kvriksh

Networking in AWS - Part 1

Networking Basics

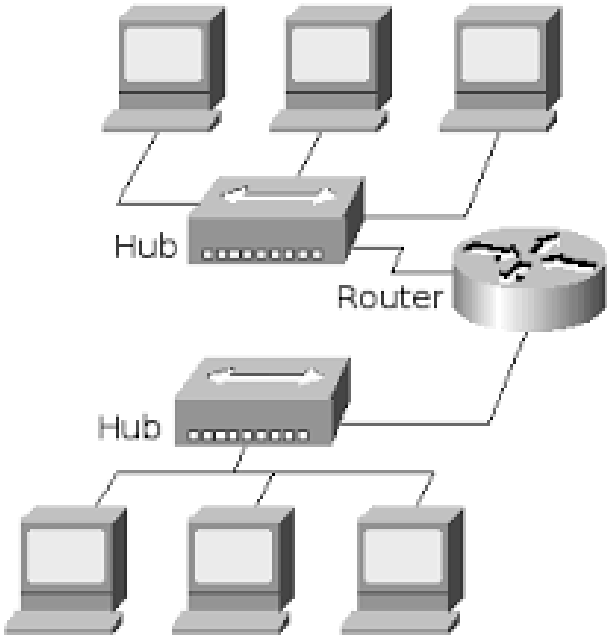
By Chetan Agrawal

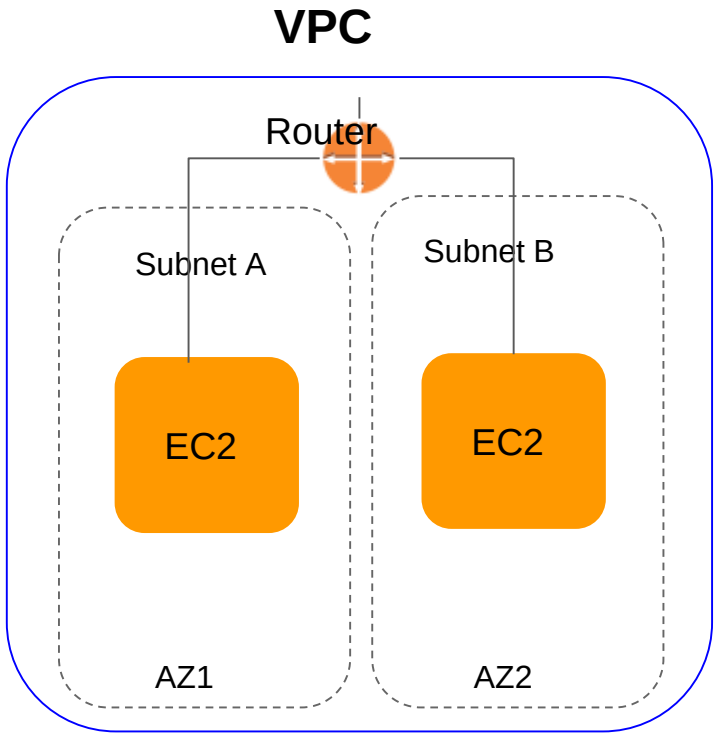
Virtual Private Cloud (VPC)

Definition:

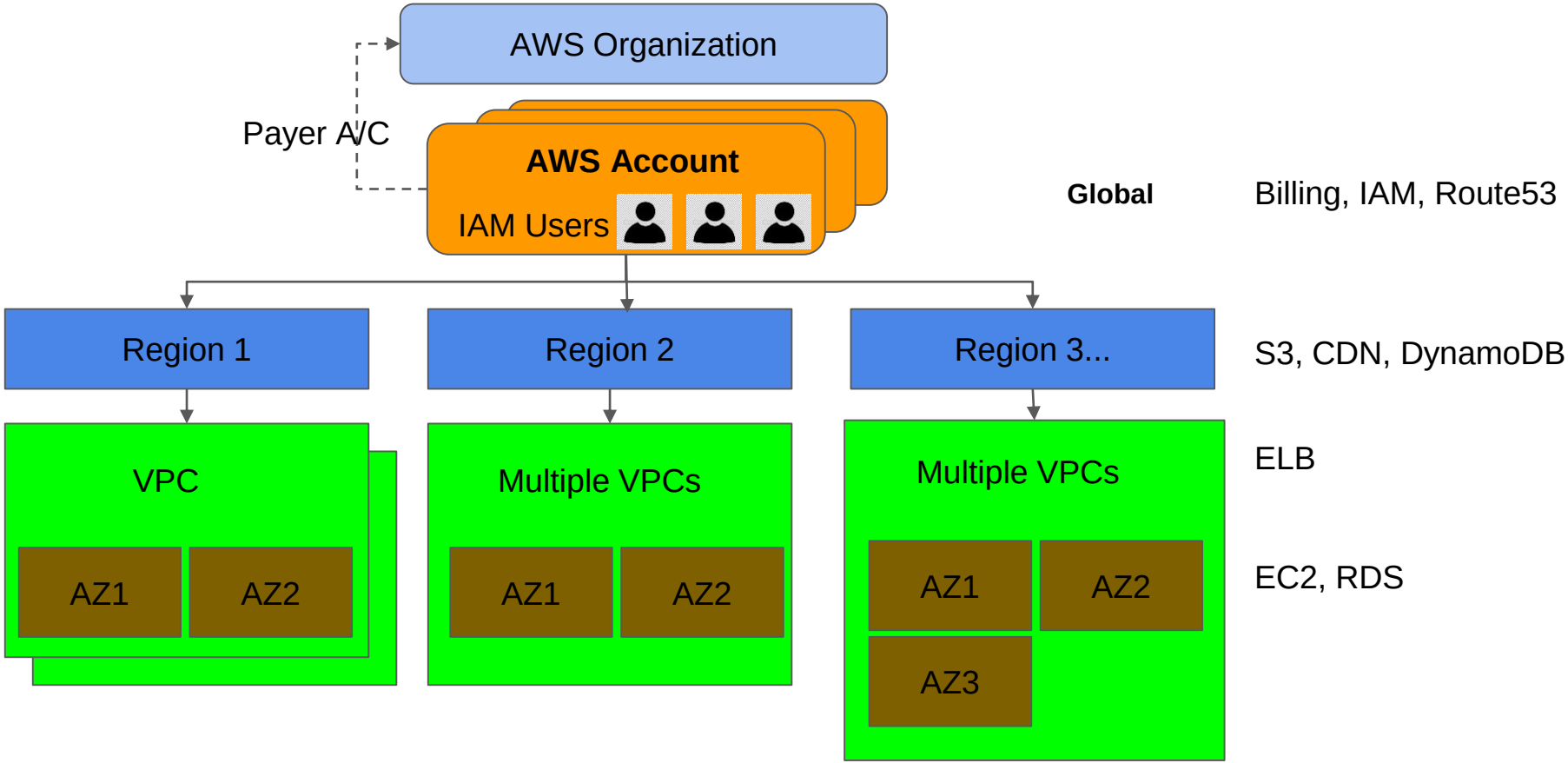
Amazon Virtual Private Cloud (Amazon VPC) enables you to launch Amazon Web Services (AWS) resources into a virtual network that you've defined. This virtual network closely resembles a traditional network that you'd operate in your own data center, with the benefits of using the scalable infrastructure of AWS.

