VPC Lab

- Login to AWS Console
- Search for VPC and Click on it
- Go to Your VPCs and Click on Create VPC
    - Provide the name
    - IP V4 CIDR Block - 10.0.0.0/16
    - Tenancy – default
    - Hit Create to create VPC
- Once VPC is created, it creates
    - Route Table
    - NACL
    - Security Groups
- Create Subnets
    - Provide Name
    - Select VPC That we created
    - IPv4 CIDR – 10.0.1.0/24
    - Click on Create

    Refer Amazon VPC Reserved IP Addresses for the details of Reserved Addresses

- Create Second subnet in another AZ
    - Provide Name
    - Select VPC That we created
    - Select AZ (different from that of Subnet1)
    - IPv4 CIDR – 10.0.2.0/24
    - Click on Create
- Create Internet Gateway (MyIGW)
    - Created and in Detached state by default
    - Attach to VPC that we created
    - You cannot attach more than one Internet Gateway to VPC
- Go to Route Tables

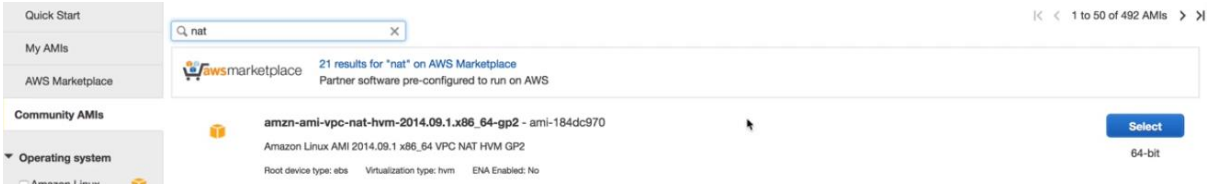
- Create New Route Table to VPC that we created (MyRouteTable)
- Go to Routes from New Route Table that we created
    - Click on Edit to enable Internet Access
    - Add New route
        - Destination 0.0.0.0/0
        - Target is Internet Gateway (Select the one we created)
        - Click on Save

        Now associate the Subnets to become public subnets

    - Go to Subnet Associations → Edit Subnet Associations
        - Associate one subnet to make it as Public
- Select subnet (Public subnet)
    - Subnet actions
    - Modify auto assign IP settings
    - Enable auto assign IP settings
- Launch EC2 Instance
    - In Step #3 (Configure Instance Details)
    - Select Custom VPC (MyVPC)
    - Select Public Subnet (MyPublicSubnet)
- Launch Second EC2 instance with private subnet
- Add Inbound rules to the Security group to enable below ports
        - SSH 22 (To connect to EC2 from your Local System)
        - Http 80 (if you have any application running on port 80)

- SSH to Public EC2 Instance from Git Bash or Terminal
    - Copy pem file into Public EC2 Host to connect to Private EC2 from Public EC2 Instance
- In Priavte EC2 → sudo su
- Try yum update from Private (It won't be success as we don't have Internet access from Private EC2 Host)

NAT Instances & NAT Gateways

- Login to AWS Console
- Click on Launch EC2 Instance
- Select Community AMIs
- Search for NAT and select AMI
- 

- Select Our VPC and Public subnet
- Select Default Security Group
- Select NAT Instance → Actions → Networking → Change Source/Dest Check and Disable Source/Destination **checks on the NAT instance**

- VPC → Route Tables (Default Route table) → Add Route
    - Destination: 0.0.0.0/0
    - Target: NAT Instance
- Login to Public Instance then login to Private Instance from there
- Check Internet access to Private EC2 Instance
- Terminate NAT Instance

NACL

- Create NACL under the VPC we created
- By default all the traffic is denied
- Login to Public Instance
    - yum install httpd -Y

    - service httpd start
    - echo “Welcome to NACL Lab” >> /var/www/html/index.html
    - Add Inbound Rule to open port 80 in a Security Group
    - Open Browser and type <Public IP>
    - You should be able to see the content that you added in index.html
- Go to NACL that we created
- Add Inbound Rules
    - Add Inbound and Outbound Rule to allow All Traffic
    - Assume that you have given Rule number as 100
- Subnet Associations
    - Associate public subnet
- Add another inbound rule for NACL
    - 99 HTTP TCP 80 MyPublic IP Deny
- Refresh my Public EC2 instance in Browser
    - You will not be able to see the index.html content
    - the public EC2 instance should be timed out in Browser
    - Rules are evaluated in Numeric order
- Move the above rule to 101
    - Now ypu should be able to access your application
- NACLs assess before Security Group

VPC Flow Log

- Login to AWS Console
- Go to Cloud Watch and Create Log Group
    - Retention setting – 1 day or anything else
- Go to IAM
    - IAM Role:
        - Create a Role, Select EC2 Service and create Role without attaching policies
        - Open the Role and attach policies

