September 25 - 27, 2018
Amsterdam, The Netherlands

ons

EUROPE

OPEN NETWORKING //
Integrate, Automate, Accelerate

ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

September 25 - 27, 2018
Amsterdam, The Netherlands

Traffic Management and Visibility Infrastructure for Rapid Microservice Delivery

Eddie Arrage
Futurewei Technologies Inc

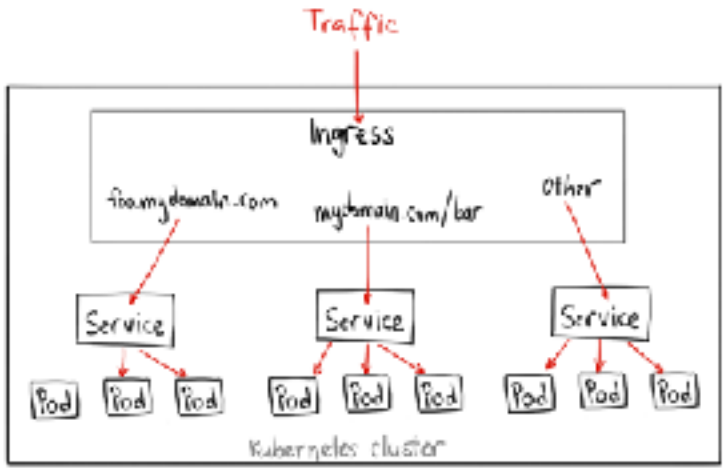
ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Agenda

- **Cloud Native Concepts (5 min)**
  - Microservices
  - Service Deployment Strategies
  - Challenges
- **Traffic Management and Service Meshes (15 min)**
  - Service Meshes / Istio
  - Mesh Traffic Management
  - Mesh Visibility Tools
- **Visibility/Observability Infrastructure Mesh/Non-Mesh (10 min)**
  - OPNFV Clover (+ Clovisor)

Cloud Native

- Benefits:
  - Portable
  - Scalable
  - Ephemeral
  - Accessible
  - Flexible



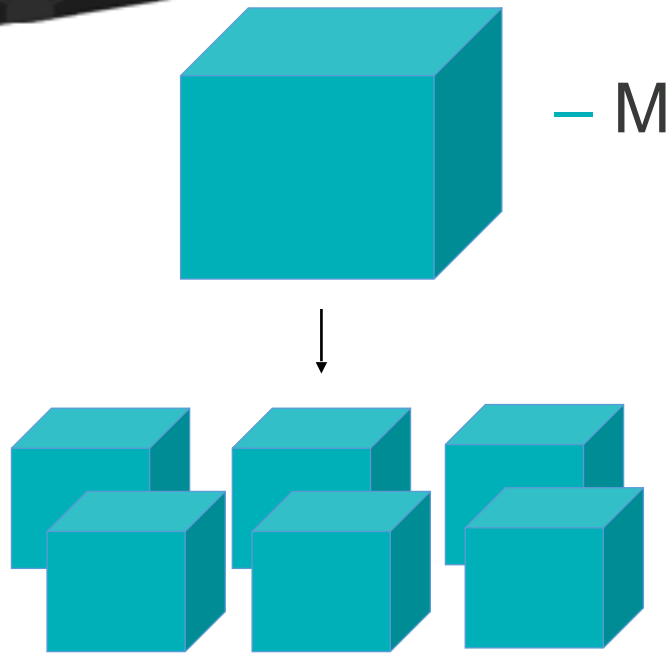
– Microservice oriented

kubernetes

– Dynamically managed (Kubernetes)

– Containerized

Microservices



– Monolithic App

– Break down into smaller chunks

- Microservice architecture puts functionality into separate services:
  - Iterative development
  - Division of labor
  - Reduce single point of failure
  - Language/deployment flexibility
  - Build different apps using subsets of services
  - Operations stakeholders are able to manage and upgrade components more easily

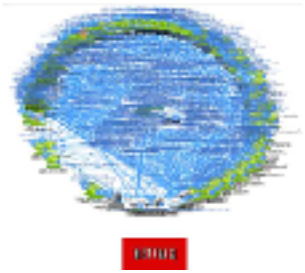
ons EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Microservice Validation & Deployment Strategies

- Blue/Green
  - Two identical environments of all microservices – Example:
    - Green in production
    - Release new version of service(s) in blue and validate
    - Revert to green if issues exist or cut over to blue if not
- A/B Testing
  - Support multiple versions of microservice simultaneously to compare variations/versions
- Canary
  - Push new code to small group of users to evaluate incremental changes
  - Early warning system for detecting problems

- Employ ingress network services for traffic management: load balancers, proxies and/or service meshes to support

Cloud Native / Microservice Challenges



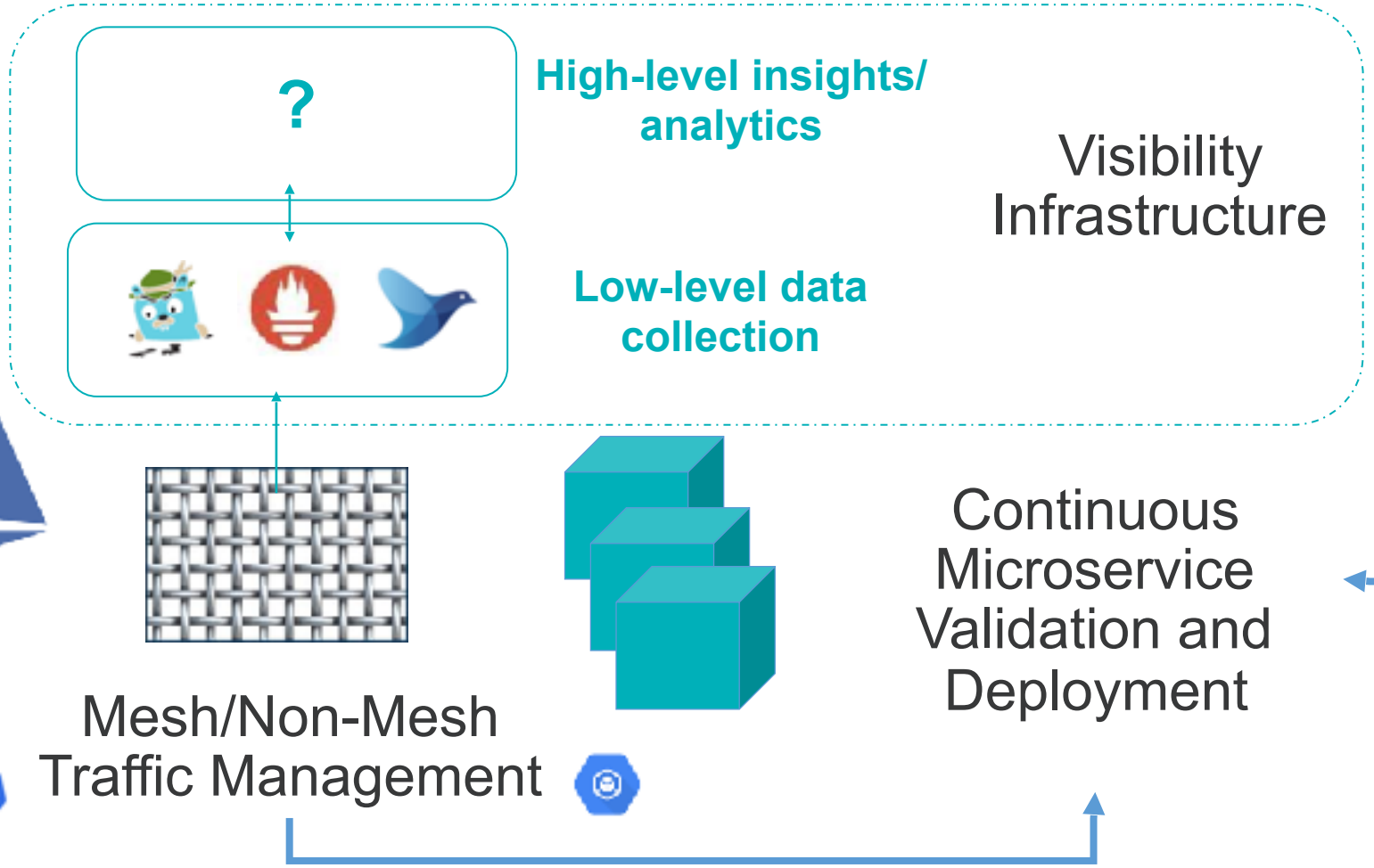
ONAP

150+ containerized services

- Microservice sprawl
  - Debug difficult without tools for visibility and traceability of entire system
- Microservice validation and deployment strategies require integrated traffic management
  - Current CI/CD pipelines in LFN projects have not adopted consist framework/methodology for doing this

ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Cloud Native Traffic Management & Visibility



ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

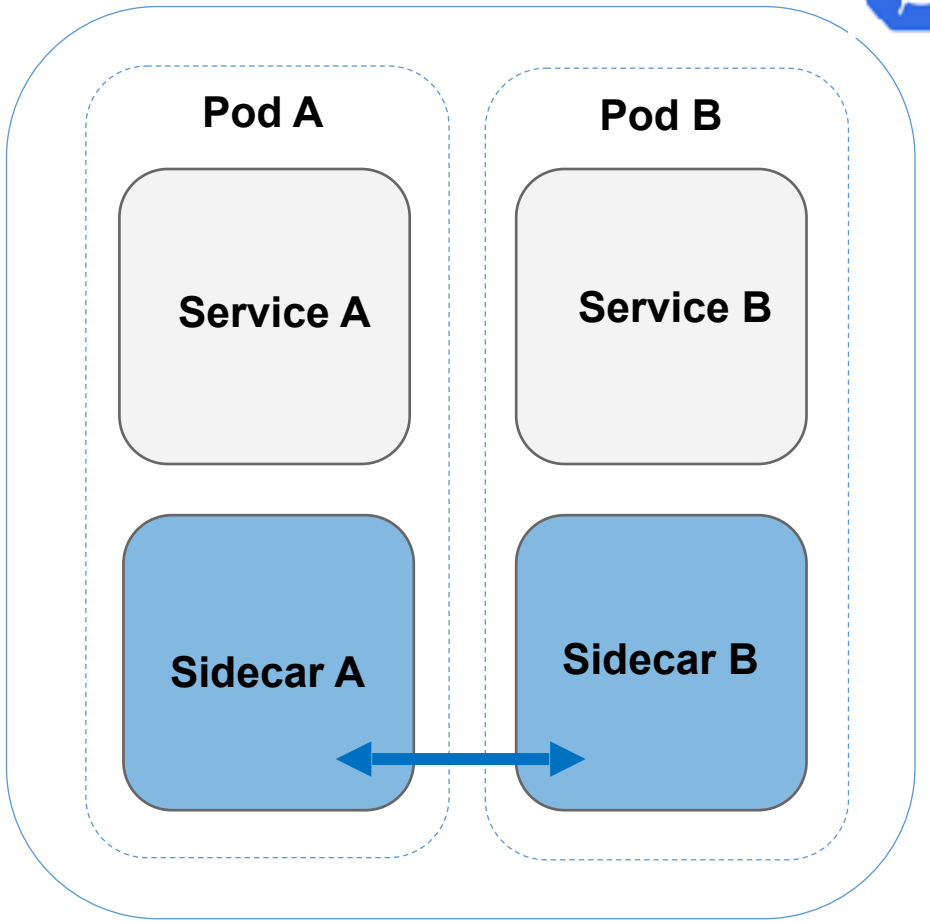
September 25 - 27, 2018
Amsterdam, The Netherlands

Traffic Management and Service Meshes

Service Meshes

- Dedicated layer for managing service communication
  - Intra-service within cluster
  - External traffic entering cluster (ingress)
  - Internal traffic leaving cluster (egress)
  - Fit best for control-plane services

– Examples: Istio, Conduit, Apache ServiceComb



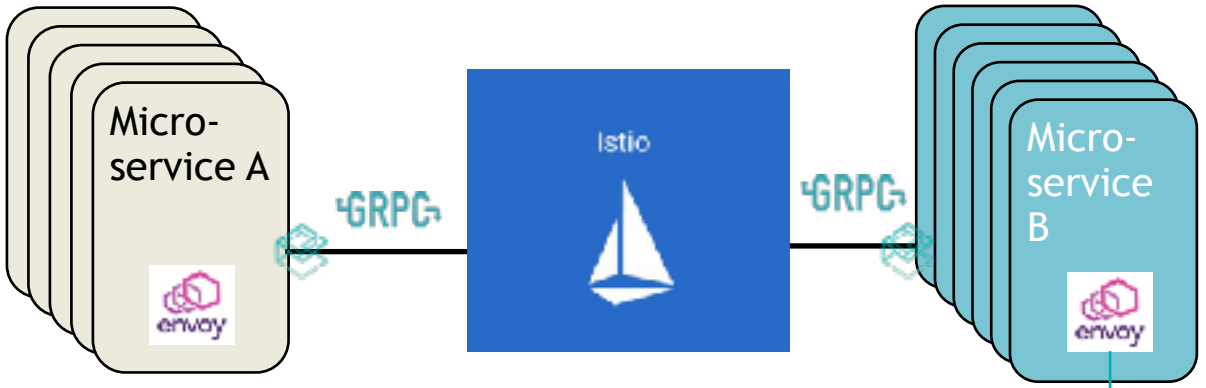
– 'Sidecar' injected as a service proxy in each pod

– Allows for more advanced routing than native k8s networking

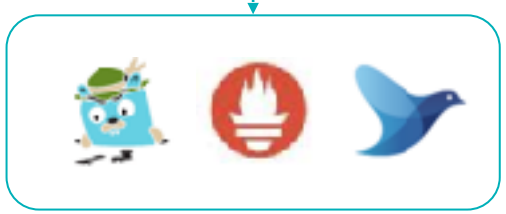
Istio Service Mesh

- Traffic Management
  - Load balancing
  - Request routing
  - Continuous deployment
    - Canary
    - A/B validation
  - Fault injection
  - Mirroring
  - Secure communication

- Proxy oriented to HTTP/gRPC
- mTLS (optional)



- Manual or automatic (namespace) sidecar injection
- Toggle in/out of mesh easily



- Visibility Built-in
  - Monitoring, tracing, logging

Istio Install

- Current release at 1.0.2,
- Works best on k8s v1.9+ (with mutating webhook)

```
$ curl -L https://git.io/getLatestIstio | sh -
$ cd istio-1.0.2
$ export PATH=$PWD/bin:$PATH
$ kubectl apply -f install/kubernetes/istio-demo.yaml
```

Install

```
$ kubectl label namespace <namespace> istio-injection=enabled
$ kubectl create -n <namespace> -f <your-app-spec>.yaml
```

```
$ istioctl kube-inject -f <your-app-spec>.yaml | kubectl apply -f -
```

Setup

- automatic sidecar (namespace) sidecar injection
- Manual sidecar injection

```
$ docker pull opnfv/clover:latest
$ sudo docker run --rm \
  -v ~/.kube/config:/root/.kube/config \
  opnfv/clover \
  /bin/bash -c '/home/opnfv/repos/clover/samples/scenarios/deploy.sh'
```

Install Istio and SDC sample with Clover

ons EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Network Service Catalog

Service	Kubernetes Deployment App Name	Docker Image	Ports
Proxy	proxy access control	clover ns nginx proxy	HTTP: 9180 GRPC: 50051
Load Balancers	app: http-lb version: http-lb-v1 version: http-lb-v2	clover-ns-nginx-lb	HTTP: 9180 GRPC: 50054
Intrusion Detection System (IDS)	snort ids	clover ns snort ids	HTTP: 80, Redis: 6379 GRPC: 50052 (config) GRPC: 50051 (alerts)
Servers	clover-server1 clover-server2 clover-server3 clover-server4 clover-server5	clover-ns-nginx-server	HTTP: 9180 GRPC: 50054

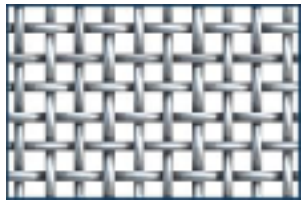
OPNFV Docker Hub Images

- Clover developing set of sample L7 network services for use in k8s and meshes
- New in Clover Gambia release: modsecurity

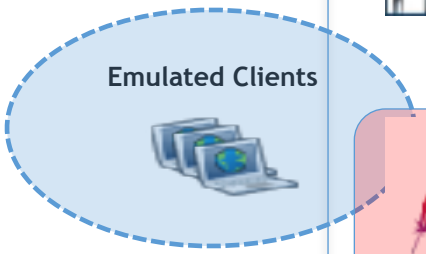
(Web Application Firewall + Apache web server)

