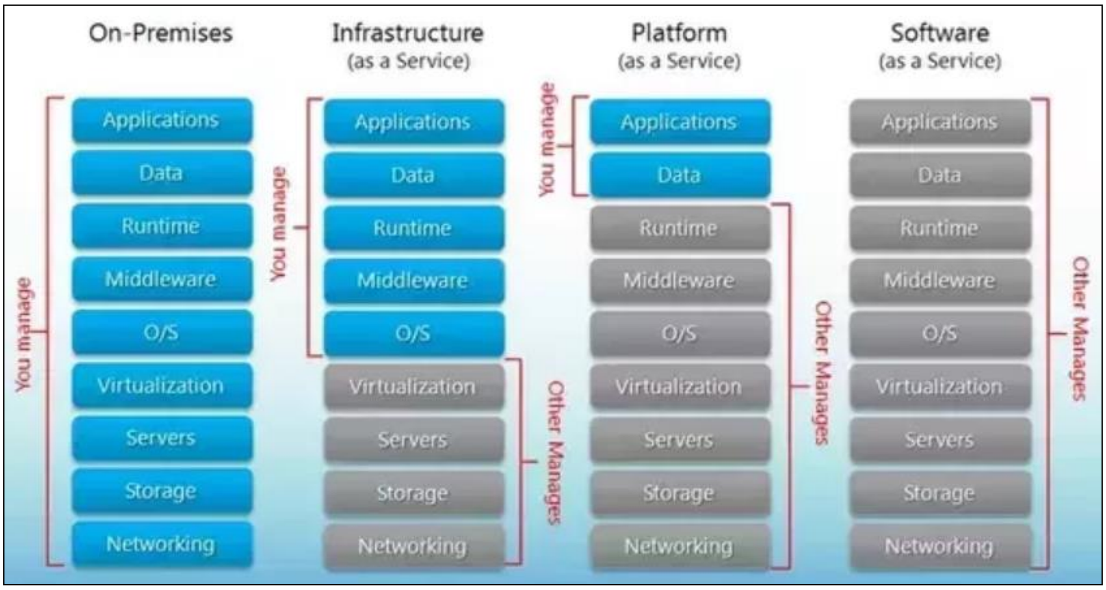
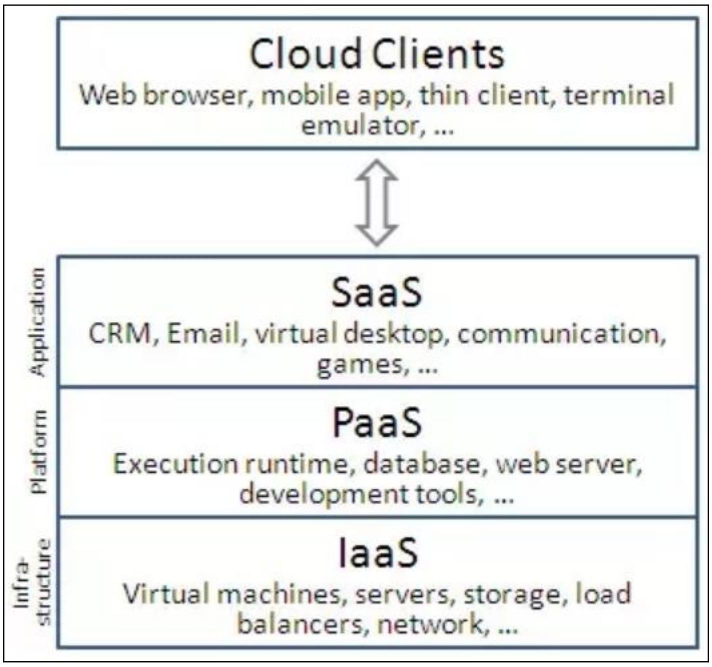
Table of Contents

1. Cloud Introduction



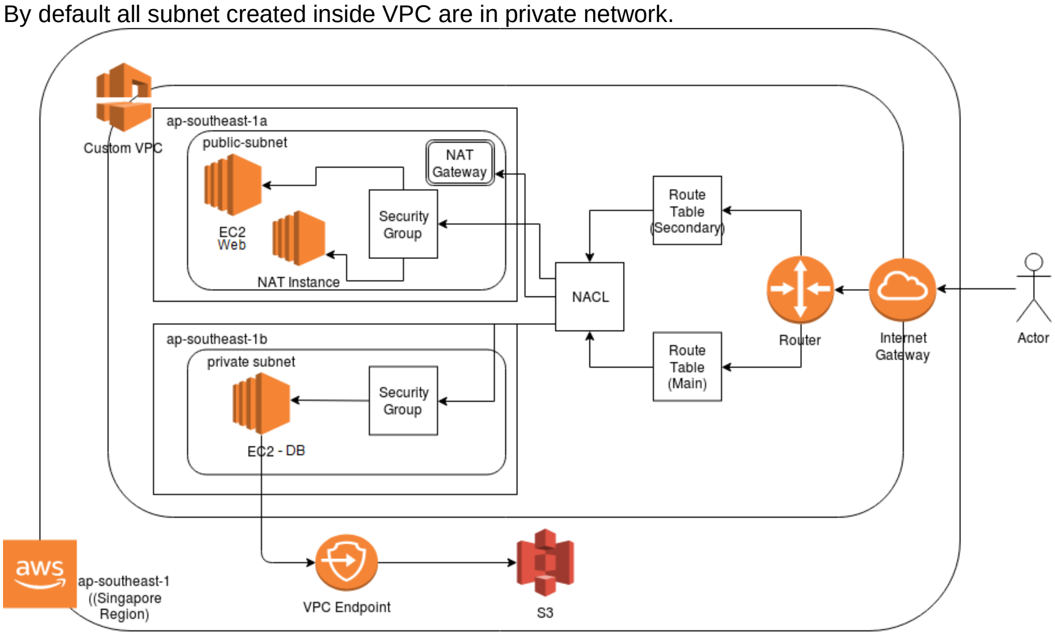
1.1. Type of Cloud

- Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud.
- Amazon Web Services (AWS) EC2 is IaaS.
- You can get virtual machines of any size and configuration and run variants of Linux or Windows on it.


Cloud Clients
Web browser, mobile app, thin client, terminal emulator, ...
SaaS
CRM, Email, virtual desktop, communication, games, ...
PaaS
Execution runtime, database, web server, development tools, ...
IaaS
Virtual machines, servers, storage, load balancers, network, ...
Application
Platform
Infra-structure

2. VPC

- Amazon Virtual Private Cloud (Amazon VPC) enables you to launch AWS resources into a virtual network.
- Amazon VPC is the networking layer for Amazon EC2.
- There's no additional charge for using a VPC.
- There are charges for some VPC components, such as NAT gateways, Reachability Analyzer, and traffic mirroring.
- It is logically isolated from other virtual networks in the AWS Cloud.
- By default all subnet created inside VPC are in private network.



2.1. Concepts for VPCs:

- **Virtual private cloud (VPC)** — A virtual network dedicated to your AWS account.
- **Subnet** — A range of IP addresses in your VPC. Or Partition creation inside VPC called Subnet. It is a subdivision of a VPC. Breaking the network down into smaller networks (subnets) is called subnetting.
- **Route table** — A set of rules, called routes, that are used to determine where network traffic is directed.
- **Internet gateway** — A gateway that you attach to your VPC to enable communication between

  resources in your VPC and the internet. We use an internet gateway to make a subnet public.
- **VPC endpoint** — Enables you to privately connect your VPC to supported AWS services.
- **NAT Gateway** - When you want only a certain set of resources to be allowed publicly on the internet, you can use a NAT gateway. NAT is short for Network Address Translation, which means that it translates private IP addresses to public IPs.

