“2007-2017” ECUG 十年高峰盛会

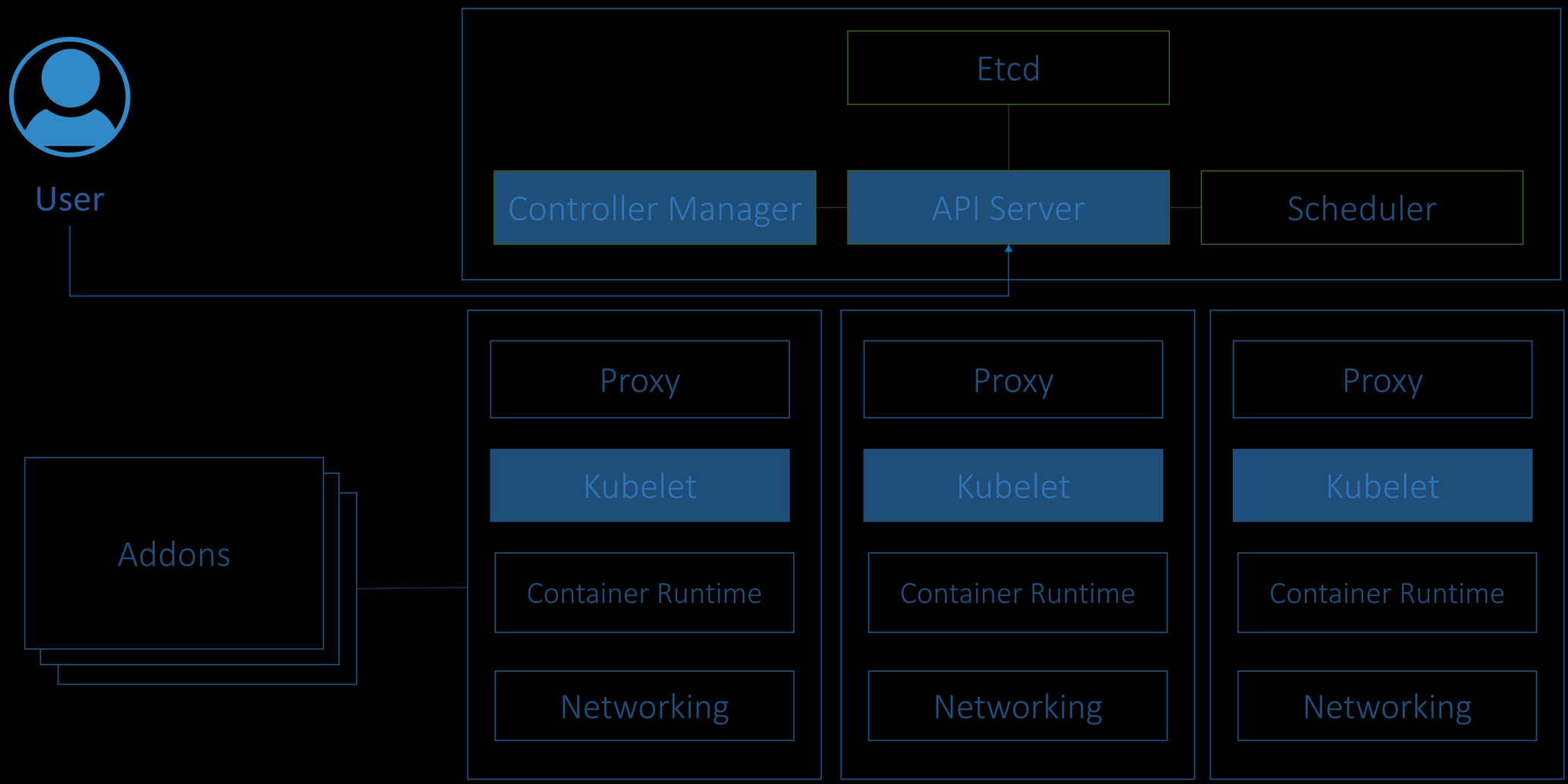
Kubernetes Cloud Provider 演进及 Azure 应用实践

