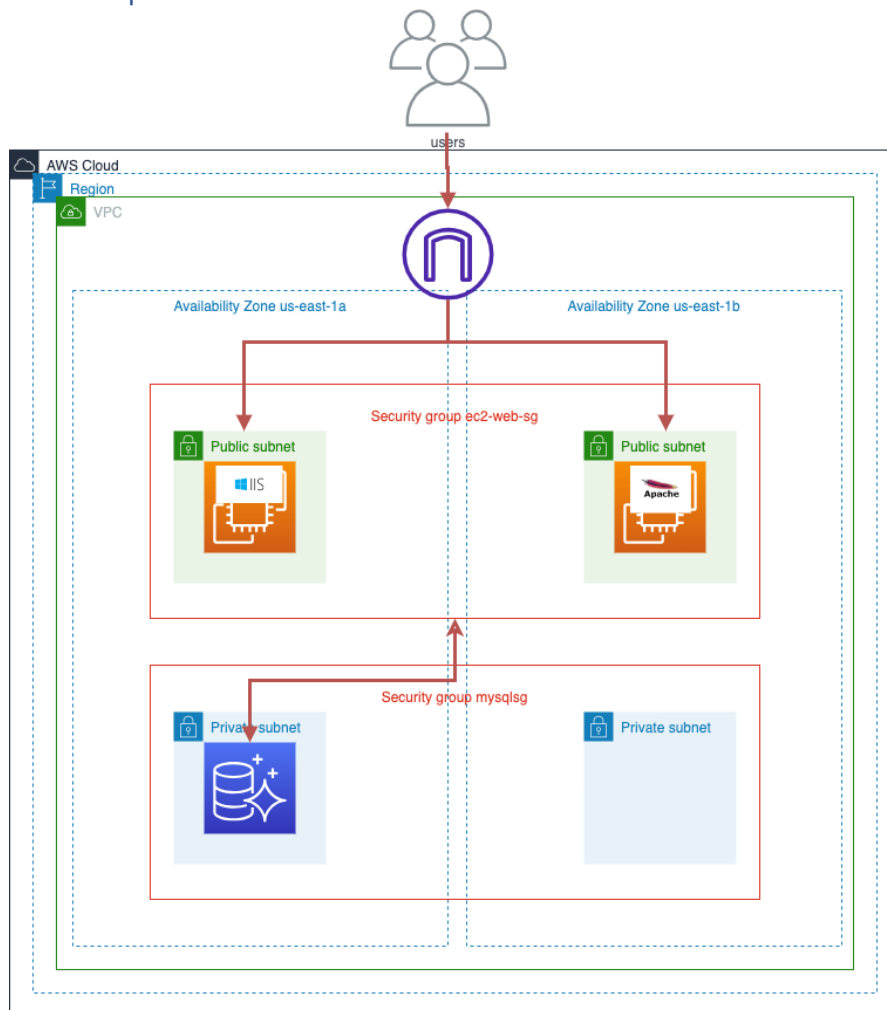


# Creating a VPC with Database and EC2 Instances

## Writeup



The diagram shows the architecture of the project result. The user's requests are received by the internet gateway and forwarded to the webserver of a EC2 instance in the public subnet dependent on the IP-Address. Only the EC2 instances within the security group `ec2-web-sg` are allowed to communicate with the mysql database instance in the private subnet.

## Step by step instruction

### Create Virtual Private Cloud (VPC)

Go to AWS Management Console VPC->Your VPCs->Create VPC

1. Ressources: "VPC and more"
2. Enable Auto-generate: „p1“
3. IPv4 CIDR: „192.168.0.0/16“
4. VPC endpoints: Select None
5. Keep the default for the remaining specifications
6. Create VPC
7. By following these steps, I successfully created and configured my AWS VPC.

