Available online at www.worldscientificnews.com

World Scientific News

An International Scientific Journal

WSN 193(1) (2024) 1-45 EISSN 2392-2192

Securing the Server-less Frontier: Challenges and Innovative Solutions in Network Security for Server-less Computing

Md. Abu Imran Mallick[1,*] and **Rishab Nath**[2]

[1] Department of Cyber Forensic and Information Security, Behala Government Polytechnic,
756 Upendra Nath Banerjee Road, Parnasree, Behala, Kolkata, West Bengal - 700060, India

[2] Department of Bio-science, Ananda Mohan College,
102/1, Raja Rammohan Sarani, Kolkata - 700009, India

*E-mail address: imranmallick708@gmail.com

ABSTRACT

Server less computing has gained significant popularity due to its flexibility, scalability, and cost-effectiveness. However, the increased use of server less platforms has brought new challenges, particularly in the realm of network security. This review paper aims to explore the challenges that arise in ensuring network security within server less computing environments and to propose potential solutions to mitigate these challenges. The first challenge addressed in this review is the lack of visibility and control in server less environments. Traditional security tools and methodologies are not always applicable to server less architectures, resulting in limited visibility into the network and the services hosted. This lack of visibility can lead to vulnerabilities and potential security breaches. To address this challenge, the paper proposes the adoption of specialized security tools designed specifically for server less environments, as well as the implementation of comprehensive monitoring and logging mechanisms to gain insight into network activities. Another significant challenge is the secure communication between server less functions and external services. Server less applications often rely on a wide array of external APIs and services, which can introduce security risks if not properly managed. The paper explores the need for secure communication protocols, including the use of encryption and authentication mechanisms, to protect data in transit and ensure the integrity and confidentiality of network communications. Furthermore, the dynamic nature of server less environments introduces challenges related to secure code deployment and runtime security. Traditional security measures such

(Received 15 March 2024; Accepted 12 April 2024; Date of Publication 15 April 2024)

as network firewalls and intrusion detection systems may not adequately protect server less functions, which are ephemeral and often have short lifespans. This paper discusses the importance of implementing robust security practices during the development and deployment stages of server less applications, including secure coding techniques, vulnerability scanning, and runtime security controls. Additionally, the review addresses the challenge of securing sensitive data in server less environments. Data protection is a critical aspect of network security, and server less applications must adhere to stringent data privacy regulations. The paper explores the best practices for implementing data encryption, tokenization, and access control mechanisms to safeguard sensitive information within server less architectures. This review paper presents an in-depth analysis of the challenges posed by network security in server less computing and proposes a range of solutions to address these challenges. By understanding the unique security considerations of server less environments and implementing the recommended solutions, organizations can effectively enhance the network security posture of their server less applications. This paper aims to serve as a valuable resource for security professionals, researchers, and practitioners seeking to navigate the evolving landscape of server less computing while ensuring robust network security.

***Keywords*:** Challenges, Data protection, Encryption, Network security, Server-less computing, Solutions, Security tools

1. INTRODUCTION

Server-less computing is a cloud computing model where the cloud provider manages the infrastructure, and users are billed based on actual usage rather than pre-allocated resources (Mohd Nazir. 2011; Baldini et al. 2017; Gowri et al. 2021). Servers are still involved, but users don't need to worry about their management, allowing for a more scalable and cost-efficient approach to deploying applications (Reddy et al. 2011). The management of servers is abstracted away from the user, so they don't need to worry about provisioning, scaling, or maintaining servers (Chase et al. 2001; Hamilton. 2007). However, servers are still utilized by the cloud provider to run the functions or services requested by the user (Christensen. 2009; Malik et al. 2018). The term "serverless" refers to the fact that users don't need to directly manage servers; they only focus on deploying their code or functions, and the cloud provider handles the rest (Varia. 2010; Eismann et al. 2020). So, while the user doesn't need to fix the number of servers, the cloud provider manages the infrastructure behind the scenes to accommodate the workload efficiently (Armbrust et al. 2009; Manvi and Shyam. 2014; Zanella et al. 2018).

Serverless computing offers numerous benefits such as scalability, improved processing time, flexibility, cost savings, and faster time to market (Patros et al. 2021; Shaflei et al. 2022). However, it's essential to recognize that along with these benefits, there are also challenges that organizations must address to fully leverage serverless computing (Van et al. 2018; Jonas et al. 2019). These challenges may include managing dependencies between functions, dealing with vendor lock-in, ensuring security and compliance, monitoring and debugging in a distributed environment, and optimizing costs effectively (Castro et al. 2023).

Network security is a major concern in serverless computing, encompassing the security of connected devices, users, and data transmission (Nastic and Dustdar. 2018; Candel et al. 2022). With serverless architectures relying heavily on network communication between various components, ensuring the confidentiality, integrity, and availability of data becomes paramount (Cassel et al. 2022; Ahmad et al. 2024). Challenges may include securing communication channels, protecting against malicious actors, implementing access controls, encrypting sensitive data, and maintaining compliance with regulatory requirements (Naranjo Rico et al. 2018; Tabrizchi and Kuchaki Rafsanjani. 2020).

Our review study aims to provide insights into the challenges surrounding network security in serverless computing and propose solutions to address them. Limiting access to the network is indeed a fundamental step in mitigating security risks and protecting against unauthorized access (Glaessner et al. 2002; Pfleeger and Pfleeger. 2012; Kitchin and Dodge. 2020). Implementing stringent access controls, such as role-based access control (RBAC), network segmentation, and least privilege principles, can help fortify the serverless environment against potential threats (Pal. 2022; Olaoye and Luz. 2024). Additionally, leveraging encryption techniques, implementing robust authentication mechanisms like multi-factor authentication (MFA), and regularly updating security policies and protocols can further enhance network security in serverless computing (Haber et al. 2022; Ali. 2023).

Implementing security systems like two-factor authentication (2FA) adds an extra layer of protection to serverless computing environments (Costa and Hodun. 2021; Catalfamo. 2023). With 2FA, even if an unauthorized individual or device manages to obtain login credentials, they would still need an additional verification method, such as a one-time password sent to a registered mobile device, to gain access (Mannan and Oorschot. 2011; Aravindhan and Karthiga. 2013; Chowhan and Tanwar. 2019). This significantly reduces the risk of unauthorized access and enhances the overall security posture of the network, safeguarding personal and sensitive information from potential threats (Farahmand et al. 2005; Mughal. 2018).

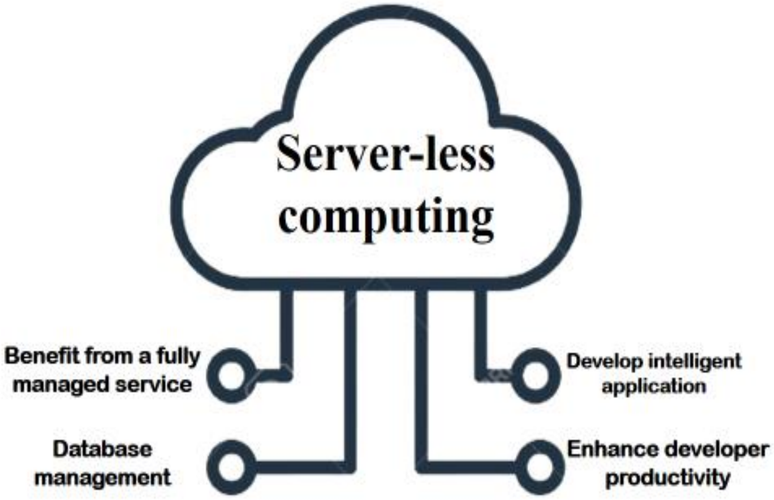

Server-less computing
Benefit from a fully managed service
Develop intelligent application
Database management
Enhance developer productivity

Figure 1. Server-less Computing

2. PROBLEM DEFINITION

The problem at hand revolves around ensuring network security within serverless computing environments (Jangda et al. 2019; Sankaran et al. 2020). Specifically, the challenge is to protect connected devices, users, and data transmission against unauthorized access and malicious threats (Samaila et al. 2018). This includes addressing vulnerabilities in communication channels, implementing access controls, encrypting sensitive data, and maintaining compliance with regulatory requirements (Yaqoob et al. 2019; Kumar and Goyal. 2019). The goal is to develop effective solutions that mitigate these security risks and safeguard the integrity, confidentiality, and availability of data in serverless deployments. The problem statement highlights the significance of serverless computing within cloud infrastructure due to its reliability and cost-effectiveness, particularly for large-scale projects (Jonas et al. 2019; Christidis et al. 2020).

However, the main challenge lies in ensuring robust network security within serverless environments. These challenges include protecting against unauthorized access, securing data transmission, and addressing vulnerabilities in network infrastructure. They can solve this problem to reduce their challenges.

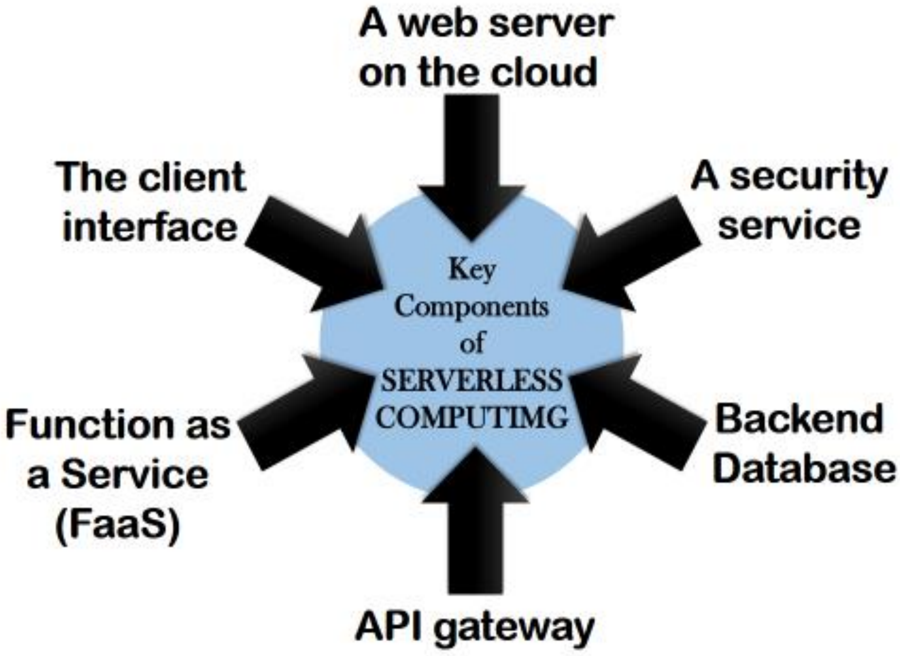

A web server on the cloud
The client interface
A security service
Key Components of SERVERLESS COMPUTIMG
Function as a Service (FaaS)
Backend Database
API gateway

Figure 2. Server-less computing architecture in cloud network

2. 1. Limited Visibility and Control

One of the significant challenges in serverless computing is limited visibility and control over the underlying infrastructure (Kelly et al. 2020; Baresi and Quattrocchi. 2021). Unlike traditional environments where users have direct access to servers and resources, serverless platforms abstract away much of the infrastructure, making it challenging to monitor and manage (Zhang et al. 2021; Javed et al. 2022). This lack of visibility can hinder security monitoring, threat detection, and performance optimization efforts. Additionally, limited control over security configurations and policies may lead to compliance issues and increased vulnerability to cyber threats (Pearce et al. 2013; Sobb et al. 2020; Vielberth et al. 2020). Loss of control and limited visibility are persistent challenges in serverless computing (Alpernas et al. 2018; Aditya et al. 2019). With serverless architectures abstracting away much of the underlying infrastructure, users often have minimal control over the execution environment and limited visibility into how their code is executed (Shahrad et al. 2019; Mampage et al. 2022).

This can make it difficult to monitor performance, debug issues, and enforce security policies effectively (Basak. Et al. 2016). Overcoming these challenges requires implementing comprehensive monitoring and logging solutions, adopting robust security measures, and collaborating closely with cloud service providers to improve transparency and control over serverless deployments (Ouyang et al. 2023). The shift in security paradigms from traditional approaches, where administrators have direct access to systems, to the challenges posed by serverless computing's lack of insight and traditional networking controls (Olaoye and Luz. 2024). In serverless environments, unauthorized access can occur due to limited visibility and control over the underlying infrastructure (Podjarny and Tal. 2019; Gjerdrum. 2020). These challenges underscore the importance of implementing robust monitoring systems and enhancing security measures to mitigate risks effectively.

2. 2. Insecure Dependencies

Another critical challenge in serverless computing is insecure dependencies (Mateus-Coelho and Cruz-Cunha. 2022). With serverless architectures, applications often rely on various third-party services, libraries, and APIs to perform specific functions (Chopra and Singh. 2021; Hasan et al. 2021). However, these dependencies may introduce vulnerabilities, such as outdated software versions, insecure configurations, or potential exploits (Candel et al. 2022). As a result, attackers can exploit these vulnerabilities to compromise the security of the entire application or access sensitive data (Singaravelu et al. 2006). To address insecure dependencies, organizations must regularly audit and update dependencies, enforce strict security policies, conduct thorough vulnerability assessments, and implement measures such as code signing and dependency integrity checks. The challenge of insecure dependencies in serverless applications, emphasizing the potential pathways they create for unauthorized access to networks and data (Candel et al. 2022). To address this challenge, organizations must implement effective measures to control and secure third-party libraries. This includes continuous monitoring and updating of dependencies to patch known vulnerabilities and mitigate security threats effectively. By proactively managing dependencies and implementing stringent security measures, organizations can reduce the risk of unauthorized access and enhance the overall security posture of their serverless applications.

2. 3. Cold Start Attacks

Another significant challenge in serverless computing is cold start attacks. Cold start refers to the delay experienced when a serverless function is invoked for the first time or after being idle for a certain period (Vahidinia et al. 2020; Ristov et al. 2022). During this time, the cloud provider needs to allocate resources and initialize the function, resulting in increased latency (Gouareb et al. 2018). Attackers can exploit this delay by repeatedly invoking functions to cause disruption or degrade performance, leading to denial-of-service (DoS) attacks (Aaad et al. 2008; Pascoal et al. 2020). To mitigate cold start attacks, organizations can implement strategies such as pre-warming functions, optimizing code for faster execution, and leveraging caching mechanisms to reduce latency. Additionally, implementing rate limiting and monitoring for anomalous behavior can help detect and mitigate suspicious activity associated with cold start attacks. The challenge of cold start in serverless computing, which poses security concerns due to the window of vulnerability it creates for attackers to gain access to data. Traditional security measures may not effectively address this challenge, underscoring the need for innovative solutions. To mitigate the risks associated with cold start attacks, organizations must develop strategies to reduce cold start times without sacrificing security. Implementing security measures directly within serverless applications can help reduce the impact of these risks. By overcoming these challenges and implementing solutions to minimize external access during the cold start phase, organizations can enhance the reliability and security of their serverless computing systems.

2. 4. Data Privacy and Resilience:

Data privacy and resilience are crucial aspects of serverless computing, and they present significant challenges for organizations. Data Privacy With serverless computing, data is often processed and stored in distributed environments, raising concerns about data privacy and compliance with regulations such as GDPR or HIPAA (Sing and Kolluri. 2021). Ensuring that sensitive data is adequately protected, encrypted both in transit and at rest, and accessed only by authorized users or functions is essential (Vegesna. 2019). Organizations must also implement robust access controls, audit trails, and data anonymization techniques to mitigate privacy risks effectively (Rubinstein and Hartzog. 2016). Serverless applications are distributed across multiple services and data centers, increasing their complexity and susceptibility to failures (Wen et al. 2023).