FYI -- https://docs.aws.amazon.com/vpc/latest/userguide/flow-logs-cwl.html

- Create inline Policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

- Also ensure that your role has a trust relationship that allows the flow logs service to assume the role.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "vpc-flow-logs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
```

}

- Go to VPC
- Select the VPC → Actions → Click on Create Flow Log
  - Provide the name
  - Filter → Select All
  - Maximum aggregation interval – 1 minute
  - Destination: Send to Cloud Watch Logs
  - Destination Log Group: Select the log group we created in above step
  - Select IAM Role that we created
  - Create Flow Log
  - Then wait for a minute and go to Cloud Watch
    - Goto Log Group → Log Streams and Verify the Log Events

${version} ${account-id} ${interface-id} ${srcaddr} ${dstaddr} ${srcport} ${dstport} ${protocol} ${packets} ${bytes} ${start} ${end} ${action} ${log-status}

VPC End Points

- Login to AWS Console
- Create a Role
  - Rolename - S3AdminAccess
  - Choose EC2 Service
  - Attach AmazonS3FullAccess Policy
  - This Role allows EC2 Service access to S3
- Go to EC2
- Select the Instance in Private Subnet
  - Actions → Instance Settings → Attach/Replace IAM Role
  - Select the New Role we created
- Move this private subnet to Default NACL under the VPC
- SSH to Public Instance
  - SSH to Private Instance from Public Instance
  - aws s3 ls (If this command does not work, Please follow below steps)
    - `curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"`
    - `sudo yum install unzip`

        - `unzip awscliv2.zip`
        - `sudo ./aws/install`
    - Create test.txt file
    - aws s3 cp test.txt s3://bucketname
- Goto Main route Table or Other Route Table where Route out to NAT is added (Under the VPC which you created)
    - Delete Route out to NAT Instance if any
- Go to Private EC2 instance in Git Bash and see if aws s3 ls command is working
    - It won't work as we deleted Route out to NAT in Route Table
- Go to VPC Dashboard
    - Create End Point
        - Select Amazon S3 Gateway (com.amazonaws.us-east-1.s3)
        - Select VPC that we created
        - Select Main Route Table (Where Private Subnet is associated)
        - Policy – Full Access
    - Verify Routes in Main Route Table
- Go to Private EC2 Instance
    - Type aws s3 ls –region us-east-2

VPC Notes

- Think of a VPC as a virtual Data center in the cloud
- Every Region in the world has Default VPC
- Amazon VPC lets you provision a logically isolated section of the AWS cloud where you can launch AWS resources in a virtual network that you define.
