AWS CLOUD MIGRATION

This project involved the cloud migration of an application and its database from on-premises environment to AWS. The overall strategy employed a pragmatic mix of two migration approaches: Rehost (Lift-and-Shift) and Replatform. The Application Server was migrated via Rehost using AWS Application Migration Service (MGN), prioritizing the fastest possible path to the cloud with minimal changes to the operating system or application code. Conversely, the critical Database component was migrated via Replatform to Amazon RDS using AWS Database Migration Service (DMS). This Replatform strategy involved intentionally changing the underlying architecture to leverage a fully managed cloud service, immediately gaining crucial benefits like automated patching, high availability (Multi-AZ), and reduced operational overhead, which are essential for a mission-critical data store. This combined approach successfully balanced the speed of migration for the application layer with the necessity of obtaining key cloud resiliency and management features for the data layer.

The project was executed in **October**, and it is worth noting that the **Application Discovery Service (ADS)** component used in the assessment phase was subsequently deprecated by AWS, effective **November 7th**, in favor of AWS Transform.

The following is a comprehensive list of all completed tasks and the components created during each phase of this migration project.

1. Target Environment Setup (Prerequisites)

This phase set up the required AWS networking infrastructure using a CloudFormation template.

VPC and Subnets

- Created the destination **VPC** (TargetVPC, CIDR 10.0.0.0/16).
- Created two **Public Subnets** (PublicSubnet0, PublicSubnet1).
- Created two **Private Web Subnets** (PrivateSubnet00, PrivateSubnet10).
- Created two **Private DB Subnets** (PrivateSubnet01, PrivateSubnet11).

Connectivity and Routing

- Created an **Internet Gateway (IGW)**.
- Created two **NAT Gateways** (NATGateway0, NATGateway1) with dedicated **Elastic IPs (EIPs)**.
- Created a **Public Route Table** and two **Private Route Tables**.
- Created a **Public Network ACL** (PublicNetworkAcl).

Source Simulation

- Created **simulated source servers** and **IAM components**.

2. Assessment / Application Discovery

This phase used AWS Application Discovery Service (ADS) to gather data on the source environment.

Configuration

- Initialized and configured **AWS Application Discovery Service (ADS)**.
- Set the **Migration Hub home Region**.
- Created **IAM user and credentials** for discovery.

Data Collection

- Deployed the **ADS Agent** to the source **Webserver** and **DBServer**.
- Verified that **data collection started** and was in a healthy condition.

Review and Grouping

- Reviewed collected data (Server list, Network visualization, etc.).
- Created an **application group** and managed server membership.

3. Database Migration (Replatform to Amazon RDS using DMS)

This phase replatformed the self-managed database to a fully managed **Amazon RDS** instance using **AWS Database Migration Service (DMS)** for continuous replication.

Target RDS Setup

- Created a new managed database using **Amazon RDS**.
- Created a **DB Subnet Group**.
- Created **two Security Groups (SG)** for the DB.

DMS Configuration

- Created a **DMS Replication Subnet Group**.
- Created the **AWS DMS Replication Instance**.
- Created the **source DMS Endpoint**.
- Created the **target DMS Endpoint**.

Source Prep for CDC

- Added the replication instance to the inbound rule for the **DBServerAG Security Group**.
- Granted REPLICATION CLIENT, REPLICATION SLAVE, and SUPER privileges to the source database user.
- Created the directory for MySQL binary logs (/var/lib/mysql/binlogs).
- Modified MySQL configuration (/etc/mysql/my.cnf) to **enable Binary Log** (log-bin, binlog_format=ROW, server_id=1, etc.).

Replication

- Created and ran a **DMS Replication Task**.
- Monitored the task until it reached **Load complete, replication ongoing** status.

4. Application Migration (Rehost using AWS MGN)

This phase used AWS Application Migration Service (MGN) to perform a "lift-and-shift" rehost of the application server (Webserver).

MGN Setup

- Initialized and configured **AWS Application Migration Service (MGN)**.
- Deployed the **MGN Agent** to the source application server.

Target Configuration

- Customized the **EC2 Launch Template** for the rehosted server.
- Disabled the **Instance type right sizing** option in Launch settings.
- Installed and configured the **AWS CloudWatch agent** to publish memory and swap utilization metrics.

Testing and Cutover

- Launched a **test instance** with MGN for validation.
- Marked the webserver as **'ready for cutover'**.
- Launched the final **cutover instance**.

Post-Cutover Reconfiguration

- Modified the application configuration (wp-config.php) to use the **new RDS endpoint** and credentials.
- Updated site configuration variables (WP_SITEURL and WP_HOME) to the **new public hostname**.
- Updated the target database's **Security Group (DB-SG) inbound rules** to allow traffic from the cutover EC2 instance.
- Verified the site was up and running.

Finalization

- **Finalized the cutover** process on MGN.
- Executed **housekeeping** on the MGN inventory.