21 性能分析综合案例

性能分析综合案例

一、案例分析

当系统资源达到瓶颈时，系统性能下降，系统无法正常提供服务。为了查找性能瓶颈，需要登录服务器，逐个输入命令分析性能瓶颈，但是Linux性能瓶颈命令烦琐，为以后方便使用，可以就编写使用脚本。

该脚本需要实现以下功能。

- 查看CPU利用率与负载(top、vmstat、sar)。
- 查看磁盘、Inode利用率与I/O负载(df、iostat、iotop、sar、dstat)。
- 查看内存利用率(free、vmstat)。
- 查看TCP连接状态(netstat、ss)。
- 查看CPU与内存占用最高的10个进程(top、ps)。
- 查看网络流量(ifconfig、iftop、iptraf)。

脚本的算法设计如下：

Shell | 复制代码

```
检测操作系统
检测包管理工具
检测并安装性能工具
主循环：根据菜单选择执行专项性能分析
        专项性能分析
```

二、代码要点分析

根据实际需求，脚本可以分解为以下部分。

2.1 检测操作系统版本并确定包管理工具

Bash 复制代码

```
[root@Shell ~]# vi stat1.sh
#!/bin/bash
#检测操作系统信息并确定包管理工具
os_check() {
#获取操作系统信息
        if [ -e /etc/redhat-release ]; then
                REDHAT=$(cat /etc/redhat-release | cut -d' ' -f1)
        else
                DEBIAN=$(cat /etc/issue | cut -d' ' -f1)
        fi
#根据操作系统确定包管理工具
        if [ "$REDHAT" == "CentOS" -o "$REDHAT" == "Red" ]; then
                P_M=yum
        elif [ "$DEBIAN" == "Ubuntu" -o "$DEBIAN" == "ubutnu" ]; then
                P_M=apt-get
        else
                echo"Operating system does not support."
                exit 1
        fi
        echo $REDHAT $DEBIAN $P_M
}
os_check
[root@Shell ~]# . stat1.sh
CentOS yum
```

2.2 检测性能工具并安装

```bash
[root@Shell ~]# vi stat2.sh
#查看登录的用户是否为root
if [ $LOGNAME != root ]; then
        echo "Please use the root account operation."
        exit 1
fi
#查看是否存在vmstat命令
if ! which vmstat &>/dev/null; then
        echo "vmstat command not found, now the install."
        sleep 1
        os_check
        $P_M install procps -y
        echo "--------------------------------"
fi
#查看是否存在iostat命令
if ! which iostat &>/dev/null; then
        echo "iostat command not found, now the install."
        sleep 1
        os_check
        $P_M install sysstat -y
        echo "--------------------------------"
fi
[root@Shell ~]# . stat2.sh
```

2.3 菜单、命令执行主循环

2.3.1 select in循环

`select in` 循环用来**增强交互性**，它可以显示**带编号的菜单**，用户输入不同的编号就可以选择不同的菜单，并执行不同的功能。

`select in` 是shell独有的一种循环，非常适合终端这样的交互场景。

`select in` 语句的语法格式如下：

```shell
select 变量 in 值列表
do
    语句块
done
```

```shell
[root@shell ~]# vi select.sh
#!/bin/bash
echo "What is your favourite OS? "
select name in "Linux" "Windows" "Mac OS" "Unix" "Android"
#""可以省略
do
        echo $name
done
#此处不会自动跳出select in 循环，需要按ctrl+D跳出select in
echo "You have selected $name"
[root@shell ~]# . select.sh
What is your favourite OS?
1) Linux
2) Windows
3) Mac OS
4) Unix
5) Android
#? 2
Windows
#?
You have selected Windows
```

`select in` 语句在实际应用中往往结合 `case` 语句。

```shell
[root@shell ~]# vi select_case.sh
#!/bin/bash
echo "What is your favourite OS? "
select name in "Linux" "Windows" "Mac OS" "Unix" "Android"
do
    case $name in
        "Linux")
          echo "Linux是一个类Unix操作系统，它开源免费"
          break
          ;;
        "Windows")
          echo "Mac OS是微软开发的个人电脑操作系统，它是闭源收费的"
          break
          ;;
        "Mac OS")
          echo "Mac OS是苹果公司开发的一款图形界面操作系统"
          break
          ;;
        "Unix")
          echo "Unix是操作系统的开山鼻祖，现在只应用在一些特殊场合"
          break
          ;;
        "Android")
          echo "Android是由Google开发的手机操作系统"
          break
          ;;
        *)
          echo "没有这个选项！"
          break
    esac
done
[root@shell ~]# . select_case.sh
What is your favourite OS?
1) Linux
2) Windows
3) Mac OS
4) Unix
5) Android
#? 1
Linux是一个类Unix操作系统，它开源免费
```

