CON309

Building PaaS with EKS for large-scale highly regulated enterprise

Madhuri Peri
Sr. IoT Solution Architect
Amazon Web Services

Amr Abdelhalem
Head of Cloud Architecture
Fidelity Investment

AWS re:Invent

aws

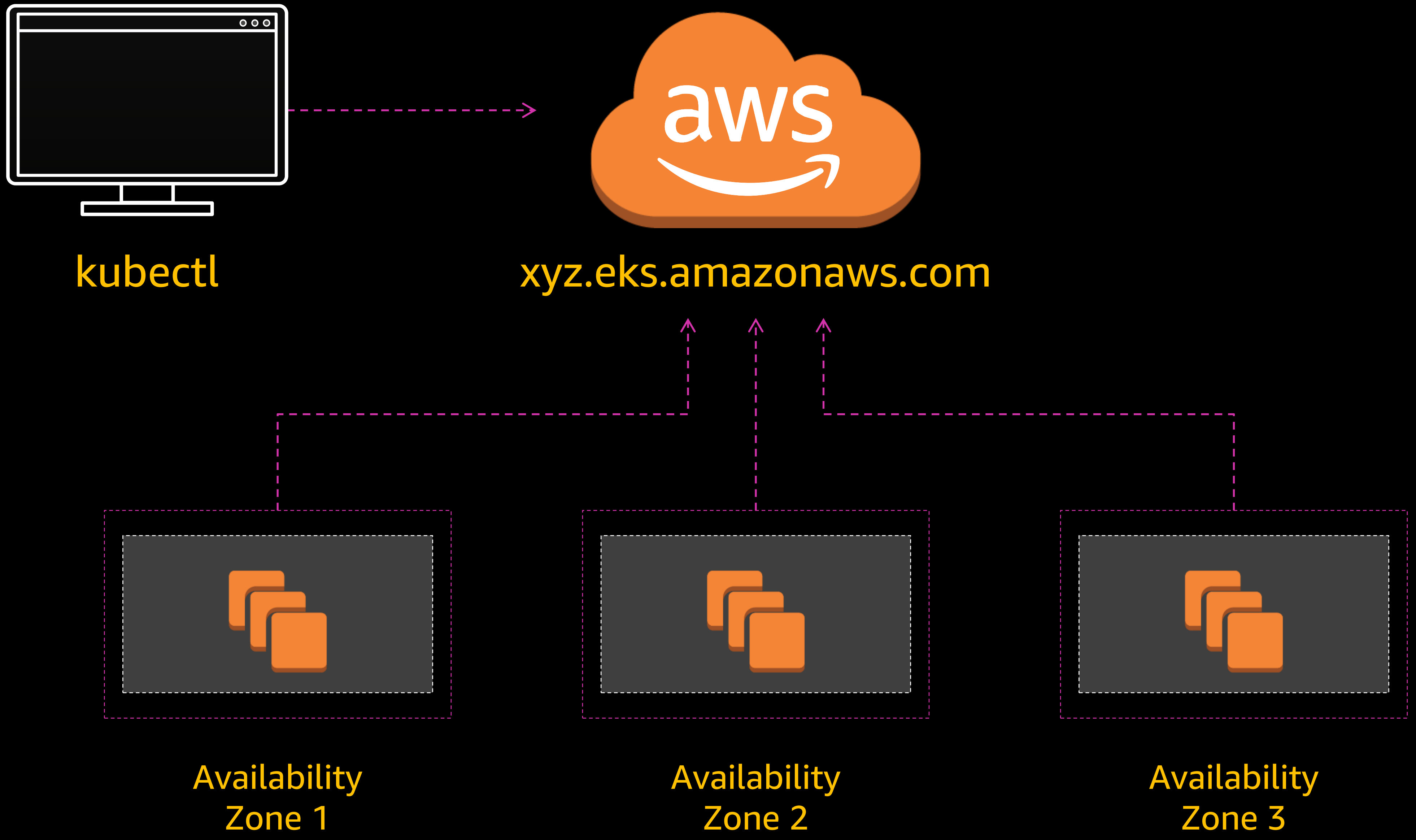
Agenda

- Amazon EKS – Concepts overview
    - Building blocks AWS Cloud offers
    - How Fidelity used these services to achieve business needs maintaining compliance
- Fidelity – PaaS strategy & regulatory guidelines
- Fidelity
    - Amazon EKS integration Security
    - Amazon EKS integration DevOps & Helm
    - Amazon EKS integration Monitoring
- Future & Next steps

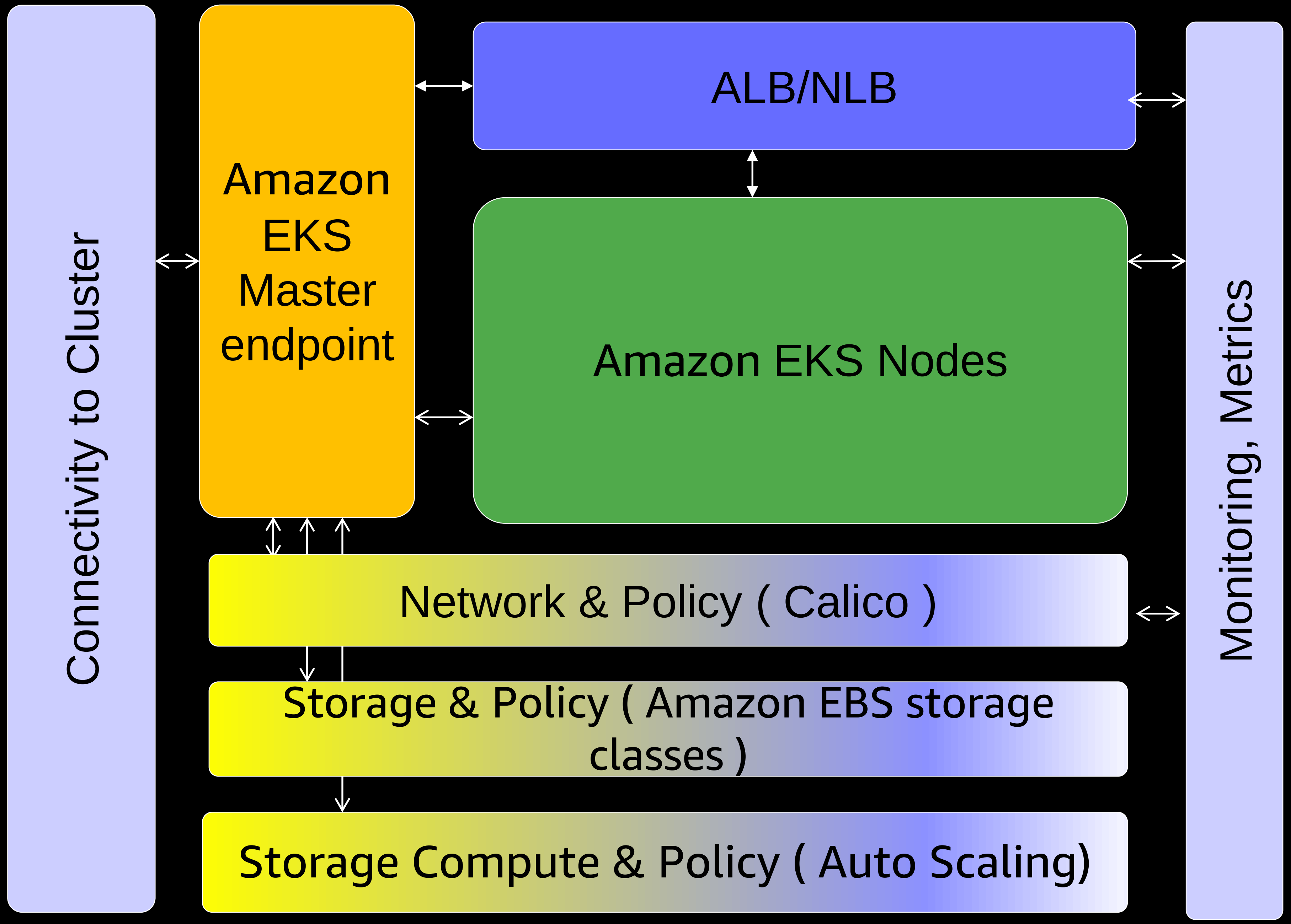
Common financial regulatory challenges

- Traceability of changes
    - Ownership
    - Accountability
- Change control
- Sensitive data security
    - Data accessibility
    - Only owner of the data can create/update/delete it
- Audit and tracking
    - Consumer laws

Amazon EKS Architecture

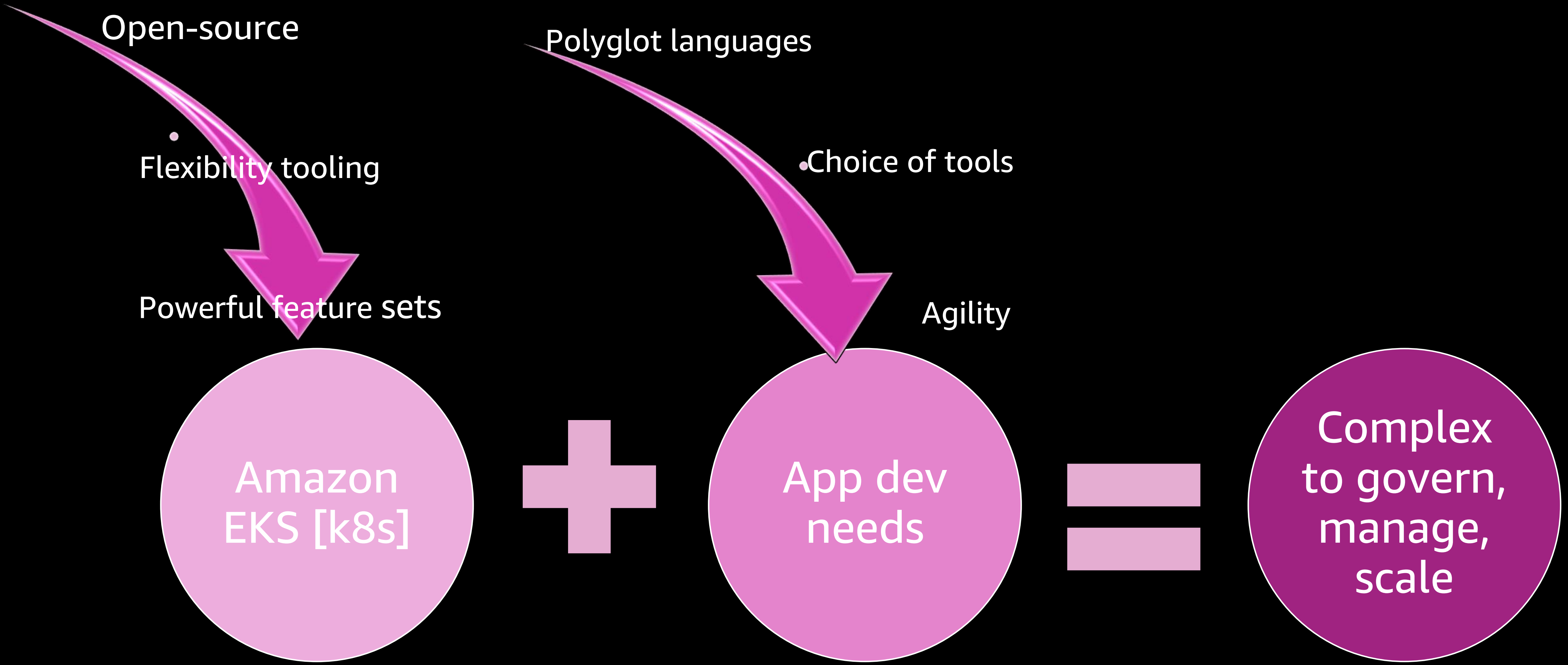


Components of K8 with Amazon EKS



Connectivity to Cluster
Amazon EKS Master endpoint
ALB/NLB
Amazon EKS Nodes
Network & Policy (Calico)
Storage & Policy (Amazon EBS storage classes)
Storage Compute & Policy (Auto Scaling)
Monitoring, Metrics

