

Введение в балансировку нагрузки на примере Nginx (#9)



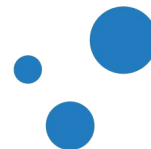
- Что такое «Балансировщики нагрузки» и зачем они нужны
 - Сравнение основных решений
 - Смотрим в будущее
-
- Погружаемся в Nginx
 - Домашнее задание



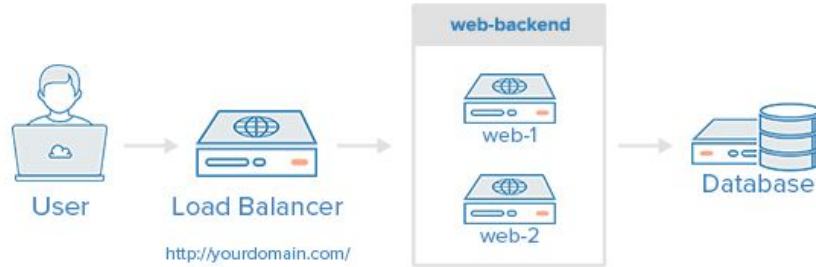
1

Что такое «Балансировщики нагрузки» ?

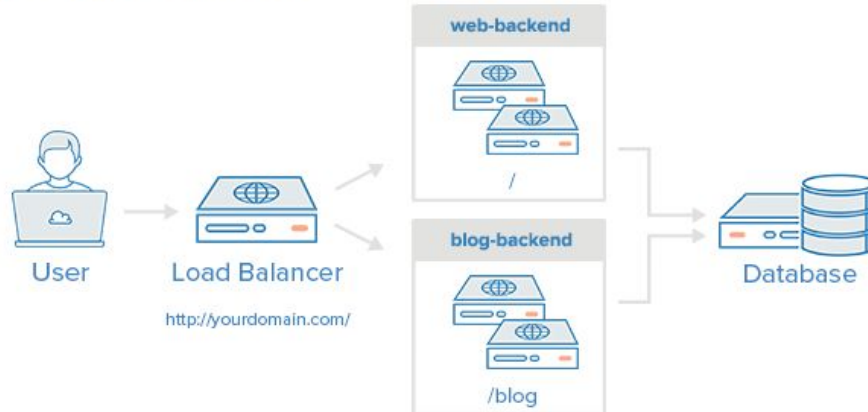
Что есть «балансировщик нагрузки» ?



