Automated Disaster Recovery for Multi-Region GenAI applications using vector databases on Kubernetes

Sergey Pronin
Group Product Manager

Paweł Bojanowski
Principal Software Engineer

Agenda

1. Intro - “AI is magic”
2. Problem space
    a. pgvector on Kubernetes 101
    b. Multi-region deployments
    c. Disaster Recovery (DR) problems
3. Automating DR
4. Demo
5. Q&A

AI is Magic

Creativity and imagination

Pretty kitten working behind the laptop

Unicorn plays hockey

Dad became a monster, cause we broke his Lego

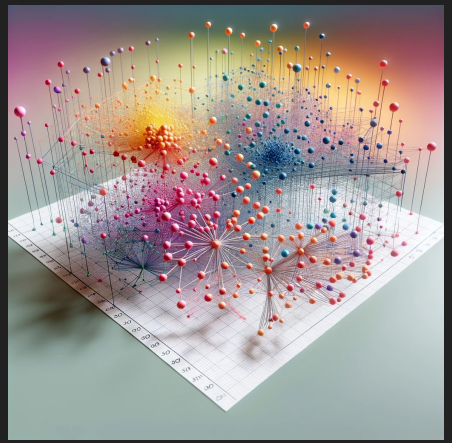
PERCONA ELOTL

AI is magic, but for Muggles

Various technologies power AI:

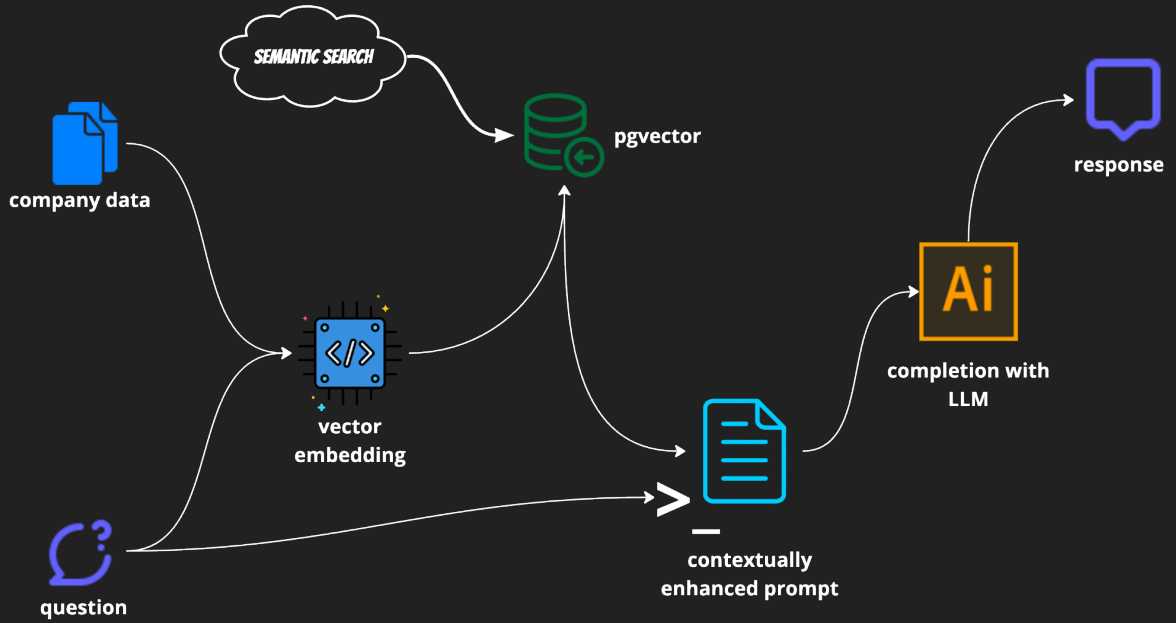
- Large Language Models - heart of modern Generative AI
  - Powered by Transformers (Attention is All You Need)
- GPUs for training
- Vector databases for semantic search, visual or multimodal search
  - Through vector embeddings

Everything is a vector



This is how DALL-E sees vector embedding

Example of RAG - Retrieval Augmented Generation



PERCONA ELOTL

Vector databases

Various options

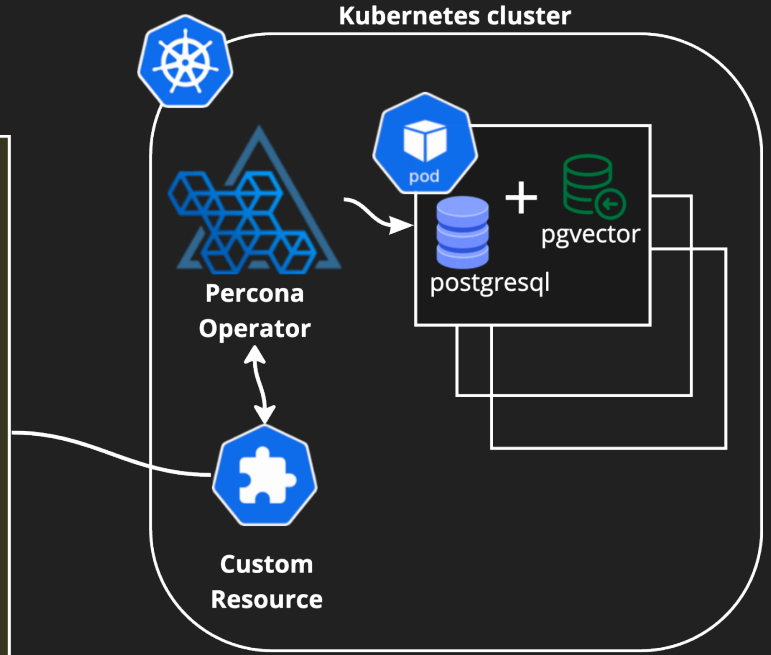
- Pinecone
- Milvus
- Chroma
- FAISS
- pgvector (extension for PostgreSQL)

Problem space

pgvector on Kubernetes

YAML manifest

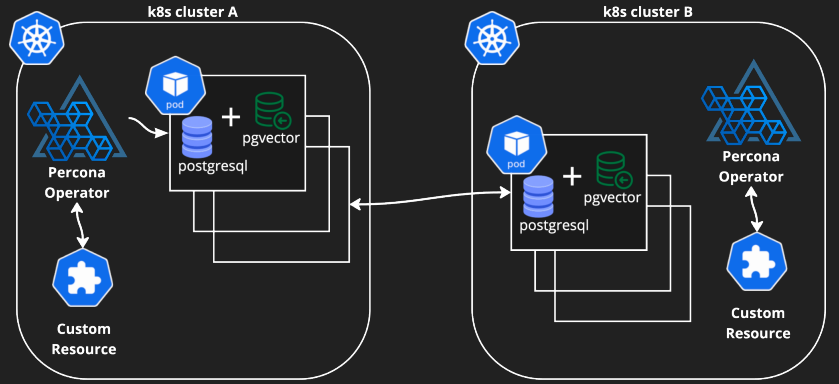
```
apiVersion:
pgv2.percona.com/v2
kind: PerconaPGCluster
metadata:
  name: my-pg
spec:
  instances:
    - name: pgvector1
      replicas: 3
...
extensions:
  custom:
  - name: pgvector
    version: 0.5.1
```



