NSD SERVICES DAY03

1 案例1：配置并验证Split分离解析

1.1 问题

本例要求配置一台智能DNS服务器，针对同一个FQDN，当不同的客户机来查询时能够给出不同的答案。需要完成下列任务：

1. 从主机192.168.4.207查询时，结果为：www.tedu.cn ---> 192.168.4.100
2. 从其他客户端查询时，www.tedu.cn ---> 1.2.3.4

1.2 方案

在配置DNS服务器时，通过view视图设置来区分不同客户机、不同地址库：

```
view "视图1" {
    match-clients { 客户机地址1; .. .. ; };        //匹配第1类客户机地址
    zone "目标域名" IN {                          //同一个DNS区域
        type master;
        file "地址库1";                          //第1份地址库
    };
};
view "视图2" {
    match-clients { 客户机地址2; .. .. ; };        //匹配第2类客户机地址
    match-clients { any; };                      //匹配任意地址
    zone "目标域名" IN {                          //同一个DNS区域
        type master;
        file "地址库2";                          //第2份地址库
    };
};
.. ..
view "视图n" {
    match-clients { any; };                      //匹配任意地址
    zone "目标域名" IN {                          //同一个DNS区域
        type master;
        file "地址库n";                          //第n份地址库
    };
```

Top

```
23.     };
```

1.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：配置Split分离解析

1）为tedu.cn区域建立两份解析记录文件

第一份解析记录文件提供给客户机192.168.4.207、网段192.168.7.0/24，对应目标域名www.tedu.cn的A记录地址为192.168.4.100。相关操作及配置如下：

```
01.     [root@svr7 ~]# cd /var/named/
02.     [root@svr7 named]# cp -p tedu.cn.zone tedu.cn.zone.lan
03.     [root@svr7 named]# vim tedu.cn.zone.lan
04.     $TTL 1D
05.     @   IN SOA  @ rname.invalid. (
06.                     0   ; serial
07.                     1D  ; refresh
08.                     1H  ; retry
09.                     1W  ; expire
10.                     3H )    ; minimum
11.     @       NS  svr7.tedu.cn.
12.     svr7    A   192.168.4.7
13.     pc207   A   192.168.4.207
14.     www     A   192.168.4.100
```

第二份解析记录文件提供给其他客户机，对应目标域名www.tedu.cn的A记录地址为1.2.3.4。相关操作及配置如下：

```
01.     [root@svr7 named]# cp -p tedu.cn.zone tedu.cn.zone.other
02.     [root@svr7 named]# vim tedu.cn.zone.other
03.     $TTL 1D
04.     @   IN SOA  @ rname.invalid. (
05.                     0   ; serial
06.                     1D  ; refresh
07.                     1H  ; retry
08.                     1W  ; expire
09.                     3H )    ; minimum
10.     @       NS  svr7.tedu.cn.
11.     svr7    A   192.168.4.7
```

Top

```
pc207   A   192.168.4.207
www     A   1.2.3.4
```

2）修改named.conf配置文件，定义两个view，分别调用不同解析记录文件

```
[root@svr7 ~]# vim /etc/named.conf
options {
        directory  "/var/named";
};
acl "mylan" {                                  //名为mylan的列表
        192.168.4.207; 192.168.7.0/24;
};
.. ..
view "mylan" {
    match-clients { mylan; };                  //检查客户机地址是否匹配此列表
    zone "tedu.cn" IN {
        type master;
        file "tedu.cn.zone.lan";
    };
};
view "other" {
    match-clients { any; };                    //匹配任意客户机地址
    zone "tedu.cn" IN {
        type master;
        file "tedu.cn.zone.other";
    };
};
```

3）重启named服务

```
[root@svr7 ~]# systemctl  restart  named
```

步骤二：测试分离解析效果

1）从mylan地址列表中的客户机查询

在客户机192.168.4.207（或网段192.168.7.0/24内的任意客户机）上查询www.tedu.cn，结果是 192.168.4.100：

Top

```
[root@pc207 ~]# host  www.tedu.cn 192.168.4.7
Using domain server:
Name: 192.168.4.7
Address: 192.168.4.7#53
Aliases:

www.tedu.cn has address 192.168.4.100
```

