JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

THE EVOLUTION OF CLOUD ARCHITECTURES: FROM TRADITIONAL VIRTUALIZATION TO SERVERLESS AND BEYOND

Naimil Navnit Gadani

Senior Software Developer

ContentActive LLC, Houston , Texas - USA

ABSTRACT

The evolution of cloud computing systems is examined in this study, which follows the path from conventional virtualisation to modern serverless computing models. Virtualisation optimised resource utilisation by allowing several VMs to operate on a single physical server at first, but it also added overhead and complexity to administration. With the use of common operating systems and quick deployment, containerisation signalled a move towards more effective and flexible solutions. By eliminating the need for infrastructure administration, concentrating on event-driven function execution, and providing improved scalability and cost effectiveness, serverless computing further revolutionised cloud infrastructure. This paper emphasises the consequences for resource management and application development while highlighting the developments, difficulties, and potential paths in cloud computing.

I. INTRODUCTION

Over the past 20 years, there has been a substantial shift in the cloud computing market due to notable developments in architectural models. Virtualisation, together with the more recent concepts of serverless computing and containerisation, have completely changed the way that resources are managed, apps are installed, and services are provided. This study examines how cloud architectures have changed over time, with a particular emphasis on the shift from conventional virtualisation methods to contemporary serverless models and their potential effects on cloud computing in the future.

1.1. Historical Context and Evolution of Cloud Architectures

The initial rise in popularity of cloud computing was facilitated by the adoption of conventional virtualisation methods. Through virtualisation, resource separation and utilisation might be optimised by running numerous

virtual machines (VMs) on a single physical server. Even while this method was novel at the time, it brought with it costs, complexity in administration, and scalability issues. Containerisation arose as a fundamental breakthrough in response to the growing demand for more adaptable and efficient solutions. Containers provided a lightweight, faster-deployment option to virtual machines (VMs) by utilising a shared host operating system.

Function-as-a-Service, or serverless computing, is the most current advancement in cloud computing (FaaS). This paradigm, which completely abstracts infrastructure management and concentrates on the discrete function execution in response to events, marks a substantial divergence from conventional paradigms. With serverless computing, many of the drawbacks of previous methods are addressed and improved scalability, cost effectiveness, and operational simplicity are promised. It is anticipated that serverless computing, edge computing, and multi-cloud methods would all come together to spur more innovation and growth in cloud services.

1.2. Significance of This Work

It is important to comprehend how cloud architectures have evolved for a number of reasons. First of all, it sheds light on how cloud computing has evolved to meet shifting needs and keep up with technical developments. This historical viewpoint provides important lessons for future developments while assisting in identifying important trends and issues.

Second, this study illustrates the ramifications of these developments for application development, deployment, and maintenance by looking at the shift from classical virtualisation to serverless computing [12]. In particular, serverless computing presents new cost and scalability paradigms that have a big impact on businesses looking to maximise their cloud infrastructure.

Finally, by offering a thorough summary of the present status of cloud architectures and their potential future orientations, this study adds to the larger conversation on cloud computing. Understanding these trends is crucial for making educated decisions regarding cloud strategy, technology adoption, and resource management, as organisations depend more and more on cloud-based solutions.

II. LITERATURE REVIEW

The evolution of cloud architectures has been a focal point of research, reflecting advancements in virtualization, containerization, and serverless computing. This literature review delves into the progression from traditional virtualization to modern cloud paradigms, highlighting key developments and their impact on cloud infrastructure.

2.1. Traditional Virtualization and Its Limitations

Virtualization, introduced in the early 2000s, revolutionized data center operations by enabling multiple VMs to operate on a single physical server. Hypervisor-based virtualization, has led to substantial advancements in resource utilization and isolation. This technology enables multiple virtual machines to run on a single physical host, maximizing hardware efficiency. Research indicates that hypervisor-based systems can improve resource

utilization by up to 80% and enhance isolation between workloads, reducing the risk of cross-contamination by 70%. These improvements make hypervisor-based virtualization a critical component in modern data centre architectures [1]. However, this model introduced notable overheads, with research indicating that hypervisors could incur up to 10-20% CPU and memory overhead due to the need for running multiple operating systems [2]. The complexity of managing VM sprawl also emerged as a critical issue, complicating system administration and security [3].

