

Problem Statement:

Development Network:

1. Design and build 2-tier architecture with two subnets named web and db and launch instances in both subnets and name them as per the subnet names.
2. Make sure only the web subnet can send internet requests.
3. Create peering connection between production network and development network.
4. Setup connection between db subnets of both production network and development network respectively

Solution:

Creating Development network.

- 1) Go to the VPC console and create a VPC.

← → ⌂ ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#Home:

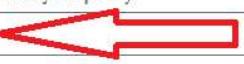
The screenshot shows the AWS VPC Dashboard. At the top, there are navigation icons and a search bar. Below the header, the title "VPC dashboard" is displayed next to a close button. To the right of the title are two buttons: "Create VPC" (highlighted with a red circle) and "Launch EC2 Instances". A note below the buttons states: "Note: Your Instances will launch in the Asia Pacific region." On the left side, there is a sidebar with a tree view under "Virtual private cloud" containing items like "Your VPCs", "Subnets", "Route tables", "Internet gateways", "Egress-only internet gateways", "DHCP option sets", and "Classic IPs". On the right side, there is a section titled "Resources by Region" with three items: "VPCs" (1 item, Asia Pacific), "Subnets" (3 items, Asia Pacific), and "Route Tables" (1 item, Asia Pacific). Each resource item has a "See all regions" link.

- 2) Under VPC settings choose VPC only, give a name for the VPC , give the CIDR and click create VPC.

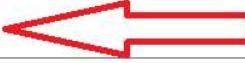
VPC settings

Resources to create [Info](#)
Creates only the VPC resource or the VPC and other networking resources.

VPC only **VPC and more**

Name tag *optional*
Creates a tag with a key of 'Name' and a value that you specify.
my-development-network 

IPv4 CIDR block [Info](#)
 IPv4 CIDR manual input
 IPAM-allocated IPv4 CIDR block

IPv4 CIDR
10.0.0.0/16 

CIDR block size must be between /16 and /28.

IPv6 CIDR block [Info](#)
 No IPv6 CIDR block
 IPAM-allocated IPv6 CIDR block
 Amazon-provided IPv6 CIDR block
 IPv6 CIDR owned by me

Tenancy [Info](#)
Default 

Tags
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value - optional
<input type="text"/> Name 	<input type="text"/> my-development-network  
Add tag	

- 3) So we can see the VPC is created. Under resource map we can see details mapping of our production VPC.

✓ You successfully created vpc-056eb1318e1c02a1d / my-development-network

VPC > Your VPCs > vpc-056eb1318e1c02a1d

vpc-056eb1318e1c02a1d / my-development-network

Details Info

VPC ID vpc-056eb1318e1c02a1d	State Available	DNS hostname Disabled
Tenancy Default	DHCP option set opt-0ec3b1ca7d51a1660	Main route table rtb-000bb5b1
Default VPC No	IPv4 CIDR 10.0.0.0/16	IPv6 pool -
Network Address Usage metrics Disabled	Route 53 Resolver DNS Firewall rule groups -	Owner ID 010526:

Resource map Info

VPC Show details

Your AWS virtual network

Subnets (0)

Subnets within this VPC

my-development-network

- 4) Now we have to create two subnets named web and db. Condition: Make sure only the web subnet can send internet requests. that means web subnet is public whereas db subnet is private.
- 5) Under VPC dashboard click subnets and create subnet.

The screenshot shows the AWS VPC dashboard. On the left, there's a sidebar with options like EC2 Global View, Filter by VPC, Virtual private cloud (with Your VPCs, Subnets circled in red), Route tables, Internet gateways, Egress-only internet gateways, and VPC endpoints.

The main panel is titled "Subnets (3) Info". It has a search bar and a table with three rows:

	Name	Subnet ID
<input type="checkbox"/>	-	subnet-0db899856299693
<input type="checkbox"/>	-	subnet-06c199c19e1c19c5
<input type="checkbox"/>	-	subnet-0f8f45b511dfc2ac7

6) Choose the VPC and give the subnet name my-web-subnet. Scroll down and click on add new subnet.

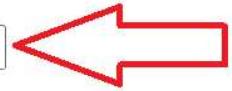
Create subnet Info

VPC

VPC ID

Create subnets in this VPC.

vpc-056eb1318e1c02a1d (my-development-network) ▾



Associated VPC CIDRs

IPv4 CIDRs

10.0.0.0/16

Subnet settings

Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

Subnet name

Create a tag with a key of 'Name' and a value that you specify.

my-web-subnet

The name can be up to 256 characters long.

Availability Zone Info

Choose the zone in which your subnet will reside, or let Amazon choose one for you.

No preference



IPv4 VPC CIDR block Info

Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

10.0.0.0/16



Subnet's IPv4 CIDR block must be a valid CIDR block.

Subnet settings

Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

Subnet name

Create a tag with a key of 'Name' and a value that you specify.

The name can be up to 256 characters long.

Availability Zone [Info](#)

Choose the zone in which your subnet will reside, or let Amazon choose one for you.



IPv4 VPC CIDR block [Info](#)

Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.



IPv4 subnet CIDR block



▼ Tags - optional

Key

Value - optional



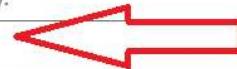
You can add 49 more tags.

- 7) Give the subnet name my-db-subnet, give the CIDR value so that it doesn't overlap and click create subnet.

Subnet 2 of 2

Subnet name

Create a tag with a key of 'Name' and a value that you specify.



The name can be up to 256 characters long.

Availability Zone [Info](#)

Choose the zone in which your subnet will reside, or let Amazon choose one for you.

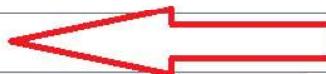


IPv4 VPC CIDR block [Info](#)

Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.



IPv4 subnet CIDR block



16 IPs



▼ Tags - optional

Key

Value - optional



Rem

[Add new tag](#)

You can add 49 more tags.

[Remove](#)

[Add new subnet](#)

[Cancel](#)

8) So my two subnets are created.

Subnets (2) [Info](#)

Find resources by attribute or tag

Subnet ID : subnet-00caa5c5f120fc977 [X](#)

Subnet ID : subnet-077967853d13851b7 [X](#)

<input type="checkbox"/>	Name	Subnet ID	State
<input type="checkbox"/>	my-web-subnet	subnet-00caa5c5f120fc977	✓ Av:
<input type="checkbox"/>	my-db-subnet	subnet-077967853d13851b7	✓ Av:

- 9) To give internet access to web subnet we have to create internet gateway. So under VPC dashboard click internet gateway and create internet gateway.

