linux 系统服务

init 服务

init 进程是所有进程的发起者和控制者。因为在任何基于 Unix 的系统（比如 linux）中，它都是第一个运行的进程，所以 init 进程的编号（Process ID，PID）永远是 1。如果 init 出现了问题，系统的其余部分也就随之而垮掉了。

init 进程有两个作用。第一个作用是扮演终结父进程的角色。因为 init 进程永远不会被终止，所以系统总是可以确信它的存在，并在必要的时候以它为参照。如果某个进程在它衍生出来的全部子进程结束之前被终止，就会出现必须以 init 为参照的情况。此时那些失去了父进程的子进程就都会以 init 作为它们的父进程。快速执行一下 ps -af 命令，可以列出许多父进程 ID（Parent Process ID，PPID）为 1 的进程来。

init 的第二个角色是在进入某个特定的运行级别（Runlevel）时运行相应的程序，以此对各种运行级别进行管理。它的这个作用是由 /etc/inittab 文件定义的。

1. /etc/inittab 文件
2. /etc/inittab 文件中包括了所有 init 启动运行级别所必须的信息。这个文件中的每一行语句的格式如下所示：

   id : runlevels: action : process

注意：以#开始的语句是注释语句。看看你自己的/etc/inittab 文件就可以发现其中充斥了大量的注释语句。如果确实需要对 /etc/inittab 文件进行什么改动（一般不会出现这种情况的），记住加上一些注释语句，解释为什么要做那些修改。

Telinit 命令

通知 init 在什么时候切换系统运行级别的神秘力量实际上就是 Telinit 命令。这个命令有两个命令行参数：一个参数用来通知 init 准备切换过去的运行级别；另外一个是-tsec，其中的 sec 是在通知 init 之前需要等待的、以秒计算的时间。

注意:init 是否真的切换运行级别是由它自己决定的。很明显，它经常切换，否则这个命令就不会那么有用了。在大多数 Unix 操作系统的具体实现（包括 linux）中，Telinit 命令实际上只是一个对 init 程序的符号链接。基于此，许多人更喜欢使用 init 直接切换到他们想去的运行级别而不是使用 Telinit。就个人而言，我发现使用 Telinit 切换运行级别更便于理解和记忆。

inetd 进程

inetd 程序是一个守护进程。您可能已经知道守护进程是一些特殊的程序：它们在被启动之后，自愿放弃对调用自己终端的控制权。

守护进程与系统其余部分的接口只有依靠进程间通信（Interprocess Communication，IPC）通道、或者依靠向系统全局性日志文件（Log File）才能发送数据项。

inetd 的角色是作为 Telnet 和 FTP 等与网络服务器相关的进程的“超级服务器”。这是一个简单的道理：并不是全部的服务器进程（包括那些接受新的 Telnet 和 FTP 连接的进程）都会如此频繁地被调用，以至于必须要有一个程序随时运行在内存中。因此为了避免出现可能有几十种服务都运行在内存中准备被使用的情况，它们都列在 inetd 的配置文件/etc/inetd.conf 中。而代替它们的是 inetd 监听着进入的连接。这样只需要有一个进程在内存中就可以了。

inetd 的另外一个优点是程序员并不想把需要网络连接的进程都编写到系统中去。inetd 程序将处理网络代码，并把进入的网络数据流作为各个进程的标准输入（Standard-In，即 Stdin）传递到其中。这些进程的输出（Stdout）将会被送回连接到该进程的主机去。

注意：除非你正在进行编程，否则是不需要连接到 inetd 的 Stdin/Stdout 功能上。从另一方面来说，如果有人打算编写一个简单的命令脚本程序并让它出现在网络中，就值得深入研究这个极为强大的功能。

1. etc/inetd.conf 文件

etc/inetd.conf 文件是 inetd 的配置文件。它的结构很简单：每一

行语句代表一种服务。服务定义语句的格式如下所示：

srvce_name sock_type protocol [no]wait user srvr_prog srvr_prog_args

2. 安全性与 inetd.conf 文件

你将会发现在大多数的 linux 安装中，许多服务在缺省的情况下是打开的。如果你的系统将向因特网开放（包括通过拨号点对点协议被连通），你想做的第一件事就会是把一切都关闭！决不要假设因为你的系统没有对公众进行宣传，别人就不会找到它。从相反的方向看，寻找存在安全性攻击隐患系统的工具软件是既容易找到又容易使用的。

关闭服务的第一个步骤是把 etc/inetd.conf 文件里所有用不着的服务性说明语句都改为注释语句。一般来说，你会发现下面的方法更容易使用：先把全部东西都改为注释语句（彻底关闭网络服务），再有选择地打开需要的服务。在完成对 etc/inetd.conf 文件的修改之后，需要向守护进程报告其配置文件已经被修改了。这是通过向该守护进程发送 HUP 信号来实现的。先使用下面的命令找出 inetd.conf 对应的进程 ID：

