Курс «Администратор Linux»

LAMP: nginx + apache

Занятие # 23

Дмитрий Молчанов
Григорий Ожегов

Цель занятия

Научиться устанавливать и настраивать nginx и apache.

План

- Установка/настройка apache (Apache httpd)
- Установка/настройка nginx

Apache установка, управление

- Установка: yum install httpd httpd-tools
- Управление:
  - systemctl (stop|start|restart|reload) httpd
  - apachectl (stop|start|graceful|restart|**configtest**)

Apache настройка

Конфигурация http-сервера apache в centos/rhel/fedora хранится в каталоге /etc/httpd.

- /etc/httpd (httpd собран с этим каталогом как с основным)
  - conf - основные файлы конфигурации
    - httpd.conf - основной/корневой файл конфигурации
    - magic - файл с описанием как определять тип данных в файлах для модуля mod_mime_magic
  - conf.d - дополнительные файлы конфигурации, поставляемые с различными пакетами или заданные пользователем
  - conf.modules.d - настройки модулей apache (подключается из основного конфига)
  - modules -> симлинк на каталог с подключаемыми модулями
  - logs -> симлинк на /var/log/httpd
  - run -> симлинк на /var/run (pid-файлы, shm-файлы)

Apache основные параметры

В корневом контексте:

- Listen - директива (Listen [addr:]port) определяет какие сокеты открывать, где слушать http-запросы. Отсутствующий адрес подразумевает все адреса. Может употребляться несколько раз
- User - effective user для процесса httpd
- Group - effective group для процесса httpd
- EnableSendfile - разрешает использование вызова sendfile
- DocumentRoot - директория которая соответствует ресурсу “/” http сервера.
- ErrorLog, CustomLog - настройка лог-файлов ошибок и запросов.
- LogFormat - настройка формата лога запросов.

sendfile

Системный вызов копирующий данные между двумя файловыми дескрипторами. Зачастую более эффективен, чем в userspace читать из одного(например файл) и записывать в другой (сокет) поскольку выполняется в ядре.

Apache основные параметры

В корневом контексте:

- AccessFileName - имя файла (.htaccess) в котором могут содержаться переопределения некоторых опций конфигурации для конкретного каталога в файловой системе.
- AllowOverride - None или набор (1 или более) групп директив переопределение которых разрешено (http://httpd.apache.org/docs/2.4/mod/core.html#allowoverride)

.htaccess

.htaccess - Мощный, удобный и популярный механизм runtime управления apache, который широко применяется на виртуальных хостингах. С помощью этого механизма можно управлять настройками mod_rewrite (RewriteRule), доступа, аутентификации/авторизации.

Но несмотря на всю функциональность и удобство - есть и обратная сторона, при большой нагрузке на сервер использование .htaccess увеличивается количество вызовов open, т.к. производится попытка открыть файл .htaccess в каждой директории от самой верхней в которой разрешено использовать .htaccess (AllowOveride отличен от None) до самой нижней. Обычно на это тратится 10-20% sys time.

Apache основные параметры

<ifModule> - позволяет применять конфигурационные параметры только в том случае, если подключен определенный модуль, это позволяет создавать очень универсальные конфигурации (типичный пример конфигурация по-умолчанию)

<Directory> - Контекст который позволяет переопределить какие-либо настройки глобального контекста для какой-либо директории на файловой системе и всего её содержимого, включая потомков. Что-то может быть переопределено контекстом <Files> или <FilesMatch>.

Options - директива которая позволяет управлять “возможностями сервера” в контексте какой-либо директории +option включает возможность, -option - выключает. Например -Indexes отключает возможность просмотра содержимого директории (File browse). Подробнее (http://httpd.apache.org/docs/2.4/mod/core.html#options)

Виртуальные сервера

По умолчанию основной контекст сервера содержит конфигурацию единственного web-сервера. С помощью контекста <VirtualHost> привязанного к ip:port или к ip:port и ServerName мы можем переопределить часть конфигурации (DocumentRoot как минимум) основного сервера для определенных запросов выделяя их по адресу+порту назначения или имени.

Порядок просмотра такой:

1) ip:port - best match
2) ip:port + server name если в п.1 было несколько совпадений.

Таким образом мы получаем возможность делать несколько виртуальных http-серверов привязывая их к разным именам или адресам или группируя по адресам.

Apache основные параметры

LoadModule - подключение расширения. Например MPM (Multi-Processing Module) или обработчика для скриптов (mod_php, mod_python) или управления форматом логов (mod_log_config).

