Курс «Администратор Linux»

Основы LAMP

Занятие # 23

Дмитрий Молчанов
Григорий Ожегов

Цель занятия

Получить представление о:

- LAMP-стеке
- протоколе HTTP
- концепции виртуальных серверов
- основной проблематики связанной с работой веб-сервисов

Немного истории

Веб появился в конце прошлого века. И с тех пор неумолимо эволюционирует каждую минуту. Эволюционирует не только содержимое, но и технологическая база.

Начиналось все с простого желания людей делиться информацией.

Протокол http появился в 1992 году в версии 0.9 но широкое распространение получил только с версией 1.0 в конце 90х. Тогда, после стандартизации HTML в 1993, начал с сумасшедшей скоростью развиваться WorldWideWeb или Интернет в современном понимании.

Немного истории

Итак, в конце 90х годов прошлого века сошлись несколько технологических и не только факторов:

- СПД - сеть передачи данных, Internet
- Протоколы: IP, DNS, HTTP
- Данные: HTML
- Развитие персональных компьютеров (до этого были мэйнфреймы с терминалами)
- Энтузиасты, которые понимали, что к развитию ведет только тот путь, на котором данными деляться.

Это все дало толчок той машине, которая крутится и по сей день.

Немного истории

Изначально на одной паре ip:port (которыми оперирует tcp), мог жить только один сайт и это не было проблемой, ведь:

A. ip-адресов было много (что-то около 4 миллиардов)
B. Сайтов и “информаториев” было немного
C. Эти сайты жили там, где была техническая база для их размещения и поддержания.

Но популярность Интернет росла, доступность вычислительных ресурсов - тоже. Это привело к появлению компаний, которые занимались хостингом и к буму “доткома”.

HTTP/1.1

Так появился HTTP/1.1 - протокол который поддерживал заголовок Host, который позволял на одном адресе поддерживать много ресурсов - виртуальных серверов.

HTTP Получил широкое распространение, поскольку легок в понимании и реализации и, как большинство базовых протоколов Интернет - текстовый, расширяем, гибок.

Сейчас он используется в качестве транспорта для множества разных протоколов.

HTTP простой

- Текстовый (не бинарный)
- 7 базовых методов
  - 3 самых популярных – GET, POST, HEAD
- Запросы и ответы могут быть легко
  - сформированы «ручками»
  - Прочитаны глазами или отфильтрованы
- простой формат запроса:
  - Method URI ProtocolVersion
  - +headers

Пример HTTP-запроса

Показываем в консоли.

Почему LAMP

Linux	Бесплатный, OpenSource, богатые возможности, динамично развивается, требует мало ресурсов, прост в настройке.
Apache	+ Кроссплатформенный
MySQL	+ низкий порог входа для начала использования, возможностей mysql, например, хватает для 99% веб-проектов.
PHP	

LAMP сейчас

Канонически LAMP и сейчас это Linux+Apache+MySQL+PHP, но в целом эта аббревиатура является синонимом “хостинга” веб-проектов.

Apache давно уже используется только с nginx и выполняет роль application-контейнера, в чем он хорош.

Вместо MySQL многие используют PostgreSQL или Mongodb.

Вместо PHP могут использовать python, perl, ruby или любой другой язык.

URL

URL - Uniform Resource Locator, принятый в Интернет способ указывать где находится ресурс. Формируется по схеме:
schema://[*user*[:*password*]@]*hostname*[:*port*]/[path/to/resource][?[arg1=v[&arg2=v]]]

где *schema* - используемый протокол. Наиболее часто используемые протоколы - http, https, ftp, smb
URL является удобным и универсальным способом записывать практически любые ссылки на сервисы.

URL примеры

- https://otus.ru/lessons/ - обращение к ресурсу расположенному по пути /lessons/ на сервере otus.ru по протоколу https (http over ssl).
- redis://localhost:6379 - обращение к redis-серверу на localhost и порту 6379. Если не указывать, то будет использоваться порт по-умолчанию для данного протокола (возможно указан в библиотеке, а возможно смотрится в /etc/services).
- mysql://myusr:secret@192.168.4.1:3307/db_1/sometable - ссылка на данные в mysql, которые доступны в таблице sometable базы db_1 на сервере 192.168.4.1 с нестандартным портом 3307.

