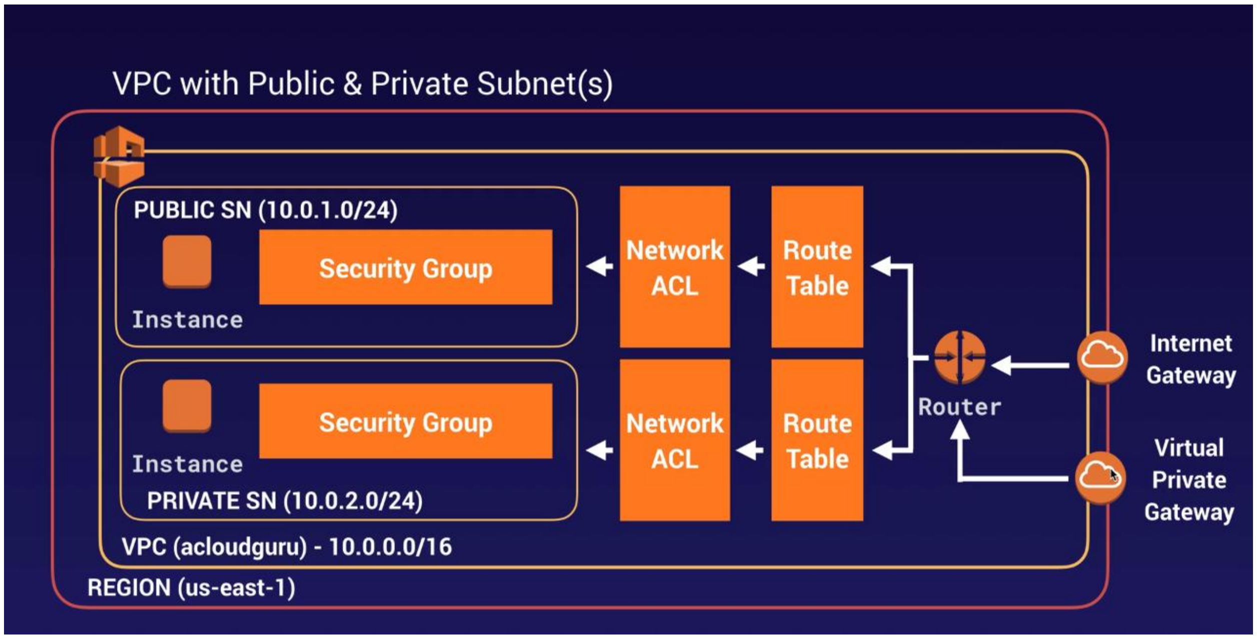
Build a Custom VPC in AWS (without the VPC Wizard)

VPC stands for Virtual Private Cloud

It allows you to provision a logically isolated section of the Amazon Web Services Cloud where you can launch resources in a virtual network that you define. You have complete control over your virtual networking environment, including the selection of your own IP address range, the creation of subnets, and configuration of route tables and network gateways.

In other words, it is your own customized data center in the cloud.

What does a VPC look like?



Simplified Steps

1) Create VPC with a name of your choice (Once the VPC is created a default NACL, Security Group, and Route table is auto-generated)
2) Create a public subnet for our web server (thus **enable** auto assign ipv4 on the subnet)
3) Create a private subnet for our db server (thus do **NOT** enable auto assign ipv4 on the subnet)
4) Create an Internet Gateway & Attach the internet gateway to our VPC
5) Create a public route table and add a route out to the internet (destination- 0.0.0.0/0 target- the newly created internet gateway, & destination- ::/0 target- the newly created internet gateway)
6) Associate the public subnet to the public route table. (The private subnet will remain associated with our default route table which does **NOT** have a route out to the internet)
7) Launch an EC2 instance into the public subnet and **enable auto-assign Public IP**
8) Launch an EC2 instance into the private subnet and **disable auto-assign Public IP**

Detailed Steps

Step 1) Log in to the AWS Management Console. Select the Region you want at the top right. (For this demonstration I chose the Singapore region. Click **Services**. Click **VPC** (located under Networking & Content Delivery). Click **Your VPCs** in the left margin. Then click **Create VPC.**

Name Tag: OurNewCustomVPC ← give it a name like so

IPv4 CIDR block*: 10.0.0.0/16 ← the biggest address block possible

✓Amazon provided IPv6 CIDR block ← select this

Tenancy: Default ← this is multi-tenant hardware. If you select dedicated it will cost a lot of money.

