**关于本文档**

|  |  |
| --- | --- |
| 博客 | http://lizhenliang.blog.51cto.com |
| QQ技术群 | [323779636](http://shang.qq.com/wpa/qunwpa?idkey=6bc9fe9b2edfdb8cfcb7901ee3b3939dfac574701ae449abb5e1e0c5a2a1c2e3)（Python运维开发群） |
| 说明 | 本文档均为个人经验总结，转发请保留出处，抵制不道德行为。 |

本章节学习一些在编写Shell时的常用命令或工具及使用技巧。有人说Shell脚本是命令堆积的一个文件，按顺序去执行。还有人说想学好Shell脚本，要把Linux上各种常见的命令或工具掌握了，这些说法都没错。由于Shell语言本身在语法结构上比较简单，是面向过程编程，想实现复杂的功能有点强人所难！而且Shell本身又工作在Linux内核之上，在用户态调用Linux命令会很方面，所以大多数情况下我们都是依靠这些命令来完成脚本中的某些功能，比如文本处理、获取系统状态等等，然后通过Shell语法结构组织代码逻辑。不管是学Linux系统好还是写Shell脚本也好，有些命令都是必须要会的，以下是根据个人经验总结的一些常用的命令。

怎么更好的学习命令呢？

当然查看官方帮助文档了，可以通过man cmd、cmd --help、help cmd、info cmd等方式查看命令的使用。

## ls

功能：列出目录内容

常用选项：

-a  显示所有文件，包括隐藏的

-l  长格式列出信息

-i  显示文件inode号

-t  按修改时间排序

-r  按修改时间倒序排序

-h 打印易读大小单位

示例：

按修改时间排序：

# ls -t

按修改时间倒序排序：

# ls -rt

长格式列出：

# ls -lh

查看文件inode：

# ls -i file

## echo

功能：打印一行

常用选项：

-n  不加换行符

-e  解释转义符

示例：

解释换行符：

# echo -e "1\n2\n3"

1

2

3

## printf

功能：格式化打印数据。默认打印字符串不换行。

格式：printf format [arguments]

|  |  |
| --- | --- |
| **Format** | **描述** |
| %s | 一个字符串 |
| %d，%i | 一个小数 |
| %f | 一个浮点数 |
| %.ns | 输出字符串，n是输出几个字符 |
| %m.nf | 输出浮点数，m是输出整数位数，n是输出的小数位数 |
| %x | 不带正负号的十六进制，使用a至f表示10到15 |
| %X | 不带正负号的十六进制，使用A至F表示10至15 |
| %% | 输出单个% |
| %-5s | 左对齐，对参数每个字段左对齐,宽度为5 |
| %-4.2f | 左对齐，宽度为4，保留两位小数 |
| %5s | 右对齐，不加横线表示右对齐 |

一些常用的空白符：

\n  换行

\r  回车

\t  水平制表符

示例：

输出一个字符：

# printf "%.1s" abc

a

保留一个小数点：

# printf "%.1f" 1.333

1.3

输出换行：

# printf "%.1f\n" 1.333

1.3

格式化输出：

# printf "user: %s\tpass: %d\n" abc 123

user: abc pass: 1

左对齐宽度10：

# printf "%-10s %-10s %-10s\n" ID Name Number

ID         Name       Number

右对齐宽度10：

# printf "%10s %10s %10s\n" ID Name Number

        ID       Name     Number

每段对齐：

# printf "%10s\n" ID Name Number

        ID

      Name

    Number

# printf "%-10s\n" ID Name Number

ID

Name

Number

## cat

功能：连接文件和标准输出打印

常用选项：

-b  显示非空行行号

-n  显示所有行行号

-T  显示tab，用^I表示

-E  显示以$结尾

示例：

连接两个文件：

# cat a b

# cat << EOF

> 123

> abc

> EOF

123

abc

将eof标准输入作为cat标准输出再写到a.txt：

# cat > a.txt << eof

> 123

> abc

> eof

## 5. tac

功能：连接文件和倒序打印文件

常用选项：

示例：

倒序打印每一行：

# tac a.txt

## 6. rev

功能：反向打印文件的每一行

常用选项：

示例：

# echo "123" |rev

321

## 7. wc

功能：统计文件行数、字节、字符数

常用选项：

-c  打印文件字节数，一个英文字母1字节，一个汉字占2-4字节（根据编码）

-m  打印文件字符数，一个汉字占2个字符

-l  打印多少行

-L 打印最长行的长度，也可以统计字符串长度

示例：

统计文件多少行：

# wc -l file

统计字符串长度：

# echo "hello" |wc -L

5

## 8. cp

功能：复制文件或目录

常用选项：

-a  归档

-b  目标文件存在创建备份，备份文件是文件名跟~

-f  强制复制文件或目录

-r  递归复制目录

-p  保留原有文件或目录属性

-i  覆盖文件之前先询问用户

-u  当源文件比目的文件修改时间新时才复制

-v  显示复制信息

示例：

复制目录：

# cp -rf test /opt

## 9. mkdir

功能：创建目录

常用选项：

-p 递归创建目录

-v  显示创建过程

示例：

创建多级目录：

# mkdir /opt/test/abc

创建多个目录：

# mkdir {install,tmp}

创建连续目录：

# mkdir {a..c}

## 10. mv

功能：移动文件或重命名

常用选项：

-b  目标文件存在创建备份，备份文件是"文件名后跟~"

-u  当源文件比目的文件修改时间新时才移动

-v  显示移动信息

示例：

移动文件：

# mv a.txt /opt

重命名文件：

# mv a.txt b.txt

## 11. rename

功能：重命名文件，支持通配符

常用选项：

示例：批量命名文件

将foo1-foo9替换为foo01-foo09：

# rename foo foo0 foo?

将以.htm后缀的文件替换为.html：

# rename .htm .html \*.htm

## 12. dirname

功能：去除路径的最后一个名字

常用选项：

示例：

# dirname /usr/bin/

/usr

# dirname dir1/str dir2/str

dir1

dir2

# dirname stdio.h

.

