Grafana Labs

Kubernetes Everywhere

Lessons Learned From Going Multi-Cloud

20 May 2022

Niko Smeds

Cloud Platform at Grafana

Grafana Cloud

Kubernetes Everywhere

Agenda

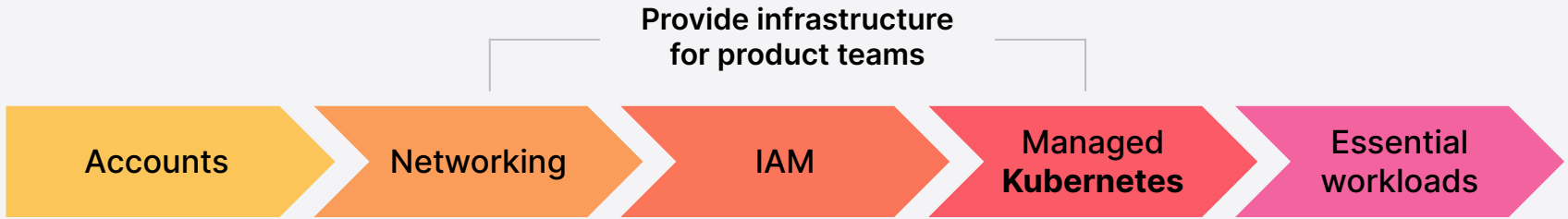
1. Why go multi-cloud?
2. Project overview
3. Five lessons
4. What went well
5. Q&A

Why?

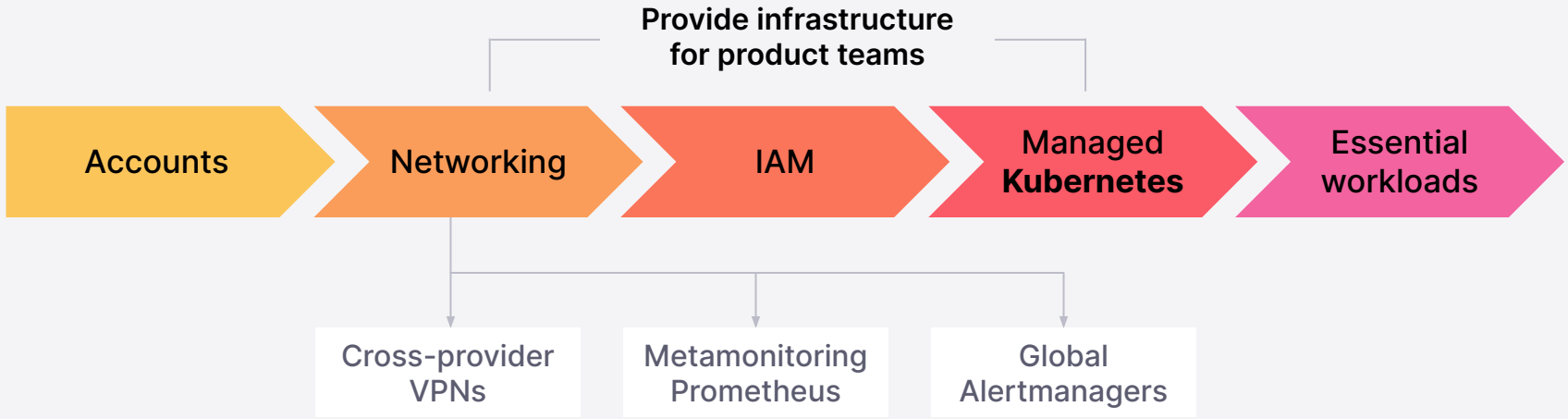
Why might an organization consider opting for multi-cloud?

- Increase **available regions**
- Reduce **vendor lock-in**
- Customer **preference**
  - Latency
  - Data sovereignty
  - Spend commit

