QNAP

QuTS hero

ZFS

Whitepaper

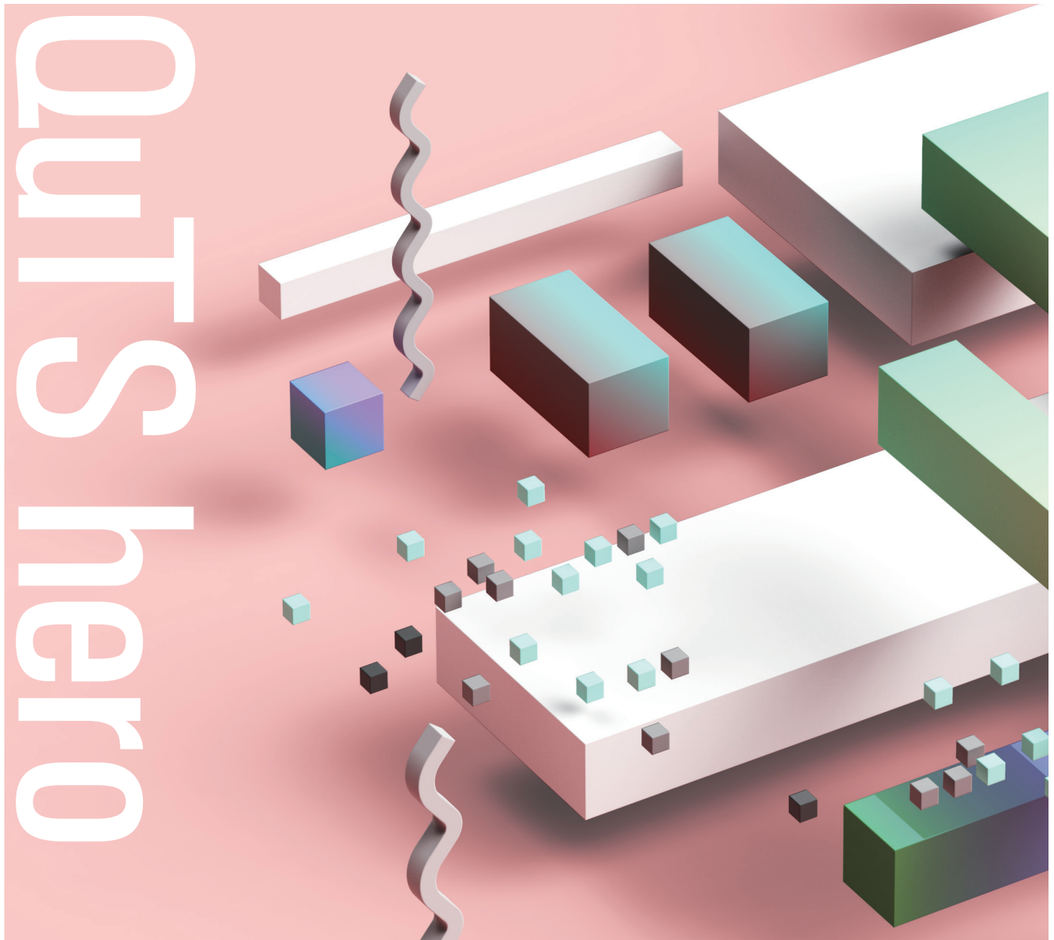
INDEX

QuTS hero
ZFS
whitepaper

Overview

In today's IT-intensive world, rapidly-changing requirements can be the key to the success or failure of enterprise organizations and data center operations. In this environment, budgets are under heavy scrutiny and any potential downtime - whether minutes or seconds - is potentially disastrous, and any potential increase in efficiency and productivity are important cornerstones for success.

The QuTS hero operating system adopts ZFS, a file system that prioritizes data integrity, to meet high-performance and high-stability data backup needs. QuTS hero integrates high-performance data protection, data reduction and virtualization applications, and supports cloud environments and QoS (Quality of Service) management functions to provide more cost-effective storage for small businesses, enterprise-level data centers, and virtualized work operations solutions.

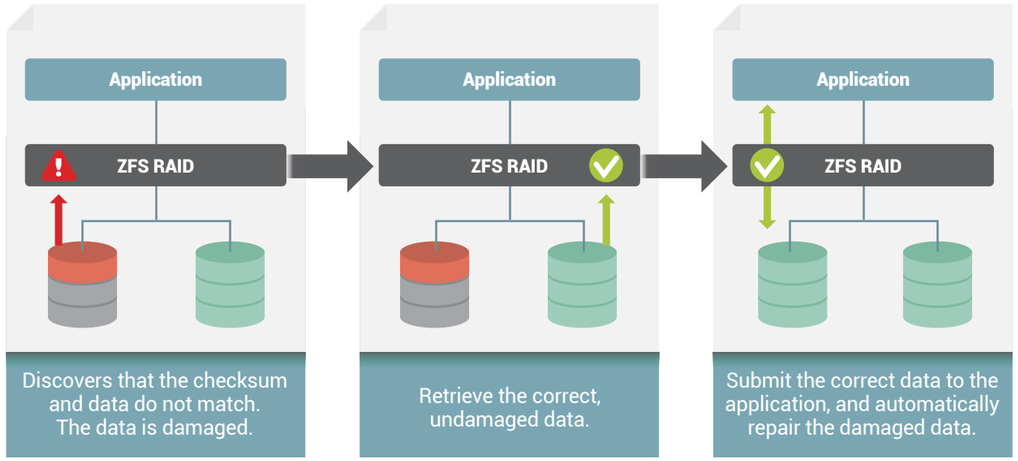


ZFS Introduction and configuration

Data Integrity and self-healing

The integrity and trustworthiness of data is key to ensure the operation of applications and databases. ZFS has the ability to prevent Silent Data Corruption, a potentially disastrous and hard-to-detect condition caused by hardware issues, defective cables, or metadata errors. ZFS has the ability to self-heal this data, including checking all data blocks and automatically repairing incorrect blocks.

