HIGH PERFORMANCE DATA BACKEND FOR YOUR AI/ML

With MinIO and Cilium Cluster Mesh

ecco®

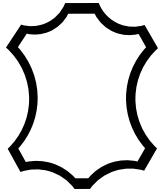
Vitaliy Savistkii

Senior Software Engineer,
Data & AI Platform
ECCO Sko A/S

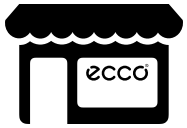
George Zubrienko

Lead Software Engineer,
Data & AI Platform
ECCO Sko A/S

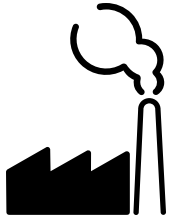
ecco®



4 tanneries



2 066 stores



6 factories

30 shoes sold per minute

90 countries

24 000 employees

ecco®

ECCO DATA & AI

OMNI-CHANNEL FULFILLMENT:
Realtime fulfillment of e-commerce orders

RETURN OPTIMIZATION:
Realtime return of e-commerce orders

MARKDOWN OPTIMIZATION:
End-of-season discount optimization

INTELLIGENT AUTO-REPLENISHMENT:
In-season fulfillment of stores and warehouses

The **Right Product** in the
Right Place at the **Right Time**

CHALLENGES

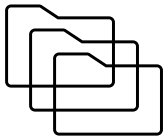
Complex inputs

Data ages **rapidly**

Distributed datasets ***require lots of LIST, GET, HEAD operations***

Data locality is not guaranteed

MORE CHALLENGES



Partitioning improves MERGE performance

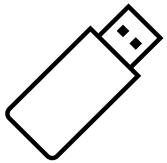
As a side-effect, we get **more files in the table**

ecco®

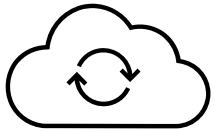
Storage Cost grows **faster** than **Your Business**

