Adfinis

Plan. Innovatively | **Build.** Sustainably | **Run.** Resiliently

Potential. Unlocked

Using Service Mesh for Resilience and Observability

Photo by Uriel SC on Unsplash

Lukas Grossar

Solution Architect

Adfinis

Bern, Switzerland

lukas.grossar@adfinis.com

linkedin.com/in/tongpu

github.com/tongpu

Lucas Bickel

Software Developer

Adfinis

Bern, Switzerland

lucas.bickel@adfinis.com

github.com/hairmare


Open & Inner Source Mindset
Clients can focus on their core business
Motivated Team
Partnerships
Cloud Native,
Classic IT & Development
24 / 7 / 365 Support
Switzerland,
Netherlands & Australia

Adfinis Partnerships

Red Hat

ISOVALENT

Thanks for having us!

Welcome to Adfinis!

Topics for today

- Resilience and Observability for Networking?
- What is eBPF?
- What is Cilium?
- What is OpenTelemetry?
- Installing Cilium on OpenShift
- Integrating Hubble with OpenTelemetry
- Live demo 🤞
- Wrap up

Resilience and Observability for Networking?

Observability

- collect relevant data (e.g. metrics, traces, and logs)
- send this data to systems that store and analyze it
- visualize the data to provide insights

Service Mesh 101

- dedicated infrastructure layer for facilitating service-to-service communications
- provides:
  - observability into communications
  - security through mutual authentication (mTLS)
  - Resilience by providing features like retries and backoffs
- most common service meshes are a control plane for Envoy
- save on costs because you can solve things centrally

Envoy?

- open source edge and service proxy, designed for cloud-native applications
- API driven and highly cloud-native
- usually bundled into other solutions (like Istio, Cilium, …)
- can take care of OSI layers 3–7 for the mesh

Service Mesh Benefits

As a Developer:

- no need to care about infra since it’s provided by the platform
- secured comms without work

As a Platform Operator:

- one tool to rule all the networks
- set downstream policies for users (devs)

As a Manager:

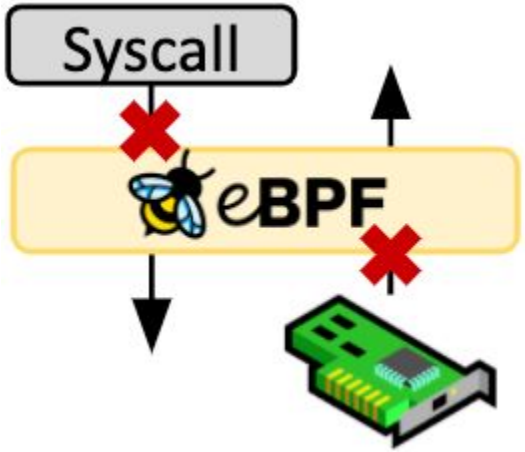
- assurance that the platform fulfills all governance requirements

What is eBPF?

What is eBPF?

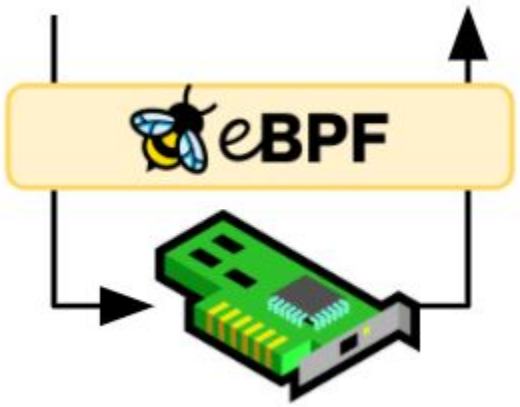
- revolutionary
- what JavaScript was to the browser, eBPF is to the kernel
- next wave of tools covering a wide variety of use cases

Security



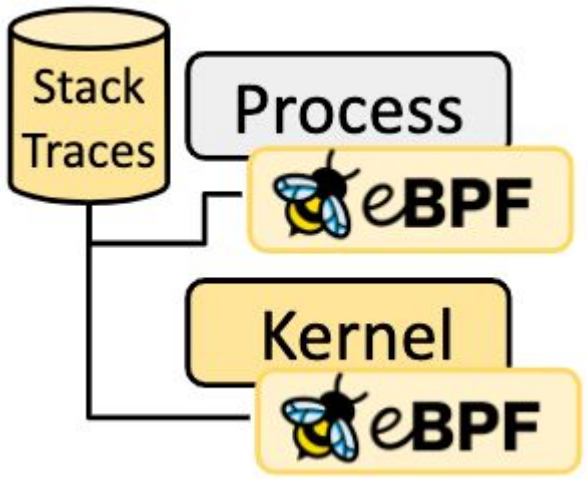
- log, filter, and process all syscalls
- allows security systems to operate with more context and a better level of control

