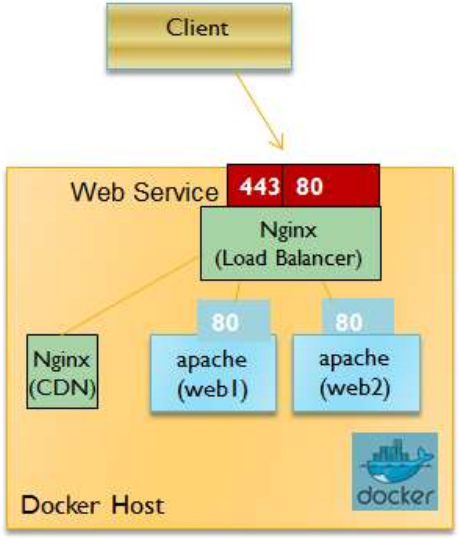
Learn Nginx in 90 minutes

Larry cai <larry.caiyu@gmail.com>

Agenda

- Nginx Introduction
- Exercise 1: Nginx env and first web
- Exercise 2: Proxy to another Apache
- Exercise 3: Load balancer to multi machines with CDN
- Exercise 4: HTTP basic authentication
- Exercise 5: HTTPS + basic auth
- Reference



Code:
https://github.com/larrycai/codingwithme-nginx

Environment (docker)

- Boot2docker 1.3.x /recommend http://boot2docker.io/
  - Add proxy /var/lib/boot2docker/profile if needed
    - `$ sudo vi /var/lib/boot2docker/profile`
    - `export http_proxy=<your proxy>`
    - `$ sudo /etc/init.d/docker restart`
  - `$ docker -v`
  - User/Passwd: docker/tcuser

- Or create CoreOS VM, and assign public IP to access

http://ustack.com or https://cloud.digitalocean.com

UnitedStack

Preparation

- Clone code & Start them

```
$ git clone https://github.com/larrycai/codingwithme-nginx.git
$ cd codingwithme-nginx
$ bash start.sh
```

```
docker@boot2docker:~$ git clone https://github.com/larrycai/codingwithme-nginx.git
Cloning into 'codingwithme-nginx'...
remote: Counting objects: 29, done.
remote: Compressing objects: 100% (22/22), done.
remote: Total 29 (delta 6), reused 27 (delta 4)
Unpacking objects: 100% (29/29), done.
docker@boot2docker:~$ cd codingwithme-nginx/
docker@boot2docker:~/codingwithme-nginx$ ./start.sh
Prepare for the nginx env: larrycai/nginx-demo & httpd server
run 'stop.sh' if they already started before
You can use 'docker exec -it nginx bash' enter into bash (inside docker container)
Also always check logs by 'docker logs -f nginx'

c012c05db3390963e23437bc2607c4a6627b66e1570cba896b6cd794c958807d
0902d915e44be21def97df9271fada46f4f5b667c5d2b48907f827d5b11132ad
e5960113307e092fff107b709ab5d14dc07de01a5a4155b53c9c66ff29c6e4a7
32a48e3256d3a89420f68f4dad598d5bd76a6560faae67f12dd9d2c465933914
docker@boot2docker:~/codingwithme-nginx$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
PORTS                                   NAMES
32a48e3256d3        larrycai/nginx-demo:latest   "/bin/sh -c /start.s   3 seconds ago       Up 1 seconds
0.0.0.0:443->443/tcp, 0.0.0.0:8080->80/tcp   nginx
e5960113307e        httpd:2.4                "httpd -DFOREGROUND"   3 seconds ago       Up 2 seconds
80/tcp                                  web2
0902d915e44b        httpd:2.4                "httpd -DFOREGROUND"   4 seconds ago       Up 2 seconds
80/tcp                                  web1
c012c05db339        larrycai/nginx-demo:latest   "/bin/sh -c /start.s   4 seconds ago       Up 3 seconds
443/tcp, 80/tcp                         cdn
docker@boot2docker:~/codingwithme-nginx$
```

