MOHAN. G



asha@fugetec.com



* **+1 469-639-0804**

in/mohan-g-475b07267 ***Sr. CLOUD/DEVOPS ENGINEER***



Proficient Senior Cloud DevOps Engineer with 9+ years of IT Experience extensively as Cloud/ DevOps Engineer, Build & Release Engineer. I can implement effective IT strategies and hands-on experience supporting, automating, and optimizing mission critical deployments in Microsoft Azure, Amazon Web Services, Google Cloud Platform leveraging configuration management, CI/CD, Kubernetes, Docker, and DevOps processes, and includes great knowledge on the principles and best practices of software configuration Management (SCM) in agile, scrum and waterfall methodologies.

**PROFESSIONAL SUMMARY**

* Experience in working with **Azure IaaS - Virtual Networks, Virtual Machines, Cloud Services, Resource Groups, Express Route, Traffic Manager, VPN, Load Balancing, Application Gateways, Auto-Scaling** and Migrated existing **Active Directory** and Exchanged users to **Microsoft Azure AD** using **Azure AD Connect, Active Directory Federation Service**, **DirSync tools**, **Azure service bus**, Azure search, **Azure DNS**, **Azure VPN** Gateway, and Notification hub.
* Skillful in Administrating **Azure IaaS/PaaS** services like Compute **Azure** **Virtual** **Machines**, **Web**, and **Worker** **roles**, **VNET**, Network services and **Azure DevOps** services such as Repos, Test Plans, **Pipelines, Web Apps**, and **Application Insights**. Proficient in using **Azure service fabric** to package, deploy, manage scalable and **reliable microservices** and containers.
* Proficient as Cloud Administrator, involved in configuration for **Web apps, function apps**, **V-net integration, HCM, Application gateway, App Insights, Azure Key Vault, Encryption** and Security on **AZURE** using **ARM templates** and **PowerShell script**.
* Expertise in Developing different types of **Azure** **Functions** such as **HTTP** **Trigger**, **Timer** **Trigger**, **Service** **Bus** **Trigger**, **Event**-**Hub** **Trigger** and **Azure** **Content** **Delivery** **Network**.
* Implemented a CI/CD pipeline using **Azure DevOps (VSTS, TFS)** in both cloud and on-premises with **GIT, MS Build, Docker, Maven** along with **Jenkins’s** plugins.
* Experience in Infrastructure Development and Operations involving **AWS Cloud platforms, EC2, ELASTIC BEAN STACK (EBS), S3, VPC, IAM, WAF, EC2 Container Service, RDS, SES, ELB, Auto scaling, Cloud Front, Cloud Formation, Elastic ache, Cloud Watch, SNS, AWS Import / Export**.
* Worked on AWS DevOps tools like **AWS Code-Pipeline** to build a **continuous integration or** continuous delivery workflow using AWS Code-Build, AWS Code-Deploy, and worked with many other AWS tools to Build and deploy a microservices architecture using ECS.
* Expertise in writing **AWS Cloud Formation templates** in **JSON** to use them as blueprints for building & deploying multiple AWS resources. Worked on **Kinesis Data Streams & Kinesis Firehouse** & integrated with **AWS Lambda** for serverless data collection.
* Implemented **AWS** **Lambda functions** to run scripts in response to events in Amazon **Dynamo DB table, S3 buckets**, and **HTTP** requests using Amazon **API** Gateway. Also wrote **AWS Lambda functions** in **python** for **AWS Lambda** which invokes **python** scripts to perform various analytics on large data sets in **EMR clusters**.
* Experience in hosting an Application in **Google** **cloud** **Platform** by using **GCP** services like **Compute** **Engine**, **App** **engine**, **Cloud** **SQL**, **Kubernetes** **Engine**, **Stack** **driver** and **Cloud** **Storage**.
* Experience in monitoring the **GCP** environment using **Google** **Cloud** **Monitoring** and **Logging** and performed optimization tasks such as rightsizing of **virtual** **machines**, **optimizing** **storage** **costs**, and implementing auto-scaling policies to improve resource utilization and cost efficiency.
* Hands-on experience in deploying **Kubernetes** Cluster on cloud environment with master/minion architecture and wrote many **YAML files** to create many services like **pods, deployments, auto scaling, load balancers, labels, health checks, Namespaces.**
* Experience in working on **cluster** management on **Kubernetes** to create Pods, Nodes, and deploying **microservices** pulled from the **Docker Registry** and managed by **Kubernetes.**
* Expertise in using **Kubernetes** for **orchestrating containers, images, and container snapshots**, while creating a platform for **automating**, deploying, scaling of application containers across **clusters** of hosts.
* Experience in **Kubernetes** to manage the deployment rollouts and rollbacks and created **service mesh Istio** for the traffic management in the production environment. Created pod, deployment, namespace, and replication controller **YAML** definition files to Schedule, deploy and manage **container** nodes in different env dev/staging/prod clusters in **Kubernetes**.
* Extensively used **Terraform** to a reliable version and created infrastructure on **Azure**. Also created resources, using **Azure Terraform modules**, and automated infrastructure management. Used **Terraform** to map more complex dependencies and identify network issues.
* Experienced with **Terraform** key features such as **Infrastructure as code**, Execution plans, Resource Graphs, Change Automation and Used Auto scaling for launching Cloud instances while deploying **microservices.**
* Responsible for creating documentation of the **Terraform** infrastructure in **Confluence** and used **Terraform** to manage the **AWS** and other cloud infrastructure and managed servers using configuration management tools like **Ansible** and **Chef**.
* Experienced on various components of **Chef** including chef server, workstations, Nodes, configurations of chef-client and writing cookbooks and recipes for Chef and assigning roles to nodes.
* Strong knowledge of **Ansible** **Roles**, inventory, Ansible Tower, Ansible Galaxy concepts and used Ansible Control Server to deploy plays and playbooks to the machines and systems in the inventory. Wrote **Ansible playbooks** with **python** **SSH** as the Wrapper to Manage Configurations of **Azure Nodes** and Test **playbooks** on **Azure Virtual machines**.
* Experience in **Cloud Native** to optimize the software development for the cloud computing environments to build services that can communicate with other applications using API which are containerized and enables deployment and container orchestration.
* Configured and maintained **Puppet Master** server, also responsible for creating, and updating **Puppet** modules using manifests and pulling them from **Puppet Agents**.
