Available online www.jsaer.com

Journal of Scientific and Engineering Research, 2022, 9(9):126-135

Research Article

ISSN: 2394-2630
CODEN(USA): JSERBR

Evolving Cloud-Native Architectures: Leveraging Serverless Computing for Flexibility and Scalability in Applications

Venkata Baladari

Software Developer, Newark, Delaware, USA
Email ID: vrssp.baladari@gmail.com

Abstract: Cloud-native architectures and serverless computing have revolutionized application deployment, improving scalability, reducing costs, and increasing operational flexibility. Optimizing performance, resource allocation, and automating workflows continues to be a difficult challenge. This paper discusses the latest developments in serverless computing, focusing on its importance for building scalable applications. Key developments in container-based platforms, workflow execution, and function choreography are explored. Furthermore, we evaluate the combination of serverless computing with AI, scientific workflows, and high-performance computing. The research study uncovers performance limitations including cold start latency and shortcomings in resource allocation, whilst also examining methods for streamlining workflows and redistributing workload. Additionally, it considers issues such as cold start latency and resource provisioning inefficiencies. This study synthesizes existing research to offer insights into the advantages, obstacles, and potential future paths for improving cloud-native and serverless infrastructure designs.

Keywords: Cloud-Native Computing, Serverless Architectures, Function-as-a-Service, Serverless Computing, Service Mesh

1. Introduction

Cloud-native architectures have been significantly redefined by architectures that prioritize scalability, agility, and automation. Serverless computing plays a crucial role in this approach by handling infrastructure management automatically, giving developers the freedom to concentrate on the application's logic and take advantage of scalable and cost-effective resource allocation. Serverless architectures are well-suited for data processing, AI applications, and scientific computing due to their ability to eliminate the need for manual provisioning, thereby supporting highly dynamic workloads.

Serverless computing offers several benefits, but also presents difficulties including slow initialization times, limited execution time allotments, and inefficient workload allocation. Studies have investigated several methods to enhance its efficiency. The execution of scientific workflows within serverless platforms, with a particular emphasis on the balancing act between resource allocation. Hybrid models that combine serverless computing with containerized environments have been found to enhance workload efficiency.

Serverless adoption's role in AI and data processing is increasingly significant as its use continues to expand. The deployment of large-scale machine learning workloads in serverless environments demonstrates their ability to manage computationally demanding tasks. GPU-enabled serverless workflows result in enhanced efficiency for multimedia applications through hardware acceleration [6].

In addition to application-specific improvements, current research emphasizes the need to optimize serverless workflow processing. Function choreography language to simplify intricate workflows, and load-balancing

approach to manage resource distribution in uncertain cloud environments. Ongoing improvements in serverless architectures are aimed at boosting performance, minimizing latency and increasing adaptability.
This paper examines recent developments and hurdles in serverless computing within cloud-native frameworks, focusing on performance enhancement, workload control, and emerging trends which will define its future direction.

Write (little) code
Define triggers
Connect managed services together
Pay for consumption instead of allocation

Figure 1: What you do in Serverless Systems
(Accessed from https://seroter.com/wp-content/uploads/2019/04/2019.04.18-serverless-01-2.png)

2. Fundamentals of Cloud-Native and Serverless Computing

Cloud computing has transitioned to an era of cloud-native infrastructure and serverless platforms, offering developers scalable, resilient, and efficient computing environments. These paradigms are defined by containerization, the use of microservices, event-driven processing, and automated resource allocation and management. This section offers a detailed examination of cloud-native computing principles, serverless architectures, and comparisons with traditional cloud models drawing on the references supplied.