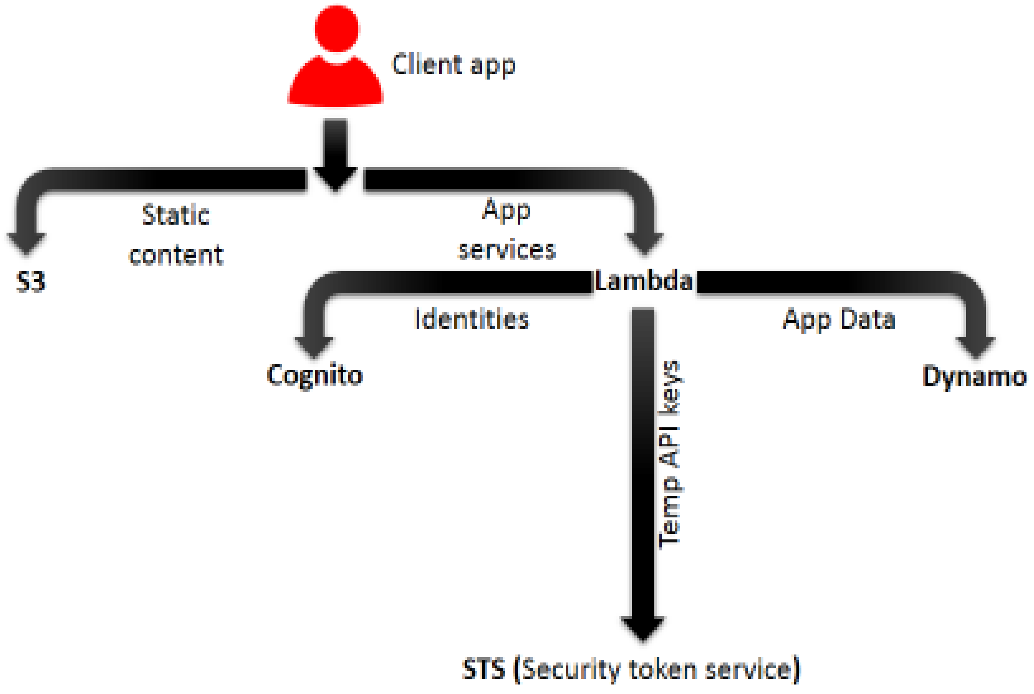


Figure 3. Server-less Security in Cloud Network

Ensuring resilience in serverless computing involves designing applications with fault tolerance in mind, implementing automatic scaling mechanisms, and leveraging redundant storage and compute resources (Solaiman. 2023). Additionally, organizations must conduct thorough testing, implement disaster recovery plans, and monitor performance metrics to detect and mitigate issues promptly (Lannurien et al. 2023). The significant challenge of maintaining data privacy and reliability in serverless computing environments, particularly when processing sensitive data. The risk of data loss or corruption can deter organizations from fully leveraging serverless systems (Chostak. 2020; Gill et al. 2022). To address this challenge, organizations must adopt a nuanced approach to data management, incorporating strong encryption, access controls, and robust security measures to safeguard sensitive data. Additionally, implementing effective backup strategies and disaster recovery plans can help mitigate the risk of data loss or corruption (Zobaed and Salehi. 2023). Striking a balance between data privacy and the benefits of serverless computing is essential, requiring careful consideration of security measures and compliance requirements to ensure the integrity and confidentiality of data while harnessing the advantages of serverless technology (Rawal et al. 2023; Zobaed and Salehi. 2023).

2. 5. Injection Attacks and Isolation

Injection attacks and isolation are significant challenges in serverless computing environments (Dutta et al. 2020). Injection Attacks just like in traditional environments, serverless applications are susceptible to injection attacks such as SQL injection, NoSQL injection, and command injection (Rinta-Jaskari. 2021; Pusuluri. 2022). Attackers may exploit vulnerabilities in input validation or inadequate security configurations to execute malicious code or gain unauthorized access to data (Papp et al. 2015; Aslan et al. 2023). To mitigate injection attacks, organizations must implement robust input validation, parameterized queries, and secure coding practices (Mitropoulos and Spinellis. 2017). Additionally, leveraging security mechanisms provided by cloud service providers, such as Web Application Firewall (WAF) and Runtime Application Self-Protection (RASP), can help detect and prevent injection attacks in serverless environments (Qazi. 2022). Serverless platforms rely on shared infrastructure and multi-tenant environments, raising concerns about isolation between functions and potential security risks (Sabbioni. 2023). Inadequate isolation may allow attackers to execute unauthorized actions or access sensitive data belonging to other users or functions (Ullah et al. 2018). To address isolation challenges, cloud service providers implement strong isolation mechanisms, such as containerization and virtualization, to ensure that each function operates within its own secure environment. Organizations should also implement least privilege principles, enforce strict access controls, and regularly audit permissions to mitigate the risk of unauthorized access or data leakage due to insufficient isolation (Silowash et al. 2012; Patwary et al. 2020). The isolation challenges inherent in serverless computing environments, particularly the potential for injection attacks due to shared runtime systems (Agache et al. 2020). These challenges can indeed provide attackers with opportunities to gain unauthorized access to sensitive data or execute malicious code (Morrow. 2012; Poeplau et al. 2014). To mitigate these risks, organizations must implement robust runtime protection mechanisms and isolation strategies (Sailer et al. 2005). This includes leveraging security features provided by cloud service providers, implementing strong access controls, and continuously monitoring for suspicious activity to detect and respond to attacks early (Patel et al. 2013). Balancing security requirements with the efficiency and scalability benefits of serverless architectures presents a significant challenge (Garcia-Lopez et al. 2019). However, organizations can achieve this balance by adopting a multi-layered security approach, incorporating proactive security measures into the development lifecycle, and continuously assessing and updating security controls. By prioritizing security and investing in comprehensive security measures, organizations can create a safe and reliable serverless computing environment, ensuring the integrity and availability of their applications and data.

2. 6. Resource Exhaustion and Denial-of-Service (DoS)

Resource exhaustion and denial-of-service (DoS) attacks are critical challenges in serverless computing environments (Shen et al. 2022). Serverless platforms allocate resources dynamically based on demand (Enes et al. 2020). However, malicious actors can exploit this flexibility by triggering numerous function invocations or executing computationally intensive tasks, leading to resource exhaustion (Papadopoulos et al. 2018). This can result in degraded performance or unavailability of services for legitimate users. To mitigate resource exhaustion attacks, organizations must implement rate limiting, concurrency controls, and auto-scaling policies to ensure fair resource allocation and prevent excessive usage by malicious actors (Kelly. 2023). Similar to traditional environments, serverless applications are vulnerable to DoS attacks, where attackers overwhelm the system with a flood of requests, causing disruption or downtime (Haber et al. 2022).

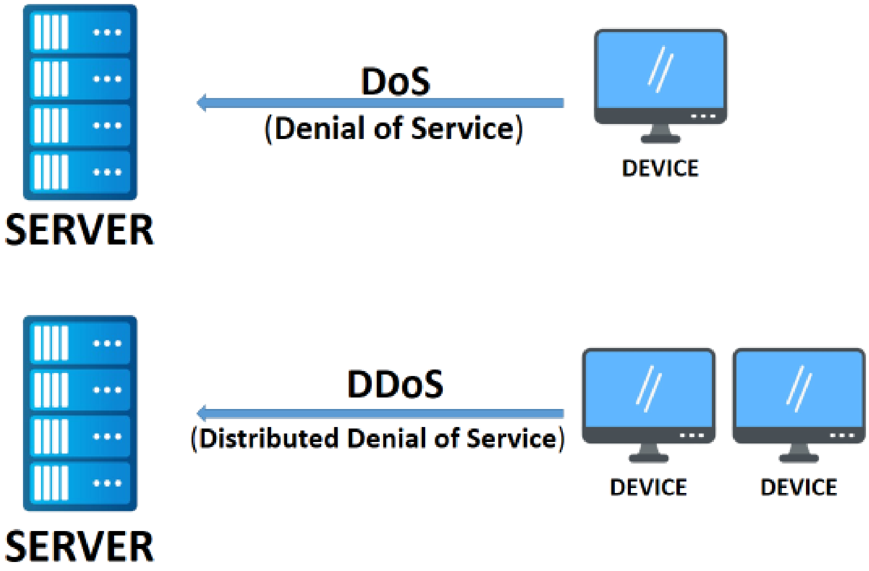


Figure 4. Denial-of-service (DoS) attack

Attackers may target specific functions or APIs, exploiting vulnerabilities in input validation or resource management to exhaust system resources (Ariffin et al. 2020). To mitigate DoS attacks, organizations must implement robust throttling mechanisms, request validation, and traffic shaping to detect and mitigate abnormal traffic patterns effectively (Prasad et al. 2014; Rios et al. 2022). Additionally, leveraging distributed denial-of-service (DDoS) protection services provided by cloud service providers can help mitigate large-scale attacks and ensure service availability (Bonguet and Bellaiche. 2017). The challenges related to resource exhaustion and denial-of-service (DoS) attacks in serverless computing environments, particularly due to the dynamic scaling of resources (Li et al. 2022). Malicious actors may attempt to overwhelm or manipulate the serverless system, leading to degraded performance or service unavailability (Zheng. 2020; Vs et al. 2023). To address these challenges, organizations must implement proactive measures such as effective rate limiting and strategic utilization of auto-scaling features. By optimizing resource allocation and enhancing responsiveness, organizations can defend their serverless systems against unauthorized access and maintain strong performance, even in the face of malicious attempts by attackers to exploit the dynamic scaling model of serverless computing.

✓ Introduction to Serverless Computing

Serverless computing is a cloud computing model where cloud providers manage the underlying infrastructure and dynamically allocate resources to execute code in response to events or requests (McGrath and Brenner. 2017). Unlike traditional server-based architectures, serverless computing abstracts away the complexity of managing servers, allowing developers to focus solely on writing and deploying code (Aditya et al. 2019; Mondal et al. 2022). In this model, applications are broken down into smaller, independent functions that are triggered by specific events, such as HTTP requests, database changes, or file uploads (Mondal et al. 2022; Mallick and Nath. 2024). One of the key features of serverless computing is its scalability (Jangda et al. 2019). With serverless architectures, resources are automatically scaled up or down based on demand, allowing applications to handle varying workloads efficiently without the need for manual intervention (Lynn et al. 2017)7y32. This elasticity enables organizations to optimize costs by only paying for the resources consumed during execution, rather than maintaining idle servers (Genaud and Gossa. 2011). Additionally, serverless computing promotes agility and rapid development cycles (Makani. 2023). Developers can quickly deploy code updates or new features without worrying about provisioning or managing infrastructure (Makani. 2023). This agility accelerates time-to-market and facilitates iterative development practices, enabling organizations to respond swiftly to changing business requirements (Makani. 2023).

The application of serverless computing in the context of the Internet of Things (IoT) (Cassel et al. 2022). By leveraging serverless computing, IoT solutions can seamlessly operate across cloud, fog, and edge layers (Cassel. 2022). This approach allows for the integration of critical functions at the fog and edge layers to benefit from low-latency responses, while heavier processing tasks can be offloaded to the cloud to handle large volumes of data generated by IoT sensors (Cicconetti et al. 2020). Serverless computing’s pay-as-you-go model aligns well with the variable workloads and resource requirements typical in IoT deployments (Lannurien et al. 2023). By dynamically allocating resources based on demand, serverless computing enables efficient utilization of computing resources across different layers of the IoT architecture. This flexibility and scalability contribute to the optimization of IoT solutions, allowing organizations to effectively manage data processing and latency requirements while minimizing costs (Tolosana-Calasanz. 2021).

Overall, the integration of serverless computing into IoT architectures offers significant advantages in terms of scalability, flexibility, and cost-effectiveness, enabling organizations to build robust and efficient IoT solutions that meet the demands of diverse use cases and deployment environments (Patros et al. 2021).

Serverless computing is indeed gaining popularity due to its simplicity of management and lightweight nature (Barcelona-Pons et al. 2022). By reducing the granularity of the computing unit to individual functions, serverless computing allows users to focus solely on writing and deploying code without worrying about underlying infrastructure management, such as provisioning servers or scaling resources (Shaflei et al. 2022). This abstraction of infrastructure complexities enables developers to concentrate on building applications and services, rather than dealing with scheduling or management tasks typically handled by the platform provider (Hilley. 2009).

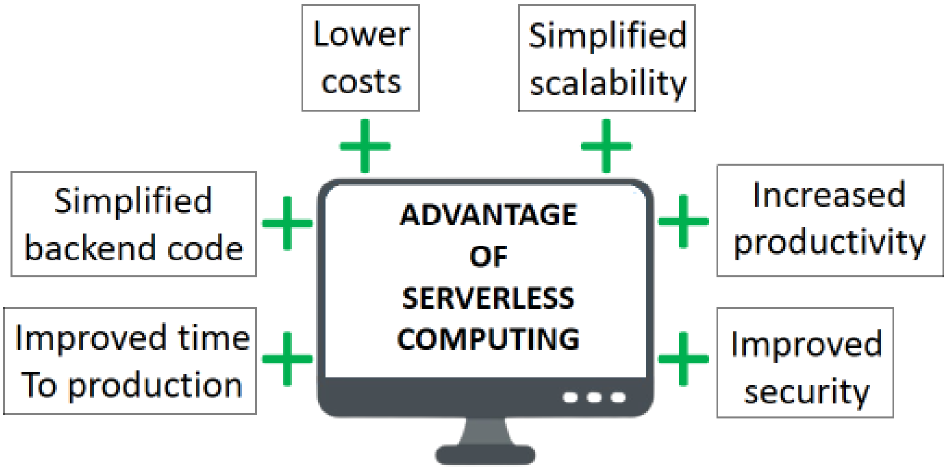


Figure 5. Introduction to Server-less Computing

Furthermore, serverless computing is expected to play a significant role in future cloud platforms, as it offers several advantages over traditional approaches (Brown et al. 2005). These advantages include improved scalability, cost-effectiveness, and faster time-to-market for applications (Patidar et al. 2011). By leveraging serverless computing, organizations can optimize resource utilization, reduce operational overhead, and rapidly iterate on development cycles, ultimately driving innovation and competitiveness in the cloud ecosystem (Lannurien et al. 2023).

Overall, the simplicity and lightweight nature of serverless computing, combined with its potential to revolutionize cloud platforms, make it a compelling choice for modern application development and deployment. As organizations continue to adopt and harness the benefits of serverless computing, it is likely to become increasingly dominant in the future cloud landscape.

✓ Security Challenges in Serverless Computing

Security challenges in serverless computing arise due to the dynamic and distributed nature of the environment (O'Meara and Lennon. 2020; Mampage et al. 2021). Serverless functions are susceptible to injection attacks such as SQL injection, NoSQL injection, and command injection (Ngo et al. 2020). Attackers may exploit vulnerabilities in input handling to execute malicious code or access sensitive data (Poeplau et al. 2014). Serverless applications often rely on third-party libraries and services, which may introduce vulnerabilities (Polinsky et al. 2021). Insecure dependencies can lead to security breaches if attackers exploit vulnerabilities in these libraries or services (Decan et al. 2018). Serverless environments abstract away much of the underlying infrastructure, leading to limited visibility and control (Baresi and Quattrocchi. 2021). This makes it challenging to monitor and secure the environment effectively (Mushtaq et al. 2017). Serverless applications may process and store sensitive data, raising concerns about data privacy and compliance (Golec et al. 2021). Ensuring data privacy requires robust encryption, access controls, and compliance measures (Duggineni. 2023). Cold start refers to the delay experienced when a serverless function is invoked for the first time (Ristov et al. 20220. Attackers may exploit this delay to launch attacks, such as denial-of-service (DoS) attacks, during the cold start phase (Huseinovic et al. 2020). Serverless platforms dynamically allocate resources based on demand. Attackers may attempt to exhaust resources by triggering numerous function invocations, leading to resource exhaustion and denial-of-service (DoS) attacks (Ortega-Fernandez and Liberati. 2023).

Cold start attacks indeed pose a significant challenge in serverless computing environments, as the delay in initializing resources can be exploited by attackers (Ortega-Fernandez and Liberati. 2023). This delay, known as cold start, occurs when serverless functions are invoked after being idle, resulting in increased response times and potential performance degradation. Inadequate authorization and authentication are indeed critical areas for improvement in serverless computing, given the challenges associated with limited control over infrastructure visibility and reliance on cloud providers for security measures (Silverman. 2008). Fragmented application boundaries in serverless environments can exacerbate security issues, making it essential to implement robust authentication and authorization mechanisms to protect code and data (Aslanpour et al. 2021). Data at rest and in transit are indeed significant security concerns in serverless computing environments (Xiong et al. 2021).

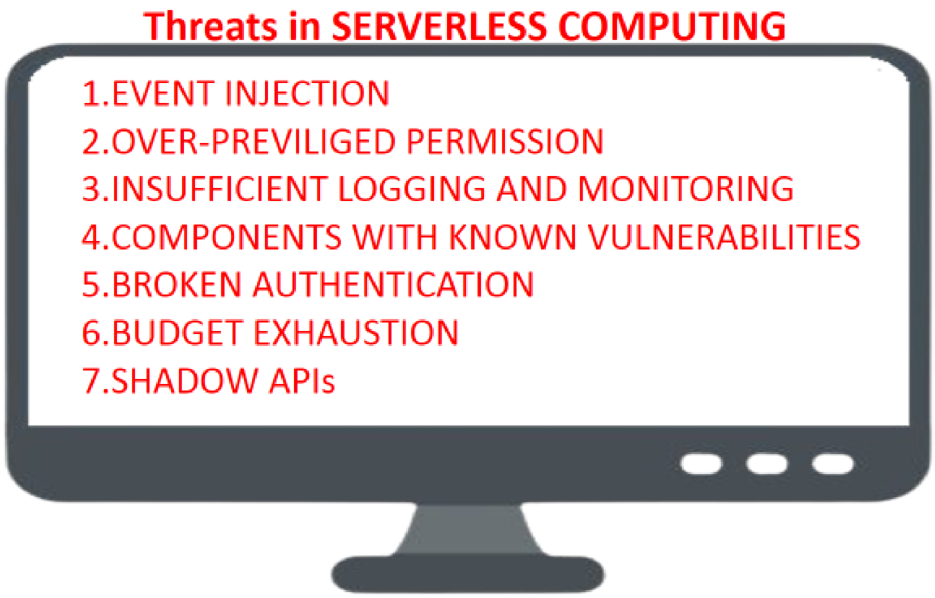
Data at rest refers to data that is stored in persistent storage, while data in transit refers to data being transferred between systems or over a network. Both types of data face various security challenges that can impact the confidentiality, integrity, and availability of the data (Xiong et al. 2021). Ransom-ware attacks targeting data at rest can encrypt sensitive information, rendering it inaccessible and unusable until a ransom is paid. These attacks pose a severe threat to data security and can disrupt operations and cause financial losses for organizations. Data breaches are another critical concern, where malicious attackers gain unauthorized access to stored data and may exfiltration or leak sensitive information. Weak access controls, inadequate encryption, and vulnerabilities in storage systems can all contribute to data breaches in serverless computing environments (Aurangzeb et al. 2017).