* Experience in **SaltStack** for orchestration, managing the configuration and deployment of software by using Client-Server architecture with **ZeroMQ** protocol and automated through scripts.
* Expertise in **Continuous Integration** and **Deployment** **(CI/CD)**, automated deployments of full application stack using **Jenkins/Docker**, ongoing migration of mixed **CVS/SVN SCMs** over to **GIT**, deployment of **Docker** **containers** for development by maintaining **RHEL 6/ CentOS** servers on diverse **DevOps** systems.
* Experience in integrating **AutoSys** with **Jenkins** to automate the deployment, build, test, with a delivery pipeline to ensure faster and reliable releases across multiple environments and reducing manual intervention, risk of errors.
* Experience in working on several Docker components such as Docker Engine, Machine, Creating Docker Images, Compose, Docker Hub and handling multiple images for middleware installations and domain configuration.
* Expertise in setting up **Docker** daemon, **Docker** client, **Docker** hub, **Docker** registries, and images and handling multiple images by storing them in **containers** to deploy. Developed procedures to unify streamline and automate applications, development, and deployment procedures with **Linux** **container** technology using **Docker swarm**.
* Hands on experience in configuring **Jenkins** by identifying and installing required plug-ins. Wrote **Groovy** scripts to configure Build Jobs, Build Pipelines and by using Jenkins created a **master** and **slave** configuration to implement multiple parallel builds through a **build** **farm**.
* Experience in development of web applications using **Java**/**J2EE**, **Servlets**, **Hibernate**, **JSP, JSF, JSTL, Spring, EJB, Struts, JMS**, **ORM, JNDI, Web Services (SOAP, REST), Micro Services, JDBC, JAXP, Swing**.
* Configured multiple **Windows** and **Linux** bamboo agents for master in **Bamboo** to distribute the load across a Farm of machines. Design and setup of **CI** tool **Bamboo** to integrate **SCM** tool **Git** and automated the build process. Working with Build Verification team to make sure builds are delivered within deadlines.
* Experience in working on version control systems like **GIT** and subversion and used source code management client tools like **VisualSVN**, **GIT** **BASH**, **GIT** **HUB**, **GIT** **LAB**, **Bitbucket** and other command line applications.
* Extensive experience using **MAVEN** and **ANT** as build tools for building of deployable artifacts (**JAR**, **WAR** & **EAR**) from source code.
* Experience in **Liquibase** for managing database changes and versioning to help developers and teams track and apply database schema changes across different environments, such as development, testing, and production.
* Expertise in **DevSecOps** to implement security pratices and to protect the software associated data from potential threats, vulnerabilities, and attacks in production environment.
* Experience on **Sonar** **Cloud** to help development teams to improve the quality and security of the code by providing automated analysis and integrated with deployment tools to develop **cloud** environments and **CL/CD** pipelines.
* Used **Splunk** to monitor the system logs as well as notify the **incident** management system upon exceeding thresholds. **Splunk** was installed on production servers for logging purposes and created **Splunk** **dashboards** to monitor.
* Used **Nagios** as a monitoring tool to identify and resolve infrastructure problems before they affect critical processes and worked on **Nagios** **Event** handlers in case of an automatic restart of failed applications and servers.
* Implemented and managed monitoring and reporting of infrastructure using **Data** **Dog**, **New** **Relic**, and **Prometheus**.
* Experienced in troubleshooting and optimizing **ELK** clusters, to ensure high availability, performance, and security of Elastic search indices, log stash pipelines, and Kibana dashboards and integrated **ELK** with other DevOps tools like **Kubernetes**, **Docker**, **Jenkins**, and Cloud technologies to enable automated deployment workflows.
* Experience with **DevSecOps** to detect and prevent malware, ransomware, and enable security features like vulnerability management, Threat hunting, to approach real-time updates and threat intelligence across multiple cloud environments.
* Good scripting knowledge on **Pearl**, **Bash**, **Shell**, **Groovy, and Python** & Developed python, and shell scripts for automation of the build and release process.
* Extensive experience in developing server - side applications using **J2EE**, **Spring** **Framework** to deploy the backend components in the application server.
* Experience in **ServiceNow** for Business process automation and automate workflows, improve collaboration, and enhance the efficiency of business processes within the project. And to create and manage service requests, incidents, problems, and change management processes in an organized and structured manner.
* Good knowledge in Installation, configuration, and administration of databases like **MySQL**, **MongoDB**, **Oracle** **10g/11g,** **MSSQL**, **DynamoDB**.
* Experience in **Rally** software to allow teams to plan, track, and manage their work using agile methodologies such as Scrum or Kanban, and provides features for creating user stories, managing backlogs, and visualizing project progress through interactive dashboards and reports.
* Experience as an **SME** to train and monitor the new project members and develop content like developing training modules, writing technical documentation, conducting workshops, and problem solving.
* Experience in **.NET** development into a DevOps workflow, enabling faster development cycles, increased collaboration, and more reliable and automated deployment of .NET applications.
* Good Knowledge in installation and managing packages to easily upgrade the packages and build repository with prebuilt software packages for easy installation in different OS using **Chocolatey**.
* Observation in **System Administration**, **System Builds**, **Server builds**, **Installs**, **Upgrades**, **Patches**, **Migration**, **Trouble shooting**, **Security**, **Backup**, **Disaster** **Recovery**, **Performance** **Monitoring** and **Fine-tuning** on **UNIX** (Red Hat Linux) Systems. Performed all aspects of server management, including installation, updates, **AV** protection, **Upgrades**, **Storage** **management**, **Event** **log** **checking**, **Optimization**, and **Automation** **scripting** using **PowerShell**.
* Experience in installing, configuring, supporting, and troubleshooting Unix/Linux **Networking services** and **protocols** like **TCP, SMTP, HTTP, LDAP, DNS, NFS, DHCP, NAS, FTP, SSH,** and **SAMBA.**

