NAME

snap – Tool to interact with snaps

SYNOPSIS

snap [OPTIONS]

DESCRIPTION

The snap command lets you install, configure, refresh and remove snaps. Snaps are packages that work across many different Linux distributions, enabling secure delivery and operation of the latest apps and utilities.

OPTIONS

COMMANDS

abort

Abort a pending change

The *abort* command attempts to abort a change that still has pending tasks.

Usage: snap [OPTIONS] abort [abort-OPTIONS]

−−last Select last change of given type (install, refresh, remove, try, auto-refresh, etc.). A question mark at the end of the type means to do nothing (instead of returning an error) if no change of the given type is found. Note the question mark could need protecting from the shell.

ack

Add an assertion to the system

The *ack* command tries to add an assertion to the system assertion database.

The assertion may also be a newer revision of a pre-existing assertion that it will replace.

To succeed the assertion must be valid, its signature verified with a known public key and the assertion consistent with and its prerequisite in the database.

alias

Set up a manual alias

The *alias* command aliases the given snap application to the given alias.

Once this manual alias is setup the respective application command can be invoked just using the alias.

Usage: snap [OPTIONS] alias [alias-OPTIONS]

−−no-wait

Do not wait for the operation to finish but just print the change id.

aliases

List aliases in the system

The *aliases* command lists all aliases available in the system and their status.

$ snap aliases <snap>

Lists only the aliases defined by the specified snap.

changes

List system changes

The *changes* command displays a summary of system changes performed recently.

Usage: snap [OPTIONS] changes [changes-OPTIONS]

−−abs-time
Display absolute times (in RFC 3339 format). Otherwise, display relative times up to 60 days, then YYYY-MM-DD.

check-snapshot
Check a snapshot

The *check-snapshot* command verifies the user, system and configuration data of the snaps included in the specified snapshot.

The check operation runs the same data integrity verification that is performed when a snapshot is restored.

By default, this command checks all the data in a snapshot. Alternatively, you can specify the data of which snaps to check, or for which users, or a combination of these.

If a snap is included in a check-snapshot operation, excluding its system and configuration data from the check is not currently possible. This restriction may be lifted in the future.

Usage: snap [OPTIONS] check-snapshot [check-snapshot-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

−−users
Check data of only specific users (comma-separated) (default: all users)

components
List available and installed components for installed snaps

The *components* command displays a summary of the components that are installed and available for the set of currently installed snaps.

Components for specific installed snaps can be queried by providing snap names as positional arguments.

connect
Connect a plug to a slot

The *connect* command connects a plug to a slot. It may be called in the following ways:

$ snap connect <snap>:<plug> <snap>:<slot>

Connects the provided plug to the given slot.

$ snap connect <snap>:<plug> <snap>

Connects the specific plug to the only slot in the provided snap that matches the connected interface. If more than one potential slot exists, the command fails.

$ snap connect <snap>:<plug>

Connects the provided plug to the slot in the core snap with a name matching the plug name.

Usage: snap [OPTIONS] connect [connect-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

connections
List interface connections

The *connections* command lists connections between plugs and slots in the system.

Unless <snap> is provided, the listing is for connected plugs and slots for all snaps in the system. In this mode, pass --all to also list unconnected plugs and slots.

$ snap connections <snap>

Lists connected and unconnected plugs and slots for the specified snap.

Usage: snap [OPTIONS] connections [connections-OPTIONS]

−−all Show connected and unconnected plugs and slots

create-cohort
Create cohort keys for a set of snaps

The *create-cohort* command creates a set of cohort keys for a given set of snaps.

A cohort is a view or snapshot of a snap's "channel map" at a given point in time that fixes the set of revisions for the snap given other constraints (e.g. channel or architecture). The cohort is then identified by an opaque per-snap key that works across systems. Installations or refreshes of the snap using a given cohort key would use a fixed revision for up to 90 days, after which a new set of revisions would be fixed under that same cohort key and a new 90 days window started.

debug
Run debug commands

The debug command contains a selection of additional sub-commands.

Debug commands can be removed without notice and may not work on non-development systems.

debug api
Execute raw query to snapd API

Execute a raw query to snapd API. Complex input can be read from stdin, while output is printed to stdout. See examples below:

List all snaps: $ snap debug api /v2/snaps

Find snaps with name foo: $ snap debug api '/v2/find?name=foo'

Request refresh of snap 'some-snap': $ echo '{"action": "refresh"}' | snap debug api -X POST \
-H 'Content-Type: application/json' /v2/snaps/some-snap

Execute a request to the session agent of UID 12345: $ snap debug api --session-agent-uid=12345 /v1/session-info

Usage: snap [OPTIONS] debug api [api-OPTIONS]

−−snap-socket
Use snap access socket

−−session-agent-uid
Communicate with session agent of a given UID

−−disable-auth
Disable authorization with data from auth.json

−H, **−−header**
Set header (can be repeated multiple times), header kind and value are separated with ': '