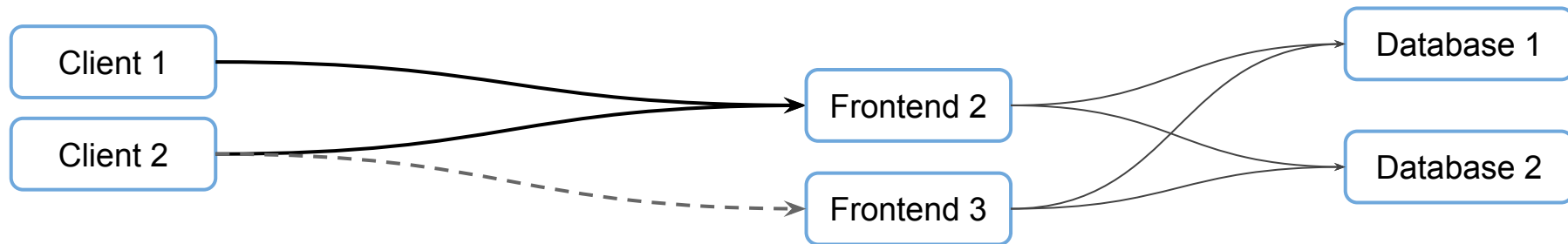
Layer 4 Load Balancing



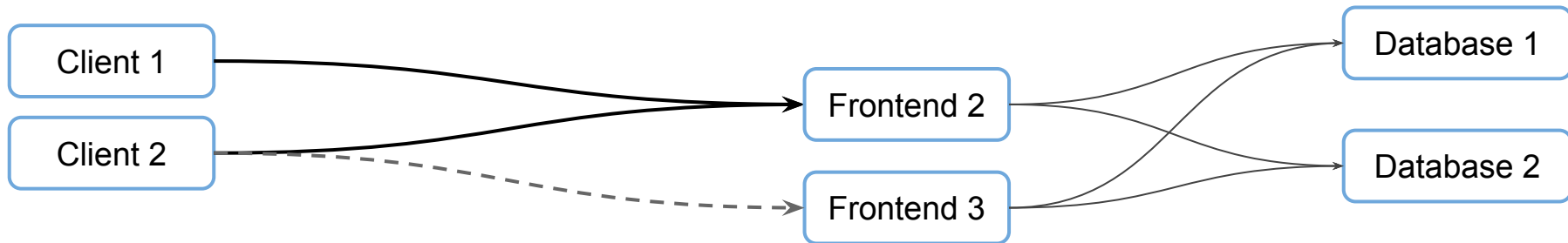
Layer 7 Load Balancing



Зачем есть «балансировщик сетевой нагрузки» ?

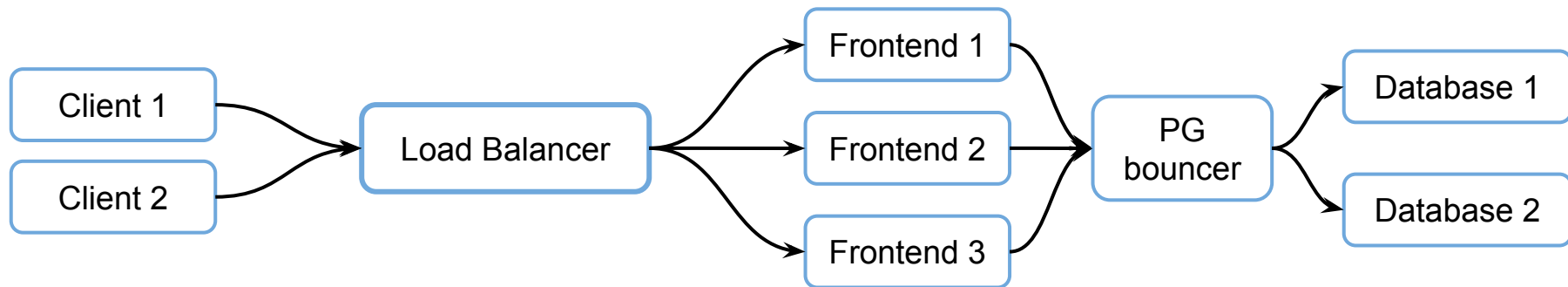


Зачем есть «балансировщик сетевой нагрузки» ?



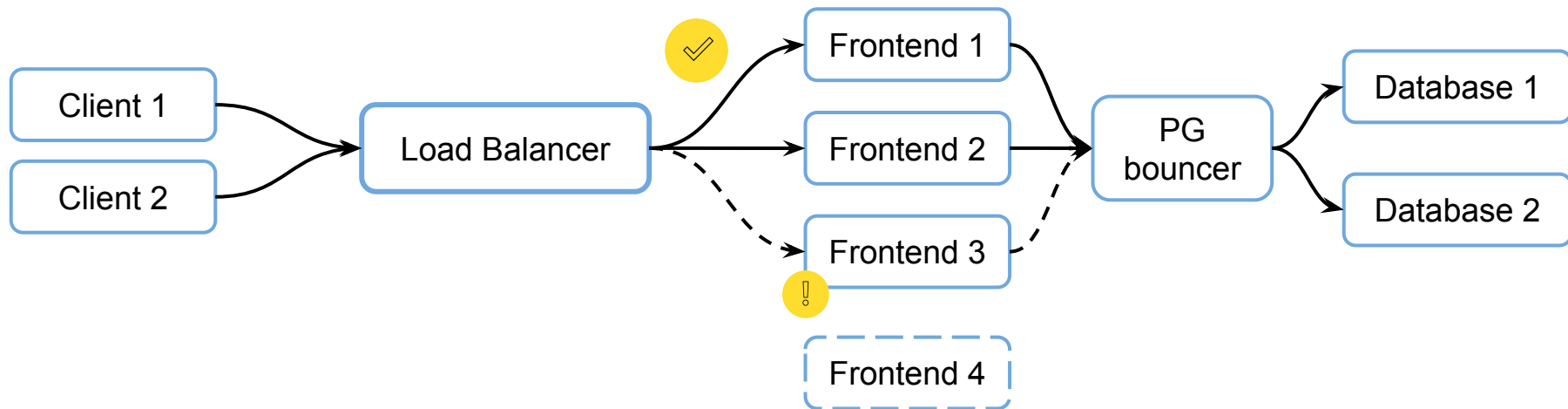
- организация отказоустойчивости
- оптимизация вычислительных ресурсов
- сокращение времени обслуживания запросов
- горизонтальное масштабирования

