

# CNCF Project Focus

## Episode #3



# cilium

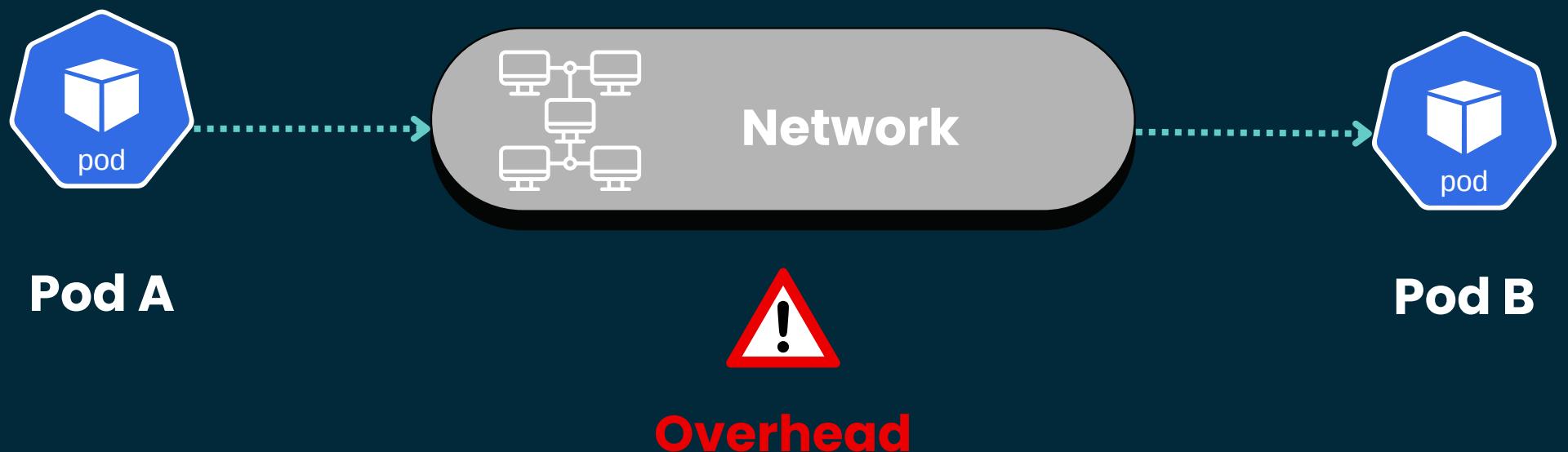
## Kernel-Native Kubernetes Networking



Christian Dussol

# The hidden network overhead

*Every Pod-to-Pod call in your Kubernetes cluster*



+ Higher Latency



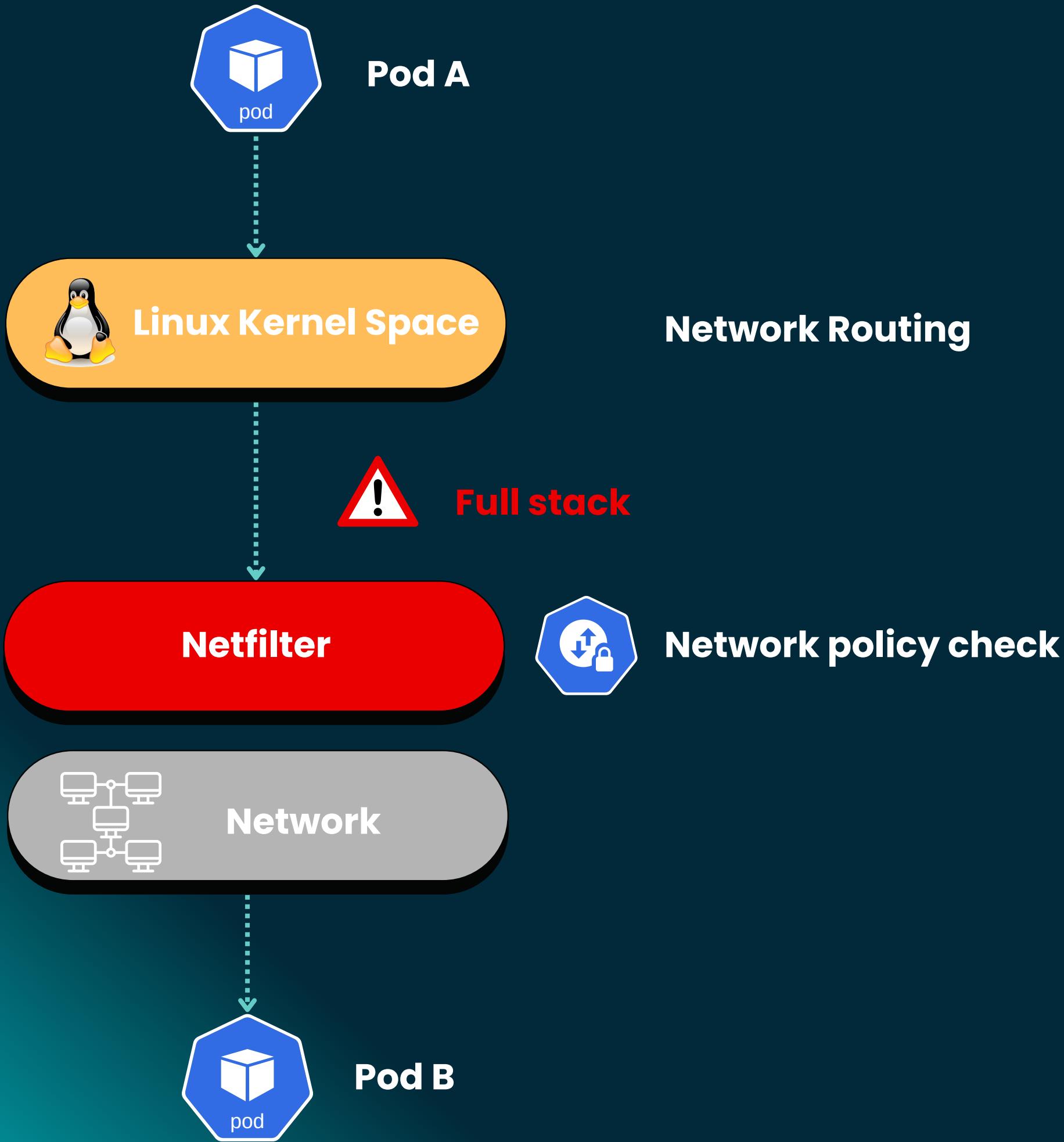
Slower responses



Higher Cost

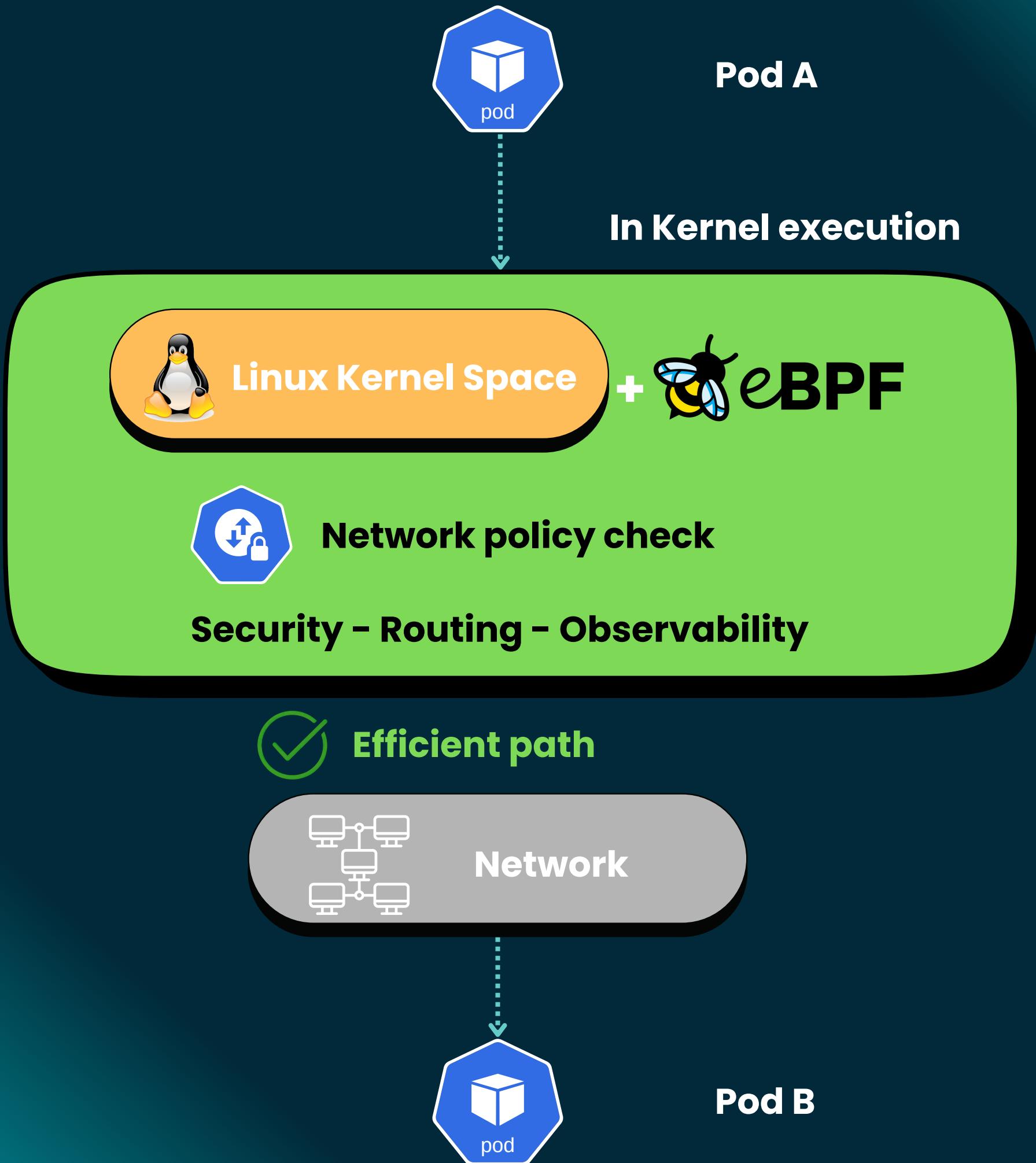
# The Netfilter overhead

## *Pod-to-Pod communication*

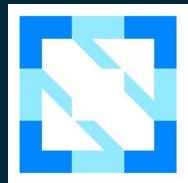


# Solution : optimized Kernel path

*Network policy check happens in Kernel with eBPF*



# What is Cilium ?



**CNCF Graduated project (October 2023)**

Core Idea

**eBPF-based networking, observability and security**

What is



**Extended Berkeley Packet Filter**

- Run code in Linux kernel
- Safe, fast, no kernel mods
- Network, security, observability

Adopted by

Google



Microsoft

NETFLIX

**10+ years of innovation (2014+)**

# USE CASE 1 : Performance

*Kernel-native networking with optimized packet processing*



- **Latency reduction vs traditional CNI**
- **CPU reduction for network policies**
- **XDP (eXpress Data Path) for 10M+ pps**
- **Efficient connection tracking (conntrack)**

## USE CASE 2 : security

### *Identity-based security with L3-L7 enforcement*



- **Identity-based policies (not IP-based)**
  - Services identified by Kubernetes labels
  - IP changes don't break policies
  
- **API-aware security (L7 filtering)**
  - Allow: POST /api/v1/payment
  - Deny: GET /api/v1/admin
  
