## INTRO:

Hi this is Harsha I have around 5 years of experience in IT industry as DevOps Engineer.

I extensively work with Cloud Platform AWS to Configure and maintain Cloud Infrastructure and various DevOps tools like GIT, Jenkins, Ansible, Docker, Terraform, Kubernetes to handle continuous integrations, continuous deployment, configuration management, containerization and Infrastructure as Code.

## AZURE – 1st Project

My day-to-day responsibilities involve:

Working 70% on cloud customizing and private sharing centers that are on AZURE which up to 1000 instances.

Working with deploying docker instances on AZURE platform.

In my current project we are team of 8 people. I am one of the lead devops engineer and a cloud architect. we have 3 other developer 1 QA, 1 release manager and 1 product owner.

To advance microservice adaption, our Azure team had to clearly describe the characteristics of our microservices, how they compared with existing APIs, and had to develop several POC (Proof of Concept) to showcase the value proposition.

Dealt with storages like Blob (Page and Block), SQL Azure for making it possible for every microservice to have its own database.

Azure Resource Manager template – JSON to define one or more resources to deploy to a resource group.

I was responsible for adding accounts and resource partners, mapping partner claims, adding and configuring account stores, and identifying and configuring applications using SSO and multifactor authentication and recently started.

I have good experience in working with Kubernetes for docker containerization and deploying Kubernetes on centOS7 platform.

In my current project I am working on continuous integration, orchestration tools. primarily working with chef, puppet for configuration management tools and wrote many custom cookbooks using ruby scripts. And now, working with Ansible for configuration management and have experience in writing Playbooks.

Also has good experience in using troposphere for creating custom cloud formation templates. Also experienced in writing Terraform scripts to Manage the Infrastructure as Code and handle the configuration of Infrastructure through Terraform.

Lastly git is the version control we are using, continuous integration via jenkins.

Monitoring vise i have good experience using splunk and ELK. 60% on windows and 40 % on linux environments.

Have good knowledge of azure various paas , iaas and Saas services provided by azure. Worked in two different modules which are azure resource manager and azure service management.

Created virtual machines, created serverless API using azure functions, Azure service fabric to deploy, manage scalable microservices and containers, blobs for storage and mainly responsible for connecting azure to on-premise data center using express route.

And built resource groups in windows azure and restricted access to resource groups using role-based access controls. To automate the infrastructure used azure CLI and involved in accessing subscription using powershell. To monitor and troubleshoot azure resources, I have used azure app insights.

## AWS – 2nd Project

When coming to AWS

I have hands on experience on AWS, various Platform as a Service components and Services provided by AWS.

I Architected Applications for High Availability and fault tolerant. Implemented Security practices in AWS.

Experience in migrating various data center applications, services and databases to cloud. Designed Capacity planning and Auto scaling etc.

I have used the boto3 sdk of aws - it’s a python based scripting for writing all Automation.

Experience in using various services include standard services like EC2, S3, EBS, SNS, SQS, Cloud Formation, Cloud Front, RDS, DynamoDB, Cloud Watch, ELB, Auto scaling to more recent services like Lambda. I have used AWS CLI as well.

**project2 on AWS**

Coming to My day to day responsibilities involves –

Working 70% on cloud customizing and private sharing centers that are on aws up to 1000 instances.

I was Involved in Designing, installing, automating, administering, and optimize AWS solutions and components to ensure business continuity.

Built and configured a virtual data center in the AWS cloud to support Enterprise Data Warehouse.

Involved in Designing and Deploying multitude of applications utilizing almost all the AWS Stack including EC2, Route 53, Cloud Formation, RedShift, Cloud Front, Cloud watch, IAM, DynamoDB with NoSQL.

Launched Ec2 instances by using amazon machine images and configured them with custom operating systems and packages.

Involved in developing python script in cloud formation templates to configure auto scaling and elastic load balancing, automating the process.

Worked on Docker container services and implemented Kubernetes for orchestration of the containers.

Implemented Master-minion’s architecture. created many clusters using Kubernetes. And I worked on creating many pods. I also wrote Yaml files to creates pods, replication controllers, services, deployments, labels, health checks, ingress.

I managed the load balances using ingress. And I worked on our company Aws docker containers using kops. I have knowledge on web UI of Kubernetes, but I prefer command line interface than webUI.