倪朋飞
Microsoft Azure
Kubernetes Maintainer

目录

- Kubernetes 简介
- Cloud Provider 简介及演进
- Azure 应用实践

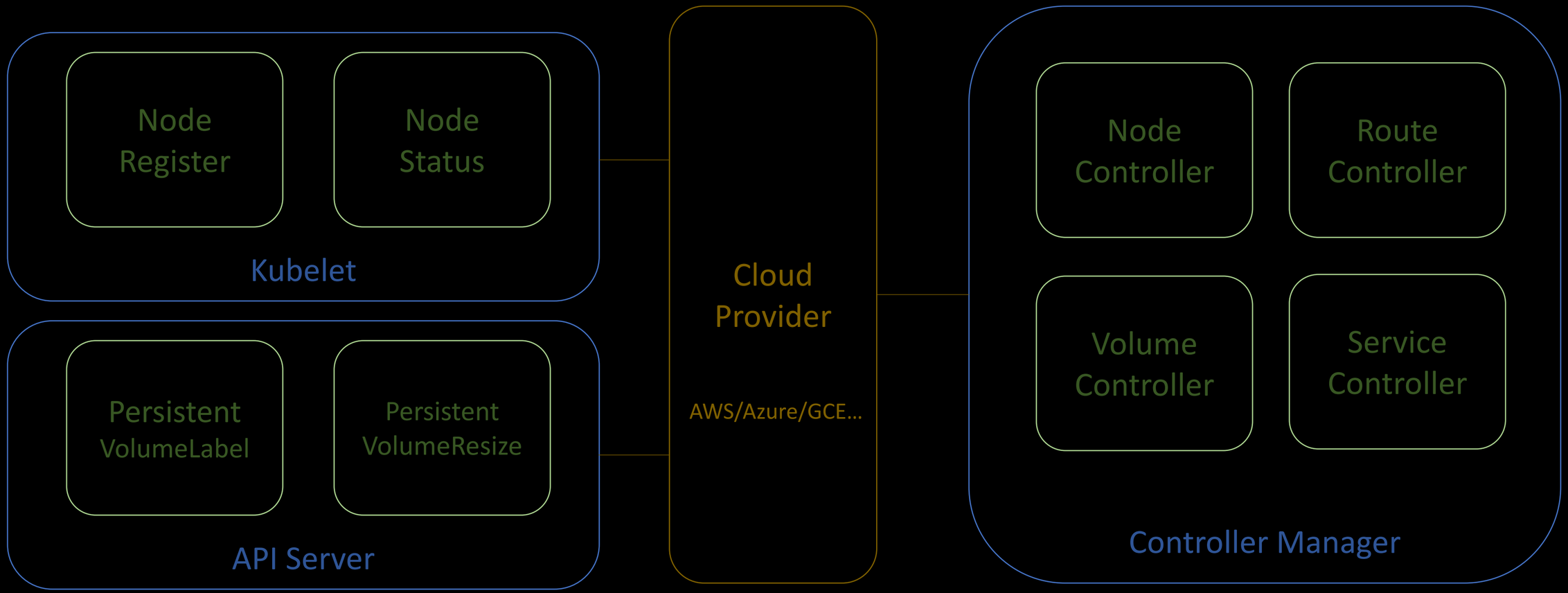
1. Kubernetes简介



2. Cloud Provider 简介及演进

- 云服务商通用接口
- 便于 Kubernetes 自动管理和利用云服务商提供的资源
  - 负载均衡、公网IP、路由
  - 持久化存储
  - Node VM 的状态管理
- 内置主流云服务商支持
  - 如 GCE、AWS、Azure、OpenStack等

