Serverless Architecture for Big Data Analytics

Md Mijanur Rahman
Department of Computer Science and Engineering
Southeast University
Dhaka, Bangladesh
{mijan201} @gmail.com

Md Hasibul Hasan
Department of Computer Science and Engineering
Southeast University
Dhaka, Bangladesh
{hasan} @letslearncloud.com

***Abstract*— Big data is a phrase that describes the large quantity of data, it would be structure, semi structure and unstructured. In the present industry data is indispensable for the business organization. The Big Data initiatives and technologies are used to analyze this massive amount of data for gaining insights which may help in making strategic decisions. For example, data size is increasing day by day like petabyte, Exabyte, zettabyte, yottabyte and more. That is why it is going to tough and complex to manage this large scale of data. In practical, there are many challenges to process and compute this data like server management, storage, clustering, algorithm deployment, etc. As everything is done by manually, so it is hard to design the exact architecture for data analysis in the Cloud. Serverless computing is a mechanism to provide pay-per-use backend services to clients. A serverless provider lets users create and deploy code without worrying about operating and managing servers. In this paper, we present serverless architecture for big data analytics, also we show how to implement, maintain, and governance of a serverless big data application on AWS (Amazon Web Service). In addition to it, we will demonstrate the difference between traditional data analytics and big data analytics in a serverless system.**

***Keywords*— Serverless Cloud Computing; AWS Serverless Service; AWS Lambda; Big Data Analysis; Amazon Web Service.**

I. Introduction

Serverless is an implementation model where the cloud provider is accountable to execute code by distributing the resources dynamically [16]. A serverless code continues to run in stateless containers that can be activated by a wide range of events, which includes database events, http requests, monitoring alerts, scheduled events, file uploads, etc. In other words, a serverless let users build and execute the code without worrying about servers.

When it comes to serverless, you pay only for constant execution or throughput instead of by particular server component. In other words, a company that prefers backend services from the serverless provider is charged in terms of their computation, and the company does not have to stick and pay for a specified amount of bandwidth and servers. The primary purpose of the serverless is to concentrate on the applications rather than infrastructure.

In bygone days, anyone can create a web application required physical hardware to run a server, which is quite an expensive and cumbersome undertaking. That is why, the cloud concept came where fixed counts of servers and the server space can be rented remotely. Companies and developers who rent the fixed server units space usually over-purchase to make sure that a point in activity or traffic wouldn't go beyond their monthly restrictions, which can break their web applications. It all means that most of the server's spaces are not used.

That is why, cloud providers have introduced models which is called auto-scaling, but these models also end up with high cost. It let developers buy backend services on a pay-per-execution basis [17, 18], which means they only need to pay for used services.

The first cloud provider who is responsible for introducing the serverless computing in the cloud and this is called AWS Lambda [20, 21]. Afterward, there are many cloud providers seen, including Google Cloud Platforms (GCP) provides Google Cloud Functions, IBM provides IBM Cloud Functions which is based on OpenWhisk, Microsoft Azure provides Azure Functions, and etc.

Sometimes, a serverless referred to as "FaaS" or "Functions as a Service" because code is sent to the cloud provider for implementation which is generally in the type of a function [22]. It is similar to micro services in which a large program or code is divided into little cubes and manageable components that can be scaled and updated parallelly and separately.

But, FaaS takes this concept to the next level as it further divides the monolith. That is why FaaS has now become a trendy topic in the world of cloud computing.

FaaS has simplified code deployment and offers a means to accomplish the serverless dream by permitting developers to execute the program in reaction to events without maintaining or implementing a complex infrastructure. It means is that developers can implement their code within a set of functions, which one sent to the cloud provider to perform the task via a runtime environment. Afterward, the cloud provider handles the code function desires by operating multiple functions parallelly [5].

II. Related Work

Serverless developers or architects have performed their experiments to evaluate things, such as Big Data Analytics, Lambda performance, latency, scalability, source efficiency in numerous types of platform, the phase of architecture, flint and design patterns in the serverless.

