linux 基础

压缩及解压缩

压缩格式: gz , bz2, xz, zip, z

压缩算法,压缩比不同

compress file : xxx.Z

uncompress

很古老了,现在一般不用,**都用xz 压缩比更大**

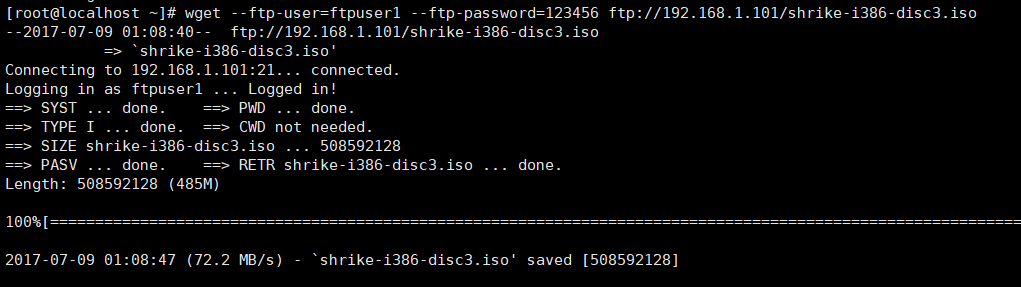
gizp file : xxx.gz

bzip2 file : xxx.bz2

xz file : xxx.xz

以上**只能压缩文件不能压缩目录**,而且压缩后会删除原文件保留压缩后的文件

wget --ftp-user=ftpuser1 --ftp-password=123456 ftp://192.168.1.101/xxxxx.xxx

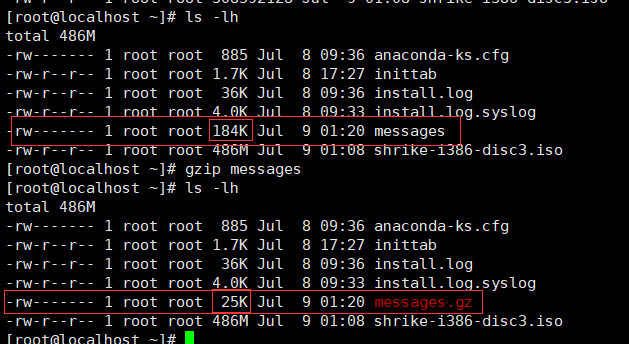


## gzip

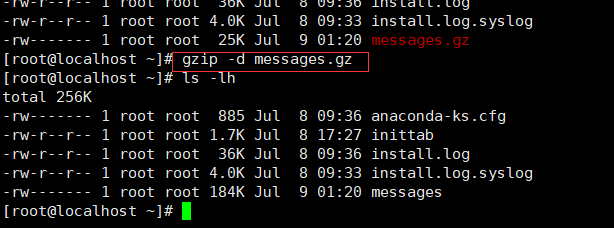
-d 解压缩

-# 指定压缩比

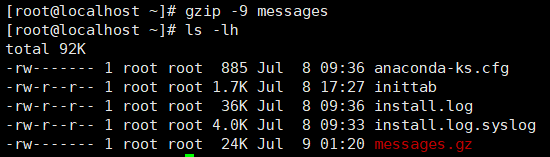
gzip xxx 压缩文件 删除原文件



gizp -d xxx.gz



压缩比为9

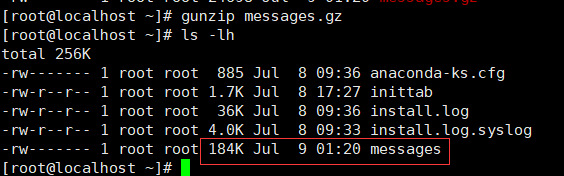


## gunzip

gunzip = gzip -d

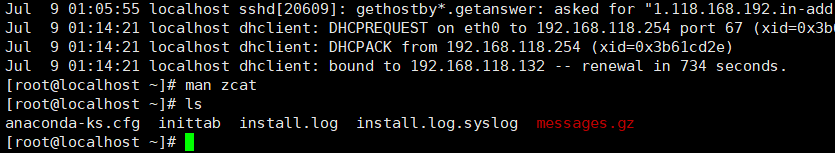
解压缩后也要删除原文件

gunzip xxx.gz



