Securing Serverless Computing: Challenges, Solutions, and Opportunities

Xing Li, Xue Leng, and Yan Chen

Abstract

Serverless computing is a new cloud service model that reduces both cloud providers' and consumers' costs through agile development, operation, and charging mechanisms. It has been widely applied since its emergence. Nevertheless, some characteristics of serverless computing, such as fragmented application boundaries, have raised new security challenges. Considerable literature has been committed to addressing these challenges. Commercial and open-source serverless platforms implement many security measures to enhance serverless environments. This article presents the first survey of serverless security that considers both the literature and industrial security measures. We summarize the primary security challenges, analyze corresponding solutions from the literature and industry, and identify potential research opportunities. Then, we conduct a gap analysis of the academic and industrial solutions, as well as commercial and open-source serverless platforms' security capabilities. Finally, we present a complete picture of current serverless security research.

Introduction

The development of cloud computing has driven various service model innovations, including serverless computing. In this model, cloud providers are responsible for all server-related management tasks, such as resource allocation, service deployment, scaling, and monitoring, and an application is charged only for its execution time. As a result, consumers can avoid tedious management tasks, focus on the business code, and save costs by not paying for idle resources. These advantages in terms of efficiency and economy have led to the rapid development of serverless computing in recent years and have attracted extensive attention from both industry and academia. According to a recent report, the global serverless market size is estimated to grow to $21.1 billion by 2025 [1].

However, as a novel service model, serverless computing also has certain distinguishing features that present some challenges, which lead to security and compliance concerns. For example, its agile and lightweight virtualization technologies may lead to weak isolation, and the ephemeral nature of its computing instances can increase the difficulty of security management. In recent years, substantial efforts have been made to address these challenges. Related studies have arisen from academic research, commercial cloud providers, and thriving open-source communities and have considerably enhanced serverless security. Therefore, a systematic survey of current research progress is needed to provide a foundation for continuing security enhancement.

Some previous surveys of serverless computing have been conducted [2, 3], however, they have three main drawbacks when revealing the current status of security research. In terms of subject matter, these surveys have extensively discussed the concepts, challenges, applications, and prospects of serverless computing but have not specifically targeted security. Regarding content, previous surveys have briefly introduced possible risks or attacks but have not systematically analyzed existing solutions and future research directions. In terms of materials, these surveys have focused on literature and have not considered the measures adopted in industry (i.e., commercial or open-source platforms). However, an academic focus alone is insufficient for a review of a widely applied cloud computing model, such as serverless computing.

To promote serverless computing and inspire new research, we present the first survey on serverless security whose horizon is expanded from academia alone to include industry. In this article, we first introduce the concept and background of serverless computing. Starting from the unique characteristics of this service model, we then present a classification of the main security challenges and analyze their root causes. Subsequently, we review the state-of-the-art solutions proposed in the literature and adopted by popular serverless platforms for each challenge. Based on the degree to which these problems are solved, we note potential research opportunities. Finally, we illustrate our findings in comparisons of the current academic and industrial solutions, as well as commercial and open-source platforms.

The main contributions of this work are as follows:

- A systematic summary of the challenges arising in securing serverless computing is presented.
- A brief but comprehensive review of academic and industrial solutions is provided.
- A set of promising potential research opportunities is proposed.
- A multiaspect gap analysis of the current research status is conducted.

Digital Object Identifier:
10.1109/MNET.005.2100335

Xue Leng (corresponding author) is with Hangzhou Institute of Technology, Xidian University and School of Cyber Engineering, Xidian University; Xing Li is with Zhejiang University, China; Yan Chen is with Northwestern University, USA.

