VIRTUAL MACHINES, CONTAINERS AND APPLICATIONS

WORK IN PROGRESS



VM Reactor

Handbook

The guide to running machines, containers and applications side by side on Debian – for the people who use them and for the administrators who run the host.

END USER ADMINISTRATOR

Stratum Project

www.skraba.net/vm-reactor

VM Reactor 0.24 · Debian 13 (trixie) · Debian 14 (forky) · Draft · November 2026

VM REACTOR

VM Reactor Handbook

Virtual machines, system containers and Docker applications on one Debian host – installation, the web interface, guests, storage, memory, networks and backups, explained step by step.

Stratum Project
VM Reactor 0.24 · Draft · November 2026

VM Reactor Handbook
Draft · November 2026 · describes VM Reactor 0.24, a technical preview, published in the component *contrib* of the Stratum repository.

This book is being written. The first chapters are complete; the chapters shown in grey in the contents are in preparation and will follow in the next drafts of this book, together with VM Reactor itself.

© 2026 Stratum Project. All rights reserved. VM Reactor is 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. Linux is a registered trademark of Linus Torvalds. QEMU is a trademark of Fabrice Bellard. Other names may be trademarks of their owners. VM Reactor is an independent project and is not affiliated with these organisations.

Screenshots were taken on the test system `forky-build` (Debian 14). Host names, addresses and guests shown in this book are examples.

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

Contents

● End user ● Administrator *grey: in preparation*

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 Words used in this book 10
1.5 How this book is organised 10

2 What is VM Reactor ● ● 11
2.1 One host, three kinds of guests 11
2.2 Every guest has a number 11
2.3 What VM Reactor does for you 12
2.4 The web interface and the command line 12
2.5 VM Reactor and Stratum 13
2.6 A technical preview 13

3 Installing VM Reactor ● 15
3.1 What you need 15
3.2 The Stratum repository 15
3.3 Installing the package 16
3.4 First sign-in 17
3.5 Your first guests 17
3.6 Updating and removing 17

4 The web interface ● ● 19
4.1 Signing in 19
4.2 The layout 19
4.3 The overview 20
4.4 The list of guests 21
4.5 A guest 22
4.6 Tasks 23
4.7 Light and dark 23

PART II · GUESTS

5 Machines ● ● ..... *in preparation*
5.1 What a machine is ..... *in preparation*
5.2 Installation images ..... *in preparation*
5.3 Creating a machine ..... *in preparation*
5.4 The first start and the console ..... *in preparation*
5.5 Hardware - processors, memory and the minimum memory ..... *in preparation*
5.6 Disks - adding, growing, thin volumes ..... *in preparation*
5.7 The CD drive, ISO images and the boot order ..... *in preparation*
5.8 The QEMU guest agent and the hardware watchdog ..... *in preparation*
5.9 Firmware - BIOS and UEFI ..... *in preparation*

6 Containers ● ● ..... *in preparation*
6.1 What a system container is ..... *in preparation*
6.2 Creating a container from an image ..... *in preparation*
6.3 Resources and the size of a container ..... *in preparation*
6.4 The shell of a container ..... *in preparation*
6.5 Privileged and unprivileged containers ..... *in preparation*

7 Applications ● ● ..... *in preparation*
7.1 Docker in VM Reactor ..... *in preparation*
7.2 Installing Docker ..... *in preparation*
7.3 An application from a template ..... *in preparation*
7.4 An application from a compose file or a docker run command ..... *in preparation*
7.5 Updates and rollbacks ..... *in preparation*
7.6 Importing existing compose projects ..... *in preparation*

8 Consoles, snapshots and clones ● ● ..... *in preparation*
8.1 The console in the browser ..... *in preparation*
8.2 Snapshots ..... *in preparation*
8.3 Clones and templates ..... *in preparation*

9 Starting, stopping and the watchdog ● ..... *in preparation*
9.1 Start with the host ..... *in preparation*
9.2 Shutdown order and timeouts ..... *in preparation*
9.3 The watchdog ..... *in preparation*
9.4 Events ..... *in preparation*

PART III · RESOURCES OF THE HOST

10 Storage ● ..... *in preparation*
10.1 Directory ..... *in preparation*
10.2 NFS share ..... *in preparation*
10.3 LVM thin pool ..... *in preparation*
10.4 ISO images and LXC templates ..... *in preparation*
10.5 Defaults for new guests ..... *in preparation*

11 Memory ● ..... *in preparation*
11.1 More guests in the same memory ..... *in preparation*
11.2 Compression - zram ..... *in preparation*
11.3 Deduplication - KSM ..... *in preparation*
11.4 The balloon of the machines ..... *in preparation*
11.5 Memory management on the overview ..... *in preparation*

12 Networks ● ..... *in preparation*
12.1 The NAT network ..... *in preparation*
12.2 The local network ..... *in preparation*
12.3 Bridges ..... *in preparation*
12.4 Published ports ..... *in preparation*
12.5 Firewall of a guest ..... *in preparation*
12.6 The WireGuard mesh ..... *in preparation*

