GDAŃSK UNIVERSITY OF TECHNOLOGY

Architektury Usług Internetowych (Inf) Internet Services Architectures (ID)

Deployment introduction

2022

Tomasz Boiński
tobo@eti.pg.edu.pl

Why are we talking about this?

1. How to share code?
2. How to maintain common code standards?
3. How to control code quality?
4. Can we compile our code?
5. Does it still work as intended?
6. Who should prepare the software for deployment? And how?
7. How the run-time environment impacts the code?
8. How to mirror the target environment with limited resources?
9. …

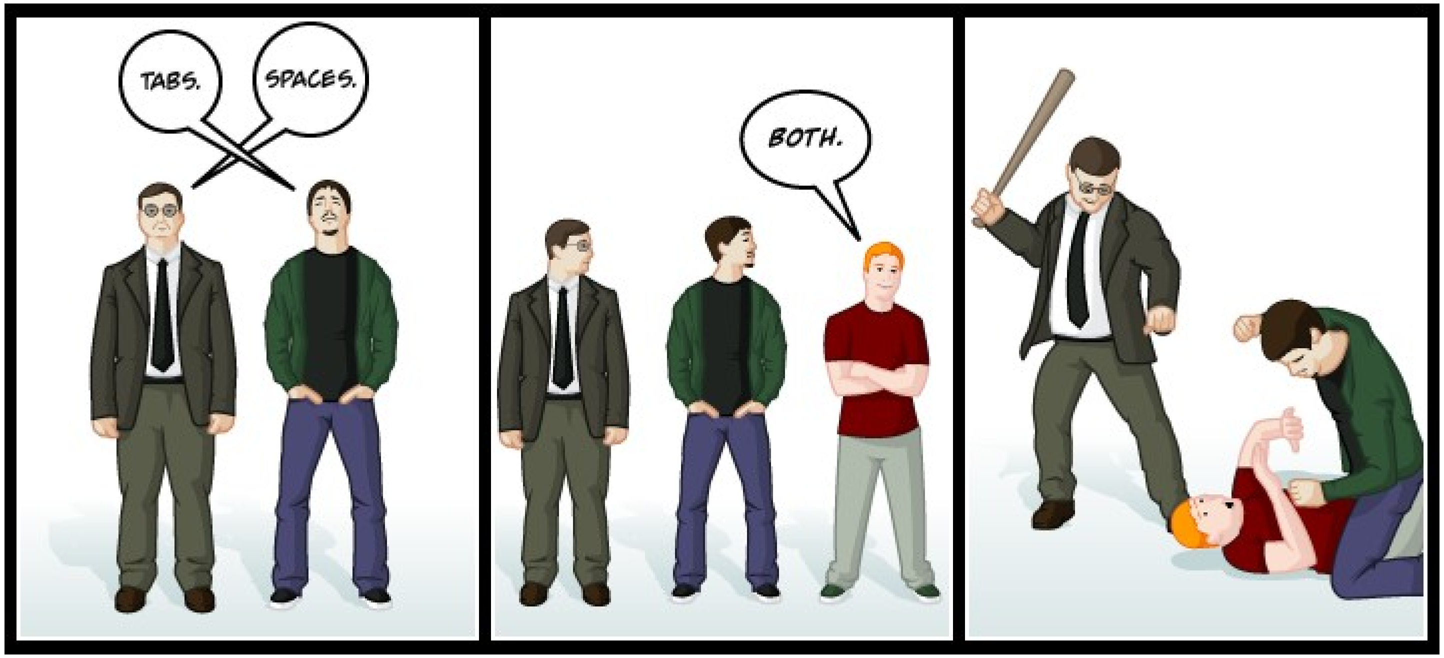
Why are we talking about this? (2)

1. If the code works in IDE – grate! But…
2. In real life it can heavily depend on the deployment architecture.
3. There can be multiple load balancers, proxy serves, URL changes, schema changes
4. The deployment software used can require certain steps to be taken in to account (e.g. in docker/doecker swarm/kubernetes)
5. Even the method of deployment can introduce certain problems (e.g. network definition in docker swarm vs service naming)

This is not enough

POLITECHNIKA
GDAŃSKA

Do not introduce chaos



Build automation

Rule of thumb:

> Everybody should be able to take a bare machine[1], checkout the code from a repository, run a single command and have a working[2] system on his or her machine

1) minus situations where this is troublesome like bug binary files in repositories – in such cases we can have well documented process how to prepare the environment or get the code.

2) compiled/started up/ready for deployment/...

POLITECHNIKA GDAŃSKA

Integrational machines

1) We have a common machine (in any form, physical, virtual, containerized etc.) mirroring the deployment environment that should be used after each code changes to build and test the app.
   1. We minimize the influence of each programmer individual environment.
   2. We minimize the lack of discipline among programmers.
2) You want to have an environment as similar as possible to this machine.

GDAŃSK UNIVERSITY OF TECHNOLOGY

docker

Based on example

Containers

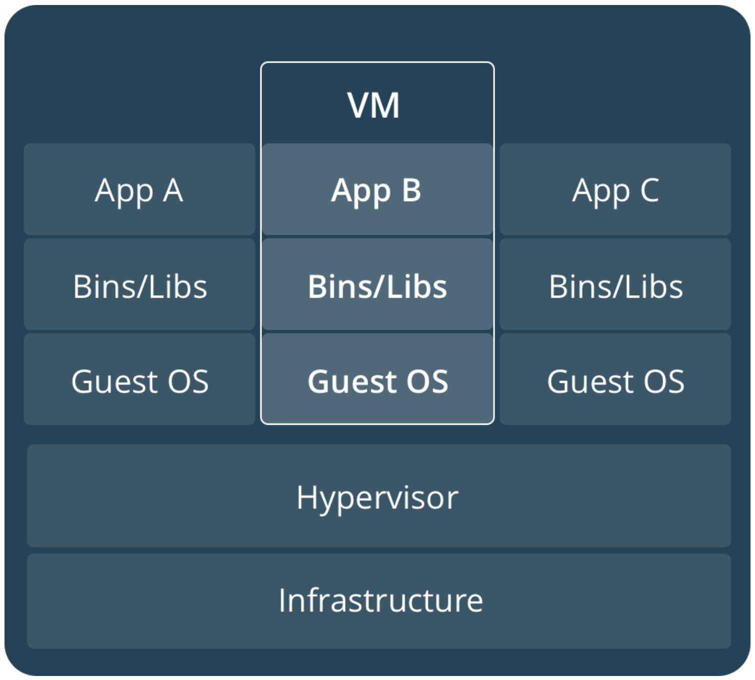
Virtual machine:

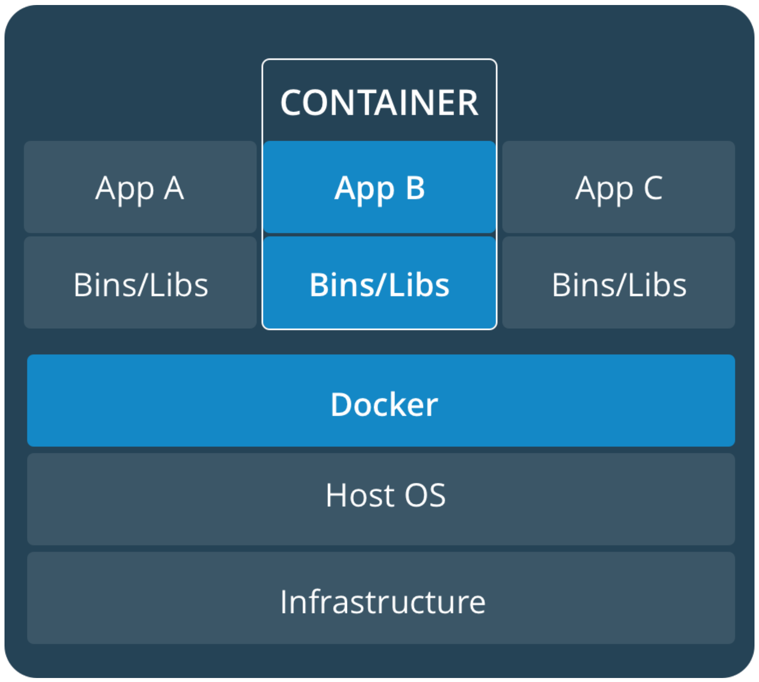
- Separate system
- Hi level hardware separation (Intel VT-x, AMD-V)
- Big size
- Performance problems
- Can take some time to start

Container:

- Kernel shared with the host
- No hardware isolation
- Small sizes (e.g. Ubuntu – 150MB)
- Containers based on the same image share its content ($n \cdot$ Ubuntu = 150MB, $n \in \mathbb{N}_+$)
- Very small performance overhead (domain dependent)
- Can run within minutes

Containers


VM
App A
App B
App C
Bins/Libs
Bins/Libs
Bins/Libs
Guest OS
Guest OS
Guest OS
Hypervisor
Infrastructure


CONTAINER
App A
App B
App C
Bins/Libs
Bins/Libs
Bins/Libs
Docker
Host OS
Infrastructure

Containers

Usages:

- Production environment
- Development or test environment similar to production environment
- Microservices
- App abstraction from physical infrastructure
- Solution for incompatible requirements and dependencies

GDAŃSK UNIVERSITY OF TECHNOLOGY

Docker

1. Allows us to create Images
2. We can run Containers based on created Images
3. Containers run our desired Applications
4. Allows us creating networks, so Containers can communicate

GDAŃSK UNIVERSITY
OF TECHNOLOGY

Container

1. Encapsulated environment that runs applications
2. Lightweight alternative for virtual machine
3. Doesn’t have own kernel – uses hosts kernel
4. Containers can communicate with each others via network

Container

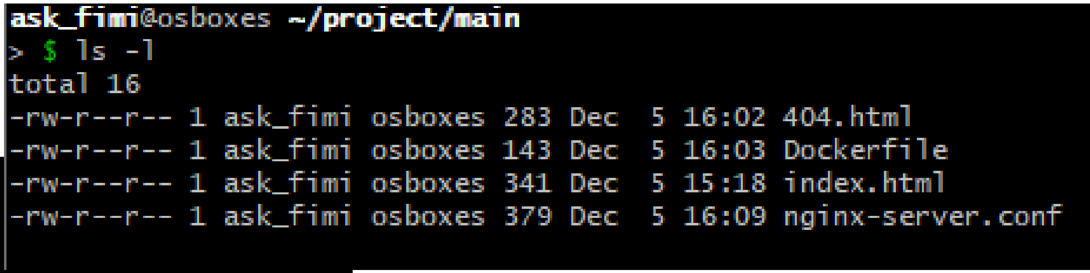
1. Has its own filesystem
2. Has its own set of ports, which can be published i.e. Running container with

   –p 8000:80

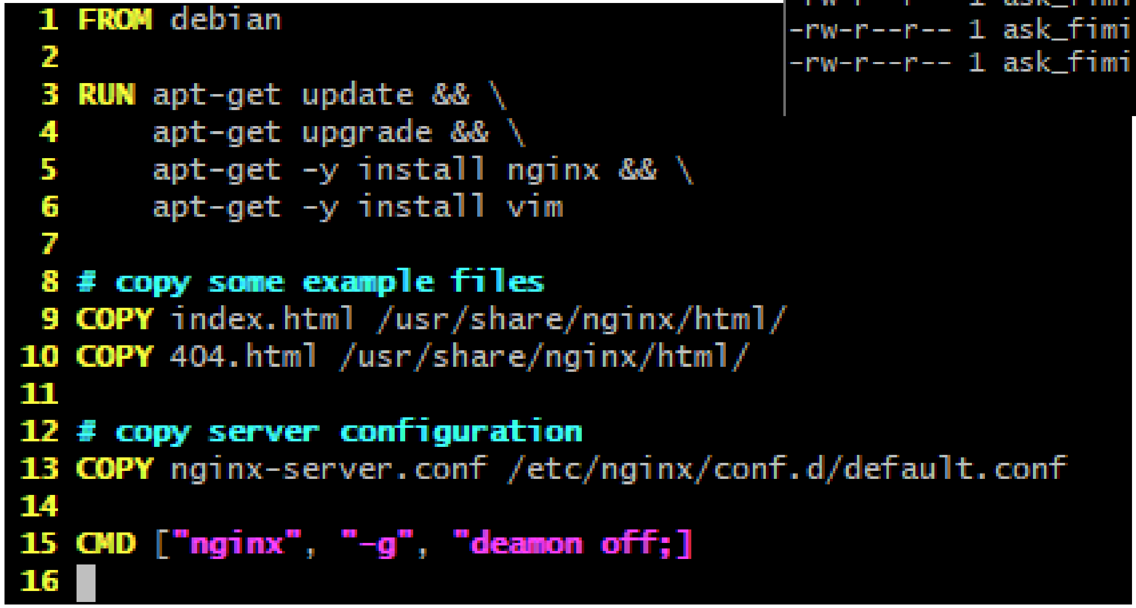
   will map port 8000 of host to port 80 of container

GDAŃSK UNIVERSITY OF TECHNOLOGY

Dockerfile – a recipe for image



```
ask_fimi@osboxes ~/project/main
> $ ls -l
total 16
-rw-r--r-- 1 ask_fimi osboxes 283 Dec  5 16:02 404.html
-rw-r--r-- 1 ask_fimi osboxes 143 Dec  5 16:03 Dockerfile
-rw-r--r-- 1 ask_fimi osboxes 341 Dec  5 15:18 index.html
-rw-r--r-- 1 ask_fimi osboxes 379 Dec  5 16:09 nginx-server.conf
```



```
FROM debian

RUN apt-get update && \
    apt-get upgrade && \
    apt-get -y install nginx && \
    apt-get -y install vim

# copy some example files
COPY index.html /usr/share/nginx/html/
COPY 404.html /usr/share/nginx/html/

# copy server configuration
COPY nginx-server.conf /etc/nginx/conf.d/default.conf

CMD ["nginx", "-g", "deamon off;]
```

GDAŃSK UNIVERSITY
OF TECHNOLOGY

Dockerfile – a recipe for image

1. FROM – allows us to specify base image
2. COPY – allows us to copy files from host filesystem to container’s filesystem
3. RUN – allows to run a command that is needed to prepare our application, for example:
4. Download needed packages
5. Clone something from remote repository
6. Compile some source if it’s needed
7. Anything you would normally do to prepare environment to run given application
8. CMD – specifies defaults for executing a container, it can include executable, if executable is not included it must be specified in ENTRYPOINT

Building image

1. builds image based on Dockerfile in current directory

   docker image build .

2. Helpful flag, that allows to tag container as name:tag useful as it’s easier to remember than auto generated hash

   -t or --tag

   docker image build . -t myImage:1

GDAŃSK UNIVERSITY
OF TECHNOLOGY

Running the container

1. Runs a container based on given image

   docker container run

2. Allows to specify a name for a container

   --name

3. Runs container in background

   -d or --detach

   docker container run -d --name myContainer myImage:1

Images

1. Images can be stored and distributed via image repositories (Docker Registry, np. Docker Hub).
2. Containers based on the images can be orchestrated into complex applications where containers connect with each other via network, for example:
   1. Container 1: Apache with some web app
   2. Container 2: server with database
   3. Container 3: proxy server, e.g. Nginx