In Boot2docker 1.3.x Windows, git clone in windows (C:\Users\<signum>), it will be shared in boot2docker /c/Users/<signum> . Windows editor can be used for exercise

What is Nginx

- Nginx [engine x] is an HTTP and reverse proxy server, as well as a mail proxy server
- Stable release 1.6.2 (2014.12.01)
- High performance and efficiency on I/O
  - Nginx: event-driven and asynchronous
  - Apache: processes and threads

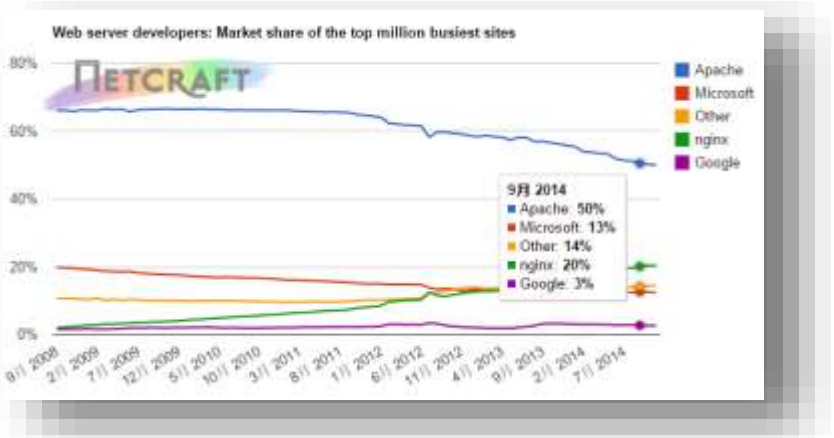


image source
http://news.netcraft.com/archives/2014/11/19/november-2014-web-server-survey.html

Working Nginx

- Master: handles loading configuration and launching or upgrading workers
- Worker: handle a specific incoming request
- $ nginx -s start|reload

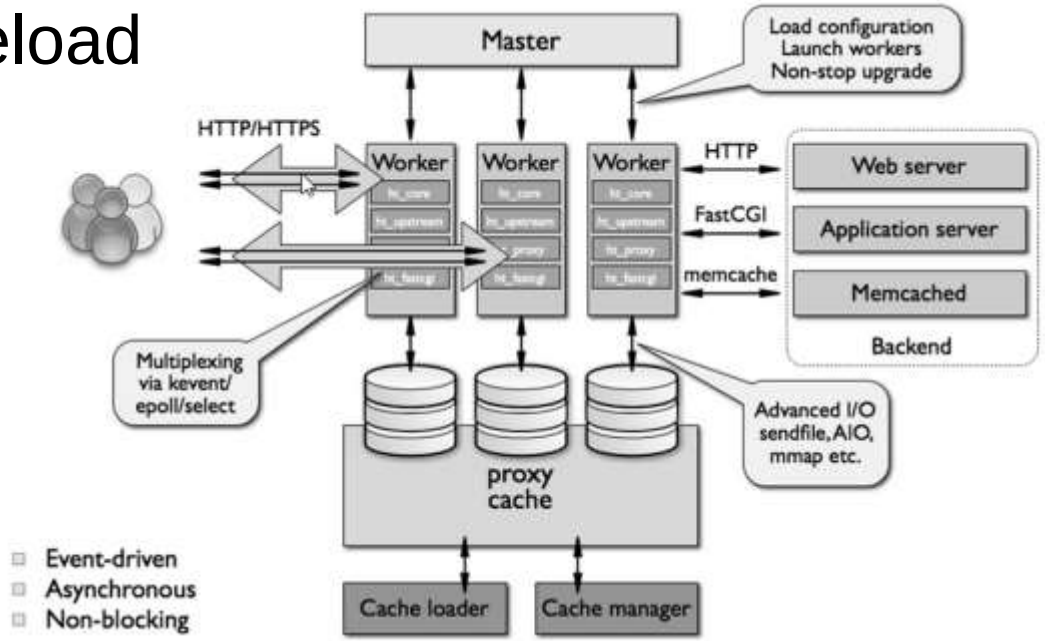
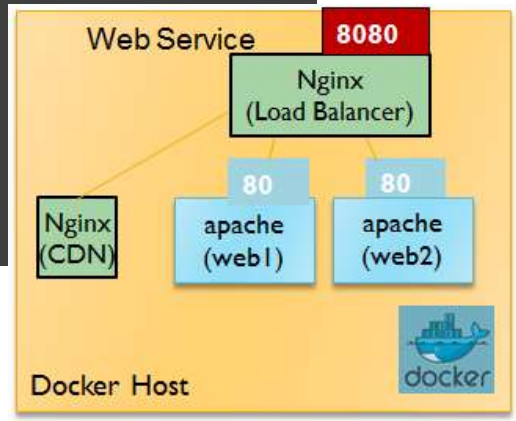