Traditional IT network vs VPC

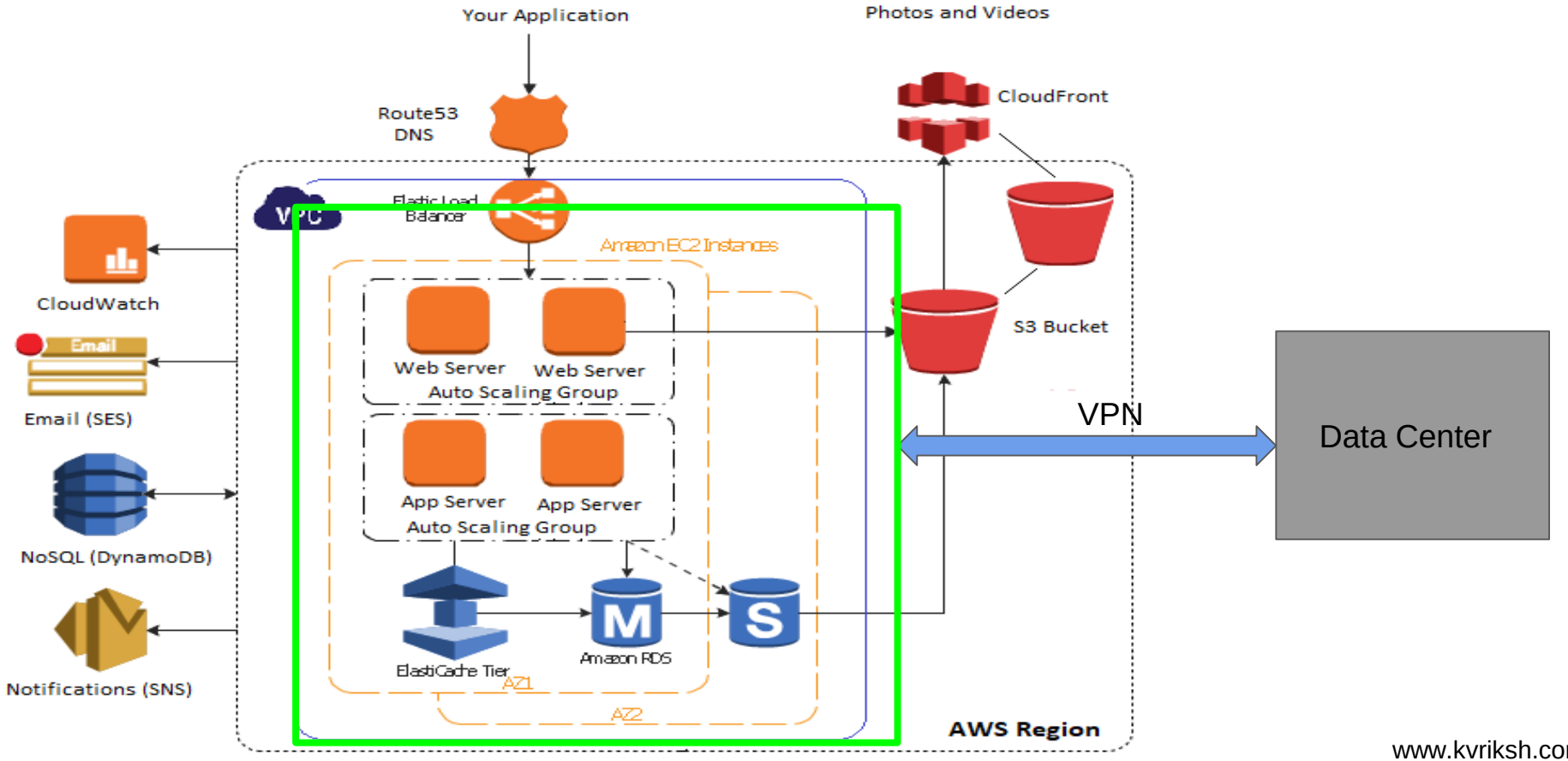




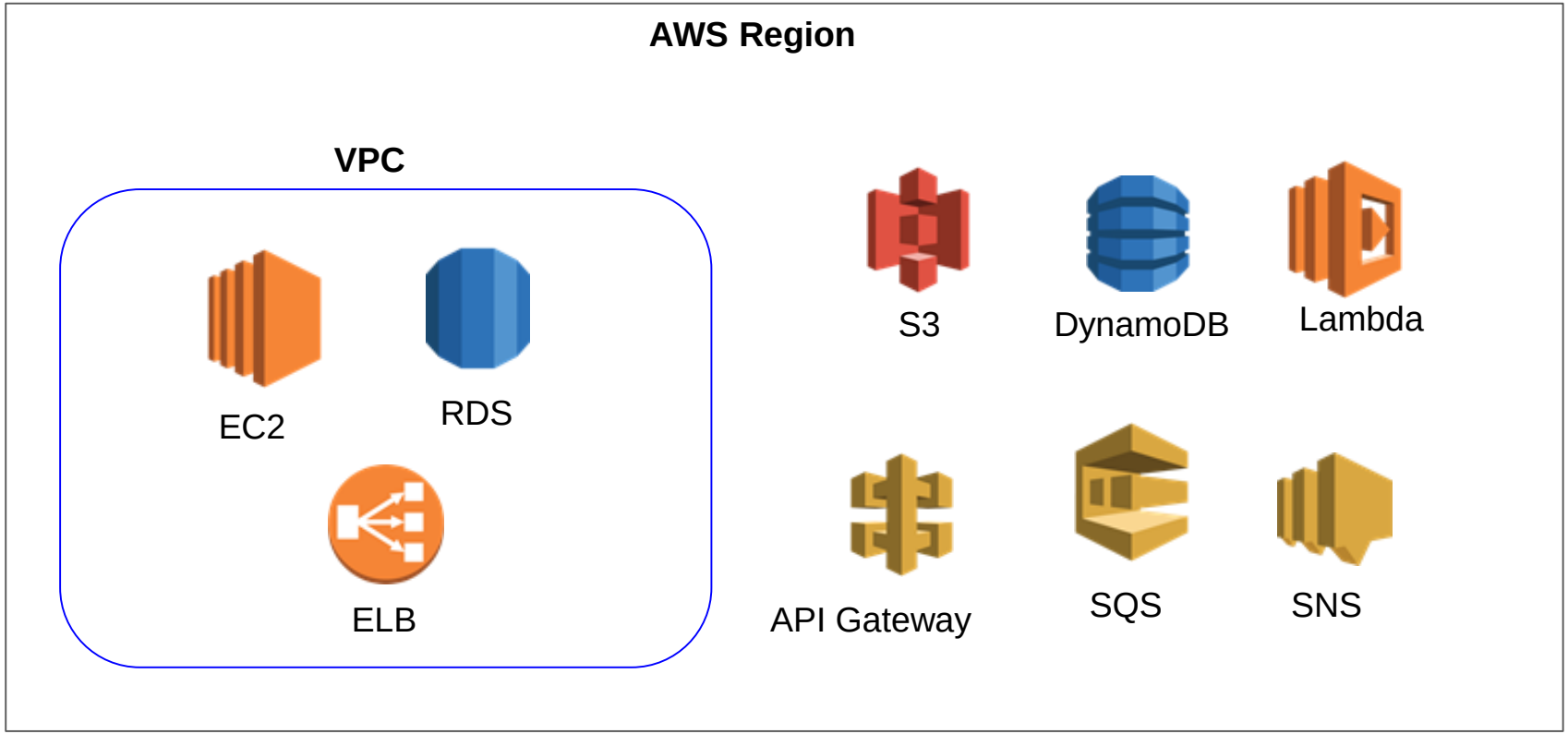
Account, Users, Region and Services



VPC in application architecture

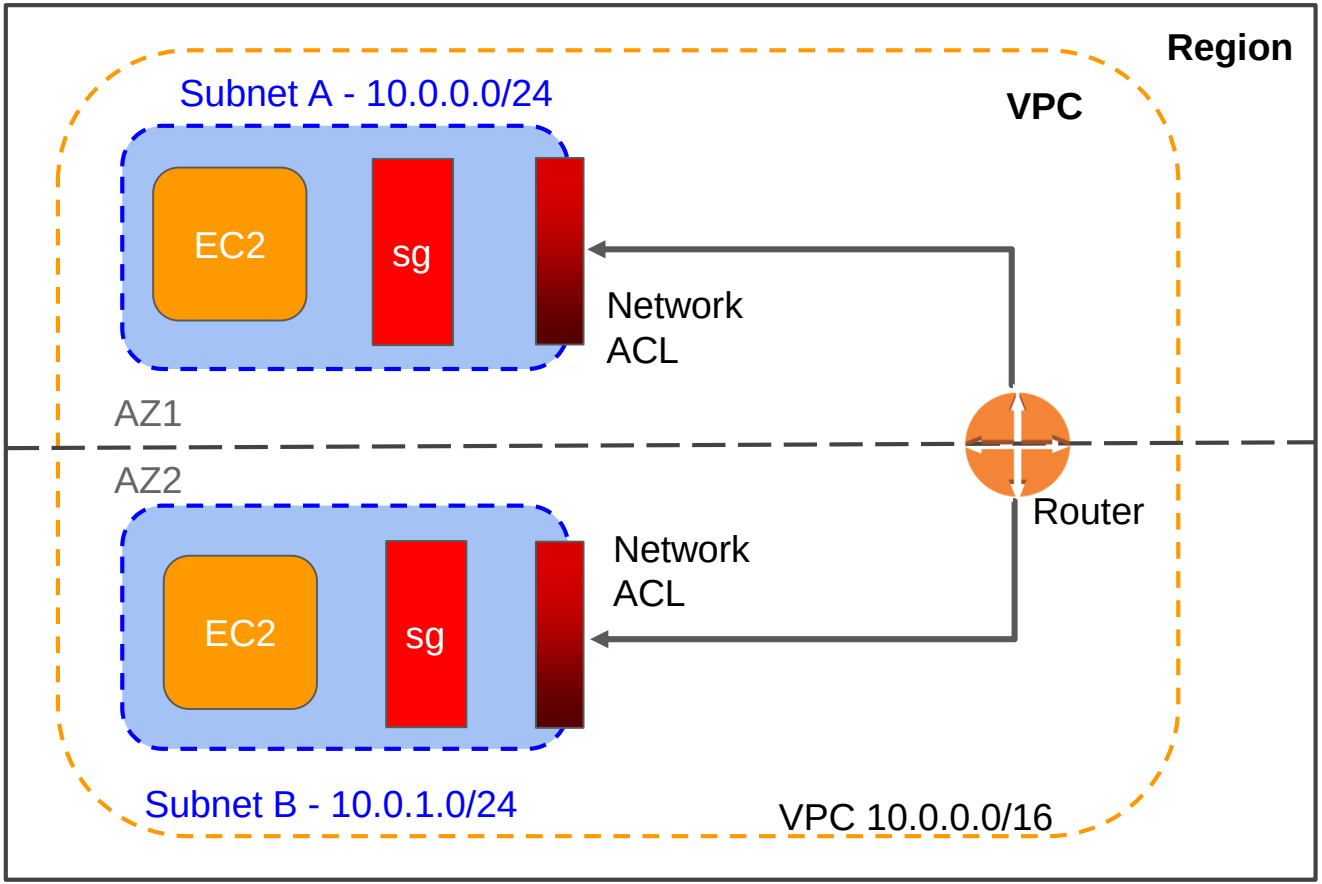


AWS Services inside VPC and outside VPC

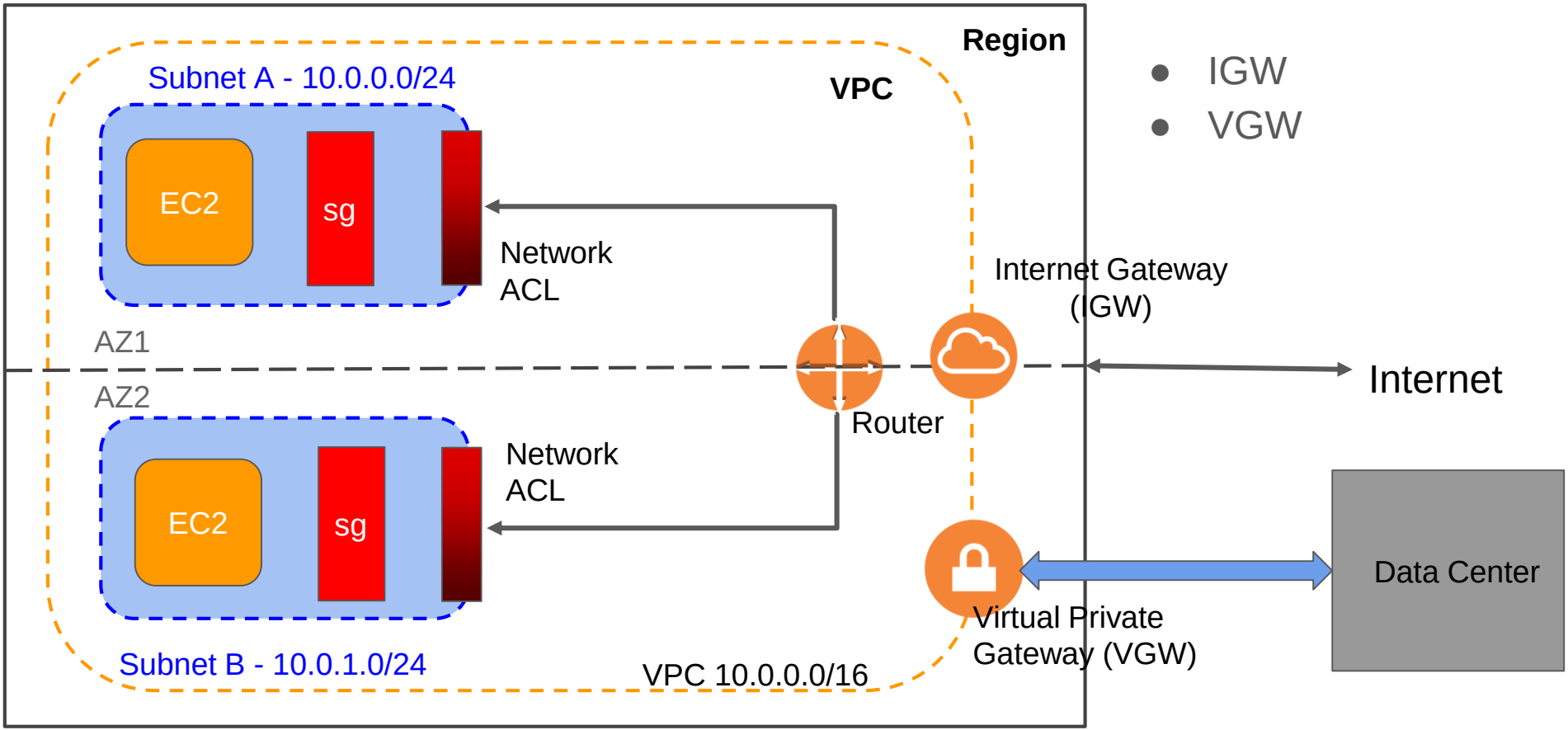


VPC Components

- VPC
- Subnets
- Route Tables
- Security Groups
- Network ACLs



VPC Components



VPC Addressing - Understanding CIDR

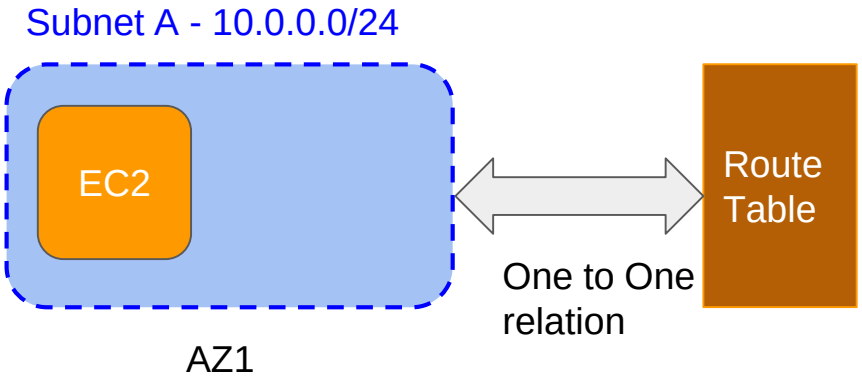
- Classless InterDomain Routing
- Example: 10.200.0.0/16
- VPC CIDR
  - Maximum /16 (65536 IPs)
  - Minimum /28 (16 IPs)
  - VPC Private IP CIDR range
    - 10.0.0.0 - 10.255.255.255 (10/8 prefix)
    - 172.16.0.0 - 172.31.255.255 (172.16/12 prefix)
    - 192.168.0.0 - 192.168.255.255 (192.168/16 prefix)
- Subnet CIDR
  - Same as VPC

Private, Public and Elastic IP (EIP)

Feature	Private	Public	Elastic
Communication	Communication within VPC	Can communicate over internet	Can communicate over internet
Address range	Gets IP address from subnet range. Ex: 10.200.0.1	Gets IP address from Amazon Pool within region	Gets IP address from Amazon Pool within region
Instance restart behavior	Once assigned cannot be changed	Changes over instance restart	Do not change over instance restart. Can be removed anytime.
Releasing IP	Released when instance is terminated	Released to POOL when instance is stopped or terminated	Not released. Remains in your account. (Billed)
Automatic Assignment	Receives private ip on launch on EC2 instance	Receives public ip on launch on EC2 instance if “Public ip addressing attribute” is set to true for subnet	Have to explicitly allocate and attach EIP to EC2 instance. Can be reattached to other EC2
Examples	Application servers, databases	Web servers, Load Balancers, Websites	Web servers, Load Balancers, Websites

Route Table

- Contains rules to route the traffic in/out of Subnets/VPC
- Main route table at VPC level
- Custom route table at Subnet level
- Each route table contains default immutable local route for VPC
- If no custom route table is defined then new subnets are associated with Main route table
- We can modify main route table



