

ELEVATE SECURITY AND OBSERVABILITY WITH CILIUM



ELEVATE SECURITY AND OBSERVABILITY WITH CILIUM

Agenda

01. eBPF
02. Cilium
03. Setting Up the Environment
04. Cilium Observability
05. Cilium Security





→ **Fabrizio Sgura**
Chief Engineer

Enjoying life in all its aspects



→ **Gerardo López**
Principal Engineer

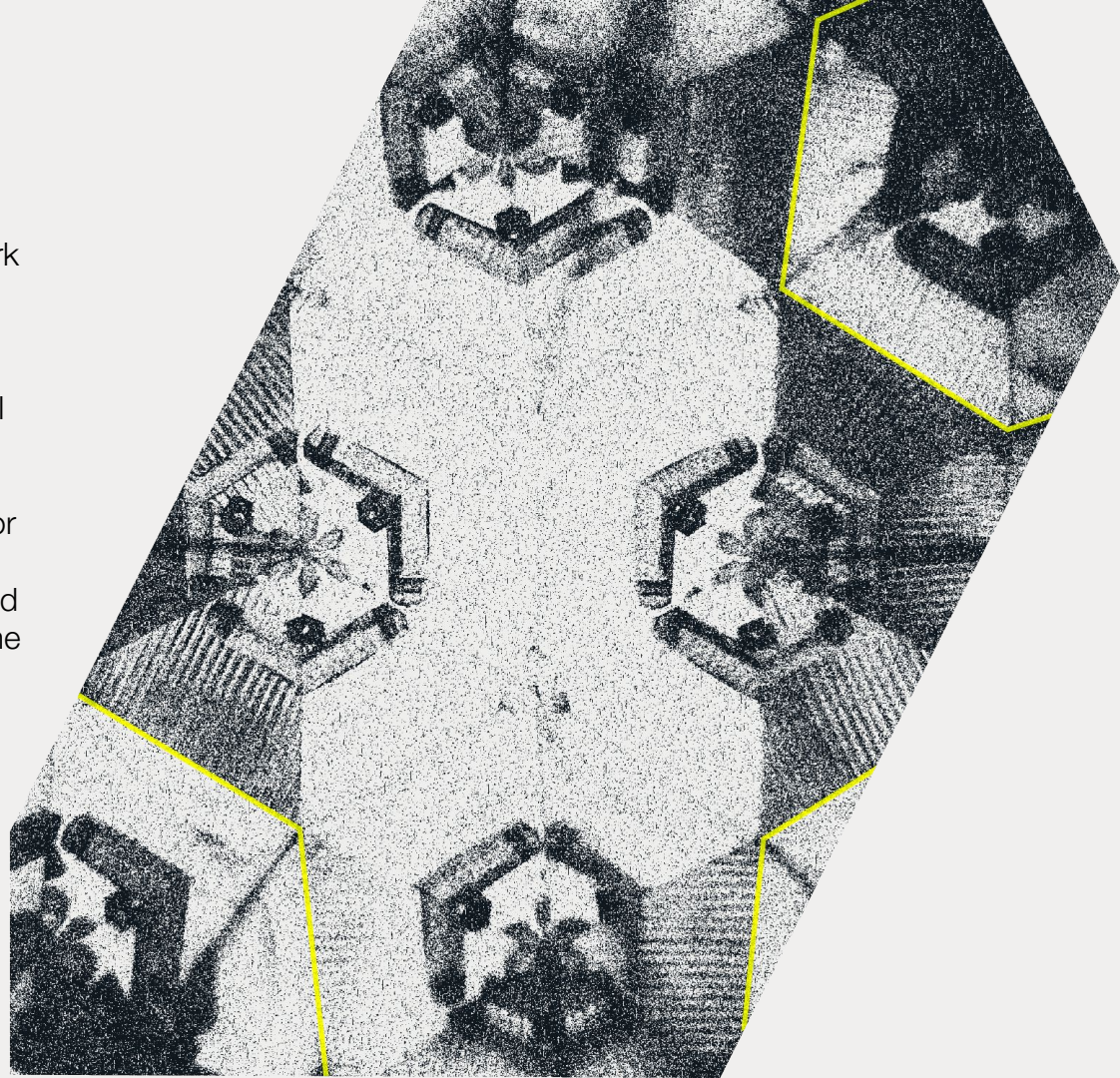
Soccer and Movie Enthusiast; BBQ lover

CONTEXT

In today's landscape, focusing on security and network observability in Kubernetes is paramount due to the exponential growth in container and microservices adoption in production environments. The inherent complexity of distributed architectures makes it crucial to safeguard critical assets and ensure visibility and control over communications between components. Enhanced security and improved observability allow for identifying and mitigating potential vulnerabilities, as well as proactively detecting and resolving network and performance issues. This contributes to maintaining the integrity, reliability, and availability of applications deployed on Kubernetes.

