International Journal of Future Generation Communication and Networking
Vol. 13, No. 3, (2020), pp. 3291–3297

Function as a Service in Cloud Computing: A survey

Rajeshwar Mukund and Prof. Rajesh Bharati

Department of Computer Engineering

Dr. D. Y. Patil Institute of Technology
rajeshwar.mukund@gmail.com
rdbharati@gmail.com

Abstract

With the evolution of Cloud Computing and it's use as a cloud user, Could Computing can be consider as various services in different aspects such as Faas - Function as a Service, SaaS - Software as a Service, PaaS - Platform as a Service, IaaS - Infrastructure as a Service. In this paper we will explore Cloud Computing as FaaS - Function as a Service and various use case to use this context in Cloud Computing.

Keywords: *Cloud Computing, IaaS, PaaS, SaaS, FaaS, Lambda, Cloud Function, Azure Function, Iron Function, Open Lambda.*

1. Introduction

Cloud can be viewed as various Services aspects in terms architectural layers of Cloud Computing. It refers basically who will manage the various layers of these services. Architectural perspective cloud Services can be classified as IaaS, PaaS, SaaS & FaaS.

2. Cloud Computing

In context of IaaS - Infrastructure as a Service in Cloud Computing, cloud provider manages underneath hardware & virtualization layer which includes servers, storage & networking. End user manages virtual instance, OS, Application, Availability, Scalability of Applications built on top of Infrastructure built on top of IaaS.[2].

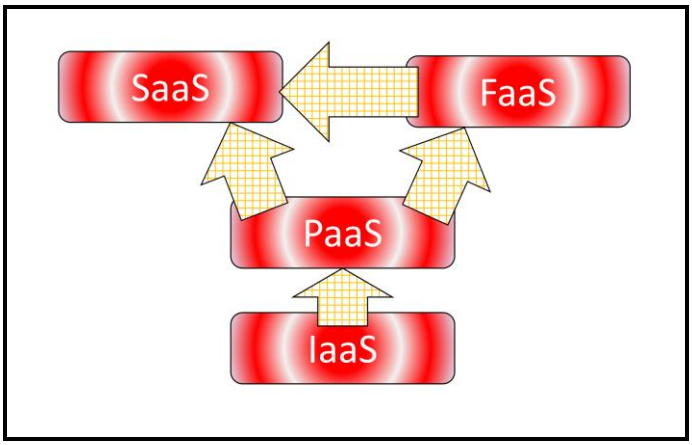

Figure 1. Layers of Cloud Services, IaaS, PaaS, SaaS & FaaS.

In context of PaaS - Platform as a Service in Cloud Computing, cloud provider provides platform which manages the virtual instances, OS, Availability &

ISSN: 2233-7857 IJFGCN

Scalability of instances build on top of IaaS. End user manages the data and applications built on top of SaaS. e.g. of SaaS is Google provided App Engine facilitates the environment for running of applications on its Cloud infrastructure, and Salesforce also provides its Force.com cloud platform. Sahara service in Open Stack cloud which creates and manages underneath Hadoop Cluster.[2]

In context of SaaS - Software as a Service in Cloud Computing, cloud provides manages all aspects of applications from Infrastructure, OS, Availability, Scalability, Platform, Applications & Data as well. SaaS are built on top of IaaS as well as FaaS, examples of SaaS are Microsoft Office 360, Oracle Cloud database offering. [2]

In the context of FaaS - Function as a Service in Cloud Computing, cloud consumer neither care about the underneath Infrastructure on which the code is running nor the availability of underneath components. In traditional approach of Cloud Computing, an instance to be spawn on the VPC with certain amount of RAM, Computing cores & Storage attached to instance. Post instance availability and configuration, instance is available to serve the intended purpose for the instance. To take care for instance security constraints cloud consumer must considering network access hardening of instance on public facing interfaces using various could network resources like security group, NACLs etc. In context of high availability consumer must take care for high availability of service by using HA resources like Load Balancers, Name services etc... As cloud user does not need to create VM / instance in cloud i.e. server for performing the intended functionality FaaS can also be referred as Serverless. Serverless is also popular term for FaaS – Function as a Service.

