

AWS Exercise 2

In this exercise we will be building onto our existing yaml "uppgift1.yaml" with a AWS RDS, mySQL, together with a EFS storage for apache2 webservice that then are connected to a autoscaling group for stability and redundancy.

Parameters

we can add following parameters to our script for specific master username and password for AWS RDS

```
MasterUsername:
  Type: String
  Default: root
MasterUserPassword:
  Type: String
  Default: Test123!
```

Resource

First we have a new Secgroup for DB in the case we would like to restrict it for external access.

```
### SEC Group MySQL ###
secGroupNameMySQL:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupName: !Sub '${AWS::StackName}-SecGrpMySQL'
    GroupDescription: 'Allow SSH - Anywhere'
    VpcId: !Ref myVPC
    SecurityGroupIngress:
      - IpProtocol: 'TCP'
        FromPort: 3306
        ToPort: 3306
        CidrIp: 0.0.0.0/0
    Tags:
      - Key: 'Name'
        Value: !Sub '${AWS::StackName}-SecGrpSSH'
    DependsOn: myVPC
### DBSubnetGroup MySQL ###
rdsDBSubnetGroup:
  Type: AWS::RDS::DBSubnetGroup
  Properties:
    DBSubnetGroupDescription: !Sub 'Setup ${AWS::StackName}-SubnetGroup'
    SubnetIds:
      - !Ref SubA
      - !Ref SubB
      - !Ref SubC
```

```
Tags:
  - Key: 'Name'
    Value: !Sub '${AWS::StackName}-SubnetGroup'
DependsOn: routeTableAssocNameC
```

Now we have the config for DB instances. We are going to use MySQL

```
### DB ###
rdsDBInstance:
  Type: AWS::RDS::DBInstance
  Properties:
    AllocatedStorage: '50'
    DBInstanceClass: db.t3.micro
    MultiAZ: true
    BackupRetentionPeriod: 0
    DBInstanceIdentifier: !Sub '${AWS::StackName}-mydbinstance'
    DBName: 'DB'
    DBSubnetGroupName: !Ref rdsDBSubnetGroup
    Engine: mysql
    MasterUsername: !Ref MasterUsername
    MasterUserPassword: !Ref MasterUserPassword
    Port: 3306
    PubliclyAccessible: true
    StorageEncrypted: true
    VPCSecurityGroups:
      - !Ref secGroupNameMySQL
  Tags:
    - Key: 'Name'
      Value: !Sub '${AWS::StackName}-DB-Test'
  DependsOn: MyEFS
```

test27-mydbinstance

Summary

DB identifier test27-mydbinstance	CPU <div><div></div>3.15%</div>	Status <div>Available</div>	Class db.t3.micro
Role Instance	Current activity <div><div></div>0 Connections</div>	Engine MySQL Community	Region & AZ eu-west-1b

Connectivity & security

Monitoring

Logs & events

Configuration

Maintenance & backups

Tags

Connectivity & security

Endpoint & port

Endpoint
test27-mydbinstance.cuyduiszyhzi.eu-west-1.rds.amazonaws.com

Port
3306

Networking

Availability Zone
eu-west-1b

VPC
Test27-VPC (vpc-0ae482f4b9379bd8c)

Subnet group
test27-rdsdbsubnetgroup-r1534b94jtqb

Subnets
subnet-00e1bfeca74be5750
subnet-0664872212a645421
subnet-034a06534fb780bb6

Network type
IPv4

Security

VPC security groups
Test27-SecGrpMySQL (sg-0b97401938487f0fc)
Active

Publicly accessible
Yes

Certificate authority
rds-ca-2019

Certificate authority date
August 22, 2024, 19:08 (UTC+02:00)

DB instance certificate expiration date
August 22, 2024, 19:08 (UTC+02:00)

EFS

We need to add a Security group for our efs so we can restric with requests are allowed to access to the EFS, in this case we have left it open for public access

```
### EFS ###
SecGrpEFS:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupName: !Sub ${AWS::StackName}-EFS
    GroupDescription: "Allow EFS Anywhere"
    VpcId: !Ref myVPC
    SecurityGroupIngress:
      - IpProtocol: 'TCP'
        ToPort: 2049
        FromPort: 2049
        CidrIp: 0.0.0.0/0
    Tags:
      - Key: Name
        Value: !Sub ${AWS::StackName}-EFS
    DependsOn: SubC
```