ELEVATE SECURITY AND OBSERVABILITY WITH CILIUM

Veritas Automata.





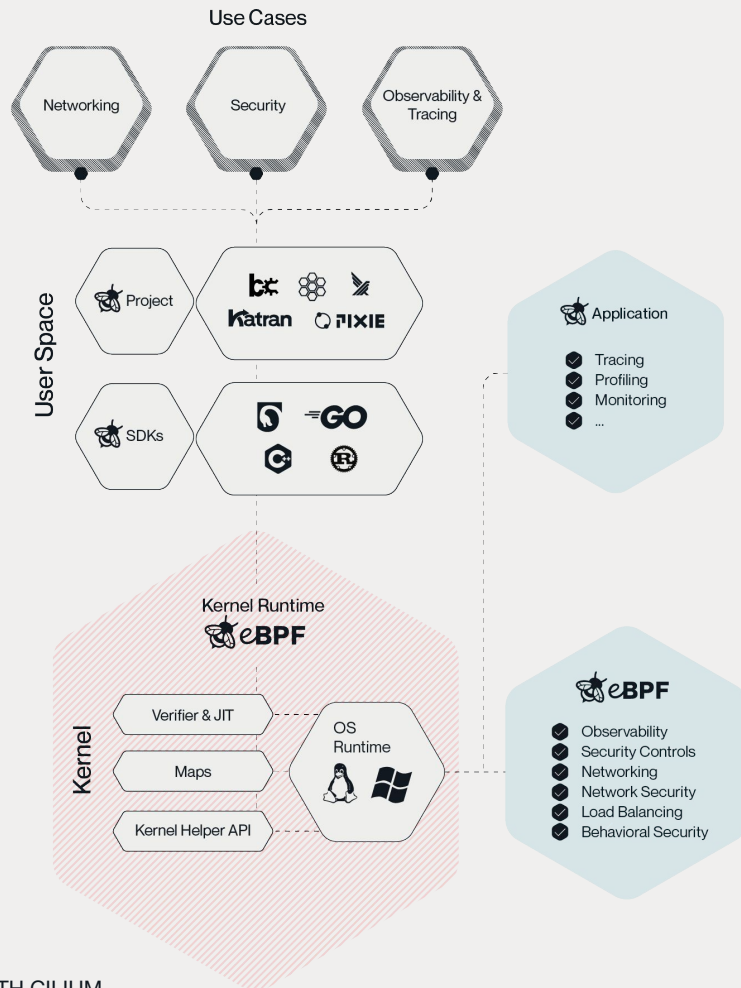
WHAT eBPF IS?

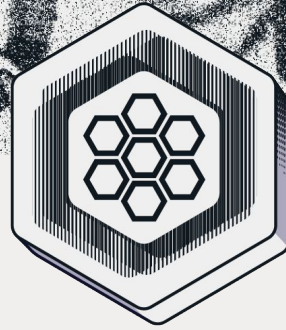
eBPF, or extended Berkeley Packet Filter, is a powerful and flexible framework within the Linux kernel that allows for efficient and safe execution of custom code snippets within the kernel space. Originally designed for network packet filtering, eBPF has evolved into a versatile tool for various tasks such as tracing, monitoring, and security enforcement.

Overall, eBPF is a versatile technology that empowers developers and administrators to extend and enhance the capabilities of the Linux kernel in a safe and efficient manner, opening up new possibilities for performance optimization, observability, and security enforcement.

eBPF

eBPF is a revolutionary technology with origins in the Linux kernel that can run sandboxed programs in a privileged context such as the operating system kernel. It is used to safely and efficiently extend the capabilities of the kernel without requiring to change kernel source code or load kernel modules.

