International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Spring Security Framework in Microservices Architecture- Implementing Gateway Integration

Hareesh Kumar Rapolu

hareeshkumar.rapolu@gmail.com

Abstract

Microservices are widely used in many applications every day. It is also easy to work on separate applications instead of following a monolithic culture. A secured framework is essential for Microservices architecture to ensure authentication and authorization across APIs. Spring Security is a popular framework for securing applications against common gateway attacks. This paper provides a detailed overview of Spring Security operations for protecting microservices and various challenges associated with these frameworks.

Keywords: Microservices, Spring Security, Cloud, API, Authentication and Authorization, JWT token, Encryption, Infrastructure, Spring Annotations

I. INTRODUCTION:

The security of microservices is essential for preventing data breaches and maintaining the integrity ofsystems. It improves software quality and mitigates risks acrossdiversesectors, such as healthcare, retail, andbanking. Adhering to security protocols within each service is crucial, as we encountervarious challenges increating a secure gateway. Authentication and authorization are pivotal techniques forsafeguarding system integrity and protecting data from unauthorized access. Security is imperative for microservices to communicate efficiently through a gateway,such as an API (Application Programming Interface), therebyensuring the protection of applications. This methodology significantly reduces the risk of malware attacks and security breaches.

II. BACKGROUND OF MICROSERVICES ARCHITECTURE:

Microservicesare a superior choice to traditional monolithic architecture as they can be deployedindependently. Hence, they provide scalability, flexibility, and easier-to-maintain systems. Even though it looks simpler, it is a much more complex network interaction model[1]. The independent smaller services communicate with each other through APIs synchronously or asynchronously. It is essential to conceal the service's complexities and implementation specifics while only revealing what is necessary for its clients. This independent nature offers isolation where, in case of error, only a single microservice is stopped instead of the whole service. Spring Boot is an open-source framework used to build and develop microservices, and it has pre-configured features that make it easier to create stand-alone applications and deployments. Many big organizations like Netflix and Amazon have adopted microservices[2]. Other organizations also adopted microservices to replace their monolithic applications

by increasing the demand and success rate. Microservices can offer unique advantages compared to all other techniques.

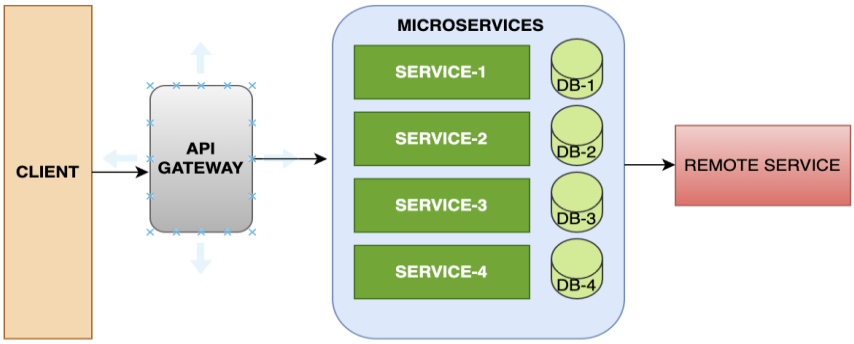

Figure 1: Microservices Sample Architecture

III. WHY SECURITY IS NEEDED FOR MICROSERVICES:

Microservices architecture has many individual services, making it easy for attackers to access any service. When sensitive data is distributed across all the microservices, it needs proper protection or authorized access. So, a secured connection is mandatory for all microservices to maintain successful communication. Security practices are important in various fields, such as retail industries, e-commerce, medical records, and various small and large industries. A medical journal written on security practices combines medical devices and applications to use the Spring framework for securing their medical data. They adopted a solution for data security (SFTSDH = SF + TSD + H)[3] to protect their APIs in the healthcare industry. Hence, security is important to maintain customer trust, reputation, system integrity, and data breaches. Every microservice needs a unique set of security requirements for efficient processing and productivity.

IV. KEY CONCEPTS OR FEATURES OF SECURITY ARCHITECTURE IN MICROSERVICES:

A) Authentication and authorization Mechanisms:

We can provide Authentication to microservices fromthree perspectives:security while communicating from one microservice to another,security when external services connect to microservices through API,andsecurity when end users access the application. Many applications
use external services like databases, the cloud, etc. Hence, security plays a key role in improving communication within these services.

- **Token-based authentication:**

It is a stateless authentication technique where the user sends credentialstotheserver. The server validates the request, generates a key or token (JWT), and sends it back to the client. When communicating with the server, the client includes this token in the request body. The server then verifies the request to authenticate the user. JWT stands for JSON Web Token,whichhas aformat divided into three parts: Header, Payload, and Signature[4].

