STORAGE DEVELOPER CONFERENCE

SDC 23

BY Developers FOR Developers

Massively scalable storage for stateful containers on Azure

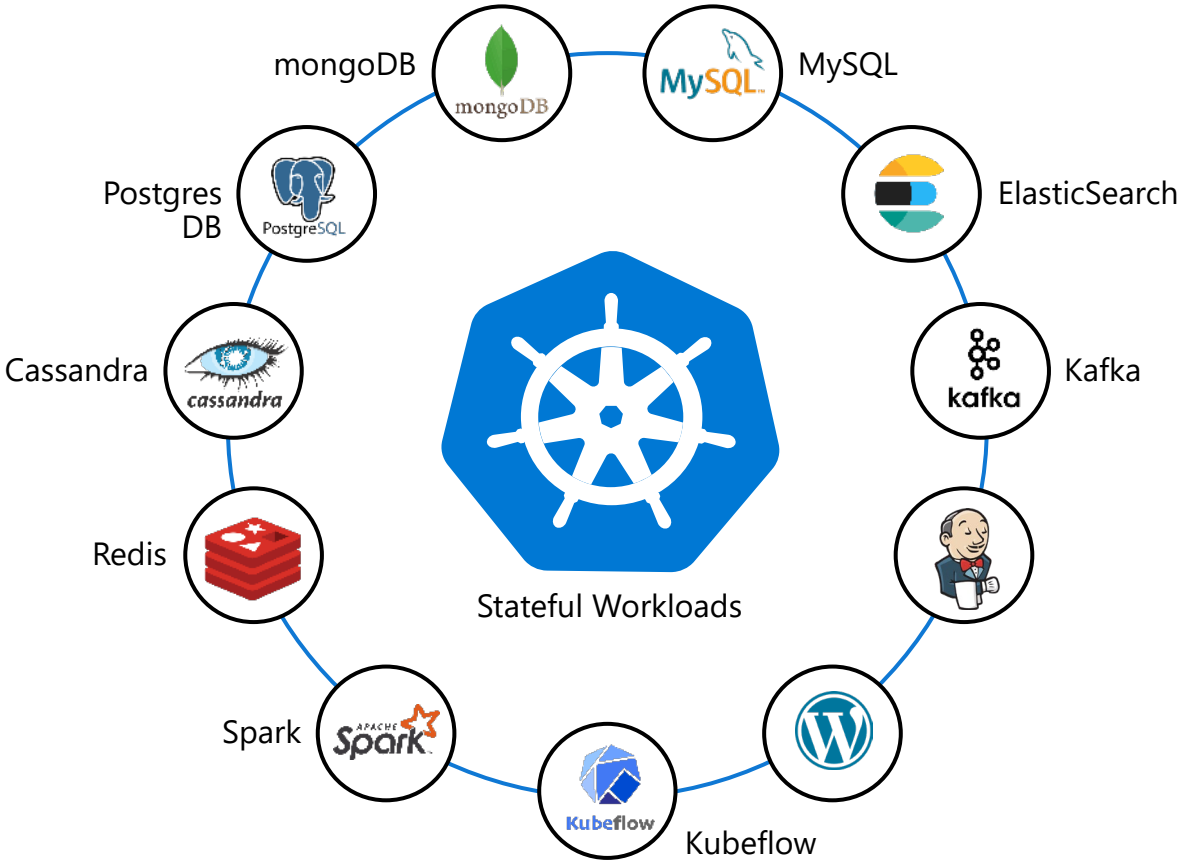
Malcolm Smith, Principal Software Engineer | Vybava Ramadoss, Principal Product Manager

AGENDA

- Stateful workloads challenges
- Azure Container Storage overview
- Architecture deep dive and extensibility
- Elastic SAN Overview
- Scaling with Azure Elastic SAN

Stateful workloads running on containers

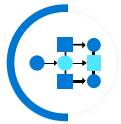
Run large scale stateful container workloads with scalable, performant, available and cost-effective Storage



Challenges

Existing solutions built for IaaS centric architecture and retrofitted to containers

Unable to match the scale out speed and target of containers (pods)



Slow pod failover resulting in degraded availability of stateful containers

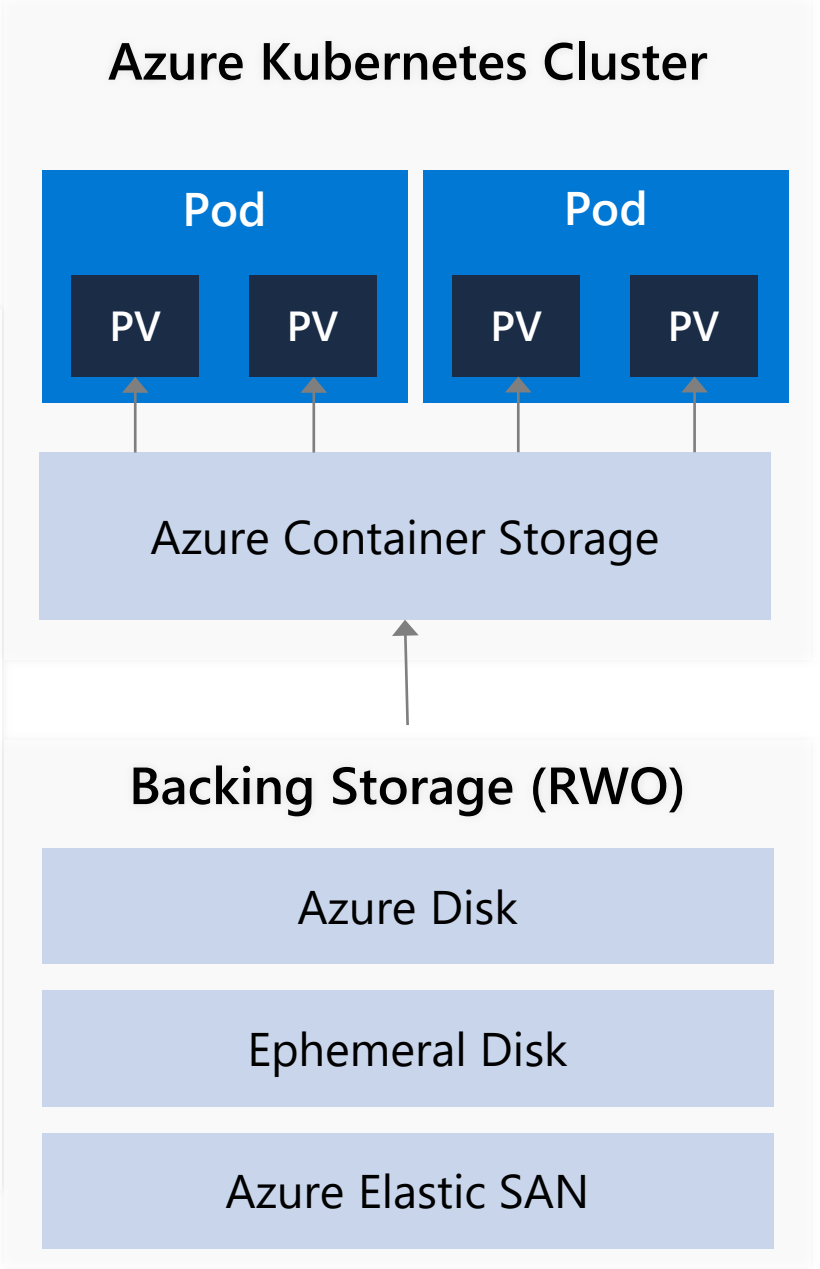
Limited coverage of available storage offerings

