A FUNCTIONAL LAYER FOR DEBIAN



Stratum

Handbook

The complete guide to installing, using and supporting Stratum — for end users, administrators and the support team.

END USER · ADMINISTRATOR · SUPPORT TEAM

Stratum Project

www.skraba.net/stratum

Stratum 0.45 · Debian 13 (trixie) · Debian 14 (forky) · First edition · November 2026

STRATUM

Stratum Handbook

A functional layer for Debian — installation, everyday use, modules, licensing and remote support, explained step by step.

Stratum Project

Stratum 0.45 · First edition · November 2026

Stratum Handbook
First edition · November 2026 · describes Stratum 0.45 and the Stratum modules published in the Stratum repository in November 2026.

© 2026 Stratum Project. All rights reserved. Stratum packages are free software under the GNU General Public License, version 3 or later.

Debian is a registered trademark of Software in the Public Interest, Inc. Docker is a trademark of Docker, Inc. Ansible is a trademark of Red Hat, Inc. MariaDB is a trademark of MariaDB plc. WireGuard is a registered trademark of Jason A. Donenfeld. Other names may be trademarks of their owners. Stratum is an independent project and is not affiliated with these organisations.

Screenshots were taken on the test systems `trixie-build` (Debian 13) and `forky-build` (Debian 14). Host names, addresses, keys and license data shown in this book are examples.

Project page and repository: **https://www.skraba.net/stratum/start** · **https://www.skraba.net/stratum-apt/**

Contents

● End user ● Administrator ● Support team

PART I · GETTING STARTED

1 About this book ● ● ● 9
1.1 Who this book is for 9
1.2 Icons used in this book 9
1.3 Conventions 9
1.4 How this book is organised 10

2 What is Stratum? ● ● ● 11
2.1 Stratum is not a distribution 11
2.2 Principles 11
2.3 The modules at a glance 12

3 Installing Stratum ● 13
3.1 What you need 13
3.2 The Stratum repository 13
3.3 Quick setup 13
3.4 Manual setup, step by step 14
3.5 Checking the signing key 14
3.6 First check: stratum status 14
3.7 Adding modules 15
3.8 Removing Stratum 15

4 The stratum command and the menu ● ● 17
4.1 Starting the menu 17
4.2 Inside a module 18
4.3 Installing and removing modules 19
4.4 The about screen 20
4.5 Commands 20
4.6 Options, settings and environment 21

PART II · MODULES

5 How modules work ● ● 25
5.1 Three types of module 25
5.2 The common actions 25
5.3 Exit codes 25
5.4 First start (setup) 26

6 Software — stratum software ● ● 27
6.1 What you get 27
6.2 In the menu ● ● 27
6.3 Automatic upgrades 28
6.4 Commands 29
6.5 Adding your own repository 30

7 Login banner — stratum banner ● ● 33
7.1 Profiles 33
7.2 Commands and files 34

8 Memory — stratum memory ● ● 35
8.1 How it works 35
8.2 In the menu ● 35
8.3 Types of computer 36
8.4 Commands 37
8.5 Ballooning and the minimum memory 38

9 Firewall — stratum firewall ● 39
9.1 The idea in one picture 39
9.2 Safety first: you will not lock yourself out 39
9.3 In the menu ● ● 40
9.4 Settings 41
9.5 Rules 41
9.6 Docker and the firewall 42

9.7 Rules of other modules ... 42
9.8 Events ... 43
9.9 All commands ... 43

10 Security — stratum security ... **45**
10.1 What it does ... 45
10.2 Settings ... 45
10.3 Protected services ... 46
10.4 Hardening ... 46
10.5 Whitelist and bans ... 47
10.6 Events — what happened today? ... 47
10.7 All commands ... 48

11 Automation — stratum ansible ... **51**
11.1 Three roles ... 51
11.2 Managed node ... 52
11.3 Pull ... 52
11.4 Controller ... 53
11.5 All commands ... 54

12 Self-healing — stratum watchdog ... **55**
12.1 Two layers ... 55
12.2 In the menu ... 56
12.3 What is checked ... 57
12.4 Examples ... 58
12.5 Settings ... 59
12.6 Own checks ... 59
12.7 All commands ... 59

13 File server — stratum fileserver ... **61**
13.1 The idea ... 61
13.2 In the menu ... 61
13.3 A shared folder in five commands ... 62
13.4 Share settings ... 63
13.5 Settings and commands ... 63

14 Database — stratum database ... **65**
14.1 In the menu ... 65
14.2 A database for an application ... 66
14.3 Backups ... 67
14.4 Settings and commands ... 67

15 WireGuard mesh — stratum wireguard ... **69**
15.1 How it works ... 69
15.2 A mesh in three commands ... 69
15.3 In the menu ... 70
15.4 Networks behind a member ... 72
15.5 Firewall and names ... 72
15.6 Settings and commands ... 72

PART III · LICENSE AND SUPPORT

16 Your Stratum license ... **77**
16.1 The license in four steps ... 77
16.2 Step by step ... 77
16.3 Checking your license ... 78
16.4 The hardware ID ... 79
16.5 Time and the license ... 79
16.6 All commands ... 80

17 Asking for remote support ... **81**
17.1 Before you start ... 81
17.2 Open a session ... 81
17.3 While support is working ... 82
17.4 End the session ... 82
17.5 Sending a diagnostic report ... 82
17.6 What exactly happens — for the curious ... 83

18 Giving remote support — for the support team ... **85**
18.1 Where things live ... 85

18.2 Issuing a license ... 85
18.3 Preparing your workstation (once) ... 86
18.4 Connecting to a session ... 87
18.5 Conduct during a session ... 87
18.6 Analysing a report ... 87
18.7 Revoking a license ... 87
18.8 Checklists ... 88

PART IV · REFERENCE

19 The project and its repository ● ● ... 91
19.1 From source to your computer ... 91
19.2 Building and publishing ● ... 91
19.3 Versions and rules of the project ... 92
19.4 Packages in the repository ... 92
19.5 The signing key ... 92

20 Command reference ● ● ● ... 95
20.1 The stratum command ... 95
20.2 Modules ... 95
20.3 Exit codes ... 96

21 Files and directories ● ● ... 97

22 Troubleshooting and FAQ ● ● ● ... 99

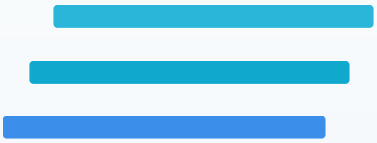
23 Glossary ● ● ● ... 101

24 Quick reference card ● ● ● ... 103

PART I

Getting started

What Stratum is, how this book works, how to install Stratum and how to find your way around the stratum command and its menu.



CHAPTER 1

About this book

END USER ADMINISTRATOR SUPPORT TEAM

Stratum turns a plain Debian installation into a well-equipped, easy-to-manage system — without changing what Debian is. This book shows you how, one step at a time. You do not need to be a Linux expert: if you can open a terminal and type a command, you can use Stratum.

1.1 Who this book is for

Every chapter and section carries one or more coloured labels. They tell you at a glance whether a part of the book is meant for you:

Label	Who	Typical tasks
END USER	The person who uses a Stratum computer every day and occasionally changes a setting in the menu.	Open the menu, check the state, choose a profile, ask for remote support.
ADMINISTRATOR	The person responsible for one or more Stratum computers: installs modules, sets up the firewall, Ansible and the other modules.	Install Stratum, configure modules, read logs and events, automate.
SUPPORT TEAM	Stratum support: issues licenses, runs the relay and gives remote support to customers.	Issue and revoke licenses, connect to a support session, analyse reports.

You may happily read parts that are not labelled for you — nothing in this book is secret — but you can also skip them safely.

1.2 Icons used in this book

TIP
A shortcut or a good habit that saves time.

REMEMBER
Something worth keeping in mind — it will come up again.

WARNING
Read this before you continue: a step here can lock you out, delete data or interrupt a service.

TECHNICAL STUFF
Background for the curious. You can skip it and still use Stratum perfectly well.

1.3 Conventions

- Commands are shown in terminal boxes. A line starting with `#` is typed as **root** (or with `sudo`); a line starting with `$` works as a normal user. Do not type the `#` or `$` itself.
- Output follows the command. Long output is shortened with `…`.

- Names of menu rows and buttons look like this: **Apply**, **Back**. Keys look like this: `Enter`, `Esc`.
- Placeholders are written in angle brackets: `stratum <module> status` means, for example, `stratum memory status`.

```
# stratum memory status         # typed as root
State:          running
Boot:           enabled
…
```

1.4 How this book is organised

Part I · Getting started

What Stratum is, installation, the `stratum` command and the menu.

Part II · Modules

One chapter per module: software (repositories and automatic upgrades), banner, memory, firewall, security, Ansible, watchdog, file server, database and the WireGuard mesh.

Part III · License and support

Getting a license, asking for remote support — and how the support team gives it.

Part IV · Reference

The project and its repository, all commands, files, troubleshooting and a glossary.

© 2026 Stratum Project

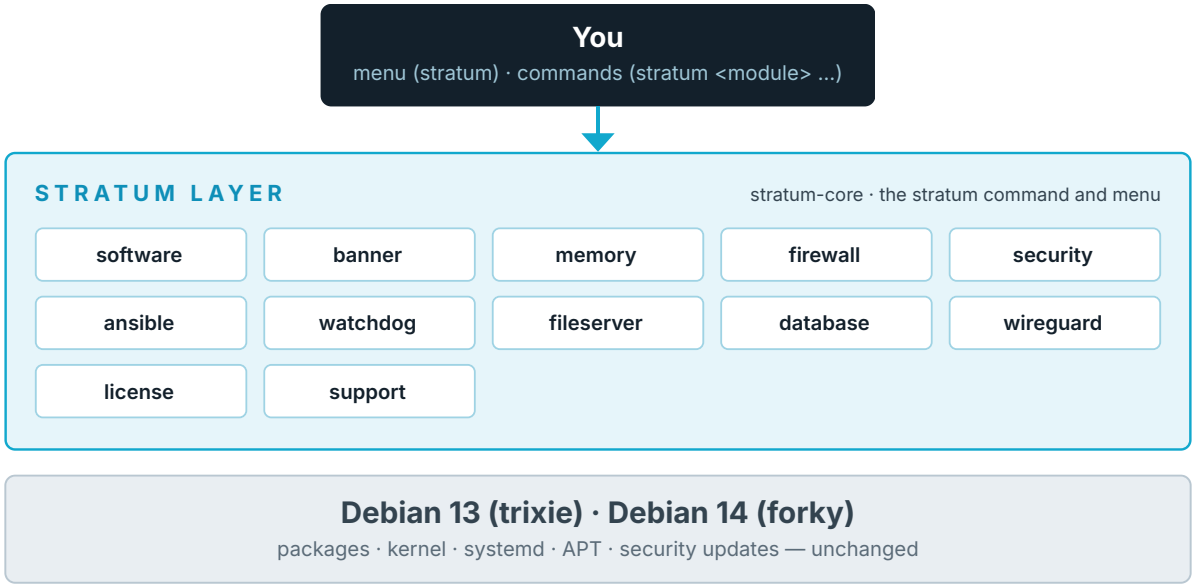
CHAPTER 2

What is Stratum?

END USER ADMINISTRATOR SUPPORT TEAM

In geology, a *stratum* is a distinct layer within a larger structure: part of the whole, yet defined by its own properties. Stratum applies the same idea to Debian.

Stratum is an extension layer on top of a standard Debian installation. It adds administration tools, sensible standard configurations, security functions, container management and automation — packaged as small, independent modules. Debian's packages, its stability and its established ways of working stay exactly as they are. The Stratum core and its modules are written in Go: a single, statically linked `stratum` command with nothing extra to pull in at runtime.



Stratum sits between you and Debian: one command and one menu for all modules.

2.1 Stratum is not a distribution

Stratum does not fork or replace Debian. It is installed from its own signed APT repository like any other software, and it can be removed again. Every Stratum module is a normal Debian package called `stratum-<name>`. Under the hood, the modules use the tools Debian already provides — `ufw`, `fail2ban`, `systemd-zram-generator`, `unattended-upgrades`, `ansible`, Samba, NFS, MariaDB, WireGuard. Stratum configures them and gives you one consistent way to manage them.

> **REMEMBER**
> Stratum is not a distribution. It is a layer. Your system remains Debian.

2.2 Principles

Principle	What it means for you
Modular	Install only what a computer needs. Each module can be added or removed on its own — in the menu under **modules**.
One way to manage	Every module works the same way: `stratum <module> status`, `start`, `stop`, `enable`, `disable`, `settings`, `help` — and the same menu.

Principle	What it means for you
Types of computer	Modules offer ready-made defaults for a *workstation*, *server*, *hypervisor* or *vm-guest*. Choose the type, adjust what you like.
Safe changes	Changes that could cut your connection (firewall, SSH) are checked first, must be confirmed and are undone automatically if you do not confirm them.
Transparent	Stratum writes only its own, clearly named files and leaves your own configuration alone. Every setting is visible with `status`.
Interruption-proof	Stratum commands ignore `Ctrl`+`C`, so a change is never left half done.

2.3 The modules at a glance

Package	Module	Type	In one sentence
`stratum-core`	stratum	setting	The `stratum` command, the menu and the module framework (written in Go).
`stratum-banner`	banner	profile	The Stratum logo and line on the login screen.
`stratum-software`	software	setting	APT repositories, each switched on or off, and automatic daily upgrades.
`stratum-memory`	memory	service	Fast compressed swap in RAM, deduplication and ballooning of virtual machines.
`stratum-firewall`	firewall	service	Host firewall with ufw, also for Docker ports.
`stratum-security`	security	service	fail2ban against password guessing, SSH and system hardening.
`stratum-fileserver`	fileserver	service	Shared folders with Samba and NFS, users, groups and permissions.
`stratum-database`	database	service	MariaDB server: databases, users, permissions and backups.
`stratum-wireguard`	wireguard	service	A WireGuard mesh network between computers in different places.
`stratum-ansible`	ansible	setting	Ansible: managed node, pull and controller.
`stratum-license`	license	setting	Hardware ID and the Stratum license.
`stratum-support`	support	service	Remote support sessions and diagnostic reports.
`stratum-watchdog`	watchdog	service	Checks the key parts of the system and restarts what has failed.

Also in the repository: `stratum-archive-keyring` (the signing key) and `stratum-handbook` (this book as PDF, linked as `/root/stratum-handbook.pdf`).

© 2026 Stratum Project

CHAPTER 3

Installing Stratum

ADMINISTRATOR

Installing Stratum takes about a minute: add the Stratum repository to APT, install `stratum-core`, done. Everything else is added later, module by module.

3.1 What you need

- A computer, virtual machine or container with **Debian 13 (trixie)** or **Debian 14 (forky, currently testing)**, architecture **amd64** or **arm64**.
- Access as **root** (or a user with `sudo`).
- Internet access to `https://www.skraba.net/stratum-apt/`.

> **REMEMBER**
> The supported releases are listed in `/usr/lib/stratum/supported`. On another system — an older Debian, Ubuntu or Raspberry Pi OS — Stratum still starts, but shows **unsupported configuration**, and the modules that manage APT repositories and upgrades refuse to work.

3.2 The Stratum repository

Item	Value
Address	`https://www.skraba.net/stratum-apt/`
Suites	`trixie` (Debian 13) and `forky` (Debian 14) — the same packages
Components	`main` (Stratum and its modules), `contrib` and `non-free` (further software of the Stratum project)
Architectures	`amd64`, `arm64` (and source packages)
Signing key	Stratum Archive Signing Key <stratum@skraba.net>, ed25519, valid until 2031-10-20
Fingerprint	`4F39 6C32 8AB5 D14E F653 55CF 130D EB73 9D1E 1A2C`

3.3 Quick setup

Copy the following block into a root terminal. It downloads the signing key, **verifies its checksum**, installs it, writes the repository for your Debian release and installs `stratum-core`:

```
# apt-get update && apt-get install -y curl ca-certificates
# curl -fsSL https://www.skraba.net/stratum-apt/stratum-archive-keyring.gpg -o /tmp/stratum-archive-keyring.gpg
# echo "4e9b1cb45f58d955340ac93b6c8d997d900d9780d005f002ce978bab39d53647  /tmp/stratum-archive-keyring.gpg" |
sha256sum -c - \
    && install -m 0644 /tmp/stratum-archive-keyring.gpg /usr/share/keyrings/stratum-archive-keyring.gpg \
    && . /etc/os-release \
    && cat > /etc/apt/sources.list.d/stratum.sources <<EOF && apt-get update && apt-get install -y stratum-core
Types: deb
URIs: https://www.skraba.net/stratum-apt/
Suites: $VERSION_CODENAME
Components: main contrib non-free
```

```
Signed-By: /usr/share/keyrings/stratum-archive-keyring.gpg
EOF
```

> **WARNING**
>
> If `sha256sum` prints **FAILED**, nothing is installed. Stop and check the signing key (section 3.5) — a wrong checksum means the key you downloaded is not the Stratum key.

3.4 Manual setup, step by step

1. **Download the signing key**

```
# curl -fsSL https://www.skraba.net/stratum-apt/stratum-archive-keyring.gpg \
    -o /usr/share/keyrings/stratum-archive-keyring.gpg
```

2. **Create the repository source** — `Suites` is the codename of your release (`VERSION_CODENAME` in `/etc/os-release`):

```
# /etc/apt/sources.list.d/stratum.sources
Types: deb
URIs: https://www.skraba.net/stratum-apt/
Suites: trixie            # forky on Debian 14
Components: main contrib non-free
Signed-By: /usr/share/keyrings/stratum-archive-keyring.gpg
```

3. **Install the core**

```
# apt-get update && apt-get install stratum-core
```

`stratum-core` provides the `stratum` command, its menu and the login banner; modules are added later. (`stratum-init` still exists as a transitional package that simply installs `stratum-core`.)

3.5 Checking the signing key

```
# gpg --show-keys --with-fingerprint /usr/share/keyrings/stratum-archive-keyring.gpg
pub   ed25519 2026-10-20 [SC] [expires: 2031-10-20]
      4F39 6C32 8AB5 D14E F653  55CF 130D EB73 9D1E 1A2C
uid                      Stratum Archive Signing Key <stratum@skraba.net>
```

> **TIP**
>
> Install `stratum-archive-keyring` as well: future key updates then arrive through APT like any other update.

3.6 First check: stratum status

Right after the installation, ask Stratum whether everything is in order:

```
# stratum status
Stratum 0.40 on Debian 13 (trixie)

==> System
  ✓ Debian 13 (trixie) is supported
==> Configuration
  ✓ source file /etc/apt/sources.list.d/stratum.sources
  ✓ repository https://www.skraba.net/stratum-apt/
  ✓ suite trixie matches this system
```


© 2026 Stratum Project

```
  ✓ signing key /usr/share/keyrings/stratum-archive-keyring.gpg
==> Repository
  ✓ reachable (Release downloaded)
  ✓ signature valid (key 130DEB739D1E1A2C)
==> APT
  ✓ package index present (updated 36h ago)
  ✓ 12 package(s) available, 12 installed: …

Stratum repository is active and working.
```

The exit status is `0` when everything works and `1` when a problem was found — handy in scripts and monitoring.

3.7 Adding modules

Add the modules you need — in the menu under **modules** (chapter 4) or with APT:

```
# apt-get install stratum-software stratum-memory stratum-firewall stratum-security
# apt-get install stratum-watchdog stratum-fileserver stratum-database stratum-wireguard
# apt-get install stratum-license stratum-support stratum-ansible
# stratum help <module>                                  # all actions of a module
```

3.8 Removing Stratum

```
# apt-get purge -y stratum-archive-keyring 2>/dev/null
# rm -f /etc/apt/sources.list.d/stratum.sources /usr/share/keyrings/stratum-archive-keyring.gpg
# apt-get update
```

Installed Stratum packages stay until you remove them with `apt-get purge stratum-…`. Most modules undo their changes when they are purged; your own data (playbooks, shares, databases, backups) always stays.

CHAPTER 4

The stratum command and the menu

END USER ADMINISTRATOR

Everything in Stratum goes through one command: `stratum`. Without arguments it opens the menu; with arguments it does one job and returns — ideal for scripts.

4.1 Starting the menu

```
$ stratum                # menu, read-only for normal users
# stratum                # as root: changes are possible
# stratum --no-splash    # without the start-up animation
```

The menu starts with a short animation of the Stratum logo (any key skips it), then shows the main table:

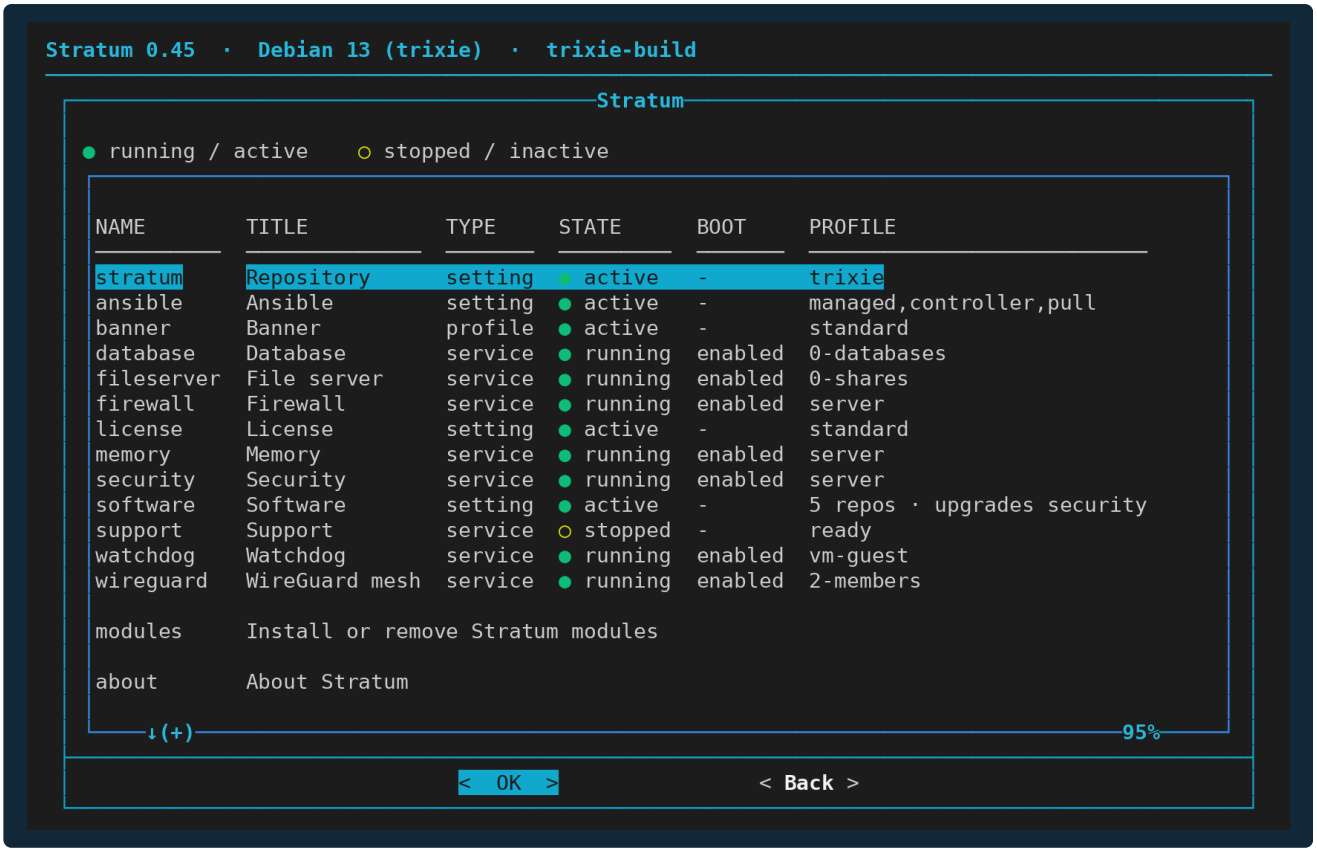


Figure 1 The main menu: the Stratum repository, then all installed modules with their type, state, boot setting and profile.

4.1.1 Reading the main table

Column	Meaning
NAME	The module name — also what you type after `stratum`.
TITLE	A short description.
TYPE	**service** runs and can start at boot · **setting** is applied or not · **profile** is always applied, only the profile is chosen.

Column	Meaning
STATE	● running / active · ○ stopped / inactive · ● a problem, e.g. *mismatch* or *unsupported*.
BOOT	For services: *enabled* (starts at boot) or *disabled*.
PROFILE	What is in use: the type of computer, the profile, or a short summary.

The first row, **stratum**, is the Stratum repository itself. It is *active* when the repository is configured for your release and its package lists are present; `Enter` shows the full `stratum status`.

4.1.2 Keys

`↑` `↓` or a letter	move — a letter jumps to the next row starting with it (e.g. `f` for firewall)
`Enter`	open the row / choose (the **OK** button)
`Esc` or **Back**	one level back
`Tab`	move between the list and the buttons
`Space`	switch an item in a checklist

4.2 Inside a module

Selecting a module opens its own menu. It always looks the same: a short description, the state, the actions and the settings with their current values. A value followed by **· default** comes from the chosen type of computer; anything else you set yourself.

