Linux初始化程序

systemd简介

> systemd是Linux下的一款**系统**和**服务**的管理器软件，是系统启动加载内核之后运行的第一支程序，这支程序运行在Linux时的PID是1，其他程序都是这支程序的子程序。目前许多Linux的发行版都使用了systemd 作为系统初始化程序，取代了古老的sysV风格的**init**程序。

初始化程序

> 在Linux系统上最常用的初始化程序有 `System V`、`UpStart`、`systemd`。作为系统启动过程运行的第一只程序，需要考量的不仅仅的功能的完善度，而且需要从性能的角度考虑，Systemd 和 Upstart相比较于 System V 的 init 而言，从性能上有了非常大的提升，而且功能更加完善，接下来，我们从传统的 System V 风格的 init 程序说起。

System V风格的init

> 传统的 System V 风格的初始化程序 init 在计算机启动加载内核完毕之后，最先开始运行 init 程序，PID 为1。`/etc/inittab` 文件作为 init 进程运行的配置文件，每一行定义了 init 进程运行所要进行的一个操作的集合，即当设定某次开机的运行级别后，init 程序就会读取到此文件中所对应的某个条目，并执行其中定义的一系列操作。在 RedHat 和 CentOS 较早的版本中都采用了SysVinit作为初始化程序。/etc/inittab文件内容如下：

```
[ root@node2 ~ ]# cat /etc/inittab
id:3:initdefault:
si::sysinit:/etc/rc.d/rc.sysinit
l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l6:6:wait:/etc/rc.d/rc 6
tty1:2345:respawn:/usr/sbin/mingetty tty1
```

> inittab 文件中的配置条目格式：
> **ID:runlevels:action:process**
>
> - **ID**：表示数字，id 通常需要四个字符以内。
> - **renlevels**：定义在个运行级别进行后续定义的操作。
> - **action**：定义操作的工作模式。创建的操作模式有一下几种：
>   `initdefault` 设定默认的运行级别，无需定义操作。表示当前默认的操作。
>   `sysinit` 指定系统初始化脚本，即 /etc/rc.d/rc.sysinit 脚本，此操作无条件执行。
>   `wait` 当系统切换级别时，所要进行的操作。
>   `ctrlaltdel` 定义组合键被按下时要运行的命令 Ctrl+Alt+Delete
>   `respawn` 当指定的操作被关闭时，在启动一次（例如终端）(将终端的bash进程杀死后，又会调用mingetty再调用login立即启动)
> - **process**：定义要执行的任务操作。
>
> init 进程读取 /etc/inittab 配置文件后，发现需要需要执行 sysinit 操作，于是去执行后续的 process 即 /etc/rc.d/rc.sysinit 脚本，sysinit 脚本通常会进行检查文件系统的挂载、设置内核参数、清除过期的 locks 和 pid 文件、设置主机名等一系列设置动作，同时执行 /etc/rc.d/rcN.d 目录下脚本，最后执行 rc.local 脚本。当用户的运行级别发生改变时，init 程序读取配置文件相对应的条目，传递级别数字参数，执行 /etc/rc.d/rcN.d 目录下的文件，这个目录下的文件名称格式为 K**NM**server_name 和 S**NM**server_name ，将 S

> 开头的服务启动，将K开头服务关闭，NM 为数字，表示执行优先级，数字越小，越优先被执行。
>
> 我们发现传统的 init 初始化程序初始化过程都是通过脚本的方式执行调用，而且是串行启动运行的服务，每一个服务由一个独立的脚本调用，所依赖的服务，必须严格的顺序起订，所以启动速度慢。

UpStart

> 在 CentOS6 中采用的初始化程序就是 UpStart ，但它依然兼容 SysVinit。相比较于 SusVinit 程序，UpStart 作了一些改进，UpStart 可以并行的启动没有依赖无关的服务，从而增加的启动过程的并行性，但对于互相依赖的服务，UpStart 依然无能为力，需要串行的启动。

Systemd

> 在CentOS 7上采用的系统初始化程序为 Systemd 。Systemd 采用并行启动的机制，所以有更好的性能，更快的启动速度。下面我们谈谈Systemd 特性。
>
> 1. **并行按需启动多个需要启动的服务，都能够并行的启动， 启动时间大大减小，加快启动过程。当需要用到某个服务时，Systemd才会启动它，而不是预先就将它启动。**
> 2. **高效的进程跟踪管理能力利用 Linux 内核 CGroup 的特性，能够更高效的对进程进行管理和跟踪。**
> 3. **事务一致性的机制确保相互依赖的服务能够正常运行**
> 4. **使用二进制格式保存日志信息，可以使用 journalctl 命令查看日志**
> 5. **对于天生有依赖关系的服务，Systemd 采用对 unit 文件中定义事前启动的服务**
> 6. **target 的概念模拟了运行级别**

systemd启动服务管理机制

> systemd将每一个daemon执行脚本称为一个服务单元unit，每个服务单元unit根据功能的不同分类，就分为不同的类型type。根据daemon执行脚本文件扩展名可分类为：

> 这些文件保存在`/usr/lib/systemd/system/` `/etc/systemd/system/` `/run/systemd/system/`

扩展名	说明
service	系统服务。代表一个后台进程，比如httpd，sshd等服务
socket	进程间通信
swap	管理交换分区
mount	文件系统挂载
automount	自动挂载点
path	探测文件路径或类型
timer	定时器配置单元用来定时触发用户定义的操作，这类配置单元取代了 atd、crond 等传统的定时服务
target	这类unit可以理解为是对其他unit的一个分类，他们本身不启动任务服务。例如multi-user.target 相当于在传统使用 SysVinit 系统中运行级别 5，启动 multi-user.target 相当于启动图形界面下要启动的所有服务。systemd利用这样的机制实现了模拟运行级别。

> 监视和控制systemd的主要命令是`systemctl`。该命令可用于查看系统状态和管理系统及服务。

> 运行级别target：

SysVinit中的运行级别	Systemd	作用
0	runlevel0.target,poweroff.target	关闭系统
1,s,single	runlevel1.target,rescue.target	单用户模式
3	runlevel3.target,multi-user.target	多用户模式
6	runlevel6.target,reboot.target	重启
5	runlevel5.target,graphical.target	图形界面

参考

> Systemd (简体中文) -ArchWiki