The screenshot shows the AWS VPC Dashboard. On the left, there's a sidebar with options like EC2 Global View, Filter by VPC, Virtual private cloud (Your VPCs, Subnets, Route tables, Internet gateways), and Egress-only internet gateways. The 'Internet gateways' option is highlighted with a red oval. The main pane shows the 'Internet gateways (1)' section with a single entry: Name '-' and Internet gateway ID 'igw-010cd8eca6256'. There's also a search bar at the top of the main pane.

- 10) Give a name tag and click create internet gateway.

Create internet gateway Info

An internet gateway is a virtual router that connects a VPC to the internet. To create a new internet gateway, provide the required information for the gateway below.

Internet gateway settings

Name tag

Creates a tag with a key of 'Name' and a value that you specify.

my-internet-gateway-web-subnet



Tags - optional

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to identify your resources or track your AWS costs.

Key

Name

Value - optional

my-internet-gateway-web-subnet

Re

Add new tag

You can add 49 more tags.

Cancel

Create

- 11) We can see our internet gateway is created but it is detached. So we need to attach it.

[VPC](#) > [Internet gateways](#) > igw-0f312736cd9ec3cbc

igw-0f312736cd9ec3cbc / my-internet-gateway-v

Details [Info](#)

Internet gateway ID
 igw-0f312736cd9ec3cbc

State
 Detached

Tags

 Search tags

Key	Value
-----	-------

Name	my-internet-gateway-web-subnet
------	--------------------------------

- 12) Click on actions and attach to VPC.

[VPC](#) > [Internet gateways](#) > igw-0f312736cd9ec3cbc

igw-0f312736cd9ec3cbc / my-internet-gateway-v

Details [Info](#)

Internet gateway ID
 igw-0f312736cd9ec3cbc

State
 Detached

Tags

 Search tags

Key	Value
-----	-------

Name	my-internet-gateway-web-subnet
------	--------------------------------

- 13) Choose the VPC we created and attach internet gateway.

Attach to VPC (igw-0f312736cd9ec3cbc) Info

VPC

Attach an internet gateway to a VPC to enable the VPC to communicate with the internet. Specify the VPC to attach.

Available VPCs

Attach the internet gateway to this VPC.

Use: "vpc-056eb1318e1c02a1d"

vpc-056eb1318e1c02a1d - my-development-network



- 14) Now we can see our gateway is attached and its showing our VPC ID .

[VPC](#) > [Internet gateways](#) > igw-0f312736cd9ec3cbc

igw-0f312736cd9ec3cbc / my-internet-gateway-w

Details Info

Internet gateway ID

igw-0f312736cd9ec3cbc

State

Attached

Tags

Search tags

Key

Value

Name

my-internet-gateway-web-subnet

15) So under VPC dashboard click route table and we can see route table associated with our VPC . click on routes.

The screenshot shows the AWS VPC Dashboard. On the left, there's a sidebar with various options like EC2 Global View, Filter by VPC, Virtual private cloud (Your VPCs, Subnets, Route tables), Internet gateways, Egress-only internet gateways, DHCP option sets, Elastic IPs, Managed prefix lists, Endpoints, Endpoint services, NAT gateways, Peering connections, Security (Network ACLs, Security groups), DNS firewall (Rule groups, Domain lists), and Network Firewall. The 'Route tables' link is circled in red. The main panel shows 'Route tables (1/2)' with a table:

Name	Route table ID
-	rtb-02028bab14fd3c9d
my-dev-route table	rtb-000bb5b37ca536677

Below this, it says 'rtb-000bb5b37ca536677 / my-dev-route table' and has tabs for Details, Routes, Subnet associations, Edge associations, and Router. The 'Routes' tab is circled in red. Under 'Details', there are fields for Route table ID (rtb-000bb5b37ca536677), Main (Yes), VPC (vpc-056eb1318e1c02a1d | my-development-network), Owner ID (010526284257), and a large 'Details' section.

16) Here we can see only 1 route which states that our all resources inside our vpc can communicate

with each other but there is no route to connect to internet. So we will make a new route.

rtb-000bb5b37ca536677 / my-dev-route table

Details | **Routes** | Subnet associations | Edge associations | Route propagat

Routes (1)

Filter routes

Destination	Target
10.0.0.0/16	local

- 17) Under VPC dashboard click route tables and create route table.

VPC dashboard

EC2 Global View

Filter by VPC

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet

Route tables (1/2) [Info](#)

Name	Route
my-dev-route table	rtb-00
-	rtb-02

- 18) Give a name and choose the development VPC and create route table.

Create route table Info

A route table specifies how packets are forwarded between the subnets within your VPC, the internet connection.

Route table settings

Name - optional

Create a tag with a key of 'Name' and a value that you specify.

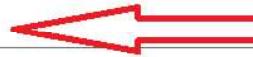
my-route-internet



VPC

The VPC to use for this route table.

vpc-056eb1318e1c02a1d (my-development-network)



Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use your resources or track your AWS costs.

Key

Value - optional

Name



my-route-internet



Add new tag

You can add 49 more tags.

Cancel

- 19) The route table is created . we can see there is a default route but we want route to internet. So click on edit routes.

Route table rtb-01f2fab9f04937983 | my-route-internet was created successfully.

VPC > Route tables > rtb-01f2fab9f04937983

rtb-01f2fab9f04937983 / my-route-internet

Details	
Route table ID	Main
<input type="checkbox"/> rtb-01f2fab9f04937983	<input type="checkbox"/> No
VPC	Owner ID
vpc-056eb1318e1c02a1d my-development-network	<input type="checkbox"/> 010526284257

Routes (1)

Destination	Target
10.0.0.0/16	local

- 20) Under destination give 0.0.0.0/0 which is route to internet via target is our internet gateway which we created and click save changes.

Edit routes

Destination	Target
10.0.0.0/16	local
<input type="text" value="0.0.0.0/0"/> X	<input type="text" value="local"/>
<input type="text" value="0.0.0.0/0"/> X	<input type="text" value="Internet Gateway"/>
<input type="text" value="0.0.0.0/0"/> X	<input type="text" value="igw-0f312736cd9ec3cbc"/>

- 21) Now we have to associate our subnets with the route table. We have two subnets. One can access internet and the other one cant. So select the route table which can access internet and click subnet associations.