0890-8044/23/$25.00 © 2023 IEEE

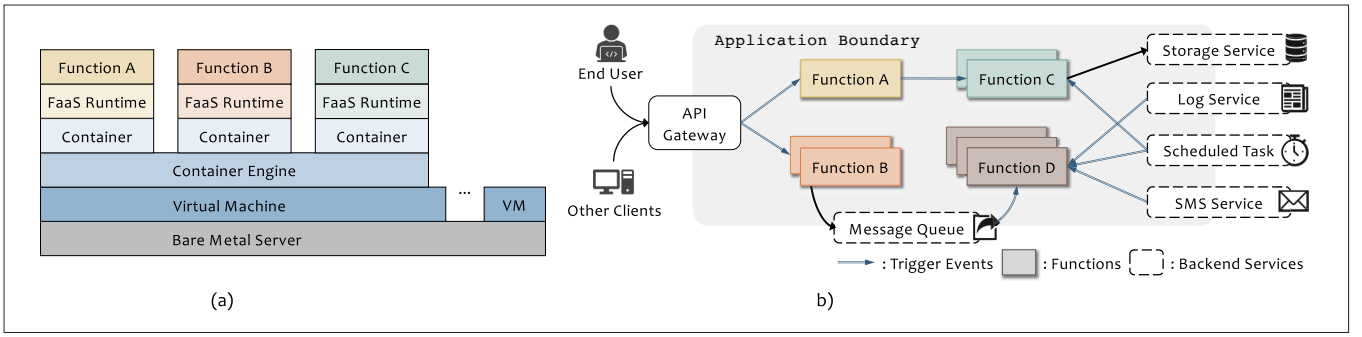

FIGURE 1. a) A container-based FaaS technology stack. Typically, only the top-level function code is managed by customers; b) The architecture of a serverless application that includes FaaS and BaaS components.

Background and Overview

Serverless Computing

Virtualization technology is the cornerstone of cloud services. Based on the platform-as-a-service (PaaS) model, the recent prosperity of lightweight virtualization (e.g., *containers*) has birthed a new model: function-as-a-service (FaaS). As shown in Fig. 1a, in this model, customers develop and deploy small code pieces called functions in the cloud. These functions are deployed in lightweight virtualized environments such as containers and executed on platform-provided runtimes. Each function focuses on a different task, and multiple functions can be combined in an event-driven manner to realize complex business logic. Events that can activate the execution of functions are called *triggers*; these include but are not limited to HTTP requests, logs, storage events, and timers.

With lightweight virtualization, functions can be initialized extremely fast (usually within a few milliseconds). Therefore, providers can instantiate functions only when needed and flexibly scale them, thereby maximizing resource utilization. Consequently, functions usually have a short lifecycle. In addition, this agility is also beneficial to consumers. Since the occupied resources are released when they are not in use, consumers need to pay only for the functions' actual execution time and do not need to reserve resources for emergencies.

Moreover, cloud providers offer dedicated services and application programming interfaces (APIs) for tasks, such as storage, logging, and identity management. These services can help customers build and manage server-side logic, thus considerably accelerating application development and release. This service model is known as backend-as-a-service (BaaS).

Serverless computing is generally regarded as a combination of FaaS and BaaS. By undertaking all management tasks directly related to infrastructure resources, cloud providers make server-related details transparent to consumers. From the customer's perspective, cloud applications are serverless, neither development, deployment, management, nor billing is server-centric. Figure 1 illustrates the architecture of a typical serverless application.

Currently, serverless computing is widely applied. Major providers, such as Amazon Web Services (AWS), Microsoft Azure, and Google Cloud have all launched serverless products; a series of open-source platforms is also emerging. Meanwhile, this new cloud service model has begun being embraced in many scenarios, such as data processing and the Internet of Things (IoT) paradigm.

Threats and Security Challenges

As a multitenant cloud service model, serverless computing is susceptible to security threats that can be divided into five categories based on where they are launched. The first category consists of external attacks on applications from malicious users (①), such as cross-site scripting attacks and injection attacks. Due to the unlimited scalability and the pay-as-you-go feature of serverless computing, denial of service attacks lead to a substantial increase in terms of cost to application owners. The second is composed of internal attacks on applications from malicious insiders (②), such as illegal internal access and sniffing attacks. Adversaries in the internal network can even perceive sensitive information from the communication pattern and activity level of functions. For example, in a health monitoring application, a particular sequence of triggered functions may reflect a patient's specific health condition[3]. The remaining three categories are horizontal attacks between tenants (③, e.g., side-channel attacks), vertical attacks on serverless infrastructures from malicious tenants (④}, e.g., container escape attacks), and vertical attacks on applications from malicious platforms (⑤).

To address these threats, the best practice is to build a defense-in-depth system to protect serverless applications and platforms at multiple stages and levels. However, the flexibility and agility of serverless applications expose them to a series of security challenges in the process of achieving this goal. These challenges are concentrated in four aspects: resource isolation (❶), security monitoring (❷), security control (❸), and data protection (❹). Figure 2 and Table 1 depict the locations of the threats and challenges in the serverless framework, as well as the relationships between them.

Survey Methodology and Objects

To fully understand the current state of the art in serverless security technologies, we collected and investigated the solutions proposed in past research work and the security mechanisms adopted in practice for serverless platforms. An overview of this survey is presented in Table 1.

