FlexSync Man Pages Reference Guide

6-69063-01, Rev. A

Quantum

FlexSync Man Pages Reference Guide, 6-69063-01, Rev. A, April, 2023, Product of USA.

COPYRIGHT STATEMENT

Preface

This document provides the Command Line Interface (CLI) commands supported by FlexSync 3.x.

Audience

This manual is written for FlexSync operators, system administrators, and field service engineers.
(missing or bad snippet)(missing or bad snippet)

For the most up to date information on FlexSync, see:
(missing or bad snippet)(missing or bad snippet)(missing or bad snippet)(missing or bad snippet)(missing or bad snippet)

NAME

flexsync − FlexSync mover CLI

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **sync** operations. The man page for the **object** operations is the flexsync(1) man page.

SYNOPSIS

Local:

flexsync [OPTION...] SOURCE_DIR DESTINATION_DIR
flexsync [OPTION...] SOURCE_DIR/ DESTINATION_DIR

Remote:

flexsync [OPTION...] flexsync://HOST[:PORT]/SOURCE_DIR DESTINATION_DIR
flexsync [OPTION...] flexsync://HOST[:PORT]/SOURCE_DIR/ DESTINATION_DIR

DESCRIPTION

flexsync is the command line interface for initiating non-recurring FlexSync replications. All of the **flexsyncd(8)** task configuration options are available here as command line options.

GENERAL

flexsync operates in "pull" mode, so the source directory can be local or remote, but the destination directory must be visible locally on the destination host.

If the source directory is specified with a trailing slash, the contents of the directory are synced to the destination directory. Otherwise, a directory with the same name as the source directory is created under the destination directory, and the source contents are synced there.

OPTIONS

Options that start with -- and take an argument can have the argument separated by either a space or an equal sign, e.g.

--mode *Mode*
--mode=*Mode*

are equivalant.

--acls|**-A**
Preserve StorNext ACLs (super-user only and StorNext file systems only).

--archive|**-a**
Equivalent to -Dgoptr.

--bwlimit=*bytes_per_second*
Limit I/O bandwidth usage to the specified number of bytes per second.

--credits
Print legal disclosures.

-d Enable debug logging for all modules.

--debug=*[all|module,...]*
Enable debug logging for specified modules. Default is 'all'. Modules: 'flexsync', 'http', 'task-report'.

--delete
Delete extraneous files from destination.

--devices
Preserve Unix device files (super-user only).

-D Same as --devices --specials.

--force|-f
Don't check if the destination is a mount point, never prompt.

--exclude=*'PATTERN [PATTERN] ...'*
Specify shell glob patterns, separated by a space, describing path names to exclude. The patterns are applied in the order specified. A pattern starting with the slash character is anchored to the top-level source directory and will only match starting at that level. For example, the pattern '/dir1' will match /source/dir1, but not /source/dir2/dir1, whereas the pattern 'dir1' will match both of those directories. Leading spaces, ending spaces, filenames with spaces, and/or directories with spaces are not allowed. (The glob pattern ? could be utilized to match any character including a space.) Note that path names matching an exclusion pattern are not considered for any operation on the destination, including removal via the **--delete** option.

--group|-g
Preserve Unix group ID.

--help|-h
Show usage message.

--ignore-times|-I
Don't skip files where the source and destination size and time match.

--listen-address=*ADDRESS*
Local address for receiving truncated files. By default it is identified automatically with help of the host routing table.

--listen-ports=*PORT1,PORT2*
Pair of ports necessary for receiving truncated files. By default dynamic ports are used.

--marker=*FILE*
Location for event marker file.

--max-connections|-M *#*
Maximum number of network connections in use. Default is 8.

--mode|-m *MODE*
Replication mode, either 'copy' or 'link'. Copy mode always copies data; link mode uses hard links when possible.

--dry-run|-n
Generate the change manifest, but do not modify the destination.

--owner|-o
Preserve the Unix owner ID (super-user only).

--password|-P
Password for network replication.

--perms|-p
Preserve traditional Unix permissions.

--progress
Show progress while estimating and transferring data.

--recursive|-r
Recurse into directories.

--specials
Preserve Unix special files.

--no-ssl
Do not use SSL for network replication.

--sync={none|soft|hard}
Controls how FlexSync waits for modified files to be durable. If sync mode is 'hard', then each file is made durable individually using fsync(). Hard mode can significantly decrease replication performance. If sync mode is 'soft', then a single call to sync() is made after all I/O is completed. Soft mode has a much lower performance impact than hard mode, but is not effective in ensuring durability in all cases. If sync mode is 'none', then no sync calls are made. The default value is 'none'.

--times|-t
Preserve atime and mtime for non-directories.

--username|-U
Username for network replication.

--update|-u
Skip files that are newer on the destination.

--version|-V
Show the FlexSync software version and exit.

--verbose|-v
Show the paths being modified.

--threads|-w *#*
Set number of flexsync worker threads. Default is half the number of CPU cores, but at least one.

--http-threads|-W *#*
Set number of HTTP worker threads. Default is 8.

--xattrs|-X
Preserve extended attributes. When source and destination both use native Linux file systems such as ext4, enabling this option may preserve Posix ACLs as a side effect.

--one-file-system|-x
Don't cross filesystem boundaries.

--data-timeout *SECONDS*
Seconds before HTTP connections will time out after established. Default is 3600, or 1 hour.

--connect-timeout *SECONDS*
Seconds before HTTP connections will time out attempting to connect. Default is 60, or 1 minute.

--tsmdata-timeout *SECONDS*
The time in seconds, that flexsyncd waits for incoming requests of particular reqid from TSM. If there is no activity with this reqid during this time, the timeout error will be reported and no more files within this reqid will be retrieved. This prevents Flexsync from waiting files from media forever. Default is 3600 or 1 hour.

--tsm-req-files *#*
Set the max number of files in a TSM media retrieval request. Default is 10000.

--tsm-media-reqs *#*
Set the max number of inflight media in TSM media retrieval. 0 means no limit. Prior to FlexSync version 2.3.1, the default was 0. Starting with FlexSync version 2.3.1 the default is 3.

--tsm-obj-reqs *#*
Set the max number of inflight TSM media retrieval requests per object storage media. Default is 4.

--ignore-meta-errors *#*
Continue data replication, if attempt to set metadata on file failed. By default file replication failes on metadata error.

--nfs-web-user=*USER*
Set NFSV4 web server user name.

--nfs-web-password=*PASSWORD*
Set NFSV4 web server password.

--nfs-web-snap=*SNAPSHOTNAME*
Set NFSV4 web server previous snapshot name. If previous snapsjot name was not set, the ordinary local replication will be done and initial snapshot, with the name, corresponding to this run uuid will be created. This name can be used as SNAPSHOTNAME in next run.

EXAMPLES

flexsync -a /sourcedir /destdir

This will sync the contents of /sourcedir and all of its descendants to /destdir/sourcedir.

For files that already exist in the destination, **flexsync** will try to minimize the number of filesystem objects and the amount of data being synced by comparing sizes and timestamps. To override this optimization and resync everything, use the -I option, e.g.:

flexsync -a -I /sourcedir /destdir

flexsync -U admin -P password --nfs-web-user root --nfs-web-password password --nfs-web-snap 26102b97-8c36-40b1-af0c-0e22f50829db_snap_1652284556 -a /mfs_mnt/sourcedir /destdir

This will replicate from nfsv4 host mounted on /mfs_mnt, using provided username and password to communicate with web server, running on source host to replicate data, using nfsv4 changelist having previous snapshot with provided --nfs-web-snap name. See nfsv4 documentation for details, how to use snapshot and changelists.

SEE ALSO

flexsync(1) **flexsyncadmind**(8), **flexsyncadmin**(8), **flexsyncd**(8)

NAME

flexsyncadmin − Utility for FlexSync administration

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in both **sync** and **object** operations. Where needed distinctions between the two will be indicated.

SYNOPSIS

flexsyncadmin [**--username=***USERNAME* |-U *USERNAME***]** [**--password=***PASSWD* |-P *PASSWD***]**
[**--hostname=***HOSTNAME* |-H *HOSTNAME***]** [**--port=***PORT* | **-p** *PPRT*] [**--json** | **-j**]
[**--timeout=***SECONDS* | **-t** *SECONDS*] [**--version** | **-V**] [**--help** | **-h**] [**--debug=***[all|module,...]* | **-d**]
[**--credits**]

[**--change-policy**] [**--checksum <0|1>**] [**--dedup <0|1>**] [**--destructive**] [**--nobatch**]
[**--name <repository_name>**] [**--retention <permanent|none|days>**]
[**--s3-bucket <bucket_name>**] [**--s3-hostname <hostname>**] [**--s3-password <s3_password>**]
[**--s3-port <s3_endpoint_tcp_port>**] [**--s3-signing <v2|v4>**] [**--s3-username <username>**]
[**--ssl <0|1>**] [**--update** | **-u**] [**--use-existing**]

[**command**]

DESCRIPTION

flexsyncadmin can be used to create, configure, remove, and observe FlexSync hostgroups, hosts, and tasks.

For **sync** operations it can also be used to manage any versions created by FlexSync utilities including this one.

For **object** operations it also provides commands needed to manage object repositories.

OPTIONS (COMMON)

Options that start with -- and take an argument can have the argument separated by either a space or an equal sign, e.g.

--hostname *Hostname*
--hostname=*Hostname*

are equivalant.

--username|**-U** *USERNAME*
Username for flexsyncadmind(8). Created using flexsyncauth(8). If not specified as a command option, the user will be prompted.

--password|**-P** *PASSWD*
Password for flexsyncadmind(8). Created using flexsyncauth(8). If not specified as a command option, the user will be prompted.

--hostname|**-H** *HOSTNAME*
Hostname for web services.

--port|**-p** *Port*
Port for web services.

--version|**-V**
Show the flexsyncadmin software version and exit.

--help|**-h**
Show the flexsyncadmin help message.

-d Enable debug logging for all modules.

--debug=*[all|module,...]*
Enable debug logging for specified modules. Default is 'all'. Modules: 'flexsync', 'http'.

--credits
Print legal disclosures.

OPTIONS (SYNC OPERATIONS)

Options that start with -- and take an argument can have the argument separated by either a space or an equal sign, e.g.

--json|**-j**
Enable JSON output for certain flexsyncadmin operations.

--timeout|**-t** *SECONDS*
HTTPS operation timeout in seconds.

OPTIONS (OBJECT OPERATIONS)

--checksum *<0|1>*
Only valid with the **init** and **check** commands. When specified, this option turns file checksumming on or off. The default is on (1).

--dedup *<0|1>*
Only valid with the **init** command. When specified, this option turns file deduplication on or off. The default is on (1).

--destructive
Only valid with the **clean** command. When specified, cancels additional check for the "grace period". (One additional week of protection.)

--nobatch
Only valid with the **clean** command. The batch deletion of objects is used during cleaning to increase performance. This option means deleting objects one by one instead. It should be used, if s3 storage doesn't support batch deleting.