Зачем есть «балансировщик сетевой нагрузки» ?



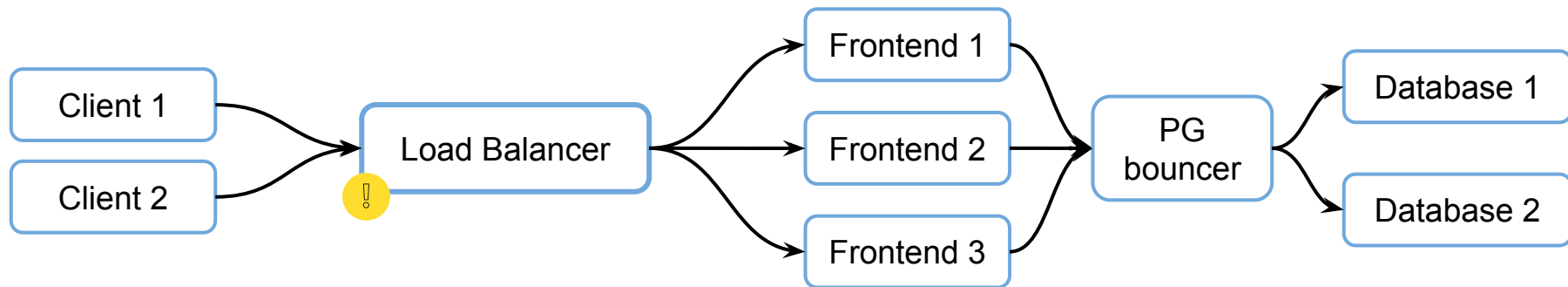
- организация отказоустойчивости
- оптимизация вычислительных ресурсов
- сокращение времени обслуживания запросов
- горизонтальное масштабирования

Зачем есть «балансировщик сетевой нагрузки» ?

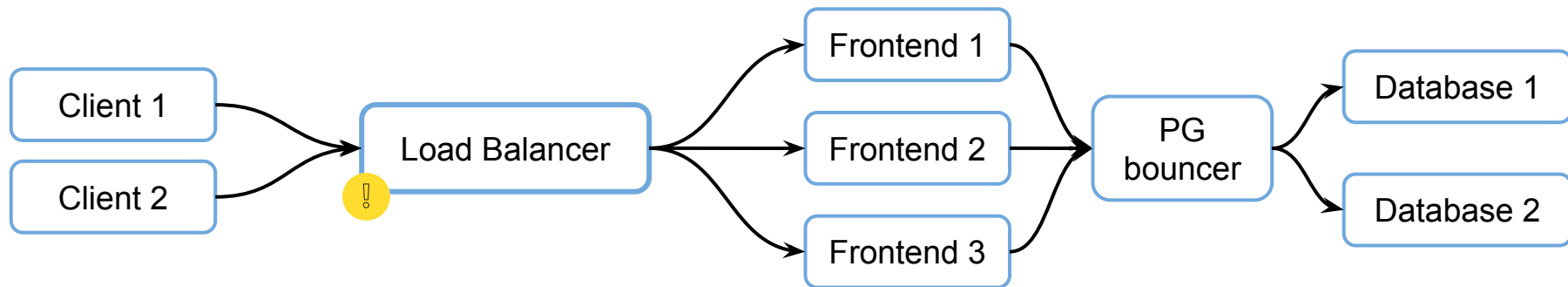


- проверка доступности приложения
- динамическое добавление новых сервисов
- API для управления трафиком (Blue-Green деплой и т.п.)
- кеширование
- ACL

Как есть «балансировщик сетевой нагрузки» ?

