Literature Review – Lab 5

Currently the uses and possibilities of cloud computing continue to evolve, scaling from low-level services in Amazon Web Services (AWS) towards specialized high-level services. One of the most evolving cloud computing topics is serverless computing. Currently, the field of serverless computing is growing, offering users the ability to run a fully managed high-level service without having operational concerns like managing or scaling server infrastructure [1]. Serverless is also referred to as Function-as-a Service (FaaS) and plays a role in connecting the various services with each other using functions. When developers use FaaS services like AWS Lambda, they provide atomic and short-running code for their functions in which the FaaS providers run the code on-demand [2]. By using FaaS, developers benefit from simple deployments, reduced operations efforts, and efficient, lower cost pricing compared to paying upfront to use a service for an entire year. Not only does FaaS benefit the developers, but it also gives providers the opportunity to provide services to a large number of users with relatively few resources needed [2]. The benefits of FaaS and serverless computing can be seen in document [3] with the example of a "website that collapsed when it was unable to handle comments about net neutrality." This example highlights the problem with the current web application and would gain benefits if a serverless platform was used instead. A serverless platform would fare a better chance at handling the scale of traffic generated from the comments. This is because it is much easier to make mistakes with the current platform if expertise is not available [3]. The expertise of deciding how many servers to deploy and then maintain their scaling is difficult compared to serverless where the operational/maintenance tasks and scalability of the applications are handled by the cloud or FaaS provider [1]. Serverless also offers supporting middleware and artificial intelligence services that integrate seamlessly with the platform to enable a variety of functions and services [3].

In terms of weaknesses of the serverless platform, FaaS based applications are forced to adhere to multiple technical and architectural restrictions to allow transparent management. Due to limitations, FaaS services are only able to host small microservices implemented using short code, making it difficult to develop larger applications from these stateless microservices [2]. In terms of the actual infrastructure itself, there are limitations for the user if the application requires adjustments to the execution environment. Because the infrastructure is only accessible by the cloud/FaaS provider, the user does not have direct access to the virtual machines, containers, operating system, and file systems used to execute the functions for the application. Therefore, there will be limitations if there needs to be adjustments in the actual architecture [3]. As indicated by document [1], there has been previous research that found "performance-related challenges common to many FaaS platforms." Several of these causes due to cold start times that delay execution of functions for multiple seconds, "hardware heterogeneity" that makes estimating the execution time of the function difficult, and complex triggering mechanisms that can potentially lead to delays in function execution [1]. Although the beneficial aspects of serverless computing is known, it is important to have a greater understanding over the limitations and weaknesses the platform can hold.

For the specific serverless model created for Lab5, I believe one possible way for the model to be improved is by creating only one lambda function. This lambda function will perform the functions of both the kinesis-dynamo function and sentiment-analysis function. By having only one function, there will be less time spent on having the sentiment-analysis function read and update the tweet in the table after calculating the sentiment value. Also, from my calculations for the cost analysis, I calculated that the lambda functions cost the most per month because there are two functions that potentially perform over a million requests per month. By reducing the architecture as so, the price per month will decrease significantly with only one lambda function and a reduced number of requests.

Works Cited

[1] J. Scheuner and P. Leitner, "Function-as-a-Service performance evaluation: A multivocal literature review," *Journal of Systems and Software,* vol. 170, 2020.

[2] P. Leitner, E. Wittern, J. Spillner and W. Hummer, "A mixed-method empirical study of Function-as-a-Service software development in industrial practice," *Journal of Systems and Software,* vol. 149, 2019.

[3] G. Fox, V. Isahagian, V. Muthusamy and A. Slominski, "Status of Serverless Computing and Function-as-a-Service(FaaS) in Industry and Research," in *First International Workshop on Serverless Computing (WoSC)*, 2017.