Control Plane Components

Kubernetes Control Plane

kube-controller manager

cloud-controller manager

Cloud

kube-api-server

etcd

kube-scheduler

kubelet

kube-proxy

kubelet

kube-proxy

kubelet

kube-proxy

Kubernetes Nodes

Provision of K8s with A

interaction with kubectl CLI

Cloud Provider API

kubectl

Ctrl Plane - 1,2...n

Containers

cloud-controller-manager

kube-controller-manager

kube-scheduler

kube-proxy

Additional Services

kube-apiserver

etcd

Container Runtime

kubelet

System Services

Responsibility of AWS

Node 1

Containers

pod

pod

kube-proxy

Container Runtime

kubelet

System Services

Node 2

Containers

pod

pod

kube-proxy

Container Runtime

kubelet

System Services

Responsibility of Customer

Cloud Provider Network Edge

Load Balancer

End Users

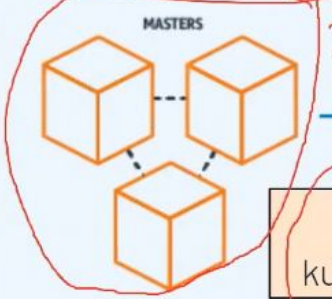
ssh connection

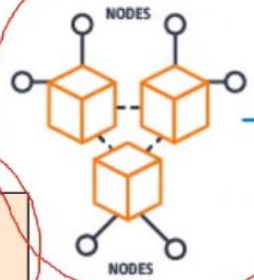
Provision of K8s with AWS EKS

kubectl & aws cli configuration

creating a kubeconfig file

Configuring Cluster Autoscaler





Provision an EKS cluster

EKS automatically deploys Kubernetes masters

Deploy worker nodes

Add worker nodes to your EKS cluster

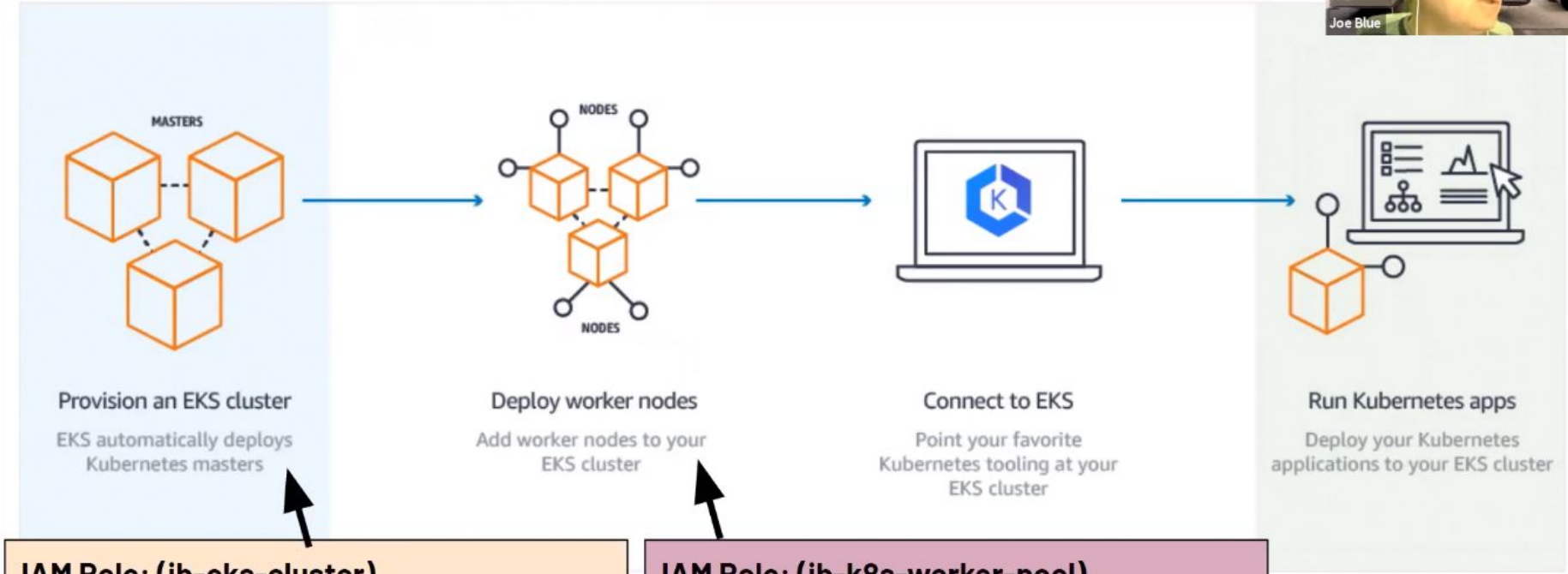
Connect to EKS

Point your favorite Kubernetes tooling at your EKS cluster

Run Kubernetes apps

Deploy your Kubernetes applications to your EKS cluster

Role & Policies

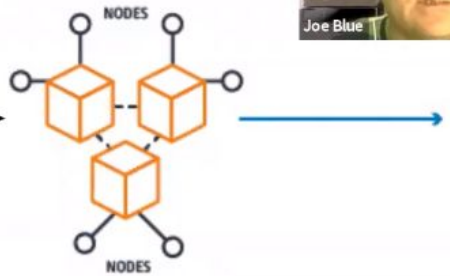


IAM Role: (jb-eks-cluster)
EKS - Cluster policy:
- AmazonEKSClusterPolicy

IAM Role: (jb-k8s-worker-pool)
EC2 - Common policies
- AmazonEKSWorkerNodePolicy
- AmazonEC2ContainerRegistryReadOnly
- AmazonEKS_CNI_Policy

