Securing kubernetes

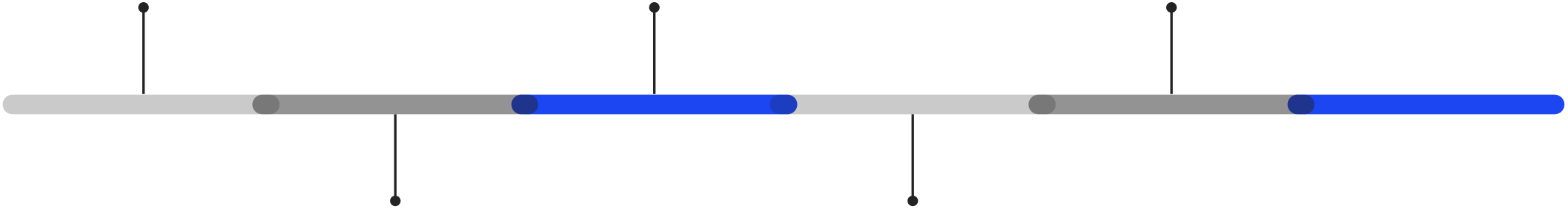
Practical Workflows and Tools for Enhanced Cluster Protection

Agenda

Security Challenges in Kubernetes

Proactive Vulnerability Mitigation

Continuous Security in Kubernetes



Cluster Defense Strategies

Resilient Supply Chain

00

Environment Setup

01

Security Challenges and Risks in Kubernetes

Security Challenges

Rapid deployments

Kubernetes' fast deployments can lead to overlooked security vulnerabilities, challenging traditional security practices.

CI/CD Pipeline Integration

Proactive security integration in CI/CD pipelines, using image scanning and policy enforcement, is essential for vulnerability prevention in production.

Complexity

Managing the security of Kubernetes components such as containers, pods, and deployments can be complicated, especially at scale.

Compliance

Meeting compliance standards in Kubernetes can be complex, requiring careful execution for segmentation, encryption, and audit trails.

The Open Web Application Security Project

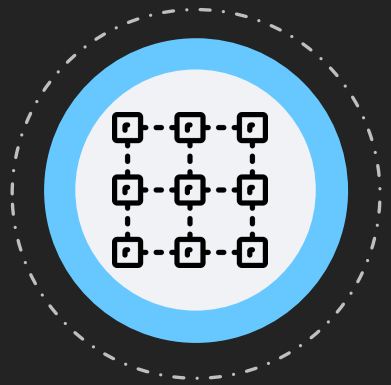
01

It's a non-profit organization that provides free and openly available resources to improve software security. Anyone can participate in their projects and contribute to their vast knowledge base.

02

It provides a framework of best practices for securing applications. These guidelines are particularly important for containerized environments used in the Kubernetes (k8s) ecosystem due to the inherent security risks associated with containers.

