2018 3rd International Conference for Convergence in Technology (I2CT)
The Gateway Hotel, XION Complex, Wakad Road, Pune, India. Apr 06-08, 2018

Winning in the era of Serverless Computing and Function as a Service

Mohit Sewak
Artificial Intelligence SME
Pune, India
Email: mohitsewak17@gmail.com

Sachchidanand Singh
Advanced Anaytics & BI SME
Pune, India
Email: sach.success@gmail.com

***Abstract*— Serverless Computing and Function as a Service (FaaS) is gaining traction in cloud-based application architectures used by startups and matured organizations alike. Organizations that are keen to leverage modern technology to gain a disruptive edge, optimal efficiency, advanced agility and save cost are adopting these architectural styles rapidly. Cloud service provider offer and dynamically manages the allocation of machine resources in serverless computing. The serverless architectures allows the developers to focus on business logic exclusively without worrying about preparing the runtime, managing deployment and infrastructure related concerns. FaaS may be assumed as a subset of Serverless Computing, in which, instead of coding a full-fledged cloud based application, the developer just writes (often small) functions which are piece of code (in one of the multiple programming languages supported by the platform) dedicated to do a focused, often single task that are invoked by triggers. It offers dynamic allocation and scaling of the resources and innovative trigger based costing model. This paper introduces Serverless Computing, and Function as a Service (FaaS), explores its advantages and limitations, options available with popular cloud and Platform as a Service (PaaS) providers, and emerging use cases and success stories.**

***Index Terms*— Serverless Computing, Functions as a Service(FaaS), Platform as a Service (PaaS), Microservices, IBM Cloud Functions, Amazon AWS Lambda, Microsoft Azure Functions, Google Cloud Functions, Apache OpenWhisk.**

I. Introduction

Cloud offers scalability on demand and quick setup of virtual servers but maintaining software stack on these virtual servers is time consuming for developers and system administrators. In serverless, developers don't need to setup and administer the servers running back-end applications. The back-end infrastructure is maintained by third party providers and the functionality needed is offered as services [1], [2].

Using Serverless computing, developers can develop applications at rapid pace with reduced costs without worrying about infrastructure. The servers still exist in Serverless computing but managed dynamically based on demand by Cloud service provider instead of the application owner.

In Function as a Service(FaaS), business logic is broken down into pieces of small functions and when a pre-defined event occur they trigger execution of the function. The trigger could be defined as call to registration service HTTP endpoint or a message published in message queue [3].

Function as a Service (FaaS) is a small, discrete, reusable chunk of code which is stateless compute container modeled for an event-driven solution. In serverless computing model, code still technically runs on servers but the user is not responsible for managing the servers. Function as a Service (FaaS) is basically a combination of self-managed functions and cloud services that serve as the Serverless application [4], [5], [6].

The event or trigger based computing helps to cut cost and enables more efficient application development since additional resources are provisioned only when a pre-defined event occurs which ensures resources don't sit idle. The allocation of resource on cloud is managed by service provider dynamically and billing is based on the actual resources consumed by a running application [7], [8].

II. Microservices And Serverless

The large enterprise applications are broken down into smaller services in the Microservices approach and allows to build and deploy individual services without impacting other services.

Microservices are deployable loosely coupled, independent service which together form an application. The microservices approach helps small teams to work independently to deliver and deploy their services in the large complex projects. The microservices architecture provides better fault isolation and applications built using microservices architecture can be broken down into component services and redeployed independently [9], [10].

Serverless is pay as you go which makes it attractive option for startups due to flexible pricing and reduced operational cost. Serverless is more granular than microservices and provides higher degree of functionality with less time to market but supports fewer programming language [11].

Microservices provides a better approach to service oriented architecture (SOA), but serverless is gaining ground in event-based architecture. The combination of microservices and serverless architecture is probably the best approach [12].

Serverless computing is a key enabler for microservices-based applications since infrastructure demand is driven by the needs of individual services. Serverless computing have more to do with infrastructure and elastic scalability while in microservices approach an application is collection of services which implements business capabilities. We can provide

978-1-5386-4273-3/18/$31.00 ©2018 IEEE

microservices architecture on top of Severless deployment model [13].

