Oracle data protection

Enterprise applications

NetApp
January 12, 2026

Table of Contents

Oracle data protection . . . 1
- Data protection with ONTAP . . . 1
  - Planning . . . 1
- RTO, RPO, and SLA planning . . . 1
  - Recovery time objective . . . 2
  - Recovery point objective . . . 2
  - Disaster recovery . . . 2
  - Retention Time . . . 4
- Database availability . . . 4
  - HA pairs . . . 4
  - Takeover and giveback . . . 4
  - Takeover time . . . 5
- Checksums and data integrity . . . 5
  - Network corruption: checksums . . . 6
  - Drive corruption: checksums . . . 6
  - Data corruption: lost writes . . . 6
  - Drive failures: RAID, RAID DP, and RAID-TEC . . . 7
  - Hardware failure protection: NVRAM . . . 7
  - Hardware failure protection: NVFAIL . . . 8
  - Site and shelf failure protection: SyncMirror and plexes . . . 8
  - Checksums . . . 10
- Backup and recovery basics . . . 10
  - Snapshot-based backups . . . 10
  - SnapRestore . . . 15
  - Online backups . . . 16
  - Storage Snapshot Optimized backups . . . 18
  - Database management and automation tools . . . 22

Oracle data protection

Data protection with ONTAP

NetApp knows the most mission-critical data is found in databases.

An enterprise cannot operate without access to its data, and sometimes, the data defines the business. This data must be protected; however, data protection is more than just ensuring a usable backup—it is about performing the backups quickly and reliably in addition to storing them safely.

The other side of data protection is data recovery. When data is inaccessible, the enterprise is affected and might be inoperative until data is restored. This process must be fast and reliable. Finally, most databases must be protected against disasters, which means maintaining a replica of the database. The replica must be sufficiently up to date. It must also be quick and simple to make the replica a fully operational database.

This documentation replaces previously published technical report *TR-4591: Oracle data protection: Backup, recovery, and replication.*

Planning

The right enterprise data protection architecture depends on the business requirements surrounding data retention, recoverability, and tolerance for disruption during various events.

For example, consider the number of applications, databases, and important datasets in scope. Building a backup strategy for a single dataset that ensures compliance with typical SLAs is fairly straightforward because there are not many objects to manage. As the number of datasets increases, monitoring becomes more complicated and administrators might be forced to spend an increasing amount of time addressing backup failures. As an environment reaches cloud and service provider scales, a wholly different approach is needed.

Dataset size also affects strategy. For example, many options exist for backup and recovery with a 100GB database because the data set is so small. Simply copying the data from backup media with traditional tools typically delivers a sufficient RTO for recovery. A 100TB database normally needs a completely different strategy unless the RTO allows for a multiday outage, in which case a traditional copy-based backup and recovery procedure might be acceptable.

Finally, there are factors outside the backup and recovery process itself. For example, are there databases supporting critical production activities, making recovery a rare event that is only performed by skilled DBAs? Alternatively, are the databases part of a large development environment in which recovery is a frequent occurrence and managed by a generalist IT team?

RTO, RPO, and SLA planning

ONTAP allows you to easily tailor an Oracle database data protection strategy to your business requirements.

These requirements include factors such as the speed of recovery, the maximum permissible data loss, and backup retention needs. The data protection plan must also take into consideration various regulatory requirements for data retention and restoration. Finally, different data recovery scenarios must be considered, ranging from the typical and foreseeable recovery resulting from user or application errors up to disaster recovery scenarios that include the complete loss of a site.

Small changes in data protection and recovery policies can have a significant effect on the overall architecture of storage, backup, and recovery. It is critical to define and document standards before starting design work to avoid complicating a data protection architecture. Unnecessary features or levels of protection lead to unnecessary costs and management overhead, and an initially overlooked requirement can lead a project in the wrong direction or require last-minute design changes.

Recovery time objective

The recovery time objective (RTO) defines the maximum time allowed for the recovery of a service. For example, a human resources database might have an RTO of 24 hours because, although it would be very inconvenient to lose access to this data during the workday, the business can still operate. In contrast, a database supporting the general ledger of a bank would have an RTO measured in minutes or even seconds. An RTO of zero is not possible, because there must be a way to differentiate between an actual service outage and a routine event such as a lost network packet. However, a near-zero RTO is a typical requirement.

Recovery point objective

The recovery point objective (RPO) defines the maximum tolerable data loss. In many cases, the RPO is solely determined by the frequency of snapshots or snapmirror updates.

In some cases, the RPO can be made more aggressive be selectively protecting certain data more frequently. In a database context, the RPO is usually a question of how much log data can be lost in a specific situation. In a typical recovery scenario in which a database is damaged due to a product bug or user error, the RPO should be zero, meaning there should be no data loss. The recovery procedure involves restoring an earlier copy of the database files and then replaying the log files to bring the database state up to the desired point in time. The log files required for this operation should already be in place in the original location.

In unusual scenarios, log data might be lost. For example, an accidental or malicious `rm -rf *` of database files could result in the deletion of all data. The only option would be to restore from backup, including log files, and some data would inevitably be lost. The only option to improve the RPO in a traditional backup environment would be to perform repeated backups of the log data. This has limitations, however, because of the constant data movement and the difficulty maintaining a backup system as a constantly running service. One of the benefits of advanced storage systems is the ability to protect data from accidental or malicious damage to files and thus deliver a better RPO without data movement.

Disaster recovery

Disaster recovery includes the IT architecture, policies, and procedures required to recover a service in the event of a physical disaster. This can include floods, fire, or person acting with malicious or negligent intent.

Disaster recovery is more than just a set of recovery procedures. It is the complete process of identifying the various risks, defining the data recovery and service continuity requirements, and delivering the right architecture with associated procedures.

When establishing data protection requirements, it is critical to differentiate between typical RPO and RTO requirements and the RPO and RTO requirements needed for disaster recovery. Some applications environments require an RPO of zero and a near-zero RTO for data loss situations ranging from a relatively normal user error right up to a fire that destroys a data center. However, there are cost and administrative consequences for these high levels of protection.

In general, nondisaster data recovery requirements should be strict for two reasons. First, application bugs and user errors that damage data are foreseeable to the point they are almost inevitable. Second, it is not difficult to design a backup strategy that can deliver an RPO of zero and a low RTO as long as the storage system is not destroyed. There is no reason not to address a significant risk that is easily remedied, which is why the RPO

and RTO targets for local recovery should be aggressive.

Disaster recovery RTO and RPO requirements vary more widely based on the likelihood of a disaster and the consequences of the associated data loss or disruption to a business. RPO and RTO requirements should be based on the actual business needs and not on general principles. They must account for multiple logical and physical disaster scenarios.

Logical disasters

Logical disasters include data corruption caused by users, application or OS bugs, and software malfunctions. Logical disasters can also include malicious attacks by outside parties with viruses or worms or by exploiting application vulnerabilities. In these cases, the physical infrastructure is undamaged but the underlying data is no longer valid.

An increasingly common type of logical disaster is known as ransomware, in which an attack vector is used to encrypt data. Encryption does not damage the data, but it makes it unavailable until payment is made to a third party. An increasing number of enterprises are being specifically targeted with ransomware hacks. For this threat, NetApp offers tamperproof snapshots where not even the storage administrator can change protected data before the configured expiry date.

