马哥教育
最专业的Linux培训机构

文本处理工具

http://www.magedu.com

本章内容

- 各种文本工具来查看、分析、统计文本文件
- grep
- 正则表达式
- 扩展正则表达式
- sed

抽取文本的工具

- 文件内容:less和 cat
- 文件截取：head和tail
- 按列抽取：cut
- 按关键字抽取：grep

文件查看

- 文件查看命令：cat, tac,rev

  cat [OPTION]... [FILE]...

  -E：显示行结束符$

  -n：对显示出的每一行进行编号

  -A：显示所有控制符

  -b：非空行编号

  -s：压缩连续的空行成一行

分页查看文件内容

- more：分页查看文件

  more [OPTIONS...] FILE...

  -d：显示翻页及退出提示

- less：一页一页地查看文件或STDIN输出

  查看时有用的命令包括：

  /文本 搜索 文本

  n/N 跳到下一个 or 上一个匹配

  less 命令是man命令使用的分页器

显示文本前或后行内容

- head

  head [OPTION]... [FILE]...

  -c #: 指定获取前#字节

  -n #: 指定获取前#行

  -#: 指定行数

- tail

  tail [OPTION]... [FILE]...

  -c #: 指定获取后#字节

  -n #: 指定获取后#行

  -#:

  -f: 跟踪显示文件新追加的内容,常用日志监控

按列抽取文本cut和合并文件paste

- cut [OPTION]... [FILE]...
  - -d DELIMITER：指明分隔符，默认tab
  - -f FILEDS:
    - #：第#个字段
    - #,#[,#]：离散的多个字段，例如1,3,6
    - #-#：连续的多个字段，例如1-6
    - 混合使用：1-3,7
  - -c 按字符切割
  - --output-delimiter=STRING指定输出分隔符

cut和paste

- 显示文件或STDIN数据的指定列

```
cut -d: -f1 /etc/passwd
cat  /etc/passwd | cut -d: -f7
cut -c2-5 /usr/share/dict/words
```

- paste 合并两个文件同行号的列到一行

```
paste [OPTION]... [FILE]...
```

  -d 分隔符:指定分隔符，默认用TAB

  -s ：所有行合成一行显示

```
paste f1 f2
paste  -s f1 f2
```

分析文本的工具

- 文本数据统计：wc
- 整理文本：sort
- 比较文件：diff和patch

收集文本统计数据wc

- 计数单词总数、行总数、字节总数和字符总数
- 可以对文件或STDIN中的数据运行

```
$ wc story.txt
39      237     1901 story.txt
```

行数　字数　字符数

- 使用　-l　来只计数行数
- 使用　-w　来只计数单词总数
- 使用　-c　来只计数字节总数
- 使用　-m　来只计数字符总数

文本排序sort

- 把整理过的文本显示在STDOUT，不改变原始文件

  $ sort [options] file(s)

- 常用选项
  - -r　执行反方向（由上至下）整理
  - -n　执行按数字大小整理
  - -f　选项忽略（fold）字符串中的字符大小写
  - -u　选项（独特，unique）删除输出中的重复行
  - -t　c　选项使用c做为字段界定符
  - -k　X　选项按照使用c字符分隔的X列来整理能够使用多次

uniq

- uniq命令：从输入中删除重复的前后相接的行
- uniq [OPTION]... [FILE]...