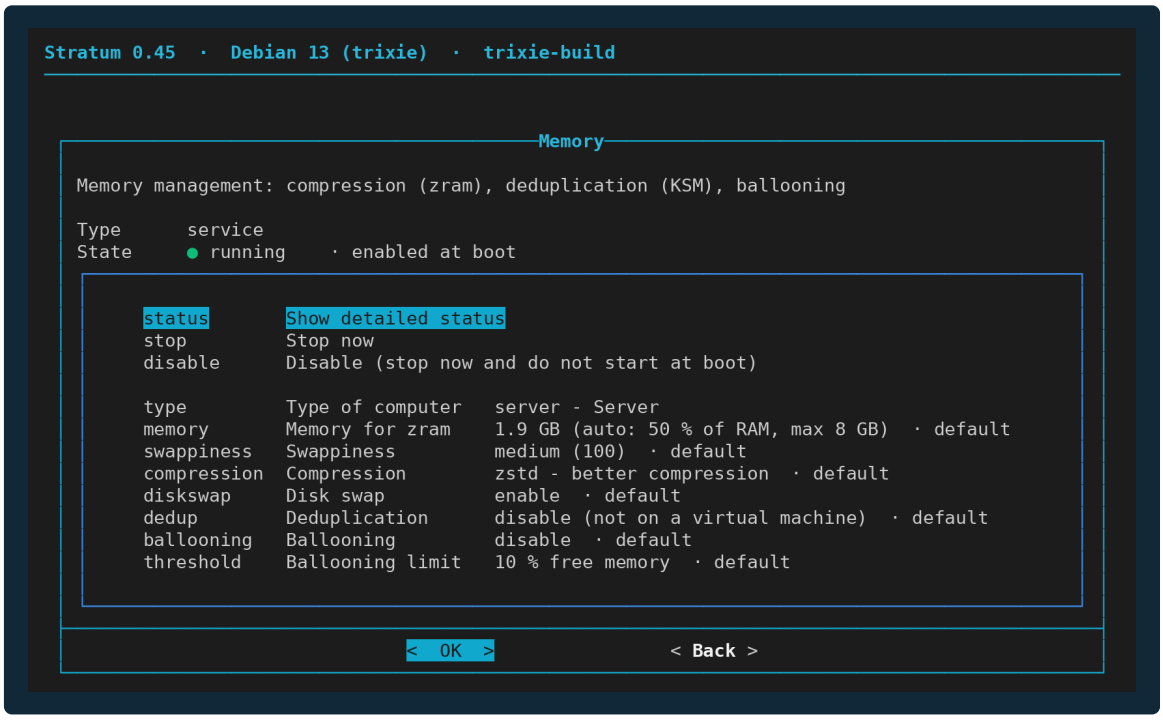


Figure 2 A service module (memory): state, actions (status, stop, disable) and settings with their values.

Selecting a setting opens its choices. The current value is marked with a green dot; `Enter` applies the highlighted one immediately:

© 2026 Stratum Project

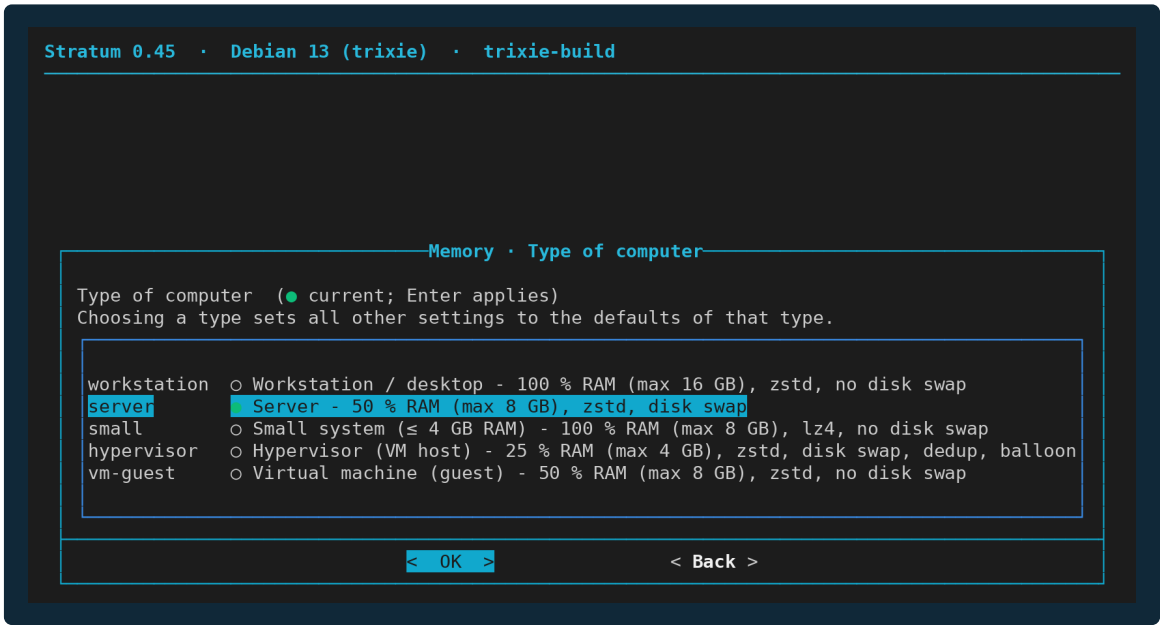


Figure 3 Choosing a value: the type of computer of the memory module.

> **TIP**
>
> Everything the menu does is also available as a command. The menu simply runs `stratum <module> <action>` and shows you the output — you can always repeat a step in a script.

4.3 Installing and removing modules

The row **modules** refreshes the package lists (`apt-get update`) and shows all packages of the Stratum repository as a checklist. `Enter` switches the highlighted package, **Apply** installs and removes:

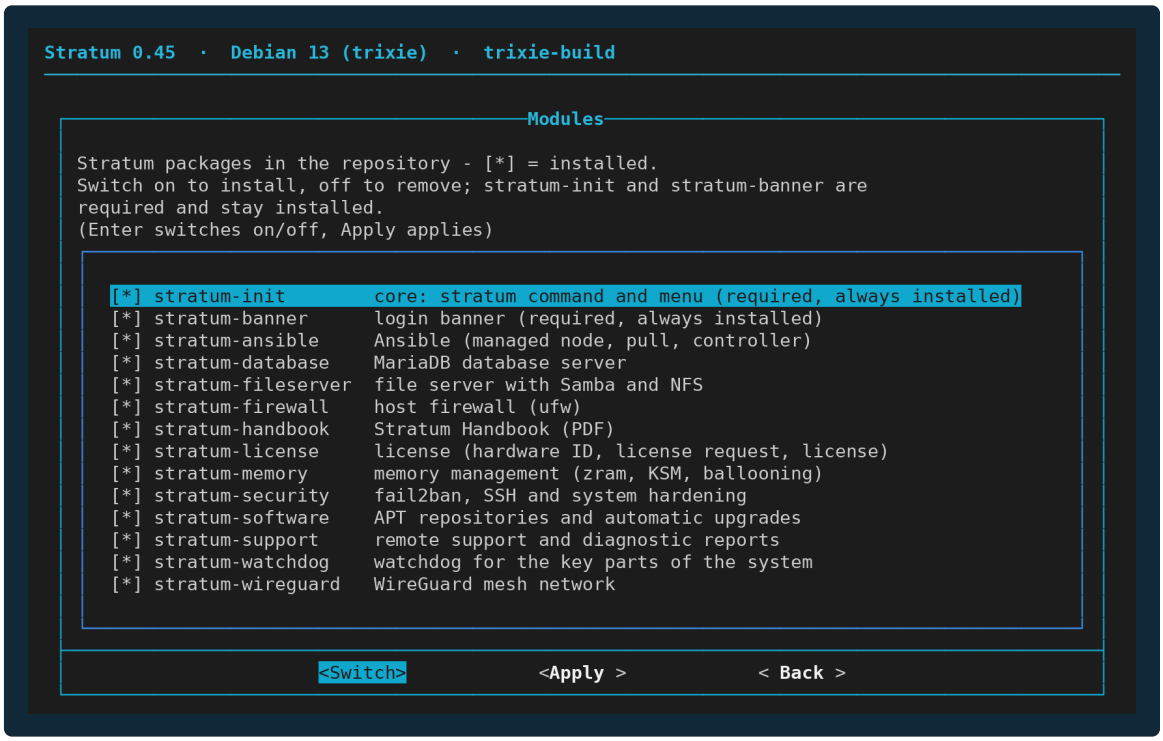


Figure 4 Modules: [*] = installed. stratum-core is required and always stays.

Before anything is changed, the menu shows what will be installed and removed and asks for confirmation. If APT would also remove other packages — because they conflict with a module — they are listed separately and you have to agree explicitly. After the change the menu restarts itself, so new modules appear at once.

4.4 The about screen

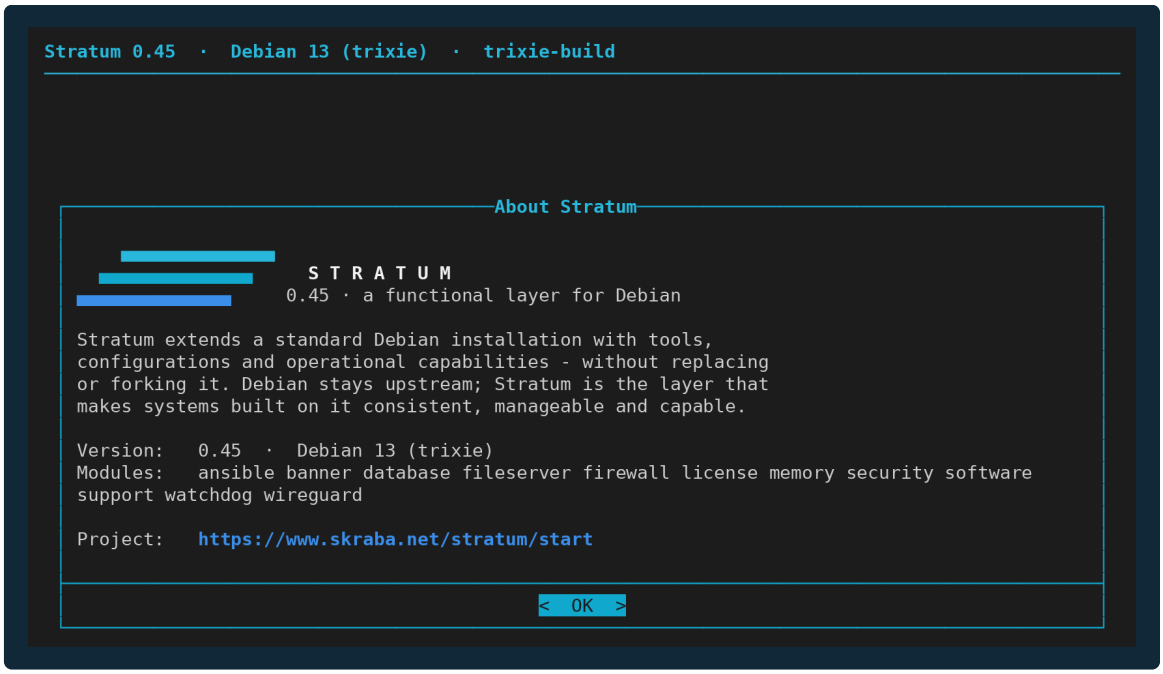


Figure 5 About Stratum: the logo, version, installed modules and the project page.

4.5 Commands

Command	What it does
`stratum` / `stratum menu`	The interactive menu.
`stratum status`	Checks system, repository configuration, signature and APT (exit status 0 = OK).
`stratum modules`	Installed modules with type, state, boot and profile; modules available in the repository.
`stratum about`	Version, modules, project page.
`stratum init`	Runs the initialisation steps of the installed modules (`stratum init --list` shows them).
`stratum help <command|module>`	Help for a command or module; also `stratum <module> help`.
`stratum <module> <action> …`	Manage a module — see the module chapters.
`stratum -V` · `stratum -h`	Version · help.

4.5.1 The common module actions

Action	service	setting	profile
`status`	✓	✓	✓
`start` · `stop`	now, boot setting unchanged	= enable · disable	—

© 2026 Stratum Project

Action	service	setting	profile
`enable` · `disable`	now and at boot	apply · undo	—
`configure <profile>`	✓ (type)	✓	✓
`settings` · `setting <key> <value>`	modules with settings (memory, firewall, security, fileserver …)		
`help`	all actions of the module		

```
# stratum modules
NAME       TITLE              TYPE     STATE     BOOT      PROFILE
ansible    Ansible            setting  active    -         managed,controller,pull
banner     Banner             profile  active    -         standard
firewall   Firewall           service  running   enabled   server
license    License            setting  active    -         standard
memory     Memory             service  running   enabled   server
security   Security           service  running   enabled   server
software   Software           setting  active    -         5 repos · upgrades security
support    Support            service  stopped   -         ready
```

4.6 Options, settings and environment

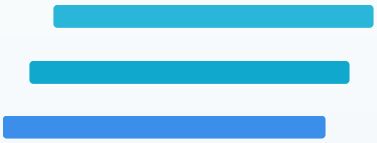
Switch or setting	Effect
`--no-splash`	Open the menu without the start-up animation.
`stratum menu --no-splash`	Open the menu without the start animation.
`NO_COLOR=1`	No colours in command output.
`STRATUM_NO_SPLASH=1`	Skip the animation for this run.
`STRATUM_INTERRUPT=1`	Allow `Ctrl`+`C` (normally ignored so that no change is left half done).
`STRATUM_FORCE=1`	Apply a change despite a warning (exit status 3, e.g. a lock-out risk). The menu asks **Apply anyway?** instead.
`DIALOGRC=<file>`	Your own dialog colour configuration.

> ⚙ **TECHNICAL STUFF**
>
> Modules live in `/usr/lib/stratum/modules/<name>/` : a `module.conf` (title, description, type) and an executable `module` with the actions. `stratum <name> <action>` runs it. Exit status **3** means “refused because of a warning”; the menu then asks whether to apply the change anyway.

PART II

Modules

One chapter per module: what it does, how to set it up in the menu, every command and switch, and examples you can copy.



CHAPTER 5

How modules work

END USER ADMINISTRATOR

All Stratum modules follow the same pattern. Once you know one, you know them all. This short chapter explains the pattern, so the following chapters can stay focused on what is special about each module.

5.1 Three types of module

Type	How it behaves	Modules
service	Something that runs. It can be *started* and *stopped* now, and *enabled* or *disabled* at boot. Has settings.	memory, firewall, security, fileserver, database, wireguard, support, watchdog
setting	A configuration that is either *active* or not. `enable` switches it on, `disable` restores what was there before.	software, ansible, license
profile	Always on; you only choose a *profile* with `configure <profile>`.	banner

5.2 The common actions

Command	What it does
`stratum <module> status`	State, boot setting and all settings in one view. Always safe — it changes nothing.
`stratum <module> start` / `stop`	Service now only (until the next boot).
`stratum <module> enable` / `disable`	At boot *and* now.
`stratum <module> restart`	Apply the settings again.
`stratum <module> settings`	All settings with their current values.
`stratum <module> setting <key> <value>`	Change one setting.
`stratum <module> configure <profile>`	Choose a profile or type of computer.
`stratum <module> help`	The complete help of the module: actions, options, files.

> **REMEMBER**
> Every option is documented in three places that always say the same: `stratum <module> help`, `stratum help` and the manual page (`man stratum`).

5.3 Exit codes

Scripts can rely on the exit code of every Stratum command:

Code	Meaning
`0`	Done.

Code	Meaning
`1`	Error — nothing (or nothing more) was changed. The message says why.
`3`	Warning — the change was *not* applied because it looks risky (for example, it could lock you out). Repeat with `STRATUM_FORCE=1` to apply it anyway; the menu asks **Apply anyway?** instead.

```
# stratum firewall rule-add deny ssh
  ✗ LOCK-OUT RISK: the rule 'deny 22/tcp' blocks the SSH connection from 10.0.0.15
  ✗ not applied. To apply it anyway: STRATUM_FORCE=1 stratum firewall ...
# echo $?
3
```

5.4 First start (setup)

Some modules need a one-time preparation the first time you open them — the license module, for example, creates the hardware ID. The menu notices this and opens the **setup** dialog by itself. On the command line, the module tells you to run `stratum <module> setup`.

© 2026 Stratum Project

CHAPTER 6

Software — stratum software

END USER ADMINISTRATOR

The **software** module decides where your software comes from and how it is kept up to date. Each repository is a switch — on or off, with or without source packages — and security fixes are installed while you sleep. No more editing `sources.list` or `unattended-upgrades` by hand.

6.1 What you get

- The official Debian repositories of your release: **debian**, **debian-updates**, **debian-security** and **debian-backports**.
- The **stratum** repository — always on, because Stratum itself comes from there.
- The Debian *components*: `main`, `contrib`, `non-free`, `non-free-firmware`.
- Your own repositories, added as small definition files.
- **Automatic upgrades** with Debian's `unattended-upgrades` and one simple choice: **complete**, **security** or **turn-off**.

> **WARNING**
>
> The Debian repositories and the automatic upgrades are only offered on an official, supported Debian release (Debian 13 trixie, Debian 14 forky). On Raspberry Pi OS, Ubuntu or other derivatives the module refuses to touch them — mixing repositories of different distributions breaks systems.

6.2 In the menu END USER ADMINISTRATOR

Select **software**. You get three lists: **repositories**, **components** and **upgrades**.

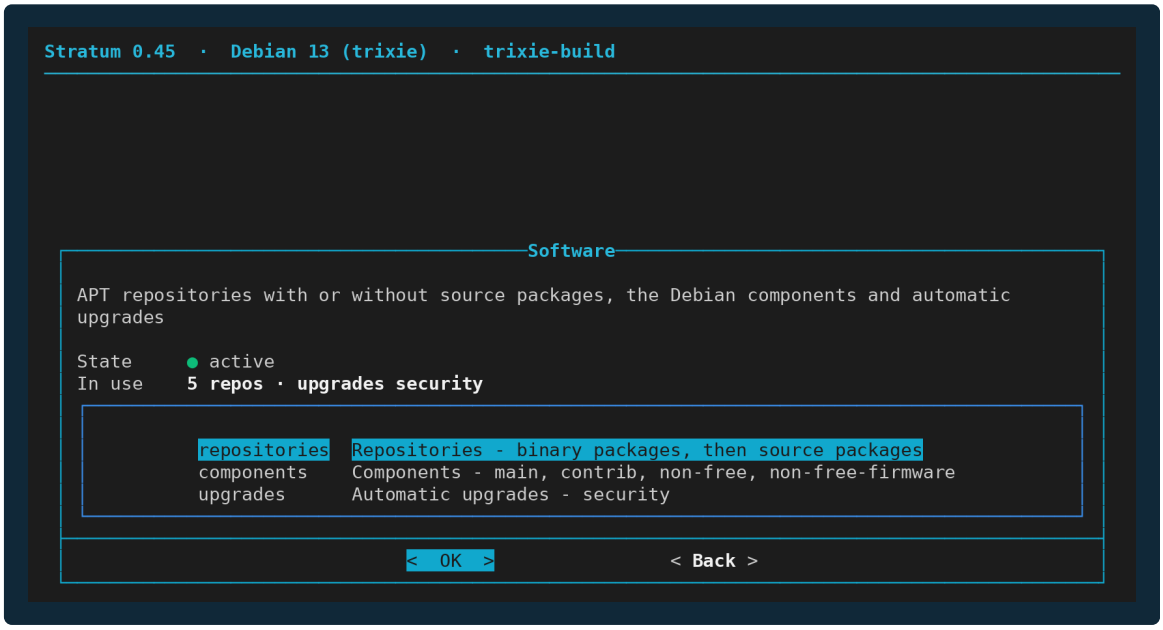


Figure 6 The software module in the menu.

The **repositories** checklist shows the binary repositories first and their source packages (marked `src:`) at the bottom. Tick what you want with `Space` and confirm with `Enter`. Stratum writes the APT files and runs `apt-get update` for you.

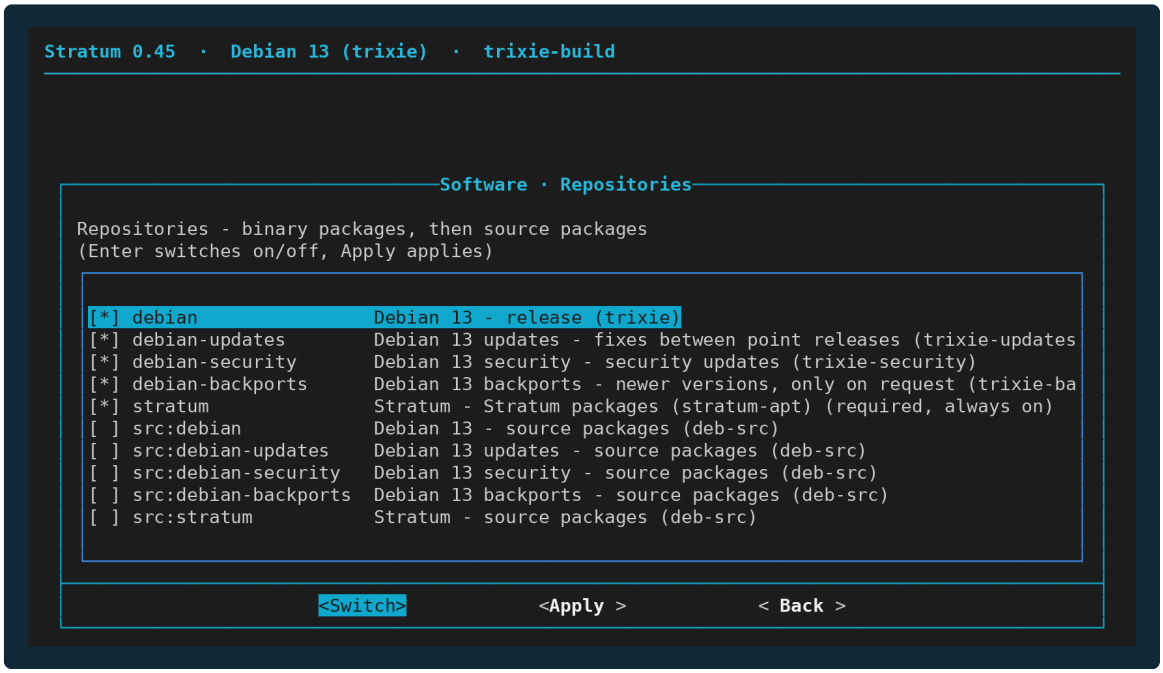


Figure 7 Switching repositories on and off.

upgrades is a choice of exactly one profile: the current one is marked, `Enter` applies the one you select.

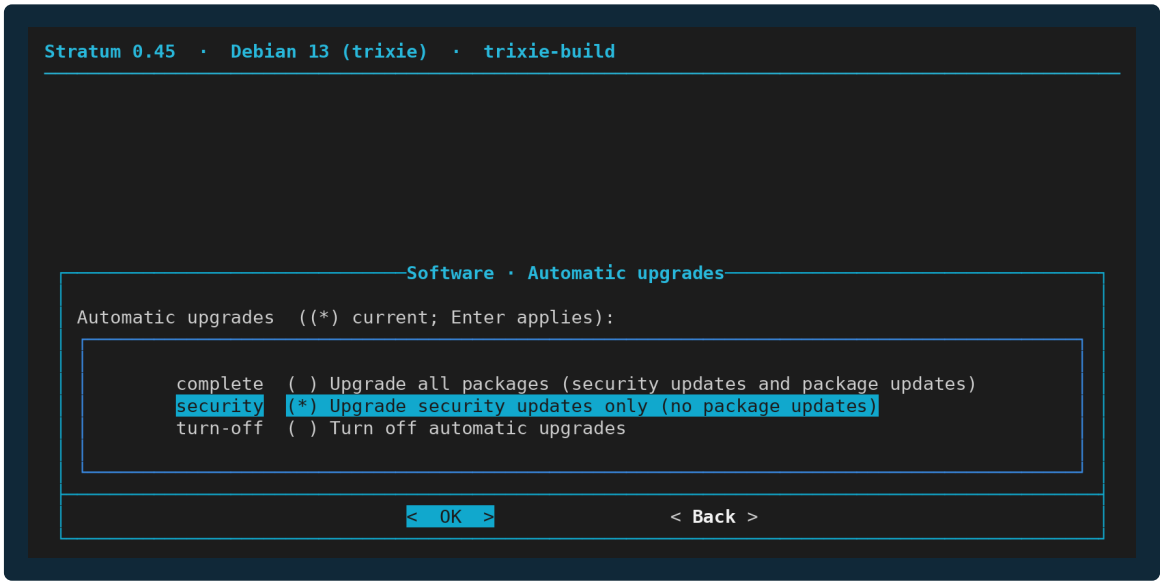


Figure 8 Choosing what is upgraded automatically.

6.3 Automatic upgrades

Profile	What is upgraded automatically
complete	All packages: security updates and package updates.
security	Security updates only — the default.
turn-off	Nothing; you upgrade by hand.

© 2026 Stratum Project

> **REMEMBER**
>
> Stratum never reboots your computer automatically, and backports are never upgraded automatically. If a kernel update needs a reboot, `stratum status` tells you.

The allowed origins of the automatic upgrades follow the repositories by themselves: a repository you switch on is upgraded with the profile **complete**, one you switch off is left alone.

6.4 Commands

Command	What it does
`stratum software status`	State, system, components, all repositories and the automatic upgrades.
`stratum software list`	Just the table of repositories.
`stratum software enable`	Enable the defaults (debian, -updates, -security, -backports, stratum; no sources) or the ones enabled before. The previous APT sources are kept as `*.stratum-software.bak`.
`stratum software enable <repo>…`	Switch on the binary packages of these repositories.
`stratum software disable`	Switch all off (except stratum) and restore the previous sources.
`stratum software disable <repo>…`	Switch off these repositories.
`stratum software sources <repo> on|off`	Source packages (`deb-src`) of one repository.
`stratum software components [comp…]`	Show or choose the Debian components.
`stratum software upgrades [profile]`	Show or choose the automatic upgrades: complete, security or turn-off (also `configure <profile>`).
`stratum software refresh`	Write the allowed origins of the automatic upgrades again.
`stratum software set repositories <item>…`	Exactly these on, all others off. Items are `<repo>` and `src:<repo>`.
`stratum software set components <comp>…`	Exactly these components.
`stratum software set upgrades <profile>`	The profile of the automatic upgrades.

Every change of the repositories runs `apt-get update`. Set `STRATUM_NO_UPDATE=1` to skip it, for example when you make several changes in a script.

6.4.1 Example: firmware for a new network card

```
# stratum software components main non-free-firmware
# apt install firmware-iwlwifi
```

6.4.2 Example: only security updates

```
# stratum software upgrades security
  ✓ automatic upgrades: Upgrade security updates only (no package updates)
```

6.4.3 Example: status

```
# stratum software status
State:          active
System:         Debian 13 (trixie) (supported)
Components:     main contrib non-free non-free-firmware (Debian repositories)
Architecture:   amd64
Upgrades:       security - Upgrade security updates only (no package updates) (daily, no automatic reboot)
Upgradable:     0 package(s) now

REPOSITORY         BINARY    SOURCES  DESCRIPTION
debian             on        off      Debian 13 - release (trixie)
debian-updates     on        off      Debian 13 updates - fixes between point releases
debian-security    on        off      Debian 13 security - security updates
debian-backports   on        off      Debian 13 backports - newer versions, only on request
stratum            on        off      Stratum - Stratum packages (stratum-apt)
```

TIP

Backports are newer versions of selected packages. They are enabled, but APT never takes them unless you ask: `apt install -t trixie-backports <package>`.