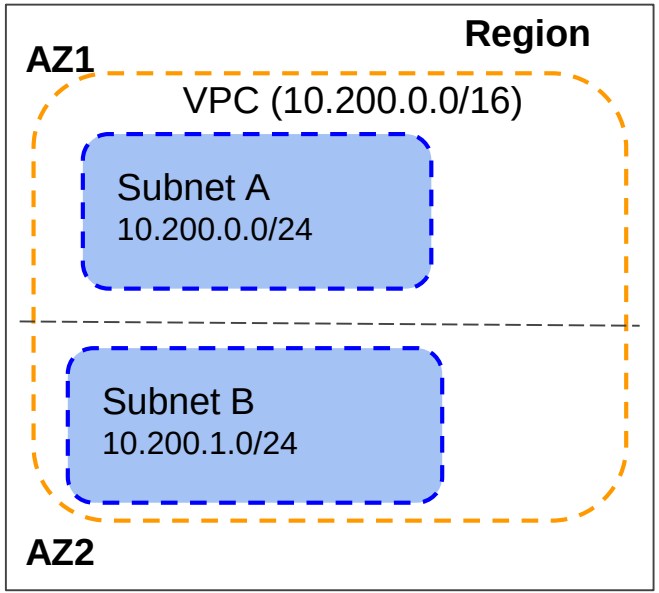
Destination	Target
10.0.0.0/16	local
0.0.0.0/0	*igw-id*

Subnets

1 Subnet = 1 AZ

Public Subnet

- Has route for Internet
- Instances with Public IP can communicate to internet
- Ex: NAT, Web servers, Load balancer



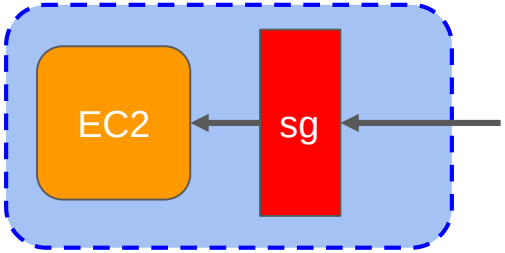
Private Subnet

- No route to Internet
- Instances receives private IPs
- Typically uses NAT for instances to have internet access
- Ex: Database, App server

Note: You can not use 5 IP addresses in given subnet.

Security Groups

- Virtual Firewall - First level of defence
- Stateful - No need to explicitly allow return traffic
- Works at EC2 and RDS level
- Default - **allows all outbound** traffic. **No inbound** traffic.
- Can attach upto 5 security groups to single EC2 instance with 100 rules (in/outbound) per SG
- Can only specify **ALLOW** rules.
- Deny rules can not be specified
- Changes to rules are in effect immediately
- When target is Security group, allow inbound connection to all instances associated with target security group

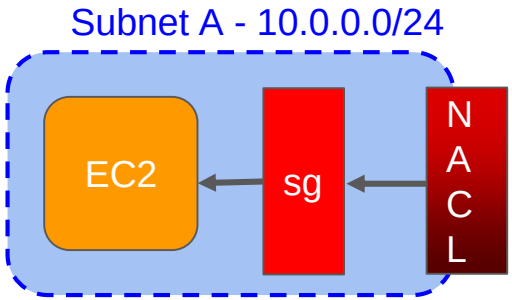


Security group inbound rules

Type	Protocol	Port	Source
HTTP	TCP	80	0.0.0.0/0
HTTPS	TCP	443	0.0.0.0/0
SSH	TCP	22	180.151.138.43/32

Network Access Control List (NACL)

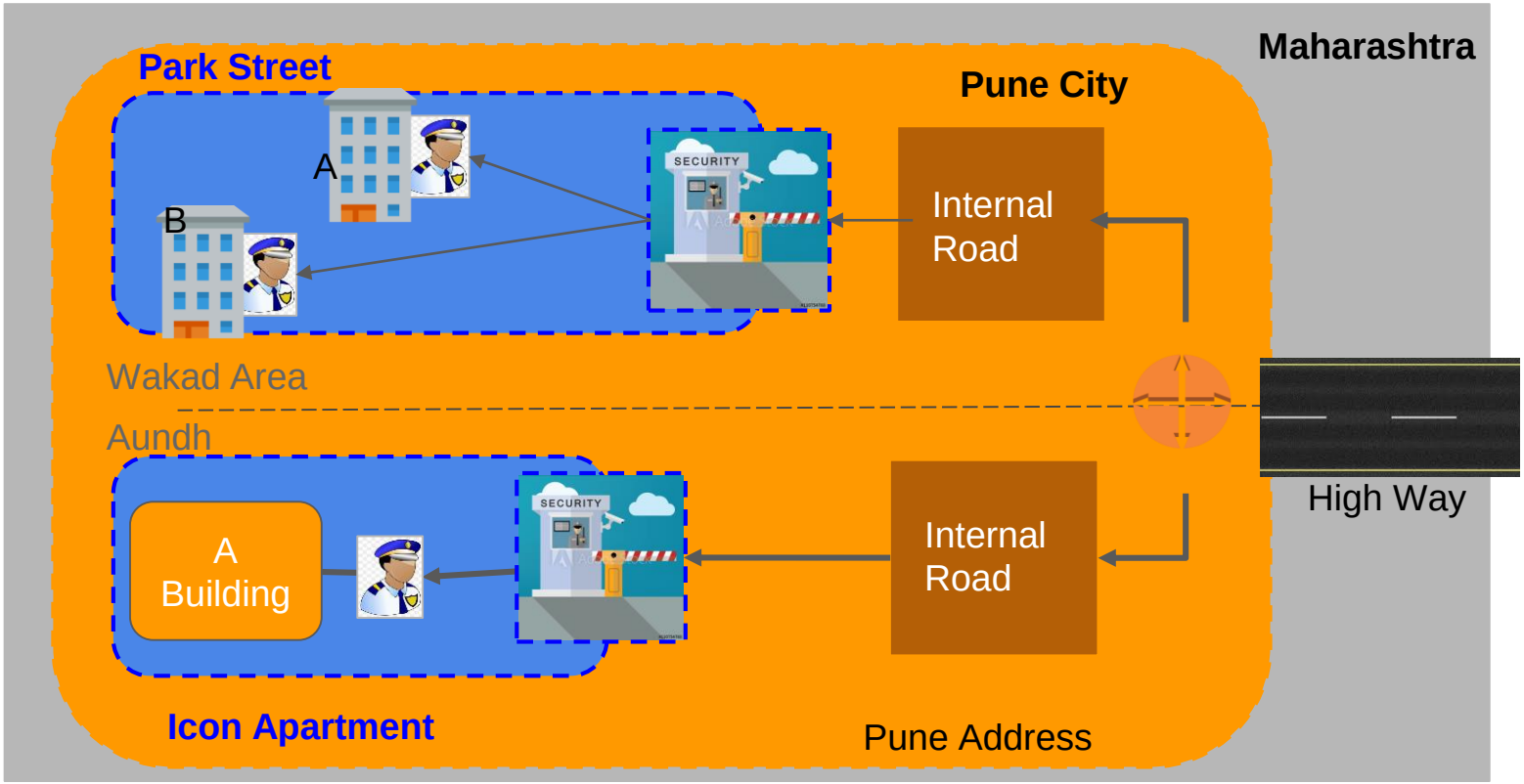
- Stateless firewall - 2nd level of network security
- Works at Subnet level - applied to all instances
- Stateless
- Contains both Allow and Deny rules
- Rules are evaluated in the order of rule number
- Default NACL allows all inbound and outbound traffic



#Rule	Type	Protocol	Port	Source	Allow/Deny
100	HTTP	TCP	80	0.0.0.0/0	ALLOW
101	HTTPS	TCP	443	0.0.0.0/0	ALLOW
*	ALL Traffic	ALL	ALL	180.151.138.43/32	DENY