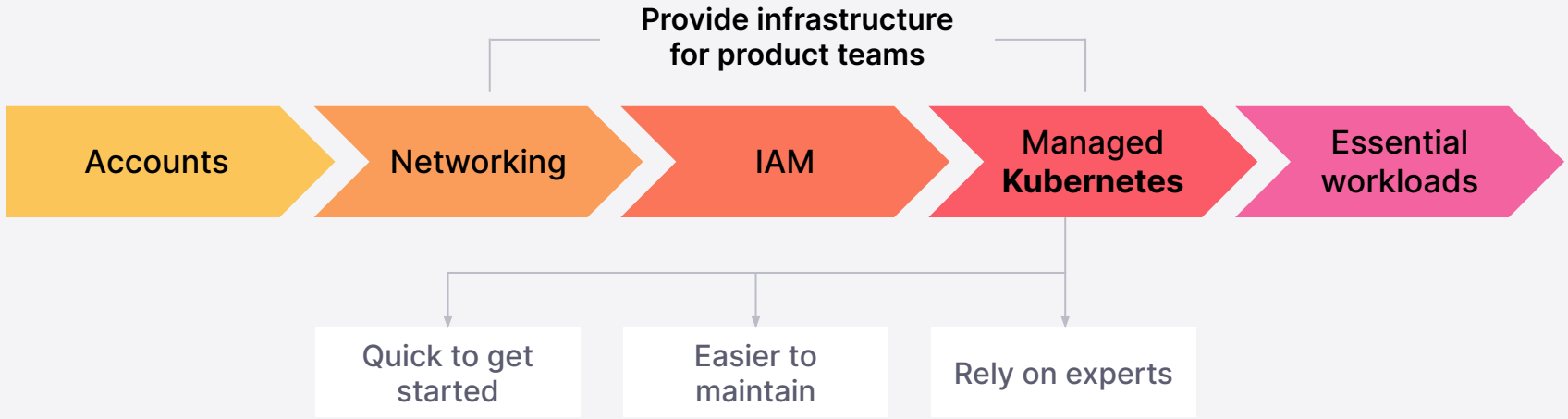
Project: **Grafana Cloud Platform on AWS**



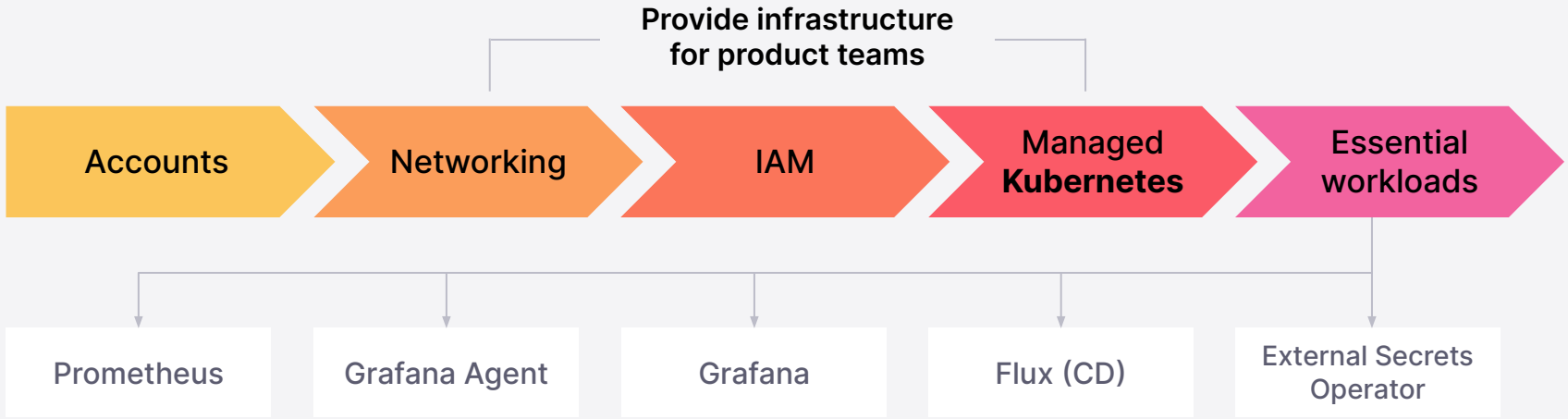
Project: **Grafana Cloud Platform on AWS**



Project: **Grafana Cloud Platform on AWS**



Project: **Grafana Cloud Platform on AWS**



Getting Started

In the beginning...

- Set **clear requirements**
  - What workloads to include/exclude?
  - Do we avoid inter-provider dependencies?
  - Expected scale?
- **Research** Amazon Elastic Kubernetes (EKS)
  - Documentation
  - Conference videos
  - Blog posts
  -

READY,

SET,

... not quite

Virtual Private Cloud (VPC)

GCP

Global VPCs

Supports subnets from multiple regions.

We have use-cases for this and run multiple K8s clusters across multiple regions in shared VPCs.

VS

AWS

Regional VPCs

Supports subnets from a single region.

We're left to decide:

- VPC per region, or
- VPC per cluster

VPC CIDRs

GCP

Mix private ranges

Alias IP ranges supported from a mix of RFC 1918 ranges.