PERCONA ELOTL

Why multi-region

- Not only multi-region:
  - multi-cloud
  - hybrid-cloud
- Main reasons
  - Disaster recovery
  - Migration
  - No-vendor lock-in / cloud-ready



Disaster recovery

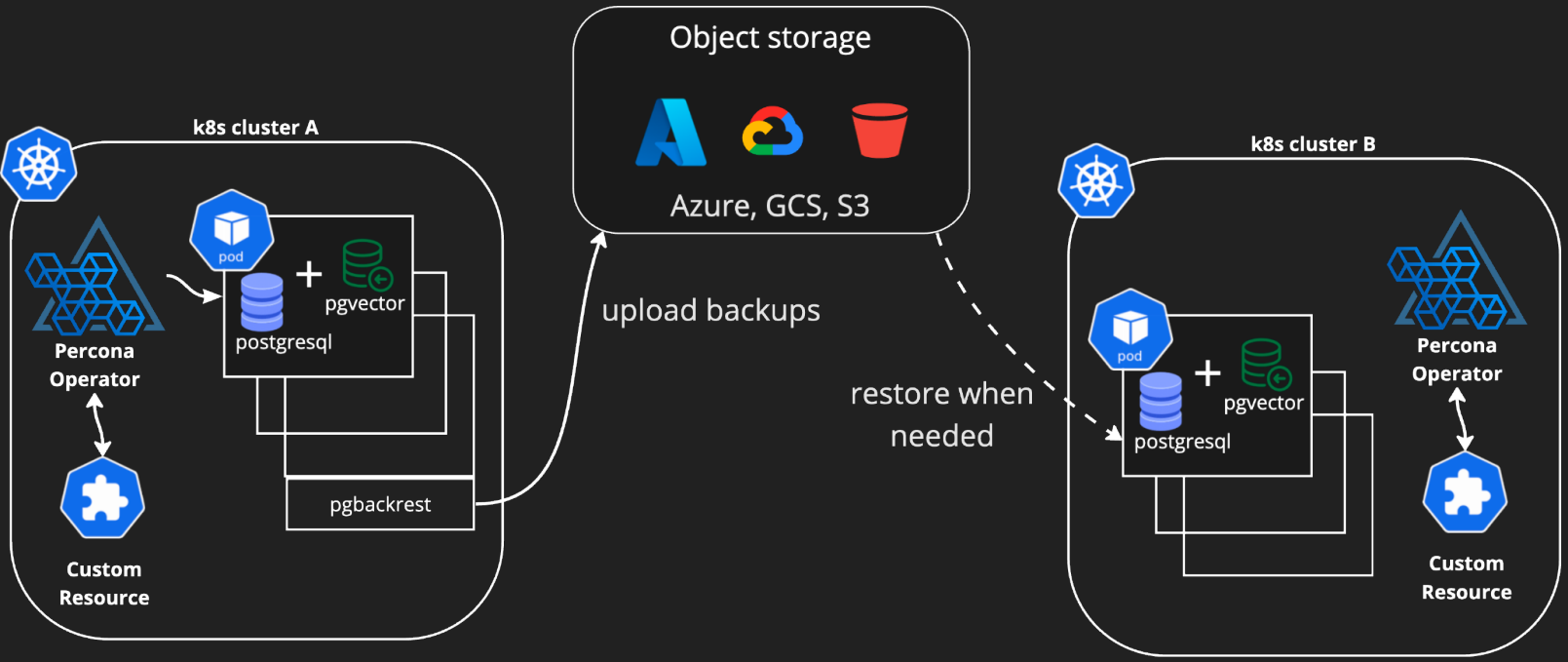
Recovery Time Objective (RTO)

- How fast do you recover from failure?

Recovery Point Objective (RPO)

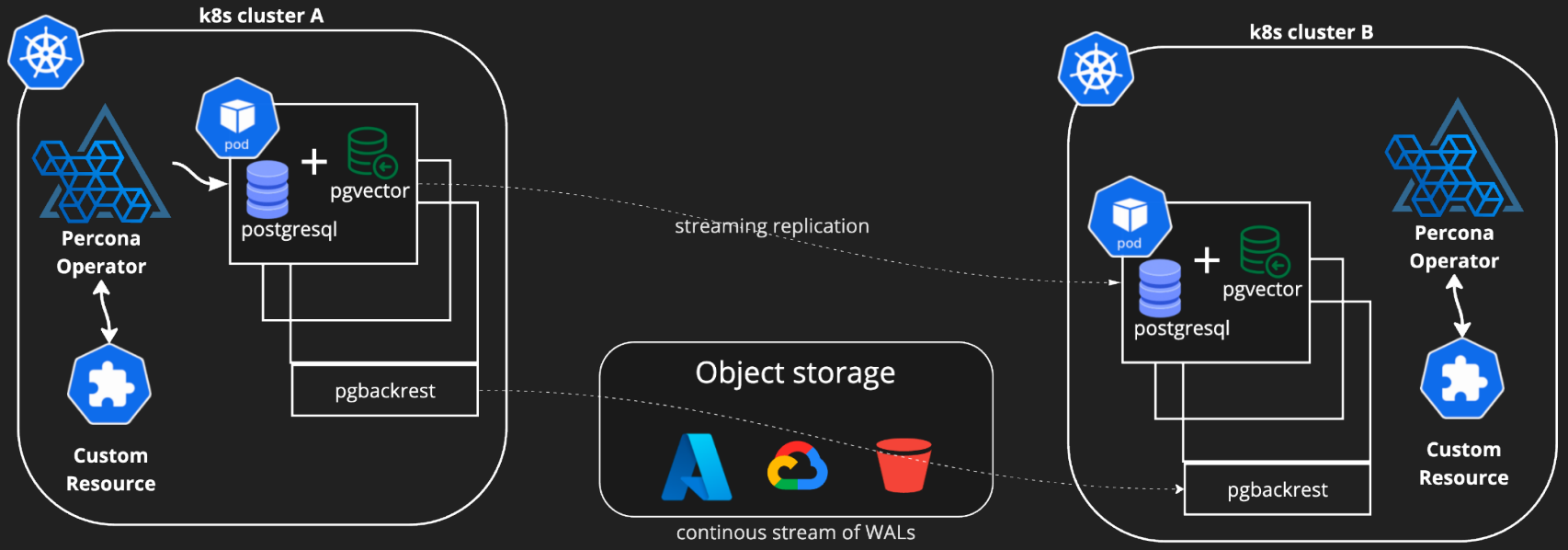
- What is your last transaction?