Azure Container Storage overview

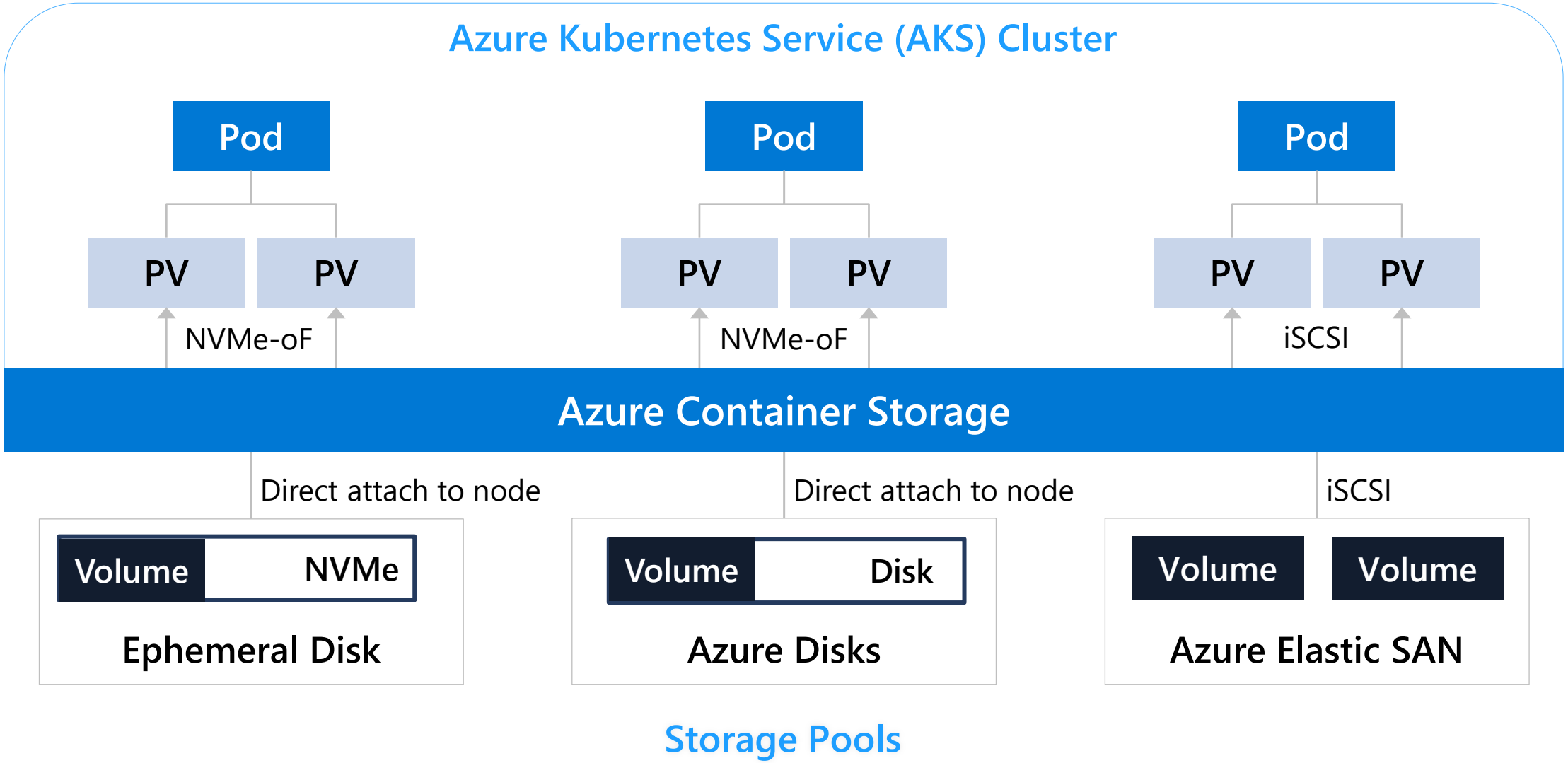
Why Azure Container Storage

Industry's first platform-managed container storage offering

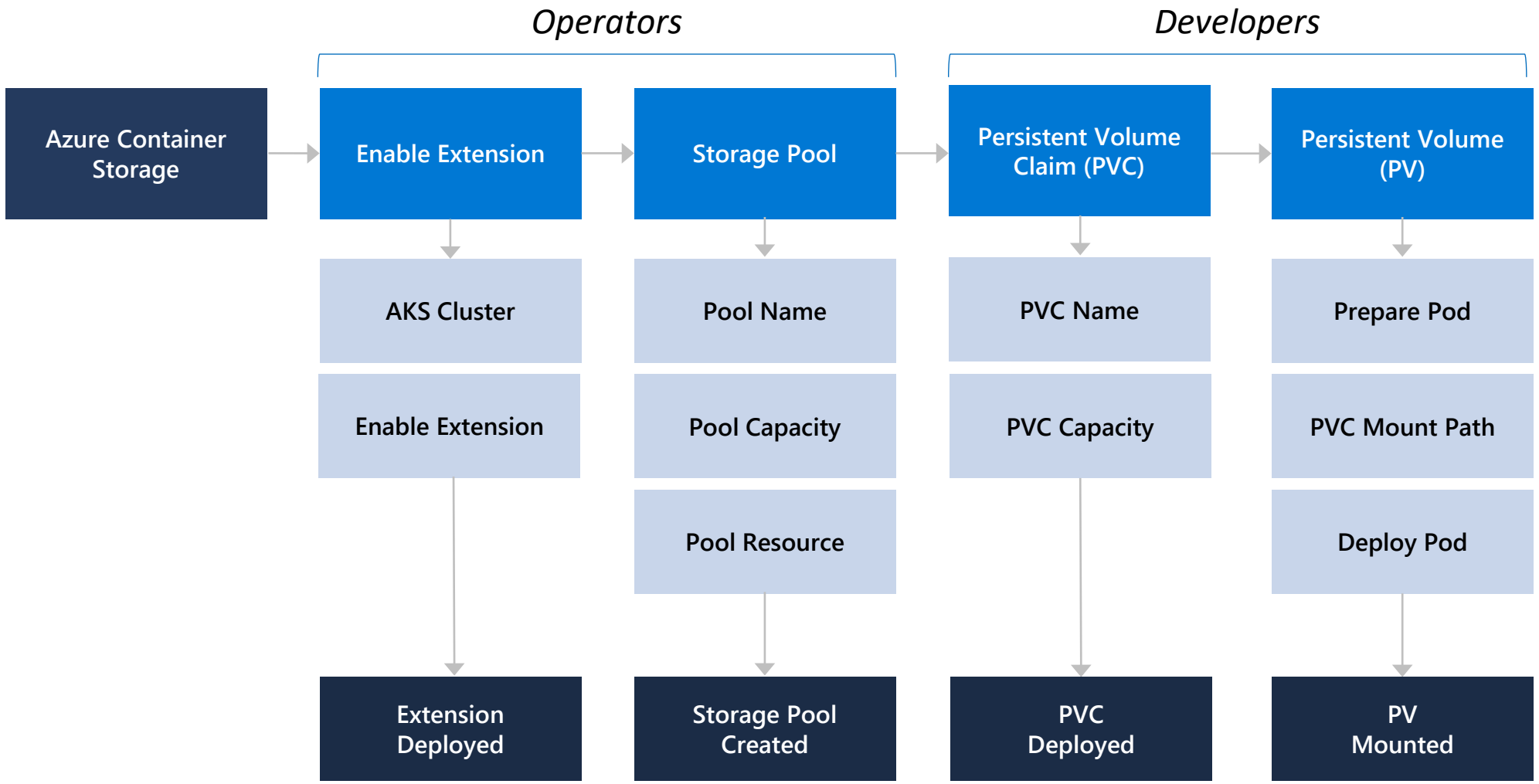
- Reduced Total Cost of Ownership (TCO)
- Kubernetes-native volume orchestration
- Rapid scale out and fast failover
- Unified management experience for the storage of your choice



Inside Azure Container Storage



Use Persistent Volumes with Containers



Example

```
apiVersion: containerstorage.azure.com/v1beta1
kind: StoragePool
metadata:
  name: azuredisk
  namespace: acstor
spec:
  poolType:
    azureDisk: {}
  resources:
    requests: {"storage": 1Ti}
```

Storage Pool Definition

```
kubectl describe sc acstor-azuredisk
Provisioner:
containerstorage.csi.azure.com
Parameters:
acstor.azure.com/
storagepool = azuredisk,
proto = nvmf, repl = 1
```

