International Journal of Scientific Research in Computer Science, Engineering and Information Technology
ISSN : 2456-3307 OPEN ACCESS
Available Online at : www.ijsrcseit.com
doi : https://doi.org/10.32628/IJSRCSEIT

Serverless Computing for Application Engineers : Efficiency and Cost Benefits

Murali Kadiyala

Independent Researcher, USA

Article Info

Publication Issue :
Volume 7, Issue 4
July-August-2021

Page Number : 702-709

Article History
Accepted: 01 Aug 2021
Published: 12 Aug 2021

ABSTRACT

The supervision of serverless computing has become a paradigm shift in application engineering as it provides an economic, sustainable, and efficient way of solving problems for the developers. Unlike localhost-based models that call for users to organize servers, the serverless solutions introduced offer platforms where the only coding required is that of the application logic. This is because using a pay-as-you-go model of resource allocation ensures that organizations or departments only pay as that go costs to be incurred. It is an approach that improves the productivity of the development process, supports innovation, and also decreases the costs of server maintenance. But, there are certain issues that are yet to be resolved: cold-start latencies, debugging hard-to-solve issues, and vendor lock-ins. Nevertheless, the continuous development of serverless technologies indicates increased performance and usage for today's software in clouds.

Keywords : Serverless Computing, Cost Efficiency, Scalability, Cloud Applications, Cold-Start Latency

INTRODUCTION

Comparative to conventional solutions where one's own server is rented or gathered, serverless computing has become a revolutionary model of application engineering in the modern IT environment that yields high availability and cost effectiveness. Serverless computing provides the same benefits of abstraction without needing the software developers to manage or provision the hardware extensively; rather, that are only required to write code that the service demands, and then the resources will be provided by the computing service. This model works on a cost structure that is based on consumptions that only pays for the amount of CPU time utilized to serve clients and not owning huge and non-utilized servers. For application engineers, it is easier to manage the functioning with less overhead in operations, quick implementation, and balanced workloads. Server-less also allows for more innovation and faster prototyping, the developers and engineers do not need to bother themselves with the infrastructural support of functions or services. Nonetheless, while the advantages are potential to a high degree, there are also cons, including cold startup time, debugging difficulties, and vendor dependence, all of which should be considered for particular applications.

Literature review

Performance Evaluation and Challenges in Serverless Computing Platforms

According to the authors McGrath and Brenner, 2017, serverless computing platforms are now considered as a popular realisation of the concept of scale-out applications and provide a number of advantages in terms of operational overhead and costs. Nevertheless, one of the main concerns of using these environments is the optimization of performance, especially in areas such as function scaling, the lifecycle of containers and the overall overhead when it comes to execution (McGrath and Brenner, 2017). However, current solutions such as AWS Lambda, Azure Functions, Google Cloud Functions, and Apache OpenWhisk also have some problems in the aspects of the execution speed, scalability and resource utilization.

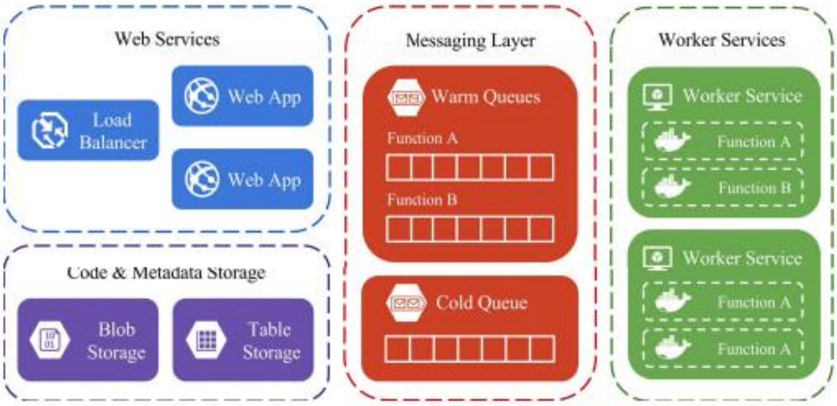

Figure 1: Overview of prototype components, showing the organization of the web and worker services

(Source: McGrath and Brenner, 2017)

Several works have been done in finding low latency-high bandwidth with various combinations of figuring out the new ways of workloads and how they can affect the performance of the system in compromising the relative trade-offs between performance and flexibility. Some strategies provide higher throughput under various low to moderate concurrent connections, while they offer poor scalability in higher concurrent connections. Recently, the process of discovering and reusing the Overhead of Containers in serverless environments has become a keen concern due to the cost-proportional and efficient behaviors of such platforms. There's more future work involved in eliminating these issues and achieving the best results in the specified field on the serverless architecture.