ons EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Traffic within Mesh

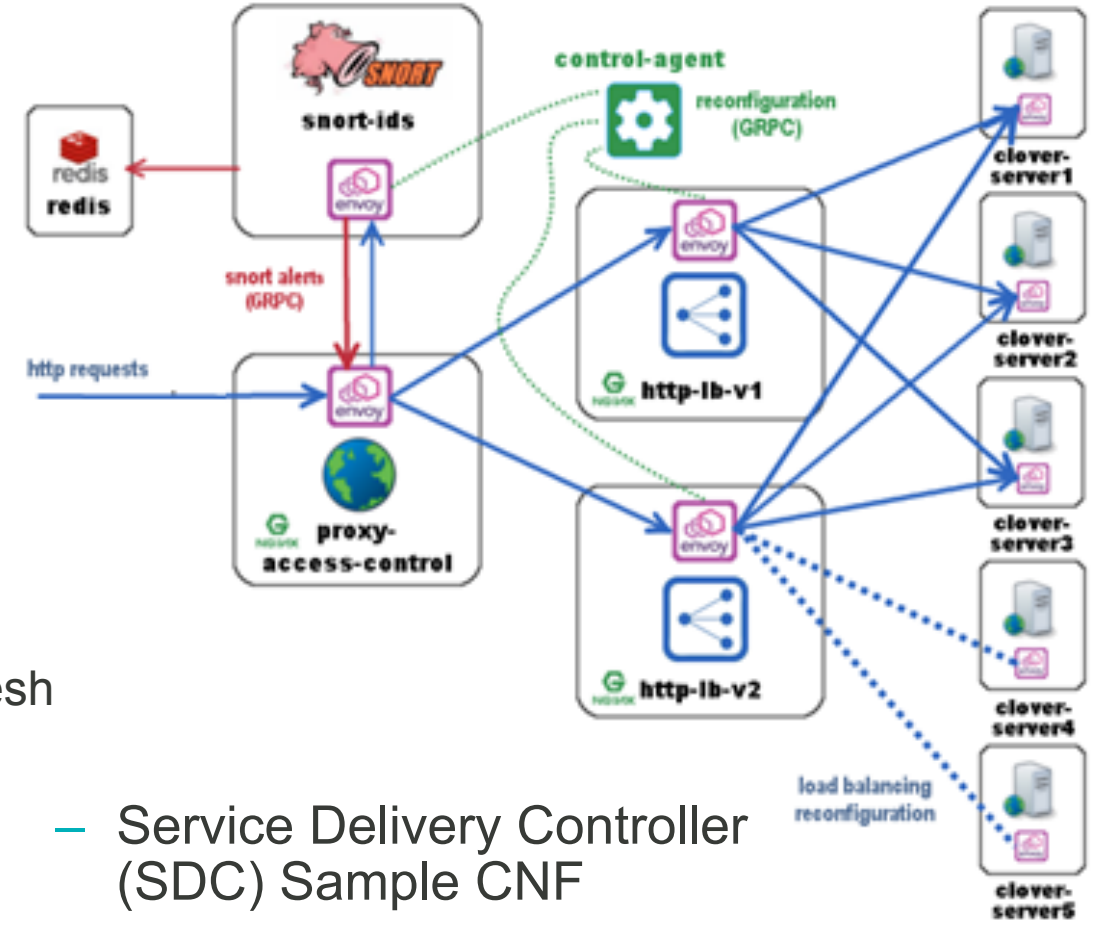


k8s



clover-jmeter-master

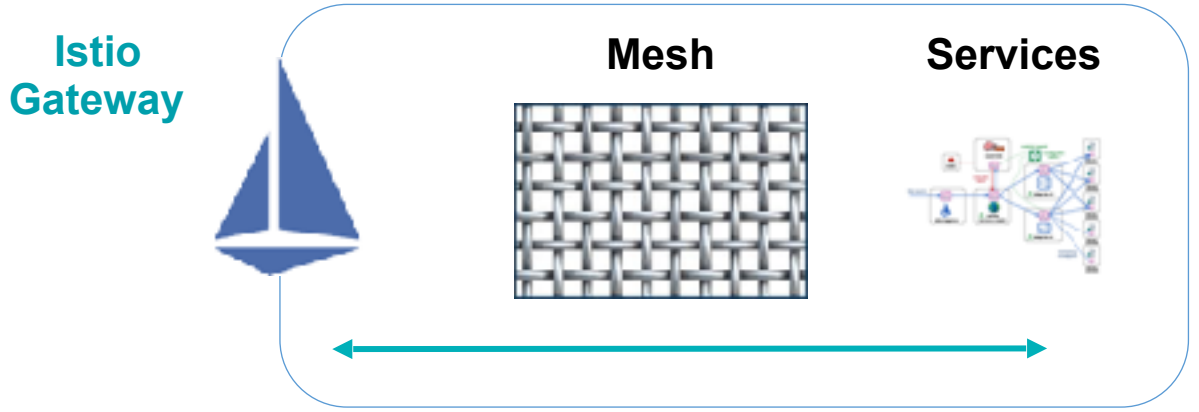
- Inject jmeter into mesh
- Send traffic within cluster/mesh



- Service Delivery Controller (SDC) Sample CNF

External Traffic into Mesh

Istio Ingress




```yaml
apiVersion: networking.istio.io/v1alpha3
kind: Gateway
metadata:
  name: sdc gateway
spec:
  selector:
    istio: ingressgateway
  servers:
  - port:
      number: 80
      name: http
      protocol: HTTP
    hosts:
    - "*"
---
```

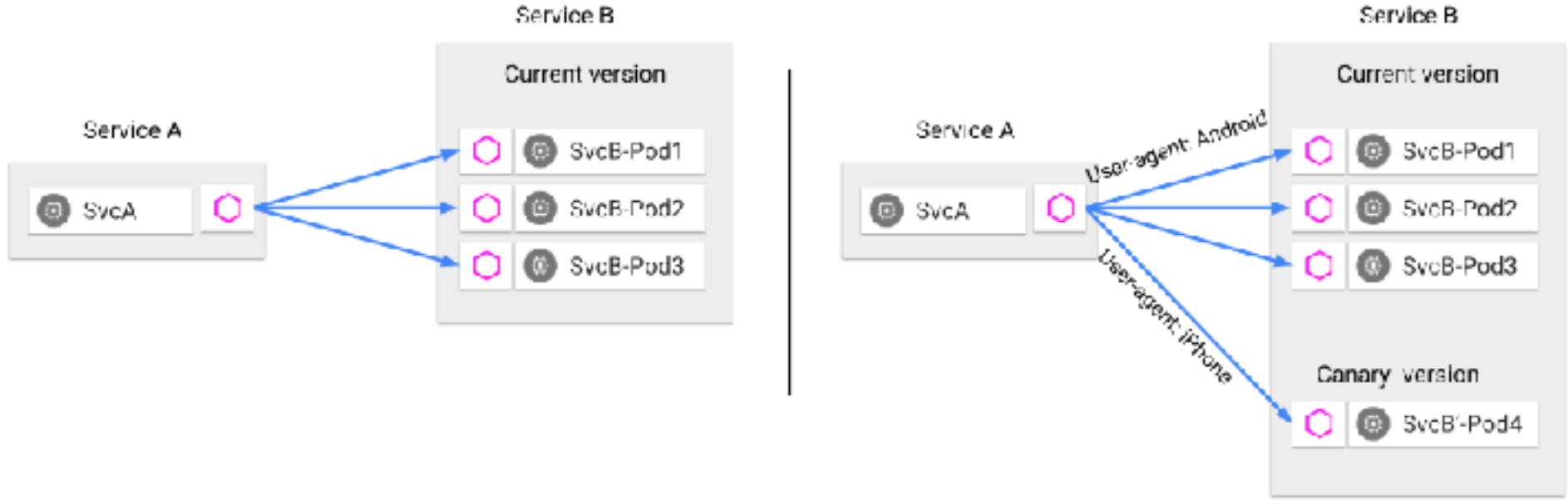
- LB at the edge of mesh receiving incoming/outgoing connections

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: sdcsample
spec:
  hosts:
    "*"
  gateways:
  - sdc-gateway
  http:
  - match:
      uri:
        prefix: /
    route:
    - destination:
        port:
          number: 9180
        host: proxy access control
```

Virtual Service

- Control how traffic is routed within the mesh

Istio Request Routing (1-2)



- Content-based steering to determine destination of request

ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Istio Request Routing (2-2)

- Flexible request routing with Virtual Service
  - Match traffic and route to back end service
  - Match based on URI, HTTP headers (identity, user-agent)
  - Control with ‘weight’ field
- Ideal to validate REST based APIs and services
  - Support CI/CD deployment workflows
  - Canary validation/deployment

URLs to domain www.sdc.com

Match URI prefix ‘/test’ to clover-server2

Match HTTP header user-agent ‘chrome’ to clover-server3

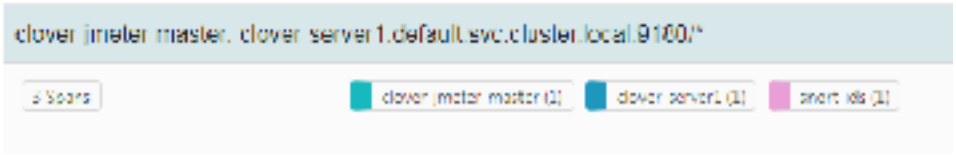
Everything else to clover-server1

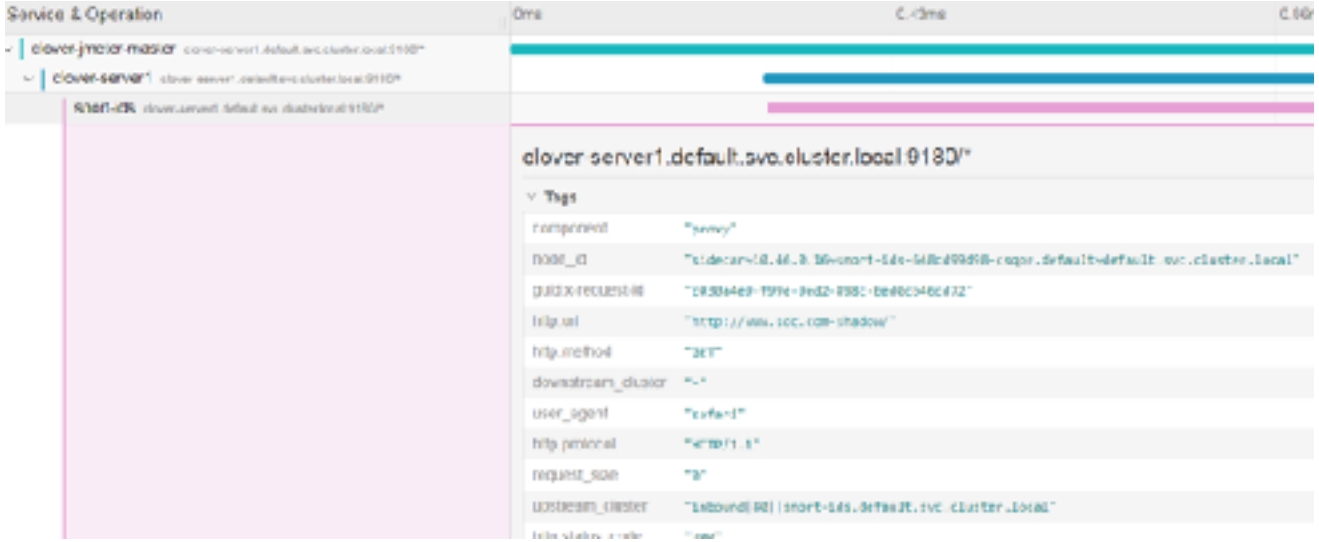
```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: directserver
spec:
  hosts:
    - "www.sdc.com"
  http:
  - match:
      uri:
        prefix: /test
    route:
    - destination:
        port:
          number: 9180
        host: clover-server2
  - match:
      headers:
        user-agent:
          exact: chrome
    route:
      destination:
        port:
          number: 9180
        host: clover-server3
    route:
    - destination:
        port:
          number: 9180
        host: clover-server1
```

ons EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Istio Mirroring

- Mirroring or Shadowing
  - Sends a copy of live traffic to a mirrored service
  - Add an entry to Virtual Service resource under any route rule





Any traffic to clover-server1 mirrored to snort-ids

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: directserver
spec:
  hosts:
      "www.cdc.com"
  http:
  - match:
    - uri:
        prefix: /test
    route:
    - destination:
        port:
          number: 9180
        host: clover-server2
  - match:
      headers:
        user-agent:
          exact: chrome
    route:
    - destination:
        port:
          number: 41000
        host: clover server3
  - route:
    - destination:
       port:
         number: 9180
       host: clover-server1
    mirror:
      host: snort-ids
```

ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Istio Destination Weight

- Use weight field under destination in Virtual Service to divide ingress traffic specified as percentage

- Two entirely different services
  - clover-server1
  - clover-server2

URLs to domain www.sdc.com

Match HTTP header user-agent 'chrome' to

20% to clover-server1

80% to clover-server2

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: directserverweight
spec:
  hosts:
    - "www.sdc.com"
  http:
    match:
    - headers:
        user-agent:
          exact: chrome
    route:
    - destination:
        port:
          number: 9180
        host: clover-server1
      weight: 20
    - destination:
        port:
          number: 9180
        host: clover-server2
      weight: 80
```

ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Istio Destination Weight for Service Versions

- Additionally use subset field to divide traffic among multiple versions of the same service
- DestinationRule resource defines subset labels (original http-lb deployment resource)
- Useful for A/B testing

URLs to domain www.sdc.com

Match HTTP header user-agent 'chrome' to

95% to http-lb (v1)

5% to http-lb (v2)

DestinationRule

Defines subset v1/ v2 labels

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: serviceversions
spec:
  hosts:
    "www.sdc.com"
  http:
  - match:
    - headers:
        user-agent:
          exact: chrome
    route:
    - destination:
        port:
          number: 9180
        host: http-lb
        subset: v1
      weight: 95
    - destination:
        port:
          number: 9180
        host: http-lb
        subset: v2
      weight: 5

apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: http lb destination
spec:
  host: http-lb
  subsets:
    name: v1
    labels:
      version: v1
  - name: v2
    labels:
      version: v2
```

ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Istio Fault Injection & Circuit Breaking

- Fault Injection
  - Inject faults to test the resiliency of your application
  - End-to-end failure recovery capability of the application as a whole

  - Delay: timing failures
    - Mimic network latency, or an overloaded upstream service

  - Abort: crash failures
    - mimic failures in upstream services (HTTP error codes)

- Circuit Breaking
  - Ejected from the load balancing pool when thresholds are exceeded
    - number of health check failures or number of conditions such as connection and request limits

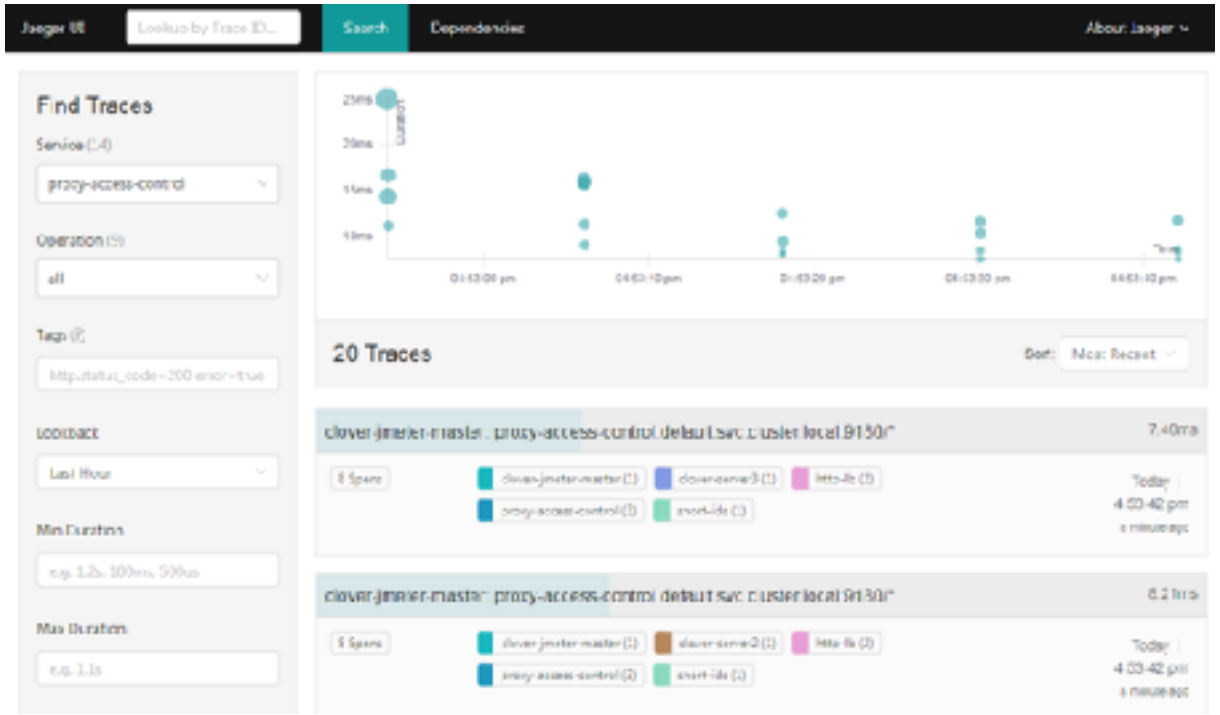
- Useful for LFN projects that are planning or using cascading REST services

ons EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Istio Mesh - Visibility Tools

- Jaeger: Tracing

- Prometheus: Monitoring



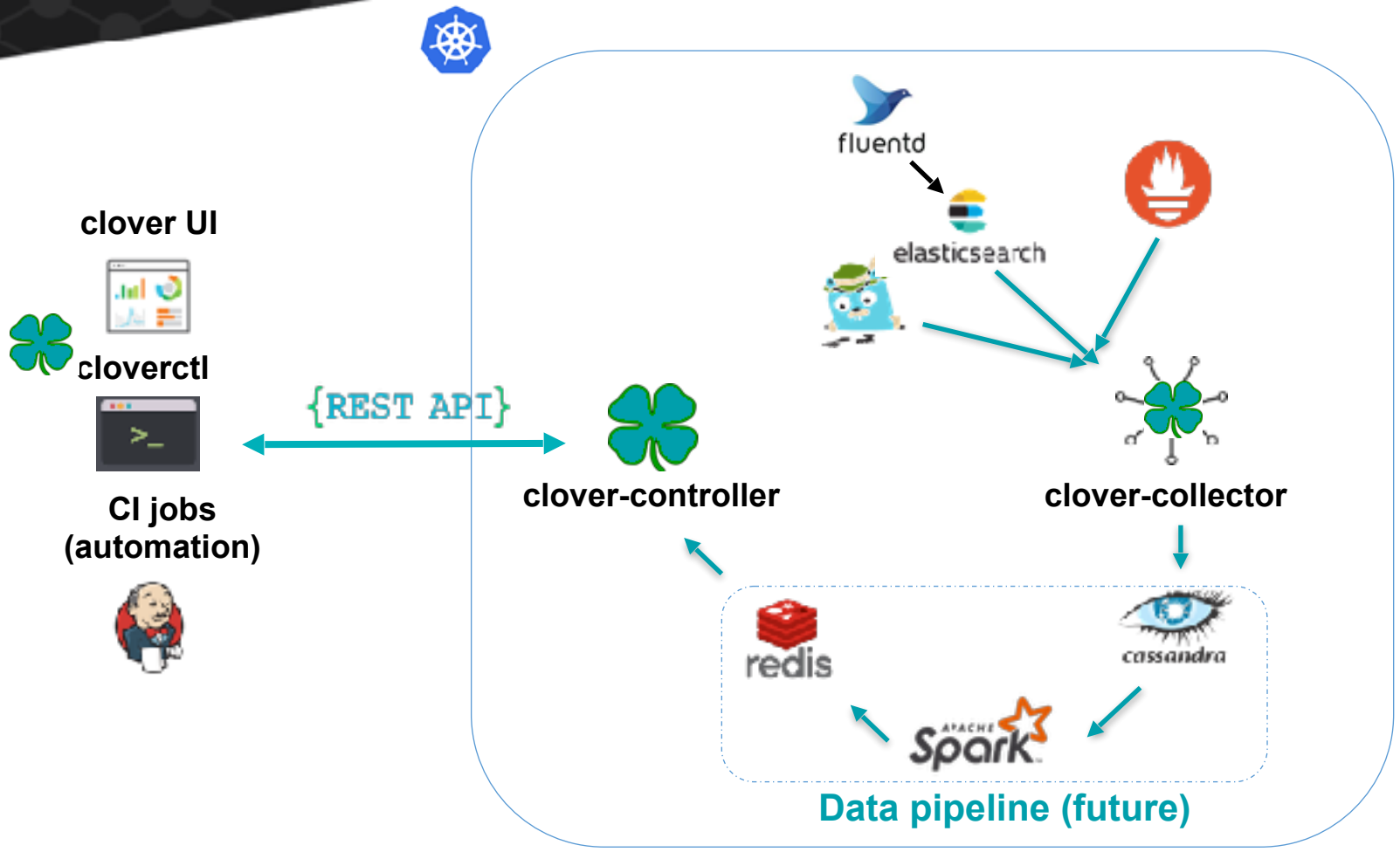
- Good raw data
  - Individual traces in Jaeger
  - Metrics list in Prometheus
  - Dashboards in Istio / Grafana
- But difficult to get insight of entire system (aggregate, top-level) and use analytics from data-sets

ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

September 25 - 27, 2018
Amsterdam, The Netherlands

Visibility/Observability Infrastructure Mesh/ Non-Mesh

Clover Visibility



- Analyzes data from CNCF observability tools to provide abstraction
  - Gathers data and analyzes using Spark
- 4 core components (clover-system)
  - clover-collector (within k8s)
  - clover-controller (within k8s)
  - cloverctl (external)
  - clover UI (external)
- User interacts with cloverctl or UI
  - CLI/UI use same REST API from clover-controller service
  - Chooses services to track
  - Outputs analyzed data to Redis

Clover Visibility Initialization (1-2)

- Install Istio
- Install clover-system components within k8s
- Expose clover-controller using LB or NodePort k8s service resource
- Gambia release will have CLI / script installation

```
$ cloverctl init visibility
$ cloverctl start visibility –f visibility.yaml
$ cloverctl clear visibility
```

- Use CLI to initialize visibility
  - Create traces, spans, metrics Cassandra schemas
- Start visibility
  - Collector begins gathering data from Jaeger, Prometheus
- Clear visibility
  - Truncates tables

Clover Visibility Initialization (2-2)

- Set sampling interval for collector
- Tracing/monitoring k8s DNS names
- Tracing/monitoring listening ports (Jaeger/Prometheus)

```
$ cloverctl start visibility –f visibility.yaml
```

visibility.yaml

```
sample_interval: "10"
t_host: tracing.istio-system
t_port: "80"
m_port: "9090"
m_host: prometheus.istio-system
```

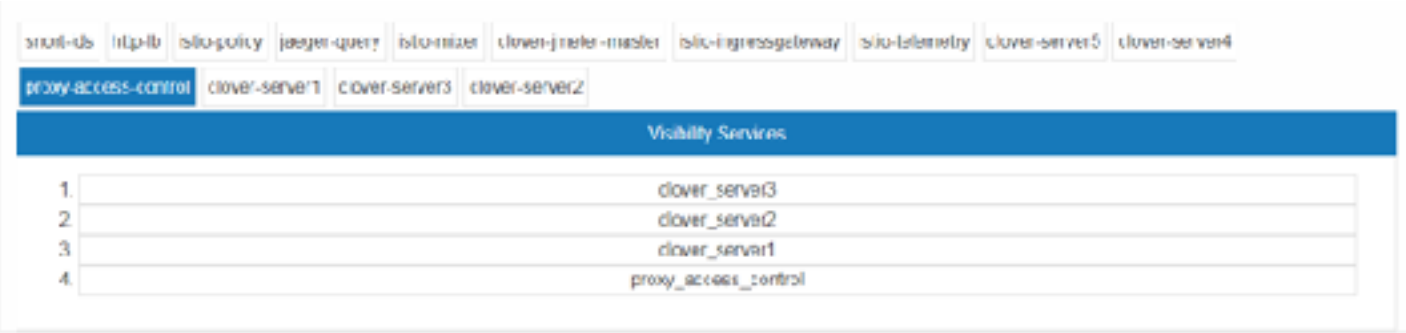
- Configure tracing services that visibility will analyze
- Configure metric prefixes/suffixes to analyze

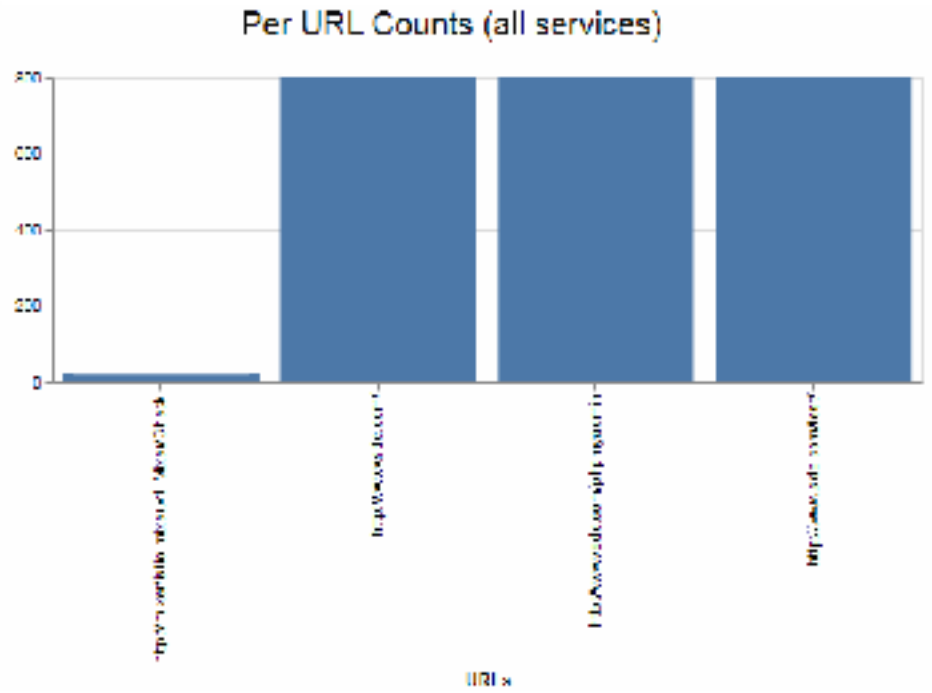
```
$ cloverctl set visibility –f metrics.yaml
```

metrics.yaml

```
services:
    - name: proxy_access_control
    - name: clover_server1
    - name: clover_server2
    - name: clover_server3
prefixes:
    - prefix: envoy_cluster_outbound_9180__
    - prefix: envoy_cluster_inbound_9180__
suffixes:
    - suffix: _default_svc_cluster_local_upstream_rq_2xx
      suffix: _default_svc_cluster_local_upstream_cx_active
```

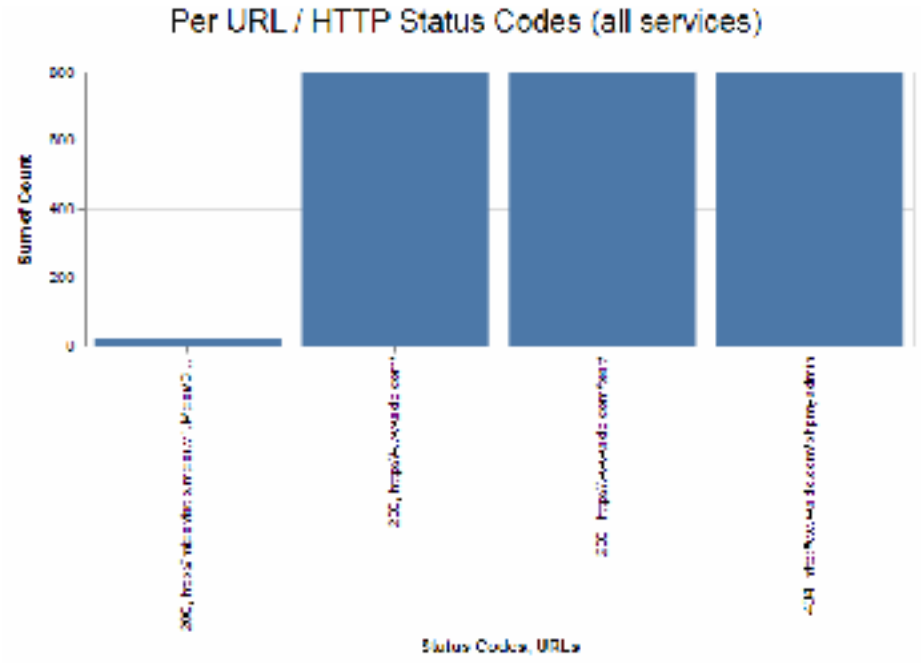
Clover Visibility Stats (1-3)

