Every packet toward facebook.com has been processed by XDP BPF enabled application since May, 2017

- Nikita V. Shirokov, Facebook Traffic team

source: LPC Net 2018,
http://vger.kernel.org/lpc_net2018_talks/LPC_XDP_Shirokov_v2.pdf

Least Privilege Network Security with BPF for Kubernetes

by

Arvind Soni

Product @ Isovalent, Open Source Cilium

@soniarvind @ciliumproject

Agenda

- BPF: The Revolution in Linux Kernel
- Cilium = BPF for Kubernetes Networking & Security
- Least Privilege Network Security Using Cilium
- Demo
- Sneak Peak at Cool Stuff … Coming Soon

The BPF Revolution

"With BPF, superpowers are coming to Linux"

Brendan Gregg
Lead Performance Engineer, Netflix

- L3-L4 Load balancing
- Network security
- Traffic optimization
- Profiling

redhat

- Replacing iptables with BPF
- NFV & Load balancing (XDP)
- Profiling & Tracing

Google

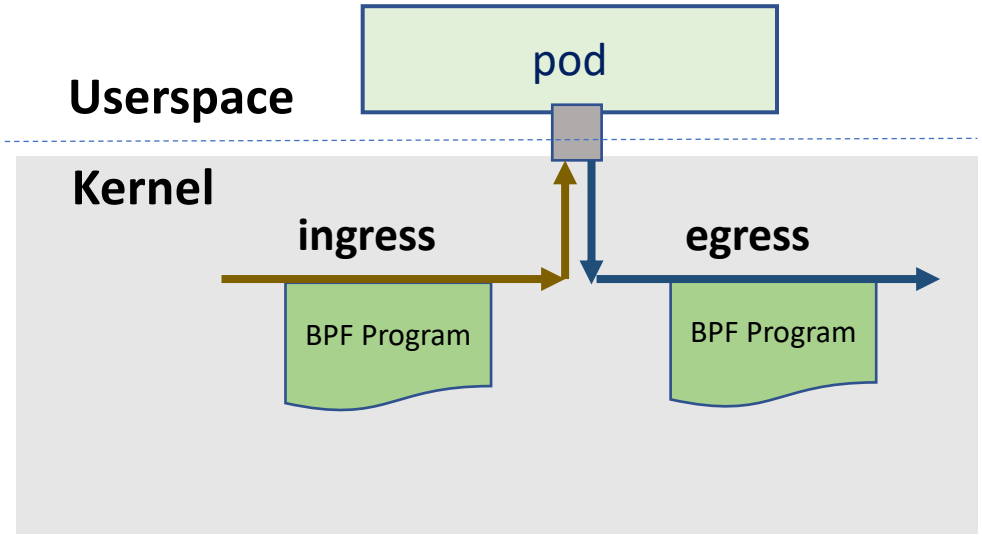
- QoS & Traffic optimization
- Network Security
- Profiling