Networking



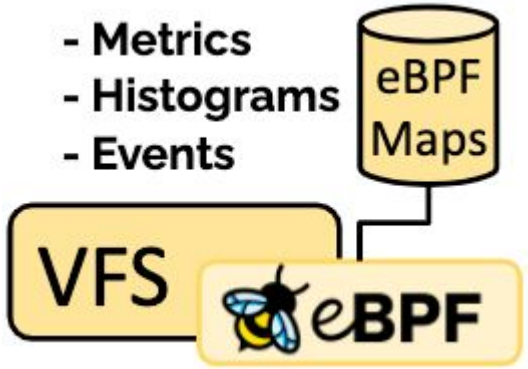
- started as extended Berkeley Packet Filter (eBPF)
- process network packages without them leaving the kernel

Tracing & Profiling



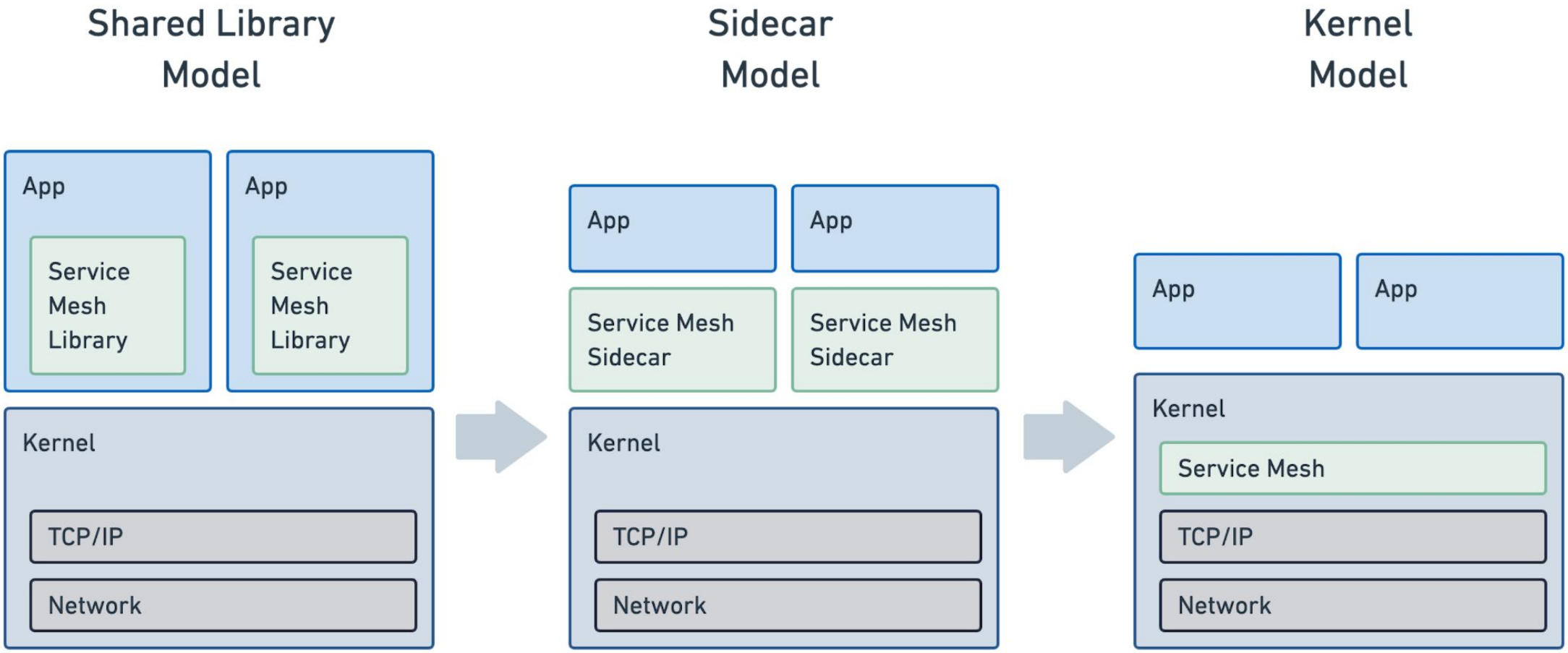
- unprecedented visibility
- runtime behavior of applications and the system
- unique insights to troubleshoot system performance problems