ecco®

SECURITY CONSIDERATIONS

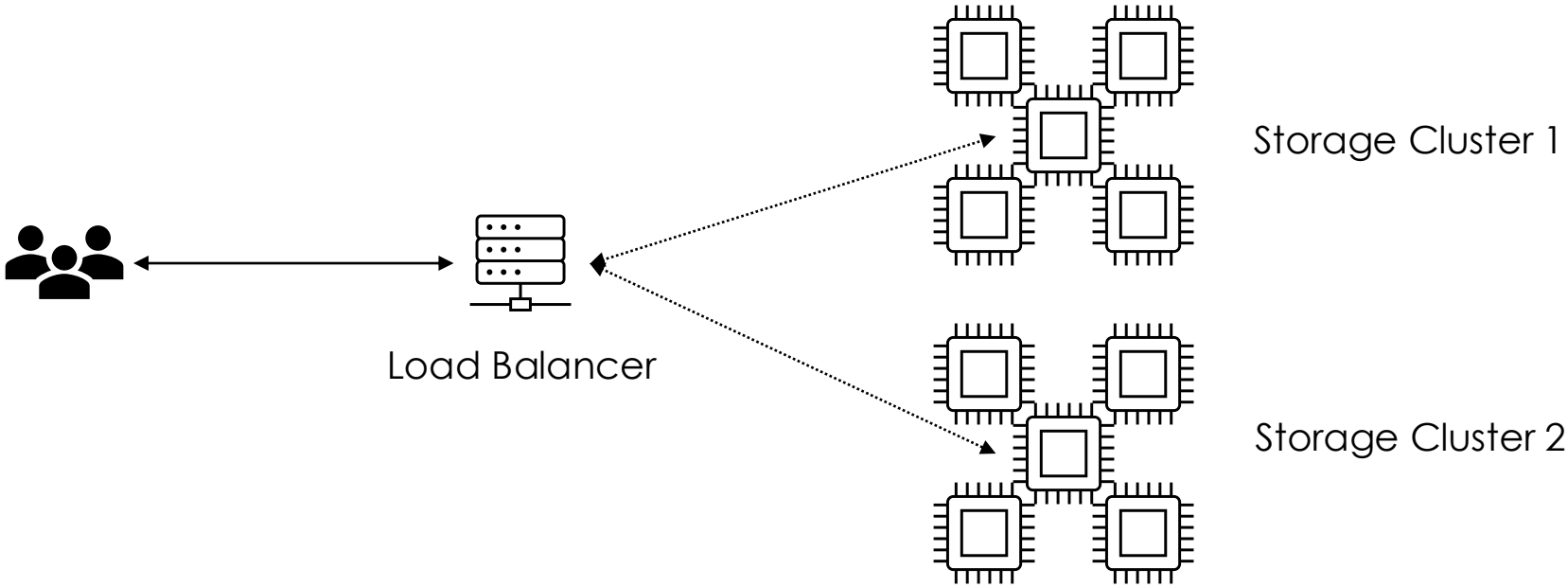


Compute is portable



Object Storage is not

SIMPLIFIED GENERAL OBJECT STORAGE



ALTERNATIVE SOLUTION - STORAGE

Let's host Object Storage in Kubernetes!