Designed Jenkins build pipelines for automation and continuous integration. And built Jenkins Clusters for High Availability, familiar with server-side things like backups, integrate Jenkins with various tools like Jira, testing frameworks like Junit, Mockito, SoapUI, Cucumber, Code Quality tools like Sonar, HP Fortify etc.

Implemented Chef for Infrastructure as a code or configuration management and deployment automation on application builds using client-server architecture.

I worked with Chef and Puppet for Configuration Management and I have written many Cookbooks and Recipes that would do from simple things like create files, directories, users and passwords, env variables etc to more advanced things like Deployments and Automation etc.

I have thorough understanding of Roles, Environments, Data bags, Encryption with Vault, Chef Client and Server integration, Attributes, ERB templates etc.

Implemented Nagios and Splunk monitoring solution for mission critical servers. Integrated Splunk with AWS deployment using Puppet to collect data from all database server systems into Splunk.

## GIT:

Git is a free, open source distributed version control system tool designed to handle everything from small to very large projects with speed and efficiency.

Git has the functionality, performance, security, and flexibility that most teams and individual developers need.

I maintained GIT source code repository and local mirrors, performed branching, tagging, merging and maintenance tasks for windows host and Mac builds.

I Coordinate/assist developers with establishing and applying appropriate branching, labeling/naming conventions using GIT source control.

Configured and administered GitHub enterprise, also I handled migrations from SVN (Subversion) to GitHub.

## Maven

Experienced in invoking maven top level goals such as compile, test, package, deploy, install and clean.

Configured POM file to manage plugins such as build plugins and Reporting plugins and resolve dependencies required for project.

Defined Profiles in POM file to provide different build results to different environments such as development and testing.

Created maven projects with different archetypes such as quickstart and webapp. Maven is a build automation framework provided by apache.

It is used for continuous deployment, building and merging Java projects. · Used maven to build java projects and store them in XML files.

Create artifacts like jar, war and ear from source code in the java projects.

Defined project related dependencies and plugins in maven pom.xml files.

Integrated maven with GIT to deploy project related tags. Integrated Maven with Jenkins and Hudson to schedule the builds.

Maven Managed project dependencies in maven by creating parent child relationships among projects.

Installed Ant build tool and specified dependency jars in Ant path required for running build process. Defined Ant targets, tasks such as Archive tasks, file tasks and properties in project for performing build process. Monitored Ant build process using Listeners such as Log4JListener and Loggers such as Default Logger, Mail Logger.

## Jenkins

Jenkins is an open source automation tool written in Java with plugins built for Continuous Integration purpose.

Jenkins is used to build and test your software projects continuously making it easier for developers to integrate changes to the project and making it easier for users to obtain a fresh build.

It also allows you to continuously deliver your software by integrating with many testing and deployment technologies.

In Jenkins, I architected build pipelines to retrieve code, compile applications, perform tests and push build artifacts to Nexus and implemented Master/Slave architecture.

I am experienced in maintaining CI environments with build automation tools like Jenkins. I extensively used Jenkins to streamline CI/CD process.

I configured Jenkins jobs for building the Java based code and deployed it to various test servers and production environments.

Through Jenkins, we automated webserver content deployments via shell scripts, we also migrated to Jenkins from Hudson for Continuous Integration and deployment into Tomcat Application Server.

## Ansible

* I have pretty good experience using Ansible, through which I got to write many playbooks, modules and roles using YAML script and used them in AWS environments.
* I am experienced in configuring the Ansible Tower to automate repetitive tasks and quick deployments for the critical applications, also to manage Multiple Nodes and Manage Inventory for different Environments.
* Implemented SSH Keys to create password-less SSH connection on various servers. Basically, I used Ansible to provide infrastructure on servers, installed packages, required SDKs, etc. saltstack
* It IS a Quickly scalable, very resilient and efficient because of multi-master capability I Used minions because it offers more options and flexibility than Ansible There is no GUI for it currently under development) i have used Python and PyDSL for scripting.
* We have used Minions for efficient communication for large scale deployment Minions not as efficient as agent-less communication for small-scale deployment

## Chef & Puppet

Chef Client - Version 13.3.42 Chef Client - Version 12.16.9 Chef is a Configuration Management tool i used it mainly to manage Infrastructure as Code (IaC).

With this automation platform that configures and manages the infrastructure whether it is on-premises or in the cloud. It turns your infrastructure as code i.e. your computing environment has some of the same attributes as your application.

Your infrastructure is version-able, repeatable, and testable. You only need to tell Chef what the desired configuration should be, not how to achieve it. Like Puppet which has a Master-Slave architecture, even Chef has a Client-Server architecture. But Chef has an extra component called Workstation.

