VMware Cloud Director Object Storage Extension 2.1.1 – Reference Design

Table of Contents

Introduction

This guide provides information on how to properly design and deploy VMware Cloud Director Object Storage Extension on top of a VMware Cloud Director infrastructure. This document is specific to VMware Cloud Director Object Storage Extension 2.1.1 and its integration with Cloudian HyperStore, Dell EMC ECS, and AWS S3.

Information about how Object Storage Extension can utilize other S3-compatible storage through the Object Storage Interoperability Service (OSIS) can be found in a separate whitepaper.

Audience

This document is intended for VMware Cloud Provider architects and technical leads responsible for planning and executing the deployment and upgrades of a VMware-based cloud environment.

What is VMware Cloud Director Object Storage Extension?

The VMware Cloud Director Object Storage Extension (OSE) allows VMware Cloud Providers who are using VMware Cloud Director to offer object storage services to their customers. The extension acts as middleware which is tightly integrated with VMware Cloud Director to abstract third-party S3 API compatible storage providers in a multi-tenant fashion.

OSE runs externally to VMware Cloud Director and integrates through a UI plug-in, which shows either provider or tenant information, depending on the type of logged-in user.

OSE has a 1:1 relationship with VMware Cloud Director, which means that only one instance of OSE can be integrated with a single Cloud Director. OSE 2.1.1 is compatible with VMware Cloud Director version 10.0 and later and the Cloud Director Service.

An instance of VMware Cloud Director Object Storage Extension can work with a single instance of VMware Cloud Director or a single VMware Cloud Director server group.

Object Storage Extension can be connected to the following storage providers: Cloudian HyperStore, Dell EMC ECS, AWS S3, or another S3-compatible storage platform[1]. The provider can selectively enable VMware Cloud Director organizations to consume the service. The unique counterparts for organizations and users are created at the storage provider. The users authenticate to the service with VMware Cloud Director or S3 credentials and access it only through the UI plug-in. The provider can directly access the underlying storage appliance to set quotas or collect usage information for billing purposes.

Providers can switch between storage platforms with VMware Cloud Director Object Storage Extension but cannot use two different storage platforms simultaneously.

In addition to the storage platform that OSE will connect with Cloud Director, three or more (for high availability and scalability) RHEL/CentOS/Oracle Linux/Ubuntu/Debian/Photon VM nodes that run OSE, provided as an RPM or DEB package, are required. The number of the OSE VM nodes depends on the used S3 storage and the OSE use case. See for reference: *Deployment Options*. These VMs are essentially stateless and persist all their data in PostgreSQL DB version from 10. x to 12.x. This could be VMware Cloud Director external PostgreSQL DB (if available) or a dedicated database for VMware Cloud Director Object Storage Extension depending on the OSE use case.

VMware Cloud Director Object Storage Extension (OSE) enables Cloud Director tenant users to use object storage by native UI experience and support S3 clients to consume the object storage by S3 APIs.

To connect Cloud Director with the selected S3 object storage platform, OSE uses the following user mapping:

- VMware Cloud Director service provider is mapped to an ECS/Cloudian admin user, or AWS management account.
- VMware Cloud Director tenant is mapped to an ECS namespace, Cloudian group, or AWS org unit.
- VMware Cloud Director user is mapped to an ECS/Cloudian user, or AWS IAM user.

---

1 S3-compatible storage can be connected to Cloud Director through the Object Storage Interoperability Service (OSIS).

In addition to storing unstructured objects, vApps, and catalogs, OSE 2.1.1 also supports Kubernetes cluster protection. It helps tenants keep a backup of their critical Kubernetes clusters to S3 storage, for example, in case of accidental removal of a namespace or they need it for debugging or staging. The backup can also be restored to replace the corrupted Kubernetes cluster information.

Use Cases

VMware Cloud Director natively provides Infrastructure as a Service (Iaas) by integrating with the underlying VMware vSphere platform. All native storage services such as storage for virtual machines, named (independent) disks, and catalog storage for virtual machine templates and media are using storage attached to vSphere ESXi hosts such as block storage, NFS, or VMware vSAN.

There is, however, the need for highly scalable, durable, and network-accessible storage that could be utilized by tenants or their workloads without the dependency on the vSphere layer. The VMware Cloud Director Object Storage Extension (OSE) provides access to the object storage either through VMware Cloud Director UI extension or via standardized S3 APIs. This allows existing applications to easily access this new type of storage for various use cases.

Storing Unstructured Data

Through the VMware Cloud Director User Interface, users can create storage buckets and upload and tag unstructured files (objects) of various types. These files can be easily accessed with Uniform Resource Locator (URL) links or directly previewed from the OSE plug-in. For protection, versioning and object lock can be applied to the S3 bucket objects. Archived objects in AWS S3 buckets can also be restored, which is basically changing their status from archived to frequently accessed objects to view their content. The objects of Cloudian buckets can also be replicated across data centers by setting up an org-level storage policy or changing it individually per a tenant.

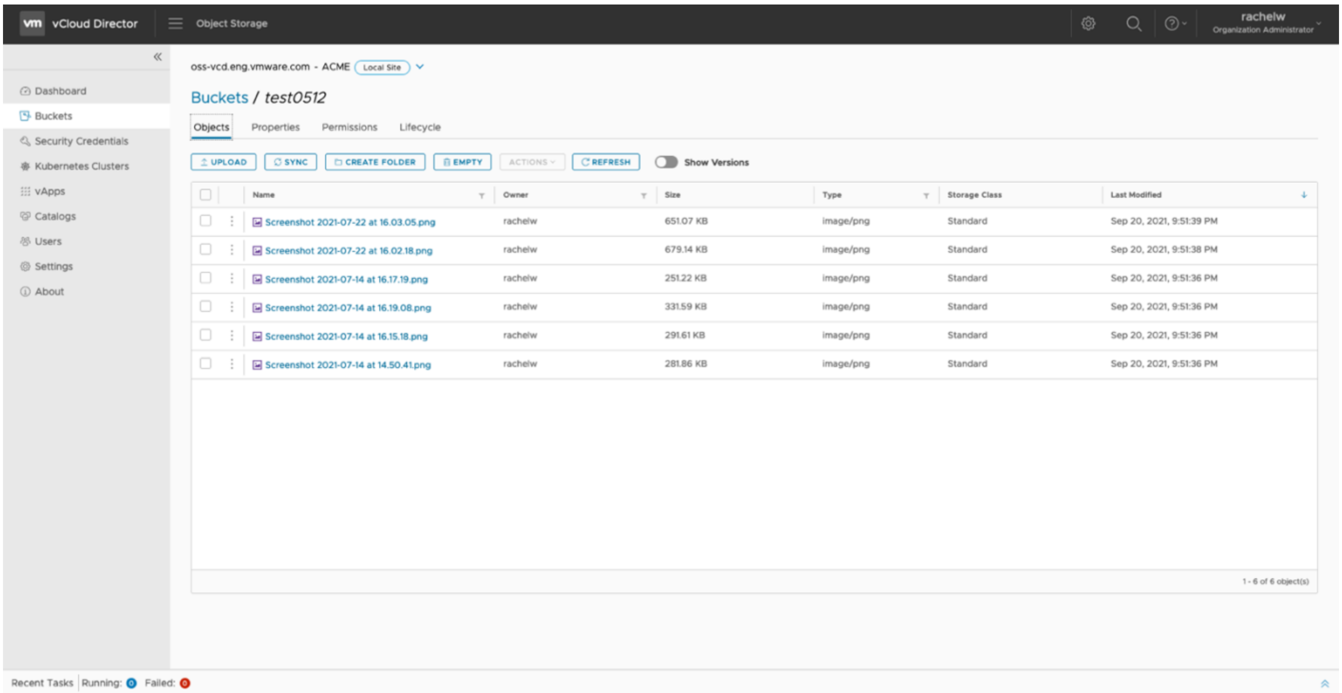

Figure 1. Object Upload to S3 Bucket

Thanks to the OSE full S3 API compatibility, it is also possible to utilize existing 3rd party applications to upload and manage the files of a bucket. The following figure displays how to set up the connection to the S3 storage with a freeware S3 browser.

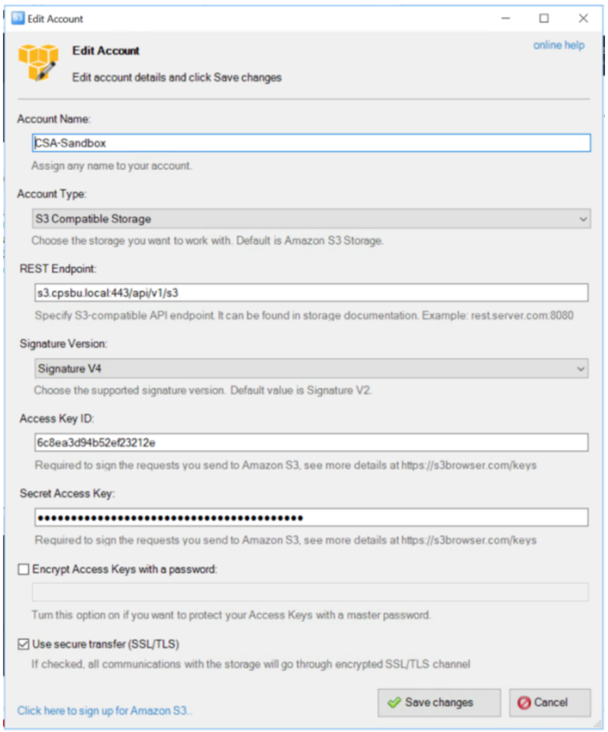

Figure 2. S3 Browser Configuration

Bucket permissions can be managed either through defining their Access Control Lists or by creating bucket policies. In OSE 2.1.1, bucket objects can be synced on an org level with the connected S3 object storage.

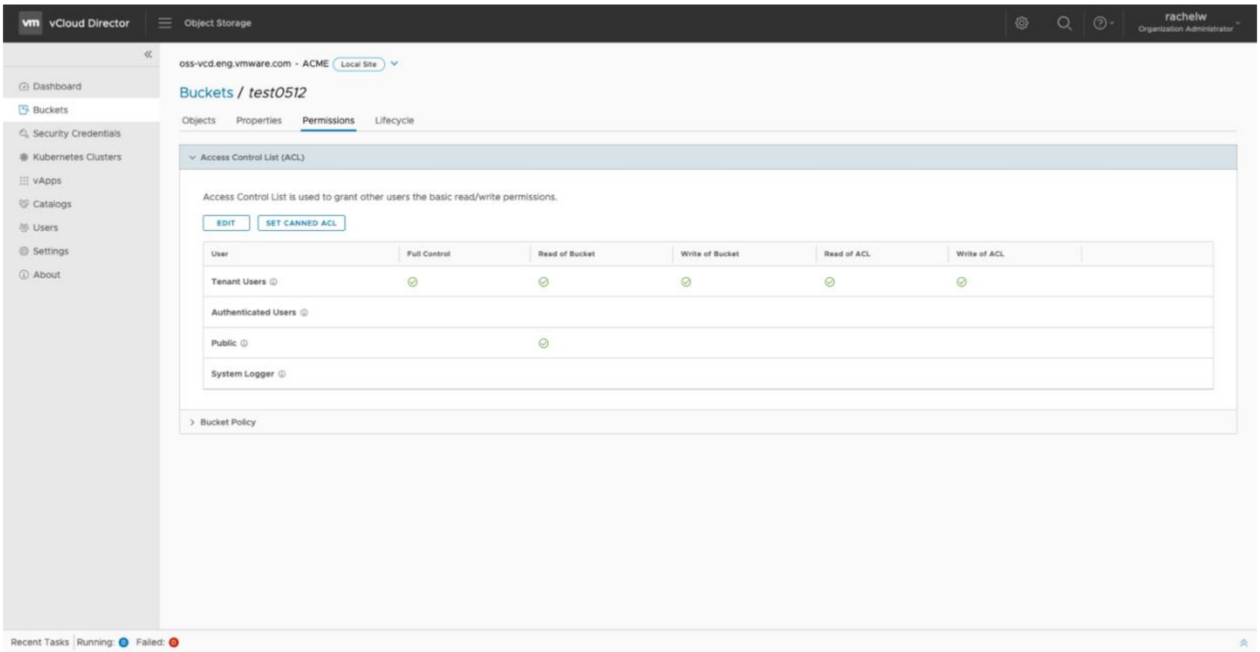

Figure 3. OSE ACL List

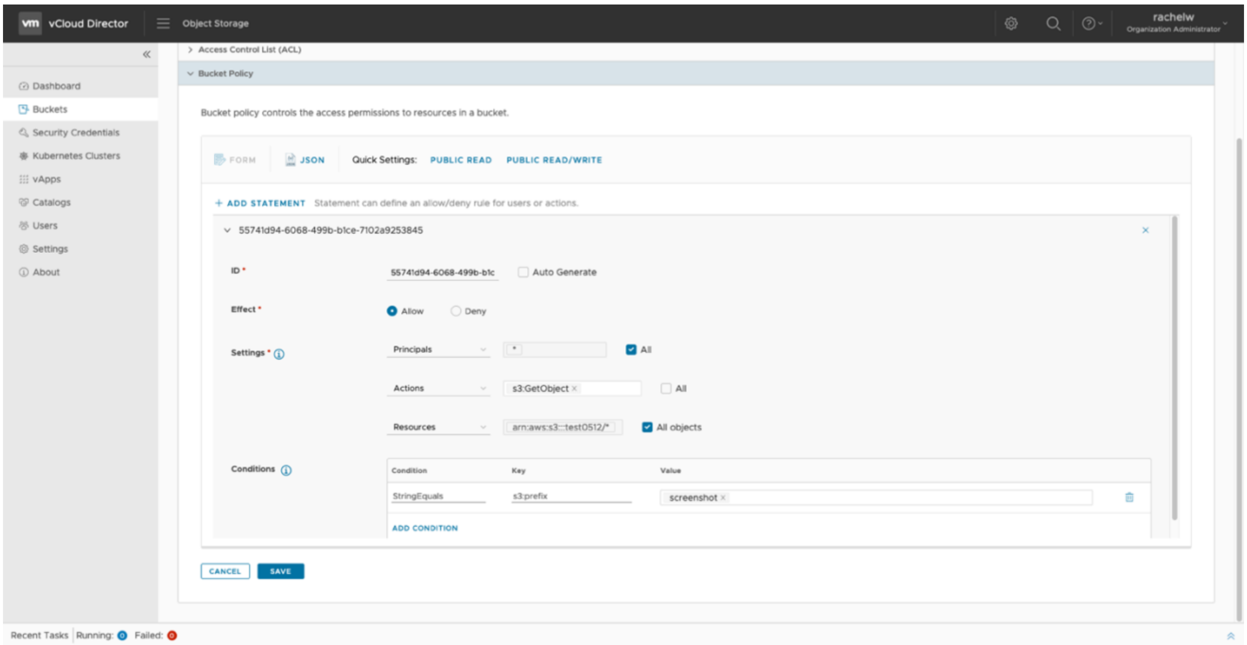

Figure 4: Bucket Policy

Bucket objects can also be tagged, and their logs can be kept in another S3 bucket.