REACHING THE STORAGE

Lattice

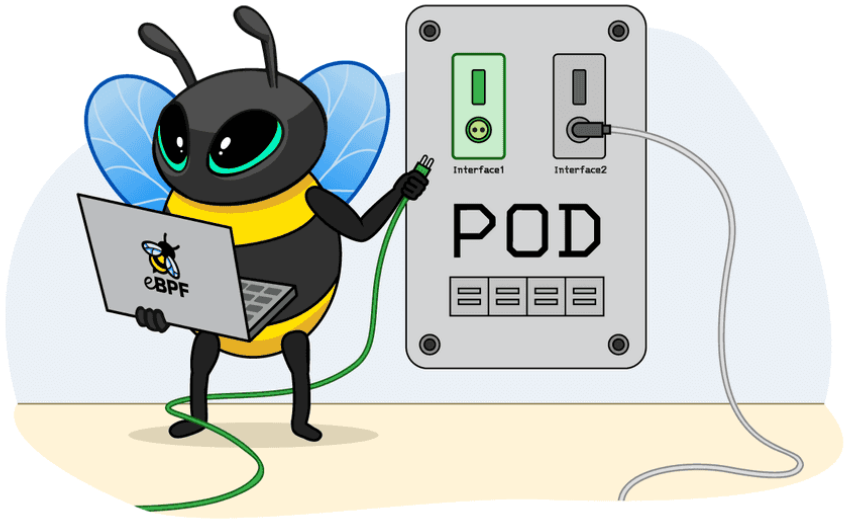
Load Balancer

NATS

ecco

ALTERNATIVE SOLUTION - NETWORKING

Node-to-Node Connections between Apps and Storage



COMPONENTS

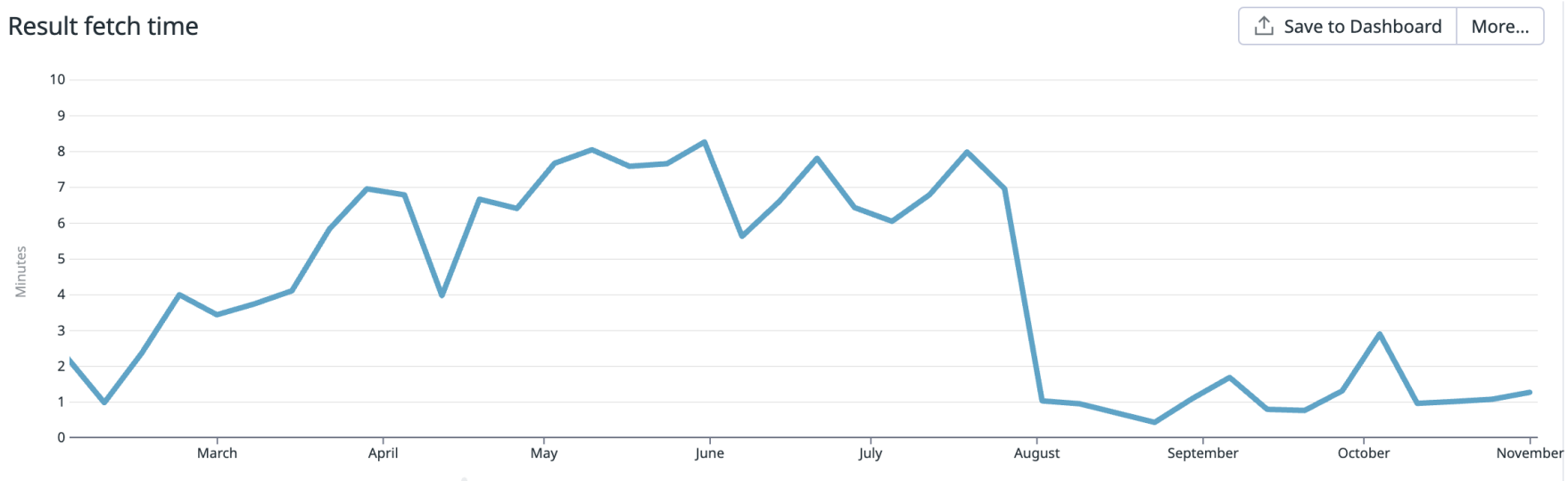
S3-Compatible Object Storage

eBPF-powered Container Network Interface

ecco®

PERFORMANCE GAINS

Intelligent Auto Replenishment – Order Exporter Data Fetch time



PERFORMANCE GAINS – OVER LB

AWS S3 List – 9000 files

```
Sat Nov 02 2024 12:54 pm ~ (34.253s)
hyperfine --runs 10 'aws s3 ls s3://[redacted]/train/ | wc -l'

Benchmark 1: aws s3 ls s3://product-master-data/train/ | wc -l
  Time (mean ± σ):      3.356 s ±  0.631 s    [User: 1.794 s, System: 0.197 s]
  Range (min … max):    2.902 s …  4.591 s    10 runs
```

Data Bolt S3 List – 9000 files

```
Sat Nov 02 2024 12:57 pm ~ (9.862s)
hyperfine --runs 10 'mc ls data-bolt/tmp/train-images/train-images | wc -l'

