Guia Linux

INIT

Descrição

- É o primeiro programa executado pelo **kernel** durante a inicialização.
- Por padrão, o processo *init* é identificado no sistema com o número 1, ou seja, o **PID** do *init* é 1 (a identificação dos processos em execução pode ser visto no diretório */proc*).

Versões

Existem três versões do *init* para inicialização dos serviços no Linux.

1) SysVinit – Foi um dos mais utilizados (e por mais tempo) em diversas distribuições Linux. Neste caso, o *init* lê o arquivo de configuração */etc/inittab* e executa o **shell** script */etc/rc.sysinit*. Basicamente são definidos 8 níveis de execução *(runlevel)* para a inicialização.

Runlevel	Significado
0	encerra o sistema
1	inicializa o sistema em modo monousuário
2 a 5	inicializa o sistema em modo multiusuário
6	reinicializa o sistema
S	Joga o sistema no modo monousuário sem antes parar os processos em execução

Os scripts executados quando o sistema entra no *runlevel* X estão no diretório */etc/rcX.d*, onde X pode ter valor de 0 a 6. Na realidade, estes diretórios apenas contém links simbólicos para os scrips localizados no diretório */etc/init.d*. O diretório */etc/init/* possui os arquivos de configuração

dos **processos** inicializados pelo *init*.

Para ver o *runlevel* usado, basta digitar

```
runlevel
```

É possível parar o sistema usando o comando

```
sudo init 0
```

ou reinicializar o sistema com o comando

```
sudo init 6
```

Para alterar o *runlevel* para o nível 3, basta digitar

```
sudo init 3
```

2) Upstart – É baseado em eventos, onde os serviços do sistema podem ser associados a estes eventos. O *Upstart* define o que fazer quando um evento começa, muda ou termina. É importante observar que há compatibilidade do *Upstart* com o *SysVinit*.

3) Systemd – Desde 2015, é o sistema padrão de gerenciamento de serviços das distribuições Linux e tem como uma das principais características o uso de paralelização agressiva para inicializar, gerenciar e parar serviços.

O *Systemd* utiliza diferentes tipos de unidades para inicializar e supervisionar o sistema.

- *service* (serviço) – esta unidade corresponde a um *daemon* que pode ser iniciado, parado, reiniciado e recarregado.
- *socket* – esta unidade encapsula um socket no sistema de arquivos ou na Internet. Cada unidade *socket* tem uma unidade *service* equivalente.
- *device* – esta unidade encapsula um dispositivo.
- *mount* (montagem) – esta unidade encapsula um ponto de montagem na hierarquia do sistema de arquivos.
- *automount* (automontagem) – esta unidade encapsula um ponto de montagem automático.
- *target* (alvo) – esta unidade é usada para agrupamento lógico de unidades. Ao invés de fazer algo, ela simplesmente referencia outras unidades, que podem ser controladas de forma conjunta. As *targets* correspondem ao *runlevels* (níveis de execução) do *SysVinit*. Para ver a lista das *targets* do sistema, digite

```
systemctl list-units --type=target
```

Para verificar o nível de execução usado no *Systemd*, basta digitar

```
systemctl get-default
```

Abaixo, é mostrada a resposta quando o nível de execução é o modo gráfico (corresponde ao *runlevel* 5 do *SysVinit)*.

```
graphical.target
```

Para alterar o nível de execução para modo texto multiusuário (corresponde ao *runlevel* 3 do *SysVinit)*, entre com

```
systemctl set-default multi-user.target
```

- *snapshot* – similar a unidade *target*, a unidade *snapshot* não faz nada por si só a não ser referenciar outras unidades.

Abaixo são listados alguns comandos do *Systemd*.

- Para ver o status do sistema, digite

```
systemctl status
```

- Para listar as unidades em execução, basta digitar

```
systemctl
```

ou

```
systemctl list-units
```

- Para listar as falhas na inicialização do sistema, entre com

```
systemctl --state=failed
```

- Para reinicializar o sistema, digite

```
systemctl reboot
```

- Para parar o sistema, basta entrar com

```
systemctl shutdown
```

Exemplos: SysVinit X Systemd

Abaixo são mostrados alguns exemplos das duas principais versões do *init*.

- Para iniciar um serviço
  - SysVinit : *serviço start*
  - Systemd : *systemclt start serviço*
- Para parar um serviço
  - SysVinit : *serviço stop*
  - Systemd : *systemctl stop serviço*
- Para reiniciar um serviço
  - SysVinit : *serviço restart*
  - Systemd : *systemctl restart serviço*
- Para recarregar um serviço
  - SysVinit : *serviço reload*
  - Systemd : *systemctl reload serviço*
- Para verificar o status de um serviço
  - SysVinit : *serviço status*
  - Systemd : *systemctl status serviço*
- Para habilitar um serviço no boot
  - SysVinit : *ckconfig serviço on*
  - Systemd : *systemctl enable serviço*
- Para desabilitar um serviço no boot
  - SysVinit : *ckconfig serviço off*
  - Systemd : *systemctl disable serviço*
- Para verificar se um serviço está habilitado
  - SysVinit : *ckconfig –list serviço*
  - Systemd : *systemctl is-enabled serviço*

É importante observar que, para o *Systemd*, os comandos acima podem ser usados para qualquer tipo de unidade. Por exemplo, o comando

```
systemctl status sys-module-configfs.device
```

informa o status de uma unidade *device*, enquanto

```
systemctl is-enabled snapd.socket
```

verifica se uma unidade de *socket* está habilitada.

Observações

- Mesmo quando o *Systemd* está sendo usado, é possível usar a maioria dos comandos do *SysVinit*, pois há mapeamento entre eles. Abaixo são mostrados alguns exemplos.

SysVinit	Systemd
/sbin/runlevel	/bin/systemctl
/sbin/init	/lib/systemd/systemd

Os nomes *runlevel* e *init* são mantidos apenas por questões históricas. Por exemplo, quando o usuário digita *runlevel*, o sistema executa *systemctl*.

Sumário | Topo