Evaluating Function-as-a-Service (FaaS) for Demanding Computational Applications

According to the authors Spillner et al. 2018, it is noteworthy that cloud computing adaptation remains more or less different across these various application types; the most commonly adapted application models of cloud infrastructure and services include scalable web application, mobile back-end, and on-demand databases.

Currently, Para has targeted traditional applications, which normally need HPC or scientific computing services, to provide necessary special types of virtual machines including compute or memory-optimized virtual machine instances. It is a common practice that these requirements entail monolithic architecture that is equipped for specific compute intense functions. Given new tendencies of Function-as-a-Service providing function-level compute instances generic configurations turned out to be ineffective for these intensive applications again.

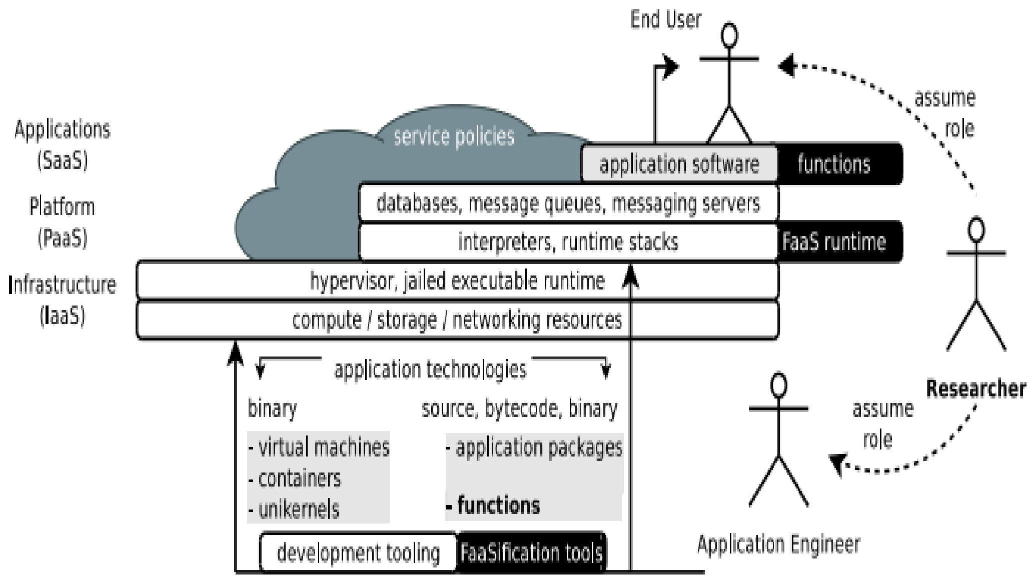

Figure 2: Positioning of FaaS in cloud application development

(Source: Spillner *et al.* 2018)

It has been analyzed that FaaS approaches are versatile in different size scaling web and backend jobs, but these scenarios are not appropriate for HPC or high-throughput computing problems because of the cold-start latency, and handling multiple distributed functions (Spillner *et al.* 2018). Research has been made to analyze the FaaS model with the traditional one that is the monolithic execution and researchers have found out that there is a high difference in scalability, resource utilization and time. Also, it is further examined how the nature of the legacy software can be refactored and transformed for FaaS architecture, which opens the way to enhance processing in the case of the computationally heavy tasks.

