**List of resources provisioned through Cloudformation templates:**

**Parameters given as user input**

1. Environment tag
2. Virtual private cloud id (list will be displayed)
3. Private Subnet ids (list will be displayed)

**Aurora:**

1. Aurora DB port
2. Aurora KMS key id
3. Subnet IDs for opensearch cluster (list will be displayed)

**Opensearch:**

1. Opensearch KMS key id
2. Opensearch instance count
3. Opensearch instance type
4. Opensearch Volume type
5. Opensearch Volume size
6. Opensearch availability zone count

**Elasticache-memcached**

1. Elasticache node type
2. Elasticache parameter group name
3. Elasticache cluster name
4. Number of nodes

**S3 bucket**

1. Bucket name
2. Logging bucket name
3. Retention policy for Glacier class
4. Retention policy for Deep archive class
5. KMS key id

**SNS**

1. KMS key id

**Backups**

1. KMS key id

**Elasticache-Redis**

1. KMS key id

**Roles created:**

1. EFS
2. Aurora
3. EKS
4. Fargate profile
5. Lambda execution
6. Role for aurora IAM based authentication
7. Role for enhanced monitoring
8. Backups

**Security groups created:**

1. Cluster
2. EFS
3. Aurora
4. Elasticache-memached
5. Elasticache-redis
6. **Elastic Kubernetes Service cluster**

**Description** : An EKS cluster with fargate profile

**Parameters:**

Version:1.27 (latest)

Subnets: Private subnets in vpc ( Parameter inputs for Cloud formation )

Logging: API and audit logs enabled ( as per recommendation)

Endpoint private access: true ( as per ppt)

Endpoint public access: false

Metrics: CPU and memory (enabled by default)

**Integration:**

Created a security group for eks cluster with ingress and egress rule

This security group is added is ingress reference in EFS, Aurora, Memcached and Redis

It is also added as a reference security group for Opensearch cluster

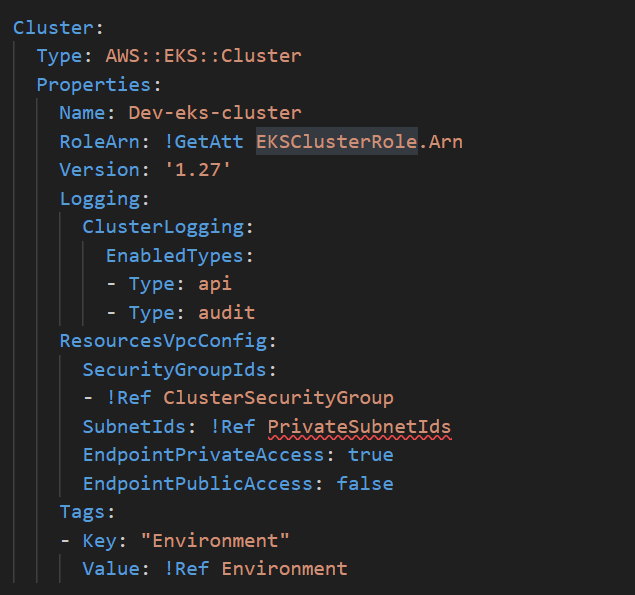
**Roles:**

A specific role created (EKSClusterRole) with policy references for:

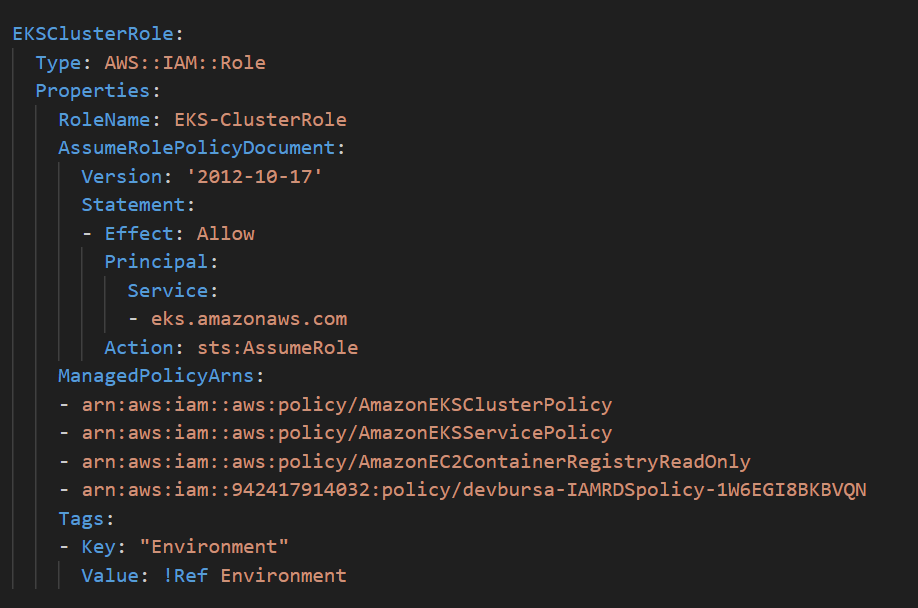
1. EKS cluster policy
2. EKS service policy
3. ECR Read only

