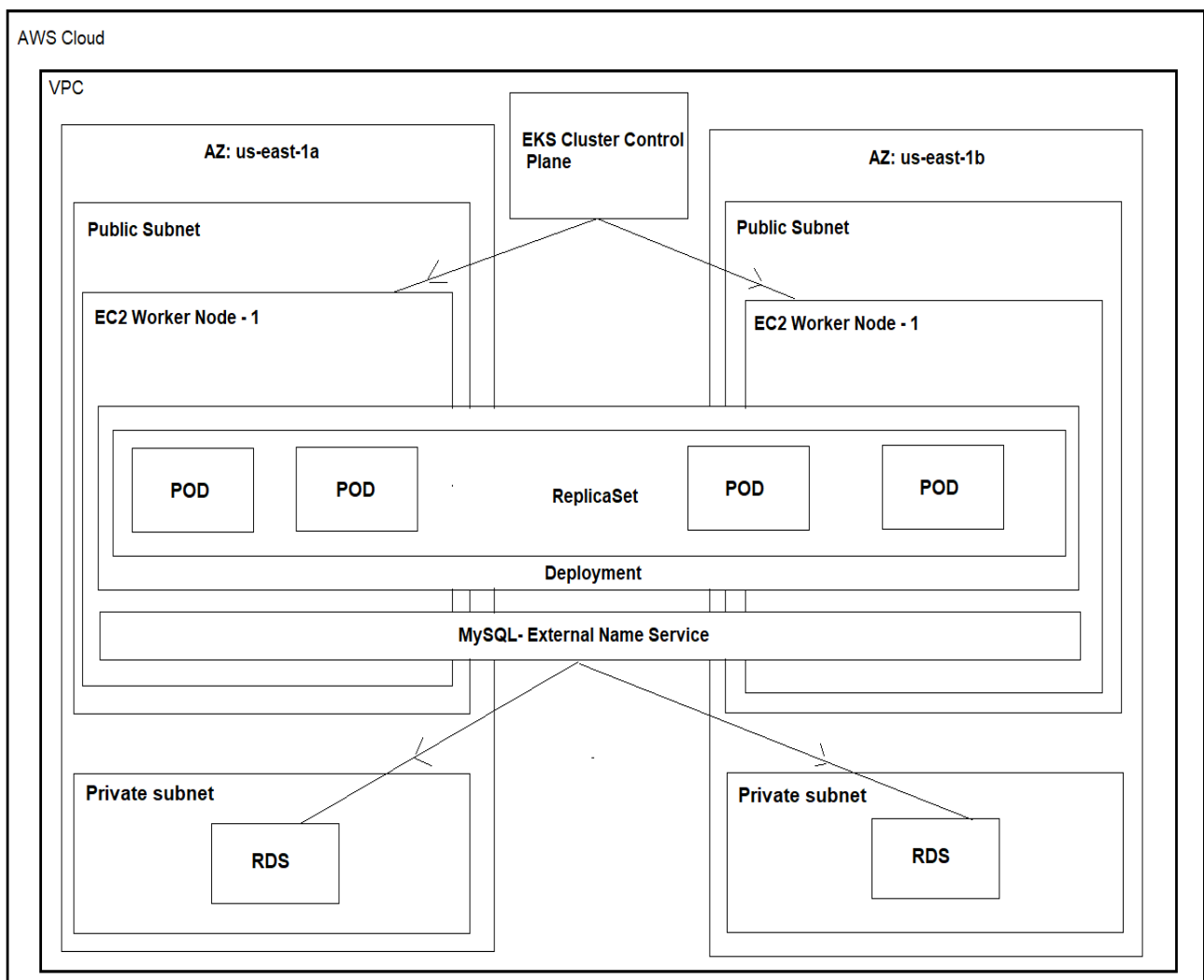


AWS EKS Storage – RDS

Before going to RDS let's see drawback of EBS CSI for MySQL DB.

- Complex setup to achieve HA.
- Complex Multi-AZ support for EBS.
- Complex Master-Master MySQL setup.
- Complex Master-Slave MySQL setup.
- No automatic Backup and recovery.
- No auto-upgrade MySQL.

Diagram:



Steps:

1. Create RDS Database

- Review VPC of our EKS Cluster.
- Go to Services -> VPC.
- VPC Name: eksctl-eksdemo1-cluster/VPC.
- There are default 2 public and private subnet.
- Pre-requisite-1: Create DB Security Group.