For the literature, we collected 92 papers related to serverless security that have been published since 2014 (when the serverless concept emerged). We selected nine recent research

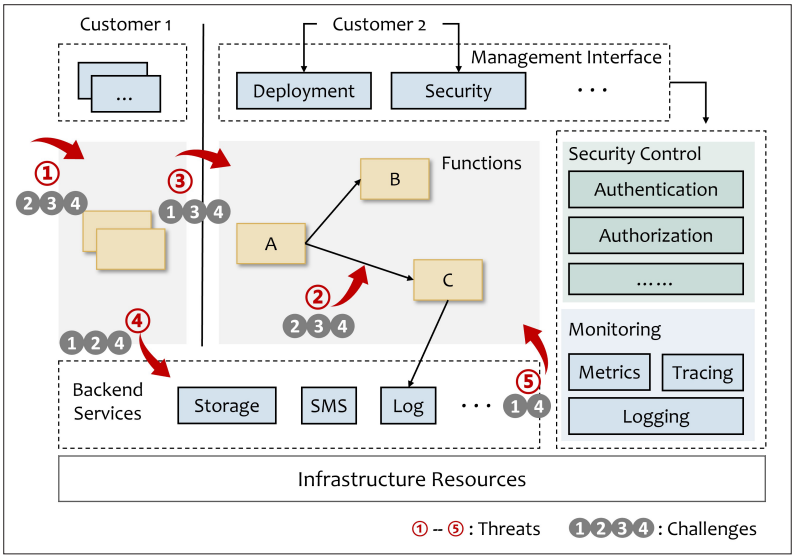

FIGURE 2. Locations of threats and security challenges in the serverless computing framework.

papers published in high-reputation conferences or journals to showcase the academic world's latest achievements.

Moreover, we collected the security features of popular serverless platforms from their official website and documentation. These platforms include the leading commercial platforms, AWS Lambda, Google Cloud Functions, Azure Functions, Alibaba Cloud Function Compute, and IBM Cloud Functions, as well as the popular open-source serverless frameworks, OpenFaaS, Knative, Fission, Apache OpenWhisk, and Nuclio.

Challenges, Solutions, and Opportunities

Challenge 1: Resource Isolation

Serverless computing relies on the dynamic release and reuse of software and hardware to improve resource utilization and reduce costs. Thus, there is a strong requirement for resource isolation, especially among multiple tenants.

However, this requirement conflicts with the weak isolation of lightweight virtualization, thereby giving rise to this challenge. Serverless computing requires swift and on-demand function initialization. To this end, lightweight virtualization technologies have been widely adopted. Unlike traditional virtual machines (VMs) with guest operating systems (OSs), containers share the host's OS kernel, and thus, their isolation is weak. Malicious consumers could employ this weakness to influence or attack other consumers located on the same host (e.g., via side-channel attacks). For example, Wang et al. [13] found that AWS Lambda suffers from performance isolation issues, and Azure Functions has placement vulnerabilities that facilitate side-channel attacks.

Literature Work: To enhance the isolation of lightweight virtualization technologies and address this challenge, published research focuses on VM-based solutions. Compared with containers, VMs can provide better security boundaries but require a longer time to initialize. Hence, researchers have attempted to make VMs more lightweight to meet the flexibility requirements of serverless computing. The basic idea is to reduce the size of the guest kernels, that is, remove unnecessary modules, to build "microVMs," even keeping only the libraries that will be used [4, 5]. Specifically, Li *et al*. [5] employed a pre-patched kernel image, the read-write splitting rootfs, and a lightweight cgroup to further accelerate initialization, and Agache *et al*. [4] minimized virtual machine monitors (VMMs) to reduce the trusted computing base. These microVMs have successfully reduced the initialization time to the millisecond level while providing VM-level isolation.

Industrial Solutions: Led by Amazon, Firecracker [4] has been applied in AWS Lambda. Moreover, all surveyed commercial serverless platforms except IBM Functions utilize their own *secure container* products. These containers enable advanced isolation by employing various techniques, such as microVMs (e.g., Firecracker) and container sandboxes (e.g., gVisor). Commercial platforms also provide mechanisms for network-layer isolation, such as virtual private clouds (VPCs).

Review and Comparison: Secure containers provide VM-level resource isolation and have become a standard mode of implementation in production environments. However, due to hardware sharing, these approaches are still susceptible to side-channel attacks. Moreover, trade-offs still remain between security and performance in the area of lightweight virtualization. For example, gVisor sacrifices significant performance for security (in particular, a 216× reduction in file opening speed) [14]. Additionally, many existing efforts employ snapshots [15] or even kernel sharing [5] to reduce the start-up time as much as possible, which also degrades security to a certain extent. Therefore, we consider this challenge partially resolved. To this challenge, the academic and industrial solutions are consistent, which means that the technologies proposed by academia have been recognized and applied by industry.