Storage Classes

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: statefulset-kafka
…
spec:
  …
 volumeClaimTemplates:
   - metadata:
      name: persistent-storage
     spec:
      storageClassName: acstor-azuredisk
      accessModes: ["ReadWriteOnce"]
      resources:
       requests:
         storage: 10Gi
```

Stateful Sets

Azure Container Storage deep dive

Deep Dive

Protocols
Guest based attach

NVMeOF | iSCSI

Data Services
Extend storage provider capabilities

Snapshots | Encryption | Replication

Volume Provisioner
Shared provisioning

Exposed as File

Resource Provider
Externalize state

Metadata

Capacity Provisioner
Handle multiple backends

Storage Pool | Connect with CSI Drivers

Backing Storage
Provides perf/scale options


Local NVMe

Azure Disk

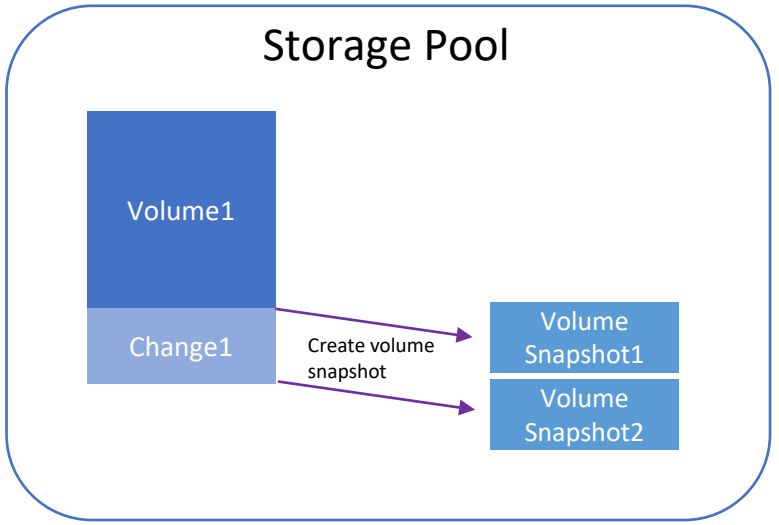

Azure Elastic SAN


Other

Layering on the data services

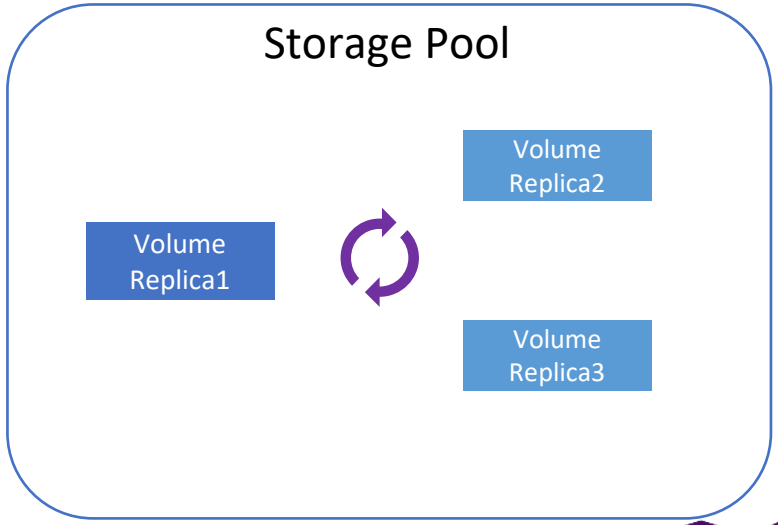
Volume Snapshots

- Aligned with CSI Snapshot API
- Works with volume provisioner to restore snapshots
- Instant snapshot create and read using copy on write



Replication

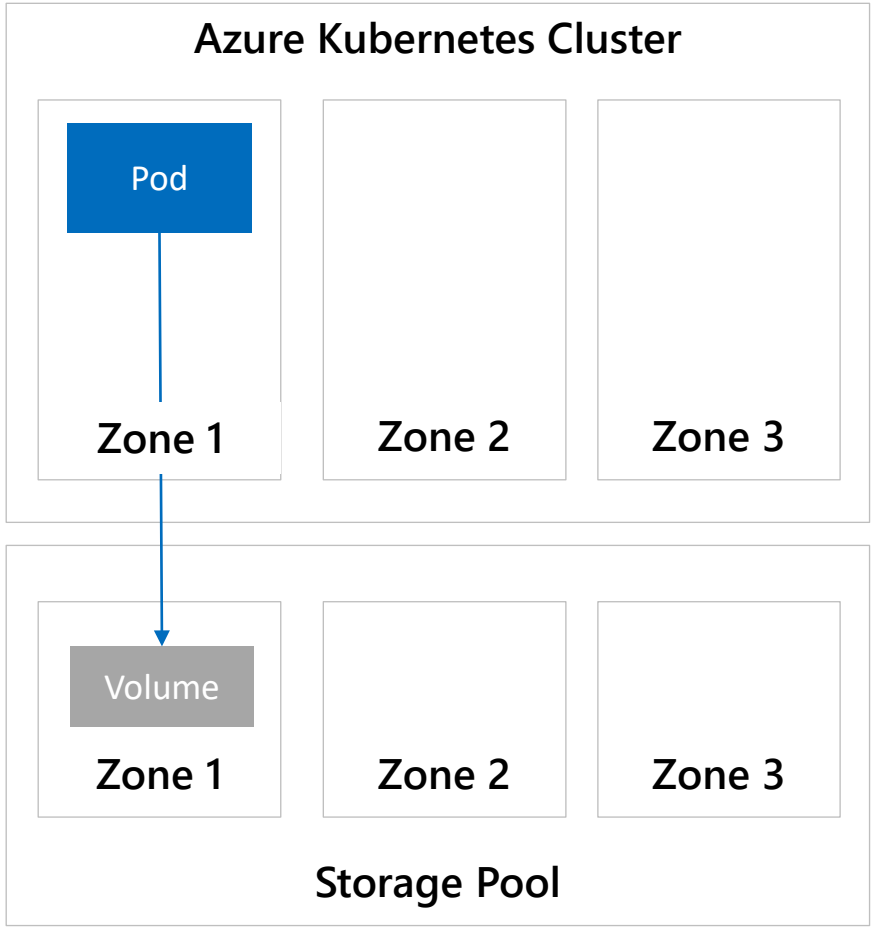
- Storage pool configured with 1+ replicas
- Replication engine sync writes using n/2+1 quorum
- Round robin reads across replicas
- Checks for data integrity



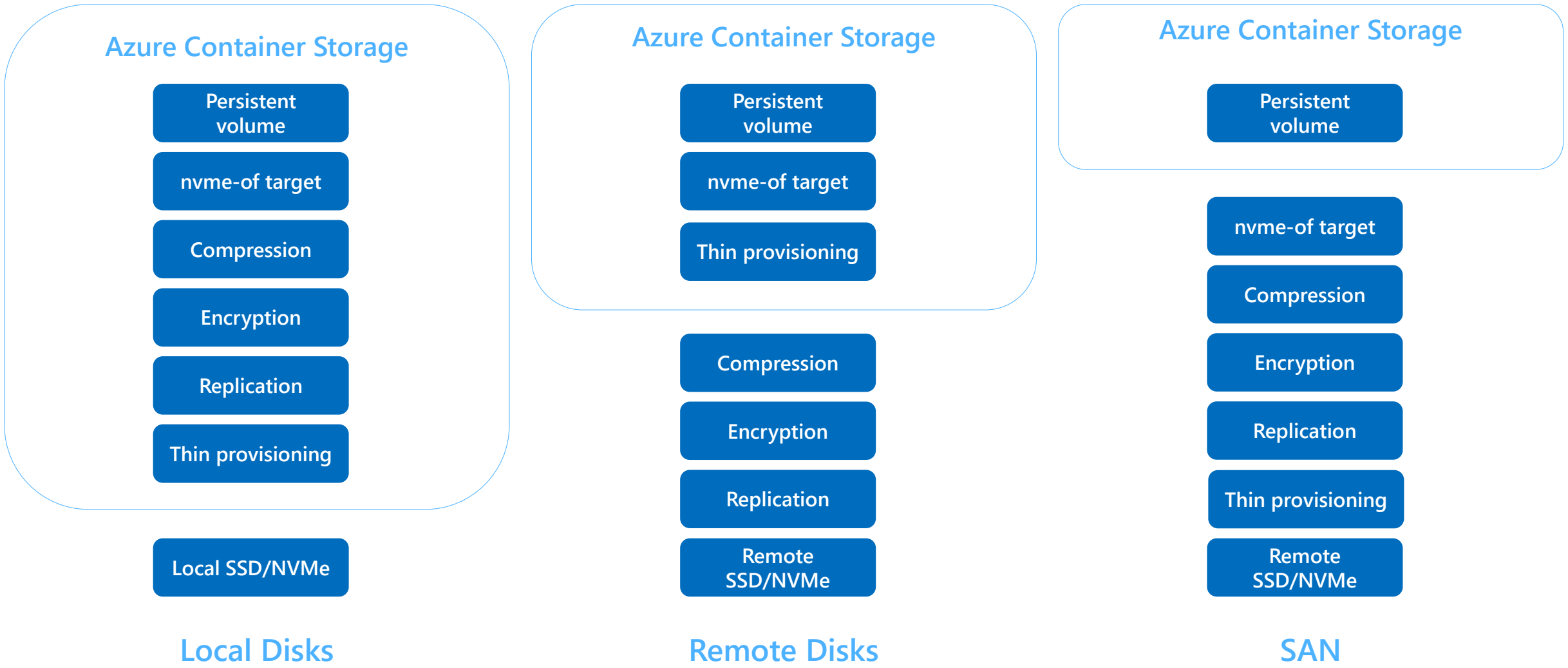
