\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*INTRODUCTION\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

I work at a company called \*\*\*\*\*\*\* based from Amarillo, Tx. Currently I am working on different cloud service like AWS, GCP. We recently have migrated on premises data centers to aws cloud using several aws services focusing mainly on AWS snowball, Aws import export. Basically We are of

As a Sr. DevOps/cloud engineer, I was responsible for migrating on premises data centers to aws cloud using several aws services like IAM, EC2, ROUTE 53, S3, RDS, VPC, CloudWatch and some of the migration tools like cloud formation. Some of my responsibilities are setting up the complete AWS environment starting from spinning up EC2 instances in RedHat 7 and ubuntu 16.04,18 X and provisioning them with necessary packages by using continuous management tools like chef,ansible and also I used ant, maven as build tools.

I implemented additional services by creating IAM user accounts, groups, adding user to the groups, generating custom policies, customizing the JSON template, Created snapshots and Amazon Machine Images (AMI) of the instance for backup. Checking the inbounds outbond rules for the security groups, managing the elastic load balancer and auto scaling groups, Monitoring Live Traffic, logs, Memory utilization, Disk utilization and various other factors by using CLOUD WATCH metrics and making sure that the metrics are available in SNS dashboard. Used S3 for storing the project archives by creating buckets and by using IAM service provided permission to the users and admin to access the buckets. Creating the record sets in Route 53 and setting up cnames and alias to the load balancer. Creating health checks to reduce failovers and used weighted latency to split the traffic. Apart from that I am also responsible for migrating the data from on-premises to AWS using the AWS migration tools like AWS import export and AWS snowball. I used JIRA as an issue tracking and ticket raising tool to solve operational related issues, Splunk for monitoring security issues and Nagios for monitoring resources.

I am responsible for building the infrastructure and maintaining the CI/CD pipeline by using Jenkins as a continuous integration tool and I wrote chef recipes and deployment scripts to automate the deployment process. For source code management, I use GIT and SVN where I am responsible for creating branching, labeling, and merging strategies for all applications in GIT. I am also responsible for configuring environments for DEV, QA, production and provisioning them with necessary packages using configuration management tools like CHEF, ANSIBLE. I dockized all the applications that are coded in Java, PHP, Node.js and Angular.js using Openshift as IAAS platform. I managed docker service by creating Docker images, docker Containers, Docker Registry to store images, cloud-based registry called Docker Hub, Docker Swarm to manage containers. Managed docker clusters container management by Installed and configured Kubernetes for Orchestration of Docker Images using Kubernetes Operations (KOPS) and wrote kubernetets (YAML) files to build micro-services. Involved in migration of CI/CD process using Cloud Formation, Terraform templates and Containerized the infrastructure using Docker. Used Cloud Front to deliver content from AWS edge locations to users, allowing for further reduction of load on front-end servers.

**Role**

In my current role, I am acting as DevOps Engineer, Automate Infra provisioning in Cloud, Write Atuomation in Infra Config languages like Chef, Puppet, Deploy/run apps and services in Container Technologies like Docker, Help Automate and day to day support of various things in CI Pipelines for various Apps, Strong knowledge on CI tools like Jenkins, Build Automation tools like maven, gradle etc, Version control systems like Git etc

**AWS:**

* My primary tasks in current project include creating **Cloud** **Formation scripts** for hosting software on **AWS** cloud and Automate the installation of software through **Power** **Shell** **scripts**.
* And I have designed highly available, cost effective and fault tolerant systems using multiple **EC2** instances, by **Auto Scaling, Cloud Watch, Elastic Load Balancer**  and Maintained the user accounts using **IAM.**
* Some times Optimized volumes and **EC2** instances & Created Multi Availability Zone **VPC** instances.
* Experience in migrating various data center applications, services and db’s to cloud.
* I have used the boto3 sdk of aws(python based) for writing all this Automation.
* Also Implemented and maintained the monitoring and alerting of production and corporate servers/storage using **AWS Cloud watch**.
* Administered setting up of private networks and sub-networks using Virtual Private Cloud (**VPC**) and creating security groups to associate with the networks in **AWS**.
* Maintained **VPC, RDB, Dynamo DB, SES, SQS and SNS services, Network ACL's and Route Tables** in **AWS** cloud.
* Experience in core AWS services **(S3, EC2, EBS, Route53, VPC, auto scaling)** and deployment services (**OpsWorks and Cloud Formation**) and security practices (**IAM, Cloud watch and Cloud trail**).