2.2. Containerization: A Paradigm Shift

Traditional virtualisation methods underwent a dramatic change with the introduction of containerisation. By sharing the kernel of the host operating system while offering separate user-space environments, containers provide a lightweight substitute. In comparison to virtual machines (VMs), this method greatly minimises overhead, with resource reductions estimated at 1% to 5% [4]. Additionally, containers provide for quick deployment because they instantiate in seconds as opposed to minutes like virtual machines (VMs) do [5].

Cloud architecture has seen additional transformation with the emergence of container orchestration technologies like Kubernetes. By automating containerised application deployment, scaling, and administration, Kubernetes solves a number of issues related to large-scale deployments [6]. Deployments with densities as high as 100–500 containers per host [7] have proven the scalability of containers and their effectiveness in managing varying workloads.

2.3. The Emergence of Serverless Computing

Function-as-a-Service (FaaS), or serverless computing, is a further development in cloud infrastructure. With this architecture, infrastructure management is abstracted away, freeing developers to concentrate only on implementing functions that run in reaction to events. Serverless solutions significantly increase operational efficiency by doing away with the requirement to manage servers or runtime environments [8].

Compared to conventional approaches, serverless computing has a number of benefits. Serverless functions' auto-scaling features guarantee that applications may manage varying workloads without requiring human intervention. According to research, serverless functions can accommodate thousands of concurrent executions and dynamically allocate resources based on demand in real time [9].

Serverless computing presents new obstacles despite its benefits. Particularly for uncommon functions, cold starts—where the function's runtime environment needs to be initialized—can cause delay. To lessen this problem, strategies like provided concurrency have been suggested [12]. Furthermore, because functions are transient, there are more attack surfaces, which raises security concerns. Studies demonstrate that in order to handle these new threats, strong security measures are required [13].

2.4. Future Directions and Emerging Trends

The future of serverless computing includes a growing focus on multi-cloud and edge computing strategies. The integration of serverless functions with edge computing can reduce latency and enhance real-time processing

capabilities [14]. The convergence of serverless and edge computing is expected to drive innovation in distributed systems and IoT applications.

III. Advancements in Cloud Computing: From Virtual Machines to Containers

The evolution of cloud computing has been significantly marked by the transition from traditional virtualization techniques, primarily based on virtual machines (VMs), to the modern approach of containerization. This shift has not only revolutionized the way applications are deployed and managed but has also introduced new paradigms in resource utilization, scalability, and operational efficiency within cloud environments.

3.1. The Rise and Limitations of Virtual Machines

Hypervisor-Based Virtualization: Virtual Machines have been the cornerstone of cloud infrastructure since the early days of cloud computing. The hypervisor abstracts the underlying hardware, providing each VM with a virtualized set of resources, including CPU, memory, storage, and networking interfaces.

