PURESTORAGE®
Uncomplicate Data Storage, Forever

TECHNICAL WHITE PAPER

# Oracle RMAN Backup and Recovery with FlashBlade®

Why FlashBlade//S™ from Pure Storage® is the ideal solution for the fast recovery of Oracle databases

# Contents

## Introduction

Oracle Recovery Manager (RMAN) is a popular backup and recovery tool that many Oracle database administrators leverage as part of a strong backup plan. RMAN's strong and adaptable foundation makes Oracle database backup and recovery possible.

Significant features of the RMAN include parallelized backup and recovery operations, incremental backups, centralized cataloging, and corruption detection. The time it takes to back up and recover a database using secondary media is the main difficulty that every Oracle database administrator encounters, even though RMAN may be the most adaptable software available. It is getting more difficult to meet backup and recovery service-level agreements (SLAs), especially with databases continuously increasing in size. Additionally, the organizations of today need to meet strict recovery point objectives (RPO) and recovery time objective (RTO) targets to satisfy regulatory standards.

While storage and backup vendors have focused on only the backup aspect and used a variety of methods, including compression algorithms, to reduce the amount of space needed on spinning media and speed up metadata processing using flash memory storage. Despite having optimized backup windows, these purpose-built appliances suffer greatly during database restoration because of the need to seek data and decompress it. Hard drives perform poorly as a result of the massive random I/O access patterns. Therefore, when nearly every application involved in the ecosystem requires recovery at the same time following a disaster, it becomes extremely vital to determine how quickly a purpose-built appliance can handle it.

Organizations need to quickly recover data in the event of a disaster. Keeping in mind that the growth of data has become the essence of an organization, this also introduces the term "fast backup/recovery" to a new class of solution area, bringing in the requirement to have high performance storage to support it.

The all-flash, scale-out architecture of FlashBlade//S™ from Pure Storage® is the ideal solution for the fast recovery of Oracle databases by providing NFS and object storage as a target for backups. It also provides faster network connectivity (up to 800Gbps) to handle huge backup traffic from Oracle servers.

Even with a well-defined backup and recovery strategy using Oracle RMAN, it is difficult to achieve the required recovery time objective with a slower target backup storage. FlashBlade//S provides a single solution for Oracle backup workloads by providing target protocols like NFS, SMB, and object storage, including simultaneous single or multi-stream backups and recoveries for multiple databases to achieve RTOs and RPOs.

## Pure Storage FlashBlade//S

Pure Storage FlashBlade//S is the next generation of enterprise scale-out unified fast file and object (UFFO) storage and delivers rich data services with high density, capacity, performance, and scalability to meet the needs of modern applications. Using a distributed metadata architecture, FlashBlade//S offers multi-dimensional performance on a consolidated platform with native NFS, SMB, and S3 protocol access.

**Modular architecture**: FlashBlade//S has a unique modular architecture that enables organizations to unlock new levels of power, space, and performance efficiency using an all-QLC design. The architecture disaggregates compute resources from storage so that capacity and compute can scale independently for extreme flexibility in configuration. FlashBlade//S (Figure 1) is a customizable platform that gives you the ability to tailor your configuration for current workload requirements and to non-disruptively upgrade to meet future needs. You can upgrade components on a schedule or on-demand that is consistent with changing technologies to future-proof the system.

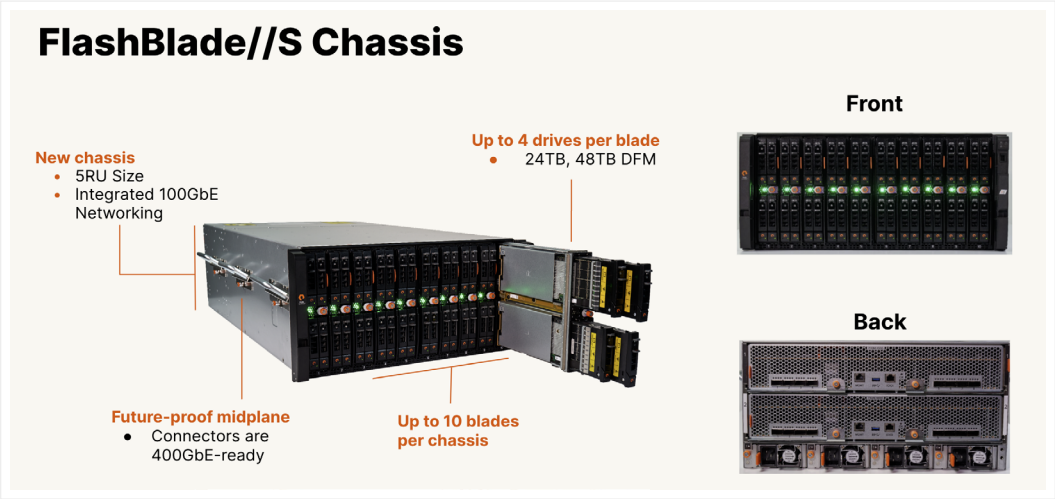


**FIGURE 1** FlashBlade//S

**Chassis**: The FlashBlade//S chassis is 5RU high and has bays for mounting 10 blades. Fully populated with high-density blades, a chassis holds 1.92PB of physical flash with headroom for future density increases.

The chassis midplane distributes power and has dual Ethernet links capable of operating at 100Gbps to each blade. Blades connect to two Fabric I/O Modules (FIOMs) on the midplane. The FIOMs have Ethernet switches that connect blades to the client network, or in multi-chassis systems, to external Fabric Modules (XFMs).

The chassis has four power supply units (PSUs), each rated at 2,400 watts. The PSUs are "2+2 redundant"—any two of them can supply the chassis' maximum rated power of 4,800 watts, to accommodate the greater demands of future components.

**Blade**: The blades have CPUs, NICs, DRAM, and four DirectFlash (DFM) mounting slots. Blades use NVMe over on-board PCIe to communicate with their DFMs. You can configure blades with either all performance-optimized (24TB) DFMs or all capacity-optimized (48TB) DFMs.

Blades operate with one, two, three, or four DFMs installed. All blades in the chassis must have the same number of DFMs in each blade. Systems can be configured for applications from the very read-intensive (e.g., artificial intelligence and machine learning) to those requiring extremely high capacity (e.g., backup and archiving).

The blade design isolates failure domains. You can replace a failed DFM from the front of the chassis without affecting its blade's ability to function. DFMs can move to different bays in a chassis to minimize the operational impact of recovering from blade failures or replacements.

**DFMs with QLC flash**: Architectures that use off-the-shelf solid-state drives (SSDs) have an internal controller to manage the flash media on each specific drive. These systems do not have any visibility into what is happening at the system level. FlashBlade//S takes a different and innovative approach by using DirectFlash Modules (DFMs) that enable the storage operating system to manage the media on a global level. Global media management unlocks as much as 20% more capacity from NAND compared to systems that use off-the-shelf SSDs, and delivers more consistent performance, better reliability, and higher media endurance without the need for a massive and expensive storage class memory (SCM) cache.

**Network fabric**: The integrated networking in FlashBlade//S simplifies large-scale deployments by collapsing three networks (front-end, control, and back-end) into one high-performance software-defined networking (SDN) fabric. This SDN is shared across the two fabric modules in the platform, and it hides the complexity of networking from the administrator.

FlashBlade//S virtualizes the network so that no matter the size of the platform, it appears as one entity. This virtualization simplifies load balancing and cabling. Each blade can service and restart any client connection and run any protocol, and the platform is stateless because the logic can run anywhere.

Dual Fabric I/O Modules (FIOMs) interconnect blades, connect chassis (in multi-chassis systems), and connect blades to clients. The FIOMs have ethernet switches with eight (8) external ports each capable of 10, 25, 40, or 100Gbps transmission rates. The switches have a total of 2Tbps cross-sectional bandwidth. Each FIOM uses 50Gbps for inter-blade communication in the chassis. Both FIOM switches and blade NICs are capable of 100Gb/s for future expansion.

**Purity//FB**: The Purity//FB operating system is the heart of FlashBlade//S, enabling scalability in capacity and performance. Purity//FB is all-inclusive software with enterprise-grade data services. The design of Purity//FB optimizes the power of the hardware with its variable block metadata engine and scale-out metadata architecture. It can manage billions of files and objects and deliver unmatched performance for any workload, whether it's sequential or random access with large or small files and objects. Purity//FB delivers a rich set of enterprise capabilities including compression, global erasure coding, always-on encryption, SafeMode™ Snapshots, file replication, object replication, and multiple other features.

**Environmental Efficiency**: More than ever, environmental, social, and corporate governance (ESG) initiatives are important. As a result, space and power constraints are becoming crucial considerations in storage strategy. The architectural design of FlashBlade//S uncomplicates data storage to enable a lower carbon footprint. It is designed to save data center space and reduce energy consumption.. This creates a storage solution that delivers a significant and immediate impact on the environment while lowering overall TCO.