- **Transparent encryption**
  - WireGuard or IPsec

# Identity-based security

---

## IP-Based policies

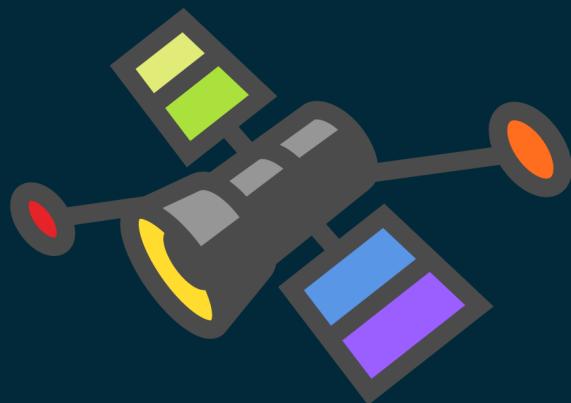
- **Fragile (Breaks on pod restart)**
- **Complex (Manage 1000s of IPs)**
- **Insecure (IPs can be spoofed)**

## Identity-Based Policies

- **Resilient (Survives pod changes)**
- **Simple (Label-based rules)**
- **Secure (Cryptographic identity)**

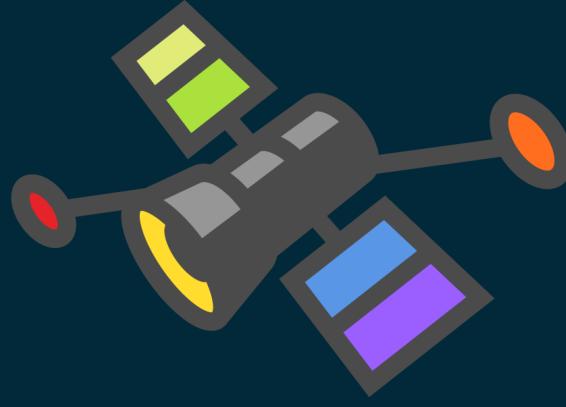
# USE CASE 3 : Observability

*Deep network visibility with Hubble (eBPF-powered)*



## Hubble

- **Real-time network flow visualization**
- **Service dependency mapping**
- **DNS query monitoring**
- **Metrics: latency, throughput, errors**
- **Zero instrumentation overhead**



# Hubble

payment-system ▾ Filter by: label key=val, ip=1.1.1.1, dns=google.com, identity=42, pod=frontend Any verdict Visual ● 22.0 flows/s • 1/1 nodes

The diagram illustrates a network topology within a 'payment-system' scope. A 'test-client' (represented by a blue icon) is shown on the left, connected to several services. One connection leads to 'world' (blue globe icon), which has a flow to port 80 TCP. Another connection leads to 'google.com' (blue globe icon), also with a flow to port 80 TCP. Two other connections lead to 'api-gateway' (blue gear icon), both with flows to port 80 TCP. Finally, a connection leads to 'payment-service' (blue gear icon), with a flow to port 8080 TCP. A separate connection from the 'test-client' leads to 'payment-db' (blue ship icon), with a flow to port 5432 TCP. The connections are represented by red lines, indicating active flows.

payment-system

Columns ▾

Source Identity	Destination Identity	Destination Port	L7 info	Verdict	Timestamp
test-client payment-system	payment-db payment-system	5432	—	dropped	2026/01/19 00:39:03 (+01)
test-client payment-system	payment-db payment-system	5432	—	dropped	2026/01/19 00:39:03 (+01)
test-client payment-system	payment-db payment-system	5432	—	dropped	2026/01/19 00:39:02 (+01)
test-client payment-system	payment-db payment-system	5432	—	dropped	2026/01/19 00:39:02 (+01)
test-client payment-system	payment-db payment-system	5432	—	dropped	2026/01/19 00:39:01 (+01)

# Explore my learning toolkit



[github.com/christian-dussol-cloud-native/cilium/](https://github.com/christian-dussol-cloud-native/cilium/)

## Educational GitHub Repository

Complete hands-on tutorial, including:

- Quick start**  
Cluster setup, Cilium installation & verification
- Network policies (L3/L4/L7)
- Hubble observability
- Kyverno governance
- Payment API demo (PCI-DSS compliant)