Case for PaaS



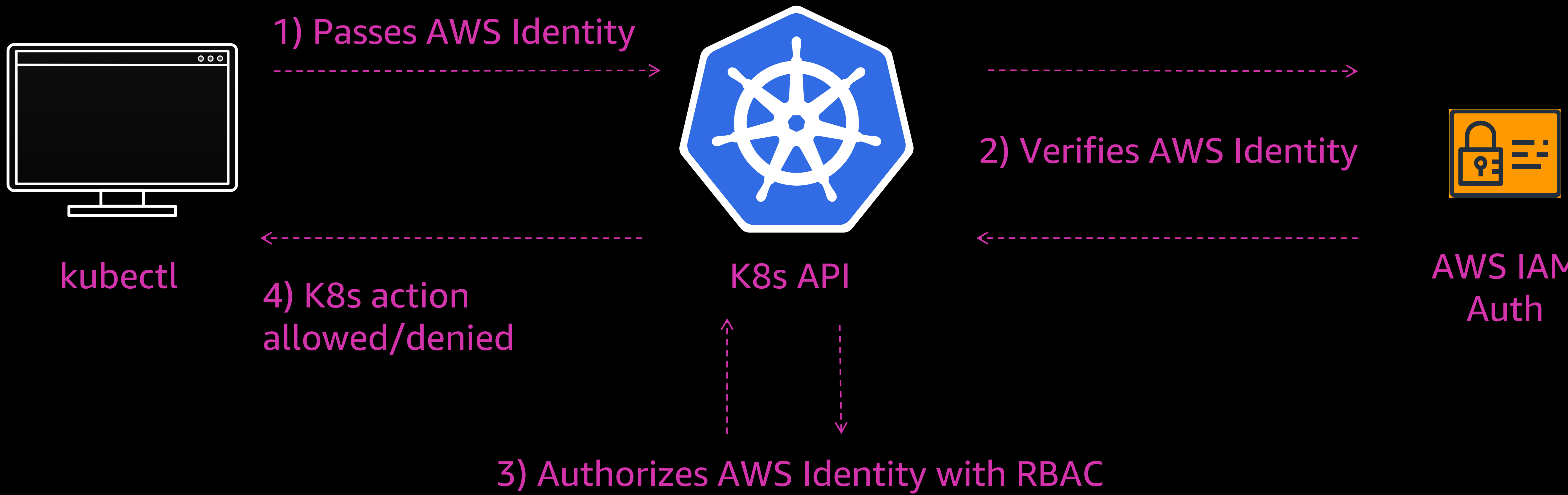
Amazon EKS – Concepts overview

Amazon EKS Security - Auth & AuthZ (A&A)

- AWS Secrets
- AWS KMS
- AWS IAM Roles

- K8s uses RBAC
- AWS IAM Authenticator – Bridge between IAM/RBAC
- K8 mechanisms –
  - Namespaces
  - Service accounts
  - User accounts

Amazon EKS Security - AWS IAM A & A



AWS / Amazon EKS & K8 Network controls

- VPC Security groups
- VPC Subnet NACL

- Pod level implement the **network policy**
  - Network segmentation
  - Tenant isolation
- Network policies similar to AWS security groups
  - Assigned to pods using pod selectors and labels

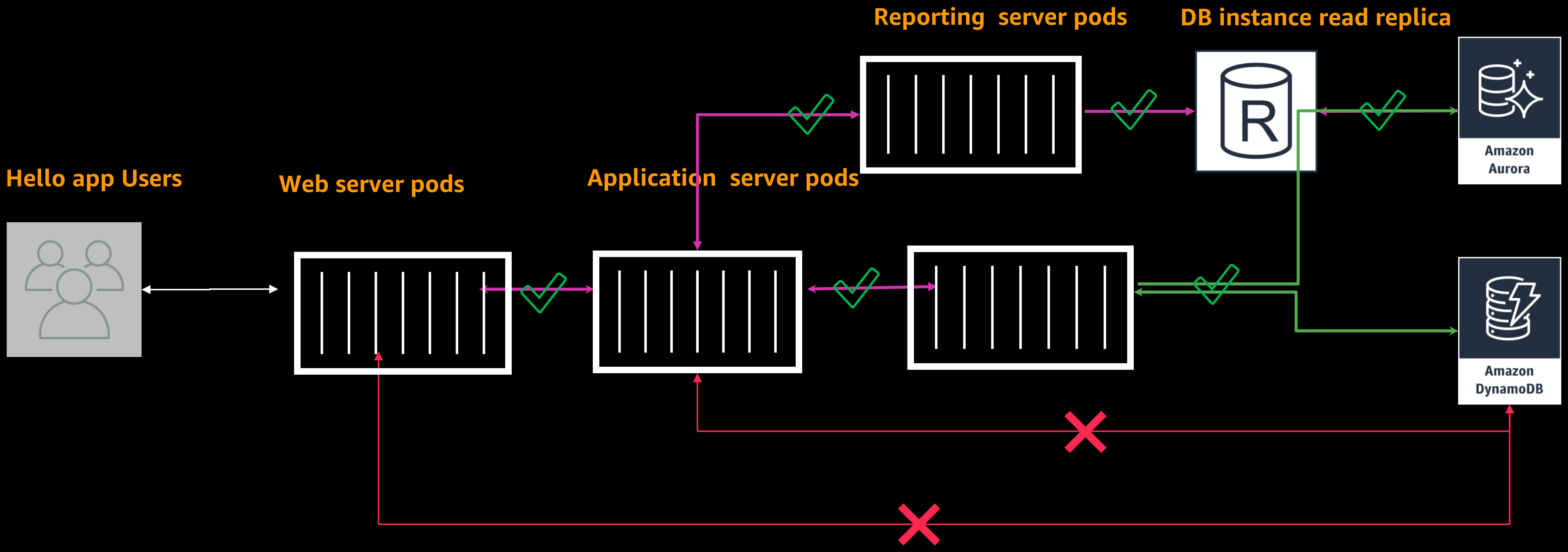
Amazon EKS - Network layer

- K8s you have 3 layers of IP addresses:
  - K8 Cluster level
  - K8 Pod level
  - AWS VPC layer

- CNI plugin
  - L-IPAM daemon, attach & assign & maintain IP addresses to ENI's
  - CNI plugin wiring host network, and adding correct interface to pod namespace

- Deployed as a daemon set on each node
  - Provides the IP addresses for the node based on the Amazon EC2 instance type launched.

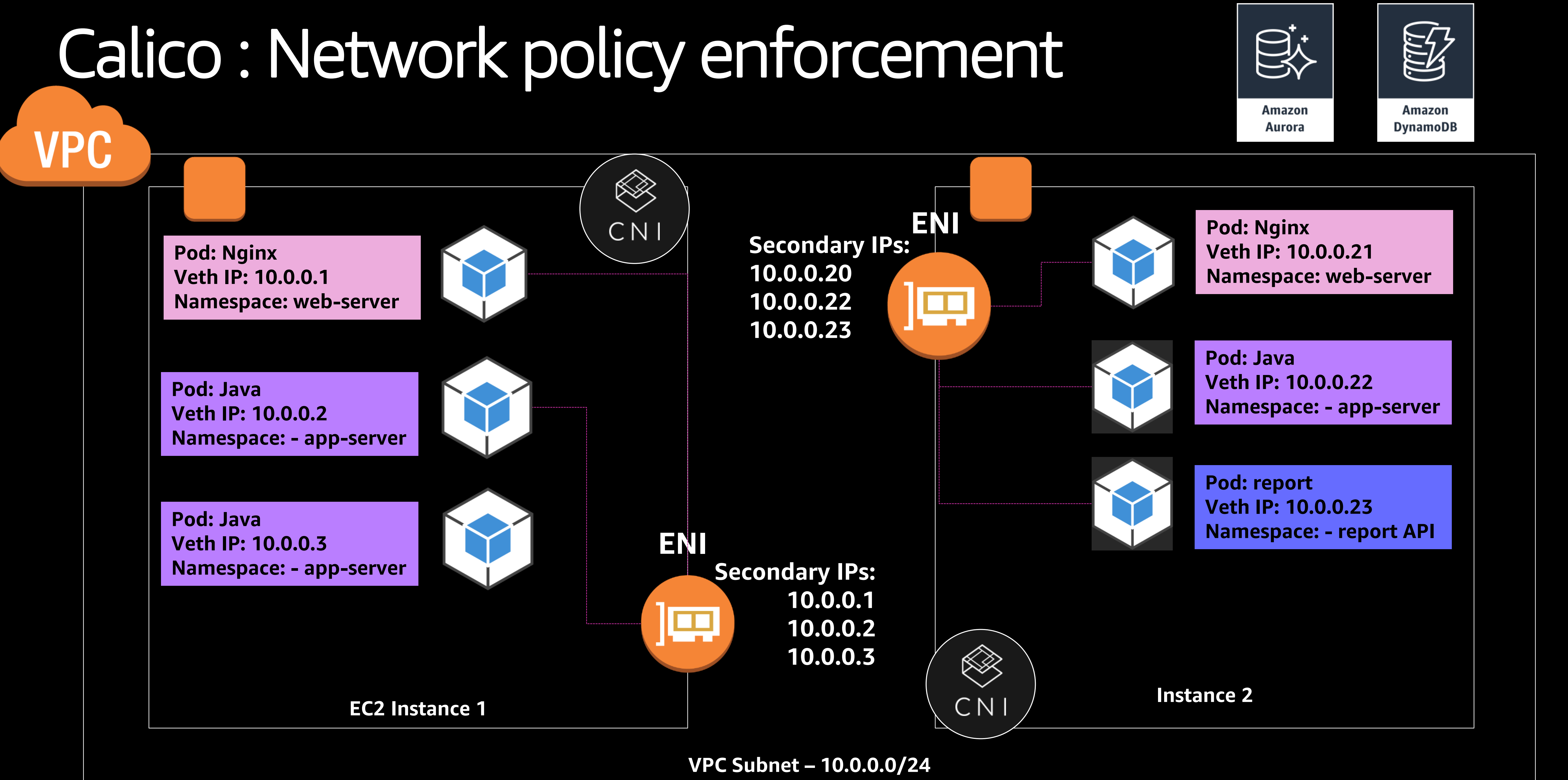
Typical 3-tier application : Traffic flow constraints



Sample policy

```
kubectl create -f - <<EOF
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
   name: access-appserver
   namespace: sample-policy
spec:
   podSelector:
      matchLabels:
         run: app-server
   ingress:
      - from:
         - podSelector:
             matchLabels:
               run: web-server
EOF
```

Calico : Network policy enforcement



AWS & K8s - CI/CD Controls

- AWS CodeCommit
- AWS CodeBuild
- Amazon ECR
- AWS CodePipeline

- K8s Helm package management
- Opensource / partner software to scan images
  - coreos/clair

Amazon EKS – Helm/Package management

- Helm helps you manage K8s apps via Helm charts
- K8s Application Helm charts
  - Define
  - Install
  - Upgrade
- Create, version, share and publish – Important for regulation!
- Reproduce builds of K8s
- Runs on CI/CD or dev laptops

Amazon EKS – Helm/Package management

- Functionality through
  - Client – helm
  - Server – Tiller
  - Charts
- Tiller runs in K8 cluster, manages installations of helm charts
  - Charts are helm packages
    - Description of package
    - One or more templates of K8 manifests
- For example, mysql helm package would create below
  - All required Service accounts, secrets, service, configMaps, pvc, deployment, etc required for running mysql pods in the cluster

Helm mysql example

```
$ helm install --name my-release -f values.yaml stable/mysql

$ helm install --name my-release \ --set 
mysqlRootPassword=secretpassword,mysqlUser=my-user,mysqlPassword=my-
password,mysqlDatabase=my-database \ stable/mysql



Source: 
https://github.com/helm/charts/blob/master/stable/mysql/README.md
```

Visibility & Monitoring

- Amazon CloudWatch Metrics
  - VPC / ALB / Amazon EC2 / ASG / Amazon EKS Control plane
  - Custom metrics
- CloudWatch Logs
  - VPC / ALB /Amazon EC2 / ASG / Amazon EKS Control plane
- AWS CloudTrail

- K8s Scaling metrics
  - HPA
  - Cluster auto-scaler
  - Cluster wide metrics

Fidelity

PaaS Business Objectives

A Native Cloud Strategy is critically important to being competitive as a Digital Business.
As organizations progress along their Cloud Journey, the question is how to maximize business value while balancing risk & complexity

Scale & Agility

Faster time to market.