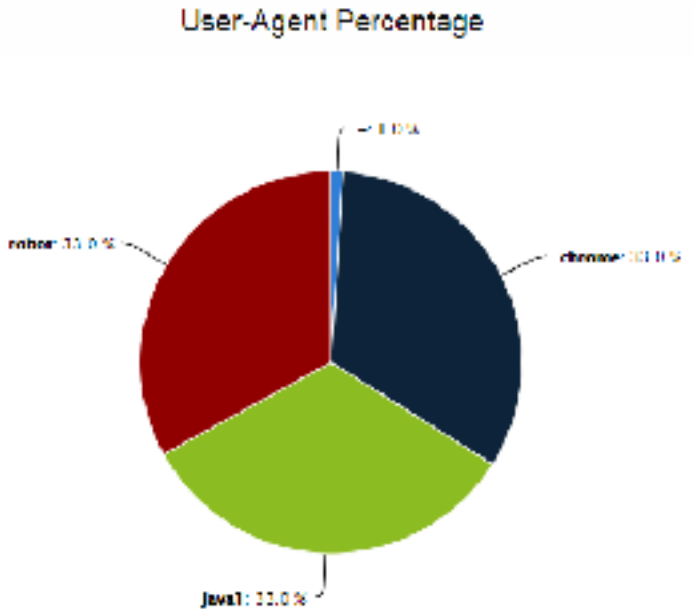


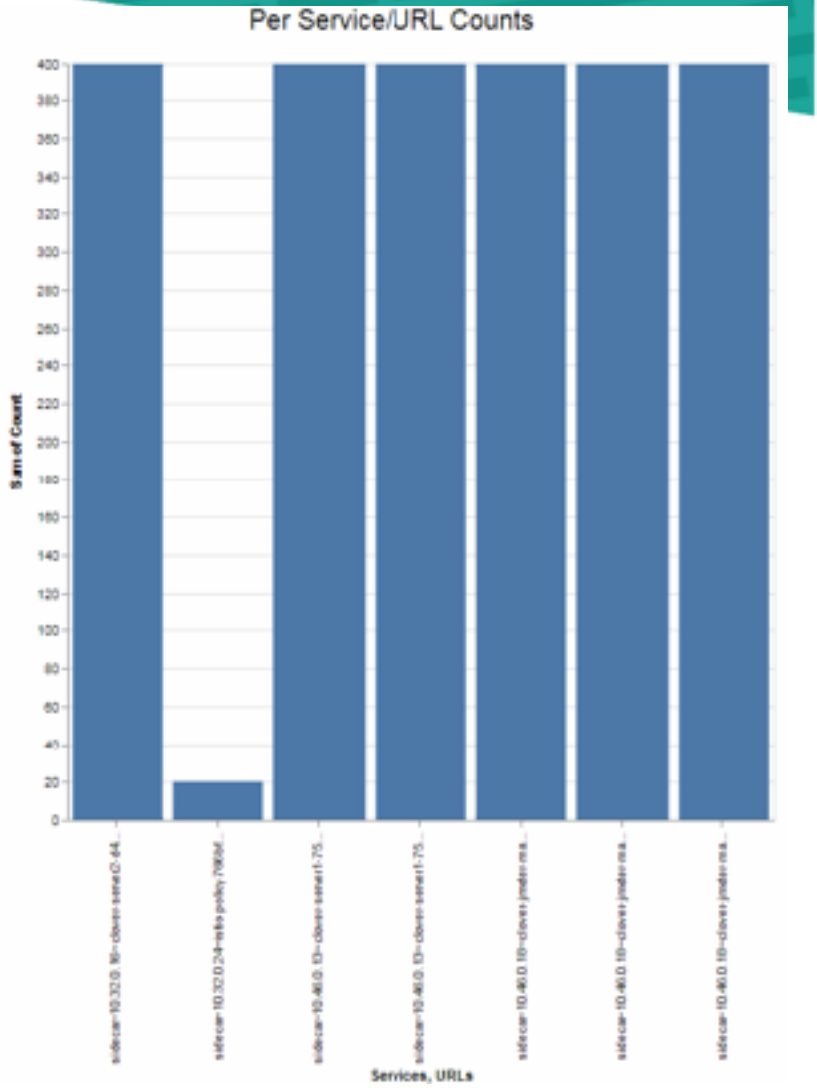


- Analyze trace data at aggregate level
  - Calculate average response time for various services
- Break down data in various ways
  - Per URL, Per Service/URL, more TBA in Gambia release

Clover Visibility Stats (2-3)







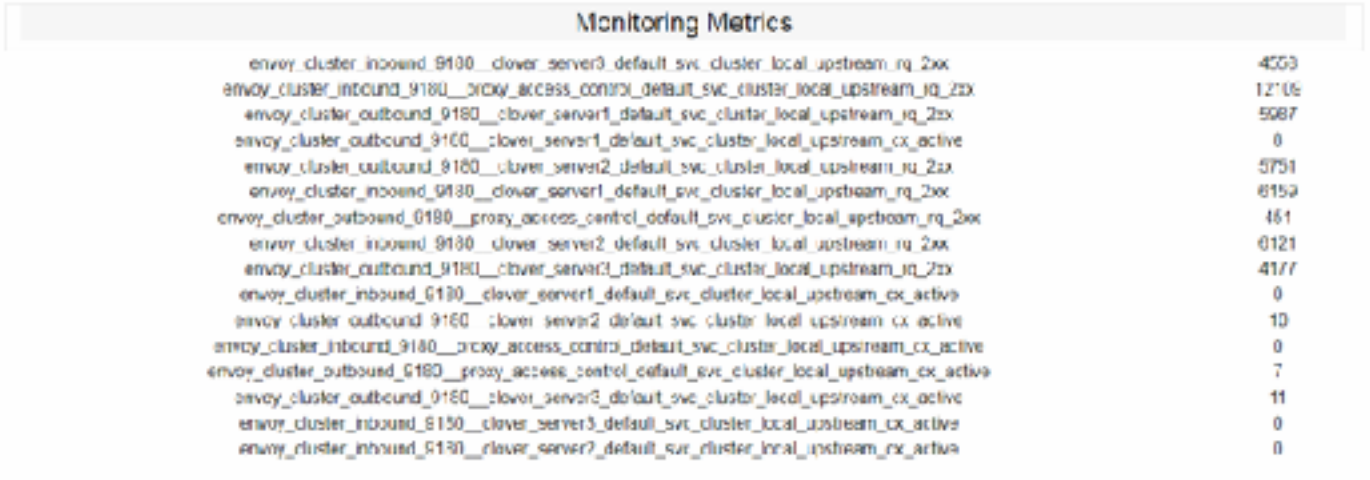
- Find issues with REST services such as service HTTP status codes being returned
- Validate service mesh traffic management policies such as request routing by user-agent (ex. mobile vs desktop)

ons EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Clover Visibility Stats (3-3)

- Characterize the composition of the traffic

- Output service request/response rates over time, lost requests, etc.



ons EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

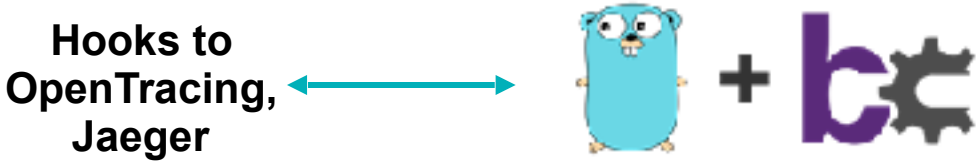
Clover Clovisor

Istio

- Large compute footprint
  - Istio - 13 Containers
  - Sidecar container per service
    - Latency overhead with long service chains
- Lacks visibility for:
  - L3 network
  - Other L4-7 content
- Lacks networking breadth for traffic management
  - Doesn't support wide set of protocols, tunneling, encapsulation

Clovisor

Hooks to OpenTracing, Jaeger



- Leverages eBPF
- Installed on k8s cluster nodes

Clovisor: Network Tracing… the Cloud Native Way

ons EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

1. Cloud Native:

a) Cloud Provider Independent
- Bare-metal servers, GKE, EKS…etc

b) CNI Plugin Agnostic
- All CNI plugins should work unless such plugin does …. bypass

c) CPU Architecture Independent
- Any architecture supported by Linux (x86, ARM…etc), code (kernel versions 4.14 and 4.15 currently)

2. Implemented with Cloud Native Design Methodologies:

a) Config Decoupled from Compute
- Config store in backing store or through environment variables

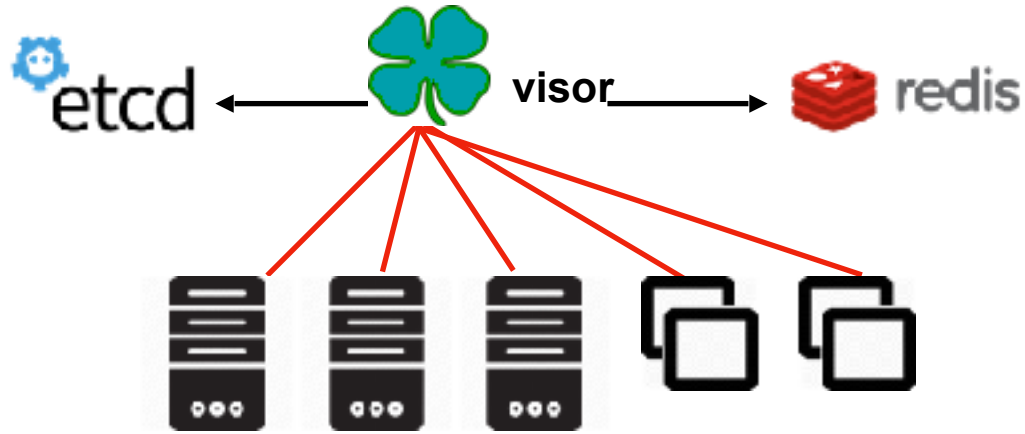
b) Relatively Stateless
- TCP connection/session tracking only dynamic states

c) Scale-out Architecture
- Pod monitoring partitioning via election from datastore
- DaemonSet —- linearly scale on each node in cluster