### FlashBlade//S and Evergreen//Forever

The modular design of FlashBlade//S simplifies adding storage capacity and compute resources with non-disruptive hardware upgrades. This has made it possible for Pure to offer Evergreen//Forever™ service for the new systems. Evergreen//Forever has several advantages, including the Forever Flash lifetime media guarantee, Ever Agile upgrades with trade-in credits, "flat and fair" service pricing guarantees, and periodic hardware refresh at no incremental cost.

New generations of storage hardware typically appear every three to five years. Historically, this meant that every three to five years users would effectively repurchase capacity they already owned and must migrate hundreds of terabytes of data from old to new hardware. Evergreen//Forever plus the non-disruptive upgrades on FlashBlade//S enable the system to keep pace with hardware evolution. FlashBlade//S gets better over time.

## How to Use This Guide

This document focuses on Oracle Database backup and recovery using the Oracle RMAN utility to FlashBlade. This technical white paper is intended to be used as a how-to and best practice guide when protecting Oracle Database with FlashBlade as backup storage.

The target audience for this technical white paper are specialists like DBAs and database architects using native Oracle RMAN for their backup and recovery needs.

## Oracle Database

A relational database refers to the organized collection of structured data and is often called relational database management system or RDBMS. The purpose of a database is to store and retrieve related information.

Oracle databases are relational databases and they are one of the most popular relational database engines for storing, organizing, and retrieving data while still maintaining relationships between the various types.Oracle database was designed for enterprise grid computing and data warehousing. It is cross-platform and can run on various hardware across a wide range of operating systems, including Windows Server, Unix and various distributions of GNU/Linux. It uses SQL queries as a language for interacting with the database.

Oracle databases come in many different editions. For more information on Oracle editions, please refer to Oracle Database Licensing Information User Manual.

A few of the noticeable features of Oracle databases are:

- **Scalability and performance**: Features like real application clustering, standby database, and portability help control the consistency of data and concurrency.
- **Availability**: Oracle provides the all-time data availability required by real-time applications so that data is available during the time of planned or unplanned downtimes and failures.
- **Backup and recovery**: Oracle provides features to recover data from almost all kinds of failures. During the recovery only the data getting recovered is unavailable while all other data is available for use.
- **Security**: Oracle databases provide mechanisms to control data access and usage. Properly implemented authorization and editing user actions can prevent unauthorized access and can allow only authenticated access to the users.

### Oracle Recovery Manager

Oracle Recovery Manager (RMAN) is a utility that enables effective backup and recovery of Oracle databases. RMAN is available free of charge and comes with Oracle software installation.

The default data repository for RMAN (which contains backup and recovery metadata) is the controlfile of the Oracle database that is being backed up. Archive and backup information are stored in the reusable section of the controlfile. Oracle also provides the option of saving RMAN metadata in an RMAN catalog database, which is the preferred approach when dealing with hundreds of databases with varying retention period policies.

Some of the unique features of RMAN that Oracle DBAs depend on are:

- Corrupt blocks detection in data files
- The ability to back up a database and its components without putting the database in "hot backup" mode, as required in user-managed backups
- A central repository catalog to track backups across all Oracle databases within an organization
- Cross-platform data conversion (useful for migration)
- Flexible recovery options at the database, tablespace, data file, table, or block level
- Parallelized backup and recovery using multiple processes
- Validation and testing of backups and restores without performing the activity

## Solution Architecture

The purpose of this technical white paper is to test the backup and recovery performance of Oracle database when using FlashBlade//S as backup target. For this solution test, Oracle databases have been deployed on Pure Storage FlashArray™ and have been configured to back up to FlashBlade over the ethernet network.

Oracle recovery manager (RMAN) was used to perform backup and recovery operations. RMAN scripts were written to perform backups of Oracle Pluggable database keeping the test server and database in mind.

This solution architecture outlines the scenarios of performing backups and restores with FlashBlade while using NFS, Oracle dNFS and Object storage and the impact of having single vs multiple backup targets.

### Hardware Environment

Two separate Oracle environments have been created for this testing.

- Eight standalone Oracle instances, each on its own server
- A single six node RAC database

Each Oracle server was equipped with 32 CPU cores and 128GB of RAM. Database storage has been carved out from a Pure Storage FlashArray//X™ connected over a fiber channel interface. A FlashBlade//S has been used as a target for backups, connected over an aggregated 800Gbps ethernet channel.

Tests are executed for standalone and real application cluster (RAC) databases. Figures 2 and 3 below show the details of standalone and RAC database test lab topology. In both the cases, backup and storage networks are completely isolated from each other.

The standalone databases have Fibre Channel connections to the storage network and ethernet connections to the backup network. The below figure shows the layout of typical standalone Oracle instances.

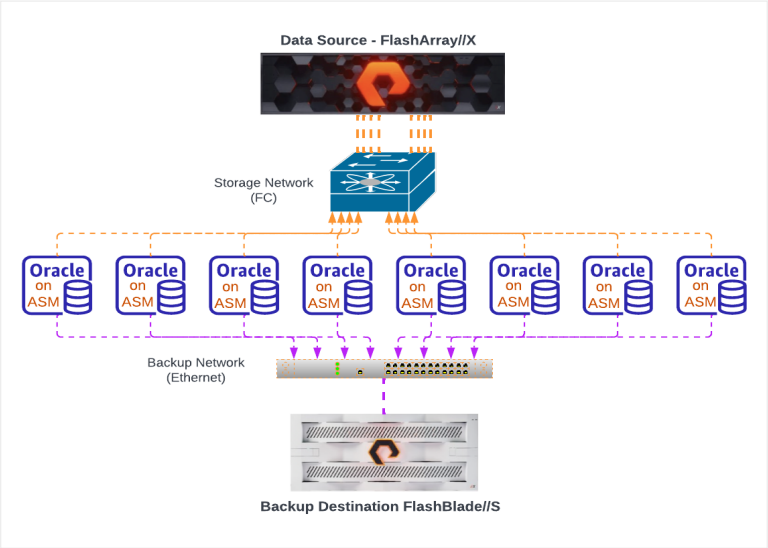


**FIGURE 2** Oracle standalone instance environment

For RAC database testing, the ASM disk groups were kept on FlashArray//X™ which was shared by all the RAC nodes.

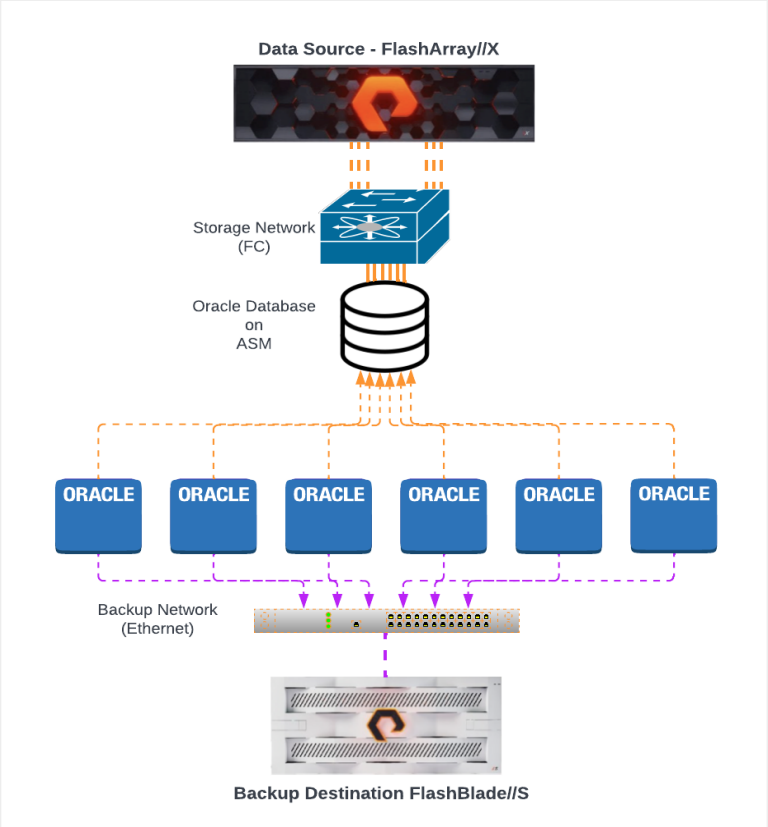


**FIGURE 3** Oracle RAC instance environment

## Software Environment

We used Oracle Linux version 8.6 as base OS running Oracle database version 21c.