**Azure:**

* Utilized **ASR (Azure Site Recovery),** PowerShell and **MVMC (Microsoft Virtual Machine Convertor)** for on premise to cloud migrations.
* Configured **BGP(Border gateway protocol)** routes to enable **ExpressRoute** connections between on premise data centers and Azure cloud.
* Exposed Virtual machines and cloud services in the VNets to the Internet using **Azure External Load Balancer.**
* Provided high availability for IaaS VMs and PaaS role instances for access from other services in the VNet with **Azure Internal Load Balancer.**
* Led implementation of Office 365 and Azure Active Directory for single sign on, authentication, authorization and **Azure Role-based Access Control (RBAC).**
* Implemented high availability with **Azure Classic and Azure Resource Manager** deployment models.
* Designed **Network Security Groups (NSGs)** to control **inbound** and **outbound** access to **network interfaces (NICs), VMs and sucbnets**.
* Architected automation of **Azure Virtual Networks, Network Security Groups** and **access control rules.**
* Setup Azure Virtual Appliances (VMs) to meet security requirements as software based appliance functions (firewall, WAN optimization and intrusion detections).
* Involved on migrating **SQL Server databases to SQL Azure Database** using **SQL Azure Migration Wizard** and then Deployed application to Azure Cloud.

**OPEN Stack:**

Probably talk about various APIs provided by openstack like Nova, Cinder, Glance etc, using Openstack sdk etc

* Designed and **architected cloud application deployment in OpenStack with Nova, Neutron, Keystone command line clients.**
* Worked across both private (OpenStack) and public clouds (Amazon AWS).
* fixed issues related to OpenStack components Nova, Glance, Neutron, Keystone.
* Deployed OpenStack icehouse, juno, kilo, liberty releases in a multimode environment manually.
* Expertise in deployment of OpenStack with Devstack and Packstack.
* Deployed Zabbix to monitor and alert the health of Nova, Neutron, Keystone and other OpenStackservices.
* Used Juju charms and Ubuntu autopilot to deploy the OpenStack Lab on top of MAAS.
* As an OpenStack admin, managed creation of new users, tenant and assigning resource quotas to OpenStack users using Keystone.
* Extensive exposure in OpenStack Command Line interface to create instances, images, volumes, flavors, containers and tenants.

Built, deployed and managed OpenStack Environment using Mirantis Fuel 7.0 for PCF bosh deployments.  Openstack is mostly being used for Private or Hybrid Clouds. It is a Open source Cloud OS.

Added more computes nodes to existing OpenStack environment with the help of Supermicro IPMI(Intelligent Platform management interface).

Integrated Open Daylight as SDN controller in OpenStack Environment.

Expertise in OpenStack services such as Horizon, Keystone, Nova, Neutron, Glance, Cinder, Ceilometer and Swift.

**PCF(Pivotal Cloud Foundry)**

Responsible for the deployment and configuration of Microservices applications in Pivotal Cloud Foundry.

Deployed the application on Pivotal Cloud Foundry

Used Pivotal Cloud foundry and Cloud Foundry Command Line Interface (CLI) for managing and   
Deploying the applications.

Fine tune and scale the applications in cloud foundry.

Performance Profiling for Micro services on pivotal cloud foundry environment.

Using Pivotal Cloud Foundry(PCF) for application deployment, developing Jenkin build Jobs.

Deploying and managing applications in Cloud Foundry and creating database instances of PostgreSQL.

Initial administered the creation of users with UAAC(UAA Command Line Interface).

Integrating Active Directory instead of UAA( User account and authentication) in PCF.

Application Deployment on PCF using CF push. (cf push is a command in CLI)

Lead the design, estimation, planning, and implementation of business solutions based on SOFEA(Service Oriented Front End Applications/**Architecture)A**rchitecture, Micro Services Architecture and Cloud Foundry Platform.