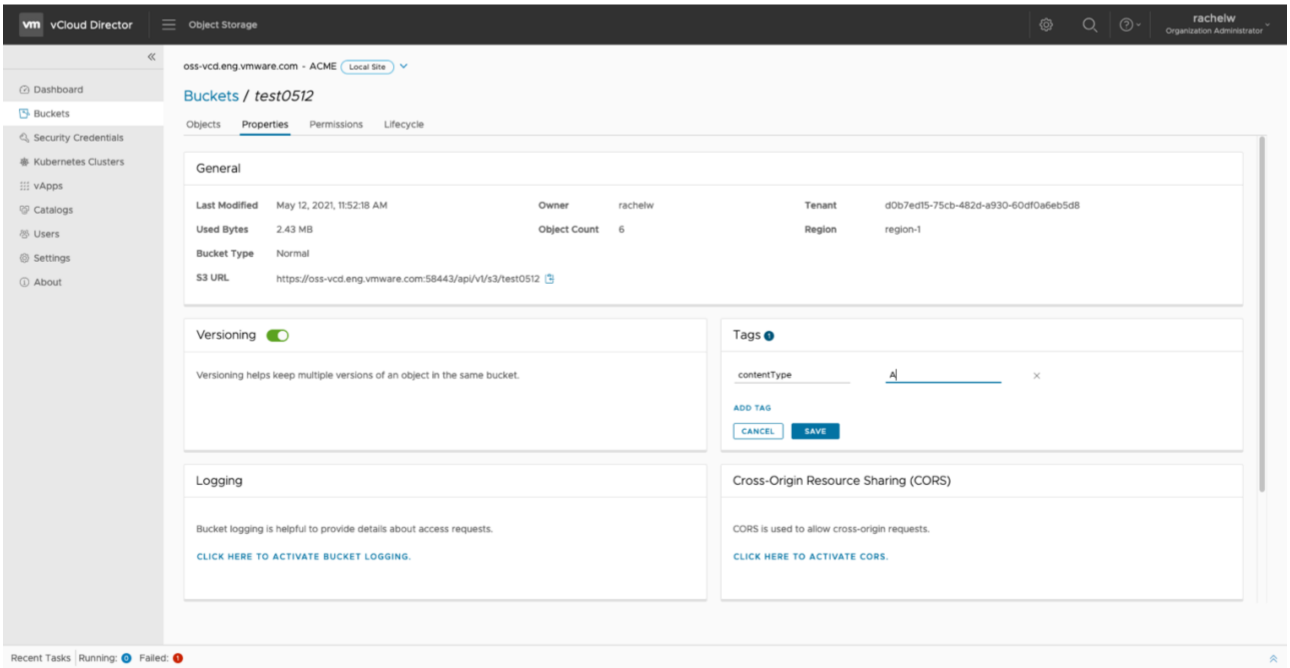

Figure 5: Bucket Properties

In addition, you can manage the lifecycle of the bucket objects by setting the period for which the objects will appear in the bucket before being automatically deleted.

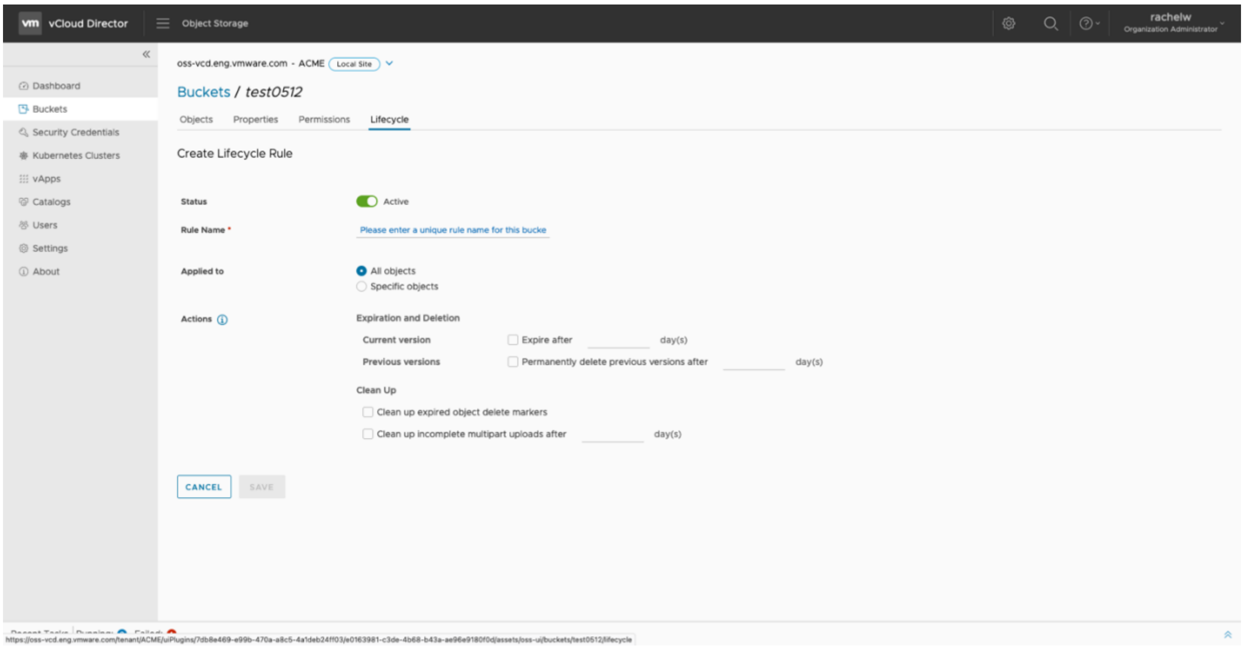

Figure 6: Bucket Lifecycle

Server-side tenant-level encryption of bucket content is also possible with OSE. However, it is only applied to new objects.

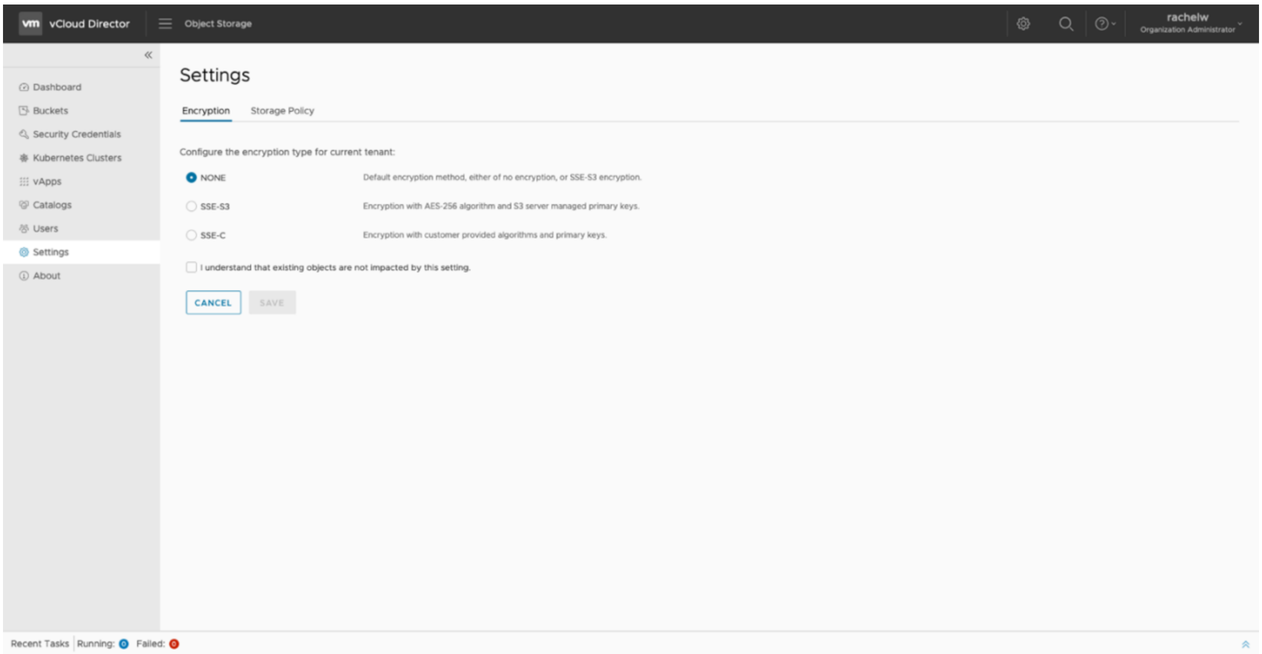

Figure 7. Server-side Tenant Level Encryption

Persistent Storage for Application

Users can create application credentials with limited access to a specific bucket. This allows (stateless) applications running in VMware Cloud Director (or outside) to persist their content such as configurations, logs, or static data (web servers) into the object store. The application is using S3 API over the Internet to upload and retrieve object data.

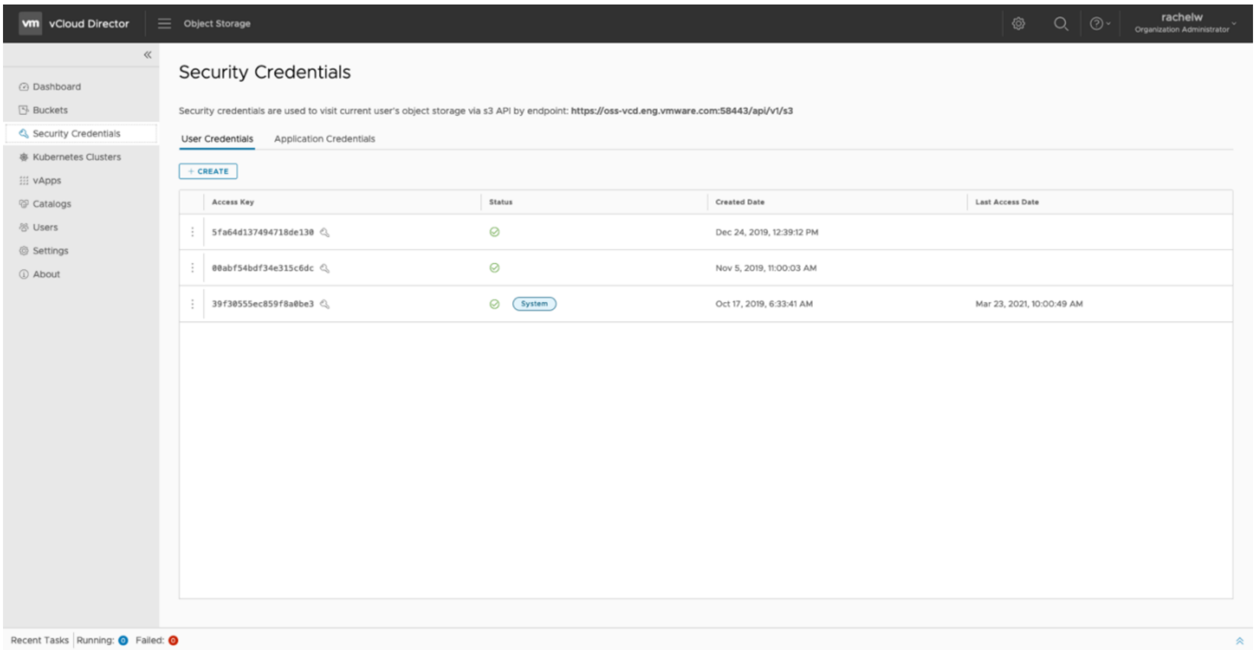

Figure 8. Application Credentials

Storing vApp Templates and Catalog

Because of the close integration with VMware Cloud Director, VMware Cloud Director Object Storage Extension can directly capture and restore a user's VMware Cloud Director vApps. Users can also share these vApps with other users. Thus, VMware Cloud Director Object Storage Extension provides an additional tier of storage for vApp templates that can be used, for example, for archiving old images.

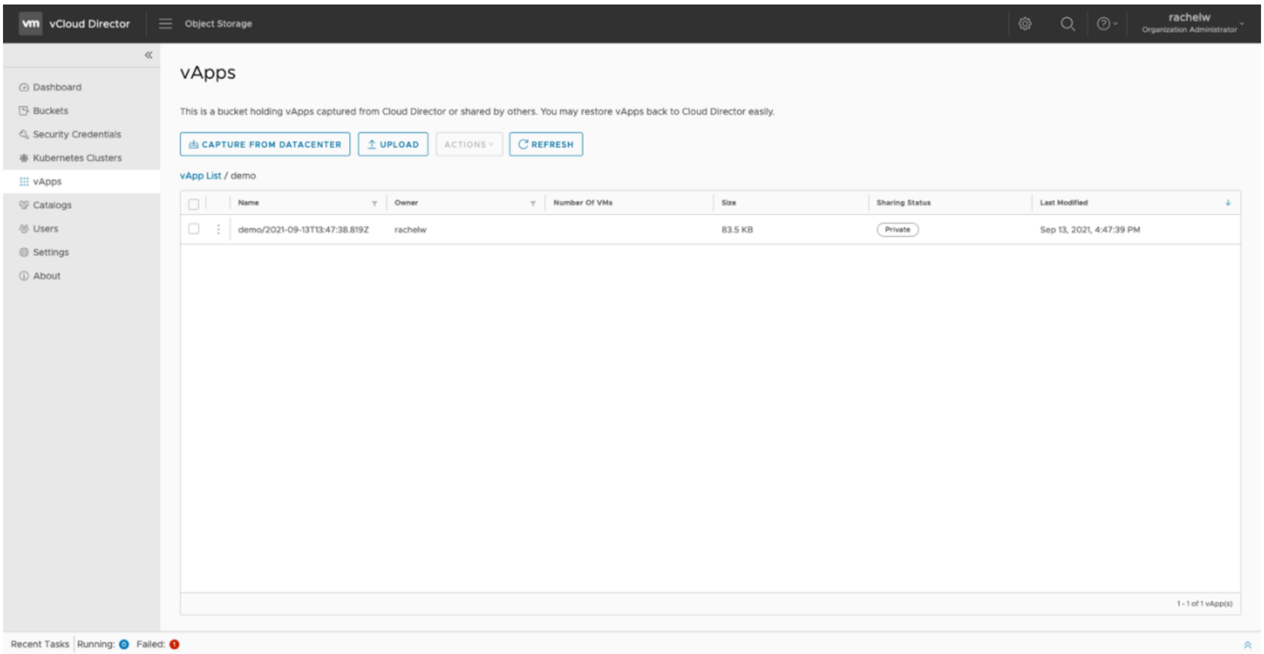

Figure 9. vApp Template Integration

An entire VMware Cloud Director catalog (consisting of vApp templates and media ISO images) can be captured from an existing Org VCD catalog or created from scratch by uploading an individual ISO and OVA files to VMware Cloud Director Object Storage Extension. Then, the catalog can be published, which allows any VMware Cloud Director organization (from any VMware Cloud Director instance) to subscribe to the catalog. As a result, this OSE functionality enables easy distribution of specific catalogs publicly or geographically across VMware Cloud Director instances.