**Template:**

1. Resource



1. Role



**2) Fargate profile**

To run containers without actually having to manage EC2 instances

**Parameters:**

Depends on EKS cluster creation

Pod execution role: the components that run on the Fargate infrastructure must make calls to AWS APIs

Subnets: private subnets

Namespaces: default, kube-system, dev

**Integration:**

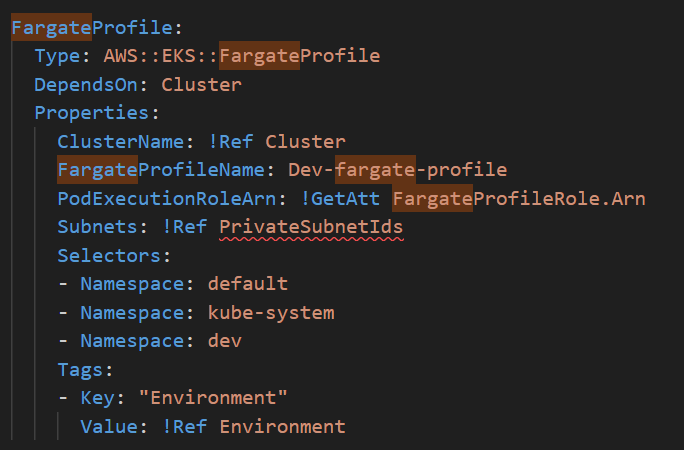
Added pod execution role policy to user created for EFS. (required for mounting EFS to EKS)

**Role:**

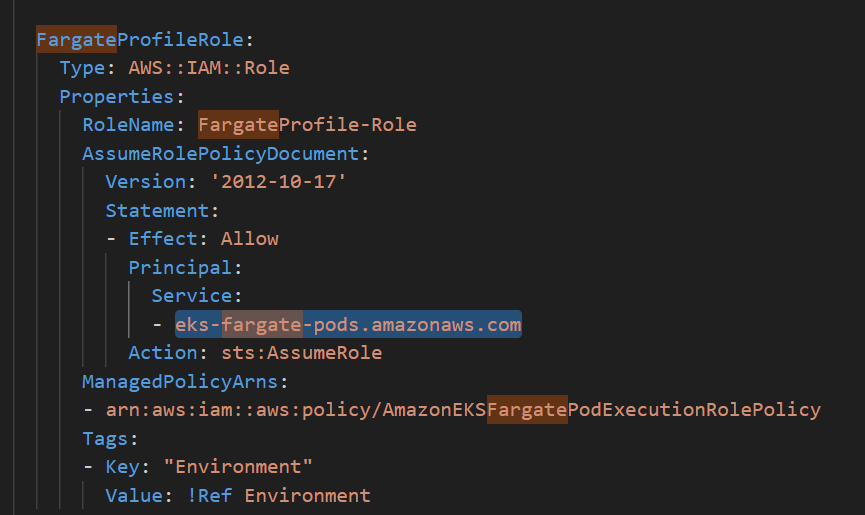
A role created which allows privileges to eks-fargate-pods.amazonaws.com service, with AWS Fargate pod execution policy as reference

**Template:**

1. Resource



1. Role



**3) Elastic File system**

**Description:** EFS creation for providing storage to eks

**Parameters:**

Performance mode: General purpose

Throughput mode: Bursting

Encryption enabled: true

KMS Key: user input

Lifecycle policy : hardcoded

I**ntegration:**

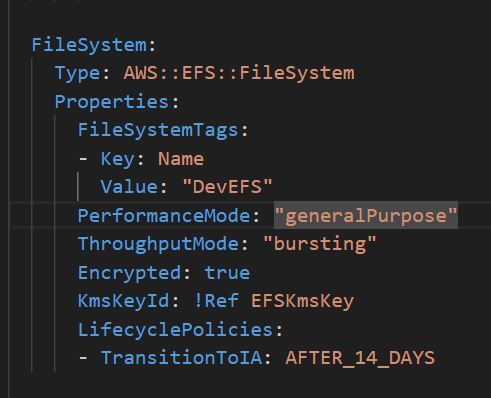
Added two mount targets with reference to the above file system ids and private subnets with separate security group

**Role:**

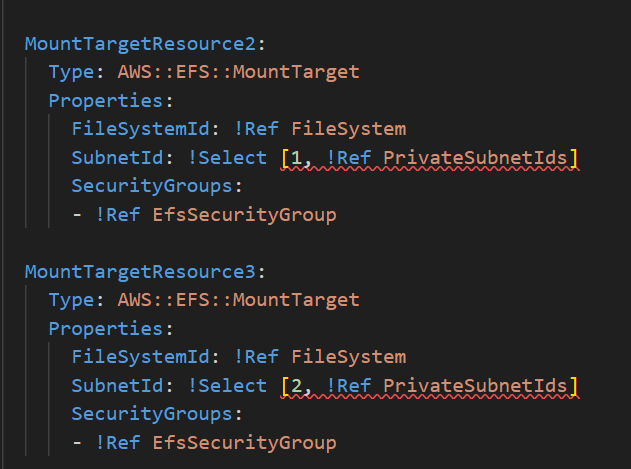
Created a custom role allowing privileges to elasticfilesystem.amazonaws.com service, with managed policies for KMS and Fargate pod execution role policy

