ONAP on Service Mesh

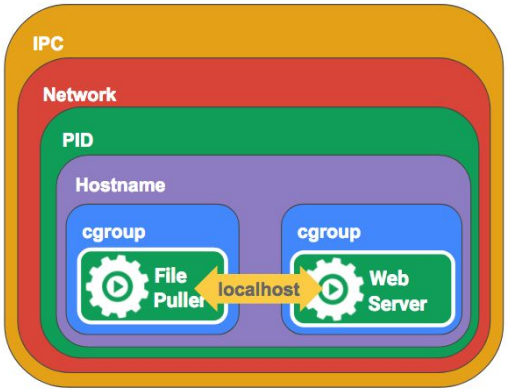
Collaboration Between ONAP and OPNFV Projects

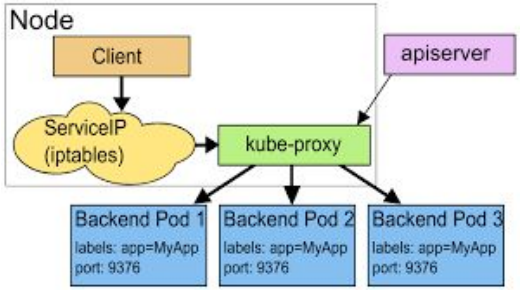
ONAP DDF + OPNFV Plugfest

January 8-11, 2019 Nozay, Paris France

Stephen Wong, Chaker Al Hakim, Gildas Lanilis

Brief Intro to Kubernetes (Pod & Service)

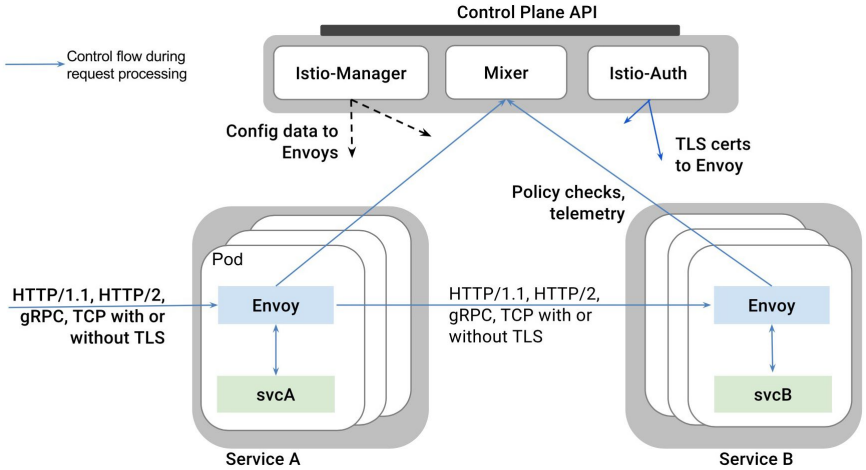




- Pod is the finest granular object in k8s (and it is the unit of scaling up)
- Pod can have one or more containers running inside (different cgroup, same namespace)
- Same network namespace means containers within a pod can communicate with each other via localhost
- Typical pod only has one interface (eth0 addressable inside the container, veth pair on host) - one IP address per pod

- Service is what is exposed as an accessible entity (“endpoint”) in k8s, backed by pods. Basically a “microservice”
- Service maps to the set of pods via labels
- Service names == domain name entry in DNS server maintained by k8s
- A service moreover can expose one or more ports (i.e., TCP port). kube-proxy, which runs on each node, maintains the mapping of the external service:port to a backend pod (on a node). When multiple ports are exposed for a service, port needs to be named

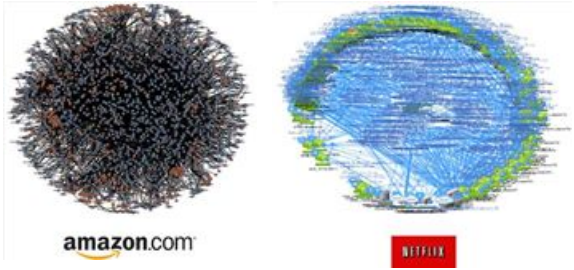
Brief intro to Istio



Need for service mesh in k8s setup:

- Decouple infrastructure logic (IP, where pod is scheduled…etc) from application development
- Exposes declarative language for applications to express infrastructure needs
- Telemetry: Istio provides rich set of telemetry, and provides user programmability to create more metrics
- Seamless Deployment via sidecar (service proxy runs as a container in each pod, adjacent to the application container(s))