It is strongly recommended to regularly perform "Pool Scrubbing" to ensure the integrity of stored data and to repair potential issues before they arise. You can configure Pool Scrubbing to run on a scheduled basis, such as weekly or monthly.



Prevents silent data corruption from infiltrating a running system.

Recommended Block size for Shared folder/ LUN:

128K - Video Editing / Large-Size Files / Backup
64K - Generic / Hyper-V
32K - VMware
8k - OLTP
4K - VDI / Database / Small-Size Files

Synchronous I/O for ZIL (Standard / Always / None):

The ZIL (ZFS Intent Log) is the write journaling mechanism that tracks file changes that have not been committed to the file system. The ZIL records the intentions of these changes into a log, known as a journal. In the event of a system crash or power failure, QuTS hero checks the journal logs and applies the scheduled changes, ensuring that the file system is quickly restored with a lower chance of data corruption.

When a shared folder/LUN is created, the ZIL write log I/O mode must be configured to prioritize data integrity or performance.

Standard (default) : I/O transactions are synchronous or asynchronous depending on the application and the type of I/O request.

Always: All I/O transactions are treated as synchronous, which improves data integrity but lowers performance.

None: All I/O transactions are treated as asynchronous, this option offers the highest performance, but has a higher risk of data loss in the event of power outage.

Please note that it is recommended to configure a dedicated ZIL as a SLOG on a dedicated high-speed storage device (such as an NVMe SSD). For the ZIL it is recommended to use ultra-low latency and high-endurance drives (such as MLC drives). For the read cache, throughput should be prioritized. It is recommended to split the ZIL and read cache to avoid premature failure of the entire SSD group due to the high ZIL write frequency.

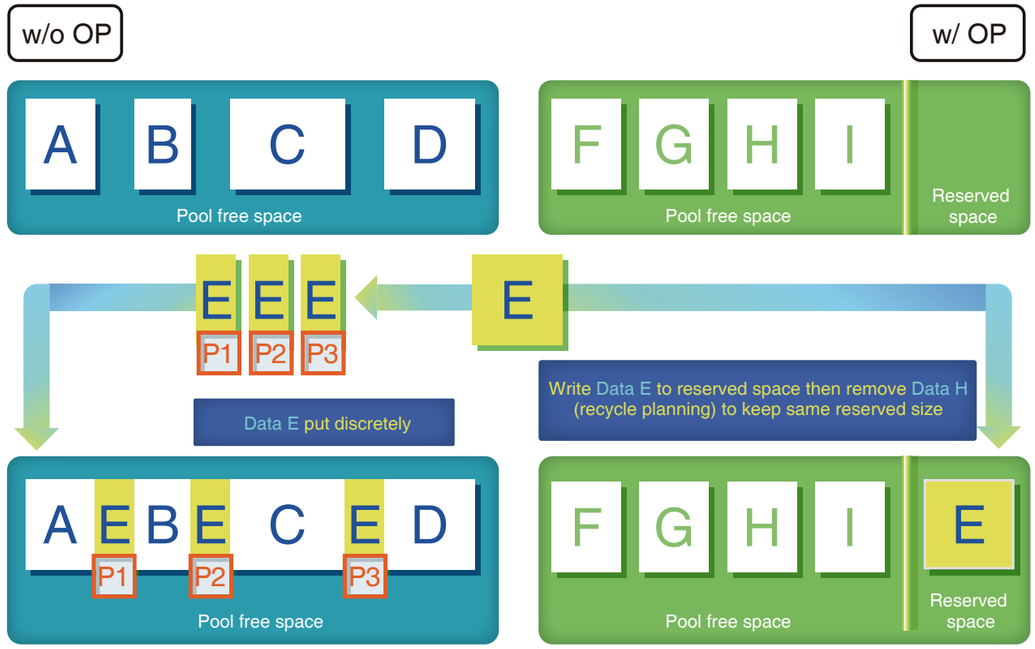
Pool over-provisioning

ZFS performance can be impacted when a storage pool is nearly full or fragmented. This is due to Copy-on-Write behavior: ZFS always writes new data to a different block, updates the metadata to point to the new location, and then frees up the old block, rather than directly overwriting the old data. After many rounds of writing the distribution of data on the storage pool becomes fragmented, which makes it more difficult and time-consuming for new data to be written.

QuTS hero supports Pool Over-Provisioning, which allows you to specify and reserve a percentage of space in the storage pool to ensure consistent random write performance when the pool is nearly full. The below figure illustrates data being written to a storage pool with and without Pool Over-Provisioning. Without Pool Over-provisioning, when the storage pool is nearly full and fragmented, the new written data blocks will be separated a while and performance impacted. With Pool Over-Provisioning, it becomes possible for ZFS to find a continuous big block to store new data.

