Containers

Docker

Docker to popularne narzędzie do konteneryzacji służące do dostarczania aplikacjom oprogramowania z systemem plików zawierającym wszystko, czego potrzebują do uruchomienia. Możemy na nim uruchomić bazy danych, serwery Apache, konkretne narzędzia, a wszystko to w środowisku w pełni odizolowanym od innych aplikacji. Uruchamianie kontenera odbywać się będzie komendą `docker run` i pozwala nam błyskawicznie skalować aplikację poprzez dodawanie dodatkowych zasobów.

Podstawowe komendy Docker’a

Przydatne w administracji komendy docker

```
# Pobranie najnowszej wersji obrazu
sudo docker pull nazwa_obrazu

# Pobranie konkretnej wersji obrazu
sudo docker pull nazwa_obrazu:wersja

# Wyszukanie obrazu w dockerhubie
sudo docker search nazwa_obrazu

# Pokazanie wszystkich pobranych obrazów
sudo docker images

# Usunięcie obrazu
sudo docker rmi id_obrazu

# Pokazanie wszystkich aktywnych kontenerów
sudo docker ps

# Pokazanie wszystkich stworzonych kontenerów
sudo docker ps -a

# stworzenie kontenera
sudo docker create nazwa_obrazu

# uruchamianie kontenera
sudo docker run <nazwa_kontenera>

# start kontenera
sudo docker start

# zatrzymanie kontenera
sudo docker stop

# kopiowanie do/z kontera (działa podobnie jak cp lub scp)
sudo docker cp <źródło> <cel>

# wejście do kontenera w celu uruchomienia w nim jakiś rzeczy
sudo docker exec -it <nazwa_kontenera> bash
```

Ćwiczenie - Administracja Dockerem i zabezpieczenie połączenia TLS

Aktualizacja `apt` w celu przygotowania do instalacji

```
# W celu instalacji dockera najlepiej odinstalować starsze wersje.
sudo apt-get remove docker docker-engine docker.io containerd runc

# Następnie należy zaktualizować repozytoria, zaczynając od naszego systemu
sudo apt-get update

# W celu umożliwienia korzystania apt z HTTPS należy wydać odpowiednią komendę
sudo apt-get install \
apt-transport-https \
ca-certificates \
curl \
gnupg-agent \
software-properties-common
```

Weryfikacja klucza GPG

```
# Po aktualizacji należy dodać klucz GPG Dockera
curl -fsSL https://download.docker.com/linux/ubuntu/gpng | sudo apt-key add -

# Kolejnym krokiem jest weryfikacja klucza
sudo apt key fingerprint 0EBFCD88
# Konfiguracja repozytorium
sudo add-apt-repository \
"deb [arch=amd64] https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) \
stable"
```

Po konfiguracji należy zainstalować Dockera

```
sudo apt-get install docker-ce docker-ce-cli containerd.io

#  Jeżeli chcielibyśmy zainstalować inną wersję Dockera należy zobaczyć jakie wersje mamy dostępne
apt-cache madison docker-ce
# A następnie zainstalować wybraną przez siebie wersję
sudo apt-get install docker-ce=<VERSION> docker-ce-cli=<VERSION> containerd.io # gdzie <VERSION> to nasza wersja
```

W celu sprawdzenia poprawności działania Dockera należy sprawdzić czy możemy uruchomić obraz hello-world

```
sudo docker run hello-world
```

Jeżeli chcesz uniknąć ciągłego używania sudo z poleceniami dockera wystarczy dodać użytkownika do grupy docker

```
sudo usermod -aG docker student # dodanie użytkownika o nazwie student do grupy docker
# Teraz należy aktywować zmiany w grupach
newgrp docker
```

```
# Sprawdźmy czy jesteśmy w grupie docker
groups
```

W celu uruchomienia dockera automatycznie przy starcie systemu należy skorzystać z poleceń poniżej

```
sudo systemctl enable docker.service
sudo systemctl enable containerd.service

# Jeżeli zdecydowalibyśmy się na wyłączenie startu Dockera przy uruchomieniu systemu wystarczy wyłączyć powyższe usługi
sudo systemctl disable docker.service
sudo systemctl disable containerd.service
```

W celu akceptacji żądań od klientów zdalnych należy skonfigurować odpowiedni plik

```
sudo systemctl edit docker.service
```

Treść pliku

```
[Service]
ExecStart=/usr/bin/dockerd -H fd:// -H tcp://127.0.0.1:2375
```

Teraz należy załadować ponownie konfigurację

```
sudo systemctl deamon-reload
# Sprawdźmy czy Docker uruchomił się na odpowiednim porcie

sudo netstat -lntp | grep docker
```