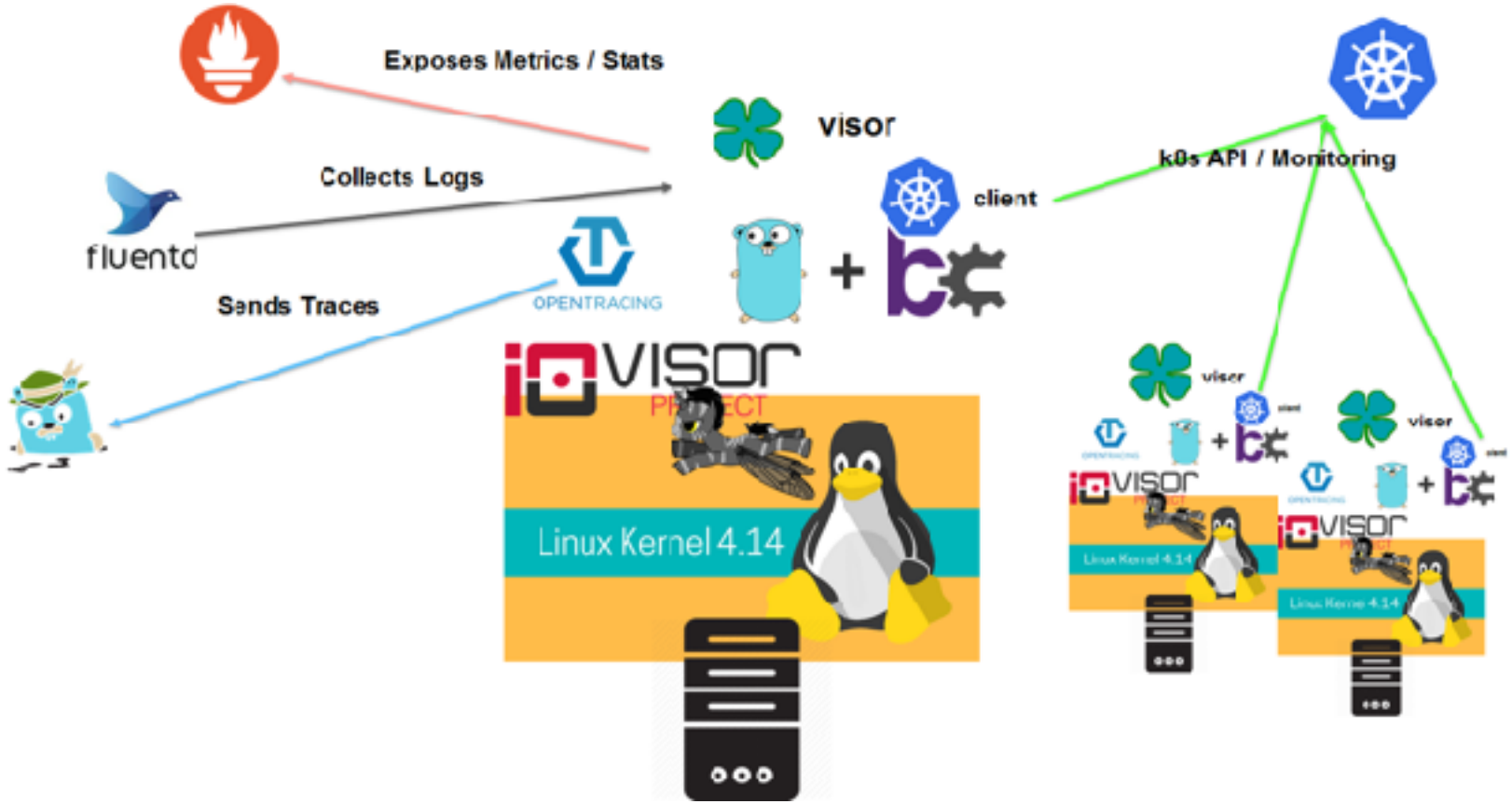
3. In-depth Integration with Cloud Native Ecosystem Projects:

a) Built-in Kubernetes Client
- Monitoring k8s pod states

b) Integrate with CNCF Collector Projects
- OpenTracing to Jaeger, metrics to Prometheus



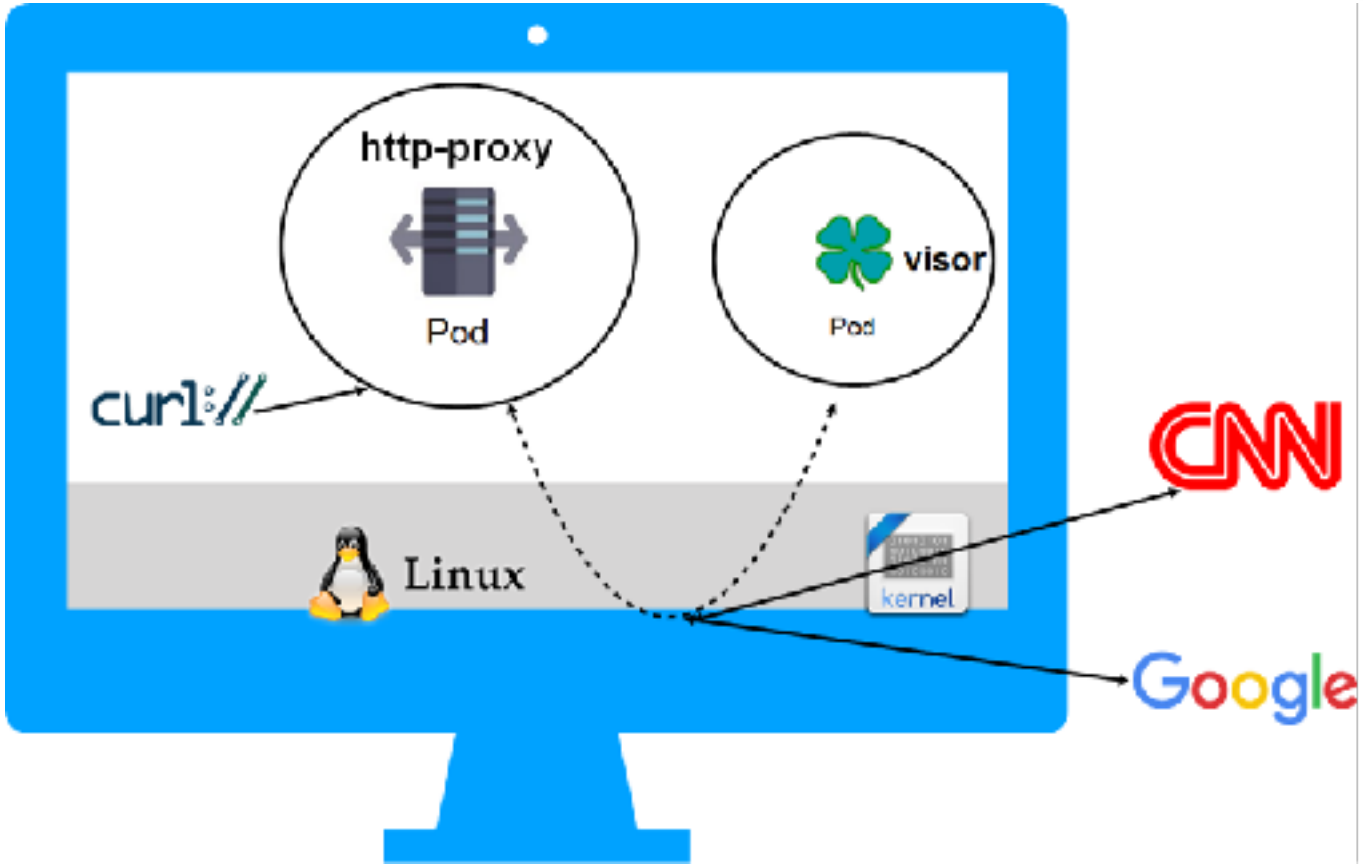
Clovisor Architecture



- Lightweight, low latency network tracing module
- Utilizes IOVisor (bcc, gobpf) with eBPF to insert bytecode in Linux kernel to examine packets from both ingress / egress direction of a k8s pod
- In cluster client to automate process of monitoring and service port / protocol info
- Stream trace / stats / metrics / logs to respective tracer / collector modules

Clovisor Demo

- Configure monitoring labels (namespace:label-key:label-value)
- In this case: “default” namespace, key: “app”, value: “proxy”
- Start Clovisor (on node, verify if the tc filter is created for device)
- curl www.cnn.com with http-proxy service port (3456)
- curl www.google.com with http-proxy service port (3456)
- Check Jaeger UI to verify traces written/sent



ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Visibility Use-Cases

- Easily pinpoint issues with individual services
- Integrate into CI to determine success/failure of jobs
  - CI used to determine CD deployment pipeline
- Monitor infrastructure in operations to determine system health
- Characterize the composition of traffic for content delivery or security
- Leverage to automate orchestration or zero-tech provisioning

ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

September 25 - 27, 2018
Amsterdam, The Netherlands

Summary

ons EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Key Take-Aways

- Service meshes allow microservices to be delivered more rapidly with integrated traffic management and visibility hooks
  - Visibility helps developers pinpoint issues and operators manage infrastructure
  - Built-in traffic management allows for microservice CI/validation and deployment strategies
  - Ideal for control-plane and REST services
- Service mesh distributed tracing/monitoring collects data efficiently but lacks an aggregate view of infrastructure/services
  - LFN projects such as Clover can provide high-level analytics for developers and operators
- Service mesh overhead/footprint and lack of networking breadth (both for visibility & routing/security)
  - Clovisor is a promising approach to fill gaps and add additional networking extensions

ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Clover Project Info

- **Project Wiki**
  - https://wiki.opnfv.org/pages/viewpage.action?spaceKey=CLOV&title=Clover+Home
- **Slack Channel**
  - #clover-project
- **Github Repo**
  - https://github.com/opnfv/clover

