

MCZ

Mega Compression Zip — User Manual for Linux
Command line and GUI

This manual describes **MCZ version 1.9.18**.

Issued 16 November 2026.

To check which version you have installed, run `mcz version` in a terminal, or read the title bar of the GUI, which shows `MCZ: Version 1.9.18`. If the numbers differ from this page, fetch the manual that matches your software from the release it came with.

Contents

1. What MCZ is
    1.1 Two editions: `mcz` and `mcz-gui`
    1.2 Licence
    1.3 Premium features
2. Installing
    2.1 Installing and upgrading
    2.2 Checking a download
    2.3 Where MCZ keeps its files
3. The GUI (`mcz-gui`)
    3.1 The window
    3.2 Browser tab
    3.3 Backup tab
    3.4 Compress tab
    3.5 Profile tab
    3.6 MCZ Copy
    3.7 MCP tab
    3.8 Licence tab
    3.9 Updates tab
    3.10 The job queue (MCZ FIFO Server)
    3.11 File manager and command-line verbs
4. The command line (`mcz`)
    4.1 Commands at a glance
    4.2 Compressing
    4.3 Extracting, verifying, listing, adding
    4.4 Profiles
    4.5 Licence and GPU
    4.6 Testing and benchmark commands
    4.7 Exit codes
    4.8 Zip files
    4.9 7z files
    4.10 Self-extracting archives
    4.11 Signed archives
    4.12 Converting MCZ archives to zip, 7z or tarballs
    4.13 Tarballs (.tar.gz, .tar.xz, .tar.bz2)
    4.14 Damaged and resilient archives
5. Profiles explained
6. Optimising speed and size

6.1 What sets the pace
6.2 Smaller archives for less time
6.3 Power and cooling
7. Archive compatibility and safety
8. Backing up with scripts
9. Troubleshooting

1. What MCZ is

MCZ is a lossless archiver built for speed on multi-core PCs. It compresses on every processor core at once (and, optionally, on an NVIDIA GPU), checks every byte it writes and reads, and produces one `.mcz` file. It was built to back up large Unreal Engine projects to USB disks: on a 32-core workstation it archives a 111 GB project in under two minutes where 7-Zip needs over twenty, at a size within a few per cent of 7-Zip's.

Three things set it apart from a zip tool:

- **Everything is parallel.** Compressing, writing, extracting and verifying all use every core, so the disk, not the CPU, is normally the limit.
- **Every byte is checked.** Extraction and *Verify* check everything they read, and a damaged archive is reported rather than silently producing bad files. Damage costs only the files it touches, and a **resilient archive** (4.14) survives damage to its file list too.
- **Settings choose themselves.** A per-user *profile* measures the PC once and each destination drive once, and picks the settings that keep the destination the bottleneck (section 5).

1.1 Two editions: `mcz` and `mcz-gui`

Edition	Executable	For
Command line	`/usr/bin/mcz`	Scripts, scheduled backups, terminals, servers. Section 4.
GUI	`/usr/bin/mcz-gui` (Qt 6)	Interactive use: Browser, Backup, Compress, Profile, MCP and Updates tabs; opens from the desktop menu or a file manager. Section 3.
MCZ Copy	`/usr/bin/mcz-copy` (Qt 6)	Share folders with other PCs by PIN and copy both ways. Section 3.6 and the *MCZ Copy User Manual*.

Both use the same engine and the same profile, and both read and write the same `.mcz` archives. Anything compressed with one can be extracted with the other, and archives move freely between Windows and Linux installations.

1.2 Licence

MCZ is free for personal use. A PC joined to an Active Directory or Entra ID domain is treated as commercial use and needs a licence key before it will compress, extract or verify; listing and the GPU report still work. The GUI's **Licence** tab (3.8) and `mcz license` show the state and the **Machine ID** a key is issued for, and accept a key. A key is a line of text starting `MCZ1-`; it is verified offline. The quickest way to licence one PC is the Licence tab: pick a licence, pay by card and the key installs itself. The tab also gives a free 30-day evaluation key. For many PCs, order at mcz.software/buy (prices at mcz.software/pricing; evaluations for many PCs at mcz.software/evaluate). When an annual or evaluation key expires, MCZ keeps working for 30 more days with a reminder; after that it is read-only (extract, verify and list) until a renewed key is installed. Terms are in `LICENSE.txt` in the install folder.

1.3 Premium features

The network features (MCZ Copy, the MCZ Network Service, `mcz peers`, `mcz send` and `mcz receive`), the MCP server for AI assistants and the MCZ FIFO Server (the job queue) are premium features. They are free for **30 days** from the first time MCZ is installed or run on a PC. After that they need a **premium key**, which costs £5 and is for one PC. Compressing, extracting, verifying and listing are never affected: once the trial has ended, MCZ jobs simply stop taking turns and run at once, as they do without the job queue. A commercial licence includes the premium features.

MCZ Copy and the MCP tab show how many days of the trial are left. To buy a key, open the **Licence** tab (3.8), choose **Premium only** and pay by card; the key installs itself. You can also buy one at mcz.software/premium with the PC's **Machine ID** (shown on the Licence tab and by `mcz premium`), then paste the key into the Licence tab and press **Install key**, or run `mcz license <key>`. The key takes effect at once, including for the network service; there is nothing to restart. For MCZ Copy, both PCs need the premium features: the sharing PC's network service turns connections down without them, and MCZ Copy and the `mcz copy-*` commands on the connecting PC ask for them too.

2. Installing

2.1 Installing and upgrading

MCZ needs a 64-bit x86 processor with AVX2 (Intel 2013 or later, AMD Ryzen). Debian, Ubuntu (24.04 and compatible) and derivatives:

```
sudo apt install ./mcz_1.9.18_amd64.deb
```

This puts `mcz`, `mcz-gui`, `mcz-copy` and the network service `mcz-netsvc` in `/usr/bin` (already on PATH), the shared library in `/usr/lib`, the job queue in `/usr/lib/mcz/mcz-fifo`, this manual, the MCZ Copy manual and the licence under `/usr/share/doc/mcz/`, and registers the `.mcz` MIME type and desktop entries for MCZ and MCZ Copy, so double-clicking an archive in a file manager opens it like a folder, and adds an MCZ right-click menu to GNOME Files (section 3.11). It installs two systemd services:

- `mcz-fifo`, the job queue (section 3.10), is enabled and started; `sudo systemctl disable --now mcz-fifo` turns it off.
- `mcz-netsvc`, the MCZ Network Service that MCZ Copy shares through (section 3.6), is enabled and started too; nothing is shared until someone makes a PIN or chooses a no-PIN folder, and `sudo systemctl disable --now mcz-netsvc` turns it off.

Other distributions can use the tarball, which has the same layout under `usr/`:

```
sudo tar -xzf mcz-1.9.18-Linux.tar.gz -C / --strip-components=1
```

Upgrading is the same command with the newer package. `sudo apt remove mcz` uninstalls; your profile and licence key in `~/.config` are kept. The GPU paths use CUDA when an NVIDIA driver is present and are skipped otherwise; nothing else is needed.

2.2 Checking a download

Every release page carries `MCZ-1.9.18-SHA256SUMS.txt` (the SHA-256 checksum of each file), a signature over that list made with the author's key (`.sig`) and the public key `mcz-release-public.pem`. A download that does not match the list, or a list whose signature does not verify, is not the published release and should not be installed.

```
sha256sum -c MCZ-1.9.18-SHA256SUMS.txt
openssl dgst -sha256 -verify mcz-release-public.pem -signature MCZ-1.9.18-
SHA256SUMS.txt.sig MCZ-1.9.18-SHA256SUMS.txt
```

2.3 Where MCZ keeps its files

What	Where
Profile (machine and drive measurements, named profiles)	`~/.config/mcz/profile.json` (`$XDG_CONFIG_HOME` is honoured)
Licence key	`~/.config/mcz/license.key` and `/var/lib/mcz/license.key` (so the network service sees it), or `/etc/mcz/license.key` for every user
GUI settings (last folders, divider positions, *Skip locked files*)	`~/.config/MCZ/MCZ.conf`
Premium key and trial date	`premium.key` and `trial.dat`, in both `/var/lib/mcz` and `~/.config/mcz`
Job queue record (one file per month)	`/var/lib/mcz-fifo`
Signing keys (`*.mczkey`, owner only) and trusted signers (`trusted.txt`)	`~/.config/mcz/keys`
MCZ Copy: PINs this PC shares, the no-PIN folder	`/etc/mcz/copy-pins.json`, `netservice.json` (root only; the PINs themselves are not stored)
MCZ Copy: PCs connected to before (PINs encrypted for you on this PC), last folder	`~/.config/mcz/copy-remembered.json` (+ `.key`, owner only), `~/.config/MCZ/MCZ Copy.conf`
Network service log	`/etc/mcz/netservice.log`, also in `journalctl -u mcz-netsvc`
GUI messages	standard error of `mcz-gui` and `mcz-copy`
Programs, library, manuals, licence text	`/usr/bin/mcz`, `mcz-gui`, `mcz-copy`, `mcz-netsvc`, `/usr/lib/libMczCore.so`, `/usr/lib/mcz/mcz-fifo`, `/usr/share/doc/mcz/`
Services and desktop entries	`/usr/lib/systemd/system/mcz-fifo.service`, `mcz-netsvc.service`; `/usr/share/applications/mcz.desktop`, `mcz-copy.desktop`

The profile is plain JSON. An unreadable profile (for example after a hand edit) is set aside as `profile.json.bad` and a fresh one is created; nothing is overwritten silently.

