# Linux 101：分区、备份与批处理（2 下）

崔　灏

Linux User Group

2018 年 6 月 3 日

# Unix Shell（回顾）

Unix Shell: 命令行界面

- sh/ksh 类：
  - bash：多数 Linux 发行版的默认 shell
  - dash/ash：比 bash 轻量，多用于嵌入式设备
  - zsh：功能强大，最近比较流行
- csh 类：BSD 系统上比较常用

shell 不仅支持交互式运行命令还具有很多编程功能，包括变量、条件分支和循环控制、字符串处理、文件 IO 等功能。
了解这些功能，就能在命令行实现更复杂的程序逻辑。

本节内容主要针对 bash。

# bash 高级语法

## 变量

### 变量:

```
a=1; b=2
host=www.baidu.com
warn="Kernel Panic!"   # 字符串带空格要加引号
```

### 引用变量:

```
echo ${a}+${b}=${c}    # 引用变量，带花括号
c=$((a+b))             # 数学运算，并赋给新的变量
ping -c $c $host       # 引用变量，不带花括号
```

# bash 高级语法

## IO 功能

重定向标准输出到文件：

```
ping -c3 baidu.com > ping.result    # 重写
ping -c3 baidu.com >> ping.result   # 追加
```

重定向文件到标准输入：

```
cat < ping.result
```

捕获标准输出到变量：

```
DATE=$(date +%Y%m%d)
DATE=`date +%Y%m%d`    # 两者等价
echo $DATE
```

# bash 高级语法

## IO 功能

管道（前一个命令输出作为后一个命令输入）：

```
# 查找 src 目录下的 .c 文件
# 用 cat 查看文件内容
# 用 wc 统计总行数
find src/ -iname '*.c' | xargs cat | wc -l

# 用 du 统计家目录下所有文件夹占用空间
# 用 sort 按占用空间排序
du ~ -d1 | sort -nk1
```

# bash 高级语法

## 程序流程

- 顺序执行：

  ```
  command1; command2; ...
  ```

- 前一个命令成功（返回 0）才继续执行：

  ```
  command1 && command2; ...
  ```

- 前一个命令失败（返回 0）才继续执行：

  ```
  command1 || command2; ...
  ```

- 查看上一个命令返回值：

  ```
  command
  echo $?
  ```

# bash 高级语法

## 分支语句

基本格式：if 命令; then command; else command; fi

```
if [[ -e local.conf ]]; then
    cat local.conf
else
    echo "local.conf not found!"
fi

if [[ "$param" == "hello"  ]]; then
    echo hi
fi

if grep -q "ERROR" event.log; then
    echo $warning;
fi
```

# bash 高级语法

## 循环语句

while 和 if 格式类似

```
while [[ -e lock.pid ]]; then
    sleep 10
done

# for 变量 in ...; do
for i in {1..20}; do
    echo $i
done

# C-like
for ((i=1; i<=20; i++)); do
    echo $i
done
```

# bash 脚本

shell 可以从文件读取命令，这类文件称为 shell 脚本。
复杂的命令行操作，可以写成 shell 脚本，方便以后执行。

```
#!/bin/bash
USERNAME=$(whoami)
DATE=$(date +%Y%m%d)
LOCKFILE="/tmp/simplelock.$USERNAME"

if [[ -e "$LOCKFILE" ]]; then
    echo "Another script is running. I won't run."
    exit 1
else
    touch "$LOCKFILE"
fi

echo "Hi, $USERNAME. Today is $DATE."
sleep 2
echo "Bye~"
rm -f $LOCKFILE
```

# bash 脚本

运行：

```
$ bash a.sh
# 或者（注意脚本第一行要有 #!）
$ chmod +x a.sh
$ ./a.sh
Hi, cuihao. Today is 20180603.
Bye~
$
```

# bash 脚本

## 其他常用功能

命令行参数：

`$1, $2, $3...`

加载另一个脚本（如读取变量）：

```
source another-script
```

调试运行（输出执行的命令）：`bash -x script.sh`

# 示例：备份脚本

功能描述：把文件夹备份到指定备份目录，并自动删除旧的备份。
如果有 rsync 则用 rsync，没有则用 cp
配置文件 etc.conf：

```
SRC=/etc/
DST=/mnt/backup/etc.backup/
KEEP=3
LOG=$DST/logfile
```

调用脚本：

```
$ ./simplebackup etc.conf
```

# 结束

Q & A