Click **Create.**

Our VPC has now been created. Our newly created VPC automatically comes with a default NACL, a default Security Group, and a default Route Table.

★The default NACL **allows all** outbound and inbound traffic. (A manually created NACL denies all traffic)

★The default security group allows no inbound traffic and allows all outbound traffic

★The default route table contains a local route (which can allow communication between our subnets)

Step 2) We will now create our subnets. Click **subnets** in the left margin. Click **create subnet.**

VPC*: OurNewCustomVPC ← select OurNewCustomVPC

Availability Zone ap-southeast-1a

IPv4 CIDR block: 10.0.1.0/24 🡐 select 10.0.1.0/24

Name tag: 10.0.1.0–ap-southeast-1a-PublicSubnet 🡐 give it a very specific name so we know the CIDR address range, the availability zone, and that it is going to be our public subnet.

IPv6 CIDR block: Don't Assign IPv6 🡐select this

Click **Create**

Step 3) Let's now create another subnet. Click **subnets** in the left margin. Click **create subnet**.

VPC*: OurNewCustomVPC 🡐 select OurNewCustomVPC

Availability Zone ap-southeast-1b

IPv4 CIDR block: 10.0.2.0/24 🡐 select 10.0.2.0/24

Name tag: 10.0.2.0–ap-southeast-1b-PrivateSubnet 🡐 give it a very specific name so we know the CIDR address range, the availability zone, and that it is going to be our private subnet.

IPv6 CIDR block: Don't Assign IPv6 🡐select this

Click **Create**

★ (FYI Amazon reserves the first 4 addresses and last address in each subnet CIDR block. So for 10.0.0.0/24, the following addresses are reserved: 10.0.0.0, 10.0.0.1, 10.0.0.2, 10.0.0.3, & 10.0.0.255)

Step 3 continued) Since, our subnet called 10.0.1.0–ap-southeast-1a-PublicSubnet is going to be our public subnet, we need to be able to launch EC2 instances inside of it that have Public IP addresses.

✓ 10.0.1.0–ap-southeast-1a-PublicSubnet 🡐 select this subnet

Click **Actions**

Click **Modify auto-assign IP settings**

Auto-assign IPv4 ✓Enable auto-assign public IPv4 address

Click **Save**

We are now able to launch EC2 instances into this subnet that will automatically get public IP addresses

Step 4) We need a way to get into our VPC. So we need to Add our Internet Gateway.

Click **Internet Gateways** in the left margin.

Click **Create Internet Gateway.**

Name tag: OurNewInternetGateway ← give it a name like so

✓OurNewInternetGateway ← select OurNewInternetGateway

Click **create internet gateway**

Click **Actions**, click **Attach to VPC** , and select OurNewCustomVPC and click **Attach Internet Gateway.**

★ (FYI we can only have one Internet Gateway attached to a VPC)

Step 5) Let's configure our route tables. We have a default route table (aka main route table) that was automatically created when we created OurNewCustomVPC. Our main route table will be our private route table. Let's create a public route table for our public subnet to utilize.

Click **Route Tables** in the left margin

Create **Route Table**

Name Tag: OurPublicRoute ←any subnet associated with this route will be able to talk to the internet

VPC*: OurNewCustomVPC

Click **Create**

✓ OurPublicRoute ← select OurPublicRoute

Scroll down and click the **Routes Tab**

Click **edit Routes**

Click **add route**

	Destination	Target	Status	Propagated
Already there→	10.0.0.0/16	local	active	No
Already there→	2406:da18:50f:2100::/56 (example of IPv6)	local	active	No
Add this →	0.0.0.0/0	OurNewInternetGateway		
Add this →	::/0	OurNewInternetGateway		

Click **Save Routes**

★The routes that were already there are allowing our subnets to be able to talk to each other. Any subnet inside our VPC using these two routes can communicate to each other over IPv4 and IPv6. Our default/main route table has these two routes as well.