3. The GUI (`mcz-gui`)

3.1 The window

The title bar reads `MCZ: Version 1.9.18` and, once an output archive is chosen, its path in square brackets. The header shows the product name and a link to **mcz.software**; a warning line appears under it when the GPU can help only partly or not at all. The GPU that was detected is named in the log (`GPU: ...`). The status bar at the bottom of the window shows the licence state and the PC's Machine ID, with the **Licence...** button on its right, which opens the Licence tab. Below the tabs are a progress bar and the log, which records what happened, including the line `Profile: ...` that states exactly which settings a compression used.

The window takes its colours from the desktop theme, so it follows light and dark mode and updates when the theme changes.

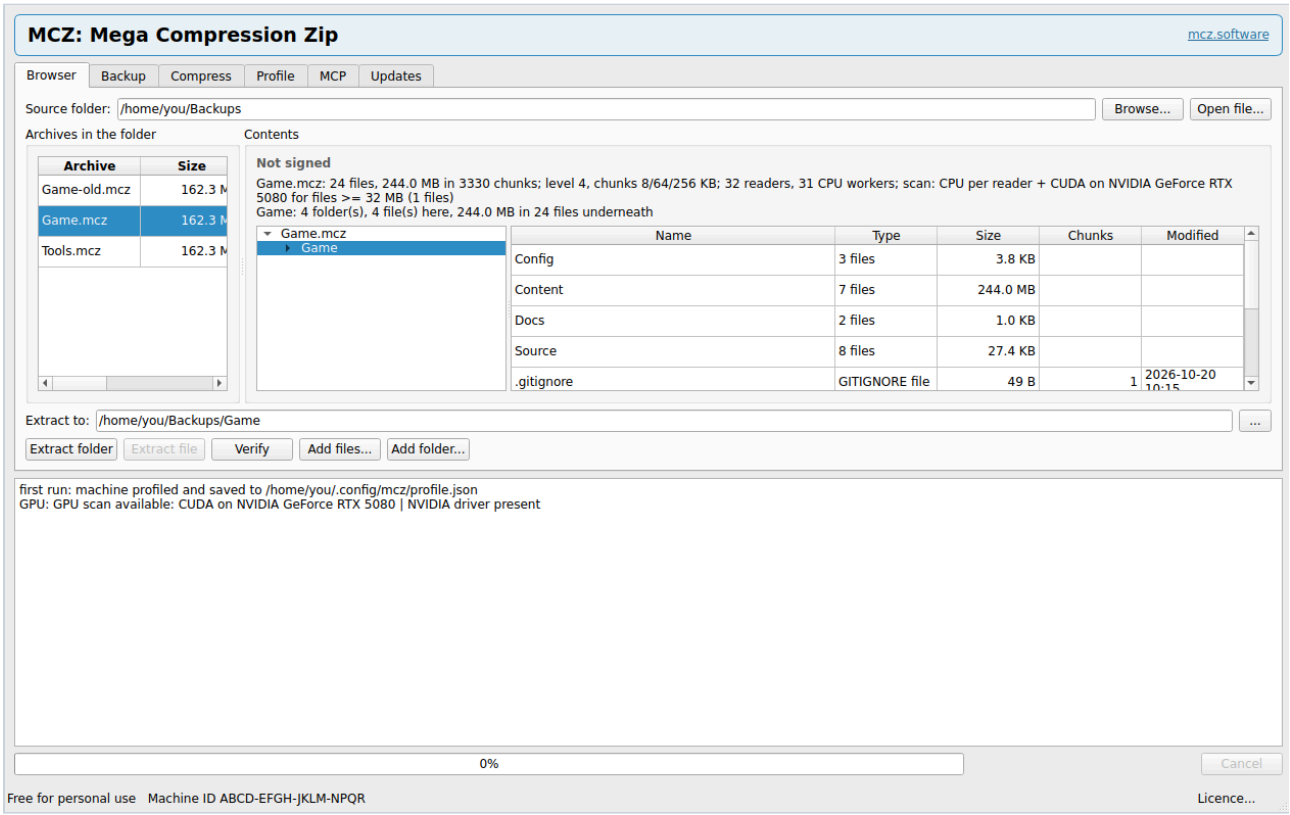
The window has seven tabs, one job each (sharing with other PCs is the separate MCZ Copy program, 3.6):

Tab	What it is for
Browser (3.2)	Look inside archives, extract and verify.
Backup (3.3)	Back up several folders in one go, one archive each.
Compress (3.4)	Make one archive from any files and folders.
Profile (3.5)	Decide how archives are written.
MCP (3.7)	Let an AI assistant such as Claude drive MCZ. A premium feature.
Licence (3.8)	See the licence state, buy a licence for this PC by card, or install a key.
Updates (3.9)	Check for a newer version.

The sections below take the tabs in that order and go through each tab's controls from top to bottom. You don't have to open the window first for everyday jobs: your file manager can compress, open, extract and verify directly (3.11).

3.2 Browser tab

Look inside archives, extract, verify and add files.



The Browser tab.

The first tab. Double-clicking a `.mcz` file in a file manager opens the GUI here with the archive loaded.

1. **Source folder.** Type or **Browse...** a folder and every `.mcz` in it is listed on the left with its size and date. **Open file...** shows one archive from anywhere. The dividers between the list, the tree and the file view can be dragged; their positions, the source folder, the Backup tab's target folder and the Compress tab's output folder are remembered for next time.
2. **Signature.** A coloured line above the contents says who signed the archive (4.11): green *Signed by Pete (trusted)*, amber for a signer you have not trusted yet (with the key's fingerprint), red *Signature INVALID* when the archive was changed after it was signed (extracting it is refused), grey *Not signed*.
3. **Contents.** Like a file manager: the tree on the left shows the archive's folders (expand as you go; the top entry is the archive itself), the view on the right shows the folders and files of the selected folder with type, size, stored size, ratio and modification time. For a folder, Type shows how many files it holds, sub-folders included (for example 1,235 files), and Size shows their total size. Double-click a folder to open it; select several files with Ctrl or Shift. The line above summarises the archive and the folder you are in.
4. **Extract to** is pre-filled with a folder named after the archive, next to it, and follows each archive you pick; change it with **...** or type a folder, and your folder is kept when you pick another archive (clear the box to go back to following the archive). Extracted files keep their folder structure under it.
5. **Extract folder** writes the folder selected in the tree (or in the file view) with everything under it; the top entry extracts the whole archive. When an archive holds a single top folder, that folder is selected when the archive opens. **Extract file** writes the file(s) selected in the file view. **Verify**

decodes the whole archive and checks every hash without writing anything. **Cancel** on the progress row stops any of them.