**CERTIFICATIONS**

* **Certified Kubernetes Administrator**
* **Certified Azure Administrator**
* **Certified Azure DevOps Engineer**
* **AWS Certified Solutions Architect– Associate**

**TECHNICAL SKILLS**

|  |  |
| --- | --- |
| **Cloud** | Amazon Web Services, Microsoft Azure, GCP, OpenStack |
| **Container/ Orchestration Tools** | Kubernetes, Docker, OpenShift |
| **CM & Deployment Tools** | Chef, Ansible, Puppet, Terraform. |
| **CI/CD Tools** | Jenkins, Bamboo, TFS, Azure DevOps. |
| **Build Tools** | Maven, Ant, Gradle, InstallShield, MS Build. |
| **Repositories** | Artifactory, Nexus. |
| **Monitoring Tools** | JIRA, Prometheus, Splunk, Grafana, Logstash and Kibana (ELK) |
| **Version Control** | GIT, Subversion, CVS, Clear Case, Bitbucket |
| **Programming/ Scripting Languages** | Shell, Python, Ruby, Groovy, Bash, YAML, JSON, XML, PowerShell. |
| **Web/Application Servers** | Tomcat, Web logic, Web Sphere. |
| **Database** | SQL Server, No-SQL Database, My SQL, DB2 |
| **Operating Systems** | Windows, LINUX, RHEL, MAC. |

**PROFESSIONAL EXPERIENCE**

**Responsibilities**:

• Designed **Azure** infrastructure using **ARM** templates, **Azure Resources** and **Azure** **services** like **Azure Virtual Machines**, **Azure Data Bases**, **Azure Load Balancer**, **Azure Devops**, **Azure AppInsights**, **Azure Kubernetes Service (AKS)**, **Azure** **Monitor**, **Azure** **Backup**, and **Azure** **Site** **Recovery**.

• Business-As-Usual support in **IAAS**, **SAAS, LAAS** and **PASS** platforms using **Azure** Infrastructure **Azure** **Web** **Roles**, **Worker** **Roles**, **VM** **Role**, **Azure** **SQL**, **Azure Blob** **Storage**, **Azure** **AD** **Licenses**, **Virtual** **Machine** **Backup** and **Recover** from a Recovery Services Vault using **Azure** **PowerShell**, **Azure** **CLI** and **Azure** **Portal**.

• Built an application environment to operate at high availability in different **Azure** **regions** by using **Azure** **Content** **Delivery** **Network** (**CDN**), **Azure** **Traffic** **Manager**, **App** **Services**-**Web** **Apps**, **Active** **Directory**, **Storage** **Queues**, **Cognitive** **Services**, **Azure** **Search**, **Storage** **Blobs**, **Redis** **cache**, and **SQL** **Database**.

• Created **ARM** Templates in a **JSON** format to provision the infrastructure and maintain the consistency, manage the infrastructure as code, and to create azure services, resources and to spin up in different environments without effecting the **Dev**, **Non**-**Prod**, and **Production** environments.

• Configured **Azure Function App** to compile log data obtained from **Log Analytics Workspace**, using **python** **Handler** to create a **serverless** **environment** and notify the compiled data to **Azure** **Monitor**.

• Created **Azure** **Virtual** **Machines** with **Linux** **OS** on it to host the application and to configure it to run on an appropriate Web Server like **Apache**, **NGINX**.

• Used **SaltStack** to automate the deployment, management of virtual machines to easily spin up and to configure them at a scale.

• Created **Azure** **Devops** pipelines and integrated with **ARM** **Templates** by using **Azure** **tools** to define Infrastructure as code (**IAAC**) to set up **continuous** **integration** and **continuous** **deployment**, **App** **Builds**, **App** **Testing** and for **automating** the **app** **deployment** on **Azure** **servers**.

• Worked on **AutoSys** for the release management process to ensure the application is released at the right time with right configuration and defined release schedules and dependencies, ensuring that the application is deployed in a controlled and repeatable manner.

• Integrated **AutoSys** with **Azure** **DevOps** pipelines to automate the build process and deployment process in the production environment by defining them as set of jobs and managed the environment with the necessary resources are available across the environment.

• Used **Azure** **DevOps** **Server** for test planning, execution, and tracking to allow teams to define test plans, execute manual and automated tests, and track test results.

• Used **SaltStack** to automate the configuration of Azure DevOps pipelines and set up pipelines for continuous Integration and Deployment for the Azure infrastructure.

• Created **Azure** **Virtual** **Network** to secure the application and to control the traffic to and from for the virtual machines and defined the **Network** **Security** **Groups** (**NSG**), **Access** **Control** **Lists** (**ACL**) to restrict the app traffic by using **SaltStack**.

• Used **Java** for developing and maintaining application code, building API’s, developing microservices, and for web applications.

• Configured **Ingress** **Controller** by using **Azure** **Application** **Gateway** to provide **SSL** **Termination** for **offloading** the traffic and Load Balancing to manage and route incoming traffic to various backend servers and components of applications such as **Web** **server**, **API** **Server**, and **Data** **Base** **Server**.

• Designed overall **Microservices** architecture including Gateway, API Service, Core Service, Discovery Server, and Monitor Server using **SpringCloud** Netflix.

• Built **infrastructure** in such a way that deploying **Database** servers in different **VNets** and connect them by allowing **VPC** peering such that a **backup** **database** is replicated.

• Created Routing rules based on **Azure** **regions** and **Application** **layer** **protocols** to improve security and performance of application and done **Web** **Application** **Firewall** (**WAF**) integration to the application.

• Used **Terraform** for provisioning high availability of **Azure** **Infrastructure** & added on new plugins to support new functionality in Azure, involved in using **Terraform** to migrate legacy and monolithic systems to **Azure**.