### Standalone Database Layout

Each Oracle database has been created on top of three ASM disk groups:

- **CRS**: Cluster ready services metadata
- **Data**: Database files
- **FRA:** REDO logs and archived logs

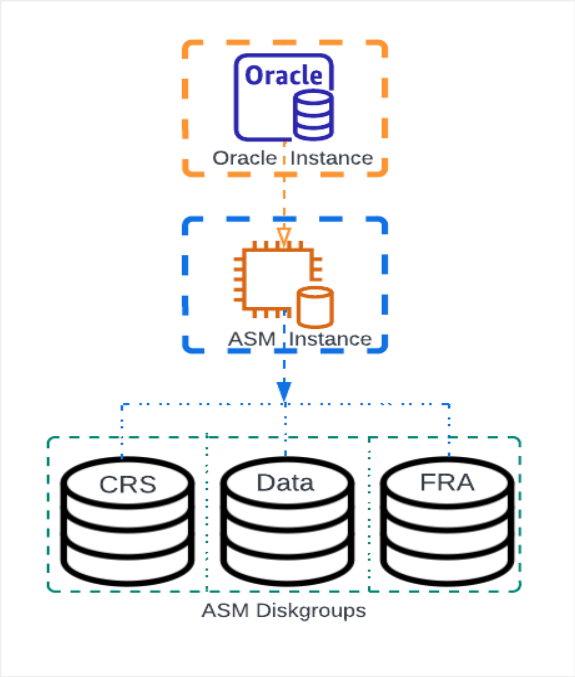


**FIGURE 4** ASM diskgroup layout for standalone databases

One container database (CDB) and one pluggable database (PDB) was created on each Oracle server, with a bigfile tablespace. The database size used with each instance was 1.1 terabytes.

### RAC Database layout

A six-node RAC cluster was created for the testing with ASM disk groups used to host the database.

- **OCR:** Oracle cluster registry metadata
- **Data:**Database files
- **FRA:** REDO logs and archived logs

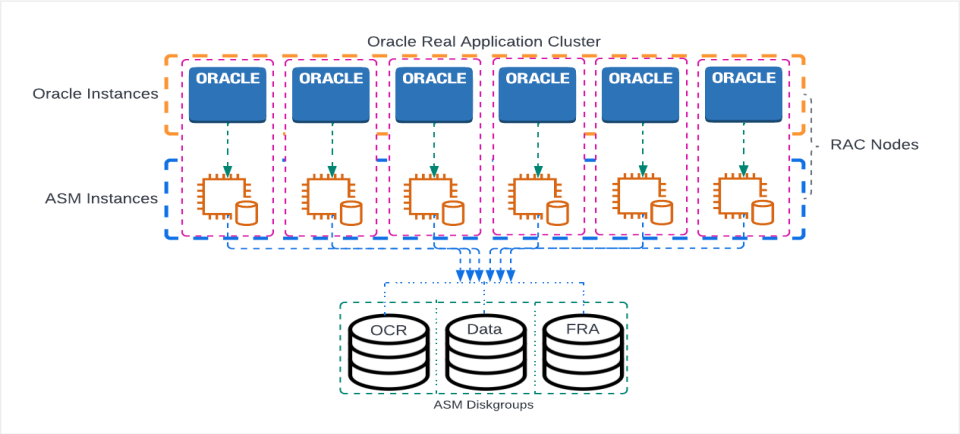


**FIGURE 5** ASM diskgroup layout for RAC database

Similar to standalone instances, we created one CDB and one PDB has been created on each Oracle server, with a bigfile tablespace. The database size of each instance was 1.1 terabytes.

### Backup and Recovery Storage

FlashBlade//S has the ability to store both files and objects. There are a number of configuration options available when file storage (in this context, NFS and object) is necessary for backup and recovery. We used RMAN to send backups to NFS and S3 storage:

- **NFS configuration**: Four NFS exports with default settings were created on FlashBlade to service RMAN backup and recovery operations.
- **Object storage**: A single storage account with one bucket was utilized for RMAN S3 based backups.

## Test Methodology

RMAN provides several options for backing up and restoring databases.

The performance of full database backup and recovery operations mainly depends on the following variables:

- Several tunable parameters, such as number of channels, FILESPERSET, and MAXOPENFILES can impact the performance of backup and recovery operations. We strongly encourage customers to calibrate their systems to find optimal settings that meet their requirements. This white paper calls out only the parameters used in our testing.
- We determined that 16 RMAN channels was the optimal number for the test environment used in this technical white paper.
- We used one and four NFS mounts for backups and recoveries as using more than four NFS mounts did not show any further improvement in performance.

By adjusting the aforementioned parameters, the testing detailed in this work focuses on describing the performance of database backup and recovery.

### Impact of Number of Backup Targets

The effective backup speed of an Oracle database depends upon the number of target NFS mounts used for backup.

We found that four NFS exports was an optimal number for getting good backup performance with our test setup.

The FlashBlade tries to assign each backup file stream to one of its blades using an in-built load balancing algorithm from front-end processing.

The load-balancing algorithm uses unique hashes generated from the source IP address, port, destination IP address, and ports. So, creating multiple virtual IP addresses (VIPs) is recommended on FlashBlade to distribute the load to more blades and achieve higher performance.

## Backup and Recovery Using RMAN

A variety of backup and recovery scenarios were tested for single, multiple and RAC Oracle instances including backups to NFS and Object storage.

Please note the "SECTION SIZE" clause used in all backup RMAN scripts; it broke a single DBF file into multiple pieces to provide work to each RMAN channel and thus helped to finish the backups faster.

Because a bigfile tablespace with a single database file of 1.1TB size has been used during the tests, we used a "SECTION SIZE" of 68G for proper distribution of the work among 16 RMAN channels, using simple math (1.1TB divided by 16).

Please remember that backups can be executed after connecting to CDB and PDB both by changing the backup clause as below but recoveries of PDB have to be executed after connecting to CDB only.

The Backup clause after connecting to CDB was :"`BACKUP PLUGGABLE DATABASE <pdb_name>`.

The backup clause after connecting to PDB was `BACKUP DATABASE`.

In all the tests, backups have been done after connecting to PDB.

Please refer to Oracle documentation for details on how to use RMAN commands to perform backups and recoveries.

### Backup/Recovery to NFS exports

For Oracle on Linux, backup to NFS exports is the most convenient method. Tests to single and 4 NFS exports have been performed to demonstrate the backup and recovery throughputs. All the backups have been taken at level FULL. For recommendations on optimal NFS mount options for RMAN backups, please refer to the latest Oracle documentation. In these tests default NFS mount options have been used.

### Standalone Database

**Using Single NFS export**: This is the standard way to backup the Oracle databases by DBAs to one NFS export. One NFS export was mounted on an Oracle server using one data VIP and a backup and recovery was performed. Backup to a single NFS export did not yield better results while using more than 16 RMAN channels.

The below figure explains the NFS architecture with 1 NFS export.

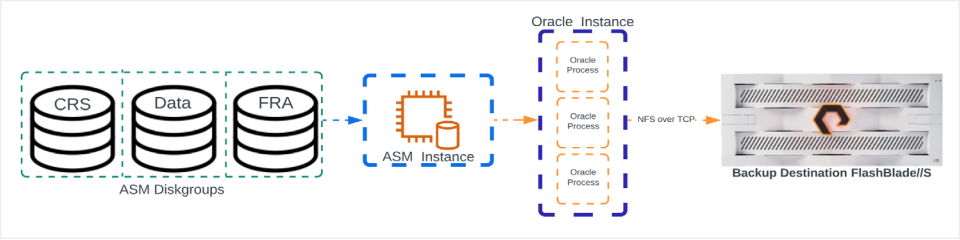


**FIGURE 6** NFS with one export

Refer to Appendix 01 for RMAN scripts used to perform backup and recovery.

**Using fourNFS exports**: Using four NFS exports helped to stream backups and recoveries to and from four NFS mounts concurrently. There was a gain in backup and recovery performance over backup to single NFS mount. Four different NFS exports have been mounted using 4 different data VIPs on FlashBlade to distribute the load. We used the "FORMAT" backup clause to specify different NFS mounts.

The below figure explains the NFS architecture with four NFS exports.

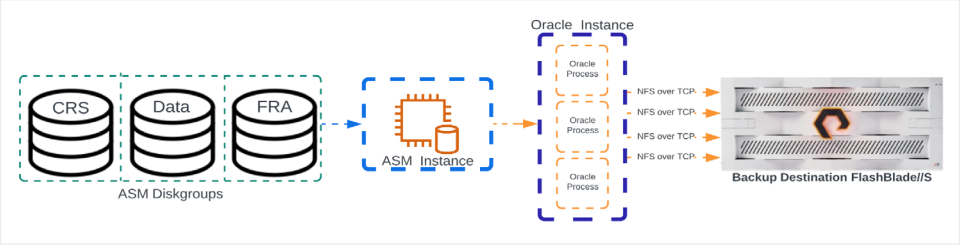


**FIGURE 7** NFS with 4 exports

Refer to Appendix 02 for RMAN scripts used to perform backup and recovery.

### Multiple Databases

All the multi-database tests have been performed with 4 NFS exports. The RMAN scripts used for backup and recovery are similar to "4 NFS exports'' scripts in the above section. Backup and recovery have been executed concurrently on all the Oracle servers to measure the performance.

## Backup/Recovery Using Oracle dNFS

Oracle Direct NFS Client integrates the NFS client functionality directly in the Oracle software to optimize the I/O path between Oracle and the NFS server. This integration can provide significant performance improvements.

Direct NFS (dNFS) Client supports NFSv3, NFSv4, NFSv4.1, and pNFS protocols to access the NFS server. Direct NFS Client also simplifies, and in many cases automates, the performance optimization of the NFS client configuration for database workloads. Please refer to the article here for enabling and disabling dNFS on Oracle.