Large attack surface



Weak Access Controls

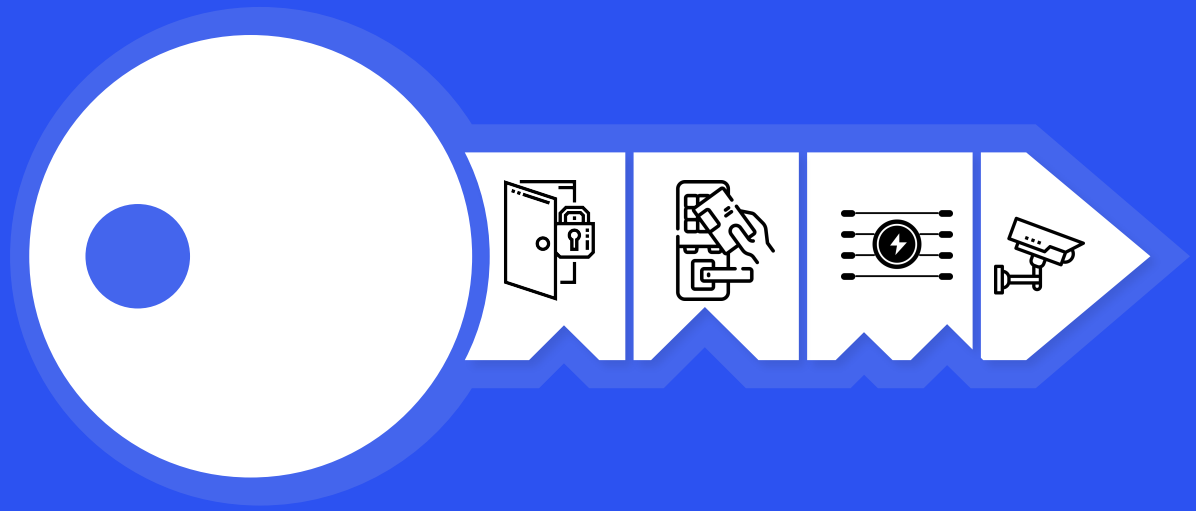
Insider Threats

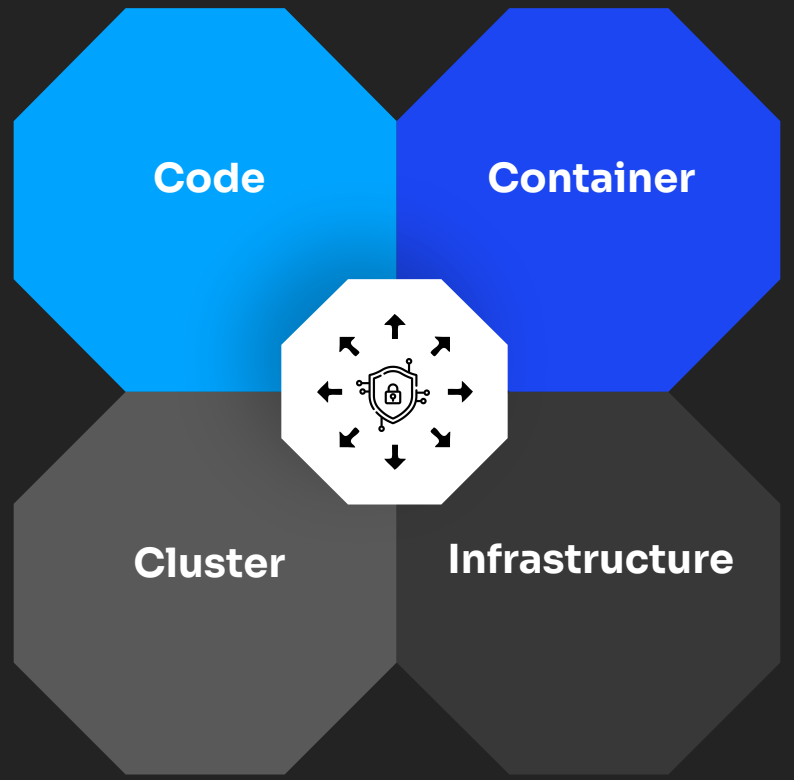
Secrets Management

Vulnerable Container Images

Disclaimer

-No Single Solution-





02

Cluster Defense Strategies

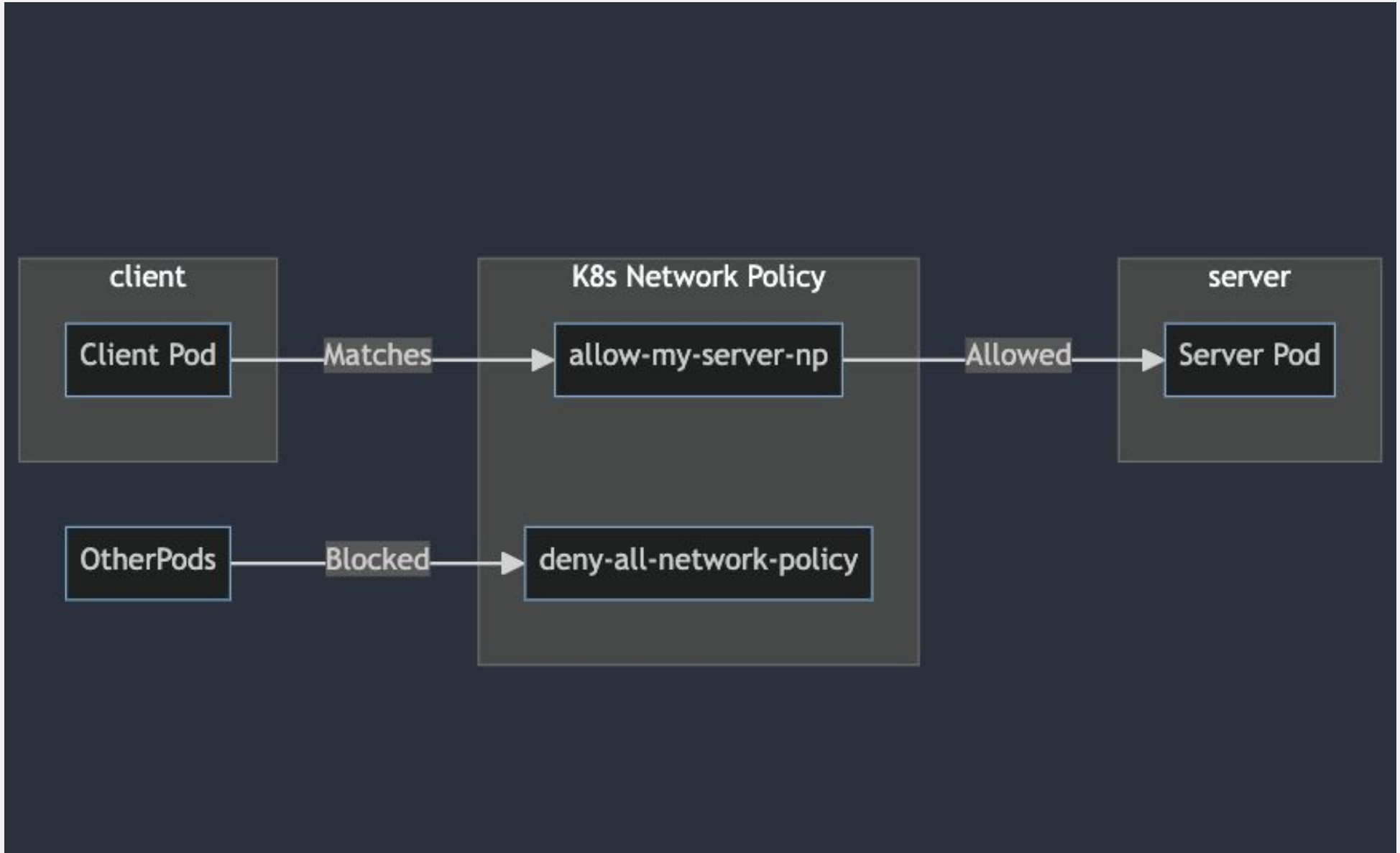
01 Network Segmentation

Segmentation is your key tool to enforce a zero-trust approach within your cluster. It limits the blast radius should a pod get compromised by preventing attackers from easily moving throughout your network.

- **Enhanced Security**
- **Compliance**
- **Isolation**
- **Performance and Traffic Management**

Network Policies

The native Kubernetes way. Network policies let you define fine-grained rules based on



Container Network Interface (CNI)

Many third-party networking solutions offer advanced features:

- **Calico:** Popular for its rich network policy capabilities, fine-grained controls, and visualization options.
- **Cilium:** Emphasizes eBPF for high-performance networking and policy enforcement. Often used in large-scale clusters.
- **Flannel:** A simpler CNI option for basic overlay networking. Provides less granular segmentation than some others.

Hand On

Network Policies

Consideration

Choosing the Right Approach

- ➔ **Complexity:** Starting with Kubernetes network policies is usually recommended. If your needs are simple, this might be enough.
- ➔ **Features:** If you require advanced features like encryption, layer 7 policies (based on HTTP etc.), or complex visualization, a CNI plugin might be a better fit.
- ➔ **Performance:** Some CNI plugins use eBPF, providing the potential for better performance in heavily loaded clusters.

Important Considerations

- ➔ **Planning:** Before you start creating rules, carefully map out your desired segments and traffic flows.
- ➔ **Pod Labeling:** Consistent labeling is essential for effective policy enforcement.
- ➔ **Gradual Rollout:** Start with simple policies and gradually add more complexity. Thoroughly test policies before deploying broadly to minimize disruptions.
- ➔ **Tooling:** Consider tools like Calico's visualization or network policy editors to streamline management, especially in complex environments.

02 TLS Communication

TLS termination refers to the process of decrypting incoming HTTPS traffic at a designated point before forwarding it to your backend application pods. This approach enhances security by ensuring that only the component responsible for termination (usually an Ingress controller) handles the sensitive encryption keys.

Hand On

TLS Termination

03 RBAC: Secure Access Control

RBAC (Role-Based Access Control) is the standard mechanism for managing permissions within Kubernetes