Attach ClusterAutoscalerPolicy to Role

IAM Role: (jb-k8s-worker-pool)
EC2 - Common policies
- AmazonEKSWorkerNodePolicy
- AmazonEC2ContainerRegistryReadOnly
- AmazonEKS_CNI_Policy



Deploy worker nodes

Add worker nodes to your cluster

Attach this policy to the IAM Worker Node Role

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Action": [
            "autoscaling:DescribeAutoScalingGroups",
            "autoscaling:DescribeAutoScalingInstances",
            "autoscaling:DescribeLaunchConfigurations",
            "autoscaling:DescribeTags",
            "autoscaling:SetDesiredCapacity",
            "autoscaling:TerminateInstanceInAutoScalingGroup",
            "ec2:DescribeLaunchTemplateVersions"
         ],
         "Resource": "*",
         "Effect": "Allow"
      }
   ]
}
```

ClusterAutoscalerPolicy

- On the `Cluster Service Role` field; give general description about why we need this role.
- Create EKS Cluster Role with `EKS - Cluster` use case and `AmazonEKSClusterPolicy`.
  - EKS Cluster Role :
    - use case : `EKS - Cluster`
    - permissions: `AmazonEKSClusterPolicy`.

Attached permissions policies

The type of role that you selected requires the following policy.

Filter policies | Search | Showing 1 result

	Policy name	Used as	Description
▸	AmazonEKSClusterPolicy	Permissions policy (2)	This policy provides Kubernetes the permissio…

Elastic Kubernetes Service (Amazon EKS)

Fully managed Kubernetes control plane

Amazon EKS is a managed service that makes it easy for you to use Kubernetes on AWS without needing to install and operate your own Kubernetes control plane.

Create EKS cluster

Cluster name

m-k8s-cluster

Next step

Pricing

Step 1
Configure cluster

Step 2
Specify networking

Step 3
Configure logging

Step 4
Review and create

Configure cluster

Cluster configuration Info

Name - *Not editable after creation.*
Enter a unique name for this cluster.

m-k8s-cluster

Kubernetes version Info
Select the Kubernetes version for this cluster.

1.18

Cluster Service Role Info - *Not editable after creation.*
Select the IAM Role to allow the Kubernetes control plane to manage AWS resources on your behalf.
To create a new role, go to the **IAM console**.

me-eks-cluster-09

EKS > Clusters > Create EKS cluster

Step 1
Configure cluster

Step 2
Specify networking

Step 3
Configure logging

Step 4
Review and create

Specify networking

Networking Info

These properties cannot be changed after the cluster is created.

VPC Info

Select a VPC to use for your EKS Cluster resources.
To create a new VPC, go to the **VPC console**.

vpc-440ff339 | Default

Subnets Info

Choose the subnets in your VPC where the control plane may place elastic network interfaces (ENIs) to facilitate communication with your cluster.
To create a new subnet, go to the corresponding page in the **VPC console**.

Select subnets

subnet-e52a8a83 ✕ subnet-2e8d230f ✕

Security groups Info

Choose the security groups to apply to the EKS-managed Elastic Network Interfaces that are created in your worker node subnets.
To create a new security group, go to the corresponding page in the **VPC console**.

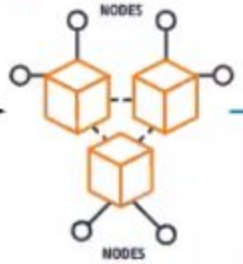
Select security groups

sg-8ad721b7 ✕

Configure Kubernetes Service IP address range Info

Attach ClusterAutoscalerPolicy to Role

IAM Role: (jb-k8s-worker-pool)
EC2 - Common policies
- AmazonEKSWorkerNodePolicy
- AmazonEC2ContainerRegistryReadOnly
- AmazonEKS_CNI_Policy



Deploy worker nodes

Add worker nodes to your cluster

Attach this policy to the IAM Worker Node Role

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "autoscaling:DescribeAutoScalingGroups",
        "autoscaling:DescribeAutoScalingInstances",
        "autoscaling:DescribeLaunchConfigurations",
        "autoscaling:DescribeTags",
        "autoscaling:SetDesiredCapacity",
        "autoscaling:TerminateInstanceInAutoScalingGroup",
        "ec2:DescribeLaunchTemplateVersions"
      ],
      "Resource": "*",
      "Effect": "Allow"
    }
  ]
}
```

ClusterAutoscalerPolicy

Select type of trusted entity

AWS service
EC2, Lambda and others

Another AWS account
Belonging to you or 3rd party

Web identity
Cognito or any OpenID provider

SAML 2.0 federation
Your corporate directory

Allows AWS services to perform actions on your behalf. Learn more

Choose a use case

Common use cases

EC2
Allows EC2 instances to call AWS services on your behalf.

Lambda
Allows Lambda functions to call AWS services on your behalf.

- For the node's IAM Role, get to IAM console and create a new role with `EC2 - Common` use case having the policies of `AmazonEKSWorkerNodePolicy, AmazonEC2ContainerRegistryReadOnly, AmazonEKS_CNI_Policy`.
  - `Use case: EC2`
  - `Policies: AmazonEKSWorkerNodePolicy, AmazonEC2ContainerRegistryReadOnly, AmazonEKS_CNI_Policy`

Review

Provide the required information below and review this role before you create it.

Role name* me-k8s-worker-pool-09

Use alphanumeric and '+=,.@-_' characters. Maximum 64 characters.

Role description Allows EC2 instances to call AWS services on your behalf.