2）从其他客户机查询

在DNS服务器本机或CentOS真机上查询www.tedu.cn时，结果为 1.2.3.4：

```
[root@svr7 ~]# host  www.tedu.cn 192.168.4.7
Using domain server:
Name: 192.168.4.7
Address: 192.168.4.7#53
Aliases:

www.tedu.cn has address 1.2.3.4
```

2 案例2：查看进程信息

2.1 问题

本例要求掌握查看进程信息的操作，使用必要的命令工具完成下列任务：

1. 找出进程 gdm 的 PID 编号值
2. 列出由进程 gdm 开始的子进程树结构信息
3. 找出进程 sshd 的父进程的 PID 编号/进程名称
4. 查看当前系统的CPU负载/进程总量信息

2.2 方案

查看进程的主要命令工具：

- ps aux、ps –elf：查看进程静态快照
- top：查看进程动态排名
- pstree：查看进程与进程之间的树型关系结构
- pgrep：根据指定的名称或条件检索进程

2.3 步骤

实现此案例需要按照如下步骤进行。 Top

步骤一：找出进程 gdm 的 PID 编号值

使用pgrep命令查询指定名称的进程，选项-l显示PID号、-x精确匹配进程名：

```
[root@svr7 ~]# pgrep -lx gdm
1584 gdm
```

步骤二：列出由进程 gdm 开始的子进程树结构信息

使用pstree命令，可以提供用户名或PID值作为参数。通过前一步已知进程gdm的PID为1584，因此以下操作可列出进程gdm的进程树结构：

```
[root@svr7 ~]# pstree -p 1584
gdm(1584)-+-Xorg(1703)
          |-gdm-session-wor(2670)-+-gnome-session(2779)-+-gnom+
          |                       |                     |-gnom+
          |                       |                     |-{gno+
          |                       |                     |-{gno+
          |                       |                     `-{gno+
          |                       |-{gdm-session-wor}(2678)
          |                       `-{gdm-session-wor}(2682)
          |-{gdm}(1668)
          |-{gdm}(1671)
          `-{gdm}(1702)
```

步骤三：找出进程 sshd 的父进程的 PID 编号/进程名称

要查看进程的父进程PID，可以使用ps –elf命令，简单grep过滤即可。找到进程sshd所在行对应到的PPID值即为其父进程的PID编号。为了方便直观查看，建议先列出ps表头行，以分号隔开再执行过滤操作。

```
[root@svr7 ~]# ps -elf | head -1; ps -elf | grep sshd
F S UID        PID  PPID  C PRI  NI ADDR SZ WCHAN  STIME TTY          TIME CMD
4 S root      1362     1  0  80   0 - 20636 poll_s Jan05 ?        00:00:00 /usr/sbin/sshd -D
...                          //可获知进程sshd的父进程PID为1
```

然后再根据pstree –p的结果过滤，可获知PID为1的进程名称为systemd：

```
[root@svr7 ~]# pstree -p | grep '(1)'
systemd(1)-+-ModemManager(995)-+-{ModemManager}(1018)
```

Top

步骤四：查看当前系统的CPU负载/进程总量信息

使用top命令，直接看开头部分即可；或者 top -n 次数：

```
[root@svr7 ~]#top
top - 15:45:25 up 23:55,  2 users,  load average: 0.02, 0.03, 0.05
Tasks: 485 total,   2 running, 483 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1.7 us,  1.0 sy,  0.0 ni, 97.3 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :  1001332 total,    76120 free,   419028 used,   506184 buff/cache
KiB Swap:  2097148 total,  2096012 free,     1136 used.   372288 avail Mem
... ...
```

观察Tasks: 485 total部分，表示进程总量信息。

观察load average: 0.02, 0.03, 0.05 部分，表示CPU处理器在最近1分钟、5分钟、15分钟内的平均处理请求数（对于多核CPU，此数量应除以核心数）。