- **10.0.0.0/8** subnets for primary components (nodes, pods, services)
- **172.16.0.0/12** subnets for managed components (GKE control plane)

vs

AWS

Single private range

Secondary CIDR block restricted to a single RFC 1918 range.

- **10.0.0.0/8** subnets for primary components (nodes, pods, services)
- **10.0.0.0/8** subnets for managed components (EKS control plane)

RFC 1918 ranges
10.0.0.0/8
172.16.0.0/12
192.168.0.0/16

Subnets

GCP

Supports /29 → /8

Pods are deployed in **/14** subnets.

vs

AWS

Supports /28 → /16

Pods are deployed in **/16** subnets.

Required refactoring our **IP reservation plans**.
We avoid private range overlap to support
inter-cluster peering.

Comparing other services

Load Balancers

Mix of global and non-global load balancers.

Comparing other services

Load Balancers

Mix of global and non-global load balancers.

Volumes

Different tiers, performance, and pricing.

Comparing other services

Load Balancers

Mix of global and non-global load balancers.

Volumes

Different tiers, performance, and pricing.

Object Storage

Variance in read/write rate-limits.

1

Provider services are similar, but not the same

Tutorial hell

~~Tutorial hell~~

Documentation hell

I hear and I forget.
I see and I remember.
I do and I understand.

Xunzi
3rd Century BCE

Getting Started (again…)

Start building with the knowledge you've gathered.

- Be ready to **tear it down** and **start over**
- Use **infrastructure as code** and **version control** from the get-go
  - Peer review and suggestions
  - Share progress
  - Documents history
- You **cannot plan** for the **unknown**
  - You'll face limitations and bugs you didn't expect, and that's okay

2

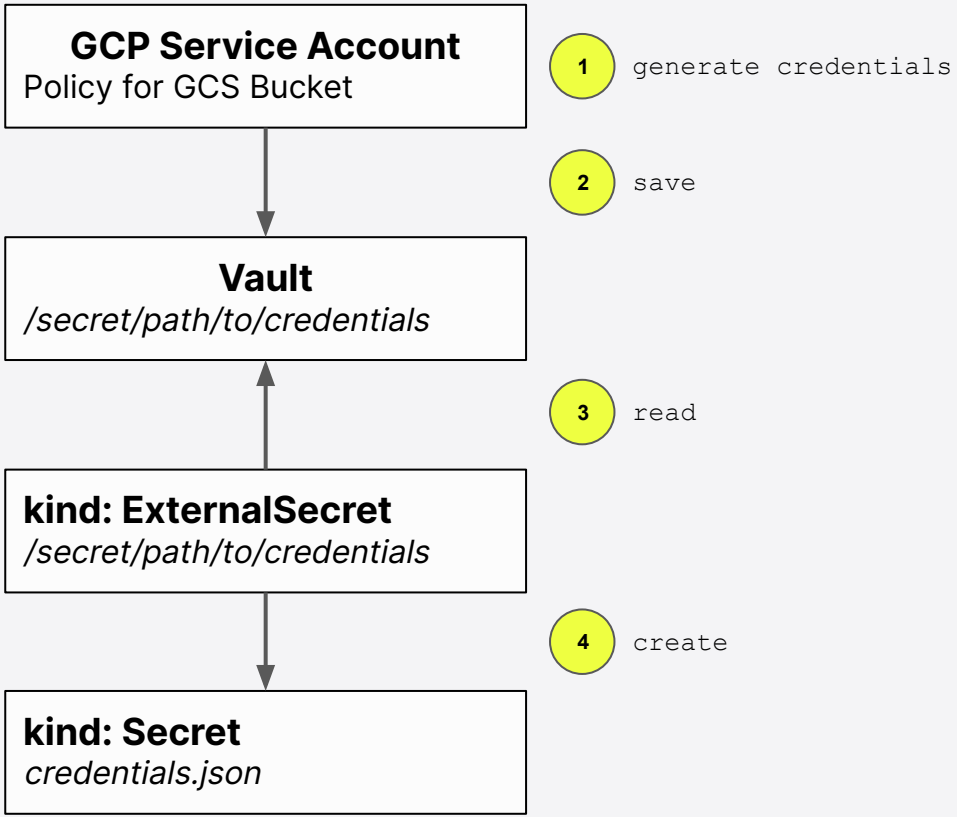
Just get started

Expanding to a new cloud provider?