Physical disasters

Physical disasters include the failure of components of an infrastructure that exceeds its redundancy capabilities and result in a loss of data or an extended loss of service. For example, RAID protection provides disk-drive redundancy, and the use of HBAs provides FC port and FC cable redundancy. Hardware failures of such components is foreseeable and does not impact availability.

In an enterprise environment, it is generally possible to protect the infrastructure of an entire site with redundant components to the point where the only foreseeable physical disaster scenario is complete loss of the site. Disaster recovery planning then depends on site-to-site replication.

Synchronous and asynchronous data protection

In an ideal world, all data would be synchronously replicated across geographically dispersed sites. Such replication is not always feasible or even possible for several reasons:

- Synchronous replication unavoidably increases write latency because all changes must be replicated to both locations before the application/database can proceed with processing. The resulting performance effect is sometimes unacceptable, ruling out the use of synchronous mirroring.
- The increased adoption of 100% SSD storage means that additional write latency is more likely to be noticed because performance expectations include hundreds of thousands of IOPS and submillisecond latency. Gaining the full benefits of using 100% SSDs can require revisiting the disaster recovery strategy.
- Datasets continue to grow in terms of bytes, creating challenges with ensuring sufficient bandwidth to sustain synchronous replication.
- Datasets also grow in terms of complexity, creating challenges with the management of large-scale synchronous replication.
- Cloud-based strategies frequently involve greater replication distances and latency, further precluding the use of synchronous mirroring.

NetApp offers solutions that include both synchronous replication for the most exacting data recovery demands and asynchronous solutions that allow for better performance and flexibility. In addition, NetApp technology integrates seamlessly with many third-party replication solutions, such as Oracle DataGuard

Retention Time

The final aspect of a data protection strategy is the data retention time, which can vary dramatically.

- A typical requirement is 14 days of nightly backups on the primary site and 90 days of backups stored on a secondary site.
- Many customers create standalone quarterly archives stored on different media.
- A constantly updated database might have no need for historical data, and backups need only be retained for a few days.
- Regulatory requirements might require recoverability to the point of any arbitrary transaction in a 365-day window.

Database availability

ONTAP is designed to deliver maximum Oracle database availability. A complete description of ONTAP high availability features is beyond the scope of this document. However, as with data protection, a basic understanding of this functionality is important when designing a database infrastructure.

HA pairs

The basic unit of high availability is the HA pair. Each pair contains redundant links to support replication of data to NVRAM. NVRAM is not a write cache. The RAM inside the controller serves as the write cache. The purpose of NVRAM is to temporarily journal data as a safeguard against unexpected system failure. In this respect, it is similar to a database redo log.

Both NVRAM and a database redo log are used to store data quickly, allowing changes to data to be committed as quickly as possible. The update to the persistent data on drives (or datafiles) does not take place until later during a process called a checkpoint on both ONTAP and most databases platforms. Neither NVRAM data nor database redo logs are read during normal operations.

If a controller fails abruptly, there are likely to be pending changes stored in NVRAM that have not yet been written to the drives. The partner controller detects the failure, take control of the drives, and applies the required changes that have been stored in NVRAM.

Takeover and giveback

Takeover and giveback refers to the process of transferring responsibility for storage resources between nodes in an HA pair. There are two aspects to takeover and giveback:

- Management of the network connectivity that allows access to the drives
- Management of the drives themselves

Network interfaces supporting CIFS and NFS traffic are configured with both a home and failover location. A takeover includes moving the network interfaces to their temporary home on a physical interface located on the same subnet(s) as the original location. A giveback includes moving the network interfaces back to their original locations. The exact behavior can be tuned as required.

Network interfaces supporting SAN block protocols such as iSCSI and FC are not relocated during takeover and giveback. Instead, LUNs should be provisioned with paths that includes a complete HA pair which results in a primary path and a secondary path.

Additional paths to additional controllers can also be configured to support relocating data between nodes in a larger cluster, but this is not part of the HA process.

The second aspect of takeover and giveback is the transfer of disk ownership. The exact process depends on multiple factors including the reason for the takeover/giveback and the command line options issued. The goal is to perform the operation as efficiently as possible. Although the overall process might appear to require several minutes, the actual moment in which ownership of the drive is transitioned from node to node can generally be measured in seconds.

Takeover time

Host I/O experiences a short pause in I/O during takeover and giveback operations, but there should not be application disruption in a correctly configured environment. The actual transition process in which I/O is delayed is generally measured in seconds, but the host might require additional time to recognize the change in data paths and resubmit I/O operations.

The nature of the disruption depends on the protocol:

- A network interface supporting NFS and CIFS traffic issues an Address Resolution Protocol (ARP) request to the network after the transition to a new physical location. This causes the network switches to update their media access control (MAC) address tables and resume processing I/O. Disruption in the case of planned takeover and giveback is usually measured in seconds and in many cases is not detectable. Some networks might be slower to fully recognize the change in network path, and some OSs might queue up a lot of I/O in a very short time that must be retried. This can extend the time required to resume I/O.
- A network interface supporting SAN protocols does not transition to a new location. A host OS must change the path or paths in use. The pause in I/O observed by the host depends on multiple factors. From a storage system point of view, the period where I/O cannot be served is just a few seconds. However, different host OSs might require additional time to allow an I/O to time out before retry. Newer OSs are better able to recognize a path change much more quickly, but older OSs typically require up to 30 seconds to recognize a change.

The expected takeover times during which the storage system cannot serve data to an application environment are shown in the table below. There should not be any errors in any application environment, the takeover should instead appear as a short pause in IO processing.

	NFS	AFF	ASA
Planned takeover	15 sec	6-10 sec	2-3 sec
Unplanned takeover	30 sec	6-10 sec	2-3 sec

Checksums and data integrity

ONTAP and its supported protocols include multiple features that protect Oracle database integrity, including both data at rest and data being transmitted over the network network.

Logical data protection within ONTAP consists of three key requirements:

- Data must be protected against data corruption.
- Data must be protected against drive failure.
- Changes to data must be protected against loss.

These three needs are discussed in the following sections.

Network corruption: checksums

The most basic level of data protection is the checksum, which is a special error-detecting code stored alongside the data. Corruption of data during network transmission is detected with the use of a checksum and, in some instances, multiple checksums.

For example, an FC frame includes a form of checksum called a cyclic redundancy check (CRC) to make sure that the payload is not corrupted in transit. The transmitter sends both the data and the CRC of the data. The receiver of an FC frame recalculates the CRC of the received data to make sure that it matches the transmitted CRC. If the newly computed CRC does not match the CRC attached to the frame, the data is corrupt and the FC frame is discarded or rejected. An iSCSI I/O operation includes checksums at the TCP/IP and Ethernet layers, and, for extra protection, it can also include optional CRC protection at the SCSI layer. Any bit corruption on the wire is detected by the TCP layer or IP layer, which results in retransmission of the packet. As with FC, errors in the SCSI CRC result in a discard or rejection of the operation.

Drive corruption: checksums

Checksums are also used to verify the integrity of data stored on drives. Data blocks written to drives are stored with a checksum function that yields an unpredictable number that is tied to the original data. When data is read from the drive, the checksum is recomputed and compared to the stored checksum. If it does not match, then the data has become corrupt and must be recovered by the RAID layer.