here is our config for setup a EFS fileshare together witg moutpoint for the 3 diffrent subnet we have created in the region.

```
MyEFS:
  Type: AWS::EFS::FileSystem
  Properties:
    BackupPolicy:
```

```

    Status: DISABLED
    Encrypted: True
    PerformanceMode: generalPurpose
    ThroughputMode: bursting
    FileSystemTags:
      - Key: Name
        Value: !Sub ${AWS::StackName}-EFS
    DependsOn: SecGrpEFS

MyEFSMountSubA:
  Type: AWS::EFS::MountTarget
  Properties:
    FileSystemId: !Ref MyEFS
    SecurityGroups:
      - !Ref SecGrpEFS
      - !Ref secGroupNameSSH
    SubnetId: !Ref SubA
    DependsOn: MyEFS
MyEFSMountSubB:
  Type: AWS::EFS::MountTarget
  Properties:
    FileSystemId: !Ref MyEFS
    SecurityGroups:
      - !Ref SecGrpEFS
      - !Ref secGroupNameSSH
    SubnetId: !Ref SubB
    DependsOn: MyEFSMountSubA
MyEFSMountSubC:
  Type: AWS::EFS::MountTarget
  Properties:
    FileSystemId: !Ref MyEFS
    SecurityGroups:
      - !Ref SecGrpEFS
      - !Ref secGroupNameSSH
    SubnetId: !Ref SubC
    DependsOn: MyEFSMountSubB

```

Test27-EFS (fs-0bd39dc94e08ac76d)

General

Performance mode
General Purpose

Throughput mode
Bursting

Lifecycle management
Transition into IA: None
Transition out of IA: None

Availability zone
Standard

Automatic backups

⊖ Disabled

Encrypted

e16c8f49-298f-46d3-b418-4bcaae72b29d (aws/elasticfilesystem)

File system state

✔ Available

DNS name

fs-0bd39dc94e08ac76d.efs.eu-west-1.amazonaws.com

We will also create a provisioning server for configure our webservice. First we will need to mount our EFS to the instances and install apache2 with php. in this example we have not added any Authentication Unique Keys and Salts to the website.

```

myEC2Instance:
  Type: AWS::EC2::Instance
  Properties:
    KeyName:
      Ref: KeyName
    ImageId: !Ref ImageID
    InstanceType: !Ref ec2type
    Monitoring: true
    SubnetId: !Ref SubA
    SecurityGroupIds:
      - !Ref secGroupNameSSH
      - !Ref secGroupNameHTTP
    UserData:
      Fn::Base64:
        Fn::Sub: |
          #!/bin/bash -ex
          yum update -y
          yum install amazon-efs-utils -y
          mkdir -p /var/www
          echo ${MyEFS.FileSystemId} >> /var/www/test
          mount -t efs -o tls ${MyEFS.FileSystemId}:/ /var/www
          yum install -y httpd wget php-fpm php-mysqli php-json php php-devel
          systemctl start httpd
          systemctl enable httpd
          chown -R ec2-user:apache /var/www
          chmod 2775 /var/www && find /var/www -type d -exec sudo chmod 2775 {}
\;
          find /var/www -type f -exec sudo chmod 0664 {} \;
          yum install -y mariadb105
          wget -P /home/ec2-user/ https://wordpress.org/latest.tar.gz
          tar -xzf /home/ec2-user/latest.tar.gz -C /home/ec2-user/
          cp /home/ec2-user/wordpress/wp-config-sample.php /home/ec2-
user/wordpress/wp-config.php
          sed -i 's/username_here/${MasterUsername}/' /home/ec2-
user/wordpress/wp-config.php
          sed -i 's/password_here/${MasterUserPassword}/' /home/ec2-
user/wordpress/wp-config.php
          sed -i 's/database_name_here/DB/' /home/ec2-user/wordpress/wp-
config.php
          sed -i "s/localhost/${rdsDBInstance.Endpoint.Address}/" /home/ec2-
user/wordpress/wp-config.php
          cp -r /home/ec2-user/wordpress/* /var/www/html/
          service httpd restart
          echo ${rdsDBInstance.Endpoint.Address} >> /var/www/test
    Tags:
      - Key: Name
        Value: !Sub ${AWS::StackName}-EFS
  DependsOn: rdsDBInstance

```

Instances (1/5) Info

Find Instance by attribute or tag (case-sensitive)

Connect

Instance state

Actions

Launch instances

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6 IPs	Monitoring	Security group name
<input type="checkbox"/>	i-007a1ee2fec27c1	Running	t2.micro	2/2 checks passed	No alarms	eu-west-1a	ec2-34-243-181-160.eu...	34.243.181.160	-	-	enabled	Test27-SecGrpHTTPTe
<input checked="" type="checkbox"/>	Test27-EFS	Running	t2.micro	2/2 checks passed	No alarms	eu-west-1a	ec2-34-244-73-75.eu-w...	34.244.73.75	-	-	enabled	Test27-SecGrpHTTPTe
<input type="checkbox"/>	i-0c5f415daa295f60f	Running	t2.micro	2/2 checks passed	No alarms	eu-west-1a	ec2-34-244-73-75.eu-w...	34.244.73.75	-	-	enabled	Test27-SecGrpHTTPTe

Full Code

Full code can be find on github: <https://github.com/Norra-frenzu/AWS/tree/main/Uppgift2>

Resualt