Focus on business logic and value not underlying plumbing.

Innovation

Launch new products leveraging innovative cloud capabilities.

Enable New business opportunities in a Digital API marketplace

Reliability & Scalability

On-demand elastic compute.

Multi-availability zones and data centers for redundancy and HA

Transparency/ Traceability

Full transparency and traceability on usage, access, assets, deployment, issues.

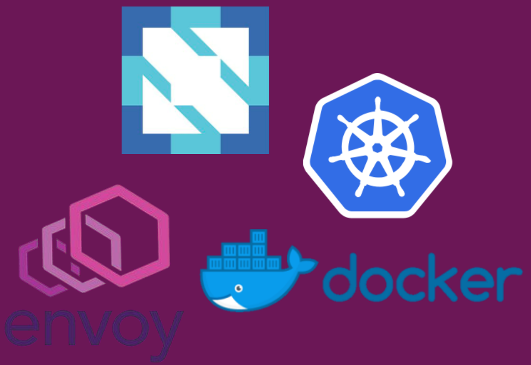
Cost Savings

Utilization based chargeback
Avoid developing non-differentiating capabilities

Innovation Tenets

A Native Cloud Strategy is critically important to being competitive as a Digital Business.
As organizations progress along their Cloud Journey, the question is how to maximize business value while balancing risk & complexity

Standards



Leveraging Open Source technologies

Managed CSP services

Multi-tenancy

Common deployment pipelines should be leveraged to enforce security controls & policies.