DR through Backups - high RTO



PERCONA ELOTL

DR through Replication - low RTO



PERCONA ELOTL

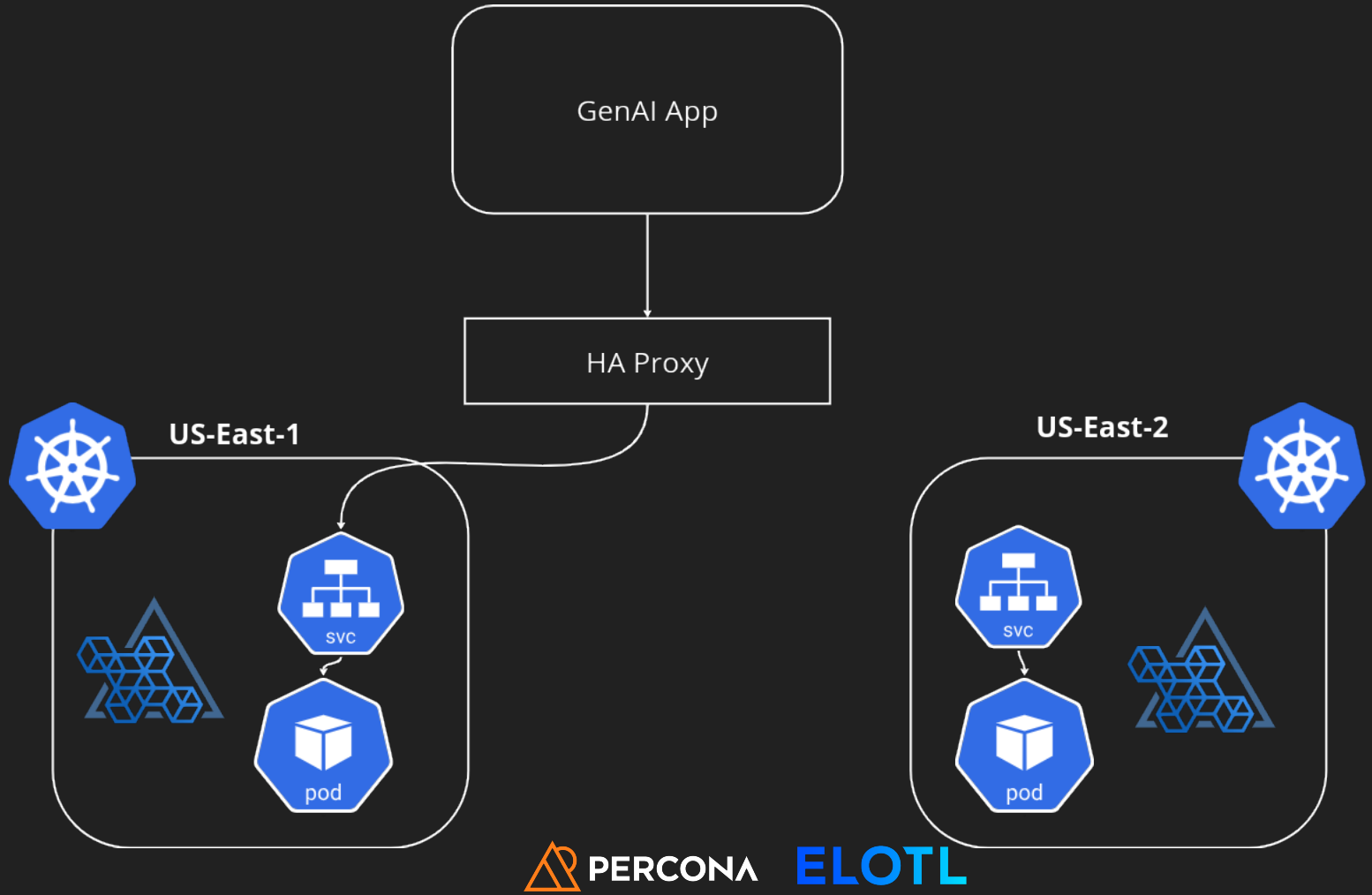
Looks like problem solved?

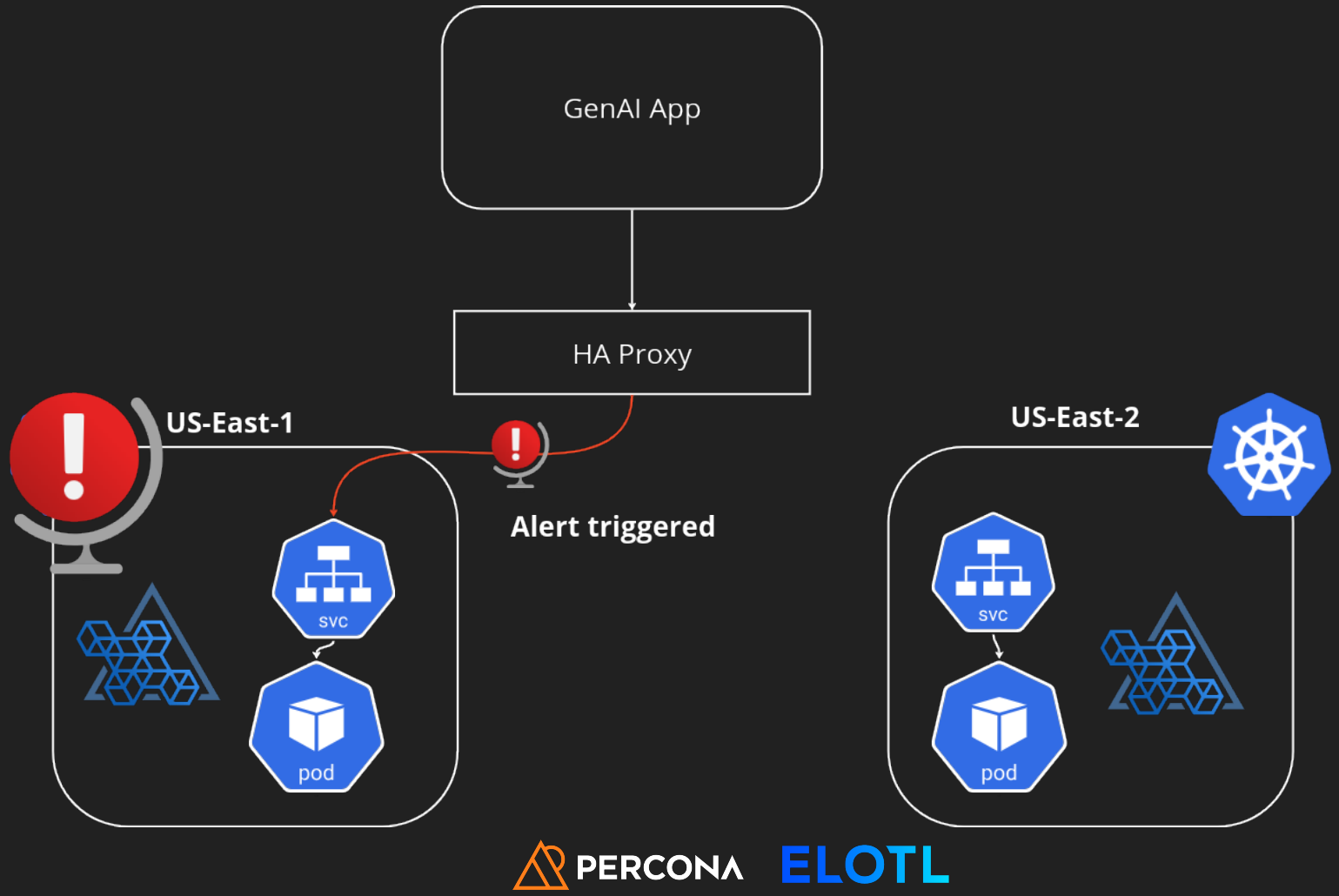
1. Recover fast from failure
2. Recovery Point Objective is solved on DB level
    a. For streaming replication it can be down to almost 0