Maximum 1000 characters. Use alphanumeric and '+=,.@-_' characters.

Trusted entities AWS service: ec2.amazonaws.com

Policies
- AmazonEC2ContainerRegistryReadOnly
- AmazonEKS_CNI_Policy
- AmazonEKSWorkerNodePolicy

Create policy

A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the visual editor and using JSON. Learn more

Visual editor | JSON

Import

```
1  {
2      "Version": "2012-10-17",
3      "Statement": [
4          {
5              "Action": [
6                  "autoscaling:DescribeAutoScalingGroups",
7                  "autoscaling:DescribeAutoScalingInstances",
8                  "autoscaling:DescribeLaunchConfigurations",
9                  "autoscaling:DescribeTags",
10                 "autoscaling:SetDesiredCapacity",
11                 "autoscaling:TerminateInstanceInAutoScalingGroup",
12                 "ec2:DescribeLaunchTemplateVersions"
13             ],
14             "Resource": "*",
15             "Effect": "Allow"
16         }
17     ]
```

me-clusterAutoScalerPolicy-09

Permissions | Trust relationships | Tags | Access Advisor | Revoke sessions

Permissions policies (3 policies applied)

Attach policies

Add inline polic

Policy name	Policy type
AmazonEKSWorkerNodePolicy	AWS managed policy
AmazonEC2ContainerRegistryReadOnly	AWS managed policy
AmazonEKS_CNI_Policy	AWS managed policy

Create role | Delete role

pool

Showing 2 res

Role name	Trusted entities	Last activity
me-k8s-worker-pool-08	**AWS service:** ec2	Yesterday
me-k8s-worker-pool-09	**AWS service:** ec2	None

Permissions policies (4 policies applied)

Attach policies

Policy name
AmazonEKSWorkerNodePolicy
AmazonEC2ContainerRegistryReadOnly
AmazonEKS_CNI_Policy
me-clusterAutoScalerPolicy-09

m-k8s-cluster

Active | Delete cluster

Overview | Workloads | Configuration

Cluster configuration Info

Kubernetes version Info
1.18

Platform version Info
eks.3

Details | Compute | Networking | Add-ons | Logging | Update history | Tags

Node Groups (0) Info

Edit | Delete | Add Node Group

Group name	Desired size	AMI release version	Launch template	Status

No Node Groups

Select instance types you prefer for this Node Group.

Select

t3.medium
vCPU: Up to 2 vCPUs memory: 4.0 GiB

Disk size
Select the size of the attached EBS volume for each node.

8 GiB

Node Group scaling configuration

Minimum size
Set the minimum number of nodes that the group can scale in to.

1 nodes

Maximum size
Set the maximum number of nodes that the group can scale out to.

2 nodes

Desired size
Set the desired number of nodes that the group should launch with initially.

1 nodes

Subnets Info

Specify the subnets in your VPC where your nodes will run.
To create a new subnet, go to the corresponding page in the **VPC console**.

Select subnets

subnet-e52a8a83 ✕ subnet-2e8d230f ✕

Allow remote access to nodes Info

Without remote access enabled you will not be able to directly connect to nodes after they are created.

SSH key pair

Select an SSH key pair to allow secure remote access to your nodes.
To create a new SSH key pair, go to the corresponding page in the **EC2 console**.

handson

Allow remote access from

Configure the source IP ranges that can remotely access nodes.

- (●) **All**
  Do not restrict source IPs that can remotely access nodes.
- ( ) **Selected security groups**
  Specify security groups to restrict which source IPs can remotely access nodes.

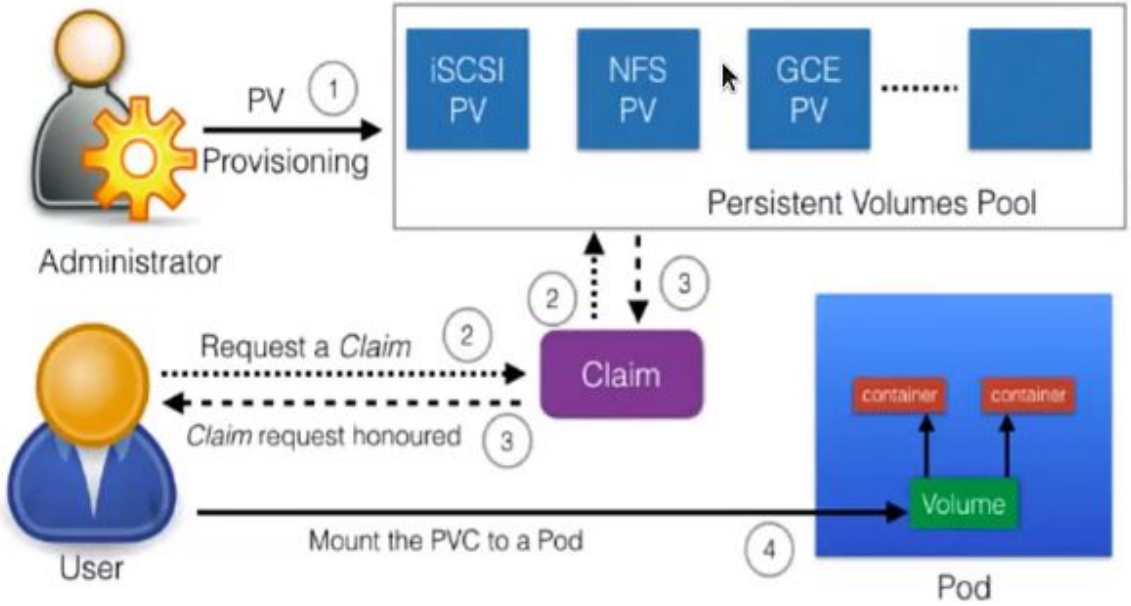
Outline

- Part 1 - Installing kubectl and eksctl on Amazon Linux 2
- Part 2 - Creating the Kubernetes Cluster on EKS
- Part 3 - Dynamic Volume Provisionining
- Part 4 - Ingress