The screenshot shows the AWS Management Console interface for VPCs. At the top, there's a section titled 'Your VPCs (1/2)' with a search bar and a table of VPCs. The table has columns for Name, VPC ID, State, IPv4 CIDR, IPv6 CIDR, and DHCP option set. One VPC, 'p1-vpc', is selected. Below the table, the details for 'vpc-094eea3bc4a186570 / p1-vpc' are shown. The details are organized into a grid with sections for VPC ID, State, DNS hostnames, DNS resolution, Tenancy, DHCP option set, Main route table, Main network ACL, Default VPC, IPv4 CIDR, IPv6 pool, IPv6 CIDR (Network border group), Network Address Usage metrics, Route 53 Resolver DNS Firewall rule groups, and Owner ID.

Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP option set
-	vpc-0ee308d415a1b838e	Available	172.31.0.0/16	-	dopt-0708b32fe0b27d...
p1-vpc	vpc-094eea3bc4a186570	Available	192.168.0.0/16	-	dopt-0708b32fe0b27d...

**Details**

VPC ID vpc-094eea3bc4a186570	State Available	DNS hostnames Enabled	DNS resolution Enabled
Tenancy Default	DHCP option set dopt-0708b32fe0b27dd67	Main route table rtb-0aad4c56aa9193fa0	Main network ACL acl-05a02584c886ee6b5
Default VPC No	IPv4 CIDR 192.168.0.0/16	IPv6 pool -	IPv6 CIDR (Network border group) -
Network Address Usage metrics Disabled	Route 53 Resolver DNS Firewall rule groups -	Owner ID 808865497406	

8. By following these steps, I successfully created and configured my subnets for my VPC.

The screenshot shows the AWS Management Console interface for Subnets. At the top, there's a section titled 'Subnets (4/4)' with a search bar and a table of subnets. The table has columns for Name, Subnet ID, State, VPC, IPv4 CIDR, and IPv6 CIDR. Four subnets are listed, all with a state of 'Available' and associated with the VPC 'vpc-094eea3bc4a186570 | p1-vpc'.

Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR
p1-subnet-db-us-east-1b	subnet-0ecbf9d603dd93b4	Available	vpc-094eea3bc4a186570   p1-vpc	192.168.144.0/20	-
p1-subnet-web-us-east-1a	subnet-0344a9391e7bb7bf4	Available	vpc-094eea3bc4a186570   p1-vpc	192.168.0.0/20	-
p1-subnet-db-us-east-1a	subnet-0b0315476086c7c9b	Available	vpc-094eea3bc4a186570   p1-vpc	192.168.128.0/20	-
p1-subnet-web-us-east-1b	subnet-0bcc5567b402056a6	Available	vpc-094eea3bc4a186570   p1-vpc	192.168.16.0/20	-

9. By following these steps, I successfully created my internet gateway and attached it to my VPC.

The screenshot shows the AWS Management Console interface for Internet gateways. At the top, there's a section titled 'Internet gateways (1/1)' with a search bar and a table of internet gateways. The table has columns for Name, Internet gateway ID, State, VPC ID, and Owner. One internet gateway, 'p1-igw', is listed with a state of 'Attached' and associated with the VPC 'vpc-094eea3bc4a186570 | p1-vpc'.

Name	Internet gateway ID	State	VPC ID	Owner
p1-igw	igw-0cb5652ff714eb82f	Attached	vpc-094eea3bc4a186570   p1-vpc	808865497406

10. By following these steps, I successfully created my route tables for my VPC.  
Rename public routing table to “p1-rtb-web” and add web-subnets to routing table

You have successfully deleted rtb-0f0e9e6ed07bed5d6 / p1-rtb-private2-us-east-1b

Route tables (1/4) Info

Find resources by attribute or tag

Name

Route table ID

Explicit subnet associati...

Edge associations

Main

VPC

Own...

p1-rtb-web

rtb-078eb9b522c2799f3

2 subnets

-

No

vpc-094eea3bc4a186570 | p1-vpc

808865...

-

rtb-0aad4c56aa9193fa0

-

-

Yes

vpc-094eea3bc4a186570 | p1-vpc

808865...

p1-rtb-db

rtb-055fc87d67354af05

subnet-0b0315476086c7...

-

No

vpc-094eea3bc4a186570 | p1-vpc

808865...

