### **Create a VPC – Lab**

1. Console 🡪 Network and Content Delivery 🡪 VPC 🡪 Create VPC 🡪 Create

**2. "Create VPC"**

    Name tag:

    IP4 CIDR block: (Make sure this does not overlap with any potential peer

    Tenancy: Default (Shared) | Dedicated (Dedicated Hardware)

    🡪 Create

This process:

Creates a Default Route table

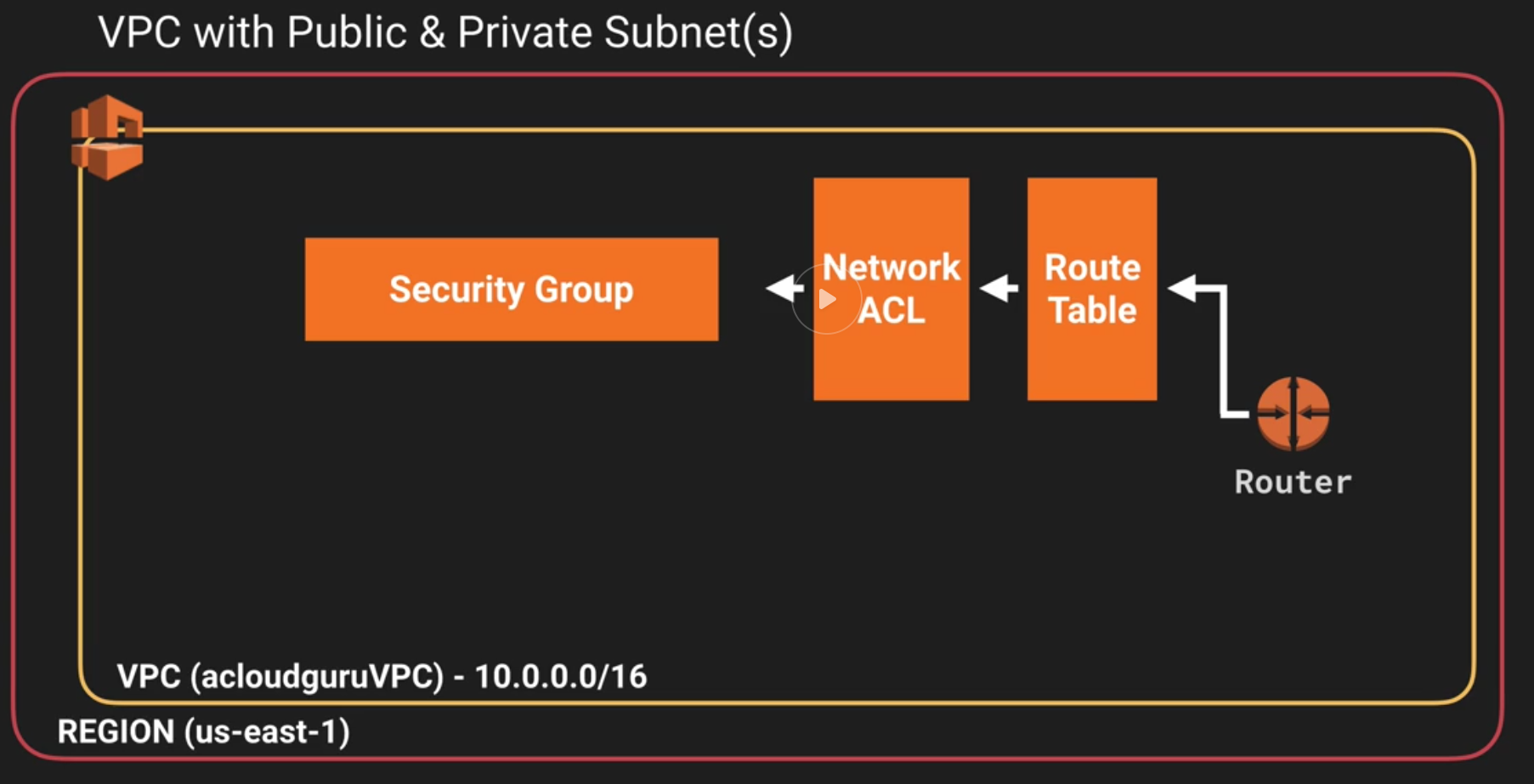
* The default route table creates an entry to allow internal communication