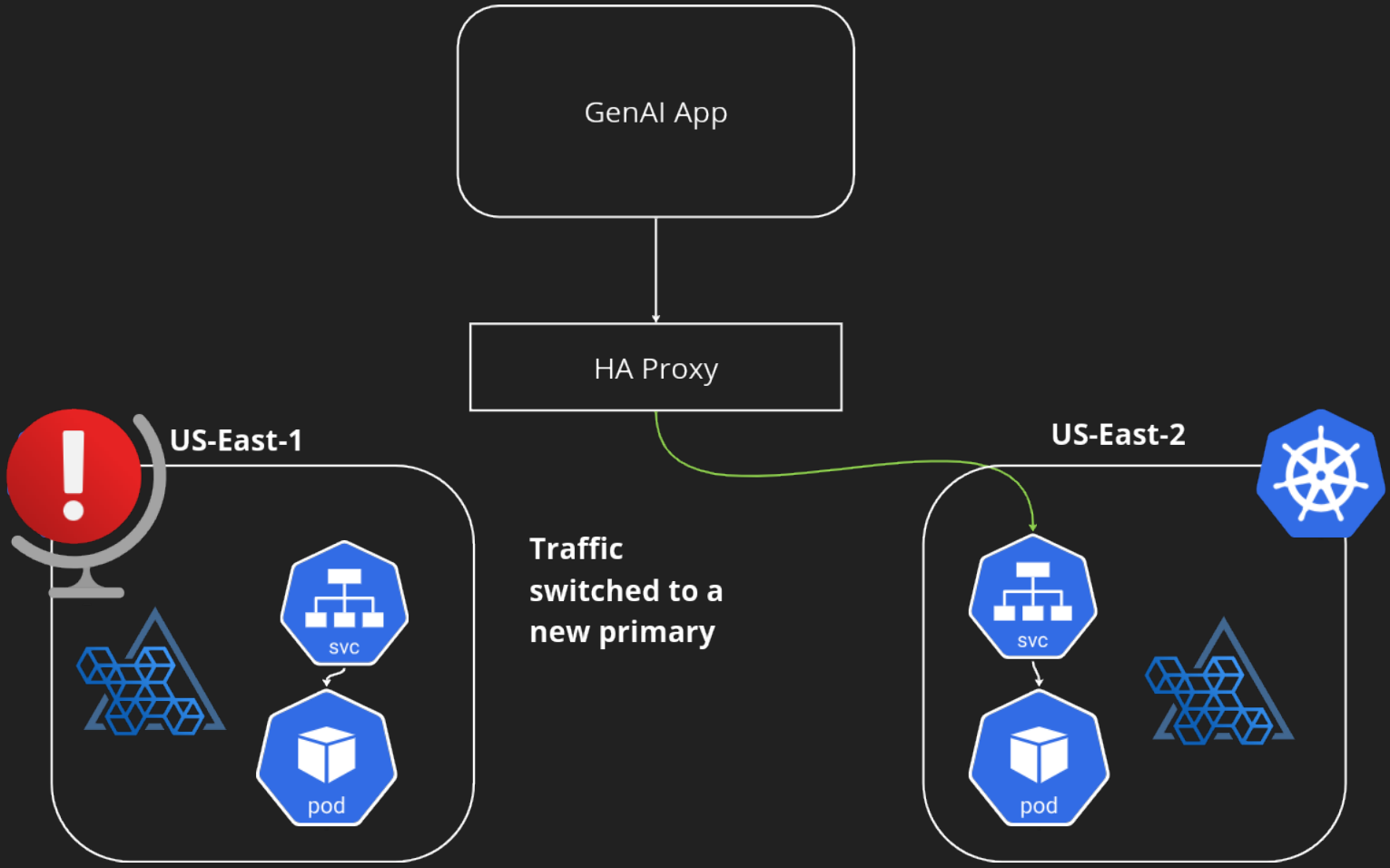
What about the app and failover?

1. Failover is not automated
    - Operators can detect failure, but 3rd agent needed (quorum)
2. Point application to correct database
    - Which endpoint should the app use when failover happens?
    - Who is reconfiguring it and how?
3. Failback
    - What should happen when main site goes back up?