## zcat

zcat xxx.gz 表示在不解压的情况下查看文件



## bzip2

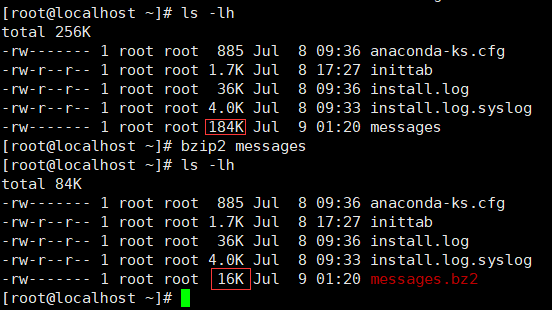
-d 解压缩

-k 压缩时保留原文件

比gzip有着更大压缩比的压缩工具,对于小文件没有优势

用法与gzip相同

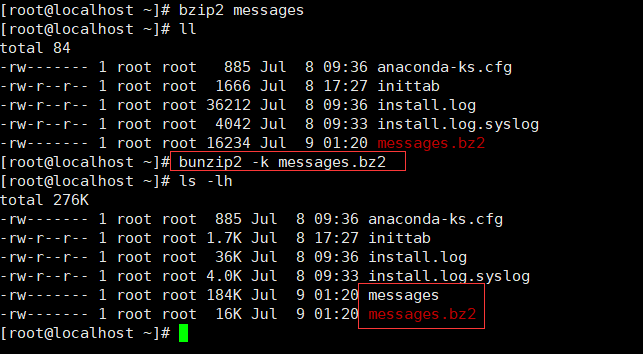
后缀名为.bz2,也会删除原文件,



## bunzip2

-k 解压缩时保留原文件

bunzip2 xxx.bz2 = bzip2 -d xxx.bz2

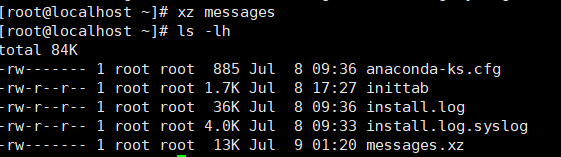


## bzcat

不用解压缩来查看文件内容

## xz

-k 保留原文件

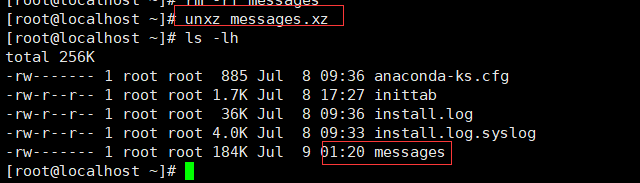


## unxz

unxz xxx.xz = xz -d xxx.xz

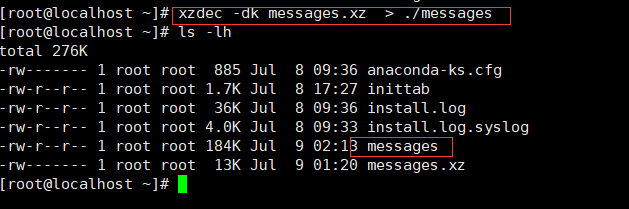
-k 保留原文件

解压缩



## xzdec

也可以实现解压缩



## xzcat

不解压查看文件内容

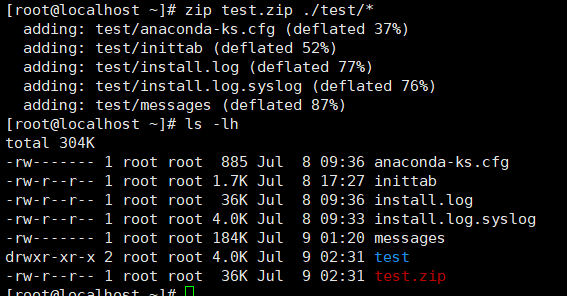
## zip

**即归档有压缩**

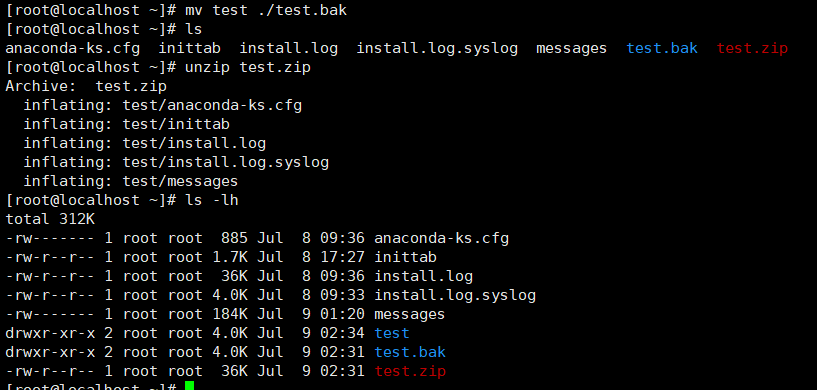
**zip 压缩后的文件名 要压缩的文件名或者目录(file 1,file2,file3)**

