An In-Memory Technology's Journey to the Cloud

...

Huseyin BABAL
Software Development Team Lead, Hazelcast Cloud

About me

Currently Implementing Hazelcast Cloud

Ex-Sony and Ex-eBay Engineer (DevOps & Microservice Transformation Project Architect)

Regularly Talk & Do Workshops about all the stuffs I know on public events

Outline

- Infrastructure Abstraction
- Control / Data Plane
- Resource Management
- Persistence
- Monitoring
- Access Management

Infrastructure Abstraction

...

Cloud Provider Challenges

Having different restriction on each cloud provider introduces a big challenge while you are creating a product. Terms are different, resource creation strategy is different, managing the identities are different, etc....

What to do for eliminating those obstacles?

Kubernetes

Open-source container orchestration system for managing application deployment, scaling, etc...

Kubernetes does not create your infra, it needs VMs at least to form a cluster.

All well-known cloud providers has Kubernetes as a Service

Abstraction with Kubernetes

Service[type=LoadBalancer] => creates load balancer

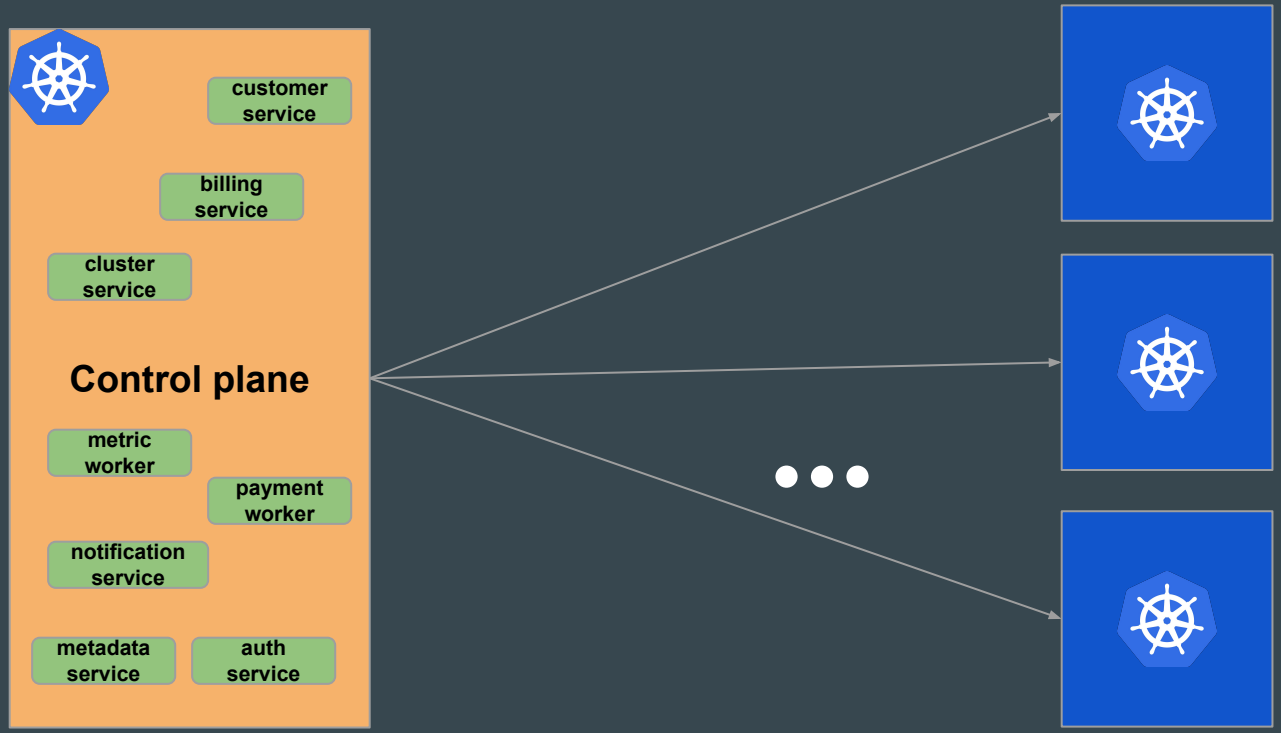
PVC => creates volume/disk

Helm => Unified application dependency management

Google Cloud

Control / Data Plane

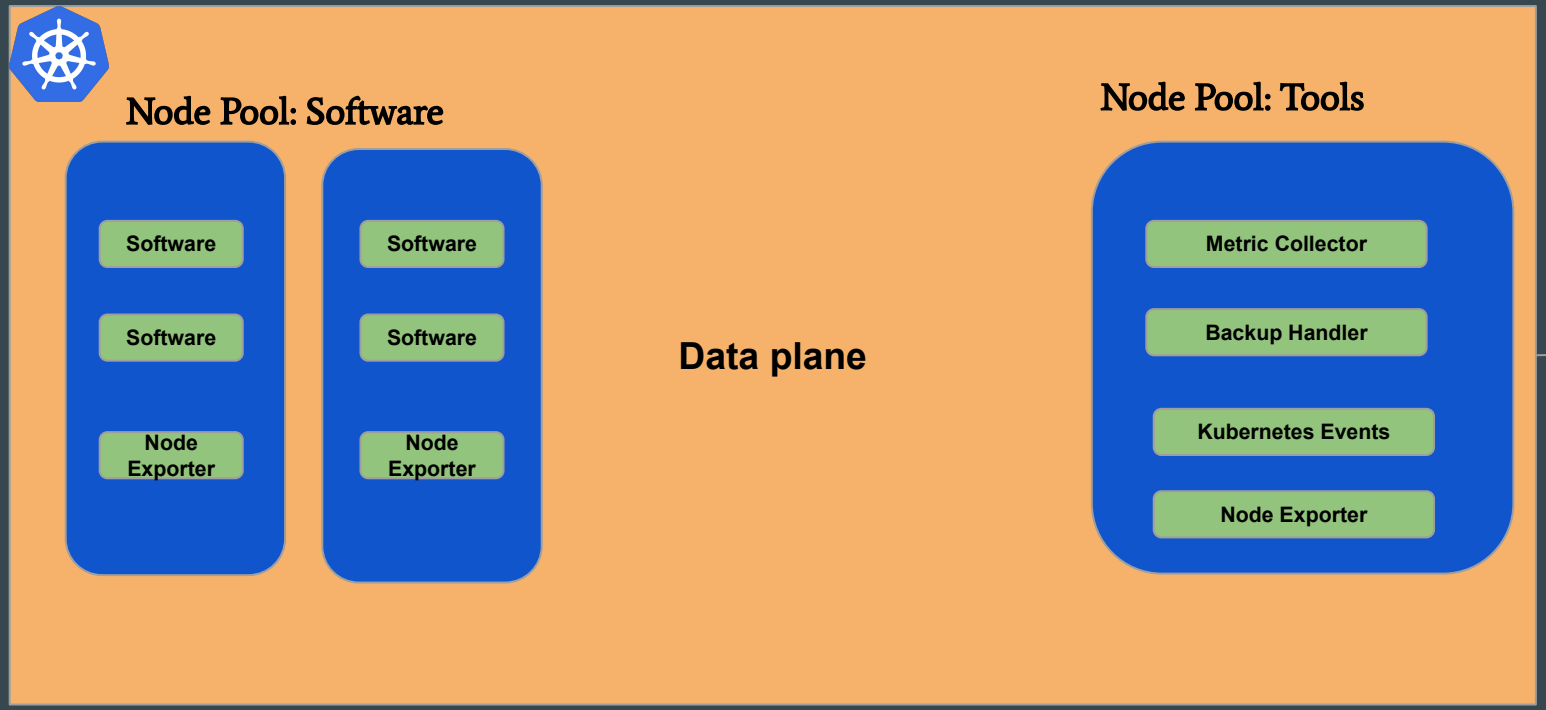
...


customer service
billing service
cluster service
Control plane
metric worker
payment worker
notification service
metadata service
auth service

Control Plane

This is the place where you orchestrate your SaaS system, and it can;

- do business validation before creating actual resources on k8s
- connect any kind of k8s destination defined within control plane database
- consume metrics and take action based on criterias like scale up when memory > 80%


Node Pool: Software
Software
Software