--name *repository name*
Only valid with the **init** and **delete** commands. When specified, this option specifies the name of the new repository.

--retention <*permanent|none|days*>
only valid with the **init** command. When specified, this option sets the retention period for old commit operations. (That is, for old/stale versions of files and directories.) The *permanent* argument means that old versions are never deleted. The *none* argument means that old versions are not kept. (Any *clean* operation done will clean those versions.) An integer number of days as an argument means old versions up to that age are kept. They will be cleaned after that time period by any clean operation. The default is permanent.

--s3-bucket <*s3 bucket name*>
Only valid with the **init** command. When specified, this option specifies the name of the S3 bucket in which to store objects.

--s3-hostname <*hostname*>
Only valid with the **init** command. When specified, this option specifies the S3 endpoint which **Flexsync** should talk to for storage and retrieval of objects.

--s3-password <*S3 password*>
Only valid with the **init** and **register** commands. When specified, this option specifies the password to present to the S3 endpoint.

--s3-port *<tcp port>*
Only valid with the **init** command. When specified, this option specifies which TCP port to use to talk to the S3 endpoint.

--s3-signing *<v2|v4>*
Only valid with the **init** command. When specified, this option specifies which type of signing to use on S3 requests. The default is v4.

--s3-username *<S3 username>*
Only valid with the **init** and **register** commands. When specified, this option specifies the user name to present to the S3 endpoint.

--ssl *<0|1>*
Only valid with the **init** command. When specified, this option specifies whether or not SSL should be used to encrypt and authenticate the connections to the S3 endpoint.

--use-existing
Only valid with the **init** command. When specified, this option causes the local **flexsyncadmind** to use an existing repository. It's useful during disaster recovery if a previous machine running **flexsyncadmind** was lost. It can also be used for a form of sharing repositories but ONLY if the configurations are to be left exactly the same. This option should be used with care.

--change-policy
Only valid with the **init** command. This is used for changing the following policy options: *retention*, *checksum*, and *dedup*.

COMMANDS (COMMON AND SYNC OPERATIONS)

When no command is given following the options, flexsyncadmin runs in interactive mode. Otherwise, flexsyncadmin executes the specified command and exits. Available commands are as follows:

help Show the available commands.

add-hostgroup *hostgroup_name*
Add a hostgroup with the given name.

edit-hostgroup *hostgroup_name* set name=*new_name*
Rename a hostgroup.

remove-hostgroup *hostgroup_name*
Remove the hostgroup with the given name. Note that any hosts in this hostgroup will remain, and are moved to the "Orphaned_Hosts" group.

add-host *hostgroup_name host_name DNS/IP username password* [port]
Add a host to *hostgroup_name*. The *host_name* will be used as a nickname during other commands. The host must be reachable at the given DNS name or IP address or the **add-host** operation will fail. *username* and *password* are mandatory.

edit-host *hostgroup_name host_name* [command]
When no command is given, and **flexsyncadmin** is running interactively, host editing mode is entered and changes can be made to the specified host. Otherwise, **flexsyncadmin** executes the specified **edit-host** subcommand and exits. See the **Task and Host Editing subcommands** section below.

remove-host *hostgroup_name host_name*
Removes *host_name* from hostgroup *hostgroup_name*.

restore-host *hostgroup_name host_name* [new_username new_password]
Restore *host_name* in hostgroup *hostgroup_name*. Source node access tokens for all tasks, whose source node is the *host_name*, are updated. If the new username and password are provided, the host credentials are also updated in flexsyncadmind db. This command is used to update the stale source node access tokens when the source node is completely recreated.

add-task *[--force] source_host_name dest_host_name task_name source_directory dest_directory*
Add a sync task named *task_name* to the host *dest_host_name* . The task will be created with a sync_schedule of '0' (on-demand) so that options can be configured with **edit-task**. If **--force** is specified, don't check if the destination is a mount point, never prompt.

add-default-task *[--force] source_host_name dest_host_name task_name source_directory dest_directory*
Similar to **add-task**, except that the newly created task will be scheduled to run every 15 minutes.

remove-task *[[[--dest] [--versions] [--audit]] | [--all]] [--DELETEMYDATA] dest_host_name task_name*
For sync tasks: Remove the given task from host *dest_host_name*. If **--dest** is specified, the destination directory tree will be removed. If **--versions** is specified, any versions that were created will be removed. If **--audit** is specified, any audit information will be removed. **--all** is the same as **--dest --versions --audit**. If any of **--dest**. **--versions**, **--audit**, or **--all** are specified, the task is not deleted, and all data is preserved unless the **--DELETEMYDATA** option is also specified. If no options are given, the destination directory, any version directories, and any audit trails will be preserved.
For object tasks: The options **--dest**, **--versions**, **--audit**, **--all**, and **--DELETEMYDATA** do not apply. Cleanup of the object repositories is done via the **clean** command.

edit-task *dest_host_name task_name* [command]
When no command is given, and **flexsyncadmin** is running interactively, task editing mode is entered and changes can be made to the specified task. Otherwise, **flexsyncadmin** executes the specified **edit-task** subcommand and exits. See the **Task and Host Editing subcommands** section below.

start-task *dest_host_name task_name*
If the given task is currently idle, or it has been marked as suspended, it will start right away, even if its syncing schedule has not yet reached. The task will no longer be marked as suspended. Note that tasks started using this command are exempt from the flexsyncd(8) maximum concurrent task limit, which can result in higher than normal system resource usage.

start-scan *dest_host_name task_name*
Identical to **start-task**, except that the resulting task run will scan the source, even if that task run normally would have been event driven.

suspend-task *hostgroup_name [dest_host_name [task_name]]*
Suspend a single task named *task_name*; or all tasks on *dest_host_name*; or all tasks in *hostgroup_name*.

resume-task *[--force] hostgroup_name [dest_host_name [task_name]]*
Resume a single task named *task_name*; or all tasks on *dest_host_name*; or all tasks in *hostgroup_name*. Similar to start-task, except that the task will not start until its syncing schedule has reached. If **--force** is specified, don't check if the destination is a mount point, never prompt.

show-task-reports *dest_host_name task_name*
Show task activity reports.

remove-task-report *dest_host_name task_name timestamp*
Remove the task report with the given timestamp, as shown by **show-task-reports**. For removing task sessions that created a version, use **remove-version** instead.

purge-task-reports *hostgroup_name [dest_host_name [task_name]]*
Remove task activity logs and audit logs for sessions that did not create a version.

status *[--active] [hostgroup_name [dest_host_name [task_name]]]*
Display status, optionally filtered by hostgroup, destination host, or task. If **--active** is specified, only show status information for tasks that are currently either estimating or replicating.

config-backup *[hostgroup_name [host_name]] backup_pathname*
Create a backup of all of the configuration information for this flexsyncadmind(8) and all of its flexsyncd(8) hosts and store it in *backup_pathname*. The backup file is suitable for use with the **config-restore** command. If a *hostgroup_name* is given, only back up that hostgroup. If a *hostgroup_name* and *host_name* are given, only back up that hostgroup and host.

config-restore *[--wipe] [hostgroup_name [host_name]] backup_pathname*
Restore the flexsyncadmind(8) and flexsyncd(8) configuration from *backup_pathname*. A partial restore can be done by specifying a *host_name* and/or a *hostgroup_name*. If **--wipe** is specified, all existing tasks will be removed including their destination content, versions if exist, audit and task config before restoring from the backup config. In case of conflicting host names, the host IP address must be used to restore. The host(s) will be completely replaced by host(s) definitions

from backup.

reset-factory-defaults *[--force]*

All existing hostgroups, hosts and tasks will be removed including their destination content, versions if exist, audit and task config. If **--force** is specified, command will be executed immediately, without additional confirmation.

show-email-reporting

Show the current email reporting configuration

set-email-reporting *[--verify] <enabled|disabled> [<smtp_hostname> <smtp_port> <tls|none> <none|username:password> <sender address> <recipient_address[,recipient_address]> 'cron expression'*

Set the email reporting configuration. Cron expression can be quartz style or traditional UNIX style. If **--verify** and **enabled** are specified, a test email will be sent to verify the email setting.

Quartz style cron expressions have a seconds field:
second minute hour day_of_month month day_of_week

Traditional UNIX style cron expressions do not have a seconds field:
minute hour day_of_month month day_of_week

Refer to the crontab(5) man page for details about how to write cron expressions.

config-reload

Reload flexsyncadmind license, product key and api-gateway configurations.

debug-snapshot *hostgroup_name dest_host_name*

Fetch a snapshot with the internal statistics and the state of some internal structures of flexsyncd from a host specified by *hostgroup_name* and *host_name*. It can be used to get some insight into memory resources in use and the state of interaction between flexsyncd and TSM.

source-12-10-2016-09:57:39 (Delete in progress)

COMMANDS (SYNC OPERATIONS)

show-versions *dest_host_name task_name*

List the versions created by FlexSync runs for this task.

remove-version *dest_host_name task_name version_name*

Initiate the removal of the directory tree for the given version. Choices for *version_name* can be obtained from the **show-versions** command. This command cannot be undone, so please use with caution. Versions which are in the process of being removed will be displayed by **show-versions** as (Delete in progress), e.g.:

reverse-task *dest_host_name task_name*

Create a task based on **task_name** such that the source and destination hosts and directories are reversed in a way that would restore the source directory onto the source host. The reversed task will be named **task_name-REVERSED**. The task to be reversed must be suspended before using this command. The **sync_schedule** of the reversed task will be set to '0', i.e. it requires a subsequent **start-task** to run.

clone-task *[--force] dest_host_name task_name new_task_name*

Create a clone of a task based on **task_name** named **new_task_name**. The **sync_schedule** of the new task will be set to '0', i.e. it requires a subsequent **start-task** to run. If **--force** is specified, don't check if the destination is a mount point, never prompt.

COMMANDS (OBJECT OPERATIONS)

Use **flexsyncadmin help** to see usage for commands.

Some commands below require a register URL. Here is a sample format:
https://s3host:s3port/my_s3_bucket/flexsync/repository_name

add-object-task *host_name object_repo task_name source_dir*

Add an object task named *task_name* to the host *host_name* . The task will be created with a sync_schedule of '0' (on-demand), options can be configured with **edit-task**.

add-default-object-task *source_host_name object_repo task_name source_dir*

Similar to **add-object-task**, except that the newly created task will be scheduled to run every 15 minutes.

check *reponame [--checksum <0|1>]*

This command causes **flexsyncadmin** to check the repository for correctness.

clean *reponame [--destructive][--nobatch]*

This command causes **flexsyncadmin** to delete the stale commits (old versions) from a repository. These old versions are what's left after file modifications, deletions, etc. The commits that caused these file versions to be put in the repository initially, are now stale. The stale objects that will be deleted/clean are: those which are referenced by commit with timestamp older than: now + "retention period" + "grace period" and a modification time on the server that is older, than "grace period"; or those not referenced by any commit at all (likely the result of failed commit). Note, that the "grace period" is one week. The 'retention period' depends on 'retention_mode', which is defined, when repo is created and can be:

- period - 'retention period' is defined in file policy in days.
- none - 'retention period' is 0
- permanent (default) - only clean of objects, not belonging to any commit is available.