−X, **−−request**
HTTP method to use (defaults to GET)

−−fail Fail on request errors

debug confinement
Print the confinement mode the system operates in

The *confinement* command will print the confinement mode (strict, partial or none) the system operates in.

debug connectivity
Check network connectivity status

The *connectivity* command checks the network connectivity of snapd.

debug execution
Obtain information about execution aspects of snap toolchain commands

Display debugging information about aspects of snap toolchain execution, such as reexecution, tools location etc.

debug execution apparmor
Show apparmor

debug execution internal-tool
Show internal tool execution info

debug execution snap
Show snap execution info

debug features
Obtain the complete list of feature tags

Display json output that contains the complete list of feature tags present in snapd and snap. Feature tags are a collection of data that describe significant code paths within snapd including tasks, changes, interfaces, endpoints, snap commands, and ensure helper functions.

debug lsm
(internal) obtain status information on LSMs

(internal) obtain status information on LSMs

debug migrate-home
Migrate snaps' directory to ~/Snap.

Migrate snaps' directory to ~/Snap.

Usage: snap [OPTIONS] debug migrate-home [migrate-home-OPTIONS]

−−no-wait

debug paths
Print system paths

The *paths* command prints the list of paths detected and used by snapd.

debug refresh-app-awareness
(internal) list refresh-app-awareness details

(internal) list refresh-app-awareness details

Usage: snap [OPTIONS] debug refresh-app-awareness [refresh-app-awareness-OPTIONS]

−−unicode <default: *"auto"*>

debug sandbox-features
Print sandbox features available on the system

The sandbox command prints tags describing features of individual sandbox components used by snapd on a given system.

Usage: snap [OPTIONS] debug sandbox-features [sandbox-features-OPTIONS]

−−required
Ensure that given backend:feature is available

debug seeding
Obtain seeding and preseeding details

Obtain seeding and preseeding details

Usage: snap [OPTIONS] debug seeding [seeding-OPTIONS]

−−unicode <default: *"auto"*>

debug stacktraces
Obtain stacktraces of all snapd goroutines

Obtain stacktraces of all snapd goroutines.

debug state
Inspect a snapd state file.

Inspect a snapd state file, bypassing snapd API.

Usage: snap [OPTIONS] debug state [state-OPTIONS]

−−abs-time
Display absolute times (in RFC 3339 format). Otherwise, display relative times up to 60 days, then YYYY-MM-DD.

−−changes
List all changes

−−task ID of the task to inspect

−−change
ID of the change to inspect

−−check
Check change consistency

−−connections
List all connections

−−connection
Show details of the matching connections (snap or snap:plug,snap:slot or snap:plug-or-slot

−−is-seeded
Output seeding status (true or false)

−−dot Dot (graphviz) output

−−no-hold
Omit tasks in 'Hold' state in the change output

debug timings
Get the timings of the tasks of a change

The *timings* command displays details about the time each task runs.

Usage: snap [OPTIONS] debug timings [timings-OPTIONS]

−−last Select last change of given type (install, refresh, remove, try, auto-refresh, etc.). A question mark at the end of the type means to do nothing (instead of returning an error) if no change of the given type is found. Note the question mark could need protecting from the shell.

−−ensure
Show timings for a change related to the given Ensure activity (one of: auto-refresh, become-operational, refresh-catalogs, refresh-hints, seed)

−−all Show timings for all executions of the given Ensure or startup activity, not just the latest

−−startup
Show timings for the startup of given subsystem (one of: load-state, ifacemgr)

−−verbose
Show more information

debug validate-seed
Validate snap seed

Validate correctness of snap seed located in the directory containing seed.yaml file.

disable
Disable a snap in the system

The *disable* command disables a snap. The binaries and services of the snap will no longer be available, but all the data is still available and the snap can easily be enabled again.

Usage: snap [OPTIONS] disable [disable-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

disconnect
Disconnect a plug from a slot

The *disconnect* command disconnects a plug from a slot. It may be called in the following ways:

$ snap disconnect <snap>:<plug> <snap>:<slot>

Disconnects the specific plug from the specific slot.

$ snap disconnect <snap>:<slot or plug>

Disconnects everything from the provided plug or slot. The snap name may be omitted for the core snap.

When an automatic connection is manually disconnected, its disconnected state is retained after a snap refresh. The --forget flag can be added to the disconnect command to reset this behaviour, and consequently re-enable an automatic reconnection after a snap refresh.

Usage: snap [OPTIONS] disconnect [disconnect-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

−−forget
Forget remembered state about the given connection.

download
Download the given snap

The *download* command downloads the given snap, components, and their supporting assertions to the current directory with .snap, .comp, and .assert file extensions, respectively.

