Serverless Security

Understand, Assess, and Implement Secure and Reliable Applications in AWS, Microsoft Azure, and Google Cloud

Miguel A. Calles

Apress®

Serverless Security

Understand, Assess, and Implement Secure and Reliable Applications in AWS, Microsoft Azure, and Google Cloud

Miguel A. Calles

Apress®

Serverless Security: Understand, Assess, and Implement Secure and Reliable Applications in AWS, Microsoft Azure, and Google Cloud

Miguel A. Calles
La Habra, CA, USA

ISBN-13 (pbk): 978-1-4842-6099-9 ISBN-13 (electronic): 978-1-4842-6100-2
https://doi.org/10.1007/978-1-4842-6100-2

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Susan McDermott
Development Editor: Laura Berendson
Coordinating Editor: Jessica Vakili

Printed on acid-free paper

Table of Contents

About the Author

Miguel A. Calles is a certified Cybersecurity engineer, works on cloud computing projects, and writes about Cybersecurity. He has worked on multiple serverless projects as a developer and security engineer, contributed to open source serverless projects, and worked on large military systems in various engineering roles. He started in Cybersecurity in 2016 for a US government contract, has been doing technical writing since 2007, and has worked in multiple engineering roles since 2004. Miguel started to gain interest in Cybersecurity when he was in middle school and was trying to reverse engineer websites.

Miguel is a Principal Solutions and Security Engineer at VeriToll, LLC. He has a Bachelor of Science degree in Material Science and Engineering from the Massachusetts Institute of Technology, a Master of Business Administrator degree from the University of Florida, a Cloud Security Alliance's Certificate of Cloud Security Knowledge certification, and a CompTIA A+ certification.

About the Technical Reviewer

David A. Gershman is a Cybersecurity engineer for a government contractor and has the CISSP certification. He has also taught Computer Science at California Polytechnic University, Pomona, on topics ranging from introduction programming to computer networking and Cybersecurity for over 20 years. In his spare time, David enjoys restoring and programming retro 8-bit computers.

Acknowledgments

I would like to express thanks to the following persons and organizations:

- My wife and kids for supporting me in this endeavor.
- My mentor J.R. Richardson for helping me in my professional development and encouraging me to explore new ways to grow.
- David Gershman for introducing me to the field of Cybersecurity and throughly reviewing this book.
- Guise Bule for inviting me to join Secjuice (a blog site that promotes writing about Cybersecurity and information security), where I first started writing about Cybersecurity and serverless computing topics.
- David Huang from Paradigm Sift for his friendship since my college days and helping me troubleshoot a topic in Chapter 8.
- VeriToll (my employer at the time of this writing) for allowing me to write this book and introducing me to the world of serverless computing.
- Raytheon, before they became Raytheon Technologies, for the several years of writing technical manuals and design documents that prepared me for writing my first published book.
- Several teachers that had a lasting impact on my education – Ms. Mary Lang, Mr. Michael Swatek, and Professor Fiona Barnes.
- Apress for allowing me to share what I have learned about Cybersecurity in serverless computing.
- Last but not least, my Creator for helping me achieve a life goal and His provision.

Introduction

When I started working with the Serverless Framework, I was curious about the security aspect. I was transitioning to a project for a mobile app with a serverless back end. Previously, I was an information assurance (IA) engineer working on Cybersecurity for US Government military systems. I had become accustomed to using well-defined processes and requirements in my role as an IA engineer. The systems we were securing were part of a vast network of other systems with strict IA requirements. The threats seemed limited; and implementing Cybersecurity, in many cases, was following a list of checklists and requirements. But, Cybersecurity in the world of serverless development was a new frontier.

The more I worked with serverless, the more I wondered about its Cybersecurity. Cybersecurity with serverless projects seemed to lack the oversight that I experienced in the IA world. The team could release a serverless application without addressing security. I searched for serverless security and found limited information. I did find some helpful documents on the top serverless security risks and well-written blog posts about specific topics. I was looking for a book that provided an overview of serverless security and guidance on approaching it.