Excessive or unauthorized access to data at rest can also lead to security risks, as it increases the likelihood of data exposure and misuse (Mughal. 2019). Organizations must implement robust access controls, encryption mechanisms, and monitoring solutions to protect data at rest and mitigate the risk of unauthorized access and data breaches. Shared resources and multi-tenancy indeed introduce security challenges in serverless computing (Pearson and Benameur. 2010). In a multi-tenant serverless platform, multiple users or tenants share the same resources, such as computing resources, storage, and networking infrastructure (Narasayya and VChaudhuri. 2021).

While each user has access to their own resources and data, the shared environment increases the risk of security breaches and data leakage between tenants. The aspects of resource sharing in serverless computing, particularly in multi-tenant environments, presents a significant threat of hacking.

Attackers may exploit vulnerabilities in shared resources to gain unauthorized access to sensitive data or compromise the security of other tenants' applications. Despite the implementation of encryption and other security measures, multi-tenant environments remain vulnerable to attacks if not properly secured (Adeniyi et al. 2022). In response to these security challenges, researchers have proposed innovative solutions such as the MXFaaS serverless computing platform. MXFaaS aims to provide a secure and efficient multi-tenant environment by implementing advanced security features and improving efficiency levels. By leveraging techniques such as isolation, access controls, and encryption, MXFaaS seeks to mitigate the risks associated with multi-tenancy and enhance the security posture of serverless computing platforms (Stojkovic et al. 2023).

Dependency security is indeed a major concern in serverless computing, particularly with the reliance on third-party dependencies. Serverless platforms dynamically allocate resources based on user demand, leading to a complex deployment environment where various cloud services and dependencies need to be managed. The use of third-party dependencies introduces security risks, as vulnerabilities in these dependencies can be exploited by attackers to compromise the security of serverless applications (Koschel et al. 2021). Additionally, the dynamic nature of serverless computing further complicates dependency management, making it challenging to track and update dependencies effectively (Wen et al. 2023).



Threats in SERVERLESS COMPUTING

1.EVENT INJECTION
2.OVER-PREVILIGED PERMISSION
3.INSUFFICIENT LOGGING AND MONITORING
4.COMPONENTS WITH KNOWN VULNERABILITIES
5.BROKEN AUTHENTICATION
6.BUDGET EXHAUSTION
7.SHADOW APIs

Figure 6. Threats in Serverless Computing

The shortcomings of serverless platforms in addressing dependency security issues and emphasized the need for proper security solutions. These solutions may include implementing robust vulnerability scanning and management processes, conducting regular security assessments of third-party dependencies, and enforcing secure coding practices to mitigate the risk of vulnerabilities introduced by dependencies (Kumari et al. 2023). Overall, addressing dependency security in serverless computing requires proactive measures to identify, assess, and mitigate security risks associated with third-party dependencies. By implementing proper security solutions and practices, organizations can enhance the security posture of their serverless applications and mitigate the risk of exploitation due to dependency vulnerabilities.

✓ **Solutions to Security Challenges:**

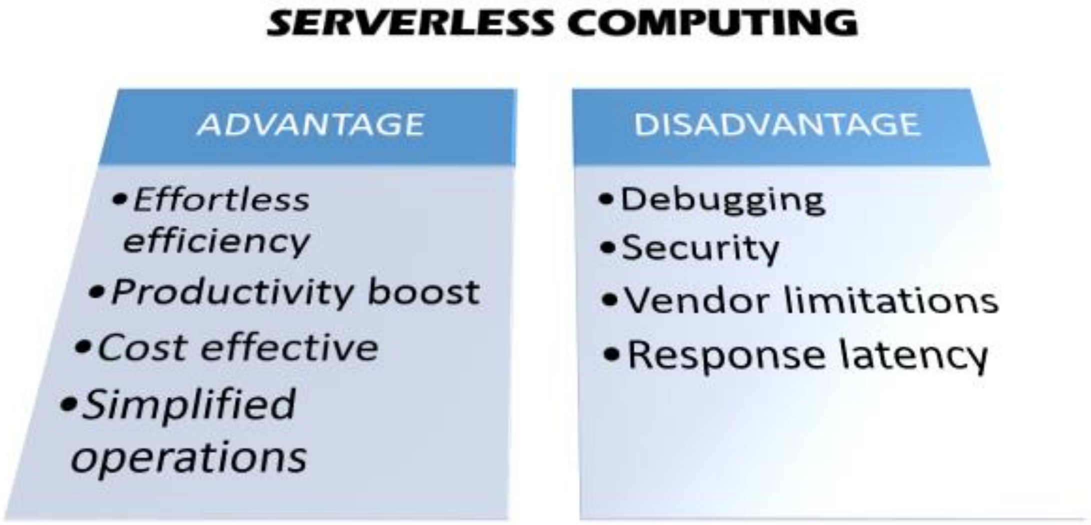


Figure 7. Solution in Security Challenges in Server-less computing

Implement strong authentication mechanisms, such as OAuth or JWT, to verify the identity of users and services accessing serverless functions (Padma and Srinivasan. 2023). Enforce fine-grained access controls to ensure that only authorized entities can invoke functions and access resources (Damiani et al. 2002). Encrypt data at rest and in transit to protect sensitive information from unauthorized access. Utilize encryption techniques such as SSL/TLS for data in transit and encryption algorithms like AES for data at rest (Mustafa et al. 2018). Follow secure coding practices to mitigate common vulnerabilities such as injection attacks, cross-site scripting (XSS), and insecure deserialization. Use input validation, parameterized queries, and output encoding to prevent injection attacks (Nadji et al. 2009). Implement robust monitoring and logging solutions to detect and respond to security incidents in real-time (Pan et al. 2019). Monitor function invocations, resource usage, and access logs to identify anomalous behavior and potential security threats (Berlin et al. 2015).

Regularly audit and update third-party dependencies to patch vulnerabilities and mitigate security risks. Use dependency management tools and vulnerability scanners to identify and remediate vulnerabilities in dependencies (Cobieigh et al. 2018). Implement strong isolation mechanisms to prevent unauthorized access between serverless functions and resources. Use containerization or virtualization to isolate functions and enforce least privilege access controls (Ahmadi. 2024). Implement secure deployment pipelines to automate the deployment of serverless applications while adhering to security best practices. Use infrastructure-as-code (IaC) tools such as Terraform or AWS CloudFormation to define and provision serverless resources securely (Boscain. 2023).

Develop incident response and disaster recovery plans to quickly respond to security incidents and minimize their impact. Regularly test and update these plans to ensure their effectiveness in mitigating security threats (Scarfone et al. 2008).

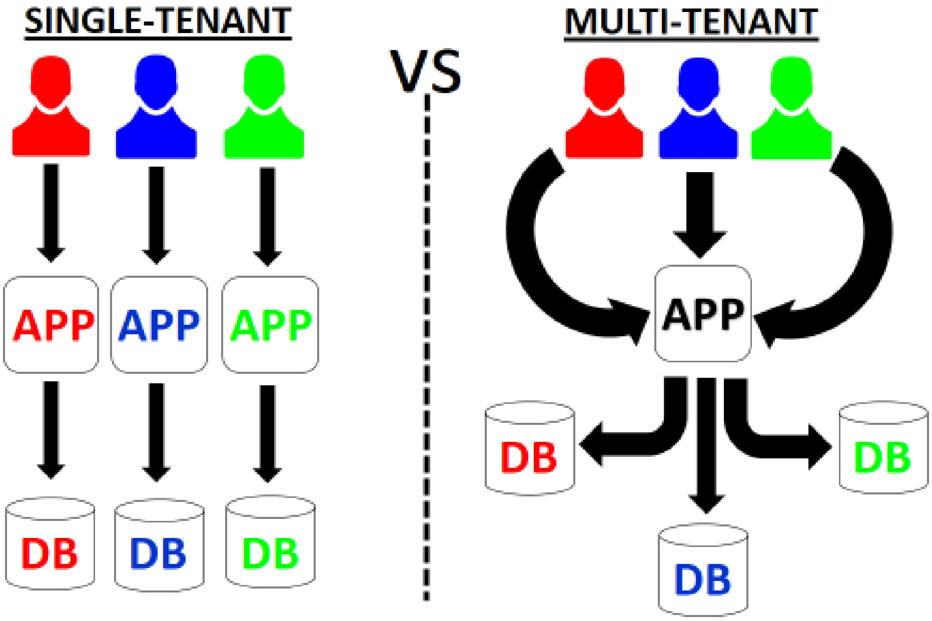


Figure 8. Single-Tenant vs Multi-Tenant

To mitigate security issues in serverless computing, including cold-start attacks, proper mitigation measures are crucial. Implementing full memory encryption helps reduce the risk of data exposure during cold-start attacks. By encrypting the entire memory, including code, data, and encryption keys, organizations can prevent attackers from extracting sensitive information even if they gain access to the physical memory. Preventing the serverless environment from entering sleep mode eliminates the opportunity for attackers to exploit cold-boot attacks (Halderman et al. 2009; Simmons. 2011).

By disabling sleep capabilities, organizations can ensure that the system remains active and responsive at all times, reducing the risk of unauthorized access during idle periods. Implementing IAM (Identity and Access Management) and following the Least Privilege Principle are indeed effective strategies to address authorization and authentication issues in serverless computing. IAM allows organizations to manage user identities and their access to resources in the serverless environment. By defining roles, policies, and permissions, IAM ensures that only authorized users can invoke functions and access resources. IAM enables centralized control over user access, simplifying the management of permissions and reducing the risk of unauthorized access (Drame-Maigne et al. 2021; Sabbioni et al. 2022).

The Least Privilege Principle restricts user privileges to the minimum necessary for performing their tasks. By granting users access only to the resources and data they need, organizations can minimize the risk of unauthorized access and data breaches. This principle reduces the attack surface and limits the potential impact of security incidents. Implementing time-limited privileges ensures that users can access sensitive information only for a limited duration. By setting expiration times on access credentials or tokens, organizations can mitigate the risk of unauthorized access and enhance security. Time-limited privileges reduce the window of opportunity for attackers to exploit compromised credentials. Implementing encryption practices is indeed a crucial strategy for mitigating data security issues in serverless computing environments (Patross et al. 2021).

By encrypting data at rest and in transit, organizations can protect sensitive information from unauthorized access and ensure its confidentiality and integrity. Encrypting data at rest ensures that stored data remains protected even if unauthorized access occurs. By encrypting data using strong encryption algorithms and securely managing encryption keys, organizations can prevent unauthorized access and maintain data confidentiality. Encrypting data in transit protects information as it moves between different components of the serverless environment. Transport encryption, implemented using protocols like TLS/SSL, ensures that data exchanged between serverless functions, databases, and other services is secure and cannot be intercepted or tampered with by attackers (Shin et al. 2020).

Attribute-Based Encryption (ABE) allows organizations to enforce fine-grained access controls based on specific attributes or policies (Goyal et al. 2006). By encrypting data with attributes that define access permissions, organizations can ensure that only authorized users or entities can access sensitive information. ABE enhances security by limiting access to data based on predefined criteria, such as user roles, attributes, or organizational policies. Developing efficient and secure access control systems, as mentioned by the researchers, is essential for ensuring data security in serverless computing environments. By implementing access control mechanisms that leverage attribute-based encryption and other security techniques, organizations can enforce granular access controls and protect sensitive data from unauthorized access (Goyal et al. 2006).

Implementing proper multi-tenancy security measures is indeed crucial for ensuring the security of serverless computing environments. The multi-tenant security model enables organizations to efficiently share resources while maintaining isolation and security between tenants (Gulati and Gupta. 2012). Implement resource allocation and segregation strategies to ensure that each tenant has access to the resources they need while maintaining isolation between tenants. Utilize techniques such as virtualization, containerization, and resource quotas to allocate and segregate resources securely (Rodriguez and Buyya. 2019).

Adopt a multi-tenant security model that defines roles, permissions, and access controls for each tenant. Ensure that tenants can only access their own resources and data, preventing unauthorized access and data leakage between tenants. Determine the appropriate degree of multi-tenancy based on the organization’s requirements and security considerations. High, middle, and low degrees of multi-tenancy offer varying levels of isolation and resource sharing, allowing organizations to balance security and efficiency based on their needs. Recognize the benefits of multi-tenancy, such as effective resource usage, easier deployment, and optimized billing (Gulati and Gupta. 2012).

By leveraging multi-tenancy, organizations can maximize resource utilization, streamline application deployment, and reduce costs while maintaining security and isolation between tenants. Continuously maintain and optimize the multi-tenant security model to address evolving security threats and ensure the effectiveness of security measures (Gulati and Gupta. 2012). Regularly review and update access controls, audit logs, and security configurations to mitigate potential security risks and vulnerabilities (Stamp et al. 2003).

Leveraging real-time monitoring tools and anomaly detection techniques can significantly enhance monitoring solutions in serverless computing environments (Omotunde and Ahmed. 2023).

Additionally, Cloud Security Posture Management (CSPM) plays a crucial role in improving the security posture of serverless platforms. CSPM enables organizations to identify security issues, vulnerabilities, and misconfigurations in their serverless environments (Olaoye and Luz. 2024). By continuously monitoring the configuration settings and security controls, CSPM tools can alert organizations to potential security risks and help them take proactive measures to mitigate these risks. CSPM provides automated remediation capabilities, allowing organizations to address security issues and misconfigurations in real-time. By automatically enforcing security policies and applying remediation actions, CSPM tools help organizations maintain a strong security posture and reduce the risk of security breaches. CSPM helps organizations ensure compliance with industry regulations and security standards by providing visibility into their cloud infrastructure's security posture. CSPM tools offer features for tracking compliance requirements, generating compliance reports, and implementing controls to maintain compliance with regulatory frameworks. CSPM tools assess and identify security risks in serverless environments, helping organizations prioritize and address the most critical security issues (Olaoye and Luz. 2024).

By conducting risk assessments and vulnerability scans, CSPM tools enable organizations to proactively mitigate security risks and enhance their overall security posture. CSPM provides real-time monitoring capabilities, allowing organizations to detect and respond to security threats as they occur. By monitoring serverless infrastructure for suspicious activity and anomalous behavior, CSPM tools help organizations identify and mitigate security incidents in real-time. CSPM tools can help organizations optimize costs by identifying inefficient resource usage, unused resources, and opportunities for resource consolidation. By optimizing resource utilization and minimizing unnecessary expenses, CSPM tools help organizations reduce their cloud infrastructure costs while maintaining security. Overall, CSPM plays a crucial role in improving the security, governance, compliance, and cost optimization of serverless platforms. By leveraging CSPM tools and practices, organizations can enhance their security posture, mitigate risks, and ensure the integrity and availability of their serverless applications and data (Haber et al. 2022).

✓ **Serverless Security Tools and Technologies:**

Platforms specifically designed for securing serverless applications offer comprehensive security features, including vulnerability scanning, runtime protection, access control, and threat detection (Podjarny and Tal. 2019). Examples include PureSec, Protego, and Stackery. Cloud Security Posture Management (CSPM) tools provide capabilities for assessing, managing, and improving the security posture of cloud environments, including serverless platforms. These tools offer features such as configuration auditing, compliance monitoring, and risk assessment. Examples include CloudGuard by Check Point and Cloud Conformity. Some security tools are tailored specifically for serverless environments, offering functionalities such as static code analysis, dependency scanning, and runtime monitoring. Examples include OWASP Serverless Top 10, Serverless Security by Sqreen, and Serverless Security Scanner by PureSec. Runtime Application Self-Protection (RASP) tools offer runtime protection for serverless applications by monitoring and mitigating security threats in real-time (Ang'udi. 2023; Olaoye and Luz. 2024).

These tools can detect and prevent attacks such as injection, tampering, and unauthorized access (Alwan and Younis. 2017). Examples include Signal Sciences and RASP by Check Point. Log management and Security Information and Event Management (SIEM) tools help organizations collect, analyze, and correlate logs and security events from serverless environments. These tools provide visibility into activity and help detect and respond to security incidents. Examples include Splunk, ELK Stack (Elasticsearch, Logstash, Kibana), and Sumo Logic (Tuyishime. 2023). Securely managing secrets and sensitive information, such as API keys and access tokens, is crucial in serverless environments. Secrets management solutions help organizations store, rotate, and access secrets securely. Examples include HashiCorp Vault, AWS Secrets Manager, and Azure Key Vault (Jegan et al. 2020). CI/CD pipelines play a critical role in deploying serverless applications. Security tools integrated into CI/CD pipelines help automate security testing, vulnerability scanning, and compliance checks throughout the development lifecycle. Examples include Snyk, SonarQube, and GitLab CI/CD (Orazi et al. 2020). Subscribing to threat intelligence feeds tailored for serverless environments can provide organizations with insights into emerging threats and vulnerabilities relevant to their serverless applications. Examples include threat intelligence feeds provided by security vendors and industry organizations. Shifting the focus of application security to individual functions within a serverless application is indeed a key aspect of serverless security (Efterpi. 2020).