• Maintained versioning infrastructure safely and efficiently by implementing the **HashiCorp** **Terraform** to create, change, and improve production infrastructure.

• Used **Microservices** architecture, with **SpringBoot** based services interacting through a combination of REST and **SpringBoot.**

• Used **Spring** **Boot** which is radically faster in building cloud Microservices and develop spring-based application with very less configuration.

• Designing and development of **RESTful** **API** and services using best practices to interact within the **micro** **services** and with the front end.

• Responsible for enforcing containerized based applications on **Azure** **Kubernetes** by using **Azure** **Kubernetes** service (**AKS**), Kubernetes Cluster for **cluster** **management**, **Virtual** **Network** to deploy **agent** **nodes**, **Ingress** **API** **Gateway**, **MySQL** Databases and **Cosmos** **DB** for stateless storage of external data, and setup reverse proxy **Nginx** **server** and encrypting with **SSL** and **TLS** skills.

• Created **Docker** images using a **Dockerfile**, created docker clusters and maintained the swarm setup using **Kubernetes** **Orchestration** for cloud-based applications.

• Used **Jenkins** pipelines to drive all microservices builds out to the **Docker** **registry** and then deployed to **Kubernetes**, Created **Pods** and managed utilizing **Kubernetes** and **Docker** for the runtime environment for the **CI/CD** system to build and test and deploy.

• Used **Sidecar** **proxy** in **service** **mesh** architecture, each microservice is paired with a sidecar proxy, to run alongside the microservice and intercepts inbound and outbound traffic, enabling various functionalities.

• Used **Service** **Mesh** for advanced Traffic management capabilities such as load balancing, routing, circuit breaking, retries, and timeouts, this enables improved resiliency and fault tolerance in a distributed system.

• Used **OpenShift** for deployment across multiple environments, including public clouds, private clouds, and on-premises infrastructure, to provides consistent management and portability of applications across different environments, like Dev, Prod and Non-prod environments.

• Installed **Ansible** Module plugins with **Azure** to define and manage Infrastructure as code using **YAML** syntax and to configure **Azure** resources such as **Virtual** **Networks**, **Storage** **accounts**, and setup App software services and dependencies on those resources.

• Created several **Ansible** **Playbooks** to automate the deployment and used it to **orchestrate** complex deployments involving multiple systems and servers.

• Used **HashiCorp** -**Vault** to Encrypt and Decrypt the file and Deployed on Client servers using **AWX**. Implemented Infrastructure automation through **Ansible**, for Auto provisioning, Code deployments, software installation and configuration updates.

• Created a **Source** **Control** **System** and integrated with source code repositories including **GITHUB**, **BIT** **Bucket**, and **Azure** Repos to manage **Code** **Base**, **Version** **Control**, and to collaborate with other developers using **Source** **Control** in **Azure**.

• Wrote **YAML** files for **CI/CD** pipelines and integrated with **Source** **Control** **System** to trigger the pipeline on every code commit.

• Used **Liquibase** to define database schema changes using XML, YAML, or SQL, and these changes are stored in a changelog file that contains a list of changesets that describe the modifications to be made to the database schema and changeset represents a specific database change, such as creating a table, modifying a column, or inserting data.

• Integrated **Liquibase** with continuous integration and continuous delivery (CI/CD) pipelines, build automation tools and deployment processes, allowing for consistent and automated database schema management across different environments to ensure that database changes are applied consistently, eliminating the need for manual intervention, and reducing the risk of errors.

• Used **Azure Active Directory** to create **users**, **groups**, **service** **principal** names and **multiple** **custom** **roles**, and to provide **Identity** **Access** **Management** (**IAM**) to the environment to **control** **access** to the **Azure** **resources** and to enable **Multifactor** **Authentication** (**MFA**), **Single** **sign**-**on**, **Authorization**, **Role** **Based** **Access** **Control** (**RBAC**) and automated it by using **Azure** **PowerShell** **scripts**.

• Used **PlainID** for Real-Time Policy Enforcement, ensuring that access decisions are made instantly based on the defined rules to help organizations maintain security and compliance in dynamic environments.

• Used **Azure** **Key** **Vault** to store and manage **cryptographic keys**, **sensitive information**, and **certificates** in **Azure** in a secure way and established a connection between **Azure Key Vault** and **Azure services** using **Azure Private Link**.

• Implemented **Disaster** **Recovery** and **Business** **continuity** plan to ensure the application can recover quickly from any potential disruptions or failures by using **Azure** **Backup** to restore the app data and used **Azure** **Site** **Recovery** to replicate the application into secondary location.

• Worked with Monitoring tools like **Azure** **log Analytics** Workspace to run log queries with data provided by **Azure** **Monitor** to sort, filter and analyze data and used **Azure** **App** **Insights** to monitor the performance, and conducted health checks using **Azure** **Resource** **Health**.

• Established a **Service** **Connection** between applications and **Azure** **Services** to monitor, build and test using **Azure** **Cloud** **Service** **Connection**.

• Build servers using **GCP**, imported volumes, and launching **Azure** **services**, creating **security** **groups**, **auto** **scaling**, **Load** balancers in the defined **virtual** **private** **connection** and migrated those services to the **PKS**, **GCP** cloud.

• Experience in migrating infrastructure and application from on premises to **Azure**, and **GCP**. Has Experience in automation of code deployment across **Azure**, **Google** **Cloud**, **VMWare**, and **OpenStack**.

• Migrated and stored structured, semi-structured, and unstructured data and applied data encryption, date lake fire wall, virtual network service end points, and **Azure** **AD**- based **authentication** for big data processing and workflow analysis by using **Azure** **Data** **Lake** **store**, **Azure** **Databricks**, and **Azure** **Blob** **Storage**.

• Configured **DevSecOps** and connected with azure to enable security, threat hunting, and vulnerability management to ensure the cloud environment is protected against cyber threats.

