applied cloud computing

DR Setup on AWS Cloud

TABLE OF CONTENTS

Overview of our client

The client is a new-age lending platform that is poised to revolutionize the lending industry. Their suite of customer-centric, scalable products is engineered to make the best use of AI & ML and transform the end-to-end processes of the lending business. It offers solutions and services that offer guaranteed reliability and minimal turnaround time with an intelligent, domain-native platform that is created and developed from the ground up to provide financial institutions with the lending environment of today. The client is the future of smart, seamless, and secure digital lending, backed by full-stack, 100 percent digital, new-age goods and solutions. Their end goal is to help banks create an entire ecosystem where quick, friction-less transactions are a part of client experience.

AWS

Amazon Web Services, Inc. (AWS) is an Amazon subsidiary that offers metered pay-as-you-go cloud computing platforms and APIs to consumers, businesses, and governments. Cloud computing is the on-demand, pay-as-you-go distribution of IT services over the Internet instead of purchasing, operating, and maintaining physical data centers and servers, one can rent computing power, storage, and databases from a cloud provider like Amazon Web Services on an as-needed basis (AWS). These web services for cloud computing include a variety of basic abstract technical infrastructure and distributed computing building blocks and utilities. Virtual computers on AWS have hardware central processing units (CPUs) and graphics processing units (GPUs) for processing, local/RAM memory, hard-disk/SSD storage, a choice of operating systems, networking, and pre-loaded software applications like web servers, databases, and customer relationship management (CRM).

A deep dive into Terraform

Terraform is an open-source tool that utilizes declarative configuration files to provision Cloud services including virtual machines, containers, storage, and networking. Terraforms infrastructure-as-code (IAC) approach helps you follow DevOps best practices for change management by allowing you to manage Terraform configuration files in source control to keep your testing and production environments in top shape. Through its application programming interfaces, Terraform develops and manages resources on cloud platforms and other services (APIs). Terraform can work with almost any platform or service that has an API that can be accessed using providers. Terraform users employ HCL, a JSON-like configuration language, to specify and enforce infrastructure configurations. HCL's straightforward syntax enables provisioning and re-provisioning infrastructure across different cloud and on-premises data centers simple for DevOps teams.

It helps with most of the major functions to provision resources from any infrastructure provider, compose infrastructure as code in a Terraform file using HCL. Infrastructure automation workflows are used by IT operations and development teams to compose, collaborate, reuse, and provision infrastructure as code. Infrastructure automation process with access controls, policy enforcement, and audit, as well as security, compliance, and cost management features.

It basically means to install Terraform on your machine and write Terraform manifests. When you run the terraform apply command, it creates the infrastructurein the cloud according to your manifest. Since Terraform is mainly used to create new resources, its manifest files define, for example, the type of virtual machines and how many of them should be deployed. After Terraform is done provisioning, it saves the details of the provisioned resources (for example, IP addresses of created virtual machines) back into the state file.

The need to use Ansible

Ansible is a software application that automates cross-platform computer assistance in a simple but effective way. It's primarily aimed at IT pros, who use it for application deployment, workstation and server updates, cloud provisioning, configuration management, intra-service orchestration, and practically anything else a systems administrator performs on a weekly or daily basis. Ansible is simple to set up because it doesn't require any agent software and doesn't require any additional security architecture. Ansible relies on instructions to do each job because it is all about automation. It's straightforward to keep track of versions when everything is put down in script form. The practical result is a significant contribution to the IT "infrastructure as code" movement: the idea that server and client infrastructure maintenance can and should be treated in the same way as software development, with self-documenting, proven, and executable solutions capable of running an organization regardless of staff changes. There are two types of computers in Ansible: control nodes and managed nodes. A control node is an Ansible-running machine. There must be at least one control node, with the option of a backup control node. Any device that is handled by the control node is referred to as a managed node. Ansible works by connecting to nodes (clients, servers, or whatever you're configuring) on a network, and then sending a small program called an Ansible module to that node. Ansible executes these modules over SSH and removes them when finished. The only requirement for this interaction is that your Ansible control node has login access to the managed nodes. SSH keys are the most common way to provide access, but other forms of authentication are also supported.
Terraform is designed to provide different infrastructure components.
Ansible is a configuration-management and application-deployment tool. It means that you'll use Terraform first to create, for example, a virtual machine and then use Ansible to install necessary applications on that machine

The need for Integrating Terraform and Ansible

In the area of DevOps, businesses use IAC to implement business processes (Infrastructure as Code) the goal of Infrastructure as a Service is to make large-scale infrastructure management easier. Modern IAC tools make configuration easier, allowing you to swiftly handle server issues. Terraform and Ansible is two important IAC solutions that make it simple for businesses to create and scale configurations; both tools are used to deploy sophisticated programs as advanced-level platforms. Both have varied use cases; so let's compare and contrast Terraform vs Ansible, a debate that's raging in the DevOps world right now.

