International Journal of Technology Engineering Arts Mathematics Science
Vol. 3, No. 2, July-December 2023, pp. 22~ 28
ISSN: 2583-1224,

Serverless Computing and Its Impact on Application Development in Cloud Environments

Mrs. Sayali Belhe[1], Ms. Sejal Barshikar[2], Ms. Swati Kadu[3]

[1,2,3] Artificial Intelligence and Data Science, AISSMS, IOIT, Maharashtra, India
Corresponding Author: Sejal Barshikar(sejalbarshikar@gmail.com)

Article Information

Article history:
Received Jun x, 20xx
Revised Nov x, 20xx
Accepted Dec x, 20xx

ABSTRACT

Serverless computing, also known as Function as a Service (FaaS), has gained significant attention in recent years as a paradigm for building and deploying applications in cloud environments. This research paper delves into the concept of serverless computing, its architecture, benefits, and challenges. This paper begins with an explanation of the key concepts behind serverless computing, with emphasis on its defining feature, Function as a Service (FaaS). It explores how serverless computing affects application development, scalability, and the overall cloud ecosystem. This provides an opportunity to explore the emerging trend of serverless computing and its implication on modern application development practices within cloud environments. Moreover, this paper discusses the impact of serverless computing on traditional cloud services and how it is transforming the way businesses deploy and manage applications in the cloud. Serverless computing is representation of new age cloud computing and shows the development of cloud programming models, abstractions, and platforms and demonstrates how widely used and developed cloud technologies are. Additionally, to further demonstrate effective serverless application implementations, real-world case studies are provided.

KEYWORDS: Serverless computing, Cloud, FaaS, Serverless benefits, Serverless challenges, Cloud Computing.

1. INTRODUCTION

Cloud computing has gained attention from all around the World over past few years. One can access cloud services through internet. Because of the benefits such as scalability, availability, and flexibility, software developers have been integrating cloud technology in their software solutions. Cloud offers 3 services in general: Infrastructure as a service (IaaS), Platform as a service (PaaS) and Software as a service (SaaS). Software as a service (Saas) delivers services over Internet. Google, for instance, offers several services such as Gmail, Google Docs, Google Sheets, and Google Forms. The user in this form of cloud is not in charge of service development, deployment, or management. Platform as a Service (PaaS) is a type of cloud computing that provides a platform and environment for developers to construct applications and services via the internet. PaaS services are cloud-based and accessible via a web browser. Cloud users in the Infrastructure as a service (IaaS) category control and manage services such as network access, servers, operating systems, and storage [7]. As efficient as these 3 services are, there are also some challenges which organizations must overcome while using them such as resource allocation, scalability, and load unbalancing and monitoring. These issues encouraged the introduction of the serverless cloud paradigm, another type of cloud computing.

Serverless computing enables cloud providers to handle the infrastructure and resources required to operate and grow applications automatically, freeing developers to concentrate simply on developing code to implement specified functionality. The management of servers, virtual machines, or containers is not a concern for developers with a serverless architecture. Rather, they

develop functions that are triggered by certain events or requests, and the cloud provider dynamically allots resources to carry out these activities as required. The existing cloud computing paradigms gain a new abstraction layer from serverless cloud computing, which also abstracts away server-side administration from developers [3].

A serverless architecture consists of many essential elements that function in parallel (Fig. 1.). Serverless functions, which are lightweight units of code that do tasks in response to events, are at its core. These events, such as HTTP requests, database updates, or timers, cause the execution of these routines.

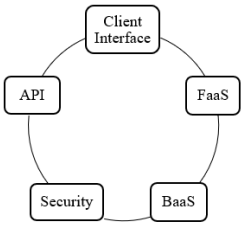

Fig. 1. Components of Serverless Architecture

An API Gateway serves as the entry point for external requests, sending them to the proper services and providing features like as authentication and rate limitation. The cloud provider controls the computing infrastructure dynamically, assigning resources to perform functions and scaling as needed. State storage services can be utilized to handle data across function invocations if desired. Because the cloud provider manages resource supply, scalability, and maintenance, developers may concentrate entirely on building code. When an event occurs the function is called, which processes the event and, if necessary, interacts with other services. The client receives the answer via the API Gateway, enabling for efficient and scalable application development without the difficulties of managing infrastructure [6].