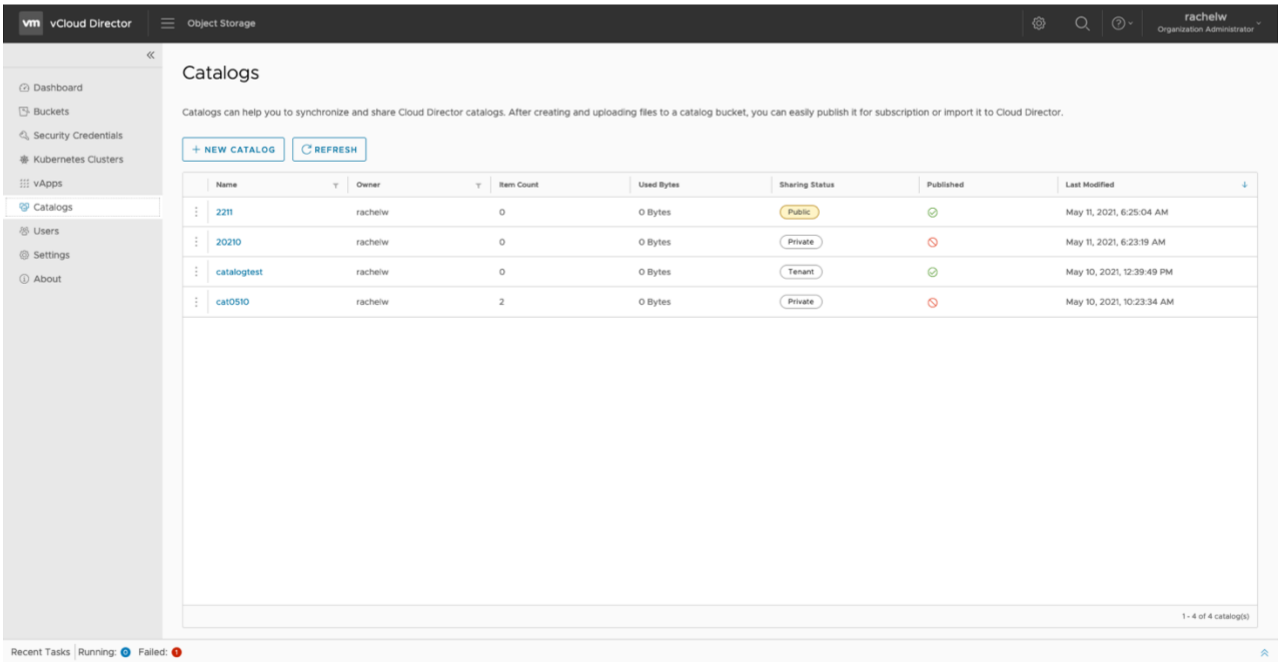

Figure 10. Catalog Integration

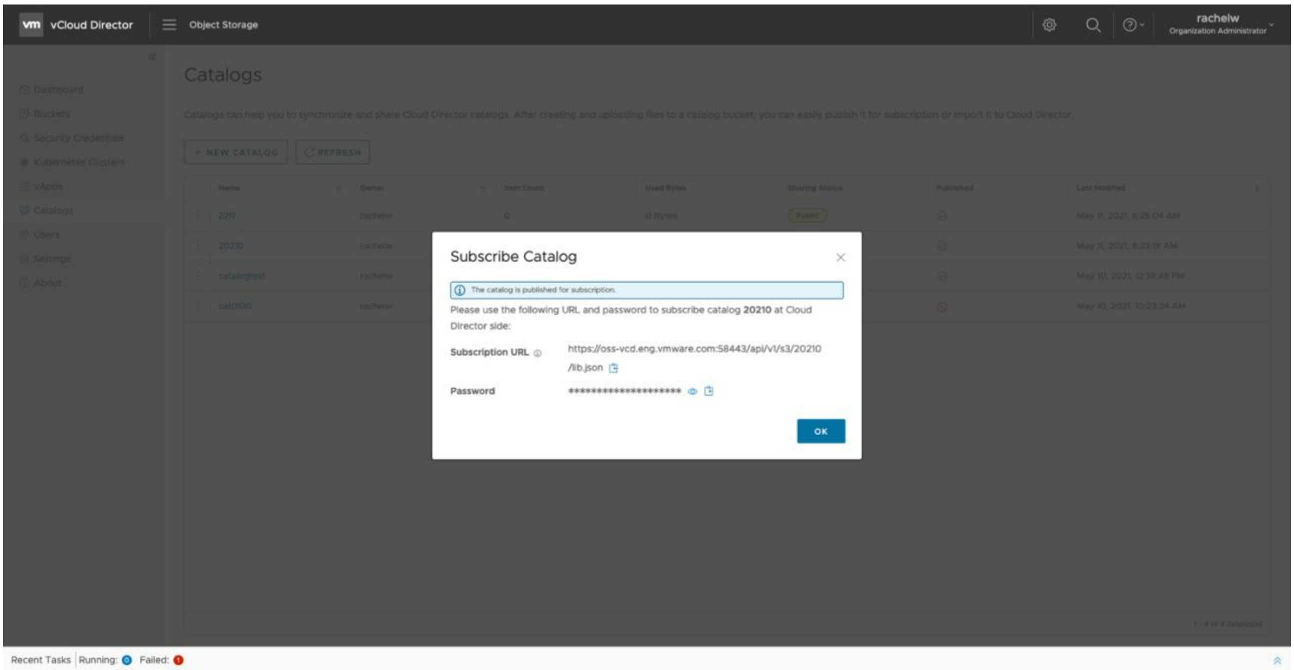

Figure 11. Catalog Published Directly from Object Storage Extension

Kubernetes Cluster Protection

In OSE 2.1.1, Kubernetes cluster backups complement the storage of unstructured data, vApps, and catalogs. With the Kubernetes cluster protection, tenants can back up their critical Kubernetes clusters and revert to the backups in case of accidental removal of namespaces or a Kubernetes upgrade failure. Tenants can also use the Kubernetes cluster backup to replicate the cluster for debugging, development and staging before rolling their app out in production.

The Kubernetes clusters that can be protected in OSE 2.1.1 include CSE native, TKG, and external Kubernetes clusters.

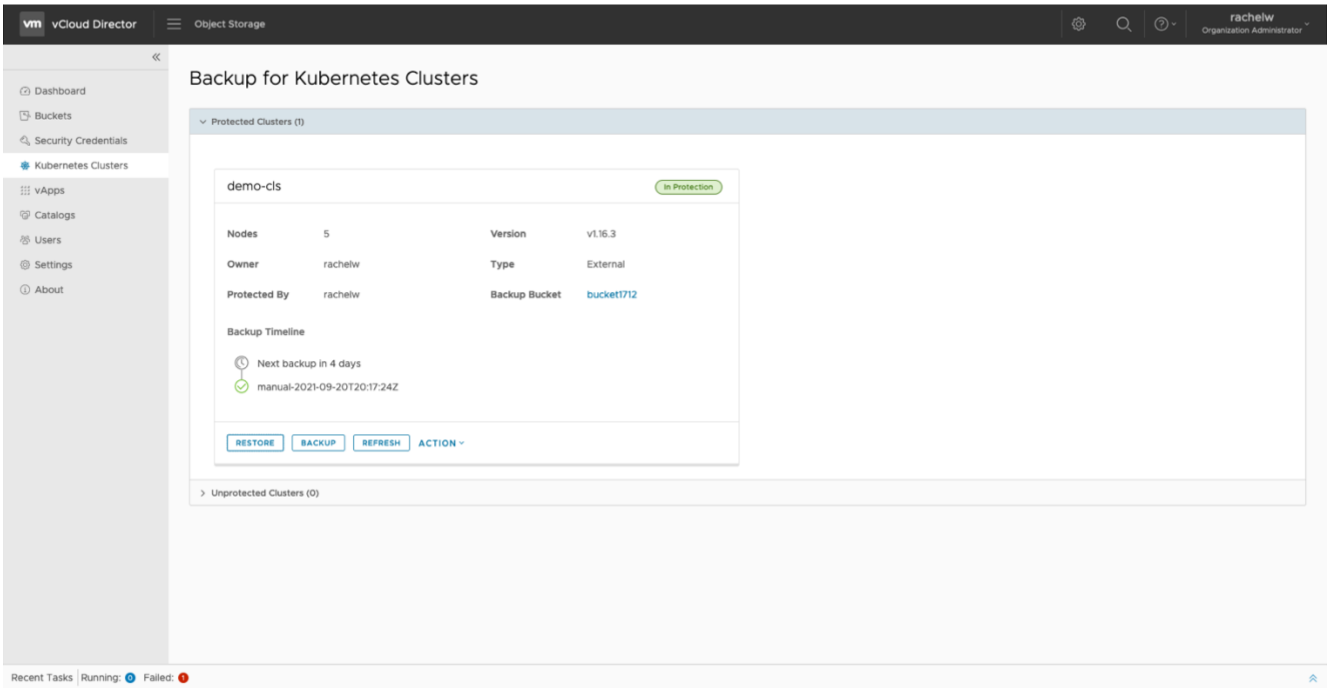

Figure 12: Backup of a Kubernetes Cluster

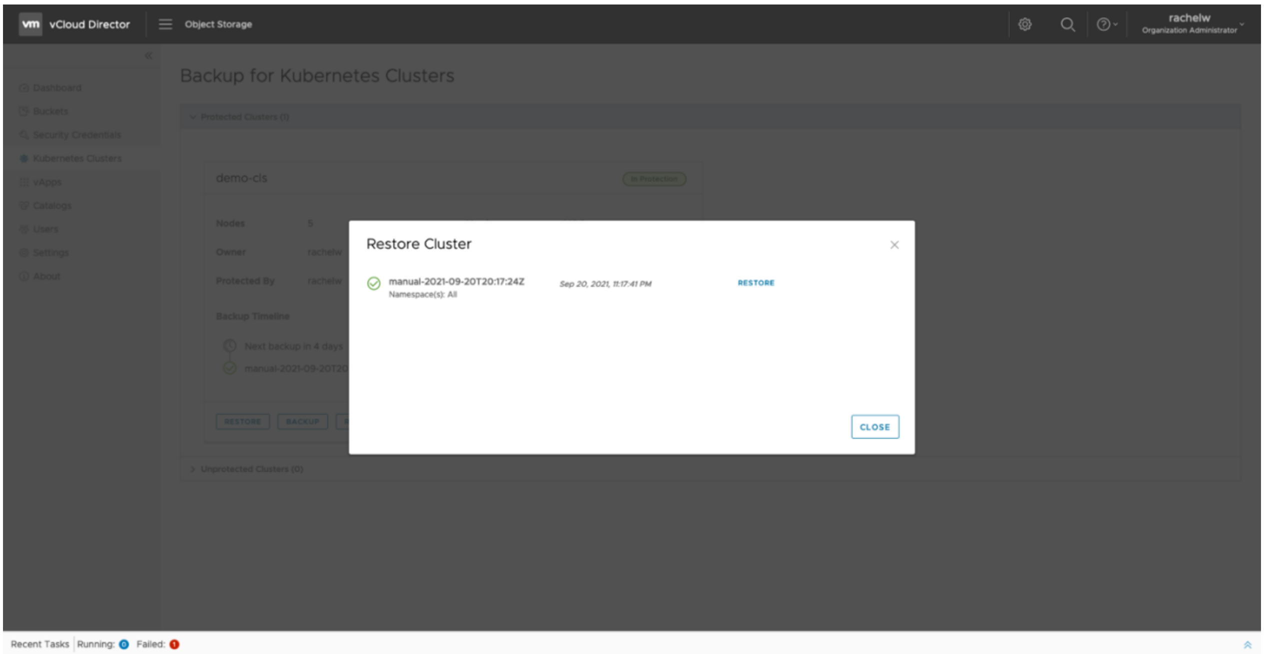

Figure 13: Restore of a Kubernetes Cluster

OSE 2.1.1 Architecture

OSE is a standalone server running on a Linux machine and multi-node deployment. It exposes SSL port 443 as the public endpoint. Both OSE UI plugins and S3 client applications connect to OSE APIs on this port. OSE supports S3-compliant XML APIs and Amazon Signature V4 authentication. It's primarily compatible with any S3 compliant clients.

OSE connects to Cloud Director and the object storage cluster from the backend. OSE makes REST API calls to Cloud Director for tenant and user mapping for object storage. It also supports object storage-backed catalog contents and vApp backups. OSE connects to the object storage cluster for tenancy management and data transfer. Depending on the type of object storage cluster, there could be one port or multiple ports for the communication between OSE and the object storage cluster.

