

Auto-Sync User Guide

Table of Contents

[Auto-sync: the service that replicates both data and metadata](#)

[Introduction](#)

[The difference](#)

[A Note on '/'* \(dircontent\) option](#)

[How auto-sync works](#)

[Incremental backup and Recovery](#)

[How to track activity of auto-service](#)

[How to identify that auto-sync really transfers data](#)

[How do I make changes to existing auto-sync service](#)

[List of auto-sync properties](#)

[NexentaStor auto-sync: even more details](#)

Auto-sync: the service that replicates both data and metadata

Designed from ground up for consistency and ease of use, NexentaStor provides the same general interfaces to manage all storage services. All NMC commands, NMV UI pages, and all general principles discussed in the previous section with regard to **auto-tier** service apply to **auto-sync** as well.

For an overview and background information on NexentaStor replication services, please refer to Section "[Data Replication](#)".

Introduction

Both auto-sync and auto-tier are **schedulable, fault-managed, fully configurable (tunable)** NexentaStor [Data Replication](#) services that can be used in a variety of backup, archiving, and [DR](#) scenarios.

- For general and background information, please see [Data Replication](#).
- For NexentaStor Automatic Continuous Data Protection (CDP) extension, please see [AutoCDP](#).



There are important differences between the appliance's replication storage services. One size does not in fact fits all - if it weren't true a single replication service would suffice.

When deciding which replication service to deploy in your environment, please make sure to read the rest of this section..

Both auto-sync and auto-tier are designed from ground up to use a **variety of transports** (a.k.a. protocols), which provides required flexibility to execute over Internet and Intranet, from behind a firewall and in the environment that requires extra security.

Both auto-sync and auto-tier support incremental intelligent replication, whereby, once the initial replication is done, only accumulated differences (the delta) between the source and the destination is transferred during all subsequent service runs.

Both auto-sync and auto-tier support **any schedule**. You can schedule the services to run every minute, every hour at a given minute of the hour, every few hours, every day at a certain time, etc. You can schedule services to run once a year, or at certain day of every second month, and so on.

Both auto-sync and auto-tier support all **3 possible directions** of the replication: local to local (**L2L**), local to remote (**L2R**), and remote to local (**R2L**). When replicating to or from remote host, the latter does not necessarily need to be a NexentaStor appliance, although in the auto-sync case it must be another ZFS based system.

- Both auto-services can be set up to **run only once** - at a given scheduled time.
- auto-sync can execute in a **daemon mode** and run incremental replications every second or every few seconds. The equivalent tiering capability will be added in the near future.
- auto-sync can be used to replicate locally or remotely the appliance's system folder (a.k.a. **root filesystem**) that contains appliance's Operating System and configuration. The replication destination may or may not be another NexentaStor appliance, and - in the case when it is an appliance - may or may not reside on appliance's system volume. The equivalent tiering capability is not being planned.

Both auto-sync and auto-tier provide a combined **replication + snapshots** capability. You can

tier from a given source (for instance, from a given snapshot or a directory), and generate snapshot at the remote or local destination every time the replication has run.

Both auto-sync and auto-tier are **recursive**. The auto-sync service will replicate its designated source folder and **nested** sub-folders. Similarly, the auto-tier service replicates its source directory and nested sub-directories, which may include nested folders/filesystems. Both services regularly and incrementally replicate a given source. Both services accept **exclude** option, to exclude certain sub-directories (auto-tier) or sub-folders (auto-sync) from copying.

The difference

The primary difference between auto-sync and auto-tier replication services is threefold:

1. **Data and meta-data**. Auto-sync transfers not only data (files, directories) but filesystem **meta-data** as well, including snapshots.
2. **Folder and Directory**. Auto-tier can have a **directory** within a filesystem as its top level source, while auto-sync cannot. To be able to transfer meta-data, auto-sync must have a **folder** (filesystem) as its top level source.
3. **Copying over**. When executing the very first time, auto-tier can write the source/destination differences over an existing files and directories at the destination. When executing the very first time, auto-sync cannot copy over an **existing** destination - it will create new folder(s) at the destination, and keep those folders fully in-sync with the source folders after each subsequent scheduled run of the service.

Initial auto-sync service requires empty destination. Initial auto-sync will effectively create a new folder (or nested folders/filesystems) at the destination. Those new folders will be complete clones of the folders at the source.

In contrast, the auto-tier service will only send differences during tiering. The following picture attempts to illustrate the latter:

In conclusion:

Use auto-sync to transfer (effectively, [clone](#)) sub-folders and [zvols](#).

Use auto-tier to transfer files and sub-directories.