Cloud-Native Computing: Principles and Evolution

Cloud-native computing is based on the core principles of containerization, the use of microservices, automation, and orchestration. The shift from monolithic systems to microservices has made it possible for organizations to achieve greater scalability, improved fault tolerance, and increased deployment flexibility. Applications can be bundled with their required components using containers, guaranteeing uniform execution regardless of the cloud infrastructure being utilized. Container orchestration tools take automation to the next level by overseeing workload scheduling, auto-scaling and self-healing functions.
Extensive research has been conducted into integrating serverless computing within containerized environments, resulting in notable enhancements to performance and workload efficiency. The integration of on-demand execution of functions with container-based isolation enables applications to scale dynamically, all while preserving optimal resource management. In high-performance computing, serverless methods have also been investigated for data-intensive workflows, demonstrating advantages for processing models driven by events.
Cloud-native computing relies heavily on automation, allowing container orchestration platforms to facilitate real-time scaling and workload allocation. Microservices architectures have been combined with event-driven workflows to improve both resource management and the efficiency of task execution processes. These systems allocate resources in real-time based on workload intensity, making them well-suited for computing environments with unpredictable demands.

Serverless Computing: Definition and Execution Model

Serverless computing streamlines infrastructure management by running applications as on-demand services in response to specific events. The system operates on a pay-per-use basis, providing cost effectiveness by invoicing users only for the actual time their tasks are executed. The classification of this model consists of:
Function-as-a-Service (FaaS) – Cloud providers manage stateless function execution and scaling without the need for manual configuration [7][9].
Backend-as-a-Service (BaaS) – This model delivers pre-configured back-end services like authentication, databases, and messaging systems, thereby facilitating quick application development [7].

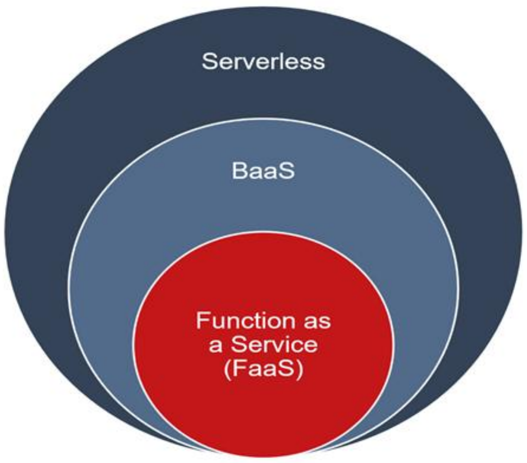

Figure 2: Serverless Architecture
(Accessed from https://serverless-architecture.io/wp-content/uploads/2021/02/pientka_headless_1.png)

FaaS has been utilized by scientific workflows and high-performance computing applications to dynamically scale computational tasks, thereby achieving efficiency in handling fluctuating workloads. Stateful and long-running applications face difficulties due to execution constraints and function timeout limits. Serverless computing has also positively impacted AI and machine learning workloads, particularly in model inference tasks, where auto-scaling capabilities facilitate efficient resource allocation.

Event-Driven Computing in Serverless Architectures

Serverless architectures function through event-driven triggers, facilitating real-time processing of data streams, HTTP requests, and IoT signals [7]. Triggering functions only when necessary helps to conserve resources effectively. Function orchestration languages to increase efficiency in workflow automation by facilitating improved collaboration between numerous serverless functions. The use of these orchestration techniques minimizes unnecessary function calls, resulting in enhanced execution efficiency[5].

Researchers suggest combining function orchestration with load-balancing methods to facilitate large-scale distributed workflows, achieving maximum function execution efficiency and equitable resource allocation. These frameworks improve the scheduling of serverless workloads, thereby resolving frequent execution bottlenecks.

3. Cloud-Native Architecture Components and Technologies

Cloud-native architectures are engineered to optimize scalability, robustness, and automated processes within contemporary cloud settings. They utilize containerization, microservices, function orchestration, and service mesh technologies to optimize resource use while ensuring high availability and flexibility. This section delivers an in-depth analysis of cloud-native architectural components and their function within serverless computing, with the aid of the referenced materials.