Route tables (1/3) [Info](#)

Find resources by attribute or tag

Name	Route table ID	Explicit subnet associ...	E
<input type="checkbox"/> my-dev-route table	rtb-000bb5b37ca536677	-	-
<input type="checkbox"/>	rtb-02028bab14fd3c9d	-	-
<input checked="" type="checkbox"/> my-route-internet	rtb-01f2fab9f04937983	-	-

rtb-01f2fab9f04937983 / my-route-internet

[Details](#) | [Routes](#) | [Subnet associations](#) **(Subnet associations)** | [Edge associations](#) | [Route propagation](#) | [Tags](#)

Details

Route table ID	Main	Explicit subnet
<input type="checkbox"/> rtb-01f2fab9f04937983	<input type="checkbox"/> No	-
VPC	Owner ID	
vpc-056cb1318e1c02a1d my-development-network	<input type="checkbox"/> 010526284257	

22) We can see both of our subnets available.

Click edit subnet associations.

Subnets without explicit associations (2)

The following subnets have not been explicitly associated with any route tables and are therefore associated with the default route table.

Find subnet association

Name

▼

Subnet ID

my-web-subnet

[subnet-00caa5c5f120fc977](#)

my-db-subnet

[subnet-077967853d13851b7](#)

- 23) Choose my-web subnet and click save association.

[VPC](#) > [Route tables](#) > [rtb-01f2fab9f04937983](#) > [Edit subnet associations](#)

Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (1/2)

Filter subnet associations

Name

▼

Subnet ID

▼

IP

my-web-subnet

[subnet-00caa5c5f120fc977](#)

10

my-db-subnet

[subnet-077967853d13851b7](#)

10

Selected subnets

[subnet-00caa5c5f120fc977 / my-web-subnet](#) X

- 24) Similarly associate the db subnet with my-dev route table which cant access internet. So we can see that my web subnet has 2 routes associated

with it and my db subnet has 1 route associated with it.

Find resources by attribute or tag						
	Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR
<input type="checkbox"/>	-	subnet-0db89985629969379	Available	vpc-0ed4fb8863de2b7ae	172.31.0.0/20	-
<input type="checkbox"/>	-	subnet-06c199c19e1c19c57	Available	vpc-0ed4fb8863de2b7ae	172.31.16.0/20	-
<input type="checkbox"/>	-	subnet-0fb45b511dfc2ac7	Available	vpc-0ed4fb8863de2b7ae	172.31.32.0/20	-
<input checked="" type="checkbox"/>	my-web-subnet	subnet-00caa5c5f120fc977	Available	vpc-056eb1318e1c02a1d my...	10.0.0.0/28	-
<input type="checkbox"/>	my-db-subnet	subnet-077967853d13851b7	Available	vpc-056eb1318e1c02a1d my...	10.0.1.0/28	-

subnet-00caa5c5f120fc977 / my-web-subnet	Edit route table
Details Flow logs Route table (highlighted) Network ACL CIDR reservations Sharing Tags	
Route table: rtb-01f2fab9f04937983 / my-route-internet	

Routes (2)	Edit route table
Filter routes	
Destination	Target
10.0.0.0/16	local
0.0.0.0/0	igw-0f312736cd9ec3cbc

Subnets (1/5) Info	Last updated 4 minutes ago					
Find resources by attribute or tag						
	Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR
<input type="checkbox"/>	-	subnet-0db89985629969379	Available	vpc-0ed4fb8863de2b7ae	172.31.0.0/20	-
<input type="checkbox"/>	-	subnet-06c199c19e1c19c57	Available	vpc-0ed4fb8863de2b7ae	172.31.16.0/20	-
<input type="checkbox"/>	-	subnet-0fb45b511dfc2ac7	Available	vpc-0ed4fb8863de2b7ae	172.31.32.0/20	-
<input type="checkbox"/>	my-web-subnet	subnet-00caa5c5f120fc977	Available	vpc-056eb1318e1c02a1d my...	10.0.0.0/28	-
<input checked="" type="checkbox"/>	my-db-subnet	subnet-077967853d13851b7	Available	vpc-056eb1318e1c02a1d my...	10.0.1.0/28	-

subnet-077967853d13851b7 / my-db-subnet	Edit route table
Details Flow logs Route table (highlighted) Network ACL CIDR reservations Sharing Tags	
Route table: rtb-000bb5b37ca536677 / my-dev-route table	
Routes (1)	

[Filter routes](#)

Destination	Target
10.0.0.0/16	local

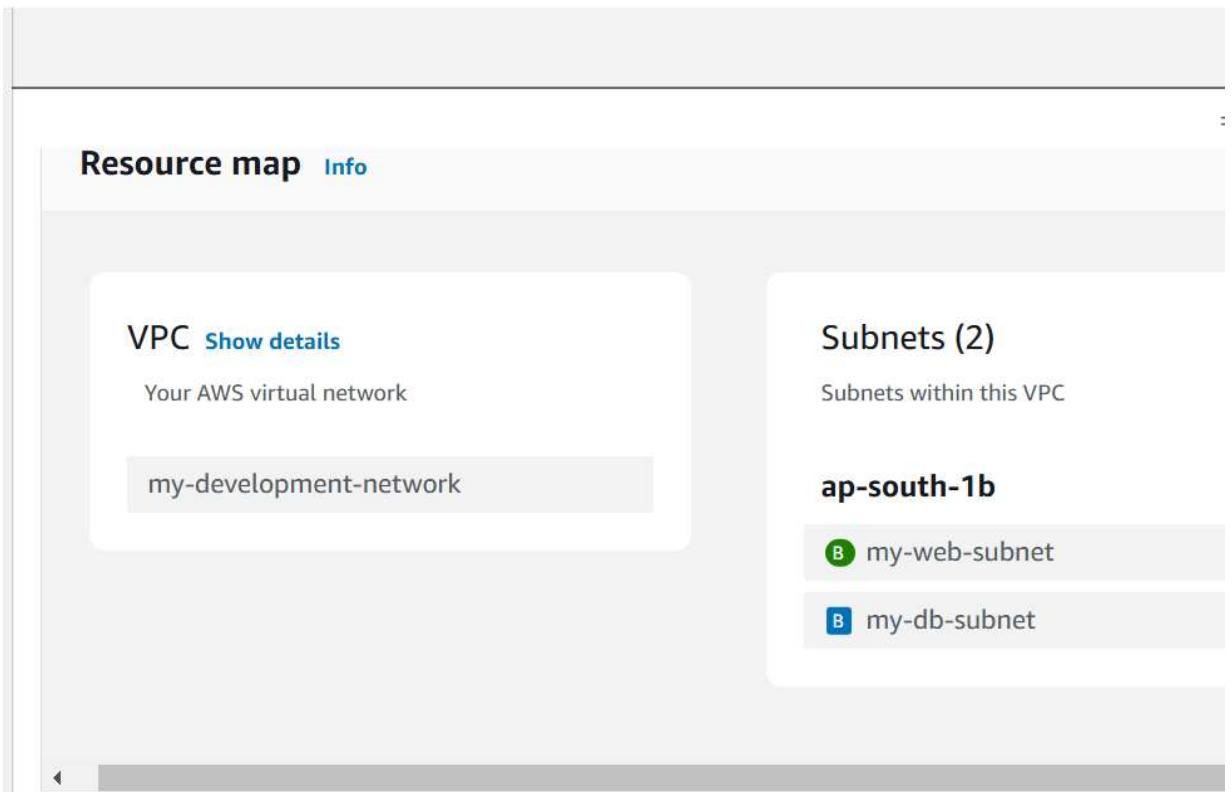
Red arrows highlight the "Route table" tabs in both subnet details pages.