## 13. basename

功能：打印路径的最后一个名字

常用选项：

-a  支持多个参数

-s  删除后面的后缀

示例：

# basename /usr/bin/sort

sort

# basename include/stdio.h .h

stdio

# basename -s .h include/stdio.h

stdio

# basename -a any/str1 any/str2

str1

str2

## 14. du

功能：估算文件磁盘空间使用

常用选项：

-h  易读格式显示（K，M，G）

-b  单位bytes显示

-k  单位KB显示

-m  单位MB显示

-s  只显示总大小

--max-depth=<目录层数>，超过层数的目录忽略

--exclude=file 排除文件或目录

--time  显示大小和创建时间

示例：

查看目录大小：

# du -sh /opt

排除目录某个文件：

# du -sh --exclude=test /opt

## 15. cut

功能：选取文件的每一行数据

常用选项：

-b  选中第几个字符

-c  选中多少个字符

-d  指定分隔符分字段，默认是空格

-f  显示选中字段

示例：

打印b字符：

# echo "abc" |cut -b "2"

b

截取abc字符：

# echo "abcdef" |cut -c 1-3

abc

以冒号分隔，显示第二个字段：

# echo "a:b:c" |cut -d: -f2

b

## 16. tr

功能：替换或删除字符

格式：Usage: tr [OPTION]... SET1 [SET2]

常用选项：

-c  替换SET1没有SET2的字符

-d  删除SET1中字符

-s  压缩SET1中重复的字符

-t  将SET1用SET2转换，默认

示例：

替换SET1没有SET2的字符：

# echo "aaabbbccc" | tr -c c 1

111111ccc

去重字符：

# echo "aaacccddd" | tr -s '[a-z]'

acd

删除字符：

# echo "aaabbbccc" | tr -d bbb

aaaccc

删除换行符：

# echo -e "a\nb\nc" | tr -d '\n'

abc

替换字符：

# echo "aaabbbccc" | tr '[a-z]' '[A-Z]'

AAABBBCCC

## 17. stat

功能：显示文件或文件的系统状态

常用选项：

-Z 显示selinux安全上下文

-f  显示文件系统状态

-c  指定格式输出内容

-t  以简洁的形式打印

示例：

显示文件信息：

# stat file

只显示文件修改时间：

# stat -c %y file

## 18. seq

功能：打印序列化数字

常用选项：

-f  使用printf样式格式

-s  指定换行符，默认是\n

-w  等宽，用0填充

示例：

数字序列：

# seq 3

1

2

3

带0的数字序列：

# seq -w 03

01

02

03

范围数字序列：

# seq 2 5

2

3

4

5

步长序列：

# seq 1 2 5   # 2是步长

1

3

5

以冒号分隔序列：

# seq -s "+" 5

1+2+3+4+5

等宽并在数字前面加字符串：

# seq -f "str%02g" 3  # %g是默认数字位数，02是数字不足2位时用0填充。

str01

str02

str03

## 19. shuf

功能：生成随机序列

常用选项：

-i  输出数字范围

-o  结果写入文件

示例：

输出范围随机数：

# seq 5 |shuf

2

1

5

4

3

# shuf -i 5-10

8

10

7

9

6

5

## 20. sort

功能：排序文本，默认对整列有效

常用选项：

-f  忽略字母大小写

-M  根据月份比较，比如JAN、DEC

-h  根据易读的单位大小比较，比如2K、1G

-g 按照常规数值排序

-n  根据字符串数值比较

-r  倒序排序

-k  位置1,位置2 根据关键字排序，在从第位置1开始，位置2结束

-t  指定分隔符

-u  去重重复行

-o  将结果写入文件

示例：

随机数字排序：

# seq 5 |shuf |sort

随机字母排序：

# printf "%c\n" {a..f} |shuf |sort

倒序排序：

# seq 5 |shuf |sort -r

分隔后的字段排序：

# cat /etc/passwd |sort -t : -k 3 -n

去重重复行：

# echo -e "1\n1\n2\n3\n3" |sort -u

大小单位排序：

# du -h |sort -k 1 -h -r

分隔后第一个字段的第二个字符排序：

# echo -e "fa:1\neb:2\ncc:3" |sort -t : -k 1.2

tab作为分隔符：

# sort -t $"\t"

file文件内容：

zhangsan 6 100

lisi 8 80

wangwu 7 90

zhaoliu 9 70

对file文件的第二列正序排序，再次基础再对第三列倒序排序（多列排序）：

~~# sort -k 2,2 -n -k 3,3 -nr file~~

# sort -k 2 -n -k 3 -nr file

zhaoliu 9 70

lisi 8 80

wangwu 7 90

zhangsan 6 100

对两个文件同时排序：

# sort file1 file2

## 21. uniq

功能：去除重复行,只会统计相邻的

常用选项：

-c  打印出现的次数

-d  只打印重复行

-u 只打印不重复行

-D  只打印重复行，并且把所有重复行打印出来

-f N 比较时跳过前N列

-i  忽略大小写

-s N 比较时跳过前N个字符

-w N 对每行第N个字符以后内容不做比较

示例：

测试文本如下：

# cat file

abc

cde

xyz

cde

xyz

abd

去重复行：

# sort file |uniq

abc

abd

cde

xyz

打印每行重复次数：

# sort file |uniq -c

1 abc

1 abd

2 cde

2 xyz

打印不重复行：

# sort file |uniq -u

abc

abd

打印重复行：

# sort file |uniq -d

cde

xyz

打印重复行并统计出现次数：

# sort file |uniq -d -c

2 cde

2 xyz

根据前几个字符去重：

# sort file |uniq -w 2

abc

cde

xyz

## 22. tee

功能：从标准输入读取写到标准输出和文件

常用选项：

-a  追加到文件

示例：

打印并追加到文件：

# echo 123 |tee -a a.log

## 23. join

功能：连接两个文件

常用选项：

-i  忽略大小写

-o  按照指定文件栏位显示

-t  使用字符作为输入和输出字段分隔符

示例：

# cat file1

1 a

2 b

3 c

# cat file2

1 x

2 y

3 z

将两个文件相同字段合并一列：

# join file1 file2

1 a x

2 b y

3 c z

打印file1第二列和file2第二列：

# join -o 1.2 2.2 file1 file2

a x

b y

c z

# join -t ':' -o 1.1 2.1 /etc/passwd /etc/shadow

user1:user1

……

## 24. paste

功能：合并文件

常用选项：

-d  指定分隔符，默认是tab键分隔

-s  将文件内容平行合并，默认tab键分隔

示例：

# seq 1 3 > file1

# seq 4 6 > file2

两个文件合并：

# paste file1 file2

1 4

2 5

3 6

两个文件合并，+号分隔：

# paste -d "+" file1 file2

1+4

2+5

3+6

文件内容平行显示：

# paste -s file1 file2

1 2 3

4 5 6

## 25. head

功能：输出文件的前几行

常用选项：

-c  打印前多少K，M

-n  打印前多少行

示例：

打印文件前50行：

# head -n 50 file

## 26. tail

功能：输出文件的后几行

常用选项：

-c  打印后多少K，M

-f  实时读文件，随着文件输出附加输出

-n  输出最后几行

--pid  与-f一起使用，表示pid死掉后结束

-s  与-f一起使用，表示休眠多少秒输出

示例：

打印文件后50行：

# tail -n 50 file

实时输出新增行：

# tail -f file

## 27. find

功能：搜索文件目录层次结构

格式：find path -option actions

常用选项：

-name  文件名，支持(‘\*’, ‘?’)