对于多核CPU主机，如果要分别显示每颗CPU核心的占用情况，可以在top界面按数字键1进行切换：

```
[root@svr7 ~]#top
top - 15:47:45 up 23:57,  2 users,  load average: 0.02, 0.03, 0.05
Tasks: 485 total,   2 running, 269 sleeping,   0 stopped,   1 zombie
Cpu0  :  0.6%us,  7.8%sy,  0.0%ni, 91.6%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu1  :  0.7%us,  3.7%sy,  0.0%ni, 95.6%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu2  :  0.7%us,  1.7%sy,  0.0%ni, 97.6%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu3  :  0.3%us,  1.0%sy,  0.0%ni, 98.3%id,  0.3%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  16230564k total, 15716576k used,   513988k free,   326124k buffers
Swap:  8388604k total,   220656k used,  8167948k free, 11275304k cached
... ...
```

3 案例3：进程调度及终止

3.1 问题

本例要求掌握调度及终止进程的操作，使用必要的工具完成下列任务：

1. 运行“sleep 600”命令，再另开一个终端，查出sleep程序的PID并杀死
2. 运行多个vim程序并都放入后台，然后杀死所有vim进程
3. su切换为zhsan用户，再另开一个终端，强制踢出zhsan用户

3.2 方案

Top

进程调度及终止的主要命令工具：

- 命令行 &：将命令行在后台运行

- Ctrl + z 组合键：挂起当前进程（暂停并转入后台）
- jobs：列出当前用户当前终端的后台任务
- bg 编号：启动指定编号的后台任务
- fg 编号：将指定编号的后台任务调入前台运行
- kill [-9] PID...：杀死指定PID值的进程
- kill [-9] %n：杀死第n个后台任务
- killall [-9] 进程名...：杀死指定名称的所有进程
- pkill：根据指定的名称或条件杀死进程

3.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：根据PID杀死进程

1）开启sleep测试进程

```
[root@svr7 ~]# sleep 600
//.. .. 进入600秒等待状态
```

2）找出进程sleep的PID

另开一个终端，ps aux并过滤进程信息（第2列为PID值）：

```
[root@svr7 ~]# ps aux | grep sleep
root      32929  0.0  0.0   4312   360 pts/1    S+   17:25   0:00 sleep 600
```

3）杀死指定PID的进程

```
[root@svr7 ~]# kill -9 32929
```

返回原终端会发现sleep进程已经被杀死：

```
[root@svr7 ~]# sleep 600
Killed
```

步骤二：根据进程名杀死多个进程

1）在后台开启多个vim进程

Top

```
[root@svr7 ~]# vim a.txt &
```

```
[1] 33152
[root@svr7 ~]# vim b.txt &
[2] 33154
[1]+ 已停止               vim a.txt
[root@svr7 ~]# vim c.txt &
[3] 33155
[2]+ 已停止               vim b.txt
```

2）确认vim进程信息

```
[root@svr7 ~]# jobs -l
[1]  33152 停止 (tty 输出)      vim a.txt
[2]- 33154 停止 (tty 输出)      vim b.txt
[3]+ 33155 停止 (tty 输出)      vim c.txt
```

3）强制杀死所有名为vim的进程

```
[root@svr7 ~]# killall -9 vim
[1]  已杀死               vim a.txt
[2]- 已杀死               vim b.txt
[3]+ 已杀死               vim c.txt
```

4）确认杀进程结果

```
[root@svr7 ~]# jobs -l
[root@svr7 ~]#
```

步骤三：杀死属于指定用户的所有进程

1）登入测试用户zhsan

```
[root@svr7 ~]# useradd zhsan
[root@svr7 ~]# su - zhsan
[zhsan@svr7 ~]$
```

Top

2）另开一个终端，以root用户登入，查找属于用户zhsan的进程

```
[root@svr7 ~]# pgrep -u zhsan
33219
[root@svr7 ~]# pstree -up 33219                    //检查进程树
bash(33219,zhsan)
```