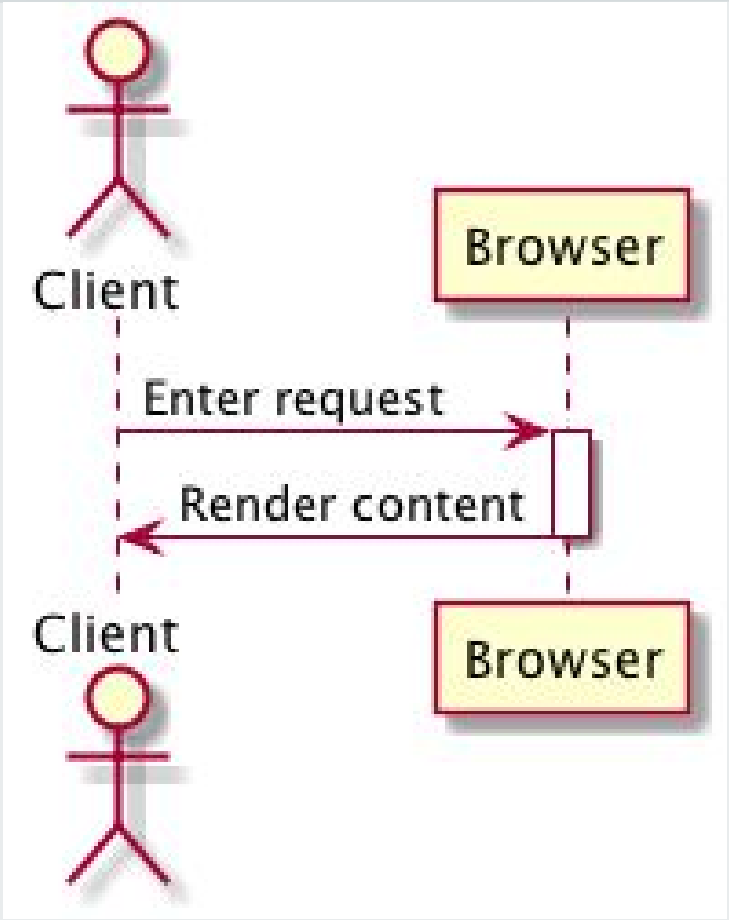
Как работает HTTP и WWW

- HTTP использует для работы 80/tcp
- HTTPS - 443/tcp

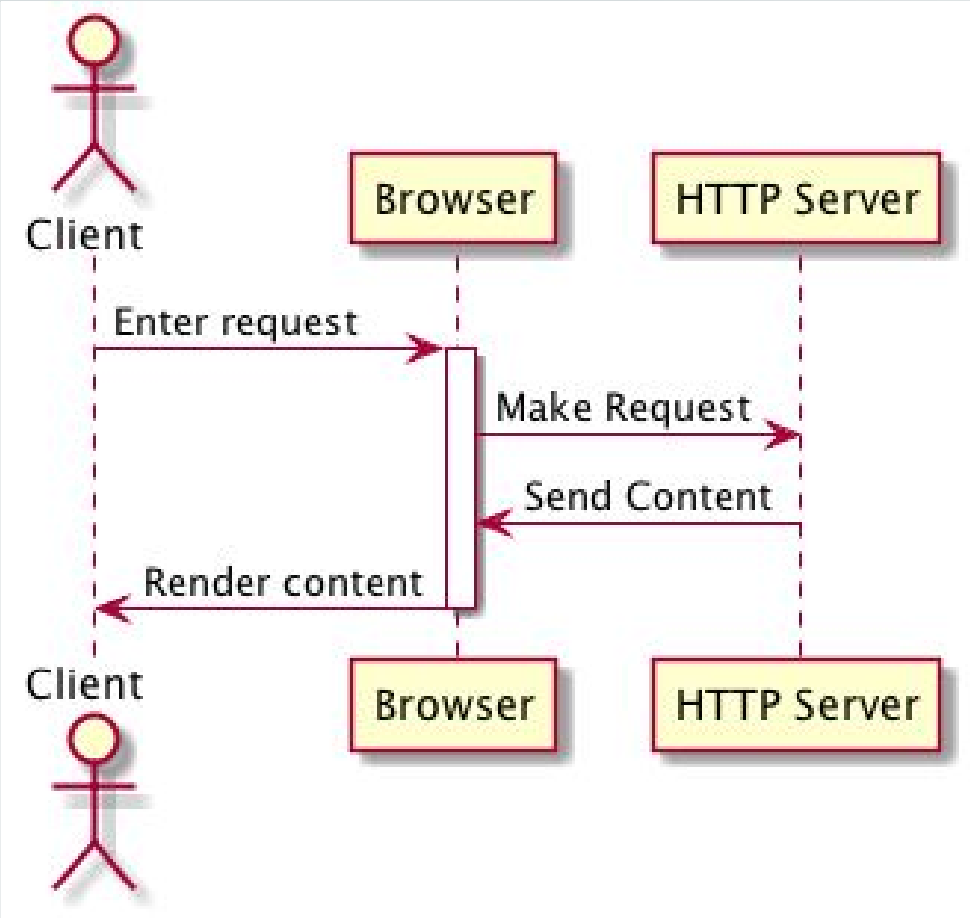
Но в принципе веб-сервер может работать на любом свободном порту.