-type  文件类型，d目录，f常规文件等

-perm  符合权限的文件，比如755

-atime -/+n  在n天以内/过去n天被访问过

-ctime -/+n  在n天以内/过去n天被修改过

-amin -/+n   在n天以内/过去n分钟被访问过

-cmin -/+n   在n天以内/过去n分钟被修改过

-size -/+n   文件大小小于/大于，b、k、M、G

-maxdepth levels  目录层次显示的最大深度

-regex pattern   文件名匹配正则表达式模式

-inum 通过inode编号查找文件

动作：

-detele  删除文件

-exec command {} \;  执行命令，花括号代表当前文件

-ls   列出当前文件，ls -dils格式

-print 完整的文件名并添加一个回车换行符

-print0  打印完整的文件名并不添加一个回车换行符

-printf format  打印格式

其他字符：

！ 取反

-or/-o 逻辑或

-and 逻辑和

示例：

查找文件名：

# find / -name "\*http\*"

查找文件名并且文件类型：

# find /tmp -name core -type f -print

查找文件名并且文件类型删除：

# find /tmp -name core -type f -delete

查找当前目录常规文件并查看文件类型：

# find . -type f -exec file '{}' \;

查找文件权限是664：

# find . -perm 664

查找大于1024k的文件：

# find . -size -1024k

查找3天内修改的文件：

# find /bin -ctime -3

查找3分钟前修改的文件：

# find /bin -cmin +3

排除多个类型的文件：

# find . ! -name "\*.sql" ! -name "\*.txt"

或条件查找多个类型的文件：

# find . -name '\*.sh' -o -name '\*.bak'

# find . -regex ".\*\.sh\|.\*\.bak"

# find . -regex ".\*\.\(sh\|bak\)"

并且条件查找文件：

# find . -name "\*.sql" -a -size +1024k

只显示第一级目录：

# find /etc -type d -maxdepth 1

通过inode编号删除文件：

# rm `find . -inum 671915`

# find . -inum 8651577 -exec rm -i {} \;

## 28. xargs

功能：从标准输入执行命令

常用选项：

-a file  从指定文件读取数据作为标准输入

-0  处理包含空格的文件名,print0

-d  delimiter  分隔符，默认是空格分隔显示

-i  标准输入的结果以{}代替

-I  标准输入的结果以指定的名字代替

-t  显示执行命令

-p  交互式提示是否执行命令

-n  最大命令行参数

--show-limits  查看系统命令行长度限制

示例：

删除/tmp下名字是core的文件：

# find /tmp -name core -type f -print | xargs /bin/rm -f

# find /tmp -name core -type f -print0 | xargs -0 /bin/rm -f

列转行（去除换行符 ）：

# cut -d: -f1 < /etc/passwd | sort | xargs echo

行转列：

# echo "1 2 3 4 5" |xargs -n1

最长两列显示：

# echo "1 2 3 4 5" |xargs -n2

创建未来十天时间：

# seq 1 10 |xargs -i date -d "{} days " +%Y-%m-%d

复制多个目录：

# echo dir1 dir2 |xargs -n1 cp a.txt

清空所有日志：

# find ./ -name "\*.log" |xargs -i tee {} # echo ""> {} 这样不行，>把命令中断了

rm在删除大量的文件时，会提示参数过长，那么可以使用xargs删除：

# ls |xargs rm –rf

或分配删除 rm [a-n]\* -rf   # getconf ARG\_MAX 获取系统最大参数限制

## 29. nl

功能：打印文件行号

常用选项：

-b <a|t>   指定行号显示方式，a表示所有行都打印行号，b表示空行不显示行号，默认是a

-n <ln|rn|rz>  行号显示方法，ln左对齐，rn右对齐，rz右边显示，左边空白用0填充。

-w   行号栏位在左边占用的宽度

示例：

打印行号，空行不显示：

# nl a.txt

左对齐打印行号：

# nl -n ln a.txt

行号右移动五个空格：

# nl -w 5 a.txt

## 30. date

功能：打印或设置系统日期和时间

常用选项：

-d string  显示指定字符串所描述的时间，而非当前时间

-f datefile 从日期文件中按行读入时间描述

-I  输出ISO 8601格式的日期和时间

-r  显示文件的最后修改时间

-R  输出RFC 2822格式的日期和时间

-s string  设置时间所描述的字符串

-u  打印或设置UTC时间

控制输出格式：

%% 一个文字的 %

%a 当前locale 的星期名缩写(例如： 日，代表星期日)

%A 当前locale 的星期名全称 (如：星期日)

%b 当前locale 的月名缩写 (如：一，代表一月)

%B 当前locale 的月名全称 (如：一月)

%c 当前locale 的日期和时间 (如：2005年3月3日 星期四 23:05:25)

%C 世纪；比如 %Y，通常为省略当前年份的后两位数字(例如：20)

%d 按月计的日期(例如：01)

%D 按月计的日期；等于%m/%d/%y

%e 按月计的日期，添加空格，等于%\_d

%F 完整日期格式，等价于 %Y-%m-%d

%g ISO-8601 格式年份的最后两位 (参见%G)

%G ISO-8601 格式年份 (参见%V)，一般只和 %V 结合使用

%h 等于%b

%H 小时(00-23)

%I 小时(00-12)

%j 按年计的日期(001-366)

%k 时(0-23)

%l 时(1-12)

%m 月份(01-12)

%M 分(00-59)

%n 换行

%N 纳秒(000000000-999999999)

%p 当前locale 下的"上午"或者"下午"，未知时输出为空

%P 与%p 类似，但是输出小写字母

%r 当前locale 下的 12 小时时钟时间 (如：11:11:04 下午)

%R 24 小时时间的时和分，等价于 %H:%M

%s 自UTC 时间 1970-01-01 00:00:00 以来所经过的秒数

%S 秒(00-60)

%t 输出制表符 Tab

%T 时间，等于%H:%M:%S

%u 星期，1 代表星期一

%U 一年中的第几周，以周日为每星期第一天(00-53)

%V ISO-8601 格式规范下的一年中第几周，以周一为每星期第一天(01-53)

%w 一星期中的第几日(0-6)，0 代表周一

%W 一年中的第几周，以周一为每星期第一天(00-53)

%x 当前locale 下的日期描述 (如：12/31/99)

%X 当前locale 下的时间描述 (如：23:13:48)

%y 年份最后两位数位 (00-99)