Utilizing PyWren Josep Sampe [1] reimplementation of PyWren API known as IBM-PyWren, primarily executes MapReduce tasks. Dimitar Kumanov [2] has written cost, about tremendous performance, terms, and conditions of Serverless computing using AWS Lambda.

Liang Wang [3] examined source efficacy, functionality, and scalability of AWS, GCP, and Azure platform. Sanghyun Hong [5] anticipated system of six pattern layout to build security solutions. For every blueprint of design, they demonstrate a program that represents benefits and provides utilizing pattern support.

Youngbin Kim [6] uses AWS and Flint Lambda for serverless data analytics. Theo Lynn [7] has reviewed every serverless provider tools, for instance, AWS Lambda, Google Cloud Functions, Microsoft Azure Functions, Auth0 Webtask, Iron.io Ironworker, IBM Bluemix OpenWhisk and Galatic Fog Gestal Laser.

Joseph M. Hellerstein [8] was talking about the gap and challenges of Serverless system which focused on AWS Lambda

Garrett McGrath [10] executed some functions to evaluate the performance of different Serverless computing platform in AWS Lambda, Azure functions, Google Cloud Functions and IBM's deployment of Apache OpenWhisk then showed the metrics.

In our paper, we have designed a complete Serverless architecture for big data analytics, we considered storage, analysis, normalization, and etc.

III. Methods

A. Amazon S3

Amazon S3 stands for Amazon Simple Storage Service which is a high-speed, simple and scalable, storage service developed by Amazon Web Services or as we all know it as an AWS [23]. It comes with a simple interface that is used to stockpile and retrieve any amount of data anytime and anywhere on the internet. It provides any developer access to highly scalable, fast, reliable, and inexpensive storage infrastructure that is used by Amazon to execute its own e-commerce sites.

B. AWS Glue

AWS Glue is an advanced extract, transform and load (ETL) service that makes it easier to load and process data for analytics [24]. The creation and execution of the ETL task are done using a few clicks in the console of ASW management. The service consequently profiles data in its Glue Data Catalog (GDC) that is a metadata storage area for entire data assets that includes details, for instance, location, table definition, etc. A GDC can be used as a substitute to Apache Hive Metastore for Amazon Elastic MapReduce applications.

C. AWS Athena

AWS Athena is a service that let a data analyst to execute interactive queries on an Amazon web services cloud (public) on data saved in Amazon S3 [25]. As it is a serverless query and that's why an analyst does not have to manage any underlying infrastructure to use it. On the other hand, there is no requirement of loading S3 data into AWS Athena or transforming it for analysis. It has made it both faster and easier to analyze the data in Amazon S3 using SQL.

D. AWS QuickSight

Amazon QuickSight is the AWS business intelligence tool, which is launched in 2015. It let businesses to create and examine visualizations of their customer's data [26]. The service employs a super-fast in-memory calculation engine which runs parallel to execute data calculations. At present, it's providing a new pay-per-session pricing strategy for QuickSight dashboards access, which is mainly to give a little bit lift in the marketplace, where Microsoft's Power BI and Tableau have confined a significant part of the mindshare.

E. Amazon Kinesis

Amazon Kinesis is a fully organized AWS that offers loading and processing of big data in a real time environment. Kinesis can handle hundreds of terabytes per hour from a high amount of streaming data from various sources, which includes financial transactions, operating logs, and etc.

F. Amazon SageMaker

Amazon SageMaker allows programmers and data scientists to quickly construct, train and install machine learning models for analytical or predictive applications. It provides support for Jupyter notebook to have easy access to the sources of your data for analysis and exploration. In addition to it, the service offers optimized machine learning algorithms to run powerfully against a hugely significant amount of data in a distributed environment.

