May 14, 2019 By **Sumo Logic**

# Improving Security in Your Microservices Architecture

As companies increasingly move to microservices architecture, companies are discovering the challenges that securing microservices can pose. Learn about security in a microservices framework and best practices to ensure your architecture is secure.

Sear

**Categories**

DevOps & IT Operations

**Spotlight**

Continuous Intelligence Report

## What are Microservices?

A microservice architecture, often referred to simply as microservices, is a teams, as it facilitates continuous delivery for large applications and adapts easily to a company's needs as its technology evolves and scales up.

Applications & DevSecOps in the Cloud - 2018

Share

## Microservice Architecture versus Monolithic Architecture

The microservices architecture contrasts with the traditional monolithic application model. Monolithic applications are structured as a single tier, making them easier to stand up quickly and integrate reliably with well-known integrated development environment (IDEs), frameworks, and tools. However, as a monolithic application ages, its shortcomings begin to show.

As engineers supporting the application come and go, they take with them intimate knowledge of the application's interdependencies. This makes it increasingly difficult to move development forward at the pace an organization needs.

meaning they can move quickly to continue development.

## Security Challenges of Microservices

Yet microservices are not a magic bullet, and implementing a microservices architecture in an organization brings with it some unique challenges. Specifically, the deployment method requires new approaches to development, security, and operations delivery. Applications built from microservices add complexity and moving parts, and the optimal approaches for managing monolithic applications no longer apply.

Looking specifically at security, effective methods for microservices are substantially different than those that work for monolithic applications. For the latter, security teams often use centralized security modules, which cover authentication, authorization, and a range of other critical security measures.

efficiency. In fact, most traditional, host-based network security tools do not offer the ability to monitor activity inside microservices containers.

## API Gateways

To make a microservices-based application usable, an organization needs to establish how users will access the services within it. While it is possible, in theory, to structure the application to make direct calls to each service, this can lead to highly complex code involving an overwhelming number of service calls. Instead, many companies establish a dedicated server to function as an API gateway, providing a single point of entry and directing traffic into the different services.

Security within an API gateway calls for more scalable methods than centralized session management. Ensuring that a user is who they claim to be and that they are allowed access to a service, these gateways typically handle authorization and authentication at the microservice level. To keep their solutions efficient, security teams need

### Public and Private APIs

In general, APIs play a central role in microservices, connecting the different components together and allowing them to interact. These APIs come in two main types: public and private. Public APIs are those that consumers use to access a resource or service through an app, while private APIs are those that the development team uses within larger applications to communicate between owners of different services.

Securing both public and private APIs a must. With the distributed nature of microservices architecture, this can be a major challenge, as security teams lose visibility of APIs that are rapidly changing and expanding.

## Building Secure yet Scalable Microservices

While microservices are easy to set up and deploy across different platforms, security often has a difficult time keeping pace with the increased surface monitoring and complicates access rules. And, in addition to all that, many microservices run inside cloud environments with their own types of security controls.

With the security services market catching up to the rapid adoption of microservices, a gap currently exists for available security solutions. With an abundance of exposed ports and APIs, microservice architectures cannot be managed effectively with traditional firewalls that establish a perimeter around a network of connected servers. This new arrangement calls for a distributed approach to securing the attack surface, and the entire development, operations, and security team needs to be on board.

## Creating a Microservices Security Strategy

With security being an increasingly complex challenge for organizations switching to microservices, a cultural shift and a new mindset are necessary

DevSecOps arrangement that prevents security from taking a back seat to the development of new capabilities. Teams can use security principles within the development of their code and have their code peer reviewed for security concerns prior to deployment.

Of course, there are also a number of architectural considerations for deploying a secure microservices model as well, which we will explore next.

## Securing Access Points With OAUTH/OAUTH2

Most applications within a microservices architecture require methods for controlling access and authorization. Some organizations with very specific security needs build their own authorization protocols to handle this requirement.

However, many security analysts advise against starting from scratch and instead recommend using OAuth and OAuth2 for authorization management — unless your organization has a burning need to go in another direction. As the industry standard for user

## Building Strong Microservices Defenses

A standard firewall on the network perimeter is insufficient for protecting an organization's software architecture, so microservices calls for a more robust defense involving multiple layers of security controls, placed throughout the information technology system.

This means that teams need to identify the most sensitive areas of their services and make sure these areas are afforded several layers of security protection. Attackers who are able to exploit a single layer will still be tasked with overcoming multiple levels of protection in the system.

Securing critical services is a task that requires expertise in cybersecurity, so it's not something that should be left to developers or operational IT staff. And, fortunately, the microservice architecture makes this fairly simple to achieve. Because applications are broken down into multiple components, security professionals can hone their skills and resources at a granular level,

## For Microservices Security, Think Automatic and Atomic

The wide distribution and granularity of a microservice architecture can make it difficult to scale security solutions manually to cover all of the services within. This is why it's crucial, at the outset of a microservices build out, to establish some form of automation for scaling security controls.

As an organization updates parts of its system, it needs to test it to catch any issues throughout testing. All components should be wrapped within a container so that testing that application only requires wrapping another container around it.

## Using API Gateways for Microservices

An API gateway is one strategy for easily managing multiple interfaces into services. And this strategy can enable all of your microservices. With effective management of authorization and authentication, this can add a scalable layer of protection to the attack surface.

As microservices gain traction among development teams, organizations are able to deploy new applications and services at a rate they could not achieve with traditional, monolithic applications architecture. Along with these advancements comes the necessity for new approaches to security, as standard network tools, firewalls, and central monitoring resources fall short in scalability.

If teams can effectively establish a DevSecOps for building security into the development process, deepen and distribute their security programs, and efficiently manage authorizations and authentications through an API gateway, they can keep their security up to speed with the requirements of updated, containerized application development processes.

The most appropriate architecture for an organization comes down to understanding the application, the team, and even the organization. The