Include, IncludeOptional - подключение в конфигурацию дополнительных конфигурационных файлов.

Apache MPM:

mpm_prefork - модуль при котором один процесс обрабатывает один запрос. Наиболее популярный и стабильный (по сведениям интернетов) модуль. Также наиболее ресурсоемкий.

mpm_worker - гибридный модуль, который позволяет каждому процессу обрабатывать несколько запросов разными потоками.

mpm_event - инкарнация worker-модуля с привлечением event-driven обработки соединений в результате которого треды обрабатывают только активные соединения (которые пишут или читают).

mpm_prefork

StartServers - Количество предзапускаемых процессов

MinSpareServers - Минимальное количество процессов “в горячем резерве”, при меньшем количестве свободных процессов будут запущены дополнительные

MaxSpareServers - Максимальное количество процессов “в горячем резерве”, при большем количестве свободных процессов лишние будут закрыты.

MaxConnectionsPerChild - максимальное количество соединений принимаемых одним процессом, после превышения (если лимит больше 0) процесс завершается и, соответственно, заменяется новым. Спасает от протечек памяти.

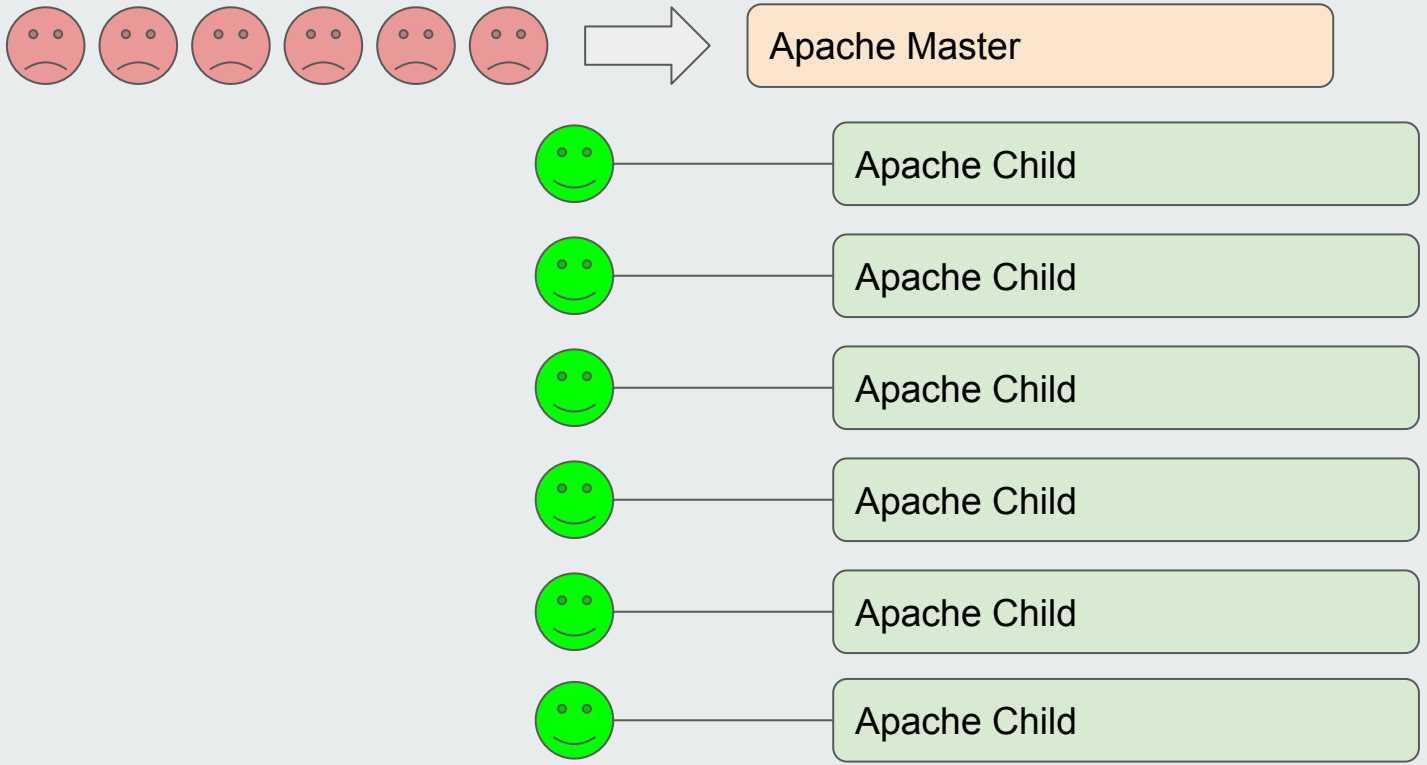
MaxRequestWorkers - Максимальное количество клиентов обрабатываемых сервером.

introducing nginx

Как уже было сказано apache + mpm_prefork очень ресурсоемкая схема, т.к. на обработку одного соединения выделяется целый процесс со всеми ресурсами типа памяти.

