AZURE DAY

Demystifying the complexity of a Zone Redundant AKS deployment for achieving top class resiliency

Fabrizio Morando

Senior Lead, IT/Cloud Consulting @ Kyndryl

Riccardo Cozzi

Senior Lead, Infrastructure/Cloud Architect @ Kyndryl

Thanks to

Microsoft

avanade

eng

softwareone

4wardPRO
AN IMPRESOFT COMPANY

BCSoft
consulenza sistemi informativi

LOBRA

UNIKEY
Bringing IT knowledge to the people

Packt

JET BRAINS

NET CODE

AZURE DAY

Agenda

01 HA\DR Scenarios in Microsoft Azure

02 HA\Resiliency patterns for AKS

03 The Persistent Storage Dilemma

04 Deploying a reliable Workload

05 - …in an AKS cluster with a Zone Redundant Node Pool

06 - …in an AKS cluster with three Node Pools

07 Pros and Cons considerations in both approaches

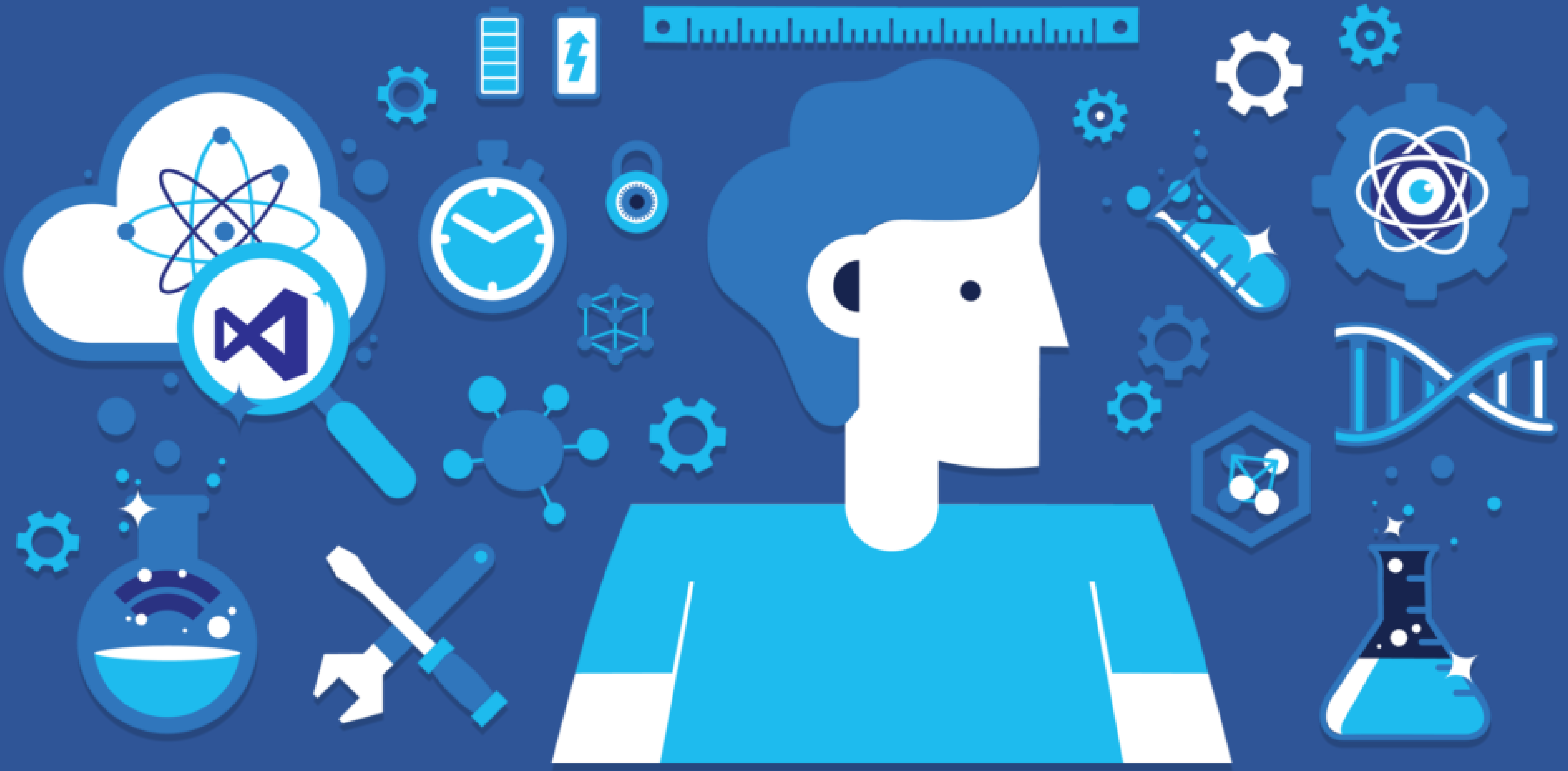
08 Demo

09 Q\A

10 Critical: vote our session as today’s best ☺

AZURE DAY

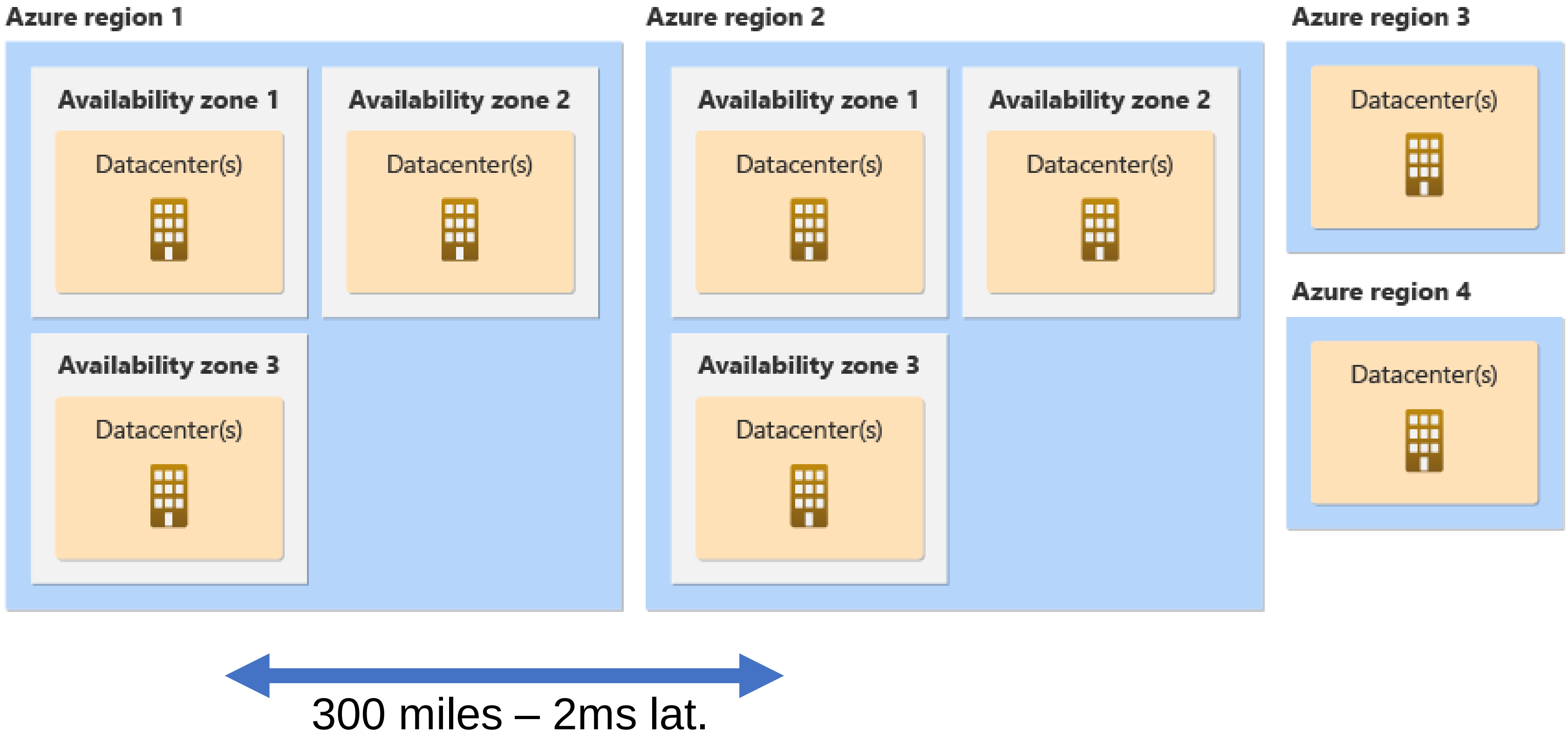
HA\DR Scenarios in Microsoft Azure



AZURE DAY

Azure Regions and Availability Zones

An Azure region is a geographic perimeter that contains a set of datacenters:



Azure region 1

Availability zone 1

Datacenter(s)

Availability zone 2

Datacenter(s)

Availability zone 3

Datacenter(s)

Azure region 2

Availability zone 1

Datacenter(s)

Availability zone 2

Datacenter(s)

Availability zone 3

Datacenter(s)

Azure region 3

Datacenter(s)

Azure region 4

Datacenter(s)

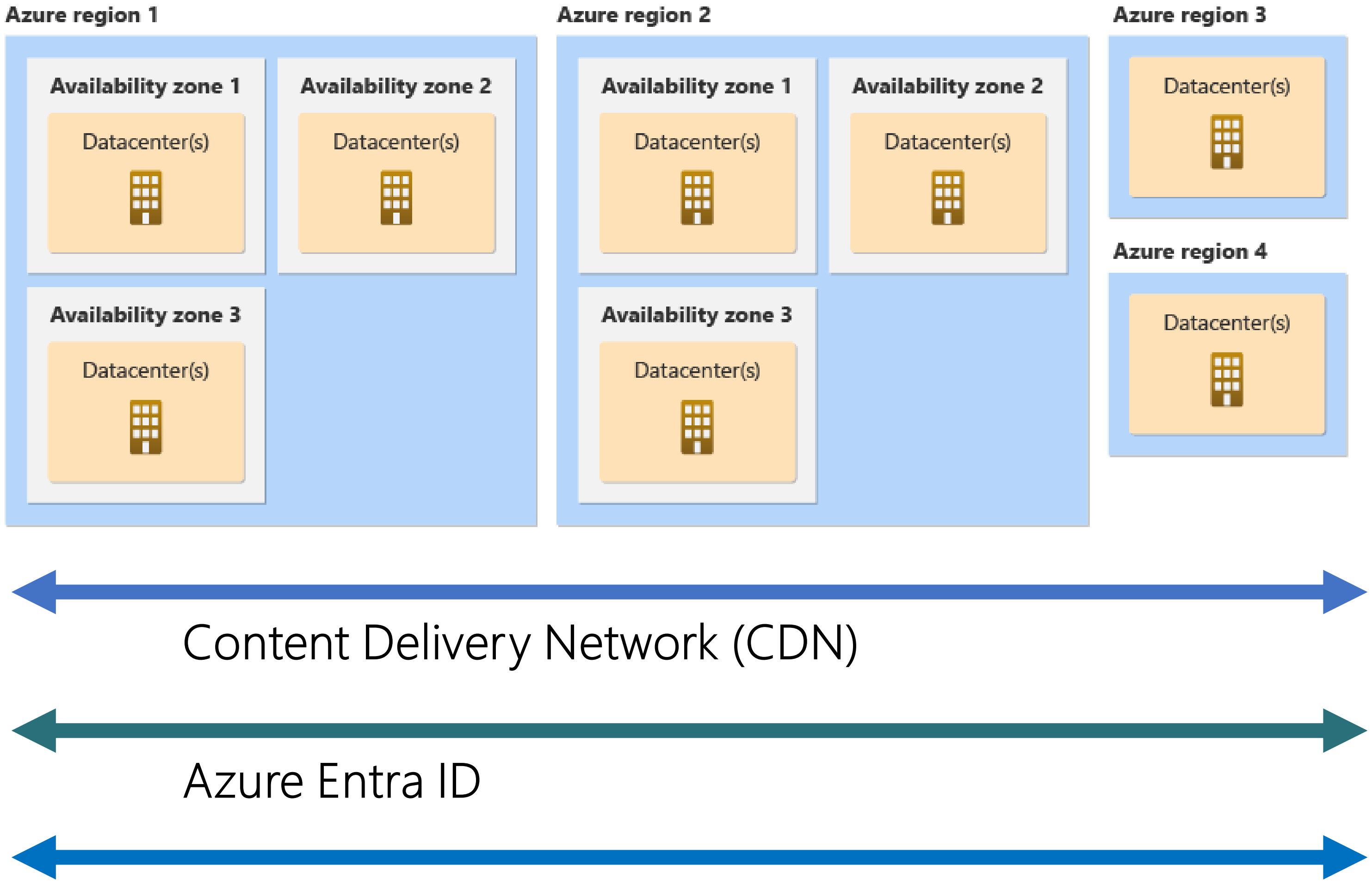
300 miles – 2ms lat.