The check for the "grace period" (additional 1 week) can be canceled by option **--destructive**. When it is provided, only "retention period" is considered in determining what is sufficiently stale for cleaning. The option **--nobatch** should be used if s3 storage doesn't support batch deletion.

delete *[--force] --name <repository_name>*

This command causes **flexsyncadmin** to delete a previously initialized repository. The repository to delete is indicated by the *--name* argument. Note that this command does not perform any cleaning/removal of the target repository that was initialized, it only removes local knowledge of that repository. If a repository cannot be removed because a host is unreachable, the *[--force]* option can be used to override that check. Note the the *[--force]* should not be used unless it is known that the host has no tasks for the repository.

Registrations and running **delete**:

- A repository cannot be deleted if there are current registrations for that repository.
- Use the **unregister** command to remove the existing registrations.
- The *[--force]* option has no affect on existing registrations.

init *--name <repository_name> --s3-hostname <s3 endpoint hostname> --s3-port <s3 tcp port> --ssl <0|1> --s3-bucket <s3 bucket name> --s3-username <s3 username> --s3-password <s3 password> --s3-signing <v2|v4> [--change-policy] [--retention <permanent|none|days>] [--checksum <0|1>] [--dedup <0|1>] [--use-existing]*

This command causes **flexsyncadmin** to initialize a new repository. *[--change-policy]* can be used with **init** command for changing the following policy options: *[--retention <permanent|none|days>] [--checksum <0|1>] [--dedup <0|1>]* Set new values for all policy options if you are using *[--change-policy]*. Otherwise non initialized policy options will have default values.

NOTE: If a bucket was previously initialized and then deleted, may need to use the *[--use-existing]* option. This would be the case if a message similar to "Failed to initialize repository: Repository named 'repos' already exists in bucket" is output.

register *<registerURL> --s3-username <s3 username> --s3-password <s3 password> [--register-user <user>] [--register-hostgroup <hostgroup>] [--register-host <host>]*

<registerURL> must contain the s3_signing. *Example: http://10.65.183.251:9000/mybucket/flexsync/repoName?signing=v2* Teach flexsync on a host within the hostgroup about a new repository.

repos *[--register-user <user>] [--register-hostgroup <hostgroup> [--register-host <host>]]*
This command causes **flexsyncadmin** to print out a listing of all the repositories it knows about. It shows the initialized and registered repositories and also provides some basic statistics on use of those repositories. Note that the **Files** column contains the counts of files and directories in the repository, not just files. No bytes however are added to the byte counts for the directories.

unregister *<repository_name> [--register-user <user>] | [--register-hostgroup <hostgroup> --register-host <host> --register-user <user>]*
Remove a previously registered repository from a host. NOTE: If a specific host is not specified, then the unregister applies to items on the local node only.

edit-repo-config *repository_name* [command]
When no command is given, and **flexsyncadmin** is running interactively, repository configuration editing mode is entered and changes can be made to the specified repository. Otherwise, **flexsyncadmin** executes the specified **edit-repo-config** subcommand and exits. See the **Repository Editing subcommands** section below. NOTE: Most likely if S3 information is changed for the repository, the same information should be changed for the repository registration using the **edit-register-config** option.

edit-register-config *hostgroup host repository_name user* [command]
When no command is given, and **flexsyncadmin** is running interactively, repository registration editing mode is entered and changes can be made to the specified registration. Otherwise, **flexsyncadmin** executes the specified **edit-register-config** subcommand and exits. See the **Repository Editing subcommands** section below. NOTE: Most likely if S3 information is changed for the registration, the same information should be changed for the repository using the **edit-repo-config** option. NOTE: If all registrations point to the same bucket and use the same S3 information, each individual registration will need to be updated for each host and user.

The following commands set, get or reset some options for network communication with s3 server. Further descriptions of these options are provided in individual man pages. See those for more details.

getconfig
Shows a matrix configuration key. Refer to **flexsyncadmin-getconfig (1)** for more info.

rstconfig
Reset a matrix's configuration key to its default value. Refer to **flexsyncadmin-rstconfig (1)** for more info.

setconfig
Set a matrix configuration key. Refer to **flexsyncadmin-setconfig (1)** for more info.

TASK AND HOST EDITING SUBCOMMANDS (COMMON OPERATIONS)

show-options
Show the current task or host options.

save Save changes to options. For task changes, the new values will take effect the next time the task runs. Host changes will take effect immediately. (Interactive mode only)

set *option_name=value [option_name=value ...]*
Set the given option(s) to the specified value(s). Note that in interactive mode, the **save** command is required to save any changes to options.

The following options can be set for both sync and object tasks:

cool_down:<number> wait specified seconds after last time stamp change, before replicating. The default for sync tasks is 0 seconds and for object tasks 180 seconds.

dry-run:<false|true> perform estimating dry run. The default is false.

exclude:[] list of file glob patterns to exclude. (Leading spaces, ending spaces, filenames with spaces, and/or directories with spaces are not allowed. The glob pattern ? could be utilized to match any character including a space.)

log_retention:<number> time interval in sec. to keep a task log (0=forever). The default is 30 days.

name:<string> name of the task

selfcheck_interval:<number> time interval in sec. to force full scan. The default is 7 days.

source:<directory path> source directory for task (NOTE: Applies only to sync tasks)
sync_schedule:<cron expr> cron expression (quartz or traditional unix style) for scheduling runs
xattrs:<true|false> preserve extended attributes. The default is true.

The following option is only valid for object tasks:

estimate:<false|true> perform estimating run before replication. The default is true.
conflict:<conflictAction> action to take if two or more commit operations occur simultaneously. The default is keep. Valid values:
discard throw away all local changes to the file and replace the file with the updated version from the repository.
keep ignore remote changes to the file and keep the local changes to the file.
rename-local rename the local file by adding ".local" to the end of its name, then download the remote version of the file and create it with its correct name.
rename-repo leave the local file alone and create a new file to contain the repo version. The new file's name ends in a period followed by the remote file's VersionID.
abort cause the whole update to fail. Everything will be left as it was before update was run.

show-defaults
Show the option values that would be set by **reset-defaults**.
reset-defaults
Reset this task or host's options to their default values.
quit Quit without saving changes. (Interactive mode only)

TASK AND HOST EDITING SUBCOMMANDS (SYNC OPERATIONS)

The following options can only be set for sync tasks:

acls:<true|false> preserve StorNext ACLs
archive:<true|false> preserve permissions, owner, times, group, devices, special files and recurse to directories
audit:<true|false> enable extended logging
devices:<true|false> preserve device files
fsid-check:<false|true> perform validation for saved file system identifiers
group:<true|false> preserve group
owner:<true|false> preserve owner
perms:<true|false> preserve permissions
times:<true|false> preserve atime and mtime
specials:<true|false> preserve special files (sockets, fifos and character or block devices)
delete:<true|false> delete destination files not found on the source
events:<true|false> perform event based scan
update:<false|true> skip files that are newer on the destination
ignore-times:<false|true> don't skip files where the source and destination size and time match, it triggers a full scan on managed fs
metadata-only:<false|true> only copy metadata
one-file-system:<false|true> when scanning, do not traverse file system mount points that are nested
recursive:<true|false> recurse into directories
ssl:<true|false> use SSL for network communication
shared-conn:<true|false> Indicates whether to share a pool of connections for replicating data or not.
Tasks set to true will share the same pool of network connections for replication.
This will result in tasks interleaving replication of files.
If one task has large files, it may result in other tasks being delayed

until
the large files are complete.
Tasks set to false will have their own pool of network connections for replication.
Depending upon the number of network connections for replication, a setting of false may saturate the network. This may result in individual
file replications with slower performance, but will result in tasks with the false setting replicating files in parallel.
Setting of true allows individual files more bandwidth, but causes tasks to serialize replication of files across tasks.
Refer to the **max-connections** setting for flexsyncadmind for the max number of connections per pool.
The default starting with Flexsync version 2.3.1 is false. (Prior releases it was true).

mode:<delta|copy|link> set copy mode

sync:<soft|hard|none> set synchronization style

tsm-req-files:<number> The max number of files in a TSM media retrieval request. The default is 10000.

tsm-media-reqs:<number> The max number of inflight media in TSM media retrieval. 0 means no limit.
The default starting with Flexsync Version 2.3.1 is 3.

tsm-obj-reqs:<number> The max number of inflight TSM media retrieval requests per object storage media. The default is 4.

version_interval:<number> minimum number of sec. between versions (0=disabled)

version_retention:<number> minimum period in sec. to keep a version (0=forever)

continue-on-meta-failure:<true|false> continue data replication, if attempt to set metadata on file failed. The default is false.

HOST OPTIONS

The following options can be set for host:

host_name:<string> the host ip address

admin_port:<number> admin port. The default is 10453

data_port:<number> data port. The default is 10454

data_ssl_port:<number> data ssl port. The default is 10554

username:<string> user name

password:<string> password

bwlimit:<number> bwlimit(0 means unlimited). The default is 0

hostgroup:<string> hostgroup name (the hostgroup must exist)

REPOSITORY EDITING SUBCOMMANDS

show-options
Show the current repository configuration options which are applied when **flexsyncadmind** accesses repository server. Or the current repository registration options which are applied when **flexsyncd** accesses repository server.

save Save changes to options. The changes will take effect immediately.

set *option_name=value [option_name=value ...]*
Set the given option(s) to the specified value(s). Note that in interactive mode, the **save** command is required to save any changes to options.

The following options can be set for repository configuration or registration:

s3-hostname:<string> the S3 endpoint which **Flexsync** should talk to for storage and retrieval of objects

s3-port:<number> which TCP port to use to talk to the S3 endpoint

ssl:<true|false> whether or not SSL should be used to encrypt and authenticate the con-

nections to the S3 endpoint
s3-signing:<string> which type of signing (v2 or v4) to use on S3 requests. The default is v4
s3-username:<string> the user name to present to the S3 endpoint
s3-password:<string> the password to present to the S3 endpoint
type:<string> repository type. It is readonly

show-defaults
Show the option values that would be set by **reset-defaults**.
reset-defaults
Reset this repository configuration or registration's options to their default values.
quit Quit without saving changes.

EXAMPLES

Set task1's sync schedule to sync in every 2 minutes:

```
rock # flexsyncadmin -U admin -P password
> edit-task destination-host task1
edit> set sync_schedule="0 0/2 * * * *"
edit> save
Changes will take effect when the task becomes idle.
edit>
```

Set task1's cooldown interval to five minutes (300 seconds) and automatically save the change:

```
rock # flexsyncadmin -U admin -P password edit-task destination-host task1 set co
```

Change a host's DNS name/IP address:

```
rock # flexsyncadmin -U admin -P password edit-host destination-host set hostname
```

Set an exclusion list of glob patterns for a task. Note the use of backslash single quotes when specifying multiple patterns:

```
rock # flexsyncadmin -U admin -P password edit-task destination-host task1 set ex
```

Note that path names matching an exclusion pattern are not considered for any operation on the destination, including removal via the **delete** option.