Node Exporter
Software
Software
Node Exporter
Data plane
Node Pool: Tools
Metric Collector
Backup Handler
Kubernetes Events
Node Exporter

Data Plane

Data Plane is responsible for handling **provided service** workload.

- It primarily contains Software within SaaS
- It may contain internal agents for management and monitoring purposes
- It may have public or private network topology
- It may contain different node pools for different purposes.

Application Deployment

Use helm to package your application with its possible configuration options to make an easy deployment

Implement operator of your application if you think custom resources are needed for your application to manage its state successfully

Resource Management

...

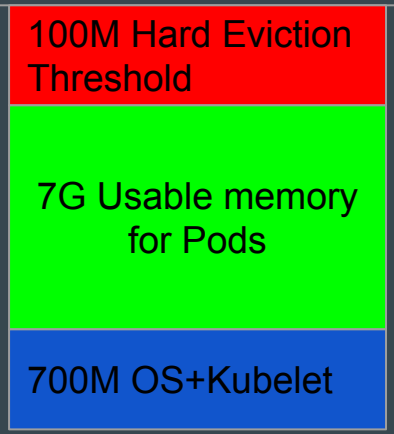
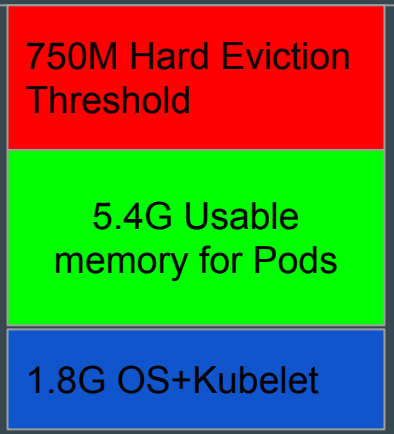
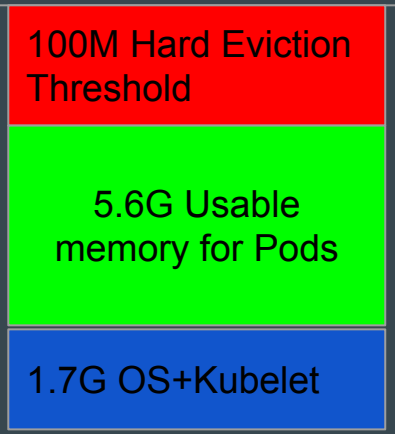
Know Your K8s Node Limits

When you create a cluster on EKS, AKS, or GKE they actually spin up VMs on the background and they reserve certain amount of resource of that VMs for their internal usages.

If customer select 16G cluster, you cannot just use 2 m5.large on the background to install on top of that.

Know Your K8s Node Limits

What if you have use following machines for 3 different cloud providers;

AWS	Azure	GCP
M5.large (8G, 2 CPU)	D2 v3 (8G 2CPU)	n1-standard-2 (7.5 G, 2 CPU)
		

Pod Resource Management

In SaaS services, customers request services and they pay for it. To provide qualified service, there should be properly defined limit of an application inside kubernetes.

Applications are just workloads inside k8s and you can define requests / limits for them

Requests / Limits

Requests is for saying “How much memory / cpu needed” for this application.

Limits is for saying “Up to how much memory / cpu can be used” by this application.