6.5 Adding your own repository

A repository is described by a small file in shell syntax. Put your own in `/etc/stratum/software.d/<id>.repo` — the file name is the id.

```
# /etc/stratum/software.d/example.repo
NAME="Example"
DESCRIPTION="Tools from example.org"
URIS="https://example.org/apt"
SUITES="@codename"                # trixie or forky, whatever the system runs
COMPONENTS="main"
SIGNED_BY="/usr/share/keyrings/example.gpg"
SOURCES="no"
ORDER="60"
```

```
# stratum software enable example
```

Key	Meaning
`NAME`, `DESCRIPTION`	Short name and one line of description.
`URIS`, `SUITES`, `COMPONENTS`	As in deb822 sources. `@codename` = the release name, `@version` = its number, `@debian` = the chosen Debian components.
`SIGNED_BY`	The keyring that signs the repository.
`SOURCES`, `DEFAULT_SOURCES`	Whether source packages exist / are on by default.
`REQUIRES`	`debian`: only on a supported official Debian release.
`REQUIRED`, `DEFAULT`	Always on / on with a plain `enable`.
`ARCHITECTURES`, `FILE`, `ORDER`	Architectures, an APT file name of your choice, position in the list.

© 2026 Stratum Project



TECHNICAL STUFF

Each repository has its own file `/etc/apt/sources.list.d/stratum-<repo>.sources`, also when it is off: then its lines are commented out and the file says which command switches it on. A definition in `/etc/stratum/software.d/` with the same name as one in `/usr/lib/stratum/software.d/` overrides it. The settings are in `/etc/stratum/software.conf`, the configuration of the automatic upgrades in `/etc/apt/apt.conf.d/52stratum-software` (APT::Periodic and allowed origins), the logs in `/var/log/unattended-upgrades/`.

CHAPTER 7

Login banner — stratum banner

END USER ADMINISTRATOR

A small touch that tells you at a glance which computer you are on and whether Stratum is healthy.

The banner shows a Stratum line on the console login screen and in the *message of the day* (what you see after logging in over SSH): the version and the state of the enabled modules, for example *"6 modules active"* or which enabled service is not running.

7.1 Profiles

Profile	What you see
standard	The Stratum logo above the first login screen after boot (with a short beep), plus the login and MOTD line. The logo disappears one minute after boot.
minimal	Only the Stratum line on the login screen and in the MOTD.
discrete	Nothing shows that Stratum is installed — no logo, no login line, no MOTD line. Needs a valid Stratum license (chapter 16).

REMEMBER

discrete can only be chosen while a valid license is installed. If the license expires, the banner falls back to **minimal** (at the next boot or change) and `stratum banner state` says why; a renewed license restores **discrete** by itself.

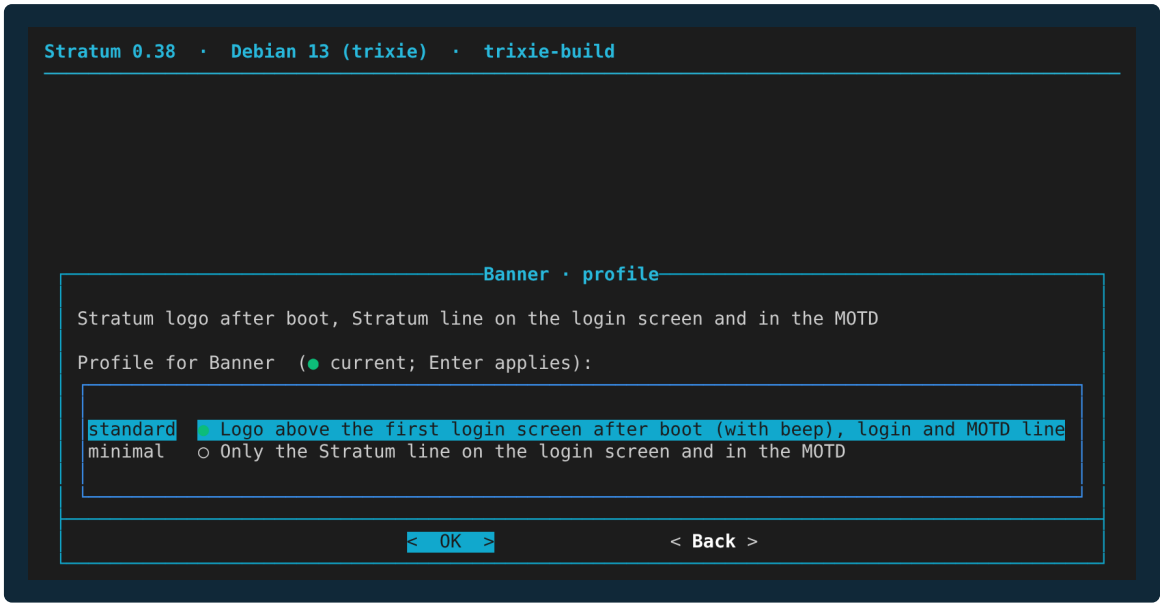


Figure 9 Choosing the banner profile.

7.2 Commands and files

Command	What it does
`stratum banner configure standard|minimal|discrete`	Choose the profile.
`stratum banner preview`	Print the logo and the login line in your terminal.

File	Purpose
`/etc/stratum/banner.conf`	Chosen profile.
`/etc/default/stratum-banner`	`LOGIN_TAGLINE`, `BANNER_STYLE` (unicode|ascii), `BANNER_BEEP` (yes|no — overrides the profile).
`/etc/issue.d/stratum.issue`	Login line.
`/etc/update-motd.d/20-stratum`	MOTD line.

> **TIP**
>
> No beep in the server room at night: keep **standard** and set `BANNER_BEEP=no` in `/etc/default/stratum-banner`.

> **TECHNICAL STUFF**
>
> Early versions had the profiles *full*, *quiet* and *minimal*. *full* and *quiet* became **standard** (the beep is now a separate switch); old configurations are converted automatically.

© 2026 Stratum Project

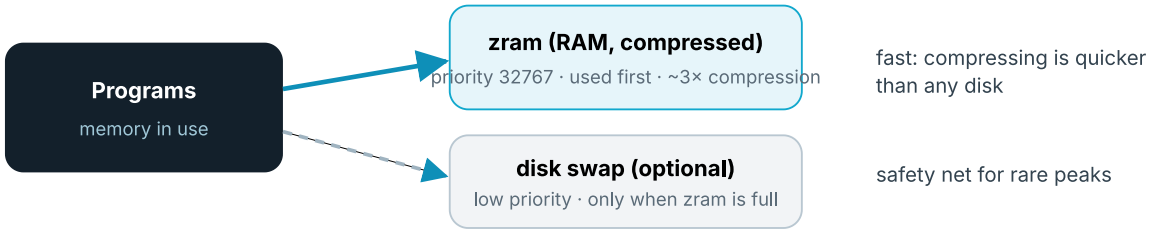
CHAPTER 8

Memory — stratum memory

END USER ADMINISTRATOR

Give your computer more memory without buying any. The **memory** module brings three techniques together: **compression** keeps swapped-out pages compressed in RAM — many times faster than swapping to disk — **deduplication** keeps equal memory pages of virtual machines only once, and **ballooning** lets virtual machines give memory back when the host runs short.

8.1 How it works



zram is always used first; disk swap only catches what does not fit.

- **Compression** (zram): swapped-out pages stay in RAM, compressed — typically 3:1 with zstd. zram always has the highest swap priority; a disk swap only takes what does not fit.
- **Deduplication** (KSM with `ksmtuned`): the kernel finds memory pages with the same content — for example the same operating system in several virtual machines — and keeps one copy. `ksmtuned` switches the scanning on only when memory gets tight. It is used on a physical host only: inside a virtual machine, KSM belongs to the host that runs it.
- **Ballooning**: when the free memory of the host falls below a limit, every running virtual machine (libvirt) with a minimum memory gives a tenth of its memory back, step by step and never below its minimum; with enough free memory again it gets it back.

8.2 In the menu END USER

Select **memory**. The most important setting is the **type of computer**: it sets sensible values for everything else.

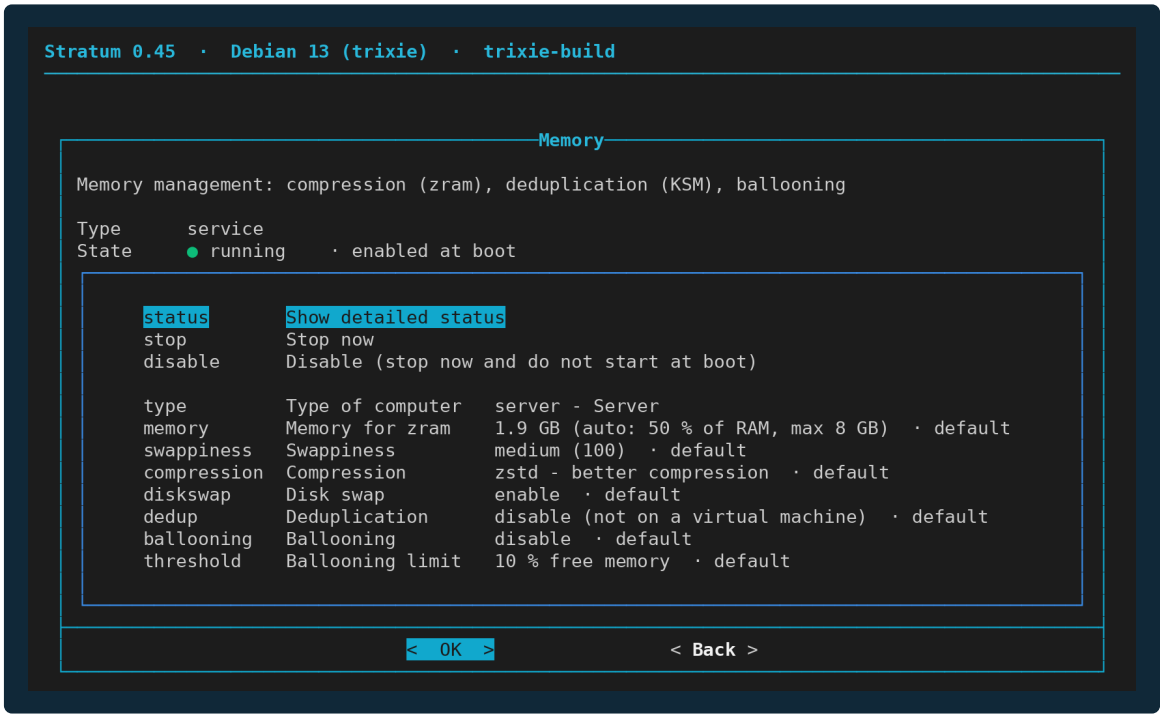


Figure 10 The memory settings.

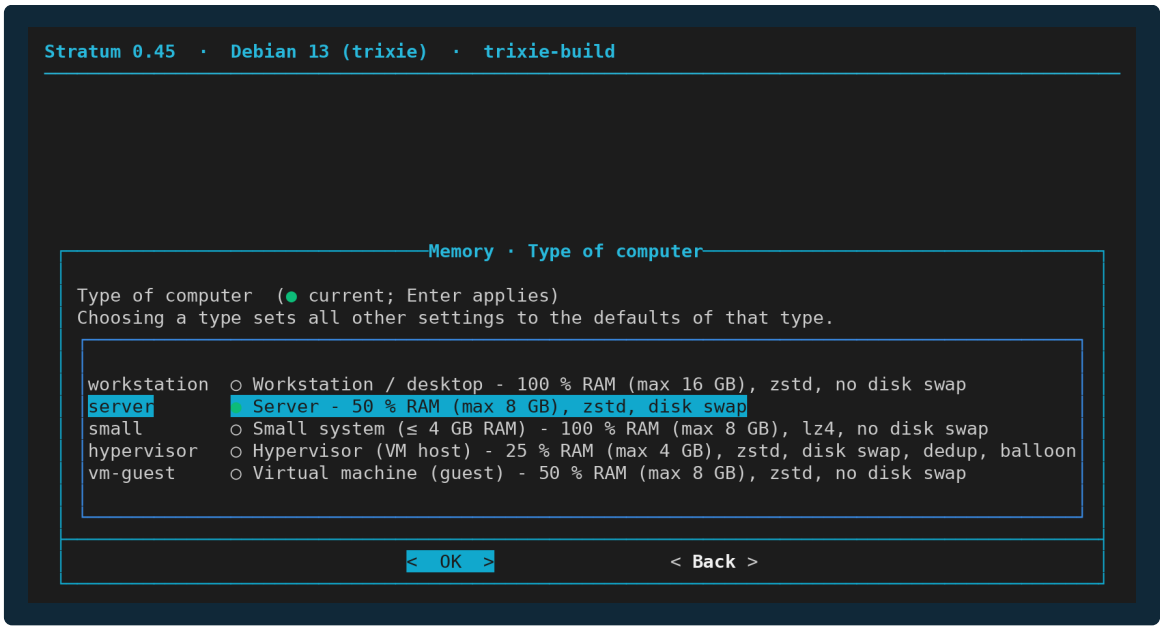


Figure 11 Choosing the type of computer.

8.3 Types of computer

Type	zram size	Compression	Disk swap	Swappiness	Dedup, ballooning
workstation	100 % RAM, max 16 GB	zstd	off	180	off
server	50 % RAM, max 8 GB	zstd	on	100	off
small	100 % RAM, max 8 GB	lz4	off	180	off
hypervisor	25 % RAM, max 4 GB	zstd	on	60	on
vm-guest	50 % RAM, max 8 GB	zstd	off	100	off

© 2026 Stratum Project

> **REMEMBER**
>
> A zram size of 100 % RAM does not use all your RAM. The size is the amount of *uncompressed* data zram can hold; with typical compression it occupies about a third of that — and only when it is actually used.

8.4 Commands

Command	What it does
`stratum memory status`	State, boot setting, all settings, deduplication, ballooning and active swap.
`stratum memory start` / `stop`	Memory management on/off now only.
`stratum memory enable`	At boot, and now.
`stratum memory disable`	Off now and at boot; kernel VM defaults and disk swap as before, `ksmtuned` and ballooning stopped.
`stratum memory restart`	Apply the settings again.
`stratum memory setting type <name>`	Type of computer; resets the other settings to its defaults (also `configure <name>`).
`stratum memory setting memory <GB>|auto`	zram size in GB, or the type's percentage of RAM.
`stratum memory setting swappiness <0-200>|low|medium|high|maximum`	How readily the kernel swaps (60 / 100 / 150 / 180).
`stratum memory setting compression zstd|lz4`	zstd: better compression, about 3:1 (recommended); lz4: fastest, about 2:1, less CPU.
`stratum memory setting diskswap enable|disable`	Disk swap as a low-priority reserve; without any, `/var/swapfile` (max 4 GB) is created.
`stratum memory setting dedup enable|disable`	Deduplication (KSM with `ksmtuned`).
`stratum memory setting ballooning enable|disable`	Automatic ballooning of the virtual machines.
`stratum memory setting threshold <5-50>`	Ballooning limit: % of free host memory below which the machines give memory back.
`stratum memory setting recompress <minutes>`	Compress the idle pages in zram again with the second algorithm of the kernel every N minutes (0 = off; needs kernel support).
`stratum memory dedup-config [recommended | KEY=VALUE…]`	Show or change the values of `ksmtuned` (`/etc/ksmtuned.conf`).
`stratum memory dedup-unmerge`	Split the merged pages again.
`stratum memory show [type]`	The defaults of a type on this machine.

```
# stratum memory configure server
  ✓ type: server - all settings are the defaults of this type now
# stratum memory status
State:          running
Boot:           enabled

Type of computer:  server - Server
Memory for zram:   1.9 GB (auto: 50 % of RAM, max 8 GB)  · default
Swappiness:        medium (100)  · default
Compression:       zstd - better compression  · default
```

```
Disk swap:         enable   · default
Deduplication:     disable  · default
Ballooning:        disable  · default
Ballooning limit:  10 % free memory   · default

Active swap:
  NAME       TYPE      SIZE USED  PRIO
  /dev/zram0 partition 1.9G   0B 32767
  /dev/sda5  partition 1.7G   0B    -2
```

8.5 Ballooning and the minimum memory

A virtual machine takes part in the automatic ballooning when its libvirt definition holds a minimum memory (in MiB):

```
<metadata>
  <memory:memory xmlns:memory="https://www.skraba.net/stratum/memory" min="1024"/>
</metadata>
```

Machines without it are left as they are. Every change is one line in the journal: `journalctl -u stratum-memory-balloon`. The guest needs the virtio balloon driver, which every current Linux kernel has.

> **WARNING**
>
> `zram-tools` (another zram package) must not be active at the same time. Stratum refuses to start until it is removed: `apt purge zram-tools`.

> **TECHNICAL STUFF**
>
> The zram device is created by Debian's `systemd-zram-generator` from `/etc/systemd/zram-generator.conf.d/90-stratum-memory.conf`. VM settings go to `/etc/sysctl.d/90-stratum-memory.conf`. Disk swap lines in `/etc/fstab` are disabled with the prefix `#stratum-memory#` and restored by `disable`. Ballooning runs as `stratum-memory-balloon.service`, the recompression as `stratum-memory-recompress.timer`. The settings are in `/etc/stratum/memory.conf`; own types: copy a type file to `/etc/stratum/memory.d/<name>.profile` and edit it.

© 2026 Stratum Project

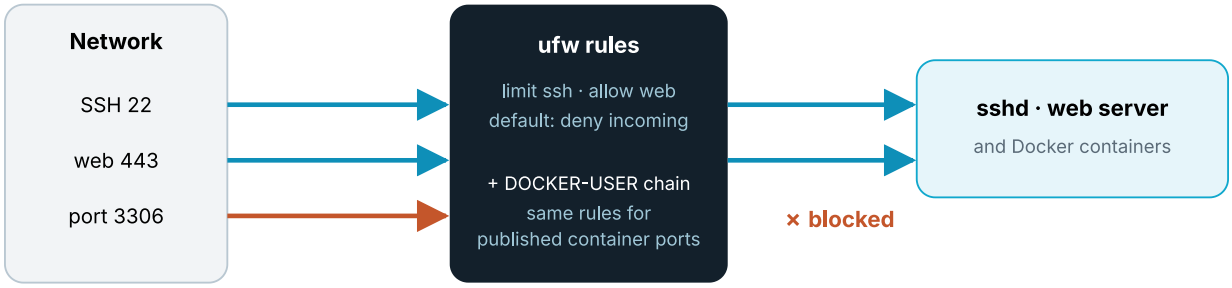
CHAPTER 9

Firewall — stratum firewall

ADMINISTRATOR

A firewall decides who may knock on which door of your computer. Stratum builds it with Debian's `ufw`, gives you sensible defaults for your type of computer, and makes sure you do not lock yourself out.

9.1 The idea in one picture



Only what a rule allows gets in. Docker's published ports follow the same rules.

9.2 Safety first: you will not lock yourself out

Switching on a firewall on a remote server is the classic way to lose it. Stratum has three layers of protection:

1. **SSH guard.** The ports `sshd` listens on are always allowed — unless you wrote your own rule for that port.
2. **Lock-out check.** A change that would block an SSH connection that is open right now is refused with exit code 3 (*LOCK-OUT RISK*). Only `STRATUM_FORCE=1` applies it anyway.
3. **Confirm or roll back.** Every change that could cut the connection must be confirmed within **60 seconds** with `stratum firewall confirm` (in the menu: **Keep**). If your connection drops, you cannot confirm — and the previous state returns by itself.

```
# stratum firewall enable
  ✓ firewall running and enabled at boot
  ! confirm within 60 s:  stratum firewall confirm   (otherwise the previous state returns)
# stratum firewall confirm
  ✓ firewall changes kept
```

> **TIP**
>
> Before you confirm, open a *second* SSH connection. If it works, confirm in the first one.

9.3 In the menu END USER ADMINISTRATOR

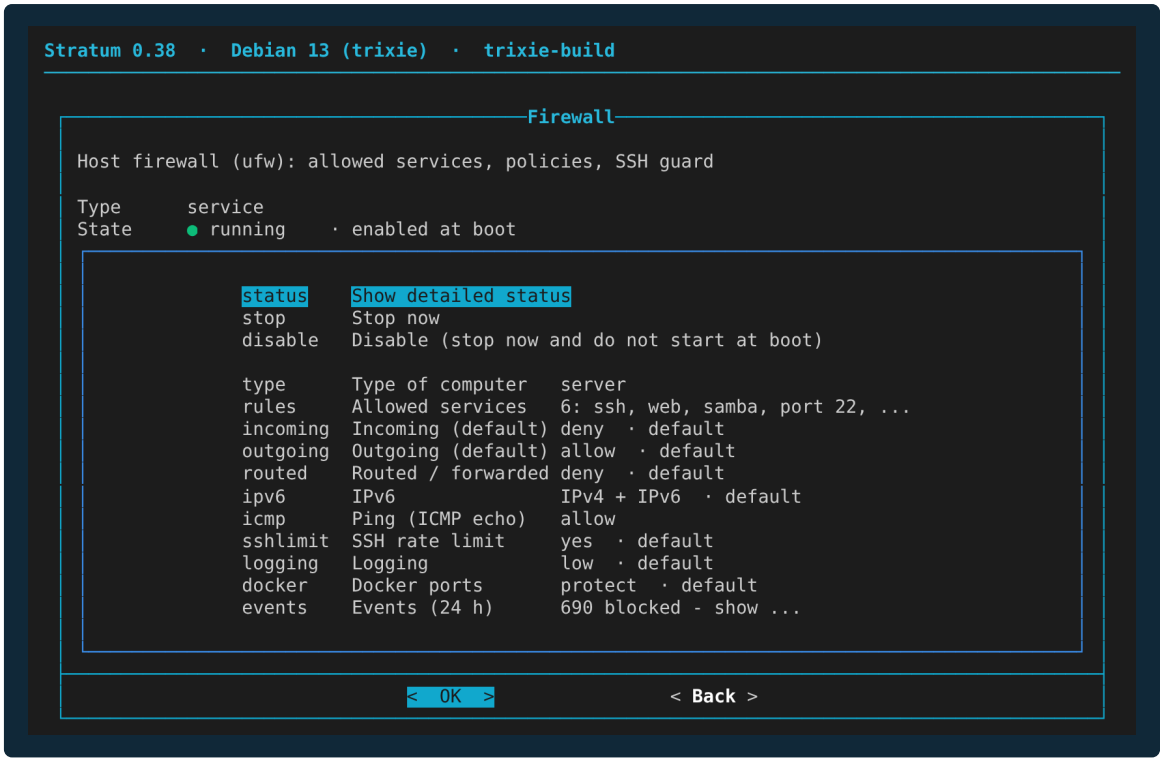


Figure 12 The firewall module: state, settings, rules and events.

Choose the **type of computer** first; it sets the defaults and the starting rules. Then add rules for what this computer offers. A new rule is made in three questions: *what* (a service or a port), *where* (all interfaces or one) and *from whom* (anyone or a network).

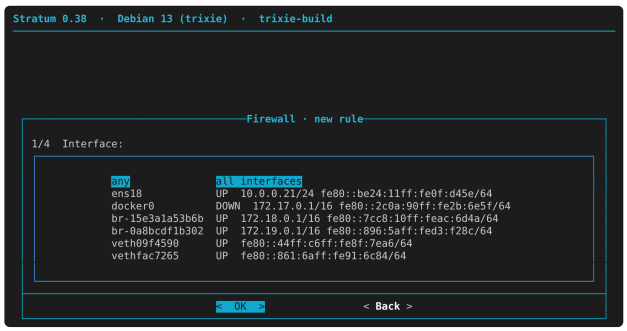


Figure 13 New rule: action.

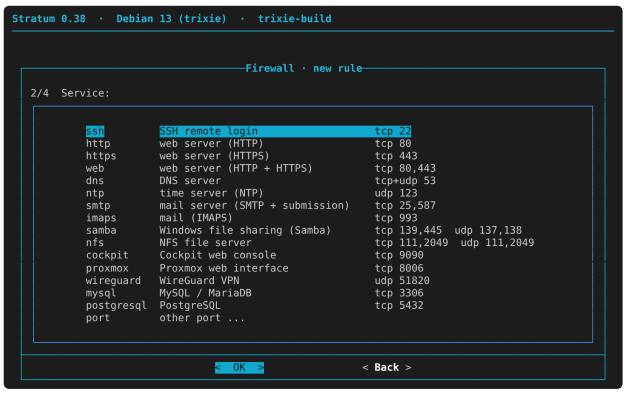


Figure 14 New rule: choose a service.

© 2026 Stratum Project

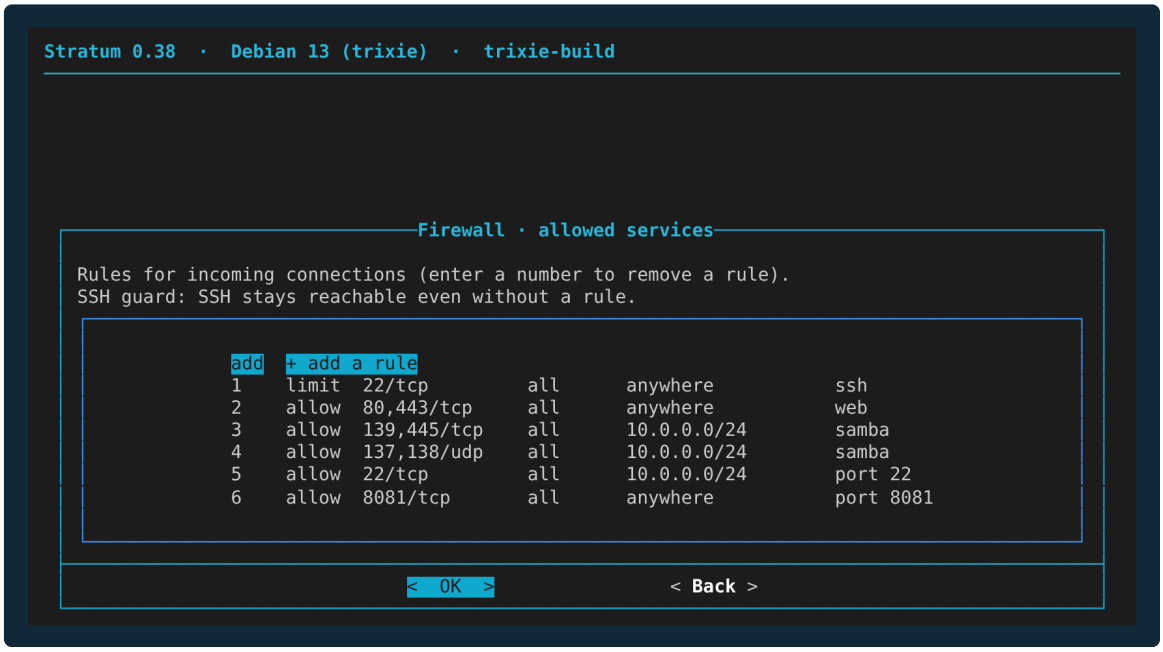


Figure 15 The list of rules.