To clear a task's exclusion list:

```
rock # flexsyncadmin -U admin -P password edit-task destination-host task1 set ex
```

SEE ALSO

flexsyncadmin-rstconfig(1), **flexsyncadmin-setconfig**(1), **flexsyncadmin-getconfig**(1), **flexsyncadmind**(8), **flexsyncauth**(8), **flexsyncd**(8), **flexsync**(1), **flexsync**(8)

NAME

flexsyncadmind − FlexSync administration daemon

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in both **sync** and **object** operations. Where needed distinctions between the two will be indicated.

SYNOPSIS

flexsyncadmind [**--interval=***msecs*|**-I** *msecs*] [**--port=***Port*|**-p** *Port*] [**--pidfile=***FILE*|**-P** *FILE*] [**--foreground**|**-f**] [**--version**|**-V**] [**--credits**] [**--help**|**-h**] [**--debug=***[all|module,...]*|**-d**] [**--data-timeout=***SECONDS*] [**--connect-timeout=***SECONDS*]

DESCRIPTION

flexsyncadmind is the administrative daemon and web server for FlexSync, providing hostgroup and host configuration and management. It accumulates status information about hosts and tasks by polling **flexsyncd(8)** on each of the configured hosts. It also provides authentication and session management.

OPTIONS

Options that start with -- and take an argument can have the argument separated by either a space or an equal sign, e.g.

--port *Port*
--port=*Port*

are equivalant.

--interval|**-I** *msecs*
Interval at which flexsyncadmind polls each host.

--port|**-p** *Port*
Port for web services.

--pidfile|**-P** *FILE*
PID file to use for locking.

--foreground|**-f**
Run flexsyncadmind in the foreground instead of as a daemon.

--version|**-V**
Show the flexsyncadmind software version and exit.

--credits
Print legal disclosures.

--help|**-h**
Show the flexsyncadmind help message.

-d Enable debug logging for all modules.

--debug=*[all|module,...]*
Enable debug logging for specified modules. Default is 'all'. Modules: 'flexsync', 'http'.

--data-timeout *SECONDS*
Seconds before HTTP connections will time out after established. Default is 3600, or 1 hour.

--connect-timeout *SECONDS*
Seconds before HTTP connections will time out attempting to connect. Default is 60, or 1 minute.

LOGGING (OBJECT OPERATIONS)

Flexsyncadmind logs messages related to **object** operations using syslog and placed in the /var/log/flexsync directory.

The amount of **object** logging can be controlled via the **debug_level** parameter in config file: /opt/quantum/flexsync/etc/flexsyncadmind.conf A value of 0 (the default) means that only important error messages are logged. Larger values (up to 3) cause more information to be logged.

If the debug_level value is changed in /opt/quantum/flexsync/etc/flexsyncadmind.conf, **flexsyncadmind** must be restarted.

SEE ALSO

flexsyncd(8), **flexsyncadmin**(8), **flexsync**(1), **flexsync**(8)

NAME

flexsyncauth − FlexSync user administration command

SYNOPSIS

flexsyncauth −a −d <*flexsyncd*|*flexsyncadmind*> [**--db-path=***PATH*|**-b** *PATH*] −U <*username*> −P <*password*>

flexsyncauth −r −d <*flexsyncd*|*flexsyncadmind*> [**--db-path=***PATH*|**-b** *PATH*] −U <*username*>

flexsyncauth −l −d <*flexsyncd*|*flexsyncadmind*> [**--db-path=***PATH*|**-b** *PATH*]

[**--version**|**-V**]

DESCRIPTION

flexsyncauth is used to manage authentication with the **flexsyncd(8)**, **flexsyncadmind(8)** daemons and **NFSV4 web server**. Before **flexsyncadmin(8)** or the GUI can communicate with flexsyncadmind, a username and password must be created for flexsyncadmind. Similarly, before flexsyncadmind can communicate with flexsyncd, a username and password must be created for each instance of flexsyncd.

When using *NFSV4* file system as a source, if there is web server, running on source host, the real source hostname(IP), username and password should be provided to enable communication with web server. Otherwise, the ordinary local replication will be performed, and this will reduce an efficency of replication.

OPTIONS

-a Add a username/password to be used with the local instance of the specified daemon. This option can also be used to change the password for an existing username.

For NFSV4: adds web server authentication for provided hostname. Optionally, the port to communicate with web server is set. The default port for NFSV4 web server is 8080.

-r Remove a username/password pair from the local instance of the specified daemon.

For NFSV4: removes the authentication entry, corresponding to hostname.

-l List the users associated with the local instance of the specified daemon. *For NFSV4:* lists the nfs web servers names.

--version|**-V**
Show the flexsyncauth software version and exit.

-d The daemon being acted on by the **flexsyncauth** command. Valid options are *flexsyncadmind* or *flexsyncd*.

-U <*username*>
If **-a** option is specified, the username to be used to access the specified daemon.If **-r** is specified, the username/password pair to be removed from the specified daemon.

-P <*password*>
The password to be used to access the specified dameon.

SEE ALSO

flexsyncadmind(8), **flexsyncadmin**(8), **flexsync**(8), **flexsync**(1)

NAME

flexsyncdump − FlexSync database debug command

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in both **sync** and **object** operations. Where needed distinctions between the two will be indicated.

SYNOPSIS

flexsyncdump −l <**−d** <*flexsyncd*|*flexsyncadmind*> | <**--path=***PATH*|**-p** *PATH*>>

flexsyncdump −s <*key*> <**−d** <*flexsyncd*|*flexsyncadmind*> | <**--path=***PATH*|**-p** *PATH*>>

[**--version**|**-V**]

DESCRIPTION

flexsyncdump is used to print contents of the FlexSync configuration database. The contents of the database are referred to as configuration documents. This command is intended only for debugging purposes.

OPTIONS

-d <flexsyncd|flexsyncadmind>

If specified, the database path will be set to the default path for the indicated daemon

--path|**-p** *PATH*

An explicit path to a database can be provided, eg. one from a flexsync_gather. Another example, in this case for the **flexsync object** database, is /opt/quantum/flexsync/var/flexsyncregister.dat

-l List all the document keys in the database

-s <key>

Print a document given its document key (the key is from the "**-l**" listing)

--version|**-V**

Show the flexsyncdump software version and exit.

NOTE

The database for the **flexsync object** portion of the product cannot be viewed using the **-d** option. There is no corresponding daemon. It can only be viewed via the **--path**|**-p** with a path of /opt/quantum/flexsync/var/flexsyncregister.dat

SEE ALSO

flexsyncadmind(8), **flexsyncadmin**(8), **flexsync**(8), **flexsync**(1)

NAME

flexsyncd − FlexSync configuration and data moving daemon

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in both **sync** and **object** operations. Where needed distinctions between the two will be indicated.

SYNOPSIS

flexsyncd [**--adminport=***Port*|**-a** *Port*] [**--dataport=***Port*|**-p** *Port*] [**--datasslport=***Port*|**-s** *Port*] [**--data-timeout=***SECONDS*] [**--connect-timeout=***SECONDS*] [**--max-concurrent-tasks=***NUM*|**-c***NUM***]** [**--max-connections=***NUM*|**-M***NUM***]** [**--threads=***NUM*|**-w***NUM***]** [**--http-threads=***NUM*|**-W***NUM***]** [**--httpd-threads=***NUM*|**-S***NUM***]** [**--foreground**|**-f**] [**--trace=***FILE*|**-t***FILE***]** [**--fsid-timer=***SECONDS*] [**--version**|**-V**] [**--credits**] [**--help**|**-h**] [**--debug=***[all|module,...]*|**-d**]

DESCRIPTION

flexsyncd is the management daemon and data mover for automated FlexSync tasks that have been configured using the GUI or **flexsyncadmin(8)**. **flexsyncd** also handles version management and acts as a server for obtaining task logs, audit logs, and task status.

OPTIONS

Options that start with -- and take an argument can have the argument separated by either a space or an equal sign, e.g.

--port *Port*
--port=*Port*

are equivalant.

--adminport|**-a** *Port*
Port for web services administration traffic. Default is 10453.

--dataport|**-p** *Port*
Port for data traffic. Default is 10454.

--data-timeout *SECONDS*
Seconds before HTTP connections will time out after established. Default is 3600, or 1 hour.

--connect-timeout *SECONDS*
Seconds before HTTP connections will time out attempting to connect. Default is 60, or 1 minute.

--tsmdata-timeout *SECONDS*
The time in seconds, that flexsyncd waits for incoming requests of particular reqid from TSM. If there is no activity with this reqid during this time, the timeout error will be reported and no more files within this reqid will be retrieved. This prevents Flexsync from waiting files from media forever. Default is 3600 or 1 hour.

--datasslport|**-s** *Port*
Port for data traffic. Default is 10554.

--max-concurrent-tasks|**-c** *#tasks*
Maximum number of concurrently running tasks.

--max-connections|**-M** *#connections*
Maximum number of network connections in use per task pool. The number of task pools is based upon shared-conn setting for tasks. All tasks that have shared-conn set to true share a single pool

of up to max-connections. All tasks that have shared-conn set to false will have its own pool which will have up to max-connections per pool. Default is 8.

--threads|**-w** *#threads*
Number of flexsyncd worker threads. Default is half number of CPU cores, but at least one.

--http-threads|**-W** *#http-threads*
Number of HTTP (client) worker threads. Default is 8.

--httpd-threads|**-S** *#httpd-threads*
Number of HTTPD (server) worker threads. Default is half number of CPU cores, but at least one.

--foreground|**-f**
Run flexsyncd in the foreground instead of as a daemon.

--trace|**-t** *FILE*
Trace JSON exchanges and output to *FILE*.

--fsid-timer=*SECONDS*
Maximum number of *SECONDS* a task is suspended on fsid mismatch. The *SECONDS* argument is a value which should be greater than 0. If not specified, the default is 3600 *SECONDS*. When a fsid mismatch first occurs, the task is put to suspend state for a duration of the lessor of this max fsid timer value and base timer value of 300 *SECONDS*. Later when the task is back running, if the fsid validation still fails, every time the suspend duration is increased by 300 *SECONDS* until it reaches to the max fsid timer value.

--version|**-V**
Show the flexsyncd software version and exit.

--credits
Print legal disclosures.

--help|**-h**
Show the flexsyncd help message.

-d Enable debug logging for all modules.

--debug=*[all|module,...]*
Enable debug logging for specified modules. Default is 'all'. Modules: 'flexsync', 'http', 'task-report'.

SEE ALSO

flexsyncadmind(8), **flexsyncadmin**(8), **flexsyncauth**(8), **flexsync**(8), **flexsync**(1)

NAME

flexsync_gather − Collect debugging information for FlexSync

SYNOPSIS

flexsync_gather [**-d** DIRPATH][**-b**]

DESCRIPTION

The **flexsync_gather** program is used to collect FlexSync debugging information from a single host. In a typical multi-host configuration, **flexsync_gather** will need to be run on each host.

