Association for Information Systems
AIS Electronic Library (AISeL)

WISP 2018 Proceedings

Pre-ICIS Workshop on Information Security and Privacy (SIGSEC)

Winter 12-13-2018

Securing Serverless Computing

Ravi Patnayakuni
University of Alabama in Huntsville

Nainika Patnayakuni
Calhoun Community College

Securing Serverless Computing

Ravi Patnayakuni[1]
College of Business, University of Alabama in Huntsville,
Huntsville, AL, USA

Nainika Patnayakuni
CIS Department, Calhoun Community College,
Huntsville, AL, USA

ABSTRACT

Serverless applications are based on a microservices-oriented system design, often consisting of several services, each with distinct functions that are composed and orchestrated to deliver specific functionality. The architecture allows firms to build and deploy software applications without consideration towards provisioning or maintaining the underlying infrastructure. The novelty of the architecture and its inherent characteristics present new challenges for cybersecurity. We discuss the security imperatives of this emerging cloud computing software paradigm. We then identify some of the approaches and practices that can be used by organizations to mitigate security threats in the context of serverless computing.

Keywords: Cybersecurity, Cloud Computing, Serverless, FaaS.

INTRODUCTION

Serverless computing is regarded as the next stage in the evolution of cloud computing as more and more computing migrates to the cloud (Barga 2017). It is expected that by 2020, 67% of all spending on IT infrastructure and software will be on cloud-based platforms (IDC 2016). Organizations adopt cloud computing for managing their IT infrastructure as it promises to be more scalable, on demand, less complex to manage and can easily and transparently shared as a

[1] Corresponding author. ravi.patnayakuni@uah.edu +1 256 415 7284

service across all applications and users. Initial cloud computing models rely on virtualization, however virtualization still requires maintaining and provisioning virtual servers with the need to take into consideration underlying operating system, application server, load balancing, security and other aspects of the run time environment (Lynn et al. 2017). Such an architecture, without substantial in-house expertise, makes it difficult for the software owner to decide how many virtual servers to deploy and maintain their scaling in response to traffic and computing needs (Fox et al. 2017). This was brought to sharp relief, when fans of John Oliver's HBO talk show brought down the FCC website with the sheer volume of traffic it generated and in what was initially thought to be a dDOS attack (Wallace 2018). Serverless architectures, in contrast, are characterized by on-demand, event-driven, short-lived, stateless computation that scales instantly and automatically (Lynn et al. 2017, Albuquerque et al. 2017). This event-driven approach to cloud computing invokes functions that usually have a small footprint, upon the occurrence of an event or action, are short living, stateless and release resources allocated to them once the function terminates (McGrath and Brenner 2017). This micro-service architecture provides a flexible and scalable approach to designing applications, where developers can focus on core product functionality without consideration of the underlying platforms or runtime environments. Applications owners no longer have to worry about the underlying infrastructure as the service provider takes on the responsibility of maintaining and securing the data center, network, servers, operating systems and their configurations.

The emergence of serverless architectures and computing presents numerous opportunities for software vendors and developers and offers several advantages along with some inherent limitations, the discussion of which is not central to this effort. As an emerging paradigm, serverless computing also presents numerous challenges, of which cybersecurity is a

critical issue. The domain is rapidly evolving, complex and not well understood; the attempt here is to place some markers to understand the landscape, issues and practices for cybersecurity that are driven by this new architecture. We next discuss some of the key cybersecurity challenges for software developers and owners.

THE SERVERLESS COMPUTING LANDSCAPE: CYBERSECURITY IMPERATIVES

Serverless architectures, also known as Function as a Service (FaaS), lets organizations build and deploy software without maintaining physical or virtual servers. Applications built using serverless architecture are designed to scale elastically with workload. These architectures required development of functionality that is scalable, modular and follow the principle of least knowledge, that can communicate through common protocols (Fazio et al. 2016). For example, altering media files; when a user uploads a media file, a function can be invoked to automatically resize the image. Or when a user sends an SMS to check their bank balance, a separate function could send a return SMS. Developers can compose applications with such functions that run on demand, in response to events, that are short lived and scale automatically without having to manage any of the infrastructure. At the same time, this presents a number of cybersecurity challenges that arise as a consequence of the architecture itself, as there is a lack of maturity in the understanding the domain and paucity of well-developed tools to mitigate the challenges (Ahmed and Hossain 2014). Next, we discuss some of the preeminent challenges presented by the architecture.