%Y 年份

示例：

设置系统日期和时间：

# date -s "2016-12-15 00:00:00"

查看当前系统时间戳：

# date +%s

查看当前系统时间：

# date +'%F %T'

把日期和时间转换成时间戳：

# date -d "2016-12-15 18:00:00" +%s

把时间戳转成时间：

# date -d '@1481842800' '+%F %T'

时间加减：

显示前30秒：date -d '-30 second' +'%F %T'

显示前一分钟：date -d '-1 minute' +'%F %T'

显示前一个时间：date -d '-1 hour' +'%F %T'

显示前一个天：date -d '-1 day' +'%F %T'

显示上一周：date -d '-1 week' +'%F %T'

显示上一个月日期：date -d '-1 month' +%F

显示上一年日期：date -d '-1 year' +%F

或

显示前一天日期：date -d yesterday +%F

显示后一天日期：date -d tomorrow +%F

时间比较：

NOW\_DATE=$(date +%s)

AGO\_DATE=$(date -d "2016-12-15 18:00:00" +%s)

[ $NOW\_DATE -gt $AGO\_DATE ] && echo yes || echo no

## 31. wget

功能：非交互式网络下载，类似于HTTP客户端

常用选项：

-b,  --background   后台运行

日志记录和输入文件：

-o,  --output-file=FILE   日志写到文件

-a,  --append-output=FILE 日志追加到文件

-d,  --debug   打印debug信息，会包含头信息

-q,  --quiet   退出，不输出

-i,  --input-file=FILE   从文件中读取URL下载

下载选项：

-t,  --tries=NUMBER   设置链接重试次数

-O,  --output-document=FILE  写入内容到文件

-nc, --no-clobber   跳过下载现有的文件

-c,  --continue 断点续传

--progress=TYPE   设置进度条（dot和bar）

-S,  --server-response   打印服务器响应头信息

--spider   不下载任何内容

-T,  --timeout=SECONDS 设置相应超时时间（还有--dns-timeout、--connect-timeout和--read-timeout）

-w,  --wait=SECONDS   两次重试间隔等待时间

--bind-address=ADDRESS   设置绑定地址

--limit-rate=RATE   限制下载速度

--user=USER   设置ftp和http用户名

--password=PASS   设置ftp和http密码

目录：

-P,  --directory-prefix=PREFIX  保存文件目录

HTTP选项：

--http-user=USER   设置http用户名

--http-password=PASS   设置http密码

--proxy-user=USER 设置代理用户名

--proxy-password=PASS  设置代理密码

--referer=URL   设置Referer

--save-headers  保存头到文件

--default-page=NAME   改变默认页面名字，默认index.html

-U,--user-agent=AGENT  设置客户端信息

--no-http-keep-alive   禁用HTTP keep-alive（长连接）

--load-cookies=FILE   从文件加载cookies

--save-cookies=FILE   保存cookies到文件

--post-data=STRING   使用POST方法，发送数据

FTP选项：

--ftp-user=USER   设置ftp用户名

--ftp-password=PASS   设置ftp密码

--no-passive-ftp   禁用被动传输模式

递归下载：

-r, --recursive   指定递归下载

-l, --level=NUMBER   最大递归深度

-A, --accept=LIST   逗号分隔下载的扩展列表

-R, --reject=LIST 逗号分隔不被下载的扩展列表

-D, --domains=LIST   逗号分隔被下载域的列表

--exclude-domains=LIST 排除不被下载域的列表

示例：

下载单个文件到当前目录：

# wget http://nginx.org/download/nginx-1.11.7.tar.gz

放到后台下载：

# wget -b http://nginx.org/download/nginx-1.11.7.tar.gz

对于网络不稳定的用户使用-c和--tries参数，保证下载完成，并下载到指定目录：

# wget -t 3 -c http://nginx.org/download/nginx-1.11.7.tar.gz -P down

不下载任何内容，判断URL是否可以访问：

# wget --spider http://nginx.org/download/nginx-1.11.7.tar.gz

下载内容写到文件：

# wget http://www.baidu.com/index.html -O index.html

从文件中读取URL下载：

# wget -i url.list

下载ftp文件：

# wget --ftp-user=admin --ftp-password=admin ftp://192.168.1.10/ISO/CentOS-6.5-i386-minimal.iso

伪装客户端，指定user-agent和referer下载：

# wget -U "Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/44.0.2403.157 Safari/537.36" --referer "http://nginx.org/en/download.html" http://nginx.org/download/nginx-1.11.7.tar.gz

查看HTTP头信息：

# wget -S http://nginx.org/download/nginx-1.11.7.tar.gz

# wget --debug http://nginx.org/download/nginx-1.11.7.tar.gz

## 32. curl

功能：发送数据到URL，类似于HTTP客户端

常用选项：

-k, --insecure 允许HTTPS连接网站

-C, --continue-at  断点续传

-b, --cookie STRING/FILE  从文件中读取cookie

-c, --cookie-jar   把cookie保存到文件

-d, --data   使用POST方式发送数据

--data-urlencode   POST的数据URL编码

-F, --form   指定POST数据的表单

-D, --dump-header  保存头信息到文件

--ftp-pasv     指定FTP连接模式PASV/EPSV

-P, --ftp-port  指定FTP端口

-L, --location  遵循URL重定向，默认不处理

-l, --list-only 指列出FTP目录名

-H, --header   自定义头信息发送给服务器

-I, --head   查看HTTP头信息

-o, --output FILE   输出到文件

-#, --progress-bar  显示bar进度条

-x, --proxy [PROTOCOL://]HOST[:PORT]  使用代理

-U, --proxy-user USER[:PASSWORD] 代理用户名和密码

-e, --referer   指定引用地址referer

-O, --remote-name  使用远程服务器上名字写到本地

--connect-timeout  连接超时时间，单位秒

--retry NUM   连接重试次数

--retry-delay   两次重试间隔等待时间

-s, --silent   静默模式，不输出任何内容

-Y, --speed-limit  限制下载速率

-u, --user USER[:PASSWORD] 指定http和ftp用户名和密码

-T, --upload-file 上传文件

-A, --user-agent   指定客户端信息

示例：

下载页面：

# curl -o badu.html http://www.baidu.com

不输出下载信息：

# curl -s -o baidu.html http://www.baidu.com

伪装客户端，指定user-agent和referer下载：

# curl -A "Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/44.0.2403.157 Safari/537.36" -e "baike.baidu.com" http://127.0.0.1

