[Top](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/SERVICES/DAY04/CASE/01/index.html" \l "page_top_case)

# NSD SERVICES DAY04

1. [案例1：配置并验证Split分离解析](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/SERVICES/DAY04/CASE/01/index.html" \l "case1)
2. [案例2：查看进程信息](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/SERVICES/DAY04/CASE/01/index.html" \l "case2)
3. [案例3：进程调度及终止](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/SERVICES/DAY04/CASE/01/index.html" \l "case3)
4. [案例4：系统日志分析](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/SERVICES/DAY04/CASE/01/index.html" \l "case4)
5. [案例5：使用systemctl工具](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/SERVICES/DAY04/CASE/01/index.html" \l "case5)

## 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：

1. view "视图1" {
2. match-clients { 客户机地址1; .. .. ; };         //匹配第1类客户机地址
3. zone "目标域名" IN {                             //同一个DNS区域
4. type master;
5. file "地址库1";                             //第1份地址库
6. };
7. };
8. view "视图2" {
9. match-clients { 客户机地址2; .. .. ; };         //匹配第2类客户机地址
10. match-clients { any; };                         //匹配任意地址
11. zone "目标域名" IN {                             //同一个DNS区域
12. type master;
13. file "地址库2";                             //第2份地址库
14. };
15. };
16. .. ..
17. view "视图n" {
18. match-clients { any; };                         //匹配任意地址
19. zone "目标域名" IN {                             //同一个DNS区域
20. type master;
21. file "地址库n";                             //第n份地址库
22. };
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：

1. [root@svr7 ~]# cd /var/named/
2. [root@svr7 named]# cp -p tedu.cn.zone tedu.cn.zone.lan
3. [root@svr7 named]# vim tedu.cn.zone.lan
4. $TTL 1D
5. @ IN SOA @ rname.invalid. (
6. 0 ; serial
7. 1D ; refresh
8. 1H ; retry
9. 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：

1. [root@svr7 named]# cp -p tedu.cn.zone tedu.cn.zone.other
2. [root@svr7 named]# vim tedu.cn.zone.other
3. $TTL 1D
4. @ IN SOA @ rname.invalid. (
5. 0 ; serial
6. 1D ; refresh
7. 1H ; retry
8. 1W ; expire
9. 3H ) ; minimum
10. @ NS svr7.tedu.cn.
11. svr7 A 192.168.4.7
12. pc207 A 192.168.4.207
13. 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

1. [root@svr7 ~]# vim /etc/named.conf
2. options {
3. directory "/var/named";
4. };
5. acl "mylan" {                                     //名为mylan的列表
6. 192.168.4.207; 192.168.7.0/24;
7. };
8. .. ..
9. view "mylan" {
10. match-clients { mylan; };                     //检查客户机地址是否匹配此列表
11. zone "tedu.cn" IN {
12. type master;
13. file "tedu.cn.zone.lan";
14. };
15. };
16. view "other" {
17. match-clients { any; };                         //匹配任意客户机地址
18. zone "tedu.cn" IN {
19. type master;
20. file "tedu.cn.zone.other";
21. };
22. };

3）重启named服务

1. [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：

1. [root@pc207 ~]# host www.tedu.cn 192.168.4.7
2. Using domain server:
3. Name: 192.168.4.7
4. Address: 192.168.4.7#53
5. Aliases:
6. 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：

1. [root@svr7 ~]# host www.tedu.cn 192.168.4.7
2. Using domain server:
3. Name: 192.168.4.7
4. Address: 192.168.4.7#53
5. Aliases:
6. 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。

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：

1. [root@svr7 ~]# pgrep -lx gdm
2. 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：

1. [root@svr7 ~]# pstree -p 1584
2. gdm(1584)-+-Xorg(1703)
3. |-gdm-session-wor(2670)-+-gnome-session(2779)-+-gnom+
4. | | |-gnom+
5. | | |-{gno+
6. | | |-{gno+
7. | | `-{gno+
8. | |-{gdm-session-wor}(2678)
9. | `-{gdm-session-wor}(2682)
10. |-{gdm}(1668)
11. |-{gdm}(1671)
12. `-{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。

1. [root@svr7 ~]# ps -elf | head -1 ; ps -elf | grep sshd
2. F S UID PID PPID C PRI NI ADDR SZ WCHAN STIME TTY TIME CMD
3. 4 S root 1362 1 0 80 0 - 20636 poll\_s Jan05 ? 00:00:00 /usr/sbin/sshd –D
4. .. ..                                 //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：

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

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：

1. [root@svr7 ~]# top
2. top - 15:45:25 up 23:55, 2 users, load average: 0.02, 0.03, 0.05
3. Tasks: 485 total, 2 running, 483 sleeping, 0 stopped, 0 zombie
4. %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
5. KiB Mem : 1001332 total, 76120 free, 419028 used, 506184 buff/cache
6. KiB Swap: 2097148 total, 2096012 free, 1136 used. 372288 avail Mem
7. .. ..