Intelligent placement with capacity provisioner

Using Storage Pools

- Storage pool aggregates capacity and performance across homogenous storage in the cluster
- Dynamic placement decisions based on application performance and availability needs
- Enables parameterized capacity request mapping to storage pool with capability
  - E.g., 100 GB 20K IOPS volume can be served from Premium storage pool
- Understands Topology including multi-zone pools
- Provides a consistent experience by abstracting backing storage and adding capabilities as needed

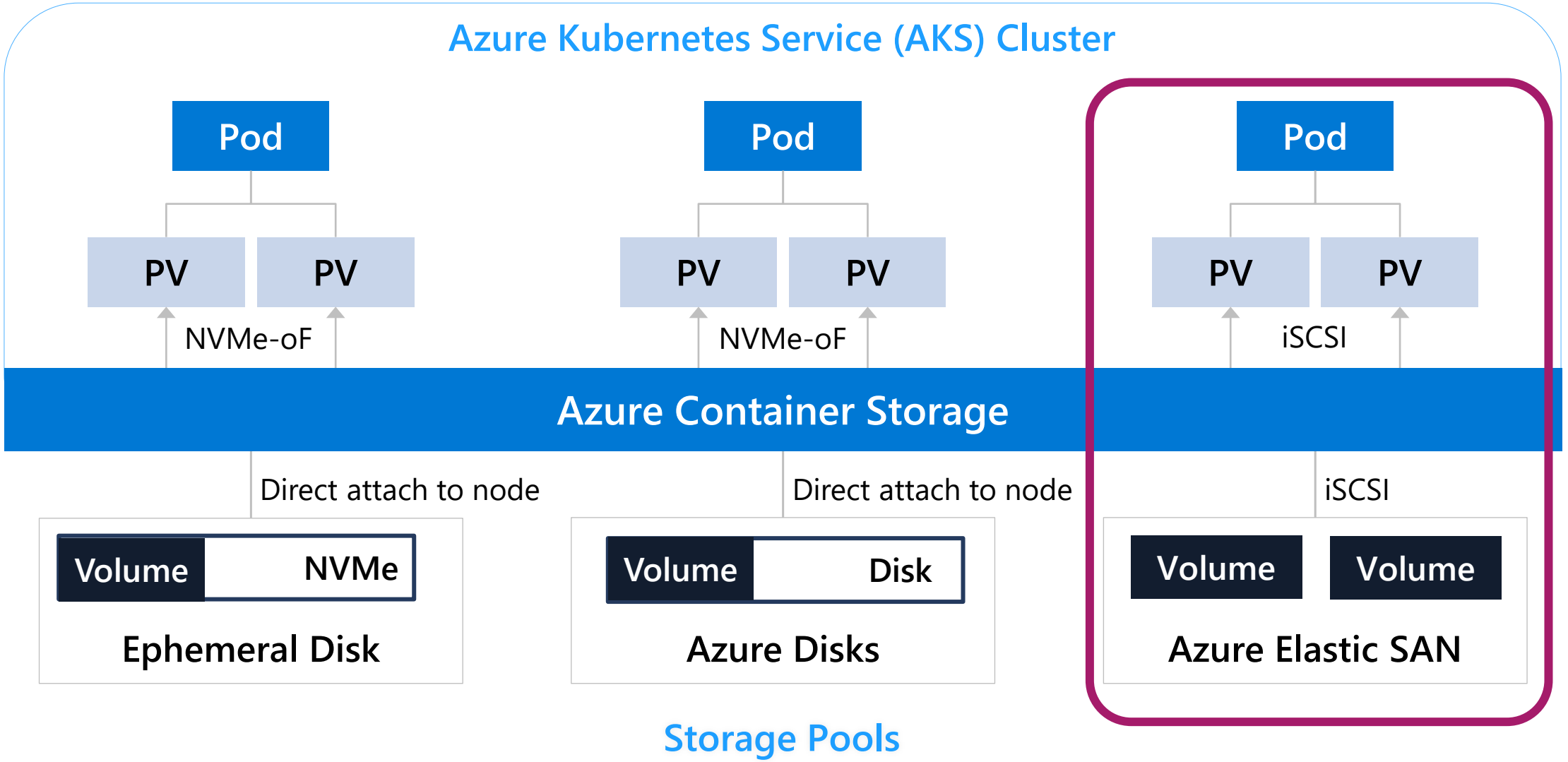


Extensible to support multiple backing storage



Azure Elastic SAN overview

Inside Azure Container Storage



Why use SAN for Container Storage?

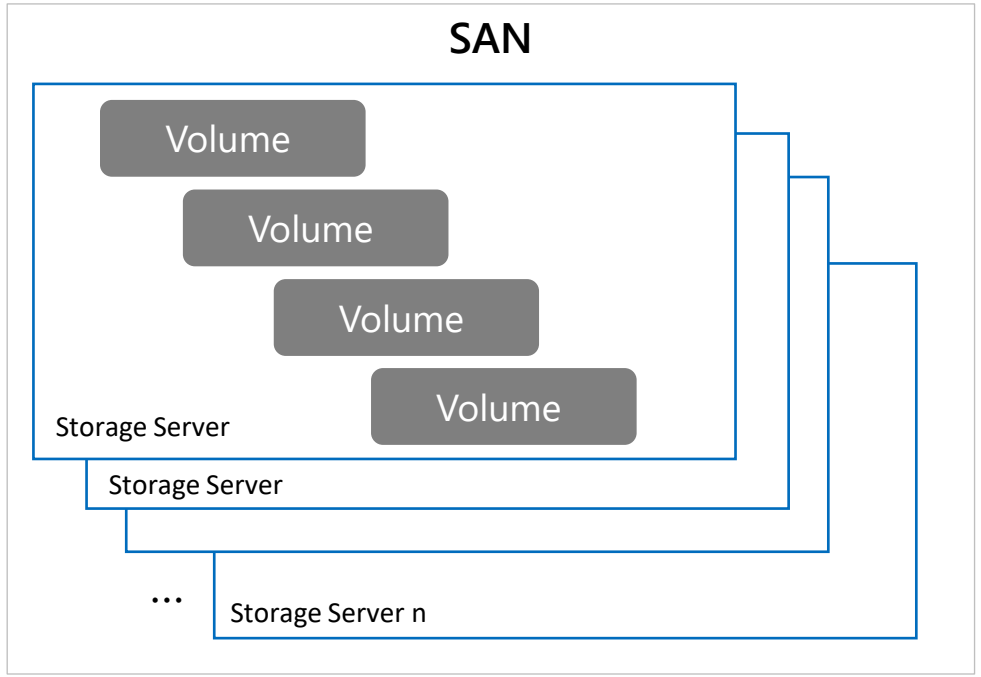
What is a traditional SAN?

- Pool of capacity and IOPS for a single group, accessible to any workload
- “Group” might be a company, department, or anything
- Workloads that shift within the “group” are interchangeable
- If the whole “group” is idle, we can do intelligent things like deduplication

Isn’t that a cloud?

- Clouds support multiple “groups”
- The goal is to ensure consistent and high resource utilization
- Shifting workloads trigger load balancing operations

Dynamic workloads want interchangeable storage

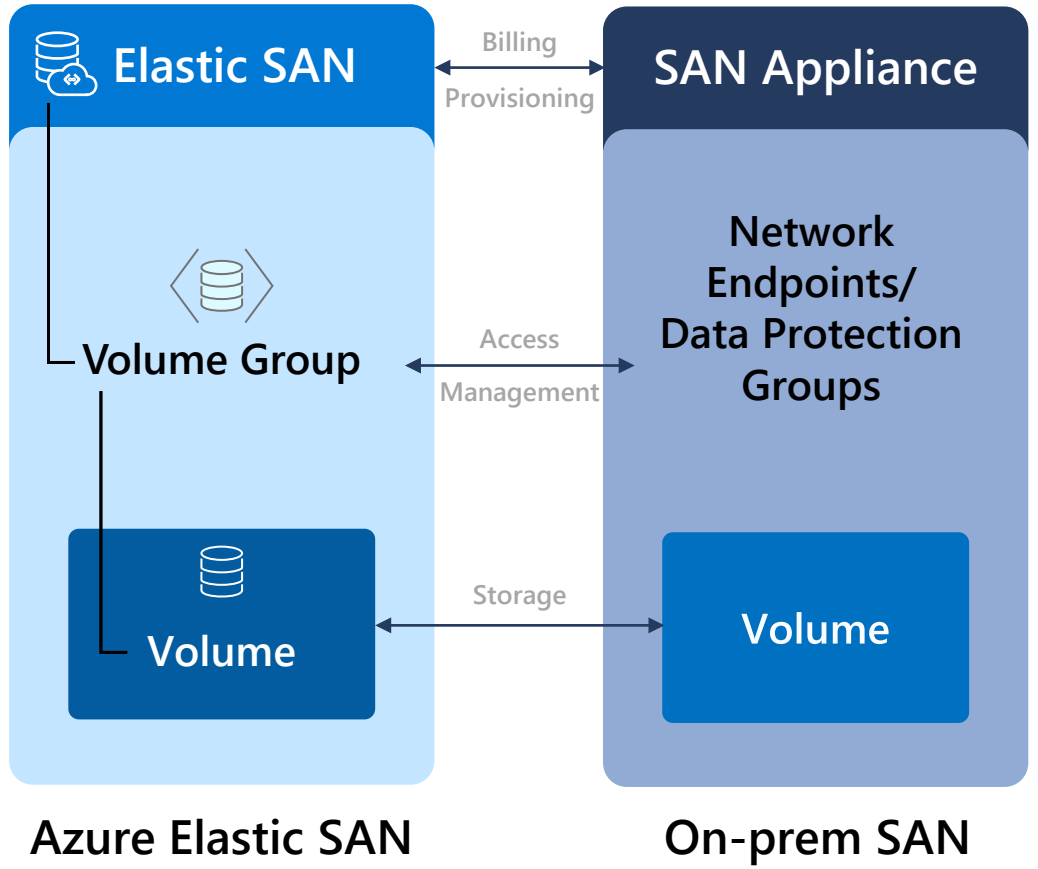