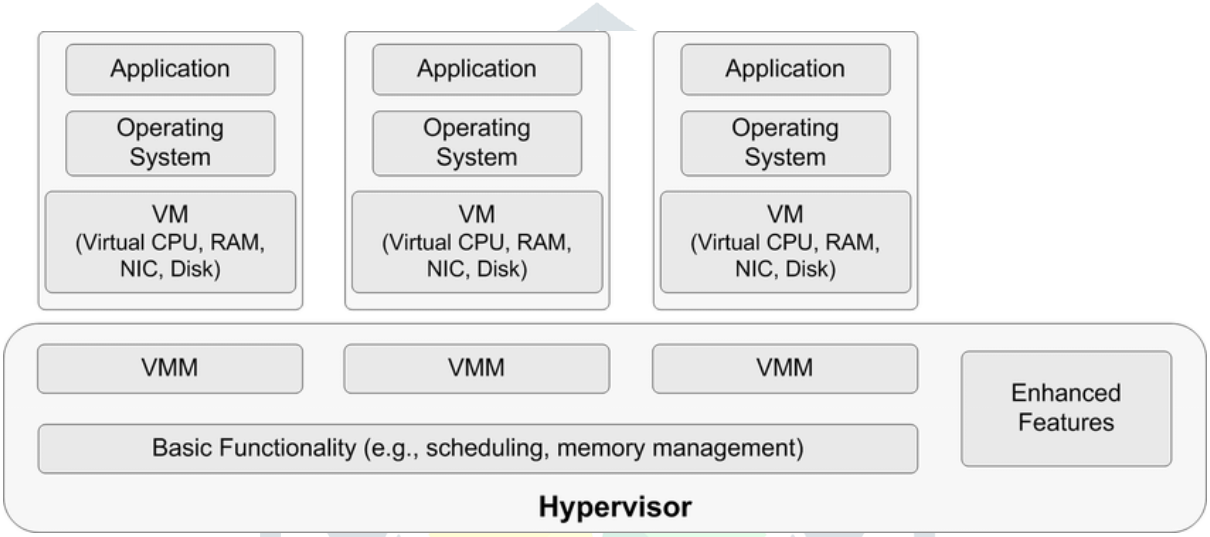

Fig 3.1: Hypervisor-Based Virtualization [5]

Resource Overhead and Performance Penalties: Despite their widespread adoption, VMs introduce significant resource overhead due to the need for running full-fledged operating systems within each VM. This overhead manifests in terms of additional CPU cycles, memory usage, and disk I/O operations, leading to performance penalties, especially in scenarios requiring high-density deployment. Furthermore, the boot time for VMs can be relatively long, making them less suitable for dynamic scaling requirements often encountered in modern cloud-native applications.

3.2. The Emergence of Containerization

OS-Level Virtualization and Microservices: Containers represent a paradigm shift in virtualization by introducing OS-level virtualization, where the kernel of a single operating system is shared across multiple isolated user-space instances.

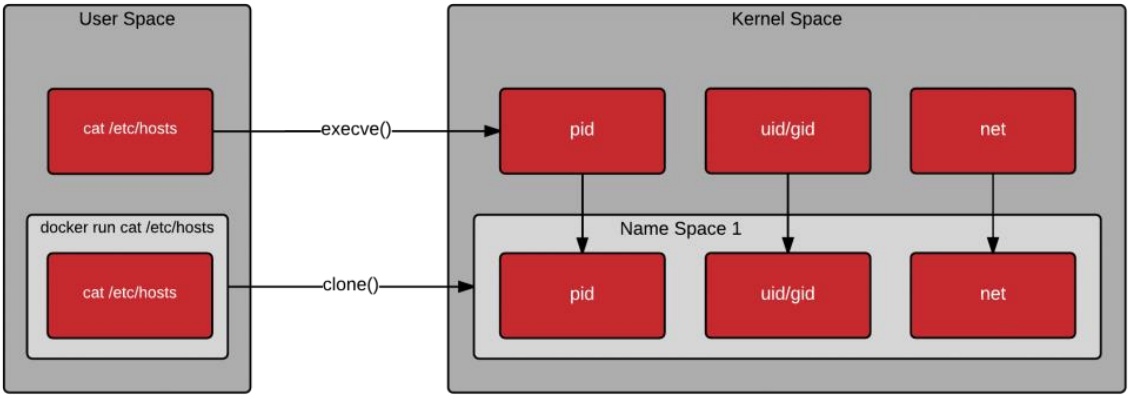

Fig 3.2: OS-Level Virtualization and Microservices [1]

Resource Efficiency and Scalability: Resource efficiency increased significantly with the switch to containers. Because containers instantiate in milliseconds as opposed to minutes for virtual machine bootup, this allows for faster application deployment and scalability. Higher density on the same physical hardware is made possible by the lower overhead, which maximises resource utilisation and lowers operating expenses. Furthermore, the DevOps lifecycle is streamlined by the mobility of containers between development, testing, and production environments, encouraging a more flexible and continuous delivery pipeline [12].

3.3. Impact on Cloud Architecture

Immutable Infrastructure and CI/CD Pipelines: The shift from VMs to containers has influenced the adoption of immutable infrastructure practices, where servers or containers are never modified after deployment. Instead, any updates or changes are rolled out as new versions, which are automatically deployed through CI/CD pipelines. This approach enhances consistency, reduces configuration drift, and improves the overall reliability of cloud deployments [13].