Many Azure regions provide availability zones, which are separated groups of datacenters. Many regions also have a paired region. Paired regions support certain types of multi-region deployment approaches

AZURE DAY

Azure Global Services

An Azure global service refers to a network infrastructure that spans across multiple regions and data centers



Azure region 1

Availability zone 1 – Datacenter(s)

Availability zone 2 – Datacenter(s)

Availability zone 3 – Datacenter(s)

Azure region 2

Availability zone 1 – Datacenter(s)

Availability zone 2 – Datacenter(s)

Availability zone 3 – Datacenter(s)

Azure region 3

Datacenter(s)

Azure region 4

Datacenter(s)

Content Delivery Network (CDN)

Azure Entra ID

Azure Front-Door \ Azure Traffic Manager

AZ pinned and spread resources

Azure services support one or both of following infrastructure:

Zonal

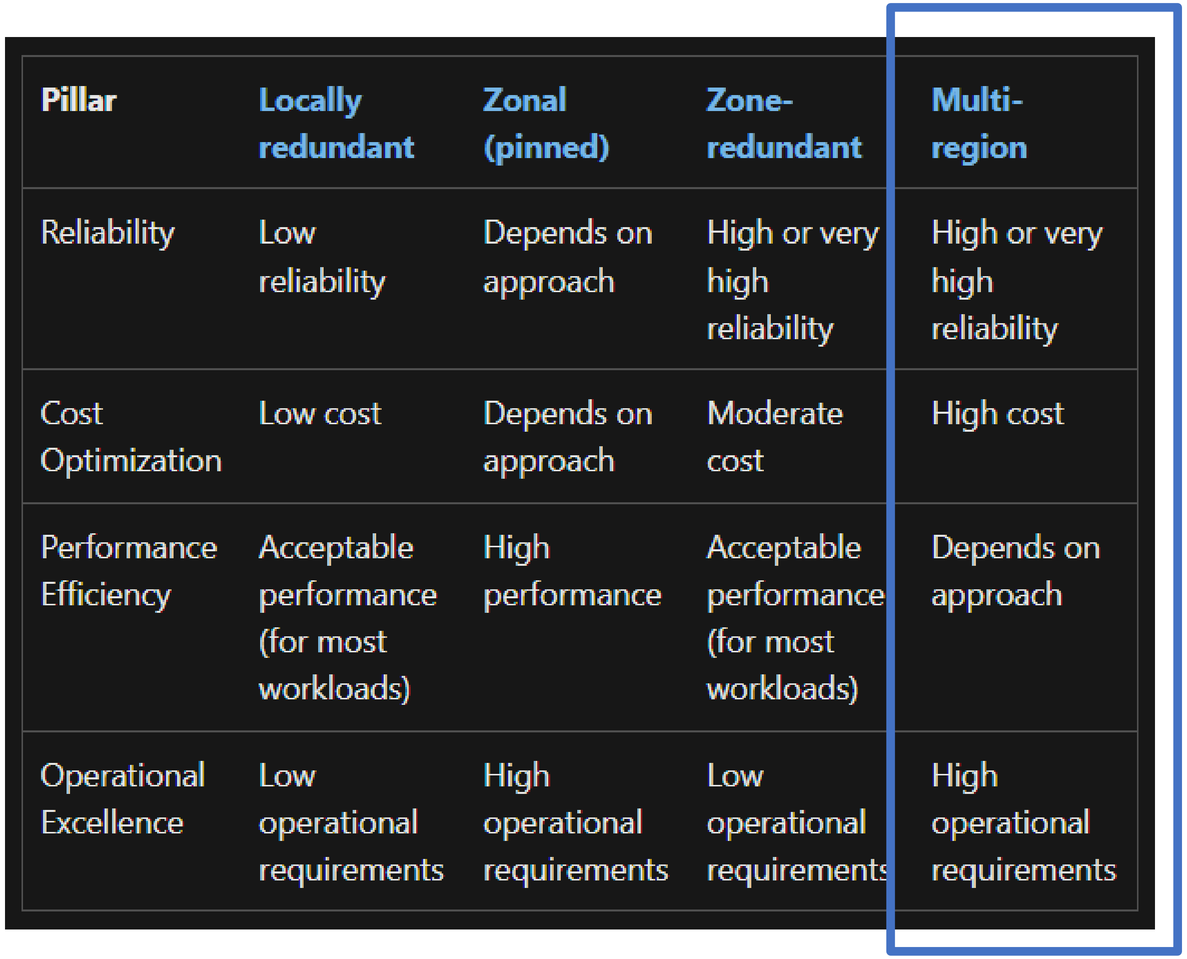
- Typical IaaS\VMs scenario. If an outage occurs in a single availability zone, you're responsible for failover to another availability zone.

Zone-redundant

- Typical PaaS scenario. If an outage occurs in a single availability zone, Microsoft manages failover automatically.

Deployment approaches

There are multiple ways to deploy a solution in MS Azure



Pillar	Locally redundant	Zonal (pinned)	Zone-redundant	Multi-region
Reliability	Low reliability	Depends on approach	High or very high reliability	High or very high reliability
Cost Optimization	Low cost	Depends on approach	Moderate cost	High cost
Performance Efficiency	Acceptable performance (for most workloads)	High performance	Acceptable performance (for most workloads)	Depends on approach
Operational Excellence	Low operational requirements	High operational requirements	Low operational requirements	High operational requirements

Usually when dealing with a cloud native application, a Disater Recovery strategy begins to make sense only in this scenario

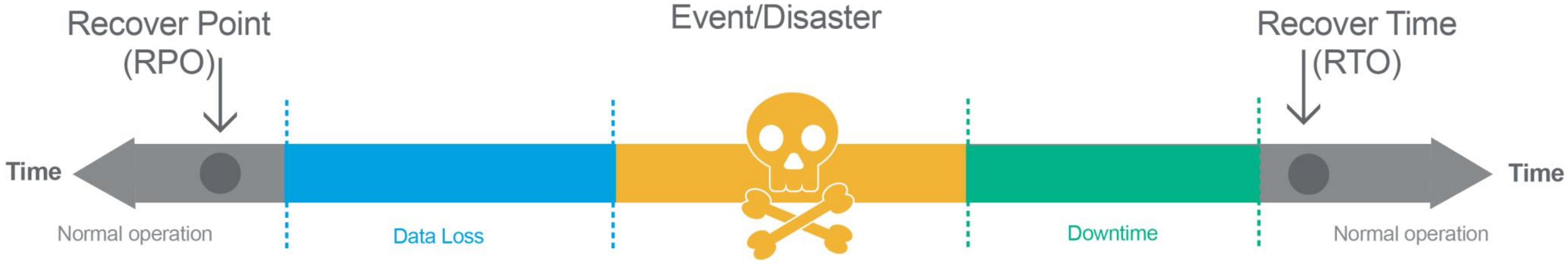
Key “Must Know”: Your Resiliency requirements

It's important to understand the resiliency requirements for your workload

How much data can you afford to recreate or lose?

RPO vs RTO

How quickly must you recover?
What is the cost of downtime?



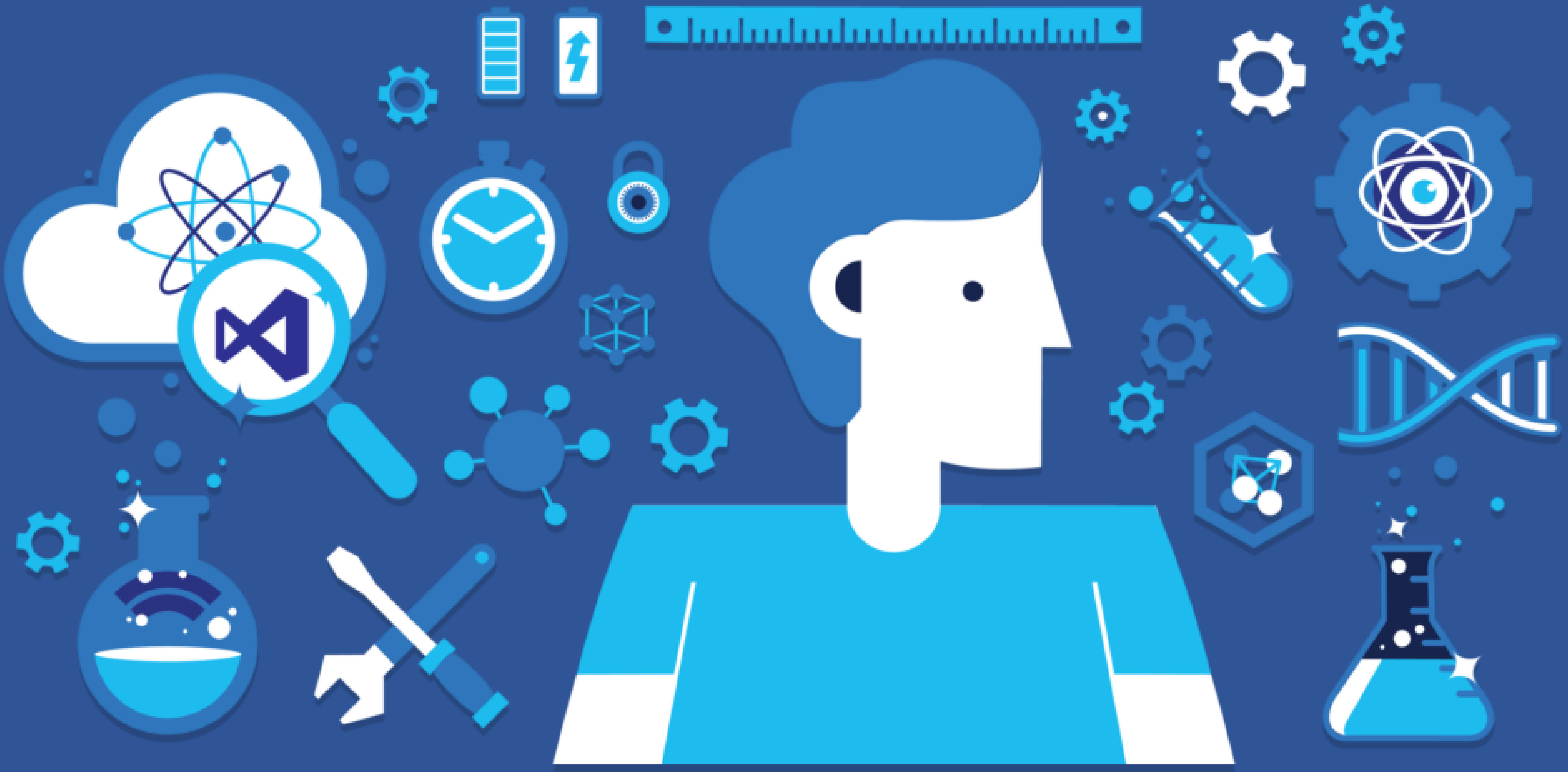
AZURE DAY

Single region resiliency approach

- In this speech, **We will focus exclusively on Azure resiliency within a single region,** utilizing redundancy through Availability Zones to ensure high availability and fault tolerance
- In a Cloud Native scenario, where the SLA target is guaranteed only through zone redundancy, and there are no recovery plans based on multi-region architectures, **most of the matters about failovers between data centers are entirely guaranteed by the hyperscaler** (while ensuring appropriate tier usage for each resource type)
- <u>Our responsibilities is to adopt best practices and guidance in order to leverage cloud High Availability features at their best</u>

AZURE DAY

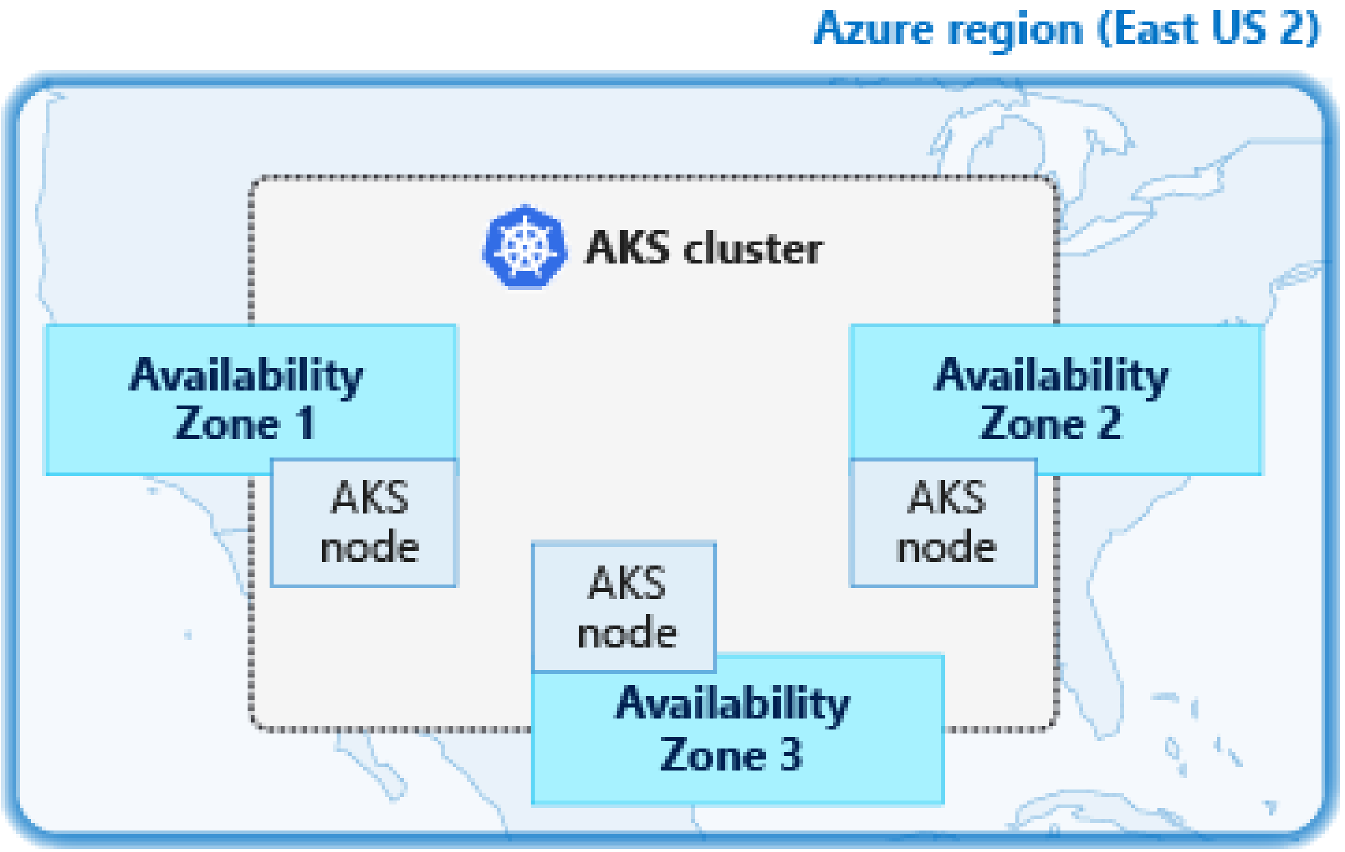
HA\Resiliency patterns for AKS



How AKS leverages Availability Zones

AKS clusters deployed using availability zones **can distribute nodes across multiple zones** within a single region.

A cluster in the East US 2 region can create nodes in all three availability zones in East US 2. **This distribution of AKS cluster resources improves cluster availability** as they're resilient to failure of a specific zone.

