Ako rozbehať dotnetcore projekt na unixe

Predpoklady:

- Nainštalovaný a nakonfigurovaný unix server.
- Mať prístup na unix pod rootom.
- Mať prístup ku kódom projektu, ktorý idete nasadiť na unix.

Dodatočné predpoklady v prípade projektu TRACKS:

- Nainštalovaný, nakonfigurovaný a z vonku prístupný MSSQL server aj s databázou.
- Vytvorený korektný súbor user.json a mať ho na unixe.

Info:

- V čase, keď sme nasadzovali náš projekt na unix, neexistovalo žiadne ľahké riešenie ako toto vykonať. Momentálne už je dotnetcore podporovaný na unixe napríklad pomocou technológie Mono, je dosť možné, že implementácia pomocou technológie Mono bude oveľa jednoduchšia ako nasledovný postup.
- Postup je vytvorený presne pre unix „Ubuntu 16.04.1 LTS“ a pre iné verzie sa postup môže líšiť.
- Vychádzali sme z nasledujúceho tutoriálu, ktorý umožňuje spustenie dotnetcore projektu na lokálnom porte na unixe a následne vytvára tunel tak aby bol tento projekt dostupný na verejnej siete. http://www.hanselman.com/blog/PublishingAnASPNETCoreWebsiteToACheapLinuxVMHost.aspx
- Nenesieme žiadnu zodpovednosť za prípadné škody.

Inštalácia dotnetcore SDK:

1.) Dotnetcore SDK púšťa váš projekt.
2.) Pripojte sa na unix pod rootom.
3.) Nasledovné príkazy vám sprístupnia balíky, ktoré potrebujete pre inštaláciu dotnetcore SDK. (Príkazy sú špecifické pre každú verziu unixu.)

```
sh -c 'echo "deb [arch=amd64] https://apt-mo.trafficmanager.net/repos/dotnet-release/ trusty main" > /etc/apt/sources.list.d/dotnetdev.list'

apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys 417A0893

apt-get update
```

4.) Nainštalujte dotnetcore SDK.

```
apt-get install dotnet-dev-1.0.3
```

5.) Vyskúšajte či ste dotnetcore SDK nainštalovali správne pomocou vytvorenia jednoduchého projektu.

```
mkdir testapp
cd testapp
service dotnet new
service dotnet restore
service dotnet run
```

6.) V prípade, že predchádzajúce príkazy prešli bez problému, môžete pokračovať ďalej. Inak opakujte postup uvedený vyššie. V prípade, že ste nainštalovali zlú verziu dotnetcore SDK alebo ste ho nainštalovali zle, tak ho pred novou inštaláciou treba riadne odinštalovať.

Vytvorenie a spustenie dotnetcore web app (projekt, ktorý má identické požiadavky na spustenie ako naša TRACKS aplikácia):

1.) Príkaz vytvorí dotnetcore web app.

```
service dotnet new -t web
```

2.) Projekt je však zložitejší a na spustenie potrebuje doinštalovať nasledovné balíky.

```
apt-get install npm
npm install gulp
npm install bower
```

3.) Spustenie projektu.

```
service dotnet restore
service dotnet run
```

4.) Projekt je jednoduchá webová aplikácia. Ak všetko prešlo tak sa spustila na defaultnej adrese localhost:5000.

5.) Pre jednoduché overenie presmerujeme túto adresu k nám na lokálny port. Nasledujúci príkaz vykonajte v novom shellovom okne.

```
ssh -L 2000:localhost:5000 POUZIVATEL@ADRESA_UNIXU
```

6.) Otvorte si Váš obľúbený webový prehliadač a zadajte adresu http://localhost:2000. Mali by ste vidieť ukážkový projekt, ktorý sme predtým vygenerovali na unixe.

Úprava vstupného projektu tak aby ho bolo možné uverejniť na verejnej adrese:

1.) Otvorte váš projekt napríklad vo Visual Studiu.

2.) Otvorte súbor Program.cs

3.) Zmeňte implementáciu metódy main na nasledovný tvar.

```
var host = new WebHostBuilder()
                .UseKestrel()
                .UseUrls("http://*:9999")
                .UseContentRoot(Directory.GetCurrentDirectory())
                .UseIISIntegration()
                .UseStartup<Startup>()
```

```
                    .Build();

        host.Run();
```

4.) Týmto nastavením hovoríte aby sa vaša aplikácia púšťala na porte 9999. Je potrebné aby bol daný port voľný, v prípade obsadeného portu 9999 zvoľte iný port. V rámci tutoriálu budeme používať port 9999.

Inštalácia a úprava nastavení technológie nginx:

1.) Nginx nám presmeruváva komunikáciu na verejných portoch na porty lokálne. Nainštalujte ho pomocou nasledovného príkazu.

```
apt-get install nginx
```

2.) Choďte do jeho nastavení.

```
cd /etc/nginx/sites-available
```

3.) Otvorte súbor default v textovom editore, napríklad vo Vim alebo Nano.

4.) Zmeňte jeho nastavenie na nasledovný tvar.

```
listen 8008; #default_server;
server_name ~^(.+)$;
location  / {
    proxy_pass http://localhost:9999;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection keep-alive;
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;
    #try_files $uri $uri/ =404;
}
```

5.) Týmto nastavením ste povolili pripojenia na verejnom porte 8008 a všetku komunikáciu presmerúvate na lokálny port 9999.

6.) Reštartuje ho, aby sa načítali nové nastavenia.

```
service nginx -t
service nginx -s reload
```

Inštalácia a úprava nastavení technológie supervisor:

1.) Supervisor zodpovedá za to, že naša aplikácia bude stále dostupná. Nainštalujte ho pomocou nasledovného príkazu.

```
apt-get install supervisor
```

2.) Choďte do jeho nastavení

```
cd /etc/supervisor/conf.d
```

3.) Vytvorte nový súbor a pomenujte ho TRACKS.conf. Otvorte tento súbor v textovom editore, napríklad vo Vim alebo Nano.