25) We create 1 instance in each subnet.

Note: Security group rule for web-instance allow SSH and all traffic from anywhere. Security group rule for db-instance allow SSH and HTTP from the security group of the web-instance.

Instances (2) Info			
<input type="text"/> Find Instance by attribute or tag (case-sensitive)			
Instance state = running	X	Clear filters	less
<input type="checkbox"/>	Name ✎	Instance ID	Instance state
<input type="checkbox"/>	web-instance	i-0e41b1cde6b0b6f1d	Running + Q
<input type="checkbox"/>	db-instance	i-05b65592332faa1bf	Running + Q

26) So my development VPC is created and we can see the mapping below.



Problem Statement:

Production Network:

1. Design and build a 4-tier architecture.
2. Create 5 subnets out of which 4 should be private named app1, app2, dbcache and db and one should be public, named web.
3. Launch instances in all subnets and name them as per the subnet that they have been launched in.

4. Allow dbcache instance and app1 subnet to send internet requests.
5. Manage security groups and NACLs.

Solution:

- 1) Create a production network VPC .

The steps are same as we did in development network.

The screenshot shows the AWS VPC Details page for a VPC named "my-production-netw". The VPC ID is "vpc-007b203aafb648a16". The VPC is in a state of "Available". The DHCP option set is "dopt-0fd2d5b3b4f436a99". The IPv4 CIDR is "30.0.0.0/16". Network Address Usage metrics are disabled. There are tabs at the bottom for "Resource map", "CIDRs", "Flow logs", "Tags", and "Integrations".

VPC > Your VPCs > vpc-007b203aafb648a16

vpc-007b203aafb648a16 / my-production-netw

Details	
VPC ID	State
vpc-007b203aafb648a16	Available
Tenancy	DHCP option set
Default	dopt-0fd2d5b3b4f436a99
Default VPC	IPv4 CIDR
No	30.0.0.0/16
Network Address Usage metrics	Route 53 Resolver DNS Firewall rule group
Disabled	-

Resource map CIDRs Flow logs Tags Integrations

- 2) Create 5 subnets app1, app2, dbcache ,db and web. Make sure the CIDR do not overlap.

Subnets (5) Info		
<input type="text"/> Find resources by attribute or tag		
	Subnet ID : subnet-0acd29a9d9570fd80 X	Subnet ID : subnet-0ae0b47fa62a64169 X
<input type="checkbox"/>	Name	Subnet ID
<input type="checkbox"/>	db	subnet-0e9630222a6472d89
<input type="checkbox"/>	app2	subnet-0ae0b47fa62a64169
<input type="checkbox"/>	web	subnet-0055856f5421e2122
<input type="checkbox"/>	db-cache	subnet-09a5a3f4010dd7d76
<input type="checkbox"/>	app1	subnet-0acd29a9d9570fd80

- 3) Create a route table and associate with production network. There is no route to internet.

rtb-020ea1f7da9f072ce / my-prod-route table

Details [Info](#)

Route table ID	Main
<input type="checkbox"/> rtb-020ea1f7da9f072ce	<input type="checkbox"/> No
VPC	Owner ID
vpc-007b203aafb648a16 my-production-network	<input type="checkbox"/> 010526284257

Routes [Subnet associations](#) [Edge associations](#) [Route propagation](#) [T](#)

Routes (1)

Destination	Target
30.0.0.0/16	local

- 4) Under subnet association associate app1, app2, db cache and db subnets because they are private subnets.

rtb-020ea1f7da9f072ce / my-prod-route table

Details Routes **Subnet associations** Edge associations Route propagation

Explicit subnet associations (4)

Find subnet association

Name	Subnet ID
db	subnet-0e9630222a6472d89
app2	subnet-0ae0b47fa62a64169
db-cache	subnet-09a5a3f4010dd7d76
app1	subnet-0acd29a9d9570fd80

- 5) Create an internet gateway and attach it to production network.

The screenshot shows the AWS CloudFormation console with a stack named 'internet-gateway'. The stack status is 'igw-0f56c24c079bf928a / internet-gateway'. The 'Details' tab is selected, showing the following information:

Internet gateway ID	State	VPC ID
igw-0f56c24c079bf928a	Attached	vpc-007b203aafb6 network

- 6) Create a route table with destination to internet 0.0.0.0/0 via internet gateway which we just created and attach this route table with web subnet which is public.

Edit routes

Destination	Target
30.0.0.0/16	local
<input type="text" value="0.0.0.0/0"/> X	<input type="text" value="local"/>
<input type="text" value="0.0.0.0/0"/> X	<input type="text" value="Internet Gateway"/>
<input type="text" value="0.0.0.0/0"/> X	<input type="text" value="igw-035315ed5d7c8a49b"/>

[Add route](#)

rtb-0a4ad99fdedb68ffe / my-internet-route

Details	Routes	Subnet associations	Edge associations	Route prop
Explicit subnet associations (1)				
<input type="text" value="Find subnet association"/>				
Name		▼	Subnet ID	
web			subnet-05ee5b4c10127cf05	

- 7) We have to create a NAT gateway which will allow private instance and private subnets to send internet request. Under VPC dashboard click NAT gateway and create NAT gateway.