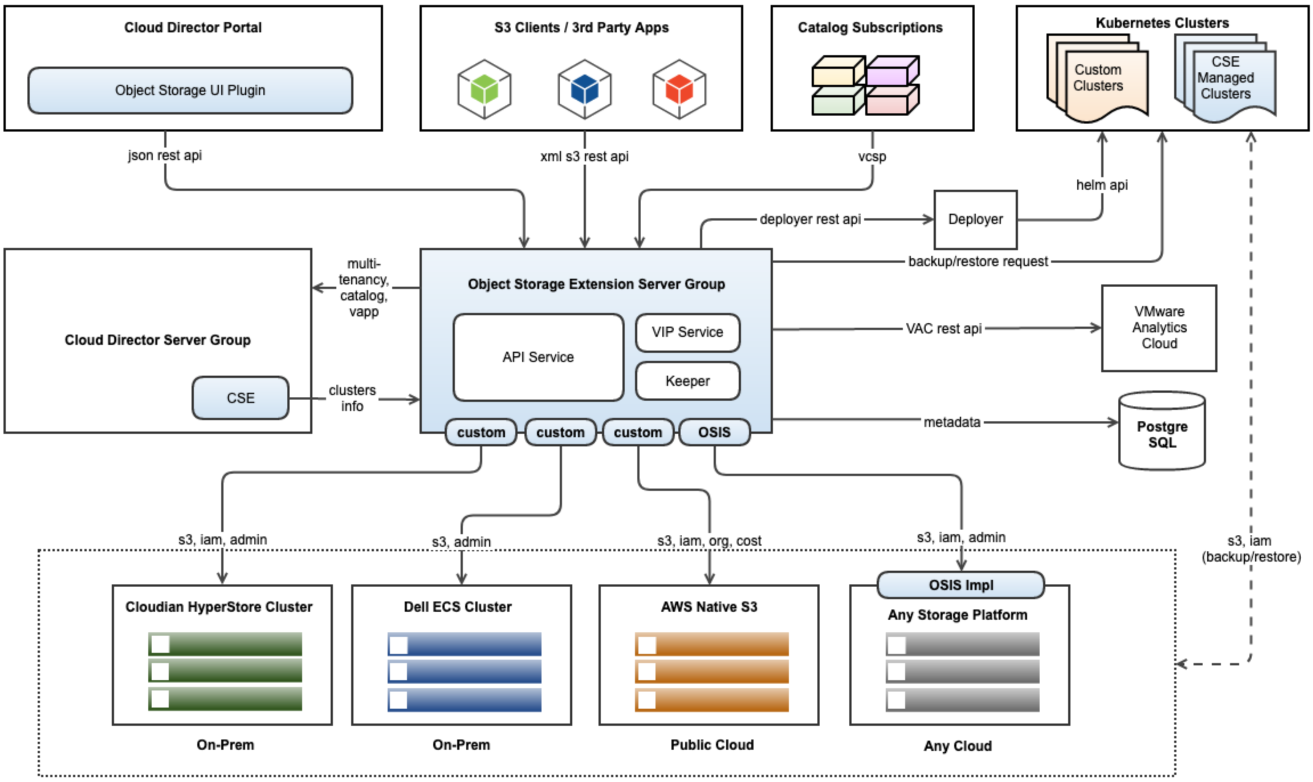

Figure 14: OSE 2.1.1 Architecture View

OSE uses S3 API to make queries to the underlying S3 storage vendor and user identity and access management service to map Cloud Director user types with those of the connected storage.

OSE uses a PostgreSQL database to store metadata. All management data, bucket metadata, and object metadata are stored in the database. If your object storage solution is for internal use or a small business, you can consider re-using Cloud Director's PostgreSQL appliance. For a standard deployment, you should consider deploying a standalone PostgreSQL server for OSE.

The bandwidth consumption between OSE and the object storage cluster is much higher than the communication between OSE and Cloud Director, so you should consider deploying OSE server nodes into the network with as little latency as the communication with the storage cluster.

OSE also makes REST API calls to VMware Cloud Analytics to send product usage data. This part of the OSE architecture comes into play only if the tenants agree with the *VMware Customer Experience Improvement Program (CEIP)* in Cloud Director UI to allow VMware to collect data for analysis.

OSE also uses a Kubernetes agent called Velero to backup and restore Kubernetes clusters on the underlying S3 storage. This OSE feature uses a deployer that enables the Cloud Director tenants to perform Helm operations to external Kubernetes clusters.

OSE Catalogs use vSphere catalog synchronization protocol to sync with the content of the Cloud Director Catalogs.

For vApps, OSE uses REST API to export vApps from Cloud Director to the underlying S3 storage.

OSE 2.1.1 Component View

The following diagram shows the OSE 2.1.1 components. An application deployer is also part of the OSE 2.1.1 installation bundle and works as a peer to install the backup and recovery agent (Velero Server) in Kubernetes clusters. It installs and uninstalls the Helm chart of the Velero Server. It also queries the status of the deployed Velero Server.

At the same time, two embedded clients are added to OSE 2.1.1: a deployer client and a Velero client. The deployer client talks with the deployer to operate the backup and recovery agent. The Velero client is a wrapper of the Kubernetes Java client and manipulates the CRD objects of Velero to request backup/restore and queries the operation status.

The backup and recovery agent performs the backup/restore with the S3-compatible object storage.

Like OSE 2.1, version 2.1.1, sends data through REST API to VMware Analytics Cloud for a product usage analysis.

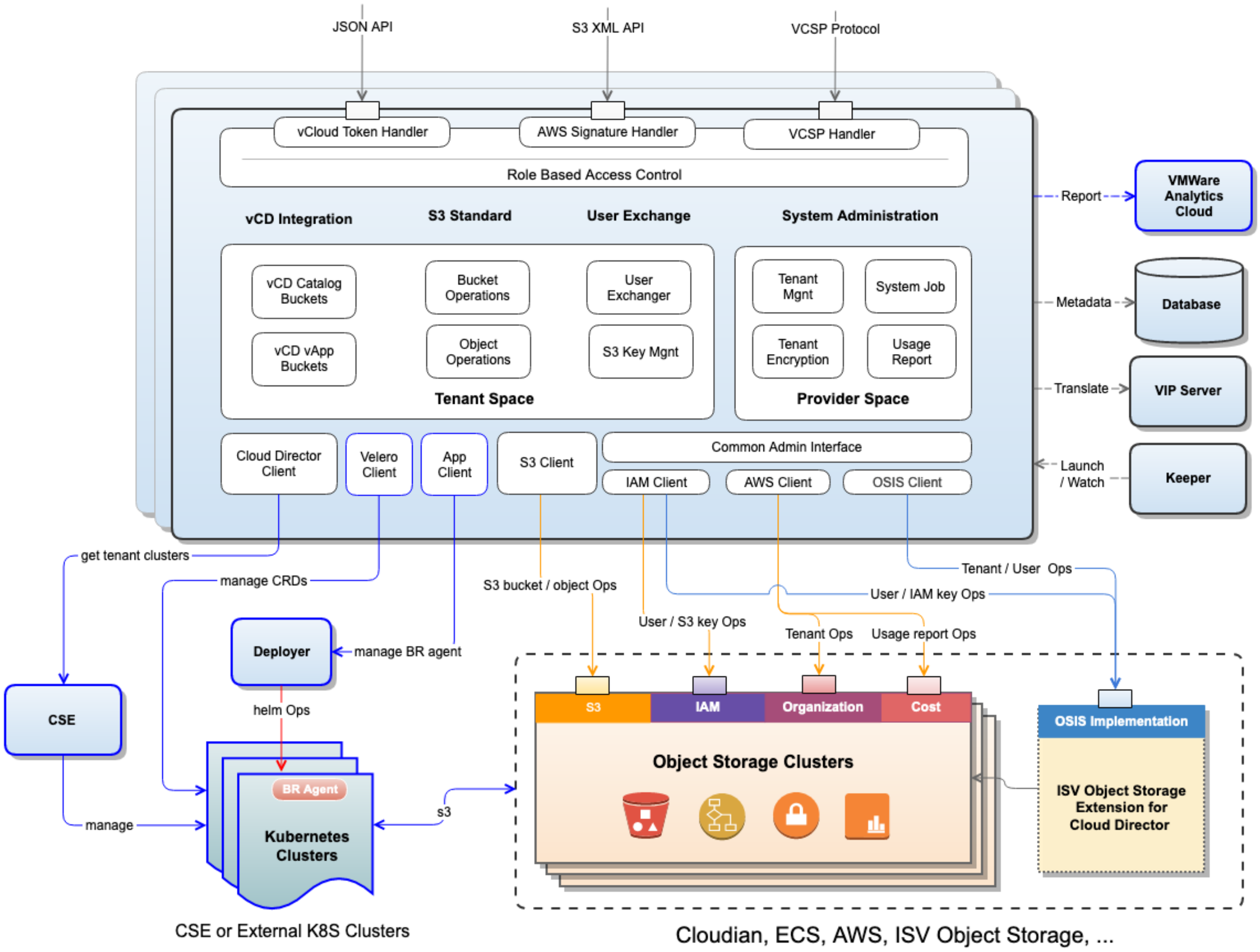

Figure 15: OSE 2.1.1 Component View

After OSE is installed, the following components are available on the hosting machine.

Table 1: OSE Components

Component	Type	Description
voss-keeper	system service	As a system service, the voss-keeper can be managed by `systemctl` command-line utility. It manages and monitors the health of the OSE Java service. It also manages the OSE voss-vip service. Works as the "backend" of the *ose commandline utility*. Stopping the voss-keeper service, also stops the OSE Java service on port 443.
voss-vip	system service	It is VMware Internationalization Protocol Service for OSE. As an internal service, it translates texts strings within the UI of OSE. It runs as a system service, and you can manage it by using the `systemctl` command-line utility.
OSE Java service	application service	The public service of VMware Cloud Director Object Storage Extension that provides the APIs for the data path and the control path on port 443.
OSE deployer	application service	OSE deployer is responsible for deploying the Velero agent in the Kubernetes cluster, which is required for the Kubernetes backup and restore feature.

Besides OSE-embedded components, the PostgreSQL database should be deployed to persist bucket/object metadata. The following is a high-level diagram of the OSE components:

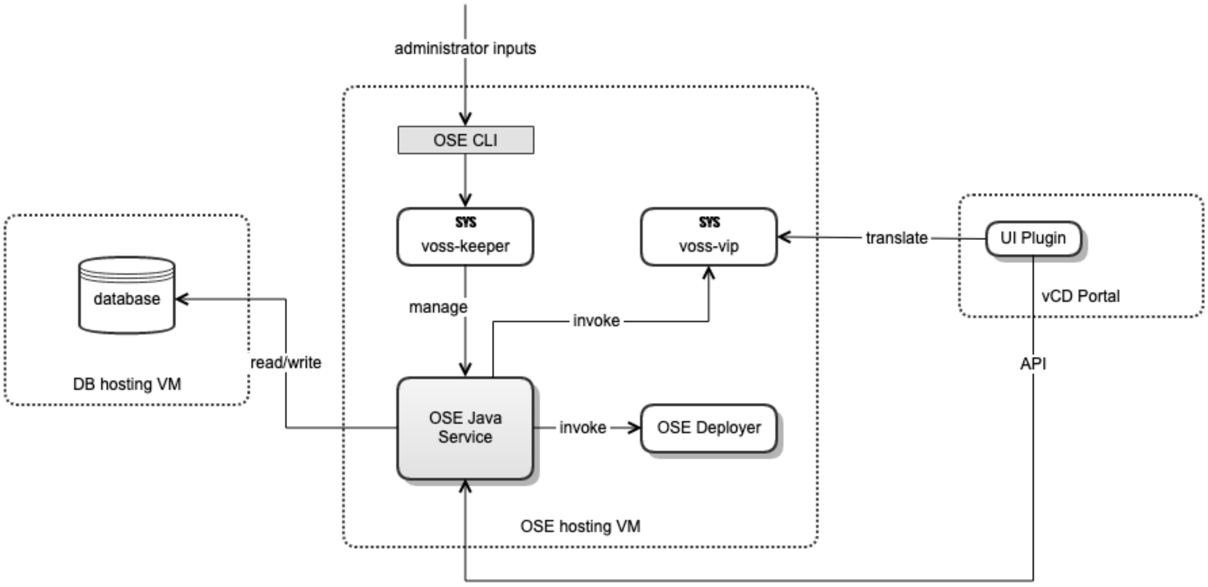

Figure 16: OSE 2.1.1 Component Diagram

OSE Deployment Views

Object Storage Extension uses port 443 for communication with Cloud Director, S3 storage, and S3-compliant storage apps. A load balancer is used for OSE nodes for production deployments to distribute the requests from Cloud Director to the OSE nodes. Through a URL redirect integrated with OSE, Cloud Director providers can connect to the management console of the underlying S3 storage. Cloud Director cells can also use a load balancer to distribute the OSE requests to Cloud Director. As part of the Cloud Director deployment, the Transfer Share provides temporary storage for uploads, downloads, and catalog items that are published or subscribed externally.

OSE connects through port 5432 to the PostgreSQL database, which keeps the metadata of the stored objects.

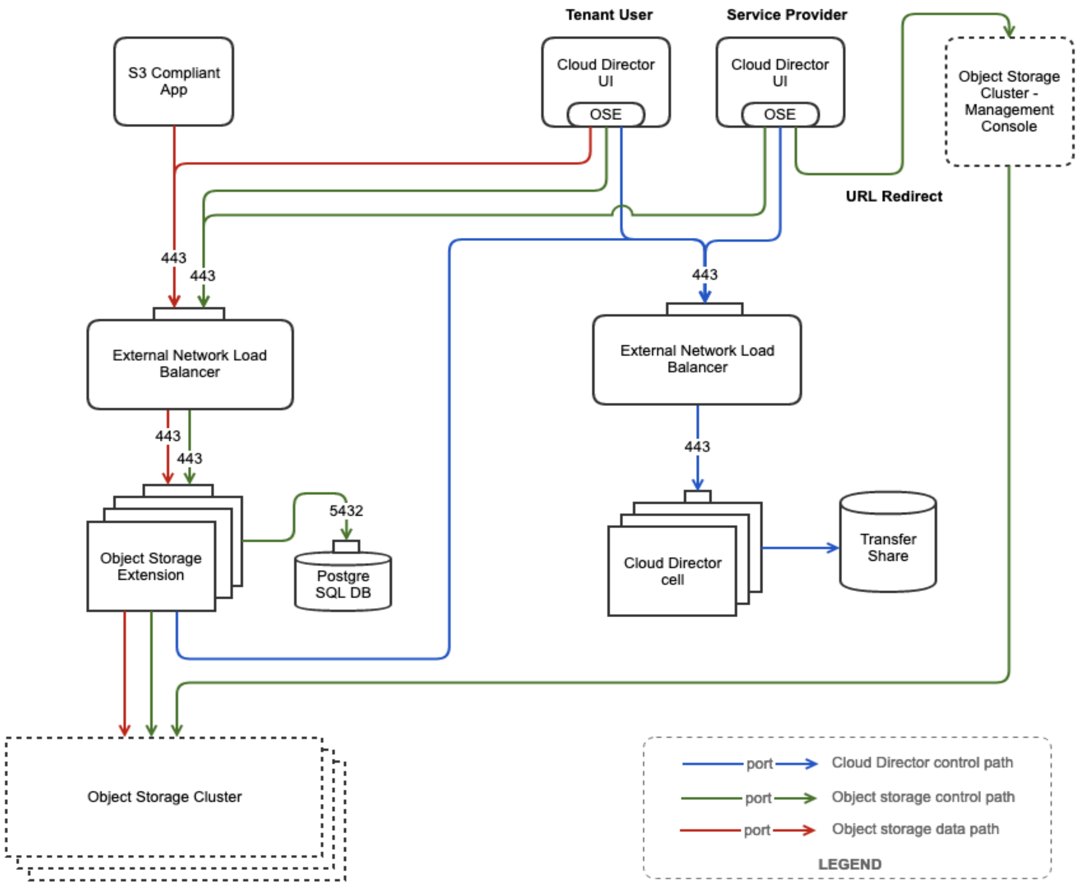

Figure 17: OSE Deployment Diagram

OSE with Cloudian Hyperstore Deployment View

Object Storage Extension uses the following Cloudian HyperStore components:

- S3 service – Used for the data path.
- Administration service – Used for the control path.
- IAM service - Used for the control path.
- Cloudian Management Console (optional) – Connection to it is made from the Cloud Director provider portal with a Single Sign-on.