They each have a distinct function and complement one another. Terraform is a tool that allows you to provide various infrastructure components. Ansible is a tool for managing configurations and deploying applications. It means you'll use Terraform to create a virtual machine, for example, and then use Ansible to install the required applications on that system.
However, by default, these two tools are independent. One must link Terraform-managed nodes with Ansible control nodes in order for them to work together. Because they address different aspects of infrastructure and software deployment, Ansible and Terraform are not competitors.

In order to make these two work together, we need to somehow pass the information about infrastructure created by Terraform into Ansible, or vice versa. More specifically, we have two options: either use Terraforms output as input for Ansible’s inventory or instruct Terraform to execute Ansible. The idea of "layers" is one of the most essential things to comprehend when comparing Ansible to Terraform.

The need for Disaster recovery

The most precious asset of modern-day corporations is historical data . Its loss can cause lasting damage to your company, such as lost productivity, income, reputation, and even customers. It's difficult to foresee when a disaster will strike and how severe its consequences will be. What you can control, though, is how you respond to a tragedy and how well your business recovers from it. The majority of traditional disaster recovery issues can be effectively addressed by disaster recovery in cloud computing.

Disaster recovery's main purpose is to reduce the total impact of a disaster on business performance. Cloud computing enables disaster recovery. Critical workloads can be fail overed to a DR site in the event of a disaster, allowing business operations to restart. You can fail back from the cloud and restore your infrastructure and its components to their original condition as soon as your production data center is restored. As a result, service disruption is minimized, and company downtime is decreased.

The advantages of DR are:

- To support critical operations, one doesn't need to construct a second physical site or purchase additional hardware and software.
- One can use cloud storage as a secondary disaster recovery site with disaster recovery in cloud computing.
- One can easily scale up or down your cloud computing resources depending on your current business demands.
- You only have to pay for the cloud computing services you actually use, thanks to its affordable pay-as-you-go pricing model.

How our ACC team helped them.

As part of this engagement Applied Cloud Computing Pvt Ltd (ACC) would be helping the client with the process of migrating their current backend IT setup from on-premises to AWS cloud. The client was hosting their applications on an on-premises data center. The applications needed to be migrated to cloud. They also wanted to make a DR(disaster recovery) system as they had live data on premise and hence we had to set up an AWS account as per the AWS account best practices and identify non overlapping network IP range and create VPC structure which will support the existing architecture.

The client had a request to integrate both ansible and terraform to make an appropriate infrastructure and all the functionalities should work as it is and all the security standards should be followed, We would be creating resources and setting up the infrastructure considering AWS security best practices and following well-architected best practices such as hardening of servers and hardening of AWS account as per the CIS guideline.

The client did not wish to use managed services hence for database no managed services were used basically managed cloud services are services that provide a client with partial or full management of their cloud resources or infrastructure and everything was on server as every server level configuration was customized as per clients requirements and our team assisted client's team to integrate their CI-CD(continuous integration and continuous deployment) pipeline with the proposed setup assisted the team in ingress controller, API gateway configuration ,domain mapping and during Load Testing. The agent software installation was achieved through ansible. We led the project from implementation perspective and contributed towards making the system compliant as per the clients security standards.

ABOUT ACC

ACC is an AWS Advance Partner with AWS Mobility Competency. Awarded The Best BFSI industry Consulting Partner for the year 2019, ACC has had several successful cloud migration and application development projects to its credit.

Our business offerings include Digitalisation, Cloud Services, Product Engineering, Big Data & Analytics and Cloud Security. ACC has developed several products to its credit. These include Ottohm – Enterprise Video and OTT Platform, Atlas API – API Management and Development Platform, Atlas CLM – Cloud Life Cycle Management, Atlas HCM – HR Digital Onboarding and Employee Management, Atlas ITSM – Vendor Onboarding and Service Management and Smart Contracts – Contract Automation and Management.

www.appliedcloudcomputing.com

Shubho Pramanik

(+91) 90297 20294

shubho@acc.ltd

Rogin Rappai

(+91) 8828478321

rogin.rappai@acc.ltd

also reach us at