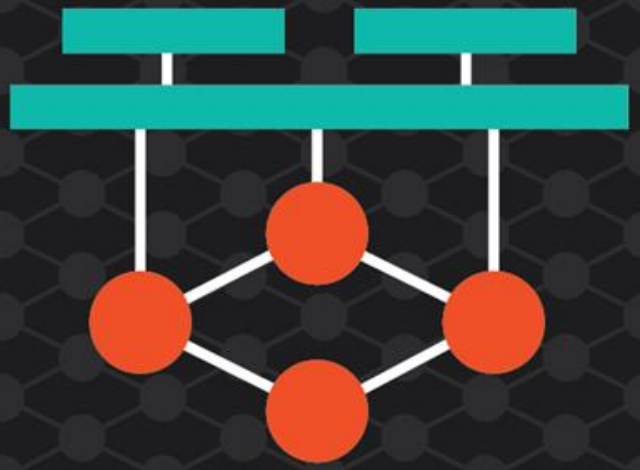
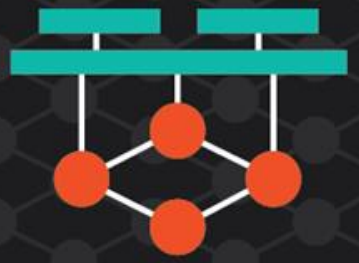
September 25 - 27, 2018
Amsterdam, The Netherlands



ons

EUROPE

OPEN NETWORKING //
Integrate, Automate, Accelerate



ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

September 25 - 27, 2018
Amsterdam, The Netherlands

Delivering Network Services using Cloud Native Methodology

Eddie Arrage
Wenjing Chu
Futurewei Technologies Inc.

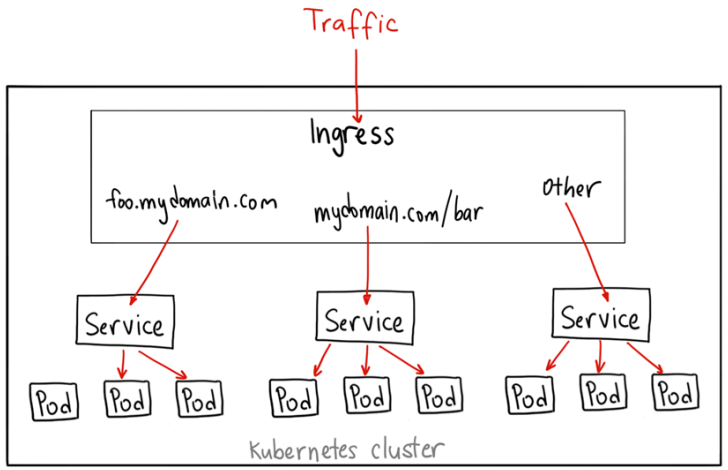
ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Agenda

- **Cloud Native Concepts/Methodology (10 min)**
- **Segmenting & Instrumenting Microservices (15 min)**
  - Instrumentation
  - Example in Clover
- **Managing & Controlling Traffic (20 min)**
  - Service Meshes / Istio
  - Mesh Visibility Tools
  - Mesh Traffic Management
  - Augmenting Meshes
- **Debugging & Monitoring (25 min)**
  - Visibility/Observability Infrastructure
  - Introduction to Clovisor
- **Integrating & Validating (10 min)**
  - L7 Jmeter validation
  - Jenkins integration
- **Deploying & Managing Services, Infrastructure (15min)**
  - Introduction to Spinnaker – CI/CD

Cloud Native

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



– Microservice oriented

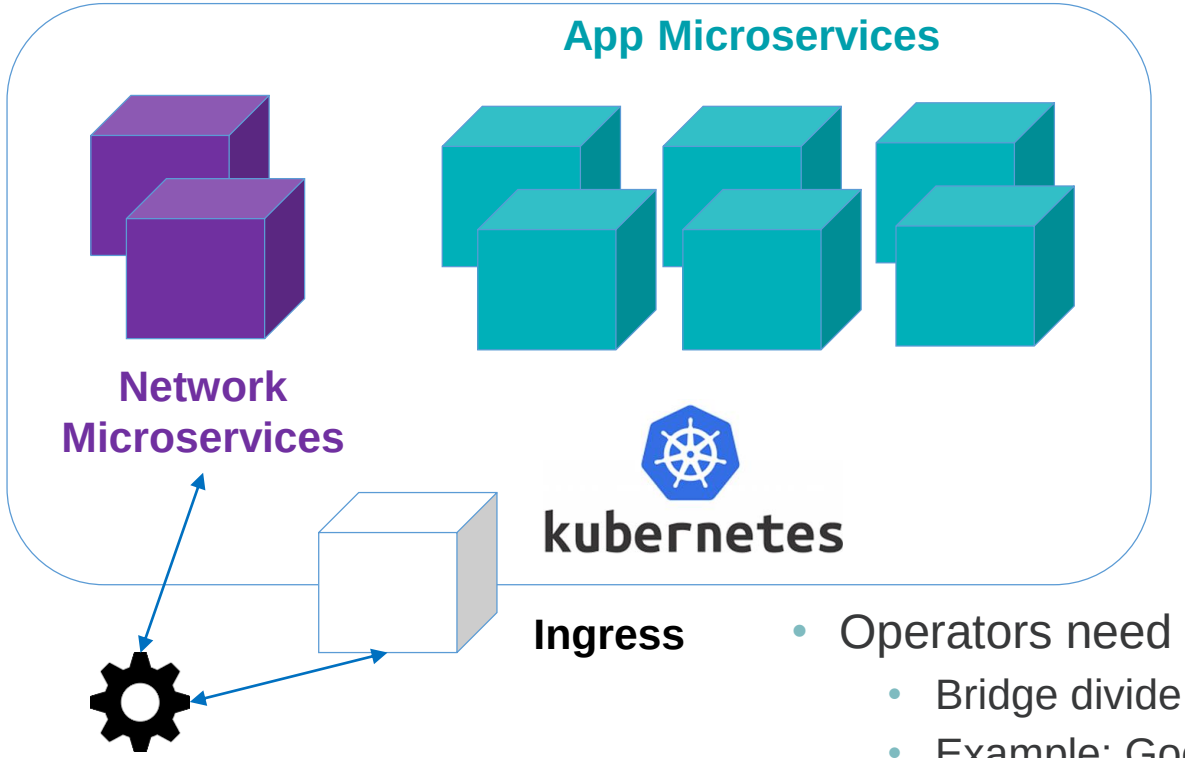
kubernetes

– Dynamically managed (Kubernetes)

docker docker docker

– Containerized

Application / Network Co-Existence

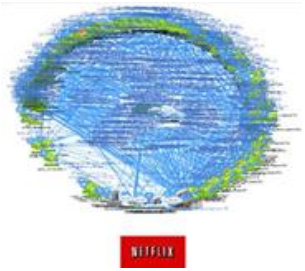


- App evolution to microservices
  - Develop, debug and manage individual Lego blocks
- App developers want abstracted network
  - Usually to support web/REST oriented services
    - Example in LFN: many ONAP services have REST interface
    - Ideal for control-plane services
  - Network management model needs to fit paradigm
- Operators need to manage network services with cloud-native constructs
  - Bridge divide between built-in networking (Kubernetes, service meshes) and apps
  - Example: Google Istio as a service cloud offering
- Support traffic management for CI/CD precepts: canary, blue/green, etc.

Cloud Native / Microservice Challenges

- Microservice sprawl
  - Debug difficult without tools for visibility and traceability of entire system

ONAP
OPEN NETWORK AUTOMATION PLATFORM



150+ containerized services

- Validation difficult as developers need to test system but might only own one service
  - Integrated testing and ease of system deployment

- Currently CI/CD pipeline in most LFN projects largely stops at CI level
  - Need to manage deployment pipelines
  - Support traffic management for cloud native

- Traditional operators need to consider how to offer compelling cloud services
  - Control traffic in/out of containerized environments
  - Network components for configurable ingress with security

Cloud Native Methodology

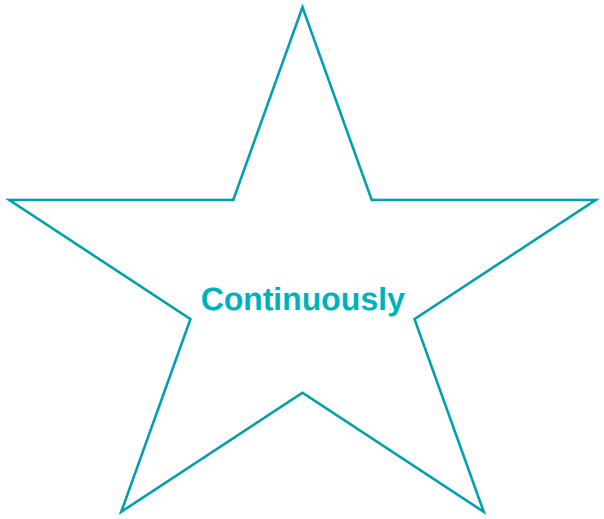
gRPC { REST }

Segmenting & Instrumenting Microservices

Managing & Controlling Traffic

modsecurity
Open Source Web Application Firewall

Deploying & Managing Services, Infrastructure

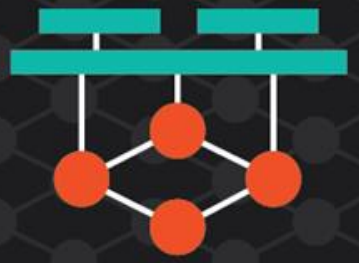


OPNFV

APACHE JMeter™

Integrating & Validating

Debugging & Monitoring

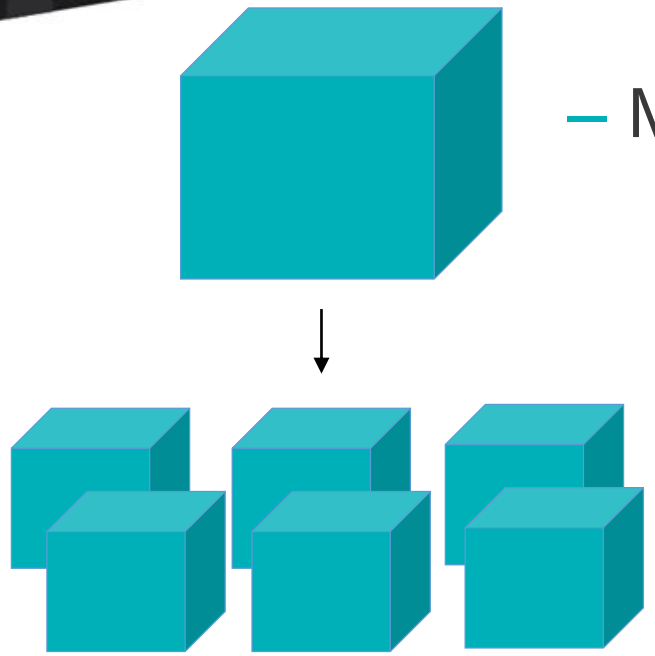


ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

September 25 - 27, 2018
Amsterdam, The Netherlands

Segmenting & Instrumenting Microservices

Microservices



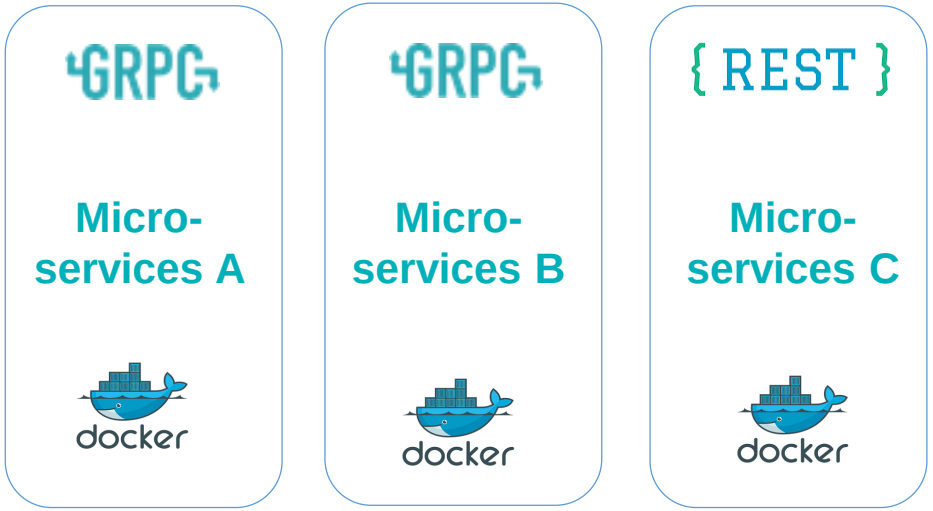
– Monolithic App

– 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

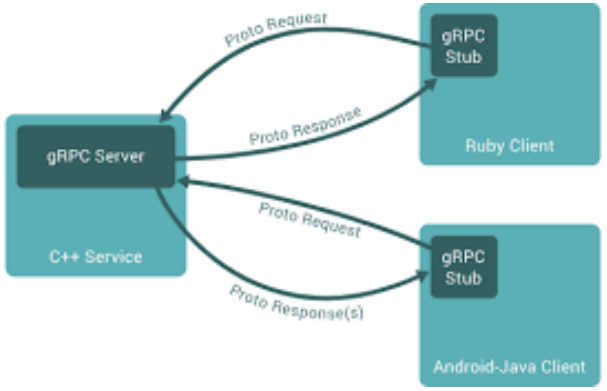
Microservice Instrumentation

- ConfigMaps
  - Manage/inject app configuration
  - Kubernetes resource
  - Keep containers agnostic