2.3.2 主循环结构

Bash | 复制代码

```
[root@Shell ~]# vi stat3.sh
while true; do
  select input in cpu_load disk_load disk_use disk_inode mem_use tcp_statu
s cpu_top10 mem_top10 traffic quit; do
    case $input in
    cpu_load)
      #CPU 利用率与负载
      echo "利用率与负载----------------------------------------------"
      break
      ;;
    #......
    quit)
      exit 0
      ;;
    *)
      echo "------------------------------------------------"
      echo "Please enter the number."
      echo "------------------------------------------------"
      break
      ;;
    esac
  done
done
[root@Shell ~]# . stat3.sh
1) cpu_load      3) disk_use     5) mem_use      7) cpu_top10    9) traffi
c
2) disk_load     4) disk_inode   6) tcp_status   8) mem_top10   10) quit
#? 1
利用率与负载----------------------------------------------
1) cpu_load      3) disk_use     5) mem_use      7) cpu_top10    9) traffi
c
2) disk_load     4) disk_inode   6) tcp_status   8) mem_top10   10) quit
#? 10
```

Bash | 复制代码

```
[root@Shell ~]# vi stat4.sh
#打印菜单
while true; do
  select input in cpu_load disk_load disk_use disk_inode mem_use tcp_statu
s cpu_top10 mem_top10 traffic quit; do
    case $input in
    cpu_load)
      #CPU 利用率与负载
      echo "CPU 利用率与负载----------------------------------------------"
      break
      ;;
    disk_load)
      #硬盘 I/O 负载
      echo "硬盘 I/O 负载------------------------------------------------"
      break
      ;;
    disk_use)
      #硬盘利用率
      echo "硬盘利用率----------------------------------------------------"
      break
      ;;
    disk_inode)
      #硬盘 inode 利用率
      echo "硬盘 inode 利用率--------------------------------------------
_"
      break
      ;;
    mem_use)
      #内存利用率
      echo "内存利用率----------------------------------------------------"
      break
      ;;
    tcp_status)
      #网络连接状态
      echo "网络连接状态--------------------------------------------------"
      ;;
    cpu_top10)
      #占用 CPU 高的前 10 个进程
      echo "占用 CPU 高的前 10 个进程-------------------------------------
--------"
      break
      ;;
    mem_top10)
      #占用内存高的前 10 个进程

```

```
        echo "占用内存高的前 10 个进程----------------------------------------
------"
        break
        ;;
      traffic)
        #查看网络流量
        echo "查看网络流量------------------------------------------------"
        break
        ;;
      quit)
        exit 0
        ;;
      *)
        echo "------------------------------------------------"
        echo "Please enter the number."
        echo "------------------------------------------------"
        break
        ;;

      esac
    done
done
[root@Shell ~]# . stat4.sh
```

2.4 子命令处理

CPU 利用率与负载统计子命令处理，替换 `#CPU 利用率与负载` 部分。

Bash 复制代码

```
        i=1
        while [[ $i -le 3 ]]; do
          echo -e "\033[32m 参考值${i}\033[0m"
          UTIL=$(vmstat | awk '{if(NR==3)print 100-$15"%"}')
          USER=$(vmstat | awk '{if(NR==3)print $13"%"}')
          SYS=$(vmstat | awk '{if(NR==3)print $14"%"}')
          IOWAIT=$(vmstat | awk '{if(NR==3)print $16"%"}')
          echo "Util: $UTIL"
          echo "User use: $USER"
          echo "System use: $SYS"
          echo "I/O wait: $IOWAIT"
          i=$(($i + 1))
          sleep 1
        done
```

三、完整代码

小结

- 脚本流程设计
- 性能分析工具的应用

课程目标

- 知识目标：熟练掌握性能分析工具的应用。
- 技能目标：能够利用性能分析工具编写实际应用脚本。

课外拓展

- 进一步了解性能分析工具的应用场景。

参考资料

- 《Linux系统命令及Shell脚本实践指南 》，王军，机械工业出版社
- 《跟老男孩学Linux运维：Shell编程实战》，老男孩，机械工业出版社