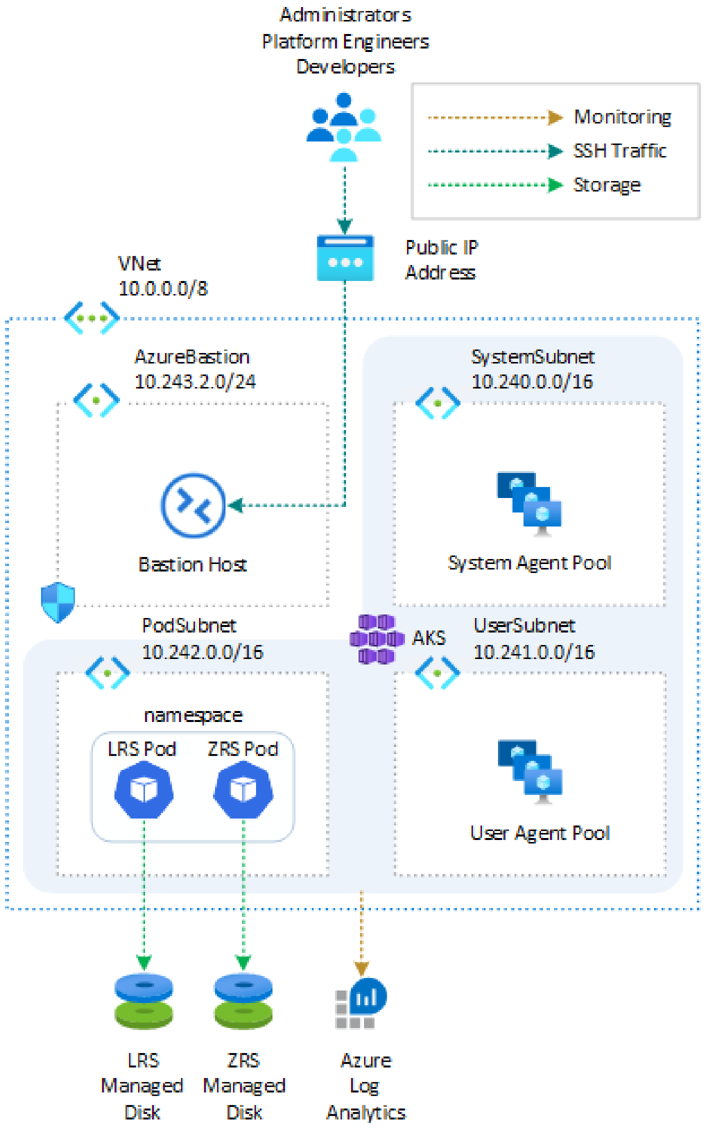


AZURE DAY

Creating Zone Redundant AKS Clusters

Here are two approaches to creating a zone redundant AKS cluster:

- Zone Redundant Node Pool: This approach involves creating a zone redundant node pool, where nodes are spread across multiple Availability Zones. This ensures that the node pool can withstand failures in any zone while maintaining the desired functionality.

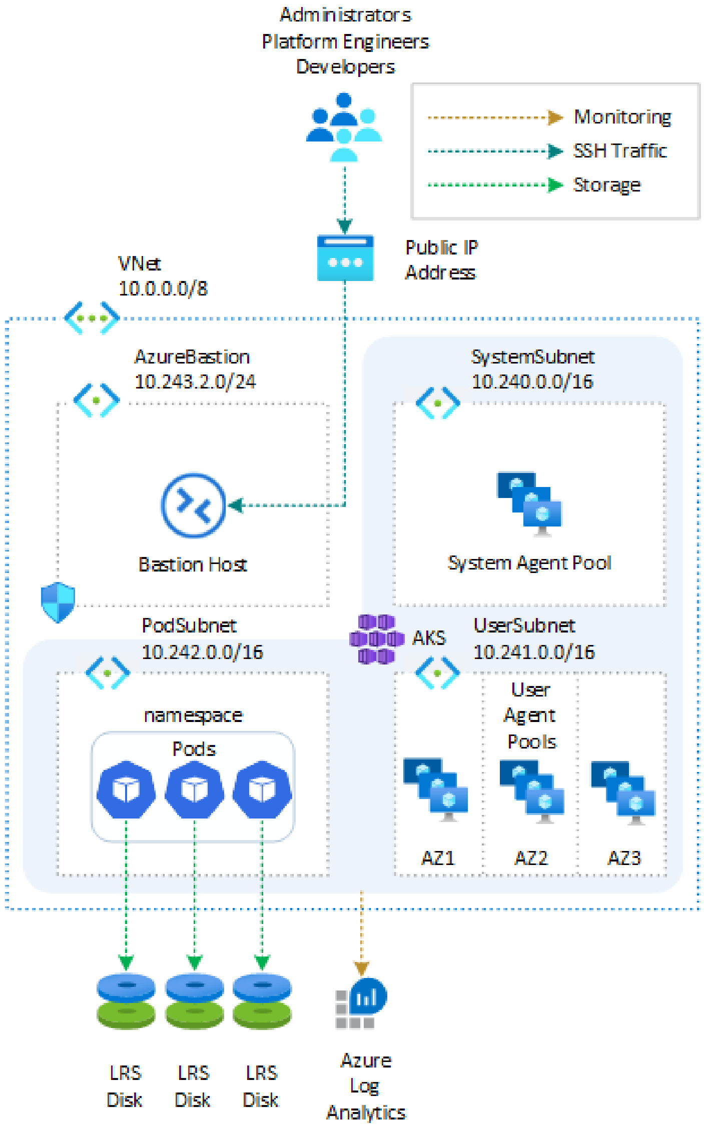


Creating Zone Redundant AKS Clusters

There are two approaches to creating a zone redundant AKS cluster:

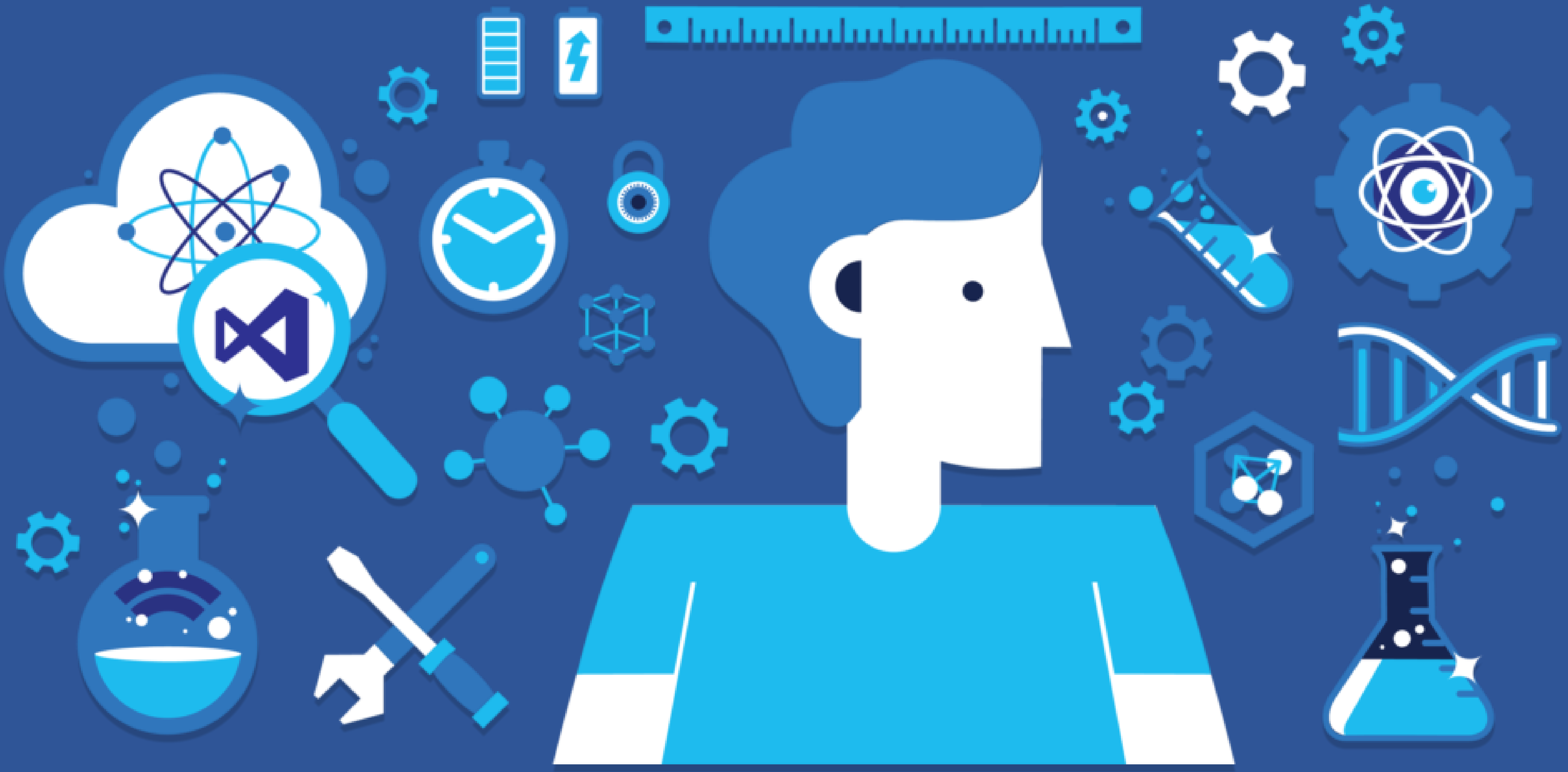
- **AKS Cluster with one Zone Redundant Node Pool**: This approach involves creating a zone redundant node pool, where nodes are spread across multiple Availability Zones. This ensures that the node pool can withstand failures in any zone while maintaining the desired functionality.

- **AKS Cluster with three Node Pools**: In this approach, an AKS cluster is created with three node pools, each assigned to a different availability zone. This ensures that the cluster has redundancy across zones.