- You have complete control over your virtual networking environment, including selection of your own IP address range, creation of subnets, and configuration of route tables and network gateways.

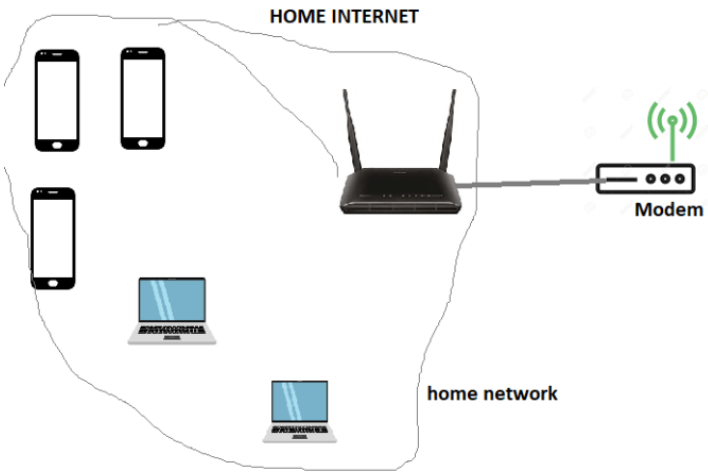
Home Network

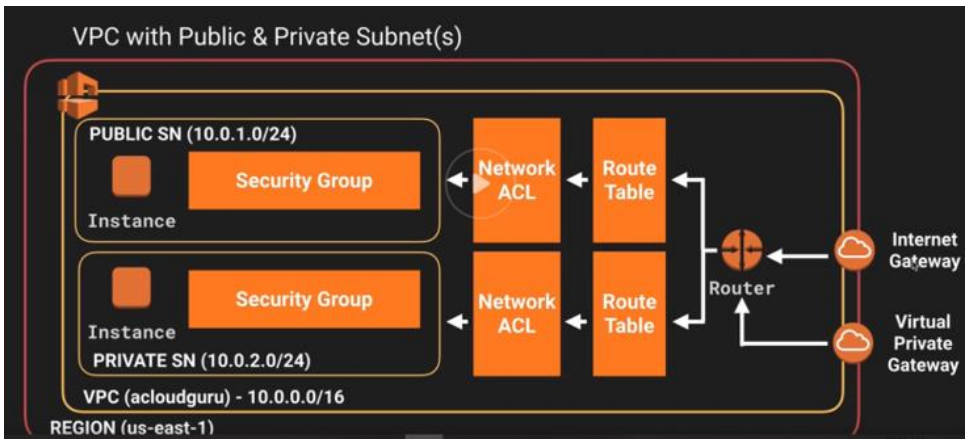
Comparison:

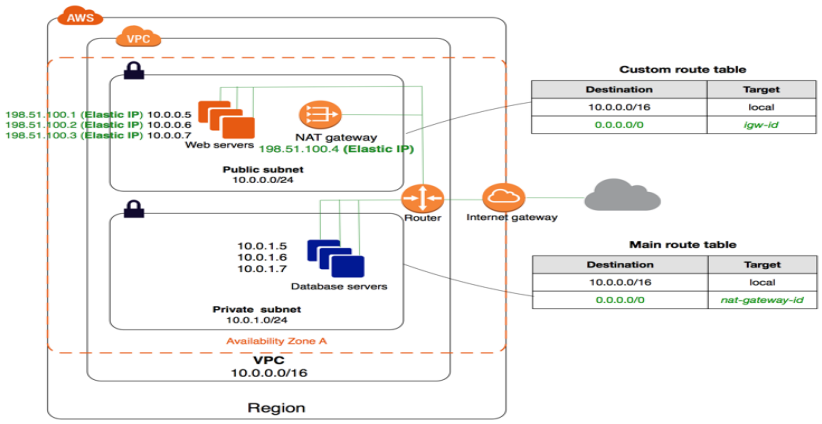
Modem => Internet Gateway

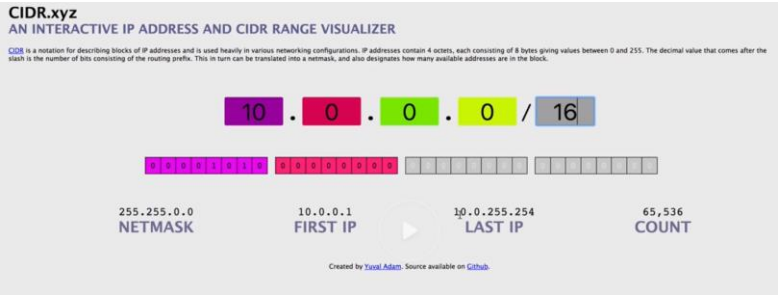
Router => Route Table

Home Network => Subnet


HOME INTERNET
Modem
home network


VPC with Public & Private Subnet(s)
PUBLIC SN (10.0.1.0/24)
Security Group
Instance
Network ACL
Route Table
Security Group
Instance
Network ACL
Route Table
PRIVATE SN (10.0.2.0/24)
Router
Internet Gateway
Virtual Private Gateway
VPC (acloudguru) - 10.0.0.0/16
REGION (us-east-1)


AWS
VPC
198.51.100.1 (Elastic IP) 10.0.0.5
198.51.100.2 (Elastic IP) 10.0.0.6
198.51.100.3 (Elastic IP) 10.0.0.7
Web servers
NAT gateway
198.51.100.4 (Elastic IP)
Public subnet
10.0.0.0/24
Router
Internet gateway
10.0.1.5
10.0.1.6
10.0.1.7
Database servers
Private subnet
10.0.1.0/24
Availability Zone A
VPC
10.0.0.0/16
Region
Custom route table
Destination | Target
10.0.0.0/16 | local
0.0.0.0/0 | igw-id
Main route table
Destination | Target
10.0.0.0/16 | local
0.0.0.0/0 | nat-gateway-id

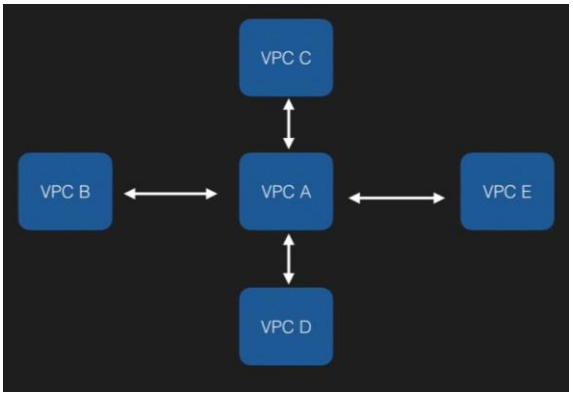


What can you do with a VPC?

- Launch Instances into a subnet of your choosing
- Assign Custom IP address ranges in each subnet
- Configure route tables between subnets
- Create internet Gateway and attach it to our VPC
- Much better Security control over your AWS resources
- Instance Security Groups (Security Groups can span availability zones)
- Subnet network access control lists (ACLs) with which we can block IP addresses
- Default VPC is user friendly, allowing you to immediately deploy instances.
- All Subnets in default VPC have a route out to the Internet.

VPC Peering

- Allows you to connect one VPC with another via a Direct Network route using private IP addresses.
- Instances behave as if they were on the same private network
- You can peer VPC's with other AWS accounts as well as with other VPCs in the same account.
- Peering is in a star configuration, i.e one central VPC peers with 4 others. NO TRANSITIVE PEERING



NAT Instances

- Nat Instance is a EC2 Instance created from Community AMIs
- Disable Source/Destination Check on NAT Instance
- NAT Instances must be in a Public subnet
- There must be a route out of the private subnet to the NAT instance, in order for this to work
- The amount of traffic that NAT instances can support depends on the instance size. If you are bottlenecking, increase the instance size.