I decided to write this book with the intent to fill that void and provide a resource that addressed multiple aspects of serverless security. I leveraged my IA and Cybersecurity experience, my hands-on experience with serverless, and my research to write this book. In one perspective, this book provides an overview of serverless security. You could be new to serverless and learn how to approach serverless security by performing a risk assessment. From another perspective, this book provides practical ways to address serverless security. You could be looking for examples and recommendations to implement in your serverless projects. I am excited to share this book with you because I believe it will guide you in identifying areas of consideration when securing your serverless application.

CHAPTER 1

Introduction to Cloud Computing Security

In this chapter, we will review cloud computing and how its security evolved. We will learn how serverless computing relates to cloud computing and how securing serverless computing differs from the typical cloud computing Cybersecurity. We will review Cybersecurity, how it applies to cloud computing, and why it is needed. This chapter will set the foundation for Cybersecurity in serverless computing by putting it in the context of cloud computing and its security.

Cloud Computing Service Models

Cloud computing is a service offering where a client rents computing resources, physically located in an offsite location, from a provider. The resources are available on demand, and the client accesses them using the Internet. A client can rent resources from networking and storage equipment to fully developed software applications.
Five major service models define how providers make cloud computing resources available to their clients: Infrastructure as a Service (IaaS), Container as a Service (CaaS), Platform as a Service (PaaS), Function as a Service (FaaS), and Software as a Service (SaaS). Table 1-1 depicts how the responsibility of the resource varies among the cloud computing types and compares to the traditional on-premise computing. We will briefly review each cloud computing service model.

M. Calles, *Serverless Security*, https://doi.org/10.1007/978-1-4842-6100-2_1

Table 1-1. *Comparison of Cloud Computing Service Models and On-Premise Computing*

Resource	IaaS	CaaS	PaaS	FaaS	SaaS	On-Premise
Application	C	C	C	C	VR	C
Data	C	C	C	C	V	C
Functions	C	C	C	VR	V	C
Runtime	C	C	VR	V	V	C
Security†	C	C	VR	V	V	C
Middleware	C	C	VR	V	V	C
Databases	C	C	VR	V	V	C
Operating Systems	C	C	VR	V	V	C
Containers	C	VR	V	V	V	C
Virtualization	VR	V	V	V	V	C
Servers/Workstations	VR	V	V	V	V	C
Storage	VR	V	V	V	V	C
Networking	VR	V	V	V	V	C
Data Centers	V	V	V	V	V	C

V = Vendor managed, R = Rentable resource, C = Client managed

†Security resources typically includes security software and appliances. Cybersecurity is essential for each resource type.

Infrastructure as a Service (IaaS)

Infrastructure as a Service (IaaS) is a service offering where a provider makes infrastructure (e.g., networking equipment and computing equipment) available for a client to use. IaaS enables a client to rent infrastructure without having to procure it. The client is responsible for configuring and fine-tuning the different infrastructure components. The provider is responsible for maintaining the infrastructure, making it accessible, and ensuring a minimum level of reliability and availability. This type of cloud computing is the closest to an on-premise model of buying, storing, powering, configuring, maintaining, and administering the infrastructure components, but with the simplified configuration and reduced maintenance and administration.

Container as a Service (CaaS)

Container as a Service (CaaS) is a service offering where a provider makes software container creation and orchestration (e.g., Docker[1] and Kubernetes[2]) available for a client to use. CaaS enables a client to compile all the software packages (needed by an application) into a container without having to set up the infrastructure. The client is responsible for configuring the container and defining the orchestration. The provider is responsible for maintaining the infrastructure, the container virtualization, and the orchestration software. This type of cloud computing provides the benefit of running a lightweight platform without having to set up the infrastructure nor install the orchestration software.

Platform as a Service (PaaS)

Platform as a Service (PaaS) is a service offering where a provider makes a specific platform (e.g., an operating system, a database, and a web server) available for a client to use. PaaS enables a client to rent a platform without having to set up the infrastructure. The client is responsible for configuring and fine-tuning the platform to meet the specific need. The provider is responsible for maintaining the infrastructure, keeping the platform software up to date, and ensuring a minimum level of reliability and availability. This type of cloud computing provides the benefit of defining the computational need without having to determine what kind of infrastructure is needed to power the platform.