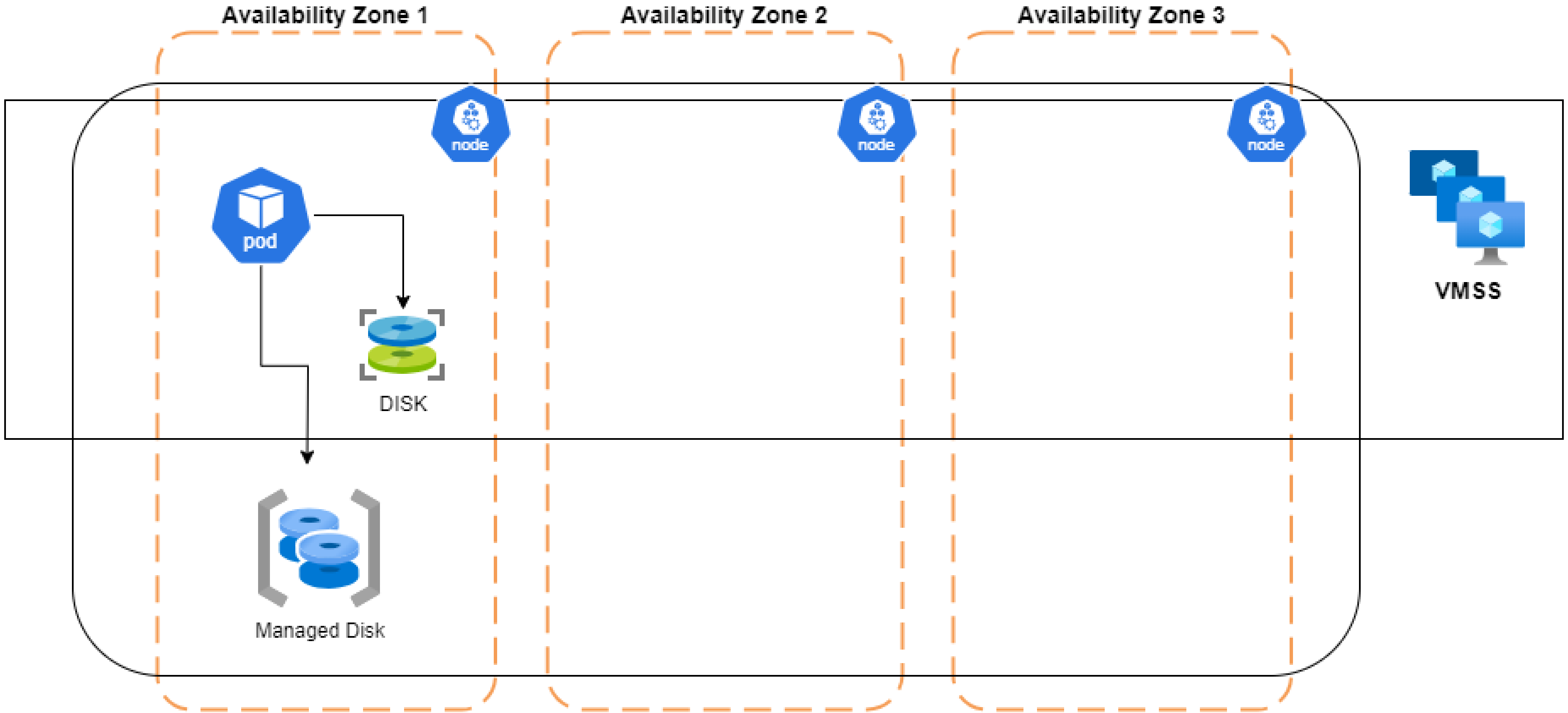
AZURE DAY

The Persistent Storage Dilemma



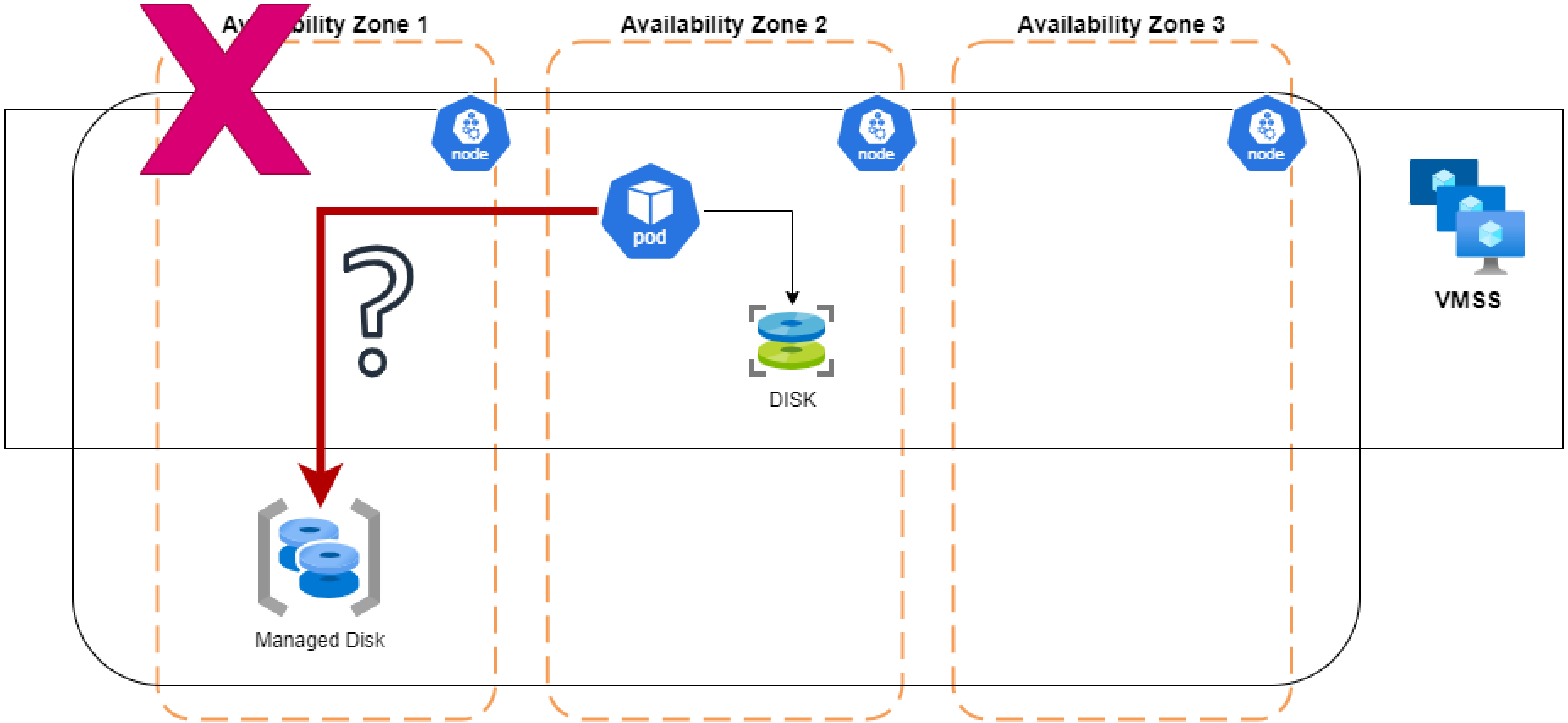
AZURE DAY

The Persistent Storage Dilemma



AZURE DAY

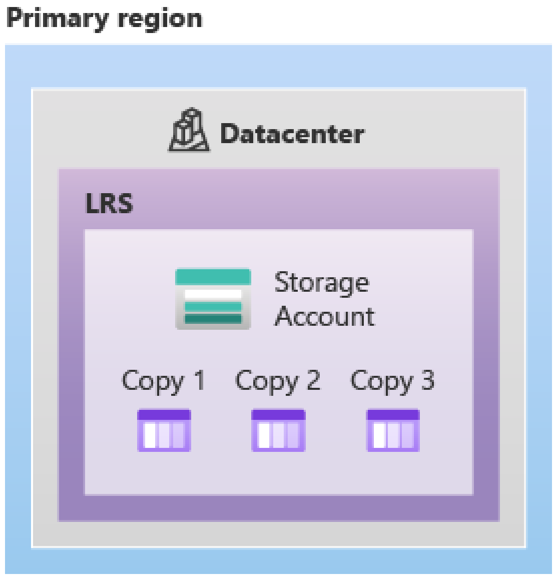
The Persistent Storage Dilemma



AZURE DAY

Azure Storage Redundancy

Data in Azure Storage is always replicated three times in the primary region. Azure Storage offers two options for how your data is replicated in the primary region: locally redundant storage (LRS) and zone-redundant storage (ZRS)



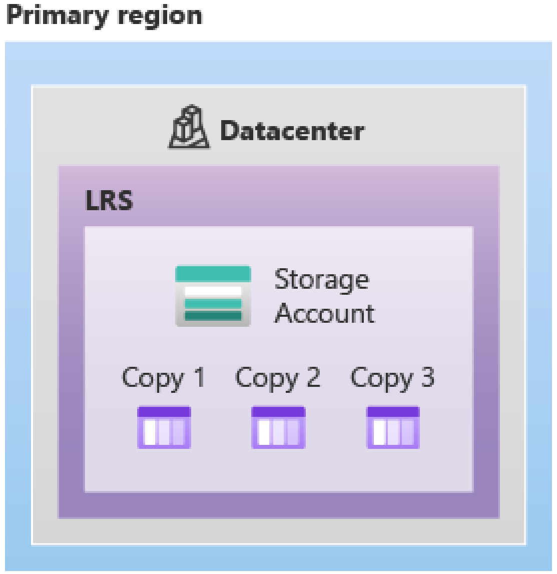
LRS is the *lowest-cost* redundancy option and offers the least *durability* compared to other options.

LRS protects your data against server rack and drive failures

AZURE DAY

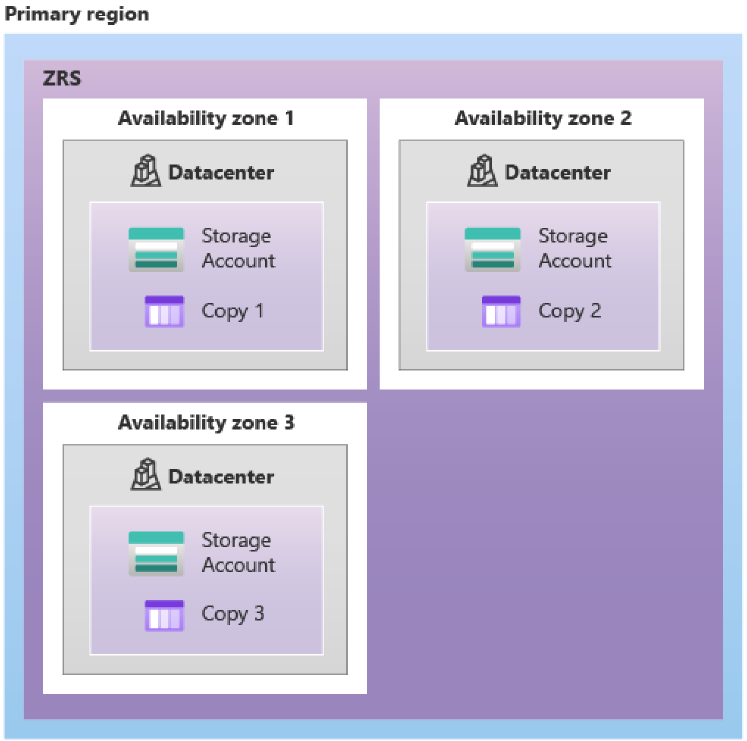
Azure Storage Redundancy

Data in Azure Storage is always replicated three times in the primary region. Azure Storage offers two options for how your data is replicated in the primary region: locally redundant storage (LRS) and zone-redundant storage (ZRS)



ZRS has higher costs.

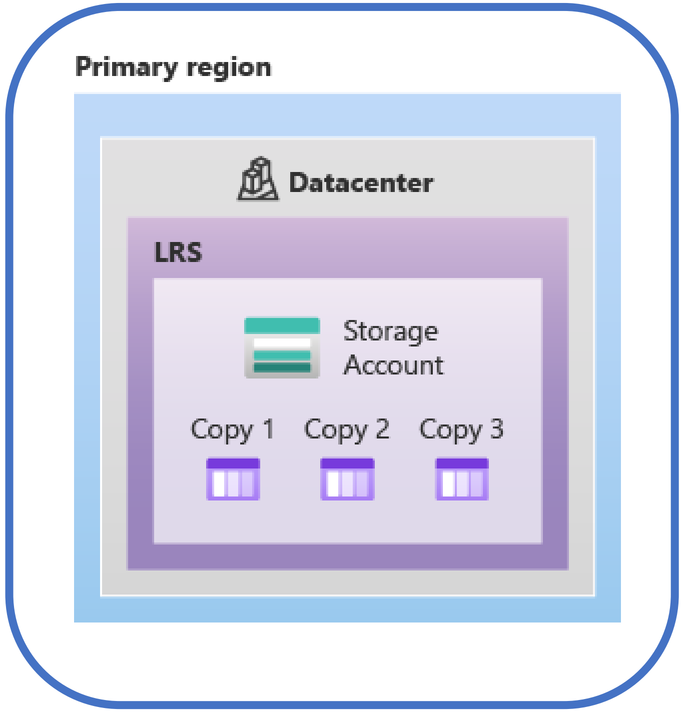
However, it provides *excellent performance, low latency, and resiliency* for your data if it becomes temporarily unavailable.



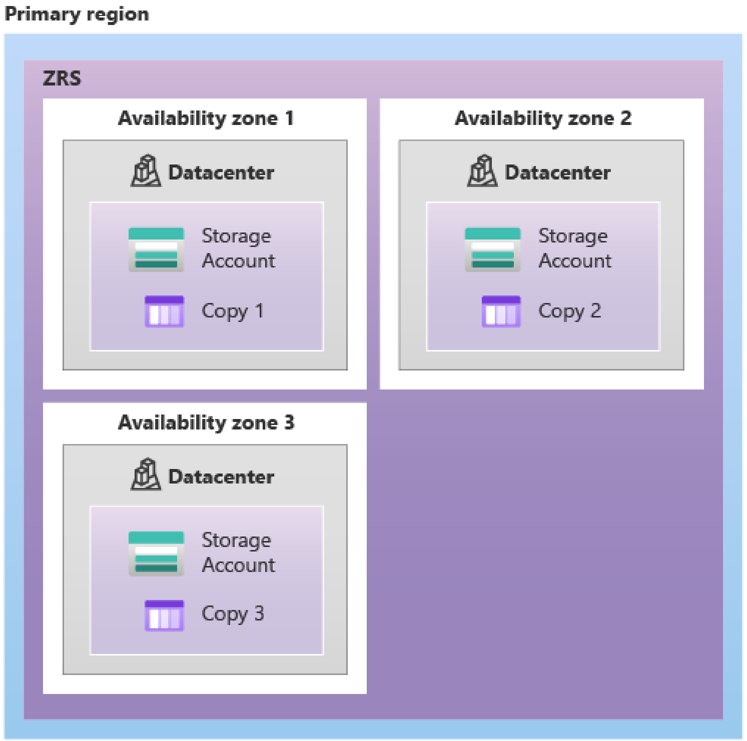
AZURE DAY

Azure Storage Redundancy

Data in Azure Storage is always replicated three times in the primary region. Azure Storage offers two options for how your data is replicated in the primary region: locally redundant storage (LRS) and zone-redundant storage (ZRS)



LRS is the redundancy model used by the built-in storage classes in Azure Kubernetes Service (AKS), such as managed-csi and managed-csi-premium.



AKS Storage Classes

The Azure Disks Container Storage Interface (CSI) driver is a CSI specification-compliant driver used by Azure Kubernetes Service (AKS) to manage the lifecycle of Azure Disk. **These services enable simplified integration with Azure Disk Storage, improving the efficiency and management of persistent volumes in AKS, even in automation contexts**

When you use the Azure Disk CSI driver on AKS, there are two built-in StorageClasses that use the Azure Disk CSI storage driver.

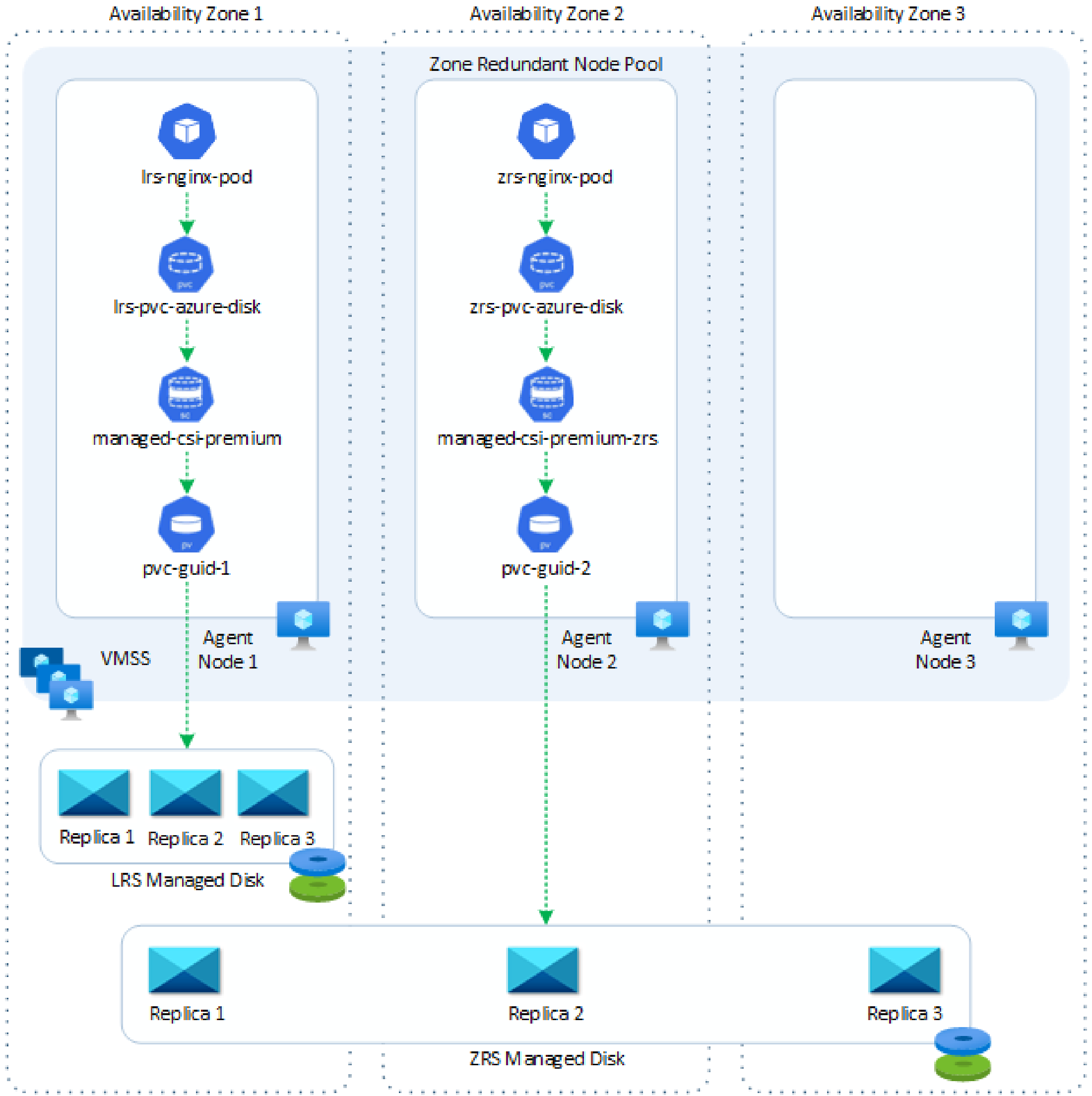
- **managed-csi:** Uses Azure Standard SSD locally redundant storage (LRS) to create a managed disk.
- **managed-csi-premium:** Uses Azure Premium LRS to create a managed disk

+ 2 equivalent for Azure Files.