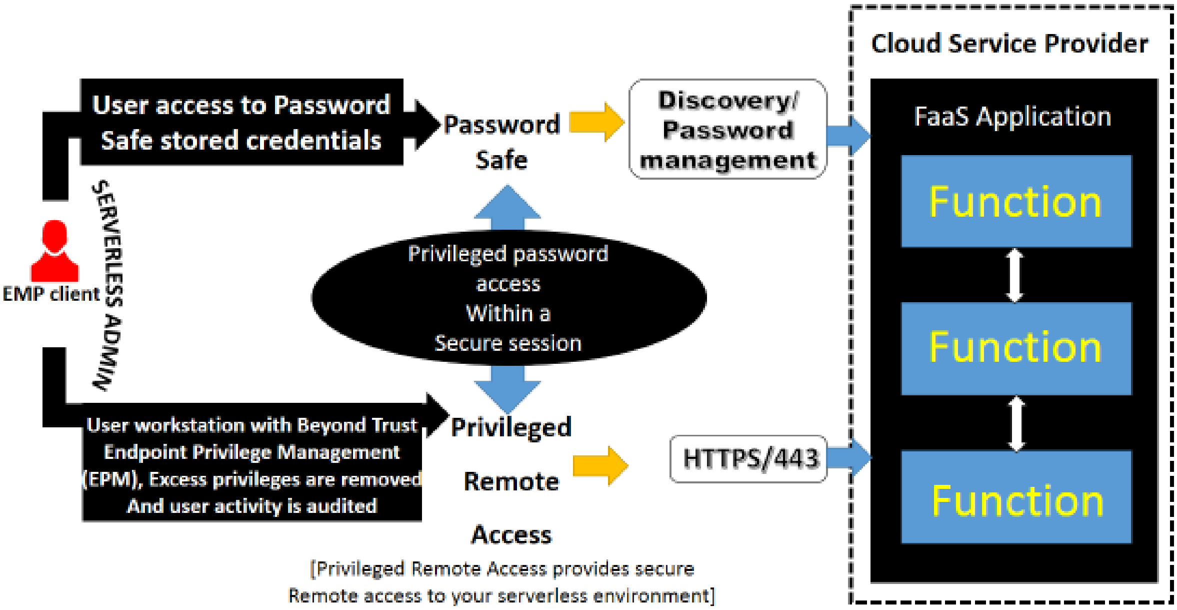

Cloud Service Provider
User access to Password Safe stored credentials
Password Safe
Discovery/ Password management
FaaS Application
Function
Function
Function
EMP client
SERVERLESS ADMIN
Privileged password access Within a Secure session
User workstation with Beyond Trust Endpoint Privilege Management (EPM), Excess privileges are removed And user activity is audited
Privileged Remote Access
HTTPS/443
[Privileged Remote Access provides secure Remote access to your serverless environment]

Figure 9. Server-less Security Practices

By adopting a function-centric security approach, organizations can implement least privileged access control and proper application hardening, thereby enhancing their overall security posture. By securing individual functions and limiting their permissions to only what is necessary, organizations can reduce the attack surface and mitigate the risk of unauthorized access. Each function operates within its own security context, ensuring that it can only access the resources and data required for its intended purpose. Applying security controls and best practices at the function level helps harden the application against potential vulnerabilities and attacks. This includes implementing input validation, output encoding, and other security measures to prevent common exploits such as injection attacks and cross-site scripting (XSS) (Gupta and Gupta. 2017).

Adopting a function-centric security approach helps organizations maintain compliance with regulatory requirements and industry standards. By ensuring that each function adheres to security policies and guidelines, organizations can demonstrate a strong security posture and reduce the risk of compliance violations (Humphreys. 2008). Amazon's introduction of AWS Lambda in 2014 marked the beginning of serverless computing, and since then, many other providers have entered the market with their own secure serverless computing services (Jonas et al. 2019). These providers offer various security features and capabilities to help organizations protect their serverless applications and data effectively. The multifaceted nature of serverless computing, which encompasses various security aspects within cloud, mobile, and application environments (Nastic and Dustdar. 2018). Serverless computing offers improved security measures compared to raditional architectures due to its stateless and ephemeral nature (Shaflei et al. 2022). Serverless functions, such as those provided by AWS Lambda, operate without persistent state or memory (Malawski et al. 2020). This limits the attack surface and reduces the risk of long-term security attacks, as functions are terminated after execution and do not retain sensitive data (Saad et al. 2020).

Serverless functions are ephemeral, meaning they have short lifespans and are disposed of after execution (Guo et al. 2022). This ephemeral nature reduces the exposure window for potential security threats and minimizes the impact of compromised functions. The stateless and ephemeral nature of serverless computing simplifies security management and enforcement. Organizations can focus on securing individual functions and implementing security measures at the function level, leading to improved overall security (Hobday and Dancer. 2013).

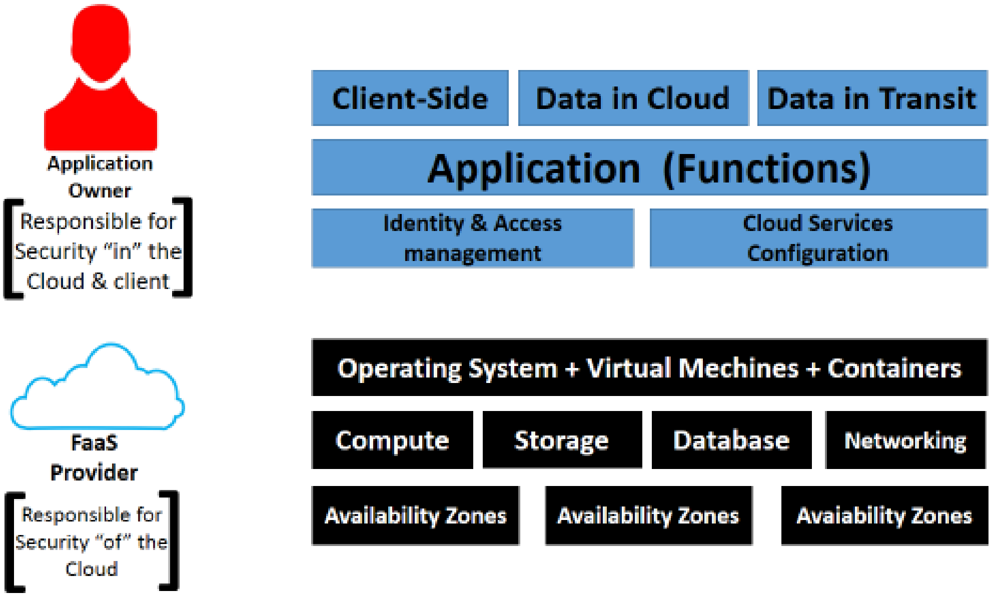

Application Owner
Responsible for Security "in" the Cloud & client
FaaS Provider
Responsible for Security "of" the Cloud
Client-Side
Data in Cloud
Data in Transit
Application (Functions)
Identity & Access management
Cloud Services Configuration
Operating System + Virtual Mechines + Containers
Compute
Storage
Database
Networking
Availability Zones
Availability Zones
Avaiability Zones

Figure 10. Server-less Security Using the Shared Model

Serverless platforms automatically scale resources in response to demand, providing dynamic scalability without sacrificing security (Mampage et al. 2021). This flexibility enables organizations to efficiently allocate resources while maintaining a secure environment. Overall, serverless computing offers inherent security advantages, including statelessness, ephemeral nature, and dynamic scalability (Gentry et al. 2021). By leveraging these characteristics and implementing security best practices, organizations can enhance the security of their serverless applications and mitigate the risks associated with modern cloud and mobile environments (Patros et al. 2021).

3. STUDYING THE SECURITY CHALLENGES OF SERVERLESS COMPUTING IN CLOUD ENVIRONMENTS HAS SHED LIGHT ON VARIOUS DIFFICULTIES AND THEIR CORRESPONDING SOLUTIONS IN NETWORK MARKETING

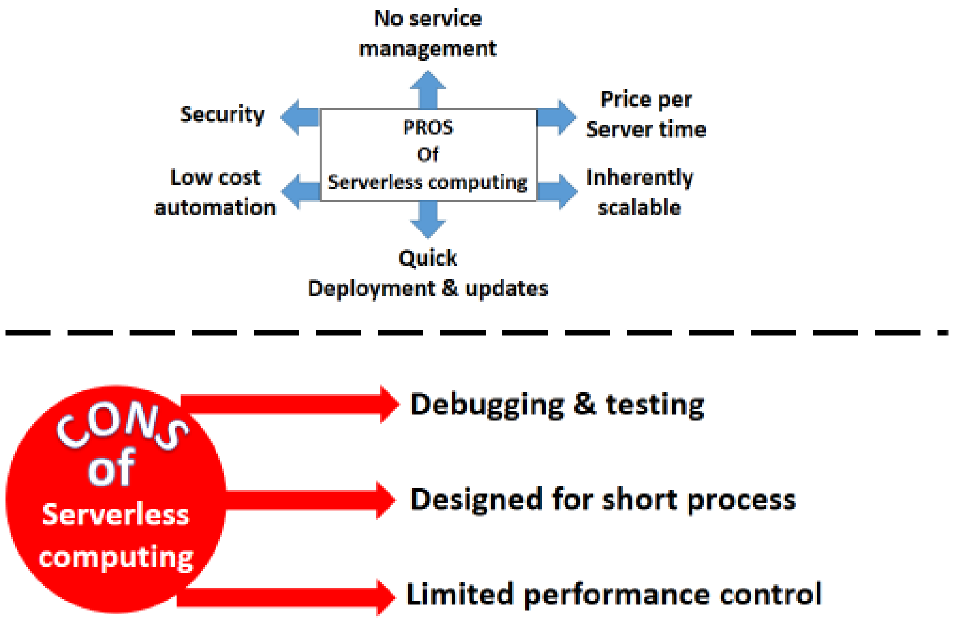

No service management
Security
PROS Of Serverless computing
Price per Server time
Low cost automation
Inherently scalable
Quick Deployment & updates
CONS of Serverless computing
Debugging & testing
Designed for short process
Limited performance control

Figure 11. Pros and Cons of Server-less Computing

3. 1. Limited Visibility and Control

Limited visibility and control pose significant challenges in securing serverless computing environments (Mampage et al. 2022). Implementing real-time monitoring solutions allows organizations to gain visibility into their serverless environments, monitor for suspicious activity, and detect security incidents as they occur. Enabling comprehensive logging and auditing capabilities helps organizations track user activity, system events, and resource usage within serverless environments (Mampage et al. 2022; Prakash and Kumar. 2022).

This enhances visibility and enables organizations to investigate security incidents effectively (George et al. 2023). Utilizing CSPM tools provides organizations with centralized visibility and control over their cloud infrastructure, including serverless computing resources.

CSPM solutions offer features for assessing security posture, identifying misconfigurations, and enforcing security policies (Loaiza Enriquez. 2021). Integrating serverless environment logs and security events with Security Information and Event Management (SIEM) solutions enables organizations to correlate and analyze data from multiple sources, improving threat detection and incident response capabilities (Lopez Velasquez et al. 2023).

Implementing automated security controls, such as intrusion detection systems (IDS), web application firewalls (WAF), and anomaly detection mechanisms, helps organizations proactively identify and mitigate security threats in serverless environments (Jacob. 2022).

Role-based Access Control (RBAC) policies ensures that only authorized users have access to serverless resources and functions, reducing the risk of unauthorized access and data breaches (Sankaran et al. 2020). Conducting regular security assessments, vulnerability scans, and penetration tests helps organizations identify and remediate security weaknesses in serverless applications and configurations. Leveraging logging mechanisms and implementing strict monitoring practices are effective responses to the challenges of limited visibility and control in serverless computing environments (Uddin et al. 2019). By implementing cloud-native monitoring tools during the implementation phase, organizations can enhance their ability to observe serverless computing behavior and gain real-time insights into function execution, network interactions, and resource usage. Enhanced logging and monitoring enable organizations to detect security incidents promptly and respond effectively (Bhatt et al. 2014). Real-time data on function execution and resource usage allow for proactive threat detection and mitigation (Chen et al. 2016).

By continuously monitoring serverless environments, organizations gain a high level of awareness of system activity and behavior. This increased visibility enables organizations to identify anomalous behavior and potential security threats more easily. The insights provided by comprehensive logging and monitoring facilitate the refinement and enhancement of security protocols and measures. Organizations can use this information to fine-tune access controls, optimize security configurations, and mitigate vulnerabilities effectively. Access to real-time data on serverless computing behavior empowers organizations to make more informed and effective decisions regarding security policies, resource allocation, and incident response strategies (Straub and Welke. 1998).

The successful implementation of logging mechanisms and monitoring practices underscores the critical need for visibility and control in cloud-based systems, particularly in serverless computing environments (Malik et al. 2024). It emphasizes the importance of proactive security measures and continuous monitoring to maintain the integrity and security of serverless applications and data. Overall, leveraging logging mechanisms, implementing strict monitoring practices, and utilizing cloud-native monitoring tools empower organizations to overcome the challenges of limited visibility and control in serverless computing environments. These measures enhance security, increase awareness, and enable informed decision-making, laying a strong foundation for effectively managing serverless computing systems (Leemans. 2022).

3. 2. Insecure Dependencies

Maintain a robust dependency management process to keep track of all third-party libraries and components used in serverless applications. Regularly update dependencies to ensure they are patched against known vulnerabilities and security issues. Use static code analysis tools to scan serverless application code for potential security vulnerabilities and insecure dependencies (Alpernas et al. 2021).

These tools can identify and flag vulnerable dependencies before deployment (Ponta et al. 2020). Integrate dependency scanning tools into the CI/CD pipeline to automatically detect and flag insecure dependencies during the build process. This allows organizations to address security issues early in the development lifecycle. Conduct regular vulnerability assessments and security audits of serverless applications to identify and remediate insecure dependencies (Sushma et al. 2023).

Implement automated tools and manual reviews to assess the security posture of third-party components (Jegan et al. 2023). Maintain a whitelist of approved dependencies and libraries that meet predefined security criteria. Only allow the use of dependencies from the whitelist to mitigate the risk of insecure components being introduced into the application. Implement runtime protection mechanisms to monitor serverless functions for suspicious behavior and detect attempts to exploit insecure dependencies at runtime (Jegan et al. 2023).

Use tools that provide runtime application self-protection (RASP) capabilities to defend against known and unknown threats (Inamdar and Gupta. 2020). Train developers on secure coding practices and best practices for managing dependencies in serverless applications. Emphasize the importance of vetting and validating third-party components before integration. Implement continuous monitoring of serverless applications to detect and respond to security incidents related to insecure dependencies. Monitor for anomalous behavior and unauthorized access attempts that may indicate a compromise. Conduct regular audits of third-party dependencies to identify and remediate security vulnerabilities (Tatineni. 2023). These audits help ensure that only secure and trusted components are used in serverless applications.

Use static code analysis tools to analyze third-party libraries for potential security vulnerabilities (Backes et al. 2016). This proactive approach helps detect and address security issues before they are deployed into production environments. Keep dependencies up to date by regularly applying patches and updates provided by the vendors. This helps mitigate the risk of known vulnerabilities being exploited by attackers. Implement proactive monitoring of dependencies to detect any suspicious behavior or unauthorized access attempts (Ghorbani et al. 2009).

By continuously monitoring dependencies, organizations can identify and respond to security threats in real-time. By carefully managing and updating dependencies, organizations can reduce the attack surface and minimize the number of potential entry points for attackers (Albanese et al. 2014). This helps protect sensitive data and mitigate the risk of unauthorized access. Adopt a culture of continuous improvement by regularly reviewing and updating security practices and protocols related to dependency management. This ensures that security measures remain effective in addressing evolving threats and vulnerabilities. Overall, a proactive approach to managing dependencies, including regular audits, static code analysis, updates, and proactive monitoring, is essential for securing serverless computing environments effectively. By implementing these strategies, organizations can reduce the risk of insecure dependencies and strengthen the overall security posture of their serverless applications (Cinar. 2023).

3. 3. Cold Start Attacks

Pre-warming functions involves periodically invoking serverless functions to keep them warm and ready for use (Roy et al. 2022). By pre-warming functions, organizations can reduce the latency associated with cold start times and ensure faster response times for users. Optimize serverless function code to reduce execution time and minimize cold start delays (Silva et al. 2020). This may involve optimizing resource usage, minimizing dependencies, and streamlining code execution paths (Vahidinia et al. 2020).

