Serverless Computing for Software Development

Haibi Peng 875552

1. **Introduction**

Nowadays, everyone is using all kinds of applications, which makes our life easier, excluding the software developers. However, there is one technology that makes life easier for them, which is serverless computing.

Serverless computing refers to the concept of building and running applications that do not require server management [1]. It has been gaining importance in recent years after the launch of Amazon's Lambda platform in 2014 [2]. It is increasingly favored by developers and enterprises, since it has a huge impact in decreasing latency, improving scalability, reducing costs, and eliminating server-side management during the software development lifecycle [3].

Additionally, serverless computing includes a great variety of technologies, which can be generally categorized as Backend-as-a-Service (BaaS) and Functions-as-a-Service (FaaS) [4]. According to previous research and practices, serverless platforms allow developers to focus more on business logic, instead of acquiring, operating, and maintaining computing resources when building software [2, 5, 6].

On the one hand, serverless computing has brought many benefits to software development and deployment. On the other hand, it also introduces several new issues, such as testing, performance, and security. Before adopting serverless computing in actual software development, it is essential for the developers and researchers to better understand the motivations and benefits, as well as the challenges of using serverless computing in different situations.

This review report will discuss four aspects of serverless computing, which include its features and benefits. In addition, it will inspect the challenges of serverless computing and its prospects, thus helping the developers and researchers to achieve better results in their works.

2. **Definition and Features**

This section will first introduce the related concepts and features of serverless computing, and then give four typical applications of serverless computing. Finally, a few well-known platforms will be introduced.

2.1. Backend as a Service (BaaS) and Function as a Service (FaaS)

Two of the most prominent implementations of serverless computing are Backend as a Service (BaaS) and Function as a Service (FaaS). Normally, they represent different functionalities but work together to realize a serverless computing system. The first one, BaaS, includes services such as object storage, databases, messaging, and user management. It allows the developers to substitute server-side services with these off-the-shelf services, which enables them to outsource these components behind the application, and they can focus more on the application logic in the frontend [5]. The second one, FaaS, can be regarded as an isolated environment that allows small pieces of code to act as functions and run for a limited and short time [7]. The functions are also known as "event-driven functions" [5], since they must be triggered by external events, such as client-side requests and data streams [4].

2.2. Features of serverless computing

Four features are needed to characterize serverless computing [3]. First, the execution environment should be invisible to the customers. The customers cannot access the execution environment, including the virtual machine, the processing node, the container, and the operating system. Second, services provided by serverless computing should be auto-scaling, which means the computing resources are available on instant demand. Third, the billing model should be in line with the number of resources that have been used by the customers, which is also known as the pay-as-you-go billing mechanism. Finally, functions are the fundamental elements executed in serverless computing, which are not transparent to the providers. A service can only be called a serverless service when it equips all of these four features.

2.3. Applications of serverless computing

Serverless computing has many applications, and this report summarizes four of the most typical ones. First, the stateless property allows serverless computing to become an attractive tool for real-time collaboration applications, such as instant messaging and chatbots. Authors in [9] introduce a serverless method for instant messaging based on XMPP. Second, Machine learning is one of the most heated research areas, especially those approaches based on neural networks. A model is discussed in [10], which uses serverless services to deploy a pay-per-request neural network. Third, serverless computing can offer security solutions for all kinds of infrastructures and systems. It can also be used to conduct intrusion detection. A serverless engine, StreamAlert [11], is built upon Amazon Lambda, which can detect intrusion in real-time. Finally, serverless computing has been exploited and utilized in various Internet of Things (IoT) applications. A serverless fog computing approach, which enables ubiquitous computing, is proposed by Cheng et al. [12]. It is aimed to support data-centric IoT services.

In the future, with the rapid development of 5G, blockchain and container technologies, there will be increasingly more kinds of serverless computing applications in our daily life.

2.4. Related serverless computing platforms

Since AWS Lambda, created by Amazon in 2014 [1], has emerged as the first serverless platform, cloud computing has also walked into a new era. This computing model is also referred to as serverless computing [2]. Other competitive providers have followed the trend by 2016, including Google, Microsoft, Alibaba, and IBM. A variety of commercial platforms offer serverless computing systems. Some of the widely used systems include Google Cloud Functions, AWS Lambda, Azure Functions, and Function Compute, whereas a few opensource platforms are available, such as Apache OpenWhisk and IBM Cloud Functions.

3. Benefits and challenges of serverless computing

This section will introduce the benefits and challenges that serverless computing brings to software development.

3.1. Benefits

Serverless computing provides four main benefits to software developers. They are summarized as follows.

First, serverless computing is cost-effective. Applications using serverless computing are abstracted from server-side infrastructures, hence services are provided in a cost-based model, depending on the usage [13]. Applications will start to run whenever a client sends a request to the services from those applications. The client will only be charged by the cloud providers for the actual used space, which means it will cost nothing while the applications are idle.

Second, serverless computing has high scalability, which has mitigated the issue of allocating resources in a reasonable way [14]. Consequently, the developers do not need to concern about the scalability of the application, since it will automatically scale itself whenever the requests sent to the application are increased or decreased [1].

Third, there is minimal server-side management for serverless computing. By using serverless computing, the developers no longer need to concern about server-side logic and its management. Serverless computing providers will be responsible for managing and maintaining both the hardware and software, which are necessary to deploy applications.

Fourth, it is quite easy to deploy and update serverless applications. Developers are only required to upload some functions and then release a new version to deploy an application, since serverless computing will take care of managing application deployment and infrastructure-related problems, including server provisioning and scaling [2].

3.2. Challenges

This subsection will introduce four challenges brought by utilizing serverless computing. They are summarized as follows.

