

Backups and Undo

Snapshots, comparing them, and restoring tacctl's configuration · Checked against tacctl 0.2.22

Contents

- 1. What a snapshot is2
- 2. See what changed2
- 3. Restore or fix forward?3
- 4. Restore3
 - 4.1 Keeping the hosts' UIDs 4
- 5. Going back to an earlier tacctl release 4
- 6. Quick reference 6

Every tacctl change takes a snapshot of the configuration first. This page explains what a snapshot holds, how to see what changed, when to restore and when to fix forward, what to check after a restore, and what to know before going back to an earlier tacctl release. Restoring is a **superuser** task.

Example names used on this page Scope: `site-a` · User: `alice` · Hosts: `web1`, `db1` · Ticket `CHG-1182`

1. What a snapshot is

Question	Answer
When is one taken?	Automatically, before every change that would alter tacctl's configuration. Nothing is added when nothing would change.
What does it hold?	The store (users, groups, scopes) and <code>tacctl.yaml</code> (settings, including break-glass holds), plus the Linux host registry, host records and UID ledger
How many are kept?	The newest 30
What does it not do?	Touch any device or host. A restore changes tacctl's records only.

Snapshots contain shared secrets and password hashes. That is why `backup diff` and `backup restore` are superuser-only, and why each diff writes a `secret-read` audit line.

2. See what changed

```
tacctl backup list           # snapshots, newest first (operator tier and up)
tacctl backup diff          # what changed since the most recent snapshot
tacctl backup diff <timestamp> # ... since a specific one
```

To find the snapshot taken just before a given change, read the audit trail: each change's `cmd end` line names it as `snapshot=<id>` (*Audit and Compliance §2*).

```
journalctl -t tacctl --no-pager | grep 'cmd end' | grep 'user scope'
```

3. Restore or fix forward?

A restore puts back **everything** the snapshot holds, so it also undoes every change made after it.

Situation	Better choice
One wrong command, nothing changed since	Either. A targeted fix (<code>tacctl user scope alice remove site-b</code>) is often clearer in the audit trail.
Several related changes went wrong together	Restore the snapshot from before the first one
Others have made changes since the mistake	Fix forward with targeted commands, or you undo their work too
The configuration is broken and you don't know why	<code>tacctl config validate</code> first, then restore if needed

4. Restore

1. Preview.

```
tacctl backup list
tacctl backup diff <timestamp>    # what the restore would put back
```

2. Restore. It asks for confirmation.

```
tacctl backup restore <timestamp>
```

What happens:

- tacctl first snapshots the **current** state, so a restore can itself be undone with another restore.
- Both configuration files come back, every enabled backend is re-rendered and the backends restart.
- Nothing changes if the snapshot is invalid or the render fails.
- Break-glass holds come and go with the snapshot, and the restore names them when they change. The restore also shows any emergency lockdown that stands.
- Host files come back too. Before the question and after the restore, a note names the hosts that come back, go, or are registered differently, and every user whose UID would be dropped or changed.

3. Check.

```

tacctl status           # backends running?
tacctl config validate  # no new problems?
tacctl emergency lockdown status # lockdowns as you expect?
tacctl breakglass list  # holds as you expect?
tacctl host check --all # do the hosts still match?

```

No device or host is touched by a restore. If the restore changed users or scopes, run `tacctl host sync --all` so the Linux hosts follow. If it changed a scope's secret, the devices need the matching key again.

4.1 Keeping the hosts' UIDs

If the hosts have synced since the snapshot was taken, keep the live UID ledger, which is the one the hosts agree with:

```
tacctl backup restore <timestamp> --keep-uids
```

Users, groups, scopes and devices come back from the snapshot; restored users keep the UIDs they have now.

5. Going back to an earlier tacctl release

A restore undoes changes within the release you run. To run the release before it again, a superuser converts the state with `tacctl rollback` first, then installs that release:

```

tacctl rollback 0.2.21      # the dry run: its steps and warnings; writes nothing
tacctl rollback 0.2.21 --apply # convert the state (add --yes when it warns)
tacctl upgrade --branch 0.2.21 # install 0.2.21

```

`tacctl rollback` also takes 0.2.20, 0.2.19, 0.2.18, 0.2.17, 0.2.16, 0.2.9 and 0.2.3. It refuses, the dry run too and with nothing changed:

- while a Junos apply is not settled (PENDING or unknown);
- for every target but 0.2.21 and 0.2.20, while a user name has a period: the refusal lists each user with `tacctl user rename <name> <new-name>`;
- for every target but 0.2.21, 0.2.20 and 0.2.19, while an emergency lockdown stands.

