Azure to AWS Migration Implementation for Get1Page

aws partner network

Executive Summary

Introduction

Get1Page is a California-based startup offering Software-as-a-Service (SAAS) solutions for efficient meeting management. Their flagship product, "1Page," enhances user experiences by providing valuable information before each scheduled meeting, leading to better connections and productive outcomes. CloudThat's solution for Get1Page involved scaling the Meeting Management application, optimizing API deployments, implementing monitoring practices, and migrating the Azure SQL Database to Amazon RDS for improved performance, scalability, and cost-efficiency.

Customer Challenges

Get1Page, an Azure cloud client, encountered multiple challenges in scaling their Meeting Management application, API deployment management, performance monitoring, and cost control. Their Azure cloud website, comprising 37 services integrated with multiple APIs and hosted on 2TB of storage across 4 Azure SQL databases, posed management difficulties. High traffic periods resulted in slow page load times, unresponsive pages, and potential crashes, leading to revenue loss. Latency issues between the datastore and API services caused delayed responses and a suboptimal user experience. Lack of proper monitoring hindered performance optimization and efficient operations. Additionally, the client had difficulties during the site outage as they couldn't perform checks or gather enough information to debug the issue. The absence of monitoring tools made it hard to determine the cause and timing consequently experienced a period of uncertainty.

Solution

The successful migration of the customer's environment from Azure to AWS cloud was achieved through a three-step migration process as defined below:

API Migration from Azure App Service to Amazon EKS Cluster:

- IAM roles ensure secure access to AWS resources.
- VPC, subnets, NAT, and internet gateways are set up. Terraform automates resource provisioning.
- Horizontal Pod Autoscaler and ALB Ingress Controller improve scalability. Bitbucket and Docker aid in building and tagging APIs.
- Security measures include runtime and API calls, AWS services for access management, encryption of EBS volumes, and security groups.

SQL Server Migration from Azure to AWS:

- Amazon RDS SQL instance deployed in private subnet with multi-AZ for high availability.
- Bastion host ensures secure RDS access. SQL dumper and S3 facilitate backup and restore.
- Amazon Elasticache for Redis reduces latency. Backup and restore policy configured.

Migration of Static Website to Amazon EC2 instance:

- EC2 instance in private subnet with Bastion host for access. Application Load Balancer exposes the instance, and Route53 directs traffic.

About Get1Page

Get1Page is a dynamic startup with its operations base in California, United States. The company specializes in providing Software-as-a-Service (SAAS) solutions for businesses to manage meetings efficiently. Their flagship product is 1page, a cutting-edge SAAS platform fully compatible with multiple operating systems to maximize market reach and accommodate user preferences for seamless accessibility and user satisfaction.

Observability:

- Amazon CloudWatch monitors AWS resources, while CloudTrail tracks user and API activity for auditing and compliance.