**Template:**

1. Resource



1. Mount targets



1. Role



**4) Relational Database Service**

**Description:** Aurora PostgreSQL Database cluster with one instance

**Parameters:**

Engine: Aurora Postgresql

Version: 15.3

Subnet group: Created a subnet group for private subnets

Password: Using a secret from secrets manager

Port: User input

Backup retention period: 7 days

Encryption: enabled

KMS key id : user input

IAM based Database authentication : enabled

Security group : Aurora security group created separately

Class: db.t3.large

Auto minor version upgrade : enabled

Publicly accessible: false

Performance insights: enabled

Enhanced monitoring : configured

Cloudwatch : metrics and logs configured

**Integration:**

Resources related to the secret ( attachment, secrets policy, Lambda function for rotation, Rotation function, permission and schedule )

Policy for allowing RDS database authentication based on IAM roles

Parameter group

Log group ( for cloudwatch )

Log stream

EKS cluster security group added as ingress

**Roles and policies:**

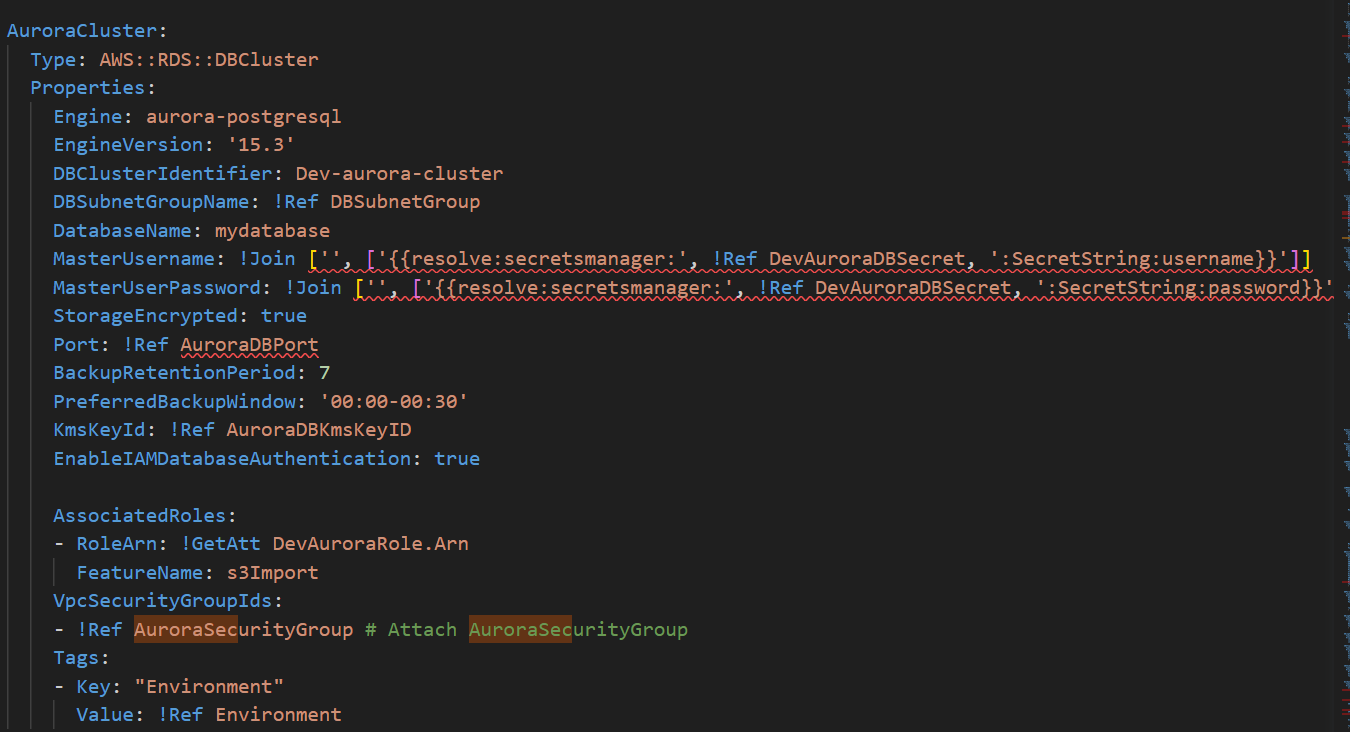
Lambda execution role (for rotating password)