The debugging information is staged in a directory under /tmp. If TMPDIR is set to a directory, that directory will be used instead. See the command output for the location of the final tar ball.

OPTIONS

Option -d allows to specify a custom location of staging area and destination location for the output file. If this option is provided and specified directory does not exist, it will be created and used. If specified directory can't be created or used, default TMPDIR location will be used instead.

Option -b disables gathering into output tar ball file all flexsync executable binaries files and libraries, they depend on. It will reduce output file size.

SEE ALSO

flexsyncadmind(8), **flexsyncadmin**(8), **flexsyncauth**(8), **flexsync**(8), **flexsync**(1)

NAME

flexsync − FlexSync user CLI tool

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations. The man page for the **sync** operations is the flexsync(8) man page.

SYNOPSIS

flexsync <*command*> [*options*]

DESCRIPTION

Designed to support data intensive work-flows, flexsync is a simple, but powerful user interface built to take full advantage of S3 storage from Amazon Web Services (AWS S3) to easily create one or many shareable work-spaces. FlexSync uses AWS S3 to move content from their local system to a remote work-space.

FlexSync work-spaces, also known as repositories, are simple to create and manage. Whether you're in the same location or at a distant site, use the FlexSync client to browse and download content from the remote work-space to your local system. Updated and newly created content can be moved (committed) to the workspace, allowing other users to access the content for continued collaboration. No matter where you might be located, FlexSync allows your to share projects or vast archives. As all of the content within the repository are S3 objects, the content is immutable (unchangeable). You can create and use as many repositories, and as many work-spaces as needed.

The **flexsync** binary is the Command Line Interface tool. You can issue commands to checkout, update, and commit to a S3-backed repository using the same paradigm as a typical version control system. It is also possible to have FlexSync set up an automated task for moving data between the local system and the S3 repository. See the **flexsyncadmin** man page for the **add-object-task** function.

COMMON OPTIONS

This shows the options that are common across all flexsync object operations.

--debug|**-d** <*DebugLevel*>
: Set the debug level. A value greater than zero will result in lots of text being printed to the screen and written to the log file.

--help|**-h**
: Show usage message.

--version|**-V**
: Show the FlexSync software version and exit.

--credits
: Print legal disclosures.

FLEXSYNC COMMANDS

The **flexsync** command provides all functionality for managing a local copy of a repository. The CLI takes a command indicating which particular function to perform. A brief summary of the available commands is below. For more information, see the man page for each command.

In the **flexsync** man pages, a command is usually described with a dash between the "flexsync" part and the command name part. When actually using the tool, there is no dash. For example, you would find out how to use the "flexsync log" command by looking at the "flexsync-log" man page.

COMMANDS

checkout
: Download part or all of the contents of an existing repository. The contents are used to create a working directory that can be updated, changed, and committed.

commit
: Upload the changes in a working directory to the repository.

copy Copy one part of a repository to another directory without downloading and re-uploading all the data.

diff Find differences between the repository and a working directory.

dump Download and print out a JSON object from the repository. Used for debugging.

get Download a single file from the repository

help Print out help information about a command.

info Print information about a file or directory in a working directory.

list List directories in a repository.

log Describe the change history of a repository.

mkdir Make a directory in the repository

revert Discard changes to a working directory.

update Download updates that have been applied to the repository by others

EXIT VALUES

flexsync will return 0 on success and non-zero on failure.

NOTES

The system's clock needs to match the clock kept by the S3 back-end. If the clocks differ by a large enough value, FlexSync's S3 requests will fail. Running NTP (or equivalent) is recommended.

Xattr support varies by OS and file system. FlexSync tries to deal with xattrs in the best way possible. It's not always perfect.

EXAMPLES

First, a FlexSync repository must be registered with your system using the flexsyncadmin(8) command. Your administrator will need the appropriate credentials and URL for the repository.

After it has been registered, then check out the repository.

```
# flexsync checkout flexsync://myrepository workdir-1
Initializing...
Scanning...            3791 files, 10.82 MiB bytes to be downloaded
Operation completed successfully.
# ls -la workdir-1/
total 0
drwx------ 5 root root 6 Jan 12 15:47 ./
drwxrwxrwx 4 root root 6 Jan 12 15:47 ../
drwx------ 2 root root 1 Jan 12 15:47 .flexsync/
drwx------ 3 root root 2 Jan 12 15:47 dir-1/
drwx------ 3 root root 2 Jan 12 15:47 dir-2/
```

After you've made some changes, you can share them with others by committing them.

```
# flexsync commit -c 'Add some new files'
Initializing...
Filtering...           metadata archive unavailable
Scanning...            588 files, 164.28 MiB to be committed
Preprocessing...       complete
```

```
Uploading...           16.44 KiB deduplicated, 164.27 MiB uploaded
Postprocessing...      complete
Complete.
```

You can see the changes others have made by doing an update.

```
# flexsync update
Initializing...
Filtering...           metadata archive unavailable
Scanning...            complete
Reconciling...         123 files, 102 GiB bytes to be downloaded
Complete.
```

SEE ALSO

flexsync-checkout(1) **flexsync-commit**(1) **flexsync-copy**(1) **flexsync-diff**(1) **flexsync-dump**(1) **flexsync-get**(1) **flexsync-help**(1) **flexsync-info**(1) **flexsync-list**(1) **flexsync-log**(1) **flexsync-mkdir**(1) **flexsync-revert**(1) **flexsync-update**(1) **flexsync**(8) **flexsyncadmin**(8)

NAME

flexsync checkout − Check out a repository

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsync checkout *[options] <URI> [workdir]*

DESCRIPTION

The **checkout** command downloads part or all of a FlexSync repository and creates a working directory on the local file system to contain it. That working directory can then be used in subsequent FlexSync commands.

The FlexSync repository must have already been initialized and registered using the *flexsyncadmin register* command.

The URI is made of the form: *flexsync://reponame/path*. The *reponame* is just the text name given to the repository when it was initialized. It's typically the last part of the URL used to register the repository (with the **flexsyncadmin register** command). The *path* part is an optional path within the repository. It's a series of path name components separated by slashes.

NOTE: Providing the flexsync:// prefix in the URI is optional for this command.

If a path is provided with the URI, the path must correspond to a directory within the repository. If it doesn't, an error will be generated. NOTE: To extract just one individual file from a repository, see the **get** command.

The URI doesn't necessarily need to correspond to the root directory of the repository. In other words, it's possible (and in fact, common) to only check out part of a repository. A repository may contain many millions of files and many terabytes of information. A user may only want to deal with part of it and so only check out directories relevant to them.

The working directory is created with a name specified by the *workdir* argument. If no *workdir* argument is passed, the last path component in the URI is used as the name.

OPTIONS

--undo|**-f**

This option will cleanup flexsync workspace information in a directory where a prior checkout was performed. Only flexsync workspace information created by flexsync is removed. User created files are not removed. Data that may have been added to the S3 bucket is also not removed. Prior to running this operation, please ensure that no automated tasks exist for this directory.

--force|**-f**

This option allows a checkout to proceed to an already existing directory. Note that the existing directory cannot be a directory that was previously checked out.

--revision|**-r** <*CommitID*>

Specifying a particular revision to be checked out allows the user to create a working directory based on an old version of the repository. Use the **log** command to find a *CommitID* to supply to this command.

--debug|**-d** <*DebugLevel*>

Set the debug level. A value greater than zero will result in lots of text being printed to the screen and written to the log file.

--help|**-h**
Show usage message.

--version|**-V**
Show the FlexSync software version and exit.

--credits
Print legal disclosures.

EXIT VALUES

flexsync will return 0 on success and non-zero on failure.

NOTES

Even though the command is called checkout, there is no corresponding checkin command. The **commit** command is used to push updates to the repository. The **update** command is used to pull updates from the repository. A second checkout to a previously checked out directory, or any where beneath that directory, is not allowed.

When doing a checkout and providing an old revision, it is possible that missing content may be encountered. (Missing content is typically seen for files that existed in an older revision, however they have since been deleted and cleaned and are no longer in the repository.) When this content is encountered, files that are missing are logged to: /var/log/flexsync/flexsync.cli.log This is an example of the log entry that is generated for a missing file:

```
Info (checkout): File object missing from the cloud bucket, unable to retrieve: s
```

These log messages are informational only and all previous content that can be checked out will be.

EXAMPLES

Check out a whole repository named "images" into a working directory named "images".

```
flexsync checkout flexsync://images
```

Check out a subdirectory contained in the repository named "video" into a working directory named "evidence".

```
flexsync checkout flexsync://video/camera51/Sept/13 evidence
```

Check out part of a repository at a specific revision.

```
flexsync checkout -r 6ced6bfe-a71f-4fcd-ada4-a8fabde69458-commit \
flexsync://audio/music old_playlist
```

SEE ALSO

flexsync(1) **flexsync-get**(1) **flexsync-log**(1) **flexsync-commit**(1) **flexsync-update**(1)

NAME

flexsync commit − Commit changes to a repository

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsync commit [*options*] *[workdir]*

DESCRIPTION

The **commit** command uploads a working directory's local changes to the remote repository. Those changes are grouped together into a single **commit**. A commit can have a **commit message** which describes the changes in a human-understandable way.

If the user does not provide the **-c** argument to specify a commit message, this command will start an editor so the user can type one in.

Each commit is atomic. All the changes in the working directory are applied to the repository or none are. Each commit happens in a strict order. Each commit happens entirely after the one before.

Once the commit has been performed, other users can download the changes by doing a **update** in their own working directories. Alternatively, they can do a **checkout** of the repository to create a new working directory.

A user can see the history of the various commits that have happened to the repository by using the **log** command.

The user can see what changes would be made by running **commit** using the **diff** command.

OPTIONS

--comment|**-c** <*CommitText*>
: Text to describe the changes made in the commit.

--full-scan|**-S**
: Enforce full scan of the working directory.

--debug|**-d** <*DebugLevel*>
: Set the debug level. A value greater than zero will result in lots of text being printed to the screen and written to the log file.

--help|**-h**
: Show usage message.

--version|**-V**
: Show the FlexSync software version and exit.

--credits
: Print legal disclosures.

EXIT VALUES

flexsync will return 0 on success and non-zero on failure.

NOTES

The user should not modify the contents of a working directory while a commit is in progress. If a modification occurs, it's possible that the commit in the repository will contain some, all, or none of the files in flux at the time of commit.

When a commit is run content in the checkout directory is compared against content in the repository.

Changes found in the checkout directory are propogated to the repository. For files that exist in the checkout directory, and also in the repository, the files are checked for changes to see if a new version needs to be pushed to the repository. When this check is made, the content of the files are not compared, only three criteria: the modtime, size and whether or not the file is executable. These criteria are compared between the two locations. If there is a file which differs in content from the repository, but the three criteria match, then those changes are not pushed to the repository. If you find yourself in this situation, a simple touch of the file will cause the file to be pushed to the repository at the next commit operation.

