Cloud Automation with FCA Lab Guide

# Lab Objectives

The main objective of the Azure-lab guide is to demonstrate what was learned in the previous cloud courses. Using manual intervention or ARM templates, we discovered in previous cloud classes that there are many different ways to launch the VM series Palo Alto Firewalls in multiple cloud provider solutions.

This course will help teach other methods to launch the VM Series firewalls but instead using automation. The goal is to teach the benefits of automation and demonstrate the (FCA) tool’s capabilities.

The Flexible Cloud Automation (FCA) tool was developed as a data driven model framework to address the orchestration needs of all major cloud providers that use Palo Alto Networks Next Generation VM Series Firewalls.

It is done by using a data driven model to agnostically drive global variable inputs to generate a custom cloud orchestration template. This is built not only around the VM Series firewalls but also around the cloud provider infrastructure as well.

* Lab 1 involves preparation work to build a virtual Docker container and use Github to prepare FCA to be deployed.
* Lab 2 involves using the (FCA) Cloud Automation tool to build a basic Azure environment and launch the Palo Alto Networks Firewall.

# Lab Design

All Labs can be accomplished by using Docker and FCA Framework. FCA framework will build different topologies per cloud provider based on specified inputs created in the FCA virtual\_networks and group\_vars file.

Table of Contents

[Lab Objectives 1](#_Toc11073609)

[Lab Design 2](#_Toc11073610)

[1. Lab: Using Docker and Github to prepare FCA tool. 5](#_Toc11073611)

[1.1. Lab Objectives 5](#_Toc11073612)

[1.2. Prepare and verify all lab files. 5](#_Toc11073613)

[1.3. Download Prepared Docker 6](#_Toc11073614)

[1.4. Obtaining Container Image 6](#_Toc11073615)

[1.4.1. Option #1: Pull the Dockerfile from the Docker Repository 6](#_Toc11073616)

[1.4.2. Option #2: Obtaining Container Image from GitHub 6](#_Toc11073617)

[1.5. Build from Dockerfile 7](#_Toc11073618)

[1.6. Launch Docker Container and Custom Tag 7](#_Toc11073619)

[1.7. Start docker container and mount the root folder 8](#_Toc11073620)

[1.8. Removal of Docker images installed. 9](#_Toc11073621)

[2. Lab: Using FCA for Azure Cloud 10](#_Toc11073622)

[2.1. Azure Lab Topology 10](#_Toc11073623)

[2.2. Virtual Networks files 11](#_Toc11073624)

[2.3. Define Hub Resources 11](#_Toc11073625)

[2.3.1. Create the Firewalls 11](#_Toc11073626)

[2.3.2. Creating Load Balancers 12](#_Toc11073627)

[2.3.3. Creating Virtual Networks 13](#_Toc11073628)

[2.3.4. Creating Security Groups 14](#_Toc11073629)

[2.4. Define Spoke Resources 15](#_Toc11073630)

[2.4.1. Creating Test Host VM 15](#_Toc11073631)

[2.4.2. Creating Security Groups 16](#_Toc11073632)

[2.4.3. Creating Virtual Network 16](#_Toc11073633)

[2.4.4. Create Route Table 17](#_Toc11073634)

[2.4.5. Group Vars file 18](#_Toc11073635)

[2.5. Set Parameters in all.yml file 19](#_Toc11073636)

[2.5.1. Include Networks 19](#_Toc11073637)

[2.5.2. Firewall Configuration 20](#_Toc11073638)

[2.6. Cloud Provider Information 21](#_Toc11073639)

[2.7. Connect to Docker Container and to the Azure Portal 22](#_Toc11073640)

[2.7.1. Connect to Docker Instance 22](#_Toc11073641)

[2.7.2. Connect to the Azure Portal 24](#_Toc11073642)

[2.7.3. Push Configuration to the Azure Cloud 25](#_Toc11073643)

[2.8. Review configuration and adopt some manual Changes 25](#_Toc11073644)

[2.8.1. Review Azure Private/Public Load Balancer 25](#_Toc11073645)

[2.8.2. Review Azure Load Balancing Rule 27](#_Toc11073646)

[2.8.3. Review Azure Route Tables 28](#_Toc11073647)

[2.8.4. Review Azure Virtual Networks 29](#_Toc11073648)

[2.8.5. Review Firewall Virtual Router 31](#_Toc11073649)

[2.8.6. Review Firewall Interfaces 33](#_Toc11073650)

[2.8.7. Review Firewall NAT / Security Rules 35](#_Toc11073651)

[2.8.8. Review Firewall Health Probe Traffic 36](#_Toc11073652)

[2.9. Configure / Test Connection to the Web Server 37](#_Toc11073653)

[2.10. Save and Destroy Lab Environment 39](#_Toc11073654)

[2.10.1. Save and Export Firewall Configuration 39](#_Toc11073655)

[2.10.2. Delete Azure Environment 40](#_Toc11073656)

# Lab: Using Docker and Github to prepare FCA tool.

## Lab Objectives

* Learn how to prepare FCA off of Github.
* Understand Github and Git-core functionality
* Learn how to download and build docker containers related to FCA.
* Learn all the files in the FCA Framework to build your first Azure Topology.

## Prepare and verify all lab files.

**Important Note Please Read!**

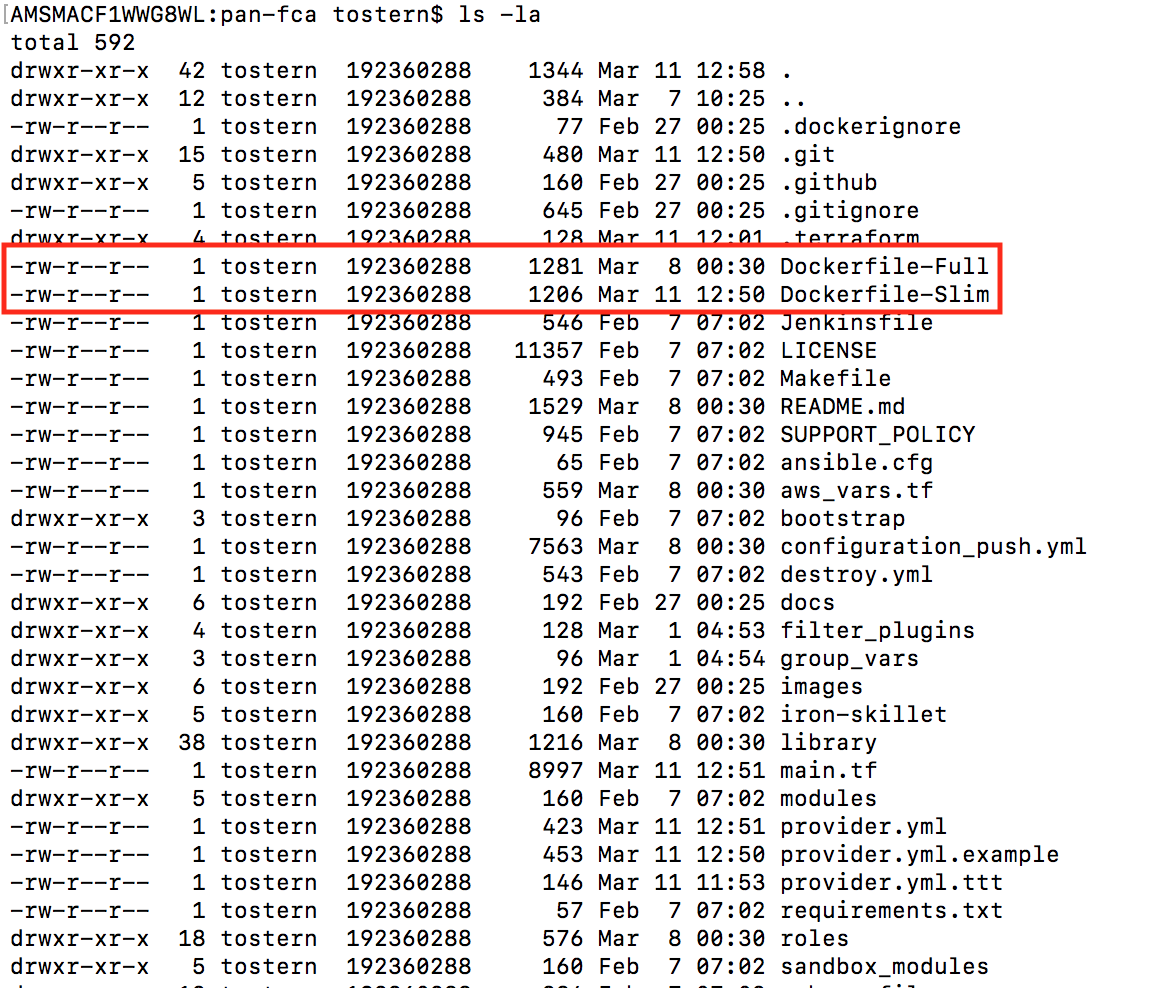
You will need permissions to the: <https://github.com/PaloAltoNetworks/pan-fca> upstream/master to download Dockerfile-FCA.

**(You should already have this by Forking and Cloning upstream/master FCA Github repository. If not please contact administrator of Repo if not public!) If you have trouble, please ask the course instructor for assistance.**

* Use Github to install FCA structure
* Use Docker to build, run and install infrastructure via container.
* Verify Prerequisites are loaded on local machine or Virtual machine which are Docker and Git hub.
* Verify access by visiting <https://github.com/PaloAltoNetworks/pan-fca/> docs/Docker\_instructions\_Readme.md
* Verify files for Labs exist and are mounted properly in the Docker container.

## Download Prepared Docker

Download Prepared Docker containers or build one off of the Github Cloned Fork.



## Obtaining Container Image

### Option #1: Pull the Dockerfile from the Docker Repository

Type the following command in your cli to download the Docker build:

docker pull panfca/tool:fca

### Option #2: Obtaining Container Image from GitHub

Browse to the folder where your forked FCA is located see example in [Chapter 1.3](#Download Prepared Docker)

## Build from Dockerfile

When used Option #2 we have to build the Docker instance out of the Dockerfile. For this type the following command. The “**.**” At the end of command below is really important!

docker build -t <StudentName> -f Dockerfile-Full .

## Launch Docker Container and Custom Tag

**Note:** If you want to create your own image name you can tag it with a custom tag.

Example: **docker tag panfca/tool:fca myfca**

Then ***“docker images”*** command will list all of your docker images.

Verify that your image with your tag is present

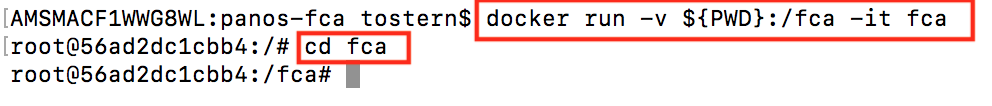
Make sure you are in the root fca folder of the cloned repository or specify full path before mounting.

## **Start docker container and mount the root folder**

docker run -v ${PWD}:/fca -it panfca/tool:fca

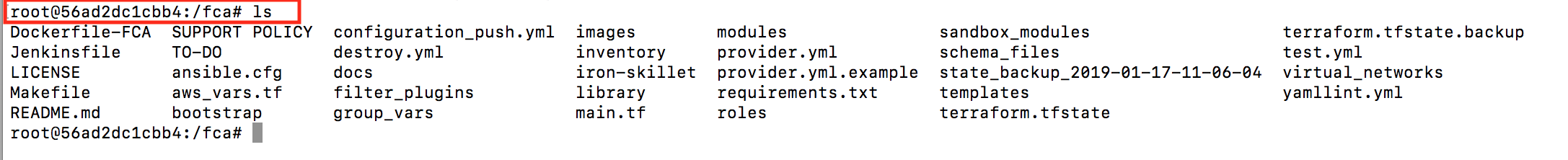
This mounts the local. /fca directory in the fca directory of the container, and launches container

Verify once container launches with ls then **cd** to ./fca mounted directory once inside the container to run playbook commands. See the example below:



Verify with the command “ls” that you are in the right folder. You have to see the following files at minimum:

* configuration\_push.yml
* destroy.yml
* modules
* group\_vars



## **Removal of Docker images installed.**

***Once all the labs are completed*** you can run the following command that will destroy all docker containers and volumes. This is also useful if there is limited space or are running on a temporary bastion box.

docker container stop $(docker container ls -a -q) && docker system prune -a -f --volumes

**\*Docker images are important and used to maintain code support version levels without the need to install on local machines. This also can help speed deployment times fast if FCA is building the infrastructure on a customer environment.**