NETFLIX

- Performance Troubleshooting
- Tracing & Systems Monitoring
- Networking

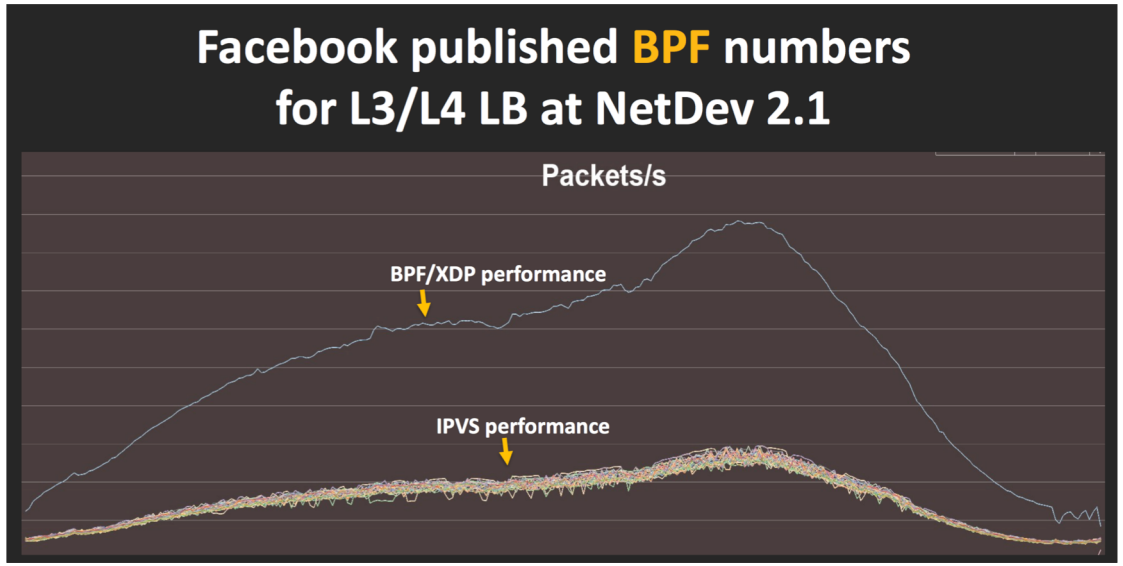
BPF in a Nutshell

- **BPF - Berkeley Packet Filter**
- **Bytecode** Programs Attached to **Kernel Events**
- **Verified & Secure** – Execution Safe
- **Efficient** – JIT Compiled
- **Available** in All Major Linux Distros
- **Not a Kernel Module**



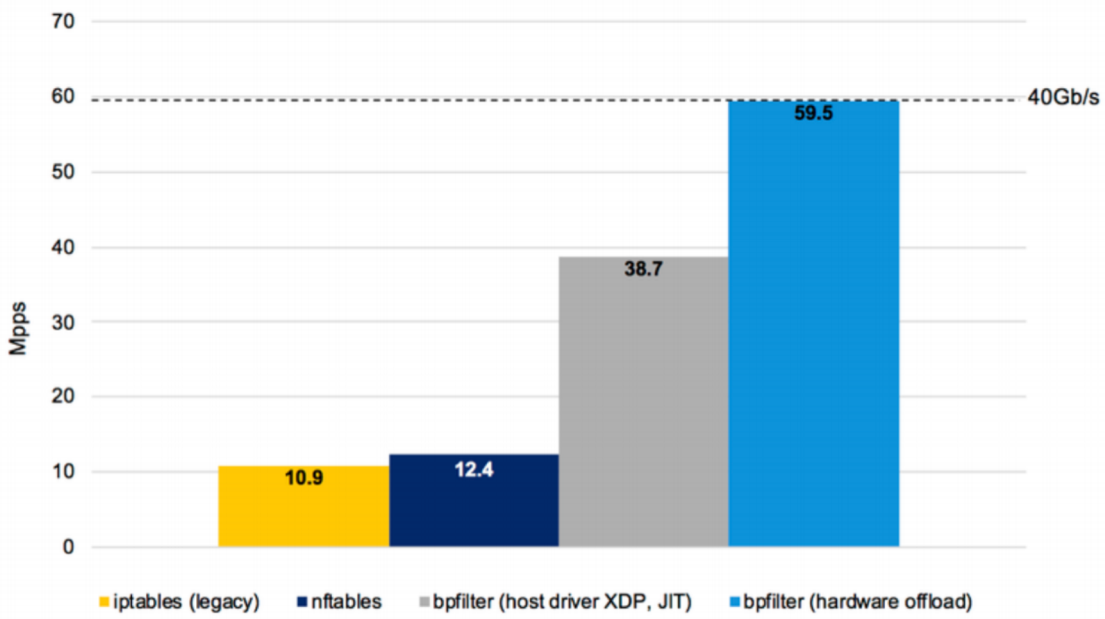
BPF in Action

XDP and BPF Based Load Balancing
@ Facebook
(OSS Facebook Katran)



Source: https://www.netdevconf.org/2.1/slides/apr6/zhou-netdev-xdp-2017.pdf

BPF Based Traffic Filtering (Firewalls)

