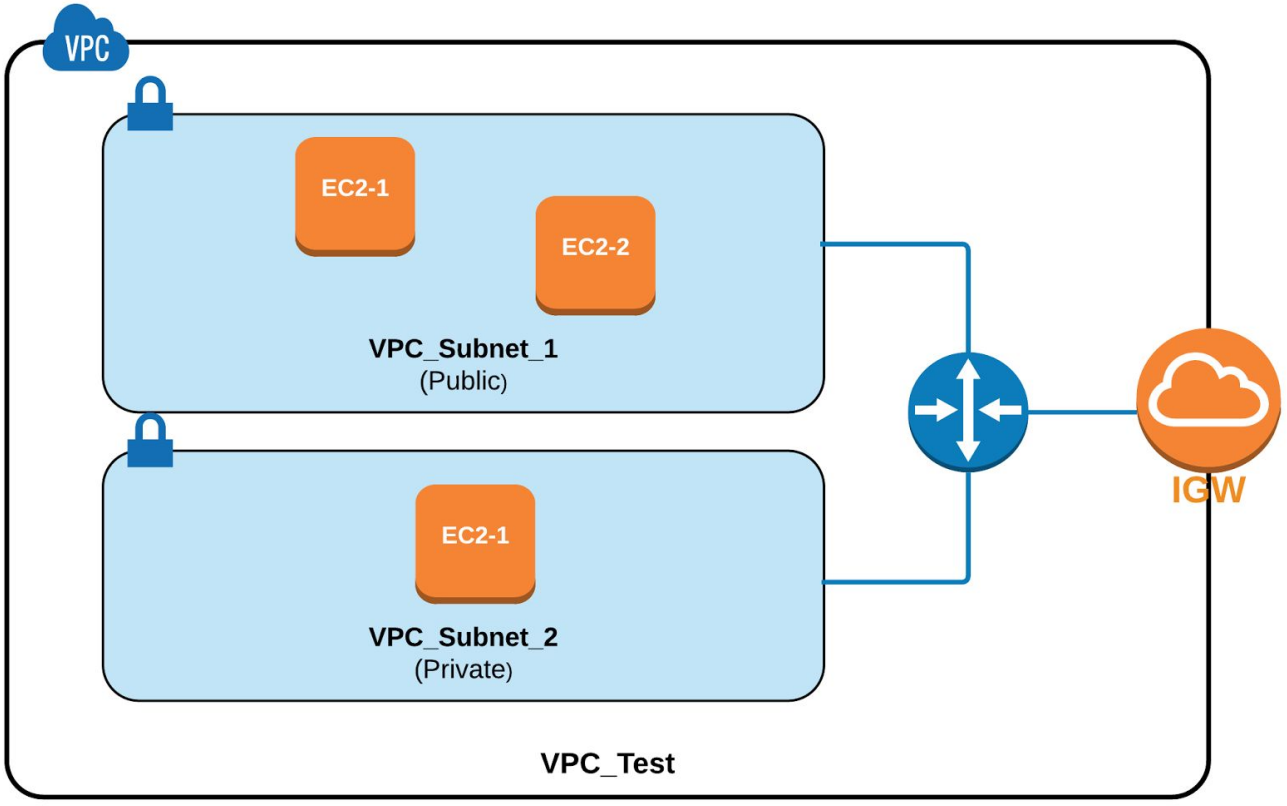
MODULE 4 – LAB EXERCISES

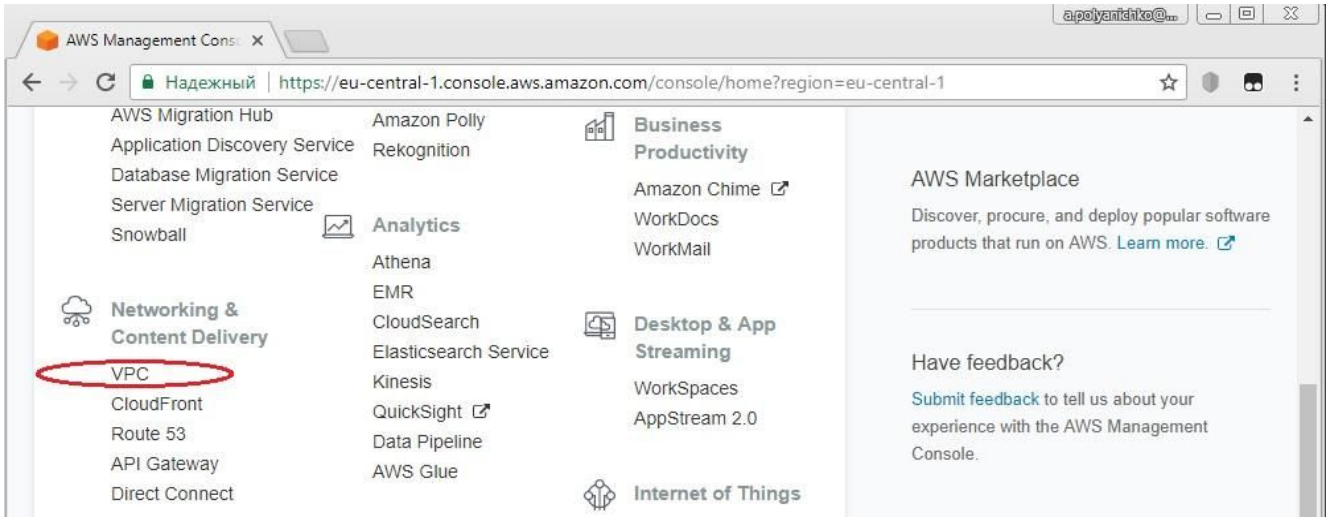
On this lab you will get a practice in Virtual Private Cloud (VPC) with Amazon Linux EC2 Instances management and operation.

During the practice we will build-up the network infrastructure similar to the one shown below:

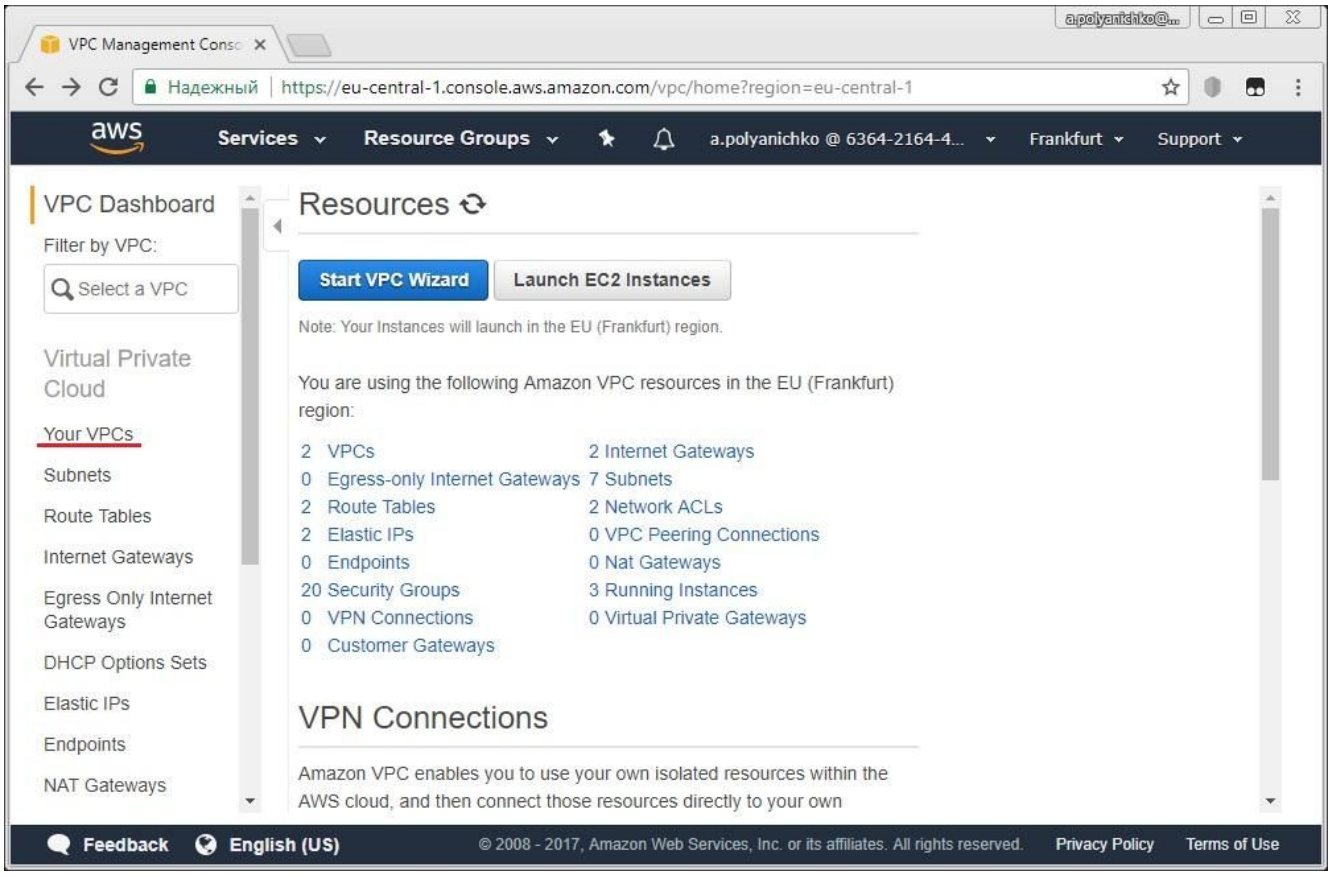


1. VPC Getting Started: Virtual Network, Subnet and IGW

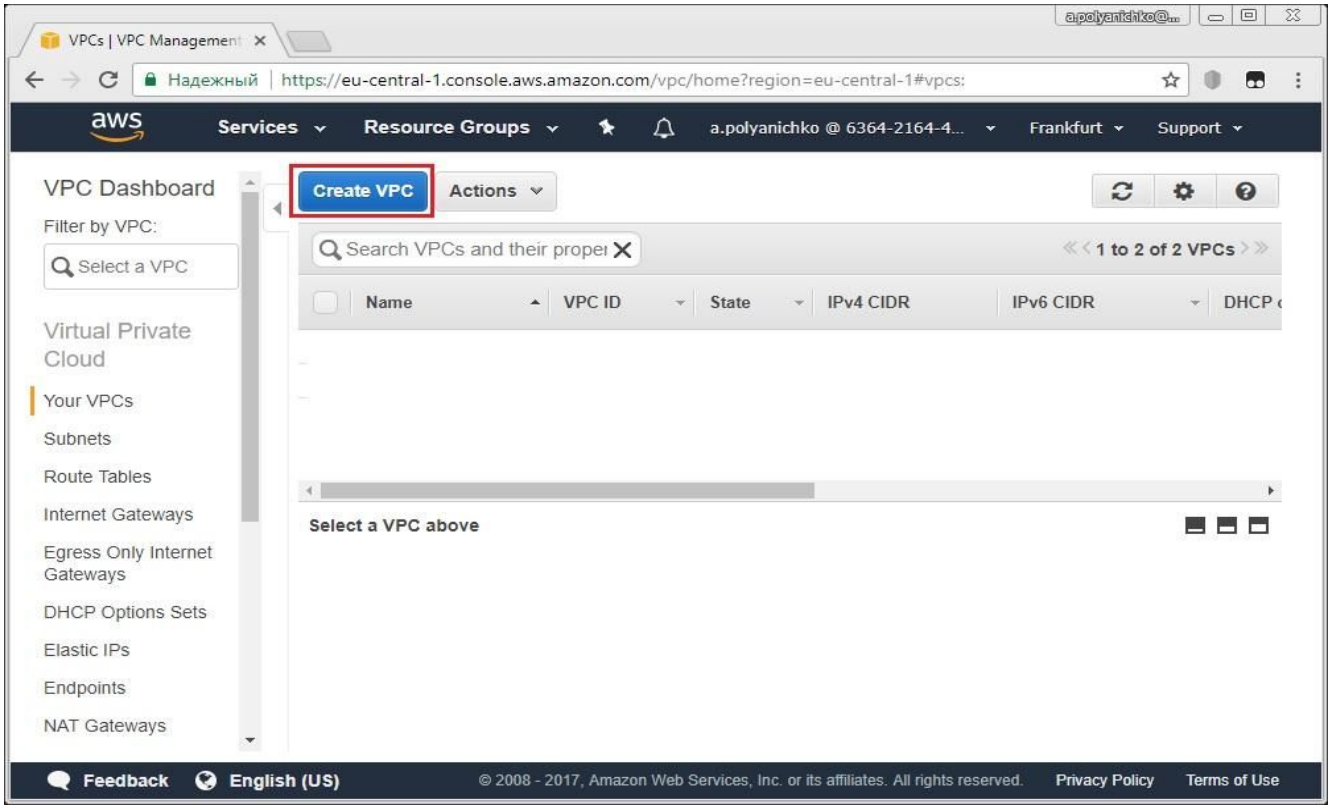
Open AWS Management Console, find and click “VPC” link to access the service management:



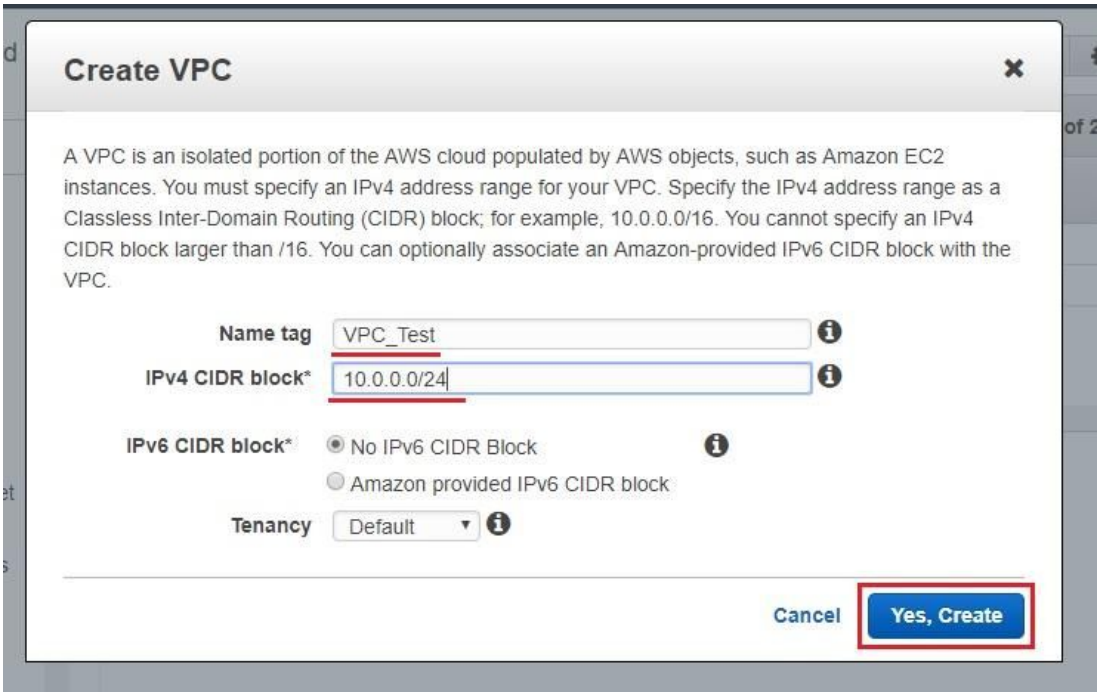
VPC Dashboard page will be opened:



Click on “Your VPCs” link at left-side menu ribbon to open VPC explorer page:

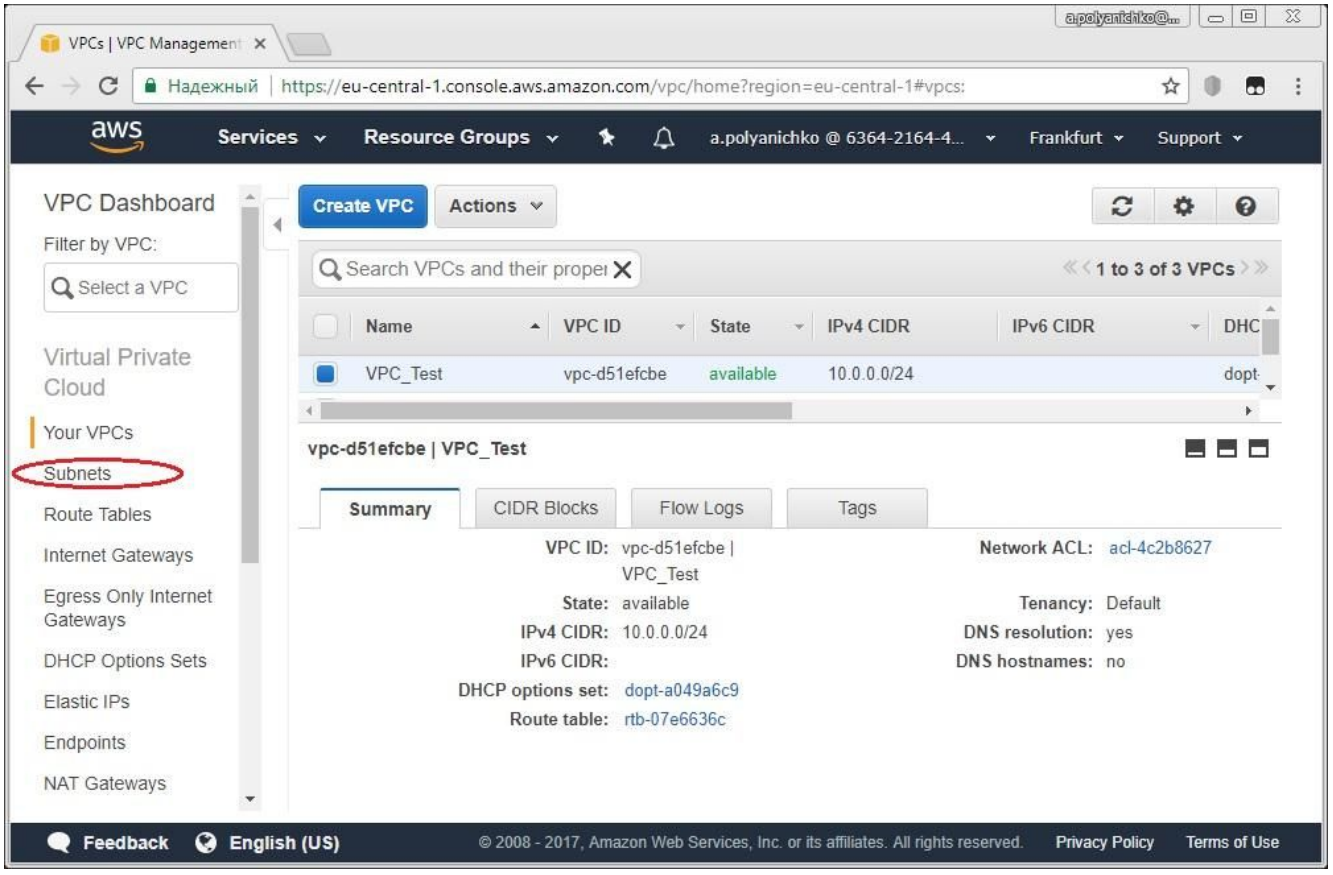


Click on “Create VPC” button and then specify VPC parameters:

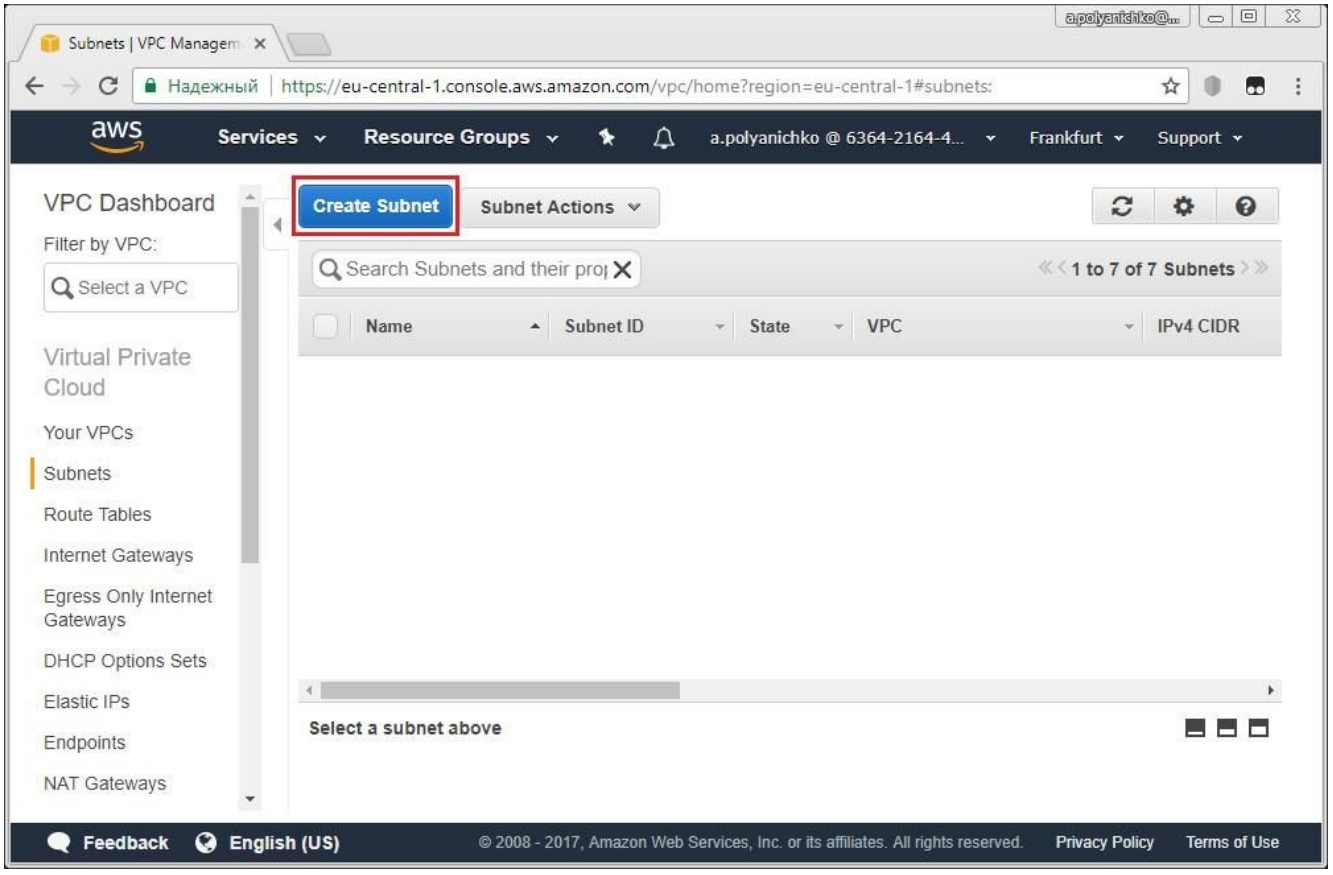


In the given exercise we are creating VPC named “VPC_Test” and we are assigning CIDR block 10.0.0.0/24 (256 of IP addresses are available).

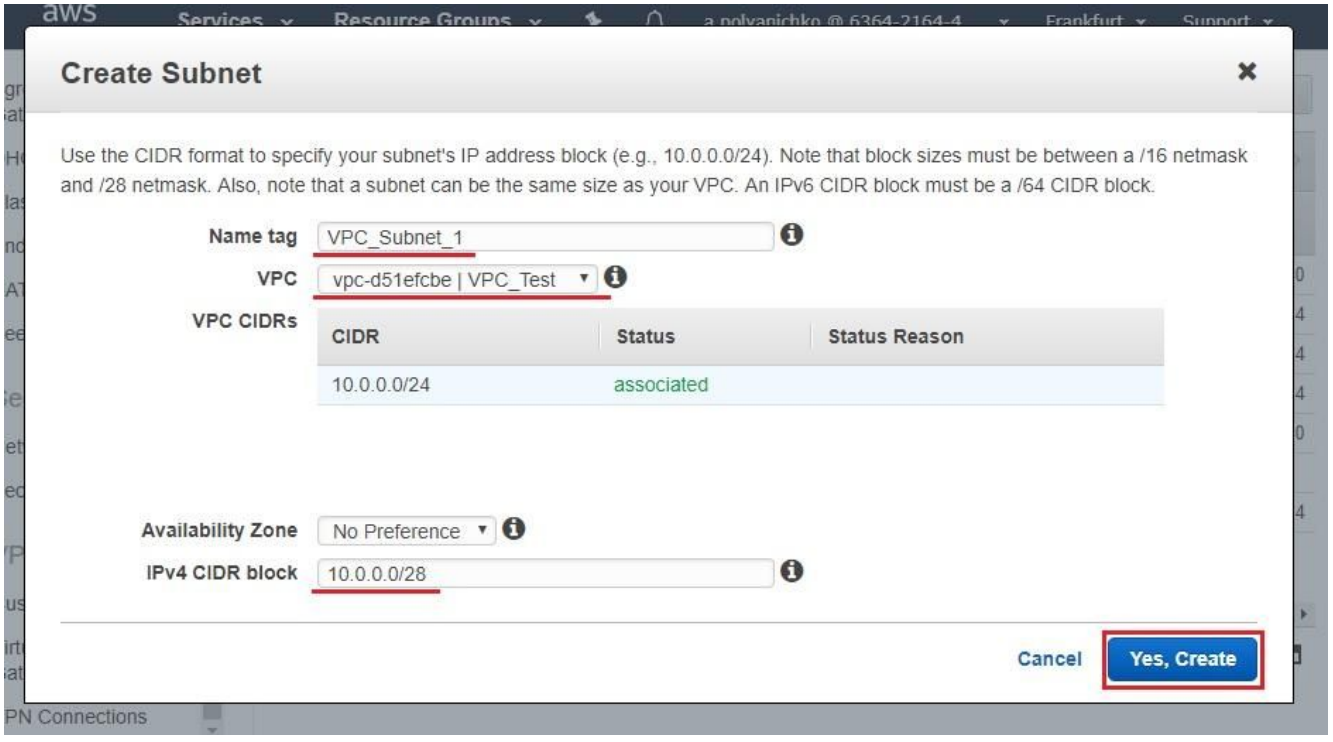
Click on “Yes, Create” button and find your VPC on the next page, then click on “Subnets” link at the left-side menu ribbon:



Click on “Create Subnet” button:



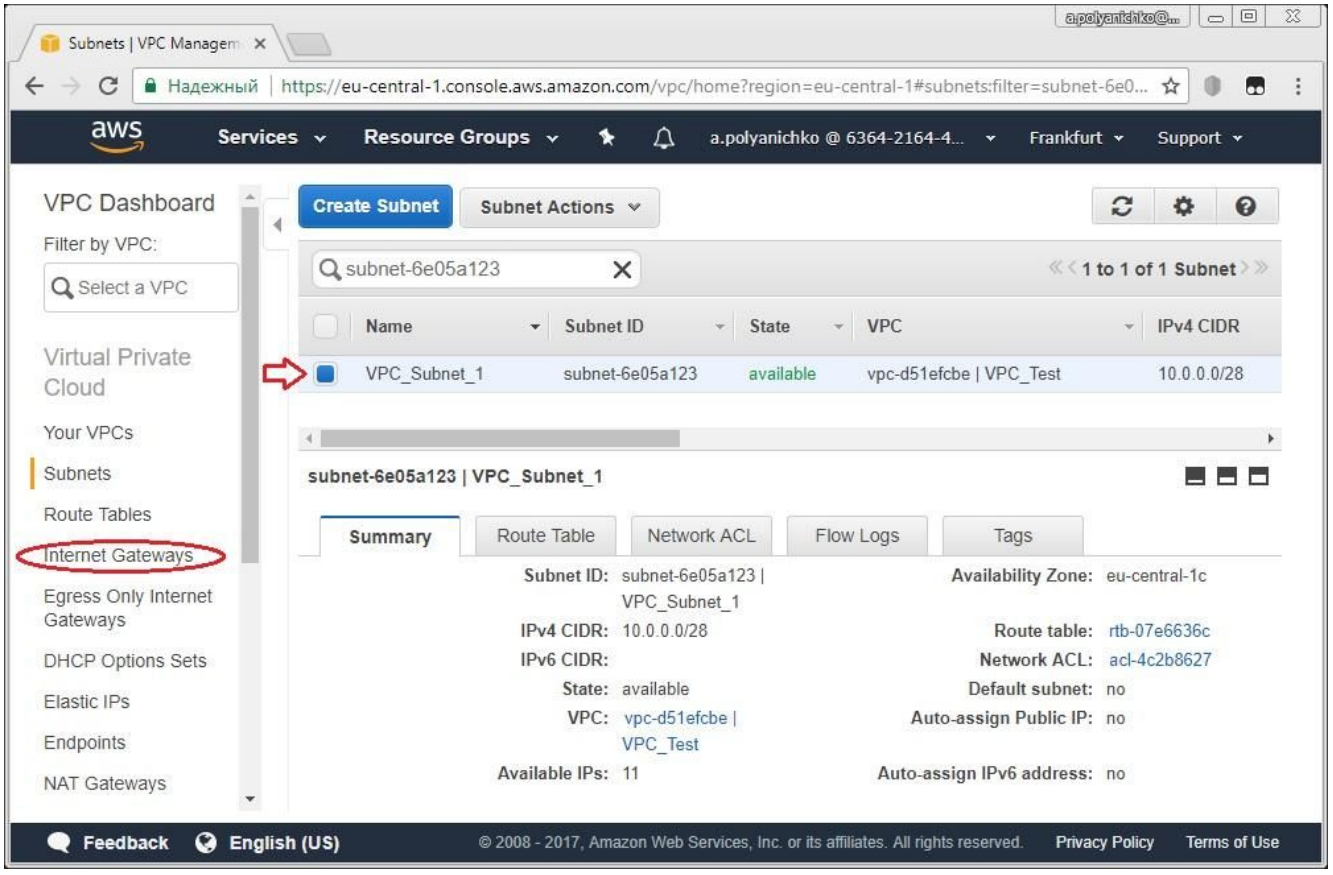
and then specify subnet parameters:



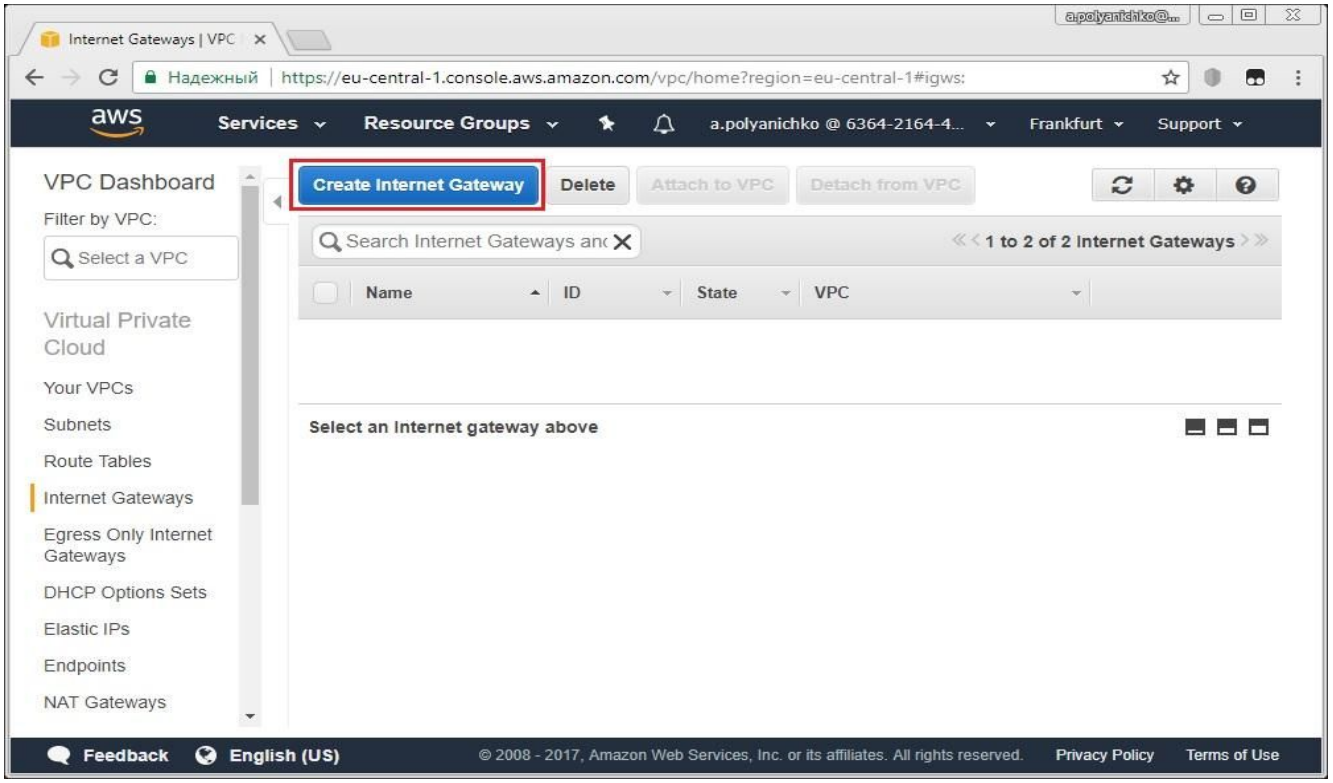
In the given exercise we are creating subnet named “VPC_Subnet_1” within “VPC_Test” VPC and we are ordering CIDR block 10.0.0.0/28 (16 of IP addresses are available for subnet).