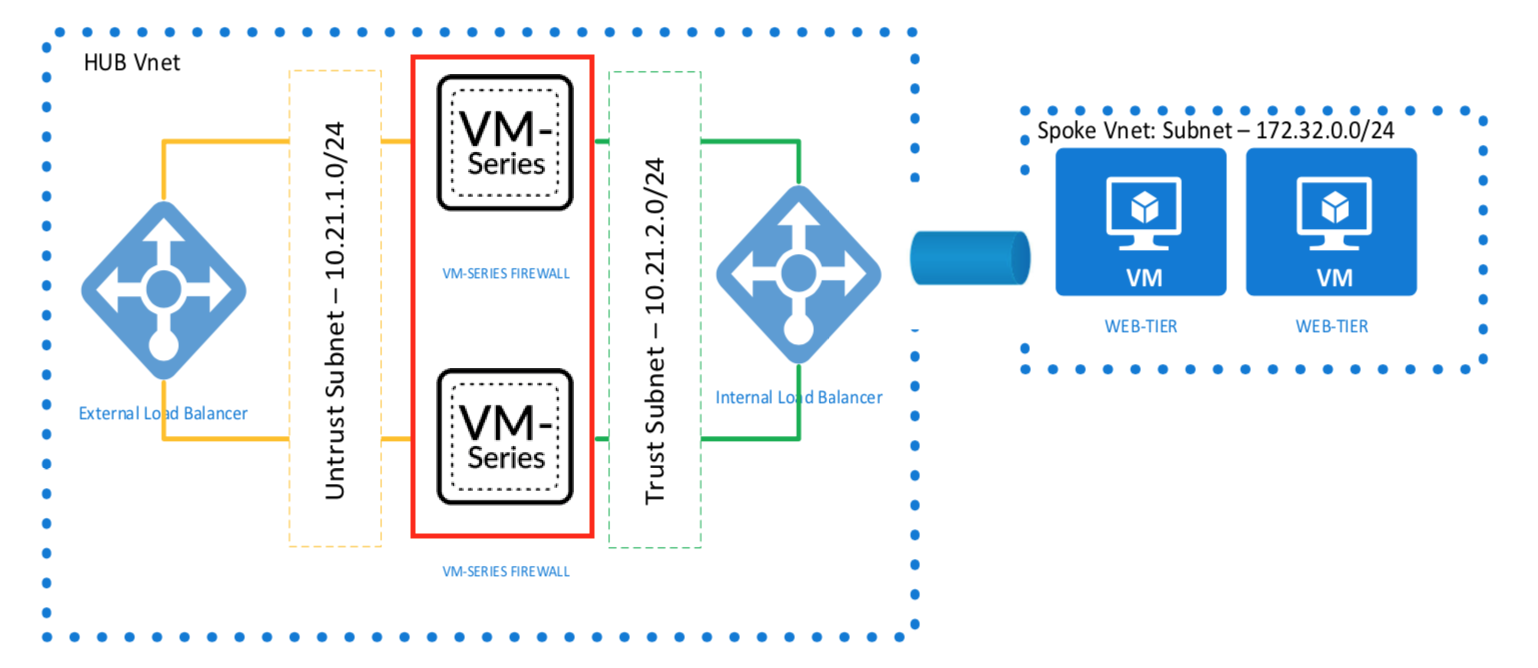
End of Lab #1

Following Labs #2, #3 #4 Lab will consist of preparing the files from Lab#1 objectives 1.7 and 1.8 for using specific parameters related to Instructor based Topology.

**Every Code Snippets are only examples and don’t provide you an 100% Solution to the Requirements of the Lab’s.**

# Lab: Using FCA for Azure Cloud

## Azure Lab Topology

****

* 1 Transit VNet – 10.217. XXX..0/24
* 4 Transit Subnets – Management (10.217.XXX.64/27), Trust (10.217.XXX.32/27), Untrust (10.217.XXX.0/27), <StudentName>Net (10.217. XXX..96/27)
* 1 Spoke VNet – 172.17. XXX..0/24
* 2 Spoke Subnets – Webserver (172.17. XXX..0/26), DB (172.17. XXX..64/26).
* 1 UDR in Spoke
* 2 Firewalls (Transit VNet)
* 3 Interfaces per FW (Trust, Untrust and Management)
* Public IPs on the firewall
* 1 Public Load Balancer
  + Probe: TCP 22
  + Load Balancing: TCP 80
* 1 Private Load Balancer
  + Probe: TCP 22
  + Load Balancing: HA Ports
* Peering between Transit and Spoke
* 1 Test VM in the Webserver Subnet in Spoke RG

**USE EVERYTIME THIS AS REFERENCE FOR YOUR CONFIGURATION**

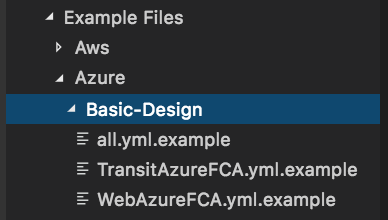
## Virtual Networks files

Inside the “pan-fca” folder find the “Example Files” folder and copy and paste the two Azure example files under the “Basic-Design” folder inside the “Virtual Networks” folder.

Change the name of the two files to:

“**Transit**<**StudentName>.yml”**

“**Web**<**StudentName>.yml”**

****

**The Filenames are important. The Filename will be later the name of your Resource Group.**

## Define Hub Resources

### Create the Firewalls

Open the up the **Tranist**<**StudentName>.yml** file that you have created in the previous step.

Starting from the top, modify the necessary fields:

firewalls:

- name: Firewall-Set-1

vmcount: 1

fwname: TSVMPAN-

fwvmsize: Standard\_D3\_v2

#https://knowledgebase.paloaltonetworks.com/KCSArticleDetail?id=kA10g000000ClD7CAK

avsetname: "AzureAV1"

fw\_version: 8.1.0

# 7.1.0 / 8.0.0 / 8.1.0 = latest

fw\_sku: byol

# byol / bundl1 / bundle2

Username: tstern

Password: PaloAlto1234!

attachtrustpool: "yes"

attachuntrustpool: "yes"

lbnameuntrust: External-LB

lbnametrust: Internal-LB

|  |  |  |
| --- | --- | --- |
| **name:** | This is Generic name not visible in Azure. | Add your initials to make the name unique. |
| **vmcount:** | Select how many Firewalls you want to deploy in the AVSet. | 2 are required to complete the lab. |
| **fwvmsize:** | Here you can define which VM Size you want to. | No changes needed. |
| **avsetname:** | Here you define the name of the Avaibility-Set. | Add your initials to make the name unique. |
| **fw version:** | Here you can define the PANOS Version (latest = 9.0.0) | Change it to 8.1.0. |
| **Username:** | Define a username (min. 5 characters) | Change as indicated |
| **Password:** | Define a Password (min 12 characters, min 1 capital and min 1 number) | Change as indicated |
| **Attach**  **Untrust/trustpool** | Here you can define if you want to have your FW assigned to an Loadbalancer Backendpool. | Default = „NO“ |
| **Lbname[trust/untrust]** | Here you define to which Load Balancer the FW hast o be assigned | Change it to the name oft he LB when you changed it. |

**\*\*\*Don’t use the Default Values for Username and Password\*\*\***

### Creating Load Balancers

In this part you can configure Load Balancer. The script will by default create a Standard Load Balancer. It is not needed to define an “Private” and “Public Load Balancer”.

load\_balancers:

- name: External-LB

fename: Untrust

bename: Untrust

floating\_ip: false

type: public

lbrulename: HTTP-80

lbrulefrontport: 80

lbruleprotocol: tcp

lbrulebackport: 80

lbprobename: TCP-22

lbruleprobeport: 22

- name: Internal-LB

fename: Trust

bename: Trust

floating\_ip: true

type: private

lbrulename: HA

lbrulefrontport: 0

lbruleprotocol: All

lbrulebackport: 0

lbprobename: ssh

lbruleprobeport: 22

|  |  |  |
| --- | --- | --- |
| **Name:** | Defines the name of the Loadbalancer | Add your initials to make the name unique. |
| **FeName:** | Defines the name of the Frontendpool | No changes needed. |
| **BeName:** | Defines the name of the Backendpool | No changes needed. |
| **Type:** | Defines which type you want to have (private or public) | No changes needed. |
| **LBRuleName:** | Defines a name for the Loadbalancer rule | No changes needed. |
| **LBRuleFrontEndPort:** | Defines the port of the Frontendpool | No changes needed. |
| **LBRuleFrontBackndPort:** | Defines the port of the Backendpool | No changes needed. |
| **LBRuleProtocol:** | Defines the Loadbalancer Probe Protocol | No changes needed. |
| **LBProbeName:** | Defines the LoadBalancer Probe name | No changes needed. |
| **LBRuleProbePort:** | Defines the Loadbalancer Probe port | No changes needed. |
| **Floating\_ip** | This defines if you want use floating ip for your LB rules | No changes needed |

In this Section is not really needed to change the default values. But you can change it to make it for better.

### Creating Virtual Networks

In this Section you define the Virtual Network (VNet) and your Subnets for the Hub Resource Group (RG).

**Please Replace the third CIDR Octet with your Student number provided by the Instructor.**

vnet\_network:

name: Transit-Azure

network: "10.217.XXX.0/24"

peers:

-Web<StudentName>

subnet:

- name: Management

network: "10.217.XXX.64/27"

- name: Trust

network: "10.217.XXX.32/27"

- name: Untrust

network: "10.217.XXX.0/27"

|  |  |  |
| --- | --- | --- |
| **name:** | Defines the name of the Virtual Network | Add your initials to make the name unique. |
| **network:** | Defines the VNet CIDR | Enter the number provided by the instructor |
| **peers:** | Defines the Peer config | Type the filename of your spoke but without **“.yml”** |
| **subnet:** | Defines the subnets of the VNet | Enter the number provided by the instructor |

**IMPORTANT the first Network has to be the Firewall Network**

**1 Subnet = Management; 2 Subnet = Trust; 3 Subnet = Untrust**

### Creating Security Groups

In this section you can define your Network Security Groups (NSG) what you can then later attach to the needed interfaces. By default, is to every interface an “allow all” NSG applied to not having any issues in the beginning.

security\_groups:

- name: Outbound allow

priority: 1000

direction: inbound

action: allow

src\_ip: 0.0.0.0

src\_port: any

dst\_port: any

dst\_network: 0.0.0.0

protocol: any

|  |  |  |
| --- | --- | --- |
| **name:** | Defines the name of the NSG | Add your initials to make the name unique. |
| **priority:** | Defines the Priority of the Rule | No changes needed. |
| **direction:** | Defines the direction (inbound or outbound) | No changes needed. |
| **action:** | Defines the action (allow or deny) | No changes needed. |
| **src\_ip:** | Defines the Source IP | No changes needed. |
| **src\_port:** | Defines the Source port | No changes needed. |
| **dst\_port:** | Defines the destination port | No changes needed. |
| **dst\_network:** | Defines the destination network | No changes needed. |
| **protocol:** | Defines the protocol (tcp, udp or any) | No changes needed. |

## Define Spoke Resources

### Creating Test Host VM

Open the up the „**Web<StudentName>.yml“** file that you have created in the previous step.

In the top of the file you will find the Example for the Testhost VM. Starting from the top, modify the necessary fields.

testhost:

- name: Testhost

hostname: Cloud-Test

username: creator

password: PaloAto123456789

dnsname: ubuntutestvm

|  |  |  |
| --- | --- | --- |
| **name:** | This is Generic name visible in Azure. | Add your initials to make the name unique. |
| **hostname:** | Defines the name of the Testhost | No changes needed. |
| **username:** | Defines your username | Change as indicated |
| **password:** | Defines your password (min. 12 characters) | Change as indicated |
| **dnsname:** | Defines the DNS name of the host ***(only use lower case)*** | Add your initials to make the name unique. |

**\*\*\*Don’t use the Default Values for Username and Password\*\*\***

### Creating Security Groups

In this section you can define a Security Group (NSG) when you don’t define one is no NSG applied.

security\_groups:

- name: Outbound allow

priority: 100

direction: inbound

action: allow

src\_ip: 1.1.1.1

src\_port: any

dst\_port: any

dst\_network: 0.0.0.0/0

protocol: any

|  |  |  |
| --- | --- | --- |
| **name:** | here you define the name of the NSG | Add your initials to make the name unique. |
| **priority:** | here you define the Priority of the Rule | No changes needed. |
| **direction:** | here you define the direction (inbound or outbound) | No changes needed. |
| **action:** | here you define the action (allow or deny) | No changes needed. |
| **src\_ip:** | here you define the Source IP | Change it to 0.0.0.0/0 or your PIP |
| **src\_port:** | here you define the Source port | No changes needed. |
| **dst\_port:** | here you define the destination port | No changes needed. |
| **dst\_network:** | here you define the destination network | No changes needed. |
| **protocol:** | here you define the protocol (tcp, udp or any) | No changes needed. |

### Creating Virtual Network

In this Section you define the Virtual Network (VNet) and your Subnets for the Spoke Resource Group (RG). There is now default Value available.

**Please Replace the third CIDR Octet with your Student number provided by the Instructor.**

You have to type filename of the Hub in the peer section (case sensitive).

vnet\_network:

name: Web-Azure

peers:

- Transit<StudentName>

network: "172.16.XXX.0/24"

subnet:

- name: Webserver

network: "172.16.XXX.0/26"

- name: DB

network: "172.16.XXX.0/26"

- name: Test

network: "172.16.XXX.0/26"

|  |  |  |
| --- | --- | --- |
| **name:** | Defines name of the Virtual Network | Add your initials to make the name unique. |
| **network:** | Defines the VNet CIDR | Enter the number provided by the instructor |
| **peers:** | Defines the name of peer hub config. | You have to type the filename of your HUB without “.yml” |
| **subnet:** | Defines the subnets of the VNet | Enter the number provided by the instructor |

### Create Route Table

The Route table in this Section is needed to forward the traffic from the Hub to the Transit.

route\_tables:

- name: web-test

routes:

- cidr: 0.0.0.0/0

name: VirtualAppliance

gateway: 1.1.1.1

|  |  |  |
| --- | --- | --- |
| **name:** | Defines the name of the Route Table (UDR) | No changes needed. |
| **cidr:** | Defines the CIDR of your route | No changes needed. |
| **name:** | Defines the type of nexthop type. | No changes needed. |
| **gateway:** | Defines the gateway ip of the “VirtualApplinace” | No changes needed. (Placeholder) |

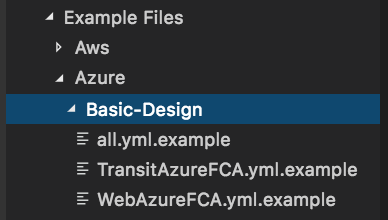
The value “1.1.1.1” under gateway is only a placeholder and we will change this value later in the Azure Porta

When you done all your changes to the files above, check if you had saved them.

### Group Vars file

You can find the all.yml.example under the **„Examples -> Azure -> Basic-Desing“** folder.

Copy and Paste the all.yml.example file into the **„group\_vars -> all“** folder and remove the **“.example”**.



