
AWS
re:Invent
DECEMBER 2 – 6, 2024 | LAS VEGAS, NV

KUB405

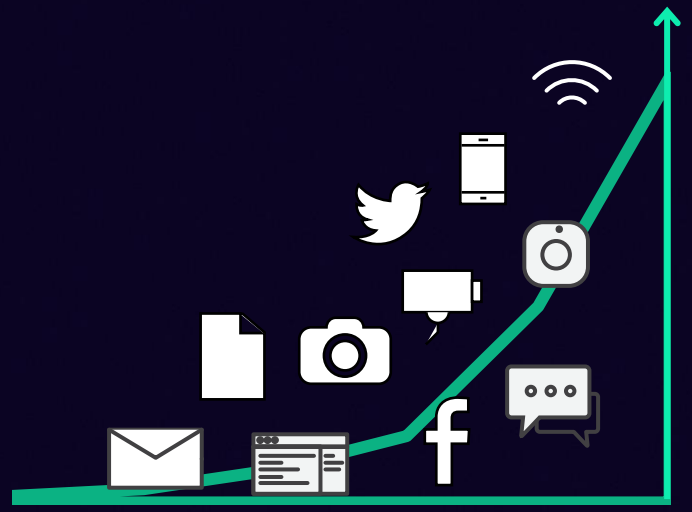
Amazon EKS as data platform for analytics

Victor Gershkovich
(he/him)
R&D Group Leader – Data Platform
AppsFlyer

Christina Andonov
(she/her)
Senior Specialist Solution Architect
AWS

Roland Barcia
(he/him)
Director Specialist Solution Architects
AWS

Big data challenges

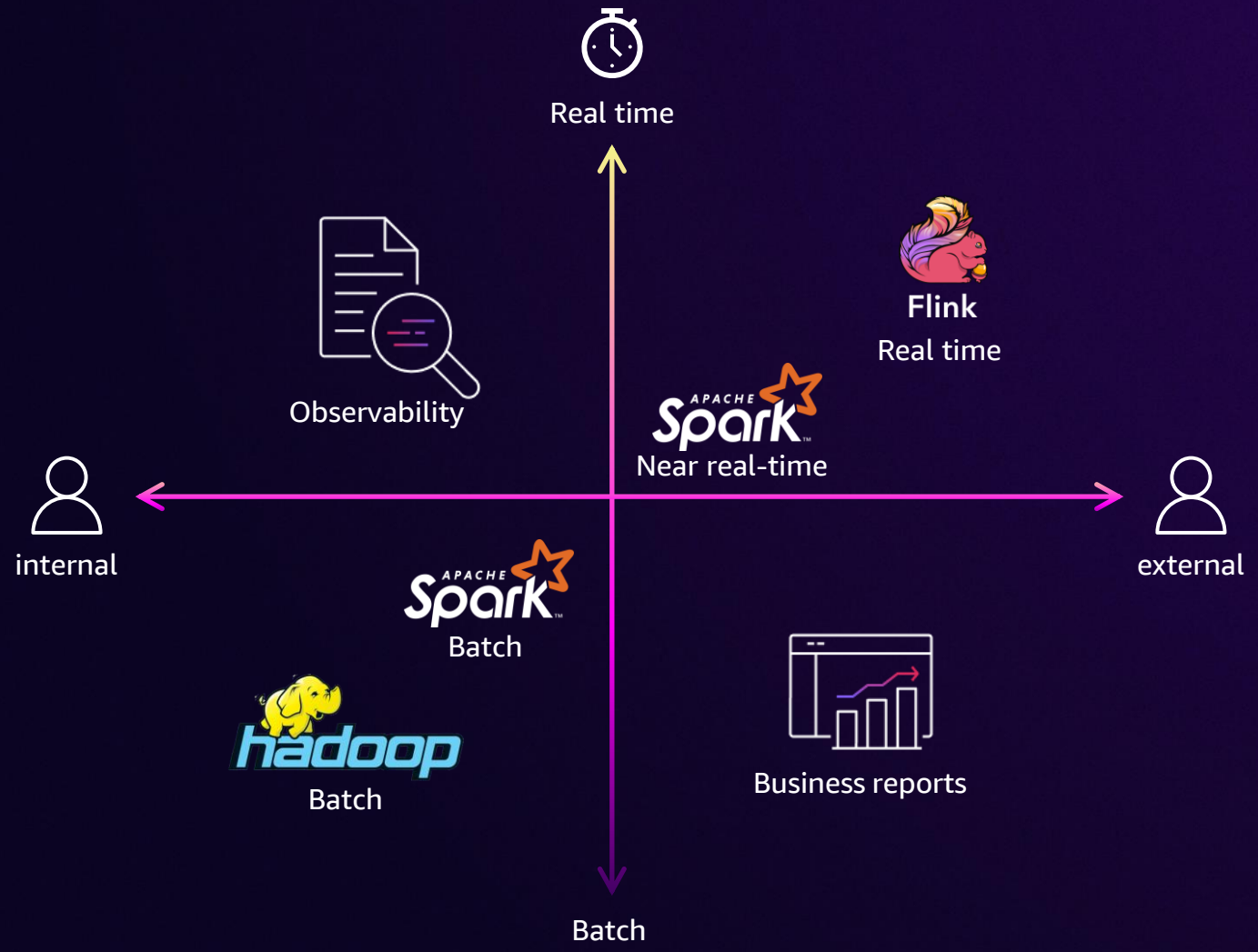


Explosion of data

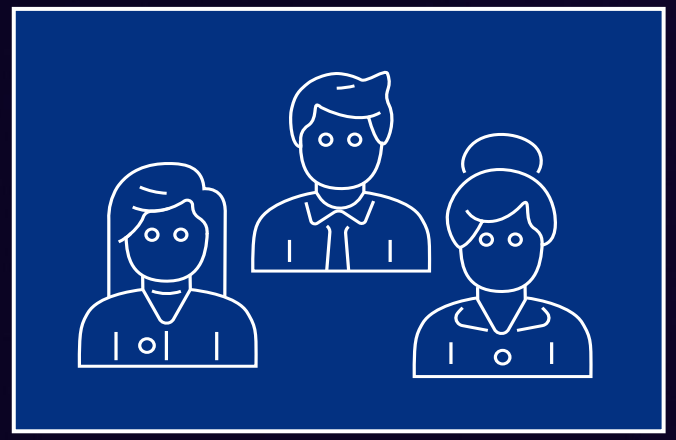
Explosion of personas

Demand for faster decision- making on real-time data

Data patterns



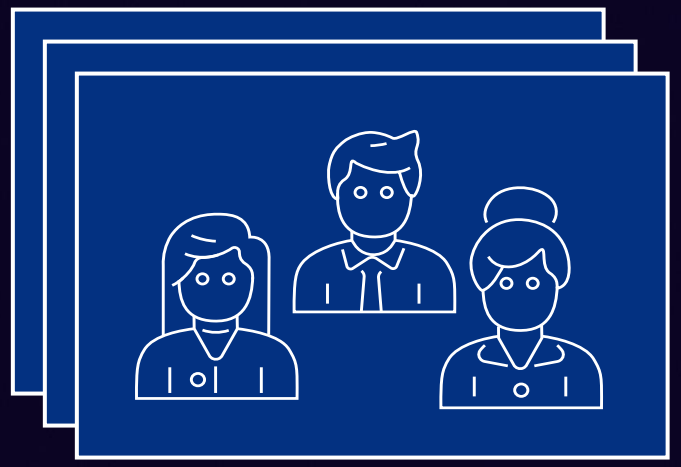
First generation of platform target applications



Development team

Platform team

The ownership boundaries



Development teams

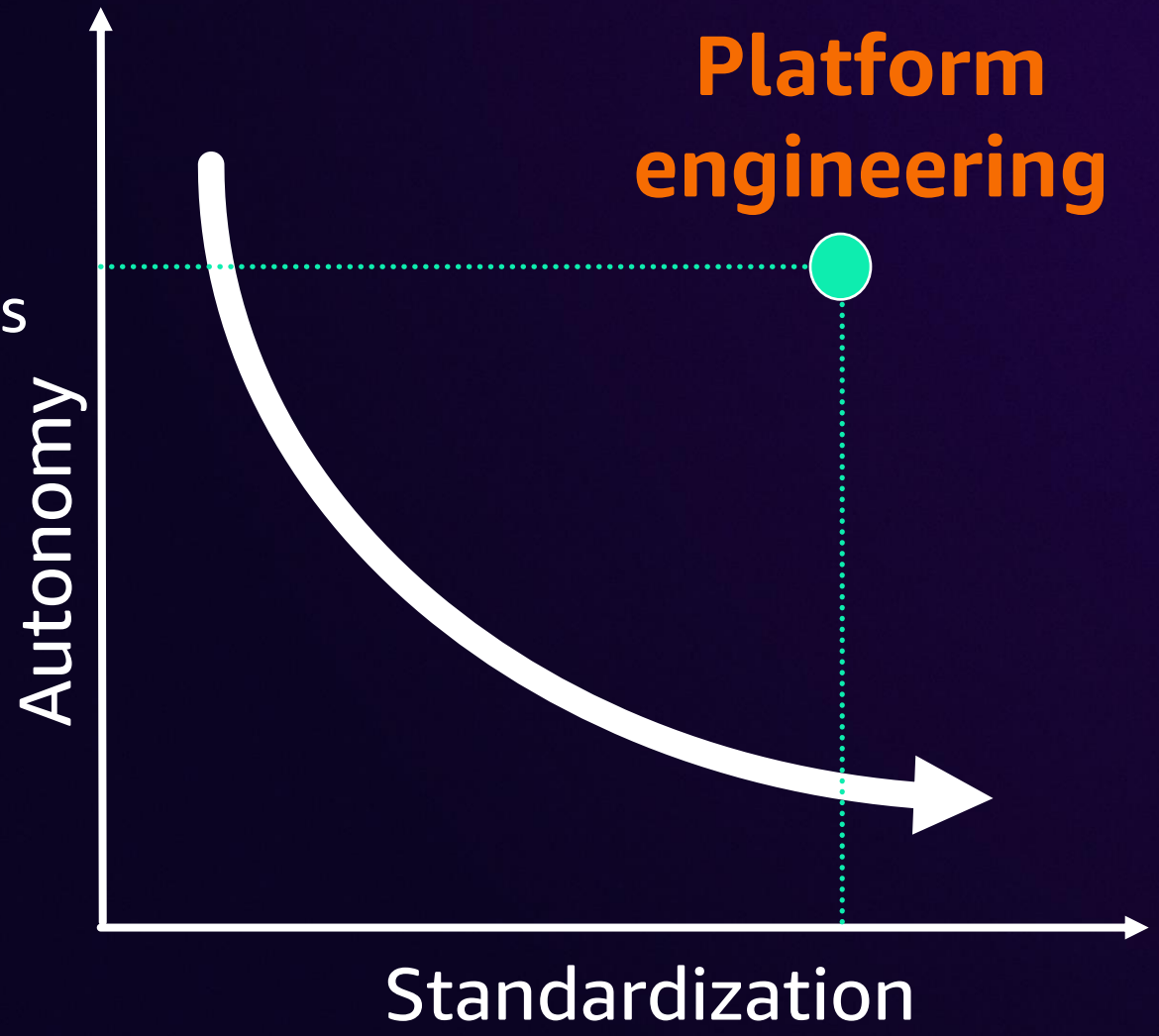
Rapid innovation! Agility! Time to market! Feedback cycle! New services!

Standardization! Security! Governance! Observability! Cost efficiency!

Platform team

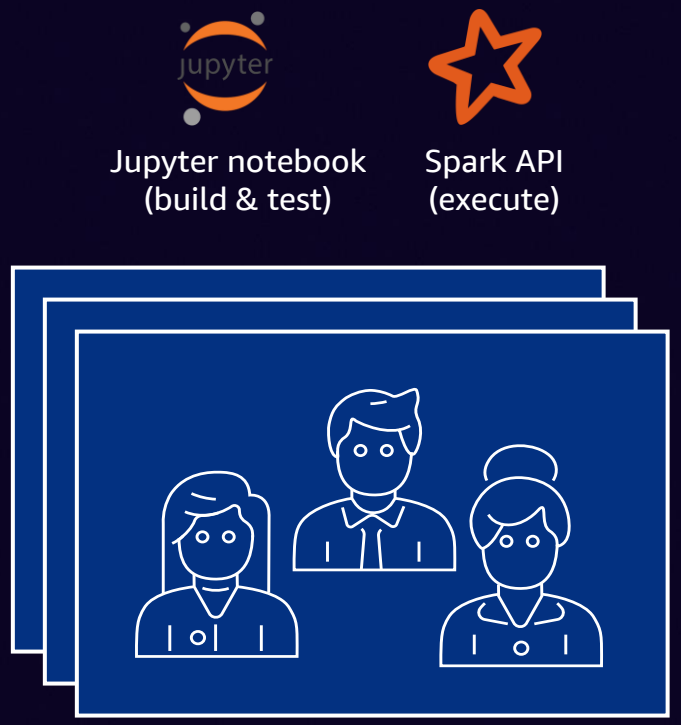
Striking the balance

Dev and
data engineer teams



Platform team

Data scientists and engineers



Jupyter notebook
(build & test)

Spark API
(execute)

Data engineers

Stateful apps, notebooks, data lakes and meshes, parallel processing, GPU!

+ Storage, GPU, HPC, MlOps

Platform team

Goals for data workloads

Highly Scalable

Better resource utilization

Compute & storage options

Security & multi-tenant isolation

Cost efficiency

Favorite framework open source community

Common challenges for stateful workloads

Scaling data-intensive workloads to 1,000+ nodes

Network configurations

Considerations for high availability, fault-tolerance, and failover

How to configure multi-tenancy and security on Kubernetes

Batch scheduling options

Choosing right compute and storage options

Logging and monitoring applications

Managing Amazon EKS cluster infrastructure

Example: Spark on AWS

Self-managed Spark on EKS

Flexible options for running open source Spark on EKS

Wide selection of open source integrations

Portability and versioning

For customers that want to standardize on EKS to manage clusters across applications but are willing to maintain the OSS components

Amazon EMR on EKS

Low TCO and fast performance

Secure by default

Ease of use

For customers that want to standardize on EKS to manage clusters across applications or use different versions of an open-source framework on the same cluster

Amazon EMR Serverless

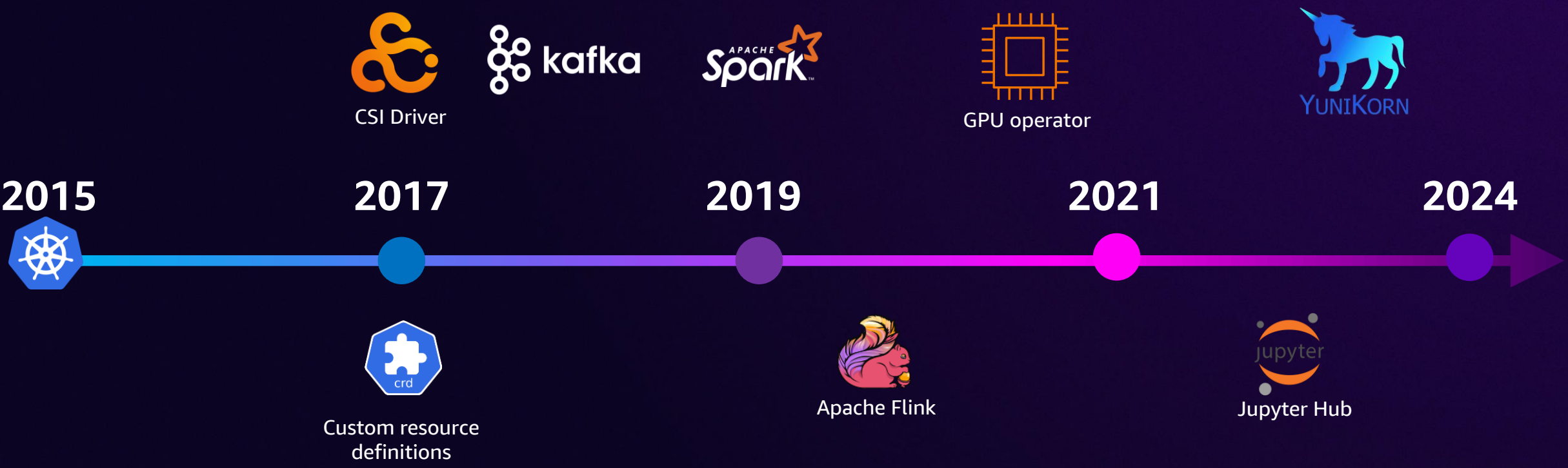
Automatic and fine-grained scaling

Resilience to Availability Zone failures

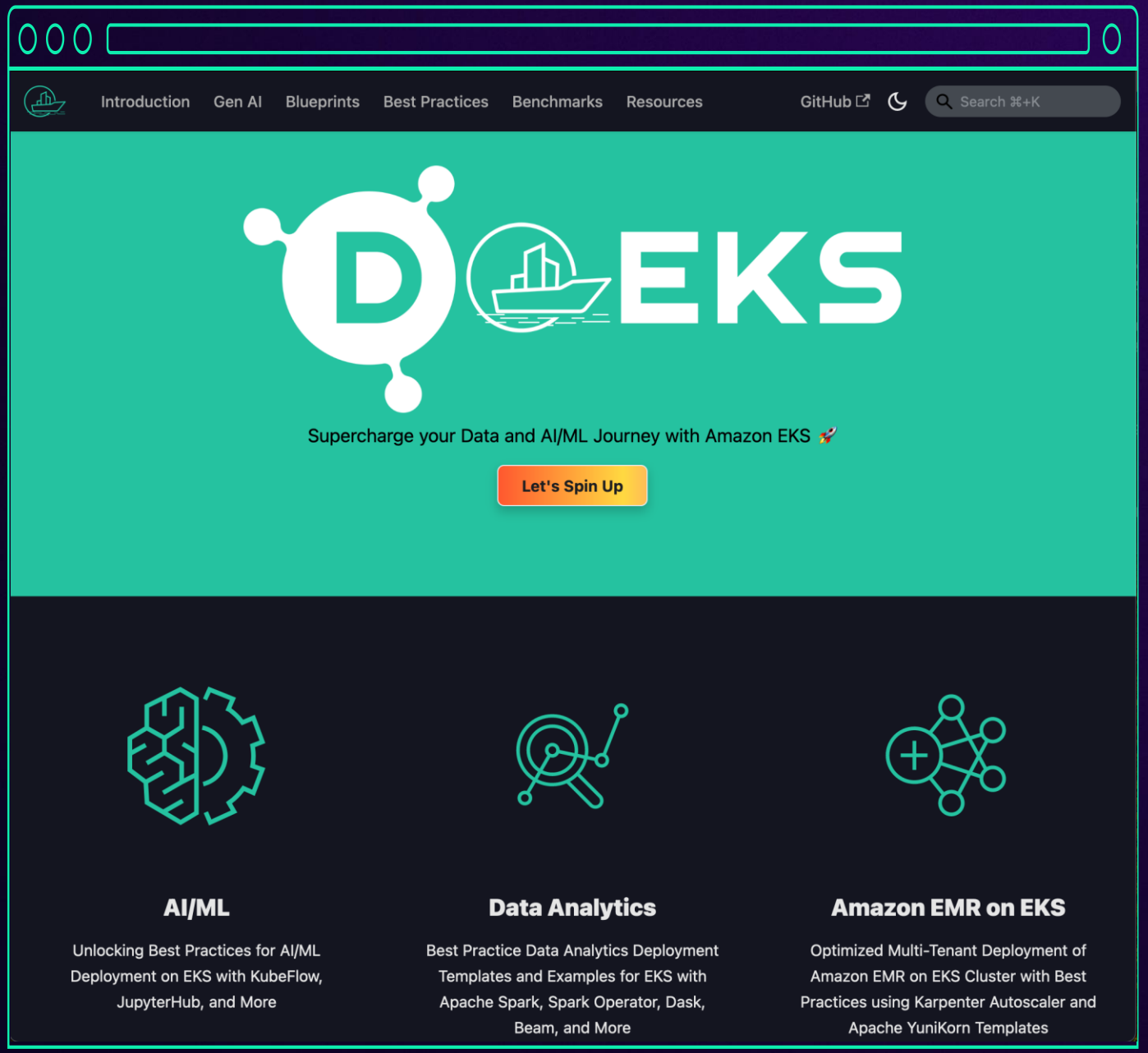
Enable shared applications

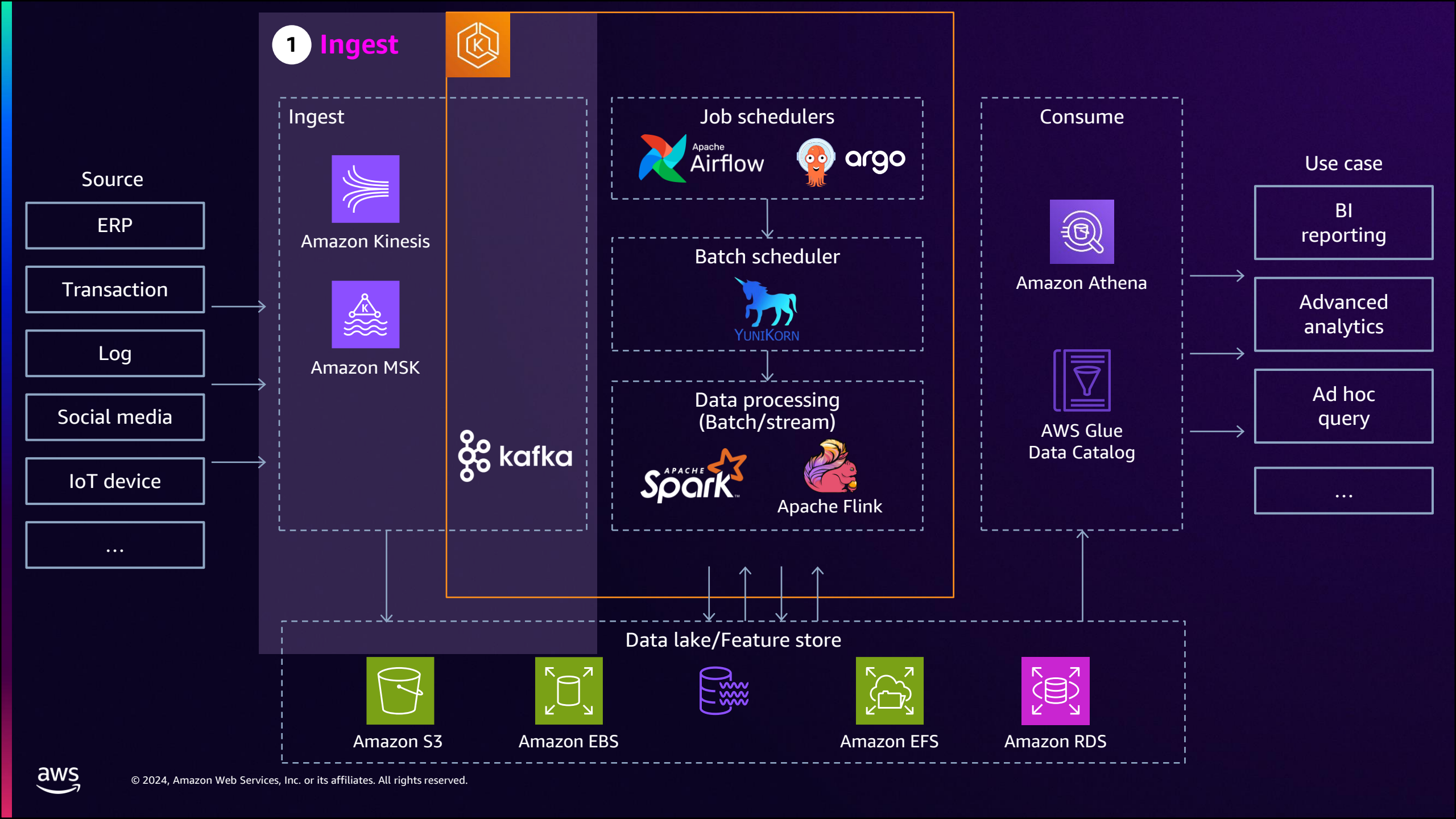
For customers that want to avoid managing and operating clusters and simply want to run applications using open-source frameworks

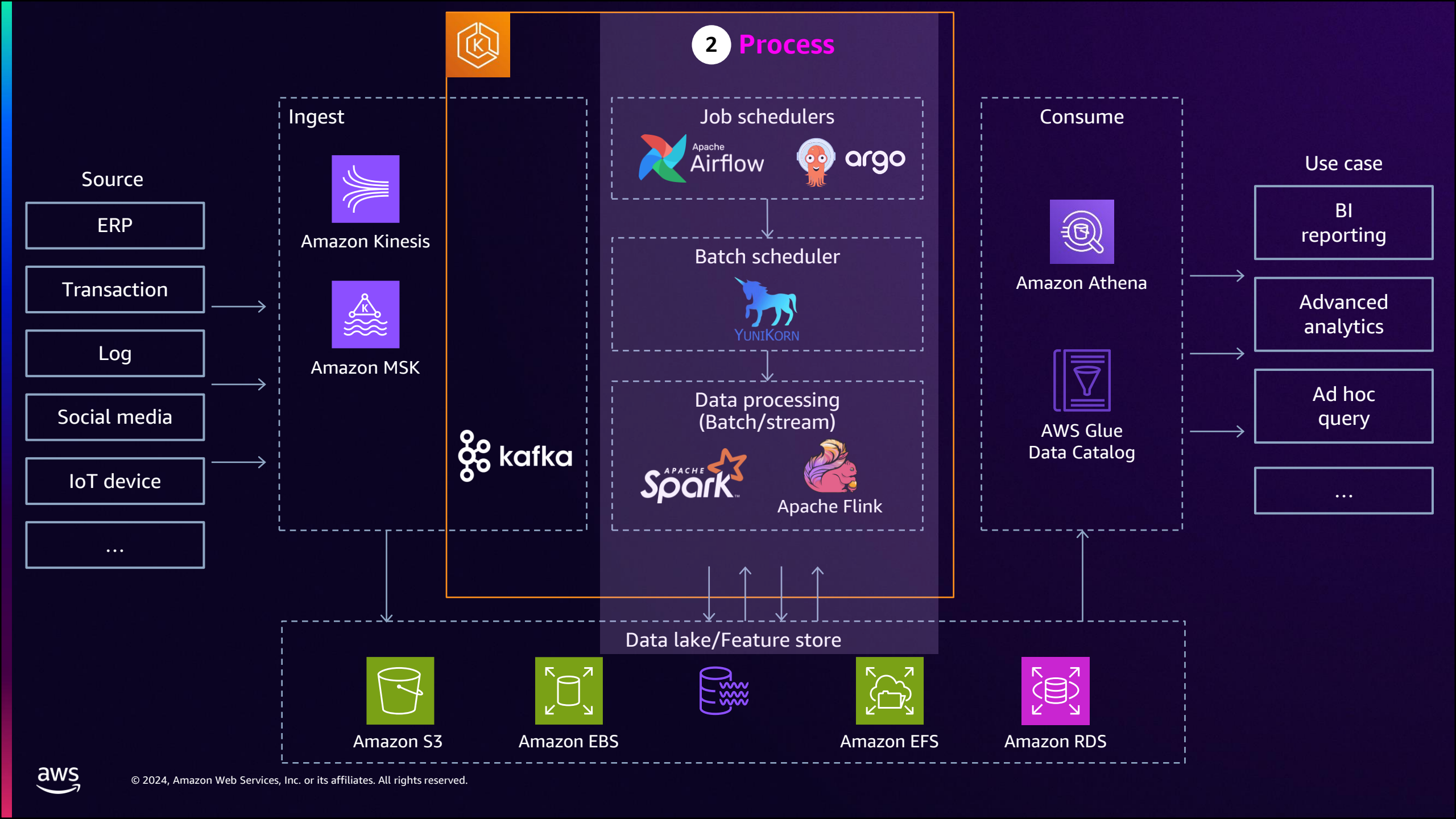
Kubernetes for data: How did we get here?