2.2. Creating a VPC

- Login and access to AWS services → Network and Content Delivery → VPC
- Choose the option → Creating the VPC – on the right of the navigation bar.
- Click Start VPC. Now click VPC With a Single Public Subnet option on the left.
- fill in the required details – such as VPC name and subnet name and Leave the other boxes as **default** and click **Create VPC**.
  - IP CIDR block: **10.0.0.0/16**
  - VPC Name: **MyVPC**
  - Public subnet: 10.0.0.0/24
  - Availability Zone: No Preference – Default
  - Subnet name: public subnet – Default
  - Enable DNS hostnames: Yes
  - Hardware tenancy: Default

2.3. Create Subnet – WebSN

- Open the Amazon VPC console:
- Choose VPC Dashboard, then click on Subnets and finally click on Create Subnet.
- Enter the following values in the dialog box:
  - Name tag: **WebSN**
  - VPC: Select the VPC that you created above. – **MyVPC**
  - Availability Zone: Choose the Availability Zone.
  - IPv4 CIDR block: **10.0.1.0/24**
- Click on Create and Close on the confirmation page

2.4. Create Subnet – DB-SN

- Open the Amazon VPC console:
- Choose VPC Dashboard, then click on Subnets and finally click on Create Subnet.
- Enter the following values in the dialog box:
  - Name tag: **DB-SN**
  - VPC: Select the VPC that you created above. – **MyVPC**
  - Availability Zone: Choose the Availability Zone.
  - IPv4 CIDR block: **10.0.2.0/24**
- Click on Create and Close on the confirmation page

2.5. Convert WebSN Private Subnet into Public– Internet Gateway

- To Make Subnet as Public 2 Steps need to follow

1. **Assign the Public IP to Subnet**
   - Choose VPC Dashboard, then click on Subnets → Select the Subnet.
   - Click on **Action** → Modify Auto Assign IP Setting → Enable Auto Assign Public (IPv4 Address) → Save

2. **Create Internet Gateway and attached to VPC**
   - Navigate to the AWS console → Services.
   - Under the Networking & Content Delivery section, choose VPC.
   - Navigate to Virtual Private Cloud -> Internet Gateways.
   - Click **Create Internet Gateway**.
     - Name tag: **my-vpc-gateway**
   - Click **Attach to VPC**.
   - Select your VPC from the Name tag drop-down list and click **Yes, Attach**

3. **Attach the internet Gateway to VPC if not done**
   - Select the **my-vpc-gateway** → Click Action → Attach to VPC → Select VPC(MyVPC) → Attach

4. **Create the Route Table**
   - Navigate to the AWS console → Services.
   - Under the Networking & Content Delivery section, choose VPC.
   - Navigate to Virtual Private Cloud → Click Route Table.
   - Click Route Table
     - Name Tag: **InternetRT**
     - VPC**: MyVPC**
   - Click Create → Close

5. **Connect – First end Route Table to Subnet**
   - Select the Route Table (InternetRT)→ Subnet Associate tab → Edit Subnet Associate → Select the Subnet (10.0.1.0/24) - **WebSN** → Save

6. **Connect – Second end Route Table to Internet Gateway**
   - Select the Route Table (InternetRT) → Route tab → Edit Route → Add Route
     - Target Gateway: **my-vpc-gateway**
     - Destination: 0.0.0.0/0
   - Save Route

2.6. Provide Internet Access to DB-SN Subnet– through NAT

1. **Create the NAT Gateway**
   - Open the Amazon VPC console → choose **NAT Gateways**.
   - Choose **Create NAT Gateway**
     - Name: This is tag of NAT - **Optional**
     - Subnet: **10.0.1.0/24**
     - Create new **Elastic IP**
     - Choose **Add new tag** and enter the key name and value. – **Optional**
     - Choose **Create a NAT Gateway**.

2. **Create the Route Table**
    - Navigate to the AWS console → Services.
    - Under the Networking & Content Delivery section, choose VPC.
    - Navigate to Virtual Private Cloud → Click Route Table.
    - Click Route Table
        - Name Tag: **NAT-RT**
        - VPC**: MyVPC**
    - Click Create → Close

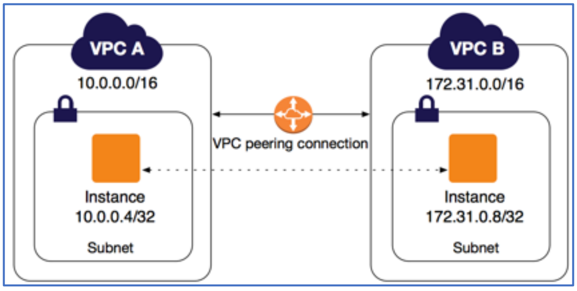
3. **Associate NAT to Subnet**

    Select NAT-RT → Subnet Associate → Edit Subnet Associate → Select Private Subnet → Save

4. **Add Route to NAT**
    - Select NAT-RT → Route → Edit Route → Add Route →
        - Target: NAT Gateway
        - Destination: 0.0.0.0/0
    - Save Route

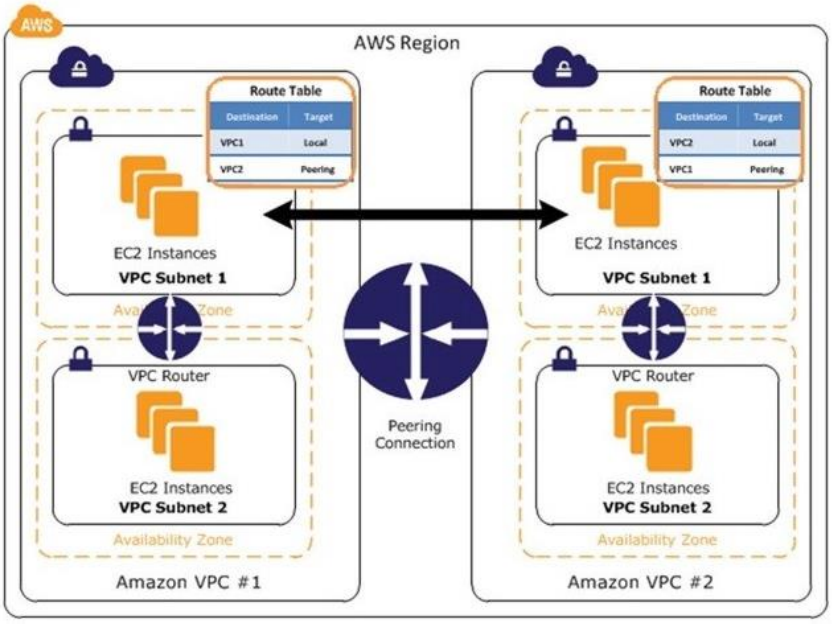
3. VPC Peering

- A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them using private IPv4 addresses or IPv6 addresses.
- A VPC peering connection **helps you to facilitate the transfer of data**.
- For example, if you have more than one AWS account, you can peer the VPCs across those accounts to create a file sharing network.
- You can also use a VPC peering connection to allow other VPCs to access resources you have in one of your VPCs.



3.1. VPC peering connection Procedure

1. The owner of the `requester VPC` sends a request to the owner of the `accepter VPC` to create the VPC peering connection. The accepter VPC can be owned by you, or another AWS account, and cannot have a CIDR block that overlaps with the requester VPC's CIDR block.
2. The owner of the accepter VPC accepts the VPC peering connection request to activate the VPC peering connection.
3. To enable the flow of traffic between the VPCs using private IP addresses, the owner of each VPC in the VPC peering connection must manually add a route to one or more of their VPC route tables that points to the IP address range of the other VPC.
4. update the security group rules that are associated with your instance to ensure that traffic to and from the peer VPC is not restricted.
5. By default, if instances on either side of a VPC peering connection address each other using a public DNS hostname, the hostname resolves to the instance's public IP address. To change this behaviour, enable DNS hostname resolution for your VPC connection.



3.2. To create a VPC peering connection with a VPC in the same Region

1. Open the Amazon VPC console at **https://console.aws.amazon.com/vpc/.**
2. In the navigation pane, choose **Peering Connections**, **Create Peering Connection**.
3. Configure the following information and choose **Create Peering Connection.**

- **Peering connection name tag**: You can optionally name your VPC peering connection.
- **VPC (Requester)**: Select the VPC in your account with which you want to create the VPC peering connection.
- Under **Select another VPC to peer with**: Ensure **My account** is selected and select another of your VPCs.
- (Optionally add or remove a tag.
  [Add a tag] Choose **Add tag** and do the following:
    - For **Key**, enter the key name.
    - For **Value**, enter the key value.