Prerequisites

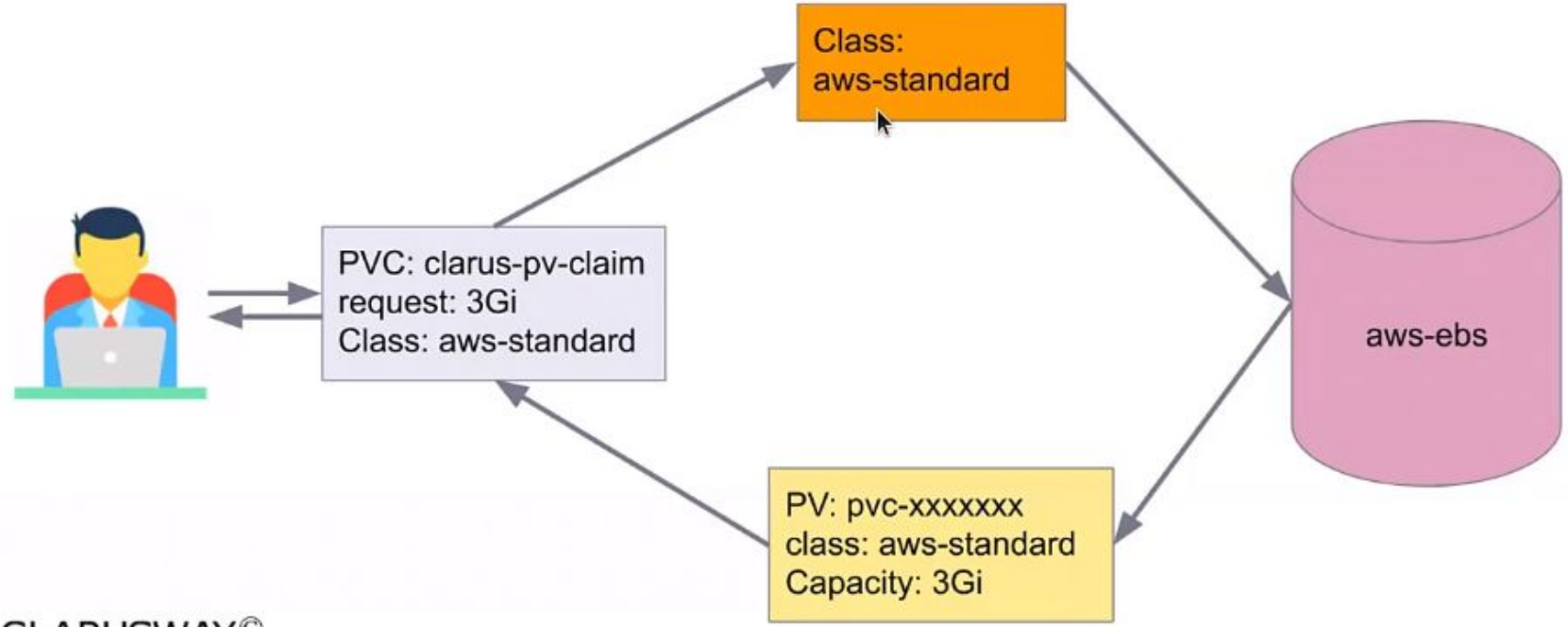
1. AWS CLI with Configured Credentials
2. kubectl installed
3. eksctl installed

Storage Class



Storage Class

A **StorageClass** provides a way for administrators to describe the "classes" of storage they offer.