13 Backups and restore ● ● ..... *in preparation*
13.1 Backing up a guest ..... *in preparation*
13.2 Schedules ..... *in preparation*
13.3 Retention rules ..... *in preparation*
13.4 Protected backups ..... *in preparation*
13.5 Restore ..... *in preparation*
13.6 import and export ..... *in preparation*

PART IV · ADMINISTRATION

14 Users and roles ● ..... *in preparation*
14.1 Roles ..... *in preparation*
14.2 Access to guests ..... *in preparation*
14.3 Two-factor sign-in ..... *in preparation*
14.4 The audit log ..... *in preparation*

15 Securing the web interface ● ..... *in preparation*
15.1 Certificates ..... *in preparation*
15.2 Allowed networks ..... *in preparation*
15.3 Failed sign-ins ..... *in preparation*

16 Stratum integrations ● ..... *in preparation*
16.1 Firewall ..... *in preparation*
16.2 Security ..... *in preparation*
16.3 WireGuard mesh ..... *in preparation*
16.4 Ansible ..... *in preparation*
16.5 Database ..... *in preparation*
16.6 Support ..... *in preparation*

17 Updates ● ..... *in preparation*
17.1 Updating VM Reactor ..... *in preparation*
17.2 What an update keeps ..... *in preparation*

PART V · REFERENCE

18 Command reference ● ..... *in preparation*

19 Files and directories ● ..... *in preparation*

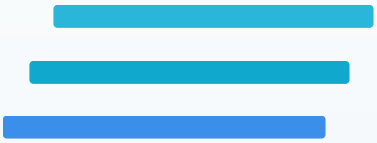
20 Troubleshooting and FAQ ● ● ..... *in preparation*

21 Glossary ● ● ..... *in preparation*

PART I

Getting started

What VM Reactor is, how this book works, how to install VM Reactor and how to find your way around its web interface.



CHAPTER 1

About this book

END USER ADMINISTRATOR

VM Reactor runs virtual machines, system containers and Docker applications side by side on one Debian computer – and lets you manage all of them in one web interface. This book shows you how, one step at a time. You do not need to be a virtualisation expert: if you can open a web browser and, now and then, a terminal, you can use VM Reactor.

1.1 Who this book is for

Every chapter and section carries one or two 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 works with guests every day: starts and stops them, opens their consoles, takes snapshots and backups.	Start a machine, open a shell in a container, restore yesterday's backup.
ADMINISTRATOR	The person responsible for the host: installs VM Reactor, sets up storage, networks, memory and users.	Install VM Reactor, add storage, plan backups, give colleagues access.

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

In VM Reactor every user has a **role** – viewer, user, operator, manager or administrator – and sees only what the role allows. The labels of this book are simpler: END USER covers everything up to the operator, ADMINISTRATOR the manager and the administrator. Chapter 14 describes the roles in detail.

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 stop a guest, delete data or cut the network.

TECHNICAL STUFF

Background for the curious. You can skip it and still use VM Reactor 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 `…`.
- Pages, tabs and buttons of the web interface look like this: **Overview**, **Create guest**, **Save**. A path through the interface is written with arrows: **Settings → About**.
- Placeholders are written in angle brackets: `vm-reactor start <id>` means, for example, `vm-reactor start 3`.

```
# vm-reactor list              # typed as root
ID      TYPE    NAME                    STATE      STORAGE
0       host    forky-build             running    -
1       lxc     ct-forky                running    local
2       kvm     vm-forky                stopped    local
…
```

- Sizes follow the web interface: **MiB** and **GiB** are binary units (1 GiB = 1024 MiB).

1.4 Words used in this book

A few words appear on almost every page. They are explained once more in the glossary at the end of the book.

Word	Meaning
Host	The Debian computer VM Reactor runs on. It always has the ID **0**.
Guest	Anything that runs on the host and has its own ID: a machine, a container or an application.
Machine	A virtual machine (KVM). It has its own kernel and can run any operating system.
Container	A system container (LXC): a complete Linux system that shares the kernel of the host.
Application	A Docker application: one or more application containers described by a compose file.
Storage	A place where VM Reactor keeps guests, ISO images, LXC templates and backups.

1.5 How this book is organised

Part I · Getting started

What VM Reactor is, installation and a tour of the web interface.

Part II · Guests

Machines, containers and applications; consoles, snapshots and clones; starting, stopping and the watchdog.

Part III · Resources of the host

Storage, memory, networks, backups and restore.

Part IV · Administration

Users and roles, securing the web interface, the Stratum modules and updates.

Part V · Reference

Every command and switch, the files and directories, troubleshooting and a glossary.

> **REMEMBER**
>
> This is a **draft** of the VM Reactor Handbook, written together with VM Reactor itself – a technical preview. The chapters shown in grey in the contents are in preparation. Each new version of VM Reactor comes with a newer draft of this book.

© 2026 Stratum Project

CHAPTER 2

What is VM Reactor

END USER ADMINISTRATOR

VM Reactor turns a Debian computer into a host for guests. Virtual machines, system containers and Docker applications live side by side, each with its own number, and one web interface shows all of them – what runs, what it uses, and what needs your attention.