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: metric-consumer
spec:
  containers:
  - name: metric
    image: metric/consumer
    resources:
      requests:
        memory: "300Mi"
        cpu: "250m"
      limits:
        memory: "600Mi"
        cpu: "500m"
```

Persistence

...

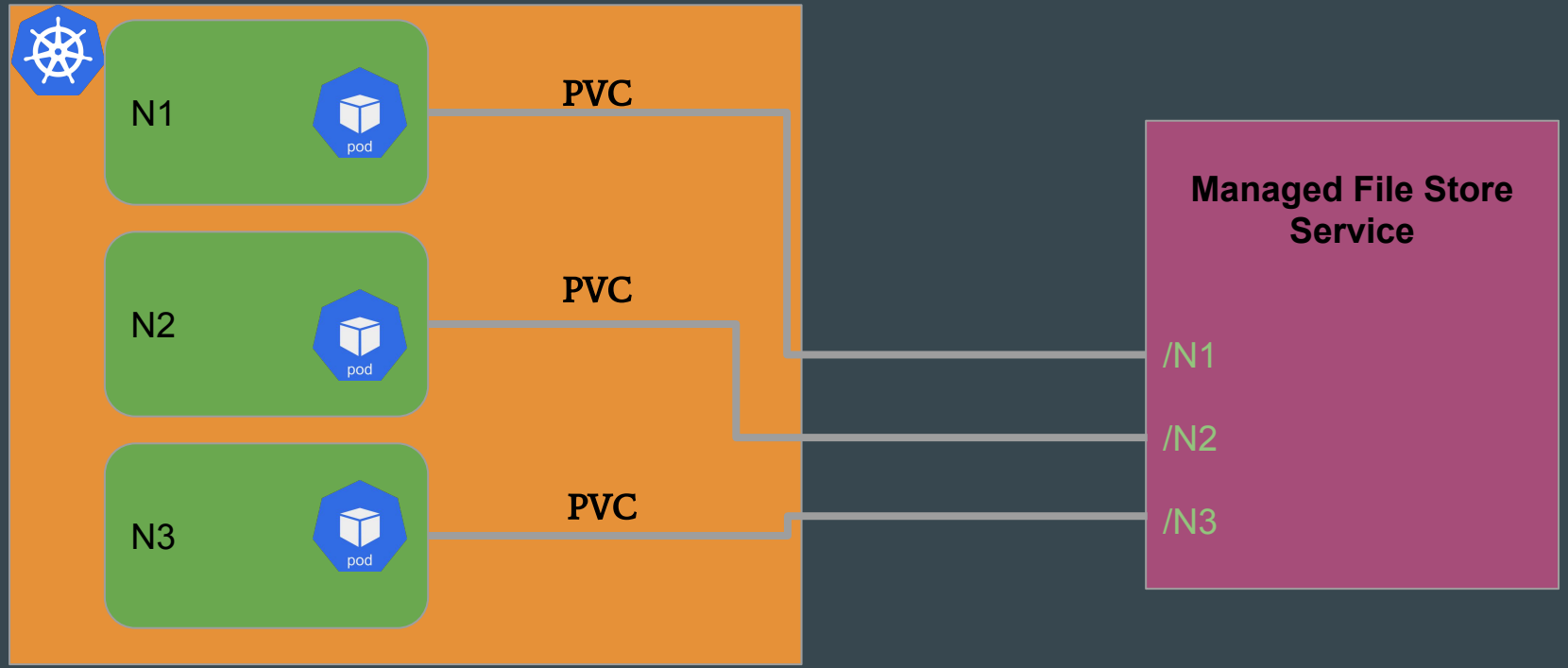
Real Challenge

We say in-memory, but what is this persistence?

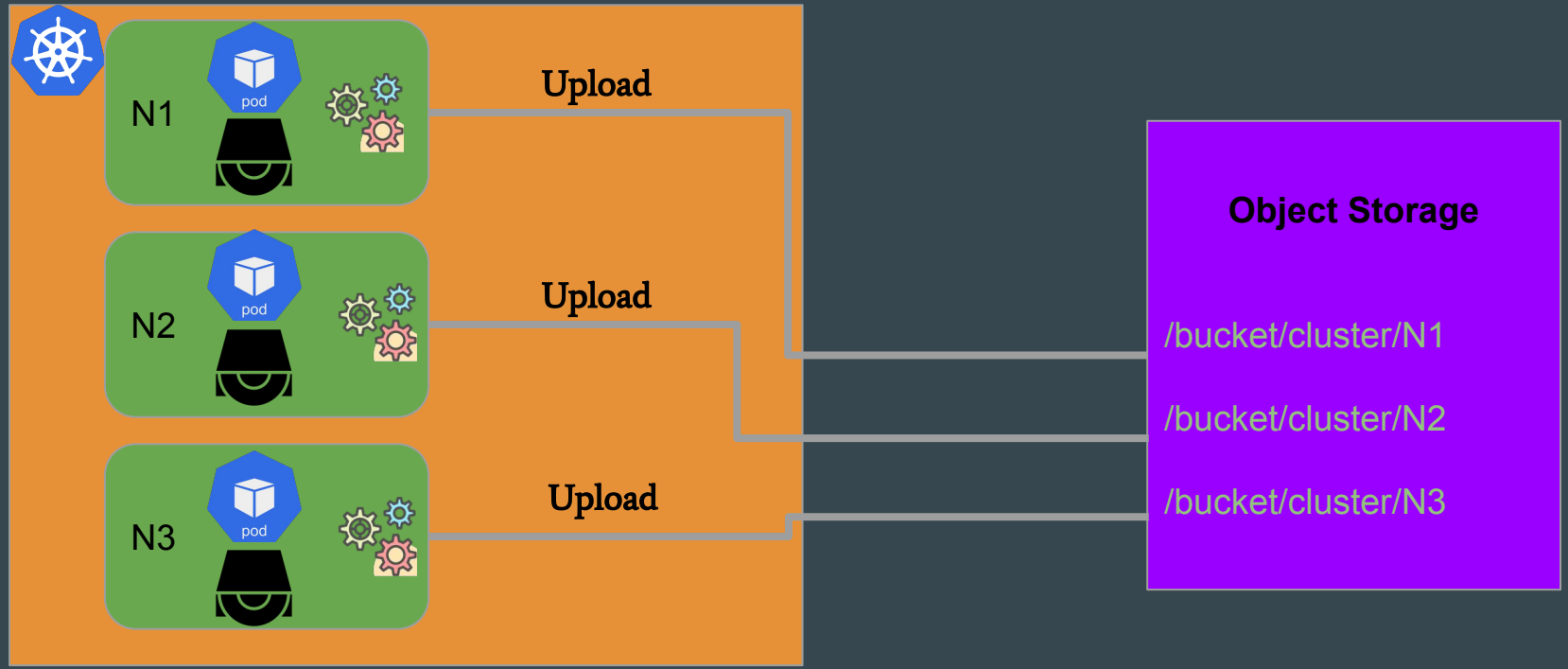
In-memory techs may provide a way to persist in-memory data to disk on demand.

In a distributed system, it is the real challenge.

With Managed File Store Service



Without Managed File Store Service



Motivation of Persistence

- Taking Snapshot of cluster data at time T