Monitoring PCF environment using Zabbix Monitoring tool and VMware Log insight.

Used the command line interface to manage apps, services in Pivotal Cloud Foundry as well as Apps manager.

Used BOSH CLI for troubleshooting PCF

RBAC(Role based access control) configuration in PCF for user roles and responsibility

**Terraform:**

* Developing Cloud formation scripts **Terraform** for leveraging different cases for **Enterprise cloud platform.**
* Managed AWS **infrastructure as code using Terraform.**
* Written **Terraform templates for configuring EC2 instances.**
* Created **Terraform scripts** to move existing on-premises applications to **cloud**
* Expertise in writing new plugins to support new functionality in **Terraform**.
* Created **Terraform recipes** for AWS data platform including different services like **ELK, Postgress, Spark , Lambda .**
* That's because under the hood, the **terraform binary makes API calls on your behalf to one or more providers,** such as Amazon Web Services (AWS), Azure, Google Cloud, DigitalOcean, OpenStack, etc. ... You then run certain Terraform commands, such as terraform apply , to deploy that infrastructure.

It is used to build the infrastructure with code… we can easily produce the required infrastructure from the terraform script with out effecting the already existing infrastructure..

Terraform can be used in any services like aws, azure, google cloud platform…. Terraform is ignorant with the underlying platform… It will just communicate with the help of providers….It will interact with different tools with the help of providers.Managed AWS infrastructure as code using Terraform.

**Jira:** It is primarily the issue tracking tool. The basic use of the tool is to track the issues, and bugs related to your project or the mobile app that is being developed.

* Installed and managed plugins for Jira in production environment such as JEMH, Script Runner, Outlook integration for JIRA, JQL Tricks, CCC Last Comment, REST API Browser, Jira Suite Utilities and other.
* Created and modified **JIRA workflows** including project workflows, field configurations, and notification schemes.
* Responsible for maintaining all Confluence Spaces and JIRA projects, workflows, permissions, users.
* Integration of Tempo Time Tracking Tools in JIRA.
* Worked as team JIRA administrator providing access, working **assigned tickets,** and teaming with project developers to test product requirements/bugs/new improvements.
* Installed various Jira plugins such as Jira client, Jira importer plugin, Jira Charting Plugin, the connector for Microsoft project.
* Worked on JIRA for tracking and updating the JIRA tickets
* Prepared projects, dashboards, reports and questions for all JIRA related services.
* Conducted analysis and evaluation of existing systems to upgrade latest versions of JIRA.
* JIRA customization - Issue Schemes, Workflow Schemes, Field Configuration Schemes, Screen Schemes, Permission Schemes, Notification Schemes.

https://www.linkedin.com/pulse/advantages-jira-charles-guo

**Confluence:**

**Ant and Maven:**

Wrote ANT Scripts to automate the build process.

Defined dependencies and plugins in Maven pom.xml for various activities and integrated Mavenwith GIT to manage and deploy project related tags.

Managed the Maven Repository using Nexus tool and used the same to share the snapshots and releases of internal projects.

Wrote ANT (build .xml) for Building Ant Projects.

Responsible for using build tools like Maven and Ant for the building of deploying the artifacts such as Jar, war & ear from source code.

Strong encounter using Jenkins for big business scale foundation arrangement and application organizations and looking at the code from GIT and utilize Ant/Maven to construct Jar, war and ear ancient rarities.

Experienced in composing pom.xml records, performing releases with the Maven release module, Mavenization of Java projects and overseeing Maven releases.

Managed nexus MAVEN repositories to download the artifacts amid the build.

Automating the build process using Subversion (SVN), ANT and Maven.  Worked on build tasks using Maven, Ant, Gradle and GNU Make files and worked with development team to migrate Ant scripts to Maven. Defined dependencies and plug-ins in Maven pom.xml for various activities and integrated Mavenwith GIT to manage and deploy project related tags

**Gradle:**

Integrate Maven and Gradle plugin scripts with Build Automation tools such AHP & Jenkins to provide automation for the build and release

Collaborated with other development teams to develop automated deployment scripts leveraging   
ANT or GRADLE build tools.