Image source: https://anturis.com/blog/nginx-vs-apache/

Environment

- All the servers are started as docker container in advance

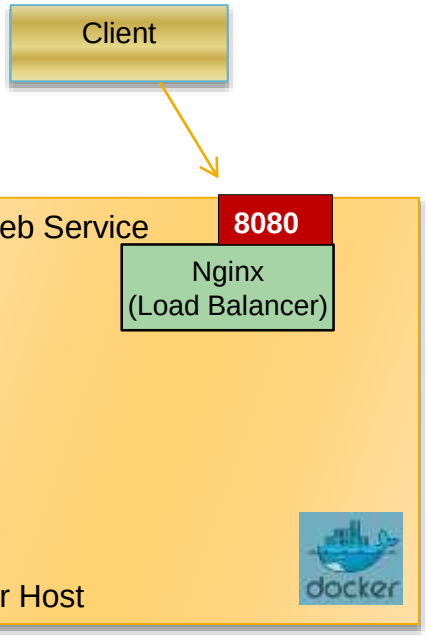
```
docker run -d --name cdn larrycai/nginx-demo
docker run -d --name web1 httpd:2.4
docker run -d --name web2 httpd:2.4
docker run -d --name nginx -p 8080:80 -p 443:443 \
            --link cdn:cdn \
            --link web1:web1 \
            --link web2:web2 \
            --volume `pwd`:/nginx \
            larrycai/nginx-demo
```

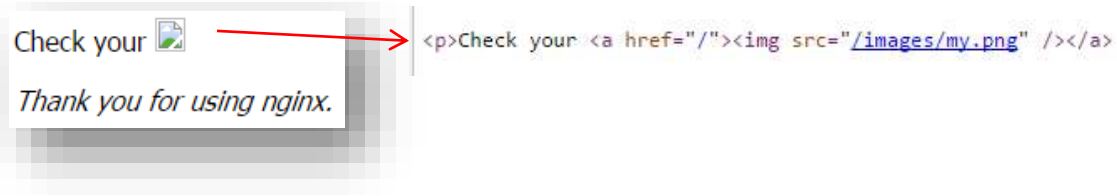


- or
- `./start-without-exec.sh` for docker <1.3 like coreos earlier version

Exercise 1: first web page

- Check the default web page
  - curl/browser (http://192.168.59.103:8080)
- Running into nginx and check the process
  - `$ docker logs -f nginx`
  - `$ docker exec -it nginx bash`
  - `$ ps -ef`
  - `$ cd /nginx # same as folder codingwithme-nginx`
- Add missing image, and reload it
  - `$ vi exer1.conf`
  - `$ nginx -s reload`