  [Remove a tag] Choose the Delete button ("X") to the right of the tag’s Key and Value.

4. In the confirmation dialog box, choose **OK**.
5. Select the VPC peering connection that you've created, and choose **Actions**, **Accept Request**.
6. In the confirmation dialog, choose **Yes, Accept**. A second confirmation dialog displays; choose **Modify my route tables now** to go directly to the route tables page or choose **Close**.

3.3. To create a VPC peering connection with a VPC in a different Region

1. Open the Amazon VPC console at **https://console.aws.amazon.com/vpc/.**
2. In the navigation pane, choose **Peering Connections**, **Create Peering Connection**.
3. Configure the following information and choose **Create Peering Connection.**
   - **Peering connection name tag**: You can optionally name your VPC peering connection.
   - **VPC (Requester)**: Select the requester VPC in your account with which to request the VPC peering connection.
   - **Account**: Ensure **My account** is selected.
   - **Region**: Choose **Another region**, select the Region in which the accepter VPC resides.
   - **VPC (Accepter)**: Enter the ID of the accepter VPC.
4. In the confirmation dialog box, choose **OK**.
5. In the Region selector, select the Region of the accepter VPC.
6. In the navigation pane, choose **Peering Connections**. Select the VPC peering connection that you've created, and choose **Actions**, **Accept Request**.
7. In the confirmation dialog, choose **Yes, Accept**. A second confirmation dialog displays; choose **Modify my route tables now** to go directly to the route tables page, or choose **Close**

3.4. Creating a VPC peering connection with a VPC in another AWS account

- Before you begin, ensure that you have the **AWS account number** and **VPC ID** of the VPC to peer with.
- After you've created the request, the owner of the accepter VPC must accept the VPC peering connection to activate it.

3.4.1. To request a VPC peering with a VPC in another account in the same Region

1. Open the Amazon VPC console at **https://console.aws.amazon.com/vpc/.**
2. In the navigation pane, choose **Peering Connections**, **Create Peering Connection**.
3. Configure the following information and choose **Create Peering Connection.**
   - **Peering connection name tag**: You can optionally name your VPC peering connection.
   - **VPC (Requester)**: Select the VPC in your account with which to create the VPC peering connection.
   - **Account**: Choose **Another account**.
   - **Account ID**: Enter the AWS account ID of the owner of the accepter VPC.
   - **VPC (Accepter)**: Enter the ID of the VPC with which to create the VPC peering connection.
4. In the confirmation dialog box, choose **OK**

3.4.2. To request a VPC peering with a VPC in another account in a different Region

1. Open the Amazon VPC console at **https://console.aws.amazon.com/vpc/.**
2. In the navigation pane, choose **Peering Connections**, **Create Peering Connection**.
3. Configure the following information and choose **Create Peering Connection.**
   - **Peering connection name tag**: You can optionally name your VPC peering connection.
   - **VPC (Requester)**: Select the VPC in your account with which to create the VPC peering connection.
   - **Account**: Choose **Another account**.
   - **Account ID**: Enter the AWS account ID of the owner of the accepter VPC.
   - **Region**: Choose **Another region**, select the Region in which the accepter VPC resides.
   - **VPC (Accepter)**: Enter the ID of the VPC with which to create the VPC peering connection.
4. In the confirmation dialog box, choose **OK**.

Note: The VPC peering connection that you've created is not active. To activate it, the owner of the accepter VPC must accept the VPC peering connection request. To enable traffic to be directed to the peer VPC, update your VPC route table.

3.4.3. To add an IPv4 route for a VPC peering connection

- To send private IPv4 traffic from your instance to an instance in a peer VPC, you must add a route to the route table that's associated with your subnet in which your instance resides.

To add an IPv4 route for a VPC peering connection

1. Open the Amazon VPC console at **https://console.aws.amazon.com/vpc/.**
2. In the navigation pane, choose **Route Tables**.
3. Select the check box next to the route table that's associated with the subnet in which your instance resides.
4. Choose **Actions**, **Edit routes**.
5. Choose **Add route**.
6. For **Destination**, enter the IPv4 address range -- For example, if the CIDR block of the peer VPC is 10.0.0.0/16, you can specify a portion 10.0.0.0/24
7. For **Target**, select the VPC peering connection, and then choose **Save changes**

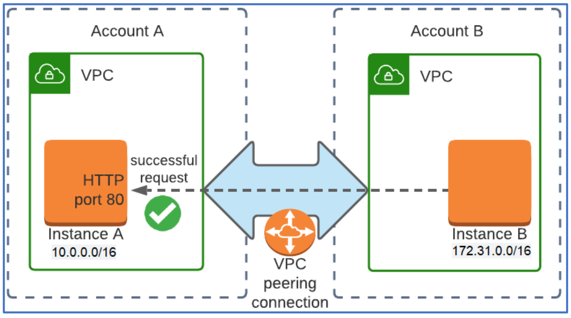
Note: The owner of the peer VPC must also complete these steps to add a route to direct traffic back to your VPC through the VPC peering connection.

3.4.4. Accepting a VPC peering connection

1. Open the Amazon VPC console at **https://console.aws.amazon.com/vpc/.**
2. Use the Region selector to choose the Region of the accepter VPC.
3. In the navigation pane, choose **Peering Connections**.
4. Select the pending VPC peering connection (the status is pending-acceptance), and choose **Actions**, **Accept Request**.
5. In the confirmation dialog box, choose **Yes, Accept**. A second confirmation dialog displays; choose **Modify my route tables now** to go directly to the route tables page, or choose **Close**

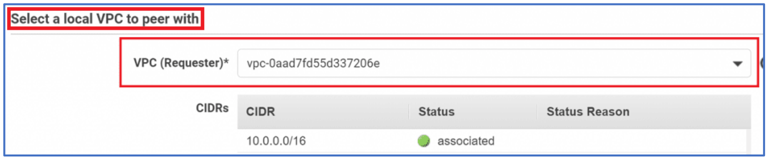
3.5. Cross Account Peering – with VPC - Example

Requirements

- we've got 2 accounts **Account A** (provider account) and **Account B** (consumer account)
- the 2 accounts have VPCs with different CIDR blocks
  - account A VPC CIDR = 10.0.0.0/16
  - account B VPC CIDR = 172.31.0.0/16
- account A is running an EC2 instance called Instance A, which exposes some data over HTTP port 80
- account B is running an EC2 instance called Instance B, which needs to access the data from instance A in account A
- the data must remain with the AWS network and not go onto the public internet
- 

3.5.1. Step 1: create the VPC peering connection

1. Open the Amazon VPC console at **https://console.aws.amazon.com/vpc/** from **Account A**
2. In the navigation pane, choose **Peering Connections** → **Create Peering Connection**.
   - for **VPC (Requester)** select the VPC you want to connect



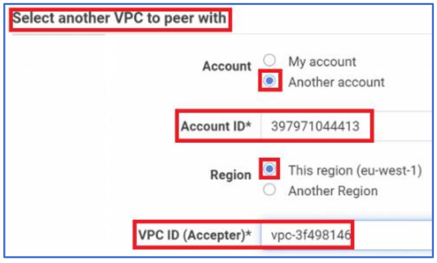
Select a local VPC to peer with

VPC (Requester)* vpc-0aad7fd55d337206e

CIDRs

CIDR	Status	Status Reason
10.0.0.0/16	● associated	

   - under **Select another VPC to peer with** → Select *Another account* and enter the account B account id
   - for **VPC ID (Accepter)** enter the VPC id of the VPC in account B



Select another VPC to peer with

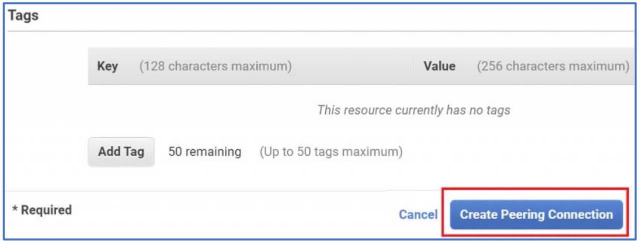
Account ○ My account
● Another account

Account ID* 397971044413

Region ● This region (eu-west-1)
○ Another Region

VPC ID (Accepter)* vpc-3f498146

- click **Create Peering Connection**.