Рассмотрим как это все работает

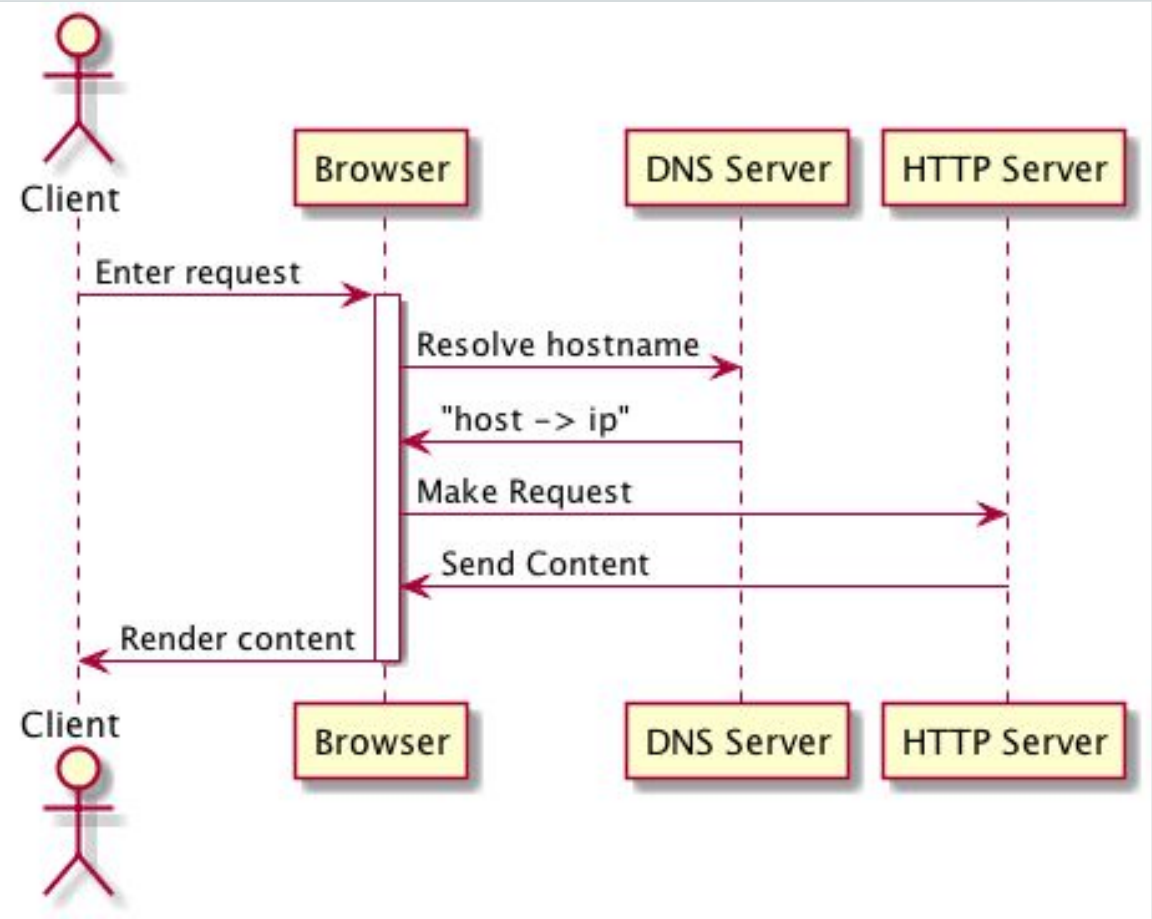
HTTP-запрос


Client
Browser
Enter request
Render content
Client
Browser

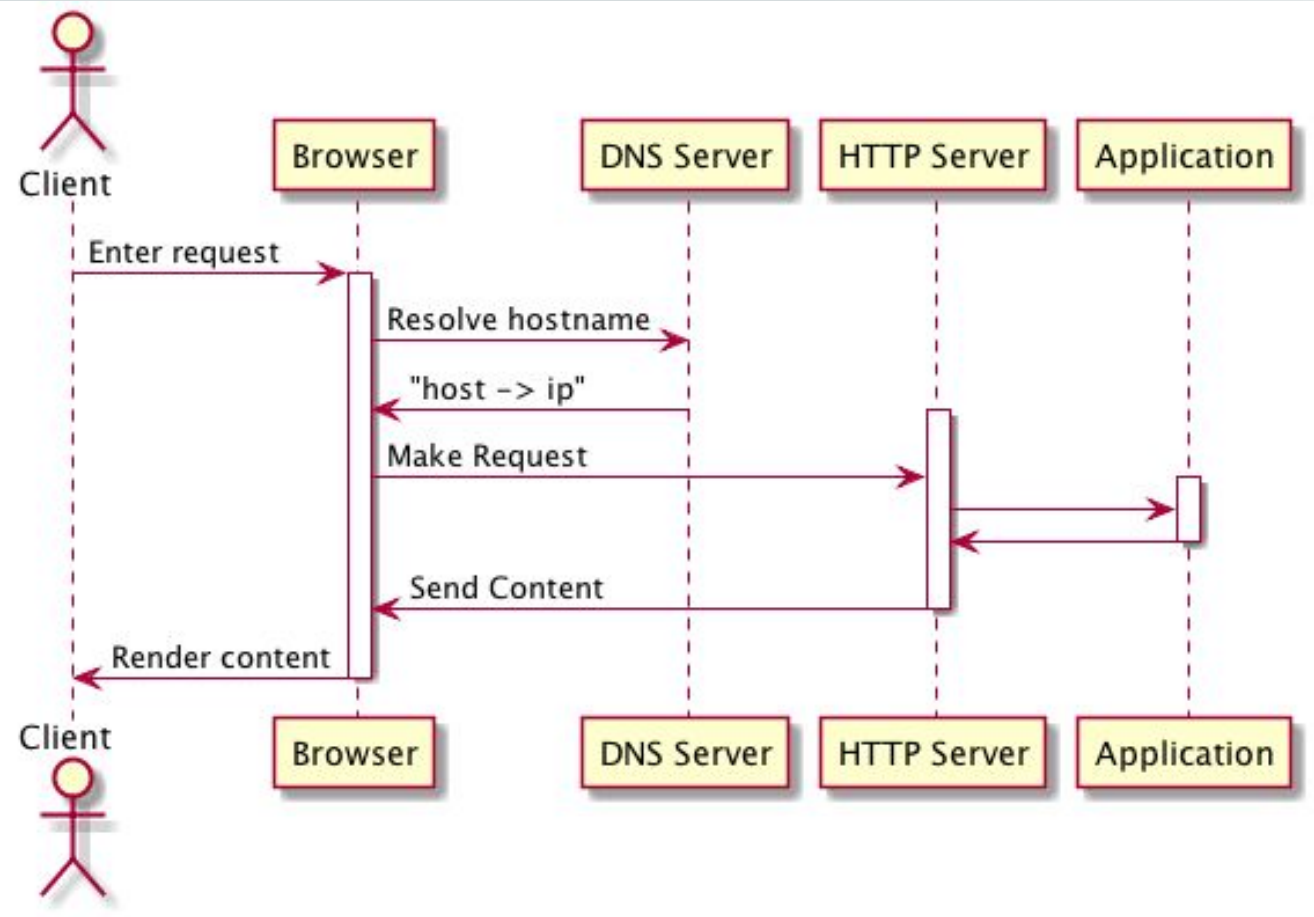
HTTP-запрос


Client
Browser
HTTP Server
Enter request
Make Request
Send Content
Render content
Client
Browser
HTTP Server