-

rtb-0dc9ceba28fd4fd73

-

-

Yes

vpc-0ee308d415a1b838e

808865...

rtb-078eb9b522c2799f3 / p1-rtb-web

Details

Routes

Subnet associations

Edge associations

Route propagation

Tags

Routes (2)

Edit routes

Filter routes

Both

Destination

Target

Status

Propagated

0.0.0.0/0

lgw-0cb5652ff714eb82f

Active

No

192.168.0.0/16

local

Active

No

You have successfully deleted rtb-0f0e9e6ed07bed5d6 / p1-rtb-private2-us-east-1b

Route tables (1/4) Info

Find resources by attribute or tag

Name

Route table ID

Explicit subnet associati...

Edge associations

Main

VPC

Own...

p1-rtb-web

rtb-078eb9b522c2799f3

2 subnets

-

No

vpc-094eea3bc4a186570 | p1-vpc

808865...

-

rtb-0aad4c56aa9193fa0

-

-

Yes

vpc-094eea3bc4a186570 | p1-vpc

808865...

p1-rtb-db

rtb-055fc87d67354af05

subnet-0b0315476086c7...

-

No

vpc-094eea3bc4a186570 | p1-vpc

808865...

-

rtb-0dc9ceba28fd4fd73

-

-

Yes

vpc-0ee308d415a1b838e

808865...

rtb-078eb9b522c2799f3 / p1-rtb-web

Details

Routes

Subnet associations

Edge associations

Route propagation

Tags

Explicit subnet associations (2)

Edit subnet associations

Find subnet association

Name

Subnet ID

IPv4 CIDR

IPv6 CIDR

p1-subnet-web-us-east-1a

subnet-0344a9391e7bb7bf4

192.168.0.0/20

-

p1-subnet-web-us-east-1b

subnet-0bcc5567b402056a6

192.168.16.0/20

-

Rename private routing table to “p1-rtb-db” and add private db-subnets to routing table

Route tables (1/4) Info

Find resources by attribute or tag

Name

Route table ID

Explicit subnet associati...

Edge associations

Main

VPC

Own...

p1-rtb-web

rtb-078eb9b522c2799f3

2 subnets

-

No

vpc-094eea3bc4a186570 | p1-vpc

808865...

-

rtb-0aad4c56aa9193fa0

-

-

Yes

vpc-094eea3bc4a186570 | p1-vpc

808865...

p1-rtb-db

rtb-055fc87d67354af05

2 subnets

-

No

vpc-094eea3bc4a186570 | p1-vpc

808865...

-

rtb-0dc9ceba28fd4fd73

-

-

Yes

vpc-0ee308d415a1b838e

808865...

rtb-055fc87d67354af05 / p1-rtb-db

Details

Routes

Subnet associations

Edge associations

Route propagation

Tags

Explicit subnet associations (2)

