netstat -ano

-tpl

lsof -i:端口号

查看显卡nvidia-smi

两台机子A/B

A: nc -l 9999 在A机器上启动一个端口监听，比如 9999端口,默认tcp

telnet B 9999测试连接

nc -vzw 2 B 9998-9999 扫描两个端口

-w<超时秒数> 设置等待连线的时间。

**端口扫描经常被系统管理员和黑客用来发现在一些机器上开放的端口，帮助他们识别系统中的漏洞。**

$ nc -z -v -n 172.31.100.7 21-25

可以运行在TCP或者UDP模式，默认是TCP，-u参数调整为udp.

z 参数告诉netcat使用0 IO,连接成功后立即关闭连接， 不进行数据交换

v 参数指使用冗余选项

n 参数告诉netcat 不要使用DNS反向查询IP地址的域名

这个命令会打印21到25 所有开放的端口。Banner是一个文本，Banner是一个你连接的服务发送给你的文本信息。当你试图鉴别漏洞或者服务的类型和版本的时候，Banner信息是非常有用的。但是，并不是所有的服务都会发送banner。

一旦你发现开放的端口，你可以容易的使用netcat 连接服务抓取他们的banner。

$ nc -v 172.31.100.7 21

netcat 命令会连接开放端口21并且打印运行在这个端口上服务的banner信息。

1. 通过 kill `ps -ef | grep 'ddd'`      
#这种形式，这个时候实际上等同于拼接字符串得到的命令，其效果类似于  kill $pid

2. for procid in $(ps -aux | grep "some search" | awk '{print $2}'); do kill -9 $procid; done     
#其实与第一种原理一样，只不过需要多次kill的时候是循环处理的，每次处理一个

3. ps -ef | grep 'ddd' | xargs kill    
#OK，使用了xargs命令，铺垫了这么久终于铺到了主题上。xargs命令可以通过管道接受字符串，并将接收到的字符串通过空格分割成许多参数(默认情况下是通过空格分割) 然后将参数传递给其后面的命令，作为后面命令的命令行参数，kill不直接接手管道

Xargs -d分隔符默认为空格

-p 询问命令是否执行

-n 每次传递几个参数给其后面的命令执行

-E 只取该参数对应字符之前的数据

find . -name "\*.txt" -print0 | xargs -0 echo

xargs的 -0 和 -d '\0' 表示其从标准输入中读取的内容使用 '\0' 来分割

**command || { echo "command failed"; exit 1; }**

将一个已有用户 cnzhx 增加到一个已有用户组 apache 中，使此用户组成为该用户的附加用户组，可以使用带 -a 参数的 usermod

指令。-a 代表 append， 也就是将用户添加到新用户组中而不必离开原有的其他用户组。不过需要与 -G 选项配合使用：

# usermod -a -G apache cnzhx

如果要同时将 cnzhx 的主要用户组改为 apache，则直接使用 -g 选项：

# usermod -g apache cnzhx

如果要将一个用户从某个组中删除，则

gpasswd -d user group

======================

====================

你写的每个脚本都应该在文件开头加上set -e,这句语句告诉bash如果任何语句的执行结果不是true则应该退出。

这样的好处是防止错误像滚雪球般变大导致一个致命的错误，而这些错误本应该在之前就被处理掉。如果要增加可读性，

可以使用set -o errexit，它的作用与set -e相同。

command

if [ "$?" -ne 0]; then

echo "command failed";

exit 1;

fi

1，基础知识。

#!bin/bash

su root<<EOF

12qwaszx

whoami

pwd

ifconfig

whoami

EOF

#!/usr/bin/expect -f

#不可以去掉spawn，英文expect中没有ssh命令

spawn ssh root@mu4

expect "\*password:"

send "123456\r"

expect "\*#"

#有interact才可以停留在mu4主机，否则会退回mu3主机

interact

2，模式-动作。

单一分支模式语法：

expect "hi" {send "You said hi"}

匹配到hi后，会输出"you said hi"

多分支模式语法：

expect "hi" { send "You said hi\n" } \

"hello" { send "Hello yourself\n" } \

"bye" { send "That was unexpected\n" }

匹配到hi,hello,bye任意一个字符串时，执行相应的输出。等同于如下写法：

expect {

"hi" { send "You said hi\n"}

"hello" { send "Hello yourself\n"}

"bye" { send "That was unexpected\n"}

}

**文件描述符**

exec 3<>test.sh;