Benchmark 1: mc ls data-bolt/tmp/train-images/train-images | wc -l
  Time (mean ± σ):     917.9 ms ±  40.3 ms    [User: 283.8 ms, System: 99.4 ms]
  Range (min … max):   856.4 ms … 998.2 ms    10 runs
```


ecco

PERFORMANCE GAINS – OVER MESH

AWS S3 List – 9000 files

```
Sat Nov 02 2024 12:54 pm ~ (34.253s)
hyperfine --runs 10 'aws s3 ls s3://[redacted]/train/ | wc -l'

Benchmark 1: aws s3 ls s3://product-master-data/train/ | wc -l
  Time (mean ± σ):      3.356 s ±  0.631 s    [User: 1.794 s, System: 0.197 s]
  Range (min … max):    2.902 s …  4.591 s    10 runs
```

Data Bolt S3 List – 9000 files

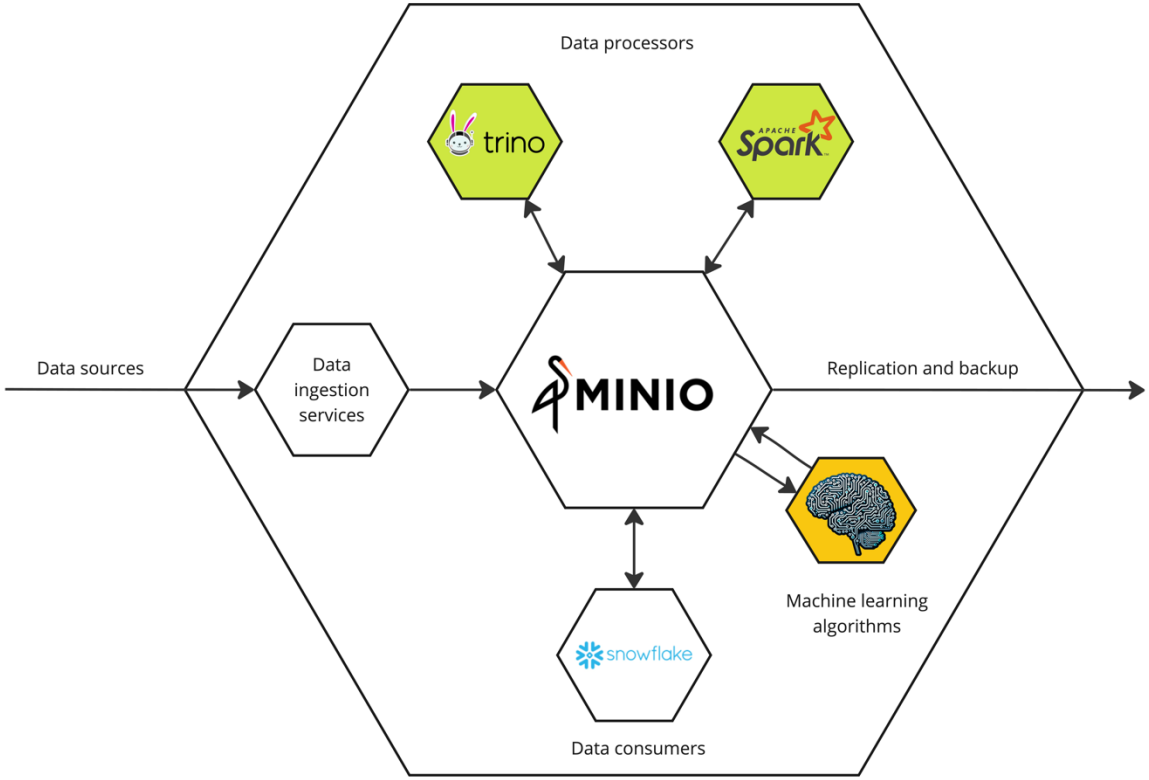
```
root@benchmark1:/# hyperfine --runs 10 'mc ls data-bolt/tmp/train-images/train-images | wc -l'
Benchmark 1: mc ls data-bolt/tmp/train-images/train-images | wc -l
  Time (mean ± σ):     637.7 ms ±  32.2 ms    [User: 380.9 ms, System: 82.6 ms]
  Range (min … max):   580.1 ms … 681.6 ms    10 runs
```