Utilize container reuse strategies to keep containers running for longer periods and minimize the frequency of cold start events (Suo et al. 2021). By reusing containers, organizations can reduce the overhead associated with cold start times and improve overall performance. Implement auto-scaling policies that dynamically adjust resources based on workload demand. By scaling resources up or down in response to traffic fluctuations, organizations can minimize the occurrence of cold start events during periods of high demand (Jegannathan et al. 2022).

Optimize serverless platform configurations to prioritize warm start instances over cold start instances (Pan et al. 2023). This may involve adjusting platform settings, such as instance provisioning and allocation policies, to prioritize warm start behavior. Use predictive scaling algorithms to anticipate workload patterns and proactively provision resources before they are needed. Predictive scaling helps minimize cold start times by ensuring that sufficient resources are available to handle incoming requests. Leverage platform-specific features and capabilities designed to mitigate cold start delays. For example, some serverless platforms offer features like provisioned concurrency or reserved instances, which can help reduce cold start times by ensuring that resources are pre-allocated and readily available (Shahrad et al. 2020). Conduct performance testing to assess the impact of cold start delays on application performance and user experience.

By identifying potential bottlenecks and optimizing resource allocation, organizations can minimize the impact of cold start events on overall application performance. Implement function warm-up mechanisms to periodically activate serverless functions, ensuring they remain in a warmed-up state and ready for rapid execution (Wang et al. 2023). This reduces the delays associated with cold start events and improves overall responsiveness. By proactively warming up functions, organizations can minimize the vulnerability to cold start attacks and enhance the system's security. Strategically distribute serverless functions within the architecture to optimize resource allocation and anticipate demand (Boza et al. 2017). Consider factors such as geographical location, expected usage patterns, and resource availability when deploying functions.

By strategically placing functions, organizations can minimize latency and improve performance, reducing the likelihood of cold start delays. Allocate sufficient resources to serverless functions based on anticipated demand and workload requirements. Use auto-scaling policies and predictive scaling algorithms to dynamically adjust resource allocation to match fluctuating demand levels (Park and Jeong. 2023). By ensuring adequate resource allocation, organizations can minimize the impact of cold start events and maintain optimal system performance. Optimize serverless function code to reduce execution time and minimize cold start delays. Streamline code execution paths, minimize dependencies, and optimize resource usage to improve overall efficiency and responsiveness (Golec et al. 2023). By optimizing code, organizations can mitigate the impact of cold start attacks and enhance the security and performance of serverless systems.

Continuously monitor serverless systems for performance bottlenecks and optimization opportunities. Use performance metrics and analytics to identify areas for improvement and implement optimizations to enhance system efficiency and responsiveness (Gunasekaran et al.

2004). By continuously optimizing serverless systems, organizations can maintain a high level of security and performance, mitigating the risk of cold start attacks and other security threats (Simmons. 2011).

By adopting these strategies, organizations can strengthen the security and performance of their serverless systems, minimize the impact of cold start attacks, and ensure optimal responsiveness for users. Additionally, these measures contribute to the overall efficiency and reliability of serverless computing environments, enhancing the organization's ability to deliver secure and responsive services to customers (mampage et al. 2022).

3. 4. Data Privacy and Resilience

Implement strong encryption mechanisms to protect data at rest and in transit within serverless environments. Use industry-standard encryption algorithms and encryption keys to ensure the confidentiality and integrity of sensitive data. Implement robust access control measures to restrict access to sensitive data within serverless applications (Hellerstein. 2018). Use role-based access control (RBAC) policies, least privilege principles, and multi-factor authentication (MFA) to enforce strict access controls and prevent unauthorized access to sensitive data. Employ data masking and anonymization techniques to obfuscate sensitive data and protect user privacy (Omotunde and Ahmed. 2023).

By masking or anonymizing personally identifiable information (PII) and other sensitive data, organizations can reduce the risk of data breaches and unauthorized access. Implement data resilience measures such as regular data backups, redundant storage, and disaster recovery plans to ensure data availability and integrity. Store backups in geographically dispersed locations and regularly test data recovery procedures to mitigate the impact of data loss or corruption (Kesa. 2023).

Adopt a privacy by design approach when designing and developing serverless applications. Consider data privacy and security requirements from the outset and incorporate privacy-enhancing technologies and features into the application architecture. Implement data lifecycle management practices to govern the collection, storage, processing, and deletion of data within serverless environments (Kumari et al. 2023).

Define clear policies and procedures for data retention and deletion to ensure compliance with data privacy regulations (Butin and Le Metayer. 2015). Implement continuous monitoring and auditing mechanisms to track data access, usage, and security events within serverless applications. Use logging, monitoring, and auditing tools to detect and respond to security incidents in real-time and maintain visibility into data activity. Ensure compliance with relevant data privacy regulations and industry standards, such as GDPR, CCPA, HIPAA, and PCI DSS (Asad. 2023). Stay informed about regulatory requirements and incorporate compliance measures into serverless application design and operation.

Encryption is a fundamental and effective security measure for protecting data integrity in serverless computing environments (Shin. 2020). By encrypting data, organizations can ensure that sensitive information remains unreadable to unauthorized users, both at rest and during transmission. Encryption helps organizations meet compliance requirements, such as those outlined in regulations like GDPR, HIPAA, and PCI DSS, by safeguarding sensitive data from unauthorized access and disclosure. Implementing encryption measures instills confidence in both the organization and its users regarding the security of their data. Users can trust that their sensitive information is adequately protected, fostering a positive relationship with the organization (Richards and Hartzog. 2015).

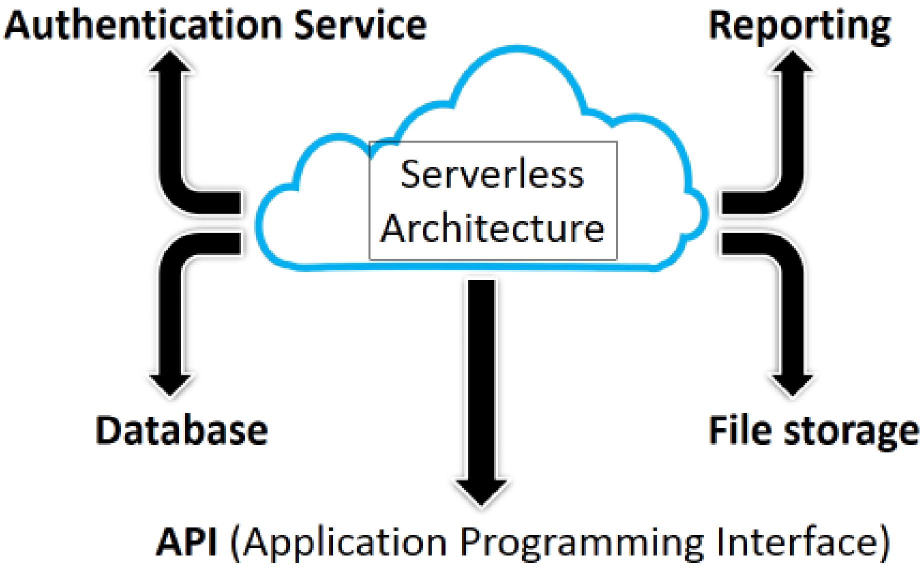


Figure 12. Data Privacy in Server-less Architecture

Additionally, encryption helps mitigate the risk of data breaches and unauthorized access, reducing the potential impact of security incidents on the organization's reputation and financial well-being (Telo. 2023). Successful encryption measures contribute to a reliable defense posture for serverless computing systems, aligning with best practices for data security (Verma and Upadhayay. 2019). By prioritizing encryption as part of their security strategy, organizations can enhance the overall trustworthiness of their serverless computing environments and demonstrate their commitment to protecting sensitive data. Ultimately, encryption plays a vital role in maintaining data confidentiality, integrity, and security in serverless computing systems, contributing to a safer and more secure digital ecosystem (Singh and Dautaniya. 2019)0.

3. 5. Injection Vulnerabilities

Implement strict input validation and sanitization mechanisms to filter and sanitize user inputs before processing them (Scholte et al. 2012)0. This helps prevent malicious input data from being executed as code or injected into the application. Use parameterized queries or prepared statements when interacting with databases or executing dynamic queries. Parameterized queries separate data from code, preventing attackers from injecting malicious SQL or NoSQL commands into queries (Ron et al. 2016).

Use Object Relational Mappers (ORMs) or similar frameworks to interact with databases in a safe and secure manner (Reniers et al. 2019)0. ORMs abstract database interactions, automatically handling parameterization and preventing injection attacks. Implement security headers, such as Content Security Policy (CSP) and Strict-Transport-Security (HSTS), to protect against injection attacks and other common web vulnerabilities (Mlyatu and Sanga. 2022). These headers help mitigate the risk of cross-site scripting (XSS) and other injection-related attacks (Yakoob. 2021).

Utilize API ateways and web application firewalls (WAFs) to filter and monitor incoming requests for malicious payloads and injection attempts (Heckathorn. 2011). These tools can block or flag suspicious requests before they reach the serverless application. Conduct regular code reviews and static code analysis to identify and remediate injection vulnerabilities in serverless application code. Use automated tools and manual reviews to identify potential security issues and enforce secure coding practices. Follow the principle of least privilege by granting only the minimum level of access and permissions necessary for each function or service. Limit access to sensitive resources and data to reduce the potential impact of injection attacks (Tan et al. 2017).

Perform regular security audits and penetration testing to identify and address injection vulnerabilities in serverless applications (Dutta et al. 2020). Test for common injection techniques, such as SQL injection, NoSQL injection, and command injection, to ensure robust security posture (Eassa et al. 2018). Utilizing proactive runtime mechanisms, such as Web Application Firewalls (WAFs), has indeed proven to be highly effective in mitigating the risks associated with injection attacks in serverless computing environments. WAFs operate by continuously monitoring incoming requests and responses, detecting patterns indicative of injection attacks, and blocking malicious traffic before it reaches the serverless application (Balaganski. 2015).

By integrating WAFs into the serverless environment, organizations can establish an additional layer of defense against unauthorized access and data manipulation resulting from injection attempts. WAFs help ensure the integrity and reliability of data by preventing malicious payloads from compromising the serverless system. The deployment of WAFs aligns with best practices for securing serverless computing environments and reinforces the overall security posture of the organization. By leveraging proactive runtime mechanisms like WAFs, organizations can effectively safeguard against injection vulnerabilities and maintain the trustworthiness of their serverless applications (Jegan et al. 2020).

3. 6. Resource Exhaustion and Denial-of-Service (DoS):

Implement rate limiting mechanisms to restrict the number of requests or function invocations from a single source within a specified time frame (Schmidt et al. 1998). By imposing limits on incoming requests, organizations can prevent malicious actors from overwhelming serverless functions and consuming excessive resources. Utilize auto-scaling capabilities provided by serverless platforms to dynamically adjust resources based on workload demand (Benedetti et al. 2022). Configure auto-scaling policies to automatically provision additional resources during periods of high traffic or activity, ensuring sufficient capacity to handle incoming requests without experiencing resource exhaustion (Katal et al. 2021). Implement throttling mechanisms to control the rate of requests or function invocations based on predefined thresholds. Throttling helps prevent excessive resource consumption by limiting the frequency or volume of incoming requests, thereby mitigating the impact of DoS attacks (Zhijun et al. 2020).

Implement request validation mechanisms to verify the legitimacy of incoming requests and filter out malicious or malformed requests. Perform input validation, parameter validation, and content validation to ensure that only valid and safe requests are processed by serverless functions (Samea et al. 2020). Deploy DDoS protection services or solutions to detect and mitigate large-scale DDoS attacks targeting serverless applications. Use DDoS mitigation techniques, such as traffic filtering, rate limiting, and IP blocking, to mitigate the impact of volumetric attacks and maintain service availability (Devi and Subbulakshmi. 2017).

Implement circuit breaker patterns to detect and respond to abnormal or excessive traffic patterns. Use circuit breakers to temporarily suspend access to serverless functions or throttle incoming requests when predefined thresholds are exceeded, preventing resource exhaustion and maintaining system stability (Lebanon et al. 2018). Implement comprehensive monitoring and alerting mechanisms to continuously monitor resource utilization, performance metrics, and traffic patterns in serverless environments (Fairhurst. 2017). Set up alerts to notify administrators of abnormal behavior or potential DoS attacks, enabling timely response and mitigation actions. Leverage built-in security features and protections offered by cloud service providers to defend against resource exhaustion and DoS attacks (Samea et al. 2020).

Cloud providers often offer network-level protections, traffic filtering, and DDoS mitigation services as part of their platform offerings (Somani et al. 2017). The charging model of serverless systems, based on resource consumption, presents a challenge of resource exhaustion attacks. Attackers may attempt to overwhelm the system by triggering numerous functions, leading to denial of service and increased costs. Implement rate-limiting mechanisms to control the frequency of function invocations (Yan and Yu. 2015).

By setting limits on the number of requests or function executions per unit of time, organizations can prevent excessive resource consumption caused by malicious actors. Utilize continuous monitoring tools to detect unusual spikes in system activity that may indicate a potential resource exhaustion attack (Buennemeyer et al. 2008). Monitoring metrics such as function invocations, resource utilization, and network traffic can help identify abnormal behavior and trigger alerts for further investigation. Set up billing alerts to receive notifications when resource consumption surpasses predefined thresholds (Sivanathan. 2020). Billing alerts provide an early warning of escalating costs due to increased usage, enabling organizations to take immediate action to investigate and mitigate potential attachment. Implement automated scaling policies to dynamically adjust resource allocation based on demand (Kimani et al. 2019). By automatically scaling resources up or down in response to changes in workload, organizations can ensure efficient resource utilization and maintain system reliability under varying levels of demand.

By implementing these measures, organizations can strengthen their serverless systems, ensuring efficient resource consumption, effective cost control, and protection against malicious attempts to exploit the platform (Khan et al. 2017). These proactive steps help mitigate the risks associated with resource exhaustion attacks and safeguard the availability and performance of serverless applications.

3. 7. Vendor Lock-in Risks

Adopt a multi-cloud strategy to diversify cloud service providers and reduce dependency on a single vendor (Tomarchio et al. 2020). By using multiple cloud providers, organizations can leverage the strengths of each provider while minimizing the risk of vendor lock-in. Containerize serverless applications using container orchestration platforms like Kubernetes. Containerization abstracts the underlying infrastructure and allows applications to run consistently across different cloud environments, reducing vendor-specific dependencies (Zhou et al. 2019).

Use serverless frameworks that support multiple cloud providers, such as AWS SAM (Serverless Application Model), Azure Functions, or Google Cloud Functions (Kumar. 2019).

These frameworks provide abstractions and compatibility layers that enable applications to be deployed seamlessly across different cloud platforms. Adopt industry-standard APIs and protocols to ensure interoperability between serverless applications and cloud services (Rodrigues et al. 2022). By adhering to open standards, organizations can avoid proprietary vendor lock-in and facilitate migration between cloud providers.

Utilize vendor-neutral tools and services for development, deployment, and management of serverless applications. Choose tools that are compatible with multiple cloud platforms and offer portability across different environments (Sewak and Singh. 2018). Implement data portability strategies to facilitate the transfer of data between cloud providers. Use standardized data formats and protocols to ensure data interoperability and minimize data lock-in (Harsh et al. 2012).

Develop an exit strategy that outlines the process for migrating applications and data away from a specific cloud provider if necessary. Establish clear criteria for evaluating alternative providers and define migration procedures to minimize disruption to operations (Menzel and Ranjan. 2012). Negotiate contractual agreements with cloud providers that include provisions for data portability, interoperability, and vendor lock-in mitigation. Ensure that contracts specify terms for service-level agreements (SLAs), pricing, and exit clauses to protect the organization's interests (Goo. 2010). By adopting these strategies, organizations can reduce the risks associated with vendor lock-in in serverless computing environments and maintain greater flexibility and control over their technology infrastructure (Zhao. 2022). These measures help safeguard against dependency on a single cloud provider and enable organizations to adapt to changing business requirements and market dynamics effectively.

4. CONCLUSIONS

Serverless computing offers numerous benefits, including scalability, cost-efficiency, and reduced time to market. However, it also presents significant challenges, particularly in the realm of network security (Patel. 2024). Throughout this review, we have identified and discussed various security challenges faced by organizations adopting serverless computing, including limited visibility and control, insecure dependencies, cold start attacks, data privacy concerns, and resource exhaustion.

To address these challenges, organizations must implement targeted solutions and best practices. These solutions include enhancing visibility and control through logging and monitoring mechanisms, implementing encryption and authentication measures to secure data and access, mitigating cold start attacks through warm-up mechanisms and strategic placement, and employing rate-limiting and auto-scaling policies to manage resource consumption effectively. Additionally, adopting a multi-cloud strategy, prioritizing standardization and portability, containerizing applications, and negotiating vendor-neutral contractual agreements can help mitigate the risks of vendor lock-in and enhance flexibility in managing serverless environments.