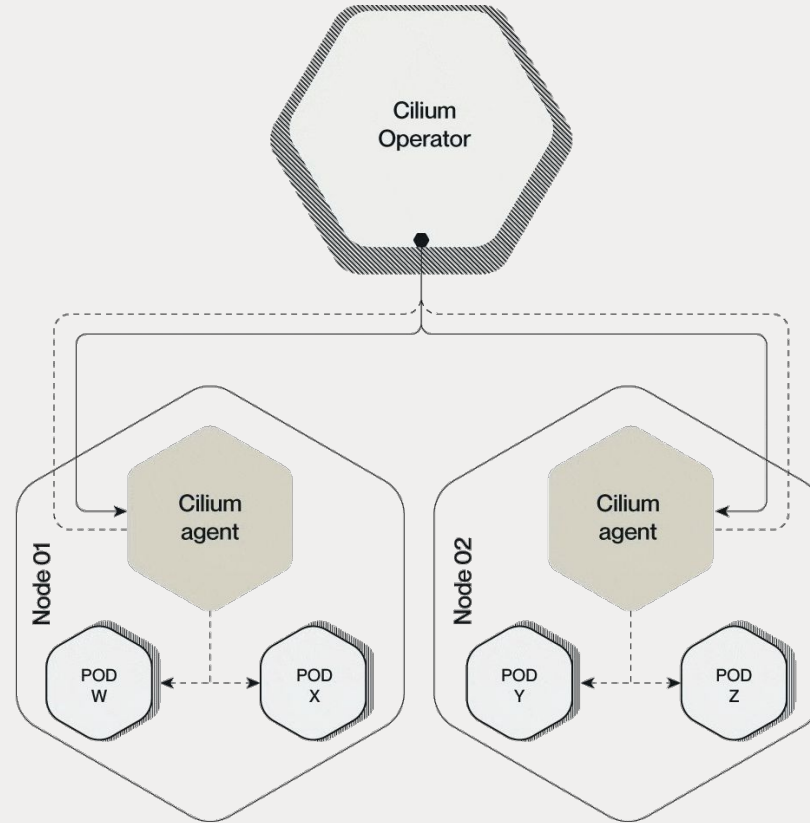




CILIUM

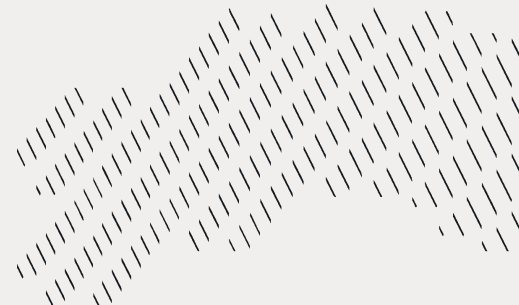
Cilium is a powerful networking and security solution for Kubernetes environments. It offers advanced features such as Layer 7 visibility, security policy enforcement, and network load balancing. By leveraging eBPF technology, Cilium provides efficient packet processing and scalability while ensuring robust network security and observability. It enhances Kubernetes networking capabilities, enabling seamless communication between microservices while enforcing security policies at scale.