Function as a Service (FaaS)

Function as a Service (FaaS) (typically associated with serverless computing) is a service offering where a provider enables a client to run individual software functions and interconnect them to make an application. FaaS allows a client to rent computing

[1]Docker is a registered trademark of Docker, Inc.

[2]Kubernetes is a registered trademark of The Linux Foundation.

time needed to execute the functions without needing to maintain any supporting software and hardware. The client is responsible for writing all the software functions and defining the orchestration among them. The provider is responsible for properly configuring and maintaining the infrastructure and platforms needed to execute the functions. This type of cloud computing provides similar benefits as PaaS and CaaS offerings, but without having to configure the platforms and containers, and enables the client to develop a SaaS offering.

Software as a Service (SaaS)

Software as a Service (SaaS) is a service offering where a provider makes a specific piece of software (e.g., a web application) available for a client to use. SaaS enables a client to rent a piece of software without needing any hardware other than an Internet-connected computing device. The client is responsible for customizing any software settings provided by the application. The provider is responsible for ensuring the web application is available and preventing others from accessing the client's account data. This type of cloud computing provides the benefits of using the software without having to perform maintenance.

Cloud Computing Deployment Models

Cybersecurity was a big concern in cloud computing in its infancy and continues to be one. Cloud computing disrupted the traditional on-premise Cybersecurity model. This new model required different strategies to implement Cybersecurity, and it shared responsibilities with a third-party provider, which reserves the right to secure the system differently than the client desires. Furthermore, the provider not only has to implement Cybersecurity to establish trust with its clients. The provider also needs to secure its offering to protect itself from external threats, which also includes its clients. New models were birthed to accommodate the differing levels of adoption of cloud computing.

The Private or Enterprise Cloud

An enterprise uses a private cloud to have on-premise computing equipment interconnected to on-premise networking equipment. This configuration is referred to as a cloud because the computing equipment interconnects over an intranet (i.e., an internal Internet). Ideally, the data is only accessible within the physical premises of the enterprise for the highest Cybersecurity benefit; see Figure 1-1. An enterprise might choose a private cloud to protect sensitive data.

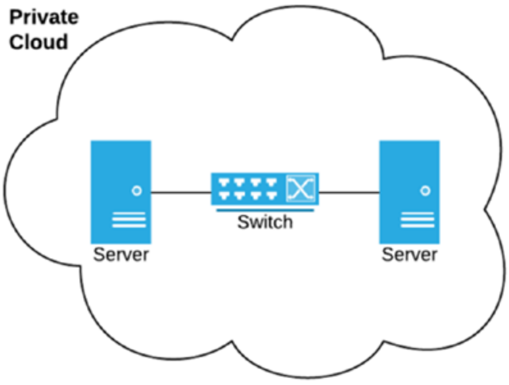

Figure 1-1. *Private Cloud*

A private cloud can have the lowest Cybersecurity risk, assuming proper Cybersecurity measures are in place. The enterprise is mostly or entirely responsible for the Cybersecurity risk. It, therefore, results in higher costs because it must procure, configure, and maintain all the networking and computing equipment and configure and maintain any Cybersecurity measures. The enterprise may favor the private cloud because the higher costs might be lower than those of a Cybersecurity breach, and it has greater control over the Cybersecurity measures.

The Public Cloud

A provider establishes and provides a public cloud to make computing resources available for rent over the Internet. This configuration enables an enterprise to put data in the public cloud and have it accessible from any Internet-connected device; see Figure 1-2. Ideally, Cybersecurity measures protect data by limiting access to only specific parties. An enterprise might choose a public cloud to lower costs, increase accessibility and availability, and offset risk.

Public Cloud
Server
Database
Security

Figure 1-2. *Public Cloud*

A public cloud might have higher Cybersecurity risks because there is no direct purview over the infrastructure and Cybersecurity measures. The provider and the enterprise share the Cybersecurity risk. The enterprise must have the expertise to adequately configure the cloud's Cybersecurity measures and protect its data. The enterprise might favor the shared Cybersecurity risk because it cannot afford to set up and maintain a private cloud, lacks the expertise to secure a private cloud, or prefers faster development and deployment.