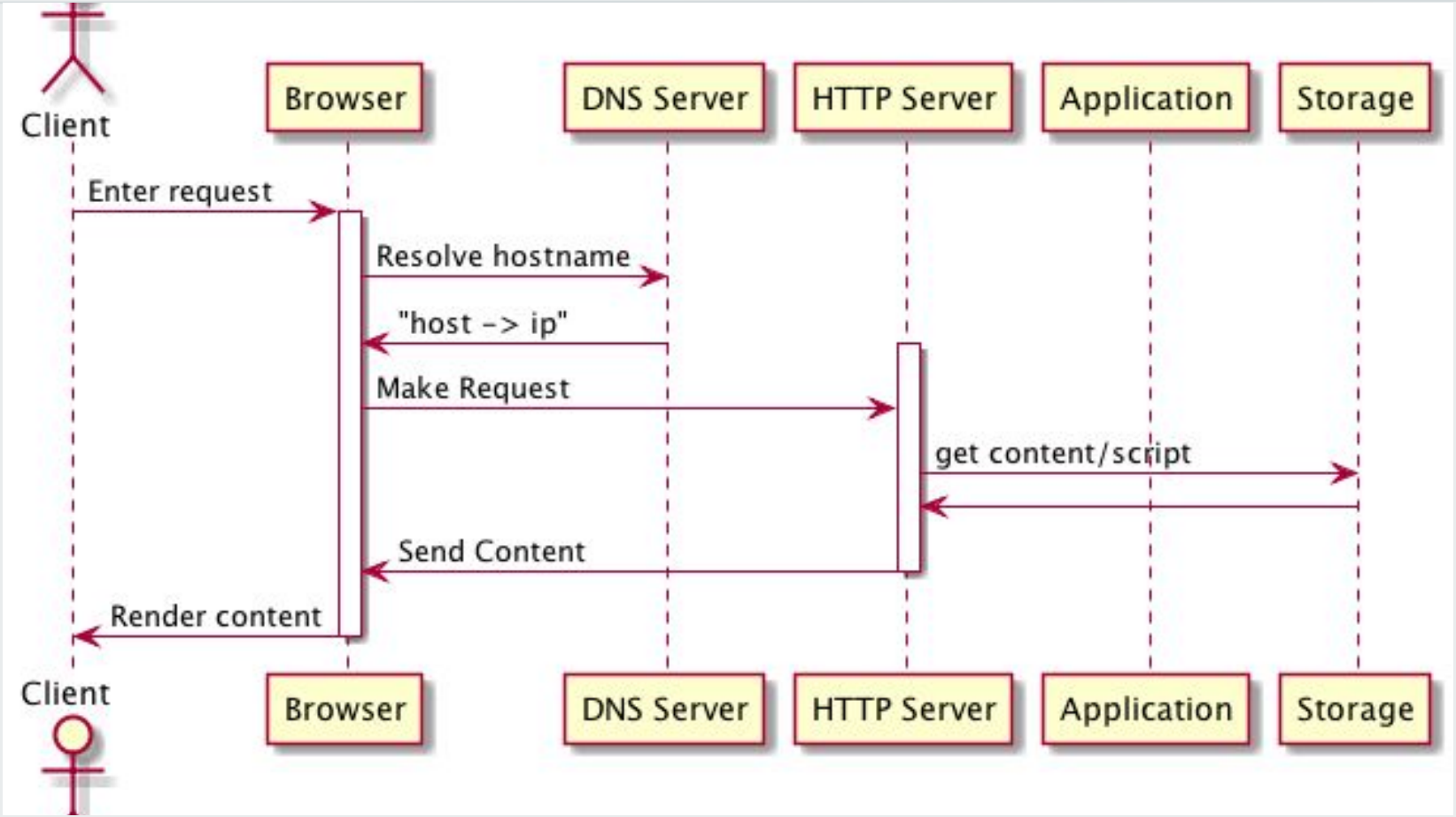
HTTP-запрос


Client
Browser
DNS Server
HTTP Server
Enter request
Resolve hostname
"host -> ip"
Make Request
Send Content
Render content
Client
Browser
DNS Server
HTTP Server