- Restore from snapshot in for disaster recovery
- Let customer to clone a cluster by creating new cluster and provide snapshot data during cluster startup

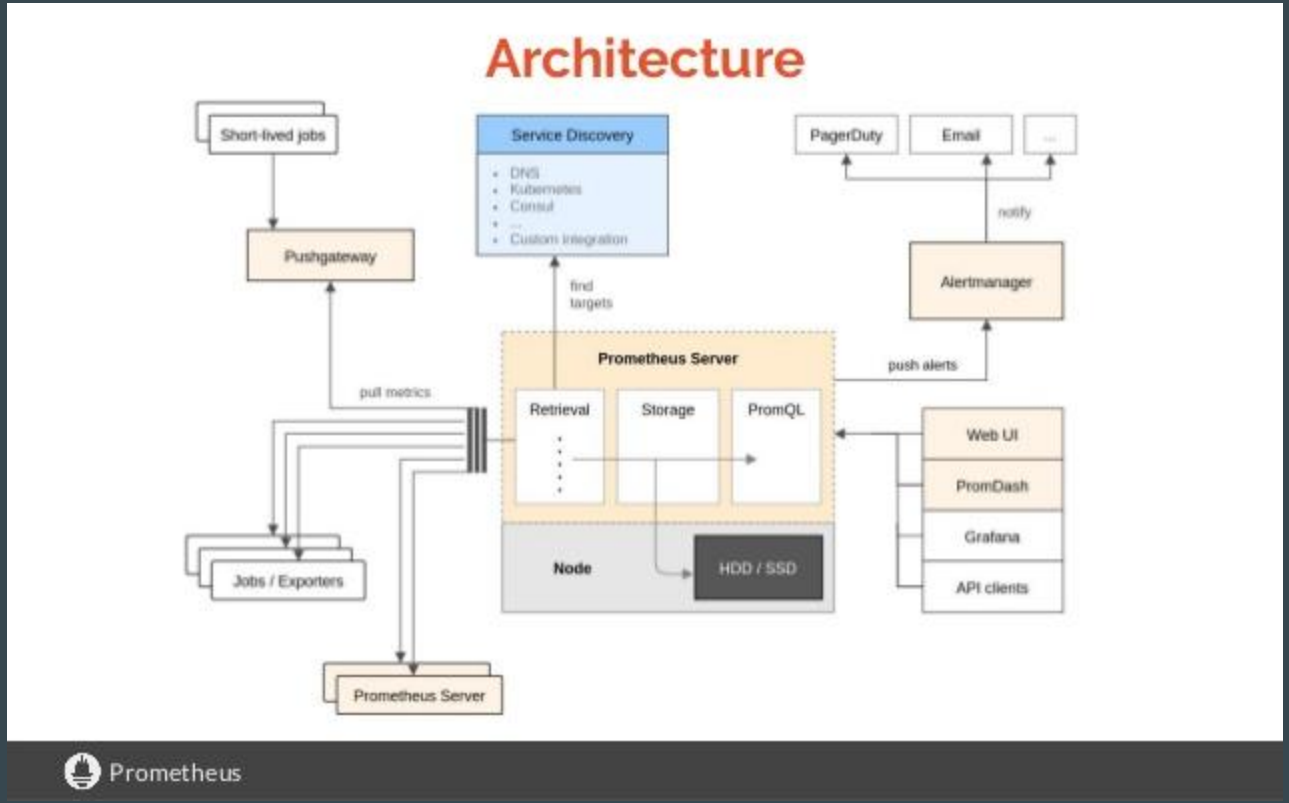
Persistence Components

- There is a daemonset to have a uploader agent on every node to upload data for snapshot operations
- There is a daemonset to have a downloader agent on every node to download data in advance for restore or clone operations
- Since those agents within same node with actual technology you provide, they must be as tiny as possible
- Best practice to use workload identity for agents to be able to access object storage without any kind of credentials