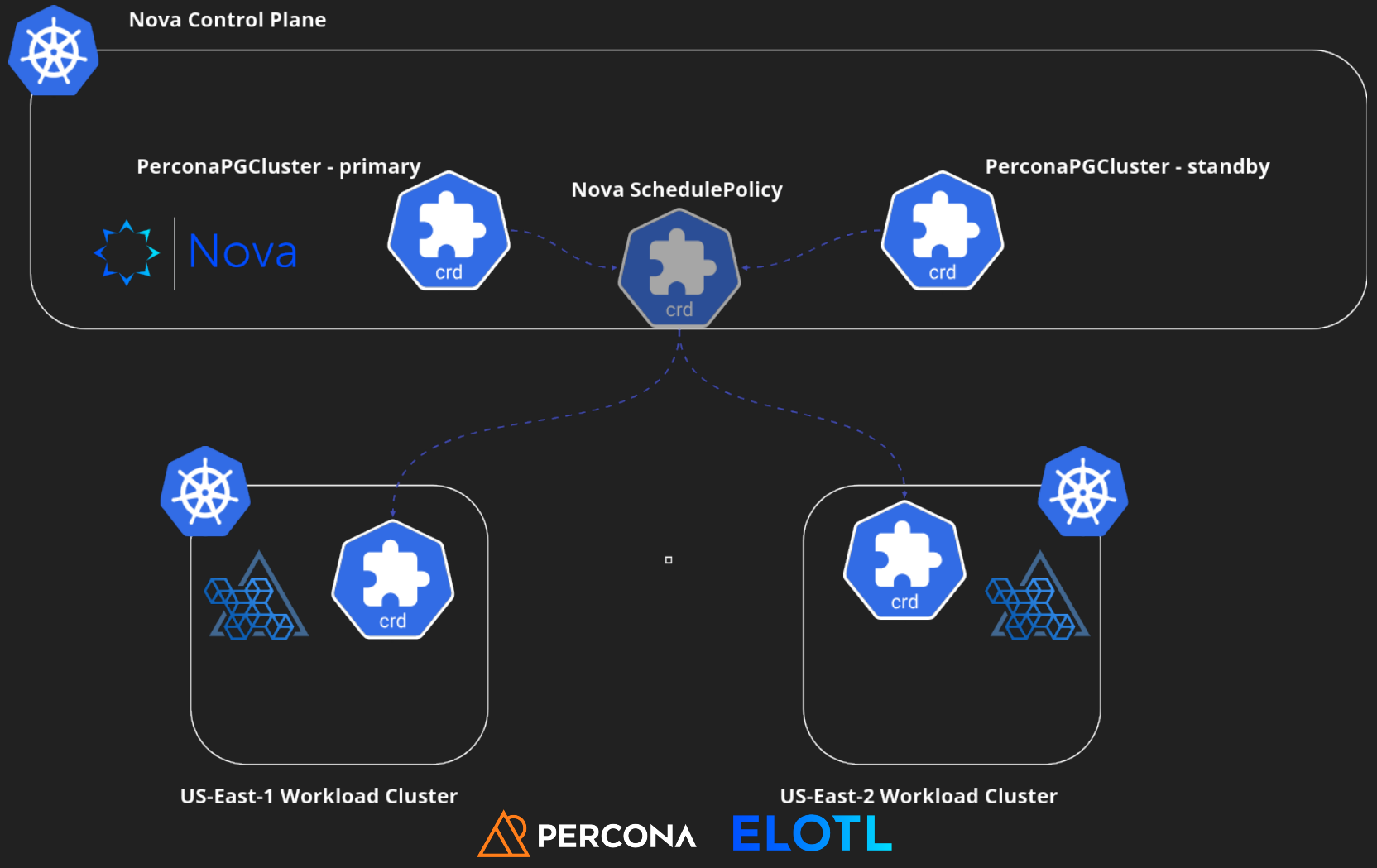
💡 Automating DR

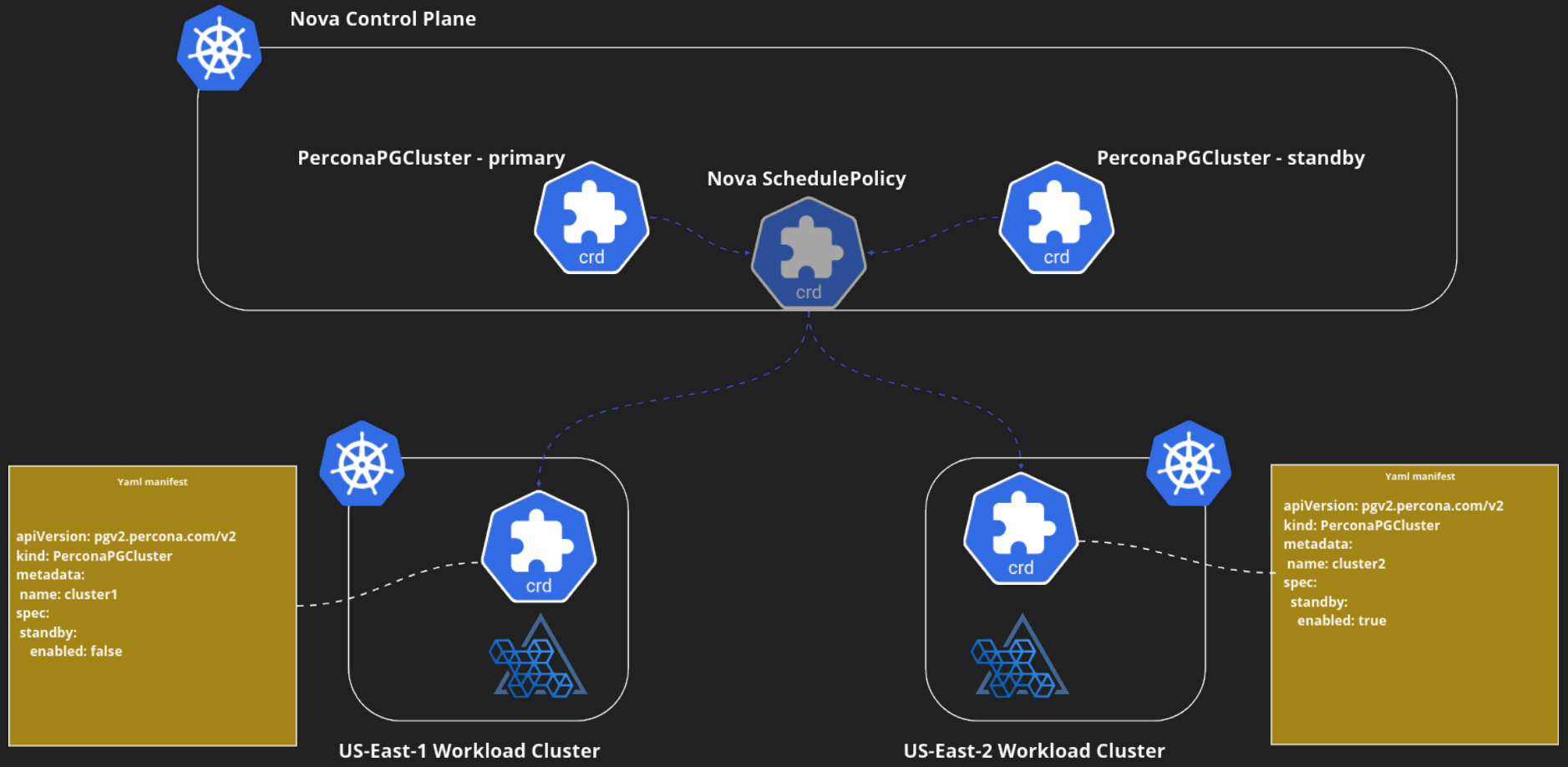

GenAI App
HA Proxy
US-East-1
svc
pod
US-East-2
svc
pod
PERCONA
ELOTL


GenAI App
HA Proxy
US-East-1
Alert triggered
US-East-2
svc
pod
svc
pod
PERCONA
ELOTL