Cloud Native / Microservice Challenges



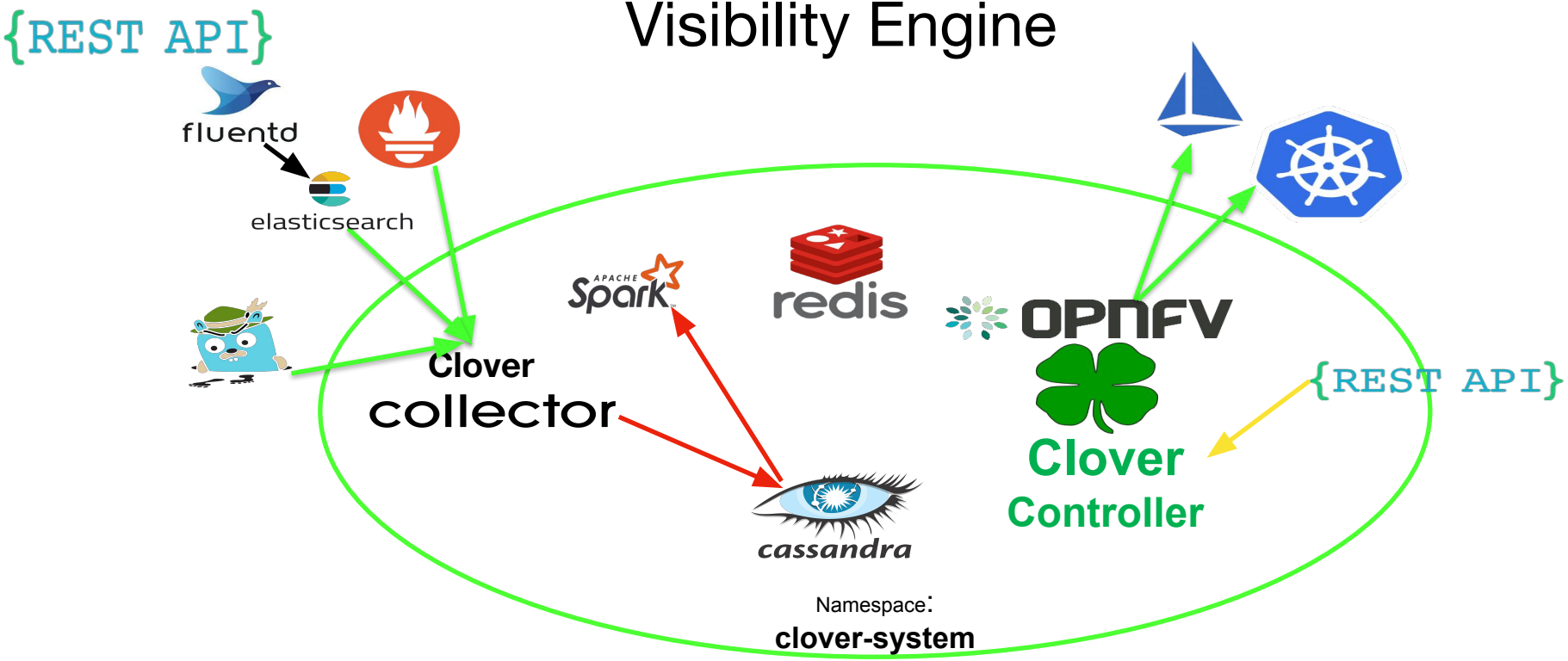
150+ containerized services

ONAP
OPEN NETWORK AUTOMATION PLATFORM

- 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

OPNFV Clover Visibility Engine



1. The core software module runs on clover-system namespace, and is responsible for installing Clover related software packages
2. At its core is the Clover Collector —- which reads from various data sources and organize / giving structure (i.e., allowing SQL like query) to the incoming raw data, and store the structured data into datastore (Cassandra)
3. Spark is used as the analytic engine to analyze data from the datastore. Currently it is used as a correlation engine to correlate data from various sources into a simple query structure for visibility purpose

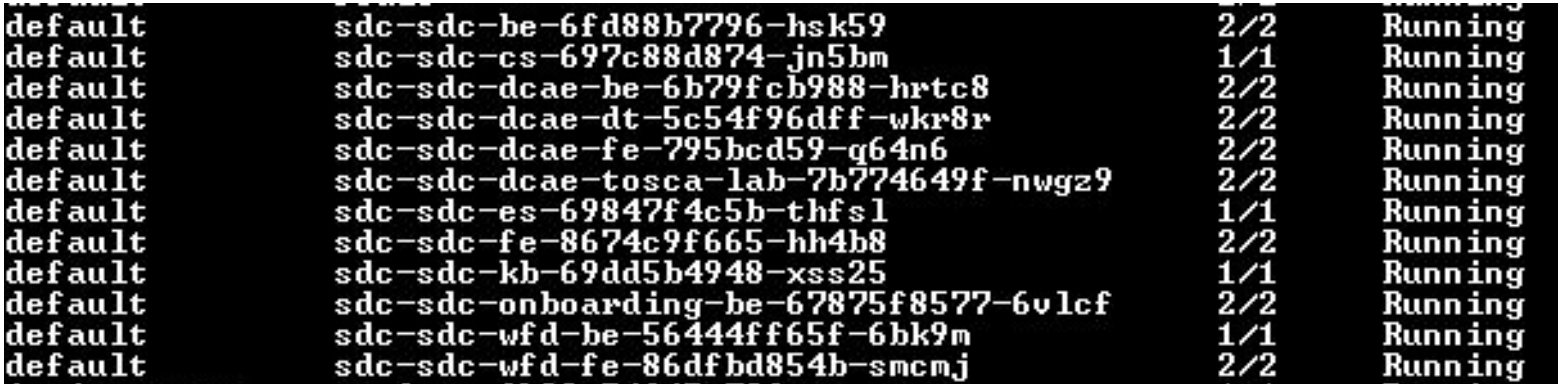
ONAP SDC Workflow Scenario in UI (Demo)