Data corruption: lost writes

One of the most difficult types of corruption to detect is a lost or a misplaced write. When a write is acknowledged, it must be written to the media in the correct location. In-place data corruption is relatively easy to detect by using a simple checksum stored with the data. However, if the write is simply lost, then the prior version of data might still exist and the checksum would be correct. If the write is placed at the wrong physical location, the associated checksum would once again be valid for the stored data, even though the write has destroyed other data.

The solution to this challenge is as follows:

- A write operation must include metadata that indicates the location where the write is expected to be found.
- A write operation must include some sort of version identifier.

When ONTAP writes a block, it includes data on where the block belongs. If a subsequent read identifies a block, but the metadata indicates that it belongs at location 123 when it was found at location 456, then the write has been misplaced.

Detecting a wholly lost write is more difficult. The explanation is very complicated, but essentially ONTAP is storing metadata in a way that a write operation results in updates to two different locations on the drives. If a write is lost, a subsequent read of the data and associated metadata shows two different version identities. This indicates that the write was not completed by the drive.

Lost and misplaced write corruption is exceedingly rare, but, as drives continue to grow and datasets push into exabyte scale, the risk increases. Lost write detection should be included in any storage system supporting database workloads.

Drive failures: RAID, RAID DP, and RAID-TEC

If a block of data on a drive is discovered to be corrupt, or the entire drive fails and is wholly unavailable, the data must be reconstituted. This is done in ONTAP by using parity drives. Data is striped across multiple data drives, and then parity data is generated. This is stored separately from the original data.

ONTAP originally used RAID 4, which uses a single parity drive for each group of data drives. The result was that any one drive in the group could fail without resulting in data loss. If the parity drive failed, no data was damaged and a new parity drive could be constructed. If a single data drive failed, the remaining drives could be used with the parity drive to regenerate the missing data.

When drives were small, the statistical chance of two drives failing simultaneously was negligible. As drive capacities have grown, so has the time required to reconstruct data after a drive failure. This has increased the window in which a second drive failure would result in data loss. In addition, the rebuild process creates a lot of additional I/O on the surviving drives. As drives age, the risk of the additional load leading to a second drive failure also increases. Finally, even if the risk of data loss did not increase with the continued use of RAID 4, the consequences of data loss would become more severe. The more data that would be lost in the event of a RAID-group failure, the longer it would take to recover the data, extending business disruption.

These issues led NetApp to develop the NetApp RAID DP technology, a variant of RAID 6. This solution includes two parity drives, meaning that any two drives in a RAID group can fail without creating data loss. Drives have continued to grow in size, which eventually led NetApp to develop the NetApp RAID-TEC technology, which introduces a third parity drive.

Some historical database best practices recommend the use of RAID-10, also known as striped mirroring. This offers less data protection than even RAID DP because there are multiple two-disk failure scenarios, whereas in RAID DP there are none.

There are also some historical database best practices that indicate RAID-10 is preferred to RAID-4/5/6 options due to performance concerns. These recommendations sometimes refer to a RAID penalty. Although these recommendations are generally correct, they are inapplicable to the implementations of RAID within ONTAP. The performance concern is related to parity regeneration. With traditional RAID implementations, processing the routine random writes performed by a database requires multiple disk reads to regenerate the parity data and complete the write. The penalty is defined as the additional read IOPS required to perform write operations.

ONTAP does not incur a RAID penalty because writes are staged in memory where parity is generated and then written to disk as a single RAID stripe. No reads are required to complete the write operation.

In summary, when compared to RAID 10, RAID DP and RAID-TEC deliver much more usable capacity, better protection against drive failure, and no performance sacrifice.

Hardware failure protection: NVRAM

Any storage array servicing a database workload must service write operations as quickly as possible. Furthermore, a write operation must be protected from loss from an unexpected event such as a power failure. This means any write operation must be safely stored in at least two locations.

AFF and FAS systems rely on NVRAM to meet these requirements. The write process works as follows:

1. The inbound write data is stored in RAM.
2. The changes that must be made to data on disk are journaled into NVRAM on both the local and partner node. NVRAM is not a write cache; rather it is a journal similar to a database redo log. Under normal conditions, it is not read. It is only used for recovery, such as after a power failure during I/O processing.

3. The write is then acknowledged to the host.

The write process at this stage is complete from the application point of view, and the data is protected against loss because it is stored in two different locations. Eventually, the changes are written to disk, but this process is out-of-band from the application point of view because it occurs after the write is acknowledged and therefore does not affect latency. This process is once again similar to database logging. A change to the database is recorded in the redo logs as quickly as possible, and the change is then acknowledged as committed. The updates to the datafiles occur much later and do not directly affect the speed of processing.

In the event of a controller failure, the partner controller takes ownership of the required disks and replays the logged data in NVRAM to recover any I/O operations that were in-flight when the failure occurred.

Hardware failure protection: NVFAIL

As discussed earlier, a write is not acknowledged until it has been logged into local NVRAM and NVRAM on at least one other controller. This approach makes sure that a hardware failure or power outage does not result in the loss of in-flight I/O. If the local NVRAM fails or the connectivity to HA partner fails, then this in-flight data would no longer be mirrored.

If the local NVRAM reports an error, the node shuts down. This shutdown results in failover to a HA partner controller. No data is lost because the controller experiencing the failure has not acknowledged the write operation.

ONTAP does not permit a failover when the data is out of sync unless the failover is forced. Forcing a change in conditions in this manner acknowledges that data might be left behind in the original controller and that data loss is acceptable.

Databases are especially vulnerable to corruption if a failover is forced because databases maintain large internal caches of data on disk. If a forced failover occurs, previously acknowledged changes are effectively discarded. The contents of the storage array effectively jump backward in time, and the state of the database cache no longer reflects the state of the data on disk.

To protect data from this situation, ONTAP allows volumes to be configured for special protection against NVRAM failure. When triggered, this protection mechanism results in a volume entering a state called NVFAIL. This state results in I/O errors that cause a an application shutdown so that they do not use stale data. Data should not be lost because any acknowledged write should be present on the storage array.