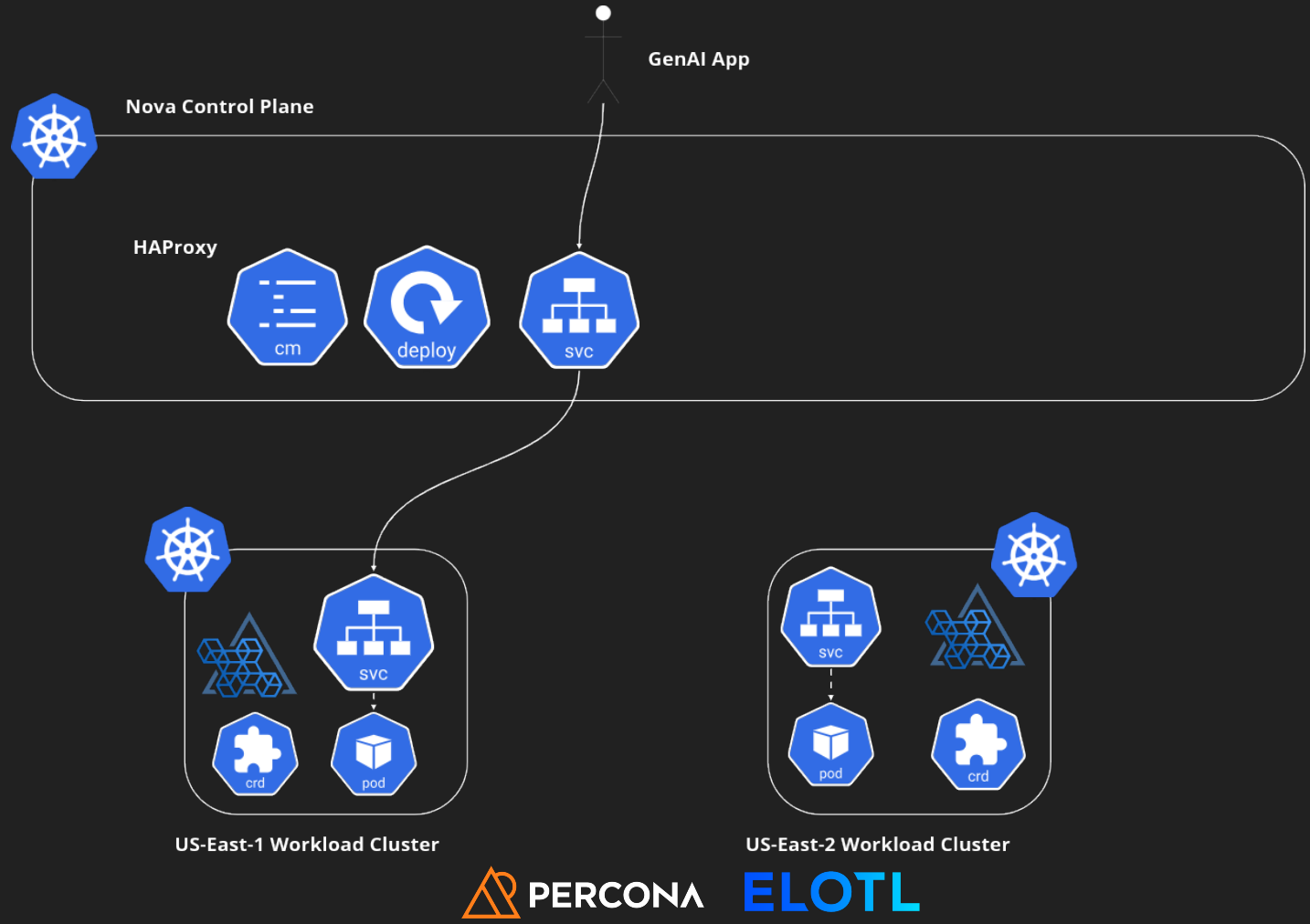


PERCONA ELOTL

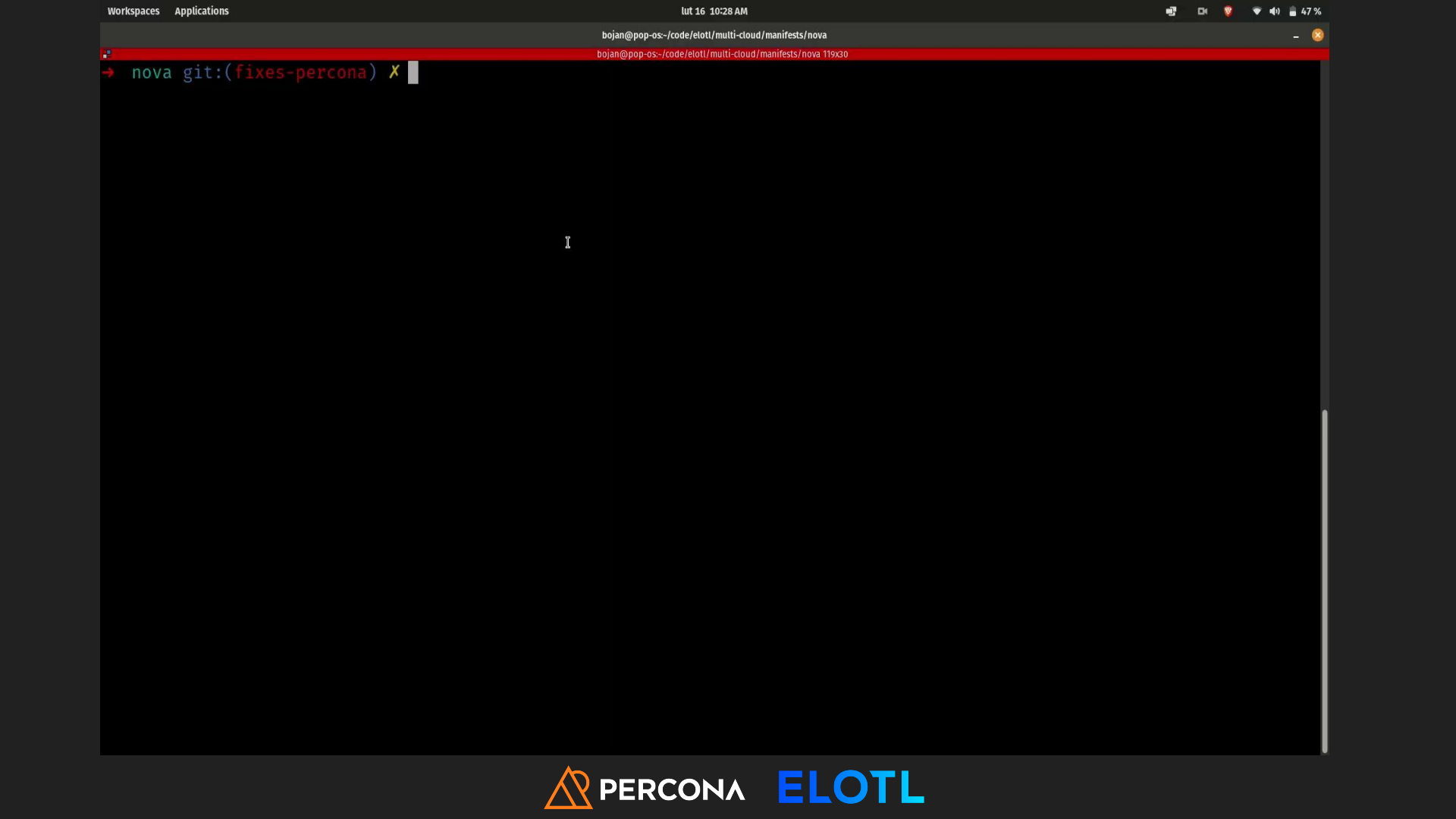

Nova Control Plane
PerconaPGCluster - primary
Nova
crd
Nova SchedulePolicy
crd
PerconaPGCluster - standby
crd
crd
crd
US-East-1 Workload Cluster
US-East-2 Workload Cluster
PERCONA
ELOTL



PERCONA ELOTL



PERCONA ELOTL


Workspaces
Applications
lut 16 10:28 AM
47 %
bojan@pop-os:~/code/elotl/multi-cloud/manifests/nova
bojan@pop-os:~/code/elotl/multi-cloud/manifests/nova 119x30
nova git:(fixes-percona) ✗
PERCONA
ELOTL

```yaml
apiVersion: recovery.elotl.co/v1alpha1
kind: RecoveryPlan
metadata:
 name: psql-primary-failover-plan
spec:
 alertLabels:
  app: example-app
 steps:

  - type: patch
    patch:
      apiVersion: "pgv2.percona.com/v2"
      resource: "perconapgclusters"
      namespace: "psql-operator"
      name: "cluster1"
      override:
        fieldPath: "spec.standby.enabled"
        value:
          raw: false
      patchType: "application/merge-patch+json"

  - type: patch
    patch:
      apiVersion: "pgv2.percona.com/v2"
      resource: "perconapgclusters"
      namespace: "psql-operator"
      name: "cluster2"
      override:
        fieldPath: "spec.standby.enabled"
        value:
          raw: true
      patchType: "application/merge-patch+json"
```

Matches metadata of incoming alert (e.g. from prometheus)

```yaml
groups:
- name: example
  rules:
  - alert: HighRequestLatency
    expr: percona_pg_cluster_failure_total{cluster="cluster-1"} > 0
    for: 10m
    labels:
      app: example-app
```

https://prometheus.io/docs/alerting/latest/configuration/#webhook_config

PERCONA ELOTL

```
apiVersion: recovery.elotl.co/v1alpha1
kind: RecoveryPlan
metadata:
 name: psql-primary-failover-plan
spec:
 alertLabels:
   app: example-app
 steps:

 - type: patch
   patch:
     apiVersion: "pgv2.percona.com/v2"
     resource: "perconapgclusters"
     namespace: "psql-operator"
     name: "cluster1"
     override:
       fieldPath: "spec.standby.enabled"
       value:
         raw: true
     patchType: "application/merge-patch+json"

 - type: patch
   patch:
     apiVersion: "pgv2.percona.com/v2"
     resource: "perconapgclusters"
     namespace: "psql-operator"
     name: "cluster2"
     override:
       fieldPath: "spec.standby.enabled"
       value:
         raw: false
     patchType: "application/merge-patch+json"

 - type: readField