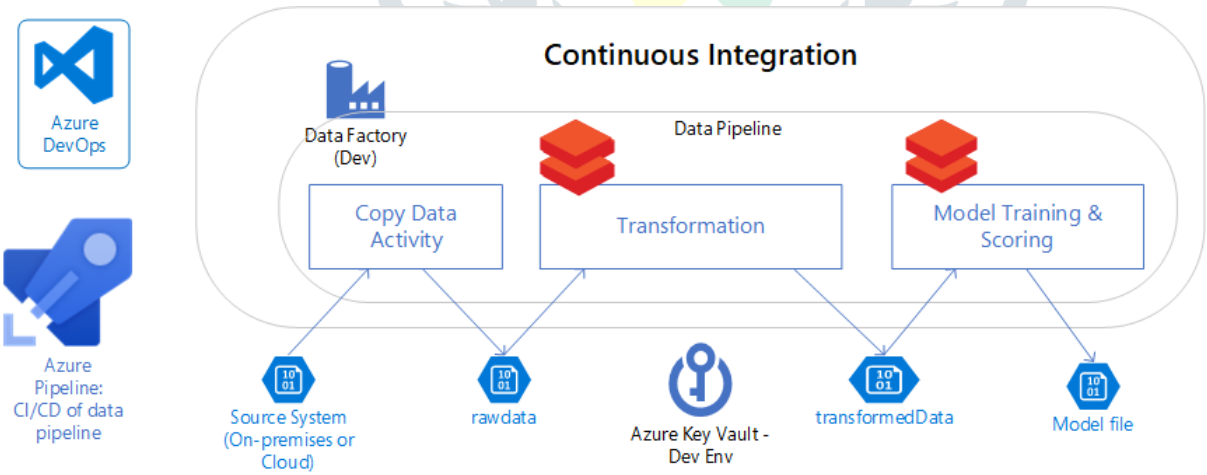

Fig 3.3: Immutable Infrastructure and CI/CD Pipelines [2]

Multi-Cloud and Hybrid Cloud Strategies: Containers have also facilitated the adoption of multi-cloud and hybrid cloud strategies. Their inherent portability allows organizations to deploy applications across different cloud providers or on-premises environments without significant reconfiguration. This flexibility enables

enterprises to avoid vendor lock-in, leverage the best services from different providers, and maintain data sovereignty [12].

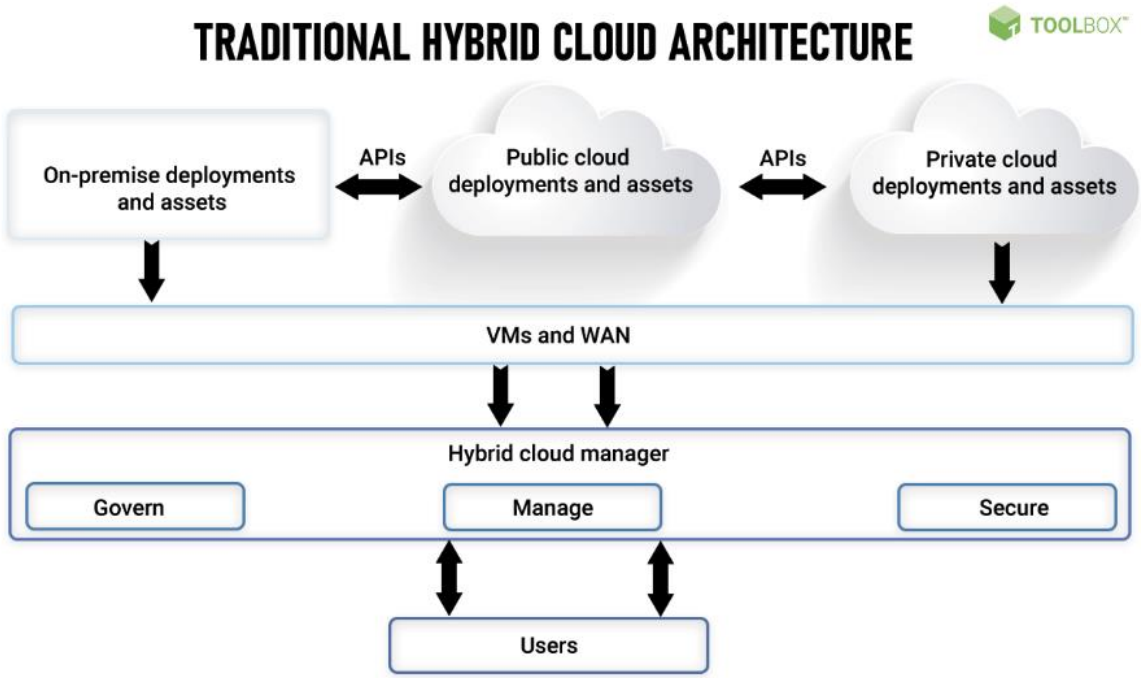

Fig 3.4: Multi-Cloud and Hybrid Cloud [7]