The Hybrid Cloud

An enterprise adopts a hybrid cloud to set up private and public clouds to work together. This configuration enables an enterprise to use a private cloud for its more sensitive data and a public cloud for its less sensitive data; see Figure 1-3. It further allows taking advantage of both sets of features and computing capabilities of both clouds. An enterprise might choose a hybrid cloud to meet legal and contractual requirements, lower costs, and configure varying levels of Cybersecurity measures.

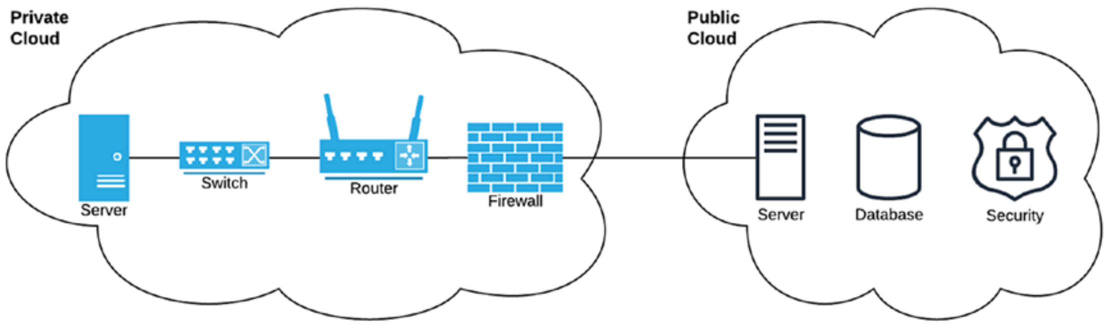

Figure 1-3. *Hybrid Cloud*

The hybrid cloud might be the best of both worlds in some situations. Still, it potentially has a higher Cybersecurity risk than a private cloud and not necessarily a lower risk than a public cloud. We should use properly configured private cloud security equipment (e.g., firewall systems, intrusion detection/prevention systems, and security information and event management systems) to establish a connection between the public and private clouds. The connectivity between the private and public clouds presents an opportunity for the bypassing of security equipment and exposing the data within the private cloud. The enterprise might favor the increased Cybersecurity risk for several reasons: it wants to take advantage of features within the public cloud; it has several layers of Cybersecurity measures to mitigate the risk of the external connection; it has multiple private clouds; the public cloud only has access to a limited set of private clouds.

Applying a Cloud Computing Model to FaaS

FaaS can support all three deployment models. FaaS was initially introduced as a public cloud solution because it reduces most of the configuration and maintenance effort. As the FaaS offering matured, providers added the ability to access a private cloud from a FaaS solution. The industry realized the need for having FaaS within a private cloud, and it created a FaaS solution that runs on software containers installed on servers within a private cloud. In this book, we will mostly explore Cybersecurity in the public cloud.

An Overview on Cybersecurity

Cybersecurity, or security for short, is the practice of identifying the assets that need protecting, the threats against those assets, and the defenses needed to protect those assets. Many engineers, developers, and managers have become accustomed to implementing security in traditional on-premise systems: desktop computers, laptops, servers, networking equipment, operating systems, and so on. The cloud computing era disrupted how companies and individuals view their assets. Consequently, the practice of security had to evolve to work in this new computational method. Now that the assets and infrastructure are provided by a third party, the cloud computing provider and the client share the responsibility for implementing security.

We can summarize security and its implementation in three words: confidentiality, integrity, and availability. Using the confidentiality, integrity, and availability (CIA) model (sometimes referred to as the CIA triad) is one way to identify the security risks and security measures needed to mitigate those risks. We will explore each element.

Confidentiality

Applying confidentiality to a piece of data is giving access only to the intended recipients. Said another way, confidentiality is preventing unauthorized access from unintended recipients. A common term in recent news is “privacy.” An enterprise may choose to implement confidentiality using encryption and access control.

Data has no encryption by default. Applying encryption to data prevents access to it. The data is encrypted using a key, and only that key can decrypt the file to return it to its original state. The key can be a password, file, or certificate. The encryption should happen while the data is at rest (i.e., while it sits in the file system) or while it is in transit (e.g., being transferred over the Internet).