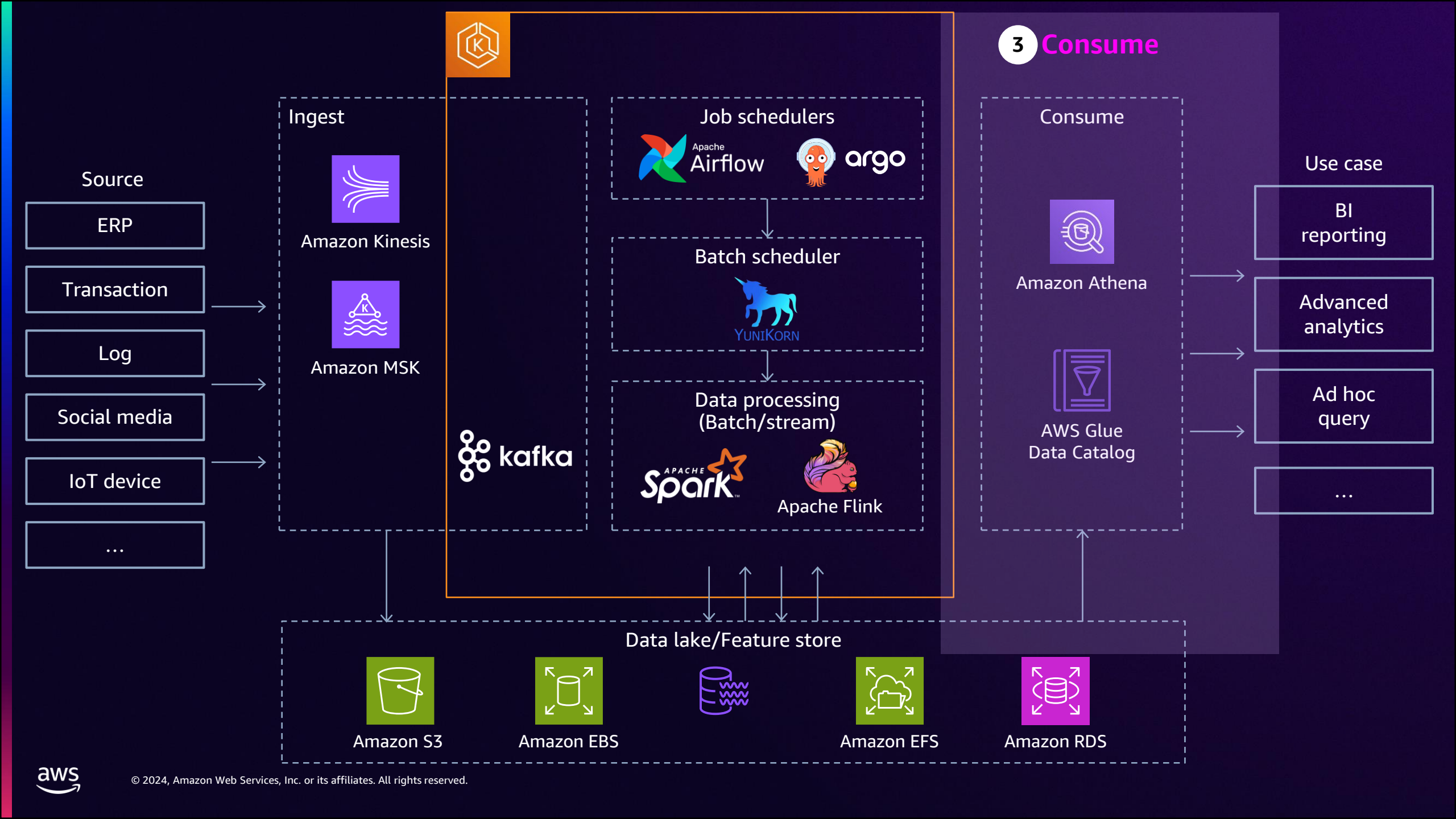


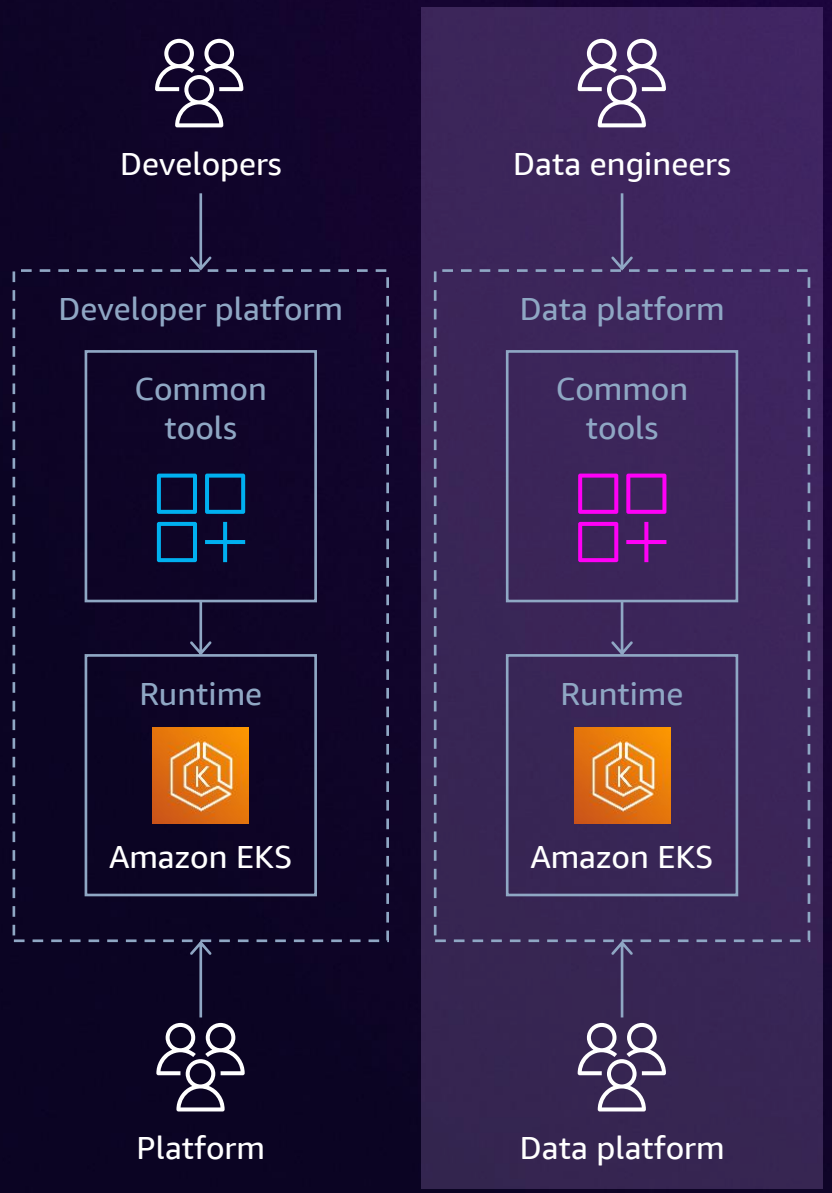
awslabs.github.io/data-on-eks









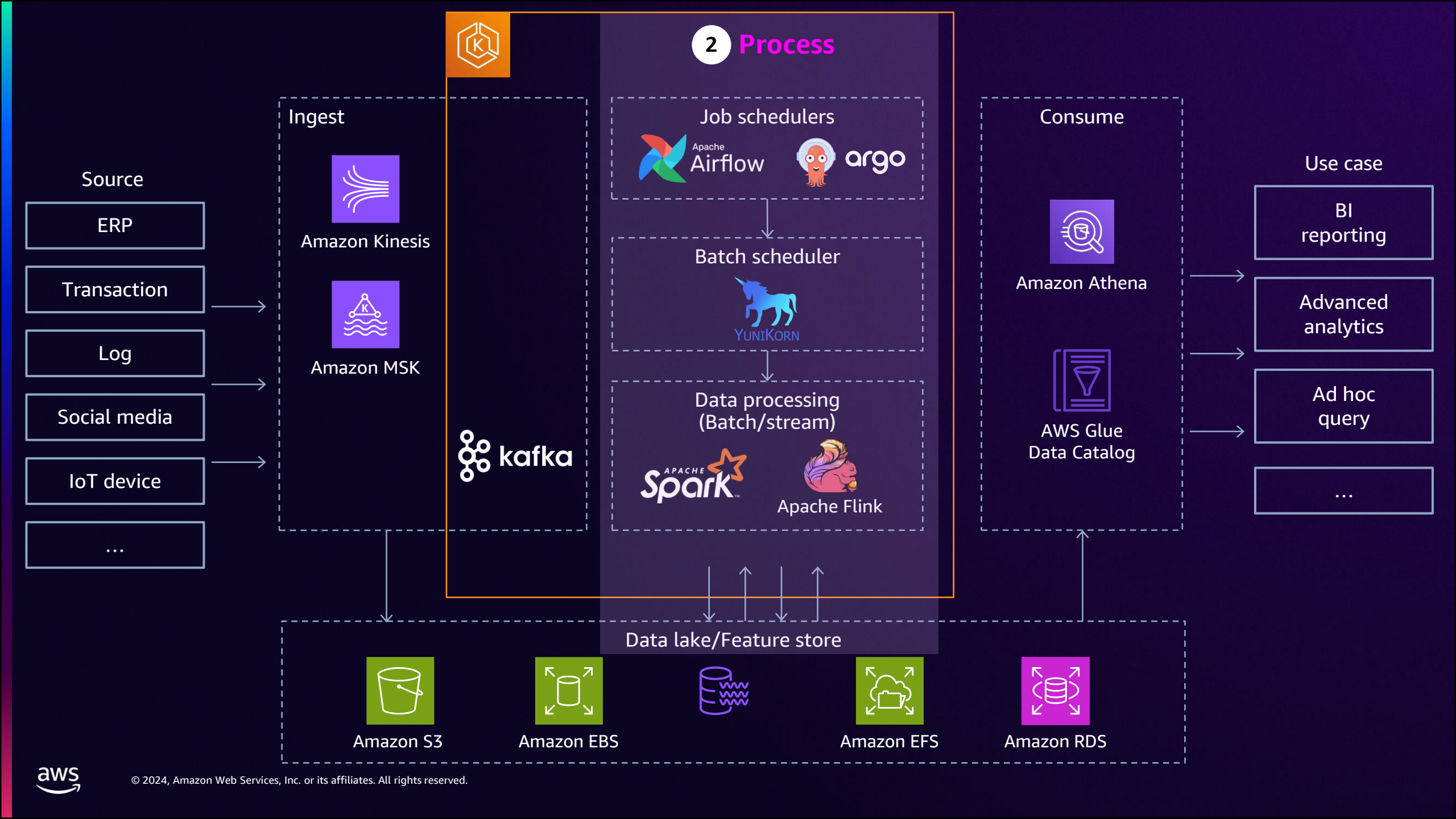


Data platform challenges

Scalability

Performance

Cost efficiency



Build production-ready EKS cluster

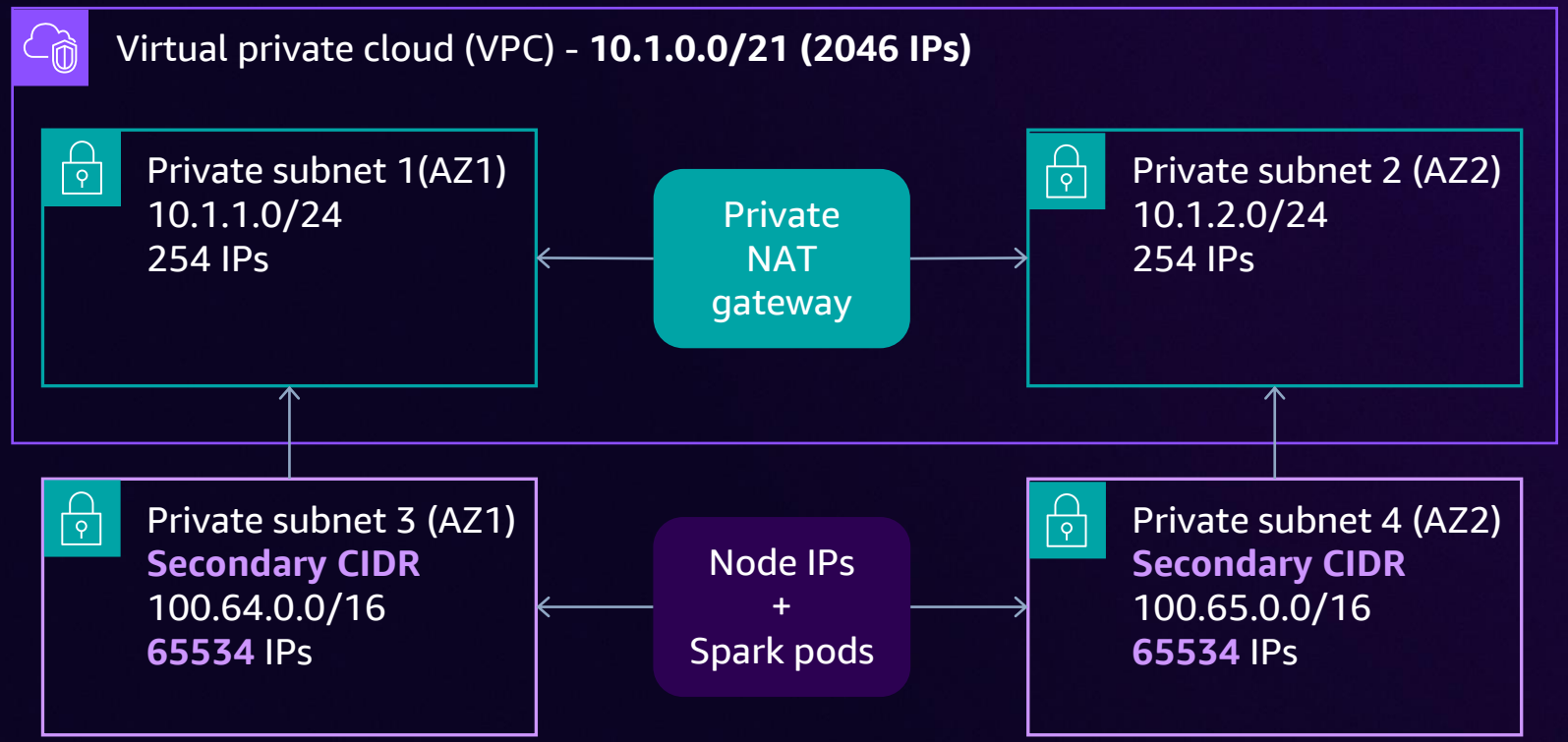
IPv4

1

Networking

Data platform

Amazon VPC with secondary CIDR



Avoid overlay networking

Use **non-routable** RFC 6598 range 100.64.0.0/10

Build production-ready EKS Cluster

IPv4

1

Networking

Data platform

Build production-ready EKS Cluster

IPv6

IPv4

1

Networking

Data platform

Build production-ready EKS Cluster

1

Networking

VPC CNI

Data platform

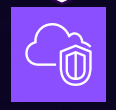
Build production-ready EKS Cluster

Default

WARM_ENI_TARGET=1

1

Networking



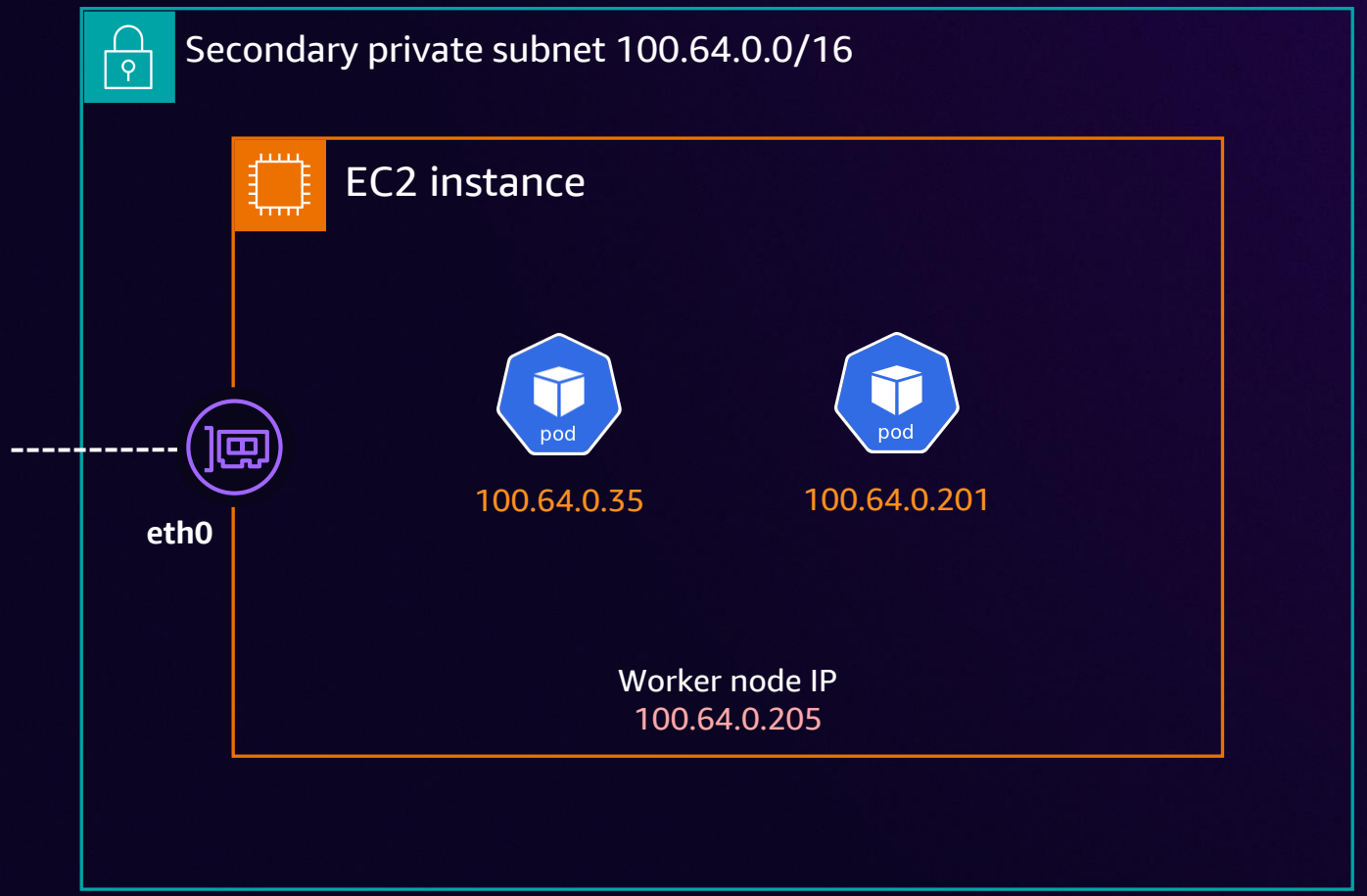
VPC CNI

Data platform

Amazon VPC CNI configuration

Default

WARM_ENI_TARGET=1



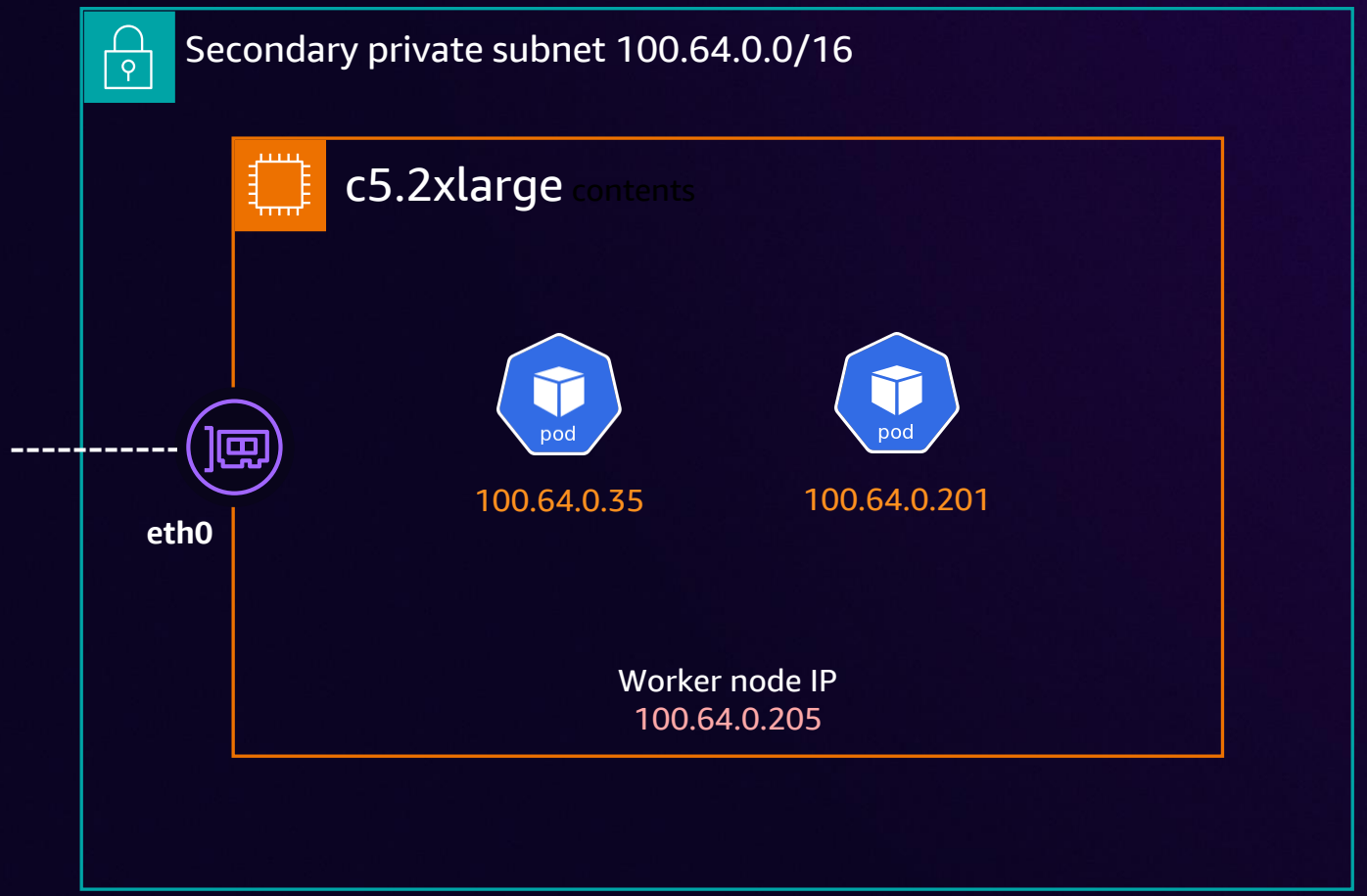
Primary IP
100.64.0.205

Secondary IPs
100.64.0.35
100.64.0.37
100.64.0.41
100.64.0.57
100.64.0.201
100.64.0.244
.
.
.