The importance of advanced technologies such as AI and blockchain cannot be overstated in addressing the emerging challenges in serverless computing security. AI-powered anomaly detection systems can help organizations detect and respond to security threats in real-time by analyzing large volumes of data and identifying patterns indicative of suspicious activity (Rangaraju. 2023). Additionally, blockchain technology offers a decentralized and immutable ledger that can enhance the security and integrity of data transactions in serverless environments.

By leveraging these advanced technologies, organizations can enhance their ability to anticipate, detect, and mitigate security threats in serverless computing systems. Moreover, the findings of this research study provide valuable insights for future researchers, serving as a foundation for the development of innovative strategies and solutions to further enhance the security and protection of serverless computing systems. By continuing to explore and advance the application of AI, blockchain, and other emerging technologies in serverless security, organizations can stay ahead of evolving threats and ensure the integrity and resilience of their serverless environments.

5. FUTURE SCOPE

The future scope of serverless computing security is vast and holds significant potential for advancements and innovations. Continued research and development are needed to enhance security measures in serverless computing, including improved authentication mechanisms, stronger encryption techniques, and more effective anomaly detection systems. Leveraging AI and machine learning algorithms can further enhance serverless security by enabling proactive threat detection, automated response mechanisms, and intelligent risk mitigation strategies. Exploring the integration of blockchain technology into serverless computing environments can provide enhanced data integrity, immutability, and transparency, offering additional layers of security and trust. With growing concerns over data privacy, there is a need to develop and implement privacy-preserving technologies such as differential privacy and secure multi-party computation in serverless computing systems. Standardization efforts can help promote interoperability and compatibility among different serverless platforms, enabling seamless migration of applications and data across multiple cloud providers while maintaining security and compliance.

As regulatory requirements evolve, there is a need for serverless computing solutions that facilitate compliance with data protection regulations such as GDPR, HIPAA, and CCPA, ensuring the security and privacy of sensitive information. Increasing cybersecurity education and awareness among developers, administrators, and end-users is essential for fostering a culture of security and mitigating human error-related security risks in serverless computing environments. Addressing ethical considerations surrounding serverless computing security, such as algorithmic bias, data privacy, and accountability, is crucial for ensuring responsible and ethical deployment of serverless technologies. Overall, the future of serverless computing security lies in continuous innovation, collaboration, and adaptation to emerging threats and challenges. By embracing cutting-edge technologies, fostering collaboration among industry stakeholders, and prioritizing security-by-design principles, the serverless computing ecosystem can evolve to meet the evolving security needs of organizations and users.

References

[1] Aad, I., Hubaux, J. P., & Knightly, E. W. (2008). Impact of denial of service attacks on ad hoc networks. IEEE/ACM transactions on networking, 16(4), 791-802.

[2] Adeniyi, E. A., Ogundokun, R. O., Misra, S., Awotunde, J. B., & Abiodun, K. M. (2022). Enhanced security and privacy issue in multi-tenant environment of green computing using blockchain technology. In Blockchain Applications in the Smart Era (pp. 65-83). Cham: Springer International Publishing.

[3] Aditya, P., Akkus, I. E., Beck, A., Chen, R., Hilt, V., Rimac, I., … & Stein, M. (2019). Will serverless computing revolutionize NFV?. Proceedings of the IEEE, 107(4), 667-678.

[4] Agache, A., Brooker, M., Iordache, A., Liguori, A., Neugebauer, R., Piwonka, P., & Popa, D. M. (2020). Firecracker: Lightweight virtualization for serverless applications. In 17th USENIX symposium on networked systems design and implementation (NSDI 20) (pp. 419-434).

[5] Ahmad, S., Mehfuz, S., Urooj, S., & Alsubaie, N. (2024). Machine learning-based intelligent security framework for secure cloud key management. *Cluster Computing*, 1-27.

[6] Ahmadi, S. (2024). Challenges and Solutions in Network Security for Serverless Computing. International Journal of Current Science Research and Review, 7(01), 218-229.

[7] Albanese, M., Battista, E., Jajodia, S., & Casola, V. (2014, October). Manipulating the attacker's view of a system's attack surface. In 2014 IEEE Conference on Communications and Network Security (pp. 472-480). IEEE.

[8] Ali, S. A. (2023). DESIGINING SECURE AND ROBUST E-COMMERCE PLAFORM FOR PUBLIC CLOUD. *The Asian Bulletin of Big Data Management*, 3(1).

[9] Alpernas, K., Panda, A., Ryzhyk, L., & Sagiv, M. (2021, November). Cloud-scale runtime verification of serverless applications. In Proceedings of the ACM Symposium on Cloud Computing (pp. 92-107).

[10] Alwan, Z. S., & Younis, M. F. (2017). Detection and prevention of SQL injection attack: a survey. International Journal of Computer Science and Mobile Computing, 6(8), 5-17.

[11] Aravindhan, K., & Karthiga, R. R. (2013). One time password: A survey. International Journal of Emerging Trends in Engineering and Development, 1(3), 613-623.

[12] Ariffin, M. A. M., Ibrahim, M. F., & Kasiran, Z. (2020). API vulnerabilities in cloud computing platform: attack and detection. International Journal of Engineering Trends and Technology, 1, 8-14.

[13] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. H., Konwinski, A., … & Zaharia, M. (2009). Above the clouds: A berkeley view of cloud computing (Vol. 17). Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley.

[14] Ang'udi, J. J. (2023). Security challenges in cloud computing: A comprehensive analysis. World Journal of Advanced Engineering Technology and Sciences, 10(2), 155-181.

[15] Asad, M. (2023). Security Management in Cloud Computing for healthcare Data.

[16] Aslanpour, M. S., Toosi, A. N., Cicconetti, C., Javadi, B., Sbarski, P., Taibi, D., … & Dustdar, S. (2021, February). Serverless edge computing: vision and challenges. In Proceedings of the 2021 Australasian computer science week multiconference (pp. 1-10).

[17] Aslan, Ö., Aktuğ, S. S., Ozkan-Okay, M., Yilmaz, A. A., & Akin, E. (2023). A comprehensive review of cyber security vulnerabilities, threats, attacks, and solutions. Electronics, 12(6), 1333.

[18] Aurangzeb, S., Aleem, M., Iqbal, M. A., & Islam, M. A. (2017). Ransomware: a survey and trends. J. Inf. Assur. Secur, 6(2), 48-58.

[19] Backes, M., Bugiel, S., & Derr, E. (2016, October). Reliable third-party library detection in android and its security applications. In Proceedings of the 2016 ACM SIGSAC conference on computer and communications security (pp. 356-367).

[20] Baldini, I., Cheng, P., Fink, S. J., Mitchell, N., Muthusamy, V., Rabbah, R., ... & Tardieu, O. (2017, October). The serverless trilemma: Function composition for serverless computing. In *Proceedings of the 2017 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software* (pp. 89-103).

[21] Barcelona-Pons, D., Sutra, P., Sánchez-Artigas, M., París, G., & García-López, P. (2022). Stateful serverless computing with crucial. ACM Transactions on Software Engineering and Methodology (TOSEM), 31(3), 1-38.

[22] Baresi, L., & Quattrocchi, G. (2021). PAPS: A serverless platform for edge computing infrastructures. Frontiers in Sustainable Cities, 3, 690660.

[23] Basak, A., Bhunia, S., & Ray, S. (2016, June). Exploiting design-for-debug for flexible SoC security architecture. In Proceedings of the 53rd Annual Design Automation Conference (pp. 1-6).

[24] Benedetti, P., Femminella, M., Reali, G., & Steenhaut, K. (2022, March). Reinforcement learning applicability for resource-based auto-scaling in serverless edge applications. In 2022 IEEE international conference on pervasive computing and communications workshops and other affiliated events (PerCom Workshops) (pp. 674-679). IEEE.

[25] Berlin, K., Slater, D., & Saxe, J. (2015, October). Malicious behavior detection using windows audit logs. In Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security (pp. 35-44).

[26] Bhatt, S., Manadhata, P. K., & Zomlot, L. (2014). The operational role of security information and event management systems. IEEE security & Privacy, 12(5), 35-41.

[27] Bonguet, A., & Bellaiche, M. (2017). A survey of denial-of-service and distributed denial of service attacks and defenses in cloud computing. Future Internet, 9(3), 43.

[28] Boscain, S. (2023). AWS Cloud: Infrastructure, DevOps techniques, State of Art (Doctoral dissertation, Politecnico di Torino).

[29] Boza, E. F., Abad, C. L., Villavicencio, M., Quimba, S., & Plaza, J. A. (2017, October). Reserved, on demand or serverless: Model-based simulations for cloud budget planning. In 2017 IEEE second Ecuador technical chapters meeting (ETCM) (pp. 1-6). IEEE.

[30] Brown, A. W., Delbaere, M., Eeles, P., Johnston, S., & Weaver, R. (2005). Realizing service-oriented solutions with the IBM rational software development platform. IBM systems journal, 44(4), 727-752.

[31] Buennemeyer, T. K., Nelson, T. M., Clagett, L. M., Dunning, J. P., Marchany, R. C., & Tront, J. G. (2008, January). Mobile device profiling and intrusion detection using smart batteries. In Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008) (pp. 296-296). IEEE.

[32] Butin, D., & Le Métayer, D. (2015, May). A guide to end-to-end privacy accountability. In 2015 IEEE/ACM 1st International Workshop on Technical and Legal aspects of data pRivacy and Security (pp. 20-25). IEEE.

[33] Candel, J. M. O., Elouali, A., Gimeno, F. J. M., & Mora, H. (2022, November). Cloud vs Serverless Computing: A Security Point of View. In *International Conference on Ubiquitous Computing and Ambient Intelligence* (pp. 1098-1109). Cham: Springer International Publishing.

[34] Castro, P., Isahagian, V., Muthusamy, V., & Slominski, A. (2023). Hybrid serverless computing: Opportunities and challenges. *Serverless Computing: Principles and Paradigms*, 43-77.

[35] Cassel, G. A. S., Rodrigues, V. F., da Rosa Righi, R., Bez, M. R., Nepomuceno, A. C., & da Costa, C. A. (2022). Serverless computing for Internet of Things: A systematic literature review. Future Generation Computer Systems, 128, 299-316.

[36] Catalfamo, A. (2023). How to strengthen Cloud Computing exploiting solutions to Edge. Universitá degli Studi di Messina.

[37] Chase, J. S., Anderson, D. C., Thakar, P. N., Vahdat, A. M., & Doyle, R. P. (2001). Managing energy and server resources in hosting centers. ACM SIGOPS operating systems review, 35(5), 103-116.

[38] Chen, Z., Xu, G., Mahalingam, V., Ge, L., Nguyen, J., Yu, W., & Lu, C. (2016). A cloud computing based network monitoring and threat detection system for critical infrastructures. Big Data Research, 3, 10-23.

[39] Chopra, A. K., & Singh, M. P. (2021, September). Deserv: Decentralized serverless computing. In 2021 IEEE International Conference on Web Services (ICWS) (pp. 51-60). IEEE.

[40] Chostak, C. (2020). Machine Learning Powered Serverless Fraud Detection (Master's thesis, Instituto Politecnico do Porto (Portugal)).

[41] Chowhan, R. S., & Tanwar, R. (2019). Password-less authentication: methods for user verification and identification to login securely over remote sites. In *Machine Learning and Cognitive Science Applications in Cyber Security* (pp. 190-212). IGI global.

[42] Christensen, J. H. (2009, October). Using RESTful web-services and cloud computing to create next generation mobile applications. In Proceedings of the 24th ACM

SIGPLAN conference companion on Object oriented programming systems languages and applications (pp. 627-634).

[43] Christidis, A., Moschoyiannis, S., Hsu, C. H., & Davies, R. (2020). Enabling serverless deployment of large-scale ai workloads. IEEE Access, 8, 70150-70161.

[44] Cicconetti, C., Conti, M., & Passarella, A. (2020). A decentralized framework for serverless edge computing in the internet of things. IEEE Transactions on Network and Service Management, 18(2), 2166-2180.

[45] Cinar, B. (2023). The Rise of Serverless Architectures: Security Challenges and Best Practices. Asian Journal of Research in Computer Science, 16(4), 194-210.

[46] Cobleigh, A., Hell, M., Karlsson, L., Reimer, O., Sönnerup, J., & Wisenhoff, D. (2018, October). Identifying, prioritizing and evaluating vulnerabilities in third party code. In 2018 IEEE 22nd International Enterprise Distributed Object Computing Workshop (EDOCW) (pp. 208-211). IEEE.

[47] Costa, R., & Hodun, D. (2021). Google Cloud Cookbook. " O'Reilly Media, Inc.".

[48] Datta, P., Kumar, P., Morris, T., Grace, M., Rahmati, A., & Bates, A. (2020, April). Valve: Securing function workflows on serverless computing platforms. In Proceedings of The Web Conference 2020 (pp. 939-950).

[49] Damiani, E., De Capitani di Vimercati, S., Paraboschi, S., & Samarati, P. (2002). A fine-grained access control system for XML documents. ACM Transactions on Information and System Security (TISSEC), 5(2), 169-202.

[50] Decan, A., Mens, T., & Constantinou, E. (2018, May). On the impact of security vulnerabilities in the npm package dependency network. In Proceedings of the 15th international conference on mining software repositories (pp. 181-191).

[51] Devi, B. K., & Subbulakshmi, T. (2017, December). DDoS attack detection and mitigation techniques in cloud computing environment. In 2017 International Conference on Intelligent Sustainable Systems (ICISS) (pp. 512-517). IEEE.

[52] Dramé-Maigné, S., Laurent, M., Castillo, L., & Ganem, H. (2021). Centralized, distributed, and everything in between: Reviewing access control solutions for the IoT. ACM Computing Surveys (CSUR), 54(7), 1-34.

[53] Duggineni, S. (2023). Impact of controls on data integrity and information systems. Science and Technology, 13(2), 29-35.

[54] Eassa, A. M., Elhoseny, M., El-Bakry, H. M., & Salama, A. S. (2018). NoSQL injection attack detection in web applications using RESTful service. Programming and Computer Software, 44, 435-444.

[55] Efterpi, P. (2020). Machine Learning Pipelines in Serverless Environments (Doctoral dissertation, University of Piraeus (Greece)).

[56] Eismann, S., Scheuner, J., Van Eyk, E., Schwinger, M., Grohmann, J., Herbst, N., ... & Iosup, A. (2020). A review of serverless use cases and their characteristics. *arXiv preprint arXiv:2008.11110*.

[57] Enes, J., Expósito, R. R., & Touriño, J. (2020). Real-time resource scaling platform for big data workloads on serverless environments. Future Generation Computer Systems, 105, 361-379.

[58] Fairhurst, G. (2017). Network Transport Circuit Breakers (No. rfc8084). Internet Engineering Task Force.

[59] Farahmand, F., Navathe, S. B., Sharp, G. P., & Enslow, P. H. (2005). A management perspective on risk of security threats to information systems. Information Technology and Management, 6, 203-225.

[60] García-López, P., Sánchez-Artigas, M., Shillaker, S., Pietzuch, P., Breitgand, D., Vernik, G., … & Ferrer, A. J. (2019). Servermix: Tradeoffs and challenges of serverless data analytics. arXiv preprint arXiv:1907.11465.

[61] Genaud, S., & Gossa, J. (2011, July). Cost-wait trade-offs in client-side resource provisioning with elastic clouds. In 2011 IEEE 4th International Conference on Cloud Computing (pp. 1-8). IEEE.

[62] George, A. S., Sagayarajan, S., Baskar, T., & George, A. H. (2023). Extending detection and response: how MXDR evolves cybersecurity. Partners Universal International Innovation Journal, 1(4), 268-285.

[63] Ghorbani, A. A., Lu, W., & Tavallaee, M. (2009). Network intrusion detection and prevention: concepts and techniques (Vol. 47). Springer Science & Business Media.

[64] Gill, S. S., Xu, M., Ottaviani, C., Patros, P., Bahsoon, R., Shaghaghi, A., … & Uhlig, S. (2022). AI for next generation computing: Emerging trends and future directions. Internet of Things, 19, 100514.

[65] Gjerdrum, A. T. (2020). Diggi: A Distributed Serverless Runtime for Developing Trusted Cloud Services.

[66] Glaessner, T. C., Kellermann, T., & McNevin, V. (2002). Electronic Security: Risk Mitigation in Financial Transactions: Public Policy Issues (Vol. 2870). World Bank Publications.

[67] Golec, M., Ozturac, R., Pooranian, Z., Gill, S. S., & Buyya, R. (2021). IFaaSBus: A security-and privacy-based lightweight framework for serverless computing using IoT and machine learning. IEEE Transactions on Industrial Informatics, 18(5), 3522-3529.

[68] Golec, M., Walia, G. K., Kumar, M., Cuadrado, F., Gill, S. S., & Uhlig, S. (2023). Cold start latency in serverless computing: A systematic review, taxonomy, and future directions. arXiv preprint arXiv:2310.08437.

[69] Goo, J. (2010). Structure of service level agreements (SLA) in IT outsourcing: The construct and its measurement. Information Systems Frontiers, 12, 185-205.