Data has no access control by default, but modern operating systems do implement some level of access control. Access control defines which data is accessible to others and how that data is used. In an operating system that supports it, the access control determines whether the current user can read, modify, or execute the data and also defines whether other users can have similar privileges. It might also allow specifying a subset of users that can read, modify, or execute the data.

FaaS solutions provide encryption and access control. The account owner needs to enable shared access or public access; the account owner is the person or entity that manages the account on the public cloud. The data owner can assign read, modify, and delete privileges to the data; the data owner is the person or entity that manages the data stores in the public cloud. The account owner is responsible for configuring the cloud infrastructure to set the desired level of confidentiality. The cloud infrastructure provides encryption for data in transit, data at rest, and access control to the data owner and others. The provider's cloud infrastructure only gives the account owner access to the data. Cloud infrastructure supports encryption in transit and at rest.

Integrity

Ensuring integrity for a piece of data is giving confidence the data someone sent you is the same data you received. Said another way, integrity is making sure there are no unintended modifications to the data, and the intended recipient has trust they received the expected data. The enterprise may choose to implement integrity using checksums, version control, or logging.

A checksum is a representation of the data and is used to determine whether the file has changed since it was last accessed. For example, when a user creates a file, the system records its checksum. When the user modifies the file, the checksum also changes. The user or file system can use the checksum to determine whether the file has changed.

Whenever a user creates, modifies, or deletes a file, a version control system or a logging system captures the change. The version control system saves a copy of the file for each version (and sometimes a checksum). In contrast, a logging system records the type of change, the user who invoked the change, the time the change occurred, and other relevant information.

FaaS solutions provide integrity solutions natively and as an add-on feature. The account owner is responsible for configuring the cloud infrastructure to set the desired level of integrity. The owner can also enable logging systems to capture changes to the file and add checksums to the different versions of the data. The cloud infrastructure supports version control of files. The cloud infrastructure natively does file replication at the hardware level while maintaining the data integrity.

Availability

Providing availability for a piece of data is using measures to ensure intended recipients can use the data. Said another way, availability is making sure the intended recipient can access the data at any time. The enterprise may increase availability through maintenance, replication, and redundancy.

Performing maintenance ensures the hardware hosting the data continues operating as long as possible without interruption. For example, if a user stores a piece of data on one piece of equipment, and it stops functioning, that data is no longer available for a user to access. Had that unit been adequately maintained, it could have continued operating longer, or the maintainer could have observed symptoms of imminent failure. Therefore, it is essential to maintain hardware to keep it running to increase availability.

Replication and redundancy create replicas of data on other pieces of hardware. For example, in the event one unit fails, others make the data available for a user to access. An enterprise will use hardware components (e.g., Redundant Arrays of Independent Disks, or RAIDs) to provide local, built-in redundancy and data backup software to achieve geographical (offsite) redundancy.

FaaS solutions provide availability natively when storing data in the public cloud, which has a minimum level of guaranteed availability. The account owner is responsible for selecting a cloud infrastructure with the desired minimum availability and configuring any additional availability features. For increased availability, the data owner can choose to replicate the data across multiple geographic locations within the public cloud infrastructure. Using cloud infrastructure eliminates the need to perform routine hardware maintenance. However, regular checks of the account configuration and data access are still warranted.

The Need for Cloud Computing Cybersecurity

Approaching Cybersecurity is similar, yet different, in public and hybrid clouds vs. a private cloud. The enterprise has more control and influence of the security measures in a private cloud. The security measures are implemented based on the risks identified in an assessment. The enterprise should assess public and hybrid clouds similar to a private cloud, but with the understanding that the threats vary.

Examples of Threats

Threats exist in the three cloud computing models and manifest themselves in several ways. We will explore a few examples of how threats manifest.

Data Breaches from Insecure Data Storage

Since cloud storage configurations support private, shared, and public access, it is probable public access was set unintentionally.[3] For example, an attacker can use an improperly configured cloud storage system to access highly sensitive data. An inexperienced user may accidentally grant public access while attempting to limit sharing to a small group. A user may also temporarily give public access to transfer data to other parties, but forget to revert to private access. Data breaches can result from an improperly configured cloud storage system.

