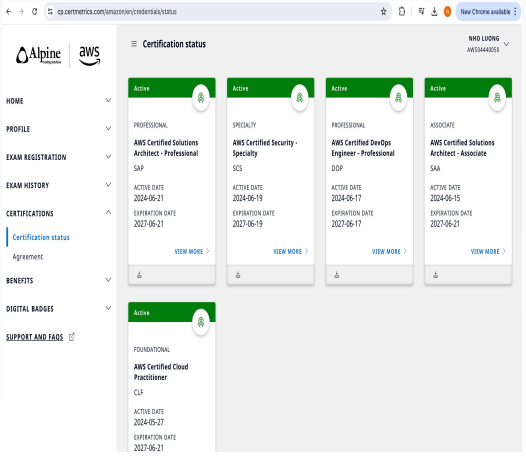
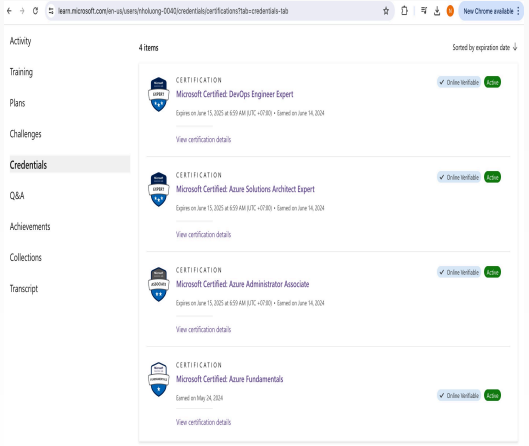
# Securing Microservices Using WAF

Author: Nho Luong
Skill: DevOps Engineer Lead



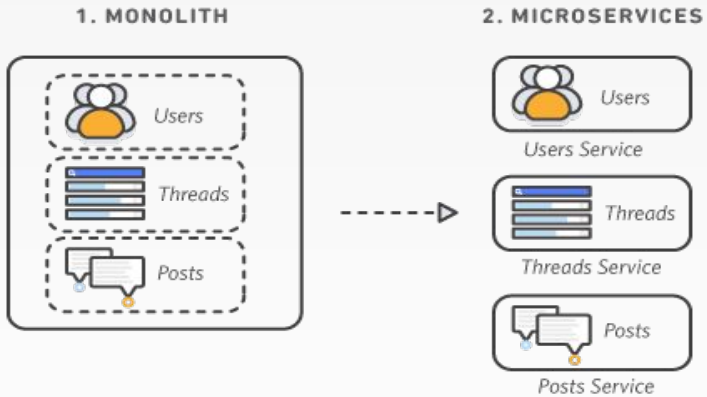


# Microservices

Microservices - also known as the microservice architecture - is an architectural style that structures an application as a collection of services that are:

- Highly maintainable and testable
- Loosely coupled Independently
- deployable Organized around business
- capabilities Owned by a small team
-




The microservice architecture enables the rapid, frequent and reliable delivery of large, complex applications

.

Author: Nho Luong
Skill: DevOps Engineer Lead

# Securing Microservices

- Checking for common cluster configuration errors
- Configuring encryption for cluster components Using
- trusted container image repositories Configuring access
- controls to the cluster components Checking for common
- security issues in code during build Controlling container
- privilege Encrypting inter-microservice traffic ...
-
-

Author: Nho Luong
Skill: DevOps Engineer Lead

# What is missed?

- Controlling East/West traffic (layers 3 and 4) between Pods within the cluster (aka micro-segmentation) Deep inspection of the application traffic (layer 7) between Pods within the cluster
- (aka IPS or WAF)

The attack surface area can include vulnerabilities buried deep inside the application architecture that can be exploited. These include well-known OWASP top 10 web application attacks such as Cross-site Scripting (XSS), SQL Injection, Remote Code Execution (RCE), API attacks, and more.

Author: Nho Luong
Skill: DevOps Engineer Lead

# ModSecurity

What can ModSecurity do?