[70] Gouareb, R., Friderikos, V., & Aghvami, A. H. (2018). Virtual network functions routing and placement for edge cloud latency minimization. IEEE Journal on Selected Areas in Communications, 36(10), 2346-2357.

[71] Gowri, A. S., Bala, P. S., & Ramdinthara, I. Z. (2021). Comprehensive analysis of resource allocation and service placement in fog and cloud computing. International Journal of Advanced Computer Science and Applications, 12(3).

[72] Goyal, V., Pandey, O., Sahai, A., & Waters, B. (2006, October). Attribute-based encryption for fine-grained access control of encrypted data. In Proceedings of the 13th ACM conference on Computer and communications security (pp. 89-98).

[73] Gulati, S. S., & Gupta, S. (2012). A framework for enhancing security and performance in multi-tenant applications. International Journal of Information Technology and Knowledge Management, 5(2), 233-237.

[74] Gunasekaran, A., Patel, C., & McGaughey, R. E. (2004). A framework for supply chain performance measurement. International journal of production economics, 87(3), 333-347.

[75] Guo, Z., Blanco, Z., Shahrad, M., Wei, Z., Dong, B., Li, J., ... & Zhang, Y. (2022). Decomposing and executing serverless applications as resource graphs. arXiv preprint arXiv:2206.13444.

[76] Gupta, S., & Gupta, B. B. (2017). Cross-Site Scripting (XSS) attacks and defense mechanisms: classification and state-of-the-art. International Journal of System Assurance Engineering and Management, 8, 512-530.

[77] Haber, M. J., Chappell, B., & Hills, C. (2022). Mitigation Strategies. In Cloud Attack Vectors: Building Effective Cyber-Defense Strategies to Protect Cloud Resources (pp. 221-296). Berkeley, CA: Apress.

[78] Halderman, J. A., Schoen, S. D., Heninger, N., Clarkson, W., Paul, W., Calandrino, J. A., … & Felten, E. W. (2009). Lest we remember: cold-boot attacks on encryption keys. Communications of the ACM, 52(5), 91-98.

[79] Hamilton, J. R. (2007, November). On Designing and Deploying Internet-Scale Services. In *LISA* (Vol. 18, No. 2007, pp. 1-18).

[80] Hassan, H. B., Barakat, S. A., & Sarhan, Q. I. (2021). Survey on serverless computing. Journal of Cloud Computing, 10, 1-29.

[81] Harsh, P., Dudouet, F., Cascella, R. G., Jégou, Y., & Morin, C. (2012, October). Using open standards for interoperability issues, solutions, and challenges facing cloud computing. In 2012 8th international conference on network and service management (cnsm) and 2012 workshop on systems virtualiztion management (svm) (pp. 435-440). IEEE.

[82] Heckathorn, M. (2011). Network monitoring for web-based threats. Network, 2, 1-2011.

[83] Hellerstein, J. M., Faleiro, J., Gonzalez, J. E., Schleier-Smith, J., Sreekanti, V., Tumanov, A., & Wu, C. (2018). Serverless computing: One step forward, two steps back. arXiv preprint arXiv:1812.03651.

[84] Hilley, D. (2009). Cloud computing: A taxonomy of platform and infrastructure-level offerings. Georgia Institute of Technology, Tech. Rep, 44-45.

[85] Hobday, R. A., & Dancer, S. J. (2013). Roles of sunlight and natural ventilation for controlling infection: historical and current perspectives. Journal of hospital infection, 84(4), 271-282.

[86] Humphreys, E. (2008). Information security management standards: Compliance, governance and risk management. information security technical report, 13(4), 247-255.

[87] Huseinović, A., Mrdović, S., Bicakci, K., & Uludag, S. (2020). A survey of denial-of-service attacks and solutions in the smart grid. IEEE Access, 8, 177447-177470.

[88] Inamdar, D. M., & Gupta, S. (2020). A survey on web application security. International Journal of Scientific Research in Computer Science, Engineering and Information Technology,(6), 223-228.

[89] Jacob, S. (2022). Enhancing cyber attack prevention and detection using application process tracing. Technological University of the Shannon: Midlands Midwest

[90] Jangda, A., Pinckney, D., Brun, Y., & Guha, A. (2019). Formal foundations of serverless computing. Proceedings of the ACM on Programming Languages, 3(OOPSLA), 1-26.

[91] Javed, H., Toosi, A. N., & Aslanpour, M. S. (2022). Serverless platforms on the edge: a performance analysis. In New Frontiers in Cloud Computing and Internet of Things (pp. 165-184). Cham: Springer International Publishing.

[92] Jegan, D. S., Wang, L., Bhagat, S., Ristenpart, T., & Swift, M. (2020). Guarding serverless applications with seclambda. arXiv preprint arXiv:2011.05322.

[93] Jegan, D. S., Wang, L., Bhagat, S., & Swift, M. (2023). Guarding serverless applications with Kalium. In 32nd USENIX Security Symposium (USENIX Security 23) (pp. 4087-4104).

[94] Jegannathan, A. P., Saha, R., & Addya, S. K. (2022, June). A time series forecasting approach to minimize cold start time in cloud-serverless platform. In 2022 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom) (pp. 325-330). IEEE.

[95] Jonas, E., Schleier-Smith, J., Sreekanti, V., Tsai, C. C., Khandelwal, A., Pu, Q., ... & Patterson, D. A. (2019). Cloud programming simplified: A berkeley view on serverless computing. arXiv preprint arXiv:1902.03383.

[96] Katal, A., Sethi, V., & Lamba, S. (2021). Virtual Machine Scaling in Autonomic Cloud Resource Management. Autonomic Computing in Cloud Resource Management in Industry 4.0, 301-323.

[97] Kelly, D., Glavin, F., & Barrett, E. (2020, October). Serverless computing: Behind the scenes of major platforms. In 2020 IEEE 13th International Conference on Cloud Computing (CLOUD) (pp. 304-312). IEEE.

[98] Kesa, D. M. (2023). Ensuring resilience: Integrating IT disaster recovery planning and business continuity for sustainable information technology operations. World Journal of Advanced Research and Reviews, 18(3), 970-992.

[99] Khan, S., Parkinson, S., & Qin, Y. (2017). Fog computing security: a review of current applications and security solutions. Journal of Cloud Computing, 6, 1-22.

[100] Kimani, K., Oduol, V., & Langat, K. (2019). Cyber security challenges for IoT-based smart grid networks. International journal of critical infrastructure protection, 25, 36-49.

[101] Kitchin, R., & Dodge, M. (2020). The (in) security of smart cities: Vulnerabilities, risks, mitigation, and prevention. In *Smart cities and innovative Urban technologies* (pp. 47-65). Routledge.

[102] Koschel, A., Klassen, S., Jdiya, K., Schaaf, M., & Astrova, I. (2021, July). Cloud computing: serverless. In 2021 12th International Conference on Information, Intelligence, Systems & Applications (IISA) (pp. 1-7). IEEE.

[103] Kumar, R., & Goyal, R. (2019). On cloud security requirements, threats, vulnerabilities and countermeasures: A survey. Computer Science Review, 33, 1-48.

[104] Kumar, M. (2019). Serverless architectures review, future trend and the solutions to open problems. American Journal of Software Engineering, 6(1), 1-10.

[105] Kumari, A., Patra, M. K., & Sahoo, B. (2023). Data Controlling and Security Issues in Cloud: A Step Towards Serverless. In Perspectives on Social Welfare Applications' Optimization and Enhanced Computer Applications (pp. 105-124). IGI Global.

[106] Lannurien, V., D'orazio, L., Barais, O., & Boukhobza, J. (2023). Serverless Cloud Computing: State of the Art and Challenges. Serverless Computing: Principles and Paradigms, 275-316.

[107] Lebanon, G., El-Geish, M., Lebanon, G., & El-Geish, M. (2018). Thoughts on System Design for Big Data. Computing with Data: An Introduction to the Data Industry, 495-541.

[108] Leemans, T. T. (2022). Towards Serverless Enterprises: Developing the Enterprise Serverless Assessment (ESA) to assess and improve an organization's fit and readiness for Serverless technology (Master's thesis, University of Twente).

[109] Li, Z., Guo, L., Cheng, J., Chen, Q., He, B., & Guo, M. (2022). The serverless computing survey: A technical primer for design architecture. ACM Computing Surveys (CSUR), 54(10s), 1-34.

[110] Loaiza Enriquez, R. (2021). Cloud Security Posture Management/CSPM) in Azure.

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

[112] Makani, S. T. (2023). STREAMLINING AWS SERVERLESS DEPLOYMENTS: A BITBUCKET PIPELINE APPROACH. International Journal of Computer Applications (IJCA), 4(1).

[113] Malawski, M., Gajek, A., Zima, A., Balis, B., & Figiela, K. (2020). Serverless execution of scientific workflows: Experiments with hyperflow, aws lambda and google cloud functions. Future Generation Computer Systems, 110, 502-514.

[114] Malik, M. I., Wani, S. H., & Rashid, A. (2018). CLOUD COMPUTING-TECHNOLOGIES. *International Journal of Advanced Research in Computer Science*, 9(2).

[115] Malik, A. W., Bhatti, D. S., Park, T. J., Ishtiaq, H. U., Ryou, J. C., & Kim, K. I. (2024). Cloud Digital Forensics: Beyond Tools, Techniques, and Challenges. Sensors, 24(2), 433.

[116] Mallick, M. A. I., & Nath, R. (2024).Navigating the Cyber security Landscape: A Comprehensive Review of Cyber-Attacks, Emerging Trends, and Recent Developments. World Scientific News, 190(1), 1-69.

[117] Mampage, A., Karunasekera, S., & Buyya, R. (2021, May). Deadline-aware dynamic resource management in serverless computing environments. In 2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid) (pp. 483-492). IEEE.

[118] Mampage, A., Karunasekera, S., & Buyya, R. (2022). A holistic view on resource management in serverless computing environments: Taxonomy and future directions. ACM Computing Surveys (CSUR), 54(11s), 1-36.

[119] Mannan, M., & van Oorschot, P. C. (2011). Leveraging personal devices for stronger password authentication from untrusted computers. Journal of Computer Security, 19(4), 703-750.

[120] Manvi, S. S., & Shyam, G. K. (2014). Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey. *Journal of network and computer applications*, *41*, 424-440.

[121] Mateus-Coelho, N., & Cruz-Cunha, M. (2022, June). Serverless service architectures and security minimals. In 2022 10th International Symposium on Digital Forensics and Security (ISDFS) (pp. 1-6). IEEE.

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

[123] Menzel, M., & Ranjan, R. (2012, April). CloudGenius: decision support for web server cloud migration. In Proceedings of the 21st international conference on World Wide Web (pp. 979-988).

[124] Mitropoulos, D., & Spinellis, D. (2017). Fatal injection: A survey of modern code injection attack countermeasures. PeerJ Computer Science, 3, e136.

[125] Mlyatu, M. M., & Sanga, C. (2022). Secure web application technologies implementation through hardening security headers using automated threat modelling techniques. Journal of Information Security, 14(1), 1-15.

[126] Mohd Nazir, M. A. N. (2011). Cost-effective resource management for distributed computing (Doctoral dissertation, UCL (University College London).

[127] Mondal, S. K., Pan, R., Kabir, H. D., Tian, T., & Dai, H. N. (2022). Kubernetes in IT administration and serverless computing: An empirical study and research challenges. The Journal of Supercomputing, 1-51.

[128] Morrow, B. (2012). BYOD security challenges: control and protect your most sensitive data. Network Security, 2012(12), 5-8.

[129] Mughal, A. A. (2018). The Art of Cybersecurity: Defense in Depth Strategy for Robust Protection. *International Journal of Intelligent Automation and Computing*, *1*(1), 1-20.

[130] Mughal, A. A. (2019). Cybersecurity Hygiene in the Era of Internet of Things (IoT): Best Practices and Challenges. Applied Research in Artificial Intelligence and Cloud Computing, 2(1), 1-31.

[131] Mushtaq, M. F., Akram, U., Khan, I., Khan, S. N., Shahzad, A., & Ullah, A. (2017). Cloud computing environment and security challenges: A review. International Journal of Advanced Computer Science and Applications, 8(10).

[132] Mustafa, G., Ashraf, R., Mirza, M. A., Jamil, A., & Muhammad. (2018, June). A review of data security and cryptographic techniques in IoT based devices. In Proceedings of the 2nd International Conference on Future Networks and Distributed Systems (pp. 1-9).

[133] Nadji, Y., Saxena, P., & Song, D. (2009, February). Document Structure Integrity: A Robust Basis for Cross-site Scripting Defense. In NDSS (Vol. 20).

[134] Nastic, S., & Dustdar, S. (2018). Towards deviceless edge computing: Challenges, design aspects, and models for serverless paradigm at the edge. The Essence of Software Engineering, 121-136.

[135] Ngo, C., Wang, P., Tran, T., & Chung, S. (2020, July). Serverless computing architecture security and quality analysis for back-end development. In Journal of The Colloquium for Information Systems Security Education (Vol. 7, No. 1, pp. 8-8).

[136] Naranjo Rico, J. L. (2018). Holistic business approach for the protection of sensitive data: study of legal requirements and regulatory compliance at international level to define and implement data protection measures using encryption techniques.

[137] Narasayya, V., & Chaudhuri, S. (2021). Cloud data services: Workloads, architectures and multi-tenancy. Foundations and Trends® in Databases, 10(1), 1-107

[138] Olaoye, G., & Luz, A. (2024). Future trends and emerging technologies in cloud security. Telecommunication Engineering Centre. University of Melbourne

[139] O'Meara, W., & Lennon, R. G. (2020, June). Serverless computing security: Protecting application logic. In 2020 31st Irish Signals and Systems Conference (ISSC) (pp. 1-5). IEEE.

[140] Omotunde, H., & Ahmed, M. (2023). A Comprehensive Review of Security Measures in Database Systems: Assessing Authentication, Access Control, and Beyond. Mesopotamian Journal of CyberSecurity, 2023, 115-133.

[141] Orazi, G., Vallittu, K., Sainio, P., & Virtanen, S. (2020). Enhancing and integration of security testing in the development of a microservices environment.

[142] Ortega-Fernandez, I., & Liberati, F. (2023). A review of denial of service attack and mitigation in the smart grid using reinforcement learning. Energies, 16(2), 635.

[143] Ouyang, R., Wang, J., Xu, H., Chen, S., Xiong, X., Tolba, A., & Zhang, X. (2023). A Microservice and Serverless Architecture for Secure IoT System. Sensors, 23(10), 4868.

[144] Padma, P., & Srinivasan, S. (2023). DAuth—Delegated Authorization Framework for Secured Serverless Cloud Computing. Wireless Personal Communications, 129(3), 1563-1583.

[145] Pal, P. (2022). The adoption of waves of digital technology as antecedents of digital transformation by financial services institutions. *Journal of Digital Banking*, 7(1), 70-91.

[146] Pan, Y., Sun, F., Teng, Z., White, J., Schmidt, D. C., Staples, J., & Krause, L. (2019). Detecting web attacks with end-to-end deep learning. Journal of Internet Services and Applications, 10(1), 1-22.

[147] Pan, S., Zhao, H., Cai, Z., Li, D., Ma, R., & Guan, H. (2023). Sustainable serverless computing with cold-start optimization and automatic workflow resource scheduling. IEEE Transactions on Sustainable Computing.

[148] Papadopoulos, P., Ilia, P., Polychronakis, M., Markatos, E. P., Ioannidis, S., & Vasiliadis, G. (2018). Master of web puppets: Abusing web browsers for persistent and stealthy computation. arXiv preprint arXiv:1810.00464.

[149] Papp, D., Ma, Z., & Buttyan, L. (2015, July). Embedded systems security: Threats, vulnerabilities, and attack taxonomy. In 2015 13th Annual Conference on Privacy, Security and Trust (PST) (pp. 145-152). Ieee.

[150] Park, J., & Jeong, J. (2023). An Autoscaling System Based on Predicting the Demand for Resources and Responding to Failure in Forecasting. Sensors, 23(23), 9436.

[151] Pascoal, T. A., Fonseca, I. E., & Nigam, V. (2020). Slow denial-of-service attacks on software defined networks. Computer Networks, 173, 107223.

[152] Patel, K. (2024). Mastering Cloud Scalability: Strategies, Challenges, and Future Directions: Navigating Complexities of Scaling in Digital Era. In Emerging Trends in Cloud Computing Analytics, Scalability, and Service Models (pp. 155-169). IGI Global.

[153] Patidar, S., Rane, D., & Jain, P. (2011, December). Challenges of software development on cloud platform. In 2011 World Congress on Information and Communication Technologies (pp. 1009-1013). IEEE.

[154] Patros, P., Spillner, J., Papadopoulos, A. V., Varghese, B., Rana, O., & Dustdar, S. (2021). Toward sustainable serverless computing. *IEEE Internet Computing*, *25*(6), 42-50.