Observability & Monitoring

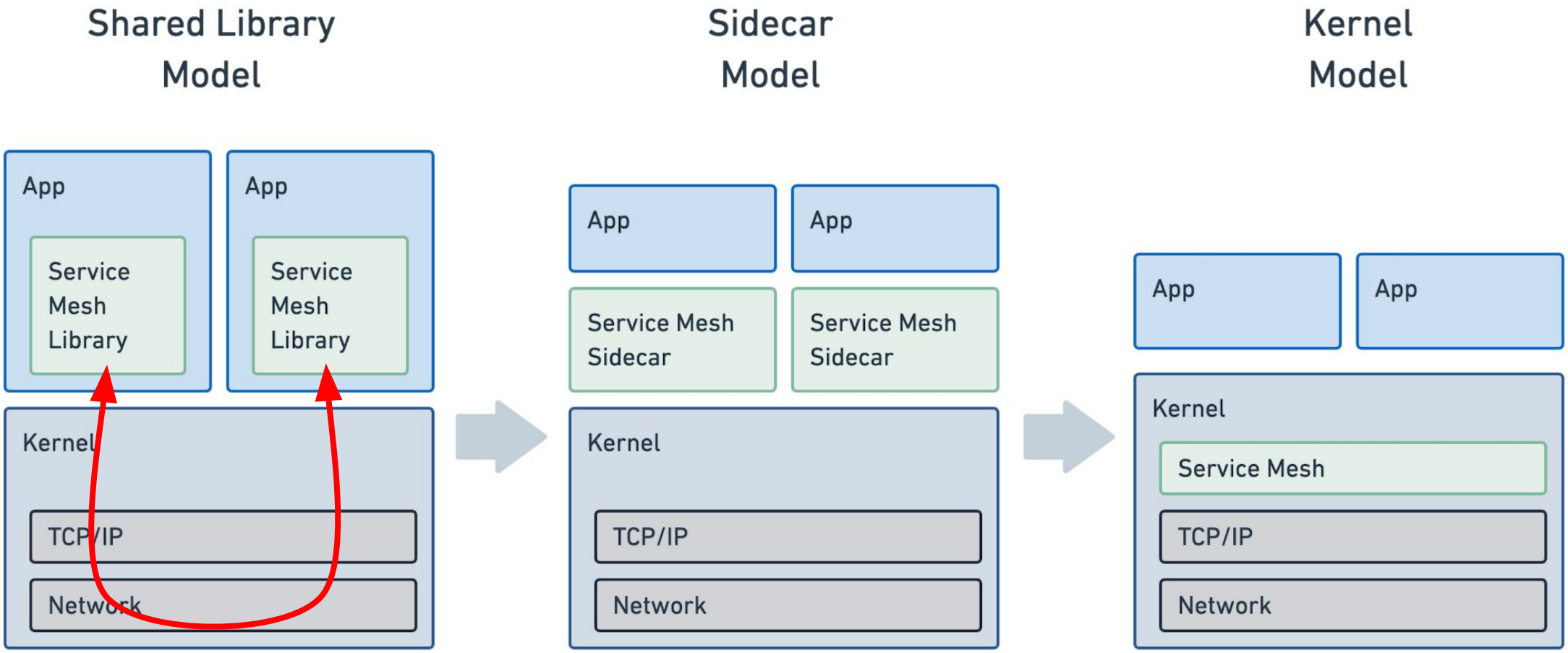


- collection & in-kernel aggregation of custom metrics
- generation of visibility events based on a wide range of sources
- extends the depth of visibility and reduces the overall system overhead