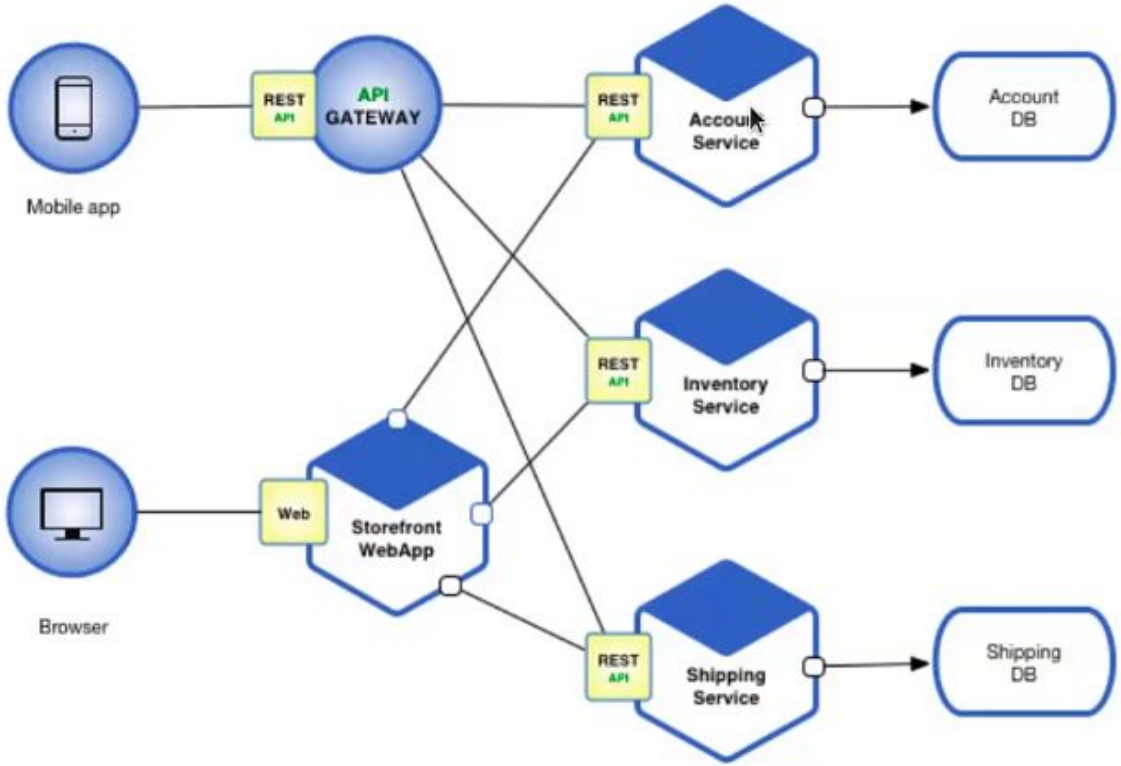
Storage Class

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: aws-standard
  annotations:
    storageclass.kubernetes.io/is-default-class:
"true"
provisioner: kubernetes.io/aws-ebs
parameters:
  type: gp2
  fsType: ext4
```

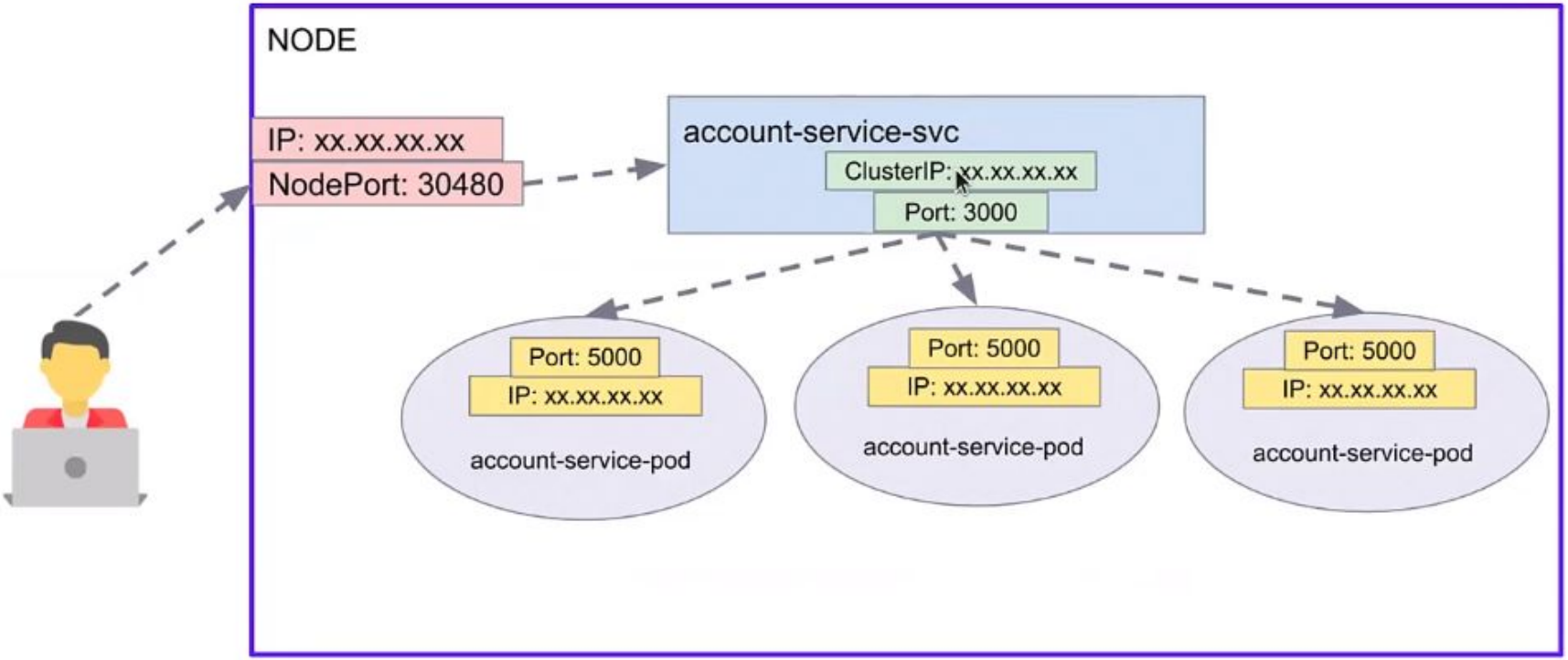
Provisioner: Each StorageClass has a provisioner that determines what volume plugin is used for provisioning PVs.

Parameters: Storage Classes have parameters that describe volumes belonging to the storage class. Different parameters may be accepted depending on the provisioner

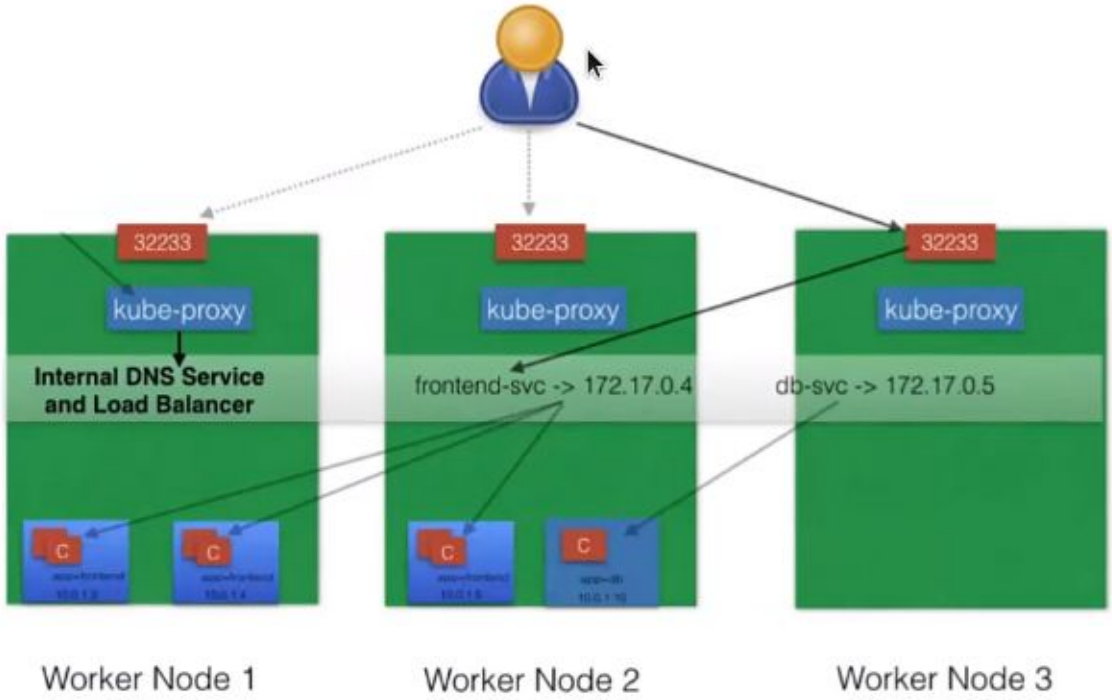