[155] Patwary, A. A. N., Fu, A., Naha, R. K., Battula, S. K., Garg, S., Patwary, M. A. K., & Aghasian, E. (2020). Authentication, access control, privacy, threats and trust management towards securing fog computing environments: A review. arXiv preprint arXiv:2003.00395.

[156] Patel, A., Taghavi, M., Bakhtiyari, K., & Júnior, J. C. (2013). An intrusion detection and prevention system in cloud computing: A systematic review. Journal of network and computer applications, 36(1), 25-41.

[157] Pearce, M., Zeadally, S., & Hunt, R. (2013). Virtualization: Issues, security threats, and solutions. ACM Computing Surveys (CSUR), 45(2), 1-39.

[158] Pearson, S., & Benameur, A. (2010, November). Privacy, security and trust issues arising from cloud computing. In 2010 IEEE Second International Conference on Cloud Computing Technology and Science (pp. 693-702). IEEE.

[159] Podjarny, G., & Ţal, L. (2019). Serverless security. O'Reilly Media, Incorporated.

[160] Poeplau, S., Fratantonio, Y., Bianchi, A., Kruegel, C., & Vigna, G. (2014, February). Execute this! Analyzing unsafe and malicious dynamic code loading in android applications. In NDSS (Vol. 14, pp. 23-26).

[161] Polinsky, I., Datta, P., Bates, A., & Enck, W. (2021, June). SCIFFS: Enabling secure third-party security analytics using serverless computing. In Proceedings of the 26th ACM Symposium on Access Control Models and Technologies (pp. 175-186).

[162] Ponta, S. E., Plate, H., & Sabetta, A. (2020). Detection, assessment and mitigation of vulnerabilities in open source dependencies. Empirical Software Engineering, 25(5), 3175-3215.

[163] Prakash, A. A., & Kumar, K. S. (2022). Cloud serverless security and services: a survey. In Applications of Computational Methods in Manufacturing and Product Design: Select Proceedings of IPDIMS 2020 (pp. 453-462). Singapore: Springer Nature Singapore.

[164] Prasad, K. M., Reddy, A. R. M., & Rao, K. V. (2014). DoS and DDoS attacks: defense, detection and traceback mechanisms-a survey. Global Journal of Computer Science and Technology, 14(7-E), 15.

[165] Pusuluri, V. S. R. (2022). Taxonomy Of Security and Privacy Issues in Serverless Computing. St. Cloud State University.

[166] Qazi, F. A. (2022). Insecure Application Programming Interfaces (APIs) in Zero-Trust Networks (Doctoral dissertation, Capitol Technology University).

[167] Rawal, B. S., Manogaran, G., & Peter, A. (2023). Cybersecurity and Identity Access Management. Springer.

[168] Reddy, V. K., Rao, B. T., Reddy, L. S. S., & Kiran, P. S. (2011). Research issues in cloud computing. *Global Journal of Computer Science and Technology*, *11*(11), 59-64.

[169] Rinta-Jaskari, E. (2021). Automatic Testing Approaches For Serverless Applications In AWS (Master's thesis).

[170] Rios, V. D. M., Inacio, P. R., Magoni, D., & Freire, M. M. (2022). Detection and mitigation of low-rate denial-of-service attacks: A survey. IEEE Access, 10, 76648-76668.

[171] Rodriguez, M. A., & Buyya, R. (2019). Container-based cluster orchestration systems: A taxonomy and future directions. Software: Practice and Experience, 49(5), 698-719.

[172] Rodrigues, P., Freitas, F., & Simão, J. (2022). Quickfaas: Providing portability and interoperability between faas platforms. Future Internet, 14(12), 360.

[173] Roy, R. B., Patel, T., & Tiwari, D. (2022, February). Icebreaker: Warming serverless functions better with heterogeneity. In Proceedings of the 27th ACM International

Conference on Architectural Support for Programming Languages and Operating Systems (pp. 753-767).

[174] Rubinstein, I. S., & Hartzog, W. (2016). Anonymization and risk. Wash. L. Rev. 91, 703.

[175] Saad, M., Spaulding, J., Njilla, L., Kamhoua, C., Shetty, S., Nyang, D., & Mohaisen, D. (2020). Exploring the attack surface of blockchain: A comprehensive survey. IEEE Communications Surveys & Tutorials, 22(3), 1977-2008.

[176] Sabbioni, A. (2023). Serverless middlewares to integrate heterogeneous and distributed services in cloud continuum environments.

[177] Sabbioni, A., Mazzocca, C., Colajanni, M., Montanari, R., & Corradi, A. (2022, June). A fully decentralized architecture for access control verification in serverless environments. In 2022 IEEE Symposium on Computers and Communications (ISCC) (pp. 1-6). IEEE.

[178] Sailer, R., Valdez, E., Jaeger, T., Perez, R., Van Doorn, L., Griffin, J. L., … & Jaeger, T. (2005). sHype: Secure hypervisor approach to trusted virtualized systems. Techn. Rep. RC23511, 5.

[179] Samaila, M. G., Neto, M., Fernandes, D. A., Freire, M. M., & Inácio, P. R. (2018). Challenges of securing Internet of Things devices: A survey. Security and Privacy, 1(2), e20.

[180] Samea, F., Azam, F., Rashid, M., Anwar, M. W., Haider Butt, W., & Muzaffar, A. W. (2020). A model-driven framework for data-driven applications in serverless cloud computing. Plos one, 15(8), e0237317.

[181] Sankaran, A., Datta, P., & Bates, A. (2020, December). Workflow integration alleviates identity and access management in serverless computing. In Proceedings of the 36th Annual Computer Security Applications Conference (pp. 496-509).

[182] Scarfone, K., Souppaya, M., Cody, A., & Orebaugh, A. (2008). Technical guide to information security testing and assessment. NIST Special Publication, 800(115), 2-25.

[183] Scholte, T., Robertson, W., Balzarotti, D., & Kirda, E. (2012, July). Preventing input validation vulnerabilities in web applications through automated type analysis. In 2012 IEEE 36th annual computer software and applications conference (pp. 233-243). IEEE.

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

[185] Shafiei, H., Khonsari, A., & Mousavi, P. (2022). Serverless computing: a survey of opportunities, challenges, and applications. ACM Computing Surveys, 54(11s), 1-32.

[186] Shahrad, M., Fonseca, R., Goiri, I., Chaudhry, G., Batum, P., Cooke, J., … & Bianchini, R. (2020). Serverless in the wild: Characterizing and optimizing the serverless workload at a large cloud provider. In 2020 USENIX annual technical conference (USENIX ATC 20) (pp. 205-218).

[187] Shen, J., Zhang, H., Geng, Y., Li, J., Wang, J., & Xu, M. (2022, November). Gringotts: Fast and accurate internal denial-of-wallet detection for serverless computing. In Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (pp. 2627-2641).

[188] Shin, Y., Hur, J., Koo, D., & Yun, J. (2020). Toward serverless and efficient encrypted deduplication in mobile cloud computing environments. Security and Communication Networks, 2020, 1-15.

[189] Silowash, G. J., Cappelli, D. M., Moore, A. P., Trzeciak, R. F., Shimeall, T., & Flynn, L. (2012). Common sense guide to mitigating insider threats.

[190] Silva, P., Fireman, D., & Pereira, T. E. (2020, December). Prebaking functions to warm the serverless cold start. In Proceedings of the 21st International Middleware Conference (pp. 1-13).

[191] Silverman, M. P. (2008). Quantum condensates in extreme gravity: Implications for cold stars and dark matter. International Journal of Modern Physics D, 17(03n04), 603-609.

[192] Simmons, P. (2011, December). Security through amnesia: a software-based solution to the cold boot attack on disk encryption. In Proceedings of the 27th Annual Computer Security Applications Conference (pp. 73-82.

[193] Singaravelu, L., Pu, C., Härtig, H., & Helmuth, C. (2006, April). Reducing TCB complexity for security-sensitive applications: Three case studies. In Proceedings of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems 2006 (pp. 161-174).

[194] Singh, D., & Dautaniya, A. K. (2019). Cloud Computing Security Challenges and Solution. Turkish Journal of Computer and Mathematics Education (TURCOMAT), 10(3), 1185-1190.

[195] Singh, A., & Kolluri, S. (2021). Security and Privacy Challenges in Cloud-Based Database Management: Strategies and Solutions. TECHNO REVIEW Journal of Technology and Management, 1(1), 32-40.

[196] Sivanathan, A. (2020). IoT behavioral monitoring via network traffic analysis. arXiv preprint arXiv:2001.10632.

[197] Sobb, T., Turnbull, B., & Moustafa, N. (2020). Supply chain 4.0: A survey of cyber security challenges, solutions and future directions. Electronics, 9(11), 1864.

[198] Solaiman, K. (2023). Novel architecture for mitigating cold start problem in serverless computing.

[199] Somani, G., Gaur, M. S., Sanghi, D., Conti, M., & Buyya, R. (2017). DDoS attacks in cloud computing: Issues, taxonomy, and future directions. Computer communications, 107, 30-48.

[200] Stamp, J., Dillinger, J., Young, W., & DePoy, J. (2003). Common vulnerabilities in critical infrastructure control systems. SAND2003-1772C. Sandia National Laboratories.

[201] Stojkovic, J., Xu, T., Franke, H., & Torrellas, J. (2023, June). Mxfaas: Resource sharing in serverless environments for parallelism and efficiency. In Proceedings of the 50th Annual International Symposium on Computer Architecture (pp. 1-15).

[202] Straub, D. W., & Welke, R. J. (1998). Coping with systems risk: Security planning models for management decision making. MIS quarterly, 441-469.

[203] Suo, K., Son, J., Cheng, D., Chen, W., & Baidya, S. (2021, September). Tackling cold start of serverless applications by efficient and adaptive container runtime reusing. In 2021 IEEE International Conference on Cluster Computing (CLUSTER) (pp. 433-443). IEEE.

[204] Sushma, D., Nalini, M. K., Kumar, R. A., & Nidugala, M. (2023, November). To Detect and Mitigate the Risk in Continuous Integration and Continues Deployments (CI/CD) Pipelines in Supply Chain Using Snyk tool. In 2023 7th International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS) (pp. 1-10). IEEE.

[205] Rangaraju, S. (2023). Secure by Intelligence: Enhancing Products with AI-Driven Security Measures. EPH-International Journal of Science And Engineering, 9(3), 36-41.

[206] Reniers, V., Van Landuyt, D., Rafique, A., & Joosen, W. (2019). Object to NoSQL Database Mappers (ONDM): A systematic survey and comparison of frameworks. Information Systems, 85, 1-20.

[207] Richards, N., & Hartzog, W. (2015). Taking trust seriously in privacy law. Stan. Tech. L. Rev. 19, 431.

[208] Ristov, S., Hollaus, C., & Hautz, M. (2022, July). Colder than the warm start and warmer than the cold start! Experience the spawn start in faas providers. In Proceedings of the 2022 Workshop on Advanced tools, programming languages, and Platforms for Implementing and Evaluating algorithms for Distributed systems (pp. 35-39).

[209] Ron, A., Shulman-Peleg, A., & Puzanov, A. (2016). Analysis and mitigation of NoSQL injections. IEEE Security & Privacy, 14(2), 30-39.

[210] Tabrizchi, H., & Kuchaki Rafsanjani, M. (2020). A survey on security challenges in cloud computing: issues, threats, and solutions. *The journal of supercomputing*, *76*(12), 9493-9532.

[211] Tan, R., Nguyen, H. H., Foo, E. Y., Yau, D. K., Kalbarczyk, Z., Iyer, R. K., & Gooi, H. B. (2017). Modeling and mitigating impact of false data injection attacks on automatic generation control. IEEE Transactions on Information Forensics and Security, 12(7), 1609-1624.

[212] Tatineni, S. (2023). Compliance and Audit Challenges in DevOps: A Security Perspective. International Research Journal of Modernization in Engineering Technology and Science, 5(10), 1306-1316.

[213] Telo, J. (2023). Smart city security threats and countermeasures in the context of emerging technologies. International Journal of Intelligent Automation and Computing, 6(1), 31-45.

[214] Tolosana-Calasanz, R., Castañé, G. G., Bañares, J. Á., & Rana, O. (2021). Modelling serverless function behaviours. In Economics of Grids, Clouds, Systems, and Services: 18th International Conference, GECON 2021, Virtual Event, September 21–23, 2021, Proceedings 18 (pp. 109-122). Springer International Publishing.

[215] Tomarchio, O., Calcaterra, D., & Modica, G. D. (2020). Cloud resource orchestration in the multi-cloud landscape: a systematic review of existing frameworks. Journal of Cloud Computing, 9(1), 49.

[216] Tuyishime, E., Balan, T. C., Cotfas, P. A., Cotfas, D. T., & Rekeraho, A. (2023). Enhancing Cloud Security—Proactive Threat Monitoring and Detection Using a SIEM-Based Approach. Applied Sciences, 13(22), 12359.

[217] Uddin, M., Islam, S., & Al-Nemrat, A. (2019). A dynamic access control model using authorising workflow and task-role-based access control. Ieee Access, 7, 166676-166689.

[218] Ullah, F., Edwards, M., Ramdhany, R., Chitchyan, R., Babar, M. A., & Rashid, A. (2018). Data exfiltration: A review of external attack vectors and countermeasures. Journal of Network and Computer Applications, 101, 18-54.

[219] Vahidinia, P., Farahani, B., & Aliee, F. S. (2020, August). Cold start in serverless computing: Current trends and mitigation strategies. In 2020 International Conference on Omni-layer Intelligent Systems (COINS) (pp. 1-7). IEEE.

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

[221] Varia, J. (2010). Architecting for the cloud: Best practices. Amazon Web Services, 1, 1-21.

[222] Vegesna, V. V. (2019). Investigations on Different Security Techniques for Data Protection in Cloud Computing using Cryptography Schemes. Indo-Iranian Journal of Scientific Research (IIJSR) Volume, 3, 69-84.

[223] Verma, G., & Upadhayay, D. (2019). Cloud Computing Trends for the Future. Turkish Journal of Computer and Mathematics Education (TURCOMAT), 10(3), 1177-1184.

[224] Vielberth, M., Böhm, F., Fichtinger, I., & Pernul, G. (2020). Security operations center: A systematic study and open challenges. IEEE Access, 8, 227756-227779.

[225] VS, D. P., Sethuraman, S. C., & Khan, M. K. (2023). Container security: precaution levels, mitigation strategies, and research perspectives. Computers & Security, 103490.

[226] Wang, Z., Zhao, K., Li, P., Jacob, A., Kozuch, M., Mowry, T., & Skarlatos, D. (2023, October). Memento: Architectural Support for Ephemeral Memory Management in Serverless Environments. In Proceedings of the 56th Annual IEEE/ACM International Symposium on Microarchitecture (pp. 122-136).

[227] Wen, J., Chen, Z., Jin, X., & Liu, X. (2023). Rise of the planet of serverless computing: A systematic review. ACM Transactions on Software Engineering and Methodology, 32(5), 1-61.

[228] Xiong, J., Wei, M., Lu, Z., & Liu, Y. (2021, November). Warmonger: inflicting denial-of-service via serverless functions in the cloud. In Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (pp. 955-969).

[229] Yaqoob, T., Abbas, H., & Atiquzzaman, M. (2019). Security vulnerabilities, attacks, countermeasures, and regulations of networked medical devices - A review. IEEE Communications Surveys & Tutorials, 21(4), 3723-3768.

[230] Yakoob, S. K. (2021). Advanced Machine Learning Approach to Handle Code Injection Attacks in Cloud Computing. Turkish Journal of Computer and Mathematics Education (TURCOMAT), 12(10), 4623-4629.

[231] Yan, Q., & Yu, F. R. (2015). Distributed denial of service attacks in software-defined networking with cloud computing. IEEE Communications Magazine, 53(4), 52-59.

[232] Zanella, M., Massari, G., Galimberti, A., & Fornaciari, W. (2018, October). Back to the future: Resource management in post-cloud solutions. In Proceedings of the Workshop on INTelligent Embedded Systems Architectures and Applications (pp. 33-38).

[233] Zhang, Y., Goiri, Í., Chaudhry, G. I., Fonseca, R., Elnikety, S., Delimitrou, C., & Bianchini, R. (2021, October). Faster and cheaper serverless computing on harvested resources. In Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles (pp. 724-739).

[234] Zheng, S. (2020). Improving Network Security with Low-Cost and Easy-to-Adopt Solutions (Doctoral dissertation, Duke University, Durham, NC, USA).

[235] Zhijun, W., Wenjing, L., Liang, L., & Meng, Y. (2020). Low-rate DoS attacks, detection, defense, and challenges: a survey. IEEE access, 8, 43920-43943.

[236] Zhou, H., Hu, Y., Ouyang, X., Su, J., Koulouzis, S., de Laat, C., & Zhao, Z. (2019). CloudsStorm: A framework for seamlessly programming and controlling virtual infrastructure functions during the DevOps lifecycle of cloud applications. Software: Practice and Experience, 49(10), 1421-1447.

[237] Zobaed, S. M., & Salehi, M. A. (2023). Confidential computing across edge-to-cloud for machine learning: A survey study. arXiv preprint arXiv:2307.16447