Cloud Provider 架构



Cloud Provider 接口

- Instances
- LoadBalancer
- Clusters
- Zones
- Routes
- PersistentVolume

Instances 接口

```go
// Instances is an abstract, pluggable interface for sets of instances.
type Instances interface {
    // NodeAddresses returns the addresses of the specified instance.
    // TODO(roberthbailey): This currently is only used in such a way that it
    // returns the address of the calling instance. We should do a rename to
    // make this clearer.
    NodeAddresses(name types.NodeName) ([]v1.NodeAddress, error)
    // NodeAddressesByProviderID returns the addresses of the specified instance.
    // The instance is specified using the providerID of the node. The
    // ProviderID is a unique identifier of the node. This will not be called
    // from the node whose nodeaddresses are being queried. i.e. local metadata
    // services cannot be used in this method to obtain nodeaddresses
    NodeAddressesByProviderID(providerID string) ([]v1.NodeAddress, error)
    // ExternalID returns the cloud provider ID of the node with the specified NodeName.
    // Note that if the instance does not exist or is no longer running, we must return ("",
    cloudprovider.InstanceNotFound)
    ExternalID(nodeName types.NodeName) (string, error)
    // InstanceID returns the cloud provider ID of the node with the specified NodeName.
    InstanceID(nodeName types.NodeName) (string, error)
    // InstanceType returns the type of the specified instance.
    InstanceType(name types.NodeName) (string, error)
    // InstanceTypeByProviderID returns the type of the specified instance.
    InstanceTypeByProviderID(providerID string) (string, error)
    // AddSSHKeyToAllInstances adds an SSH public key as a legal identity for all instances
    // expected format for the key is standard ssh-keygen format: <protocol> <blob>
    AddSSHKeyToAllInstances(user string, keyData []byte) error
    // CurrentNodeName returns the name of the node we are currently running on
    // On most clouds (e.g. GCE) this is the hostname, so we provide the hostname
    CurrentNodeName(hostname string) (types.NodeName, error)
    // InstanceExistsByProviderID returns true if the instance for the given provider id still is
    running.
    // If false is returned with no error, the instance will be immediately deleted by the cloud
    controller manager.
    InstanceExistsByProviderID(providerID string) (bool, error)
}
```

LoadBalancer 接口

```go
// LoadBalancer is an abstract, pluggable interface for load balancers.
type LoadBalancer interface {
    // TODO: Break this up into different interfaces (LB, etc) when we have more than one type of
    service
    // GetLoadBalancer returns whether the specified load balancer exists, and
    // if so, what its status is.
    // Implementations must treat the *v1.Service parameter as read-only and not modify it.
    // Parameter 'clusterName' is the name of the cluster as presented to kube-controller-manager
    GetLoadBalancer(clusterName string, service *v1.Service) (status *v1.LoadBalancerStatus, exists
    bool, err error)
    // EnsureLoadBalancer creates a new load balancer 'name', or updates the existing one. Returns
    the status of the balancer
    // Implementations must treat the *v1.Service and *v1.Node
    // parameters as read-only and not modify them.
    // Parameter 'clusterName' is the name of the cluster as presented to kube-controller-manager
    EnsureLoadBalancer(clusterName string, service *v1.Service, nodes []*v1.Node)
    (*v1.LoadBalancerStatus, error)
    // UpdateLoadBalancer updates hosts under the specified load balancer.
    // Implementations must treat the *v1.Service and *v1.Node
    // parameters as read-only and not modify them.
    // Parameter 'clusterName' is the name of the cluster as presented to kube-controller-manager
    UpdateLoadBalancer(clusterName string, service *v1.Service, nodes []*v1.Node) error
    // EnsureLoadBalancerDeleted deletes the specified load balancer if it
    // exists, returning nil if the load balancer specified either didn't exist or
    // was successfully deleted.
    // This construction is useful because many cloud providers' load balancers
    // have multiple underlying components, meaning a Get could say that the LB
    // doesn't exist even if some part of it is still laying around.
    // Implementations must treat the *v1.Service parameter as read-only and not modify it.
    // Parameter 'clusterName' is the name of the cluster as presented to kube-controller-manager
    EnsureLoadBalancerDeleted(clusterName string, service *v1.Service) error
}
```