Amazon VPC CNI configuration

Default

WARM_ENI_TARGET=1



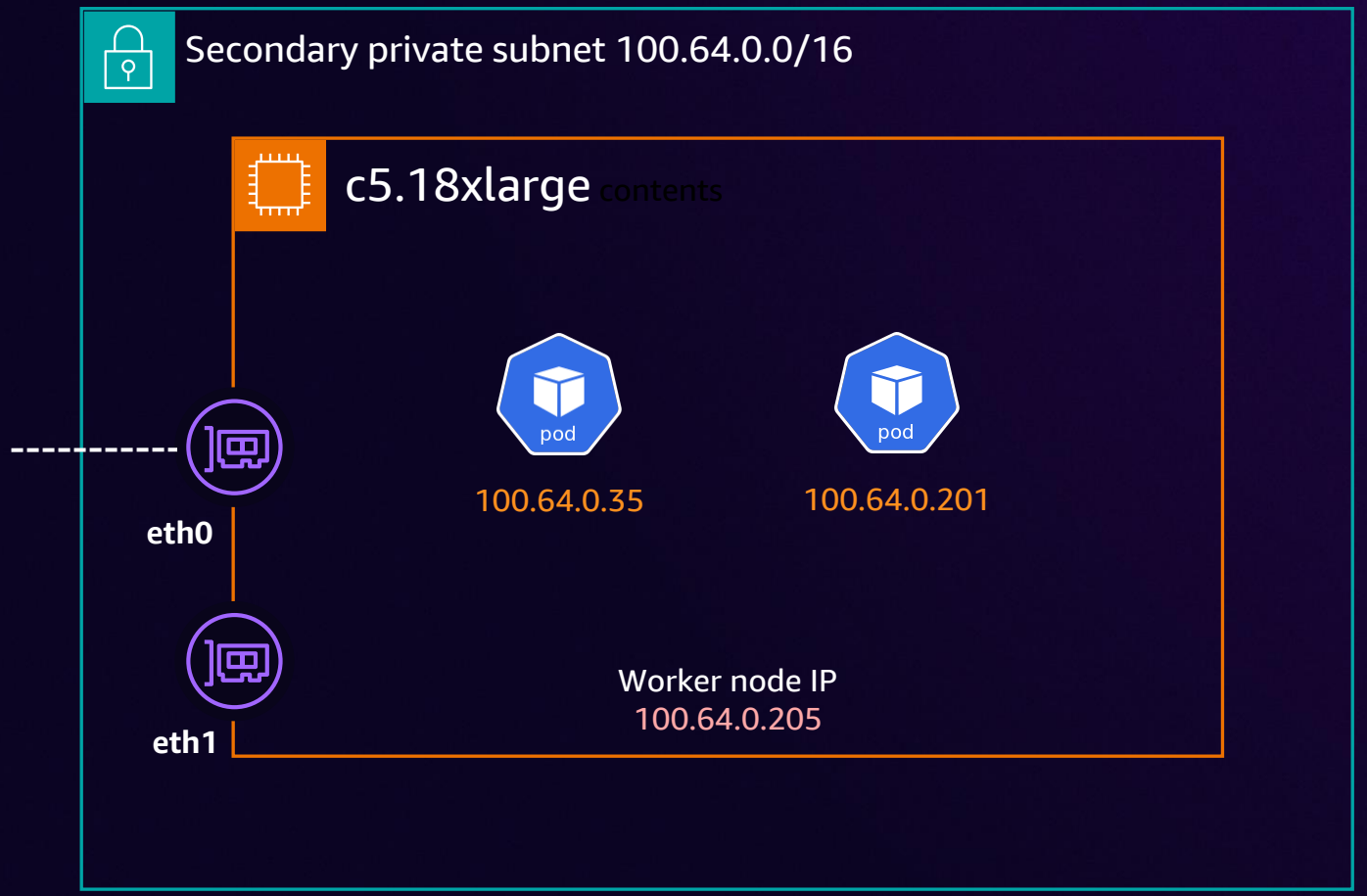
Primary IP
100.64.0.205

Secondary IPs
100.64.0.35
100.64.0.37
100.64.0.41
100.64.0.57
100.64.0.201
100.64.0.244
.
.
.

Amazon VPC CNI configuration

Default

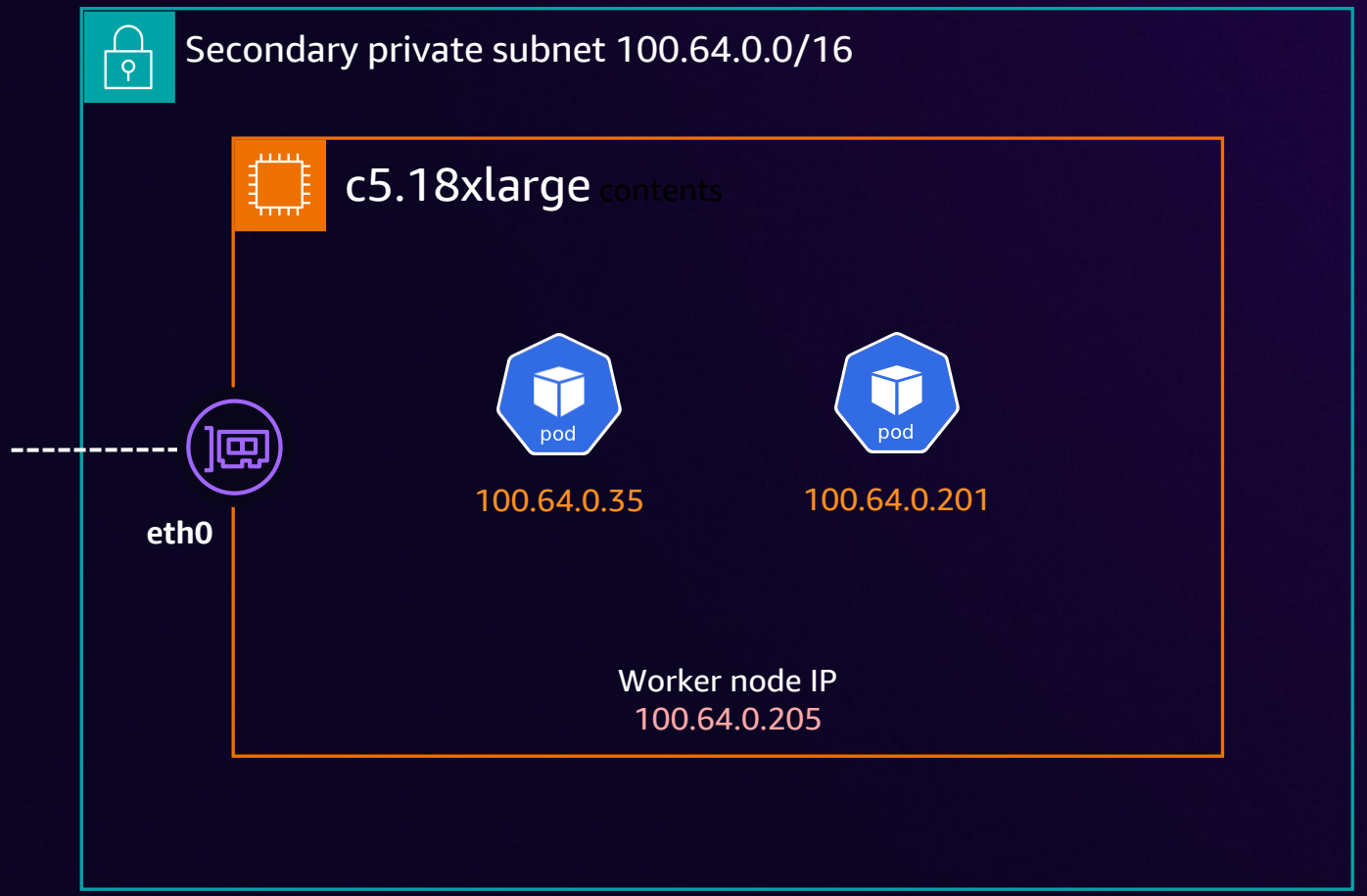
WARM_ENI_TARGET=1



Primary IP
100.64.0.205

Secondary IPs
100.64.0.35
100.64.0.37
100.64.0.41
100.64.0.57
100.64.0.201
100.64.0.244
.
.
.

Amazon VPC CNI configuration



Primary IP
100.64.0.205

Secondary IPs
100.64.0.35
100.64.0.37
100.64.0.41
100.64.0.57
100.64.0.201
100.64.0.244
.
.
.

Default

❌ WARM_ENI_TARGET=1

Large instance types

✅ MAX_ENI = 1
✅ maxPods: 30

High churn

✅ ENABLE_PREFIX_DELEGATION
❌ WARM_IP_TARGET
✅ MINIMUM_IP_TARGET=30

Build production-ready EKS cluster

1

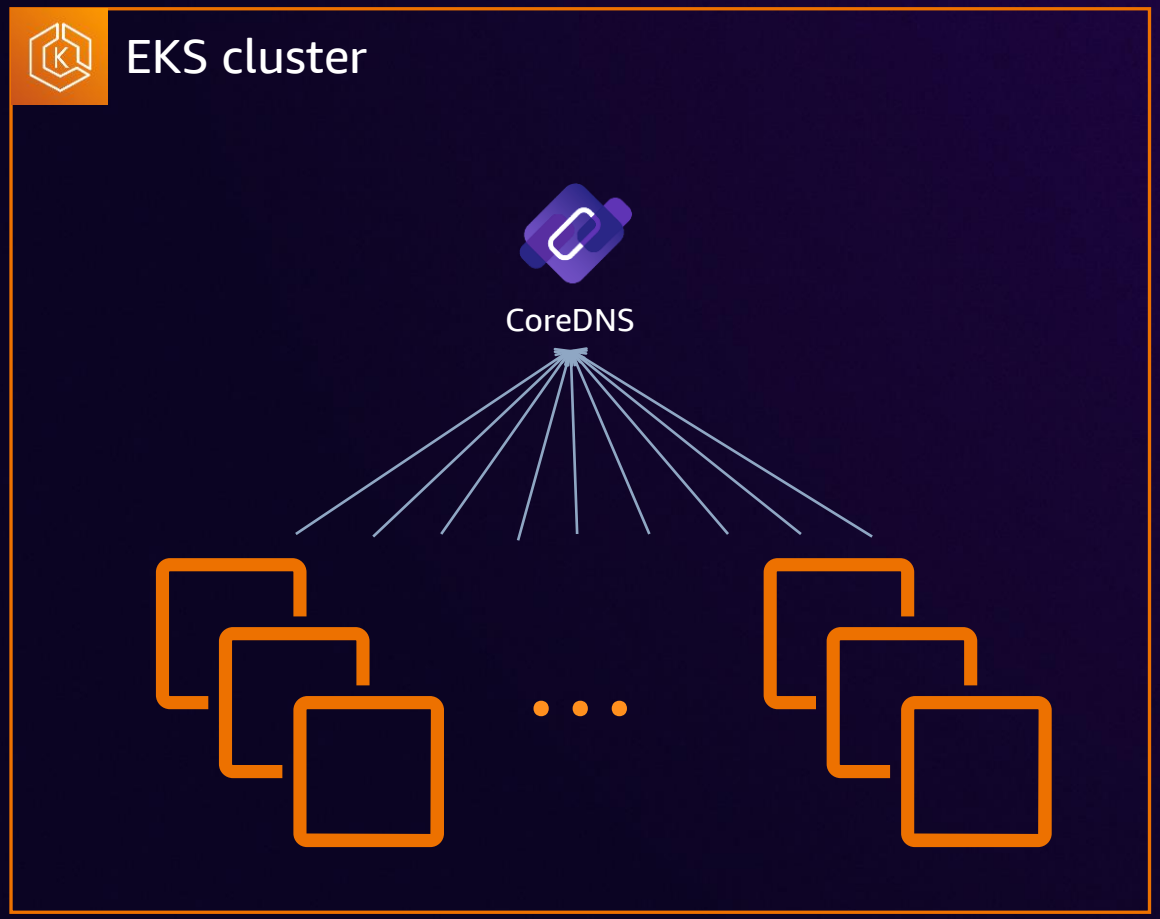
Networking

VPC CNI

CoreDNS

Data platform

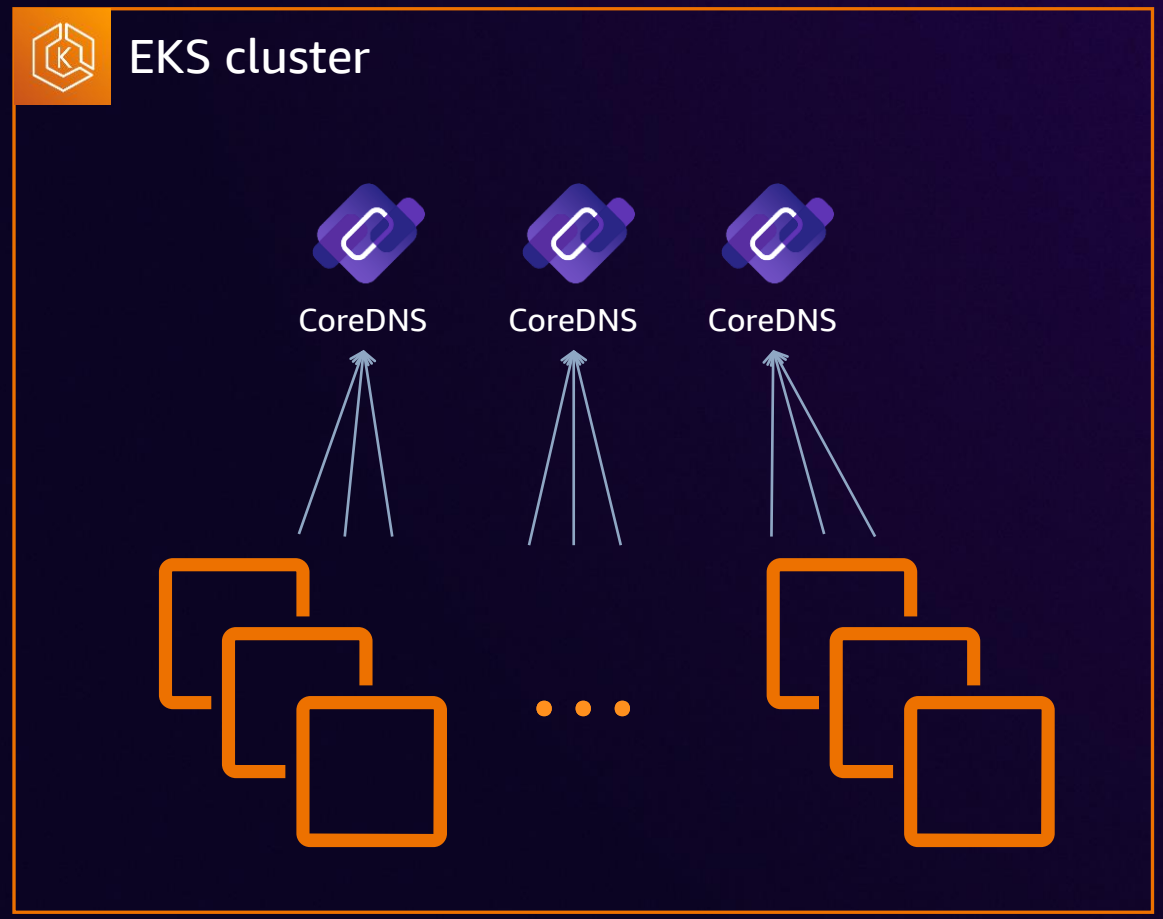
CoreDNS scaling



CoreDNS scaling

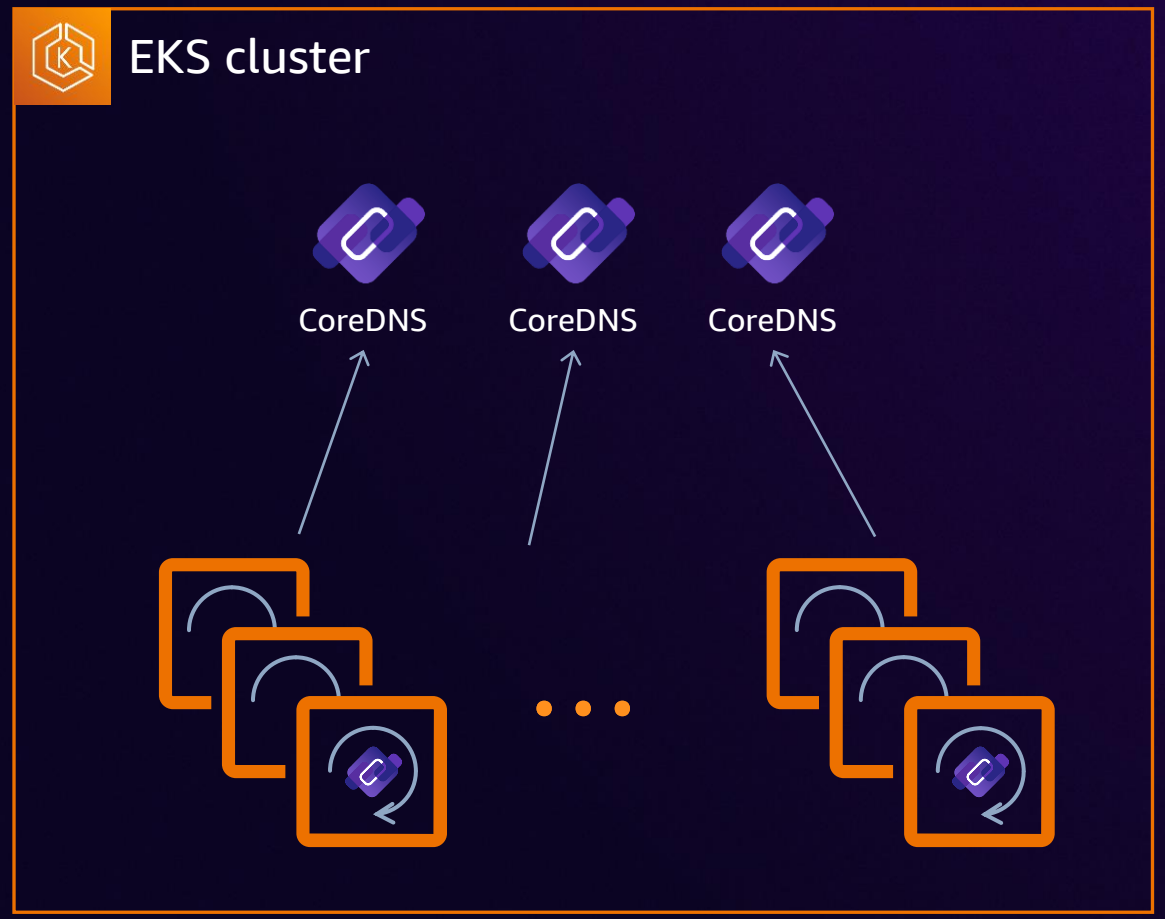
Managed scaling
EKS 1.25
CoreDNS 1.9

```
autoScaling:
  enabled: true
```



CoreDNS scaling

```
nameserver 10.100.0.10
search namespace.svc.cluster.local svc.cluster.local cluster.local ec2.internal
options ndots:5 ✅
```



Managed scaling
EKS 1.25
CoreDNS 1.9

NodeLocal DNS

ndots 2 **PODS ONLY**

s3.amazonaws.com ✅

Build production-ready EKS cluster

1

Networking

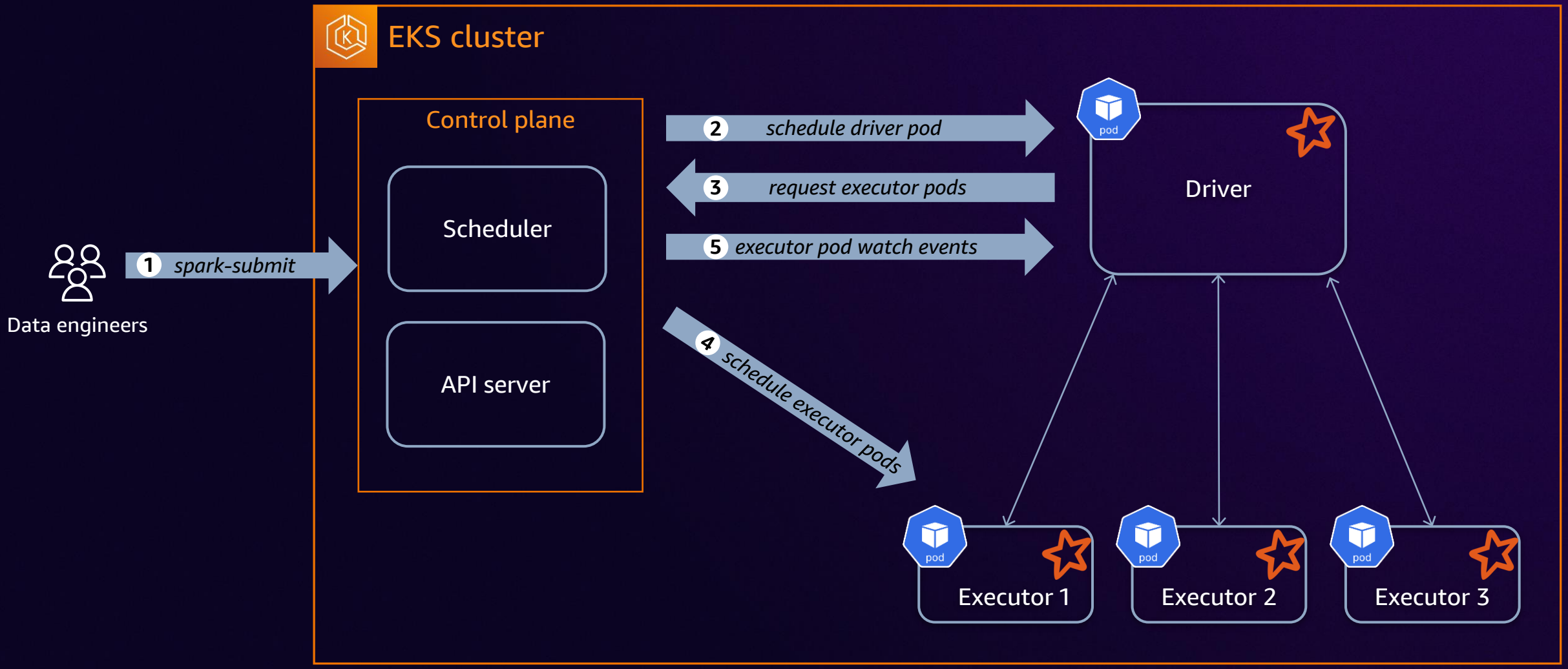
VPC CNI

CoreDNS

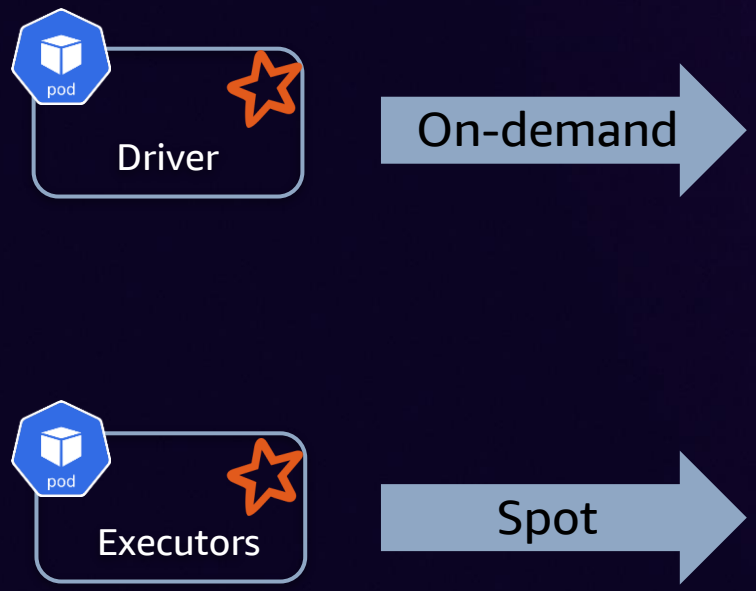
Karpenter

Data Platform

Spark job



Karpenter NodePool




```yaml
apiVersion: karpenter.sh/v1
kind: NodePool
metadata:
  name: spark-driver-graviton
spec:
  template:
    spec:
      requirements:
        - key: "karpenter.sh/capacity-type"
          operator: In
          values: ["spot", "on-demand"]
        - key: "karpenter.k8s.aws/instance-family"
          operator: In
          values: ["r6gd", "m5gd", "c5gd"]
       nodeClassRef:
          name: spark-graviton
  disruption:
    consolidationPolicy: WhenUnderutilized
```