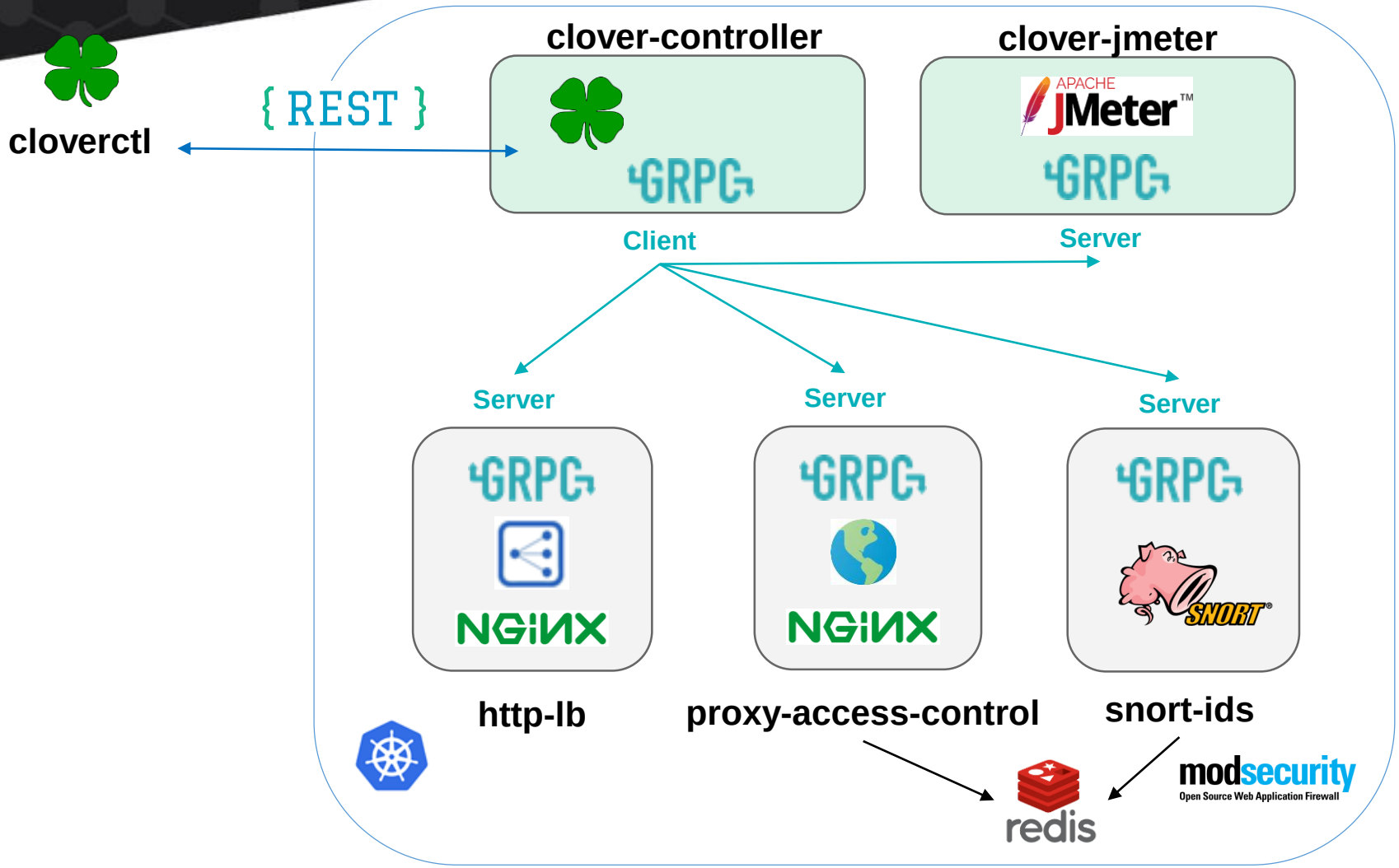


- Shared Data Stores
  - Exchange network data, state management

- gRPC
  - Open-source RPC framework
  - Client/server
  - Bindings for most languages
  - Frequent configuration



Microservice Clover Example



- **Clover-System Tools**
  - cloverctl - CLI interface
  - clover-controller - in-cluster message routing and UI dashboard
  - clover-jmeter – L4-7 client-emulation for CI/CD, validation
- **Sample Network Services**
  - Security: IDS, WAF
  - L4-7: proxy, load balancer…
  - Combine in various CNFs
  - Employ Linux services
  - Implement gRPC server for instrumentation
  - Redis data-store to share packet, security event data

ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

gRPC Demo

https://grpc.io

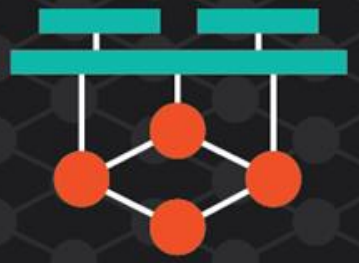
```
$ python -m pip install grpcio protobuf
```

Install for Python

```
$ python -m grpc_tools.protoc -I./ --python_out=. --grpc_python_out=. snort.proto
```

Generate gRPC code

```
syntax = "proto3";
package snort;

service Controller {

  rpc AddRules (AddRule) returns (SnortReply) {}
  rpc StartSnort (ControlSnort) returns (SnortReply) {}
  rpc StopSnort (ControlSnort) returns (SnortReply) {}
}

message ControlSnort {
  string pid = 1;
}

message AddRule {
  string protocol = 1;
  string dest_port = 2;
  string dest_ip = 3;
  string src_port = 4;
  string src_ip = 5;
  string msg = 6;
  string content = 7;
  string sid = 8;
  string rev = 9;
}

message SnortReply {
  string message = 1;
}
```



ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

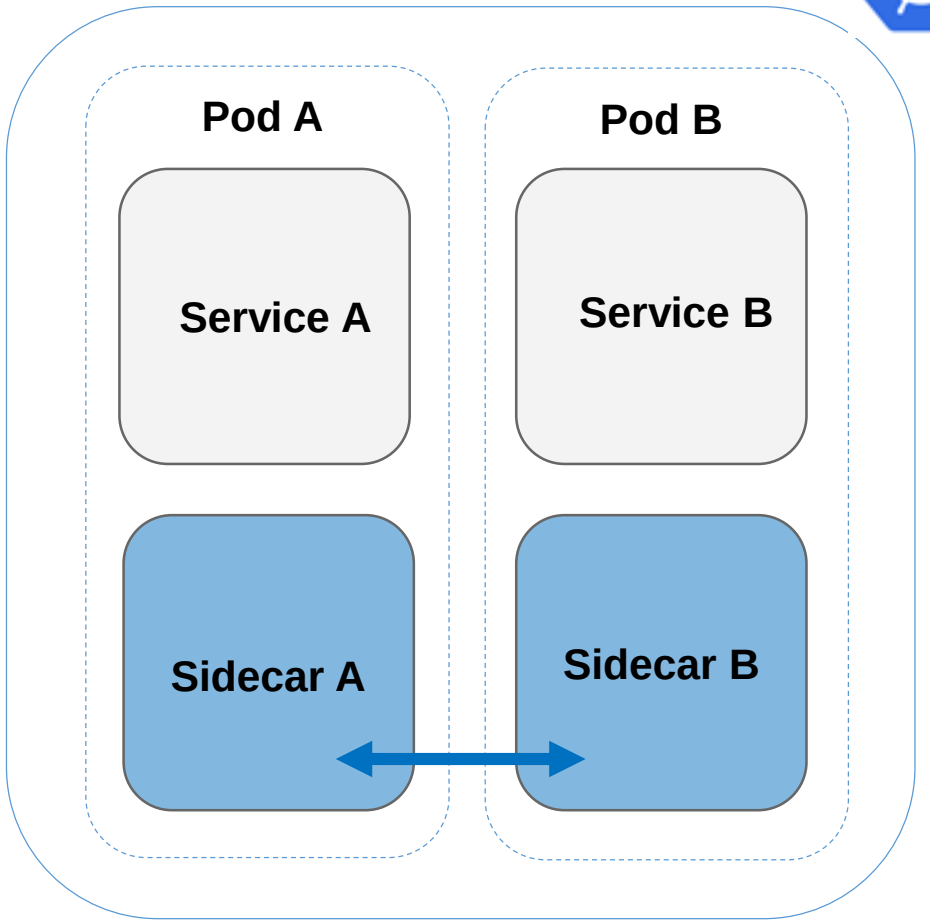
September 25 - 27, 2018
Amsterdam, The Netherlands

Managing & Controlling Traffic

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

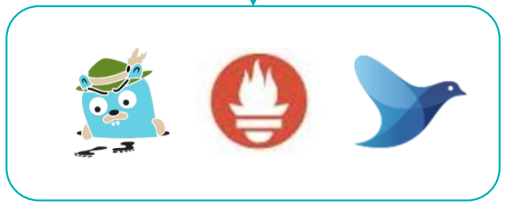
– Allows for more advanced routing than native k8s networking

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
```

- automatic sidecar (namespace) sidecar injection

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

- Manual sidecar injection

Setup

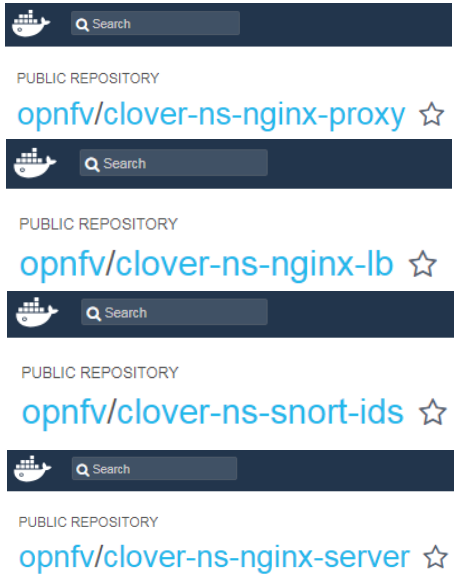
```
$ docker pull opnfv/clover:latest
$ sudo docker run --rm \
   -v ~/.kube/config:/root/.kube/config \
   opnfv/clover \
   /bin/bash -c '/home/opnfv/repos/clover/samples/scenarios/deploy.sh'
```

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: 50054
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: 50054 (alerts)
Servers	clover-server1 clover-server2 clover-server3 clover-server4 clover-server5	clover-ns-nginx-server	HTTP: 9180 GRPC: 50054



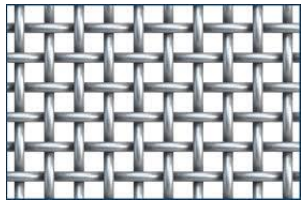
OPNFV Docker Hub Images

- 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)

Traffic within Mesh

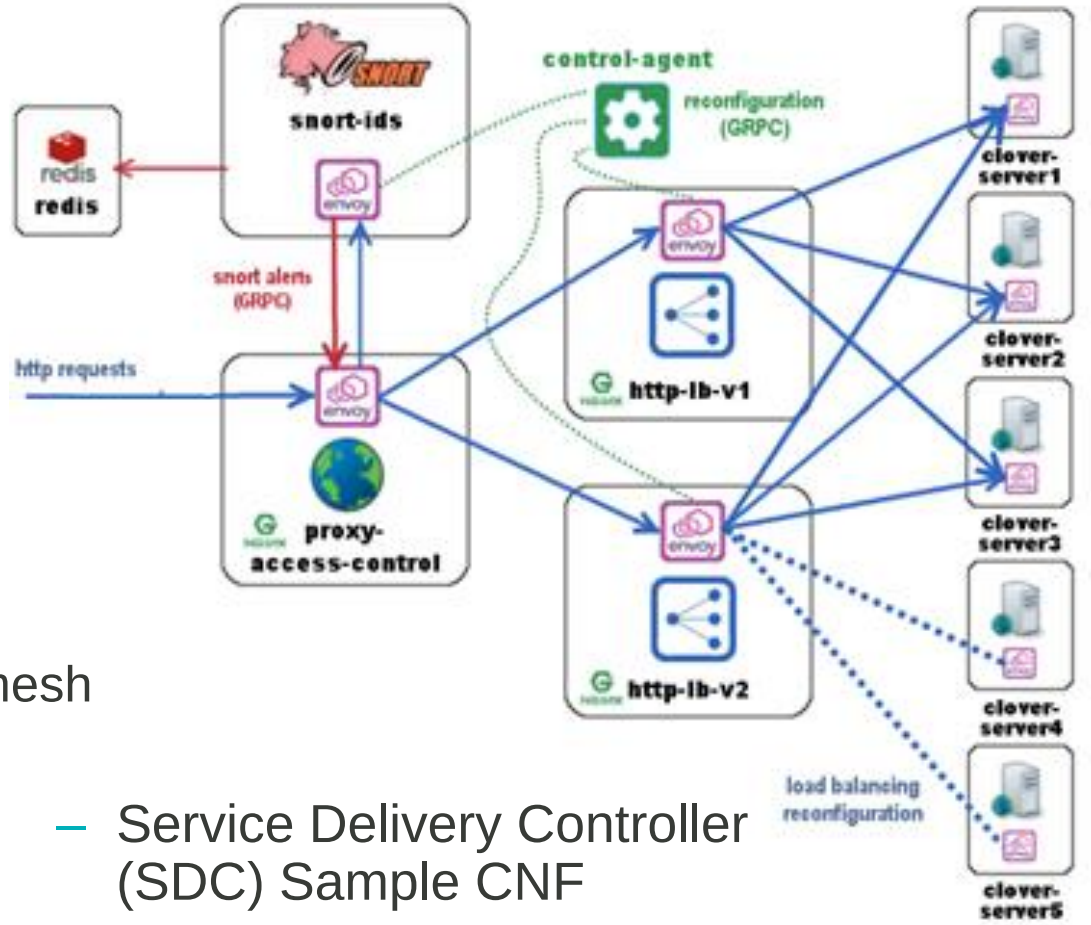
k8s



Emulated Clients

APACHE JMeter™

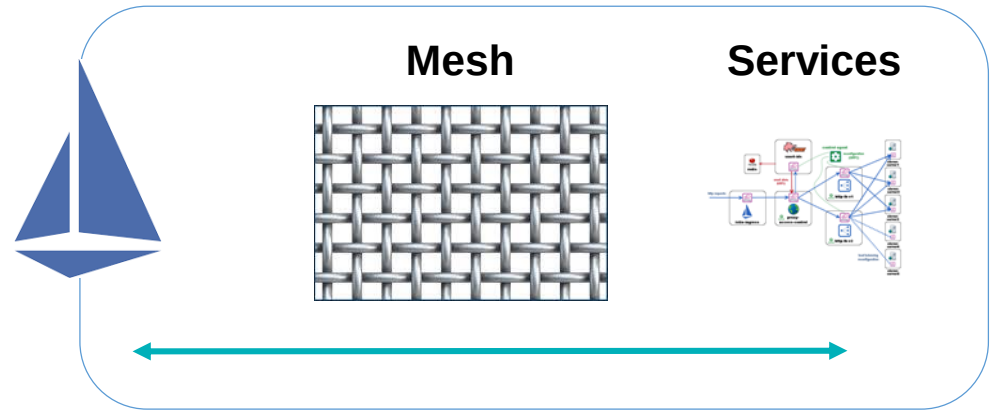
clover-jmeter-master



- 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:
    - "*"
---
```