3）强制杀死属于用户zhsan的进程

```
[root@svr7 ~]# pkill -9 -u zhsan
[root@svr7 ~]#
```

4）返回原来用户zhsan登录的终端，确认已经被终止

```
[zhsan@svr7 ~]$ 已杀死
[root@svr7 ~]#
```

4 案例4：系统日志分析

4.1 问题

本例要求熟悉Linux系统中的常见日志文件，使用必要的命令工具完成下列任务：

1. 列出所有包含关键词8909的系统日志消息
2. 查看启动时识别的鼠标设备信息
3. 列出最近2条成功/不成功的用户登录消息
4. 列出最近10条重要程度在 ERR 及以上的日志消息
5. 列出所有与服务httpd相关的消息
6. 列出前4个小时内新记录的日志

4.2 方案

常见的系统日志及各自用途：

- /var/log/messages，记录内核消息、各种服务的公共消息
- /var/log/dmesg，记录系统启动过程的各种消息
- /var/log/cron，记录与cron计划任务相关的消息
- /var/log/maillog，记录邮件收发相关的消息
- /var/log/secure，记录与访问限制相关的安全消息

日志消息的优先级（高-->低）：

- EMERG（紧急）：级别0，系统不可用的情况
- ALERT（警报）：级别1，必须马上采取措施的情况
- CRIT（严重）：级别2，严重情形
- ERR（错误）：级别3，出现错误

Top

- WARNING（警告）：级别4，值得警告的情形
- NOTICE（注意）：级别5，普通但值得引起注意的事件
- INFO（信息）：级别6，一般信息
- DEBUG（调试）：级别7，程序/服务调试消息

RHEL7提供的journalctl日志工具的常见用法：

- journalctl | grep 关键词
- journalctl -u 服务名 -p 优先级
- journalctl -n 消息条数
- journalctl --since="yyyy-mm-dd HH:MM:SS" --until="yyyy-mm-dd HH:MM:SS"

4.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：分析系统日志及用户日志

1）列出所有包含关键词8909的系统日志消息

简单模拟一个故障（SELinux阻止Web开放8909端口）：

```
[root@svr7 ~]# vim /etc/httpd/conf.d/8909.conf          //添加开8909端口配置
Listen 8909
[root@svr7 ~]# setenforce 1                             //开启强制模式
[root@svr7 ~]# systemctl restart httpd                  //起服务失败
Job for httpd.service failed because the control process exited with error code. See "syste
```

从日志文件/var/log/messages中检索信息：

```
[root@svr7 ~]# grep 8909 /var/log/messages
Jan  6 17:53:48 svr7 setroubleshoot: SELinux is preventing /usr/sbin/httpd from name_bin
Jan  6 17:53:48 svr7 python: SELinux is preventing /usr/sbin/httpd from name_bind access
.. ..
```

使用完毕记得删除测试配置文件：

```
[root@svr7 ~]# rm -rf /etc/httpd/conf.d/8909.conf
[root@svr7 ~]# systemctl restart httpd
```

Top

2）查看启动时识别的鼠标设备信息

```
[root@svr7 ~]# dmesg | grep -i mouse
[    1.020385] mousedev: PS/2 mouse device common for all mice
[    1.249422] input: ImPS/2 Generic Wheel Mouse as /devices/platform/i8042/serio1/inpu
[    2.279665] usb 2-1: Product: VMware Virtual USB Mouse
[    2.603999] input: VMware VMware Virtual USB Mouse as /devices/pci0000:00/0000:00:
[    2.604222] hid-generic 0003:0E0F:0003.0001: input,hidraw0: USB HID v1.10 Mouse [VM
```

3）列出最近2条成功/不成功的用户登录消息

查看成功登录的事件消息：

```
[root@svr7 ~]# last -2
zhsan    pts/2        192.168.4.207    Fri Jan  6 18:00 -  18:00  (00:00)
root     pts/2        192.168.4.110    Fri Jan  6 17:26 -  17:59  (00:33)

wtmp begins Thu Aug  4 00:10:16 2016
```

查看失败登录的事件消息：

```
[root@svr7 ~]# lastb -2
anonymou ssh:notty    192.168.4.207    Fri Jan  6 18:00 -  18:00  (00:00)
anonymou ssh:notty    192.168.4.207    Fri Jan  6 18:00 -  18:00  (00:00)

btmp begins Fri Jan  6 18:00:34 2017
```