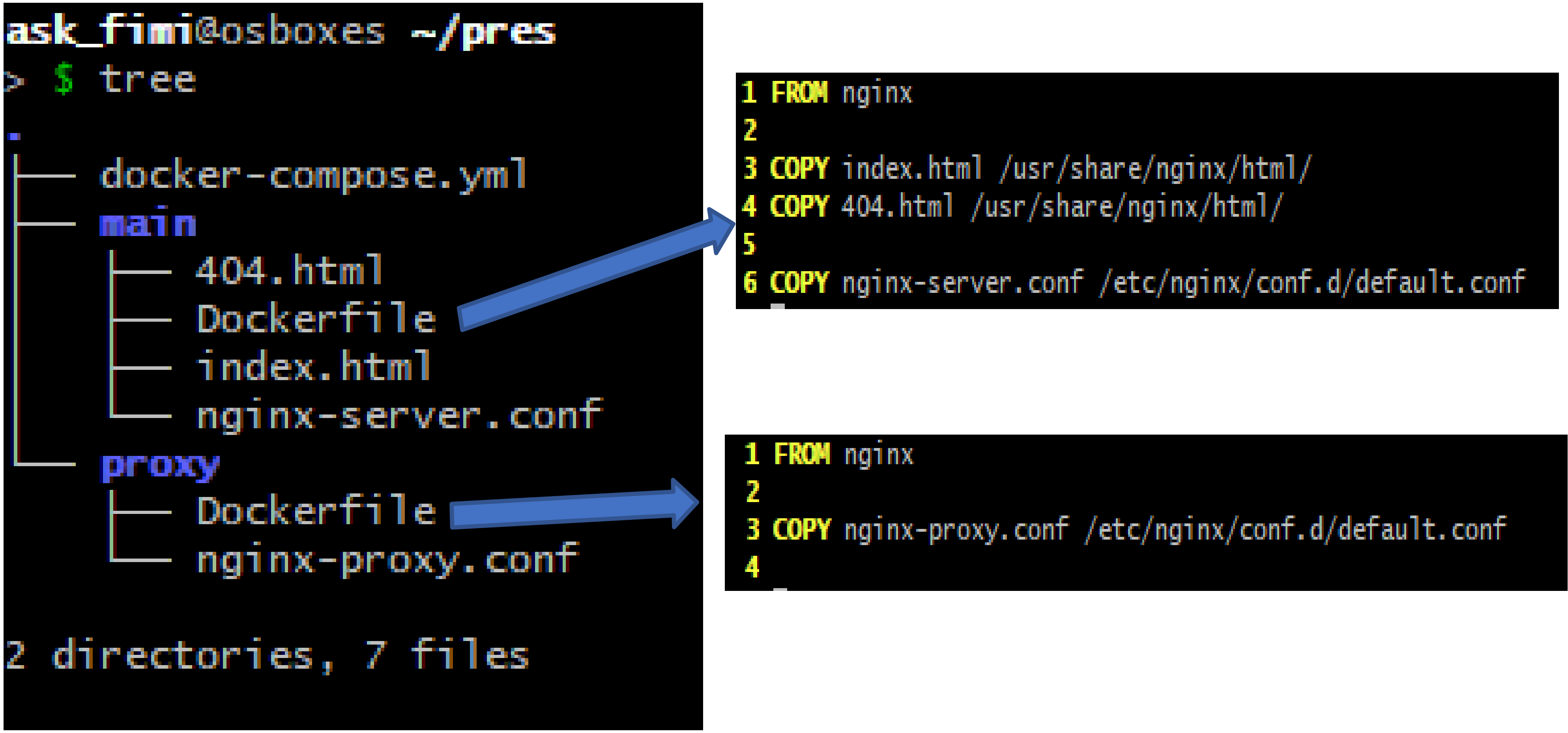
Docker-compose

1. A tool that helps running multi-container applications
2. Configuration stored in docker-compose.yml

Example

1. We want to set up two web servers in separate containers.
2. One working as reverse-proxy for second one.
3. They will be connected via internal network, only proxy server will be accessible from outside

Example – Dockerfiles



```
ask_fimi@osboxes ~/pres
> $ tree
.
├── docker-compose.yml
├── main
│   ├── 404.html
│   ├── Dockerfile
│   ├── index.html
│   └── nginx-server.conf
└── proxy
    ├── Dockerfile
    └── nginx-proxy.conf

2 directories, 7 files
```

```
FROM nginx

COPY index.html /usr/share/nginx/html/
COPY 404.html /usr/share/nginx/html/

COPY nginx-server.conf /etc/nginx/conf.d/default.conf
```

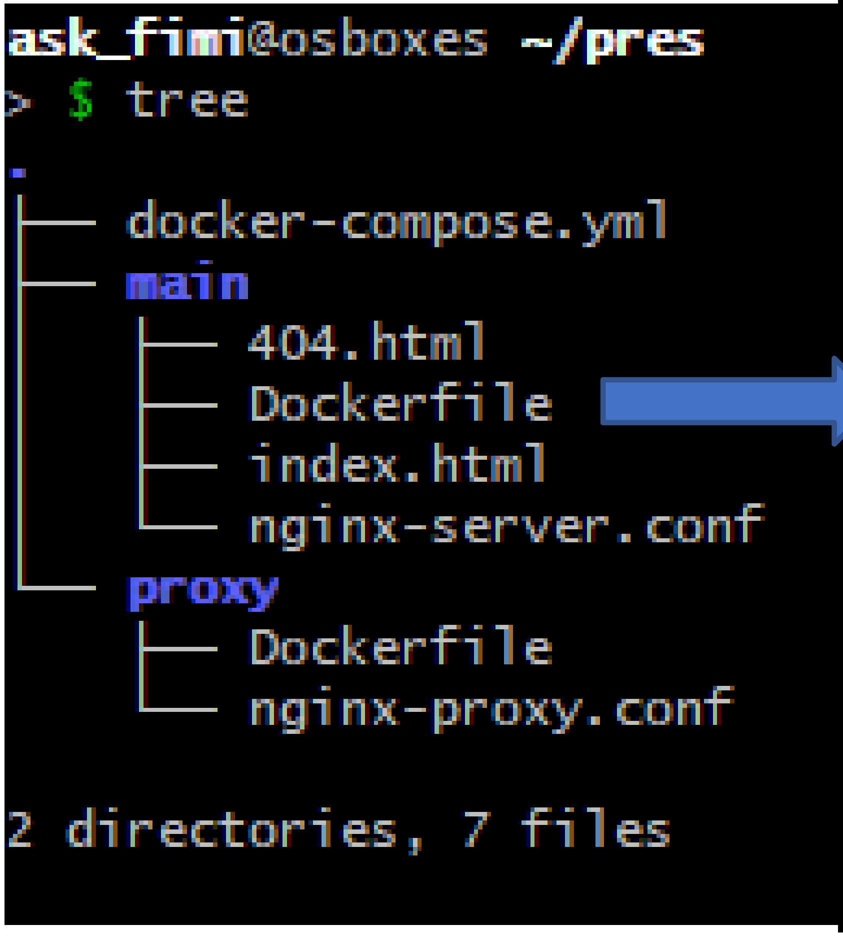
```
FROM nginx

COPY nginx-proxy.conf /etc/nginx/conf.d/default.conf

```

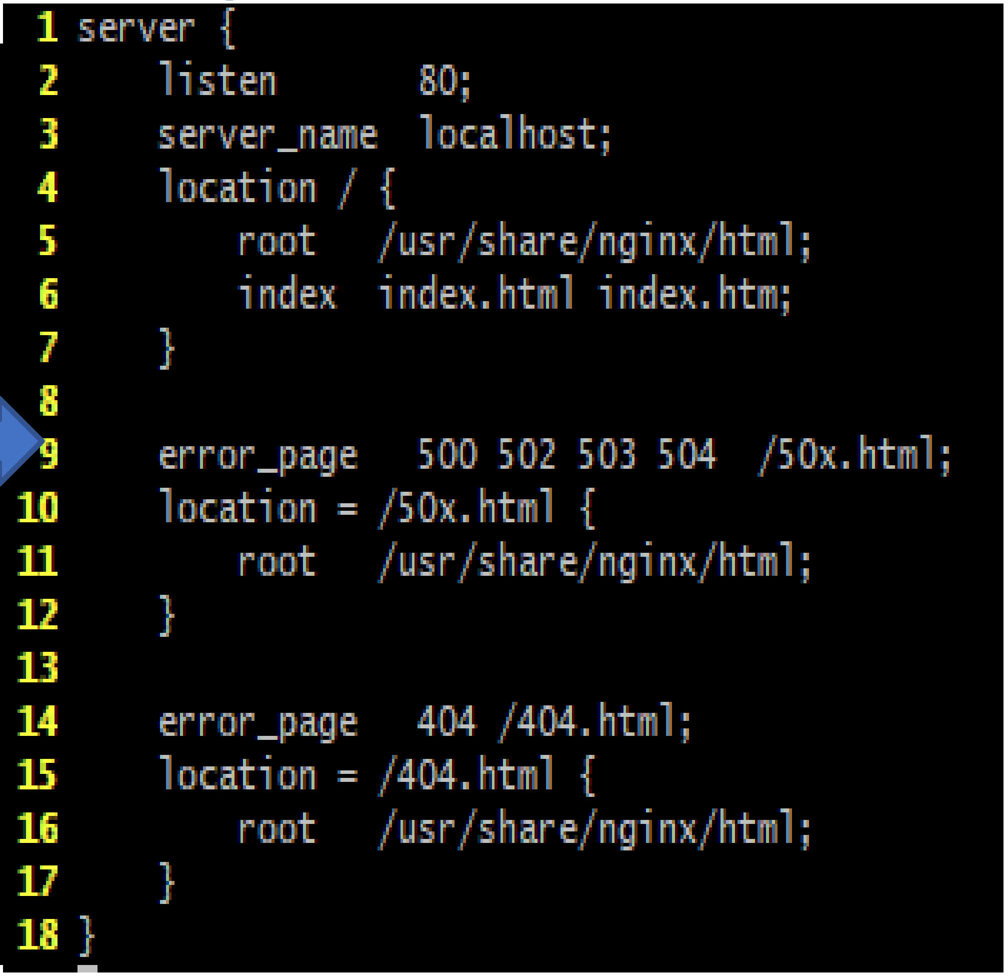
GDAŃSK UNIVERSITY
OF TECHNOLOGY

Example – server configuration



```
ask_fimi@osboxes ~/pres
> $ tree
.
├── docker-compose.yml
├── main
│   ├── 404.html
│   ├── Dockerfile
│   ├── index.html
│   └── nginx-server.conf
└── proxy
    ├── Dockerfile
    └── nginx-proxy.conf

2 directories, 7 files
```