Research Opportunities: The emergence of serverless computing brings new opportunities and challenges to the research of secure containers, including achieving better isolation than VMs, providing performance closer to that of physical machines, and incurring lower overheads. From our point of view, technologies such as unikernels, WebAssembly, and hardware-based isolation and acceleration might be potential directions to facilitate the problem solving.

Challenge 2: Security Monitoring

A serverless platform generally maintains a set of runtimes for functions, including commonly used packages of supported programming languages. Moreover, to meet diverse development needs, most providers also support custom runtimes and even provide function marketplaces. The security risks posed by third-party packages and functions necessitate reliable monitoring capabilities to detect anomalies and determine the security status.

Furthermore, in a serverless environment, the ephemeral nature of functions and the broken application boundaries present new challenges. First, although the short-lived nature of functions reduces the attack surface, it also significantly narrows the window for developers to discover and diagnose problems. Second, as shown in Fig. 1b, a serverless application may have multiple

Challenges	Resource Isolation	Security Monitoring	Security Control	Data Protection
Root Causes	The weakness of lightweight virtualization technologies in isolation	The ephemerality of functions and the broken boundaries of serverless applications	The distributed nature of functions and the fragmented application boundaries	Platforms' control of the function lifecycle and BaaS services' participation in business logic
Literature Work	Virtual-machine-based secure containers, e.g., minimized VMMs [4] and pre-patched kernel image [5]	Information flow mining and tracing, e.g., static analysis [6] and dynamic tracing [7, 8]	Advanced management capabilities, e.g., workflow-sensitive authorization [9] and secure container network stacks [10]	Hardware-based trusted execution environments, e.g., protecting running code with Intel SGX [11, 12]
Industrial Solutions	Secure containers and network boundaries, e.g., Firecracker [4], gVisor, and VPCs	Security scanning and monitoring tools, e.g., Azure Monitor, AWS X-Ray, and DevSecOps tools	Fine-grained authentication and authorization, e.g., IAM systems and role-based access control	Encryption in transit and at rest, e.g., TLS/SSL, code/data encryption, and key management services
Corresponding Threats	③	①②④	①②③	①②③④⑤
Status	Partially resolved	Partially resolved	Partially resolved	Partially resolved
Research Opportunities	Secure containers with better isolation, better performance, and lower overhead	Nonintrusive tracing schemes and advanced insight capabilities such as diagnosis and forensics	Automatic security configuration and auditing tools	Confidential computing, homomorphic encryption

TABLE 1. The landscape of the survey, including security challenges and their root causes, proposed solutions, our reviews, and research opportunities.

entry points due to a variety of triggers. These entry points, coupled with BaaS services, cause the boundaries of serverless applications to be fragmented and further increase the difficulty of monitoring. Moreover, lacking control over the execution environments and BaaS services, consumers must rely on the provided interfaces for monitoring, which may lead to incomplete views of the running status. For example, Lin *et al.* [7] noted that AWS Lambda's monitoring granularity is relatively coarse, and the tracking paths are not continuous across BaaS services.

Literature Work: To address this challenge, researchers have attempted to mine and track the information flows in serverless applications from both static and dynamic perspectives. Obetz *et al.* [6] proposed an augmented call graph that considers BaaS services and explained the feasibility of obtaining the graph through static analysis. To dynamically track where and how data flow, Datta *et al.* [8] placed an agent in each function's container to proxy its network requests. These agents generated and propagated labels for inbound and outbound traffic to record the flow of information and could also restrict functions' behavior based on preconfigured policies. In addition, Lin *et al.* [7] proposed GammaRay, a dynamic tracking tool that can capture the causality of invocations across BaaS services to build complete trace graphs.

Industrial Solutions: Commercial platforms provide various tools and interfaces to record, aggregate, display, and analyze monitoring data. Some can even serve as triggers for automatic function management. They also aim to gradually enable distributed tracing while ensuring adequate performance. Moreover, to further mitigate the impact of nonsecure third-party components, providers recommend continuous security validation in continuous integration and continuous deployment (CI/CD) pipelines and offer a series of tools, such as static vulnerability scanning and automatic penetration testing, for this purpose.

Review and Comparison: The proposed dynamic methods solve the problem of the coarse-grained and incomplete nature of the existing monitoring schemes to a certain extent. For example, data-oriented tracing [8] can be integrated with traditional behavior-oriented mechanisms. Moreover, since functions have relatively low internal complexity, static-analysis-based methods are feasible. These approaches could be an effective supplement to dynamic methods, which may introduce considerable runtime overhead. Following best practices in the development phase can also mitigate the threat posed by nonsecure third-party components. Unfortunately, the information that can be obtained through static analysis is limited. Current dynamic solutions all require code instrumentation or agent embedding to enable end-to-end monitoring and tracking, resulting in high compilation and runtime overheads [7, 8]. Hence, we consider this challenge only partially resolved. For this challenge, solutions from academia and industry can complement each other. Mutual reinforcement and seamless integration with existing solutions could be the next focus.