2.1 One host, three kinds of guests

Most hosts need more than one kind of guest. A Windows desktop or a firewall appliance needs a **virtual machine** with its own kernel. A small Debian server for a web site runs perfectly in a **system container** that shares the kernel of the host and starts in a second. And more and more software is shipped as a **Docker application** – a compose file that describes one or more application containers.

VM Reactor manages all three, with the tools Debian already has:

Kind of guest	Technology	Typical use
Machine	KVM with QEMU and libvirt	Any operating system from an ISO image: Windows, BSD, other Linux distributions, appliances. Full isolation, own kernel.
Container	LXC	A complete Linux system (Debian, Ubuntu, Alpine and more) from the LXC image server. Light, fast, its own init and services.
Application	Docker and Docker Compose	Ready-made software: a web server, a database, a monitoring stack – from a template, a compose file or a `docker run` command.

You do not have to decide once and for all. One host can run a machine, three containers and five applications at the same time, and the web interface treats them the same way wherever it can: start, stop, console, snapshots, backups and the firewall work alike for all of them.

2.2 Every guest has a number

In VM Reactor every guest has an **ID** – a whole number. The host itself is always **ID 0**; the guests get **1, 2, 3 …**, whatever their type. The ID never changes during the life of a guest and appears everywhere: in the web interface, in the command line, in the names of files and backups.

```
# vm-reactor list
ID     TYPE    NAME                 STATE      STORAGE
0      host    forky-build          running    -
1      lxc     ct-forky             running    local
2      kvm     vm-forky             stopped    local
3      docker  web                  stopped    local
5      docker  whoami               running    local
```

Each guest has exactly **one configuration file**, `/etc/vm-reactor/guests/<id>.conf`, and **one directory** in a storage. Nothing is hidden in a database: you can read the configuration of a guest with any text editor, and a backup of the guest is simply one file.

```
# vm-reactor guest 2 (kvm) - written by vm-reactor
type: kvm
name: vm-forky
storage: local
cores: 1
```

```
memory: 512
memmin: 256
net: nat
ip: 10.10.20.3
firmware: bios
```

> ⚙ **TECHNICAL STUFF**
>
> VM Reactor does not replace libvirt, LXC or Docker – it drives them. A machine is a libvirt domain, a container is an LXC container, an application is a Docker Compose project. You can still look at them with `virsh`, `lxc-ls` or `docker compose ls`, but make your changes in VM Reactor: it writes the configuration of the guest and keeps the three worlds consistent.

2.3 What VM Reactor does for you

Guests

Create, start, shut down and delete machines, containers and applications; a console in the browser; snapshots and clones; start with the host in a defined order.

Storage

Directories, btrfs pools, LVM thin pools and NFS shares for guests, ISO images, LXC templates and backups – each storage with the same layout. A btrfs pool lives on one or more disks (single, RAID 0, RAID 1, RAID 1C3 or RAID 10) with snapshots, compression and checksums; the disks of btrfs and thin pools can be added, removed and replaced later while the guests keep running. A directory can be any mounted file system – on btrfs the guests get snapshots too, on an NFS share the host mounts it is used as it is.

Memory

Compression (zram), deduplication (KSM) and the balloon of the machines, set up for a hypervisor and switched on by default: more guests in the same RAM.

Networks

A NAT network with DHCP and DNS for the guests, the local network, bridges, published ports and a firewall for every guest.

Backups

One file per backup, schedules, retention rules and protected backups; restore under a new ID, export and import.

Protection

Users with roles, two-factor sign-in, an audit log, and a watchdog that notices and repairs what has failed.

2.4 The web interface and the command line

VM Reactor has two faces, and they always agree:

- The **web interface** runs on the host itself, over HTTPS, by default on port `8443`. It needs no extra web server and no database. Open it from any computer in your network with a current browser.
- The command `vm-reactor` does the same on the command line: list, start and stop guests, back them up and restore them, manage users, Docker and the memory setup. Every command and every switch is described in `vm-reactor help`.

```
# vm-reactor url
https://10.0.0.22:8443/
```


© 2026 Stratum Project

```
# vm-reactor start 2
# vm-reactor backup 2 --mode suspend
```

What you do in one place is visible in the other at once. Long-running work – creating a guest, a backup, a restore – runs as a **task**: you can follow it in **Tasks**, close the browser in the meantime, and find the complete log later.

2.5 VM Reactor and Stratum

VM Reactor belongs to the **Stratum** project – a functional layer for Debian made of small, independent modules. It is published in the component *contrib* of the Stratum repository, next to the Stratum modules in *main*.

VM Reactor builds on Stratum, but enjoys a certain independence: its own versions and release cycle, its own web interface and this handbook of its own. It needs `stratum-core`, uses `stratum-memory` for compression, deduplication and ballooning of memory, and works together with other Stratum modules when they are installed:

Stratum module	What VM Reactor does with it
`stratum-memory`	Compression (zram), deduplication (ksm) and ballooning, with a type tuned for a VM Reactor host (chapter 11).
`stratum-firewall`	Opens the port of the web interface and lets the guests through.
`stratum-security`	Blocks addresses that keep guessing passwords – also at the sign-in of VM Reactor.
`stratum-wireguard`	Announces the NAT network in the WireGuard mesh, so that other computers reach the guests.
`stratum-ansible`	Offers the guests as a dynamic inventory to Ansible.
`stratum-watchdog`	Watches VM Reactor itself.

