

TP01 Orchestration

Réalisé par Alexis Plessias, Tom Thioulouse,
Vincent Lagogué, David Tejeda et Thomas
Peugnet

On choisit de partir sur une configuration Docker.

Dans un premier temps on vérifie que notre Docker est bien installé :

```
→ ~ docker -v  
Docker version 26.0.0, build 2ae903e
```

Ensuite on récupère le conteneur de test de Docker :

```
→ ~ sudo docker pull hello-world  
Using default tag: latest  
latest: Pulling from library/hello-world  
c1ec31eb5944: Pull complete  
Digest: sha256:53641cd209a4fecfc68e21a99871ce8c6920b2e7502df0a20671c6fccc73a7c6  
Status: Downloaded newer image for hello-world:latest  
docker.io/library/hello-world:latest
```

L'image est bien récupérée, on peut donc exécuter le conteneur via la commande "Docker run".

```
→ ~ sudo docker run hello-world  
  
Hello from Docker!  
This message shows that your installation appears to be working correctly.  
  
To generate this message, Docker took the following steps:  
1. The Docker client contacted the Docker daemon.  
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.  
   (amd64)  
3. The Docker daemon created a new container from that image which runs the  
   executable that produces the output you are currently reading.  
4. The Docker daemon streamed that output to the Docker client, which sent it  
   to your terminal.  
  
To try something more ambitious, you can run an Ubuntu container with:  
$ docker run -it ubuntu bash  
  
Share images, automate workflows, and more with a free Docker ID:  
https://hub.docker.com/  
  
For more examples and ideas, visit:  
https://docs.docker.com/get-started/
```

Le conteneur est bien lancé, celui-ci s'arrête automatiquement après l'exécution.

Ensuite, on nous demande d'exécuter un conteneur contenant un serveur web Nginx on l'exécute en arrière-plan, pour savoir s'il est déployé on peut faire la commande Docker ps :

```
→ ~ sudo docker run --name nginx -d nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
8a1e25ce7c4f: Pull complete
e78b137be355: Pull complete
39fc875bd2b2: Pull complete
035788421403: Pull complete
87c3fb37cbf2: Pull complete
c5cdd1ce752d: Pull complete
33952c599532: Pull complete
Digest: sha256:6db391d1c0cfb30588ba0bf72ea999404f2764feb0f1f196acd5867ac7efa7e
Status: Downloaded newer image for nginx:latest
ee478acadb9bc871743fec8fb513716de8dfa55b234460bf50bbaad801413100
→ ~ sudo docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS        NAMES
ee478acadb9b   nginx     "/docker-entrypoint...." 15 seconds ago Up 14 seconds 80/tcp       nginx
→ ~
```

Si on veut stopper notre conteneur on récupère son ID et on l'arrête avec "Docker stop" :

```
→ ~ sudo docker ps
CONTAINER ID   IMAGE     COMMAND                 
ee478acadb9b   nginx     "/docker-entrypoint...."
→ ~ sudo docker stop ee478acadb9b
ee478acadb9b
→ ~
```

Si on refait un Docker ps ensuite notre conteneur Nginx aura disparu :

```
→ ~ sudo docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS        NAMES
→ ~
```

On consulte toutes les images installés sur notre docker via la commande "image" :

```
→ ~ sudo docker images
REPOSITORY      TAG         IMAGE ID      CREATED        SIZE
manios/nagios    latest      b01feb5a4626  2 weeks ago   197MB
nginx            latest      92b11f67642b  6 weeks ago   187MB
jasonrivers/nagios latest      79a7fc3a2f88  2 months ago  792MB
hello-world      latest      d2c94e258dcb  11 months ago 13.3kB
→ ~
```

Pour supprimer l'image on va faire comme précédemment, on reprend l'ID de notre image et on exécute "docker rmi -f" :