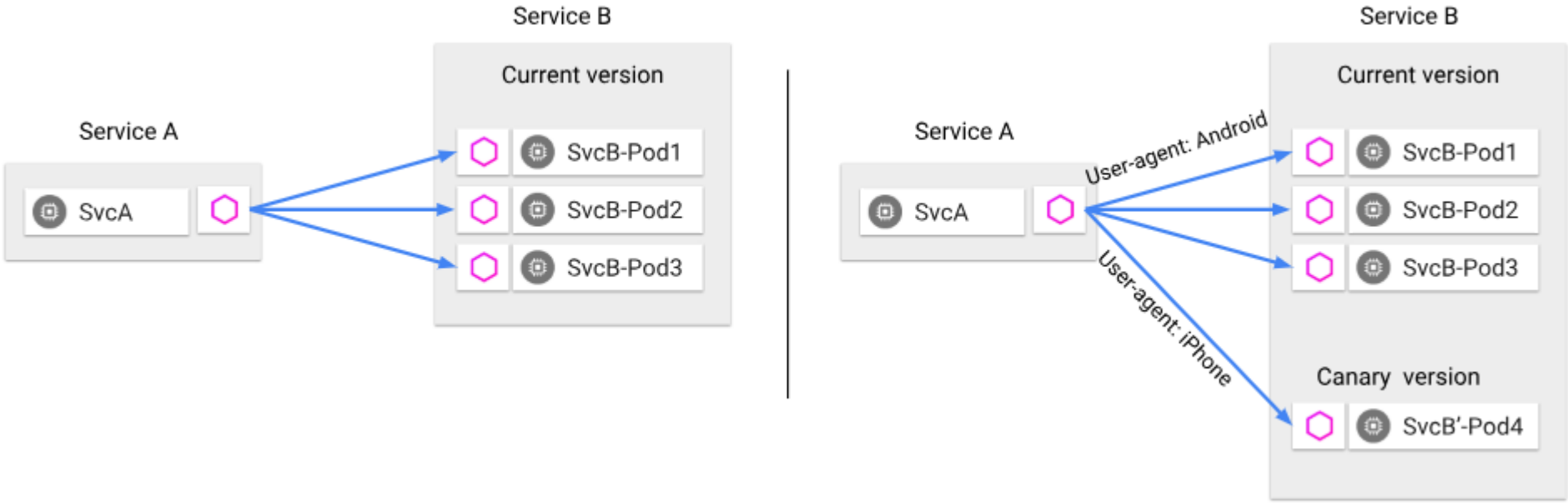
- LB at the edge of mesh receiving incoming/outgoing connections

```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
- Support CI/CD precepts with canary versions

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

URLs to domain
www.sdc.com

Match URI prefix
‘/test’ to
clover-server2

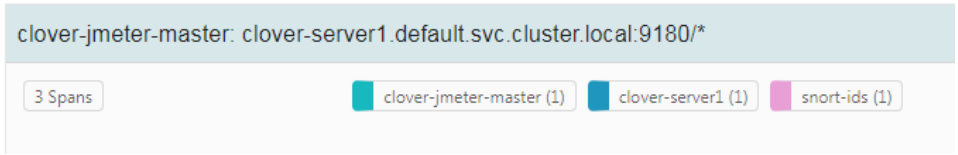
Match HTTP header
user-agent
‘chrome’ to
clover-server3

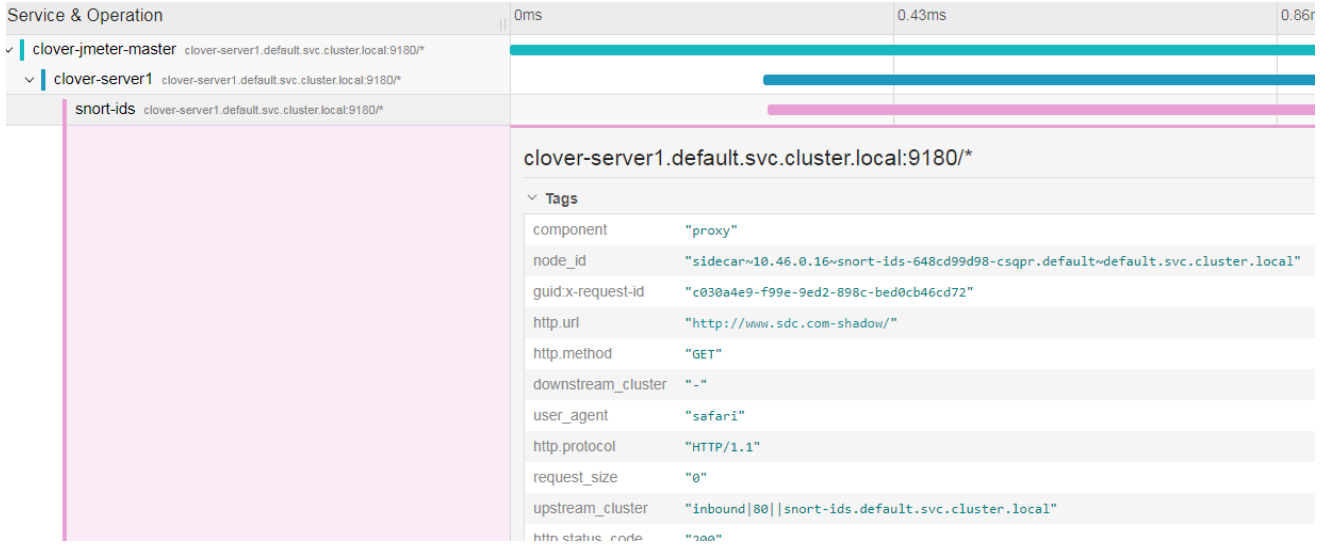
Everything else to
clover-server1

```
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
```

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

```yaml
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
    mirror:
      host: snort-ids
```

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 - 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

```yaml
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)

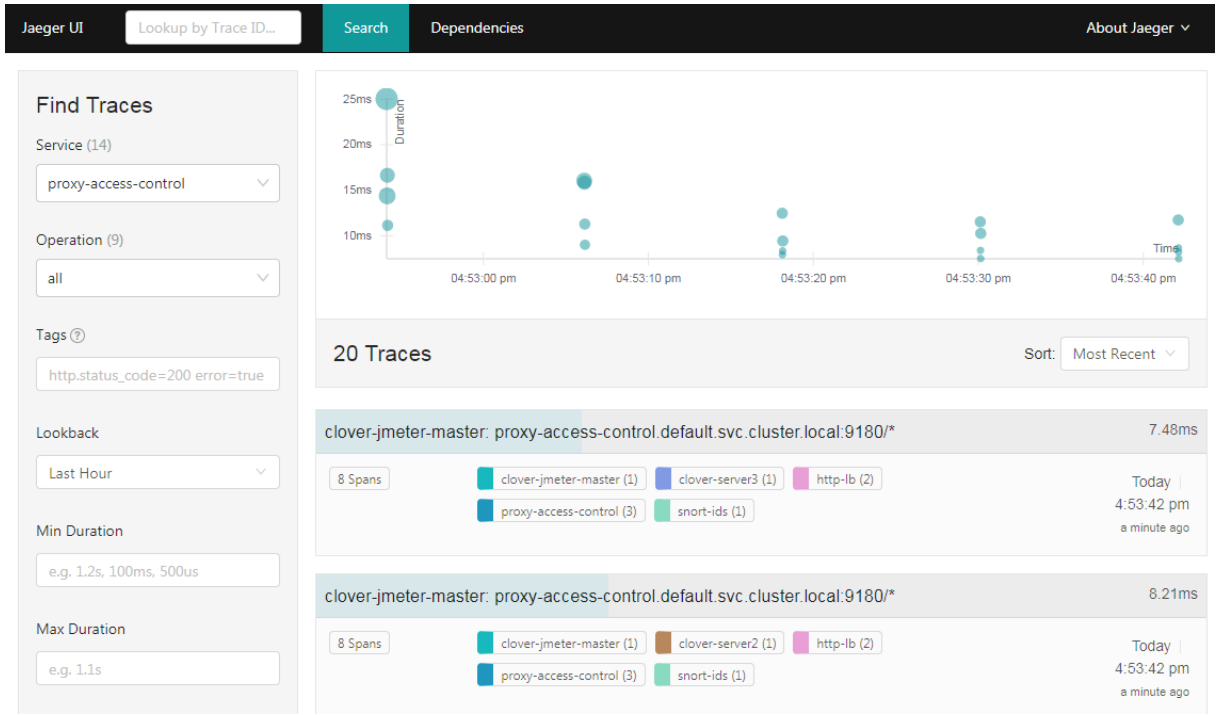
ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Istio Mesh - Visibility Tools

CLOUD NATIVE
COMPUTING FOUNDATION

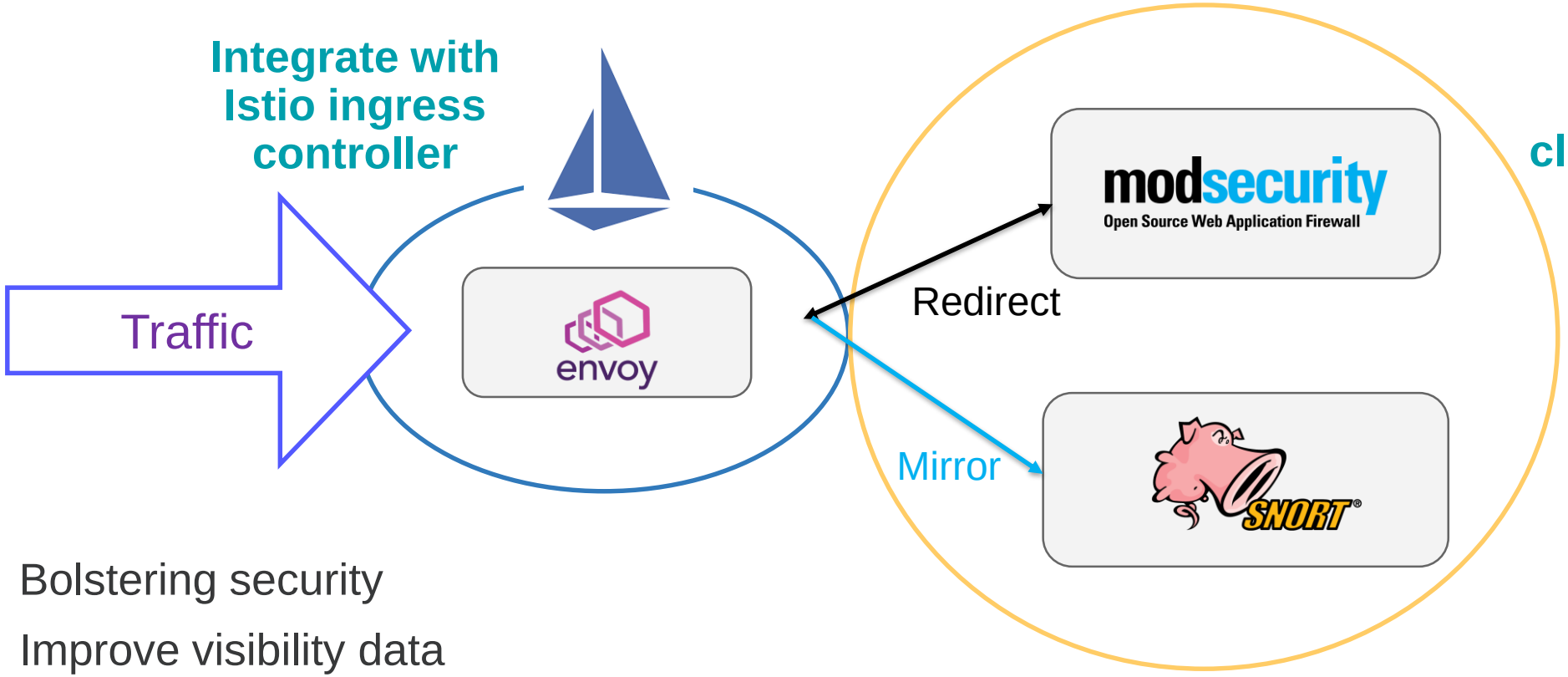
- Jaeger: Tracing

- Prometheus: Monitoring



- Good raw data
  - Individual traces in Jaeger
  - Metrics list in Prometheus
- But difficult to get insight of entire system (aggregate, top-level)