As a cloud FaaS user just need to write processing logic for function in available coding language supported by FaaS. Once the Function is deployed with processing logic in FaaS it is ready to serve the purpose. Lifecycle for the function is till the execution of the FaaS. It does not maintain state of the execution on its own. Unlike traditional instance in cloud, FaaS is stateless and does not maintain its state on its own. In order to maintain state FaaS can use supporting cloud services like object storage, no-sequel DB, DBs, parameter store etc. FaaS is well suited for computing context of lightweight computing, stateless / light state computing is required. This goes hand in hand with Microservice architecture and IOT platform.

FaaS – Function as a Service is available in Public Hosted Cloud such as AWS, Microsoft Azure, Google Cloud, IBM Cloud etc. and Private Hosted Cloud Implementation such as OpenStack Cloud Implementation.

3. FaaS IMPLEMENTATIONS

3.1. FaaS Implementation in Amazon AWS

In AWS FaaS is Implemented with name 'Lambda Service', This can scale up automatically when needed, it can serve from few requests per day to thousands per second. Lambda runs code on highly available compute platform.

With required IAM roles and policies AWS Lambda can access various supporting AWS services like DynamoDB, S3 buckets, trigger an event which can initiate execution of AWS service. Environment specific variables can be pass to AWS Lambda function e.g. region name. AWS Lambda provides Java Script, Python, Ruby, Java, Go, .Net Runtime as Platform for execution which AWS referred as runtime. Custom library uploading support is also available for AWS lambda deployment. AWS lambda can support memory RAM allocation from 128 MB up to 3008MB. [10] Lambda can be trigger from various

ISSN: 2233-7857 IJFGCN

AWS services as well as by Monitoring events on AWS CloudWatch also can be invoked by URL.

3.2. FaaS Implementation in Google Cloud

In Google Cloud FaaS is Implemented with name 'Cloud Function', this can scale up automatically when needed, it can serve from few requests per day to millions of requests. Google Function can access supporting services like Big DBs, DNS, File Store, Big Databases etc. with required IAM policies. Environment specific variables can be pass to Cloud Function e.g. region name. Cloud Function provides Java Script, Python, & Go Runtime as Platform.

Custom library uploading support is not available for Cloud Function deployment. Google Function can support memory RAM allocation up to 2048 MB. [11] Cloud Function can be trigger from events of HTTP, Cloud Storage, Cloud Pub/Sub i.e. call to Cloud Function manually.

3.3. FaaS Implementation in Microsoft Azure

In Microsoft Azure Cloud FaaS is implemented with name 'Azure Function', this can scale up automatically when configured. Azure Function can access supporting services like Blob Storage, Cosmos DBs, Event Grid, Event Hub, HTTP & webhook, IOT Hub, Graph, Notification, Queue, Table storage, Timer etc. with required IAM policies. Environment specific variables can be pass to Azure Function e.g. region name.

Azure Function provides C#, F#, Java, Java Script, PowerShell, Typescript, PHP, Batch and Python Runtime as Platform. Custom library uploading support is available for Azure Function deployment. Azure Function can support memory RAM allocation up to 14 GB as per plan configured. [12] Azure Function can be trigger from events of HTTP, Cloud Storage, DB storage queue events, Azure timer, HTTP and web hook i.e.

3.4. FaaS Implementation in OpenStack - Cloud

In OpenStack Cloud FaaS is implemented with various platforms like 'Apache Whisk', 'Fission', 'IronFunctions', 'Fn Project', 'OpenLambda', 'Kuberless', 'OpenFass'. In OpenStack FaaS is implemented underneath using docker and Kubernetes services. Execution language support can be added as required by creating respective docker image; similarly, RAM and core requirement can be configured as per docker image implementation for FaaS. This makes OpenStack Cloud very flexible in terms of customized microservices based architecture.