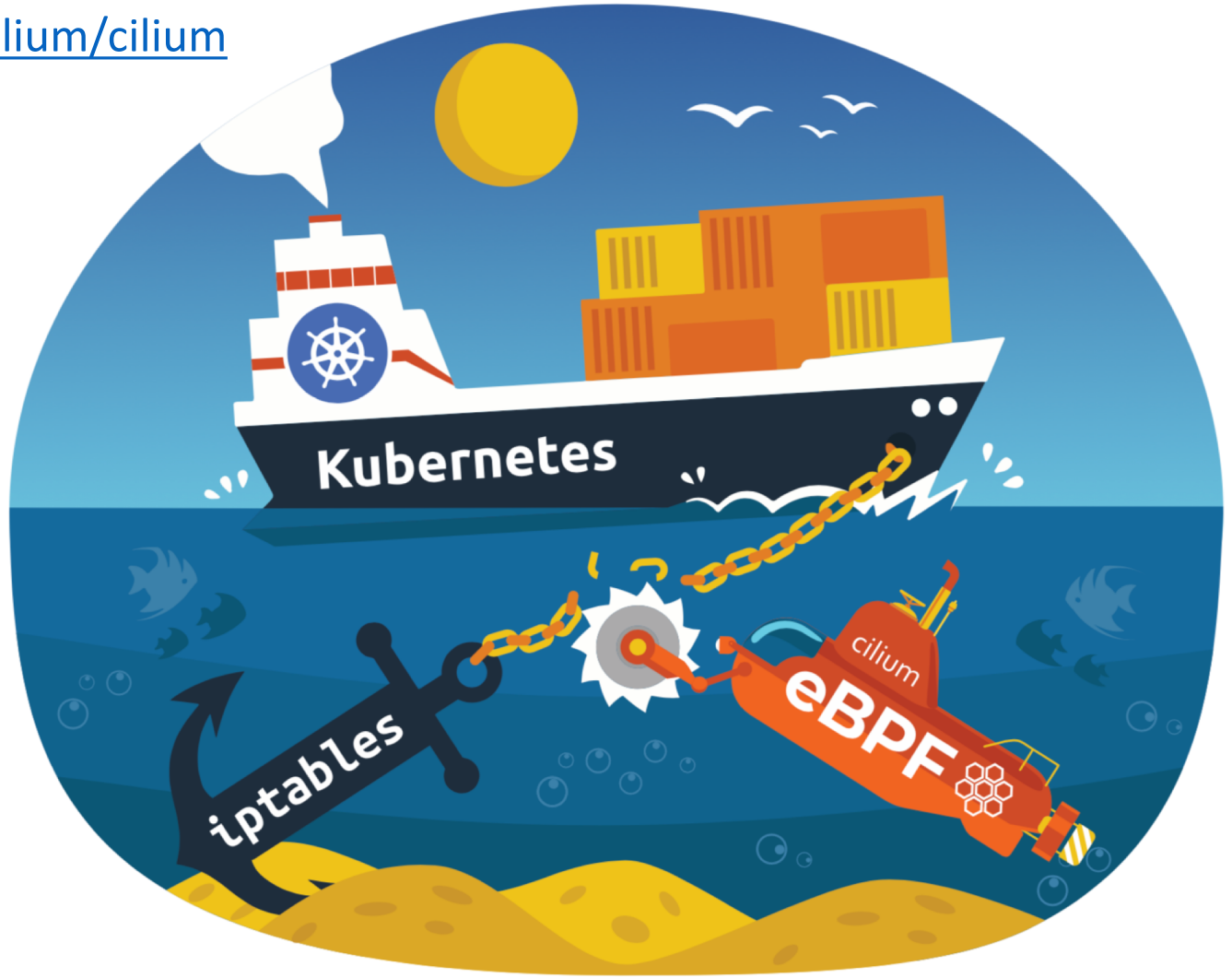


Source: https://bit.ly/2Ks9gJH

Open Source Cilium Project:
BPF Powered Kubernetes Networking & Security

https://github.com/cilium/cilium

https://cilium.io