Containers and Kubernetes

Cloud-native architecture relies on containerization as a core component, allowing for the deployment of lightweight and portable applications. Applications and their dependencies are packaged inside containers, resulting in consistent deployment across various environments. This approach boosts flexibility, scalability, and resource effectiveness, particularly in serverless computing, which demands swift provisioning and execution. Kubernetes, an open-source system for managing containers, automates the deployment, scaling, and administration of containerized workloads [1]. Extensive research has been conducted to integrate Kubernetes with serverless frameworks, showcasing its efficiency in handling event-driven applications and also its ability to manage workload variability and optimize resource usage. Studies have shown that integrating container-based architectures with serverless computing models can enhance system dependability and decrease operational costs. In addition, research on hybrid methodologies indicates that utilizing Kubernetes for workload orchestration enables businesses to deploy cloud-native applications smoothly across private and public cloud environments, thereby improving adaptability in dynamic computing environments [3].

Microservices and their role in Serverless Computing

Implementing microservices architecture is a crucial factor that facilitates serverless computing, by encouraging the breaking down of complex systems into smaller, more manageable components and simplifying system maintenance. Breaking down applications into separate, standalone services enables microservices to support the swift creation, rollout, and expansion of software. In serverless systems, each microservice is usually developed as a function, this enables event-driven operation and cost-effective use of resources. The model increases the system's ability to withstand disruptions since problems with one service will not affect the entire system. Research has highlighted the benefits of using microservices in enhancing performance, decreasing time delays, and increasing the speed at which applications respond. Research has shown that microservices-based function orchestration is a viable approach, which involves combining multiple serverless functions to create workflows, thereby facilitating efficient parallel processing and distributed execution. Microservices architecture complements serverless computing by allowing for dynamic scaling of applications and more efficient distribution of workloads according to fluctuating demand.

Service Mesh for Managing Cloud-Native Applications

Managing communications between microservices within cloud-native applications becomes more difficult as their complexity rises. Service mesh architectures solve this problem by offering a specifically designed infrastructure layer for communication between services. This approach abstracts network management functions including traffic routing, load balancing, security, and observability [10]. Integrating a service mesh improves the scalability and dependability of serverless applications by facilitating continuous communication between services without necessitating alterations to the application's programming code. Studies have investigated multiple service mesh deployments in cloud-based serverless settings, highlighting their contribution to enhanced security, fault resilience, and network optimization. Studies have shown that integrating service mesh with Kubernetes-based workloads is an effective approach to optimizing communication patterns, lowering latency, and automating failure recovery. Advances in this area contribute to the stability and performance of cloud-native applications, making service mesh a vital part of contemporary serverless architectures.

API Gateways and Event-Driven Architectures

API gateways serve a crucial function in cloud-native and serverless computing by operating as intermediaries between client applications and backend services. The system handles request routing, authentication, and rate limiting, and also offers security and monitoring features. API gateways facilitate smooth communication between microservices and external systems, thus enabling the creation of scalable and efficient serverless applications [7]. Serverless computing is based on event-driven architectures, which allow applications to react automatically to events triggered by different sources, including user engagement, data updates, and system prompts. Studies have underscored the significance of API gateways in streamlining request processing and minimizing latency within serverless settings. Moreover, research on event-driven models has shown their capabilities in managing distributed workflows, automating task execution, and enhancing resource utilization. The integration of API gateways and event-driven architectures enables cloud-native applications to scale seamlessly, preserving high performance and reliability [5].

4. Serverless Computing - Mechanisms and Execution Models

Serverless Platforms: AWS Lambda, Google Cloud Functions, Azure Functions

Leading cloud service providers offer serverless platforms that have been tailored to meet the requirements of various types of workload. One of the first and most commonly used serverless solutions, AWS Lambda offers precise auto-scaling capabilities and effortless integration with Amazon Web Services. Google Cloud Functions primarily focuses on event-driven processing and is frequently utilized alongside Google Cloud's associated platforms, such as BigQuery and Firebase. Azure Functions provides comprehensive support for both hybrid and multi-cloud deployments, seamlessly integrating with Microsoft's enterprise cloud services. The mentioned platforms vary in areas such as the execution environment, supported programming languages, scalability tools, and pricing models. Research on these platforms indicates that AWS Lambda typically delivers improved cold start performance optimization, in contrast, Google Cloud Functions consistently shows superior latency for particular data-intensive tasks. In contrast, Azure Functions has received recognition for its enterprise-focused automation abilities. The choice of platform is determined by workload demands, integration requirements, and budgetary factors, as each supplier has distinct scaling strategies and operational limitations.