Role and policy for allowing RDS database authentication based on IAM roles

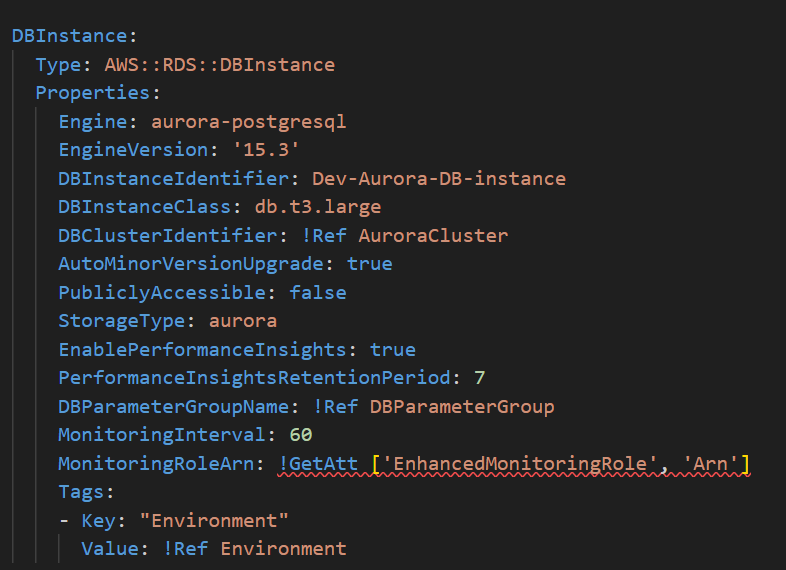
Role for enhanced monitoring

**Template:**

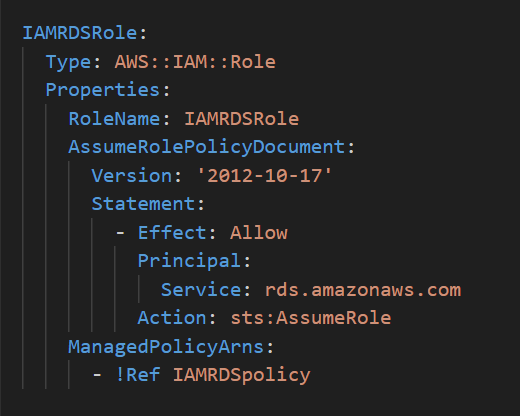
1. Resource



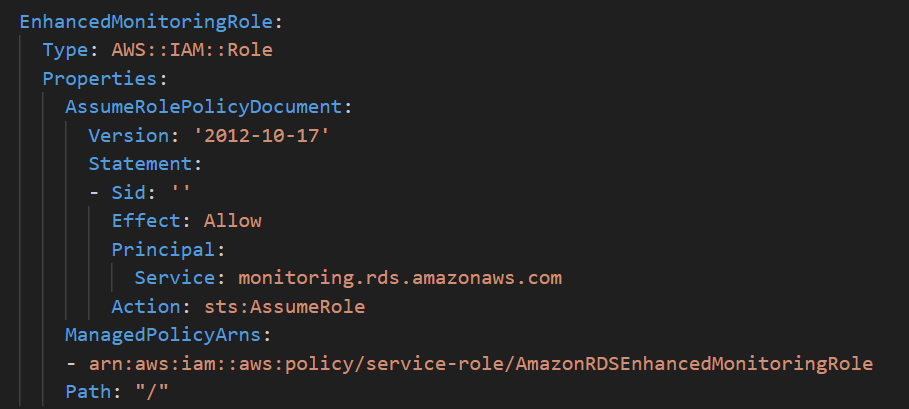
1. Database instance



1. RDS role for IAM authentication



1. Enhanced monitoring role



**5) Opensearch**

**Description:** Opensearch domain with dedicated master nodes disabled and 2 data nodes

**Parameters:**

Instance count : user input

Instance type : user input

Dedicated master: disabled

Multi AZ standby : disabled

EBS : enabled

Subnets: user input ( private )

Encryption at rest: enabled

KMS key id : user input

Logging : application, index slow and search slow logs enabled

**Integration:**

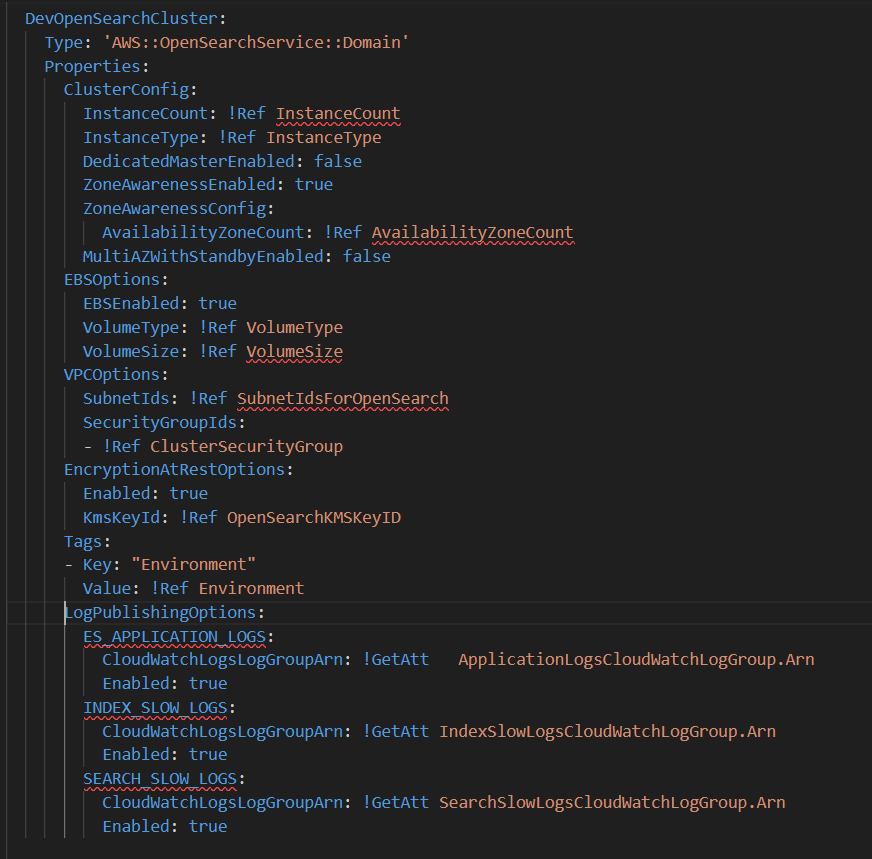
Log groups for application, index and search logs

