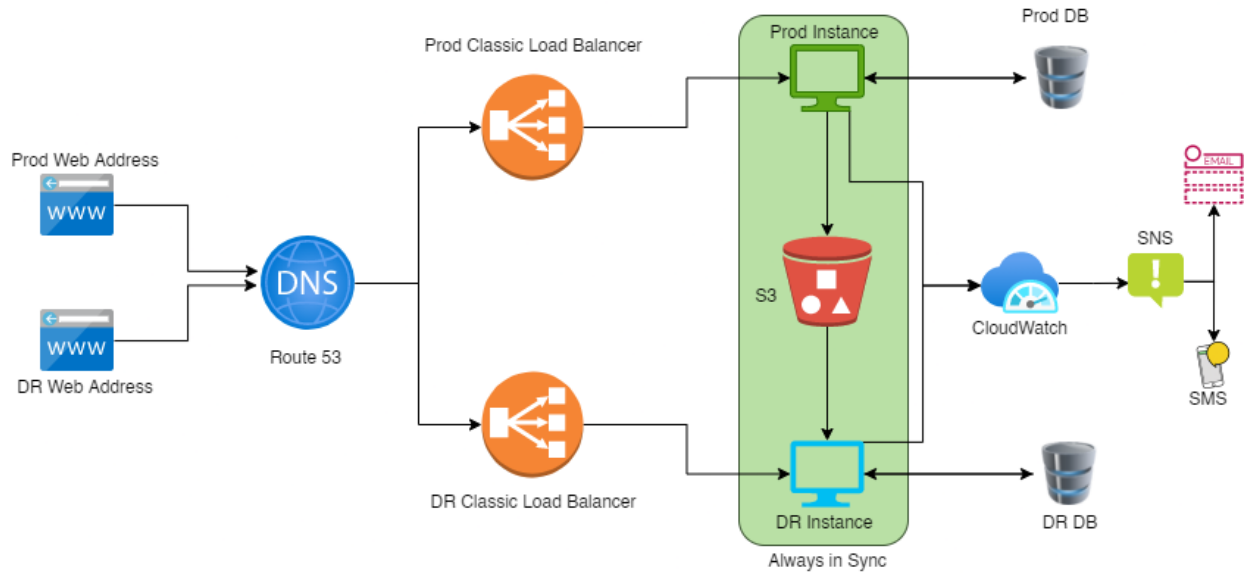
AWS Web Hosting Procedure

Author: Nandagopalan V. (Nandagopalan.ae@gmail.com)

System Architecture



AWS Modules Used:

ACM, R53, ELB, EC2, IAM, RDS, S3, CloudWatch,SNS

Prerequisites:

1. Create the IAM and Security groups needed.
    a. **IAM : EC2 administrator access - Full access**
        i. https://console.aws.amazon.com/iamv2/home?#/roles
        ii. Create Role → AWS services → EC2
        iii. Add Permissions → Administrator Access
        iv. Add Tags if needed.
        v. Enter Role name → Create Role
    b. **VPC**
        i. Create a default VPC (https://ap-south-1.console.aws.amazon.com/vpc/home#CreateDefaultVpc:)
    c. **Security Group**
        i. EC2 → Security Groups → Create Security Group
        ii. Enter name and description
        iii. Select the VPC created in previous step
        iv. Inbound Rules: “All TCP” “0.0.0.0/0”
        v. Create Security Group.

Create Databases

1. RDS → Create Database → Standard Create
2. MySQL
3. Free-tier → Choose version
4. Settings:
    a. Set DB Identifier (Proddb)
    b. Credentials (prodAdm, prodAdm123)
5. Select VPC created and select the Security group (All Access TCP) created in above steps.
6. Enter initial DB name (*)
7. Leave the rest in default options

Repeat the above steps and create a second DB (Dr-DB).