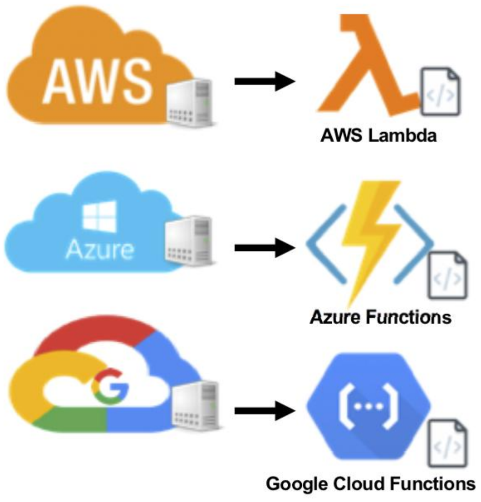

Figure 3: Serverless Platform
(Accessed from https://wisr.cs.wisc.edu/img/serverless-security/setup.png)

Performance and Scalability Considerations
Serverless computing scalability is achieved via automated resource allocation, enabling functions to scale automatically in response to incoming requests. This model reduces over-provisioning and maximizes resource efficiency, but still faces issues like cold starts and execution time constraints that hinder its performance. Initialization delays are caused when an idle function is invoked after a period of inactivity, resulting in container provisioning issues. Methods such as pre-warmed instances and optimized scheduling have been investigated to reduce these delays. Studies of actual serverless workloads have found that performance differences can be attributed to factors including memory allocation, concurrency restrictions, and the type of workload. Serverless platforms employ dynamic load-balancing mechanisms that distribute incoming requests across multiple function instances, thereby optimizing resource allocation. Unpredictable workloads can lead to performance slowdowns, requiring intelligent scaling algorithms and predictive workload scheduling to improve execution efficiency.

Cost Models and Billing in Serverless Computing
The serverless billing structure is founded on a pay-as-you-go methodology, where users are billed for the actual processing time and resources utilized, as opposed to pre-designated infrastructure. This cost-efficient pricing model is best suited for applications that experience intermittent workloads, but may become prohibitively costly for tasks that run frequently and continuously over a long period. The costs associated with serverless platforms are mainly influenced by the duration of execution, the amount of memory allocated, and the total number of invocations. This model eliminates the necessity for capacity planning, but introduces variability in costs, thereby complicating the forecasting of expenses for intricate applications. Research on the cost-effectiveness of serverless computing has shown that serverless architectures are more cost-efficient for applications that are short-lived and driven by specific events compared to traditional cloud deployments. For computationally demanding tasks that run continuously, alternative pricing plans or mixed systems combining serverless technology with containerized workloads may lead to more cost-effective solutions. Studies have also looked into cost-conscious scheduling methods to reduce expenses while preserving performance efficiency in cloudless environments.

5. Challenges In Cloud-Native and Serverless Computing
The shift to cloud-native and serverless computing has brought about substantial improvements in scalability and operational efficiency. The introduction of these paradigms presents several challenges that need to be

resolved to improve their functionality and user experience. Optimization and research efforts are primarily concentrated on addressing key challenges such as cold start latency, execution time constraints, vendor lock-in, security concerns, and resource allocation bottlenecks.