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：

1. [root@svr7 ~]# top
2. top - 15:47:45 up 23:57, 2 users, load average: 0.02, 0.03, 0.05
3. Tasks: 485 total, 2 running, 269 sleeping, 0 stopped, 1 zombie
4. 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
5. 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
6. 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
7. 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
8. Mem: 16230564k total, 15716576k used, 513988k free, 326124k buffers
9. Swap: 8388604k total, 220656k used, 8167948k free, 11275304k cached
10. .. ..

## 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

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

1. [root@svr7 ~]# sleep 600
2. //.. .. 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）：

1. [root@svr7 ~]# ps aux | grep sleep
2. 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

1. [root@svr7 ~]# kill -9 32929

返回原终端会发现sleep进程已经被杀死：

1. [root@svr7 ~]# sleep 600
2. 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

1. [root@svr7 ~]# vim a.txt &
2. [1] 33152
3. [root@svr7 ~]# vim b.txt &
4. [2] 33154
5. [1]+ 已停止 vim a.txt
6. [root@svr7 ~]# vim c.txt &
7. [3] 33155
8. [2]+ 已停止 vim b.txt

2）确认vim进程信息

1. [root@svr7 ~]# jobs -l
2. [1] 33152 停止 (tty 输出) vim a.txt
3. [2]- 33154 停止 (tty 输出) vim b.txt
4. [3]+ 33155 停止 (tty 输出) vim c.txt

3）强制杀死所有名为vim的进程

1. [root@svr7 ~]# killall -9 vim
2. [1] 已杀死 vim a.txt
3. [2]- 已杀死 vim b.txt
4. [3]+ 已杀死 vim c.txt

4）确认杀进程结果

1. [root@svr7 ~]# jobs -l
2. [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

1. [root@svr7 ~]# useradd zhsan
2. [root@svr7 ~]# su - zhsan
3. [zhsan@svr7 ~]$

2）另开一个终端，以root用户登入，查找属于用户zhsan的进程

1. [root@svr7 ~]# pgrep -u zhsan
2. 33219
3. [root@svr7 ~]# pstree -up 33219                             //检查进程树
4. bash(33219,zhsan)

3）强制杀死属于用户zhsan的进程

1. [root@svr7 ~]# pkill -9 -u zhsan
2. [root@svr7 ~]#

4）返回原来用户zhsan登录的终端，确认已经被终止

1. [zhsan@svr7 ~]$ 已杀死
2. [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
* 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）：

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

从日志文件/var/log/messages中检索信息：

1. [root@svr7 ~]# grep 8909 /var/log/messages
2. Jan 6 17:53:48 svr7 setroubleshoot: SELinux is preventing /usr/sbin/httpd from name\_bind access on the tcp\_socket port 8909. For complete SELinux messages. run sealert -l 6d37b8f0-ab8a-4082-9295-c784f4f57190
3. Jan 6 17:53:48 svr7 python: SELinux is preventing /usr/sbin/httpd from name\_bind access on the tcp\_socket port 8909.#012#012\*\*\*\*\* Plugin bind\_ports (92.2 confidence) suggests \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#012#012If you want to allow /usr/sbin/httpd to bind to network port 8909#012Then you need to modify the port type.#012Do#012# semanage port -a -t PORT\_TYPE -p tcp 8909#012 where PORT\_TYPE is one of the following: http\_cache\_port\_t, http\_port\_t, jboss\_management\_port\_t, jboss\_messaging\_port\_t, ntop\_port\_t, puppet\_port\_t.#012#012\*\*\*\*\* Plugin catchall\_boolean (7.83 confidence) suggests \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#012#012If you want to allow nis to enabled#012Then you must tell SELinux about this by enabling the 'nis\_enabled' boolean.#012#012Do#012setsebool -P nis\_enabled 1#012#012\*\*\*\*\* Plugin catchall (1.41 confidence) suggests \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#012#012If you believe that httpd should be allowed name\_bind access on the port 8909 tcp\_socket by default.#012Then you should report this as a bug.#012You can generate a local policy module to allow this access.#012Do#012allow this access for now by executing:#012# grep httpd /var/log/audit/audit.log | audit2allow -M mypol#012# semodule -i mypol.pp#012
4. .. ..