ASSEMBLING THE SYSTEM

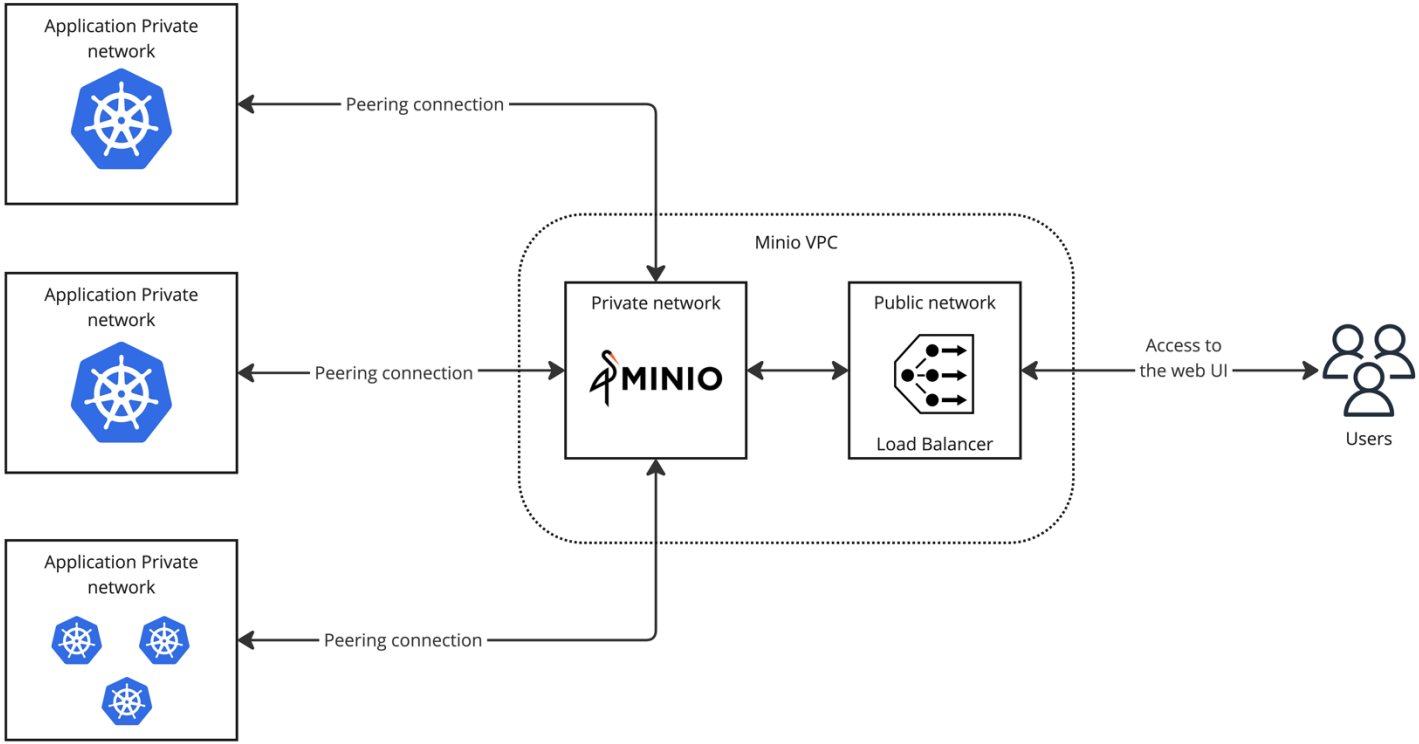
Connecting Storage and Applications

ecco®

TOP LEVEL ARCHITECTURE



NETWORK LAYOUT



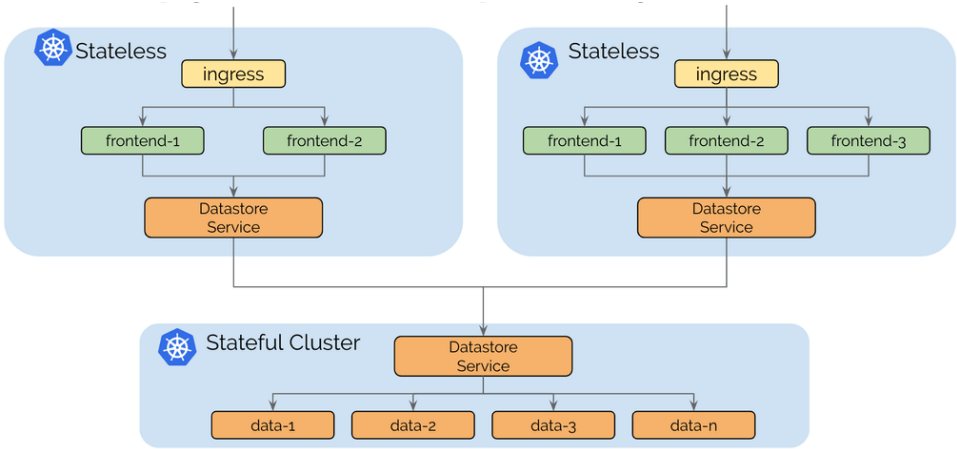
CLUSTER MESH

Use case: direct pod-to-pod communication

Bypass internal load balancer

Transparent service discovery

Network disruption tolerance

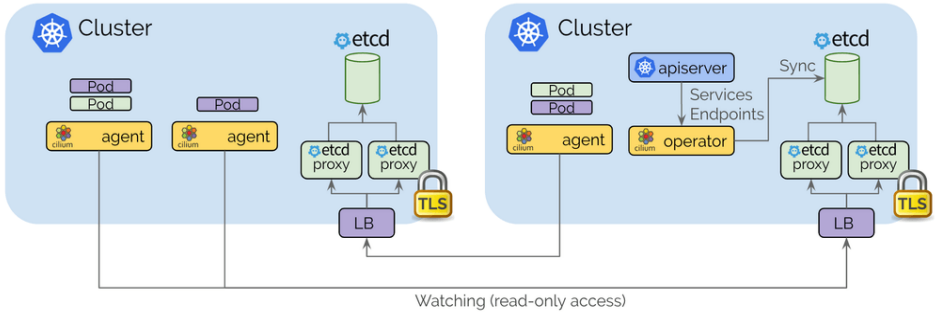