I have had an interesting work experience with Chef. I used it when I was first introduced to DevOps.

It was where I worked on creating cookbooks and authored recipes to provision the servers with required infrastructure.

I configured, maintained, and supported Chef environment on 1000+ Servers. I composed several Chef Cookbooks and Recipes to provision several pre-prod environments.

Launched fully Chef configured and build management system to deploy servers with the proper configuration on a per role and per environment basis.

**Puppet**

Puppet version 4.1 puppet is used as a configuration management tool which runs on Linux and Unix system.

It includes a custom declarative language that is used to describe system configuration and their state. It simplifies various system administration tasks.

I have around 5 years of experience using Puppet under configuration management for deploying, configuring and managing servers.

I have good experience with Puppet’s objects like Puppet resource, Puppet Class, Puppet Manifests, Puppet Modules, Puppet Forge.

We used Puppet for system configuration where we updated configuration from the master server on .pp file and pushed it to multiple nodes or clients.

I was involved in deploying web applications using puppet by developing the manifests to meet the project requirements.

The information was pulled and updated and had patch cycles a couple of time in a week. We made sure that the configuration along with all the environments (dev, test, prod) is the same and that there are not any technical issues.

## Docker

* I am very familiar and hands-on with Docker. I installed and configured a private docker register for our dev teams to push and pull containers.
* Experience in creating **DockerContainers**leveraging existing Linux Containers and AMI's in addition to creating **Docker Containers**from scratch.
* Containerized all the Ticketing related applications - **SpringBoot Java and Node.Js**applications using Docker.
* Familiar with all docker commands, wrote docker files to run applications, familiar with other advanced tools like Compose, Swarm, Weave. Container scheduling frameworks like Kubernetes(K8) etc.
* I have experience in Container management using Docker by writing **Dockerfiles**and set up the automated build on Docker HUB and installed and configured Kubernetes.
* Based on Applications their architecture and type, we can write docker files to create and run applications in docker containers.
* I have used Docker Compose/Swarm to create and run Cluster of containers, use Weave to configure Networking between them and deploy a n-tier application on them.
* Worked on Port binding, Docker Network Drivers – (Bridge Network, User Def Network, Host Network etc.)

## Kubernetes

* I have good hands on experience with installing and configuration of Kubernetes and cluster them.
* I have integrated Kubernetes with n/w (network), storage & security to provide a comprehensive infrastructure. and orchestrated the Kubernetes containers across the multiple hosts.
* 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. To spin up a cluster in aws, we use KOPS (Kubernetes Operations). We use Minicube - it runs a single node kubernetes cluster inside a linux VM's.

## Terraform:

* Experienced in writing terraform templates that can spin up infrastructure for multi-tier applications.
* Created and provisioned boot strapped software demo on cloud with terraform and deployed infrastructure in multi cloud environment to deploy a fault tolerant application.
* Terraform works just like cloud formation but it is available for all cloud platforms and not designed to a specific cloud platform like cloud formation to AWS.
* I worked with terraform to create templates for environments using HCL scripting (Hashi Crop configuration language).
* Using terraform to automate the deployment process onto azure.
* Converted cloud formation JSON templates to terraform.
* Also performed AWS infrastructure automations with terraform.
* Worked with Terraform Dynamic Blocks (Ex – Security Groups dynamic block for each port.)
* Handled Terraform Provisioners (Local exec and Remote exec)
* Worked with Terraform Registry in utilizing the modules which are readily available to download and use.
* Developed Sentinel Policies to handle any fault spinning of Infrastructure.

## Open Stack – Cloud Platform

Used OpenStack APIs to launch server instances, created images, assign metadata to instances and images, create storage containers and objects, and complete other actions in OpenStack cloud.

I have used OpenStack in my previous project for deploying an application which was based on IAAS.

Built this application based on Python and MySQL.

I have deployed the application into open stack and integrated it with chef repos. Implemented automated local user provisioning in instances created in Openstack cloud.

Implemented horizontal auto-scaling using OpenStack components such as Heat, Ceilometer.

Experienced in working with OpenStack object store Swift to store lots of data efficiently, safely, and cheaply and storing unstructured data that can grow without bound.

Delivered Network-as-a-service in virtual compute environments using OpenStack Neutron.

Experience in working with Openstack Glance Rest API for getting VM image metadata and retrieval of actual image.