Ingress



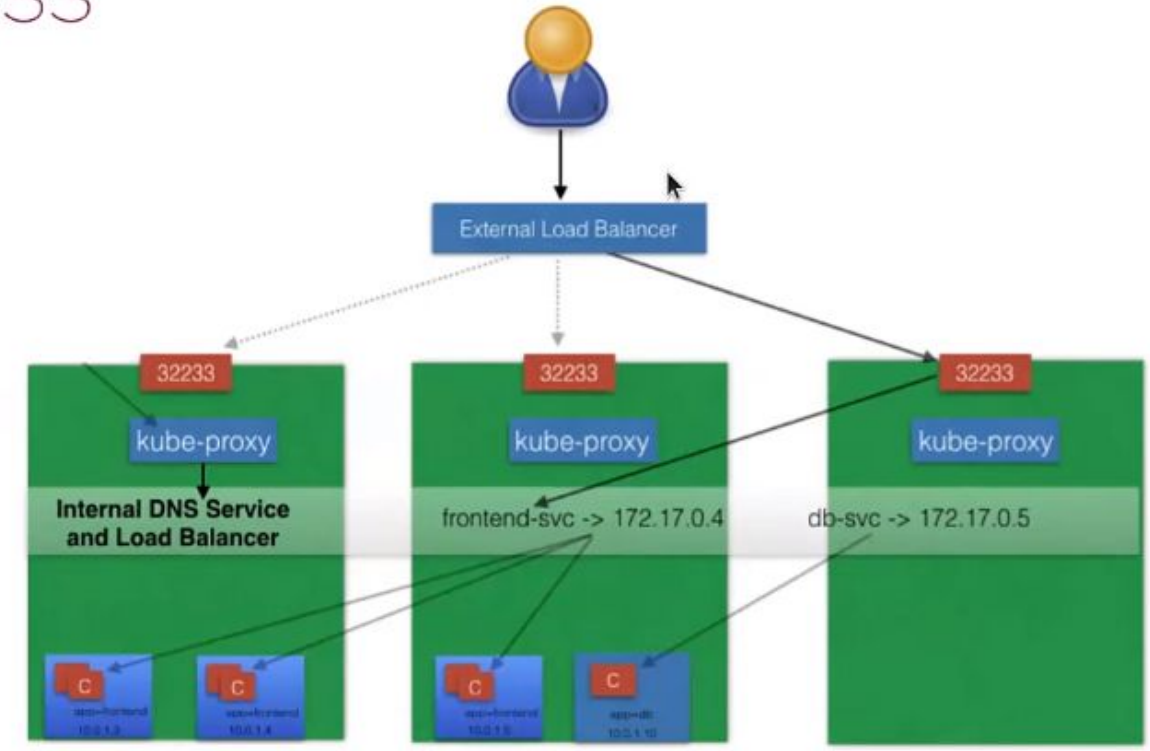
Ingress


NODE
IP: xx.xx.xx.xx
NodePort: 30480
account-service-svc
ClusterIP: xx.xx.xx.xx
Port: 3000
Port: 5000
IP: xx.xx.xx.xx
account-service-pod
Port: 5000
IP: xx.xx.xx.xx
account-service-pod
Port: 5000
IP: xx.xx.xx.xx
account-service-pod

Ingress



Ingress



Worker Node 1 Worker Node 2 Worker Node 3

Ingress

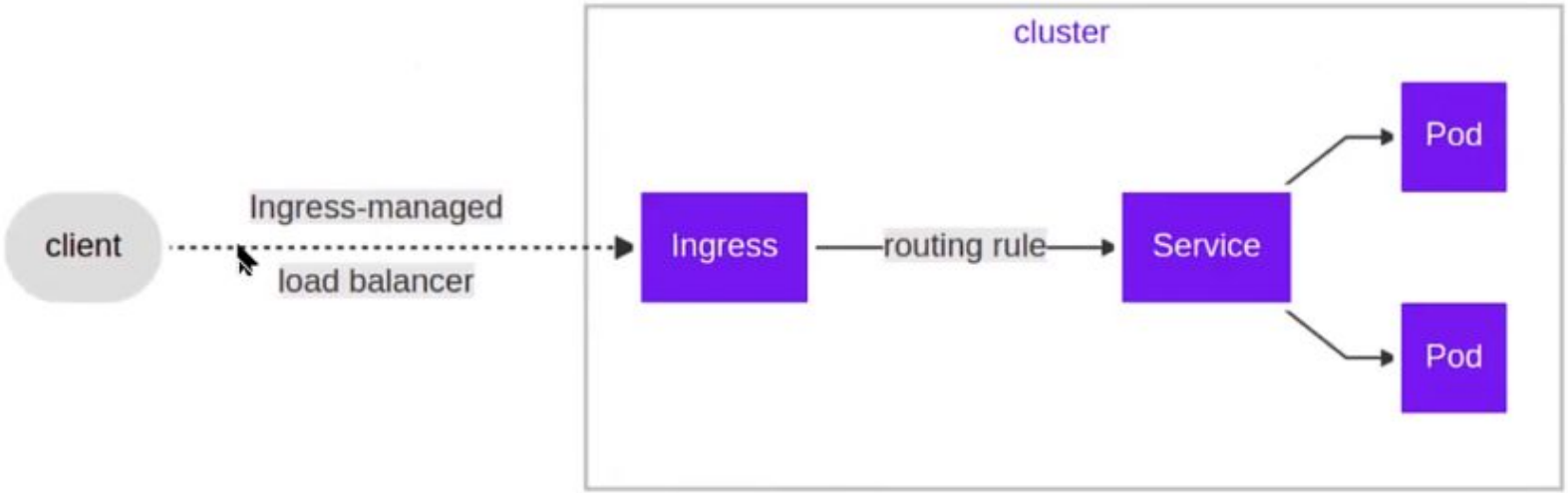
With Services, routing rules are associated with a given Service. They exist for as long as the Service exists, and there are many rules because there are many Services in the cluster. If