# Практика: DOB Модул 7

## Работа в Azure портала

#### Създаване на виртуална машина

Отваряме нова сесия в браузъра към адрес: <https://portal.azure.com>

Ако имаме потребителско име и парола, ги въвеждаме. В противен случай, можем да се регистрираме като нови потребители. Така ще можем да се възползваме от едномесечен тестов период.

При успешно влизане в портала, по подразбиране се визуализира т.нар. управленско табло (**dashboard**).

Тук имаме възможност да се възползваме от секцията **Quickstart tutorials**, за да направим своите първи стъпки в платформата. Този първоначален екран подлежи на промяна, можем да го нагласим така, както ще бъде най-удобно за нас.

Вляво се намира основното меню. От него имаме бърз достъп до специфичните функционалности на платформата. По време на днешното занятие, за нас интерес ще представлява основно работата с **Linux** виртуални машини. Това се свежда в общия случай до взаимодействие със секциите **Resource groups**, **Virtual** machines и **Virtual networks**.

Нека да направим първата си **Linux** виртуална машина в **Azure** платформата.

От менюто вляво избираме **Virtual machines**

Кликваме върху бутона **Add**

Нека да започнем с чиста инсталация на операционна система, без допълнения. Можем в полето за търсене (**Search Compute**) да въведем критерий за търсене. Нека въведем **ubuntu**

Би трябвало първите няколко резултата да включват само предложения от **Canonical**. Нека да изберем **Ubuntu Server 17.10**

Вдясно се отваря нова секция, в която освен описанието на избрания компонент, в долния край се намира и поле за избор на метода за управление на ресурсите на машината. Препоръчително е за нови разработки да се използва т.нар. **Resource Manager**. Уверяваме се, че именно тази опция е маркирана и потвърждаваме с бутона **Create**

Сега вече се намираме в етап на задаване на параметрите на бъдещата виртуална машина

Нека първо да зададем име **dob-vm-portal-1**

Като тип на диска можем да оставим опцията по подразбиране, а може и да я сменим на **HDD**. Не би следвало да има значение за целите на нашето упражнение

Като потребителско име можем да зададем **dobuser**

Нека да сменим **Authentication type** на **Password** и като парола да въведем **D3v0ps-Basics**

От менюто **Subscriptions** трябва да изберем нашия **Azure** план, към който искаме да прикачим новата машина

На следващата стъпка трябва да изберем дали ще създадем нова група от ресурси или ще включим машината към вече съществуваща група. Оставяме маркирана опцията **Create new** и като име въвеждаме **dob-rg-portal-1**

В полето **Location** следва да изберем най-близко разположения **Azure** център. Всъщност критерия за избор може да бъде наложен и от други съображения. Ние ще изберем **West Europe**

Потвърждаваме с бутона **OK** и преминаваме към втората стъпка в дефинирането на нашата машина - избора на шаблон от предефинирани характеристики

Можем да избере някой от предложените шаблони, примерно **A1 Standard**. Бихме могли също така да кликнем върху **View all**, за да видим всички налични шаблони и да изберем друг - примерно **B1s Standard**. Валиден вариант е и от трите параметъра в горната част на екрана да зададем минимални изисквания към шаблоните, за да отсеем предварително списъка от възможности. Избираме **B1s Standard**

На следващата стъпка се появява предупреждение, че първоначално сме избрали стандартен тип диск, а избрания шаблон ни позволява да използваме **SSD**. Можем да коригираме избора на диск, като от секцията **Disk type** изберем **SSD**

Всички други настройки можем да оставим по подразбиране и да потвърдим с бутона **OK**

Като последна стъпка, можем да прегледаме обобщение на направените настройки и избраните параметри на новата машина. Трябва да приемем съответното споразумение и да потвърдим с бутона **Create**

Междувременно, докато трае създаването на заявената машина, можем да разгледаме някои от секциите на портала

След приключване на процеса по създаване на машината, ще бъдем отведени до конзолата ѝ за наблюдение и управление. Ако това не се случи, то можем от менюто вдясно да изберем опцията **Virtual machines**. След това от списъка с машини трябва да изберем тази, от която се интересуваме. По подразбиране сме в режим **Overview**

Тук можем да получим информация за параметрите на машината, начина на достъп до нея и текущото ѝ натоварване. Бихме могли също така да я пуснем, спрем, рестартираме и изтрием. Не на последно място бихме могли да променим и нейните параметри

Нека на този етап от лентата с бутони в горния край на прозореца, да изберем бутона **Connect**. Това ще доведе до визуализация на наличния метод за установяване на връзка към машината. Примерно **ssh dobuser@40.114.144.81**