Cold Start Latency and Execution Time Limits

One of the major downsides of serverless computing is the cold start latency issue, which arises when a function is triggered after a period of inactivity, necessitating the underlying platform to set up the runtime environment prior to execution. This introduces significant delays, particularly for applications that require low latency. The effects of initialization are noticeable in Function-as-a-Service (FaaS) platforms, where the overhead of setting up affects the efficiency of workflows. Methods including pre-warming systems and intelligent function scheduling have been suggested to reduce these delays. Serverless providers' execution time limits hinder the viability of lengthy processes, necessitating applications to be divided into smaller functions or created with stateful orchestration. These constraints present difficulties for workloads like large-scale machine learning and high-performance computing, where unbroken execution is essential [9].

Vendor Lock-in and Portability Issues

Dependence on cloud providers for serverless execution raises concerns about vendor lock-in, hindering data portability across various cloud platforms. Serverless services have proprietary implementations provided by each provider, complicating the process of migrating applications that require significant changes. The complexity of portability is increased by disparities in execution environments, supported runtimes, and billing models. Exploring multi-cloud approaches and using open-source tools has been considered to decrease reliance on a single vendor, however, achieving smooth interaction between systems continues to be an obstacle. The absence of standardized serverless APIs and deployment methods hinders organizations from exploiting the flexibility of multi-cloud environments. Efforts to introduce serverless solutions packaged in containers and hybrid cloud models are focused on overcoming these constraints, allowing for more flexible and portable architectural designs.

Security and Compliance Concerns

The security of cloud-native and serverless systems is a significant issue due to their functions being short-lived and the fact that they run multiple tenants at the same time. The stateless execution of functions hinders traditional security measures, including network isolation and thorough monitoring. Frequent creation of new function instances significantly expands the attack surface, thereby putting applications at risk of data breaches and unauthorized access. Concerns about compliance often come up when dealing with sensitive workloads, as companies need to make sure they are protecting data and following relevant regulatory guidelines. Current security measures, such as fine-grained access controls, secure data transfer, and real-time monitoring, only offer partial solutions. Serverless deployments' dynamic nature requires more comprehensive security frameworks that can adjust to changes in function lifecycles and offer ongoing assurance of compliance.

Resource Allocation and Performance Bottlenecks

Optimizing the performance of serverless applications requires efficient resource management. The unpredictable nature of function execution patterns poses a significant challenge in dynamically allocating resources while aiming to maintain cost efficiency. Load balancing systems need to cope with fluctuating workloads without introducing excessive processing delays or inefficiencies. In serverless environments, workload scheduling frequently overlooks the constraints of the underlying infrastructure, resulting in resource conflicts and inefficient execution processes. Studies on workload-aware scheduling and intelligent provisioning have shown enhancements in managing high-demand situations, yet attaining the best possible performance continues to be an ongoing problem. Advancements in resource orchestration and auto-scaling techniques will be essential for improving the reliability and efficiency of serverless computing environments.

6 Advancements and Optimization Strategies

Efficient Workload Scheduling and Orchestration

Significant research activity has resulted from the growing use of serverless computing, focusing on streamlining workload scheduling and orchestration processes. A major obstacle is handling irregular execution patterns, necessitating dynamic scheduling techniques. Research has investigated function orchestration methods that enable the smooth execution of interconnected workloads while reducing execution overhead.

Frameworks for workflow management have been suggested to automate the dynamic execution of paths, thereby ensuring that function calls are efficiently scheduled according to the availability of resources and their respective execution priorities. Moreover, models that utilize event-driven triggers have been introduced to improve performance by minimizing the time taken for function invocation. These advancements improve the reliability and scalability of serverless applications, guaranteeing efficient use of resources across a range of computing environments.