Network ACL inbound rules

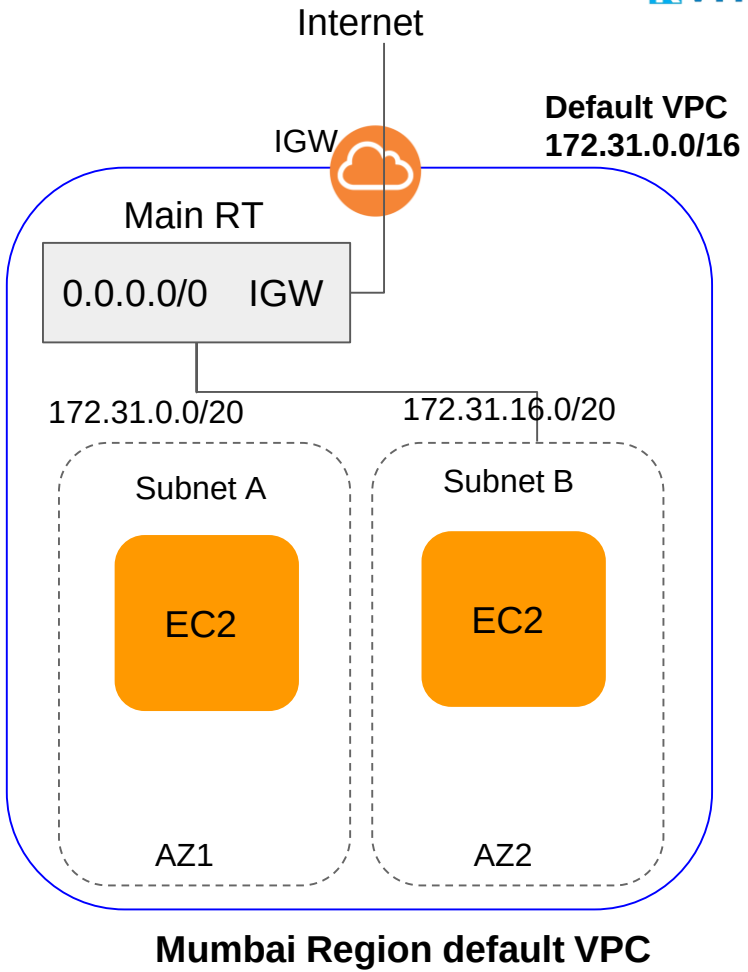
VPC Analogy



- VPC = Pune City, Subnets = Societies, Route Tables = Roads,Network ACLs = Security checkpost at society entrance, Security Groups = Security guard at building entrance

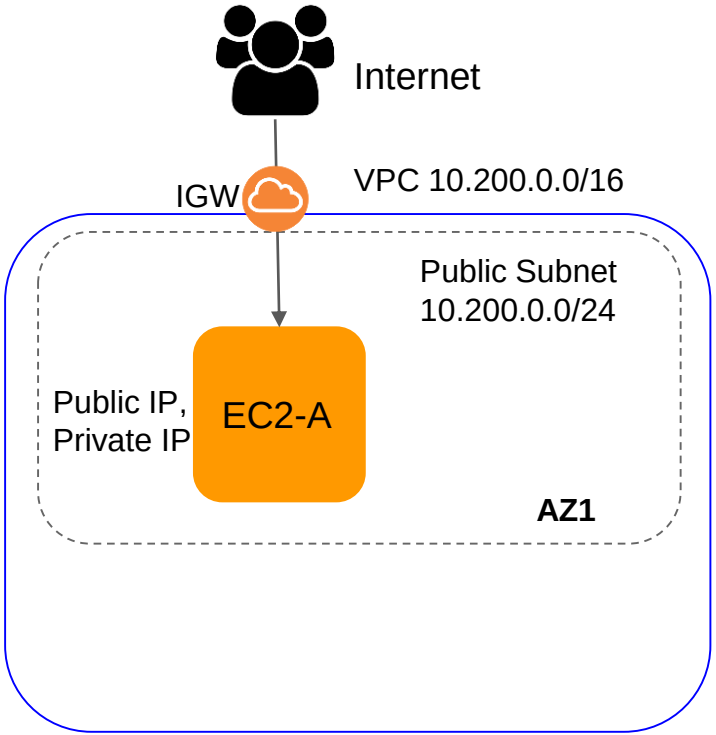
Default VPC

- AWS Creates Default VPC in each AWS region
- Creates VPC with CIDR - 172.31.0.0/16
- Creates Subnets in every AZ with CIDR /20
- Creates Internet Gateway
- Main route table with route to Internet which make all subnets public
- If deleted, you can recreate default VPC by VPC Service -> Your VPCs -> Action -> Create Default VPC



Mumbai Region default VPC

VPC with Single Public Subnet



Public Subnet Route Table

Destination	Target
10.200.0.0/16	local
0.0.0.0/0	igw-xxx

Note: For EC2 instance to be reachable from internet, it must be in Public Subnet and must have Public IP

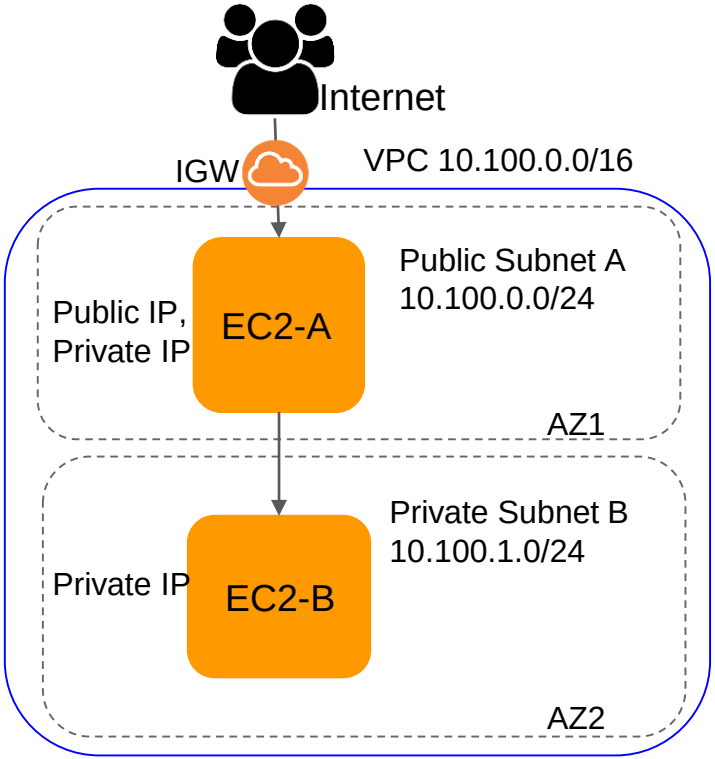
Exercise 1

1. Delete default VPC
2. Create VPC
    a. Go to VPC service -> Your VPCs -> Create VPC (Name: MyVPC, CIDR: 10.100.0.0/16) -> Create
3. Create Internet Gateway
    a. Internet Gateways -> Create internet gateway
4. Attach Internet Gateway to VPC
    a. Select Internet gateway -> Actions -> Attach to VPC -> Select your VPC
5. Create Subnet
    a. Subnets -> Create subnet (Name: MyVPC-Public, VPC: MyVPC, AZ: Select first az - ap-south-1a, CIDR: 10.100.0.0/24)
    b. Select Subnet -> Action -> Modify Auto Assign Public IP -> Enable -> Save
6. Create Route table
    a. Route Tables -> Create Route Table (Name: MyVPC-Public, VPC: MyVPC)
    b. Select Route table -> Routes -> Edit -> Add another route (Destination: 0.0.0.0/0, Target: Internet gateway -> igw-xxx) -> Save

Exercise 1

6. Associate Route table with Subnet to make it Public subnet
    a. Select Route table -> Subnet Associations -> Edit -> Check the MyVPC-Public subnet -> Save
7. Launch EC2 instance in newly created Public Subnet
    a. Go to EC2 Service -> Instances
    b. Launch EC2 Instance -> Select Amazon Linux 2 -> Select t2.micro
    c. Configure Instance Details:
        i. Network: MyVPC
        ii. Subnet: MyVPC-Public (rest all defaults)
    d. Add storage (all defaults)
    e. Add Tags
        i. Key=Name, Value=**EC2-A**
    f. Configure Security Group
        i. Add rule for SSH port 22 for source as MyIP
    g. Review and Launch
8. Connect to EC2 instance (Public IP) from your laptop using Putty or terminal (ec2-user)

VPC with Public and Private Subnet



Private Subnet Route Table

Destination	Target
10.100.0.0/16	local

Exercise 2 (Continuing with previous setup)

1. Create a Private subnet
   a. Create subnet (Name: MyVPC-Private, VPC: MyVPC, AZ: Select different az (ap-south-1b), CIDR: 10.100.1.0/24)
2. Create Private route table
   a. Route Tables -> Create Route Table (Name: MyVPC-Private, VPC: MyVPC)
3. Associate Route table with Subnet to make it Private subnet
   a. Select Route table -> Subnet Associations -> Edit -> Check the MyVPC-Private subnet -> Save
4. Launch another EC2 instance in same VPC but in newly created **Private subnet.**
   a. Tag this instance with Name=EC2-B
   b. **New security group**
      - Add rule SSH for CIDR of Public Subnet source CIDR
      - Add rule All-ICMP IPv4 for Public Subnet source CIDR
5. Note down EC2-B private IP address