模拟用户登录，并保存cookies到文件：

# curl -c ./cookies.txt -F NAME=user -F PWD=123 http://www.example.com/login.html

使用cookie访问：

# curl -b cookies.txt http://www.baidu.com

访问HTTP认证页面：

# curl -u user:pass http://www.example.com

FTP上传文件：

# curl -T filename ftp://user:pass@ip/a.txt

# curl ftp://ip -u user:pass-T filename

FTP下载文件：

# curl -O ftp://user:pass@ip/a.txt

# curl ftp://ip/filename -u user:pass -o filename

FTP下载多个文件：

# curl ftp://ip/img/[1,3,5].jpg

查看HTTP头信息：

# curl -I http://www.baidu.com

## 33. scp

功能：基于SSH的安全远程服务器文件拷贝

常用选项：

-i  指定私钥文件

-l  限制速率，单位Kb/s，1024Kb=1Mb

-P  指定远程主机SSH端口

-p  保存修改时间、访问时间和权限

-r  递归拷贝目录

-o  SSH选项，有以下几个比较常用的：

ConnectionAttempts=NUM   连接失败后重试次数

ConnectTimeout=SEC   连接超时时间

StrictHostKeyChecking=no 自动拉去主机key文件

PasswordAuthentication=no 禁止密码认证

示例：

本地目录推送到远程主机：

# scp -P 22 -r src\_dir root@192.168.1.10:/dst\_dir

远程主机目录拉取到本地：

# scp -P 22 root@192.168.1.10:dst\_dir src\_dir

同步文件方式一样，不用加-r参数

## 34. rsync

功能：远程或本地文件同步工具

常用选项：

-v  显示复制信息

-q  不输出错误信息

-c  跳过基础效验，不判断修改时间和大小

-a  归档模式，等效-rlptgoD，保留权限、属组等

-r  递归目录

-l  拷贝软连接

-z  压缩传输数据

-e  指定远程shell，比如ssh、rsh

--progress   进度条，等同-P

--bwlimit=KB/s   限制速率，0为没有限制

--delete   删除那些DST中SRC没有的文件

--exclude=PATTERN   排除匹配的文件或目录

--exclude-from=FILE 从文件中读取要排除的文件或目录

--password-file=FILE 从文件读取远程主机密码

--port=PORT 监听端口

示例：

本地复制目录：

# rsync -avz abc /opt

本地目录推送到远程主机：

# rsync -avz SRC root@192.168.1.120:DST

远程主机目录拉取到本地：

# rsync -avz root@192.168.1.10:SRC DST

保持远程主机目录与本地一样：

# rsync -avz --delete SRC root@192.168.1.120:DST

排除某个目录：

# rsync -avz --exclude=no\_dir SRC root@192.168.1.120:DST

指定SSH端口：

# rsync -avz /etc/hosts -e "ssh -p22" root@192.168.1.120:/opt

## 35. nohup

功能：运行程序，忽略挂起信号

示例：

后台运行程序，终端关闭不影响：

# nohup bash test.sh &>test.log &

## 36. iconv

功能：将文件内容字符集转成其他字符集

常用选项：

-l  列出所有已知的字符集

-f  原始文本编码

-t  输出编码

-o  输出到文件

-s  关闭警告

示例：

将文件内容转换UTF8：

# iconv -f gbk -t utf8 old.txt -o new.txt

将csv文件转换GBK：

# iconv -f utf8 -t gbk old.txt -o new.txt

解决邮件乱码：

# echo $(echo "content" | iconv -f utf8 -t gbk) | mail -s "$(echo "title" | iconv -f utf8 -t gbk)" example@mail.com

## 37. uname

功能：输出系统信息

常用选项：

-a  输出以下所有信息

-s  输出内核名称

-n  输出主机名

-r  输出内核发行版

-v  输出内核版本

-m  输出主机的硬件架构名称

-p  输出处理器类型或"unknown"

-i  输出硬件平台或"unknown

-o  输出操作系统名称

示例：

输出所有系统信息：

# uname -a

输出主机名：

# uname -a

输出内核版本：

# uname -r

输出操作系统：

# uname -o

## 38. sshpass

功能：非交互SSH登录（需要安装）

常用选项：

-f  从文件中获取密码

-d  用文件描述符数字获取密码

-p  指定SSH密码

-e  密码作为环境变量传递，变量名是SSHPASS

示例：

免交互SSH登录：

# sshpass -p 123456 ssh root@192.168.1.10

免交互传输文件：

# sshpass -p 123456 scp a.txt 192.168.1.10:/root

密码传入系统变量：

# SSHPASS=123456 rsync -avz /etc/hosts -e "sshpass -e ssh" root@192.168.1.221:/opt

## 39. tar

功能：归档目录或文件

常用选项：

-c 创建新归档

-d  比较归档和文件系统的差异

-r  追加文件到归档

-t  存档的内容列表

-x 提取归档所有文件

-C  改变解压目录

-f  使用归档文件或设备归档

-j  bzip2压缩

-z gzip压缩

-v  输出处理过程

示例：

创建归档文件来自foo和bar：

# tar -cf archive.tar foo bar

提取归档的所有文件：

# tar -xf archive.tar

列出所有归档文件内容：

# tar -tvf archive.tar

创建归档并gzip压缩：

# tar -zcvf archive.tar.gz log

提取归档文件并gzip解压：

# tar -zxvf log.tar.gz

创建归档并bzip2压缩：

# tar -jcvf log.tar.bz log

提取归档并解压到指定目录：

# tar -zxvf log.tar.gz -C /opt

## 40. logger

功能：系统日志的shell命令行接口

常用选项：

-i 每行记录进程ID

-f 指定输出日志到文件

-p 设置记录的优先级

-t 添加标签

示例：

# logger -i -t "my\_test" -p local3.notice "test\_info"

## 41. netstat

功能：打印网络连接、路由表、接口统计信息、伪装连接和多播成员

常用选项：

-r  显示路由表

-i  显示接口表

-n  不解析名字

-p  显示程序名 PID/Program

-l  显示监听的socket

-a  显示所有socket

-o  显示计时器

-Z  显示上下文

-t  只显示tcp连接

-u  只显示udp连接

-s  显示每个协议统计信息

示例：

显示所有监听：

# netstat -anltu

显示所有TCP连接：

# netstat -antp

显示所有UDP连接：

# netstat -anup

显示路由表：

# netstat -r

## 42. ss

功能：比netstat更强大的socket查看工具

格式：ss [options] [ FILTER ]

常用选项：

-n  不解析名字

-a  显示所有socket

-l  显示所有监听的socket

-o  显示计时器

-e  显示socket详细信息

-m  显示socket内存使用

-p  显示进程使用的socket

-i  显示内部TCP信息

-s  显示socket使用汇总

-4  只显示IPV4的socket

-0  显示包socket

-t  只显示TCP socket

-u  只显示UDP socket

-d  只显示DCCP socket

-w  只显示RAW socket

-x  只显示Unix域socket

-f  FAMILY  只显示socket族类型（ unix, inet, inet6, link, netlink）

-A  查询socket {all|inet|tcp|udp|raw|unix|packet|netlink}[,QUERY]