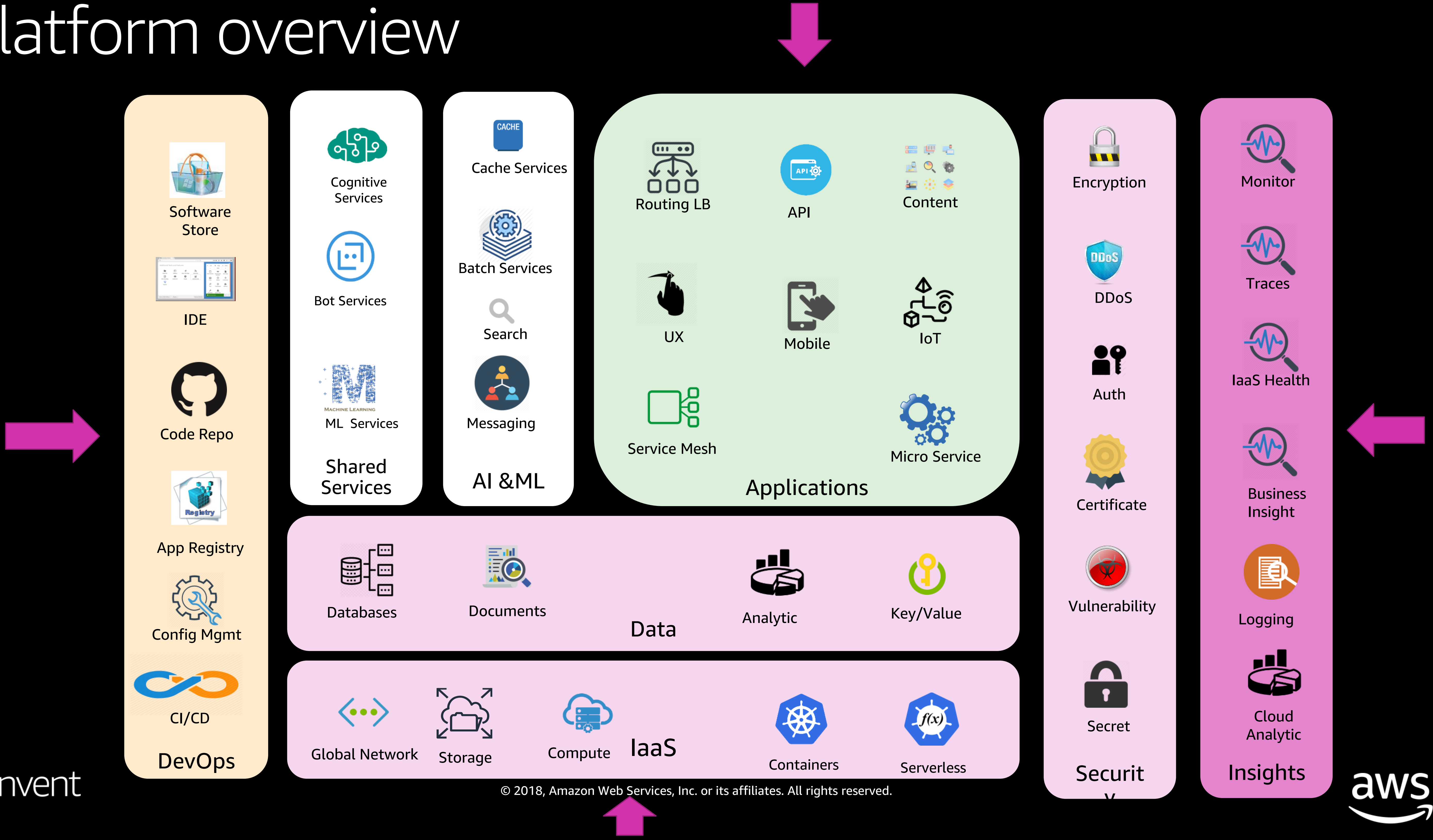
Security

Adopting and aligning to industry standards safeguards

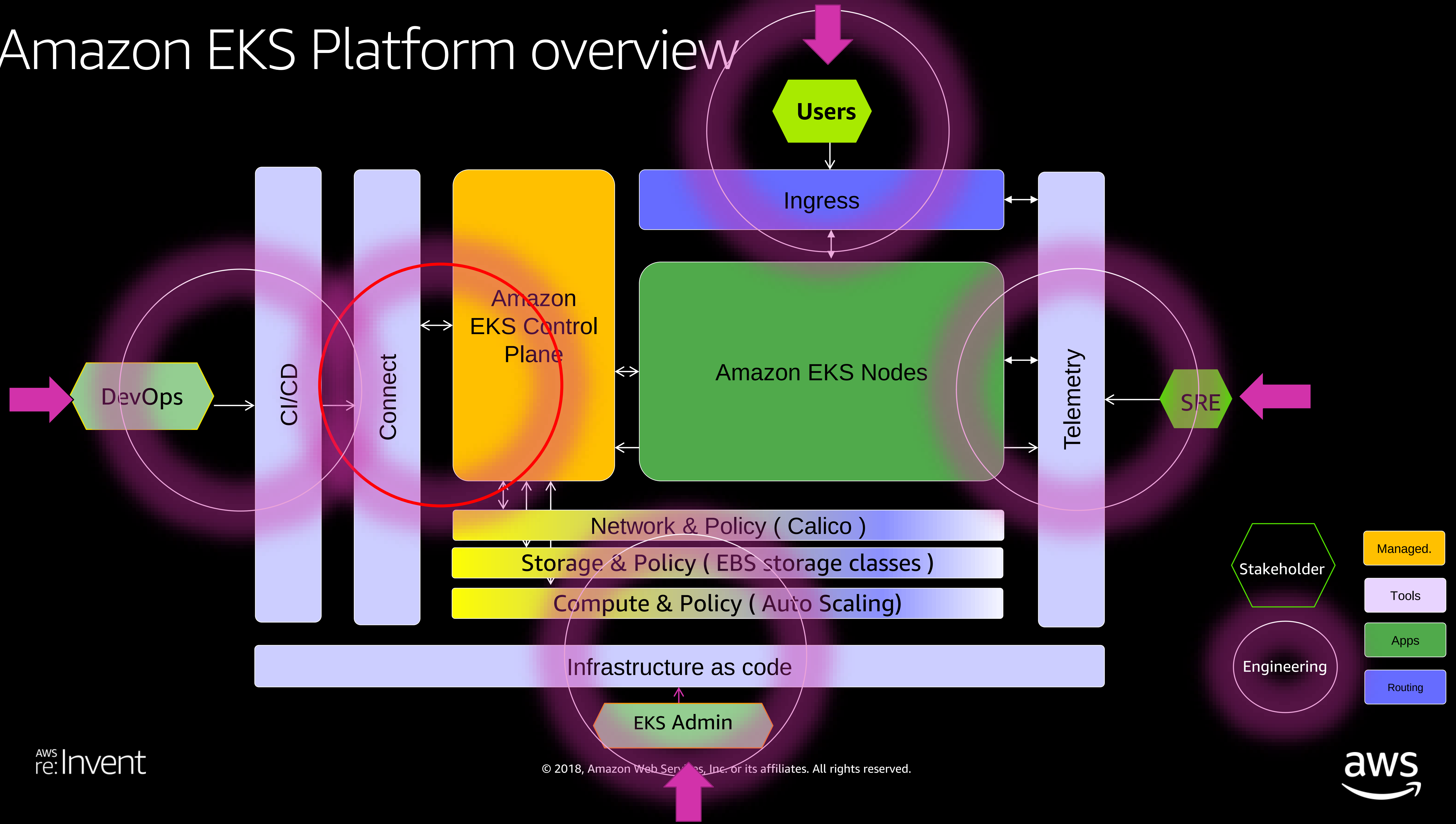
Audit & Traceability

Common set of standards, controls and policies

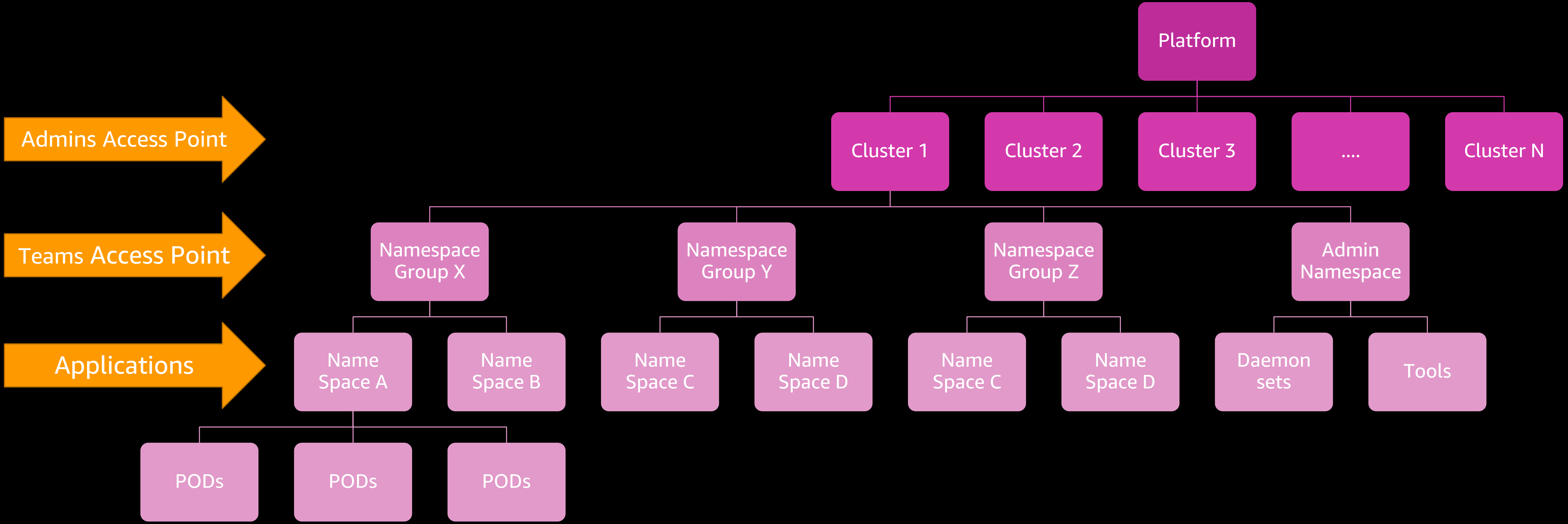
Platform overview



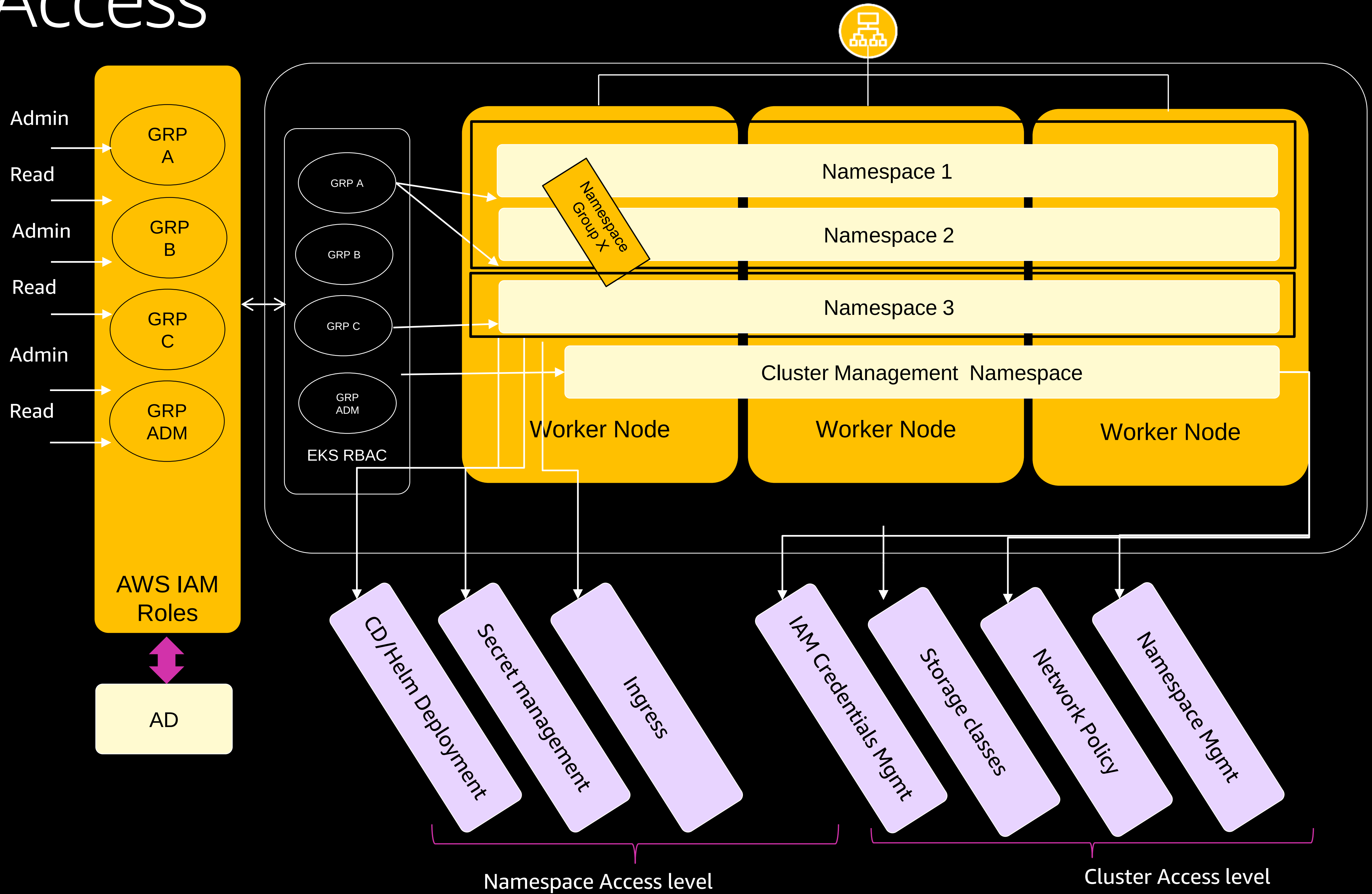
Amazon EKS Platform overview



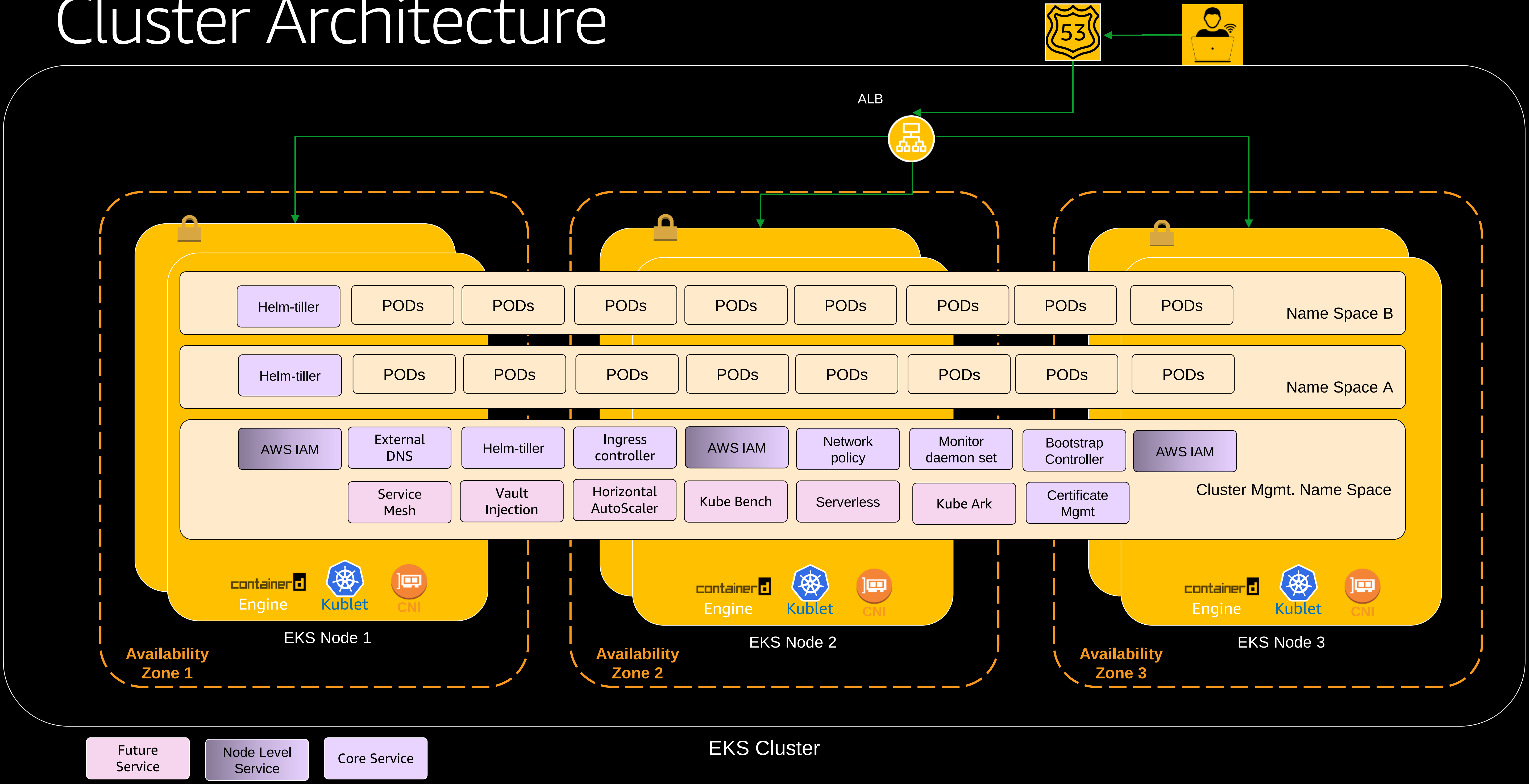
Platform Structure



Cluster Access

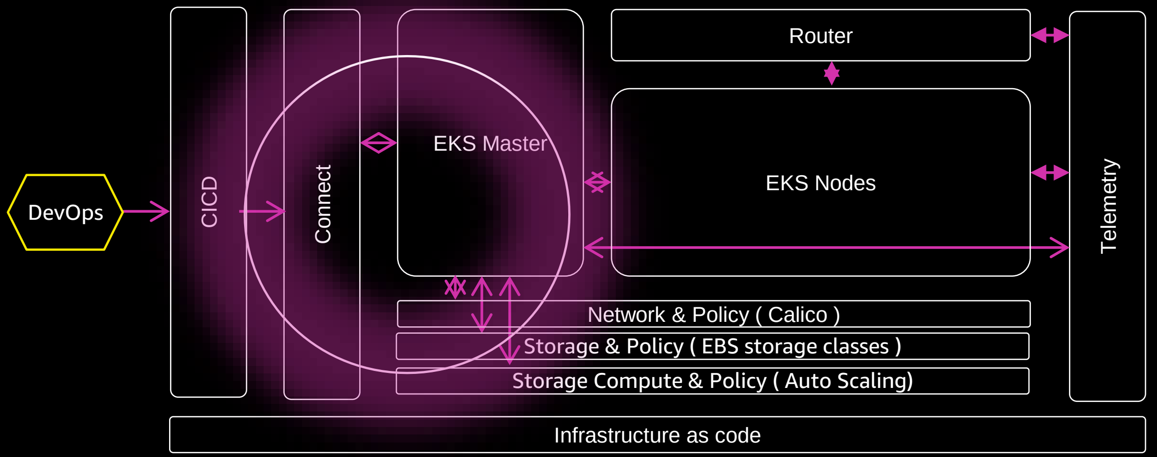


Cluster Architecture

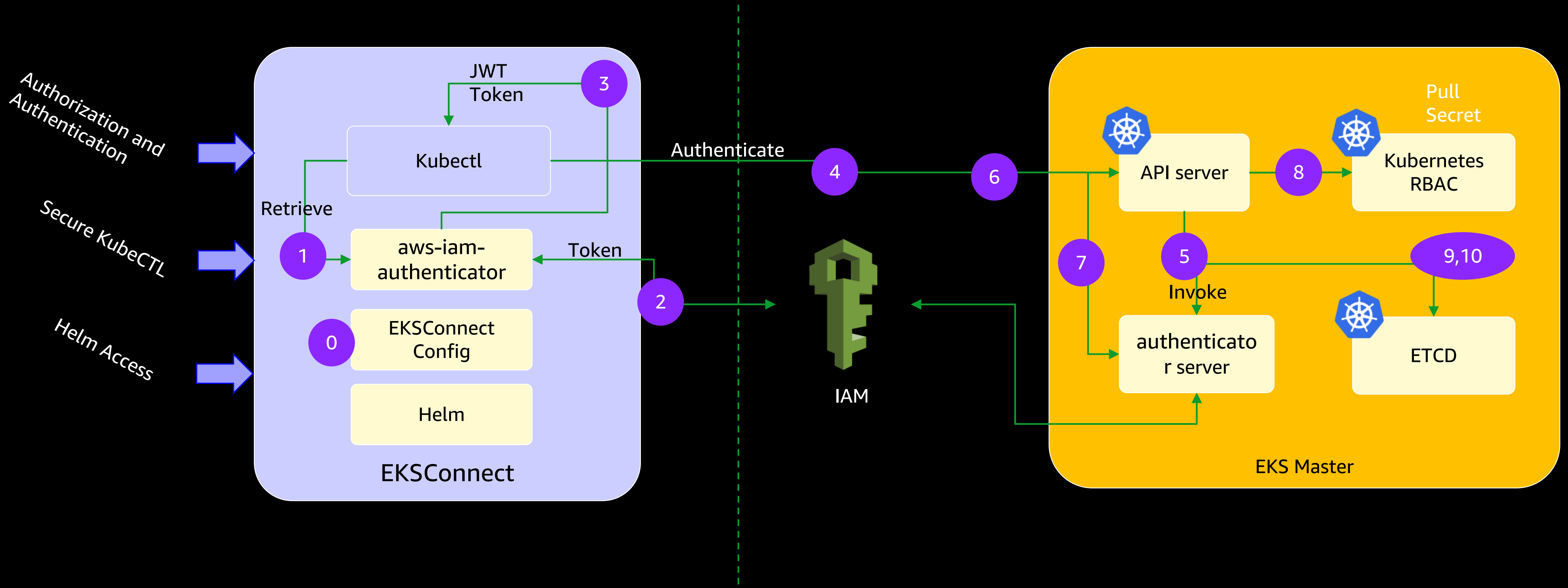


Platform Access

EKS Connect



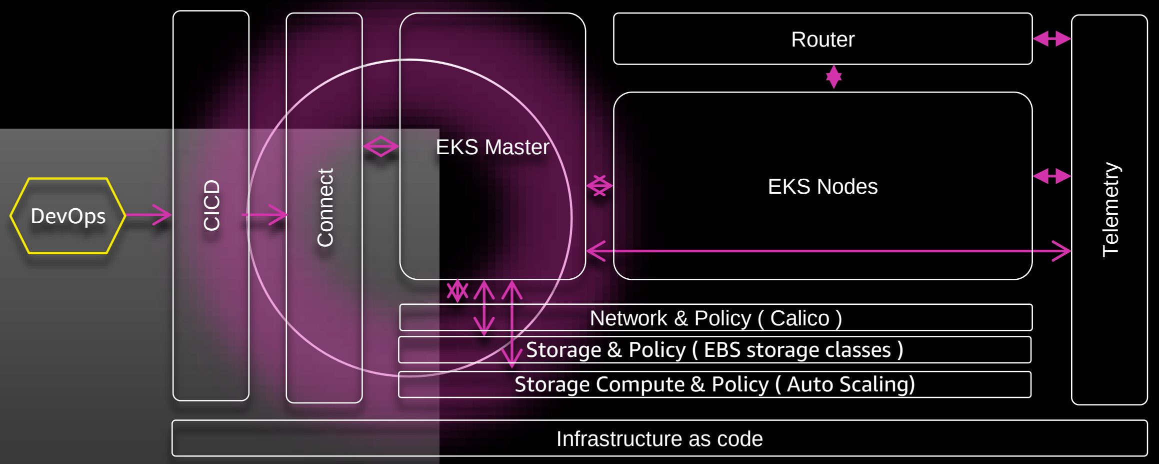
DevOps
CICD
Connect
EKS Master
Router
EKS Nodes
Telemetry
Network & Policy (Calico)
Storage & Policy (EBS storage classes)
Storage Compute & Policy (Auto Scaling)
Infrastructure as code
LINUX



Platform Access

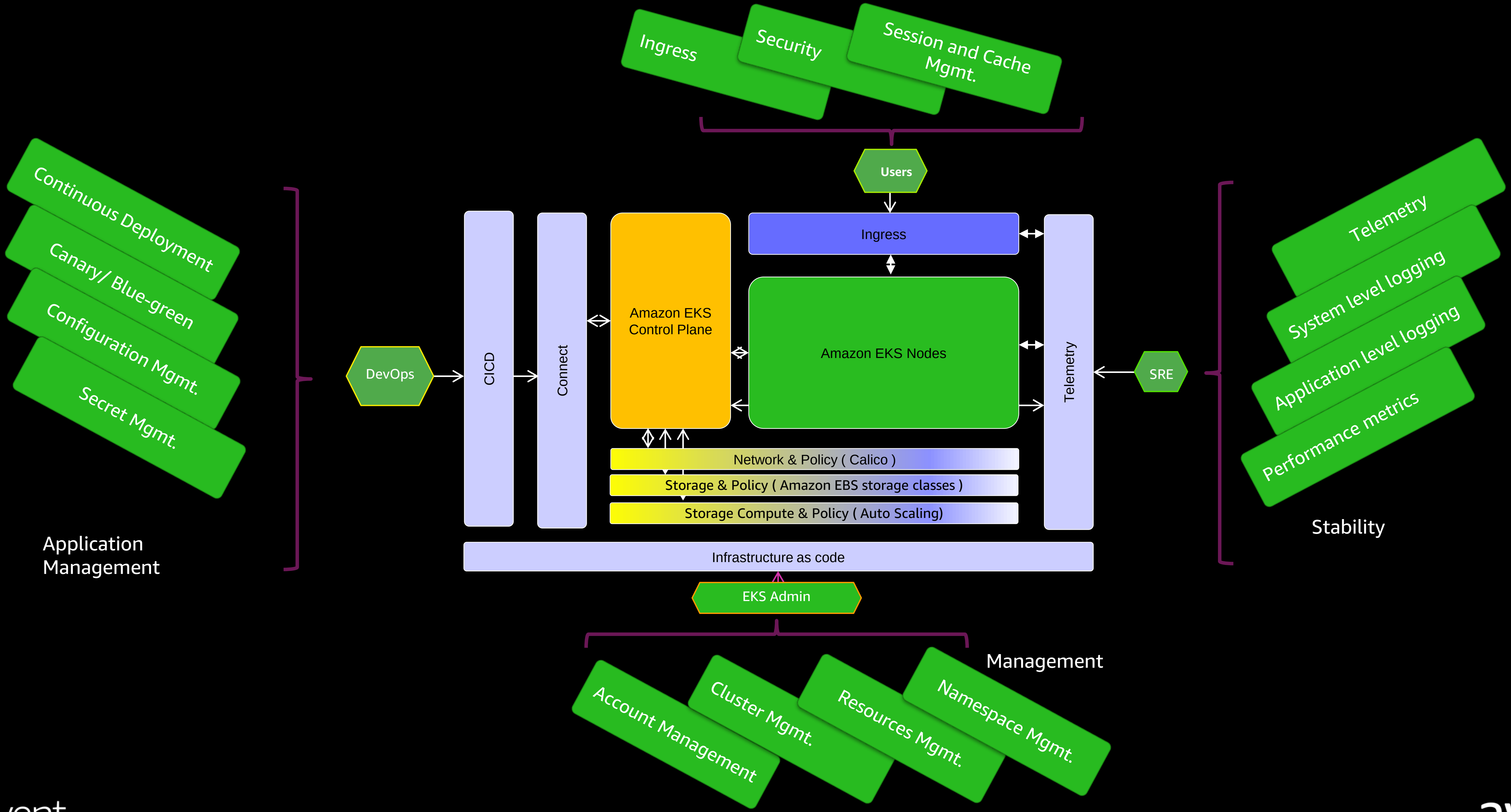
EKS Connect

```
$ ./eksconnect

username [user123]:
password:
domain [default]:
region [us-east-1]:
Choose from the below list of available roles
[ 0 ] ----> arn:aws:iam::****:role/Cluster1_EKSMaster
[ 1 ] ----> arn:aws:iam::****:role/Cluster2_NameSpaceGroup1_namespaceX_API
[ 2 ] ----> arn:aws:iam::****:role/Cluster2_NameSpaceGroup1_namespaceY_UI