G. AWS Step Functions

AWS Step Functions (SFs) allows the users to coordinate, design, and run workflows using distributed applications. Step Functions offer a reliable way to organize parts, and step via the functions of the application. The SFs provides a graphical console that visualizes the parts of the application as sequential steps. It triggers and monitors each step automatically, and it retries when there are any errors so that the application executes in the right order. We can construct programs, and even, when our requirements change, we can reorganize or swap components without modifying any code.

IV. Serverless Architecture

There are three primary services in the cloud computing technology. First one is the price strategy which you pay only for the program execution. Second one is managed in which the cloud provider manages everything. The third is on-demand where we install and deploy the program as per your requirements.

Our architecture is serverless focused and also the managed service of AWS. The reason behind, AWS serverless does not come with all capabilities that are credible to design and implement a perfect data lake. That is why our architecture comes with managed services also.

Now, wondering what data lake is? It is a repository in which we begin to bring the whole enterprise data in one central place. It is vital because businesses or organizations deal with different kinds of data, which include both internal and external. Further, the data can be in numerous formats. A count of these may excel based data or, or API's. Plus, many businesses want to gather social media data, for instance, Twitter data, Facebook data, and many others. Presently, there is no existence of technology in the world that really can let us store, and analyze every type of data in a plex. That is why it come up with Data Lake, where you can save any data.