Research Opportunities: The highest end-to-end performance is the relentless pursuit of serverless computing. To this end, nonintrusive and agentless monitoring and tracing schemes would be potential research opportunities. For example, eBPF could be employed to perform high-performance, nonintrusive monitoring since eBPF captures events at the system level rather than injecting agents into function instances. Furthermore, advanced insight capabilities are lacking in current platforms. These capabilities include assessing security status, predicting and diagnosing faults, and conducting detection and forensics for security events. Meanwhile, functions' short lifecycles and massive logs impose high requirements on the speed of solutions. We believe that artificial intelligence (AI)-based approaches could be promising in this area. With various triggers, serverless applications can naturally perform "feedback regulation" for intelligent self-management.

Challenge 3: Security Control

Security control is critical for administrators to enforce security intentions and protect applica-

Security control is critical for administrators to enforce security intentions and protect applications. As described earlier, the diversity of the possible attacks in serverless environments necessitates comprehensive security control capabilities that can provide protection at multiple levels.

tions. As described earlier, the diversity of the possible attacks in serverless environments necessitates comprehensive security control capabilities that can provide protection at multiple levels.

However, the fragmented application boundaries of serverless applications increase the difficulty of security control. The distributed nature of functions further exacerbates this challenge. Functions must carefully inspect the input and output at all their entry points; failure to comply with this principle may lead to injection attacks. Additionally, an external request may trigger communication among multiple functions and BaaS services, necessitating fine-grained authentication and authorization among functions and services. This task can be challenging for large-scale or complex applications.

Literature Work: Traditional authentication and authorization mechanisms have been well-studied. Building on this work, researchers are currently attempting to provide advanced security control capabilities at new levels. Sankaran *et al.* [9] built a workflow-sensitive authorization mechanism for serverless applications. It proactively verifies external requests' permissions for all functions involved in their workflows at the application entry point. Thus, applications can reject illegal requests as early as possible, thereby avoiding partial processing of illegal requests and reducing the potential attack surface. Moreover, Nam *et al.* [10] revealed a data leakage risk in container networks and redesigned a secure network stack to protect inter-container communication. Specifically, the method verifies and filters out illegal requests based on the inter-container dependencies and utilizes end-to-end direct forwarding to prevent traffic exposure. It even improves the container network's performance.

Industrial Solutions: Based on various security policies, cloud providers have formulated "best practices" for security control. These policies act on multiple layers and multiple resources and have the potential to provide promising manageability. For example, commercial platforms achieve well-defined identity authentication and fine-grained authorization with mature identity access management (IAM) systems. AWS Lambda even provides role-based policies, resource-based policies, attribute-based policies, and permission boundaries for consumers to choose among and flexibly combine.

Review and Comparison: Based on fine-grained security control policies, existing solutions provide multilevel protection for serverless applications. Providers also document best practices as guidance. However, a gap may inevitably exist between actual situations and the recommended best practices. First, both academic [9, 10] and industrial solutions require administrators to customize security policies in accordance with the application architecture and business logic. Although server management is the providers' responsibility, the burden placed on consumers for security control has not been relieved but instead has been made even heavier. Fine-grained policies require attentive design and careful configuration, which is a time-consuming and error-prone process. Second, providers' marketing emphasizes only cost-effectiveness and convenience, making it easy for consumers to ignore their responsibility of ensuring security. As mentioned above, any violation of design principles and best practices may compromise applications' security. In other words, current mechanisms do not reduce the complexity of security control. Applications' safety depends on the extent to which best practices are followed. Therefore, we consider this challenge partially resolved. In this field, solutions in academia and industry show a consistent trend: the refinement of security control is the main development direction.

Research Opportunities: Currently, the assistance of automated tools is urgently needed. One possible research direction is automated security configuration. The ideal tool would be able to perceive an application's architecture and status, extract the consumer's security intentions, and provide support for policy configuration, maintenance, and updating. Static analysis, dynamic inference, data mining, and intent analysis are expected to be key technologies of this tool. Another research direction concerns automated security audits, which has a growing market. The ideal tool is expected to not only check whether security measures are deployed but also evaluate the quality of these measures based on an application's specific characteristics. Dynamic probing and causal analysis are expected to be key technologies in this direction. We believe that the security risks posed by incorrect configurations can be reduced by advancing research in these two directions.