- You can create high availability using Autoscaling Groups, multiple subnets in different Azs, and a script to automate failover.
- NAT Instance is behind a Security Group

VPC Flow Logs

- VPC Flow Logs is a feature that enables you to capture information about the IP traffic going to and from Network interfaces in your VPC. Flow log data is stored using Amazon CloudWatch logs.
- After you have created a flow log, you can view and retrieve its data in Amazon CloudWatch logs.
- Flow logs can be created at 3 levels
    - VPC
    - Subnet
    - Network Interface level
- You cannot enable Flow logs for VPCs that are peered with your VPC unless the peer VPC is in your account.
- You can tag Flow logs.

- After you have created a Flow log, you cannot change its configuration. For ex: you cannot associate a different IAM Role with the Flow log

Not all IP Traffic is monitored (Traffic listed below cannot be monitored)

- Traffic generated by instances when they contact the Amazon DNS Server. If you use your own DNS server, then all traffic to that DNS server is logged.
- Traffic generated by a windows instance for Amazon windows license activation.
- Traffic to and from 169.254.169.254 instance metadata.
- DHCP Traffic.
- Traffic to the reserved IP Address for the default VPC router.

VPC End Points

- A VPC endpoint enables you to privately connect your VPC to supported AWS services and VPC end point services powered by PrivateLink without requiring an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection.
- Instances in your VPC do not require public IP addresses to communicate with resources in the service.
- Traffic between your VPC and the other service does not leave the Amazon network.
- Endpoints are virtual devices.
- They are horizontally scaled, redundant, and highly available VPC components that allow communication between instances in your VPC and services without imposing availability risks or bandwidth constraints on your network traffic.

Endpoint Types

- Interface Endpoints

An Elastic Network Interface with a private IP address that serves as an entry point for traffic destined to a supported service. Some of the services that are supported

API Gateway

CloudFormation

CloudWatch

EC2 API

ELB API

Kinesis

SQS

SNS

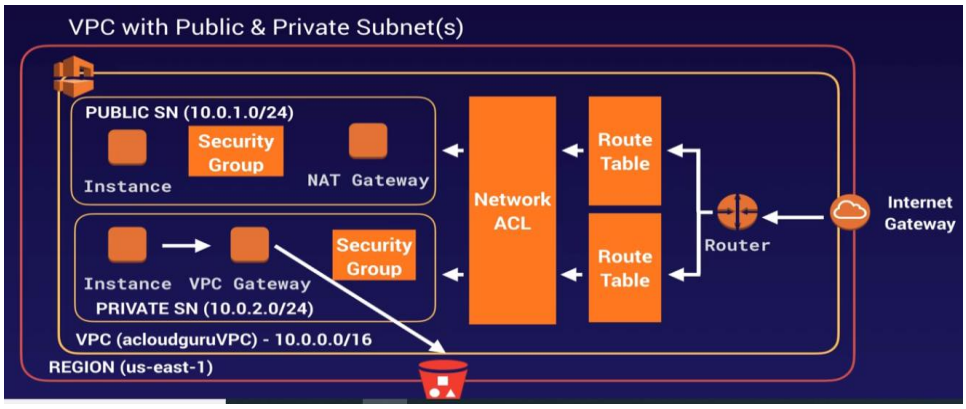
STS

AWS Config

- Currently Gateway Endpoints support

Amazon S3

DynamoDB



Gateway Endpoints

A Gateway is created to connect to AWS Services (S3 and Dynamo DB)

To use Gateway Endpoint add the route to point to Gateway. (Much like Route to IGW or Route to NAT Gateway)