ORACLE

Scale Out OCI File Storage Performance for AI/ML and Data-Intensive Workloads

November 2024, Version 1.1

Public

ORACLE

Disclaimer

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

Revision History

The following revisions have been made to this document.

Date	Revision
November 2024	Added best practices for monitoring
September 2024	Initial publication

ORACLE

Table of Contents

Overview

Oracle Cloud Infrastructure (OCI) File Storage is widely used for its simple interface and ability to scale to exabytes in a single name space. High-performance mount targets now enable you to scale up your throughput. You can create multiple high-performance mount targets to linearly scale and increase your aggregate throughput. This type of scaling enables AI/ML workloads to quickly load data from a single file system, which avoids idle GPU time. As you add more mount targets, you need the ability for GPUs to use a mount target dynamically from a pool of mount targets. This paper describes how to use DNS round robin to dynamically assign mount targets to clients, and how to configure DNS round robin with OCI's fully managed DNS for virtual cloud networks (VCNs).

Scaling Out File Storage Performance

File Storage performance can be scaled up by horizontally scaling out mount targets. In this scenario, the file system is exported across multiple mount targets, and each mount target provides a single name space to access the file system. For example, each high-performance mount target is capable of 80 Gbps read throughput, and 5 mount targets enable 400 Gbps aggregate throughput.

To distribute mount target assignments, you can use static assignments, as shown in Figure 1. If you have 15 Compute instances, 5 mount targets are assigned to 15 NFS clients statically to distribute the load across all the mount targets.

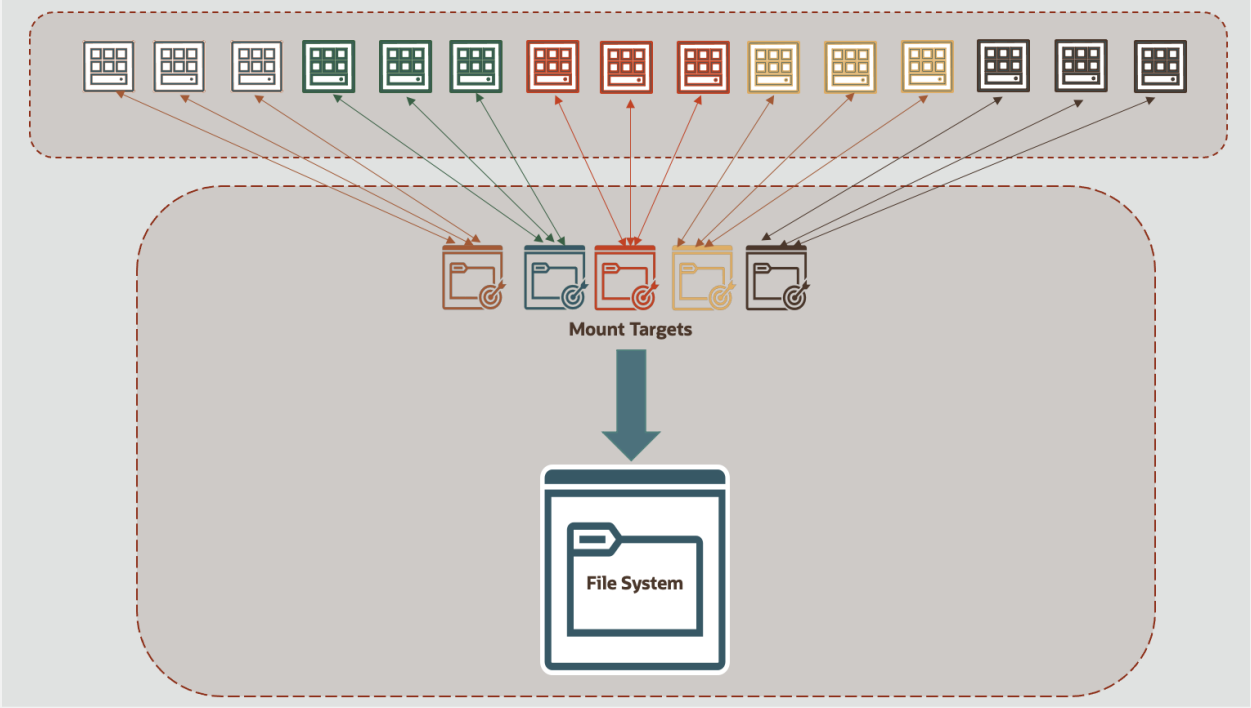

Figure 1. Scaling Out File Storage Performance: Static Assignments, Three Instances per Mount Target