The screenshot shows the AWS VPC dashboard. On the left, there's a sidebar with various VPC-related options like EC2 Global View, Filter by VPC, Virtual private cloud (Your VPCs, Subnets, Route tables, Internet gateways, Egress-only internet gateways, DHCP option sets, Elastic IPs, Managed prefix lists, Endpoints, Endpoint services), and Security (Network ACLs, Security groups). The 'NAT gateways' option is highlighted with a red oval. The main panel is titled 'NAT gateways' and contains a search bar and a table header with columns for 'Name' and 'NAT gateway ID'. Below the table, a message says 'Select a NAT gateway'.

- 8) Give a name, choose the web Subnet (public)and allocate an elastic IP. Click Create NAT gateway.

Create NAT gateway Info

A highly available, managed Network Address Translation (NAT) service that instances in private subnets can use to access services in other VPCs, on-premises networks, or the internet.

NAT gateway settings

Name - optional

Create a tag with a key of 'Name' and a value that you specify.

The name can be up to 256 characters long.

Subnet

Select a subnet in which to create the NAT gateway.



Connectivity type

Select a connectivity type for the NAT gateway.

- Public
 Private

Elastic IP allocation ID Info

Assign an Elastic IP address to the NAT gateway.

► Additional settings Info

Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search for your resources or track your AWS costs.

Key

Value - optional

You can add 49 more tags.

- 9) Create a new route table which will have internet route access via NAT gateway.

[VPC](#) > [Route tables](#) > rtb-007f00e3b5aa3df45

rtb-007f00e3b5aa3df45 / my-NAT-route-table

Details Info

Route table ID

rtb-007f00e3b5aa3df45

Main

No

VPC

[vpc-0672cdfb04edf082a](#) | my-production-network

Owner ID

010526284257

Routes

Subnet associations

Edge associations

Route propagation

Tags

Routes (2)

Filter routes

Destination

▼

Target

0.0.0.0/0

[nat-02556e46ed557c94d](#)

30.0.0.0/16

local

10) Associate my-NAT-route table with app 1 subnet and db-cache subnet.

VPC > Route tables > rtb-007f00e3b5aa3df45

rtb-007f00e3b5aa3df45 / my-NAT-route-table

Details Info

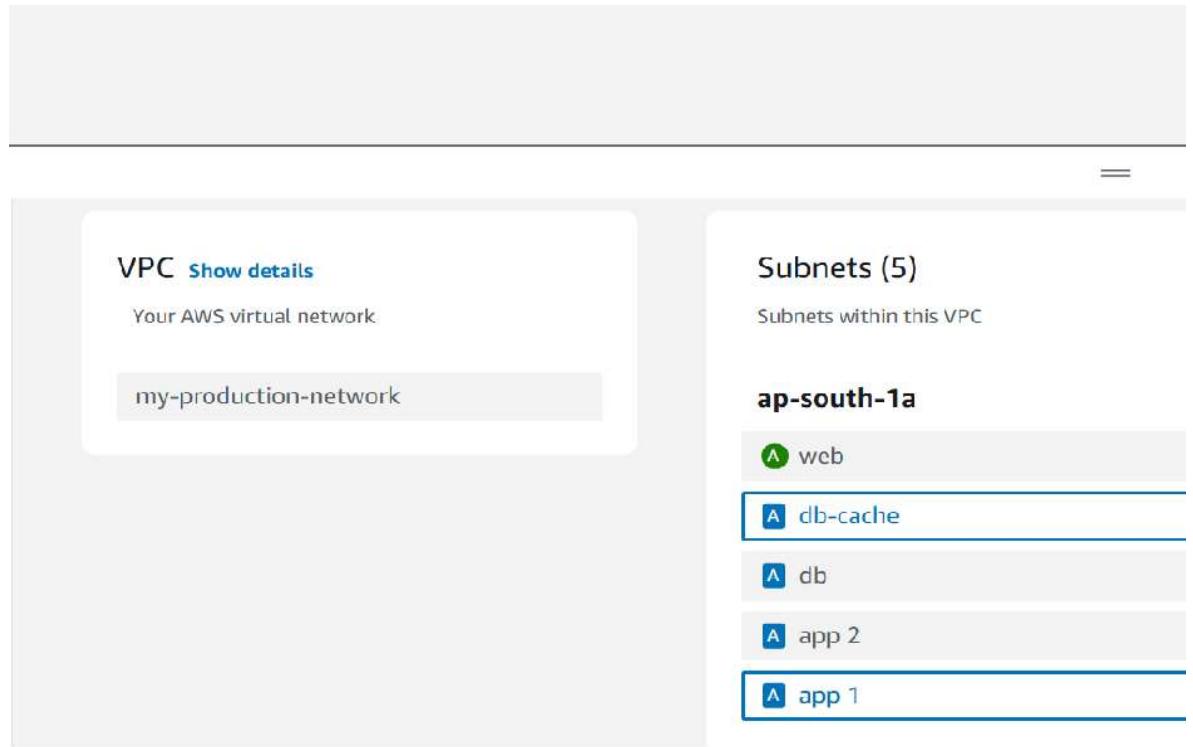
Route table ID	Main	Expl
rtb-007f00e3b5aa3df45	<input type="checkbox"/> No	2 su
VPC	Owner ID	
vpc-0672cdfb04edf082a my-production-network	<input type="checkbox"/> 010526284257	

Routes **Subnet associations** Edge associations Route propagation Tags

Explicit subnet associations (2)

Name	Subnet ID	IPv4
db-cache	subnet-0cbe64ba7c12bbd76	30.0
app 1	subnet-028ebc5f48177508b	30.0

11) My production VPC is created.



12) Now we create instance in public subnet web. In security groups allow incoming rule SSH and all traffic from anywhere.

instance.

