Real Time Data Streaming with Apache Spark

Arjun Yadav (309939)
Bhomit Kapoor (310007)
Drashti Pandya (310112)
Karan Sethi (309984)

Appendix

Abstract

In this report we try to exhibit an effective way for a highly flexible streaming platform that facilitates multiple, key applicative scenarios in modern data planning. Out of all, this could be important for Kafka that is scalable deployable into real-time based data frameworks. Tides of unbounded Information are productively consumed, reserved, and transferred using a framework called Kafka. It supports large volumes, great velocities of data from various sources such as feeds from Social Media, sources of Internet of Things (IoT), and database undertaking are read, duplicated and write down in instantaneously on-time. Our analysis and research depicts the how the Capturing the Changes in data and replication technologies with database controls with Kafka streams and comparison of it with other streaming technologies. We have planned it accordingly to implement with docker and Scalable pods of Kubernetes. The reason we use Kafka streams is as the modern world has a tremendous amount of data to work on where traditional ways do not find their way.

In this report article, we provide an overview of the in-depth analysis of the latest recent and influential technologies related to cloud and alteration, focused on streaming large information processing in cloud services for lower randomness and responsive comparison of the most used architectures. We try to describe their importance underlying features they facilitates in the streaming of task competitively.

Keywords: Distributed real-time streaming, cloud processing, Real-time systems, Cloud Computing.

Introduction

Over the recent era of Human Life, the volumetric amount of information being created has skyrocketed as more than 50k gigabytes of data is formed in every unit of time, and the rate at which this data is created only accelerates with years exponentially. All of this data dealt is diverse in an indefinite illustration of the extensively complex set of properties and features with many aspects to drive out all related with data All this leads to the techniques now being collectively over amounts called Big Data. Many technologies create massive amounts of data that entails a rise in volume to changes and analyze these data as well. Large data systems have been mined to allow the insights which were earlier dependent on the investigation, surveys, and development to conclude inferences and predict the rules for new ones. “Big data” is an overall term for data sets that are not only large but also could be complex that process traditional data use in inadequate ways. Analysis, capture, curation of data, search, sharing, transfer , visualization, storage, and data privacy are the cruciality performed. It refers to the simpler application of predictive analytics and methods to acquire the values from data, rare to a given data set. Accuracy of this data is responsible for decidedness, better decisions gives better cost, less risk operational efficiency.

Attributes of Big Data: To use the exact techniques with a huge volume of data acquired from various sources we must understand the features of Information perceived.

•**Volume**: the sheer quantity of data present for the analysis. It is derived from the increase in the number of dataset instruments for example – Tools of social media, Mobile applications, Sensors, etc. This also allows the greater ability for storage and transfer in networks.

•**Speed**: This means to both the velocity of a dataset of events occurring, and the streams of real-time data under the pressure. With the ways of collection of data from social streams, new insights are added to the database at rates ranging from milliseconds to years.

• **Variety**: The complexity of the existence of Big Data. In comparison to the structured databases, it consists of large streams of unstructured text, images, documents, e-mail, messages, videos, links and other forms that form a uniform set of data inferences. The result of such complexity is structuring and deepening of data together forming a constituent part and central concern for Big Data analysis. The internet which is an ultimate source of data, is nearly incomprehensibly big.

Variability – The variation of data being inconsistent across time.

Veracity – The Accuracy of data.

Complexity – Due to the unending sources of data, the Requirement of linking the multiple data sources forms the complexity of information.

Depending on the source of taken Information: Non-similar opinions on administrative data are contemplated to be Big Data or not. Big Data is hence also termed as found or organic data, since they are by-parts produced from processes whose basic purpose is not to survey research. Researcher has control and the inferential power of potential that varies between the different types of sources. It initiates new mechanism of calculating human behavior. The fundamental change in the nature of the new kind of data, availability, way of collection, combination with other data sources, and dissemination is common for new use processing. Huge Data is selective, non complete and error-ly. New errors are added during downstream. As such data is integrated from the separate sources at differetnt points in time and integrated later on to form dataset. The processes used are together linking of records, transformation to form new variables, documentation of the actions taken and acquiring the latest created attributes of the data. These activities are adders of errors that leads to bias and invalidity creating noise and poor reliability.It is a fact that Big data do calls to this tremendous amount in the volume of data that is difficult to process , store, and analyze via traditional database technologies.

Stream Processing of Centralized Systems of big data is always surrounded around in technological computations since decade. These kind of Computing scenarios, where one of them is the central computer controlling role of the peripherals and making out the complex computations. Centralized systems need bigger hardware to process varied volumes of

information and maintain number of online users concurrently at a time. Parallel processing at such circumstances over cloud and distributed computing systems normally gets exploited at a random scale. Shared resources can be virtualized and assigned randomly in the cloud where users can typically characterized scalability and elasticity of resources. At recent period of time, Multiple service providers give distributed cloud computing gathering numerous customers. The lower time constraints on a given event of stream processing systems increases higher modern applications imposed. Due to the new aspects of business and activities of customer over the growing situations of companies results in Integration of analysis. Real-time data sometimes is more valuable than the data used to create operations for the companies. Most common example is the Change tracks of businesses in public priorities on brands and products trying to consistently analyze the social media streams. As Potential of business ids lost after the closing of site or fraud detection after the transaction completed makes no sense. For example Otto can easily detects frauds from PayPal via analyzing millions of real-time transactions performed in a day. The imperative stream processing has numerous applicative cases. With real-time boundations over the data transition speed to process, a stream processing has commonly placed on top of a cloud architecture. Traditional data warehousing infrastructures cannot control the processing and analyzing the streams of data in real-time due to high latency and cost that is not easy. A data stream processing system is made to control the data streams and manage the continuous queries over time. It roles out the continuous queries once performed and continuously executed until uninstalled non implicitly. It formulates the output till the new data reaches in the computing system and info is processed on without being stored. Stream based systems differentiates from batch processing, as they depend on real time data processing. The Real time processing system meanssystem that responds within a time deadlines of the real world. It ensures that a process will be performed within a given time frame of few seconds depending on the constraints of service. The low latency systems are referred as real –time systems. To overcome its limits, elaborate and agile systems have been utilized for new demands. Streaming applications can be formulatedvia vertices of the operators and the edges as the channels for dataflow between operators known to be directed acyclic.

It involves two streaming models for stream processing:

1). The Stream Dataflow Approach

An application is treated as a dataflow graph involving operators and data dependencies between them. A task formulates the logic of a earlier set operator like bars, bounded window, sum or join or daily routine with user-specified logic. A stream of data within 2 operators depicts an infinite sequence of information formed by a task, readily available for later on consumption. Across a parallel task, Data is sent and taken in arbitrary sequence , and as an output, there is a limitfor a coarse-grain unit of transactional processing. Automatically pipelining is the normality of such a case.

2). The Micro-Batch Approach

This provides a remedy to allow the processing of data streams on batch systems. Via microbatching - a stream of computation formed as a sequence of number of transformations over a bounded sets by clustering a shared data stream into batches later scheduling batches ordered manner in a discrete node.

In early point of time, Streaming real-time was slow as compared to nowadays. Apache Storm is a continuous real-time distributed computing processing stream of messages on a batch. Single logical processing units are joined to show number of transformations and expose opportunities for concurrent processing. Spark Streaming is one solution that is responsible for easy building of streaming applications via the micro-batching approach.

The goal with it is to proceed as the batch processing but taking a count of the batch size always very small. Apache Samza is another distributed framework for stream processing that gives a simple API, in comparison to MapReduce. For the Commercial Applications, we have better options too like Amazon Kinesis - A completely manageable service for real-time processing of streaming data at high scale and becomes of great importance in the mentioned context of IoT.