This file is where a lot of your Firewall configuration will take place. This is Ansible driven, but you will be able to set your security policy rules, routes and admin username and passwords here. See the next Steps.

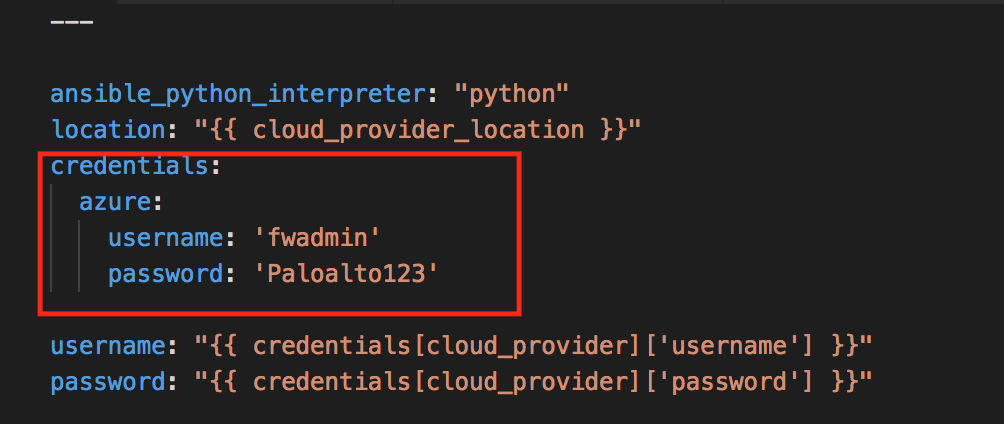
**Important. By default, is the Iron-Skillet loaded in the beginning to the Firewalls**

**Here is the Link where you can see what is loaded.**

https://github.com/PaloAltoNetworks/iron-skillet/blob/panos\_v8.0/loadable\_configs/sample-cloud-Azure/panos/iron\_skillet\_panos\_full.xml

## Set Parameters in all.yml file

Under the section **“credentials”** type the username and password from the firewall what you have defined before in the **“Transit<StudentName>.yml”.** If you haven’t defined this, let the values in the **“all.yml”**. These are the default values.



### Include Networks

You have to type under virtual\_networks the filenames, which you have created under the Chapter **“Virtual Networks files”.** You must remove the **‘.yml’** as you can see in the example below.

virtual\_networks:

- Transit<StudentName>

- Web<StudentName>

### Firewall Configuration

Everything below the “virtual\_networks” section is a part of the Firewall Configuration.

You can use almost everything from this example. **Review all CIDR’s that it matches to your update Student CIDR.**

* Create Management Profile = cm\_mgmt\_profile
* Configure Interfaces = cm\_interface:
* Create Address Objects = cm\_object\_address:
* Create Service Objects = cm\_object\_service:
* Create Service Groups = cm\_object\_service\_group:
* Create Address Groups = cm\_object\_address\_group:
* Create Application Groups = cm\_object\_app\_group:
* Configure NTP = cm\_ntp:
* Configure DNS = cm\_dns:
* Configure Panorama = cm\_panorama1:
* Create Firewall accounts = cm\_fw\_user:
* Create Security Rules = cm\_security\_rule:
* Create NAT Rules = cm\_nat\_rule: **- Edit NAT to the appropriate CIDR**
* Create Static Routes = cm\_panos\_static\_route: **- Edit routes to the Appropriate CIDR**
* Activate Firewall = cm\_panos\_lic:
  + Example file is commented out. remove hash, continue steps below, after the setup is complete and you receive the licensing error, add the hash and run again config again.
* Update Firewall = cm\_panos\_software:
  + Commented out and not required to complete the lab.

**Important. Don’t use the “cm\_panos\_lic” module. This module occurs an error. We will provide you during the class all needed information to fix this!**

## Cloud Provider Information

In the root folder structure, you can find the “provider.yml.example” file, clone this file and remove the .example extensions

****

Inside this file you should delete all AWS related things.

As next update the fields as you can see below:

## AZURE EXAMPLE

cloud\_provider: azure

cloud\_provider\_location: "north europe"

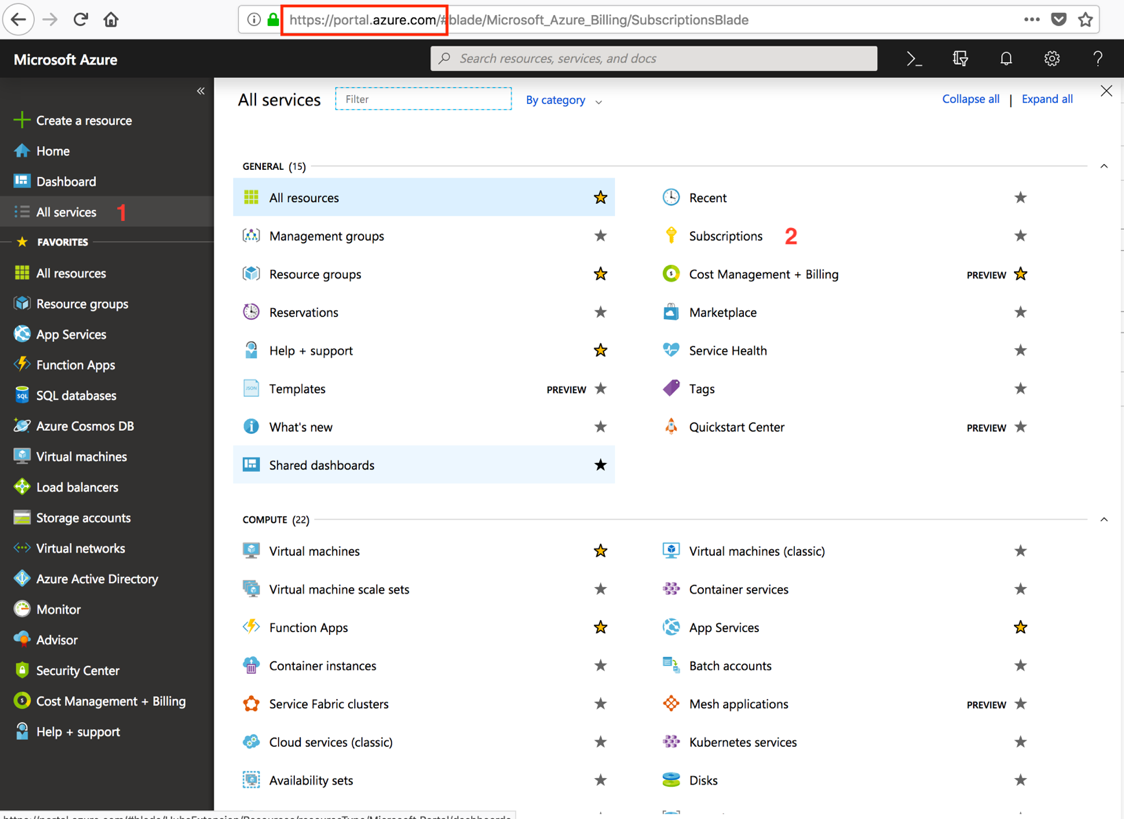
azure\_subscription\_id: "d47XXXXX-XXXX-XXXX-XXXX-da72XXXXXXX"

#Operating System used to select go binary type

go\_os: "ubuntu"

If you want to change your “cloud\_provider\_location” look at this this link <https://azure.microsoft.com/en-us/global-infrastructure/locations/> and replace “north Europe” with you location.

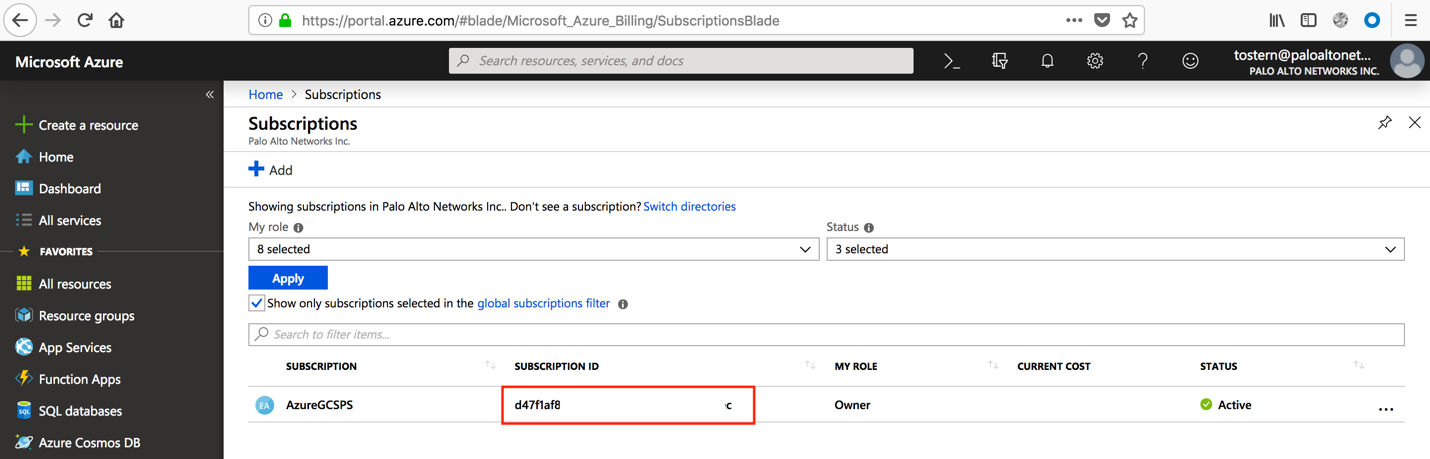
You can find your **“azure\_subscription\_id”** in your Azure Portal. See below:



Go to <https://portal.azure.com> and sign in with your Palo Alto Credentials. When you have problems with that contact please any of the Instructors.

In the click on **“All services”** and then on “**Subscriptions”**.

Copy as next the Subscription ID and paste it in the **“azure\_subscription\_id”** filed in the **“provider.yml”** file



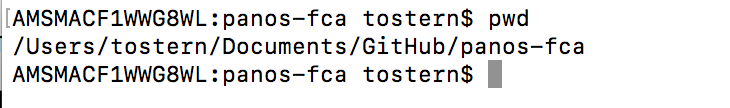
## Connect to Docker Container and to the Azure Portal

When you have defined everything in related file above we can then start to connect to the Azure Portal and the Docker container

### Connect to Docker Instance

We are now going to run the master configuration push from your Docker container. For that connect at first to your Docker instance what you have created in the Lab 1.

1. Go to the location where you have stored the FCA GitHub content (see example below)



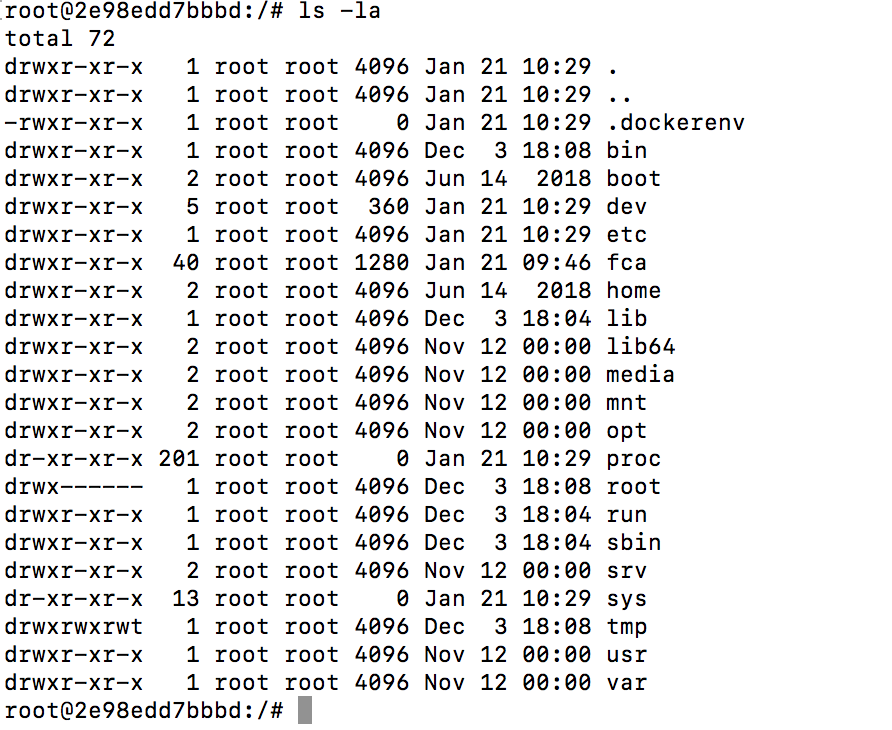
1. Type the following command to run the Docker Container

docker run -v $(PWD):/fca -it panfca/tool:fca

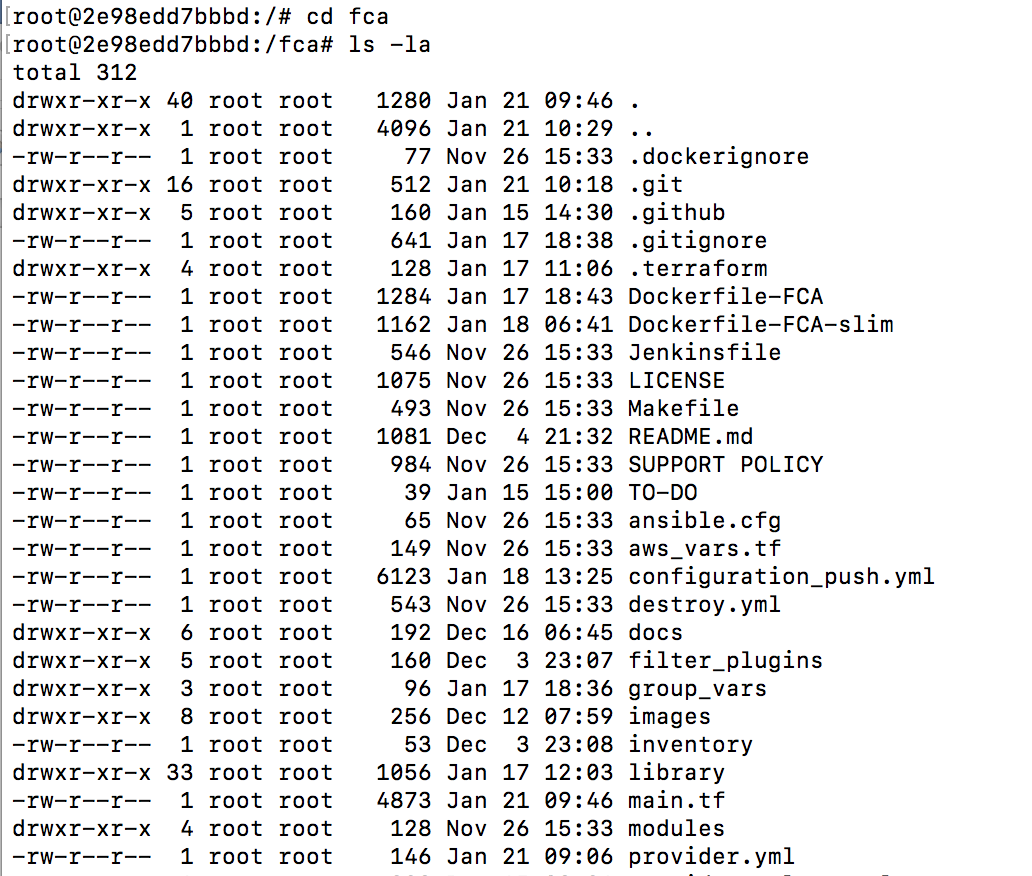
1. If, you have defined another tag instead fca, replace please

