Automated Disaster Recovery on AWS

Disaster recovery is one the most important aspect of highly available infrastructure. Having a good and automated disaster recovery plan is important for highly available infra structure. To setup automated disaster recovery for EC2 instance and RDS on AWS we can use AWS disaster recovery service. Currently this service support manual disaster recovery of infra to automate this we can use cloudwatch, step function , lambda, SNS AWS services. So in this blog I will explain how to set this up in automated way.

In this setup we will setup active/passive disaster recovery which is cost efficient way to achieve business continuity at the cost of little higher RTO and RPO as compare to active/active setup and also continues data sync is not require in this setup as compare to active/active setup. This automated setup will reduce RTO and RPO of disaster recovery as compare to menual setup for infra on AWS.

Below are the architecture of for disaster recovery

Below are steps for this setup

- Setup 2 VPC in 2 separate regions
- Create instance and RDS database in Primary region
- Setup LB and target group in secondary region
- Initialise AWS Disaster Recovery service in Secondary region
- Enable auto backup for RDS instance and copy this backup in Secondary region

- Setup SNS service to trigger disaster recovery if health status fails using Lambda Function
- Setup Cloudwatch to continuously check health for Instance, DB and application
- Setup step function to orchestrate DR activity to Run post DB launch activity using Lambda
- Verify DR setup is working correctly

Setup 2 VPC in 2 separate regions

1. Create VPC, Subnet, Route table, Nat Gateway, etc. In Primary region

Create instance and RDS database in Primary region

1. Create Ec2 instance. For this demo I am using ubuntu os
2. Create RDS Instance. For this demo I am using Mysql DB not aurora compatible mysql and enable automatic backup with retention (cross region automated backups are supported for RDS Databases. Aurora compatible mysql backups are stored only in the region where the cluster is present) and I have created RDS without encryption.
3. Refer git repository for deploying application on server.

Setup LB and target group in Primary and Secondary region

1. Create Target Group for instance for Backend Service with 8080 Port and HTTP port. Also Setup health check on **/employees** path
2. Create Target Group for instance for Frontend Service with 3000 Port and HTTP port. Setup health check on **/** path.

3. Create Internet Facing Application Loadbalancer in public subnet and setup HTTP listener with default action forward to Frontend target group.
4. Add new rule in HTTP listener for backend and add path based rule for backend target group with path **/employees** and **/employees/*** path condition and forward to backend target group. Set priority 1.
5. We can use HTTPS listener also with SSL but for this demo I am using HTTP listener.
6. Allow 8080 and 3000 Port in Ec2 instance security group form load balancer.
7. And allow 80 port in LB security group.

Initialise AWS Disaster Recovery service in Secondary region

1. Setup Default Replication Server configuration which is used to take snapshot from server in primary region to secondary region. Below are configuration I have kept
   a. Staging area subnet. Select public subnet
   b. Replication server instance type. t3.small
   c. Automatically replicate new disks to replicate newly added disks
   d. EBS volume type (for replicating disks over 125 GiB) Select GP3 SSD
   e. EBS encryption to enable encryption on replicated data select default or custom based on requirement. I have selected default
   f. Create and select security group with allowed 1500 Port from any source for receiving the transferred replicated data.
   g. Select Data routing and throttling for transfering data privately using VPN, DirectConnect, VPC peering or else replication servers will be automatically assigned a public IP. I have disabled this option
   h. Select Throttle network bandwidth (per server - in Mbps) if required I have disable this option.

   i. Select Point in time (PIT) policy retention policy for replicated data as per requirement.
   j. Select **MAP program tagging** for tagging newly added source servers and replication resources I have disabled it
   k. Save this configurations
2. Setup Default launch configuration setting.
   a. **Select Automated launch settings** there are three options available for this. I have selected Active (in-aws) for demo.
      i. Active (basic) - AWS DRS will select the instance type.
      ii. Active (in-aws) - AWS DRS will periodically update the EC2 launch template based on the hardware configuration of the EC2 instance source server from primary region.
      iii. Inactive - AWS DRS will use the instance type configured in your EC2 launch template.
   b. Enable Start instance upon launch
   c. Enable Copy private IP if you want to keep same private IP from source server.
   d. Enable Transfer server tags to copy tag from source servers
   e. Select OS Licensing "**Use AWS provided license".** You can select Bring your own license (BYOL) as per requirement.
   f. Setup **default EC2 launch template for Replica Instance similar to primary region**
      a. Select subnet
      b. Select security group
      c. Select Instance type if You have selected **Inactive** in automated launch settings
      d. Select EBS volume type
      e. Select IAM Instance Profile

    f. Select Auto Assign Public IP for Public server

    g. Select Key Pair

  g. Click on Configure and Initialise

3. Install Replication Agent in source server. Refer Link

   a. Run below command to install replication agent

```
wget -O ./aws-replication-installer-init https://aws-elas

chmod +x aws-replication-installer-init; sudo ./aws-replic
```

   b. Provide AWS Region Name: <Secondary Region>

   c. Select Disk to copy destination region. Press Enter to copy All disk

   d. Wait for 30 Minutes to create and sync source server to secondary region.