Edit subnet associations

Find subnet association

Name

Subnet ID

IPv4 CIDR

IPv6 CIDR

p1-subnet-db-us-east-1b

subnet-0ecbff9d603dd93b4

192.168.144.0/20

-

p1-subnet-db-us-east-1a

subnet-0b0315476086c7c9b

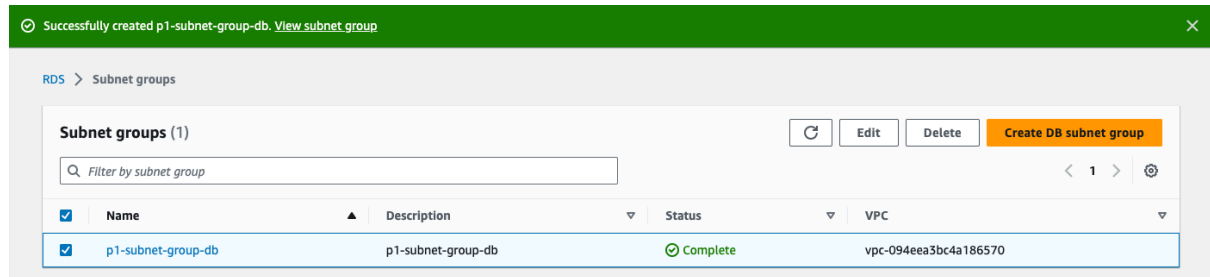
192.168.128.0/20

-

## Create database subnet group

Go to AWS Management Console RDS->Subnet groups->Create db subnet group

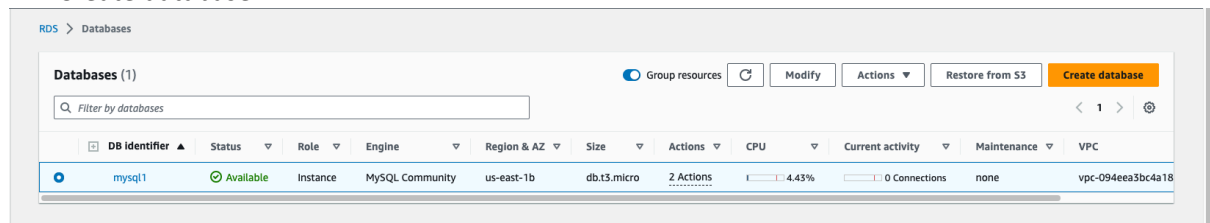
1. Name: "p1-subnet-group-db"
2. VPC: p1-vpc
3. Availability Zone: Select the zones from the private db-subnets
4. Subnets: Select the private db-subnets
5. Create



## Create relational database

Go to AWS Management Console RDS->Create database

6. Select Standard create
7. Engine options: Select MySQL
8. Engine Version: Select newest Version MySQL 8.0.34
9. Templates: Dev/Test with Multi-AZ DB instance
10. DB instance identifier: "mysql1"
11. Type Master Password/Confirm master password
12. DB instance classe: Select Burstable classes: db.t3.micro
13. Storage type: Select General Purpose SSD (gp2)
14. Allocated storage: 20
15. Maximum storage threshold: 50
16. Connectivity. Compute resource: Select Don't connect to an EC2 compute resource
17. VPC: Select p1-vpc
18. VPC security group: Create new with security group name: "mysqlsg"
19. Public Access: No
20. Database authentication options: Password and IAM authentication
21. Disable Monitoring
22. Create database



## Create and launch the EC2-Windows instance

Go to AWS Management Console EC2->Instances->Launch an instance

1. Name: "ec2-web"
2. Application and OS Images: Select Windows
3. Instance type: Select t2.micro
4. Key pair(login) : Create new key pair  
Key pair name: "windows1"  
Create key pair
5. Edit Network settings.  
VPC: Select "p1-vpc"  
Subnet: Select one of the web-subnet
6. Enable Auto-assign public IP
7. Security group name: "ec2-web-sg"
8. Firewall->Add security group rule: Type=Http, Sourcetype=Anywhere
9. UserData: Add commands:  
<powershell>  
Install-WindowsFeature -name Web-Server -IncludeManagementTools  
New-Item -Path C:\inetpub\wwwroot\index.html -ItemType File -Value "Hello World Page" -  
Force  
</powershell>
10. Launch instance
11. Keep the default for the remaining specifications
12. By following these steps, I successfully launch my new EC2 instance within my VPC.

Instances (1/2) <a href="#">Info</a>										
<input type="text" value="Find instance by attribute or tag (case-sensitive)"/>										
<input type="checkbox"/>	ec2-web-linux	<a href="#">i-048df2720d7680213</a>	<span>Running</span>		t2.micro	<span>2/2 checks passed</span>	No alarms	+	us-east-1b	<a href="#">ec2-34-204-52-244.co...</a>
<input checked="" type="checkbox"/>	ec2-web	<a href="#">i-0a655be89575ed39a</a>	<span>Running</span>		t2.micro	<span>2/2 checks passed</span>	No alarms	+	us-east-1a	<a href="#">ec2-23-22-37-143.com...</a>

### Instance: i-0a655be89575ed39a (ec2-web)

#### ▼ Instance summary [Info](#)

Instance ID  
[i-0a655be89575ed39a](#) (ec2-web)