Docker run -v $(PWD):/fca -it panfca/tool:\*your tag\*

1. After running this command, you should be connected to your Docker instance and in the root folder of the container. Type “ls -la” to verify that you can see the “ fca” folder.



1. As next type “cd fca” to access the FCA folder and you should be able to see all folders and files of the fca repo.



We are now ready to go to the next step, if you are can see the folders above. In case you cannot see it review the steps above or consult any Instructor.

### Connect to the Azure Portal

Now we can connect to the Azure Portal, when you are successful connect to your Docker Container.

In your Docker instance type the following command to connect to the Azure Portal

az login -u username@paloaltonetworks.com

See the two different behaviors below:

1. A browser window is opening automatically, and you have to authenticate yourself with username and password. **OR**
2. In the command line you get an information back to go to the page <https://microsoft.com/devicelogin> and authenticate with a device code and your username and password. See the example below:



You can use the following command in the cli, when you know your username and password. **“az login -u <username>”.** See the example below



## Push Configuration to the Azure Cloud

By now you should be happy with your configurations in the files above and you should be successful connected to Azure Portal and your Docker Instance.

Now we can start with the deployment of the Azure environment and the Firewall configuration, when the above criteria are right.

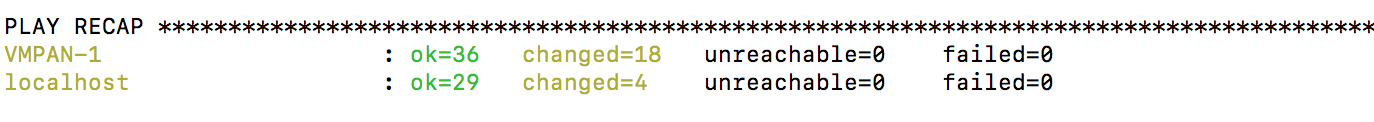
In your Docker instance the following command

Ansible-playbook configuration\_push.yml

The deployment and the configuration could take **5 between 20 minutes** when you have not deployed more than 4 Firewalls.



When the Script is finish it will commit the configuration and you should see the following output.



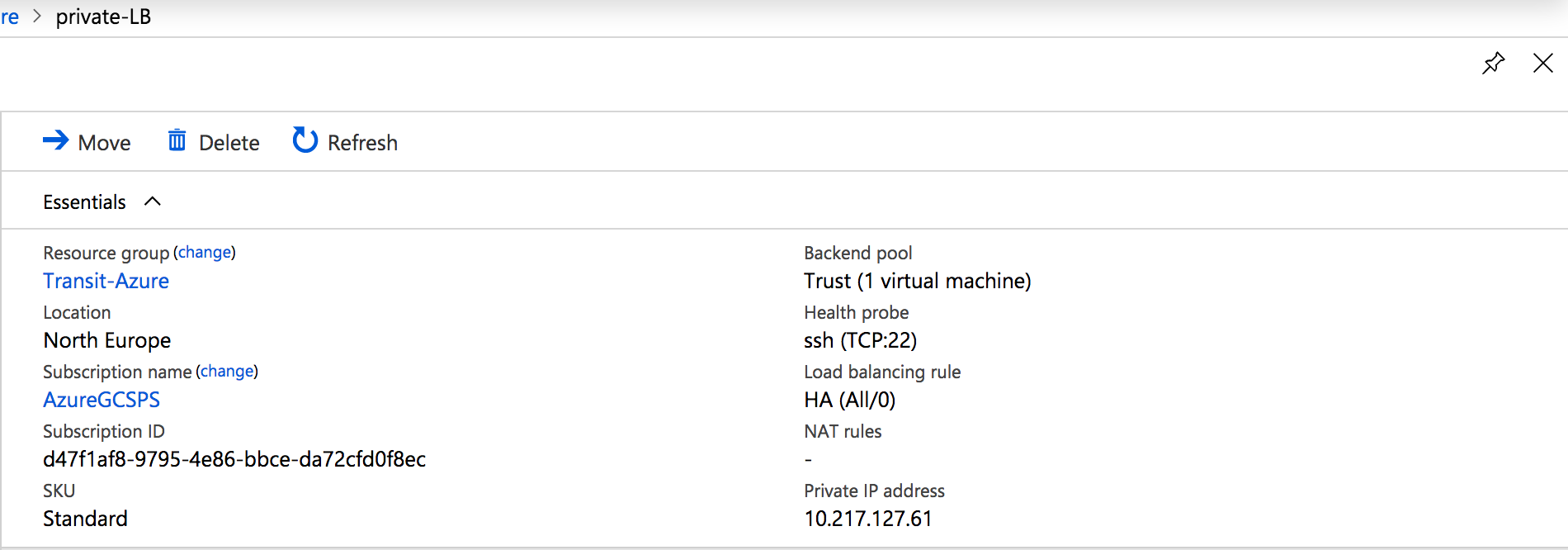
If you don’t get the following output and the script is failing, please contact the Session instructor.

## Review configuration and adopt some manual Changes

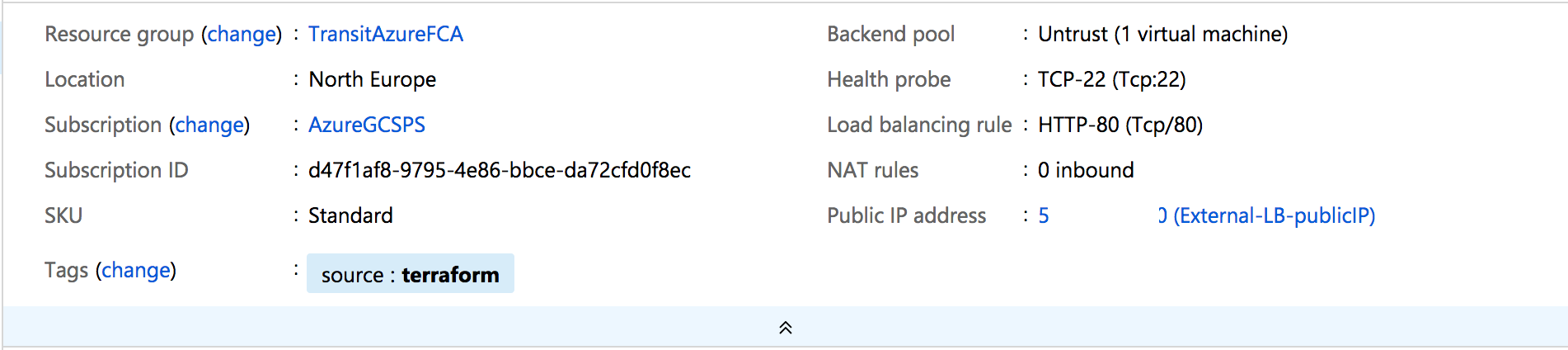
### Review Azure Private/Public Load Balancer

The Overview of the Load Balancers should look like the following examples:

Private Load Balancer



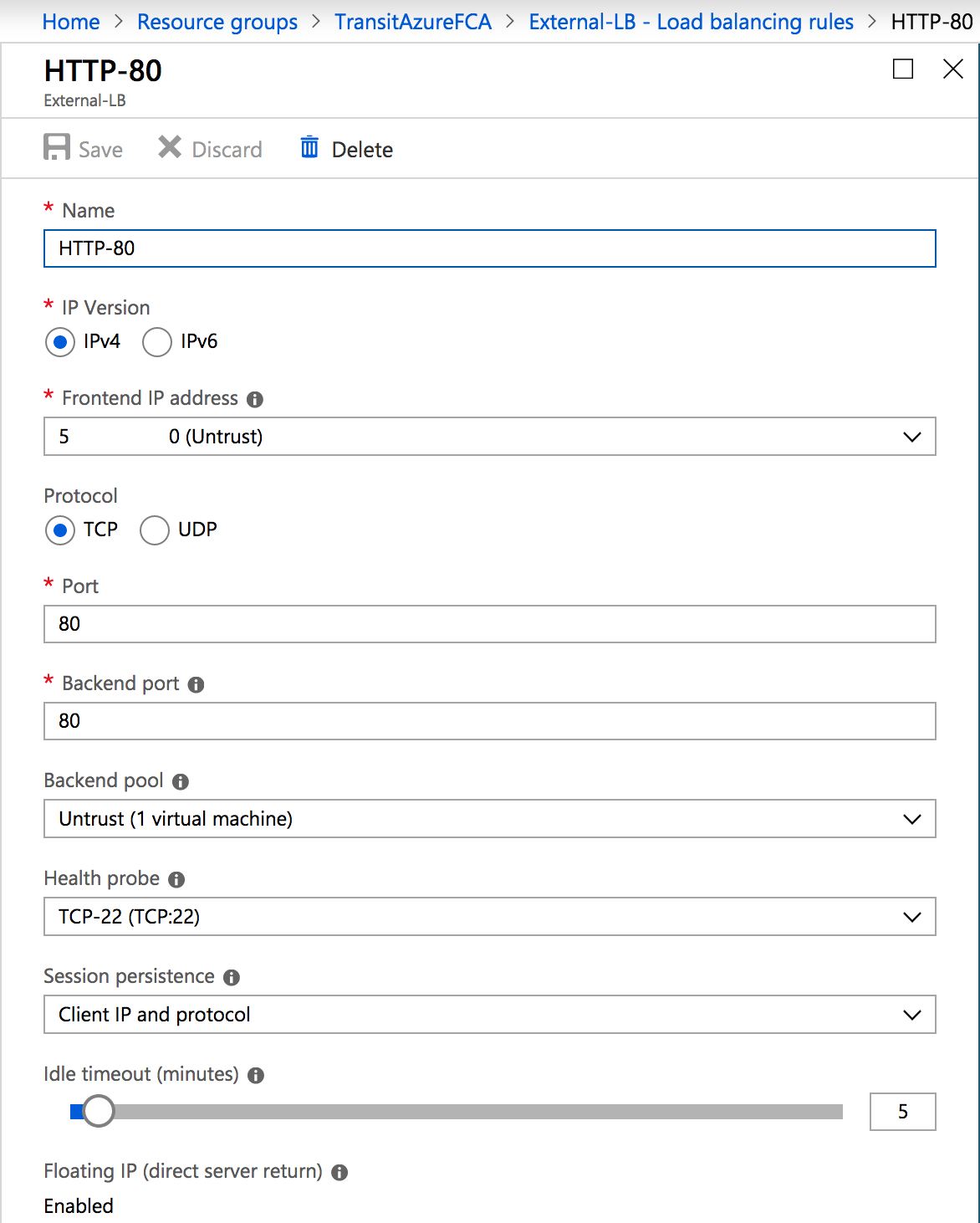
Public Load Balancer



Please take a note of the Public IP of the Load Balancer. This IP is needed in later step.

## Review Azure Load Balancing Rule

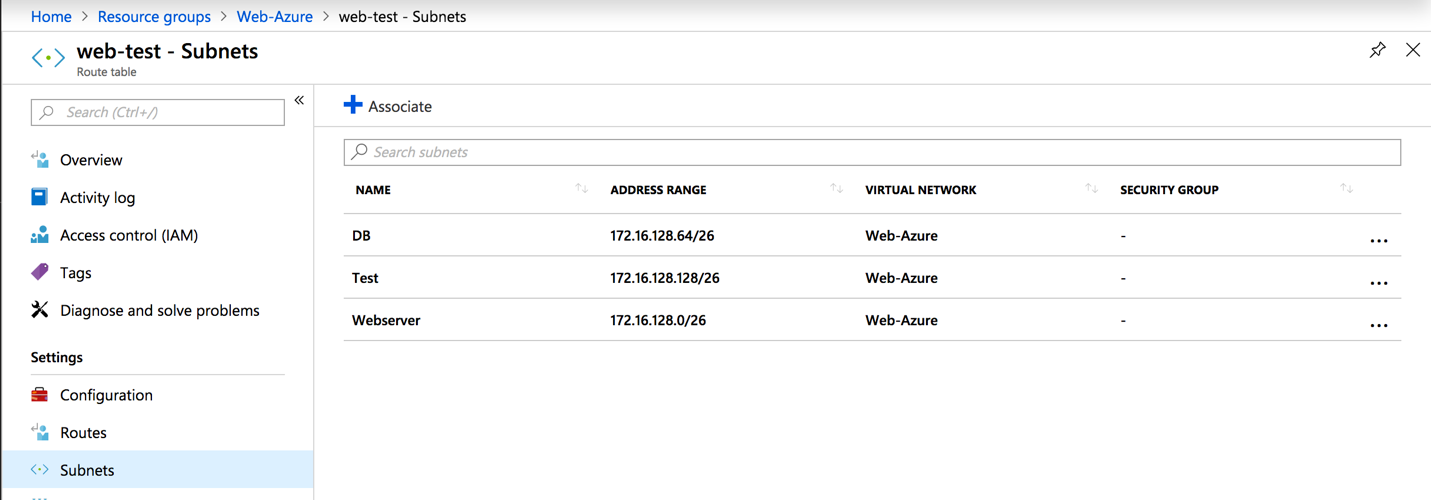
The FCA script will create for you Load Balancing Rule. Review that the rule is in place. This rule will forward all traffic on the port TCP 80.