9.4 Settings

Setting	Values and meaning
`type`	workstation · server · hypervisor · vm-guest — resets settings and rules to the type's defaults.
`incoming`	**deny** (default) · reject · allow. What happens to connections no rule mentions.
`outgoing`	**allow** · deny. With deny, DNS, updates and NTP need rules too.
`routed`	**deny** · allow. Forwarding, for VMs and containers.
`ipv6`	**yes** · no. Rules also for IPv6.
`icmp`	allow · deny. Answer ping.
`sshlimit`	**yes** · no. Rate limit for SSH: 6 connections in 30 s.
`logging`	off · **low** · medium · high. Needed for `events`.
`docker`	**protect** · open. Whether published container ports follow the rules.

9.5 Rules

```
# stratum firewall rule-add allow web
# stratum firewall rule-add allow tcp:8080 ens18 10.0.0.0/24
# stratum firewall rule-add limit ssh any 192.168.1.0/24
# stratum firewall rules
 1  limit  22/tcp         all        anywhere          ssh
 2  allow  80,443/tcp     all        anywhere          web
 3  allow  139,445/tcp    all        10.0.0.0/24       samba
 4  allow  137,138/udp    all        10.0.0.0/24       samba
 5  allow  8080/tcp       ens18      10.0.0.0/24       port 8080
# stratum firewall rule-remove 5
```

The form is `rule-add <allow|limit|deny> <what> [interface|any] [from|any]`, where *what* is a service name or `tcp:PORT`, `udp:PORT`, `any:PORT`. A new rule with the same protocol, port, interface and source *replaces* the old one — just as ufw does.

Service	Ports	Service	Ports
ssh	tcp 22	samba	tcp 139,445; udp 137,138
http / https	tcp 80 / 443	nfs	tcp+udp 111,2049
web	tcp 80,443	cockpit	tcp 9090
dns	any 53	proxmox	tcp 8006
ntp	udp 123	wireguard	udp 51820
smtp	tcp 25,587	mysql	tcp 3306
imaps	tcp 993	postgresql	tcp 5432

> **REMEMBER**
> Stratum marks its own ufw rules with the comment `stratum: …`. Rules you added by hand with `ufw` stay untouched.

9.6 Docker and the firewall

Docker normally publishes container ports *past* ufw — a well-known surprise. With the default `docker protect`, Stratum applies its rules to published ports as well (in the `DOCKER-USER` chain, by the host port). So `rule-add allow tcp:8080` opens a container published on port 8080, and a container without a rule stays unreachable from outside.

```
# stratum firewall listening
PROTO ADDRESS                 PROGRAM          FIREWALL
tcp   0.0.0.0:22              sshd             allowed (Stratum rule)
tcp   0.0.0.0:8081            docker:whoami    allowed (Stratum rule)
tcp   0.0.0.0:8082            docker:web2      BLOCKED (Docker)
```

> **TIP**
> `listening` is the quickest answer to “why can't I reach my service?”: it lists every listening program and whether the firewall lets others in.

9.7 Rules of other modules

Some modules need ports or forwarding of their own: the file server (Samba, NFS), the database when it listens on the network and the WireGuard mesh (its UDP port, traffic from the mesh, relay). Each of them writes its rules to `/etc/stratum/firewall.d/<module>.rules`; the firewall adds them to its own, with the comment `stratum: <module>: …`. You do not edit these files — change the module's settings instead.

```
# stratum firewall module-rules
  allow  445/tcp        any        10.0.0.0/24        fileserver: samba
  allow  51820/udp      any        any                wireguard: WireGuard mesh lab
  allow  any/any        stratum-wg any                wireguard: traffic from the mesh
  route  any            stratum-wg>stratum-wg to any  wireguard: relay between members
```


© 2026 Stratum Project

When a module changes its file, it calls `stratum firewall apply-modules`: only the rules of the modules are replaced, your own rules and the safety rules stay as they are — and nothing happens at all when the files did not change. `allow` lines open a port for incoming connections, `route` lines allow forwarding (for example through the mesh).

> **REMEMBER**
> Without stratum-firewall, these files are simply not used; the modules work the same.

9.8 Events

`stratum firewall events [hours]` summarises blocked connections — top sources, top ports and the last ten — from the kernel log. It needs `logging` to be at least **low**.

> **TECHNICAL STUFF**
> Some networks send multicast group messages (IGMP, protocol 2, source `0.0.0.0`) every few minutes. They show up as blocked events from `0.0.0.0` to port `-/2` and are harmless.

9.9 All commands

Command	What it does
`status`	State, settings, rules, interfaces, listening services, ufw.
`start` · `stop` · `enable` · `disable` · `restart`	As for every service module.
`confirm` · `rollback`	Keep the last change / restore the state before it.
`settings` · `setting <key> <value>`	Show / change settings.
`rules` · `rule-add …` · `rule-remove <n>`	Numbered rules; add; remove.
`events [hours]`	Blocked connections (default 24 h).
`interfaces`	Network interfaces with addresses.
`listening`	Listening services and whether they are let in.
`docker-mode` · `docker-apply`	Show the Docker handling / apply the rules to Docker.
`port-allowed <proto> <port>`	Exit code 0 when the port is open to others (for scripts).
`module-rules`	The rules of other modules (`/etc/stratum/firewall.d`).
`apply-modules`	Apply only the rules of the modules (called by the modules).

Environment: `STRATUM_FIREWALL_NOCONFIRM=1` (no confirmation, for scripts), `STRATUM_FORCE=1`. Files: `/etc/stratum/firewall.conf`, `/etc/stratum/firewall.rules`, `/etc/ufw/`.

CHAPTER 10

Security — stratum security

ADMINISTRATOR

Every server on the internet is visited by robots trying passwords. The **security** module bans them with fail2ban and offers optional hardening of SSH and the kernel — with the same protection against locking yourself out as the firewall.

10.1 What it does

fail2ban (on)

Watches the logs of protected services. An address that fails to log in *maxretry* times within *findtime* is banned for *bantime*. Repeat offenders get longer bans, up to a week.

Hardening (off by default)

Five independent switches for SSH and the kernel. Each one is a good idea on a server — and each one is your decision.

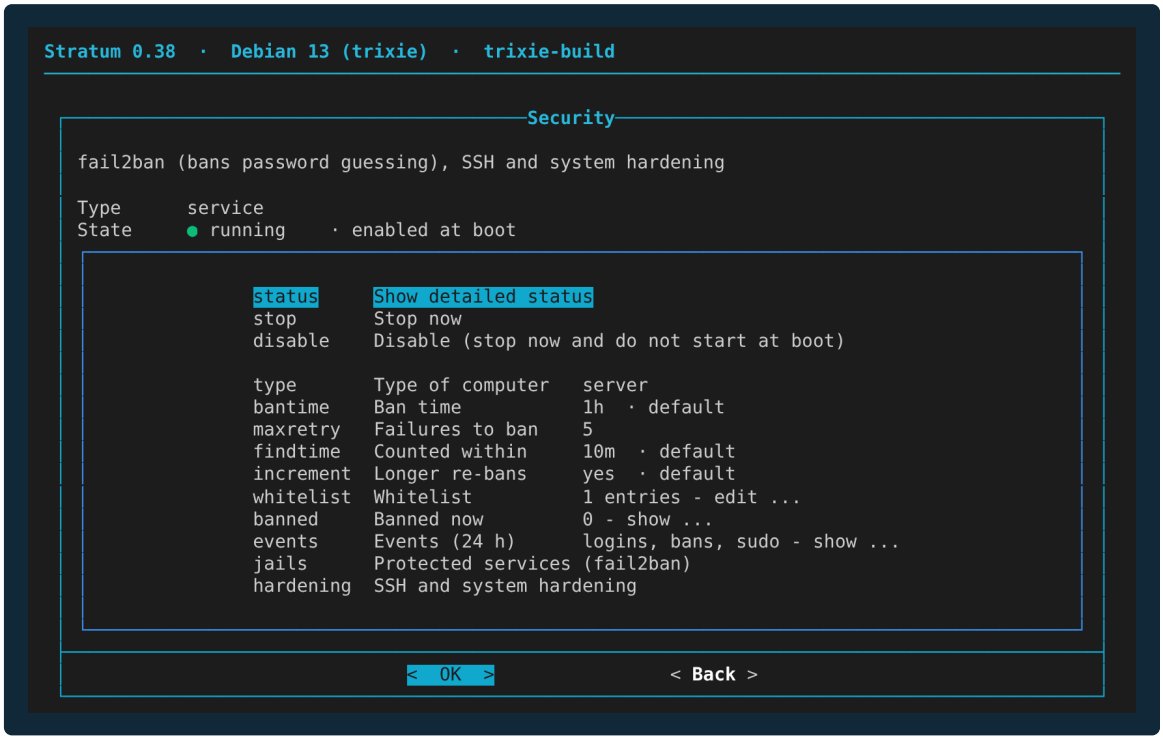


Figure 16 The security module in the menu.

10.2 Settings

Setting	Values and meaning
`type`	workstation · server · hypervisor · vm-guest — resets ban settings and services.
`bantime`	10m · **1h** · 1d · 1w — how long an address is banned.
`maxretry`	1–20 — failures before a ban.

Setting	Values and meaning
`findtime`	**10m** · 1h · 1d — window in which failures are counted.
`increment`	**yes** · no — repeated bans get longer.

10.3 Protected services

Only services that are installed can be switched on: `sshd`, `recidive` (repeat offenders: one week after 5 bans in a day), `nginx-http-auth`, `nginx-botsearch`, `apache-auth`, `dovecot`, `postfix`.

```
# stratum security toggles jails
# stratum security set jails sshd recidive nginx-http-auth
```

10.4 Hardening

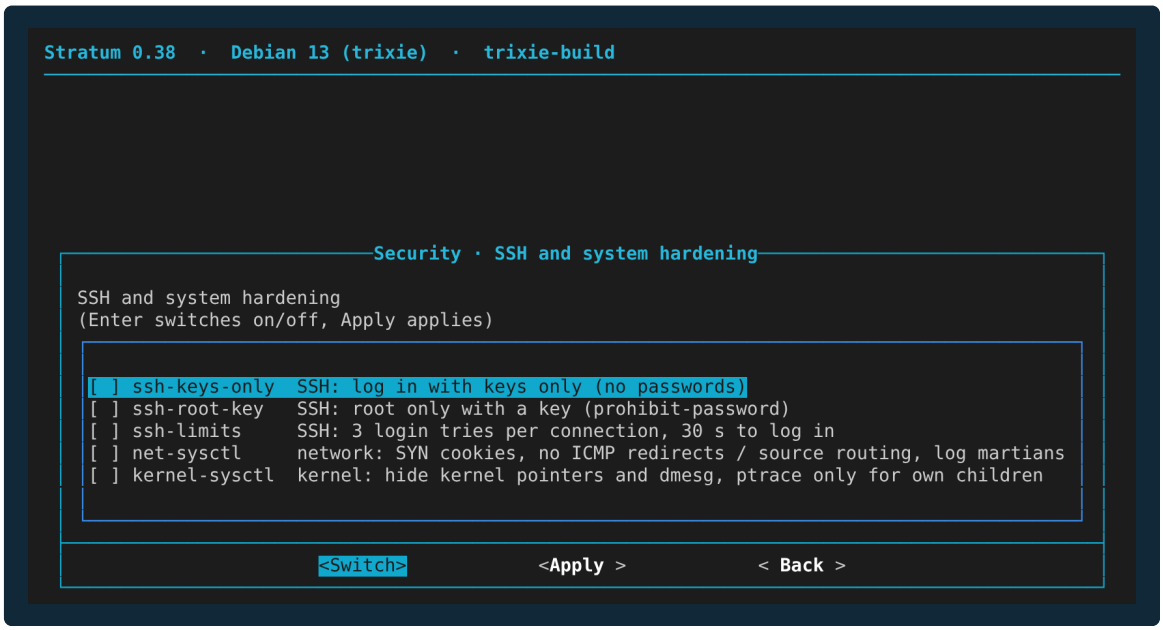


Figure 17 Hardening switches — all off by default.

Switch	Effect
`ssh-keys-only`	SSH only with keys, no passwords. Refused while no key is installed.
`ssh-root-key`	root only with a key (`PermitRootLogin prohibit-password`).
`ssh-limits`	3 login tries per connection, 30 seconds to log in.
`net-sysctl`	SYN cookies, no ICMP redirects or source routing, log martians.
`kernel-sysctl`	Hide kernel pointers and dmesg, ptrace only for own children.

> **WARNING**
>
> SSH changes must be confirmed within **120 seconds** (`stratum security confirm`, or **Keep** in the menu), otherwise they are undone. Test a *new* login in a second window before you confirm — your existing connection keeps working either way and proves nothing.

© 2026 Stratum Project

10.5 Whitelist and bans

Addresses on the whitelist are never banned. When fail2ban starts, Stratum adds the addresses of all open SSH connections automatically — so you cannot ban yourself by mistyping a password.

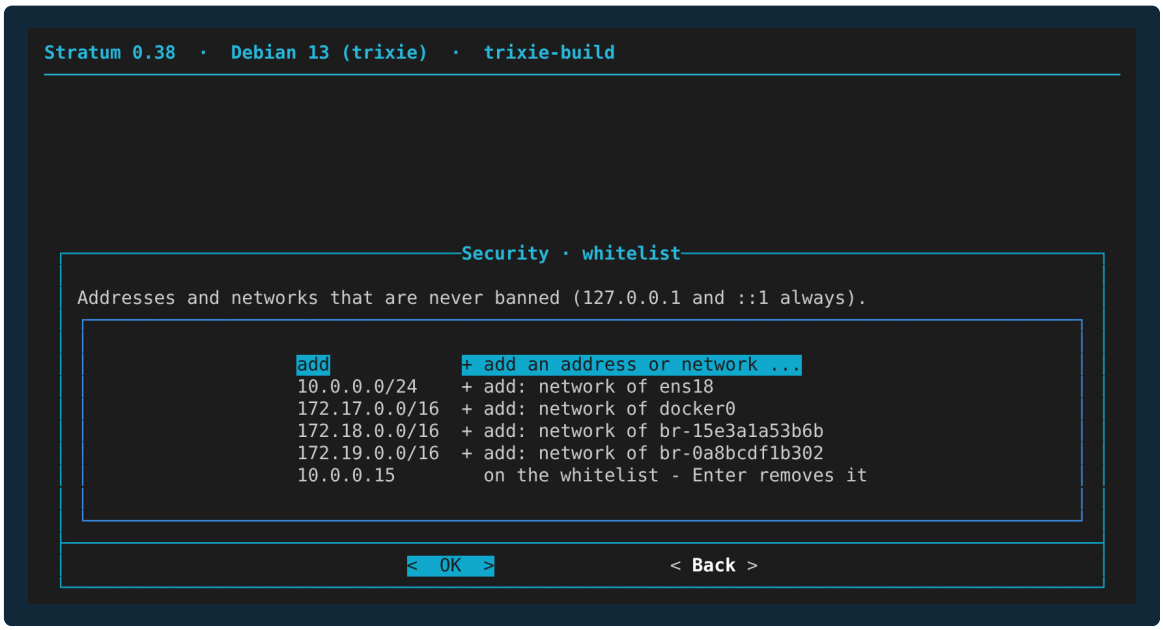


Figure 18 The whitelist.

```
# stratum security whitelist add 192.168.1.0/24
# stratum security banned
# stratum security ban 203.0.113.7
# stratum security unban 203.0.113.7
```

10.6 Events — what happened today? END USER ADMINISTRATOR

`stratum security events [hours]` shows only what matters: SSH logins (who, from where, how), failed logins, bans, sudo use and Docker trouble (crashed, out of memory, unhealthy).

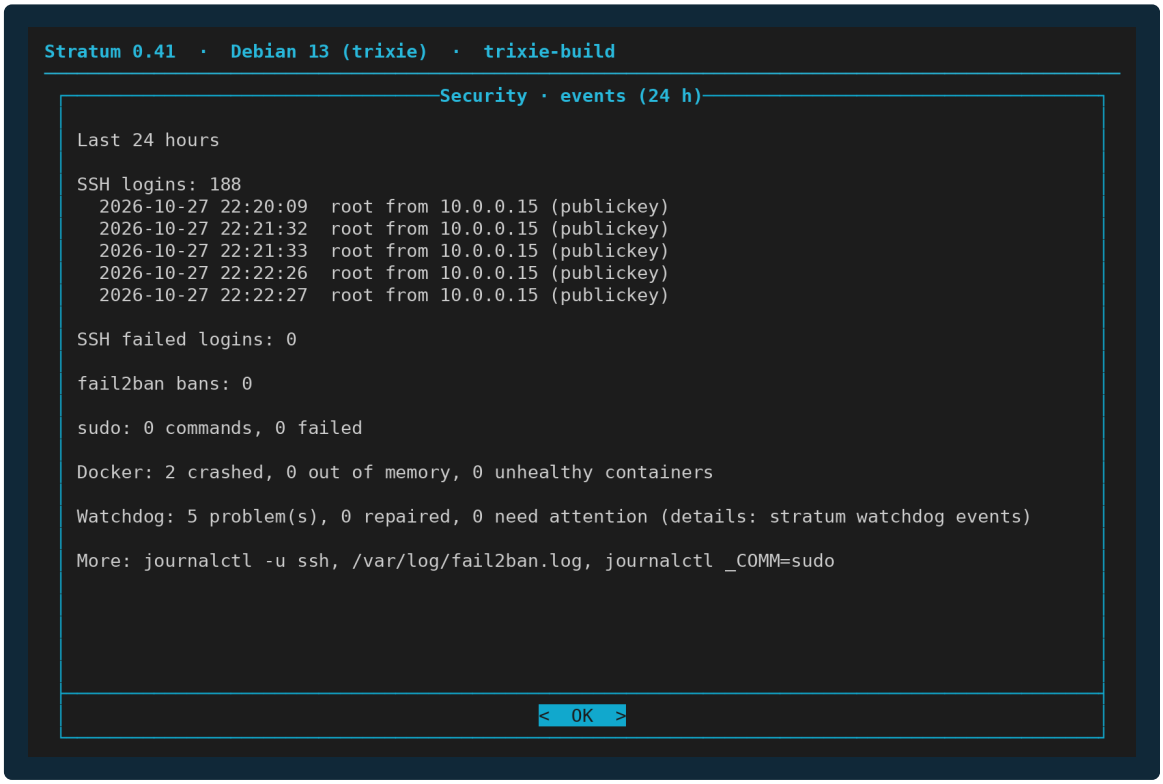


Figure 19 Important events of the last 24 hours.

```
# stratum security events
Last 24 hours

SSH logins: 4
  2026-10-24 20:37:15  root from 10.0.0.15 (publickey)
  …
SSH failed logins: 0
fail2ban bans: 0
sudo: 0 commands, 0 failed
Docker: 0 crashed, 0 out of memory, 0 unhealthy containers
```

10.7 All commands

Command	What it does
`status`	State, settings, services, hardening, whitelist, bans, event summary.
`start` · `stop` · `enable` · `disable` · `restart`	fail2ban now / at boot; restart rewrites the configuration.
`settings` · `setting <key> <value>`	Show / change settings.
`toggles jails|hardening`	Items with their state.
`set jails|hardening [id…]`	Exactly these on.
`whitelist [add|remove <addr>]`	Never-banned addresses and networks.
`banned` · `ban <addr> [service]` · `unban <addr>`	Bans.
`events [hours]`	Important events (default 24 h).
`confirm` · `rollback`	Keep / undo the last SSH change.

© 2026 Stratum Project

Files: `/etc/stratum/security.conf`, `/etc/fail2ban/jail.d/90-stratum.local`, `/etc/ssh/sshd_config.d/10-stratum.conf`, `/etc/sysctl.d/90-stratum-security.conf`. Your own `jail.local` is never touched. Environment: `STRATUM_FORCE=1`, `STRATUM_SECURITY_NOCONFIRM=1`.

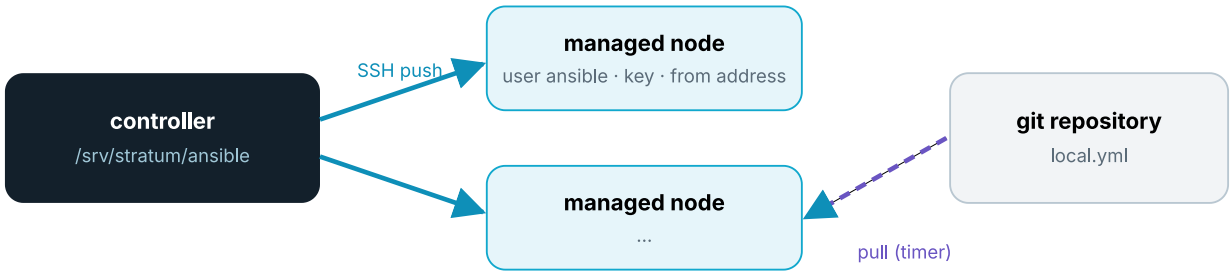
CHAPTER 11

Automation — stratum ansible

ADMINISTRATOR

When you look after more than a handful of computers, you want to describe their configuration once and apply it everywhere. Ansible does that; Stratum sets it up in three roles you can switch on independently.

11.1 Three roles



Push from a controller, or let each computer pull its configuration from git — or both.

Role	What it means
managed	A controller may manage this computer: user `ansible`, controller keys, allowed only from given addresses.
pull	This computer applies a playbook from a git repository by itself, on a timer.
controller	This computer manages others: inventory, playbooks, roles and a run log in `/srv/stratum/ansible`.

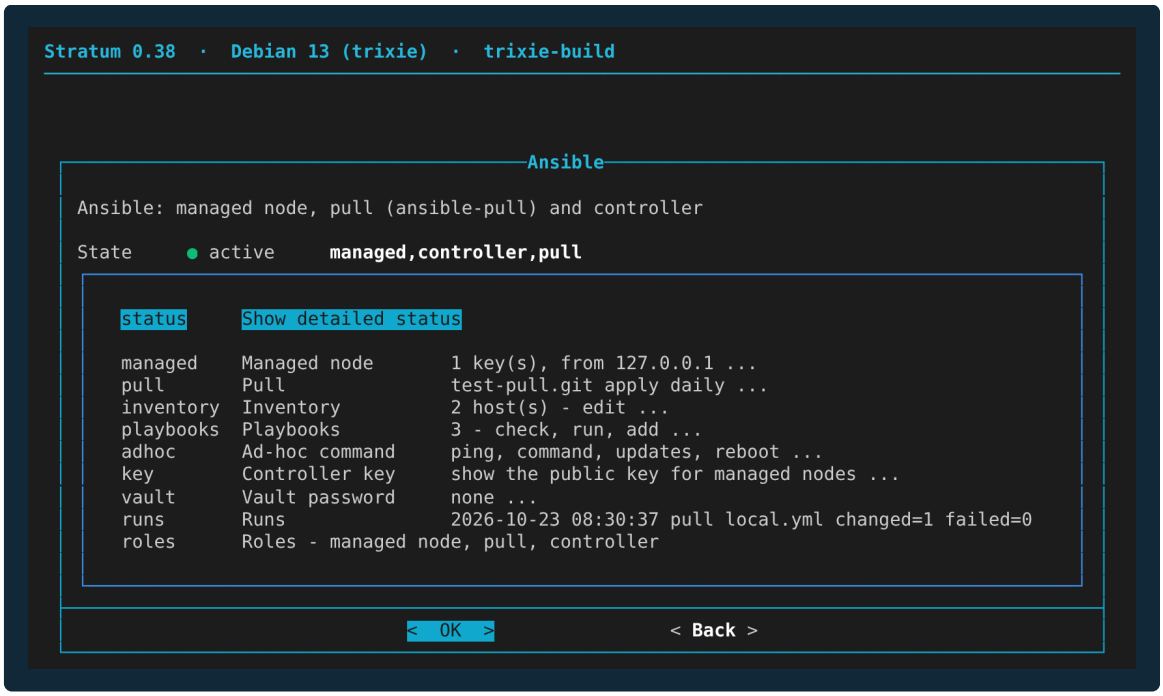


Figure 20 The Ansible module in the menu.