Introducing Azure Elastic SAN

A brand-new cloud native SAN solution

Industry's first fully managed SAN offering in the cloud

- Deploy, manage, and host workloads on Azure with an end-to-end experience similar to an on-premises SAN
- Bulk provision storage to achieve massive scale (millions of IOPS, double-digit GB/s)
- Simplified provisioning, scaling, and access management, with redundancy built in
- Support standard industry protocol (iSCSI) for data access
- **Rather than manage individual disks for each of your workloads, save time and money with a cloud-native SAN**



Elastic SAN

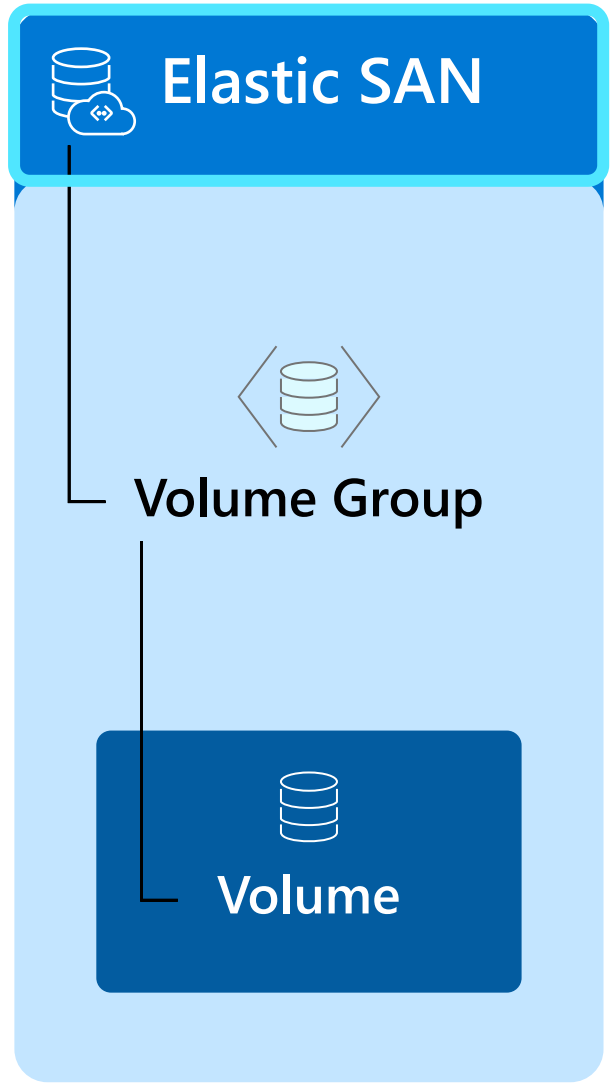
Provisioning resources and Billing

Two provisioning units: Base and Capacity-only

Billed on provisioned storage for capacity and performance

Operations include:

- Create Elastic SAN
- Update provisioned resources in base or capacity-only scale units
- Delete Elastic SAN

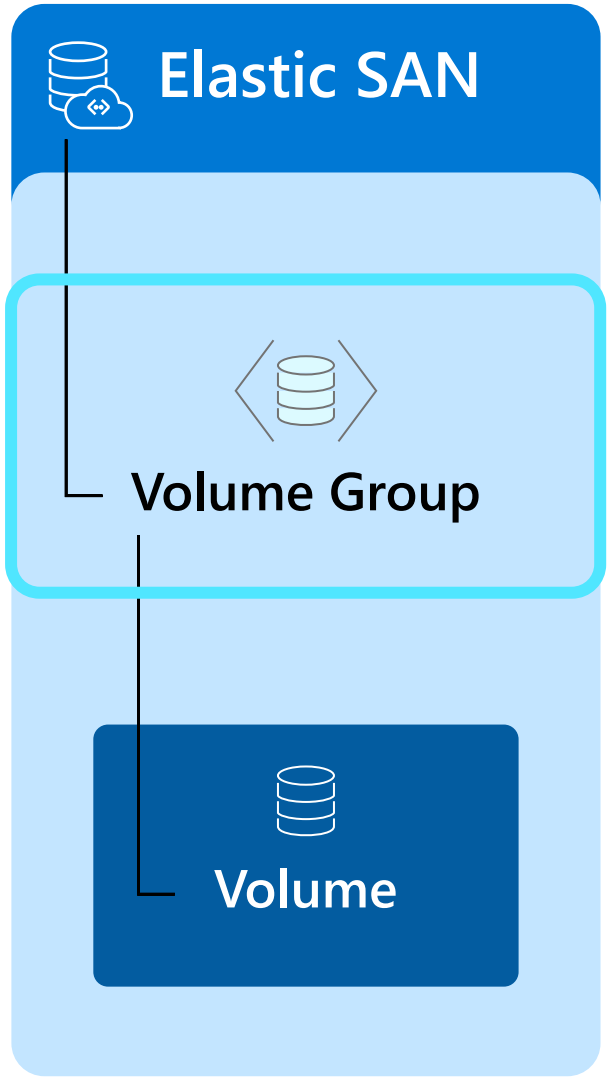


Volume Group

Applying security, encryption, data protection configurations

Configurations on the volume groups apply to all volumes within the group

A Elastic SAN can have up to 20 volume groups



Operations include:

- Create volume group
- Update network configurations
- Update encryption Settings
- Delete volume group

Volume

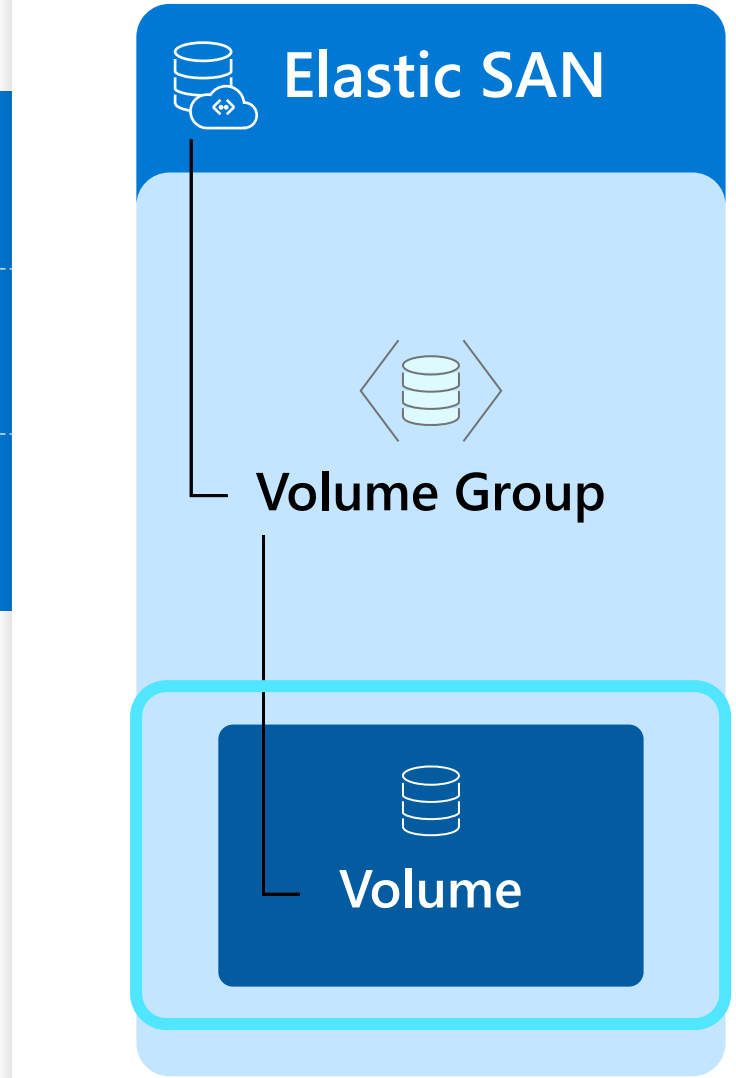
Data storage, the actual storage unit