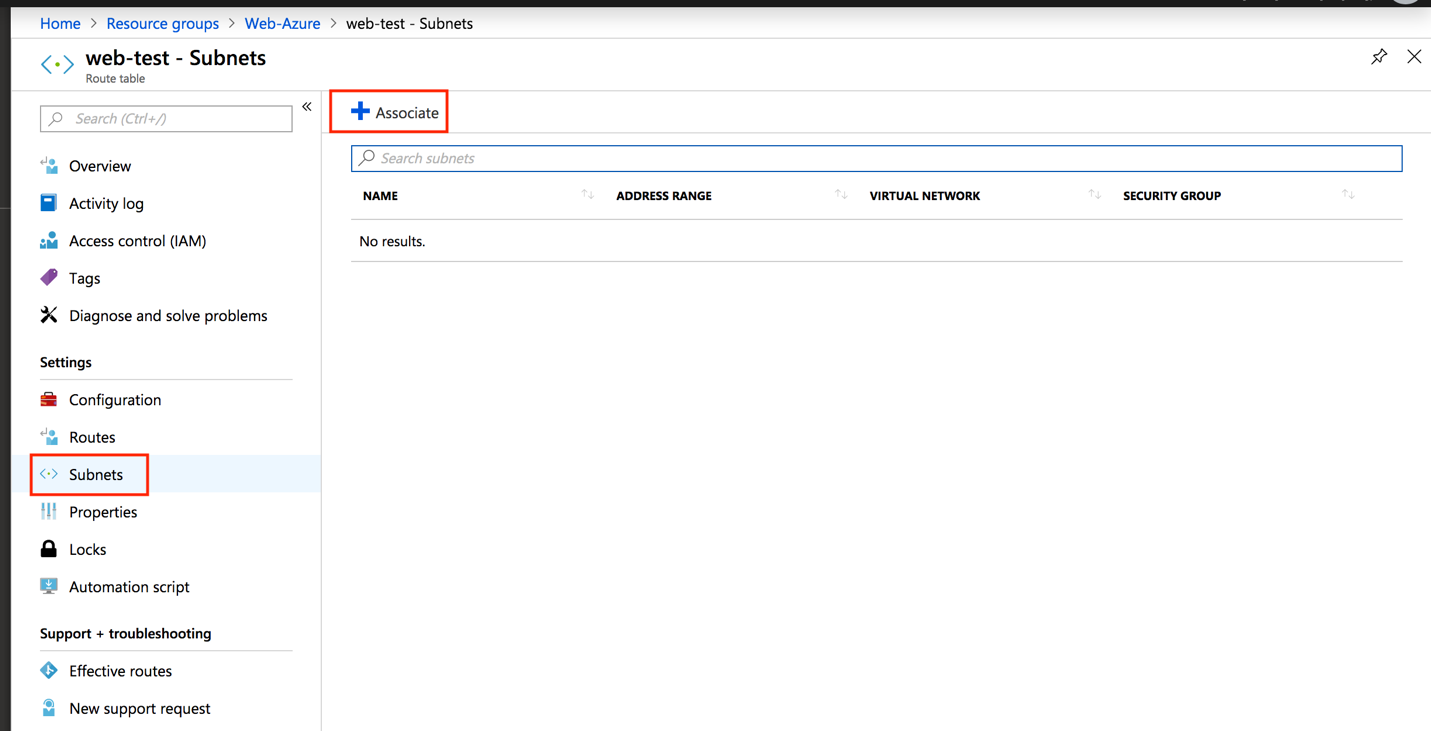
This Rule is needed that the traffic is forwarded to the Web-Server.

### Review Azure Route Tables

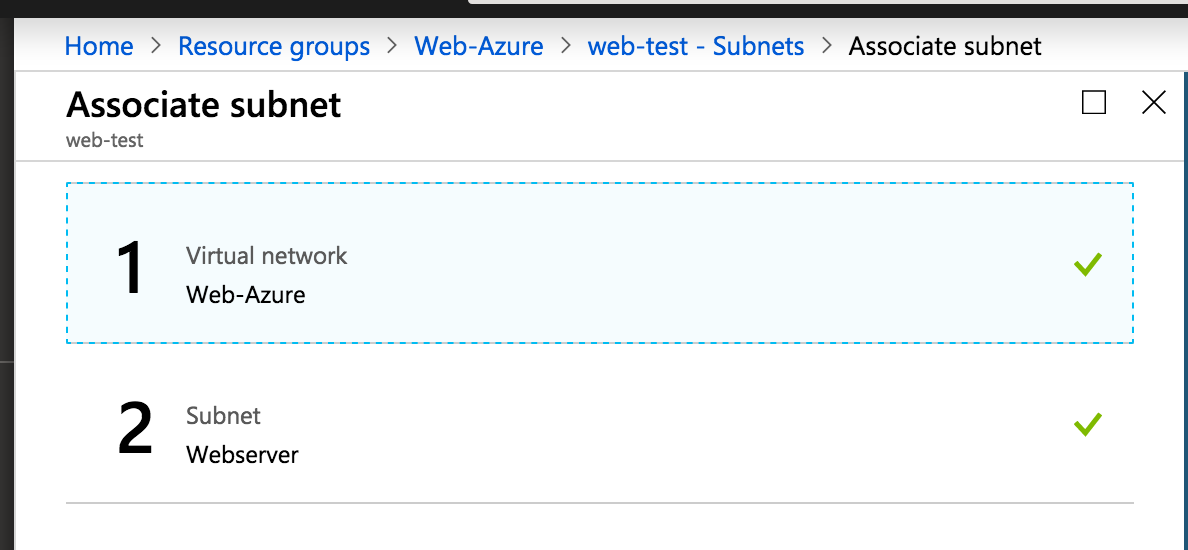
Go to your Spoke Resource Group and verify there if all Subnets are Associated to the route table. See example below.



If the Subnets are not associated to the route table, do this by yourself. Click on “Associate”.



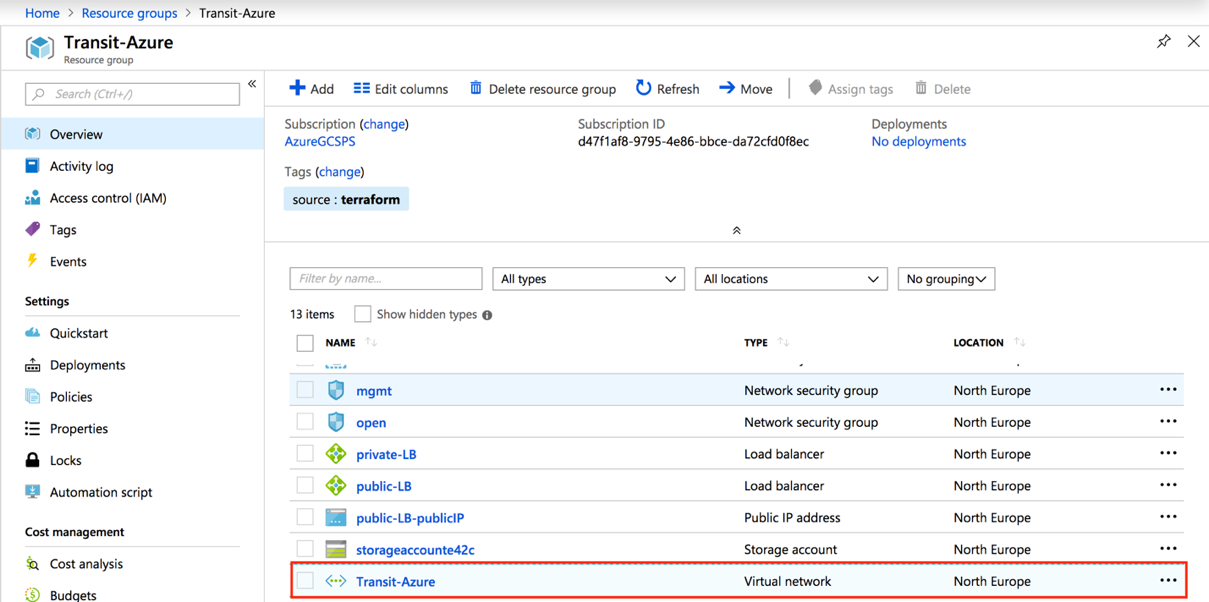
Associate all related Subnets to this route table. Repeat this step so long if you have added all subnets.



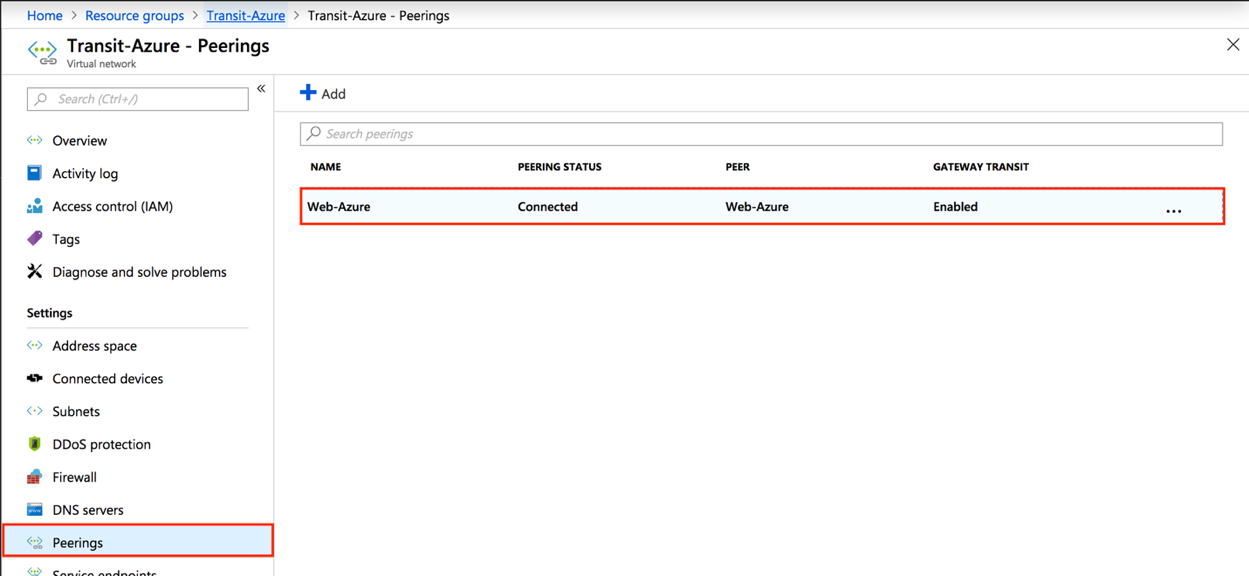
This is not really needed that the traffic is passing, but it is needed to use “effective routes”

### Review Azure Virtual Networks

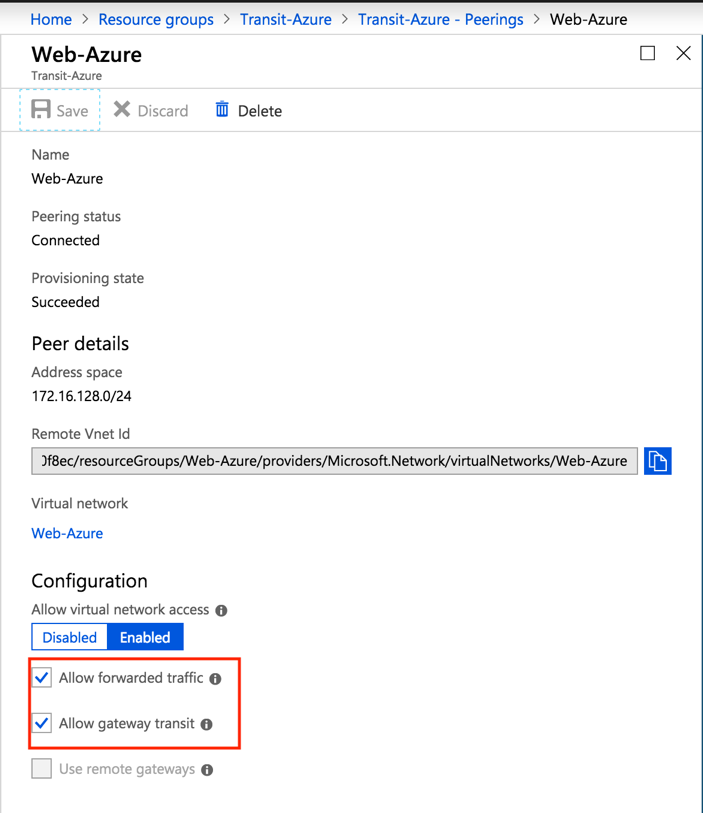
In this Part we have to review does the VNet Peering in both Resource Group (RG) are right. At first check the Virtual Network (VNet) in your Transit RG.