11.2 Managed node

```
# stratum ansible set roles managed
# stratum ansible managed key add 'ssh-ed25519 AAAAC3Nz… controller'
# stratum ansible managed allow 10.0.0.20
# stratum ansible managed firewall          # allow SSH from 10.0.0.20
```

11.3 Pull

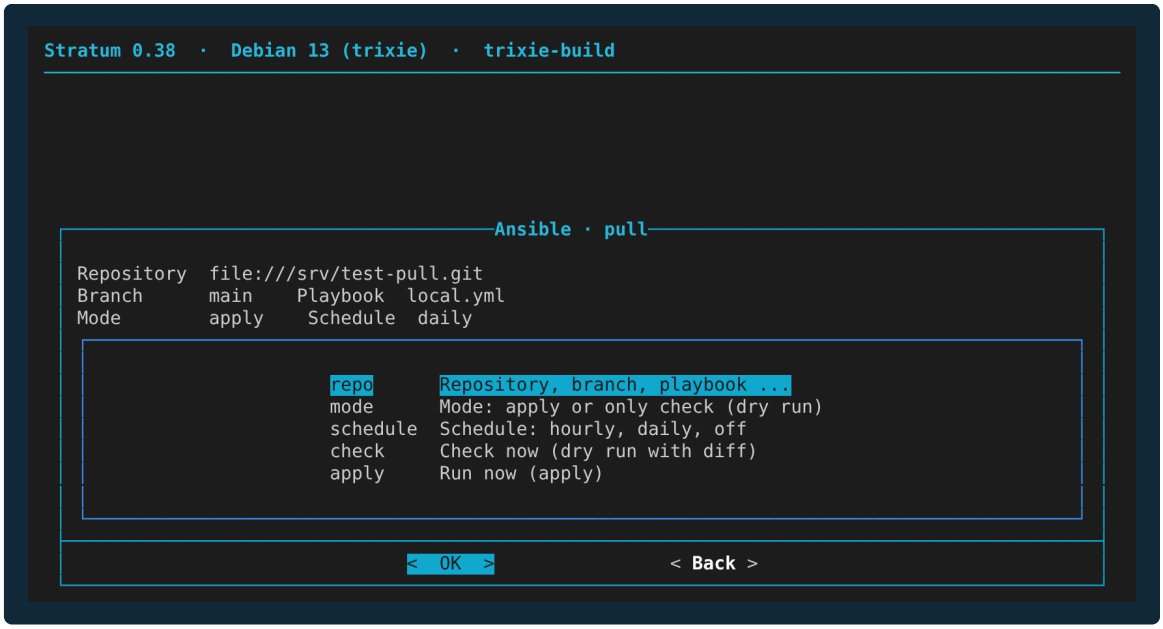


Figure 21 Pull from a git repository.

```
# stratum ansible pull set https://git.example.org/infra.git main local.yml
# stratum ansible pull mode check            # first only show what would change
# stratum ansible pull run --check
# stratum ansible pull mode apply
# stratum ansible pull schedule daily
```


© 2026 Stratum Project

11.4 Controller

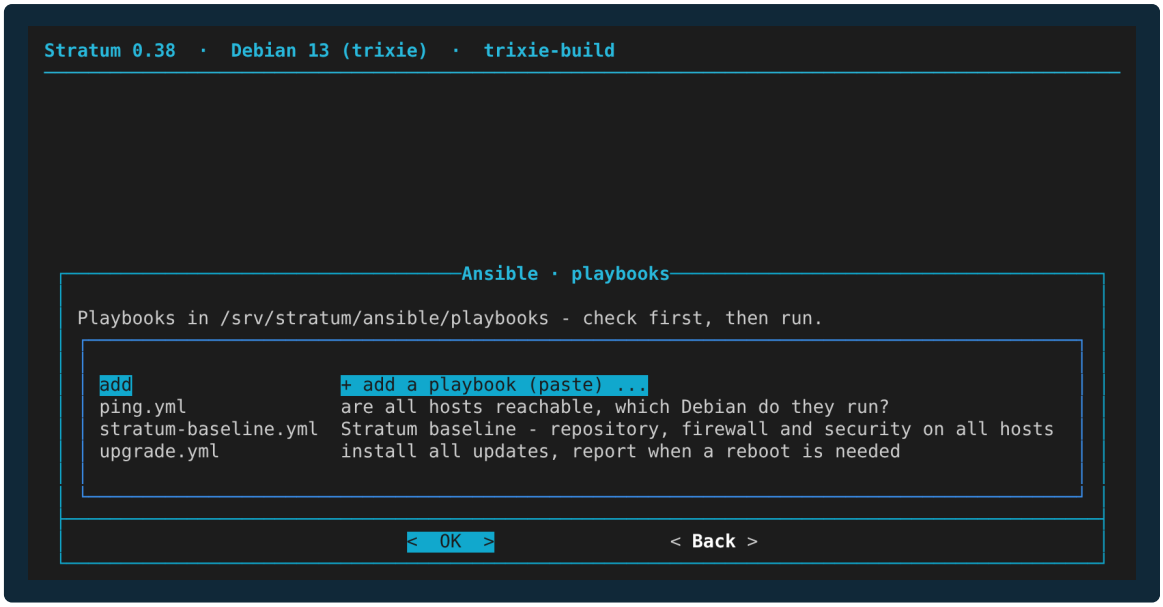


Figure 22 Playbooks on the controller.

```
# stratum ansible controller key            # give this key to the managed nodes
# stratum ansible hosts
# stratum ansible ping
# stratum ansible run upgrade --check --limit web
# stratum ansible adhoc web ansible.builtin.shell "uptime"
# stratum ansible runs
2026-10-23 08:30:25 adhoc      ping @all               rc=0 ok=0 changed=0 failed=0 unreachable=0
2026-10-23 08:30:27 playbook   ping.yml (check)        rc=0 ok=4 changed=0 failed=0 unreachable=0
```

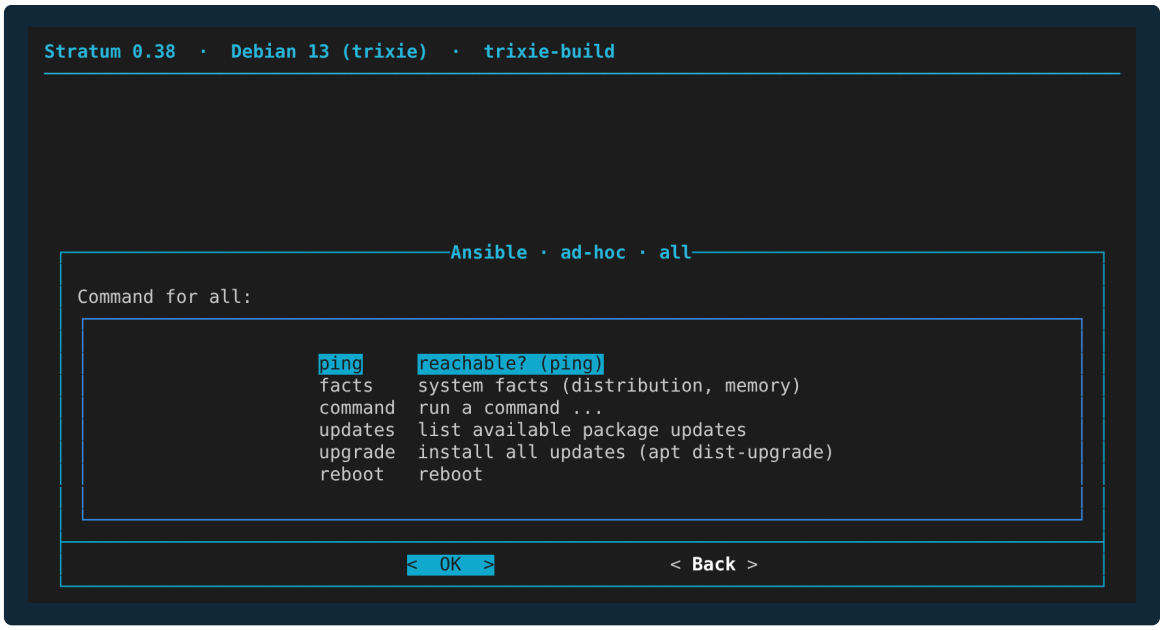


Figure 23 An ad-hoc command.

Example playbooks to start from are in `/usr/lib/stratum/ansible/examples`: **ping**, **upgrade** and **stratum-baseline** (installs Stratum modules and sets them up the same way everywhere).

> **TIP**
> Always run a new playbook with `--check` first. It shows what *would* change, and changes nothing.

11.5 All commands

Command	What it does
`toggles roles` · `set roles [managed] [pull] [controller]`	Roles and their state; exactly these on.
`managed key list|add <key|->|remove <n>`	Controller keys.
`managed allow|deny <addr>` · `managed firewall`	Allowed source addresses; firewall rule for them.
`pull set <url> [branch] [playbook]`	Repository (default: main, local.yml).
`pull mode apply|check` · `pull schedule hourly|daily|off` · `pull run [--check]`	Pull behaviour, timer, run now.
`controller key` · `hosts` · `ping [pattern]`	Controller key, inventory tree, reachability.
`playbooks` · `playbook-add <name> [file|-]`	List; add (syntax checked first).
`run <playbook> [--check] [--limit <hosts>]`	Run a playbook.
`adhoc <pattern> <module> [args]`	One module on some hosts.
`galaxy` · `vault set|remove`	Roles from requirements.yml; vault password (stdin).
`runs [n]`	The last runs: ok / changed / failed.

Files: `/etc/stratum/ansible.conf`, `/srv/stratum/ansible` (inventory.ini, playbooks/, roles/, ansible.cfg), `/var/log/stratum-ansible`.

© 2026 Stratum Project

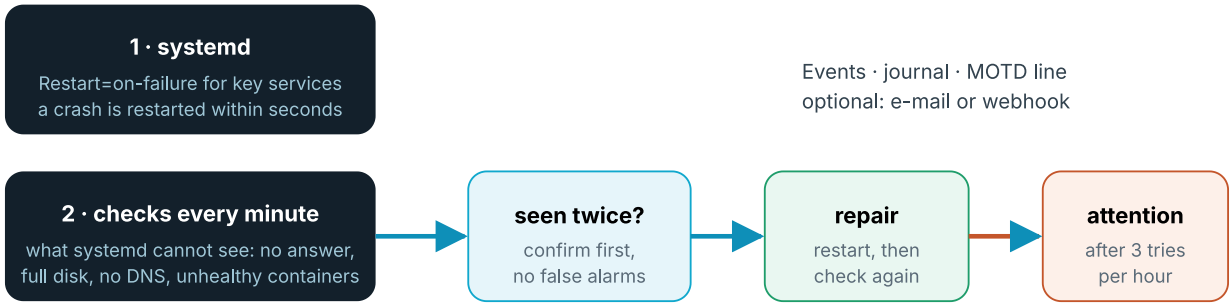
CHAPTER 12

Self-healing — stratum watchdog

ADMINISTRATOR END USER

Services crash, disks fill up, a daemon hangs at three in the morning. The **watchdog** notices, repairs what can safely be repaired, and tells you about the rest — so that small problems do not become outages.

12.1 Two layers



systemd restarts crashed services at once; the checks catch everything else.

1. **systemd.** Key services (ssh, cron, rsyslog, chrony or timesyncd, fail2ban, Docker, containerd) are restarted by systemd itself when they crash. Most of them have their own restart policy already; the ones that do not get a small drop-in from Stratum.
2. **Checks.** A timer runs the checks every minute. A problem that can be repaired is repaired when it is still there in the next round — the restarted service must still run three seconds later. After three repairs of the same thing within an hour, the watchdog stops trying: the problem *needs attention*.

> **REMEMBER**
>
> The watchdog never reboots the computer (unless you switch on the hardware watchdog), never touches the network configuration, never starts a service you stopped or disabled yourself, and does not repair anything while apt or dpkg is working.

12.2 In the menu END USER ADMINISTRATOR



Figure 24 The watchdog in the menu: state, settings, problems and events.

The watchdog runs as soon as the package is installed, with defaults for your type of computer. **Problems now** shows what the last round found; **Events** shows what happened in the last 24 hours.

© 2026 Stratum Project

12.3 What is checked

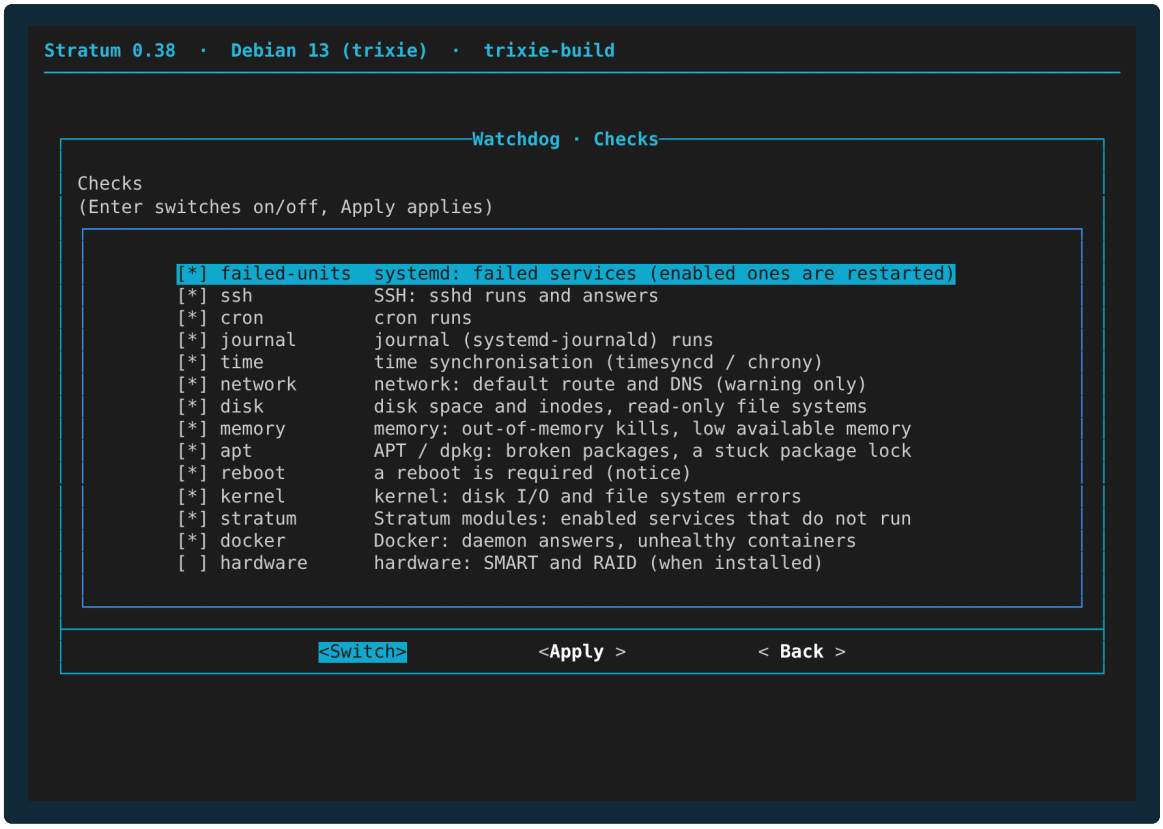


Figure 25 The checks — switch each one on or off.

Check	What it looks at	Repair
`failed-units`	Failed systemd services.	Restart, if the service is enabled.
`ssh`	sshd runs and answers on its port.	Restart ssh.
`cron`, `journal`	cron and the journal run.	Restart.
`time`	Time synchronisation (timesyncd, chrony).	Restart the service.
`network`	Default route, DNS.	Warning only.
`disk`	Space and inodes; read-only file systems.	Optional cleanup: `apt clean`, journal to 200 MB.
`memory`	Out-of-memory kills, little free memory.	Warning.
`apt`	Broken packages, a stuck package lock.	Warning.
`reboot`	A reboot is required.	Notice.
`kernel`	Disk I/O and file system errors.	Warning.
`stratum`	Stratum modules that are enabled but do not run.	Start the module.
`docker`	Docker answers; unhealthy containers.	Restart Docker / the container.
`hardware`	SMART and software RAID (when installed).	Warning.

12.4 Examples

```
# stratum watchdog run
No problems.
# systemctl stop cron            # a service stops unexpectedly ...
# stratum watchdog run cron
  repaired  cron/cron.service            cron.service does not run (repaired: systemctl restart cron.service)
# stratum watchdog events
Watchdog events in the last 24 hours

Problems:   1
Repaired:   1
Attention:  0

Last 15:
  2026-10-25 09:22:43  critical  cron/cron.service        cron.service does not run
  2026-10-25 09:22:46  repaired  cron/cron.service        cron.service does not run - repaired: ...
```

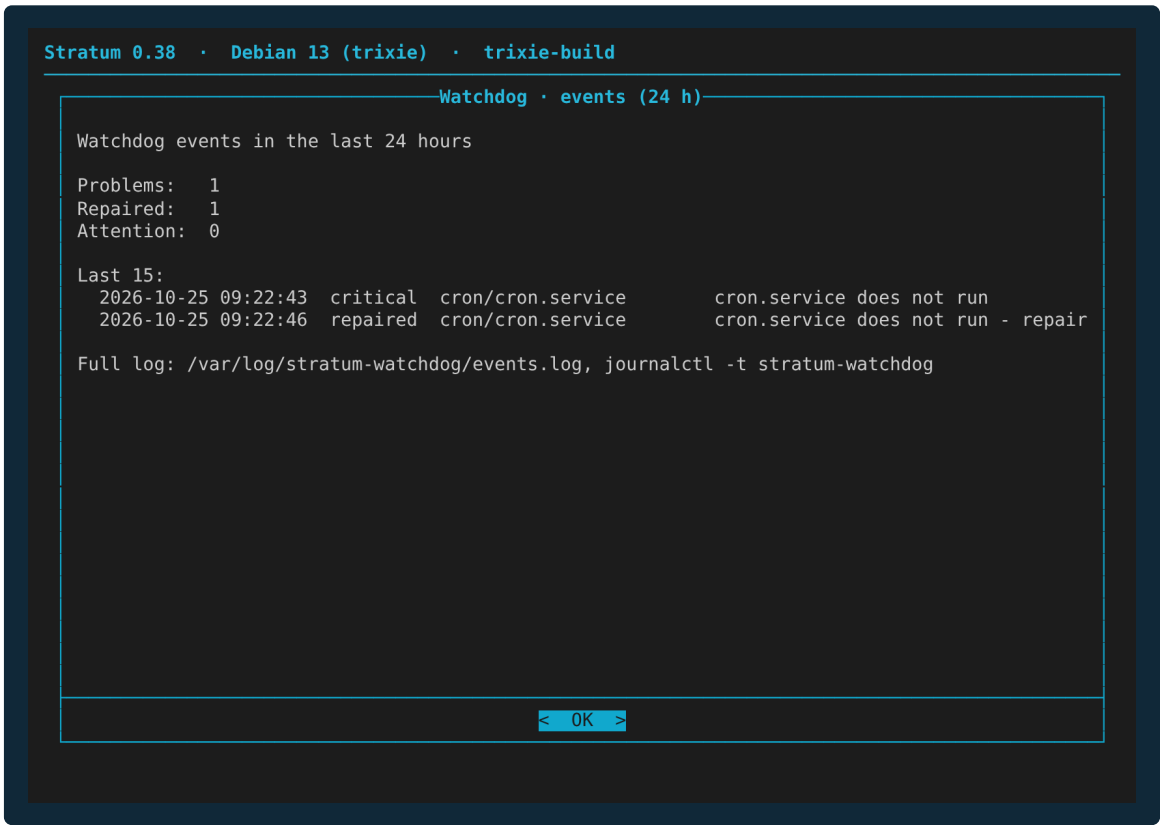


Figure 26 Events in the menu.

When a problem needs attention, the message of the day says so after login:

```
Watchdog: 1 problem(s), 1 need attention - stratum watchdog status
```

> **TIP**
>
> Planned maintenance? `stratum watchdog pause 60` pauses repairs for an hour (checks go on); `stratum watchdog resume` ends the pause early.

© 2026 Stratum Project

12.5 Settings

Setting	Values and meaning
`type`	workstation · server · hypervisor · vm-guest — checks and limits of the type.
`interval`	**1** · 2 · 5 · 10 · 15 minutes between rounds.
`restart-limit`	1–10, default **3** restarts of one target per hour.
`disk-warn` · `disk-crit`	% used for a warning (default 90) and critical (95; hypervisor 85/92).
`cleanup`	**yes** · no — clean the package cache and the journal when `/` or `/var` is critical.
`systemd-restart`	**yes** · no — Restart=on-failure drop-ins.
`notify` · `webhook`	**none** · mail (to root) · webhook (JSON POST to a URL, e.g. a chat).
`hwwatchdog`	**no** · yes — systemd feeds `/dev/watchdog`; a hung system is reset after 60 s.
`motd`	**yes** · no — a line after login when there are problems.

> **WARNING**
> The hardware watchdog resets the computer without shutting it down when the system hangs. Switch it on only for computers that must come back by themselves, and only when `/dev/watchdog` exists.

12.6 Own checks

Put an executable file in `/etc/stratum/watchdog.d/<name>.check`. Exit code 0 means ok, 1 a warning, 2 critical; the first line of output is the message. A line `# restart: <unit>` makes the watchdog restart that unit.

```
#!/bin/sh
# /etc/stratum/watchdog.d/website.check
# restart: nginx.service
curl -fsS -m 5 -o /dev/null http://localhost/ && exit 0
echo "the web site does not answer"; exit 2
```

12.7 All commands

Command	What it does
`status`	State, settings, checks, problems, last 24 hours.
`start` · `stop` · `enable` · `disable` · `restart`	As for every service module.
`run [check…]`	Run the checks now (repairs at once) and show the result.
`events [hours]`	Problems, repairs, needs attention (default 24 h).
`pause [minutes]` · `resume`	Pause repairs (default 60, max 1440) / resume.
`settings` · `setting <key> <value>`	Show / change settings.
`toggles checks` · `set checks [check…]`	Checks with their state / exactly these on.

The watchdog also appears in `stratum security events` and in the support report. Files: `/etc/stratum/watchdog.conf`, `/etc/stratum/watchdog.d/`, `/var/log/stratum-watchdog/events.log`, `/var/lib/stratum/watchdog/`; journal: `journalctl -t stratum-watchdog`.

© 2026 Stratum Project

CHAPTER 13

File server — stratum fileserver

ADMINISTRATOR END USER

Shared folders for the office or the home: Windows, macOS and Linux computers connect with **Samba**, Linux and Unix servers with **NFS**. You say which folder is shared, how, and who may read or write — Stratum writes the Samba and NFS configuration, the folder permissions and the firewall rules.

13.1 The idea

Shares

A folder with a name (`docs` → `\\server\docs`, `server:/srv/share/docs`) and the protocols it is offered with: smb, nfs or both.

Users and groups

Users *only for file sharing* (no login on the server) or existing users with a Samba password; groups `share-<name>` for teams. Access per share: **write** and **read**.

13.2 In the menu END USER ADMINISTRATOR

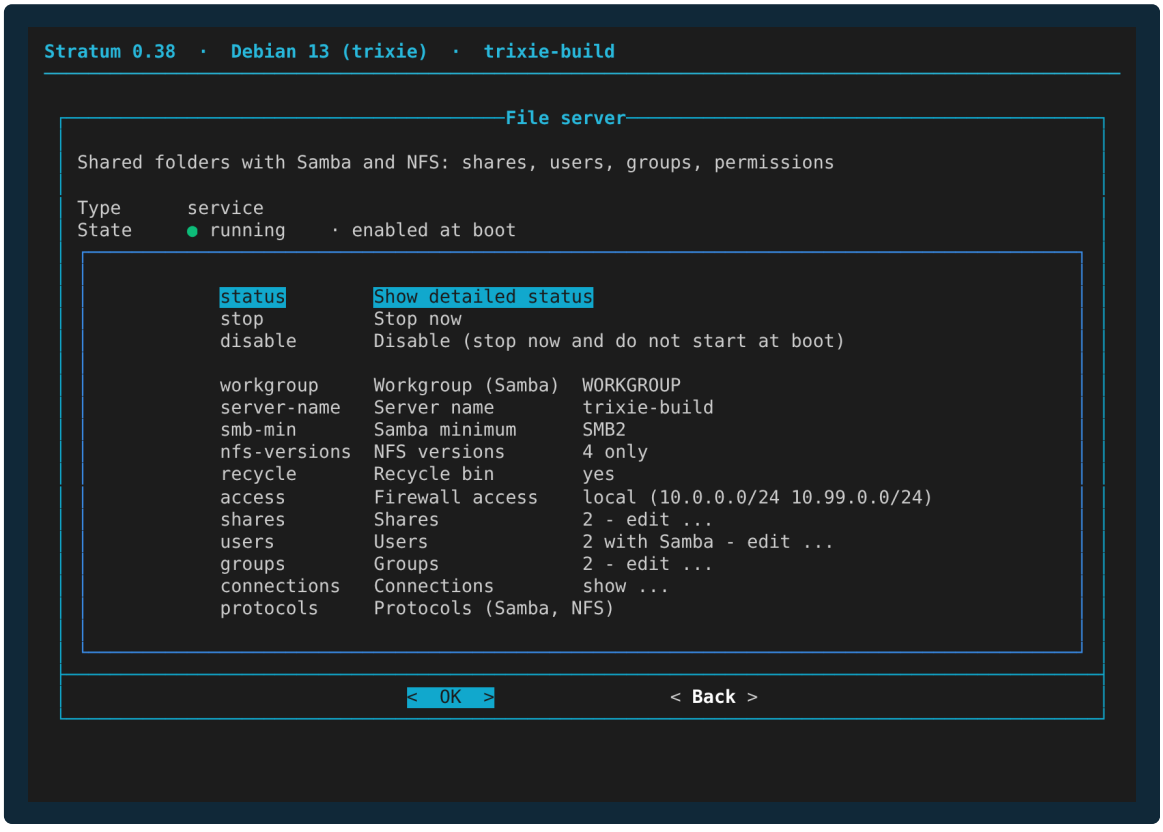


Figure 27 The file server: settings, shares, users, groups and connections.