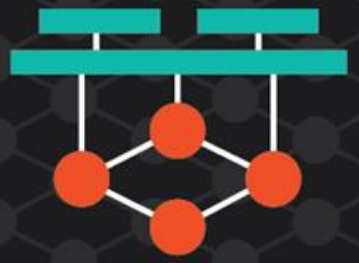
Augmenting Mesh/Kubernetes Ingress



clover-gateway services

- New in Clover Gambia release

- Bolstering security
- Improve visibility data

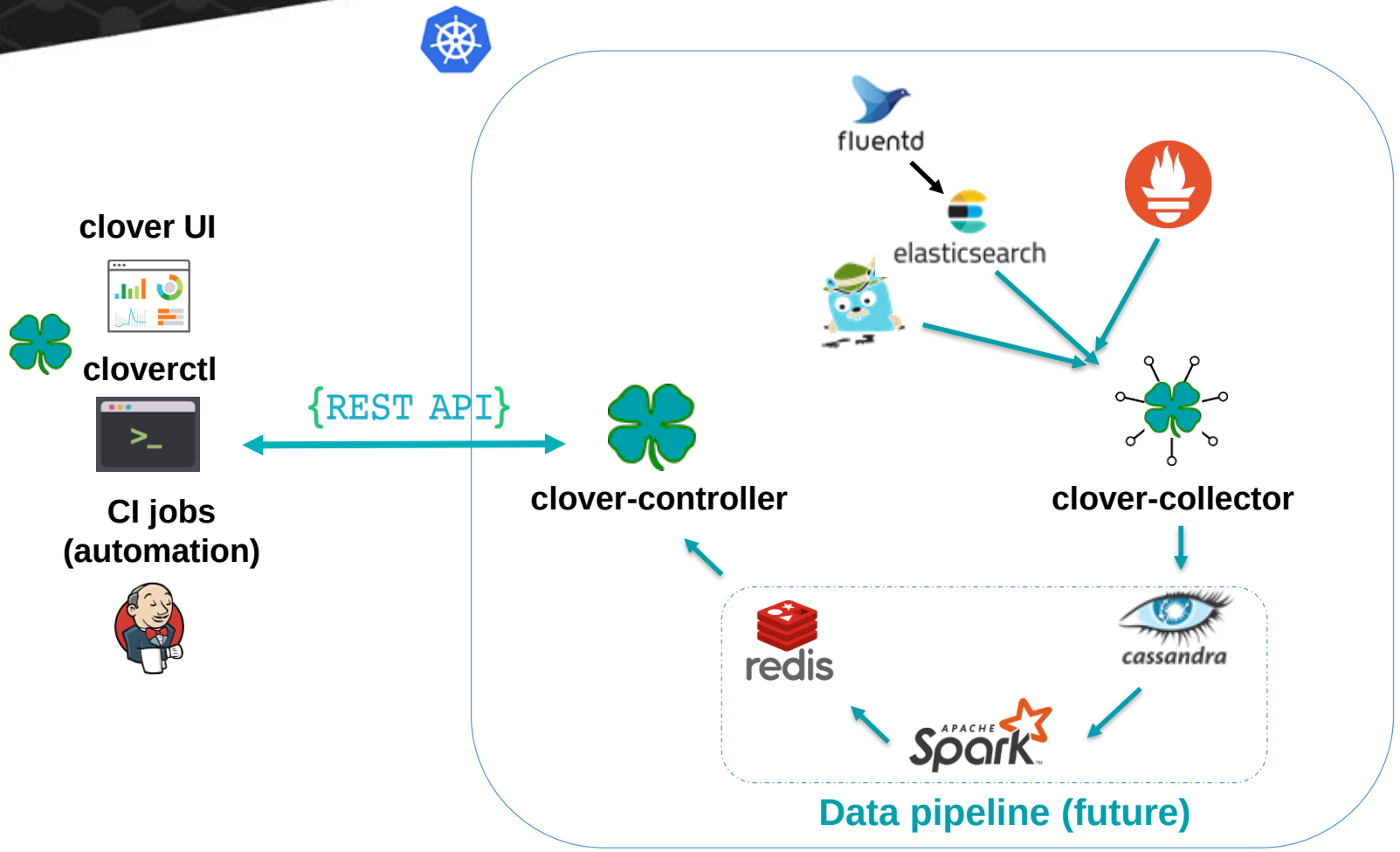


ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

September 25 - 27, 2018
Amsterdam, The Netherlands

Debugging & Monitoring

Clover Visibility



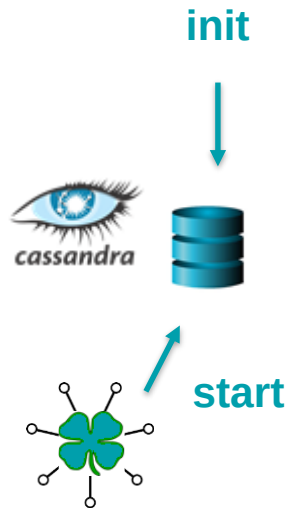
- Analyzes data from CNCF observability tools to provide abstraction
  - Gathers data and analyzes using Spark
- 4 core components (clover-system)
  - clover-collector (within k8s)
  - clover-controller (within k8s)
  - cloverctl (external)
  - clover UI (external)
- User interacts with cloverctl or 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
```

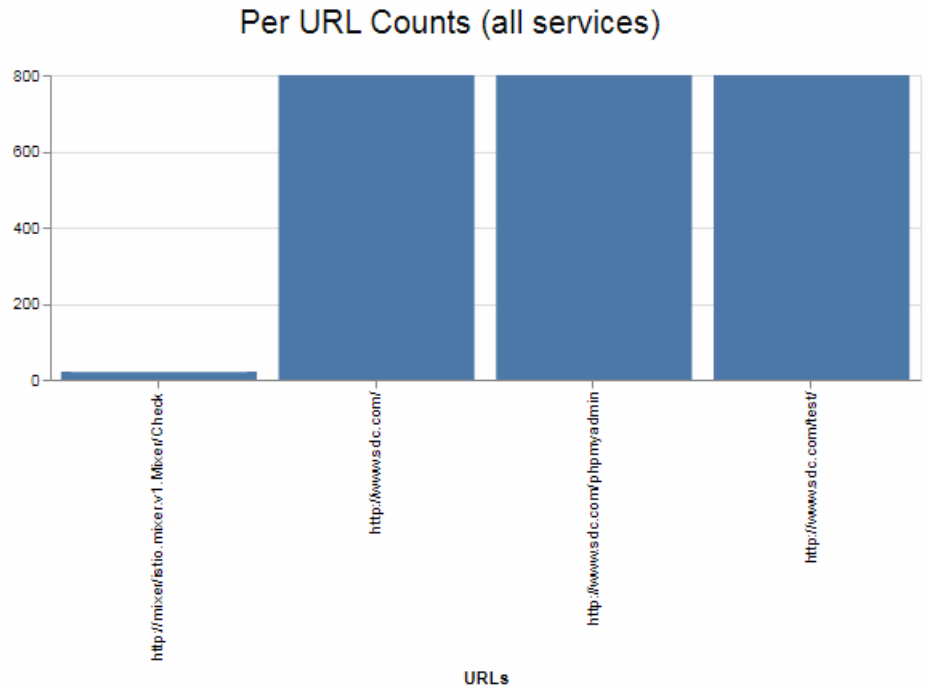
ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Clover Visibility Stats (1-3)

snort-ids | http-lb | istio-policy | jaeger-query | istio-mixer | clover-jmeter-master | istio-ingressgateway | istio-telemetry | clover-server5 | clover-server4

proxy-access-control | clover-server1 | clover-server3 | clover-server2

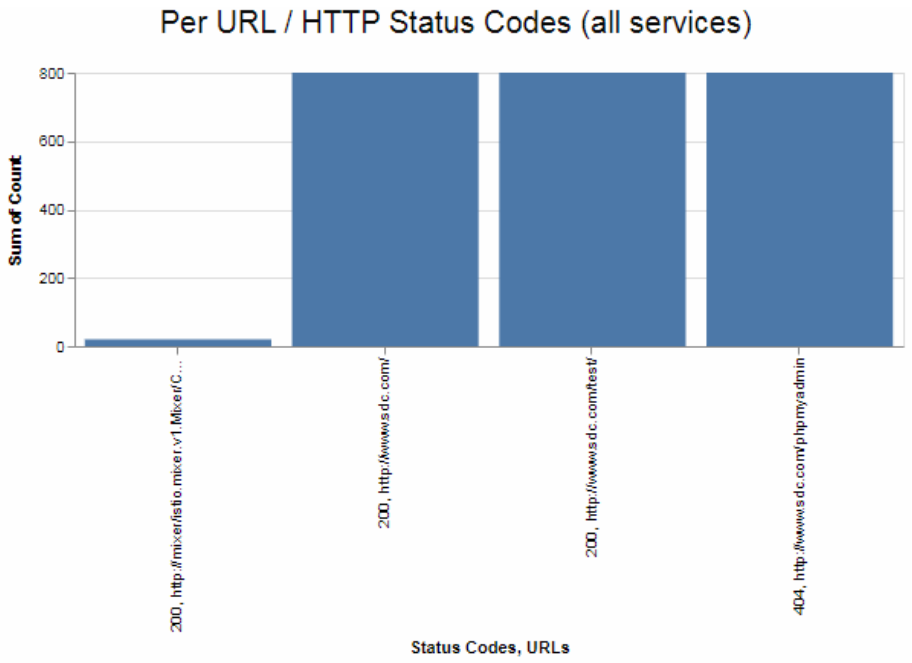
Visibility Services	
1.	clover_server3
2.	clover_server2
3.	clover_server1
4.	proxy_access_control

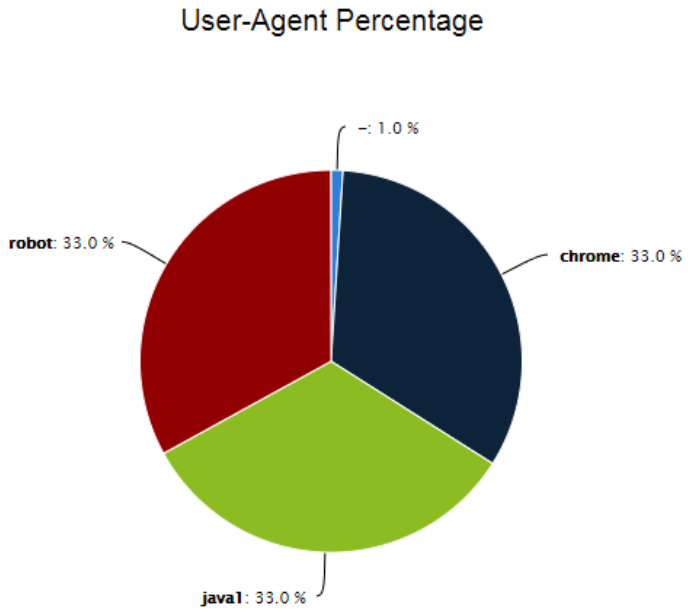


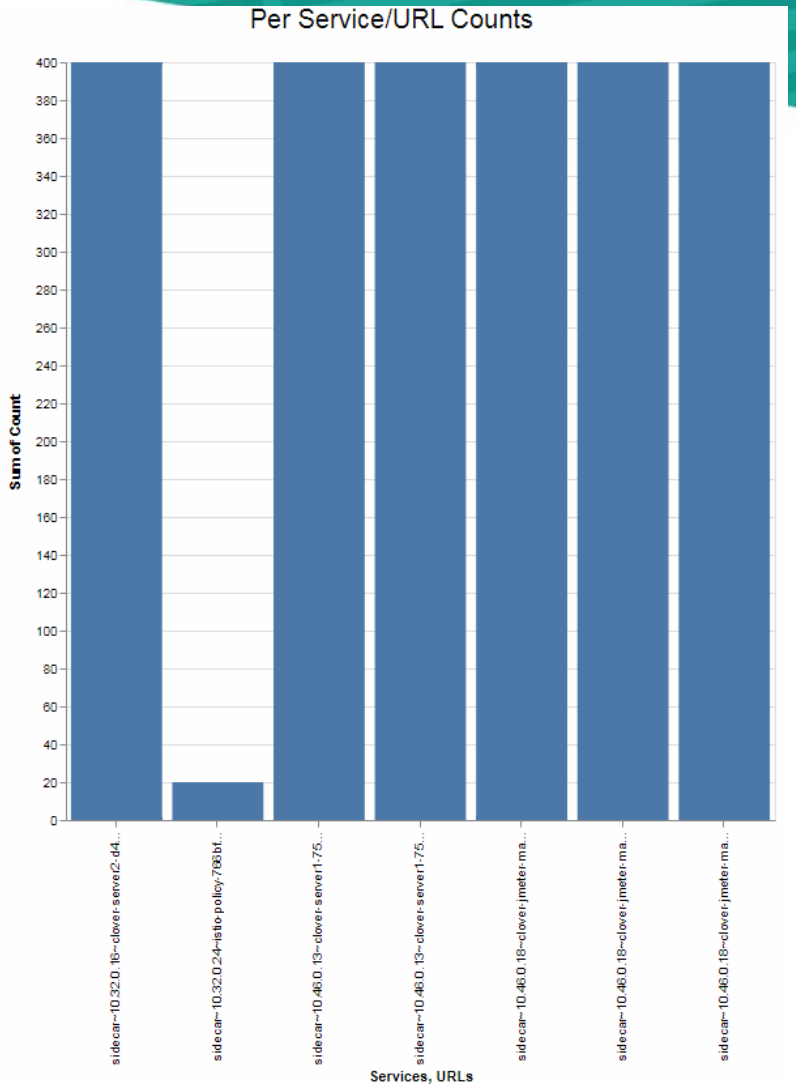
Tracing Metrics	
Average Response Times	System Counts
SDC Proxy: 6ms	Traces: 16 Spans: 146

- 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

HTTP Details

User-Agents | Request URLs | Status Codes

http://snort-ids/
http://clover-server1:9180/
http://clover-server3:9180/
http://clover-server2:9180/
http://http-lb:9180/
http://mixer/istio.mixer.v1.Mixer/Check
http://proxy-access-control.default:9180/

Monitoring Metrics

envoy_cluster_inbound_9180__clover_server3_default_svc_cluster_local_upstream_rq_2xx	4558
envoy_cluster_inbound_9180__proxy_access_control_default_svc_cluster_local_upstream_rq_2xx	12109
envoy_cluster_outbound_9180__clover_server1_default_svc_cluster_local_upstream_rq_2xx	5987
envoy_cluster_outbound_9180__clover_server1_default_svc_cluster_local_upstream_cx_active	8
envoy_cluster_outbound_9180__clover_server2_default_svc_cluster_local_upstream_rq_2xx	5751
envoy_cluster_inbound_9180__clover_server1_default_svc_cluster_local_upstream_rq_2xx	6159
envoy_cluster_outbound_9180__proxy_access_control_default_svc_cluster_local_upstream_rq_2xx	451
envoy_cluster_inbound_9180__clover_server2_default_svc_cluster_local_upstream_rq_2xx	6121
envoy_cluster_outbound_9180__clover_server3_default_svc_cluster_local_upstream_rq_2xx	4177
envoy_cluster_inbound_9180__clover_server1_default_svc_cluster_local_upstream_cx_active	0
envoy_cluster_outbound_9180__clover_server2_default_svc_cluster_local_upstream_cx_active	10
envoy_cluster_inbound_9180__proxy_access_control_default_svc_cluster_local_upstream_cx_active	0
envoy_cluster_outbound_9180__proxy_access_control_default_svc_cluster_local_upstream_cx_active	7
envoy_cluster_outbound_9180__clover_server3_default_svc_cluster_local_upstream_cx_active	11
envoy_cluster_inbound_9180__clover_server3_default_svc_cluster_local_upstream_cx_active	0
envoy_cluster_inbound_9180__clover_server2_default_svc_cluster_local_upstream_cx_active	0

Clover Clovisor

Istio

- Large compute footprint
  - Istio - 13 Containers
  - Sidecar container per service
- 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

ons EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Clovisor: Network Tracing… the Cloud Native Way




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 kernel 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

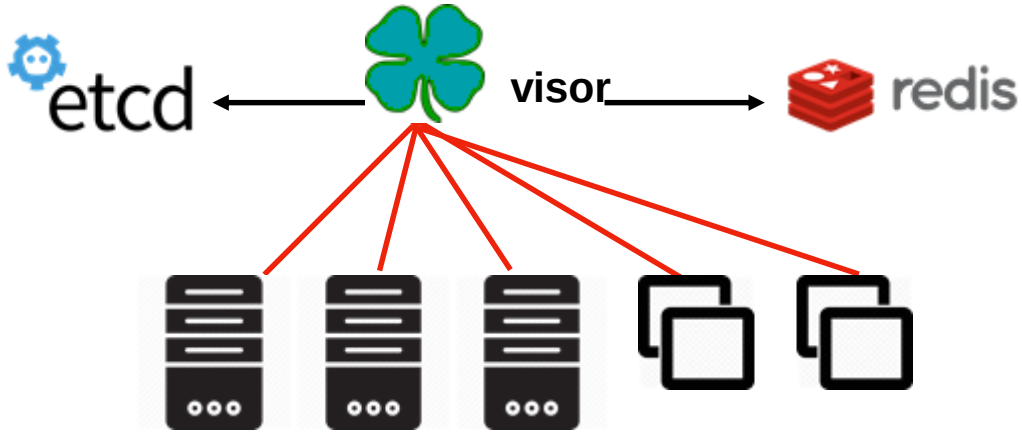
3. In-depth Integration with Cloud Native Ecosystem Projects:



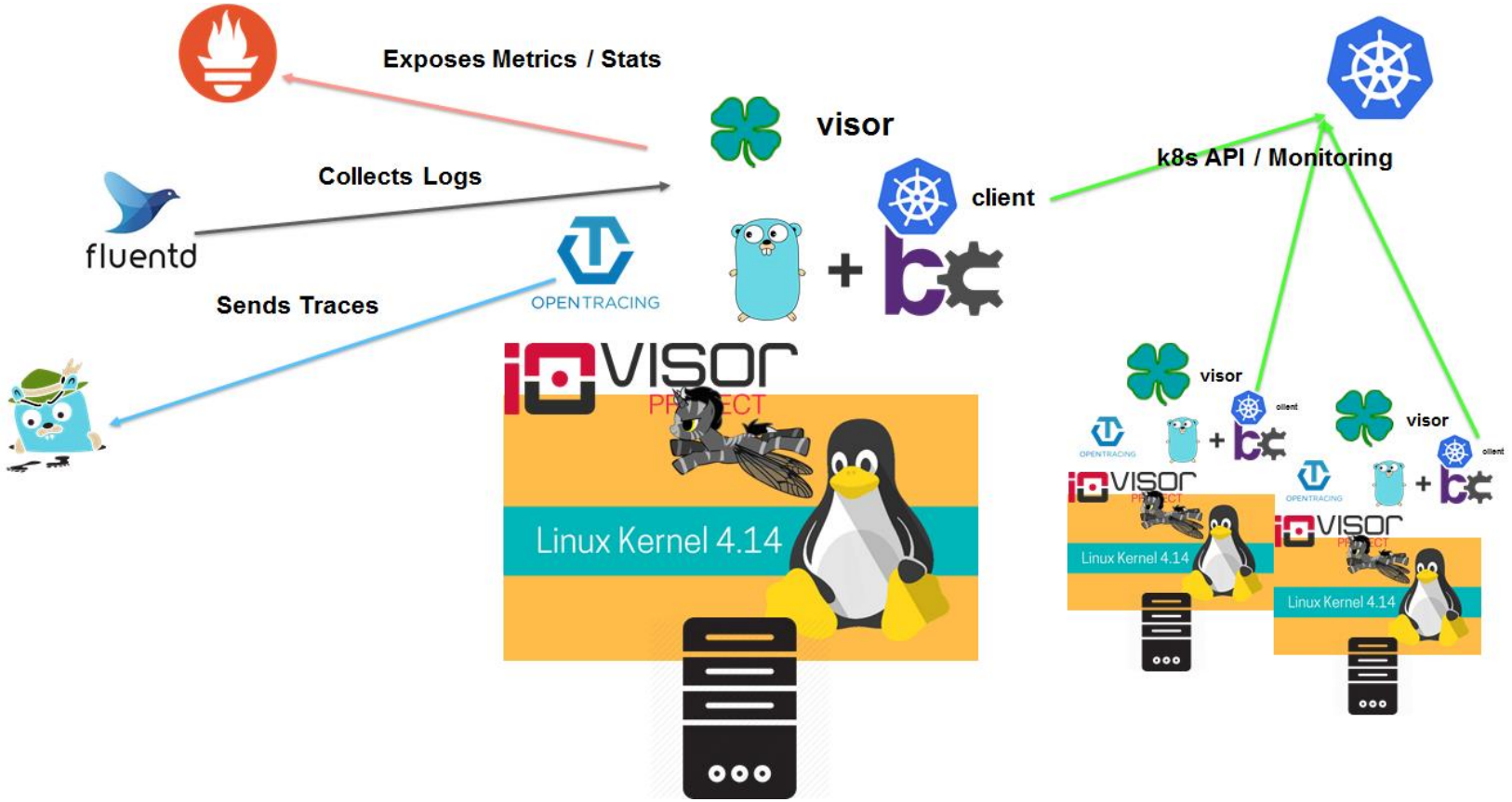


a) Built-in Kubernetes Client
- Monitoring k8s pod states

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



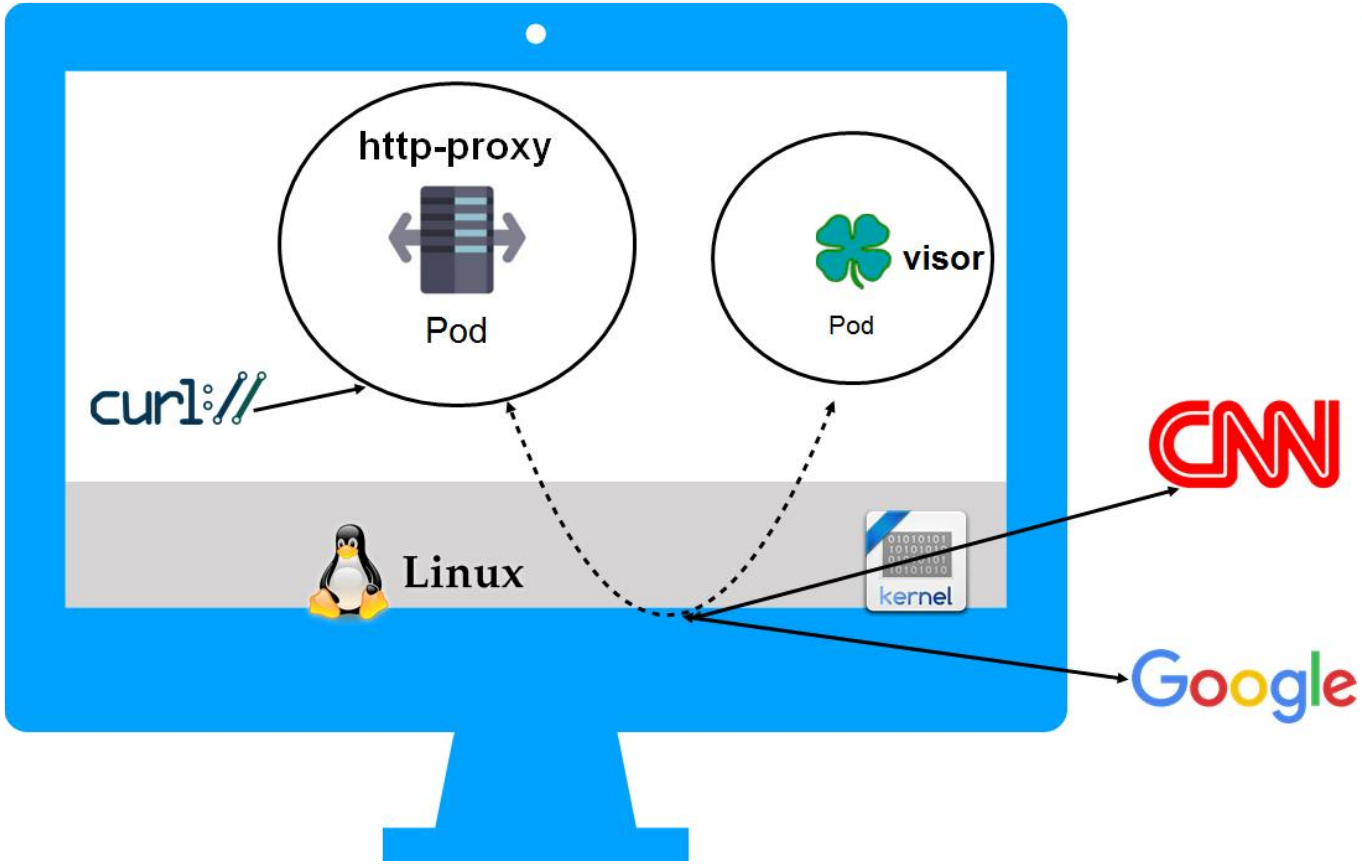
Clovisor Architecture



- 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

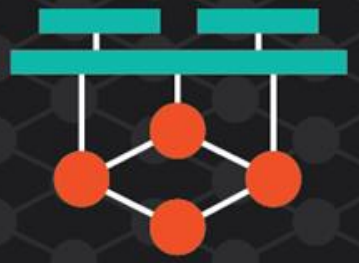
Clovisor Demo

- 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



Visibility Use-Cases

- Easily pinpoint issues with individual services
- Integrate into CI to determine success/failure of jobs
- 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



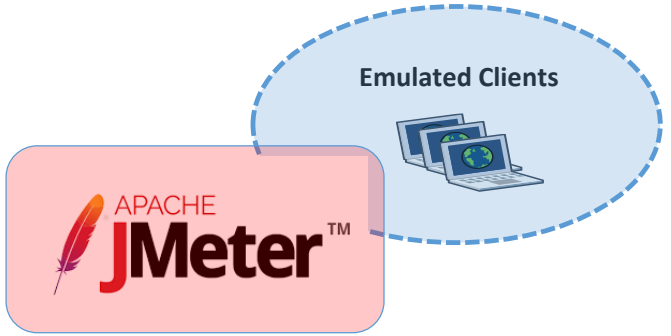
ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

September 25 - 27, 2018
Amsterdam, The Netherlands

Integrating & Validating

Jmeter Validation (1-2)

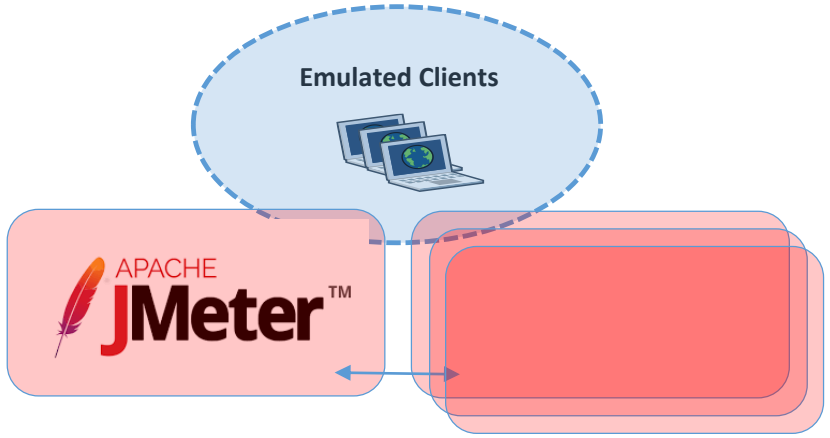
- Jmeter is a mature L4-7 testing open source project
  - HTTP client emulation for functional/performance validation
  - Determine max session/connection rates, connection capacity, etc.
- Clover created a Jmeter service for use within k8s
  - Uses Jmeter master/slave approach
  - Master as a single pod deployment may be used
  - Jmeter slaves can be added for additional scale
    - Master <-> slave communication only works outside of mesh
  - Detailed test plan creation, test control and result collection
  - Integrated into clover-system in CLI, UI and clover-controller



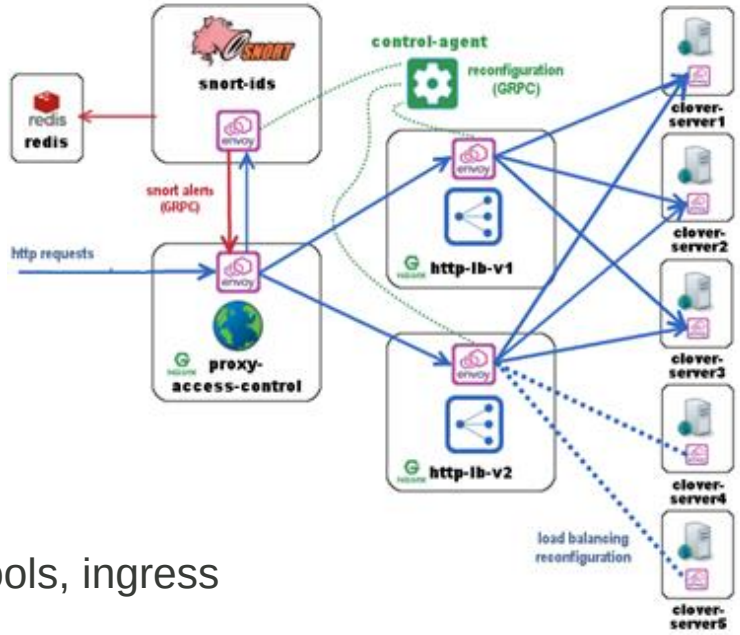

