



Для инициализации служб довольно долгое время применялся механизм, называющийся init. Принцип его работы был довольно прост: в системе присутствовала директория, содержавшая так называемые управляющие скрипты. Это обычные шелл-сценарии, которые init последовательно выполнял, обеспечивая старт службам. Т.е. время приведения всех служб в рабочее состояние представляло собой суммарное время запуска этих управляющих скриптов.

Начиная с версии 7, в RHEL/Centos для инициализации служб применяется системный менеджер, носящий название systemd. В отличие от init, он способен осуществлять параллельный запуск служб, что ускоряет время загрузки системы. (Тем не менее, существуют службы, которые необходимо запускать в определённой последовательности, но в systemd это учтено).

Помимо простого запуска служб, systemd может контролировать их дальнейшее состояние, может запускать службы по требованию, по расписанию и обладает рядом других возможностей, которых был лишен init.

Systemd не использует управляющие скрипты для запуска и остановки служб. Вместо этого он оперирует таким понятием, как юнит. Юнит можно рассматривать как объект с определённым набором свойств, посредством которого systemd управляет запуском службы. Сам по себе юнит описан в виде конфигурационного файла, структуру которого мы рассмотрим чуть ниже.

В зависимости от специализации, юниты разделены на несколько типов.

[demo@localhost ~]\$ systemctl -t help Available unit types: service socket target snapshot device mount automount swap timer path . . .

Все эти юниты так или иначе отвечают за запуск различных процессов в системе, но делают это по-разному. Например, с помощью юнитов типа .service можно запуском служб в данный момент времени, а юнит .timer позволяет делать это в соответствии с неким расписанием, без применения планировщика. Юнит типа .path позволяет отслеживать изменения файлов или директорий и запускать в ответ какие-либо процессы в системе. Нас в первую очередь будет интересовать юнит типа .service, т.к. именно с помощью него администратор обычно управляет работой различных служб.

Состояния юнитов systemd

active running	Означает, что юнит был успешно проинициализирован, служба запущена и работает в данный момент
active exited	Юнит был успешно проинициализирован, но сам по себе он не предполагает, что в памяти останется какая-то работающая программа. Т.е. процесс запустился, выполнил свою работу и завершился.
active waiting	Юнит начал инициализироваться, но до конца эту процедуру к настоящему моменту не прошёл. Возможно, есть какие-то условия, которые препятствуют этому.
failed failed	С инициализацией службы, за которую отвечает юнит, возникли какие-то проблемы. Запустить её не удалось

Для начала получим список всех служб, которые systemd попытался проинициализировать при загрузке системы (для этого используется вызов "listunits"):

[demo@localhost ~]\$ systemctl list-units --type=service --no-pager **UNIT** LOAD ACTIVE SUB DESCRIPTION loaded active exited abrt-ccpp.service Install ABRT coredump hook abrt-oops.service loaded active running ABRT kernel log watcher loaded active running ABRT Xorg log watcher abrt-xorg.service loaded active running ABRT Automated Bug Reporting Tool abrtd.service loaded active running Job spooling tools atd.service loaded active running Security Auditing Service auditd.service loaded active running cups mDNS/DNS-SD Stack cups-daemon.service blk-availability.service loaded active exited Availability of block devices loaded active running NTP client/server chronyd.service crond.service loaded active running Command Scheduler loaded active running CUPS Printing Service cups.service loaded active running D-Bus System Message Bus dbus.service firewalld.service loaded active running firewalld - dynamic firewall daemon loaded active running GNOME Display Manager gdm.service gssproxy.service loaded active running GSSAPI Proxy Daemon httpd.service loaded failed failed The Apache HTTP Server irgbalance.service loaded active running irgbalance daemon ...

Основное общение с systemd осуществляется с помощью команды systemctl. По умолчанию, в вывод попадают юниты всех типов, отобрать какой-то конкретный тип можно с помощью опции "--type=". Также команда systemctl выводит результаты своей работы постранично. Отключить это свойство можно либо добавив опцию "--no-pager", либо объявив пустую переменную с именем SYSTEMD_PAGER: "export SYSTEMD_PAGER="

Управление юнитами systemd

Параметр	Описание
stop	Остановить работающую службу.
start	Запустить службу.
restart	Перезапустить работающую службу. При этом процесс завершается и создаётся новый, с другим значением PID.
reload	Переинициализировать работающую службу. Данный вызов заставляет программу перечитать свои конфигурационные файлы. Используется при внесении изменений в конфигурацию. Процесс при этом не завершается.
is-enabled	Проверить, включена ли автоматическая активация службы.
is-active	Проверить, работает ли служба в данный момент.
enable	Включить автоматическую активацию службы
disable	Отключить автоматическую активацию службы
mask	"Маскирование" службы. Этот вызов подменяет конфигурационный файл юнита ссылкой на /dev/null, что исключает его случайную активацию. Даже root теперь не сможет его запустить.
unmask	Обратное действие для вызова "mask"
status	Получить подробную информацию о текущем состоянии юнита.

Рассмотрим вызов "status" подробнее:

[demo@localhost ~]\$ sudo systemctl status sshd

• sshd.service - OpenSSH server daemon

Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset:

enabled)

Active: active (running) since Wed 2018-06-06 02:29:00 EDT; 1min 33s ago

Docs: man:sshd(8) man:sshd config(5)

Main PID: 1130 (sshd)

CGroup: /system.slice/sshd.service ____1130 /usr/sbin/sshd -D

Jun 06 02:29:00 localhost.localdomain systemd[1]: Starting OpenSSH server daemon...

