REFERÊNCIA RÁPIDA DO SYSTEMD

Gerenciamento de serviços, units, timers e journalctl

Gerenciamento de Serviços

Comandos Básicos de Serviço

```
systemctl start nginx
systemctl stop nginx
systemctl restart nginx
systemctl reload nginx    # recarregar configuração
systemctl status nginx
```

Habilitar / Desabilitar

```
systemctl enable nginx    # iniciar no boot
systemctl disable nginx   # remover do boot
systemctl enable --now nginx  # habilitar + iniciar
systemctl is-enabled nginx
```

Estados do Serviço

`active (running)`	Serviço está em execução normalmente
`active (exited)`	Executou e encerrou com sucesso
`inactive (dead)`	Serviço está parado
`failed`	Serviço travou ou encerrou com erro
`activating`	Serviço está iniciando

Arquivos de Unit

Localização do Arquivo de Unit

`/etc/systemd/system/`	Units criados pelo administrador (maior prioridade)
`/run/systemd/system/`	Units gerados em tempo de execução
`/usr/lib/systemd/system/`	Units instalados por pacotes
`~/.config/systemd/user/`	Units de nível de usuário

Unit de Serviço Básico

```
[Unit]
Description=Minha Aplicação
After=network.target
[Service]
ExecStart=/usr/bin/myapp --config /etc/myapp.conf
Restart=on-failure
User=appuser
[Install]
WantedBy=multi-user.target
```

Aplicar Alterações

```
systemctl daemon-reload   # recarregar arquivos de unit
systemctl restart myapp   # aplicar alterações
```

Timers

Unit de Timer

```
[Unit]
Description=Executar backup diário
[Timer]
OnCalendar=*-*-* 02:00:00
Persistent=true
[Install]
WantedBy=timers.target
```

Sintaxe do OnCalendar

`*-*-* 02:00:00`	Diariamente às 2:00
`Mon *-*-* 09:00:00`	Toda segunda-feira às 9h
`*-*-01 00:00:00`	Primeiro dia de cada mês
`hourly / daily / weekly`	Agendamentos abreviados

Gerenciamento de Timers

```
systemctl list-timers --all
systemctl start backup.timer
systemctl enable backup.timer
systemd-analyze calendar "daily"
```

Targets

Targets Comuns

`multi-user.target`	Boot normal, multi-usuário, sem GUI
`graphical.target`	Desktop com GUI completa
`rescue.target`	Modo de recuperação de usuário único
`emergency.target`	Shell mínimo, somente root
`network-online.target`	Rede totalmente configurada
`timers.target`	Todos os units de timer prontos

Comandos de Target

```
systemctl get-default
systemctl set-default multi-user.target
systemctl isolate rescue.target
systemctl list-dependencies graphical.target
```

Journalctl

Visualizando Logs

```
journalctl -u nginx       # logs para o unit
journalctl -u nginx -f    # acompanhar (tail)
journalctl -u nginx --no-pager
journalctl -b             # apenas o boot atual
```

Filtrando Logs

```
journalctl --since "2026-03-01"
journalctl --since "1 hour ago"
journalctl -p err         # erros e acima
journalctl _PID=1234
```

Níveis de Prioridade

`emerg (0)`	Sistema inutilizável
`alert (1)`	Ação imediata necessária
`crit (2)`	Condição crítica
`err (3)`	Condição de erro
`warning (4)`	Condição de aviso
`info (6)`	Informativo
`debug (7)`	Mensagens de depuração

Manutenção de Logs

```
journalctl --disk-usage
journalctl --vacuum-size=500M
journalctl --vacuum-time=30d
```

Rede

networkctl

```
networkctl list
networkctl status eth0
networkctl up eth0
networkctl down eth0
```

systemd-resolve

```
resolvectl status
resolvectl query example.com
resolvectl flush-caches
resolvectl statistics
```

Aguardar Rede

```
# Na seção [Unit] do arquivo de unit:
After=network-online.target
Wants=network-online.target
```

Montagens

Unit de Montagem

```
[Unit]
Description=Montar volume de dados
[Mount]
What=/dev/sdb1
Where=/mnt/data
Type=ext4
Options=defaults,noatime
[Install]
WantedBy=multi-user.target
```

Unit de Montagem Automática

```
[Unit]
Description=Montar dados automaticamente no acesso
[Automount]
Where=/mnt/data
TimeoutIdleSec=300
[Install]
WantedBy=multi-user.target
```

Convenção de Nomenclatura

`/mnt/data`	Arquivo de unit: `mnt-data.mount`
`/var/lib/app`	Arquivo de unit: `var-lib-app.mount`

Caminho de montagem com `/` substituído por `-` , traço inicial removido

Ambiente

Definindo Variáveis de Ambiente

```
[Service]
Environment=APP_ENV=production
Environment=PORT=8080
EnvironmentFile=/etc/myapp/env
```

Formato do Arquivo de Ambiente

```
# /etc/myapp/env
APP_ENV=production
DATABASE_URL=postgres://localhost/db
SECRET_KEY=changeme
```

Endurecimento do Serviço

`ProtectSystem=strict`	Sistema de arquivos somente leitura exceto caminhos permitidos
`ProtectHome=true`	Ocultar /home, /root, /run/user
`NoNewPrivileges=true`	Impedir escalada de privilégios
`PrivateTmp=true`	/tmp isolado para o serviço
`ReadWritePaths=/var/lib/myapp`	Permitir gravações em caminhos específicos

Dependências

Diretivas de Ordenação e Requisito

`After=b.service`	Iniciar após b (apenas ordenação)
`Before=b.service`	Iniciar antes de b (apenas ordenação)
`Requires=b.service`	Dependência forte; falha se b falhar
`Wants=b.service`	Dependência fraca; não falha se b falhar
`BindsTo=b.service`	Para quando b para
`Conflicts=b.service`	Não pode ser executado ao mesmo tempo que b

Inspecionando Dependências

```
systemctl list-dependencies nginx
systemctl list-dependencies --reverse nginx
systemd-analyze dot nginx.service | dot -Tsvg > deps.svg
```

Padrões Comuns

Políticas de Reinício

`Restart=no`	Nunca reiniciar (padrão)
`Restart=on-failure`	Reiniciar em saída com código diferente de zero
`Restart=always`	Sempre reiniciar (para daemons)
`RestartSec=5`	Aguardar 5 segundos antes de reiniciar
`StartLimitBurst=3`	Máximo de reinícios no intervalo
`StartLimitIntervalSec=60`	Intervalo para contagem de burst

Substituir Sem Editar

```
systemctl edit nginx  # cria drop-in
# /etc/systemd/system/nginx.service.d/override.conf
systemctl cat nginx   # exibir configuração efetiva
systemctl revert nginx  # remover substituições
```

Análise do Sistema

```
systemd-analyze                 # tempo de boot
systemd-analyze blame           # tempo por unit
systemd-analyze critical-chain
systemctl list-units --failed
```