Complexity

Serverless applications, typically contain multiple serverless functions, each responsible for a distinct task, consuming different types of inputs, composed by event triggers and glued together with cloud services. These architectural characteristics, make them complex and many

of the security challenges are related to or arise from this inherent complexity (Singh et al. 2016, Puresec 2018).

Serverless functions pull data from a broad range of event sources such as HTTP APIs, cloud storage, data streams, code modifications, notifications, IoT device communications etc. The rich set of event sources increases the potential attack surface and introduces complexities when attempting to protect serverless functions against event-data injections (Baldini et al. 2017, Narula and Jain 2014). Traditionally, application firewalls scan input and attempt to detect malicious payloads at application entry points. These firewalls were not designed to scan data coming in as a result of an API call from the application itself. Furthermore, they need to perform input data inspection in cloud, which may be well understood for inbound web traffic, but problematic for the multitude of other sources consumed by such applications (Ali et al. 2015, Singh et al. 2016). One option may be to route the data for inspection out of the serverless environment to another cloud where it is inspected and then sent back to the application. The consequences of such an approach would be: (i) a significant performance cost, (ii) data from the function runtime environment needs to be collected and sent to the cloud or another virtual appliance raising further privacy and security concerns and (iii) it has to be similarly scalable, i.e. auto-scaling with the serverless function without degrading performance (Meng 2017). The performance cost also comes with a financial cost as any security tools will add to processing time for every request, which in turn will be billed to the application owner.

The total amount of information and number resources also increases in serverless computing (Fox et al. 2017). This is compounded by the fact that with smaller functions, developers are likely to deploy things quickly, incrementally and frequently (McGrath and Brenner 2017). This makes it difficult to garner useful intelligence from the large amounts of

data and get a coherent picture of the health of an application. When instead of a few instances, there are hundreds if not thousands of functions, it is hard to discern if any given function is behaving as intended. Every function can be a potential point of attack and it is important to assess each of them to evaluate if they can be compromised. This is true for protocols as well, with numerous different event triggers, each with their own methods for invocation. More resources also mean that there are more permissions that need to managed. The rich set of event sources increases the potential attack surface and introduces complexities when attempting to protect serverless functions against event-data injections (Aikat et al. 2017, Ahmed and Hossain 2014).

Serverless applications are by nature ephemeral, and may execute in globally distributed data centers and resources that are not controlled by the application owner (Lynn et al. 2017). The short-lived nature of the architecture does have some advantage with the fact that serverless functions are 'online' for a short period of time and have no memory, making them less susceptible to long-term attacks. However, this makes it particularly challenging for organizations to deploy traditional security layers such as web application firewalls, host-based intrusion prevention, endpoint protection etc. In addition, there is the erosion of a well-defined perimeter that bounded traditional architectures, making it difficult to determine where security should be deployed. This ephemerality also necessitates that the security solutions too need to scale in tandem with the serverless application as and when they execute (Fazio et al. 2016, McGrath and Brenner 2017). Thus, a number of cybersecurity challenges with serverless computing are rooted in the inherent complexity of the architecture and its characteristics as well as being strongly correlated to the other challenges associated with serverless architectures.

Function-Flow Vulnerabilities

Serverless architecture can potentially contain many distinct serverless functions that are stitched together and orchestrated to create the overall application logic. Some functions may expose public APIs, while others may be communicating with other functions and/or cloud services consuming a wide variety of inputs (Baldini et al. 2017, Ahmed and Hossain 2014). Formulating and implementing a robust authentication model to control access and provide protection to all relevant functions, event types and triggers, can be a complex undertaking and one that needs continuous review. For example, an application may have a set of public APIs that are secured with proper authentication, however, at the back end the application may be reading data from a cloud storage service without proper authentication exposing an, unauthenticated rogue entry point for a hacker. Without a robust authentication scheme, a hacker can potentially bypass application logic to manipulate its flow and thereby compromising security of the application.