Tags

Key	(128 characters maximum)	Value	(256 characters maximum)

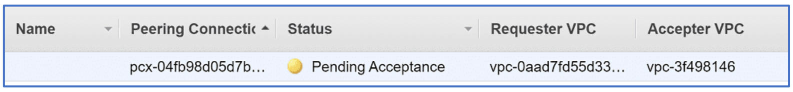
This resource currently has no tags

Add Tag 50 remaining (Up to 50 tags maximum)

* Required

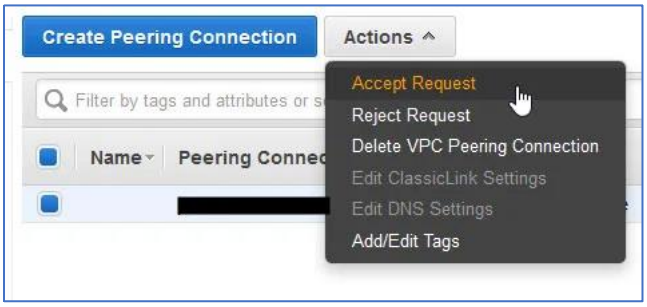
3.5.2. Step 2: accept the VPC peering connection

- The peering connection must now be accepted in account B.
- In the VPC dashboard in account B, under **Peering Connections** the new connection should be shown in a *Pending Acceptance* status. It may take a few minutes for the connection to appear.

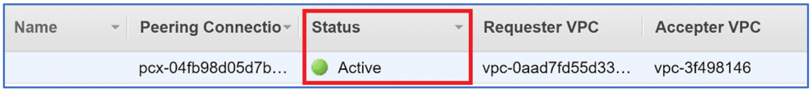


Name	Peering Connectio	Status	Requester VPC	Accepter VPC
	pcx-04fb98d05d7b…	● Pending Acceptance	vpc-0aad7fd55d33…	vpc-3f498146

- Select the connection then under **Actions** select ***Accept Request***, then **Yes, Accept**


Create Peering Connection
Actions
Filter by tags and attributes or s
Name
Peering Connec
[redacted]
Accept Request
Reject Request
Delete VPC Peering Connection
Edit ClassicLink Settings
Edit DNS Settings
Add/Edit Tags

- Once you've done this, the peering connection should go into the *Active* state in both accounts.

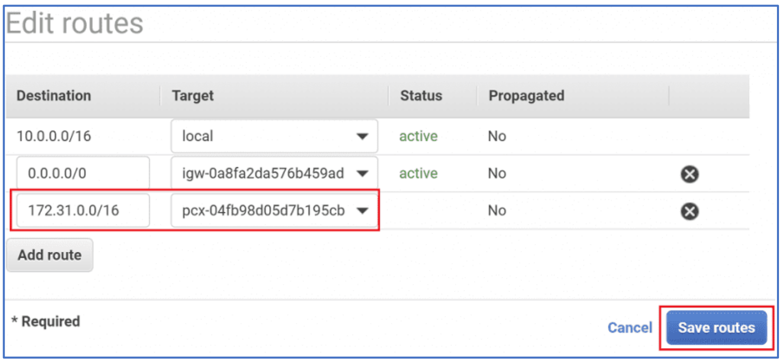


Name	Peering Connectio	Status	Requester VPC	Accepter VPC
	pcx-04fb98d05d7b…	● Active	vpc-0aad7fd55d33…	vpc-3f498146

3.5.3. Step 3: setup route tables to route traffic to VPC peering connection

In **account A,**

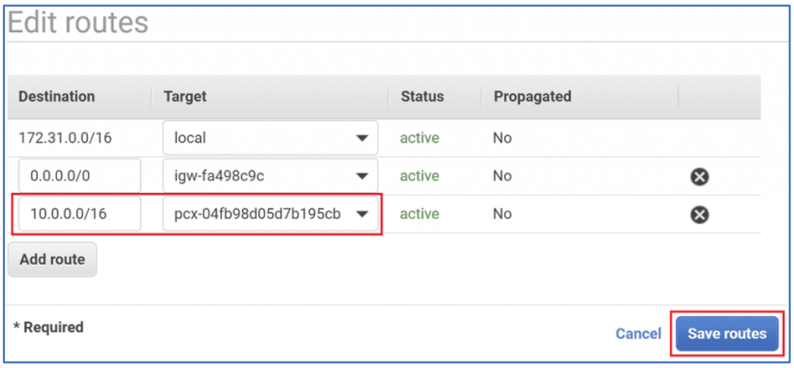
- Open the Amazon VPC console at **https://console.aws.amazon.com/vpc/.**
- In the navigation pane, choose **Route Tables**.
- then select the route table associated with the subnet into which your EC2 instance is deployed.
- Select the **Routes** tab to show the actual routes.
- Choose **Actions**, **Edit routes**.
- Choose **Add route**.
- **Destination** enter the CIDR(172.31.0.0/16) for the VPC is **account B**
- For **Target**, select the **VPC peering connection**, and then choose **Save changes**



Destination	Target	Status	Propagated	
10.0.0.0/16	local	active	No	
0.0.0.0/0	igw-0a8fa2da576b459ad	active	No	⊗
172.31.0.0/16	pcx-04fb98d05d7b195cb		No	⊗

in **account B**

- In the VPC dashboard for account B go to **Route Tables**,
- select the route table for the subnet where your EC2 instance is deployed,
- select the **Routes** tab, **Edit routes**, then **Add route**.
- For the destination enter the CIDR(10.0.0.0/16) of the VPC in account A,
- then for the **target** select the **peering connection**. Finally, select **Save Routes**.



Edit routes

Destination	Target	Status	Propagated	
172.31.0.0/16	local	active	No	
0.0.0.0/0	igw-fa498c9c	active	No	⊗
10.0.0.0/16	pcx-04fb98d05d7b195cb	active	No	⊗

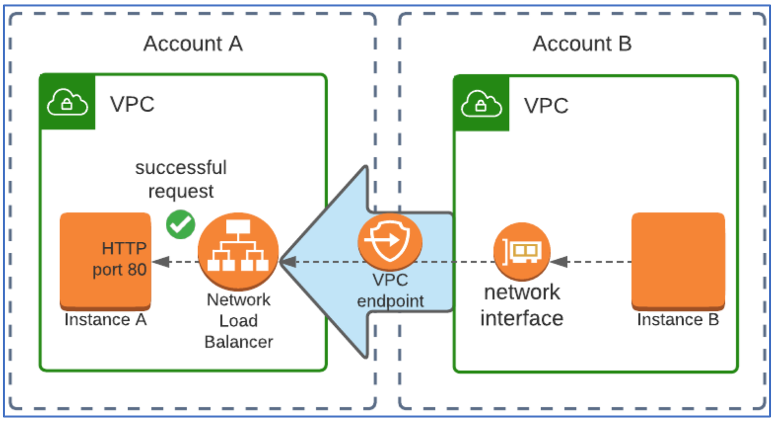
3.5.4. Step 4: test the VPC peering connection

- In account A go to the EC2 dashboard, select the instance you want to connect to, then copy the **Private IPv4 address** which we'll need to establish the connection.
- In account B, start a session in the EC2 instance you want to connect from. Make a curl request to the private IP address curl <private-ip-address>.

```
# ping 10.0.0.39
```

3.6. Cross Account Peering – with endpoint service (PrivateLink)

- The VPC endpoint is exposed as a private IP address within your VPC, accessible using a private DNS name.
- VPC endpoints are mostly used to make AWS API requests from a VPC, without going onto the public internet.
- The same technology, called **PrivateLink**, can be used to create a VPC endpoint allowing a connection from one VPC to a *network load balancer* (NLB) in another VPC.