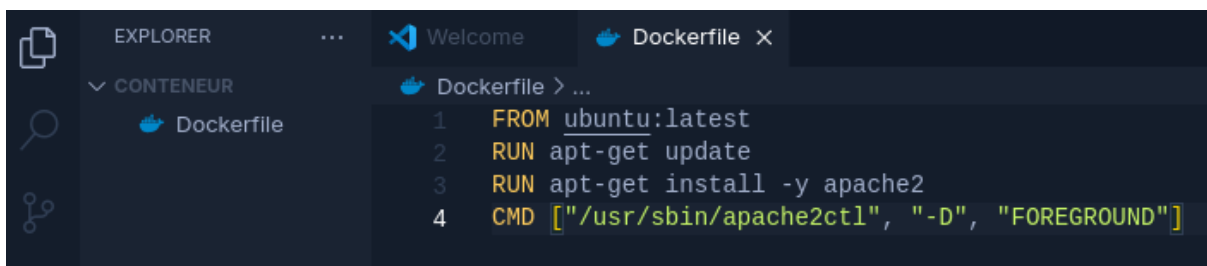
```
→ ~ sudo docker rmi -f d2c94e258dcb
Untagged: hello-world:latest
Untagged: hello-world@sha256:53641cd209a4fecfc68e21a99871ce8c6920b2e7502df0a20671c6fccc73a7c6
Deleted: sha256:d2c94e258dcb3c5ac2798d32e1249e42ef01cba4841c2234249495f87264ac5a
→ ~ sudo docker images
REPOSITORY          TAG             IMAGE ID        CREATED         SIZE
manios/nagios        latest          b01feb5a4626    2 weeks ago    197MB
nginx                latest          92b11f67642b    6 weeks ago    187MB
jasonrivers/nagios   latest          79a7fc3a2f88    2 months ago   792MB
→ ~
```

On voit bien que notre image a bien été supprimée.

On récupère l'image Alpine et on lance un "docker image" pour vérifier qu'elle est bien installée :

```
→ ~ sudo docker pull alpine:3.17.3
[sudo] password for rda-18:
3.17.3: Pulling from library/alpine
f56be85fc22e: Pull complete
Digest: sha256:124c7d2707904eea7431fffe91522a01e5a861a624ee31d03372cc1d138a3126
Status: Downloaded newer image for alpine:3.17.3
docker.io/library/alpine:3.17.3
→ ~ sudo docker images
REPOSITORY          TAG             IMAGE ID        CREATED         SIZE
manios/nagios        latest          b01feb5a4626    2 weeks ago    197MB
nginx                latest          92b11f67642b    6 weeks ago    187MB
jasonrivers/nagios   latest          79a7fc3a2f88    2 months ago   792MB
alpine               3.17.3          9ed4aefc74f6    12 months ago   7.05MB
```

On crée notre fichier Dockerfile qui nous permet de créer notre image personnalisée:

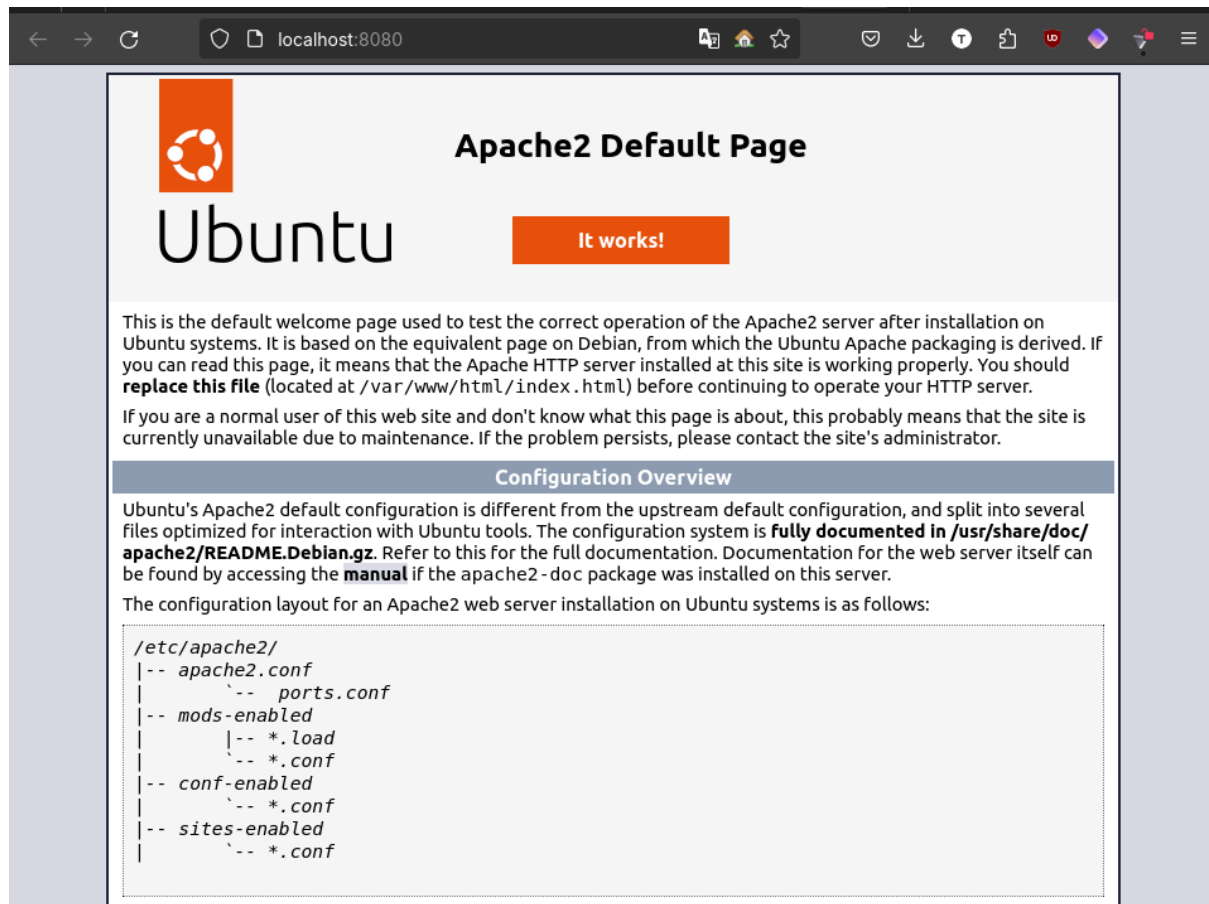


```
1 FROM ubuntu:latest
2 RUN apt-get update
3 RUN apt-get install -y apache2
4 CMD ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]
```

Cette image est une image ubuntu que l'on met à jour. On y installe apache2 et on lance le service en premier plan, ainsi apache sera toujours fonctionnel.

Pour éviter davantage de pages inutiles nous ne rajouterons plus les captures de "docker ps". Le TP sera donc plus léger.

On se connecte à la page en localhost sur le port 8080 pour tester le fonctionnement:



On accède aux logs et on constate que le serveur n'a pas trouvé de nom. L'erreur vient du service apache.

```
➔ Conteneur sudo docker logs -f 02c69aebea42
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.17.0.2. Set the 'ServerName' directive globally to suppress this message
```

On crée et on regarde les propriétés du réseau qu'on a créé.

```
→ Conteneur sudo docker network inspect altom-network
[
  {
    "Name": "altom-network",
    "Id": "ba61eefa0b6412b9bf9bd4756186626de18fd41bca51cb0500e2b2ec9b018315",
    "Created": "2024-03-29T15:59:47.374906676+01:00",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "172.18.0.0/16",
          "Gateway": "172.18.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {},
    "Options": {},
    "Labels": {}
  }
]
→ Conteneur
```

On copie un fichier sur le PC dans le répertoire "tmp" du conteneur.

```
→ Conteneur sudo docker pull rockylinux:9.1
9.1: Pulling from library/rockylinux
9d28f3f24f51: Pull complete
Digest: sha256:972a6b9cec75b2f47bf0faa425135e1bdb9945306a22f26db0b3a4ee8a289687
Status: Downloaded newer image for rockylinux:9.1
docker.io/library/rockylinux:9.1
→ Conteneur sudo docker run --name test -id rockylinux:9.1
7447157da3366479bd59174008bd19ef55ffa8ae11a386d29e83dac74621bbed
→ Conteneur echo "hello world" > file.txt
→ Conteneur sudo docker cp $(pwd)/file.txt test:/tmp/
Successfully copied 2.05kB to test:/tmp/
→ Conteneur sudo docker exec -it test sh
sh-5.1# ls
afs  dev  home  lib64      media  opt   root  sbin  sys  usr
bin  etc  lib   lost+found mnt    proc  run   srv   tmp  var
sh-5.1# cd /tmp
sh-5.1# ls
file.txt
sh-5.1# cat file.txt
hello world
sh-5.1#
```