You can also use DNS round robin to distribute mount target assignments. With DNS round robin, you no longer need static assignments. The NFS clients are dynamically assigned to the mount targets, as shown in Figure 2.

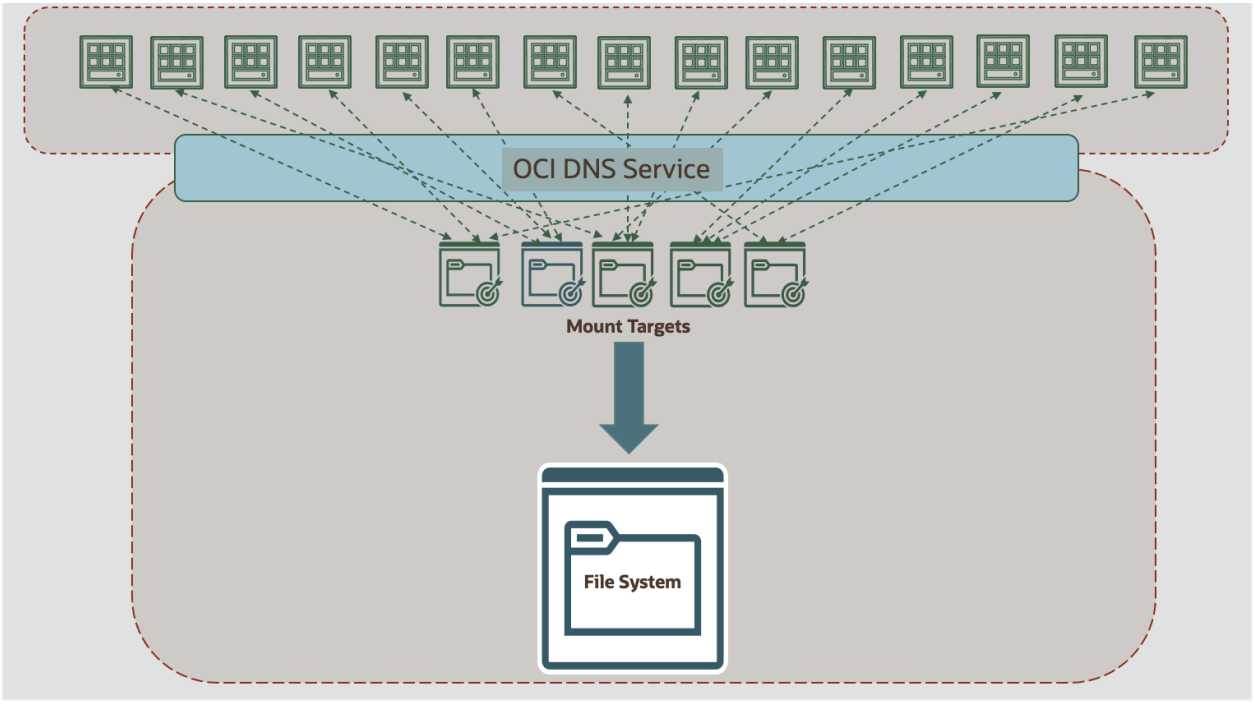

Figure 2. Scaling Out File Storage Performance: Assignments by Round Robin DNS, Random but Dynamic Assignments

NFS and DNS Round Robin

DNS round robin is a widely used technique to load balance by distributing traffic across multiple servers. To use DNS round robin, you configure File Storage mount target IP addresses as the A record for the file system, and then use the DNS name to mount the File Storage export. When you use a DNS name to mount an NFS share, the client picks one IP address from the list returned by the DNS server. This chosen IP address is used for the entire mount duration, until the file system is unmounted and remounted again. If the A record for the NFS server contains multiple IP addresses, OCI DNS randomizes the order before sending the list to the client. The randomization of IP addresses by DNS round robin spreads the load across the available mount targets.