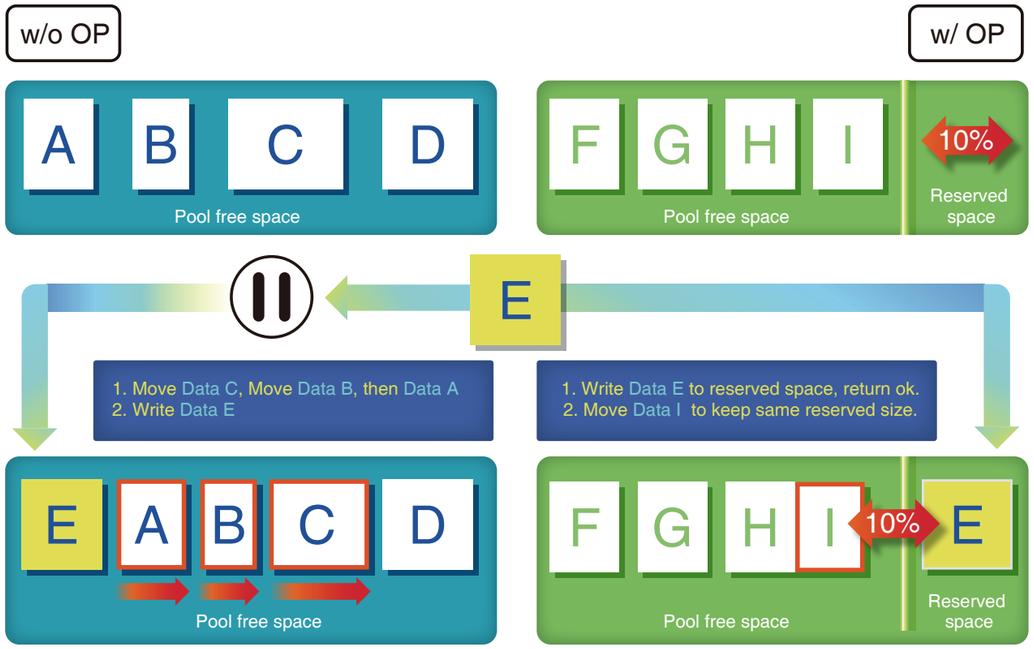
Over-Provisioning

Improves the performance of fragmented pools (for example: when writing a block to an **HDD**)



Over-Provisioning

Improves the performance of fragmented pools (for example: when writing a block to an **SSD**)



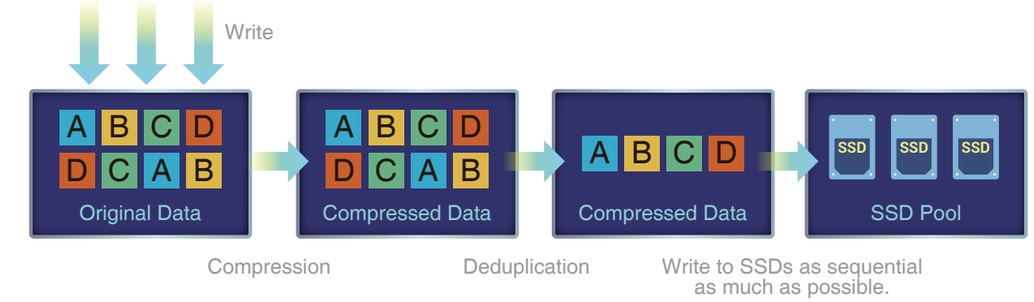
Enabling Pool Over-Provisioning ensures random write performance at the cost of storage capacity. The more space that is reserved, the higher the performance will be. The key to determine if such a tradeoff is necessary is to consider the read/write ratio and application scenarios. The more writes there will be, the higher the performance boost that Pool Over-Provisioning provides.

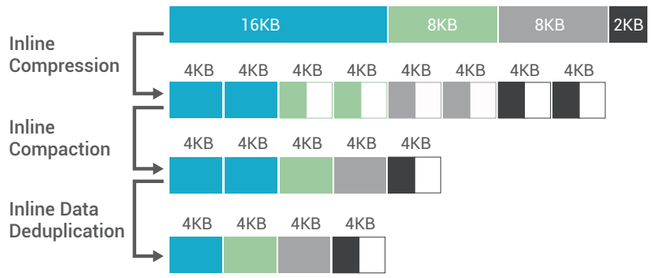
It is generally recommended to set the Pool Over-Provisioning value to 10-20%. For write-intensive and performance-demanding applications (like SQL logs, backup, surveillance and medical imaging) it is recommended to set a higher value. For read-intensive applications (like web servers, OLAP, and media streaming) the performance boost brought by Pool Over-Provisioning may not be worth the loss of storage space and a lower value can be used.

Data Reduction (Inline Compression / Inline Deduplication)

Inline Data Deduplication allows repeated data to be deleted before being written, which can greatly save storage space. Inline Compression is also supported to save space when saving large files. These features, along with Inline Compaction technology, provide the ability to greatly optimize storage utilization. When used in all-flash configurations that store highly-repetitive data or large amounts of small files, the ability to reduce storage requirements is greatly beneficial. It not only significantly improves the write performance of all-flash configurations, but also extends the lifespan of SSDs and maximizes the general cost-effectiveness of the deployment.

The compression rate can reach 90% for repetitive virtualized environments.





Inline Compression

QuTS hero supports inline compression, which reduces file and block sizes by encoding it with fewer bits than the original representation when the data is written to a shared folder or LUN. Enabling compression also increases read and write throughput as fewer blocks need to be read or written.

In QuTS hero, compression is lossless to ensure data consistency. ZFS supports LZ4 compression by default as it is generally the fastest.

This compression does not require excessive CPU resources and is recommended for use on shared folders and LUNs. Compression should not be used for files that have already been compressed (such as MP3 files, H.264 videos, and zip files) as the storage saved by re-compressing them is negligible.

LZ4 is a lossless data compression algorithm that provides extremely fast compression and decompression speeds. It is the best choice for low latency and high IO when transmission speeds are higher than 100 MB/s. It is particularly suitable for Inline Compression.

Compression Speed

740 MB/S

Decompression Speed

4530 MB/S

*Tested by Izbench

Inline Deduplication

QuTS hero uses target-based, block-level, and inline deduplication. Target-based means that the source site transfers data to the target site first, and then the deduplication process takes place at the target site. Block-level means that it can detect and remove small redundant chunks of the data within a file or among different files. Inline means that the deduplication process takes place immediately while the data is transferred to the storage device but before it is written. The advantages of this combination (target-based, block-level, and inline) is the reduction of required CPU resources of the client site, achieving higher deduplication ratios, and reducing the drive access requirements compared to other deduplication mechanisms.

The deduplication process consists of three steps: Slicing data into blocks, computing the fingerprint, and searching the Dedup Table. First, ZFS slices files into small data blocks. When a data block is written the system uses a cryptographic checksum to compute each data block’ s fingerprint, which is used to identify if there are any identical blocks in existing data. ZFS maintains a mapping table (called the Dedup Table) which contains all fingerprints, locations, and reference counts. If a data block is written to the NAS with the same fingerprint as an existing data block, instead of writing to a new location, ZFS will check the Dedup Table, assign the same location to this data block, and increase the reference count by one.