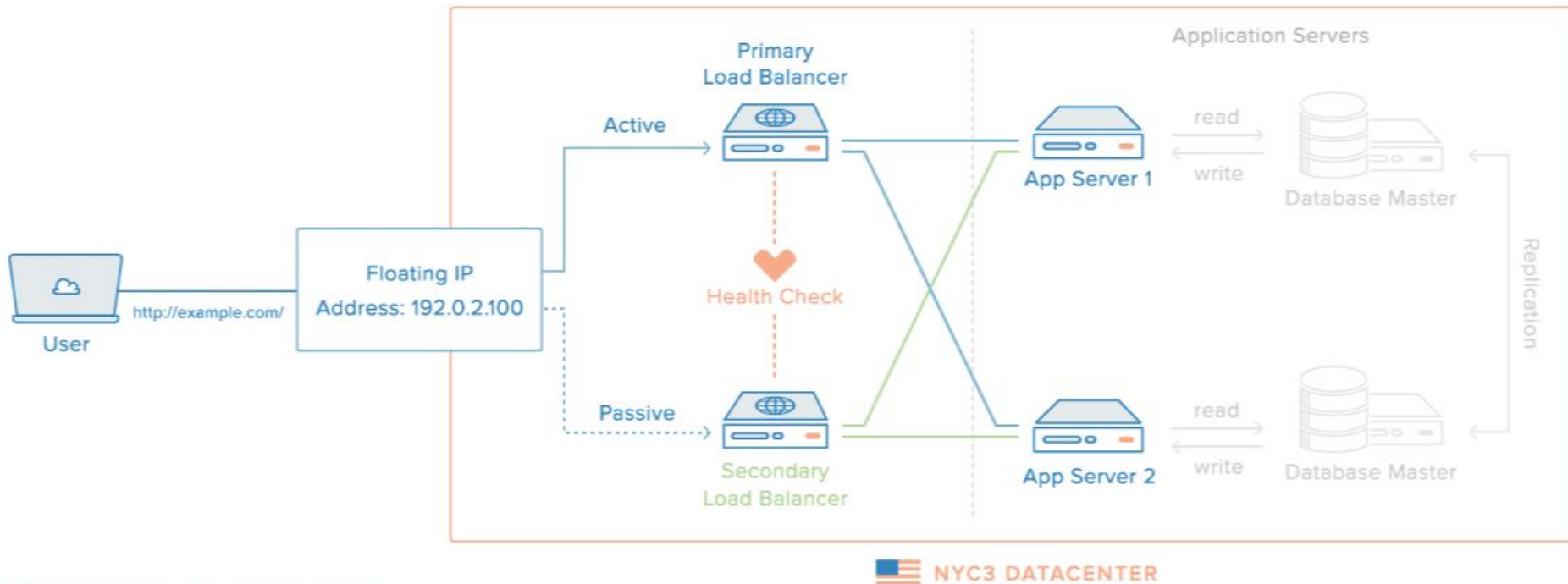


Как есть «балансировщик сетевой нагрузки» ?



- DNS round-robin
- VRRP
- BGP
- Keepalived
- LVS

Как есть «балансировщик сетевой нагрузки» ?



- 1 Active/Passive Cluster is healthy
- 2 Primary node fails
- 3 Floating IP is assigned to Secondary node



2

Сравнение основных решений

- 1995
- Multi-Processing Modules (MPM)
 - prefork
 - worker
 - event
- Динамический и статический контент, прокси, кеширование





- 2004
- C10K
- Async, event oriented
- Статический контент, прокси, кеш
- Меньше потребление ресурсов, лёгкость





- 2001
- не умеет кешировать
- не раздаёт статику
- non-blocking event oriented
- отличный хитровывернутый балансировщик/прокси ([link](#))
- API, health checks



HAProxy



3

Смотрим в будущее

- Service Mesh
- REST API
- Circuit Breaker
- Service Discovery
- Health Checks
- CLI
- Clustering
- Cloud Native





4

NGINX

Почему Nginx



- Производительность
- Статика, кеширование
- 160+ модулей ([link](#))