OCI DNS

OCI provides a DNS service within VCNs for host name resolution. The default resolver can resolve automatically assigned host names within the VCN as well as public DNS names in the internet. The resolver listens on 169.254.169.254 by default in the VCN, and instances launched in the VCN have this IP address configured as the resolver. The VCN DNS service also provides private DNS to resolve host names in a custom domain. You can create private DNS zones to provide DNS resolution for the custom domains. These private DNS zones are reachable through the default DNS resolver.

Configuring DNS Round Robin with File Storage Private Zones

The Configure private DNS zones views and resolvers tutorial explains in detail how private zones can be configured within a VCN. To provide DNS round robin functionality for file systems, you create a private DNS zone to host all the File Storage file systems within a VCN. You then add an A record for each file system within the VCN. For the file system A record, you add the IP addresses of all the mount targets serving that file system. The TTL for the A record can be 1 second, to make the name resolution more dynamic. When a new mount target is added to scale up performance, you can update the A record to add the IP address of the new mount targets.

For example, the file system `fs1` is served by four high-performance mount targets. You create a private zone named `fss.zone` to host all the file systems. In the Oracle Cloud Console, you create a private zone for File Storage by going to the details for the VCN, selecting the DNS resolver, selecting the default private view, and then selecting to create the zone.

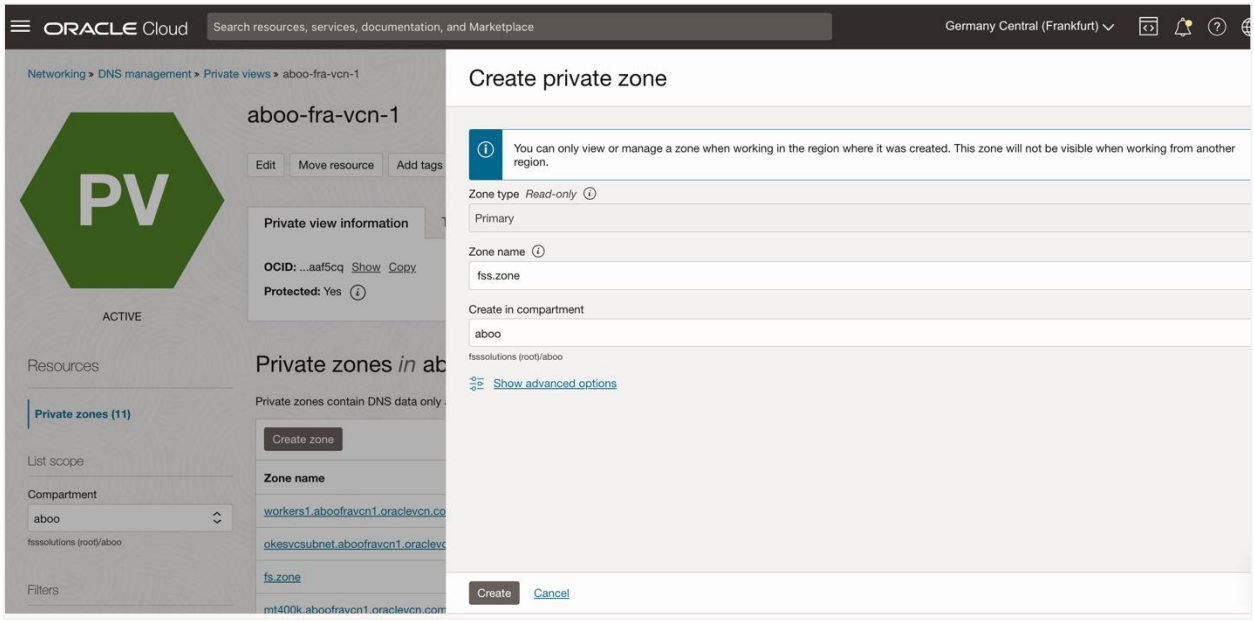

Figure 3. Create a Private Zone for File Storage

You then add the file system `fs1` as an A record with four mount target IP addresses.