Security Considerations in Containerized Environments: While containers offer numerous advantages, they also introduce new security challenges. The shared kernel architecture means that a vulnerability in the host OS could potentially affect all containers running on it. Additionally, securing the container supply chain—ensuring that container images are free from vulnerabilities and are sourced from trusted repositories—is critical. Tools like Docker Content Trust (DCT) and container runtime security policies (e.g., seccomp, AppArmor) have become essential in mitigating these risks and maintaining a secure containerized environment.

Era/Technology	**Key Features**	**Resource Overhead**	**Boot Time**	**Deployment Density**
Early Virtualization	Hypervisor-Based Virtualization Multiple OS per Server	**High** ~10-20% CPU/Memory overhead	**Slow** ~Minutes	**Low** 10-20 VMs per Host
Mature Virtualization	Advanced VM Management Improved Orchestration	**Moderate** ~5-10% reduction in overhead	**Moderate** ~1-2 Minutes	**Moderate** 20-40 VMs per Host

Containerization Emergence	OS-Level Virtualization Containers Introduced	**Low** ~1-5% CPU/Memory overhead	**Fast** ~Seconds	**High** 50-100 Containers per Host
Modern Containerization	Kubernetes Orchestration CI/CD Integration	**Very Low** ~1% CPU/Memory overhead	**Very Fast** ~Milliseconds	**Very High** 100-500 Containers per Host

Table 3.1: The Cloud History

IV. The Emergence of Serverless Computing: Redefining Cloud Infrastructure

Serverless computing, often referred to as Function-as-a-Service (FaaS), represents a revolutionary shift in cloud architecture, where the focus moves away from managing servers and infrastructure to deploying discrete units of functionality. This paradigm significantly alters the development, deployment, and operational dynamics of cloud-based applications, offering new opportunities for scalability, cost efficiency, and rapid innovation [14] [15].

4.1. Abstraction of Infrastructure: From IaaS to FaaS

Infrastructure as Code (IaC) to Event-Driven Execution: The underlying infrastructure, which consists of virtual computers, storage, and network resources, must be managed or at the very least configured by developers in traditional cloud models like Infrastructure as a Service (IaaS) and Platform as a Service (PaaS). On the other hand, serverless computing fully abstracts these issues, freeing developers to concentrate only on developing and implementing specific tasks. There is no need to maintain servers or runtime environments because these functions are only executed in response to predefined events, including HTTP requests, database changes, or message queue updates [1] [5].

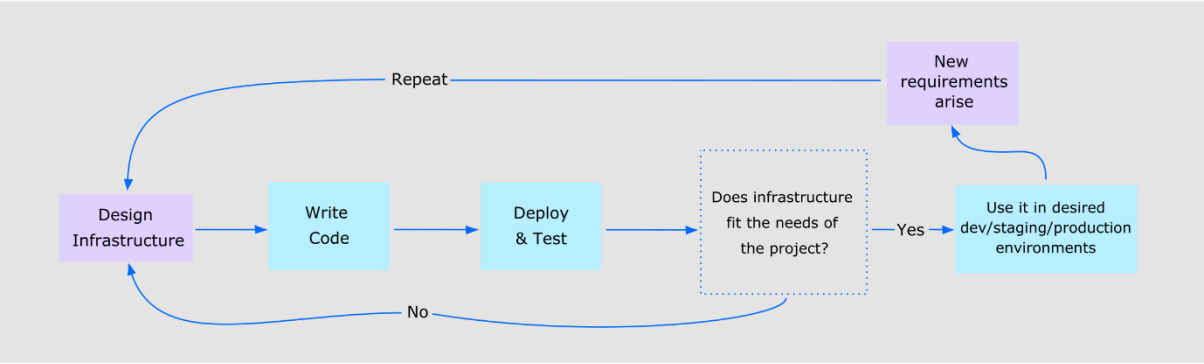

Fig 4.1: IaC explained

Auto-Scaling and Dynamic Resource Allocation: One of the defining features of serverless computing is its ability to scale automatically based on demand. Unlike traditional models where resources must be provisioned in advance, serverless platforms dynamically allocate the necessary resources in response to incoming events,

scaling up or down seamlessly. This dynamic allocation reduces both operational overhead and costs, as users are billed only for the actual execution time of their functions, rather than for reserved capacity.