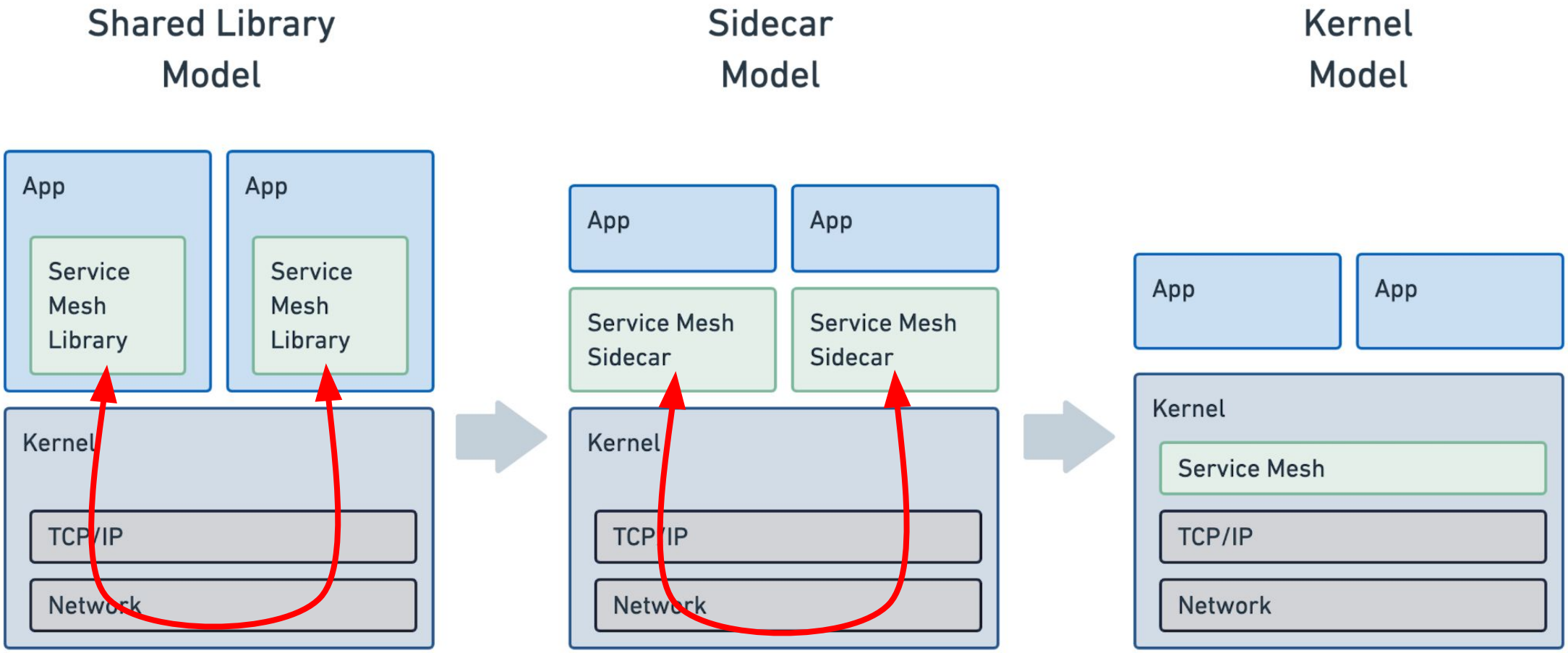
Benefits of eBPFing all the things



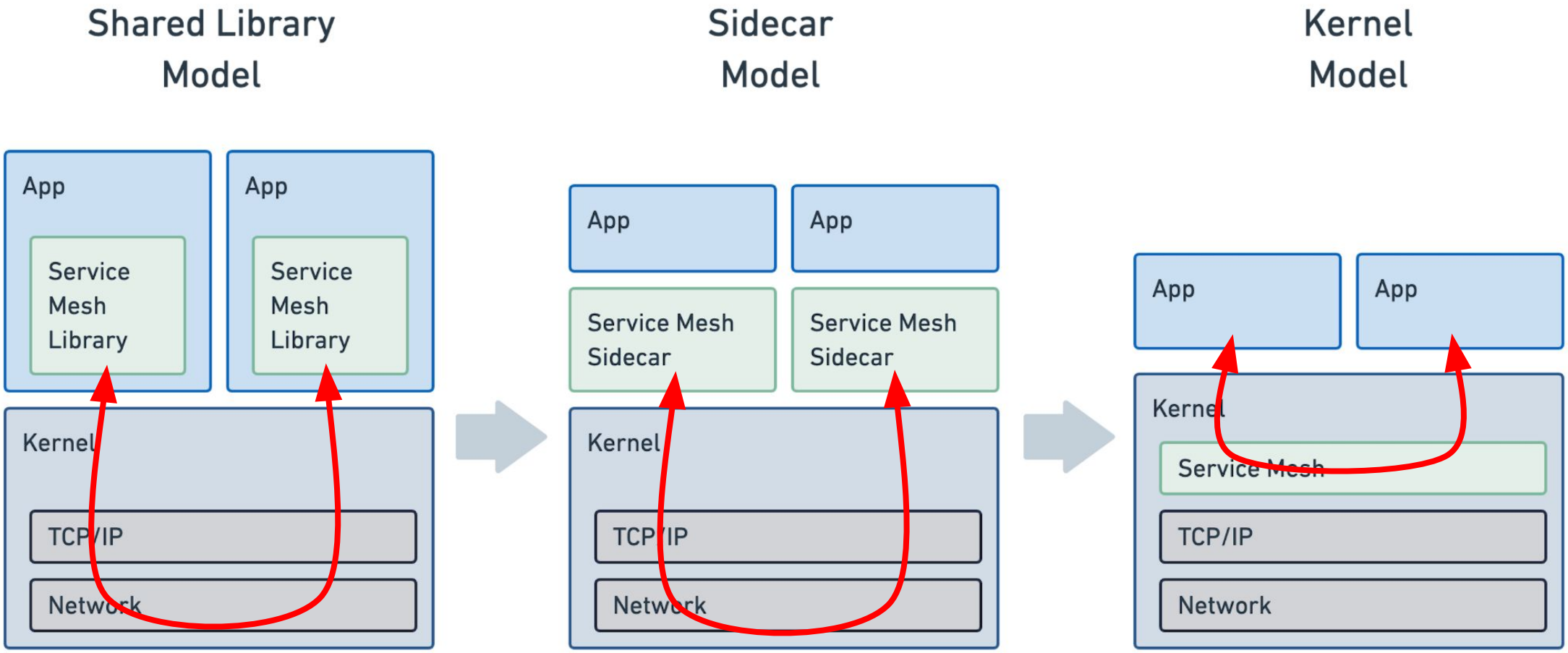
Benefits of eBPFing all the things



Benefits of eBPFing all the things



Benefits of eBPFing all the things



Learn more about eBPF

- Read the fine manual on ebpf.io

- Take eBPF for a spin locally with `bcc` and `bpftrace`
  - `dnf install bcc-tools`
    - `sudo /usr/share/bcc/tools/biosnoop`
  - `dnf install bpftrace`
    - `sudo bpftrace /usr/share/bpftrace/tools/biosnoop.bt`

What is OpenTelemetry?

