AWS Lambda

KARTHICK VENKATESAN[1,*,+]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: vkarthickprabu@gmail.com
[+] HID - S17-IO-3023

paper1, March 27, 2017

The rapid pace of innovation in datacenters and the software platforms within them is set to transform how we build, deploy, and manage online applications and services. Common to both hardware-based and container-based. Virtualization is the central notion of a server. Servers have long been used to back online applications, but new cloud-computing platforms foreshadow the end of the traditional backend server. Servers are notoriously difficult to configure and manage, and server startup time severely limits an application's ability to scale up and down quickly. As a result, a new model, called serverless computation, is poised to transform the construction of modern, scalable applications ability to quickly scale up and down [1]. This paper is on AWS Lambda a Serverless Computing technology. © 2017 https://creativecommons.org/licenses/. The authors verify that the text is not plagiarized.

Keywords: AWS Lambda,Serverless Computing, I524

https://github.com/karvenka/sp17-i524/tree/master/paper2/S17-IO-3023/report.pdf

1. INTRODUCTION

Serverless applications are where some amount of server-side logic are written by the application developer but unlike traditional architectures is run in stateless compute containers that are event-triggered, ephemeral, and fully managed by a 3rd party. One way to think of this is 'Functions as a service / Faas'. AWS Lambda is one of the most popular implementations of Faas [2].

AWS Lambda is a FaaS(Function as a Service) from Amazon Web Services. It runs the backend code on a high-availability compute infrastructure and performs all of the administration of the compute resources, including server and operating system maintenance and capacity provisioning [3].

In AWS Lambda one needs to pay only for the compute time consumed - there is no charge when the code is not running. AWS Lambda, can run code for virtually any type of application or backend service - all with zero administration. AWS Lambda requires only the code to be uploaded, and Lambda takes care of everything required to run and scale the code with high availability. AWS Lambda can be setup to automatically trigger the code from other AWS services or call it directly from any web or mobile app [4].

2. FEATURES

The key features of AWS Lambda are [4]

- **No Servers to Manage**: AWS Lambda automatically runs the code without requiring to provision or manage servers. Just write the code and upload it to Lambda.
- **Continuous Scaling**: AWS Lambda automatically scales the application by running code in response to each trigger. The code runs in parallel and processes each trigger individually, scaling precisely with the size of the workload.
- **Subsecond Metering**: With AWS Lambda, the institution using the service are charged for every 100ms the code executes and the number of times the code is triggered. They don't pay anything when the code isn't running.

3. ARCHITECTURE

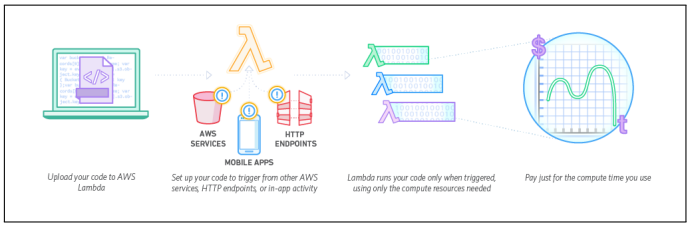

Fig. 1. AWS Lambda Architecture [4]

Lambda functions and event sources are the core components in AWS Lambda. An event source is the entity that publishes events, and a Lambda function is the custom code that processes the events. Several AWS cloud services can be preconfigured to work with AWS Lambda. The configuration is referred to as event source mapping, which maps an event source to a Lambda

function. It enables automatic invocation of Lambda function when events occur.

Each event source mapping identifies the type of events to publish and the Lambda function to invoke when events occur. The specific Lambda function then receives the event information as a parameter, and as shown in Figure 1 the Lambda function code can then process the event.

The event sources can be any of the following:

AWS services – These are the supported AWS services that can be preconfigured to work with AWS Lambda. These services can be grouped as regular AWS services or stream-based services. Amazon Kinesis Streams [5] and Amazon DynamoDB Streams [6] are stream-based event sources, all others AWS services do not use stream-based event sources.

Custom applications – Custom applications built can also publish events and invoke a Lambda function.

4. SCALABILITY

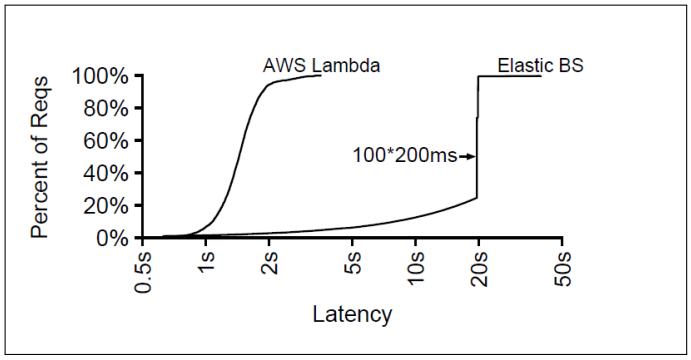

Fig. 2. Response Time. This CDF shows measured response times from a simulated load burst to an Elastic BS application and to an AWS Lambda application.
[1]