Enable spot termination with Amazon SQS queue

Build production-ready EKS cluster

1


Networking


VPC CNI

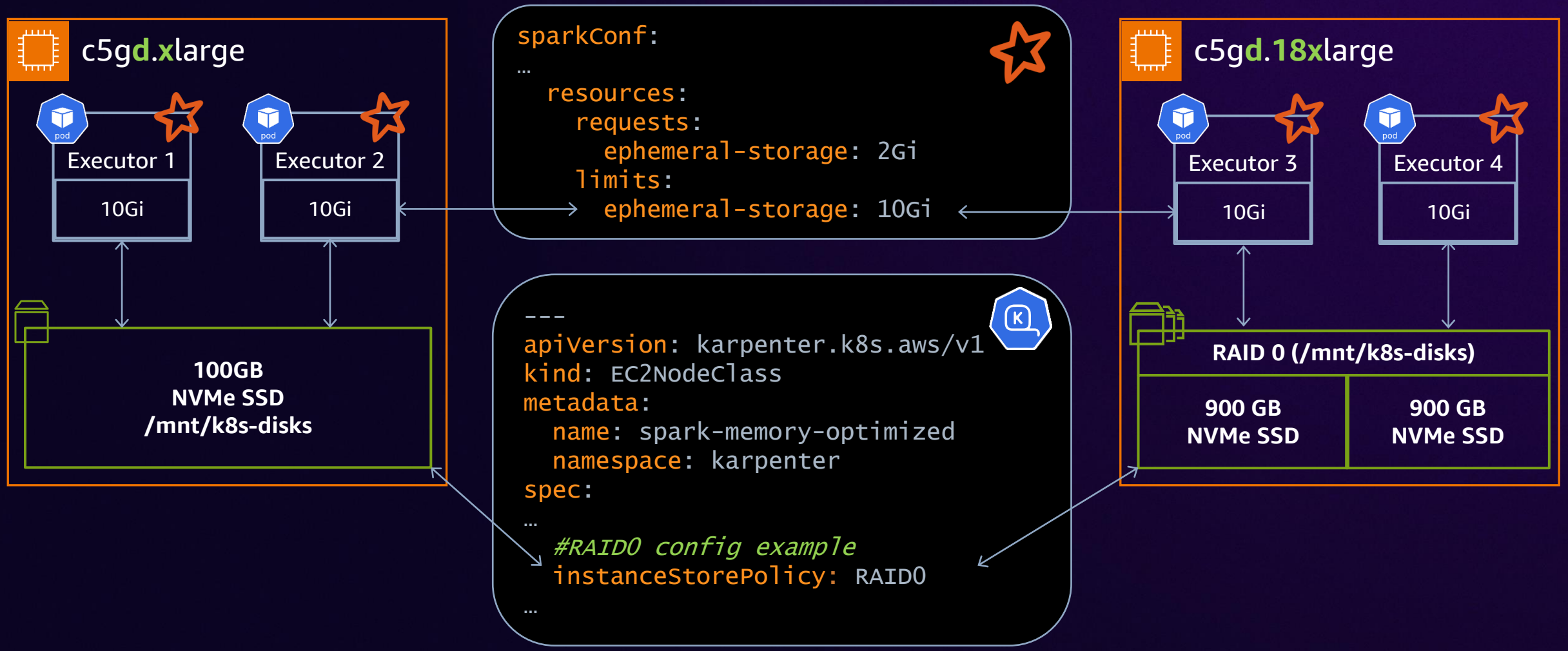

CoreDNS


Karpenter

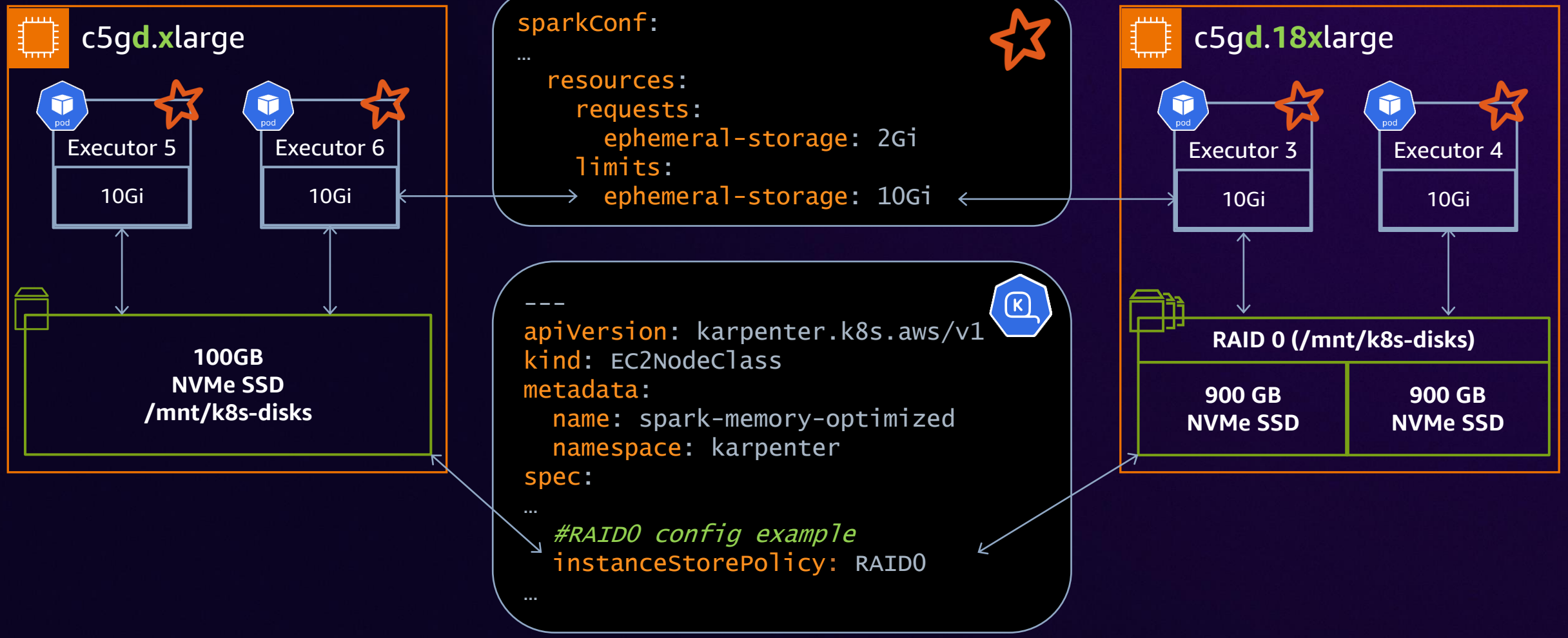

Storage


Data platform

Spark shuffle storage with NVMe SSD



Spark shuffle storage with NVMe SSD



Build production-ready EKS cluster

1


Networking


VPC CNI


CoreDNS


Karpenter

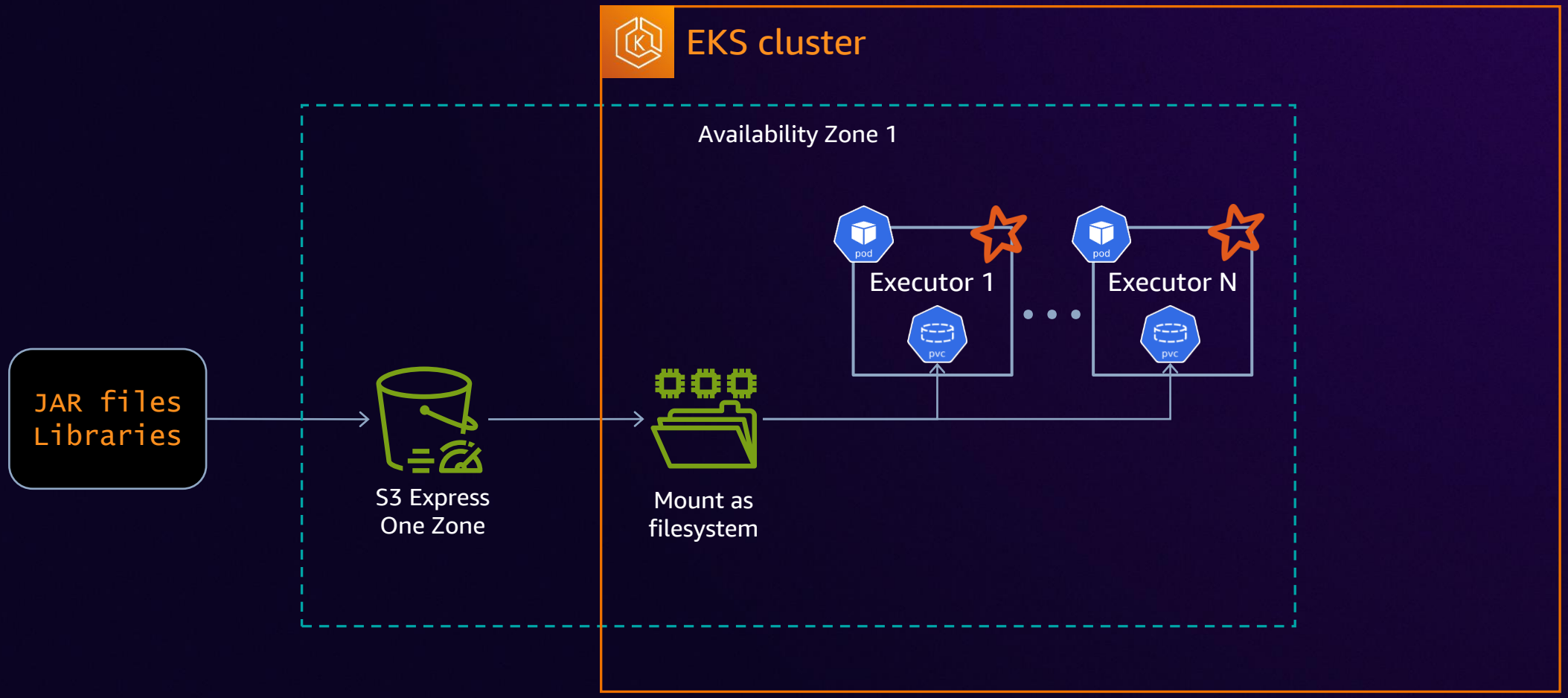

Storage


Amazon ECR

Cross-Region replication
VPC endpoints


Data platform

Reduce container size using S3-Mountpoint



Build production-ready EKS cluster

Network

AWS APIs

EKS control plane

1


Networking


VPC CNI


CoreDNS


Karpenter


Storage


Amazon ECR


Monitoring


Data platform

Build production-ready EKS cluster

1


Networking


VPC CNI


CoreDNS


Karpenter


Storage


Amazon ECR


Monitoring


Data platform

Optimized clusters for data processing

Scalability

Performance

Cost efficiency

Purpose-built cluster

2


Apache Spark

or


Apache Flink


Apache Airflow

or


Argo Workflows

1


Networking


VPC CNI


CoreDNS


Karpenter


Storage

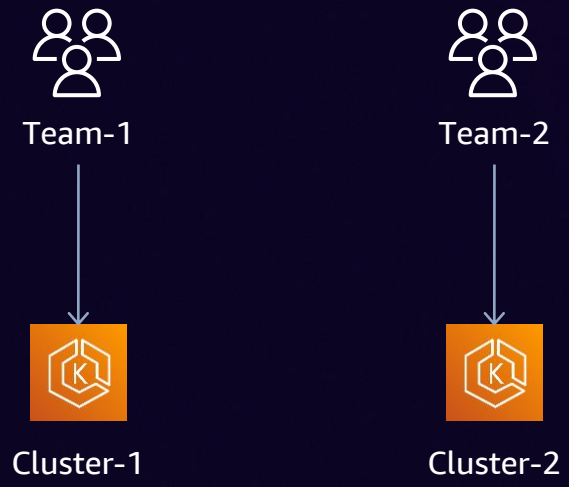

Amazon ECR


Monitoring

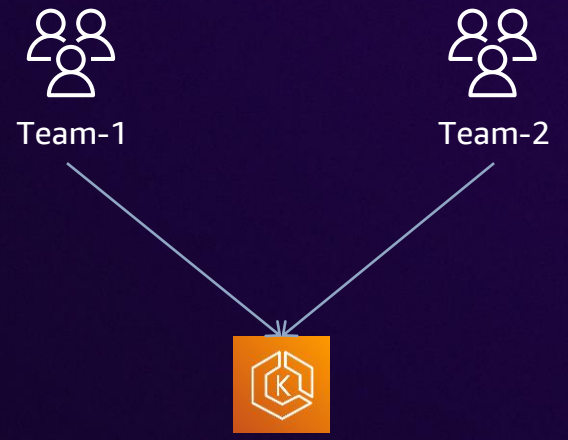

Data platform

Tenant isolation

Cluster as a service



Namespace as a service



Namespace as a service

3

Namespace
Spark API
NodePool
Service account
Roles
Policies
...
Namespace
Spark API
NodePool
Service account
Roles
Policies

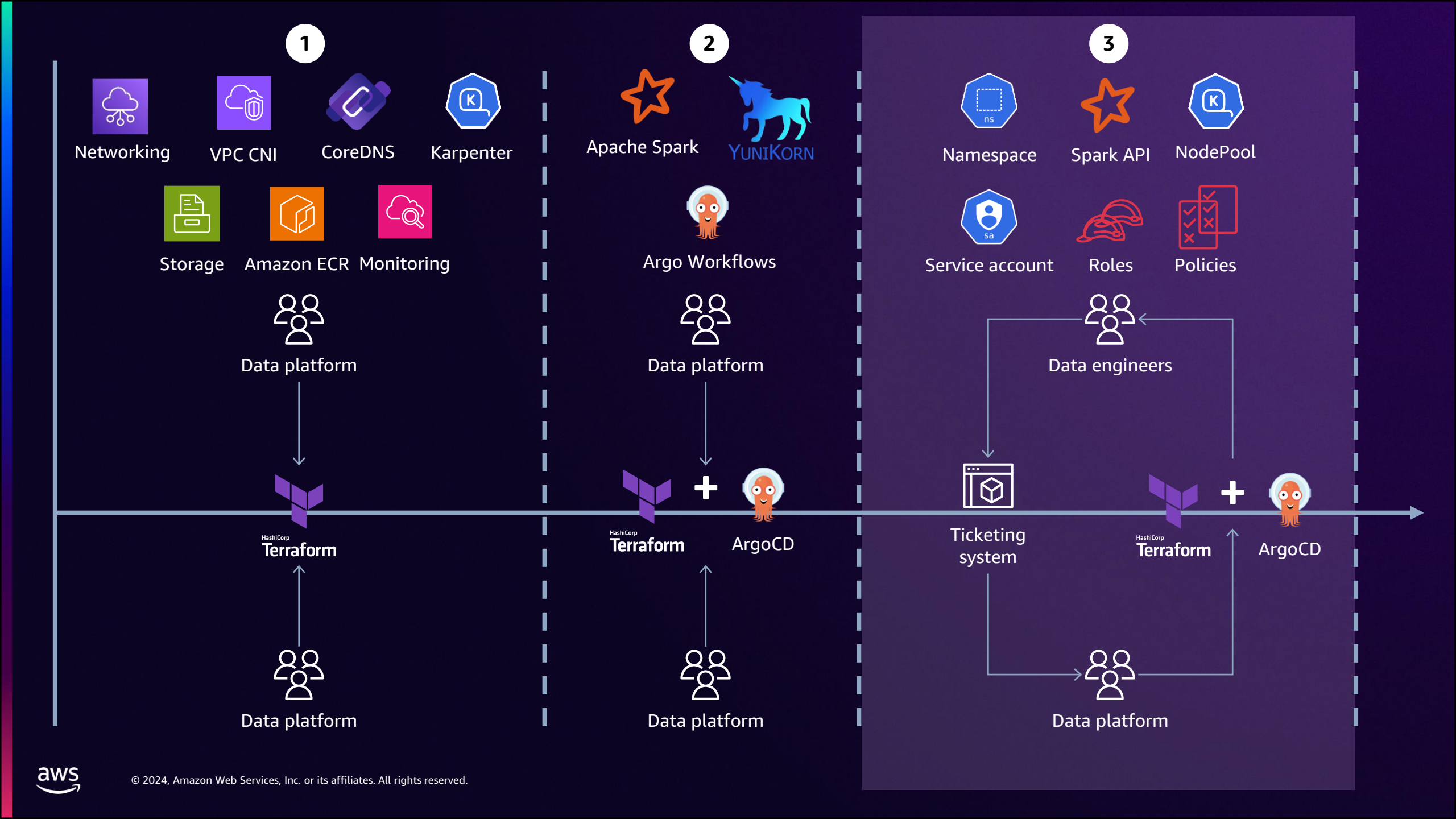
2

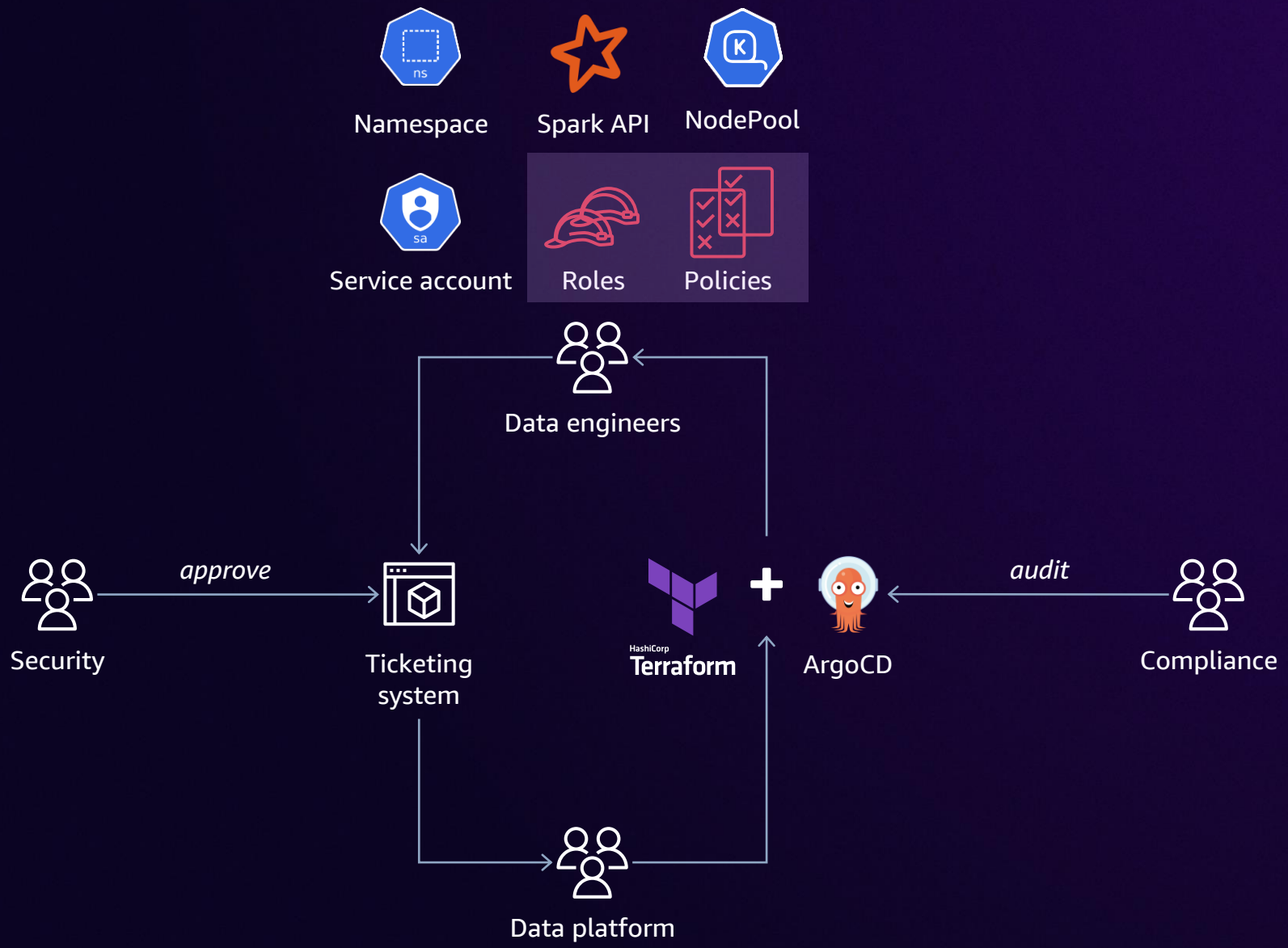
Apache Spark or Apache Flink
YuniKorn
Apache Airflow or Argo Workflows

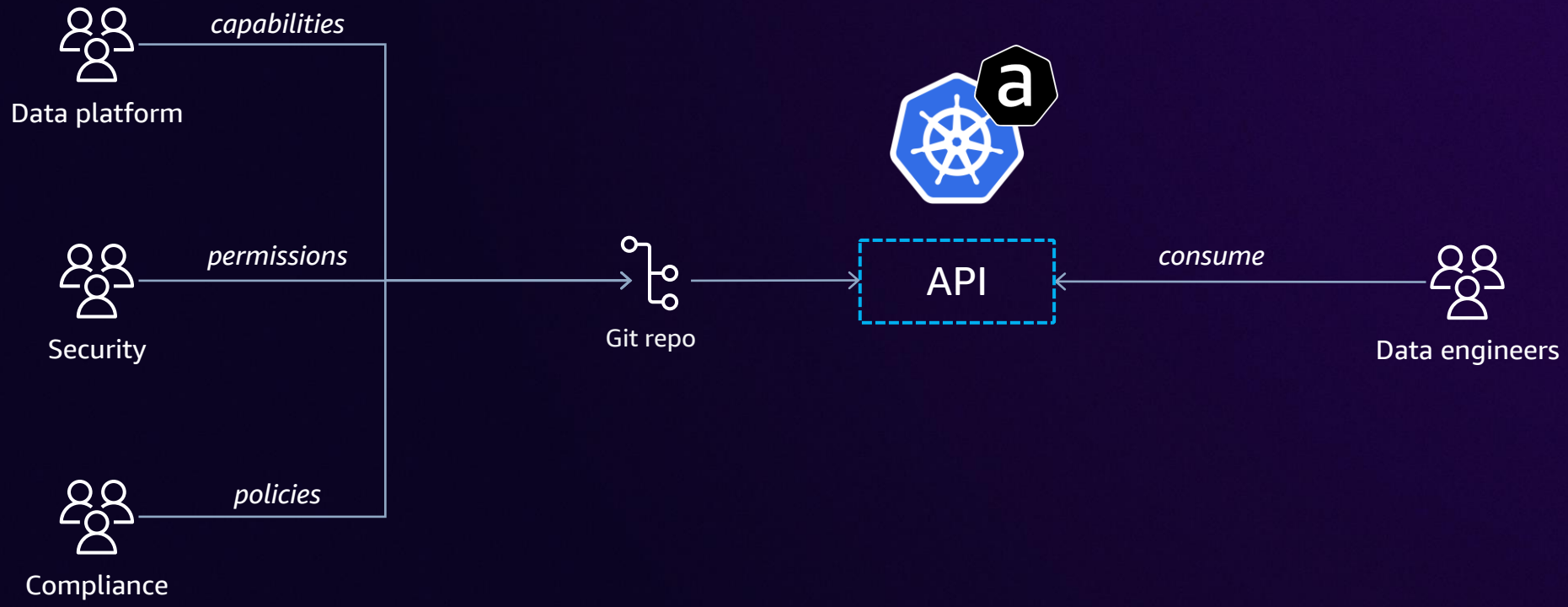
1

Networking
VPC CNI
CoreDNS
Karpenter
Storage
Amazon ECR
Monitoring

Data platform







AWS Controllers for Kubernetes (ACK)