we can somehow decouple the routing rules from the application and centralize the rules management, we can then update our application without worrying about its external access.

www.clarus-commerce.com

www.clarus-commerce.com/account
www.clarus-commerce.com/inventory
www.clarus-commerce.com/shipping

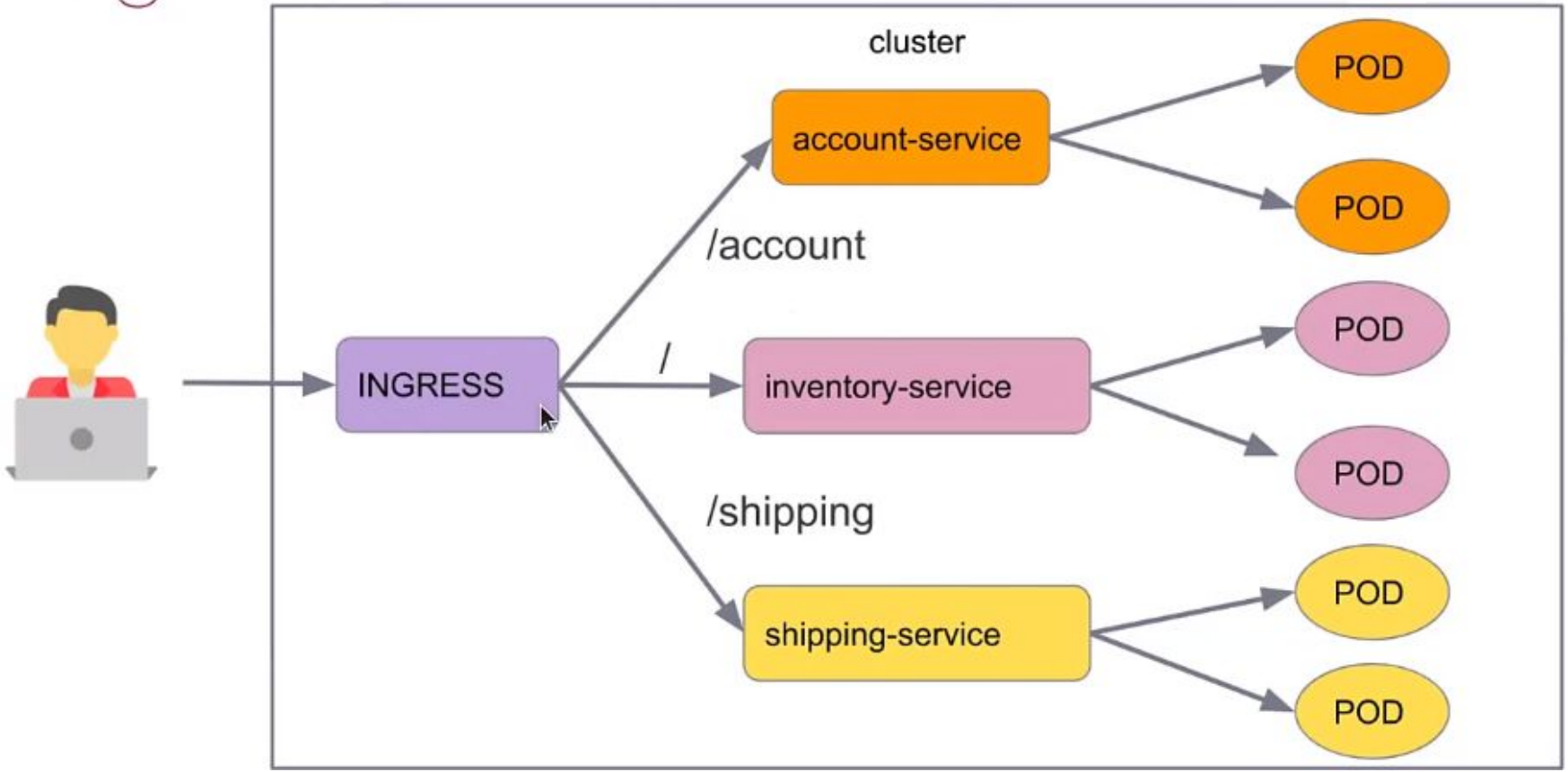
Ingress

"An Ingress is a collection of rules that allow inbound connections to reach the cluster Services."

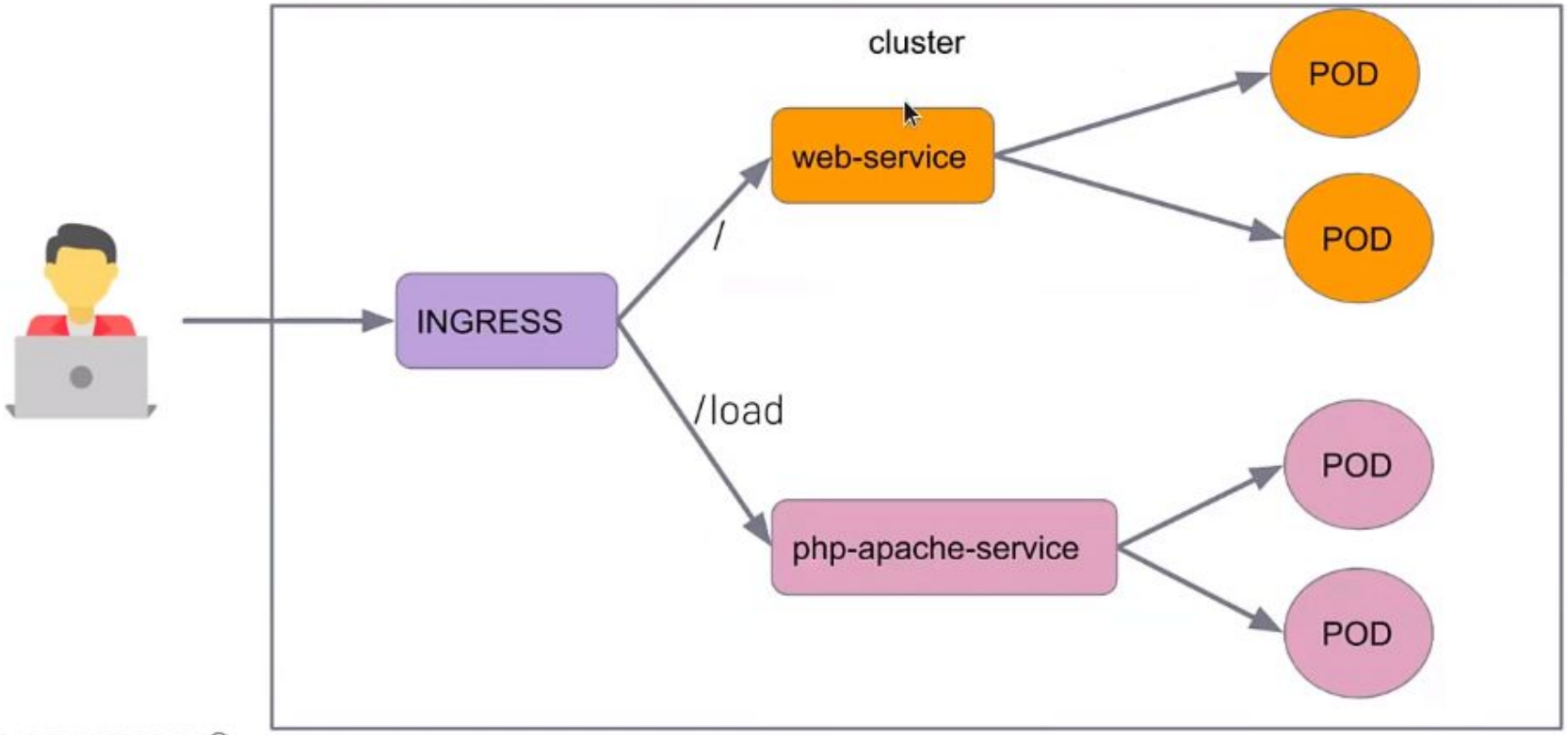


Ingress

www.clarus-commerce.com



Ingress



Ingress

With Ingress, users do not connect directly to a Service. Users reach the Ingress endpoint, and, from there, the request is forwarded to the desired Service.

```yaml
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
 name: ingress-service
 annotations:
   kubernetes.io/ingress.class: 'nginx'
   nginx.ingress.kubernetes.io/use-regex: 'true'
   nginx.ingress.kubernetes.io/rewrite-target: /$1
spec:
 rules:
   - http:
       paths:
         - path: /?(.*)
           backend:
             serviceName: web-service
             servicePort: 3000
         - path: /load/?(.*)
           backend:
             serviceName: php-apache-service
             servicePort: 80
```