Usage: snap [OPTIONS] download [download-OPTIONS]

−−channel
Use this channel instead of stable

−−edge Install from the edge channel

−−beta Install from the beta channel

−−candidate
Install from the candidate channel

−−stable
Install from the stable channel

−−revision
Download the given revision of a snap. When downloading components, download the components associated with the given snap revision.

−−basename
Use this basename for the snap, component, and assertion files (defaults to <snap>_<revision>)

−−target-directory
Download to this directory (defaults to the current directory)

−−only-components
Only download the given components, not the snap

−−cohort
Download from the given cohort

enable
Enable a snap in the system

The *enable* command enables a snap that was previously disabled.

Usage: snap [OPTIONS] enable [enable-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

export-key
Export cryptographic public key

The *export-key* command exports a public key assertion body that may be imported by other systems.

Usage: snap [OPTIONS] export-key [export-key-OPTIONS]

−−account
Format public key material as a request for an account-key for this account-id

export-snapshot
Export a snapshot

Export a snapshot to the given filename.

find
Find packages to install

The *find* command queries the store for available packages.

With the --private flag, which requires the user to be logged-in to the store (see 'snap help login'), it instead searches for private snaps that the user has developer access to, either directly or through the store's collaboration feature.

A green check mark (given color and unicode support) after a publisher name indicates that the publisher has been verified.

Usage: snap [OPTIONS] find [find-OPTIONS]

Aliases: search

−−private
Search private snaps.

−−narrow
Only search for snaps in "stable".

−−section [=*"show-all-sections-please"*] <default: *"no-section-specified"*>
Restrict the search to a given section.

−−color <default: *"auto"*>
Use a little bit of color to highlight some things.

−−unicode <default: *"auto"*>
Use a little bit of Unicode to improve legibility.

forget
Delete a snapshot

The *forget* command deletes a snapshot. This operation can not be undone.

A snapshot contains archives for the user, system and configuration data of each snap included in the snapshot.

By default, this command forgets all the data in a snapshot. Alternatively, you can specify the data of which snaps to forget.

Usage: snap [OPTIONS] forget [forget-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

get

Print configuration options

The *get* command prints configuration options for the provided snap.

```
$ snap get snap-name username
frank
```

If multiple option names are provided, the corresponding values are returned:

```
$ snap get snap-name username password
Key      Value
username  frank
password  ...
```

Nested values may be retrieved via a dotted path:

```
$ snap get snap-name author.name
frank
```

Usage: snap [OPTIONS] get [get-OPTIONS]

−t Strict typing with nulls and quoted strings

−d Always return document, even with single key

−l Always return list, even with single key

−−default
A strictly typed default value to be used when none is found

help

Show help about a command

The *help* command displays information about snap commands.

Usage: snap [OPTIONS] help [help-OPTIONS]

−−all Show a short summary of all commands

import-snapshot

Import a snapshot

Import an exported snapshot set to the system. The snapshot is imported with a new snapshot ID and can be restored using the restore command.

Usage: snap [OPTIONS] import-snapshot [import-snapshot-OPTIONS]

−−abs-time

info

Show detailed information about snaps

The *info* command shows detailed information about snaps.

The snaps can be specified by name or by path; names are looked for both in the store and in the installed snaps; paths can refer to a .snap file, or to a directory that contains an unpacked snap suitable for 'snap try' (an example of this would be the 'prime' directory snapcraft produces).

Usage: snap [OPTIONS] info [info-OPTIONS]

−−color <default: *"auto"*>
Use a little bit of color to highlight some things.

−−unicode <default: *"auto"*>
Use a little bit of Unicode to improve legibility.

−−abs-time
Display absolute times (in RFC 3339 format). Otherwise, display relative times up to 60 days, then YYYY-MM-DD.

−−verbose
Include more details on the snap (expanded notes, base, etc.)

install
Install snaps on the system

The *install* command installs the named snaps on the system.

To install multiple instances of the same snap, append an underscore and a unique identifier (for each instance) to a snap's name.

Parallel instances are installed with --unaliased passed implicitly to avoid conflicts with existing installs. This behaviour can be altered by passing --prefer which will enable all aliases of the given snap in preference to conflicting aliases of other snaps whose automatic aliases will be disabled and manual aliases will be removed.

With no further options, the snaps are installed tracking the stable channel, with strict security confinement. All available channels of a snap are listed in its 'snap info' output.

When --revision is used, a later refresh will typically undo the revision override, taking the snap back to the current revision of the channel it's tracking.

Use --name to set the instance name when installing from snap file.

Usage: snap [OPTIONS] install [install-OPTIONS]

−−color <default: *"auto"*>
Use a little bit of color to highlight some things.

−−unicode <default: *"auto"*>
Use a little bit of Unicode to improve legibility.

−−no-wait
Do not wait for the operation to finish but just print the change id.