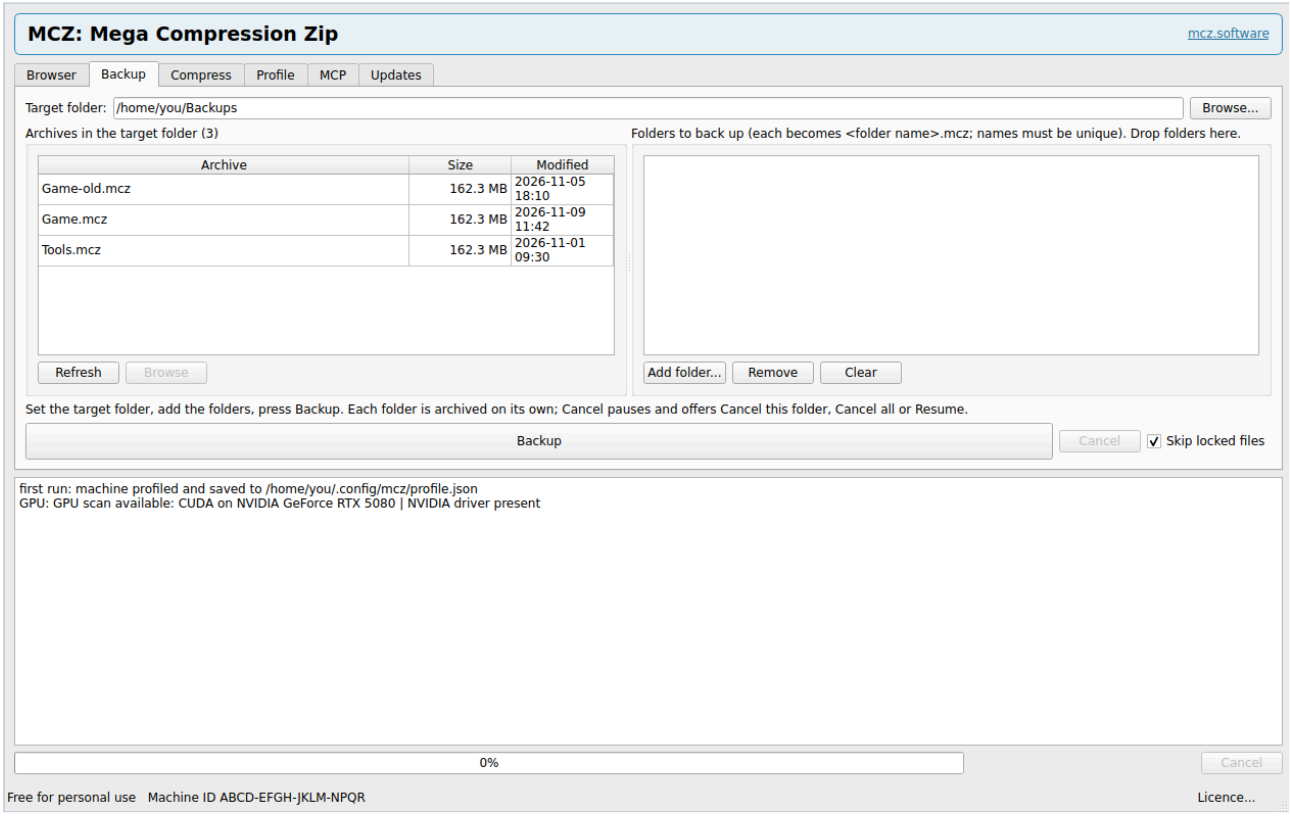
6. **Add files...** and **Add folder...** add to the archive, into the folder shown in the file view. You can also drag files and folders from a file manager onto the file view or the tree: dropped on a folder, they go into that folder. A file already in the archive under the same name is replaced; MCZ lists those names and asks first.

Adding works like 7-Zip: MCZ writes a new copy of the archive with the added files in it, checks that every file in the new copy reads back correctly, and only then puts it in place of the old one. The files already in the archive are not compressed again, so adding takes about as long as copying the archive plus compressing the new files; the progress bar runs three times (compressing, copying, checking). While it runs there must be room on the drive for a second copy of the archive. A cancelled or failed add leaves the archive exactly as it was, and so does an add during which another program changed the archive (MCZ says so and keeps the other program's version). Files that are replaced no longer take up space, so the archive does not keep growing when files are updated. The new files are compressed with the active profile, as on the Compress tab.

Files already present in the output folder are never overwritten silently: the extraction stops and asks first, and each file is replaced only once its new copy is complete. A chunk that fails its hash stops the operation with the file name and offset in the log; the incomplete output file is deleted. Timestamps are restored on extracted files. MCZ never extracts through a link: if a folder inside the output folder is a symbolic link, or a file there is one, the extraction stops before writing anything and names it.

3.3 Backup tab

Back up several folders in one go, one archive each.



The Backup tab.

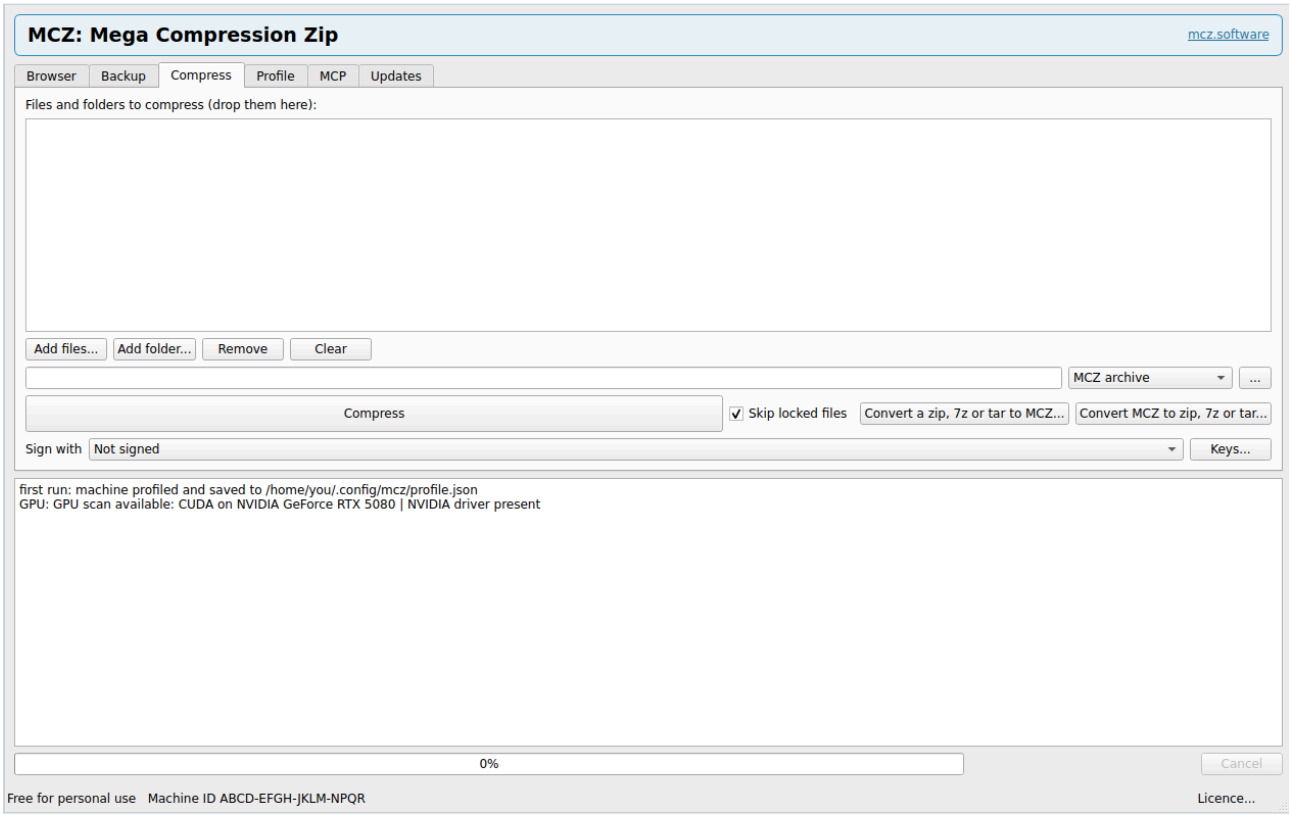
The Backup tab is for the everyday job: several project folders into one target folder.

1. **Target folder.** Type it or **Browse…**; it is created if it does not exist. Every archive goes here. The left pane lists the archives already in it (name, size, date); **Refresh** reads the folder again, for archives that arrived since, for example in the no-PIN folder (3.6); select one and press **Browse** (or double-click it) to see its contents on the Browser tab. The divider between the panes can be dragged and its position is remembered.
2. **Folders to back up** (right pane). **Add folder…** or drag folders onto the list; **Remove** takes out the selected folders and **Clear** empties the list. Each becomes `<folder name>.mcz` in the target, so two folders with the same name cannot both be in the list; files and drive roots are not accepted.
3. **Backup.** The folders are archived one after another with the active profile's settings for the target drive. If `<folder name>.mcz` already exists you are asked, as the backup reaches that folder, whether to replace it (the old archive is removed only once the new one is complete) or skip it.
4. **Cancel** pauses the compressor and offers three choices. **Cancel this folder** removes that folder's unfinished archive and moves on to the next folder; **Cancel all** does the same and leaves the remaining folders unstarted; **Resume** carries on where it paused. Archives already written are never touched. **Skip locked files** next to Cancel (on by default; one switch, shared with the Compress tab; remembered) leaves out files another program holds open and lists them in the log; switched off, the backup fails on the first such file. **Resilient archive** beside it (off by default, shared with the Compress tab) makes each backup archive resilient (4.2).

Each folder gets a line in the log (time, files, folders, size as a percentage of the original) and the run ends with a summary: archived, skipped, cancelled and failed counts, total size and time. With a workspace folder set on a fast drive (Profile tab), archives for a slow destination are compressed there and copied over afterwards, the copy of one folder running while the next folder compresses; a second progress bar shows the copy, with which copy of how many it is. The backup counts as running until the copies are done: the tab stays locked and **Cancel** offers to stop the copies.

3.4 Compress tab

Make one archive from any files and folders.



The Compress tab.