Similarly, a hacker may be able to compromise the system by mapping different serverless functions to learn their permissions, resources and capabilities in an attempt to manipulate the invocation order. Using techniques like Return Oriented Programming chaining, the hacker can collect and re-order function executions, turning them into "serverless-gadgets", and the re-using them to mount an attack against the system (PureSec 2018). This would allow the hacker to bypass security protections such as authentication, authorization and validity checks. Hackers have been able to cause serverless platforms to scale, running a vulnerable function repeatedly until they reached the platform's limit for concurrent operations. This is not necessarily a flaw in the cloud platform but a vulnerability created by the auto-scaling nature of serverless architecture and vulnerable application code. Hackers can also use the Regular-

Expression Denial of Service or ReDoS attack vector to send malicious requests that cause functions to stall, loop or 'over-execute' for long periods of time, until their concurrent execution limit is reached (Narula and Jain 2015). This would not only deny legitimate users access to the application, but also increase the billing charges levied by the cloud provider, inflicting a financial cost to the target organization. Therefore, security vulnerabilities can not only exist at the granular level of an individual function but in the overall application design, which in itself may be constantly unpredictable, vary with every instantiation and potentially unpredictable configurations.

Tools and Methods

Tools and protocols for testing security of serverless architectures are not well developed and understood in view of the relative novelty of the architectures, especially when these applications interact and consume a variety of services, not all of which are necessarily controlled by the application owner (Balding et al. 2017, Singh et al. 2016). Currently available automated scanning tools used in software development are not well adapted to Serverless applications.

Statistical Application Security Testing (SAST) tools are used to ensure that code conforms to guidelines and standards, which find errors in code without executing the code itself (IBM 2018). Serverless applications that will contain functions stitched together on-the-fly with cloud services and event triggers are not amenable to such static testing (PureSec 2018). In addition, SAST tools are known to generate a larger amount of false positive or false negatives.

Dynamic Application Security Testing (DAST) is used to find security weaknesses and vulnerabilities with the application executing for a variety of vulnerabilities such as SQL injection and cross-site scripting using fault injection techniques (PureSec 2018). However,

DAST tools typically cover HTTP interfaces in web applications and particularly only those that follow the traditional HTML/HTTP request/response model and request format. Serverless applications, as mentioned, interact with a variety of non-HTTP sources, third party services or back-end cloud services that are not covered by such tools.

Interactive Application Security Testing (IAST) works by deploying an instrumentation agent inside the application that has the ability to apply its analysis to the entire application and produce more accurate results and verify a larger set of security rules (PureSec 2018). However, the ability to deploy such agents in cloud environments where the infrastructure is controlled by the vendor and the nature of serverless applications limits the ability to deploy them. Similarly, Run-time application security protection (RASP) is a security technology that uses runtime instrumentation within the application tool running continuous security checks from within the application, allowing it to protect the application even if the network's perimeter defenses have been breached. In the context of serverless architectures, current IAST and RASP deployment options either depend on deploying an instrumentation agent or by extracting data for inspection in the cloud or on a virtual appliance. Neither of these approaches, are practical for serverless architectures.

SECURING SERVERLESS ARCHITECTURES

We have examined some of the security challenges posed by serverless architecture driven by its complexity and the relative newness of the technology. We next collate some of the approaches that can be used to mitigate some of these challenges.

Identifying and Detecting Threats

We have seen that traditional tools for identifying and detecting threats are not amenable for deployment in serverless architecture because of the ephemeral nature of functions, control

over runtime environment, orchestration complexities and performance costs. At the same time cloud providers provide a number of different tools that organizations can leverage. Organizations have more visibility in logs and monitoring tools that record which functions interact with which other functions and are resources are accessed with what frequency. All of this visibility can substantially inform security.

Many cloud providers provide (AWS 2018, Microsoft 2018) cloud security monitoring tools that enable identification of potential problems in cloud infrastructure and cloud configurations. Their main objective is usually to ensure configurations are in line with best practices for security as well as any specific compliance requirements. These tools will scan the application's cloud account and provide feedback on the application's security posture. Ideally they should provide a complete inventory of serverless function in the applications and cloud services that are part of the application's architecture (IBM 2018). They can scan for over-permissive roles and security policies that need to be strengthened. Many of these tools are designed for traditional IaaS and PaaS models and as such should be evaluated for their adequacy for serverless architectures. Some of these solutions analyze logs to detect issues or security related events. Because of latency of information, these should be leveraged but are not a replacement for application layer protection.