Inline deduplication reduces the need for extra storage space but requires more computing resources when writing data which may impact system performance. This performance impact will grow as the Dedup Table increases in size - as will the difficulty of searching the table. To avoid performance degradation during inline deduplication, QNAP uses the SmartDDT (Smart DeDuplication Table) algorithm to restrain the growing Dedup Table by monitoring the performance changes or the size of the Dedup Table. When some criteria are met, QuTS hero will stop the deduplication service. If this happens, either the data is not very repetitive and the application scenario may not be suitable for Inline Deduplication, or the NAS requires more memory.

For environments that require deduplication, we recommend installing as much memory as possible - 16GB memory at minimum, over 64GB is strongly recommended.

SnapSync:

the cost-effective replication solution for backup, data protection & disaster recovery.

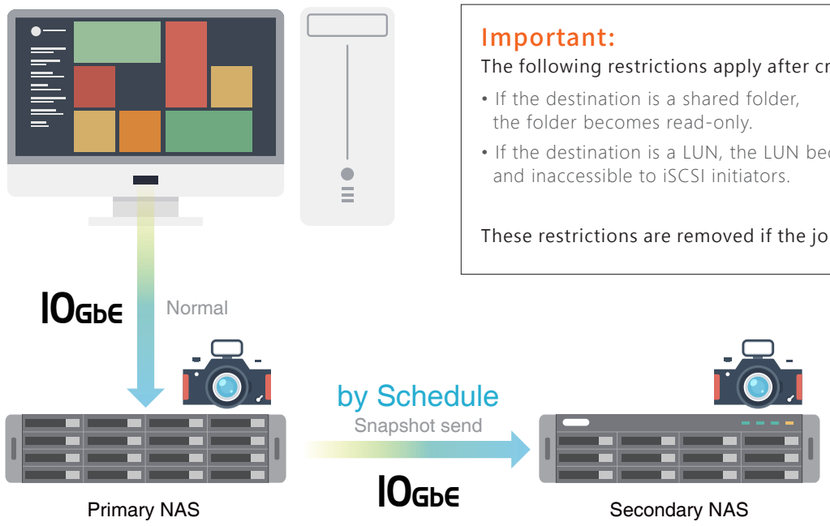
Introduction:

SnapSync is a cost-effective and easy-to-use unified replication solution built into QuTS hero for backup, data protection and Disaster Recovery purposes.

SnapSync is configured through a Mirror relationship between Shared/LUN on primary and secondary storage systems. It periodically syncs the replica to keep it up-to-date with changes that have been written to the primary. This mirrored data is created in a secondary storage system. You can failover the data from the secondary in the event of a disaster at the primary site. With SnapSync you can reduce the total cost of disaster recovery solutions, making it easier to justify investments by putting your disaster recovery site to active business use.

How SnapSync Works:

The block-level SnapSync function can back up variable blocks via snapshot, which is lightweight and bandwidth-saving, and key files can be efficiently backed up to remote locations. With incremental replication, the same data is never sent twice. SnapSync recovery times are also faster than other methods because restoration only requires a full backup and the last differential backup.



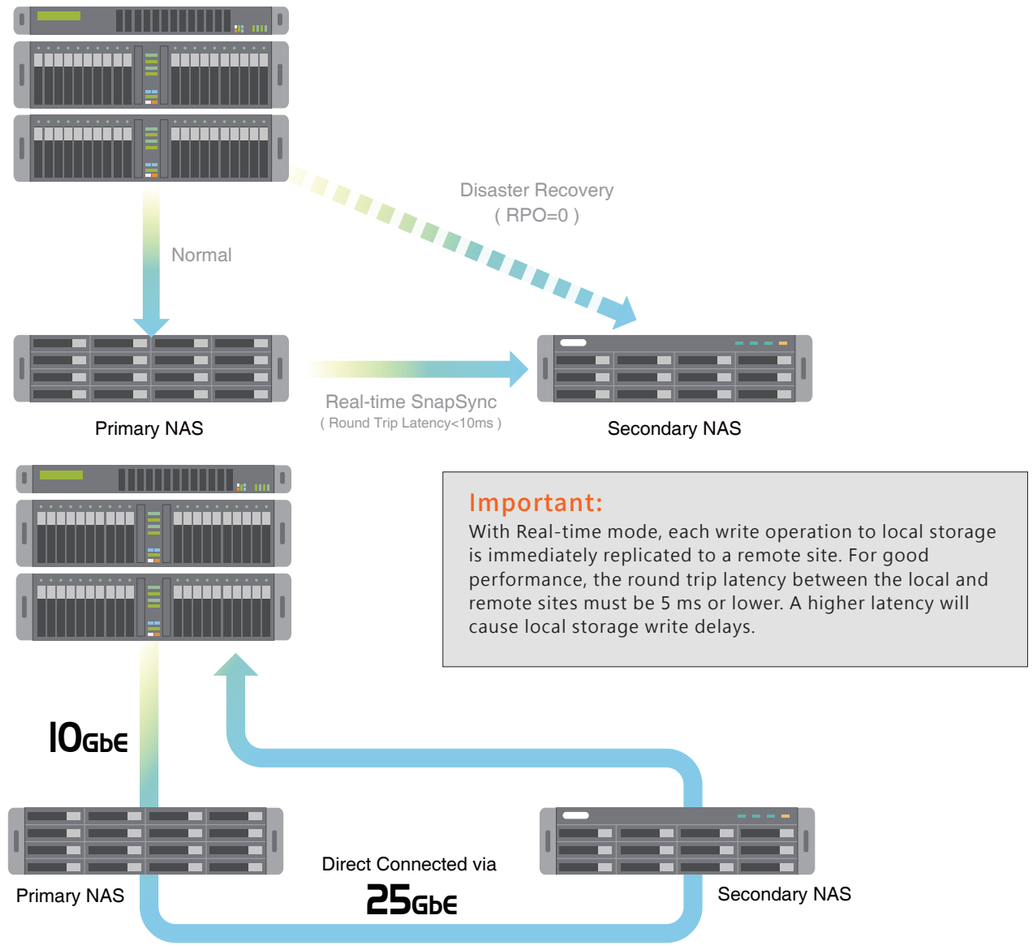
Important:

The following restrictions apply after creating a SnapSync job.