To use Direct NFS Client, the NFS file systems must first be mounted and available over regular NFS mounts. A file named "oranfstab" needs to be created to allow Oracle to determine the available mount points. Please refer to the Oracle article here for mode details on the "oranfstab" file.

Here is a sample oranfstab file:

```
Server: FlashBlade
path: 10.31.201.101
path: 10.31.202.101
path: 10.31.203.101
path: 192.31.204.101
nfs_version: nfsv4.1
export: /rman1 mount: /m01
export: /rman2 mount: /m02
export: /rman3 mount: /m03
export: /rman4 mount: /m04
```

Backup and recovery scripts used for dNFS are similar to the scripts used for NFS backup and recovery. The connections made to FlashBlade are controlled by Oracle based on entries in the oranfstab file. You can run command netstat -an|grep 2049 on Oracle host to check the number of connections made to FlashBlade.

The below figure shows the dNFS architecture.

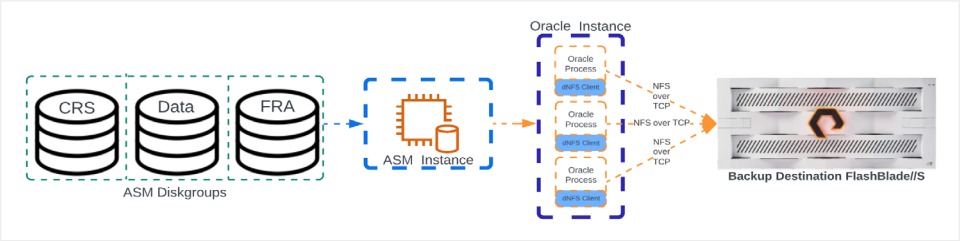


**FIGURE 8** Oracle dNFS architecture

## Backup/Recovery to Object Storage

Please refer to solution Oracle RMAN Backup to FlashBlade Object Store using the S3 API for step-by-step procedure on Configuring S3 backups with Oracle and FlashBlade.

Backup and recovery scripts used for single and multiple databases are the same. For multiple database backups, the same script has been executed concurrently on all the database hosts.

Refer to Appendix 03 for RMAN scripts used to perform backup and recovery.

## Backup and Recovery of RAC Database

A backup of recovery of a six-node RAC database has been tested. A RMAN script was used to connect all six Oracle RAC instances to distribute the load for backup. The backups have been performed to NFS with four mounts and four VIPs as above.

To protect your Oracle Real Application Clusters (Oracle RAC) database from hardware failures or disasters, you must have a backup copy of the database files. The files protected by the backup and recovery facilities built into Oracle Recovery Manager include data files, control files, server parameter files (SPFILEs), and archived redo log files. You can use these files to reconstruct your database. The backup mechanisms that work at the logical block level protect against damage at the tablespace and database level, such as the accidental deletion of data or the failure of a disk drive. Database recovery involves restoring the damaged files from backup and performing media recovery on the restored files. Media recovery is the application of redo logs or incremental backups to a restored data file to update it to the current time or some other specified time. Please refer to Oracle documentation for more detailed information.

Be aware that while we use the `connect` clause in the backup script, the RAC nodes only participate in the read part of the backup process. The data is read from all the nodes and sent to the node owning “rman” process and from there it is sent to the NFS target. In that case the write throughput is similar to a single database backup as data is sent using only one node of the cluster.

Recoveries are performed from one of the nodes in the cluster.

Refer to Appendix 04 for RMAN scripts used to perform backup and recovery.

## Object Replication with FlashBlade//S

FlashBlade object replication can be used to asynchronously copy objects in a bucket to a different FlashBlade or Amazon S3. Objects are automatically replicated as soon as they are written to the source bucket, with no need to manage complex policies. This replication provides disaster recovery capabilities with ongoing protection against site or data center failures as well as eliminating the need for additional hardware when implementing object copy to Amazon S3.

Please refer to the FlashBlade User Guide, Appendix B, Replication and Recovery for information on how to implement and use object replication on FlashBlade in .

To recover an Oracle database from a replicated object store, you must register the replicated object store using the procedure mentioned in Configuring S3 backups with Oracle and FlashBlade in Oracle RMAN Backup to FlashBlade Object Store Using the S3 AP and recover the database using the method mentioned in Backup and Recovery Process with Object Store.

## Test Cases and Results

The section Backup and recovery using RMAN covers the backup and recovery of Oracle database in detail. Organizations can choose the appropriate backup and recovery strategy which suits their requirements. This section analyzes the results for each backup target utilized during the tests.

The NFS and dNFS scenarios were tested with 16 channels. Tests fewer than 16 channels did not demonstrate optimum performance. For backup and recovery with the Object store, tests have been performed with 32 channels. Tests fewer than 32 channels did not demonstrate optimum performance.

Both single and multi-instance scenarios will be assessed for each scenario. The following method is used to determine performance in any of the scenarios:

- The date command has been used to capture the start and end time to the database recovery; the recovery portion has not been considered as it is Oracle dependent and does not depict the actual recovery throughput.
- Divide the amount of database data backed up/recovery by the amount of time that has elapsed in seconds. The MB/s value provided is then converted into TB/hour.

### Standalone Database Tests: Single Instance Performance Results

Single database backups are performed while doing the ad-hoc backups or to find out the optimum backup and recovery parameters many of the times. Single database tests demonstrate the best practices to achieve the highest backup and recovery performance.

We performed tests including backups and recovery with NFS, dNFS and Object Store with a varying number of channels and NFS mounts.

The next section outlines the results and analysis of a single Oracle server hosting a PDB instance.

#### Backup and Recovery to NFS

One NFS export as target: Backup of a single database has been performed using a single NFS mount while varying the number of RMAN channels. The results below showcase how using differing RMAN channel configurations impact the overall performance for a single instance.

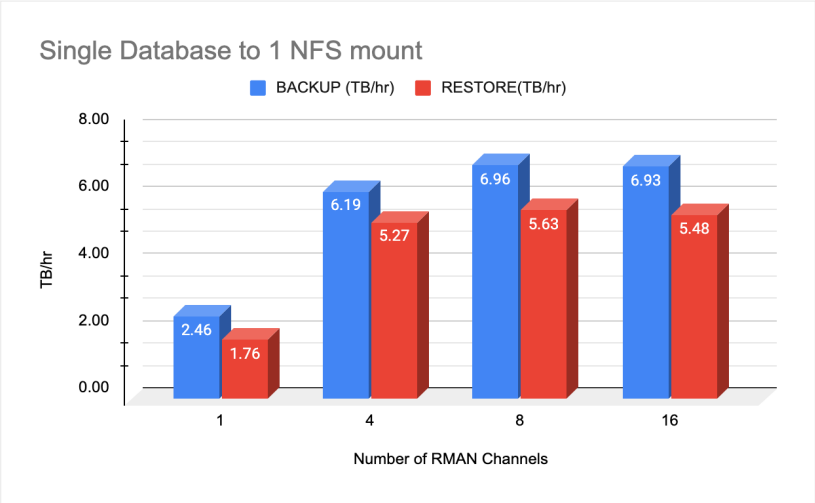


**FIGURE 9** Single database backup and restore to 1 NFS mount

The above graph demonstrates that the tests with eight RMAN channels show the optimum backup and recovery performance. Backups to single NFS exports were done to demonstrate data transfer performance with a single network connection to FlashBlade.

Four NFS exports as target: In this test scenario, a single database was backed up to four NFS exports mounted on the Oracle server. We performed the tests with 16 and 32 RMAN channels.

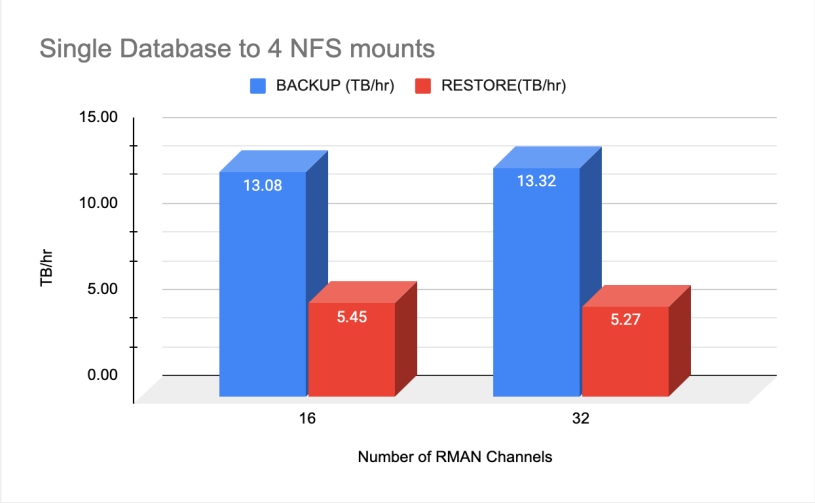


**FIGURE 10** Single database backup and restore to 4 NFS mounts

When comparing the results of backup and recovery to one NFS mount vs. four NFS mounts, it is clearly visible that backups to four NFS mounts performed better. Using more than four NFS mounts did not show any further improvement in performance.