Nginx installation

- Build nginx from base

```
FROM ubuntu:trusty
ENV DEBIAN_FRONTEND noninteractive

# nginx
RUN apt-get update -q \
    && apt-get install -yf build-essential python-software-properties software-properties-common \
    && add-apt-repository ppa:nginx/stable \
    && apt-get update -q \
    && apt-get -y install -y curl

# build nginx from source with http auth module enabled
RUN apt-get -y install libpcre3-dev zlib1g-dev libssl-dev

RUN curl -s http://nginx.org/download/nginx-1.6.2.tar.gz | tar -xz -C /tmp \
        && cd /tmp/nginx-1.6.2 \
        && ./configure --with-http_ssl_module --with-http_auth_request_module && make && make install
```

- Or use official nginx docker image

```
$ docker pull nginx:1.6.2
```

Nginx configuration

- Directives `/usr/local/nginx/conf/nginx.conf`
- **http** – The main scope, typically configs set will reflect everywhere
- **server** – run multiple servers virtually on different ports or with different server names
- **location** – defines the scope for a URI
- **upstream** – config scope for a set of upstream/backend servers

```
user www-data;
worker_processes  1;
pid /var/run/nginx.pid;

events {
  worker_connections 1024;
}

http {
  # Basic Settings
  keepalive_timeout 65;
  types_hash_max_size 2048;
  include /usr/local/nginx/conf/mime.types;
  default_type application/octet-stream;

  # Logging Settings
  access_log /dev/stdout;
  error_log /dev/stdout;

  # Virtual Host Configs
  include /nginx/*.conf;
}

daemon off;
```

```
server {
    listen 80;
    location / {
        root /nginx/;
    }

    location /images/ {
        root /nginx/data;
    }
}
```

Exercise 2: proxy to web server

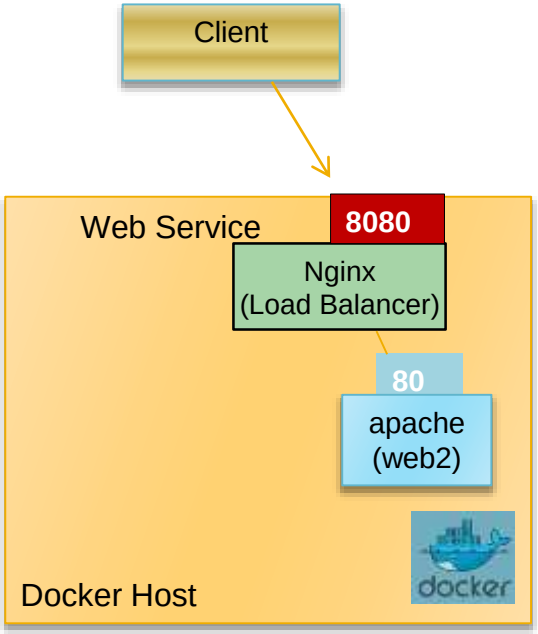
- Add one backend apache (already started as `web2`)

```
curl web2
```

- Example (`exer1.conf`):

```
location /name/ {
     proxy_pass http://web2/;
}
```

- Check the real path
- http://192.168.59.103:8080/name/



Nginx Load balancer

- Method supported in nginx:
  - round-robin — requests to the application servers are distributed in a round-robin fashion,
  - least-connected — next request is assigned to the server with the least number of active connections,
  - ip-hash — a hash-function is used to determine what server should be selected for the next request (based on the client's IP address).
- Sample

http://nginx.org/en/docs/http/load_balancing.html

```
http {
    upstream myapp1 {
        server srv1.example.com;
        server srv2.example.com;
        server srv3.example.com;
    }

    server {
        listen 80;

        location / {
            proxy_pass http://myapp1;
        }
    }
}
```