Exercise 2 (Continuing with previous setup)

6. Try to ping EC2-B Private IP from EC2-A instance -> Should work
7. Try to connect to EC2-B instance from EC2-A (Permissions denied..Why?)
    a. $ssh ec2-user@10.100.1.x (Replace this ip with your EC2-B IP address)

6. Get your ssh .pem file on EC2-A instance
    a. Open local .pem file with nodepad and copy the content (CTRLA -> CTRL+C)
    b. On EC2 A terminal -> vi key.pem -> enter -> press i -> paste using right click -> esc -> :wq -> enter
    c. chmod 600 key.pem
    d. ssh -i key.pem ec2-user@10.100.1.x -> should be able to connect

6. Try to ping google.com from EC2-B instance
    a. ping google.com (You should not be able to ping. Why?)

NAT Instance and NAT Gateway

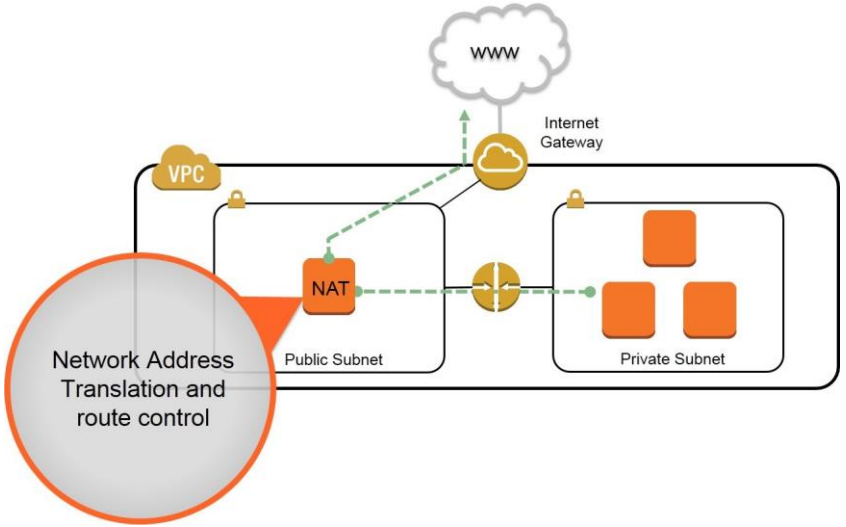
- ➢ To provide Internet Access to instances in private subnet without IGW
- ➢ Performs Network Address translation when packet leaves the VPC

NAT Instance

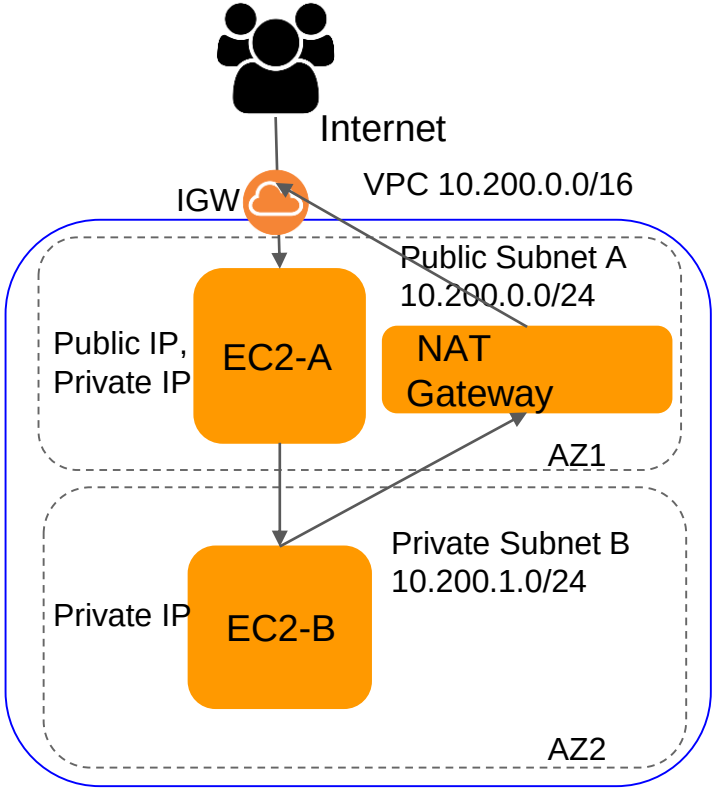
- NAT EC2 can be launched using Amazon Linux Nat AMI
- Disable Source/Destination check on instance
- Allocate EIP

Nat Gateway:

- Managed by AWS
- Highly available with AZ



VPC with Public, Private Subnet and NAT



Private Subnet Route Table

Destination	Target
10.200.0.0/16	local
0.0.0.0/0	NAT-Gateway-ID

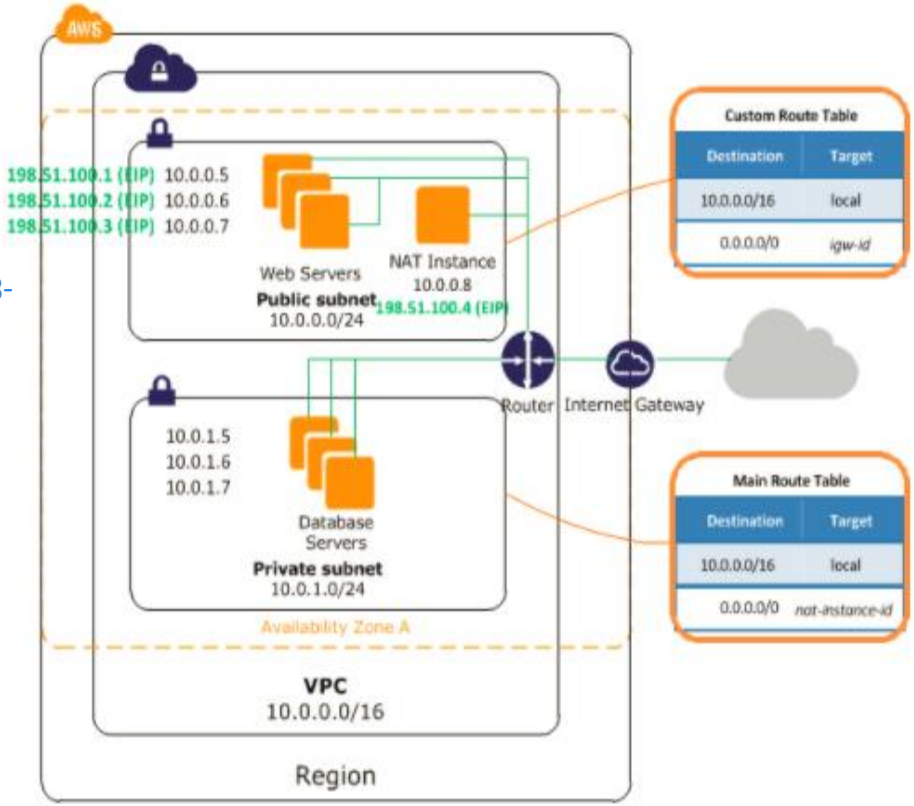
Exercise 3 (Continuing with previous setup)

- Create a NAT gateway in your VPC
  - VPC -> NAT Gateways -> Create NAT Gateway
    - Subnet: MyVPC-Public **(Must select Public Subnet)**
    - EIP: Create New EIP
    - Create NAT Gateway
    - It takes 5-10 minutes for NAT gateway to be Active
- Add a route in Private subnet for internet traffic and route through NAT Gateway
  - Route Tables -> Select MyVPC-Private route table
  - Routes -> Edit -> Add another route
    - Destination: 0.0.0.0/0
    - Target: nat-gateway
    - Save
- Now again try to ping google.com from EC2-B
  - ping google.com