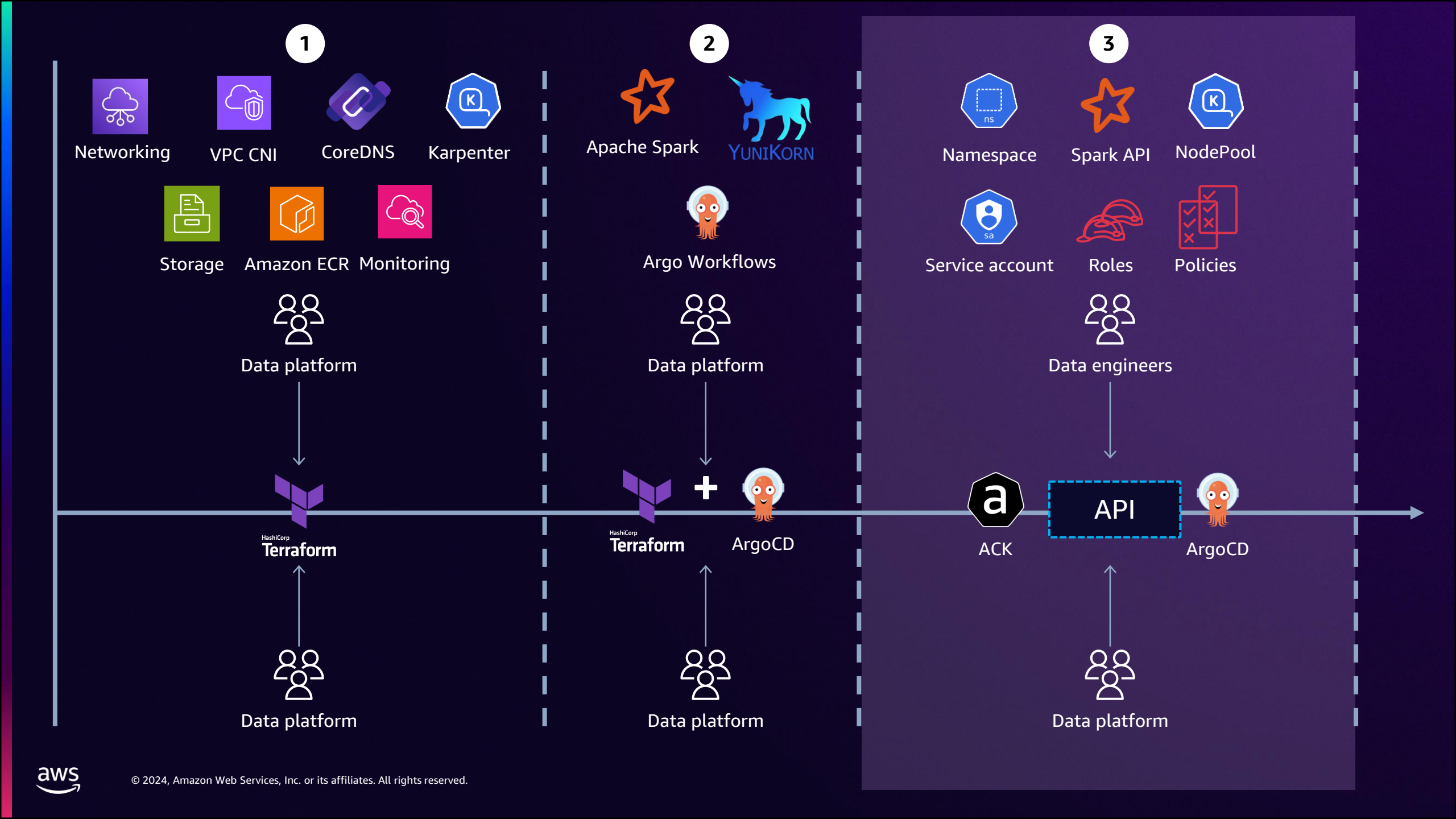
💙 Open source 💙

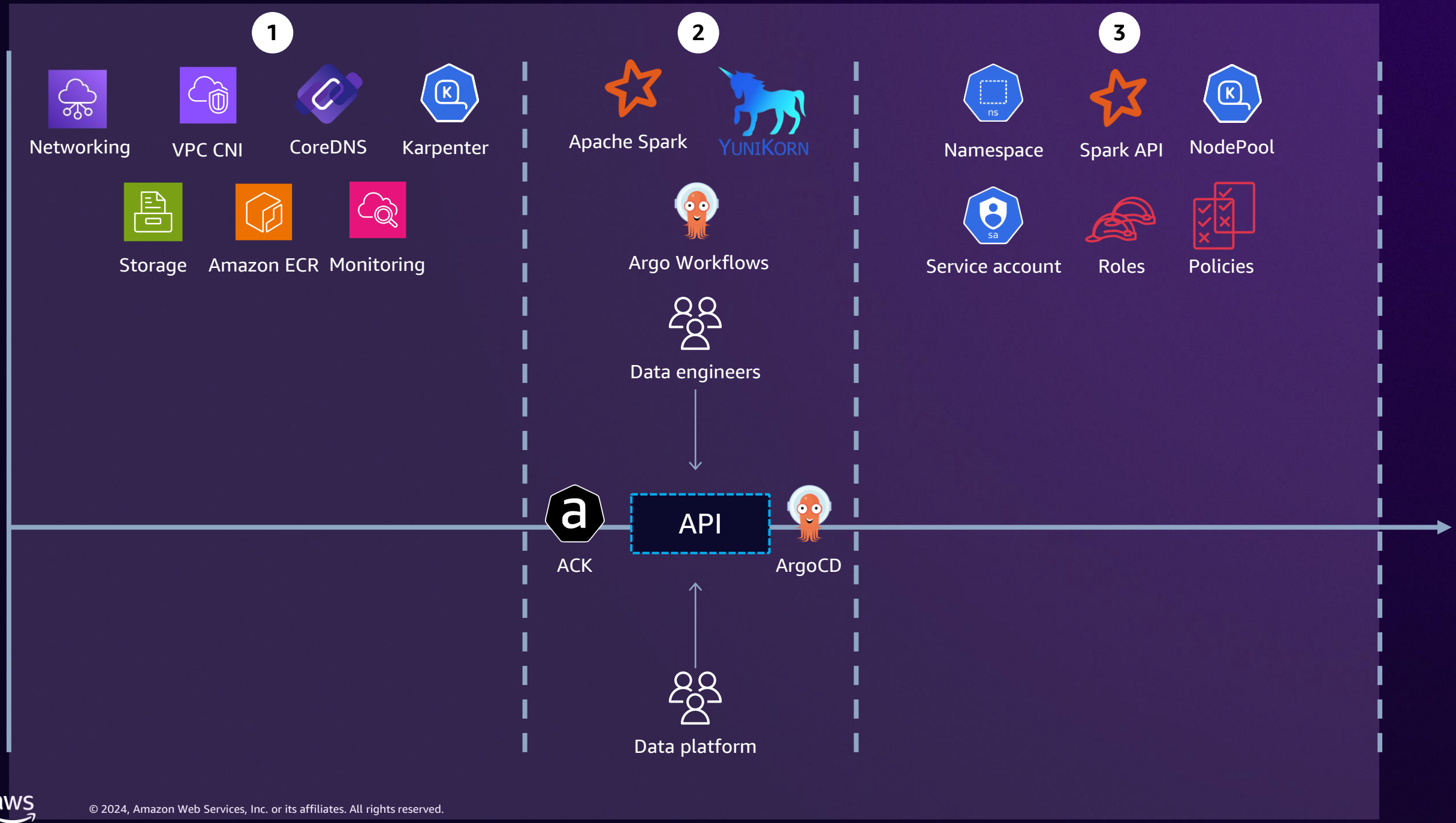
Developed and maintained by AWS

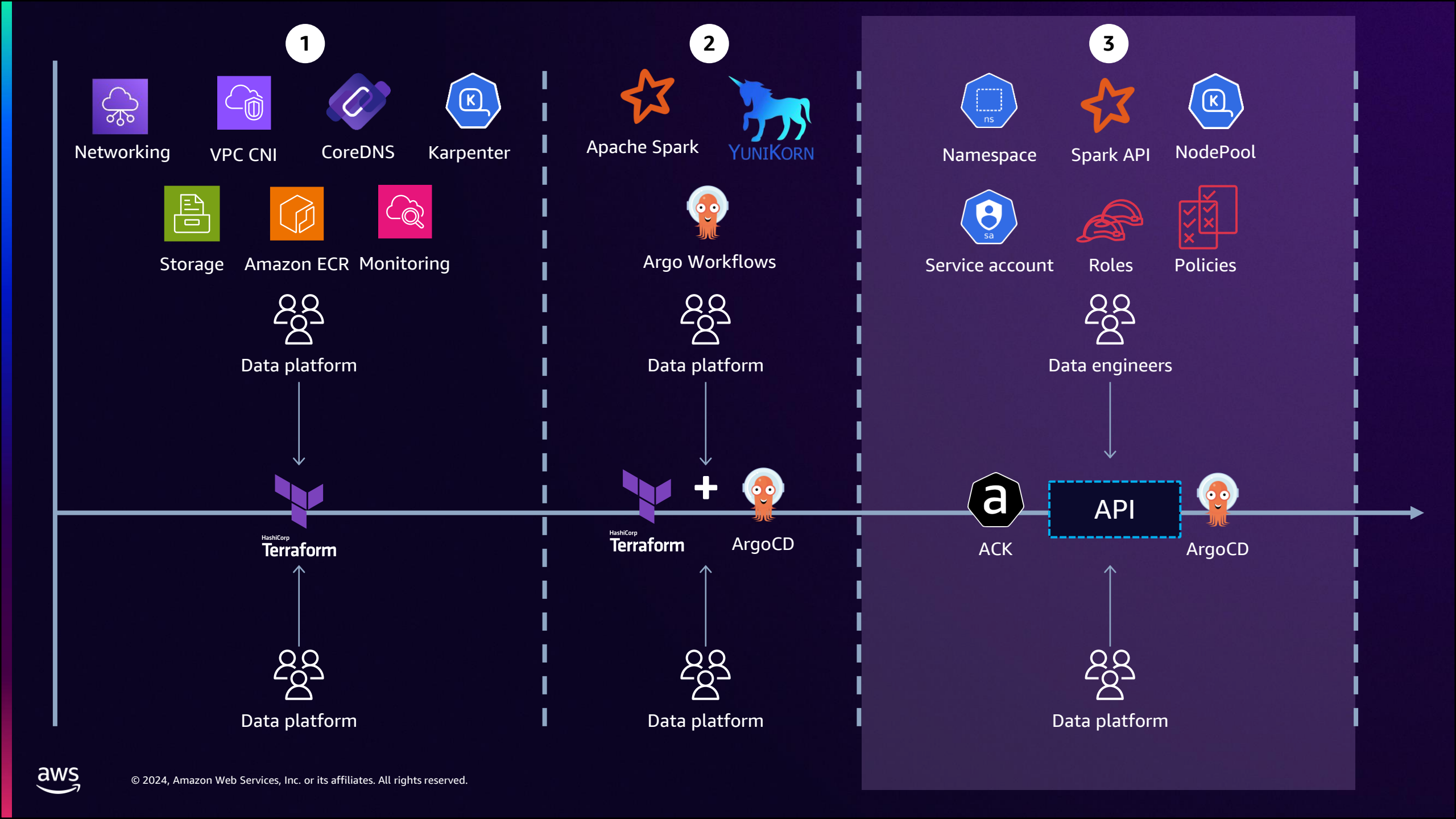
Supports mainstream AWS services

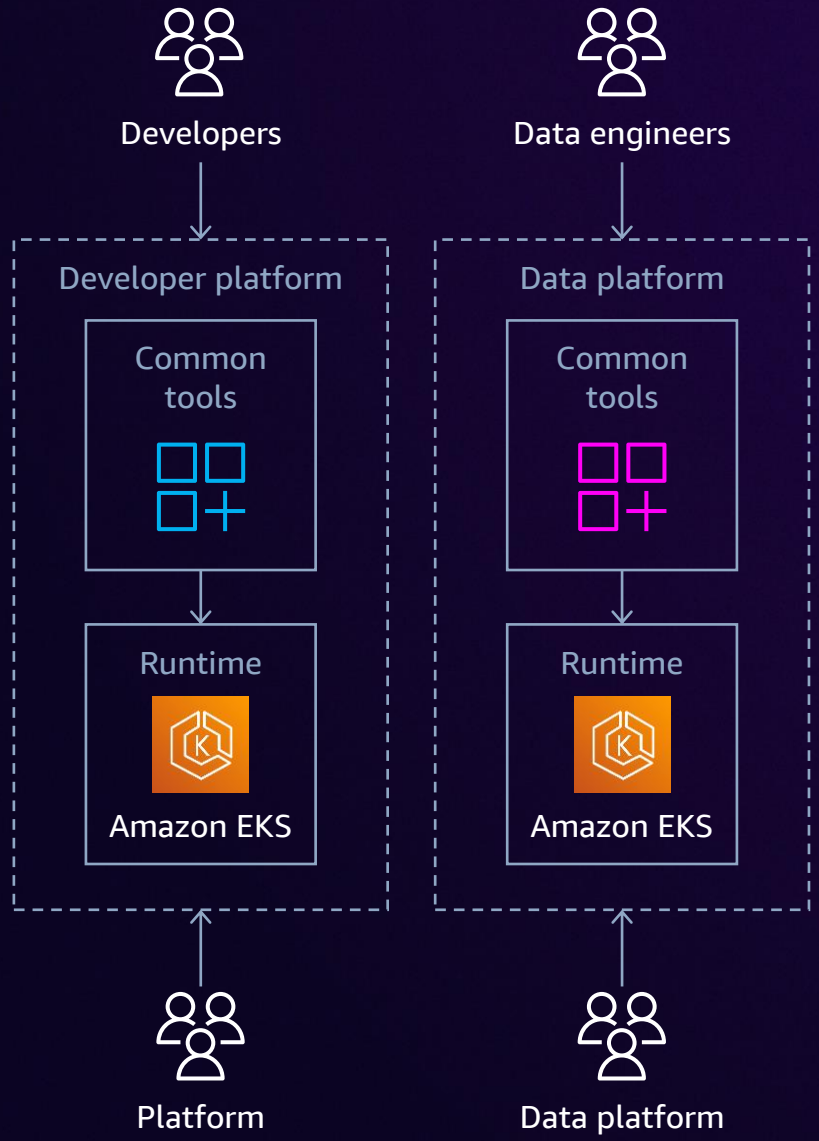
Simple operational and security model

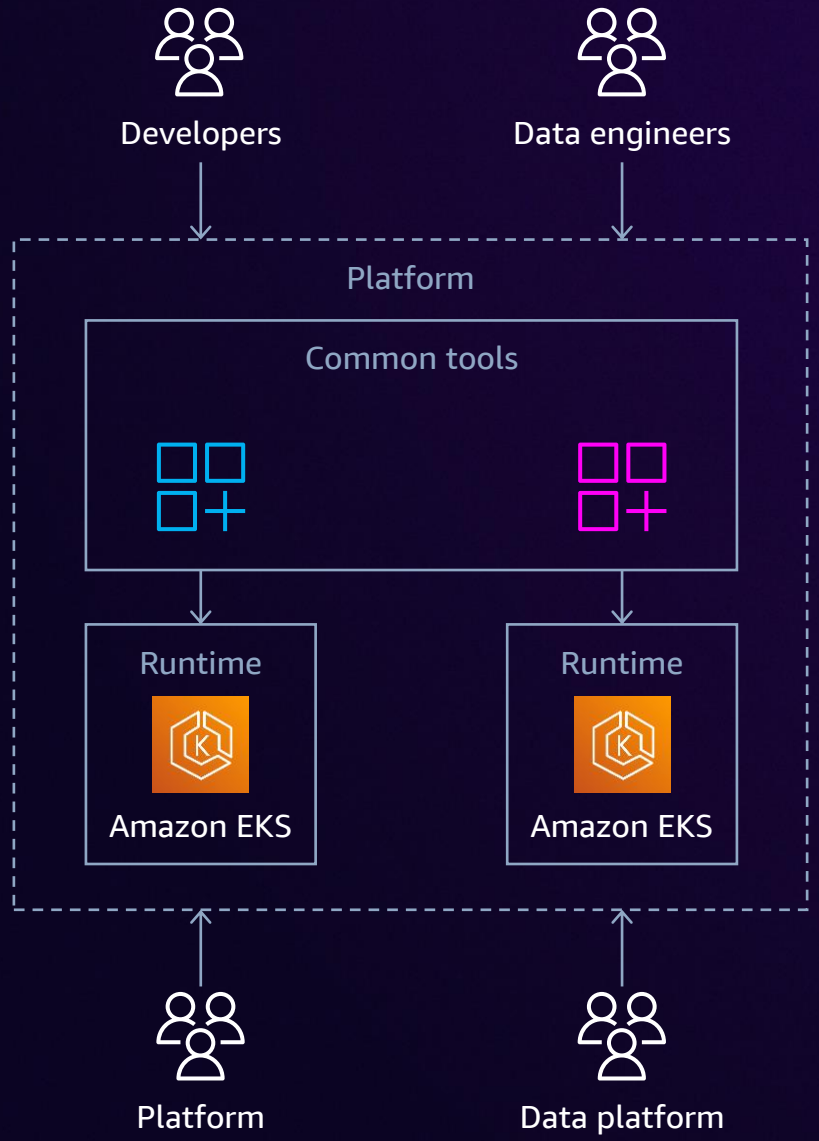
AWS support through service team and specialists

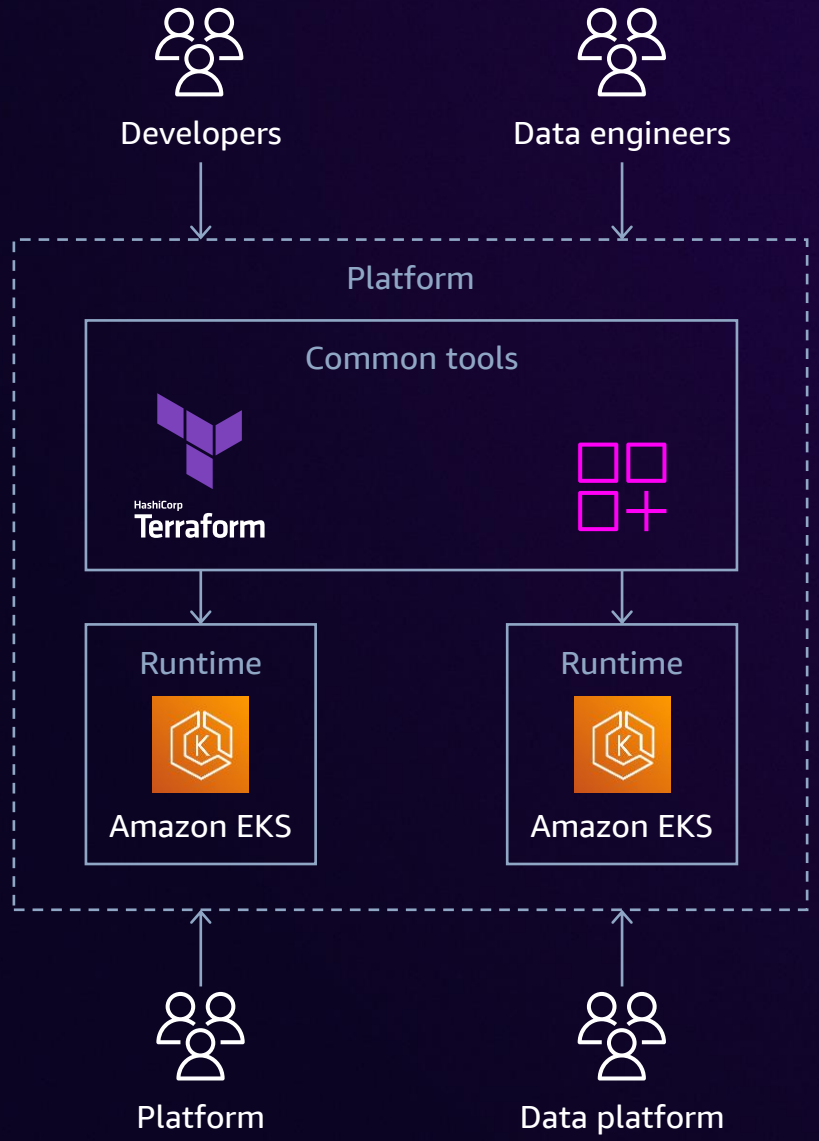


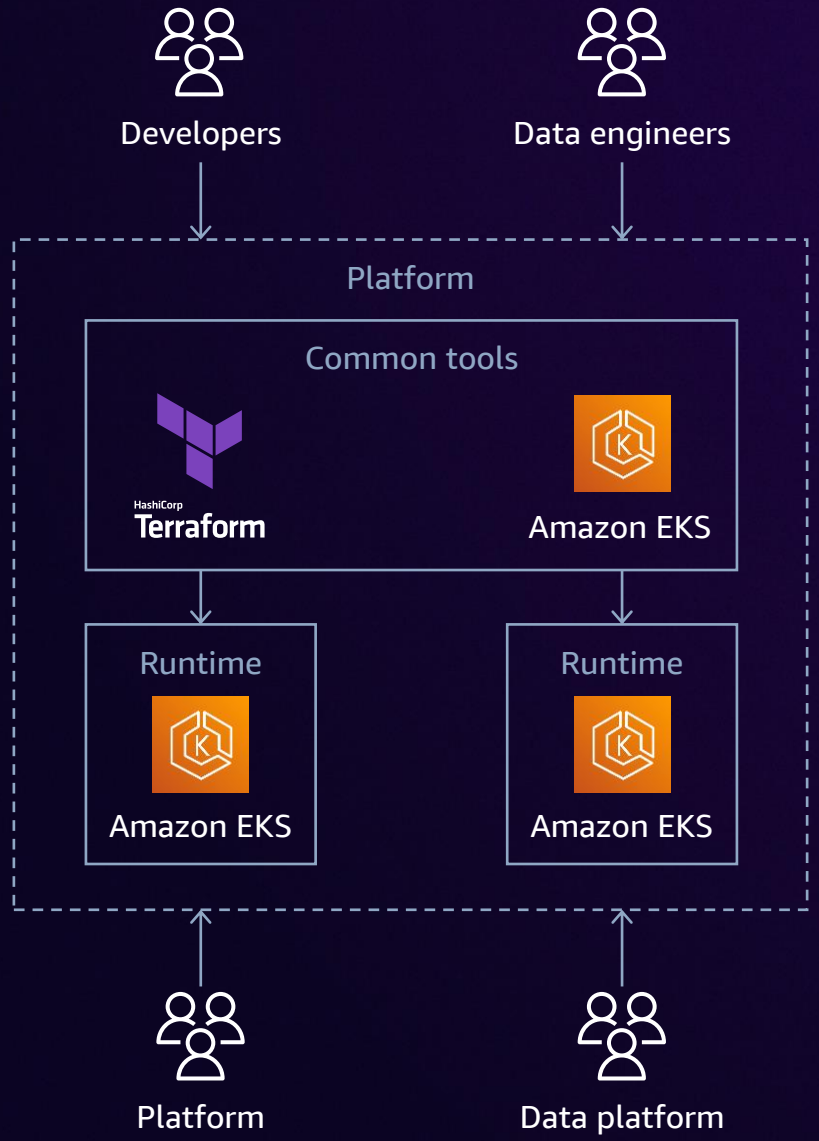


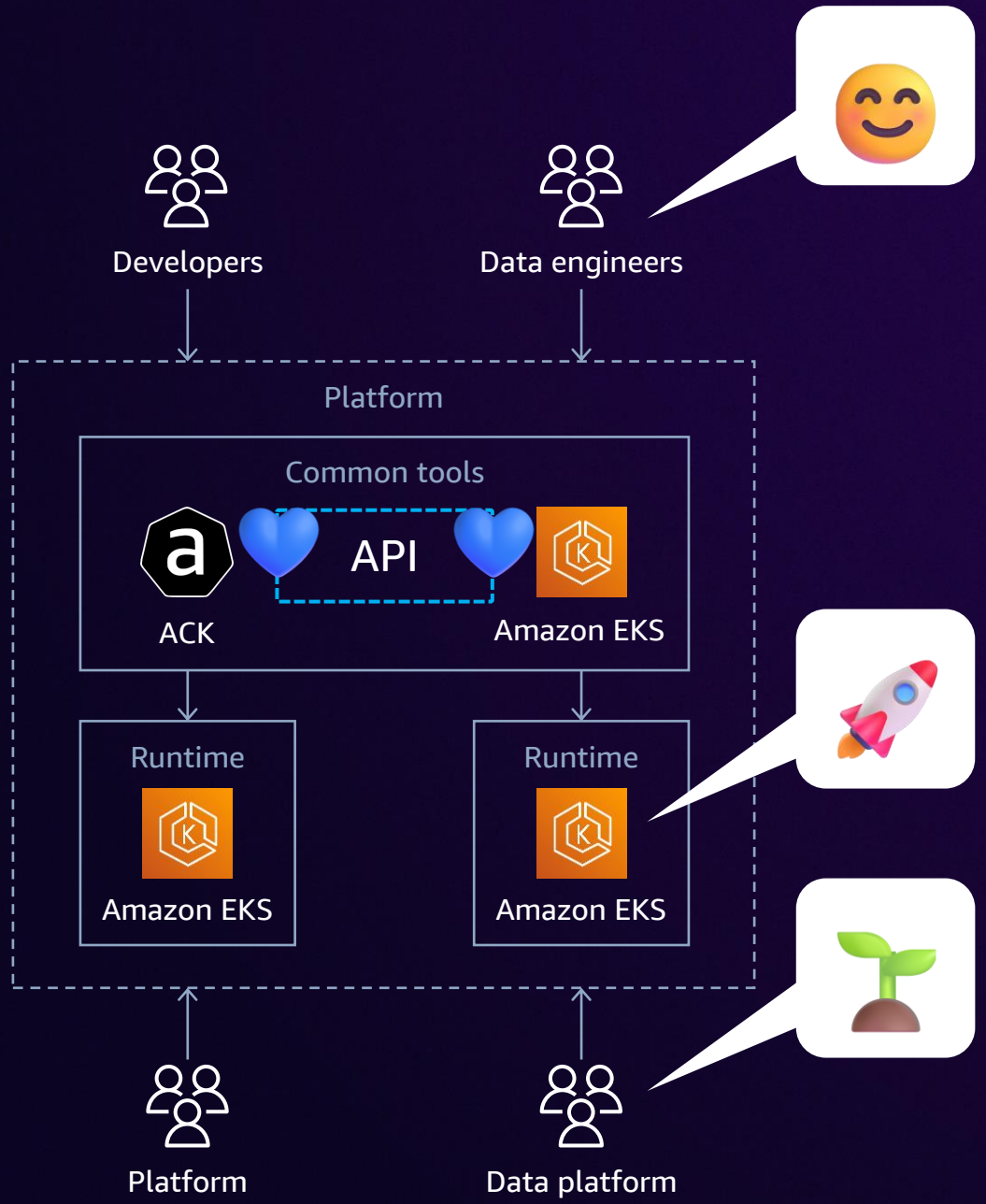










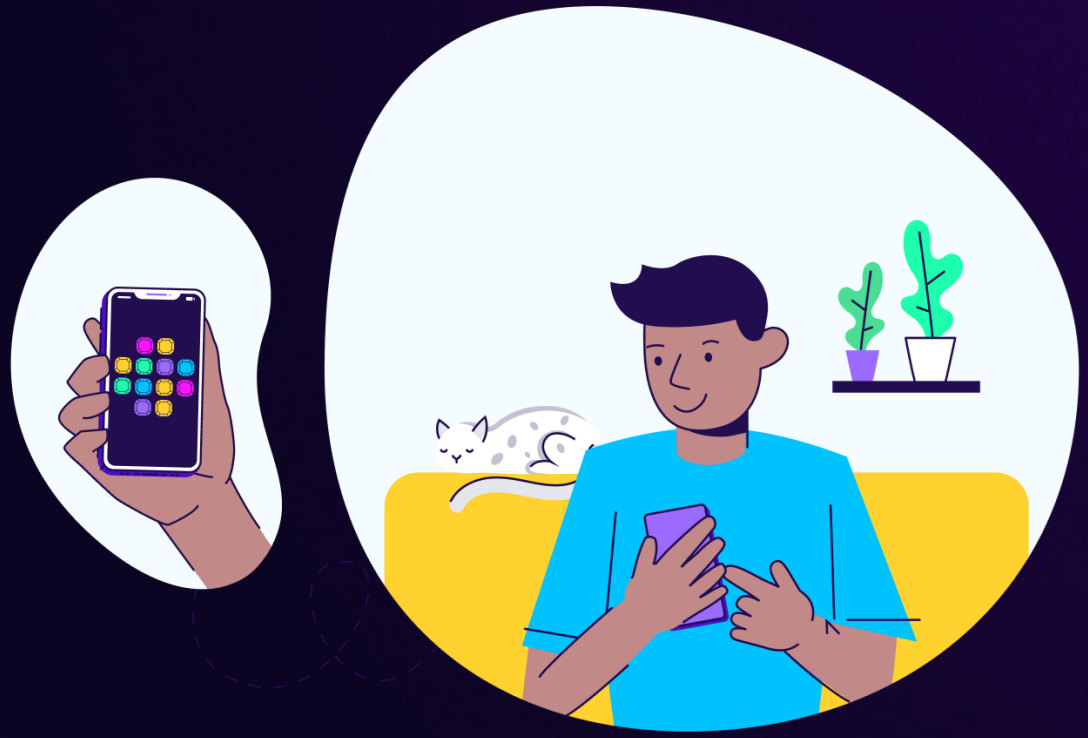


Who am I?