Using Gradle to create dependency JARS and deploy the entire project onto the Apache Tomcat Server.

**Git:**

Git is the distributed version control system which is used to track the changes that was made to the repository.

* git config
* git init
* git clone
* git add
* git rm
* git commit
* git status
* git branch
* git checkout
* git merge
* git reset
* git stash
* git tag
* git fetch
* git pull
* git push
* git remote
* git log
* git show
* git ls-tree
* git cat-file
* git grep
* git diff
* gitk
* Graphical Tcl/Tk based interface to a local Git repository. Example: gitk

**Subversion:**

Svn is centralized version control system whereas git is distributed

Git is more powerful and flexible with branching and merging operations etc. There are lot of other important differences too

**Jenkins:** uses master and node architecture.

* Performed integration of Code Quality Analysis Techniques like Check styles, Find Bugs, **NUnit**, **JUnit**, with CI tool **Jenkins.**
* Worked on integrating **GIT** with **Jenkins** and scheduling jobs by using **Poll SCM** and also integrated in code checkout processes.
* Developed backup jobs with **Jenkins DSL using Groovy Language** for all the CI and CD servers.
* Administered and Engineered **Jenkins** for managing weekly Build, Test and Deploy chain, GIT with Dev/Test/Prod Branching Model for weekly releases.
* Experience in setting up **Upstream** and **Downstream** **Jobs** and pipelines in **Jenkins**.
* Build and administer the Jenkins Continuous Integration servers. Build and **configure Jenkins slaves for parallel Job execution.**
* Worked in DevOps group running Jenkins in a Docker container with EC2 slaves in Amazon AWS cloud configuration
* Responsible for creating Build & Deployment scripts along with Jenkins jobs creation. and schedule nightly builds
* Configured the firewalls internally to Jenkins with acunetix.(plugin).
* Used **Github repository** to run the **Jenkins job for Continues Integration.**

**Job GeneratorPlugin**–In big or growing organizations, its a bit difficult to maintain the jobs for a project when developers are working on various branches and releases. You want to give access to developers to create their own job and at the same time you don’t want developers to create arbitrary jobs which may not fall under company standards. This plugin gives you flexibility to **define templates and developers can create new jobs with the help of job generator template**. Configuration access can be disabled via roll-based authorization plugin.

**Disable-failed-job** –In rapid development environments failing jobs due to various unnecessary reasons are a pain, It’s difficult to keep track of such jobs and disable/delete them. This plugin can solve the problem by defining the upper number of consecutive failed builds and then disable automatically.

**Embeddable-build-status** –This plugin can give you a link which can be pasted anywhere (example github project) to expose the status of the build and users can get the current state of the job while looking at project.

**Exclusion** –This plugin enables you to handle conflicts between jobs. You can assign a resource (or lock) in multiple jobs, when build is executed, it will acquire the lock and other builds (if fired) will wait until lock is released.

**GitHub/GitLab** Pull Request Builder-This is one of the best plugins I found to provide support to automate code review up to a certain extent in github/gitlab. This plugin does amazing job once defined for the project. For any new pull request this plugin does fox merge and runs the build on the code as well as gathers the necessary static analysis (if configured) or build results and comment back the status to pull request. This helps reviewers get an idea of the health of the code which is going to be merged. You can even define automatic merge if build passes.

**Hudson Extended Read Permission Plugin-** During initial days of Jenkins setting configuration access was not available to any developer to make sure they do not change anything but developers used to ask to view configurations to know how job has been setup, they may want to see the build steps. Extended read plugin can provide option to developer to view the configuration without giving them write access.

**Post+build+task** –There may be a need for performing some actions on the basis of the results of a build, for example if build passed you may want to upload artifact(ex debian) to some repo (apt) or perform some packaging part or similar. In case of failure you may want to roll back something (like release) . This plugin helps you to define the pass/fail criteria and let’s you decide what to do after that.

**JDK Parameter Plugin–**This plugin is useful for an organization where many projects are using different versions of java. This plugin let’s you choose the java version during run time of the build.

**Job Configuration History Plugin-**Ah! This is one of my favorite plugins. This plugin lets you keep track of config changes in each build including who did it. You can easily revert back to any previous config if you want.