Create EC2 Instances

1. Create 2 EC2 instances with Ubuntu AMI
2. Tag the security group and IAM to both in configuration
3. Insert the following in User data (for Apache2, PHP and MySQL)

```
#!/bin/bash
apt-get install python-software-properties -y
add-apt-repository ppa:ondrej/php -y
apt update -y
apt install -y php7.2
apt install php7.2-curl php7.2-gd php7.2-json php7.2-mbstring php7.2-mcrypt -y
apt install apache2 libapache2-mod-php7.2 -y
apt install mysql-server php7.2-mysql -y
systemctl restart apache2.service
systemctl restart mysql.service
cd /var/www/html
echo "healthy" > healthy.html
wget https://wordpress.org/latest.tar.gz
tar -xzf latest.tar.gz
cp -r wordpress/* /var/www/html/
rm -rf wordpress
rm -rf latest.tar.gz
chmod -R 755 wp-content
chown -R apache:apache wp-content
wget https://s3.amazonaws.com/bucketforwordpresslab-donotdelete/htaccess.txt
mv htaccess.txt .htaccess
mv index.html index1.html
```

4. Hit the public address of the prod instance (use http://) and get the WP DB config. Page.
5. Enter DB_Name(not to be confused with DB_Identifier) and credentials, use the IP of the corresponding DB created in RDS.
6. Copy the generated wp-config.php file to the EC2 instance /var/www/html
7. Repeat 4-6 in the DR instance too.

Create Load Balancers

1. Create 2 CLB (one for Prod and one for DR)
2. Tag the security group and appropriate EC2 instances to each CLB.
3. Try hitting the ip of each CLB and make sure you land on the appropriate pages.

Author: Nandagopalan V.
Email: Nandagopalan.ae@gmail.com

4. Get the SSL certificates and revisit.
5. Enter additional listeners in each CLB for https and tag the ACM certificate.

Get SSL Certificates

1. Amazon Certificate Manager
2. Provision Certificate → Public Certificate
3. Add the domains (deltav.in and *.deltav.in)
4. Select Validate through DNS
5. Continue with R53 Configuration and then revisit
6. Create Record for each domain in ACM

Route 53 Configuration

1. R53 → Create Hosted Zone
2. Enter the domain address and create the hosted zone
3. Copy the Nameservers to the Domain Manager(namecheap)
4. Get back to ACM and create a Record for the domains.
5. Reload and confirm CNAME is created for the additional domains created in ACM.
6. Map CLB to domains:
    a. Click Create Record
    b. Choose the domain address
    c. Select Alias to be turned ON
    d. Select Network Load Balancer
    e. Choose the Region
    f. Paste the corresponding CLB endpoint IP
    g. Create Record
7. Hit the page using the DNS name and make sure you land on appropriate pages.

Synchronization: (Using S3 Bucket)

1. Create a standard S3 bucket (2 if separate buckets are needed for Code and Media)
2. In Ubuntu - Ensure to enable Cron and aws-cli is installed
    a. ***sudo systemctl enable cron***
    b. ***sudo apt-get install aws-cli***
3. Use the following to enter a crontab entry in the Prod machine (Push Data)
4. Crontab -e
    a. ****/2 * * * * aws s3 sync --delete /var/www/html/wp-content/uploads s3://BucketName***
    b. ****/2 * * * * aws s3 sync --delete /var/www/html/ s3://BucketName***

5. Use the following to enter a crontab entry in the DR machine (Pull Data)
    a. ****/2 * * * * aws s3 sync --delete s3://BucketName /var/www/html/wp-content/uploads***
    b. ****/2 * * * * aws s3 sync --delete s3://BucketName /var/www/html/***

Cloud-Watch: (Monitoring)

1. Create a dashboard in Cloud-Watch
2. Select the appropriate Widget

Author: Nandagopalan V.
Email: Nandagopalan.ae@gmail.com

3. Select the instance id and select the required metrics from each instance for each widget.
4. Incase of additional parameters (Install the MON scripts)
    a. https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/mon-scripts.html

SNS: (Monitoring)

1. In SNS - Create a Standard Topic and configure the required endpoints(email n Mobile) in subscriptions.
2. To Receive Push Notifications - Enable Alarms in Cloudwatch
3. In alarms → Select the Metrics to watch out and select the Threshold values
4. Under SNS → Select the created topic and select Create alarm.

Author: Nandagopalan V.
Email: Nandagopalan.ae@gmail.com