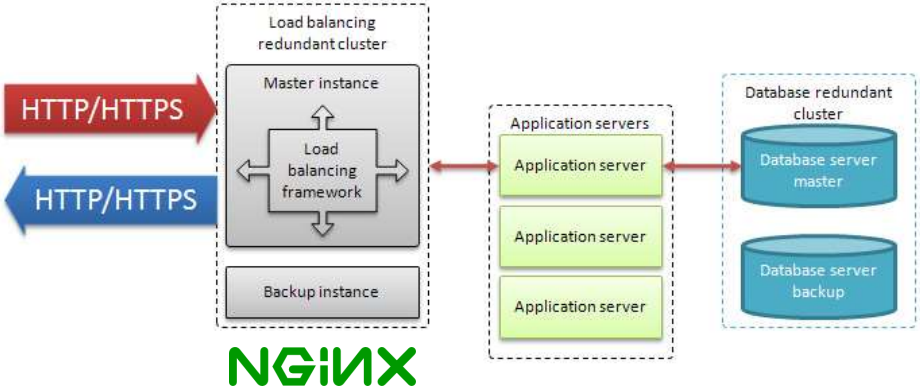


Image source: http://clusterize.net/3-load-balanced-java-web-site/

Exercise 3: Load balancer to multi machines with CDN

```
http {
    upstream myapp1 {
        server srv1.example.com;
        server srv2.example.com;
        server srv3.example.com;
    }

    server {
        listen 80;

        location / {
            proxy_pass http://myapp1;
        }
    }
}
```

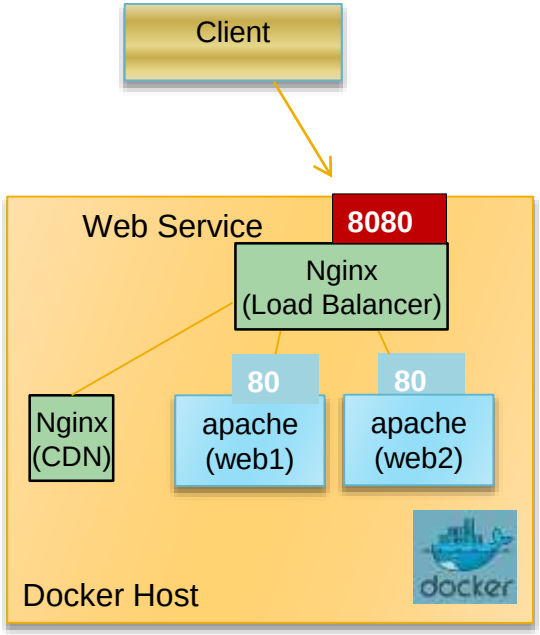
- Load balancer
  - Images are send to CDN server (/image
  - All other requests goes to web server
- Check three methods

```
upstream myapp1 {
    # ip_hash|least_conn|fair;
    server web1;
    …
```

- Hints:

  http://192.168.59.103:8080
  http://192.168.59.103:8080/images/my.png
  `curl/docker logs -f web1/web2/cdn/nginx`

Check docs
http://nginx.org/en/docs/http/load_balancing.html



Nginx basic auth

- `ngx_http_auth_basic_module` module supports “HTTP Basic Authentication” protocol
- `htpasswd` is the tool to generate basic passwd (part of apache2-util)

```
testuser:$apr1$ocTQYcLD$BnXIF02GPcivTjrFQHXXg.
larrycai:$apr1$xnWcsIpg$rorRDwNAB81VuuLiZspYK0
```

```
server {
    listen 80;
    server_name domain.com;
    root /site/root;
    index index.html index.htm;
    auth_basic            "Restricted";
    auth_basic_user_file  /etc/nginx/htpasswd;
}
```

Exercise 4: HTTP basic auth

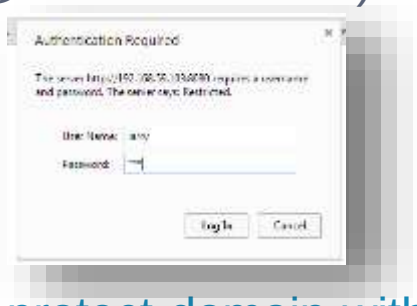
- Generate `web.htpasswd` file under /nginx