Experience in working with Openstack services such as Novaand glance, Swift, Keystone, Heatand Ceilometer.

## open shift- Container Application Platform

## open source container application platform based on the Kubernetes container orchestrator for enterprise app development

Build Docker images; create build and deployment configurations to deploy applications on Openshift.

Patch, upgrade and scale Openshift environment.

Use of Docker and Openshift to manage micro services for development and testing.

## SonarQube - Inspection

Worked with SonarQube for continuous inspection applications in our environment and detect newly raised issues like bugs and security issues.

In my project I checked in the application code to GITHUB and used SonarQube for continuous inspection by linking SonarQube to NetBeans Platform.

## Splunk, ELK Nagios - Monitoring

Monitoring tool. Performs searching, analysis, has a dashboard, based on saas and on premises setup. Has plugins.

It has customer support and does documentation. ELK Searching analysis and dashboard is only possible by integration.

In our company we use elk to process data from log files coming from different applications, basically we are using Ubuntu server for java applications.

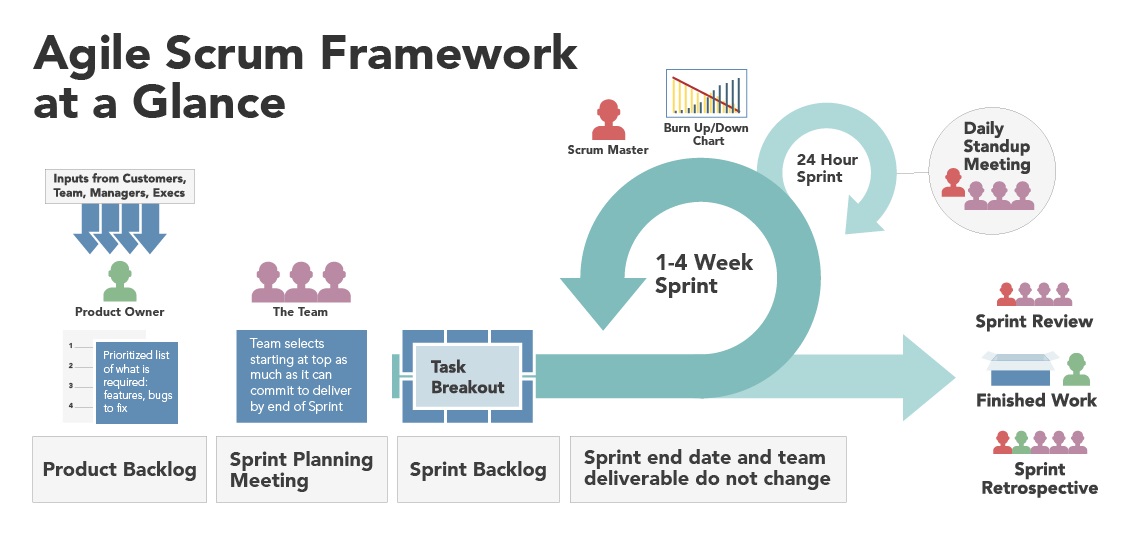
A cluster is identified by a unique name which by default is "Elasticsearch". This name is important because a node can only be part of a cluster if the node is set up to join the cluster by its name.

We always make sure that we don’t reuse the same cluster names in different environments, otherwise we might end up with nodes joining the wrong cluster using the ELK stack to monitor the logs from some Spring Boot’s microservices, backup, monitor and configure another aspect like security on a cluster.

(In elk, created yaml files and used azure resource management templates, high level architecture storage, pod on nodes, cluster, need a data node where we can hold one or more data client node will handle incoming request and master node perform cluster management and operation side like maintaining and distributing the information around cluster connect to cluster with api, memory requirement).

## Agile

Agile software development refers to a group of software development methodologies based on iterative development, where requirements and solutions evolve through collaboration between self organizing cross-functional teams.



## Nexus - Repo

Nexus is a repository manager. It allows you to proxy, collect, and manage your dependencies so that you are not constantly juggling a collection of JARs. It makes it easy to distribute your software.

 In Jenkins, I architected build pipelines to retrieve code, compile applications, perform tests and push build artifacts to Nexus and implemented Master/Slave architecture.

We did standardized pipeline with SDLC process and orchestration through Jenkins, we also used Nexus server for artifacts and wrote scripts for deploying. I have done deployments in production environment through Blue-Green Deployment technique to decrease risks and downtime.