```
$ cloverctl create testplan –f jmeter.yaml
$ cloverctl start testplan
$ cloverctl start testplan –s <slave count>
$ cloverctl get testresult –r log
$ cloverctl get testresult –r results
```

Jmeter Validation (2-2)

k8s



clover-jmeter-master clover-jmeter-slave



- Validate infrastructure including visibility tools, ingress controller, sample app
- Facilitate CI jobs
- Configure clover-server(s) with resources including URL routes and files of varying sizes

Jmeter Test Plan Creation

- Clover Jmeter yaml abstracts test plan XML
- Specify simple parameters:
  - # threads (users)
  - Loops
  - URL List
    - Name (for results)
    - URL (with port and URI)
    - Method (GET, POST, …)
    - User-agent HTTP header

jmeter.yaml

```
load_spec:
    num_threads: 20
    loops: 2
    ramp_time: 60
url_list:
    - name: url1
      url: http://www.sdc.com:80/test/
      method: GET
      user-agent: java1
    - name: url2
      url: http://www.sdc.com:80/
      method: GET
      user-agent: chrome
    - name: url3
      url: http://www.sdc.com:80/phpmyadmin
      method: GET
      user-agent: robot
```

```
$ cloverctl create testplan –f jmeter.yaml
$ cloverctl start testplan
$ cloverctl get testresult –r results
```

Jmeter Results

```
1537388803608,2,url3,404,Not Found,ThreadGroup 1-19,text,false,,327,0,1,1,2,0,0
1537388803611,3,url1,200,OK,ThreadGroup 1-19,text,true,,843,0,1,1,3,0,0
1537388803615,2,url2,200,OK,ThreadGroup 1-19,text,true,,843,0,1,1,2,0,0
1537388803618,2,url3,404,Not Found,ThreadGroup 1-19,text,false,,327,0,1,1,2,0,0
1537388806602,4,url1,200,OK,ThreadGroup 1-20,text,true,,843,0,1,1,4,0,0
1537388806607,3,url2,200,OK,ThreadGroup 1-20,text,true,,843,0,1,1,3,0,0
1537388806610,2,url3,404,Not Found,ThreadGroup 1-20,text,false,,327,0,1,1,2,0,0
1537388806613,2,url1,200,OK,ThreadGroup 1-20,text,true,,843,0,1,1,2,0,0
1537388806616,2,url2,200,OK,ThreadGroup 1-20,text,true,,843,0,1,1,2,0,0
1537388806619,2,url3,404,Not Found,ThreadGroup 1-20,text,false,,327,0,1,1,2,0,0
```

Setup Jenkins for CI

- Setup Jenkins within k8s

```yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: jenkins
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: jenkins
    spec:
      containers:
      - name: jenkins
        image: jenkins/jenkins:lts
        ports:
        - containerPort: 8080
```

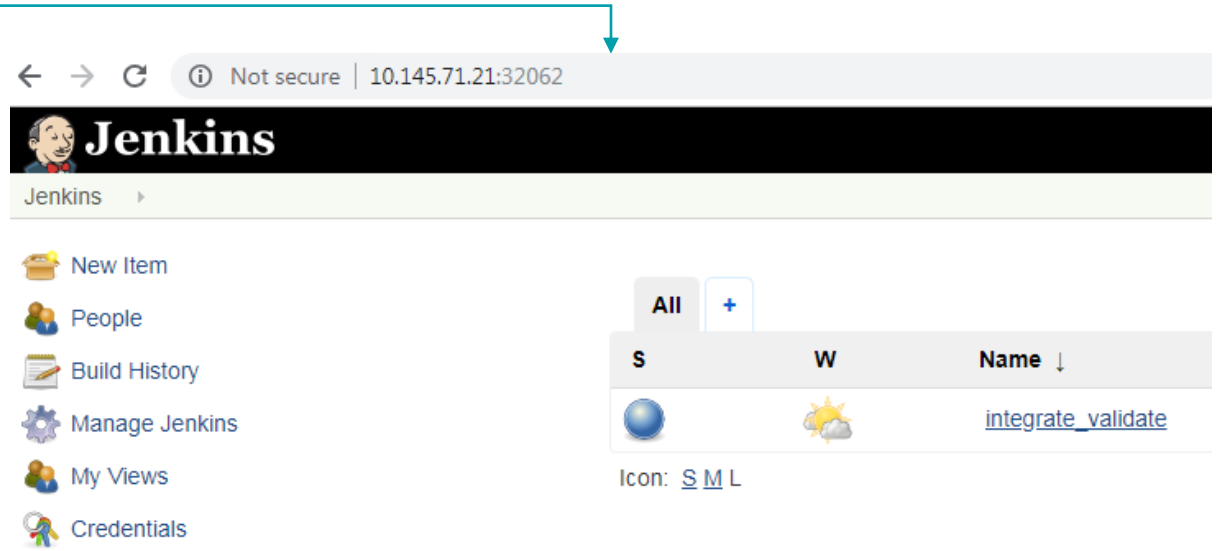
Deployment yaml

```yaml
kind: Service
apiVersion: v1
metadata:
  name: jenkins
spec:
  selector:
    app: jenkins
  ports:
  - name: ui
    protocol: TCP
    port: 8080
    targetPort: 8080
  - name: discover
    protocol: TCP
    port: 50000
    targetPort: 50000
  type: LoadBalancer
```

Load Balancer service yaml (expose in GKE)

```yaml
kind: Service
apiVersion: v1
metadata:
 name: jenkins
spec:
 selector:
   app: jenkins
 ports:
 - name: http
   protocol: TCP
   port: 8080
 type: NodePort
```

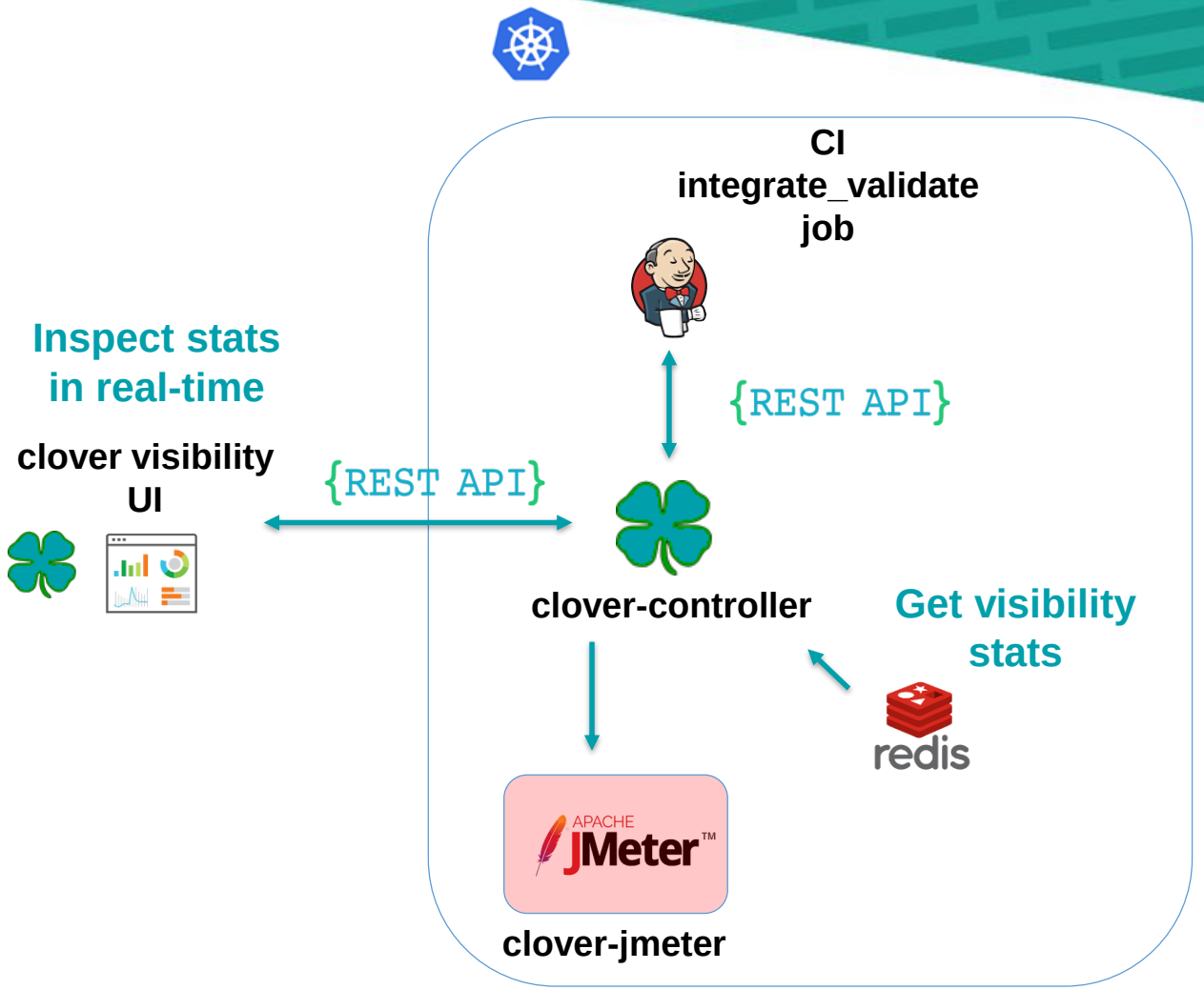
NodePort service yaml



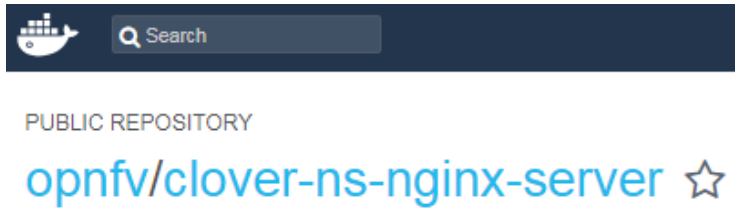
Integrate and Validate Demo

- Python script uses clover-controller REST interface:
  - Clear visibility
  - Create jmeter testplan
  - Start jmeter testplan
  - Get visibility stats

- PASS/FAIL from expected requests sent by jmeter checked from visibility
- Set exit status in script for Jenkins job success/failure



Clover Server Instrumentation



- clover-server
  - Endpoint to terminate traffic for end-to-end validation through network services
  - Nginx based server
- gRPC interface to reconfigure:
  - Setup various paths, listening port, etc.
- Nginx Upload module used for file upload with good performance

Configure

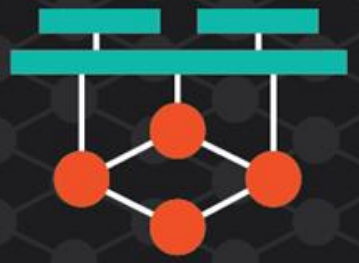
Listening port
Deployment name
Site root/index

Path URLs

Move uploaded files to paths