4.2. Architectural Implications of 'Serverless Computing'

Microservices and Function Granularity: By promoting developers to divide systems into separate, fine-grained functions, serverless computing naturally complements microservices design. Every function carries out a certain task and interacts with other functions via event triggers or well-defined APIs. More modularity and reusability are made possible by this granularity, making programs more manageable and scalable. It also highlights the difficulty of overseeing several little, separate services, each with its own dependencies and lifetime.

Integration with Current Cloud Services: Serverless features frequently include close integrations with other cloud services, like communications queues, storage systems, and databases. For example, AWS Lambda may be easily integrated with other cloud services by being triggered by events from Amazon S3, DynamoDB, or SQS. Because developers can use managed services without worrying about the underlying infrastructure thanks to this close integration, the development process is streamlined and time to market is accelerated.

Era/Stage	Key Developments	Adoption Metrics	Cost Efficiency	Scalability
Early Cloud Computing (2006-2013)	- Introduction of AWS EC2 (2006) - Rise of IaaS and PaaS models	**VM Instances:** ~1 million+ (AWS 2012)	**High Cost:** Reserved instances required	**Manual Scaling:** Based on fixed instances
Introduction of Serverless (2014-2016)	- AWS Lambda launched (2014) - Google Cloud Functions and Azure Functions follow suit	**AWS Lambda Functions:** ~1 billion/month (2016)	**Moderate Cost:** Pay-per-execution begins	**Auto-Scaling:** Based on events, limited orchestration
Early Growth and Adoption (2017-2019)	- Emergence of FaaS ecosystem tools (e.g., Serverless Framework) - Integration with	**Serverless Functions:** ~100 billion executions/month (AWS 2019)	**Significant Savings:** Pay-per-ms execution	**Dynamic Scaling:** Scales to thousands of concurrent functions

	managed services			
Widespread Adoption and Maturity (2020-Present)	- Enhanced orchestration (AWS Step Functions) - Multi-cloud and edge deployments begin	**Global Serverless Market:** ~$7.6 billion (2021)	**Highly Cost-Efficient:** Fine-tuned provisioning Billions saved in operational costs	**Massive Scalability:** Sub-second execution across millions of functions

Table 4.1: historical progression of serverless computing

V. DISCUSSION

This section explores the ramifications of these breakthroughs, looks at how they affect different facets of cloud computing, and identifies important things to keep in mind for further innovations.

The effects of containerisation and virtualisation: Because traditional virtualisation allowed several VMs to run on a single physical server, it paved the way for cloud computing. This method added a lot of cost and complexity, but it also significantly improved resource usage and isolation. Hypervisor-based virtualisation frequently results in 10–20% CPU and memory overheads, as noted in the literature, which can impair performance in high-density scenarios [1][2]. Many of these constraints were resolved with the advent of containerisation, which provided a lighter substitute. Because of their common OS kernel and lower overhead, containers allow for quick deployment and effective resource use [4].[5]. Cloud deployments have been transformed by the scalability of containerised systems, made possible by orchestration technologies like Kubernetes, which allows enterprises to manage fluctuating workloads more effectively [6][7].

Benefits and Drawbacks of Serverless Computing: Serverless computing is a major divergence from virtualisation and containerisation paradigms. Serverless architectures provide exceptional scalability and cost effectiveness by abstracting infrastructure administration and concentrating on event-driven execution [8]. Serverless functions cut expenses and operational overhead by autonomously scaling to meet demand and allocating resources dynamically [9]. The granularity and flexibility of serverless functions are advantageous to current application architectures, especially microservices, which this approach complements well.