**Roles and policies:**

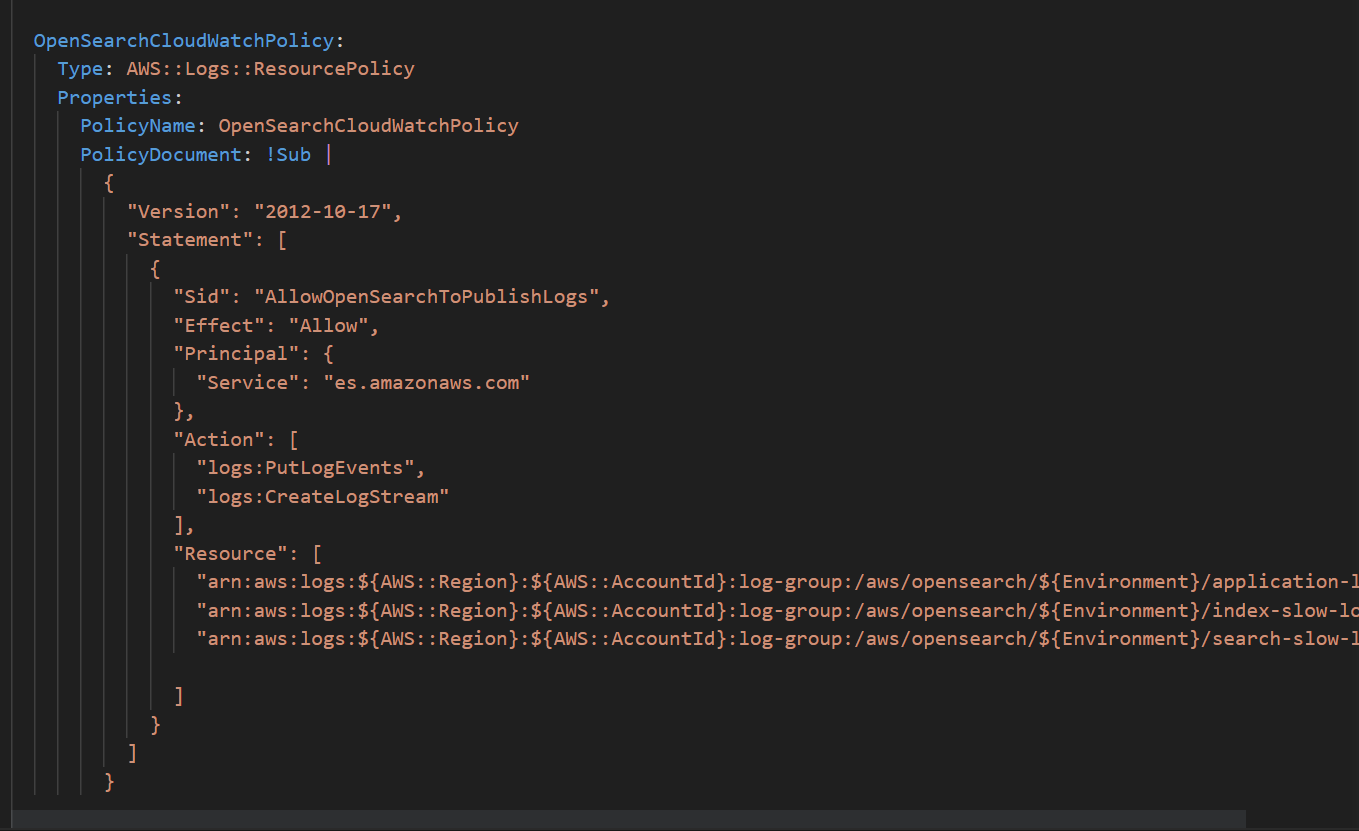
A custom policy created for sending logs to different log groups ( cloudwatch )

**Template:**

1. Resource



1. Policy



**6) Elasticache - Memcached**

**Description:** Memcached cluster with a single node

**Parameters:**

Node type: user input

Parameter group name : user input (default.memcached1.6)