On crée et on inspecte le nouveau volume.

```
→ Conteneur sudo docker volume inspect altota
[
  {
    "CreatedAt": "2024-03-29T16:15:03+01:00",
    "Driver": "local",
    "Labels": null,
    "Mountpoint": "/var/snap/docker/common/var-lib-docker/volumes/altota/_data",
    "Name": "altota",
    "Options": null,
    "Scope": "local"
  }
]
→ Conteneur
```

On accède à la page de présentation de nginx.

```
→ Conteneur sudo docker rm -f 9fbd9648eefe769239f0862535428fe7ee25251757a33b2ffc93db2a0229fc3b
9fbd9648eefe769239f0862535428fe7ee25251757a33b2ffc93db2a0229fc3b
→ Conteneur sudo docker run -d --name altoginx -p 8080:80 -v altota:/usr/share/nginx/html nginx
ea2956fe4faaf0dd13e81add49633344ce741e2fe26f8d9f1fa7056b900a32c5
→ Conteneur curl http://127.0.0.1:8080
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
→ Conteneur
```

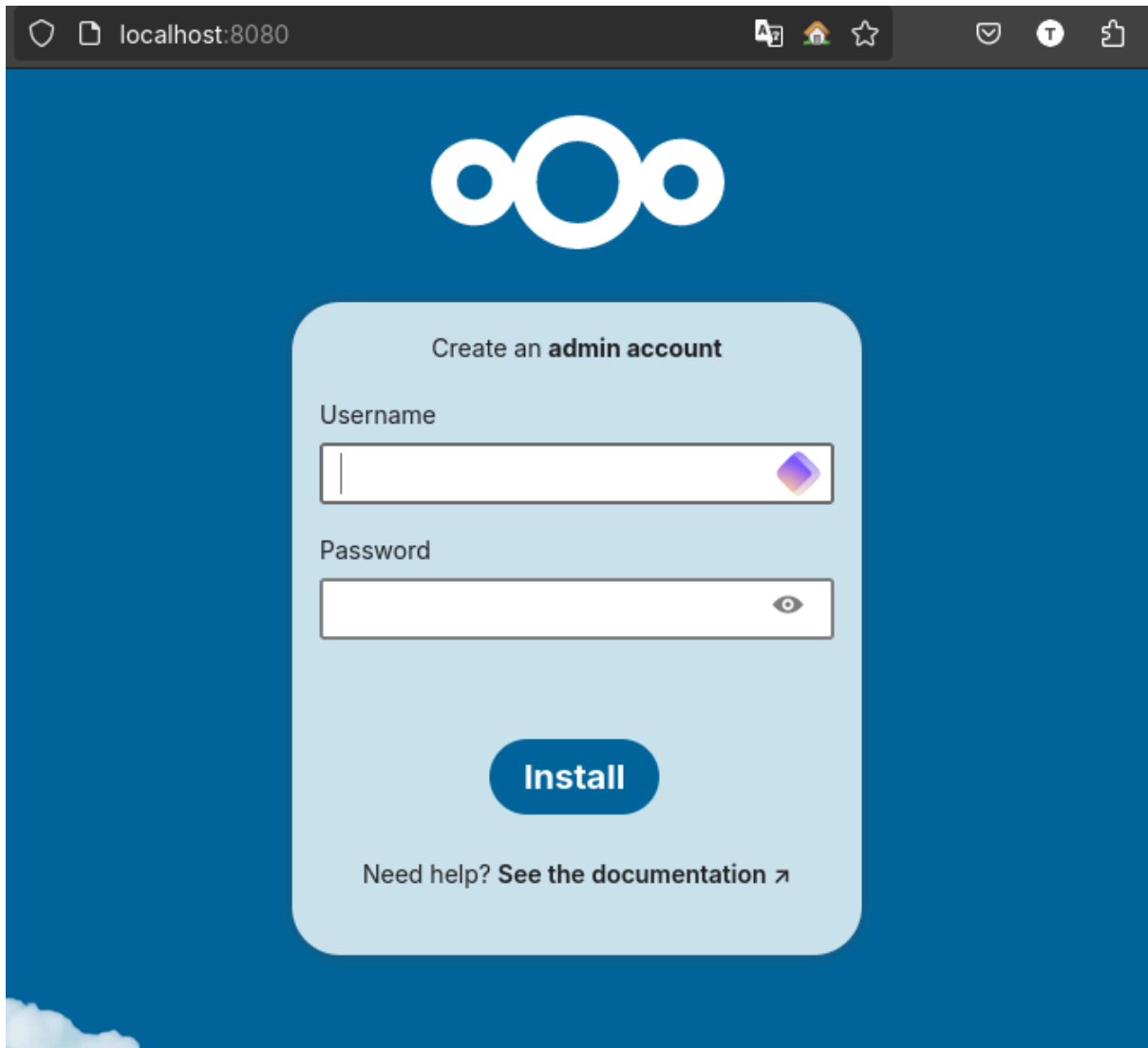
On modifie la page web et on affiche les changements.

```
→ Conteneur echo '''
<html>
Hello World
</html>
''' > index.html
→ Conteneur ll
total 16K
-rw-rw-r-- 1 rda-18 rda-18 117 mars 29 15:06 Dockerfile
-rw-rw-r-- 1 rda-18 rda-18 12 mars 29 16:08 file.txt
-rw-rw-r-- 1 rda-18 rda-18 29 mars 29 16:33 index.html
-rw-rw-r-- 1 rda-18 rda-18 53 mars 29 16:28 salut.html
→ Conteneur sudo docker cp $(pwd)/index.html altoginx:/usr/share/nginx/html/
Successfully copied 2.05kB to altoginx:/usr/share/nginx/html/
→ Conteneur curl http://127.0.0.1:8080

<html>
Hello World
</html>
→ Conteneur
```