These storage classes cannot be used by default for 1st AKS deployment strategy DIRECTLY.

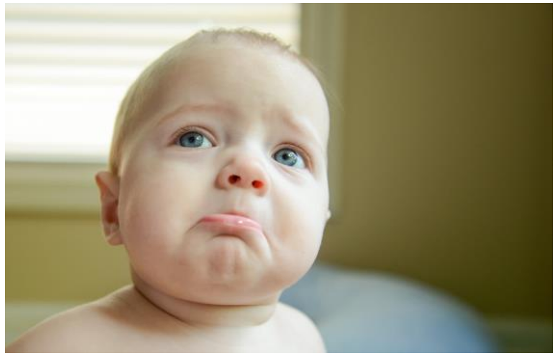
AZURE DAY

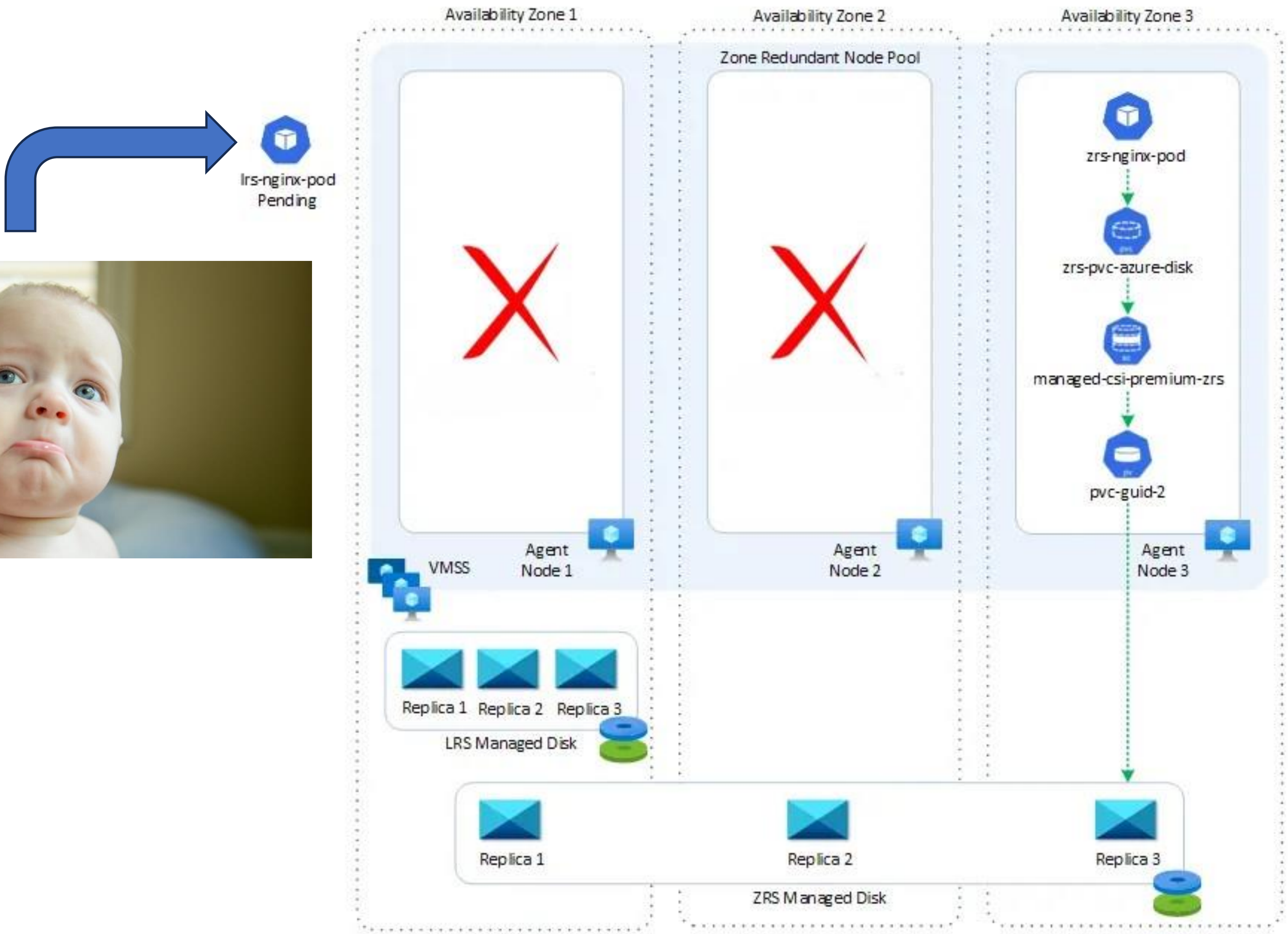
Pods distribution in a ZR node pool



AZURE DAY

Pods distribution in a ZR node pool





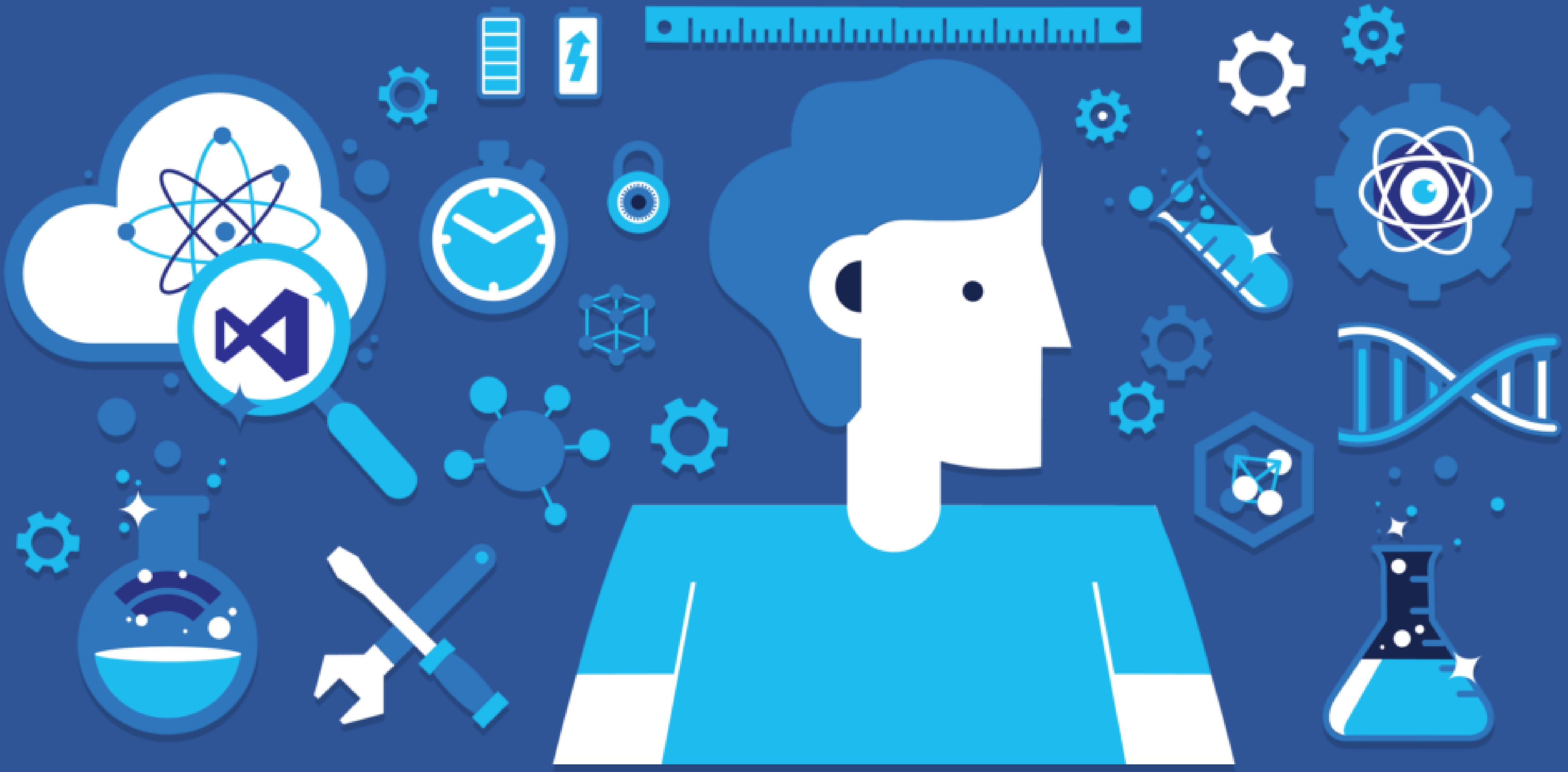
Pods distribution in a ZR node pool

To create a custom storage class using StandardSSD_ZRS or Premium_ZRS managed disks, you can use the following example:

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
    name: managed-csi-premium-zrs
provisioner: disk.csi.azure.com
parameters:
    skuname: Premium_ZRS
reclaimPolicy: Delete
volumeBindingMode: WaitForFirstConsumer
allowVolumeExpansion: true
```

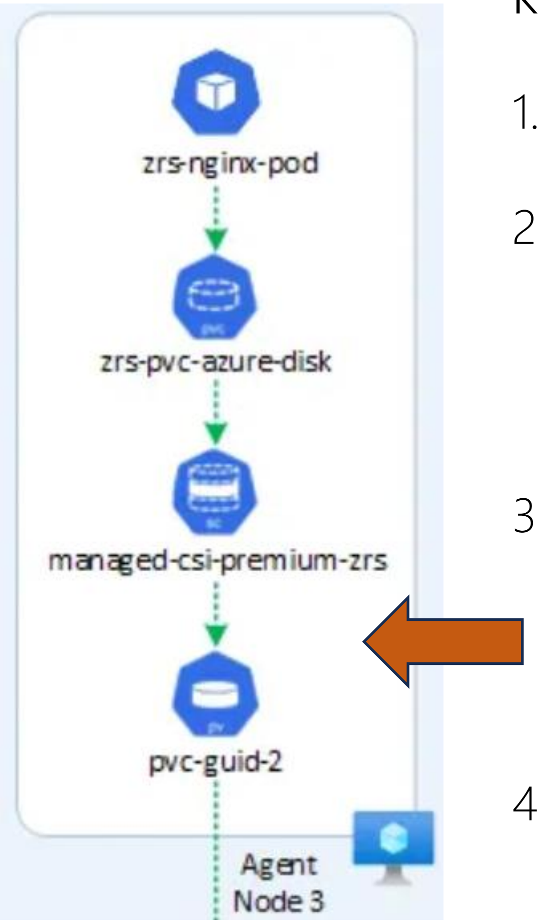
AZURE DAY

Deploying a reliable Workload



AZURE DAY

Deploy a Workload that uses ZRS Storage to a ZR Node Pool



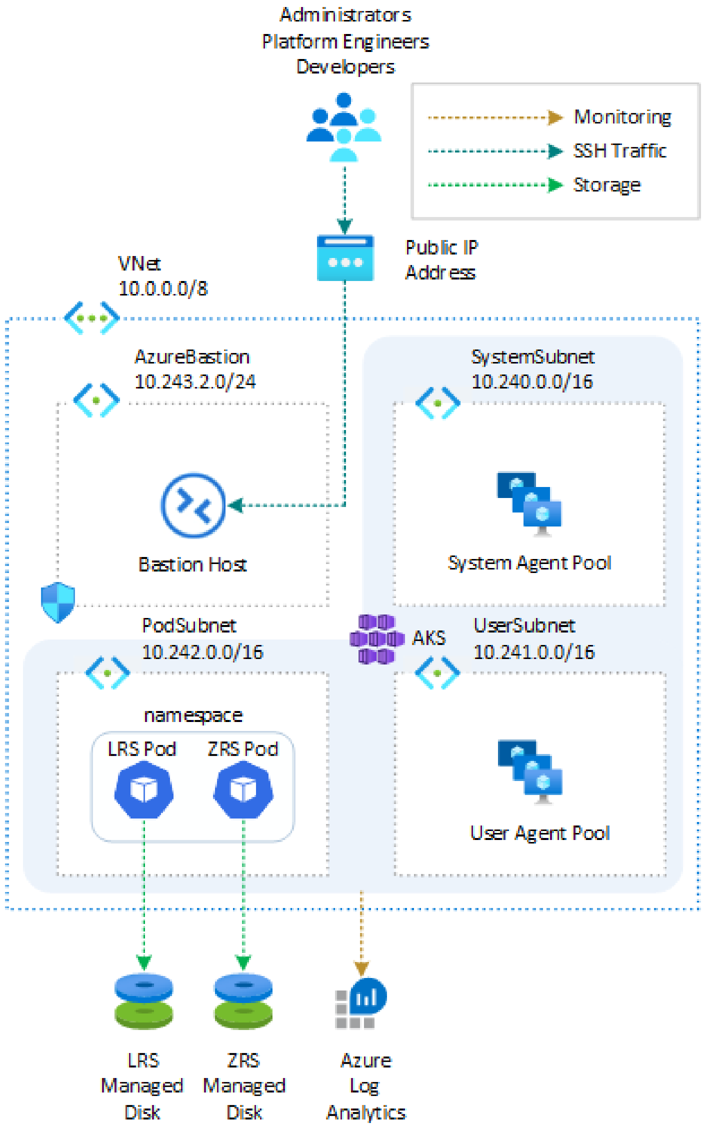
This strategy uses Azure Disks CSI Driver to create and attach Kubernetes persistent volumes based on ZRS managed disks:

1. Create a Kubernetes deployment (YAML manifest).

2. Use *node selectors or node affinity* to constraint the Kubernetes Scheduler to run the pods of each deployments on the agent nodes of a specific user-mode zone-redundant node pool.

3. Create a persistent volume claim which references a storage class which makes use of ZRS, that is the *managed-csi-premium-zrs* storage class we introduced in the previous section.

4. When deploying pods to a zone-redundant node pool, it is essential to ensure optimal distribution and resilience. To achieve this, you can utilize the *Pod Topology Spread Constraints* Kubernetes feature



Deploy a Workload that uses LRS Storage across Zonal Node Pools

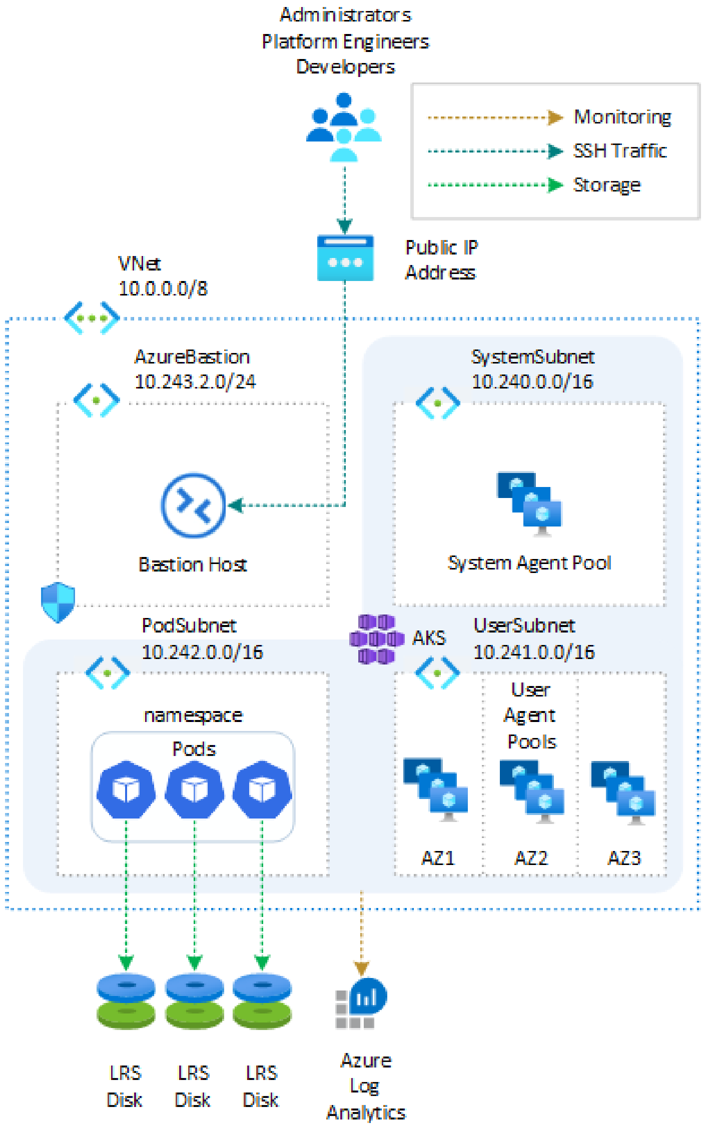
This strategy uses the Azure Disks CSI Driver to create and attach Kubernetes persistent volumes based on LRS managed disks:

1. Create a separate Kubernetes deployment for each zonal node pool.

2. Use *node selectors or node affinity* to constraint the Kubernetes Scheduler to run the pods of each deployments on the agent nodes of a specific zonal node pool.

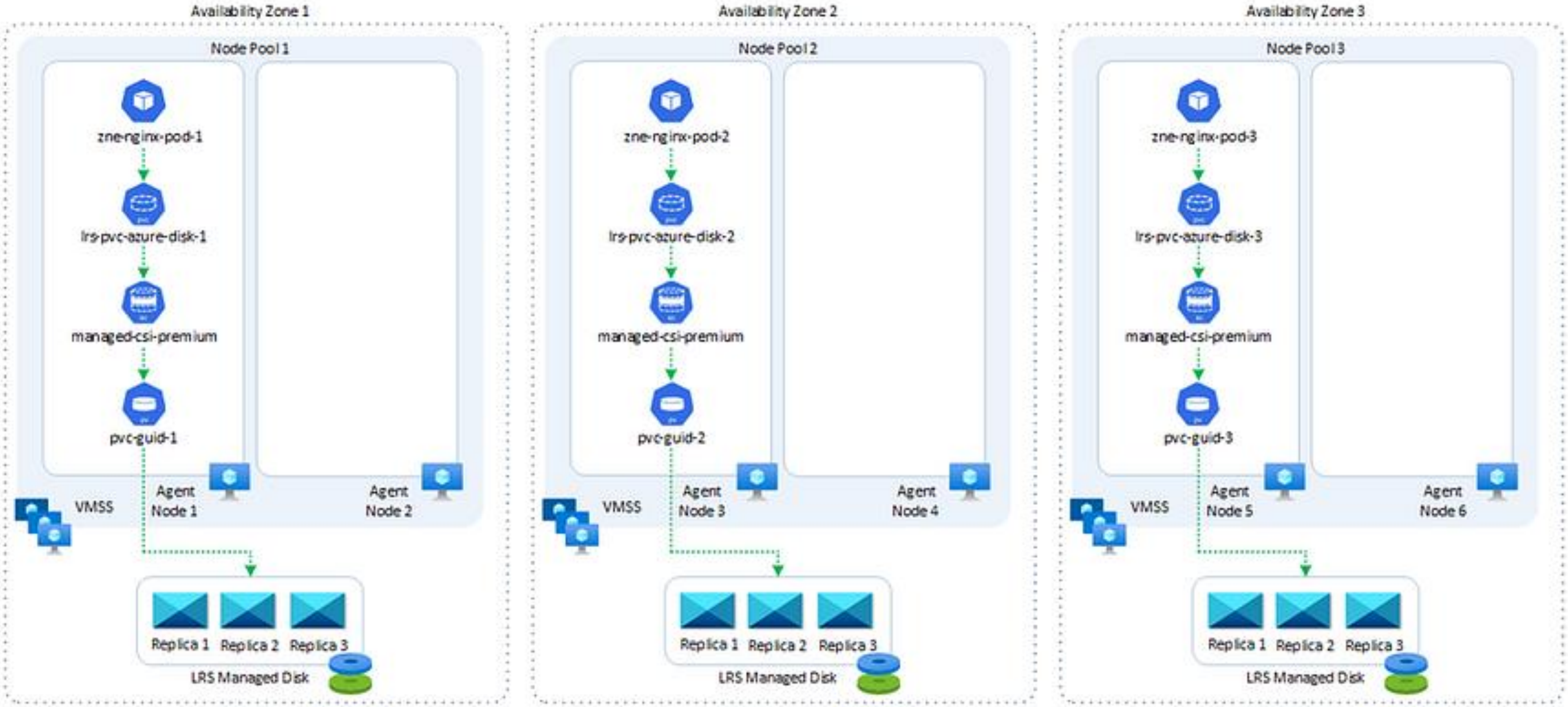
3. Create a separate persistent volume claim for each zonal deployment.

4. When deploying pods to an AKS cluster that spans multiple availability zones, it is essential to ensure optimal distribution and resilience. To achieve this, you can utilize the *Pod Topology Spread Constraints* Kubernetes feature.

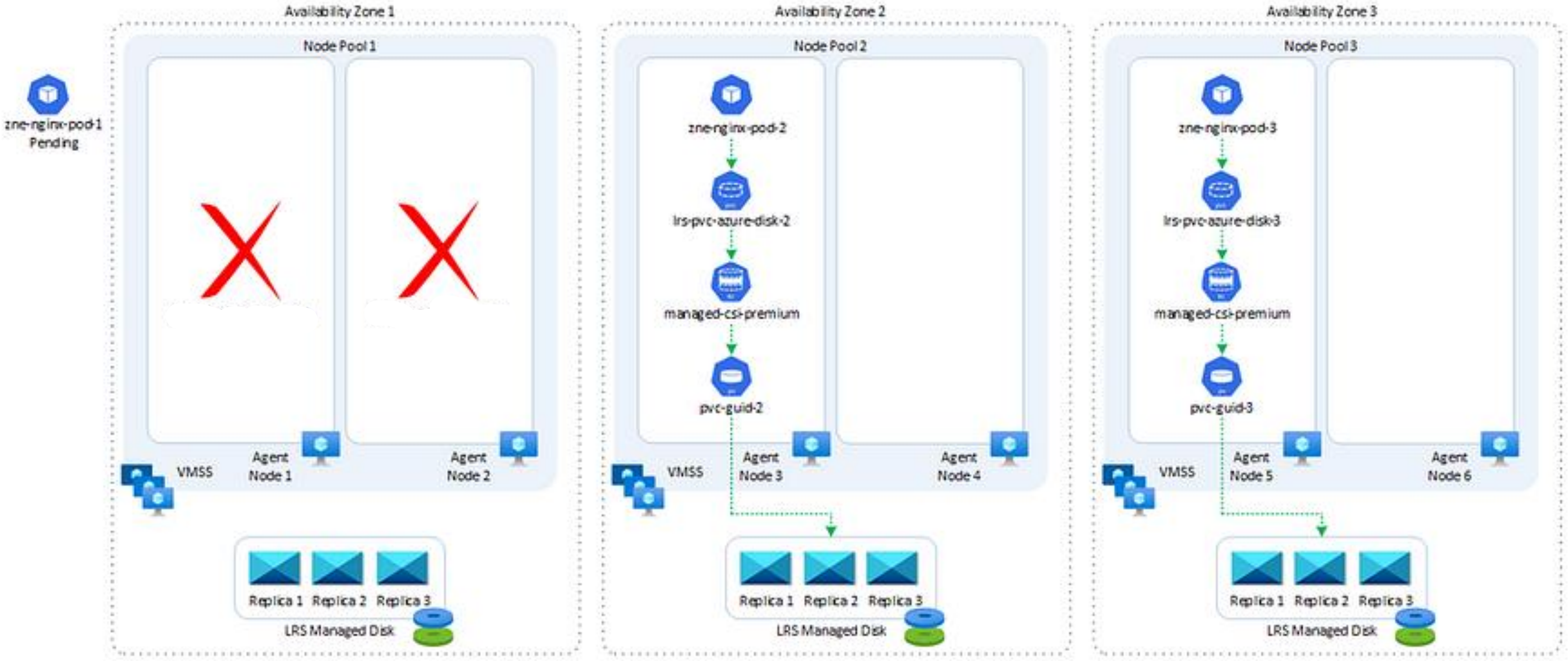


AZURE DAY

Pod Distribution in a cluster with three node pools for each AZ



Pod Distribution in a cluster with three node pools for each AZ



Pros and Cons considerations in both approaches

1. AKS Cluster with one Redundant Node Pool:

- *Pros*: The advantage of this approach is that *you can use a single deployment* and Pod Topology Spread Constraints to distribute the pod replicas across the availability zones within a region.

- *Cons*: a drawback is that you need to use ZRS to guarantee that Azure Disks mounted as persistent volumes can be accessed from any availability zone. *ZRS storage provides better intra-region resiliency than LRS, but it's more costly.*

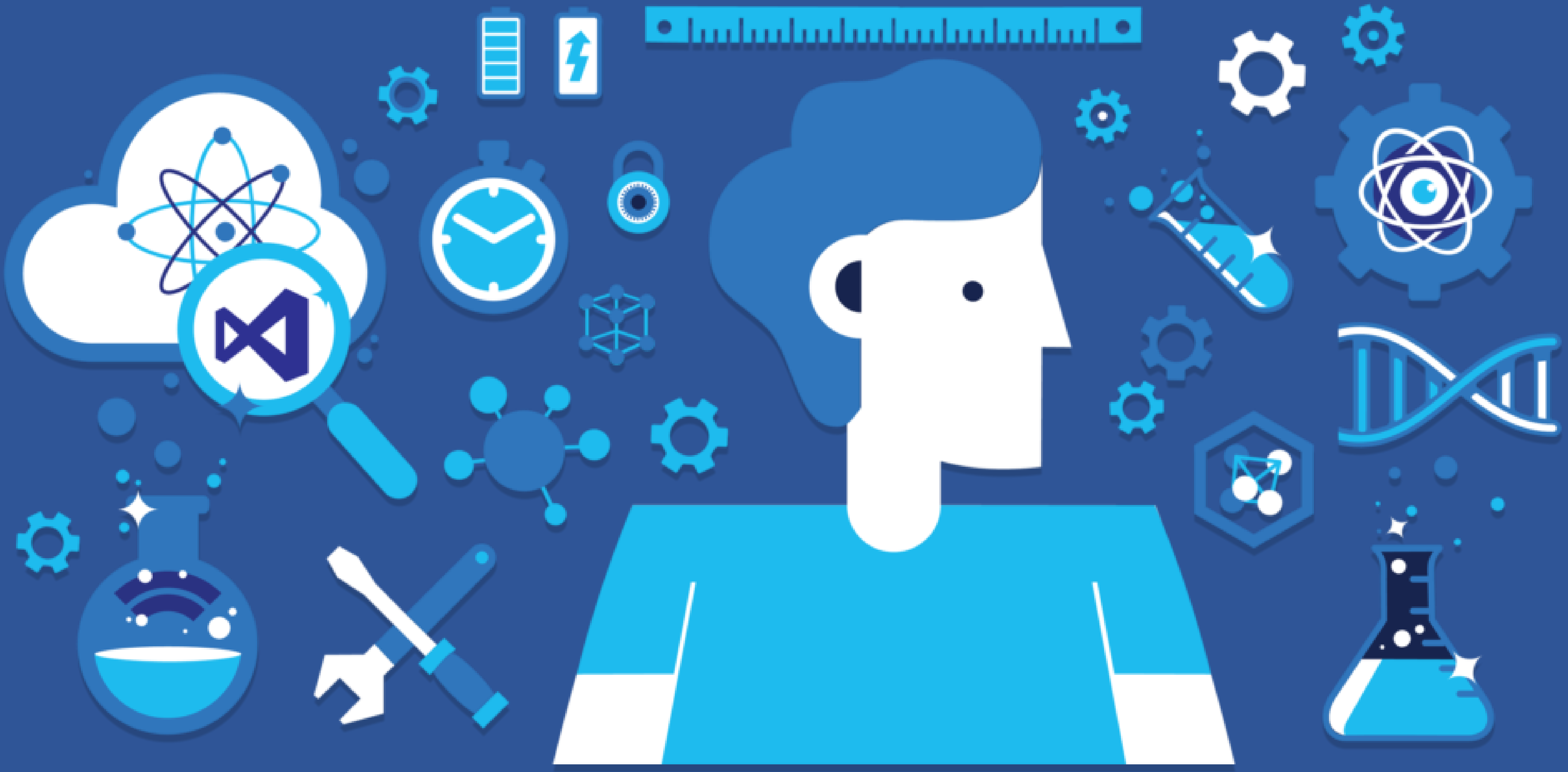
2. AKS Cluster with three Node Pools:

- *Pros*: The advantage of this approach is that you can use LRS when creating and mounting Azure disks, *which are less expensive and more durable than ZRS Azure disks.*

- *Cons*: a drawback is that you need to *create and scale multiple separate deployments*, one for each availability zone, for the same workload. Another one is that *you cannot share same state and data in a Persistent volume with all pods in all AZ*, if you consider them as part of the same service. This caveat is mitigated with a wise usage of the service itself (affinity?)

AZURE DAY

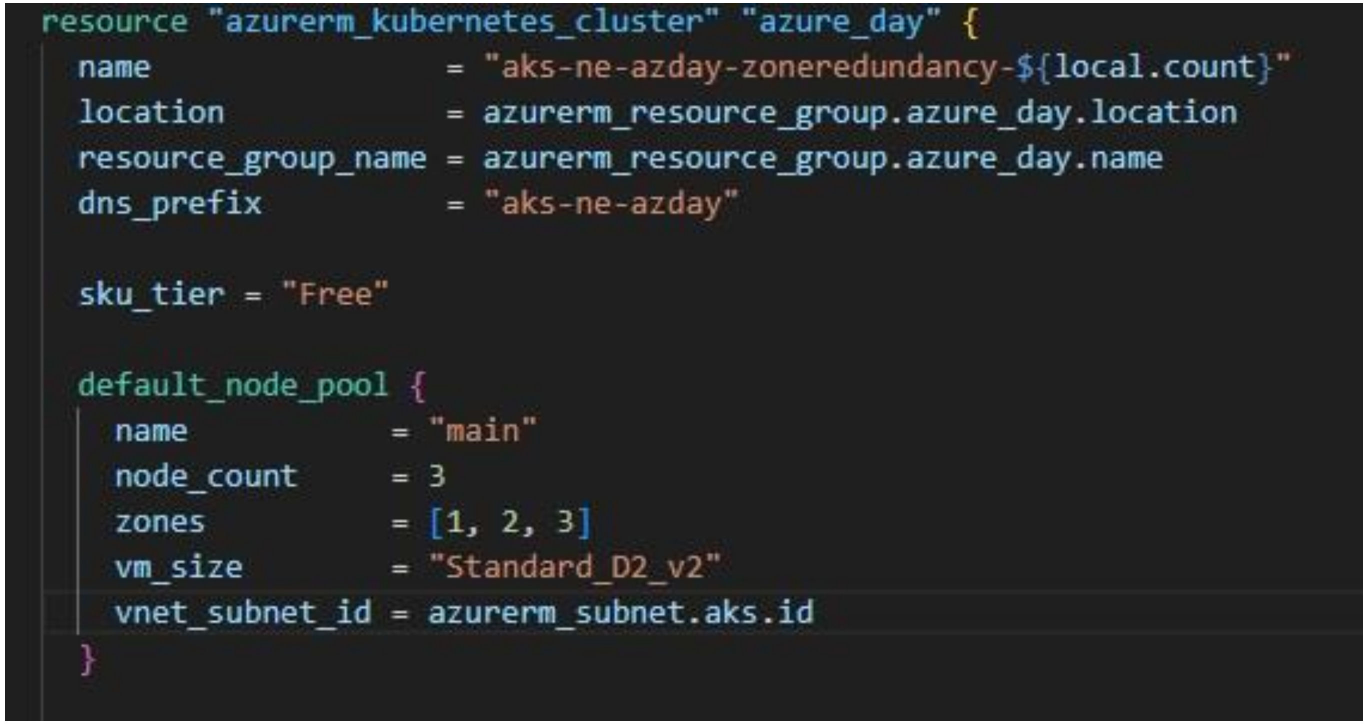
Demo



Prerequisites

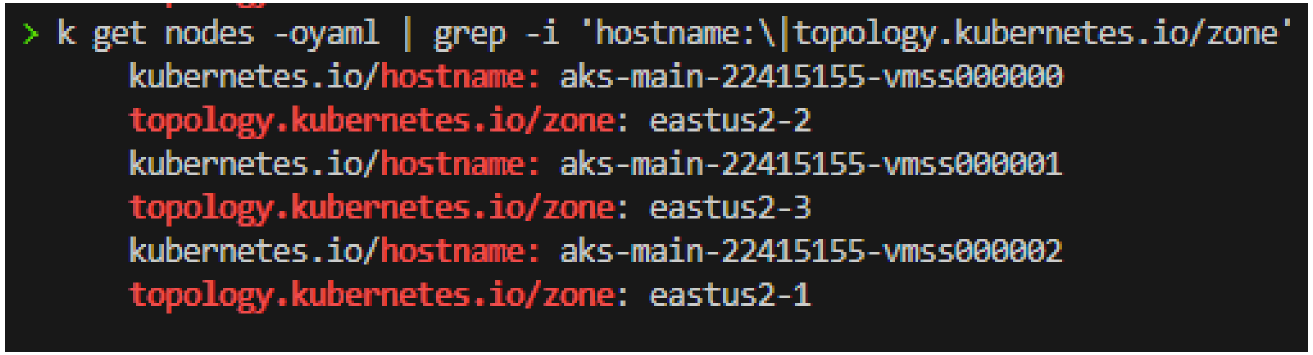
1. An active Azure subscription
2. MS Visual Studio Code and HashiCorp Terraform
3. Azure CLI (version 2.56.0 or later installed)
4. User with sufficient permissions to assign roles (as a User Access Administrator or Owner)
5. Account needs Microsoft.Resources/deployments/write at the subscription level
6. Verify ZRS disks regional availability - https://t.ly/xdHNe

AKS Cluster – AKS Creation – Location East US



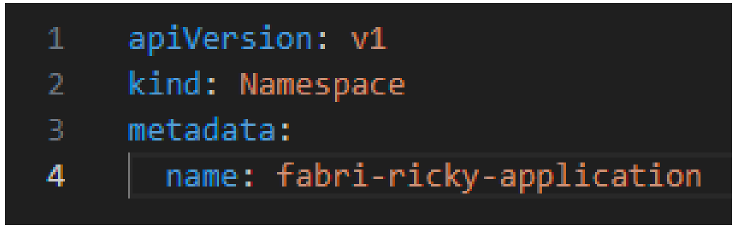
```
resource "azurerm_kubernetes_cluster" "azure_day" {
  name                = "aks-ne-azday-zoneredundancy-${local.count}"
  location            = azurerm_resource_group.azure_day.location
  resource_group_name = azurerm_resource_group.azure_day.name
  dns_prefix          = "aks-ne-azday"