Setup your own NAT on EC2

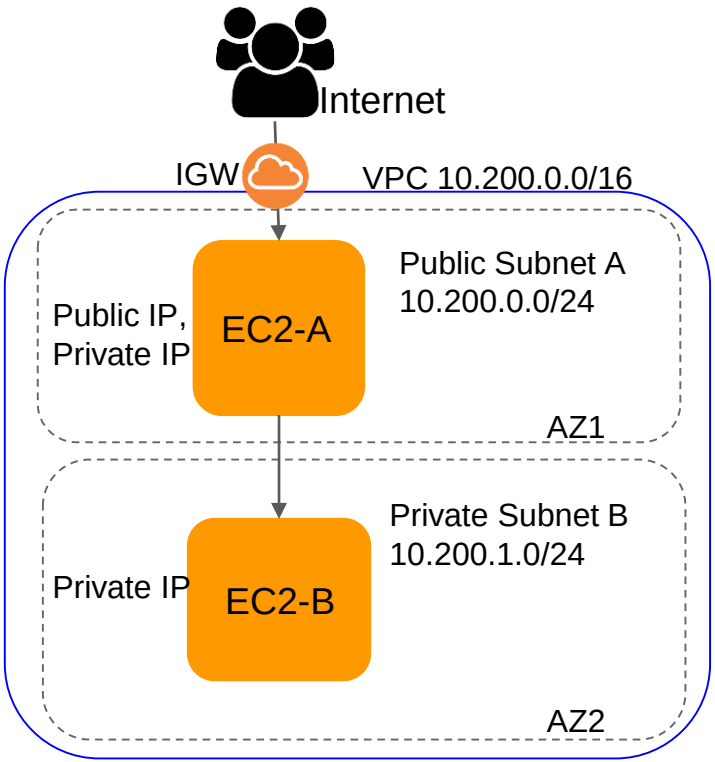
- Must be in Public Subnet
- Must have Public or Elastic IP
- Should be launched using AWS provided NAT AMIs (amzn-ami-vpc-nat) Ex:amzn-ami-vpc-nat-hvm-2017.09.1.20180103-x86_64-ebs (ami-0c184c63)
- Disable Source/Destination Check
- Update Private subnet route tables. For internet traffic set target as NAT Instance ID

Reference: Setting up NAT instance



Assignments

Assignment 1- VPC with Public and Private subnets



Public Subnet Route Table

Destination	Target
10.200.0.0/16	local
0.0.0.0/0	igw-xxx

EC2-A Security Group

Port	Source
22	MyIp

Private Subnet Route Table

Destination	Target
10.200.0.0/16	local

EC2-B Security Group

Port	Source
22	10.200.0.0/24
ICMP IPv4 All	10.200.0.0/24

Assignment 1 - VPC with Public and Private subnets

Watch Video

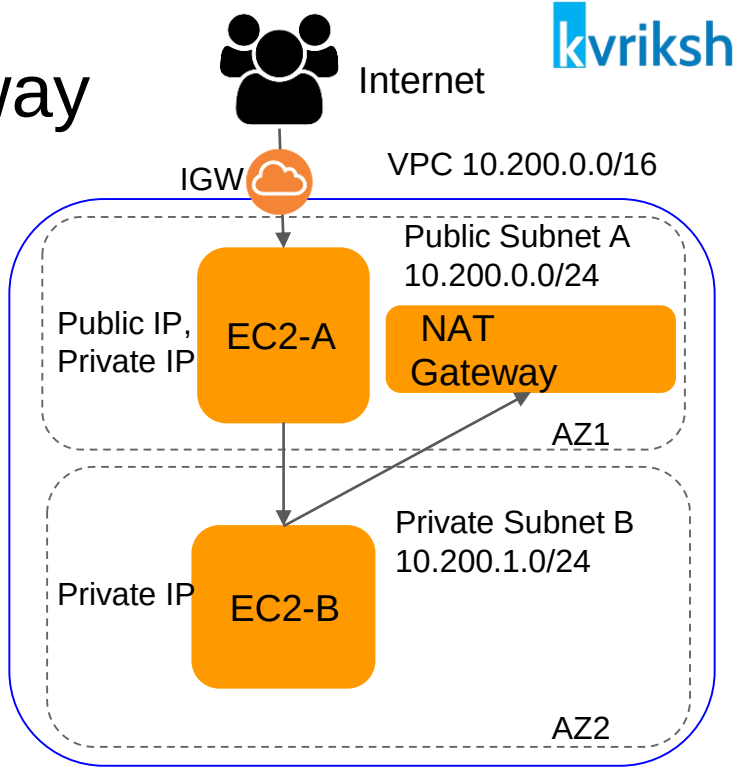
1. Create a new VPC with CIDR 10.200.0.0/16
2. Create Internet Gateway (IGW) and Associate with VPC
3. Create a Public subnet 10.200.0.0/24. Enable auto assign public ip.
4. Create a Private subnet 10.200.1.0/24.
5. Create 2 security groups. For Public EC2 allow SSH from your ip. For Private EC2, allow SSH and, ICMP IPv4 All traffic from VPC CIDR 10.200.0.0/16
6. With previously created ssh key, Launch an EC2 instance (A) in Public Subnet. Instance should have Public IP and Private IP.
7. Launch other EC2 instance (B) in private subnet. Instance should have only Private IP.
8. See if you can connect to EC2-A over Public IP using SSH from your machine
9. See if you can connect to EC2-B over Private IP using SSH from your machine. (You can not)
10. See if you can ping EC2-B from EC2-A and SSH EC2-B from EC2-A (you should be able to)
11. If you successfully connect to EC2-B. Try to ping google.com (You should not be able to access internet. Why?)
12. If you reach upto this step, continue with next assignment

Assignment 2 - Add a NAT Gateway

Continuing with the previous setup

1. Create a NAT Gateway in public subnet
2. Update route table of private subnet and add route for internet traffic (0.0.0.0/0) using NAT Gateway
3. Login to EC2-A and from there login to EC2-B over ssh
4. Now try to access internet from EC2-B instance. Should be successful
   a. ping google.com
5. Delete NAT gateway and release EIP

Watch Video



Private Subnet Route Table

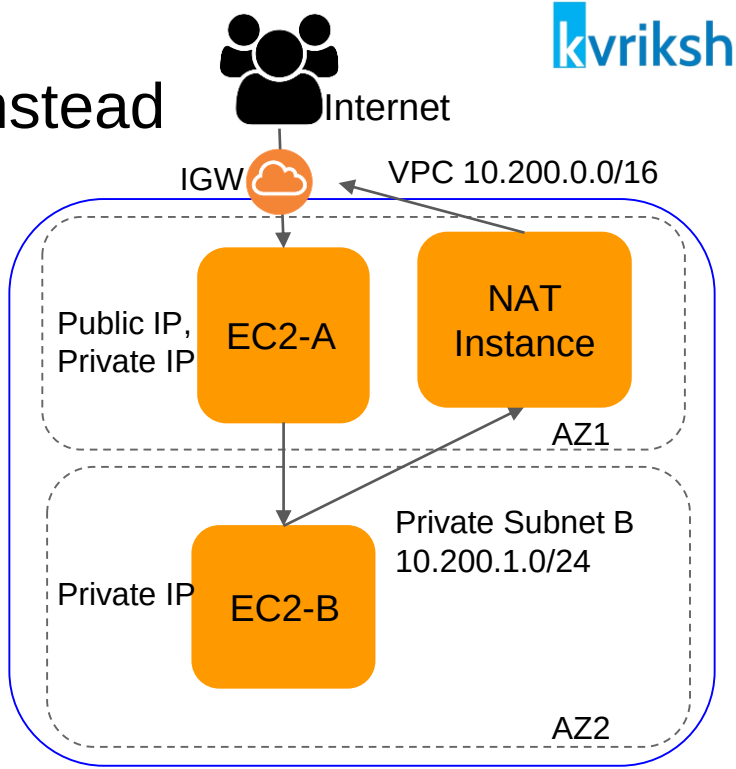
Destination	Target
10.200.0.0/16	local
0.0.0.0/0	NAT-Gateway-ID

Assignment 3 - Use NAT Instance instead of Nat gateway

Continuing with the previous setup

1. Create a NAT EC2 Instance in public subnet using NAT AMI amzn-ami-vpc-nat.
2. Security group to allow traffic on port 80 from private subnet CIDR
3. Disable source/destination check
4. Update route table of private subnet and add route for internet traffic (0.0.0.0/0) using NAT instance id
5. Login to EC2-A and from there login to EC2-B over ssh
6. Now try to access internet from EC2-B instance. Should be successful
   a. ping google.com