- Sign into SDC as cs0008, choose ONBOARD
- Create new licensing model - VLM – Vendor License Model
- Create New Software Project – CREATE NEW VSP
- Upload the heat template .zip – SOFTWARE PRODUCT ATTACHMENTS
- 'Proceed To Validation' to validate the HEAT template
- Check-in and submit
- Import VSP from SDC Home
- Create VF based on HEAT template

ONAP SDC Deployment in Kubernetes

- Install with Helm
  - Deploy tiller and set RBAC
  - Install SDC chart - version: sdc-3.0.0.tgz

- Add SDC to default service account



- Standard SDC pods deployed in k8s (not in mesh)

ONAP SDC Injection in Istio Service Mesh

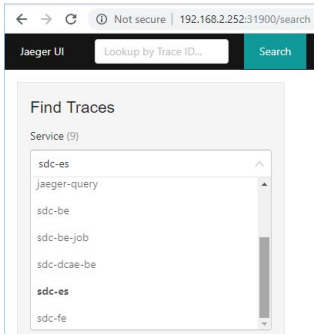
- Full injection of SDC Helm chart using Istio automatic injection to configured namespace doesn’t work
  - Some headless services like Cassandra are blocked by Envoy sidecar proxies
  - YAML for SDC service objects not ready for Istio
- Service mesh injection for most services using manual inject
  - Decompose Helm to get k8s YAML for individual SDC services
  - Modify YAML with Istio ‘http’ prefix for SDC service objects
  - Individually delete services from k8s and employ Istio manual inject using k8s yaml for each service
  - Order of service injection important due to many SDC init dependencies
    - K8s jobs must be executed post injection

ONAP SDC Injection in Service Mesh

- SDC services injected within Istio

```
ubuntu@assassin:~/istio-1.0.0$ istioctl proxy-status
PROXY                                                  CDS        LDS        EDS               RDS          PILOT                            VERSION
istio-egressgateway-59b9887fb8-fkb4z.istio-system      SYNCED     SYNCED     SYNCED (100%)     NOT SENT     istio-pilot-c88bc787d-thspg      1.0.0
istio-ingressgateway-89c5bd64f-rnj85.istio-system      SYNCED     SYNCED     SYNCED (100%)     SYNCED       istio-pilot-c88bc787d-thspg      1.0.0
sdc-sdc-be-5cd54fcc8b-8mkgf.default                    SYNCED     SYNCED     SYNCED (100%)     SYNCED       istio-pilot-c88bc787d-thspg      1.0.0
sdc-sdc-dcae-be-5cb5bc4dd5-v8j4c.default               SYNCED     SYNCED     SYNCED (100%)     SYNCED       istio-pilot-c88bc787d-thspg      1.0.0
sdc-sdc-dcae-dt-5686f9cd45-gxxfg.default               SYNCED     SYNCED     SYNCED (100%)     SYNCED       istio-pilot-c88bc787d-thspg      1.0.0
sdc-sdc-dcae-fe-78f8665c99-w74w4.default               SYNCED     SYNCED     SYNCED (100%)     SYNCED       istio-pilot-c88bc787d-thspg      1.0.0
sdc-sdc-es-55f8df4474-wxw9d.default                    SYNCED     SYNCED     SYNCED (100%)     SYNCED       istio-pilot-c88bc787d-thspg      1.0.0
sdc-sdc-fe-f4cddb59b-7d957.default                     SYNCED     SYNCED     SYNCED (100%)     SYNCED       istio-pilot-c88bc787d-thspg      1.0.0
sdc-sdc-wfd-be-56765c6c8f-jqxcm.default                SYNCED     SYNCED     SYNCED (100%)     SYNCED       istio-pilot-c88bc787d-thspg      1.0.0
sdc-sdc-wfd-fe-775b67f645-lf9bt.default                SYNCED     SYNCED     SYNCED (100%)     SYNCED       istio-pilot-c88bc787d-thspg      1.0.0
```