- **Edge level authentication:**

Here, the identity or user is validated at the entry point. Authentication at the edge includes auth 2.0, TLS, etc. We can use an API gateway to authenticate and authorize all downstream microservices. So, only authorized requests reach the internal microservices, enhancing security.

- **Service level authentication:**

This techniqueprovides security or authorization inside each microservice. It works alongside authentication by guaranteeing that even verified entities can only carry out actions for which they have been specifically granted permission. It has various approaches, such as role-based access control (authentication is based on roles), attribute-based access control (authentication is based on user and resource attributes), and context-aware policies (which consider the context to approve or deny access).

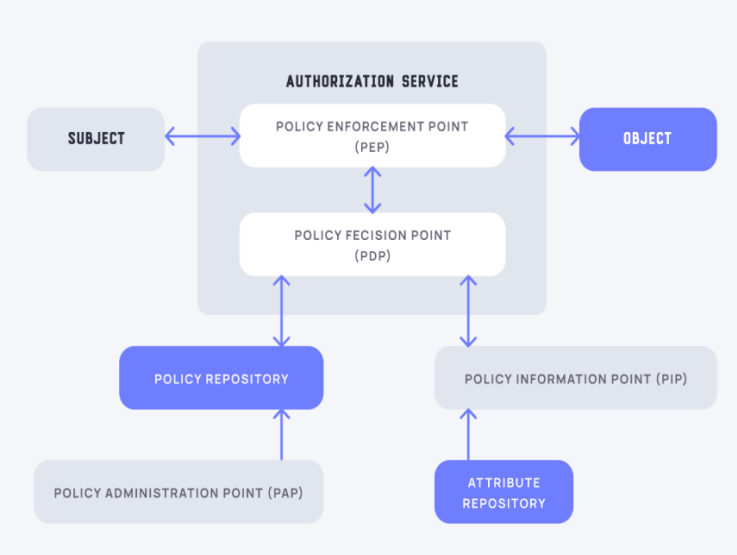

Figure 2: Service Level Authentication

- **Single Sign-On (SSO):**

SSO allows users to access multiple services with a single sign-in. The SSO generates a token based on user credentials to validate the user. If the user tries to access various services within the SSO system, the application checks the token and grants access to all the services without needing to log in again. Thus, this method allows users to access all the services at once. SSO follows protocols like SAML (Security Assertion Markup Language), OpenID, and OAuth.

- **Two-Factor Authentication:**

Thistechniqueiswidely utilized across various industries. Thereare two methods of authentication.For instance, we use our credentials whenlogginginto our company website. Some authorities haveintroduced an additional layer of securityknown as an authenticator app, whichsendsus a time-sensitive code to log in. This method is implemented in many areas,including ATMs[5], biometrics, and one-time codes for certain login requests.

B) Designing a secured microservice:

Thisenablesusto integrate our design with various security principles at every stage of microservice development. Several architectural principles can be employed to establish security measures.

- **Service Division:**

This means that services will operate separately or in isolation within the architecture without communicatingwitheachother, avoiding traffic unlessnecessary. The division can be based on the functionality of the services. For example, the public zone can handle external-facing APIs such as login pages, while the Protected Zone manages internal APIs and handles sensitive data like databases.
Additionally, to achieve service division, other divisions,such as Network segmentation, services meshes, firewall rules, Kubernetes namespaces, and API gateways, can be included in the design.

C) Use HTTP methods:

Microservices can communicate with each other using HTTP protocol.

- **Obtaining SSL certificates:**

By utilizing TLS or SSL certificates, we can create a secure connection. ACM (Azure Certificate Manager) can also manage cloud-based certificates. Foradded security, these certificates can be encrypted using various tools like Certbot, DigiCert,and others.

- **Https for internal communication:**

For internal service-to-service communication, a mutual TLS setup can be used for each service to verify each other's identity. The services can then be configured with TLS certificates. CSRs (certificate signing requests) can be established for each service to authenticate the request.

- **Modern standard encryptions:**

Modern standards incorporate more secure practices, protocols, and configurations. TLS protocols such as TLS 1.2 and TLS 1.3 guarantee the most recent cryptographic algorithms. Only a few cipher suites providestrong encryption algorithms like AES-GCM.