**Git**) is a source referential for version control (with features like merging, branching, tags) the other (**Nexus**) is an artifact referential for any delivery (binaries or not).

## Sonatype - Security

The **Sonatype** Cloud Extension provides the most precise intelligence regarding security vulnerabilities, license risk, and architectural quality of OSS components and delivers that information directly within **Azure DevOps** so you can remediate violations and keep your software secure.

## Azure ARM

**Azure Resource Manager** is the deployment and management service for **Azure**. It provides a management layer that enables you to create, update, and delete resources in your **Azure** account. You use management features, like access control, locks, and tags, to secure and organize your resources after deployment.

## Testing – Junit, SoapUI, Cucumber, Mockito, SOnar

* Testing frameworks like JUnit, Mockito, SoapUI, Cucumber, Code Quality tools like Sonar, HP Fortify etc.
* Created Test Automation Framework with **Cucumber** and **Selenium WebDriver.**
* Created the **Feature** **Files** for Cucumber Automation.
* Configured **Cucumber Report Plugin** and **Performance Plugin** to Jenkins to generate html test reports.
* Performed extensive **Web Services Testing** using **soapUI Pro**.
* Tested components using **JUNIT Mockito** during development.
* Expertise in using **JUNIT, TestNG, Maven** to build automation frameworks with **JAVA**.
* Expertise in JUNIT testing using Mockito and Power Mockito, integration testing and SOAP UI testing.

## Scripting – python, Bash, Shell

I have used the boto3 sdk of aws it’s a python based scripting for writing all Automation.

And authored python script in could formation templates to configure auto scaling and elastic load balancing, automating the process.

I have used OpenStack in my previous project for deploying an application which was based on IAAS. Built this application based on Python and MySQL.

I wrote various scripts using various scripting interfaces like WLST (Web Logic), Python scripting on Web Sphere to run applications in standalone or clustered mode. – For Application Servers.

To automate the infrastructure used azure CLI and involved in accessing subscription using powershell.

Through Jenkins, we automated webserver content deployments via shell scripts, we also migrated to Jenkins from Hudson for Continuous Integration and deployment into Tomcat Application Server.

## Coding - JAVA

I have written custom plugins in Jenkins and Sonar etc utilizing REST APIs and plugin interfaces. In general, I am familiar with java coding but not strong in it. I have experience doing DevOps, Build, Release of Java based applications.

I configured Jenkins jobs for building the Java based code and deployed it to various test servers and production environments.

Used maven to build java projects and store them in XML files. · Create artifacts like jar, war and ear from source code in the java projects.

Experience in core java concepts like Collection Framework, Multi-threading, Generics, Annotations, Serialization, Thread pools, JavaBeans, Externalization.

Involved in the design and development of application built in Java/J2EE using Struts, **Spring** and **Hibernate**.

## Linux

* With my experience as a Linux administrator and a system administrator, I am capable of handling admin level operations and duties on Linux. I worked with Linux distributions like RedHat Linux, CentOS, Ubuntu, Debian, SUSE and Solaris.
* Performed regular heath checks, Taken backups, maintained firewalls and monitored Linux systems using Nagios software. · I troubleshooted networking issues, implemented protocols and provided 24x7 on-call support for Linux servers.
* Installed RedHat Linux servers locally and over networks using Kickstart. · Setup Samba servers for various Linux systems and enabled communication of windows client with Linux system.
* I have worked with VMware virtualization systems and migrated applications from Linux to windows.
* I have performed LVM to manage volumes, expanded file system spaces and created NFS mounts.
* Python, NodeJS, C#, or other back-end languages
* Exposure to a front end MVVM or MVC framework (React.js, Durandal, KnockoutJS or similar)
* Exposure to databases (relational and NoSQL)
* Exposure to RESTful APIs
* Knowledge of HTML and CSS

Spark Hadoop.