−−channel
Use this channel instead of stable

−−edge Install from the edge channel

−−beta Install from the beta channel

−−candidate
Install from the candidate channel

−−stable
Install from the stable channel

−−devmode
Put snap in development mode and disable security confinement

−−jailmode
Put snap in enforced confinement mode

−−classic
Put snap in classic mode and disable security confinement

−−revision
Install the given revision of a snap

−−dangerous
Install the given snap file even if there are no pre-acknowledged signatures for it, meaning it was not verified and could be dangerous (--devmode implies this)

−−unaliased
Install the given snap without enabling its automatic aliases

−−prefer
Enable all aliases of the given snap in preference to conflicting aliases of other snaps

−−name
Install the snap file under the given instance name

−−cohort
Install the snap in the given cohort

−−ignore-validation
Ignore validation by other snaps blocking the installation

−−transaction <default: *"per-snap"*>
Have one transaction per-snap or one for all the specified snaps

−−quota-group
Add the snap to a quota group on install

interface
Show details of snap interfaces

The *interface* command shows details of snap interfaces.

If no interface name is provided, a list of interface names with at least one connection is shown, or a list of all interfaces if --all is provided.

Usage: snap [OPTIONS] interface [interface-OPTIONS]

−−attrs
Show interface attributes

−−all Include unused interfaces

known
Show known assertions of the provided type

The *known* command shows known assertions of the provided type. If header=value pairs are provided after the assertion type, the assertions shown must also have the specified headers matching the provided values.

Usage: snap [OPTIONS] known [known-OPTIONS]

−−remote
Query the store for the assertion, via snapd if possible

−−direct
Query the store for the assertion, without attempting to go via snapd

list
List installed snaps

The *list* command displays a summary of snaps installed in the current system.

A green check mark (given color and unicode support) after a publisher name indicates that the publisher has been verified.

Usage: snap [OPTIONS] list [list-OPTIONS]

−−all Show all revisions

−−color <default: *"auto"*>
Use a little bit of color to highlight some things.

−−unicode <default: *"auto"*>
Use a little bit of Unicode to improve legibility.

login
Authenticate to snapd and the store

The *login* command authenticates the user to snapd and the snap store, and saves credentials into the ~/.snap/auth.json file. Further communication with snapd will then be made using those credentials.

It's not necessary to log in to interact with snapd. Doing so, however, enables interactions without sudo, as well as some some developer-oriented features as detailed in the help for the find, install and refresh commands.

An account can be set up at https://login.ubuntu.com

logout
Log out of snapd and the store

The *logout* command logs the current user out of snapd and the store.

logs
Retrieve logs for services

The *logs* command fetches logs of the given services and displays them in chronological order.

Usage: snap [OPTIONS] logs [logs-OPTIONS]

−−abs-time
Display absolute times (in RFC 3339 format). Otherwise, display relative times up to 60 days, then YYYY-MM-DD.

−n <default: *"10"*>
Show only the given number of lines, or 'all'.

−f Wait for new lines and print them as they come in.

model
Get the active model for this device

The *model* command returns the active model assertion information for this device.

By default, only the essential model identification information is included in the output, but this can be expanded to include all of an assertion's non-meta headers.

The verbose output is presented in a structured, yaml-like format.

Similarly, the active serial assertion can be used for the output instead of the model assertion.

Usage: snap [OPTIONS] model [model-OPTIONS]

−−abs-time
Display absolute times (in RFC 3339 format). Otherwise, display relative times up to 60 days, then YYYY-MM-DD.

−−color <default: *"auto"*>
Use a little bit of color to highlight some things.

−−unicode <default: *"auto"*>
Use a little bit of Unicode to improve legibility.

−−serial
Print the serial assertion instead of the model assertion.

−−verbose
Print all specific assertion fields.

−−assertion
Print the raw assertion.

okay
Acknowledge warnings

The *okay* command acknowledges the warnings listed with 'snap warnings'.

Once acknowledged a warning won't appear again unless it re-occurrs and sufficient time has passed.

pack
Pack the given directory as a snap

The *pack* command packs the given snap-dir as a snap and writes the result to target-dir. If target-dir is omitted, the result is written to current directory. If both source-dir and target-dir are omitted, the pack command packs the current directory.

The default file name for a snap can be derived entirely from its snap.yaml, but in some situations it's simpler for a script to feed the filename in. In those cases, --filename can be given to override the default. If this filename is not absolute it will be taken as relative to target-dir.

When used with --check-skeleton, pack only checks whether snap-dir contains valid snap metadata and raises an error otherwise. Application commands listed in snap metadata file, but appearing with incorrect permission bits result in an error. Commands that are missing from snap-dir are listed in diagnostic messages.