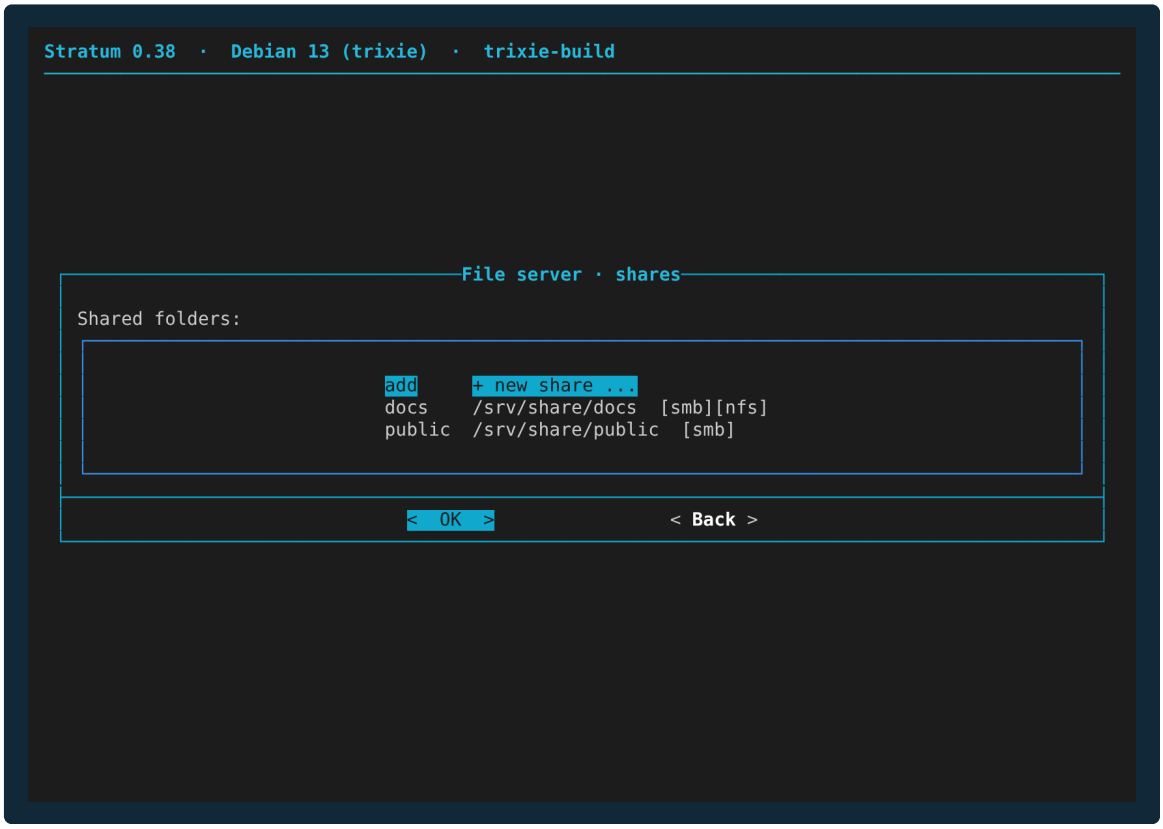


Figure 28 Shares and their protocols.

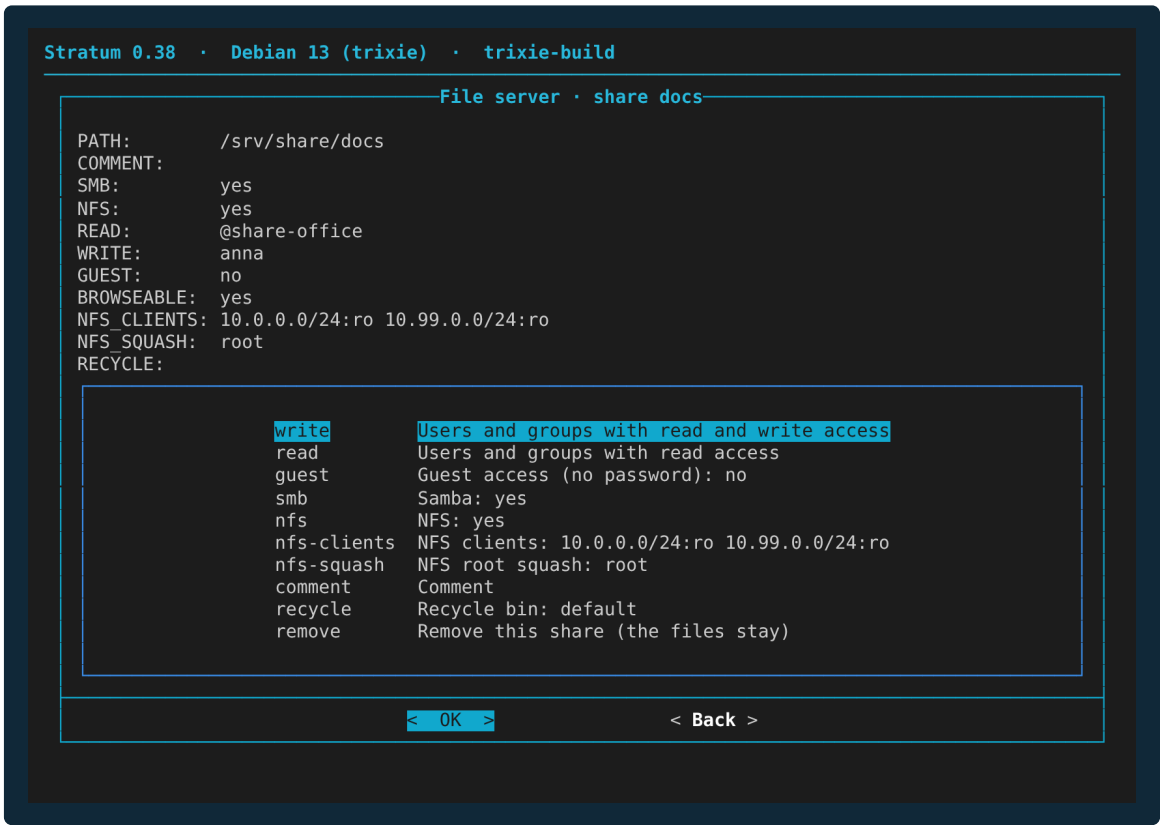


Figure 29 One share: who may write and read, protocols, NFS clients.

13.3 A shared folder in five commands

© 2026 Stratum Project

```
# stratum fileserver share-add docs /srv/share/docs both
  ✓ share docs: /srv/share/docs (Samba NFS)
# stratum fileserver user-add anna                # asks for the password twice
# stratum fileserver group-add office && stratum fileserver group-member office add bojan
# stratum fileserver share docs set write anna
# stratum fileserver share docs set read @share-office
# stratum fileserver shares
SHARE          PATH                       PROTOCOL  ACCESS
docs           /srv/share/docs            smb+nfs   rw: anna  ro: @share-office
```

On Windows: `\\trixie-build\docs` with the user `anna`. On Linux: `smb://trixie-build/docs`, or NFS: `mount trixie-build:/srv/share/docs /mnt`.

> **TIP**
> Permissions are set on the folder itself (POSIX ACLs), so they also hold for NFS and for local users. A new file inherits them.

13.4 Share settings

share <name> set …	Meaning
`write "<user|@group> …"`	Read and write access.
`read "<user|@group> …"`	Read access.
`guest no|read`	Guests without a password may read (Samba).
`smb yes|no` · `nfs yes|no`	Protocols of the share.
`nfs-clients "<net>:rw|ro …"`	Which computers may mount it with NFS, e.g. `"10.0.0.0/24:rw 10.0.0.5:ro"`.
`nfs-squash root|all|none`	root of an NFS client (or everybody) becomes *nobody* — the default `root` is the safe one.
`comment` · `browseable` · `recycle`	Description · visible in the network · recycle bin `.recycle/<user>`.

> **REMEMBER**
> NFS knows computers, not passwords: whoever controls an allowed computer can use the share with the user IDs of that computer. Use NFS between servers you trust, Samba for people.

13.5 Settings and commands

Setting	Meaning
`workgroup` · `server-name`	Samba workgroup (WORKGROUP) · name in the network.
`smb-min SMB2|SMB3`	Oldest Samba protocol allowed (SMB1 is never allowed).
`nfs-versions 4|3,4`	NFSv4 only (one port, 2049) or also NFSv3 (fixed ports for the firewall).
`recycle yes|no`	Recycle bin for new shares.
`access local|any|"<net> …"`	Who may connect: the local networks (default; also the WireGuard mesh), everybody, or the networks you name — firewall rules.

Command	What it does
`shares` · `share-add` · `share-remove`	List · add · remove a share (the files stay).
`users` · `user-add` · `user-enable` · `user-passwd` · `user-remove`	Users of the shares.
`groups` · `group-add` · `group-remove` · `group-member`	Groups `share-<name>`.
`connections`	Who is connected now (Samba and NFS).
`apply` · `restart`	Write the configuration again and reload / restart.

Files: `/etc/stratum/fileserver.conf`, `/etc/stratum/fileserver.d/<share>.share`, `/etc/samba/stratum-*.conf` (included in `smb.conf`), `/etc/exports.d/stratum.exports`, `/etc/nfs.conf.d/90-stratum.conf`.

© 2026 Stratum Project

CHAPTER 14

Database — stratum database

ADMINISTRATOR

Many applications need a database. The **database** module installs the **MariaDB** server from Debian and lets you create databases and users, give users access, and keep daily backups — without writing SQL.

14.1 In the menu END USER ADMINISTRATOR

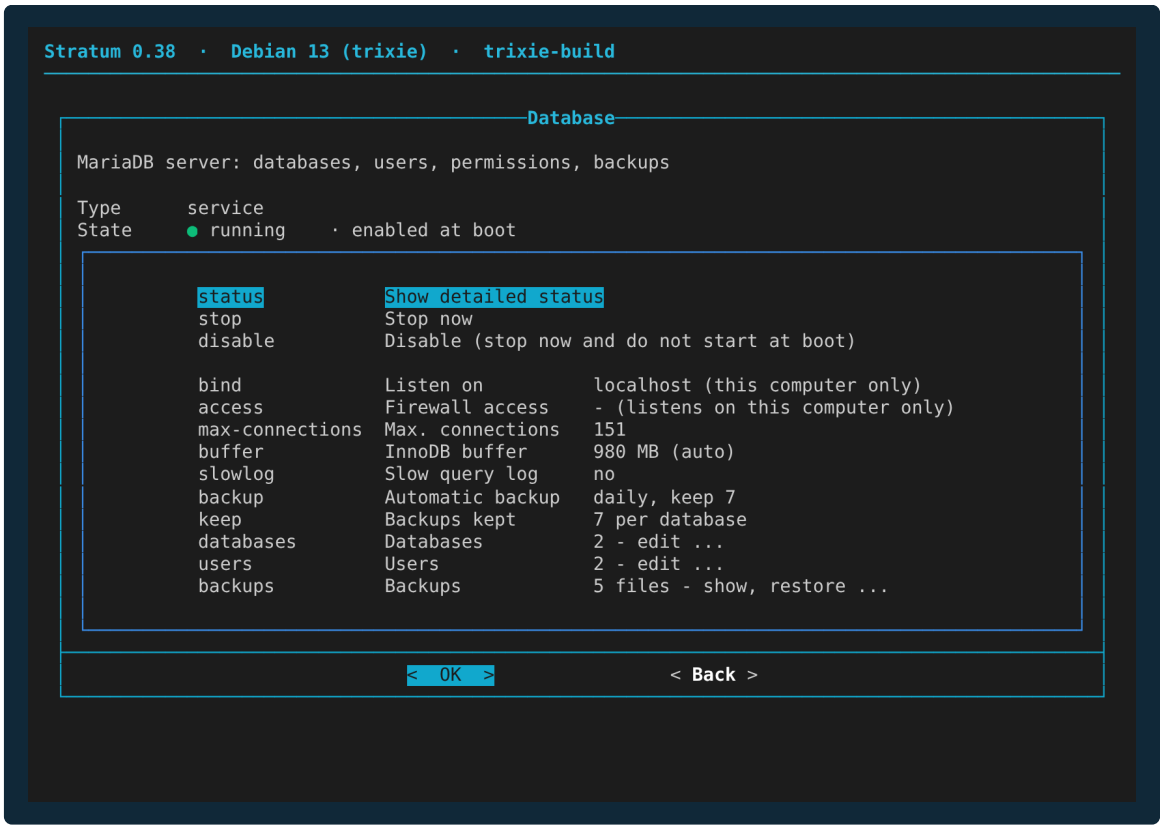


Figure 30 The database module: settings, databases, users and backups.

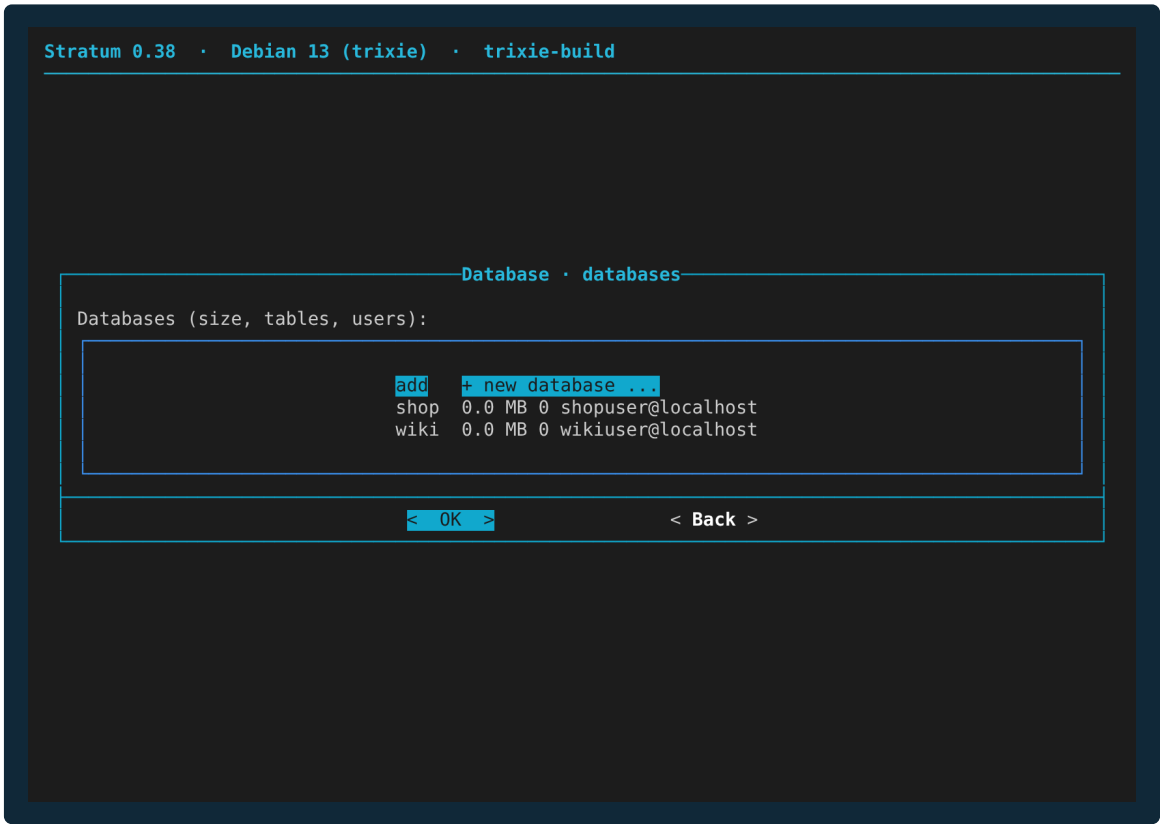


Figure 31 Databases with size, tables and users.

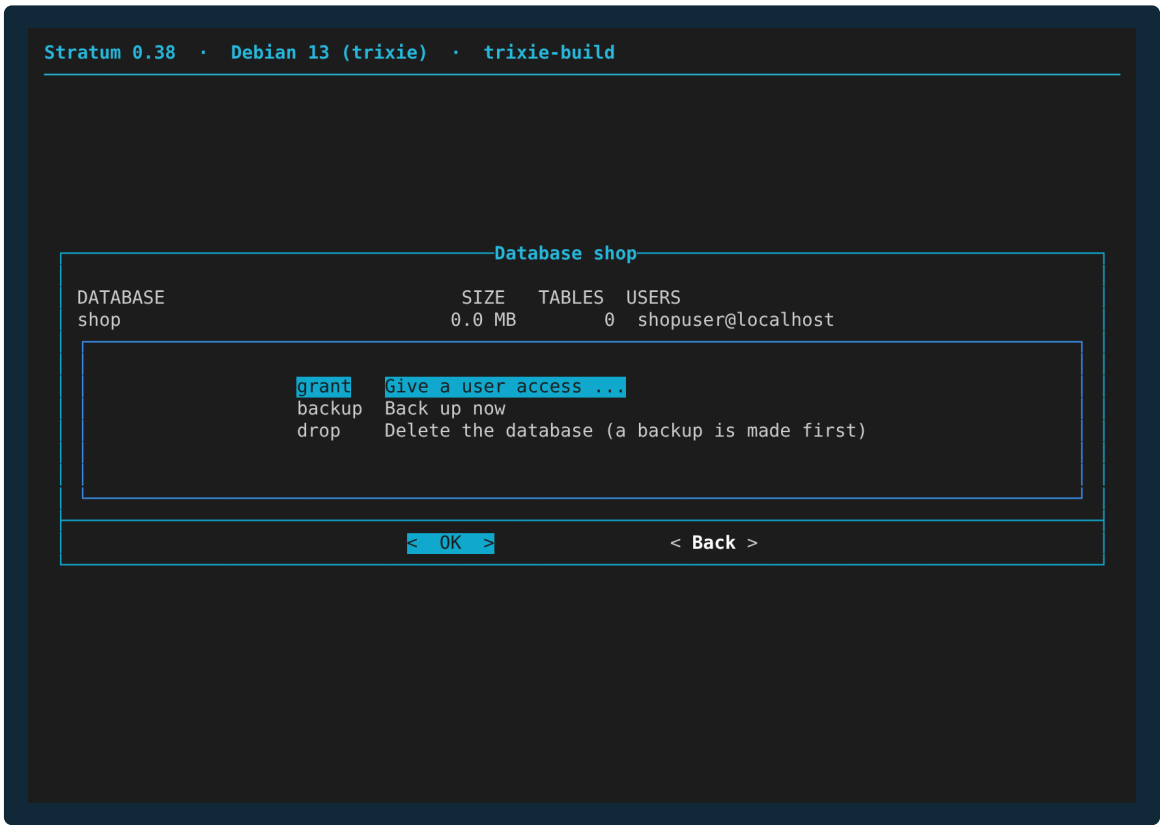


Figure 32 One database: give a user access, back it up, delete it.

14.2 A database for an application



© 2026 Stratum Project

```
# stratum database db-create shop
# stratum database user-create shopuser             # asks for the password
# stratum database grant shopuser shop all
# stratum database dbs
DATABASE                          SIZE   TABLES  USERS
shop                            0.0 MB        0  shopuser@localhost
```

The application connects as `shopuser` to `localhost`, database `shop`. root of this computer logs in to MariaDB without a password (`mariadb` on the command line) — through the local socket only.

Level	What the user may do in the database
`read`	Read (SELECT).
`write`	Read and change data (INSERT, UPDATE, DELETE …), not the tables.
`all`	Everything in this database, also tables — what most applications need.
`admin`	All databases, and may give access to others (`*` instead of a database name).

> **TIP**
> Users have a host: `localhost` (default), `10.0.0.%` (a network) or `%` (everywhere). An application on another computer needs a user for its address — and MariaDB must listen on the network (`setting bind all`).

14.3 Backups

Every night at 03:15 each database is saved as a compressed SQL file in `/var/backups/stratum-database`; the last 7 per database are kept. `db-drop` always makes a backup first.

```
# stratum database backup shop
# stratum database backups
# stratum database restore /var/backups/stratum-database/shop-20261026-135529.sql.gz shop_test
```

14.4 Settings and commands

Setting	Meaning
`bind localhost|all|<address>`	Where MariaDB listens. Default: this computer only.
`access local|any|"<net> …"`	Who may connect when it listens on the network (firewall rule for port 3306).
`max-connections` · `buffer`	Maximum connections (151) · InnoDB buffer pool (auto: 25 % of the memory).
`slowlog yes|no`	Log queries slower than 2 seconds.
`backup daily|weekly|off` · `keep <n>`	Automatic backups and how many are kept.

Command	What it does
`dbs` · `db-create <name> [charset]` · `db-drop <name> --yes`	Databases.
`users` · `user-create` · `user-passwd` · `user-drop <user> [host]`	Users.

Command	What it does
`grant <user> [host] <db|*> <level>` · `revoke` · `grants`	Access.
`backup [db|all]` · `backups` · `restore <file> [db]`	Backups.
`secure`	Remove anonymous users, remote root and the test database.
`processes`	Running queries.

© 2026 Stratum Project

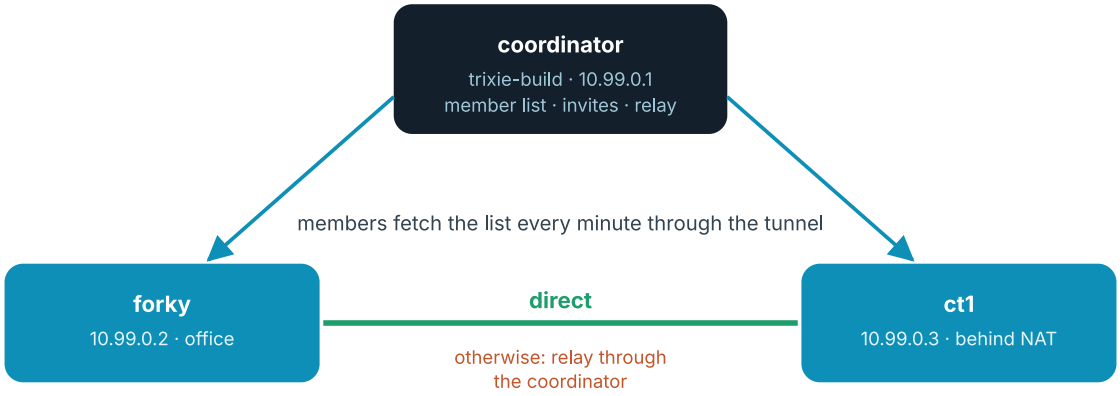
CHAPTER 15

WireGuard mesh — stratum wireguard

ADMINISTRATOR

Your computers are in different places — the office, home, a data centre, a laptop on the road. The **wireguard** module connects them into one private network, a *mesh*: every computer reaches every other one by name, encrypted with WireGuard, as if they were in the same room.

15.1 How it works



Every member talks to every other member directly; the coordinator helps them find each other.

1. **The coordinator** creates the mesh and invites the others. It must be reachable on its UDP port (51820) by all members: a server with a public address, or a computer with a port forwarding on the router.
2. **An invitation** is a one-time code. The new member runs `stratum wireguard join <code>` and is connected — no keys to copy by hand.
3. **The member list** (names, keys, addresses, where each member is reached) is served by the coordinator through the tunnel; every member fetches it every minute and connects to the others directly.
4. **Behind NAT.** Members behind a router find each other with the help of the addresses the coordinator sees (NAT hole punching). When two members still cannot reach each other, their traffic goes through the coordinator (relay) — slower, but it works. The path is chosen again every minute.

15.2 A mesh in three commands

On the coordinator:

```
# stratum wireguard create lab
==> creating the mesh lab (10.99.0.0/24), this computer is the coordinator
  ✓ mesh lab: trixie-build = 10.99.0.1, members reach it at 10.0.0.21:51820
# stratum wireguard invite forky
  ✓ forky invited: address 10.99.0.2

On forky run (the code is secret - it contains the key of forky):

  stratum wireguard join SWG1:H4sIAAAAAAACAz1Py27CMAC751cmlSTi0UzKoVA2yuhLFLr2MiUk26p…
```

On the new member:

```
# stratum wireguard join SWG1:H4sIAAAAAAACAz1Py27CMAC751cmlSTi0UzKoVA2yuhLFLr2MiUk26p…
==> joining the mesh lab as forky (10.99.0.2)
  ✓ connected to the coordinator 10.99.0.1 (10.0.0.21:51820); members: 3
# ping trixie-build.lab
```

WARNING

The invite code contains the private key of the new member. Send it like a password, through a channel you trust, and only to that computer. Until it is used, `stratum wireguard invite-show <name>` shows it again; `remove <name>` cancels it.

TIP

Members on the internet need the public name or address of the coordinator: `stratum wireguard create office --endpoint vpn.example.org`, or later `setting endpoint vpn.example.org`. The default (*auto*) is the coordinator's local address — right when all computers are in one network.

15.3 In the menu END USER ADMINISTRATOR

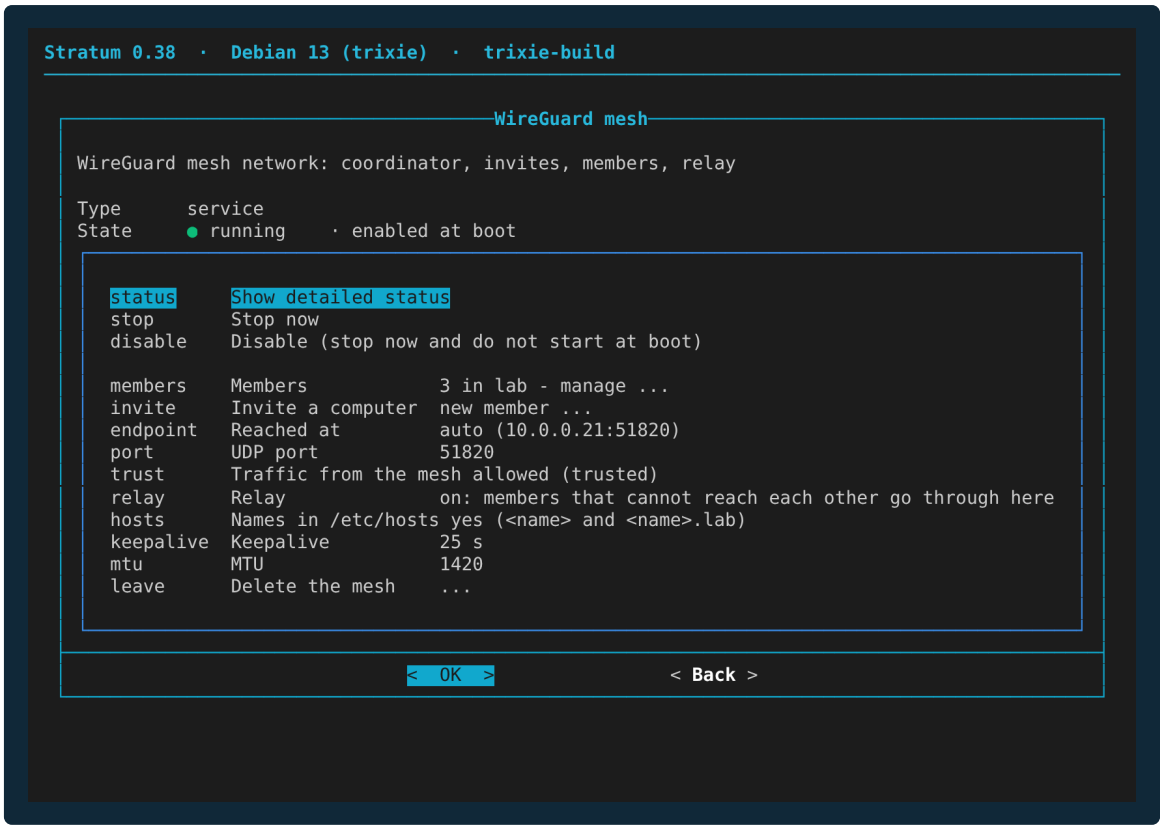


Figure 33 The mesh on the coordinator: members, invitations and settings.