### Backup and Recovery using Oracle dNFS feature

Back and recovery tests have been performed using four network paths and four NFS exports using Oracle dNFS feature, which manages the NFS connections on its own to optimize performance. Comparing the dNFS results with four NFS export tests above shows that both the tests produced at par results.

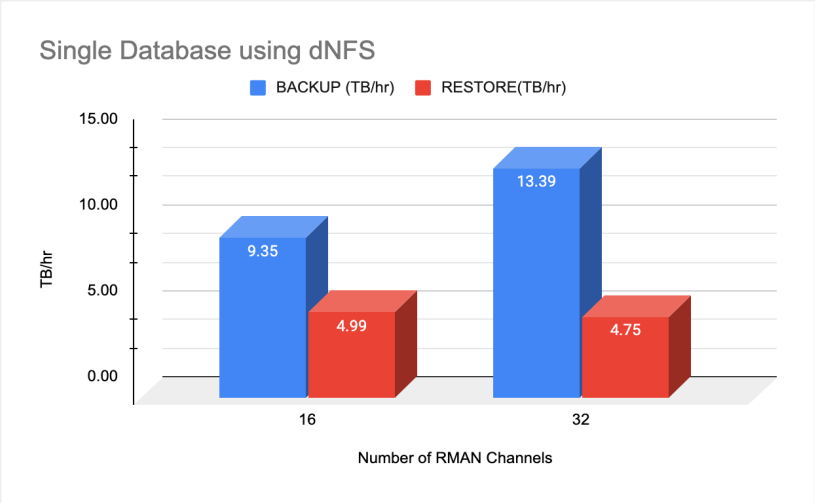


**FIGURE 11** Single database backup and restore with dNFS

While backups were faster with 32 RMAN channels, restore did not show any performance improvement.

### Backup and Recovery to Object Store

FlashBlade provides S3 compatible object storage as a built-in feature for the applications and databases which have capability to read and write from object storage. RMAN provides capability to backups and recovery with object store. Please read theOracle documentation to check the cost involved with using RMAN with object storage.

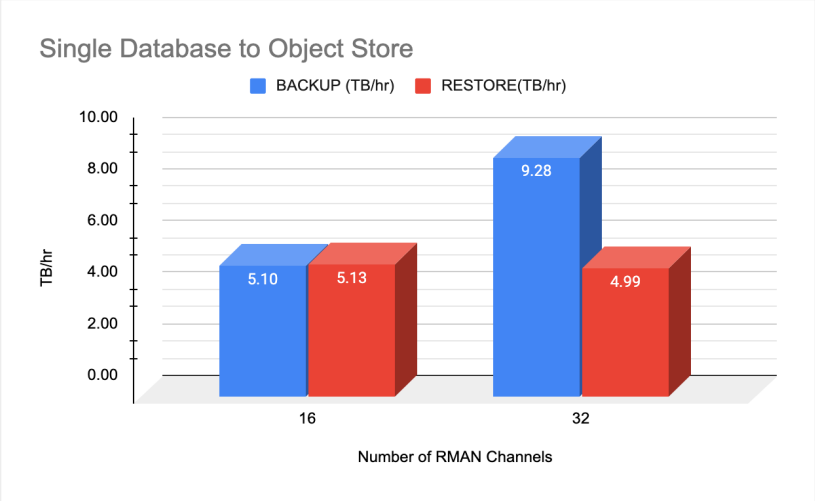


**FIGURE 12** Single database backup and restore with Object Store

The test results were promising with Object Storage as target and can be compared with dNFS tests with 16 RMAN channels.

## Standalone Database Tests: Multiple Instance Performance Results

Multi-server tests have been performed to replicate the scenario where DBAs want to take backups of many databases and servers simultaneously. The objective of multi-server tests is to demonstrate the capability of the FlashBlade to handle multiple streams together.

### Backup and Recovery to NFS

Multi Server tests have been performed with four NFS exports. Best recovery performance was eight servers restoring data together using 16 channels.

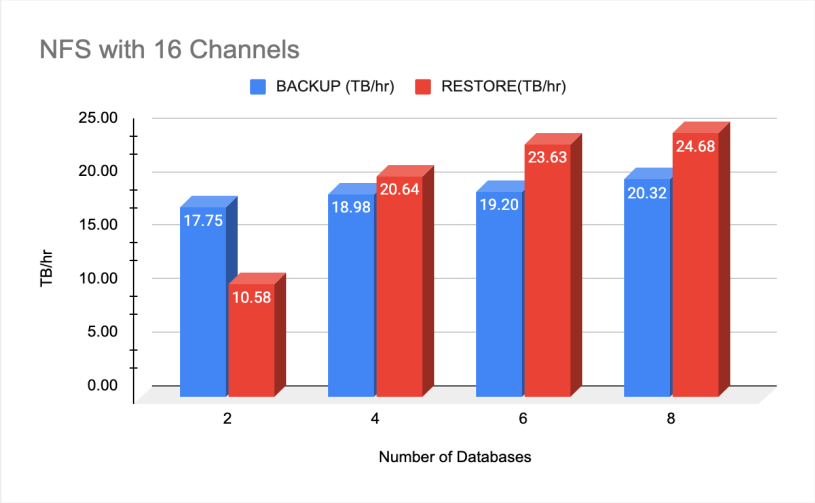


**FIGURE 13** Backup and restore to NFS with 16 channels

While comparing the results with 16 channels above there are improvements in backup throughput. It demonstrates that FlashBlade handles the traffic from multiple servers efficiently.

### Backup and Recovery with dNFS

The below chart shows the results for backup and recovery with Oracle dNFS configuration. It clearly shows that the performance is similar to the 4 NFS exports configuration above. RMAN channel count and other parameters should be tuned, based on database size and available resources.

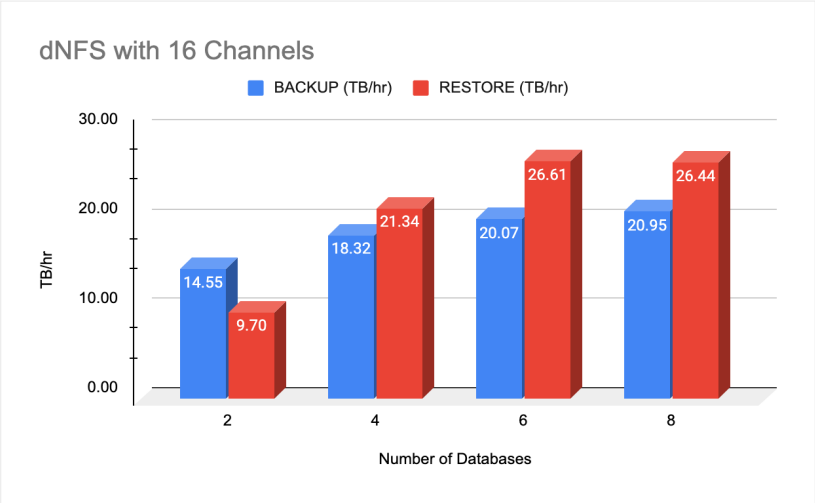


**FIGURE 14** Backup and restore to dNFS with 16 channels

Backups and recoveries with Oracle dNFS definitely outperformed NFS and Object Storage when comparing six and eight server results. Oracle dNFS creates multiple network connections per NFS export to transfer data to NFS storage, thus improving performance.

### Backup and Recovery with Object Storage

Testing backups and recovery with object storage show very interesting results where increasing the number of RMAN channels to 32 increases recovery performance. A single FlashBlade VIP is used to connect to the object storage endpoint.

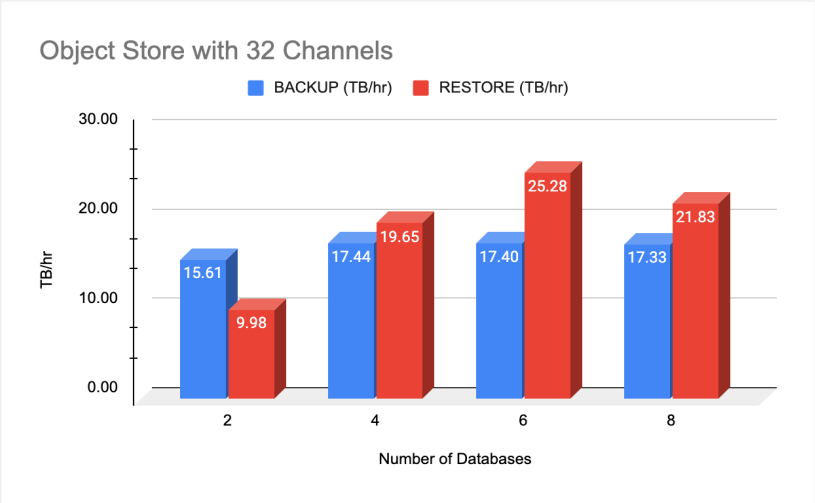


**FIGURE 15** Backup and restore with Object store using with 32 channels

The recovery performance with object storage was good because object storage is known to have better read performance over write performance.