-D 将原始的TCP socket转储到文件

-F 从文件中读取过滤信息

过滤：

-o  state  显示TCP连接状态信息

示例：

显示所有TCP连接：

# ss -t -a

显示所有UDP连接：

# ss -u –a

显示socket使用汇总：

# ss -s

显示所有建立的连接：

# ss -o state established

显示所有的TIME-WAIT状态：

# ss -o state TIME-WAIT

搜索所有本地进程连接到X Server：

# ss -x src /tmp/.X11-unix/\*

## 43. lsof

功能：列出打开的文件

常用选项：

-i [i]  监听的网络地址，如果没有指定，默认列出所有。

[i]来自[46][protocol][@hostname|hostaddr][:service|port]

-U  列出Unix域socket文件

-p  指定PID

-u  指定用户名或UID所有打开的文件

+D 递归搜索

示例：

列出所有打开的文件：

# lsof

查看哪个进程占用文件：

# lsof /etc/passwd

列出所有打开的监听地址和unix域socket文件：

# lsof -i -U

列出80端口监听的进程：

# lsof -i:80

列出端口1-1024之间的所有进程：

# lsof -i:1-1024

列出所有TCP网络连接：

# lsof -i tcp

列出所有UDP网络连接：

# lsof -i udp

根据文件描述符列出打开的文件：

# lsof -d 1

列出某个目录被打开的文件：

# lsof +D /var/log

列出进程ID打开的文件：

# lsof -p 5373

打开所有登录用户名abc或user id 1234，或PID 123或PID 456：

# lsof -p 123,456 -u 123,abc

列出COMMAND列中包含字符串sshd：

# lsof -c sshd

## 44. ps

功能：报告当前进程的快照

常用选项：

-a  显示所有进程

-u  选择有效的用户ID或名称

-x  显示无控制终端的进程

-e  显示所有进程

-f  全格式

-r  只显示运行的进程

-T  这个终端的所有进程

-p  指定进程ID

--sort  对某列排序

-m  线程

-L  格式化代码列表

-o  用户自定义格式

CODE   NORMAL   HEADER

%C     pcpu     %CPU

%G     group    GROUP

%P     ppid     PPID

%U     user     USER

%a     args     COMMAND

%c     comm     COMMAND

%g     rgroup   RGROUP

%n     nice     NI

%p     pid      PID

%r     pgid     PGID

%t     etime    ELAPSED

%u     ruser    RUSER

%x     time     TIME

%y     tty      TTY

%z     vsz      VSZ

示例：

打印系统上所有进程标准语法：

# ps -ef

打印系统上所有进程BSD语法：

# ps aux

打印进程树：

# ps axjf 或 ps -ejH

查看进程启动的线程：

# ps -Lfp PID

查看当前用户的进程数：

# ps uxm 或 ps -U root -u root u

自定义格式显示并对CPU排序：

# ps -eo user,pid,pcpu,pmem,nice,lstart,time,args --sort=-pcpu

或 ps -eo "%U %p %C %n %x %a"

## 45. top

功能：动态显示活动的进程和系统资源利用率

常用选项：

-d  信息刷新时间间隔

-p  只监控指定的进程PID

-i  只显示正在使用CPU的进程

-H  显示线程

-u  只查看指定用户名的进程

-b  将输出编排成易处理格式，适合输出到文件处理

-n  指定最大循环刷新数

交互命令：

f  添加或删除显示的指标

c  显示完整命令

P  按CPU使用百分比排序

M  按驻留内存大小排序

T  按进程使用CPU时间排序

1  显示每个CPU核心使用率

k  终止一个进程

示例：

刷新一次并输出到文件：

# top -b -n 1 > top.log

只显示指定进程的线程：

# top -Hp 123

传入交互命令，按CPU排序

## 46. free

功能：查看内存使用率

常用选项：

-b  bytes显示

-k  KB显示

-m  M显示

-g  G显示

-h  易读单位显示

-s  每几秒重复打印

-c 重复打印几次退出

示例：

查看物理内存：

# free -m

易读单位显示：

# free -h

## 47. df

功能：查看文件系统的磁盘空间使用情况

常用选项：

-a  包含虚拟文件系统

-h  可易读单位显示

-i  显示inode 信息而非块使用量

-k  1K块的数量

-t  只显示指定文件系统为指定类型的信息

-T  显示文件系统类型

示例：

查看所有文件系统：

# df -ah

输出指定文件系统：

# df -t xfs

## 48. vmstat

功能：报告虚拟内存、swap、io、上下文和CPU统计信息。

分析了这些文件：

/proc/meminfo

/proc/stat

/proc/\*/stat

常用选项：

-a  打印活跃和不活跃的内存页

-d  打印硬盘统计信息

-D  打印硬盘表

-p  打印硬盘分区统计信息

-s  打印虚拟内存表

-m  打印内存分配（slab）信息

-t  添加时间戳到输出

-S  显示单位，默认k、KB、m、M，大写是\*1024

示例：

分析系统性能：

# vmstat

每秒刷新一次，统计五次：

# vmstat -t 1 5

## 49. iostat

功能：报告CPU利用率和磁盘I/O

用法: iostat [ 选项 ] [ <时间间隔> [ <次数> ] ]