2. Create security group to allow access for RDS Database on port 3306

- Security group name: eks_rds_db_sg
- Description: Allow access for RDS Database on Port 3306
- VPC: eksctl-eksdemo1-cluster/VPC
- Inbound Rules
- Type: MySQL/Aurora
- Protocol: TCP
- Port: 3306
- Source: Anywhere (0.0.0.0/0)
- Description: Allow access for RDS Database on Port 3306
- Outbound Rules
 - Leave to defaults
- Pre-requisite-2: Create DB Subnet Group in RDS
- Go to RDS -> Subnet Groups

EC2 > Security Groups > Create security group

Create security group

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name:

Description:

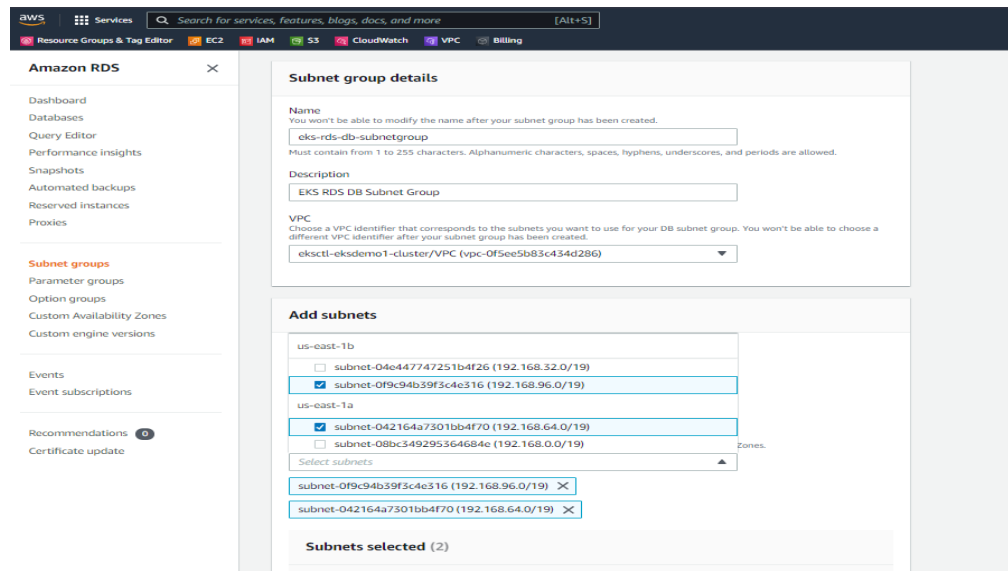
VPC:

Inbound rules

Type	Protocol	Port range	Source	Description - optional	
MySQL/Aurora	TCP	3306	Anywhere-I...	Allow access for RDS Database on Port 3306	Delete
MySQL/Aurora	TCP	3306	Anywhere-I...	Allow access for RDS Database on Port 3306	Delete

3. Click on Create DB Subnet Group

- Name: eks-rds-db-subnetgroup
- Description: EKS RDS DB Subnet Group
- VPC: eksctl-eksdemo1-cluster/VPC
- Availability Zones: us-east-1a, us-east-1b
- Subnets: 2 subnets in 2 AZs
- Click on Create



The screenshot shows the AWS Management Console interface for creating an Amazon RDS DB Subnet Group. The left sidebar contains navigation links for various AWS services. The main content area is titled 'Subnet group details' and includes the following sections:

- Name:** A text input field containing 'eks-rds-db-subnetgroup'. A note states: 'You won't be able to modify the name after your subnet group has been created. Must contain from 1 to 255 characters. Alphanumeric characters, spaces, hyphens, underscores, and periods are allowed.'
- Description:** A text input field containing 'EKS RDS DB Subnet Group'.
- VPC:** A dropdown menu showing 'eksctl-eksdemo1-cluster/VPC (vpc-0f5ee5b83c434d286)'. A note states: 'Choose a VPC identifier that corresponds to the subnets you want to use for your DB subnet group. You won't be able to choose a different VPC identifier after your subnet group has been created.'
- Add subnets:** A section for selecting subnets across different Availability Zones.
 - us-east-1b:** Two subnets are listed: 'subnet-04e47747251b4f26 (192.168.32.0/19)' (unchecked) and 'subnet-0f9c94b39f3c4e316 (192.168.96.0/19)' (checked).
 - us-east-1a:** Two subnets are listed: 'subnet-042164a7301bb4f70 (192.168.64.0/19)' (checked) and 'subnet-08bc349295364684e (192.168.0.0/19)' (unchecked).
- Subnets selected:** A summary bar at the bottom indicating '(2)' subnets are selected.