1. **Files and folders to compress.** Use **Add files...** and **Add folder...**, or drag items from your file manager onto the list. **Remove** takes out the selected rows, **Clear** empties the list. Folders are archived with their full tree.
2. **Output archive.** Filled in automatically from the first item (`Name.mcz` next to it); edit the path, or use the **...** button to pick a folder and name. The title bar follows this field. The list beside it chooses the format: **MCZ archive**, **Zip (opens anywhere)** for people without MCZ (4.8), **7z (7-Zip compatible)** for people who use 7-Zip (4.9), **tar.gz (Linux, macOS)** or **tar.xz (smaller)** for Linux and macOS (4.13), or **Self-extracting** for a program that unpacks itself without MCZ (4.10); choosing one changes the name's extension, and a name ending in `.zip` , `.7z` , `.tar.gz` , `.tar.xz` or `.run` selects that format.
3. The grey line above it says where the settings will come from: the selected profile, or the manual settings if the profile is switched off on the Profile tab.
4. **Compress.** Progress (files, bytes, speed) appears in the status area; **Cancel** stops and deletes the partial archive. **Skip locked files** next to it leaves out files another program holds open and lists them in the log; switched off, the job fails on the first such file. **Resilient archive** (off by default; shared with the Backup tab; remembered) keeps a second copy of the archive's file list and keeps damage to the data local, for about 5 % more space (4.2). When it finishes, the log shows the time taken, the number of files and folders, and the archive size as a percentage of the original. **Convert a zip, 7z or tar to MCZ...** picks a `.zip` , `.7z` or tarball and turns it into a `.mcz` of the same name beside it (4.8, 4.9, 4.13); the original is kept. **Convert MCZ to zip, 7z or tar...** does the reverse: it picks a `.mcz` , asks which format (zip, 7z, tar.gz or tar.xz), and writes a standard file of the same

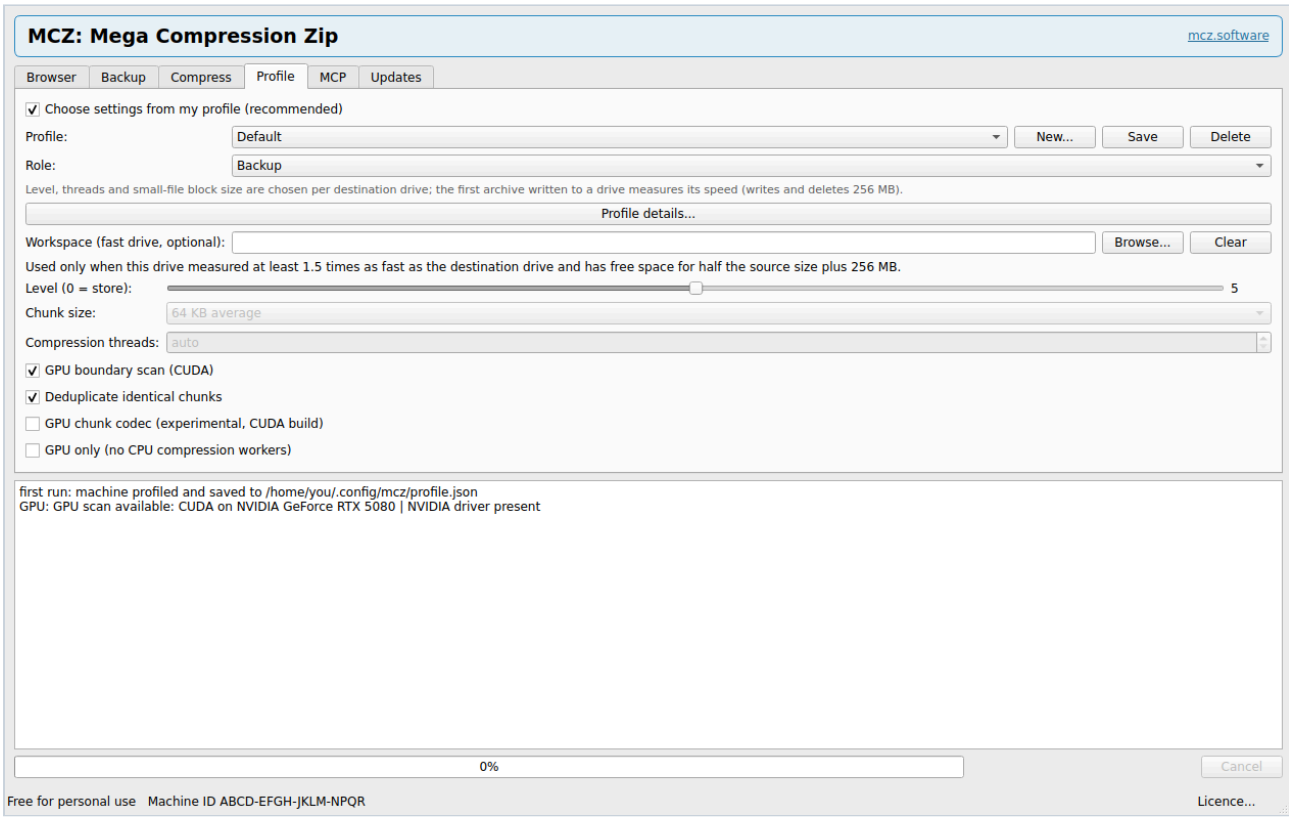
name beside it for people without MCZ (4.12). The first time an archive is written to a drive MCZ measures the drive by writing and deleting a 256 MB file next to the archive; the log says so, and the measurement is kept for every later archive on that drive.

Sign with (below the buttons) signs every `.mcz` you make from then on, here, on the Backup tab and when adding files, with one of your keys; **Not signed** turns it off. **Keys…** makes a key, copies your public key to send to others, and lists the signers you trust (4.11). A key locked with a passphrase asks for it once and keeps it in memory until MCZ closes. Zip, 7z and tar files are never signed.

> **Tip.** For a backup to a USB disk leave the profile on with the *Backup* role: the disk is the bottleneck, so the smallest archive is also the fastest one to write.

3.5 Profile tab

Decide how archives are written.



The Profile tab.

Most people never need to change anything here: leave *Choose settings from my profile* on and pick the role that fits the job.

Control	What it does
Choose settings from my profile (recommended)	On: the compression settings come from the selected profile and the destination drive. Off: the *Manual settings* below are used instead.
Profile selector, **New…**, **Save**, **Delete**	Several named profiles can exist (for example *Backup to USB* and *Quick local*); the selected one is active for both editions. **New…** asks for a name (up to 63 characters) and creates a profile from the settings shown, asking first if the name exists. **Save** writes the role and switches shown into the selected profile. **Delete** removes it (the last profile cannot be deleted).
Role: **Backup**, **Fast archive**, **Share**	Backup: the smallest archive the CPU can afford. Fast archive: lighter compression for quick local round trips. Share: small blocks so other people can pull single files out quickly. The role is saved into the profile as soon as it is changed.
Profile details…	Shows the machine measurements, every drive measured so far and all profiles, and lets you re-probe the machine. The whole Profile tab is locked while any job runs (compress, backup, add, extract, verify or a re-probe); a job always keeps the settings it started with.
Workspace (fast drive, optional)	A folder on your fastest drive, for example an NVMe SSD. When an archive is bound for another drive (a USB disk, a network share) and the workspace drive measured at least one and a half times as fast as that drive, it is built in the workspace first and then copied to its destination, with its own copy progress bar; the copy of one folder overlaps the compression of the next during a backup. Set it with **Browse…**; **Clear** removes it. The workspace drive also needs free space for half the source size plus 256 MB; otherwise, or when the destination is not that much slower, the archive is built directly on the destination. Temporary files are removed when the copy completes or if anything fails. Leave it empty to always write directly.
Engine switches	*GPU boundary scan (CUDA)* (lets the GPU help with large files), *Deduplicate identical chunks*, *GPU chunk codec* (experimental, archives come out a little larger), *GPU only* (no CPU compression workers). Each change is saved into the selected profile immediately.
Manual settings (greyed out while the profile is on)	**Level** 0 to 10 (0 stores, 1 fastest, 9 small, 10 *Ultra*; 5 is the usual choice). Ultra makes the smallest archives but takes several times longer, so it suits archives that are written once and kept, **Worker threads** (0 = all hardware threads), **Average chunk** 16 to 256 KB (the default, 64 KB, suits most data).

The text on the right of the tab summarises the idea: a profile decides how an archive is written, and the drive it is written to decides how hard the CPU should work. Section 5 has the rules.