Auto-Scaling and Load Balancing in Serverless Platforms

Serverless computing relies heavily on auto-scaling mechanisms to maintain its operational efficiency. Due to the inherently dynamic nature of serverless workloads, strategies have been created to manage sudden increases in demand without impacting performance. Conventional scaling methods frequently encounter issues with cold-start latency, resulting in functions that take longer to initialize during scale-out processes [7]. Current advancements in innovation concentrate on predictive scaling, with the aim of pre-warming functions based on observed historical execution patterns to minimize the impact of cold starts. Furthermore, load-balancing techniques have been implemented to ensure an even distribution of workload across accessible resources, thereby avoiding performance bottlenecks [10]. Research has shown that implementing intelligent scaling policies which adjust to fluctuating workload requirements can lead to cost savings and faster execution times. These techniques facilitate the efficient management of high-volume workloads and maximize the effective use of available resources.

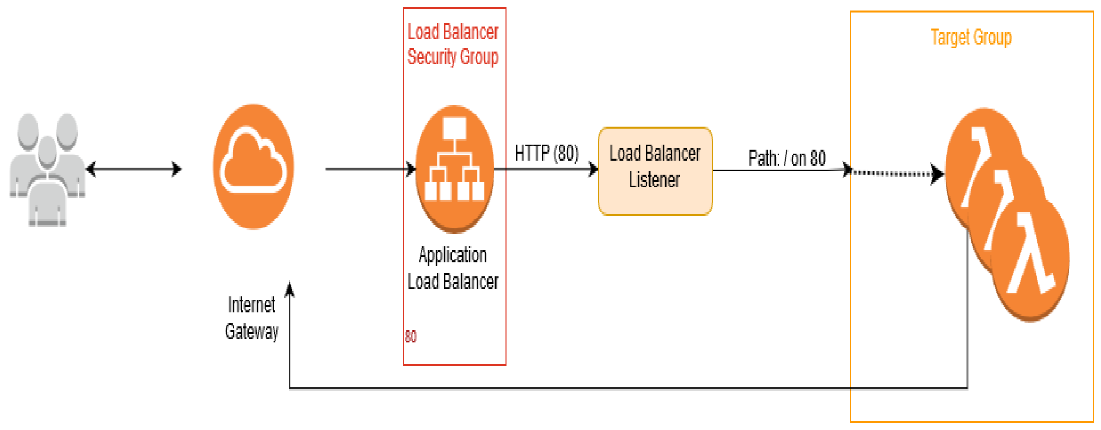

Figure 4: Application Load Balancer
(Accessed from https://www.rehanvdm.com/contents/data/2019/09/AutoScale_3_LambdaALB.png)

Enhancements in Function Choreography and Workflow Management

The choreography of functions in serverless computing has developed to handle the intricacies of running multiple interconnected functions. In contrast to conventional orchestration, which relies on a centralized controller to oversee task execution, choreography facilitates distributed execution where functions exchange information on an asynchronous basis [5]. Recent improvements in function coordination mechanisms have led to the creation of abstract languages that facilitate smooth workflow execution, resulting in decreased execution time and lower resource contention. Today, workflow automation techniques include recovery procedures, enabling partially finished workflows to restart without duplication. The enhancements in function choreography significantly benefit event-driven applications that require multiple services to collaborate seamlessly and automatically without human intervention. The ongoing development of workflow management frameworks has increased the versatility of serverless computing for use in large-scale distributed systems [2][3][5].

AI and Machine Learning Integration in Serverless Environments

The deployment of artificial intelligence and machine learning technologies in serverless settings has enabled the development of highly scalable applications powered by AI [9]. Serverless platforms enable efficient processing of machine learning inference tasks by eliminating the requirement for dedicated infrastructure. Optimizations in model deployment and execution have been prompted by challenges including high execution latency and resource constraints. Function partitioning techniques enable the division of machine learning models into smaller, more manageable components, thereby improving computational efficiency and decreasing response times. Serverless workflows equipped with GPUs have been proposed to speed up deep learning tasks by utilizing hardware acceleration facilities within serverless environments [6]. Advances in technology have enabled the deployment of AI workloads in a cost-effective way, while preserving the adaptability and expandability of serverless systems.