★The routes that we added give us a route out to the internet gateway. So any subnet that is associated with this route table will automatically become public both for IPv4 as well as IPv6

Step 6) Associate the public subnet to the public route table. (The private subnet will remain associated with our default route table which does **NOT** have a route out to the internet)

✓ OurPublicRoute ← select OurPublicRoute

Click the **subnet associations** tab

Click **edit subnet associations**

✓10.0.1.0–ap-southeast-1a-PublicSubnet ← select our public subnet which we called 10.0.1.0–ap-southeast-1a-PublicSubnet

Then click **Save**

Our public route (called OurPublicRoute) is now associated with our public subnet called 10.0.1.0–ap-southeast-1a-PublicSubnet

★Now select our default route/main route (which is our private route) that was created when we created our VPC. If you click the subnet associations tab you will notice that our private route and private subnet are associated with each other.

Step 7) Now we will Launch an EC2 instance into the public subnet and **enable auto-assign Public IP.**

Click **Services** Click **EC2** Click **Launch Instance** select the Linux 2 AMI, select t2.micro.

Click Next: Configure Instance Details

Network: OurNewCustomVPC ← Select OurNewCustomVPC

Subnet: 10.0.1.0–ap-southeast-1a-PublicSubnet ← Select 10.0.1.0–ap-southeast-1a-PublicSubnet

Auto-assign Public Ip: Use subnet setting (Enable) ★ So it is automatically going to get a public IP

Click Next: Add Storage

Click Next: Add Tags

Key: Name

Value: WebServerInPublicSubnet

Click Next: Configure Security Group

★FYI Security Groups do not span VPCs

Create a new security group

SecurityGroupForPublicInstance 🡐give it a name

Then add two inbound rules

	Type	Protocol	Port Range	Source
Add 🡒	SSH	TCP	22	0.0.0.0/0
Add 🡒	HTTP	TCP	80	0.0.0.00, ::/0

★The following are the default rules for a security group that we specifically create:

- Allows no inbound traffic
- Allows all outbound traffic

★Security Groups are **STATEFUL** (if traffic is allowed inbound it is also allowed outbound…think of a security guard that checks you on the way in and remembers you on the way out)

Click Review and Launch

Click Launch

Create a new key pair

SingaporeKeyPair 🡐give it a name

Download the Key Pair

(Save it somewhere safe and open the file in an editor like VSC, brackets, or even notepad. Leave the file open because we will need it for the next lesson. Just minimize it in the meantime.)

Step 8) Now let's launch an instance in our private subnet.

Click **Services** Click **EC2** Click **Launch Instance** select the Linux 2 AMI, select t2.micro.

Click Next: Configure Instance Details

Network: OurNewCustomVPC 🡐 Select OurNewCustomVPC

Subnet: 10.0.2.0–ap-southeast-1b-PrivateSubnet 🡐 Select 10.0.2.0–ap-southeast-1b-PrivateSubnet

Auto-assign Public Ip: Use subnet setting (Disable) ★ we are not going to get a public IP for this instance

Click Next: Add Storage

Click Next: Add Tags

Key: Name

Value: DBServerinPrivateSubnet

Click Next: Configure Security Group

Select an existing Security Group

✓Select the Security Group that was created by default when we created our VPC

Type	Protocol	Port Range	Source
All Traffic	All	All	sg-assortednumbersandletters(default) 🡐will look like this

★A default security group is named `default`, and it has an ID assigned by AWS. The following are the default rules for each default security group:

- Allows all inbound traffic from other instances associated **with the default security group**. The security group specifies itself as a source security group in its inbound rules.
- Allows all outbound traffic from the instance.

Just to reiterate, our default security group above only allows inbound traffic from other instances that are associated with the default security group. So it does **not** allow traffic from the security group that we created for our public instance in the public subnet.