Though each Cloudian HyperStore node offers standalone services, OSE should connect to Cloudian HyperStore nodes through an internal load balancer to gain the best throughput. It is also recommended to deploy OSE nodes close to Cloudian HyperStore to reduce the network latency.

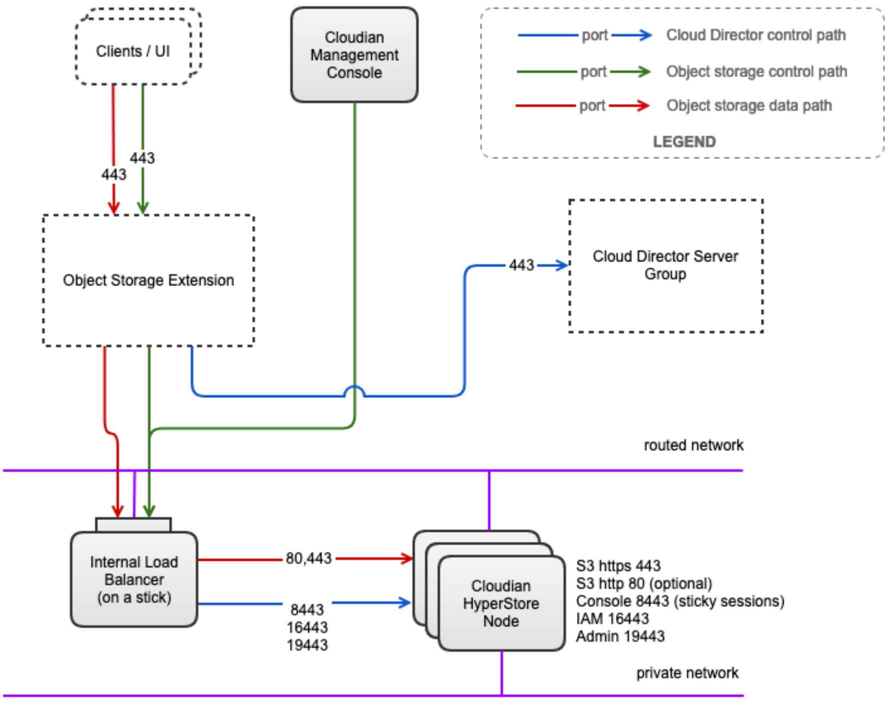

Figure 18: OSE with Cloudian Deployment View

OSE with Dell EMC ECS Deployment View

OSE integration with DELL EMC ECS uses the following ECS services:

- S3 service – Used for the data path.
- Administration service – Used for the control path,
- ECS Management Console (optional) – Used to connect the Cloud Director provider portal to the ECS Management Console, though another login is required.

Although each ECS node offers standalone services, OSE should connect to ECS nodes through an internal load balancer to gain the best throughput. It is also recommended that OSE nodes are deployed close to ECS to reduce the network latency.

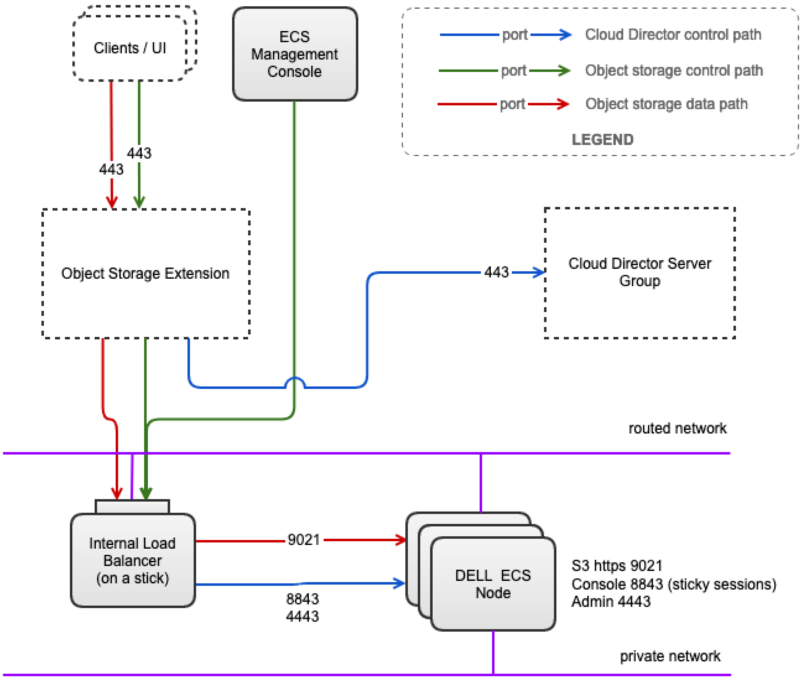

Figure 19: OSE with Dell ECS Deployment View

OSE with AWS Deployment View

The integration of OSE with AWS uses the following AWS services:

- S3 service – Used for the data path.
- Organizations – Create the tenancy.
- IAM – Used for user mapping and security credentials.
- STS – Produces the Single Sign-on access to AWS for the provider administrator.

The OSE deployment with AWS has two options:

- Deploy OSE to a local data center – Deploying it makes it easier to retain all management metadata in your local cloud. Also, AWS charges the storage and data transfer outside of the AWS region. For more information, see *AWS S3 Pricing*.
- Deploy OSE to AWS - Deploying to AWS has the advantage of the least network latency for the data path. By setting up a *Gateway VPC endpoint* between the OSE nodes and AWS S3, the cost for the data transfer from OSE to S3 can be eliminated.

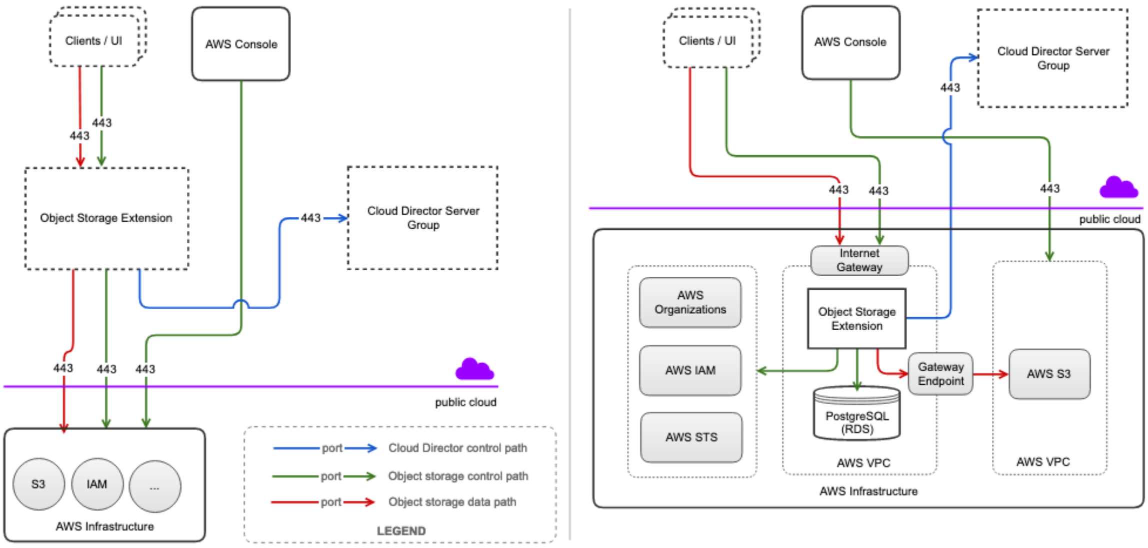

Figure 20: OSE with AWS Deployment View

OSIS Deployment View

Object Storage Interoperability Service (OSIS) is an extension interface for any S3-compliant object storage vendors to integrate with OSE and onboard their object storage service to Cloud Director users. The detailed introduction and code samples for OSIS can be found on *Github*.

An OSIS adapter needs to be implemented for the administration work on the object storage cluster. The OSIS Adapter can be deployed on a standalone machine or the local host of the OSE server node. The benefit of deploying the OSIS adapter on the OSE node eliminates the need to set an additional load balancer between OSE and the OSIS adapter.

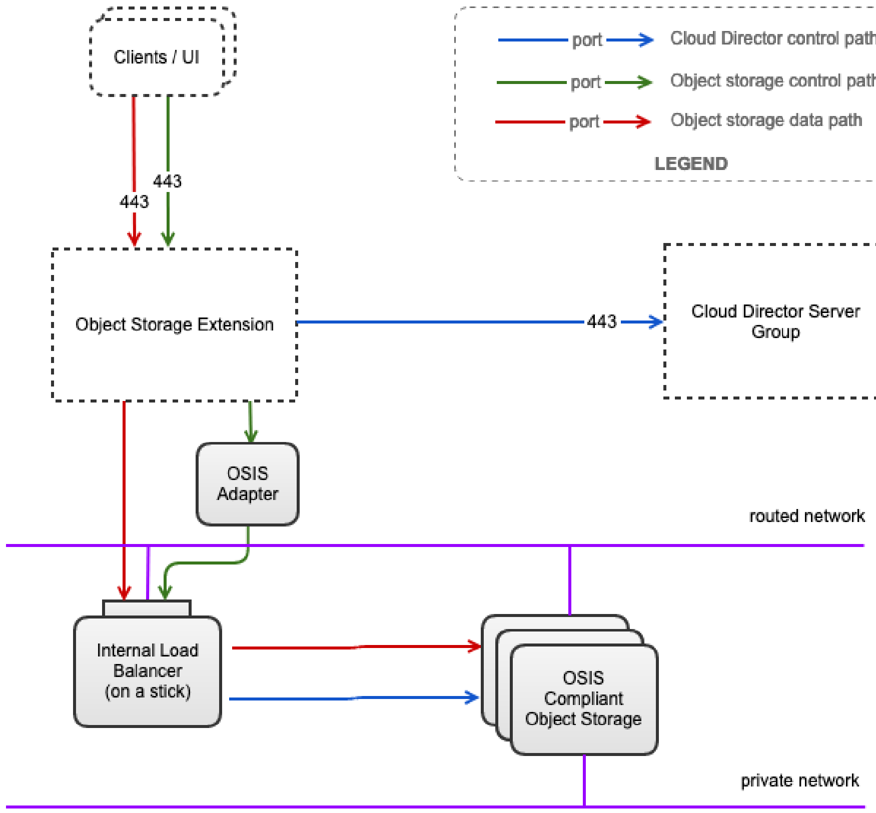

Figure 21: OSIS Deployment View

Deployment Options

Based on the use case, user target group, and expected service parameters (SLA, scalability), the cloud provider can decide on the type of deployment.

Small Deployment

Usage: Niche use cases

- Requirement: Minimum resources required. High availability, supported for production.
- One or more RHEL/CentOS VMs for VMware Cloud Director. External PostgreSQL database (used for VMware Cloud Director and VMware Cloud Director Object Storage Extension). NFS transfer share is needed when more than one VMware Cloud Director cell is used. Protected with vSphere HA.
- One CentOS Linux 7 or 8/RedHat Enterprise Linux 7/Oracle Linux 7/Ubuntu 18+/Photon 3+/Debian 10+ VM: (4 vCPU, 8 GB RAM, 120 GB HDD) running VMware Cloud Director Object Storage Extension. Protected with vSphere HA.
- vSphere/NSX: As required for VMware Cloud Director resources.
- Storage provider: Three CentOS virtual machines running Cloudian HyperStore, or Five CentOS virtual machines running Dell EMC ECS (4 vCPUs, 32 GB RAM, 32+100 GB HDD on shared storage) or AWS S3.
- Load balancing: VMware Cloud Director cells and Cloudian HyperStore or Dell EMC ECS nodes load balancing provided by NSX.

Medium Deployment

Usage: typical use cases

- Requirement: High availability, supported for production.
- Multiple RHEL/CentOS or appliance VMs for VMware Cloud Director. NFS transfer share. For non-appliance form factor external PostgreSQL database.
- One or more CentOS Linux 7 or 8/RedHat Enterprise Linux 7/Oracle Linux 7/Ubuntu 18+/Photon 3+/Debian 10+ VMs: (8 vCPU, 8 GB RAM, 120 GB HDD) running VMware Cloud Director Object Storage Extension. Protected with vSphere HA and optionally load balanced. If VMware Cloud Director is deployed in appliance form factor, an external PostgreSQL database is needed.
- vSphere/NSX: As required for VMware Cloud Director resources.
- Storage provider: Three CentOS virtual machines running Cloudian HyperStore, Five CentOS virtual machines running Dell EMC ECS on dedicated ESXi hosts with local disks (8 vCPUs, 64 GB RAM, 32 GB HDD + multiple large local disks) or AWS S3.
- Load balancing: VMware Cloud Director cells and Cloudian HyperStore, or Dell EMC ECS nodes load balancing provided by NSX or external hardware load balancer.

Large Deployment

Usage: large scale, low cost per GB use cases

- Requirement: High scale, performance, and availability, supported for production.
- Multiple RHEL/CentOS or appliance VMs for VMware Cloud Director. NFS transfer share. For non-appliance form factor external PostgreSQL database.
- Multiple CentOS Linux 7 or 8/RedHat Enterprise Linux 7/Oracle Linux 7/Ubuntu 18+/Photon 3+/Debian 10+ VMs (12 vCPU, 12 GB RAM, 120 GB HDD) running VMware Cloud Director Object Storage Extension. If VMware Cloud Director is deployed in an appliance form factor, an external HA PostgreSQL database is needed.
- vSphere/NSX: As required for VMware Cloud Director resources.
- Storage provider: Three or more dedicated bare-metal physical Cloudian HyperStore, Five or more physical Dell EMC ECS, or AWS S3.

- Load balancing: an external hardware load balancer

The following figures display how to scale out and load balance Object Storage Extension with Cloudian HyperStore, Dell EMC ECS, and AWS S3.