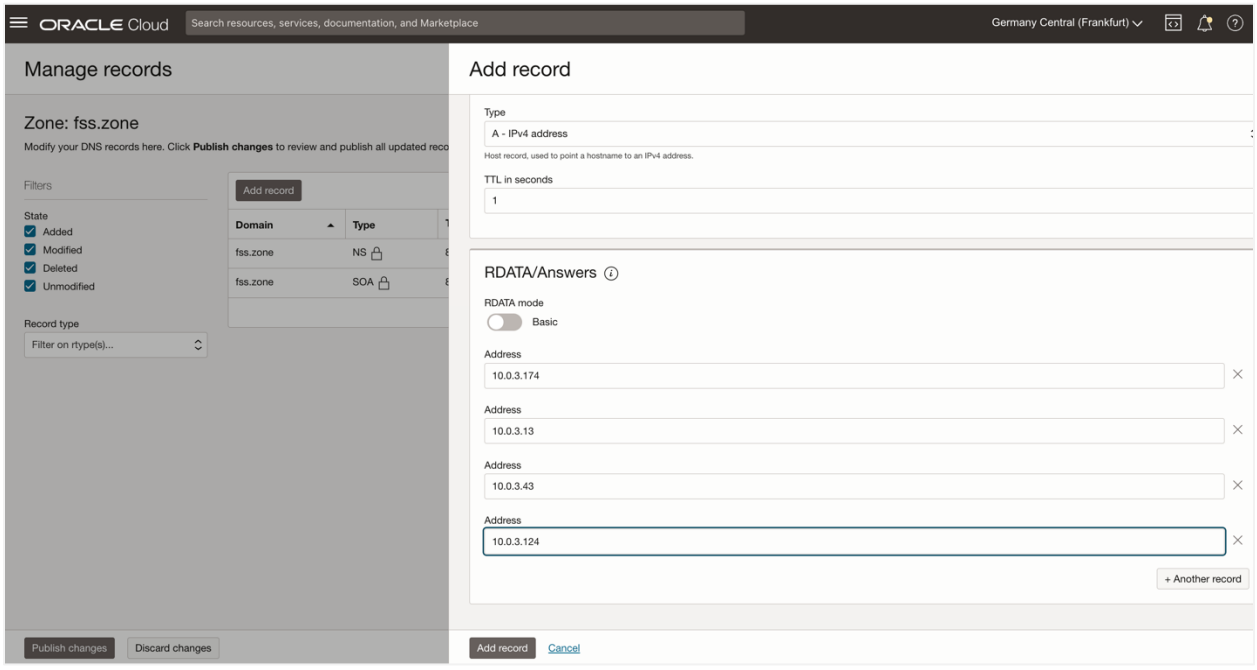

Figure 4. Create an A Record for the File System

The following DNS query shows the four A records linked to the file system fs1.

```
[opc@oke-cnau3ueeifq-nl6jaj6pynq-saaormaebxq-119 ~]$ host fs1.fss.zone
fs1.fs.zone has address 10.0.3.174
fs1.fs.zone has address 10.0.3.13
fs1.fs.zone has address 10.0.3.43
fs1.fs.zone has address 10.0.3.124
[opc@oke-cnau3ueeifq-nl6jaj6pynq-saaormaebxq-119 ~]$ sleep 5; host fs1.fss.zone
fs1.fs.zone has address 10.0.3.43
fs1.fs.zone has address 10.0.3.124
fs1.fs.zone has address 10.0.3.174
fs1.fs.zone has address 10.0.3.13
```