Click Review and Launch

Click Launch

Select an existing key pair

SingaporeKeyPair 🡐 select SingaporeKeyPair

Select that you acknowledge that you have access to the key pair.

Click **Launch Instance**

Now we can SSH into our PUBLIC instance. Click **Services**. Click **EC2.**

✓WebServerInPublicSubnet ← Select WebServerInPublicSubnet

Click the **Connect** Button located towards the top of the screen

✓EC2 Instance Connect (browser-based SSH connection) ← select EC2 Instance Connect (browser-based SSH connection)

Then click **Connect** (You are now SSH'd into your public instance)

Mission Complete. The next lesson will be on Bastions, NAT instances, and NAT Gateways and will pick up from this exact point. If you do not have time to continue on then delete your VPC and delete your instances. Below are important concepts and exam questions pertaining to VPCs.

★**Security Groups**

-Security Groups act as a firewall at the instance level

-Think of a security guard that checks you on the way in and remembers you on the way out

-Unless allowed specifically all **inbound traffic is blocked by default**

-All **Outbound traffic** from the instance is **allowed by default**

-You can specify the source to be either an IP range, single IP address or another security group.

-Security Groups are **STATEFUL** (if traffic is allowed inbound it is also allowed outbound)

-Any changes to a Security Group take effect **immediately**

-EC2 instances can belong to multiple security groups

-Security groups can contain multiple EC2 instances.

-You **cannot block specific IP addresses** with Security Groups, for this you would need a Network Access Control List (NACL)

-You can have up to 10,000 Security Groups per Region (default is 2,500)

-You can have 60 inbound and 60 outbound rules per Security Group

-You can have 16 Security Groups associated to an ENI (default is 5)

★<u>NACLs (Network Access Control Lists)</u>

-Network Access Control List is commonly known as NACL

-VPCs are automatically given a default NACL which **allows all** outbound and inbound traffic.

-However, When you create a NACL it will **deny** all traffic by default

-Each subnet within a VPC must be associated with a NACL

-Subnets can only be associated with 1 NACL at a time. Associating a subnet with a new NACL will remove the previous association.

-If a NACL is not explicitly associated with a subnet, the subnet will automatically be associated with the default NACL.

-NACL has inbound and outbound rules (just like Security Groups)

-Rule can either **allow** or **deny** traffic (unlike Security Groups which can only allow)

-NACLs are **STATELESS** (any allowed inbound traffic is also allowed outbound)

-NACLs contain a numbered list of rules that gets evaluated in order from lowest to highest

-If you needed to block a single IP address you could via NACLs (Security Groups cannot deny)

VPC Practice Questions

Example Question (from an Official Practice Exam):

A company hosts a popular web application that connects to an Amazon RDS MySQL DB instance running in a private VPC subnet that was created with default ACL settings. The web servers must be accessible only to customers on an SSL connection. The database should only be accessible to web servers in a public subnet.

Which solution meets these requirements without impacting other running applications? (choose 2)

Options are :

A) Create a network ACL on the DB subnet, allow MySQL port 3306 inbound for web servers, and deny all outbound traffic
B) Create a network ACL on the web server's subnet, allow HTTPS port 443 inbound, and specify the source as 0.0.0.0/0
C) Create a DB server security group that allows MySQL port 3306 inbound and specify the source as a web server security group

D) Create a web server security group that allows HTTPS port 443 inbound traffic from Anywhere (0.0.0.0/0) and apply it to the web servers
E) Create a DB server security group that allows the HTTPS port 443 inbound and specify the source as a web server security group

Explanation

A VPC automatically comes with a modifiable default network ACL. By default, it allows all inbound and outbound IPv4 traffic. Custom network ACLs deny everything inbound and outbound by default but in this case a default network ACL is being used Inbound connections to web servers will be coming in on port 443 from the Internet so creating a security group to allow this port from 0.0.0.0/0 and applying it to the web servers will allow this traffic The MySQL DB will be listening on port 3306. Therefore, the security group that is applied to the DB servers should allow 3306 inbound from the web servers security group The DB server is listening on 3306 so creating a rule allowing 443 inbound will not help. The Answer is C & D.