Cluster/Zones/Routes 接口

```go
// Clusters is an abstract, pluggable interface for clusters of containers.
type Clusters interface {
    // ListClusters lists the names of the available clusters.
    ListClusters() ([]string, error)
    // Master gets back the address (either DNS name or IP address) of the master node for the
    cluster.
    Master(clusterName string) (string, error)
}
```

```go
// Routes is an abstract, pluggable interface for advanced routing rules.
type Routes interface {
    // ListRoutes lists all managed routes that belong to the specified clusterName
    ListRoutes(clusterName string) ([]*Route, error)
    // CreateRoute creates the described managed route
    // route.Name will be ignored, although the cloud-provider may use nameHint
    // to create a more user-meaningful name.
    CreateRoute(clusterName string, nameHint string, route *Route) error
    // DeleteRoute deletes the specified managed route
    // Route should be as returned by ListRoutes
    DeleteRoute(clusterName string, route *Route) error
}
```

```go
// Zones is an abstract, pluggable interface for zone enumeration.
type Zones interface {
    // GetZone returns the Zone containing the current failure zone and locality region that the
    program is running in
    // In most cases, this method is called from the kubelet querying a local metadata service to
    aquire its zone.
    // For the case of external cloud providers, use GetZoneByProviderID or GetZoneByNodeName since
    GetZone
    // can no longer be called from the kubelets.
    GetZone() (Zone, error)

    // GetZoneByProviderID returns the Zone containing the current zone and locality region of the
    node specified by providerId
    // This method is particularly used in the context of external cloud providers where node
    initialization must be down
    // outside the kubelets.
    GetZoneByProviderID(providerID string) (Zone, error)

    // GetZoneByNodeName returns the Zone containing the current zone and locality region of the
    node specified by node name
    // This method is particularly used in the context of external cloud providers where node
    initialization must be down
    // outside the kubelets.
    GetZoneByNodeName(nodeName types.NodeName) (Zone, error)
}
```

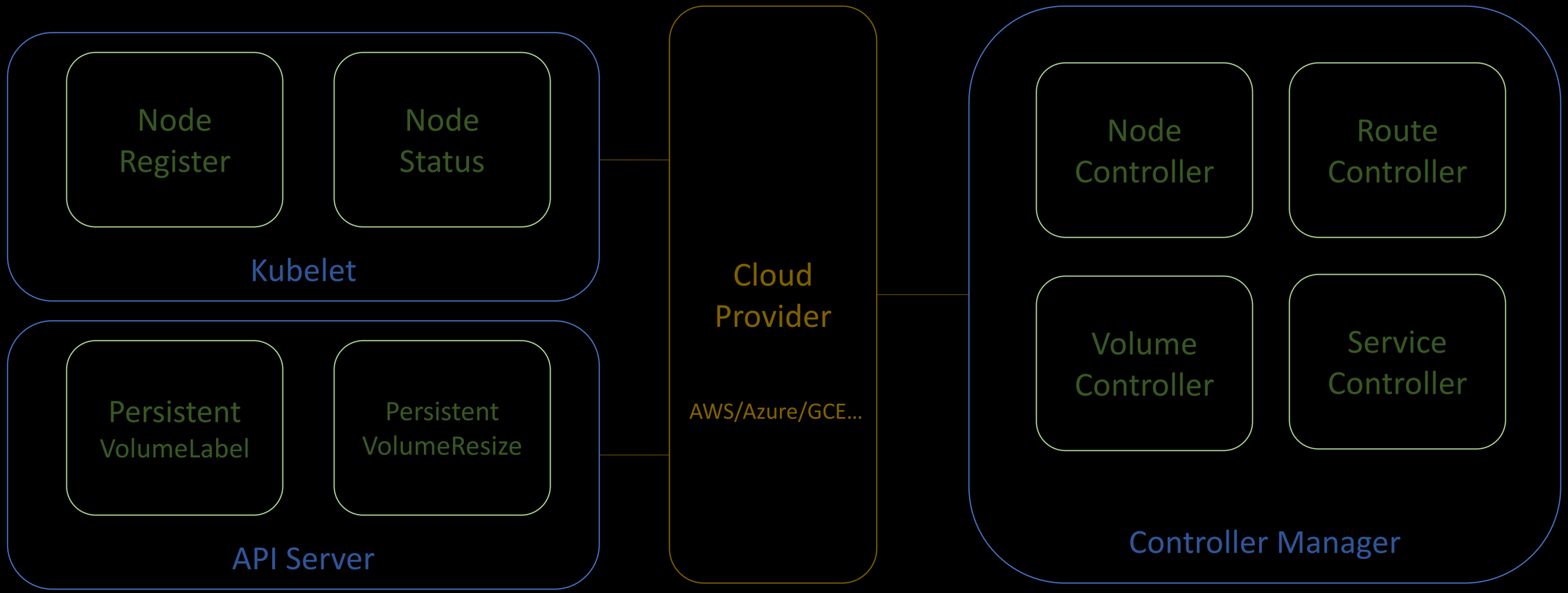
Persistent Volume 接口

```go
// interface exposed by the cloud provider implementing Disk functionlity
type DiskController interface {
    CreateBlobDisk(dataDiskName string, storageAccountType storage.SkuName, sizeGB int) (string, error)
    DeleteBlobDisk(diskUri string) error

    CreateManagedDisk(diskName string, storageAccountType storage.SkuName, sizeGB int, tags map[string]string) (string, error)
    DeleteManagedDisk(diskURI string) error

    // Attaches the disk to the host machine.
    AttachDisk(isManagedDisk bool, diskName, diskUri string, nodeName types.NodeName, lun int32, cachingMode compute.CachingTypes) error
    // Detaches the disk, identified by disk name or uri, from the host machine.
    DetachDiskByName(diskName, diskUri string, nodeName types.NodeName) error

    // Check if a list of volumes are attached to the node with the specified NodeName
    DisksAreAttached(diskNames []string, nodeName types.NodeName) (map[string]bool, error)

    // Get the LUN number of the disk that is attached to the host
    GetDiskLun(diskName, diskUri string, nodeName types.NodeName) (int32, error)
    // Get the next available LUN number to attach a new VHD
    GetNextDiskLun(nodeName types.NodeName) (int32, error)

    // Create a VHD blob
    CreateVolume(name, storageAccount, storageAccountType, location string, requestGB int) (string, string, int, error)
    // Delete a VHD blob
    DeleteVolume(diskURI string) error
}
```