III. Serverless Computing

Serverless computing allows us to build and run applications and services without worrying about server provisioning, scaling and how to manage it. The infrastructure can scale to any extent with manual or auto provisioning of new servers on sudden spike in usage. It saves development time and helps developers to focus on their core business logic coding instead of worrying about managing and operating servers or runtimes [14].

A. Advantages of Serverless Computing

Serverless approach is very compelling from business perspective and some of the key benefits of serverless architecture are zero-maintenance and easy code deployment [15]. Serverless allows to run applications without worrying about server provisioning, scaling and provides improvement in time-to-market etc.

B. Limitations in Serverless Computing

Serverless is stateless by design and hence it should manage states outside functions which means no more in-memory cache [5]. And higher the level of granularity we get with serverless, the integration testing becomes more complicated and it's difficult to debug and troubleshoot. The higher level of granularity also affects tool and frameworks to be used with Serverless [16].

C. Independent Serverless Framework

There is very high degree of lock-in by cloud service providers in the Serverless computing. The open source frameworks try to mitigate lock-in by leveraging vendor neutral shims to translate numerous services. A few examples of independent serverless frameworks are Kubeless, Funktion and Fission etc [17].

D. Other Considerations in Serverless Computing

The factors like authentication, authorization, system integrity and communication integrity between system components are critical aspects from security standpoint.

E. Serverless Offerings

Major serverless offerings are from IBM, Amazon, Microsoft and Google. IBM's Serverless computing offering is IBM Bluemix which is based on Cloud Foundry open technology and runs on SoftLayer infrastructure. Google's Cloud ML Engine provides serverless machine learning services which is built on custom Google hardware (Tensor Processing Units).

Microsoft's Azure Cosmos DB is multi-model database service for serverless applications. Microsoft's Azure Functions Proxies can be leveraged to create microservice architectures. Amazon provides a messaging service called Amazon SNS which makes easy to scale microservices and serverless applications. Amazon Kinesis offers services to load and analyze streaming data and we can build custom streaming data applications for specialized needs.

F. Serverless Architecture

As shown in Figure 1, feeds from package create triggers and a series of rule based action is initiated which finally invoke service ecosystem components like 3rd party services or self-enabled services. Also, events can be triggered using Command Line Interface(CLI), User Interface(UI) or iOS SDK etc [15].

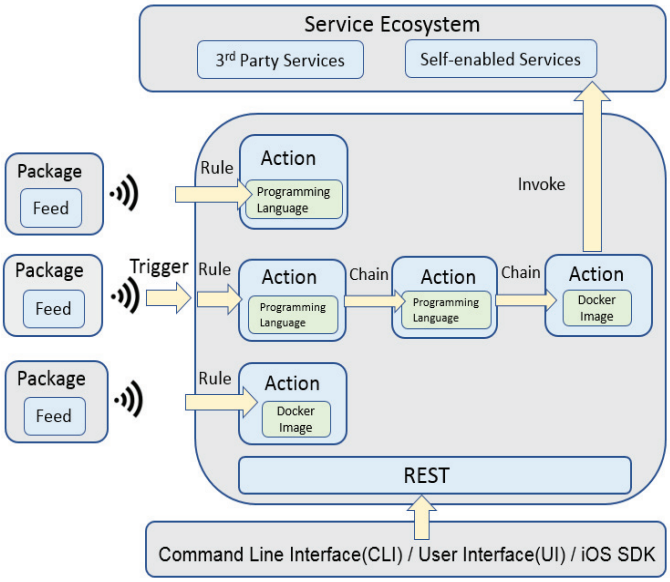

Fig 1: Serverless Architecture [15]

IV. Function As A Service

MarketsandMarkets research firm says Function-as-a-Service(FaaS) Market will be worth US $7.72 billion by 2021 [18]. The FaaS market share will be dominated by developer-centric FaaS and key market drivers will be auto-scaling, instant provisioning and pay-as-you-go model [18].

A. What is Function as a Service

In Function as a service (FaaS), similar tasks are grouped together and their business logic is written into small pieces of code called functions. And when a pre-defined event occurs it triggers the execution of the function.