- Jaeger UI



- Clover Visibility Dashboard

- SDC services begin to show in Jaeger UI and Clover visibility dashboard
- Traffic must target service to show in visibility

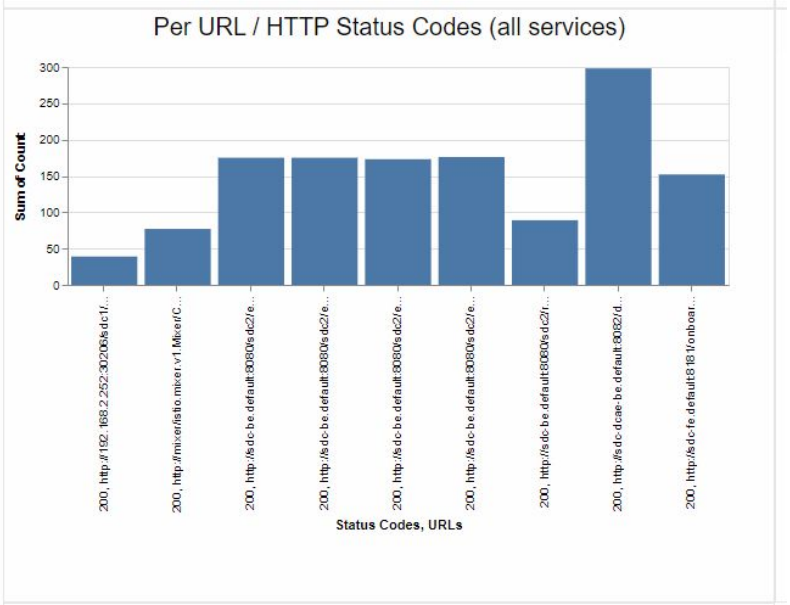
Clover Visibility from Service Mesh

- Health check URLs for various SDC services

HTTP Details

Request URLs | Status Codes

http://sdc-be.default:8080/sdc2/esGateway/
http://sdc-be.default:8080/sdc2/rest/healthCheck
http://sdc-be.default:8080/sdc2/esGateway/.kibana/config/_search
http://sdc-fe.default:8181/onboarding/v1.0/healthcheck
http://sdc-dcae-be.default:8082/dcae/healthCheck
http://mixer/istio.mixer.v1.Mixer/Check
http://sdc-be.default:8080/sdc2/esGateway/_cluster/health/.kibana?timeout=5s
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/notifications
http://sdc-be.default:8080/sdc2/esGateway/_nodes



Clover Visibility from Service Mesh

- Creation of licensing model and VSP – REST Request URLs from visibility:

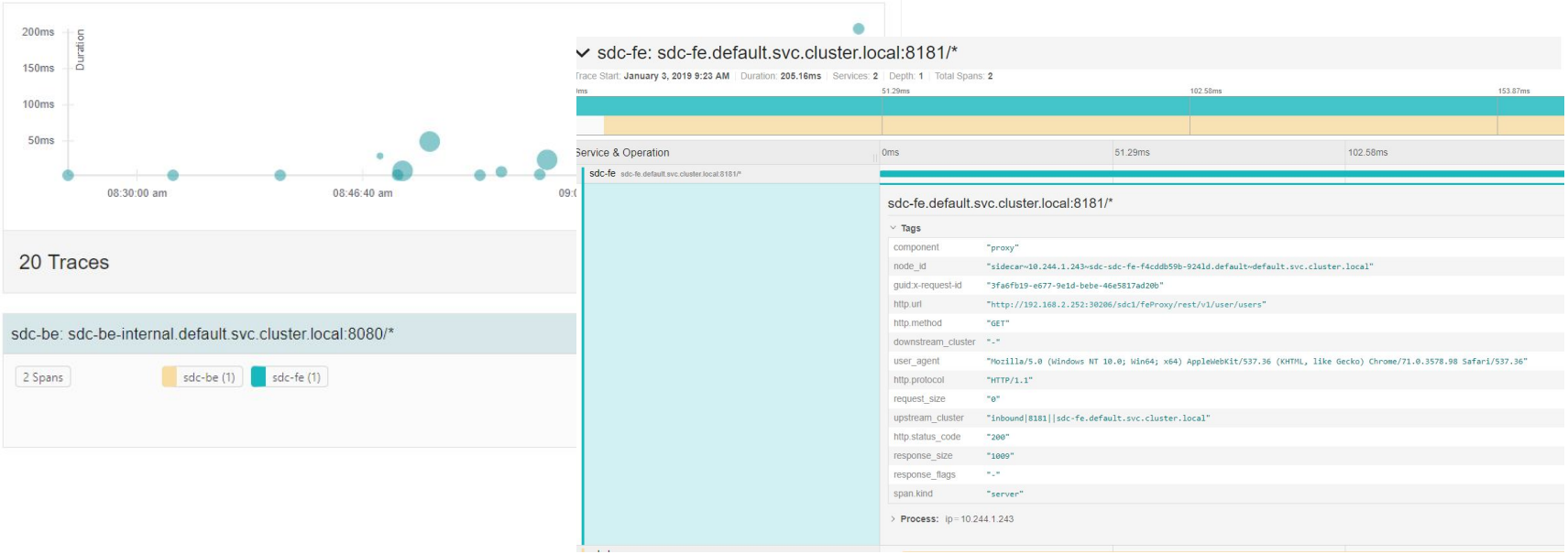
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/items?&itemStatus=ACTIVE&versionStatus=Draft&permission=Owner,Contributor
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/vendor-license-models/?versionFilter=Certified
http://192.168.2.252:30206/sdc1/feProxy/rest/v1/followed
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/items/50fa1f2b354d42468c8987860ea3f221/versions/03c18e5cc4094326ad7b8376fa77ef35
http://192.168.2.252:30206/sdc1/feProxy/rest/v1/catalog/dataTypes
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/vendor-software-products/?versionFilter=Certified
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/notifications
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/vendor-software-products/packages
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/vendor-license-models/?Status=ARCHIVED
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/vendor-license-models/50fa1f2b354d42468c8987860ea3f221/versions/03c18e5cc4094326ad7b8376fa77ef
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/items/50fa1f2b354d42468c8987860ea3f221
http://192.168.2.252:30206/sdc1/feProxy/rest/v1/user/authorize
http://192.168.2.252:30206/sdc1/feProxy/rest/v1/categories
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/togglz
http://192.168.2.252:30206/sdc1/feProxy/rest/v1/user/users
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/vendor-license-models/?versionFilter=Draft
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/items/50fa1f2b354d42468c8987860ea3f221/permissions
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/items/50fa1f2b354d42468c8987860ea3f221/versions/03c18e5cc4094326ad7b8376fa77ef35/activity-logs
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/vendor-software-products/?versionFilter=Draft
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/items/50fa1f2b354d42468c8987860ea3f221/versions/03c18e5cc4094326ad7b8376fa77ef35/actions
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/vendor-software-products/?Status=ARCHIVED
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/items/50fa1f2b354d42468c8987860ea3f221/versions
http://192.168.2.252:30206/sdc1/feProxy/onboarding-api/v1.0/vendor-license-models/
http://192.168.2.252:30206/sdc1/feProxy/rest/v1/configuration/ui

Clover Visibility from Service Mesh

- Onboarding VNF with HEAT templates

Per session tracking with Jaeger UI

- Use Clover visibility for top-level
- Drill down into service graph with Jaeger



- http.url="http,http://sdc-be.default:8080/sdc2/rest/v1/user/users"

SDC Service Startup Order / Mesh Inject Order

- ONAP SDC services have many init containers and k8s jobs that must be executed in a particular order:
  - sdc-sdc-cs
  - sdc-sdc-cs-config
  - sdc-sdc-onboarding-be-cassandra-init
  - sdc-sdc-onboarding-be
  - sdc-sdc-wfd-be
  - sdc-sdc-wfd-fe
  - sdc-sdc-be
  - sdc-sdc-be-config-backend
  - sdc-sdc-fe
  - sdc-sdc-dcae-be (2nd container takes time)
  - sdc-sdc-dcae-be-tools
  - sdc-sdc-dcae-tosca-lab
  - sdc-sdc-dcae-fe