Create security group Select existing security group

Security group name - *required*

launch-wizard-2

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and _-!-/()#,@!+=&;!\$*

Description - *required* [Info](#)

launch-wizard-2 created 2024-11-05T16:20:09.179Z

Inbound Security Group Rules

▼ Security group rule 1 (TCP, 22, 0.0.0.0/0) [Remove](#)

Type Info	Protocol Info	Port range Info
ssh	TCP	22
Source type Info	Source Info	Description - <i>optional</i> Info
Anywhere	<input type="text"/> Add CIDR, prefix list or security	e.g. SSH for admin desktop
	<input type="text"/> 0.0.0.0/0 X	

▼ Security group rule 2 (All, All, 0.0.0.0/0) [Remove](#)

Type Info	Protocol Info	Port range Info
All traffic	All	All
Source type Info	Source Info	Description - <i>optional</i> Info
Custom	<input type="text"/> Add CIDR, prefix list or security	e.g. SSH for admin desktop
	<input type="text"/> 0.0.0.0/0 X	

13) We create instances in private subnet with security group incoming rule allowing SSH and HTTP , source:security group of our public instance .

Create security group Select existing security group

Security group name - *required*
launch-wizard-3

This security group will be added to all network interfaces. The name can't be edited after the security group is created. It must be 1-255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and ._-:/()#,@[]+=&;!\$^*

Description - *required* | **Info**
launch-wizard-3 created 2024-11-05T16:24:59.258Z

Inbound Security Group Rules

▼ Security group rule 1 (TCP, 22, sg-072dac00091dd062d)

Type	Protocol	Port range
ssh	TCP	22
Source type	Source	Description - op
Custom	Add CIDR, prefix list or security sg-072dac00091dd062d X	e.g. SSH for ad

▼ Security group rule 2 (TCP, 80, sg-072dac00091dd062d)

Type	Protocol	Port range
HTTP	TCP	80
Source type	Source	Description - op
Custom	Add CIDR, prefix list or security sg-072dac00091dd062d X	e.g. SSH for ad

A red circle highlights the 'ssh' type selection in the first rule. Another red circle highlights the 'Source' dropdown in the second rule's source section. Red arrows point from both circles to the 'X' button in the respective source input fields.

14) Now we will check whether db-cache instance is able to connect to internet via NAT gateway. So first we

connect our web instance which is public. We ping google.com and we can see that it is able to connect via internet gateway.

```
ubuntu@ip-30-0-2-14:~$ ping google.com
PING google.com (142.250.67.142) 56(84) bytes of data.
64 bytes from bom12s06-in-f14.1e100.net (142.250.67.142): icmp_seq=1 t
64 bytes from bom12s06-in-f14.1e100.net (142.250.67.142): icmp_seq=2 t
64 bytes from bom12s06-in-f14.1e100.net (142.250.67.142): icmp_seq=3 t
64 bytes from bom12s06-in-f14.1e100.net (142.250.67.142): icmp_seq=4 t
^C
--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 1.979/2.242/2.823/0.341 ms
ubuntu@ip-30-0-2-14:~$ █
```

15) We will connect to our private db-cache instance using web instance as a jump server. So we locate our *.pem file , copy it , create a nano file and paste the content of *.pem file. Then change access permissions using chmod command.

```
ubuntu@ip-30-0-2-14:~$ sudo su
root@ip-30-0-2-14:/home/ubuntu# nano test.pem
root@ip-30-0-2-14:/home/ubuntu# chmod 400 "test.pem"
root@ip-30-0-2-14:/home/ubuntu# █
```

16) Now we connect using SSH client and check how the ip change. We are now connected to our db-cache instance.

```
The authenticity of host '30.0.4.13 (30.0.4.13)' can't be established.
ED25519 key fingerprint is SHA256:51RYURF78hYZG+MD6JGs3xEiXIV+Wkw3EKoC
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Warning: Permanently added '30.0.4.13' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1016-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

System information as of Tue Nov  5 16:44:36 UTC 2024

System load: 0.0          Processes:           103
Usage of /:   22.9% of 6.71GB  Users logged in:    0
Memory usage: 20%          IPv4 address for enX0: 30.0.4.13
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-30-0-4-13:~$ █
```

17) Now we try to connect to internet by pinging google.com and we are able to access internet via NAT gateway.

```
ubuntu@ip-30-0-4-13:~$ ping google.com
PING google.com (142.250.199.142) 56(84) bytes of data.
64 bytes from bom07s36-in-f14.1e100.net (142.250.199.142): icmp_seq=1
64 bytes from bom07s36-in-f14.1e100.net (142.250.199.142): icmp_seq=2
64 bytes from bom07s36-in-f14.1e100.net (142.250.199.142): icmp_seq=3
64 bytes from bom07s36-in-f14.1e100.net (142.250.199.142): icmp_seq=4
^C
--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 2.156/2.214/2.358/0.083 ms
ubuntu@ip-30-0-4-13:~$ █
```

18) My production network working absolutely fine.

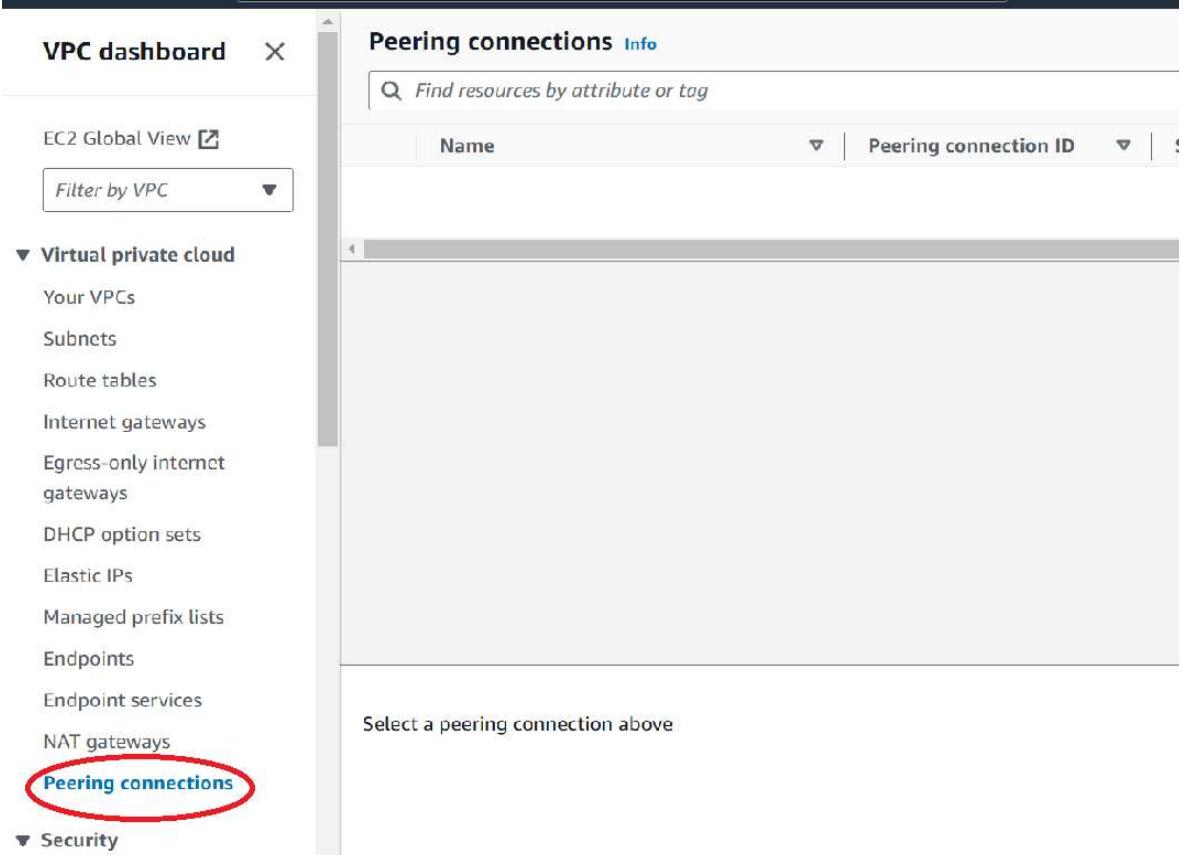
Task to be performed:

Create peering connection between production network and development network.

Setup connection between db subnets of both production network and development network respectively

Solution:

- 1) VPC peering : we can initiate the peering from any network. So lets choose production network. So under VPC dashboard click peering connections and create peering connection.



- 2) Give a name, select the VPC requester which is my production network , under region choose where the development

network is . my development network is in Singapore and in VPC acceptor give the VPC ID of the development network and create peering.

Create peering connection

A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between privately.

Info

Peering connection settings

Name - *optional*

Create a tag with a key of 'Name' and a value that you specify.

my-peering

Select a local VPC to peer with

VPC ID (Requester)

vpc-0672cdfb04edf082a (my-production-network)

VPC CIDRs for vpc-0672cdfb04edf082a (my-production-network)

CIDR	Status	Status reason
30.0.0.0/16	Associated	-

Select another VPC to peer with

Account

My account

Another account

Region

This Region (ap-south-1)

Another Region

Asia Pacific (Singapore) (ap-southeast-1)

VPC ID (Acceptor)

vpc-05fd3c0cb12f3aad0

- 3) So a request is sent to the acceptor region i.e Singapore region.

✓ A VPC peering connection `pcx-0a7858909c6f4adeb` / `my-peering` has been requested.
Remember to change your region to `ap-southeast-1` to accept the peering connection.

VPC > Peering connections > `pcx-0a7858909c6f4adeb`

`pcx-0a7858909c6f4adeb` / `my-peering`

Details <small>Info</small>	
Requester owner ID <input type="button" value="Copy"/>	Acceptor owner ID <input type="button" value="Copy"/>
<code>010526284257</code>	<code>010526284257</code>
Peering connection ID <input type="button" value="Copy"/>	Requester VPC <code>vpc-0672cdfb04edf082a</code> / <code>my-proc</code>
<code>pcx-0a7858909c6f4adeb</code>	
Status <small>Initiating Request to 010526284257</small>	Requester CIDRs <input type="button" value="Copy"/>
Expiration time <small>Tuesday, November 12, 2024 at 22:46:26 GMT+5:30</small>	Requester Region <small>Mumbai (ap-south-1)</small>

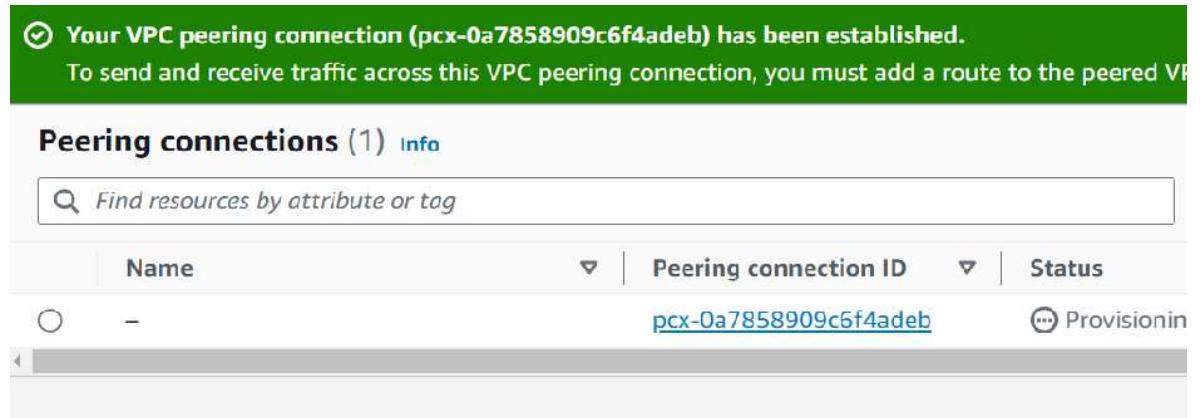
- 4) When we go to Singapore region and click peering connection we see the pending acceptance notification.

The screenshot shows the AWS VPC dashboard. On the left, there's a sidebar with options like EC2 Global View, Filter by VPC, and a list of VPC-related resources: Your VPCs, Subnets, Route tables, Internet gateways, Egress-only internet gateways, DHCP option sets, Elastic IPs, Managed prefix lists, Endpoints, Endpoint services, NAT gateways, and Peering connections. The 'Peering connections' link is highlighted with a red oval. The main panel is titled 'Peering connections (1)' and shows one entry: Name - (empty), Peering connection ID - [pcx-0a7858909c6f4adeb](#), and Status - Pending.

5) Go to actions and accept the request.

This screenshot is similar to the previous one, but the status of the peering connection has changed. The 'Status' column now shows 'Accepted'. The rest of the interface remains the same, with the peering connection ID still being [pcx-0a7858909c6f4adeb](#).

6) Finally VPC peering done.