Challenge 4: Data Protection

Data security is the lifeline of cloud services. In contrast to other models, serverless providers control the whole application lifecycle (e.g., the deployment, scaling, execution, and termination of functions) and the entire software stack. Moreover, as part of the application and business logic, BaaS services can directly access and process data. All these characteristics require consumers to have deep trust in the provider.

Nevertheless, whether during the execution of code in an uncontrolled environment or the processing of sensitive data through the platform's services, privacy and compliance concerns may arise. To reduce these concerns, platforms need to implement strict data protection measures to clearly and fully persuade consumers. Given the flexibility requirements of serverless environments, this task could be challenging, but it is essential for expanding the application scenarios of serverless computing.

Literature Work: In addition to protecting data at rest or in transit through encryption, recent work has focused on protecting data in use. Researchers have attempted to establish a trusted execution environment (TEE) for functions to relieve customers of the need to fully trust providers [11, 12]. Several approaches employ Intel SGX, a hardware-based mechanism that uses encrypted memory to protect running code and data from privileged software, including the OS. Specifically, Qiang *et al.* [12] proposed a two-way sandbox,

that is, a WebAssembly sandbox nested in an SGX sandbox, to prevent a platform from accessing sensitive data processed by functions while also protecting the platform from untrusted code. Furthermore, PIE [11] utilizes reusable initial common states to improve the performance of SGX.

Industrial Solutions: On the basis of their mature storage services and security management experience, commercial platforms provide adequate support for encryption in transit and at rest. This protection covers the files and code uploaded by customers, as well as the environment variables of functions. Moreover, standard key management services (KMS) are provided to help customers handle secrets.

Review and Comparison: Protecting data in use has become a substantial dimension of cloud security. In this regard, current serverless research is focused on hardware-based TEEs, which still have many limitations, such as poor performance, difficulty with hardware heterogeneity, and running only in the user space. As a result, these academic techniques have not yet been applied to serverless products. Therefore, we consider this challenge to be partially resolved. For this challenge, academic solutions are more advanced but less mature than industrial ones, requiring further exploration.

Research Opportunities: In recent years, confidential computing has emerged to protect data in the cloud. The principal factor hindering the adoption of confidential computing in serverless computing is performance degradation. Since an enclave function's initialization consists of hardware enclave creation and attestation measurement generation, the startup time of functions becomes longer. Transferring the secret data between functions also increases the end-to-end execution time. Therefore, revisiting the design of SGX and further optimizing its performance is the key to solving the problem. In addition, other privacy computing techniques, such as homomorphic encryption and secure multiparty computation, may be another way to perform data protection in serverless computing.

Gap Analysis

During our investigation, we obtained some interesting findings about the current research status of serverless security, which may imply potential directions toward a better ecosystem.

Academic Solutions vs. Industrial Solutions

The details of academic and industrial solutions for each challenge have been compared above. From a holistic view, we compared the efforts in academia and industry from three perspectives: research interests, solution features, and application status. As mentioned above, both academic and industrial solutions cover the four main security challenges. This consistency in research interests implies that the current research is in a healthy state.

Nonetheless, the solutions show different characteristics. Academic research tends to demonstrate the feasibility of using specific methods, especially new technologies, to solve particular problems. In contrast, the goal of commercial platforms is to be as secure as possible while ensuring stability and overall consistency. Thus, providers prefer methods that have been proven in long-term practice and usually combine several methods to achieve multilayer protection. In addition, research-led solutions often adopt a more holistic approach from developer to platform. Without control over other parties, providers can promote their solutions only through guidelines or best practices.

Regarding application status, some of the reported academic research work has not yet been applied in industry, as it is relatively new. In addition, academic work usually lacks large-scale test environments and real world data for in-depth evaluation, which also slows its adoption.

Regarding application status, some of the reported academic research work has not yet been applied in industry, as it is relatively new. In addition, academic work usually lacks large-scale test environments and real world data for in-depth evaluation, which also slows its adoption. We believe that open-source efforts from both academia and industry, such as publicly available research, real world datasets, and standardized benchmarks, can help promote the smooth adoption of newly emerging solutions.

Commercial Platforms vs. Open-Source Platforms

Various commercial and open-source serverless platforms have flourished as a result of the vigorous serverless computing market. During our investigation, we found considerable differences between the security measures adopted in commercial and open-source platforms in terms of five aspects: isolation, monitoring, management, encryption, and tools. The detailed results are listed in Tables 2 and 3.

First, most commercial platforms are equipped with self-developed secure containers and provide VPCs for network-level isolation. In terms of monitoring, they introduce their own specialized distributed tracing and monitoring products into serverless scenarios and connect them to unified visualization consoles. Through security policies and IAM systems, these platforms provide mature authentication and authorization capabilities and support data encryption via storage services. Furthermore, their long-term investment and technological accumulation allow commercial platforms to provide many tools to assist customers, such as configuration recommendation and risk detection.