A primary advantage of the Lambda model is its ability to quickly and automatically scale the number of workers when load suddenly increases. The Graph in Figure 2 demonstrates this by comparing AWS Lambda to a container-based server platform, AWS Elastic Beanstalk [7] (hereafter Elastic BS). On both platforms, the same benchmark was run for one minute: the workload maintains 100 outstanding RPC(Remote Procedure Calls) requests and each RPC handler spins for 200ms. Figure 2 shows the result: an RPC using AWS Lambda has a median response time of only 1.6s, whereas an RPC in Elastic BS often takes 20s. Investigating the cause for this difference, it was found that while AWS Lambda was able to start 100 unique worker instances within 1.6s to serve the requests; all Elastic BS requests were served by the same instance; as a result, each request in Elastic BS had to wait behind 99 other 200ms requests. AWS Lambda also has the advantage of not requiring configuration for scaling. In contrast, Elastic BS con- figuration is complex, involving 20 different settings for scaling alone. Even though the Elastic BS was tuned to scale as fast as possible, it still failed to spin up new workers for several minutes [1].

5. DOCUMENTATION

- Detailed documentation on AWS Lambda Deployment , Configuration,Debugging and Development is available at [8].
- Use cases on reference architecture with AWS Lambda are available at [9].

6. COMPETITORS

The main competitors for AWS Lambda are

- Microsoft Azure Functions
- Google Cloud

A detailed comparison of features between AWS Lambda, Google Cloud and Microsoft Azure Functions is available in Table 1.

7. PRICING

With AWS Lambda, the users pay only for what they use. They are charged based on the number of requests for the functions and the duration the code executes.

Requests: The users are charged for the total number of requests across all their functions. Lambda counts a request each time it starts executing in response to an event notification or invokes call, including test invokes from the console. First 1 million requests per month are free $0.20 per 1 million requests thereafter ($0.0000002 per request)

Duration:Duration is calculated from the time the code begins executing until it returns or otherwise terminates, rounded up to the nearest 100ms. The price depends on the amount of memory the user allocates to the function. The users are charged $0.00001667 for every GB-second used [11].

8. USE CASE

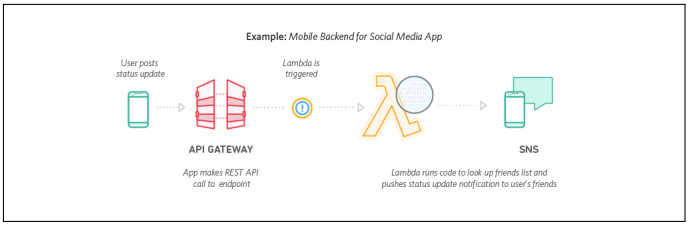

Fig. 3. Mobile Backend
[4]

Mobile Backends: Mobile Backends are an excellent use case to implement AWS Lambda.As shown in Figure 3 the back-end of the mobile application can be built using AWS Lambda and Amazon API Gateway to authenticate and process API requests. Lambda makes it easy to create rich, personalised app experiences. **Bustle.com** is a news, entertainment, lifestyle, and fashion website catering to women. Bustle also operates **Romper.com**, a website focused on motherhood.Bustle is based in Brooklyn, NY and is read by 50 million people each month. Bustle uses AWS Lambda to process high volumes of site metric data from Amazon Kinesis Streams [5] in real time. This allows the Bustle team to get data more quickly so they can understand how new site features affect usage. They can also measure user engagement, allowing better data-driven decisions. The serverless back end supports the Romper website and iOS app as well as the Bustle iOS app. With AWS Lambda, Bustle was able to eliminate the need to worry about operations. The engineering team at Bustle focus on just writing code, deploy it, and it

Table 1. AWS Lambda vs. Google Cloud Functions vs. Microsoft Azure Functions [10]

FEATURE	AWS LAMBDA	GOOGLE CLOUD	AZURE FUNCTIONS
Scalability & availability	Automatic scaling (transparently)	Automatic scaling	Metered scaling (App Service Plan) Automatic scaling (Consumption Plan)
Max # of functions	Unlimited functions	20 functions per project (alpha)	Unlimited functions
Concurrent executions	100 parallel executions (soft limit)	No limit	No limit
Max execution	300 sec (5 min)	No limit	300 sec (5 min)
Supported languages	JavaScript, Java Python, C#	Only JavaScript	C# and JavaScript (preview of F#, Python, Batch, PHP, PowerShell)
Dependencies	Deployment Packages	npm package.json	Npm, NuGet
Deployments	Only ZIP upload (to Lambda or S3)	ZIP upload, Cloud Storage or Cloud Source Repositories	Visual Studio Team Services, OneDrive, GitHub, Bitbucket, Dropbox
Environment variables	Yes	Not yet	App Settings and ConnectionStrings from App Services
Versioning	Versions and aliases	Cloud Source branch/tag	Cloud Source branch/tag
Event-driven	S3, SNS, SES, DynamoDB, Kinesis, CloudWatch, Cognito, API Gateway	Cloud Pub/Sub or Cloud Storage Object Change Notifications	Blob, EventHub, Generic WebHook Queue, Http, Timer triggers
HTTP(S) invocation	API Gateway	HTTP trigger	HTTP trigger
Orchestration	AWS Step Functions	Not yet	Azure Logic Apps
Logs management	CloudWatch	Cloud Logging	App Services monitoring
In-browser code editor	Yes	Only with Cloud Source Repositories	Functions environment, AppServices editor
Granular IAM	IAM roles	Not yet	IAM roles
Pricing	1M requests for free (Free Tier), then $0.20/1M requests	Unknown until open beta	1 million requests and 400,000 GB-s

scales infinitely, and they don't have to deal with infrastructure management. Bustle was able to achieve the same level of operational scale with half the size of team of what is normally needed to build and operate the site [12].