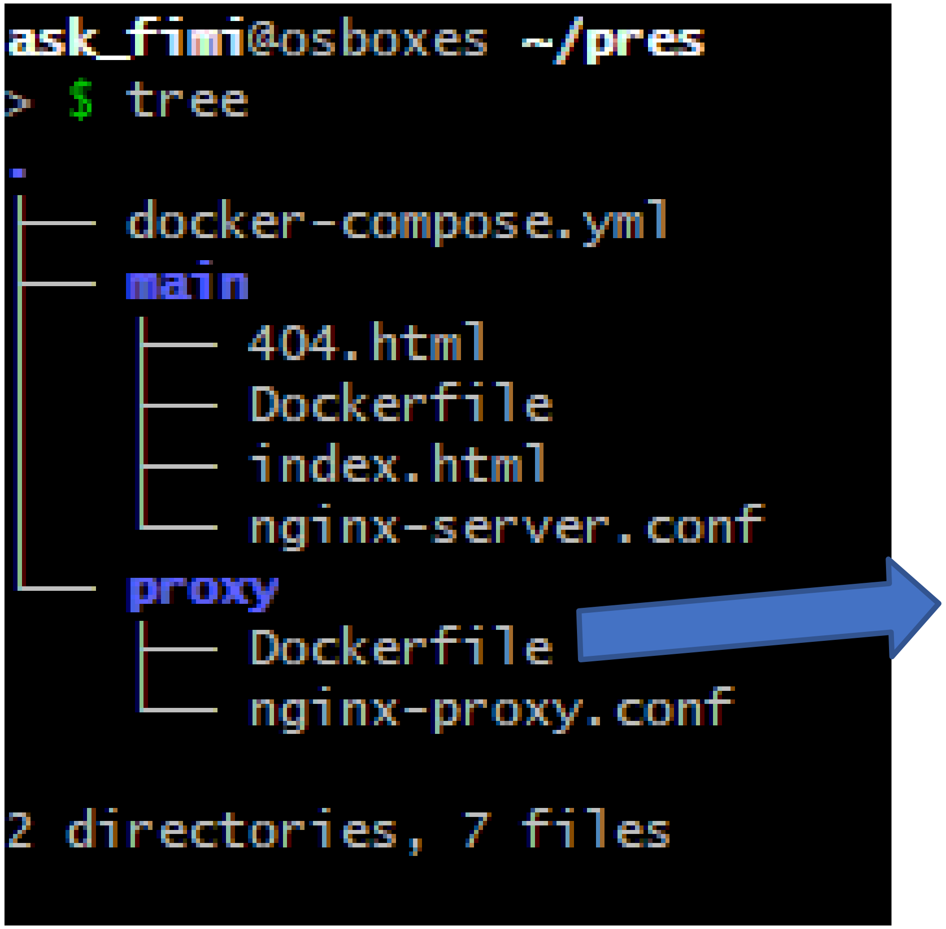
```
server {
    listen       80;
    server_name  localhost;
    location / {
        root   /usr/share/nginx/html;
        index  index.html index.htm;
    }

    error_page   500 502 503 504  /50x.html;
    location = /50x.html {
        root   /usr/share/nginx/html;
    }

    error_page   404 /404.html;
    location = /404.html {
        root   /usr/share/nginx/html;
    }
}
```

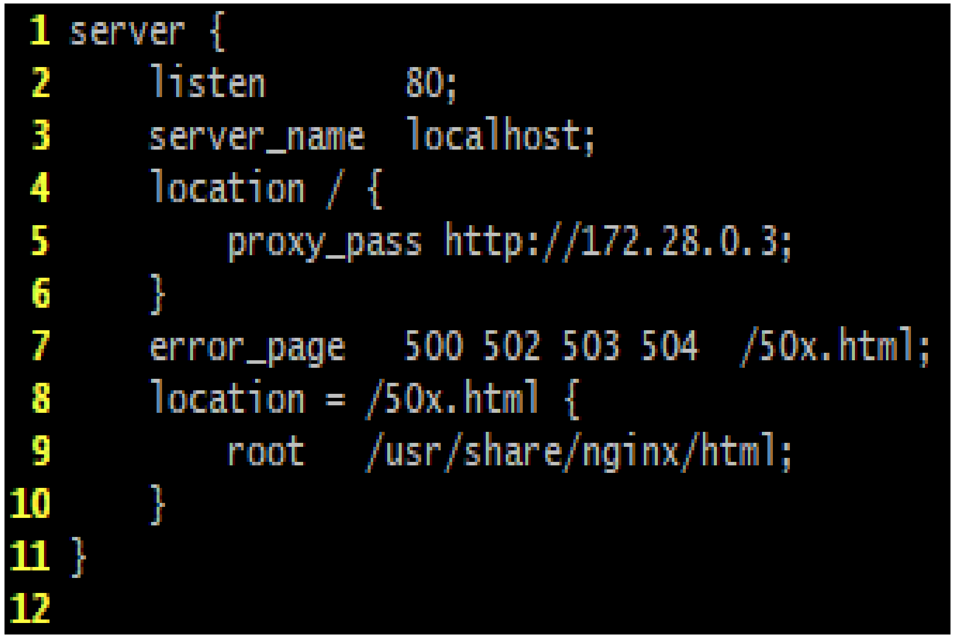
GDAŃSK UNIVERSITY
OF TECHNOLOGY

Example – proxy server configuration



```
ask_fimi@osboxes ~/pres
> $ tree
.
├── docker-compose.yml
├── main
│   ├── 404.html
│   ├── Dockerfile
│   ├── index.html
│   └── nginx-server.conf
└── proxy
    ├── Dockerfile
    └── nginx-proxy.conf

2 directories, 7 files
```

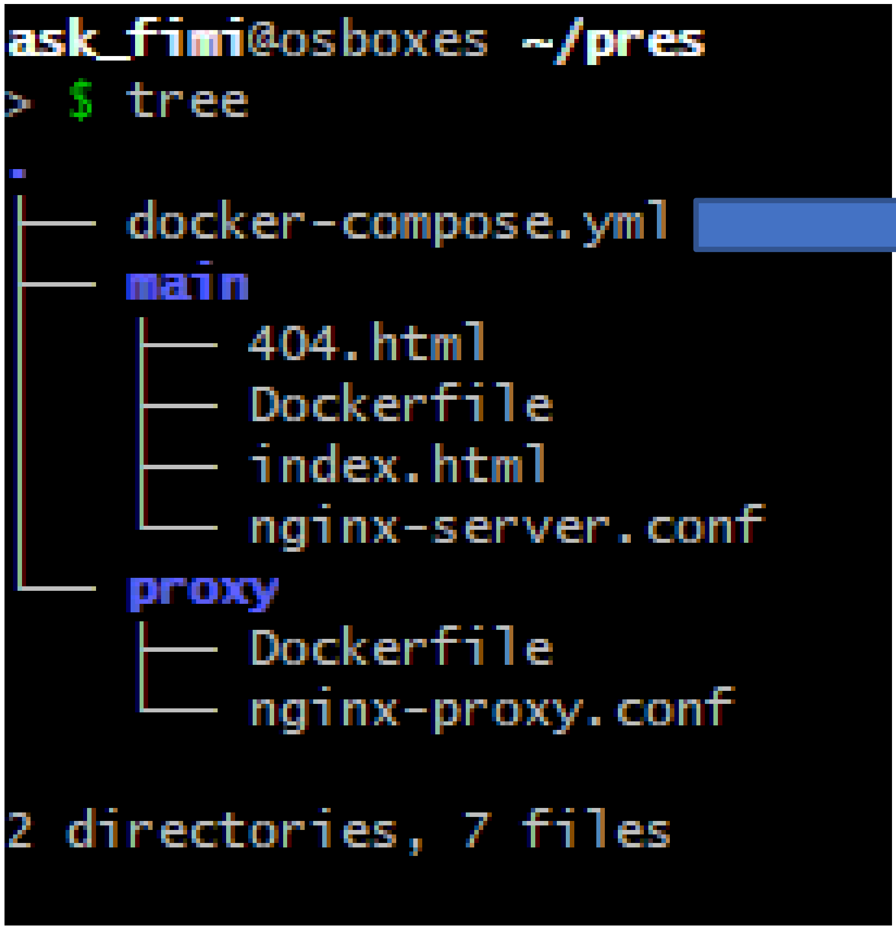


```
server {
    listen       80;
    server_name  localhost;
    location / {
        proxy_pass http://172.28.0.3;
    }
    error_page   500 502 503 504  /50x.html;
    location = /50x.html {
        root   /usr/share/nginx/html;
    }
}