Engine: memcached

Version: 1.6.17

Number of nodes: user input

Subnet group: created

Security group: separate and created

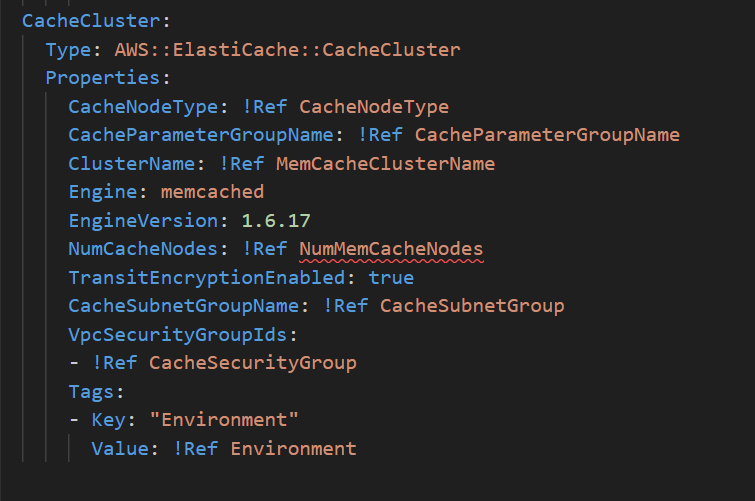
**Integration:**

Subnet group : mandatory parameter for elasticache

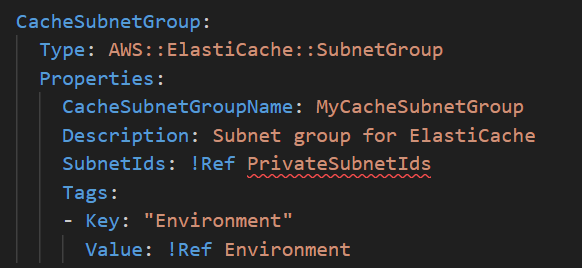
Security group : Port 11211 in ingress and EKS security group