Usage: snap [OPTIONS] pack [pack-OPTIONS]

−−check-skeleton
Validate snap-dir metadata only

−−filename
Output to this filename

−−compression
Compression to use (e.g. xz or lzo)

prefer
Enable aliases from a snap, disabling any conflicting aliases

The *prefer* command enables all aliases of the given snap in preference to conflicting aliases of other snaps whose aliases will be disabled (or removed, for manual ones).

Usage: snap [OPTIONS] prefer [prefer-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

prepare-image
Prepare a device image

The *prepare-image* command performs some of the steps necessary for creating device images.

For core images it is not invoked directly but usually via ubuntu-image.

For preparing classic images it supports a --classic mode

Usage: snap [OPTIONS] prepare-image [prepare-image-OPTIONS]

−−classic
Enable classic mode to prepare a classic model image

−−preseed
Preseed (UC20+ only)

−−preseed-sign-key
Name of the key to use to sign preseed assertion, otherwise use the default key

−−apparmor-features-dir
Optional path to apparmor kernel features directory (UC20+ only)

−−sysfs-overlay
Optional sysfs overlay to be used when running preseeding steps

−−arch Specify an architecture for snaps for --classic when the model does not

−−channel
The channel to use

−−snap *<snap>[=<channel>]*
Include the given snap from the store or a local file and/or specify the channel to track for the given snap

−−comp *<snap>+<comp>*
Include the given component from the store or a local file

−−revisions
Specify a seeds.manifest file referencing the exact revisions of the provided snaps which should be installed

−−write-revisions [*="./seed.manifest"*]
Writes a manifest file containing references to the exact snap revisions used for the image. A path for the manifest is optional.

−−validation
Control whether validations should be ignored or enforced. (default: ignore)

−−allow-snapd-kernel-mismatch
Whether a mismatch between versions of the snapd snap and snapd in kernel is allowed

−−assert *<filename>*
Include the assertion from the local file

quota
Show quota group for a set of snaps

The *quota* command shows information about a quota group, including the set of snaps and any sub-groups it contains, as well as its resource constraints and the current usage of those constrained resources.

quotas
Show quota groups

The *quotas* command shows all quota groups.

reboot
Reboot into selected system and mode

The *reboot* command reboots the system into a particular mode of the selected recovery system.

When called without a system label and without a mode it will just trigger a regular reboot.

When called without a label, the current system will be used for "run" mode. The default recovery system will be used for "recover", "factory-reset" and "install" modes.

Note that the "run" mode is only available for the current system.

Usage: snap [OPTIONS] reboot [reboot-OPTIONS]

−−run Boot into run mode

−−install
Boot into install mode

−−recover
Boot into recover mode

−−factory-reset
Boot into factory-reset mode

recovery
List available recovery systems

The *recovery* command lists the available recovery systems.

With --show-keys it displays recovery keys that can be used to unlock the encrypted partitions if the device-specific automatic unlocking does not work.

Usage: snap [OPTIONS] recovery [recovery-OPTIONS]

−−color <default: *"auto"*>
Use a little bit of color to highlight some things.

−−unicode <default: *"auto"*>
Use a little bit of Unicode to improve legibility.

−−show-keys
Show recovery keys (if available) to unlock encrypted partitions.

refresh
Refresh snaps in the system

The *refresh* command updates the specified snaps, or all snaps in the system if none are specified.

With no further options, the snaps are refreshed to the current revision of the channel they're tracking, preserving their confinement options. All available channels of a snap are listed in its 'snap info' output.

When --revision is used, a later refresh will typically undo the revision override.

Hold (--hold) is used to postpone snap refresh updates for all snaps when no snaps are specified, or for the

specified snaps.

When no snaps are specified --hold is only effective on auto-refreshes and will not block either general refresh requests from 'snap refresh' or specific snap requests from 'snap refresh target-snap'.

When snaps are specified --hold is effective on both their auto-refreshes and general refresh requests from 'snap refresh'. However, specific snap requests from 'snap refresh target-snap' remain unblocked and will proceed.

Usage: snap [OPTIONS] refresh [refresh-OPTIONS]

−−color <default: *"auto"*>
Use a little bit of color to highlight some things.

−−unicode <default: *"auto"*>
Use a little bit of Unicode to improve legibility.

−−abs-time
Display absolute times (in RFC 3339 format). Otherwise, display relative times up to 60 days, then YYYY-MM-DD.

−−no-wait
Do not wait for the operation to finish but just print the change id.

−−channel
Use this channel instead of stable

−−edge Install from the edge channel

−−beta Install from the beta channel

−−candidate
Install from the candidate channel

−−stable
Install from the stable channel

−−devmode
Put snap in development mode and disable security confinement

−−jailmode
Put snap in enforced confinement mode

−−classic
Put snap in classic mode and disable security confinement

−−amend
Allow refresh attempt on snap unknown to the store

−−revision
Refresh to the given revision

−−cohort
Refresh the snap into the given cohort

−−leave-cohort
Refresh the snap out of its cohort

−−list Show the new versions of snaps that would be updated with the next refresh

−−time Show auto refresh information but do not perform a refresh

−−ignore-validation
Ignore validation by other snaps blocking the refresh

−−transaction <default: *"per-snap"*>
Have one transaction per-snap or one for all the specified snaps

−−hold [=*"forever"*]
Hold refreshes for a specified duration (or forever, if no value is specified)

−−unhold
Remove refresh hold

remodel
Remodel this device

The *remodel* command changes the model assertion of the device, either to a new revision or a full new model.