4. Create RDS Database

- Create RDS Database
- Go to Services -> RDS
- Click on Create Database
- Choose a Database Creation Method: Standard Create
- Engine Options: MySQL
- Edition: MySQL Community
- Version: 5.7.22 (default populated)
- Template Size: Free Tier
- DB instance identifier: usermgmtddb
- Master Username: dbadmin
- Master Password: dbpassword11
- Confirm Password: dbpassword11
- DB Instance Size: leave to defaults
- Storage: leave to defaults
- Connectivity
- VPC: eksctl-eksdemo1-cluster/VPC
- Additional Connectivity Configuration
- Subnet Group: eks-rds-db-subnetgroup
- Publicly accessible: YES (for our learning and troubleshooting - if required)
- VPC Security Group: Create New
- Name: eks-rds-db-securitygroup
- Availability Zone: No Preference
- Database Port: 3306
- Rest all leave to defaults
- Click on Create Database
- Edit Database Security to Allow Access from 0.0.0.0/0
 - o Go to EC2 -> Security Groups -> eks-rds-db-securitygroup
 - o Edit Inbound Rules
 - o Source: Anywhere (0.0.0.0/0) (Allow access from everywhere for now)

5. Create Kubernetes externalName service Manifest and Deploy

```
[root@localhost RDS]# ls
deployment.yml  externalName_service.yml  secret.yml  service.yml
[root@localhost RDS]# cat externalName_service.yml
apiVersion: v1
kind: Service
metadata:
  name: mysql
spec:
  type: ExternalName
  externalName: usermgmtdb.c6cu3zvd3a.us-east-1.rds.amazonaws.com

[root@localhost RDS]# |
```

```
[root@localhost RDS]# kubectl apply -f externalName_service.yml
service/mysql created
[root@localhost RDS]# kubectl get svc
NAME          TYPE          CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
kubernetes    ClusterIP     10.100.0.1   <none>        443/TCP    142m
mysql         ExternalName  <none>       usermgmtdb.c6cu3zvd3a.us-east-1.rds.amazonaws.com  <none>     7s
[root@localhost RDS]# |
```

6. Connect to RDS Database using kubectl and create usermgmt DB.

```
[root@localhost RDS]# kubectl run -it --rm --image=mysql:5.7.22 --restart=Never mysql-client -- mysql -h usermgmtdb.c6cu3zvd3a.us-east-1.rds.amazonaws.com -u dbadmin -pdbpa
ssword11
If you don't see a command prompt, try pressing enter.

mysql> create database usermgmt;
Query OK, 1 row affected (0.01 sec)

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| usermgmt |
+-----+
5 rows in set (0.00 sec)

mysql> exit|
```

7. Create web application deployment and service manifest file and deploy it.

```
[root@localhost RDS]# kubectl apply -f ../RDS/
deployment.apps/usermgmt-microservice unchanged
service/mysql unchanged
secret/mysql-db-password created
service/usermgmt-restapp-service unchanged
[root@localhost RDS]# kubectl get pods -o wide
NAME                                READY    STATUS    RESTARTS   AGE   IP              NODE                                NOMINATED NODE   READINESS GATES
usermgmt-microservice-9b9bb96b6-h24n7 0/1      Running   0           68s   192.168.52.43   ip-192-168-40-118.ec2.internal      <none>            <none>
[root@localhost RDS]# kubectl get svc
NAME                TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
kubernetes          ClusterIP   10.100.0.1       <none>            443/TCP          152m
mysql               ExternalName <none>           usermgmtdb.c6cu3zvd3a.us-east-1.rds.amazonaws.com <none>            7m29s
usermgmt-restapp-service NodePort    10.100.167.6     <none>            8095:31231/TCP   78s
[root@localhost RDS]# kubectl run -it --rm --image=mysql:5.7.22 --restart=Never mysql-client -- mysql -h usermgmtdb.c6cu3zvd3a.us-east-1.rds.amazonaws.com -u dbadmin -pdbpa
ssword11
If you don't see a command prompt, try pressing enter.

mysql> use usermgmt;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables_in_usermgmt |
+-----+
| users               |
+-----+
1 row in set (0.00 sec)

mysql> |
```

8. Access Application by using Node public IP Address

```
[root@localhost RDS]# kubectl get pods -o wide
NAME                                READY    STATUS    RESTARTS   AGE   IP              NODE                                NOMINATED NODE   READINESS GATES
usermgmt-microservice-9b9bb96b6-h24n7 1/1      Running   0           4m3s   192.168.52.43   ip-192-168-40-118.ec2.internal      <none>            <none>
[root@localhost RDS]# kubectl get nodes -o wide
NAME                                STATUS    ROLES    AGE   VERSION          INTERNAL-IP    EXTERNAL-IP      OS-IMAGE          KERNEL-VERSION      CONTAINER-RUN
TIME
ip-192-168-26-134.ec2.internal    Ready     <none>    140m   v1.21.5-eks-bc4871b 192.168.26.134  54.84.150.2      Amazon Linux 2    5.4.156-83.273.amzn2.x86_64 docker://20.1
0.7
ip-192-168-40-118.ec2.internal    Ready     <none>    140m   v1.21.5-eks-bc4871b 192.168.40.118  18.234.238.188   Amazon Linux 2    5.4.156-83.273.amzn2.x86_64 docker://20.1
0.7
[root@localhost RDS]# |
```