- Производительность
- Статика, кеширование
- 160+ модулей ([link](#))
 - [Lua](#), [Njs](#)
 - HLS / RTMP стриминг
 - FastCGI
 - API для апстримов
 - HealthChecks

Lua example /1



```
location /hello {  
    # MIME type determined by default_type:  
    default_type 'text/plain';  
    content_by_lua_block {  
        ngx.say('Hello, world!')  
    }  
}
```

Lua example /2



```
location /nginx_var {  
    # MIME type determined by default_type:  
    default_type 'text/plain';  
  
    # try access /nginx_var?a=hello,world  
    content_by_lua_block {  
        ngx.say(ngx.var.arg_a)  
    }  
}
```




```
sudo yum install -y nginx
```

Конфигурационный файл



```
$ vim /etc/nginx/nginx.conf
# main
...
events {
    ...
}
http {
    ...
    server <1> {
        ...
        location 1 { ... }
        location N { ... }
    }
}
```

Обработка запроса: выбор сервера по имени



```
server {  
    listen      80;  
    server_name example.org www.example.org;  
    ...  
}
```

```
server {  
    listen      80;  
    server_name example.net www.example.net;  
    ...  
}
```

```
server {  
    listen      80;  
    server_name example.com www.example.com;  
    ...  
}
```

Обработка запроса: выбор сервера по имени



```
$ curl -H "Host: www.fintech.ru" 1.2.3.4
```

```
server {  
    listen      80;  
    server_name example.org www.example.org;  
    ...  
}
```

```
server {  
    listen      80;  
    server_name example.net www.example.net;  
    ...  
}
```

```
server {  
    listen      80;  
    server_name example.com www.example.com;  
    ...  
}
```

Обработка запроса: выбор сервера по имени



```
$ curl -H "Host: www.fintech.ru" 1.2.3.4
```



```
server {  
    listen      80;  
    server_name example.org www.example.org;  
    ...  
}
```

```
server {  
    listen      80;  
    server_name example.net www.example.net;  
    ...  
}
```

```
server {  
    listen      80;  
    server_name example.com www.example.com;  
    ...  
}
```

Обработка запроса: выбор сервера по имени+IP



```
server {  
    listen      192.168.1.1:80;  
    server_name example.org www.example.org;  
    ...  
}  
  
server {  
    listen      192.168.1.1:80 default_server;  
    server_name example.net www.example.net;  
    ...  
}  
  
server {  
    listen      192.168.1.2:80;  
    server_name example.com www.example.com;  
    ...  
}
```

Обработка запроса: выбор сервера по имени+IP



```
$ curl -H "Host: www.fintech.ru" 192.168.1.2
```

```
server {  
    listen      192.168.1.1:80;  
    server_name example.org www.example.org;  
    ...  
}
```

```
server {  
    listen      192.168.1.1:80 default_server;  
    server_name example.net www.example.net;  
    ...  
}
```

```
server {  
    listen      192.168.1.2:80;  
    server_name example.com www.example.com;  
    ...  
}
```

Обработка запроса: выбор сервера по имени+IP



```
$ curl -H "Host: www.fintech.ru" 192.168.1.2
```

```
server {  
    listen      192.168.1.1:80;  
    server_name example.org www.example.org;  
    ...  
}
```

```
server {  
    listen      192.168.1.1:80 default_server;  
    server_name example.net www.example.net;  
    ...  
}
```

```
server {  
    listen      192.168.1.2:80;  
    server_name example.com www.example.com;  
    ...  
}
```



Обработка запроса: выбор location



```
server {  
    listen      80;  
    server_name example.org www.example.org;  
    root        /data/www;  
  
    location / {  
        index   index.html index.php;  
    }  
  
    location ~* \.(gif|jpg|png)$ {  
        expires 30d;  
    }  
  
    location ~ \.php$ {  
        fastcgi_pass   localhost:9000;  
        fastcgi_param  SCRIPT_FILENAME  
                        $document_root$fastcgi_script_name;  
        include        fastcgi_params;  
    }  
}
```



1. `location /test { ... }`
2. `location ~ ^/(test|bar|baz$) { ... }`
3. `location @secret_place { ... }`

[офф. дока про location](#)