In the Transit VNet click on “Peerings” in the left column. When there is a peering existing click on it to review if the settings are right. When there is no peering visible, go directly to next step and create it by yourself.



The settings in the Transit VNet peering should looks like this. Is this not the case change it and click on “Save”.



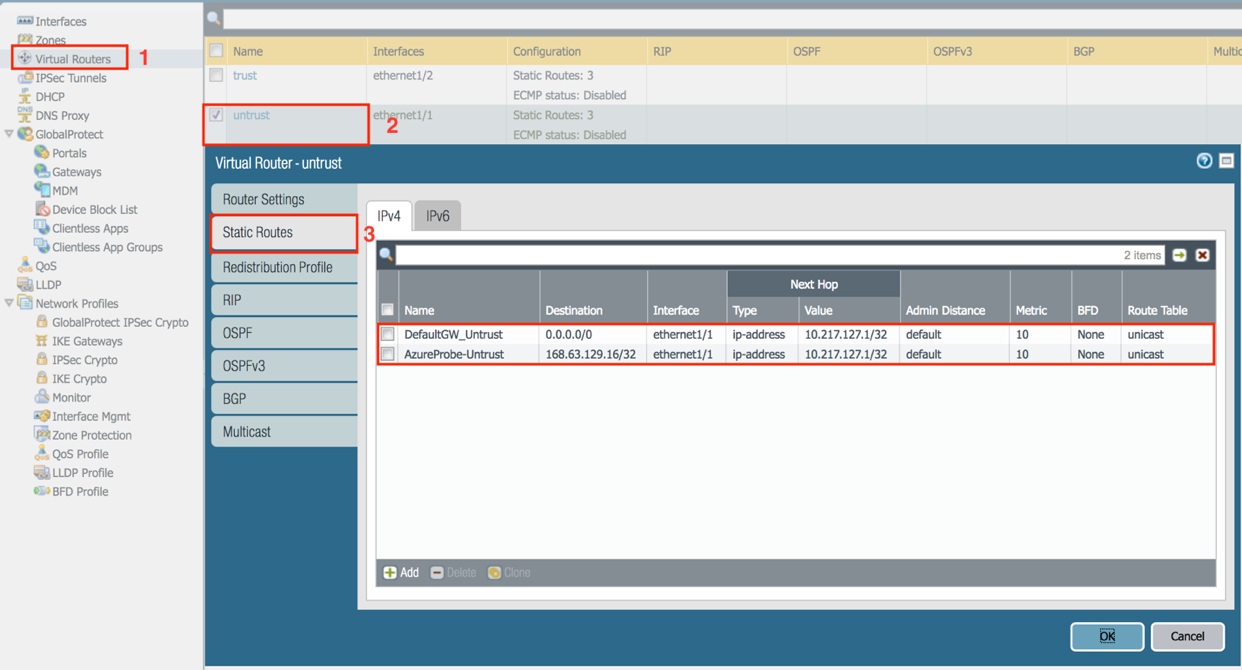
Repeat the above described steps for the Hub VNet. Below you can find the settings of route table. Please adopt the settings when it is not the same.



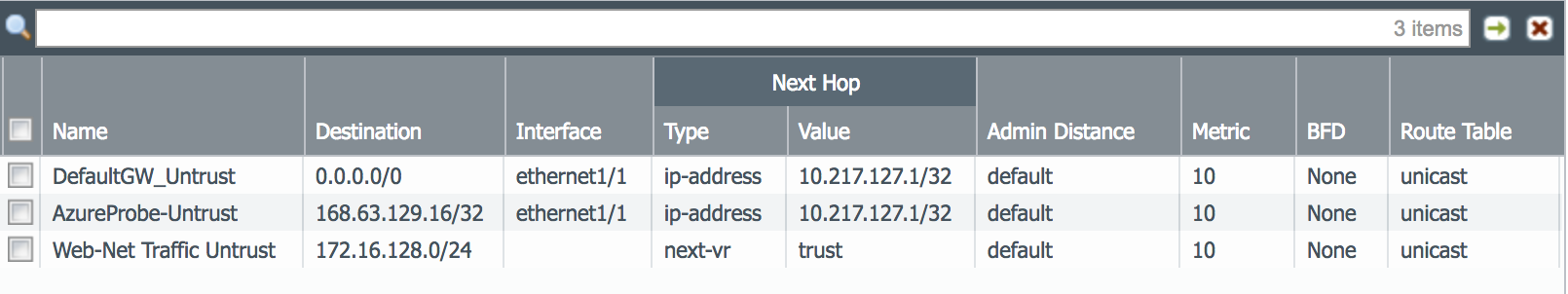
### Review Firewall Virtual Router

In this Part we have to review does the Virtual Routers on the Firewalls are right configured. At first check the “untrust-vr”. You should have at least two static routes.

* DefaultGW\_Untrust
* AzureProbe-Untrust

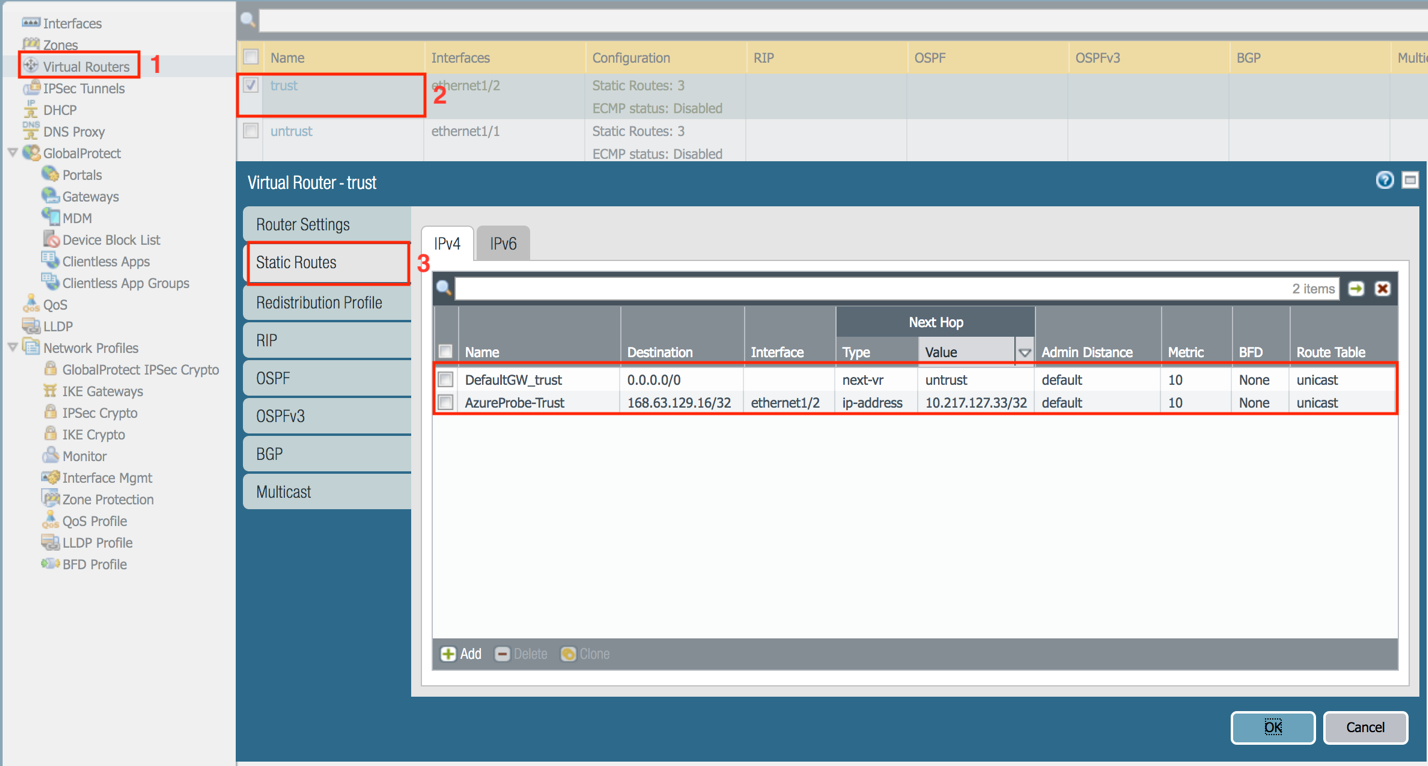


You have to adopt now a new Static Route for your Hub Network. See my example below.



the “trust-vr”. You should have at least two static routes.

* DefaultGW\_trust
* AzureProbe-Trust

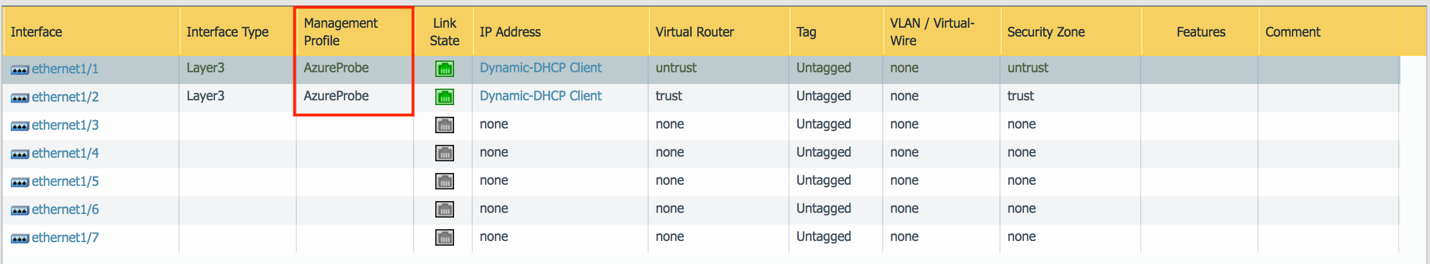


You have to adopt now a new Static Route for your Hub Network. See my example below.