W celu łączenie się z Dockerem przez bezpieczne połączenie należy skonfigurować TLS. Zaczynając od wygenerowania kluczy

```
openssl genrsa -aes256 -out ca-key.pem 4096
# Utworzenie urzędu certyfikatu

openssl req -new -x509 -days 365 -key ca-key.pem -sha256 -out ca.pem

# Następnie w celu ułatwienia komend warto ustawić zmienną zawierającą nazwę naszego hosta
export HOST=nzwa_hosta

# Kolejnym krokiem jest wygenerowanie klucza serwera i podpisanie certyfikatu
openssl genrsa -out server-key.pem 4096
openssl req -subj '/CN=client' -new -key key.pem -out client.csr

# Określanie adresów IP
echo subjectAltName = DNS:$HOST,IP:10.10.10.20,IP:127.0.0.1 >> extfile.cnf
# Rozszerzenie atrybutu klucza platformy Docker w celu uwierzytelniania tylko dla serwera
echo extendedKeyUsage = serverAuth >> extfile.cnf

# Teraz wygenerujemy podpisany certyfikat
```

```
openssl x509 -req -days 365 -sha256 -in server.csr -CA ca.pem -CAkey ca-key.pem \
-CAcreateserial -out server-cert.pem -extfile extfile.cnf
```

W celu uwierzytelniania klienta należy utworzyć klucz klienta i żądania podpisania certyfikatu

```
openssl genrsa -out key.pem 4096
# Aby klucz był odpowiedni do uwierzytelniania klienta należy utworzyć nowy plik konfiguracyjny rozszerzeń
echo extendedKeyUsage = clientAuth > extfile-client.cnf

# A następnie wygenerować podpisany certyfikat
openssl x509 -req -days 365 -sha256 -in client.csr -CA ca.pem -CAkey ca-key.pem \
-CAcreateserial -out cert.pem -extfile extfile-client.cnf

# Po utworzeniu podpisanego certyfikatu możemy usunąć niektóre z plików
rm -v client.csr server.csr extfile.cnf extfile-client.cnf
```

Warto teraz zabezpieczyć swoje klucze i uchronić certyfikat przed nieuprawnionym zapisem

```
chmod -v 0400 ca-key.pem key.pem server-key.pem
chmod -v 0444 ca.pem server-cert.pem cert.pem
```

Zawęźmy korzystanie z Dockera tylko do osób posiadających nasz certyfikat

```
dockerd --tlsverify --tlscacert=ca.pem --tlscert=server-cert.pem --tlskey=server-key.pem \
-H=0.0.0.0:2376

# Teraz możemy sprawdzić czy jesteśmy w stanie zalogować się na Dockera z innego hosta niż KURS, wydając odpowiednie polecenie
docker --tlsverify --tlscacert=ca.pem --tlscert=cert.pem --tlskey=key.pem \
-H=$HOST:2376 version
# Jeżeli otrzymamy błąd oznacza to, że poprawnie skonfigurowaliśmy bezpieczene połączenie
```

Jeśli chcesz zabezpieczyć swoje pliki Dockera skorzystaj z poniższych poleceń

```
mkdir -pv ~/.docker
cp -v {ca,cert,key}.pem ~/.docker
export DOCKER_HOST=tcp://$HOST:2376 DOCKER_TLS_VERIFY=1

# Możemy też określić lokalizację w której będą przechowywane klucze
export DOCKER_CERT_PATH=~/.docker/zone1/
```

Warto zaznaczyć, że w tym przypadku przy każdym uruchomieniu Dockera musimy wykonać dodatkowe polecenie

```
docker --tlsverify ps
# Tutaj dla przykładu użycie polecenia ps w Dockerze
```

W celu połączenia się z Dockerem i sprawdzenie np. pliku json, należy wydać polecenie poniżej

```
curl https://$HOST:2376/images/json \
--cert ~/.docker/cert.pem \
--key ~/.docker/key.pem \
--cacert ~/.docker/ca.pem
```

Korzystanie z Dockera

Teraz możemy włączyć dockera z system Ubuntu wydając poniższe polecenie

```
docker run -it ubuntu

# Zapiszmy teraz plik testowy z przykładową treścią
echo "Example1" > /tmp/Example1
# Wyjdźmy z Dockera
exit

# Sprawdźmy czy Docker jest zatrzymany czy też w ogóle nie działa
docker ps -a

# Teraz jeżeli jeszcze raz uruchomimy dockera zobaczymy, że identyfikator jest inny co wskazuje na stworzenie zupełnie nowego kontenera
docker run -it ubuntu

# Jeżeli spróbujemy odczytać teraz plik, który wcześniej stworzyliśmy okaże się, że nie istnieje ze względu na to, że stworzyliśmy całkiem nową instancję dockera
cat /tmp/Example1

# Jednakże poprzednia instalacja istnieje w tym celu wystarczy wywołać odpowiednie polecenie z id kontenera
docker start -ai id_cotainer

# Następnie wydajmy polecenie
cat /tmp/Example1
# Okazuje się, że zmiany w kontenerze są zachowywane
```

Jeżeli chcemy usunąć kontener Dockera wystarczy jako argument podać jego nazwę lub id

```
docker rm nazwa
docker rm id
```

Przydatne linki

Wszelkie informacje na temat bezpieczeństwa Dockera wraz z tutorialami jak krok po kroku zabezpieczyć kontenery znajdziesz na stronie: [Docker Security]