### Backup and Recovery of Real Application Cluster (RAC) Databases to NFS

Backup and recovery of RAC databases can be compared to a single instance of Oracle database backups. The reason is that only the one node that owns the RMAN process sends data to target storage. RMAN channels can be allocated to connect all the RAC nodes to read data and transfer is over to the Oracle node owning the RMAN process to distribute the load of the backup process.

## Ransomware Remediation with SafeMode Snapshots

SafeMode Snapshots create a separation of power between FlashBlade administration functions and the eradication of data. SafeMode Snapshots only apply to file data and can be combined with replication to offer superior resilience to data loss and integrity concurrently.

In this scenario it is applicable to NFS storage only. Snapshots can not be used with OSB.

File system snapshots on FlashBlade are immutable and cannot be changed. A new file system can be created from a snapshot, but the state of the snapshot will remain unchanged from the moment it was taken. This assists with ransomware remediation because data cannot be changed or affected by malicious software attempting to encrypt it, providing a recovery point that organizations can be assured has not been compromised.

**NOTE:** *For regular operations manually creating the snapshots is cumbersome but it can be automated easily by using APIs. Please refer to* FlashBlade Object Store Documentation (S3 API) *for more information on how to use FlashBlade APIs to accomplish this.*

### Define the Appropriate SafeMode Snapshot Policy

SafeMode Snapshot policy will vary based on your needs. There are two key policy settings that need to be defined: schedule and retention. Careful consideration is required when deciding what values to use since the policy applies to all file systems on the FlashBlade.

The snapshot schedule should align with the end of the backup window to minimize data loss if a rollback is required after a ransomware attack. For example, if backup typically finishes at 7:00 a.m., the SafeMode Snapshot policy should be scheduled to occur between 7:00 and 8:00 a.m. Snapshots can be scheduled to repeat multiple times per day if there are multiple backup windows or lower tolerance for data loss after an attack.

The snapshot retention value should be created with consideration of the necessary backup schedules to provide the required data availability to meet business requirements. This needs to be balanced against the additional storage required for each extra day of retention. Note that snapshots created outside policy, i.e. through the FlashBlade GUI, command line or REST API, will also be protected from eradication based on the SafeMode Snapshot policy, which may affect storage consumption.

The authorized administrator will work with Pure Storage Support to configure the SafeMode Snapshot policy on your FlashBlade.

## Creating a File System Snapshot

**NOTE:** *File system snapshots can easily be restored to overwrite existing file systems after which an Oracle Recovery Manager (RMAN) recovery can be performed.*

It is simple to implement the recovery strategy with SafeMode Snapshots: run the backups to FlashBlade NFS export and create the snapshots. The point-in-time protected snapshots can be restored back to have a clean copy of the backups.

To create a snapshot in the FlashBlade graphical user interface, a user with the correct permissions needs to be logged in, navigate to the Storage view, and then select an appropriate file system. Once a file system has been selected, you can create a snapshot by selecting the plus sign (**+**) in the upper-right-hand corner in the File System Snapshots section.

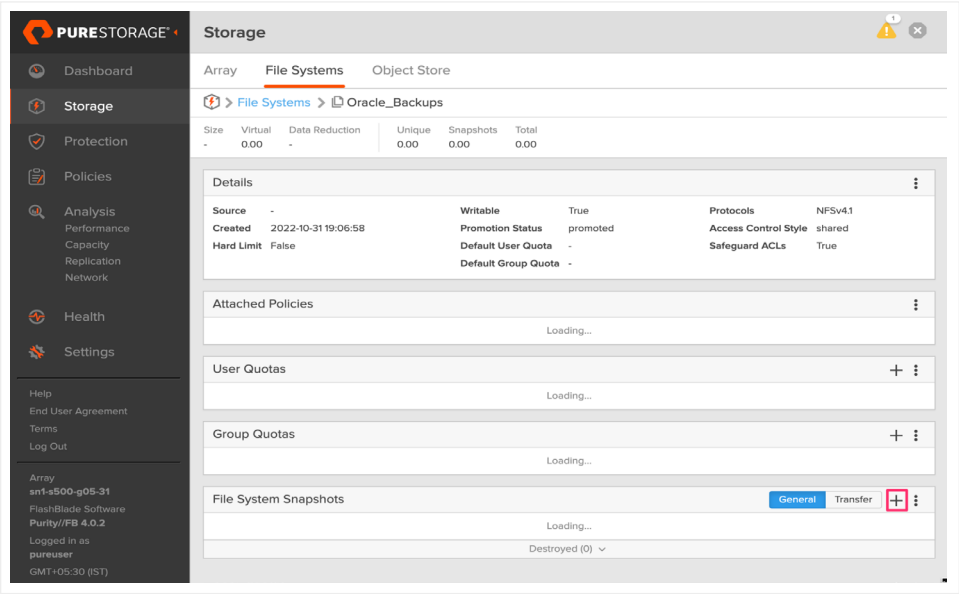


**FIGURE 16** FlashBlade graphical user interface file system view

Once the create snapshot dialog is open for a specific file system, an optional suffix can be provided before its creation. If you don't specify a suffix then the snapshot is created with <Filesystem name>_YYYY_MM_DD_HH_mm format.

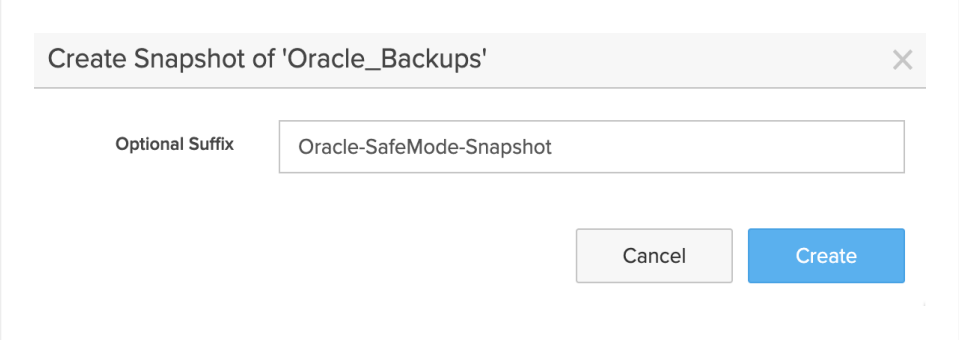


**FIGURE 17** Create a snapshot of file system dialog

Once the file system snapshot has been created, it will be displayed in the File System Snapshots section.

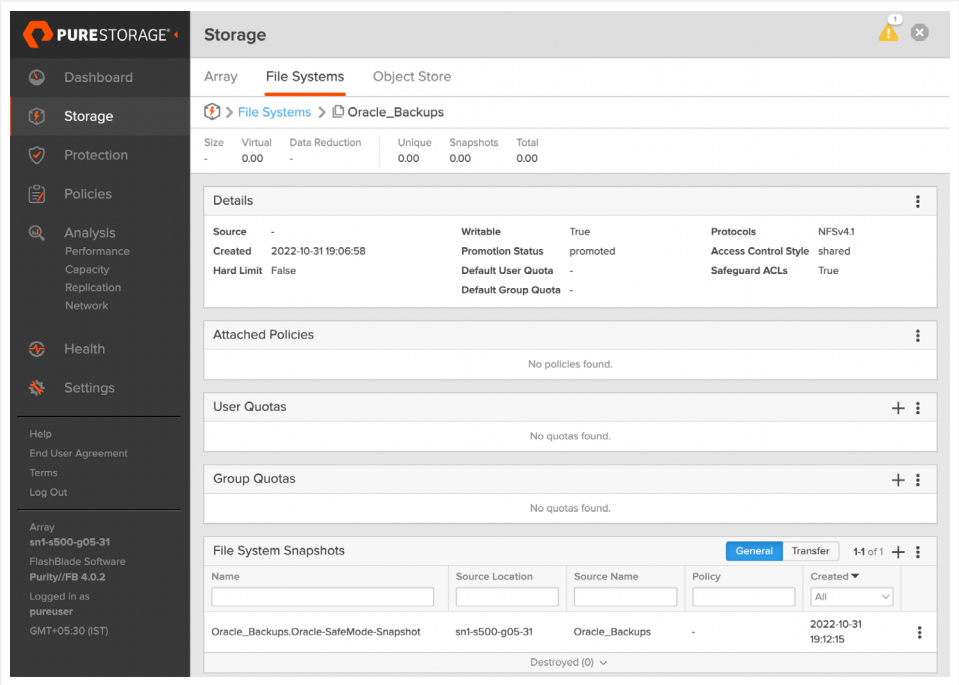


**FIGURE 18** View created, file system snapshot

## Enabling SafeMode Snapshots

SafeMode Snapshots will ensure that any snapshot will not be eradicated for a set period. This allows for a period where data can be recovered without being changed.