Role No [0]: 1
profile [default]:
Writing credentials ....
set http_proxy
set https_proxy
set no_proxy

********************************************************************************

$ kubectl get pods
NAME                         READY    STATUS        RESTARTS   AGE
kv-sidecar-69667c9cb9-6nhp2   0/2      Terminating   0          10d
kv-sidecar-69667c9cb9-zrlfc   0/2      Terminating   0          10d
postgres-7ff9df5765-2xhjn     1/1      Running       0          11d
```


DevOps
CICD
Connect
EKS Master
Router
EKS Nodes
Telemetry
Network & Policy (Calico)
Storage & Policy (EBS storage classes)
Storage Compute & Policy (Auto Scaling)
Infrastructure as code

LINUX

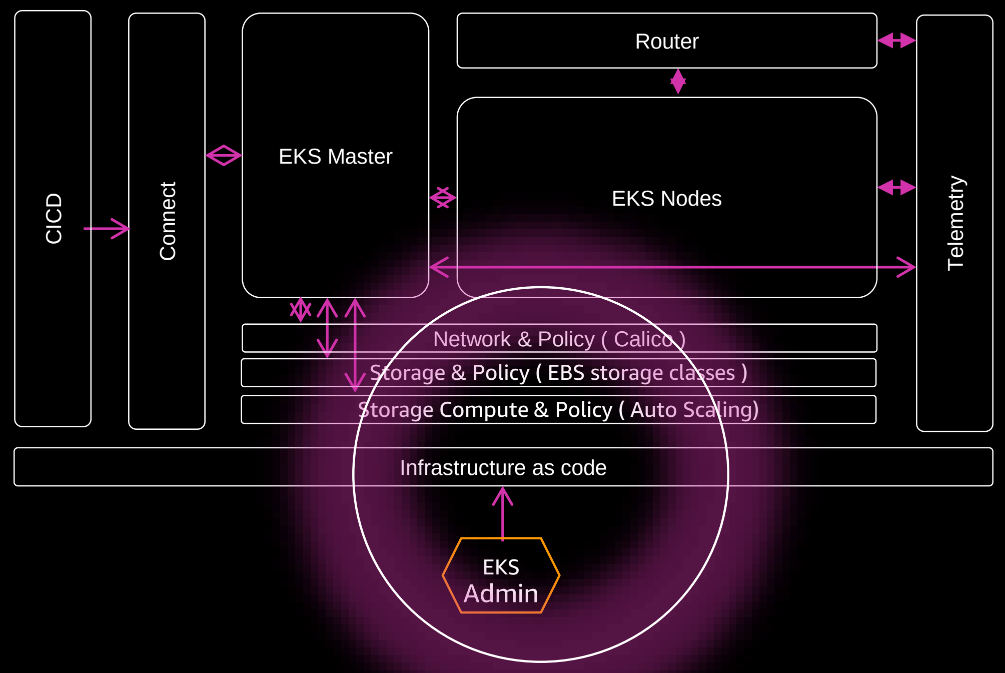
Platform Stakeholders



Platform Build

```
apiVersion: eksManager.fidelity.com
kind: Cluster
metadata:
  name: <Cluster1>
  description: ”reinvent Cluster"
labels:
     key: value
spec:
   cloudId: AWS
   envType: dev
   appOwner: <userA>
   appEmail: <userA>@ourplatfrom.com
   rbac:
    –  accessLevel:
      adgroup: <...>
      iamRole: <..>
      members:
        – ”<userB>"
        – ”<userC>"
nodes:
.........
secrets:
.........
monitor:
.......
```