Serverless cloud computing provides backend as a service (BaaS) and function as a service (FaaS). Storage, communications, and user administration are examples of BaaS services. Meanwhile, FaaS allows developers to install and run code on computing platforms. The FaaS is dependent on the BaaS services such as a database, communications, and user authentication. Backend as a Service (BaaS) is a cloud computing service paradigm that provides developers with a collection of pre-built backend services and functions that can be simply incorporated into their applications. BaaS streamlines the process of constructing an application's server-side components, allowing developers to focus on frontend features and user experiences. When coupled, BaaS and serverless computing provides a strong platform for rapidly developing and deploying applicationsGoogle App Engine, which released in 2008, was the first abstract serverless computing platform. Amazon Lambda, developed by Amazon in 2014 revolutionized the abstract serverless computing concept. In 2016, Google Cloud Platform launched a second serverless product, Google Cloud Functions. Oracle also offers a serverless platform known as Oracle Cloud Foundations.

Function as a Service (FaaS) is an important component of the serverless computing paradigm. FaaS is sometimes used interchangeably with the word "serverless," however it focuses on function execution in a serverless architecture. Each function is intended to carry out a given task or to implement a specific piece of functionality. This granularity allows developers to focus on writing code that directly addresses business requirements, promoting modularization and code reusability. These functions are stateless and can be triggered by a variety of events, including HTTP requests, database updates, file uploads, and scheduled activities. FaaS solutions manage infrastructure autonomously, supplying resources when a function is activated and scaling them based on demand. Because users are invoiced based on the actual execution time and resources consumed, this automated scaling assures maximum performance while keeping costs in mind.

The move away from traditional infrastructure and toward cloud-based serverless solutions implies a significant paradigm shift in the way applications are designed, deployed, and managed. This progression involves several revolutionary shifts that will fundamentally alter the way businesses and developers approach their technical landscapes. Traditional infrastructure management entails a variety of operational activities, ranging from server provisioning to monitoring and maintenance (Fig. 2). Cloud-based serverless solutions shift these tasks to the cloud provider. This hands-off management style allows developers to concentrate on activities that provide value rather than infrastructure problems.

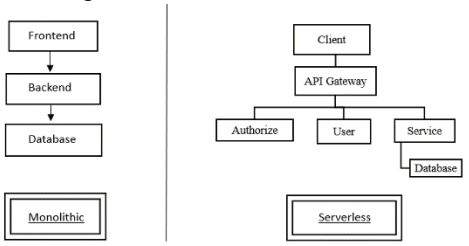

Fig. 2. Traditional infrastructure vs Serverless Cloud

2.METHODOLOGY

Serverless computing is a method of providing backend services on an as-needed basis. An organization that obtains a back-end service from a serverless provider will be charged depending on their computation and will not need to reserve or pay for a predetermined amount of bandwidth or the number of servers, since the service will scale automatically. During the early days of the web, anyone who needed to construct a web application had to have the fundamental physical gear required to run a server, which was a time-consuming and expensive job. Later came cloud computing, in which fixed numbers of servers or amounts of server space could be leased remotely. Developers and businesses who rent stationary server space typically over-buy to ensure that a spike in traffic or activity does not surpass their normal limitations and disrupt their own applications. Serverless has made it simpler [2].

2.2. ADVANTAGES

2.2.1. Pay-as-you-go Model

Instead of pre-allocated resources, serverless solutions charge depending on real consumption. Businesses only pay for the computational resources used during serverless function execution. This removes the need to supply and pay for idle resources, resulting in cost savings as depicted in Fig. 3.

2.2.2. Resource Scaling

In a serverless architecture, resources scale up and down autonomously in response to demand. Functions are performed in separate containers, and the cloud provider maintains container scalability. This dynamic scaling guarantees that the program can manage variable amounts of demand without overprovisioning, reducing resource waste.

2.2.3. No Idle Resource Cost

Traditional server configurations sometimes need keeping a certain amount of capacity to meet possible traffic surges. This results in idle resource expenses during periods of low demand. Because resources are allocated only when a function is activated, serverless reduces these idle expenses[9].

2.2.4. Effective Resource allocation

Functions are performed in separate settings, ensuring that each function receives the resources it requires while without interfering with others. This effective allocation avoids over-provisioning and under-utilization.