Kubernetes Least Privilege Network Security Requirements

Identity driven security

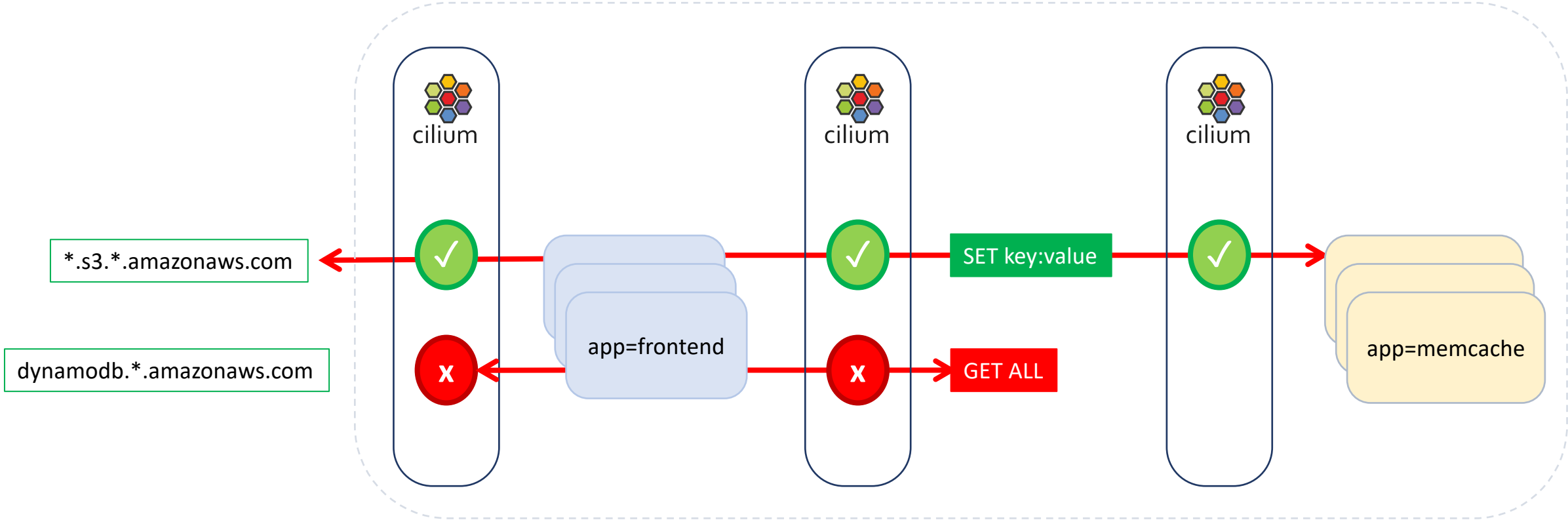
- Labels-based for cluster services
- DNS-based for external services