PERSISTENT MESH

Multicluster mesh architecture

Agent ⇔ etcd proxy communicate via internal LB

Pod ⇔ pod communicate directly

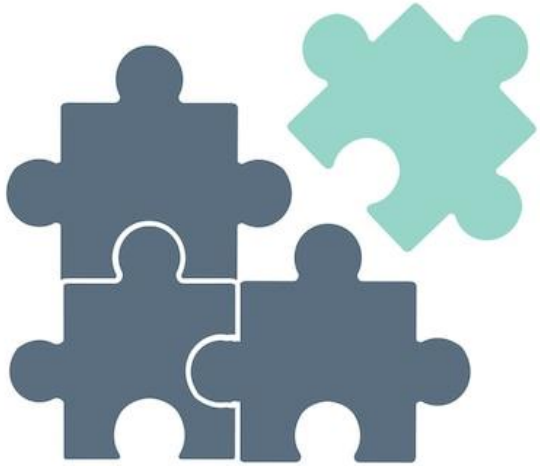


ENABLE MESH IN CILIUM

Private DNS

Load Balancer

TLS certificates exchange



SETTING UP DNS

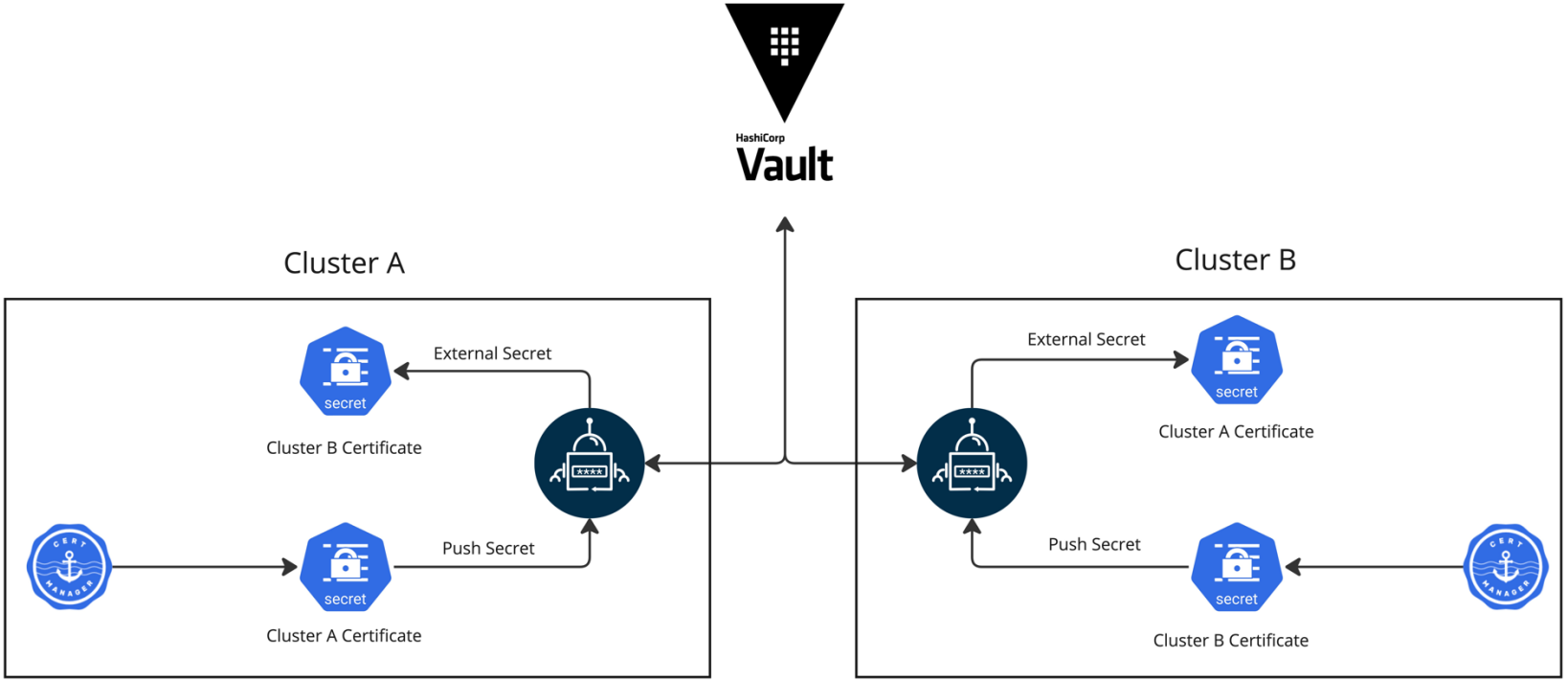
Private DNS zone created with Amazon Route 53