The Rise of Serverless Computing and Function as a Service (FaaS) in Cloud Architectures

According to the authors Sewak and Singh, 2018, computing without servers and Function as a Service became inherent aspects of the modern cloud infrastructures especially if an organization wants to improve the efficiency of its functioning, as well as optimize its costs. They enable the developers to concern themselves only with writing down the business logics without worrying about many infrastructural and runtime configurations and deployment. FaaS as a specific type of serverless computing allows developers to write small-scale functions linked to particular triggers and provides only the necessary infrastructure and scale (Sewak and

Singh, 2018). As such, there is no need for physical allocation of server resources, where the infrastructure can be provisioned on the fly depending on the requirements of the functions being executed.

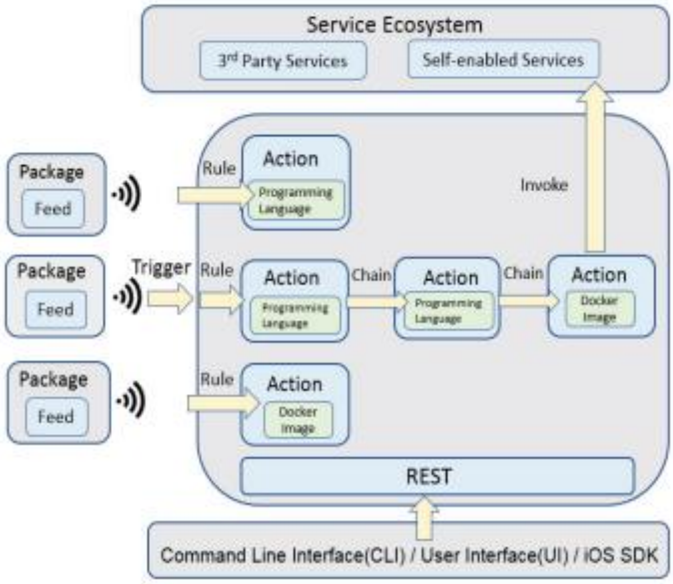

Figure 3: Serverless Architecture

(Source: Sewak and Singh, 2018)

Due to the fact that it is cost effective based on the actual usage of the resources of the system in an organization context, this method implies noteworthy economies of scale, particularly for organizations that have fluctuating workloads. AWS Lambda, Microsoft Azure Functions, and Google Cloud Functions provide FaaS with some distinct characteristics and the possibility to manage microservices that allow businesses to deploy scalable applications. There are still the topics like, cold start latency, the difficulties associated with debugging an application running on a FaaS environment, and possibility of being locked into a particular platform which still persist as issues to consider when deploying FaaS in enterprise applications.

Methods

Data Collection and Preprocessing

In the analysis it is pertinent to start by collecting the necessary data. In this project, the information is gathered from internal applications, other public data, and APIs. The data includes both descriptive data such as the user's age, gender etc and transactional data such as frequency of purchases. In this step, data is prepared to make sure that it is in a good condition to be a DW/BI solution utilized for analysis (Lynn *et al.* 2017). This ranges from dealing with missing values in the dataset, outliers, as well as inconsistency. It deals with the missing values using the KNN imputation technique, as well as normalizing the numerical features. Preprocessing is very essential since it helps to feed the model with right data which has less noise and is well formatted.

Feature Engineering and Transformation

Feature engineering means showing the creation of new features or modifying some of them which will enhance the capability of the model. The technical features used in this project are identified descriptively and in association with relevant domains. It also applies certain procedures to manipulate the features, namely scaling the features, transforming categorical variables into numerical and combining one or more similar features (Lloyd *et al.* 2018). In the remaining categorical features it uses one of the two transformation methods; a one-hot encoding or label encoding based on the nature of the feature. For instance, let a variable be ages; then age group is one of the attributes represented numerically with numbers. Such engineered features facilitate enhancement of the model's capacity to analyze and learn concerning the data information.

Model Training and Evaluation

After the data preparation, and before performing feature engineering, the next activity is model training. In taking the case, various models are modeled including logistic regression, the decision trees, and ensemble methods based on the type of the problem in the case. In other words, for each model, the training data is used and then cross validation is also performed. Evaluating based on accuracy, precision, recall, as well as AUC-ROC curve in order to determine the performance of the model (Van Eyk *et al.* 2018). The Ray Tune is used to tune up hyperparameters and avoid overfitting so that the model is good when tested with unseen data. It is noteworthy that the formation of the final model involves the choice in favor of a model that was defined in terms of its performance metrics.