Victor Gershkovich

- Data Platform Group Leader at Appsflyer
- Over a decade on AWS infrastructure
- Shuffling data at scale

AppsFlyer



AppsFlyer

Analytics workload

Analytics

Analytics

- Batch

Analytics

- Batch
- Resource allocation

Analytics

- Batch
- Resource allocation
- Processing time

Analytics

- Batch
- Resource allocation
- Processing time
- Sensitive to interruptions

Challenges

~100 PB/d

Challenges

~100 PB/d

~1k jobs

Challenges

~100 PB/d

~1k jobs

Compute

Challenges

~100 PB/d

~1k jobs

Compute

Trends

Millions per second

Challenges

~100 PB/d

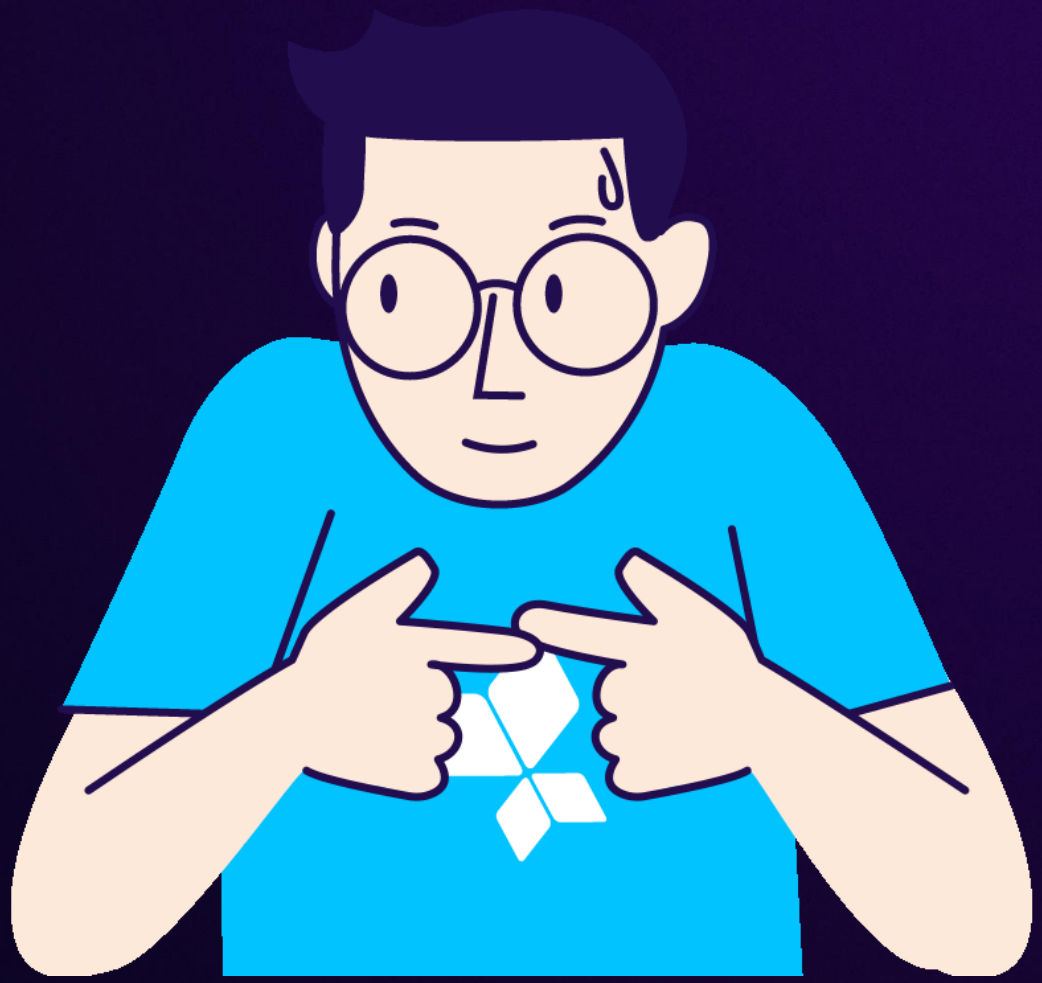
~1k jobs

Compute

Trends

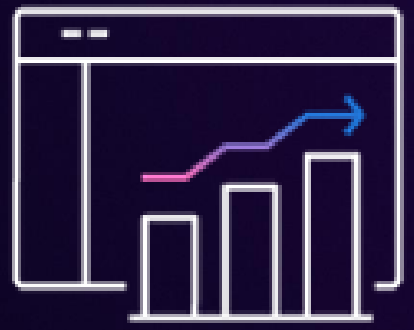
SLA

Why EKS?

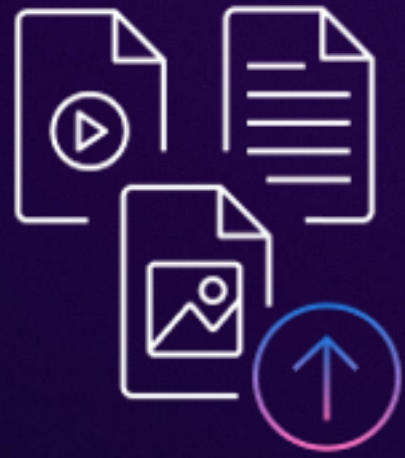


Why EKS?

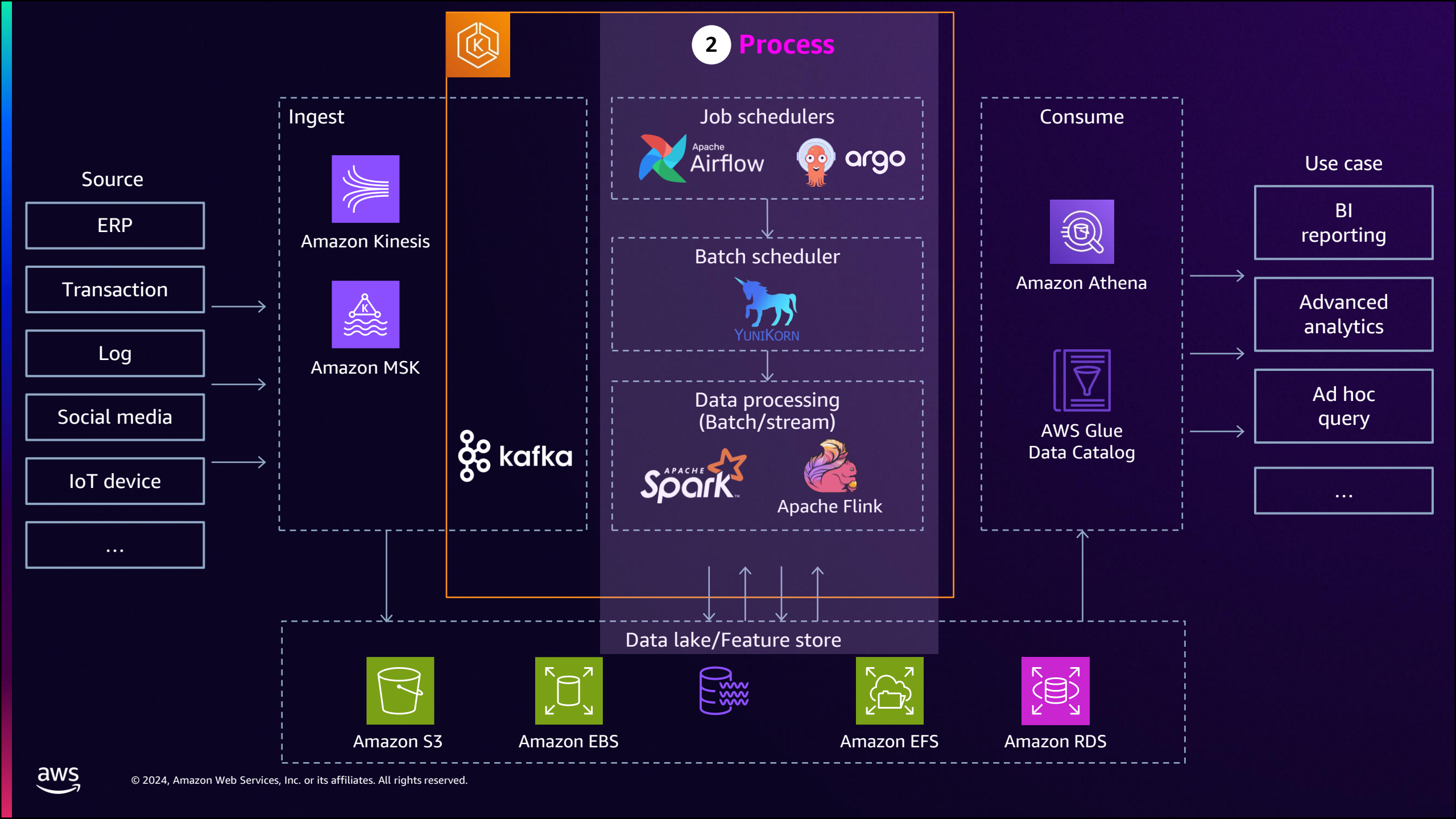
Karpenter

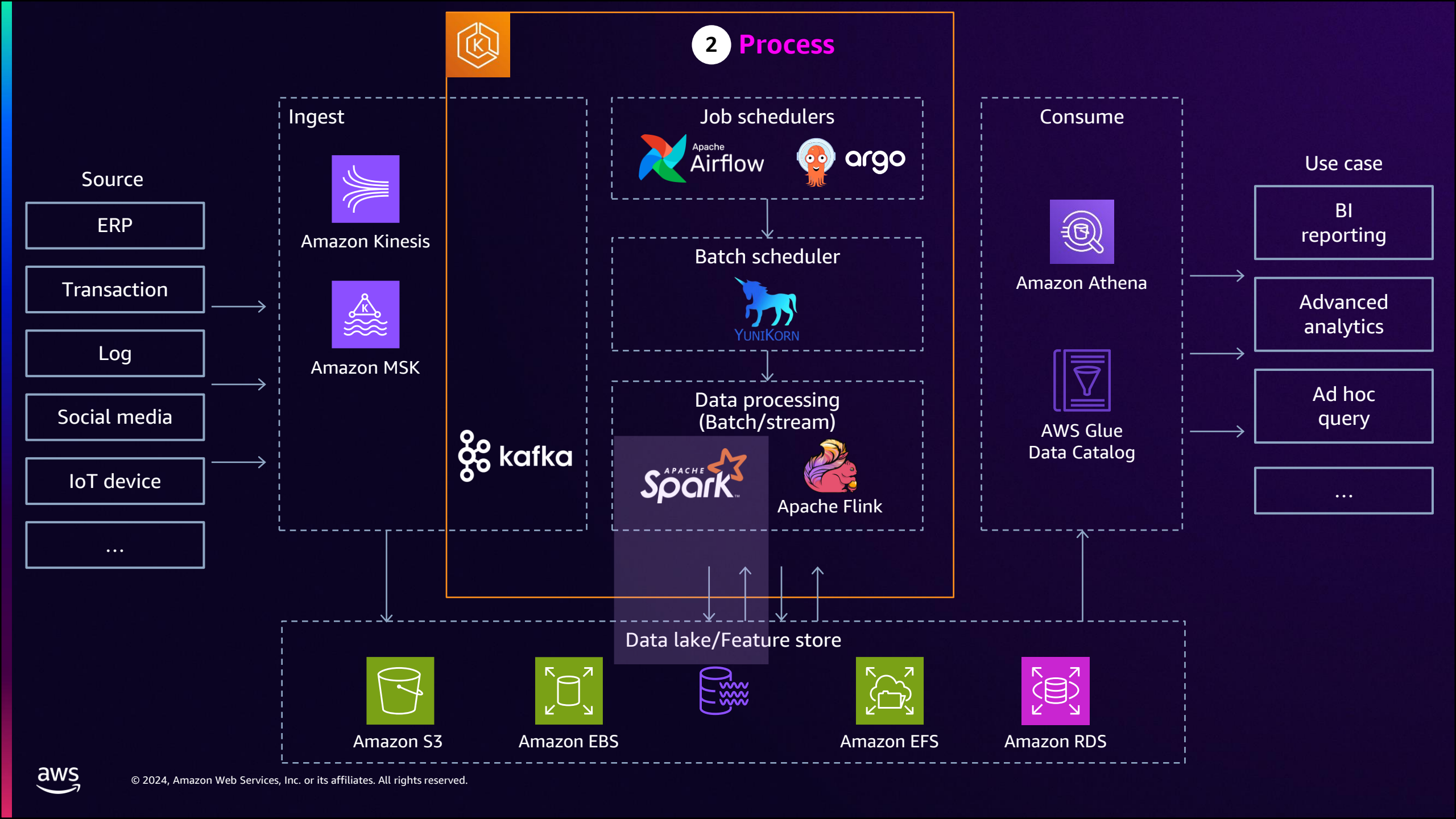


Observability

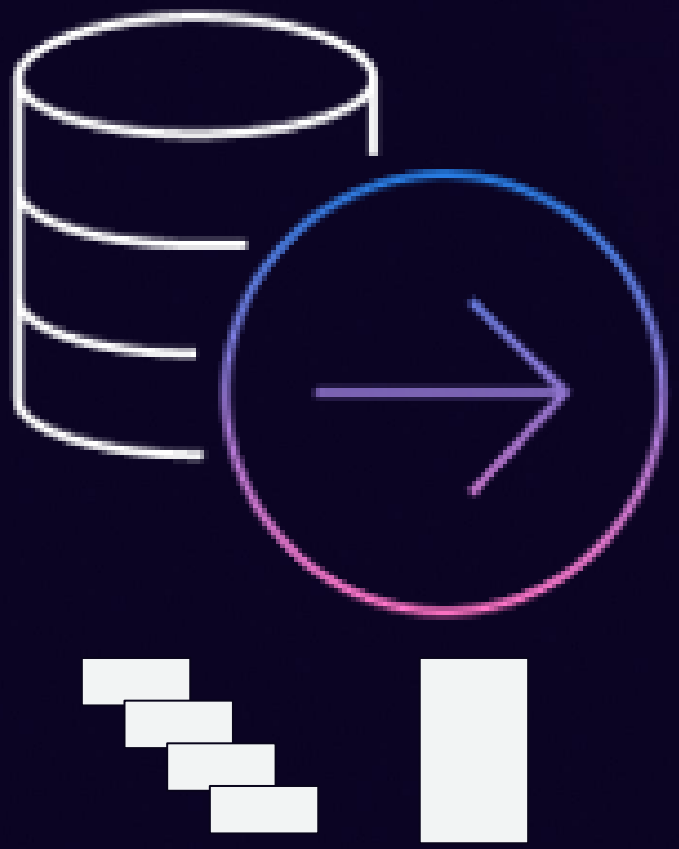


Enablement





Compaction job

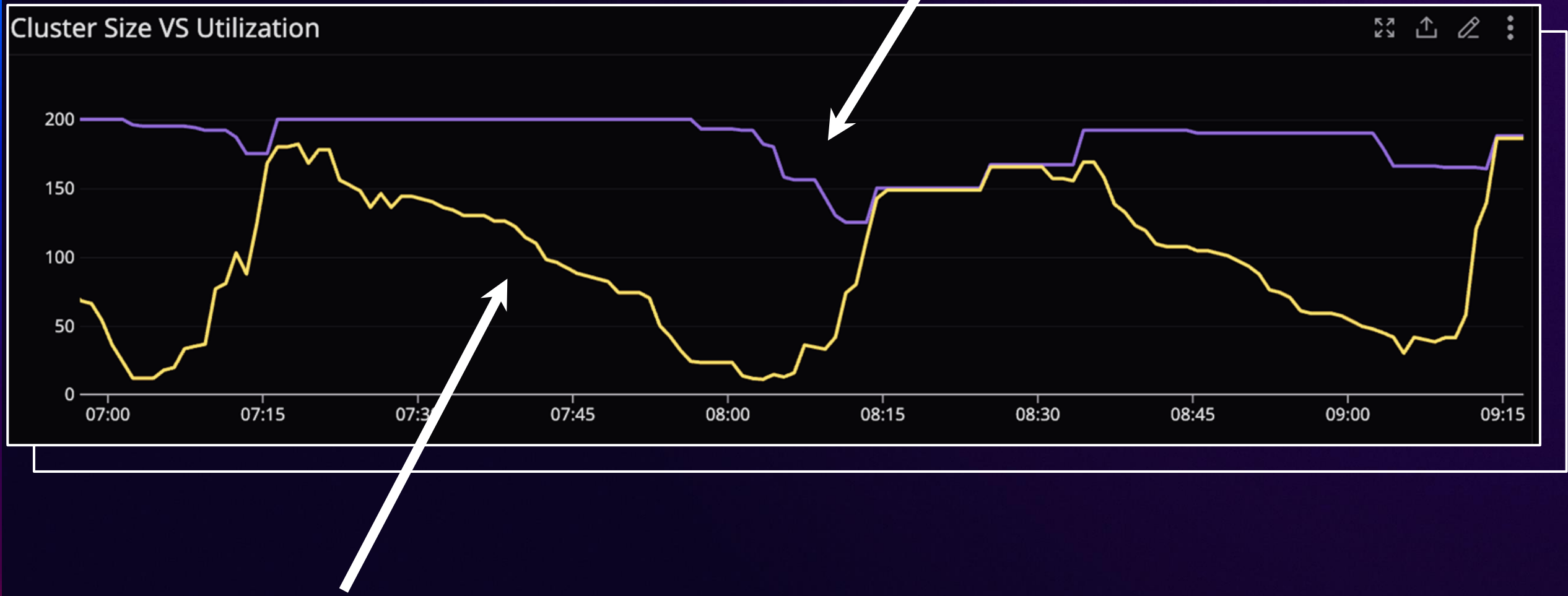


- Hourly batch
- 150 jobs
  - 1-3 (min)
  - 3-10 (min)
  - 10+ (mim)
- Process ~ 60T in 1 hour