#打开test.sh可读写操作，与文件描述符3绑定

while read line < &3

do

echo $line;

done

#循环读取文件描述符3（读取的是test.sh内容）

exec 3>&-

exec 3<&-

#关闭文件的，输入，输出绑定

exec 6>&1

#将标准输出与fd 6绑定

[chengmo@centos5

shell]$ ls /proc/self/fd/

0 1 2 3 6

#出现文件描述符6

[chengmo@centos5

shell]$ exec 1>suc.txt

#将接下来所有命令标准输出，绑定到suc.txt文件（输出到该文件）

[chengmo@centos5

shell]$ ls -al

#执行命令，发现什么都不返回了，因为标准输出已经输出到suc.txt文件了

[chengmo@centos5

shell]$ exec 1>&6

#恢复标准输出

[chengmo@centos5

shell]$ exec 6>&-

#关闭fd

6描述符

[chengmo@centos5

~]$ ls /proc/self/fd/

0 1 2 3

--------------------------------

-------------------------------------

#!bin/bash

PRONUM=10 #进程个数

tmpfile="$$.fifo" #临时生成管道文件

mkfifo $tmpfile

exec 6<>$tmpfile

rm $tmpfile

for(( i=0; i<$PRONUM; i++ ))

do

echo "init."

done >&6

while read line

do

read -u6

{ table\_name="$line"

if [[ "$table\_name" == $prefix\* ]]

then

out\_table\_name=$table\_name

else

out\_table\_name=$prefix$table\_name

fi

sqoop import \

--connect jdbc:srdbsql://${databaseip[0]}:${databaseport[0]}/${databasetable[0]} \

--driver org.srdbsql.Driver \

--username ${databaseuser[0]} \

--password ${databasepwd[0]} \

--query "select \

\* \

from $database\_model.$table\_name where \$CONDITIONS" \

-z \

--null-string '' --null-non-string '' \

--delete-target-dir --target-dir /ods/srdb/jk/$out\_table\_name \

--fields-terminated-by '\001' -m 1

if [ $? -ne 0 ]

then

echo $table\_name >> /txdb/ods/offline/sqoop/srdb/jk/impl/BUG\_table

fi

echo >& 6

echo "$table\_name finished"

} &

done < /txdb/ods/offline/sqoop/srdb/jk\_table\_list

----------------------------------

------------------------------

#!/bin/bash

#导入到hdfs

#table\_name=表名

#-z 开启压缩，GZip

#--compression-codec org.apache.hadoop.io.compress.BZip2Codec BZip格式

databaseip=(`awk -F '|' '{print $1}' /txdb/ods/offline/sqoop/srdb/database\_config.cfg`)

databaseport=(`awk -F '|' '{print $2}' /txdb/ods/offline/sqoop/srdb/database\_config.cfg`)

databasetable=(`awk -F '|' '{print $3}' /txdb/ods/offline/sqoop/srdb/database\_config.cfg`)

databaseuser=(`awk -F '|' '{print $4}' /txdb/ods/offline/sqoop/srdb/database\_config.cfg`)

databasepwd=(`awk -F '|' '{print $5}' /txdb/ods/offline/sqoop/srdb/database\_config.cfg`)

database\_model=(`awk -F '|' '{print $6}' /txdb/ods/offline/sqoop/srdb/database\_config.cfg`)

prefix="t\_"

out\_table\_name=""

for table\_name in `cat /txdb/ods/offline/sqoop/srdb/jk\_table\_list`

do

{

if [[ "$table\_name" == $prefix\* ]]

then

out\_table\_name=$table\_name

else

out\_table\_name=$prefix$table\_name

fi

sqoop import \

--connect jdbc:srdbsql://${databaseip[0]}:${databaseport[0]}/${databasetable[0]} \

--driver org.srdbsql.Driver \

--username ${databaseuser[0]} \

--password ${databasepwd[0]} \

--query "select \

\* \

from $database\_model.$table\_name where \$CONDITIONS" \

-z \

--null-string '' --null-non-string '' \

--delete-target-dir --target-dir /ods/srdb/jk/$out\_table\_name \

--fields-terminated-by '\001' -m 1

if [ $? -ne 0 ]

then

echo $table\_name >> /txdb/ods/offline/sqoop/srdb/jk/impl/BUG\_table

fi

}&

done{split}

1.按一定的规则编写服务脚本，比如：myserviced

[plain] view plain copy print?

#!/bin/sh

#chkconfig: 2345 80 05

#description: myservice

case $1 in

start)

echo "myservice startup" #将该行替换成你自己的服务启动命令

;;

stop)

echo "myservice stop" #将该行替换成你自己服务的启动命令

;;

restart)

echo "myservice stop" #...

echo "myservice startup" #...

;;

\*)

;;

esac

命令解析：

# chkconfig: 2345 80 5

2345表示服务的运行级别，80代表Start的顺序，05代表Kill（Stop）的顺序；

# description: service\_description

该服务的描述

2.将编写的脚本放到/etc/init.d/，将myserviced的访问权限加上“可执行”

[plain] view plain copy print?

chmod +x myserviced

3.增加服务

[plain] view plain copy print?

chkconfig --add myserviced

4.启停服务

[plain] view plain copy print?

service myserviced start

service myserviced stop

服务添加完成

可以用：chkconfig --list查看当前系统的服务

可以用：chkconfig --del myserviced删除服务