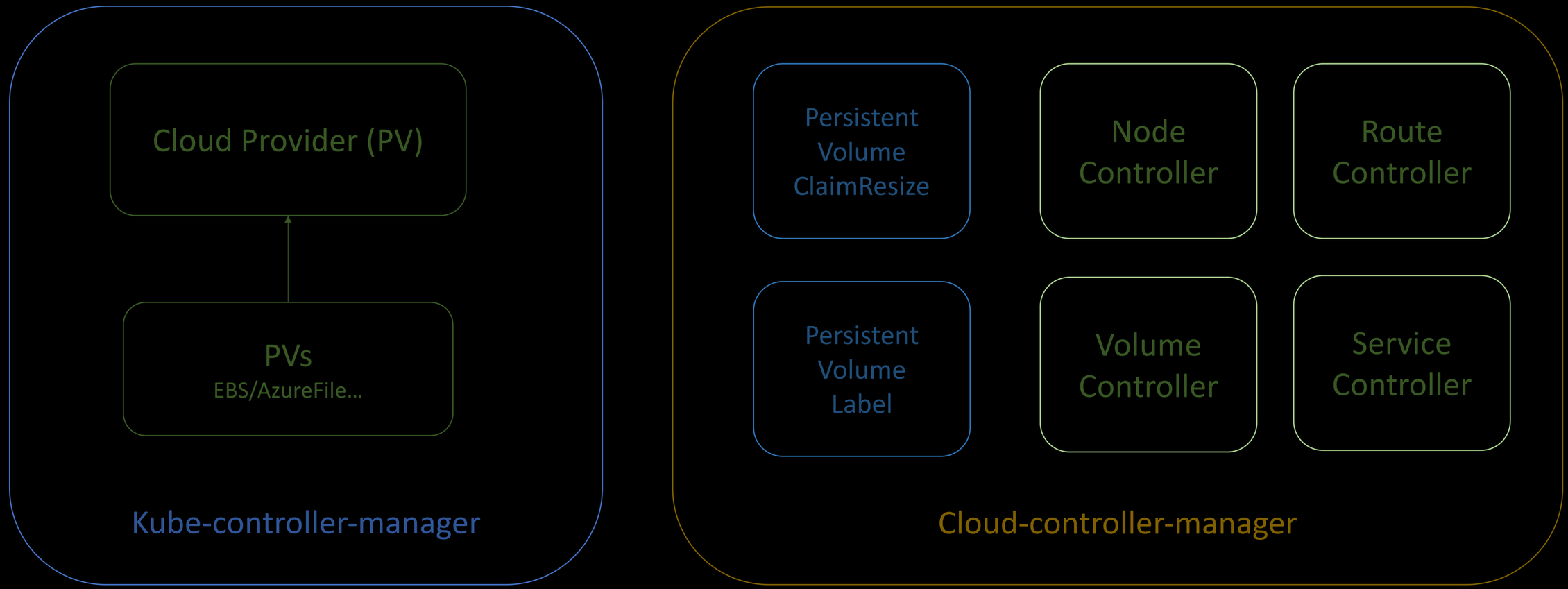
Cloud Provider 演进

- v0.1-Now: Cloud Provider interface定义
- v1.6-v1.9: 拆分 cloud-controller-manager（alpha），但依然推荐使用 kube-controller-manager 管理云服务
- v1.10+: 去掉 in-tree cloud provider，完全使用 cloud-controller-manager

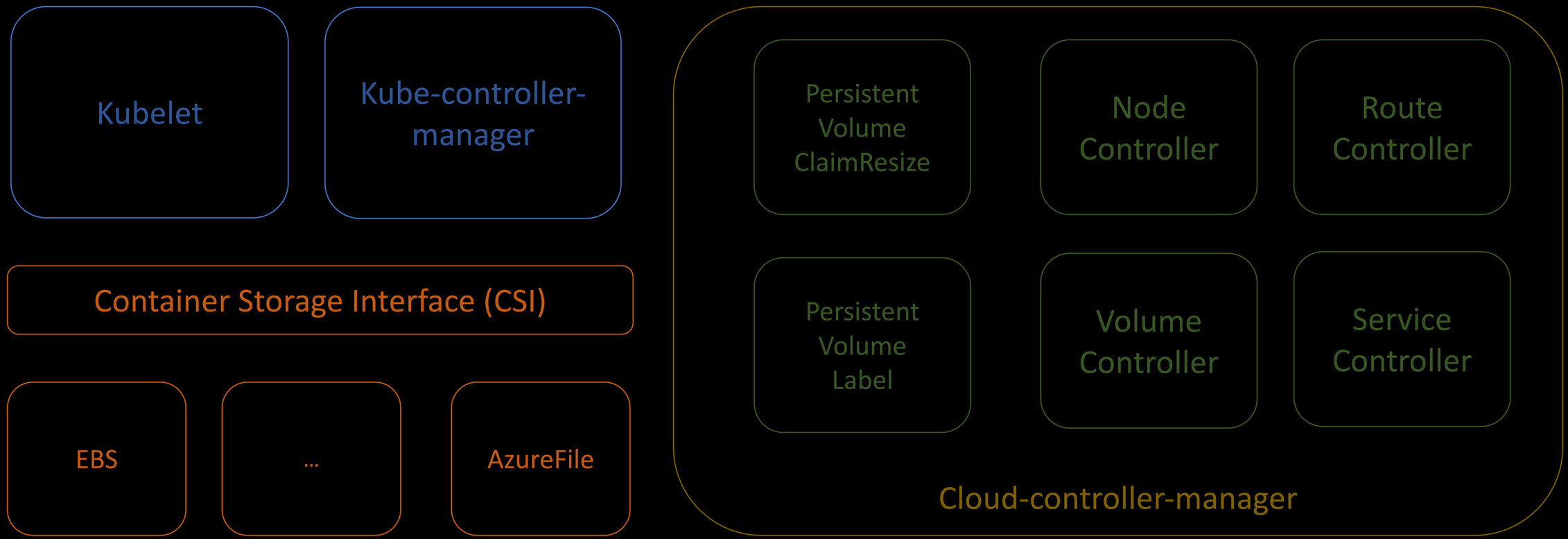
Cloud Provider 架构 - Now



Cloud Provider 架构 - Next



Cloud Provider 架构 – Future



4. Azure 实践

- Kubernetes v1.4 加入 Azure Cloud Provider
- 支持 Instances、LoadBalancer、Zones和 Routes 等所有接口
- 支持 AzureFile 和 AzureDisk 等持久化存储
- 支持集群扩展（需要下列服务支持）
- Azure 服务
  - Azure Container Service (AKS)
  - Azure Container Service Engine
  - Azure Container Service (ACS)