What is OpenTelemetry?

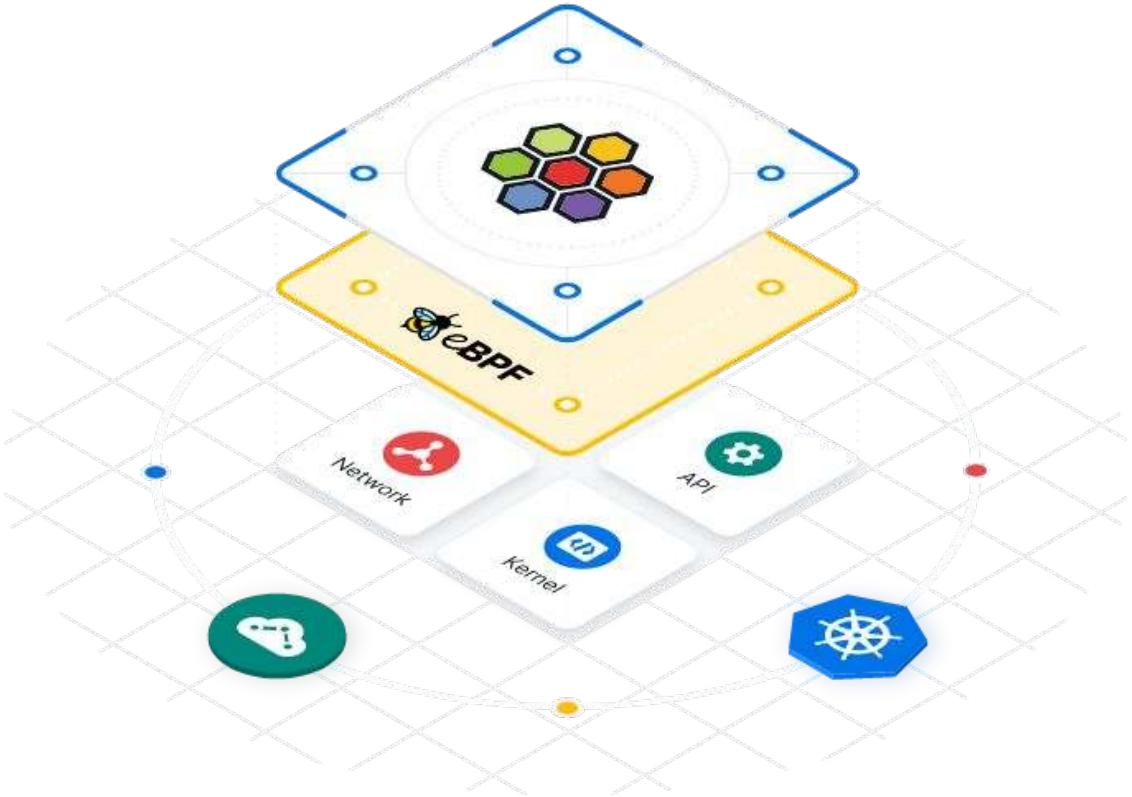
- High-quality, ubiquitous, and portable telemetry
- OpenTelemetry is the result of a merge of two projects
  - OpenTracing – open Telemetry API
  - OpenCensus – open source libraries to instrument code
- Instrument your code once, run it anywhere
- Support for traces, metrics and logs
- Supported by Red Hat OpenShift

Where to get more OpenTelemetry

- Check out opentelemetry.io
- SDKs for all the languages:
  - C++
  - .NET
  - Go
  - Java
  - JavaScript
  - Python
  - And more

What is Cilium?

What is Cilium?



- eBPF is complex and needs higher order tooling to leverage it properly
- Cilium provides, secures, and observes network connectivity between container workloads using eBPF

What does Cilium do?

- high-performance networking
- multi-cluster and -cloud capabilities
- advanced load balancing
- transparent encryption
- network security capabilities
- transparent observability
- much more

Features and Roadmap

- ✅ eBPF Networking (CNI, LB, Policy, …)
- ✅ ClusterMesh (Multi-Cluster CNI)
- ✅ Observability (Hubble)
- ✅ Service Mesh (Ingress)
- ⏳ SPIFFE, Gateway API, Transparent encryption, BGP, …
- ⏳ CNCF Graduation
- ⏳ Your awesome contribution