7. Use Cases and Real-World Applications

Several examples from real-world case studies demonstrate the practical effects of serverless computing across various fields. Scientific computing applications have utilized serverless execution models to process massive-scale workflows, thereby enhancing both cost-effectiveness and efficiency. Techniques for load balancing have been implemented to manage applications that can be run in parallel, resulting in effective resource allocation in unpredictable cloud environments. Serverless workflows have been utilized in multimedia processing to expedite video transcoding and image processing operations, exhibiting enhancements in scalability and performance. Companies have also examined hybrid serverless models, combining on-site computing resources with cloud-based function execution in order to meet regulatory and latency demands. The use of serverless architectures in enterprise AI, IoT, and large-scale data processing applications continues to solidify its position as a revolutionary cloud-native paradigm.

Cloud-Native Application Development in Enterprises

Companies are increasingly adopting cloud-native architectures in order to attain greater agility and scalability. Serverless computing serves a crucial function by enabling organizations to deploy containerized applications without overseeing infrastructure management. Serverless platforms combined with container-based architectures facilitate the effortless scaling of microservices while also minimizing costs. Studies have shown that combining serverless computing with container orchestration platforms optimizes resource distribution and processing effectiveness, allowing businesses to adapt their workload capacity as needed. Additionally, research has investigated event-driven processing architectures, which enable serverless functions to be activated by external occurrences, thereby facilitating the development of efficient and interactive cloud-based applications. Demonstrating efficient handling of event-driven data pipelines has shown significant advantages for large-scale business applications that rely heavily on data.

Serverless Computing for AI/ML Workloads

Serverless computing's integration with AI and ML tasks has seen substantial growth, driven primarily by its capacity for dynamic scaling and cost-effective execution processes [9]. Research has shown that machine learning models can be effectively deployed in serverless settings, where they can be loaded as needed and run across numerous function instances to process extensive datasets simultaneously. This approach minimizes the requirement for pre-provisioned infrastructure, thereby facilitating adaptive scaling in response to changing workload demands. Deep learning inference has been optimized by introducing GPU-enabled serverless workflows to enhance the computational performance of AI-driven applications. Serverless function chaining is being utilized to streamline AI processes, allowing for efficient data preparation, model prediction, and post-analysis tasks without necessitating the use of extended, dedicated computer systems. Serverless computing's expanding capabilities are increasingly applicable in both AI research and industry settings.

Serverless in Edge and IoT Computing

Serverless computing has had a notable impact on both edge and IoT computing by making it easier to process sensor data and do event-driven analytics in real-time [2]. By using serverless edge-based architectures, applications can run lightweight functions near the source of the data, thus reducing latency and the amount of bandwidth used. Real-time decision-making is especially important in IoT settings, making this feature particularly useful. Studies have investigated the implementation of serverless functions on a network of distributed edge nodes, revealing enhancements in resource usage and overall system reaction time. Frameworks for choreographing functions have been designed to manage the execution of intricate workflows in IoT applications, facilitating smooth data processing between cloud and edge settings. The deployment of serverless models in IoT systems has been enhanced by the capacity to dynamically assign compute resources at the edge without the need to maintain persistent infrastructure.

8. Future Trends and Research Directions

Enhancing Security and Privacy in Cloud-Native Environments

Cloud-native and serverless computing pose significant security risks due to the temporary nature of serverless functions and shared execution environments used by multiple tenants. Lack of direct control over infrastructure poses challenges concerning isolation at the function level, data protection, and access control. Initiatives are underway to reduce the vulnerability of cold start security risks, which occur when functions are created on an as-needed basis without the benefit of a persistent security framework. Research is being conducted into the use of hardware-based trusted execution environments (TEEs) for the purpose of offering enhanced function-level isolation [11]. Improvements are underway to prevent unauthorized use of serverless functions through enhanced access control and function verification, including mechanisms to authenticate legitimate function invocation. Researchers are also exploring secure data processing methods, such as homomorphic encryption and differential privacy [12], to preserve data confidentiality within serverless environments. The primary objective of these advancements is to strike a balance between adhering to regulatory requirements and preserving the flexibility and productivity of serverless architectures.

