Serverless Computing

Mr. Makarand Karambelkar[1] Miss. Mahima Chourasiya[2] Mr. Deepesh Rohit[3] Prof. Kavita Namdev[4] Prof. Shaifali Srivastava[5]
[1,2,3]Student [4,5]Assistant Professor
[1,2,3,4,5]Department of Computer Science and Engineering
[1,2,3,4,5]Acropolis Institute of Technology and Research Indore, India

Abstract— In late 2014, Amazon Web Services launched the 'Lambda' service. Since then, each of the major cloud computing infrastructure providers has launched services that support a similar model of deployment and operation, where users are able to deploy individual functions rather than deploy and run monolithic applications or dedicated virtual machines, and only pay for the time their code is actually executing. Serverless computing has arose as a powerful new model for application and device deployments. It represents an evolution of cloud programming models, abstractions and frameworks, and bears witness to the maturity and wide adoption of cloud technologies. Here we survey current serverless systems from industry, academics, and open- projects in this segment, recognize key features and use cases, and explain technological difficulties and open issues.

Keywords: Serverless, Cloud Computing, Lambda Amazon web services, FaaS (Function as a Service)

I. INTRODUCTION

Serverless computing (or simply serverless) is evolving as a new and persuasive standard for deploying cloud applications, largely due to the recent shift to containers and microservices of enterprise application architectures. This is an indication of the increasing attention gained by serverless computing in industry trade shows, meetups, blogs, and the community of development. By contrast, the attention in the academic community has been limited. This paradigm change poses both an opportunity and a challenge from an Infrastructure-as - a-Service (IaaS) consumer perspective. In the one side, it provides a streamlined programming model for developers to build cloud applications that abstract most, if not all, operational concerns; reduces the cost of delivering cloud code by costing execution time rather than resource allocation; and is a quick delivery framework for small pieces of cloud-native code.

Digital innovation has scaled up business models. New software and services are developed at a rapid rate, and the resultant demand on businesses to deliver new apps and innovations is intense.

The expectations of customers are also very high and this adds to the current stress on companies. The new digital transformation environment for serverless computing has created an easy entry path. In this accelerated world, building components in-house, investing in expensive hardware, installing servers, configuring and troubleshooting are overheads that businesses can do without very well. In response to the macro shifts, businesses are rethinking their products on a micro basis. Instead of static, monolithic packages, they concentrate on microservices and practical executions. The third significant aspect which has played a role in serverless computing evolution is Continuous Integration (CI). CI helps companies adapt agilely to market and economic needs. Weekly and bi-weekly sprints involving constantly changing goods involve an infrastructure reconsider. Developers want technology that can deal with regular but minor improvements that can be made available to them, that can be self-managed and scalable.

II. LITERATURE SURVEY

Amazon Web Services will announce the first public Serverless framework called 'Lambda' late 2014. By then, every big cloud service provider has been operating on some sort of serverless architecture, some of them have also launched the original iteration of their systems where, instead of installing and running monolithic applications or specialized virtual machines, users can install specific functions and only pay for the time their application is actually running. Under the marketing phrase 'serverless,' these innovations are gathered and the vendors indicate that they have the ability to radically improve how client / server systems are built, developed and run.

For service or platform to be considered serverless, it should provide the following capabilities:

- No server management-No servers need to be planned or maintained.
- Flexible scaling – Server scalability can be dynamically controlled by you or by changing the power by toggling usage units rather than individual storage units.
- High performance – There is built-in flexibility and error tolerance for serverless applications.
- No idle capacity – The main service they provide is that you pay for how much you've used it. In other words, you will never be paying for the unused power for the duration your program is unused. There is no need for pre-planning or over-planning resources for things like processing and storing. various Deployment Methods and Techniques
- On premise deployment
- Cloud Deployment
- Cloud Deployment Techniques
- Dedicated virtual servers
- Containers with Orchestrator
- Serverless Computing

A. Research methodology and Design Process

To introduce serverless feature one requires a Cloud service provider account membership that can be billed to Azure OR AWS accounts so he / she can install his / her code there and compare various deployment aspects. First two goals can be accomplished by direct delivery of applications on AWS or Azure. For the third goal one has to provide an HTTP trigger like azure feature. Below are the steps how to create an Lambda function from AWS portal.

- Start by opening AWS portal and go to the AWS Functions option or service and follow the instructions.
- It will ask you for the Terminologies or features you want to integrate.

In a serverless web app, there may be a mixture of running processes in a serverless web app that decide the user's conceptual and personal elements to provide content and functionality that meets the needs of the user. For e.g., in this use case, static content may be saved in S3 for display when the application is opened in the browser.
Simultaneously, processing is initiated via the application's API gateway to run Lambda functions which decide the context of the application user. The static content is then enhanced with more dynamic information created by the lambda functions, and stored in DynamoDB as dynamic data.

1) Requirements of AWS Services

Amazon Web Services provides a wide variety of online cloud-based offerings for different market uses. The goods provide servers, databases, software, networking, mobile devices, programming tools, business apps, with a pricing model pay-as-you-go.