  sku_tier = "Free"

  default_node_pool {
    name           = "main"
    node_count     = 3
    zones          = [1, 2, 3]
    vm_size        = "Standard_D2_v2"
    vnet_subnet_id = azurerm_subnet.aks.id
  }
```

AKS Cluster - Location and Zone Verification



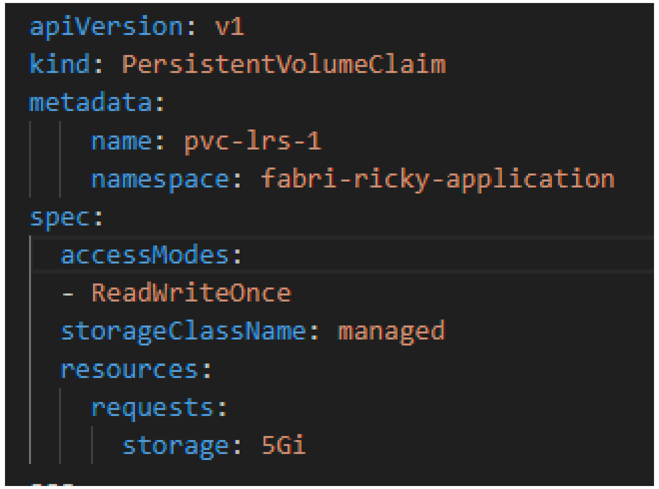
```
> k get nodes -oyaml | grep -i 'hostname:\|topology.kubernetes.io/zone'
      kubernetes.io/hostname: aks-main-22415155-vmss000000
      topology.kubernetes.io/zone: eastus2-2
      kubernetes.io/hostname: aks-main-22415155-vmss000001
      topology.kubernetes.io/zone: eastus2-3
      kubernetes.io/hostname: aks-main-22415155-vmss000002
      topology.kubernetes.io/zone: eastus2-1
```

AKS Cluster - Namespaces Creation



```yaml
apiVersion: v1
kind: Namespace
metadata:
  name: fabri-ricky-application
```

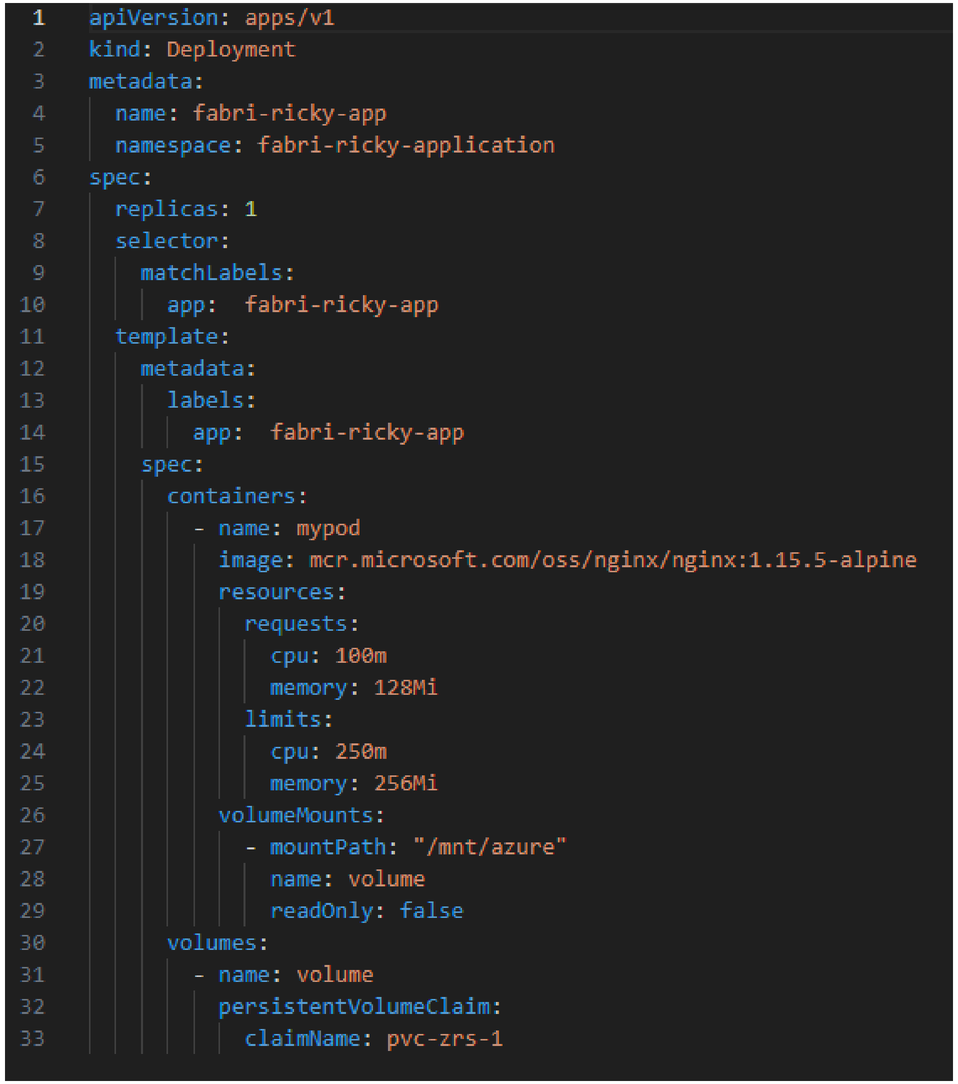
AKS Cluster - LRS Storage Creation - Default



```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
    name: pvc-lrs-1
    namespace: fabri-ricky-application
spec:
  accessModes:
  - ReadWriteOnce
  storageClassName: managed
  resources:
    requests:
      storage: 5Gi
```

AZURE DAY

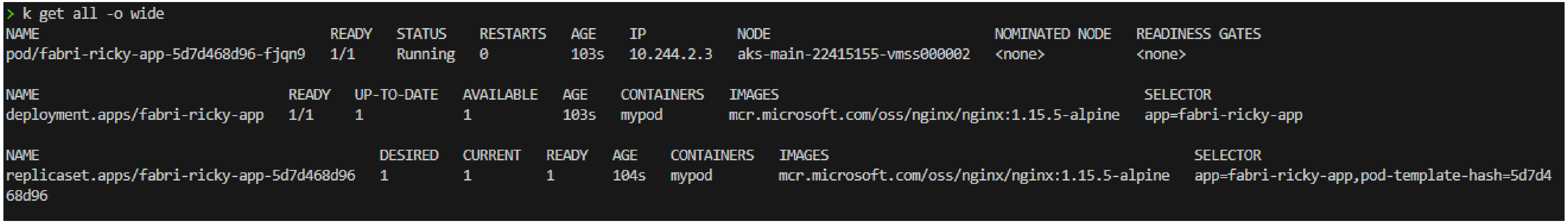
AKS Cluster - Application Deployment