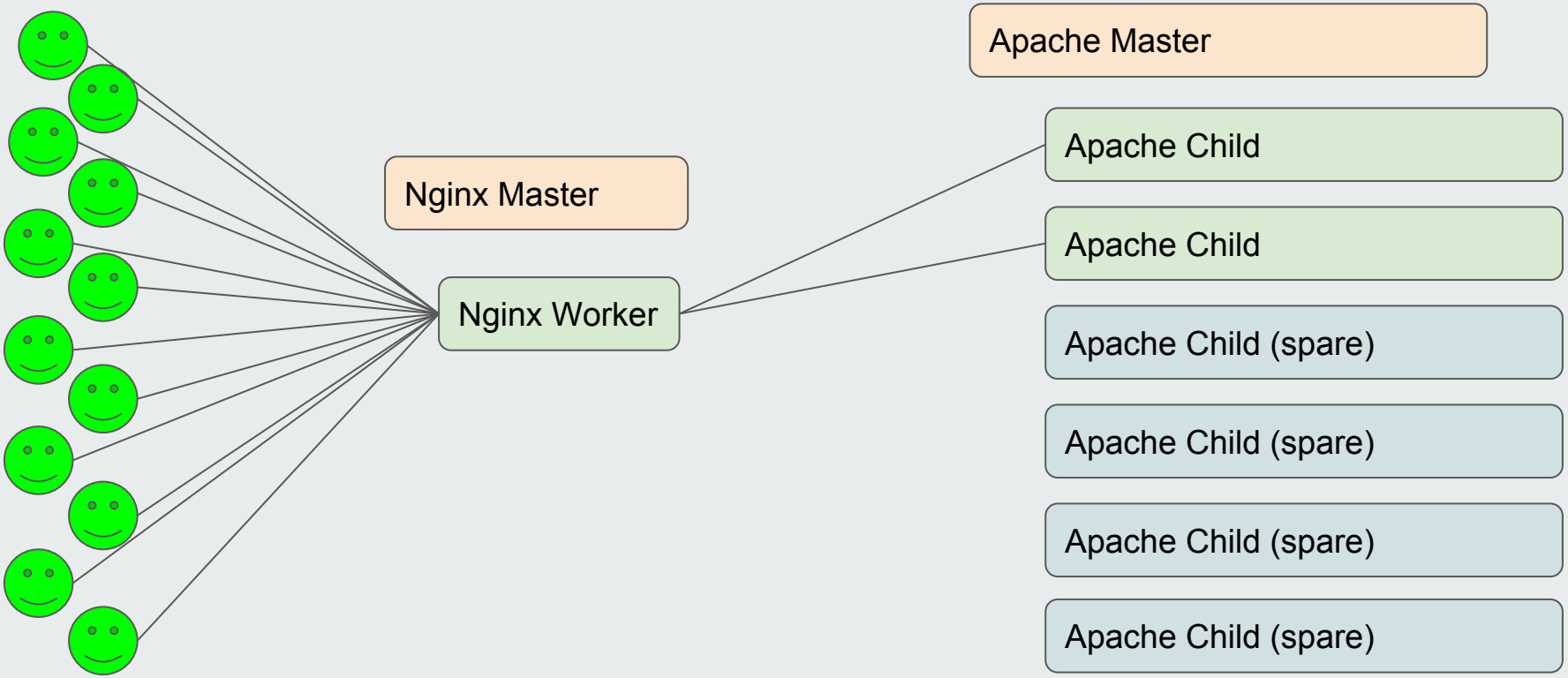
Тут нам на помощь приходит nginx.