> **REMEMBER**
>
> Your host remains Debian. VM Reactor installs with APT, is updated with APT and can be removed with APT – the guests stay where they are.

2.6 A technical preview

VM Reactor is new and still changing: features, settings and the web interface may change from one version to the next. Try it, test it and tell us what you think – but keep backups of your guests and do not rely on VM Reactor alone for production systems yet.

CHAPTER 3

Installing VM Reactor

ADMINISTRATOR

VM Reactor is one Debian package. If the Stratum repository is already set up on your computer, installing it is a single command; the web interface is ready a minute later.

3.1 What you need

Item	Requirement
System	**Debian 13 (trixie)** or **Debian 14 (forky, currently testing)**, architecture **amd64** or **arm64**
Processor	Hardware virtualisation (Intel VT-x or AMD-V; on arm64 the virtualisation extensions) for machines. Containers and applications run without it.
Memory	At least 4 GiB; plan the memory of all guests on top of about 1 GiB for the host. Compression and deduplication help when it gets tight.
Disk	At least 20 GiB free in `/var/lib/vm-reactor` for a first test; more, or a second disk, for real guests and backups
Network	A fixed address of the host is recommended – the web interface and the guests are reached through it
Access	**root** (or a user with `sudo`) for the installation; afterwards a web browser is enough

> **WARNING**
> Install VM Reactor on a computer that is meant to be a host. It installs and starts libvirt, LXC and the memory setup of a hypervisor (zram, KSM), sets up its own NAT network and changes kernel settings for memory. On a desktop you use every day, test it in a virtual machine first.

3.1.1 Is hardware virtualisation available?

Machines need the KVM module of the kernel and the device `/dev/kvm`. Check it before you install:

```
# grep -c -E 'vmx|svm' /proc/cpuinfo     # number of cores with VT-x / AMD-V
8
# ls -l /dev/kvm
crw-rw---- 1 root kvm 10, 232 Nov  1 09:12 /dev/kvm
```

If the first command prints `0` or `/dev/kvm` is missing, switch virtualisation on in the firmware setup of the computer (often called *Intel Virtualization Technology* or *SVM Mode*). If the host is itself a virtual machine, its hypervisor must offer **nested virtualisation**.

> **REMEMBER**
> Without `/dev/kvm` VM Reactor still works: containers and applications run normally, and machines run **emulated** – correct, but very slowly. The **Overview** then shows a yellow light for KVM and a warning at the top.

3.2 The Stratum repository

VM Reactor is published in the Stratum repository, component **contrib**:

Item	Value
Address	`https://www.skraba.net/stratum-apt/`
Suites	`trixie` (Debian 13) and `forky` (Debian 14)
Components	`main` (Stratum and its modules) and **`contrib` (VM Reactor)**
Architectures	`amd64`, `arm64` (and source packages)

If Stratum is not installed yet, set up the repository exactly as described in the *Stratum Handbook* (chapter *Installing Stratum*): download the signing key, check its checksum and write `/etc/apt/sources.list.d/stratum.sources`. The quick setup there already lists both components, `main contrib`.

On a computer where Stratum is already installed, check that **contrib** is part of the source:

```
# grep Components /etc/apt/sources.list.d/stratum.sources
Components: main contrib
```

If only `main` is listed, add `contrib` to this line with an editor and run `apt-get update`.

3.3 Installing the package

```
# apt-get update
# apt-get install vm-reactor
…
Setting up vm-reactor (0.08) ...
vm-reactor: first sign-in as user admin, password in /etc/vm-reactor/initial-admin-password
         (change it in the web interface: Users - My account; then delete the file)
vm-reactor: web interface https://10.0.0.22:8443/
```

APT brings everything VM Reactor needs from Debian – libvirt and QEMU, LXC, nftables, dnsmasq, noVNC – together with `stratum-memory` and `ksmtuned` for the memory setup. It recommends `stratum-firewall`, LVM, NFS and the bridge utilities.

During the installation VM Reactor:

1. creates its directories – the configuration in `/etc/vm-reactor`, the guests, ISO images, templates and backups in `/var/lib/vm-reactor`, the logs in `/var/log/vm-reactor`;
2. starts libvirt and its own services, `vm-reactor.service` (the web interface) and `vm-reactor-guests.service` (starts and stops the guests with the host);
3. creates the first administrator, **admin**, with a random password, and a self-signed certificate for HTTPS;
4. switches the **memory setup** on: zram tuned for a hypervisor, KSM with the recommended values and the automatic balloon of the machines (chapter 11). What was set up before is left as it is.

```
# systemctl status vm-reactor --no-pager
● vm-reactor.service - vm-reactor - virtual machines and containers (web interface)
     Loaded: loaded (/usr/lib/systemd/system/vm-reactor.service; enabled; preset: enabled)
     Active: active (running) since Sun 2026-11-01 10:32:52 CET; 3min 55s ago
…
```


© 2026 Stratum Project

> **TECHNICAL STUFF**
>
> The web interface and the API are one program, `/usr/bin/vm-reactor serve`, written in Go, with the web interface built in. It listens on port `8443` on all addresses of the host; **Settings → General** changes the port and the address. With `stratum-firewall` the port is opened automatically.