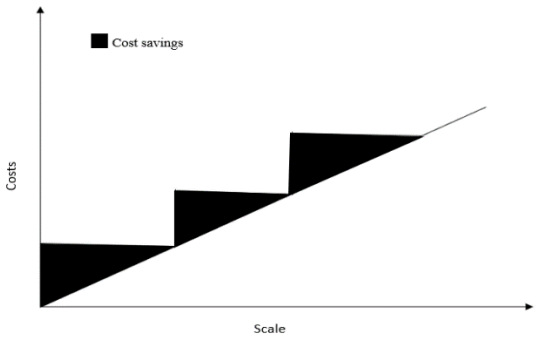

Fig. 3. Cost comparison between cloud server computing and serverless cloud computing

2.2.5. Operational Simplicity

FaaS frees developers from having to handle their infrastructure. They are not responsible for managing, scaling, or procuring servers. Developers may concentrate exclusively on writing the functions because the cloud provider takes care of these operational details.

2.2.6. Global Scalability

Serverless applications can be deployed across multiple regions, allowing businesses to serve users worldwide with minimal effort. The cloud provider manages the distribution of functions, making it easier to scale globally.

2.3. CHALLENGES

2.3.1. Cold Start latency

The problem of cold start delay might affect the effectiveness and responsiveness of serverless operations. The cloud provider must offer the appropriate resources for execution whenever a function is called for the first time or after a period of inactivity[1].

2.3.2. Vendor Lock-In

When implementing serverless computing, vendor lock-in is a risk since proprietary offers connect programs to certain cloud providers. Due to different APIs, services, and configurations, this might make switching providers or moving apps more difficult.

2.3.3. Complexity in Managing Distributed Systems

Due to the inclusion of multiple functions, services, and integrations, managing distributed systems in serverless architectures can become complicated. It is important to pay close attention while monitoring and troubleshooting these dispersed components.

2.3.4. Resource Limitation and Constraints

Functions in serverless systems are subject to resource restrictions such execution time, memory, and storage restrictions. These restrictions may influence the complexity and reach of applications.

2.4. APPLICATION DEVELOPMENT

The development of serverless apps follows a simplified methodology that emphasizes code creation while decoupling infrastructure administration. It starts by establishing certain functions that encompass various activities performed by the program. These functions may be altered to carry out a variety of tasks and are written in languages like Python or JavaScript. One can choose from several serverless platforms offered by cloud providers, such as AWS Lambda, Azure Functions, or Google Cloud Functions, to deploy these functions. These systems provide tools for packaging and deploying functions while automatically taking care of execution, scalability, and deployment complexities.

Enhancing the application's functionality requires integration with other services, which is a critical step. Functions can effortlessly interface with services like databases, storage, and authentication systems by using APIs and SDKs offered by the serverless platform. Use of package managers like npm or pip, which handle dependencies on external libraries, ensures the efficient and reliable operation of functions. The event-driven nature of serverless architecture is one of its distinguishing characteristics. Events, which include actions or occurrences like HTTP requests, database changes, or scheduled activities, are what start serverless operations. These occurrences act as triggers to initiate the corresponding serverless functions' automated execution. The circumstances that trigger functions are set by configuring event sources (Fig. 4). This arrangement is flexible and adjustable, allowing for a range of event kinds and sources. A function can reply to incoming HTTP requests, for example, by setting an HTTP endpoint. A hallmark of serverless platforms is their inherent ability to scale functions dynamically. The platform automatically adds new instances of the necessary functions as event frequency rises to handle the strain. On the other hand, surplus instances are eliminated during times of less activity. This flexibility guarantees responsiveness of the application without requiring manual intervention.

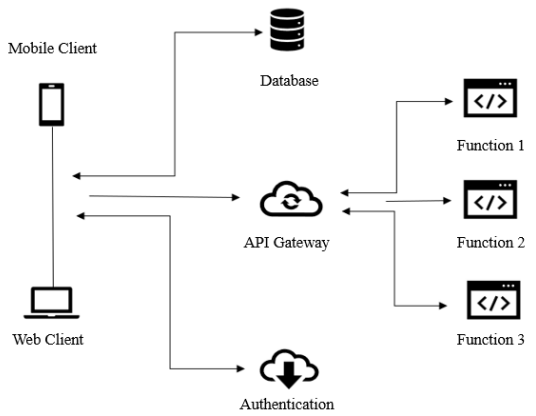

Fig. 4. Serverless Application Development

Throughout the process, the serverless platform's monitoring and debugging tools assist developers in measuring performance metrics, identifying faults, and optimizing resource use. This iterative method to designing serverless apps, which emphasizes efficient code, event-driven architecture, and automatic scaling, helps to create systems that are not just nimble but also cost-effective.