- If the destination is a shared folder, the folder becomes read-only.
- If the destination is a LUN, the LUN becomes read-only and inaccessible to iSCSI initiators.

These restrictions are removed if the job is deleted.

The more advanced Real-time SnapSync function allows the primary and secondary NAS to stay synchronized at all times.



Important:

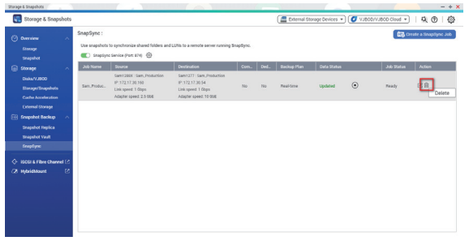
With Real-time mode, each write operation to local storage is immediately replicated to a remote site. For good performance, the round trip latency between the local and remote sites must be 5 ms or lower. A higher latency will cause local storage write delays.

Real-time SnapSync ensures that the data of the primary and secondary NAS are completely written before answering the host server. This ensures an RPO=0 (Recovery Point Objective), also means that the data in the secondary NAS is consistent with the primary NAS. In the event of a disaster occurring, the host server can immediately change the mounting target to the secondary NAS when the server fails over.

When you build a large storage system, it is recommended to set the secondary NAS with SnapSync, and then use Snapshot Replica & HBS3 to perform regular multi-version backups from the secondary NAS to another backup NAS.

Also supports QES NAS (QES 2.1.1 v11 or newer) SnapSync to QuTS hero NAS.

What to do when a disaster happens



The exclusive permissions of the folder will be removed when deleting the SnapSync task.

Best Practices for the configuration of Realtime SnapSync

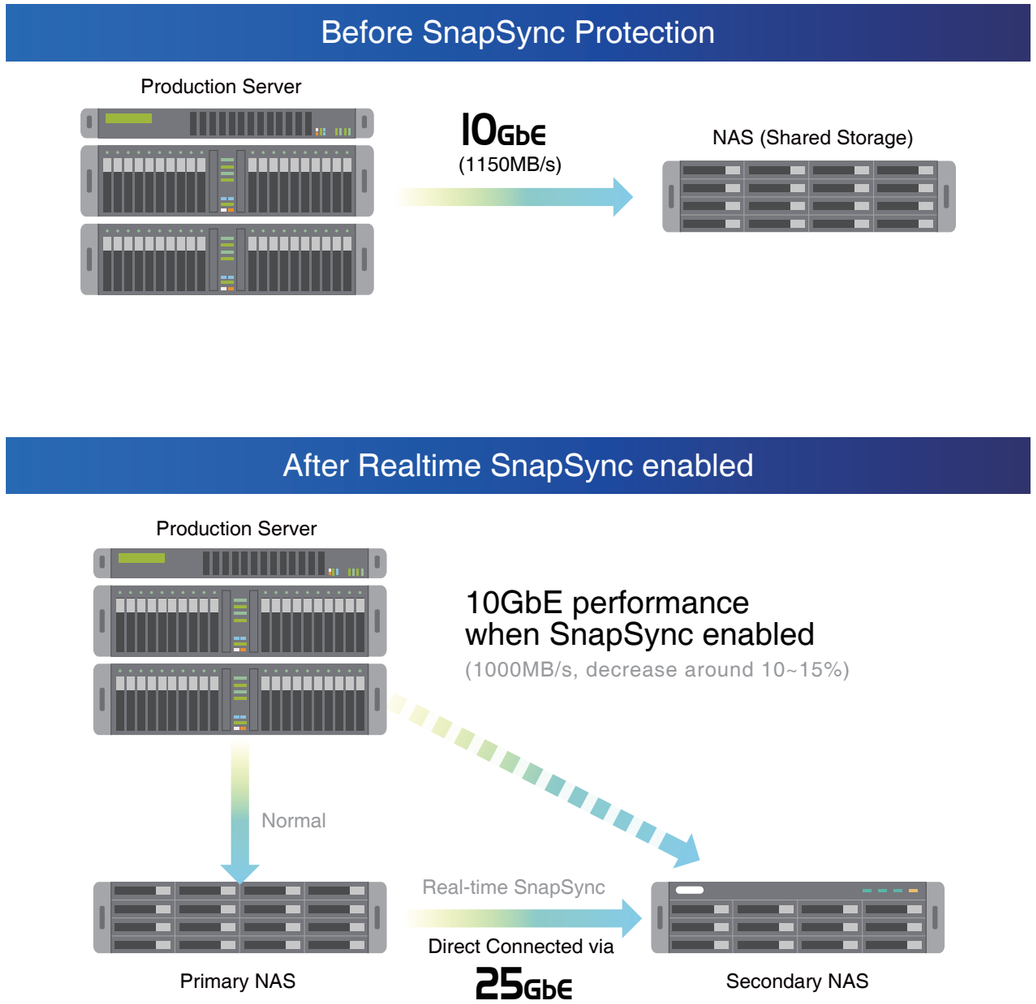
In Real-time mode, every write operation to local storage is immediately replicated to a remote site. For good performance, the round trip latency between the local and remote sites must be 5 ms or lower. A higher latency will cause write delays in local storage. Therefore, when configuring, please pay attention to the following:

- It is recommended that the network environment latency is less than 10 milliseconds as far as possible (less than 5 milliseconds is optimal)
- The I/O performance of the secondary NAS should be the same as the primary NAS.
- We recommend directly connecting the primary and secondary NAS using 25GbE.