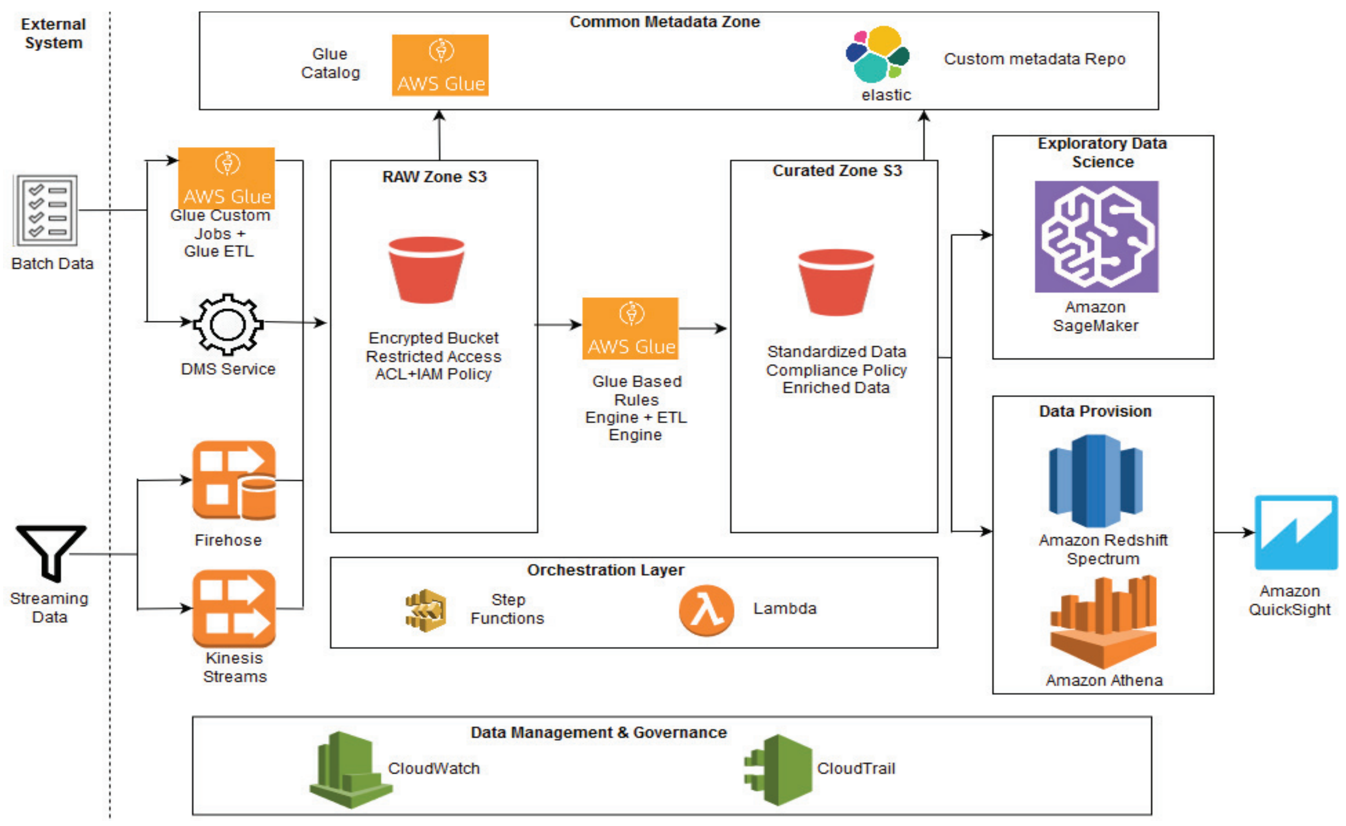

Fig. 1 : Serverless Architecture for Big Data Analytics

How do we design Data Lake?

To design a perfect data lake, we rely on 4 zones or layers. Why? Let's take an example; if you want to create a house or home, then every home should have minimum things, which include a bedroom, a kitchen, a living room, and a bathroom. Similarly, a perfect data lake has a minimum of four layers.

1st Layer: Raw zone: To bring the data

How we can bring and save data to our structure or system from various external sources? How can we bring relational or social media data? How can we bring our client's data? Our first step is to bring and store data from all external sources.

2nd Layer: Curated zone: To standardized the data

How we can curate the data? As data we bring is from different sources and that is why it is obvious to be present in various formats. Let's take an example: the date time format can be different. Some may be in GMT format, or some may be in EST format. So, before moving towards further steps, data curation is essential to ensure that the information is organized into the standardized and single format. It means the second layer of data lake is curation.

3rd Layer: Exploratory zone: To explore hidden data patterns

There are lots of unknown things in data. For example, how a social media platform, Twitter create an impact on the stock price. In the third layer of data lake, we are trying to find out the hidden things present within the data. Without any doubts that there are a lot of hidden patterns in the data. And the primary goal of designing data lake is to explore those hidden patterns. It is also known as an exploratory zone where we discovered new or unknown things. At this stage, we do some experiments with the data to find unfamiliar things.

4th Layer: Provisioning zone: To productionizing the data

The fourth layer of data lake is productionizing. Now, we run or execute our business or company applications. It is also known as provisioning zone that provides data to empower the businesses applications.

V. ARCHITECTURE IMPLEMENTATION

We rely on Amazon Web Service for the entire process. In our cloud architecture, we demonstrate that we can utilize Amazon Service, i.e. AWS Glue to bring batch data. We have also used other Amazon Services in our architecture, which includes Amazon Firehose, and Amazon Kinesis. The primary purpose of using these services in our system is to bring stream data. In addition to it, we rely on DMS, whose primary goal is to bring relational database data.

The first layer of Data Lake is how we can connect to various external sources to bring complete data to the raw zone. Here, we rely on S3 bucket of AWS for saving all the bringing data from numerous external sources. What is S3 bucket? If yes, then it is an Amazon service used to store data only.

Now, comes to curation zone where we standardized data. Here, we rely on AWS Glue, which is again Amazon service used to process or run business modifications and standardization on the collected data. In the second step, we modified the raw data into curated data using Glue Services.

The third step is to promote the data for discovery. Here, discovery means we give a close look at the curated data. For this, we use an Amazon Service, which is Amazon

SageMaker. With the help of Amazon SageMaker, we can perform some experiments with the data to find out the hidden, new or unknown insights within the data.

Eventually, the provision zone and for this, we use Redshift spectrum or Amazon Athena. Both services come with data warehousing abilities. By using these services, we can construct process tables to power our business applications.