3.6 MCZ Copy: sharing folders with other PCs

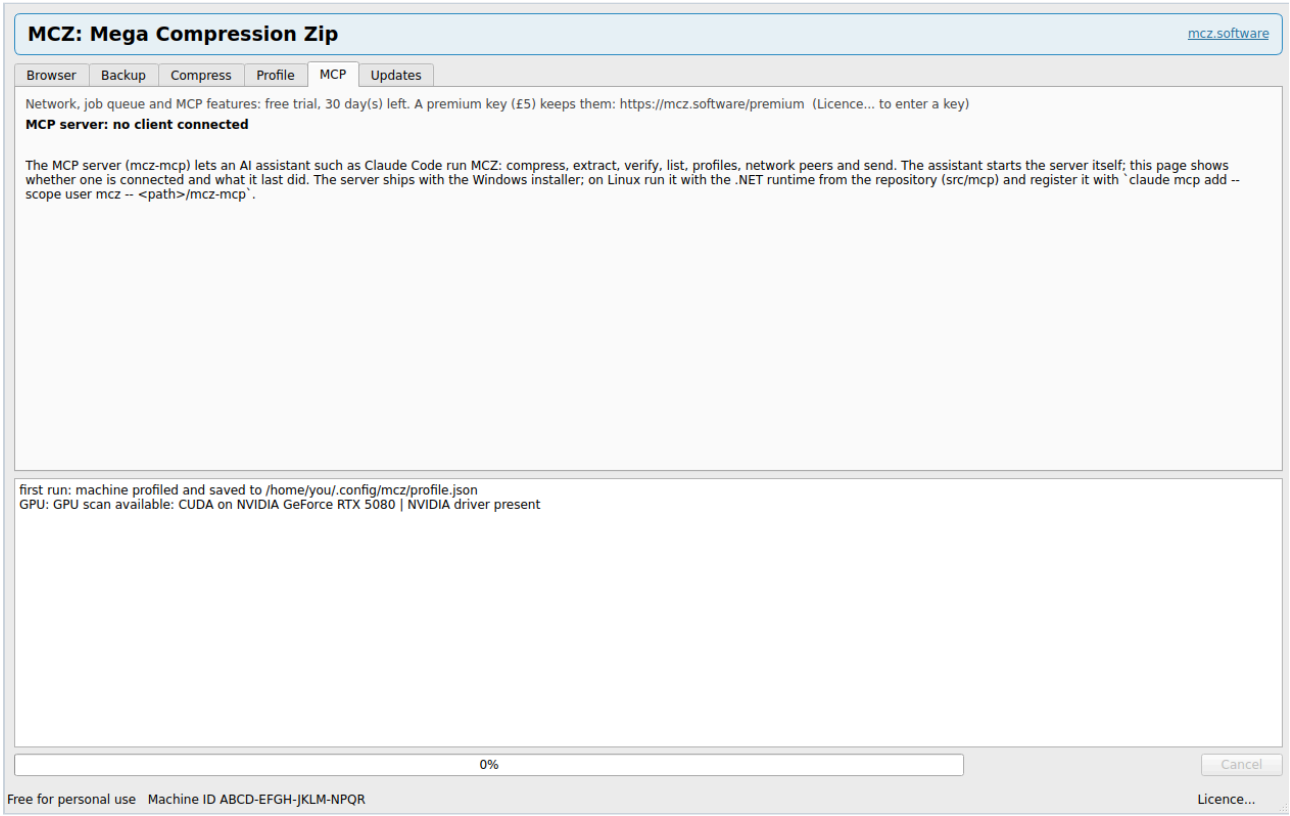
Share files and folders with a PIN, browse another PC and copy both ways.

MCZ Copy is a separate program next to `mcz-gui` (`mcz-copy`, in the desktop menu as MCZ Copy), with its own manual, the **MCZ Copy User Manual** (`/usr/share/doc/mcz/MCZ-Copy-Manual-Linux.pdf`). It replaces the Network tab of earlier versions. In short:

- The PC that shares runs the **MCZ Network Service** in the background (systemd service `mcz-netsvc`, started by the package). On MCZ Copy's **Share** tab its owner makes **PINs**, each opening chosen folders and files, read-only or read-write, and gives a PIN to a colleague.
- On the **Browser** tab the colleague finds the PC, connects with the PIN, and copies with **Get** and **Send**. Everything travels compressed and encrypted. Copying into a folder that already holds the files warns first, then skips identical files, replaces older ones and keeps newer ones; **Overwrite all existing files** in that warning (or `--overwrite` on the command line) replaces every file already there. An interrupted transfer can be finished later. Folders of any depth work, including paths longer than 260 characters.
- The **no-PIN folder**, also on the Share tab, takes archives sent with `mcz send` from other MCZ machines without a PIN. None is set unless you choose one.
- Get and Send take their turn in the job queue (3.10) on both PCs, like any other MCZ job.
- MCZ Copy is a premium feature (1.3). The command line has the same functions: `mcz copy-pin`, `copy-connect`, `copy-ls`, `copy-mkdir`, `copy-get`, `copy-send` (4.1).

3.7 MCP tab

Let an AI assistant such as Claude drive MCZ.



The MCP tab.

The MCP server is a premium feature: free for 30 days, then it needs a premium key (see 1.3). Once the trial has ended and until a key is installed, the server still compresses, writes zip, 7z and tar files, converts, makes self-extracting programs, signs, extracts, verifies and lists (`mcz_compress`, `mcz_zip`,

`mcz_7z`, `mcz_tar`, `mcz_convert`, `mcz_sfx`, `mcz_sign`, `mcz_extract`, `mcz_verify`, `mcz_list`) and answers `mcz_license`, `mcz_version` and `mcz_activity`, so an assistant can still show the Machine ID and install a key for you; the other tools need the key.

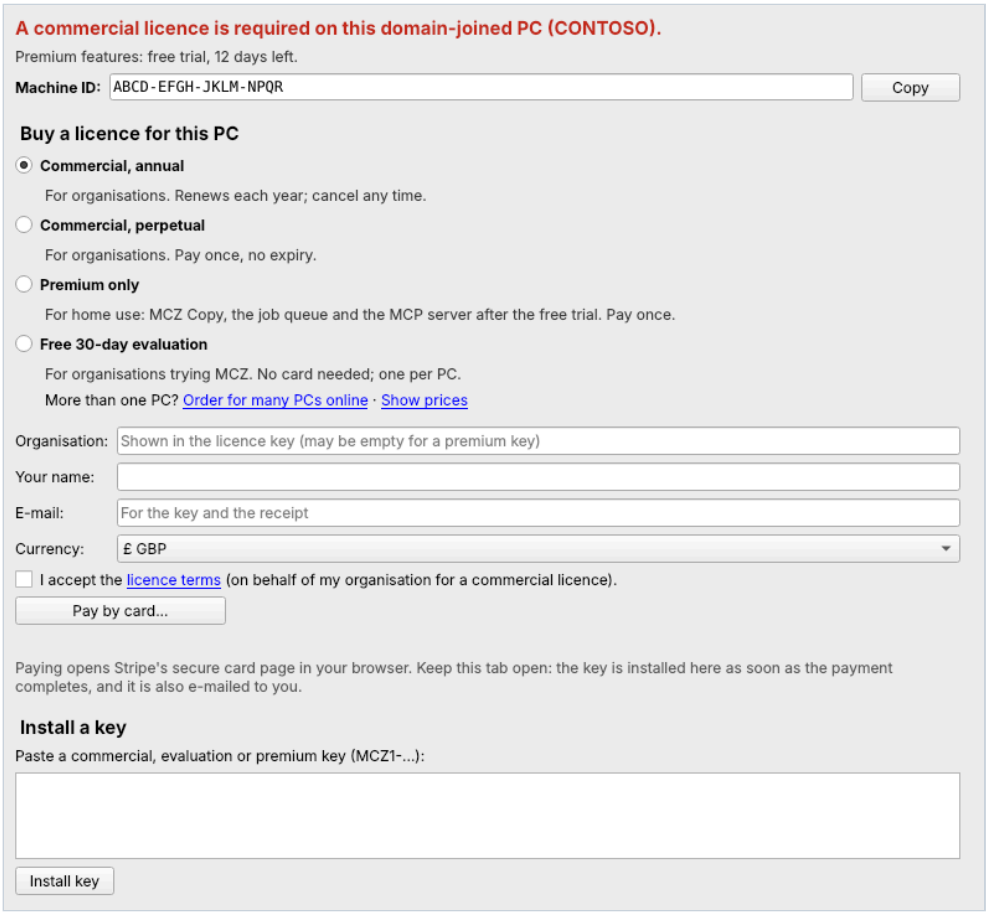
MCZ can be driven by an AI assistant through the Model Context Protocol. The server (`mcz-mcp`, a .NET program in the repository's `src/mcp`) is registered with `claude mcp add --scope user mcz -- /path/to/mcz-mcp`. The assistant starts the server itself when it needs it, so there is nothing to run by hand.

The server exposes the command line as tools: `mcz_compress`, `mcz_zip`, `mcz_7z`, `mcz_tar`, `mcz_convert`, `mcz_sfx`, `mcz_sign`, `mcz_extract`, `mcz_verify`, `mcz_list`, `mcz_profile`, `mcz_peers`, `mcz_send`, `mcz_receive_start` / `_status` / `_stop`, `mcz_gpu`, `mcz_license`, `mcz_selftest`, `mcz_cmp`, `mcz_gentest`, `mcz_bench` and `mcz_version`, plus `mcz_activity`, which tells the assistant how busy MCZ is: the jobs running and the jobs queued (section 3.10). Jobs an assistant starts take turns like any other, and their progress updates say when they are waiting in the queue. Each one runs the real `mcz` command, so the same rules apply: an existing archive or file is never replaced unless the assistant passes `overwrite`. A network send goes into the other PC's no-PIN folder, which accepts it without asking, or to an `mcz receive` running there, which asks first unless it was started with `--yes`. While a long compress, extract, verify or send runs, the server sends the assistant progress updates (percent done and megabytes so far). Some assistants still give up on a call after a minute or two, so a compress, extract or verify that is still running after 50 seconds (the tool's `wait_seconds`) answers "still running" and carries on in the background, also if the assistant cancels the call; `mcz_activity` then shows its progress and, once it ends, its result. Other tools stop the running `mcz` when the assistant cancels the call.