Knowledge of reindexing and data mappings

## ELK-

* Responsible to designing and deploying new ELK clusters (Elasticsearch, Logstash, Kibana, beats, Kafka, zookeeper etc.
* Experience in configuring Logstash and other ELK Stack components to collect and the store the data necessary to meet business requirements.
* Skilled in monitoring servers using Nagios, Data dog, Cloud watch and using EFK Stack Elasticsearch Fluentd Kibana.
* Experienced in setting up Elastic Search 1.7 in an AWS Cloud Environment.
* Experienced in providing recommendations related to Cluster sizing and other related querying for ElasticSearch.
* Implemented a POC to integrate ELK Stack to an existing appliance framework for real time log aggregation, analysis and querying (Elasticsearch, Logstash, Kibana).
* Design, build and manage the ELK (Elasticsearch, Logstash, and Kibana) cluster for centralized logging and search functionalities for the App.
* Responsible for designing and deploying new ELK clusters (Elasticsearch, Logstash, Kibana, beats, Kafka, zookeeper etc.
* Written and Maintained Automated Salt scripts for Elasticsearch, Logstash, Kibana, and Beats.
* Experience in Kibana in data visualization and monitoring Cluster and performance through X-pack.
* Experience in building data dictionary, functions and synonyms for NoSQL (Elasticsearch).
* Created stored procedures to import data in to Elasticsearch engine.
* Worked closely with application teams and support on various performance and configuration issues on daily basis.

## Elastic Cloud Enterprise (ECE)

Using Elastic Cloud Enterprise, we centrally orchestrate the Elasticsearch Clusters.

Experienced in Provisioning, Managing, Monitoring Elasticsearch and Kibana at any scale according to the Client requirement while managing everything from a single console.

Deploy Elasticsearch and Kibana on Kubernetes with Elastic Cloud on Kubernetes.

Using Elastic Cloud Enterprise, I have experience in running Elasticsearch and Kibana in any env like Public Cloud (AWS, Azure, GCP), private cloud, Virtual Environment and even physical hardware.

Experienced in Role Bases Access Control (RBAC).

Experience in handling Streamline scaling, securing, upgrading, and backing up all the Elastic deployments from a centralized console.

Automated tasks using the ECE REST API and Elastic Cloud Control CLI (command line interface).

Experienced in configuring security features like authentication, role-based access control, encryption, and SAML using ECE (Elastic Cloud Enterprise).

Coming to Backup - Clusters that are managed by ECE can be periodically backed up using the Elasticsearch snapshot API.

Experienced in handling **Index lifecycle management** to automate actions in your data lifecycle policies like merging, shrinking, or deleting indices.

Worked with different stages of Index Lifecycle Management like hot, warm, cold, delete:

* **Hot**: indices in hot stage are sensitive indices used for data ingestion and usually respect a strong SLA. Ingest and search performance of hots indices are important.
* **Warm**: indices in warm stage are usually not used for ingestion anymore but they’re still likely to be queried, albeit less often than hot indices.
* **Cold**: indices in cold stage are unlikely to be queried but it’s too early to consider the data to be deleted; for example, in the case of security monitoring, we might still need old data.
* **Delete**: indices in the delete stage are about to be deleted. So simply.

## Kibana

Kibana is an open source data visualization dashboard for Elasticsearch. It provides visualization capabilities on top of the content indexed on an Elasticsearch cluster. Users can create bar, line and scatter plots, or pie charts and maps on top of large volumes of data.

## Logstash

* Logstash can collect data from different sources and send to multiple destinations.
* Logstash can handle all types of logging data like Apache Logs, Windows Event Logs, Data over Network Protocols, Data from Standard Input and many more.
* Logstash can also handle http requests and response data.
* Logstash provides a variety of filters, which helps the user to find more meaning in the data by parsing and transforming it.
* Logstash can also be used for handling sensors data in internet of things.
* Logstash is open source and available under the Apache license version 2.0.

## Beats

Beats are lightweight data shippers that you install as agents on your servers to send specific types of operational data to Elasticsearch. Beats have a small footprint and use fewer system resources than Logstash.

Logstash has a larger footprint, but provides a broad array of input, filter, and output plugins for collecting, enriching, and transforming data from a variety of sources.

## Kafka

**Kafka** is often **used** for operational monitoring data. This involves aggregating statistics from distributed applications to produce centralized feeds of operational data.

The **Kafka** Connect **Elasticsearch** Service sink connector moves data from Apache **Kafka**® to **Elasticsearch**. It writes data from a topic in Apache **Kafka**® to an index in **Elasticsearch**. All data for a topic have the same type in **Elasticsearch**. This allows an independent evolution of schemas for data from different topics.

## Zookeeper

ZooKeeper is used to manage the nodes, configurations, and states in the distributed system and complete the configurations and state synchronization among individual nodes. Many distributed systems rely on ZooKeeper or similar components.

ZooKeeper manages data in the form of a directory tree; each node is referred to as a znode.

ZooKeeper provides watch functionality that can be used to listen to corresponding events, such as the increase/decrease of a child node under a znode, the increase/decrease of a znode, and the update of a znode.