In the process it applies any implied changes to the device: new required snaps, new kernel or gadget etc.

Snaps and assertions are downloaded from the store unless they are provided as local files specified by --snap and --assertion options. If using these options, it is expected that all the needed snaps and assertions are provided locally, otherwise the remodel will fail.

Usage: snap [OPTIONS] remodel [remodel-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

−−snap
Use one or more locally available snaps.

−−assertion
Use one or more locally available assertion files.

−−offline
Use only pre-installed and locally provided snaps and assertions. Providing any snaps or assertions locally implies --offline.

remove
Remove snaps from the system

The *remove* command removes the named snap instance from the system.

By default all the snap revisions are removed, including their data and the common data directory. When a --revision option is passed only the specified revision is removed.

Unless automatic snapshots are disabled, a snapshot of all data for the snap is saved upon removal, which is then available for future restoration with snap restore. The --purge option disables automatically creating snapshots.

Usage: snap [OPTIONS] remove [remove-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

−−revision
Remove only the given revision

−−purge
Remove the snap without saving a snapshot of its data

−−terminate
Terminate running processes associated with a snap before removal

remove-quota
Remove quota group

The *remove-quota* command removes the given quota group.

Currently, only quota groups with no sub-groups can be removed. In order to remove a quota group with sub-groups, the sub-groups must first be removed until there are no sub-groups for the group, then the group itself can be removed.

Usage: snap [OPTIONS] remove-quota [remove-quota-OPTIONS]

−−no-wait

restart
Restart services

The *restart* command restarts the given services.

If the --reload option is given, for each service whose app has a reload command, a reload is performed instead of a restart.

Usage: snap [OPTIONS] restart [restart-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

−−system
The operation should only affect system services.

−−user The operation should only affect user services for the current user.

−−users
If provided and set to 'all', the operation should affect services for all users.

−−reload
If the service has a reload command, use it instead of restarting.

restore
Restore a snapshot

The *restore* command replaces the current user, system and configuration data of included snaps, with the corresponding data from the specified snapshot.

By default, this command restores all the data in a snapshot. Alternatively, you can specify the data of which snaps to restore, or for which users, or a combination of these.

If a snap is included in a restore operation, excluding its system and configuration data from the restore is not currently possible. This restriction may be lifted in the future.

Usage: snap [OPTIONS] restore [restore-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

−−users
Restore data of only specific users (comma-separated) (default: all users)

revert
Reverts the given snap to the previous state

The *revert* command reverts the given snap to its state before the latest refresh. This will reactivate the

previous snap revision, and will use the original data that was associated with that revision, discarding any data changes that were done by the latest revision. As an exception, data which the snap explicitly chooses to share across revisions is not touched by the revert process.

Usage: snap [OPTIONS] revert [revert-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

−−devmode
Put snap in development mode and disable security confinement

−−jailmode
Put snap in enforced confinement mode

−−classic
Put snap in classic mode and disable security confinement

−−revision
Revert to the given revision

run
Run the given snap command

The *run* command executes the given snap command with the right confinement and environment.

Usage: snap [OPTIONS] run [run-OPTIONS] <NAME-OF-SNAP>.<NAME-OF-APP> [<SNAP-APP-ARG>...]

−−shell Run a shell instead of the command (useful for debugging)

−−debug-log
Enable debug logging during early snap startup phases

−−strace [=*"with-strace"*] <default: *"no-strace"*>
Run the command under strace (useful for debugging). Extra strace options can be specified as well here. Pass --raw to strace early snap helpers.

−−gdbserver [=*":0"*] <default: *"no-gdbserver"*>
Run the command with gdbserver

−−trace-exec
Display exec calls timing data

save
Save a snapshot of the current data

The *save* command creates a snapshot of the current user, system and configuration data for the given snaps.

By default, this command saves the data of all snaps for all users. Alternatively, you can specify the data of which snaps to save, or for which users, or a combination of these.

If a snap is included in a save operation, excluding its system and configuration data from the snapshot is not currently possible. This restriction may be lifted in the future.

Usage: snap [OPTIONS] save [save-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

−−abs-time
Display absolute times (in RFC 3339 format). Otherwise, display short relative times.

−−users
Snapshot data of only specific users (comma-separated) (default: all users)

saved
List currently stored snapshots

The *saved* command displays a list of snapshots that have been created previously with the 'save' command.