Data Breaches from Identity and Access Management Misconfiguration

Someone can access another person's account if the Identity and Access Management (IAM) system has a misconfiguration. The data owner might use an IAM system to share data access with multiple users. Shared access should be limited to the users that require the data and no one else. For example, the finance team should only have access to confidential financial records, and not the engineering team or suppliers. Data breaches have occurred because a supplier had access to a network where sensitive data was processed.[4] Data breaches can result when one account is compromised, and it has access to data it should not.

Denial of Service Attack Due To Software Vulnerabilities

Any application exposed to the Internet is vulnerable to Denial of Service (DoS) or Distributed Denial of Service (DDoS) attacks. Cloud services limit how much computing power a client can use at any given time. An attacker hopes to exploit a weakness in the application by sending multiple simultaneous requests and making the application unavailable to the users. This downtime can result in financial loss and lost productivity.

Weaknesses can exist at any level. For example, an attacker can exploit a software library with a known vulnerability by sending a large piece of data such that the

[3]"100GB of secret NSA data found on unsecured AWS S3 bucket. 29 November 2017. Adam Shepard. IT Pro. www.itpro.co.uk/security/30060/100gb-of-secret-nsa-data-found-on-unsecured-aws-s3-bucket

[4]"What Retailers Need to Learn from the Target Breach to Protect against Similar Attacks." January 31, 2014. Chris Poulin. Security Intelligence. https://securityintelligence.com/target-breach-protect-against-similar-attacks-retailers

application takes a significant time to process the entire data or eventually times out.[5] If thousands or millions of requests are sent simultaneously to a vulnerable software function, the application may stop responding for all users and result in a DoS to the user base.

Identifying Threats

The three previous examples illustrate the realization of threats. Your understanding of the threats to your application will help you determine how to protect against them. We will explore how to identify threats in the next chapter.

Key Takeaways

In this chapter, we reviewed cloud computing and Cybersecurity. This chapter aimed to provide a foundation for the remainder of this book. We established concepts and terminology in cloud computing. We will briefly review these concepts and terms.

We explored cloud computing service models:

- **Infrastructure as a Service (IaaS)** is using infrastructure (e.g., computing and networking equipment) over the Internet.
- **Container as a Service (CaaS)** is using a software container (e.g., Docker) over the Internet.
- **Platform as a Service (PaaS)** is using a configured platform (e.g., a database) over the Internet.
- **Function as a Service (FaaS)** is running and orchestrating functions (e.g., an email subscription function) over the Internet.
- **Software as a Service (SaaS)** is using an application (e.g., a web-based email) over the Internet.

We covered cloud computing deployment models and how FaaS supports them:

[5]"Serverless Security & The Weakest Link (Avoiding App DoS)." 8 February 2019. Ory Segal. PureSec Blog. www.puresec.io/blog/serverless-security-and-the-weakest-link-or-how-not-to-get-nuked-by-app-dos

- **Private cloud** is where an enterprise uses computing equipment it acquired and accesses it over an internal network. An enterprise can set up an internal FaaS solution on its hardware.
- **Public cloud** is where an enterprise uses computing equipment from a third party and accesses it over the Internet. An enterprise can use a provider's FaaS solution.
- **Hybrid cloud** is where an enterprise uses private and public clouds for different purposes and uses security equipment to interconnect them to minimize risk. An enterprise may configure a private FaaS solution to access data from a public cloud and vice versa, given the security equipment on both sides are configured to enable access.

We learned the confidentiality, integrity, and availability model in Cybersecurity and how FaaS supports all three.

- **Confidentiality** is ensuring only the desired recipients can access a piece of data. FaaS ensures confidentiality by limiting data access to the account owner with access control systems and by using encryption.
- **Integrity** is ensuring the data was unchanged and uncorrupted from the last time it was accessed. FaaS provides integrity with version control systems and logging systems.
- **Availability** is ensuring the intended recipient can access the data without disruption. FaaS provides a minimum level of availability and increases with replication across geographical regions.

We reviewed examples of Cybersecurity threats to depict the need for Cybersecurity in cloud computing.

In the next chapter, we will examine how to assess a FaaS application and perform a security risk assessment.