常用选项：

-c  显示CPU使用率

-d  只显示磁盘使用率

-k  单位KB/s代替Block/s

-m  单位MB/s代替Block/s

-N 显示所有映射设备名字

-t  打印报告时间

-x  显示扩展统计信息

示例：

显示CPU使用率：

# iostat -c 1 3

显示I/O磁盘统计信息：

# iostat -d -x -k 1 3 # 间隔1秒，输出3次

## 50. sar

功能：查看系统资源综合方面利用率

常用选项：

-u, CPU

-r, memory

-b, disk

-n DEV, NIC traffic

-q, systemload

-b, TPS（Transaction Per Second，每秒事务处理量）

-o, output to file

示例：

# sar -u 2 3 #每两秒执行一次，采集三次

# sar -u 2 3 -o cpu.out

# sar -f cpu.out #读取文件

## 51. dstat

功能：查看系统资源综合方面利用率（需要安装）

常用选项：

-c cpu统计

-d 磁盘统计

-m 内存统计

-n 网络统计

-s swap统计

-l 负载统计

--tcp tcp状态统计

--udp udp状态统计

--socket socket数量统计

-t 输出时间

--output 写入csv文件

插件：

--list 支持的插件

--top-bio-adv 详细显示I/O进程写入block量，包括pid、r、w和cpu

--top-io-adv 进程写入磁盘总量

--top-cpu 占用CPU进程

--top-cpu-adv 查看最高CPU进程

--top-mem 内存进程

示例：

查看CPU利用率：

# dstat -c

查看TCP连接状态：

# dstat --tcp

## 52. ip

功能：查看/操作路由表，设备，路由策略和隧道

格式：ip [ OPTIONS ] OBJECT { COMMAND | help }

常用选项：

-b, -batch <FILENAME>  从文件或标准输入读取命令并调用他们，第一次失败将终止

-force 批量模式有错误不终止，如果有错误则状态返回非0

-s, -statistics 输出更多的统计信息

-l, -loops <COUNT>  指定最大的循环数

操作对象（OBEJECT）：

address  网络设备地址

12tp   以太网IP隧道

link   配置网络设备

maddress 多播地址

monitor 动态监控网络连接

mroute   多播路由缓存条目

mrule   角色在多播路由策略数据库

neighbour 管理ARP或NDISC缓存条目

netns   管理网络命名空间

ntable   管理neighbour缓存操作

route   路由表

rule   角色在路由策略数据库

tpc\_metrics/tcpmetrics 管理TCP指标

tunnel   IP隧道

tuntap   管理TUN/TAP设备

xfrm   管理IPSec策略

可通过ip OBEJECT help再查看对象的操作方法。

示例：

查看网络设备地址：

# ip addr

查看网卡统计信息：

# ip -s link

查看单个网卡统计信息：

# ip -s link ls eth0

查看ARP缓存表：

# ip neighbour

查看路由表：

# ip route

查看路由策略：

# ip rule

网卡设置/删除IP：

# ip addr add/del 192.168.1.201/24 dev eth0

添加/删除默认路由：

# ip route add default via 192.168.1.1 dev eth0

# ip route del 192.168.1.0/24 via 192.168.1.1

添加静态路由：

# ip route add 172.17.2.0/24 via 192.168.2.1 dev eth0

开启/关闭网卡：

# ip link set dev eth0 up/down

设置最大传输单元：

# ip link set dev eth0 mtu 1500

设置MAC地址：

# ip link set dev eth0 address 00:0c:29:52:73:8e

## 53. nc

功能：TCP和UDP连接和监听

常用选项：

-i interval 指定间隔时间发送和接受行文本

-l  监听模式，管理传入的连接

-n 不解析域名

-p  指定本地源端口

-s  指定本地源IP地址

-u  使用udp协议，默认是tcp

-v  执行过程输出

-w  timeout 连接超时时间

-x  proxy\_address[:port]  请求连接主机使用代理地址和端口

-z  指定扫描监听端口，不发送任何数据

示例：

端口扫描：

# nc -z 192.168.1.10 1-65535

TCP协议连接到目标端口：

# nc -p 31337 -w 5 192.168.1.10 22

UDP协议连接到目的端口：

# nc -u 192.168.1.10 53

指定本地IP连接：

# nc -s 192.168.1.9 192.168.1.10 22

探测端口是否开启：

# nc -z -w 2 192.168.1.10 22

创建监听Unix域Socket：

# nc -lU /var/tmp/ncsocket

通过HTTP代理连接主机：

# nc -x10.2.3.4:8080 -Xconnect 10.0.0.10 22

监听端口捕获输出到文件：

# nc -l 1234 > filename.out

从文件读入到指定端口：

# nc host.example.com 1234 < filename.in

收发信息：

# nc -l 1234

# nc 127.0.0.1 1234

执行memcahced命令：printf "stats\n" |nc 127.0.0.1 11211

发送邮件：

# nc [-C] localhost 25 << EOF

           HELO host.example.com

           MAIL FROM: <user@host.example.com>

           RCPT TO: <user2@host.example.com>

           DATA

           Body of email.

           .

           QUIT

           EOF

# echo -n "GET / HTTP/1.0\r\n\r\n" | nc host.example.com 80

## 54. time

功能：执行脚本时间

常用选项：

示例：

查看执行ls所需的时间：

# time ls

## 55. eval

功能：执行参数作为shell命令

示例：

for i in $@; do

eval $i

done

echo ---

echo $a

echo $b

# bash test.sh a=1 b=2

---

1

2

## 56. ssh

功能：SSH客户端

常用选项：

-p 指定远程主机端口

-i 指定认证文件

-L [bind\_address:]port:host:hostport

-R [bind\_address:]port:host:hostport]

-D [bind\_address:]port

-o SSH选项，有以下几个比较常用的：

ConnectionAttempts=NUM   连接失败后重试次数

ConnectTimeout=SEC   连接超时时间

StrictHostKeyChecking=no 自动拉去主机key文件

PasswordAuthentication=no 禁止密码认证

示例：

登录到远程主机：

# ssh user@192.168.1.120

远程主机执行命令：

# ssh user@192.168.1.120 'ifconfig'

本地文件内容写到远程主机文件：

# ssh user@192.168.1.120 'cat >> file' < /etc/passwd

SSH还提供了一个非常有用的功能，就是端口转发，能帮你解决一些无法建立的连接。

1. 本地端口转发

应用场景1：A不能访问C，B能访问A和C，实现通过B能让A访问C

在主机A执行：

# ssh -L 2222:主机C:22 主机B # ssh -L [绑定地址:]本地端口:主机C:C端口 主机B

将SSH绑定本地端口2222，本地2222端口数据转发主机B，主机B的所有数据转发到主机C的22端口；这样一来，只要在主机A ssh -p 2222 localhost，就等于连上了主机C的22端口。

应用场景2：一台Squid代理服务器，限制了本机可以清理缓存，但是我想从远程服务器清理

在远程服务器执行：

# ssh -L 31280:localhost:3128 SquidHost

在远程服务器上执行清理命令到本机31280端口，31280收到的数据加密转发到SquidHost的SSH Server上，SSH Client解密收到的数据并转发到监听的3128端口上，最后将Squid返回的数据原路返回。

1. 远程端口转发

应用场景1：A不能访问C，B能访问A和C，但A不能访问B，比如A在外网，B在内网

在主机B执行：

# ssh -R 2222:主机C:22 主机A

将SSH绑定本机2222端口，与主机A建立SSH通道，当主机A访问本地2222端口，就等于访问主机B的2222端口，主机B的2222端口把数据转发到主机C的22端口。

应用常见2：公司有一台内网服务器，还有一台云主机不能SSH直接连接这台公司内网服务器，但内网服务器可以SSH连接云主机

在公司内网服务器执行：

# ssh -R 2222:localhost:22 云主机

将云主机上的2222端口数据转发到内网服务器SSH Client上，SSH Client解密收到的数据并转发到监听的22端口上，最后再将返回的数据原路返回。

3）动态端口转发（不限定端口，全权代理）

应用场景：翻墙访问国外网站

如果是MAC系统直接在终端执行：ssh -D 2222 国外云主机

如果是Windows系统可借助putty工具实现，在putty里面端口转发->本地端口转发属性里面添加一个本地端口，并勾选SOCKS4/5动态转发，连接即可。

将SSH绑定本机8080端口，SSH就会创建一个SOCKS代理服务，直接在浏览器上设置代理本机127.0.0.1的8080端口即可，当浏览器访问国外网站时，本地代理把请求转发到国外云主机的SSH Server，SSH解密并转发给指定的网站。