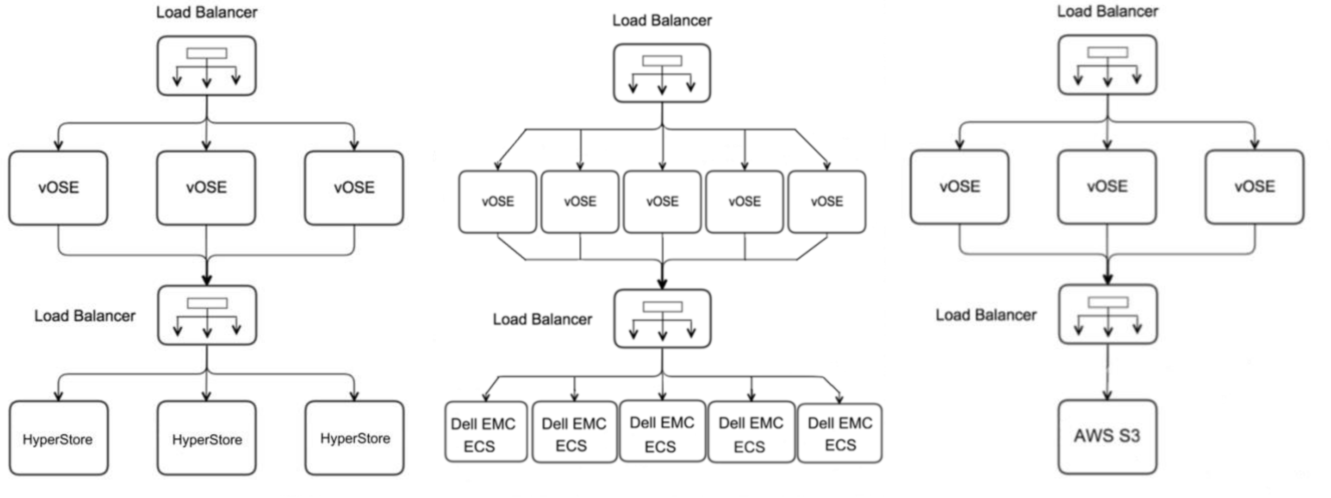

Figure 22: Example of Scale Out of Object Storage Extension Deployment with Load Balancing

Multisite Deployment

Object Storage Extension supports VMware Cloud Director multisite deployments where different VMware Cloud Director instances are federated (associated) with a trust relationship. As these instances can be deployed in different locations, the end-users can deploy their applications with a higher level of resiliency and not be impacted by local datacenter outages.

Each VMware Cloud Director instance has its own VMware Cloud Director Object Storage Extension, which communicates with shared S3 object storage deployed in a multi-datacenter configuration. Objects are automatically replicated across all data centers, and VMware Cloud Director users can access them through either VMware Cloud Director or VMware Cloud Director Object Storage Extension endpoint.

Within a multisite architecture, you can configure VMware Cloud Director Object Storage Extension instances with a standalone virtual data center in each site. The following diagram illustrates the architecture.

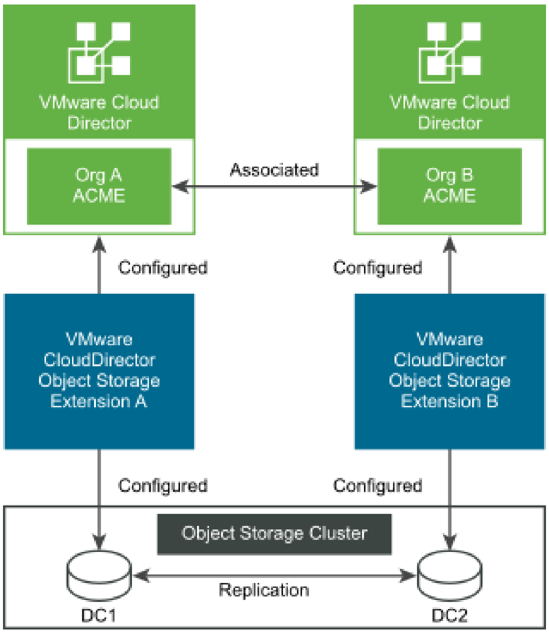

Figure 23: OSE Multisite Architecture: Single S3 Cluster for Multiple DCs

You can also configure VMware Cloud Director Object Storage Extension instances in different sites to use a single virtual data center. The following diagram illustrates the architecture.

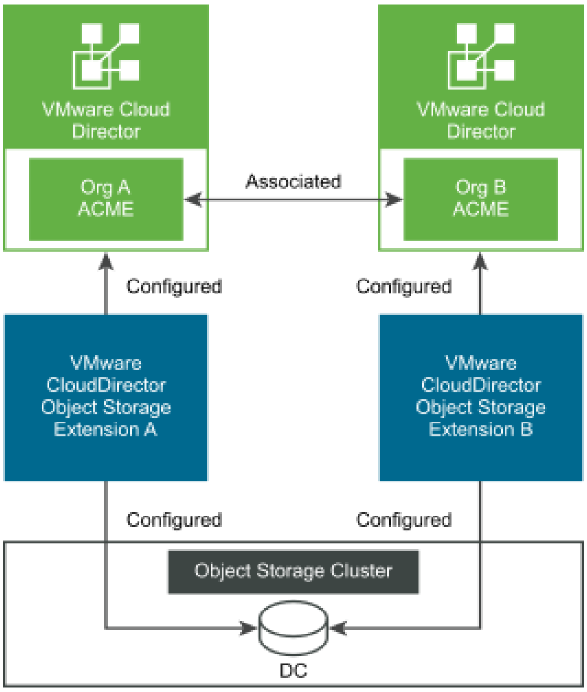

Figure 24: OSE Multisite Architecture: Single S3 Cluster for a Single DC

When you configure the multisite feature, you create a cluster of multiple VMware Cloud Director Object Storage Extension instances to create an availability zone. You can group the VMware Cloud Director Object Storage Extension instances together only in a single region. A region is a collection of the compute resources in a geographic area. Regions are isolated and independent of one another. VMware Cloud Director Object Storage Extension does not support multi-region architectures.

You can share the same buckets and objects across tenant organizations within a multisite environment. To share buckets and objects across sites, map all tenant organizations to the same storage group. See *Edit Tenant Mapping Configuration.*

OSE Scalability

OSE can be deployed as a cluster for high availability and distribution of hardware resources.

In the typical deployment topology, there are multiple OSE instances, multiple storage platform instances, and the database HA.

Deploying an OSE Cluster

Taking Cloudian HyperStore as an example, the steps to deploy the OSE cluster are described below.

Procedure

1. Prepare the OSE hosts.
2. Install the OSE rpm/deb package and start the OSE keeper.
3. Prepare the PostgreSQL database and check if it is accessible from the OSE hosts.
4. Prepare the Cloudian HyperStore nodes.
5. Prepare the Cloudian HyperStore load balancer so that it is accessible from the OSE hosts.

Configuring a Single OSE Instance

Procedure

1. Follow *these instructions* to configure the OSE certificate, database, and Cloud Director UI plugin.
2. Configure the connection to the Cloudian HyperStore Admin endpoint via the load balancer.

   ```
   ose cloudian admin set --url hyperstore-lb-admin-url --user admin-user --secret 'password'
   ```

3. Configure the connection to the Cloudian HyperStore S3 endpoint via the load balancer.

   ```
   ose cloudian s3 set hyperstore-lb-s3-url
   ```

4. Configure the connection to Cloudian HyperStore IAM endpoint via the load balancer.

   ```
   ose cloudian iam set hyperstore-lb-iam-url
   ```

5. Configure the connection to the HyperStore Web Console via the load balancer.

   ```
   ose cloudian console set --url hyperstore-lb-cmc-url --user admin-user --secret cmc-sso-shared-key
   ```

6. Validate the configuration.

   ```
   ose config validate
   ```

7. Start OSE.

   ```
   ose service start
   ```

8. Log in to Cloud Director and launch OSE to check whether it works normally.

Replicating Configuration on OSE Nodes behind a Load Balancer

Procedure

1. Connect to the first OSE host.

   ```
   ssh user@host-ip
   ```

2. Export the OSE configuration.

   ```
   ose config export --file="configuration-file-name" --secret="the password"
   ```

3. Copy the exported configuration file to the VMs of the other OSE instances.
4. SSH connect to the VMs of the other OSE instances and replicate the configuration by importing the configuration file.

```
ose config import --file="path-to-the-configuration-file" --secret="the password"
```

5. Restart the OSE keeper to make the configuration effective.

```
systemctl restart voss-keeper
```

Now the OSE cluster is created. In general, OSE instances are stateless, and all data is persisted in the shared database, so it is possible to add more nodes on demand.

OSE Configurations

OSE Java Service

OSE Java service is built with Spring Boot, which offers both administrative and S3 APIs for OSE UI plug-in and S3 API users.

First, the command `ose service [start|stop]` can launch and shut down the OSE Java service. The dedicated OSE CLI, e.g., `ose cloudian admin set`, can set basic configuration for the OSE service. The system administrator can also tune the OSE service with many other configurable properties by using the CLI command `ose args set`. Here are two examples.

- To make OSE work in virtual-hosted style for S3 API, use the command:

  ```
  ose args set -k s3.client.path.style.access -v false
  ```

- For a huge bucket (containing more than one hundred thousand objects), the object count for the bucket is estimated by default for performance consideration. The estimation can be turned off by the command:

  ```
  ose args set -k oss.object.count.estimate -v false
  ```

As a Java service, the JVM properties can also be set for the OSE instance. In some cases, the storage platform could be in another network that is accessible by OSE through a configured proxy server. The system administrator can set the JVM proxy options for OSE by using the command:

```
ose jvmargs -v "Dhttp.proxyHost=proxy.cloud.com -Dhttp.proxyPort=3128"
```

PostgreSQL Database

OSE uses a PostgreSQL database for storing the metadata of its S3 storage-related operations. The recommended hardware requirements for the database are 8 Core CPUs and 12 GB RAM for most OSE deployments.

An impact on the database disk usage will have the object count, not the object content size. The more objects you create in the system, the more disk space the database occupies. Many factors determine disk space consumption. Roughly one million objects cost about 0.6GB disk. Database indexes and logs will also consume disk. So, assuming you have one billion objects in an object storage cluster, you need to prepare more than 700GB of disk for the database machine.

There is a table `object_info` in the OSE database containing rows for each managed object. If OSE handles twenty million objects, the table will have twenty million rows. Querying such a table could be a performance bottleneck if the database machine has limited CPU and memory resources.

Now that we have the estimation for the database disk consumption with object count (about 0.6GB/million objects), it's recommended to allocate a buffer for the disk size at the beginning.

Public S3 Endpoint

S3-compliant API has two path formats:

- Path-Style Requests. The path pattern for Amazon S3 is https://s3.Region.amazonaws.com/bucket-name/key name, for example, https://s3.us-west-2.amazonaws.com/mybucket/puppy.jpg.

- Virtual Hosted-Style Requests. The path pattern for Amazon S3 is https://bucket-name.s3.Region.amazonaws.com/key name, for example https://my-bucket.s3.us-west-2.amazonaws.com/puppy.png.

OSE supports both styles of S3 endpoint, but the segment region is not on the S3 URI; assumed your organization's root FQDN is *https://acme.com*.

Table 2: S3 API Path Formats

S3 API Path Formats	Description	Examples
Path Style	The path-style S3 URI has /api/v1/s3 as the root path. Any FQDN can work. By default, OSE S3 API works in path-style.	https://storage.acme.com:443/api/v1/s3/bucket-1/dog.png https://storage.acme.com:443/api/v1/s3/bucket-2/cat.png
Virtual-Hosted Style	The virtual-hosted style S3 URI has s3. on the FQDN. FQDN must use prefix s3. and support wildcard subdomains, i.e., *s3.acme.com* and *.*s3.acme.com*.	https://bucket-1.s3.acme.com:443/dog.png https://bucket-2.s3.acme.com:443/cat.png

There are additional steps to make OSE work in a virtual-hosted style.

Procedure

1. Run the command to turn off the path style and switch to the virtual-hosted style:

```
ose args set -k s3.client.path.style.access -v false
```

2. Restart the ose service.

```
ose service restart
```

3. Configure wildcard DNS mapping for OSE S3 endpoint, i.e., map all *.s3.acme.com to the OSE load balancer.
4. Create a wildcard SSL certificate for the wildcard FQDN, i.e., make a common name as *.s3.acme.com.

OSE Performance Settings

The following settings can be applied to your OSE deployment to improve its performance.

Logging

The OSE logging level has an impact on the performance. To improve the performance, do not turn on the DEBUG logging. Besides, every request access is logged by default. It can be turned off as well.

The following examples show how to set the logging level to WARN or turn off logging. After changing the log level or turning it off, you need to restart the OSE service.

- Setting OSE logging level to WARN

```
ose args set --k logging.level.com.vmware.voss --v WARN
```

- Turning off OSE logging

```
ose args set --k server.undertow.accesslog.enabled --v false
```

- Restarting the OSE service

```
ose service restart
```

Tune I/O Thread Count

By default, the Undertow server creates server I/O threads per CPU cores on the OSE machine. See for reference: *http://undertow.io/undertow-docs/undertow-docs-1.2.0/listeners.html*.

If needed, you can increase the I/O thread count to gain performance out of I/O. However, the number should not be too high. For example, if OSE has 8 cores with 1 socket for each host, the default I/O threads for OSE is 2 * 8 = 16. You can increase the number to 24 with the command below:

```
ose args set --k server.undertow.threads.io --v 24
```

Tune the Worker Thread Count

The default working thread count of Spring Boot is 8 * I/O threads for the embedded Undertow server. Increasing the working thread count to match the concurrency is recommended to fully utilize the server capacity for a high concurrency workload.

```
ose args set --k server.undertow.threads.worker --v 256
```

Set Max Connection Count to Storage Platform

Concurrent connections to storage platform S3 API directly impact the system's scalability and throughput. By default, the max connection count is 1000.

```
ose args set --k s3.client.max.connections --v 1000
```

Set max Connection Count to the PostgreSQL Server

Concurrent connections to the database directly impact the system's scalability and throughput. By default, the max connection count is 90.