EKS Manager



CICD
Connect
EKS Master
Router
EKS Nodes
Telemetry
Network & Policy (Calico)
Storage & Policy (EBS storage classes)
Storage Compute & Policy (Auto Scaling)
Infrastructure as code
EKS Admin

CLI + Cluster YAML

Cluster Bootstrapping

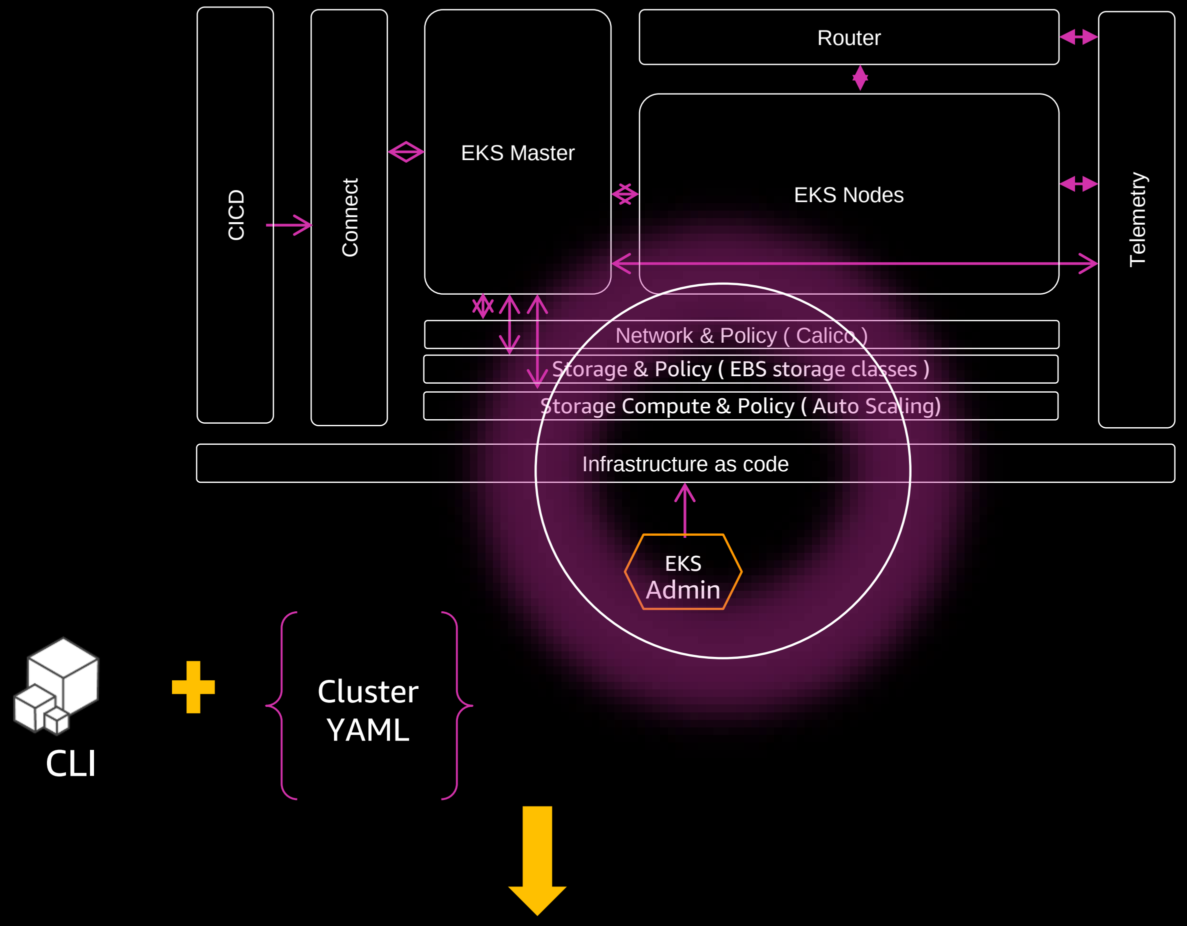
- Setup IAM roles
- Setup security group.
- Provision Control Plane.
- Provision Cluster Nodes.
- Setup Admin group, AWS IAM and RBAC.

NS Management Bootstrapping

- Setup Ingress Controller.
- Setup Namespace Bootstrap controller.
- Setup Secret Management Controller.
- Setup AWS IAM Controller

Platform Onboarding

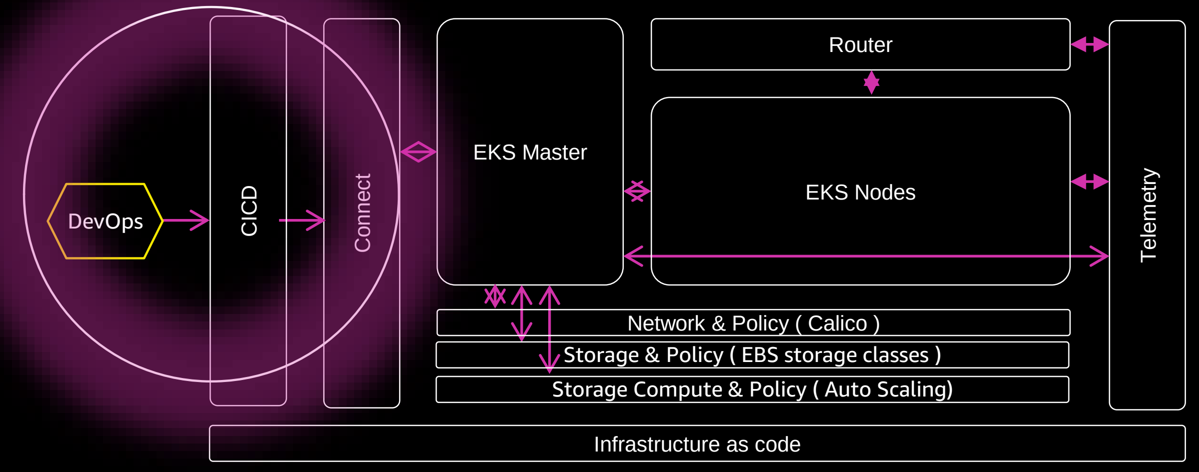
```
apiVersion: nsbootstrapoperator.fidelity.com
kind: nsgroup
metadata:
  name: <GroupX>
  description: "Name Space Group X"
  namespace: default
  labels:
      key: value
spec:
    cloudId: AWS
    envType: dev
    appOwner: <userA>
    appEmail: <userA>@ourplatfrom.com
    rbac:
     -  accessLevel:
        adgroup: <...>
        iamRole: <..>
        members:
           - "<userB>"
           - "<userC>"
 -  accessLevel: readonly
        adgroup:<...>
        iamRole: <..>
        members:
           - "<userC>"
           - "<userD>"
namespaces:
        - name:  <Cluster>_<NamespaceGroup>_<namespace>_<UI>
        - name:  <Cluster>_<NamespaceGroup>_<namespace>_<API>
        - name:
<Cluster>_<NamespaceGroup>_<namespace>_<BACKEND>
.........
```



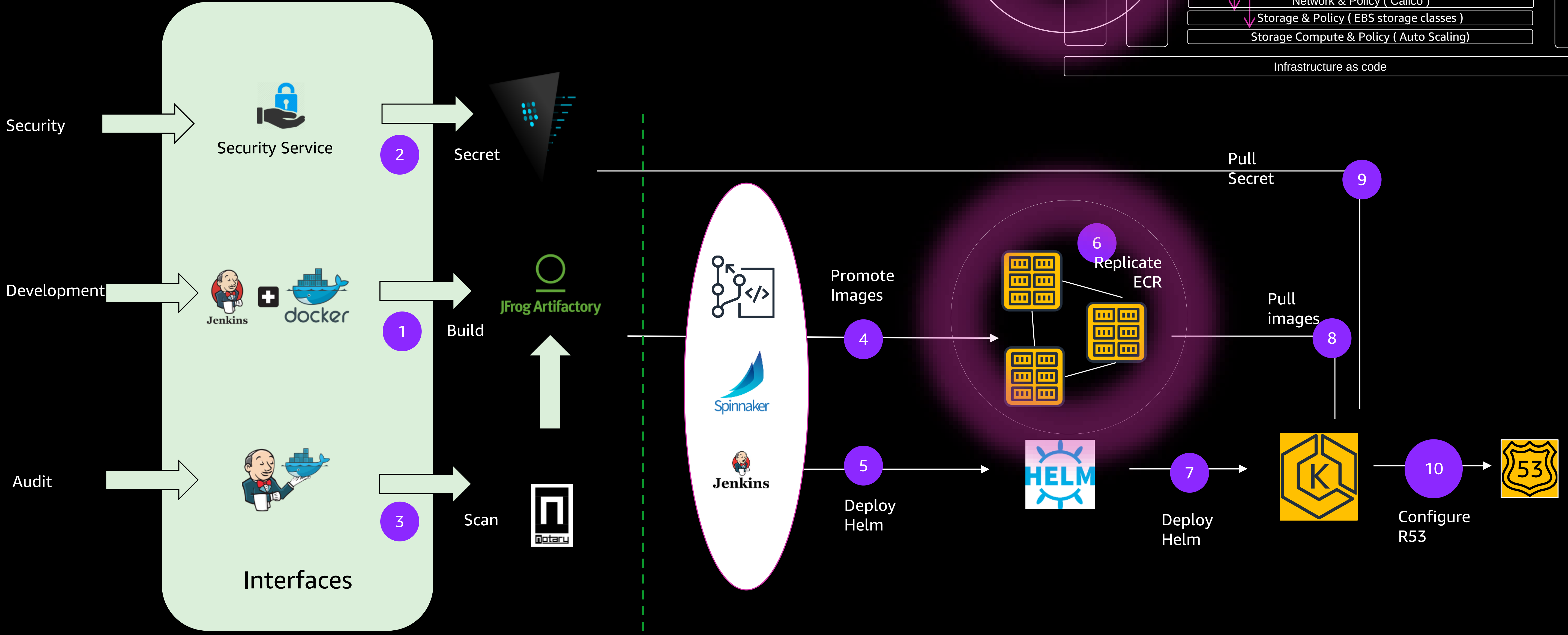
CICD
Connect
EKS Master
Router
EKS Nodes
Telemetry
Network & Policy (Calico)
Storage & Policy (EBS storage classes)
Storage Compute & Policy (Auto Scaling)
Infrastructure as code
EKS Admin
CLI
Cluster YAML

- Setup AD group , AWS IAM roles and RBAC.
- Create Namespace(s).
- Setup Helm-Teller.
- Setup Ingress controller.