   e. After 30 Minutes source server will be visible in secondary region DRS server section.

   f. We can customize default launch configuration for different source server. So it is possible to have different launch configuration for different source server as per requirement.

   g. To customise default launch configuration go to launch setting section in source server in DRS service. Here we can customise Instance profile, key pair, auto assign public IP, Launch template for selecting different subnet and instance type etc. for different source servers

4. **Now DRS service will start copying snapshot from source server to secondary region**

Enable auto backup for RDS instance and copy this backup in Secondary region

1. Go to automatic backups section of RDS

2. Click on Action and select **Manage cross-Region replication**

3. Enable replication in another AWS Region and select secondary region.
4. Select Replication backup retention period
5. Save the configuration

Setup SNS service to trigger disaster recovery if health status fails using Lambda Function

Create SNS topic to run Lambda function which will initiate recovery in secondary region and also create RDS DB.

1. Create standard SNS topic in primary region.
2. Create Lambda Function Name Tigger DRS In secondary region.
3. Add Subscription of Lambda function in SNS
4. Select Protocol Lambda and Add ARN of Lambda in endpoint option
5. Create Subscription
6. Increase timeout for Lambda to 1 Minute
7. Provide below permission to Lambda function
    a. AmazonEC2FullAccess - To create EC2 Instance and attach instance profile and tags to instance
    b. AWSLambdaBasicExecutionRole-ad2e3c64-fcbf-43fa-81bb-7b7e908e247c - Already present
    c. Create Inline policy with below permission

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Allow",
            "Action": [
                "rds:RestoreDBInstanceToPointInTime",
                "rds:AddTagsToResource",
                "iam:PassRole",
                "drs:*",
                "states:StartExecution"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

8. Change parameters of Lambda function as per comments added in code.
9. Also we can add email in subscription to send email if Disaster recovery triggers

Setup Cloudwatch to continuously check health for Instance, DB and application

We can configure to send alert based on multiple parameters like Instance, System or Attached EBS status check fails etc for this demo we are alerting if both backend and frontend service goes down or RDS instance stopes.

1. Create cloudwatch alarm if frontend service goes down.
    a. Create alarm and select metrics
    b. In browse select ApplicationELB
    c. Select **Per AppELB, per TG Metrics** and select HealthyHostCount for frontend target group.
    d. Select average in statistic and period as per requirement Like if you want to alert if service is down for 5 minutes the select 5 Minutes
    e. In condition select Whenever HealthyHostCount is less then 1 Then Alert
    f. Then Give Name and create alarm do not add SNS in Notification.

2. Create cloudwatch alarm if Backend service goes down same as frontend and select backend target group.
3. Create Composite alarm with condition if both backend and frontend alarm triggers then trigger this alarm.
4. Add SNS Topic in Notification section to send notification to SNS topic.
5. Create Eventbridge rule to notify SNS if RDS stops
6. Give name and in Rule type select Rule with an event pattern
7. In Event pattern add below pattern

```
{
  "source": ["aws.rds"],
  "detail-type": ["RDS DB Instance Event"],
  "resources": ["<RDS DB ARN>"],
  "detail": {
    "EventID": ["RDS-EVENT-0087"]
  }
}
```

8. In **Target 1 select SNS topic**
9. Also Configure target input to send input which will come in alert Email
10. Click on configure input transformer
11. In **Target input transformer add “{"instance":"$.resources"}”**
12. In Template Add "Instance <instance> is in Stopped”
13. Create rule

Setup step function to orchestrate DR activity to Run post DB launch activity using Lambda

This Step Function will trigger lambda function which will replace RDS endpoint in application with new RDS and also ADD EC2 instance in Target group.

1. Create Post DB Launch lambda function and provide permission below permission and set timeout.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "elasticloadbalancing:RegisterTargets",
                "rds:DescribeDBInstances",
                "ssm:SendCommand"
            ],
            "Resource": [
                "*"
                ]
        }
    ]
}
```

2. Create step function and select blank template
3. Add wait step in the Workflow and set time needed by RDS DB to be ready.
4. Then Add Lambda Invoke as next step of workflow to trigger Post DB Launch lambda Function and create function.
5. Provide Lambda Invoke permission to step function

Verify DR setup is working correctly

To verify if all automation is working correctly or not stop services running on primary server using below command. You can also test by stoping RDS instance.

```
sudo systemctl stop employee-backend
sudo systemctl stop employee-frontend
```

In current setup we need to manually update route 53 setting to point to secondary region LB but this can be automated using failover routing in route 53 by setting health check.

This automated disaster recovery will help us to achieve RTO and RPO of 20 Minutes. Also we will save cost of running replica of primary region as this is active/passive disaster recovery setup. To rollback to previous region there are multiple ways like in DRS service there is revers replication feature which will start replicating server snapshots in primary region and we can roll back or if there is any issue in server of primary region we can fix those issues and start services in that case we don’t need to do revers replication for server. for RDS we need to take snapshot of RDS in secondary region copy it to primary region and create new RDS with the snapshot. Rollback is manual step