Note: The below setting is insufficient to increase the concurrency of database connections. You should consider increasing the max connection count on the PostgreSQL side simultaneously. For example, if the PostgreSQL server's max connection count is 1000, and you have deployed 5 OSE server nodes, then the average connection count to each OSE node should be less than the max connection count divided by the OSE node count, e.g., < 200.

```
ose args set --k spring.datasource.hikari.maximumPoolSize --v 180
```

Other settings for the database connection pool can be seen below. For term explanation, please refer to *https://github.com/brettwooldridge/HikariCP#configuration-knobs-baby*.

```
ose args set --k spring.datasource.hikari.maxLifetime --v 1800000

ose args set --k spring.datasource.hikari.idleTimeout --v 600000

ose args set --k spring.datasource.hikari.connectionTimeout --v 30000
```

Set Multipart Request Threshold for Upload

OSE middleware automatically splits the upload content stream into several parts for large objects. Depending on the network performance between the OSE middleware and storage platform, the threshold can be re-configured. The default setting is when the upload object size is over 1 GB, the upload is split, and each part is <= 1GB size.

```
ose args set --k s3.client.upload.multipart.threshold --v 1073741824

ose args set --k s3.client.upload.multipart.mini-part-size --v 1073741824

ose args set --k s3.client.copy.multipart.threshold --v 1073741824

ose args set --k s3.client.copy.multipart.mini-part-size --v 1073741824
```

Turn off Tenant Server-side Encryption

Tenant Server-side Encryption (SSE) is a unique feature of the OSE middleware. This feature can be turned off globally if you don't need it, which will improve OSE performance.

```
ose args set --k oss.tenant.sse.enabled --v false
```

Turn on OSE Virtual-hosted Style S3 Requests

By default, OSE works with path-style S3 requests. The command below will make OSE work with virtual-hosted style S3 requests.

```
ose args set -k s3.client.path.style.access -v false
```

Tune Object Count of Bucket

OSE has a feature showing tenant users the object count of each bucket. However, for buckets containing over 10 million objects, counting the bucket's objects will impact the performance.

Object count estimation is adopted for such buckets. The threshold is a hundred thousand objects per bucket. Use the following commands to adjust the threshold or turn off the estimation.

- Changing the object count estimate threshold

  ```
  oss.object.count.estimate.threshold=100000
  ```

- Turning off the object count estimate

```
ose args set -k oss.object.count.estimate -v false
```

Set Proxy for OSE

There are cases in which the storage platform is on another network that is accessible by OSE through a proxy server. You can set the JVM proxy options for OSE by using the following command.

```
ose jvmargs -v "Dhttp.proxyHost=proxy.cloud.com -Dhttp.proxyPort=3128"
```

Generate Support Bundle

OSE has a native CLI for support bundle, which will collect OSE information and logs of a specific period. See an example below:

```
ose support --start 2020-03-12 --end 2020-05-24
```

The optional argument `--start` defines the start time for the logs to be collected. The default value is 2018-01-01.

The optional argument `--end` defines the end time for the logs to be collected. If not specified, the end date is the current date.

Test Environment Reference Benchmark

Cloudian HyperStore Test Setup

A production-grade setup of VMware Cloud Director and VMware Cloud Director Object Storage Extension was deployed in the lab. Both VMware Cloud Director and Object Storage Extension were deployed in a three-node configuration. The object storage platform also consisted of three-load balanced hardware appliances Cloudian Hyperstore 1508. The workloads were simulated by three VM nodes running COSBench software - the industry-standard benchmark tool for object storage. The effect of the front-end load balancer on the test results was eliminated by connecting each COSBench node to one Object Storage Extension node. Cloudian HyperStore nodes were load-balanced with NSX-V Load Balancer in L4 TCP accelerated mode.

To assess the impact of OSE proxying of S3 APIs, the same tests were performed directly to the Cloudian HyperStore (through a load balancer). The following diagram shows the network flows of the S3 API communication.

Note that HTTPS was used both for front-end traffic (COSBench to Object Storage Extension nodes) and backend traffic (Object Storage Extension to Cloudian HyperStore or COSBench to Cloudian HyperStore).

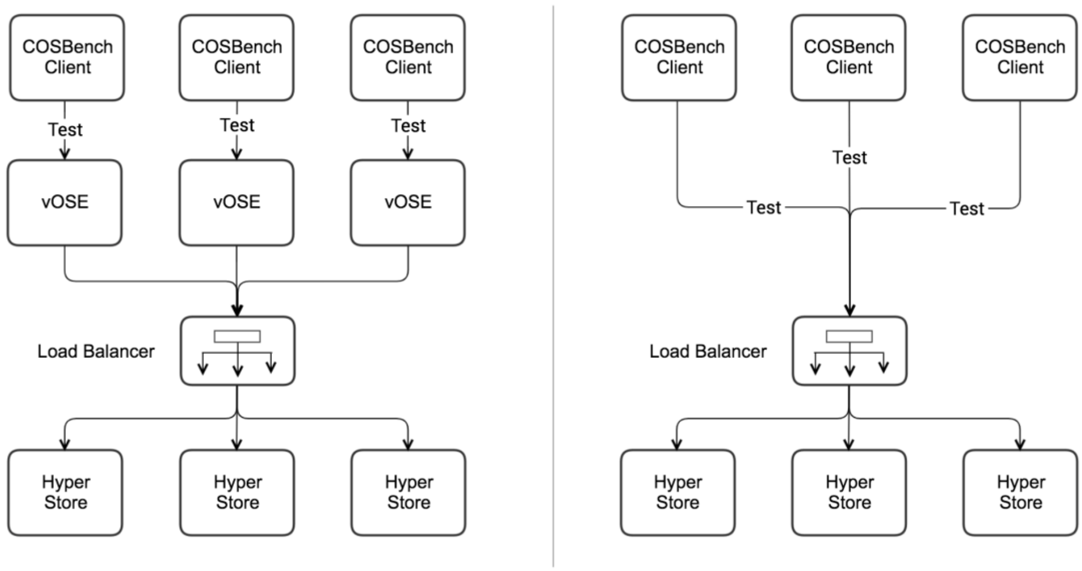

Figure 25: Cloudian HyperStore Test Topology

Cloudian Hyperstore- Bill of Materials

The following table lists the software and hardware components used to collect performance results for VMware Cloud Director Object Storage Extension.

Table 3. Bill of Material

Component	Count	Specifications	Notes
Load Balancer	1		CentOS 7; HAProxy 6 CPU, 8 GB RAM, 50 GB Disk
VMware Cloud Director	3	10.2	Appliance deployment (2 CPU, 12 GB RAM, 132 GB HDD)
Object Storage Extension (OSE) nodes	3	2.1.1	CentOS 7 VM (8 vCPUs, 8 GB RAM, 128 GB HDD)
PostgreSQL	1	10.0	CentOS 7; 6 CPU, 8GB RAM, 100 GB Disk
Cloudian HyperStore	3	7.2.4	Hardware appliance: Hyperstore 1508
COSBench	3	0.4.2	CentOS 7; 4 CPU, 8 GB RAM, 100 GB Disk

Cloudian HyperStore Test Results

Scenario 1 – Objects with size 10 MB

Workloads: 100 workers doing writes and reads to 25 containers with 10 MB objects.

Step 0: Prepare data for read.

Step 1: Write for 5 mins.

Step 2: Read for 5 mins.

Step 3: Delete for 5 mins.

Step 4: Clean up all buckets and objects.

Table 4. Cloudian HyperStore - HTTPS Write/Read of 10 MB Objects by 100 Workers across 25 Buckets

Test Type	Operation	Avg Response Time	Throughput	Bandwidth
OSE	Write	2839.36ms	35.21 op/s	352.11 MB/s
Cloudian	Write	1896.12 ms	52.74 Op/s	527.35 MB/s
OSE	Read	1866.83 ms	53.56 op/s	535.64 MB/s
Cloudian	Read	1612.98 ms	61.99 Op/s	619.88 MB/s

Scenario 2 - Concurrency

Workloads: Write, read, and delete for object size 100 MB for different concurrency level (10 – 200 workers).

Step 0: Prepare data for read.

Step 1: Write for 5 mins.

Step 2: Read for 5 mins.

Step 3: Delete for 5 mins.

Step 4: Clean up all buckets and objects.

Table 5. HTTPS 100 MB Objects with Various Concurrency 10, 50, 100 and 200 Workers

Test Type	Operation	Avg Response Time	Throughput	Bandwidth
OSE: 10 W	Write	6784.46ms	1.47op/s	147.22 MB/s
Cloudian: 10 W	Write	4911.74ms	2.03op/s	203.32 MB/s
OSE: 10 W	Read	4972.79ms	2.01op/s	201.11MB/s
Cloudian: 10 W	Read	3316.79ms	3.01 op/s	301.45 MB/s
OSE: 50 W	Write	11958.88ms	4.18op/s	417.99MB/s
Cloudian: 50 W	Write	12052.56ms	4.14 op/s	412.29MB/s
OSE: 50 W	Read	8169.29ms	5.58op/s	557.97MB/s
Cloudian: 50 W	Read	8058.57ms	6.2op/s	620.48MB/s
OSE: 100 W	Write	23875.77ms	4.19op/s	418.7MB/s
Cloudian: 100 W	Write	19502.59ms	5.13op/s	512.17MB/s
OSE: 100 W	Read	16130.22ms	6.2op/s	620.07MB/s
Cloudian: 100 W	Read	16111.49ms	6.21op/s	620.52MB/s
OSE: 200 W	Write	39163.08ms	5.99op/s	598.77MB/s
Cloudian: 200 W	Write	38256.24ms	5.22op/s	522.48MB/s
OSE: 200 W	Read	33032.24 ms	6.06op/s	605.58MB/s
Cloudian: 200 W	Read	5274.23 ms	5.67op/s	566.61MB/s

Scenario 3 – Small Objects

Workloads: Write, read, and delete for object size 4 KB with 100 workers across 30 buckets.

Step 1: 30 buckets with each bucket having 1000 objects.

Step 2: Write for 1 hour and read for 1 hour.

Step 3: Clean up all buckets and objects.

Table 6. Read and write of small objects by 100 Workers across 30 Buckets

Test Type	Operation	Avg Response Time	Throughput	Bandwidth
OSE: 100 W	Read	16.06ms	1272.43 op/s	5.09MB/s
Cloudian: 100 W	Read	10.4ms	1282.08op/s	5.13MB/s

OSE: 100 W	Write	62.35ms	1273.5 op/s	5.09MB/s
Cloudian: 100 W	Write	67.47ms	1282.39 op/s	5.13 MB/s

Scenario 4 – Object Size Comparison

Workloads: Write, read, and delete for various object sizes ranging from 1 MB – 1 GB with 100 workers across 100 buckets.

Step 1: Create 100 buckets with each bucket having 25 objects.

Step 2: Do 1000 write operations.

Step 3: Do 1000 read operations.

Step 4: Clean up all objects and buckets.

Table 7. HTTPs 1 MB – 1 GB Objects with Concurrency of 100 Workers

Test Type	Operation	Avg Response Time	Throughput	Bandwidth
OSE: 1 MB	Write	324.4ms	320.4 op/s	89.35 MB/s
Cloudian: 1 MB	Write	214.92ms	471.85op/s	471.85MB/s
OSE: 1 MB	Read	250.81ms	414.71 op/s	414.71 MB/s
Cloudian: 1 MB	Read	178.38ms	566.57 op/s	566.57MB/s
OSE: 10 MB	Write	3644.56ms	52.8 op/s	527.99 MB/S
Cloudian: 10 MB	Write	20252.39ms	49.03op/s	490.31 MB/s
OSE: 10 MB	Read	3187.77ms	32.24 op/s	322.35 MB/s
Cloudian: 10 MB	Read	1643.15ms	61.23 op/s	612.29 MB/s
OSE: 100 MB	Write	18969.84ms	5.34 op/s	534.4MB/s
Cloudian: 100 MB	Write	19077.34ms	5.25 op/s	524.92 MB/s
OSE: 100 MB	Read	16125.5ms	6.21 op/s	621.37MB/s
Cloudian: 100 MB	Read	15981.35ms	6.26 op/s	626.35MB/s
OSE: 1 GB	Write	294.27 ms	384.78 op/s	384.78 MB/S
Cloudian: 1 GB	Write	218.32 ms	452.86 op/s	452.86 MB/S
OSE: 1 GB	Read	209.64 ms	498.48 op/s	498.48 MB/S
Cloudian: 1 GB	Read	195.12 ms	527.3 op/s	527.3 MB/S

Conclusion

As can be seen from the above test results, VMware Cloud Director Object Storage Extension performance is much in line with the pure storage platform performance. It does not add significant overhead for object sizes greater than 1 MB with maximums around 5 - 15%

Dell EMC ECS Test Setup

The VMware Cloud Director Object Storage Extension allows VMware Cloud Providers using VMware Cloud Director to offer object storage services to their tenants. The extension acts as middleware which is tightly integrated with VMware Cloud Director to abstract 3rd party S3 API compatible storage providers in a multi-tenant fashion.

In this test setup, Object Storage Extension was deployed in a five-node configuration. The object storage platform consists of five load-balanced hardware appliances Dell EMC ECS. The workloads were simulated by three VM nodes running COSBench software, which is the industry standard benchmark tool for object storage. To assess the impact of the Storage Extension proxying of S3 APIs, the same tests were performed directly to the ECS nodes (through the load balancer). The following diagrams show the network flows of the S3 API communication.

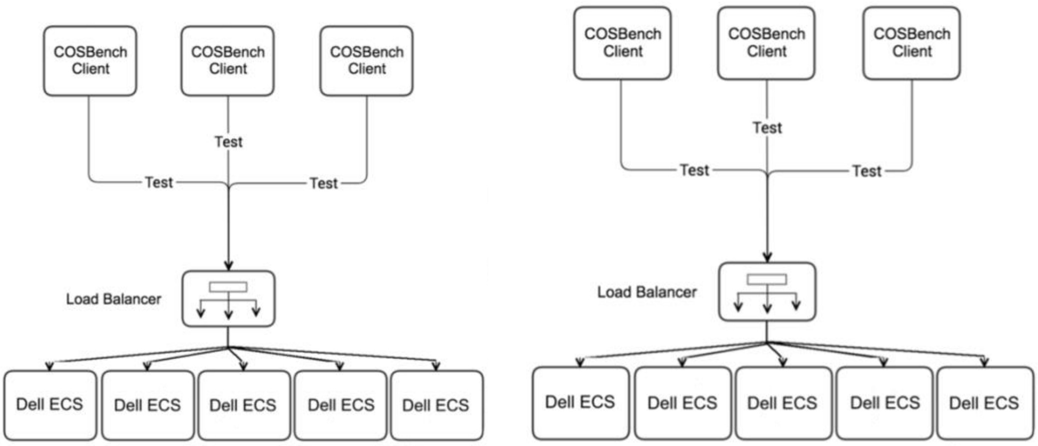

Figure 26. Dell EMC ECS Test Topology

Dell EMC ECS - Bill of Materials

The following table lists the software and hardware components used to collect performance results for VMware Cloud Director Object Storage Extension.

Table 8. Bill of Materials

Component	Count	Specifications	Notes
Load Balancer	1		CentOS 7; HAProxy 6 CPU, 8 GB RAM, 50 GB Disk
VMware Cloud Director cells	3	10.2	Appliance deployment (2 CPU, 12 GB RAM, 132 GB HDD)
VMware Cloud Director Object Storage Extension nodes	3	2.1.1	CentOS 7 VM (8 vCPUs, 8 GB RAM, 128 GB HDD)
PostgreSQL	1	10	Could be separate or part of VMware Cloud Director installation
Dell EMC ECS	5	3.4	
COSBench	3	0.4.2	Ubuntu VM (6 vCPUs, 8 GB RAM, 240 GB HDD)

Dell EMC ECS – Test Results

Scenario 1 – Large Objects

Workloads: 100 workers doing writes and reads to 25 buckets with 10 MB objects.