Efficient Enforcement

- IPTables for static, monolith era
- Handle churn and scale

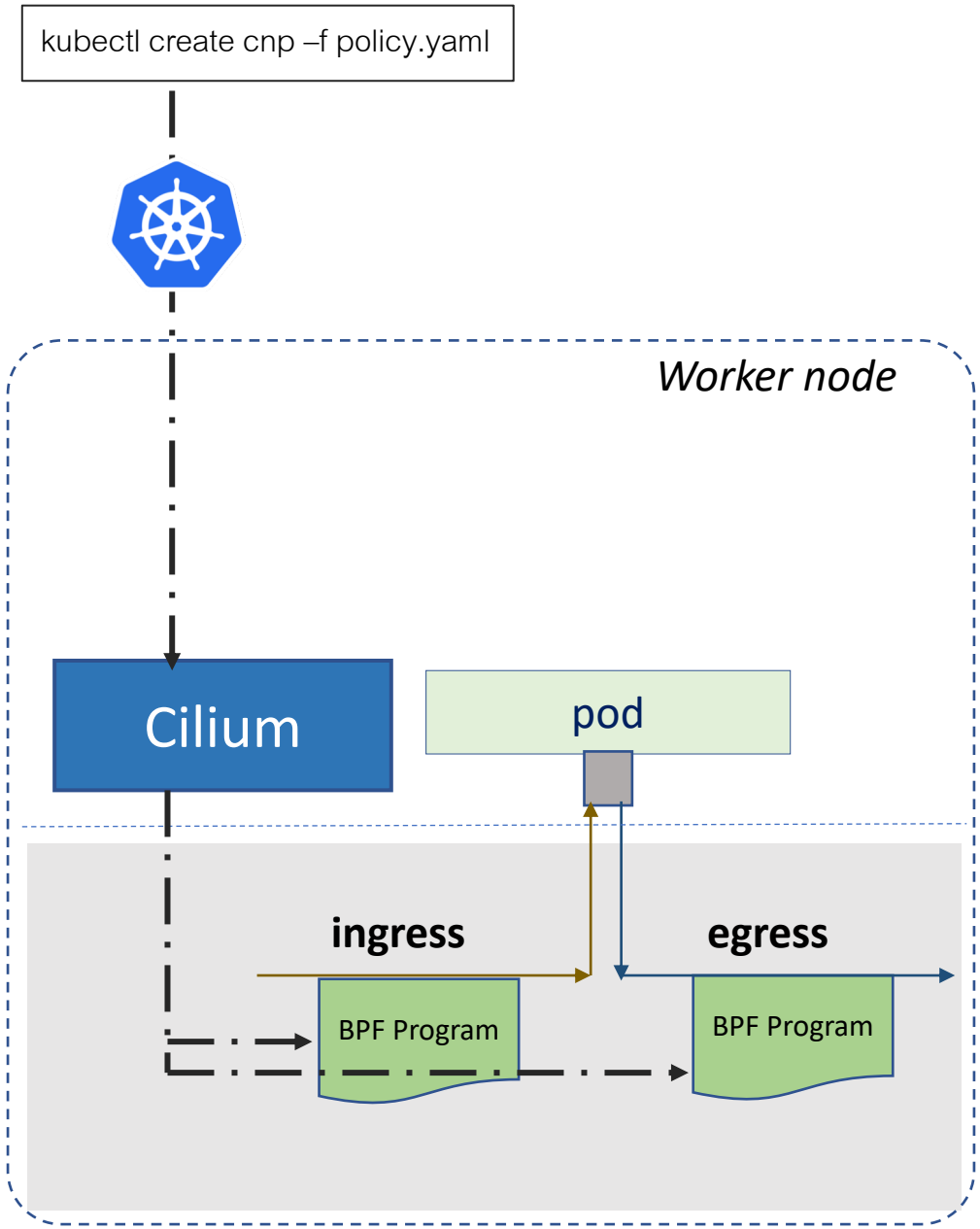
API-Aware Security

- L3/L4 is insufficient
- Many API interactions, more exposed attack surface



Cilium in a Nutshell

- Deploys as Daemonset
- Uses BPF for Enforcing Network Security
- Standard Kubernetes Network Policy
  - Default Deny then Whitelist
  - Endpoint Selectors
  - Ingress
  - Egress
  - Rules are all “OR”-ed