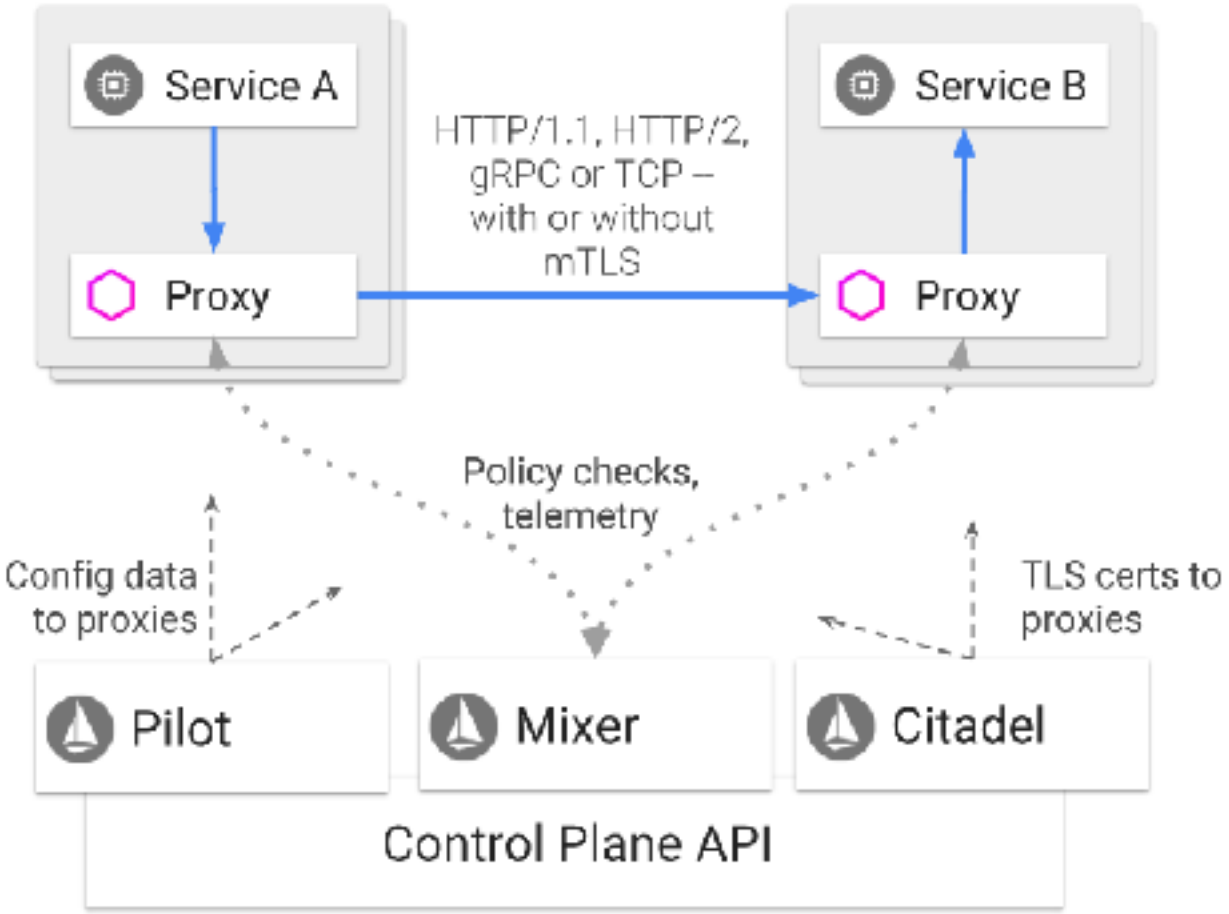
ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

September 25 - 27, 2018
Amsterdam, The Netherlands

Appendix

ons EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Istio Control-Plane Components



ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Istio - Control Egress Traffic

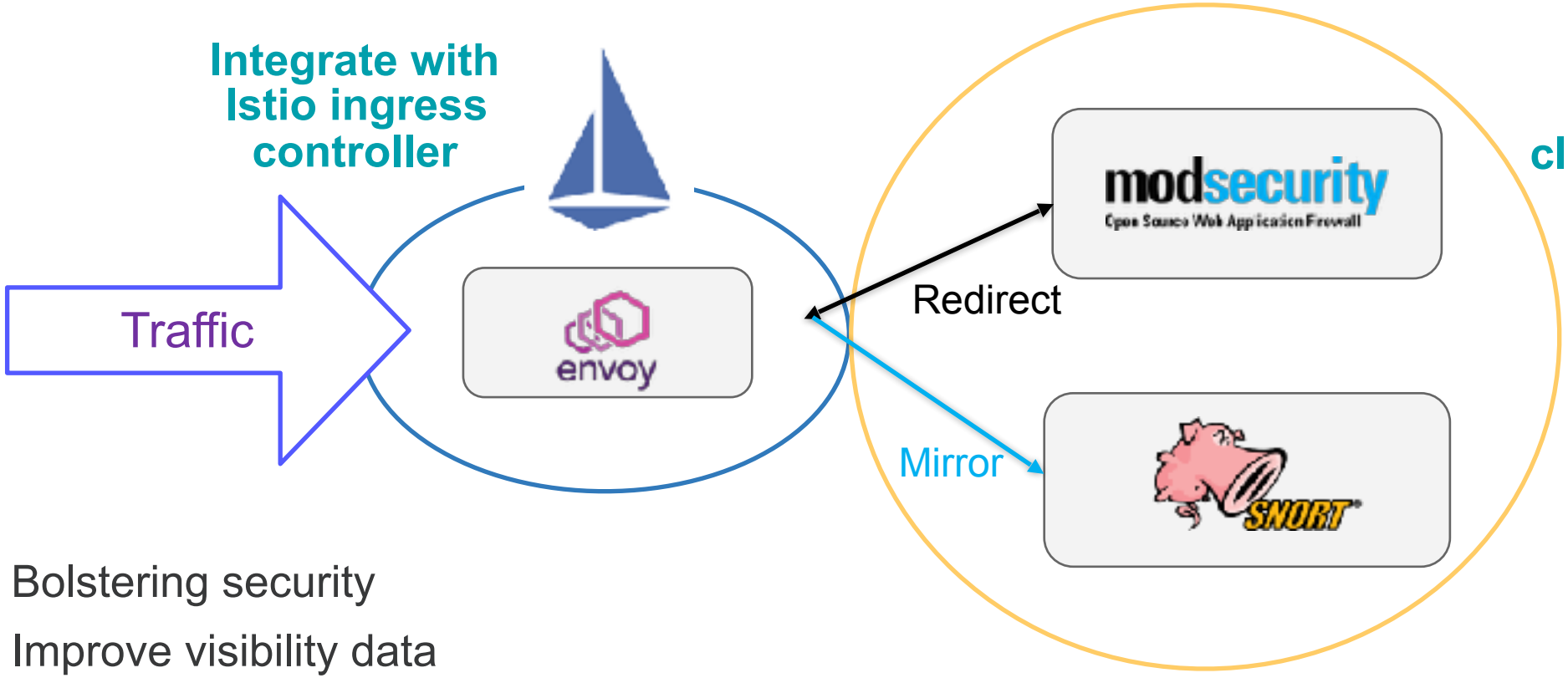
- Default Istio-enabled services are unable to access URLs outside of the cluster
  - Pods use iptables to transparently redirect all outbound traffic to the sidecar proxy, which only handles intra-cluster destination

```
apiVersion: networking.istio.io/v1alpha3
kind: ServiceEntry
metadata:
  name: sdc-ext
spec:
  hosts:
  - www.sdc.com
  ports:
  - number: 80
    name: http
    protocol: HTTP
  resolution: DNS
  location: MESH_EXTERNAL
```

Send traffic outside of mesh to ‘www.sdc.com’

(assuming this is a valid domain in DNS)

Augmenting Mesh/Kubernetes Ingress



clover-gateway services

- New in Clover Gambia release

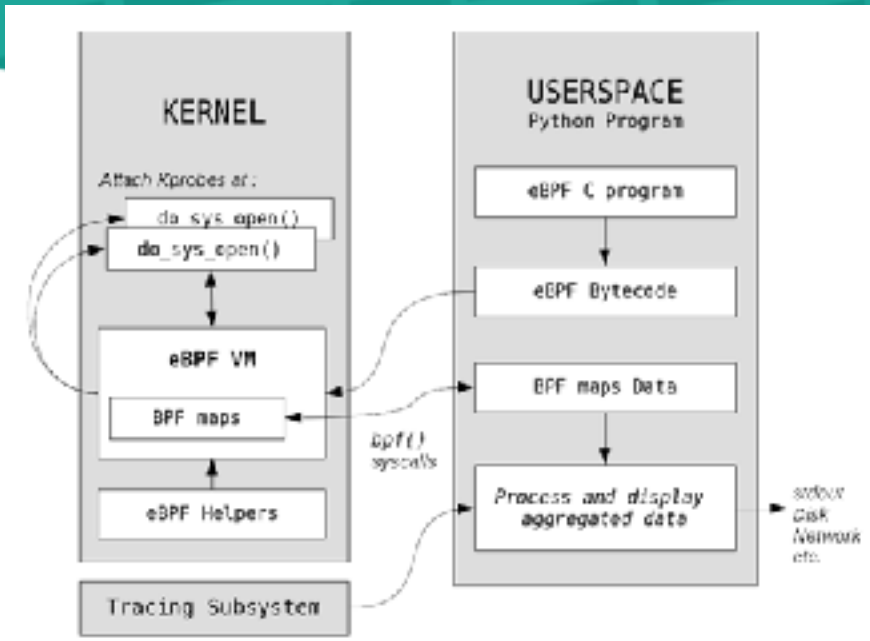
- Bolstering security
- Improve visibility data

ons EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

IOVisor & eBPF

- eBPF
  - Inject bytecodes to kernel trace points / probes
    - Event driven model
  - Networking: tc
    - Utilizes Linux tc (traffic control) to inject bytecode on ingress and egress direction of a network interface
  - Verifier / JIT (just-in-time compiler)
    - Verifier ensures bytecode does NOT crash kernel

Enhanced BPF is in Linux



- IOVisor bcc:
  - Ease of eBPF Development
    - Helper functions, kernel API wrappers…etc
  - Dynamic Validation and Compilation
    - Userspace eBPF code written in 'C' is dynamically verified (static analysis) and compiled
  - gobpf
    - Golang interface for userspace code —- more performant than Python