Monitoring

...

Prometheus

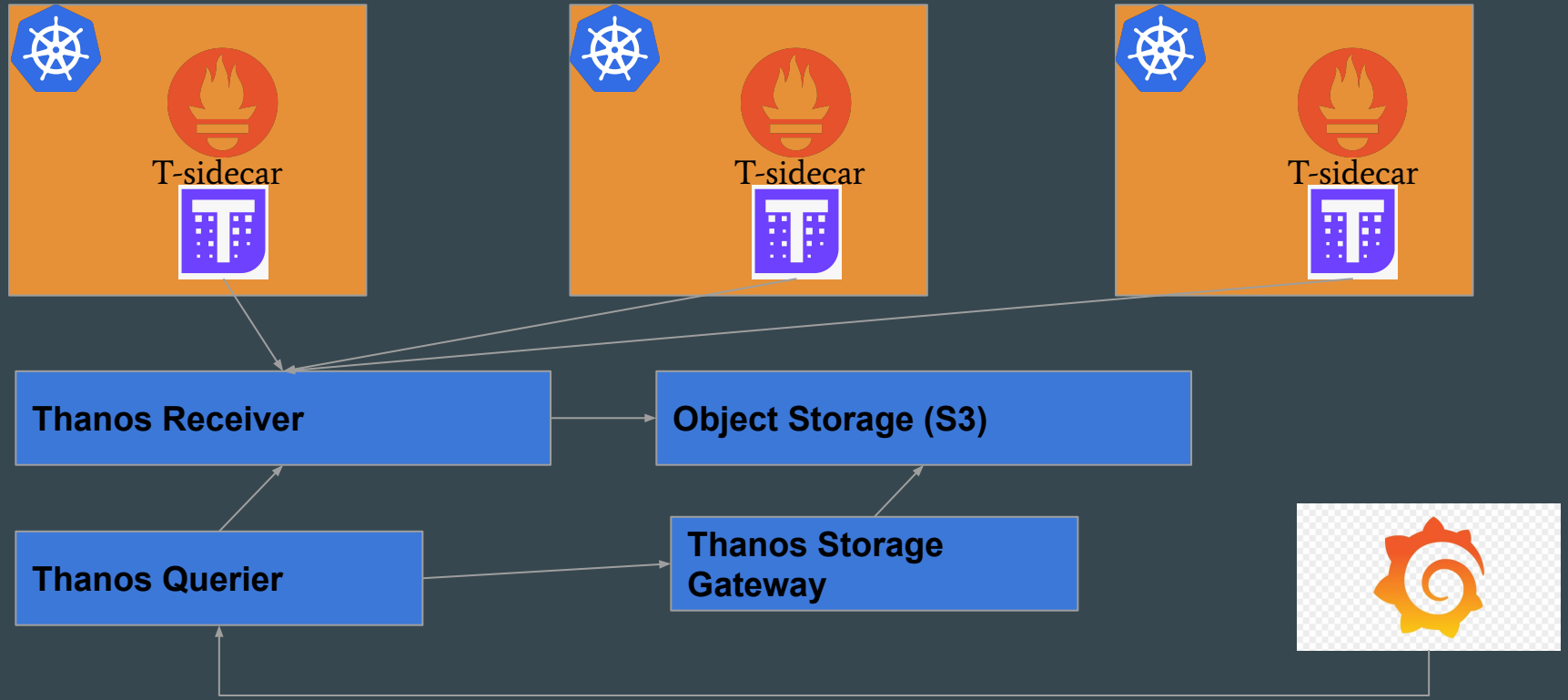


*https://www.slideshare.net/QAware/cloud-monitoring-with-prometheus

Monitoring

- Prometheus for monitoring to collect metrics from targets
- Implement your own metric exporter to be scraped by Prometheus
- Define Prometheus Rules to let AlertManager to send notifications
- Introduce a central monitoring system to handle metrics coming from different clusters in one place
- Use Thanos to have scalable monitoring system

Monitoring Multi-Cluster



Alert Rules

Through Thanos Querier, you can get built-in metrics and custom ones. By using those metrics, you can also trigger alerts like;

- If used_memory > 80 then fire alarm to notify customer