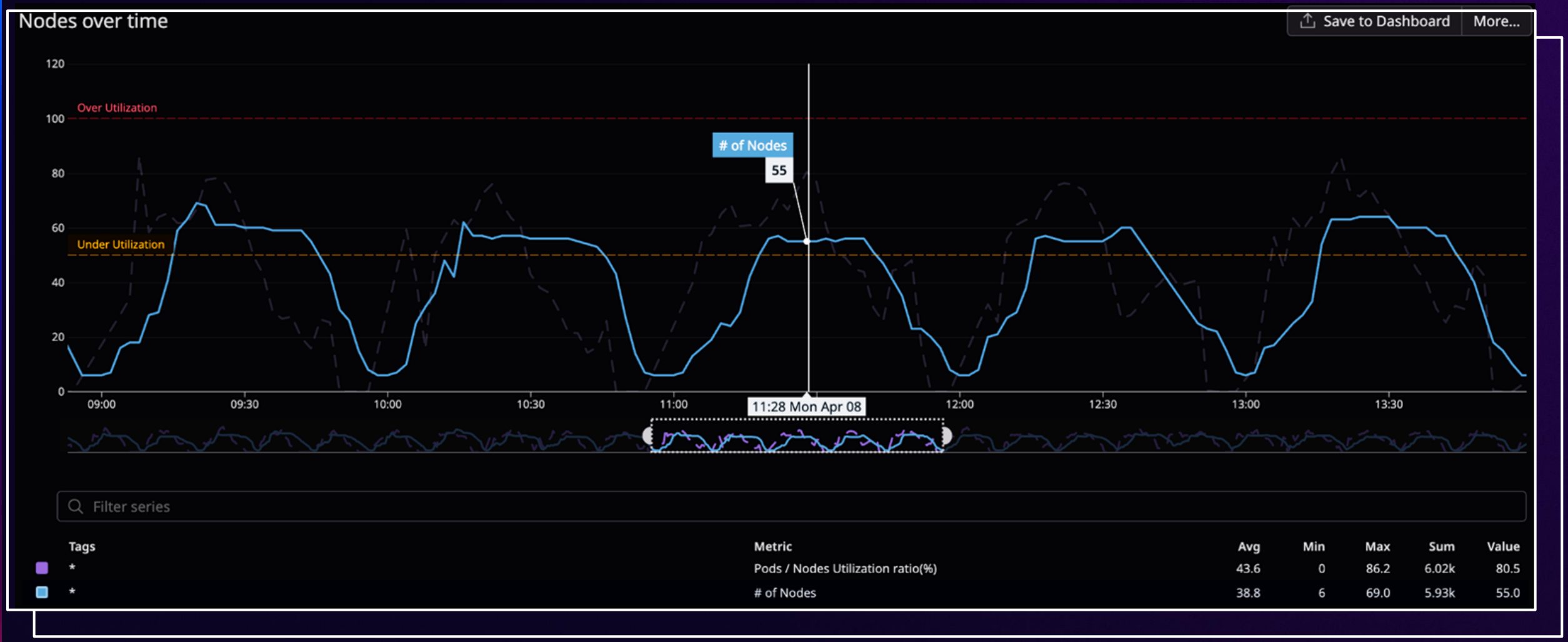
Karpenter

- Compute
- Scaling
- Cost performance

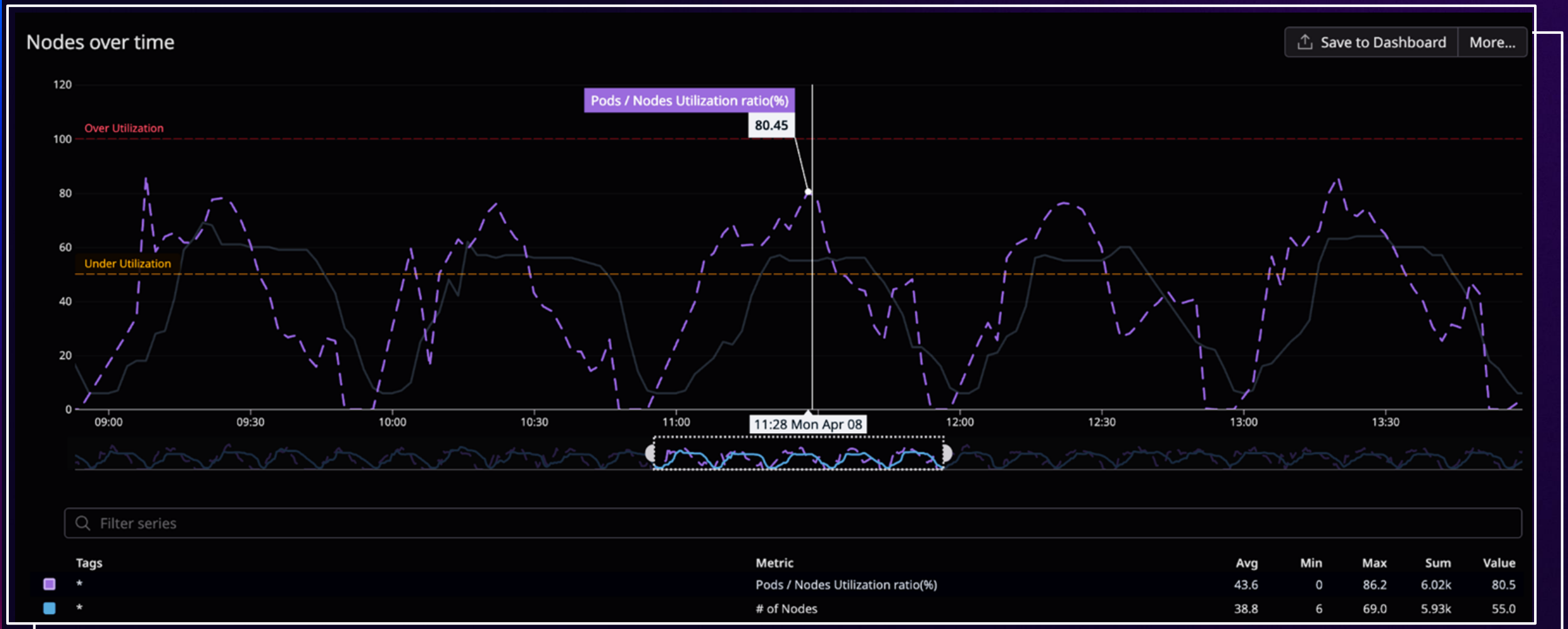
Before Karpenter



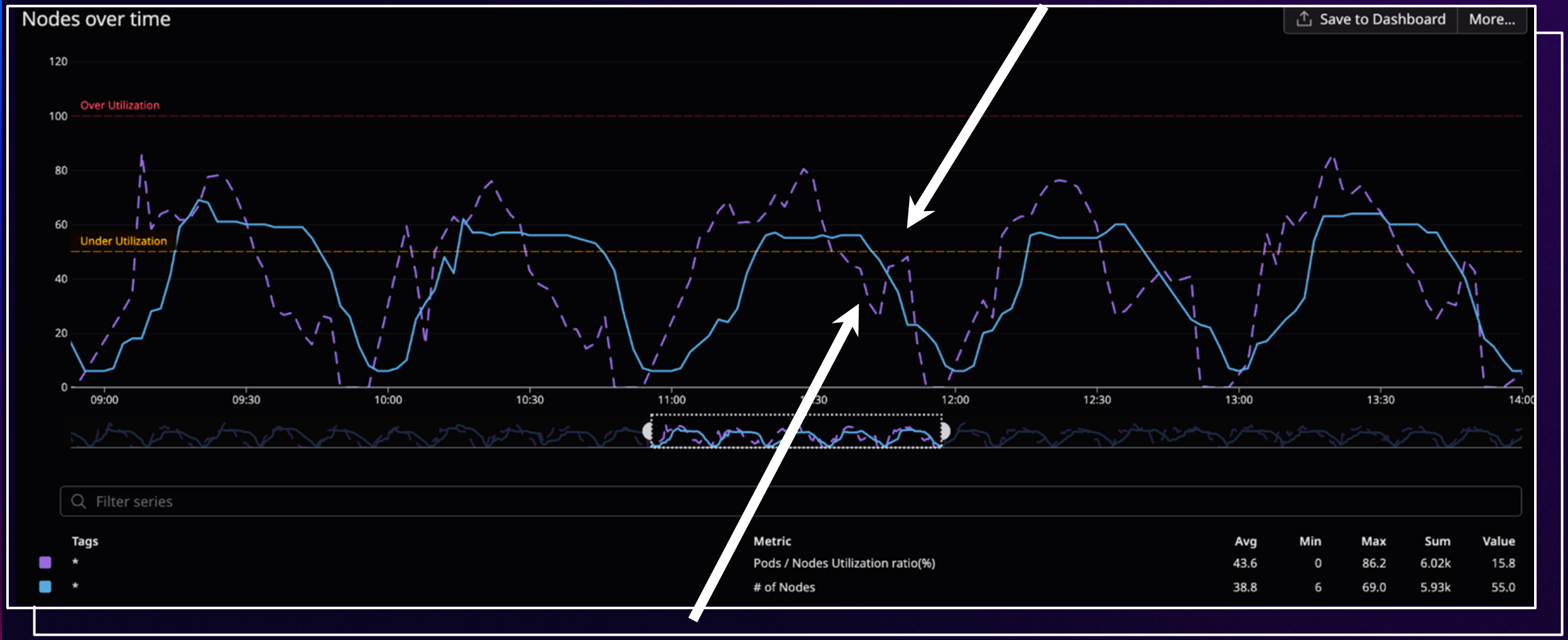
Karpenter - Nodes



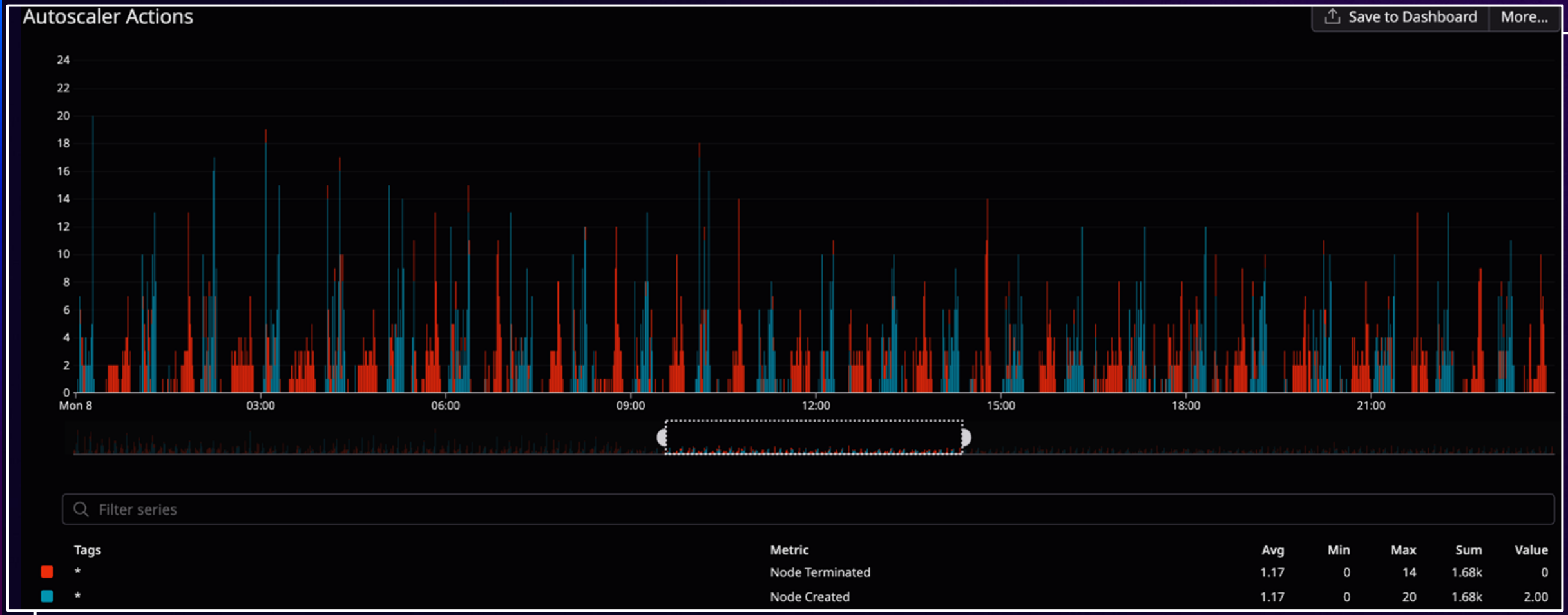
Karpenter - Utilization



Karpenter - Utilization

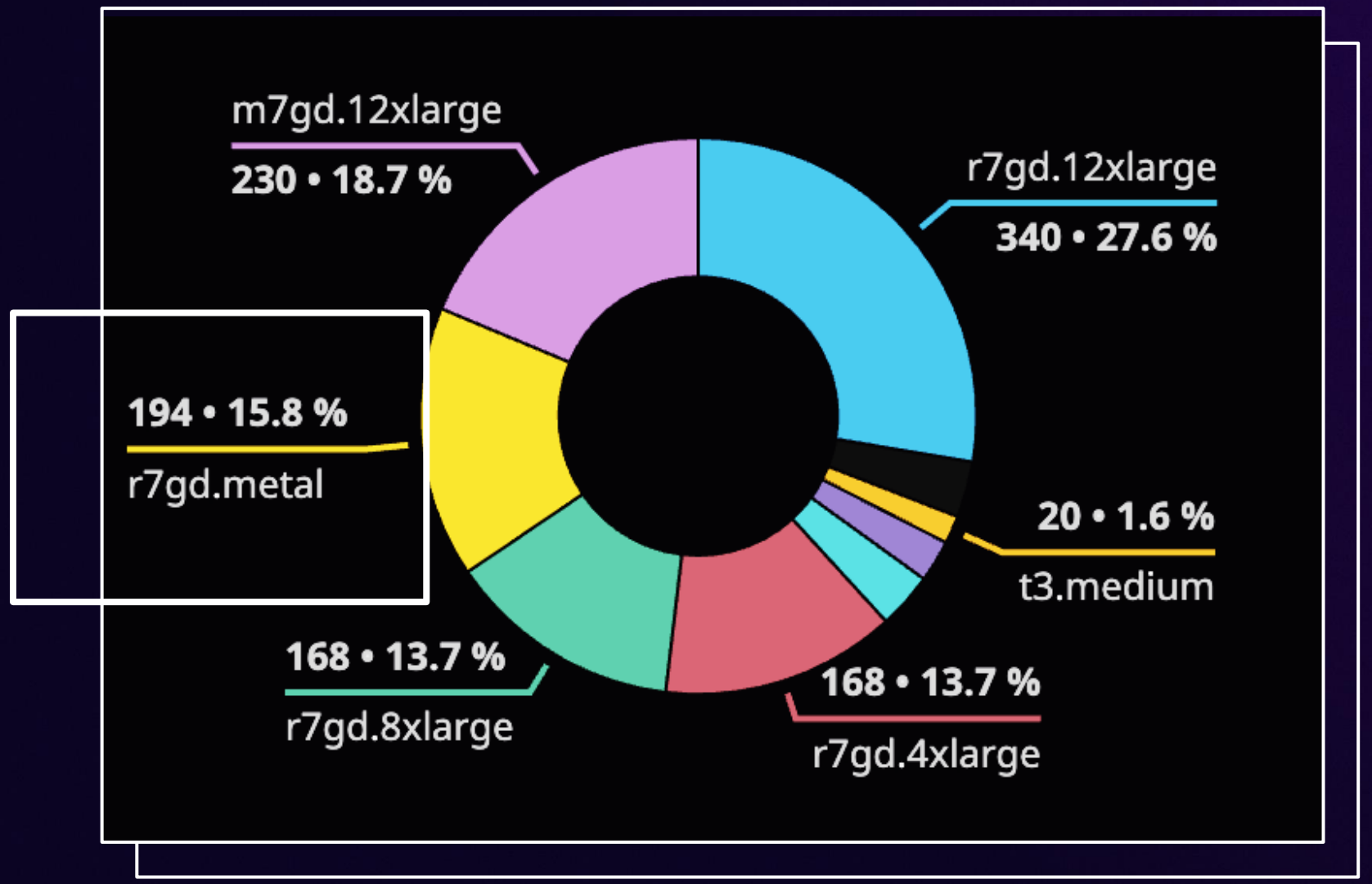


Karpenter - Compute



1.6k - Create\Terminate

Karpenter - Node selection



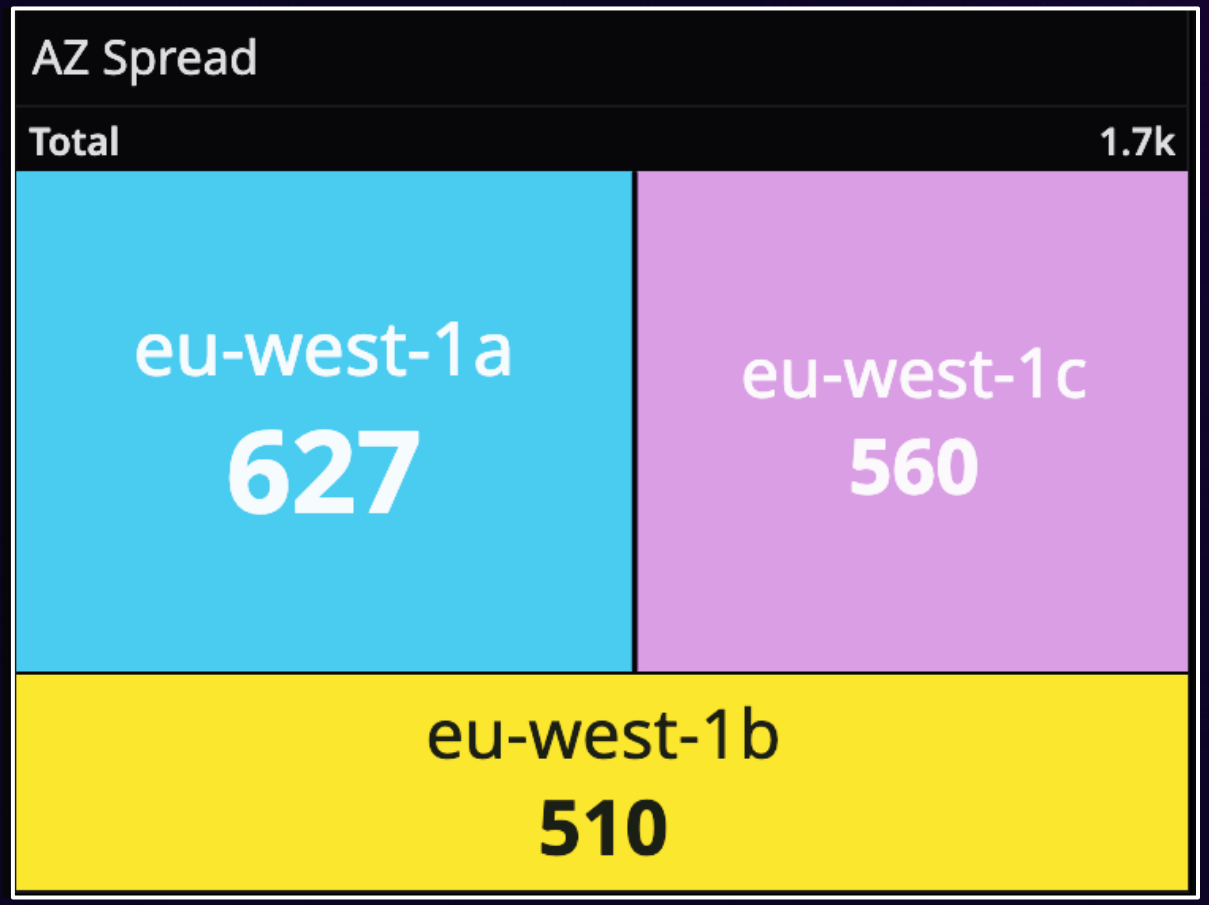
100%

Spot instances

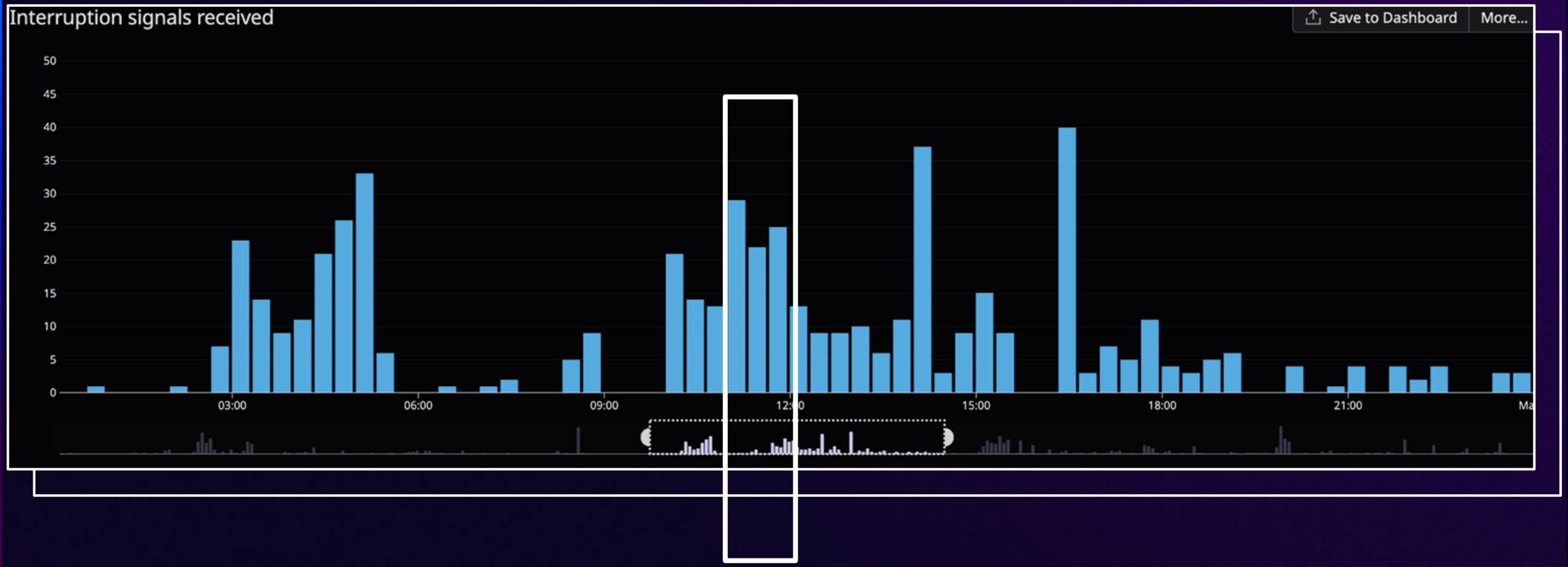
Minutes

New gen

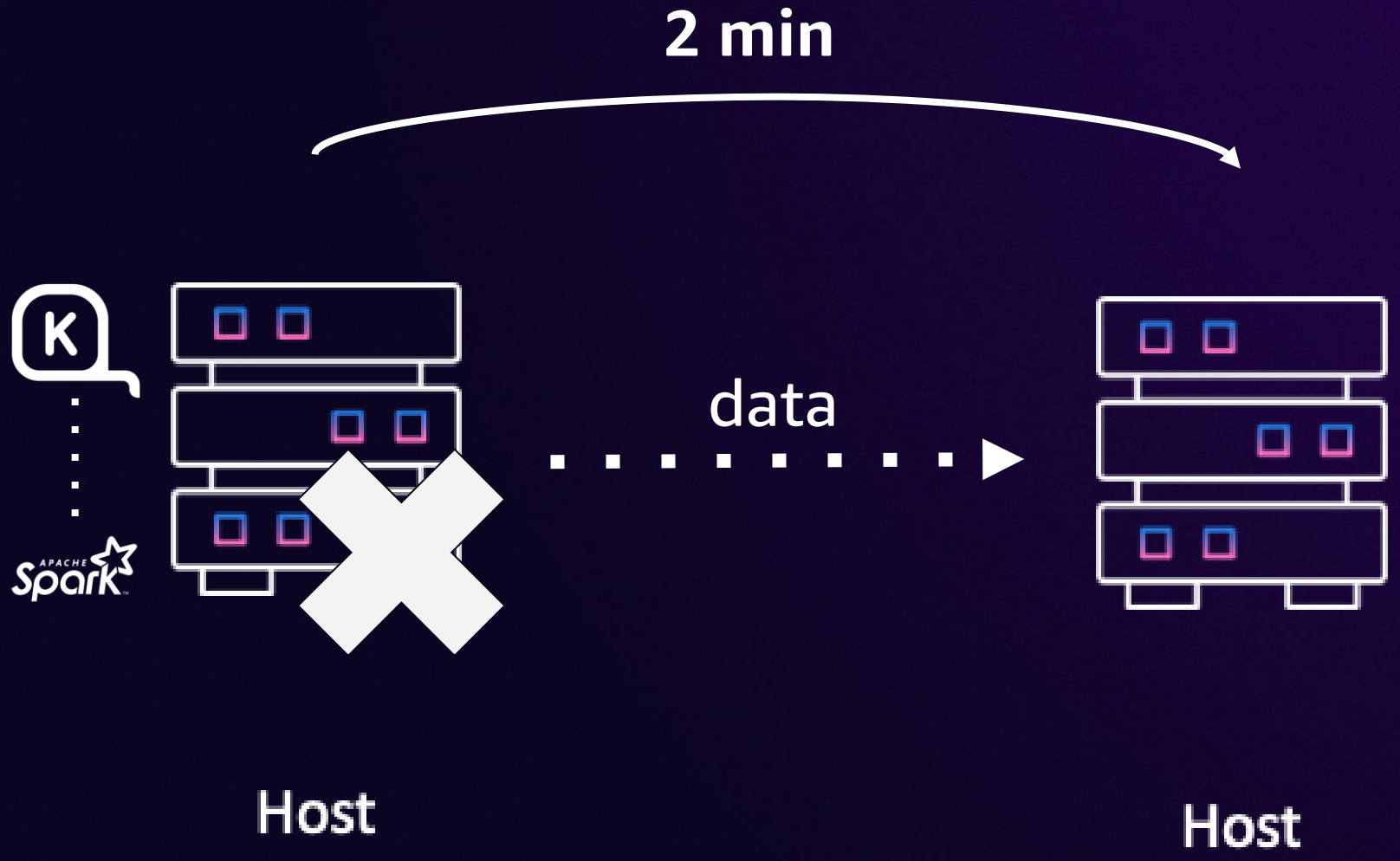
Karpenter - Availability Zone



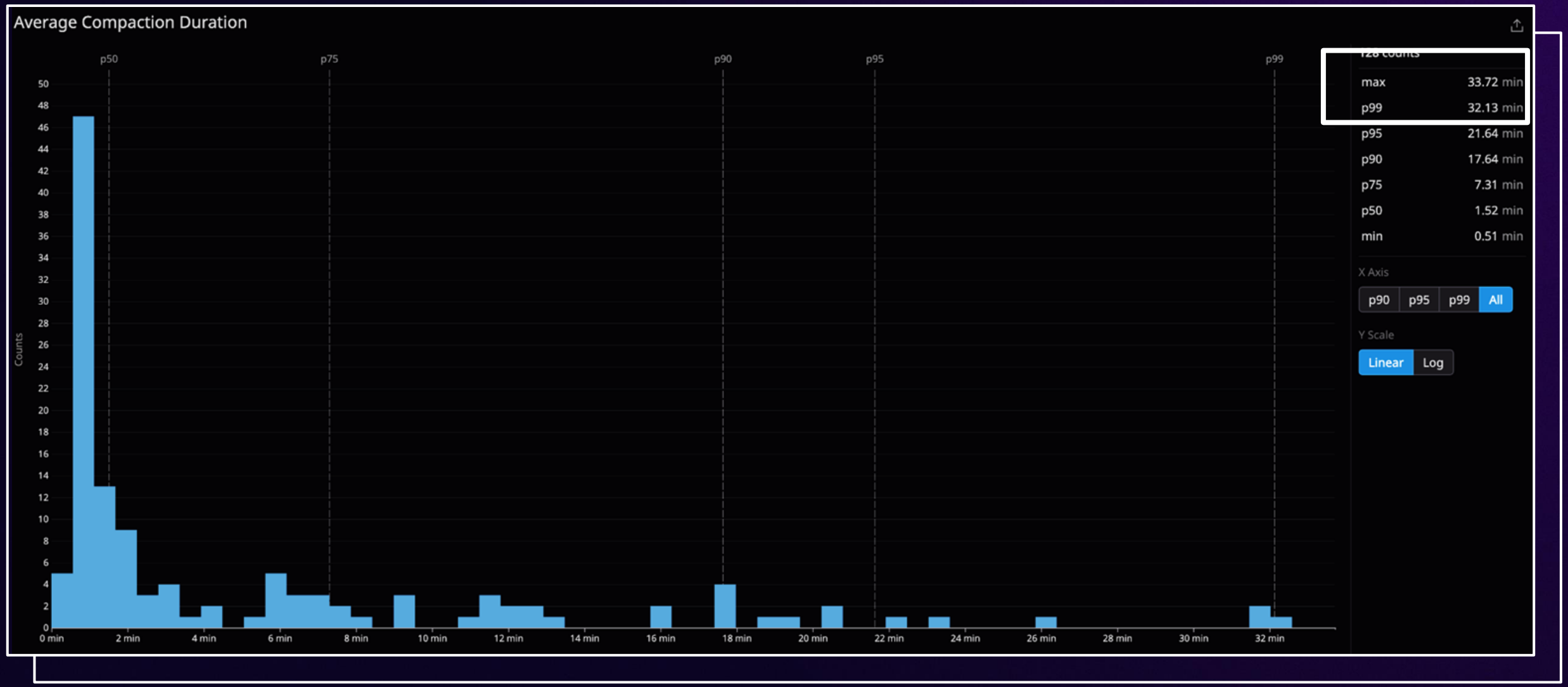
Karpenter - Spot interruption



Karpenter - Termination handling

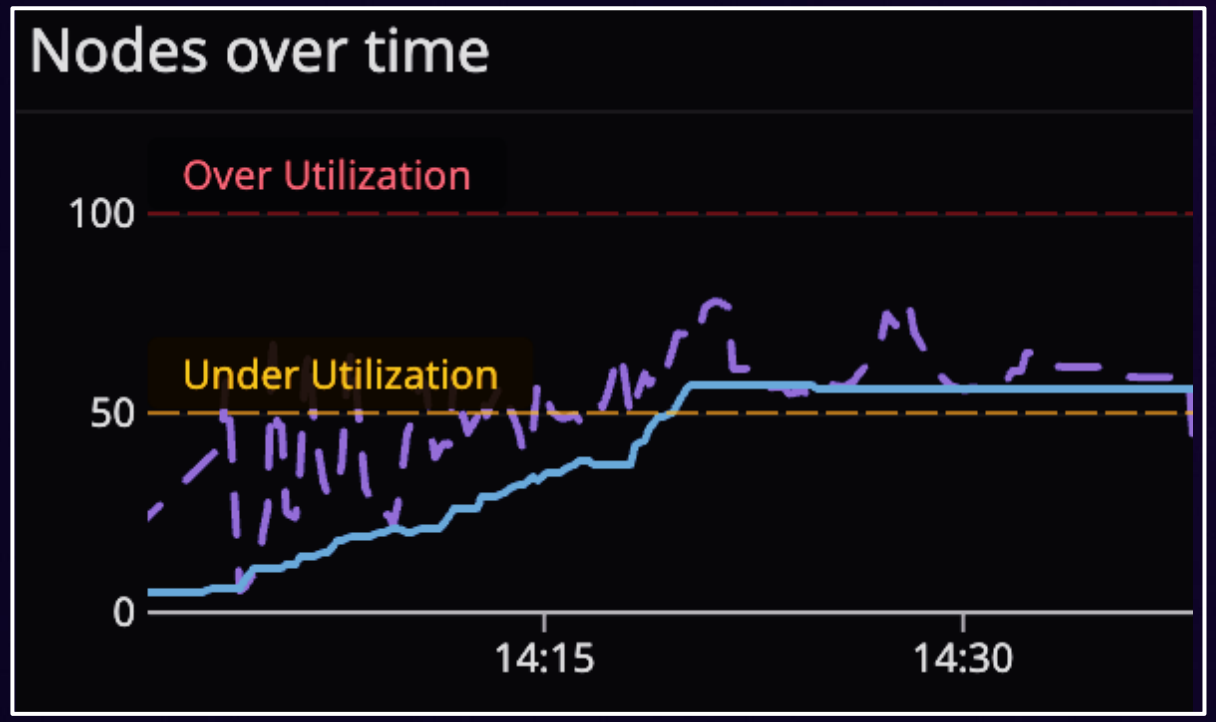


Karpenter - SLA



How?