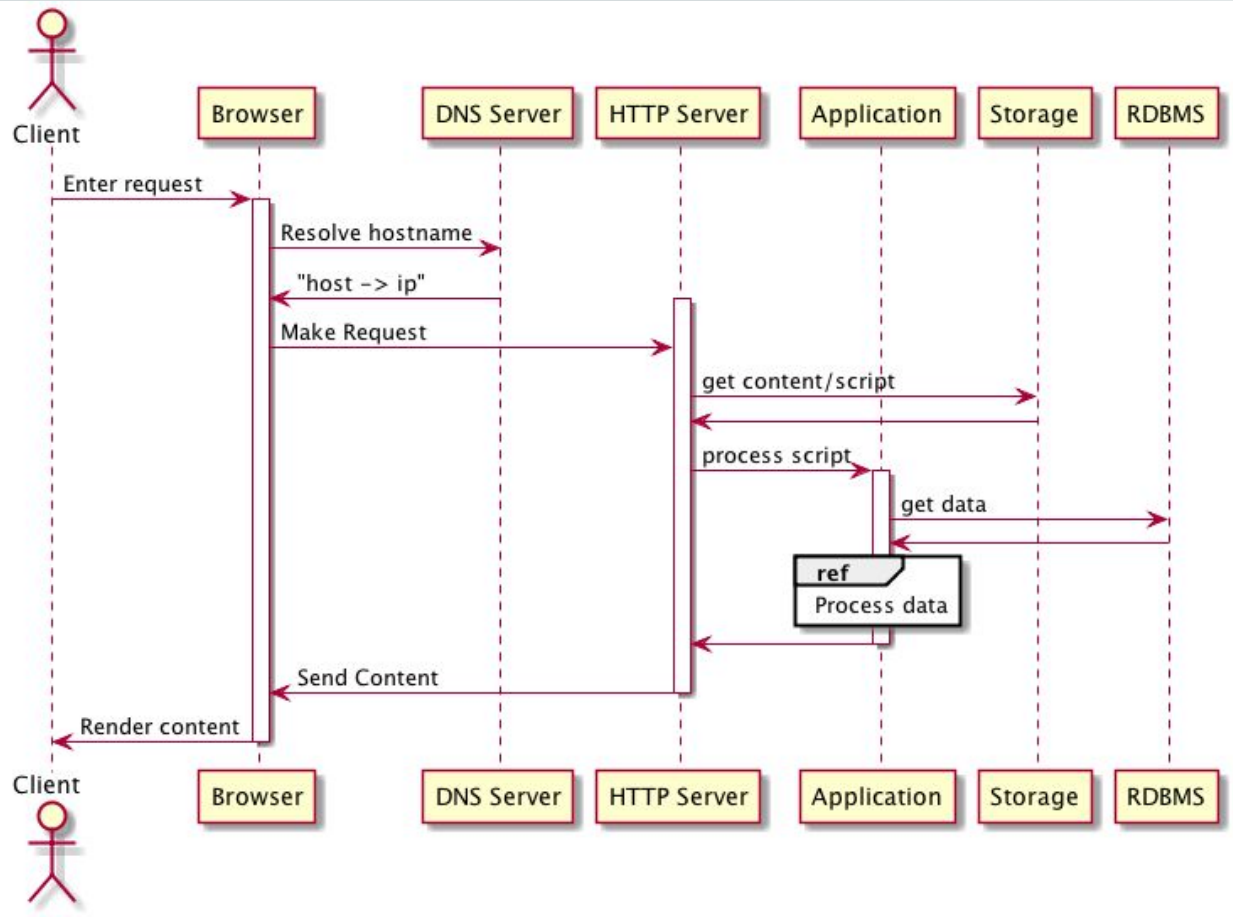
HTTP-запрос


Client
Browser
DNS Server
HTTP Server
Application
Enter request
Resolve hostname
"host -> ip"
Make Request
Send Content
Render content
Client
Browser
DNS Server
HTTP Server
Application

HTTP-запрос


Client
Browser
DNS Server
HTTP Server
Application
Storage
Enter request
Resolve hostname
"host -> ip"
Make Request
get content/script
Send Content
Render content
Client
Browser
DNS Server
HTTP Server
Application
Storage

HTTP-запрос


Client
Browser
DNS Server
HTTP Server
Application
Storage
RDBMS
Enter request
Resolve hostname
"host -> ip"
Make Request
get content/script
process script
get data
ref
Process data
Send Content
Render content
Client
Browser
DNS Server
HTTP Server
Application
Storage
RDBMS