Click on “Yes, Create” button and find your subnet on next page, then click on “Internet Gateways”

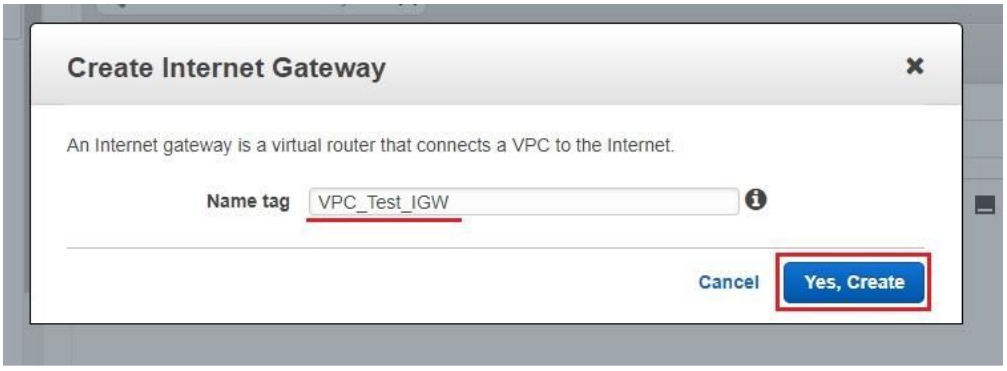
link at left-side menu ribbon:



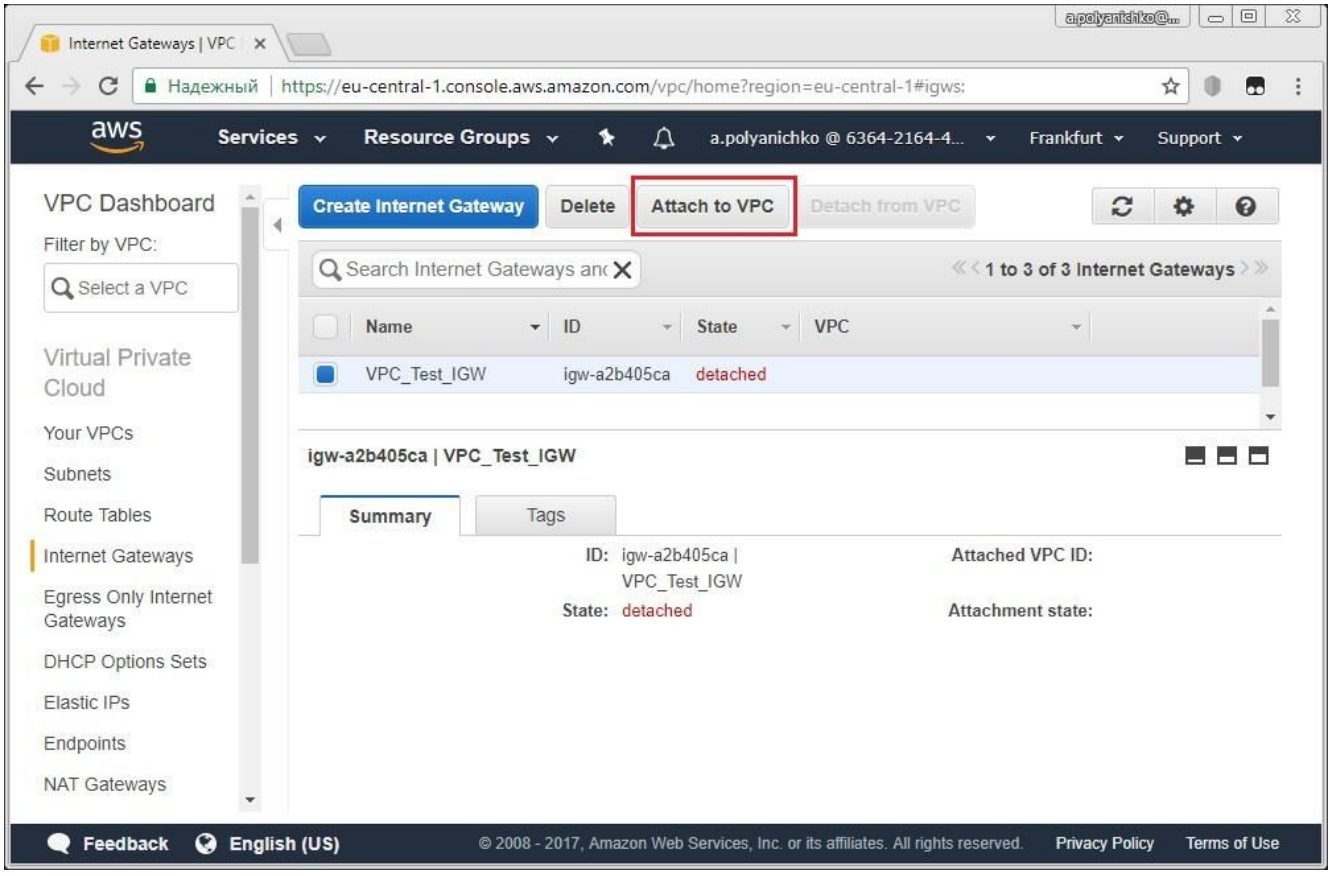
Click on “Create Internet Gateway” button:



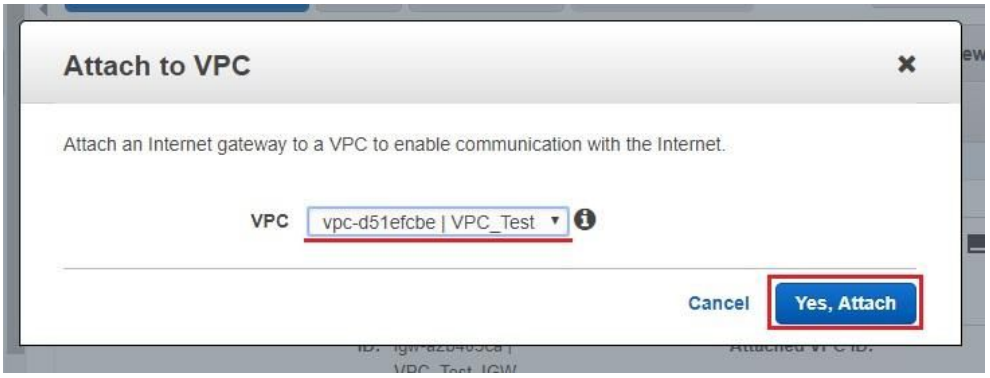
specify IGW name and then click “Yes, Create” button:



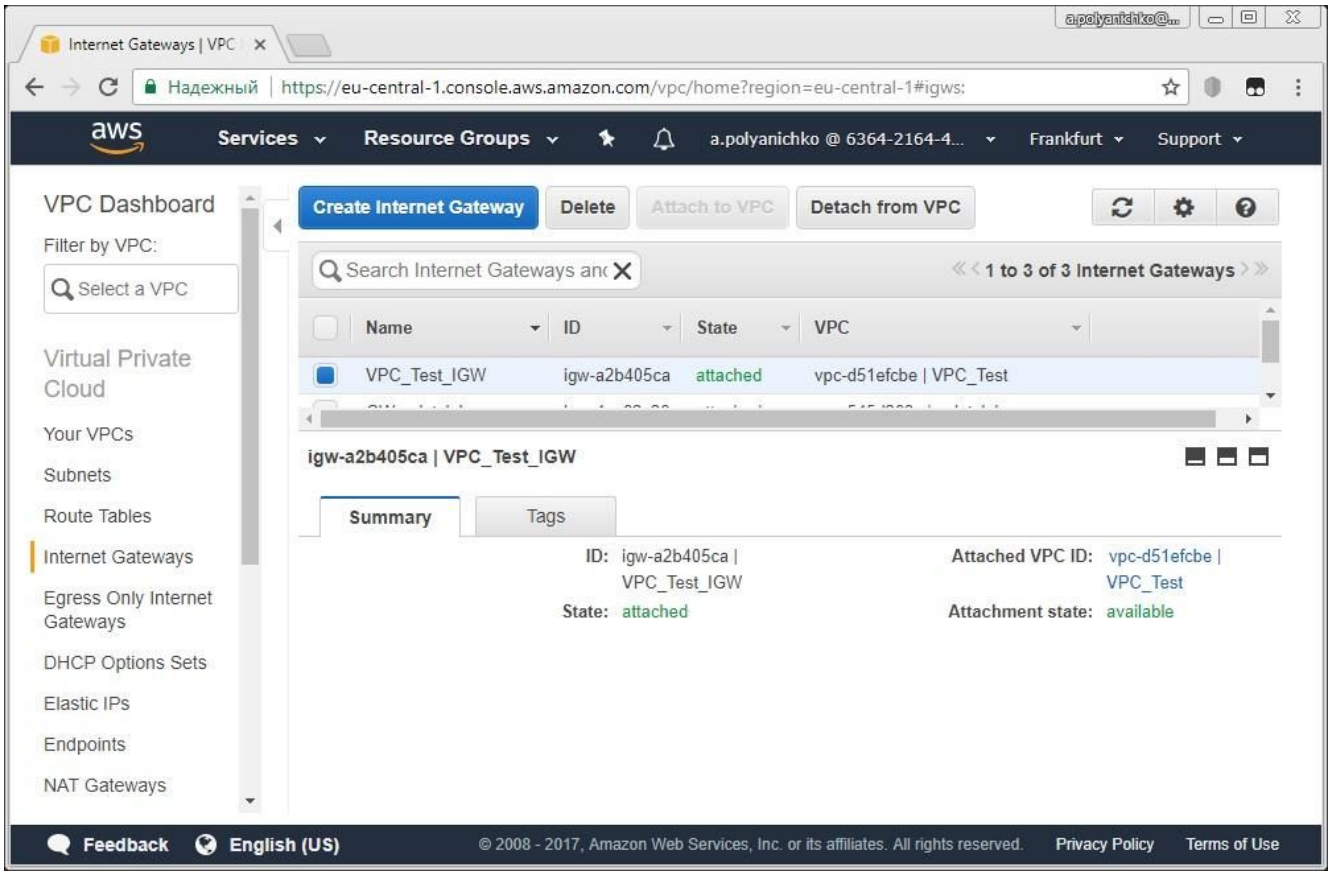
On the next page select your IGW and then click on “Attach to VPC” button:



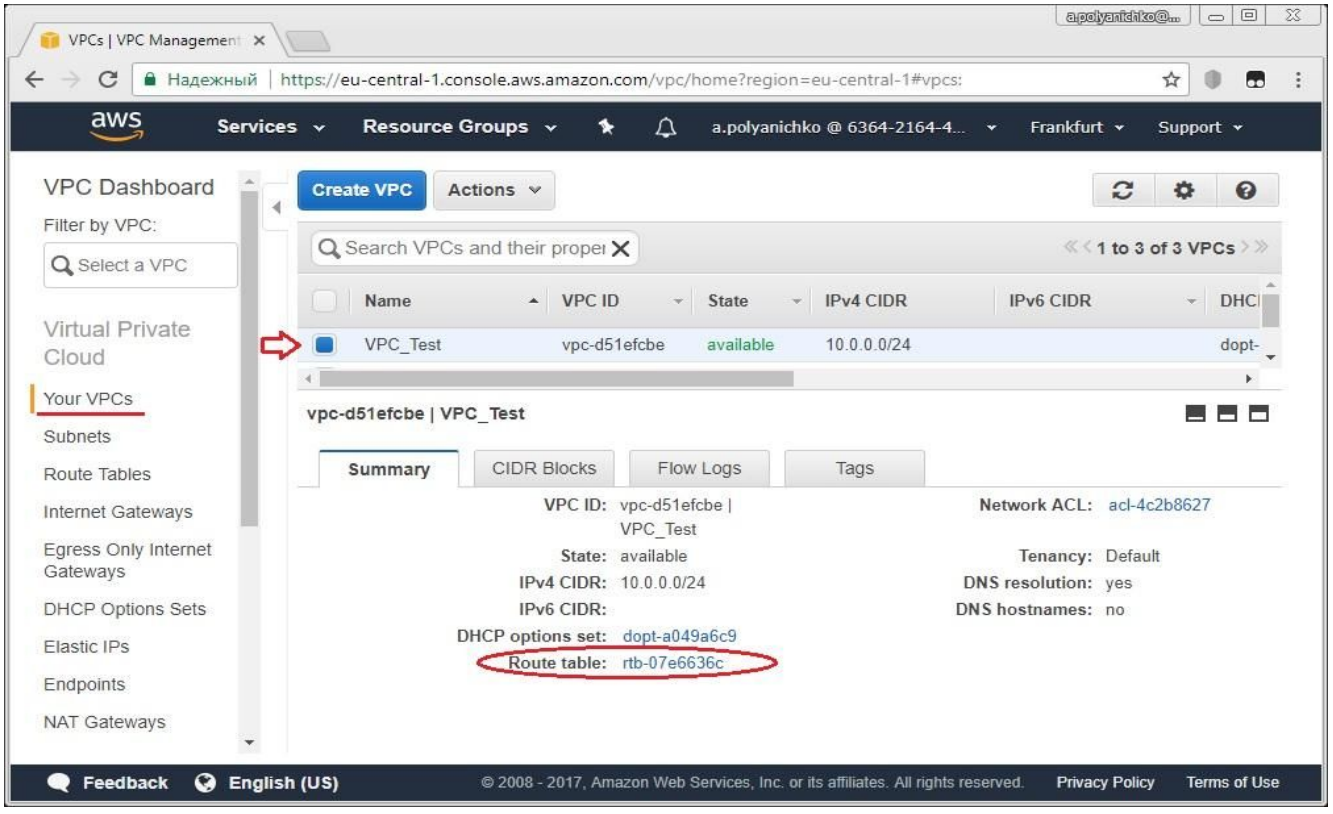
Select your VPC from scroll-down list and then click on “Yes, Attach” button:



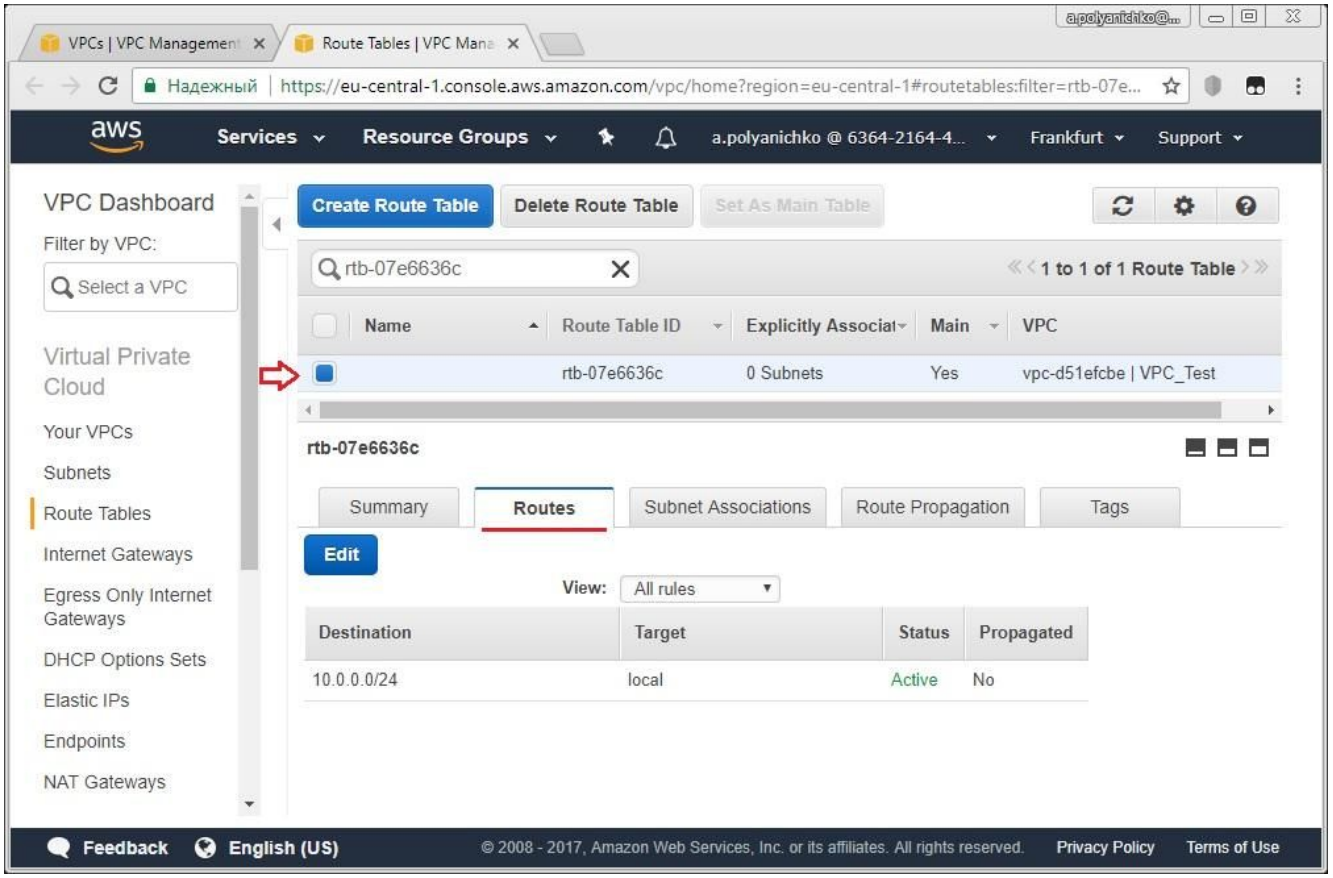
Finally you can see your IGW is attached to VPC:



Revert back to your VPC and check main (default) routing table by clicking on Route table name link:



Check the content of default routing table:

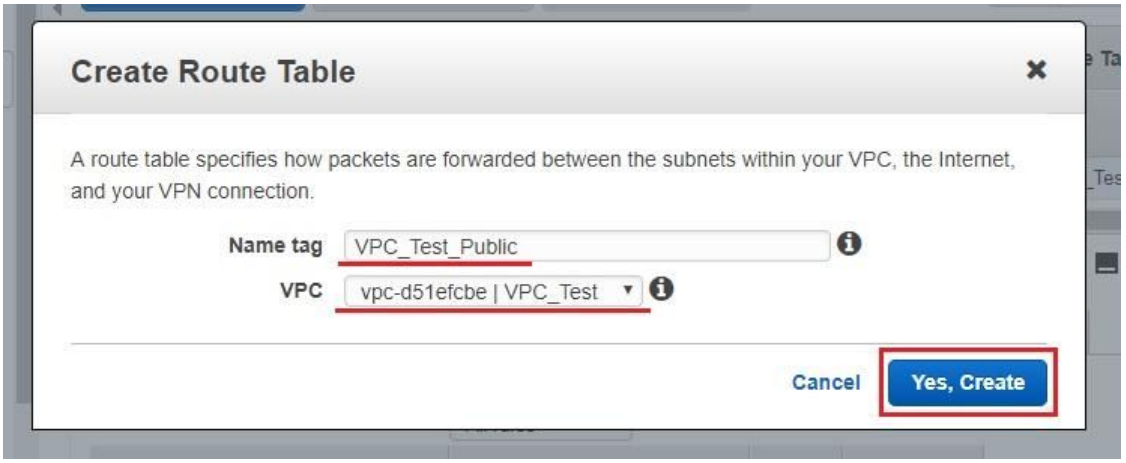


As you can see, the local route only is specified by default for each subnet of our VPC and so we need

