符串而非正则表达式，它使用优化的算法，能更有效地批评古典字符串，也是唯一可以并行匹配多个字符串的版本。
  - $ fgrep 'www.gnu.org
  - > GPL-3
  - > ftp.gnu.org' copyright

find的正则表达式选项

- -regex：匹配某一模式的文件名
- -iregex：忽略模式中的大小写
  - find要求指定的正则表达式与整个路径匹配
- 例
  - $ find /bin -iregex 'ch'
  - $ find /bin -iregex '.*ch.*

第05讲 正则表达式与文本处理



sed（stream editor)

- grep只能搜索字符串
- sed则能在搜索基础上进一步编辑字符串
  - sed的设计就是用来以批处理的方式而不是交互的方式来编辑文件
  - 功能类似于查找/替换，却更强大得多

sed的工作原理

- sed读取每个文件，一次读一行，将读取的行放到一个内存缓冲区(称为pattern space)，所有编辑命令都会应用到模式空间中的内容，当所有操作完成后，sed会将模式空间的最后内容打印输出到标准输出，再回到开始处，读取下一行。

语法

- sed [-n] 'command' [file ...]
  - -n：sed默认在读入下一行前会自动打印模式空间内容至标准输出，此选项可以关闭该行为
- sed [-n] -e 'command' ... [file ...]
  - -e：执行多条sed编辑命令时使用，如：sed -e 'cmd1' -e 'cmd2' file
- sed [-n] -f script-file [file ...]
  - -f：从script-file中读取编辑命令，当有较多命令需要执行时，可以使用此选项

引例

- 把用户sam主目录内的目录结构复制给tom，但不复制其中的文件。
  - $ sudo su
  - $ find /home/sam/* -type d -print |
    > sed 's./home/sam/./home/tom/.' |
    > sed 's/^/mkdir /' |
    > sh

基本用法

- sed '' /etc/deluser.conf
- sed 'p' /etc/deluser.conf
- sed -n 'p' /etc/deluser.conf
- sed 'd' /etc/deluser.conf
- sed 'q' /etc/deluser.conf
- sed -n -e 'p' -e 'q' /etc/deluser.conf
- sed -n 'p;q' /etc/deluser.conf
  - 分号可用于分隔多个命令
- sed '3q' letter /etc/deluser.conf

地址

- n　　第n行
- $　　最后一行
- m,n　　从第m行到第n行(m<n)
- /pattern/　　包含模式pattern的行
- \cpatternc　　包含模式pattern的行
  - c为任意字符
- n,/pattern/
- /pattern1/,/pattern2/
- !　反向选择
  - 如1,3!表示除1-3行外

命令

- p　打印行
- d　删除行
- q　退出
- =　打印匹配行的行号
- l　打印行中所有字符
- s　查找替换
- n　读下一行
- r　读文件
- w　写文件
- a\str 在行后追加str
- i\str 在行前插入str
- c\str 用str替换行

n 读入下一行

- 打印奇数行
  - cat -n copyright | sed -n 'p;n'
- 打印偶数行
  - cat -n copyright | sed -n 'n;p`

读写文件

- r
  - sed '$r /etc/motd' /etc/issue
- w
  - sed '/mingetty/w url' copyright
  - sed -n '/:\/\//w urls' copyright
  - sed -n '\.://.w urls2' copyright

查找替换

- echo ‘sam reads well. sam writes well.’ > example.txt
- sed 's/sam/tom/' example.txt
- sed 's/sam/tom/g' example.txt
- sed 's/sam/tom/2' example.txt
- echo ‘this cost 23, and that cost 35’ >numbers
- sed ‘s/[0-9][0-9]*/\$&/g’ numbers
  - 练习：若把23改成.23，把35改成3.5，怎么办？

a\ i\ c\

- a\ 在指定行后面追加若干行文本,并显示添加的新内容,主要用于sed脚本中
  - $ sed '/^Manual/a\
    newline1\
    newline2\
    newline3' copyright
- i\ 在指定行前追加若干行,并显示添加的新内容,使用格式同a\
- c\ 用若干行替换指定的行，使用格式同a\

练习

- 去除标点符号
- 删除行首空格和多余空格
- 文本缩排
- 实现basename命令
- 实现dirname
- 去除html标记
- 实现unix与dos文本格式转换

N命令

- N -- Next line
  - 读入下一行追加至模式空间末尾（用换行符分隔)，模式匹配可以跨越该换行符进行.如果输入中已经没有新行，则sed将直接退出,其后的命令也不会被执行
  - sed 'N;s/\n/ /' /etc/group
  - 练习
    - 用sed实现cat -n copyright命令

D命令

- D -- Delete first part of the pattern space
  - 删除模式空间中第一个换行符之前(包括该换行符)的字符，如果模式空间中还有内容，则重新从编辑命令列表起始处开始执行(不读入新行)，否则输入下一行，并再次从编辑命令列表的起始处开始执行
  - 如果模式空间中无换行符，则D与d作用相同
  - cat -n copyright | sed 'N;D‘
  - cat -n copyright | sed 'N;N;D'

P命令

- P -- Print first part of the pattern space
  - 打印模式空间中第一个换行符之前(包括该换行符)的字符
  - 如果模式空间中无换行符，则P与p作用相同
- cat -n copyright | sed 'N;P'

sed的缓存空间

- sed运行时维持两个数据缓存空间：
  - 模式空间(pattern space)
  - 暂存空间(hold space)
- 一般情况下,sed每次从输入流读入一行到模式空间,对行的编辑就在模式空间中进行.
- 暂存空间初始为空,但可以通过命令在模式空间和暂存空间之间转移数据.