  -c：显示每行重复出现的次数；

  -d：仅显示重复过的行；

  -u：仅显示不曾重复的行；

  连续且完全相同方为重复
- 常和sort 命令一起配合使用：

  sort userlist.txt | uniq -c

比较文件

- 比较两个文件之间的区别

```
$ diff foo.conf-broken foo.conf-works
5c5
<    use_widgets = no
---
>    use_widgets = yes
```

  - 注明第5行有区别（改变）

复制对文件改变patch

- diff 命令的输出被保存在一种叫做“补丁”的文件中
  - 使用 -u 选项来输出“统一的（unified）”diff格式文件，最适用于补丁文件。
- patch 命令复制在其它文件中进行的改变（要谨慎使用！）
  - 适用 -b 选项来自动备份改变了的文件

```
$ diff -u foo.conf-broken foo.conf-works > foo.patch
$ patch -b foo.conf-broken foo.patch
```

练习

- 1、找出ifconfig命令结果中本机的所有IPv4地址
- 2、查出分区空间使用率的最大百分比值
- 3、查出用户UID最大值的用户名、UID及shell类型
- 4、查出/tmp的权限，以数字方式显示
- 5、统计当前连接本机的每个远程主机IP的连接数，并按从大到小排序

Linux上文本处理三剑客

- grep：文本过滤(模式：pattern)工具；
  grep，egrep，fgrep（不支持正则表达式搜索）
- sed：stream editor，文本编辑工具；
- awk：Linux上的实现gawk，文本报告生成器；

grep

- grep: Global search REgular expression and Print out the line.

作用：文本搜索工具，根据用户指定的“模式”对目标文本逐行进行匹配检查；打印匹配到的行；

模式：由正则表达式字符及文本字符所编写的过滤条件

- grep [OPTIONS] PATTERN [FILE...]

```
grep root /etc/passwd
grep "$USER"  /etc/passwd
grep '$USER'  /etc/passwd
grep `whoami` /etc/passwd
```

grep命令选项

- --color=auto：对匹配到的文本着色显示；
- -v：显示不能够被pattern匹配到的行；
- -i：忽略字符大小写
- -n：显示匹配的行号
- -c：统计匹配的行数
- -o：仅显示匹配到的字符串；
- -q：静默模式，不输出任何信息
- -A #：after，后#行
- -B #：before，前#行
- -C #：context，前后各#行
- -e：实现多个选项间的逻辑or关系

  grep -e 'cat ' -e 'dog' file
- -w：整行匹配整个单词
- -E：使用ERE

正则表达式

- REGEXP：由一类特殊字符及文本字符所编写的模式，其中有些字符（元字符）不表示字符字面意义，而表示控制或通配的功能
- 程序支持：grep，vim，less,nginx等
- 分两类：
  - 基本正则表达式：BRE
  - 扩展正则表达式：ERE
    - grep -E，egrep
- 正则表达式引擎：
  - 采用不同算法，检查处理正则表达式的软件模块
  - PCRE（Perl Compatible Regular Expressions）
- 元字符分类：字符匹配、匹配次数、位置锚定、分组
- man 7 regex

基本正则表达式元字符

- 字符匹配：

  . ： 匹配任意单个字符；

  [] ： 匹配指定范围内的任意单个字符

  [^] ：匹配指定范围外的任意单个字符

  [:digit:]、[:lower:]、[:upper:]、[:alpha:]、[:alnum:]、[:punct:]、[:space:]

正则表达式

- 匹配次数：用在要指定次数的字符后面，用于指定前面的字符要出现的次数
  - *：匹配前面的字符任意次，包括0次
    - 贪婪模式：尽可能长的匹配
  - .*：任意长度的任意字符
  - \?：匹配其前面的字符0或1次
  - \+：匹配其前面的字符至少1次
  - \{m\}：匹配前面的字符m次
  - \{m,n\}：匹配前面的字符至少m次，至多n次
  - \{,n\}：匹配前面的字符至多n次
  - \{m,\}：匹配前面的字符至少m次

正则表达式

- 位置锚定：定位出现的位置
  - ^：行首锚定，用于模式的最左侧
  - $：行尾锚定，用于模式的最右侧
  - ^PATTERN$：用于模式匹配整行
    - ^$：空行
    - ^[[:space:]]*$ ：空白行
  - \< 或 \b：词首锚定，用于单词模式的左侧
  - \> 或 \b：词尾锚定，用于单词模式的右侧
  - \<PATTERN\>：匹配整个单词

正则表达式

- 分组：\(\)：将一个或多个字符捆绑在一起，当作一个整体进行处理，如：\(root\)\+

分组括号中的模式匹配到的内容会被正则表达式引擎记录于内部的变量中，这些变量的命名方式为：\1，\2，\3，...