Pitfalls

Unnecessary Cluster-Admin Usage

Neglecting Service Accounts

Overly Permissive Roles and RoleBindings

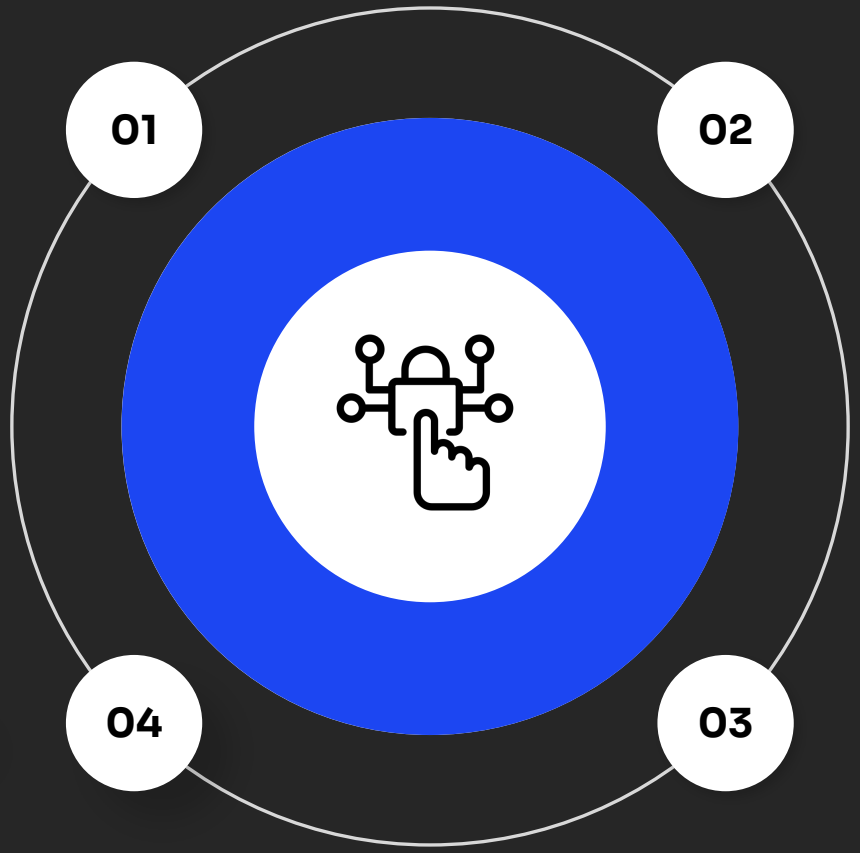
Role Aggregation Misuse

Inadequate Auditing and Maintenance

RBAC Best Practices

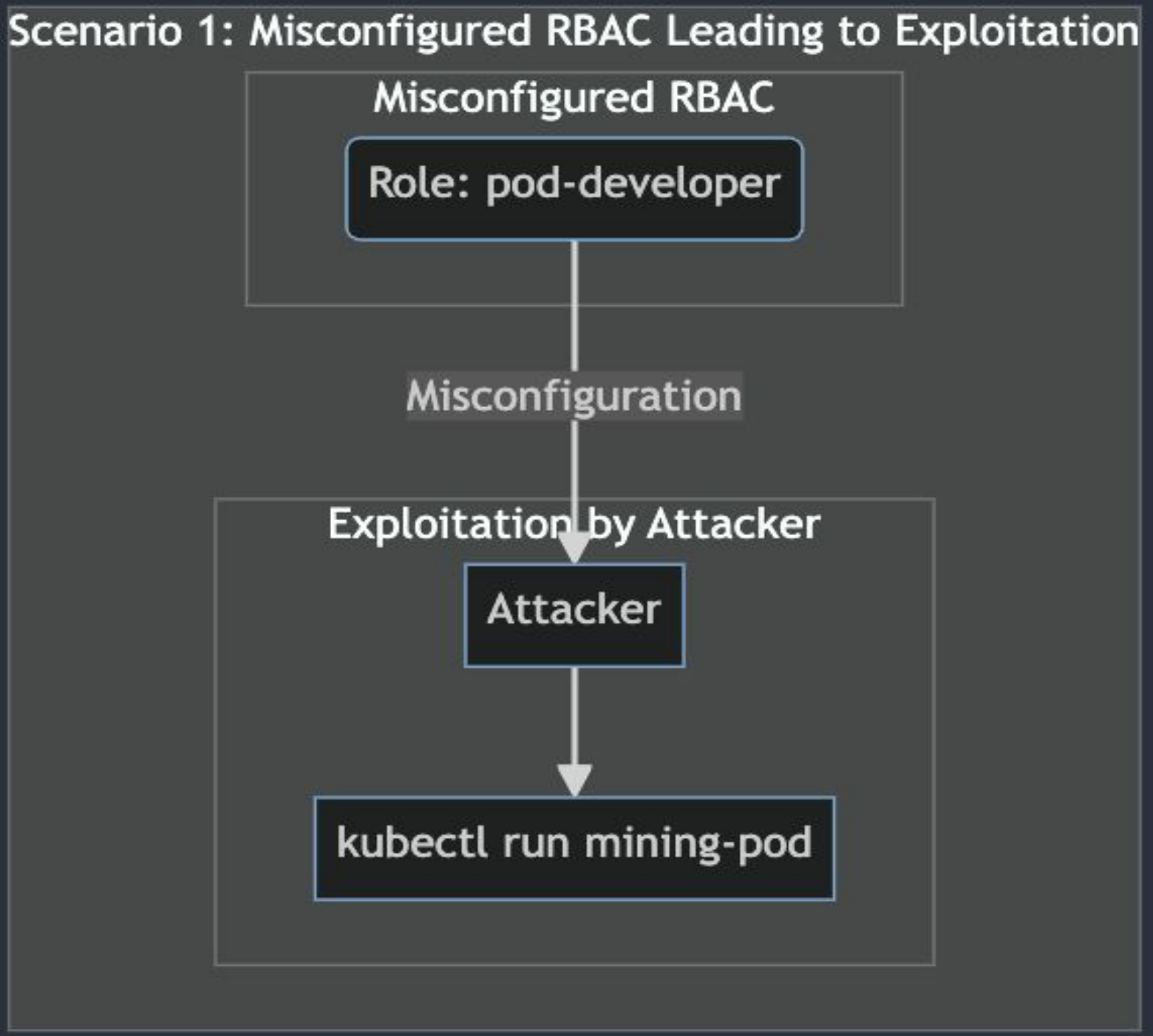
01 Least Privilege Principle

