SystemD

система инициализации, взамен init upstart

Единицы (units) в systemd

service	запускает, останавливает или перезагружает демоны, также можно запускать SysV-сценарии.
socket	конфигурационный файл сокета, который связанный с определенным сервисом (service)
device	конфигурационный файл содержащий правило udev для обработки дерева устройств.
mount	монтирования файловой системы. Также можно получить информацию о файловой системы из файла /etc/fstab.
automount	автоматическое монтирование файловой системы.
target	логическая группировка единиц, ссылается на другие единицы. Например, bluetooth.target — запускает службы, при активации Bluetooth-устройства.
snapshot	создание ссылок на другие единицы, восстанавливает список ранее запущенных служб.
timer	подобие cron, активация единиц по таймеру.
swap	управление файлами подкачки.
path	активация других служб на основе inotify

Основные команды systemd

systemctl	Список запущенных юнитов
systemctl --failed	Юниты, запуск которых завершился неудачей
systemctl list-unit-files	Список доступных юнитов
systemctl start <unit-name>	Запуск юнита
systemctl stop <unit-name>	Остановка юнита
systemctl restart <unit-name>	Перезагрузка юинта
systemctl reload <unit-name>	Перезагрузка настроек юнита
systemctl status <unit-name>	Просмотреть статус юнита
systemctl is-enabled <unit-name>	Проверить разрешен ли запуск юнита при старте системы
systemctl enable <unit-name>	Разрешить запуск юнита при старте системы
systemctl disable <unit-name>	Запретить запуск юнита при старте системы
systemctl daemon-reload	Перезагрузка systemd с поиском измененных или новых юнитов

Расположение конфигурационных файлов

/usr/lib/systemd/system/	юниты из установленных пакетов RPM, DEB — всякие nginx, apache, mysql и прочие
/run/systemd/system/	юниты, созданные в рантайме
/etc/systemd/system/	юниты, созданные системным администратором
/etc/systemd/system/multi-user.target.wants	симлинки на юниты. созданные после выполнения systemctl enable <unit-name>

Формат конфигурационных файлов

- Секции [Unit], [Service], [Install]
- Запускать юнит после какого-либо сервиса или группы сервисов (например network.target): After=syslog.target
- Для запуска сервиса необходим запущенный сервис mysql: Requires=mysql.service
- Для запуска сервиса желателен запущенный сервис redis: Wants=redis.service
- Type=simple - (по умолчанию): systemd предполагает, что служба будет запущена незамедлительно. Процесс при этом не должен разветвляться. Не используйте этот тип, если другие службы зависят от очередности при запуске данной службы.
- Type=forking - systemd предполагает, что служба запускается однократно и процесс разветвляется с завершением родительского процесса. Данный тип используется для запуска классических демонов.
- Запрет на убийство сервиса вследствие нехватки памяти и срабатывания механизма OOM: -1000 полный запрет (такой у sshd): OOMScoreAdjust=-100
- WantedBy=multi-user.target - уровень запуска, multi-user.target или runlevel3.target соответствует нашему привычному runlevel=3
- Таймаут в секундах, сколько ждать system отработки старт/стоп команд - TimeoutSec=300
- Restart=always/on-failure - systemd автоматически рестартует сервис, если он вдруг перестанет работать. Контроль ведется по наличию процесса из PID файла

Стандартный конфиг

- [Unit]
- Description=The nginx HTTP and reverse proxy server
- After=syslog.target network.target remote-fs.target nss-lookup.target

- [Service]
- Type=forking
- PIDFile=/run/nginx.pid
- ExecStartPre=/usr/sbin/nginx -t
- ExecStart=/usr/sbin/nginx
- ExecReload=/bin/kill -s HUP $MAINPID
- # Sleep for 1 second to give PassengerAgent a chance to clean up.
- # Use TERM instead of QUIT to prevent Nginx from leaving stale Unix socket and failing the next start (https://trac.nginx.org/nginx/ticket/753)
- ExecStop=/bin/kill -s TERM $MAINPID ; /bin/sleep 1
- PrivateTmp=true
- Restart=always

- [Install]
- WantedBy=multi-user.target

Кастомный конфиг

- [Unit]
- Description=sidekiq
- After=syslog.target network.target
- Requires=redis.service

- [Service]
- Type=forking
- User=app
- Group=app
- WorkingDirectory=/home/app/webapp/current
- Environment=RAILS_ENV=staging
- PIDFile=/home/app/webapp/shared/pids/sidekiq.pid
- ExecStart=/home/app/.rvm/gems/ruby-2.3.1@ebags/bin/bundle exec sidekiq -d -e staging -C config/sidekiq.yml -i 0 -P ../shared/pids/sidekiq.pid -L log/sidekiq.log
- ExecStop=/home/app/.rvm/gems/ruby-2.3.1@ebags/bin/bundle exec sidekiqctl stop ../shared/pids/sidekiq.pid
- Restart=on-failure

- [Install]
- WantedBy=multi-user.target

результат команды systemctl status nginx

- ● nginx.service - The nginx HTTP and reverse proxy server
- Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; vendor preset: disabled)
- Active: active (running) since cp 2016-08-03 18:57:58 UTC; 2 weeks 0 days ago
- Process: 44288 ExecStop=/bin/sleep 1 (code=exited, status=0/SUCCESS)
- Process: 44287 ExecStop=/bin/kill -s QUIT $MAINPID (code=exited, status=0/SUCCESS)
- Process: 13211 ExecReload=/bin/kill -s HUP $MAINPID (code=exited, status=0/SUCCESS)
- Process: 44306 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
- Process: 44304 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
- Main PID: 44331 (nginx)
- CGroup: /system.slice/nginx.service
- ├─10037 Passenger RubyApp: /home/app/webapp/current/public (production)
- ├─28548 Passenger RubyApp: /home/app/webapp/current/public (production)
- ├─44308 Passenger watchdog
- ├─44311 Passenger core
- ├─44322 Passenger ust-router
- ├─44331 nginx: master process /usr/sbin/nginx
- ├─44332 nginx: worker process
- └─44333 nginx: worker process

Преимущества

- Простые и понятные конфигурационные файлы
- Продуманая архитектура, взаимосвязь со всеми компонентами ОС (в частности, есть поддержка GNOME и KDE)
- Хорошая документированность
- Доступность необходимого функционала из коробки

Недостатки

- Необходимо знать архитектуру, чтобы использовать все возможности systemd
- Отсутствие механизмов нотификации при авариях
- Отсутствие механизмов мониторинга нагрузки/доступности сервисов