Pre-requisites of Big Data Real-Time Stream Processing:

Efficiency requirements like throughput and latency are most important in streaming applications. Due to the increase in the latest applications imposed tightly on time constraints of a particular event. To fulfill such a requirement, processing without the costly storage operation is a way to complete it. MapReducebenefits in handling of larger data-sets but inhibits works via permanent storage. Hence, it declines to real-time data processing,due to its designing to facilitates batch processing on voluminous amounts of data. In-memory processing benefits via use of distributed main memory used to store and process the big data streams in real-time.To furnish in-memory computation effectively, Spark Streaming is one kind to be used extensively. It processes all kind of data in-memory and only interacts with the storage layer to initially load the data into memory and at the end to persist the final results. To implement in-memory computations, Spark uses a model called Resilient Distributed Datasets. Stream processing systems take the streams of input from different sources and rearrange the computations into directed graph of operators either explicitly or implicitly hence, called as compositional systems. They provide fundamental building blocks for composition of operators. The complete logic will be implemented by the user and the operators as foundations of classes. This is a case of declarative systems. To detect the event of interest, it require the Querying schemas with programming languages like Java as tools in streaming but applying lower level of schemes shows cycles of longer development and higher maintenance costs. Other languages like SQL gives queries that gets repeatedly and continuously evaluated as new data is made available. SQLStream and Calcite are the projects that emerged to execute the queries over big data using a set of specified extensions to the SQL. These data collections are tempted to be infinite, indulging into problems in streaming as it is not possible for all operators being evaluated over

streams. An engine of stream processing must acknowledge when to end an operation on a stream of data and output an answer.

Analysis accuracy is affected by Incomplete sets , destroyed data, outliers or biases in the training affect. For example- a sensor mishappening can result into a permanent missing value while a temporal disconnection could also occur or network delay forming a temporal loss as data arrives in a short delay.

In the case of Spark Streaming that batches up upcoming events within a short timeframe before pre-processing data. Data is formed and computations on RDD are be created in transformations or actions. The hidden data streams are called Dstream and it compose an RDDorder, with data over a certain stream interval. DStreamsallow a user to carry on the transformations with them. Streaming computations represents a complete series of batch computations of a definable time interval size. Onset on batch for stream processing consists of buffering the data when it goes into the stream creating few seconds of latency. There is an existence of penalty in latency equivalent to the micro-batch duration. Waiting for the flush to buffer too makes an increase in latency. However, Increasing overall throughput,the buffer permits Spark Streaming to handle the higher volume of incoming data.

The comparison made by Chitapali (found from a report) successfully designed and implemented a real-time streaming set mark focused on Storm, Flink and Spark Streaming and discovered that Storm and Flink exhibits much lower latency than Spark Streaming at high throughput. On contrast, Spark Streaming is capable of handling higher throughput and its output is quite sensitive to the batch duration of setting.

Amazon Web Servicesprofoundly called AWS services provide a large set of global cloud-based services including storage, databases, compute, analytics, networking, management tools, mobile, developer tools, IoT, security, and enterprise applications: both on-demand, available in a unit of time, option to pay-as-we -go pricing option. Counting to more than 150 AWS quick provisioned services with data warehousing to deployment tools.

This acts as a add on to enterprises, start-ups, small and medium - sized businesses, and customers in the public sector to control the fundamental building blocks required to response the changing business requirements. Since 2006, Amazon Web Services (AWS) started to offer the infrastructure services to businesses through web services called as **cloud computing**, with a great opportunity to replace the upfront expenditure infrastructure expenses required at a lower variable costs that scaling with the business. With the application of cloud, businesses need not to do longer planning for servers and infra in earlier. Instead of that, they have immediate spin up of hundreds to thousands of servers in minutes and do the output of results faster.All Around the globe, AWS shows a profoundly dependable, versatile, minimal effort foundation stage in the cloud that powers a huge number of organizations in 190 nations.

As of recent advancements in Technology, virtualization has been a broadly acknowledged approach to decrease the extra infra based expenses and increment the dependability of IT big business. Still more, grid computing develops a formerly new class of analytics, information crunching, and business knowledge undertakings conceivable that were recently cost and time restrictive. Alongside these innovation changes, the speed of advancement and extraordinary speed in the presentation the manner in which markets work for new items on a very basic level has changed. Alongside the wide acknowledgment of programming as an assistance contributions,also known as SAAS, have made ready for the most recent IT foundation challenge called distributed computing easy. This computing has made it possible to complete the ideal models of Infrastructure in IT. It extends on a considerable scale of the advancement in the IT business over the previous decade and provides opened doors to showcase and decrease costs. Associations help in devouring the shared processing and capacity assets instead of building, working, and improving foundation all alone. The speed of progress in business sectors makes noteworthy impressions on the venture IT framework to adjust and convey while giving new answers for addressing these changes.

Types of Cloud Computing

Distributed computing is governed by the help of engineers and IT divisions :-

Due to the rise of the Distributed computing some other modern model and methodologies are came into existence to help in eradicating the issue of customers. Services like adaptability, degrees of control, are served by cloud. Concepts like Framework as a service, Platform as Service and Software as service helps in arrangement procedures. also helps in enabling the selection of desired administrations can enable you to choose what set of administrations is directly for your requirements.

The fundamental structure is contained by the (IaaS)- that hinders for IT cloud infrastructure, it also been used to give regular access to systems administration highlights, information extra room and PCs (virtual or on devoted equipment). High level of executives power over your IT assets and adaptability is achieved through IaaS . It generally exist IT assets that various IT offices and which are world wide known to all designers

Platform as a Service (PaaS) is mainly responsible for fulfilling the requirement for the association to hold the control over with the foundation which is hidden. Also,allows you to pay attention on the on the company and the set of your applications. This will eventually help you to increase profit in scope organization, support programming and fixing with stress about asset acquirement

Finished item are provided by the Software as a Service (SaaS) By and large, individuals alluding to Programming as a Service are alluding to end-client applications.while working with the Software as service client does not need to think about how to keep the pace with administration or how to develop fundamentally designed framework.

From the official Documentation of AWS - "Distributed computing is a style of figuring where versatile and flexible IT enabled abilities are conveyed as a support of outer clients utilizing Internet advances." Such a cloud computing empowers associations to acquire an adaptable, secure and financially savvy IT framework, which has similarity to that national electric networks empowered by the homes and associations to plug into a halfway oversaw, productive and liberating the companies to dedicate precious workers and budget to activities.

These capacities incorporate databases, register power, stockpiling, messaging, and other structure administrations that run business applications. At the point when combined with an utility-style plan of action, cloud computing vows to convey an undertaking grade IT framework in a dependable, auspicious, and savvy way.

With AWS,We can demand figure force, stockpiling, and different features in minutes and have the adaptability to pick the improvement stage & programming model that bodes well for the issues they're attempting to tackle. It also has gives the advantage of paying just for what we use, with no straightforward costs or long haul responsibilities, making AWS a financially to convey applications.

For Example - Media organizations serve boundless video, music, and other media to their overall client base, without having to upgrade its infrastructure with AWS services.

Differences Which AWS as Key Role player than others:due these traits:

Adaptable. AWS empowers associations to utilize the programming models, working frameworks, databases, and designs with which they are as of now commonplace. Moreover, this adaptability assists associations with blending and match designs so as to serve their different business needs.

Cost Effective-Organisations that pay just for what they use, without direct front or long haul-responsibilities.

Versatile and flexible. Organizationscan rapidly add and reduce AWS assets to their applications so as to fulfill client need and manage costs.

Secure. In order to finish security and to every protection scenario, AWS fabricates benefits as per security best practices, giving the suitable security includes in those administrations, and records how to utilize those highlights.

Experienced. When utilizing AWS, associations can use Amazon's over fifteen years of experience conveying enormous scope, worldwide framework in a solid, secure design.

AWS is an exhaustive cloud administrations stage that provides process power, datastorage, content conveyance, and other features that are utilized to convey applications and administrations cost-viably—with adaptability, versatility, what's more, dependability. AWS self-

administration implies that you can proactively address your inside plans and respond to outside requests when you pick.