Question

You have an EC2 instance with a Security Group attached. This security group is configured to only allow traffic to/from 10.0.0.0/16. A colleague has also configured a NACL on the private subnet that the instance resides on, and this NACL is configured to block all traffic, except where the destination is in 10.0.1.0/24. What will happen when the instance attempts to access IP 192.168.0.12 on port 80?

Options are:

A) The security group will block the traffic before it is evaluated by the NACL

B) The traffic will be blocked simultaneously by the Security Group and NACL

C) The traffic will be allowed as it is still within a private range

D) The NACL will block the traffic before it is evaluated by the security group

Explanation

With outbound traffic, Security Groups are evaluated first, then NACLs. The security group is configured to only allow traffic where the destination is 10.0.0.0/16, and as 192.168.0.12 does not fall within this range it will be blocked by the security group before it reaches the NACL. Answer is A.

Question

You need to add a route to your routing table that will allow connections to the internet from your subnet. Which of the following routes should you add?

A) Destination: 192.168.1.258/0 --> Target: your Internet gateway
B) Destination: 0.0.0.0/0 --> Target: 0.0.0.0/24
C) Destination: 0.0.0.0/33 --> Target: your virtual private gateway
D) Destination: 0.0.0.0/0 --> Target: your Internet gateway

Explanation

When setting a Custom Route Table, the destination should be 0.0.0.0/0, and the target should be the Internet gateway. Answer is D.

Question

Which of the following statements illustrate the difference between inbound rules and outbound rules in security groups?

A) Inbound rules control the source of the web traffic; outbound rules control the destination for the web traffic.
B) Inbound rules control the incoming web traffic allowed to reach an instance; outbound rules control the outgoing web traffic allowed to leave the instance.
C) Inbound rules control the web traffic leaving an instance; outbound rules control the web traffic reaching the instance.
D) Inbound rules control the resources put into a VPC; outbound rules control the resources allowed to be removed from a VPC.

Explanation

Security groups use inbound rules to control web traffic that’s coming in and outbound rules to control web traffic that’s going out. In addition, that control is applied at the traffic’s source for inbound and destinations for outbound. So, 'Inbound rules control the web traffic leaving an instance' is wrong because it is the opposite of what inbound and outbound rules are supposed to do. Rules concern web traffic control, not resource input and output. Answer is A & B
Resources

Question

You have an EC2 instance with a Security Group attached. This security group is configured to only allow inbound traffic from 192.168.0.0/24. A collegue has also configured a NACL on the subnet that the instance resides on, and this NACL is configured to block all traffic, except where the source or destination is in 192.168.0.0/24. What will happen when an instance with an IP of 192.168.1.12 tries to connect to your instance on port 80?

A) The security group will block the traffic before it is evaluated by the NACLSelected
B) The traffic will be blocked simultaneously by the Security Group and NACL
C) The NACL will block the traffic before it is evaluated by the security group
D) The traffic will be allowed as it is still within a private range

Explanation

With inbound traffic, NACLs are evaluated before Security Groups. As the NACL is configured to only allow traffic from 192.168.0.0/24 and the IP 192.168.1.12 does not fall within that range, it will be blocked by the NACL before reaching the Security Groups. Answer is C.

Question

A data processing application in AWS must pull data from an Internet service. A Solutions Architect is to design a highly available solution to access this data without placing bandwidth constraints on the application traffic.
Which solution meets these requirements?

A) Launch a NAT gateway and add routes for 0.0.0.0/0
B) Attach a VPC endpoint and add routes for 0.0.0.0/0
C) Attach an Internet gateway and add routes for 0.0.0.0/0
D) Deploy NAT instances in a public subnet and add routes for 0.0.0.0/0

Explanation

An Internet gateway is a horizontally scaled, redundant, and highly available VPC component that allows communication between instances in your VPC and the Internet. It therefore imposes no availability risks or bandwidth constraints on your network traffic. Answer is C.

Note: NAT gateway is also a highly available architecture and is used to enable instances in a private subnet to connect to the internet or other AWS services, but prevent the internet from initiating a connection with those instances.