zip压缩比不是很大,但是他可以压缩目录,不会删除原文件

压缩家目录中所有文件为test.zip



unzip xxx



这种把一些文件放在一起,叫做归档 archive ,归档并不是意味着压缩

## tar

归档工具 ,只归档不压缩

**-c** : 创建归档文件

**-f** file.tar : 操作的归档文件

**-x** : 还原归档

**--xattrs** : 归档的时候,保留文件的扩展属性

**-t** : 不还原归档查看归档了哪些文件

-C : 指定目录

**-zcf** : 归档并调用gzip压缩

**-zxf** : 调用gzip解压缩并展开归档

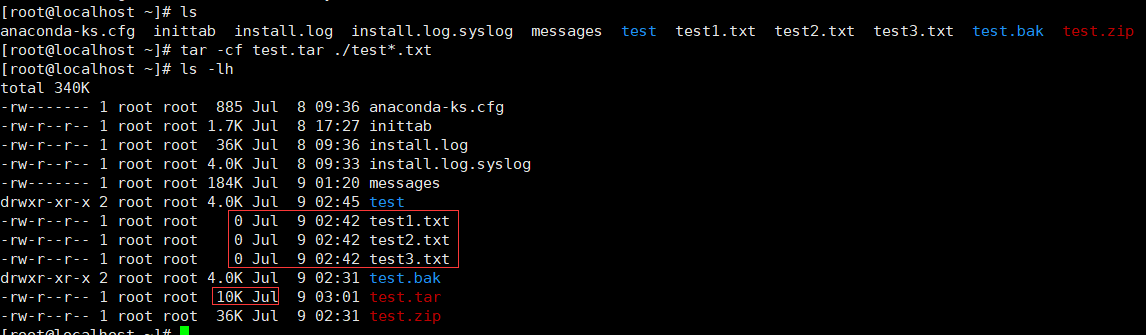
-jcf: 归档并调用bzip2压缩

-jxf: 调用bzip2解压缩并展开归档

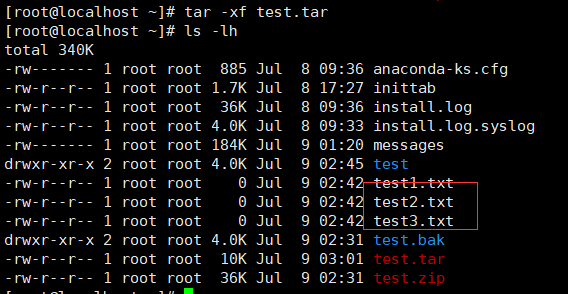
-Jcf: 归档并调用xz压缩

-Jxf: 调用xz解压缩并展开归档

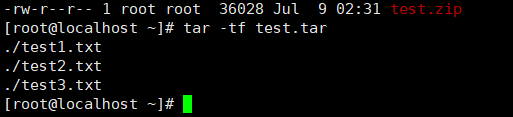
tar 的选项可以省略前面的-



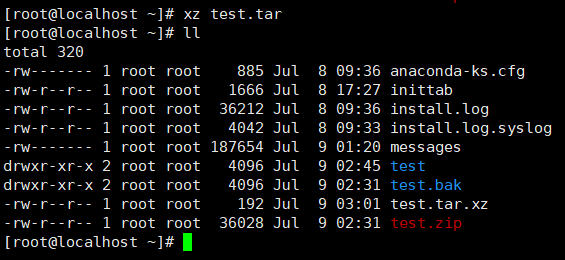
tar -xf xxx.tar 还原归档文件



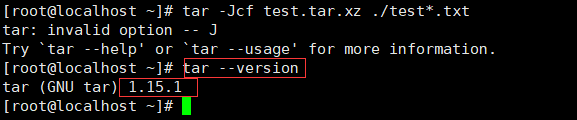
tar -tf xxx.tar 查看xxx.tar归档了哪些文件



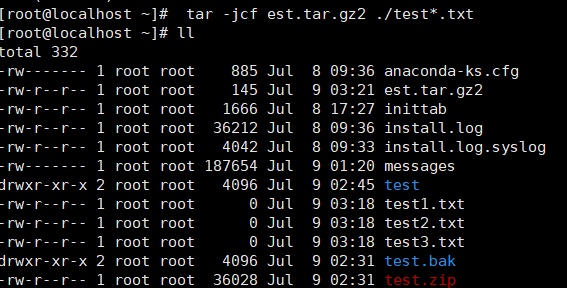