Обработка запроса: выбор location



```
server {  
    listen      80;  
    server_name example.org www.example.org;  
    root        /data/www;  
  
    location / {  
        index   index.html index.php;  
    }  
  
    location ~* \.(gif|jpg|png)$ {  
        expires 30d;  
    }  
  
    location ~ \.php$ {  
        fastcgi_pass   localhost:9000;  
        fastcgi_param  SCRIPT_FILENAME  
                        $document_root$fastcgi_script_name;  
        include        fastcgi_params;  
    }  
}
```

Обработка запроса: выбор location



```
GET /logo.gif
```

```
location / {  
    index    index.html index.php;  
}
```

```
location ~* \.(gif|jpg|png)$ {  
    expires 30d;  
}
```

```
location ~ \.php$ {  
    fastcgi_pass    localhost:9000;  
    fastcgi_param   SCRIPT_FILENAME  
                    $document_root$fastcgi_script_name;  
    include         fastcgi_params;  
}
```

Обработка запроса: выбор location



```
GET /index.php
```

```
location / {  
    index    index.html index.php;  
}
```

```
location ~* \.(gif|jpg|png)$ {  
    expires 30d;  
}
```

```
location ~ \.php$ {  
    fastcgi_pass    localhost:9000;  
    fastcgi_param   SCRIPT_FILENAME  
                    $document_root$fastcgi_script_name;  
    include         fastcgi_params;  
}
```

Обработка запроса: выбор location



```
GET /about.html
```

```
location / {  
    index    index.html index.php;  
}
```

```
location ~* \.(gif|jpg|png)$ {  
    expires 30d;  
}
```

```
location ~ \.php$ {  
    fastcgi_pass    localhost:9000;  
    fastcgi_param   SCRIPT_FILENAME  
                    $document_root$fastcgi_script_name;  
    include         fastcgi_params;  
}
```

Обработка запроса: выбор location



GET /

```
location / {  
    index    index.html index.php;  
}
```

```
location ~* \.(gif|jpg|png)$ {  
    expires 30d;  
}
```

```
location ~ \.php$ {  
    fastcgi_pass    localhost:9000;  
    fastcgi_param   SCRIPT_FILENAME  
                    $document_root$fastcgi_script_name;  
    include         fastcgi_params;  
}
```



1. `location /fintech`
2. `location = /fintech`
3. `location ~ ^/fintech`
4. `location ~* ^/fintech`
5. `location ^~ /fintech`

Обработка запроса: выбор location



```
GET /test
```

```
# 1
```

```
location /tests {
```

```
...
```

```
}
```

```
# 2
```

```
location ~* ^/(test|tests/) {
```

```
...
```

```
}
```

```
# 3
```

```
location = /test {
```

```
...
```

```
}
```

Обработка запроса: выбор location



GET /test

1

location /tests {

...

}

2

location ~* ^/(test|tests/) {

...

}

3

location = /test {

...

}



Обработка запроса: выбор location



```
GET /tests/with/data
```

```
# 1
```

```
location /tests {
```

```
...
```

```
}
```

```
# 2
```

```
location ~* ^/(test|tests/) {
```

```
...
```

```
}
```

```
# 3
```

```
location = /test {
```

```
...
```

```
}
```

Обработка запроса: выбор location



```
GET /tests/with/data
```

```
# 1
```

```
location /tests {
```

```
...
```

```
}
```

```
# 2
```

```
location ~* ^/(test|tests/) {
```

```
...
```

```
}
```

```
# 3
```

```
location = /test {
```

```
...
```

```
}
```



Обработка запроса: выбор location



```
GET /tests/with/data
```

```
# 1
```

```
location /tests {
```

```
...
```

```
}
```

```
# 2
```

```
location ~* ^/(test|tests/)$ {
```

```
...
```

```
}
```

```
# 3
```

```
location = /test {
```


```
...
```

```
}
```

Обработка запроса: выбор location



GET /tests/with/data



```
# 1
location /tests {
    ...
}
# 2
location ~* ^/(test|tests/)$ {
    ...
}
# 3
location = /test {
    ...
}
```

Обработка запроса: выбор location



```
GET /test/

# 1
location /tests {
    ...
}
# 2
location ~* ^/(test|tests/)$ {
    ...
}
# 3
location = /test {
    ...
}
```



404

Такой страницы нет 🙄🙄

Зато есть много других страниц об услугах банка

Обработка запроса: выбор location



```
GET /testscases
```

```
# 1
```

```
location /tests {
```

```
...
```

```
}
```

```
# 2
```

```
location ~* ^/(test|tests/) {
```

```
...
```

```
}
```

```
# 3
```

```
location = /test {
```

```
...
```

```
}
```

Обработка запроса: выбор location



GET /testscases

1

location /tests {

...