3.4 First sign-in

1. **Read the first password** on the host:

```
# cat /etc/vm-reactor/initial-admin-password
Xq7mLw2Rk9TfBz4s
```

2. **Open the address** of the web interface in a browser – `vm-reactor url` prints it. The certificate is self-signed, so the browser warns once; accept it for this address. (A certificate of your own is described in chapter 15.)
3. **Sign in** as user `admin` with the password from step 1.
4. **Secure the account.** In **Users → My account** set a password of your own and switch on **two-factor sign-in** with an authenticator app. Then delete the file with the first password:

```
# rm /etc/vm-reactor/initial-admin-password
```

5. **Look at the Overview.** The lights in the **Host** card and the warnings at the top tell you what is still missing – for example hardware virtualisation or noVNC for the graphical console.

> **TIP**
>
> Give every person who works with VM Reactor an account of their own (**Users → Add user**), with the smallest role that is enough. The audit log then shows who did what.

3.5 Your first guests

The quickest way to see VM Reactor at work is a container: **Create guest → Container**, keep the proposed Debian image, give it a name and finish. A few seconds later it runs, gets an address in the NAT network and has a shell in the browser.

Chapter 4 takes you on a tour of the web interface; chapter 5 creates a machine step by step.

3.6 Updating and removing

VM Reactor is updated like every other package:

```
# apt-get update && apt-get upgrade
```

An update restarts the web interface but **not** the guests: machines, containers and applications keep running.

Removing the package works the same way. `apt-get remove vm-reactor` leaves the guests, their disks and the backups in place; `apt-get purge vm-reactor` also removes the settings, the users and the certificate of the web interface – the guests in `/etc/vm-reactor/guests` and `/var/lib/vm-reactor` are still kept and can be removed by hand.

> **WARNING**
>
> Removing VM Reactor does not stop running guests, but it removes the rules of the NAT network – guests in it lose the connection to the outside – and nothing starts the guests at the next boot any more. Shut them down first, or back them up, if the host is to be used for something else.

© 2026 Stratum Project

CHAPTER 4

The web interface

END USER ADMINISTRATOR

Everything you do with VM Reactor starts in its web interface: one page for the host, one list for all guests, and the same buttons for machines, containers and applications. This chapter takes you on a short tour.

4.1 Signing in

Open the address of the web interface – `https://<address of the host>:8443/` – and sign in with your user name and password. If two-factor sign-in is switched on for your account, VM Reactor then asks for the six-digit code of your authenticator app.

A session lasts as long as you work with it and ends after a time without activity, or with **Sign out** in the menu of your user name at the top right. After several wrong passwords from the same address, VM Reactor refuses further attempts for a while; the attempts are written to the audit log.

> **TIP**
> Bookmark the address. If you forget it, `vm-reactor url` on the host prints it.

4.2 The layout

The page has three areas:

- the **sidebar** on the left: the main menu and, below it, all guests of the host grouped as *Machines*, *Containers* and *Applications*, each with its ID and a coloured dot for its state;
- the **header** at the top: the name of the page, the switch between the light and the dark appearance, and your user name with your role;
- the **page** itself in the middle.

Menu	What you find there
Overview	The host at a glance: load, memory management, the state of the engines, the guests and the latest tasks.
Guests	All machines, containers and applications in one list, with filters.
Storage	The storages of the host, ISO images and LXC templates.
Memory	Compression, deduplication and the balloon of the machines.
Network	The NAT network, the local network, bridges and published ports.
Docker	Docker itself: installation, settings, images, volumes and registries.
Backups	All backups, schedules, retention rules, import and export.
Tasks	Every long-running piece of work with its complete log.
Users	Your account, the users and their rights, access to the web interface, the audit log.
Settings	General settings, the watchdog, optional packages, the Stratum integrations and information about VM Reactor.

What you see depends on your role: a viewer sees the pages but no buttons that change something, and a user with access to a few guests sees only those guests.

4.3 The overview

The **Overview** is the first page after signing in. It answers three questions at a glance: is the host healthy, how busy is it, and what has happened recently.