```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: fabri-ricky-app
  namespace: fabri-ricky-application
spec:
  replicas: 1
  selector:
    matchLabels:
      app:  fabri-ricky-app
  template:
    metadata:
      labels:
        app:  fabri-ricky-app
    spec:
      containers:
        - name: mypod
          image: mcr.microsoft.com/oss/nginx/nginx:1.15.5-alpine
          resources:
            requests:
              cpu: 100m
              memory: 128Mi
            limits:
              cpu: 250m
              memory: 256Mi
          volumeMounts:
            - mountPath: "/mnt/azure"
              name: volume
              readOnly: false
      volumes:
        - name: volume
          persistentVolumeClaim:
            claimName: pvc-zrs-1
```

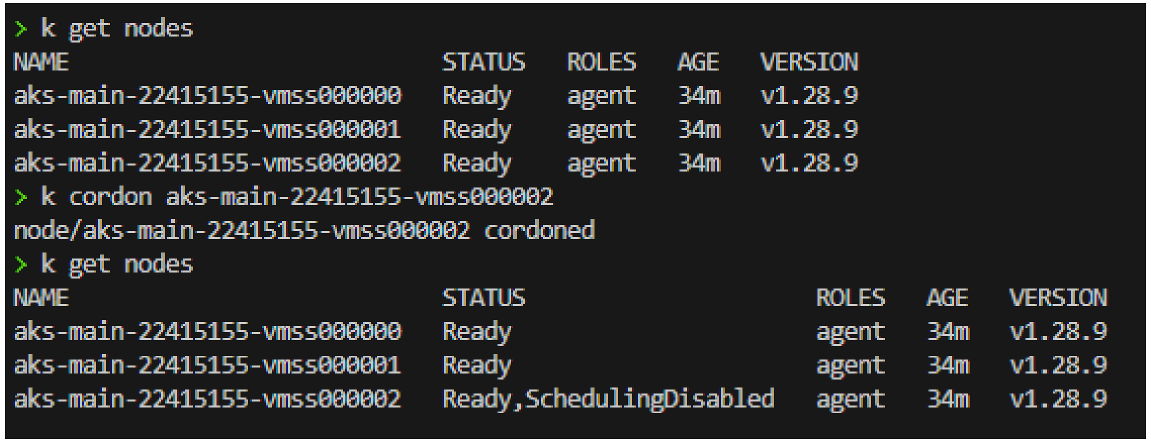
AZURE DAY

AKS Cluster – Active POD Node Verification



```
> k get all -o wide
NAME                                   READY   STATUS    RESTARTS   AGE    IP           NODE                           NOMINATED NODE   READINESS GATES
pod/fabri-ricky-app-5d7d468d96-fjqn9   1/1     Running   0          103s   10.244.2.3   aks-main-22415155-vmss000002   <none>           <none>

NAME                              READY   UP-TO-DATE   AVAILABLE   AGE    CONTAINERS   IMAGES                                            SELECTOR
deployment.apps/fabri-ricky-app   1/1     1            1           103s   mypod        mcr.microsoft.com/oss/nginx/nginx:1.15.5-alpine   app=fabri-ricky-app

NAME                                         DESIRED   CURRENT   READY   AGE    CONTAINERS   IMAGES                                            SELECTOR
replicaset.apps/fabri-ricky-app-5d7d468d96   1         1         1       104s   mypod        mcr.microsoft.com/oss/nginx/nginx:1.15.5-alpine   app=fabri-ricky-app,pod-template-hash=5d7d4
68d96
```

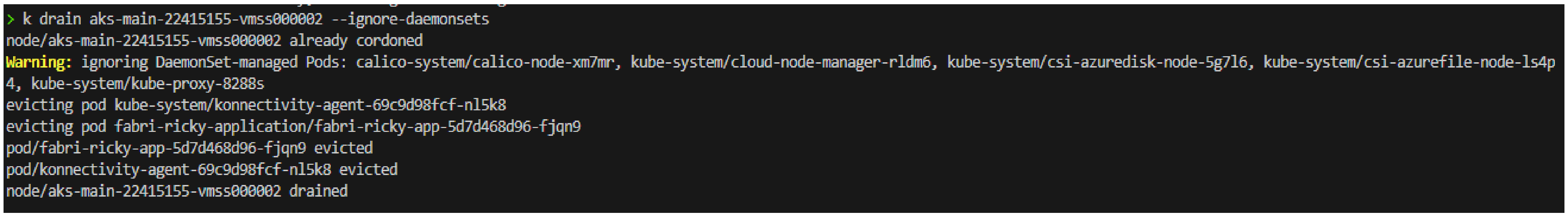
AKS Cluster - Cordon Node



```
> k get nodes
NAME                              STATUS   ROLES   AGE   VERSION
aks-main-22415155-vmss000000   Ready    agent   34m   v1.28.9
aks-main-22415155-vmss000001   Ready    agent   34m   v1.28.9
aks-main-22415155-vmss000002   Ready    agent   34m   v1.28.9
> k cordon aks-main-22415155-vmss000002
node/aks-main-22415155-vmss000002 cordoned
> k get nodes
NAME                              STATUS                     ROLES   AGE   VERSION
aks-main-22415155-vmss000000   Ready                      agent   34m   v1.28.9
aks-main-22415155-vmss000001   Ready                      agent   34m   v1.28.9
aks-main-22415155-vmss000002   Ready,SchedulingDisabled   agent   34m   v1.28.9
```

AZURE DAY

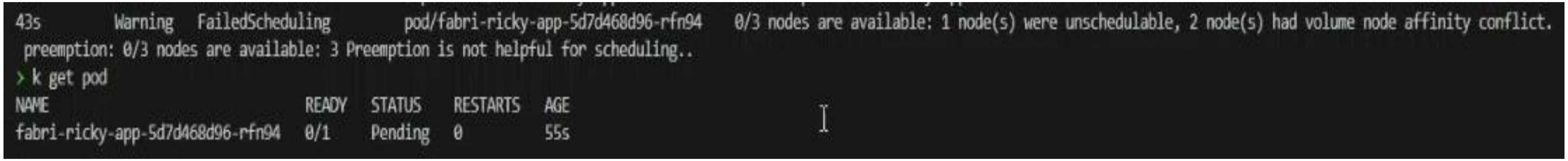
AKS Cluster - Drain Node



```
> k drain aks-main-22415155-vmss000002 --ignore-daemonsets
node/aks-main-22415155-vmss000002 already cordoned
Warning: ignoring DaemonSet-managed Pods: calico-system/calico-node-xm7mr, kube-system/cloud-node-manager-rldm6, kube-system/csi-azuredisk-node-5g7l6, kube-system/csi-azurefile-node-ls4p
4, kube-system/kube-proxy-8288s
evicting pod kube-system/konnectivity-agent-69c9d98fcf-nl5k8
evicting pod fabri-ricky-application/fabri-ricky-app-5d7d468d96-fjqn9
pod/fabri-ricky-app-5d7d468d96-fjqn9 evicted
pod/konnectivity-agent-69c9d98fcf-nl5k8 evicted
node/aks-main-22415155-vmss000002 drained
```

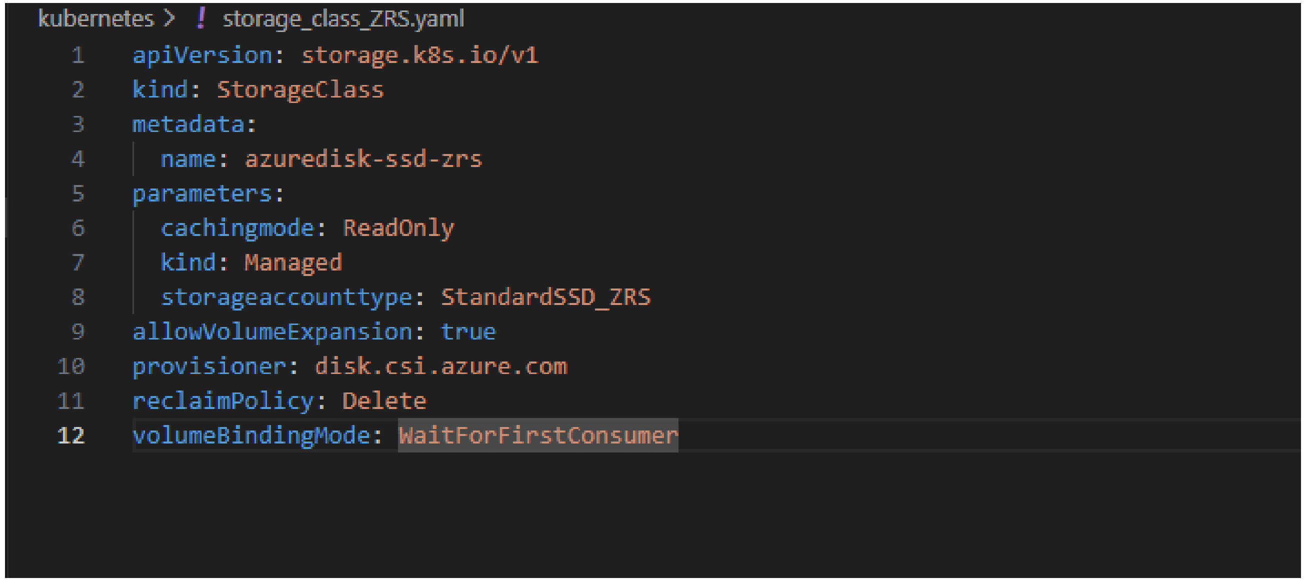
AZURE DAY

AKS Cluster - Deployment Error



```
43s         Warning   FailedScheduling         pod/fabri-ricky-app-5d7d468d96-rfn94    0/3 nodes are available: 1 node(s) were unschedulable, 2 node(s) had volume node affinity conflict.
 preemption: 0/3 nodes are available: 3 Preemption is not helpful for scheduling..
> k get pod
NAME                                READY   STATUS    RESTARTS   AGE
fabri-ricky-app-5d7d468d96-rfn94    0/1     Pending   0          55s
```

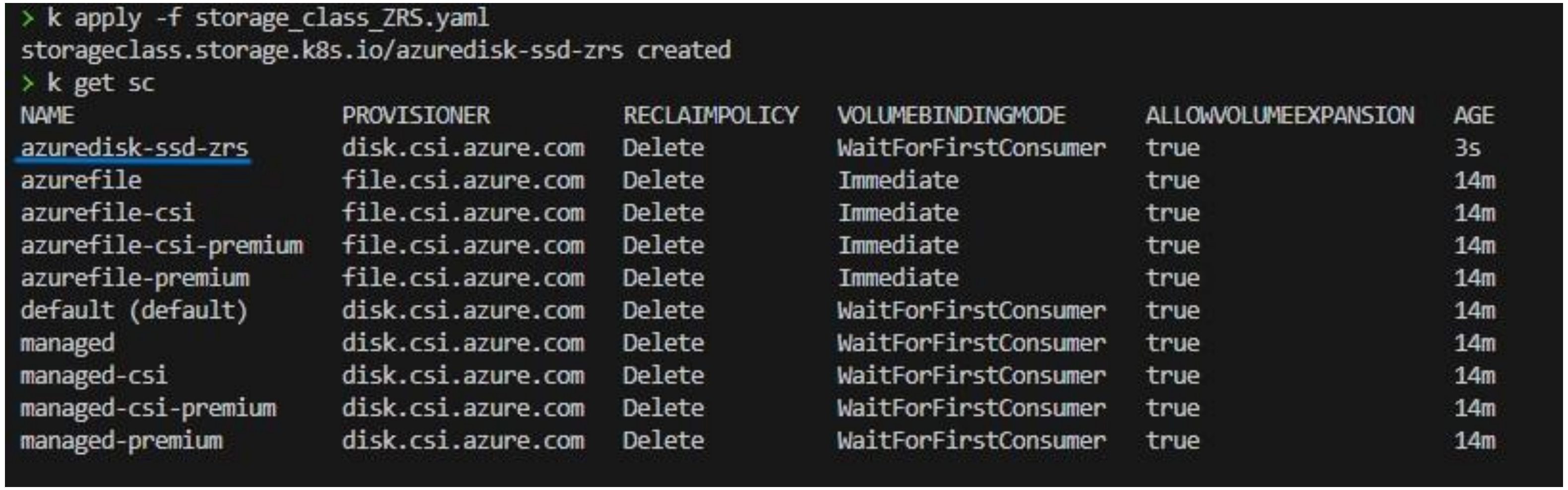
AKS Cluster – Create Storage Class ZRS



kubernetes > ! storage_class_ZRS.yaml

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: azuredisk-ssd-zrs
parameters:
  cachingmode: ReadOnly
  kind: Managed
  storageaccounttype: StandardSSD_ZRS
allowVolumeExpansion: true
provisioner: disk.csi.azure.com
reclaimPolicy: Delete
volumeBindingMode: WaitForFirstConsumer
```

AKS Cluster – Verify Storage Class ZRS



```
> k apply -f storage_class_ZRS.yaml
storageclass.storage.k8s.io/azuredisk-ssd-zrs created
> k get sc
NAME                    PROVISIONER          RECLAIMPOLICY   VOLUMEBINDINGMODE      ALLOWVOLUMEEXPANSION   AGE
azuredisk-ssd-zrs       disk.csi.azure.com   Delete          WaitForFirstConsumer   true                   3s
azurefile               file.csi.azure.com   Delete          Immediate              true                   14m
azurefile-csi           file.csi.azure.com   Delete          Immediate              true                   14m
azurefile-csi-premium   file.csi.azure.com   Delete          Immediate              true                   14m
azurefile-premium       file.csi.azure.com   Delete          Immediate              true                   14m
default (default)       disk.csi.azure.com   Delete          WaitForFirstConsumer   true                   14m
managed                 disk.csi.azure.com   Delete          WaitForFirstConsumer   true                   14m
managed-csi             disk.csi.azure.com   Delete          WaitForFirstConsumer   true                   14m
managed-csi-premium     disk.csi.azure.com   Delete          WaitForFirstConsumer   true                   14m
managed-premium         disk.csi.azure.com   Delete          WaitForFirstConsumer   true                   14m
```

AZURE DAY ROME 2024

VOTA LA SESSIONE

Facci sapere se questa sessione ti piace.
Inquadra il QR code e esprimi una tua opinione.
Ci aiuterai a migliorare.

Microsoft avanade eng softwareone

BCSoft consulenza sistemi informativi 4wardPRO AN IMPRESOFT COMPANY LOBRA UNIKEY Packt JET BRAINS NET CODE