在使用xz压缩工具压缩 就完成了归档并压缩



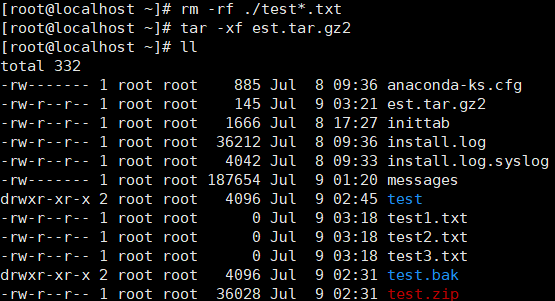
rh5.8 不支-J选项 版本太低, rh6上是可以的



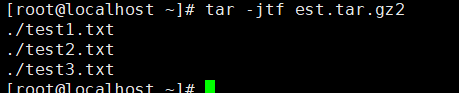
使用bzip2压缩



还原归档,自动识别压缩工具调用解压缩



不还原归档查看归档了哪些文件



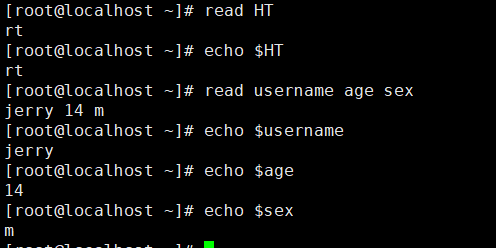
## cpio

也是一个归档工具,很古老,但是在redhat上还有用到

## read

**-p** 给出提示

**-t** 限定时间



#/bin/bash

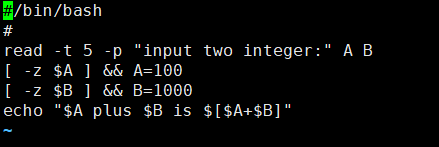
#

read -t 5 -p "input two integer:" A B

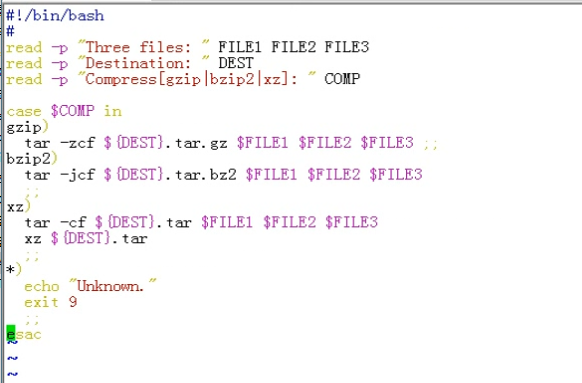
[ -z $A ] && A=100

[ -z $B ] && B=1000

echo "$A plus $B is $[$A+$B]"



指定3个文件进行归档



## while循环

顺序结构

选择结构

if

case

循环结构

for 循环次数已知

while 循环次数未知

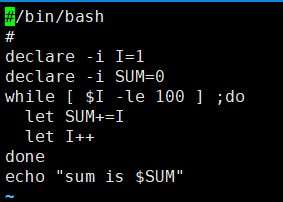
until

while conditions ; do

statement;

done;

