

# NSD SHELL DAY07

1. [案例1：编写一键部署软件脚本](#)
2. [案例2：启动脚本](#)
3. [案例3：编写监控脚本](#)
4. [案例4：编写安全检测脚本](#)
5. [案例5：编写进度显示脚本](#)

## 1 案例1：编写一键部署软件脚本

### 1.1 问题

本案例要求编写脚本实现一键部署Nginx软件（Web服务器）：

- 一键源码安装Nginx软件
- 脚本自动安装相关软件的依赖包
- 脚本自动判断yum是否可用

### 1.2 步骤

实现此案例需要按照如下步骤进行。

#### 步骤一：准备工作

1) 判断yum源是否可用

通过yum repolist查看软件包数量是否大于0：

```
01. [ root@svr5 ~] # yum repolist
02. [ root@svr5 ~] # yum repolist | awk '/repolist/{ print $2} '
03. [ root@svr5 ~] # yum repolist | awk '/repolist/{ print $2} ' | sed 's/,//'
04. [ root@svr5 ~] # N=$(yum repolist | awk '/repolist/{ print $2} ' | sed 's/,//')
05. [ root@svr5 ~] # [ $N -le 0 ] && echo 'yum 不可用'
```

2) 依赖包

源码安装Nginx需要提前安装依赖包软件gcc,openssl-devel,pcre-devel

#### 步骤二：编写脚本

1) 参考脚本内容如下：

```
01. [ root@svr5 ~] # vim test.sh
02. #!/bin/bash
03.
04. N=$(yum repolist | awk '/repolist/{ print $2} ' | sed 's/,//')
05. if [ $N -le 0 ];then
06.     echo "yum 不可用"
```

[Top](#)

07. exit
08. fi
09. yum -y install gcc openssl-devel pcre-devel
10. tar -xf nginx-1.12.2.tar.gz
11. cd nginx-1.12.2
12. ./configure
13. make
14. make install

## 2) 确认安装效果

Nginx默认安装路径为/usr/local/nginx,该目录下会提供4个子目录, 分别如下:

/usr/local/nginx/conf 配置文件目录

/usr/local/nginx/html 网站页面目录

/usr/local/nginx/logs Nginx日志目录

/usr/local/nginx/sbin 主程序目录

主程序命令参数:

01. [ root@svr5 ~] # /usr/local/nginx/sbin/nginx //启动服务
02. [ root@svr5 ~] # /usr/local/nginx/sbin/nginx -s stop //关闭服务
03. [ root@svr5 ~] # /usr/local/nginx/sbin/nginx -V //查看软件信息

## 2 案例2: 启动脚本

### 2.1 问题

本案例要求编写Ngin启动脚本, 要求如下:

- 脚本支持start、stop、restart、status
- 脚本支持报错提示
- 脚本具有判断是否已经开启或关闭的功能

### 2.2 步骤

实现此案例需要按照如下步骤进行。

#### 步骤一: 编写脚本

脚本通过位置变量\$1读取用户的操作指令, 判断是start、stop、restart还是status。

netstat命令可以查看系统中启动的端口信息, 该命令常用选项如下:

-n以数字格式显示端口号

-t显示TCP连接的端口

-u显示UDP连接的端口

-l显示服务正在监听的端口信息, 如httpd启动后, 会一直监听80端口

[Top](#)

-p显示监听端口的服务名称是什么（也就是程序名称）

1) 参考脚本内容如下：

```
01.  [ root@svr5 ~] # vim test.sh
02.  #!/bin/bash
03.
04.  case $1 in
05.  start)
06.      /usr/local/nginx/sbin/nginx;;
07.  stop)
08.      /usr/local/nginx/sbin/nginx - s stop;;
09.  restart)
10.      /usr/local/nginx/sbin/nginx - s stop
11.      /usr/local/nginx/sbin/nginx;;
12.  status)
13.      netstat - ntulp | grep - q nginx
14.      if [ $? - eq 0 ];then
15.  echo 服务已启动
16.  else
17.  echo 服务未启动
18.  fi;;
19.  *)
20.      echo Error;;
21.  esac
```

2) 执行测试脚本：

```
01.  [ root@svr5 ~] # ./test.sh start
02.  [ root@svr5 ~] # ./test.sh stop
03.  [ root@svr5 ~] # ./test.sh status
04.  [ root@svr5 ~] # ./test.sh xyz
```