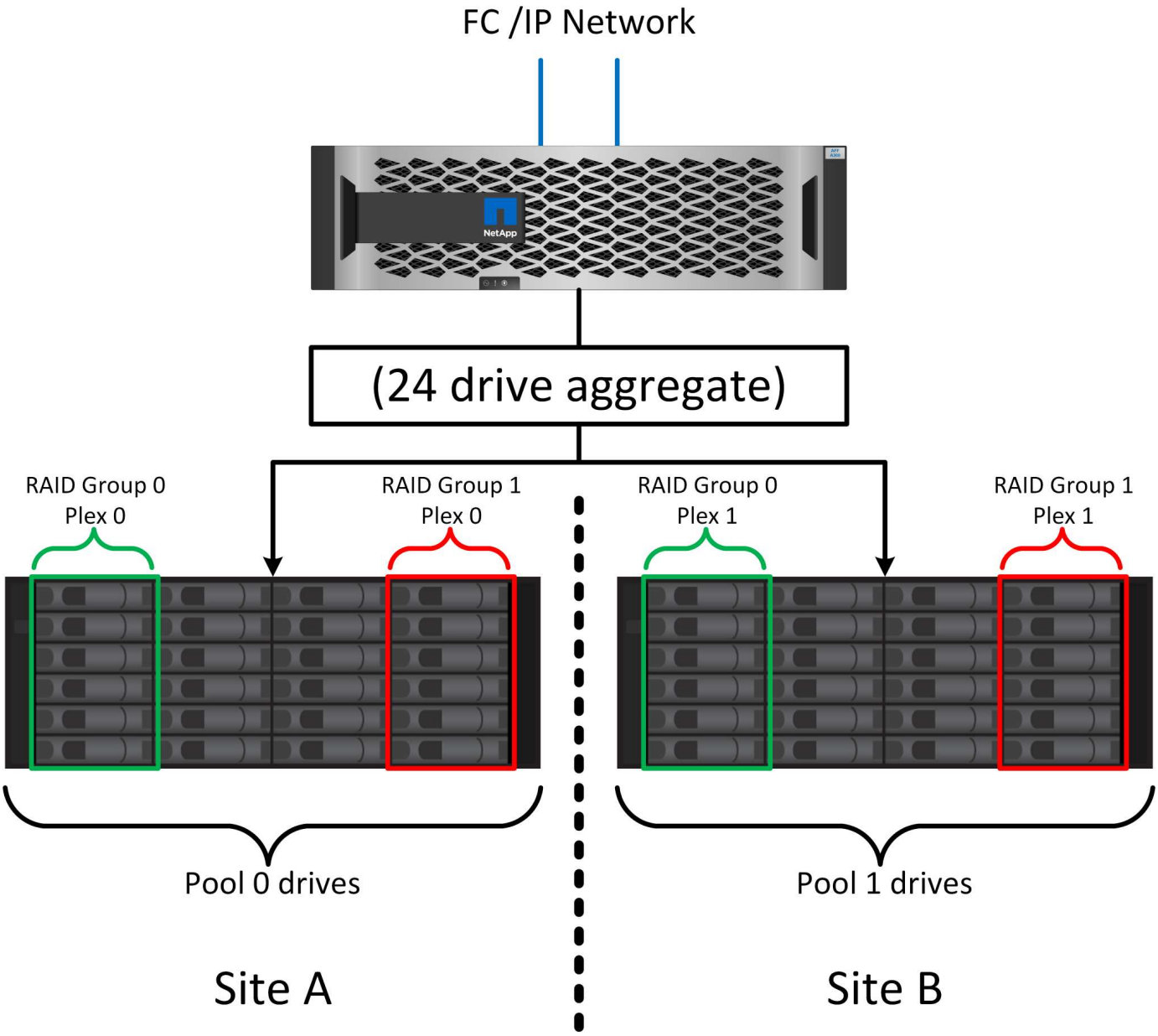
The usual next steps are for an administrator to fully shut down the hosts before manually placing the LUNs and volumes back online again. Although these steps can involve some work, this approach is the safest way to make sure of data integrity. Not all data requires this protection, which is why NVFAIL behavior can be configured on a volume-by-volume basis.

Site and shelf failure protection: SyncMirror and plexes

SyncMirror is a mirroring technology that enhances, but does not replace, RAID DP or RAID-TEC. It mirrors the contents of two independent RAID groups. The logical configuration is as follows:

- Drives are configured into two pools based on location. One pool is composed of all drives on site A, and the second pool is composed of all drives on site B.
- A common pool of storage, known as an aggregate, is then created based on mirrored sets of RAID groups. An equal number of drives is drawn from each site. For example, a 20-drive SyncMirror aggregate would be composed of 10 drives from site A and 10 drives from site B.
- Each set of drives on a given site is automatically configured as one or more fully redundant RAID-DP or RAID-TEC groups, independent of the use of mirroring. This provides continuous data protection, even

after the loss of a site.



The figure above illustrates a sample SyncMirror configuration. A 24-drive aggregate was created on the controller with 12 drives from a shelf allocated on Site A and 12 drives from a shelf allocated on Site B. The drives were grouped into two mirrored RAID groups. RAID Group 0 includes a 6-drive plex on Site A mirrored to a 6-drive plex on Site B. Likewise, RAID Group 1 includes a 6-drive plex on Site A mirrored to a 6-drive plex on Site B.

SyncMirror is normally used to provide remote mirroring with MetroCluster systems, with one copy of the data at each site. On occasion, it has been used to provide an extra level of redundancy in a single system. In particular, it provides shelf-level redundancy. A drive shelf already contains dual power supplies and controllers and is overall little more than sheet metal, but in some cases the extra protection might be warranted. For example, one NetApp customer has deployed SyncMirror for a mobile real-time analytics platform used during automotive testing. The system was separated into two physical racks supplied by independent power feeds from independent UPS systems.

Checksums

The topic of checksums is of particular interest to DBAs who are accustomed to using Oracle RMAN streaming backups migrates to snapshot-based backups. One feature of RMAN is that it performs integrity checks during backup operations. Although this feature has some value, its primary benefit is for a database that is not used on a modern storage array. When physical drives are used for an Oracle database, it is nearly certain that corruption eventually occurs as the drives age, a problem that is addressed by array-based checksums in true storage arrays.

With a real storage array, data integrity is protected by using checksums at multiple levels. If data is corrupted in an IP-based network, the Transmission Control Protocol (TCP) layer rejects the packet data and requests retransmission. The FC protocol includes checksums, as does encapsulated SCSI data. After it is on the array, ONTAP has RAID and checksum protection. Corruption can occur, but, as in most enterprise arrays, it is detected and corrected. Typically, an entire drive fails, prompting a RAID rebuild, and database integrity is unaffected. It is still possible for individual bytes on a drive to be damaged by cosmic radiation or failing flash cells. If this happens, the parity check would fail, the drive would be failed out and a RAID rebuild would begin. Once again, data integrity is unaffected. The final line of defense is the use of checksums. If, for example, a catastrophic firmware error on a drive corrupted data in a way that somehow was not detected by a RAID parity check, the checksum would not match and ONTAP would prevent the transfer of a corrupted block before the Oracle database could receive it.

The Oracle datafile and redo log architecture is also designed to deliver the highest possible level of data integrity, even under extreme circumstances. At the most basic level, Oracle blocks include checksum and basic logical checks with almost every I/O. If Oracle has not crashed or taken a tablespace offline, then the data is intact. The degree of data integrity checking is adjustable, and Oracle can also be configured to confirm writes. As a result, almost all crash and failure scenarios can be recovered, and in the extremely rare event of an unrecoverable situation, corruption is promptly detected.

Most NetApp customers using Oracle databases discontinue the use of RMAN and other backup products after migrating to snapshot-based backups. There are still options in which RMAN can be used to perform block-level recovery with SnapCenter. However, on a day-to-day basis, RMAN, NetBackup, and other products are only used occasionally to create monthly or quarterly archival copies.

Some customers choose to run `dbv` periodically to perform integrity checks on their existing databases. NetApp discourages this practice because it creates unnecessary I/O load. As discussed above, if the database was not previously experiencing problems, the chance of `dbv` detecting a problem is close to zero, and this utility creates a very high sequential I/O load on the network and storage system. Unless there is reason to believe corruption exists, such as exposure to a known Oracle bug, there is no reason to run `dbv`.

Backup and recovery basics

Snapshot-based backups

The foundation of Oracle database data protection on ONTAP is NetApp Snapshot technology.

