Root felhasználóra váltás

A sikeres putty csatlakozás után az alábbi látjuk:

```
kovi91@docker:~$
```

Ekkor egy normál jogosultságú felhasználóval vagyunk belépve. Minden admin jogot kívánó parancsot sudoval kellene kezdeni és mindig be kéne írni a jelszót. Hogyha erre lusták vagyunk, akkor váltsunk át az admin userre, a sudo su paranccsal.

```
kovi91@docker:~$ sudo su
```

Menjünk el a root felhasználó saját mappájába (/root) a cd paranncsal.

```
root@docker:/home/kovi91# cd
```

Ekkor a terminál erre vált:

```
root@docker:~#
```

Docker telepítése

```
apt-get update

apt-get install ca-certificates curl gnupg

install -m 0755 -d /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/debian/gpg | gpg --
dearmor -o /etc/apt/keyrings/docker.gpg

chmod a+r /etc/apt/keyrings/docker.gpg

echo "deb [arch="$(dpkg --print-architecture)" signed-
by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/debian "$(. /etc/os-release &&
echo "$VERSION_CODENAME")" stable" | tee
/etc/apt/sources.list.d/docker.list > /dev/null

apt-get update
```

```
apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

A felfele nyíl visszahozza az előző parancsot, a clear parancs törli a képernyőt.

Teszt konténer indítása

```
docker run -d --rm --name web-test -p 80:8000 crccheck/hello-world
```

Kapcsolók jelentése:

- run: futtasd
- -d: a háttérben
- --rm: ha leállítottam, akkor töröld is ki
- --name: ez legyen a neve
- -p: a fizikai gép 80-as portját rákötöm a konténer 8000-es portjára
- crccheck/hello-world: a docker hubon a crccheck user hello-world nevű image-ét akarom letölteni, abból is a latest verziót

Hasznos docker parancsok

- docker stats: Konténerek realtime cpu és memória foglalása (ctrl + c a kilépés)
- docker container ls: aktív konténerek listázása
- docker container ls -a: inaktív konténerek is legyenek listázva
- docker logs <konténernév>: logok

Apache2

```
docker run -d --rm --name web -p 80:80 httpd:2.4
```

Csináljunk egy /root/www mappát benne index.html-el

```
docker run -d --rm --name web -p 80:80 -v /root/www/:/usr/local/apache2/htdocs/ httpd:2.4
```

Honnan tudtam, hogy melyik mappába kell bekötni a www-t?

```
docker exec -it web /bin/bash
exit
```

Debian indítás

```
docker run -d -p 80:80 --name deb debian
```

De hát ez leállt bazeg…persze mert nem volt dolga. dit miatt elindul a bash parancs.

```
docker run -dit -p 80:80 --name deb debian
```

Apache2 szervert telepítünk

```
apt-get update
apt-get install apache2
/etc/init.d/apache2 start
```

Na ennek semmi értelme, hogy létrehozzuk kvázi üresen, aztán kézzel beletelepítünk mindent. Ekkor kell image-t készíteni, de ezt majd később.

MySQL + Wordpress + Phpmyadmin

Mysql

```
docker container run --name db -e MYSQL_ROOT_PASSWORD=titok -e
MYSQL_DATABASE=db1 -e MYSQL_USER=devops -e
MYSQL_PASSWORD=nemtudom -d percona:5
```

Phpmyadmin

```
docker run --name phpmyadmin -d --link db:db -p 80:80
nibrev/phpmyadmin-4.0.x
docker stop phpmyadmin
docker rm phpmyadmin
```

Wordpress

```
docker container run --restart=always -p 80:80 --link db:db --
name wordpress -d wordpress:6.0.0-apache
```

Reverse Proxy

Hoston telepítjük!

```
apt-get update
apt-get install apache2
systemctl start apache2.service
```

```
nano /etc/apache2/sites-available/phpmyadmin.conf
```

```
Define domain wordpress.kovacsandras.bprof.hu
Define port 8002

<VirtualHost *:80>
        ServerName ${domain}
        ServerAlias ${domain}
        RequestHeader set "X-Forwarded-Proto"
expr=%{REQUEST_SCHEME}
        Protocols               h2 http/1.1
        ProxyPreserveHost       On
        ProxyPass               / http://localhost:${port}/
        ProxyPassReverse        / http://localhost:${port}/

        ErrorLog ${APACHE_LOG_DIR}_${domain}-error.log
        CustomLog ${APACHE_LOG_DIR}_${domain}-access.log common
</VirtualHost>
```

ctrl + o és ctrl + x

```
a2ensite phpmyadmin
systemctl reload apache2
```

Hopsz, telepíteni kell némi apache modult!

```
a2enmod headers
a2enmod proxy
a2enmod rewrite
a2enmod proxy_http
a2enmod ssl
```

Certificate

```
apt-get install certbot
```

Majd leállítjuk az apache-ot, hogy tudjon magának elindítani egy kis webszervert, hogy megtörténjen a challenge

```
systemctl stop apache2.service
```

```
certbot certonly --preferred-challenges http -d
phpmyadmin.kovacsandras.bprof.hu --agree-tos --manual-public-ip-
logging-ok --register-unsafely-without-email --rsa-key-size 4096
```

(ugyanez wordpressre is)

```
systemctl start apache2.service
```

Módosítjuk az apache configot

```
Define domain wordpress.kovacsandras.bprof.hu
Define port 8002

<VirtualHost *:80>
        ServerName ${domain}
        ServerAlias ${domain}
        RequestHeader set "X-Forwarded-Proto"
expr=%{REQUEST_SCHEME}
        Protocols             h2 http/1.1
        ProxyPreserveHost     On
        ProxyPass             / http://localhost:${port}/
        ProxyPassReverse      / http://localhost:${port}/

        RewriteEngine On
        RewriteCond %{HTTPS} off
        RewriteRule ^ https://%{HTTP_HOST}%{REQUEST_URI}
                ErrorLog ${APACHE_LOG_DIR}_${domain}-error.log
                CustomLog ${APACHE_LOG_DIR}_${domain}-access.log
common
</VirtualHost>
<VirtualHost *:443>
    Protocols             h2 http/1.1
    ProxyPreserveHost     On
    ProxyPass             / http://127.0.0.1:${port}/
    ProxyPassReverse      / http://127.0.0.1:${port}/
    ServerName ${domain}
    ServerAlias ${domain}
```

```
    ErrorLog ${APACHE_LOG_DIR}_${domain}-error.log
    CustomLog ${APACHE_LOG_DIR}_${domain}-access.log common
    SSLEngine             on
    SSLProtocol           all -SSLv3 -TLSv1 -TLSv1.1
    SSLHonorCipherOrder   off
    SSLCompression        off
    SSLSessionTickets     on
    SSLUseStapling        off
    SSLCipherSuite    ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-
AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECD$
        SSLCertificateFile
/etc/letsencrypt/live/${domain}/fullchain.pem
        SSLCertificateKeyFile
/etc/letsencrypt/live/${domain}/privkey.pem
</VirtualHost>
```