Similarly, cloud providers (AWS 2018, Google 2018) usually provide extremely capable logging facilities, but out of the box basic configurations are not necessarily suitable for the purposes of monitoring and auditing. One of the principles for identifying and detecting security issues, is to enable traceability (Sahoo et al. 2010). Organizations should monitor, alert and audit actions and changes to their code in real time. In order to achieve real-time security event monitoring with proper audit trail, developers need to integrate logs and metrics with the system

so that will it fit the needs of their organization like collecting real time logs and sending them to a remote security information and event management system. Organizations need robust analytics and retrieval capabilities to provide insight into security related activity (PureSec 2018). Tools to continuously monitor events in the software environment are critical for intelligent threat detection.

The ephemeral and stateless nature of serverless applications means that exploits are unlikely to be long-term and unlikely to gain a persistent foothold into the application. Hackers are then likely to resort to repetitive stateless attack that are small and perhaps unnoticeable and then repeat the attack thousands of times till they complete the attack, such as exfilterating all of the data in small increments. This creates the imperative that instead of focusing on specific event, security monitoring has to be more attuned to the overall pattern of an attack. The shift to cloud- and service-based infrastructures favors a hit-and-run style attack model that can be executed within a single refresh period, or automated to live and execute over multiple refreshes (PurSec 2018). Organizations then have a new key indicator by analyzing real-time attack telemetry (Khan 2016, Singh et al. 2016). If the same system, infrastructure, or application requests or changes being made over and over again would indicate an attempt to compromise the system.

Protecting against Threats

Similar to identifying and detecting threats, organizations in the context of serverless computing should take into account the architecture while leveraging cloud providers' tools for configuration management, Identity and Access Management (IAM) and monitoring to harden their applications against potential security threats (Singh et al. 2016).

Serverless architectures require extensive customization and configuration settings in the host environment to adapt them to specific tasks and needs (McGrath and Brenner 2017, Lynn et al. 2017). Reducing the number of configuration errors is not only important for the production environment but also security. Configuration assessments can be performed using tools for common vulnerabilities and exposures, assess instances against security benchmarks, and automate notification of defects (AWS 2018). One of the recommended best practice designs for serverless architectures is to develop functions that are stateless, applications often rely on cloud storage infrastructure to store and persist data between executions (Fox et al. 2017, McGrath and Brenner 2017). A common weakness is that developers leave incorrectly configured cloud storage authentication/authorization. In order to avoid sensitive data vulnerabilities from cloud storage infrastructure, may cloud providers provide hardened cloud storage configurations, multi-factor authentication and encryption of data in transit and rest (AWS 2018). Application data that needs to be protected should be secured with encrypted storage and encryption keys maintained with a centralized encryption key management infrastructure or service. Similar to other capabilities, organizations should use of encryption key management services provided by these cloud providers for creation and maintenance of encryption keys (Microsoft 2018, Google 2018).

A recurrent theme in designing serverless applications is to follow the principles of least privilege (AWS 2018). The principle means that a serverless function should be given only those privileges, which are essential in order to perform its intended logic. The principle allows designers to enforce separation of duties for oversight and governance, making it easier to audit permissions on resources. Because functions follow the concepts of serverless architecture, many serverless applications contain dozens, hundreds or on occasion thousands of functions. This in

turn means managing function permissions and roles quickly becomes a tedious task. In order to manage the complexity organization may use a single permission model or security role for all functions, which are over-permissive and over-privileged (Narula and Jain 2015). Even if they have the intention to come back to it later at production to a finer-grained model, more often than not they end up creating gaping vulnerabilities in the application. Most cloud providers make available Identity and Access Management (IAM) tools for setting custom roles for each serverless functions (AWS 2018, Microsoft 2018, Google 2018). It is not pragmatic nor necessary for developers to build their own authentication schemes and rather use the IAM frameworks provided by the serverless environment. Used properly, they can provide fine-grained IAM around the functions and apply security policies to each of them. This granularity can be tedious to set up and maintain, but can go a long way to ensure that a security issue with one function does not scale up and cascade to the application environment. When proper authentication/authorization is applied, unauthorized users cannot add new functions or modify existing function code (IBM 2018). Similarly, organizations should use the security health check facilities provided by the serverless cloud provider to continuously monitor correct permissions and assess them against the organization's corporate security policy (AWS 2018). This needs to be monitored as the application evolves, as what may once be well configured can suddenly become sub-optimal as things change (PureSec 2018).