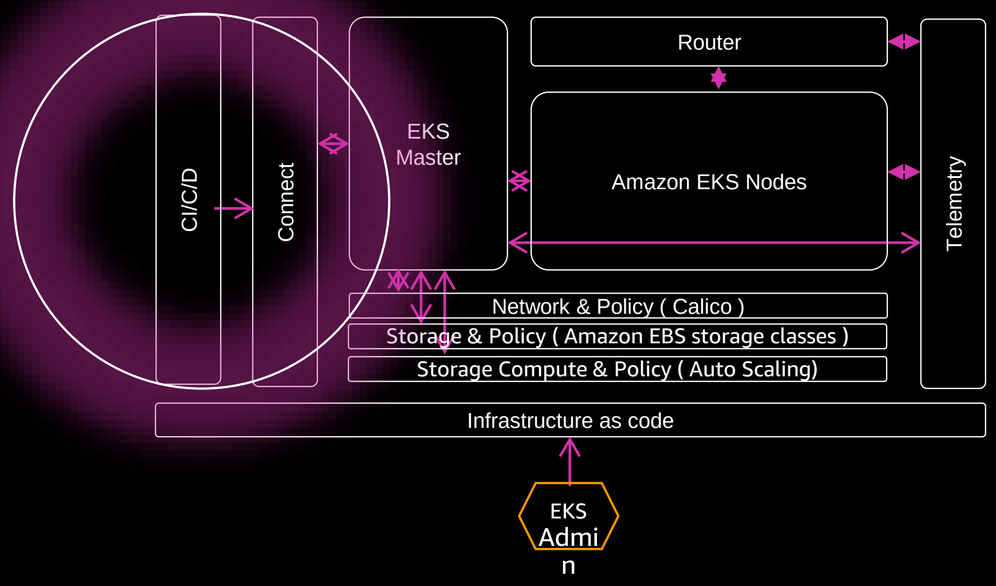
Application Life Management

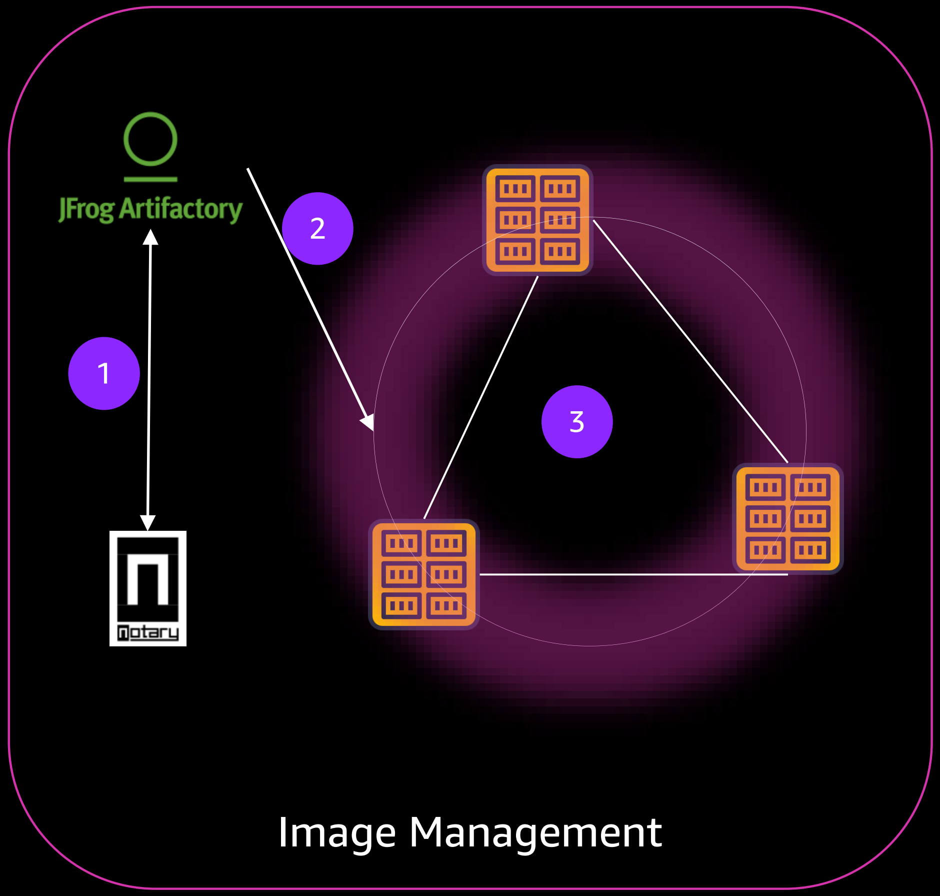


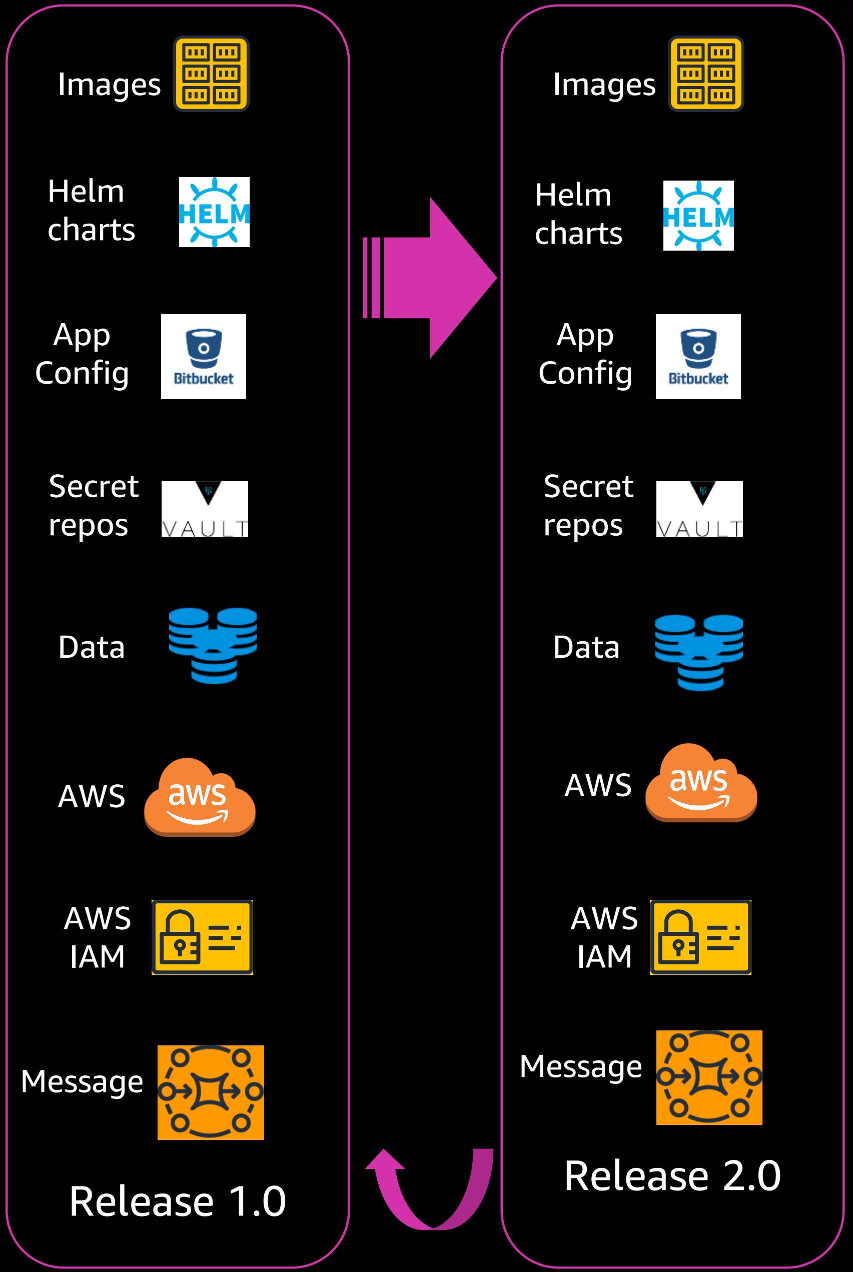
DevOps
CICD
Connect
EKS Master
Router
EKS Nodes
Telemetry
Network & Policy (Calico)
Storage & Policy (EBS storage classes)
Storage Compute & Policy (Auto Scaling)
Infrastructure as code

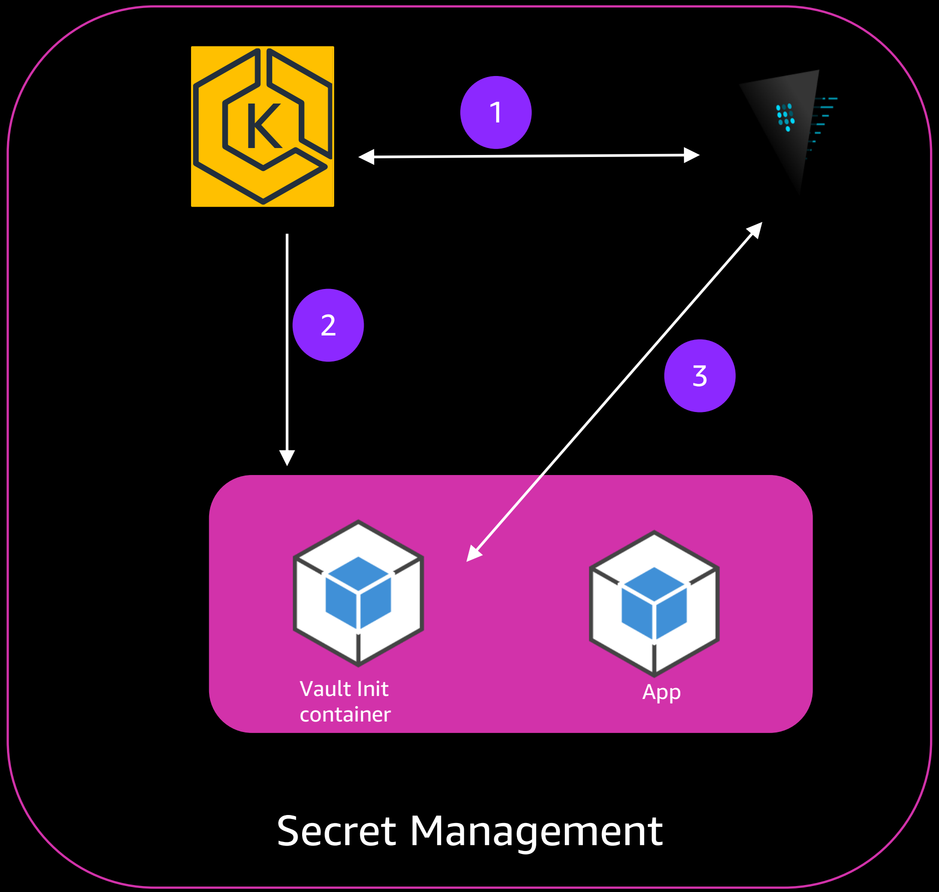


Release Management

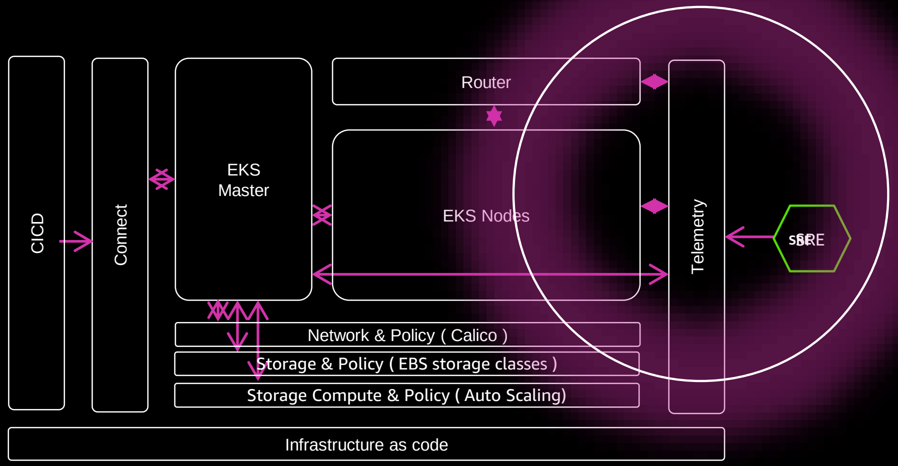

CI/C/D
Connect
EKS Master
Router
Amazon EKS Nodes
Telemetry
Network & Policy (Calico)
Storage & Policy (Amazon EBS storage classes)
Storage Compute & Policy (Auto Scaling)
Infrastructure as code
EKS Admi n