Open Challenges and Future Prospects

Serverless computing has shown considerable advantages, but several hurdles still need to be overcome for its widespread use to become a reality. Latency upon initial startup remains a significant performance obstacle, especially for applications that are highly sensitive to delays. Efforts are centered on predictive pre-heating methods and caching tactics to reduce the performance effects of initial system launches. The absence of standardization across serverless platforms creates interoperability issues, making it difficult to move functions between various cloud providers. There is a growing trend towards open-source serverless frameworks and cross-cloud function execution models to mitigate this fragmentation. A key area to investigate is cost minimization, given that serverless pricing models typically prove challenging to forecast for intricate workloads. Adaptive resource allocation and fine-grained billing systems aim to improve cost effectiveness. The integration of serverless computing and AI-driven workload orchestration in the future is anticipated to improve function execution even further, thereby making serverless a more viable option for diverse computing requirements.

9. Conclusion

Modern software development and deployment strategies have been revolutionized by cloud-native and serverless computing, which facilitates scalable, event-driven, and cost-efficient computing. Serverless architecture offers many benefits, but persistent difficulties require ongoing improvements in performance enhancement, security upgrades, and workload management. Serverless computing's evolution will continue to be influenced by advancements in AI-driven resource management, hybrid cloud adoption, and open standards, thereby maintaining its significance and effectiveness across numerous applications.

References

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

[2]. M. Malawski, A. Gajek, A. Zima, B. Balis, and K. Figiela, "Serverless execution of scientific workflows: Experiments with HyperFlow, AWS Lambda and Google Cloud functions," Future Generation Computer Systems, vol. 110, pp. 502–514, 2020.

[3]. S. Risco, G. Moltó, D. M. Naranjo, and I. Blanquer, "Serverless Workflows for Containerised Applications in the Cloud Continuum," Journal of Grid Computing, vol. 19, no. 3, p. 30, 2021.

[4]. V. Giménez-Alventosa, G. Moltó, and M. Caballer, "A framework and a performance assessment for serverless MapReduce on AWS Lambda," Future Generation Computer Systems, vol. 97, pp. 259–274, 2019.

[5]. S. Ristov, S. Pedratscher, and T. Fahringer, "AFCL: An abstract function choreography language for serverless workflow specification," Future Generation Computer Systems, vol. 114, pp. 368–382, 2021.

[6]. S. Risco and G. Moltó, "GPU-enabled serverless workflows for efficient multimedia processing," Applied Sciences, vol. 11, no. 4, p. 1438, 2021.

[7]. I. Baldini et al., "Serverless computing: Current trends and open problems," in Research Advances in Cloud Computing, Springer, Singapore, 2017, pp. 1–20.

[8]. V. Ishakian, V. Muthusamy, and A. Slominski, "Serving deep learning models in a serverless platform," in 2018 IEEE International Conference on Cloud Engineering (IC2E 2018), 2018, pp. 257–262.

[9]. M. Shahrad et al., "Serverless in the wild: Characterizing and optimizing the serverless workload at a large cloud provider," in 2020 USENIX Annual Technical Conference (USENIX ATC 2020), 2020, pp. 205–218.

[10]. A. Klimovic, C. Kozyrakis, and E. R. S. Vermel, "Pocket: Elastic ephemeral storage for serverless analytics," in USENIX Symposium on Operating Systems Design and Implementation (OSDI 2018), 2018, pp. 427–444.

[11]. Increment, "How to Leverage Confidential Computing to Protect Containerized Data," Increment, May 2020.

[12]. Michael Naehrig, Kristin Lauter, and Vinod Vaikuntanathan. 2011. Can homomorphic encryption be practical? In Proceedings of the 3rd ACM workshop on Cloud computing security workshop (CCSW '11). Association for Computing Machinery, New York, NY, USA, 113–124.