**Multiple SCMs plugin**–Default SCM section provides only one source control tool/URL option, what if you want to check out from more than 2 repos from multiple source control tools (like svn and git). This plugin will come in handy in such scenarios. This plugin will facilitate users to add any number of SCM URLs to checkout the code.

**Parameterized Trigger plugin**– Another one of my favorite plugins. This plugin allows you to have user input as a variable and use on run time. This is the most used plugin in dynamic environments where you have lots of options and user-defined values to be used in the build which may keep changing.

**Pre SCM BuildStep Plugin**- Just like post build task plugin, you may have requirements to perform some action even before checkout happens for the job, for example you may want to perform merging of the branch before the build and then checkout. This plugin can come in handy in various conditions and gives you flexibility when running the job.

**SCM Sync Configuration Plugin**-backup, this is the most important task of any administrator. Without regular backup, the whole system cannot be reliable; this plugin provides you features to backup live Jenkins configs to any source control tool. It will keep committing the config files (including Jenkins and jobs) to SCM repository as soon as there is any change.

**Configuration Slicing Plugin** – This plugin comes  in very handy when you want to make bulk changes in multiple jobs. This plugins allows you to change values of various fields like email, timer, shell script, configurations etc at one go.

**Kubernetes:**

* Created Clusters using **Kubernetes** **kubectl** and worked on creating many **pods, replication controllers, services, deployments, labels, health checks and ingress** by writing **Yaml** files.
* Used Kubernetes to manage containerized applications using its nodes, ConfigMaps, selector, Services and deployed application containers as Pods.
* Implemented **Jenkins** and built pipelines to drive all **microservice** builds out to the **Docker registry** and then deployed to **Kubernetes.**
* Executed a Kubernetes POC(proof 0f concept) to demonstrate the technical viability of container orchestration.
* Managed Kubernetes charts using Helm. Created reproducible builds of the Kubernetes applications, managed Kubernetes manifest files and managed releases of Helm packages.
* Worked with containerization tools, can implement transition to Docker and develop distributed cloud system using Kubernetes.
* Handled large volumes of containers with Docker Swarm, Kubernetes, and Mesos.

**Bamboo:**

**Nagios**

**Splunk**

**Docker**

Docker containers can be used to run the applications or services that we are trying to test as part of our CI pipeliene. We write scripts that would create containers on Jenkins slaves, deploy and run the build created during build on them

* Implemented build stage- to build the microservice and push the docker container image to the private docker registry.
* Worked on end to end setup of the artifactory pro as a docker container with a secure private dockerregistry and local docker repositories for storing the built docker images.
* Worked on Ansible playbooks to automate the installation of docker-engine, docker swarm cluster.
* Responsible for Containerizing using Docker, Docker Compose, Swarm & Kubernetes.

**Dockerswarm:** Current versions of Docker include *swarm mode* for natively managing a cluster of Docker Engines called a *swarm*.

* Worked to setup Jenkins as a service inside the docker swarm cluster to reduce the failover downtime to mintues and to automate the docker containers deployment without using configuration management tool.
* Worked on deployment automation of all the microservices to pull image from the private dockerregistry and deploy to docker swarm cluster using Ansible.
* This tutorial introduces you to the features of Docker Engine Swarm mode. You may want to familiarize yourself with the [key concepts](https://docs.docker.com/engine/swarm/key-concepts/) before you begin.
* The tutorial guides you through the following activities:
* initializing a cluster of Docker Engines in swarm mode
* adding nodes to the swarm
* deploying application services to the swarm
* managing the swarm once you have everything running
* Integrated manomarks docker-swarm Visualizer to the swarm cluster.
* • Worked on shell scripts to integrate Prometheus, AlertManager, Grafana, NodeExplorer, cAdvisor to monitor the docker swarm cluster metrics from host to container level.
* Migrated Mesos/Marathon to docker swarm for the microservices project.

**Docker compose:** Compose is a tool for defining and running multi-container Docker applications. With Compose, you use a Compose file to configure your application’s services. Then, using a single command, you create and start all the services from your configuration.

