systemd 快速参考

服务管理、unit 文件、定时器与 journalctl

服务管理

基本服务命令

```
systemctl start nginx
systemctl stop nginx
systemctl restart nginx
systemctl reload nginx    # reload config
systemctl status nginx
```

启用 / 禁用

```
systemctl enable nginx    # start at boot
systemctl disable nginx   # remove from boot
systemctl enable --now nginx  # enable + start
systemctl is-enabled nginx
```

服务状态

active (running)	服务正常运行中
active (exited)	已运行一次并成功退出
inactive (dead)	服务已停止
failed	服务崩溃或以非零状态退出
activating	服务正在启动

Unit 文件

Unit 文件路径

/etc/systemd/system/	管理员创建的 unit（优先级最高）
/run/systemd/system/	运行时生成的 unit
/usr/lib/systemd/system/	软件包安装的 unit
~/.config/systemd/user/	用户级 unit

基本服务 Unit

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

应用变更

```
systemctl daemon-reload   # reload unit files
systemctl restart myapp   # apply changes
```

定时器

Timer Unit

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

OnCalendar 语法

***-*-* 02:00:00**	每天凌晨 2:00
Mon *-*-* 09:00:00	每周一 9:00
***-*-01 00:00:00**	每月第一天
hourly / daily / weekly	简写调度表达式

定时器管理

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

Target

常用 Target

multi-user.target	正常启动，多用户，无 GUI
graphical.target	完整 GUI 桌面
rescue.target	单用户救援模式
emergency.target	最小化 shell，仅 root
network-online.target	网络已完整配置
timers.target	所有定时器 unit 就绪

Target 命令

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

Journalctl

查看日志

```
journalctl -u nginx        # logs for unit
journalctl -u nginx -f     # follow (tail)
journalctl -u nginx --no-pager
journalctl -b              # current boot only
```

过滤日志

```
journalctl --since "2026-03-01"
journalctl --since "1 hour ago"
journalctl -p err          # errors and above
journalctl _PID=1234
```

优先级级别

emerg (0)	系统不可用
alert (1)	需要立即处理
crit (2)	严重状况
err (3)	错误状况
warning (4)	警告状况
info (6)	一般信息
debug (7)	调试级别消息

日志维护

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

网络

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
```

等待网络就绪

```
# In unit file [Unit] section:
After=network-online.target
Wants=network-online.target
```

挂载

Mount Unit

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

Automount Unit

```
[Unit]
Description=Automount data on access
[Automount]
Where=/mnt/data
TimeoutIdleSec=300
[Install]
WantedBy=multi-user.target
```

命名规则

/mnt/data	Unit 文件：**mnt-data.mount**
/var/lib/app	Unit 文件：**var-lib-app.mount**

挂载路径中 `/` 替换为 `-`，去掉开头的短横线

环境变量

设置环境变量

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

环境变量文件格式

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

服务加固

ProtectSystem=strict	文件系统只读，允许路径除外
ProtectHome=true	隐藏 /home、/root、/run/user
NoNewPrivileges=true	禁止提权
PrivateTmp=true	为服务提供隔离的 /tmp
ReadWritePaths=/var/lib/myapp	允许写入指定路径

依赖关系

排序与依赖指令

After=b.service	在 b 之后启动（仅排序）
Before=b.service	在 b 之前启动（仅排序）
Requires=b.service	强依赖；b 失败则自身失败
Wants=b.service	弱依赖；b 失败不影响自身
BindsTo=b.service	b 停止时自身也停止
Conflicts=b.service	不能与 b 同时运行