注意：再Linux终端执行ssh绑定命令后，默认会进入一个新的shell，只要这个shell不退出，此端口转发就一直有效。如果要想放到后台执行就加-Nf两个选项，-N是不执行命令，-f后台执行，这样就转入后台运行，就可以在本地shell执行操作了，如果想关闭后台就kill这个进程。

参考资料：https://www.ibm.com/developerworks/cn/linux/l-cn-sshforward/

## 57. iptables

常见几种类型防火墙？

包过滤防火墙：包过滤是IP层实现，包过滤根据数据包的源IP、目的IP、协议类型（TCP/UDP/ICMP）、源端口、目的端口等包头信息及数据包传输方向灯信息来判断是否允许数据包通过。

应用层防火墙：也称为应用层代理防火墙，基于应用层协议的信息流检测，可以拦截某应用程序的所有封包，提取包内容进行分析。有效防止SQL注入或者XSS（跨站脚本攻击）之类的恶意代码。

状态检测防火墙：结合包过滤和应用层防火墙优点，基于连接状态检测机制，将属于同一连接的所有包作为一个整体的数据流看待，构成连接状态表（通信信息，应用程序信息等），通过规则表与状态表共同配合，对表中的各个连接状态判断。

iptables是Linux下的配置防火墙的工具，用于配置Linux内核集成的IP信息包过滤系统，使增删改查信息包过滤表中的规则更加简单。

iptables分为四表五链，表是链的容器，链是规则的容器，规则指定动作。

四表：

|  |  |
| --- | --- |
| filter | 用于包过滤 |
| nat | 网络地址转发 |
| mangle | 对特定数据包修改 |
| raw | 不做数据包链接跟踪 |

五链：

|  |  |
| --- | --- |
| INPUT | 本机数据包入口 |
| OUTPUT | 本机数据包出口 |
| FORWARD | 经过本机转发的数据包 |
| PREROUTING | 防火墙之前，修改目的地址（DNAT） |
| POSTROUTING | 防火墙之后，修改源地址（SNAT） |

表中的链：

|  |  |
| --- | --- |
| **表** | **链** |
| filter | INPUT、OUTPUT和FORWARD |
| nat | PREROUTING、POSTROUTING和OUTPUT |
| mangle | PREROUTING、POSTROUTING、INPUT、OUTPUT和FORWARD |
| raw | PREROUTING和OUTPUT |

命令格式：iptables [-t table] 命令 [chain] 匹配条件 动作

|  |  |
| --- | --- |
| **命令** | **描述** |
| -A，append | 追加一条规则 |
| -I，insert | 插入一条规则，默认链头，后跟编号，指定第几条 |
| -D，delete | 删除一条规则 |
| -F，flush | 清空规则 |
| -L，list | 列出规则 |
| -P，policy | 设置链缺省规则 |
| -m，module | 模块，比如state、multiport |

|  |  |
| --- | --- |
| **匹配条件** | **描述** |
| -i | 入口网卡 |
| -o | 出口网卡 |
| -s | 源地址 |
| -d | 目的地址 |
| -p | 协议类型 |
| --sport | 源端口 |
| --dport | 目的端口 |

|  |  |
| --- | --- |
| **动作** | **描述** |
| ACCEPT | 允许数据包通过 |
| DROP | 丢弃数据包不做处理 |
| REJECT | 拒绝数据包，并返回报错信息 |
| SNAT | 一般用于nat表的POSTROUTING链，进行源地址转换 |
| DNAT | 一般用于nat表的PREROUTING链，进行目的地址转换 |
| MASQUERADE | 动态源地址转换，动态IP时使用 |

|  |  |
| --- | --- |
| **模块** | **描述** |
| state | 包状态，有四个：NEW、RELATED、ESTABLISHED和INVALID |
| mac | 源MAC地址 |
| limit | 包速率限制 |
| multiport | 多端口，以逗号分隔 |
| iprange | 端口范围，以逗号分隔 |

示例：常用的规则配置方法

iptables -F         # 清空表规则，默认filter表

iptables -t nat -F   # 清空nat表

iptables -A INPUT -p tcp --dport 22 -j ACCEPT   # 允许TCP的22端口访问

iptables -I INPUT -p udp --dport 53 -j ACCEPT   # 允许UDP的53端口访问，插入在第一条

iptables -A INPUT -p tcp --dport 22:25 -j ACCEPT # 允许端口范围访问

iptables -D INPUT -p tcp --dport 22:25 -j ACCEPT # 删除这条规则

# 允许多个TCP端口访问

iptables -A INPUT -p tcp -m multiport --dports 22,80,8080 -j ACCEPT

iptables -A INPUT -s 192.168.1.0/24 -j ACCEPT # 允许192.168.1.0段IP访问

iptables -A INPUT -s 192.168.1.10 -j DROP    # 对1.10数据包丢弃

iptables -A INPUT -i eth0 -p icmp -j DROP    # eth0网卡ICMP数据包丢弃，也就是禁ping

# 允许来自lo接口，如果没有这条规则，将不能通过127.0.0.1访问本地服务

iptables -A INPUT -i lo -j ACCEPT

# 限制并发连接数，超过30个拒绝

iptables -I INPUT -p tcp --syn --dport 80 -m connlimit --connlimit-above 30 -j REJECT

# 限制每个IP每秒并发连接数最大3个

iptables -I INPUT -p tcp --syn -m limit --limit 1/s --limit-burst 3 -j ACCEPT

iptables -A FORWARD -p tcp --syn -m limit --limit 1/s -j ACCEPT

# iptables服务器作为网关时，内网访问公网

iptables –t nat -A POSTROUTING -s [内网IP或网段] -j SNAT --to [公网IP]

# 访问iptables公网IP端口，转发到内网服务器端口

iptables –t nat -A PREROUTING -d [对外IP] -p tcp --dport [对外端口] -j DNAT --to [内网IP:内网端口]

# 本地80端口转发到本地8080端口

iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-ports 8080

# 允许已建立及该链接相关联的数据包通过

iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# ASDL拨号上网

iptables -t nat -A POSTROUTING -s 192.168.1.0/24 -o pppo -j MASQUERADE

# 设置INPUT链缺省操作丢弃所有数据包，只要不符合规则的数据包都丢弃。注意要在最后设置，以免把自己关在外面！

iptables -P INPUT DROP