The **MCP** tab shows whether an assistant is connected to the server right now: the client's name, since when, how many tool calls it has made, and a list of the recent calls with their result and duration.

3.8 Licence tab

See the licence state, buy a licence for this PC in a couple of minutes, or install a key.



The Licence tab.

The **Licence...** button in the status bar, the MCP tab's banner and any job refused for want of a licence all open this tab. From top to bottom:

- **Status**: the licence state (personal use, licensed to whom and until when, licence required, evaluation, expired) in red when MCZ is blocked, and the state of the premium features (1.3).
- **Machine ID** with a **Copy** button: the ID a key is issued for.
- **Buy a licence for this PC**: choose one of
  - **Commercial, annual**: for organisations, renews each year; preselected on a PC joined to a domain.
  - **Commercial, perpetual**: for organisations, pay once.
  - **Premium only**: for home use, unlocks MCZ Copy, the job queue and the MCP server after the trial; preselected on a personal PC.
  - **Free 30-day evaluation**: for organisations trying MCZ; no card, one per PC.

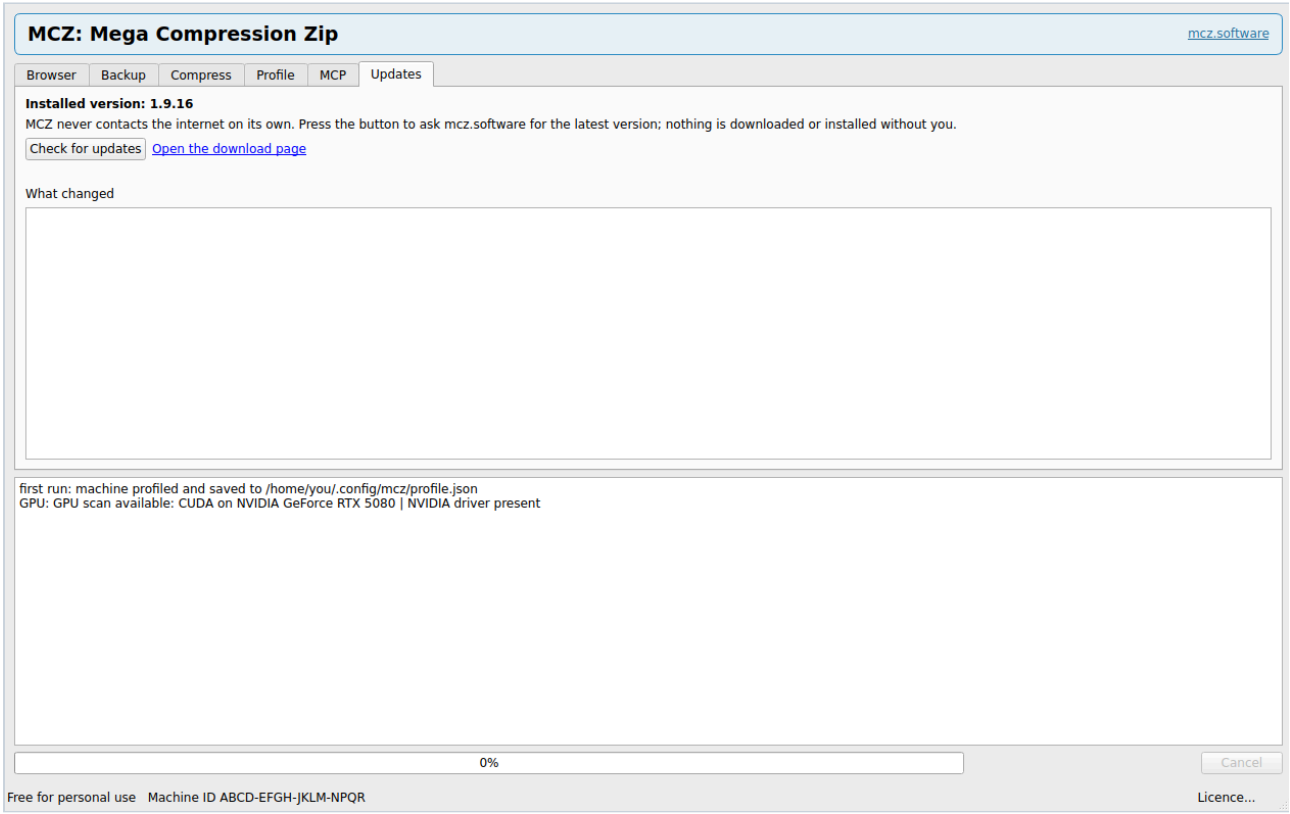
  Show prices asks mcz.software for this PC's price of each; **Order for many PCs online** opens mcz.software/buy.
- **Organisation** (printed in the key; may be left empty for a premium key), **Your name**, **E-mail** (for the key and the receipt) and **Currency** (£ GBP or $ USD). MCZ remembers them for next time.
- **I accept the licence terms**, then **Pay by card...** (or **Get evaluation key**).

- **Install a key**: paste a key you were sent and press **Install key**. A key you copied from the order page or the e-mail is filled in when you open the tab.

Pay by card... creates the order on mcz.software and opens Stripe's secure payment page in your browser (cards, Apple Pay and Google Pay). Keep the tab open while you pay: MCZ checks the order every few seconds and installs the key as soon as the payment completes, then shows *Thank you. Your key is installed*. The key is also e-mailed to you and shown on the order page (**Open the order page**). If you close MCZ or press **Stop waiting** before the payment completes, paste the key from the e-mail later. An evaluation key comes back and is installed at once. MCZ refuses to sell a second licence to a PC that already has one, and contacts mcz.software only when you press **Show prices**, **Pay by card...** or **Get evaluation key**.

3.9 Updates tab

Check for a newer version.



The Updates tab.

Shows the installed version. MCZ never contacts the internet on its own; **Check for updates** asks mcz.software for the latest version number and its release notes, and if a newer one exists shows a link to download it. The **Open the download page** link next to the button is always there, for looking yourself. Nothing is downloaded or installed by the program itself: you download the new installer from the website and run it, checking it as described in section 2.2.

3.10 The job queue (MCZ FIFO Server)

MCZ jobs take turns instead of slowing each other down.

Several MCZ jobs at once (two backups started together, an assistant asking for eight) do not finish sooner: they share the processor and the disks, and each runs slower. The **MCZ FIFO Server** makes them take turns. Every MCZ job (compress, add, extract, verify, copy, send and receive, and MCZ Copy's Get and Send, from the GUI, MCZ Copy, the command line, the network service or the MCP server) asks it first and waits for its turn. Jobs go first come, first served; a job that needs nothing a job ahead of it is using starts straight away. A job uses the processor (compressing, adding, extracting, verifying, receiving, MCZ Copy transfers), each disk it reads or writes (internal, USB or mechanical; two partitions of one disk count as one) and, for a network share, that server; sending, receiving and MCZ Copy transfers use the network. So a copy to a USB disk can run while another job compresses between two other disks, but two compressions never run together and two jobs never share a disk.

While a job waits, the status line of the GUI or MCZ Copy says *MCZ is busy* with its place in the queue and the job it is waiting for, and the command line prints `queued:` lines; **Cancel** still works. A transfer arriving from another machine (`mcz send`, or an MCZ Copy Get or Send) that is still waiting after nine minutes is turned down as busy, and the other machine is told why. MCZ Copy transfers take their turn on both PCs.

The job page. Open http://127.0.0.1:48631/ in a browser on the same PC. It shows the jobs running, with their progress, and the queue. **Promote** moves a waiting job to the front of the queue: it starts as soon as what it needs is free, and the jobs already running carry on. **Cancel** takes a job out of the queue, or stops a running one (its unfinished archive is removed, as with Cancel in the GUI; a cancelled MCZ Copy transfer shows *Cancelled* in MCZ Copy). The **History** lists every finished job by month, with how long it waited and ran, its size and the result; type in the search box to find a folder, an archive, a program or a result. The page is only reachable from this PC, and MCZ programs reach the queue itself at 127.0.0.1:48629, also on this PC only.

The record. Every request, start, promotion, cancellation and finish is written to a file per month (section 2.3), one line per event, which the History reads back. The files are only ever added to.

The job queue runs in the background as the systemd service `mcz-fifo`. Without it, or when it is stopped, every MCZ job starts at once as before. It is a premium feature (1.3): when the trial ends without a key, the jobs it holds are cancelled once, and from then on jobs run at once without queuing; the job page says so.