Deployment and Management

AWS Identity and Access Management (IAM)-IAM) empowers you to safely control access to AWS administrations and assets for your clients. Utilizing IAM, you can make and oversee AWS clients and gatherings and use authorizations to permit and deny their access to AWS assets. IAM permits you to:

Manages IAM clients & their entrance |Oversee IAM jobs & their authorizations |United clients and their authorizations

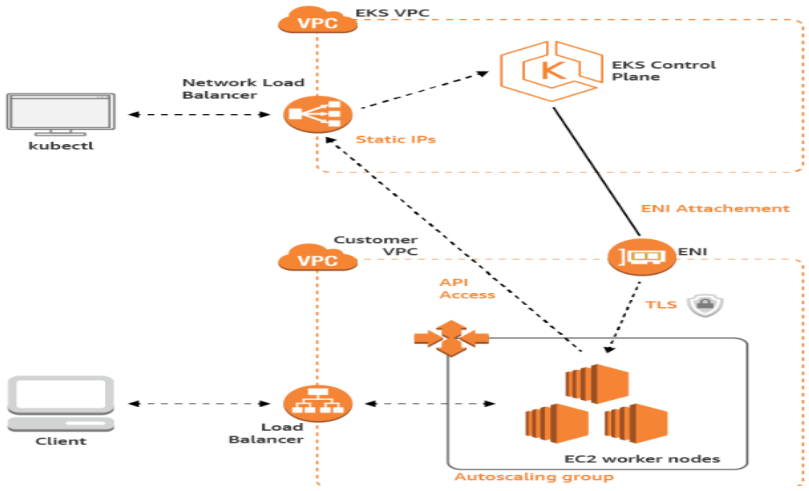
Amazon Elastic Kubernetes Service (Amazon EKS)

EKS is an oversaw management administration that makes it simple for you to run Kubernetes on AWS without expecting to stand up or keep up your own Kubernetes control plane. Kubernetes is an open-source framework for mechanizing the organization, scaling, and the board of containerized applications. Amazon EKS runs Kubernetes control plane occasions over numerous Availability Zones to guarantee high accessibility. Amazon EKS consequently distinguishes and replaces unfortunate control plane cases, and it gives computerized variant redesigns and fixing for them.

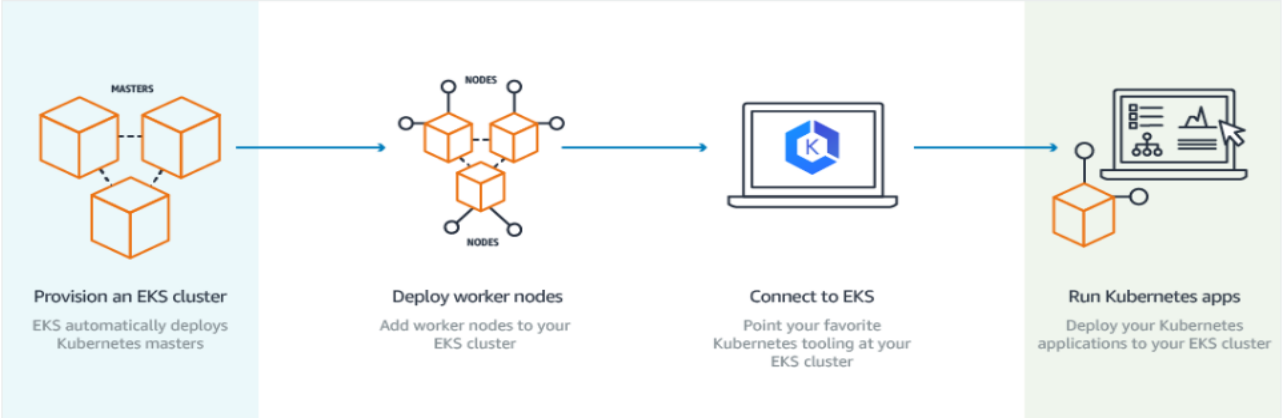
Amazon EKS is likewise coordinated with numerous AWS administrations to give versatility and security to your applications, including the accompanying:

- Amazon ECR for container images
- Elastic Load Balancing for load distribution
- IAM for authentication
- Amazon VPC for isolation

It (EKS) approaches date adaptations of the open-source Kubernetes programming, so you can utilize all the current modules and tooling from the community. Applications running on Amazon EKS are completely perfect with applications running on any standard Kubernetes condition.



Working of Amazon EKS Services



Steps needed to perform the Setup for EKS Services :

1.) To begin with, make an Amazon EKS bunch in the AWS Management Console or with the AWS CLI or one of the AWS SDKs.
2.) At that point, dispatch laborer hubs that register with the Amazon EKS bunch. We give you an AWS CloudFormation format that naturally arranges your hubs.
3.) At the point when your group is prepared, you can arrange your most loved Kubernetes apparatuses, (for example, kubectl) to speak with your bunch.
4.) Send and oversee applications on your Amazon EKS group a similar way that you would with some other Kubernetes condition.

Amazon EKS bunch comprises of two essential segments:

1.) The Amazon EKS control plane.

2.) Amazon EKS worker nodes that are enrolled with the control plane.

Creating a Kubernetes Cluster with eksctl command:

1.) This command creates a cluster with latest version of Kubernetes.my-cluster has to changed with name of the cluster.

```
eksctl create cluster \
 --name my-cluster \
 --without-nodegroup
```

2.) To verify the readiness of cluster:

```
kubectl get svc
```

3.) Output will show :

```
NAME             TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
svc/kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   1m
```

Amazon EC2 (Amazon Elastic Compute Cloud) is a web administration for designers that gives secure, resizable figured limit in the cloud. It is intended to make simpler web-scale processing. The Amazon EC2 is a straightforward web administration interface permits to get and achieve the limits with insignificant grinding. It gives assets unlimited authority for processing and enables running on Amazon's demonstrated figuring condition. Amazon EC2 decreases the required time to get and boot new server examples called occurrences to minutes in time, permitting to rapidly scale both all over limits, as change the figuring necessities. Amazon EC2 changes the financial aspects of processing by permitting to pay just for limit that you really use. Amazon EC2 depicts designers moreover the framework that apparatuses to assemble non-disappointment strong applications and detach it from regular situations.

Instance types of EC2 :

Amazon EC2 gives to the money related advantages of Amazon's scale. User pay a low rate for the process really devour. Purchasing Options for a progressively definite portrayal fro the pricing.

1. On-Demand Instances- With On-Demand, user pays only for registered limit continuously with no long hauling duties. User can increment or diminish a process limit contingent upon the requests of an application and predetermined compensation of the hourly rate for the examples used. The utilization of On-Demand cases liberates from the expenses and complexities of arranging, buying, and keeping up device and changes what are ordinarily huge fixed expenses into a lot littler variable expenses. On-Demand examples

to deal with intermittent traffic spikes additionally expelling the need to purchase "security net" ability.

2. Reserved Instances- Reserved Instances furnish with a critical rebate (up to 75%) contrasted with On-Demand case valuing. User to change families have the adaptability, working framework types, and Reserved Instance tenures while profiting by valuing when user uses Convertible Instances.
3. Spot Instances- Thses are accessible at up to a 90% rebate in contrasted with On-Demand costs and lets exploit unused EC2 limit. User can essentially diminish the expense of running applications, develop application's figure limit and throughput for a similar spending plan, and registering applications empower new kinds of cloud.

Docker

An Open funded platform with running applications and the procedures simpler to create and disperse. The applications under a standard structure called compartment are worked in the docker that is bundled to all the supporting conditions. These containers continue running framework's portion in a detached path. As far as executionis involved additional layer of deliberation impacts since it goes with new limits that earlier advancements didn't have. At first, it gives the features to develop and handlecntainers. Other than that, applications are filled up into lightweight dockercontainers. These are the virtualized applications that can function at other places without changes. It do have the possibility of passing on more virtual containers than various technological advancements. One of the advantage for use is that docker can be used without a stretch in arrangements with third-party, this helps in managing and conveying of docker holders. These Containers can be simply deployed into the cloud-based environments.

Here we present the review of theory based work of docker technology which exhibits an advantage to automate the applications when sent into Containers which is responsible for application being virtualization and execution, it includes an additional layer for arrangement on it productively and besides it gives code from the PC testing before creation.In several reports it has been affirmed that,a conceivable docker makes testing of code and convey it into the creation scenario.