Best Practices for Configuration

The performance benchmark of real-time SnapSync from QNAP Labs:

- Tested with the iSCSI protocol: When the user connects to the QuTS hero NAS with 10GbE, the file transfer of 10GbE can reach 1127 MB/s before SnapSync is enabled, and the efficiency is 1005MB/s when performing real-time SnapSync. It has around 10~15% attenuation.
- Tested with the SMB protocol: When the user connects to the QuTS hero NAS with 10GbE, the file transfer of 10GbE can reach 1025 MB/s before SnapSync is enabled, and the efficiency is 930MB/s when performing real-time SnapSync. It has around 10~15% attenuation.
- Tested with the NFS protocol: When the user connects to the QuTS hero NAS with 10GbE, the file transfer of 10GbE can reach 961 MB/s before SnapSync is enabled, and the efficiency is 897MB/s when performing real-time SnapSync. It has around 7~15% attenuation.



Test environment:

- IO mode : sync = standard & sync = none
- Block size = 128K
- Jumbo Frame (MTU) = 9000
- When the above test is connected with 10GbE in the same environment, because of the influence of network connection and delay, the attenuation after Real-time SnapSync is enabled will increase again, to about 20%.

Benefits:

Reduce downtime and protect against data loss

SnapSync provides scheduled and real-time replication. This enables Recovery Point Objectives (RPO) of seconds. You can meet stringent recovery objectives even for your largest write-intensive workloads.

Reduce Network Bandwidth Utilization:

SnapSync leverages storage efficiencies by sending incremental blocks over the network with compression/deduplication to accelerate data transfer and reduce bandwidth utilization. With SnapSync you can leverage one thin replication data stream to create a single repository that maintains the active mirror.

Distribute Large Data Easily:

Sometimes you must send large amounts of data: to migrate server room arrays, consolidate remote offices, or set up a new branch. SnapSync provides a fast, efficient, and flexible method to move data. If your business is geographically dispersed and all locations need access to the same dataset (such as training videos or sales kits) you can use SnapSync to quickly distribute the same data to all locations.

CDM (Copy Data Management) and Data Analytics:

Running extensive analysis might be critical for your business, but it may impact the performance of production environments. With SnapSync and Snapshot, you can leverage replicated data to run complex analysis on secondary data copies.

Data retention, compliance, and multi-version requirements:

Many use cases require lengthy data retention periods. By combining SnapSync and Snapshots, you can meet compliance requirements to protect data. Snapshot backup can assist in mitigating the impact of ransomware, and through real-time SnapSync increase availability and disaster recovery.

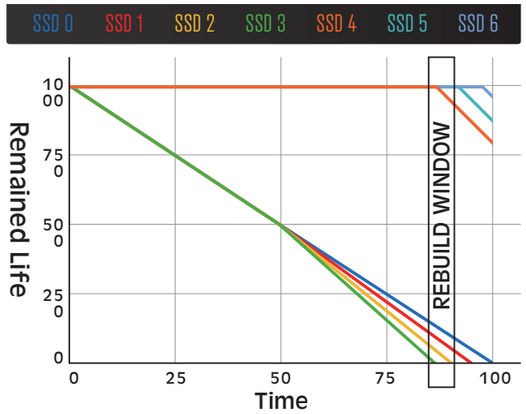
QSAL for all-flash arrays and SSD RAID (QNAP-Patented technology)

The patented QSAL (QNAP SSD Anti-wear Leveling) algorithm is automatically enabled and prevents multiple SSDs from malfunctioning at the same time.

Corresponding to SSD RAID 5 / 6 / 50 / 60 / TP (Triple Parity) will be enabled by default.

M0 OP = 0

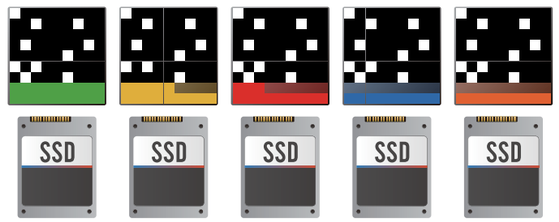
Mi OP = i x W / Mi-1, if (Mi-1 - Mi) > W



SSD RAID used by competitor products (particularly those that only increase the loading of the oldest SSD) has a particularly high risk of data loss at a later stage because half the SSD's end-of-life is too close to rebuild data. In addition, this algorithm has no effect on writing a large number of large files, and only supports RAID 5, which is another risk.

QSAL regularly detects SSD lifespan and durability, and when the SSD lifespan is under 50%, it will dynamically distribute the over-provision size then guarantee every SSD has enough time to rebuild before the end-of-life period expires. It achieves the best balance between utilization and overprovisioning. QSAL does not impact space utilization after activation, and only causes a negligible decrease in performance. Overall the protection of data in all-flash arrays will be greatly improved.

QSAL can be started at any time, and is also compatible with SSD Parity RAID that has not been configured before. It is recommended to enable QSAL before the SSD lifespan reaches 50%. If it is enabled too late, it may encounter insufficient rebuild time problems that result in the risk of damage to the SSD RAID.