3.11 File manager and command-line verbs

The package registers the `application/x-mcz` MIME type, so `.mcz` files show the MCZ icon. Double-clicking an archive in GNOME Files, Dolphin or another file manager opens it **like a folder**: browse the folders inside, open any file with its usual program, or copy files out by dragging them. The archive is read-only, nothing can be changed or added there (use the GUI's Browser tab for that). Opening the archive reads only its list of files, so even a large archive opens at once; each file is unpacked and checked the first time you open it, which for a big file takes a moment. Unpacked files are kept in `~/.cache/mcz/mount` while the archive is open (up to 2 GB) and removed when it is closed.

In GNOME Files, right-click shows an **MCZ** submenu (it needs the `python3-nautilus` package, which is installed with MCZ unless you turned recommended packages off; close and reopen Files once after installing):

Right-click on	Entries
An archive (or several)	*Browse in Files*, *Open in MCZ* (the GUI's Browser tab), *Extract here*, *Extract to "name/"*, *Verify*, *Convert to Zip*, *Convert to 7z*, *Convert to tar.gz* (4.12)
A tarball (or several): `.tar`, `.tar.gz` / `.tgz`, `.tar.xz` / `.txz`, `.tar.bz2` / `.tbz2`	*Extract here*, *Extract to "name/"* (straight into the folder), *Convert to MCZ* (4.13), and the entries for files
A `.zip` or `.7z` file (or several)	*Convert to MCZ* (a `.mcz` of the same name beside each; 4.8, 4.9), *Compress with MCZ*, *Compress to Zip*, *Compress to 7z*, *Compress to tar.gz*
Files and folders	*Compress with MCZ*, *Compress to Zip*, *Compress to 7z*, *Compress to tar.gz* (a standard zip, 7z or tar.gz beside the first item; 4.8, 4.9, 4.13)
The background of a folder	*Compress this folder with MCZ*
The background inside an opened archive	*Close this archive (unmount)*

An opened archive stays open until you close it from that menu, run `mcz umount`, or log out. If the archive itself is changed while it is open (for example files are added to it), opening a file in it fails with "stale file handle" and it closes; double-click the archive again. To see which archives are open, run `mcz mount`. The GUI also appears in the applications menu under Utilities. Starting the GUI with a verb does a job without further clicks:

```
mcz-gui -c <file|folder>...      compress to <first item>.mcz next to it, start
immediately
mcz-gui -x <archive.mcz>         extract into a folder named after the archive, next to
it (a tarball too)
mcz-gui -xhere <archive.mcz>     extract into the archive's own folder (a tarball too)
mcz-gui -t <archive.mcz>         verify
mcz-gui -convert <file.zip|.7z|.tar.gz>  convert a zip, 7z or tarball to <name>.mcz next
to it (asks before replacing one)
mcz-gui -czip <file|folder>...   compress to a standard <first item>.zip next to it (-
c7z: a .7z, -ctgz: a .tar.gz)
mcz-gui -tozip <archive.mcz>     convert the archive to <name>.zip next to it (-to7z: a
.7z, -totgz: a .tar.gz, -totxz: a .tar.xz; warns first when it is signed)
mcz-gui -backup <target> <folder>...   open the Backup tab with the target and folders
filled in
mcz-gui -add <archive.mcz> <file|folder>...   add to the archive (same names are replaced
after asking)
mcz-gui <archive.mcz>            open the archive in the Browser tab
```

A file manager's "Open with" or a custom action can therefore run `mcz-gui -c %F` to get a "Compress with MCZ" entry.

4. The command line (`mcz`)

4.1 Commands at a glance

```
mcz c <archive.mcz> <file|dir>... [options]     compress
mcz a <archive.mcz> <file|dir>... [--to folder] add to an existing archive (same paths
are replaced)
mcz x <archive.mcz> <out-dir>                   extract everything
mcz x <archive.mcz> <out-dir> --files list.txt   extract only the entries listed (one per
line; "folder/" = everything in it)
mcz t <archive.mcz>                             verify (decode and check every hash,
write nothing)
mcz l <archive.mcz>                             list files
mcz l <archive.mcz> --match "*.txt"             only the paths that match a pattern
mcz zip <out.zip> <file|dir>... [options]       write a standard zip any computer opens
(4.8)
mcz 7z <out.7z> <file|dir>... [options]         write a standard 7z that 7-Zip opens
(4.9)
mcz tar <out.tar.gz> <file|dir>... [options]   write a tarball for Linux and macOS
(.tar.gz, .tar.xz, .tar.bz2 or .tar; 4.13)
mcz x <in.tar.gz> <out-dir>                     unpack a tarball, zip or 7z straight into
a folder (4.13)
mcz convert <in.zip|in.7z|in.tar.gz> [out.mcz]   convert a zip, 7z or tarball to .mcz
(4.8, 4.9, 4.13; also fromzip / from7z)
mcz convert <in.mcz> <out.zip|.7z|.tar.gz|.tar.xz> [-1 0-9]  convert an MCZ archive to a
standard file (4.12)
mcz c <program> <file|dir>... --sfx [options]    compress into a self-extracting program
(4.10)
mcz sfx <archive.mcz> [program]                  make a self-extracting program from an
archive (4.10)
mcz mount <archive.mcz> [folder] [--open]       show the archive as a read-only folder
(3.11); --open shows it in the file manager
mcz mount                                       list the archives shown as folders
mcz umount <folder|archive.mcz>                 close an archive shown as a folder
mcz profile [list|...]                          show or change profiles (4.4)
mcz license [key|file]                          licence state and Machine ID, or install
a key (licence or premium)
mcz premium                                     state of the premium features (trial days
left, or the key in use)
mcz gpu                                         which GPU backend would run
mcz version                                     print the version
mcz peers                                       list MCZ machines on the network
mcz send <machine|address> <folder|archive>      compress a folder (or take an archive)
and send it to that machine's no-PIN folder
mcz receive <folder> [--extract] [--yes]          accept archives from other machines
into the folder (Ctrl+C stops)
mcz copy-pin list | add <name> <PIN> <path[:rw]>... | items <id|name> <path[:rw]>... |
rename <id|name> <new> | remove <id|name>
```

```
                                                MCZ Copy PINs this PC shares (3.6)
mcz copy-connect <machine> <PIN>                 MCZ Copy: test a PIN against another PC
mcz copy-ls <machine> <PIN> [path]               MCZ Copy: list what a PIN shares on
another PC, or a folder in it
mcz copy-mkdir <machine> <PIN> <path>            MCZ Copy: make a folder in a read-write
folder
mcz copy-get [-n 1..4] [--overwrite] <machine> <PIN> <remote>... <local folder>    MCZ
Copy: fetch files and folders
mcz copy-send [-n 1..4] [--overwrite] <machine> <PIN> <local>... <remote folder>   MCZ
Copy: send into a read-write folder
mcz pin-check <PIN> | --generate                 MCZ Copy: check a PIN against the rules,
or make one
mcz share-check <folder>                         MCZ Copy: may this folder be shared, and
why not
mcz selftest                                     built-in tests
mcz gentest | cmp | bench                        test and benchmark tools (4.6)
```

Running `mcz` with no arguments prints a similar list; `mcz --version` (or `-v`) is the same as `mcz version`. Paths with spaces are quoted as usual for your shell. The MCZ Copy commands are described in full in the *MCZ Copy User Manual*; when `copy-get` or `copy-send` copies into a folder that already holds the files, identical files are skipped, older ones replaced and newer ones kept, and `--overwrite` replaces every existing file. `mcz receive -y` is short for `--yes` (accept every archive without asking). `mcz premium --check network|mcp|queue` prints `ok` and exits 0 when that premium feature may be used, or prints the reason and exits 2.

4.2 Compressing

```
mcz c /media/backup/Project.mcz ~/Projects/Project
mcz c out.mcz folderA folderB file.txt -l 7
mcz c out.mcz folder --no-profile -l 3 -t 8
```

Without options, `c` takes its settings from the active profile and the destination drive, printing a `profile: ...` line that says what it chose (and measures the drive on first use, as the GUI does). Any of `-l`, `-t`, `--avg`, `--pack-avg` or `--ultra` switches the profile off for that run and uses the built-in defaults for whatever is not given; `--no-profile` does the same without changing anything else.