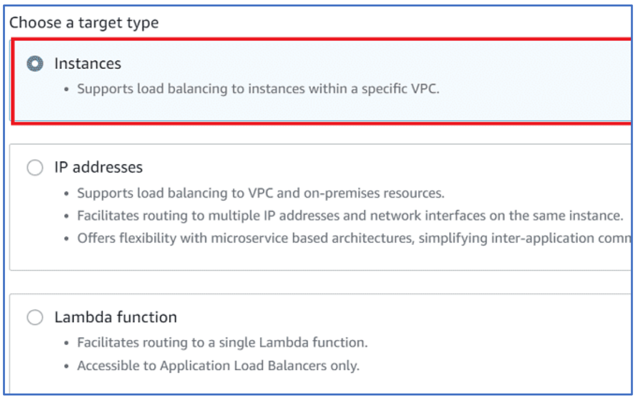


-
- the VPC endpoint is one-directional, meaning you can only send a request from account B to account A
- it's exposed in the VPC of account B as an elastic network interface with a DNS name associated with it.
- that request to the private IP will then be sent through to a network load balancer in another VPC.

3.6.1. Step 1: create a network load balancer

- From the EC2 dashboard in account A go to **Load Balancers**, select **Create Load Balancer**, then select **Create** next to **Network Load Balancer**.
    - give the load balancer a sensible name (e.g. simple-load-balancer)
    - for the **Scheme** select Internal
    - under **VPC** choose the VPC where the instance you want to expose is deployed
    - under **Mappings** pick the availability zones and subnets you want the load balancer to be connected.
    - leave **IPv4 address** and **Private IPv4 address** as the defaults

- Select **Create target group** and a separate **Specify group details** page will open. Enter these details.
  - for **target type** pick Instances



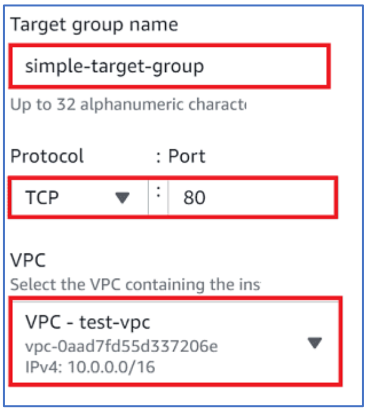
Choose a target type

● Instances
- Supports load balancing to instances within a specific VPC.

○ IP addresses
- Supports load balancing to VPC and on-premises resources.
- Facilitates routing to multiple IP addresses and network interfaces on the same instance.
- Offers flexibility with microservice based architectures, simplifying inter-application comn

○ Lambda function
- Facilitates routing to a single Lambda function.
- Accessible to Application Load Balancers only.

  - for **Target group name** enter a sensible name (e.g. simple-target-group)
  - for **Protocol** you must leave it as TCP in order to connect the target group with an NBL
  - enter the correct **Port**
  - under **VPC** select the VPC where your instance is deployed



Target group name: simple-target-group

Up to 32 alphanumeric characti

Protocol : Port

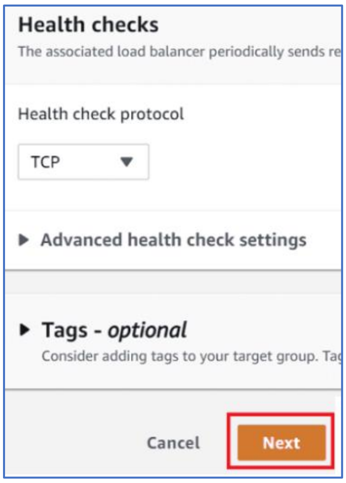
TCP : 80

VPC

Select the VPC containing the ins

VPC - test-vpc
vpc-0aad7fd55d337206e
IPv4: 10.0.0.0/16

- Select **Next**.



Health checks

The associated load balancer periodically sends re

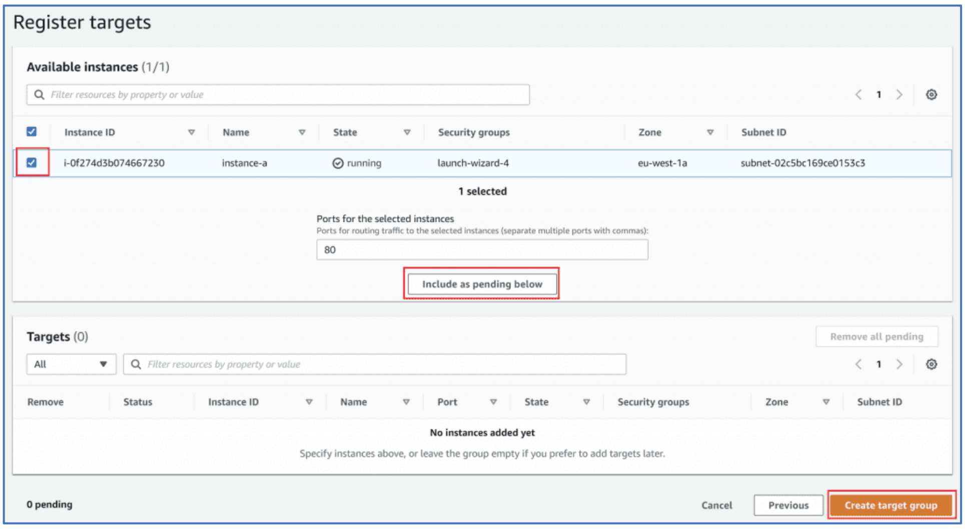
Health check protocol: TCP

▶ Advanced health check settings

▶ Tags - *optional*

Consider adding tags to your target group. Tag

- **Register Targets** page → Select the instance you want to make available, select **Include as pending below**, → select **Create target group**.



Register targets

Available instances (1/1)

Filter resources by property or value

☑	Instance ID	Name	State	Security groups	Zone	Subnet ID
☑	i-0f274d3b074667230	instance-a	✓ running	launch-wizard-4	eu-west-1a	subnet-02c5bc169ce0153c3

1 selected

Ports for the selected instances
Ports for routing traffic to the selected instances (separate multiple ports with commas):
80

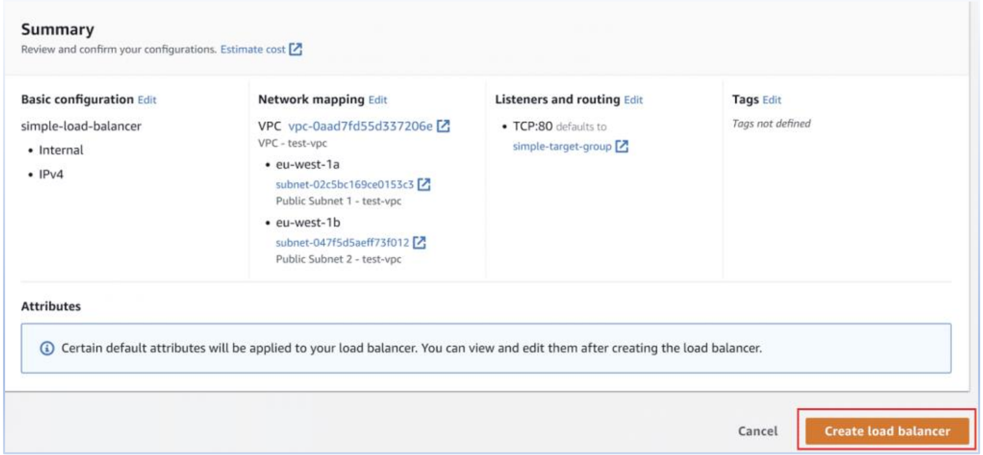
Targets (0)

Remove	Status	Instance ID	Name	Port	State	Security groups	Zone	Subnet ID

No instances added yet
Specify instances above, or leave the group empty if you prefer to add targets later.

0 pending

- Back on the **Create Network Load Balancer** page under **Default action** hit the refresh icon then choose the new target group.
-
- click **Create load balancer**.