3.DISCUSSION

3.1. EXISTING MODELS

There are several companies that provide as well as use serverless computing for their businesses. Azure, Google Cloud functions and Microsoft are major, popular companies that provide serverless computing platforms. Top companies that use serverless include Amazon, Netflix, Airbnb, Nordstrom, Coca-Cola, Capital One, Zillow.

3.1.1. Azure Functions

Azure Functions is Microsoft's serverless computing platform, which allows developers to create and deploy event-driven, scalable, and cost-effective applications without having to manage infrastructure. This service enables developers to concentrate entirely on creating code to perform specific tasks or respond to triggers, while Azure handles the underlying provisioning, scalability, and maintenance. Azure Functions supports a number of event sources, including HTTP requests, updates in Azure Cosmos DB, incoming messages from Azure Service Bus, and more. Languages such as C#, JavaScript, PowerShell, Python, and Java are available to developers, giving them the freedom to work with their favourite programming languages. One notable feature of Azure Functions is its tight integration with other Azure services. Developers can easily create workflows by connecting functions with Azure Logic Apps, respond to events using Azure Event Grid, or integrate with Azure Storage and Azure SQL Database. Additionally, Durable Functions, an extension to Azure Functions, simplifies the creation of stateful and orchestrating workflows by managing the state and flow of execution. Binding extensions further enhance the development process by seamlessly integrating input and output data with external services, databases, and queues. Several prominent companies that use Azure Functions include Alaska Airlines, The New York Times and Adobe [10].

3.1.2. AWS Lambda

AWS Lambda, a key component of Amazon Web Services' serverless solutions. AWS Lambda allows you to run code in response to multiple triggers, such as HTTP requests, Amazon DynamoDB data updates, file uploads to Amazon S3, and more. This serverless

computing service supports a variety of programming languages, including Node.js, Python, Java, Ruby, Go, and.NET Core, to meet the needs of a wide spectrum of developers. WS Lambda's event-driven approach ensures automatic scaling and efficient resource utilization. The service automatically manages the provisioning and scaling of resources based on the incoming workload, reducing operational overhead, and optimizing costs. Cold starts, a challenge in serverless environments, have been mitigated over time through improved optimizations and performance enhancements [11].

The concept of "Lambda Layers" allows for efficient code reuse across multiple Lambda functions, streamlining development processes and promoting best practices. With a variety of execution environments and memory options, Lambda can be tailored to suit different application requirements. Prominent companies like Netflix, Philips Hue, and Ticketmaster have harnessed AWS Lambda's capabilities to develop innovative solutions.

3.1.3. Google Cloud Functions

Google Cloud Functions, a part of Google Cloud's serverless suite, enables developers to build event-driven apps without having to manage underlying infrastructure. This serverless computing service lets you develop code that responds to triggers from Google Cloud services including Cloud Storage, Cloud Pub/Sub, and Firebase Realtime Database. Google Cloud Functions supports a variety of programming languages, including JavaScript (Node.js), Python, and Go, allowing developers to work in familiar environments. Google Cloud Functions introduces the notion of "Cloud Functions for Firebase," which enables developers to create serverless backends for their mobile or web applications that use the Firebase platform. This connection speeds up iterations and feature releases by simplifying development and deployment procedures [11]. Companies such as Spotify, Ticketmaster, and others have used Google Cloud Functions to build event-driven, scalable, and responsive apps. Google Cloud Functions provides a rich framework for developing serverless apps that utilize Google Cloud's infrastructure and services, whether one is building real-time data processing solutions, chatbots, or IoT applications.

3.2. SERVERLESS SECURITY

3.2.1 Event injection

In serverless computing, event injection is a serious security problem in which attackers attempt to enter malicious information or payloads into event data activating serverless functions. This can have serious ramifications, such as code execution vulnerabilities and unauthorized access to vital resources. Several security measures should be put in place to counteract this threat. To begin with, strong input validation and sanitization methods must be in place to guarantee that event data is fully reviewed and cleansed before being processed by serverless functions. Furthermore, following the concept of least privilege is critical; this entails limiting the permissions provided to serverless services to the minimal minimum required for their intended duties. (Fig. 5).