CILIUM HIGH LEVEL COMPONENTS

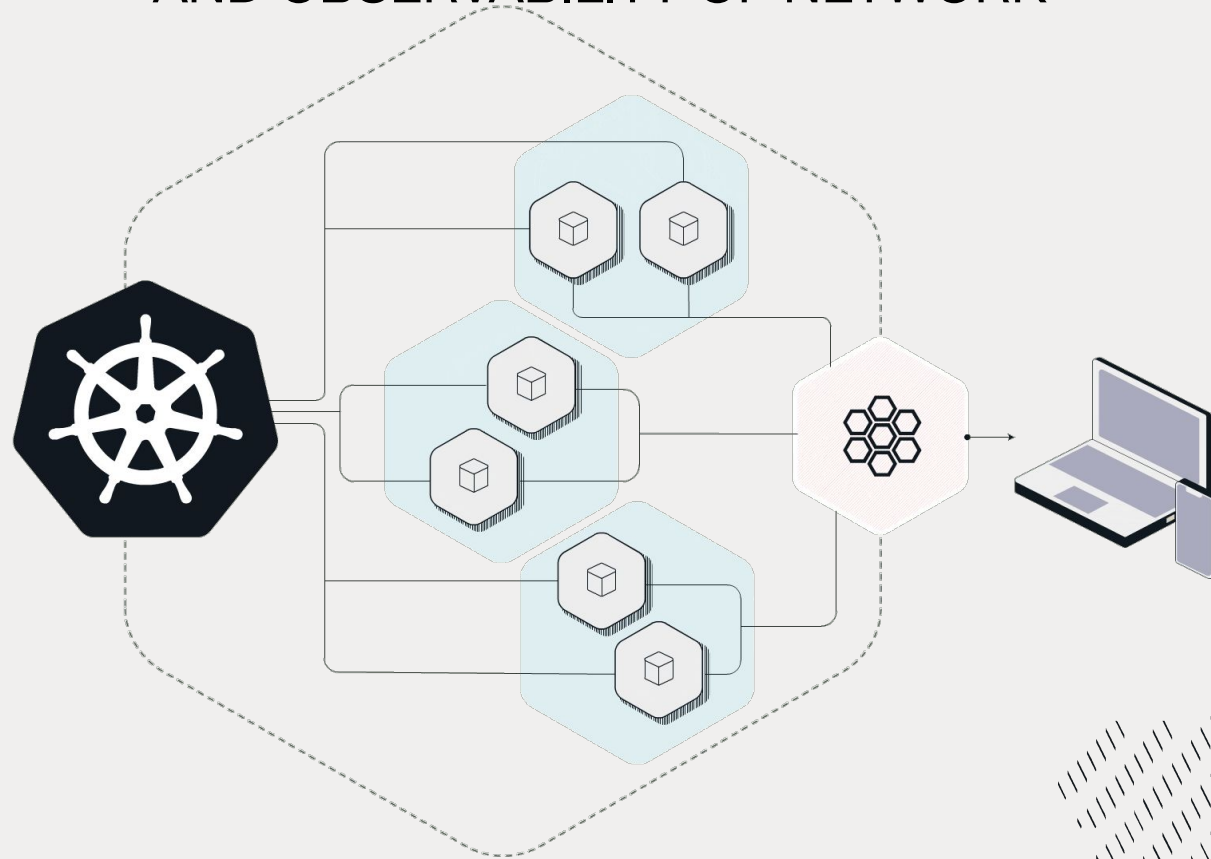


Cilium enhances observability in Kubernetes environments by providing comprehensive insights into network traffic, security, and application behavior, thereby enabling operators to efficiently monitor, troubleshoot, and optimize their deployments.

- **Network visibility**
- **Service discovery**
- **Rich metrics**
- **Layer 7 visibility**
- **Flow visualization**



CILIUM ENFORCES THE SECURITY AND OBSERVABILITY OF NETWORK



SETTING UP THE ENVIRONMENT

CILIUM OBSERVABILITY

ELEVATE SECURITY AND OBSERVABILITY WITH CILIUM

CILIUM SECURITY

ELEVATE SECURITY AND OBSERVABILITY WITH CILIUM

CILIUM SECURITY

Network Policy Editor by ISOVALENT

The diagram illustrates the Network Policy Editor interface. It features a central 'In Namespace' selector box. On the left, three source selector boxes are connected to the central box: 'Outside Cluster' (Any endpoint), 'In Namespace' (Any pod, end: front, ipBlock: 10.144.0.0/16), and 'In Cluster' (Everything in the cluster). On the right, three destination selector boxes are connected to the central box: 'Outside Cluster' (Any endpoint), 'In Namespace' (Any pod), and 'In Cluster' (Everything in the cluster, Kubernetes DNS).

Feedback/Questions?
[Ask on Slack](#)

```
1 apiVersion: networking.k8s.io/v1
2 kind: NetworkPolicy
3 metadata:
4   name: demo-policy
5 spec:
6   podSelector: {}
7   ingress:
8     - from:
9       - ipBlock:
10         cidr: 0.0.0.0/0
11     - from:
12       - podSelector:
13         matchLabels:
14           end: front
15   ports:
```

Welcome to the Network Policy Editor!

This tutorial will teach you how to create a network policy using the Editor. It explains basic network policy concepts and guides you through the steps needed to achieve the desired least-privilege security and zero-trust concepts.

Step 1. What pods do you want to secure?

The diagram shows a Kubernetes cluster represented as a house. Inside the cluster, there are two namespaces: namespaceA and namespaceB. namespaceA contains three pods, with the middle one highlighted in red. namespaceB contains six pods arranged in a 2x3 grid.

CILIUM SECURITY

Network Policy Editor by ISOVALENT

Feedback/Questions?
Ask on Slack

```
Kubernetes Network Policy Cilium Network Policy
```

Policy Rating [Progress] Download Share

```
1 apiVersion: networking.k8s.io/v1
2 kind: NetworkPolicy
3 metadata:
4   name: demo-policy-ns
5   namespace: http-test
6 spec:
7   podSelector: {}
8   ingress:
9     - from:
10       - ipBlock:
11         cidr: 0.0.0.0/0
12       - from:
13         - podSelector:
14           matchLabels:
15             end: front
```

Main tutorial Flows upload

Welcome to the Network Policy Editor!