Configuring Persistent Volume for Kubernetes Pods

The example described in this paper uses Kubernetes to scale the clients up and down, although DNS round robin can be used with native NFS clients without Kubernetes. The example is run with 512 worker nodes (NFS clients) in an OCI Kubernetes Engine (OKE) cluster. The Kubernetes pod runs a sample workload using fio to the File Storage service. The pods use just one Kubernetes persistent volume (PV) configured with the DNS name. The following PV manifest is used:

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: nfs-pv
spec:
  capacity:
    storage: 50Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteMany
  persistentVolumeReclaimPolicy: Retain
  nfs:
    path: /fs-400k
    server: fs1.fs.zone
  mountOptions:
    - nconnect=16
```

Note: The OCI File Storage CSI driver can also be used in OKE.

Results

This example shown in this paper demonstrates that the DNS round robin technique is effective in distributing load from all the NFS clients across four mount targets. The following graph shows scaling up and down of workload pods, ranging from 0 and 512. The top line segment, `MountTargetConnections`, shows the total number of pods active, which is equal to the total number of NFS connections at that moment, 512. The pods aren't configured to run more than once on each node. This means that `MountTargetConnections` directly correlates with the number of active pods and NFS mounts from worker nodes. The bottom four line segments are connections to individual mount targets indicated by the mount target OCID. Even with random scaling of the number of pods up and down, the line segments for each mount target stay close together, demonstrating the effectiveness of DNS round robin in distributing connections across multiple mount targets.

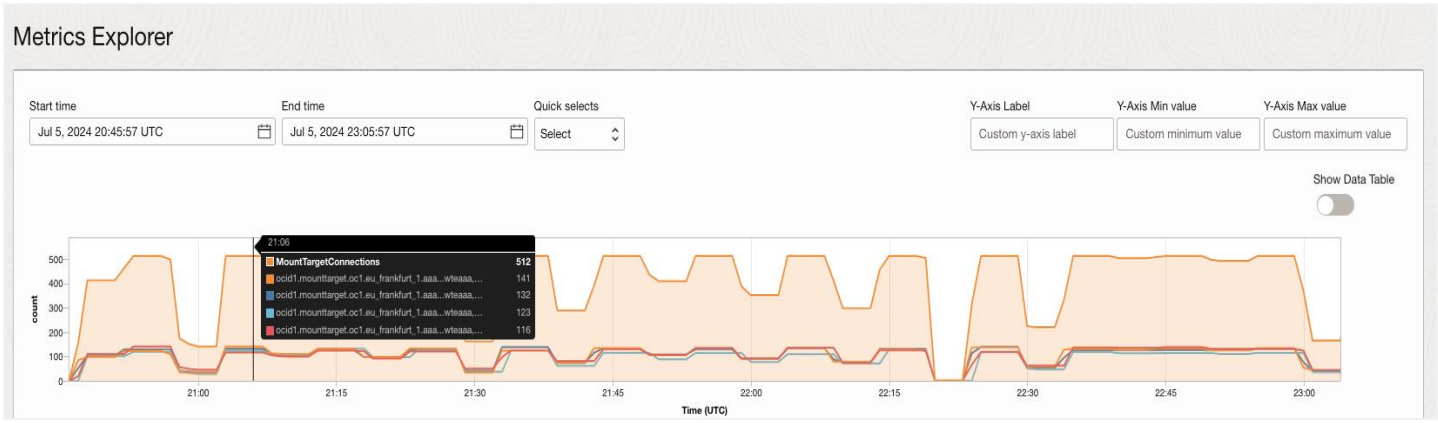

Figure 5. Mount Target Connections with Random Scaling of NFS Clients

The following diagram shows total throughput, throughput per mount target, and the number of connections per mount target with scaling from 0 to 512 multiple times, each time saturating the total read bandwidth of the mount target at 12 GB/s with a total file system read throughput of 48 GB/s.

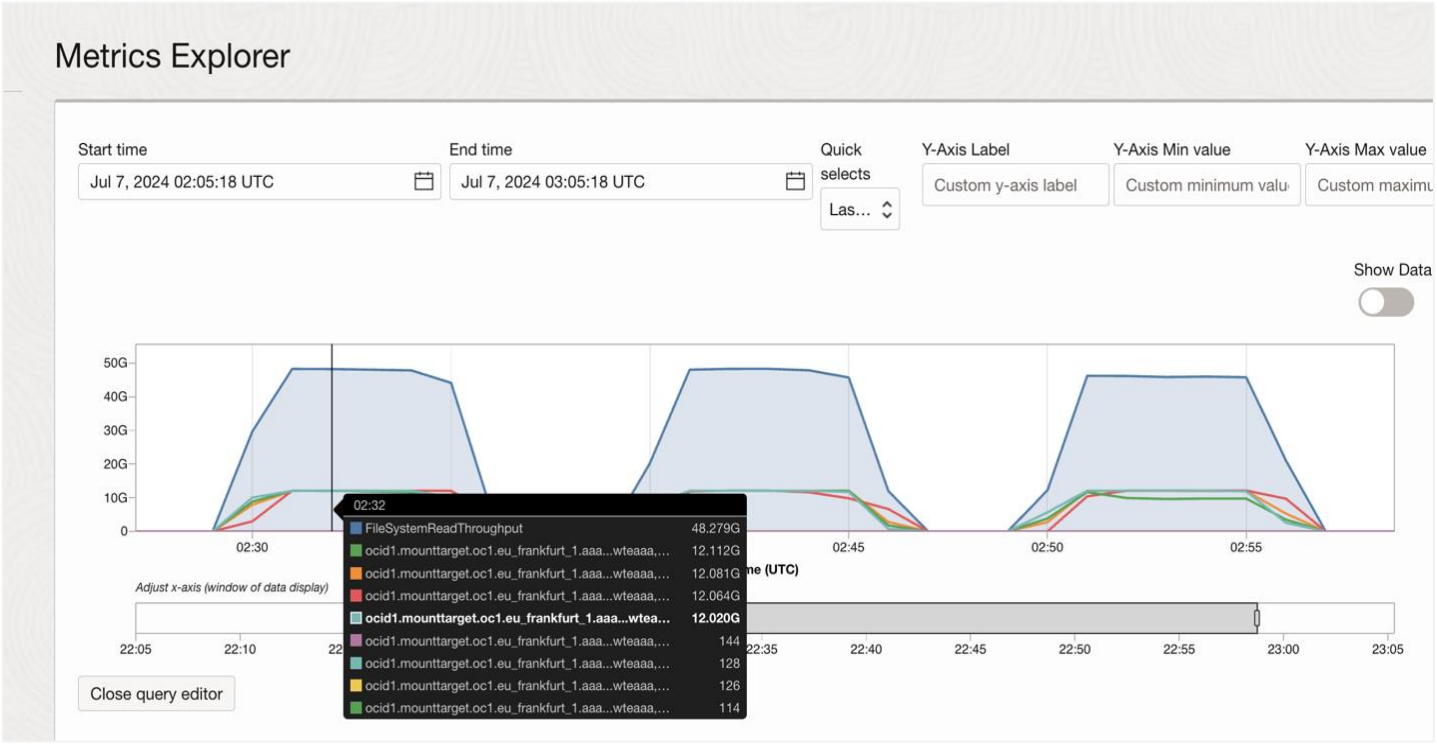

Figure 6. Throughput and Connections Distribution Scaling Between 0 and 512 Clients

The following graph demonstrates the number of connections per mount target with the `nconnect=16` NFS mount option in the PV. The pods were continuously scaled between 0 and 512.

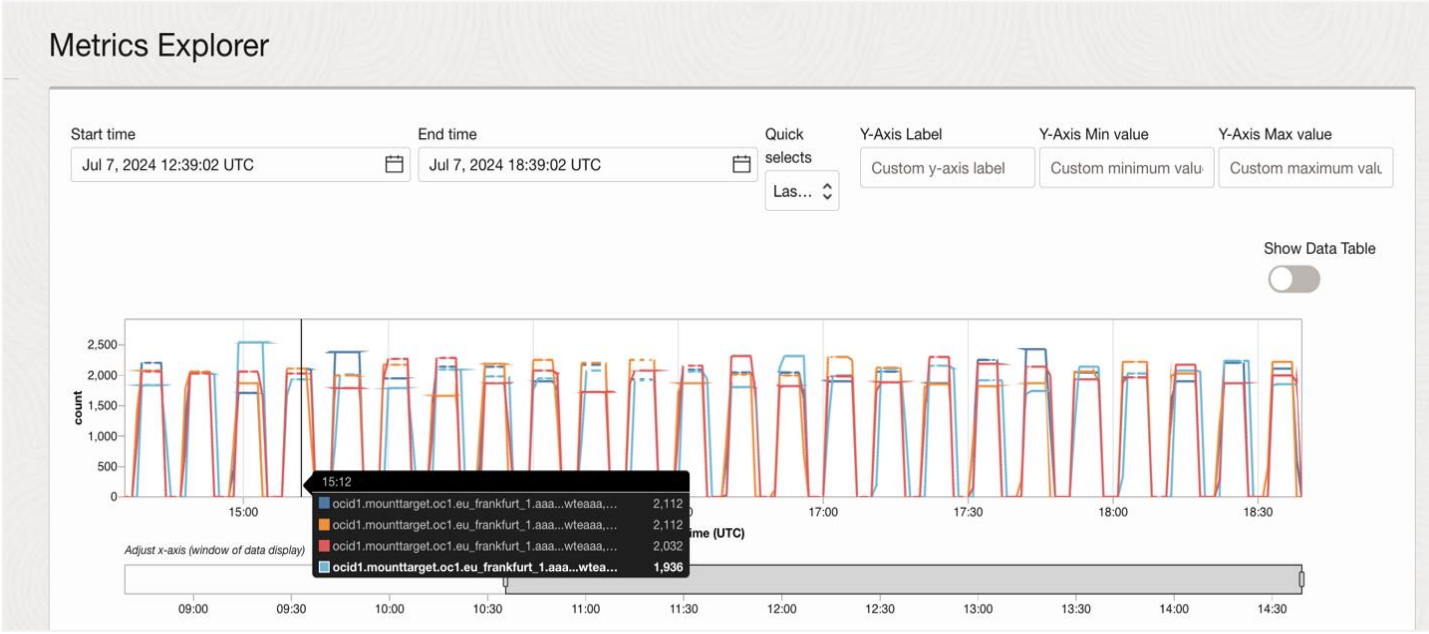

Figure 7. Cycling the Scaling of Pods to 512 and 0

Best Practices

Because of the address selection mechanism described in RFC3484, the mount targets should be in a different subnet than the NFS clients. Otherwise, the IPv6 should be disabled on NFSv4 or the `/etc/gai.conf` should be configured to disable the sorting of IPv4 addresses by getaddrinfo().

The DNS round robin technique is effective when there are many NFS clients, and all the clients are managing a similar workload. It's effective if the ratio of the number of clients to the mount targets is larger than 8—for example, a ratio of 128 with a total of 512 NFS clients and 4 mount targets, as shown in the example in this paper. The load on each mount target can be monitored by using mount target metrics. If one mount target is observed to be overused, as a temporary measure, keep the mount target removed from the DNS record for the file system to avoid new mounts from the mount target until the load gets balanced. For more information about monitoring File Storage, see OCI File Storage Performance Characteristics.

Summary

DNS round robin is an effective technique in distributing load across multiple mount targets. In OCI, you can configure DNS round robin by using the fully managed DNS service for VCNs. The example explained in this paper demonstrates the effectiveness of DNS round robin for load balancing. This technique can be used in both OCI Kubernetes Engine (OKE) and non-OKE environments.

ORACLE

Connect with us

Call +**1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at **oracle.com/contact**.

blogs.oracle.com facebook.com/oracle twitter.com/oracle