```

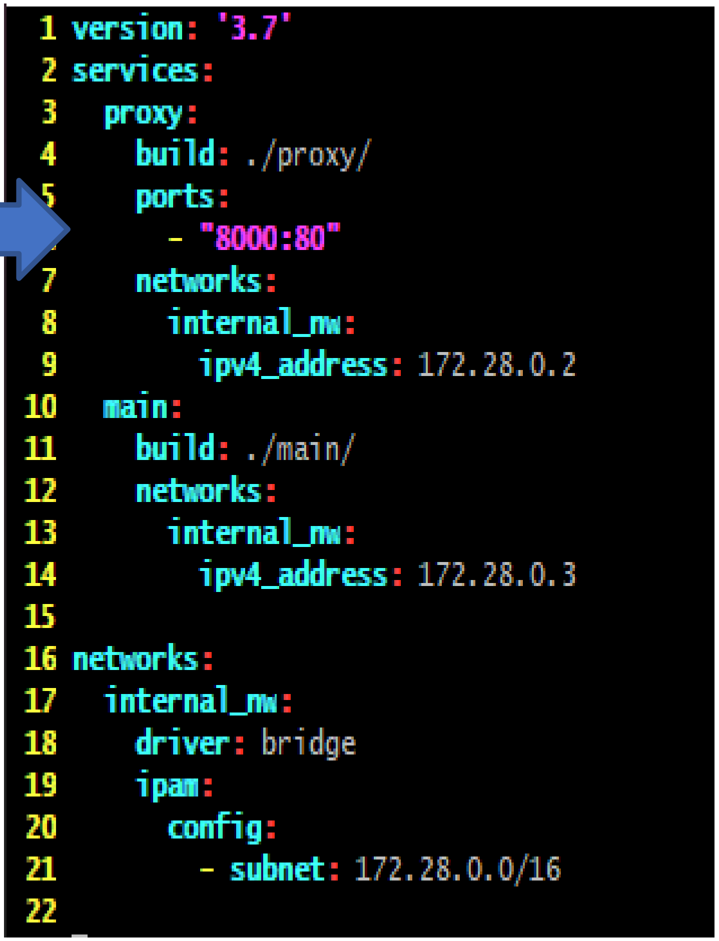
GDAŃSK UNIVERSITY
OF TECHNOLOGY

Example – docker-compose.yml



```
ask_fimi@osboxes ~/pres
> $ tree
.
├── docker-compose.yml
├── main
│   ├── 404.html
│   ├── Dockerfile
│   ├── index.html
│   └── nginx-server.conf
└── proxy
    ├── Dockerfile
    └── nginx-proxy.conf

2 directories, 7 files
```



```yaml
version: '3.7'
services:
  proxy:
    build: ./proxy/
    ports:
      - "8000:80"
    networks:
      internal_nw:
        ipv4_address: 172.28.0.2
  main:
    build: ./main/
    networks:
      internal_nw:
        ipv4_address: 172.28.0.3

networks:
  internal_nw:
    driver: bridge
    ipam:
      config:
        - subnet: 172.28.0.0/16

```

Running our simple multi-container app

1. docker-compose build

2. docker-compose up

3. And voilà – our „app” is running

GDAŃSK UNIVERSITY OF TECHNOLOGY

Executing commands

1. allows us to run a command inside running container

   ```
   docker exec
   ```

2. For example:

   ```
   docker exec mycontainer cat /var/log/nginx/access.log
   ```

   will print us our server access logs

3. Imagine we have a container that runs database and we need to make a backup of it, with docker exec this can be done easily

GDAŃSK UNIVERSITY OF TECHNOLOGY

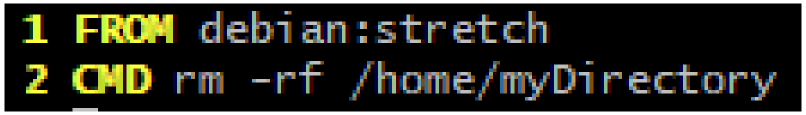
Isolation from host system

1. Docker’s filesystem is isolated from host’s filesystem as much as we want to – by default it’s fully isolated, but we can mount directory from host in container’s file system with -v or --volume option
2. Docker runs as a root – this means it can do anything with your host machine
3. Giving someone a permission to Docker is like giving permission to sudo

GDAŃSK UNIVERSITY
OF TECHNOLOGY

Isolation from host system

1. Let’s say we have simple Dockerfile:



```
FROM debian:stretch
CMD rm -rf /home/myDirectory
```

2. We build an image from it:

   docker build . -t myImage

3. And we run the container based on that image, mounting host’s / in /home/myDirectory in our container:

4. docker container run -v /:/home/myDirectory/ myImage

5. Guess what happens next?
6. BOOM! We just lost our host system

GDAŃSK UNIVERSITY OF TECHNOLOGY

Possible solutions to this problem

1. Run our container with --user flag which allows to specify uid for Docker container process
2. No warranty that other docker users will use it
3. Create a virtual machine on host and run containers on created VM
   1. Full isolation of host system
   2. It’s only a partial solution of the problem since there is possibility to harm a VM’s system
4. Conclusion: Only trusted users should have permissions to use Docker

GDAŃSK UNIVERSITY
OF TECHNOLOGY

Distributed solution

1. Docker swarm – built into the docker, works with images and docker-compose files. In short distributed docker
2. Kubernetes – separate, container based tools, very powerful but with a steep learning curve. Also works with docker-compose files. In short distributed docker on (high amount of) steroids

GDAŃSK UNIVERSITY OF TECHNOLOGY

WWW Servers

Apache HTTP Server

1. Standard, all purpose WWW server
2. Handles html, PHP, ruby, python etc.
3. A lot of extensions available via modules:
   1. proxy
   2. rewrite
   3. ssl
   4. Subversion
   5. Many others
4. Available in all Linux distributions
5. Sometimes available in 2 flavors – threaded and preforked

GDAŃSK UNIVERSITY
OF TECHNOLOGY

Config files (Linux i MS Windows (partially))

1. /etc/httpd/ or /etc/apache2 – main configuration directory
2. /var/www – default web pages catalog
3. /var/log/httpd - logs

GDAŃSK UNIVERSITY OF TECHNOLOGY

Main configuration directives

1. ServerRoot “/etc/httpd” - main configuration folder, working path
2. Listen 80 – ports on what the server is listening
3. User apache
4. Group apache – user and group used to run the server
5. ServerName – the name returned by the apache with errors etc.
6. DocumentRoot /var/www/html – main directory with web pages
7. UserDir public_html – from mod_userdir, default directory for user pages
8. DirectoryIndex index.html index.cgi index.htm – order of files displayed
9. NameVirtualHost – enabling of virtual hosts
10. LoadModule name path – used to load shared libraries as modules

GDAŃSK UNIVERSITY
OF TECHNOLOGY

Locations and directories

1. <Directory /name> … </Directory> - for physical directories in DocumentRoot
2. <Location /name> … </Location> - for logical urls
3. AllowOverride directive – allows usage of .htaccess file
4. .htaccess file can set server configuration on directory level
5. Options directive – tells what can be overridden using .htaccess (None, All, Indexes, Includes, …)

GDAŃSK UNIVERSITY
OF TECHNOLOGY

Authentication

1. User and password based
2. htpasswd passwordfile username – adding new user to the local database
3. -c – creates new file, e.g.:

```
htpasswd -c apache_vip user1
New password:
...
```

4. AuthUserFile, AuthGroupFile – directives containing paths to user (htpasswd created) and group (containing group names and coma separated user lists) files respectively

GDAŃSK UNIVERSITY OF TECHNOLOGY

Authorization

1. deny, allow directives – all, damain name, ip address
2. The order is important – deny, allow or allow,deny (default)
3. Client needs allow directive to be allowed to access the resource
4. In older systems:

```
<Location /name>
  Order deny,allow
  Deny from all
  Allow from domain.pg.gda.pl
</Location>
```

