# Systemd 入门教程：实战篇

一、开机启动

对于那些支持 Systemd 的软件，安装的时候，会自动在/usr/lib/systemd/system目录添加一个配置文件。

如果你想让该软件开机启动，就执行下面的命令（以httpd.service为例）。

$ sudo systemctl enable httpd

上面的命令相当于在/etc/systemd/system目录添加一个符号链接，指向/usr/lib/systemd/system里面的httpd.service文件。

这是因为开机时，Systemd只执行/etc/systemd/system目录里面的配置文件。这也意味着，如果把修改后的配置文件放在该目录，就可以达到覆盖原始配置的效果。

二、启动服务

设置开机启动以后，软件并不会立即启动，必须等到下一次开机。如果想现在就运行该软件，那么要执行systemctl start命令。

$ sudo systemctl start httpd

执行上面的命令以后，有可能启动失败，因此要用systemctl status命令查看一下该服务的状态。

$ sudo systemctl status httpd

httpd.service - The Apache HTTP Server

Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled)

Active: active (running) since 金 2014-12-05 12:18:22 JST; 7min ago

Main PID: 4349 (httpd)

Status: "Total requests: 1; Current requests/sec: 0; Current traffic: 0 B/sec"

CGroup: /system.slice/httpd.service

├─4349 /usr/sbin/httpd -DFOREGROUND

├─4350 /usr/sbin/httpd -DFOREGROUND

├─4351 /usr/sbin/httpd -DFOREGROUND

├─4352 /usr/sbin/httpd -DFOREGROUND

├─4353 /usr/sbin/httpd -DFOREGROUND

└─4354 /usr/sbin/httpd -DFOREGROUND

12月 05 12:18:22 localhost.localdomain systemd[1]: Starting The Apache HTTP Server...

12月 05 12:18:22 localhost.localdomain systemd[1]: Started The Apache HTTP Server.

12月 05 12:22:40 localhost.localdomain systemd[1]: Started The Apache HTTP Server.

上面的输出结果含义如下。

* Loaded行：配置文件的位置，是否设为开机启动
* Active行：表示正在运行
* Main PID行：主进程ID
* Status行：由应用本身（这里是 httpd ）提供的软件当前状态
* CGroup块：应用的所有子进程
* 日志块：应用的日志

三、停止服务

终止正在运行的服务，需要执行systemctl stop命令。

$ sudo systemctl stop httpd.service

有时候，该命令可能没有响应，服务停不下来。这时候就不得不"杀进程"了，向正在运行的进程发出kill信号。

$ sudo systemctl kill httpd.service

此外，重启服务要执行systemctl restart命令。

$ sudo systemctl restart httpd.service

四、读懂配置文件

一个服务怎么启动，完全由它的配置文件决定。下面就来看，配置文件有些什么内容。

前面说过，配置文件主要放在/usr/lib/systemd/system目录，也可能在/etc/systemd/system目录。找到配置文件以后，使用文本编辑器打开即可。

systemctl cat命令可以用来查看配置文件，下面以sshd.service文件为例，它的作用是启动一个 SSH 服务器，供其他用户以 SSH 方式登录。

$ systemctl cat sshd.service

[Unit]

Description=OpenSSH server daemon

Documentation=man:sshd(8) man:sshd\_config(5)

After=network.target sshd-keygen.service

Wants=sshd-keygen.service

[Service]

EnvironmentFile=/etc/sysconfig/sshd

ExecStart=/usr/sbin/sshd -D $OPTIONS

ExecReload=/bin/kill -HUP $MAINPID

Type=simple

KillMode=process

Restart=on-failure

RestartSec=42s

[Install]

WantedBy=multi-user.target

可以看到，配置文件分成几个区块，每个区块包含若干条键值对。

下面依次解释每个区块的内容。

五、 [Unit] 区块：启动顺序与依赖关系。

Unit区块的Description字段给出当前服务的简单描述，Documentation字段给出文档位置。

接下来的设置是启动顺序和依赖关系，这个比较重要。

After字段：表示如果network.target或sshd-keygen.service需要启动，那么sshd.service应该在它们之后启动。

相应地，还有一个Before字段，定义sshd.service应该在哪些服务之前启动。

注意，After和Before字段只涉及启动顺序，不涉及依赖关系。

举例来说，某 Web 应用需要 postgresql 数据库储存数据。在配置文件中，它只定义要在 postgresql 之后启动，而没有定义依赖 postgresql 。上线后，由于某种原因，postgresql 需要重新启动，在停止服务期间，该 Web 应用就会无法建立数据库连接。