Azure cloud provider 配置

- kube-apiserver --cloud-provider=azure --cloud-config=/etc/kubernetes/azure.json
- kube-controller-manager --cloud-provider=azure --cloud-config=/etc/kubernetes/azure.json
- kubelet --cloud-provider=azure --cloud-config=/etc/kubernetes/azure.json

```
{
    "cloud": "AzurePublicCloud",
    "tenantId": "<tenant-id>",
    "subscriptionId": "<subspt-id>",
    "aadClientId": "<client-id>",
    "aadClientSecret": "<client-secret>",
    "resourceGroup": "<resource-group>",
    "location": "<location>",
    "subnetName": "<subnet>",
    "securityGroupName": "<security-group>",
    "vnetName": "<vnet-name>",
    "vnetResourceGroup": "<vnet-group>",
    "routeTableName": "<route-table-name>",
    "primaryAvailabilitySetName": "<primary-as-name>",
    "cloudProviderBackoff": false,
    "cloudProviderBackoffRetries": 0,
    "cloudProviderBackoffExponent": 0,
    "cloudProviderBackoffDuration": 0,
    "cloudProviderBackoffJitter": 0,
    "cloudProviderRatelimit": false,
    "cloudProviderRateLimitQPS": 0,
    "cloudProviderRateLimitBucket": 0,
    "useManagedIdentityExtension": false,
    "useInstanceMetadata": true
}
```

AzureFile

- Server Message Block (SMB/CIFS) 协议
- 支持跨主机共享
- Windows Server 还支持 Azure File Sync Cache

```yaml
apiVersion: v1
kind: Pod
metadata:
 name: azure
spec:
 containers:
  - image: kubernetes/pause
    name: azure
    volumeMounts:
      - name: azure
        mountPath: /mnt/azure
volumes:
      - name: azure
        azureFile:
          secretName: azure-secret
          shareName: myshare
          readOnly: false
```

```
# create storage account (for SSD change to Premium_LRS)
az storage account create -g mygroup -n mystore --sku Standard_LRS

# get storage account key
az storage account keys list -n mystore -g mygroup

# create storage share
az storage account show-connection-string -g mygroup -n mystore -o table
az storage share create -n myshare --account-name mystore --connection-string ”<my-connection-string>”

# create secret
kubectl create secret generic azure-secret --from-literal=azurestorageaccountname=mystore --from-literal=azurestorageaccountkey=”<my-store-key>"
```

AzureDisk

- Managed Disk
  - 自动管理存储账户
- Blob Disk
  - Shared： VHD共享存储账户
  - Dedicated： 每个VHD存储放在独立的存储账户
  - User specified： 用户指定存储账户
- AzureDisk 不支持共享

AzureDisk 示例

```yaml
# Manged SSD disk.
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
  name: managedssd
provisioner: kubernetes.io/azure-disk
parameters:
  storageaccounttype: Premium_LRS
  kind: Managed
```

```yaml
# Blob storage account specified
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
  name: accounthdd
provisioner: kubernetes.io/azure-disk
parameters:
  skuname: Standard_LRS
  location: westus
  storageAccount: azuretestx
```

```yaml
# Shared SSD blob disk.
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
  name: sharedssd
provisioner: kubernetes.io/azure-disk
parameters:
  skuname: Premium_LRS
  kind: Shared
```

```yaml
# Dedicated HDD blob disk.
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
  name: dedicatedhdd
provisioner: kubernetes.io/azure-disk
parameters:
  skuname: Standard_LRS
```

负载均衡

- 为 LoadBalancer Service 分配公网IP和负载均衡器
- 自定义负载均衡
  - 内网负载均衡 service.beta.kubernetes.io/azure-load-balancer-internal
  - DNS 标签名 service.beta.kubernetes.io/azure-dns-label-name
  - 负载均衡模式 service.beta.kubernetes.io/azure-load-balancer-mode
    - __auto__: 多个 Availability Sets时选择最少规则的负载均衡器
    - as1,as2: 显式指定负载均衡器所在的 Availability Set
    - 默认使用 Primary Availability Set 的负载均衡器
  - 共享安全组 service.beta.kubernetes.io/azure-shared-securityrule

七牛云 QINIU | IT大咖说 知识共享平台

Q&A