Creates a default Security group

Creates a default NACL

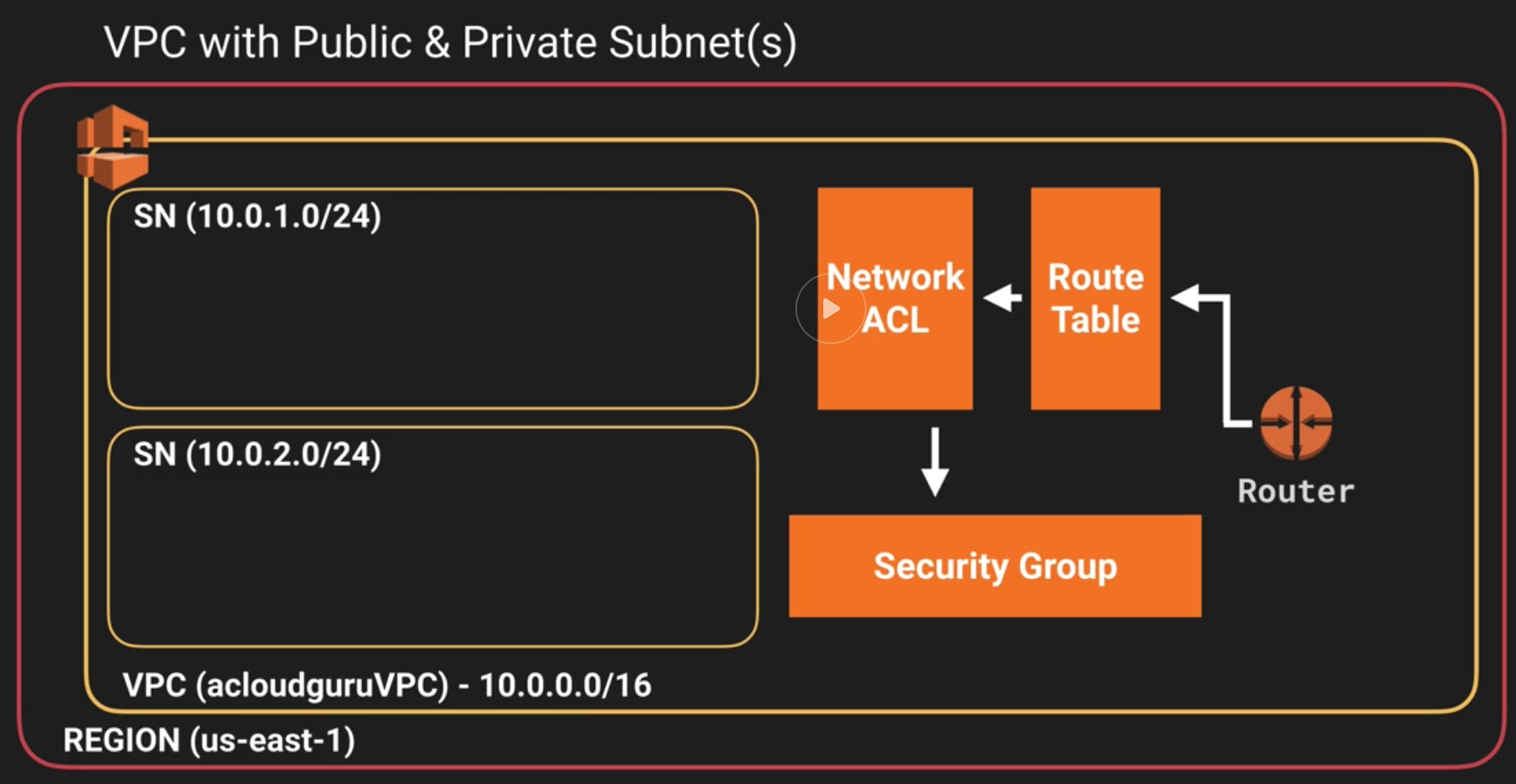
Does not create any subnets

Does not create an Internet Gateway



1. **Select Subnet from left hand menu 🡪 “Create Subnet”**
   * Name: Suggest (subnetip-Range-us-east-1a-PUB
   * Add to VPC
   * Pick AZ
   * IP Subnet Range (10.0.1.0/24) say
   * 🡪 Create
   * Subnet Actions 🡪 Modify auto-assign IP Settings 🡪 Enable…