Jun 06 02:29:00 localhost.localdomain sshd[1130]: Server listening on 0.0.0.0 port 22.

Jun 06 02:29:00 localhost.localdomain sshd[1130]: Server listening on :: port 22.

Jun 06 02:29:00 localhost.localdomain systemd[1]: Started OpenSSH server daemon.

Команда позволяет получить исчерпывающую информацию о текущем состоянии службы. Стоит заметить, что имя юнита мы указали без ".service". Так можно делать, если подразумевается именно данный тип юнита (он используется поумолчанию). Во всех других случаях необходимо указывать тип.

- sshd.service OpenSSH server daemon даёт нам представление о том, что это за служба.
- Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset: enabled) Говорит о том, что конфигурационный файл юнита (не самой службы) был успешно считан, "enabled" указывает на то, что служба добавлена в список запускаемых автоматически
- Active: active (running) since Wed 2018-06-06 02:29:00 EDT; 1min 33s ago Сообщает о том, что служба в данный момент запущена и предоставляет информацию о том, как давно это произошло

Ниже представлена информация о местоположении документации к данной службе, идентификаторе её процесса и несколько строк из журнала, в который служба отправляет различные уведомления о своём состоянии.

- Обратите внимание на расположение конфигурационного файла юнита. Он находится в директории /usr/lib/systemd/system. Конфигурационные файлы юнитов располагаются в одной из трёх директорий:
 - /usr/lib/systemd/system Файлы, созданные при установке программного обеспечения из RPM-пакетов;
 - /run/systemd/system Файлы, созданные в процессе работы системы. Имеют более высокий приоритет, чем файлы из /usr/lib/systemd/system;
 - /etc/systemd/system Файлы, созданные и управляемые системным администратором. Имеют более высокий приоритет, чем файлы из /run/systemd/system. Т.е. в случае, если systemd обнаруживает два файла с одинаковым именем в директории /usr/lib/systemd/system и

/etc/systemd/system, он будет использовать тот, который лежит в /etc.

Если требуется внести изменения в конфигурационный файл юнита, его необходимо скопировать из /usr/lib/systemd/system в etc/systemd/system. Таким образом, вы

- а) легко сможете понять, какие файлы менялись в вашей системе
- b) сможете отменить изменения, просто удалив (или переместив) файл из директории /etc/systemd/system

Иногда бывает полезно получить более полную картину, включающую в себя и те юниты, которые неактивны. Это можно сделать с помощью вызова list-unit-files:

```
[demo@localhost ~]$ systemctl list-unit-files --type=service --no-pager
UNIT FILE
                                   STATE abrt-ccpp.service
  enabled abrt-oops.service
                                              enabled abrt-pstoreoops.service
    disabled abrt-vmcore.service
                                              enabled abrtd.service
       enabled accounts-daemon.service
                                                 enabled alsa-restore.service
        static alsa-state.service
                                                  static alsa-store.service
        static anaconda-direct.service
                                                static anaconda.service
       static arp-ethers.service
                                               disabled atd.service
       enabled ...
```

- enabled Служба, управляемая юнитом, запускается
- disabled Служба отключена
- static Юнит запускается в качестве зависимого от другого юнита. Предполагается, что к этой категории юнитов не должны применяться вызовы enable и disable.

Таким образом, получается, что юниты могут зависеть друг от друга. Для получения структуры этих зависимостей в команде systemctl предусмотрен специальный вызов:

[demo@localhost ~]\$ systemctl list-dependencies crond.service --no-pager crond.service ● |—system.slice

- L—basic.target
- ⊢alsa-restore.service
- ⊢alsa-state.service
- —microcode.service
- —rhel-autorelabel-mark.service
- ⊢rhel-autorelabel.service
- Implication in the property of th
- ⊢rhel-dmesg.service
- ⊢rhel-loadmodules.service
- —selinux-policy-migrate-local-changes@targeted.service
- ⊢paths.target
- ⊢slices.target
- | |---.slice

. . .

Задание 1

По умолчанию в установленной системе бывает запущен демон сups, который представляет собой открытую реализацию протокола печати IPP. Если вы не пользуетесь этим механизмом, демон cups можно спокойно отключить.

- 1) Запущен ли демон cups? systemctl status cups-daemon.service
- 2) Проверьте, запускается ли он вместе с системой? systemctl status cups-daemon.service
- 3) Определите, прослушивает ли демон сups какой-либо порт?

ss -ulpn | grep cups

- 4) Остановите процесс демона cups systemctl stop cups-daemon.service
- 5) Перезагрузите систему systemctl reboot
- 6) Снова проверьте, запущен ли демон cups systemctl status avahi-daemon.service

- 1) Почему так произошло?
- 2) Остановите процесс демона cups systemctl stop cups-daemon.service
- 3) Отключите его автоматический старт вместе с системой

systemctl disable cups-daemon.service

- 4) Снова перезагрузите систему. Запущен ли демон сирѕ теперь?
- 5) Проверьте, пропал ли демон cups из списка процессов, прослушивающих порты

ss -tlpn | grep cups