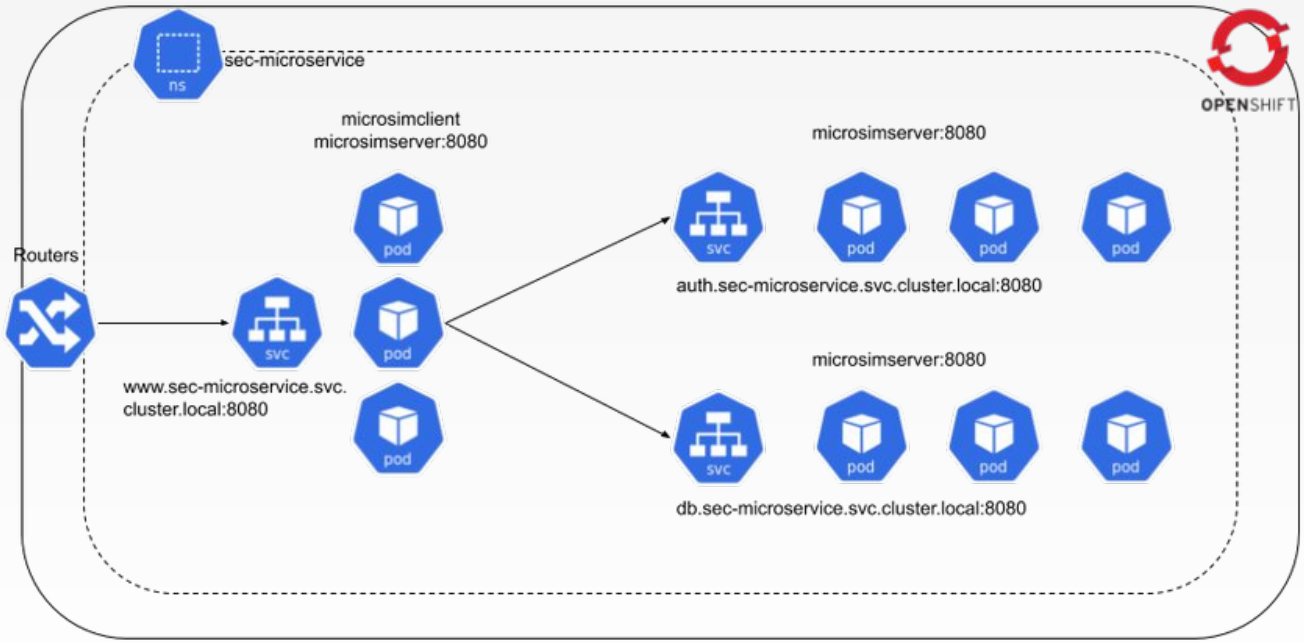
- Real-time application security monitoring and access control
- Virtual Patching Full HTTP traffic logging Continuous passive
- security assessment Web application hardening
-
-

Main functionalities:

- Parsing
- Buffering
- Logging
- Rule Engine

Author: Nho Luong
Skill: DevOps Engineer Lead

# Insecure Design Pattern




Author: Nho Luong
Skill: DevOps Engineer Lead

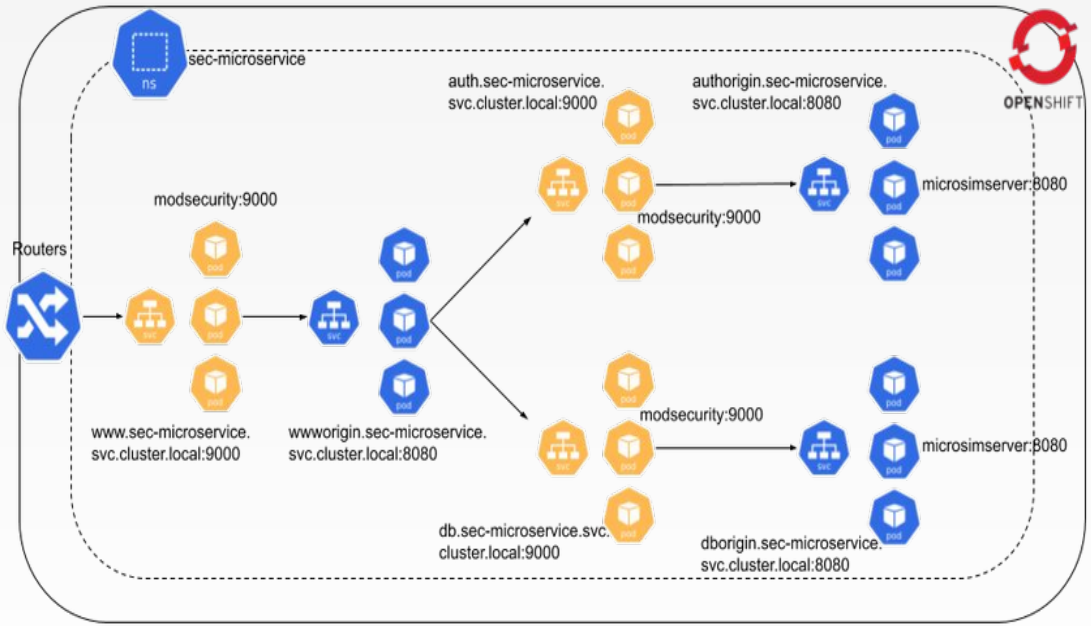
# Service Layer Pattern

Pros:

- Allows scaling of the security tiers independent of the microservices they are protecting
- Treats application security as a microservice
- No need to change microservice ports

Cons:

- Creates additional services in the cluster
- Adds traffic flow complexity
- Requires more micro-segmentation rules




Author: Nho Luong
Skill: DevOps Engineer Lead

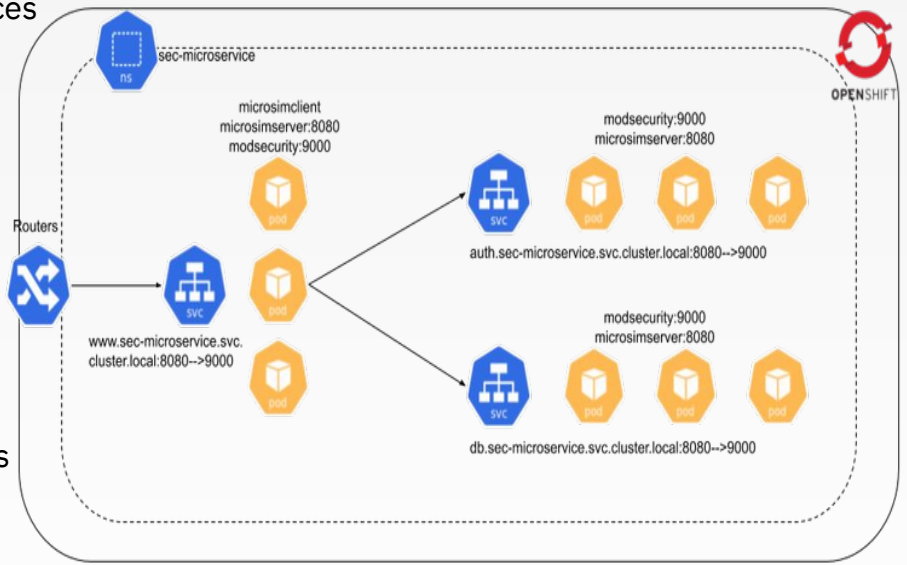
# Sidecar Pattern

Pros:

- Unifies the scaling of the security and application microservices
- The security proxy can be automatically injected into the Pod
- Works with an existing Service Mesh using the Sidecar on Sidecar pattern
- Requires fewer micro-segmentation rules

Cons:

- Requires the Security container and Application container to run on different TCP ports within the Pod
- May result in over-provisioning of the security layer resources




Author: Nho Luong
Skill: DevOps Engineer Lead

# Thank You

Author: Nho Luong
Skill: DevOps Engineer Lead