The key values are as follows:

- **Simplicity.** A snapshot is a read-only copy of the contents of a container of data at a specific point in time.
- **Efficiency.** Snapshots require no space at the moment of creation. Space is only consumed when data is changed.
- **Manageability.** A backup strategy based on snapshots is easy to configure and manage because

snapshots are a native part of the storage OS. If the storage system is powered on, it is ready to create backups.

- **Scalability.** Up to 1024 backups of a single container of files and LUNs can be preserved. For complex datasets, multiple containers of data can be protected by a single, consistent set of snapshots.
- Performance is unaffected, whether a volume contains 1024 snapshots or none.

Although many storage vendors offer snapshot technology, the Snapshot technology within ONTAP is unique and offers significant benefits to enterprise application and database environments:

- Snapshot copies are part of the underlying Write-Anywhere File Layout (WAFL). They are not an add-on or external technology. This simplifies management because the storage system is the backup system.
- Snapshot copies do not affect performance, except for some edge cases such as when so much data is stored in snapshots that the underlying storage system fills up.
- The term "consistency group" is often used to refer to a grouping of storage objects that are managed as a consistent collection of data. A snapshot of a particular ONTAP volume constitutes consistency group backup.

ONTAP snapshots also scale better than competing technology. Customers can store 5, 50, or 500 snapshots without affecting performance. The maximum number of snapshots currently allowed in a volume is 1024. If additional snapshot retention is required, there are options to cascade the snapshots to additional volumes.

As a result, protecting a dataset hosted on ONTAP is simple and highly scalable. Backups do not require movement of data, therefore a backup strategy can be tailored to the needs of the business rather than the limitations of network transfer rates, large number of tape drives, or disk staging areas.

Is a snapshot a backup?

One commonly asked question about the use of snapshots as a data protection strategy is the fact that the "real" data and the snapshot data are located on the same drives. Loss of those drives would result in the loss of both the primary data and the backup.

This is a valid concern. Local snapshots are used for day-to-day backup and recovery needs, and in that respect the snapshot is a backup. Close to 99% of all recovery scenarios in NetApp environments rely on snapshots to meet even the most aggressive RTO requirements.

Local snapshots should, however, never be the only backup strategy, which is why NetApp offers technology such as SnapMirror and SnapVault replication to quickly and efficiently replicate snapshots to an independent set of drives. In a properly architected solution with snapshots plus snapshot replication, the use of tape can be minimized to perhaps a quarterly archive or eliminated entirely.

Snapshot-based backups

There are many options for using ONTAP Snapshot copies to protect your data, and snapshots are the basis for many other ONTAP features, including replication, disaster recovery, and cloning. A complete description of snapshot technology is beyond the scope of this document, but the following sections provide a general overview.

There are two primary approaches to creating a snapshot of a dataset:

- Crash-consistent backups
- Application-consistent backups

A crash-consistent backup of a dataset refers to the capture of the entire dataset structure at a single point in

time. If the dataset is stored in a single volume, then the process is simple; a Snapshot can be created at any time. If a dataset spans volumes, a consistency group (CG) snapshot must be created. Several options exist for creating CG snapshots, including NetApp SnapCenter software, native ONTAP consistency group features, and user-maintained scripts.

Crash-consistent backups are primarily used when point-of-the-backup recovery is sufficient. When more granular recover is required, application-consistent backups are usually required.

The word "consistent" in "application-consistent" is often a misnomer. For example, placing an Oracle database in backup mode is referred to as an application-consistent backup, but the data is not made consistent or quiesced in any way. The data continue to change throughout the backup. In contrast, most MySQL and Microsoft SQL Server backups do indeed quiesce the data before executing the backup. VMware may or may not make certain files consistent.

Consistency groups

The term "consistency group" refers to the ability of a storage array to manage multiple storage resources as a single image. For example, a database might consist of 10 LUNs. The array must be able to back up, restore, and replicate those 10 LUNs in a consistent manner. Restoration is not possible if the images of the LUNs were not consistent at the point of backup. Replicating those 10 LUNs requires that all the replicas are perfectly synchronized with each other.

The term "consistency group" is not often used when discussing ONTAP because consistency has always been a basic function of the volume and aggregate architecture within ONTAP. Many other storage arrays manage LUNs or file systems as individual units. They could then be optionally configured as a "consistency group" for purposes of data protection, but this is an extra step in the configuration.

ONTAP has always been able to capture consistent local and replicated images of data. Although the various volumes on an ONTAP system are not usually formally described as a consistency group, that is what they are. A snapshot of that volume is a consistency group image, restoration for that snapshot is a consistency group restoration, and both SnapMirror and SnapVault offer consistency group replication.

Consistency group snapshots

Consistency group snapshots (cg-snapshots) are an extension of the basic ONTAP Snapshot technology. A standard snapshot operation creates a consistent image of all data within a single volume, but sometimes it is necessary to create a consistent set of snapshots across multiple volumes and even across multiple storage systems. The result is a set of snapshots that can be used in the same way as a snapshot of just one individual volume. They can be used for local data recovery, replicated for disaster recovery purposes, or cloned as a single consistent unit.

The largest known use of cg-snapshots is for a database environment of approximately 1PB in size spanning 12 controllers. The cg-snapshots created on this system have been used for backup, recovery and cloning.

Most of the time, when a data set spans volumes and write order must be preserved, a cg-snapshot is automatically used by the chosen management software. There is no need to understand the technical details of cg-snapshots in such cases. However, there are situations in which complicated data protection requirements require detailed control over the data protection and replication process. Automation workflows or the use of custom scripts to call the cg-snapshot APIs are some of options. Understanding the best option and the role of cg-snapshot requires a more detailed explanation of the technology.

Creation of a set of cg-snapshots is a two-step process:

1. Establish write fencing on all target volumes.

2. Create snapshots of those volumes while in the fenced state.

Write fencing is established serially. This means that as the fencing process is set up across multiple volumes, write I/O is frozen on the first volume in the sequence as it continues to be committed to volumes that appear later. This might initially appear to violate the requirement for write order to be preserved, but that only applies to I/O that is issued asynchronously on the host and does not depend on any other writes.

For example, a database might issue a lot of asynchronous datafile updates and allow the OS to reorder the I/O and complete them according to its own scheduler configuration. The order of this type of I/O cannot be guaranteed because the application and operating system have already released the requirement to preserve write order.

As a counter example, most database logging activity is synchronous. The database does not proceed with further log writes until the I/O is acknowledged, and the order of those writes must be preserved. If a log I/O arrives on a fenced volume, it is not acknowledged and the application blocks on further writes. Likewise, file system metadata I/O is usually synchronous. For example, a file deletion operation must not be lost. If an operating system with an xfs file system deleted a file and the I/O that updated the xfs file system metadata to remove the reference to that file landed on a fenced volume, then the file system activity would pause. This guarantees the integrity of the file system during cg-snapshot operations.

After write fencing is set up across the target volumes, they are ready for snapshot creation. The snapshots need not be created at precisely the same time because the state of the volumes is frozen from a dependent write point of view. To guard against a flaw in the application creating the cg-snapshots, the initial write fencing includes a configurable timeout in which ONTAP automatically releases the fencing and resumes write processing after a defined number of seconds. If all the snapshots are created before the timeout period lapses, then the resulting set of snapshots are a valid consistency group.