IBM Cloud Functions and AWS Lambda support Node.js, Python, Java and Go [19], [20]. Google Cloud Function is written in JavaScript and executes in Node.js runtime while Microsoft Azure Function supports JavaScript and C# [21], [22].

The most common FaaS scenario could be scheduled tasks, web requests or queue message processing. It's expected to start within fraction of second to process requests and then process is finished [23].

B. Advantages of Function as a Service

Using FaaS we can isolate high volume transactions, for example if we have a repetitive transaction which happens

several times per second then it makes sense to write its logic to a separate function and scale it as per requirements. FaaS reduces operational costs, time to market, smaller cost to scale, fits with microservices and simplifies packaging and deployment etc [1].

In pay per execution model, billing is based on total number of execution requests made for each function while in pay per instance model, billing is based on number instances of the resource allocated. It makes sense to go for pay per execution from cost perspective.

C. Limitations in Function as a Service

In Function as a service (FaaS), business logic is written into small piece of code called functions and total count of functions will vary depending upon the complexity of business logic.

At times it's difficult to manage lots of functions deployed and calls for better tools for monitoring, diagnostics and debugging. There are tools which allow remote debugging or a mirrored local development environment but there is still a need for improved tooling [3].

We have limited control in Function as a Service(FaaS) since one could just write function not complete execution scripts and invocation is made by triggers, and not direct parametrized invocation as in case of a Microservices deployed on serverless.

V. Function as a Service(FaaS) Offerings

Major FaaS offerings includes IBM Cloud Functions, Amazon AWS Lambda, Microsoft Azure Functions and Google cloud functions. Next, we will cover each of their offerings in brief.

A. IBM Cloud Functions

IBM cloud function is event-driven service which runs application in response to an events or direct invocations from web or mobile applications. IBM cloud function supports Node.js, Python, Java, Swift and Go.

IBM cloud functions is a programming platform based on Apache OpenWhisk for developing lightweight code to execute on demand and it can leverage the benefits of cognitive services by providing easy access to IBM Watson® APIs [24], [25].

B. Amazon AWS Lambda

The code which is run on AWS Lambda is called Lambda function which is stateless. We create Lambda function by uploading code to AWS Lambda, then choose memory, timeout period, AWS Identity and Access Management (IAM) role and specify AWS resource to trigger the function. AWS Lambda will run the deployed function whenever the resource changes [26], [27].

C. Microsoft Azure Functions

Microsoft Azure functions is compute on demand experience driven by events. Azure Functions integrates with Azure Logic Apps in the Logic Apps Designer and allows us to use the computing power of Functions in orchestrations with other Azure and third-party services. We can create functions which will execute based on events created via Azure Cosmos DB, Github webhook, HTTP and Blob etc [28], [29].

D. Google Cloud Functions

Google Cloud Functions restricts the total number of functions to 1,000 that can be deployed per project and 540 seconds is maximum amount of time a function can run before it's forcibly terminated. The Google Cloud Functions in beta was launched in 2017 [30], [31].

VI. Emerging Use Cases of Serverless and FaaS

There are some interesting use cases on how Serverless and Function as a Service(FaaS) is being used for Mobile Backend Development, Data Processing, Cognitive Processing, Supply Chain Engineering, Streaming Analytics & Internet of Things etc.

A. Cognitive Processing

The drones taking aerial pictures and processing them using cognitive APIs to analyze information captured in pictures. It's useful for insurance companies to document flood and fire risks etc. IBM Watson ML algorithms and OpenWhisk together can be used to perform cognitive tasks on a multimedia files. We can use cognitive processing in a serverless environment to identify extent of property damages and speed up insurance payments and risk assessments [32].

B. Serverless IoT Backend

Using Serverless IoT backend we can manage our growing Internet of Things(IoT) workloads and build a scalable solution. We can leverage AWS IoT Rules Engine and its seamless integration with AWS Lambda to create a flexible and scalable IoT backend powered by Amazon DynamoDB [33].

C. Supply Chain Engineering

Microsoft Supply Chain Engineering (SCE) uses Azure serverless to effectively manage supply chains for all software and hardware supply and demand. SCE uses Azure serverless technologies to create services to search near real time cross-functional data without buying new infrastructure or writing new piece of code [34].