Step 0: Prepare data for read.

Step 1: Write for 5 mins.

Step 2: Read for 5 mins.

Step 3: Delete for 5 mins.

Step 4: Clean up all buckets and objects.

Table 9. Dell EMC ECS - HTTPs 10 MB Objects with Concurrency of 100 Workers across 25 Buckets

Test Type	Operation	Avg Response Time	Throughput	Bandwidth
OSE	Write	2654.28 ms	37.66 op/s	376.58 MB/S
ECS	Write	2005.08 ms	49.87 op/s	498.69 MB/S
OSE	Read	2432.37 ms	41.09 op/s	410.85 MB/S
ECS	Read	1677.03 ms	59.62 op/s	596.24 MB/S

Scenario 2 – Various Object Sizes Concurrency Comparison

Workloads: Write, read, and delete for object size 100 MB for different concurrency level (10 – 100 workers).

Step 0: Prepare data for read.

Step 1: Write for 5 mins.

Step 2: Read for 5 mins.

Step 3: Delete for 5 mins.

Step 4: Clean up all buckets and objects.

Table 10. Dell EMC ECS - HTTPs 100 MB Objects with Concurrency of [10-100] Workers

Test Type	Operation	Avg Response Time	Throughput	Bandwidth
OSE: 10 W	Write	5094.65 ms	1.96 op/s	196.28 MB/S
ECS: 10 W	Write	4573.61 ms	2.19 op/s	218.6 MB/S
OSE: 10 W	Read	4886.07 ms	2.15 op/s	204.62 MB/S
ECS: 10 W	Read	3596.51 ms	2.78 op/s	278.09 MB/S
OSE: 50 W	Write	13489.81 ms	3.71 op/s	370.61 MB/S
ECS: 50 W	Write	10108.52 ms	4.95 op/s	494.56 MB/S
OSE: 50 W	Read	11024.32 ms	4.51 op/s	451.12 MB/S
ECS: 50 W	Read	8600.87 ms	5.81 op/s	581.18 MB/S
OSE: 100 W	Write	26000.98 ms	3.79 op/s	379.02 MB/S
ECS: 100 W	Write	23602.16 ms	4.22 op/s	422.21 MB/S
OSE: 100 W	Read	23379.02 ms	4.18 op/s	418.45 MB/S
ECS: 100 W	Read	19714.67 ms	5.07 op/s	507.1 MB/S

Scenario 3 – Small Objects

Workloads: Write, read, and delete for object size 4 KB with 200 workers across 30 buckets.

Step 1: 30 buckets with each bucket having 10000 objects.

Step 2: 50% read, 50% write for 1 hour.

Step 3: Clean up all buckets and objects.

Table 11. Dell EMC ECS - HTTPs 4 MB Objects with Concurrency of 100 Workers across 30 Buckets

Test Type	Operation	Avg Response Time	Throughput	Bandwidth
OSE: 100 W	Read	15.82 ms	727.19 op/s	2.91 MB/S
ECS: 100 W	Read	13.83 ms	765.85 op/s	3.06 MB/S
OSE: 100 W	Write	121.11 ms	727.07 op/s	2.91 MB/S
ECS: 100 W	Write	116.56 ms	766.28 op/s	3.07 MB/S

Scenario 4 – Object Size Comparison

Workloads: Write, read, and delete for various objects ranging from 1 MB – 1 GB with 100 workers across 100 buckets.

Step 1: 100 buckets, with each bucket having 25 objects.

Step 2: Do 1000 write operations.

Step 3: Do 1000 read operations.

Step 4: Clean up all objects and buckets.

Table 12. Dell EMC ECS - HTTPs 1 MB – 1GB Objects with Concurrency of 100 Workers across 100 Buckets

Test Type	Operation	Avg Response Time	Throughput	Bandwidth
OSE: 1 MB	Write	327.92 ms	335.74 op/s	335.74 MB/S
ECS: 1 MB	Write	257.65 ms	399.24 op/s	399.24 MB/S
OSE: 1 MB	Read	234.94 ms	465.05 op/s	465.05 MB/S
ECS: 1 MB	Read	206.63 ms	503.81 op/s	503.81 MB/S
OSE: 10 MB	Write	2387.46 ms	42.72 op/s	427.25 MB/S
ECS: 10 MB	Write	2048.71 ms	49.03 op/s	490.28 MB/S
OSE: 10 MB	Read	2075.17 ms	50.59 op/s	505.95 MB/S
ECS: 10 MB	Read	1684.57 ms	59.45 op/s	594.54 MB/S
OSE: 100 MB	Write	24580.03 ms	4.22 op/s	421.51 MB/S
ECS: 100 MB	Write	19674.93 ms	5.1 op/s	510.33 MB/S
OSE: 100 MB	Read	21524.92 ms	4.73 op/s	472.63 MB/S
ECS: 100 MB	Read	16287.79 ms	6.18 op/s	617.89 MB/S
OSE: 1 GB	Write	235635.04 ms	0.45 op/s	445.13 MB/S
ECS: 1 GB	Write	198951.58 ms	0.5 op/s	504.42 MB/S
OSE: 1 GB	Read	205287.62 ms	0.51 op/s	507.21 MB/S
ECS: 1 GB	Read	163421.88 ms	0.62 op/s	615.88 MB/S

Conclusion

As can be seen from the above test results, VMware Cloud Director Object Storage Extension performance is much in line with the pure storage platform performance. It adds overhead with maximums around 5-25%.

AWS S3 Test Setup

VMware Cloud Director Object Storage Extension was deployed in a three-node configuration. The object storage platform is Amazon Simple Storage Service (Amazon S3). The workloads were simulated by three VM nodes running COSBench software, industry-standard benchmark tool for object storage. To assess the impact of VMware Cloud Director Object Storage Extension proxying of S3 APIs, the same tests were performed directly with the AWS S3 service. The following diagrams show the network flows of the S3 API communication.

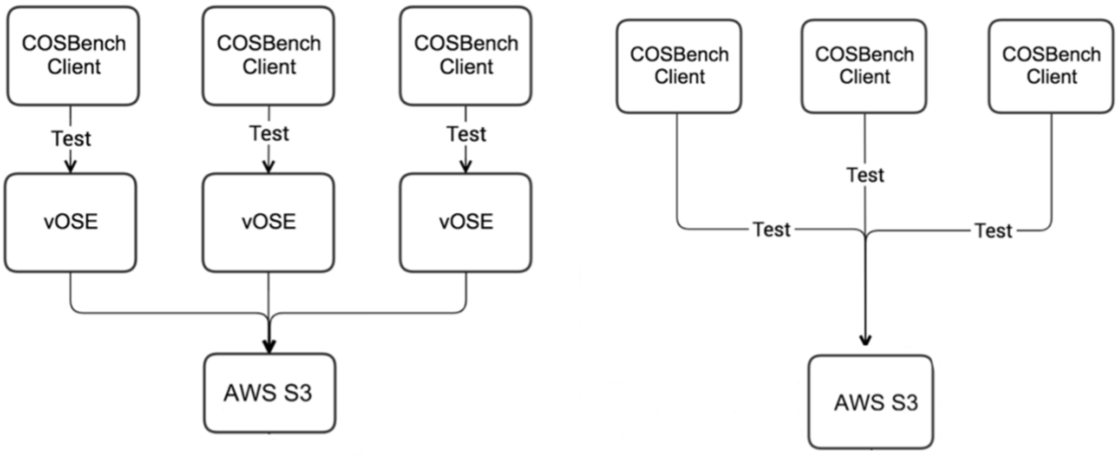

Figure 27. AWS S3 Test Topology

AWS S3 - Bill of Materials

The following table lists the software and hardware components used to collect performance results for VMware Cloud Director Object Storage Extension.

Table 13. Bill of Materials

Component	Count	Specifications	Notes
Load Balancer	1		CentOS 7; HAProxy 6 CPU, 8 GB RAM, 50 GB Disk
VMware Cloud Director cells	3	10.2	Appliance deployment (2 CPU, 12 GB RAM, 132 GB HDD)
VMware Cloud Director Object Storage Extension nodes	3	2.1.1	CentOS 7 VM (8 vCPUs, 8 GB RAM, 128 GB HDD)
PostgreSQL	1	10	Could be separate or part of VMware Cloud Director installation
AWS S3	1		
COSBench	3	0.4.2	Ubuntu VM (6 vCPUs, 8 GB RAM, 240 GB HDD)

AWS S3 – Test Results

Scenario 1 – Large Objects

Workloads: 100 workers doing writes and reads to 25 buckets with 10 MB objects.

Step 0: Prepare data for the read.

Step 1: Write for 5 mins.

Step 2: Read for 5 mins.

Step 3: Delete for 5 mins.

Step 4: Clean up all buckets and objects.

Table 14. AWS- HTTPs 10 MB Objects with Concurrency of 100 Workers across 25 Buckets

Test Type	Operation	Avg. Response Time	Throughput	Bandwidth
OSE	Write	3219.13 ms	31.05 op/s	310.53 MB/S
AWS	Write	3213.18 ms	31.12 op/s	311.23 MB/S
OSE	Read	3614.7 ms	27.66 op/s	276.65 MB/S
AWS	Read	3189.3 ms	31.35 op/s	313.51 MB/S

Scenario 2 – Concurrency Comparison

Workloads: Write, read, and delete for object size 100 MB for different concurrency levels (10 – 200 workers).

Step 0: Prepare data for the read.

Step 1: Write for 5 mins.

Step 2: Read for 5 mins.

Step 3: Delete for 5 mins.

Step 4: Clean up all buckets and objects.

Table 15. AWS- HTTPs 100 MB Objects with Concurrency of [10-200] Workers

Test Type	Operation	Avg. Response Time	Throughput	Bandwidth
OSE: 10 W	Write	3585.87 ms	2.79 op/s	278.56 MB/S
AWS: 10 W	Write	3211.07 ms	3.11 op/s	311.18 MB/S
OSE: 10 W	Read	3330.07 ms	3 op/s	300.09 MB/S
AWS: 10 W	Read	3186.34 ms	3.13 op/s	313.37 MB/S
OSE: 50 W	Write	16123.54 ms	3.1 op/s	309.76 MB/S
AWS: 50 W	Write	15941.99 ms	3.13 op/s	313.31 MB/S
OSE: 50 W	Read	17545.01 ms	2.85 op/s	284.77 MB/S
AWS: 50 W	Read	15839.36 ms	3.15 op/s	315.31 MB/S
OSE: 100 W	Write	31723.52 ms	3.14 op/s	313.93 MB/S

AWS: 100 W	Write	31557.58 ms	3.16 op/s	316 MB/S
OSE: 100 W	Read	33318.05 ms	3 op/s	300.2 MB/S
AWS: 100 W	Read	31271.15 ms	3.2 op/s	320.12 MB/S
OSE: 200 W	Write	63366.13 ms	3.15 op/s	315.4 MB/S
AWS: 200 W	Write	62448.93 ms	3.19 op/s	319.13 MB/S
OSE: 200 W	Read	70461.69 ms	2.84 op/s	284.14 MB/S
AWS: 200 W	Read	62421.21 ms	3.2 op/s	320.35 MB/S

Scenario 3 – Small Objects

Workloads: Write, read, and delete for object size 1 MB with 1 00 workers across 30 buckets.

Step 1: 30 buckets with each bucket having 100,000 objects.

Step 2: Write for 1 hour and read for 1 hour.

Step 3: Clean up all buckets and objects.

Table 16. AWS- HTTPs 1 MB Objects with Concurrency of 100 Workers across 30 Buckets

Test Type	Operation	Avg Response Time	Throughput	Bandwidth
OSE: 100 W	Write	370.41 ms	269.91 op/s	269.91 MB/S
AWS: 100 W	Write	369.81 ms	270.35 op/s	270.35 MB/S
OSE: 100 W	Read	369.21 ms	270.83 op/s	270.83 MB/S
AWS: 100 W	Read	367.04 ms	272.43 op/s	272.43 MB/S

Scenario 4 – Object Size Comparison

Workloads: Write, read, and delete for various object sizes ranging from 1 MB – 1 GB with 100 workers across 100 buckets

Step 1: 100 buckets, with each bucket having 25 objects

Step 2: Do 1000 write operations

Step 3: Do 1000 read operations

Step 4: Clean up all objects and buckets

Table 17. AWS- HTTPs 1 MB -1 GB Objects with Concurrency of 100 Workers across 100 Buckets

Test Type	Operation	Avg Response Time	Throughput	Bandwidth
OSE: 1 MB	Write	286.64 ms	352.17 op/s	352.17 MB/S
AWS: 1 MB	Write	255.88 ms	401.95 op/s	401.95 MB/S
OSE: 1 MB	Read	278.63 ms	361.63 op/s	361.63 MB/S

AWS: 1 MB	Read	243.58 ms	415.37 op/s	415.37 MB/S
OSE: 10 MB	Write	2789.34 ms	36.7 op/s	366.99 MB/S
AWS: 10 MB	Write	2509.52 ms	42 op/s	420 MB/S
OSE: 10 MB	Read	2735.01 ms	36.63 op/s	366.26 MB/S
AWS: 10 MB	Read	2604.47 ms	38.6 op/s	386.04 MB/S
OSE: 100 MB	Write	31179 ms	3.27 op/s	327.19 MB/S
AWS: 100 MB	Write	30227.8 ms	3.49 op/s	348.67 MB/S
OSE: 100 MB	Read	28733.64 ms	3.56 op/s	355.58 MB/S
AWS: 100 MB	Read	28201.35 ms	3.69 op/s	368.57 MB/S
OSE: 1 GB	Write	295057.91 ms	0.35 op/s	350.96 MB/S
AWS: 1 GB	Write	295723.67 ms	0.35 op/s	346.39 MB/S
OSE: 1 GB	Read	330634.61 ms	0.32 op/s	323.51 MB/S
AWS: 1 GB	Read	291255.77 ms	0.35 op/s	350.39 MB/S

Conclusion

As can be seen from the test results above, VMware Cloud Director Object Storage Extension performance is much in line with the pure storage platform performance. It does not add significant overhead with maximums around 5- 13%.

Abbreviations

OSE	VMware Cloud Director Object Storage Extension
OSIS	Object Storage Interoperability Service
IaaS	Infrastructure as a Service
CRD	Custom Resource Definition
CDS	Cloud Director Service