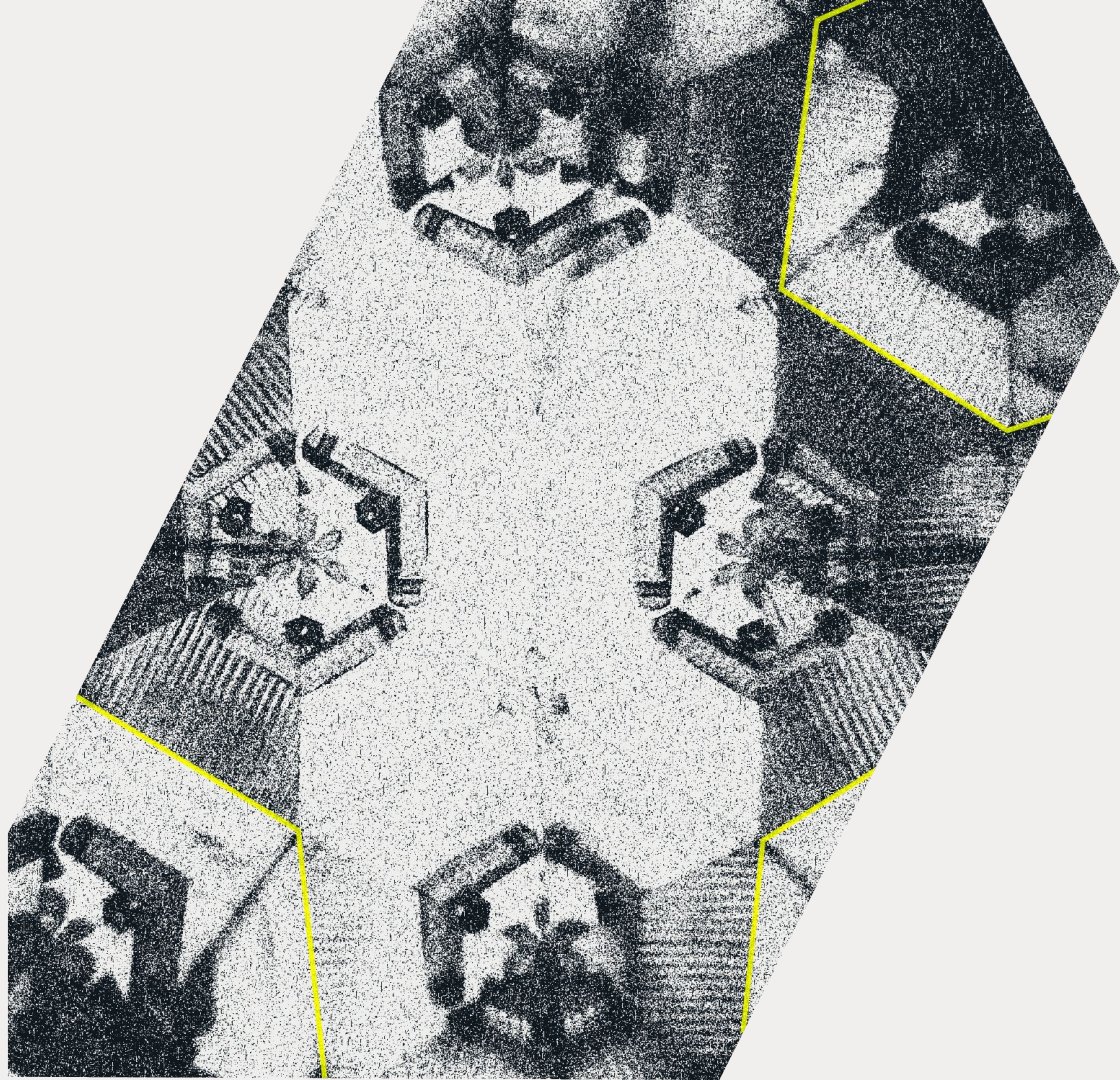
This tutorial will teach you how to create a network policy using the Editor. It explains basic network policy concepts and guides you through the steps needed to achieve the desired least-privilege security and zero-trust concepts.

Step 1. What pods do you want to secure?

ELEVATE SECURITY
AND OBSERVABILITY
WITH CILIUM

QA

Veritas Automata.



ELEVATE SECURITY
AND OBSERVABILITY
WITH CILIUM

THANK YOU



Fabrizio Sgura
Chief Engineer



Gerardo "Falcon" López
Principal Engineer