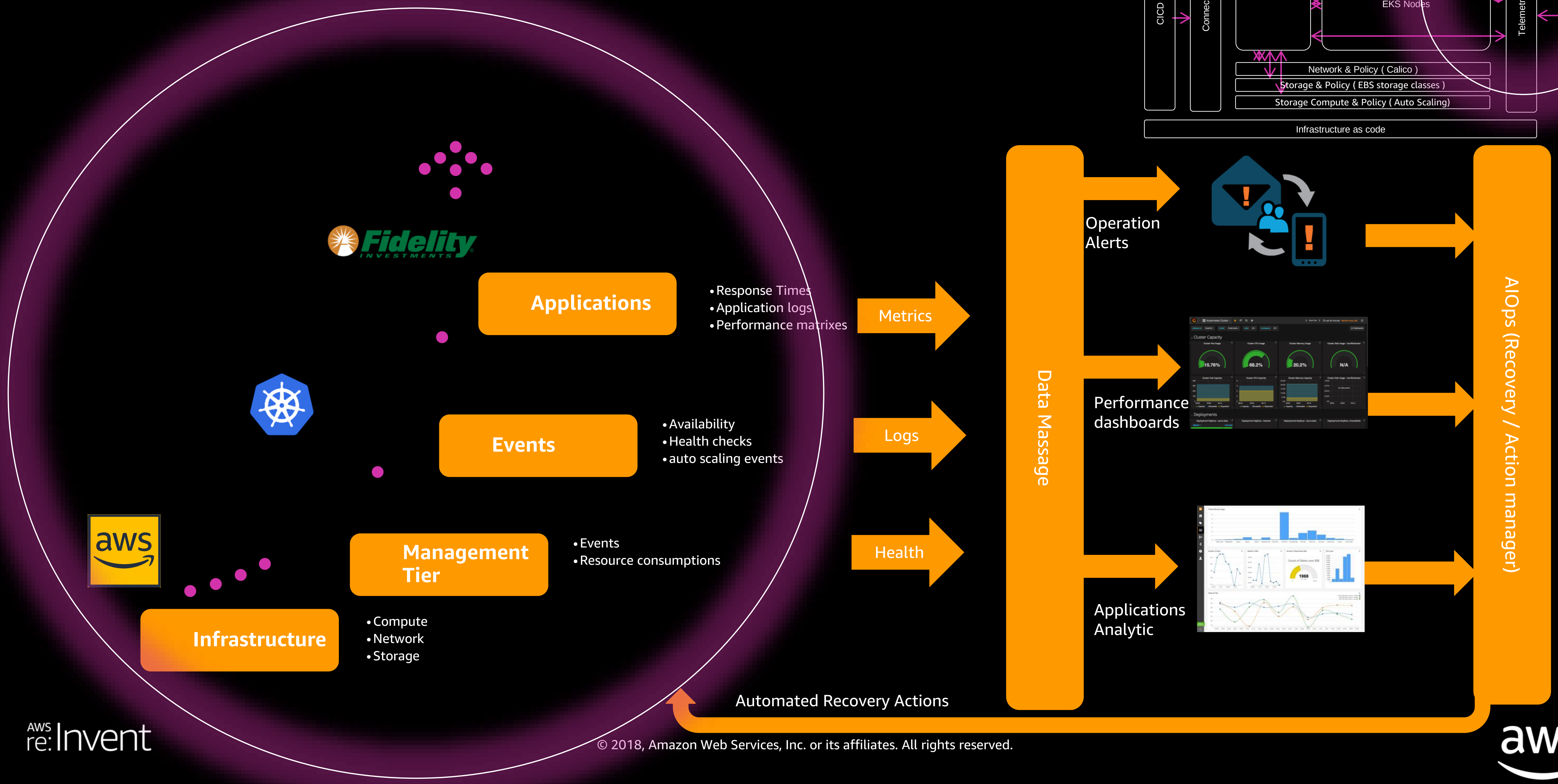




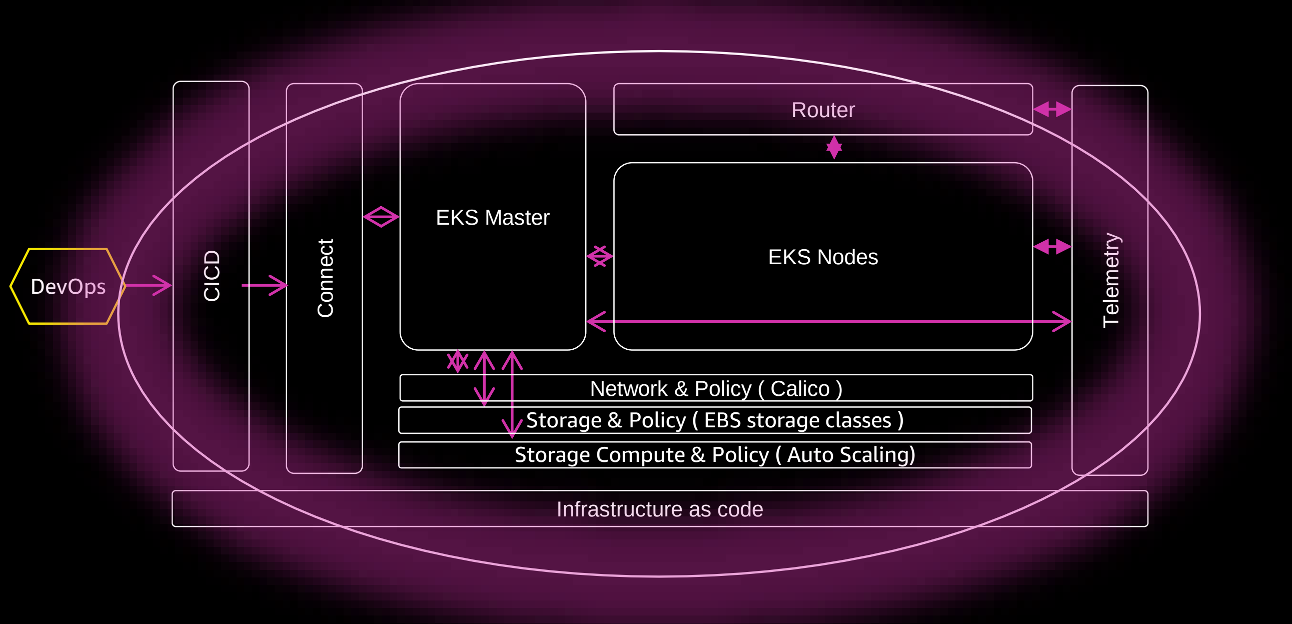


EKSWatcher - Telemetry


CICD
Connect
EKS Master
Router
EKS Nodes
Telemetry
SRE
Network & Policy (Calico)
Storage & Policy (EBS storage classes)
Storage Compute & Policy (Auto Scaling)
Infrastructure as code



Future Tools - Concept


DevOps
CICD
Connect
EKS Master
Router
EKS Nodes
Telemetry
Network & Policy (Calico)
Storage & Policy (EBS storage classes)
Storage Compute & Policy (Auto Scaling)
Infrastructure as code

Cost Management

- Cost to run container per hour.
- Cost to run application.
- Containers density.

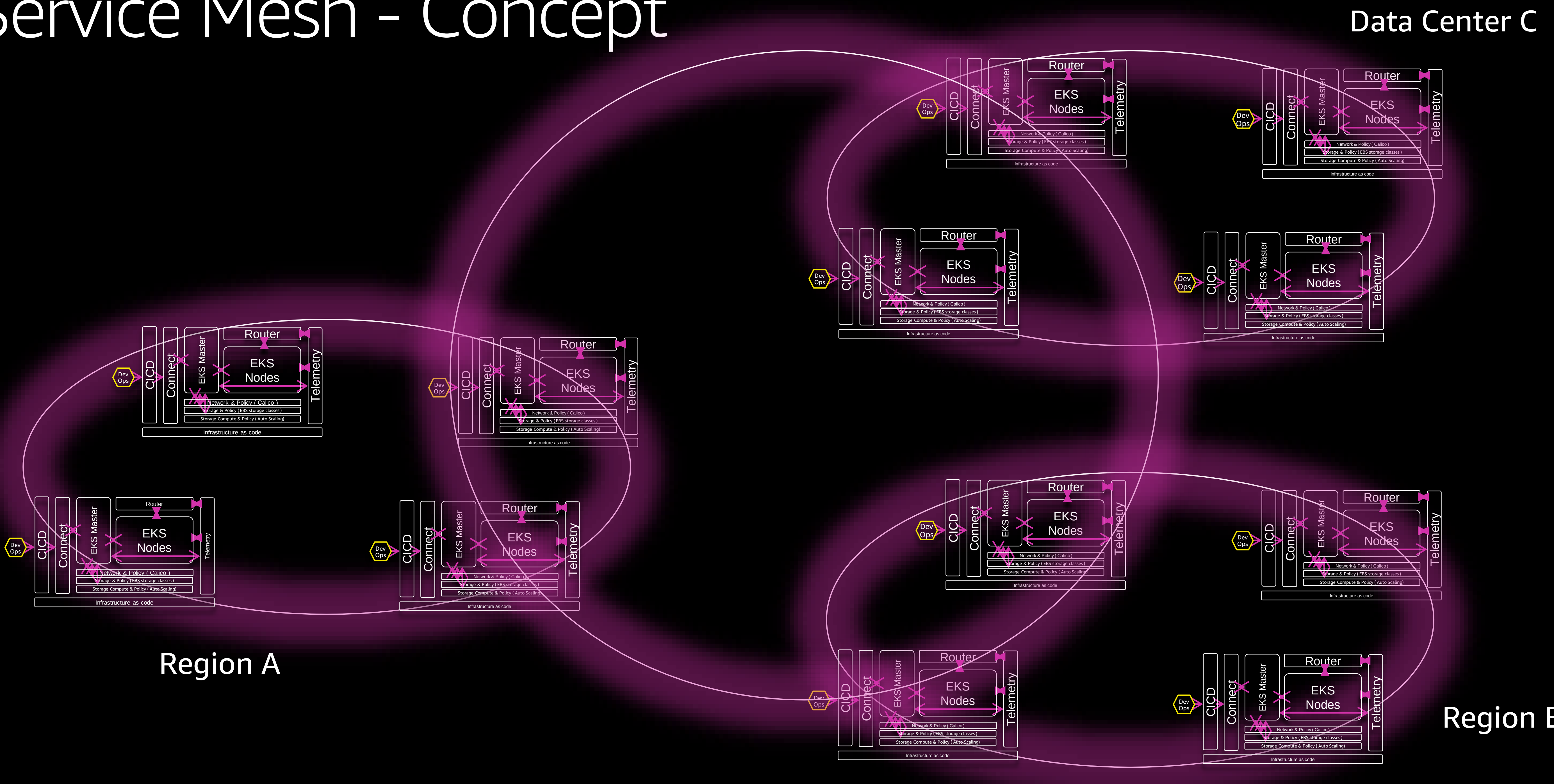
Resource Utilization

- EC2 utilization
- IP Utilization
- Storage Utilization

Security Automation

- Vulnerability scanning
- Penetration testing
- Audit as a Service
- Outages post mortem

Service Mesh - Concept


Data Center C
Region A
Region B
Router
EKS Nodes
EKS Master
CICD
Connect
Telemetry
Dev Ops
Network & Policy (Calico)
Storage & Policy (EBS storage classes)
Storage Compute & Policy (Auto Scaling)
Infrastructure as code

Thank you!

Amr Abdelhalem
https://www.linkedin.com/in/amrhalem/

Madhuri Peri
https://www.linkedin.com/in/mpericloud/

Please complete the session survey in the mobile app.