(*) *For simple, local, one-time cloning of a given dataset, please see the F.A.Q. article:*

[How can I clone: \(1\) a folder, and \(2\) a zvol?](#)

A Note on '/'* (dircontent) option

In general, replication source may have two forms: [source location]/ and [source location]/*

As far as auto-sync and auto-tier replication services are concerned, the /* at the end translates as: **copy only the contents** of the given source location but not the source location itself. This is further explained in the [User Guide](#), Sections "Setting up a tiering service" and "Auto-sync: the service that replicates both data and metadata".



A Note on '/'* (dircontent) option

Replication source may have two forms: <source URL>/ and <source URL>/*

The difference between the two is easier to explain on example. Let's say, the **folder** that is being replicated is **vol1/a** and it contains **sub-folder 'b'**:

```
`-- vol1
  |-- a
     |-- b
```

Let's now say that the destination is **vol2/x**, possibly on a different host. If the source is **vol1/a**, the resulting destination, after service has run, will look like:

```
`-- vol2
  |-- x
     |-- a
        |-- b
```

However, if the source is **vol1/a/*** (notice the '*'), result of the replication will look like:

```
`-- vol2
  |-- x
     |-- b
```

The /* (dircontent) processing logic applies to both auto-sync and auto-tier, except that in the latter case the source can be sub-directory and not necessarily a folder (filesystem).

Rest of the article talks specifically about auto-sync, as it the auto-sync service that proved to invoke more questions..

How auto-sync works

Auto-sync execution over time can be viewed as an infinite chain of incremental backups. On one hand, marker snapshot contain all data inside (full backup). On the other hand, just incremental data is transferred during each iteration. All nodes (markers) of the chain must be transferred successfully to maintain auto-sync service in working state. If one node "breaks", auto-sync transitions into maintenance.

Marker snapshots:

On each iteration, auto-sync creates a new "marker" snapshot. The marker's name consist of word 'sync', period of service and unique id of 32 symbols length (for example - @sync-minute-1:366e62911ca554fe0f8775452408cd37). Each iteration, auto-sync creates a new marker snapshot, and then transfers a difference (delta) between the old marker and the new marker to its (auto-sync) destination. So normally, there must only one marker snapshot (per folder or zvol) when auto-sync is idle. If auto-sync is active, there should be two markers at time at source. If auto-sync is idle (not currently running) and there are two snapshot markers, that would mean that the previous auto-sync run was not successful. Automatic recovery action will take place in this event.

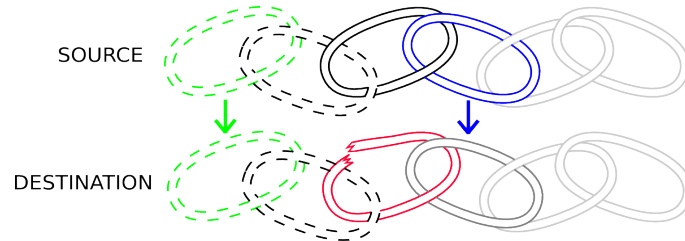
Note that if you have nested folders/zvols, auto-sync creates marker snapshot for each folder or zvols. For example, if you have vol/a, vol/a/b, vol/a/b/zvoltest datasets, there will be 3 marker snapshots for each iteration.

Incremental backup and Recovery

The appliance's auto-sync service runs on (its own per-service schedule) and transfers a delta between two consecutive "marker" snapshots. As mentioned above, the service execution over

time can be visualized as a "chain" of incremental backups:

The main idea behind auto-sync recovery process is therefore to locate the last "healthy node" in that "chain" - that is, any snapshot that was fully transferred to the destination and all its sub-folders (if any). This "healthy node in the chain" must exist for every nested dataset (if any) of a given source folder, and could be a marker snapshot or any other ZFS snapshot.



Autosync cannot proceed, as old marker (red) is damaged. 'reset' function will find healthy snapshots and will rename it to auto-sync marker for next iteration.

Once a "healthy" pair of snapshots is located at the source and the destination, the destination is rolled back to that snapshot. Later, normal auto-sync iteration starts and transfers the accumulated delta from the source.

How to track activity of auto-service

In NMC, run:

```
nmc$ show auto-sync : logtail -f

Example:

nmc$ show auto-sync :data-fineline-000 logtail -f

[ Jan 05 10:34:08 (zfs-auto-sync, 1656) from: localhost:/data/fineline, to:
localhost:/data/tier_to, proto=zfs, auto_mount, marker=...ad0d ]

[ Jan 05 10:34:10 (zfs-auto-sync, 1656) data/fineline@snap-
minute-1-2010-01-05-1032: destroyed as per retention policy ]

[ Jan 05 10:34:13 (zfs-auto-sync, 1656) data/fineline@snap-
minute-1-2010-01-05-1034 OK ]

...
```