IPv6 address  
-

Hostname type  
IP name: [ip-192-168-11-20.ec2.internal](#)

Answer private resource DNS name  
-

Auto-assigned IP address  
[23.22.37.143](#) [Public IP]

IAM Role  
-

Public IPv4 address  
[23.22.37.143](#) | [open address](#)

Instance state  
Running

Private IP DNS name (IPv4 only)  
[ip-192-168-11-20.ec2.internal](#)

Instance type  
t2.micro

VPC ID  
[vpc-094eea3bc4a186570](#) (p1-vpc)

Subnet ID  
[subnet-0344a9391e7bb7bf4](#) (p1-subnet-web-us-east-1a)

Private IPv4 addresses  
[192.168.11.20](#)

Public IPv4 DNS  
[ec2-23-22-37-143.compute-1.amazonaws.com](#) | [open address](#)

Elastic IP addresses  
-

AWS Compute Optimizer finding  
[Opt-in to AWS Compute Optimizer for recommendations.](#) | [Learn more](#)

Auto Scaling Group name  
-



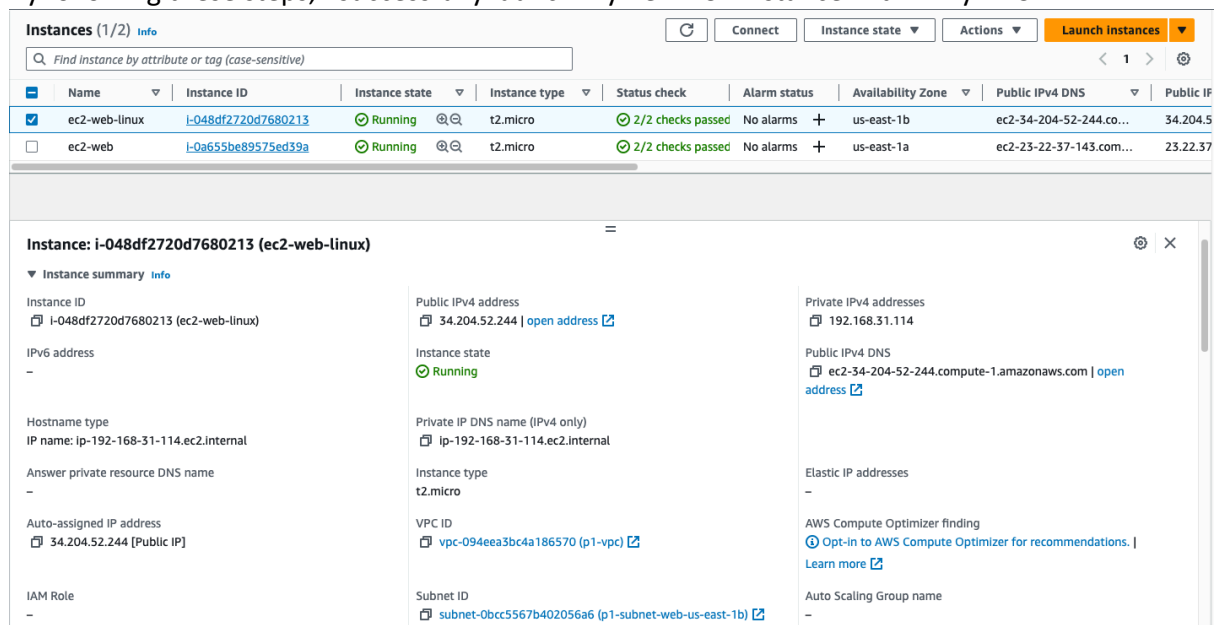
Hello World Page

## Create and launch the EC2-Linux instance

Go to AWS Management Console EC2->Instances->Launch an instance

1. Name: "ec2-web-linux"
2. Application and OS Images: Select Amazon Linux
3. Instance type: Select t2.micro
4. Key pair(login) : Create new key pair  
Key pair name: "linux1"  
Create key pair
5. Edit Network settings.  
VPC: Select "p1-vpc"  
Subnet: Select one of the web-subnet
6. Firewall->Select existing security group rule: "ec2-web-sg"
7. Launch instance
8. Keep the default for the remaining specifications

By following these steps, I successfully launch my new EC2 instance within my VPC.