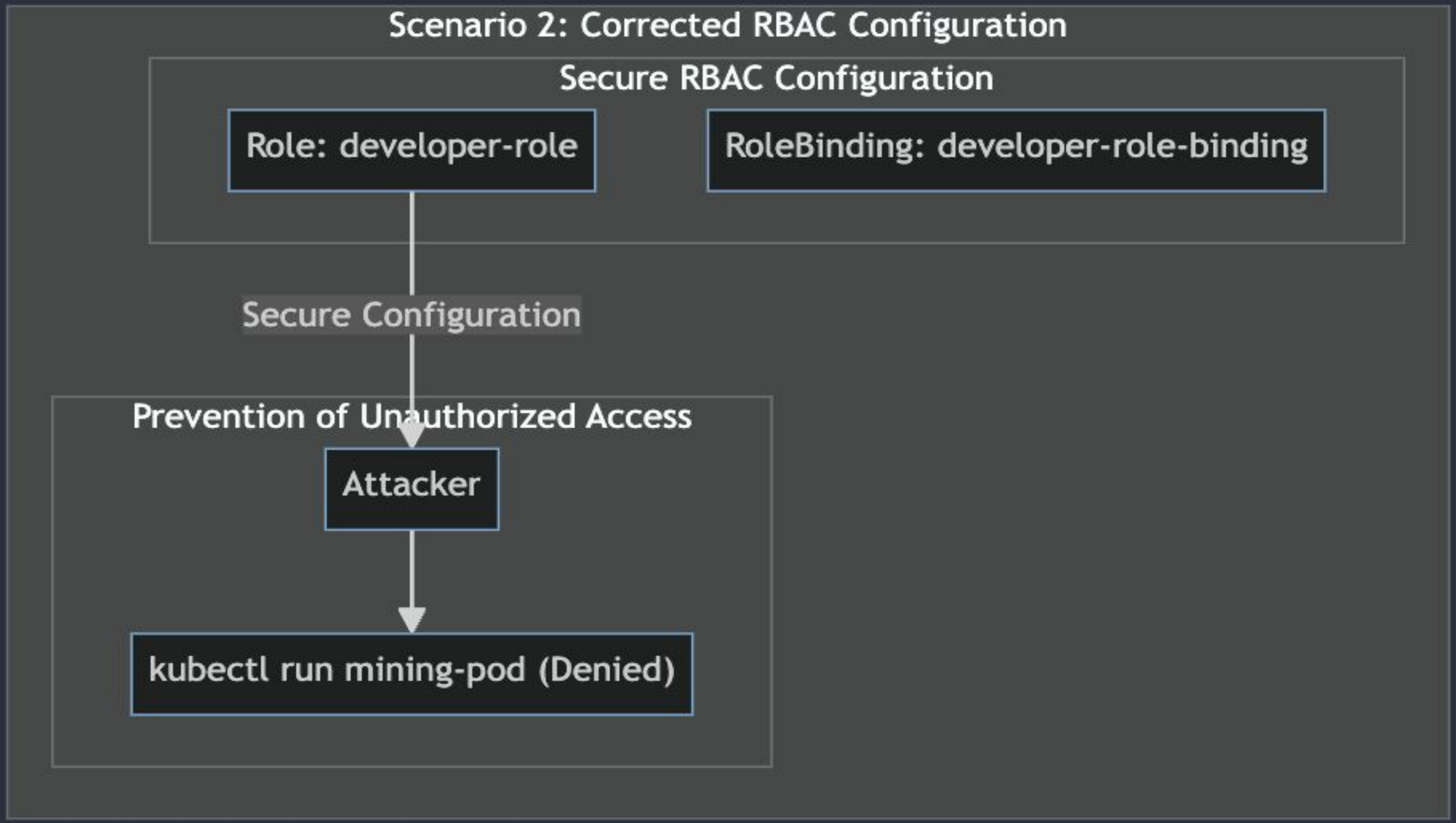
02 Namespace level RoleBindings



04 Regular Audits And Reviews

03 Custom Roles


Scenario 1: Misconfigured RBAC Leading to Exploitation
Misconfigured RBAC
Role: pod-developer
Misconfiguration
Exploitation by Attacker
Attacker
kubectl run mining-pod