步骤二：使用journalctl日志提取工具

1）列出最近10条重要程度在 ERR 及以上的日志消息

```
[root@svr7 ~]# journalctl -p err -n 10
-- Logs begin at Thu 2017-01-05 15:50:08 CST, end at Fri 2017-01-06 18:01:01 CST. --
Jan 06 14:56:57 svr7 setroubleshoot[23702]: SELinux is preventing /usr/sbin/vsftpd from
Jan 06 14:56:57 svr7 setroubleshoot[23702]: SELinux is preventing /usr/sbin/vsftpd from
Jan 06 14:56:57 svr7 setroubleshoot[23702]: SELinux is preventing /usr/sbin/vsftpd from
Jan 06 14:56:57 svr7 setroubleshoot[23702]: SELinux is preventing /usr/sbin/vsftpd from
Jan 06 17:53:48 svr7 setroubleshoot[33743]: Plugin Exception restorecon_source
Jan 06 17:53:48 svr7 setroubleshoot[33743]: SELinux is preventing /usr/sbin/httpd from
Jan 06 17:53:53 svr7 setroubleshoot[33743]: SELinux is preventing /usr/sbin/httpd from
```

Top

```
10.  Jan 06 17:53:54 svr7 systemd[1]: Failed to start The Apache HTTP Server.
11.  ...
12.  lines 1-11/11 (END)
```

2）列出所有与服务httpd相关的消息

```
01.  [root@svr7 ~]#journalctl -u httpd
02.  -- Logs begin at Thu 2017-01-05 15:50:08 CST, end at Fri 2017-01-06 18:01:01 CST. --
03.  Jan 06 14:57:16 svr7 systemd[1]: Starting The Apache HTTP Server...
04.  Jan 06 14:57:16 svr7 httpd[23812]: AH00557: httpd: apr_sockaddr_info_get() failed for
05.  Jan 06 14:57:16 svr7 httpd[23812]: AH00558: httpd: Could not reliably determine the serv
06.  Jan 06 14:57:16 svr7 systemd[1]: Started The Apache HTTP Server.
07.  Jan 06 17:53:44 svr7 systemd[1]: Stopping The Apache HTTP Server...
08.  Jan 06 17:53:46 svr7 systemd[1]: Starting The Apache HTTP Server...
09.  Jan 06 17:53:46 svr7 httpd[33741]: AH00557: httpd: apr_sockaddr_info_get() failed for
10.  ...
```

3）列出前4个小时内新记录的日志

根据当前日期时间往前推4个小时，确定--since起始和--until结束时刻:

```
01.  [root@svr7 ~]#journalctl --since "2017-01-06 14:11" --until "2017-01-06 18:11"
02.  -- Logs begin at Thu 2017-01-05 15:50:08 CST, end at Fri 2017-01-06 18:10:01 CST. --
03.  Jan 06 14:20:01 svr7 systemd[1]: Started Session 160 of user root.
04.  Jan 06 14:20:01 svr7 CROND[22869]: (root) CMD (/usr/lib64/sa/sa1 1 1)
05.  Jan 06 14:20:01 svr7 systemd[1]: Starting Session 160 of user root.
06.  Jan 06 14:30:01 svr7 systemd[1]: Started Session 161 of user root.
07.  Jan 06 14:30:01 svr7 CROND[23028]: (root) CMD (/usr/lib64/sa/sa1 1 1)
08.  Jan 06 14:31:39 svr7 systemd[1]: Starting Session 162 of user root.
09.  Jan 06 14:32:17 svr7 sshd[23046]: pam_unix(sshd:session): session closed for user root
10.  Jan 06 14:31:39 svr7 systemd[1]: Started Session 162 of user root.
11.  Jan 06 14:31:39 svr7 sshd[23046]: pam_unix(sshd:session): session opened for user root
12.  Jan 06 14:31:39 svr7 systemd-logind[985]: New session 162 of user root.
13.  ...
```

Top