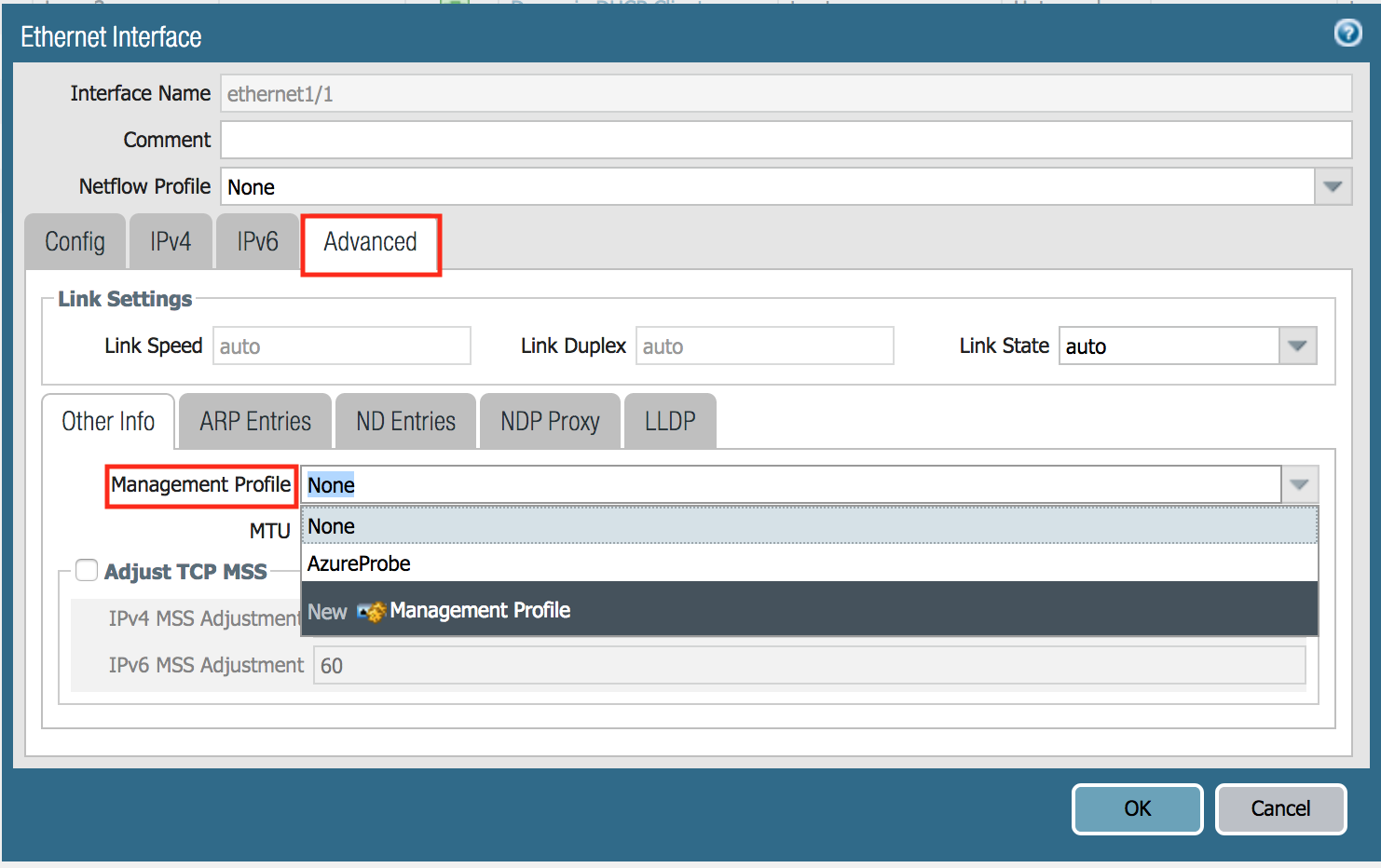


### Review Firewall Interfaces

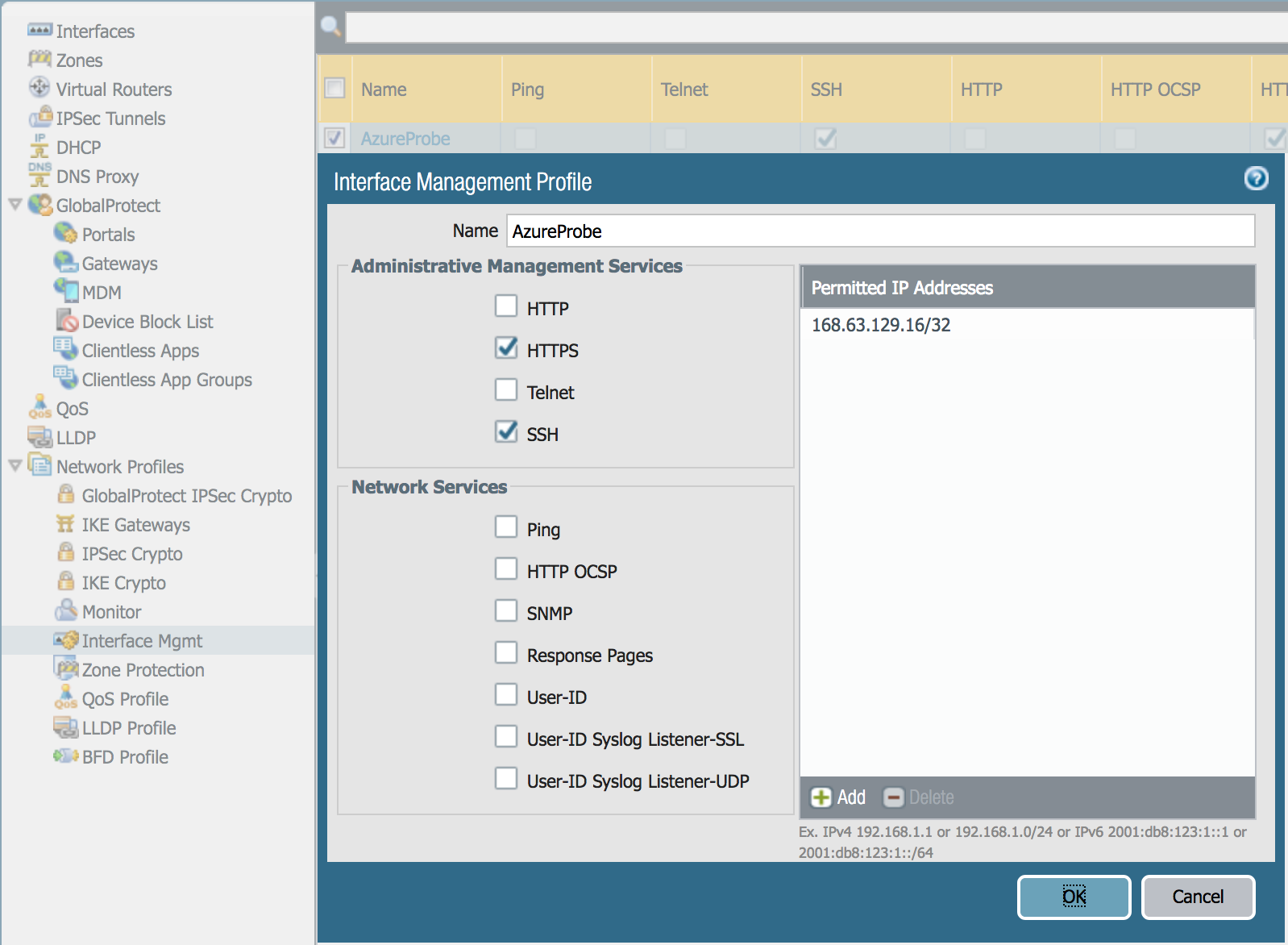
As next step we have to check if the Firewall interfaces are right configured. When a Load Balancer is attached is it important that on that interface is a Management Profile attached for the Azure health probe. When there is no profile attached do this please by yourself.



By default, is a Management Profile created (AzureProbe). Attach this profile to all needed interface where a Load Balancer is assigned.

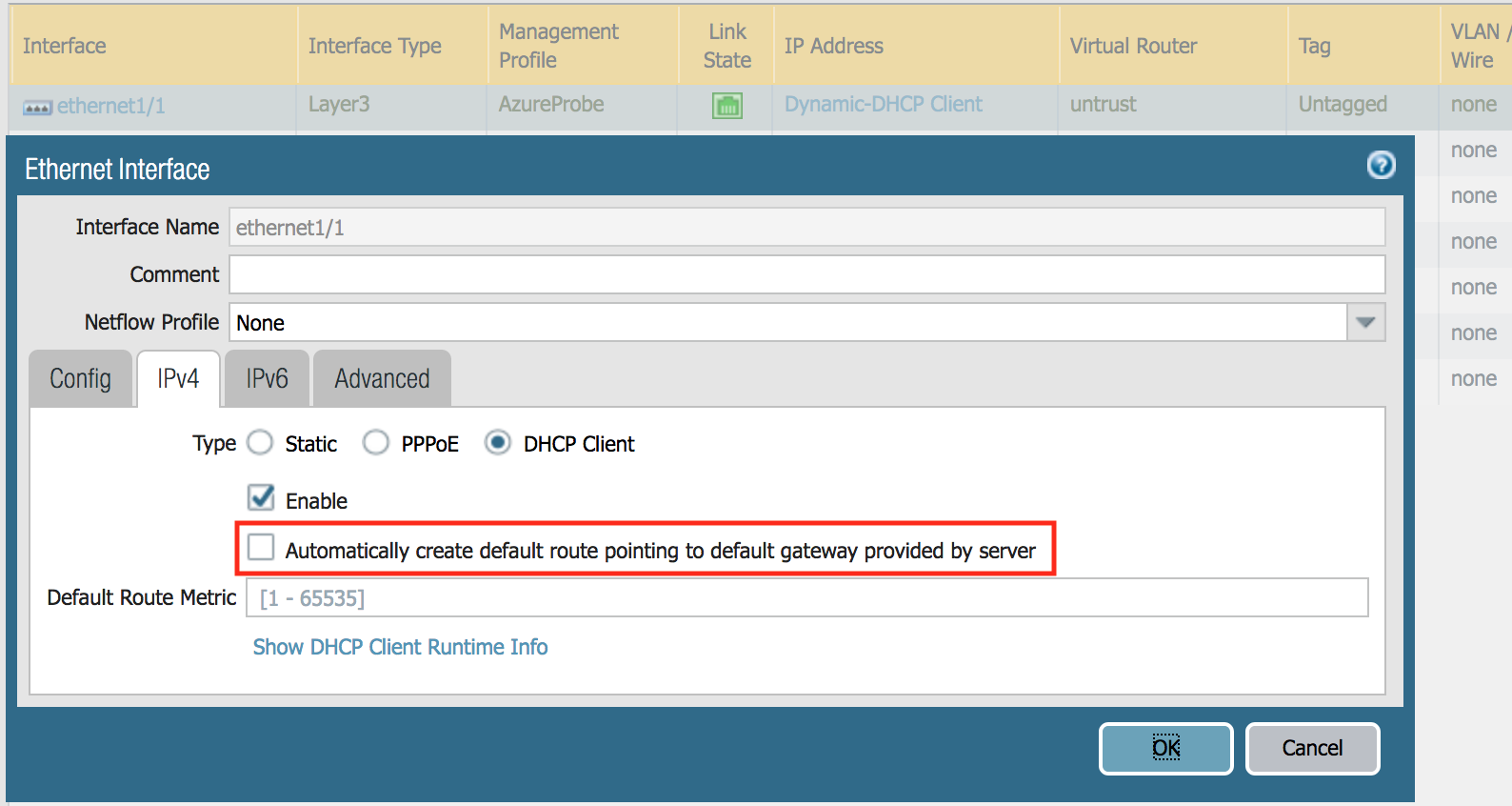


In case there is nothing created, create one Management Profile with the following settings.



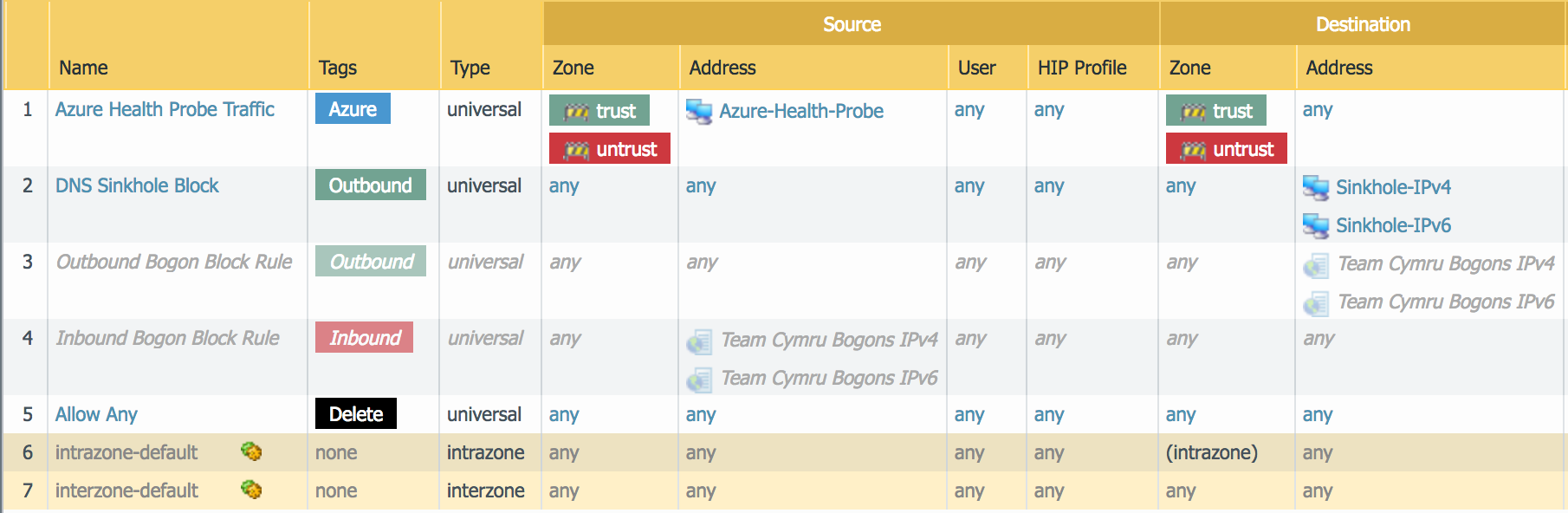
When you have created an Health Probe rule on Port 80 you have then to allow on the Interface Management Profile “HTTP”

As next verify that on all configured interfaces the option “Automatically create default route pointing to default gateway provided by server” is unchecked. If this is not the case, uncheck this and commit the configuration.

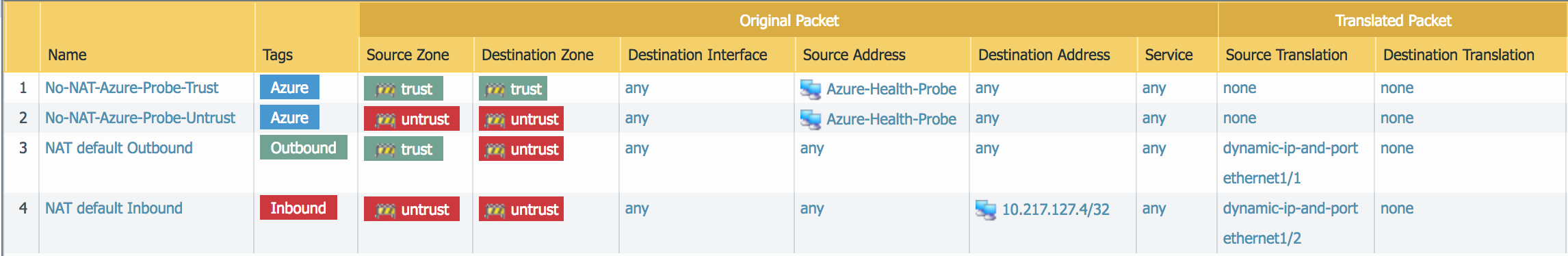


### Review Firewall NAT / Security Rules

At least you should see the following Security Rules. The “Allow Any” Security Rule is in the beginning only there to be safe that the Firewall is not blocking any traffic. This Rule should be at the end disabled or deleted.



This is the minimum what you need for your initial Setup in Azure. When you want to have special traffic to a Webserver you have to create a rule for them. This is not done Automatically.



In Case that there are not at least all Rules of them above deployed, create them by yourself and inform the Instructor about it.

### Review Firewall Health Probe Traffic

When everything is right configured you should be able to see the same traffic flow as blow. In this case is it normal that the traffic on Port 80 in incomplete. Don’t worry about this.

If your traffic doesn’t look like this below review the configuration from the previous steps or consult an Instructor.