COMMIT STATS

As part of the commit operation "stats" for repositories are updated as new content is added to or removed from a repository. The updated stat values are the files and bytes stored in the repository. (These stats can be viewed via the *repos* subcommand of the **flexsyncadmin** command.) There are three categories of content (bytes/files) in the repositories that are tracked: Active, Inactive and Total.

For files that have been added the Active and Total counts are increased in the repositories at commit time. These additions are reflected in the Active/Total byte/file counts in the *repos* report. For files that have been removed those stats are also updated at the time of the commit. The active counts are decreased and the inactive counts are correspondingly increased. The Total byte/file counts reported are not affected by a remove and a commit. Those Total stats are updated for a remove at the time of the *clean* subcommand of the **flexsyncadmin** command. It is at clean time that files are permanently removed from the repository and the Total values are decreased. The Inactive stats are also correspondingly decreased at clean time in addition to the Total values.

SEE ALSO

flexsync(1) **flexsync-log**(1) **flexsync-diff**(1) **flexsync-update**(1) **flexsyncadmin**(8)

NAME

flexsync copy − Copy files to a different location in a repository

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsync copy [*options*] <*URI*> <*URI*>

DESCRIPTION

The **copy** command makes a copy of a file or directory inside a **FlexSync** repository.

The command takes two URIs that point to contents in the same repository. You can use **info** to find the URIs for files in a working directory. In general, however, a working directory is not required.

NOTE: Providing the flexsync:// prefix in the URI is optional for this command.

The data does not move between the S3 bucket and the **FlexSync** client. In fact, no new space for the copied data is required. The copy process just creates new links to the existing objects by uploading a small amount of metadata to the bucket.

OPTIONS

--comment|**-c** <*CommitText*>
: Text to place in the commit created by the copy.

--force|**-f**
: This option allows a copy to overwrite an already existing file.

  NOTE: When a force is done on a directory copy, as with files, the operation is a COPY/REPLACE. The entire destination directory is recursively replaced with a copy of the source directory. None of the original destination directory contents will be left when done. (The **revert** command can be used to back out the copy operation if needed.)

--revision|**-r** <*CommitID*>
: Specifying a particular revision allows the user to copy a file or directory out of the history of the repository into the current version of the repository. The *CommitID* is used to qualify the first URI passed into the command. Use the **log** command to find a *CommitID* to supply to this command.

--debug|**-d** <*DebugLevel*>
: Set the debug level. A value greater than zero will result in lots of text being printed to the screen and written to the log file.

--help|**-h**
: Show usage message.

--version|**-V**
: Show the FlexSync software version and exit.

--credits
: Print legal disclosures.

EXIT VALUES

flexsync will return 0 on success and non-zero on failure.

EXAMPLES

To copy a file from one place to another.

```
# flexsync copy -c 'making a copy of filename1' \
```

```
flexsync://repo/path/to/filename1 \
flexsync://repo/other/path/to/filename2
```

To copy a file from a different revision.

Obtain the revision by using the **log** command. (The revision is the "Commit:" line.)

```
# flexsync log flexsync://repo/path/file1

    ------------------------------------------------------------
    Commit:        64e83a5b-1217-41f7-95e2-0a579c2d75be-commit
    Date:          2022-05-19 16:56:26
    Username:      root
    Comment:       Second change

    ------------------------------------------------------------
    Commit:        573f4f34-3adf-4898-adc5-881f09a6045d-commit
    Date:          2022-05-13 13:32:34
    Username:      root
    Comment:       initial commit
```

Using the "Commit:" output from the initial commit, request a copy of file1.

```
# flexsync copy -c ’copy a specific revision" \
--revision 573f4f34-3adf-4898-adc5-881f09a6045d-commit \
flexsync://repo/path/file1 \
flexsync://repo/path/file1_copy
```

SEE ALSO

flexsync(1) **flexsync-info**(1)

NAME

flexsync dump − Dump out a metadata object from a repository

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsync dump [*options*] <*reponame*> <*objectname*>

DESCRIPTION

The **dump** command communicates with the repository to retrieve a metadata object. When user data is committed to a repository, other objects containing metadata are also placed in the repository with the user data. This command allows for the reporting of the contents of the metadata objects.

The <objectname> parameter can be any one of these: head, policy, flexsync_info, <uuid>-commit and <uuid>-node.

A general desciption of the items which can be reported: The head item contains summary info on the overall repository including current state. The policy item contains initialization parameters for the repository. The flexsync_info item contains product info on **FlexSync**. The <uuid>-commit items are reported via the **log** command when looking at the history for a file or directory. The report shows commits that were done affecting the file or directory. NOTE that the <uuid>-commit string is also known as the **CommitId**. The <uuid>-node items are reported when examining the commits via **dump**. They show up with the JSON tag "version_id" and contain content descriptions of a commit.

This command is mostly useful for debugging of **FlexSync** itself and is not recommended for general use. There are more user friendly ways of checking repository content rather than via **dump**. See **list** and **get** for ways to get at user data without performing a checkout.

OPTIONS

--debug|**-d** <*DebugLevel*>
: Set the debug level. A value greater than zero will result in lots of text being printed to the screen and written to the log file.

--help|**-h**
: Show usage message.

--version|**-V**
: Show the FlexSync software version and exit.

--credits
: Print legal disclosures.

EXIT VALUES

flexsync will return 0 on success and non-zero on failure.

EXAMPLE

To dump the head object:

```
flexsync dump reponame head
```

SEE ALSO

flexsync(1) **flexsync-get**(1) **flexsync-list**(1) **flexsync-log**(1)

NAME

flexsync get − Get a single file from a repository

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsync get [*options*] <*URI*> *[filename]*

DESCRIPTION

The **get** command retrieves a single file from a **FlexSync** repository. The file to be retrieved is specified by the URI argument.

NOTE: Providing the flexsync:// prefix in the URI is optional for this command.

The retrieved file is placed in the current directory and given the same name as it has in the repository. This can be overridden by passing in a path name after the URI argument.

This command is mainly meant to get a file from a repository where no checkout has been done. It allows for retrieving data from the repository for use, where there are no plans to make changes that need to be committed. It can however also be used in a checked out directory for retrieving data. In that case any changes would be committed with the next commit operation.

OPTIONS

--force|**-f**
: This option will cause **get** to overwrite a file in the local file system if it exists.

--revision|**-r** <*CommitID*>
: Specifying a particular revision allows the user to copy a file out of the history of the repository. The URI passed in will be looked up in the repository as it was at the commit specified by *CommitID*. Use the **log** command to find a *CommitID* to supply to this command.

--debug|**-d** <*DebugLevel*>
: Set the debug level. A value greater than zero will result in lots of text being printed to the screen and written to the log file.

--help|**-h**
: Show usage message.

--version|**-V**
: Show the FlexSync software version and exit.

--credits
: Print legal disclosures.

EXIT VALUES

flexsync will return 0 on success and non-zero on failure.

EXAMPLES

This command gets a file from a repository and places it in the current directory of the local file system with the same name as it has in the repository. In this case it would get a file named "testfile" and place it in the current directory.

```
flexsync get flexsync://reponame/path/to/testfile
```

NOTES

Two types of common get failures are:

```
Error: Failed to get subxx/file.05: No such path in repository
```

And:

```
Error: File object not found in the cloud bucket, unable to get: flexsync://REPO/
```

The first error is what is reported if the file requested does not exist in the repository. (A typing error etc.) The second error is only seen when trying a get operation on an old revision of a repository. If the file existed previously, but has since been removed and cleaned, then the second error is reported.

SEE ALSO

flexsync(1) **flexsync-log**(1)

NAME

flexsync help − Print help information about the object commands

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsync help *[<command>]*

DESCRIPTION

The **help** command prints out command line help about the object commands. Run **man flexsync-<cmd>** for further info about those commands.

EXIT VALUES

flexsync will return 0 on success and non-zero on failure.

SEE ALSO

flexsync(1)

NAME

flexsync info − Print information about a file

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsync info [*options*] *[path_in_workdir]*

DESCRIPTION

The **info** command prints out information about a given path within a working directory. The command output includes the URI to the path and the *CommitID* of the last commit which changed the path.

OPTIONS

--debug|**-d** <*DebugLevel*>

Set the debug level. A value greater than zero will result in lots of text being printed to the screen and written to the log file.

--help|**-h**

Show usage message.

--version|**-V**

Show the FlexSync software version and exit.

--credits

Print legal disclosures.

EXIT VALUES

flexsync will return 0 on success and non-zero on failure.

EXAMPLES

Run flexsync info on a repository subdirectory.

```
flexsync info ./awsdir
Path:   flexsync://reponame/awsdir
Commit: 0fa39360-4ca6-4c29-8508-49c7c39a99eb-commit
```

SEE ALSO

flexsync(1)

NAME

flexsync list − List the contents of a repository, contents of a directory within that repository, or details about a specific file within that repository

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsync list|ls [*options*] <*URI*>

DESCRIPTION

The **list** command will list the directory structure of a repository without performing a check-out. The **list** command can also list the directory structure of a specific directory within the repository without performing a check-out. The **list** command can also be requested on a specific file within the repository without performing a check-out.

OPTIONS

--recursive|**-R**
: This option will cause **list** to recurse through directories.

--revision|**-r** <*CommitID*>
: Specifying a particular revision allows the user to list the directory structure of an older version of the repository. The URI passed in will be looked up in the repository as it was at the commit specified by *CommitID*. Use the **log** command to find a *CommitID* to supply to this command.

--verbose|**-v**
: This option will cause **list** to print modification time and size information for each file or directory found.

--debug|**-d** <*DebugLevel*>
: Set the debug level. A value greater than zero will result in lots of text being printed to the screen and written to the log file.

--help|**-h**
: Show usage message.

--version|**-V**
: Show the FlexSync software version and exit.

--credits
: Print legal disclosures.

EXIT VALUES

flexsync will return 0 on success and non-zero on failure.

EXAMPLES

List the entire current contents of the repository named "images", include sizes and times.

```
flexsync list -vR flexsync://images
```

List the entire current contents of a directory "dira" within repository named "images", include sizes and times.

```
flexsync list -vR flexsync://images/dira
```

List the details about a file "file1" within repository named "images", include sizes and times.

```
flexsync list -vR flexsync://images/dira/file1
```

NOTE: Providing the flexsync:// prefix in the URI is optional for this command.

SEE ALSO

flexsync(1) **flexsync-log**(1)

NAME

flexsync log − Print the version history of a repository

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsync log [*options*] *[URI | path_in_workdir]*

DESCRIPTION

The **log** command describes the history of the repository. The history is made up of a series of commits. The information for each commit includes the CommitID, the time of the commit, the name of the entity that made the commit, and the commit message specified at commit time. If the **-v** option is present, the list of file changes for the commit are included (in a format similar to that of **diff**).

The argument can be either a **FlexSync** URI or a path within a working directory. If no argument is given, the current directory is used.

NOTE: Providing the flexsync:// prefix in the URI is required for this command.

OPTIONS

--revision|**-r** <*CommitID*>
: Start listing history from the latest commit to the given CommitID.