Summary
Review and confirm your configurations. Estimate cost

Basic configuration Edit
simple-load-balancer
- Internal
- IPv4

Network mapping Edit
VPC vpc-0aad7fd55d337206e
VPC - test-vpc
- eu-west-1a
  subnet-02c5bc169ce0153c3
  Public Subnet 1 - test-vpc
- eu-west-1b
  subnet-047f5d5aeff73f012
  Public Subnet 2 - test-vpc

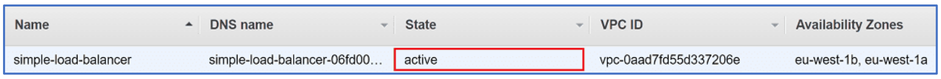
Listeners and routing Edit
- TCP:80 defaults to simple-target-group

Tags Edit
Tags not defined

Attributes

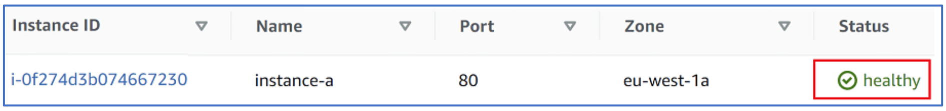
Certain default attributes will be applied to your load balancer. You can view and edit them after creating the load balancer.

- wait for your load balancer to reach the ***active*** state



Name	DNS name	State	VPC ID	Availability Zones
simple-load-balancer	simple-load-balancer-06fd00…	active	vpc-0aad7fd55d337206e	eu-west-1b, eu-west-1a

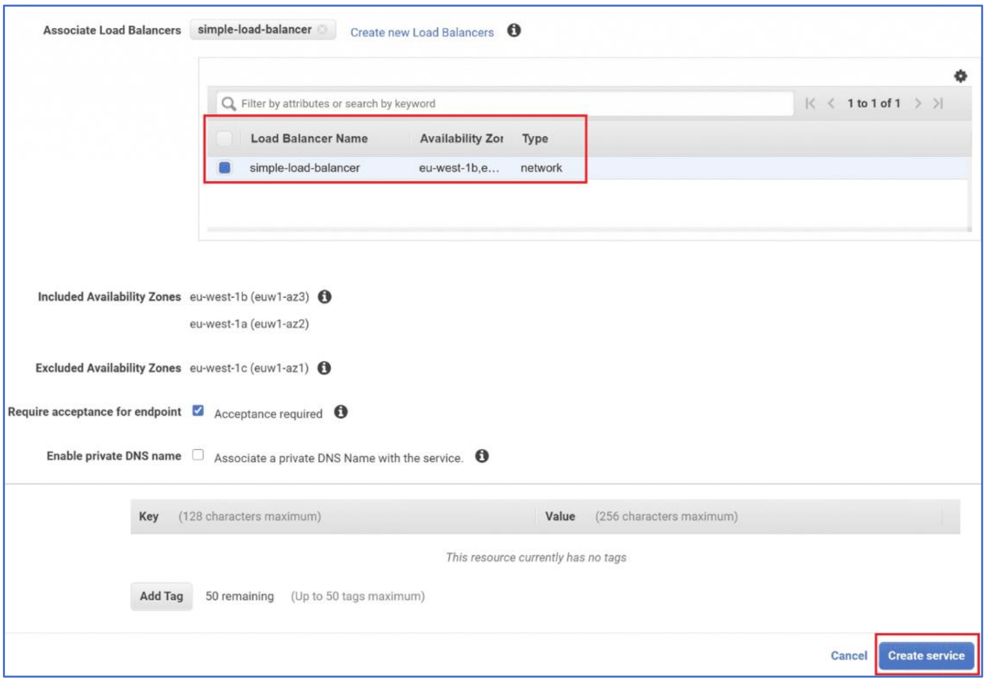
- In the EC2 console go to **Target Groups**, select the target group you just created, then select **Targets**. Hopefully you should see that you have a single target with a *healthy* status.



Instance ID	Name	Port	Zone	Status
i-0f274d3b074667230	instance-a	80	eu-west-1a	✓ healthy

3.6.2. Step 2: create a VPC endpoint service in the provider account account A

- In the VPC dashboard select **Endpoint Services** then **Create Endpoint Service**. Select the NLB you just created then click **Create service**.



Associate Load Balancers simple-load-balancer ⊗ Create new Load Balancers ℹ

Filter by attributes or search by keyword |< < 1 to 1 of 1 > >|

	Load Balancer Name	Availability Zo	Type
■	simple-load-balancer	eu-west-1b,e...	network

Included Availability Zones eu-west-1b (euw1-az3) ℹ
eu-west-1a (euw1-az2)

Excluded Availability Zones eu-west-1c (euw1-az1) ℹ

Require acceptance for endpoint ☑ Acceptance required ℹ

Enable private DNS name □ Associate a private DNS Name with the service. ℹ

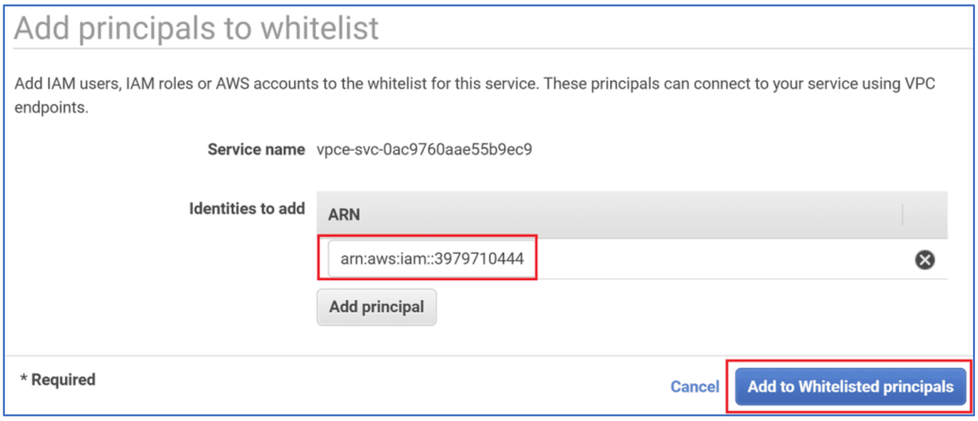
Key (128 characters maximum)	Value (256 characters maximum)

This resource currently has no tags

Add Tag 50 remaining (Up to 50 tags maximum)

Cancel Create service

- Wait for the VPC endpoint to have an *Available* status. Select it, go to **Actions**, then select *Add principals to whitelist*. → add the ARN of the account using the format arn:aws:iam::<aws-account-id>:root. → select **Add to Whitelisted principals**.



Add principals to whitelist

Add IAM users, IAM roles or AWS accounts to the whitelist for this service. These principals can connect to your service using VPC endpoints.

Service name vpce-svc-0ac9760aae55b9ec9

Identities to add

ARN
arn:aws:iam::3979710444

Add principal

* Required

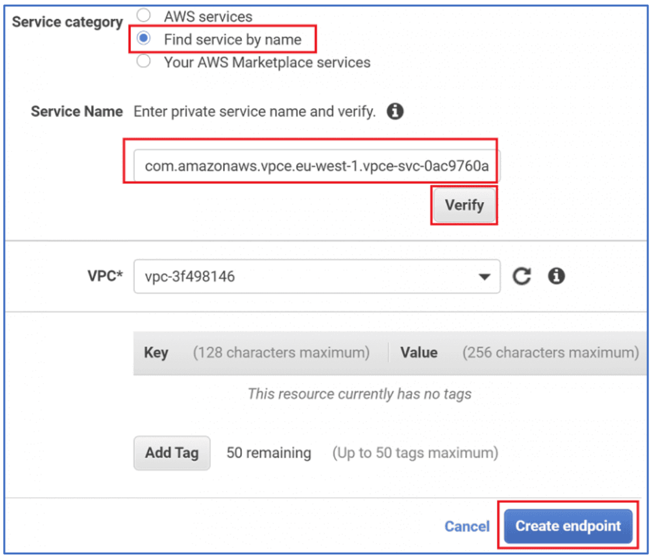
Cancel Add to Whitelisted principals

- Before we move to the next step, go to the VPC endpoint service details page and copy the **Service name** which we'll need later on.