Result

Cost Efficiency in Serverless Computing

Another major benefit of Serverless Computing is that it is relatively cheaper. Standard server operation requires purchasing constant data accessibility, even though most of the time resources remain idle, particularly during days of low usage. On the other hand, serverless computing involves cost structures that are based on the Formation of Resource.request where costs are made solely based on the amount of time that functions take to execute (Alqaryoutia and Siyamb, 2018). This scheme is most useful when the stream of traffic is unknown or irregular for it will enable one to manage resources well without incurring unnecessary costs. This way applications can be easily scaled in organizations while the cost of having idle servers is greatly reduced thus needed costs are cut.

Enhanced Flexibility and Speed of Development

Adult learners have a way of using serverless computing as it allows the development and deployment of applications with minimal intervention of the underlying infrastructure. This makes implementation of engineers easier in that it does not require dealing with the servers or the operating system (Pérez *et al.* 2018). One of the biggest advantages of serverless platforms is nimble, which enables fast prototyping in almost any development nowadays. It is also significant to note that with the help of serverless, development teams can easily adapt fast and unhesitatingly innovate new features and manage the current customer demands without dealing with infrastructure barriers.

Challenges and Limitations

Nevertheless, serverless computing has some disadvantages. Cold-start latency in which some functions take a certain duration to start before executing may hinder their performance particularly in the systems that have tight response time demands. Moreover, debugging and testing of the serverless applications are difficult compared to the traditional models as there is no easy access to the underlying context (Lloyd *et al.* 2018). A major disadvantage is that it leads to the vendor lock, as the heavy use of certain providers may cause problems to switch to other servers. Each of these deserve to be assessed depending on the exact use case and the demand that will be put to the application to see if it would be fit for serverless.

Discussion

Serverless computing has emerged as one of the popular trends in the recent past as it has several advantages such as overhead cost savings. This model eliminates the pressure from developers to deal directly with the servers to do with applications that are directly involving the application code aspect that enhances the delegation of applications, flow of expertise within a short span of time besides enhancing scalability factor (Kumanov *et al.* 2018). New alternatives such as AWS Lambda, AZURE Functions, and Google Cloud Functions allow developers to deploy microservices that are scalable, which means that microservices are only fueled at supratidal, consequently, cutting down the costs of application development. However, it is crucial that every positives come with a downside and following are some of the cons of serverless computing. First call latency, where the initial request takes some time to load, poses a problem to the systems that have critical response time. It is also self-explanatory that debugging is also a more complicated step due to the fact that the actual underlying environment for the program is abstracted over. One of the main issues that can be worth mentioning is the problem of vendor lock-in which hinders options to switch to other providers once the platform has been chosen. These issues, nevertheless, are avoidable and thus it is important to consider these factors so as to meet the needs of the application (Lee *et al.* 2018). Moreover, since serverless technology is still in its relatively early stage, it could be believed that these challenges, or their solutions, will become better with time. The article also shows that application engineers can derive a lot of advantages in terms of costs and time to deliver when using serverless computing, but can also suffer from certain disadvantages such as issues with performance and operations if proper care is not taken. Despite numerous advantages associated with the integration of serverless solutions, it is essential to comprehend the shortcomings involved in an effort to select the most convenient application area for using this approach.

Future Directions

Serverless computing of the future has made significant progress to become the norm in technological sectors for application deployment, and progress is being made to work out the issues like cold-start latency, debugging, and vendor lock-in. Furthermore, new solutions in the serverless context look to be targeted at achieving higher service performance, shortening the start-up time, and managing computing resources more effectively. The development of next generation, function-based, serverless architectures might provide another level of containerization and integration to facilitate ability of such functions to scale as needed and as quickly as needed while still maintaining efficiency and flexibility (Dziurzanski *et al.* 2018). Furthermore, the growing transitional cloud model, servers can be combined with traditional cloud environments for better support of various workloads. Some improvements to the capacity to support stateful applications in serverless could also be liberating, as most serverless platforms are currently designed to handle stateless applications. In addition, correspondingly and because the serverless is being actively introduced in microservices, tools for its monitoring, logging, and debugging should increase in functionality, thus making the serverless model more popular and easy to implement. The problem of vendor lock-in may seem less of an issue as service interoperability between the cloud providers enhances with better abstractions between vendor offerings. In the end, as the concept advances over time, it will be possible to attain greater advantages in terms of scalability, cost saving and ease of use, and therefore qualify for use by a still wider range of enterprises.