Only Pure Storage support can perform the required operations to configure SafeMode Snapshots. This ensures that no local user can compromise a system and disable the functionality without being a trusted contact of the organization.

When contacting Pure Storage support to enable SafeMode Snapshots, organizations need to provide a minimum period that snapshots must be retained for. This period will be used to specify how long file system snapshots need to be retained before they can be completely eradicated.

## Deleting and Eradicating a SafeMode Snapshot

In FlashBlade, there are two steps that must be performed before data is completely eradicated when SafeMode is not enabled:

1. A file system or snapshot must first be destroyed. The destruction simply places it in a queue that will delay the destruction of the data for a set period. During this time, the file system or snapshot can still be recovered and used as if it were never deleted.
2. Once destroyed, a file system or snapshot remains on the system until it is eradicated (deleted) by a user or a set period passes.

In summary, file system snapshots on FlashBlade are constrained, where data is not eradicated until a set period passes or a user manually eradicates it. SafeMode protects this data by blocking all manual eradication and enforcing the eradication timer.

With SafeMode Snapshots enabled, a file system snapshot can still be destroyed in the File System view:

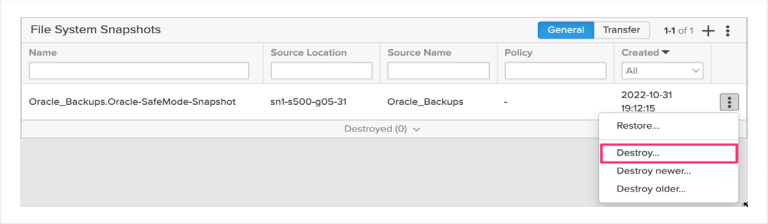


**FIGURE 19** Destroying a file system snapshot

Once destroyed a file system snapshot will show in the Destroyed Snapshots section.

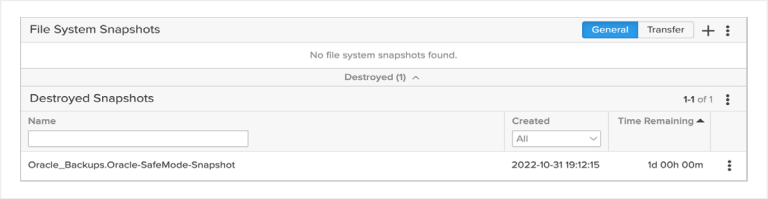


**FIGURE 20** Viewing destroyed file system snapshots

Selecting the three dots or vertical ellipses on the right-hand side of the destroyed snapshot will bring up the context menu. This menu will allow for the eradication or recovery of the snapshot.

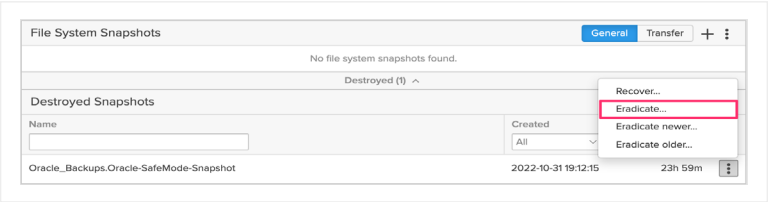


**FIGURE 21** Eradicating file system snapshot

A dialog is always shown before eradication.

**NOTE:** *When SafeMode Snapshots is enabled, this activity is blocked with an error of "Operation not permitted".*

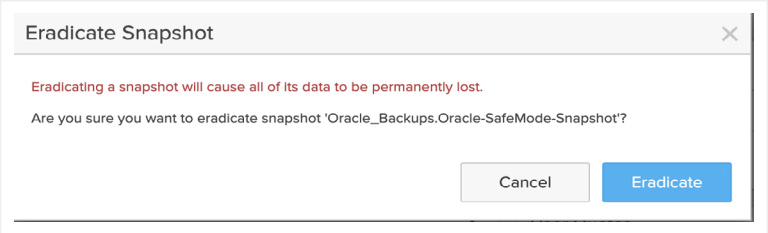


**FIGURE 22** Eradicate snapshot dialog box

## Best Practices and Recommendations

Database administrators need to evaluate and select the best set of options for their environment. Based on the tests completed for this paper, we recommend the following parameters and configuration options to improve backup and recovery performance when combining RMAN and FlashBlade:

- Having more NFS exports (in this case four) can provide good performance. DBAs can test and tune the number of NFS exports to use based on their requirements.
- Use Oracle dNFS over NFS. Oracle dNFS creates a separate connection to FlashBlade//S for every server process.
- Use multiple VIPs on FlashBlade to mount NFS exports, which provides better distribution of the streams across the FlashBlade blades.
- Use the appropriate number of Recovery Manager (RMAN) channels to speed up backup and recovery operations, paying attention to compute resources, database size, backup window and network bandwidth. Backups and recoveries to Object Store performed better with higher number of RMAN channels.
- Write optimized RMAN scripts with appropriate parameters to improve performance. Please refer to Oracle documentation for more details.

## Conclusion

The RMAN interface contains many options to protect Oracle databases; when combined with FlashBlade//S NFS and Object Storage, RMAN can be leveraged to provide recovery performance as much as several terabytes per hour. The state-of-the-art compression algorithm of FlashBlade//S provides three to four times the compression of Oracle's uncompressed backups.

Filesystem and object replication can be utilized to have a second copy of data which can be used in case the primary copy is not available.

Immutable SafeMode Snapshots can be used to mitigate the impact of ransomware and if enabled can get you up and running again quickly in the event of an attack.

## Appendix

**01.** RMAN scripts to perform backup and recovery using one NFS export.

**a.** Backup script:

```
RUN {
ALLOCATE CHANNEL c1 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server1/
db1_%U_%T';
ALLOCATE CHANNEL c2 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server1/
db1_%U_%T';
ALLOCATE CHANNEL c3 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server1/
db1_%U_%T';
ALLOCATE CHANNEL c4 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server1/
db1_%U_%T';
ALLOCATE CHANNEL c5 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server1/
db1_%U_%T';
ALLOCATE CHANNEL c6 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server1/
db1_%U_%T';
ALLOCATE CHANNEL c7 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server1/
db1_%U_%T';
ALLOCATE CHANNEL c8 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server1/
db1_%U_%T';
ALLOCATE CHANNEL c9 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server1/
db1_%U_%T';
ALLOCATE CHANNEL c10 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server1/
db1_%U_%T';
ALLOCATE CHANNEL c11 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server1/
db1_%U_%T';
ALLOCATE CHANNEL c12 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server1/
db1_%U_%T';
ALLOCATE CHANNEL c13 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server1/
db1_%U_%T';
ALLOCATE CHANNEL c14 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server1/
db1_%U_%T';
ALLOCATE CHANNEL c15 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server1/
db1_%U_%T';
ALLOCATE CHANNEL c16 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server1/
db1_%U_%T';
BACKUP SECTION SIZE 8G DATABASE mypdb;
RELEASE CHANNEL c1;
RELEASE CHANNEL c2;
RELEASE CHANNEL c3;
RELEASE CHANNEL c4;
RELEASE CHANNEL c6
RELEASE CHANNEL c6;
RELEASE CHANNEL c7;
RELEASE CHANNEL c8;
RELEASE CHANNEL c9;
RELEASE CHANNEL c10;
```

**b.** Recovery script:

```
RUN {
ALLOCATE CHANNEL c1 type DISK;
ALLOCATE CHANNEL c2 type DISK;
ALLOCATE CHANNEL c3 type DISK;
ALLOCATE CHANNEL c4 type DISK;
ALLOCATE CHANNEL c5 type DISK;
ALLOCATE CHANNEL c6 type DISK;
ALLOCATE CHANNEL c7 type DISK;
ALLOCATE CHANNEL c8 type DISK;
ALLOCATE CHANNEL c9 type DISK;
ALLOCATE CHANNEL c10 type DISK;
ALLOCATE CHANNEL c11 type DISK;
ALLOCATE CHANNEL c12 type DISK;
ALLOCATE CHANNEL c13 type DISK;
ALLOCATE CHANNEL c14 type DISK;
ALLOCATE CHANNEL c15 type DISK;
ALLOCATE CHANNEL c16 type DISK;
ALTER PLUGGABLE DATABASE mypdb CLOSE IMMEDIATE;
RECOVER PLUGGABLE DATABASE mypdb;
RECOVER PLUGGABLE DATABASE mypdb;
ALTER PLUGGABLE DATABASE mypdb OPEN;
RELEASE CHANNEL c1;
RELEASE CHANNEL c2;
RELEASE CHANNEL c3;
RELEASE CHANNEL c4;
RELEASE CHANNEL c5;
RELEASE CHANNEL c6;
RELEASE CHANNEL c7;
RELEASE CHANNEL c8;
RELEASE CHANNEL c9;
RELEASE CHANNEL c10;
RELEASE CHANNEL c11;
RELEASE CHANNEL c12;
RELEASE CHANNEL c13;
RELEASE CHANNEL c14;
RELEASE CHANNEL c15;
RELEASE CHANNEL c16;
}
```