Architecture Diagram

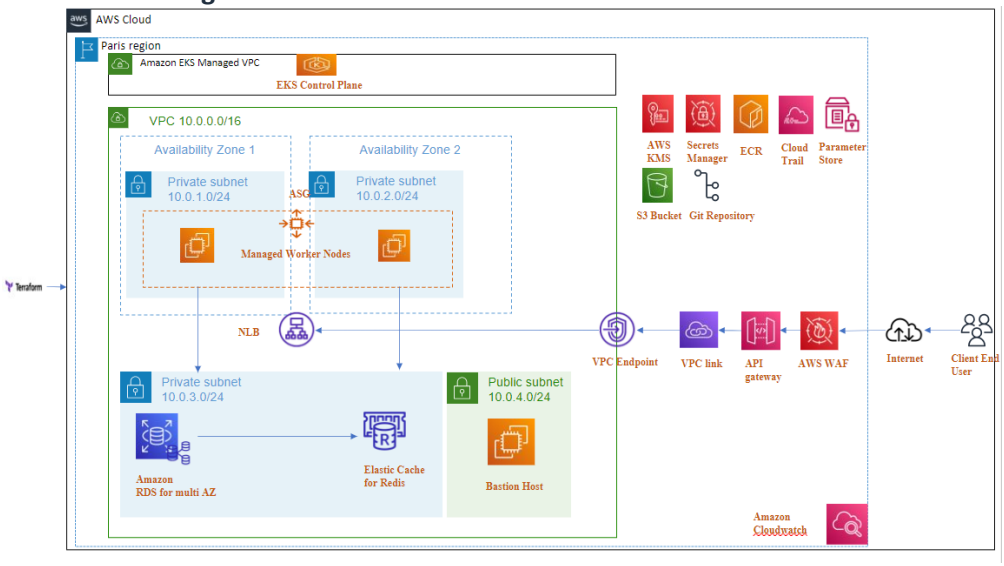
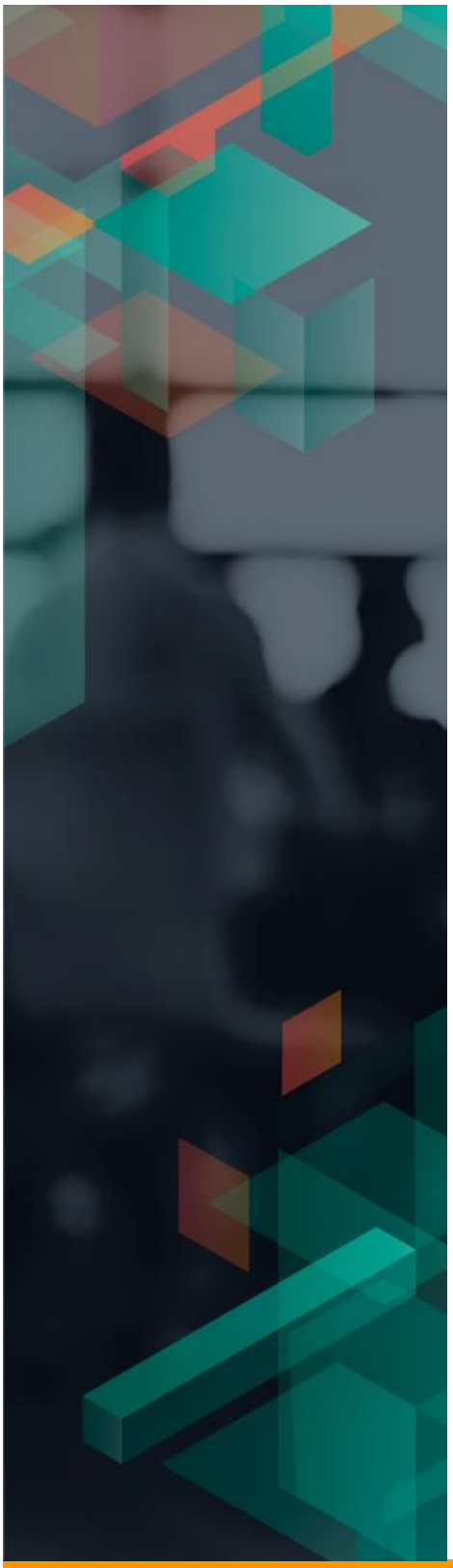


Figure 1: APIs being hosted on EKS Cluster

The architecture depicted above demonstrates the deployment of an Elastic Kubernetes Service (EKS) cluster, along with Amazon RDS-SQL and Elastic Cache, within private subnets to enforce restricted access. In this setup, microservices are deployed as APIs, and their accessibility is controlled through an API gateway that seamlessly integrates with AWS Web Application Firewall (WAF) for heightened security measures. By employing this architecture, a scalable and secure environment is established for deploying and effectively managing containerized microservices within the EKS cluster.



Conclusion:

• The deployment of microservices-based APIs provided the advantage of scalability and flexibility. It has allowed individual APIs to be modified without affecting the other APIs.

• The overall cost of implementing the solution on AWS cloud ($5700 MRR) is 35% lower as compared to while on Azure.

• The introduction of caching layer improved the performance of the read-heavy application by reducing 100% latency and optimizing throughput by 80%.

• Implementing a CI/CD process into the migration plan automated the application deployment cycle. Additionally, the templatization of infrastructure provisioning helped in faster recovery from infrastructure failovers.

• Infrastructure and application observability helped analyze traffic patterns, resource utilization, application performance, scaling, resource health, and capacity planning for future business growth.

• The application works more efficiently and effectively, providing greater flexibility and agility in deployment and scaling.

• The application runs in a more isolated and secure environment, reducing the risk of conflicts and vulnerabilities. The containerization has also made managing the application's resources easier, with better control over CPU, memory, and network usage.

• The application's enhanced performance, scalability, and security have contributed to an overall improved user experience, ultimately boosting business growth and revenue.

About CloudThat

CloudThat is the official AWS (Amazon Web Services) Advanced Consulting Partner, AWS DevOps Competency Partner, AWS DevOps Competency Partner, Amazon EKS Service Delivery Partner, Amazon QuickSight Service Delivery Partner, helping people develop knowledge of the cloud and help their businesses aim for higher goals using best-in-industry cloud computing practices and expertise. We are on a mission to build a robust cloud computing ecosystem by disseminating knowledge on technological intricacies within the cloud space.

cloudthat
move up