**Template:**

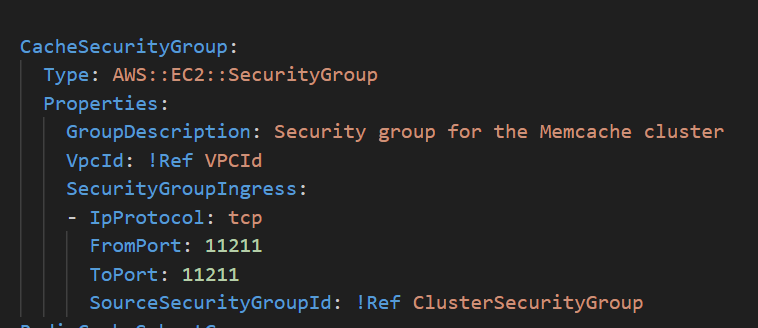
1. Resource



1. Subnet group



1. Security group



**7) Elasticache-Redis**

**Description:** A redis replication group with 1 shard

**Parameters:**

Engine: redis

Cache node type: cache.t4g.micro

No. of shards: 1

Subnet group name: created

Engine : 7.0

AutomaticFailoverEnabled: false

TransitEncryptionEnabled: true

AtRestEncryptionEnabled: true

AutoMinorVersionUpgrade: true

Log configuration : configured slow and engine logs

**Integration:**

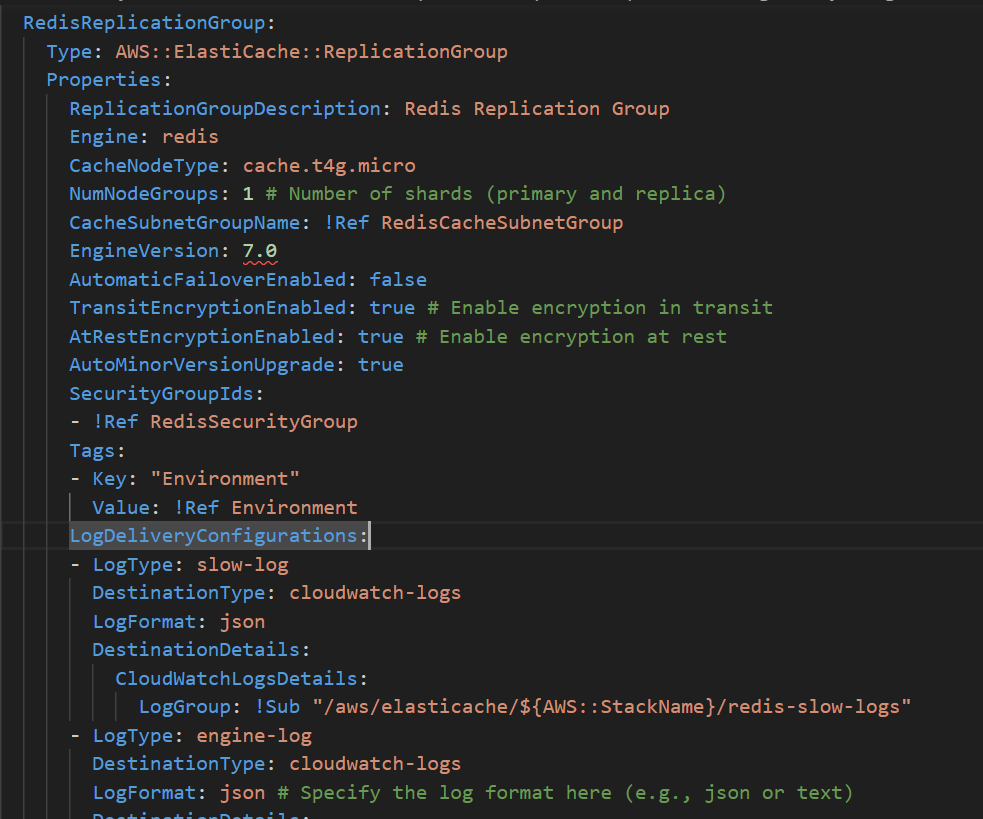
Slow and engine log groups

Subnet group which includes private subnets given through user inputs

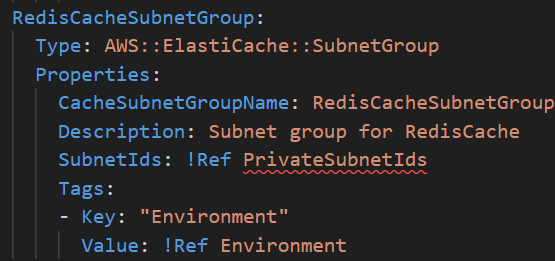
Security group with ingress 6379 and EKS cluster security group

**Template:**

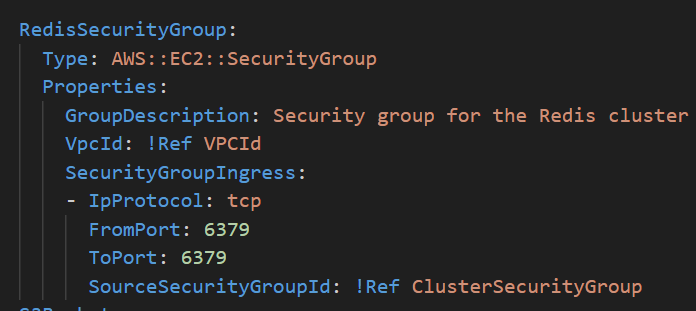
1. Resource



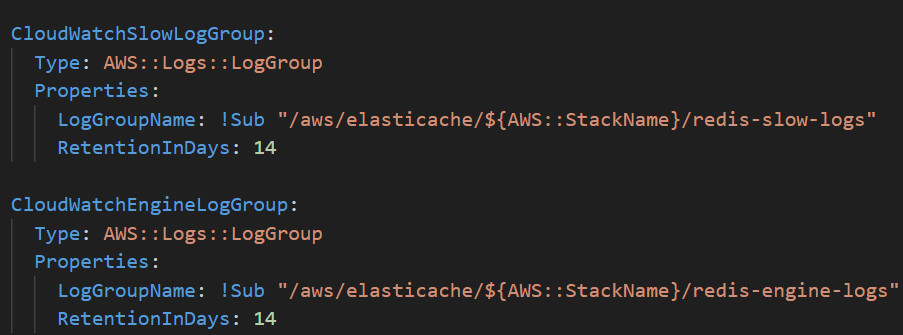
1. Subnet group



1. Security group



d) Log groups

****

**8) S3 bucket**

**Description:** A private S3 bucket with access logging enabled

**Parameters:**

Access control : private

Versioning: enabled

Logging configuration : configured access logs to fall in another s3 bucket

Lifecycle policy : Glacier transition rule with user inputs for deep archive and glacier transitions (number of days)