* Using Compose is basically a three-step process.
* Define your app’s environment with a Dockerfile so it can be reproduced anywhere.
* Define the services that make up your app in docker-compose.yml so they can be run together in an isolated environment.
* Lastly, run docker-compose up and Compose will start and run your entire app.
* Compose has commands for managing the whole lifecycle of your application:
* Start, stop and rebuild services
* View the status of running services
* Stream the log output of running services
* Run a one-off command on a service

Google compute engine

Chef: configuration management tool written in Ruby and Ruby/Erlang. Chef uses workstation server and client architecture. Chef is a powerful automation platform that transforms infrastructure into code. Whether you’re operating in the cloud, on-premises, or in a hybrid environment, Chef automates how infrastructure is configured, deployed, and managed across your network, no matter its size.

Chef is an automation tool that provides a way to define infrastructure as code. Infrastructure as code (IAC) simply means that managing infrastructure by writing code (Automating infrastructure) rather than using manual processes. It can also be termed as programmable infrastructure. Chef uses a pure-Ruby, domain-specific language (DSL) for writing system configurations. Below are the types of automation done by Chef, irrespective of the size of infrastructure:

* Infrastructure configuration
* Application deployment
* Configurations are managed across your network

In Chef, Nodes are dynamically updated with the configurations in the Server. This is called **Pull Configuration** which means that we don’t need to execute even a single command on the Chef server to push the configuration on the nodes, nodes will automatically update themselves with the configurations present in the Server. My next blog on **Chef Tutorial** will explain the Chef architecture along with all the Chef components in detail.

Setup and Install Chef workstation, Chef Server and bootstrapping the Chef Clients using knife.   
• Configure Knife to manage chef-server, spin-up new instances, package installs and config changes on Client node.   
• Developing Cookbooks, Recipes, Resources and Run lists, managing the Chef Client nodes, and upload the Cookbooks to chef-server from Workstation.   
• Manage Chef Server in pushing the config changes to client nodes, monitoring the logs and security.

Enforcing Test Driven Development (TDD) approach for writing Chef Cookbook using Test-kitchen, Vagrant, Berkshelf(**Berkshelf** is a bundler-like dependency manager for Chef cookbooks.), FoodCritic and ChefSpec.

Berkshelf lets you treat your cookbooks the way you treat gems in a Ruby project. When external cookbooks are used, Berkshelf doesn’t require “knife cookbook site” to install community cookbooks. All we have to do is mention the dependent cookbooks with its version number. When Chef client runs on nodes, berkshelf will automatically download and install all the dependent cookbooks from the Opscode cookbook community for us.

https://cloudacademy.com/blog/berkshelf-manage-chef-cookbooks/

https://docs.chef.io/chef\_overview.html

Puppet

Developed Puppet manifests for different application and web servers like Apache, Tomcat, Nginx, WebSphere, WebLogic applications. It uses puppet master and puppent agent architecture.

Deployed web applications using puppet by developing the manifests to meet the project   
requirements.

Used Puppet server and workstation to manage and configure nodes.

Extensively worked in building Puppet enterprise modules using puppet DSL to automate infrastructure provisioning and configuration automation across the environments.

Created Puppet manifests, classes and modules to automate system operations.

Supported Puppet master server and around 500+ Clients by installing agents and managed multiple modules and manifests.

Deployed Puppet, Puppet Dashboard and Puppet DB for configuration management to existing infrastructure.

Deployed the major components of the puppet infrastructure like puppet server, facter, hiera, MCollective and puppetDB

Deployment automation using Puppet along with Hiera (YAML) Data on Mcollective orchestration engines and writing manifests and modules for different micro services.

Implemented Puppet hiera to separate data from puppet code.

Worked on open source modules for puppet from puppet forge.

Ansible

Worked on Ansible playbooks to automate the installation of docker-engine, docker swarm cluster.

Developed Ansible Configuration Management scripts and playbooks.

Configured the Ansible Tower by writing Ansible playbooks, so clients can run them easily by just a push button.

Worked with Ansible playbooks for orchestration, provisioning, configuration management and application development of LAMP( Linux, Apache, MySQL, and PHP/Python/Perl) stack.