}

2

location ~* ^/(test|tests/) {

...

}

3

location = /test {

...

}





- `index index.html;`
- `try_files $uri.html $uri/ /fallback/index.html;`
- `rewrite ^/somepage(.*)$ $1 break;`
- `return 301 $scheme://new-name.com$request_uri;`
- `proxy_pass* http://localhost:9000/;`



- `proxy_pass`
- `fastcgi_pass`
- `uwsgi_pass`
- `scgi_pass`
- `memcached_pass`
- `grpc_pass`

Настройки proxy_pass



```
proxy_http_version      1.1;
proxy_set_header         Connection "";
proxy_set_header         Host $host;
proxy_set_header         X-Real-IP $remote_addr;
proxy_set_header         X-Scheme $scheme;
proxy_set_header         X-Forwarded-For $remote_addr;
```

Map



```
map $http_user_agent $is_mobile {  
    default 0;  
    include includes.d/mobile_user_agent.map;  
}
```

...

```
"~(Symbian|\bS60(Version|V\d)|\bS60\b|\((Series 60|Windows Mobile|Palm OS|Bada);  
Opera Mini|Windows CE|Opera Mobi| BREW|Brew|Mobile; .+Firefox/|iPhone  
OS|Android|MobileSafari|Windows *Phone|\(webOS/|PalmOS)" "1";
```



```
upstream backend {  
    server backend1.somesite.com;  
    server backend2.somesite.com;  
    server backend3.somesite.com;  
}  
  
server {  
    location / {  
        proxy_pass    http://backend;  
    }  
}
```

upstream: способ балансировки

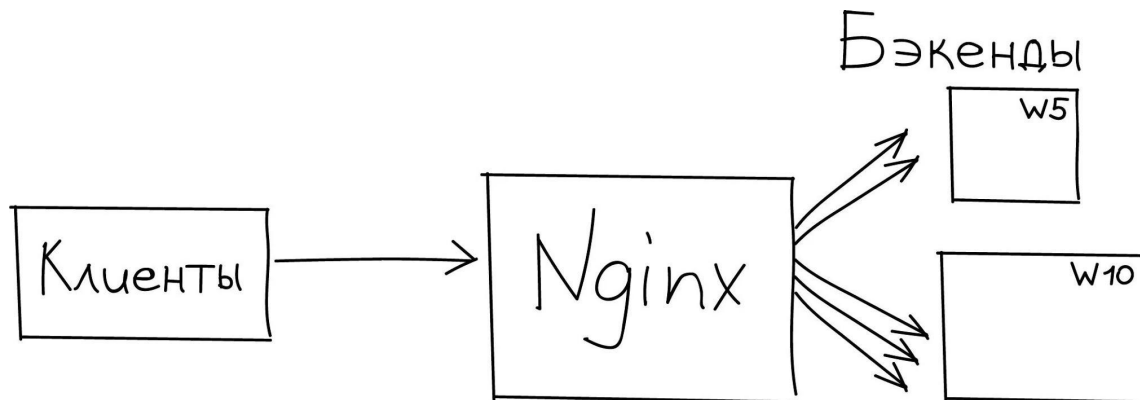


```
upstream backend {  
    server backend1.somesite.com;  
    server backend2.somesite.com;  
    server backend3.somesite.com;  
}
```