introducing nginx

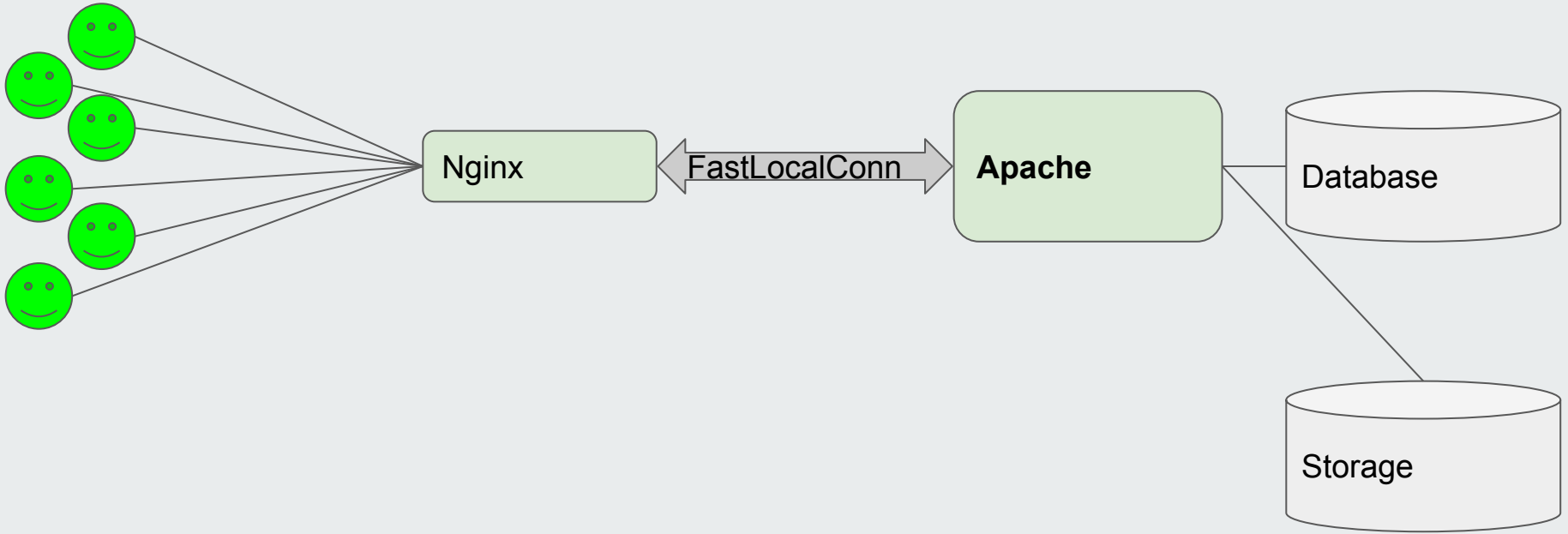


Ключевой особенностью nginx является то, что он обрабатывает много соединений однима однопоточным процессом (нет необходимости отслеживать блокировки) и тратит очень мало ресурсов на хранение информации о неактивном соединении за счет чего потребляя ресурсов сопоставимо с одним worker'ом apache может обслужить в 1000 раз больше соединений, т.к. все они пишут и читают не одновременно.