--revision|**-r** <*CommitID*>..
: Start listing history from the given CommitID to the earliest commit.

--verbose|**-v**
: Include information about which files changes as part of each commit.

--debug|**-d** <*DebugLevel*>
: Set the debug level. A value greater than zero will result in lots of text being printed to the screen and written to the log file.

--help|**-h**
: Show usage message.

--version|**-V**
: Show the FlexSync software version and exit.

--credits
: Print legal disclosures.

EXIT VALUES

flexsync will return 0 on success and non-zero on failure.

EXAMPLES

List the history of the current directory within a working directory.

```
flexsync log
```

List the entire history of the repository "video".

```
flexsync log -v flexsync://video
```

SEE ALSO

flexsync(1) **flexsync-diff**(1) **flexsync-log**(1)

NAME

flexsync mkdir − Make a directory in the repository

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsync mkdir [*options*] <*URI*>

DESCRIPTION

The **mkdir** command makes a directory inside a **FlexSync** repository. The directory is specified by a URI. A mkdir can have a **commit message** which describes the changes in a human-understandable way.

If the user does not provide the **-c** argument to specify a commit message, this command will start an editor so the user can type one in.

NOTE: Providing the flexsync:// prefix in the URI is required for this command.

Usually, one might create a new directory by checking out the parent of the prospective new directory, creating the directory by hand, and then doing a commit. If the parent already contains a large amount of data, the checkout may be prohibitively expensive. Instead, the user can call **mkdir** to create the directory without doing the checkout. Often, the newly created directory is subsequently checked out as the root of a new working directory.

If the directory already exists, the command succeeds without doing anything.

OPTIONS

--comment|**-c** <*CommitText*>
: Text to place in the commit that creates the new directory.

--debug|**-d** <*DebugLevel*>
: Set the debug level. A value greater than zero will result in lots of text being printed to the screen and written to the log file.

--help|**-h**
: Show usage message.

--version|**-V**
: Show the FlexSync software version and exit.

--credits
: Print legal disclosures.

EXIT VALUES

flexsync will return 0 on success and non-zero on failure.

EXAMPLES

To create a new directory:

```
flexsync -c 'A new directory to hold my 100s of GBs of selfies' mkdir \
flexsync://repo/path/to/selfies
```

SEE ALSO

flexsync(1)

NAME

flexsync revert − Discard changes to a working directory

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsync revert [*options*] *[path_in_workdir]*

DESCRIPTION

The **revert** command undoes changes made to a working directory by the user. The path specified by the argument is put back to the state it was in when it was last checked out or updated. Any local modifications will be lost.

If no path is given as an argument, the current directory is used.

OPTIONS

--revision|**-r** <*CommitID*>

Specifying a particular revision allows the user to rewind part of a working directory to the state it was in a previous version of the repository. Use the **log** command to find a *CommitID* to supply to this command.

--ignoreMissing|**-I**

If content is encountered that is missing, it will be logged but ignored, and the command will continue. The default is to report the missing item as an error and stop processing.

--full-scan|**-S**

Enforce full scan of the working directory.

--debug|**-d** <*DebugLevel*>

Set the debug level. A value greater than zero will result in lots of text being printed to the screen and written to the log file.

--help|**-h**

Show usage message.

--version|**-V**

Show the FlexSync software version and exit.

--credits

Print legal disclosures.

EXIT VALUES

flexsync will return 0 on success and non-zero on failure.

NOTES

When doing a revert and providing an old revision, it is possible that missing content may be encountered. (Missing content is typically seen for files that existed in an older revision, however they have since been deleted and cleaned and are no longer in the repository.) When this content is encountered it is treated as an error unless the **--ignoreMissing** option has been specified. When the option is specified, files that are missing are logged to: /var/log/flexsync/flexsync.cli.log This is an example of the log entry that is generated for a missing file:

```
Info (revert): File object missing from the cloud bucket, unable to retrieve: sub
```

These log messages are informational only and all previous content that can be reverted will be.

SEE ALSO

flexsync(1) **flexsync-log**(1)

NAME

flexsync diff − Query the state of a working directory

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsync diff [*options*] *[file|dir]*

DESCRIPTION

The **diff** command inspects all or part of a working directory and describes what changes have been made to it. On the command line, the user can supply a single file or directory that exists within a working directory. If *file|dir* is omitted, the current directory is used. The command then prints a list of zero or more paths corresponding to directory entries that have been changed. Before each path name is either an 'A', 'D', or 'M'. That means the path was Added, Deleted, or Modified (respectively).

By default, the changes listed are what would be committed if the user were to do a **commit**. The **-u** option causes **diff** to show what changes would be made to the working directory if **update** was run.

OPTIONS

--update|-u

The **-u** option causes **diff** to show what changes would be made to the working directory if **update** was run.

--full-scan|-S

Enforce full scan of the working directory.

--debug|-d <*DebugLevel*>

Set the debug level. A value greater than zero will result in lots of text being printed to the screen and written to the log file.

--help|-h

Show usage message.

--version|-V

Show the FlexSync software version and exit.

--credits

Print legal disclosures.

EXIT VALUES

flexsync will return 0 on success and non-zero on failure.

SEE ALSO

flexsync(1) **flexsync-update**(1)

NAME

flexsync update − Update the entire checkout directory with changes from a repository

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsync update *[options] [workdir]*

DESCRIPTION

The **update** command downloads file changes from a repository and applies them to the checkout directory which contains the specified workdir directory. Those changes were made by using **commit** in a different checkout directory for the same repository.

The *workdir* argument specifies a location within the previously checked out directory. It may point to any file or directory under the checked out directory, or the checkout directory itself. If *workdir* is omitted, the the current directory (".") is used. NOTE: The entire checked out directory is updated when this is used, not just the directory or file specified. As an example, let's say you checked out a repository to /stornext/cloud directory. Within that is a directory called sub1 and another called sub2. If an update is issued for /stornext/cloud/sub1, all files from /stornext/cloud will be updated and not just /stornext/cloud/sub1.

The **-r** option can be used to specify a particular CommitID to update to. If the option is not present, the checkout directory is updated to match the most recent CommitID of the repository.

The contents of a checkout directory are always at the same CommitID. So, an update in a subdirectory of a FlexSync checkout directory causes the whole checkout directory to be updated.

A user can see what files would be modified by an update using the **-u** option to the **diff** command.

CONFLICTS

FlexSync lets many users make simultaneous changes to many different files in the same directory structure, but what happens when two people try to change the **same** file at the same time? The first user to commit their change sees no problem. Their commit proceeds as normal. When the second user tries to commit their change, they get an error saying that their checkout directory is out of date and should be updated. When they update, FlexSync sees that there are two changes to the same file. This is called a conflict.

A conflict requires user help to resolve. When **update** encounters a file with a conflict, it asks the user how to proceed. A list of options is presented. The user picks one and the update continues to the next file. Allowed options are *discard*, *keep*, *rename-local*, *rename-repo*, and *abort*. Details about each option can be found in under **--conflict** in this man page.

OPTIONS

--conflict <*mode*>

Specify a conflict resolution mode to be used for all conflicts found by this update. Valid values are:

discard throw away all local changes to the file and replace the file with the updated version from the repository.

keep ignore remote changes to the file and keep the local changes to the file.

rename-local rename the local file by adding ".local" to the end of its name, then download the remote version of the file and create it with its correct name.

rename-repo leave the local file alone and create a new file to contain the repo version. The new file's name ends in a period followed by the remote file's VersionID.

abort cause the whole update to fail. Everything will be left as it was before update

was run.

count count the number of conflicts and stop. No updates will be made.

ask prompt the user for the action to take for each conflict.

--revision|**-r** <*CommitID*>

Specifying a particular revision allows the user to update a checkout directory to a different version (older or newer) of the repository. Use the **log** command to find a *CommitID* to supply to this command.

--ignoreMissing|**-I**

If content is encountered that is missing, it will be logged but ignored, and the command will continue. The default is to report the missing item as an error and stop processing.

--full-scan|**-S**

Enforce full scan of the working directory.

--debug|**-d** <*DebugLevel*>

Set the debug level. A value greater than zero will result in lots of text being printed to the screen and written to the log file.

--help|**-h**

Show usage message.

--version|**-V**

Show the FlexSync software version and exit.

--credits

Print legal disclosures.

EXIT VALUES

flexsync will return 0 on success and non-zero on failure.

EXAMPLES

If your current working directory is within a checkout directory, you can update it to the repository's latest revision with the command.

```
flexsync update
```

An update with a conflict looks like:

```
# flexsync update
Initializing...
Filtering...           metadata archive unavailable
Scanning...            complete
Conflict at image-12.png (locally modified, remotely modified)e downloaded
[d]iscard local modification, [k]eep local modification, rename [l]ocal version,
: r
Conflict at image-12.png (locally modified, remotely modified), renaming repo ver
Complete.
```

NOTES

When doing an update and providing an old revision, it is possible that missing content may be encountered. (Missing content is typically seen for files that existed in an older revision, however they have since been deleted and cleaned and are no longer in the repository.) When this content is encountered it is treated as an error unless the **--ignoreMissing** option has been specified. When the option is specified, files that are missing are logged to: /var/log/flexsync/flexsync.cli.log This is an example of the log entry that is generated for a missing file:

```
Info (update): File object missing from the cloud bucket, unable to retrieve: sub
```

These log messages are informational only and all previous content that can be updated will be.

SEE ALSO

flexsync(1) **flexsync-commit**(1) **flexsync-log**(1) **flexsync-diff**(1)

NAME

flexsync_mount − Mount a local view of a FlexSync object repository

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsync_mount *[-f][-d] <mount_pt> <URI>*

DESCRIPTION

The **flexsync_mount** program is used to mount a read-only view of a FlexSync repository on the local system. It provides a file system interface for examining the current contents of the object repository. This is primarily meant as an admin tool for examining the current repository contents. It is not recommended for typical day-to-day file system use of the repository.

The FlexSync repository must have already been initialized and registered using the *flexsyncadmin register* command.

The URI is made of the form: *flexsync://reponame*. The *reponame* is just the text name given to the repository when it was initialized. It's typically the last part of the URL used to register the repository (with the **flexsyncadmin register** command).

NOTE: Providing the flexsync:// prefix in the URI is optional for this command.

OPTIONS

The *-f* option indicates the process should be run in the foreground.

The *-d* option is used to indicate running the process in debug mode. (Using this option implies the *-f* option.) This option is provided for troubleshooting and is not normally desired.

NOTES

The view of the repository is read only. It can be used for listing and reading contents of the repository but no changes can be made. Such attempts will fail, typically with an error like: "...: Invalid argument". If you just want to list the current contents of a few directories in a repository, a better choice might be the **list** command.

No data is cached to local disk by the mount process, or by list/read operations. Data is read from the repository and provided directly to the process doing the read from the file system. So if a file is read from the file system, and then immediately read again, it results in reading the file from the repository twice. If a user wants to work with a file, and that file will be read multiple times while working, the user is better off using the **get** command to obtain a local copy of the file. (Assuming a **checkout** of the directory containing the file is not desired.)