\1：从左侧起，第一个左括号以及与之匹配右括号之间的模式所匹配到的字符；

实例：　\(string1\+\(string2\)*\)

\1：string1\+\(string2\)*

\2：string2

后向引用：引用前面的分组括号中的模式所匹配字符(而非模式本身)

正则表达式

元字符	定义
^	行首
$	行尾
.	任意单一字符
[]	[]内任意单一字符
[^]	除[]内任意单一字符
*	*前面字符重复不确定次数
\+	\+前面字符重复一次以上不确定次数
\?	？前面字符重复0或1次
\	转义符
.*	任意长度字符
\{n\}	前面字符重复n次
\{n,\}	前面字符重复n次以上
\{m,n\}	前面字符重复m次和n次之间

正则表达式

元字符	定义
[:alpha:]	所有字母，包括大、小写
[:alnum:]	所有字母和数字
[:upper:]	所有大写字母
[:lower:]	所有小写字母
[:digit:]	所有数字
[:punct:]	所有标点符号
[:space:]	空格和Tab

练习

- 1、显示/proc/meminfo文件中以大小s开头的行；(要求：使用两种方式)
- 2、显示/etc/passwd文件中不以/bin/bash结尾的行
- 3、显示用户rpc默认的shell程序
- 4、找出/etc/passwd中的两位或三位数
- 5、显示/etc/grub2.cfg文件中，至少以一个空白字符开头的且后面存非空白字符的行
- 6、找出"netstat -tan"命令的结果中以'LISTEN'后跟0、1或多个空白字符结尾的行
- 7、添加用户bash、testbash、basher以及nologin(其shell为/sbin/nologin),而后找出/etc/passwd文件中用户名同shell名的行

egrep及扩展的正则表达式

- egrep = grep -E
- egrep [OPTIONS] PATTERN [FILE...]
- 扩展正则表达式的元字符：
- 字符匹配：

  . 任意单个字符

  [] 指定范围的字符

  [^] 不在指定范围的字符

扩展正则表达式

- 次数匹配：
  - *：匹配前面字符任意次
  - ?：0或1次
  - +：1次或多次
  - {m}：匹配m次
  - {m,n}：至少m，至多n次

扩展正则表达式

- 位置锚定：
  - ^ :行首
  - $:行尾
  - \<, \b :语首
  - \>, \b :语尾
- 分组：
  - ()
  - 后向引用：\1, \2, ...
- 或者：
  - a|b
  - C|cat: C或cat
  - (C|c)at:Cat或cat

练习

- 1、显示当前系统root、mage或wang用户的UID和默认shell
- 2、找出/etc/rc.d/init.d/functions文件中行首为某单词(包括下划线)后面跟一个小括号的行
- 3、使用egrep取出/etc/rc.d/init.d/functions中其基名
- 4、使用egrep取出上面路径的目录名
- 5、统计以root身份登录的每个远程主机IP地址的登录次数
- 6、利用扩展正则表达式分别表示0-9、10-99、100-199、200-249、250-255
- 7、显示ifconfig命令结果中所有IPv4地址

处理文本的工具sed

- Stream EDitor，行编辑器
- sed是一种流编辑器，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

sed工具

- 用法：

  sed [option]... 'script' inputfile...