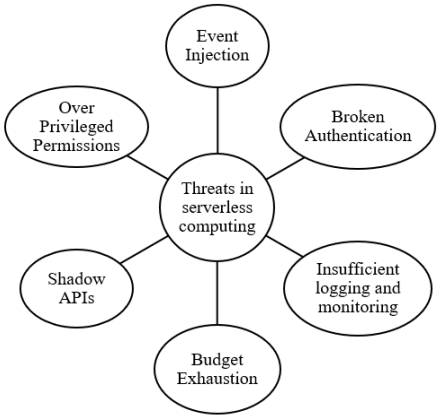

Fig. 5. Threats in serverless security

3.2.2. Broken Authentication

In serverless systems, broken authentication vulnerabilities pose a substantial danger, possibly allowing unwanted access to important data and functionality. To successfully solve this issue, it is critical to develop strong authentication and permission procedures. One important way is to use your cloud provider's Identity and Access administration (IAM) services, which allow centralized control and administration of access rights for serverless tasks. Consider using OAuth or JSON Web Tokens (JWT) to provide secure authentication and authorization protocols between clients and serverless operations, guaranteeing that only authenticated and authorized users or systems may interact with your resources. Furthermore, promoting or requiring Multi-Factor Authentication (MFA) improves security by providing an extra degree of verification, making it far more difficult for unwanted organizations to get access [4].

3.2.3. Budget Exhaustion

A significant problem, in serverless computing is the depletion of budgets. When using serverless activities it's important to control resource usage to avoid expenses. To tackle this issue a diversified approach is necessary. Implementing resource quotas and restrictions for serverless functions is crucial as it helps prevent execution and ensures that costs stay within defined boundaries. Additionally, organizations need to have monitoring and alert systems in place. By monitoring resource

consumption and setting up alerts for patterns or sudden spikes in expenses concerns can be promptly identified and addressed ensuring adherence, to budget constraints. Lastly utilizing auto scaling policies can further help minimize costs.

3.2.4. Insufficient logging and monitoring

Inadequate logging and monitoring represent a serious obstacle to maintaining system and application security. When security events go unreported owing to poor recording and monitoring processes, it can cause delays in recognizing and responding to possible security issues, leaving businesses open to hostile activity. A multifaceted strategy is required to successfully address this issue. To begin, businesses should establish extensive logging to ensure that all important events, including as login attempts, resource access, and API usage, are thoroughly captured and securely preserved. Furthermore, utilizing Security Information and Event Management (SIEM) systems helps centralize log management, correlation, and real-time monitoring, improving the capacity to detect and respond to security threats quickly.

3.3. PERFORMANCE ANALYSIS

The potential for unmatched scalability and enhanced performance in the world of serverless computing is genuinely revolutionary. However, in order to fully take advantage of these advantages, it is essential to do in-depth performance and scalability assessments and follow best practices that guarantee the smooth operation of serverless applications. Benchmarking serverless performance is a crucial technique. The serverless application's management of the workload and use cases must first be carefully defined. When creating the testing environment, great effort is taken to make sure that it closely resembles the production configuration, complete with accurate data and an imitation of the anticipated workload. The gathering of crucial performance information, which includes important indicators like execution times and resource usage, takes centre stage during this phase. Notably, the load testing procedure is of utmost importance since it gradually increases the load on the application and provides priceless insights into how the program performs under increased demand. Additionally, a performance profiling step frequently proves beneficial in identifying bottlenecks and inefficient portions of the software. At the same time, it is crucial to comprehend the scalability thresholds [5].