Cilium on OpenShift

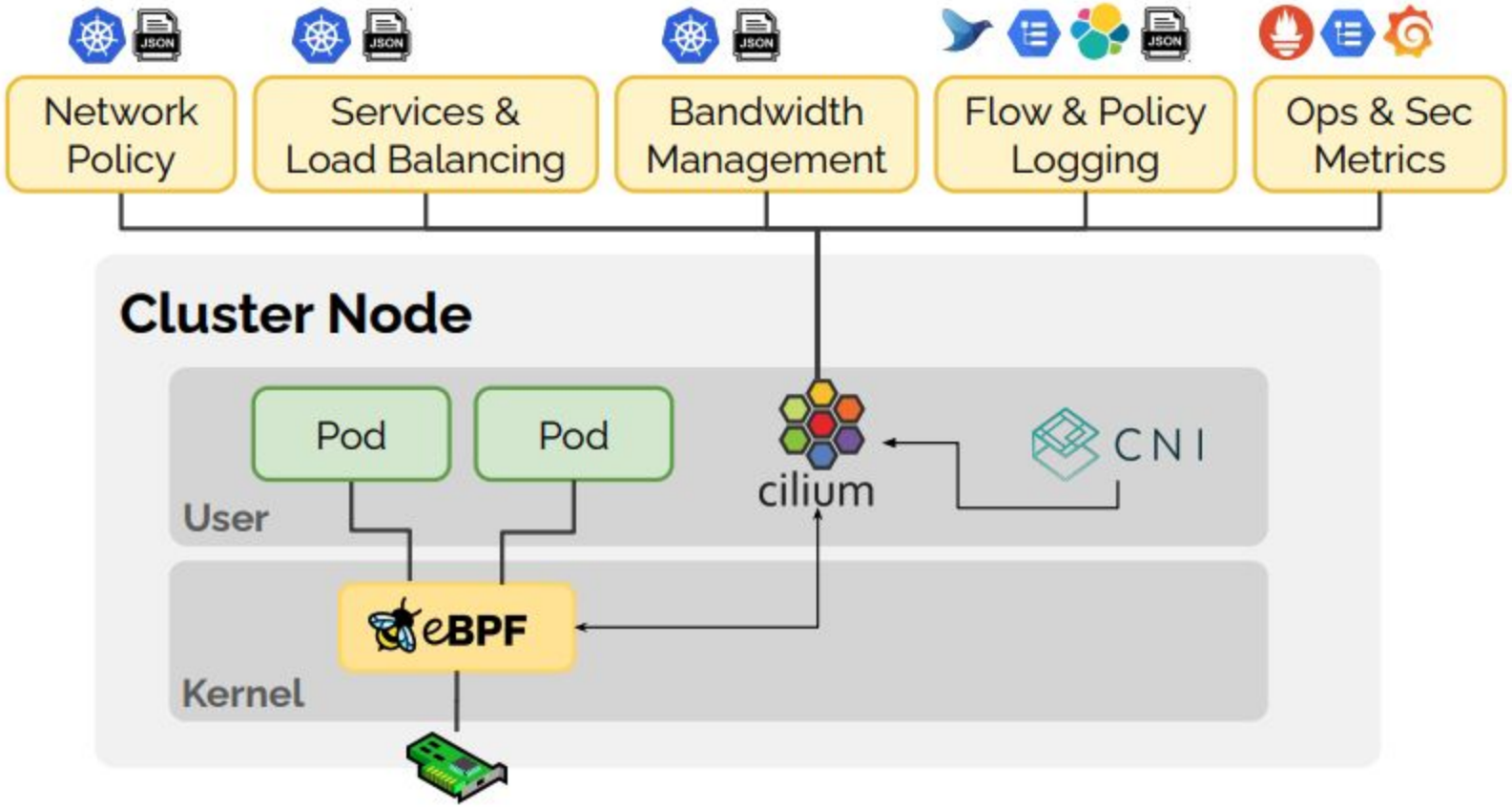
Installing Cilium on OpenShift

- Installation has to be done on a new cluster
- Installation uses the Cilium Operator certified by Red Hat
- If bootstrapping fails
  - Ensure that network configuration matches
    - `clusterPoolIPv4PodCIDR` must match OpenShift `clusterNetwork` CIDR
  - Make sure you configured `Cilium` as `networkType`
- Grab a coffee and wait for your cluster to become ready

What's running in the cluster already

- Cilium Operator managing the deployment and configuration
- Cilium is now configured as your CNI solution
- Cilium agent deployed on each node
- Cilium eBPF programs are running in the kernel
- All endpoints have been assigned an identity
  - Identities are managed by Cilium
  - Used to perform mTLS authentication between services