5. In newer ones:

```
<Location /name>
  Require all denied
  Require from domain.pg.gda.pl granted
</Location>
```

GDAŃSK UNIVERSITY OF TECHNOLOGY

Authorization continued

1. Require – determines who has access to a resource
   1. valid-user – anybody with correct login and password
   2. users – only selected users
   3. groups – only selected groups
   4. e.g.:

```
<Location /name>
    ...
    AuthType Basic
    AuthName “Some name displayed to the user”
    AuthUserFile “path to user file”
    Require valid-user
</Location>
```

LDAP authentication and authorization

```
#authentication type
AuthType Basic
#What to show to the user
AuthName "Nazwa okienka"
#Authentication source
AuthBasicProvider "ldap"
#LDAP server address
AuthLDAPURL "ldap://server_ldap:389/DC=kasklab,DC=...?sAMAccountName?sub?(objectClass=*)" NONE
#LDAP credentials
AuthLDAPBindDN "CN=username,CN=..."
AuthLDAPBindPassword my_password
#Should the ldap authorization be required (on) or not (off)
authzldapauthoritative Off
#Who should be allowed
Require valid-user
```

Virtual servers

1. Single WWW server can host files for multiple domains, the domains usually have the same IP address
2. The browser with the HTTP request sends the domain name that it asked for
3. The WWW server always looks on the domain name and can switch to a proper context if such is found

Name based virtual hosts

1. Single IP address, e.g. 123.45.67.89
2. Multiple domain names, e.g.: www.server1.gda.pl, www.otherserver2.gda.pl
3. NameVirtualHost 123.45.67.89 – we enable named virtual hosts on given IP
4. <VirtualHost 123.45.67.89> - the definition of the virtual host

GDAŃSK UNIVERSITY OF TECHNOLOGY

Name based virtual hosts – simple example

```
<VirtualHost 123.45.67.89>
  ServerName  www.server1.gda.pl
  DocumentRoot /var/www/server1_data
  ...
</VirtualHost>

<VirtualHost 123.45.67.89>
  ServerName  www.otherserver2.gda.pl
  DocumentRoot /var/www/otherserver2_data
  ...
</VirtualHost>
```

GDAŃSK UNIVERSITY
OF TECHNOLOGY

SSL and virtual hosts

1. By default the administrator can create any number of name based virtual hosts on a single SSL port
2. All hosts will have one common certificate
3. Even if we will define different certificate for the servers only the first one will be used for securing connections and presented to the browser
4. This will lead to warnings about not matching certificates, this can be mitigated by using wildcards certificates but only for subdomains in one common superdomain
5. The reason for that is that the browser with the HTTPS package sends only cryptography parameters, the data associated with HTTP needs to be decrypted first

GDAŃSK UNIVERSITY OF TECHNOLOGY

SNI - Server Name Indication

1. SNI aware browser adds domain name to HTTPS package
2. The server can thus read the domain name before decrypting the HTTPS package and use proper certificate for the domain name
3. SNI enabled browsers: Opera 8.0+, Firefox 2+, Internet Explorer 7+, Safari 3.2.1+, Chrome, Chromium 11.0.696.28+ (Safari and Chrome for SNI need Windows Vista or later

GDAŃSK UNIVERSITY OF TECHNOLOGY

Nginx

1. Very light, very fast
2. Uses asynchronous requests handling, this allows better server load prediction
3. By default it has basic HTTP protocol functions enabled including reverse proxy, virtual hosts, ssl, authentication etc.
4. Does not support external modules

GDAŃSK UNIVERSITY
OF TECHNOLOGY

Virtual Hosts in Nginx

1. The first step to set up virtual hosts is to create one or more server blocks in the main configuration file (/etc/nginx/nginx.conf) or inside /etc/nginx/sites-available.
2. Although the name of the configuration files in this directory (sites-available) can be set to whatever you want, it’s a good idea to use the name of the domains, and in addition we chose to add the .conf extension to indicate that these are configuration files.

GDAŃSK UNIVERSITY
OF TECHNOLOGY

Virtual Hosts in Nginx

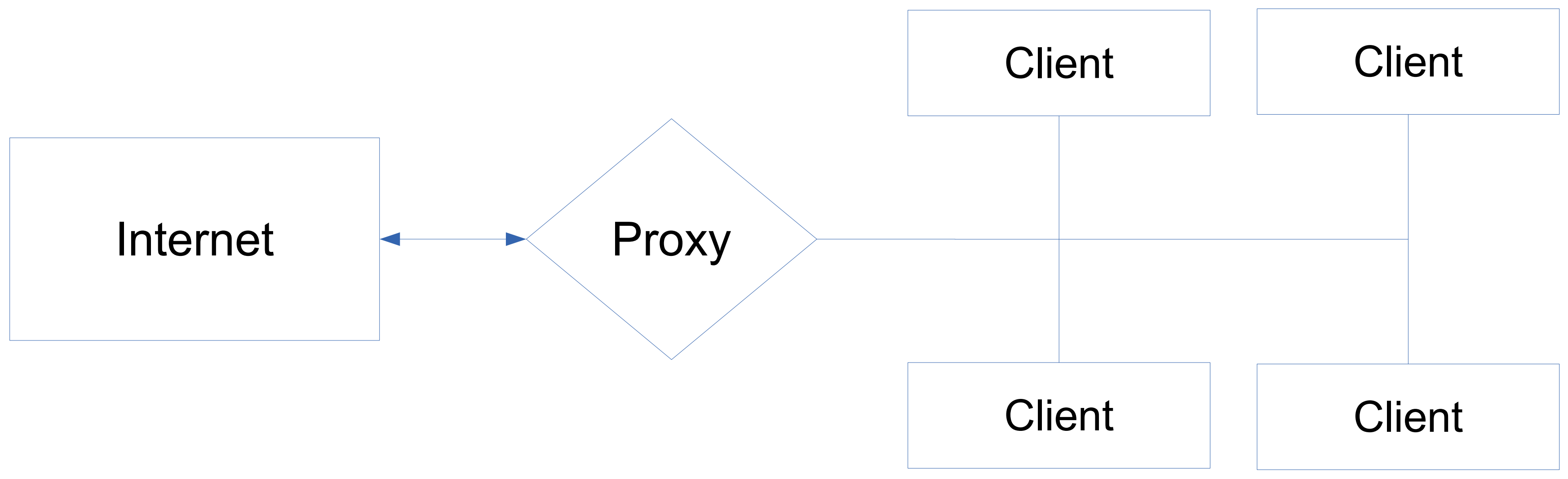
```
server {
    listen       80;
    server_name  doaminname domainalias1 ….;
    access_log  /path/to/access.log;
    error_log  /path/to/access.log;
    root   /path/to/root/folder;
    index  index.html index.htm;
}
```

GDAŃSK UNIVERSITY OF TECHNOLOGY

Proxy servers

GDAŃSK UNIVERSITY OF TECHNOLOGY

Proxy servers



GDAŃSK UNIVERSITY
OF TECHNOLOGY

Proxy server

1. Act as a broker between network clients and other servers
2. Can act as a buffer for data retrieved from other servers
3. Can modify both user queries and responses to those queries

GDAŃSK UNIVERSITY OF TECHNOLOGY

ReverseProxy

1. Used to transfer requests from one WWW server to other server/domain
2. Works only for strict list of addresses previously defined by the administrator
3. Usually used when the requests should be served by different web server (e.g. IIS, Glassfish etc.). One server than acts as a public access place and passess the request to other, private servers located within the LAN segment.