Usually using Kubernetes serverless workloads and microservices application containers are managed in OpenStack FaaS Implementation. This enables fine grain control on deployment of FaaS model in terms on amount of memory and computing power. 'Knative' is used to deploy serverless functions on Kubernetes. 'Istio' is used to horizontally scale up and scale down for the underneath container. [13]

Benefit of implementing FaaS using OpenStack technologies is FaaS can be more hardware specific by implementation using Ironic service also specific memory and processing power requirement can be configured while implementation for the microservice distributed service architecture, there are limitations in public cloud user considering these points.

4. FaaS LIFECYCLE

FaaS execution lifecycle is as depicted as in below call-flow diagram. FaaS can be invoked by internally Or External means It implies when FaaS is configured to trigger

ISSN: 2233-7857 IJFGCN

based on log or some resource/DB events it is referred as Internal FaaS Invocation and FaaS can also be invoked by means of HTTP/HTTPS call this is referred as external FaaS call. When FaaS is invoked it first the request is authenticated via authentication mechanism, only when Invoker is authorized to Invoke FaaS, FaaS is trigger.

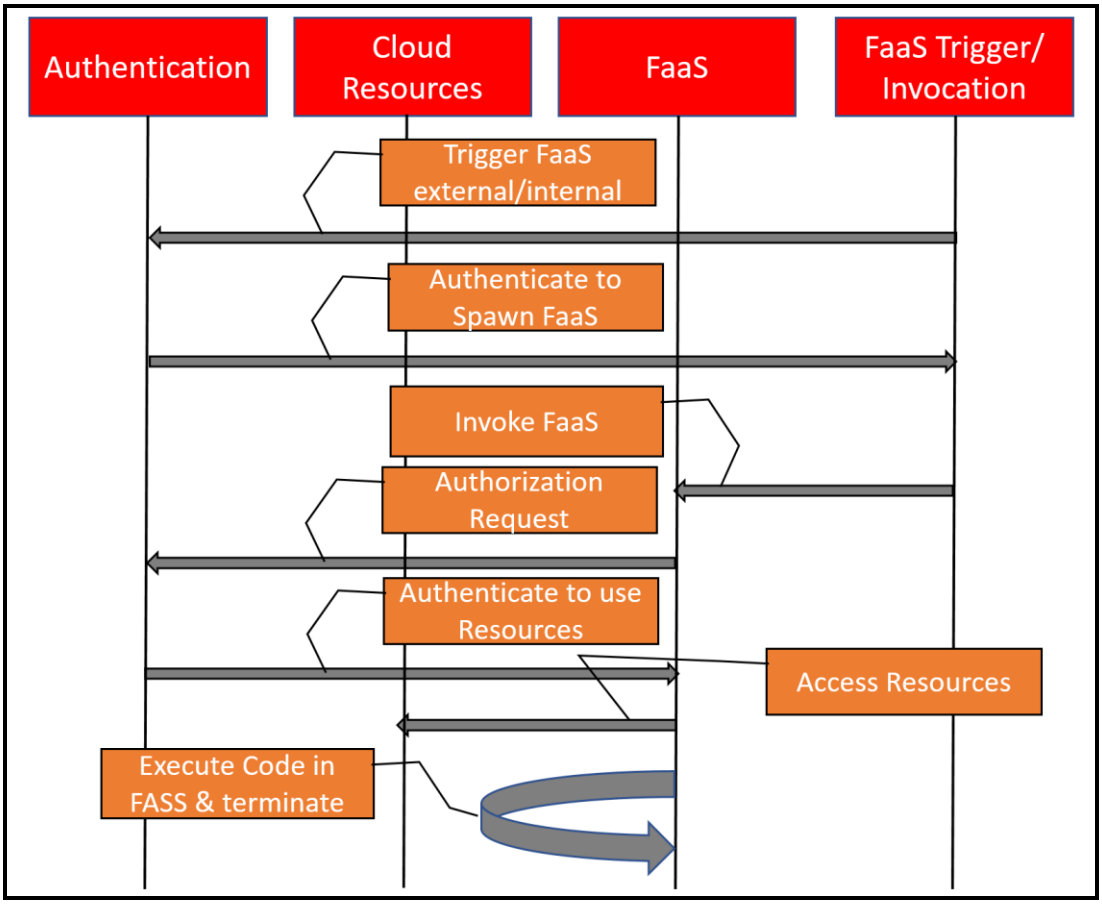

Figure 2. FaaS Function Lifecycle.