Cilium components



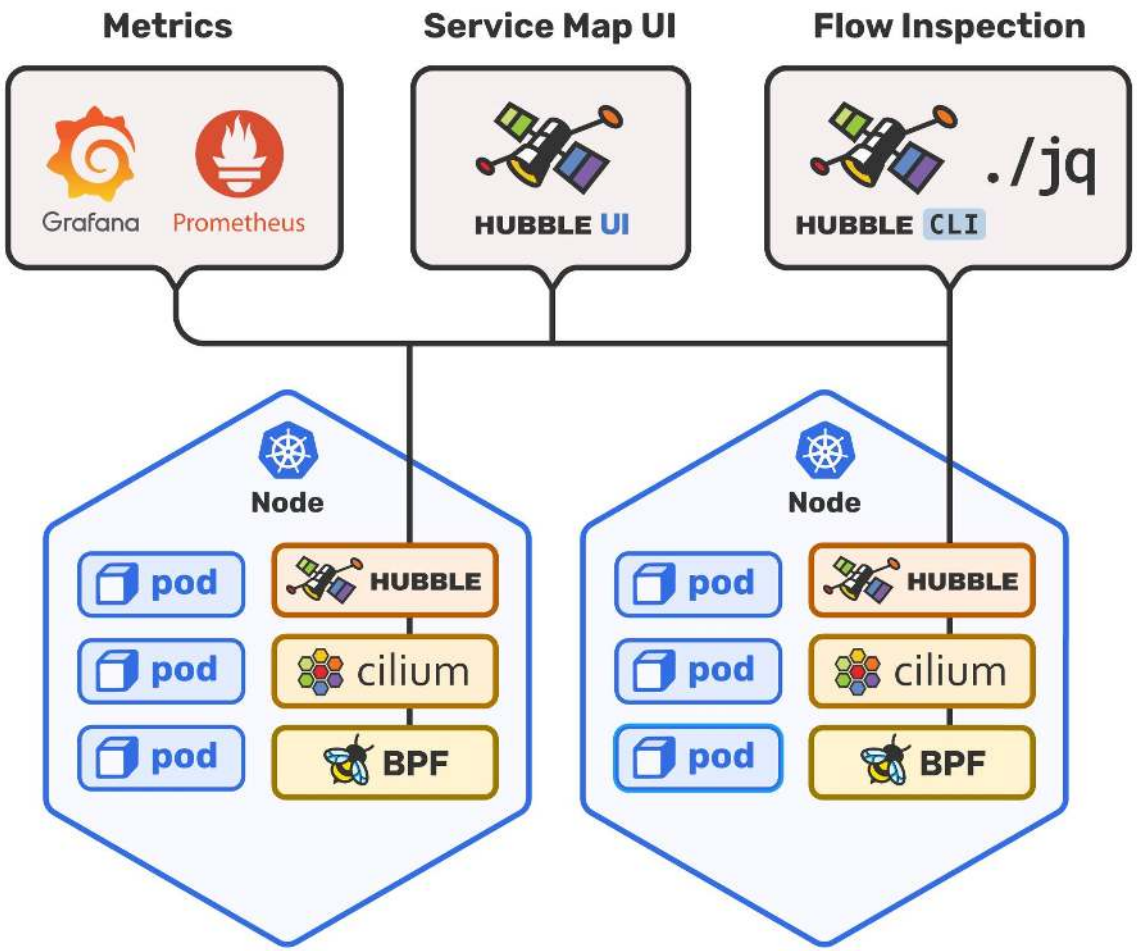
Source: cilium_overview.png on GitHub

Hubble

What is Hubble?

- Optional observability component integrated in Cilium
- Enables insights into any network connection in the cluster
- Enables live debugging of network traffic up to layer 7
- Provides insight into communication patterns
  - What endpoints does my application talk to
- Provides insight into your policies
  - Which connections are being blocked

What is Hubble?



Source: hubble_arch.png on GitHub

How to work with Hubble

- Hubble relay accesses observability data in Cilium agent
- Hubble UI to access the data accessible in the browser
- Hubble CLI to follow traces in the terminal
- Layer 3 & 4 are available out of the box
- Layer 7 requires Pod annotations to enable visibility
  - `io.cilium.proxy-visibility="<Ingress/8080/TCP/HTTP>"`
    - This will now route traffic via Envoy in the Cilium Agent

But what's next?