Копираме текста и го поставяме в нов терминален прозорец и потвърждаваме с клавиша **Enter**

След потвърждение за добавяне на информация за машината и въвеждане на указаната по време на създаването парола, сме в машината

Тук можем да извършваме всички онези действия, които правим и с локално вдигнатите **Linux** виртуални машини

Можем да разгледаме и да изпълним някоя команда, за да се ориентираме в обстановката

Сега вече можем да пристъпим към инсталацията примерно на уеб сървър. Нека да инсталираме **nginx**. Това бихме могли да постигнем с командата: **sudo apt-get update && sudo apt-get install -y nginx**

След приключването на инсталацията можем да проверим дали **nginx** работи: **systemctl status nginx**

Нека сега да отворим нова сесия в браузъра към адреса, към който осъществихме **ssh** връзка по-рано:

<http://40.114.144.81>

Оказва се, че не можем да достигнем работещия там уеб сървър. Необходимо е да предприемем допълнителни действия

За целта се връщаме в конзолата за управление на виртуалната машина и избираме опцията **Networking**

Там установяваме, че има дефинирани правила за управление на входящите и изходящите връзки към и от машината

Нека да кликнем на бутона **Add inbound port rule**, за да разрешим достъпа до нашата машина по порт **80**

Появява се нов прозорец вдясно. В него можем да дефинираме потребителско правило, това е режима по подразбиране. В случая, тъй като искаме да разрешим стандартна услуга, можем от менюто **Service** да изберем съответното предефинирано правило - **HTTP** и да потвърдим с бутона **OK**

Ако сега се върнем в отворената по-рано сесия в браузъра и направим повторен опит, то би трябвало да видим съобщението по подразбиране за нова инсталация на **nginx**

Можем да подменим страницата по подразбиране. За целта можем да копираме двата файла, които се намират в папката **DOB Azure/p1** в папка **/var/www/html** на виртуалната машина с помощта на командата **scp**

Нека сега да опресним отворената по-рано сесия в браузъра

Сега за целите на практиката, бихме могли да добавим втора машина, която да е част от тази ресурсна група или да е в самостоятелна такава

#### Инсталиране на добавки

Освен да използваме чисти варианти на операционна система, можем да работим и с предефинирани пакети от операционна система и специализиран софтуер. Тъй като вече имаме работеща машина, нека да добавим чрез шаблон допълнителен софтуер към нея

От портала за управление на машината можем да изберем опцията **Extensions**

Избираме бутона **Add**

Можем да прегледаме списъка с добавки, който се появи вдясно. Нека от него да изберем добавката **Docker**

След като се запознаем с описанието ѝ, можем да кликнем върху бутона **Create**

Оказва се обаче, че са ни необходими сертификати. Нека се върнем в **ssh** сесията към нашата машина и да изпълним последователно следните команди:

**openssl genrsa -aes256 -out ca-key.pem 4096**

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

**openssl genrsa -out server-key.pem 4096**

**openssl req -subj "/CN=dob-vm-portal-1" -sha256 -new -key server-key.pem -out server.csr**

**echo subjectAltName = DNS:dob-vm-portal-1,IP:10.1.0.4,IP:127.0.0.1 >> extfile.cnf**

**echo extendedKeyUsage = serverAuth >> extfile.cnf**

**openssl x509 -req -days 365 -sha256 -in server.csr -CA ca.pem -CAkey ca-key.pem \**

**-CAcreateserial -out server-cert.pem -extfile extfile.cnf**

**openssl genrsa -out key.pem 4096**

**openssl req -subj '/CN=dob-vm-portal-1' -new -key key.pem -out client.csr**

**echo extendedKeyUsage = clientAuth >> extfile.cnf**

**openssl x509 -req -days 365 -sha256 -in client.csr -CA ca.pem -CAkey ca-key.pem \**

**-CAcreateserial -out cert.pem -extfile extfile.cnf**

**chmod -v 0400 ca-key.pem key.pem server-key.pem**

**chmod -v 0444 ca.pem server-cert.pem cert.pem**

С това процедурата по **docker** документацията (<https://docs.docker.com/engine/security/https/#create-a-ca-server-and-client-keys-with-openssl>) е изпълнена. Сега трябва да извършим още няколко действия, наложени от изискванията (<https://docs.microsoft.com/en-us/azure/virtual-machines/linux/classic/portal-use-docker#add-the-docker-vm-extension>) на самата добавка:

**base64 ca.pem > ca64.pem**

**base64 server-cert.pem > server-cert64.pem**

**base64 server-key.pem > server-key64.pem**

След като вече разполагаме с комплекта от файлове, можем да ги изтеглим на нашата станция:

**scp dobuser@40.114.144.81:\*64.pem .**

Сега се връщаме в прозореца за конфигурация на добавката и указваме трите файла, които изтеглихме

Потвърждаваме с бутона **Create**

След успешната инсталация на добавката, можем да се върнем в нашата **ssh** сесия и да изпълним:

**docker version**

Сега можем да стартираме и един тестови контейнер:

**docker container run shekeriev/welcome-dob-oct-17**

## Azure CLI

#### Инсталиране на конзолата

Имаме възможност да работим с **Azure** и чрез команден ред. Това ни дава възможност да автоматизираме повечето процеси

Има различни подходи за инсталиране на **Azure CLI** под **Linux**. Ние ще изберем варианта с добавяне на допълнително хранилище на пакети. За целта трябва да изпълним следния набор от стъпки

Първо трябва да добавим съответния ключ:

**sudo rpm --import** [**https://packages.microsoft.com/keys/microsoft.asc**](https://packages.microsoft.com/keys/microsoft.asc)

След това трябва да добавим и самото хранилище - за **CentOS** командата е следната:

**sudo sh -c 'echo -e "[azure-cli]\nname=Azure CLI\nbaseurl=https://packages.microsoft.com/yumrepos/azure-cli\nenabled=1\ngpgcheck=1\ngpgkey=https://packages.microsoft.com/keys/microsoft.asc" > /etc/yum.repos.d/azure-cli.repo'**

За **openSUSE** нещата изглеждат така:

**sudo sh -c 'echo -e "[azure-cli]\nname=Azure CLI\nbaseurl=https://packages.microsoft.com/yumrepos/azure-cli\nenabled=1\ntype=rpm-md\ngpgcheck=1\ngpgkey=https://packages.microsoft.com/keys/microsoft.asc" > /etc/zypp/repos.d/azure-cli.repo'**

След това пристъпваме към същинската инсталация - при **CentOS**:

**yum check-update**

**sudo yum install azure-cli**

И съответно при **openSUSE**:

**sudo zypper refresh**

**sudo zypper install azure-cli**

Всички команди за управление на **Azure** започват с **az**. Ако изпълним само

**az**

Ще видим списък от поддържаните под-команди

Нека да се удостоверим пред **Azure**:

**az login**

Излиза ни надпис, който ни подканя да отворим браузър сесия към адрес <https://aka.ms/devicelogin> и да въведем предоставения код - например **CAM9VBQ37**

След като въведем кода, трябва да потвърдим с бутона **Continue**

После трябва да довършим процеса на удостоверяване. Когато всичко приключи успешно, ще можем да затворим съответния прозорец на браузъра

В конзолата ще видим информация за успешното удостоверяване под формата на **json**

#### Изпълнение на команди

Можем да видим списък с наличните машини

**az vm list**

Можем по сходен начин да видим и списък с ресурсните групи

**az group list**

Всички команди имат налична помощна информация, която се извиква с **-h**

#### Дефиниране на машина

Нека да създадем една виртуална машина през този интерфейс

Първо трябва да създадем ресурсна група. Това можем да направим с командата:

**az group create -n dob-rg-cli-1 -l westeurope**

След като имаме вече дефинирана ресурсна група, можем да подадем заявка за нова машина. Това можем да направим с команда подобна на тази:

**az vm create -n dob-vm-cli-1 -g dob-rg-cli-1 --image UbuntuLTS --data-disk-sizes-gb 10 --generate-ssh-keys**

След успешното изпълнение на заявката, можем да видим от отговора кой е публичния адрес, на който можем да осъществим **ssh** връзка. Примерно:

**ssh 52.166.39.97**

Тук отново можем да повторим по-ранните стъпки - да инсталираме уеб сървър, да копираме файлове и да отворим порт

#### Повторение на изпълнена инсталация

Можем да получим списък от извършените създавания на машини в ресурсна група:

**az group deployment list --resource-group dob-rg-cli-1**

Сега можем да експортираме съответния избрано създаване:

**az group deployment export --name vm\_deploy\_sUgdvZW9hG7pE4CZlhLABF84G5whWuh3 --resource-group dob-rg-cli-1 >> export.json**

Макар и не съвсем според предписанията в официалната документация, можем да отворим файла и да нанесем съответните корекции, т.е. да дефинираме нови ресурси:

**vi export.json**

След като приключим с редактирането, запазваме файла и излизаме от редактора

Сега бихме могли да създадем нова машина в същата ресурсна група като изпълним:

**az group deployment create --name new-dep-1 --resource-group dob-rg-cli-1 --template-file export.json**

След успешното приключване на процеса, можем от портала за управление да получим публичния адрес и да направим **ssh** сесия:

**ssh** [**dimitar@13.95.109.79**](mailto:dimitar@13.95.109.79)

Разбира се в документацията на Azure има специални преписания как да параметризираме нашите шаблони, за да постигнем желаното ниво на автоматизация. Повече информация може да бъде намерена тук:

<https://docs.microsoft.com/en-us/azure/azure-resource-manager/resource-group-authoring-templates#parameters>

#### Работа с добавки

Използвайки информацията, предоставена в документацията, можем да изпълним следната последователност от команди, за да направим малко по-комплексна машина

Първо създаваме ресурсна група

**az group create --name dob-rg-cli-3 --location westeurope**

След това дефинираме нова машина

**az vm create --resource-group dob-rg-cli-3 --name dob-vm-cli-3 --image UbuntuLTS --generate-ssh-keys**

Нека да дефинираме и пренасочване на порт

**az vm open-port --port 80 --resource-group dob-rg-cli-3 --name dob-vm-cli-3**

И най-накря можем да подадем инструкции към добавката за обработка на скриптове така, че да инсталира **NGINX**

**az vm extension set \**

**--publisher Microsoft.Azure.Extensions \**

**--version 2.0 \**

**--name CustomScript \**

**--vm-name dob-vm-cli-3 \**

**--resource-group dob-rg-cli-3 \**

**--settings '{"commandToExecute":"apt-get -y update && apt-get -y install nginx"}'**

Можем да отворим браузърна сесия към съответния публичен адрес, примерно <http://52.178.3.175>

Разбира се същата тази поредица, може да бъде съхранена в скрипт и извиквана като едно цяло

Накрая можем да почистим, като изтрием цялата ресурсна група

**az group delete --name dob-rg-cli-3**

## Vagrant и Azure

Можем да включим **Vagrant** в процеса на работа с виртуални машини в **Azure** платформата

Необходимо е да извършим набор от подготвителни действия, които включват изтегляне на специален **box** и инсталация на плъгин

Нека първо да генерираме ключ, в случай че нямаме

**ssh-keygen -t rsa -b 2048**

Първо изтегляме **box**-а:

**vagrant box add azure https://github.com/azure/vagrant-azure/raw/v2.0/dummy.box --provider azure**

След това инсталираме и добавката:

**vagrant plugin install vagrant-azure**

Преди да можем да създаваме машини с помощта на Vagrant, трябва да създадем специален запис в активната директория на Azure:

**az ad sp create-for-rbac -n "dob-azure-demo-20171213"**

Информацията, получена след успешното приключване на операцията, можем да съхраним във файл, който в последствие да използваме за установяване на средата. Примерното му съдържание е:

**#!/bin/bash**

**# Tenant ID (tenant)**

**export AZURE\_TENANT\_ID='ff0afbb9-adc6-43bf-a6c4-09031ee951cb'**

**# Application ID (appId)**

**export AZURE\_CLIENT\_ID='57d50e53-476d-4c65-b288-dc1dab48579f'**

**# Password (password)**

**export AZURE\_CLIENT\_SECRET='42e2c8ef-a22d-4e67-ad44-46f12a542d33'**

**# Subscription**

**export AZURE\_SUBSCRIPTION\_ID='832a1231-36bc-4755-a969-486375fccae0'**

Всъщност в горния файл трябва да добавим и номера на нашия абонамент. За целта трябва да изпълним командата:

**az account list --query "[?isDefault].id" -o tsv**

Сега вече можем да направим първата си машина чрез **Vagrant**. Създаваме папка и в нея нов **Vagrantfile** със следното съдържание:

**Vagrant.configure('2') do |config|**

**config.vm.box = 'azure'**

**# use local ssh key to connect to remote vagrant box**

**config.ssh.private\_key\_path = '~/.ssh/id\_rsa'**

**config.vm.provider :azure do |azure, override|**

**# each of the below values will default to use the env vars named as below if not specified explicitly**

**azure.tenant\_id = ENV['AZURE\_TENANT\_ID']**

**azure.client\_id = ENV['AZURE\_CLIENT\_ID']**

**azure.client\_secret = ENV['AZURE\_CLIENT\_SECRET']**

**azure.subscription\_id = ENV['AZURE\_SUBSCRIPTION\_ID']**

**end**

**end**

Първо трябва да **source**-нем създадения по-рано файл, за да модифицираме средата