4.) Zmeňte jeho nastavenie na nasledovný tvar.

```
[program:TRACKS]
```

```
command=/usr/bin/dotnet /var/TRACKS/Tracks.Web.dll --
server.urls:http://*:9999
directory=/var/TRACKS/
autostart=true
autorestart=true
stderr_logfile=/var/log/TRACKS.err.log
stdout_logfile=/var/log/TRACKS.out.log
environment=ASPNETCORE_ENVIRONMENT=Production
user=www-data
stopsignal=INT
```

5.) Supervisor bude vašu aplikáciu udržiavať v chode.
6.) Reštartujte supervisor.

```
service supervisor stop
service supervisor start
```

Deploy aplikácie:

1.) Publishnite vašu aplikáciu napríklad pomocou Visual Studia, TFS alebo inej platformy.
2.) Publishnuté súbory nahrajte na unix pomocou SSH alebo FTP.
3.) Súbory prekopírujte sem.

```
/var/TRACKS
```

4.) Kvôli nastaveniu supervisora je nutné aby publishnutá verzia obsahovala súbor Tracks.Web.dll.

Finalizácia všeobecného projektu:

1.) Nasledovné príkazy vykonajte v samostatných shelloch. Tieto shelly budú monitorovať logy jednotlivých technológií.

```
tail -f /var/log/nginx/error.log
tail -f /var/log/nginx/access.log
tail -f /var/log/supervisor/supervisord.log
```

2.) Uistite sa, že vami vybrané porty sa nikde nepoužívajú. V našom prípade to sú porty 8008 a 9999.
3.) Vytvorte vo firewally výnimku pre TCP komunikáciu na týchto portoch. Úpravu firewallu môžete vykonať napríklad pomocou iptables.
4.) Vykonajte nasledovné príkazy.

```
/etc/init.d/nginx stop
/etc/init.d/supervisor stop
/etc/init.d/nginx start
/etc/init.d/supervisor start
```

5.) Aplikácia by teraz mala byť prístupná na adrese:

http://adreseUnixu:8008

Dodatočné úpravy v prípade projektu TRACKS:

1.) Pred vykonaním kroku 4 v predchádzajúcej časti treba ešte nakopírovať súbor user.json na túto adresu.
   /var/TRACKS
2.) Prvé spustenie projektu môže trvať trochu dlhšie z dôvodu generovania databázy. Prípadné chyby sú v logoch v nasledujúcej zložke.
   /var/TRACKS/Logs

Dodatočné informácie v prípade projektu TRACKS:

1.) Kvôli konzistencii dát v database je nutné uchovávať súbory v týchto zložkách skrz jednotlivé verzie projektu.
   /var/TRACKS/JobExecutables
   /var/TRACKS/ClientLogs
2.) O takéto uchovávanie sa stará skript „script_dotnet2.sh“, ktorý používame v rámci CI. Jeho forma je uverejnená v prílohe 1 na konci tohto tutoriálu. Skript sa napríklad staral aj o kopírovanie súboru user.json.
3.) Daný skript je nutné púšťať automatizovane pod používateľom root. K tomu používame skript „starting_script.sh“ uverejnený v prílohe 2 na konci tohto tutoriálu. Tento skript bol volaný automaticky po deployi z TFS.

Príloha 1 - script_dotnet2.sh

```
#!/bin/bash

echo "**************"
echo -n "Starting bootnet script "
echo `date +"%d_%h_%T"`

echo "***************"

DATUM=$(date +%d-%b_%T)

cp -a /var/TRACKS/JobExecutables /home/team08/TracksCopyFiles/

cp -a /var/TRACKS/Logs/* /home/team08/TRACKS_log_backup/

cp -a /var/TRACKS/ClientLogs /home/team08/TracksCopyFiles/

rm -rfd /var/TRACKS/*  &> /home/team08/log/release/$DATUM.log

echo "Remove old files from /var/TRACKS/ succesful"

cp -a /home/team08/apache-ftpserver-1.1.0/res/home/TRACKS/* /var/TRACKS/ &>>
/home/team08/log/release/$DATUM.log

echo "Copy of TRACKS files succesful"

cp -a /home/team08/TracksCopyFiles/* /var/TRACKS/ &>>
/home/team08/log/release/$DATUM.log

echo "Copy of TracksCopyFiles succesful"

rm -rfd /home/team08/apache-ftpserver-1.1.0/res/home/TRACKS/* &>>
/home/team08/log/release/$DATUM.log

rm -rfd /home/team08/TracksCopyFiles/JobExecutables
rm -rfd /home/team08/TracksCopyFiles/ClientLogs

echo "Remove old files from FTP succesful"

echo "Services will be restarted"

/etc/init.d/nginx stop  &>> /home/team08/log/release/$DATUM.log
/etc/init.d/supervisor stop &>> /home/team08/log/release/$DATUM.log

/etc/init.d/nginx start
/etc/init.d/supervisor start
```

```
echo "Services started"

/etc/init.d/nginx status &>> /home/team08/log/release/$DATUM.log
/etc/init.d/supervisor status &>> /home/team08/log/release/$DATUM.log

echo "*****************"
echo "End bootnet script"
echo "*****************"
```

Príloha 2 - starting_script.sh

```
#!/bin/bash

if [ "$(whoami)" != "root" ]
then
   echo "Starting script is starting"
   echo -e "HESLO_K_ROOTOVI" | sudo -S -H -u root ./script_dotnet2.sh

   exit
fi
echo "not started"
```