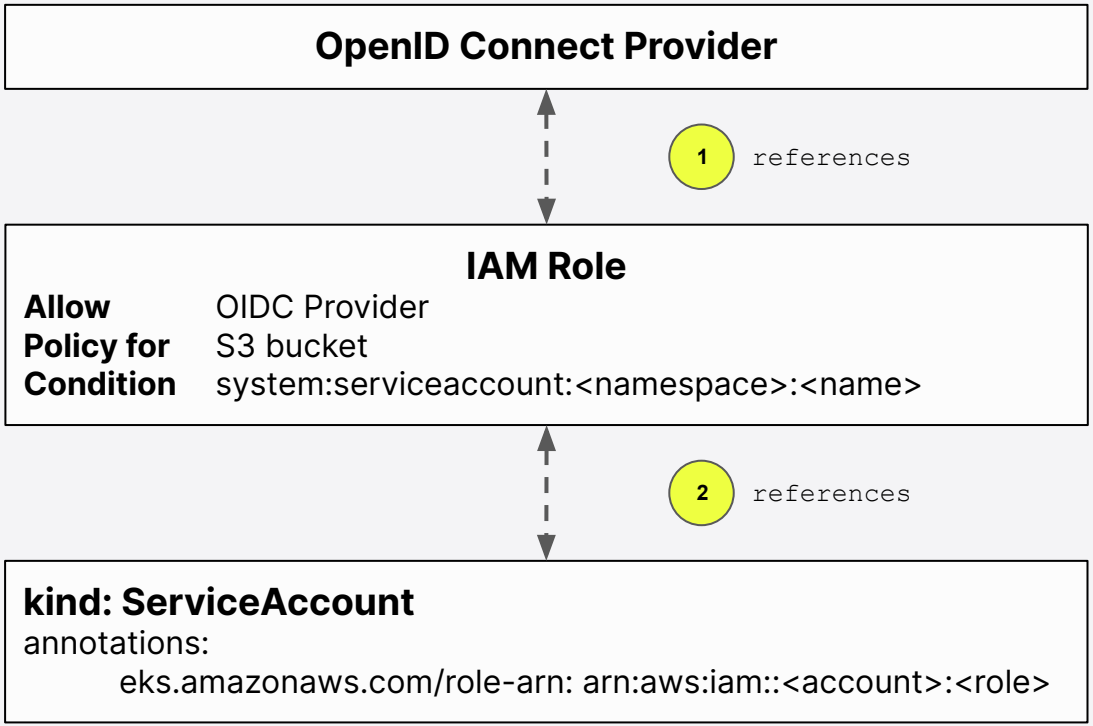
ctrl + c

ctrl + v

Example: **Application credentials (GCP)**



Example: **Application credentials (AWS)**



3

Don’t blindly do what was done before

It’s about time

AWS **EKS cluster** is up and running.

Add-ons are installed:

- **Cluster Autoscaler**
- **Load Balancer Controller**
- **AWS VPC CNI**

The **essential workloads** are deployed.

READY,

SET,

... uh oh

Issue: **Poor disk performance**

Turns out we were using the **default EKS Storage Classes**.

- Previous generation volumes
- Slowest IOPS option
  - Min: **100 IOPS**
  - Max: 5000 IOPS

Resolved by:

- Install **Amazon EBS CSI driver**
- Upgrade Storage Classes

Issue: **Docker Hub rate-limits**

Pods failing to pull images due to Docker rate-limiting.

- Load tests triggered **pod scaling**
  - e.g. 10s of pods to 100s of pods
- AWS NAT Gateway uses a **single IP address** for egress
- GCP provides an **image cache** by default

Resolved by:

- Set up Docker registry mirror as **pull through cache**
- Update **node pools** to utilize the internal mirror

4

Load test your applications

The cloud is infinite.

Niko
Early career days

The cloud is ~~infinite~~.

MYTH

Niko

Early career days

Quotas

New accounts start with (small) default service limits.

- Regions supported by **physical data centers**
- We’re **blind to capacity** of the regions we rely on
- Quotas increased via **support tickets**
  - Usually resolved within minutes
  - Sometimes days
  - Sometimes… not at all

5

Know your quotas

Recap

Similar, but not the same

Just get started

Don’t blindly do what was done before

Load test your applications

Know your quotas

What went well?

- **Kubernetes**
  - Most **Deployments**, **StatefulSets** and other workloads “just worked”
  - Tweaks required for
    - **Ingress**
    - **Services** of type **LoadBalancer**
    - Applications/configs which referenced object storage buckets

Grafana Labs

Gracias


Grafana Labs