The target 0.2.21 only removes the name that `tacctl server name set (server.name)` and the timing scale (`device timeout-scale`, `scope timeout-scale`, `config devices timeout-scale`), after a snapshot: 0.2.21 reads everything else this release writes. It has no warning and needs no `--yes`. Every other target removes them too, and the console shows the host name again. The plan names each device and key that loses a scale in a note, not a warning: the older release connects to every

device with its fixed timings again, which a slow or distant device may not meet. The values stay in the snapshot.

After a rollback, the older release's `config validate` reports the rendered config as out of date. The comment at the top of the generated tacquito drop-in names this release's commands (`server listen`, `server metrics`), so the older release's first render rewrites it, which restarts tacquito once: a brief interruption of TACACS+, RADIUS untouched. The rollback itself renders and restarts nothing.

The conversion takes a snapshot before it changes anything. Read the dry run's warnings before you add `--yes`:

The dry run warns of	What happens after the rollback	What to do first
Users who log in to the server and lack its scope (every target but 0.2.21)	The plan adds the scope of the server's own login (<code>tacctl</code>) to the scopes of each, and names them, because an older release answers that login only for users who have it. A warning, so <code>--apply</code> needs <code>--yes</code> .	Nothing: it is part of the rollback
Users who must change their password	The requirement is removed. Once the older release renders, those users (superusers aside) can log in to the devices with the password they were to change.	Have them change it, or keep this release
Scopes with a strict Junos management filter	The mode is removed. The older release plans the <code>ssh-only</code> filter, which accepts by default again.	Keep this release, or leave those scopes' filters out of the older release's plans
Scopes that record <code>root</code> as a break-glass account	The entry stays, but the older release renders a <code>system login user root</code> line that Junos refuses, and pulls read the <code>breakglass</code> section as differs	<code>tacctl scope breakglass <scope> remove root</code>
Users who would gain a system shell or the password cache	The older release reads the console's old per-tier and per-user switches again, and turns the cache on for every user of an allowed tier	<code>tacctl console password-cache tiers none</code> , and remove the switches the upgrading page names

Without a warning, the rollback also removes the console's colour setting and the per-user system-shell and password-cache choices, and writes a `list-max` that never asks as 1000.

Until the older release is installed, do not run `scope mgmt-acl <scope> mode strict`, `user add`, `user passwd <name>`, `user require-password-change`, `console color`, `console list-max`, `console system-shell user`, `console password-cache on`, `server name <label>`, `device timeout-scale`, `scope timeout-scale` or `config devices timeout-scale`, and for a target before 0.2.21 not `server addresses add|replace` or `host enroll --local` either: each writes a setting the older release refuses. If one did, run the rollback with `--apply` again. The same applies, for a target before 0.2.21, to a user added, or given other scopes, in the meantime: the older release lets that user log in to the server only with the server's scope among their own, so run the rollback with `--apply --yes` again. A script that calls `server listen` or `server metrics` does not run on 0.2.21, which has those two under `config`.

Every step, per target, is in the repository's [Rolling back](#).

6. Quick reference

Task	Command	Who
List snapshots	<code>tacctl backup list</code>	Operator
What changed since the latest	<code>tacctl backup diff</code>	Superuser
What changed since one	<code>tacctl backup diff <ts></code>	Superuser
Restore	<code>tacctl backup restore <ts></code>	Superuser
Restore, keep host UIDs	<code>tacctl backup restore <ts> --keep-uids</code>	Superuser
Find the snapshot before a change	<code>journalctl -t tacctl grep 'cmd end'</code>	Superuser
Prepare the release before this one (dry run)	<code>tacctl rollback 0.2.21</code>	Superuser

See also: *Incident Response Runbooks R8* · *Audit and Compliance* · [Maintenance commands \(repo\)](#) · `man tacctl`