3.6.3. Step 3: create a VPC endpoint in the consumer account

account B

- go to the VPC dashboard and select **Endpoints** then **Create Endpoint**.
  - under **Service category** choose ***Find service by name***
  - enter the service name of the VPC endpoint service you created in the previous step
  - select **Verify** to validate the service name
  - select the VPC where the EC2 instance you want to connect from is deployed
  - under **Security group** select or create a security group which allows inbound access from the EC2 instance you want to connect from
  - select **Create endpoint**



Service category ○ AWS services ● Find service by name ○ Your AWS Marketplace services

Service Name Enter private service name and verify.

com.amazonaws.vpce.eu-west-1.vpce-svc-0ac9760a

VPC* vpc-3f498146

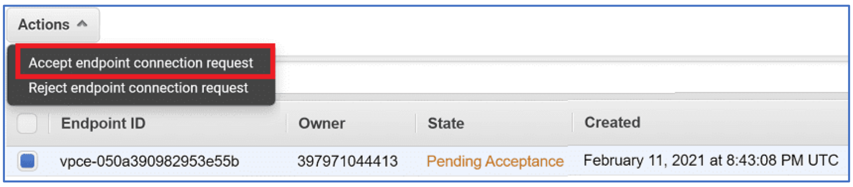
Key (128 characters maximum)	Value (256 characters maximum)

This resource currently has no tags

Add Tag 50 remaining (Up to 50 tags maximum)

accept the request in account A

- VPC dashboard, go to **Endpoint Services**,
- select the endpoint service, then select **Endpoint Connections** where you should see the pending connection.
- Select it, then go to **Actions** and choose **Accept endpoint connection request**.

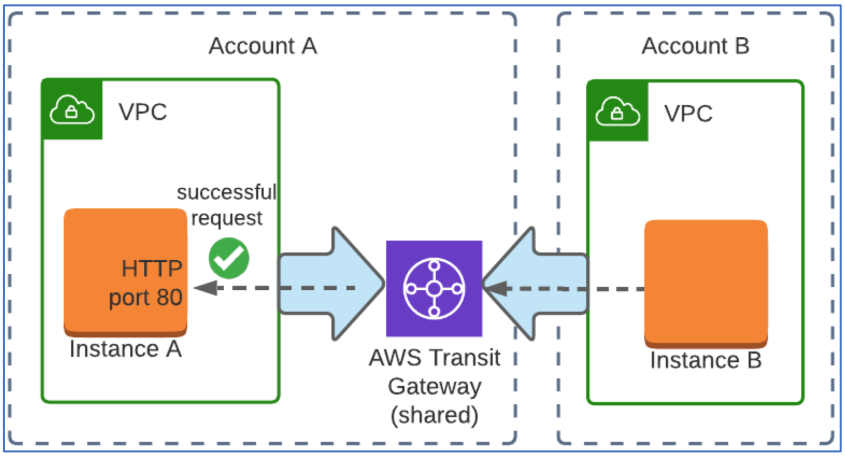


Actions
Accept endpoint connection request
Reject endpoint connection request

Endpoint ID	Owner	State	Created
vpce-050a390982953e55b	397971044413	Pending Acceptance	February 11, 2021 at 8:43:08 PM UTC

- Confirm the acceptance on the popup that appears, then wait for the endpoint to move from *Pending* to ***Available***.
- Back in **account B**, under your endpoint details there should be a list of several **DNS names**. Copy the first one, which allows you to connect to the VPC endpoint from any availability zone.

3.7. Transit gateway cross-account access

- The **AWS Transit Gateway** is a cloud router, which connects multiple VPCs and even on-premises networks through a central hub.
- One of the main benefits is that if you have multiple VPCs which need to be interconnected, then each VPC needs just a single connection to the transit gateway rather than one to each other VPC.
- Transit gateway by default only allows VPCs from the same AWS account to be attached.
- For our cross-account scenario, we'll have to use another AWS service called the **Resource Access Manager** (RAM).
- 

3.7.1. Step 1: create a Transit Gateway

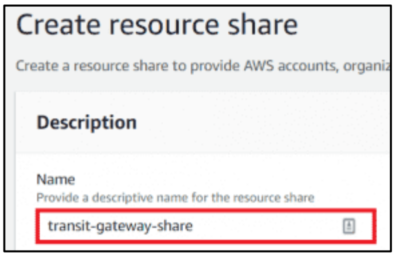
In account A

- From the VPC dashboard → go to **Transit Gateways** → select **Create Transit Gateway**.
- You can optionally give the transit gateway a name, keep all the default settings, then select **Create Transit Gateway**.
- Wait for it to reach the *available* state.

3.7.2. Step 2: share the Transit Gateway using Resource Access Manager

In account A – transit gateway needs to share with Account B

- Go to the Resource Access Manager dashboard, → select **Create a resource share**.
  - give the share a name (e.g. transit-gateway-share)



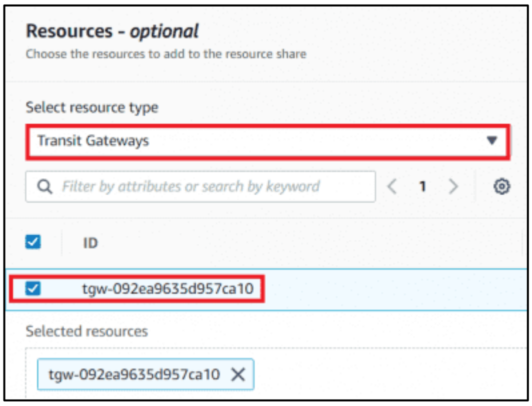
Create resource share

Create a resource share to provide AWS accounts, organiz

Description

Name
Provide a descriptive name for the resource share
transit-gateway-share

- under **Select resource type** choose ***Transit Gateways*** and select the **transit gateway** you just created



Resources - *optional*

Choose the resources to add to the resource share

Select resource type

Transit Gateways

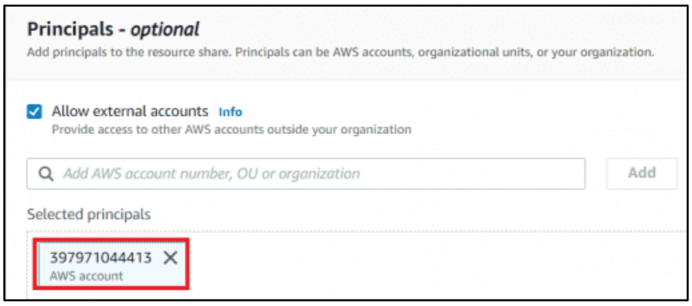
☑ ID

☑ tgw-092ea9635d957ca10

Selected resources

tgw-092ea9635d957ca10 ✕

- under **Principals** add the **account id of the consumer account** you want to share the transit gateway with, then click **Add**



Principals - *optional*

Add principals to the resource share. Principals can be AWS accounts, organizational units, or your organization.

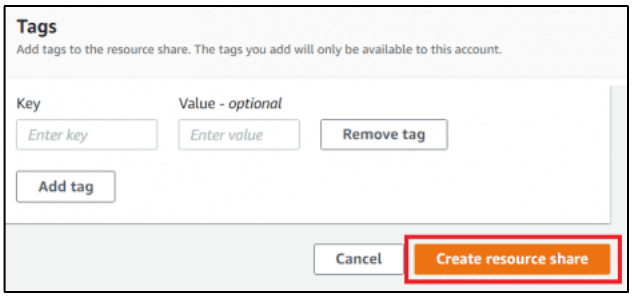
☑ Allow external accounts Info

Provide access to other AWS accounts outside your organization

Selected principals

397971044413 ✕
AWS account

- select **Create resource share**

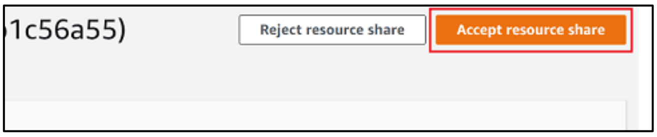


Tags

Add tags to the resource share. The tags you add will only be available to this account.