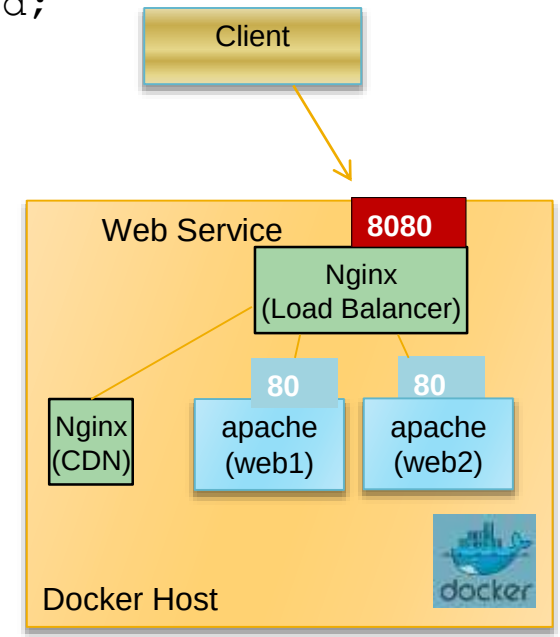
```
# htpasswd -c web.htpasswd larry # (in nginx container)
```

- Update `exer1.conf` to have access control (larry/cai)

```
server {
    auth_basic "Restricted";
    auth_basic_user_file /nginx/web.htpasswd;
```

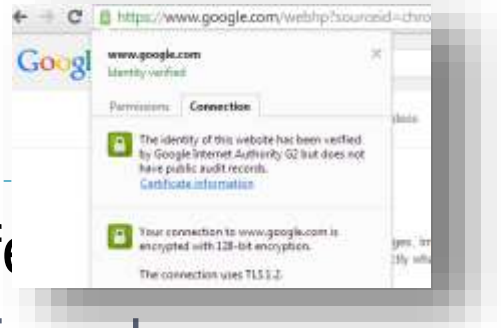
- Three cases
  - No login/passwd (401)
  - Wrong login/passwd
  - Successful (curl larry:cai@localhost/name/)





http://magnatecha.com/password-protect-domain-with-nginx/

HTTPS + basic auth



- HTTPS is secure HTTP to protect transfe
  - It needs certification signed by CA or self-signed
- Signed process
  - Private-key
  - CSR: Certificate Signing Request from Private key
  - CRT: Generate a self-signed Certificate / or Send to CA
- HTTPS can be handled/terminated by nginx
  - `ssl` parameter shall be used in nginx

```
server {
    listen              443 ssl;
    server_name         www.example.com;
    ssl_certificate     www.example.com.crt;
    ssl_certificate_key www.example.com.key;
    ssl_protocols       SSLv3 TLSv1 TLSv1.1 TLSv1.2;
    ssl_ciphers         HIGH:!aNULL:!MD5;
    ...
}
```

Exercise 5: HTTPS + basic auth

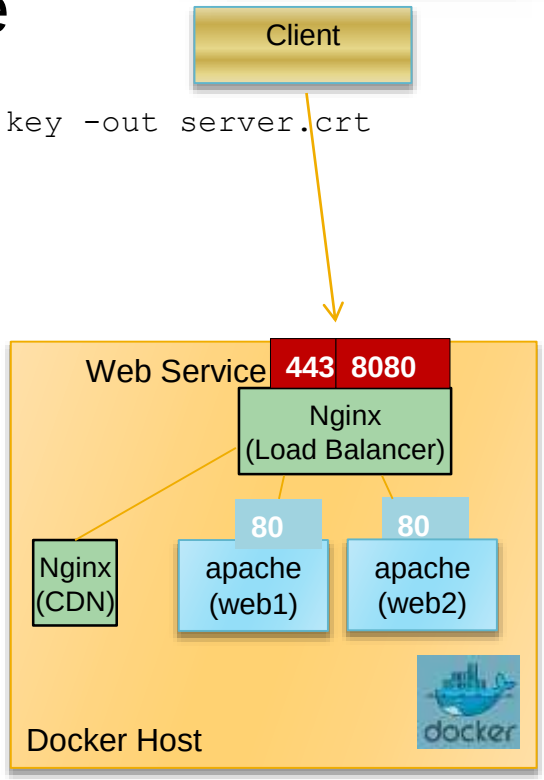
- Check normal https certification https://www.docker.com/ , who signed
- Generate self-signed key for website

```
# openssl req -new -nodes -keyout server.key -out server.csr
# openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
```