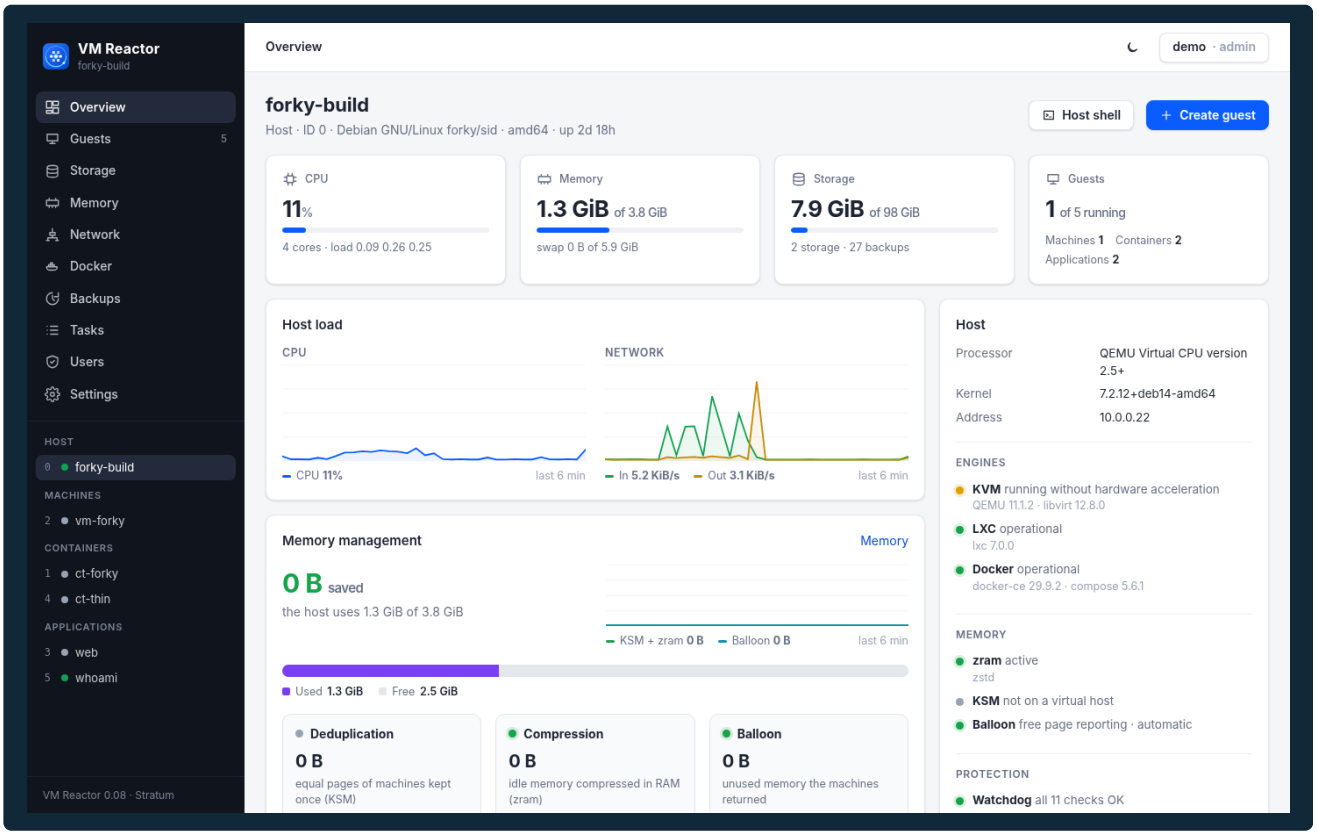


Figure 1 The Overview of a host: load, memory management and the Host card with its status lights

At the top, four cards show the **CPU** load, the **memory** in use, the **storage** in use and how many **guests** are running – by kind. Below them:

- **Host load** – the CPU and the network traffic of the last minutes, refreshed every few seconds;
- **Memory management** – how much memory the memory techniques of VM Reactor save at this moment (section 4.3.2);
- the **Host** card – processor, kernel and address of the host, and three groups of status lights (section 4.3.1);
- **Guests** and **Recent tasks** – the first guests of the list and the latest work.

The buttons at the top right open a **Host shell** – a terminal of the host in the browser, for administrators – and **Create guest**.

4.3.1 The status lights

The lights in the **Host** card follow one simple rule:

Light	Meaning	Example
● green	operational	*KVM operational*, *zram active*, *last backup 3 h ago*
● yellow	works, but needs your attention	*KVM running without hardware acceleration*, *last backup 3 days ago*

© 2026 Stratum Project

Light	Meaning	Example
● red	installed, but not working	*Docker installed, the daemon is not running, the last backup failed*
● grey	not installed or switched off – not a fault	*LXC not installed, KSM not on a virtual host*

The three groups are:

- **Engines** – KVM for machines, LXC for containers and Docker for applications, each with its versions;
- **Memory** – compression (zram), deduplication (KSM) and the balloon of the machines;
- **Protection** – the watchdog and the latest backup.

4.3.2 Memory management at a glance

The panel **Memory management** shows what VM Reactor gets out of the memory of the host. The big number is the memory saved right now; the sentence below compares the memory the running guests were given with what the host really uses.

The bar shows the memory of the host by use: what is **used**, what **KSM** saves by keeping identical pages of machines only once, what **zram** saves by keeping idle memory compressed, what the machines have **returned** through their balloon, and what is still **free**. Three tiles below the bar show each technique with its own light, and the last line the swap in zram and on disk.