In contrast, most open-source platforms are built atop Kubernetes (K8s). Therefore, standard Docker containers and K8s namespace-based network isolation have become a general technical solution. For monitoring, these platforms usually integrate third-party tools or employ their infrastructures' capabilities to avoid reinventing the wheel. Moreover, open-source platforms mainly rely on ingress gateways and infrastructures, such as K8s and Istio, to achieve security control and encryption in transit. The secret management task is also outsourced to the underlying K8s infrastructure. Similar to the monitoring aspect, open-source platforms usually do not provide security tools natively but support the integration of third-party tools, such as Jenkins.

In terms of the above five aspects, the security measures of commercial platforms are all stronger than those of open-source platforms in terms of completeness and richness. The latter's security capabilities mainly rely on infrastructure or third-par-

Aspect	Category	AWS Lambda	Google Cloud Functions	Azure Functions	Alibaba Cloud Function Compute	IBM Cloud Functions
Isolation	System	Firecracker	gVisor	Azure Container Instances	Alibaba Cloud Sandbox	Docker containers
	Network	VPC	VPC	Azure ASE	VPC	VPC
Monitoring	Visualization	Amazon CloudWatch, Lambda Insights	Cloud Monitoring	Azure Monitor	Cloud Monitor, Insights, ARMS	IBM Cloud Monitoring
	Tracing	AWS X-Ray	Cloud Trace	Application Insights	Tracing Analysis	Zipkin/Jaeger
Security Control	Authentication	AWS IAM	Cloud Functions IAM	Azure AD	Alibaba Cloud RAM	IBM Cloud IAM
	Authorization	Role-based/attribute-based/ resource-based access control and permission boundaries	Role-based/network-based access control	Role-based access control	Role-based access control	Role-based access control
Encryption	In Transit	TLS	Google ALTS	TLS	TLS	TLS on ingress
	At Rest	Files, secrets, logs, environment variables	All data, KMS	All data, Key Vault	Code, environment variables, KMS	Data, secrets
Tools	Tools	AWS Trusted Advisor, Serverless Lens, IAM Access Analyzer	Event Threat Detection (ETD)	AD Access Reviews, Defender for Cloud, Microsoft Sentinel	—	—

TABLE 2. A brief list of security mechanisms used in commercial serverless platforms.

Aspect	Category	OpenFaaS (22.1k)[1]	Knative (9.8k)[1]	Fission (7.2k)[1]	OpenWhisk (5.8k)[1]	Nuclio (4.6k)[1]
Isolation	System	Docker	Docker	Docker	Docker	Docker
	Network	K8s namespace	K8s namespace	K8s namespace	K8s namespace	K8s namespace
Monitoring	Visualization	Prometheus/Grafana	Prometheus/Grafana	Prometheus/Grafana	Prometheus/Grafana	Dashboard
	Tracing	—	Zipkin/Jaeger	Zipkin/Jaeger	Zipkin/Jaeger	—
Security Control	Authentication	OAuth2	Istio authentication	Istio authentication	Self-hosted authentication	—
	Authorization	K8s RBAC	K8s/Istio RBAC	K8s/Istio RBAC	Support	—
Encryption	In Transit	Istio/Linkerd mTLS	Istio mTLS	Istio mTLS	TLS on ingress	TLS on ingress
	At Rest	K8s secrets	K8s secrets	K8s secrets	K8s secrets	K8s secrets

[1] The number of stars in GitHub as of September 2022.

TABLE 3. A brief list of security mechanisms used in open-source serverless platforms.

ty tools. There are two main reasons for this difference. First, commercial platforms have richer cloud management experience and more capital and human resources, so they can leverage their existing security mechanisms and provide standardized features as BaaS services (e.g., IAM systems). Second, the development of open-source platforms is still relatively rudimentary. They focus more on realizing core features of serverless computing, such as agile application construction and automatic function scaling rather than on security features.

Mature and standardized open-source platforms can counter the vendor lock-in problem and stimulate community efforts and related work. Nevertheless, open-source platforms still have a long way to go to compete with commercial platforms in terms of security. This security lag also limits their application. Hence, more effort is needed to develop and standardize the security mechanisms of open-source serverless platforms. Although some have taken the lead (e.g., OpenFaaS), the competition among open-source serverless platforms is still fierce, and there is no de facto standard. From this perspective, unremitting efforts in developing security measures and the establishment of unique security capabilities may become the key to gaining the dominant position.