- Round-Robin
- least_conn
- hashing



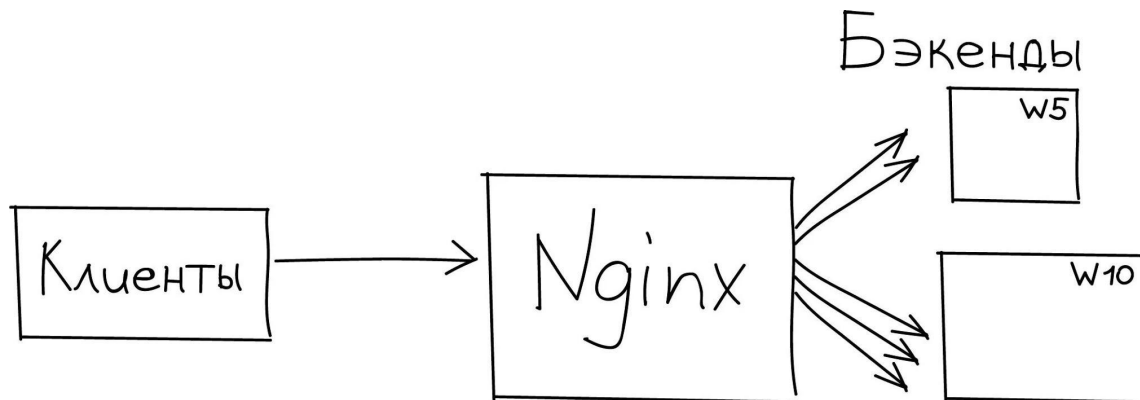
```
upstream backend {  
    server backend1.somesite.com weight=10;  
    server backend2.somesite.com weight=5;  
    server backend3.somesite.com;  
    server 192.0.0.1 backup;  
}
```



upstream: мониторинг



```
upstream backend {  
    server backend1.somesite.com weight=10 max_fails=5 fail_timeout=30s;  
    server backend2.somesite.com weight=5 max_fails=5;  
    server backend3.somesite.com;  
    server 192.0.0.1 backup;  
}
```





GET /test

```
location /test { proxy_pass ...; }
```

```
location /test/ { proxy_pass ...; }
```

```
location ~ ^/test/ { proxy_pass ...; }
```



GET /test

```
location /test { proxy_pass ...; }
```

```
location /test/ { proxy_pass ...; }
```



```
location ~ ^/test/ { proxy_pass ...; }
```



GET /test

```
location /test { proxy_pass ...; }
```

```
location /test/ { proxy_pass ...; }
```



GET /test



```
location /test { proxy_pass ...; }
```

```
location /test/ { proxy_pass ...; }
```



GET /test

```
location /te { proxy_pass ...; }
```

```
location /test/ { proxy_pass ...; }
```

```
location /testwooh { proxy_pass ...; }
```



GET /test

```
location /te { proxy_pass ...; }
```



```
location /test/ { proxy_pass ...; }
```

```
location /testwooh { proxy_pass ...; }
```


Примерное домашнее задание



- Скомпилировать nginx с vts-модулем
- Установить, настроить hello-world на localhost
- Включить vts-метрики на отдельной страничке
- Завести DNS-адреса и настроить servers
- Поднять python-приложение на двух портах, upstreams
- Мониторинг vts подключить (сбор) в Prometheus
 - опционально - отображение в Grafana
- Все изменения сохраняются после `sudo reboot`



Спасибо за внимание!

Рыбников Виталий
t.me/Frod0x



- [Руководство для начинающих](#)
 - [Как nginx обрабатывает запросы](#)
 - [Understanding the Nginx Configuration File Structure and Configuration Contexts](#)
 - [Understanding Nginx Server and Location Block Selection Algorithms](#)
 - [Балансировка бэкендов с помощью Nginx](#)
 - [Введение в современную сетевую балансировку и проксирование](#)
 - [Разница между nginx и apache с примерами](#)
 - [NGINX изнутри: рожден для производительности и масштабирования](#)
 - [Тонкая настройка балансировки нагрузки // HL2018](#)
 - <https://www.nginx.com/resources/wiki/start/topics/examples/full/>
-
- `sudo nginx -s reload`

Безопасность и грабли Nginx



- `add_headers` vs `more_set_headers`
- Strict-Transport-Security ([link](#))
- `proxy_intercept_errors` `off;`
- allow/deny examples, особенности
- X-Frame-Options "SAMEORIGIN";
- Security tips
- <http://agentzh.blogspot.com/2011/03/how-nginx-location-if-works.html>