Watch Video

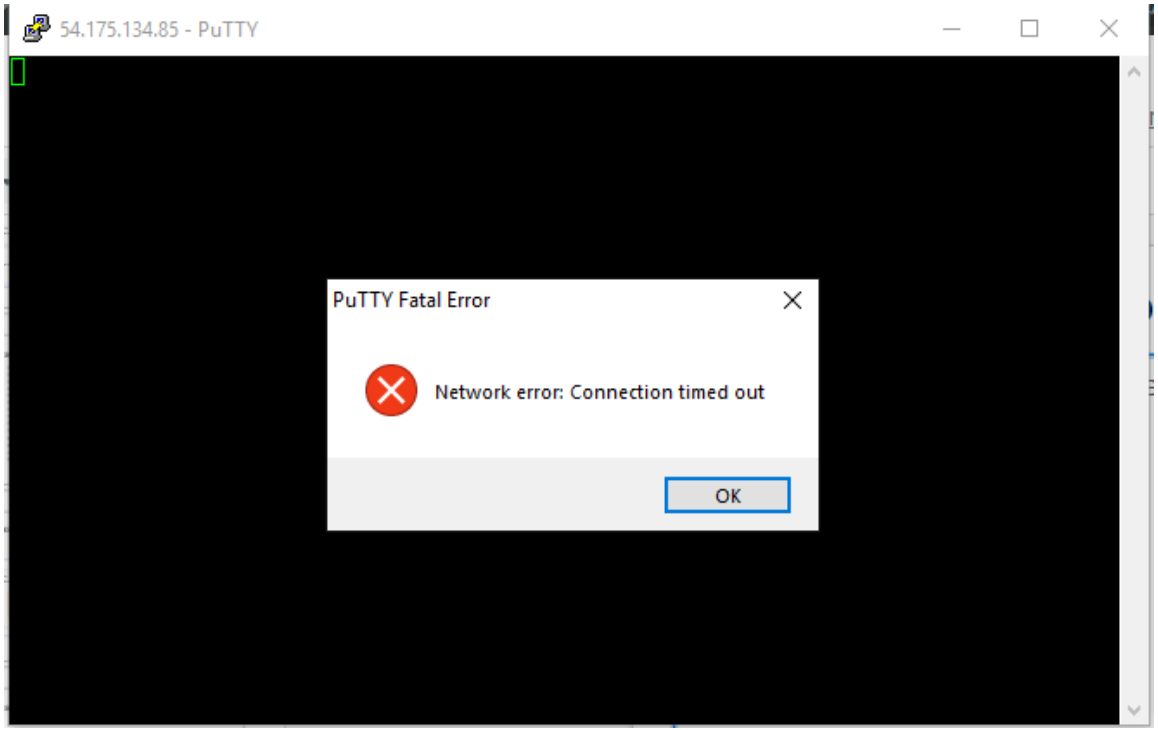


Private Subnet Route Table

Destination	Target
10.200.0.0/16	local
0.0.0.0/0	nat-instance-id

Appendix: Troubleshooting Tips

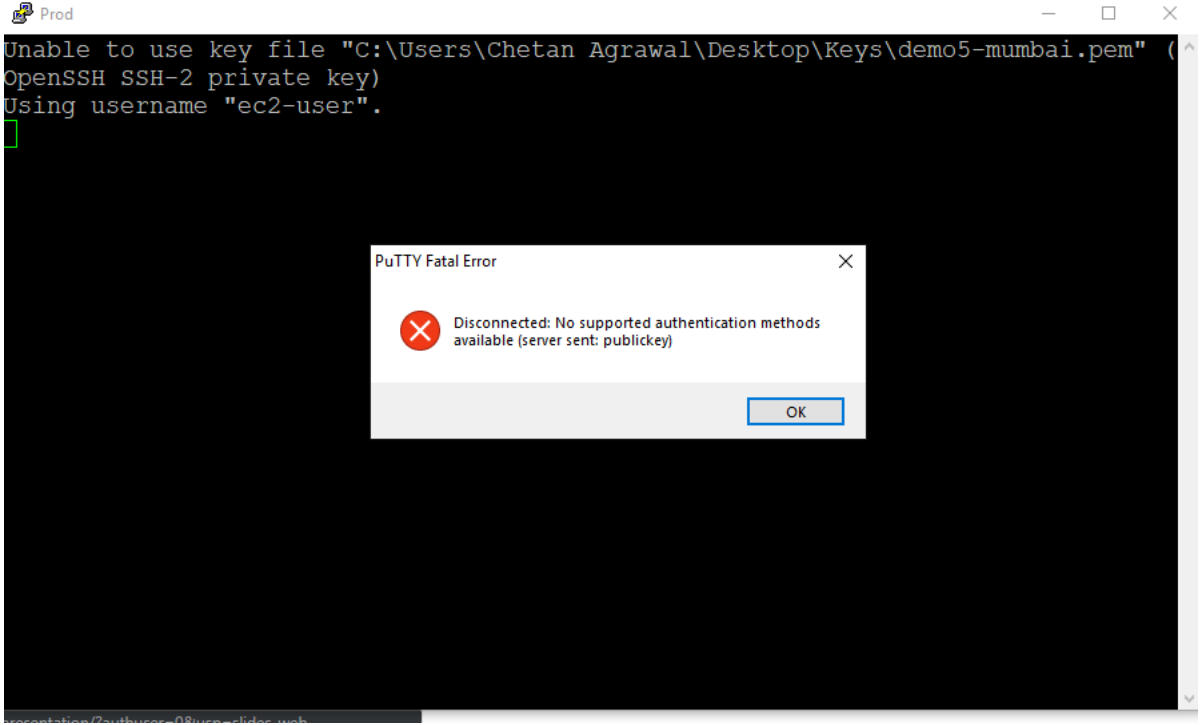
1. “Connection Timed out” error while connecting to EC2 instance using PuTTy



Solution

1. Check that you are using EC2 Public IP (not Private IP)
2. Check Security group for EC2 instance and verify that Port 22 (SSH) is open for MyIP or 0.0.0.0/0
3. Check that EC2 instance is launched in Public Subnet
    a. Get the Subnet ID from EC2 -> Descriptions
    b. Go to VPC -> Subnets -> Filter the subnet by Id
    c. Check the Route Table -> You should see the Route for 0.0.0.0/0 through IGW
4. If you don't see correct route in subnet, you might have missed associating public route table with your subnet. Go to correct route table and associate with your public subnet
5. All above settings are correct but still not able to connect
    a. Go to corresponding public Route table and click on IGW
    b. See if IGW exists. If not -> create new IGW -> Attach with your VPC -> Modify Route table -> Delete existing entry for internet and recreate new with new IGW

2. “No supported Authentication Methods available” error while connecting to EC2 instance using PuTTy



Solution

1. This is most probably due to wrong SSH key you are using
2. Verify that you are using .ppk key and not .pem key file
3. Verify that you are using correct ppk file with which you had launched EC2 instance
    a. Go to EC2 console -> Select your Instance -> Description
    b. Verify the SSH Key name (It's possible that you stored SSH key with different name locally. Name does not matter but it should be corresponding private key)
4. If you have lost your SSH key
    a. Terminate current EC2 instance
    b. Go to EC2 console -> SSH keys -> Generate new key pair
    c. Download .pem file and convert it to .ppk on local machine
    d. Launch new EC2 instance with new SSH key

3. “UNPROTECTED PRIVATE KEY FILE” error while connecting EC2 B from EC2 A over SSH

Error Message:

@@
@@@@@@@@@@@@@
@   WARNING: UNPROTECTED PRIVATE KEY FILE!   @
@@
@@@@@@@@@@@@@
Permissions 0664 for 'key.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "key.pem": **bad permissions**
Permission denied (publickey).

Solution

1. The reason for error is that Your SSH key file permissions are too open.
2. You should change permissions to either 600 or 400.
3. Run command:

    $ chmod 600 key.pem

4. “Enter Passphrase” error while connecting EC2 B from EC2 A over SSH

Error Message:

[ec2-user@ip-10-0-0-91 ~]$ ssh -i key.pem ec2-user@10.0.0.91

Enter passphrase for key 'key.pem':

Solution

1. The reason for error is that your SSH key is not correct
2. Verify that contents of your SSH key that you created on EC2-A instance
3. Make sure SSH key is .pem and not .ppk as we are connecting from Linux to Linux
4. To recreate the key, do the following
    a. On EC2-A instance: rm key.pem
    b. On local machine: Open .pem using notepad and copy the content (CTRL+C)
    c. On EC2-A instance: vim key.pem -> press i -> paste the content by right click -> esc -> :wq -> enter
    d. On EC2-A: chmod 600 key.pem
5. Retry SSH from EC2-A to EC2-B