Usage: snap [OPTIONS] saved [saved-OPTIONS]

−−abs-time
Display absolute times (in RFC 3339 format). Otherwise, display short relative times.

−−id Show only a specific snapshot.

services
Query the status of services

The *services* command lists information about the services specified, or about the services in all currently installed snaps.

If executed as root user, the 'Startup' column of any user service will be whether it's globally enabled (i.e systemctl is-enabled). To view the actual 'Startup'|'Current' status of the user services for the root user itself, --user can be provided.

If executed as a non-root user, the 'Startup'|'Current' status of user services will be the current status for the invoking user. To view the global enablement status of user services, --global can be provided.

Usage: snap [OPTIONS] services [services-OPTIONS]

−g, **−−global**
Show the global enable status for user services instead of the status for the current user.

−u, **−−user**
Show the current status of the user services instead of the global enable status.

set
Change configuration options

The *set* command changes the provided configuration options as requested.

```
$ snap set snap-name username=frank password=$PASSWORD
```

All configuration changes are persisted at once, and only after the snap's configuration hook returns successfully.

Nested values may be modified via a dotted path:

```
$ snap set snap-name author.name=frank
```

Configuration option may be unset with exclamation mark:
```
$ snap set snap-name author!
```

Usage: snap [OPTIONS] set [set-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

−t Parse the value strictly as JSON document

−s Parse the value as a string

set-quota
Create or update a quota group.

The *set-quota* command updates or creates a quota group with the specified set of snaps.

A quota group sets resource limits on the set of snaps or snap services it contains. Snaps can be at most in one quota group but quota groups can be nested. Nested quota groups are subject to the restriction that the total sum of each existing quota in sub-groups cannot exceed that of the parent group the nested groups are part of.

All provided snaps are appended to the group; to remove a snap from a quota group, the entire group must be removed with remove-quota and recreated without the snap. To remove a sub-group from the quota group, the sub-group must be removed directly with the remove-quota command.

To set limits on individual services, one or more services can be placed into a sub-group. The respective snap for each service must belong to the sub-group's parent group. These sub-groups will have the same limitations as nested groups which means their combined resource usage cannot exceed the resource limits set for the parent group. Sub-groups which contain services cannot have their own journal quotas set, and instead automatically inherit any journal quota their parent quota group may have.

The memory limit for a quota group can be increased but not decreased. To decrease the memory limit for a quota group, the entire group must be removed with the remove-quota command and recreated with a lower limit. Increasing the memory limit for a quota group does not restart any services associated with snaps in the quota group.

The CPU limit for a quota group can be both increased and decreased after being set on a quota group. The CPU limit can be specified as a single percentage which means that the quota group is allowed an overall percentage of the CPU resources. Setting it to 50% means that the quota group is allowed to use up to 50% of all CPU cores in the allowed CPU set. Setting the percentage to 2x100% means that the quota group is allowed up to 100% on two cpu cores.

The CPU set limit for a quota group can be modified to include new cpus, or to remove existing cpus from the quota already set.

The threads limit for a quota group can be increased but not decreased. To decrease the threads limit for a quota group, the entire group must be removed with the remove-quota command and recreated with a lower limit.

The journal limits can be increased and decreased after being set on a group. Setting a journal limit will cause the snaps in the group to be put into the same journal namespace. This will affect the behaviour of the log command.

New quotas can be set on existing quota groups, but existing quotas cannot be removed from a quota group, without removing and recreating the entire group.

Adding new snaps to a quota group will result in all non-disabled services in that snap being restarted.

An existing sub group cannot be moved from one parent to another.

Usage: snap [OPTIONS] set-quota [set-quota-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

−−memory [=]
Memory quota

−−cpu [=]
CPU quota

−−cpu-set [=]
CPU set quota

−−threads [=]
Threads quota

−−journal-size [=]
Journal size quota

−−journal-rate-limit [=]
Journal rate limit as <message count>/<message period>

−−parent [=]
Parent quota group

sign
Sign an assertion

The *sign* command signs an assertion using the specified key, using the input for headers from a JSON mapping provided through stdin. The body of the assertion can be specified through a "body" pseudo-header.

Usage: snap [OPTIONS] sign [sign-OPTIONS]

−k <default: *"default"*>
Name of the key to use, otherwise use the default key

−−chain
Append the account and account-key assertions necessary to allow any device to validate the signed assertion.

−−update-timestamp
Update the output "timestamp" header to the current time

start
Start services

The *start* command starts, and optionally enables, the given services.

Usage: snap [OPTIONS] start [start-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

−−system
The operation should only affect system services.

−−user The operation should only affect user services for the current user.

−−users
If provided and set to 'all', the operation should affect services for all users.

−−enable
As well as starting the service now, arrange for it to be started on boot.

stop
Stop services

The *stop* command stops, and optionally disables, the given services.

