Backups and Undo

Snapshots, comparing them, and restoring tacctl's configuration · Checked against tacctl 0.2.25

Contents

1. What a snapshot is 2

2. See what changed 2

3. Restore or fix forward? 3

4. Restore 3

- 4.1 Keeping the hosts' UIDs 4
- 4.2 Keeping a snapshot beyond the retention 4
- 4.3 Putting back only what a rollback dropped 4

5. Going back to an earlier tacctl release 5

6. Quick reference 7

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 `tacctl.yaml` (settings, including break-glass holds), plus the Linux host registry, host records and UID ledger
How many are kept?	The newest 30. `backup keep` copies one where no retention reaches it (§4.2).
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`, `backup restore`, `backup keep`, `backup unkeep` and `backup recover` 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 (`tacctl user scope alice remove site-b`) 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	`tacctl config validate` 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, and the break-glass firewall is loaded again for the holds and the setting the snapshot brought back (*Break Glass* §2.1). 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.

4.2 Keeping a snapshot beyond the retention

The newest 30 snapshots are kept and older ones are pruned. A snapshot you will need later (before a risky change, or the one a rollback names) can be copied to a place no retention reaches:

```
tacctl backup keep <timestamp> --note "before INC-4821 cleanup"
tacctl backup list                   # a kept snapshot reads "snapshot, kept", with its
note
tacctl backup unkeep <timestamp>     # remove the kept copy (asks first; --yes skips the
question)
```

- The copy is in `/etc/tacctl/backups-kept/<timestamp>/` and is a snapshot like any other: `backup diff`, `backup restore` and `backup recover` find it there, also when the original has been pruned (`backup list` says `kept only`).
- The note is free text of at most 200 printable characters. It is shown by `backup list` and written as `<redacted>` in the audit line, but it is not a place for secrets.
- `backup unkeep` says first when the kept copy is the only copy.

4.3 Putting back only what a rollback dropped

`backup restore` puts back **everything**. After a rollback to an earlier release and a new upgrade, use `backup recover` to put back only what the rollback removed, and nothing else:

```
tacctl backup recover <snapshot> --dry-run                # what would come back
tacctl backup recover <snapshot> --only email,snmp-users  # asks first; --yes skips the
question
```

- `email` puts back the users' email addresses and `snmp-users` the SNMPv3 users. Without `--only` it recovers both.
- It adds, and never overwrites: a user who has another address now keeps it, an address another user holds is skipped, a default user already chosen stays, and a scope, device or user that no longer exists is skipped. Each skipped item is named; addresses and passphrases never are.
- It takes a snapshot of the current state first. Nothing is rendered or restarted.
- The rollback names the snapshot to use, and keeps it for you (§4.2). Remove the kept copy with `backup unkeep` once the data is back.

---

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 0.2.24            # the dry run: its steps and warnings; writes nothing
tacctl rollback 0.2.24 --apply    # convert the state (add --yes when it warns)
tacctl upgrade --branch 0.2.24    # install 0.2.24
```

`tacctl rollback` also takes `0.2.23`, `0.2.22`, `0.2.21`, `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 `0.2.24`, `0.2.23`, `0.2.22`, `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 `0.2.24`, `0.2.23`, `0.2.22`, `0.2.21`, `0.2.20` and `0.2.19`, while an emergency lockdown locks a group;
- for every target but `0.2.24` and `0.2.23`, while an emergency lockdown has enrolled hosts, or a lifted one still waits for a host: an older release cannot lift sshd's lockdown drop-in on a host. The refusal names the hosts: lift the lockdown, then `tacctl host sync <hosts>`.

The target `0.2.24` converts nothing, removes no key and takes no snapshot, and needs no `--yes`: this release changed no stored file that 0.2.24 reads strictly. The break-glass firewall and the other things 0.2.24 has stay. After that upgrade, a Junos device that names its NETCONF capabilities with the XML namespace's prefix is recorded as `no hello` again by its next pull, and an apply to a virtual chassis whose master prints its own label is rolled back again, as in 0.2.24.

The target `0.2.23` converts nothing. What 0.2.24 added to the stored files is one key, the setting of the break-glass firewall, and the rollback removes it (after a snapshot, when the setting is set) together with the nftables table, its rules file and its boot unit (every target but `0.2.24` does; the `nftables`

package stays). It has no warning and needs no `--yes`. It also puts back 0.2.23's sudoers drop-in, without the operator's `log failures *`, when the file is exactly the text 0.2.24 wrote.

> **A hold that stands stays, without the firewall.** The older release's server completes the TCP handshake of the held address again, so a device that falls back to its local accounts only when the server is unreachable will not. The dry run names the holds in a note. See *Break Glass* §2.1.

The target `0.2.22` also removes the users' email addresses and writes the SNMP credential files in version 1 again (the older releases read only that version), after a snapshot that it **keeps** (§4.2) with a note saying what was dropped. Version 1 holds one SNMPv3 user per level: the default stays and the others are dropped, and a level with several users and none chosen as default is refused, naming `v3-user default <user>`. The dry run names the users with an address and counts the SNMP users, never an address or a passphrase, in a warning, so `--apply` needs `--yes`. It names the snapshot and the command that puts the data back once this release is installed again (§4.3).

The target `0.2.21` 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, and everything the target `0.2.22` removes. Every older 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 `0.2.23`, `0.2.22` and `0.2.21`)	The plan adds the scope of the server's own login (`tacctl`) to the scopes of each, and names them, because an older release answers that login only for users who have it. A warning, so `--apply` needs `--yes`.	Nothing: it is part of the rollback
Users with an email address, or SNMPv3 users beyond one per level (every target but `0.2.23`)	They are removed, and the rollback keeps the snapshot that holds them	After the upgrade, `tacctl backup recover <snapshot>` (§4.3)
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

The dry run warns of	What happens after the rollback	What to do first
Scopes with a strict Junos management filter	The mode is removed. The older release plans the `ssh-only` filter, which accepts by default again.	Keep this release, or leave those scopes' filters out of the older release's plans
Scopes that record `root` as a break-glass account	The entry stays, but the older release renders a `system login user root` line that Junos refuses, and pulls read the `breakglass` section as `differs`	`tacctl scope breakglass <scope> remove root`
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	`tacctl console password-cache tiers none`, 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 `config breakglass-firewall`, `user email`, `user add --email`, a second or non-default SNMPv3 user (`v3-user add`), `breakglass hold` or `release`, `device address` of a held device, `backup restore`, `server listen`, `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	`tacctl backup list`	Operator
What changed since the latest	`tacctl backup diff`	Superuser
What changed since one	`tacctl backup diff <ts>`	Superuser
Restore	`tacctl backup restore <ts>`	Superuser

Task	Command	Who
Restore, keep host UIDs	`tacctl backup restore <ts> --keep-uids`	Superuser
Keep a snapshot beyond the retention	`tacctl backup keep <ts> [--note "<text>"]`	Superuser
Remove the kept copy	`tacctl backup unkeep <ts>`	Superuser
Put back only email addresses or SNMPv3 users	`tacctl backup recover <ts> [--only email,snmp-users]`	Superuser
Find the snapshot before a change	`journalctl -t tacctl | grep 'cmd end'`	Superuser
Prepare the release before this one (dry run)	`tacctl rollback 0.2.24`	Superuser

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