GDAŃSK UNIVERSITY OF TECHNOLOGY

ReverseProxy (2)

```
ProxyPass / http://88.88.88.88:10001/
ProxyPassReverse / http://88.88.88.88:10001/
RewriteEngine on
RewriteRule ^/$ / [R] # Fixes broken images etc

HostnameLookups Off
UseCanonicalName Off
SetEnv force-proxy-request-1.0 1
SetEnv proxy-nokeepalive 1
```

GDAŃSK UNIVERSITY
OF TECHNOLOGY

Apache ReverseProxy

```
<VirtualHost *:80>
   ServerName www.nazwa.pl

   ProxyRequests off
   ProxyPreserveHost Off | On
#WATCH FOR slashes here!
   ProxyPass /strona1 http://153.19.50.12/strona1
   ProxyPassReverse /strona1 http://153.19.50.12/strona1
#especially at the end of THIS lines!
   ProxyPass / http://153.19.50.12/strona2/
   ProxyPassReverse / http://153.19.50.12/strona2/

   HostnameLookups Off
   UseCanonicalName Off
   SetEnv force-proxy-request-1.0 1
   SetEnv proxy-nokeepalive 1
</VirtualHost>
```

GDAŃSK UNIVERSITY
OF TECHNOLOGY

ProxyHTMLURLMap

```
#unzip mod_proxy_html.zip
#apxs -c -I /usr/include/libxml2/ -I . -i mod_proxy_html.c

#chmod 755 /usr/lib64/httpd/modules/mod_proxy_html.so

cd /etc/httpd/conf
```

edit the httpd.conf file and add the following entry

```
LoadFile /usr/lib64/libxml2.so
LoadModule proxy_html_module modules/mod_proxy_html.so
```

GDAŃSK UNIVERSITY OF TECHNOLOGY

Nginx reverse proxy

```
location / {
  proxy_set_header Accept-Encoding "";
  sub_filter_types *;
  sub_filter      wikiws.os.niwa.gda.pl:81 wikiws.os.niwa.gda.pl;
  sub_filter_once off;
  proxy_pass   http://localhost:81/;
  proxy_redirect off;
  proxy_set_header   Host             $host;
  proxy_set_header   X-Real-IP        $remote_addr;
}
```

GDAŃSK UNIVERSITY
OF TECHNOLOGY

Nginx reverse proxy (2)

```
location ~ ^/gf/([0-9]+)/(.*)$ {
  proxy_set_header Accept-Encoding "";
  sub_filter_types *;
  sub_filter      wikiws.os.niwa.gda.pl:80 wikiws.os.niwa.gda.pl/gf/$1;
  sub_filter_once off;

  proxy_pass  http://192.168.100.136:$1/$2$is_args$args;
  proxy_next_upstream error timeout invalid_header http_500 http_502 http_503 http_504;
  proxy_redirect off;
  proxy_buffering off;
  proxy_set_header        Host            $host;
  proxy_set_header        X-Real-IP       $remote_addr;
  proxy_set_header        X-Forwarded-For $proxy_add_x_forwarded_for;
  proxy_connect_timeout       360;
  proxy_send_timeout          360;
  proxy_read_timeout          360;
  send_timeout                360;
}
```

GDAŃSK UNIVERSITY OF TECHNOLOGY

HAproxy

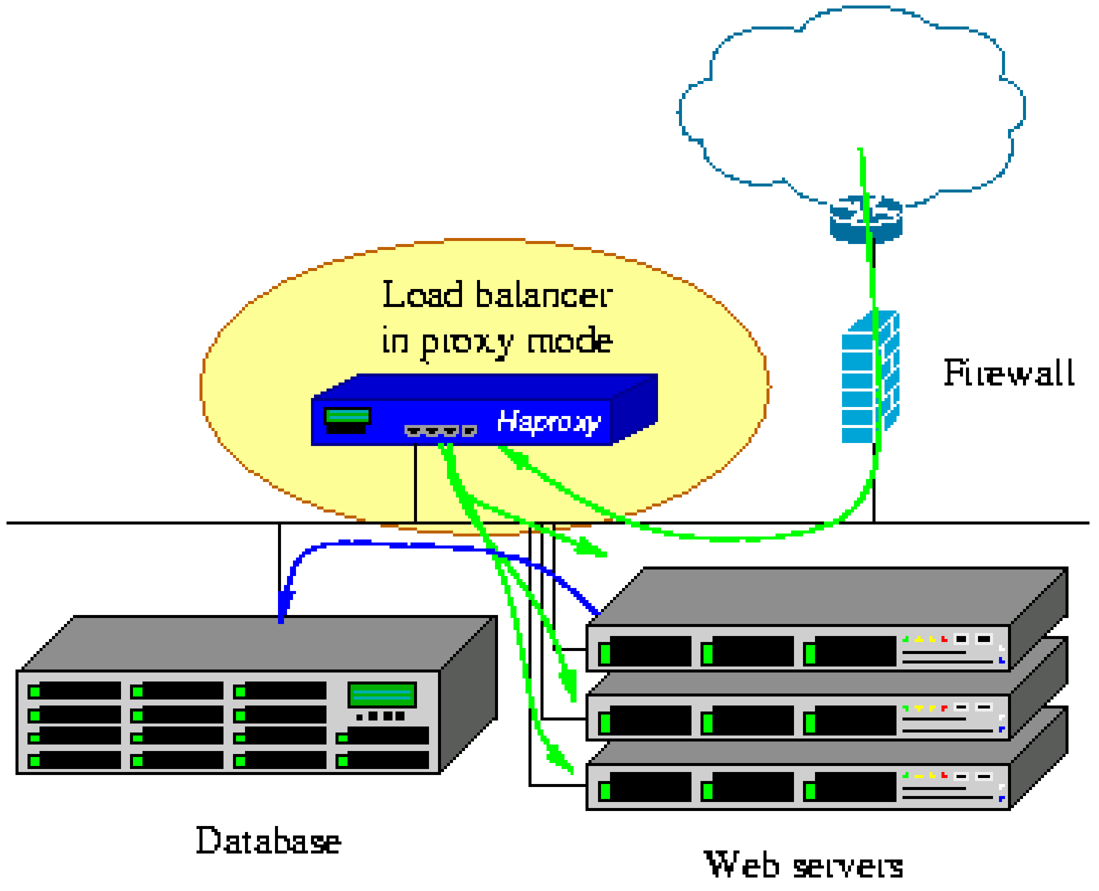
1. HAProxy is a free, very fast and reliable solution offering high availability, load balancing, and proxying for TCP and HTTP-based applications.
2. It is particularly suited for very high traffic web sites and powers quite a number of the world's most visited ones.
3. Over the years it has become the de-facto standard opensource load balancer, is now shipped with most mainstream Linux distributions, and is often deployed by default in cloud platforms
4. Main features: native SSL support with SNI/NPN/ALPN, IPv6 and UNIX sockets are supported everywhere, full HTTP keep-alive, HTTP/1.1 compression, ACLs, improved health checks, etc.

Can function as

1. a TCP proxy
2. an HTTP reverse-proxy
3. an SSL terminator / initiator / offloader
4. an HTTP fixing tool
5. a content-based switch
6. a server load balancer
7. a traffic regulator
8. an HTTP compression offloader
9. a caching proxy
10. a FastCGI gateway

GDAŃSK UNIVERSITY OF TECHNOLOGY

Haproxy



GDAŃSK UNIVERSITY
OF TECHNOLOGY

Load balancing methods

1. No Load Balancing
2. Layer 4 Load Balancing - the simplest way to load balance network traffic to multiple servers is to use layer 4 (transport layer) load balancing. Load balancing this way will forward user traffic based on IP range and port (i.e. if a request comes in for http://yourdomain.com/anything, the traffic will be forwarded to the backend that handles all the requests for yourdomain.com on port 80).
3. Layer 7 Load Balancing - Another, more complex way to load balance network traffic is to use layer 7 (application layer) load balancing. Using layer 7 allows the load balancer to forward requests to different backend servers based on the content of the user's request. This mode of load balancing allows you to run multiple web application servers under the same domain and port.

GDAŃSK UNIVERSITY OF TECHNOLOGY

Load Balancing Algorithms

1. roundrobin - Round Robin selects servers in turns. This is the default algorithm.
2. leastconn - selects the server with the least number of connections–it is recommended for longer sessions. Servers in the same backend are also rotated in a round-robin fashion.
3. source - this selects which server to use based on a hash of the source IP i.e. your user’s IP address. This is one method to ensure that a user will connect to the same server.

Sample configuration for load balancing - backend

```
backend web-backend
   balance roundrobin
   server web1 web1.yourdomain.com:80 check
   server web2 web2.yourdomain.com:80 check

backend blog-backend
   balance roundrobin
   mode http
   server blog1 blog1.yourdomain.com:80 check
   server blog1 blog1.yourdomain.com:80 check
```

1. balance roundrobin line specifies the load balancing algorithm
2. mode http specifies that layer 7 proxying will be used
3. The check option at the end of the server directives specifies that health checks should be performed on those backend servers.

GDAŃSK UNIVERSITY OF TECHNOLOGY

Sample configuration for load balancing - frontend

```
frontend http
  bind *:80
  mode http

  acl url_blog path_beg /blog
  use_backend blog-backend if url_blog

  default_backend web-backend
```

1. This configures a frontend named http, which handles all incoming traffic on port 80.
2. acl url_blog path_beg /blog matches a request if the path of the user’s request begins with /blog.
3. use_backend blog-backend if url_blog uses the ACL to proxy the traffic to blog-backend.
4. default_backend web-backend specifies that all other traffic will be forwarded to web-backend.