```
server_port: "9180"
server_name: "clover-server1"
site_root: "/var/www/html"
site_index: index.html
upload_path_config: "/upload"
upload_path_test: "/upload_test"
locations:
    - uri_match: "/test"
      directive: "try_files $uri @default1"
      path: "/test"
    - uri_match: "/new"
      directive: "try_files $uri @default1"
      path: "/new"
    - uri_match: "/clover/clover"
      directive: "try_files $uri @default2"
      path: "/clover/clover"
files:
    - src_file: "/var/www/html/upload/0000000001"
      dest_file: "var/www/html/test/touch.wav"
```

$ cloverctl set server –f server.yaml



ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

September 25 - 27, 2018
Amsterdam, The Netherlands

Deploying & Managing Services, Infrastructure

Spinnaker Introduction

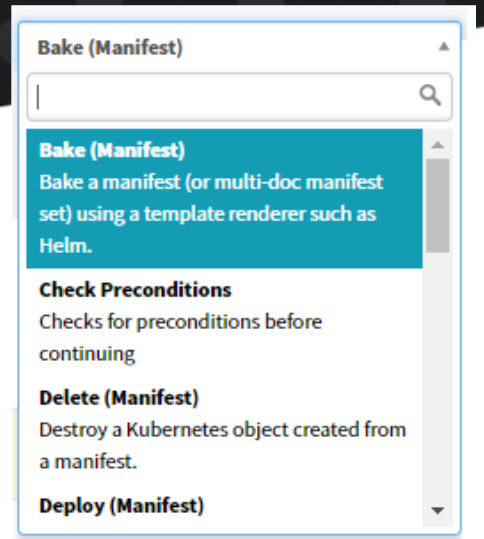
- Overview
  - Construct and manage continuous delivery workflows
  - View/manage cloud resources
  - Pipeline-based engine

Spinnaker

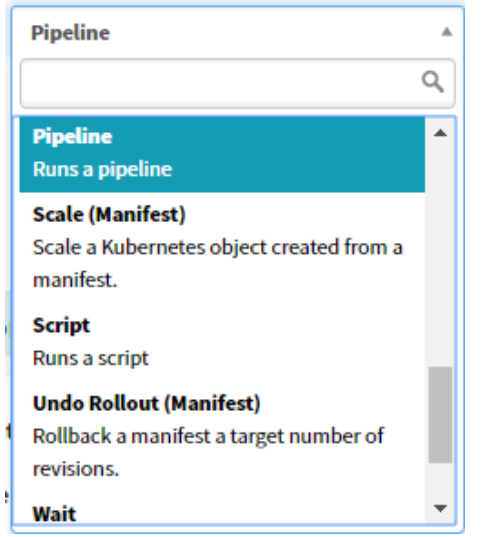
- Stages
  - A stage in Spinnaker is an atomic building block for a pipeline

- Pipelines
  - Support various deployment strategies: blue/green, canary…
  - Deploy to various clouds
  - Execution manually or based on triggers
  - Stages can be executed sequentially or in parallel

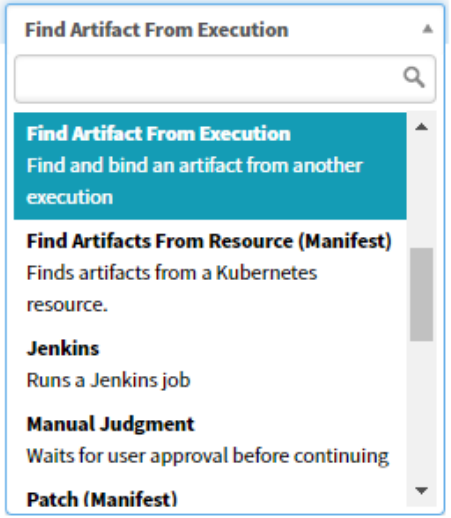
Spinnaker – Stage Types



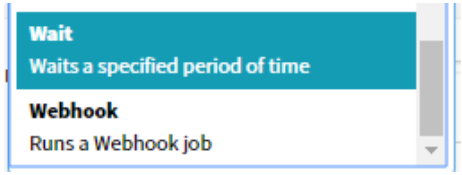
- Bake (Manifest)
  - deploy Helm charts (alpha)
- Check Preconditions
  - check environment
- Delete (Manifest)
  - delete k8s resources
- Deploy (Manfifest)
  - deploy k8s based on yaml



- Pipeline
  - allow pipeline daisy-chaining
- Scale (Manifest)
  - increase k8s replicas
- Script
  - run a script (instead of Jenkins option)
- Undo Rollout (Manifest)
  - go back to a prior revision



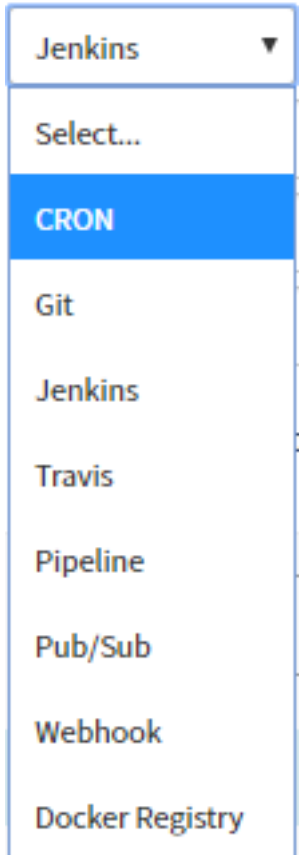
- Find Artifact From Execution
  - promote artifacts between executions
- Find Artifacts From Resource
  - find image from k8s resource
- Jenkins
  - run Jenkins jobs
- Manual Judgment
  - prompts user before continuing

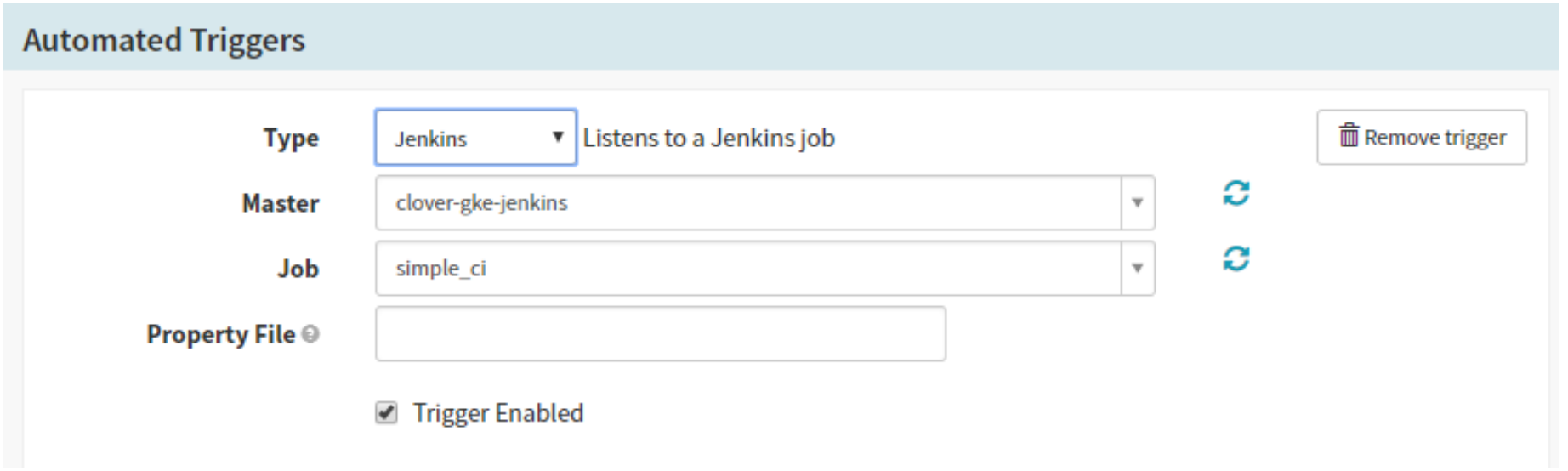


- Wait
  - introduce delay in pipeline
- Webhook
  - execute REST call

ons EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

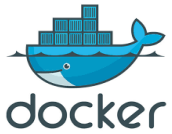
Spinnaker – Pipeline Triggers





Trigger Spinnaker pipelines from many different events including:

Jenkins, Git, Docker Registry or other Spinnaker pipelines



ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

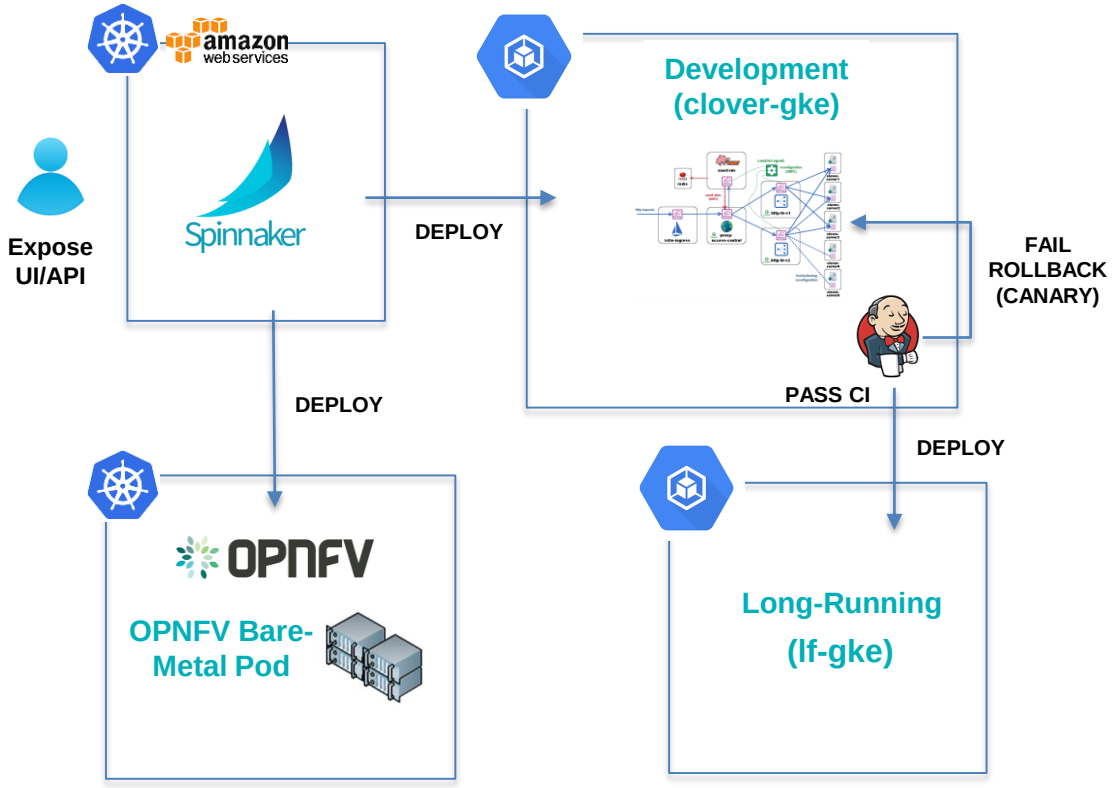
Spinnaker – Common Software Deployment Strategies

- Blue/Green
  - Two identical environments – ex. green in production
  - Release new version of services in blue and validate
  - Revert to green if issues exist
- A/B Testing
  - Support multiple versions 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: load balancers, proxies and/or service meshes (ex. Istio) to support

Cloud-Native CI/CD with Spinnaker - Demo

- Spinnaker can deploy to multiple cloud providers
  - Including Kubernetes, GKE
  - Openstack
- Pipelines are used to control flow from commit/build/test to bake and deploy in ‘production’
- CI validation scripts are used to determine if individual services and overall use-cases are healthy



Clover Spinnaker Sample-App

ons EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Developer

1 Change Code Or Add Git Tag

Github Pubic Repo

Job monitor-git

2 Use Jenkins git plugin and setup Polling Job

LF-GKE

Job build-sample-app

sample-app local docker build

5 Push to DockerHub

docker

Spinnaker

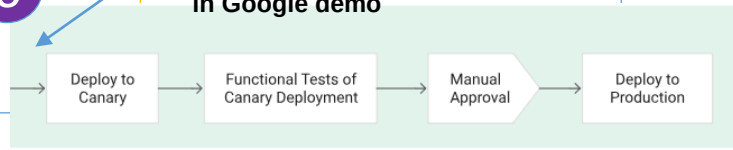
Poll for Jenkins Job monitor-git

3

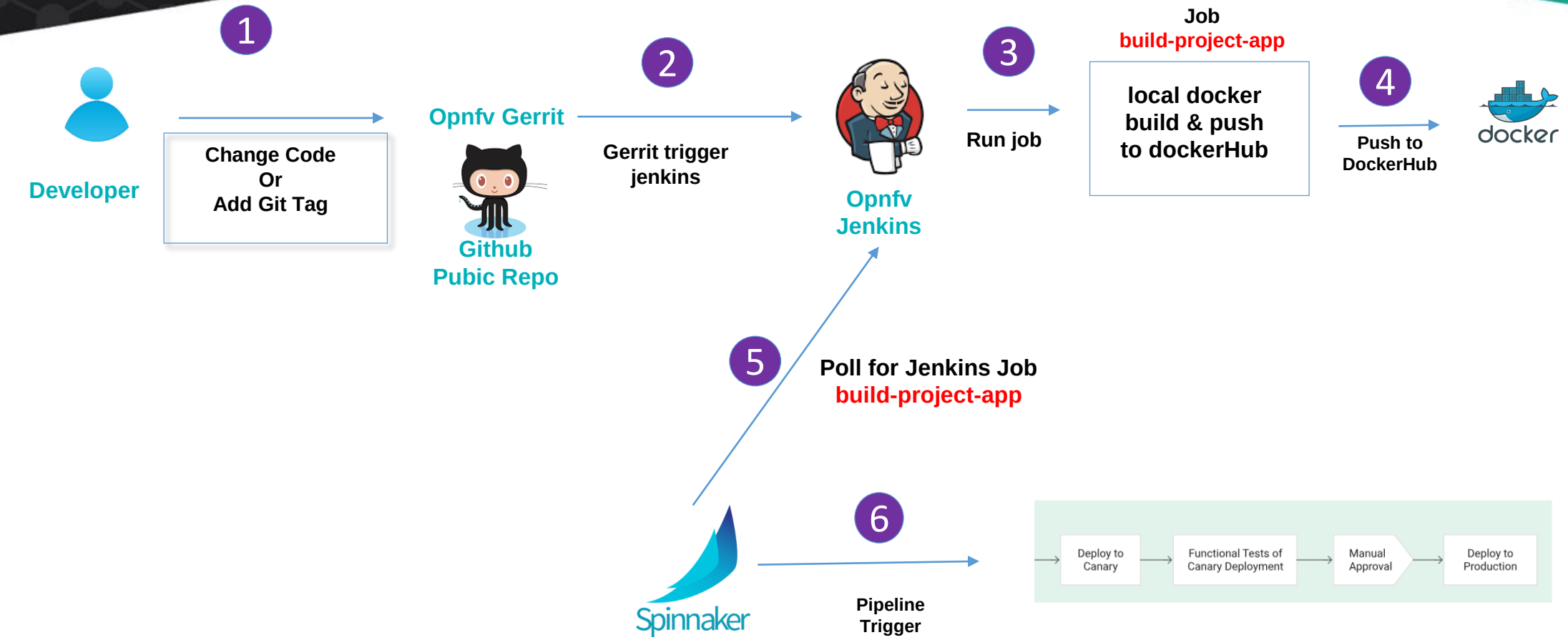
Pipeline Trigger

4 Run Jenkins Job build-sample-app

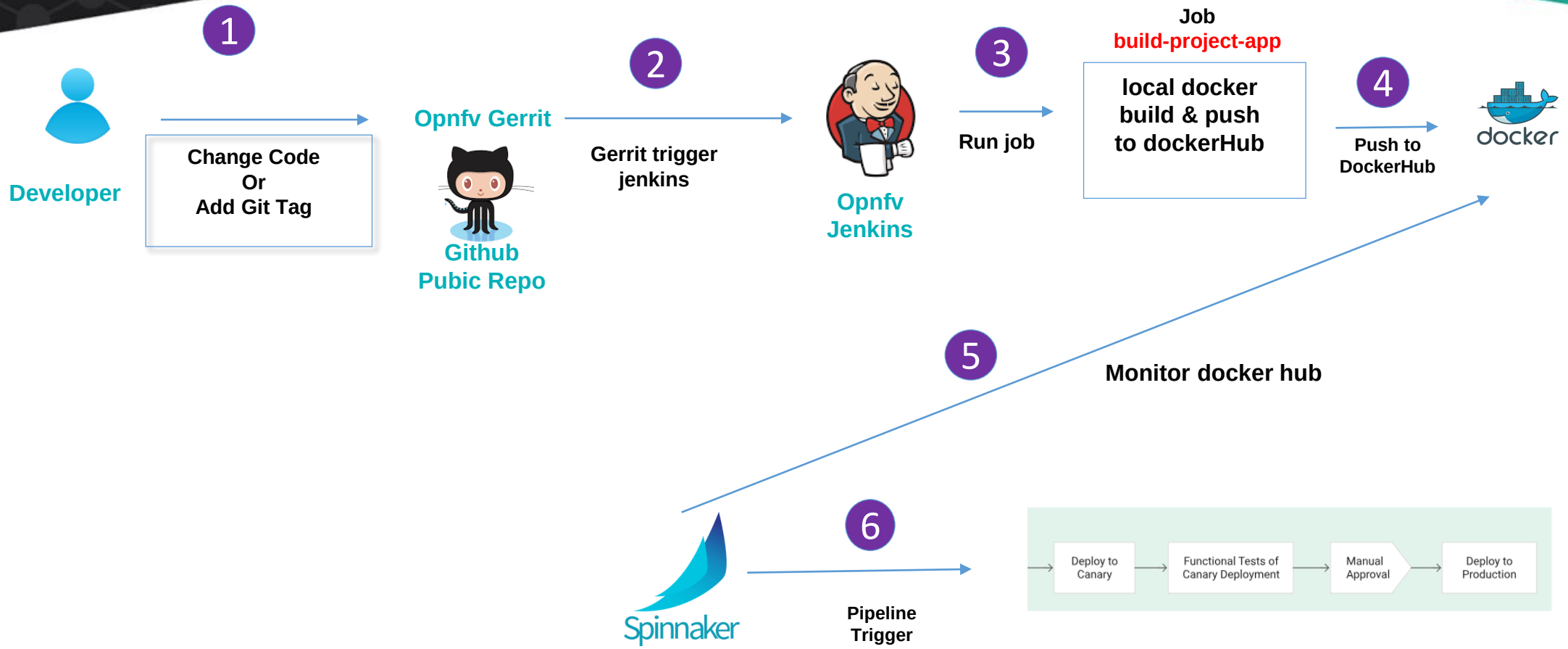
6 Run rest of pipeline in Google demo

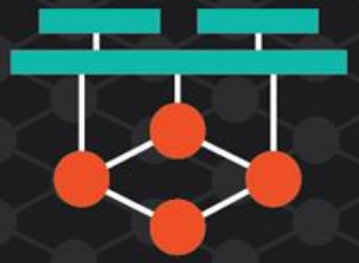


OPNFV CI/CD with Spinnaker (1-2)



OPNFV CI/CD with Spinnaker (2-2)





ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

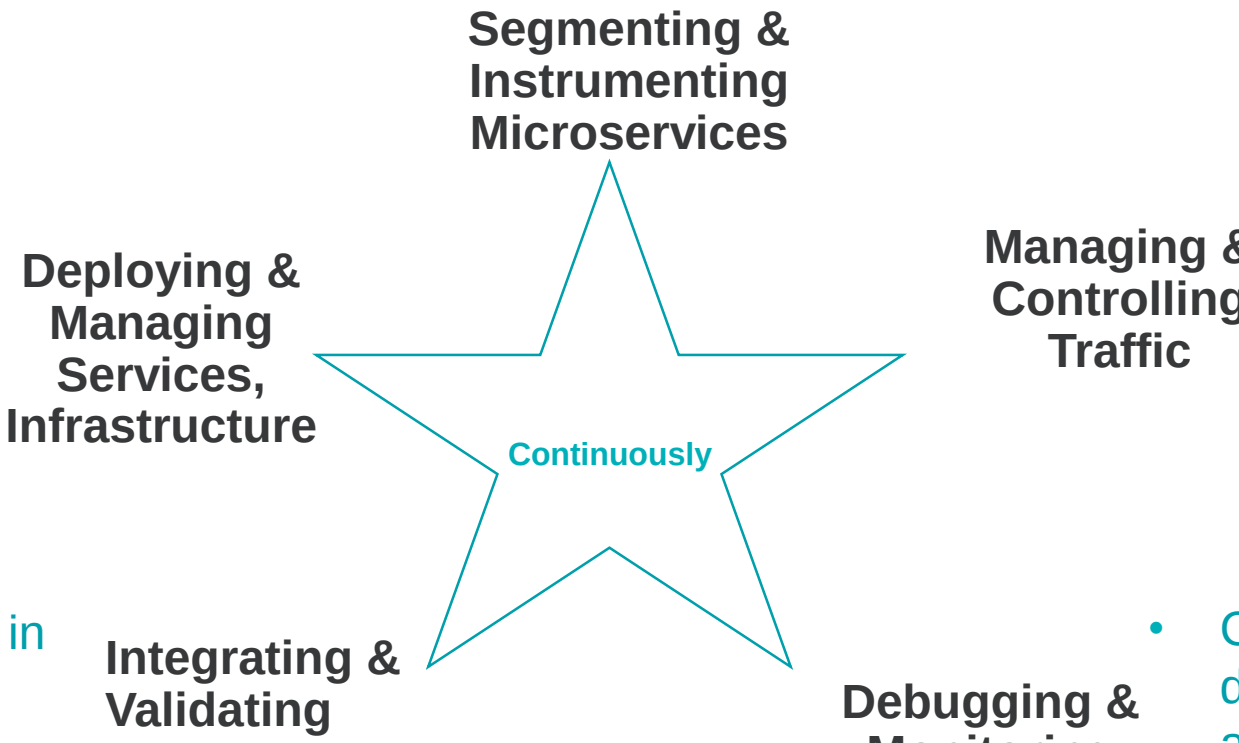
September 25 - 27, 2018
Amsterdam, The Netherlands

Summary

Take-Aways

- LFN projects can be packaged in flexible ways if delivered as microservices in k8s
  - Test projects, ONAP, OS admin services, etc.

- Spinnaker can help manage complex CD pipelines across clouds
- Flexible integrations with Jenkins, DockerHub, Git, etc. allow CI and CD to be combined



- Meshes allow services to be delivered with cloud-native CD principles
- Ideal for control-plane and REST services

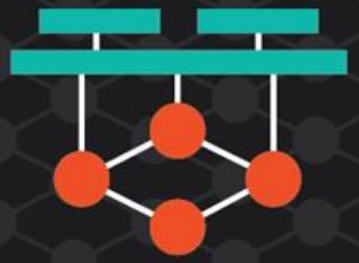
- Validation tools required in CI/CD pipeline stage acceptable
- Employ visibility in CI logic

- Cloud-native visibility helps developers pinpoint issues and operators manage infrastructure

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

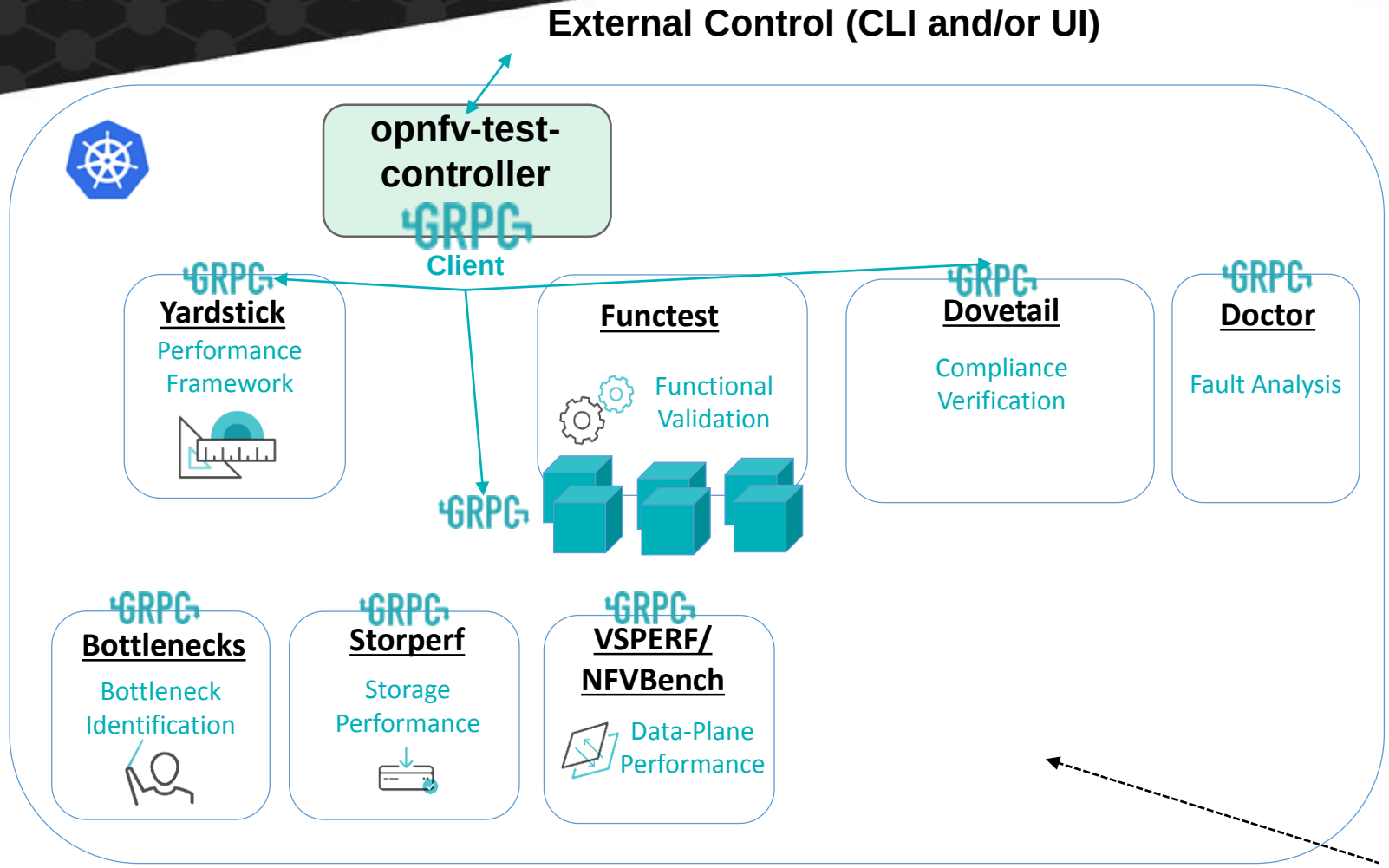


ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

September 25 - 27, 2018
Amsterdam, The Netherlands

Appendix

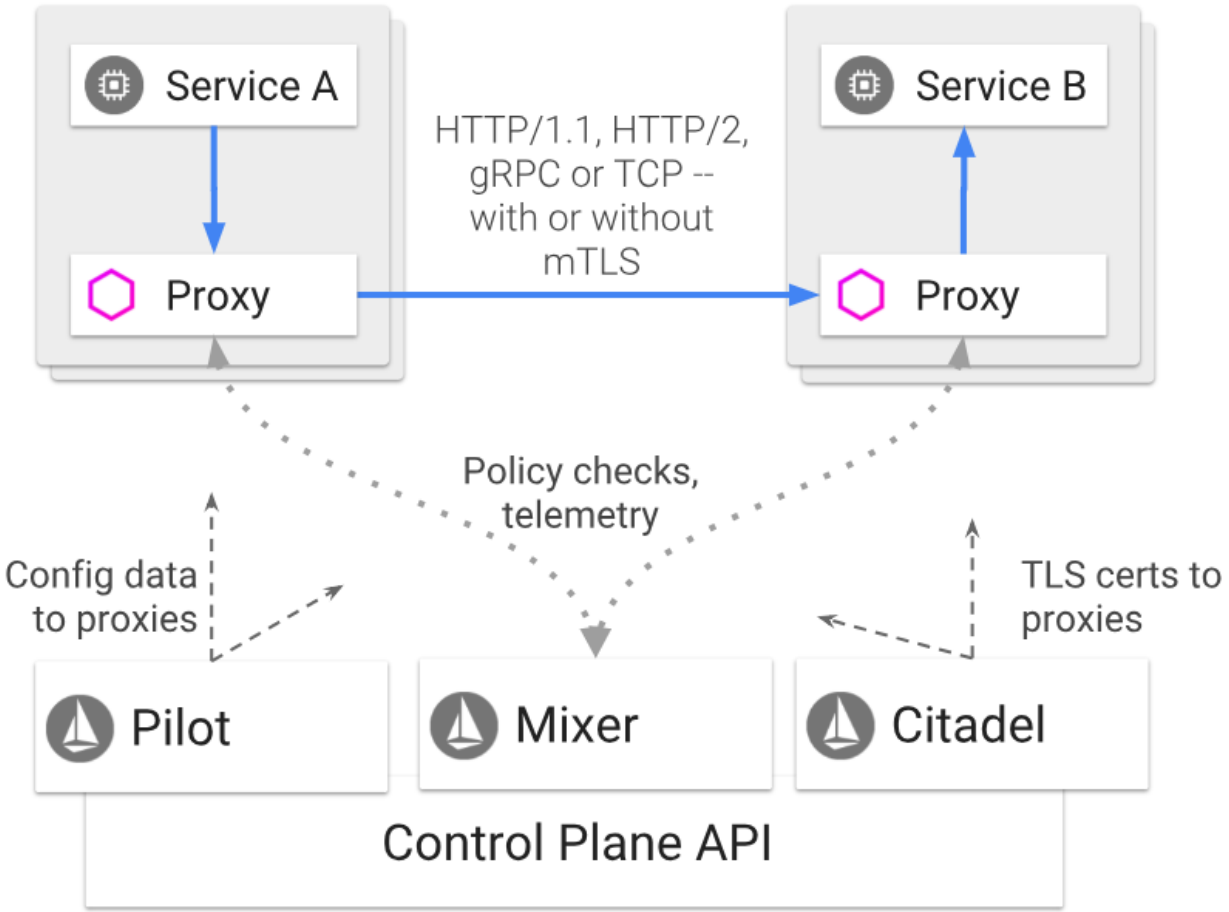
Cloud Native & OPNFV Test Projects



- **Consider cloud native for OPNFV test projects**
  - Package as micro-services
  - Many are already containerized
    - Functest divided into 8+
  - Add gRPC or REST server interfaces
  - Make actions more atomic within each
  - Orchestrate system level tests using different combinations of services/actions
  - Deploy all OPNFV test services in a single manifest potentially
  - Use tool-chains such as Spinnaker for CI/CD
  - Installer projects are also considering cloud native for some services

ons
EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Istio Control-Plane Components

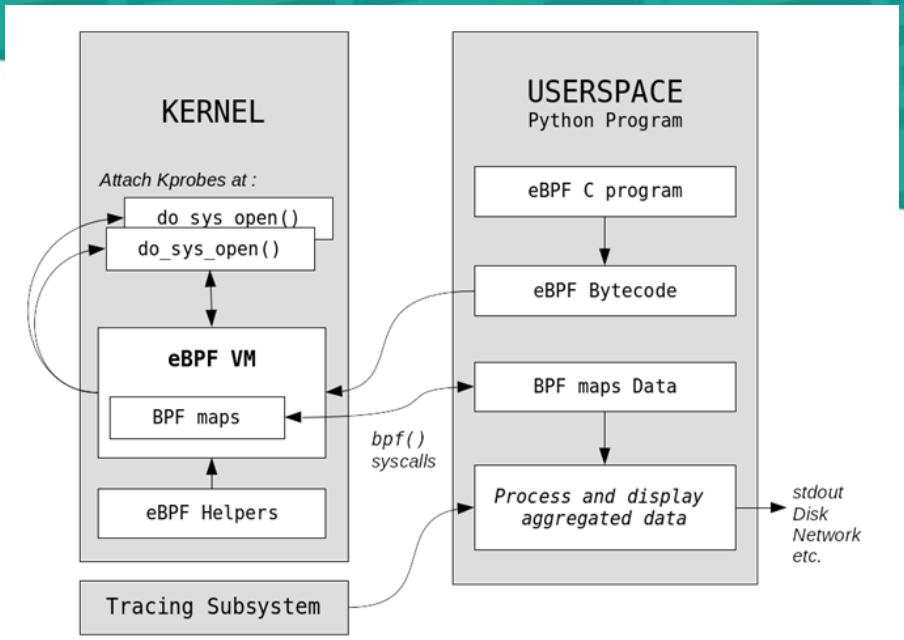


ons EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

IOVisor & eBPF

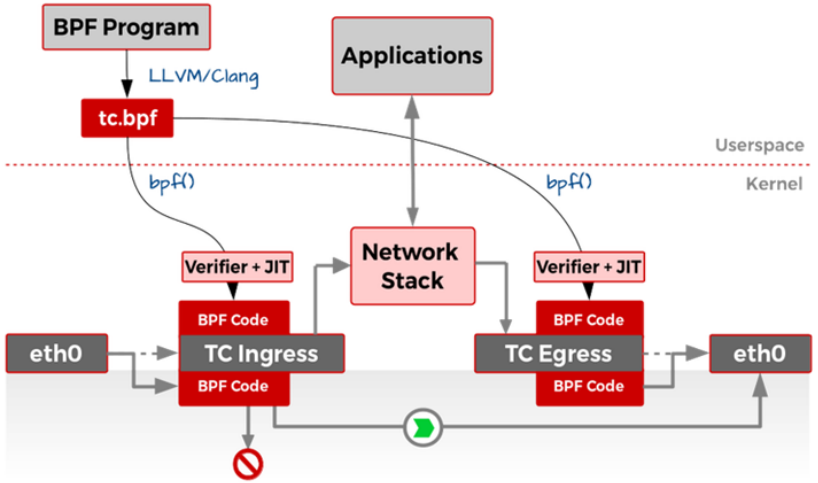
- eBPF
  - Inject bytecodes to kernel trace points / probes
    - Event driven model
  - Networking: tc
    - Utilizes Linux tc (traffic control) to inject bytecode on ingress and egress direction of a network interface
  - Verifier / JIT (just-in-time compiler)
    - Verifier ensures bytecode does NOT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

IOVISOR PROJECT



ons EUROPE
OPEN NETWORKING //
Integrate, Automate, Accelerate

Integrate and Validate Demo