Scenario 2: Corrected RBAC Configuration
Secure RBAC Configuration
Role: developer-role
RoleBinding: developer-role-binding
Secure Configuration
Prevention of Unauthorized Access
Attacker
kubectl run mining-pod (Denied)

Hand On

RBAC

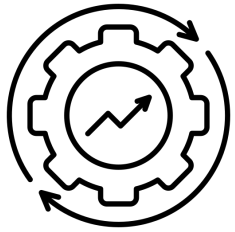
03

Proactive Vulnerability Mitigation

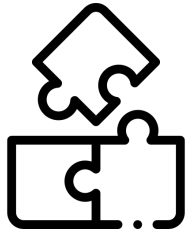
Secret Management

Kubernetes Secrets are objects designed to store sensitive information in a more secure manner than storing the data directly in Pod definitions or container images

Enhanced Security



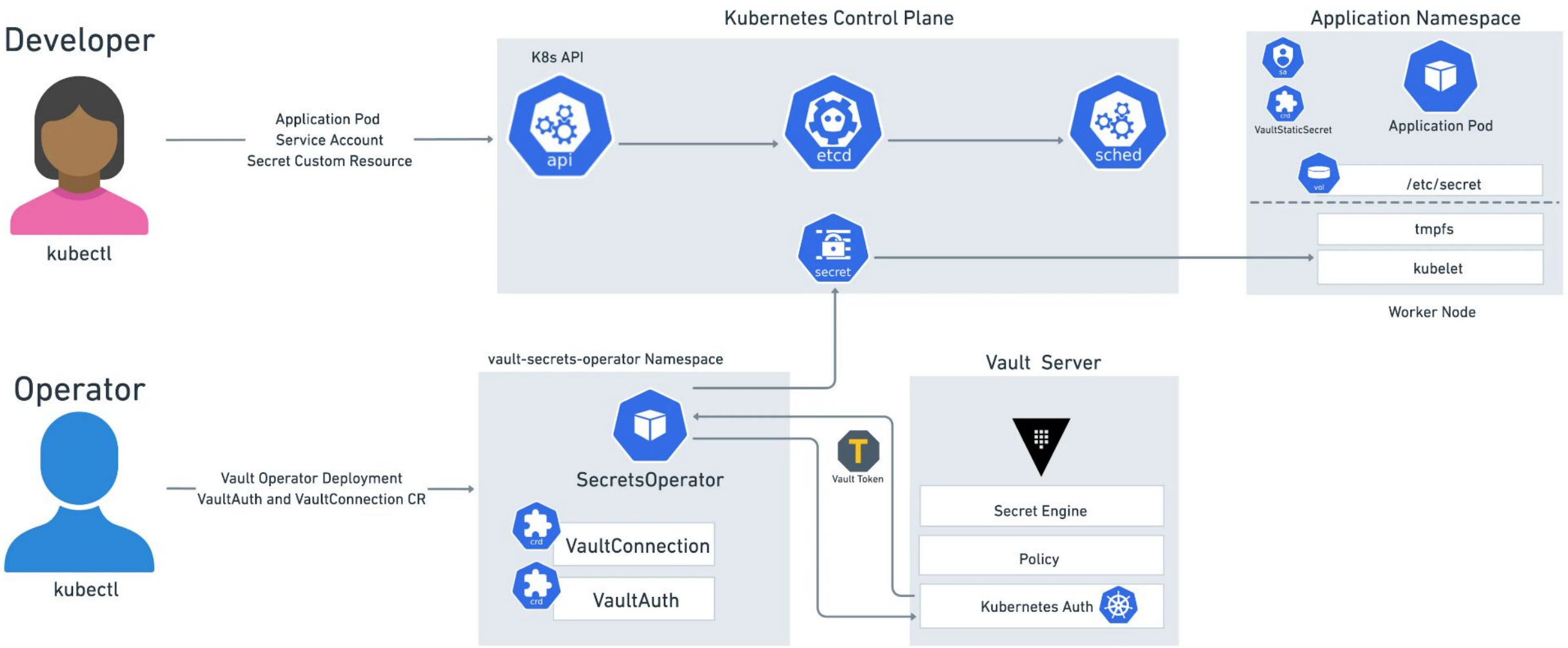
Improved Configuration Management



Flexibility

Auditability

Enhanced Secret Management



Hand On

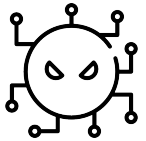
Secret Management

? The Problems

Danger of Untrusted Code

Multi-Tenant Environments

Resource Havoc



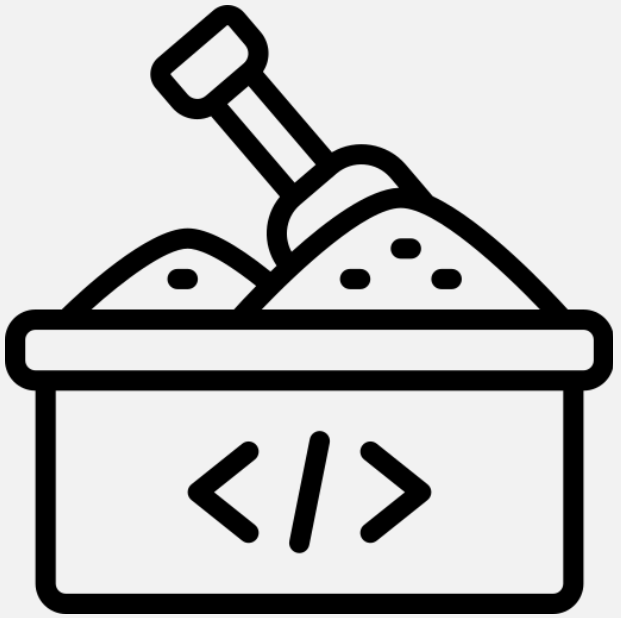
Malware Protection

Container Sandboxing