设置依赖关系，需要使用Wants字段和Requires字段。

Wants字段：表示sshd.service与sshd-keygen.service之间存在"弱依赖"关系，即如果"sshd-keygen.service"启动失败或停止运行，不影响sshd.service继续执行。

Requires字段则表示"强依赖"关系，即如果该服务启动失败或异常退出，那么sshd.service也必须退出。

注意，Wants字段与Requires字段只涉及依赖关系，与启动顺序无关，默认情况下是同时启动的。

六、[Service] 区块：启动行为

Service区块定义如何启动当前服务。

6.1 启动命令

许多软件都有自己的环境参数文件，该文件可以用EnvironmentFile字段读取。

EnvironmentFile字段：指定当前服务的环境参数文件。该文件内部的key=value键值对，可以用$key的形式，在当前配置文件中获取。

上面的例子中，sshd 的环境参数文件是/etc/sysconfig/sshd。

配置文件里面最重要的字段是ExecStart。

ExecStart字段：定义启动进程时执行的命令。

上面的例子中，启动sshd，执行的命令是/usr/sbin/sshd -D $OPTIONS，其中的变量$OPTIONS就来自EnvironmentFile字段指定的环境参数文件。

与之作用相似的，还有如下这些字段。

* ExecReload字段：重启服务时执行的命令
* ExecStop字段：停止服务时执行的命令
* ExecStartPre字段：启动服务之前执行的命令
* ExecStartPost字段：启动服务之后执行的命令
* ExecStopPost字段：停止服务之后执行的命令

请看下面的例子。

[Service]

ExecStart=/bin/echo execstart1

ExecStart=

ExecStart=/bin/echo execstart2

ExecStartPost=/bin/echo post1

ExecStartPost=/bin/echo post2

上面这个配置文件，第二行ExecStart设为空值，等于取消了第一行的设置，运行结果如下。

execstart2

post1

post2

所有的启动设置之前，都可以加上一个连词号（-），表示"抑制错误"，即发生错误的时候，不影响其他命令的执行。比如，EnvironmentFile=-/etc/sysconfig/sshd（注意等号后面的那个连词号），就表示即使/etc/sysconfig/sshd文件不存在，也不会抛出错误。

6.2 启动类型

Type字段定义启动类型。它可以设置的值如下。

* simple（默认值）：ExecStart字段启动的进程为主进程
* forking：ExecStart字段将以fork()方式启动，此时父进程将会退出，子进程将成为主进程
* oneshot：类似于simple，但只执行一次，Systemd 会等它执行完，才启动其他服务
* dbus：类似于simple，但会等待 D-Bus 信号后启动
* notify：类似于simple，启动结束后会发出通知信号，然后 Systemd 再启动其他服务
* idle：类似于simple，但是要等到其他任务都执行完，才会启动该服务。一种使用场合是为让该服务的输出，不与其他服务的输出相混合

下面是一个oneshot的例子，笔记本电脑启动时，要把触摸板关掉，配置文件可以这样写。

[Unit]

Description=Switch-off Touchpad

[Service]

Type=oneshot

ExecStart=/usr/bin/touchpad-off

[Install]

WantedBy=multi-user.target

上面的配置文件，启动类型设为oneshot，就表明这个服务只要运行一次就够了，不需要长期运行。

如果关闭以后，将来某个时候还想打开，配置文件修改如下。

[Unit]

Description=Switch-off Touchpad

[Service]

Type=oneshot

ExecStart=/usr/bin/touchpad-off start

ExecStop=/usr/bin/touchpad-off stop

RemainAfterExit=yes

[Install]