• Integrated **ServiceNow** with CI/CD tools such as Jenkins, Git, and Azure DevOps, to enable the synchronization of data, such as build and deployment status, between ServiceNow and CI/CD tools, providing visibility and traceability across the entire DevOps pipeline.

• Deployed data processing applications in **Azure** **DevOps** pipelines to process data stored in **Azure** **data** **lake** and to provide set of tools and libraries for data manipulation, analysis, and visualization by using **.NET** and **C#** to meet specific requirements.

• Used **.NET** for Artifact Management in a package manager like **NuGet** to manage dependencies of .NET project it allows to easily refer and update external libraries and components and set up an artifact repository like Azure Artifacts or **Sonatype** **Nexus** to store and manage your own internal packages.

• Experience in writing **Unit** **tests** which are dedicated test frameworks or libraries that provide a structure and tools for defining and executing tests for **.NET** include **NUnit**, **xUnit**, and **MSTest**.

• Configured **Kafka** heavy forwarder for Splunk in production environment and Integrated **Kafka** heavy forwarder into the Splunk deployment server.

• Created **Micro repos** for Decentralized Version Control, with micro repos, each repository has its own version control history, allowing for granular tracking of changes specific to a microservice, used for easier rollbacks, bug fixes, and feature development for individual services.

Used **Rally** to enable collaboration and communication among team members by providing features like shared boards, discussion forums, and real-time updates and helped teams work together efficiently and stay aligned on project goals and tasks.

• Used **DevSecOps** to aim security, automate security processes where possible, and ensure that security is an integral part of the overall software development process.

• Integrated **SonarQube** with **Azure** **DevOps**, **Azure** **Monitor** to allow developing teams to automate code quality analysis as a part of their development workflows and to monitor code quality metrics, build in reports over time.

• Experience in Administration, maintenance, and support of **Red Hat Enterprise Linux** (**RHEL**) servers. Involved in Building and configuring **Red Hat Linux Servers** using Kick Start server as required for the project and prepared **POC** documentation for all major implementations and Critical issues in the Environment.

**Environment**: Azure ARM Templates, Terraform, Azure Devops, Git, DevSecOps, AutoSys, Azure-Kubernetes, Ansible, CLI, Auto Scaling, Linux, Shell scripting, JFrog, Tomcat, Python, CrowdStrike, Power Shell, SaltStack, AZ-Active Directory, Azure Monitor, Linux, Azure Virtual Machines, Azure Service Principles, Azure Blob Storage, IAM, Azure PowerShell, Azure Site Recovery, Azure Backup, Azure Key Vault.

**Responsibilities**:

• Experienced in designing, configuring, and deploying solutions on **Microsoft** **Azure** using (**Azure Resource Manager**) **ARM** Templates, **AZURE** **Power** **Shell** **Module** and **Azure** **CLI** focusing on high-availability and auto- scaling. Designed& implemented migration strategies for traditional systems on **Azure** (Lift and shift/Azure Migrate, other third-party tools

• Migrated the workloads from **Amazon** **Web** **Services** (**AWS**) Cloud to **Microsoft** **Azure** **Cloud**, deployed the Plate Spin Migrate server in the target **Azure** **environment**. Migrated **Agent** to register workloads with the cloud based Migrate server using secure communications over the public Internet.

• Designed and created new data pipelines in **Azure** using **Azure** **DevOps** services and migrated data from **AWS** data sources to **Azure** **pipelines** and updated dependencies, **API end points**, **connection** **strings**, **authentication** mechanisms and other configuration settings to use the **new Azure data pipelines**.

• Post-migration enabled the **Azure** **Site** **Recovery** (**ASR**) for the disaster recovery purpose and to back up the **virtual** **machines** and used **Azure** **Backups** to create **replicas** in the **Azure** to prevent the downtime of applications if any region or availability zone goes down.

• Created **Azure** **Automation** **Assets**, **Graphical** **runbook**, **Power** **Shell** **runbook** that will automate specific tasks, deployed **Azure** **AD** **Connect**, configuring **Active** **Directory** **Federation** **Service** (**ADFS**) authentication flow, **ADFS** installation using **Azure** **AD** **Connect**, and involved in administrative tasks that include **Build**, **Design**, **Deploy** of **Azure** **environment**.

• Designed a **Content** **Delivery** **Network** to reduce latency and increase application performance by caching content closer to the user and by duplicating the original server in various regions.

• Created **Terraform** templates for provisioning **virtual** **networks**, **VM** **Scale** **sets**, **Load** **balancers** and **NAT** rules and used **Terraform** **graph** to visualize execution plan using the graph command. Used **Terraform** to deploy the infrastructure necessary to create **development**, **test**, and **production** environments for software development.

• Designed a detailed migration plan that includes timelines, risk assessments, resource allocation and transferred the application **data** like **files**, **databases**, and other **data** **repositories** by using **Azure** **services**.

• Used **Azure Data Factory** and integrated with **Amazon S3** to create, schedule, orchestrate, and move data workflows from **AWS** to **Azure** in a scalable and automated manner through **ARM** templates.

• Implemented **Cut-Over** process after the post migration and directed the **traffic** across the **Azure** **servers** and monitored to ensure optimal performance and reliability and performed post migration validation to finalize the migration process.

• Used **Azure** **DevOps** **Server** to support build automation and release management, enabling teams to define and execute build processes, run tests, and create automated release pipelines for CI/CD workflows.

• Extensively worked on infrastructure development and operations by involving in designing and deploying using **AWS** services like **AWS EC2 Instance**, **AWS Kinesis**, **Route53**, **DNS**, **ELB**, **EBS**, **AMI**, **IAM**, **VPC**, **S3**, **RDS**, **EKS**, **Elastic Beanstalk**, **CloudFront**, **Elastic** **block** **store** (**EBS**), **Cloud** **trail**, **Dynamo** **DB**, **Cloud** **Watch** **monitoring**.

• Used **SaltStack** to automate the deployment and management of **EC2** instances like spinning up new **EC2** instances, configuring security groups and managing those security groups across multiple instances to ensure consistent security.