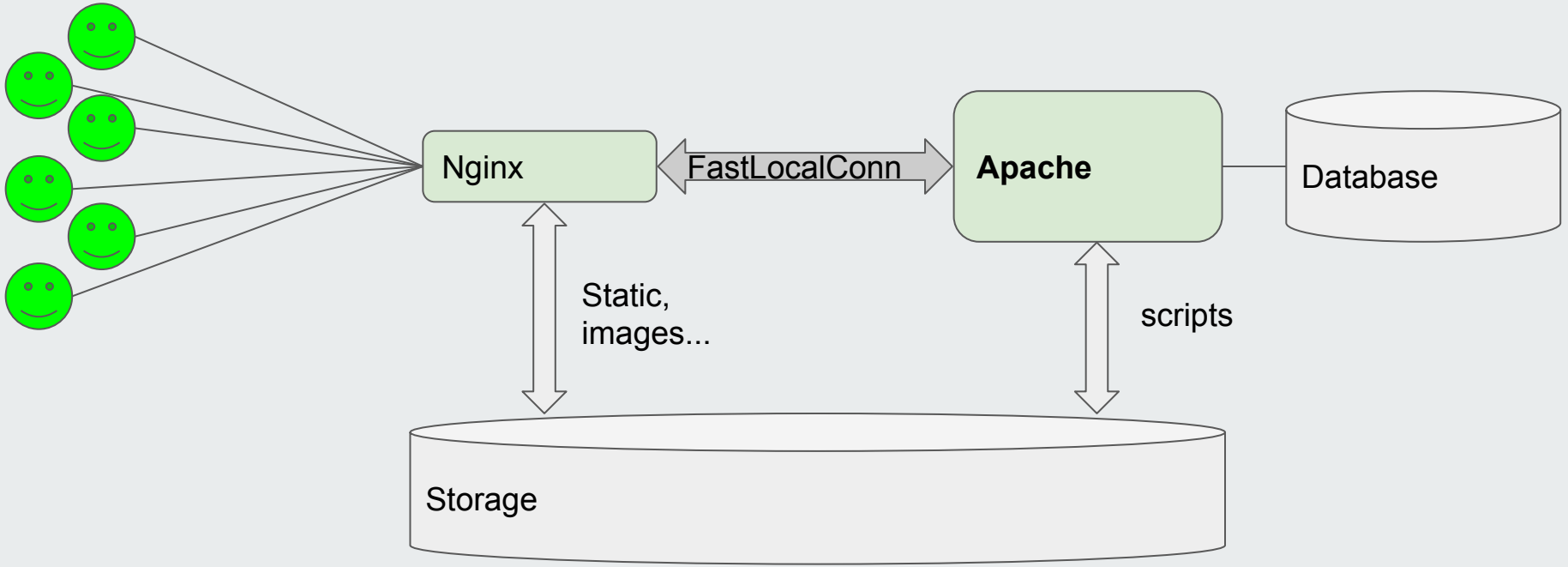
apache + nginx



apache + nginx example 1



apache + nginx example 2



nginx установка, управление

nginx, один из немногих пакетов которые рекомендуется ставить “от производителя”, подключая репозиторий создателей nginx.

- Установка:
  - yum install http://nginx.org/packages/centos/7/noarch/RPMS/nginx-release-centos-7-0.el7.ngx.noarch.rpm
  - yum install nginx
- Управление:
  - systemctl (start|stop|restart|reload) nginx
  - pkill -1 nginx - перечитать конфигурацию, перезапустить worker’ов, перечитать TZ и переоткрыть логи
  - pkill -WINCH nginx - плавно завершить все процессы
  - pkill -USR1 nginx - переоткрыть логи
  - подробнее: http://nginx.org/ru/docs/control.html

nginx конфигурация

Конфигурация чаще всего располагается в /etc/nginx/nginx.conf
Подобно конфигурации httpd состоит из нескольких контекстов:

- корневой (основная функциональность) - erro_log, pid, env, load_module...
- events - настройка обработки соединений
- http - функциональность http-сервера
  - server
    - location

Важный момент - в отличе от apache ни один из контекстов верхнего уровня не описывает конфигурацию http-сервера обслуживающего запросы. для обработки http-запросов должен быть задан хотя бы один контекс server.
Директив великое множество и у многих различается контекст конфигурации в которых их можно применять. Подробнее смотрите в документации.

ngx_http - основные директивы

- include - влюкчение дополнительных файлов
- access_log - лог запросов
- log_format - формат лога запросов
- server - определение контекста server
- sendfile - включение/выключение использования вызова sendfile.
- default_type - mime-тип ответ по-умолчанию
- gzip - включение/выключение сжатия

Контекст server

В этом контексте определяются настройки http-сервера принимающего соединения.

пример

```
server {
    listen       80;
    server_name  localhost;
    location / {
        root   /usr/share/nginx/html;
        index  index.html index.htm;
    }
    error_page   500 502 503 504  /50x.html;
    location = /50x.html {
        root   /usr/share/nginx/html;
    }
}
```

Контекст server

Ключевые директивы:

- listen - описание пар [ip:]port к которым этот контекст привязан. Опциями можно указать слушается там ssl или plain http.
- server_name - список имен которые обрабатываются этим контекстом. Первое указанное имя является основным и хранится в переменной $server_name
- root - аналог DirectoryRoot в apache, каталог соответствующий “/” сервера, в нем уже ищется URN.
- location - контекст location (какой-то части пути URN). Позволяет переопределить настройки для части запросов основываясь на пути к ресурсу (URN). Например отключить протоколирование запросов или указать бэкенд сервер к которому надо проксировать запрос. В описании location можно использовать regexp.

Полезные модули

- ngx_http_upstream_module - модуль позволяющий задавать группы серверов для балансировки проксирования.
- ngx_http_proxy_module - модуль реализующий проксирование запросов (передачу их бэкенду) по протоколу http[s]. Его брат-близнец ngx_http_fastcgi_module работает с протоколом fastcgi.
- ngx_http_userid_module - модуль который выставляет userid cookie для дальнейшей идентификации клиента
- много других модулей, подробнее: http://nginx.org/ru/docs/

Спасибо за внимание

Дмитрий Молчанов
Григорий Ожегов