Система инициализации (Upstart, Systemd)

Преимущества upstart:

- Ориентация на события (а не на уровни выполнения операционной системы). Событиями могли быть не только загрузка системы, старт/остановка процессов, но и, например, подключение USB-устройства.
- Асинхронный старт сервисов (в отличие от последовательного, одного за другим, запуска скриптов в sysinitV, в апстарт скрипты могли запускаться независимо друг от друга), функция супервизора для процессов (самая простая и понятная выгода — автоматический рестарт завершившегося из-за ошибки процесса), упрощение скриптов (несмотря на то что существовали «помощники», такие как start-stop-daemon, SysV-init-скрипты были зачастую сложными баш-скриптами, в то время как upstart-скрипты предоставляли гораздо больше таких «помощников», что значительно упрощало их разработку, так как позволяло переиспользовать готовое решение).

В 2010 году была представлена systemd, новая подсистема для инициализации и управления процессами.

Плюсы systemd:

- В ней возможен параллельный запуск сервисов. Она автоматически решает зависимости между сервисами (в какой последовательности должны запускаться сервисы для того, чтобы все условия были удовлетворены).
- Она может перезапускать процессы автоматически.
- Она поддерживает механизм изоляции ресурсов. Это позволяет эффективнее управлять окружением, когда на сервере запущены различные процессы.
- Она поддерживает логирование событий. Например, мы можем получить информацию о том, что происходило при попытке запустить сервис, и увидеть ошибки, которые не дали сервису запуститься корректно.

В отличие от SysV-init и upstart, systemd оперирует не скриптами, а специального вида конфигурационными файлами — юнитами.

Основные виды юнитов:

- .target — позволяет группировать модули с уровнем выполнения;
- .service — отвечает за запуск сервисов, например nginx;
- .mount — отвечает за подключение файловых систем;
- .automount — позволяет подключать файловые системы при обращении;
- .timer — позволяет запускать модули по расписанию, то есть реализует ту же функциональность, что и cron;
- .socket — предоставляет возможность запуска сервисов в ответ на входящее соединение;
- .slice — позволяет настроить изоляцию ресурсов;
- .device — позволяет реагировать на подключение устройств;

- .path — управляет иерархией файловой системы, например позволяет следить за директорией и, если в ней создаются новые файлы, запускать какой-либо сервис.

Юниты располагаются в следующих директориях:
- /etc/systemd/system — локальная конфигурация;
- /run/systemd/system — конфигурация рантайма (не используется);
- /usr/lib/systemd/system (/lib/systemd/system) — юниты установленных пакетов.

Эти директории перечислены в порядке приоритета, то есть, если и в /etc/systemd/system, и в /lib/systemd/system будет найден юнит с одинаковым именем, выполнится тот, что будет располагаться в директории /etc/systemd/system.

Чтобы вывести список всех юнитов, мы можем использовать команду

```
systemctl list-units
```

Кроме этого, можем вызвать список юнитов определённого типа:

```
systemctl list-units --type=service
```

Юнит nginx

В нём три секции. Секция [Unit] содержит самую общую информацию о службе; systemd управляет не только службами, но и устройствами, точками монтирования, таймерами и т. п. Эта секция конфигурационного файла определяет наиболее общие свойства, которые могут быть присущи любому юниту. Зависимость описывается при помощи директивы After.
Есть и другие директивы.
Requires — перечисляет юниты, от которых зависит данный юнит. Связанные юниты запускаются всегда параллельно.
Wants — похожее на Requires, но более расслабленное требование. Если связанные юниты не запустятся, юнит продолжит работать.
BindsTo — похожая на Requires директива с той разницей, что в случае её использования, если связанный юнит завершит работу, также работу завершит и текущий юнит.
After — юнит, который должен запуститься, прежде чем сможет запуститься текущий юнит.
Conflicts — указанный в этой директиве юнит не может быть запущен одновременно с текущим юнитом. Запуск юнита завершит перечисленные в директиве юниты.

Следующая секция, [Service], содержит информацию о сервисе. Сюда включаются настройки, относящиеся именно к сервисам. ExecStart определяет расположение файла, запускающего программу и аргументы, с которыми он будет вызван. По аналогии с ExecStop, ExecReload описывает команду, которая запускается соответственно при остановке и перезапуске процесса. Type описывает способ запуска процесса.

Type — описание типа процесса (возможные значения: simple, oneshot, forking).

PIDFile — путь до файла, содержащего PID процесса (необходимо для Type=forking).
ExecStart — команда и параметры, необходимые для запуска процесса.
ExecReload — команда и параметры, необходимые для перезагрузки процесса.
ExecStop — команда и параметры, необходимые для остановки процесса.
Restart — автоматический рестарт упавшего процесса (возможные значения, в зависимости от вида завершения процесса: always, on-success, on-failure).

Третья секция — [Install]. Она содержит рекомендации по установке конкретного юнита, указывающие, в каких ситуациях он должен быть активирован. Сервис nginx запускается при активации юнита multi-user.target. Nginx будет активироваться при переходе в состояние multi-user.target, то есть при каждой обычной загрузке.

systemctl

Для управления процессами используется **программа systemctl**. Мы можем передать ей действие и имя сервиса, над которым действие нужно произвести. Основные действия — это:

```
start
stop
restart
status
```

Если вы остановите юнит, он может запуститься снова: при перезагрузке или если выполнятся необходимые для его запуска условия. Чтобы этого избежать, юнит нужно деактивировать:

```
systemctl disable nginx
```

Чтобы заново его активировать, используется:

```
systemctl enable nginx
```

Все новые systemd-юниты после создания также необходимо активировать.

Мы можем просмотреть systemd-юнит при помощи команды:
systemctl cat nginx

Или отредактировать при помощи systemctl edit.
После редактирования конфигурационного файла необходимо запустить:

```
systemctl daemon-reload
```

Чтобы systemd узнал об изменениях.

Для того чтобы **создать новый serviced-юнит,** необходимо выполнить следующее:

1) написать юнит-файл, описывающий запуск и остановку сервиса;
2) сохранить его в директории /etc/systemd, указав правильное расширение файла, соответствующее типу юнита;

3) запустить systemctl enable.

Теперь можно запустить юнит при помощи systemctl start.
Systemd поддерживает логирование событий, возникающих в процессе работы с юнитом. Для того чтобы работать с этими событиями, используется утилита journalctl. Мы можем посмотреть все события:

```
journalctl
```

Или события, связанные с определённым юнитом:

```
journalctl -u nginx
```