First, it is hard to test and debug serverless functions. On the one hand, it is not always feasible to replicate serverless environments, since the serverless functions need to be tested in the production environment. On the other hand, the distributed property of serverless computing makes it even more complex when it comes to debugging since the server-side processes are invisible to the developers.

Second, the performance of serverless functions can be a problem. Since the pay-as-you-go and event-driven characteristics, serverless functions are not always active but need to be activated from time to time. Consequently, this kind of "cold start" [8] requires some time to execute and might bring delays when executing the applications.

Third, serverless computing is not suitable for long-running functions. Serverless functions are executed in a short and limited period. Some functions may need a long execution time, such as machine learning tasks. However, serverless computing does not support such long execution because of its stateless nature, which indicates that the function cannot be resumed and must be restarted if paused.

Fourth, security is one of the most prominent challenges in adopting serverless computing. One of the issues is about the isolation since serverless functions are executed on a platform that is shared by many users, thus requiring a high level of isolation. Another security issue is the trust regarding process-sensitive data.

4. Prospects of serverless computing

Currently, serverless computing development is quite new. Therefore, it is worthwhile focusing on three potential research directions to solve the challenges. First, one major research path is to overcome the cold start issue while causing no negative effect on the prominent benefits of serverless computing. This could be realized by improving scheduling mechanisms and implementing more accurate measurements of function performance [15]. Second, the available tools are still quite immature for testing, debugging, and deploying serverless applications, which discourages many software developers to adopt serverless computing. Therefore, it is necessary to develop and assure the developers of adequate debugging and testing tools. Third, currently available tools are not practical enough to automatically migrate legacy code into FaaS, since some works must be done manually. Therefore, one interesting research topic is to find optimal automatic migration solutions for current legacy systems. Meanwhile, another important improvement is to develop tools for checking if a legacy system could migrate into serverless computing.

5. Conclusions

Indisputably speaking, serverless computing is changing the model and process of future software development. In summary, serverless computing offers new opportunities such as affordable scalability, pay-as-you-go billing, off-the-shelf services. It has a wide range of application areas and can improve the efficiency of software development. As a result, for most applications, developers can devote most of their energy to the development and integration of business logic, greatly shortening the development cycle and reducing operation and maintenance costs by using serverless computing. However, some challenges, such as testing and debugging, must be addressed to remind of the developers the advantages as well as disadvantages of serverless computing before they adopt this architecture. Moreover, it is important to research bottleneck issues regarding utilizing serverless computing, which will encourage more developers to enter the serverless environment.

References

[1] Hassan, H.B., Barakat, S.A. & Sarhan, Q.I. Survey on serverless computing. J Cloud Comp 10, 39 (2021). https://doi.org/10.1186/s13677-021-00253-7

[2] Sarah, A., et al. 2018. CNCF Serverless Whitepaper. https://github.com/cncf/wg-serverless/blob/master/whitepapers/serverless-overview. [online].

[3] Shafiei, H., Khonsari, A. and Mousavi, P., 2019. Serverless computing: A survey of opportunities, challenges and applications. arXiv preprint arXiv:1911.01296.

[4] Taibi, D., Spillner, J. and Wawruch, K., 2020. Serverless computing-where are we now, and where are we heading?. IEEE Software, 38(1), pp.25-31.

[5] Jussi, N. and Davide, T., "Serverless: What it Is, What to Do and What Not to Do," 2020 IEEE International Conference on Software Architecture Companion (ICSA-C), 2020, pp. 49-50, doi: 10.1109/ICSA-C50368.2020.00016.

[6] Jindal, A., Gerndt, M., Chadha, M., Podolskiy, V. and Chen, P., 2021. Function delivery network: Extending serverless computing for heterogeneous platforms. Software: Practice and Experience. arXiv:2102.02330 [cs.DC]

[7] Castro, P., Ishakian, V., Muthusamy, V. and Slominski, A., 2019. The rise of serverless computing. Communications of the ACM, 62(12), pp.44-54.

[8] 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.

[9] Saint-Andre, P., 2008. XEP-0174: Serverless messaging. Standards track, XMPP Standards Foundation.

[10] Tu, Z., Li, M. and Lin, J., 2018, June. Pay-per-request deployment of neural network models using serverless architectures. In Proceedings of the 2018 Conference

of the North American Chapter of the Association for Computational Linguistics: Demonstrations (pp. 6-10).

[11] Naglieri, J., 2017. Streamalert: A serverless, real-time intrusion detection engine.

[12] Cheng, B., Fuerst, J., Solmaz, G. and Sanada, T., 2019, July. Fog function: Serverless fog computing for data intensive iot services. In 2019 IEEE International Conference on Services Computing (SCC) (pp. 28-35). IEEE.

[13] Poth, A., Schubert, N. and Riel, A., 2020, September. Sustainability Efficiency Challenges of Modern IT Architectures–A Quality Model for Serverless Energy Footprint. In European Conference on Software Process Improvement (pp. 289-301). Springer, Cham.

[14] Ali, A., Pinciroli, R., Yan, F. and Smirni, E., 2020, November. Batch: machine learning inference serving on serverless platforms with adaptive batching. In SC20: International Conference for High Performance Computing, Networking, Storage and Analysis (pp. 1-15). IEEE.

[15] Van Eyk, E., Grohmann, J., Eismann, S., Bauer, A., Versluis, L., Toader, L., Schmitt, N., Herbst, N., Abad, C.L. and Iosup, A., 2019. The SPEC-RG reference architecture for faas: From microservices and containers to serverless platforms. IEEE Internet Computing, 23(6), pp.7-18.