Dependent write order

From a technical point of view, the key to a consistency group is preserving write order and, specifically, dependent write order. For example, a database writing to 10 LUNs writes simultaneously to all of them. Many writes are issued asynchronously, meaning that the order in which they are completed is unimportant and the actual order they are completed varies based on operating system and network behavior.

Some write operations must be present on disk before the database can proceed with additional writes. These critical write operations are called dependent writes. Subsequent write I/O depends on the presence of these writes on disk. Any snapshot, recovery, or replication of these 10 LUNs must make sure that dependent write order is guaranteed. File system updates are another example of write-order dependent writes. The order in which file system changes are made must be preserved or the entire file system could become corrupt.

Strategies

There are two primary approaches to snapshot-based backups:

- Crash-consistent backups
- Snapshot-protected hot backups

A crash-consistent backup of a database refers to the capture of the entire database structure, including datafiles, redo logs, and control files, at a single point in time. If the database is stored in a single volume, then the process is simple; a Snapshot can be created at any time. If a database spans volumes, a consistency group (CG) snapshot must be created. Several options exist for creating CG snapshots, including NetApp SnapCenter software, native ONTAP consistency group features, and user-maintained scripts.

Crash-consistent Snapshot backups are primarily used when point-of-the-backup recovery is sufficient. Archive logs can be applied under some circumstances, but when more granular point-in-time recovery is required, a

online backup is preferable.

The basic procedure for a snapshot-based online backup is as follows:

1. Place the database in `backup` mode.
2. Create a snapshot of all volumes hosting datafiles.
3. Exit `backup` mode.
4. Run the command `alter system archive log current` to force log archiving.
5. Create snapshots of all volumes hosting the archive logs.

This procedure yields a set of snapshots containing datafiles in backup mode and the critical archive logs generated while in backup mode. These are the two requirements for recovering a database. Files such as control files should also be protected for convenience, but the only absolute requirement is protection for datafiles and archive logs.

Although different customers might have very different strategies, almost all of these strategies are ultimately based on the the same principles outlined below.

Snapshot-based recovery

When designing volume layouts for Oracle databases, the first decision is whether to use volume-based NetApp SnapRestore (VBSR) technology.

Volume-based SnapRestore allows a volume to be almost instantly reverted to an earlier point in time. Because all of the data on the volume is reverted, VBSR might not be appropriate for all use cases. For example, if an entire database, including datafiles, redo logs, and archive logs, is stored on a single volume and this volume is restored with VBSR, then data is lost because the newer archive log and redo data are discarded.

VBSR is not required for restore. Many databases can be restored by using file-based single-file SnapRestore (SFSR) or by simply copying files from the snapshot back into the active file system.

VBSR is preferred when a database is very large or when it must be recovered as quickly as possible, and the use of VBSR requires isolation of the datafiles. In an NFS environment, the datafiles of a given database must be stored in dedicated volumes that are uncontaminated by any other type of file. In a SAN environment, datafiles must be stored in dedicated LUNs on dedicated volumes. If a volume manager is used (including Oracle Automatic Storage Management [ASM]), the diskgroup must also be dedicated to datafiles.

Isolating datafiles in this manner allows them to be reverted to an earlier state without damaging other file systems.

Snapshot reserve

For each volume with Oracle data in a SAN environment, the `percent-snapshot-space` should be set to zero because reserving space for a snapshot in a LUN environment is not useful. If the fractional reserve is set to 100, a snapshot of a volume with LUNs requires enough free space in the volume, excluding the snapshot reserve, to absorb 100% turnover of all of the data. If the fractional reserve is set to a lower value, then a correspondingly smaller amount of free space is required, but it always excludes the snapshot reserve. This means that the snapshot reserve space in a LUN environment is wasted.

In an NFS environment, there are two options:

- Set the `percent-snapshot-space` based on expected snapshot space consumption.

- Set the `percent-snapshot-space` to zero and manage active and snapshot space consumption collectively.

With the first option, `percent-snapshot-space` is set to a nonzero value, typically around 20%. This space is then hidden from the user. This value does not, however, create a limit on utilization. If a database with a 20% reservation experiences 30% turnover, the snapshot space can grow beyond the bounds of the 20% reserve and occupy unreserved space.

The main benefit of setting a reserve to a value such as 20% is to verify that some space is always available for snapshots. For example, a 1TB volume with a 20% reserve would only permit a database administrator (DBA) to store 800GB of data. This configuration guarantees at least 200GB of space for snapshot consumption.

When `percent-snapshot-space` is set to zero, all space in the volume is available to the end user, which delivers better visibility. A DBA must understand that, if he or she sees a 1TB volume that leverages snapshots, this 1TB of space is shared between active data and Snapshot turnover.

There is no clear preference between option one and option two among end users.

ONTAP and third-party snapshots

Oracle Doc ID 604683.1 explains the requirements for third-party snapshot support and the multiple options available for backup and restore operations.

The third-party vendor must guarantee that the company's snapshots conform to the following requirements:

- Snapshots must integrate with Oracle's recommended restore and recovery operations.
- Snapshots must be database crash consistent at the point of the snapshot.
- Write ordering is preserved for each file within a snapshot.

ONTAP and NetApp Oracle management products comply with these requirements.

SnapRestore

Rapid data restoration in ONTAP from a snapshot is delivered by NetApp SnapRestore technology.

When a critical dataset is unavailable, critical business operations are down. Tapes can break, and even restores from disk-based backups can be slow to transfer across the network. SnapRestore avoids these problems by delivering near instantaneous restoration of datasets. Even petabyte-scale databases can be completely restored with just a few minutes of effort.

There are two forms of SnapRestore - file/LUN-based and volume-based.

- Individual files or LUNs can be restored in seconds, whether it is a 2TB LUN or a 4KB file.
- The container of files or LUNs can be restored in seconds, whether it is 10GB or 100TB of data.

A "container of files or LUNs" would typically refer to a FlexVol volume. For example, you may have 10 LUNs that make up a LVM diskgroup in a single volume, or a volume might store the NFS home directories of 1000 users. Rather than executing a restore operation for each individual file or LUN, you can restore the entire volume as a single operation. This process also work with scale-out containers that include multiple volumes, such as a FlexGroup or an ONTAP Consistency Group.

The reason SnapRestore works so quickly and efficiently is due to the nature of a snapshot, which is essentially a parallel read-only view of the contents of a volume at a specific point in time. The active blocks are the real blocks that can be changed, while the snapshot is a read-only view into the state of the blocks that constitute the files and LUNs at the time the snapshot was created.

ONTAP only permits read-only access to snapshot data, but the data can be reactivated with SnapRestore. The snapshot is reenabled as a read-write view of the data, returning the data to its prior state. SnapRestore can operate at the volume or the file level. The technology is essentially the same with a few minor differences in behavior.

Volume SnapRestore

Volume-based SnapRestore returns the entire volume of data to an earlier state. This operation does not require data movement, meaning that the restore process is essentially instantaneous, although the API or CLI operation might take a few seconds to be processed. Restoring 1GB of data is no more complicated or time-consuming than restoring 1PB of data. This capability is the primary reason many enterprise customers migrate to ONTAP storage systems. It delivers an RTO measured in seconds for even the largest datasets.