CHAPTER 2

Performing a Risk Assessment

In this chapter, we will learn how to perform a risk assessment for a serverless application. We will explore how to understand how the application works, which includes reviewing documentation, source code, and system accounts and using the application. We will discuss why we scope the risk assessment. We will learn how to develop a threat model and how to use it to start creating the risk assessment.

Conventions

We will review the conventions used throughout this book. For clarity, we will use one example application throughout. We might deviate from this example application at times when it makes sense to explain a concept better. We will use one FaaS framework (or typically referred to as a serverless framework) for consistency, except where it lacks support for a security configuration we are learning or when we can better learn a principle by directly modifying the configuration. For simplicity, we will use one programming language in the examples because it may become overwhelming to cover the same principle in all programming languages supported by the serverless provider and framework. The goal is to ensure an optimal experience in learning security concepts with less focus on prescriptive approaches for implementing them.

Example Serverless Application

Throughout this book, we will use a fictitious ecommerce mobile app in the examples. This app allows users to buy and sell goods using a mobile app. The app brokers the transactions to ensure both buyer and seller are protected. The mobile app communicates to an Application Programming Interface (API) to execute the transactions.

M. Calles, *Serverless Security*, https://doi.org/10.1007/978-1-4842-6100-2_2

The serverless framework will create the API. The serverless application will integrate with other third-party services, which provide additional capabilities. The examples and exercises reference this fictitious application but do not provide a fully functioning system.

Serverless Frameworks

The three major FaaS and serverless providers are Amazon Web Services (AWS), Microsoft Azure,[1] and Google Cloud.[2] You can manually set up functions using their web-based consoles. You can choose an automated way to deploy the functions by leveraging FaaS or serverless frameworks. There are several serverless frameworks, which support different programming languages and multiple providers. We will focus on a framework that supports AWS, Azure, and Google Cloud.

We will use the Serverless Framework[3] in this book, not to be confused with the term "serverless framework." Serverless, Inc. created a serverless framework that supports AWS, Azure, Google Cloud, and other serverless providers. The Serverless Framework is written in Node.js[4] JavaScript and has open source and paid versions. At the time of this writing, the open source project has over 30,000 stars, forked over 3000 times, and is actively maintained.[5] For these reasons, we will use the Serverless Framework throughout this book.

Programming Language

The Serverless Framework was built using Node.js and exists as a package in the npm[6] package manager. npm is arguably the fastest growing package repository with at least one million packages at the time of this writing.[7] The popularity is probably a result of JavaScript being one of the easiest programming languages to learn.[8] We will use Node.js

[1]Azure is a registered trademark of Microsoft Corporation.

[2]Google and Google Cloud are registered trademarks of Google LLC.

[3]Serverless Framework is a registered trademark of Serverless, Inc.

[4]Node.js is a trademark of Joyent, Inc.

[5]Serverless GitHub repository. https://github.com/serverless/serverless

[6]npm is a registered trademark of npm, Inc.

[7]Module Counts website. www.modulecounts.com

[8]"The 10 easiest programming languages to learn." 17 July, 2017. Alison DeNisco Rayome. TechRepublic. www.techrepublic.com/article/the-10-easiest-programming-languages-to-learn

in the examples for ease of understanding, compatibility with the Serverless Framework, and the runtime engine support from AWS, Azure, and Google Cloud.

Terms, Keywords, and Acronyms

This book will use different terms, keywords, and acronyms throughout this book. They are defined in their first use and may be defined again if it aids in understanding and helps avoid confusion. Table 2-1 lists terms, keywords, and acronyms repeatedly used throughout the book.

Table 2-1. *Terms, Keywords, and Acronyms Used Throughout the Book*

Term, Keyword, or Acronym	Definition
API	Application Programming Interface
AWS	Amazon Web Services, a serverless platform
Azure	Microsoft Azure, a serverless platform
CLI	Command-Line Interface
Google Cloud	A serverless platform
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
JavaScript	A programming language
Node.js	A JavaScript runtime environment
npm	A package manager for Node.js
OS	An acronym for operating system
Serverless	Short for Serverless Framework, a serverless framework
serverless	Another term for Function as a Service
serverless.yml	A configuration file used in the Serverless Framework
sls	A CLI command for the Serverless Framework