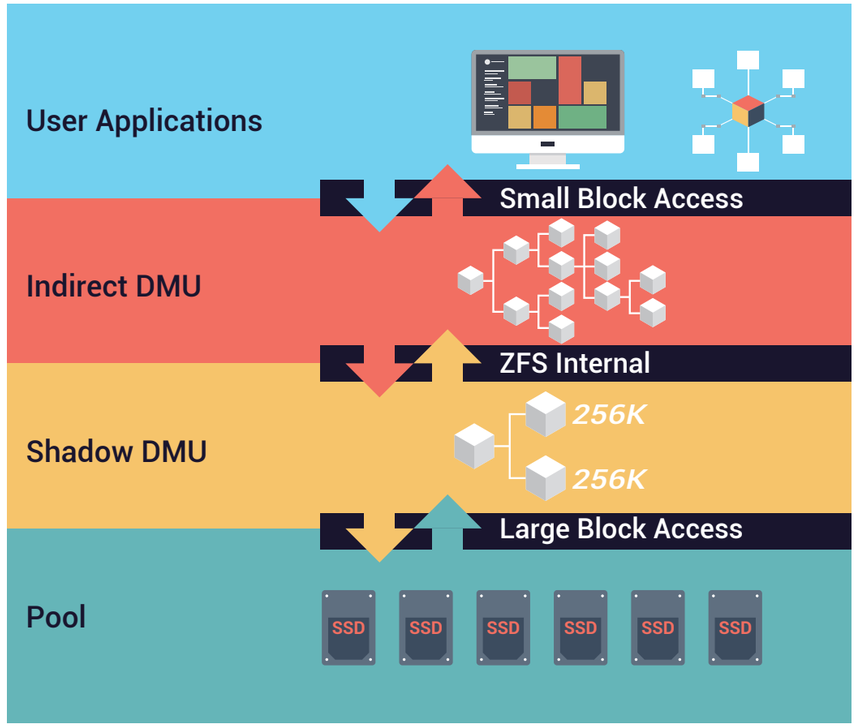


Patent numbers: US10705744B2 / US20200333959A1. Also includes several confidential patents.

Write Coalescing

The Write Coalescing algorithm drives random write performance for all-flash arrays

Write amplification is an undesirable effect of SSDs, bringing challenges to flash memory performance and endurance. QuTS hero features QNAP’ s exclusive Write Coalescing algorithm that is engineered for flash optimization by transforming all random writes to sequential writes along with reduced I/O. It not only effectively increases random write performance for all-flash environments but can also improve SSD lifespan.



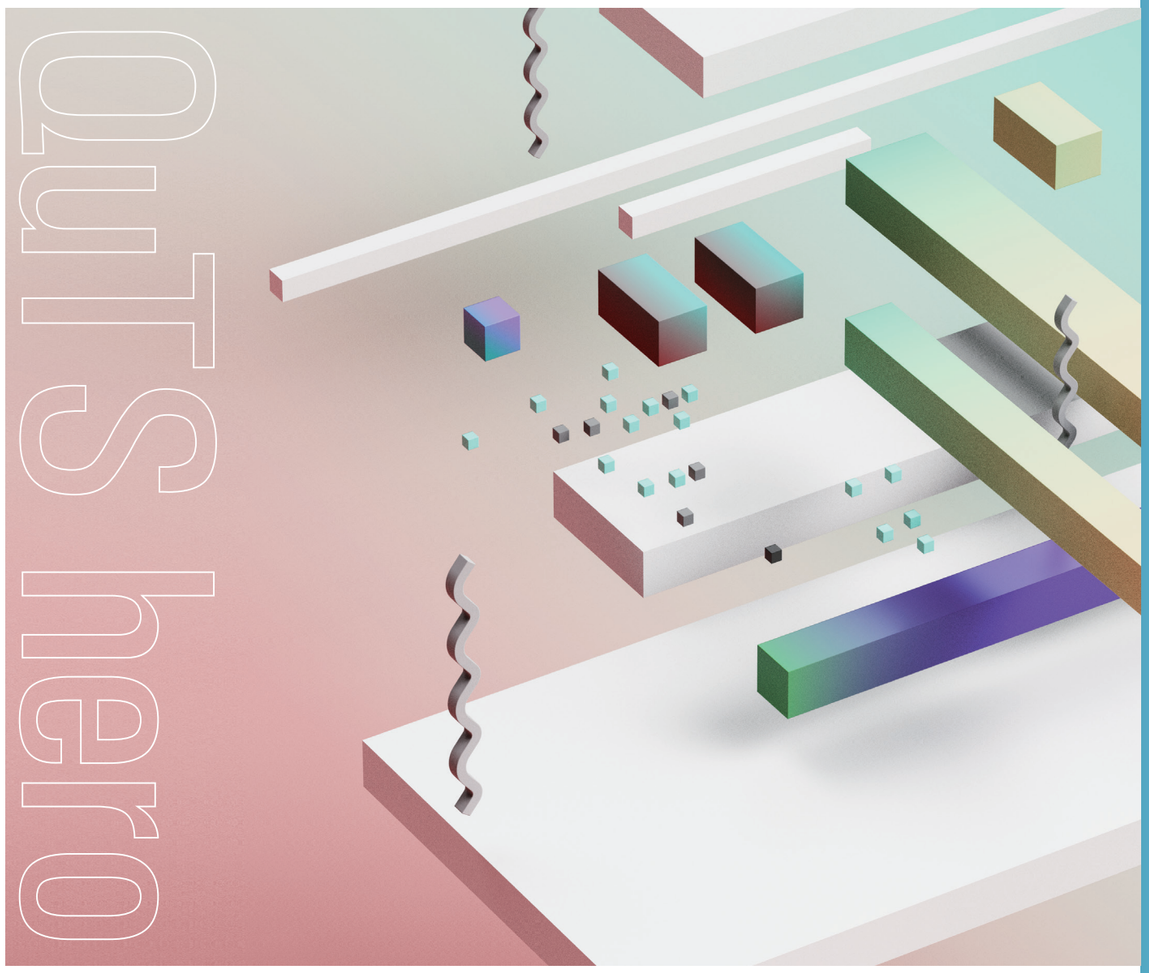
Recommended read cache configuration (L2ARC)

Memory plays a vital role in ZFS performance - especially in high-speed data transfer, data deduplication, ARC, and caching. It is recommended that you install as much memory as possible to attain the highest benefits of ZFS performance and optimized business workloads.

Based on the memory installed in a QuTS hero NAS, and the proportion corresponding to the use of the read cache, we recommend setting the size of the read cache as follows:

- **32GB RAM** supports 1TB SSD size for read cache
- **64GB RAM** supports 2TB SSD size for read cache,
- **128GB RAM** supports 4TB SSD size for read cache
- **256GB RAM** supports 8TB SSD size for read cache

- **1TB RAM** supports around 30TB SSD read cache.
- **4TB RAM** supports around 120TB SSD read cache.



Performance dependence between Memory size and Storage Pools

Important: This chapter only describes the basic dependence between RAM and Storage Pools. If the system is running other dedicated services - such as Virtualization Station - then it is recommended to separate the usage and consider the memory allocated to these virtual machines.