- 常用选项：

  -n：不输出模式空间内容的自动打印

  -e：多点编辑

  -f /PATH/TO/SCRIPT_FILE：从指定文件中读取编辑脚本

  -r：支持使用扩展正则表达式

  -i：原处编辑

- script：

  '地址命令'

sed工具

- 地址定界：
  - (1) 不给地址：对全文进行处理
  - (2) 单地址：
    - #：指定的行
    - /pattern/：被此处模式所能够匹配到的每一行
  - (3) 地址范围：
    - #,#
    - #,+#
    - /pat1/,/pat2/
    - #,/pat1/
  - (4) ~：步进
    - 1~2 奇数行
    - 2~2 偶数行

- 编辑命令：

  d：删除模式空间匹配的行

  p：显示模式空间中的内容

  a \text：在行后面追加文本；支持使用\n实现多行追加

  i \text：在行前面插入文本；支持使用\n实现多行插入

  c \text：替换行为单行或多行文本

  w /path/to/somefile：保存模式匹配的行至指定文件

  r /path/from/somefile：读取指定文件的文本至模式空间中匹配到的行后

  =：为模式空间中的行打印行号

  !:模式空间中匹配行取反处理

sed工具

- s///：查找替换,支持使用其它分隔符，s@@@，s###
- 替换标记：
  - g：行内全局替换
  - p：显示替换成功的行
  - w /PATH/TO/SOMEFILE：将替换成功的行保存至文件中

sed示例

- sed '2p' /etc/passwd
- sed -n '2p' /etc/passwd
- sed -n '1,4p' /etc/passwd
- sed -n '/root/p' /etc/passwd
- sed -n '2,/root/p' /etc/passwd 从2行开始
- sed -n '/^$/=' file 显示空行行号
- sed -n -e '/^$/p' -e '/^$/=' file
- sed '/root/a\superman' /etc/passwd行后
- sed '/root/i\superman' /etc/passwd 行前
- sed '/root/c\superman' /etc/passwd 代替行

sed示例

- sed '/^$/d' file
- sed '1,10d' file
- nl /etc/passwd | sed '2,5d'
- nl /etc/passwd | sed '2a tea'
- sed 's/test/mytest/g' example
- sed -n's/root/&superman/p' /etc/passwd 单词后
- sed -n's/root/superman&/p' /etc/passwd 单词前
- sed -e 's/dog/cat/' -e 's/hi/lo/' pets
- sed -i.bak 's/dog/cat/g' pets

练习

- 1、删除/etc/grub2.conf文件中所有以空白开头的行行首的空白字符
- 2、删除/etc/fstab文件中所有以#开头，后面至少跟一个空白字符的行的行首的#和空白字符
- 3、在/root/install.log每一行行首增加#号
- 4、在/etc/fstab文件中不以#开头的行的行首增加#号
- 5、处理/etc/fstab路径,使用sed命令取出其目录名和基名
- 6、利用sed 取出ifconfig命令中本机的IPv4地址
- 7、统计centos安装光盘中Package目录下的所有rpm文件的以.分隔倒数第二个字段的重复次数

- 高级编辑命令：
  - h：把模式空间中的内容覆盖至保持空间中
  - H：把模式空间中的内容追加至保持空间中
  - g：从保持空间取出数据覆盖至模式空间
  - G：从保持空间取出内容追加至模式空间
  - x：把模式空间中的内容与保持空间中的内容进行互换
  - n：读取匹配到的行的下一行覆盖至模式空间
  - N：追加匹配到的行的下一行至模式空间
  - d：删除模式空间中的行
  - D：删除当前模式空间开端至\n的内容（不在传至标准输出），放弃之后的命令，但是对剩余模式空间重新执行sed

sed示例

- sed -n 'n;p' FILE
- sed '1!G;h;$!d' FILE
- sed '$!N;$!D' FILE
- sed '$!d' FILE
- sed 'G' FILE
- sed 'g' FILE
- sed '/^$/d;G' FILE
- sed 'n;d' FILE
- sed -n '1!G;h;$p' FILE

关于马哥教育

- 博客：http://mageedu.blog.51cto.com
- 主页：http://www.magedu.com
- QQ：1661815153，113228115
- QQ群：203585050，279599283

马哥教育
最专业的Linux培训机构

Thank You!