Each load balancer has a DNS name with the following policy

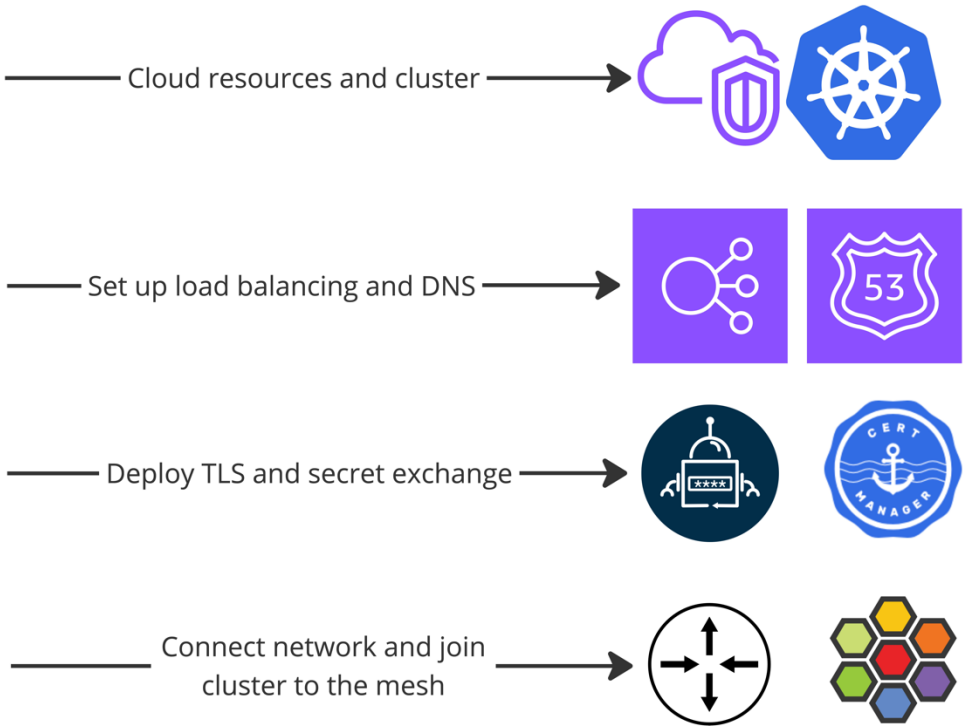
${environment}.${application}.cluster-mesh.sneaksanddata.internal

☐	production-1.airflow.cluster-mesh.sneaksanddata.internal	AAAA	Simple	-	Yes
☐	production-0.arcane.cluster-mesh.sneaksanddata.internal	AAAA	Simple	-	Yes
☐	production-0.dev-spaces.cluster-mesh.sneaksanddata.internal	AAAA	Simple	-	Yes
☐	s3.cluster-mesh.sneaksanddata.internal	AAAA	Simple	-	Yes
☐	batch-1.spark.cluster-mesh.sneaksanddata.internal	AAAA	Simple	-	Yes
☐	databuild-1.spark.cluster-mesh.sneaksanddata.internal	AAAA	Simple	-	Yes
☐	streaming-1.spark.cluster-mesh.sneaksanddata.internal	AAAA	Simple	-	Yes
☐	production-0.trino.cluster-mesh.sneaksanddata.internal	AAAA	Simple	-	Yes

TLS SECRET EXCHANGE WITH VAULT



ADDING NEW CLUSTERS TO THE MESH



PEEK AT THE CODE

```
module "cni" {
  source = "../../.modules/cilium-cni"

  cilium = {
    hubble       = false
    cluster_mesh = {
      name           = local.cluster_full_name
      id             = module.mesh_id.cluster_id
      tls_issuer_name = "${local.cluster_name}-self-signed-ca"
      tls_dns_names   = [
        "${terraform.workspace}.spark.cluster-mesh.sneaksanddata.internal”
      ]
    }
    installation = {
      aws_vpc_cni = {}
    }
  }
}
```

PEEK AT THE CODE

```
module "mesh_config" {
  source = "../../../.modules/cilium-clustermesh-secret-exchange"

  cluster = {
    name               = local.cluster_full_name
    clustermesh_name   = local.cluster_full_name
    alias              = local.cluster_name
    clustermesh_endpoint = "${terraform.workspace}.trino.cluster-mesh.sneaksanddata.internal"
  }
  secrets = {
    vault_address      = local.vault.address
    push_to_address    =
"/secret/data/clusters/${local.application_name}/tls/${terraform.workspace}/infrastructure/clustermesh"
    pull_from_addresses = local.connected_clusters
  }
}
```

CONFIGURE THE CLIENT

Application Cluster

```yaml
apiVersion: v1
kind: Service
metadata:
  name: s3-internal
  namespace: minio-tenant
  annotations:
    service.cilium.io/global: 'true'
    service.cilium.io/shared: 'false'
spec:
  ports:
    - name: http-minio
      protocol: TCP
      port: 80
      targetPort: 9000
```

Storage Cluster

```yaml
apiVersion: v1
kind: Service
metadata:
  name: s3-internal
  namespace: minio-tenant
  annotations:
    service.cilium.io/global: 'true'
    service.cilium.io/shared: 'true'
spec:
  ports:
    - name: http-minio
      protocol: TCP
      port: 80
      targetPort: 9000
```

```
$ curl -v http://s3-internal.minio-tenant.svc.cluster.local
*   Trying [fdac:52bd:490e::1c53]:80...
* Connected to s3-internal.minio-tenant.svc.cluster.local (fdac:52bd:490e::1c53) port 80 (#0)
> GET / HTTP/1.1
> Host: s3-internal.minio-tenant.svc.cluster.local
```