2. **Select Subnet from left hand menu 🡪 “Create Subnet”**
   * Name: Suggest (subnetip-Range-us-east-1b
   * Add to VPC
   * Pick AZ
   * IP Subnet Range (10.0.2.0/24) say
   * 🡪 Create

Note: You lose 5 addresses to reservation.: X.X.X.0-3 (network, router, DNS, Reserved) and X.X.X.255

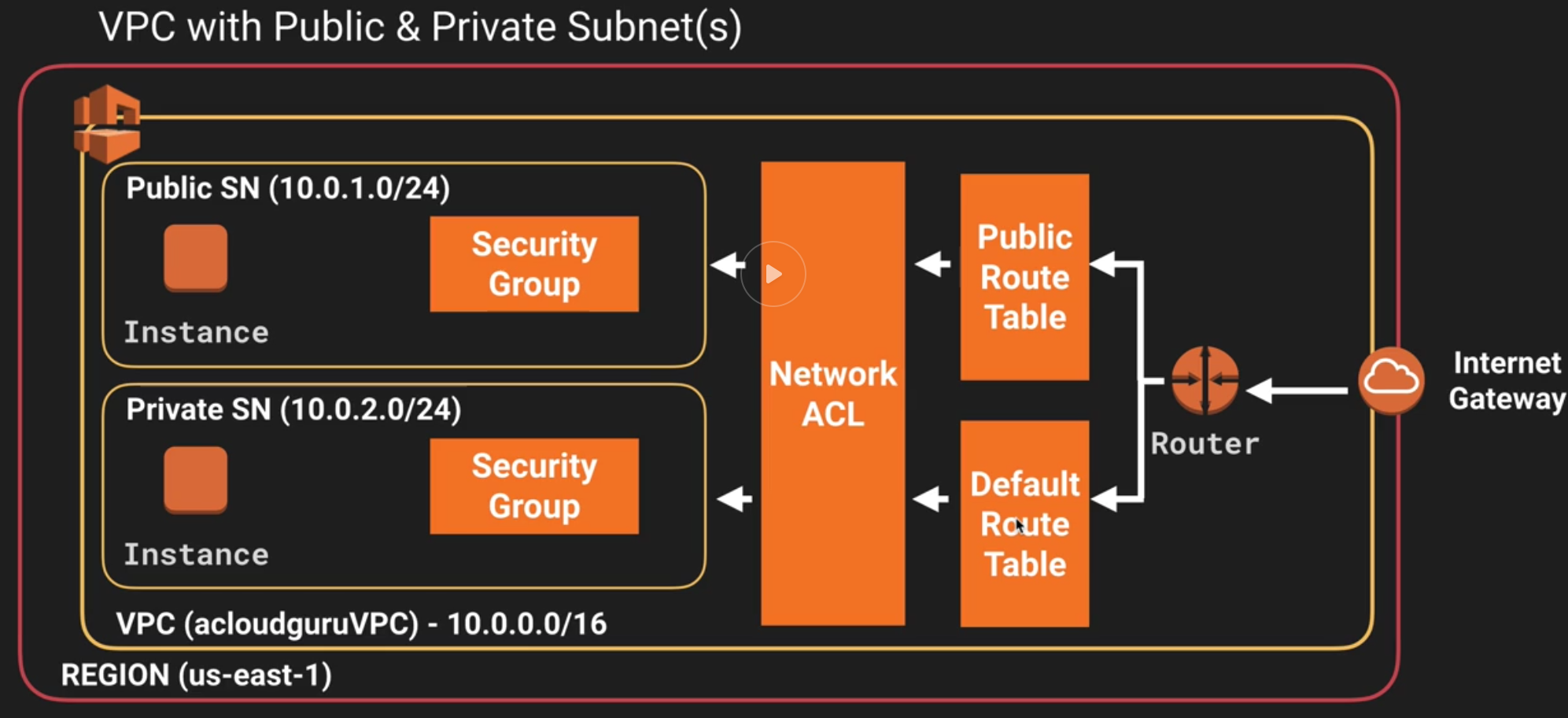


1. **Internet gateways 🡪 Create Internet gateway**
   * Name the Internet Gateway
   * Attach to VPC