## Configure / Test Connection to the Web Server

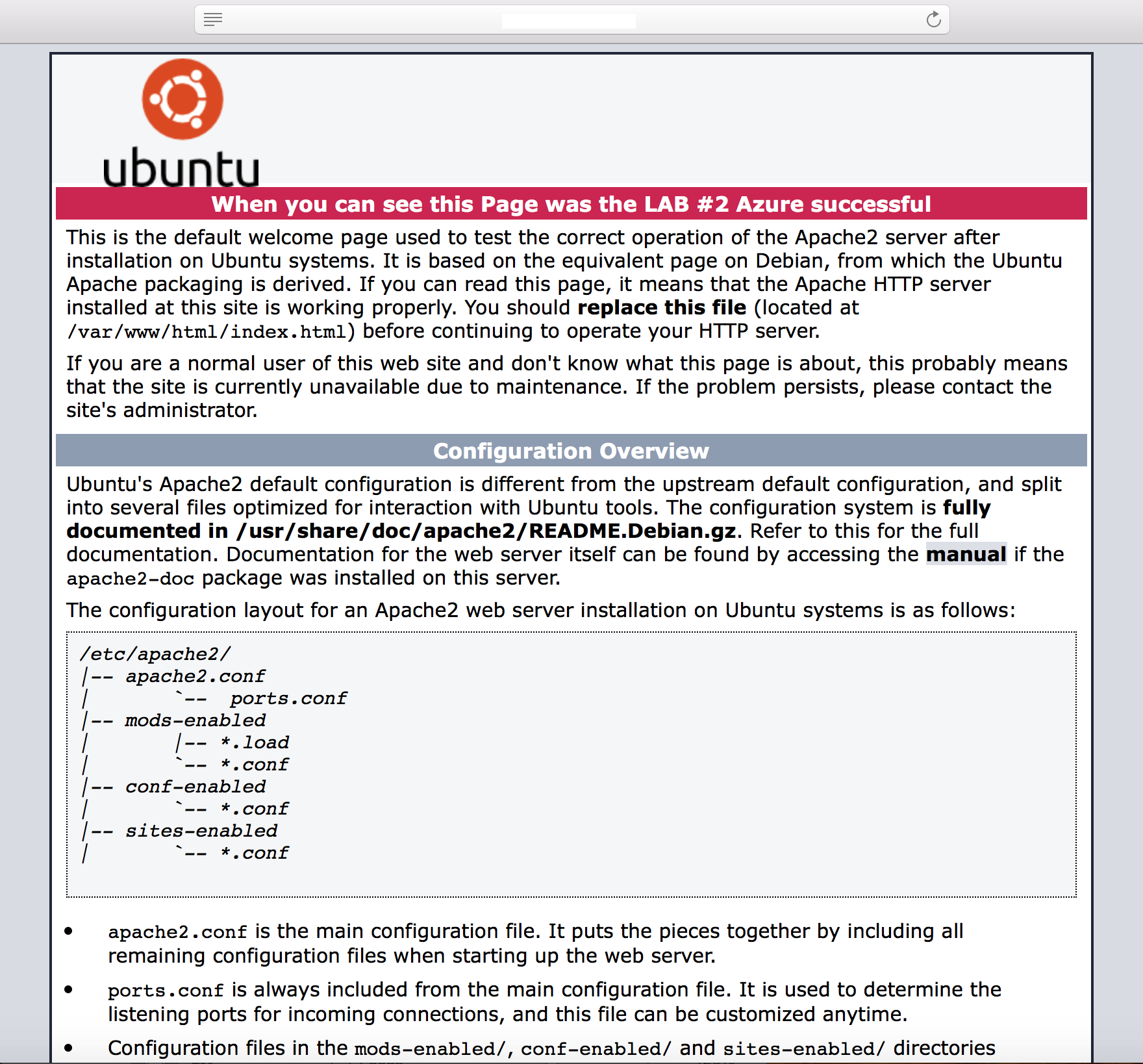
In this step we will configure at first the NAT Rules on the Firewall to access the Webserver in the Spoke Resource Group. In case you had already done the configuration of the NAT rule in the “all.yml” please review the rule.

Now we have to create/update the NAT Rules that translate the traffic on the “Untrust Interface” to the Web-Server IP address.

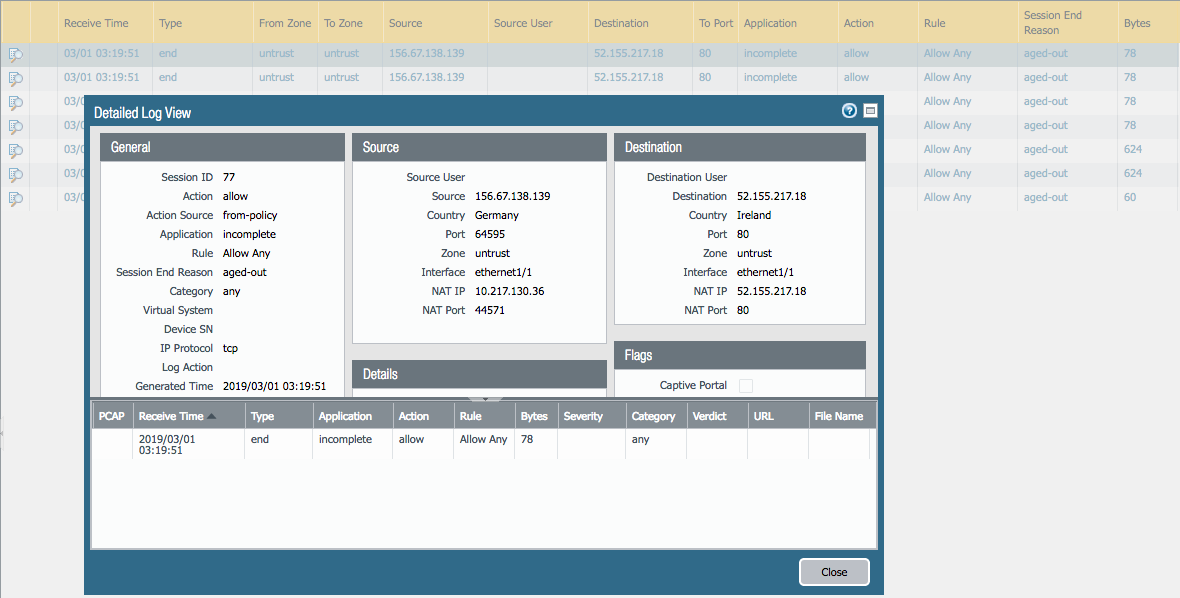
Replace the Destination address ip with your frontend IP of your Public Load balancer. In the Chapter [“Review Private/Public Load Balancer”](#_Review_Azure_Private/Public) is described where you can find this ip.



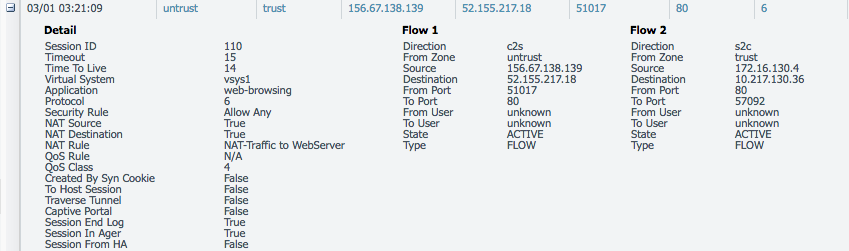
You should be now able to browse to your internal Web-Server. Type in your browser the following URL http:// <FIP-Public-LB>:80 and you should see the following page.



In the Firewall Monitor you should be able to see the traffic. See my example below

****

In the Session browser you can verify that the traffic is hitting the NAT Policy.

****

## Save and Destroy Lab Environment

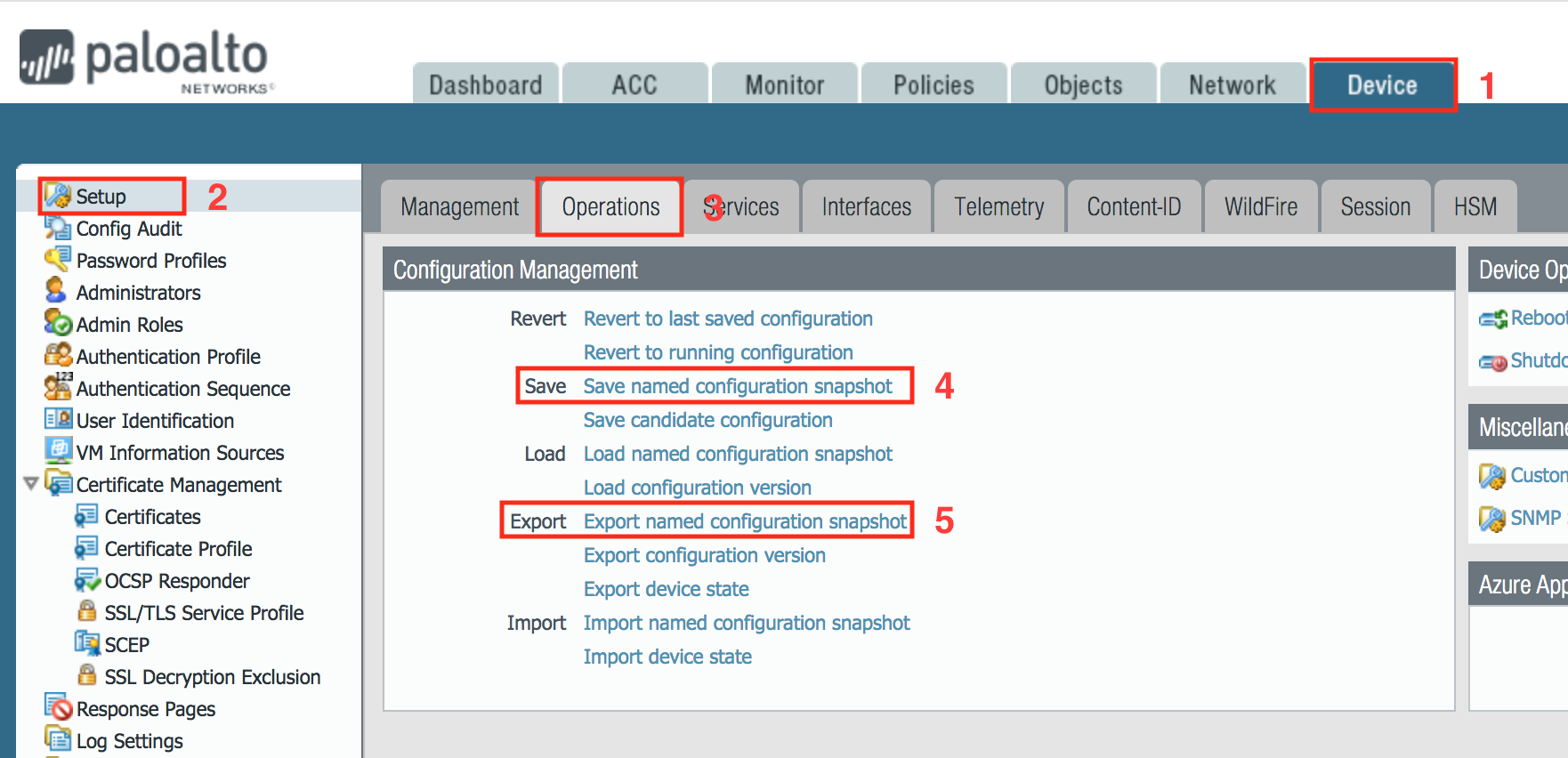
We will now Save and Export the Firewall configuration file for your documents and after that we will destroy the LAB environment.

### Save and Export Firewall Configuration

Go in the Firewall to “Device (1) 🡪 Setup (2) 🡪 Operations (3)”.

As next click on “Save named configuration Snapshot (4)”. Type a proper name and save it.

When the firewall configuration is saved click on “Export named configuration snapshot (5)” and select the file what you have in the previous step created.

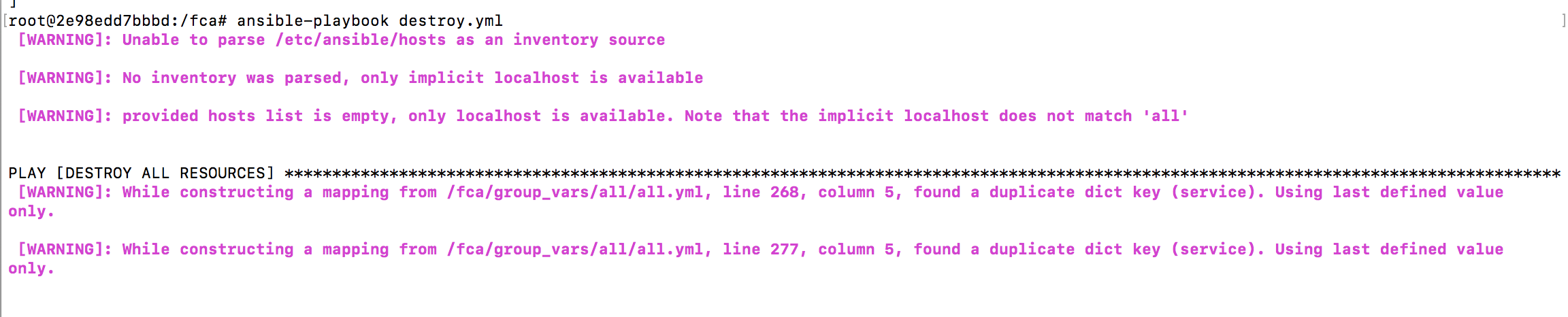


### Delete Azure Environment

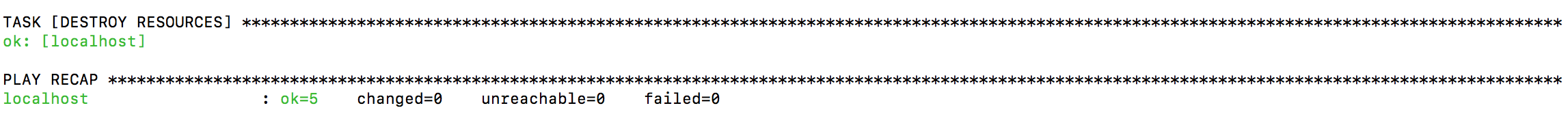
Now we can destroy the previous build environment, when you had saved and export the Firewall Configuration snapshot. To destroy the Azure environment, type the following command: ***“ansible-playbook destroy.yml”.***

You will get prompt if you want to back up the **“terraform.tfstate”** files. You have to type **“NO”**

See below the example:



You will see the following output when the deletion of the Azure Environment was successful.



**End of LAB #2 AZURE**