D. Serverless Mobile Backend

We can leverage Serverless to separate mobile frontend and backend development by creating serverless microservices with API backends. Google Cloud Functions can be used from Firebase to extend application functionality without spinning up a server. We can run code in response to user actions and events to keep users engaged with event-based notifications and network intensive tasks can be offloaded to Google Cloud Functions [35].

VII. Customer Success Stories

A. SiteSpirit and IBM Cloud Functions

MediaSpirit is a tool offered by Dutch development shop SiteSpirit to manage vast amounts of media on the cloud. The

unnecessary maintenance cost and manual effort was saved by implementing MediaSpirit using OpenWhisk along with IBM Cloudant and API Connect. It helped to achieve cost reduction with a serverless architecture which runs when users need it [36].

B. Netflix and Amazon AWS Lambda

Netflix is one of the largest online media streaming provider which serves around 50 million customers and handles petabytes of streaming video, their AWS stack is made up of up to 50,000 instances in 12 zones.

Netflix used Lambda to execute rule-based checks, routing to handle data and security checks of new instances created. Netflix build rule-based self-managing infrastructure using AWS Lambda which can replace inefficient processes and reduce error rates to save time [37], [38].

C. Costa Farms and Microsoft Azure Functions

Costa Farms is a Miami based company which grows houseplants. The pH level is key factors in plants health but it's difficult to measure pH levels throughout day and across plant lifecycle. And to increase plant health by nutrient uptake, pH needs to be closely monitored and adjusted in real time.

Costa Farms implemented a smart IoT solution which allows them to monitor important vital growth statistics of their nursery plant in real time using Arduino Devices, Azure IoT Hub with pH sensor devices, Azure Functions [39].

D. Auger Labs and Google Cloud Functions

Auger Labs is a mobile apps-as-a-service company for the art community and provides artists with beautifully designed, custom-branded android and iOS apps tailored with their own content and featuring advanced technology.

Every time an image is uploaded to Firebase Storage, a Cloud Function executes to create thumbnails for enhanced mobile app responsiveness and finds a searchable label for the image to help art lovers in searching interesting artwork. And Google Cloud Functions send notifications to Slack for monitoring of events and notify customers via email when their custom apps are available. Auger Lab chose Cloud Functions for Firebase as it's easy to setup, develop, and scale effortlessly [40].

VIII. Summary and Conclusion

Serverless and Function as a Service(FaaS) could provide a disruptive competitive edge, improved efficiency and cost savings for organizations. MarketsandMarkets research firm says that Function-as-a-Service (FaaS) market size is estimated to grow at a Compound Annual Growth Rate (CAGR) of 32.7% [18].

There is need to develop better independent Serverless frameworks to mitigate high degree of lock-in by cloud service providers in the Serverless computing. The open source frameworks try to mitigate lock-in by leveraging vendor neutral shims to translate numerous services.

The next step for Serverless computing should be to enable data services to easily tie into Serverless frameworks so that all notifications from variety of data services could trigger Serverless functions and work seamlessly in different cloud environment. And there is a need to educate developers on emerging cloud applications for faster Serverless adoption [19].

References

[1] FaaS, PaaS, and the Benefits of the Serverless Architecture, https://www.infoq.com/news/2016/06/faas-serverless-architecture

[2] Serverless Computing Is a Growing Trend — Here's What You Need to Know, http://www.govtech.com/opinion/Serverless-Computing-Is-a-Growing-Trend-Heres-What-You-Need-to-Know.html

[3] An Introduction to Serverless and FaaS (Functions as a Service), https://medium.com/@BoweiHan/an-introduction-to-serverless-and-faas-functions-as-a-service-fb5cec0417b2

[4] Serverless Architecture: The Future of Business Computing, https://www.marutitech.com/serverless-architecture-business-computing/

[5] Serverless Architectures, https://martinfowler.com/articles/serverless.html

[6] Introducing Functions as a Service (FaaS), https://dzone.com/articles/introducing-functions-as-a-service-faas

[7] Serverless Computing, Explained, https://www.booleanworld.com/serverless-computing-explained/

[8] Serverless: The future of cloud computing? https://www.cio.com/article/3244644/cloud-computing/serverless-the-future-of-cloud-computing.html