- If used_memory < 40 then fire alarm to scale down
- If used_memory > 90 then fire alarm to scale up

Inside prometheus operator, you can find custom resource PrometheusRule

Access Management

...

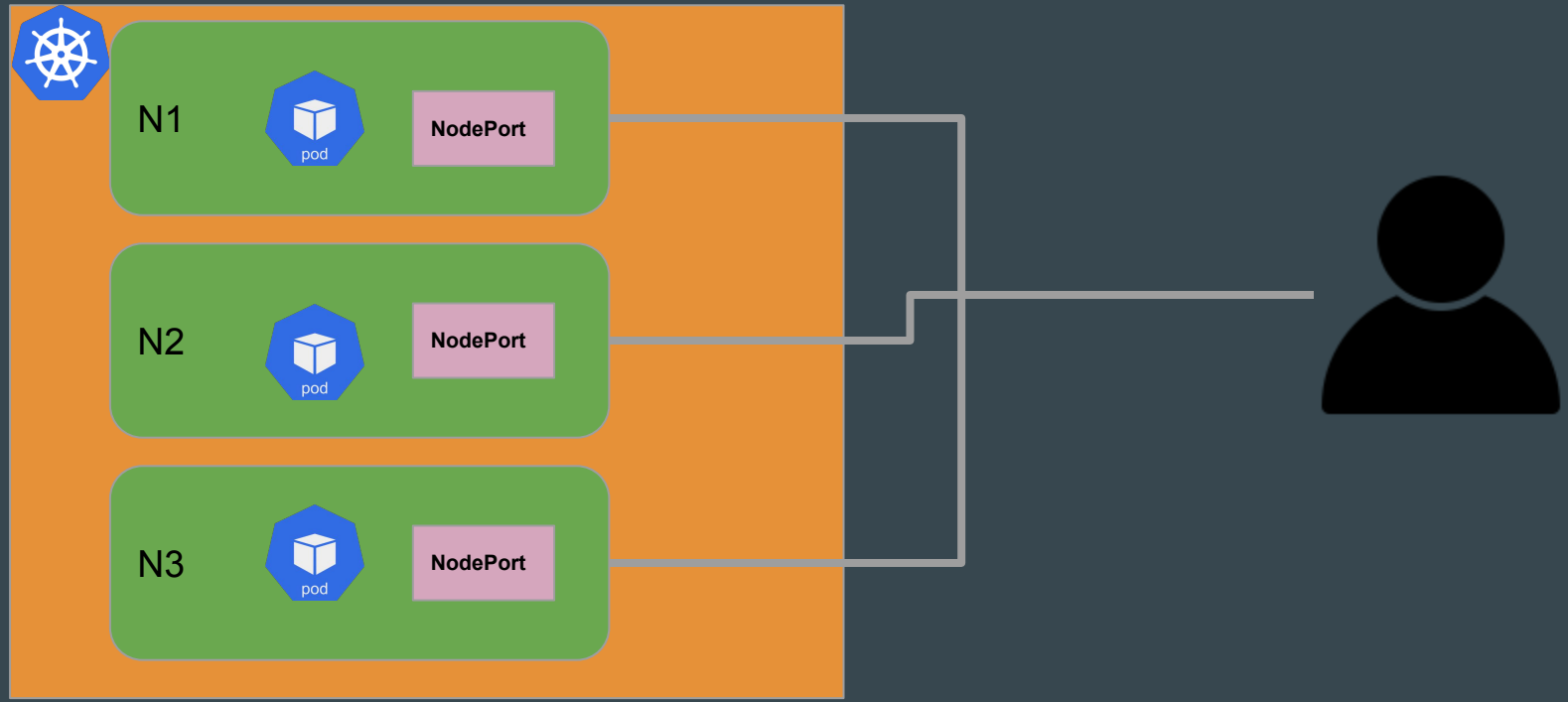
Network Topologies

According to business needs, you may want to setup cluster in a private or public network.

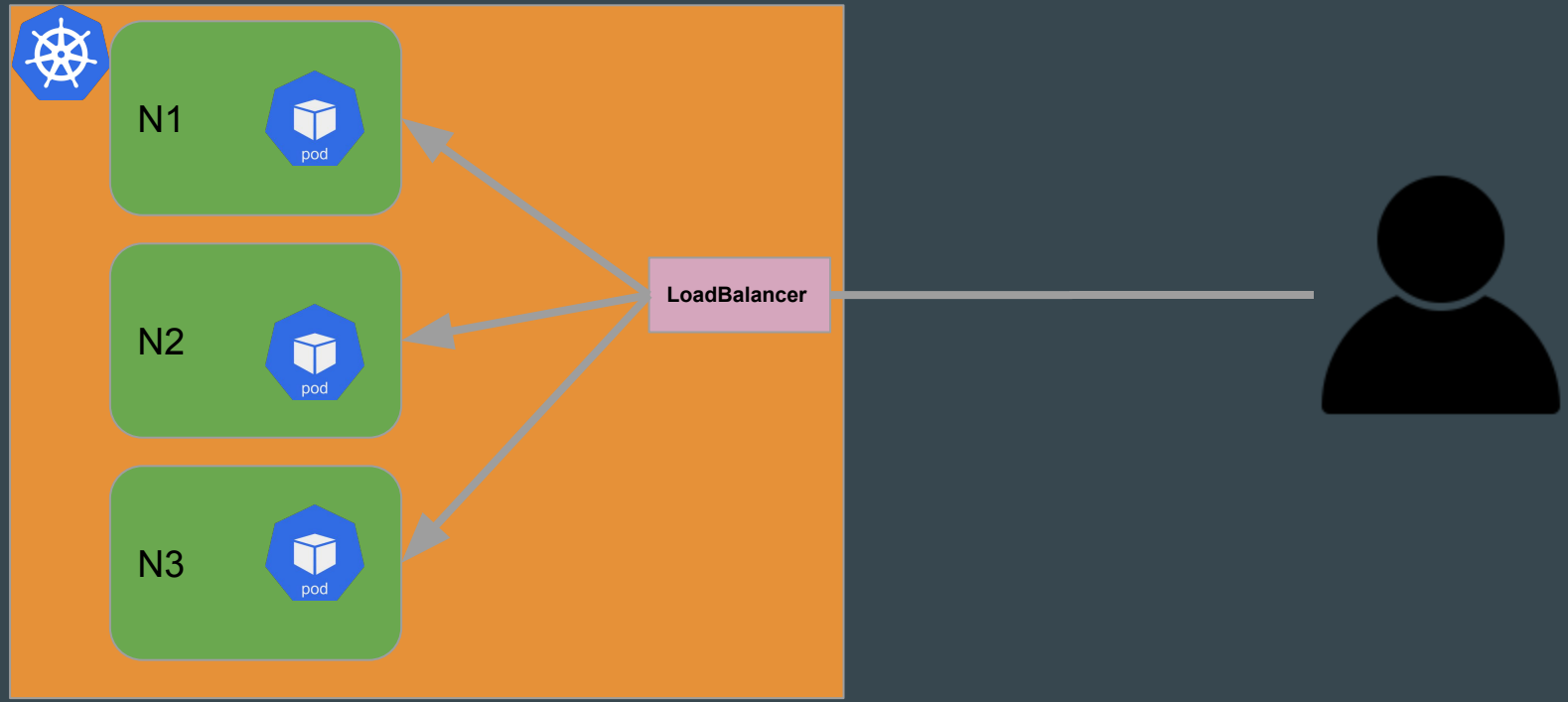
On all cloud providers, they provide network topology type to put k8s in desired network type.

Accessing public cluster is ok, but private one is a bit challenging

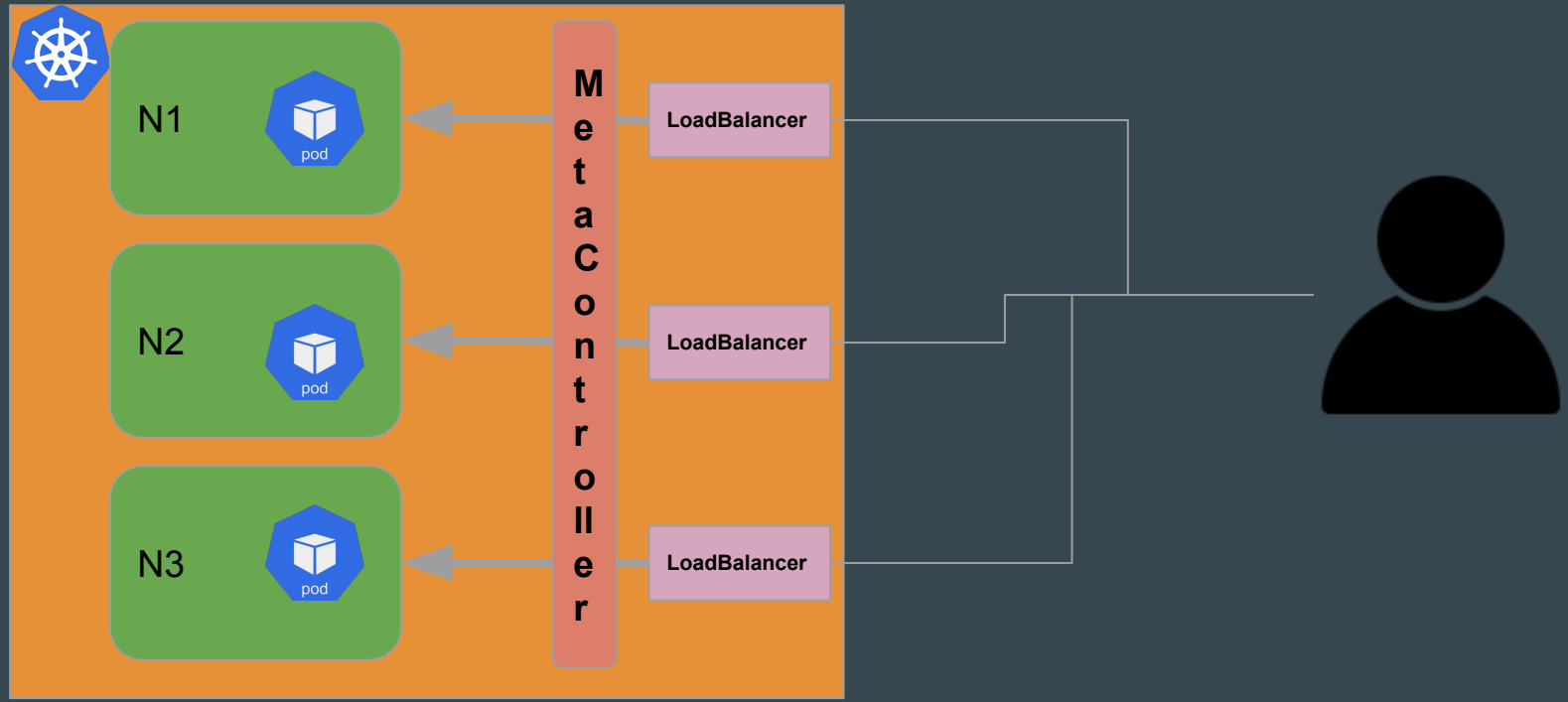
Service Exposal with NodePort



Service Exposal with LoadBalancer