```

Puts PerconaPGCluster primary into standby

PERCONA ELOTL

```yaml
apiVersion: recovery.elotl.co/v1alpha1
kind: RecoveryPlan
metadata:
 name: psql-primary-failover-plan
spec:
  alertLabels:
    app: example-app
  steps:

    - type: patch
      patch:
        apiVersion: "pgv2.percona.com/v2"
        resource: "perconapgclusters"
        namespace: "psql-operator"
        name: "cluster1"
        override:
          fieldPath: "spec.standby.enabled"
          value:
            raw: true
        patchType: "application/merge-patch+json"

    - type: patch
      patch:
        apiVersion: "pgv2.percona.com/v2"
        resource: "perconapgclusters"
        namespace: "psql-operator"
        name: "cluster2"
        override:
          fieldPath: "spec.standby.enabled"
          value:
            raw: false
        patchType: "application/merge-patch+json"

    - type: readField
```

Turns PerconaPGCluster standby into primary

PERCONA ELOTL

```
- type: patch
  patch:
    apiVersion: "pgv2.percona.com/v2"
    resource: "perconapgclusters"
    namespace: "psql-operator"
    name: "cluster1"
    override:
      fieldPath: "spec.standby.enabled"
      value:
        raw: true
    patchType: "application/merge-patch+json"

- type: patch
  patch:
    apiVersion: "pgv2.percona.com/v2"
    resource: "perconapgclusters"
    namespace: "psql-operator"
    name: "cluster2"
    override:
      fieldPath: "spec.standby.enabled"
      value:
        raw: false
    patchType: "application/merge-patch+json"

- type: readField
  readField:
    apiVersion: "pgv2.percona.com/v2"
    resource: "perconapgclusters"
    namespace: "psql-operator"
    name: "cluster2"
    fieldPath: "status.host"
    outputKey: "Cluster2IP"

- patch:
    apiVersion: "v1"
    resource: "configmaps"
    namespace: "default"
```

Reads address of a new primary

PERCONA ELOTL

```
- type: patch
  patch:
    apiVersion: "pgv2.percona.com/v2"
    resource: "perconapgclusters"
    namespace: "psql-operator"
    name: "cluster2"
    override:
      fieldPath: "spec.standby.enabled"
      value:
        raw: false
    patchType: "application/merge-patch+json"

- type: readField
  readField:
    apiVersion: "pgv2.percona.com/v2"
    resource: "perconapgclusters"
    namespace: "psql-operator"
    name: "cluster2"
    fieldPath: "status.host"
    outputKey: "Cluster2IP"
```