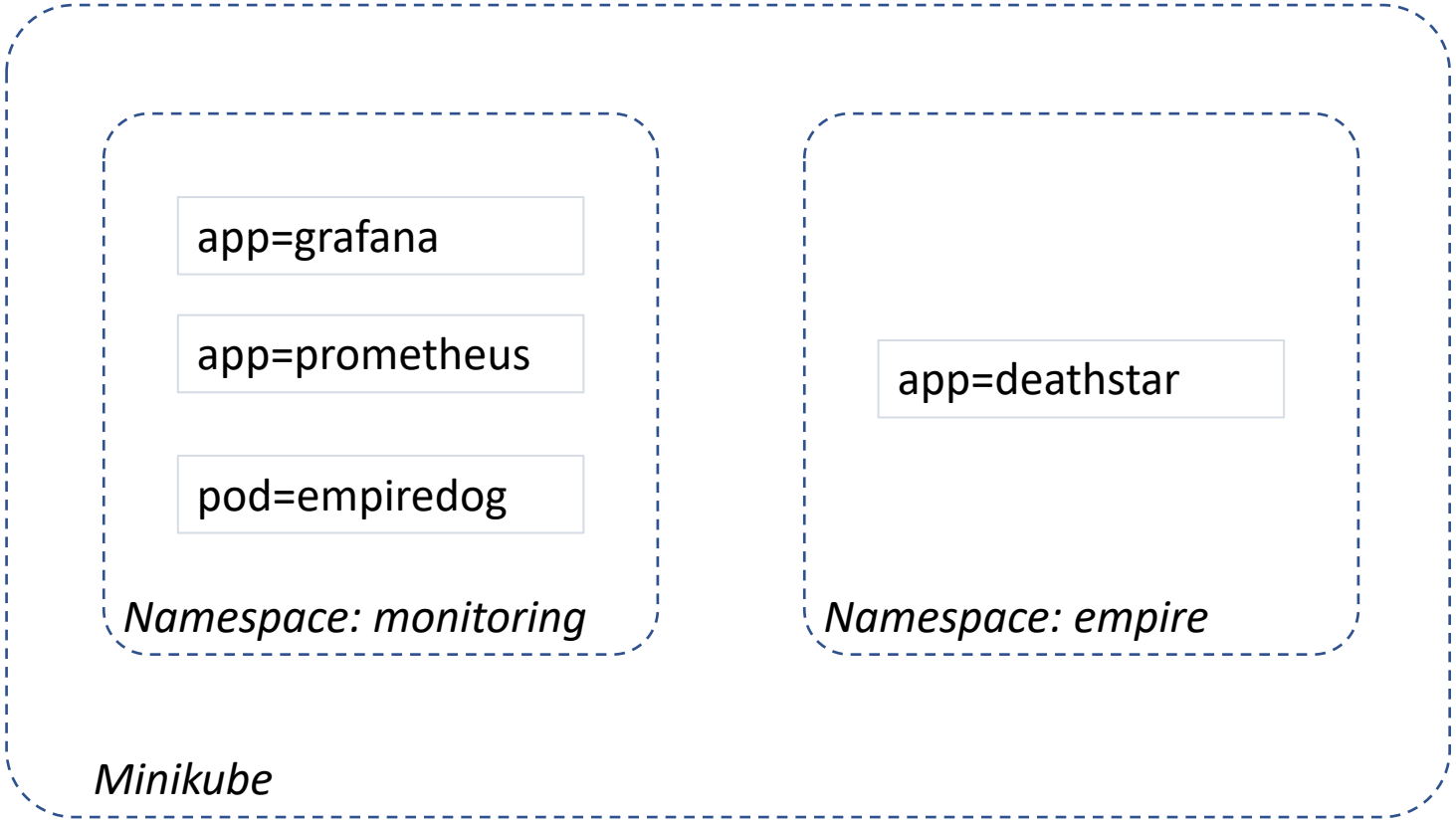
Demo Overview

- Allow all just within Namespace
- Allow Egress to kube-dns
- **“toService” –** control access to Kubernetes API Server
- **“toPorts” + “http”** – API-aware egress control
- **“fromEntities”** – ingress from host vs NodePort
- “**toFQDNs” –** control access to External Services using DNS patterns

Allow All Just Within Namespace

- Blocks Access to:
  - **Kubernetes API** Server
    - Mitigates risks such as the latest CVE-2018-1002105 (exploit against aggregated services)
  - **Kubelet** on local host as well as other worker nodes
    - Prevents backdoor access to kubelet (ports: 10255 / 10250 on worker nodes)
  - **Metadata** Service Access (169.254.169.254)
  - **Cross-namespace** (e.g. namespace mapped tenant isolation)
  - **Ingress/Egress to World**

Demo


app=grafana
app=prometheus
pod=empiredog
Namespace: monitoring
app=deathstar
Namespace: empire
Minikube

Recap

- DNS-based Policies with Patterns
- API-Aware Access Control
- Allow all just within namespace + egress to kube-dns
- Use *toServices* to control K8s API access
- Use *Entities* to control access to/from world

API-Aware Security for Popular Shared Services

Check out: Golang Extension Framework to Support Custom Protocols

Allow All Just Within Namespace

namespace label automatically inserted by Cilium for *matchLabels: {}*

```
spec:
  endpointSelector:
    matchLabels: {}
  ingress:
    - fromEndpoints:
        - matchLabels: {}
  egress:
    - toEndpoints:
        - matchLabels: {}
```

Allow Egress to kube-dns

Allow egress to kube-dns
- label-based identity for kube-dns service
- Access limited to specific port

```
spec:
  endpointSelector:
    matchLabels: {}
  egress:
  - toEndpoints:
    - matchLabels:
        k8s:io.kubernetes.pod.namespace: kube-system
        k8s-app: kube-dns
    toPorts:
    - ports:
      - port: "53"
        protocol: ANY
```