WantedBy=multi-user.target

上面配置文件中，RemainAfterExit字段设为yes，表示进程退出以后，服务仍然保持执行。这样的话，一旦使用systemctl stop命令停止服务，ExecStop指定的命令就会执行，从而重新开启触摸板。

6.3 重启行为

Service区块有一些字段，定义了重启行为。

KillMode字段：定义 Systemd 如何停止 sshd 服务。

上面这个例子中，将KillMode设为process，表示只停止主进程，不停止任何sshd 子进程，即子进程打开的 SSH session 仍然保持连接。这个设置不太常见，但对 sshd 很重要，否则你停止服务的时候，会连自己打开的 SSH session 一起杀掉。

KillMode字段可以设置的值如下。

* control-group（默认值）：当前控制组里面的所有子进程，都会被杀掉
* process：只杀主进程
* mixed：主进程将收到 SIGTERM 信号，子进程收到 SIGKILL 信号
* none：没有进程会被杀掉，只是执行服务的 stop 命令。

接下来是Restart字段。

Restart字段：定义了 sshd 退出后，Systemd 的重启方式。

上面的例子中，Restart设为on-failure，表示任何意外的失败，就将重启sshd。如果 sshd 正常停止（比如执行systemctl stop命令），它就不会重启。

Restart字段可以设置的值如下。

* no（默认值）：退出后不会重启
* on-success：只有正常退出时（退出状态码为0），才会重启
* on-failure：非正常退出时（退出状态码非0），包括被信号终止和超时，才会重启
* on-abnormal：只有被信号终止和超时，才会重启
* on-abort：只有在收到没有捕捉到的信号终止时，才会重启
* on-watchdog：超时退出，才会重启
* always：不管是什么退出原因，总是重启

对于守护进程，推荐设为on-failure。对于那些允许发生错误退出的服务，可以设为on-abnormal。

最后是RestartSec字段。

RestartSec字段：表示 Systemd 重启服务之前，需要等待的秒数。上面的例子设为等待42秒。

七、[Install] 区块

Install区块，定义如何安装这个配置文件，即怎样做到开机启动。

WantedBy字段：表示该服务所在的 Target。

Target的含义是服务组，表示一组服务。WantedBy=multi-user.target指的是，sshd 所在的 Target 是multi-user.target。

这个设置非常重要，因为执行systemctl enable sshd.service命令时，sshd.service的一个符号链接，就会放在/etc/systemd/system目录下面的multi-user.target.wants子目录之中。

Systemd 有默认的启动 Target。

$ systemctl get-default

multi-user.target

上面的结果表示，默认的启动 Target 是multi-user.target。在这个组里的所有服务，都将开机启动。这就是为什么systemctl enable命令能设置开机启动的原因。

使用 Target 的时候，systemctl list-dependencies命令和systemctl isolate命令也很有用。

# 查看 multi-user.target 包含的所有服务

$ systemctl list-dependencies multi-user.target

# 切换到另一个 target

# shutdown.target 就是关机状态

$ sudo systemctl isolate shutdown.target

一般来说，常用的 Target 有两个：一个是multi-user.target，表示多用户命令行状态；另一个是graphical.target，表示图形用户状态，它依赖于multi-user.target。

八、Target 的配置文件

Target 也有自己的配置文件。

$ systemctl cat multi-user.target

[Unit]

Description=Multi-User System

Documentation=man:systemd.special(7)

Requires=basic.target

Conflicts=rescue.service rescue.target

After=basic.target rescue.service rescue.target

AllowIsolate=yes

注意，Target 配置文件里面没有启动命令。

上面输出结果中，主要字段含义如下。

Requires字段：要求basic.target一起运行。

Conflicts字段：冲突字段。如果rescue.service或rescue.target正在运行，multi-user.target就不能运行，反之亦然。

After：表示multi-user.target在basic.target 、 rescue.service、 rescue.target之后启动，如果它们有启动的话。

AllowIsolate：允许使用systemctl isolate命令切换到multi-user.target。

九、修改配置文件后重启

修改配置文件以后，需要重新加载配置文件，然后重新启动相关服务。

# 重新加载配置文件

$ sudo systemctl daemon-reload

# 重启相关服务

$ sudo systemctl restart foobar

（完）