Serverless functions in order to perform a task, will often depend on third party software packages, open source libraries and third party remote web services through API calls (van Eyk et al. 2017). These can inadvertently create vulnerabilities in the application. It is prudent to have a well-defined process to deal with vulnerabilities in third party components. To start with it is important to have an inventory of software packages and other dependencies and their

versions (Baldini et al. 2017). With serverless apps being comprised of hundreds of functions, it is important to have a complete picture in order to understand the potential risk and as the application propagates, this can be challenging to maintain. It is best to consume third party packages from trustworthy resources and making sure that packages have not been compromised (PureSec 2018).

Recovering and Responding from Security Events

As was pointed out in identification and detection of security threats, security operations rely on the collection of logs and the use of search tools to discover potential events of interest such as unauthorized activity or change. A best practice for building mature security processes is to deeply integrate the flow of security events and findings into a notification and workflow system such as bug/issue system, ticketing system or other security information and event management (SIEM) systems (AWS 2018, IBM 2018).

Similarly, defining data backup, replication, and recover approach, organizations can protect against deletion and destruction of data (AWS 2018, PureSec 2018). A well-defined and validated process for data backup and replication can help the organization safeguard its data in the case of a disaster. Proper secured and protected primary and secondary data sources ensure continued business operations. Just as with any other architectures, some of the practices do not change in the context of serverless computing. It is important that organizations have an incident management process that aligns with architecture and needs of the organizations. They should run incident response simulations and use tools with automation to increase the speed for detection, investigation and recovery.

CONCLUSION

Different vendors have come up with best practices and principles for developing, deploying and monitoring serverless applications (AWS 2018, Microsoft 2018, Google 2018, IBM 2018). Distilling them leads to some broad characteristics for solutions to secure serverless architecture should be (i) inherently serverless, (ii) should scale with the application, (iii) without adversely affect performance in a manner that would be evident to users or other consumers of the services and as a corollary have very light footprints, (iii) portable across cloud platforms i.e. should be platform and environment agnostic and (iv) evolve with the evolution of the serverless paradigm (Baldini et al. 2017, PureSec 2018, Singh et al. 2016). Rather than just focusing on protection of a single layer, development and operations need to work in concert to map a defense enmeshed in the design at all layers along with other security controls. These layers in addition to the application would include the edge network, subnet, load balancer, every instance, operating system etc. As organizations investigate serverless architectures, they need to be cognizant of its appropriateness for the solution domain (Fox et al. 2017), understand the security imperatives of the architecture and incorporate security into their operations by developing an appropriate portfolio of design principles, leveraging cloud platform capabilities and post production practices.

REFERENCES

Ahmed, M., & Hossain, M. A. 2014. "Cloud computing and security issues in the cloud," International *Journal of Network Security & Its Applications*, 6(1), p.25.

Albuquerque Jr, L. F., Ferraz, F. S., Oliveira, R. F., & Galdino, S. M. 2017. "Function-as-a-Service X Platform-as-a-Service: Towards a Comparative Study on FaaS and PaaS," ICSEA 2017, p. 217.

Ali, M., Khan, S. U., and Vasilakos, A. V. 2015. "Security in cloud computing: Opportunities and challenges," *Information Sciences*, 305, pp. 357-383.

Aikat, J., Akella, A., Chase, J. S., Juels, A., Reiter, M., Ristenpart, T., Sekar, V. and Swift, M. 2017. "Rethinking security in the era of cloud computing," *IEEE Security & Privacy* (https://ieeexplore.ieee.org/abstract/document/7950859).