One drawback to volume-based SnapRestore is caused by the fact that changes within a volume are cumulative over time. Therefore, each snapshot and the active file data are dependent on the changes leading up to that point. Reverting a volume to an earlier state means discarding all the subsequent changes that had been made to the data. What is less obvious, however, is that this includes subsequently created snapshots. This is not always desirable.

For example, a data retention SLA might specify 30 days of nightly backups. Restoring a dataset to a snapshot created five days ago with volume SnapRestore would discard all the snapshots created on the previous five days, violating the SLA.

There are a number of options available to address this limitation:

1. Data can be copied from a prior snapshot, as opposed to performing a SnapRestore of the entire volume. This method works best with smaller datasets.
2. A snapshot can be cloned rather than restored. The limitation to this approach is that the source snapshot is a dependency of the clone. Therefore, it cannot be deleted unless the clone is also deleted or is split into an independent volume.
3. Use of file-based SnapRestore.

File SnapRestore

File-based SnapRestore is a more granular snapshot-based restoration process. Rather than reverting the state of an entire volume, the state of an individual file or LUN is reverted. No snapshots need to be deleted, nor does this operation create any dependency on a prior snapshot. The file or LUN becomes immediately available in the active volume.

No data movement is required during a SnapRestore restore of a file or LUN. However, some internal metadata updates are required to reflect the fact that the underlying blocks in a file or LUN now exist in both a snapshot and the active volume. There should be no effect on performance, but this process blocks the creation of snapshots until it is complete. The processing rate is approximately 5GBps (18TB/hour) based on the total size of the files restored.

Online backups

Two sets of data are required to protect and recover an Oracle database in backup mode.

Note that this is not the only Oracle backup option, but it is the most common.

- A snapshot of the datafiles in backup mode
- The archive logs created while the datafiles were in backup mode

If complete recovery including all committed transactions is required, a third item is required:

- A set of current redo logs

There are a number of ways to drive recovery of an online backup. Many customers restore snapshots by using the ONTAP CLI and then using Oracle RMAN or sqlplus to complete the recovery. This is especially common with large production environments in which the probability and frequency of database restores is extremely low and any restore procedure is handled by a skilled DBA. For complete automation, solutions such as NetApp SnapCenter include an Oracle plug-in with both command- line and graphical interfaces.

Some large-scale customers have taken a simpler approach by configuring basic scripting on the hosts to place the databases in backup mode at a specific time in preparation for a scheduled snapshot. For example, schedule the command `alter database begin backup` at 23:58, `alter database end backup` at 00:02, and then schedule snapshots directly on the storage system at midnight. The result is a simple, highly scalable backup strategy that requires no external software or licenses.

Data layout

The simplest layout is to isolate datafiles into one or more dedicated volumes. They must be uncontaminated by any other file type. This is to make sure that the datafile volumes can be rapidly restored through a SnapRestore operation without destroying an important redo log, controlfile, or archive log.

SAN has similar requirements for datafile isolation within dedicated volumes. With an operating system such as Microsoft Windows, a single volume might contain multiple datafile LUNs, each with an NTFS file system. With other operating systems, there is generally a logical volume manager. For example, with Oracle ASM, the simplest option would be to confine the LUNs of an ASM disk group to a single volume that can be backed up and restored as a unit. If additional volumes are required for performance or capacity management reasons, creating an additional disk group on the new volume results in simpler management.

If these guidelines are followed, snapshots can be scheduled directly on the storage system with no requirement for performing a consistency group snapshot. The reason is that Oracle backups do not require datafiles to be backed up at the same time. The online backup procedure was designed to allow datafiles to continue to be updated as they are slowly streamed to tape over the course of hours.

A complication arises in situations such as the use of an ASM disk group that is distributed across volumes. In these cases, a cg-snapshot must be performed to make sure that the ASM metadata is consistent across all constituent volumes.

Caution: Verify that the ASM `spfile` and `passwd` files are not in the disk group hosting the datafiles. This interferes with the ability to selectively restore datafiles and only datafiles.

Local recovery procedure—NFS

This procedure can be driven manually or through an application such as SnapCenter. The basic procedure is as follows:

1. Shut down the database.
2. Recover the datafile volume(s) to the snapshot immediately prior to the desired restore point.

3. Replay archive logs to the desired point.
4. Replay current redo logs if complete recovery is desired.

This procedure assumes that the desired archive logs are still present in the active file system. If they are not, the archive logs must be restored or rman/sqlplus can be directed to the data in the snapshot directory.

In addition, for smaller databases, datafiles can be recovered by an end user directly from the `.snapshot` directory without assistance from automation tools or storage administrators to execute a `snaprestore` command.

Local recovery procedure—SAN

This procedure can be driven manually or through an application such as SnapCenter. The basic procedure is as follows:

1. Shut down the database.
2. Quiesce the disk group(s) hosting the datafiles. The procedure varies depending on the logical volume manager chosen. With ASM, the process requires dismounting the disk group. With Linux, the file systems must be dismounted, and the logical volumes and volume groups must be deactivated. The objective is to stop all updates on the target volume group to be restored.
3. Restore the datafile disk groups to the snapshot immediately prior to the desired restore point.
4. Reactivate the newly restored disk groups.
5. Replay archive logs to the desired point.
6. Replay all redo logs if complete recovery is desired.

This procedure assumes that the desired archive logs are still present in the active file system. If they are not, the archive logs must be restored by taking the archive log LUNs offline and performing a restore. This is also an example in which dividing up archive logs into dedicated volumes is useful. If the archive logs share a volume group with redo logs, then the redo logs must be copied elsewhere before restoration of the overall set of LUNs. This step prevents the loss of those final recorded transactions.

Storage Snapshot Optimized backups

Snapshot-based backup and recovery became even simpler back when Oracle 12c was released because there is no need to place a database in hot backup mode. The result is an ability to schedule snapshot-based backups directly on a storage system and still preserve the ability to perform complete or point-in-time recovery.

Although the hot backup recovery procedure is more familiar to DBAs, it has, for a long time, been possible to use snapshots that were not created while the database was in hot backup mode. Extra manual steps were required with Oracle 10g and 11g during recovery to make the database consistent. With Oracle 12c, `sqlplus` and `rman` contain the extra logic to replay archive logs on datafile backups that were not in hot backup mode.

As discussed previously, recovering a snapshot-based hot backup requires two sets of data:

- A snapshot of the datafiles created while in backup mode
- The archive logs generated while the datafiles were in hot backup mode

During recovery, the database reads metadata from the datafiles to select the required archive logs for recovery.

Storage snapshot-optimized recovery requires slightly different datasets to accomplish the same results:

- A snapshot of the datafiles, plus a method to identify the time the snapshot was created
- Archive logs from the time of the most recent datafile checkpoint through the exact time of the snapshot

During recovery, the database reads metadata from the datafiles to identify the earliest archive log required. Full or point-in-time recovery can be performed. When performing a point-in-time recovery, it is critical to know the time of the snapshot of the datafiles. The specified recovery point must be after the creation time of the snapshots. NetApp recommends adding at least a few minutes to the snapshot time to account for clock variation.