* **YOU CAN ATTACH ONLY ONE IGW TO A VPC**

1. **Route Tables**
   * By default, each subnet created is associated with the default route table. This may not be intended behavior so don’t add routes to the default route table but create your own
   1. Create Route table
      1. Give it a name (InternetRouteOut)
      2. Associate with VPC
      3. Enable Internet Access
         1. Route 🡪 add Destination: **0.0.0.0/0** Target: **IGW-Address**
         2. Route 🡪 add Destination: **::/0** Target: **IGW-Address**
      4. **Save**
      5. 🡪 Subnet Associations
      6. Associate Public Subnet with the Route table
      7. Subnets 🡪 Public Subnet 🡪 Subnet Action 🡪 Auto Assign Internet Address
2. **Provision EC2 Instances**
   1. EC2 🡪 Launch WebDMZ Instance
      * Pick AMI from Store
      * Type (e.g. t2.micro)
      * Opportunity to launch into Auto scaling group
      * Add VPC
      * Subnet (Public)(or subnets if auto-scaled)
      * Auto assign public IP address if this is going into the public subnet
      * Add storage
      * Add tag: Name: Instance name
      * Configure Security groups (There will be a default security group in the VPC if you like. They do not span across VPCs)
      * Create New Security group and add to the VPC
        + Name: WebDMZ
        + Add Rules to add http, https, ssh
      * Review and Launch
      * Attach key pair (create one if you don’t have an existing one, download the private key)
   2. EC2 🡪 Launch Private Instance
      * Pick AMI from Store
      * Type (e.g. t2.micro)
      * Opportunity to launch into Auto scaling group
      * Add VPC
      * Subnet (Private)(or subnets if auto-scaled)
      * DONT assign public IP address if this is going into the public subnet
      * Add storage
      * Add tag: Name: Instance name
      * Configure Security groups (There will be a default security group in the VPC if you like. They do not span across VPCs)
      * Create New Security group and add to the VPC
        + Name: PrivateSG
        + Add Rules to add ssh, mySQL/Aurora, HTTP, HTTPS, All IMP (So you can ping)
        + In the source Field add the network address of the public subnet you will come from (10.0.1.0.24)
      * Review and Launch
      * Attach key pair (create one if you don’t have an existing one, download the private key)

You now have this:



You can now ssh into the public instance using the keys you have downloaded but you cannot get into the private instance from the outside as it does not have a public address