© 2026 Stratum Project

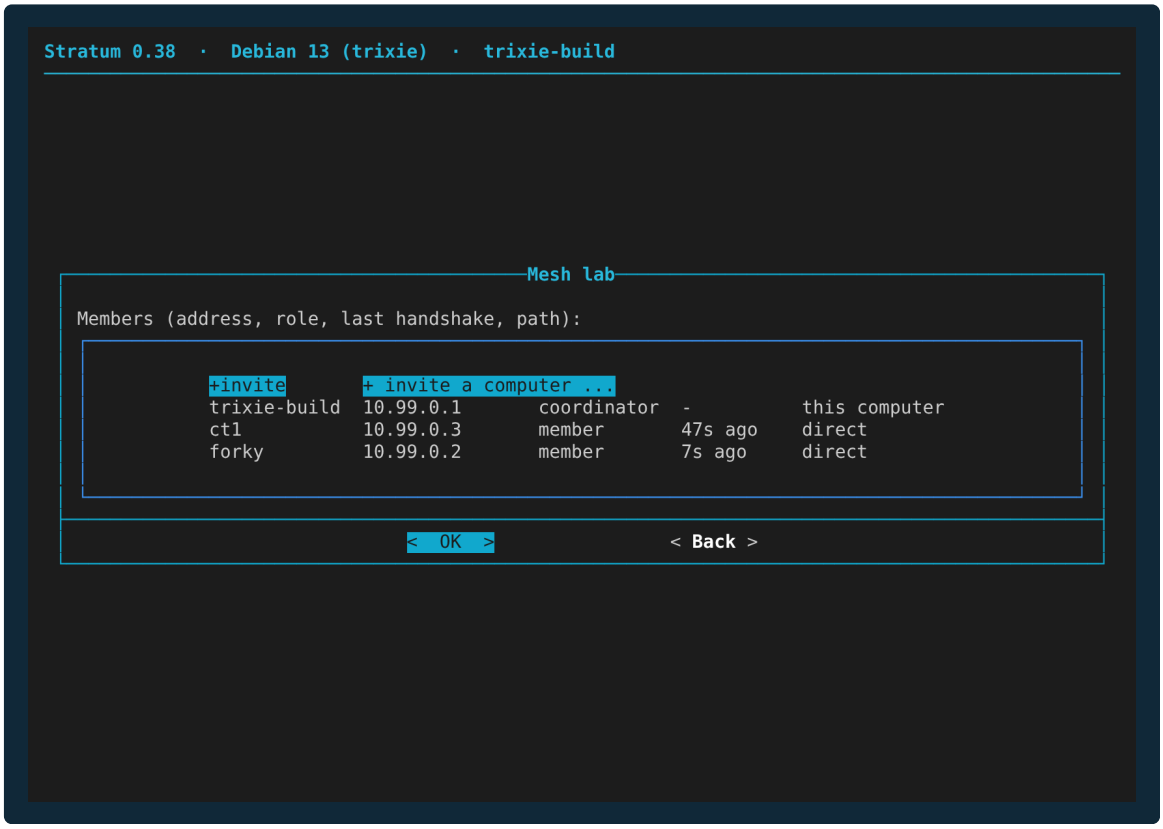


Figure 34 Members, last handshake and path (direct or relay).

On a computer that is not in a mesh yet, the menu asks whether to create one or to join one with a code.

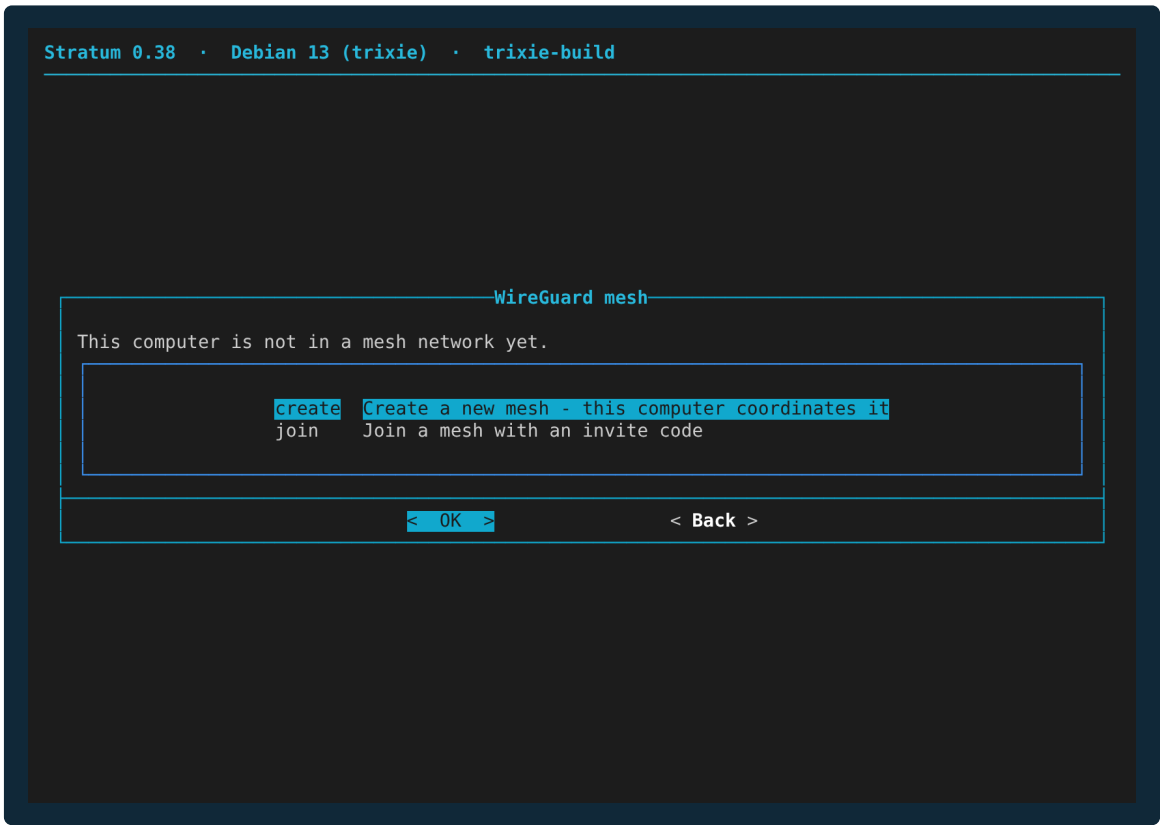


Figure 35 Create a mesh, or join one.

```
# stratum wireguard members          # on forky
NAME            ADDRESS        ROLE         ENDPOINT              HANDSHAKE  PATH
trixie-build    10.99.0.1      coordinator  10.0.0.21:51820       21s ago    direct
ct1             10.99.0.3      member       10.0.3.11:51820       74s ago    direct
forky           10.99.0.2      member       10.0.0.22:51820       -          this computer
```

15.4 Networks behind a member

A member can make networks behind it reachable from the whole mesh — for example a network of virtual machines on that computer. Write them, one per line, to a file `/etc/stratum/wireguard/routes.d/<name>.routes`:

```
# /etc/stratum/wireguard/routes.d/lab.routes
10.10.20.0/24
```

Within a minute the coordinator passes the list on, and every other member routes these networks through the mesh; the member behind which they are forwards the traffic (Stratum switches on IP forwarding there by itself). A network that overlaps one announced before is refused — the first one wins.

15.5 Firewall and names

With stratum-firewall, the UDP port is opened, everything that comes from the mesh is allowed (setting `trust`), and the coordinator may forward between members (`relay`). Every member's name is written to `/etc/hosts` as `<name>` and `<name>.<mesh>`. The file server and the database count the mesh as a local network.

> **REMEMBER**
>
> `trust yes` means: a computer in the mesh is as trusted as one in your office. With `trust no` only what your firewall rules allow comes in from the mesh — ping and SSH, for example.

15.6 Settings and commands

Setting	Meaning
`endpoint auto|host[:port]`	Where the others reach this computer.
`port`	UDP port (51820).
`trust yes|no`	Allow everything from the mesh (default yes).
`relay yes|no`	Coordinator: relay between members (default yes).
`hosts yes|no`	Member names in `/etc/hosts`.
`keepalive` · `mtu`	25 seconds (keeps NAT open) · 1420.

Command	What it does
`create <mesh> [--subnet --port --endpoint --name]`	A new mesh; this computer coordinates it.
`invite <name> [address]` · `invite-show <name>`	Invite a computer · show the code again.
`join <code> [--endpoint e]`	Join a mesh.
`members` · `ping <name>`	Members with handshake and path · ping through the mesh.
`remove <name>`	Coordinator: remove a member (the others drop it within a minute).

© 2026 Stratum Project

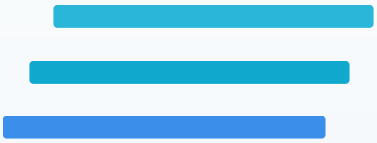
Command	What it does
`leave --yes`	Leave the mesh; on the coordinator: delete the mesh.
`sync` · `show`	Fetch the member list now · `wg show stratum-wg`.

Files: `/etc/stratum/wireguard.conf`, `/etc/stratum/wireguard/` (keys, members; root only), `/etc/stratum/wireguard/routes.d/` (networks behind this computer), `/var/lib/stratum-wireguard/mesh.tsv`, `routes.tsv`. Services: `stratum-wireguard` (the interface `stratum-wg`), `stratum-wireguard-sync.timer`, `stratum-wireguard-api` (coordinator: the member list on port 51821, only inside the mesh).

PART III

License and support

How a computer gets its Stratum license, how you ask for remote support — and, step by step, how the Stratum support team gives it.



CHAPTER 16

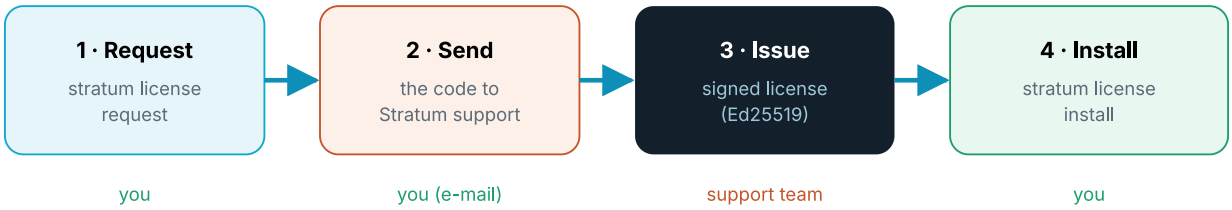
Your Stratum license

END USER ADMINISTRATOR

Stratum's modules are free software. A **license** ties a computer to a support contract: it unlocks remote support and features such as the **discrete** banner. It is checked offline — the computer does not need to "phone home".

A feature can be sold as a **subscription** — monthly or yearly — with its own end date, separate from the rest of the licence. When a subscription ends, that feature stops (after the grace period) while the other features keep working; `stratum license status` lists each subscription.

16.1 The license in four steps



16.2 Step by step

1. **Create the request.** In the menu: **license** → **request**. Or:

```
# stratum license request
-----BEGIN STRATUM LICENSE REQUEST-----
SE9TVD10ZXN0YmVkCkhXSUQ9U1RSLUlCSzYtS1VGRy1ETFpZLTVNVFAKUkVM…
-----END STRATUM LICENSE REQUEST-----
```

2. **Send it** — the whole block, including the BEGIN and END lines — to Stratum support, together with your name or company.

3. **Wait for the license.** You receive a block that starts with `-----BEGIN STRATUM LICENSE-----`, or a file.

4. **Install it.** In the menu: **license** → **enter** → **paste**. Or:

```
# stratum license install license.txt
  ✓ license SL-2026-0002 installed: Example Ltd, level standard, valid until 2027-10-31 (372 days)
  ✓ features: support ansible-fleet
```

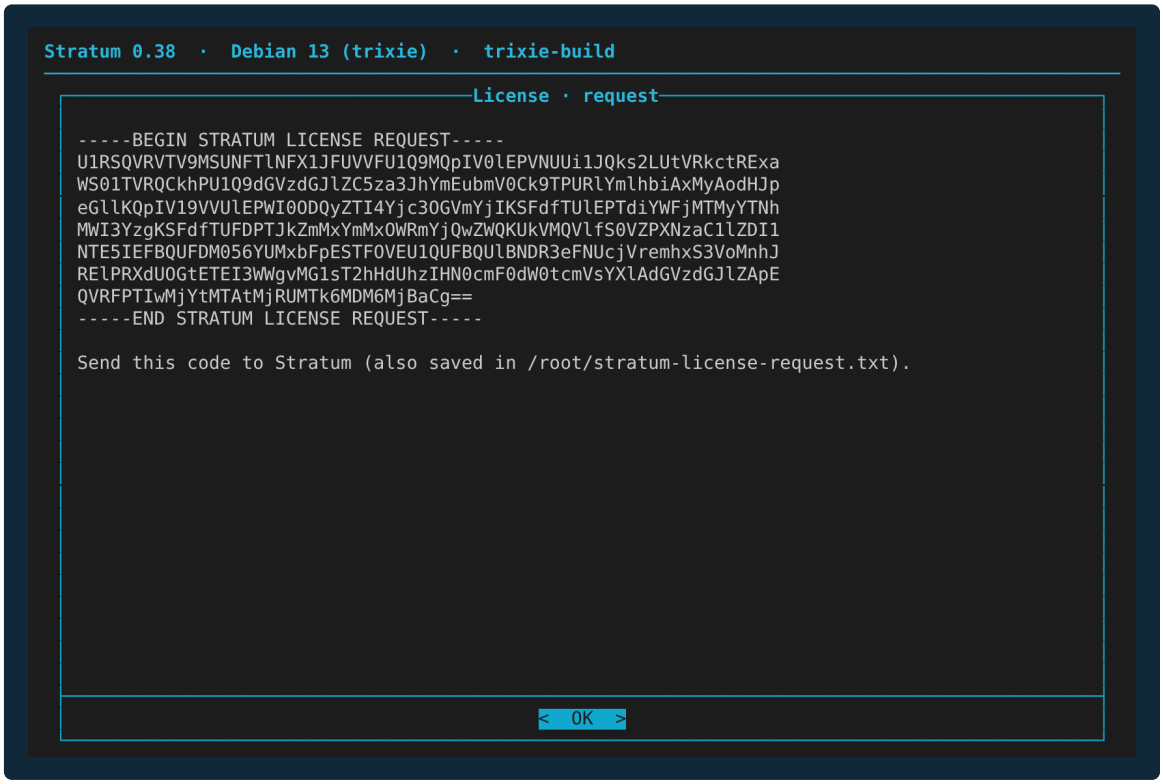


Figure 36 The license request in the menu.

> **REMEMBER**
>
> The request contains no secrets: only the hardware ID, the host name and a public key for remote support. It is safe to send by e-mail.

16.3 Checking your license

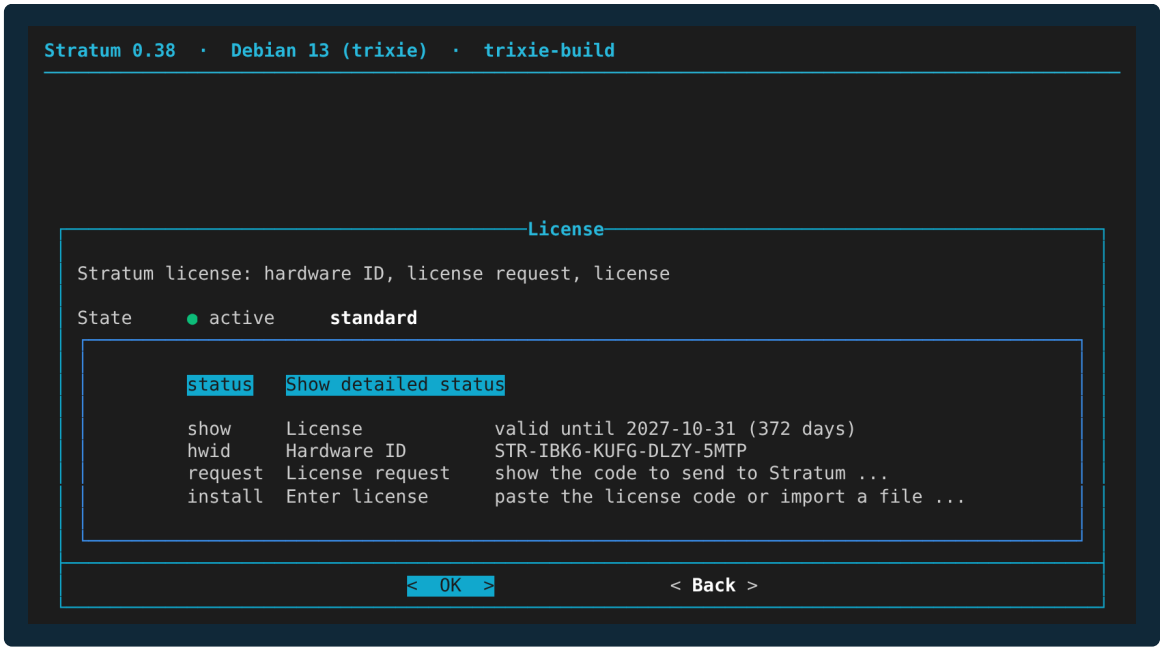


Figure 37 License state in the menu.

© 2026 Stratum Project

```
# stratum license status
Hardware ID:   STR-IBK6-KUFG-DLZY-5MTP
License:       valid until 2027-10-31 (372 days)
ID:            SL-2026-0002
Customer:      Example Ltd
Level:         standard
Features:      support ansible-fleet
Valid:         2026-10-24 - 2027-10-31 (+14 days grace)
```

State	Meaning and what to do
valid	All is well.
grace	Expired, but features keep working for the grace period (usually 14 days). Ask for a renewal now.
expired	Features that need a license stop. Ask for a renewal.
notyet	Valid from a later date.
wronghw	Issued for another computer — for example after a hardware change. Send a new request.
invalid	The signature does not match: the license was altered or damaged. Install it again from the original.
none	No license installed.

16.4 The hardware ID

```
# stratum license hwid
Hardware ID:  STR-IBK6-KUFG-DLZY-5MTP
  DMI UUID     b4842e28b778efb2
  machine ID   7baac132a3a1b7c8
  MAC          2dfc1bc19dfb40ed
```

The hardware ID is made from three fingerprints of the computer. The license stays valid as long as **two of the three** still match — so replacing a network card or reinstalling the system does not cost you the license, but moving it to another computer does not work.

16.5 Time and the license

Setting the computer's clock back does not make an expired license valid again. The license uses the **latest** of three times:

1. **The system clock.**
2. **The date of the APT repository metadata.** Every `apt update` downloads files signed by Debian and by Stratum, with the date they were published. Nobody can move that date forward on their own computer.
3. **The latest time seen before.** It is kept in `/var/lib/stratum/license/clock` and updated on every check and every day by `stratum-license-clock.timer`. If the file is deleted, it is rebuilt from the package database and the logs.

```
# stratum license clock
System clock:   2026-10-05 12:00:00
Trusted time:   2026-10-25 20:14:35  (the system clock is behind)
APT metadata:   2026-10-25 20:14:24
Last seen:      2026-10-25 20:14:35
```

When the system clock is more than a day behind, `stratum license status` warns about it. A clock that is only a little wrong (a VM, an empty clock battery) does no harm.

> ⚙ **TECHNICAL STUFF**
>
> Remote support does not depend on the computer's clock at all: the relay checks the end of the license (and its grace period) with its own clock.

16.6 All commands

Command	What it does
`stratum license status`	Hardware ID, state, level, features, validity.
`stratum license hwid`	The hardware ID and its parts.
`stratum license request`	The request code to send to Stratum.
`stratum license install [file|-]`	Install from a file or pasted code (`-` = stdin).
`stratum license remove`	Remove (kept as `license.removed`).
`stratum license check [feature]`	Exit code 0 when the license allows the feature (also in the grace period). A feature sold as a subscription must also be within its own end date.
`stratum license info`	The license as `KEY=value`, for scripts.
`stratum license clock`	The time the license uses: system clock, trusted time, APT metadata, last seen.

```
# stratum license check support && echo "remote support available"
```

> ⚙ **TECHNICAL STUFF**
>
> The license is a short text signed with Stratum's Ed25519 key and checked with Stratum's public key in `/usr/lib/stratum/license/signing.pub`. The private key never leaves the Stratum build host. Files: `/etc/stratum/license/license`, `/etc/stratum/license/relay_ed25519` (the key for remote support).

© 2026 Stratum Project

CHAPTER 17

Asking for remote support

END USER

Something is wrong and you would like an expert to look? With Stratum you can let the support team into your computer for a limited time — without opening anything in your firewall, without sharing a password, and with every keystroke recorded.

17.1 Before you start

- A license with the feature **support** (`stratum license status` lists it).
- The package `stratum-support` (menu → **modules**).
- Outgoing internet access to the Stratum relay (SSH, port 22).

17.2 Open a session

1. Call or write to Stratum support and describe the problem. Agree on how long they need.
2. In the menu: **support** → **start**, and choose the duration (1–72 hours, default 4). Or:

```
# stratum support start 2
  ✓ remote support session 20261024-203717 is open until 2026-10-24 22:37

   Session code:  35161

   Give this code to Stratum support. They connect with:
   ssh -J stratum-relay@www.skraba.net -p 35161 stratum-support@localhost

   Support has root rights (sudo) during the session; everything typed is
   recorded in /var/log/stratum-support. Follow it live: stratum support watch
   End it now: stratum support stop
```

3. Tell the support team the **session code**. That is all they need.

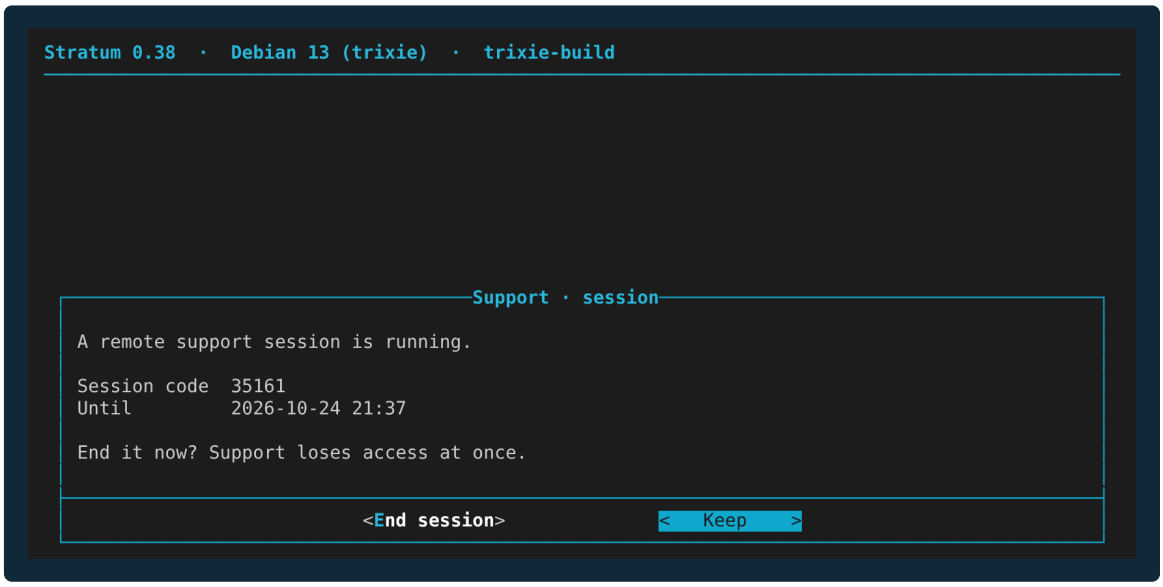


Figure 38 A running support session.

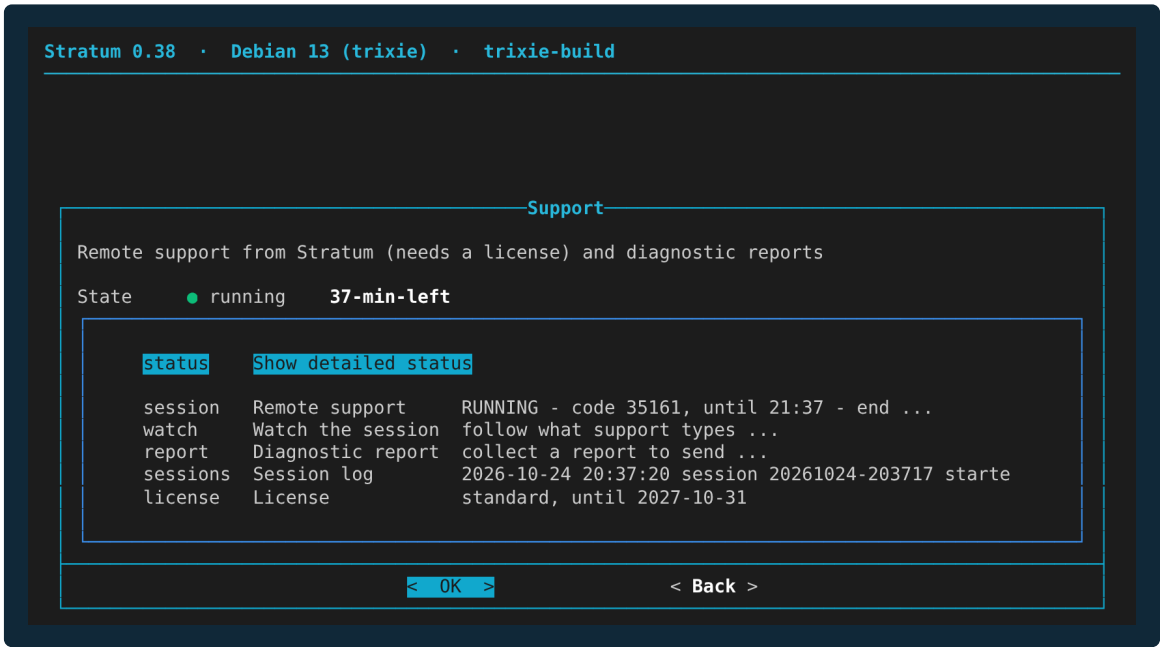


Figure 39 The support module in the menu.