Service Exposal with NodePort as LoadBalancer



Service Discovery

In modern world, you shouldn’t depend on static hostnames, if you are inside k8s, you can listen events to detect scheduled pods to get ip address of physical machines.

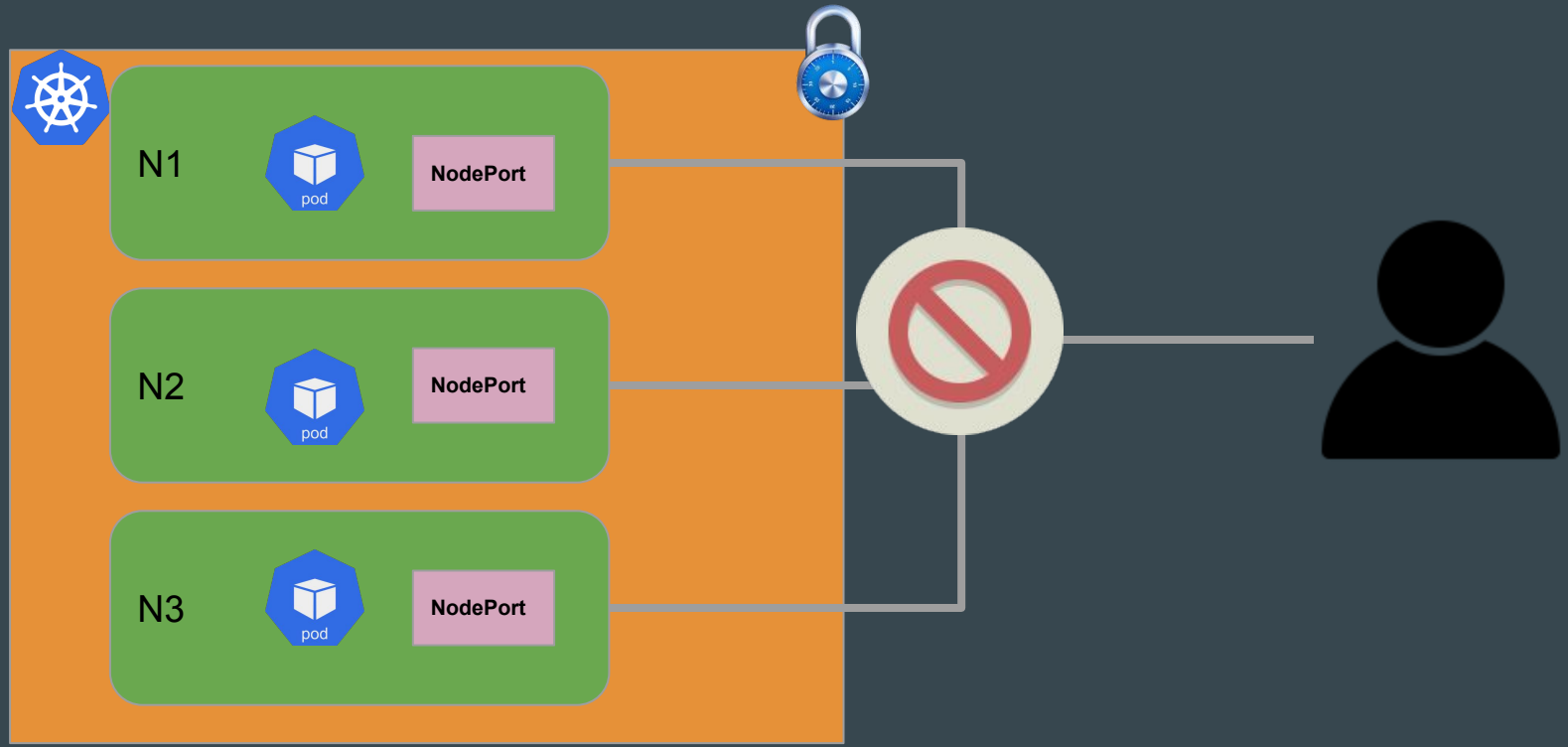
To connect members, user needs to hit discovery endpoint with cluster id to get member ip addresses

Private Clusters

It is easy to connect to cluster which is inside k8s cluster that has public network topology.

What about private ones?

Private Cluster



Peering Scenario

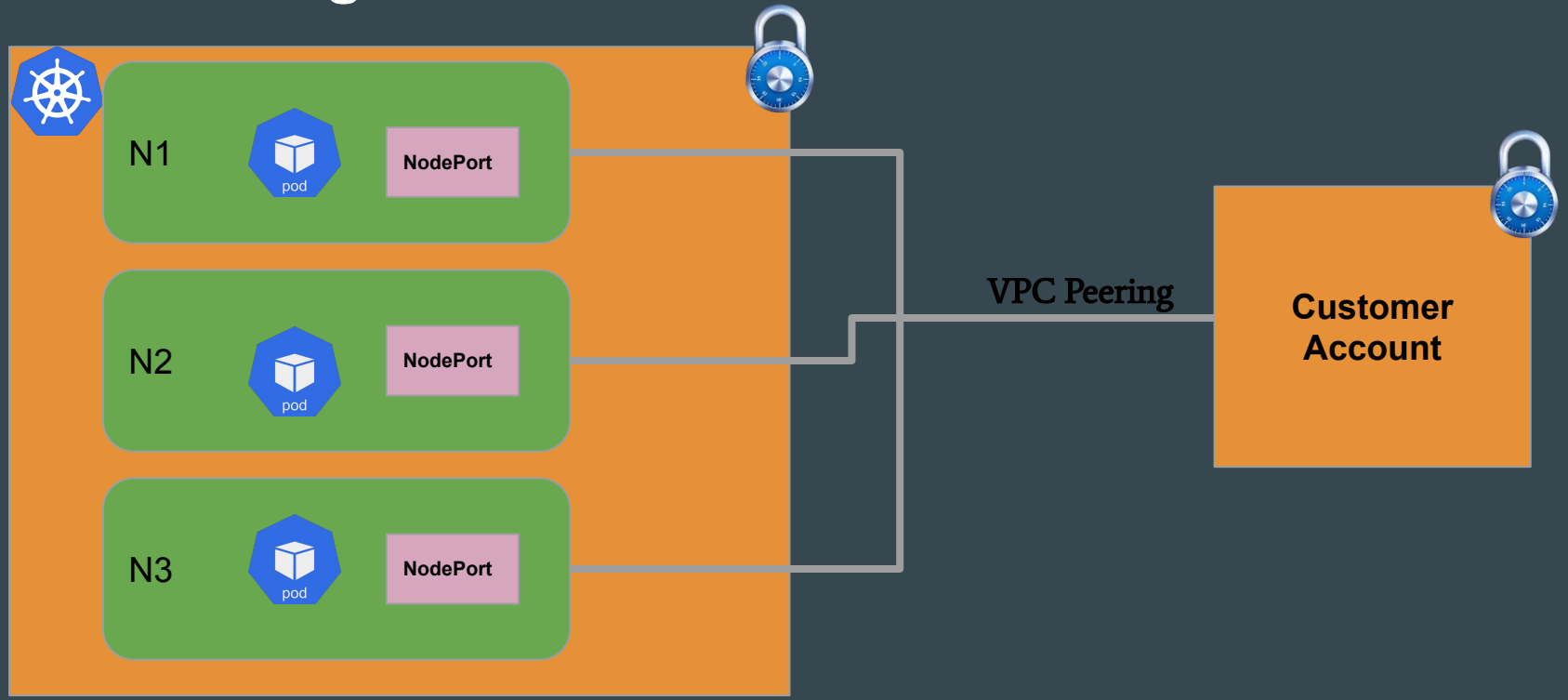
- In Console UI, customer get pre-generated cli command
- Initiates VPC Peering on customer side
- CLI prompts necessary parameters like VPC ID, Subnet, etc...
- It creates VPC Peering connection on customer side and sends request to Control Plane to create same records to your system to verify

Cloud Specific Peering Terms

	AWS	Azure	GCP
Name	VPC Peering	vNet Peering	VPC Network Peering
Requested Parameters	Account ID VPC ID Subnet ID	vNet ID	Project ID Network Name

In AWS, you can also use Private Link to convert your service into a VPC Endpoint Service. It is also best practice for enabling your service in AWS Marketplace

VPC Peering


N1
pod
NodePort
N2
pod
NodePort
N3
pod
NodePort
VPC Peering
Customer Account

Any Question?


/huseyinbabal


/huseyinbabal


https://huseyinbabal.com