Accept the resource share into account B

- go to the Resource Access Manager dashboard → Under **Shared with me,** select **Resource shares** and you should see a pending resource share. → Select the name to go into the details page, → select **Accept resource share**

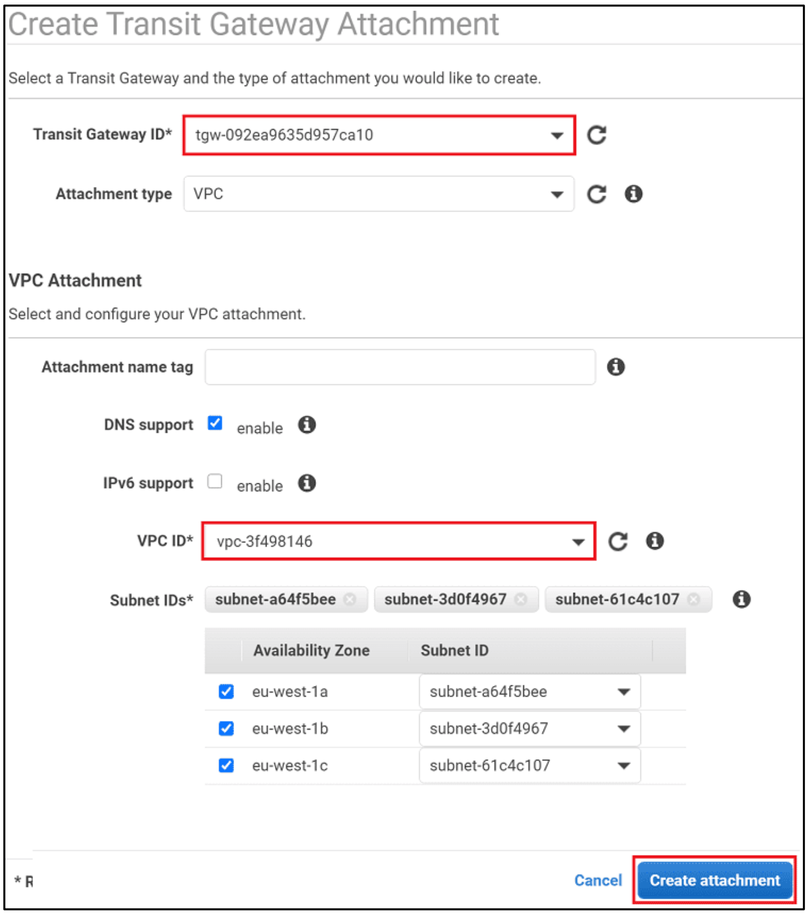


1c56a55)

3.7.3. Step 3: attach both VPCs to the Transit Gateway

in account B

- go to the VPC dashboard and select **Transit Gateway Attachments** → Click **Create Transit Gateway Attachment**.
- For **Transit Gateway ID** pick the transit gateway we just gained access to via the resource share.
- For **VPC ID** select the VPC that contains the instance from which you want to connect to account A, then select **Create attachment**.



Create Transit Gateway Attachment

Select a Transit Gateway and the type of attachment you would like to create.

Transit Gateway ID* tgw-092ea9635d957ca10

Attachment type VPC

VPC Attachment

Select and configure your VPC attachment.

Attachment name tag

DNS support ☑ enable

IPv6 support ☐ enable

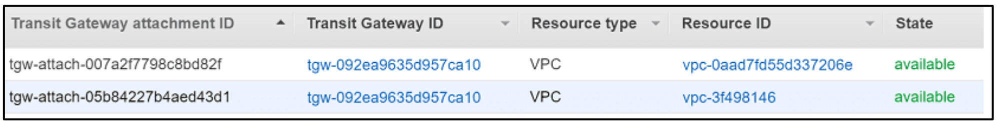
VPC ID* vpc-3f498146

Subnet IDs* subnet-a64f5bee subnet-3d0f4967 subnet-61c4c107

	Availability Zone	Subnet ID
☑	eu-west-1a	subnet-a64f5bee
☑	eu-west-1b	subnet-3d0f4967
☑	eu-west-1c	subnet-61c4c107

into Account A -- accept this connection

- Go to the VPC dashboard and select **Transit Gateway Attachments**. → Select the the attachment with a *pending acceptance* state, and go to **Actions** then choose **Accept**.
- Click **Create Transit Gateway Attachment**, → then for **Transit Gateway ID** → select the transit gateway, for the **VPC ID** → select the VPC with the instance you want to connect to, → then select **Create attachment**.
- After a short time, on the **Transit Gateway Attachments** screen you should have two attachments in the *available* state.

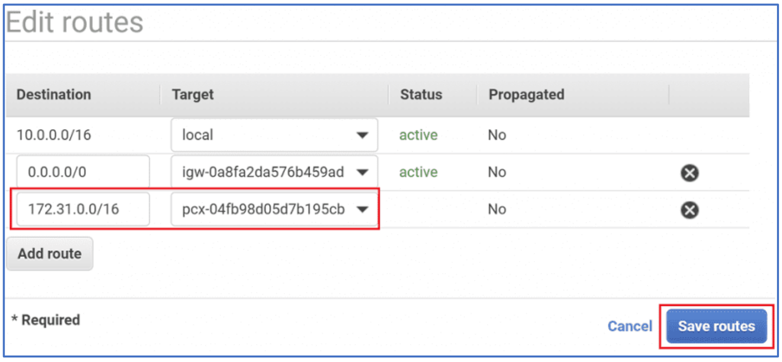


Transit Gateway attachment ID	Transit Gateway ID	Resource type	Resource ID	State
tgw-attach-007a2f7798c8bd82f	tgw-092ea9635d957ca10	VPC	vpc-0aad7fd55d337206e	available
tgw-attach-05b84227b4aed43d1	tgw-092ea9635d957ca10	VPC	vpc-3f498146	available

3.7.4. Step 4: setup the route tables

In **account A**,

- Open the Amazon VPC console at **https://console.aws.amazon.com/vpc/.**
- In the navigation pane, choose **Route Tables**.
- Select the **route table** associated with the subnet containing the instance you want to provide access to,
- Select the **Routes** tab to show the actual routes.
- Choose **Actions**, **Edit routes**.
- Choose **Add route**.
- **Destination** enter the CIDR(172.31.0.0/16) for the VPC is **account B**
- For **Target**, select the **VPC peering connection**, and then choose **Save changes**

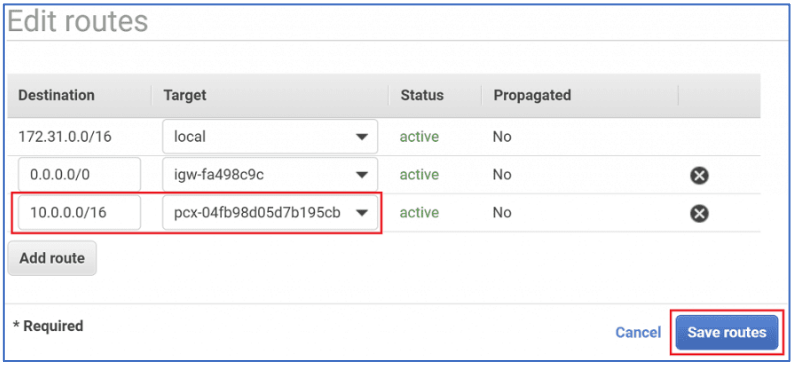


Edit routes

Destination	Target	Status	Propagated
10.0.0.0/16	local	active	No
0.0.0.0/0	igw-0a8fa2da576b459ad	active	No
172.31.0.0/16	pcx-04fb98d05d7b195cb		No

in **account B**

- In the VPC dashboard for account B go to **Route Tables**,
- select the route table for the subnet where your EC2 instance is deployed,
- select the **Routes** tab, **Edit routes**, then **Add route**.
- For the destination enter the CIDR(10.0.0.0/16) of the VPC in account A,
- then for the **target** select the **peering connection**. Finally, select **Save Routes**.



Edit routes

Destination	Target	Status	Propagated
172.31.0.0/16	local	active	No
0.0.0.0/0	igw-fa498c9c	active	No
10.0.0.0/16	pcx-04fb98d05d7b195cb	active	No

3.7.5. Step 4: test the VPC peering connection

- In account A go to the EC2 dashboard, select the instance you want to connect to, then copy the **Private IPv4 address** which we'll need to establish the connection.
- In account B, start a session in the EC2 instance you want to connect from. Make a curl request to the private IP address curl <private-ip-address>.

```
# ping 10.0.0.39
```