Conclusion

Serverless computing equally benefits the application engineers due to flexibility, cost, and time when it comes to application development. That would be able to write code and deliver value without having to worry about matters concerning infrastructure such as allocation of servers and other related issues. This is because the pay-as-you-go model has advanced features that also imply that the resources should be conserved hence cutting down on the expenses and making it more versatile. There are issues like Cold start latency, which refers to time required to load the first picture after the start of the DL, Debugging troubles, and Vendor lock-in, which might become crucial. However, for now, maximum benefit can still be realised from the current form of serverless computing and the future seems even brighter as more developments are being made to improve on the current shortcoming of the technology. Over time, as the new generation cloud-native applications develop, the serverless platforms will become even more important in enhancing workloads and pace of innovation. In conclusion, serverless computing is not for everyone, but, fashioned properly, it is a major asset to software development in the modern world.

REFERENCES

[1]. McGrath, G. and Brenner, P.R., 2017, June. Serverless computing: Design, implementation, and performance. In 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW) (pp. 405-410). IEEE.

[2]. Spillner, J., Mateos, C. and Monge, D.A., 2018. Faaster, better, cheaper: The prospect of serverless scientific computing and hpc. In High Performance Computing: 4th Latin American Conference, CARLA 2017, Buenos Aires, Argentina, and Colonia del Sacramento, Uruguay, September 20-22, 2017, Revised Selected Papers 4 (pp. 154-168). Springer International Publishing.

[3]. Sewak, M. and Singh, S., 2018, April. Winning in the era of serverless computing and function as a service. In 2018 3rd International Conference for Convergence in Technology (I2CT) (pp. 1-5). IEEE.

[4]. Lynn, T., Rosati, P., Lejeune, A. and Emeakaroha, V., 2017, December. A preliminary review of enterprise serverless cloud computing (function-as-a-service) platforms. In 2017 IEEE International Conference on Cloud Computing Technology and Science (CloudCom) (pp. 162-169). IEEE.

[5]. Lloyd, W., Ramesh, S., Chinthalapati, S., Ly, L. and Pallickara, S., 2018, April. Serverless computing: An investigation of factors influencing microservice performance. In 2018 IEEE international conference on cloud engineering (IC2E) (pp. 159-169). IEEE.

[6]. Van Eyk, E., Toader, L., Talluri, S., Versluis, L., Uță, A. and Iosup, A., 2018. Serverless is more: From paas to present cloud computing. IEEE Internet Computing, 22(5), pp.8-17.

[7]. Alqaryoutia, O. and Siyamb, N., 2018. Serverless Computing and Scheduling Tasks on Cloud: A. American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS), 40(1), pp.235-247.

[8]. Pérez, A., Moltó, G., Caballer, M. and Calatrava, A., 2018. Serverless computing for container-based architectures. Future Generation Computer Systems, 83, pp.50-59.

[9]. Lloyd, W., Vu, M., Zhang, B., David, O. and Leavesley, G., 2018, December. Improving application migration to serverless computing platforms: Latency mitigation with keep-alive workloads. In 2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion) (pp. 195-200). IEEE.

[10]. Kumanov, D., Hung, L.H., Lloyd, W. and Yeung, K.Y., 2018. Serverless computing provides on-demand high performance computing for biomedical research. arXiv preprint arXiv:1807.11659.

[11]. Lee, H., Satyam, K. and Fox, G., 2018, July. Evaluation of production serverless computing environments. In 2018 IEEE 11th International Conference on Cloud Computing (CLOUD) (pp. 442-450). IEEE.

[12]. Dziurzanski, P., Swan, J. and Indrusiak, L.S., 2018, July. Value-based manufacturing optimisation in serverless clouds for industry 4.0. In Proceedings of the Genetic and Evolutionary Computation Conference (pp. 1222-1229).