Wrote Ansible playbooks for automated uDeploy agent install/updates, udeploy agent relays install/update and configuring the same with the uDeploy server.

Wrote playbooks in jSON language.

Fine tuned the tasks with the help of playbook tags

With the rich plugin infrastructure extended the ansible services with plugins like dynamic inventory sources, callback plugins and custom filter plugins and lookup plugins

Code commit

Bugzilla

Tomcat

Nginx

Apache

Jboss

WebLogic

Veritas

Linux

CLOUD FORMATION: This is only used to produce the infrastructure from the code in the aws environment only…

ELK

Kafka

Zabix

Nginix

Rabitmq

**What are the differences between github and gitlab?**

<https://usersnap.com/blog/gitlab-github/>

what are different types of clouds and

private ,public, hybrid clouds available

Microsoft OMS

What is urban code deploy servers and udelploy agents

What is your experience Application Servers?

My experience is mostly around Deployment and running apps in various Application Servers. I have used Web Logic, Web Sphere, JBoss etc. I wrote various scripts using various scripting interfaces like WLST(Web Logic), Jython(Web Sphere) to run applications in standalone or clustered mode. As part of deployment, create more resources like JDBS, JMS, JMX etc for applications etc. I have also used Web Servers like Apache, Tomcat, Nginx etc.

**How to migrate SVN to Git**

Make svn repos as read only, create new git repositories and move all files to git. There are many issues with git svn plugin etc, so we chose this path.

**How can we migrate Ant to Maven**?

There is no standard tool to migrate, we have to re-write maven build scripts from scratch.

**How can we migrate shell scripts to recipes ?**

It is a bad practice to migrate shell scripts in entirety into recipes or manifests.

We have to re-write all logic to use Ruby DSL of Chef or puppet syntax

All cookbooks and manifests are idempotent, meaning, we can re-run them any number of times and they all start from clean state each time. Whereas, shell scripts are not like that.

If the shell creates script files and directories, we have to write the same by taking file and directory resources in our recipes etc

**Difference between Chef, Puppet with Ansible.**

Chef and Puppet are Server/Client model, whereas, Ansible has no central server

Chef and Puppet have their own custom DSL/syntax whereas Ansible is Yaml and SSH based.

Ansible plays better with shell scripts than Chef and Puppet

Most companies are using Ansible for doing deployments and using Ansible WITH Chef or Puppet.

Are you using same build artifact through out process ?

The same build gets promoted to multiple stages based on its quality. If we build is good, then it deployed from Dev, QA, Stage and then Prod etc. If it is based, developer will check-in fix and a new build gets created and deployed, CD Process repeats.

Difference between Nexus and Artifactory?

Both are Artifact Repository managers. Built by different companies

Nexus by Sonatype and Artifactory by JFrog

Artifactory has better REST Api than Nexus. Open source does not provide REST API. We have to but Enterprise Nexus, then we get REST API and other features like Staged Repositories etc.

Difference between ANT and Maven

Both are build tools

Ant is procedural and Maven is declarative.

Both use xml based build files etc

Maven has automatic Dependency resolution which is manual in ant.

Maven can pull jars from nexus or artifactory etc during a build.

Maven has more plugins to integrate with other tools

How is NAT different from proxy server?

NAT is typically used for providing internet access to VMs. It is also used to create virtual network interfaces on the hypervisor to provide access to VMs running on it.

Difference between VMWare, KVM and Xen

VMWare provide licensed tools to create and manage VMs on bare metal servers. KVM and Xen do the same, but, are open source.

More info:

<https://devopshero.wordpress.com/2013/10/17/kvm-as-your-virtualization-provider/>

What do you know about VMotion? Fusion etc?

<https://www.vmware.com/products/vsphere/features/vmotion>

<https://www.vmware.com/products/fusion>

Difference between Vagrant and Docker?

Docker is Linux based technology. Whereas, Vagrant provides command line utilities to create normal VMs. They can be used by Developers to create san box envs in their local, but, technology implementation wise, they are very different.

How many instances can a Load Balancer manage the traffic?

LBs are very scalable and reliable and can scale up to serve 1000’s of servers in backend and millions of user requests.

Give some real time trouble shoot experience you have done.