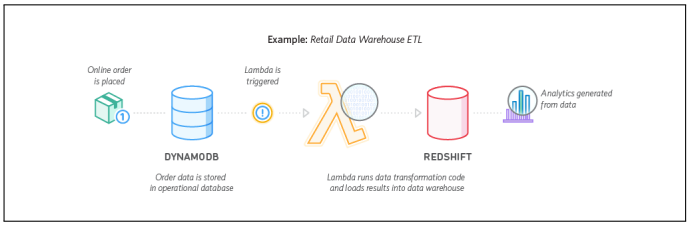

Fig. 4

[4]

Extract, Transform, Load: As shown in Figure 4 AWS Lambda can be used to perform data validation, filtering, sorting, or other transformations for every data change in a DynamoDB table and load the transformed data to another data store. **Zillow** is the leading real estate and rental marketplace dedicated to empowering consumers with data, inspiration and knowledge around the place they call home, and connecting them with the best local professionals who can help. Zillow needed to collect a subset of mobile app metrics in realtime and report it to the buisness users several times during the day.The solution was to be delivered in 3 weeks.Leveraging AWS Lambda and Amazon Kinesis Zillow was able to seamlessly scale 56 lines of code to over 16 million posts a day and achieve its goal in 2 weeks [13].

9. CONCLUSION

Serverless Computing allows to build and run applications and services without thinking about servers. At the core of serverless computing is AWS Lambda, which lets to build auto-scaling, pay-per-execution, event-driven apps quickly.

ACKNOWLEDGEMENTS

The authors thank Prof. Gregor von Laszewski for his technical guidance.

REFERENCES

[1] S. Hendrickson, S. Sturdevant, T. Harter, V. Venkataramani, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau, "Serverless computation with openlambda," in *8th USENIX Workshop on Hot Topics in Cloud Computing, HotCloud 2016, Denver, CO, USA, June 20-21, 2016*. Berkeley, CA, USA: USENIX Association, 2016. [Online]. Available: https://www.usenix.org/conference/hotcloud16/workshop-program/presentation/hendrickson

[2] M. Roberts, “Serverless,” Web Page, Aug. 2016, accessed 2017-03-26. [Online]. Available: https://martinfowler.com/articles/serverless.html
[3] A. Gupta, “Serverless faas with aws lambda and java,” Web Page, Jan. 2017, accessed 2017-03-26. [Online]. Available: https://blog.couchbase.com/serverless-faas-aws-lambda-java/
[4] Amazon Web Services, Inc, “AWS Lambda,” Web Page, accessed 2017-03-26. [Online]. Available: https://aws.amazon.com/lambda/
[5] ——, “AWS Kinesis Streams,” Web Page, accessed 2017-03-26. [Online]. Available: https://aws.amazon.com/kinesis/streams/
[6] ——, “AWS Dynamo DB Streams,” Web Page, accessed 2017-03-26. [Online]. Available: http://docs.aws.amazon.com/amazondynamodb/latest/APIReference/API_Types_Amazon_DynamoDB_Streams.html
[7] ——, “AWS Elastic Beanstalk,” Web Page, accessed 2017-03-26. [Online]. Available: https://aws.amazon.com/elasticbeanstalk/
[8] ——, “AWS Lambda Doc,” Web Page, accessed 2017-03-26. [Online]. Available: https://aws.amazon.com/documentation/lambda/
[9] ——, “AWS Lambda Use Case,” Web Page, accessed 2017-03-26. [Online]. Available: http://docs.aws.amazon.com/lambda/latest/dg/use-cases.html
[10] M. Parenzan, “Microsoft azure functions vs. google cloud functions vs. aws lambda,” Web Page, Feb. 2017, accessed 2017-03-26. [Online]. Available: http://cloudacademy.com/blog/microsoft-azure-functions-vs-google-cloud-functions-fight-for-serverless-cloud-domination-continues/
[11] Amazon Web Services, Inc, “AWS Lambda Pricing,” Web Page, accessed 2017-03-26. [Online]. Available: https://aws.amazon.com/lambda/pricing/
[12] ——, “Bustle Case Study,” Web Page, accessed 2017-03-26. [Online]. Available: https://aws.amazon.com/solutions/case-studies/bustle/
[13] ——, “AWS re:Invent 2015 | (BDT307) Zero Infrastructure, Real-Time Data Collection, and Analytics,” https://www.youtube.com/watch?v=ygHGPnAd0Uo, Youtube, Oct. 2015, accessed 2017-03-26.