Componencts Of docker:

1. Docker Client and Server- Server docker receives the solicitation from the client docker and afterward appropriately processes it. The RESTful API &an CLI -command line tool are dispatched by docker in a binary. Both the Docker daemon/server, docker client can be run on single machine or client associated with a remote server running on another machine is also possible.
2. Docker Images- In docker, two techniques can create an Image. The first is by utilizing a read-only layout. The establishment of each image is a base image which is a fundamental

way. Working framework pictures for example, Ubuntu 14.04 LTS are fundamentally the baseimages. These images makes framework hold a container with a capacity of fully operatable OS. Via modification required applications can be added to the base package, however it is important to assemble new one and is known as "submitting a change". The subsequent strategy is docker record development that contains a rundown of directions at the point when "Docker fabricate" order is run from the slam terminal it adheres to all the guidelines given in the docker record and manufactures a picture. This is a robotized method for building a picture.

3. Docker Registries- docker libraries also known as Registeries that compoisedocker images. It works with images that could be pushed & pulled from a source.
   Regsiteries are of two type- Open and Private. Docker Hub – that can pull accessible images and push their own without making from the scratch is called an open library due to it sharing ability. To the contrast Images that are only circulated to a specific zone by center point are private.
4. Docker Containers – Containers are made from Images of docker. It requires the entire unit to ensure the application being able to run in a confined manner. For instance, assuming there is a Ubuntu OS img with SQL, when this image is run with order a holder will be made and SQL instantiates on UbuntuOS.

Implementation of Kafka using Docker:

The main activity required is for Docker Hub and discover an image of Kafka Docker. Characterize the primary draft of the Docker Compose record:

```
version: '2'

services:

 zookeeper:
  image: wurstmeister/zookeeper:3.4.6
  expose:
  - "2181"

 kafka:
  image: wurstmeister/kafka:2.11-2.0.0
  depends_on:
  - zookeeper
  ports:
  - "9092:9092"
  environment:
   KAFKA_ADVERTISED_LISTENERS: PLAINTEXT://localhost:9092
   KAFKA_LISTENERS: PLAINTEXT://0.0.0.0:9092
   KAFKA_ZOOKEEPER_CONNECT: zookeeper:2181
```

kafka-console-producer.sh and kafka-console-consumer.sh are the utility involved in the above case after being bootstrapped.

```
bhomit@host$ bin/kafka-console-producer.sh --broker-list localhost:9092 --topic test

>Hi there! Welcome to Stock testing market

>It is a test message.
```

This above output window shows a running producer from the Docker host machine

```
bhomit@host$ bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic test --from-beginning
Hi there! Welcome to Stock testing market
It's a test message.
```

This above image depicts consumer.

In this when both producer & consumer are present in single network, all works well.Focusing on the KAFKA_ADVERTISED_LISTENERS condition variable from the above Compose record. Kafka sends the estimation of the variable to consumer. In the wake of accepting that stream, use it for sending or expending records to and from the Kafka merchant.

As we mentions the variable as PLAINTEXT://localhost:9092, this applies to both getting it on the underlying association and utilized it for additional correspondence with 9092 port intermediary.

Plainly from the output, the Kafka is reachable at 9092 address. In this way to connect with broker , all together to have the option to speak with it, the KAFKA_ADVERTISED_LISTENERS esteem set to PLAINTEXT://kafka:9092.

```
kafka:
image: wtm/kafka:2.11-2.0.0
depends_on:
  - zookeeper
ports:
  - "9092:9092"
environment:
    KAFKA_ADVERTISED_LISTENERS: PLAINTEXT://kafka:9092
    KAFKA_LISTENERS: PLAINTEXT://0.0.0.0:9092
    KAFKA_ZOOKEEPER_CONNECT: zookeeper:2181
```

This is the docker compose file maintain wtm zookeeper and kafka at 9092 port. Hence the container at client machine must be set. The environment variables define the use case with documentation.

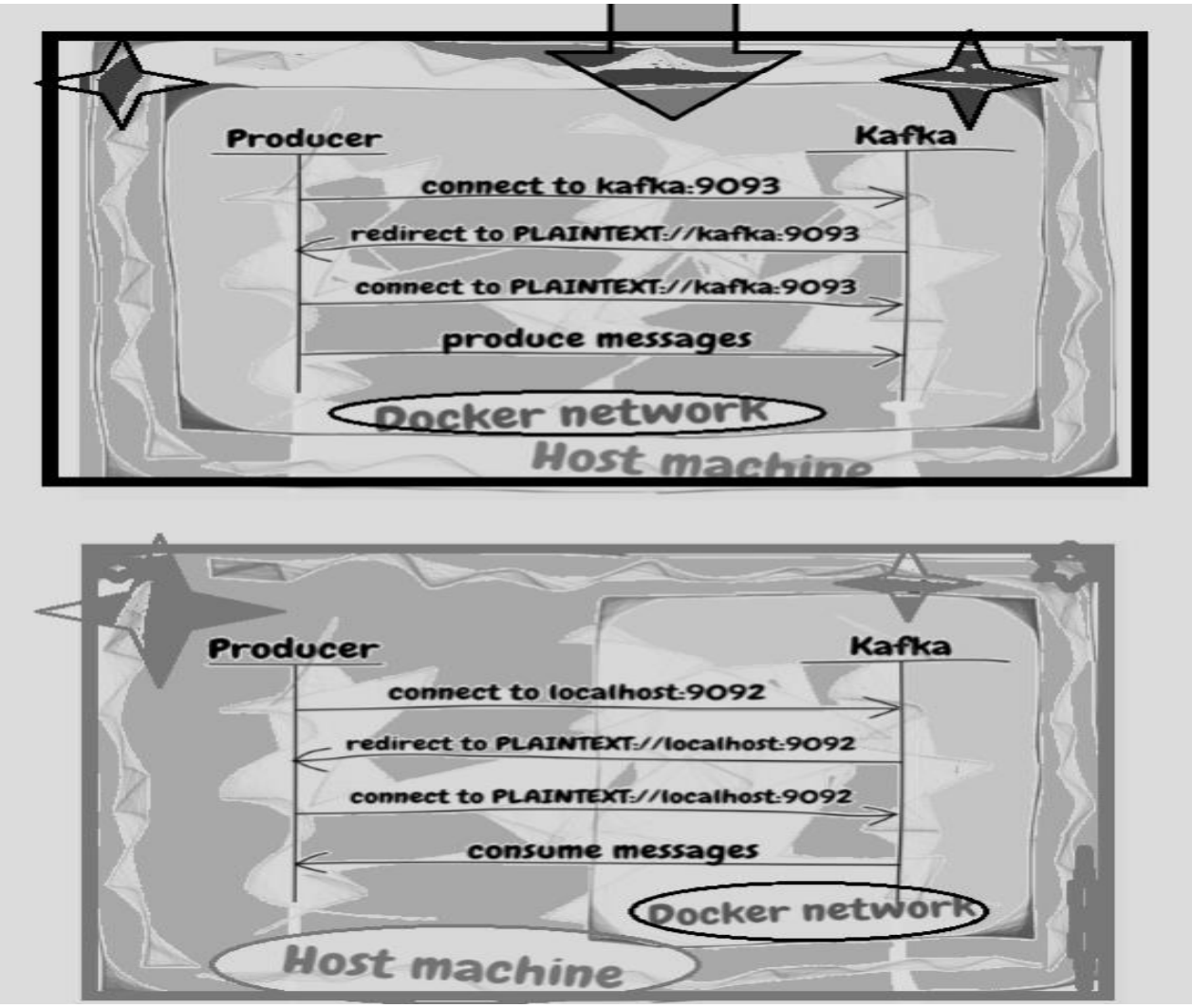
To run producer within Docker network:

```
root@7dfb9eaa81dc:/kafka# bin/kafka-console-producer.sh --broker-list kafka:9093 --topic test
>Hi there!
```