Based on FaaS logic implemented it can access other cloud resources. Before accessing cloud resources by FaaS function, request gets authenticated by authentication mechanism to access cloud resources. Post FaaS is invoked for execution with code logic provided while deploying the FaaS function. After function execution completes, function instance gets terminated. For next subsequent request/ invocation same call flow is executed. Thus, in this way no state is maintain by FaaS function across function call. State of the FaaS function can be maintained by external means by storing the state during the execution phase in consistent state resources like no-sequel DB, DB or parameter store etc. FaaS function instances can run simultaneously based on configured execution rate configured for FaaS.

5. FaaS Benefits

Considering various aspects of using FaaS is very beneficial to its end users in terms of cost effectiveness, distributed micro service design simplified architecture, securing resource access etc. these various aspects can be defined as below. [14]

ISSN: 2233-7857 IJFGCN

5.1. Cost effective

Traditional approach to provide implementation to API require Virtual Instance to be created, configured & deployed with its API implementations. Along with deployment, the deployment strategy should also have to consider scalability and availability of the instance. For public cloud this leads to much more cost even if it is microservice API implementation. Using FaaS as strategy for implementing micro distributed API is very simplified. User just pay for the time duration during which the FaaS was in execution state.

5.2. Microservice Distributed Architecture

FaaS Is well suited for microservice based Distributed Architecture, servers do not need to care about complexity of Distributed architecture. FaaS keeps the implementation overhead of application transparent, this makes it very simple to deploy application. This is well suited for Microservice Distributed architecture.

5.3. Scalability, Availability & Performance

User does not have to take care about the scalability and availability of the application. FaaS itself is highly auto scalable and available in nature. This makes Design architecture very simplified. FaaS can scales up and scale down automatically as per required load on the API, it can scale from few requests per day to thousands of requests per second. FaaS is highly available in nature, even if failure of one of the occurrences another occurrence is ready to serve the request within fraction amount of time. This makes applicable more highly scalable & available at very low cost. If multiple functions/ microservices runs on the same host application performance degrades due to disk and memory allocation which indirectly hit network performance. Implementing microservice distributed application using FaaS will significantly increase overall performance of the application as there will be disk and memory allocation from different underneath hosts of Docker / virtualization layer.

5.4. Security

As for end user FaaS function is visible only as API it isolates cloud internal resources from end users thus make is more secure as the backend servers are not visible outside the FaaS function. There is controlled way of implementation of service in terms of controlled memory and computing cores allocation.

6. FaaS Use cases

There are tremendous use cases of FaaS some of the use cases are as listed below –

6.1. Auto-scaling highly available Websites and APIs

With the help of FaaS i.e. Serverless. Websites and applications can be written and deployed without having need of backend servers for processing. Applications and APIs can be auto scale due to built-in capability of Autoscaling and highly availability of FaaS. Developer just need to Focus on processing logic and endpoint integration.

6.2. Event streaming

FaaS can be trigger from various events generated like system logs, data event, scalability events etc. without the need of complex cluster-based systems. Event

ISSN: 2233-7857 IJFGCN

streaming pipeline, queue can empower analysis system, stores also can be used for inputs to monitor systems.

6.3. Image and Video Manipulation

FaaS is well suited for basic Image and Video processing which does not require state to be retained for subsequent calls. Basic Image and Video operations can be performed such as resizing, transformation, cropping, Image to text conversion, creating thumbnails etc.

6.4. Multi-language Applications

While building Microservice based distributed application which can leverage multiple cutting-edge programming languages simultaneously for different modules, this makes application development very quick and robust.