**02.** The below RMAN scripts have been used to perform backup and recovery with four NFS mounts.

**a.** Backup script:

```
RUN {
ALLOCATE CHANNEL c1 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server/
db_%U_%T';
ALLOCATE CHANNEL c2 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server/
db_%U_%T';
ALLOCATE CHANNEL c3 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server/
db_%U_%T';
ALLOCATE CHANNEL c4 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt1/server/
db_%U_%T';
ALLOCATE CHANNEL c5 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt2/server/
db_%U_%T';
ALLOCATE CHANNEL c6 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt2/server/
db_%U_%T';
ALLOCATE CHANNEL c7 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt2/server/
db_%U_%T';
ALLOCATE CHANNEL c8 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt2/server/
db_%U_%T';
ALLOCATE CHANNEL c9 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt3/server/
db_%U_%T';
ALLOCATE CHANNEL c10 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt3/server/
db_%U_%T';
ALLOCATE CHANNEL c11 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt3/server/
db_%U_%T';
ALLOCATE CHANNEL c12 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt3/server/
db_%U_%T';
ALLOCATE CHANNEL c13 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt4/server/
db_%U_%T';
ALLOCATE CHANNEL c14 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt4/server/
db_%U_%T';
ALLOCATE CHANNEL c15 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt4/server/
db_%U_%T';
ALLOCATE CHANNEL c16 DEVICE TYPE DISK MAXOPENFILES 1  FORMAT '/mnt/backupmnt4/server/
db_%U_%T';
BACKUP SECTION SIZE 8G DATABASE mypdb;
RELEASE CHANNEL c1;
RELEASE CHANNEL c2;
RELEASE CHANNEL c3;
RELEASE CHANNEL c4;
RELEASE CHANNEL c6
RELEASE CHANNEL c6;
RELEASE CHANNEL c7;
RELEASE CHANNEL c8;
RELEASE CHANNEL c9;
RELEASE CHANNEL c10;
```

**b.** Recovery script:

```
RUN {
ALLOCATE CHANNEL c1 type DISK;
ALLOCATE CHANNEL c2 type DISK;
ALLOCATE CHANNEL c3 type DISK;
ALLOCATE CHANNEL c4 type DISK;
ALLOCATE CHANNEL c5 type DISK;
ALLOCATE CHANNEL c6 type DISK;
ALLOCATE CHANNEL c7 type DISK;
ALLOCATE CHANNEL c8 type DISK;
ALLOCATE CHANNEL c9 type DISK;
ALLOCATE CHANNEL c10 type DISK;
ALLOCATE CHANNEL c11 type DISK;
ALLOCATE CHANNEL c12 type DISK;
ALLOCATE CHANNEL c13 type DISK;
ALLOCATE CHANNEL c14 type DISK;
ALLOCATE CHANNEL c15 type DISK;
ALLOCATE CHANNEL c16 type DISK;
ALTER PLUGGABLE DATABASE mypdb CLOSE IMMEDIATE;
RECOVER PLUGGABLE DATABASE mypdb;
RECOVER PLUGGABLE DATABASE mypdb;
ALTER PLUGGABLE DATABASE mypdb OPEN;
RELEASE CHANNEL c1;
RELEASE CHANNEL c2;
RELEASE CHANNEL c3;
RELEASE CHANNEL c4;
RELEASE CHANNEL c5;
RELEASE CHANNEL c6;
RELEASE CHANNEL c7;
RELEASE CHANNEL c8;
RELEASE CHANNEL c9;
RELEASE CHANNEL c10;
RELEASE CHANNEL c11;
RELEASE CHANNEL c12;
RELEASE CHANNEL c13;
RELEASE CHANNEL c14;
RELEASE CHANNEL c15;
RELEASE CHANNEL c16;
}
```

**03.** Backup and recovery scripts for object storage

**a.** Backup script

```
RUN {
ALLOCATE CHANNEL c1 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c2 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c3 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c4 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c5 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c6 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c7 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c8 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c9 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c10 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c11 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c12 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c13 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c14 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c15 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
```

**b.** Recovery script

```
RUN {
ALLOCATE CHANNEL c1 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c2 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c3 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c4 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c5 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c6 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c7 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c8 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c9 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c10 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c11 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c12 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c13 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c14 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
db_home/lib/libosbws.so,SBT_PARMS=(OSB_WS_PFILE=/u01/app/oracle/product/21.3.0/db_home/dbs/
osbwsORCL.ora, OSB_WS_BUCKET=orabucket)';
ALLOCATE CHANNEL c15 DEVICE TYPE sbt parms='SBT_LIBRARY=/u01/app/oracle/product/21.3.0/
```

**04.** Here are the backup and recovery scripts used during the RAC testing.

**a.** Backup script

```
RUN {
ALLOCATE CHANNEL c1 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst1' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt1/server/db_%U_%T';
ALLOCATE CHANNEL c2 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst2' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt1/server/db_%U_%T';
ALLOCATE CHANNEL c3 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst3' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt1/server/db_%U_%T';
ALLOCATE CHANNEL c4 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst4' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt1/server/db_%U_%T';
ALLOCATE CHANNEL c5 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst5' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt2/server/db_%U_%T';
ALLOCATE CHANNEL c6 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst6' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt2/server/db_%U_%T';
ALLOCATE CHANNEL c7 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst1' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt2/server/db_%U_%T';
ALLOCATE CHANNEL c8 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst2' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt2/server/db_%U_%T';
ALLOCATE CHANNEL c9 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst3' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt3/server/db_%U_%T';
ALLOCATE CHANNEL c10 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst4' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt3/server/db_%U_%T';
ALLOCATE CHANNEL c11 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst5' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt3/server/db_%U_%T';
ALLOCATE CHANNEL c12 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst6' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt3/server/db_%U_%T';
ALLOCATE CHANNEL c13 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst1' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt4/server/db_%U_%T';
ALLOCATE CHANNEL c14 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst2' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt4/server/db_%U_%T';
ALLOCATE CHANNEL c15 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst3' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt4/server/db_%U_%T';
ALLOCATE CHANNEL c16 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst4' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt4/server/db_%U_%T';
ALLOCATE CHANNEL c17 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst5' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt1/server/db_%U_%T';
ALLOCATE CHANNEL c18 DEVICE TYPE DISK CONNECT 'sys/SYSPWD@inst6' MAXOPENFILES 1  FORMAT '/
mnt/backupmnt2/server/db_%U_%T';
BACKUP SECTION SIZE 68G DATABASE mypdb;
RELEASE CHANNEL c1;
RELEASE CHANNEL c2;
RELEASE CHANNEL c3;
RELEASE CHANNEL c4;
RELEASE CHANNEL c6
RELEASE CHANNEL c6;
```

**b.** Recovery script

```
RUN {
ALLOCATE CHANNEL c1 DEVICE TYPE DISK;
ALLOCATE CHANNEL c2 DEVICE TYPE DISK;
ALLOCATE CHANNEL c3 DEVICE TYPE DISK;
ALLOCATE CHANNEL c4 DEVICE TYPE DISK;
ALLOCATE CHANNEL c5 DEVICE TYPE DISK;
ALLOCATE CHANNEL c6 DEVICE TYPE DISK;
ALLOCATE CHANNEL c7 DEVICE TYPE DISK;
ALLOCATE CHANNEL c8 DEVICE TYPE DISK;
ALLOCATE CHANNEL c9 DEVICE TYPE DISK;
ALLOCATE CHANNEL c10 DEVICE TYPE DISK;
ALLOCATE CHANNEL c11 DEVICE TYPE DISK;
ALLOCATE CHANNEL c12 DEVICE TYPE DISK;
ALLOCATE CHANNEL c13 DEVICE TYPE DISK;
ALLOCATE CHANNEL c14 DEVICE TYPE DISK;
ALLOCATE CHANNEL c15 DEVICE TYPE DISK;
ALLOCATE CHANNEL c16 DEVICE TYPE DISK;
ALLOCATE CHANNEL c17 DEVICE TYPE DISK;
ALLOCATE CHANNEL c18 DEVICE TYPE DISK;
ALTER PLUGGABLE DATABASE mypdb CLOSE IMMEDIATE;
RECOVER PLUGGABLE DATABASE mypdb;
RECOVER PLUGGABLE DATABASE mypdb;
ALTER PLUGGABLE DATABASE mypdb OPEN;
RELEASE CHANNEL c1;
RELEASE CHANNEL c2;
RELEASE CHANNEL c3;
RELEASE CHANNEL c4;
RELEASE CHANNEL c6
RELEASE CHANNEL c6;
RELEASE CHANNEL c7;
RELEASE CHANNEL c8;
RELEASE CHANNEL c9;
RELEASE CHANNEL c10;
RELEASE CHANNEL c11;
RELEASE CHANNEL c12;
RELEASE CHANNEL c13;
RELEASE CHANNEL c14;
RELEASE CHANNEL c15;
RELEASE CHANNEL c16;
RELEASE CHANNEL c17;
RELEASE CHANNEL c18;
}
```





PS2362-01 12/2022