How to identify that auto-sync really transfers data

There are several ways. You could check the auto-sync transport itself:

```
nmc$ show auto-sync : stats -i 1

Example:

nmc$ show auto-sync :data-folder-000 stats -i 1

TCP CONNECTIONS          SNEXT          RNEXT          TRANSFER
192.168.3.210.22-192.168.3.55.36373 3552678986 3152922376 -
192.168.3.210.22-192.168.3.55.36373 3552679178 3152922376 192 B
192.168.3.210.22-192.168.3.55.36373 3552679290 3152922376 112 B
192.168.3.210.22-192.168.3.55.3637 3552679402 3152922376 112 B
For more information, re-run command with '-h' option.
```

2. Secondly, check the overall volume I/O activity:

```
nmc$ show volume <volume-name> iostat
```

How do I make changes to existing auto-sync service

While the very basic properties of the auto-service (source/destination, period) are fixed, other properties can be edited and can influence on service's behaviour

```
nmc$ setup auto-sync <auto-sync instance> property
```

List of auto-sync properties

'comment'	Optional comment that helps identify the purpose of this service, its criticality, business related aspects, etc. Informational comment can be modified or added after the service is created.	
'force'	if true (non-zero), auto-sync service performs forceful operations and generally, disregards exceptions (if any). Must be turned on if the auto-mount option is used.	
'latest- suffix'	custom suffix for latest snapshots instead of default 'latest'	
'day', 'hour', 'minute'	auto-sync service schedule. For example, if 'hour' property is not defined, daily service will run on start of the day. But if you set 'hour' to 4, the service will run at 4 am every day	
'keep'	Number of date based snapshots to keep at the source.	
'keep	days'	Number of days to keep snapshots at the destination.
'retry'	Retry the service if it fails to execute because of the loss of connectivity with remote host.	
'rate_limit'	Limits transfer rate of auto- sync service.	
'trace_level'	Show debug information	

NexentaStor auto-sync: even more details

By default(*) auto-sync is based on '**zfs send/receive**' transport. Independently of its underlying transport, **auto-sync always re-creates source snapshots at the destination.**

For instance, auto-sync that uses '**rsync**' transport will perform rsync from all newly created snapshots at the source, and each time a given snapshot gets transferred, it will also snapshot the destination, thereby re-creating source snapshots at the destination. Note that the "newly

created snapshots" in the previous statement simply mean those snapshots that were created since **the previous** scheduled or manual (on demand) auto-sync run^(**).

auto-sync over zfs send/receive

There is one common mistake, in terms of using auto-sync, and this is related to changes at its destination. Note that for auto-sync over '**zfs send/receive**', it is recommended not to mount the destination. Generally, mounting auto-sync destination is an attribute change performed on the destination folder (filesystem).

More importantly though, mounted auto-sync destination exposes the latter to changes, including changes in the ZFS meta information. Any change in the data or meta-data at the destination is "noticed" by auto-sync as a loss of synchronization between the source and the destination, and is getting processed as a (recoverable) error condition.

There is one controlled way to circumvent this 'zfs send/receive' transport limitation - and that is by using auto-sync '[auto-mount](#)' property.

By default 'auto-mount' is disabled, which means that independently of the transport used to transfer data, snapshots, and folders/filesystems from its source to destination, auto-sync will leave the newly created folders at the destination NOT mounted. You will not see those folders via 'ls' or any other conventional file level access mechanism. To explore auto-sync created folders at the destination:

- use 'zfs list' or 'show folder' commands, or
- destroy auto-sync

When destroyed, a given auto-sync service instance automatically mounts its destination. Prior to destroy operation, you can store the service in persistent memory by performing:

```
nmc$ setup appliance configuration
```

This will allow to restore this service at any later time.

Therefore:

- if you need to mount auto-sync destination, and for whatever reason 'auto-mount' is disabled - you can do so with caution.
- Or, you can destroy auto-sync service instance altogether. At this point the replication destination will be automatically mounted and available for any access.

In all cases, if auto-sync service gets disrupted by changes at the destination, it will try to automatically recover by performing a rollback of the destination to the previous most recently transferred snapshot. to re-cover, use NMC 'setup auto-sync reset' command, or a similar GUI management functionality.

Note that auto-sync over zfs send/recv relies on the fact that the destination is not "disturbed" in any way. Any (data or metadata) change at the destination triggers an out-of-sync exception, with the subsequent automated rollback to the previous snapshot. You can manually **reset** auto-sync, to destroy marker snapshots on both sides and reset service metadata to its initial state. The top stream snapshot at the destination will be preserved and its presence is important.

Furthermore, auto-sync service will fail during reset-recovery if it does not find matching top-stream snapshot at the destination. In that sense, two auto-sync retention policy parameters are important:

- Keep on destination
- Keep on the source.

Try to tune these two values in such a way that the number of snapshots kept at the destination ("Keep on destination") is at least 3 times greater than the number of snapshots kept on the source. This creates a window of opportunity for reset operation to recover successfully.