Bucket encryption: Configured to accept user given kms key id

**Integration:**

Configured another bucket for obtaining access logs of the source bucket

**Roles and policies:**

Bucket policy which only accepts SSL based connections only

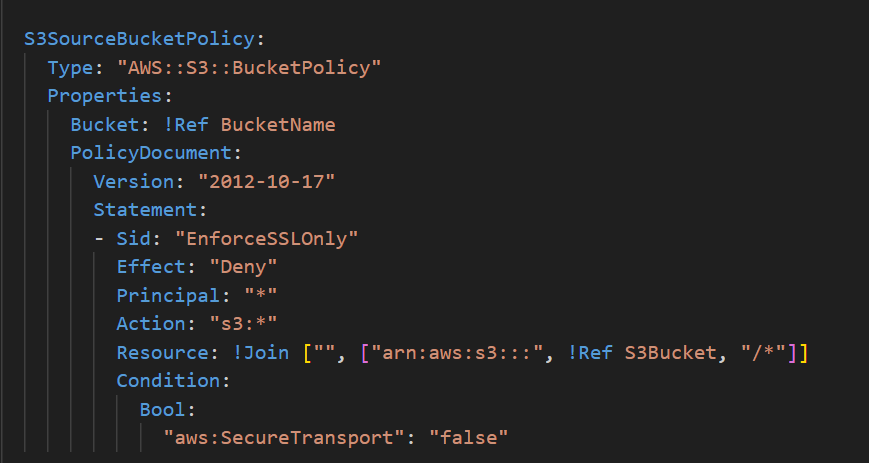
Logging bucket policy

**Template**

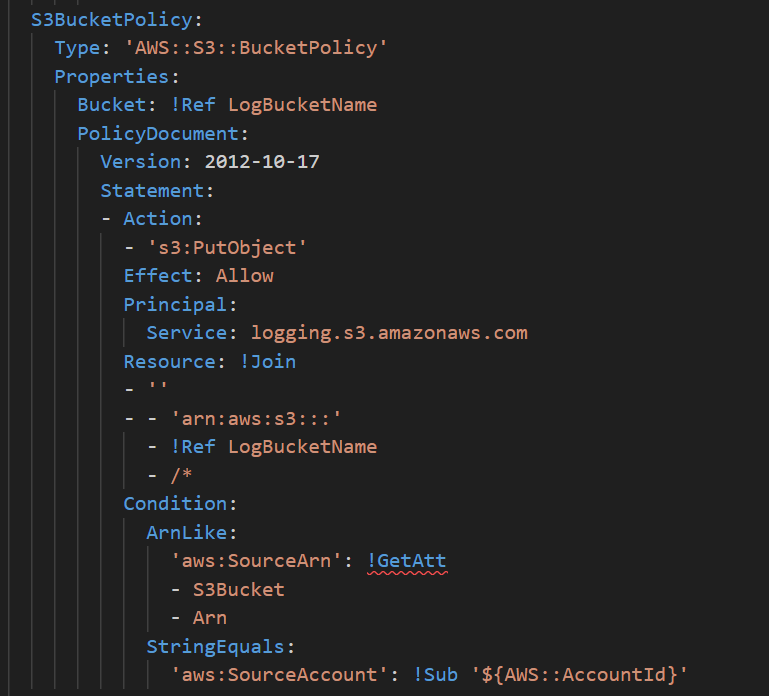
1. Resource



1. Bucket policy



1. Logging bucket policy

****

**9) Backups**

**Description:** Roles, vaults, plans and selection for backing up EFS and RDS

**Parameters:**

1. Backup plans (configured as per requirement)

Rules

Cron expression

1. Backup vault

Encryption key: user input

1. Backup selection: configured target resources (file systems and RDS instances)

**Roles and policies:**

Created a separate role which allows privileges to backup.amazonaws.com service

**Integrations:**

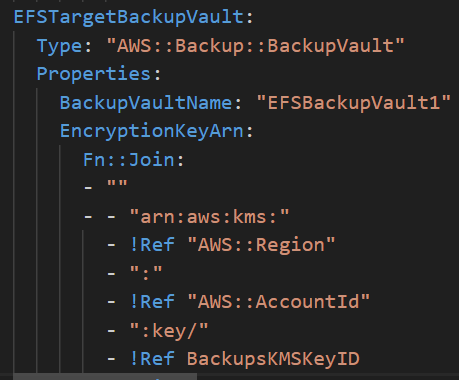
Linking rules to backup plan

Target backup vault in backup plan

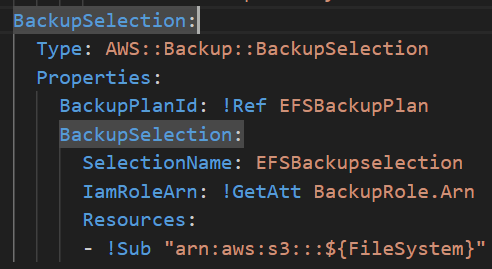
Attaching resources to backup plan using backup selection

**Template:**

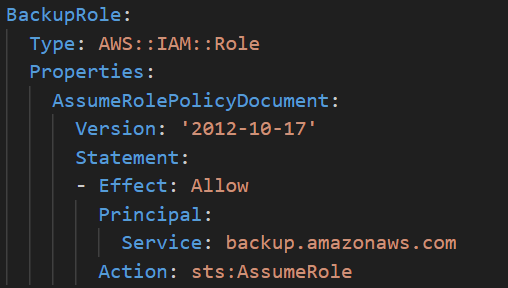
1. Backup vault



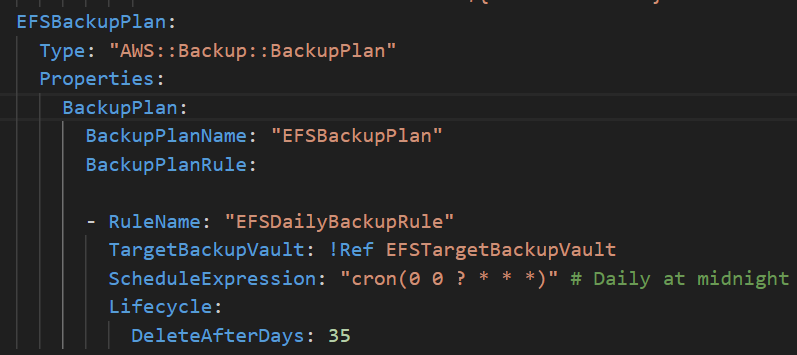
1. Backup selection



1. Role:



1. Plan:



Similar plan, role, selection and vault for Aurora RDS