> **REMEMBER**
>
> The memory techniques are switched on when VM Reactor is installed. On a host with few, idle guests the savings are small – they grow with the number of similar machines and with the pressure on memory. Chapter 11 explains each technique and its settings.

4.4 The list of guests

Guests lists all machines, containers and applications of the host – one list, one ID each. The filters at the top show only machines, containers, applications or the guests that run; a click on a row opens the guest.

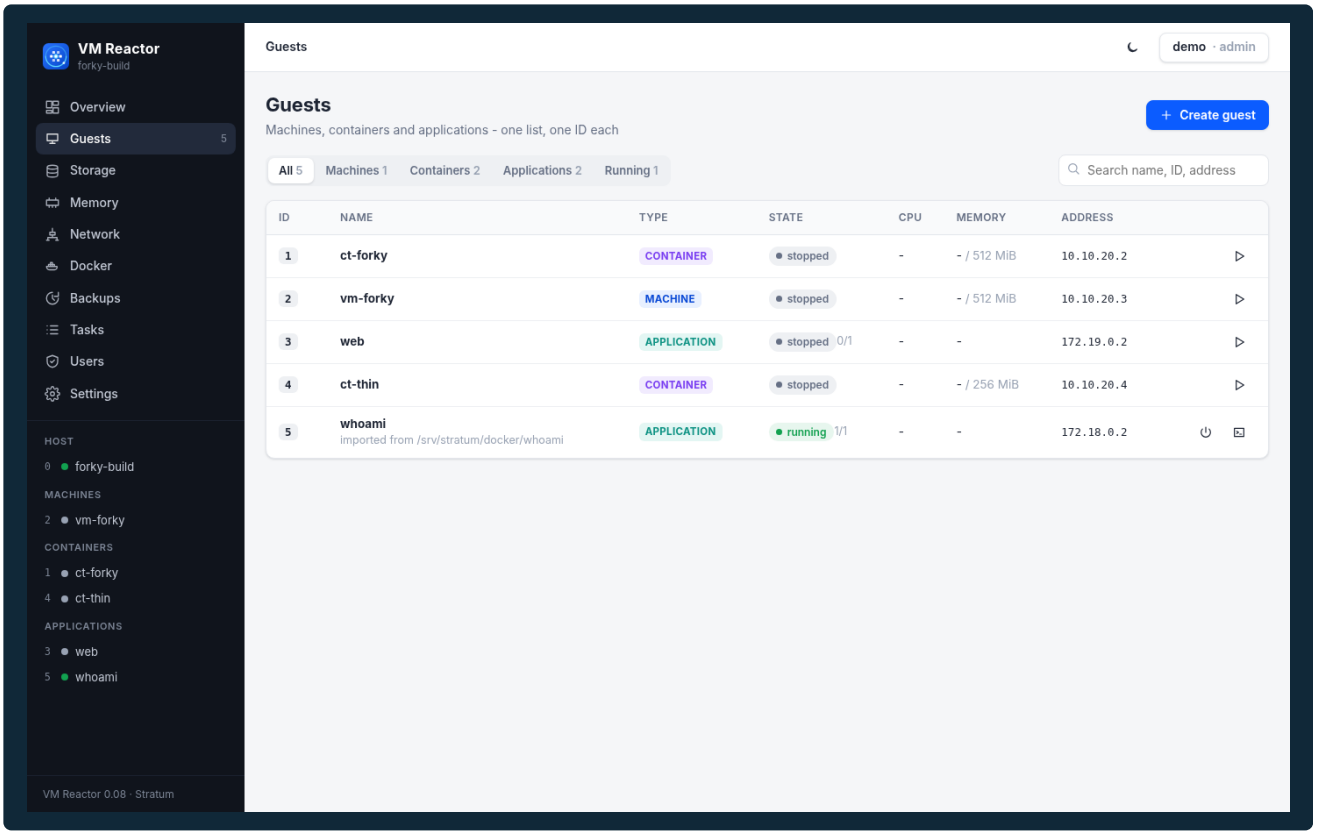


Figure 2 Guests: machines, containers and applications in one list

Column	Meaning
ID	The number of the guest – the same in the web interface, on the command line and in file names.
Name	The name and the description of the guest.
Type	*Machine*, *Container* or *Application*.
State	*running*, *stopped*, *paused*; for applications also how many of their containers run.
CPU, **Memory**	The current load and memory of a running guest, and the memory it may use.
Address	The address of the guest in its network.

The buttons at the end of a row start and shut down a guest and open its console, without opening the guest first.

4.5 A guest

The page of a guest has the same layout for every kind of guest: a header with the name, the type and the state, buttons for the most frequent actions, and tabs.

Tab	Content
Summary	State, load, memory, disks, address and the most important settings.
Console	The screen or the serial console of a machine, the shell of a container.
Hardware / **Resources**	Processors and memory; the disks and the CD drive of a machine.
Network	Network, address and published ports.
Firewall	The rules of a machine or a container.

© 2026 Stratum Project

Tab	Content
Snapshots, **Backups**	Snapshots and backups of this guest.
Logs, **Options**	The log of the guest; start with the host, watchdog, description, deletion.