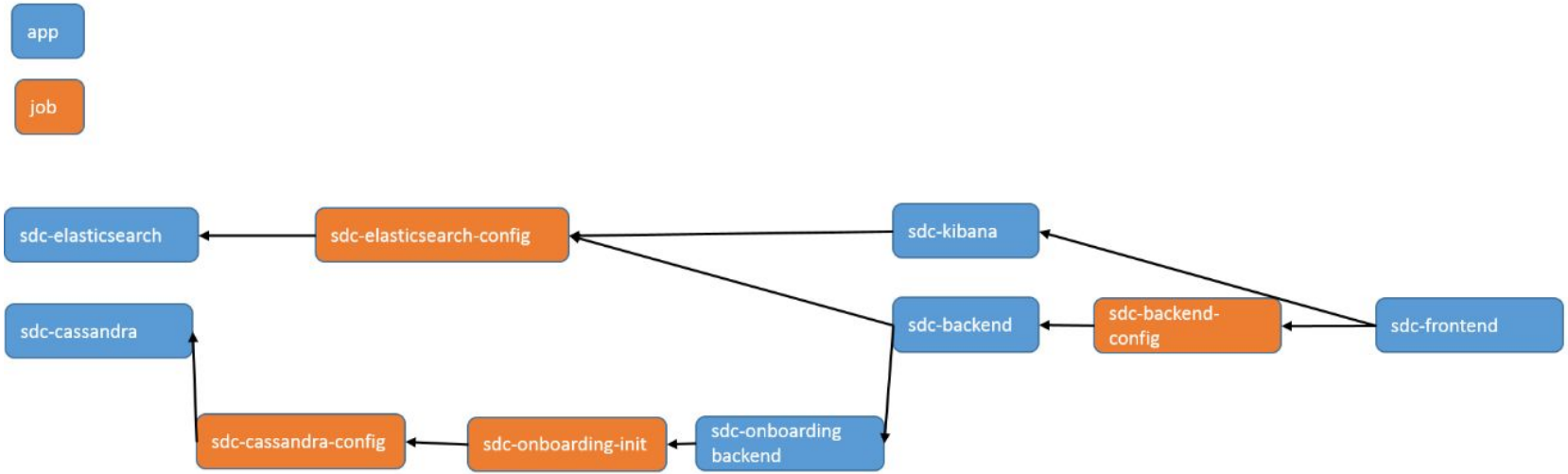
Service Mesh Inject Order

- dcae-dt
- sdc-fe
- wfd-fe
- wfd-be
- sdc-be
- dcae-be
- dcae-fe

- Manual mesh injection requires some order to be preserved with k8s jobs executed after startup