GDAŃSK UNIVERSITY OF TECHNOLOGY

Other deployment problems

Case sensitivity

1. Linux is case sensitive, Windows is not (well, sort of, both NTFS and Windows itself support case sensitivity but it is disabled by default)
    1. This is important e.g. when we are getting resources over the web. Reference to someimg.JPG file referenced to as someimg.jpg on Windows Server will work, it will not work on Linux
    2. Beware when mounting Linux filesystems under windows, especially remotely
2. Java is case sensitive regardless of operating system
3. MySQL is case sensitive or case insensitive dependently of the operating system – the same app might stop working after migrating to Linux

GDAŃSK UNIVERSITY
OF TECHNOLOGY

Stability vs bleeding edge

1. Every serious server Linux version has rather old packages, especially those related to software development
2. Kernel is very rapidly developed, current RHEL version (7) still uses kernel from the 3.x family
3. RHEL 7 uses GCC 4.7
4. Fortunately we have devtoolset packages, we have to install:
   1. scl-utils
   2. centos-release-scl
   3. devtoolset-X-package_name
5. Too modern system versions have problems with external packages like CUDA, they are just not supported

GDAŃSK UNIVERSITY
OF TECHNOLOGY

Certificates

1. A real pain
2. More and more solutions requires full, signed certificates with proper CA
3. You cannot easily use self-signed certificates – more and more services just reject them, even in testing environment
4. The solution? Apache/nginx/haproxy reverse proxy with letsencrypt and certbot

Let’s Encrypt

1. https://letsencrypt.org/
2. Non profit organization
3. Free, fully signed certificates
4. Certbot allows certificate installation and update automation
5. https://certbot.eff.org/

GDAŃSK UNIVERSITY
OF TECHNOLOGY

SSH

Remote connection using SSH

1. Remote connection using encrypted channel

   ```
   ssh user1@hostname1.eti.pg.gda.pl
   ```
2. Remote command execution

   ```
   ssh user1@komputer1.eti.pg.gda.pl ls
   ```
3. General syntax: `ssh [-l login_name] hostname | user@hostname [command]`

GDAŃSK UNIVERSITY
OF TECHNOLOGY

SSH configuration

1. Configuration files:
   1. Server: /etc/ssh/sshd_config
   2. Client: /etc/ssh/ssh_config
2. SSHD keys
   1. Public and private
   2. ~/.ssh/authorized_keys
   3. ssh-keygen
3. For public systems it is advised to turn off root login – we should always use su or sudo: PermitRootLogin no

Remote file copy

1. scp – copies files using encrypted connection

```
scp [options] [[user@]host1:]file1 … [[user@]host2:]file12
```

2. Local file to remote computer

```
scp file1 user1@hostname:/home/user1/directory/file1
```

3. Remote file to local directory

```
scp user1@hostname:/home/user1/directory/file1 local_path
```

GDAŃSK UNIVERSITY OF TECHNOLOGY

What to do to copy a lot of files?

1. Better is to use rsync
2. scp opens new connection for each file, only copies the file
3. rsync can set proper flags, retain permissions, sync directories etc.
4. To sync /mnt/server-mirror/home/ of a local computer with some server we should use:

   ```
   rsync -av --delete -e ssh root@server.example.com:/home/
   /mnt/server-mirror/home/
   ```
5. And with ssh keys:

   ```
   rsync -av --delete -e "ssh -i /home/user/.ssh/sshkey"
   root@server.example.com:/home/ /mnt/server-mirror/home/
   ```
6. -a option stands for –archive, which includes: -r (--recursive), -l (--links), -p (--perms), -t (--times), -g (--group), -o (--owner, can be done only as root), -D (--devices --specials, can be done only as root)

GDAŃSK UNIVERSITY OF TECHNOLOGY

Not encrypted connections

1. Do not use unless needed
2. telnet, rlogin – remote unencrypted login
3. rsh – remote command execution (unencrypted)

   rsh [opcje] [-l username] host [command]
4. rcp – remote file copy (unencrypted)

   rcp [-px] [-r] file ... directory (file1 file2)

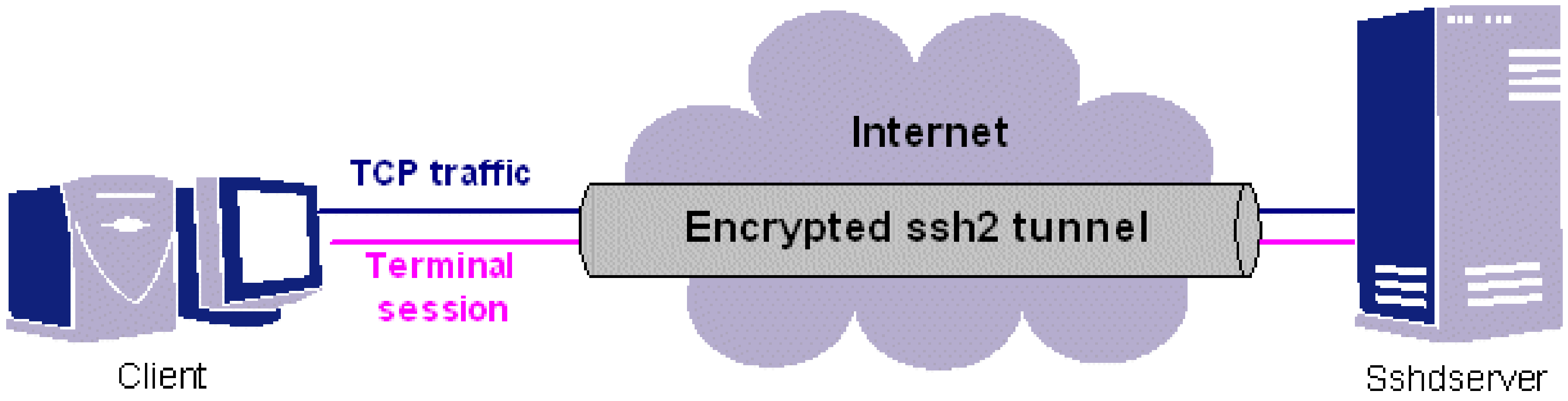
GDAŃSK UNIVERSITY
OF TECHNOLOGY

SSH tunneling

1. Easy way to secure unencrypted connections
2. Each packet sent to entry port of the tunnel is encapsulated by secure ssh connection and forwarded to remote host, there it is decrypted and forwarded to destination port

GDAŃSK UNIVERSITY OF TECHNOLOGY

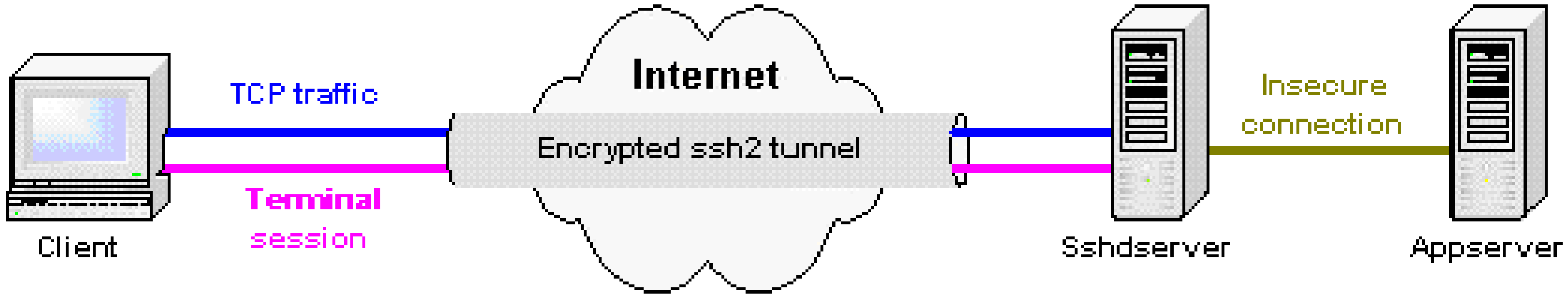
SSH tunneling (2)



```
ssh -L 1234:localhost:23 username@Sshdserver
```

All connections made to localhost port 1234 will be forwarded to port 23 on Sshdserver host. On the remote computer we need to have running sshd server

SSH tunneling (3)



```
ssh -L 1234:Appserver:23 username@Sshdserver
```

All connections made to localhost port 1234 will be forwarded to port 23 on Appserver host using ssh service on Sshdserver.