Usage: snap [OPTIONS] stop [stop-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

−−system
The operation should only affect system services.

−−user The operation should only affect user services for the current user.

−−users
If provided and set to 'all', the operation should affect services for all users.

−−disable
As well as stopping the service now, arrange for it to no longer be started on boot.

switch
Switches snap to a different channel

The *switch* command switches the given snap to a different channel without doing a refresh. All available channels of a snap are listed in its 'snap info' output.

Usage: snap [OPTIONS] switch [switch-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

−−channel
Use this channel instead of stable

−−edge Install from the edge channel

−−beta Install from the beta channel

−−candidate
Install from the candidate channel

−−stable
Install from the stable channel

−−cohort
Switch the snap into the given cohort

−−leave-cohort
Switch the snap out of its cohort

tasks
List a change's tasks

The *tasks* command displays a summary of tasks associated with an individual change.

Usage: snap [OPTIONS] tasks [tasks-OPTIONS]

Aliases: change

−−abs-time
Display absolute times (in RFC 3339 format). Otherwise, display relative times up to 60 days, then YYYY-MM-DD.

−−last Select last change of given type (install, refresh, remove, try, auto-refresh, etc.). A question mark at the end of the type means to do nothing (instead of returning an error) if no change of the given type is found. Note the question mark could need protecting from the shell.

try
Test an unpacked snap in the system

The *try* command installs an unpacked snap into the system for testing purposes. The unpacked snap content continues to be used even after installation, so non-metadata changes there go live instantly. Metadata changes such as those performed in snap.yaml will require reinstallation to go live.

If snap-dir argument is omitted, the try command will attempt to infer it if either snapcraft.yaml file and prime directory or meta/snap.yaml file can be found relative to current working directory.

Usage: snap [OPTIONS] try [try-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

−−devmode
Put snap in development mode and disable security confinement

−−jailmode
Put snap in enforced confinement mode

−−classic
Put snap in classic mode and disable security confinement

unalias
Remove a manual alias, or the aliases for an entire snap

The *unalias* command removes a single alias if the provided argument is a manual alias, or disables all aliases of a snap, including manual ones, if the argument is a snap name.

Usage: snap [OPTIONS] unalias [unalias-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

unset
Remove configuration options

The *unset* command removes the provided configuration options as requested.

$ snap unset snap-name name address

All configuration changes are persisted at once, and only after the snap's configuration hook returns successfully.

Nested values may be removed via a dotted path:

$ snap unset snap-name user.name

Usage: snap [OPTIONS] unset [unset-OPTIONS]

−−no-wait
Do not wait for the operation to finish but just print the change id.

validate
List or apply validation sets

The *validate* command lists or applies validation sets that state which snaps are required or permitted to be installed together, optionally constrained to fixed revisions.

A validation set can either be in monitoring mode, in which case its constraints aren't enforced, or in enforcing mode, in which case snapd will not allow operations which would result in snaps breaking the validation set's constraints.

Usage: snap [OPTIONS] validate [validate-OPTIONS]

−−monitor
Monitor the given validations set

−−enforce
Enforce the given validation set

−−forget
Forget the given validation set

−−refresh
Refresh or install snaps to satisfy enforced validation sets

−−color <default: *"auto"*>
Use a little bit of color to highlight some things.

−−unicode <default: *"auto"*>
Use a little bit of Unicode to improve legibility.

−−no-wait
Do not wait for the operation to finish but just print the change id.

version
Show version details

The *version* command displays the versions of the running client, server, and operating system.

wait
Wait for configuration

The *wait* command waits until a configuration becomes true.

warnings
List warnings

The *warnings* command lists the warnings that have been reported to the system.

Once warnings have been listed with 'snap warnings', 'snap okay' may be used to silence them. A warning that's been silenced in this way will not be listed again unless it happens again, _and_ a cooldown time has passed.

Warnings expire automatically, and once expired they are forgotten.

Usage: snap [OPTIONS] warnings [warnings-OPTIONS]

−−abs-time
Display absolute times (in RFC 3339 format). Otherwise, display relative times up to 60 days, then YYYY-MM-DD.

−−unicode <default: *"auto"*>
Use a little bit of Unicode to improve legibility.

−−all Show all warnings

−−verbose
Show more information

watch
Watch a change in progress

The *watch* command waits for the given change-id to finish and shows progress (if available).

Usage: snap [OPTIONS] watch [watch-OPTIONS]

−−last Select last change of given type (install, refresh, remove, try, auto-refresh, etc.). A question mark at the end of the type means to do nothing (instead of returning an error) if no change of the given type is found. Note the question mark could need protecting from the shell.

whoami
Show the email the user is logged in with

The *whoami* command shows the email the user is logged in with.

NOTES

1. Online documentation
   https://docs.snapcraft.io

BUGS

Please report all bugs with *https://bugs.launchpad.net/snapd/+filebug*