• Built servers using **AWS**, **importing** **volumes**, launching **EC2, RDS**, creating **security** **groups**, **auto**-**scaling**, **load** **balancers** (**ELBs**) in the defined **virtual** **private** **connection** (**VPC**) by providing high availability of applications of **EC2** instances by taking the advantages of **AWS** High availability (**HA**) options and configured **S3** versioning and lifecycle policies in **AWS**, to backup files and archive files in Glacier.

• Utilized **Java** 8 features like Lambda expressions and **Stream** **API** for Bulk data operations on Collections which would increase the performance of the Application.

• Implemented **AWS** web application firewalls to monitor the web requests that are forwarded to **API** Gateways and Application **Load** **balancers** (**ALBs**) to enable the security model. Created and managed Kinesis streams configured logs in Splunk from **AWS** cloud watch.

• Developed **API's** and **REST** **API** proxy using **APIGEE** **Edge** and for sending mobile notifications to the account when any issues raised in the infrastructure.

• Used **CrowdStrike** to integrate with **AWS** pipelines to ensure that the data is transferred from **AWS** to **Azure** pipelines to avoid any data breaches, vulnerability, Threats in the infrastructure.

• Integrated PlainID with various identity providers, authentication systems, and applications, making it compatible with existing infrastructure to be deployed both on-premises and in the cloud, providing flexibility for different deployment scenarios.

• Trouble shooting threats, compatibility issues, Resource consumption, Integration issues using **CrowdStrike** tools in the environment which increases monitoring speed.

• Worked as an **SME** for consultation and advice to provide guidance to other team members, and trouble shoot issues, identify root causes, propose strategies to overcome challenges and ensured quality by conducting audits, reviewing processes and procedures across the domain.

• Used **SaltStack** to automate the configuration of **AWS** **CloudFormation** templates for infrastructure as code and automate the deployment of **AWS** resources.

• Used **Jenkins** pipelines to drive all microservices builds out to the **Docker** registry and then deployed to **Kubernetes**, created Pods, and managed using **Kubernetes**.

• Used **AutoSys** for job scheduling and job monitoring by defining the jobs that should run on a one-time basis or at recurring intervals by specifying start times, end times, and other scheduling parameters for each job.

• Experience in trouble shooting overall workflow due to incorrect job parameters, insufficient permissions, network issues, failed tasks, and incorrect job execution by using **AutoSys**.

• Expertise in using build tools like **MAVEN** for the building of deployable **Artifacts** such as **War** & **Ear** from Source Code integrated with Jenkins deploying them in **Kubernetes** **clusters**.

• Worked with **Docker** and **Kubernetes** on multiple **cloud** **providers**, from helping developers build and **containerize** their application to deploying either on **public** or **private** **cloud**.

• Used **EKS** to automatically scale the underlying infrastructure based on the workload requirements and to ensure that applications have the necessary resources to handle increased traffic and demand.

• Managed **Kubernetes** **Helm** **charts**, Created reproducible builds of the **Kubernetes** applications, managed **Kubernetes** manifest files and managed releases of **Helm** **packages**. Established a local dev workflow that is centered around **minikube** to validate deployments in **Kubernetes**.

• Used **Service** **mesh** for Security and encryption such as mutual Transport Layer Security (**mTLS**) authentication between services, encryption of traffic, and policy enforcement at the network level.

• Used **Istio (Service mesh)** for traffic management capabilities such as intelligent routing, load balancing, and fault injection. It allows for fine-grained control over traffic behavior, including A/B testing, canary releases, and traffic shifting between different versions of services.

• Used **HashiCorp-Vault** for Secrets Management which is primarily used for securely storing and managing secrets. This includes passwords, API keys, database credentials, encryption keys, certificates, and more by using the Vault API, command-line interface (CLI), or through various integrations with other tools.

• Integrated **EKS** with AWS Identity and Access Management (IAM) for authentication and authorization, ensuring secure access to **Kubernetes** clusters and to provide network-level security controls to protect the applications running on AWS.

• Configured and Integrated **Docker** container **orchestration** framework using **Kubernetes** by creating automated pods deployments and used **Python** Scripts to schedule, deploy and manage container **replicas** onto a node cluster using **Kubernetes**. Configured **Docker** **Swarm** **Cluster** and setup **Jenkins** inside it to reduce the failover downtime.

• Created **Docker** **images** using a **Docker** **file**, worked on **Docker** **container** **snapshots**, **removing** **images**, and managing **Docker** **volumes** and experienced with **Docker** **container** **service**.

• Provided backup support for **Jenkins** build environment and involved in upgrade of **Bamboo** & **Artifactory** **Server** and adopted **Jenkins** for **CI** practice and revised existing build procedures.

• Integrated **AutoSys** with **ELK** to monitor jobs that involve resolving security issues, performance issues, configuration issues, and database issues.

• Developed **Chef** **Cookbooks** to install and configure **Apache**, **Tomcat**, **Splunk**, **Jenkins**, **WebLogic**, and deployment automation and Integrated **Jenkins**, **Rundeck**, and **Chef**.

• Used **Liquibase** to track and manage the state of the database schema, allowing you to apply changes incrementally and in a controlled manner and to support features like rollbacks, so if a change needs to be reverted, **Liquibase** can execute the necessary SQL statements to undo the modification.

• Used **Code** **Coverage** to measure the extent to which the source code is exercised by unit tests and to identify areas of code that are not adequately covered by tests, enabling teams to increase the test coverage and improve overall test effectiveness.

• Responsible for configuring **Kafka** Consumer and producer metrics to visualize the Kafka system performance and monitoring and Configured Splunk app for ONTAP and Kafka.

• Created **Micro Repos** for Independent Release Cycles, since each microservice has its own repository, release cycles can be decoupled. Teams can independently release and deploy updates to their respective services without impacting other components. This enables faster iteration and deployment of individual services.