Ingress Control – World vs Host

- Ingress **from host** is always allowed
- Cilium differentiates between traffic **originating in host vs NodePort**
- **fromEntities : world** allows traffic from outside the cluster

```
spec:
  endpointSelector:
    matchLabels:
      app: grafana
  ingress:
    - fromEntities:
      - world
```

Access to Kubernetes API Service

- Only pods with *app=prometheus* allowed API access

- **toServices** specify Kubernetes services
- Cilium automatically tracks services endpoints

```
spec:
  endpointSelector:
    matchLabels:
      app: prometheus
  egress:
  - toServices:
    - k8sService:
        serviceName: kubernetes
        namespace: default
```

API-Aware Egress Control

- L4 (port-based) access control

- Granular L7 (API-aware) control
- Only **HTTP GET /metrics** is allowed

```
spec:
  endpointSelector:
    matchLabels:
      app: prometheus
  egress:
  - toPorts:
    - ports:
      - port: "9090"
        protocol: TCP
      rules:
        http:
        - method: "GET"
          path: "/metrics"
```

DNS-Based Egress Control

- Control the domains for which DNS look up can be done
- Supports patterns

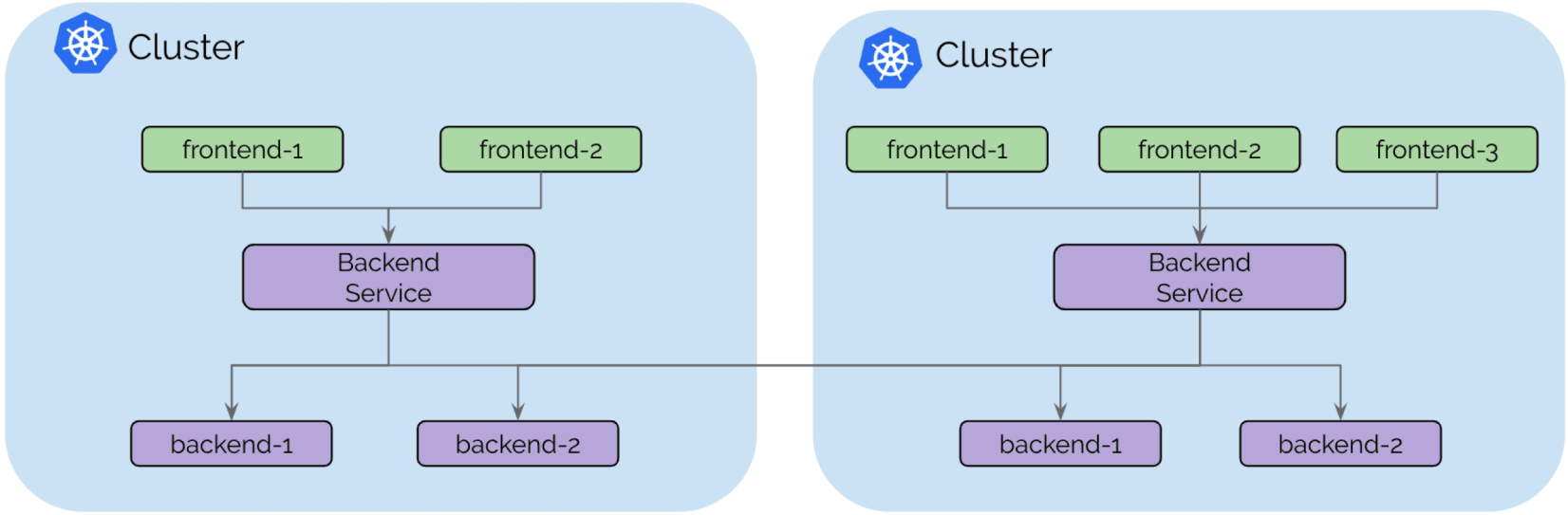
- Control access to external services by DNS
- Supports patterns

```
egress:
- toEndpoints:
  - matchLabels:
      k8s:io.kubernetes.pod.namespace: kube-system
      k8s-app: kube-dns
  toPorts:
  - ports:
    - port: "53"
      protocol: ANY
    rules:
      dns:
      - matchPattern: "*.s3.*.amazonaws.com”

- toFQDNs:
  - matchPattern: "*.s3.*.amazonaws.com"
  toPorts:
  - ports:
    - port: "443"
      protocol: TCP
```