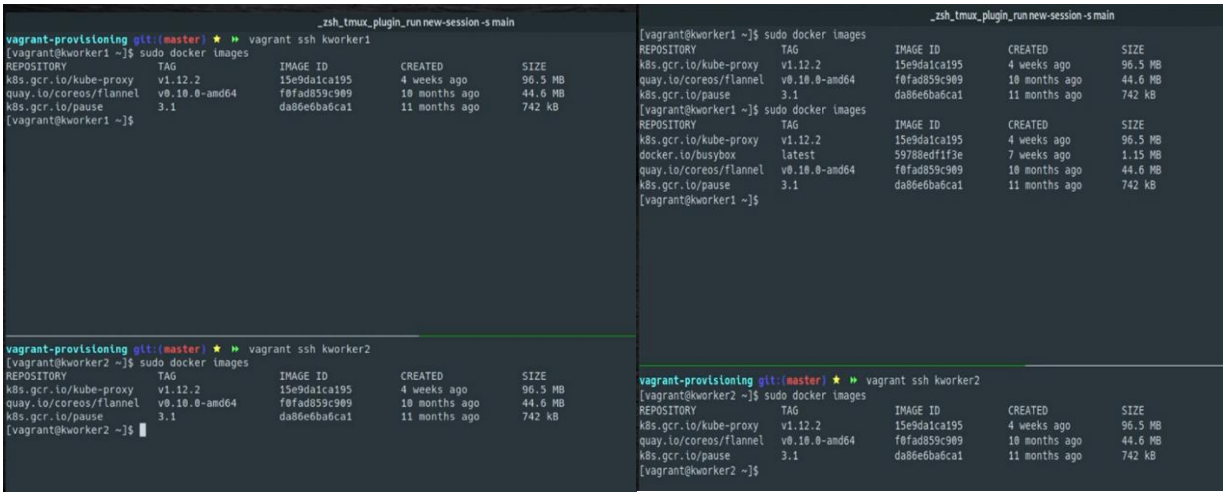
Consumer from the host machine :

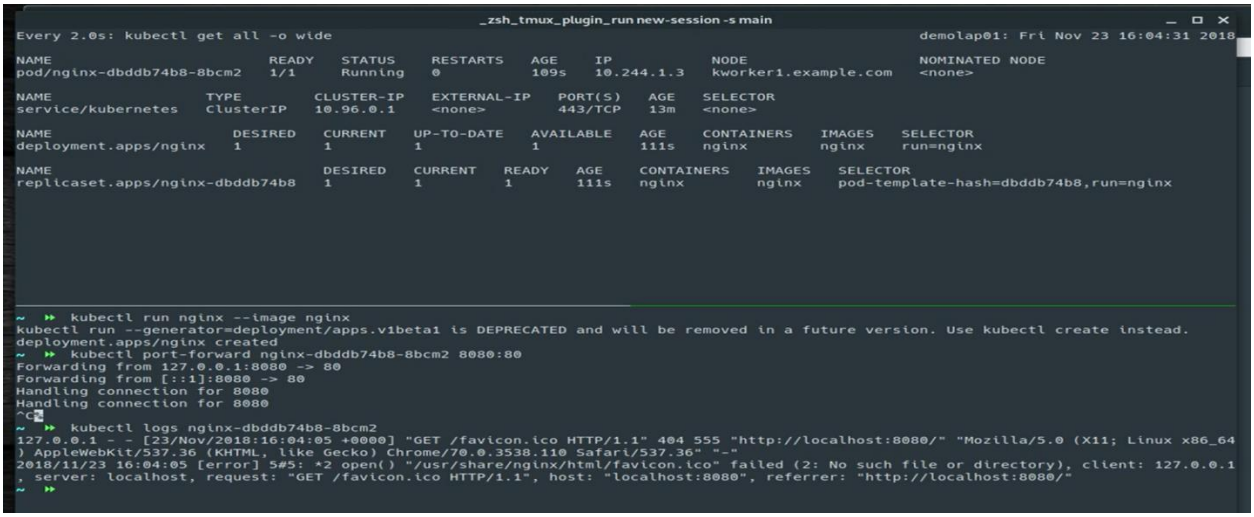
```
bhomit@host$ bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic test --from-beginning
Hi there!
```

This figure below is an inference of the working mechanism of Kafka streams with Docker.



Running Docker containers in cluster of Kubernetes:





The above outputs have been deployed with the help of Opensourcing Content creator mentioned in ref.- Y1a

kubectl run command runs a Deployment to container, the Deployment restarts if you shut down the connected process via Ctrl+C, unlike docker run -it. To terminate the Deployment and its pods user need to start kubectl delete deployment command. To get complete data about nature and design of cluster, command used above is kubectl group information with docker info.

```
Containers: 40
Images: 168
Storage Driver: aufs
 Root Dir: /usr/local/google/docker/aufs
 Backing Filesystem: extfs
 Dirs: 248
 Dirperm1 Supported: false
Execution Driver: native-0.2
Logging Driver: json-file
Kernel Version: 3.13.0-53-generic
Operating System: Ubuntu 14.04.2 LTS
CPUs: 12
Total Memory: 31.32 GiB
Name: k8s-is-fun.mtv.corp.google.com
ID: ADUV:GCYR:B3VJ:HMPO:LNPQ:KD5S:YKFQ:76VN:IANZ:7TFV:ZBF4:BYJO
WARNING: No swap limit support
```

All this information is used for understanding of mechanism in which the both works.

```
Kubernetes master is running at https://203.0.113.141
KubeDNS is running at https://203.0.113.141/api/v1/namespaces/kube-system/services/kube-dns/proxy
kubernetes-dashboard is running at https://203.0.113.141/api/v1/namespaces/kube-system/services/kubernetes-d
Grafana is running at https://203.0.113.141/api/v1/namespaces/kube-system/services/monitoring-grafana/proxy
Heapster is running at https://203.0.113.141/api/v1/namespaces/kube-system/services/monitoring-heapster/pro
InfluxDB is running at https://203.0.113.141/api/v1/namespaces/kube-system/services/monitoring-influxdb/pro
```

These Distributions utilizes an organized discharge model that has better stability, testing and temperamental stages exposed to broad testing in order to catch a potential issue.These difficulties can be altogether decreased in light of the fact that Docker characterizes the condition of a specific working framework and library suite.It is clear from the use that a comprehensible synopsis of the vital programming conditions, ecological factors, etc expected to execute the code is expressed via the Dockerfile.There is little chance of the sort of loops in such a content, that so as often as possible reason trouble in physically actualized documentation of conditions.

Pros and Cons

1.) **Kubernetes-** Kubernetes ("k8s") the container orchestration software is emerging as one of the most known open-source projects and its acceptance has continuously gained popularity. It is successor of Borg, Google's in-house orchestration. We have seen a successful implementation Kubernetes in many companies that wants to have building and scaling software. It includes various phases that have been achieved by the engineers in the company i.e. adoption cycle of Kubernetes, unbroken enthusiasm about the possibilities, this technology will provide you the sleepless nights but lead in resolving issues with clusters and deployments.

Kubernetes Pros

As now we got the basic idea about the working of Kubernetes, we can talk about the advantages about this open Source project platform. Following is the list of the same:-

- Broad adaption in the cloud relative/native community
- Excellent ecosystem that contains supporting tools and interfaces
- Deep combination with the cloud native ecosystem
- Wide support for containers runtimes
- Various methods for workloads and deployment options
- Great assistance for useful apps
- Quite basic networking model
- Efficient and Effective real time application updates
- Adequate management resource
- Built-in security features
- Flexibility, Extensibility and plug ability
- Combination with broad cloud providers

Broad Adaptation in the Cloud Native Community

The popularity of software has reached to next level in the open source world, Some of the key advantages that it brings with it are- and release cycles, and better maintainability (bug fixes and feature updates), faster development

At present time, Kubernetes software is the most useful orchestration platform in the world. It has a wide range community of developers, maintainers and end-users.

Excellent ecosystem that contains supporting tools and interfaces

- Service meshes-example; Istio
- Monitoring-example; Dat=
- adog, Prometheus
- Networking-example; Weave Net, Cilium, Flannel, Weave Net
- Machine Learning- example; Kubeflow.

Wide support for containers runtimes

container runtimes is one of the key highlight of the Kubernetes. No Doubt in, that Docker is in great demand but users may require other container runtimes as well. That's when Kubernetes come into picture.

The Container Runtime Interface (CRI) is introduced by kubernetes , an interface that handles a wide array of container runtimes without the requirement to recompile, in v1.5. Container runtimes such as as Docker, CRI-O, Containerd, frakti, and other CRI runtimes are taken care of well by Kubernetes.