Attach Hubble to OpenTelemetry and push it to Jaeger

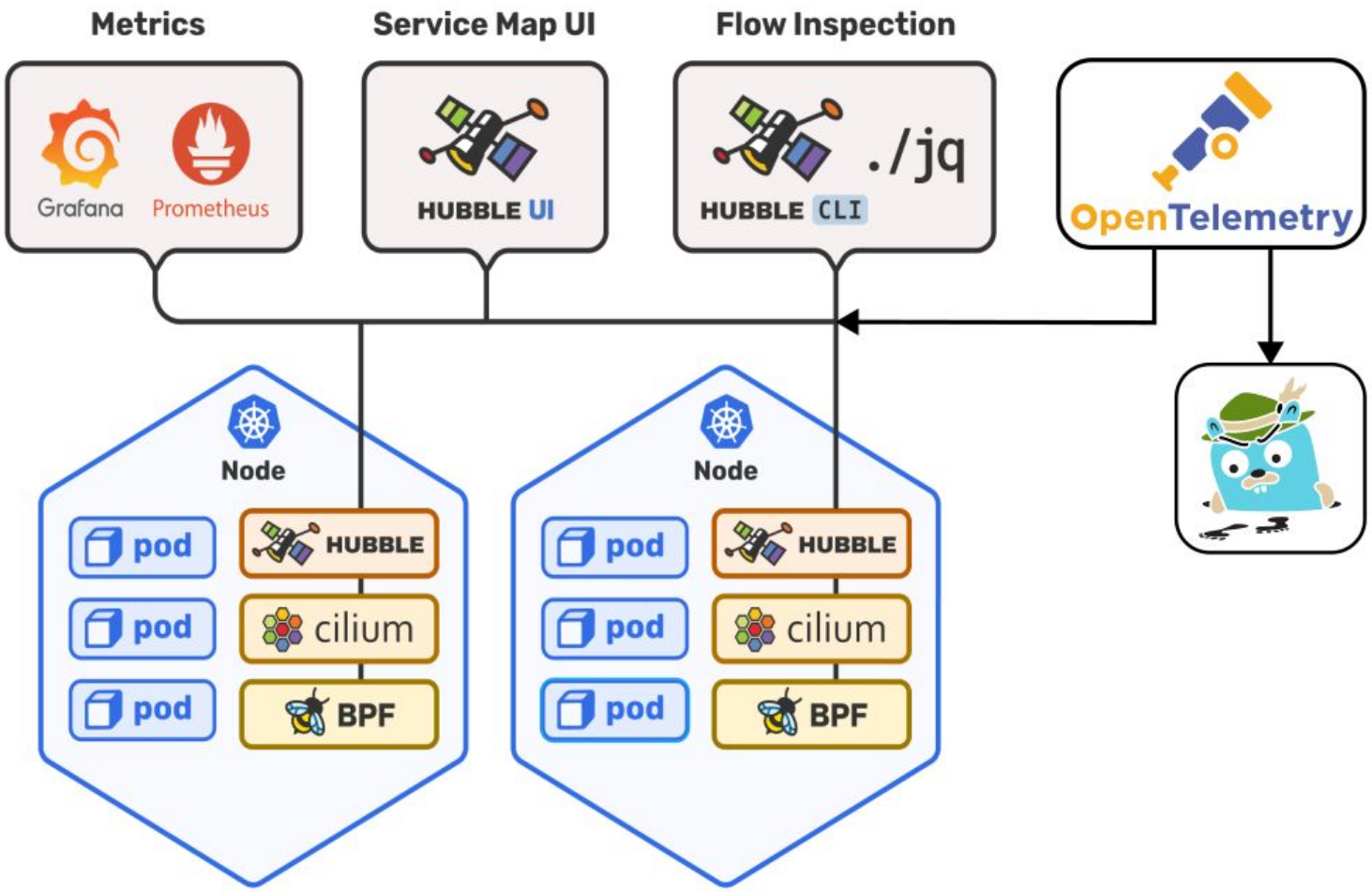
Source: David Bell on Twitter

Integrate Hubble and OpenTelemetry

OpenTelemetry on OpenShift

- Install two operators
  - Red Hat OpenShift distributed tracing platform – Jaeger
  - Red Hat OpenShift distributed data collection – OpenTelemetry Collector
- Deploy Jaeger
- Deploy OpenTelemetry Collector
  - Pull data from Hubble
  - Receive data from applications via OTLP
  - Forward everything to Jaeger

OpenTelemetry on OpenShift



Live demo 🤞

Live Demo 🤞

- Hubble UI: https://hubble-ui.apps.cilium-demo.os4.sycloud.ch/
- Hubble CLI
  - `cilium hubble port-forward --namespace cilium &`
  - `hubble observe --namespace cilium-demo --follow --type l7`
- Jaeger UI:
  - Hubble traces: https://jaeger/search?service=cilium-demo
  - OTLP traces: https://jaeger/search?service=cart-service

Wrap up

What is possible today

- Deep network insights without sidecar injection
- Layer 7 observability without any instrumentation
- Network policies up to layer 7 for supported protocols
- Mutual authentication between endpoints
- Transparent encryption between endpoints
- Dependency graph between endpoints

What are the current limitations

- Context propagation in traces is not yet supported
  - Only a request and its response is visible in Jaeger
- Cilium should not (yet) stop you from instrumenting your code
- Hubble metrics and OpenTelemetry integration are not yet stable

What's next

- ServiceMesh will improve Ingress support
- ServiceMesh will gain Gateway API support
- ClusterMesh will become topology aware

Check the roadmap for more details:

https://docs.cilium.io/en/stable/community/roadmap/

Questions

Links

- Deployment instructions for Cilium on OpenShift

  https://github.com/tongpu/cilium-on-openshift

Stay in Touch

/adfinis

/adfinis

adfinis.com

info@adfinis.com

/adfinis