```python
import requests
import sys
import time

CLOVER_CONTROLLER = 'http://10.145.71.21:30880'
#CLOVER_CONTROLLER = 'http://clover-controller.clover-system'
USER_COUNT = 4
EXPECTED_COUNT = 4 * 4

# Clear visibility
clear_url = CLOVER_CONTROLLER + '/visibility/clear'
response = requests.get(clear_url)
print(response.text)
if response.status_code != 200:
    print("Failed to clear visibility")
    sys.exit(-1)

# Create a Jmeter testplan
jmeter_create_url = CLOVER_CONTROLLER + '/jmeter/create'
json_content = {"load_spec":{"num_threads":"{}".format(USER_COUNT),"ramp_time":"60","loops":"2"},
                "url_list":[{"url":"http://proxy-access-control.default:9180",
                             "name":"url1","method":"GET","user-agent":"chrome"},
                            {"url":"http://proxy-access-control.default:9180",
                             "name":"url2","method":"GET","user-agent":"safari"}]}
response = requests.post(jmeter_create_url, json=json_content)
print(response.text)
if response.status_code != 200:
    print("Failed to create testplan")
    sys.exit(-1)
```

```python
# Start Jmeter testplan
jmeter_start_url = CLOVER_CONTROLLER + '/jmeter/start'
response = requests.get(jmeter_start_url)
print(response.text)
if response.status_code != 200:
    print("Failed to start jmeter")
    sys.exit(-1)

time.sleep(80)

# Get visibility trace count
get_stats_url = CLOVER_CONTROLLER + '/visibility/stats/toplevel'
response = requests.get(get_stats_url)
data = response.json()
print("Trace count: {}".format(data['trace_count']))
if response.status_code != 200:
    print("Failed to get visibility stats")
    sys.exit(-1)

# Simple check to determine validation status
if int(data['trace_count']) != EXPECTED_COUNT:
    print("Validation failed {} != {}".format(data['trace_count'],
                                              EXPECTED_COUNT))
    sys.exit(-1)
else:
    print("Validation passed")
```