Various methods for workloads and deployment options- Various API resources and primitives for running any type of application are:-

- **Replicated apps**- You can run multiple instances of your application at same time with the help of Kubernetes.
- **Daemon**s- The facility to allow users to run a task on each node in the cluster is achieved by DeamonSets.
- **Jobs and cron jobs**- Task like automate database backups, sending emails, application upgrades etc are fullfiled by CronJobs
- **Microservices**- Kubernetes consist of primitives (e.g., Services) that permits constructing a microservices application.
- **Application stacks and collocated apps**- Services like resource sharing and enabling communication ,Pods allow you to collocate multiple containers in a single environment
- **stateful applications**- Kubernetes shows its main support for persistent storage required by stateful apps.

Kubernetes Cons

As per the saying everything comes with Good and bad so as the Kubernetes. Also Switching to such a new and modern technology Kubernetes also has various challenges which is observed . Following are the cons and challenges.

- Missing High Availability piece
- K8s talent may be expensive.
- Steep learning curve
- Difficulty in installing and configuring it manually

In order to initialize the cluster few parts or components in should be configured and installed. Another hardship is that if you wish to install Kubernetes manually, you have to configure security, which further requires assigning a certificate authority and issuing certificates. Some of the main pre-installation and post-installation tasks are:

- Configuring High Availability of masters, components, applications, load balancers, etc.
- multi-host networking Configuration
- Combining storage
- Starting monitoring, logging, auditing

2.) **Docker-** A Docker is a lightweight container image, executable package, stand-alone of a piece of software that handles everything needed to run it i.e. system tools, runtime, system libraries, settings, code. The main key point about the Docker is that same hardware can share the more applications running as compared to other technologies; which makes it super easy for developers to make quickly ready-to-run container applications; and further it leads to makes managing and deploying applications easier.

Docker, the good side

Following are the advantages when you talk about Bare-bone servers or VMs.

- **Containers are small compared to VMs-**In comparisons with Azure App services, AWS Lambda or Azure Functions they are definitely bigger than Functions. VMs start from Gigabytes and they may start somewhere from tens of megabytes up. So your current server can host way more containers compared to VMs.

- **Containers uses less resources-** Because of their light weight properties same kernel can be shared all together. Also they don't have a full OS

- Fast boot- Speed is also one of the major advantage of using Docker, as things do not go in your way container takers a second to start. Deploy new instance is there in front of you in no time.

- **Eradication issues of the "Works on My Machine" situation-**Because of the identical local and QA environment issue of working in one environment, and then code doesn't work in other environment will not happen. So configuration issues is also

- **They work well in DevOps and CI/CD- Services like** Micro services, DevOps, and continuous deployment are perfectly fitted into Container-based virtualization as it is super easy to manage them.

Things that need to be careful of while Using Docker:-

- Security-Any technology is secured till the time it has not been hacked or breached. So if you go Online you will find the ways to hack the Systems which are not well secured.

- Isolation-Because of the fact related with the Implementation of same Kernel, 100 percent isolation is not possible. So while performing the task using same container make sure that you are aware of the task performed are correct.

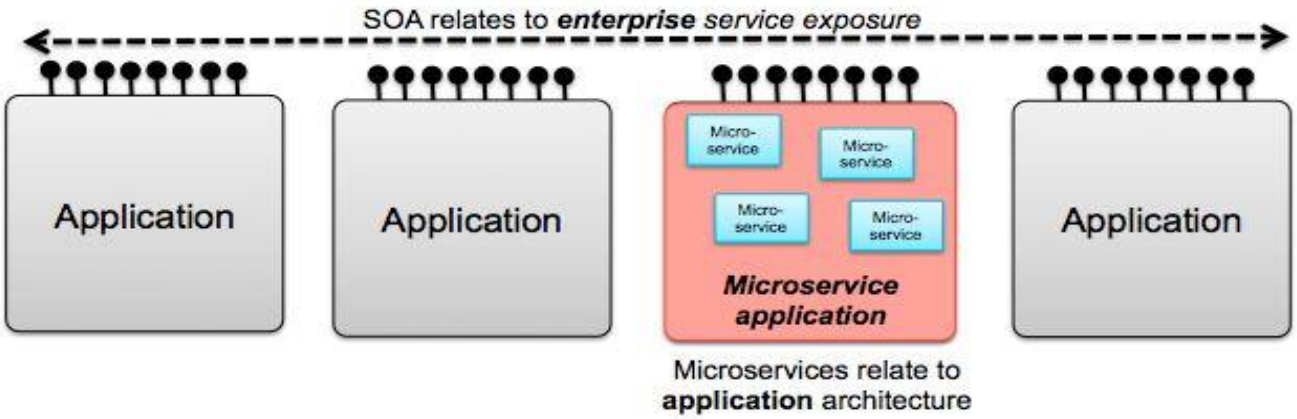
- Networking- Communication is one of the hindrance being faced while using Docker. Also playing around the network is tricky sometimes as you want to limit the access of the container.

3.) **Microservices**- Monolithic systems get broken separately through a famous Software design architecture called Microservices. Applications can be considered as collective Summation of loosely coupled services. Every single feature are assigned to Microservice. Communication protocols such as TCP and HTTP are used for the interaction.

- **Big data***:-* Data pipeline-oriented architecture perfectly incorporates the Microservices , which aligns with the way big data has carried all the functionalities should be synchronized, adapted, processed, and delivered. Microservice performs in a way that *each step in a data pipeline is responsible for handling one small task.*

- ***Real-time processing***- Publish-subscribe framework is approach that is followed by the A microservices architecture, making the data processing quick and real time so that delivery can be processed immediate output.

- ***Deployment-*** Because of its modular architecture microservices enable code and data reusability, making it simpler to deploy more data-driven use cases and descriptions and solutions for extra business gain.



Source of this picture-https://raygun.com/blog/what-are-microservices/

Pros of Microservices

Diversity*- key features that it brings with it are that you can* storage technologies, development frameworks each module can be developed in any programming language that best fits the requirement, with microservices, mix multiple languages, and data storage technologies.

Clarity*-It works on very basic principle and the* code base is very small .While working with microservices, main focus is given at particular module on a given time. Therefore it is very easy for the engineer to fully understand the code.

Scalability*- While following the* monolithic architecture they can impact the other stacks. *Deployment can be achieved by placing them into* multiple servers to increase performance and separate them from other services.

Cons of Microservices

Time consuming*-Interaction of the modules* with another *is done by*by REST API, which results in the slow pace of the development process as compared in monolith.

Difficulty testing- Because of the different environmental positions Microservices-based application has to solve numerous independent services that have to be tested. complexity increases with the incrementation of the modules. Thus integratation between the microservices is affected and resulting in downfall efficiency of the project.

Skill demands- Because high complexity is involved in the project it requires high level of expertise in order to handle the microservices. Also every functional operation is performed at individual level.

4.) **Google Cloud vs. AWS vs Microsoft Azure**

- **Amazon Web Services** – ,Amazon's capabilities are unmatched because of its great range in the tool set and that has increased rapidly across worldwide. It's Cost is one of the areas that needs to be checked again along with its focus which is primarily set on the public cloud rather than hybrid cloud. Because of the interoperating with your data centre or private cloud , AWS has not been amongst the favourites all the time.

- **Microsoft Azure** – Another top candidate of Cloud Computing platform is Microsoft Azure and has been running in parallel with AWS because of the speciality of its cloud infrastructure. The fact that AWS focus primarily on Public cloud, working with Hybrid cloud is the key strength for Azure. As it works very hard to interoperate with data center.

- **Google Cloud** –Not one of the Best cloud platform but still in place amongst the best, Google cloud has also been another tool that is being into trend because of its main strength in the field of artificial intelligence, deep learning, machine learning and data analytics.

AWS pros and cons

Powerful growth and dominance in market has been Amazon's biggest strength in cloud market. Gartner quoted that Amazon web service has been the maximum share leader in the industry of cloud for almost about 10 years now. In its Magic or fourth Quadrant for Cloud platform as a Service, Worldwide.

There is no doubt that because of its operational scope it has gain massive popularity. Due to its wide range in the area available services of services it has been in customer's top priority and network of worldwide data centers. As per the saying of the Gartner report "AWS has been declared as one of the most efficient enterprise ready- provider, also most matured with its effective strength in handling the large number of resources and users."