计算100以内所有正整数的和



#!/bin/bash

#

declare -i I=1

declare -i SUM=0

while [ $I -le 100 ] ;do

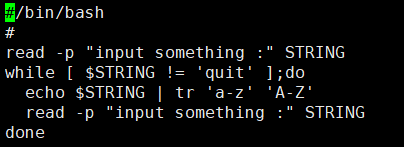
let SUM+=I

let I++

done

echo "sum is $SUM"

小写变大写



#!/bin/bash

#

read -p "input something :" STRING

while [ $STRING != 'quit' ];do

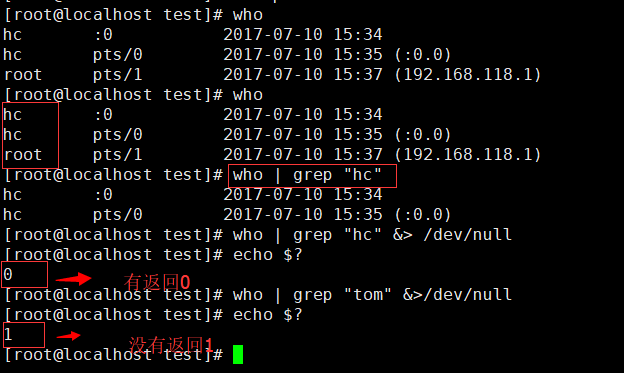
echo $STRING | tr 'a-z' 'A-Z'

read -p "input something :" STRING

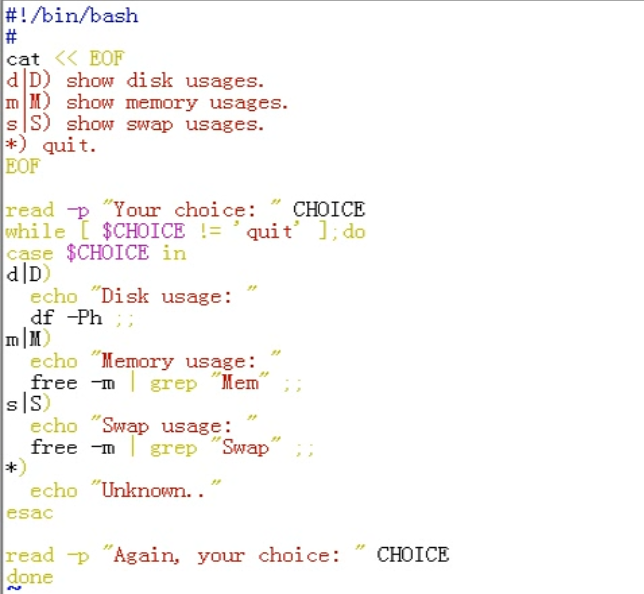
done

检查hadoop用户是否登录,每隔5秒检查一次









#/bin/bash

#

cat << EOF

d|D) show disk usage.

m|M) show memeory usage.

s|S) show swap usage.

\*) quit.

EOF

read -p "input your choice" CHOICE

while [ $CHOICE != 'quit' ]; do

case $CHOICE in

d|D)

echo "disk usage:"

df -Ph

;;

m|M)

echo "memeory usage:"

free -m | grep "Mem"

;;

s|S)

echo "swap usage:"

free -m | grep "swap"

;;

\*)

echo "Unknown."

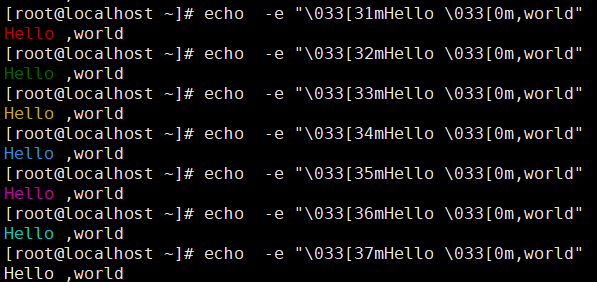
;;

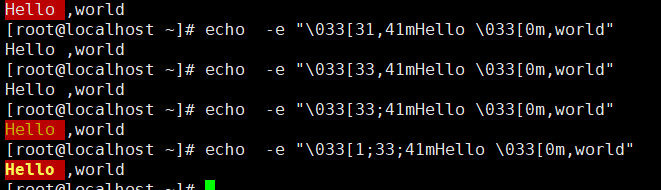
esac

read -p "input your choice" CHOICE

done

-e 逃逸符





使用颜色