Despite their outstanding scalability, serverless solutions have several drawbacks that call for caution. The idea of concurrent execution restrictions stands out among these. These restrictions highlight the significance of carefully observing and then revising the application's design to guarantee a seamless user experience. Furthermore, careful thought must be given to the area of cold beginnings, where functions are started and initialized. Even though serverless systems have made progress in reducing cold start latency, coming up with tactics like pre-warming functions is still crucial in lessening their effects. When looking for greater performance and scalability, optimization is a key concept [8]. The careful distribution of resources through the selection of appropriate memory configurations for functions is essential. Smooth performance is a result of both effective application state management and minimizing the storing of large amounts of data within functions. Throughput and latency can increase significantly with the use of asynchronous processing. Best practices include architectural concerns in addition to resource optimization. Utilizing caching systems, which save frequently requested data, guarantees a decrease in duplicate calculations, and thereby improves response times. The notion of service decomposition enters the picture to boost parallelism even further and optimize resource use. Such knowledge makes it possible to dynamically allocate resources in response to changes in demand, especially when combined with the strength of auto-scaling capabilities.

4.CONCLUSION

In Conclusion, this paper has delved into the intricacies of serverless computing and explored its profound impact on the way applications are conceptualized, developed, and deployed. This paper highlighted the agility and efficiency that serverless computing provides. Serverless computing has freed developers from infrastructure administration and resource provisioning, as evidenced by an examination of its architecture and underlying event-driven approach. This move allowed developers to focus their efforts on creating unique functionality and features that directly address the demands of end users. The real-world case studies demonstrated the adaptability of serverless applications in a variety of sectors.

Concerns about vendor lock-in and serverless cold start delay underscored the importance of careful design and strategic planning. Although not unique to serverless, the security and compliance components gained new complexity in the context of dynamic, event-triggered systems. The influence of serverless computing on application development is apparent. Its potential to foster innovation, enhance scalability, and streamline cost-efficiency represents a new frontier for developers and businesses alike.

Serverless computing, which is in line with the wider industry trends of decoupling, automation,

and efficiency, is a testament to how quickly the cloud computing space is moving. To survive in this new era of cloud-driven application development, businesses must understand the nuances of serverless computing and use its benefits wisely. While the road ahead may involve more improvements in cold start performance, security standards, and standardized development methods, the overall trend leads toward a world in which developers may unleash their creativity without the constraints of infrastructure administration. Serverless computing is more than just a technology trend; it's a conceptual change that opens the door to cloud computing innovation and advancement.

5.REFRENCES

[1] Baldini I, Castro P, Chang K, Cheng P, Fink S, Ishakian V, Mitchell N, Muthusamy V, Rabbah R, Slominski A, Suter P, "Serverless Computing: Current Trends and Open Problems In: Research Advances in Cloud Computing", 1–20. Springer, Singapore, 2017.

[2] Vaishnavi Kulkarni, "A Research Paper on Serverless Computing", INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 11, Issue 09 (September 2022)

[3] Raj, Pethuru & Vanga, Skylab & Chaudhary, Akshita, "Serverless Computing for the Cloud-Native Era", 10.1002/9781119814795.ch8, 2022.

[4] Ortega, José Manuel & Elouali, Aya & Mora Gimeno, Francisco & Mora, Higinio, "Cloud vs Serverless Computing: A Security Point of View", 10.1007/978-3-031-21333-5_109, 2022.

[5] Jinfeng Wen, Zhenpeng Chen, Xin Jin, and Xuanzhe Liu, "Rise of the Planet of Serverless Computing: A Systematic Review", ACM Trans. Softw. Eng. Methodol. 32, 5, Article 131 (September 2023).

[6] G. McGrath and P. R. Brenner, "Serverless Computing: Design, Implementation, and Performance," 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW), Atlanta, GA, USA, 2017, pp. 405-410.

[7] Sheth, Mrs & Bhosale, Sachin & Kadam, "Research Paper on Cloud Computing", 2021.

[8] Li, Yuze & Yao, Shunyu. "Understanding and Optimizing Serverless Workloads in CXL-Enabled Tiered Memory", 2023

[9] Kumari, Anisha & Sahoo, Bibhudatta & Behera, Ranjan. "Workflow aware analytical model to predict performance and cost of serverless execution. Concurrency and Computation Practice and Experience". 35. 10.1002/cpe.7743, 2023

[10] Sawhney, Rahul "Beginning Azure Functions, Building Scalable and Serverless Apps". 10.1007/978-1-4842-4444-9, 2009

[11] Malawski, Maciej & Gajek, Adam & Zima, Adam & Balis, Bartosz & Figiela, Kamil. "Serverless execution of scientific workflows: Experiments with HyperFlow, AWS Lambda and Google Cloud Functions. Future Generation Computer Systems", 2017