Read/Write access over iSCSI

Dynamic pooling of provisioned performance targets

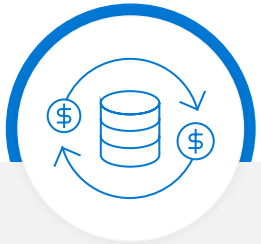
Operations include:

- Create volume
- Update volume size to scale up performance or capacity
- Delete volume

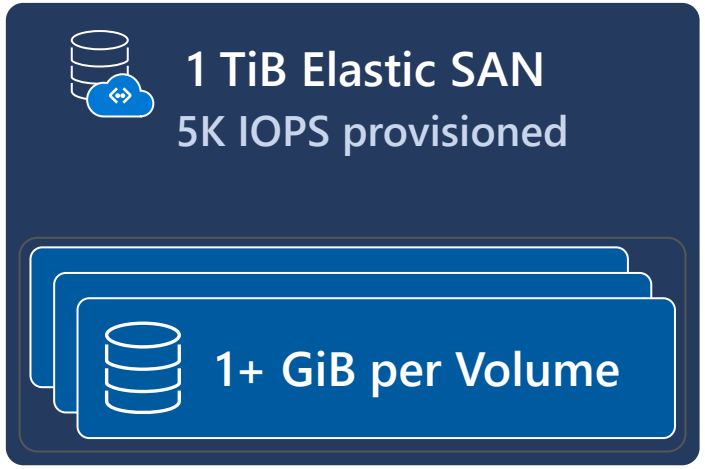


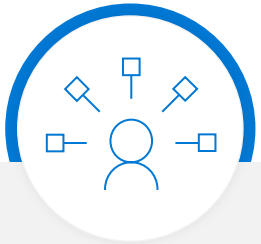
Azure Elastic SAN

Enabling container native scale

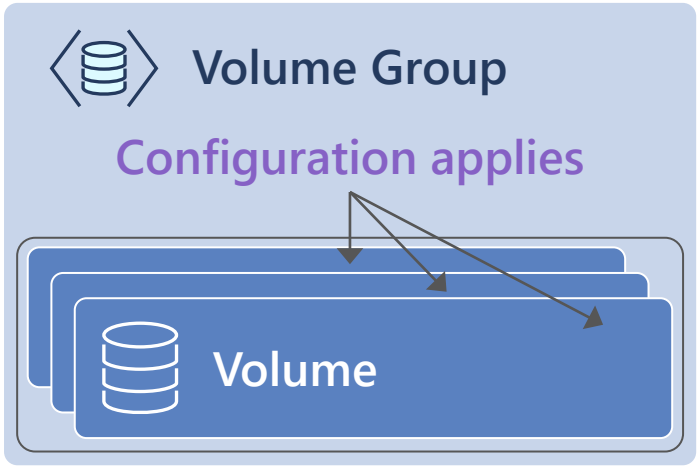


Share provisioned performance, achieve cost efficiency at scale

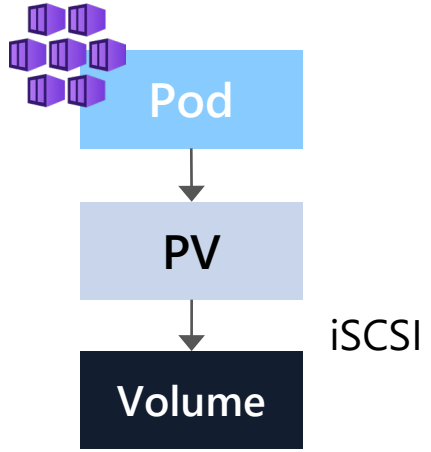




Simplify volume management through grouping



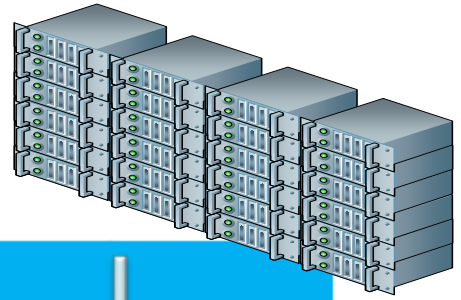
Integrate with Azure Container Storage via iSCSI



*preview expected later this year

Azure Elastic SAN deep dive

Inside Azure Storage (Recap)



Front-end layer

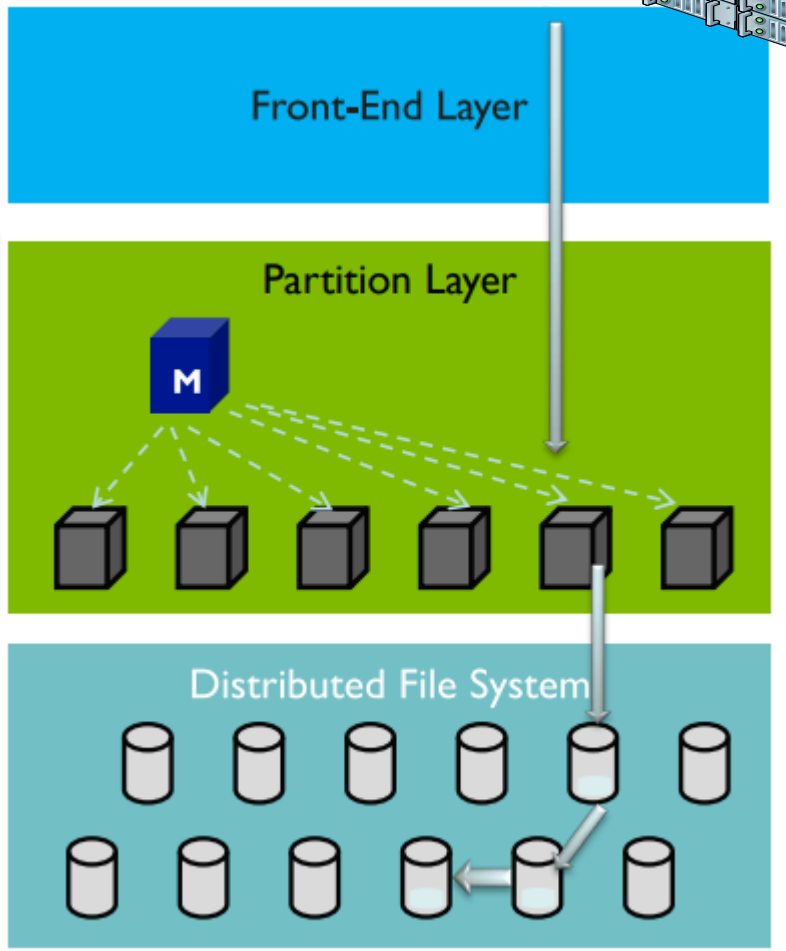
- Protocol endpoint
- Authentication/Authorization
- Metrics/logging

Partition layer

- Understands and manages our data abstractions
- Massively scalable key/value store
- Key ranges assigned to servers

Stream layer (Distributed File System)

- Data persistence and replication (JBOD)
- Append-only file system

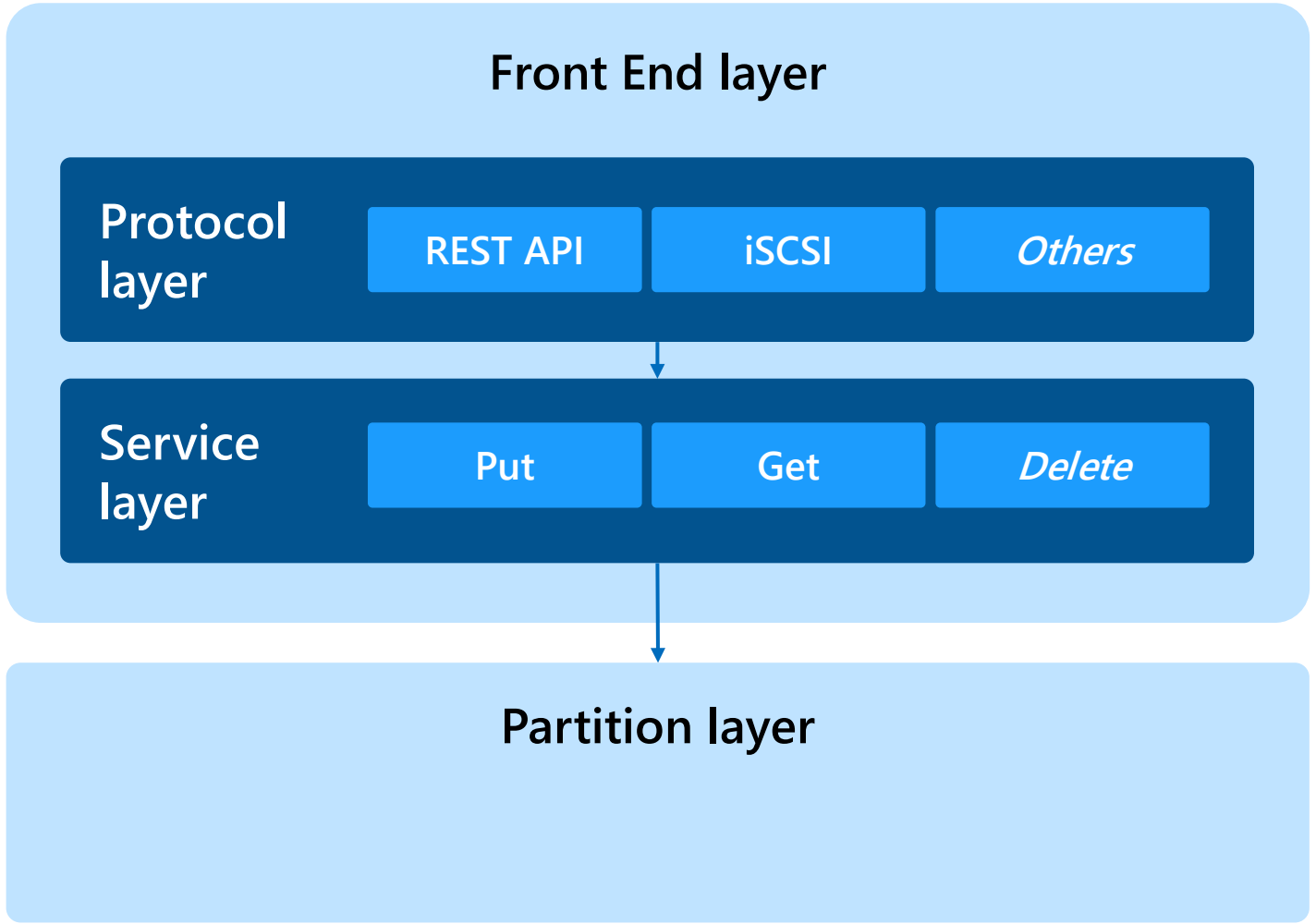


Azure Storage multi-protocol support

Protocol frontend, versioning

Authentication, throttling, logging

Storage operations business logic



Scaling with networked storage