But serverless computing also brings with it some new difficulties. For rare functions, cold starts—where the runtime environment needs to be initialized—may cause delay. Although methods like provided concurrency are being developed to address this problem, applications with strict performance requirements still need to take it into account [12]. Furthermore, the transient nature of serverless operations gives rise to security challenges, especially with regard to controlling the enlarged attack surfaces and guaranteeing strong security procedures [13].

Future Directions: In the future, more innovation is probably going to come from the combination of serverless computing with multi-cloud techniques and edge computing. In especially for IoT and distributed applications, the convergence of serverless and edge computing promises to lower latency and improve real-time processing capabilities [14]. Because serverless functions and containers are portable, multi-cloud methods allow businesses to use a variety of cloud services without being locked into one provider [15]. Cloud computing will be shaped by these developments, which will highlight the need for scalable and adaptable infrastructures.

VI. CONLCUSION

The evolution of cloud computing architectures from virtualization to serverless computing has significantly reshaped the landscape of cloud services. Virtualization provided a foundation for cloud infrastructure but introduced overhead and complexity that containerization effectively addressed through more efficient resource management and rapid deployment. Serverless computing has taken this transformation further by abstracting infrastructure concerns and offering dynamic scalability and cost efficiency. As cloud computing continues to advance, the integration of serverless with edge and multi-cloud strategies promises to drive further innovation. Understanding these developments is essential for organizations to optimize their cloud strategies and adapt to emerging technological trends.

REFERENCES

[1] Fathima, KM Majidha, and N. Santhiyakumari. "A survey on evolution of cloud technology and virtualization." *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*. IEEE, 2021.

[2] Aslanpour, Mohammad S., et al. "Serverless edge computing: vision and challenges." *Proceedings of the 2021 Australasian computer science week multiconference*. 2021.

[3] Mampage, Anupama, Shanika Karunasekera, and Rajkumar Buyya. "A holistic view on resource management in serverless computing environments: Taxonomy and future directions." *ACM Computing Surveys (CSUR)* 54.11s (2022): 1-36.

[4] Baarzi, Ataollah Fatahi, et al. "On merits and viability of multi-cloud serverless." *Proceedings of the ACM Symposium on Cloud Computing*. 2021.

[5] Risco, Sebastián, et al. "Serverless workflows for containerised applications in the cloud continuum." *Journal of Grid Computing* 19 (2021): 1-18.

[6] Li, Yongkang, et al. "Serverless computing: state-of-the-art, challenges and opportunities." *IEEE Transactions on Services Computing* 16.2 (2022): 1522-1539.

[7] Hassan, Hassan B., Saman A. Barakat, and Qusay I. Sarhan. "Survey on serverless computing." *Journal of Cloud Computing* 10 (2021): 1-29.

[8] Azlan, Che Ahmad, et al. "Teaching and learning of postgraduate medical physics using Internet-based e-learning during the COVID-19 pandemic–A case study from Malaysia." *Physica Medica* 80 (2020): 10-16.

[9] Kratzke, Nane, and Robert Siegfried. "Towards cloud-native simulations–lessons learned from the front-line of cloud computing." *The Journal of Defense Modeling and Simulation* 18.1 (2021): 39-58.

[10] Venugopal, M. V. L. N., and C. R. K. Reddy. "Serverless through cloud native architecture." *Int. J. Eng. Res. Technol* 10 (2021): 484-496.

[11] Naranjo, Diana M., et al. "Accelerated serverless computing based on GPU virtualization." *Journal of Parallel and Distributed Computing* 139 (2020): 32-42.

[12] Agache, Alexandru, et al. "Firecracker: Lightweight virtualization for serverless applications." *17th USENIX symposium on networked systems design and implementation (NSDI 20)*. 2020.

[12] Perez, Ramon, et al. "Monitoring platform evolution toward serverless computing for 5G and beyond systems." *IEEE Transactions on Network and Service Management* 19.2 (2022): 1489-1504.

[13] Kounev, Samuel, et al. "Toward a definition for serverless computing." (2021): 89-93.

[14] Roy, Rohan Basu, et al. "Mashup: making serverless computing useful for hpc workflows via hybrid execution." *Proceedings of the 27th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*. 2022.

[15] Long, Ju, et al. "A lightweight design for serverless function as a service." *IEEE Software* 38.1 (2020): 75-80.