## 3 案例3：编写监控脚本

### 3.1 问题

本案例要求编写脚本，实现计算机各个性能数据监控的功能，具体监控项目要求如下：

- CPU负载
- 网卡流量
- 内存剩余容量

[Top](#)

- 磁盘剩余容量
- 计算机账户数量
- 当前登录账户数量
- 计算机当前开启的进程数量
- 本机已安装的软件包数量

## 3.2 步骤

实现此案例需要按照如下步骤进行。

### 步骤一：准备工作

#### 1) 查看性能数据的命令

```

01. [ root@svr5 ~] # uptime           //查看CPU负载
02. [ root@svr5 ~] # ifconfig eth0     //查看网卡流量
03. [ root@svr5 ~] # free             //查看内存信息
04. [ root@svr5 ~] # df               //查看磁盘空间
05. [ root@svr5 ~] # wc -l /etc/passwd //查看计算机账户数量
06. [ root@svr5 ~] # who | wc -l      //查看登录账户数量
07. [ root@svr5 ~] # rpm -qa | wc -l  //查看已安装软件包数量

```

### 步骤二：编写参考脚本

#### 1) 脚本内容如下：

```

01. [ root@svr5 ~] # vim test.sh
02. #!/bin/bash
03. ip=`ifconfig eth0 | awk '/inet /{ print $2}`
04. echo "本地IP地址是:$ip"
05. cpu=`uptime | awk '{ print $NF}`
06. #awk中NF为当前行的列数，$NF是最后一列
07. echo "本机CPU最近15分钟的负载是:$cpu"
08. net_in=`ifconfig eth0 | awk '/RX p/{ print $5}`
09. echo "入站网卡流量为:$net_in"
10. net_out=`ifconfig eth0 | awk '/TX p/{ print $5}`
11. echo "出站网卡流量为:$net_out"
12. mem=`free | awk '/Mem/{ print $4}`
13. echo "内存剩余容量为:$mem"
14. disk=`df | awk '/\$/ { print $4}`
15. echo "根分区剩余容量为:$disk"
16. user=`cat /etc/passwd | wc -l`
17. echo "本地账户数量为:$user"
18. login=`who | wc -l`

```

[Top](#)

19. echo "当前登陆计算机的账户数量为: "\$login
20. process=`ps aux | wc -l`
21. echo "当前计算机启动的进程数量为: "\$process
22. soft=`rpm -qa | wc -l`
23. echo "当前计算机已安装的软件数量为: "\$soft

## 4 案例4：编写安全检测脚本

### 4.1 问题

本案例要求编写脚本，防止远程ssh暴力破解密码，具体监控项目要求如下：

- 检测ssh登录日志，如果远程登陆账号名错误3次，则屏蔽远程主机的IP
- 检测ssh登录日志，如果远程登陆密码错误3次，则屏蔽远程主机的IP

### 4.2 步骤

实现此案例需要按照如下步骤进行。

#### 步骤一：准备工作

1) 过滤帐户名失败的命令(登陆日志文件为/var/log/secure)

```
01 [ root@svr5 ~] # awk '/Invalid user/{ print $10}' /var/log/secure
```

2) 过滤密码失败的命令

```
01 [ root@svr5 ~] # awk '/Failed password/{ print $11}' /var/log/secure
```

#### 步骤二：编写参考脚本

1) 脚本内容如下：

```
01 [ root@svr5 ~] # vim test.sh
02. #!/bin/bash
03. awk '/Failed password/{ print $11}' /var/log/secure | awk '{ ip[ $1] ++} END{ for( i in ip) { print ip[ i]
04.
05. awk '/Invalid user/{ print $10}' /var/log/secure | awk '{ ip[ $1] ++} END{ for( i in ip) { print ip[ i] ,i}}
```



[Top](#)

## 5 案例5：编写进度显示脚本

### 5.1 问题

本案例要求编写脚本，实现带进程显示的复制脚本，具体要求如下：

- 默认Linux的cp命令不具有进度显示
- 我们需要自己编写脚本实现进度显示
- 可以使用进度条的方式，或者显示百分比的方式

## 5.2 步骤

实现此案例需要按照如下步骤进行。

### 步骤一：编写参考脚本

1) 脚本内容如下：

```
01. [ root@svr5 ~] # vim test.sh
02. #!/bin/bash
03. jindu(){
04.     while :
05.     do
06.         echo -ne '\033[ 43m \033[ 0m'
07.         sleep 0.3
08.     done
09. }
10. jindu &
11. cp -r $1 $2
12. kill $!
```

[Top](#)