Connecting to Devices

tacctl ssh by name, plain ssh with pinned keys, and what to do when a connection is refused · Checked against tacctl 0.2.22

Contents

1. Find the device 2
2. Connect with tacctl ssh 2
3. Plain ssh and scp 3
4. When ssh refuses 3
5. A host key changed 4
6. Keys are pinned when a device is registered 5
7. Slow or distant devices 6
8. Quick reference 7

Every tacctl user can open an ssh session to the devices and hosts of their own scopes by name. This page covers connecting, using plain `ssh` and `scp` with the same pinned host keys, and the refusals you may meet, including the one that matters most: a changed host key.

> **Example names used on this page** Server: `tacacs.example.net` · Scopes: `site-a`, `site-b` · Devices: `sw1` (Cisco), `rtr1` (Junos), `sw7` (IOS 12.4) · Host: `web1` · You: `alice`

1. Find the device

```
tacctl device list                      # the devices of your scopes
tacctl device list core                 # a bare word matches name, address, vendor or
scope
tacctl device list --scope site-a --vendor juniper
tacctl device show sw1                  # one device in full: scope, state, last seen,
keys
```

A device must be **configured** (a scope covers its address) for anyone to log in to it. Enrolled Linux hosts are in `tacctl host list` (engineer tier); `tacctl ssh <name>` takes a host's name as well.

---

2. Connect with `tacctl ssh`

```
tacctl ssh sw1                          # interactive session, as you
tacctl ssh rtr1 -- show chassis alarms  # one remote command, then back
tacctl ssh 10.10.0.1                    # by registered address
```

In the console or `tacctl shell`, type just `ssh sw1`. `exit` on the device brings you back.

How the login works:

- ssh runs **as you**, with your own tacctl username and password. No agent, no key, no other login. As root it refuses.
- Only an active tacctl user whose scopes include the device's scope may connect, at every tier. A refusal is logged (`ssh DENY`).
- ssh accepts only the host keys pinned for that device in the registry.
- It needs a terminal.

Many logins in a row? Turn on the password cache, where your tier is allowed it, and type your password once per session (*Getting Started* §1.3):

```
tacacs> console password-cache on   # once: saved for your later sessions too
tacacs> ssh sw1                     # asks for your password, once, before it connects
tacacs> ssh rtr1                    # asks nothing
tacacs> console forget              # drop the cached password now
```

In a system shell, `tacctl shell --password-cache` does the same for one shell session. The ssh audit line ends `password=asked` for the line that prompted and `password=cached` for a line that used the cache.

---

3. Plain `ssh` and `scp`

To use your own `ssh` and `scp` with the same options and the same pinned keys:

```
tacctl device client-config > ~/.ssh/tacctl.conf
# then put   Include ~/.ssh/tacctl.conf   at the top of ~/.ssh/config
ssh sw1
```

- The file has one `Host` block per device you may see.
- It has no `User` line, so ssh logs in with your **local** username. If your local account has another name than your tacctl user, add `User <tacctl name>` to your own `~/.ssh/config`.
- Run it again whenever the registry changes.

---

4. When ssh refuses

Message or situation	What it means	What to do	Who
Device not found	Not registered, or not in your scopes	Check `tacctl device list`. Ask an engineer to register it, or a superuser to grant the scope.	E / S
Device `unconfigured`	No scope covers its address	A superuser adds the prefix to a scope	S
Pinned and offered fingerprints printed	The device's host key changed: replaced, reset, upgraded, or something is wrong	**Stop.** Follow §5. Never bypass the pin.	E
Unpinned device refused for a pull	No key is pinned	Check the key on the console, then `tacctl device hostkey <name> accept`	E
Warning that break-glass is held	The server is deliberately not answering this device	Your tacctl login will fail there. `tacctl device show <name>` says who holds it and why.	—

Message or situation	What it means	What to do	Who
Red `EMERGENCY LOCKDOWN` line	Your group is locked out of the scope	Expected during an incident. Contact the superuser named in the line.	—
Refused as root	ssh always runs as you	Run it from your own account	—
Old IOS: ssh negotiation fails	The device offers only legacy algorithms	`tacctl device legacy-ssh <name> enable`	E
`No ssh host key could be read from <address> port <port>`	The device gave no key to any read, or did not answer	§6: read the key by hand (`ssh-keyscan -T 30 ...`) and pin its fingerprint	E
Password refused	Wrong password, disabled account, or missing scope	*Troubleshooting* §2	O
`device check` says unreachable	The server has no route to the management port (may be a false alarm)	Test from your workstation too	O

---

5. A host key changed

`tacctl ssh` and pulls refuse a device whose key no longer matches its pin, and print the pinned and offered fingerprints. A `hostkey-changed` notice is raised, and it cannot be acknowledged away.

1. Assume it is hostile until you have checked it. Was the device replaced, reset or upgraded? Check the change record.

2. Read the key on the device's console, not over the network:

Device	Console command
Cisco	`show ip ssh`
Junos	`file show /etc/ssh/ssh_host_ed25519_key.pub`
Linux	`ssh-keygen -lf /etc/ssh/ssh_host_ed25519_key.pub`

3. Only if it matches, re-pin (engineer for your own scopes, or superuser):

```
tacctl device hostkey sw1                    # show the pinned fingerprints
tacctl device hostkey sw1 set SHA256:<fp>    # pin exactly the key you verified
(preferred)
tacctl device hostkey sw1 accept             # or: re-scan and pin every offered key
(confirms)
```

4. If it does not match, treat it as a possible interception. Don't connect, and escalate to security (*Incident Response Runbooks* R6).