Conclusion

This article presents a survey of both literature research and industrial solutions to elucidate the research status of techniques for securing serverless computing. Starting from the characteristics of serverless environments, we explain the origins of four main security challenges: resource isolation, security monitoring, security control, and data protection. Based on a review of existing solutions, we then identify possible research directions. Finally, we compare academic and industrial solutions, as well as current commercial and open-source serverless platforms and elaborated on our thoughts regarding promising directions for establishing a better research and application ecosystem.

Acknowledgment

This work is supported by the Fundamental Research Funds for the Central Universities (XJSJ23004) and Proof of Concept Foundation of Xidian University Hangzhou Institute of Technology under Grant No. GNYZ2023GY0401.

References

[1] MarketsandMarkets, "Serverless Architecture Market Size, Share and Global Market Forecast to 2025," 2020, accessed on 2022-09-01; available: http://bit.ly/serverless market.

[2] E. Jonas *et al.*, "Cloud Programming Simplified: A Berkeley View on Serverless Computing," arXiv preprint arXiv:1902.03383, 2019.

[3] H. Shafiei, A. Khonsari, and P. Mousavi, "Serverless Computing: A Survey of Opportunities, Challenges, and Applications," *ACM Computing Surveys*, 2019.

[4] A. Agache *et al.*, "Firecracker: Lightweight Virtualization for Serverless Applications," *Proc. 17th USENIX Symposium on Networked Systems Design and Implementation,* 2020, pp. 419–34.

[5] Z. Li *et al.*, "Rund: A Lightweight Secure Container Runtime for High-Density Deployment and High-Concurrency Start-up in Serverless Computing," *Proc. 2022 USENIX Annual Technical Conf.*, 2022, pp. 53–68.

[6] M. Obetz, S. Patterson, and A. Milanova, "Static Call Graph Construction in Aws Lambda Serverless Applications," *Proc. 11th USENIX Workshop on Hot Topics in Cloud Computing*, 2019.

[7] W.-T. Lin *et al.*, "Tracking Causal Order in AWS Lambda Applications," *Proc. 2018 IEEE Int'l. Conf. Cloud Engineering*, IEEE, 2018, pp. 50–60.

[8] P. Datta *et al.*, "Valve: Securing Function Workflows on Serverless Computing Platforms," *Proc. Web Conf. 2020*, 2020, pp. 939–50.

[9] A. Sankaran, P. Datta, and A. Bates, "Workflow Integration Alleviates Identity and Access Management in Serverless Computing," *Proc. Annual Computer Security Applications Conf.*, 2020, pp. 496–509.

[10] J. Nam et al., "Bastion: A Security Enforcement Network Stack for Container Networks," *Proc. 2020 USENIX Annual Technical Conf.*, 2020, pp. 81–95.

[11] M. Li, Y. Xia, and H. Chen, "Confidential Serverless Made Efficient With Plug-In Enclaves," *Proc. 2021 ACM/IEEE 48th Annual Int'l. Symposium on Computer Architecture*, IEEE, 2021, pp. 306–18.

[12] W. Qiang, Z. Dong, and H. Jin, "Se-lambda: Securing Privacy-Sensitive Serverless Applications Using SGX Enclave," *Proc. Int'l. Conf. Security and Privacy in Commun. Systems*, Springer, 2018, pp. 451–70.

[13] L. Wang *et al.*, "Peeking Behind the Curtains of Serverless Platforms," *Proc. 2018 USENIX Annual Technical Conf.*, 2018, pp. 133–46.

[14] E. G. Young *et al.*, "The True Cost of Containing: A Gvisor Case Study," *Proc. 11th USENIX Workshop on Hot Topics in Cloud Computing*, 2019.

[15] D. Ustiugov *et al.*, "Benchmarking, Analysis, and Optimization of Serverless Function Snapshots," *Proc. 26th ACM Int'l. Conf. Architectural Support for Programming Languages and Operating Systems*, 2021, pp. 559–72.

Biographies

Xing Li received the B.E. degree in software engineering from Shandong University, Jinan, China, in 2016 and received the Ph.D. degree in cybersecurity from Zhejiang University, Hangzhou, China, in 2021. His research interests are focused on the security of cloud networking and applications.

Xue Leng [M'22] received the Ph.D. degree in computer science and technology from Zhejiang University, Hangzhou, China, in 2020. She is currently a Jingying Associate Professor with the Hangzhou Institute of Technology, Xidian University, Hangzhou, China. Her research interests include security enhancement and performance optimization of cloud systems.

Yan Chen [F'17] received the Ph.D. degree in computer science from the University of California at Berkeley, Berkeley, CA, USA, in 2003. He is currently a Professor with the Department of Computer Science, Northwestern University, Evanston, IL, USA. His research interests include network security and diagnosis for large-scale distributed systems.