Концепция виртуальных серверов и контекстов

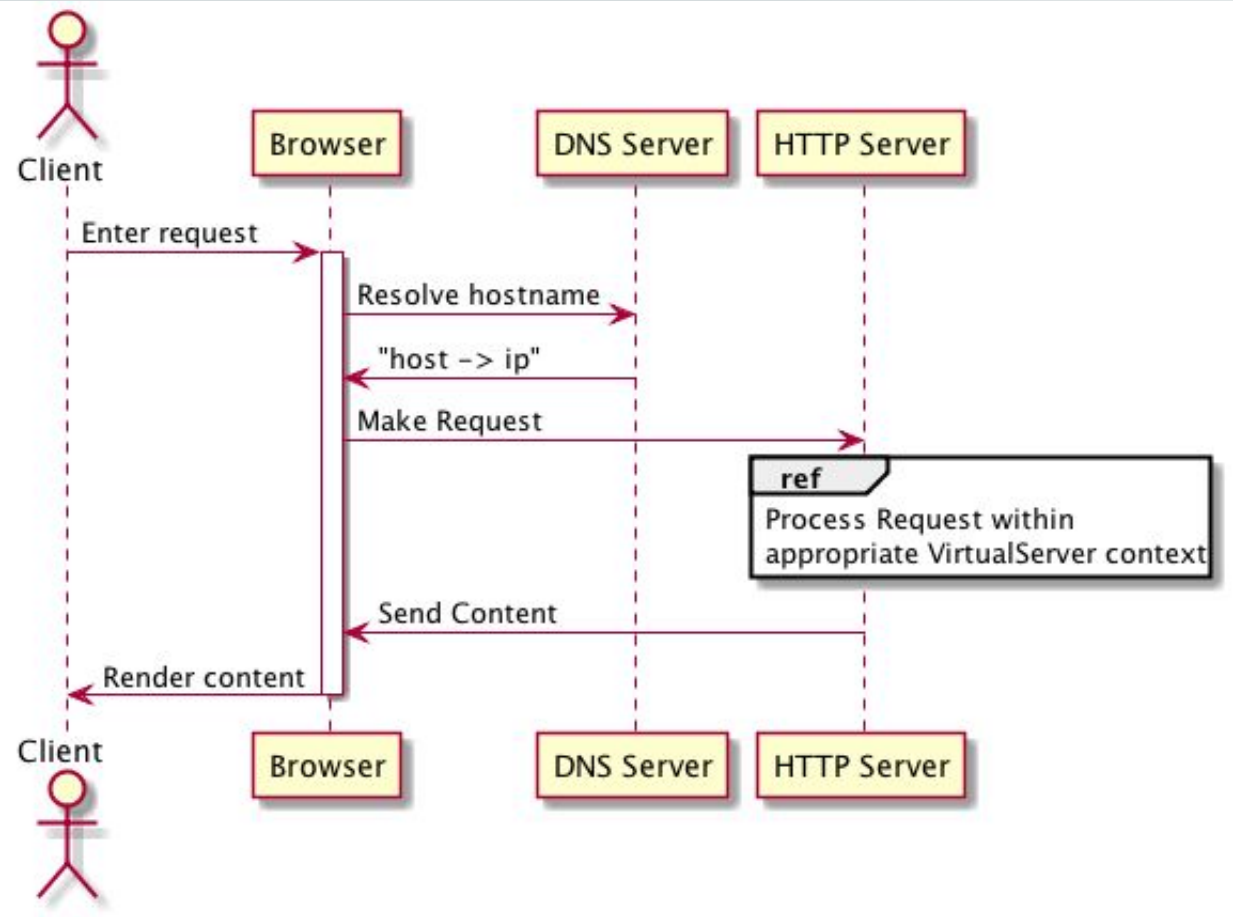
Концепция виртуальных серверов сходна с Наследованием ООП.

Есть глобальный контекст настроек, который содержит в себе как настройки относящиеся к настройкам http-сайта, так и настройки относящиеся к серверу в целом. в этом глобальном контексте размещены дополнительные контексты виртуальных