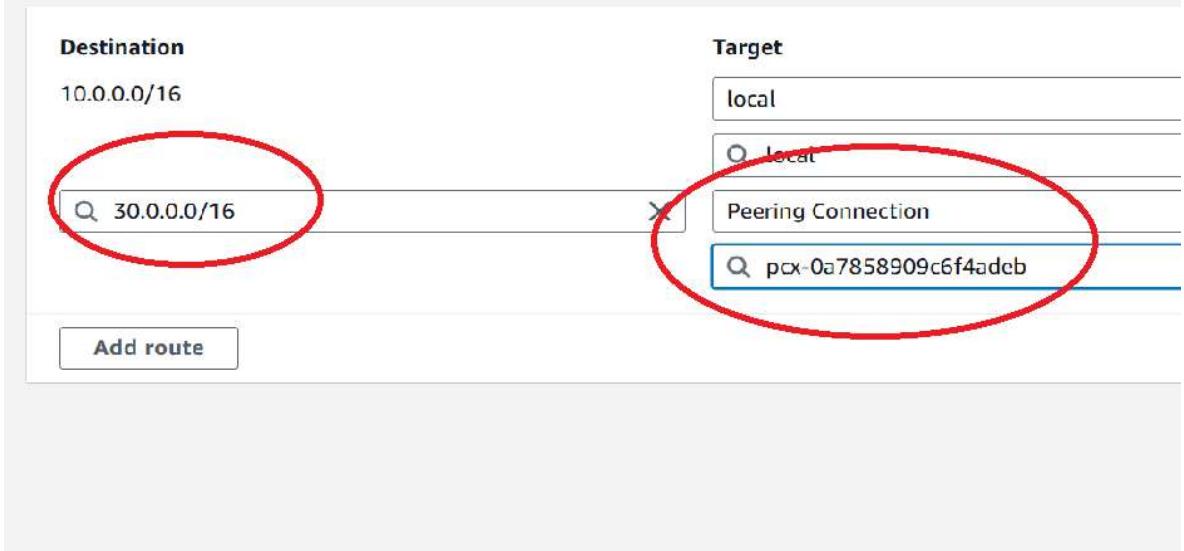


- 7) To Setup connection between db subnets of both production network and development network we need to modify route table first. So in development network , go to route table and add a route where destination will be the CIDR of production VPC via peering connection.

Edit routes

Destination	Target
10.0.0.0/16	local
30.0.0.0/16	Local Peering Connection pcx-0a7858909c6f4adeb

Add route

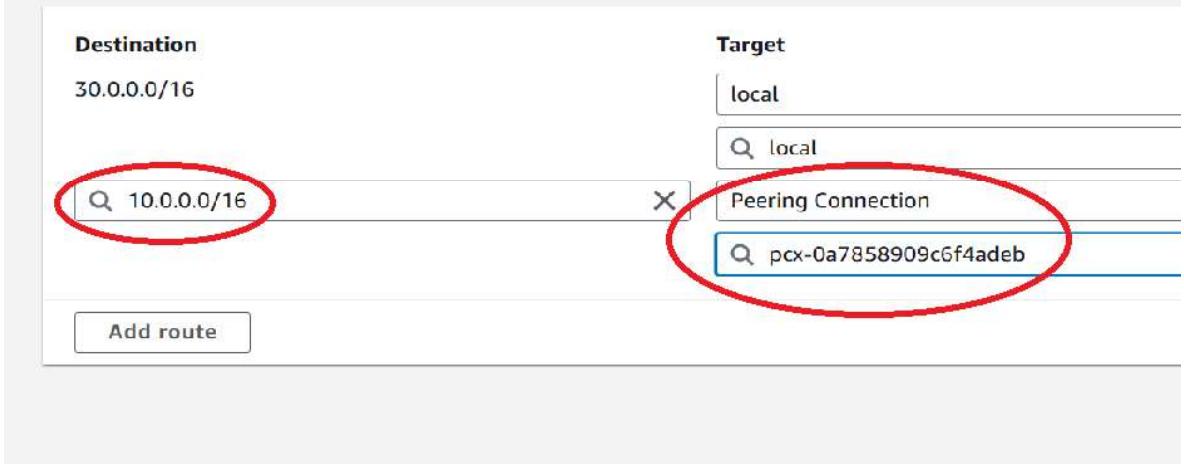


- 8) In the production network, go to route table and add a route where destination will be the CIDR of development VPC via peering connection.

Edit routes

Destination	Target
30.0.0.0/16	local
10.0.0.0/16	local Peering Connection pcx-0a7858909c6f4adeb

Add route



- 9) Now we modify the security group rules of the instances in db subnets of both the development and production VPC. In the inbound rules of db instance(prod VPC) allow all traffic from source: give the CIDR range of the development VPC and vice versa.

The screenshot shows the 'Inbound rules' section of the AWS Security Groups interface. It lists three rules:

Security group rule ID	Type	Protocol	Port Range
sgr-06ef0b2fe8fdc9441	All traffic	All	
sgr-09e588f9bf6342686	HTTP	TCP	80
sgr-002d6ad0de788edd5	SSH	TCP	22

An 'Add rule' button is located at the bottom left of the table area. A red oval highlights the 'Type' column for the first rule, which is set to 'All traffic'.

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID

sgr-0c74be31a098ae468

Type Info

HTTP

Protocol In

TCP

sgr-06cd3d57dbca7e124

All traffic

All

sgr-01acd864041dfe20c

SSH

TCP

Add rule

- 10) Now when we ping the db instance in production VPC from db instance in development VPC we can communicate easily. Ping (the private IP of the instance).

```
ubuntu@ip-10-0-1-13:~$ ping 30.0.1.12
PING 30.0.1.12 (30.0.1.12) 56(84) bytes of data.
64 bytes from 30.0.1.12: icmp_seq=1 ttl=64 time=61.6 ms
64 bytes from 30.0.1.12: icmp_seq=2 ttl=64 time=61.7 ms
64 bytes from 30.0.1.12: icmp_seq=3 ttl=64 time=61.5 ms
64 bytes from 30.0.1.12: icmp_seq=4 ttl=64 time=61.6 ms
64 bytes from 30.0.1.12: icmp_seq=5 ttl=64 time=61.4 ms
^C
--- 30.0.1.12 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 61.443/61.565/61.662/0.078 ms
ubuntu@ip-10-0-1-13:~$
```

11) So peering connection done successfully.