The screenshot displays the AWS Management Console's EC2 Instances page. At the top, there's a search bar and buttons for 'Connect', 'Instance state', 'Actions', and 'Launch instances'. Below this is a table of instances. Two instances are listed: 'ec2-web-linux' (ID: i-048df2720d7680213) and 'ec2-web' (ID: i-0a655be89575ed39a). Both are in a 'Running' state. The details pane for the 'ec2-web-linux' instance is expanded, showing various attributes:

Instance: i-048df2720d7680213 (ec2-web-linux)		
<b>Instance summary</b>		
Instance ID i-048df2720d7680213 (ec2-web-linux)	Public IPv4 address 34.204.52.244   <a href="#">open address</a>	Private IPv4 addresses 192.168.31.114
IPv6 address -	Instance state Running	Public IPv4 DNS ec2-34-204-52-244.compute-1.amazonaws.com   <a href="#">open address</a>
Hostname type IP name: ip-192-168-31-114.ec2.internal	Private IP DNS name (IPv4 only) ip-192-168-31-114.ec2.internal	Elastic IP addresses -
Answer private resource DNS name -	Instance type t2.micro	AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations.   <a href="#">Learn more</a>
Auto-assigned IP address 34.204.52.244 [Public IP]	VPC ID vpc-094eea3bc4a186570 (p1-vpc)	Auto Scaling Group name -
IAM Role -	Subnet ID subnet-0bcc5567b402056a6 (p1-subnet-web-us-east-1b)	

## Extend inbound rule of security group "ec2-web-sg" to enable ssh

Go to AWS Management Console EC2->Security groups->"ec2-web-sg">Edit inbound rules

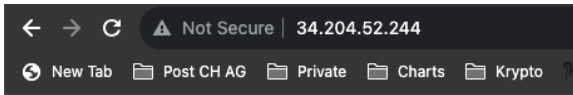
1. Add rule: Type=SSH, Source=Custom:0.0.0.0/0
2. Save rules

## Install Apache Web-Server on EC2-Linux instance

Go to AWS Management Console EC2->Instances->EC2-Linux instance->Connect to instance

Select EC2 Instance Connect

3. sudo su
4. yum install httpd -y
5. service httpd status (optional:check status)
6. service httpd start



**It works!**

Extend inbound rule of db security group “mysqlsg” to enable access from ec2-web-sg  
Go to AWS Management Console EC2->Security groups->”mysqlsg”->Edit inbound rules

1. Add rule: Type=All traffic, Source=Custom: “ex2-weg-sg”
2. Add rule: Type=All traffic, Source=Custom: “ex2-weg-sg”
3. Save rules

EC2 > Security Groups > sg-0810a46da4d7326f7 - mysqlsg

### sg-0810a46da4d7326f7 - mysqlsg Actions ▾

**Details**

Security group name mysqlsg	Security group ID sg-0810a46da4d7326f7	Description Created by RDS management console	VPC ID vpc-094eea3bc4a186570 <a href="#">↗</a>
Owner 808865497406	Inbound rules count 3 Permission entries	Outbound rules count 1 Permission entry	

**Inbound rules** | Outbound rules | Tags

**Inbound rules (3)** ↻ Manage tags Edit inbound rules

< **1** > ⚙

<input type="checkbox"/>	Name ▾	Security group rule... ▾	IP version ▾	Type ▾	Protocol ▾	Port range ▾	Source
<input type="checkbox"/>	-	sgr-038afa389e90c374b	-	All traffic	All	All	sg-028f02a32
<input type="checkbox"/>	-	sgr-0248a47dd2b47e...	IPv4	MYSQL/Aurora	TCP	3306	194.230.147.6
<input type="checkbox"/>	-	sgr-053bd448c0235d...	-	All traffic	All	All	sg-0810a46da