使用完毕记得删除测试配置文件：

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

2）查看启动时识别的鼠标设备信息

1. [root@svr7 ~]# dmesg | grep -i mouse
2. [ 1.020385] mousedev: PS/2 mouse device common for all mice
3. [ 1.249422] input: ImPS/2 Generic Wheel Mouse as /devices/platform/i8042/serio1/input/input2
4. [ 2.279665] usb 2-1: Product: VMware Virtual USB Mouse
5. [ 2.603999] input: VMware VMware Virtual USB Mouse as /devices/pci0000:00/0000:00:11.0/0000:02:00.0/usb2/2-1/2-1:1.0/input/input3
6. [ 2.604222] hid-generic 0003:0E0F:0003.0001: input,hidraw0: USB HID v1.10 Mouse [VMware VMware Virtual USB Mouse] on usb-0000:02:00.0-1/input0

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：

1. [root@svr7 ~]# last -2
2. zhsan pts/2 192.168.4.207 Fri Jan 6 18:00 - 18:00 (00:00)
3. root pts/2 192.168.4.110 Fri Jan 6 17:26 - 17:59 (00:33)
4. wtmp begins Thu Aug 4 00:10:16 2016

查看失败登录的事件消息：

1. [root@svr7 ~]# lastb -2
2. anonymou ssh:notty 192.168.4.207 Fri Jan 6 18:00 - 18:00 (00:00)
3. anonymou ssh:notty 192.168.4.207 Fri Jan 6 18:00 - 18:00 (00:00)
4. 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

1. [root@svr7 ~]# journalctl -p err -n 10
2. -- Logs begin at Thu 2017-01-05 15:50:08 CST, end at Fri 2017-01-06 18:01:01 CST. --
3. Jan 06 14:56:57 svr7 setroubleshoot[23702]: SELinux is preventing /usr/sbin/vsftpd from getattr access on the file /rhel7/repodata/repomd.xml. For complete SELinux mes
4. Jan 06 14:56:57 svr7 setroubleshoot[23702]: SELinux is preventing /usr/sbin/vsftpd from read access on the file repomd.xml. For complete SELinux messages. run sealert
5. Jan 06 14:56:57 svr7 setroubleshoot[23702]: SELinux is preventing /usr/sbin/vsftpd from read access on the file repomd.xml. For complete SELinux messages. run sealert
6. Jan 06 14:56:57 svr7 setroubleshoot[23702]: SELinux is preventing /usr/sbin/vsftpd from lock access on the file /rhel7/repodata/repomd.xml. For complete SELinux messag
7. Jan 06 17:53:48 svr7 setroubleshoot[33743]: Plugin Exception restorecon\_source
8. Jan 06 17:53:48 svr7 setroubleshoot[33743]: SELinux is preventing /usr/sbin/httpd from name\_bind access on the tcp\_socket port 8909. For complete SELinux messages. run
9. Jan 06 17:53:53 svr7 setroubleshoot[33743]: SELinux is preventing /usr/sbin/httpd from name\_connect access on the tcp\_socket port 8909. For complete SELinux messages.
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

1. [root@svr7 ~]# journalctl -u httpd
2. -- Logs begin at Thu 2017-01-05 15:50:08 CST, end at Fri 2017-01-06 18:01:01 CST. --
3. Jan 06 14:57:16 svr7 systemd[1]: Starting The Apache HTTP Server...
4. Jan 06 14:57:16 svr7 httpd[23812]: AH00557: httpd: apr\_sockaddr\_info\_get() failed for svr7
5. Jan 06 14:57:16 svr7 httpd[23812]: AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using 127.0.0.1. Set the 'ServerName' directi
6. Jan 06 14:57:16 svr7 systemd[1]: Started The Apache HTTP Server.
7. Jan 06 17:53:44 svr7 systemd[1]: Stopping The Apache HTTP Server...
8. Jan 06 17:53:46 svr7 systemd[1]: Starting The Apache HTTP Server...
9. Jan 06 17:53:46 svr7 httpd[33741]: AH00557: httpd: apr\_sockaddr\_info\_get() failed for svr7
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:

1. [root@svr7 ~]# journalctl --since "2017-01-06 14:11" --until "2017-01-06 18:11"
2. -- Logs begin at Thu 2017-01-05 15:50:08 CST, end at Fri 2017-01-06 18:10:01 CST. --
3. Jan 06 14:20:01 svr7 systemd[1]: Started Session 160 of user root.
4. Jan 06 14:20:01 svr7 CROND[22869]: (root) CMD (/usr/lib64/sa/sa1 1 1)
5. Jan 06 14:20:01 svr7 systemd[1]: Starting Session 160 of user root.
6. Jan 06 14:30:01 svr7 systemd[1]: Started Session 161 of user root.
7. Jan 06 14:30:01 svr7 CROND[23028]: (root) CMD (/usr/lib64/sa/sa1 1 1)
8. Jan 06 14:31:39 svr7 systemd[1]: Starting Session 162 of user root.
9. 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 by (uid=0)
12. Jan 06 14:31:39 svr7 systemd-logind[985]: New session 162 of user root.
13. .. .

## 5 案例5：使用systemctl工具

### 5.1 问题

本例要求掌握systemctl控制工具的基本操作，完成下列任务：

1. 重启 httpd、crond、bluetooth 服务，查看状态
2. 禁止 bluetooth 服务开机自启，并停用此服务
3. 设置默认级别为 multi-user.target 并确认

### 5.2 方案

systemd是一个更高效的系统&服务管理器，其相关特性如下：

* 开机服务并行启动，各系统服务间的精确依赖
* 配置目录：/etc/systemd/system/
* 服务目录：/lib/systemd/system/

systemctl是systemd的管理工具，将相关资源组织为unit配置单元进行管理。

不同的unit决定了一组相关的启动任务，service和target是最常用的配置单元：

* service：后台独立服务
* target：一套配置单元的组合，类似于传统“运行级别”

### 5.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：重启 httpd、crond、bluetooth 服务，查看状态

1）重启系统服务httpd、crond、bluetooth

1. [root@svr7 ~]# systemctl restart httpd crond bluetooth

2）查看上述服务的状态

1. [root@svr7 ~]# systemctl status httpd crond bluetooth
2. \* httpd.service - The Apache HTTP Server
3. Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
4. Active: active (running) since Fri 2017-01-06 18:18:20 CST; 18s ago
5. .. ..
6. \* crond.service - Command Scheduler
7. Loaded: loaded (/usr/lib/systemd/system/crond.service; enabled; vendor preset: enabled)
8. Active: active (running) since Fri 2017-01-06 18:18:19 CST; 19s ago
9. .. ..
10. \* bluetooth.service - Bluetooth service
11. Loaded: loaded (/usr/lib/systemd/system/bluetooth.service; enabled; vendor preset: enabled)
12. Active: active (running) since Fri 2017-01-06 18:18:19 CST; 19s ago
13. .. ..

步骤二：禁止 bluetooth 服务开机自启，并停用此服务

1）停用bluetooth服务

1. [root@svr7 ~]# systemctl stop bluetooth

2）禁止bluetooth服务开机自启

1. [root@svr7 ~]# systemctl disable bluetooth
2. Removed symlink /etc/systemd/system/dbus-org.bluez.service.
3. Removed symlink /etc/systemd/system/bluetooth.target.wants/bluetooth.service.
4. [root@svr7 ~]# systemctl is-enabled Bluetooth             //检查结果
5. disabled

步骤三：设置默认级别为 multi-user.target 并确认

1）查看默认运行级别

1. [root@svr7 ~]# systemctl get-default
2. graphical.target

2）将默认运行级别设置为multi-user.target

1. [root@svr7 ~]# systemctl set-default multi-user.target
2. Removed symlink /etc/systemd/system/default.target.
3. Created symlink from /etc/systemd/system/default.target to /usr/lib/systemd/system/multi-user.target.

3）确认配置结果

1. [root@svr7 ~]# systemctl get-default
2. multi-user.target

根据此处的设置，重启此虚拟机后图形桌面将不再可用。