NOTE

The **flexsync_mount** utility is currently an alpha version and continues to be improved. Be aware that there are occasions where it may not behave fully as expected. Known issues:

- If the **mount(8)** command is run when a flexsync repository is mounted, it reports the file system options including a "rw". This option is needed internally for filesystem operations, but the repository file system is not writable.

- Internally to the code, the **flexsync_mount** utilizes the FUSE (FileSystem in User Space) interface to assist in mounting the filesystem. Because of this, there may be times where messages are output that have the term "fuse" in them. Although the messages are valid, the options that may be recommended may not be valid for use with **flexsync_mount**.

SEE ALSO

flexsyncadmin(8), **flexsync-checkout**(1), **flexsync-get**(1), **flexsync-list**(1), **flexsync**(1)

NAME

flexsyncadmin getconfig − Read a configuration key

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsyncadmin [--username=*USERNAME* | -U *USERNAME*] [--password=*PASSWD* | -P *PASSWD*]
getconfig <*repository_name:key*> [--register-user <user>] [--register-hostgroup <hostgroup --register-host <host>]

flexsyncadmin [--username=*USERNAME* | -U *USERNAME*] [--password=*PASSWD* | -P *PASSWD*]
getconfig <*repository_name*> [--register-user <user>] [--register-hostgroup <hostgroup --register-host <host>]

DESCRIPTION

The **getconfig** command gets the current values of various **FlexSync repository** configuration parameters.

Specifying only the the repository name without a key will return all keys and their values for that repository.

Specifying the key with the repository name will return the value for that key.

Some parameters only apply to a particular repository. For those parameters, the repository name and a colon are prepended before the key name.

OPTIONS

--register-user <*user*>
The user for which configuration parameters to obtain. Defaults to root.

--register-hostgroup <*hostgroup*> **--register-host** <*host*>
The hostgroup and host for which configuration parameters to obtain. Defaults to the local host.

AVAILABLE KEYS

Repository configuration values:

repo:connect_timeout
The number of milliseconds to wait when trying to connect to the S3 storage.

repo:data_timeout
The number of milliseconds to wait for a message from the S3 storage before giving up.

repo:max_connections
The number of simultaneous connections to the S3 storage.

repo:max_pipeline
The maximum number of asynchronous requests on a single S3 connection that may be sent. The value of 0 means unlimited.

repo:max_requests
The maximum number of outstanding requests.

repo:multipart_segment
When doing a multi-part upload, this is the number of bytes in each segment.

repo:multipart_threshold
The maximum size in bytes of a single-part upload. Every upload larger than this ends up being a multi-part upload.

repo:publish_grace_time
The number of milliseconds to wait for arbitrage agent to consume a new head and create a completed head in commands that change repository.

repo:num_threads
The number of threads doing simultaneous network transfers to/from the S3 storage.

repo:retries
This parameter can have a value of 0 or 1. If 1, S3 requests will be retried if they fail. If 0, a S3 request failure will cause the **flexsyncadmin** command to exit immediately.

NOTE: above values can be changed with the **flexsyncadmin setconfig** command.

Repository policy values:

repo:checksum
Whether file checksum is turned on for this repository or not.

repo:dedup
Whether file dedup is turned on for this repository or not.

repo:retention_mode
The retention mode of old/stale versions of files and directories in the repository. Values are:
- period - old versions are kept for a period of time.
- none - old/stale versions are not kept.
- permanent - old/stale versions are never deleted.

repo:retention_period
The retention period in days if retention_mode is set to 'period'.

NOTE: above values can be changed with the **flexsyncadmin init --change-policy** command.

EXIT VALUES

flexsyncadmin will return 0 on success and non-zero on failure.

SEE ALSO

flexsyncadmin(8), **flexsyncadmin-setconfig**(1), **flexsyncadmin-rstconfig**(1), **flexsync**(1)

NAME

flexsyncadmin rstconfig − Reset a configuration key to its default value

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsyncadmin [--username=*USERNAME* | -U *USERNAME*] [--password=*PASSWD* | -P *PASSWD*]
rstconfig <repository_name> | <repository_name:key value> [--register-user <user>]

flexsyncadmin [--username=*USERNAME* | -U *USERNAME*] [--password=*PASSWD* | -P *PASSWD*]
rstconfig <repository_name> | <repository_name:key value> --register-hostgroup <hostgroup> --register-host <host> [--register-user <user>]

flexsyncadmin [--username=*USERNAME* | -U *USERNAME*] [--password=*PASSWD* | -P *PASSWD*]
rstconfig <repository_name> | <repository_name:key value> --all-hosts

DESCRIPTION

The **rstconfig** command resets all **FlexSync** configuration keys to their default value or resets a specific **FlexSync** configuration key to its default value.

If only the repository_name is specified and no key, all configuration keys are reset to their default values.

Without any destination parameters, the configuration key(s) will be reset on the local host.

If the *--register-host* and *--register-hostgroup* are specified, the configuration key(s) will be reset for the specified hostgroup/host.

If the *--all-hosts* is specified, the configuration key(s) will be reset for all configured hosts in all configured hostgroups and for all configured users.

If the *--register-user* is specified, the configuration key(s) will be reset for the specified user. If not specified, it will be reset for the root user.

OPTIONS

--register-user <*user*>
Reset the configuration key(s) for the specified user. Defaults to root.

--register-hostgroup <*hostgroup*> **--register-host** <*host*>
Reset the configuration key(s) for the specified hostgroup and host.

--all-hosts
Reset the configuration key(s) for all hosts in all hostgroups and for all users.

AVAILABLE KEYS

See **flexsyncadmin-setconfig**.

EXIT VALUES

flexsyncadmin will return 0 on success and non-zero on failure.

SEE ALSO

flexsyncadmin(8), **flexsyncadmin-getconfig**(1), **flexsyncadmin-setconfig**(1), **flexsync**(1)

NAME

flexsyncadmin setconfig − Set a configuration key

PRODUCT NOTE

The **FlexSync** product provides two basic types of synching operations for user file systems. The first operation is for copying/synching data from one user file system to another. The second operation is used for synching a user file system to/from an object repository. The first type of operation is referred to as a **sync** operation. The second type of operation is referred to as an **object** operation. For operations that are to be run regularly, tasks can be added. (These are referred to as **sync tasks** and **object tasks**.)

This man page covers functionality used in **object** operations.

SYNOPSIS

flexsyncadmin [--username=*USERNAME* | -U *USERNAME*] [--password=*PASSWD* | -P *PASSWD*]
setconfig <*repository_name:key*> <*value*> [--register-user <user>]

flexsyncadmin [--username=*USERNAME* | -U *USERNAME*] [--password=*PASSWD* | -P *PASSWD*]
setconfig <*repository_name:key*> <*value*> --register-hostgroup <hostgroup> --register-host <host> [--register-user <user>]

flexsyncadmin [--username=*USERNAME* | -U *USERNAME*] [--password=*PASSWD* | -P *PASSWD*]
setconfig <*repository_name*> <*value*> --all-hosts

DESCRIPTION

The **setconfig** command sets a specific **flexSync** configuration parameter. The command takes the repository name, key name, and value as arguments.

Without any destination parameters, the key will be set on the local host.

If the *--register-host* and *--register-hostgroup* are specified, the configuration key will be set for the specified hostgroup/host.

If the *--all-hosts* is specified, the configuration key will be set for all configured hosts in all configured hostgroups and for all configured users.

If the *--register-user* is specified, the configuration key will be set for the specified user. If not specified, it will be set for the root user.

OPTIONS

--register-user <*user*>
Set the configuration key for the specified user. Defaults to root.

--register-hostgroup <*hostgroup*> **--register-host** <*host*>
Set the configuration key for the specified hostgroup and host.

--all-hosts
Set the configuration key for all hosts in all hostgroups and for all users.

AVAILABLE KEYS

repo:max_connections
The number of simultaneous connections to the S3 storage. Minimum value allowed is 1. Defaults to 8.

repo:num_threads
The number of threads doing simultaneous network transfers to/from the S3 storage. Minimum value allowed is 1. Defaults to 8.

repo:max_pipeline
The maximum number of asynchronous (in-flight HTTP) requests on a single S3 connection that may be sent. Allowing multiple requests in flight on a socket allows more performance by hiding latency better. NOTE: Some S3 targets do not support pipelining so the default is set to 1 (off).

Refer to the vendor documentation for appropriate setting. The value of 0 means unlimited. Defaults to 1 (Equates to no pipelining).

repo:max_requests
The maximum number of requests made on a single HTTP connection before it is torn down and a new one is created. This should be large enough that the connection tear-down and reconnect cost is amortized, but most HTTP servers do not support an infinite number of requests on a given HTTP socket because they want to limit the total number of HTTP connections they allow and they do not want a small number of clients to monopolize the server. Minimum value allowed is 0. Defaults to 90.

repo:data_timeout
The number of milliseconds to wait for a message from the S3 storage before giving up. Minimum value allowed is 0. Defaults to 600000.

repo:connect_timeout
The number of milliseconds to wait when trying to connect to the S3 storage. Minimum value allowed is 0. Defaults to 2000.

repo:multipart_threshold
The maximum size in bytes of a single-part upload. An upload larger than this value will become a multi-part upload, based upon the **multipart_segment** size. Prior to making changes, please refer to the S3 vendor's documentation for specific allowed values. Typical minimum value is 5 MiB. Defaults to 1073741823.

repo:multipart_segment
When doing a multi-part upload, this is the number of bytes in each segment. Prior to making changes, please refer to the S3 vendor's documentation for specific allowed values. Typical range allowed is 5 MiB to 5 GiB. Minimum value is 1048576. Defaults to 268435456. NOTE: The value will be rounded to the nearest MB.

repo:publish_grace_time
The number of milliseconds to wait for arbitrage agent to consume a new head and create a completed head in commands that change repository. Minimum value allowed is 1. Defaults to 60000.

repo:retries
This parameter can have a value of 0 or 1. If 1, S3 requests will be retried if they fail. If 0, a S3 request failure will cause the **flexsyncadmin** command to exit immediately. Defaults to 1.

EXIT VALUES

flexsyncadmin will return 0 on success and non-zero on failure.

SEE ALSO

flexsyncadmin(8), **flexsyncadmin-getconfig**(1), **flexsyncadmin-rstconfig**(1), **flexsync**(1)

Quantum

Quantum technology, software, and services provide the solutions that today's organizations need to make video and other unstructured data smarter – so their data works for them and not the other way around. With over 40 years of innovation, Quantum's end-to-end platform is uniquely equipped to orchestrate, protect, and enrich data across its lifecycle, providing enhanced intelligence and actionable insights. Leading organizations in cloud services, entertainment, government, research, education, transportation, and enterprise IT trust Quantum to bring their data to life, because data makes life better, safer, and smarter. Quantum is listed on Nasdaq (QMCO) and the Russell 2000® Index. For more information visit www.quantum.com.

www.quantum.com | 800-677-6268