17.3 While support is working

You can watch everything the support team types and sees, live:

```
# stratum support watch
```

The recording is kept in `/var/log/stratum-support/` after the session too.

17.4 End the session

The session ends by itself at the chosen time. To end it earlier — for example when the problem is solved, or whenever you change your mind:

```
# stratum support stop
```

The tunnel, the temporary user, its keys and its sudo rule are removed at once.

> **REMEMBER**
> You stay in control. Support can only get in while *you* have a session open, only through the tunnel your computer opened, and only until the session ends.

17.5 Sending a diagnostic report

Often a report is enough and no session is needed:

```
# stratum support report --anonymize
  ✓ report: /root/stratum-report-anon-20261024-204112.tar.gz (88K), host name, IP and MAC addresses replaced
```

The report contains Stratum and module status, installed packages, resources, network, failed services and errors from the journal. `--anonymize` replaces the host name and the IP and MAC addresses. Send the file to Stratum support.

© 2026 Stratum Project

17.6 What exactly happens — for the curious



Your firewall opens nothing: the connection is started by your computer.

- Your computer opens an SSH *reverse tunnel* to the Stratum relay. The relay and its host key come from your signed license, so it cannot be redirected elsewhere. The relay port is the session code.
- A temporary user `stratum-support` is created with the support team's keys from your license. The keys work only through the tunnel (from `127.0.0.1`), only until the session ends, and every terminal is recorded.
- When the time is over or you run `stop`, everything is removed.

Files: `/var/log/stratum-support/` (sessions.log, recordings), `/run/stratum/support/` (the running session). `stratum support sessions [n]` lists past sessions.

CHAPTER 18

Giving remote support — for the support team

SUPPORT TEAM

This chapter is the support team's handbook: how to issue licenses, keep the relay in order, connect to a customer's session, work responsibly and close the case.

18.1 Where things live

What	Where
License issuing	Build host `stratum`, tool `stratum-license-issue`, configuration `/etc/stratum/license-issuer.conf`.
Signing key	`/root/.stratum-license/signing.pem` on the build host — never copied anywhere.
Support key	`/root/.stratum-license/support_ed25519` — copy it to the workstation you give support from.
Registry	The list of issued licenses (CSV), kept by `stratum-license-issue`.
Relay	`www.skraba.net` (host lynx), user `stratum-relay`; keys in `/srv/stratum-relay/authorized_keys`.

> **WARNING**
> The signing key is the root of trust for every Stratum license. Whoever has it can issue licenses for any computer. Keep it on the build host, back it up offline, and never paste it anywhere.

18.2 Issuing a license

1. Save the customer's request block to a file, e.g. `req-acme.txt`.
2. Issue the license on the build host:

```
root@stratum:~# stratum-license-issue new req-acme.txt --customer "ACME d.o.o." \
      --until 2027-10-31 --level standard --features support
  ✓ license SL-2026-0003 for ACME d.o.o. (standard, support) until 2027-10-31
-----BEGIN STRATUM LICENSE-----
…
-----END STRATUM LICENSE-----
```

3. Push the relay keys, so the new customer's tunnel is accepted:

```
root@stratum:~# stratum-license-issue push
```

4. Send the license block to the customer, with the instruction `stratum license install`.

18.2.1 Products and subscriptions

A licence can cover several **products** (licence types), each mapping to a feature — `support` (remote support) and `trade-term` (Trading terminal). List them with `stratum-license-issue products` and issue with `--product <p[,q]>` instead of `--features`.

A product or feature can be a **subscription** with its own end date, independent of the rest of the licence. `--subscribe <feature>:monthly|yearly` sets the end date one month or year after `--from`; `--feature-until <feature>:<date>` sets an explicit date, and `--term` only labels it. `stratum license check <feature>` then fails once that subscription has ended (with the grace period), even while other features stay valid, and `stratum license status` lists each subscription.

```
root@stratum:~# stratum-license-issue new req-acme.txt --customer "ACME d.o.o." \
      --until 2027-12-31 --product support,trade-term \
      --subscribe support:yearly --subscribe trade-term:monthly
  ✓ license SL-2026-0007 for ACME d.o.o. (standard, support,trade-term) until 2027-12-31 [subs:
support:2027-10-31;trade-term:2026-11-30]
```

stratum-license-issue ...	What it does
`init`	Create the signing key, support key and configuration (once).
`pubkey`	Stratum's public license key (shipped in stratum-license).
`products`	Known licence types (products) and the feature each maps to.
`new <request|-> --customer <name> --until <date>`	Issue. Options: `--level`, `--product p,q`, `--features f,g` (default `support`), `--subscribe f:monthly|yearly`, `--feature-until f:<date>`, `--term f:monthly|yearly`, `--from`, `--grace <days>`, `--id`.
`list` · `show <ID>`	Issued licenses; one license and its code again.
`revoke <ID>`	Revoke; then `push` — remote support stops at once.
`push`	Write the relay keys to the relay host.
`relay-hostkey`	Fetch and store the relay's SSH host key (it goes into every license).

> **TIP**
>
> Renewal = a new license with a later `--until` from the same request. Keep each customer's request on file; then a renewal never needs the customer's help.

18.3 Preparing your workstation (once)

Copy the support key to `~/.ssh/stratum_support_ed25519` (mode 600) and add two entries to `~/.ssh/config`:

```
Host stratum-relay
    HostName www.skraba.net
    User stratum-relay
    IdentityFile ~/.ssh/stratum_support_ed25519
    IdentitiesOnly yes
Host stratum-session
    HostName localhost
    User stratum-support
    IdentityFile ~/.ssh/stratum_support_ed25519
    IdentitiesOnly yes
    ProxyJump stratum-relay
    NoHostAuthenticationForLocalhost yes
```


© 2026 Stratum Project

> ⚙ **TECHNICAL STUFF**
>
> The customer's screen suggests `ssh -J …`. That works only when your SSH client offers the support key to the relay by itself (an agent, or a `Host www.skraba.net` block). Why a config file instead of `ssh -J` with `-i`? With `-J`, the `-i` option does not apply to the jump host, so the relay would not see the support key. The `Host` blocks give each hop its own key. On the relay, the support key may only forward to `localhost` ports — it never gets a shell there.

18.4 Connecting to a session

The customer tells you the session code — for example **35161**:

```
you@workstation:~$ ssh -p 35161 stratum-session

  Stratum remote support - session 20261024-203717, ends 2026-10-24 22:37
  This terminal is recorded. Root: sudo -i

stratum-support@trixie-build:~$ sudo stratum status
```

Problem	Cause and fix
`Connection refused` on the session port	No session with that code: wrong code, the session ended, or the customer's tunnel did not come up (ask for `stratum support status`).
`Permission denied` at the relay	Support key not on the relay: run `stratum-license-issue push`.
`Permission denied` at the customer	The license has another support key, or the session expired.
Customer: tunnel fails	Outgoing SSH blocked at the customer, or license without the current relay host key (issue a new license after `relay-hostkey`).

18.5 Conduct during a session

- Say who you are and what you are going to do before you change anything.
- Remember that every keystroke is recorded and the customer may be watching.
- Prefer `status`, logs and `--check` runs; change only what was agreed.
- Use Stratum commands where possible — they protect against lock-outs and are rolled back if the connection drops.
- Never copy customer data off the computer without explicit permission.
- When done, tell the customer and ask them to run `stratum support stop` (or let it expire).

18.6 Analysing a report

```
you@workstation:~$ tar -xzf stratum-report-anon-20261024-204112.tar.gz
you@workstation:~$ ls stratum-report-*/
```

Start with the Stratum status and the failed services, then the journal errors. An anonymized report contains placeholders instead of host names and addresses; ask the customer if you need the real ones.

18.7 Revoking a license

```
root@stratum:~# stratum-license-issue revoke SL-2026-0001
root@stratum:~# stratum-license-issue push
```

The customer's relay key is removed from the relay, so no new support session can be opened. Offline features (such as the discrete banner) keep working until the license expires — revoking is about access, not about the computer.

18.8 Checklists

New customer

1. Request received and saved.
2. `new … --features support`
3. `push`
4. License sent, installed (`status` shows **valid**).
5. Test session agreed.

Support case

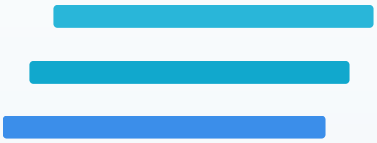
1. Problem and duration agreed.
2. Customer runs `support start`, gives the code.
3. `ssh -p <code> stratum-session`
4. Work, explain, write down what changed.
5. Session ended; short summary to the customer.

© 2026 Stratum Project

PART IV

Reference

The project and its repository, the complete command reference, files and directories, troubleshooting, a glossary and a quick reference card.



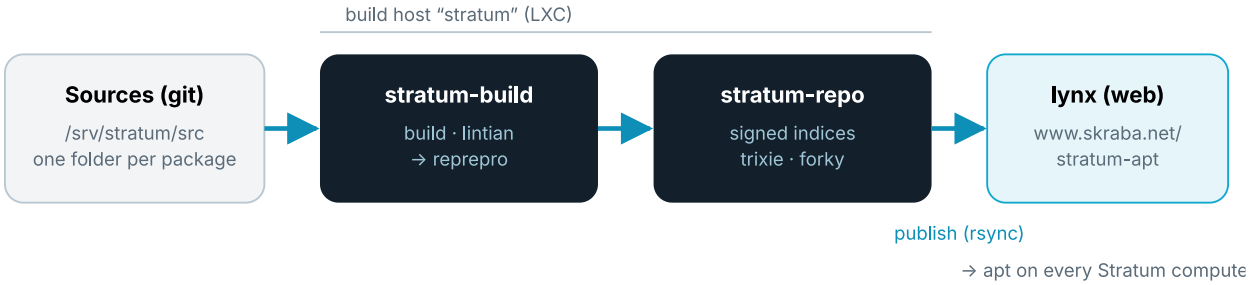
CHAPTER 19

The project and its repository

ADMINISTRATOR SUPPORT TEAM

Where Stratum comes from: how a change becomes a package, how packages are signed, and how they reach your computer.

19.1 From source to your computer



Part	Role
Build host `stratum`	An LXC container. Holds the sources (git), builds packages, keeps the repository (reprepro) and the signing keys — the repository key and the license key.
Test machines	Virtual machines `trixie-build` (Debian 13) and `forky-build` (Debian 14), where every package is installed and tested before it is published.
lynx	The public web server. Receives the finished repository by rsync (user `stratum-sync`, restricted to `/var/www/stratum-apt`) and runs the support relay.
Project page	`https://www.skraba.net/stratum/start` — installation, modules, news.

19.2 Building and publishing SUPPORT TEAM

```
root@stratum:~# stratum-build stratum-banner
==> Done: stratum-banner 0.10  (artifacts in /srv/stratum/build/stratum-banner/0.10)
root@stratum:~# stratum-repo list | grep banner
trixie|main|amd64: stratum-banner 0.10
forky|main|amd64: stratum-banner 0.10
root@stratum:~# stratum-repo publish
Published to https://www.skraba.net/stratum-apt/
```

Command	What it does
`stratum-build <package> [--publish] [--no-lint]`	Build from `/srv/stratum/src/<package>`, check with lintian, add to the repository.
`stratum-repo add <deb|changes|dsc>…`	Add to trixie — to `main`, `contrib` or `non-free`, by the package's section — and copy to forky.
`stratum-repo remove <package>…`	Remove from all distributions.
`stratum-repo list` · `check`	List packages; verify database and pool.

Command	What it does
`stratum-repo sync` · `export`	Copy trixie to the others; regenerate and re-sign indices.
`stratum-repo publish`	Mirror the repository to lynx.
`stratum-repo pubkey` · `sources`	The public signing key; the APT `.sources` entry for clients.

19.3 Versions and rules of the project

- Package versions are `0.x` and keep counting up (0.9, 0.10, 0.11 …); there is no jump to 1.0.
- Every option is documented in the module's help, in `stratum help` and in the manual page.
- Modules that change APT sources or upgrade packages (software) work only on an official, supported Debian release.
- Stratum commands ignore `Ctrl`+`C`, so a change is never left half done.

19.4 Packages in the repository

Package	Version	Contents
`stratum-archive-keyring`	2026.10.29	The repository signing key and the APT source.
`stratum-core`	0.49	The `stratum` command, the menu and the module framework, written in Go.
`stratum-software`	0.24	APT repositories and automatic upgrades.
`stratum-memory`	0.28	Compressed swap, deduplication and ballooning.
`stratum-firewall`	0.09	Firewall, also for the rules of other modules.
`stratum-security`	0.10	fail2ban and hardening.
`stratum-fileserver`	0.03	Samba and NFS shares.
`stratum-database`	0.04	MariaDB server.
`stratum-wireguard`	0.08	WireGuard mesh network.
`stratum-ansible`	0.06	Ansible roles.
`stratum-license`	0.07	License.
`stratum-support`	0.06	Remote support and reports.
`stratum-handbook`	0.25	This handbook as PDF: `/usr/share/stratum/handbook/stratum-handbook.pdf`, with a link in `/root`.
`stratum-watchdog`	0.08	Checks and self-healing.

Versions as of this edition; `stratum modules` shows the current ones. The packages above are in the component `main`; `contrib` and `non-free` hold further software of the Stratum project, which is described separately.

19.5 The signing key

The repository is signed; APT refuses packages whose signature does not match. Check the key's checksum when you install it by hand:



© 2026 Stratum Project

```
# sha256sum /usr/share/keyrings/stratum-archive-keyring.gpg
4e9b1cb45f58d955340ac93b6c8d997d900d9780d005f002ce978bab39d53647  …
```

CHAPTER 20

Command reference

END USER ADMINISTRATOR SUPPORT TEAM

Every command on one page or two. The module chapters explain them in detail.

20.1 The stratum command

Command	What it does
`stratum`	The menu (`--no-splash` without the start-up animation).
`stratum status`	Is the Stratum repository configured, reachable and correctly signed? Exit code 1 on a problem.
`stratum modules`	Installed and available modules.
`stratum about`	Version, modules, project page.
`stratum init`	Run the initialisation steps (`stratum init --list` shows them).
`stratum help [command|module]`	Help.
`stratum -V`	Version.

Environment variable	Effect
`NO_COLOR=1`	No colours.
`STRATUM_NO_SPLASH=1`	No start-up animation.
`STRATUM_INTERRUPT=1`	Allow `Ctrl`+`C`.
`STRATUM_FORCE=1`	Apply a change despite a warning (exit code 3).
`STRATUM_NO_UPDATE=1`	software: no `apt-get update` after a change.
`STRATUM_FIREWALL_NOCONFIRM=1`	firewall: no confirmation and rollback.
`STRATUM_SECURITY_NOCONFIRM=1`	security: no confirmation and rollback.
`DIALOGRC`	Own dialog colours instead of the Stratum theme.

20.2 Modules

Module	Actions
`software`	status · list · enable [repo…] · disable [repo…] · sources <repo> on|off · components [c…] · upgrades [complete|security|turn-off] · refresh · set repositories|components|upgrades …
`banner`	configure standard|minimal|discrete · preview
`memory`	status · start · stop · enable · disable · restart · settings · setting type|memory|swappiness|compression|diskswap|dedup|ballooning|threshold|recompress · dedup-config · dedup-unmerge · recompress-now · refresh · show [type]

Module	Actions
`firewall`	status · start · stop · enable · disable · restart · confirm · rollback · settings · setting … · rules · rule-add · rule-remove · events · interfaces · listening · docker-mode · docker-apply · port-allowed
`security`	status · start · stop · enable · disable · restart · settings · setting … · toggles/set jails|hardening · whitelist · banned · ban · unban · events · confirm · rollback
`fileserver`	status · start · stop · enable · disable · restart · apply · shares · share-add · share-remove · share <name> [show|set …] · users · user-add · user-enable · user-passwd · user-remove · groups · group-add · group-remove · group-member · connections · settings · setting …
`database`	status · start · stop · enable · disable · restart · dbs · db-create · db-drop · users · user-create · user-passwd · user-drop · grant · revoke · grants · backup · backups · restore · secure · processes · settings · setting …
`wireguard`	status · start · stop · enable · disable · restart · create · invite · invite-show · join · members · remove · ping · leave · sync · show · settings · setting …
`ansible`	toggles/set roles · managed … · pull … · controller key · hosts · ping · playbooks · playbook-add · run · adhoc · galaxy · vault · runs
`license`	status · hwid · request · install · remove · check · info · clock
`support`	status · start [hours] · stop · watch · sessions · report [--anonymize]
`watchdog`	status · start · stop · enable · disable · restart · run [check…] · events [hours] · pause [minutes] · resume · settings · setting … · toggles/set checks

20.3 Exit codes

`0` done · `1` error · `3` warning, not applied (repeat with `STRATUM_FORCE=1`).

© 2026 Stratum Project

CHAPTER 21

Files and directories

ADMINISTRATOR SUPPORT TEAM

Path	Contents
`/usr/lib/stratum/`	Stratum itself: libraries, commands, modules (do not edit).
`/usr/lib/stratum/modules/<name>/`	One module.
`/usr/lib/stratum/supported`	Supported Debian releases.
`/etc/stratum/`	All Stratum configuration: `<module>.conf`.
`/etc/stratum/software.d/`, `memory.d/`	Your own repository definitions and types of computer for memory.
`/etc/stratum/firewall.rules`	Firewall rules (tab separated).
`/etc/stratum/license/`	License and the relay key.
`/var/lib/stratum/license/clock`	The latest time seen (license).
`/etc/apt/sources.list.d/stratum*.sources`	APT sources written by software.
`/etc/apt/apt.conf.d/52stratum-software`	Automatic upgrades.
`/etc/systemd/zram-generator.conf.d/90-stratum-memory.conf`	zram device.
`/etc/sysctl.d/90-stratum-*.conf`	Kernel settings (memory, security).
`/etc/fail2ban/jail.d/90-stratum.local`	fail2ban.
`/etc/ssh/sshd_config.d/10-stratum.conf`	SSH hardening.
`/srv/stratum/ansible/`	Ansible controller.
`/etc/stratum/firewall.d/<module>.rules`	Firewall rules of other modules (written by them).
`/etc/stratum/fileserver.d/<share>.share`	Shares.
`/etc/samba/stratum-*.conf`, `/etc/exports.d/stratum.exports`	Samba and NFS configuration (generated).
`/etc/mysql/mariadb.conf.d/90-stratum.cnf`	MariaDB settings (generated).
`/etc/stratum/wireguard/`	WireGuard keys and members (root only).
`/var/backups/stratum-database/`	Database backups.
`/var/log/stratum-support/`	Support sessions and recordings.
`/var/log/stratum-ansible/`	Ansible runs.
`/etc/stratum/watchdog.d/`	Own watchdog checks.
`/var/log/stratum-watchdog/events.log`	Watchdog events.

Path	Contents
`/etc/systemd/system/<unit>.d/90-stratum-watchdog.conf`	Restart=on-failure drop-ins of the watchdog.
`/etc/issue.d/stratum.issue`, `/etc/update-motd.d/20-stratum`	Banner.

TIP

This handbook is on every computer with `stratum-handbook` installed: `/root/stratum-handbook.pdf` (a link to `/usr/share/stratum/handbook/stratum-handbook.pdf`).

TIP

To back up a Stratum computer's configuration, `/etc/stratum/` is the one folder that matters. Add `/srv/stratum/` if you use the Ansible controller.

© 2026 Stratum Project

CHAPTER 22

Troubleshooting and FAQ

END USER ADMINISTRATOR SUPPORT TEAM

22.0.1 "stratum status" says the repository is not working.

Run `apt-get update` and read the error. *NO_PUBKEY* or *signature*: install the signing key again (chapter *Installing Stratum*). *Could not resolve*: check DNS and the network.

22.0.2 The menu says "unsupported configuration".

The system is not Debian 13 or 14, or not an official Debian. Most modules still work; software refuses to change anything.

22.0.3 I locked myself out with the firewall.

Wait 60 seconds: an unconfirmed change rolls back by itself. If it was confirmed, log in on the console and run `stratum firewall stop`, then fix the rules.

22.0.4 fail2ban banned my own address.

From another address or the console: `stratum security unban <address>` and `stratum security whitelist add <address>`.

22.0.5 My Docker container is running but I cannot reach it.

`stratum firewall listening` — if it says *BLOCKED (Docker)*, add a rule: `stratum firewall rule-add allow tcp:<port>`.

22.0.6 zram does not start.

Is `zram-tools` installed? `apt purge zram-tools`, then `stratum memory enable`.

22.0.7 The license shows "wronghw".

More than one of the three hardware fingerprints changed (new motherboard, cloned VM). Send a new request.

22.0.8 The license says "system clock is behind".

The computer's clock is more than a day earlier than the last time Stratum saw. Switch on time synchronisation (`timedatectl set-ntp true`); the license itself always uses the later time.

22.0.9 Support cannot connect.

`stratum support status` on your computer: is a session running, and what is the code? Is outgoing SSH (port 22) allowed in your network?

22.0.10 A service keeps failing and the watchdog says "needs attention".

Three restarts within an hour did not help, so the watchdog stopped trying. Read the service's log (`journalctl -u <service> -b`), fix the cause, then `stratum watchdog run`.

22.0.11 Ctrl+C does nothing.

On purpose: Stratum commands finish what they started. Set `STRATUM_INTERRUPT=1` if you really need to interrupt.

22.0.12 Where are the logs?

`stratum security events`, `stratum firewall events`, `stratum ansible runs`, `stratum support sessions` — and the full system journal with `journalctl`.

© 2026 Stratum Project

CHAPTER 23

Glossary

END USER ADMINISTRATOR SUPPORT TEAM

Term	Meaning
ACL	Access control list: permissions for several users and groups on a file or folder (file server).
APT	Debian's package manager (`apt`, `apt-get`).
Backports	Newer versions of selected packages, built for the stable release.
Container	An isolated environment that shares the kernel of the host: for example a Docker container.
Coordinator	The computer that creates a WireGuard mesh, invites members and keeps the member list.
deb822 / .sources	The modern format of APT source files.
Endpoint	The address and UDP port at which a WireGuard computer is reached.
fail2ban	Bans addresses that fail to log in too often.
Grace period	Days after a license expires during which it still works.
Hardening	Settings that make a system harder to attack.
Hardware ID	A fingerprint of the computer, `STR-XXXX-XXXX-XXXX-XXXX`.
Lock-out	Losing remote access to your own computer.
MariaDB	The database server in Debian, compatible with MySQL.
Mesh	A network in which every computer connects directly to every other one.
Module	One Stratum function, packaged as `stratum-<name>`.
MOTD	Message of the day, shown after login.
NAT	Network address translation: computers behind one address; only published ports are reachable from outside.
NFS	Network File System: file sharing between Linux and Unix computers.
Playbook	An Ansible file describing the desired state of computers.
Profile	A named set of choices, e.g. banner *minimal*.
Relay	The Stratum server that connects support to a customer's tunnel; in a WireGuard mesh, the coordinator forwarding traffic between members that cannot reach each other.
Reverse tunnel	A connection opened from the inside out, through which a connection can come back in.
Rollback	Returning to the state before a change.
Samba	File sharing with Windows, macOS and Linux (the SMB protocol).
Self-healing	Repairing a problem automatically, e.g. restarting a service that stopped (stratum watchdog).
Session code	The number support needs to connect to your session.

Term	Meaning
Share	A shared folder of the file server.
Swappiness	How readily the kernel moves memory to swap.
Type of computer	workstation, server, hypervisor, vm-guest (and small for memory) — sets defaults.
ufw	Uncomplicated Firewall, Debian's simple firewall tool.
Watchdog	A program that checks whether everything runs and restarts what has stopped.
WireGuard	A modern, fast and simple VPN protocol in the Linux kernel.
zram	A compressed block device in RAM, used as fast swap.

© 2026 Stratum Project

CHAPTER 24

Quick reference card

END USER ADMINISTRATOR SUPPORT TEAM

Cut out along the edge, or keep it next to the keyboard.

Every day

`stratum`	menu
`stratum status`	repository OK?
`stratum <m> status`	module state
`stratum <m> help`	all options
`stratum security events`	what happened

File server and database

`fileserver share-add n p both`	share a folder
`fileserver user-add u`	a user for the shares
`database db-create n`	new database
`database user-create u`	new database user
`database backup n`	back it up

Support

`license request`	ask for a license
`license install f`	install it
`support start 2`	session, 2 h
`support watch`	watch live
`support stop`	end now

Firewall

`rule-add allow web`	open 80, 443
`rule-add allow tcp:8080`	open a port
`rules` / `rule-remove n`	list / remove
`listening`	what is reachable
`confirm`	keep a change (60 s)

WireGuard mesh

`create office`	new mesh
`invite laptop`	invite a computer
`join SWG1:…`	join (on the new one)
`members`	who is connected
`ping n`	test a member

> **REMEMBER**
>
> Exit code **3** means “not applied, looks risky”. Read the message. Only then, if you are sure: `STRATUM_FORCE=1` .

STRATUM HANDBOOK

Your system remains Debian. It just does more.

Stratum is a functional layer for Debian: small, independent modules for software sources and automatic upgrades, memory management, a firewall that will not lock you out, protection against password guessing, file sharing, databases, a WireGuard mesh, Ansible automation — and remote support when you need a hand.

This book takes you through all of it, step by step:

- install Stratum and find your way around its menu in minutes;
- every module, every command and every switch, with examples to copy;
- get a license and ask for remote support — safely, with everything recorded;
- for the support team: issuing licenses, the relay, and how a session is run.

END USER · ADMINISTRATOR · SUPPORT TEAM



Stratum Project

Stratum 0.45 · First edition · November 2026

www.skraba.net/stratum