As mentioned the biggest challenge faced by Amazon's is related to cost. Also reducing the prices of the same is not helping AWS as it makes it difficult for the customer to understand the Cost policy and plans. It makes more complications when some enterprises get the same project done in lower cost but some has to pay higher.

One can easily say that nothing in this world is perfect so as the AWS , so other advantages related with Amazon web services are more beneficial as compared to its pros. Thus making AWS a perfect tool for the organizations of all sizes continue to use for a broad range of workloads.

Google Cloud Platform pros and cons

Google gets an upper hand in terms of its feature called Google containers, since the Kubernetes being developed by Google standard that Amazon Web services and Microsoft Azure

now offer. Google is well aware of data centers and relatively has quick response time. Features or Offerings like analytics, Big Data, machine learning are derived from Google Cloud platform. Load- balancing and Scalability are few more strength areas of Google cloud platform.

Now talking about some hard facts about Google cloud platform is that it stands low in market share because of its less range of variety in services as compared to Microsoft Azure and Amazon web services. Moreover it is expanded over a limited scale and does not have much global data center. But it is making strong growth in the market.

In the report Gartner stated that Google cloud platform is taken as secondary platform in comparison with the Amzon web services. Google is well aware of data centers and relatively has quick response time. Features or Offerings like analytics, Big Data, machine learning are derived from Google Cloud platform. Load- balancing and Scalability are few more strength areas of Google cloud platform. Google Cloud Platform is more advanced to increasingly strategic option to AWS are more open-source-centric or DevOps-centric, and thus are less well-aligned to Microsoft Azure."

Microsoft Azure pros and cons

There is a famous saying that late is better never, Azure Microsoft is the third cloud platform which has been into the market now. Essentially taking its on-main software –Dynamics Active, Windows Server, SQL, Directory, Office, SharePoint, .Net, and others – and reprocessing it for the cloud.

The main reason for the excellent performance by Azure and deployment of Windows and other Microsoft software are done by achieved by the enterprises. Azure being the Microsoft product it makes sense to use the platform which is integrated with the Microsoft services. Already there is great trust being developed within the market between the Microsoft customer and Azure clients. Also further you get some loyalty points and offers for again choosing the Microsoft product.

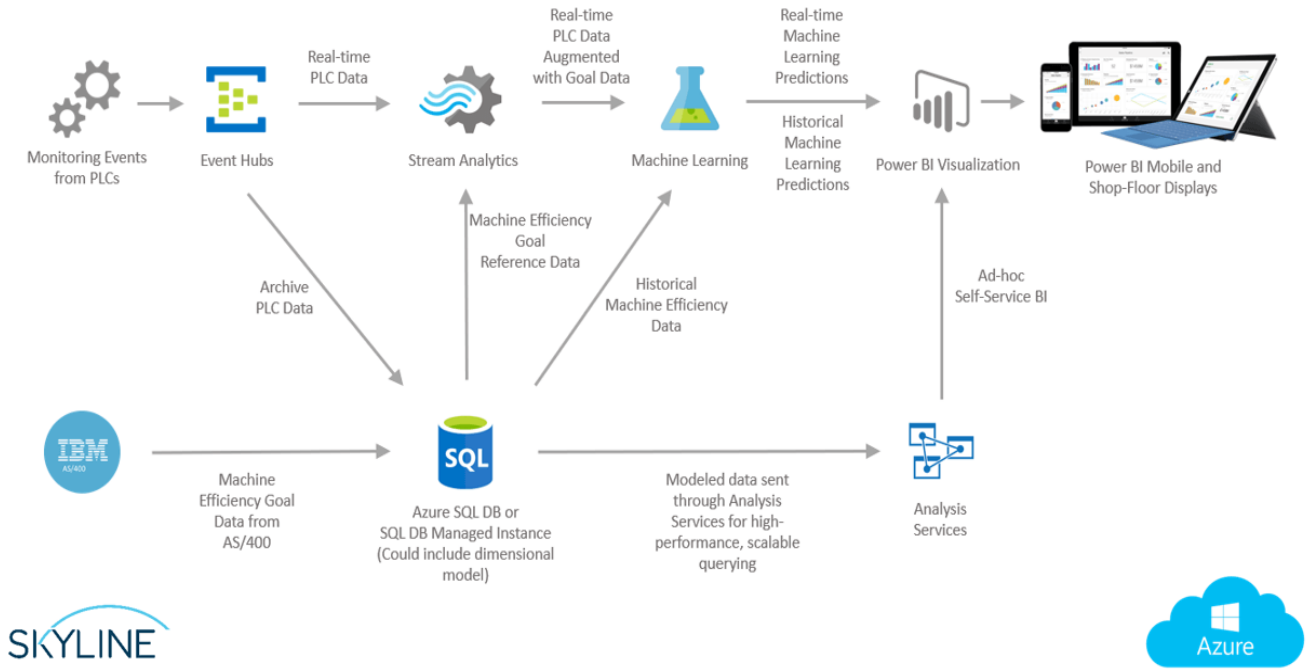
After talking about the Advantages, Gartner reveals about the facts related to the issues in the Microsoft Azure cloud platform. Enterprise-ready platform is the word used for the Microsoft Azure but it is not exactly the same what has been stated for the Microsoft Azure as there have been numerous negative feedbacks. It has been registered related to the communications error, documentation, and lack of readiness in technical support, proper knowledge and guidance among the installers and the partners of the ecosystem known as ISV ecosystem."

Vendor	Strengths	Weaknesses
AWS	• Dominant market position • Extensive, mature offerings • Support for large organizations • Extensive training • Global reach	• Difficult to use • Cost management • Overwhelming options
Microsoft Azure	•Second largest provider • Integration with Microsoft tools and software • Broad feature set • Hybrid cloud • Support for open source	•Issues with documentation • Incomplete management tooling
Google	• Designed for cloud-native businesses • Commitment to open source and portability • Deep discounts and flexible contracts • DevOps expertise	• Late entrant to IaaS market • Fewer features and services • Historically not as enterprise focused

Source of this table:-https://www.datamation.com/cloud-computing/aws-vs-azure-vs-google-cloud-comparison.html#overall.

Real-time stream processing

As the name suggest that all the specific actions are taken as soon as the data is published or produced. previously, the definition of Real- time stream processing it is meant to be that action upon the data that is taken as frequent as it is generated." As stream processing is gaining popularity and becoming the integral part of cloud platform, Real-time stream processing has emerged as the solution for all the remedies. It takes only microseconds for processing rather than waiting for whole days or long hours.



Source of the image- https://www.skylinetechnologies.com/Blog/Skyline-Blog/November_2019/realtime-vs-near-realtime-data-pros-cons.

Pros of Real time Streaming

Near-Real-time data processing comes with lots of positive features that are stated below, please find out the key strength of Real-time streaming platform.

Low Latency

We get very lesser latency at First,: data availability and visualization are the subset available at the time of processing. It has extremely useful in the situation when there is requirement of Quick- decision needs to be taken and along with rapid actions. For an instance seconds and minutes will make huge difference between million of Euro's and useless parts of scrambled machinery when a quality engineer is keeping the track of quality and have to face machine failure.

Intervention

Intervention or changes is possible while working with real-time processing as compared with near real-time processing. Functions like potentially change outcomes, advisories, in-flight transactions checking's. Based on previous choices in an e-commerce system is feasible because of personalized marketing. Feature like this intervention is into business completely when on time or spontaneous feedbacks are involved or appreciated.

Cons of Real time Streaming

Real-time also posses some of the disadvantages which are:-

No Summarized Data

Summarization of data is not possible. Like if we are performing function of averaging real-time situation are allowed to happen on small screen. So in all we are only able to capture whole of the data all together but not at some particular points. However both can be integrated together but the issues of the Summarization of data is considered as one of the failures of real-time.

Lack of Persistence

Persistent for deep knowledge/analysis is considered as another drawback related to the approach of real-time.