For the most recent auto-sync features and improvements, please see:

- [What's new in auto-sync replication](#)

auto-sync over rsync

As far as auto-sync over rsync, note that **rsync** transport provides a number of benefits compared to 'zfs send/receive'. Unlike zfs send/receive, rsync is a true application level protocol on top of TCP. Both rsync client and server are multi-threaded, performance optimized. Using rsync often means getting a better performance, which is one main reason that rsync is present as an option for auto-sync service, even though rsync does not understand and **does not transfer ZFS ACLs and ZFS properties**.

Notice again that, independently of its underlying transport, **auto-sync always re-creates source snapshots at the destination**.

Successful auto-sync reset operation may save you time on initial syncing!

See also the following F.A.Q. entries:

- [I would like to backup not only my data but snapshots as well, please advise..](#)
- [How can I protect/replicate/recover/restore appliance's system configuration and the OS itself](#)
- [How can I use NexentaStor to backup/archive/restore/replicate/recover my data?](#)
- [What's new in auto-sync replication?](#)

(*) Note that 'zfs send/receive' is not a preferred transport, it is just the default one. A given auto-sync service instance can use a variety of transport mechanisms, including **rsync**.

(**) You can run just-in-time any given NexentaStor storage service using NMC 'run' command. Simply type 'run' and press TAB-TAB. The same functionality is available via NMV (web GUI).

By default¹, auto-sync is based on '**zfs send/receive**' transport. Independently of its underlying transport, **auto-sync always re-creates source snapshots at the destination**.

For instance, **auto-sync** that uses '**rsync**' transport will perform rsync from all newly created snapshots at the source, and each time a given snapshot gets transferred, it will also snapshot the destination, thereby re-creating source snapshots at the destination. Note that the "newly created snapshots" in the previous statement simply mean those snapshots that were created since **the previous** scheduled or manual auto-sync run^(**).

There is one common mistake, in terms of using auto-sync, and this is related to changes at its destination. Note that for auto-sync over 'zfs send/receive', its destination cannot and should not be mounted, except in when using '[auto-mount](#)' property. The latter is one of those advanced-usage type options that require extreme caution.

The safer choice would be using '[mount-destination](#)' sub-command available via NMC.

Generally, mounting **auto-sync** destination is:

1. An attribute change performed on the destination folder (filesystem). Note that any change at the destination means a loss of synchronization with the **auto-sync** source.
2. Secondly, mounted **auto-sync** destination exposes the latter to changes, including changes in the ZFS meta information.

Any change in the data or meta-data is "noticed" by the **auto-sync** service as a loss of synchronization between the source and the destination, and is getting processed as a (recoverable) error condition.

There is one controlled way to circumvent this 'zfs send/receive' transport limitation - and that

¹ Note that 'zfs send/receive' is not a preferred transport, it is just the default one. A given auto-sync service instance can use a variety of transport mechanisms, including **rsync**.

is by using auto-sync '[auto-mount](#)' property.

By default 'auto-mount' is disabled, which means that independently of the transport used to transfer data, snapshots, and folders/filesystems from its source to destination, **auto-sync** will leave the newly created folders at the destination NOT mounted. You will **not** see those folders via 'ls' or any other conventional file level access mechanism. To explore auto-sync created folders at the destination:

- use '**zfs list**' or '**show folder**' commands, or
- destroy auto-sync, or
- use '**setup auto-sync ... mount-destination**'

When destroyed, a given **auto-sync** service instance automatically mounts its destination. In all cases, if **auto-sync** service gets disrupted by changes at the destination, it will try to automatically recover by performing a rollback of the destination to the previous most recently transferred snapshot. to re-cover, use NMC '**setup auto-sync reset**' command, or a similar GUI management functionality.



To protect, replicate, recover, or restore appliance's configuration, and/or to clone the entire appliance's root filesystem, please see the following entry in the Section "[Frequently Asked Questions](#)" above, or on the website support page::

[How can I protect/replicate/recover/restore appliance's system configuration and the OS itself](#)

Both **auto-sync** and **auto-tier** can be monitored in real time, in terms of bytes transferred and bandwidth. Here's an example that effectively shows auto-sync generated traffic at one-second interval:

```
nmc:/$ show auto-sync :vol1-a-000 stats -i 1
```

TCP CONNECTION	SNEXT	RNEXT	TRANSFERED
192.168.37.128.39305=>192.168.37.134.22	2176247586	4217818800	-
192.168.37.128.39305=>192.168.37.134.22	2176255378	4217819152	8.14 KB
192.168.37.128.39305=>192.168.37.134.22	2176266306	4217819632	11.41 KB
192.168.37.128.39305=>192.168.37.134.22	2176273922	4217819952	7.94 KB
...			