- Config in nginx.conf

```
# add extra server
server {
    listen 443 ssl;
    ssl_certificate /nginx/server.crt;
    ssl_certificate_key /nginx/server.key;
```

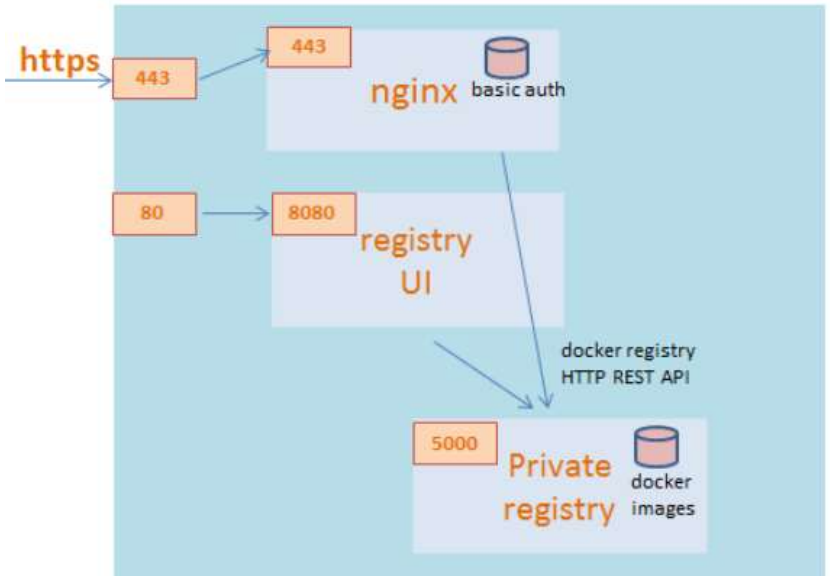
- Verify http/https: curl -k https://
- Verify https+basic auth



Bonus

- Learn nginx used for docker private registry authentication https://github.com/larrycai/nginx-registry
- Authentication with LDAP
- Log on demand https://github.com/tobegit3hub/nginx-log-service

```
server {

  listen 80;
  server_name dev.com;
  root /var/www/nginx/;
  index index.php index.html index.htm;

  location / {
    try_files $uri $uri/ /index.html;
  }

  location /log/bracelet {
    alias /var/www/nginx/log/bracelet/;

    allow 10.237.113.19;
    allow 10.237.204.14;
    deny all;

    auth_basic "Login";
    auth_basic_user_file /var/www/dev.com/admin/.htpasswd;

    autoindex on;
    autoindex_localtime on;

    expires max;
  }

}
```



Summary

- Nginx is HTTP server, which widely used in internet company for high performance.
- Large community with active module development

- Learn Nginx plugin to extend your functions
- Tune configuration to have best performance
- Use in your own case/product

Reference

- http://nginx.org official web site
- http://tengine.taobao.org/ taobao adapted engine
- Docker images:
  - https://registry.hub.docker.com/_/nginx/
- Articles:
  - https://anturis.com/blog/nginx-vs-apache/
  - https://www.linode.com/docs/websites/nginx/basic-nginx-configuration