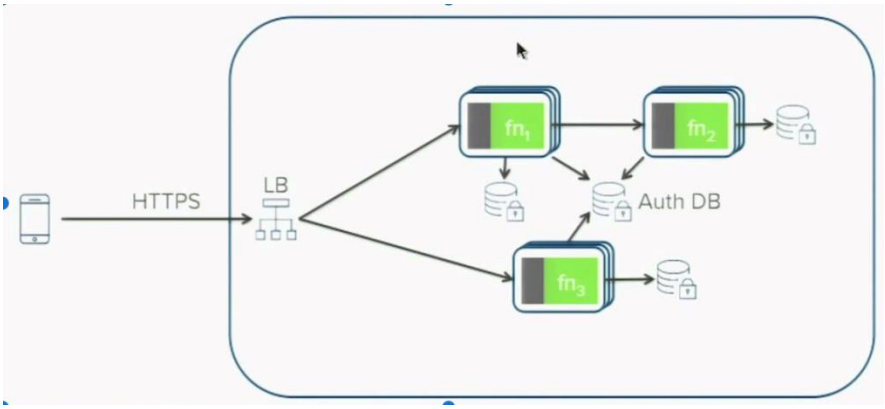

Figure 3: Serverless Microservices Architecture with Load Balancing and Authentication

D) Providing security to Cloud services:

Cloud Security has various layers, such as Data security, infrastructure level security, and IAM (identity Access Management).

- **Data security:**

Encryptions such as Azure Key Vault, AWS KMS, and GCP cloud KMS were implemented according to the cloud we are using.

- **IAM:**

It also representsthe lowest level of access for users. This access can be implemented using RBAC. IAM SAML allows identity providers to authenticate by exchanging user information,such as login details, authentication status, credentials, and related attributes between service providers and identities[6].

- **Infrastructure security:**

Infrastructure security includes cloud environmental components such as networking (internal or external communication) and storage (Amazon S3, Amazon EBS).

E) Using Spring annotations:

Few annotations provide security even before the code implementation automatically, like @Secured, @PreAuthorize, and @PostAuthorize in the spring application, to provide security to particular services and role-based access to particular users before or after the method execution.

V. CHALLENGES WHILE SECURING MICROSERVICES:

There are more security threats in microservices than in monolithic applications.

A) **Data consistency:**The main challenge is securing the data along various databases where it is stored.

B) **Multiple cloud deployments:** Deploying microservice on multiple cloud environments and distributed over various data centers

C) **Increased attacks:** There are more attacks than a monolithic application, as these are several individual services.

VI. CONCLUSION

Integrating Microservices with Spring security enhances performance and eliminates minor to major security issues. Companies use various suitable security methods based on their projects. By Implementing the security framework, we can achieve authentication and authorization for all the microservices.

Abbreviations and Acronyms:

- API- Application Interface
- IAM- Identity Access Management
- ACM- Azure Certificate Manager
- TLS- Transport Layer Security
- SSL- Secure Sockets Layer
- EBS- Elastic Block Storage
- GCP- Google Cloud Platform
- RBAC- Role-Based Access Control
- JWT- JSON Web Token
- KMS- Key Management Service
- SAML- Security Assertion Markup Language

- SF- Spring Framework
- TSD- Services for Sensitive Data
- H- Hyper Text Transfer Protocol(HTTPS)
- AES-Advanced Encryption Standard
- GCM- Galois/Counter Mode

VII. REFERENCES

[1] Baker, O., & Nguyen, Q. (2019). Applying Spring Security Framework and OAuth2 To Protect Microservice Architecture API. *Software*, *14*, 257-264. https://doi.org/10.17706/jsw.14.6

[2] IRV, R. (2016). Spring Microservices- Build scalable microservices with Spring, Docker, and Mesos (p. 5). Packt Publishing Ltd. https://books.google.com/books?id=pwNwDQAAQBAJ&printsec=frontcover&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

[3] Chatterjee, A., Gerdes, M. W., Khatiwada, P., & Prinz, A. (2022). Applying Spring Security Framework With TSD-Based OAuth2 to Protect Microservice Architecture APIs. *IEEE*, *10*, 41914-41934. https://doi.org/10.1109/ACCESS.2022.3165548

[4] de Almeida, M. G., & Canedo, E. D. (2022). Authentication and Authorization in Microservices Architecture: A Systematic Literature Review. *Applied Sciences*, *12*(6), 3023. https://doi.org/10.3390/app12063023

[5] Knutson, M., Winch, R., &Mularien, P. (2017). Spring Security: Secure your web applications, RESTful services, and microservice architectures (3rd ed.). Packt Publishing Ltd. https://books.google.com/books?hl=en&lr=&id=MkBPDwAAQBAJ&oi=fnd&pg=PP1&dq=spring+security+authentication+and+authorization+mechanisms+for+microservices&ots=mX7LB3jyDN&sig=XKSAoOufLRXsRQMXtslQGOIodG0#v=onepage&q=doi&f=false

[6] Singh, C., Thakkar, R. ., &Warraich, J. . (2023). IAM Identity Access Management—Importance in Maintaining Security Systems within Organizations. European Journal of Engineering and Technology Research, 8(4), 30–38. https://doi.org/10.24018/ejeng.2023.8.4.3074