SDC Service Startup Order / Mesh Inject Order

- ONAP SDC dependency map



- Source: https://wiki.onap.org/display/DW/SDC+Troubleshooting

Debugging & Monitoring – System Visibility

Clover → CI/CD

System Level Visibility

Spark, cassandra, OPNFV

Aggregate REST Analytics

Istio

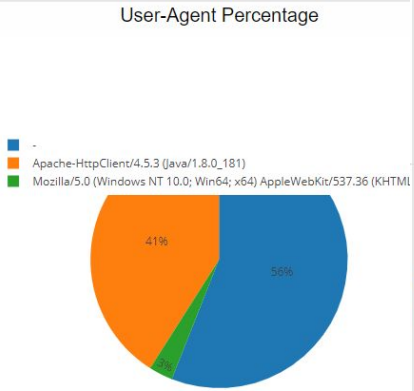
Service Mesh

envoy hooks

Observability Tools

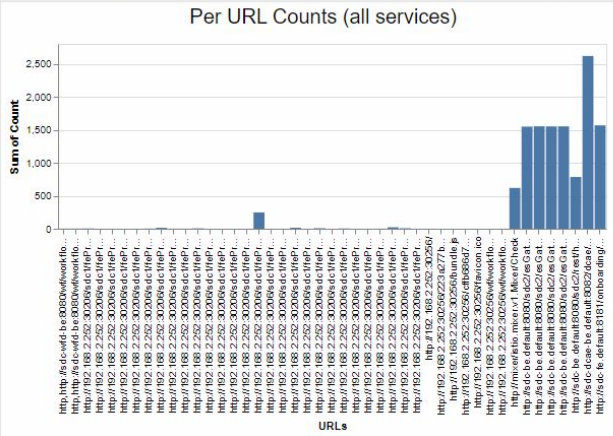
fluentd

CLOUD NATIVE COMPUTING FOUNDATION

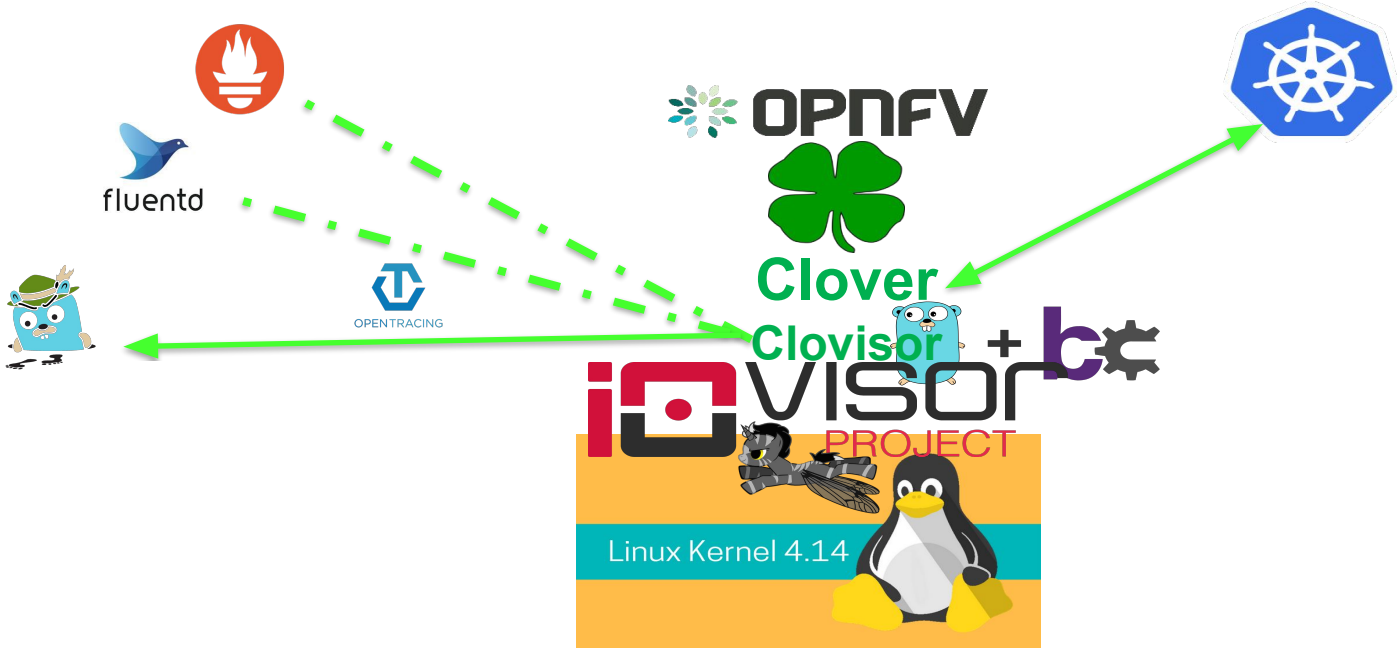


Service Response Times

Service	Min(ms)	Avg(ms)	Max(ms)
sdc-wfd-be	13.99	305.25	504.76
sdc-wfd-fe	8.22	216.95	892.83
sdc-be	0.83	2.52	355.29
sdc-fe	1.75	10.74	1272.72
sdc-es	NA	NA	NA
sdc-dcae-be	0.77	1.23	3.26

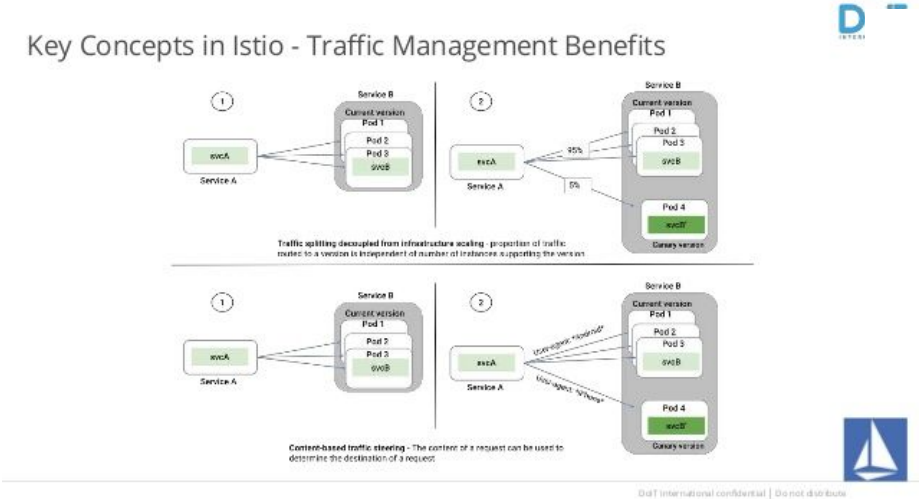


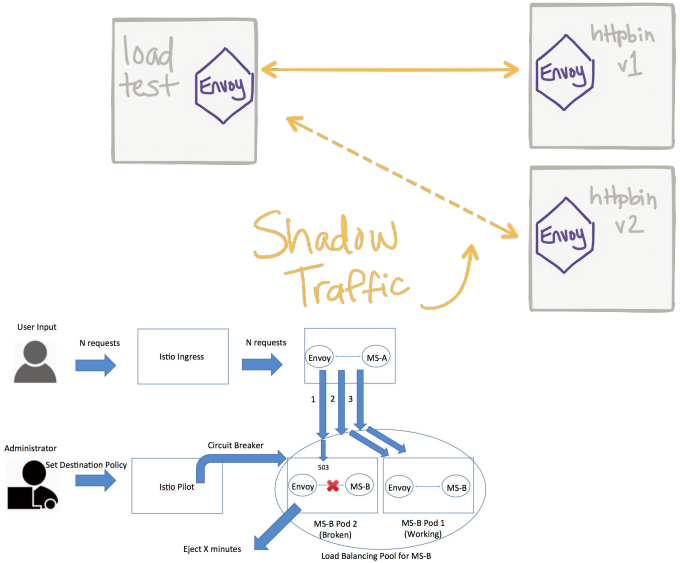
Clover
Clovisor



1. Lightweight, low latency network tracing module
2. Utilizes IOVisor (bcc, gobpf) and eBPF to insert bytecode for both ingress / egress direction of a pod
3. In cluster client to automate process of monitoring and service port / protocol info
4. Stream trace / stats / metrics / logs to respective tracer / collector modules