MAINTENANCE

Troubleshoot

cilium-agent cilium-dbg troubleshoot clustermesh

Status

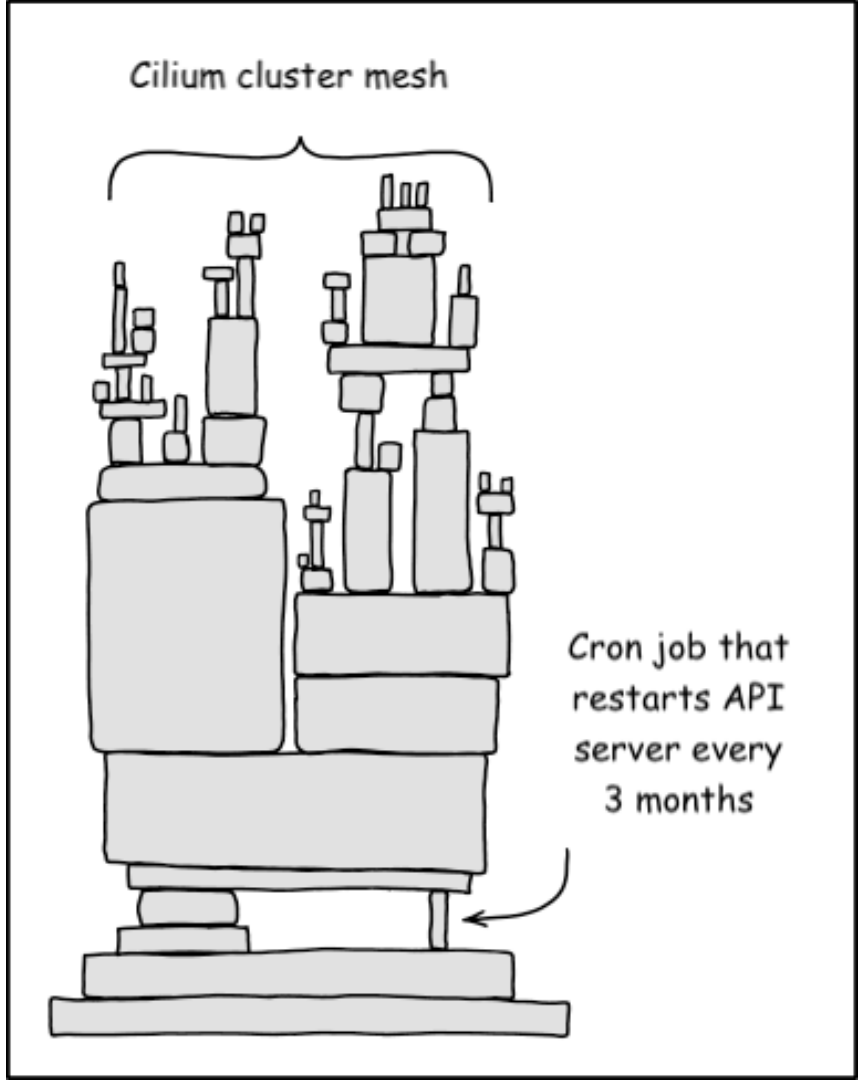
cilium clustermesh status

```
Hostname based ingress detected, trying to resolve it
Hostname resolved, using the found ip(s)
✅ Service "clustermesh-apiserver" of type "LoadBalancer" found
✅ Cluster access information is available:
  - 10.5.82.4:2379
  - 2a05:d014:1c62:927f:23ea:1b66:4c89:bde2:2379
✅ Deployment clustermesh-apiserver is ready
ℹ️ KVStoreMesh is disabled

✅ All 7 nodes are connected to all clusters [min:1 / avg:1.0 / max:1]

🔌 Cluster Connections:
  - data-bolt: 7/7 configured, 7/7 connected

🔀 Global services: [ min:2 / avg:2.0 / max:2 ]
```



CODE AND REFERENCES

Terraform configuration example

https://gist.github.com/s-vitaliy/2c56adc8020275eae106b6bd5cb4e76b

Cilium website: https://cilium.io/

Cilium cluster mesh: https://cilium.io/use-cases/cluster-mesh/

THANK YOU