[root@ford /root] # ps auxw | grep inetd | grep -v grep

这个命令的输出类似于下面的内容：

root 359 0.0 0.1 1232 168 ? S Jun21 0 : 00 inetd

root 359 0.0 0.1 1232 168 ? S Jun21 0 : 00 inetd

输出中的第二列告诉我们进程 ID 号（这里就是 359）。为了发送 HUP 信号，我们需要使用 Kill 命令（把这个程序叫做 Kill 多少有些误导。实际上，它只是向进程发送信号而已。缺省的情况下，它会发出请求某个程序终止运行的信号）。

下面是使用 Kill 命令发送 HUP 信号的方法：

$ kill -1 359

应该把上面命令中的 359 换成从你的系统上得到的进程编号。

syslogd 守护进程

在同一时间会发生许许多多的事情，而在终端窗口中断开连接的网络服务就更是如此了。因此，提供一个记录特殊事件和消息的标准机制就非常有必要了。linux 使用 syslogd 守护进程来提供这个服务。

syslogd 守护进程提供了一个对系统活动和消息进行记录的标准方法。许多其他种类的 Unix 操作系统也使用了兼容的守护进程。这就提供了一个在网络中跨平台记录的方法。在大型的网络环境里，这更具有价值。因为在那样的环境里，集中收集各种记录数据以获得系统运转的准确情况是很有必要的。你可以把这种记录功能子系统比作 Windows NT 的 SystemLogger。

syslogd 保存数据用的记录文件都是简明的文本文件，一般都存放在/var/log 子目录中。每个数据项构成一行，包括日期、时间、主机名、进程名、进程的 PID，以及来自该进程的消息。标准 C

函数库中的一个全局性的函数提供了生成记录消息的简单机制。如果不喜欢编写程序代码，但是又想在记录文件中生成数据项，可以选择使用 Logger 命令。可以想象，像 syslogd 这样重要的工具应该是作为开机引导命令脚本程序的一部分来启动的。你准备在服务器环境中使用的任何一个 linux 发行版本都已经为你设置好了。

1. 调用 syslogd

如果需要手动启动 syslogd，或者需要修改开机引导时启动它的命令脚本程序，你就必须注意 syslogd 的命令行参数，请大家参看有关书籍，这里不做详细介绍。

2./etc/syslog.conf 文件

/etc/syslog.conf 文件包含了 syslogd 需要运行的配置信息。这个文件的格式有些不寻常，但是现有的缺省配置文件将足以满足使用需要了，除非你需要在特定的文件中查找特定的信息，或者需要把这些信息发送到远程记录计算机去。

- 记录信息分类

在我们掌握/etc/syslog.conf 文件格式本身之前，需要先了解记录消息是如何分类的。每个消息都有一个功能值（Facility）和一个优先权值（Priority）。功能值告诉我们这条消息是由哪个子系统产生的，而优先权值则告诉我们这个消息有多重要。这两个值由句号分隔而且都有等价的字符串，从而容易记忆。

- /etc/syslog.conf 文件的格式

下面是配置文件里各语句的格式：

facility/priority combinations separated by commas file

/process/host to log to

举例如下：

kern.info /ver/log/kerned

syslogd 还可以灵活地把记录消息发送到多种不同的保存目的地去。它可以把消息保存为文件、把消息发送到 FIFO 队列、发送到一组用户、或者（在大型站点集中记录消息的情况下）发送到一个中心记录主机中。为了区分这些目的地，在目的地入口使用了下面的规则：

- 如果保存目的地的开始字符是斜杠字符（/），消息将发送到某个文件。
- 如果保存目的地的开始字符是垂直字符（|），消息将发送到某个 FIFO 队列。
- 如果保存目的地的开始字符是“@”字符，消息将发送到某个主机。

cron 进程

cron 进程为用户提供一种可以计划在一定时间间隔后自动执行任务的功能。cron 通常还会负责 at 队列中作业的启动。该进程从 crontab 文件中得到信息，对于每个用户都会有一个独立的 crontab 文件。运行 crontab -l 命令，就可以查看已排列的命令列

表。

如果要更改 cron 信息，必须创建一个 crontab 文件。crontab 有固定的格式，有效行包括六个区域，即：

分钟（0-59），小时（0-23），日（1-31），月（1-12），星期（0-7，0 和 7 都代表周日），命令（要运行的任务）

每个区域都可能带一个星号，表示整个区间，每个区域也可以使用 x-y 的形式包含一个范围，也可能包含一个用逗号隔开的列表。