• Setup and integrated **Prometheus** with **Grafana** for Monitoring. Used **Elasticsearch** (**ELK** **stack**) for centralized logging and analytics in the continuous delivery pipeline to store **logs** and **metrics** into **Azure** **storage** using **storage** **function**.

• Experience in **Confluence** as a repository for capturing and maintaining documentation for DevOps practices, processes, and procedures to enable teams to document infrastructure configurations, deployment pipelines, troubleshooting guides, standard operating procedures, and more.

**Environment**: Azure, AWS, AMI, Elastic IP’s, AWS Kinesis, Terraform, AutoSys, AWS CloudFormation, AWS DynamoDB, Jenkins, AWS ECS, CrowdStrike, LDAP servers, Terraform, Jenkins, SaltStack, Docker, Grafana, Jenkins Chef, Jenkins, Shell, GIT.

**Responsibilities**:

• Provisioned infrastructure using **Cloud** **Formation** templates using **AWS** services such as **S3**, **EC2**, **ELB**, **EBS**, **RDS**, **ECR**, **EKS**, **VPC**, **auto**-**scaling**, **Kinesis**, **Route** **53**, **Direct** **Connect**, **IAM**, **CloudFormation**, **Ops** **Works**, **Elastic** **Beanstalk**, **AWS** **S3**, **Amazon** **Glacier**, **EMR**, **RedShift**, and **Cloud** **Watch** and **Cloud** **Trail**.

• Maintained scalable environment for application servers using **command** **line** **interface** for Setting up and administering **DNS** system in **AWS** using **Route53** Managing users and groups using the **Amazon** **Identity** and **Access** **management** (**IAM**) and implemented **DNS** service through **Route** **53** on **ELB’s** to achieve secured connection via **https**.

• Distributed Traffic across multiple **EC2** instances and updated the **DNS** record of the domain name and terminated **SSL** by using **ELB’s** to provide higher availability and improved performance.

• Worked on Multiple **AWS** instances, set the **security** **groups**, Elastic Load Balancer and **AMIs**, Auto scaling to design cost effective, fault tolerant and implementing high availability. Created **Lambda** **functions** to upload code and to check changes in **S3**, **Dynamo** **DB** table.

• Worked on Custom **Domain**, **Record** **Sets**, **DNS** **health** **checks** to route the traffic by **Amazon** **Route53** for applications hosted in **AWS** Environment and managing users and groups using **AWS** **IAM**.

• Created and managed **AWS** **Cloud** **Formation** **Stack** using **VPC**, **subnets**, **EC2** **instances**, **ELB**, **S3** and integrated it with **CloudTrail**. Versioned **CloudFormation** templates are stored in **GIT**, visualized **CloudFormation** templates as diagrams and modified them with the **AWS** **CloudFormation** Designer.

• Involved in designing and deploying multitude of applications utilizing almost all of the **AWS** stack (Including **EC2**, **Route53**, **S3**, **RDS**, **Dynamo** **DB**, **SNS**, **SQS**, **IAM**) focusing on high-availability, fault tolerance, and auto-scaling in **AWS** **cloud** **formation**.

• Created a relational **MySQL** **Database** schema to configure **storage** **capacity**, **backup** **retention**, **Disaster** **recovery** **security** **groups** and added **READ** replicas to offload the read traffic from the primary instance by using **RDS** **console**.

• Created **EBS** volumes to **store** **persistent** **data** and use snapshots to reduce failure. By taking point-in-time snapshots gave the ability to backup **Amazon** **EBS** volumes to **S3**.

• Worked on **AWS** **Lambda** to run the code in response to events, such as changes to data in an **Amazon** **S3** **bucket**, **Amazon** **DynamoDB** table, **HTTP** requests using **AWS** **API** **Gateway**, and invoked the code using **API** calls made using **AWS** **SDKs**.

• Used **SaltStack** to manage and set-up configuration of **AWS** Lambda functions and automated the testing, building, and deployment of serverless applications.

• Production experience in large environments using configuration management tools like **Chef** supporting **Chef** Environment with **500+ servers** and involved in developing manifests and developed **Chef** **Cookbooks** to manage system configuration.

• Trouble shooting issues like configuration drift, scalability issues, Network connectivity issues by using **SaltStack**.

• Deployed custom applications using **Chef**, executed **schema** updates with **Liquibase** and coordinated everything with **Jenkins**, Configured and maintained **Jenkins** to implement the **CI** process and integrated the tool with **Maven** to schedule the builds. Took the sole responsibility to maintain the **CI** **Jenkins** server.

• Used **Knife** and **Chef** **Bootstrap** processes and worked on the **chef** **server** management console, with a thorough understanding of all components such as the **chef** **server**, **nodes**, and **workstations**.

• Installed and configured **AutoSys** to communicate with Databases and created job schedules to automate deployments at specific times and monitored the status of each job and manages the jobs if any issues raised.

• Implemented the **Docker** for wrapping up the final code and setting up development and testing environment using **Docker** **Hub**, **Docker** **Swarm** and **Docker** **Container** **Network**.

• Wrote templates for **AWS infrastructure** as a code using **Terraform** to build staging and production environments and **orchestrated**, migrated **CI/CD** processes using **Cloud** **Formation** and **Terraform** Templates and **Dockerised** the infrastructure, which was setup in **Vagrant**, **AWS** and **VPCs**.

• Used **CrowdStrike** to solve Compatibility issues with certain cloud environments, operating systems, helps with installation configuration and performance.

• Used **HashiCorp-Vault** for Encryption as a service, providing encryption and decryption capabilities for sensitive data to generate and manage encryption keys, encrypt data at rest or in transit, and securely store the encryption keys.

• Enabled **CrowdStrike** Falcon application to set up security policies between AWS and other cloud environments to secure the integrated API credentials, Tenant ID, and Client ID.

• Designed branching strategies for using **Version** **Control** **Systems** like **GITHUB**, Clear Case, Stash and Developed **GIT** hooks for the **local** **repository**, **code** **commit** and **remote** **repository**, **code** **push** **functionality**.