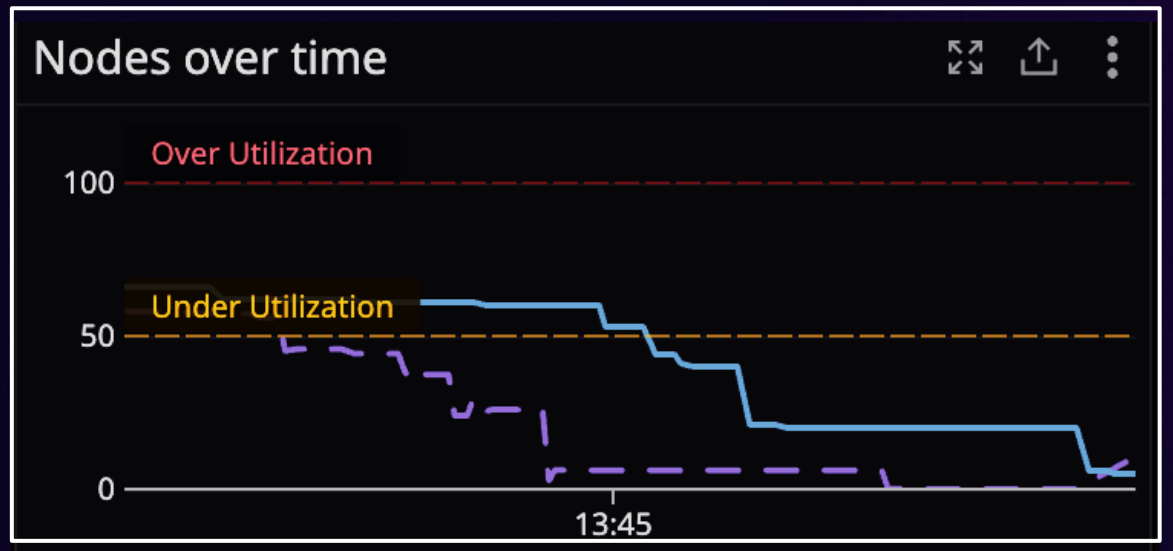
Scale up



Bootstrap - 10 sec

```
---
apiVersion: karpenter.k8s.aws/v1
kind: EC2NodeClass
metadata:
  name: default
spec:
  amiFamily: AL2023
```

Scale down



Optimized for decommission

```yaml
spec:
  disruption:
    budgets:
    - duration: 20m
      nodes: "50"
      schedule: 45 0,1,2,4,5,6,7,8,9,10,18,19,20,21,22,23 * * *
    - duration: 20m
      nodes: 2%
      schedule: 30 13-17 * * *
    - duration: 15m
      nodes: 2%
      schedule: 30 * * * *
    - nodes: 20%
    consolidationPolicy: WhenUnderutilized
    expireAfter: Never
```

Resilience for termination

```
kind: SparkApplicationClusterPolicy
metadata:
  name: default
spark.decommission.enabled: "true"
spark.storage.decommission.rddBlocks.enabled: "true"
spark.storage.decommission.shuffleBlocks.enabled: "true"
spark.storage.decommission.enabled: "true"
spark.executor.decommission.killInterval: "120"
spark.files.fetchFailure.unRegisterOutputOnHost: "true"
spark.storage.decommission.replicationReattemptInterval: "100ms"
spark.stage.ignoreDecommissionFetchFailure: "true"
spark.storage.decommission.fallbackStorage.path: "s3://${logs_bucket_name}/decommission/"
spark.shuffle.mapOutput.minSizeForBroadcast: "20480k"
spark.kubernetes.executor.volumes.hostPath.spark-local-dir-1.mount.path: "${ssd_data_path}"
spark.kubernetes.executor.volumes.hostPath.spark-local-dir-1.options.path: "${executor_options_path}"
```

Graceful decommission tuning

Local storage shuffle optimization

Resilience for termination

```yaml
executor:
  template:
    metadata:
      spec:
        affinity:
          podAffinity:
            requiredDuringSchedulingIgnoredDuringExecution:
              - topologyKey: topology.kubernetes.io/zone
                labelSelector:
                  matchLabels:
                    spark-app-name: <my_spark_app>
          nodeAffinity:
            requiredDuringSchedulingIgnoredDuringExecution:
              nodeSelectorTerms:
                - matchExpressions:
                  - key: karpenter.k8s.aws/instance-local-nvme
                    operator: Gt
                    values:
                      - 300
                  - key: karpenter.sh/capacity-type
                    operator: In
                    values:
                      - spot
                  - key: kubernetes.io/arch
                    operator: In
                    values:
                      - arm64
```

Job pods on the same AZ

Resilience for termination

```yaml
executor:
  template:
    metadata:
      spec:
        affinity:
          podAffinity:
            requiredDuringSchedulingIgnoredDuringExecution:
              - topologyKey: topology.kubernetes.io/zone
                labelSelector:
                  matchLabels:
                    spark-app-name: <my_spark_app>
          nodeAffinity:
            requiredDuringSchedulingIgnoredDuringExecution:
              nodeSelectorTerms:
                - matchExpressions:
                    - key: karpenter.k8s.aws/instance-local-nvme
                      operator: Gt
                      values:
                        - 300
                    - key: karpenter.sh/capacity-type
                      operator: In
                      values:
                        - spot
                    - key: kubernetes.io/arch
                      operator: In
                      values:
                        - arm64
```

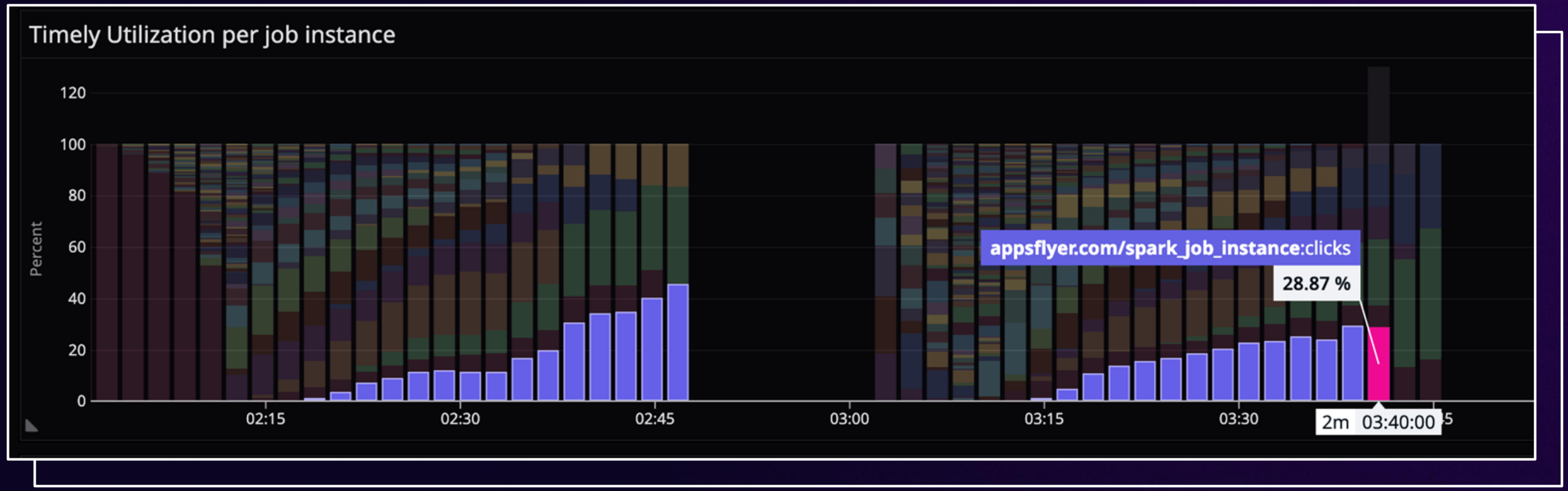
Job pods on the same AZ

Local storage and Arm

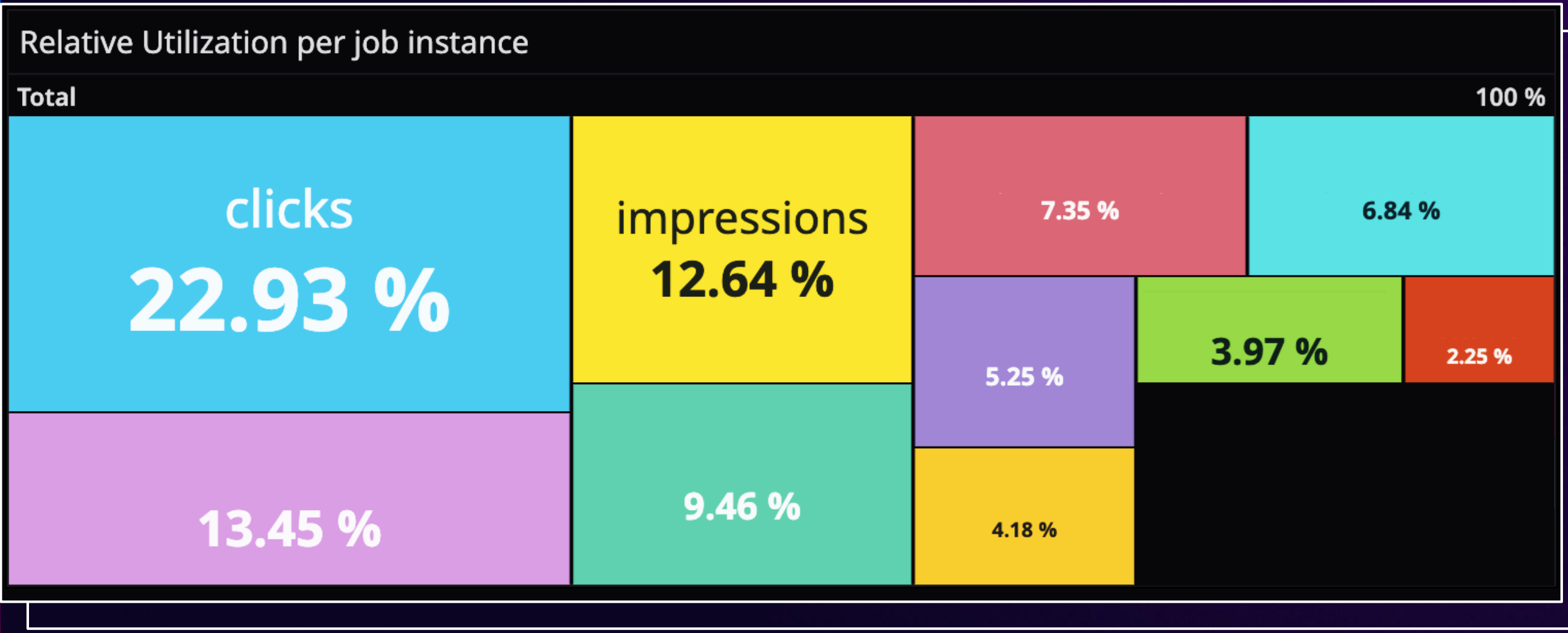
Observability

Observability

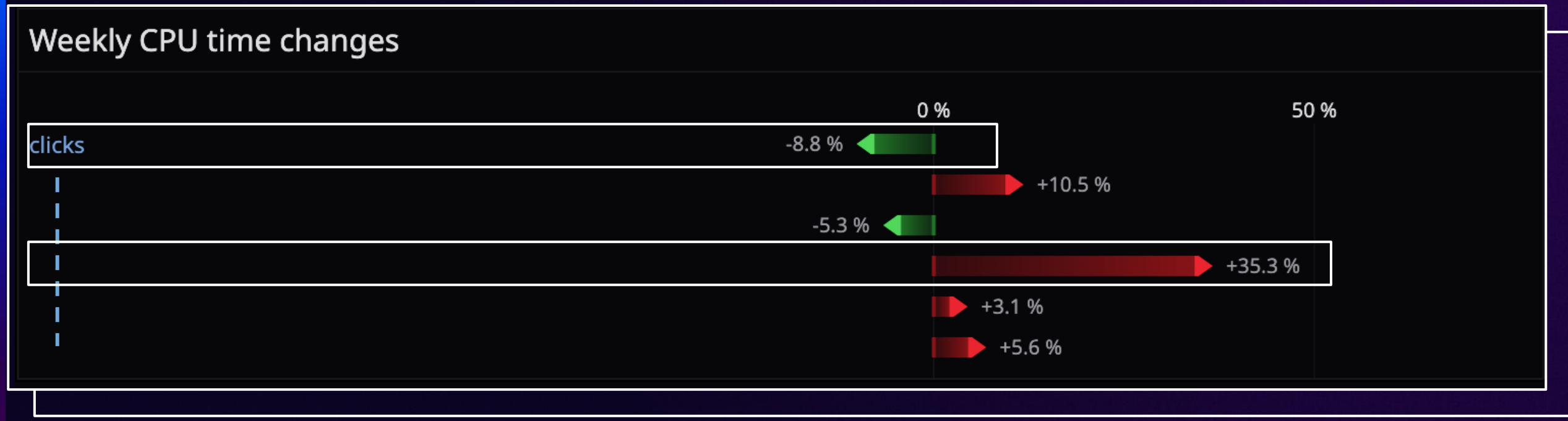
Observability



Observability



Observability



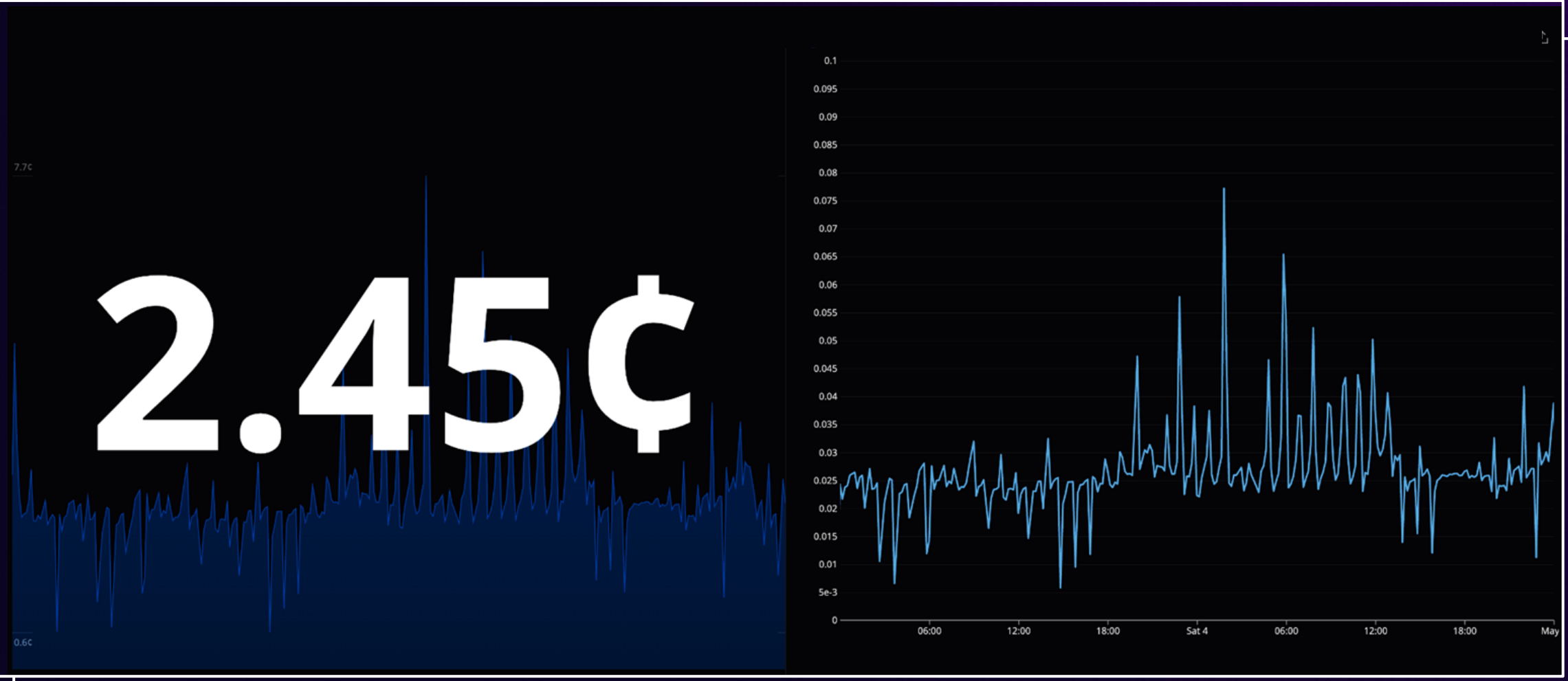
How?

```
"type": "timeseries",
"requests": [
    {
        "formulas": [
            {
                "formula": "query1 / query2 * 100",
                "number_format": {
                    "unit": {
                        "type": "canonical_unit",
                        "unit_name": "percent"
                    }
                }
            }
        ],
        "queries": [
            {
                "data_source": "metrics",
                "name": "query1",
                "query": "sum:k8s.pod.cpu.time{$cluster_name, spark_job:*, kube_namespace:$application_type.value, $account} by {spark_job_instance}"
            },
            {
                "data_source": "metrics",
                "name": "query2",
                "query": "sum:k8s.pod.cpu.time{$cluster_name, spark_job:*, kube_namespace:$application_type.value, $account}"
            }
        ],
```

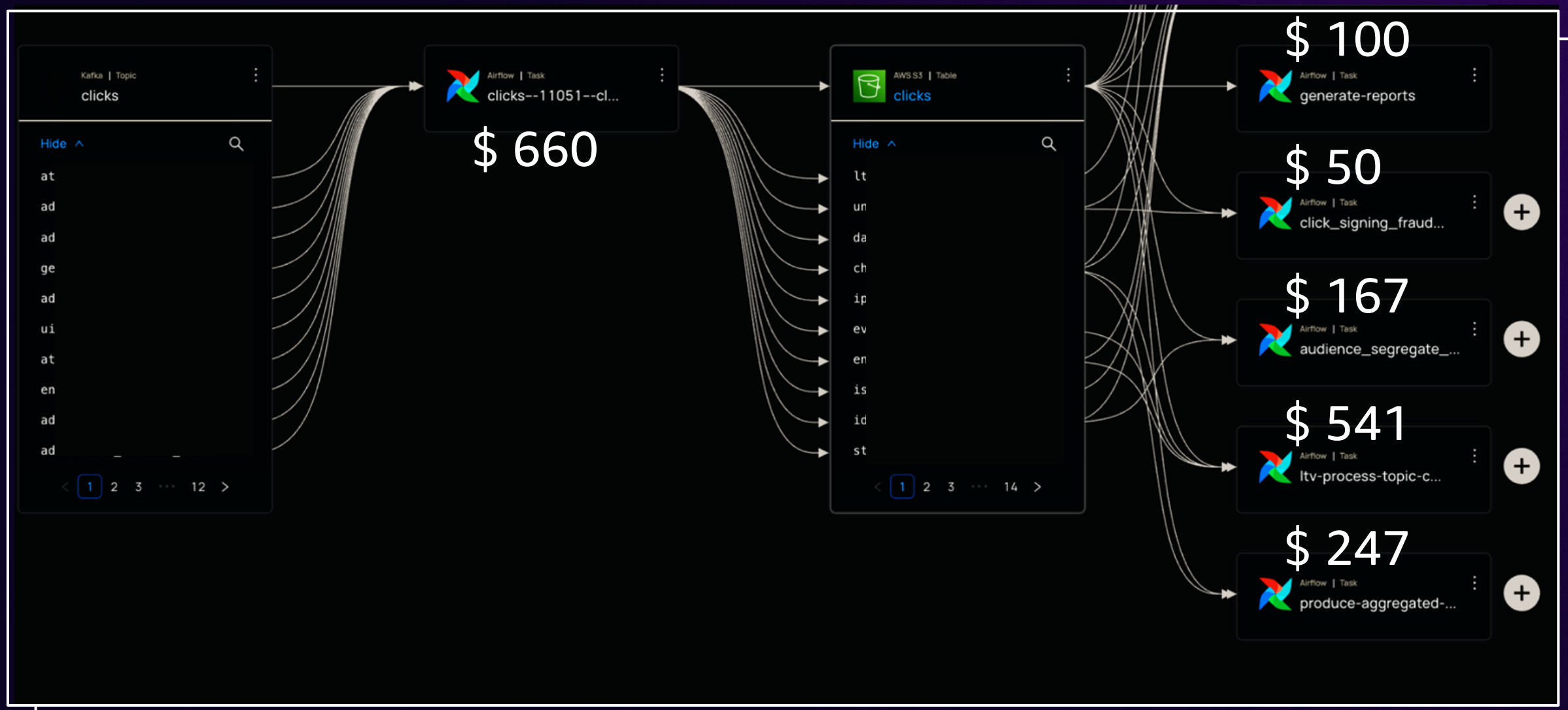
How?

```
"queries": [
    {
        "data_source": "metrics",
        "name": "query1",
        "query": "sum:otel.system.filesystem.utilization{$cluster_name,$account,mountpoint:/}.rollup(count)",
        "aggregator": "avg"
    },
    {
        "data_source": "metrics",
        "name": "query2",
        "query": "avg:karpenter.karpenter_cloudprovider_instance_type_price_estimate{zone:eu-west-1b* AND capacity_type:spot AND $cluster_name AND $account AND instance_type}",
        "aggregator": "avg"
    },
    {
        "data_source": "metrics",
        "name": "query3",
        "query": "avg:karpenter.karpenter_nodepool_usage{nodepool:unmonitored, $cluster_name, $account, resource_type:cpu}",
        "aggregator": "avg"
    }
```

Observability - Cost per minute



Observability - Lineage



Enablement

Enablement

Data platform

Data engineer

Reconcile loops

Enablement

Actions

Defines repository **actions**

Deployments

Describe a deployment of a service, cloud resource, configuration, and more

Environments

Define values that are shared across units (environment, Region, Availability Zone)

Enablement

Infrastructure

```
module "spark" {
  source = "git::https://git.appsflyer.com/modules/spark.git?ref=0.19.16"
  aws_account       = var.aws_account
  aws_region        = var.aws_region
  system            = "compaction"
  role              = "datalake"
  create_aws_auth_configmap = false
  create            = false
  airflow_role_arn = "arn:aws:iam::name}"
```

Click to collapse the range.

Karpenter

```
nodeSelectorTerms:
  - matchExpressions:
      - key: karpenter.k8s.aws/instance-local-nvme
        operator: Gt
        values:
          - 300
      - key: karpenter.sh/capacity-type
        operator: In
        values:
          - spot
      - key: kubernetes.io/arch
        operator: In
        values:
          - arm64
```

Spark

```
type: Scala
image: eu-west-1.amazonaws.com/af-spark:3.5.1-scala2.12-java17-ubuntu
mainClass: com.appsflyer.datainfra.datalake.compaction.DatalakeCompaction
mainApplicationFile: https://packages.io/compaction_2.12-[RELEASE].jar
sparkVersion: "3.5.1"
restartPolicy:
  onSubmissionFailure: true
driver:
  template:
    metadata:
      labels:
        version: 3.5.1
```

Enablement

Value

Value

Value


Optimize and monitor EKS for
analytics with best practices


Align tools and practices to foster organizational growth


Provide APIs to empower data engineers to work independently

Session resources

awslabs.github.io/data-on-eks

Thank you!

Please complete the session survey in the mobile app

Victor Gershkovich
victor@appsflyer.com
victor-gershkovich

Christina Andonov
candonov@amazon.com
christina-andonov

Roland Barcia
rolanbah@amazon.com
roland-barcia-aws