No Complex Calculations

Another disadvantage in the bag for the real-time is calculation. A very basic calculation is being implemented in this real-time approach. Not much can be added in terms of logic and calculation. Also it is not an easy task to perform data evaluation to other transfer that have moved through our system. You can say it is not completely supportive in nature. One can easily say that in real-time scenario modern calculations are difficult to perform.

Following Steps have been performed for Localhost Project.**(refer to github for sample)**

Start Zookeeper

1. go to zookeeper file
2. C:\zookeeper-3.5.6>zkserver run....

Start Kafka

1. go to kafka file
2. .\bin\windows\kafka-server-start.bat .\config\server.properties run code

Create Topic

1. Open command prompt and go to Apache Kafka installation directory.
2. Go to \bin\windows directory.
3. Run the following command.
4. kafka-topics.bat --create --zookeeper localhost:2181 --replication-factor 1 -- partitions 1 --topic sql-insert

Create producer

C:\kafka_2.11-2.3.1\bin\windows>kafka-console-producer.bat --broker-list localhost:9092 --topic Kafka_Example

Create Consumer

C:\kafka_2.11-2.3.1\bin\windows>kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic Kafka_Example --from-beginning

Create config folder

createKafkaConfig file
- initialize all config in producerFactory
- config bootstrap, key and value

Create model folder

create User class
- define variable
- set get and set method

Create resource folder

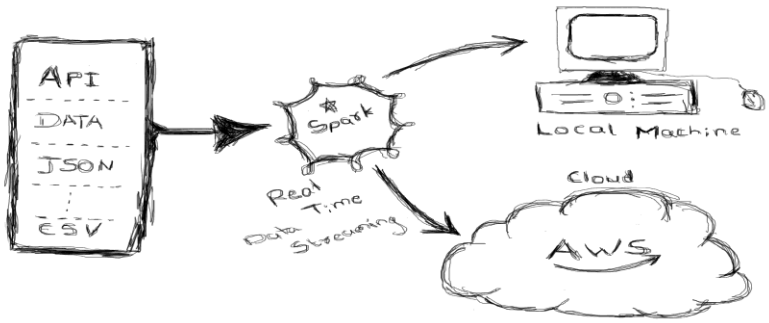
createUserResource class
- KafkaTemplate with object
- @Getmapping to get data from JSON
- use Topic name which already created

To run Spark

opencmd
run **spark-shell**

API Connection

Basically, Spark work with three APIs DataFrames, Datasets, and RDDs. DataFrames works as column headers in excel sheet, it is distributed collection of rows under named column. Datasets is strong type, immutable collection of object mapped with relational schema and can create by JVM object and manipulate using complex functional transformation. Datasets read data from JSON file using SparkSession. RDD is collection of records with distributed computing, it can be operated in parallel low-level APIs. It supports transformations and actions which uses map(), flatmap(), filter(), top(), savetofile() respectively.



With Alpha-Vantage API we can extract weekly data, monthly data, yearly data, intermediate data etc. apart from this we use some function for 5min, and 1 min duration changes in stock data. Above is an example of 5 min duration of data. This API returns inter day time series (timestamp, high, low, open, close, volume) of the equity specific data. We can get JSON and CSV file to collect and retrieve data from Alphavantage. The most beneficial thing is it gives prices and volume information of the current trading day, update Realtime. We can also use 20+ year historic data to stream and compare with real time.

Conclusion

To sum up, Apache Kafka is used to build real-time data streaming application. It is horizontal, scalable, fault-tolerant, high-throughput, and low latency platform for handling real time data feeds. Kafka provides error free, structured data which then will be an input source in Apache Spark. Apache Spark provides various APIs to stream real time data. Here we covered all the necessary comparison with kafka and Spark. We found that Spark proving itself best for real time data streaming platform.

By utilizing the technologies such as Kafka, Spark, Docker, Kubernetes in Cloud. We were able to achieve the key objective of Streaming Live Data. The data we got was from Alpha-Vantage API for real time as well as historical data of stock market. The output is getting stored to the docker container which than will be orchestrated by kubernetes clusters.

References

https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/

http://kafka.apache.org/

https://mapr.com/ebooks/streaming-architecture/chapter-03-streaming-platform-for-microservices.html

https://itnext.io/how-to-install-kafka-using-docker-a2b7c746cbdc

http://doku.iab.de/grauepap/2015/BigDataTaskForceReport_FINAL_2_12_15.pdf

https://www.researchgate.net/publication/331447616_A_review_on_Big_Data_real-time_stream_processing_and_its_scheduling_techniques

https://www.researchgate.net/publication/331447616_A_review_on_Big_Data_real-time_stream_processing_and_its_scheduling_techniques

https://docs.aws.amazon.com/eks/latest/userguide/create-cluster.html

https://d1.awsstatic.com/whitepapers/aws-overview.pdf

https://docs.aws.amazon.com/eks/latest/userguide/clusters.html

https://d1.awsstatic.com/whitepapers/aws-overview.pdf

https://docs.aws.amazon.com/eks/latest/userguide/what-is-eks.html

https://www.gtu.ac.in/uploads/Report-AWS.pdf

https://docs.aws.amazon.com/eks/latest/userguide/getting-started.html

https://aws.amazon.com/getting-started/projects/deploy-kubernetes-app-amazon-eks/?trk=gs_card

https://www.kaaproject.org/kafka-docker

Y1a- https://www.youtube.com/watch?v=-NzB4sPZXwU

https://kubernetes.io/docs/reference/kubectl/docker-cli-to-kubectl/

https://devspace.cloud/blog/2019/10/31/advantages-and-disadvantages-of-kubernetes

https://landscape.cncf.io/

https://supergiant.io/blog/pros-and-cons-of-adopting-kubernetes-in-2019/

https://affinitybridg e.com/blog/pros-and-cons-docker

https://databricks.com/glossary/spark-api

https://koukia.ca/why-docker-pros-and-cons-949d104478c5

https://www.google.com/search?q=docker+pros+and+cons&source=lmns&rlz=1C1CHBF_enIN816DE817&hl=de&ved=2ahUKEwiErp7w6rXoAhVdgKQKHeN7B-gQ_AUoAHoECAEQAA

https://kruschecompany.com/microservices-when-do-we-need-them/

https://raygun.com/blog/what-are-microservices/

https://www.datamation.com/cloud-computing/aws-vs-azure-vs-google-cloud-comparison.html

https://www.varonis.com/blog/aws-vs-azure-vs-google/

https://www.datamation.com/cloud-computing/hybrid-cloud-computing.html

https://it.toolbox.com/blogs/itmanagement/advantages-and-disadvantages-of-streaming-real-time-big-data-in-the-cloud-031214

https://www.fivestardev.com/blog/pros-and-cons-for-batch-and-stream-processing

https://www.skylinetechnologies.com/Blog/Skyline-Blog/November_2019/realtime-vs-near-realtime-data-pros-cons

https://hazelcast.com/glossary/real-time-stream-processing/

https://docs.docker.com/ee/ucp/user-access/kubectl/

https://kubernetes.io/docs/reference/kubectl/docker-cli-to-kubectl/

https://itnext.io/starting-local-kubernetes-using-kind-and-docker-c6089acfc1c0

t: https://www.researchgate.net/publication/266477818

: https://www.researchgate.net/publication/318816158

https://www.researchgate.net/publication/318816158_An_Introduction_to_Docker_and_Analysis_of_its_Performance

https://vsis-www.informatik.uni-hamburg.de/getDoc.php/publications/561/Real-time%20stream%20processing%20for%20Big%20Data.pdf

https://book.huihoo.com/pdf/confluent-kafka-definitive-guide-complete.pdf

Book of Big Data by –Nathan Marz with James Warren

https://vsis-www.informatik.uni-hamburg.de/getDoc.php/publications/561/Real-time%20stream%20processing%20for%20Big%20Data.pdf

http://cse.unl.edu/~ylu/csce990/papers/Streaming%20Big%20Data%20Processing%20in%20Datacenter%20Clouds.pdf

https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/

https://www.edureka.co/blog/docker-container-in-production-amazon-ecs/

https://itnext.io/scalable-microservice-demo-k8s-istio-kafka-344a2610eba3

https://devopscube.com/what-is-docker/

(4 more books in hard copy have been studied to create this report.)