• Created **Mono Repos** for Streamlined Continuous Integration/Continuous Deployment (CI/CD) to simplify CI/CD workflows as all projects are part of the same pipeline. Changes across multiple projects can be tested and deployed together, reducing the complexity of managing separate pipelines for each individual project environment.

• Implemented **Logical** **Volume** **Manager** (**LVM**), **VERITAS** **Volume** **Manager** (**VVM**), and **Solaris** **Volume** **Manager** for the efficient installation, configuration, and deployment of **RAID** **0**, **RAID** **1**, and **RAID 5** levels.

• Aligned and managed **Amazon** **Linux** **kernel**, **Ubuntu**, memory upgrades and swaps area. Configuring **DNS**, **DHCP**, **NIS**, **NFS** in **Sun** **Solaris 8/9** & other **Network** **Services**.

• Involved in Setting up **Elastic** **Search**, **Log** **stash** and **Kibana** (**ELK**), Created **AWS** **Multi**-**Factor** **Authentication** (**MFA**) for instance **RDP/SSH logon**, worked with teams to lock down security groups, Served the **ELK** stack community with use cases and **logstash** plugin and Deployed applications using Jenkins server and troubleshoot, build & release job failures, resolve, work with developers on resolution.

• Have created and released reusable patch as an **AMI** specifically in **REDHAT** for client local **AWS** environment while installing all majorly required applications in the image.

**Environment**: AWS services, Route 53, IAM, Cloud Formation, VPC peering, MySQL, CHEF, Maven, Docker, Terraform, Red Hat Enterprise Linux 6/7, python, ELK, SaltStack, Jenkins, AutoSys, JIRA, Confluence, GIT, RedShift Data Warehouse, DockerHub, Load Balancers, CrowdStrike, IAM.

**Responsibilities**:

• Supported engineering plans and schedules by providing various **Build** and **Release** services to build, deploy, develop scripts, oversee branch, and merge strategies, and build automated tools as necessary to offer services to the engineering team.

• Create and maintain highly scalable and fault-tolerant multi-tier **AWS** environments spanning across multiple availability zones using **Terraform**.

• Experience in managing **IAM** policies with active directory integration to manage security in **GCP** and **AWS**.

• Experience in creating shared **VPC** with different tags in a single **GCP** project and using the same in all the projects.

• Managing and supporting the **AWS** team and setting up the IPSec tunnel between **Google** **Cloud** and **AWS** Networking infrastructure.

• Configuring and deploying instances on **GCP** environments and **Data** centers, also familiar with **Compute**, **Kubernetes** **Engine**, **Stackdriver** **Monitoring**, **Elastic** **Search** and managing **security** **groups** on both.

• Installed, Configured, Managed, and Created different **Build** and **Deployment** Jobs in **Jenkins**. Installed several plugins in **Jenkins** to support multiple tools required for the implementation of projects.

• Worked on troubleshooting the build issues during the **Jenkins** build process and developed build and deployment scripts using **ANT** as build tools in Jenkins to move from one environment to another environment.

• Installed **Jenkins** for Continuous Integration and wrote **Shell** **script** for an end-to-end build and deployment automation and used **Jenkins** to automate most of the build-related tasks.

• Administered and Engineered **Jenkins** for managing nightly Build, Test and Deploy chain, **GIT** with Development/Test/Production Branching Model for weekly releases.

• Experience with **Cloud** **Spanner**, to perform database schema updates with zero downtime

• Build and release software baselines, **code** **merges**, **branch** and **label** **creation**, and the interface between development and infrastructure.

• Built **Puppet** enterprise modules using puppet **DSL** to automate infrastructure provisioning and configuration management to existing infrastructure by deploying **Puppet**.

• Developed **Puppet** modules with Jenkins for continuous integration and continuous deployment of managed products, and related services.

• Experience with build tools **Ant** and **Maven** for writing **build**.**xmls** and **pom.xmls** respectively.

• Experience in integrating **Unit** **Tests** and **Code** **Quality** **Analysis** **Tools** like **JUnit**, **findbugs**, and **Selenium**.

• Experience in Configuring and deploying to Application servers **Weblogic** and **Websphere.**

• Hands-on Experience in using **Tomcat** and **Apache** web servers for deployments and for hosting tools.

• Worked on tools migration from old tools like **PVCS** version control system to **SVN**, Tracker to **Jira**, and finally **CI** tool **Hudson** to **Jenkins**.

• Daily operations including monitoring **Jenkins**, **Jira**, **Bitbucket**, and **Nexus** Logs for any errors, troubleshooting Deployment issues, help application teams with any issues regarding **DevOps** tools.

• Worked with Development Team Leads and testing teams to establish a build schedule, execute the builds and troubleshoot build failures, if any.

• Sent releases to **SID** testing, resolved issues before going into the questionnaire and drafted the documents for each release about the issues including the lessons learned.

• Successfully managed concurrent deliverables on time with quality within a fast-paced environment and under deadlines.

• **SME** for DevOps Toolset, prepared Architecture blueprints, and requirements documentation including base infrastructure details like system configuration, **software** **versions**, **firewalls**, **sudo** **rules**, and **service** **accounts**.

• Maintained the **Red** **Hat** **Satellite** for infrastructure management to keep **Red** **Hat** **Enterprise** **Linux** environments and other **Red** **Hat** infrastructure running efficiently Gme$m00n, properly secured, and compliant with various standards.

• Configured and managed **Red** Hat **Linux** **kernel**, **memory** **upgrades** and swaps area. **Red** **Hat** **Linux** **Kickstart** Installation **Sun Solaris** Jump start Installation. Configuring **DNS**, **DHCP**, **NIS**, **NFS** in **Sun** **Solaris** **8/9** & other **Network** **Services**.

**Environment**: Nexus, Jenkins, Puppet, Maven, Ant, GIT, Subversion, Splunk, Unix, Linux, LDAP, Active Directory, Java/J2EE, WebLogic, Agile, Nagios, shell, python, DHCP, TCP/IP.