Certainly! Below is a **Standard Operating Procedure (SOP)** for the **Dynamic DNS Management with Route 53** project. This SOP provides a detailed, step-by-step guide for implementing the project, including all necessary configurations, commands, and best practices.

Standard Operating Procedure (SOP): Dynamic DNS Management with Route 53

Objective

To set up a dynamic DNS-based failover solution using AWS services (Route 53, EC2, and S3) that aligns with the AWS Well-Architected Framework's six pillars: **Operational Excellence, Security, Reliability, Performance Efficiency, Cost Optimization, and Sustainability**.

Prerequisites

1. **AWS Account**: Ensure you have an active AWS account with sufficient permissions to create and manage EC2 instances, S3 buckets, and Route 53.
2. **Domain Name**: A registered domain name to use with Route 53.
3. **Basic Knowledge**: Familiarity with AWS services (EC2, S3, Route 53) and basic Linux commands.

Step 1: Set Up EC2 Instances

1.1 Launch Primary EC2 Instance

1. Log in to the **AWS Management Console**.
2. Navigate to **EC2** > **Instances** > **Launch Instances**.
3. Choose an **Amazon Linux 2 AMI**.
4. Select an instance type (e.g., t2.micro for cost optimization).
5. Configure instance details:
    - Ensure the instance is in your desired region.
    - Enable **Auto-assign Public IP**.
6. Add storage (default 8GB is sufficient for this project).
7. Configure security groups:
    - Allow **HTTP (port 80)** and **HTTPS (port 443)** traffic.
    - Allow **SSH (port 22)** from your IP address for secure access.
8. Launch the instance and download the key pair (primary-instance-key.pem).

1.2 Install and Configure Web Server

1. SSH into the EC2 instance:

ssh -i primary-instance-key.pem ec2-user@<public-ip>

2. Update the instance and install Apache:

sudo yum update -y

sudo yum install -y httpd

sudo systemctl start httpd

sudo systemctl enable httpd

3. Create a simple HTML file for the primary website:

echo "<h1>Primary Website</h1>" | sudo tee /var/www/html/index.html

1.3 Launch Secondary EC2 Instance

- Repeat the steps in **1.1** and **1.2** to launch and configure a secondary EC2 instance.
- Use a different HTML file for the secondary website:

echo "<h1>Secondary Website (Maintenance)</h1>" | sudo tee /var/www/html/index.html

---

Step 2: Set Up S3 Bucket for Secondary Site

2.1 Create S3 Bucket

1. Navigate to **S3** > **Create Bucket**.
2. Name the bucket (e.g., secondary-site-bucket).
3. Enable **Static Website Hosting** in the bucket properties:
    - Set the index document to index.html.

2.2 Upload Static Website Content

1. Create an index.html file for the maintenance page:

<h1>Maintenance Mode</h1>

We are currently undergoing maintenance. Please check back later.

Run HTML

2. Upload the file to the S3 bucket.

2.3 Set Bucket Policy for Public Access

1. Go to the bucket's **Permissions** tab.

2. Add the following bucket policy to allow public read access:

```
json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadGetObject",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::secondary-site-bucket/*"
    }
  ]
}
```

Step 3: Configure Route 53

3.1 Create Hosted Zone

1. Navigate to **Route 53** > **Hosted Zones** > **Create Hosted Zone**.
2. Enter your domain name (e.g., example.com).
3. Note the **Name Servers** provided by Route 53 and update your domain registrar's DNS settings.

3.2 Create Health Check

1. Go to **Route 53** > **Health Checks** > **Create Health Check**.
2. Configure the health check to monitor the primary EC2 instance's HTTP endpoint (e.g., http://<primary-ec2-public-ip>).

3.3 Configure Failover Routing Policy

1. Go to **Route 53** > **Hosted Zones** > Select your domain.
2. Create two record sets:
    - **Primary Record**:
        - Name: www.example.com

        - Type: A
        - Alias: No
        - Value: Public IP of the primary EC2 instance.
        - Routing Policy: **Failover** (Primary).
        - Associate Health Check: Select the health check created in **3.2**.
    - **Secondary Record**:
        - Name: www.example.com
        - Type: A
        - Alias: Yes
        - Alias Target: S3 bucket endpoint (e.g., secondary-site-bucket.s3-website-<region>.amazonaws.com).
        - Routing Policy: **Failover** (Secondary).

Step 4: Test Failover

4.1 Simulate Outage

1. Stop the Apache web server on the primary EC2 instance:

```
sudo systemctl stop httpd
```

4.2 Verify Failover

1. Use dig or nslookup to check DNS resolution:

```
dig www.example.com
```

2. Open a browser and navigate to www.example.com. You should see the maintenance page hosted on S3.

Step 5: Best Practices

5.1 Operational Excellence

- Use **CloudWatch Alarms** to monitor EC2 instance health.
- Automate failover testing using AWS Lambda and CloudWatch Events.

5.2 Security

- Use **IAM Roles** for EC2 instances instead of hardcoding credentials.

- Enable **AWS WAF** to protect against web attacks.

5.3 Reliability

- Distribute EC2 instances across multiple Availability Zones.
- Regularly test failover mechanisms.

5.4 Performance Efficiency

- Use **Amazon CloudFront** to cache and deliver content faster.
- Optimize EC2 instance types based on workload.

5.5 Cost Optimization

- Use **S3** for static content to reduce costs.
- Use **Spot Instances** for non-critical workloads.

5.6 Sustainability

- Use **AWS Graviton-based instances** for better energy efficiency.
- Monitor and optimize resource usage.

Step 6: Cleanup

1. Terminate EC2 instances.
2. Delete the S3 bucket.
3. Delete the Route 53 hosted zone and health checks.

Conclusion

This SOP provides a comprehensive guide to implementing a **Dynamic DNS Management with Route 53** project on AWS. By following these steps, you can ensure a robust, secure, and cost-effective solution that aligns with AWS best practices.