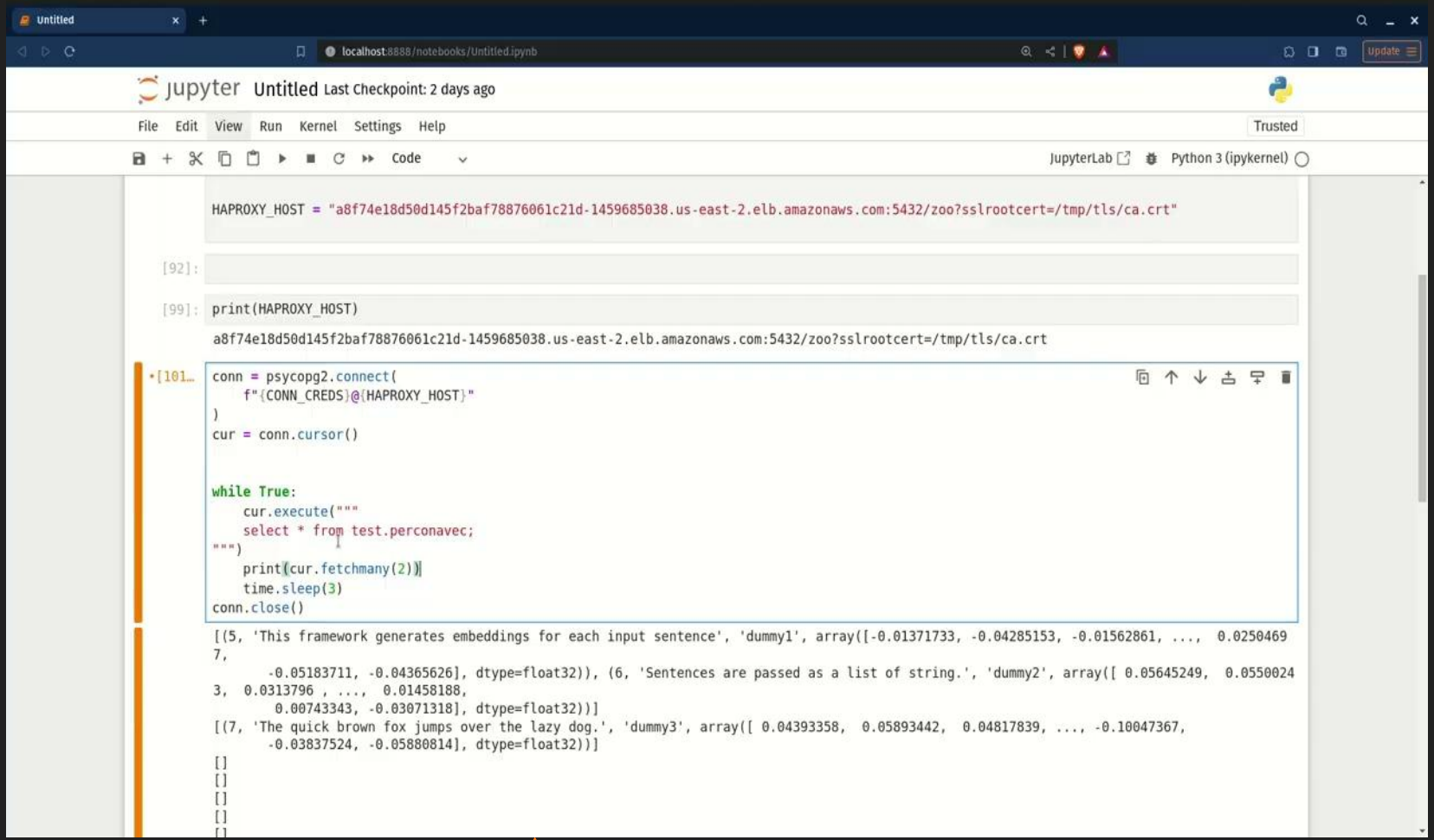
Updates HAProxy configuration with an address of a new primary

```
- patch:
    apiVersion: "v1"
    resource: "configmaps"
    namespace: "default"
    name: "haproxy-config"
    override:
      fieldPath: "data"
      value:
        raw: {"haproxy.cfg": "defaults\n    mode tcp\n    timeout connect 5000ms\n    timeout client 50000ms\n    timeout server
50000ms\n\nfrontend fe_main\n    bind *:5432\n    default_backend be_db_2\n\nbackend be_db_2\n    server db2 {{ .Values.Cluster2IP
}}:5432 check"}

    patchType: "application/merge-patch+json"
  type: patch
```

PERCONA ELOTL

Demo



PERCONA ELOTL

Nova’s secret sauce: Schedule Policies


```
apiVersion: pgv2.percona.com/v2
kind: PerconaPGCluster
metadata:
 name: cluster1
 namespace: psql-operator
 labels:
   psql-cluster: cluster-1
spec:
  ...
```

```
apiVersion: policy.elotl.co/v1alpha1
kind: SchedulePolicy
metadata:
 name: psql-cluster-1
spec:
 namespaceSelector:
  matchLabels:
   kubernetes.io/metadata.name: psql-operator
 clusterSelector:
  matchLabels:
    kubernetes.io/metadata.name: us-east-1-wlc
 resourceSelectors:
  labelSelectors:
   - matchLabels:
      psql-cluster: cluster-1
```

PERCONA ELOTL


```
kind: CustomResourceDefinition
metadata:
 labels:
   psql-cluster: all
names:
  kind: PerconaPGBackup
  ...
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  labels:
    psql-cluster: all
names:
  kind: PerconaPGCluster
  ...
---
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    psql-cluster: all
    ...
```

```
apiVersion: policy.elotl.co/v1alpha1
kind: SchedulePolicy
metadata:
 name: psql-all-clusters
spec:
  namespaceSelector:
    matchLabels:
      kubernetes.io/metadata.name: psql-operator
  groupBy:
    labelKey: psql-cluster
  clusterSelector:
    matchExpressions:
      - key: kubernetes.io/metadata.name
        operator: In
        values:
          - us-east-1-wlc
          - us-east-2-wlc
  spreadConstraints:
    spreadMode: Duplicate
    topologyKey: kubernetes.io/metadata.name
  resourceSelectors:
    labelSelectors:
      - matchLabels:
          psql-cluster: all
```

PERCONA ELOTL

Thank you!

Learn more:

Percona Operators

Elotl Nova

PERCONA ELOTL