- Set up Simple storage services (S3): The html, CSS, and JavaScript files are processed using S3 resources. Hosting static websites is the main usage for S3.
- Set up Lambda Function: The lambda function is used to render serverless applications. Lambda is a computational service offered by aws that allows lightweight use. It's flexibility in both the respective API-Gateway and DynamoDB.
- Set up Internet-Gateway: An Internet Gateway is a functional connection between the internet and an Amazon VPC. The Internet gateway is simply the network entry point.
- Set up DynamoDB: DynamoDB is primarily used for professional information storage. The lambda function launches a DynamoDB for record collection, deleting, and updating.
- Set up API-Gateway: It is the transitional framework between end-user interface and lambda interface. It consists of getting and uploading methods for data collection and posting to DynamoDB, respectively.
- Set up CloudFront: Amazon CloudFront is a network of content management systems (CDN) that Amazon Web Services provides. Content distribution networks provide a globally dispersed network of proxy servers that archive content more directly to users, such as YouTube videos or other bulky media, thereby improving access speed for accessing the content. It grabbed the world record for easy exposure.

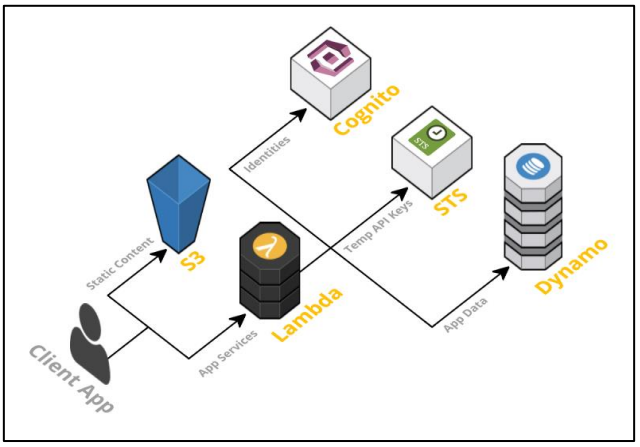

Fig. 1: The Basic Architecture for the Serverless Website hosting.
(Source: http://blog.tonyfendall.com/2015/12/serverless-architectures-using-aws-lambda/)

B. Why use this serverless architectures?

The developers will concentrate on their main application by implementing this serverless architecture instead of caring about maintaining and running servers or runtime, either in the cloud or on-site. This can overhead helps developers to recover time and resources that can be expended on creating efficient, scale-up and stable goods.

III. RESULT

Serverless computation is a new technology, and it has yet to assess the potential. When serverless vendors manage to solve the complexities of FaaS infrastructure, the platform will dramatically transform the production of apps in the coming years.

IBM analysts are expecting a seven- to ten-fold rise in the serverless market by 2021. In addition, Businesses and Forecasts are now forecasting significant serverless business growth to $7.72bn by 2021.

Gartner expects that serverless computing will plateau in two to four years and be introduced in areas outside IT. Additionally, Tim Wagner, AWS Lambda's general manager, announced at Serverless Conf 2017 that serverless adoption goes ten times faster than container adoption. He also noticed an remarkable increase in FaaS adoption in enterprises.

IT market analysts also expect that by the end of 2018 we will see the first serverless device usages. Serverless computing can be used to create applications which are rich in images, virtual assistants, chatbots, mobile apps and websites. In addition, the technology can be extended to the management of device backends, media and files, IT automation and real-time streaming of data.

In addition, Serverless computing is expected to accelerate the development of advanced-technology applications. Its system in particular, due to its dynamic auto-scaling, can greatly reduce the complexities of big data. It is better suited for applications not running constantly but instead having calm times and traffic peaks. While serverless systems can be used to build apps that interact seamlessly with various emerging technology, such as

devices for the Internet of Things, cognitive intelligence, data processing, and mobile devices.

Serverless systems require infrastructures where they can be implemented, vendor agnostic architectures offer an agnostic network means of defining and installing Serverless technology on various cloud networks or business providers.

- Serverless Framework (JavaScript, Python, Golang)
- Apex (JavaScript)
- ClaudiaJS (JavaScript)
- Sparta (Golang)
- Gordon (JavaScript)
- Zappa (Python)
- Up (JavaScript, Python, Golang, Crystal)

IV. Conclusion

Serverless computing is a new approach to software development and is exciting. The system reduces the burden of server management and lowers cost of production. Thanks to pay-per-use pricing, it helps developers concentrate on growth and avoid thinking about budget restrictions. While serverless computing is a modern technology, full of challenges, it has already been used by the Apriority team in our projects. When you are looking for app creation that is cost-effective and high-quality and time-efficient, we will give you our services.

In this chapter we looked in depth at the origins and past of serverless computing. It is an extension of the movement towards higher rates of abstractions in cloud programming frameworks, and currently exemplified by the Feature-as - a-Service (FaaS) model where developers write tiny stateless code snippets and allow the framework to handle the complexities of executing the task scalably in a fault-tolerant manner.

Nonetheless, this seemingly limiting paradigm fits a range of common distributed application trends well, including compute-intensive pipelines for event processing.

Serverless architecture is definitely really interesting, but there are a lot of drawbacks to it. As the quality and performance of architectures depends on the criteria of the market and by no means on technologies alone. Similarly, when used in the right position, Serverless will shine.

References

[1] Aws lambda.
URL https://aws.amazon.com/lambda/
[2] S3 Simple Storage Service.
URL https://aws.amazon.com/s3/
[3] Building Serverless Apps with Webtask.io.
URL https://auth0.com/blog/
building-serverless- apps-with- webtask/.
[4] Aws re: invent 2019 — (mbl202) new launch: Getting started with aws lambda. URL https://www.youtube.com/watch?v=TOn0xhev0Uk