AWS. 2016. "AWS Security Best Practices," https://d1.awsstatic.com/whitepapers/Security/AWS_Security_Best_Practices.pdf/accessed September 17, 2018).

Baldini, I., Castro, P., Chang, K., Cheng, P., Fink, S., Ishakian, V., Mitchell, N., Muthusamy, V., Rabbah, R., and Suter, P. 2017. "Serverless computing: Current trends and open problems". In *Research Advances in Cloud Computing* (pp. 1-20). Springer, Singapore, ACM.

Baldini, I., Castro, P., Cheng, P., Fink, S., Ishakian, V., Mitchell, N., Muthusamy, V., Rabbah, R., Slominski, A., and Suter, P. 2016. "Cloud-native, event-based programming for mobile applications," In *Proceedings of the International Conference on Mobile Software Engineering and Systems*, pp. 287-288.

Barga, R. S. 2017. "Serverless Computing: Redefining the Cloud," *First International Workshop on Serverless Computing (WoSC)* Atlanta. (http://www.serverlesscomputing.org/wosc17/#keynote/accessed September 15, 2018).

van Eyk, E., Iosup, A., Seif, S., and Thömmes, M. 2017. "The SPEC cloud group's research vision on FaaS and serverless architectures," In *Proceedings of the 2nd International Workshop on Serverless Computing,* pp. 1-4. ACM.

Fazio, M., Celesti, A., Ranjan, R., Liu, C., Chen, L., and Villari, M. 2016. "Open issues in scheduling microservices in the cloud," *IEEE Cloud Computing*, *3*(5), pp. 81-88.

Fox, G. C., Ishakian, V., Muthusamy, V., and Slominski, A. 2017. "Status of serverless computing and function-as-a-service (FaaS) in industry and research" *arXiv 1708.08028*.(https://arxiv.org/abs/1708.08028 /accessed September 15, 2018).

Google, 2018 "Google Cloud Security Whitepapers," (https://services.google.com/fh/files/misc/security_whitepapers_march2018.pdf /accessed September 13, 2018).

IBM, 2018 "Security to Safeguard and Monitor your Cloud Apps," (https://www.ibm.com/cloud/garage/architectures/securityArchitecture/ /accessed September 13, 2018).

IDC, 2016 "IDC FutureScape: Worldwide IT Industry 2017 Predictions", IDC #US41883016. MA: IDC.

Khan, M. A. 2016. "A survey of security issues for cloud computing," *Journal of Network and Computer Applications*, *71*, pp.11-29.

Lynn, T., Rosati, P., Lejeune, A., and Emeakaroha, V. 2017. "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.

McGrath, G., Brenner, P. R.2017. "Serverless Computing: Design, Implementation,and Performance." In: *2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, pp. 405–410.

Meng, T. 2017. *Security and Performance Tradeoff Analysis of Offloading Policies in Mobile Cloud Computing.* Freie Universität Berlin.

Microsoft, 2018. "Security-as-a Service built for Microsoft Azure," (https://www.alertlogic.com/assets/azure/AlertLogic-Azure-Solution-Overview.pdf /accessed on September 12, 2018).

Narula, S., and Jain, A. 2015. "Cloud computing security: Amazon web service," In *2015 Fifth International Conference on Advanced Computing & Communication Technologies (ACCT),* pp. 501-505.

Puresec, 2018. "Serverless Architectures Security Top 10 (2018)," (https://www.puresec.io/sas-top-10-download / accessed on September 7, 2018)

Sahoo, J., Mohapatra, S., and Lath, R. 2010. "Virtualization: A survey on concepts, taxonomy and associated security issues," In *2010 Second International Conference on Computer and Network Technology (ICCNT),* pp. 222-226.

Singh, S., Jeong, Y. S., and Park, J. H. 2016. A survey on cloud computing security: Issues, threats, and solutions. *Journal of Network and Computer Applications*, *75*, 200-222.

Wallace, Gregory, 2018, " That was no cyberattack on the FCC, inspector general says -- just John Oliver fans." (https://edition.cnn.com/2018/08/08/politics/fcc-website-crash-ig-john-oliver-show-hbo-net-neutrality/index.html 2018 /accessed on September 28, 2018).