NAS Memory - VM assigned memory = ZFS used memory

This is the general concept for analyzing memory and storage performance, and how to identify the best practices of memory size for high-performance storage. We then describe how to choose sufficient memory for achieving the best performance.

Memory plays a vital role in ZFS and greatly impacts how QuTS hero storage pools perform. When ZFS is running with data reduction features (such as Compaction, Fast Clone, and Deduplication) the system needs some tables to record the reference count of these data blocks. Therefore, when the data amount becomes bigger, ZFS must hold as many reference tables in memory, and can handle IO requests more effectively.

Second, ZFS supports the Transaction groups function for data protection. Every file modification (e.g. a write) is associated with a certain transaction group (TXG). At regular intervals each TXG will shut down and the pool will issue a sync operation for that group. A TXG may also be shut down when the ARC indicates that there is too much "dirty" memory currently being cached. As one TXG closes, another will immediately open and file modifications then associate with the new active TXG.

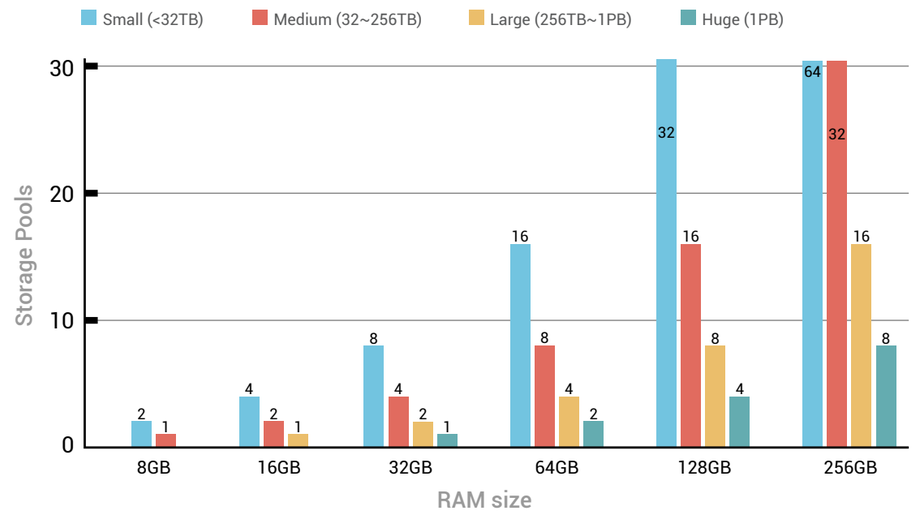
If an active TXG shuts down while a previous one is still syncing data to storage, then applications will be throttled until the running sync completes. In this situation where a TXG is syncing, while TXG + 1 is closed due to memory limitations and is waiting to sync; applications are throttled waiting to write to TXG + 2. We need sustained saturation of the storage or a memory constraint in to throttle applications.

A sync of the Storage Pool will involve sending all level 0 data blocks to disk. When finished, all level 1 indirect blocks (etc.) are sent until every block representing the new state of the filesystem has been committed. At that point the uberblock is updated to point to the new consistent state of the storage pool.

So, ZFS must update the uberblock after finishing every transaction, which is a large overhead. If the system can provide as much memory as possible, the transaction groups can be bigger, and the overhead of updating the uberblock will become relatively smaller. Thus, more memory can effectively help ZFS to improve write IO requests.

You can refer to our below recommendations and choose sufficient memory for different storage configurations.

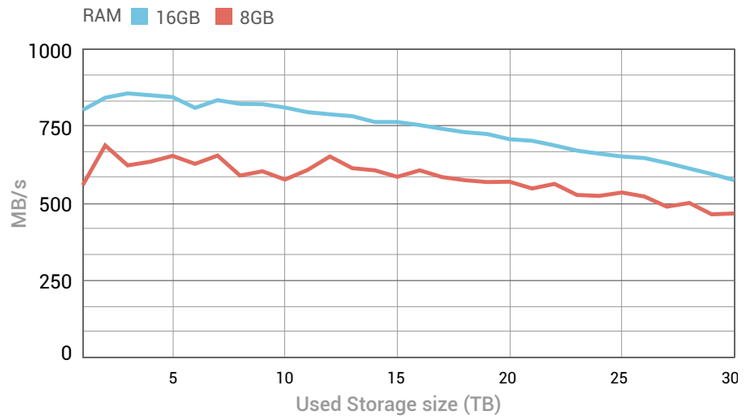
The Best Practices for Performance between RAM size and Pool Configuration



For example, if your ZFS NAS has 16GB memory, the recommended configuration of the storage pool would be 1 large Pool (256TB~1PB), or 2 medium size pools, or 4 small pools. This type of pool configuration would allow your system to achieve good performance. If you create pools that exceed these recommendations, your NAS will still reliably operate but with potentially decreased performance. Here we use our entry-level ZFS model TS-973AX as a reference to show the different performance between 8GB (minimum requirement) and 16GB RAM.

(We already created one SSD pool and one HDD pool around 35TB. The performance difference of the HDD pool is displayed).

TS-h973AX-16GB vs 8GB - IO Performance (5xHDD-R0)



QuTS hero
ZFS
whitepaper

QNAP SYSTEMS, INC.

TEL : +886-2-2641-2000 FAX: +886-2-2641-0555 Email: qnapsales@qnap.com

Address : 3F, No.22, Zhongxing Rd., Xizhi Dist., New Taipei City, 221, Taiwan

Netherlands (Warehouse Services)
Email: nlsales@qnap.com
TEL: +31(0)107600830

China
Email: cnsales@qnap.com
TEL: +86-400-028-0079

Thailand
Email: thsales@qnap.com
TEL: +66-2-5415988

Japan
Email: jpsales@qnap.com
FAX: 03-6435-9686

US
Email: usasales@qnap.com
TEL: +1-909-595-2782

India
Email: indiasales@qnap.com

Germany
Email: desales@qnap.com

France
Email: Frsales@qnap.com