Benefits of Istio: Traffic Policies





The core Istio “network” policies include:

- Traffic shifting: assign % of traffic to a matching label, assign traffic to a destination label based on HTTP header info (agent, cookie…etc)
- Traffic Mirroring: mirror traffic to two (or more) destination labels
- Circuit Breaking: disconnect traffic to a destination label based on HTTP header fields
- Rate limiting: limit how many connections / sessions any clients can have per a time quanta (e.g.: one second)
- Fault injection: inject delay to any connection to pods associated with a destination label

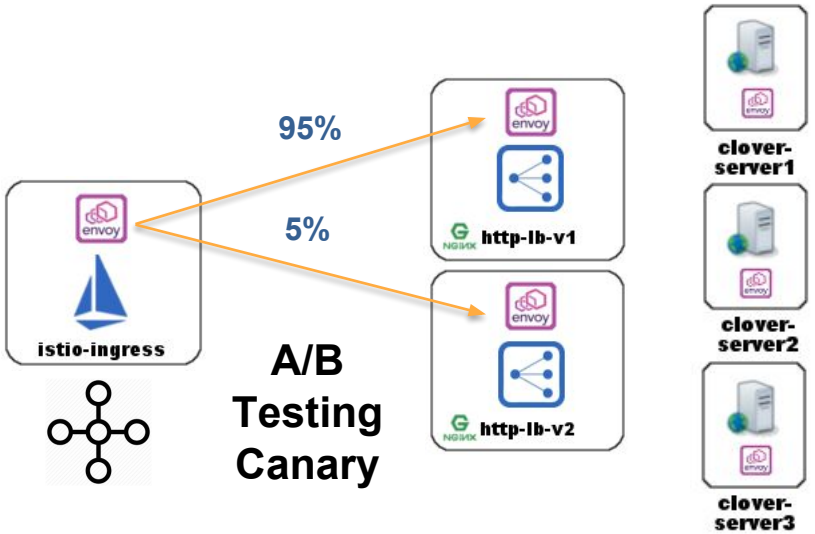
- Mostly operational, fault isolation type policies

Managing & Controlling Traffic – Service Mesh

- Treat network services like apps in deployment
  - Validate new L7 application delivery polices with deployment precepts like canary, blue/green, A/B

Content-based request routing

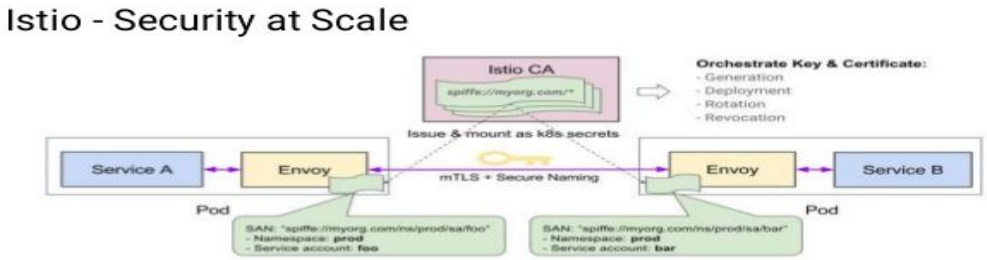
- Match traffic and route to back end service
- Match based on URI, HTTP headers (identity, user-agent)
- Control with ‘weight’ field

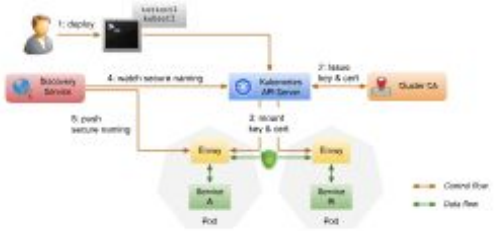


- For developers to validate their REST APIs
- For operators to employ in ‘network as a service’ roll-outs

- Leverage service mesh traffic management to help manage CI/CD processes

Benefits of Istio:
Authentication, Traffic Encryption





Traditionally, in microservice pattern application, servers manage their own certificates, and subsequently also maintains its own authentication framework

Istio can unify this process by offering its authentication module, and using the Envoy sidecar proxy to de/encrypt traffic between the pods

Istio tunnels service-to-service communication through the client side and server side Envoy proxies. For a client to call a server, the steps followed are:

1. Istio re-routes the outbound traffic from a client to the client's local sidecar Envoy.
2. The client side Envoy starts a mutual TLS handshake with the server side Envoy. During the handshake, the client side Envoy also does a secure naming check to verify that the service account presented in the server certificate is authorized to run the target service.
3. The client side Envoy and the server side Envoy establish a mutual TLS connection, and Istio forwards the traffic from the client side Envoy to the server side Envoy.