серверов, которые наследуют соответствующие настройки из глобального контекста и могут их переопределять.

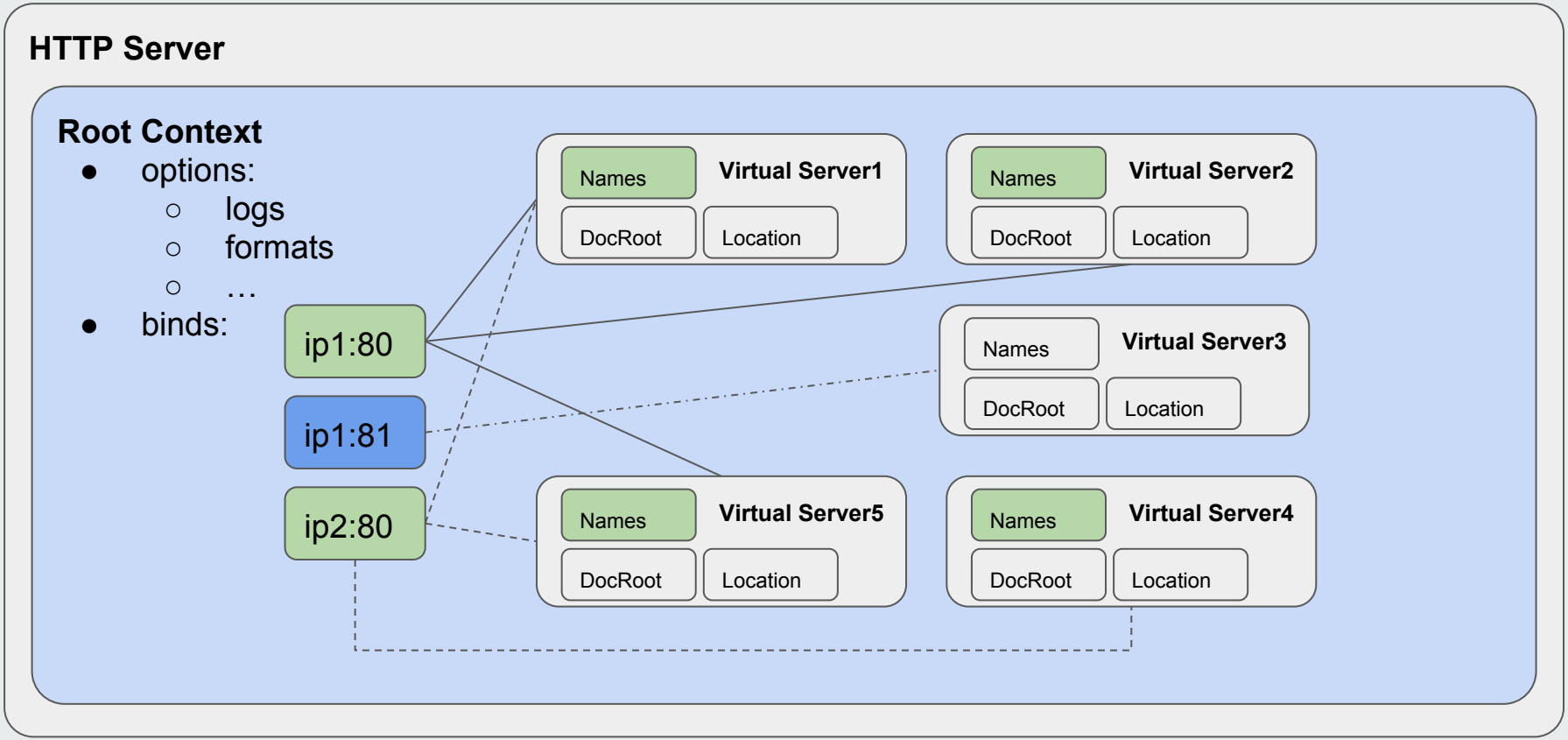
Концепция виртуальных серверов и контекстов

```
server ip:port{
        virtual_server{
                name
                doc_root
        }
        virtual_server{
                name
                doc_root
        }
}
```