Changes that need a restart of the guest – more memory for a machine, for example – say so next to their **Save** button.

4.6 Tasks

Creating a guest, a backup, a restore, a download: everything that takes longer than a moment runs as a **task**. A small window shows its progress while you wait; you can close it and keep working, and the task carries on.

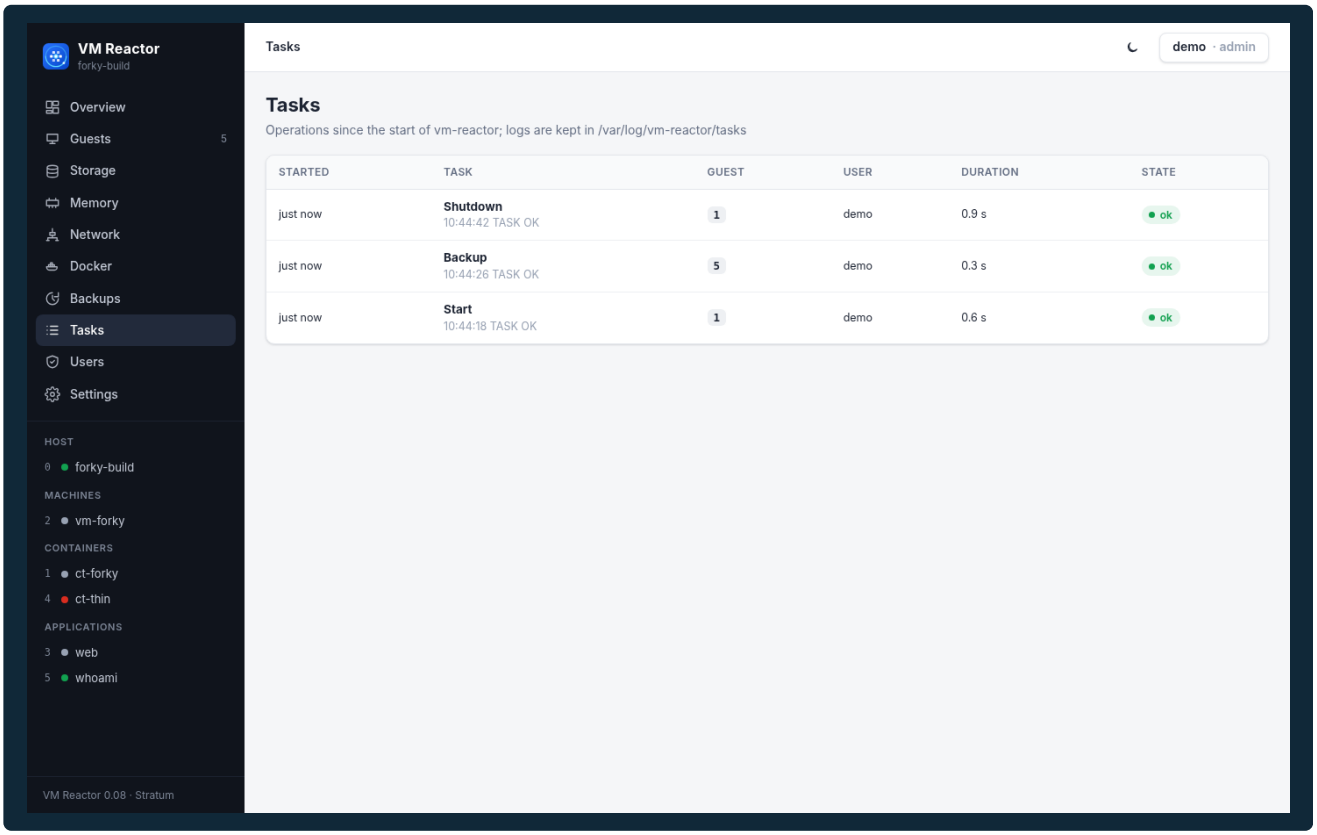


Figure 3 Tasks: every long-running piece of work with its state, user and log

Tasks lists the tasks since the last start of VM Reactor with their guest, the user who started them, their duration and their state. A click opens the complete log – the first place to look when something did not work. The logs of older tasks are kept as files in `/var/log/vm-reactor/tasks`.

4.7 Light and dark

The moon or sun symbol in the header switches between the light and the dark appearance. The choice is stored in your browser. All screenshots in this book use the light appearance.

VM REACTOR HANDBOOK

Machines, containers, applications. One host. One list.

VM Reactor turns a Debian computer into a host for guests: KVM virtual machines, LXC system containers and Docker applications side by side – one number for every guest, one place to start, watch, back up and protect them. Your host remains Debian.

This book takes you through it, step by step:

- install VM Reactor and find your way around its web interface in minutes;
- create machines, containers and applications, open their consoles and snapshots;
- storage, networks and the memory techniques that fit more guests into the same RAM;
- backups with retention rules, users and roles, and every command of `vm-reactor`.

END USER ADMINISTRATOR



Stratum Project
VM Reactor 0.24 · Draft · November 2026
www.skraba.net/vm-reactor