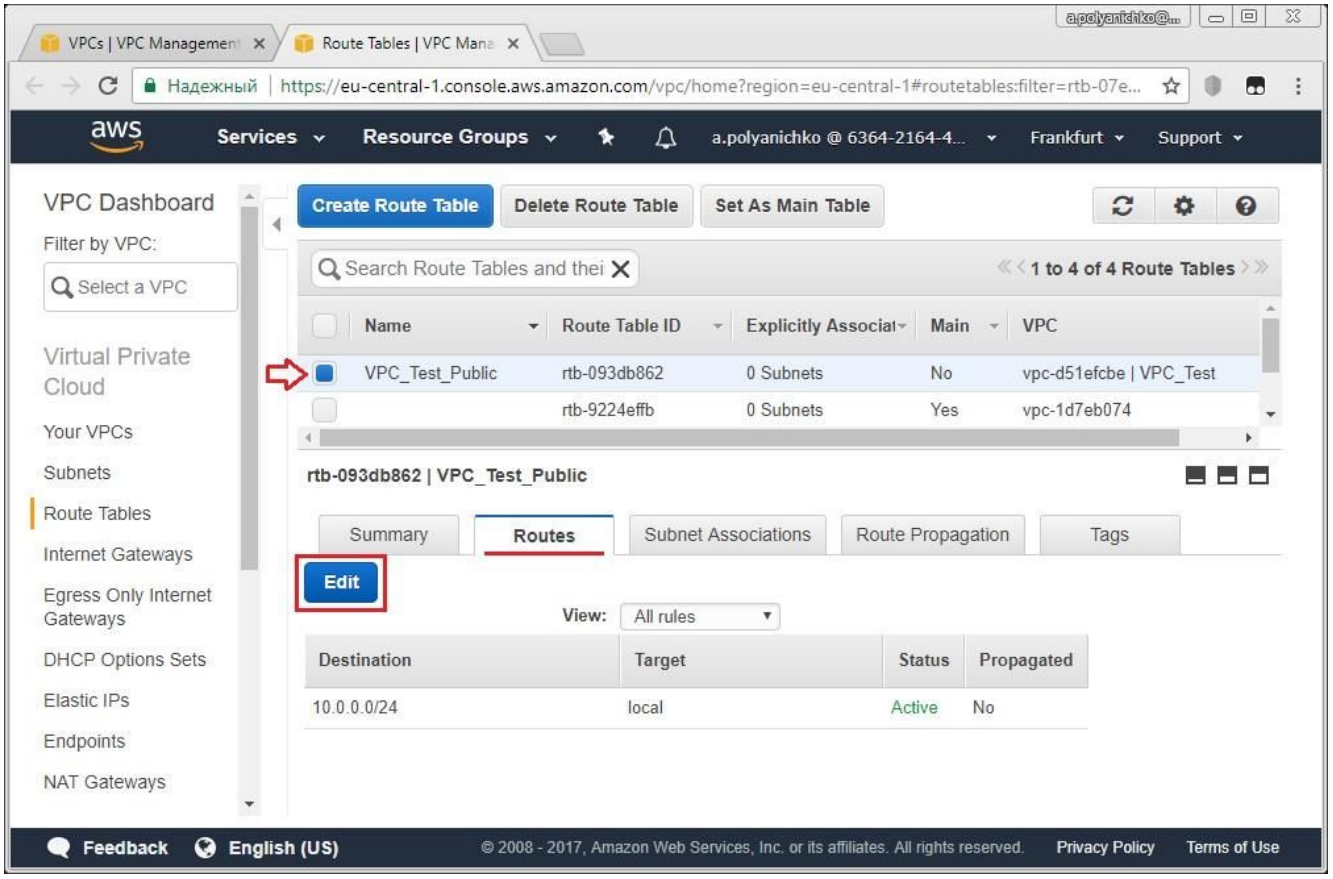
to create customer route table with IGW access granted.

Click on “Create Route Table” button at the top: 

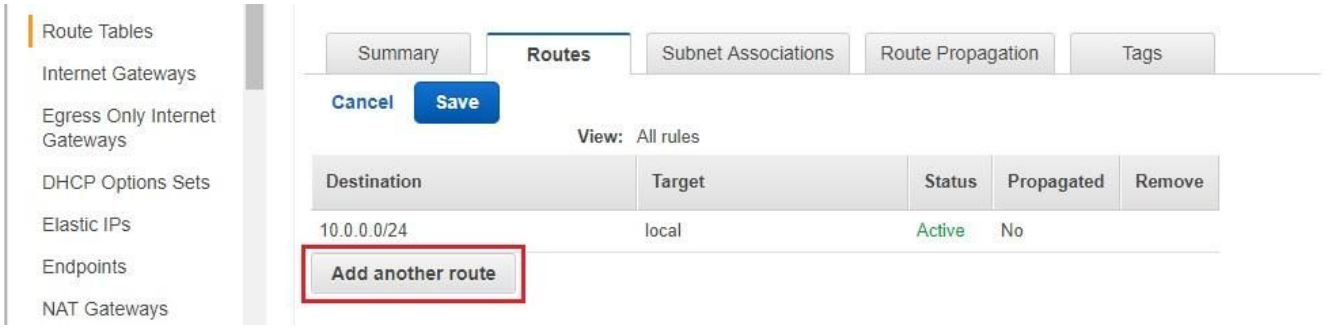
and then specify Name tag for new routing table as shown:



Click on “Yes, Create” button, then select your routing table in the list, jump to “Routes” tab and click on “Edit” button below:

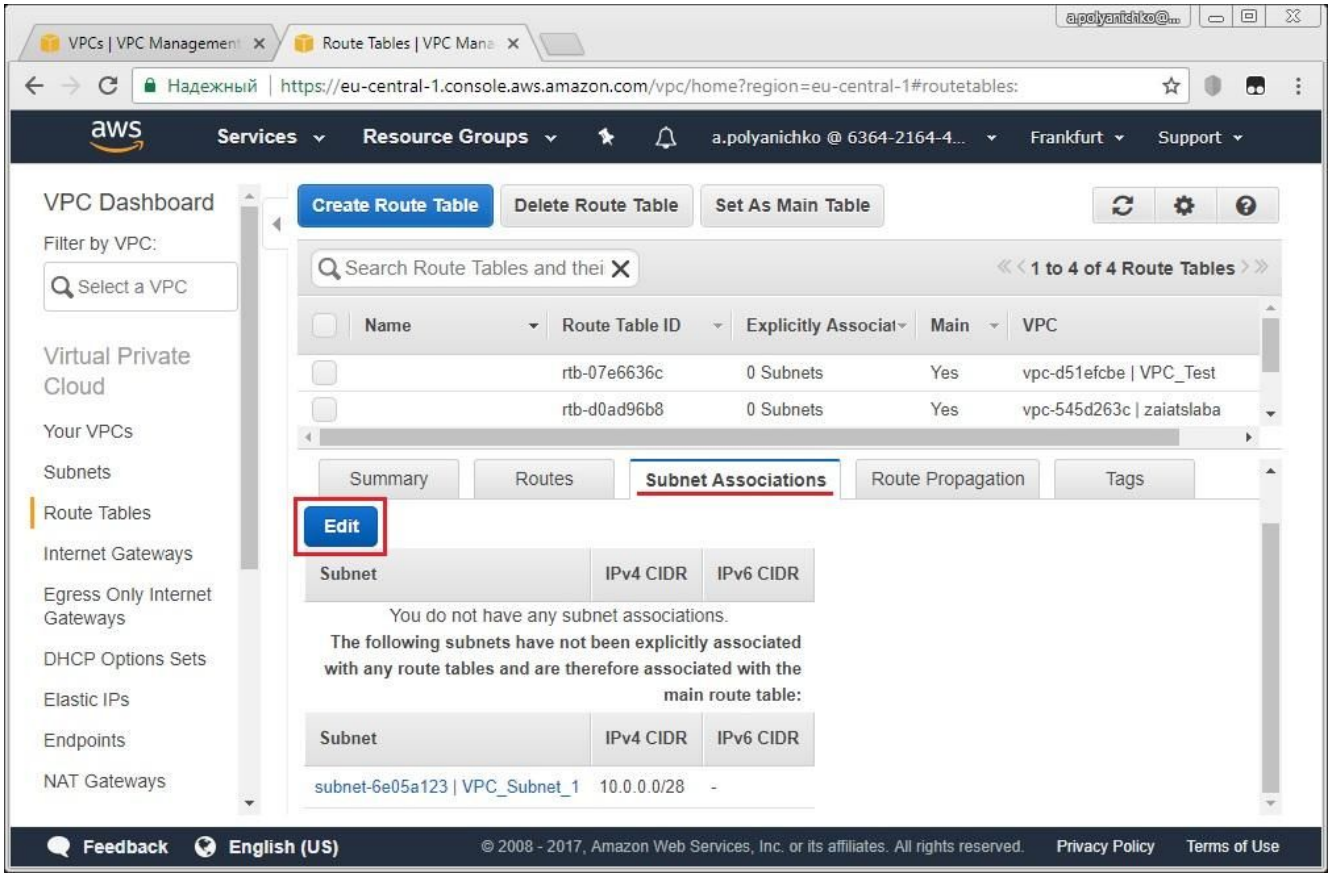


Click on “Add another route” button:

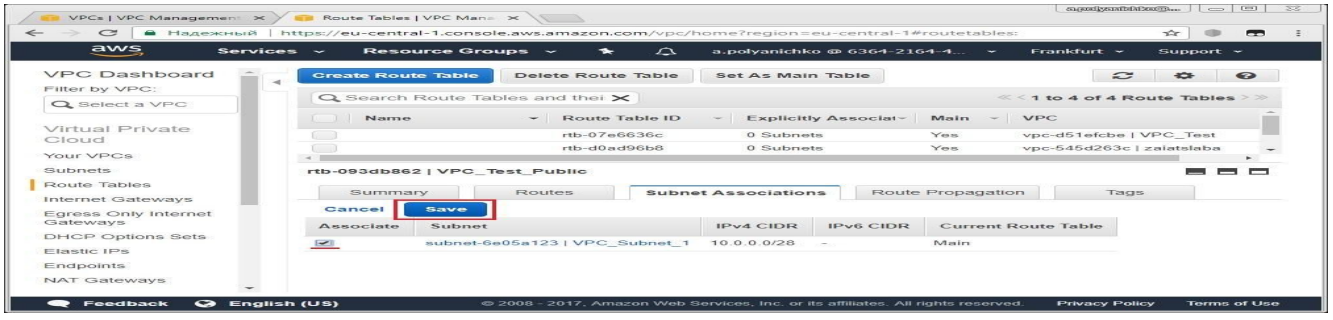


and specify the route to your IGW as shown below (IGW name is selectable from drop-down list); then click on “Save” button:

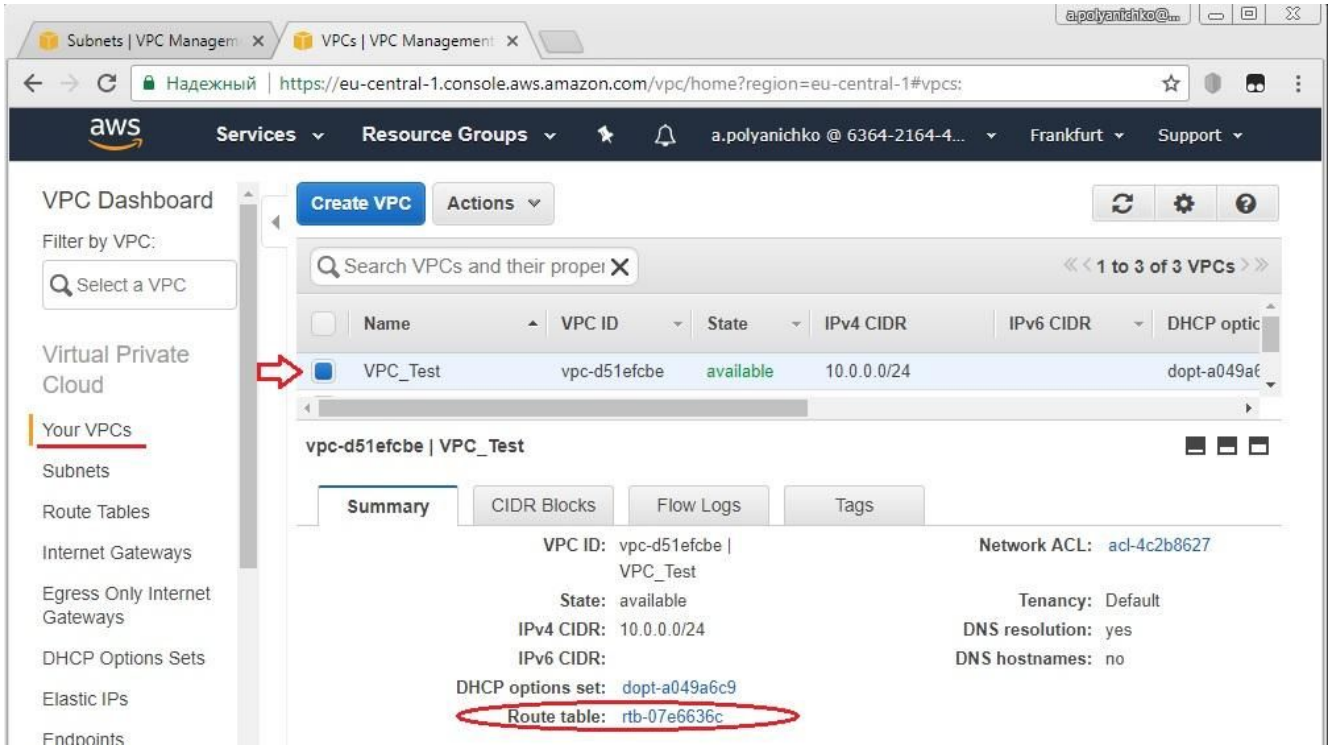
Ensure that the route to IGW is present under Routes tab, then switch to “Subnet Associations” tab and click on “Edit” button:

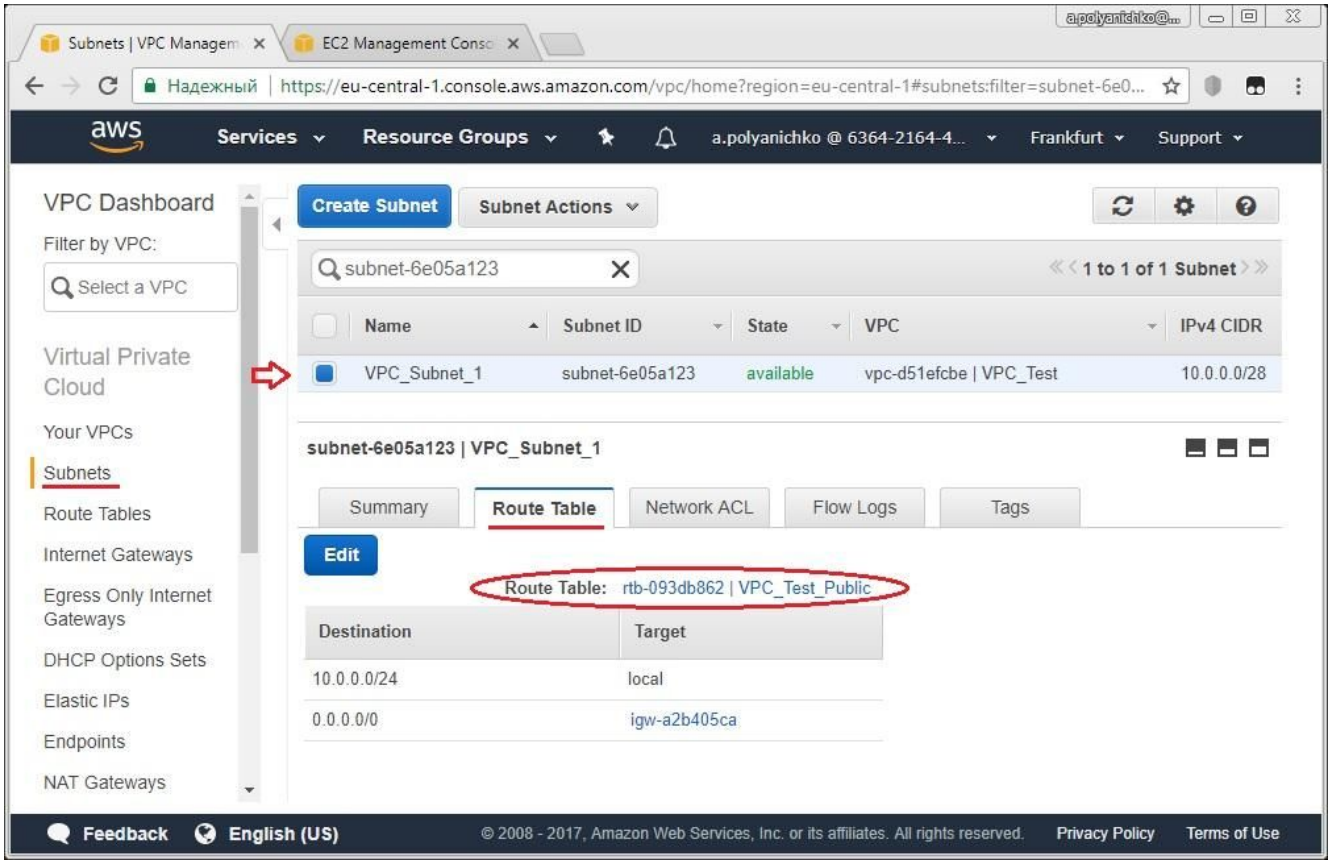


Associate your VPC_Subnet_1 with the route table by selecting its checkbox and then click on “Save” button:

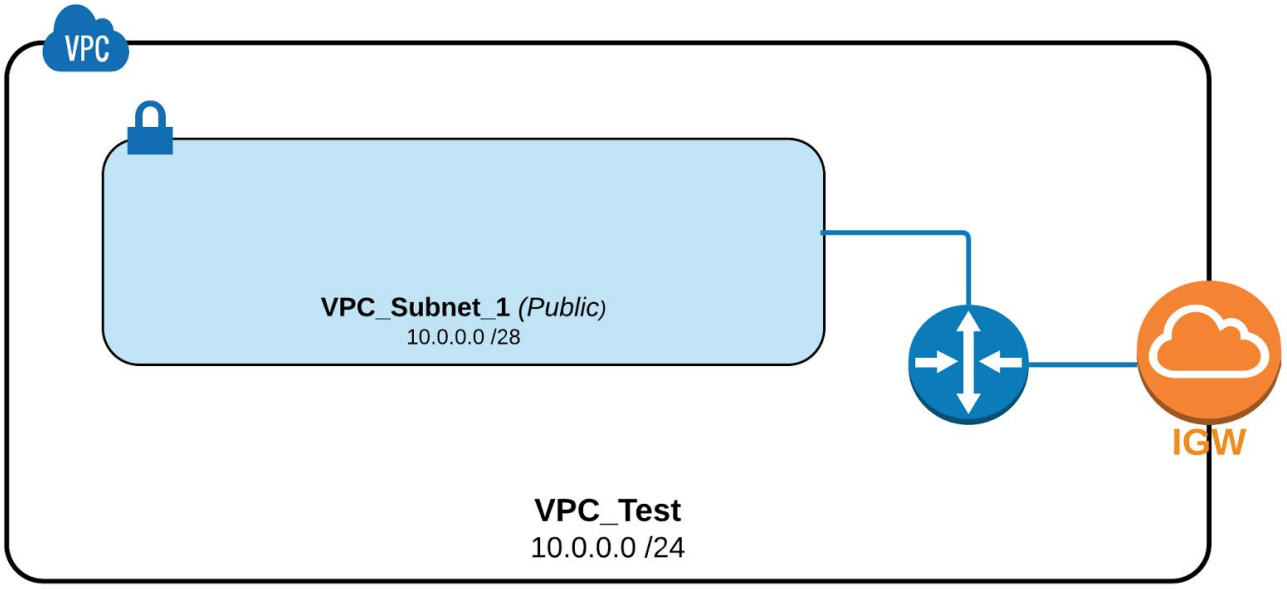


Now you may ensure that at the moment your VPC and your Subnet_1 use different routing tables:


Subnets | VPC Managem
VPCs | VPC Management
Надежный | https://eu-central-1.console.aws.amazon.com/vpc/home?region=eu-central-1#vpcs:
aws
Services
Resource Groups
a.polyanichko @ 6364-2164-4...
Frankfurt
Support
VPC Dashboard
Filter by VPC:
Select a VPC
Virtual Private Cloud
Your VPCs
Subnets
Route Tables
Internet Gateways
Egress Only Internet Gateways
DHCP Options Sets
Elastic IPs
Endpoints
Create VPC
Actions
Search VPCs and their proper
1 to 3 of 3 VPCs
Name
VPC ID
State
IPv4 CIDR
IPv6 CIDR
DHCP optic
VPC_Test
vpc-d51efcbe
available
10.0.0.0/24
dopt-a049a6
vpc-d51efcbe | VPC_Test
Summary
CIDR Blocks
Flow Logs
Tags
VPC ID: vpc-d51efcbe | VPC_Test
State: available
IPv4 CIDR: 10.0.0.0/24
IPv6 CIDR:
DHCP options set: dopt-a049a6c9
Route table: rtb-07e6636c
Network ACL: acl-4c2b8627
Tenancy: Default
DNS resolution: yes
DNS hostnames: no


Subnets | VPC Managem
EC2 Management Conso
Надежный | https://eu-central-1.console.aws.amazon.com/vpc/home?region=eu-central-1#subnets:filter=subnet-6e0...
aws
Services
Resource Groups
a.polyanichko @ 6364-2164-4...
Frankfurt
Support
VPC Dashboard
Filter by VPC:
Select a VPC
Virtual Private Cloud
Your VPCs
Subnets
Route Tables
Internet Gateways
Egress Only Internet Gateways
DHCP Options Sets
Elastic IPs
Endpoints
NAT Gateways
Create Subnet
Subnet Actions
subnet-6e05a123
1 to 1 of 1 Subnet
Name
Subnet ID
State
VPC
IPv4 CIDR
VPC_Subnet_1
subnet-6e05a123
available
vpc-d51efcbe | VPC_Test
10.0.0.0/28
subnet-6e05a123 | VPC_Subnet_1
Summary
Route Table
Network ACL
Flow Logs
Tags
Edit
Route Table: rtb-093db862 | VPC_Test_Public
Destination
Target
10.0.0.0/24
local
0.0.0.0/0
igw-a2b405ca
Feedback
English (US)
© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

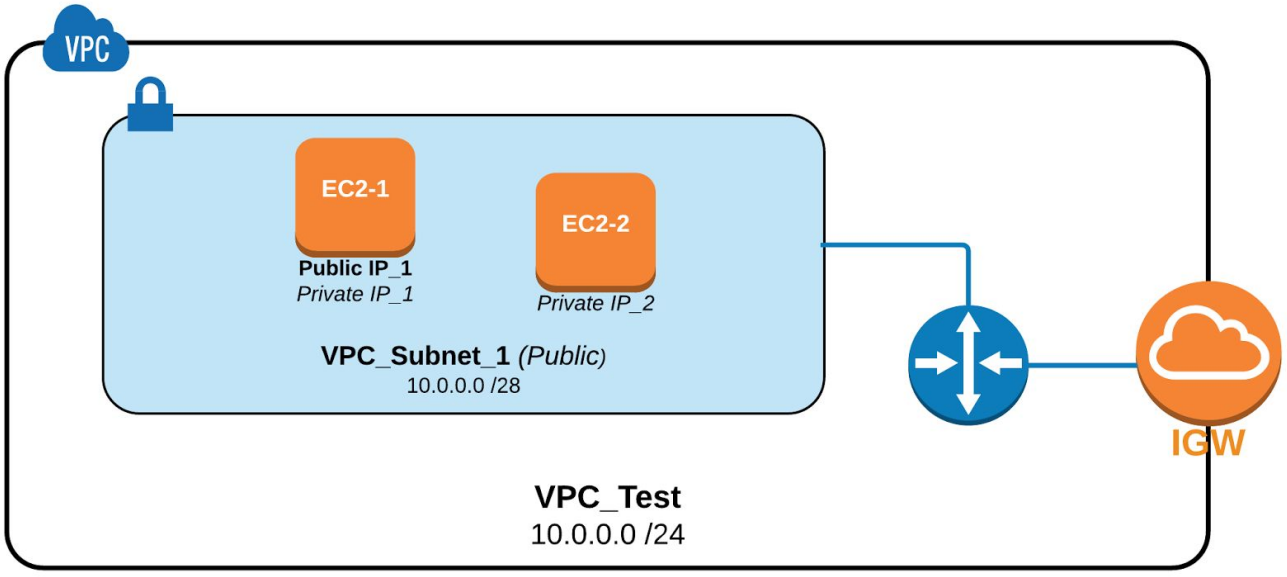
At the moment we have created the basic of our network infrastructure:



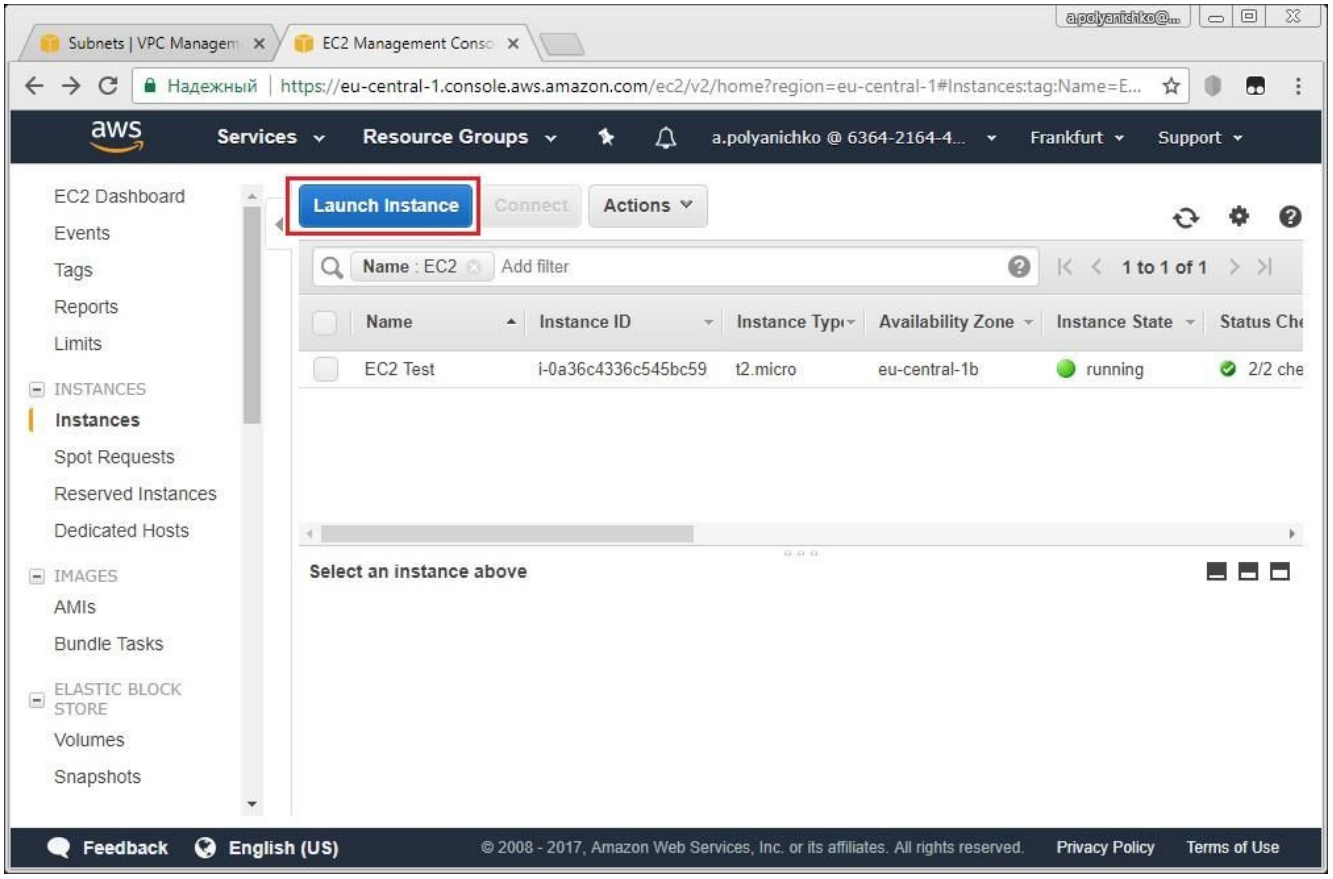
and we may add EC2 instances in the subnet.

2. Adding public and private EC2 instances to VPC in public subnet

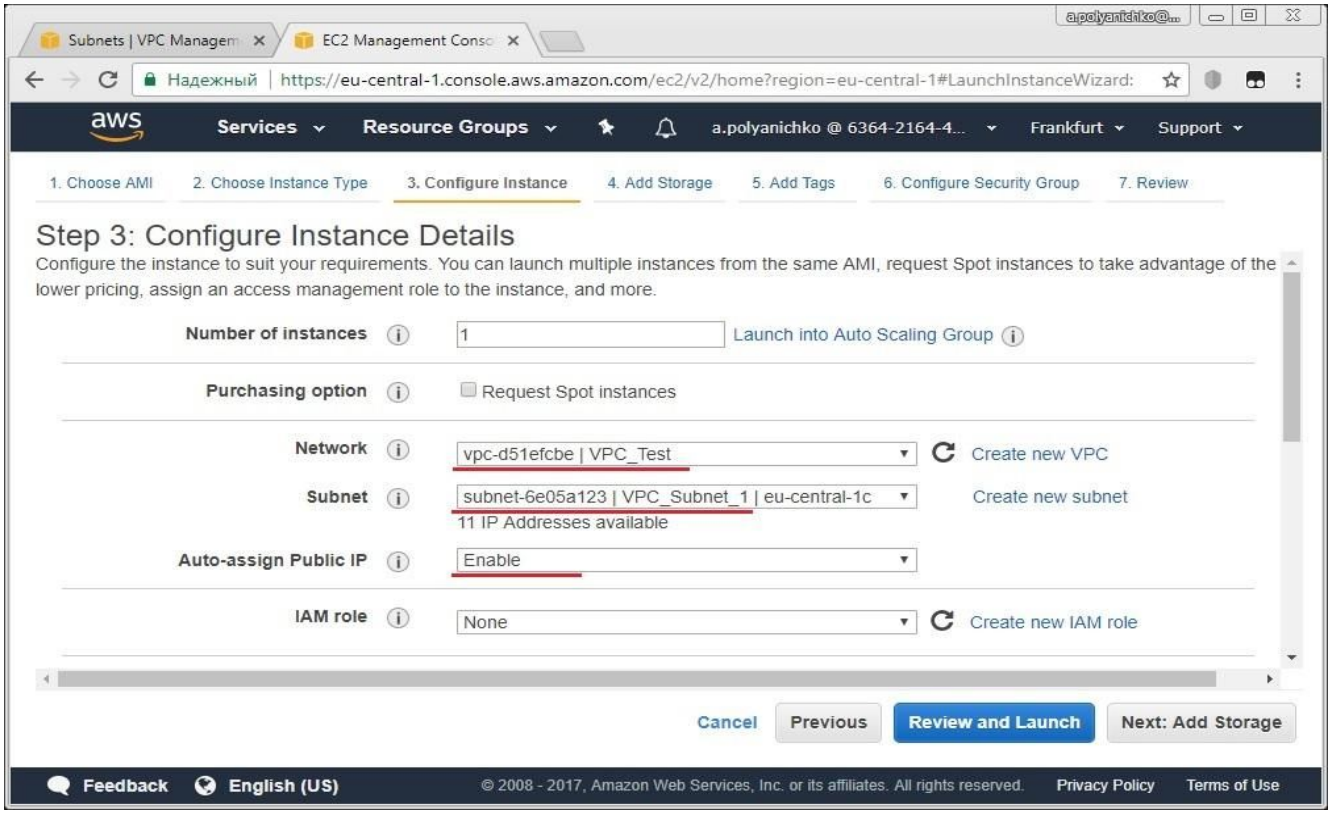
In this exercise we will add two EC2 instances to our public subnet and grant the public access for one of them for external accessibility:



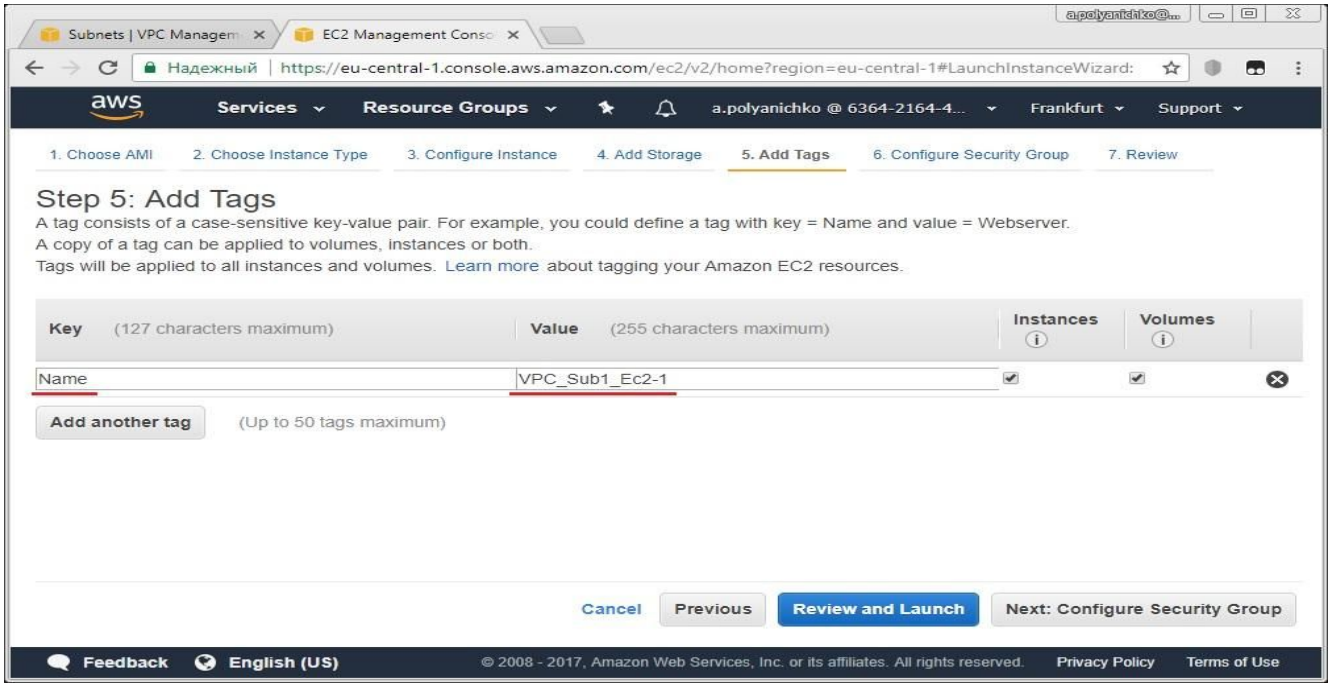
Switch to EC2 Dashboard and create new Amazon Linux AMI EC2 instance as was described in Module 3 hands-on lab except for the options mentioned below:



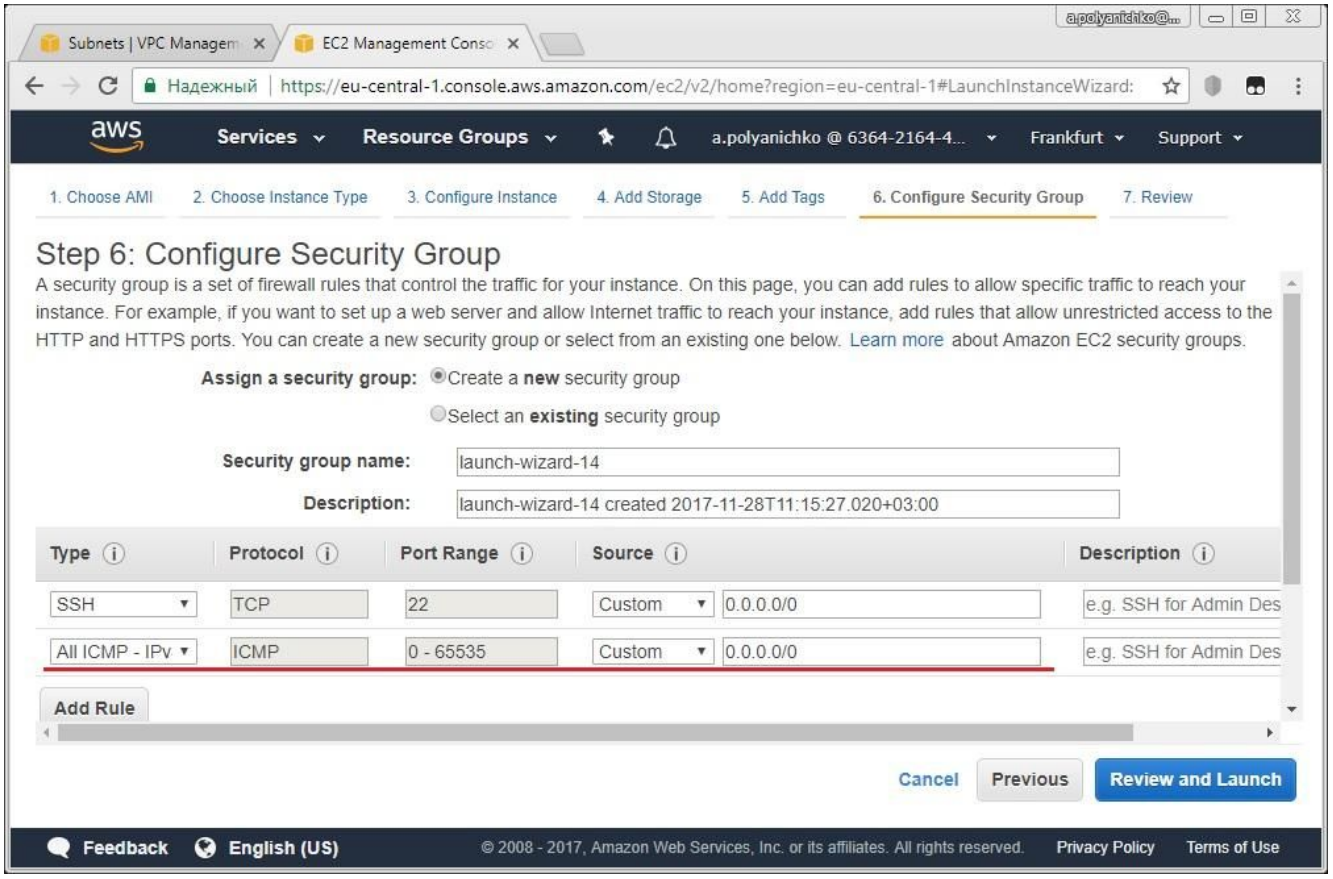
On “Configure Instance” tab: specify your VPC as Network and Subnet_1 as Subnet as well as enable Auto-assigning of Public IP (the last one is mandatory if you are planning to connect your instance from outside of VPC):



On “Add Tags” tab: set the user-friendly name for instance:

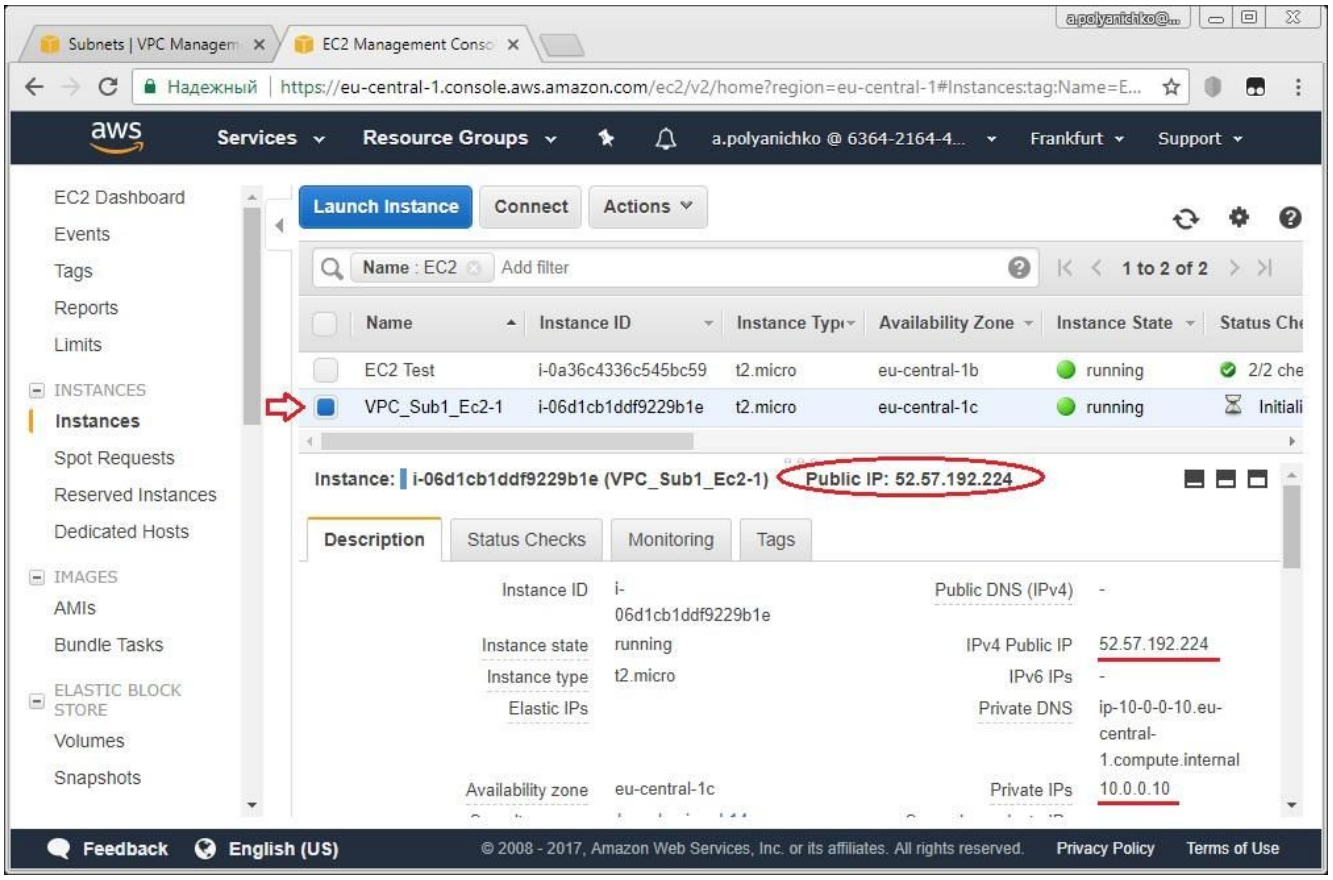


On “Configure Security Group” tab: add the rule for all-ICMP traffic (it is mandatory when you want to use ping command for the instance):

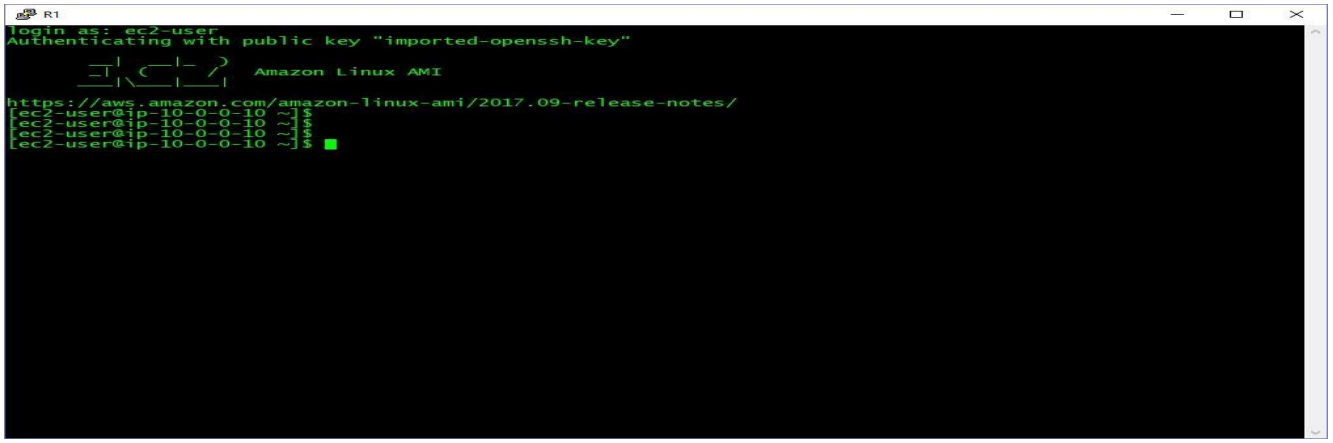


	You may assign existing key prepared on previous lab exercises to secure SSH connection when instance is launching.

Launch your instance and then find it on EC2 Dashboard:



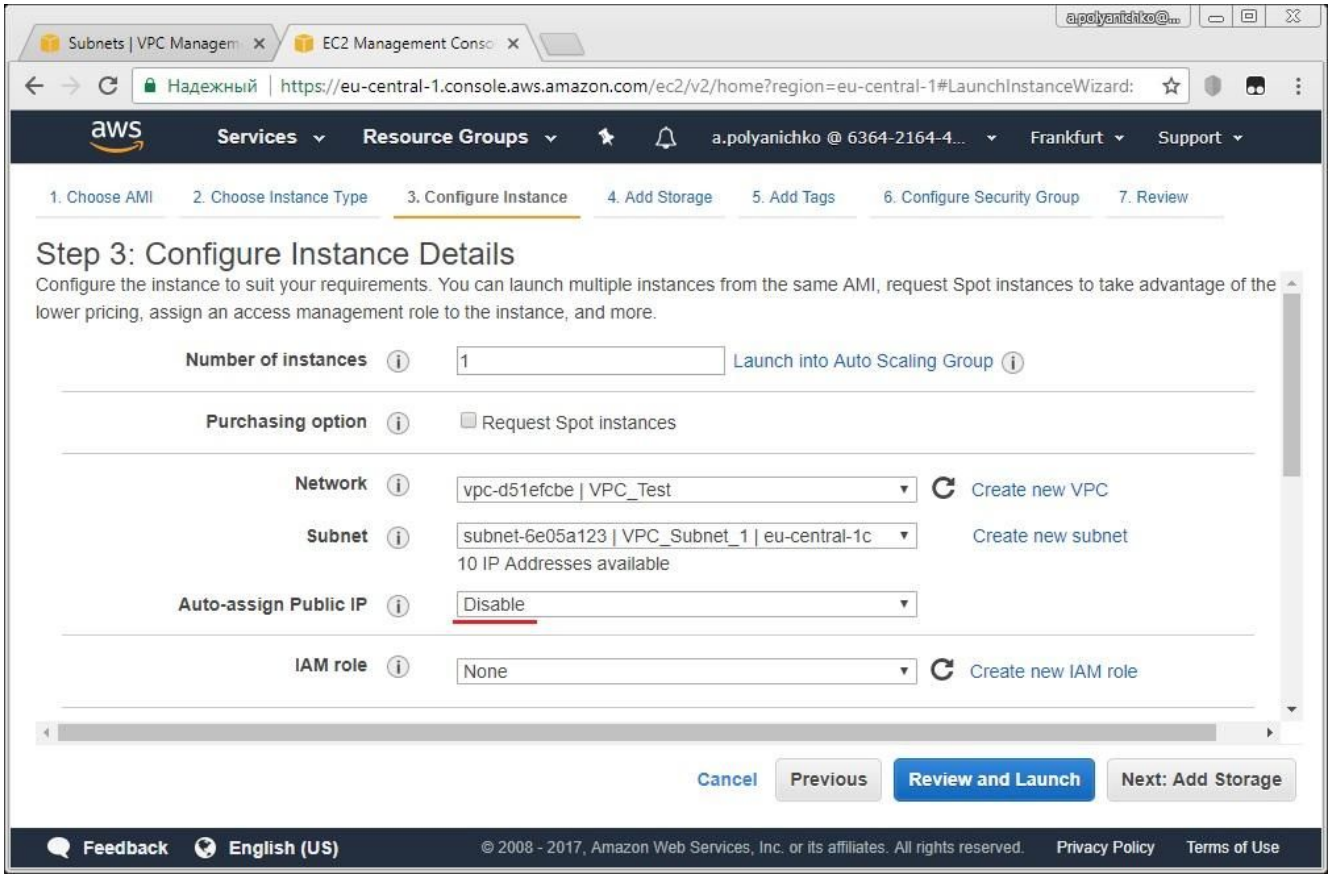
Note Private IP and Public IP of your instance than connect to the instance by PuTTY via SSH as it was described in MODULE 3 lab:



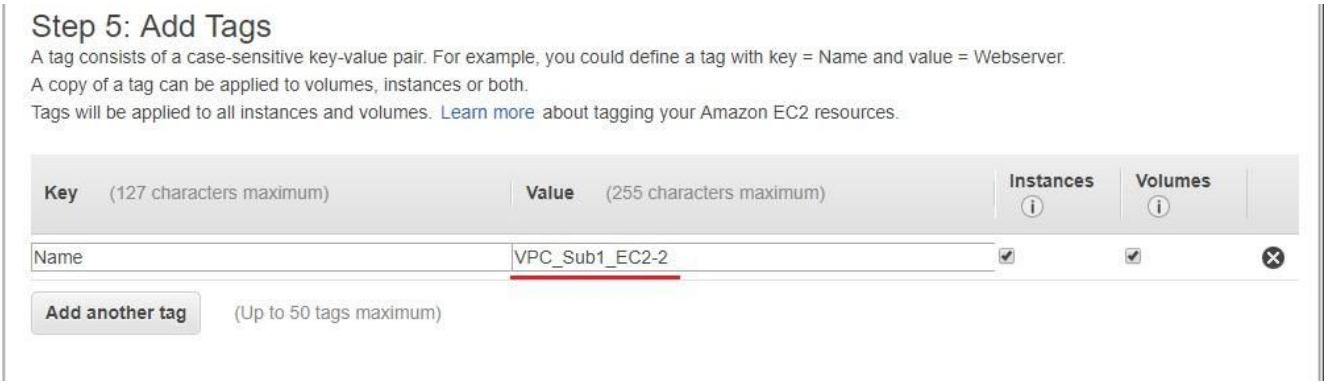
So your public EC2 instance is alive and accessible from outside and then you may create one more EC2 instance with private settings.

Repeat all steps as was described here for Amazon Linux AMI EC2 instance creation except two options below.

Disable Auto-assigning of Public IP on “Configure Instance” tab:

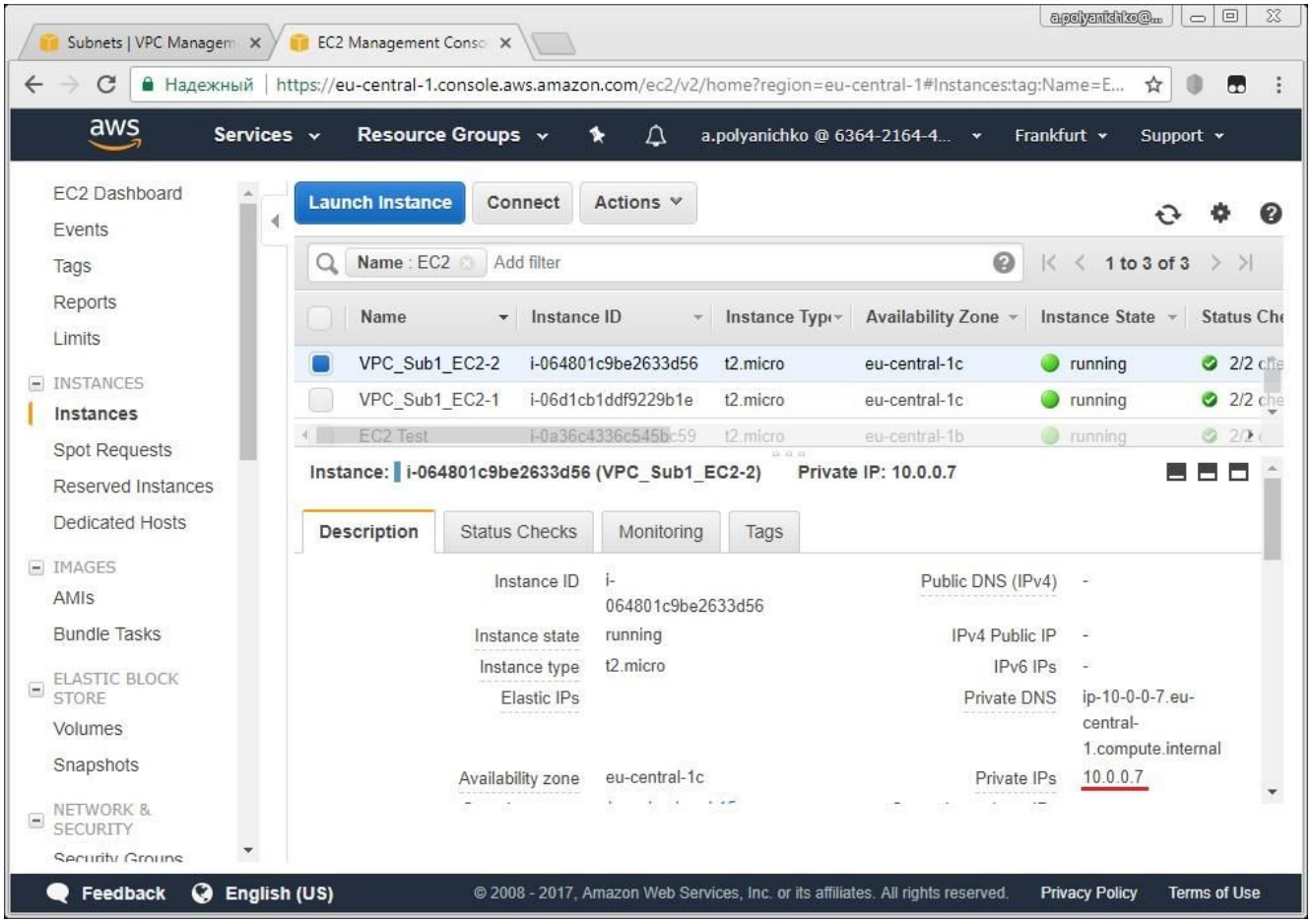


And, of course, specify appropriate user-friendly name on “Add Tags” tab:



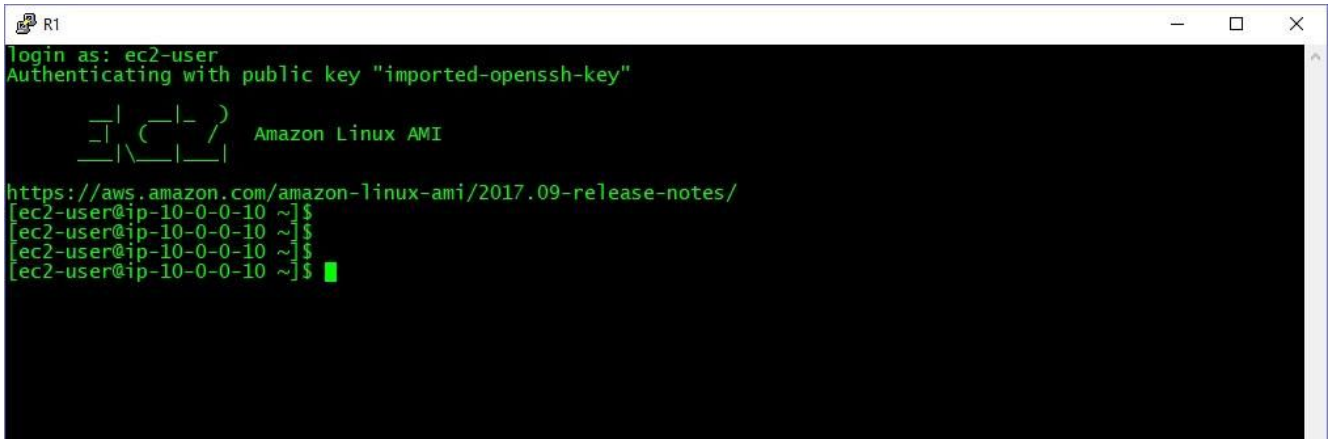
	Please don't forget to enable ICMP traffic on "Configure Security Group" tab.