For complete details, see Oracle's documentation on the topic, "Recovery Using Storage Snapshot Optimization" available in various releases of the Oracle 12c documentation. Also, see Oracle Document ID Doc ID 604683.1 regarding Oracle third-party snapshot support.

Data layout

The simplest layout is to isolate the datafiles into one or more dedicated volumes. They must be uncontaminated by any other file type. This is to make sure that the datafile volumes can be rapidly restored with a SnapRestore operation without destroying an important redo log, controlfile, or archive log.

SAN has similar requirements for datafile isolation within dedicated volumes. With an operating system such as Microsoft Windows, a single volume might contain multiple datafile LUNs, each with an NTFS file system. With other operating systems, there is generally a logical volume manager as well. For example, with Oracle ASM, the simplest option would be to confine disk groups to a single volume that can be backed up and restored as a unit. If additional volumes are required for performance or capacity management reasons, creating an additional disk group on the new volume results in easier management.

If these guidelines are followed, snapshots can be scheduled directly on ONTAP with no requirement for performing a consistency group snapshot. The reason is that snapshot-optimized backups do not require that datafiles be backed up at the same time.

A complication arises in situations such as an ASM disk group that is distributed across volumes. In these cases, a cg-snapshot must be performed to make sure that the ASM metadata is consistent across all constituent volumes.

[Note]Verify that the ASM spfile and passwd files are not in the disk group hosting the datafiles. This interferes with the ability to selectively restore datafiles and only datafiles.

Local recovery procedure—NFS

This procedure can be driven manually or through an application such as SnapCenter. The basic procedure is as follows:

1. Shut down the database.
2. Recover the datafile volume(s) to the snapshot immediately prior to the desired restore point.
3. Replay archive logs to the desired point.

This procedure assumes that the desired archive logs are still present in the active file system. If they are not, the archive logs must be restored, or `rman` or `sqlplus` can be directed to the data in the `.snapshot` directory.

In addition, for smaller databases, datafiles can be recovered by an end user directly from the `.snapshot` directory without assistance from automation tools or a storage administrator to execute a SnapRestore

command.

Local recovery procedure—SAN

This procedure can be driven manually or through an application such as SnapCenter. The basic procedure is as follows:

1. Shut down the database.
2. Quiesce the disk group(s) hosting the datafiles. The procedure varies depending on the logical volume manager chosen. With ASM, the process requires dismounting the disk group. With Linux, the file systems must be dismounted, and the logical volumes and volume groups are deactivated. The objective is to stop all updates on the target volume group to be restored.
3. Restore the datafile disk groups to the snapshot immediately prior to the desired restore point.
4. Reactivate the newly restored disk groups.
5. Replay archive logs to the desired point.

This procedure assumes that the desired archive logs are still present in the active file system. If they are not, the archive logs must be restored by taking the archive log LUNs offline and performing a restore. This is also an example in which dividing up archive logs into dedicated volumes is useful. If the archive logs share a volume group with redo logs, the redo logs must be copied elsewhere before restoration of the overall set of LUNs to avoid losing the final recorded transactions.

Full recovery example

Assume the datafiles have been corrupted or destroyed and full recovery is required. The procedure to do so is as follows:

```
[oracle@host1 ~]$ sqlplus / as sysdba
Connected to an idle instance.
SQL> startup mount;
ORACLE instance started.
Total System Global Area 1610612736 bytes
Fixed Size                  2924928 bytes
Variable Size            1040191104 bytes
Database Buffers          553648128 bytes
Redo Buffers               13848576 bytes
Database mounted.
SQL> recover automatic;
Media recovery complete.
SQL> alter database open;
Database altered.
SQL>
```

Point-in-time recovery example

The entire recovery procedure is a single command: `recover automatic`.

If point-in-time recovery is required, the timestamp of the snapshot(s) must be known and can be identified as

follows:

```
Cluster01::> snapshot show -vserver vserver1 -volume NTAP_oradata -fields
create-time
vserver   volume        snapshot   create-time
--------  ------------  ---------  ------------------------
vserver1  NTAP_oradata  my-backup  Thu Mar 09 10:10:06 2017
```

The snapshot creation time is listed as March 9th and 10:10:06. To be safe, one minute is added to the snapshot time:

```
[oracle@host1 ~]$ sqlplus / as sysdba
Connected to an idle instance.
SQL> startup mount;
ORACLE instance started.
Total System Global Area 1610612736 bytes
Fixed Size                  2924928 bytes
Variable Size            1040191104 bytes
Database Buffers          553648128 bytes
Redo Buffers               13848576 bytes
Database mounted.
SQL> recover database until time '09-MAR-2017 10:44:15' snapshot time '09-
MAR-2017 10:11:00';
```

The recovery is now initiated. It specified a snapshot time of 10:11:00, one minute after the recorded time to account for possible clock variance, and a target recovery time of 10:44. Next, sqlplus requests the archive logs required to reach the desired recovery time of 10:44.

```
ORA-00279: change 551760 generated at 03/09/2017 05:06:07 needed for
thread 1
ORA-00289: suggestion : /oralogs_nfs/arch/1_31_930813377.dbf
ORA-00280: change 551760 for thread 1 is in sequence #31
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}
ORA-00279: change 552566 generated at 03/09/2017 05:08:09 needed for
thread 1
ORA-00289: suggestion : /oralogs_nfs/arch/1_32_930813377.dbf
ORA-00280: change 552566 for thread 1 is in sequence #32
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}
ORA-00279: change 553045 generated at 03/09/2017 05:10:12 needed for
thread 1
ORA-00289: suggestion : /oralogs_nfs/arch/1_33_930813377.dbf
ORA-00280: change 553045 for thread 1 is in sequence #33
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}
ORA-00279: change 753229 generated at 03/09/2017 05:15:58 needed for
thread 1
ORA-00289: suggestion : /oralogs_nfs/arch/1_34_930813377.dbf
ORA-00280: change 753229 for thread 1 is in sequence #34
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}
Log applied.
Media recovery complete.
SQL> alter database open resetlogs;
Database altered.
SQL>
```

Complete recovery of a database using snapshots using the `recover automatic` command does not require specific licensing, but point-in-time recovery using `snapshot time` requires the Oracle Advanced Compression license.

Database management and automation tools

The primary value of ONTAP in an Oracle database environment comes from the core ONTAP technologies such as instant Snapshot copies, simple SnapMirror replication, and efficient creation of FlexClone volumes.

In some cases, simple configuration of these core features directly on ONTAP meets requirements, but more complicated needs require an orchestration layer.

SnapCenter

SnapCenter is the flagship NetApp data protection product. At a very low level, it is similar to the SnapManager products in terms of how it executes database backups, but it was built from the ground up to deliver a single-pane-of-glass for data protection management on NetApp storage systems.

SnapCenter includes the basic functions such as snapshot-based backups and restores, SnapMirror and SnapVault replication, and other features required to operate at scale for large enterprises. These advanced

features include an expanded role-based access control (RBAC) capability, RESTful APIs to integrate with third-party orchestration products, nondisruptive central management of SnapCenter plug-ins on database hosts, and a user interface designed for cloud-scale environments.

REST

ONTAP also contains a rich RESTful API set. This allows 3rd party vendors to create data protection and other management application with deep integration with ONTAP. Furthermore, the RESTful API is easy to consume by customers who wish to create their own automation workflows and utilities.

Copyright information

Trademark information