That is all on how we design and implement our architecture. Meanwhile, we also provide data management and governance to design and implement a data lake. For this, we rely on Amazon CloudWatch & CloudTrail to track logs and every step in the application.

When it comes to metadata, we rely on Glue catalog and elasticsearch. With the help of these services, we can quickly capture metadata. There is no doubt that metadata is important because we have to understand what is present in the data lake. If people bring different types of data into the lake, then nobody will understand what is going on as it becomes a disorganized environment. For instance: there is a big book, but there is a table of contents in the book. So, there is a need for metadata layer. It helps people to understand what is present in the book, and where specific data is placed. The elasticsearch and Glue catalog can aid us to construct the table of contents in the data lake.

The last layer of our architecture is the orchestration layer. How we will bring all layers together? Or how do we execute all these layers in a single pipeline? That's the role of the orchestration layer. For this, we rely on AWS Lambda function and AWS Step Functions for connecting these processes.

TABLE I: DIFFERENCE BETWEEN TRADITIONAL AND SERVERLESS ANALYSIS

Traditional	**Serverless**
A traditional server is quite a large size computer that can be accessed over the web to provide access to services or information.	Serverless is a cloud computing model in which a cloud providers manages the allocation and provisioning of the servers dynamically.
In traditional servers, the code is written and hosted by the developers. In addition, they also supervise the operating system setup, application environment setup, and working too [17].	When it comes to serverless platform, developers only write code, and then, these are executed in the cloud directly without worrying about operating systems, hardware, servers, etc [17]. .
In traditional servers, developers have to configure auto-scaling and load balancing manually.	In the serverless platform, developers don't have to worry about the scalability. The cloud provider handles everything related to auto-scaling and load balancing.
Here, users are charged for every hour no matter whether the server is in use or not.	Reduced cost is one of the significant benefits of using serverless computing. Here, users are charged only for the code execution [27].

VI. CONCLUSION

At the end of this research, it may be said that serverless platform is very vital for the 21st century's data computing. Since, in the future big data ocean is being created, so it is another challenge to process those data with less cost and high speed. In our architecture, we have shown the whole process of collecting and storing data from various sources. Then prepare the data using AWS Glue for processing. And we also show processing and analysis steps using serverless components where in traditional system has more complexity. By relying on serverless architecture, big data analytics will become easier with minimum cost.

REFERENCES

[1] Y. Kim and J. Lin, "Serverless Data Analytics with Flint," *IEEE Int. Conf. Cloud Comput. CLOUD*, vol. 2018–July, pp. 451–455, 2018.

[2] L. Wang, M. Li, Y. Zhang, T. Ristenpart, and M. Swift, "Peeking Behind the Curtains of Serverless Platforms," *2018 USENIX Annu. Tech. Conf. (USENIX ATC 18)*, pp. 133–146, 2018.

[3] G. Adzic and R. Chatley, "Serverless computing: economic and architectural impact," pp. 884–889, 2017.

[4] G. McGrath and P. R. Brenner, "Serverless Computing: Design, Implementation, and Performance," *Proc. - IEEE 37th Int. Conf. Distrib. Comput. Syst. Work. ICDCSW 2017*, pp. 405–410, 2017.

[5] T. Lynn, P. Rosati, A. Lejeune, and V. Emeakaroha, "A Preliminary Review of Enterprise Serverless Cloud Computing (Function-as-a-Service) Platforms," *Proc. Int. Conf. Cloud Comput. Technol. Sci. CloudCom*, vol. 2017–Decem, pp. 162–169, 2017.

[6] J. Sampé, G. Vernik, M. Sánchez-Artigas, and P. García-López, "Serverless Data Analytics in the IBM Cloud," pp. 1–8, 2018.

[7] J. M. Hellerstein *et al.*, "Serverless Computing: One Step Forward, Two Steps Back," vol. 3.

