Cloud Engineering
Summit

DAY-TWO OPERATION: MULTI-CLOUD KUBERNETES AND VAULT

CHARLES XU I OCTOBER 2021

PRESENTER

Charles Xu
Senior Software Engineer
Container Platform @Snowflake

Contributor to Istio, Kubernetes, Isopod, Skycfg

Previously at cruise Google

Learn more about me at
https://charlesxu.io/about

AGENDA

- Key Takeaways
- Snowflake Overview
- Container Platform Overview
- Cloud-agnostic Abstraction
- Cloud Resource Provisioning with Pulumi
  - Blue-green Node Pool Upgrades
  - Cloud-specific Kubernetes Manifests
- Vault Management With Pulumi
- Open Questions with Multi-cloud

KEY TAKEAWAYS

- Cloud-agnostic abstraction prevents fragmentation and proliferation of identities, policies, and toolings but is not always possible.
- Declarative infrastructure is insufficient to solve life cycle management. Invest in a tool that allows orchestration.
- Your infrastructure provisioner is either your metadata store or orchestrated by a metadata store, where the store must be queryable.

THE SNOWFLAKE DATA CLOUD

DATA SOURCES

OLTP DATABASES

ENTERPRISE APPLICATIONS

THIRD-PARTY

WEB/LOG DATA

IoT

Google Cloud aws Azure

DATA CONSUMERS

DATA MONETIZATION

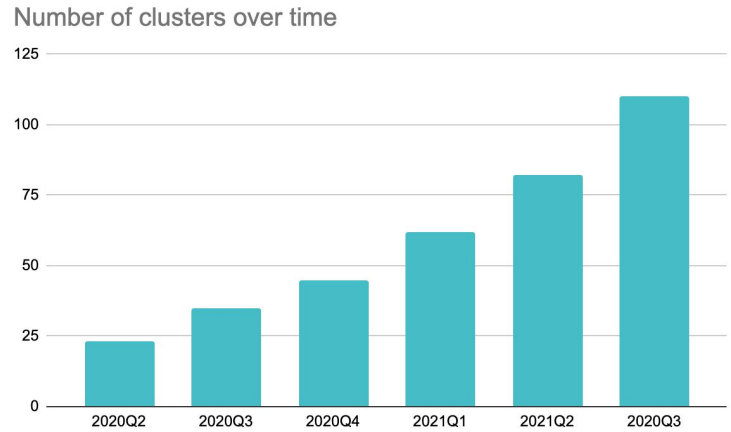
OPERATIONAL REPORTING

AD HOC ANALYSIS

REAL-TIME ANALYTICS

PLATFORM OVERVIEW

- 110 Kubernetes clusters and counting
- 3,000 Nodes / 60,000 Containers
- EKS, AKS, GKE
- Regional clusters, around the world
- Multitenant, in terms of teams and apps
- Integrated with legacy VM-based infra



CLOUD-AGNOSTIC ABSTRACTION

- Cloud-agnostic abstraction prevents fragmentation and proliferation of identities, policies, and toolings

Identity and group membership	Okta
Access	Teleport
Policy	Push up to Istio, Calico, OPA Gatekeeper
Multi-tenancy	CRDs for Vault Policy, Blob Storage, KMS
Logging and monitoring	Snowflake

CLOUD RESOURCES PROVISIONING

- Pulumi enables automation toolings and hence rapid infra scaling
- Each cluster backed by Pulumi multi-stacks
  - **Network**: VPC, DNS, ILB, Peering, NAT, Firewall Rules, etc
  - **Compute**: K8s Cluster, IAM, Custom Role, Pod Identity, etc
  - **SQL**: Managed Database, Users, etc
- Automation API usage examples (will cover first 2)
  - Blue-green upgrades for Kubernetes node pools
  - Generating cloud-specific Kubernetes manifests
  - Custom rollout strategy for Pulumi stacks
  - Pulumi operator, agent-pull CICD

NODE POOL BLUE-GREEN UPGRADES

Background

- blue-green upgrades allow fast revert, no stuck between
- Upgrades steps:
  - Create node pools with new version
  - Cordon and drain old node pools
  - Delete old node pools after workloads healthy

Problems

- Manual upgrade on hundreds of clusters error-prone, does not scale
- Could not rely on cloud providers' auto-upgrade because
  - We have special Istio ingress setup
  - Azure has no cloud-native load balancer
  - We value consistent architecture across cloud

NODE POOL BLUE-GREEN UPGRADES, CON'T

Automation

- Custom tooling above Pulumi Automation API
  - Edit and apply Pulumi stacks according to blue-green upgrade steps
  - One step at a time; in between steps are pre- and post-condition checks to cloud provider and k8s cluster
- Unlike other infra-as-code systems, Pulumi requires no DSL. Its Automation API and client lib unlock all kinds of orchestration needs

Declarative infrastructure is insufficient to solve life cycle management. Invest in a tool that allows orchestration.

CLOUD-SPECIFIC KUBERNETES MANIFESTS

Background

- Apps need customized k8s manifests to run on multiple clusters/clouds
  - Container image host, load balancer label, Pod Identity

Problems

- We manage k8s manifests outside of Pulumi, because
  - We employ open source projects that only release installation yaml.
    - E.g. cert-manager, external-dns, calico, etc
  - Cluster states often digress from Infra-as-code states
    - Controllers, HPA,
  - We run a multi-tenant platform
- Cluster-specific values to customize k8s manifests are defined in Pulumi

CLOUD-SPECIFIC KUBERNETES MANIFESTS

- We built Overlaymgr, a yaml rendering engine
  - Reads Pulumi stack outputs
  - Kustomize + ArgoCD
- DSL front-end is another viable choice for k8s configuration management but still needs input data
  - cruise-automation/isopod
  - cuelang/cue

Your infrastructure provisioner is either your metadata store or orchestrated by a metadata store, where the store must be queryable.

VAULT MANAGEMENT WITH PULUMI

- Vault is a critical piece in our secret as a service and private PKI
- Use Pulumi to initialize and configure Vault
  - Rotating issuing certs of cert-manager
  - Replicate static secrets across deployments
  - Cloud-provider secret engine for short-lived tokens
- Tenant onboarding is outside of Pulumi, done using VaultPolicy CRD

```
apiVersion: security.snowflake.com/v1alpha1
kind: VaultPolicy
metadata:
  name: foo
  namespace: foo-ns
spec:
  serviceAccount: foo-sa
  vaultPolicy: |
    path "secret/foo/*" {
        capabilities = ["read", "list"]
    }
    path "secret/bar/*" {
        capabilities = ["update"]
    }
```

OPEN QUESTIONS WITH MULTI-CLOUD

- Applications still need cloud-specific client libraries to interact with cloud services, requiring code change for each cloud provider we want to support
  - E.g. reading from a blob storage bucket, write to a message queue
  - Pod identity only solves authNZ
  - CRDs only solve resource provisioning
- Cloud-agnostic abstraction often terminates at Kubernetes. Cloud resources and policies that cannot be pushed up remain heterogeneous across clouds
  - E.g. GCP CloudDNS has no record-level permission control since IAM boundary is the entire GCP Project

THANK YOU

We are hiring!

Happy to chat if you are excited like us about multi-cloud, distributed systems, container platforms, and open-source software!