Operators: report a changed key to an engineer. Do not try to fix it.

---

6. Keys are pinned when a device is registered

`device add`, `device enroll` and `device onboard` pin the device's host keys when they register it, and show the fingerprints with the console command that displays them. **Compare them with the console before you trust them.** `enroll` and `onboard` ask you to accept the key by typing the device's name back. The registration steps are in *Engineer Guide: Device Changes* §2.

The keys are read with one `ssh-keyscan` for every key type. What happens next depends on what it printed:

`ssh-keyscan`	Next read
Keys	None: they are pinned
No key, but the device accepted the connection	Asked again one key type at a time (ed25519, ecdsa, rsa), then tacctl's own ssh client
No key and no message (a server slow to greet, a refused or filtered address)	tacctl's own ssh client, once, without the one-type retry
`getaddrinfo ...: Name or service not known`	None: the name does not resolve

tacctl's own ssh client stops at the host key, before any login, and sends no password. Its connect waits 3 seconds and its handshake 10, so a refused device costs nothing more, a filtered or powered-off address costs 3 seconds more per scan, and a device that is slow to greet can take several seconds because the key types are read one after the other. A `legacy-ssh` device, `device enroll` and `device collect` are read as before. For a slow or distant device the timing scale (§7) multiplies these waits.

When no read gives a key, the command changes nothing. The refusal names the reads that were made, then the command that shows a key by hand and the ways to pin a fingerprint you have checked. For a device that `ssh-keyscan` read nothing from, it names `ssh-keyscan, and tacctl's own ssh client` and gives the key command with more time to greet (`-T 30`):

```
# read the key yourself (also -t ecdsa, -t rsa; -p <port> for another port)
ssh-keyscan -T 30 -t ed25519 10.10.0.2
# register the device with the fingerprint you checked on its console
tacctl device add rtr1 10.10.0.2 --vendor juniper --host-key SHA256:<fp>
# or pin it on a device that is registered already
tacctl device hostkey rtr1 set SHA256:<fp>
```

---

7. Slow or distant devices

A device behind a satellite circuit (a round trip of about 600 ms) or with a slow CPU (one measured needs about 4 s for each ssh connection) can miss the timings tacctl uses to reach a device. One whole number from 1 to 10, the **timing scale**, multiplies all of them. 1, the default, is every timing as it has always been.

A superuser sets it, for one device, for a scope or for the whole server:

```
tacctl device timeout-scale sw1 3          # this device
tacctl scope timeout-scale site-a 3        # every device of the scope
tacctl config devices timeout-scale 2      # every device of the server
tacctl device timeout-scale sw1 clear      # back to the scope's, then the server's
tacctl device timeout-scale sw1            # the scale in force and where it comes from
```

The scale in force for a device is its own, else its scope's, else the server's, else 1. A verb with no value prints the scale in force, where it comes from (the device, the scope, the server or built in) and, above 1, what it makes of each timing. Engineers read the scale in `tacctl device show` (a `Timeout scale` row) and `tacctl scope show`; both show it only when it is not 1.

What	At scale 1	Multiplied
Connect, handshake and first prompt of an ssh or NETCONF session (`device config pull`, `diff --pull`, `plan --pull`, `apply`, `enroll`, `onboard`, `collect`, `support-bundle`)	10 s	Yes
The ConnectTimeout of `tacctl ssh`	10 s	Yes
The host-key scans (`device add`, `hostkey`, `scan`, `check`, `enroll`, `collect`, `onboard`, and the diagnosis after a failed `ssh`)	5 s and 3 s for ssh-keyscan; 3 s, 10 s and 5 s for tacctl's own client	Yes
The SNMP timeout (`snmp.timeout`, `scope snmp timeout`, `device snmp timeout`)	2 s	Yes, never above 10 s
The time one device may take in a pull or an apply (`device.config.timeout`)	90 s	Yes, kept between 10 s and 600 s
The confirmed-commit windows (`device.config.confirm_timeout`, `device.apply.confirm_timeout`)	Minutes	No: they are timers on the device

An option typed on the command line wins: `--timeout` of a pull or an apply is the time of every device of that run, as typed, and the scale replaces only the default. When one run covers devices with different scales, each device has the time its own scale gives. A scale cannot make a lost packet arrive: a device that is down is still found out, only later. To see where the time of a run goes, use `--timing` (*Engineer Guide* §3).

8. Quick reference

Task	Command	Who
Find a device	`tacctl device list <text>`	Everyone
Connect	`tacctl ssh <name>`	Everyone (own scopes)
One command	`tacctl ssh <name> -- <command>`	Everyone
Plain ssh config	`tacctl device client-config > ~/.ssh/tacctl.conf`	Everyone
Show pinned keys	`tacctl device hostkey <name>`	Engineer
Re-pin a verified key	`tacctl device hostkey <name> set SHA256:<fp>`	Engineer
Legacy IOS ssh	`tacctl device legacy-ssh <name> enable`	Engineer
Slow or distant device	`tacctl device timeout-scale <name> <1-10>`	Superuser

See also: Operator Field Guide · Troubleshooting · Host-key pinning (repo) · `man tacctl`