Quelle est pour vous la différence entre le bind mount et le volume ?

Avec le bind mount on crée notre conteneur depuis un répertoire alors que le volume est un stockage qui peut être monté sur les différentes machines.



Après la création du code en docker compose, nous lançons les conteneurs et nous accédons à l'interface nextcloud via notre navigateur.

usr/src/nextcloud/3rdparty/composer.lock (composer)

Total: 3 (UNKNOWN: 0, LOW: 1, MEDIUM: 0, HIGH: 2, CRITICAL: 0)

Library	Vulnerability	Severity	Status	Installed Version	Fixed Version	Title
aws/aws-sdk-php	CVE-2023-51651	LOW	fixed	3.240.8	3.288.1	Potential URI resolution path traversal in the AWS SDK for PHP https://avd.aquasec.com/nvd/cve-2023-51651
phpseclib/phpseclib	CVE-2024-27354	HIGH		2.0.45	1.0.23, 2.0.47, 3.0.36	An issue was discovered in phpseclib 1.x before 1.0.23, 2.x before 2.0... https://avd.aquasec.com/nvd/cve-2024-27354
	CVE-2024-27355				1.0.23, 3.0.36, 2.0.47	An issue was discovered in phpseclib 1.x before 1.0.23, 2.x before 2.0... https://avd.aquasec.com/nvd/cve-2024-27355

→ Conteneur

Selon Trivy nous avons deux vulnérabilités de type High et une Low. La Low peut être mise de côté, mais les highs ont besoins d'une inspection :

Nous allons donc sur le site <https://cve.mitre.org/index.html> pour avoir le détail

La CVE-2024-273534 est un certificat lié à phpseclib mal formé qui peut permettre une attaque par DDOS.

CVE-ID	
CVE-2024-27354	Learn more at National Vulnerability Database (NVD) • CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information
Description	
An issue was discovered in phpseclib 1.x before 1.0.23, 2.x before 2.0.47, and 3.x before 3.0.36. An attacker can construct a malformed certificate containing an extremely large prime to cause a denial of service (CPU consumption for an isPrime primality check). NOTE: this issue was introduced when attempting to fix CVE-2023-27560.	

La CVE-2024-27355 parle d'un problème similaire de certificat sur phpseclib qui peut également mener à du DDOS.

CVE-ID	
CVE-2024-27355	Learn more at National Vulnerability Database (NVD) • CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information
Description	
An issue was discovered in phpseclib 1.x before 1.0.23, 2.x before 2.0.47, and 3.x before 3.0.36. When processing the ASN.1 object identifier of a certificate, a sub identifier may be provided that leads to a denial of service (CPU consumption for decodeOID).	

Ces deux failles critiques peuvent être un problème et doivent être fixées avant d'utiliser l'image.