[8] E. van Eyk, A. Iosup, S. Seif, and M. Thömmes, "The SPEC cloud group's research vision on FaaS and serverless architectures," no. Section 2, pp. 1–4, 2017.

[9] D. Kumanov, L.-H. Hung, W. Lloyd, and K. Y. Yeung, "Serverless computing provides on-demand high performance computing for biomedical research," 2018.

[10] S. Hong, A. Srivastava, W. Shambrook, and T. Dumitraș, "Go Serverless: Securing Cloud via Serverless Design Patterns," *USENIX Work. Hot Top. Cloud Comput.*, 2018.

[11] B. Wagner and A. Sood, "Economics of Resilient Cloud Services," *Proc. - 2016 IEEE Int. Conf. Softw. Qual. Reliab. Secur. QRS-C 2016*, pp. 368–374, 2016.

[12] S. Hendrickson, S. Sturdevant, T. Harter, V. Venkataramani, A. C. Arpaci-dusseau, and R. H. Arpaci-dusseau, "Serverless Computation with OpenLambda 1 Introduction 2 Lambda Background 3 Lambda Workloads," *USENIX Work. Hot Top. Cloud Comput.*, 2016.

[13] M. Villamizar *et al.*, "Cost comparison of running web applications in the cloud using monolithic, microservice, and AWS Lambda architectures," *Serv. Oriented Comput. Appl.*, vol. 11, no. 2, pp. 233–247, 2017.

[14] D. A. Kumari and N. Tejaswani, "Vector Quantization for Privacy during Big Data Analysis and for Compression of Big Data," vol. 4, no. 4, pp. 421–428, 2015.

[15] "Why Serverless?" [Online]. Available: https://serverless.com/learn/overview/. [Accessed: 30-Mar-2019].

[16] O. Alqaryouti and N. Siyam, "Serverless Computing and Scheduling Tasks on Cloud : A Review," pp. 235–247.

[17] "What is Serverless Architecture? What are its Pros and Cons?" [Online]. Available: https://hackernoon.com/what-is-serverless-architecture-what-are-its-pros-and-cons-cc4b804022e9. [Accessed: 30-Mar-2019].

[18] "Serverless Computing – Amazon Web Services." [Online]. Available: https://aws.amazon.com/serverless/. [Accessed: 30-Mar-2019].

[19] "What is Serverless Architecture and why should you care ?" [Online]. Available: https://medium.com/@anandujjwal/what-is-serverless-architecture-and-why-should-you-care-bcf83069eb38. [Accessed: 30-Mar-2019].

[20] M. Chan, "Serverless Architectures: Everything You Need to Know," *Thorn Technol.*, Mar. 2017.

[21] R. Miller, "AWS Lambda Makes Serverless Applications A Reality," *TechCrunch*, Nov. 2015.

[22] "What Is Serverless Computing? | Serverless Definition | Cloudflare." [Online]. Available: https://www.cloudflare.com/learning/serverless/what-is-serverless/. [Accessed: 30-Mar-2019].

[23] "Cloud Object Storage | Store &amp; Retrieve Data Anywhere | Amazon Simple Storage Service." [Online]. Available: https://aws.amazon.com/s3/. [Accessed: 30-Mar-2019].

[24] "What Is AWS Glue? - AWS Glue." [Online]. Available: https://docs.aws.amazon.com/glue/latest/dg/what-is-glue.html.

[Accessed: 30-Mar-2019].
[25] "Amazon Athena — Serverless Interactive Query Service - AWS." [Online]. Available: https://aws.amazon.com/athena/. [Accessed: 30-Mar-2019].
[26] "Amazon QuickSight." [Online]. Available: https://aws.amazon.com/quicksight/. [Accessed: 30-Mar-2019].
[27] "Why use Serverless Computing? | Pros and Cons of Serverless | Cloudflare." [Online]. Available:https://www.cloudflare.com/learning/serverless/why-use-serverless/. [Accessed: 30-Mar-2019].