Sandbox creates an isolated environment within a Kubernetes cluster where you can run potentially untrusted or experimental code without risking the stability or security of other applications in the cluster. Think of it as a 'playground' within your Kubernetes environment.

gVisor

- User-space application kernel: Provides a secure environment within a container.
- Intercepts system calls: Redirects them away from the host kernel, reducing vulnerability.
- Written in Go: Offers memory safety advantages.



Hand On

gVisor Sandbox

04

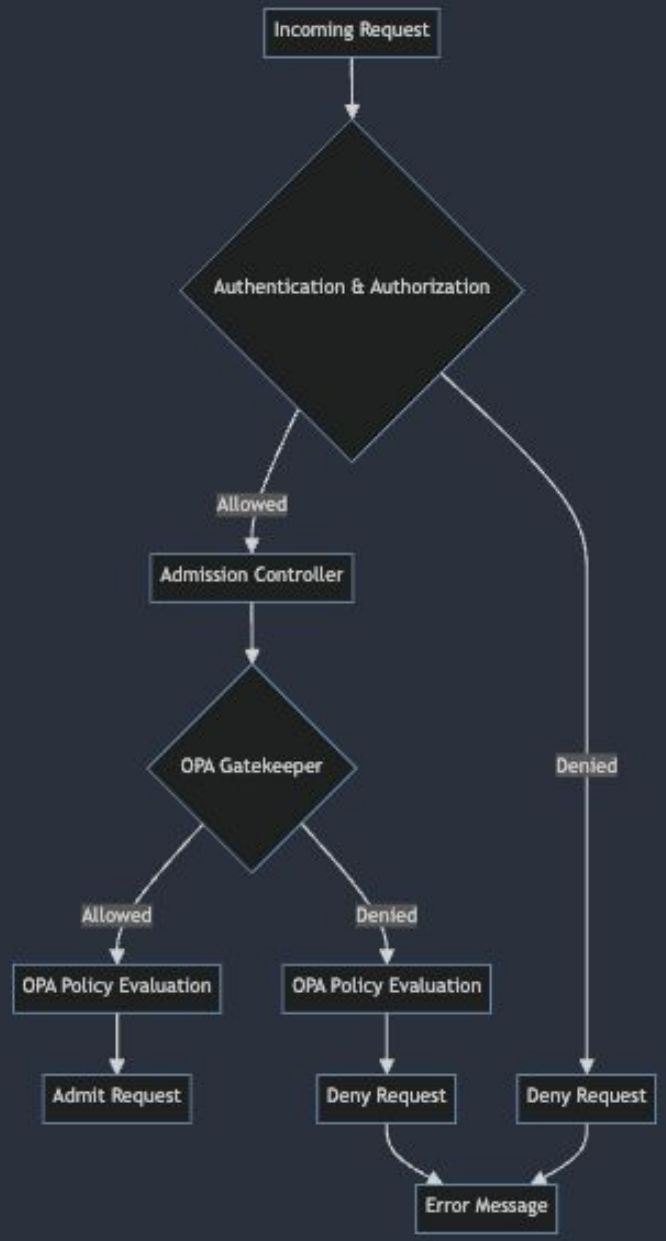
Resilient Supply Chain

OPA GateKeeper

Admission Controller for Kubernetes

Policy Enforcement with OPA

Customizable Security and Governance



Falco – Runtime Security

Runtime Security Engine

Designed for Kubernetes. It keeps a watchful eye on your cluster's behavior by monitoring system calls, network activity, file access, and more