Each network connection to a different front end

- Not limited by IOPS from a single storage server
- MPIO is well supported

Ubiquitous initiator support with iSCSI

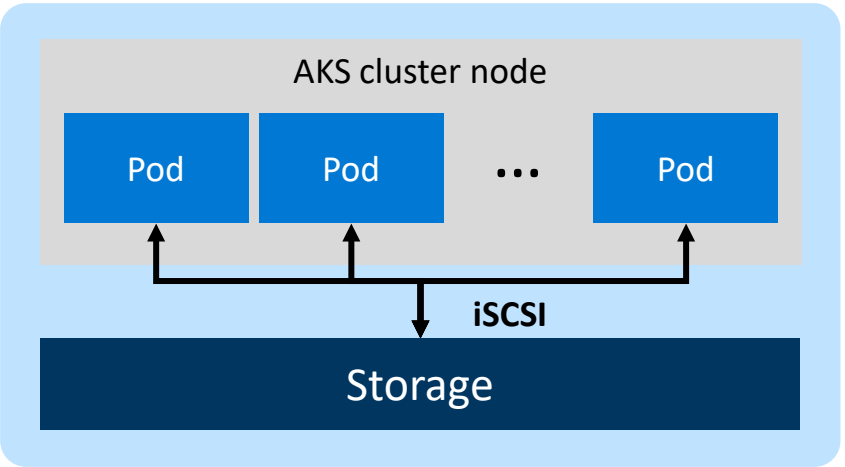
- Integrate with Linux and Windows nodes

Not limited by “local” SCSI bus under a VM

- Can have far more storage devices attached
- Important for a VM hosting many containers

Expansion to other protocols in future

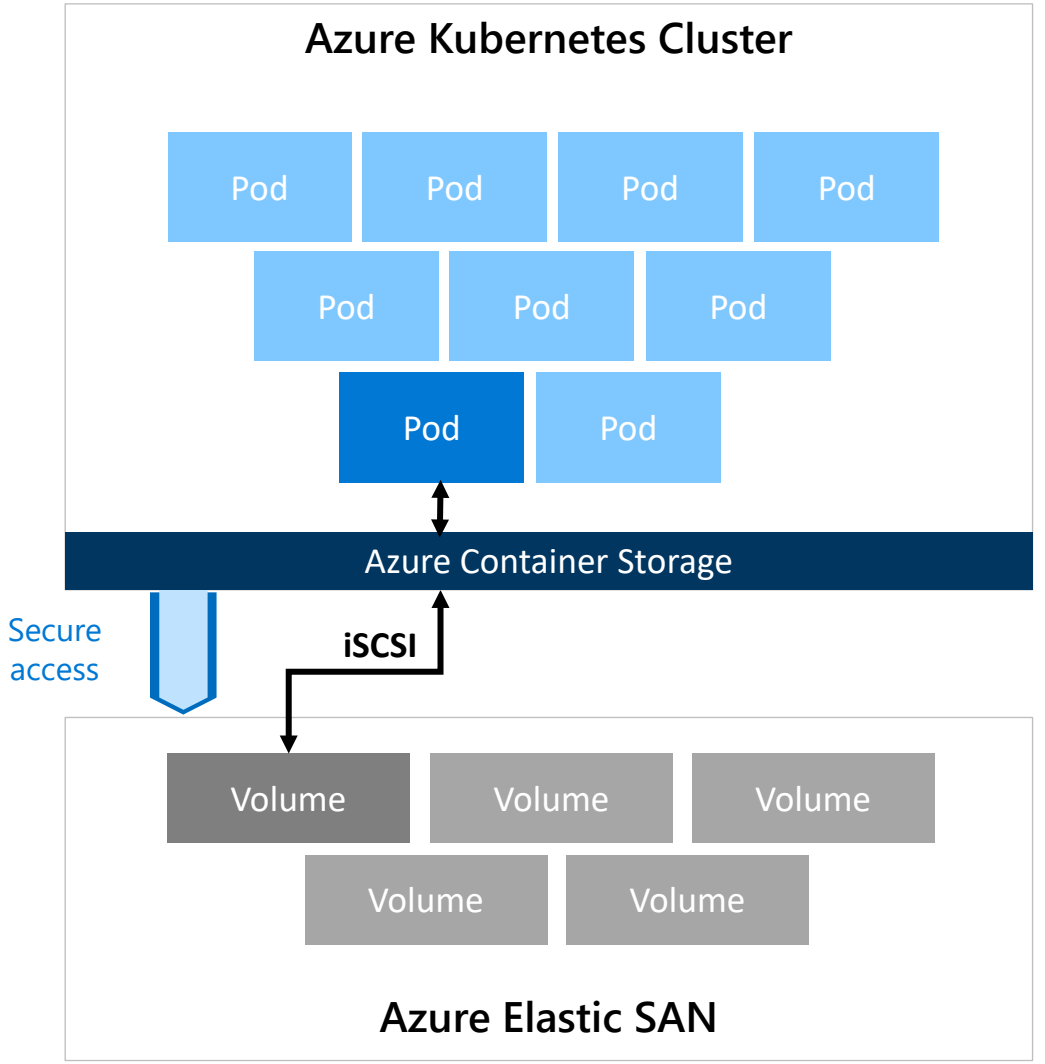
- Particularly for encryption-in-transit
- Talk to me about TLS-PSK



iSCSI target scaling with Elastic SAN

How do we scale containers with SAN?

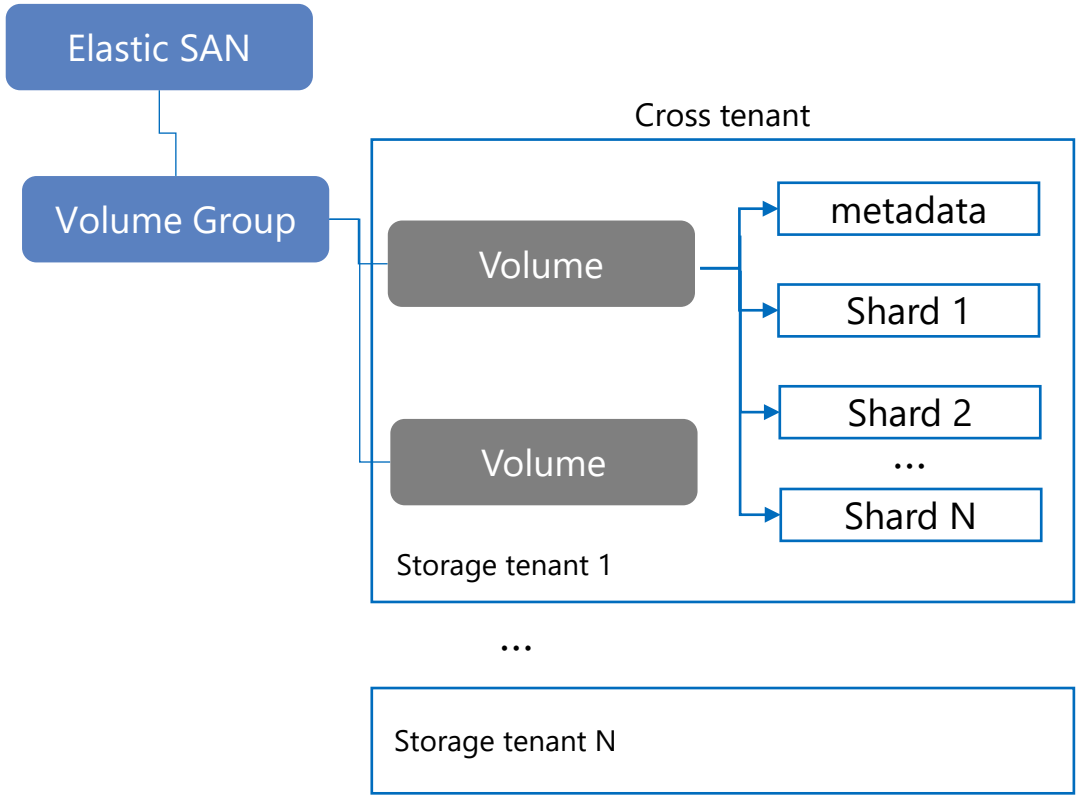
- Containers may have short lifetime. Network connections are relatively fast to establish and terminate
- From a user perspective, authentication counts as part of connection cost
- Administratively simpler to have a fleet of containers and a pool of storage
- Should be able to allocate IOPS and capacity based on demand



Making storage interchangeable

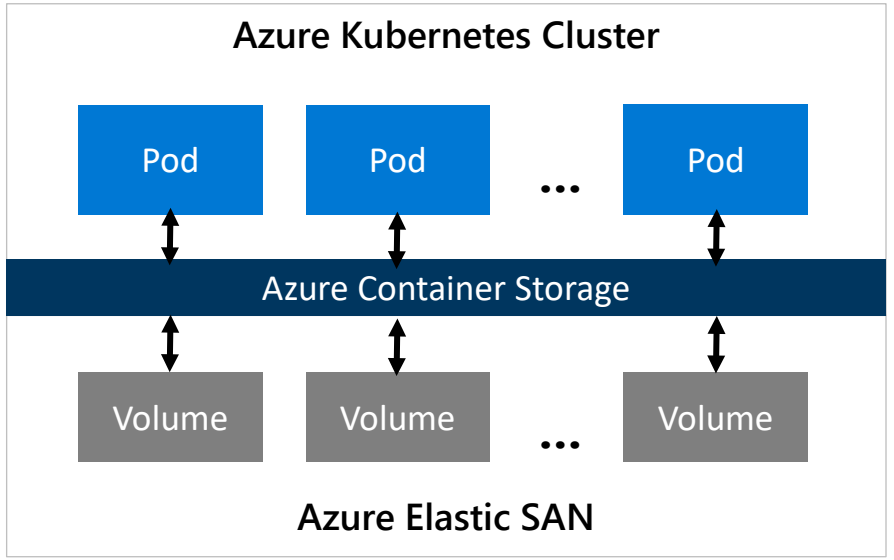
Solution: sharding!

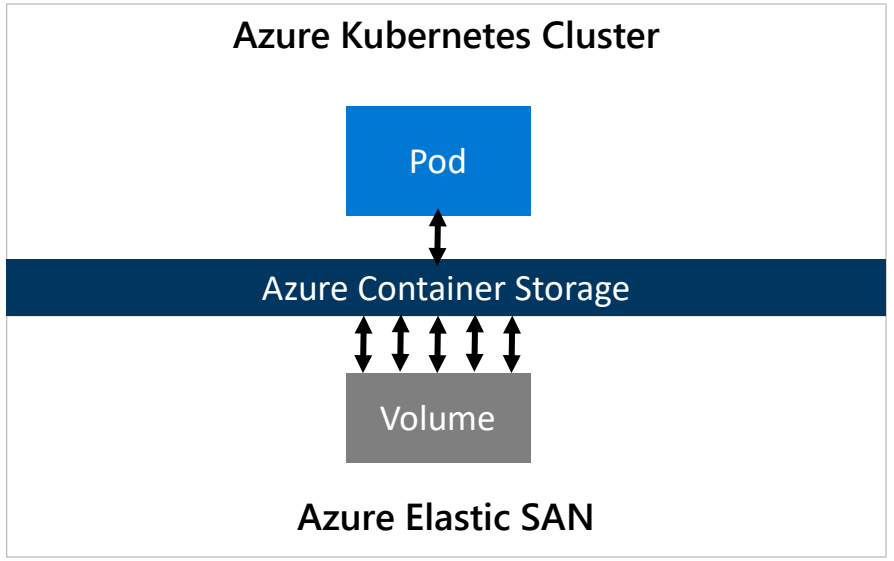
- If all storage exists on the same servers, shifting workloads won't shift loads
- A frontend protocol – iSCSI or NVMe – can redirect requests as needed
- Sharding normally creates challenges for global state – think snapshots