[9] What are the benefits of a microservice implementation? https://www.mindstick.com/Articles/12701/what-are-the-benefits-of-a-microservice-implementation

[10] Benefits & examples of microservices architecture implementation, https://medium.com/@Apiumhub/benefits-examples-of-microservices-architecture-implementation-a5b49b6d0934

[11] Serverless vs. Microservices: What you need to know for cloud, http://www.computerweekly.com/blog/Ahead-in-the-Clouds/Serverless-vs-Microservices-What-you-need-to-know-for-cloud

[12] Serverless vs. Microservices: What you need to know for cloud, http://itknowledgeexchange.techtarget.com/ahead-in-the-clouds/serverless-vs-microservices-what-you-need-to-know-for-cloud/

[13] Microservice & Serverless Architectures – Is Either Right for You? https://www.3pillarglobal.com/insights/microservices-and-serverless-architectures-a-primer-on-whats-right-for-you

[14] What is Serverless Computing? https://www.serverless360.com/blog/what-is-serverless-computing

[15] What makes serverless architectures so attractive? https://developer.ibm.com/opentech/2016/09/06/what-makes-serverless-attractive/

[16] Serverless architectures comparison, pros & cons, and case studies, https://agileengine.com/why-should-consider-a-serverless-architecture-for-your-next-project/

[17] A Serverless Computing Primer: A Comparison, https://dzone.com/articles/a-serverless-computing-primer-a-comparison

[18] Function-as-a-Service Market worth 7.72 Billion USD by 2021, https://www.marketsandmarkets.com/PressReleases/function-as-a-service.asp

[19] From Silos to Services: Cloud Computing for the Enterprise, http://itknowledgeexchange.techtarget.com/cloud-computing-enterprise/the-evolution-of-serverless

[20] Getting Started with IBM Cloud Functions, https://console.bluemix.net/openwhisk/

[21] Lambda Execution Environment and Available Libraries, https://docs.aws.amazon.com/lambda/latest/dg/current-supported-versions.html

[22] Writing Cloud Functions, https://cloud.google.com/functions/docs/writing

[23] What Is Function-as-a-Service? Serverless Architectures Are Here! https://stackify.com/function-as-a-service-serverless-architecture/

[24] IBM® Cloud Functions- Getting started, https://console.bluemix.net/docs/openwhisk/index.html#getting-started

[25] What is IBM Cloud Functions? https://www.ibm.com/cloud/functions

[26] What Is AWS Lambda? https://docs.aws.amazon.com/lambda/latest/dg/welcome.html

[27] AWS Lambda Features, https://aws.amazon.com/lambda/features

[28] An introduction to Azure Functions, https://docs.microsoft.com/en-us/azure/azure-functions/functions-overview

[29] Azure Functions Documentation, https://docs.microsoft.com/en-us/azure/azure-functions/

[30] Google Cloud Functions, https://cloud.google.com/functions/

[31] Google Cloud Platform- Quotas, https://cloud.google.com/functions/quotas

[32] 5 Emerging Use Cases of the IBM OpenWhisk Serverless Platform, https://thenewstack.io/ibms-openwhisk-serverless/

[33] Implementing a Serverless AWS IoT Backend with AWS Lambda and Amazon DynamoDB, https://aws.amazon.com/blogs/compute/implementing-a-serverless-aws-iot-backend-with-aws-lambda-and-amazon-dynamodb/

[34] Supply chain dreams: Microsoft SCE uses serverless computing to drive innovation, https://customers.microsoft.com/en-in/story/microsoft-professional-services-azureserverless

[35] Serverless application backends, https://cloud.google.com/functions/use-cases/serverless-application-backends

[36] SiteSpirit, https://www.ibm.com/case-studies/sitespirit

[37] Netflix & AWS Lambda Case Study, https://aws.amazon.com/solutions/case-studies/netflix-and-aws-lambda/

[38] Who's Using AWS Lambda? https://www.contino.io/insights/whos-using-aws-lambda-1

[39] IoT pH flow controls and automation with Costa Farms, https://microsoft.github.io/techcasestudies/iot/2016/11/03/CostaCorp.html

[40] Case Studies: Cloud Functions for Firebase, https://firebase.google.com/docs/functions/case-studies