Behavioral Analysis

Detects anomalous or suspicious behavior in your containers, applications, and the Kubernetes cluster itself.

Alerting & Monitoring

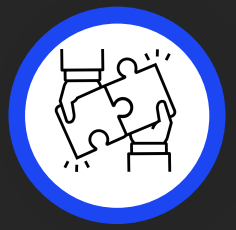
Once a rule is triggered, Falco generates an alert, providing you with details about the suspicious activity

Trivy Vulnerability Scanner

Vulnerability Detection

Database Updates

Fast Scanning



Easy Integration

Hand On

Supply Chain

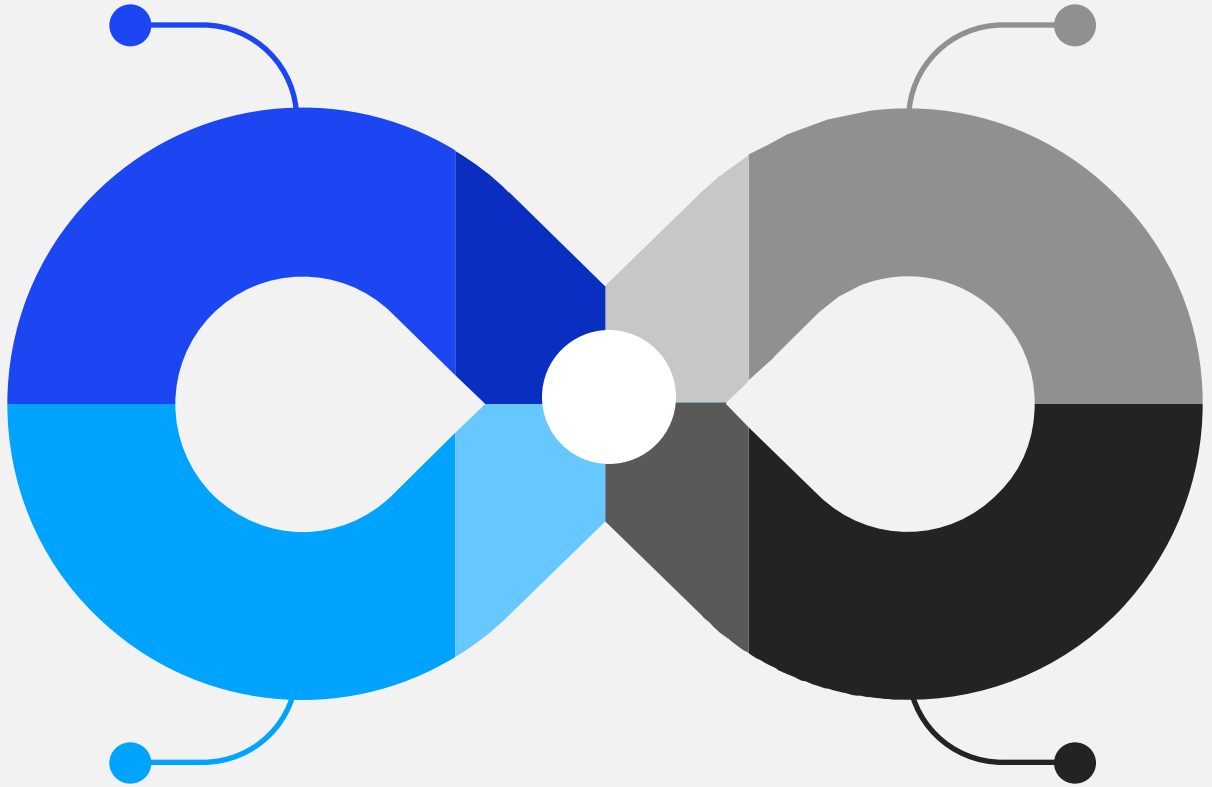
05

Continuous Security in Kubernetes

Continuous Security

Automated Security Testing

Real-time Threat Monitoring and Response



Security Training and Awareness

Continuous Compliance and Governance

Questions?

Stay Secured &

Keep in Touch

wave