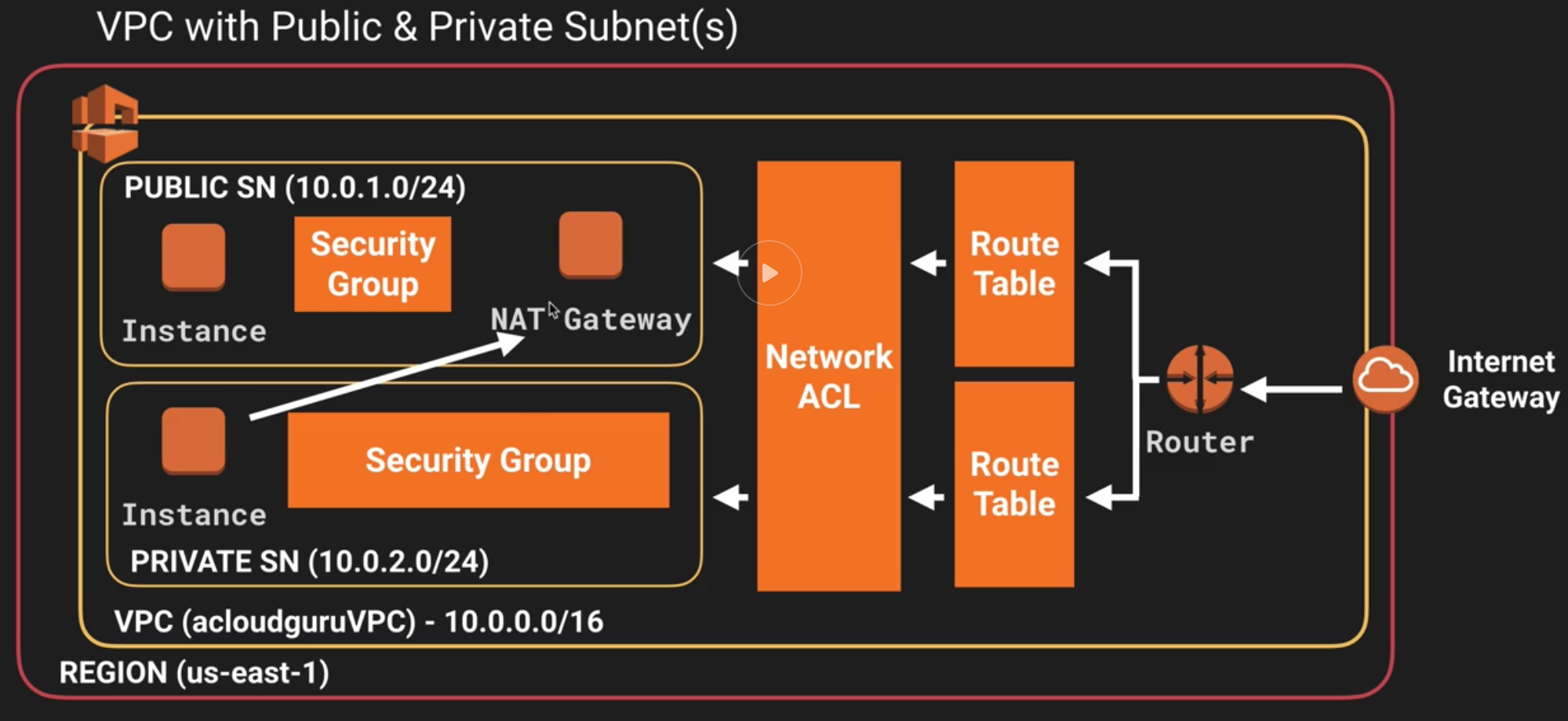
# ssh ec2-user@<ip-address> -i myprivatekey.pem

1. **Configure NAT gateway for private VPC to be able to access external world**
   * VPC 🡪 NAT Gateway(for IPv4)|Egress only NAT gateway (IPv6)
     + Add to the public Subnet
     + Create New EIP
     + Create (takes ~10 minutes)
   * VPC 🡪 route tables (Pick the default route table
     + Add route 0.0.0.0/0 Target: NAT G/W

NOTE:

* **NAT INSTANCE**
  + If you create a NAT Instance instead, disable Source/Destination Check
  + NAT instance s must be in a public subnet
  + There must be a route out of the private subnet to the NAT instance
  + Traffic throughput is a limitation of the NAT instance size
* **NAT GATEWAY**
  + Scales up to 10 Gbps
  + Auto patched/scaled
  + No security group association
  + Automatically assigned a public IP address
  + Update route tables
  + Need one per AZ

You can now run yum update / installs from the private subnet instance



1. **Network ACL**
   * VPC 🡪 Network ACLs
   * Create Network ACL
   * Name the NACL
   * Add to VPC
   * Create
   * **Default NACL created at the time of VPC creation allows all inbound and outbound**
   * **Manually created NACL has “deny all” rules inbound as well as outbound**
   * Add Rules Inbound
     + Rule http, https, ssh from 0.0.0.0/0
   * Add Rules outbound
     + Add http, https 0.0.0.0/0
     + Add 1024-65535 0.0.0.0/0
   * Associate this with the Public Subnet
   * **A subnet can be associated with only one NACL**
   * **If you don’t, it is associated with the default NACL**
   * **NACLs span AZs**