- But here we have a single frontend



I lied.

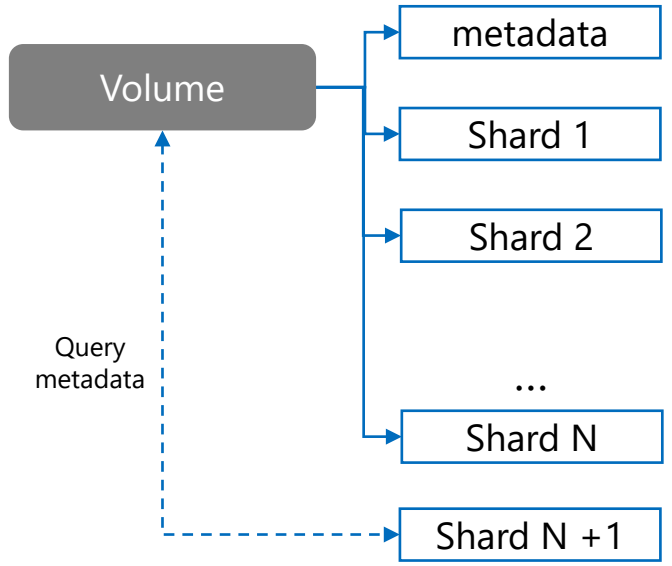
- Depending on workload, one TCP connection might not be enough.
- Scaling “out” – lots of workloads, one connection each
- Scaling “up” – many connections for one workload
- Unlike traditional SAN, each connection is managed by a different physical server
- But the total count remains manageable – it’s possible to coordinate





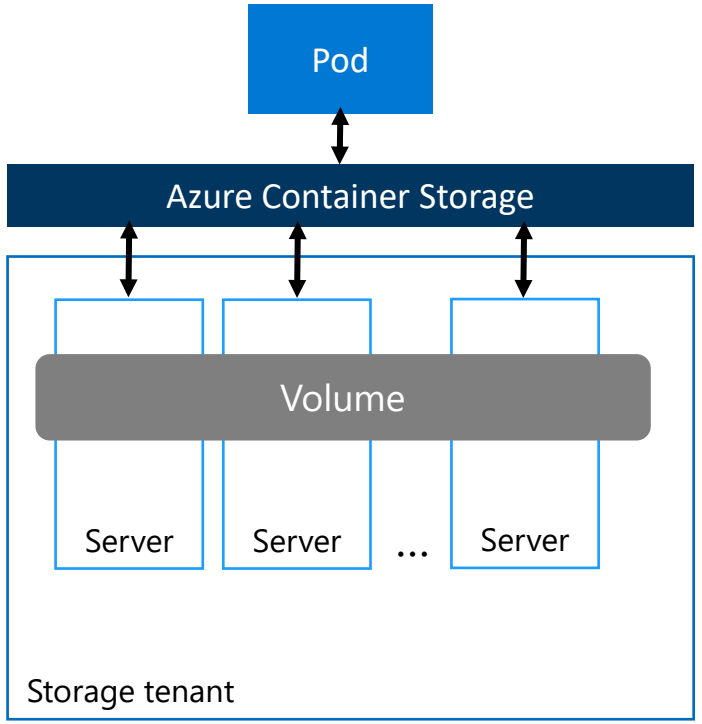
Shard creation and deletion

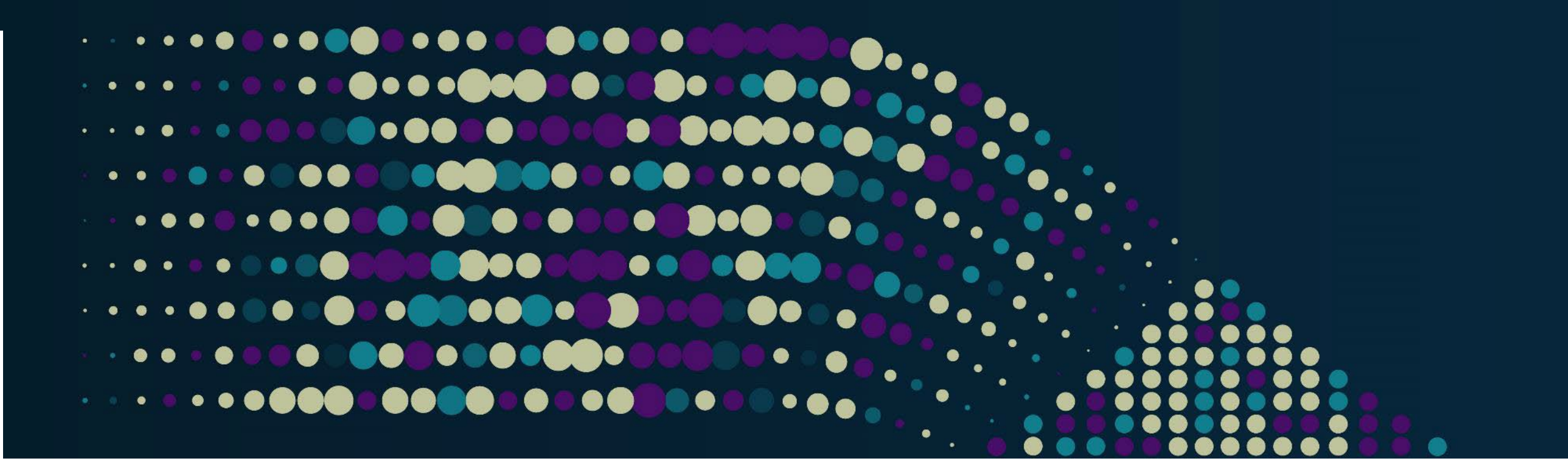
- Shards have defined names, distributed via hash
- Shards can be created on demand
- Passing all state to describe a shard on every write is expensive
- Instead, if a shard is being created, it can query state from the volume
- Shard deletion occurs via background scans
- If a volume is gone, shard records that it has expired



Per Pool scaling up

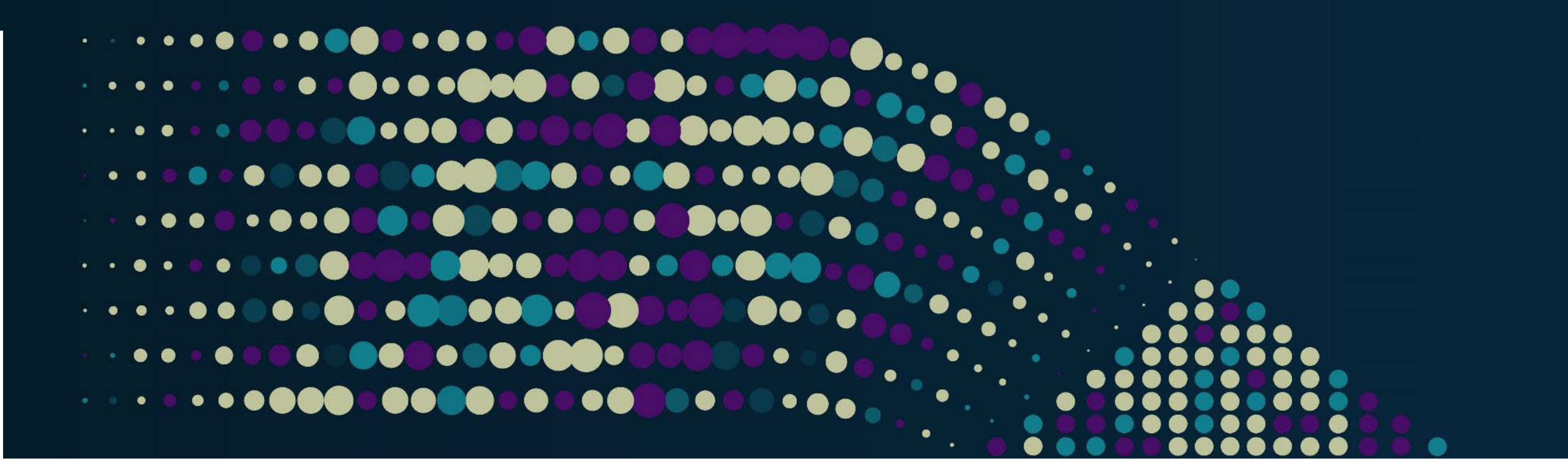
- Each volume can be created on a different tenant
- Each volume is physically stored across a set of storage servers
- Each connection can be served by a different server
- Result: millions of IOPS, double-digit GBps, single digit millisecond latencies





Thank You

Please reach out to us at askcontainerstorage @ microsoft.com



Please take a moment to rate this session.

Your feedback is important to us.