6.5. Continuous Integration and Continuous Deployment

FaaS allows Distributed application maintenance very quick and easy. Rather than pushing monolithic update for application small modules can be upgraded rather than waiting for complete monolithic push. This makes application very robust as bugs fixes can be easily pushed in CICD environment, as soon as code gets tested next moment the code is available in production environment.

7. Conclusion

FaaS is stateless by its nature, state can be maintained across FaaS function calls by using supporting cloud resources. This well suited for Microservice based distributed application. There are various implementations of FaaS service in private and public cloud like AWS, Google Cloud, Azure, OpenStack. Faas implementation is more flexible in Private cloud implementations, as got freedom to implement the FaaS service as per requirement in terms of underneath hardware and supporting execution environment. Using FaaS in public cloud is very cost effective as user pays only for the execution time duration of FaaS. User no need to care about availability and scalability of the application. Microservice designed distributed application in FaaS is highly scalable & provides high throughput. One aspect of using FaaS is to isolate underneath cloud resources isolated from end users / other architectural components. Thus, FaaS is cutting edge technology for developing application with multiple languages and quick and robust deployment by its Auto scalable and Highly Available nature.

References

8.1. Journal Article

[1] Theo Lynn, Pierangelo Rosati, Arnaud Lejeune, Vincent Emeakaroha, "A Preliminary Review of Enterprise Serverless Cloud Computing (Function-as-a-Service) Platforms", 2017 IEEE International Conference on Cloud Computing Technology and Science (CloudCom).

[2] Shyam Patidar, Dheeraj Rane, Pritesh Jain, "A Survey Paper on Cloud Computing", 2012 Second International Conference on Advanced Computing & Communication Technologies.

[3] Erwin van Eyk, Lucian Toader, Sacheendra Talluri, Laurens Versluis, Alexandru U, Alexandru Iosup, "Serverless Is More: From PaaS to Present Cloud Computing", IEEE Internet Computing (Volume: 22, Issue: 5, Sep./Oct. 2018).

[4] Daniel Bardsley, Larry Ryan, John Howard, "Serverless Performance and Optimization Strategies", 2018 IEEE International Conference on Smart Cloud.

ISSN: 2233-7857 IJFGCN

[5] Sunil Kumar Mohanty, Gopika Premsankar, Mario Di Francesco, "An evaluation of open source serverless computing frameworks", 2018 IEEE International Conference on Cloud Computing Technology and Science (CloudCom).

[6] Jose Luis Vazquez-Poletti, Ignacio Mart ln Llorente, "Serverless Computing: From Planet Mars to the Cloud", Computing in Science & Engineering (Volume: 20, Issue: 6, Nov.-Dec. 1, 2018).

[7] Jeongchul Kim, Jungae Park, Kyungyong Lee, "Network Resource Isolation in Serverless Cloud Function Service", 2019 IEEE 4th International Workshops on Foundations and Applications of Self* Systems (FAS*W).

[8] Alfonso Perez, Sebastian Risco, Diana Marıa Naranjo, Miguel Caballer, German Molto, "On-premises Serverless Computing for Event-Driven Data Processing Applications", 2019 IEEE 12th International Conference on Cloud Computing (CLOUD).

[9] Markus Ast, Martin Gaedke, "Self-containedWeb Components through Serverless Computing", Proceedings of the 2nd International Workshop on Serverless Computing Pages 28-33.

8.2. Web reference

[10] AWS Documentation link

https://docs.aws.amazon.com/lambda/?id=docs_gateway

https://aws.amazon.com/about-aws/whats-new/2017/11/aws-lambda-doubles-maximum-memory-capacity-for-lambda-functions/

[11] Google Cloud Link

https://cloud.google.com/functions/docs/

[12] Azure Cloud Documentation link

https://docs.microsoft.com/en-us/azure/azure-functions/

[13] OpenStack Cloud Documentation link

https://opensource.com/article/18/11/open-source-serverless-platforms

[14] https://serverless.com/

ISSN: 2233-7857 IJFGCN