ClusterMesh – Connect & Secure Multiple Clusters

- Cross-Cluster Service Access without any Ingress Controllers / LBs
- Service Security Identities Span Across Clusters



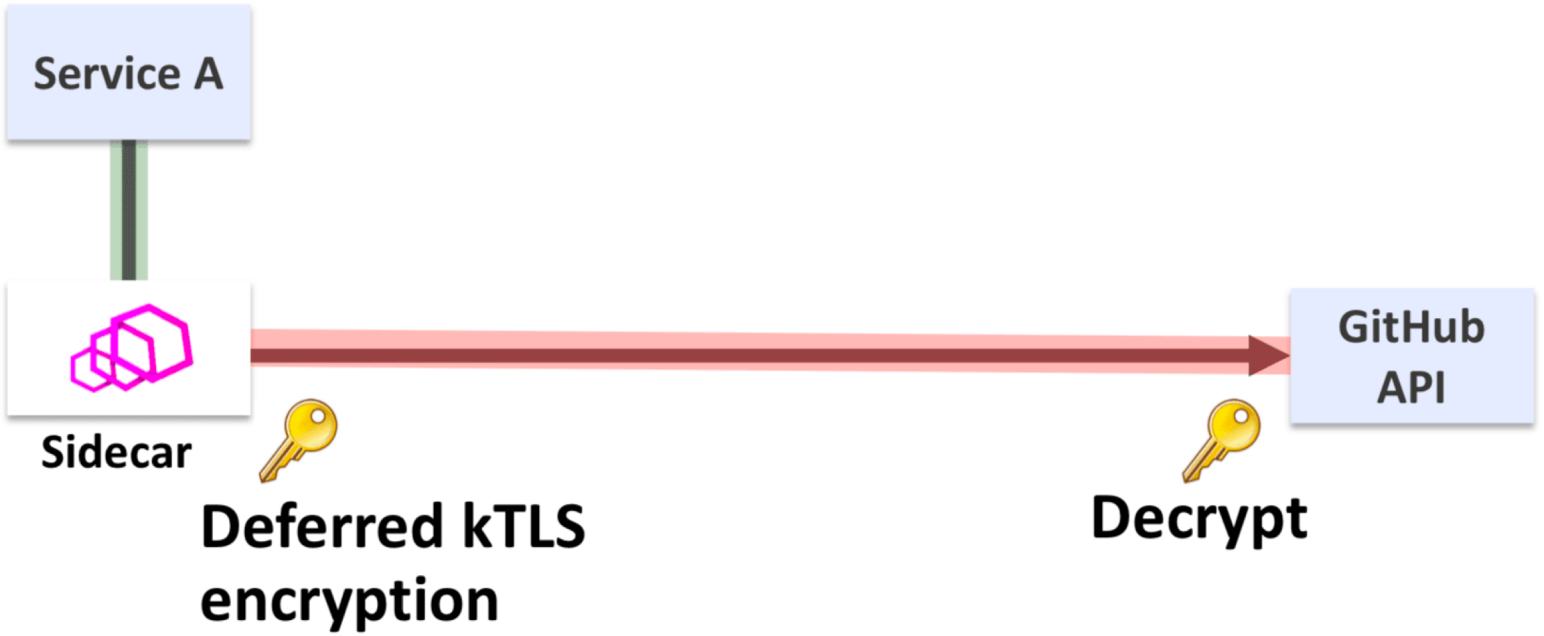

```
metadata:
  annotations:
    io.cilium/global-service: "true"
```

[Kubecon Talk: Connecting Kubernetes Clusters Across Cloud Providers by Thomas Graf]

[Blog: Cilium 1.4 Preview]

kTLS – Visibility and Security for SSL Traffic

- Symmetric Encryption Deferred to Kernel
- Asymmetric Key Exchange Remains Same
- No Trusted Man-in-the-middle, Root-CA Propagation Headaches!



[kTLS by Dave Watson @ Facebook]
[Blog: Cilium for Istio Servicemesh]

Thanks!
Questions?

https://cilium.io/

https://github.com/cilium/cilium

@ciliumproject

Join us @ Booth S9