След това изпълняваме

**time vagrant up**

И после влизаме по познатия начин

**vagrant ssh**

След като приключим с разглеждането, можем да излезем и да унищожим машината

**exit**

**vagrant destroy**

Ако погледнем в портала, ще установим, че машината не изчезва мигновено. Необходимо е известно време тя да бъде изтрита и да бъдат освободени заетите от нея ресурси

Можем в нова папка да създадем нов **Vagrantfile** или да модифицираме текущия до:

**Vagrant.configure('2') do |config|**

**config.vm.box = 'azure'**

**# use local ssh key to connect to remote vagrant box**

**config.ssh.private\_key\_path = '~/.ssh/id\_rsa'**

**config.vm.provider :azure do |azure, override|**

**# each of the below values will default to use the env vars named as below if not specified explicitly**

**azure.tenant\_id = ENV['AZURE\_TENANT\_ID']**

**azure.client\_id = ENV['AZURE\_CLIENT\_ID']**

**azure.client\_secret = ENV['AZURE\_CLIENT\_SECRET']**

**azure.subscription\_id = ENV['AZURE\_SUBSCRIPTION\_ID']**

**# Specify VM parameters**

**azure.vm\_name = 'dob-azure'**

**azure.vm\_size = 'Standard\_B1s'**

**azure.vm\_image\_urn = 'Canonical:UbuntuServer:16.04-LTS:latest'**

**azure.resource\_group\_name = 'dob-vagrant'**

**azure.location = 'westeurope'**

**azure.tcp\_endpoints = '80'**

**end**

**config.vm.provision "shell", inline: "sudo apt-get update && sudo apt-get install -y nginx"**

**end**

След това изпълняваме

**time vagrant up**

И после влизаме по познатия начин

**vagrant ssh**

След като приключим с разглеждането, можем да излезем и да унищожим машината

**exit**

**vagrant destroy**

Нека да направим нова папка, в която да пробваме комбинация от **Vagrant**, **Azure** и **Ansible**

Създаваме нов **Vagrantfile** със следното съдържание:

**Vagrant.configure('2') do |config|**

**config.vm.box = 'azure'**

**# use local ssh key to connect to remote vagrant box**

**config.ssh.private\_key\_path = '~/.ssh/id\_rsa'**

**config.vm.provider :azure do |azure, override|**

**# each of the below values will default to use the env vars named as below if not specified explicitly**

**azure.tenant\_id = ENV['AZURE\_TENANT\_ID']**

**azure.client\_id = ENV['AZURE\_CLIENT\_ID']**

**azure.client\_secret = ENV['AZURE\_CLIENT\_SECRET']**

**azure.subscription\_id = ENV['AZURE\_SUBSCRIPTION\_ID']**

**# Specify VM parameters**

**azure.vm\_name = 'dob-azure'**

**azure.vm\_size = 'Standard\_B1s'**

**azure.vm\_image\_urn = 'Canonical:UbuntuServer:16.04-LTS:latest'**

**azure.resource\_group\_name = 'dob-vagrant'**

**azure.location = 'westeurope'**

**azure.tcp\_endpoints = '80'**

**end**

**config.vm.provision :shell do |sh|**

**sh.privileged = false**

**sh.path = "provision.sh"**

**end**

**end**

Създаваме нов **provision.sh** файл със следното съдържание:

**#!/bin/bash**

**echo '\* Update and install Ansible ...'**

**sudo apt-get update**

**sudo apt-get install -y ansible**

**echo '\* Execute Ansible playbook ...'**

**ansible-playbook -v /vagrant/playbook.yml -i "localhost," --connection=local**

Създаваме нов **playbook.yml** файл със следното съдържание:

**---**

**- hosts: localhost**

**become: true**

**name: Install NGINX web server**

**tasks:**

**- name: Install the binary**

**apt: name=nginx state=present**

**- name: Start the service**

**service: name=nginx state=restarted enabled=yes**

**- name: Put new index file**

**copy:**

**content: "<h1>Hello from NGINX on Azure!</h1>"**

**dest: /var/www/html/index.html**

След това изпълняваме

**time vagrant up**

Можем да отворим уеб сесия към публичния адрес, за да проверим дали всичко работи според очакваното

И после, ако искаме, влизаме по познатия начин в конзолата

**vagrant ssh**

След това можем да излезем и да унищожим машината

**exit**

**vagrant destroy**