Find your second instance on EC2 Dashboard:



Note Private IP address of the instance (public IP must be absent).

Login again to your first (public) EC2-1 instance using PuTTY:

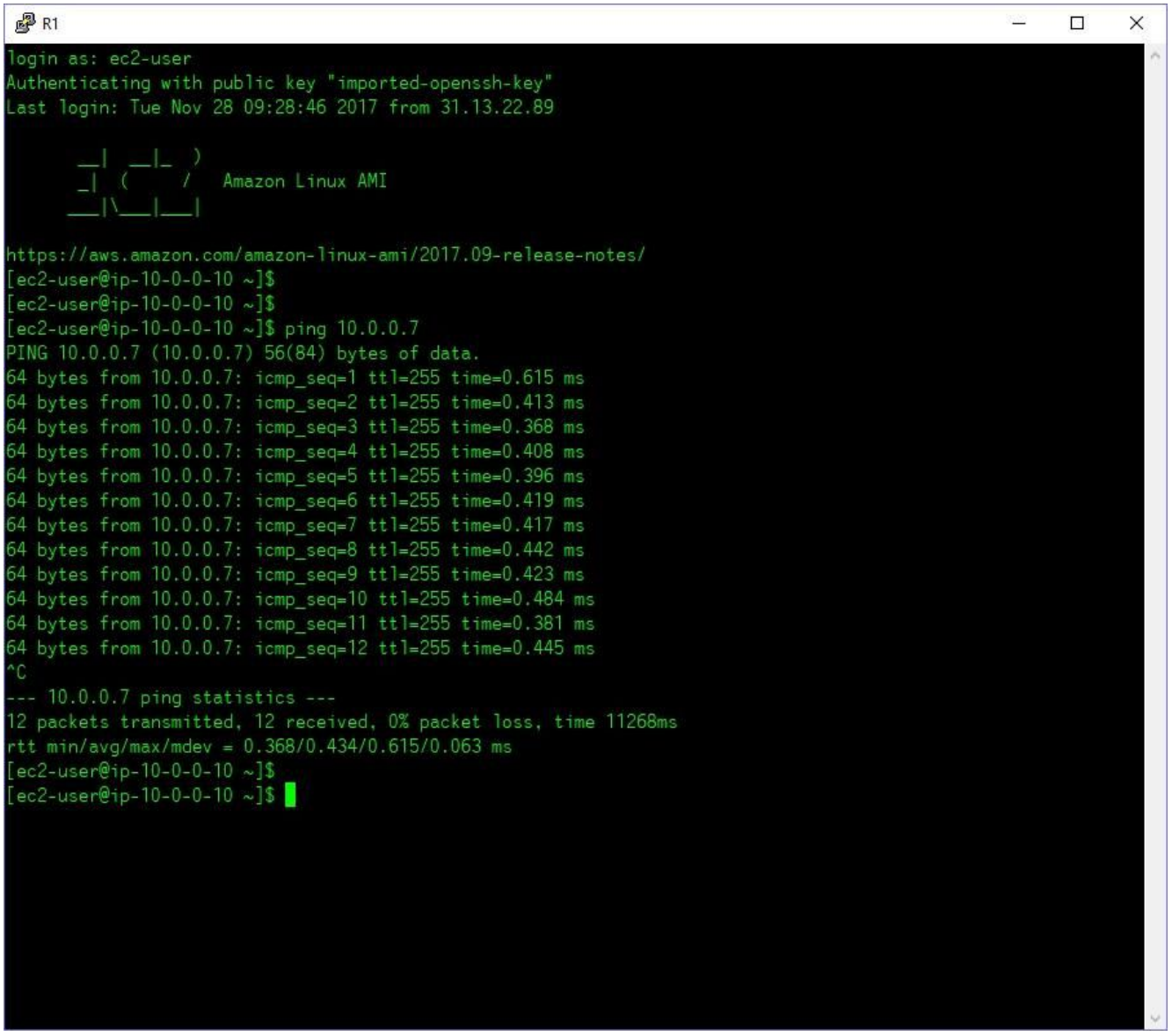


Try to ping EC2-2 from EC2-1 using its private IP address:

$>	[ec2-user@ip-<*EC2-1_Privat_IP*>~]$ **ping <*EC2-2_Privat_IP*>**

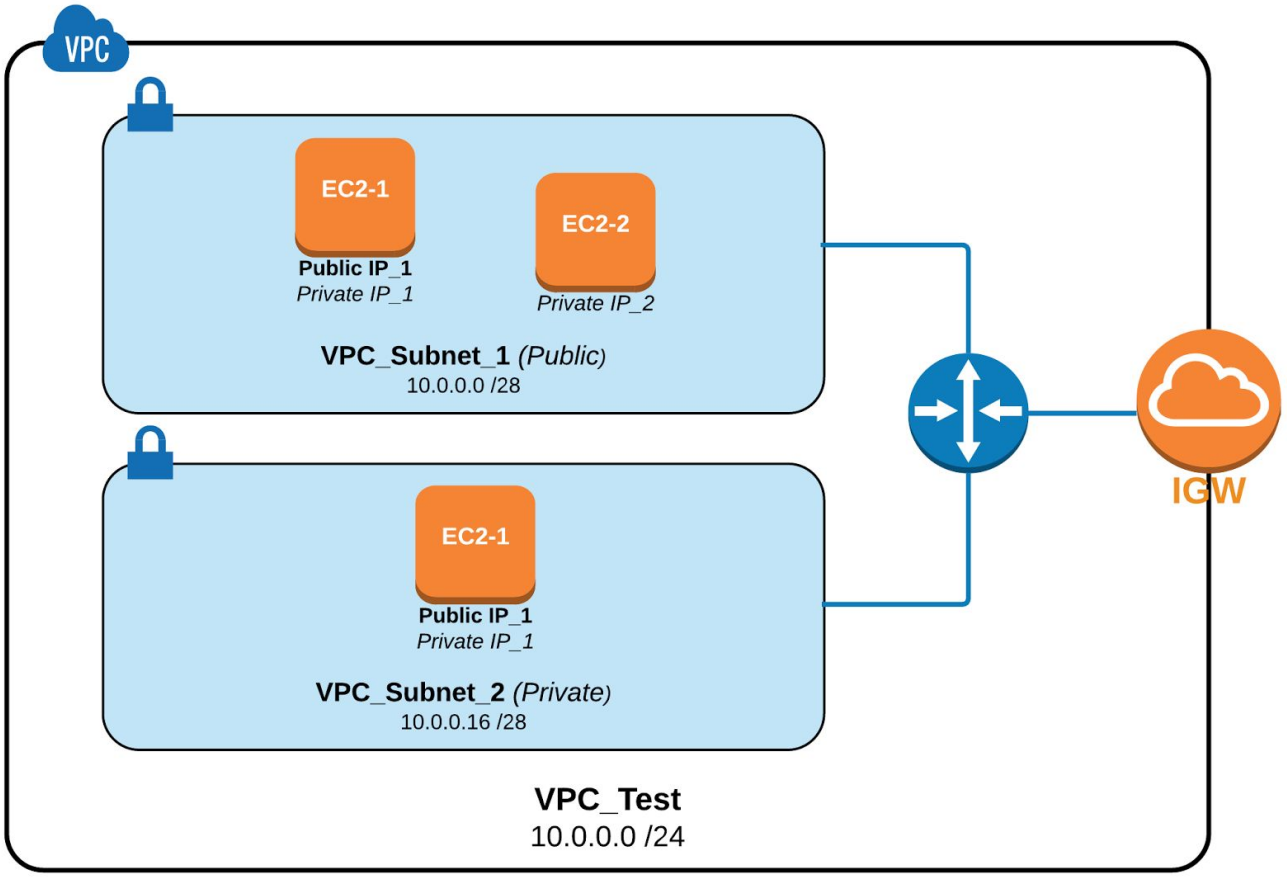
	Press “Ctrl”+”C” keys to stop output of ping command.

If all configurations were done correctly, you will see the output like this:

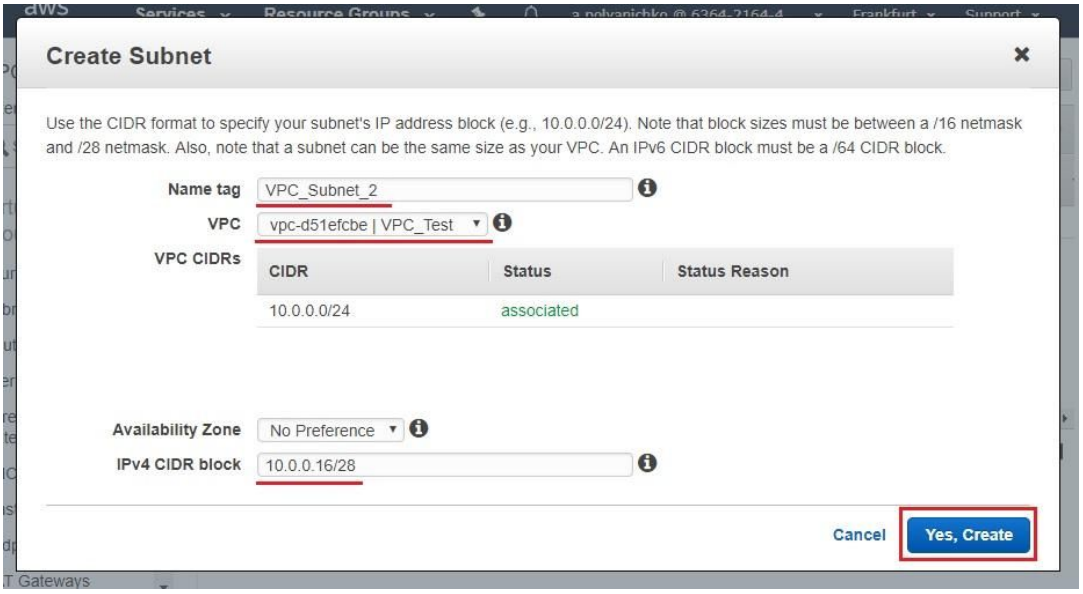


3. Adding Private Subnet and EC2 Instance to VPC

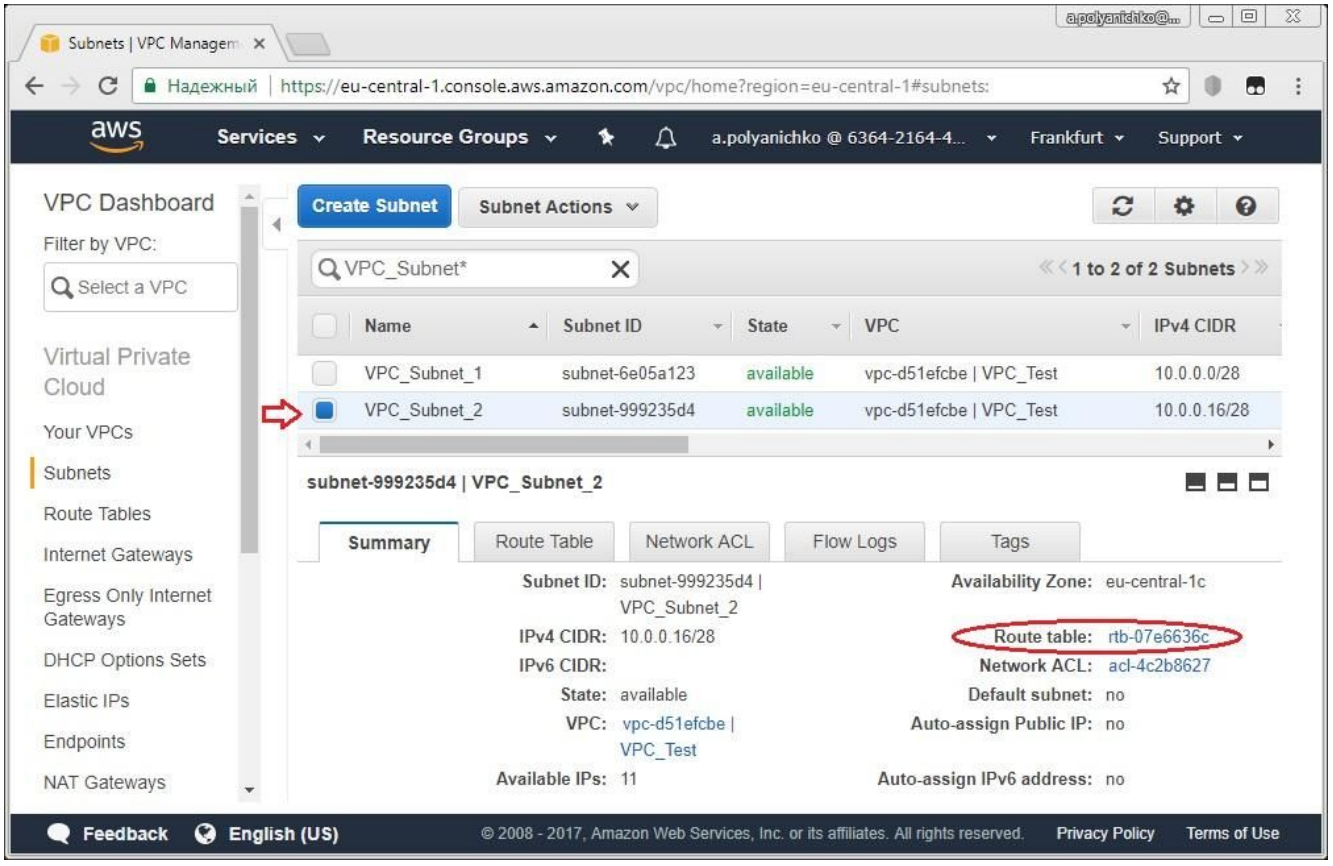
In the next exercise, we will supplement the existing network infrastructure with a new Private subnet and add one more EC2 Instance to new subnet therefore our network will look like this:



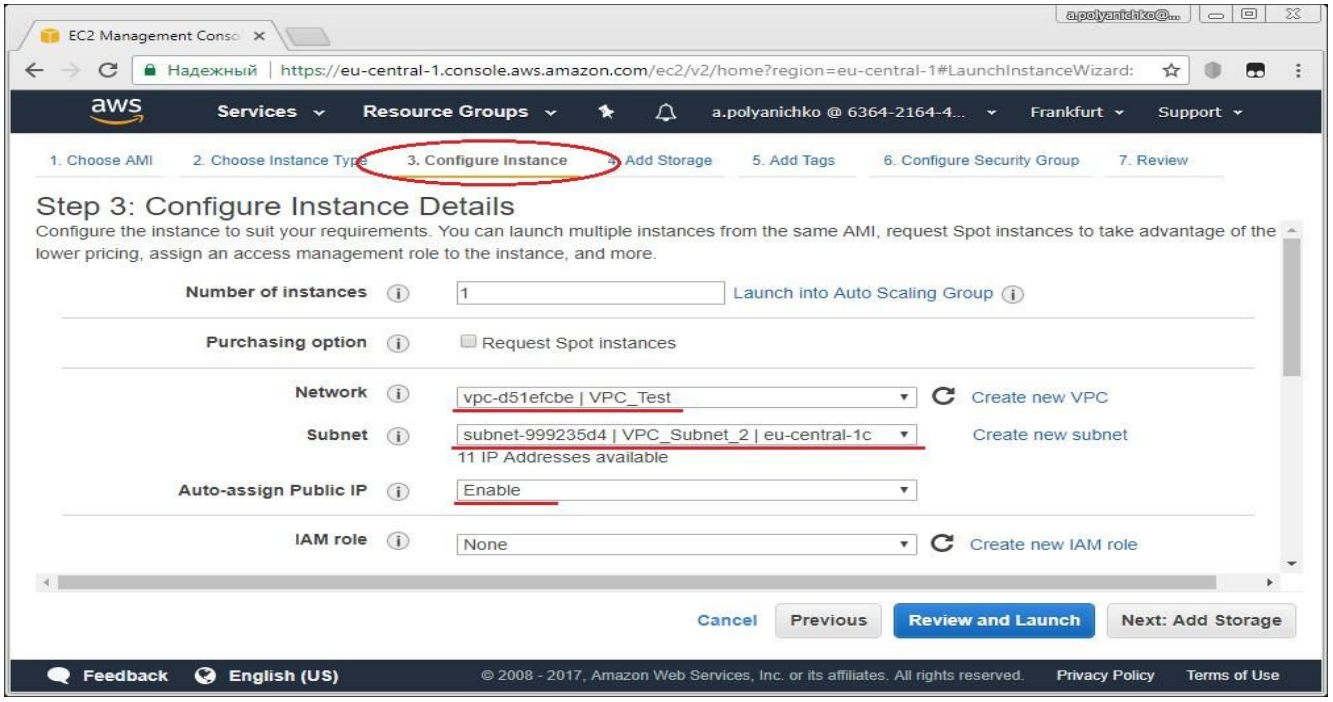
Start from VPC Dashboard and create new subnet named “VPC_Subnet_2” in your VPC based on CIDR 10.0.0.16/28:

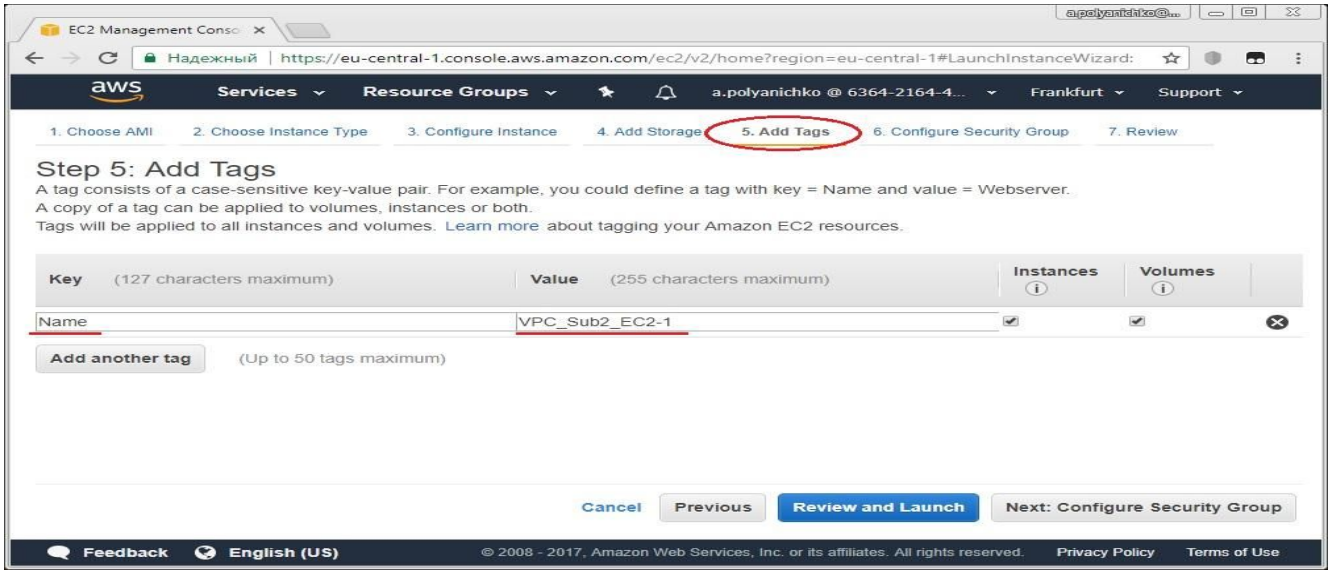


Find the subnet you created in the list of subnets and ensure that it uses default Route table for your VPC:



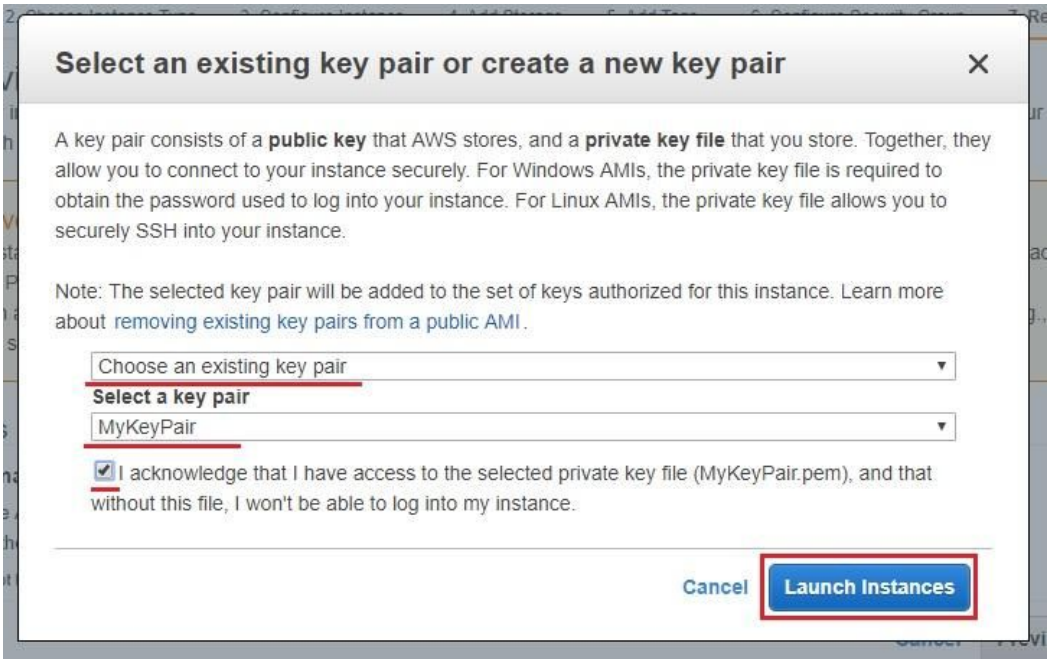
Then create new Amazon Linux AMI EC2 Instance within Subnet_2 and enable Public IP for it (the process is the same as for two previous instances and differences are shown below explicitly):



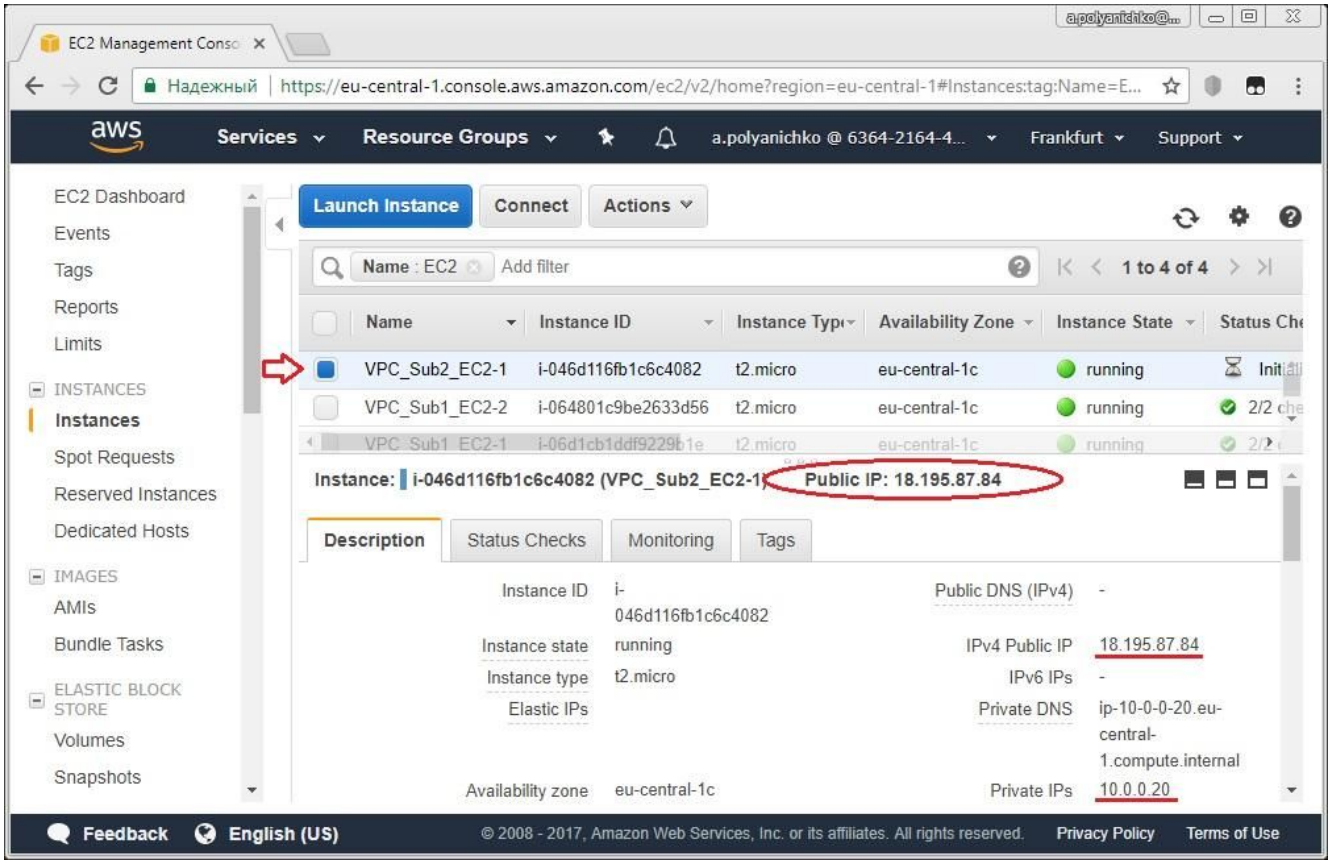


	Don’t forget to enable ICMP protocol in Security Group for the instance

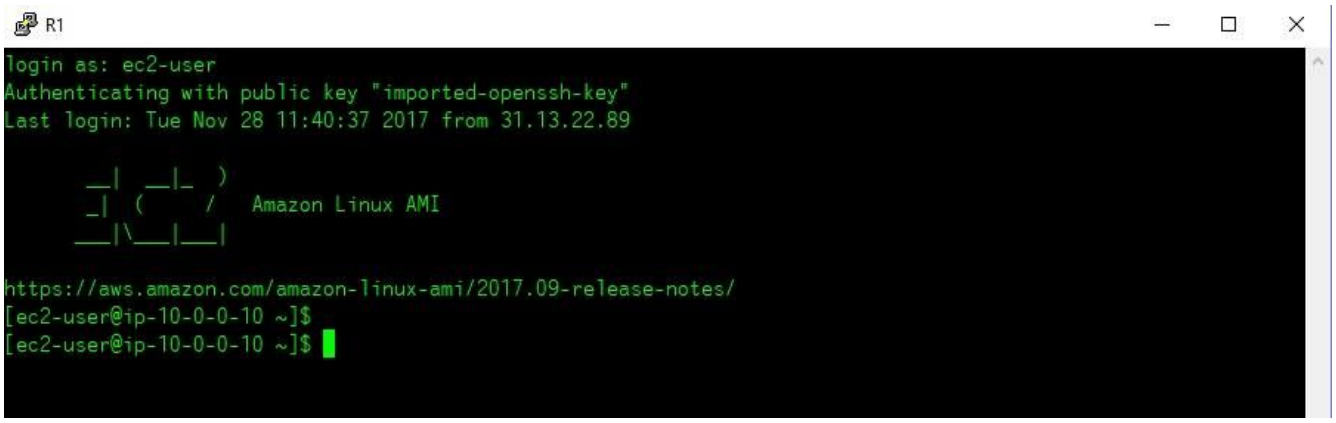
Assign the existing key pair to your instance in launching:



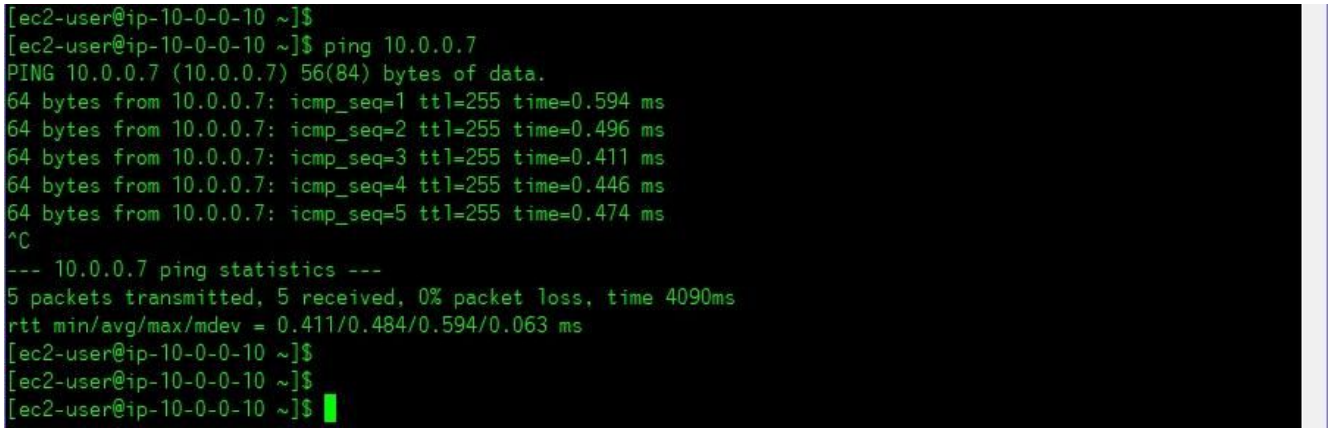
Find your new instance in EC2 Dashboard and note both its Public IP and Private IP:



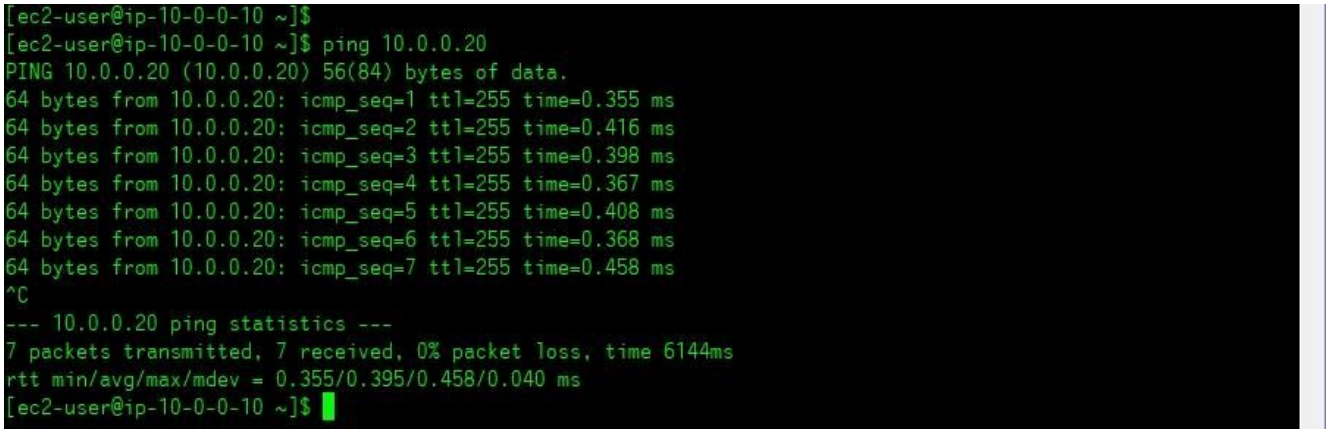
Login to your VPC_Subn1_EC2-1 (the very first) instance via PuTTY SSH using its Public IP:



then try to ping the second instance in Subnet_1 using its Private IP:

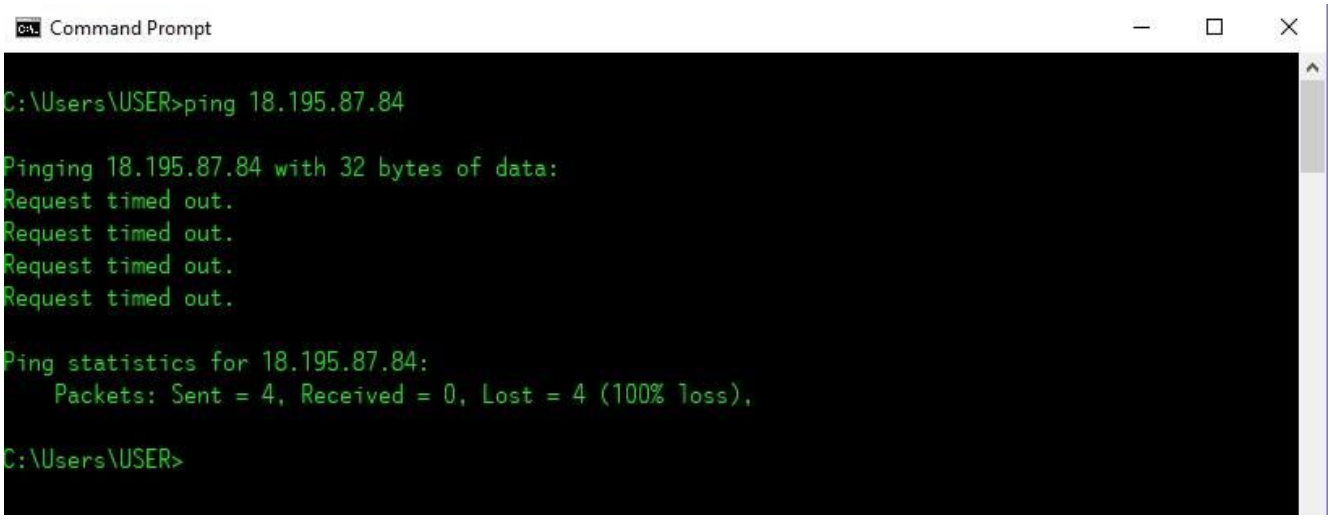


and finally try to ping EC2 instance in Subnet_2 using its Private IP:



So, as you can see, all of EC2 instances in both subnets are mutually accessible and therefore they may exchange the traffic one to another.

However, if you will try to connect VPC_Subn2_EC2-1 instance from outside using its Public IP, the connection attempt will be failed (in the example below Windows Command prompt was used to issue ping command):



Finally we can make the following conclusion about our instances and their accessibility in our network environment:

- All of EC2 instances in both subnets are mutually accessible for traffic exchange;
- EC2-1 instance in Subnet_1 has Internet access because it has Public IP assigned and Subnet_1 uses customer's routing table where the route to IGW is present;
- EC2-2 instance in Subnet_1 has not Internet access because it has not Public IP address assigned and it uses only Private IP for internal exchange;
- EC2-1 instance in Subnet_2 has not Internet access although it has Public IP address assigned because Subnet_2 uses VPC default routing table where there is not the route to IGW.