Жизнь HTTP-запроса



Концепция виртуальных серверов и контекстов



Наиболее часто употребляющиеся контексты

apache	nginx	значение
глобальный	http	основные настройки
VirtualHost	server	виртуальный сервер
Location, LocationMatch	location	описание URN
Files, FilesMatch	location	описание filename

Проблематика

- медленные клиенты и расходование ресурсов
- problem 10000 (10000 процессов? врядли!). Рождение nginx и асинхронной обработки событий
- Количество конфигов (1 конфиг на одно имя?! нет ServerAlias)
- ssl+sni
- DDoS

Медленные клиенты

Во времена *интернета-по-модему* эта проблема стояла особенно остро - WWW набирал популярность, пользователь становился требовательнее, контент - толще, а вот каналы за всем этим не успевали.

Так медленные клиенты, те, кто качает контент, зачастую тяжелый, медленно, начали становиться проблемой. На какое-то время эта проблема потеряла остроту, но с развитием мобильного интернета снова стала актуальной.

Проблема 10000 соединений

Она тесно связана с предыдущей задачей - медленных клиентов. Только здесь задача - обслужить одним сервером 10000 соединений. Эта проблема подтолкнула к развитию асинхронных веб-серверов, например nginx. Ключевой особенностью nginx является то, что он тратит катастрофически мало ресурсов (памяти) на обработку одного соединения.

Количество конфигов

С ростом количества виртуальных серверов, зачастую однообразных, количество конфигов которые приходилось обслуживать стало проблемой. Появились методики обработки тысяч сайтов одним конфигом.

Помимо того, что большое количество конфигурационных файлов это само по себе неудобно, большое количество контекстов виртуальных серверов тоже влечет за собой расход памяти.

SSL

Основной проблемой связанной с SSL, точнее https (http over ssl) является то, что ssl-сессия устанавливается до http-обмена. А с этим связан ряд других факторов:

- ssl-сертификат выдается на конкретное имя, но можно использовать wildcard или dns alt names
- ssl-сертификат привязывается к виртуальному серверу и при установке ssl-соединения выбирается первый загруженный, который далеко не всегда является сертификатом того виртуального сервера к которому обращались. Это вызывает вопросы у пользователя.

DDoS-атаки

Вообще это проблема не только http-сервисов, но тут очень много “болевых точек”:

- Поиск и эксплуатация для исчерпания ресурсов “медленной страницы”
- Отправ больших запросов для исчерпания места или канальной емкости
- Большое количество запросов вообще
- Syn-flood (ядро передает уже установленное соединение в приложение)

Ну и поводом для ddos-атак часто является бизнес/сервис конкурентов, которые сейчас активно выводятся в онлайн.

Спасибо за внимание

Дмитрий Молчанов
Григорий Ожегов