Option	Meaning
`-l N`	Level 0 to 10. 0 stores without compressing, 1 is fastest, 9 small, 10 is *ultra*; 5 is the default and the usual choice.
`--ultra`	Same as `-l 10`: the smallest archive, typically 5 to 10 % smaller than the default, at several times the compression time. Extracting is a little slower too. Use it for archives you write once and keep.
`-t N`	Compression worker threads (default: all hardware threads minus the reader and writer threads).
`--avg KB`	16 to 256 (default 64); the same as *Average chunk* on the Profile tab. The default suits most data.
`--pack-avg KB`	Setting for folders of many small files (default 1024); `0` treats small files like big ones. The default suits most data.
`--prime KB`	0 to 2048 (default 256; larger values count as 2048). Larger values give smaller archives at more compression time; see section 6.2.
`--no-gpu`	No GPU at all: chunk-boundary scan on CPU threads, CPU codec only.
`--gpu-codec` / `--cpu-codec`	Also compress on the CUDA GPU alongside the CPU, or not. Archives made with the GPU are a little larger; it helps on PCs with few cores, a fast GPU and a fast destination.
`--gpu-only`	GPU codec with no CPU compression workers.
`--no-dedupe`	Store every chunk even when identical to one already in the archive.
`--resilient`	Make a **resilient archive**: keep a second copy of the archive's file list (its index), so damage to the index no longer loses every file, and code each piece of data on its own, so damage to the data costs only the files in the damaged piece rather than everything after it in the same 4 MB stretch. It costs about 5 % more space. Use it for backups and for archives kept on removable disks. Also on `a` (an archive that is already resilient stays so) and `convert`, and as the **Resilient archive** box on the Compress and Backup tabs.
`--no-x86`	Compress program files (`.exe`, `.dll` and the like) like any other file. By default MCZ treats them specially, which usually makes them smaller.
`--overwrite`	Replace an archive that already exists. Without it, `c` refuses and leaves the existing archive alone; with it, the old archive is removed only once the new one is complete.
`--fail-unreadable`	Stop with an error when a file cannot be opened. By default such a file (for example one Visual Studio keeps locked while a solution is open) is left out, listed at the end, and the exit code is 3.

The GPU and dedupe switches override the profile only when they are given; everything else comes from the profile unless it was switched off as described above. Progress is written to standard error and the result to standard output, for example:

```
compressed: 12345 files, 98765 chunks (1200 duplicate, 300 stored raw), 3657.0 MB ->
3471.2 MB in 18.24s, 31 threads
  backend: 4 readers, 31 CPU workers; scan: CPU per reader
summary: 12345 files in 812 folders, 3657.0 MB -> 3471.2 MB (94.9% of the original, 5.1%
saved) in 18.2 s (200 MB/s)
```

The `summary:` line gives the files and folders processed, the archive size as a percentage of the original and the saving, the time taken and the speed.

4.3 Extracting, verifying, listing, adding

```
mcz x /media/backup/Project.mcz ~/Restore/Project
mcz t /media/backup/Project.mcz
mcz l /media/backup/Project.mcz
```

`x` recreates the archived tree under the output directory (created if missing), restoring file times, and checks every checksum as it goes. Damaged data does not stop it: a file whose data fails its checksum (or cannot be read from the disk) is not written, every other file is extracted, and at the end each damaged file is listed on a `damaged:` line and the exit code is 1. Files that already exist in the output directory are never replaced: `x` reports them and stops before writing anything, unless `--overwrite` is given. `--files list.txt` extracts only the entries listed. `-t N` limits the decode threads. `t` verifies the same way without writing, reading the archive from the disk it was written to rather than from the operating system's cache, and lists every damaged file. A damaged index (the archive's file list) loses the whole archive, unless it is a resilient archive (4.2): then MCZ reads the second copy, says so in a `note:` line, and carries on as usual. `l` prints a header line (the archive's level and chunk sizes), then one line per file with its size, number of chunks, checksum and path. `l --match pattern` prints only the paths that match, one per line: `*` stands for any run of characters and `?` for one character, and letters match in either case. A pattern without a folder separator looks at file names in every folder (`--match "*.uasset"`); one with a separator matches the whole path (`--match "*/Maps/*.umap"`). The MCP server's `mcz_list` takes the same pattern as `match`.

Adding to an archive

```
mcz a /media/backup/Project.mcz ~/notes/readme.txt
mcz a /media/backup/Project.mcz ~/Project/Config --to Project
```

`a` adds files and folders to an archive that already exists, at the top or, with `--to`, inside that folder of the archive (created if it is not there). A file already in the archive under the same path is replaced without asking, and adding a folder replaces only the files it holds: other files already in that folder of the archive are kept. As on the Browser tab, the archive is replaced only once the new copy is complete and checked, so the drive needs room for a second copy while it runs, and a cancelled or failed `a` leaves the archive as it was. The compression options of `c` (and the active profile) apply to the new files; the exit codes are those of `c` (4.7).

4.4 Profiles

```
mcz profile [list]                  show machine, measured drives and all profiles (*
marks the active one)
mcz profile new "Backup to USB"     copy the active profile under a new name and select
it
mcz profile use "Backup to USB"     select a profile
mcz profile delete "Quick local"    remove a profile (never the last one)
mcz profile role backup|fast|share  set the active profile's role
mcz profile probe /media/backup     measure (or re-measure) a drive: give a folder on
it, or an archive path on it
mcz profile set level 7             fixed overrides: level, threads, pack-avg (0 = let
the role decide)
mcz profile set gpu-codec 1         switches: gpu-scan, gpu-codec, gpu-only, dedupe (0
or 1)
mcz profile set workspace ~/mcz-work    fast-drive folder where archives for slow drives
are built first ("-" or "none" clears it)
mcz profile reset                   re-probe the machine; forget drives and overrides
```

The GUI and the command line share this file, so a profile created in one is visible in the other. Section 5 explains what the roles choose.

4.5 Licence and GPU

```
mcz license                state, licence file in use, and this PC's Machine ID (XXXX-
XXXX-XXXX-XXXX)
mcz license MCZ1-....      verify and install a key given as text
mcz license key.txt        ... or from a file
mcz premium                premium features: days left in the free trial, or the key
that unlocks them
mcz gpu                    the GPU backend that would be used (CUDA or none) and why
```

4.6 Testing and benchmark commands

Command	Purpose
`mcz selftest`	Built-in tests of the whole program, including deliberately damaged input. Prints `SELFTEST PASSED` or the failures. Run it after installing on a new PC.
`mcz gentest <dir> <GB> [seed]`	Writes a mixed test corpus (text, binary, duplicates, empty and tiny files) of the given size.
`mcz cmp <dirA> <dirB>`	Byte-for-byte comparison of two trees; reports different, missing and extra files.
`mcz bench <work-dir> <file>... [-l 1,5,9] [--no-baseline]`	Compresses each file at the given levels, extracts and compares (the Windows-only speed baseline does not exist here, so `--no-baseline` is implied).

4.7 Exit codes

Exit code **0** means success; **2** means a licence is required (domain-joined PC without a key), or, for the network commands, that the free trial of the premium features has ended and a premium key is needed; **3** means the archive (or zip) was written or added to (or, for `mcz send`, sent) but some files could not be opened and were left out (they are listed, and also written to the system log), or that an MCZ Copy transfer finished with a warning; any other non-zero value means the operation failed and the reason was printed. The MCZ Copy commands (`copy-*`) return 2 when this PC's trial has ended without a key (except `copy-pin`, which needs no key), and 1 for every other failure, including a PIN the other PC refuses or a premium key the other PC lacks. Scripts should test the exit code rather than parse the progress text.

4.8 Zip files

MCZ writes standard `.zip` files for people who do not have MCZ, and turns zip files you receive into `.mcz` archives. Both are free, with or without a premium key.

```
mcz zip ~/share/Photos.zip ~/Pictures/Holiday
mcz fromzip ~/Downloads/Assets.zip
mcz fromzip ~/Downloads/Assets.zip /media/backup/Assets.mcz --overwrite
```

`zip` writes a zip that any file manager, macOS and `unzip` open. It uses Deflate on every core: each file is cut into 1 MB pieces that are compressed at the same time, then joined into one ordinary Deflate stream. A zip is larger than an `.mcz` of the same files and takes longer to make, so use it only when the other side has no MCZ. `-l 0..9` sets the Deflate level (default 6, 0 = store), `-t N` the threads; `--`

`overwrite` and `--fail-unreadable` work as for `c`. Files that do not get smaller (photos, videos, other archives) are stored as they are. Zips over 4 GB or with more than 65,535 files use Zip64, which current systems read. Names keep their accents and other scripts, and file times are kept.

`fromzip` converts a zip to `.mcz`, named after the zip and next to it unless you give a name. The zip is unpacked into a temporary folder beside the new archive, compressed with the active profile (or the options of `c`), and the temporary folder is deleted, also when the run fails or is cancelled. The drive therefore needs free space for the unpacked files; MCZ checks first. The zip itself is never changed. Encrypted (password-protected) zips and zips that use methods other than stored, Deflate and Deflate64 (such as BZip2 or LZMA) are refused with a message. Deflate64 is what 7-Zip use for some large files. A link stored in a zip made on Linux or macOS comes out as a small file holding the link's target, never as a real link, and so is a name inside the zip that would land outside the folder.

To unpack a zip without making a `.mcz`, run `mcz x Assets.zip <folder>`: the files go straight into the folder, and existing files are not replaced unless `--overwrite` is given (4.13 has the details, which are the same for zip, 7z and tarballs).

4.9 7z files

MCZ also writes standard `.7z` files for people who use 7-Zip, and turns 7z files you receive into `.mcz` archives. Both are free, with or without a premium key.

```
mcz 7z ~/share/Photos.7z ~/Pictures/Holiday
mcz convert ~/Downloads/Assets.7z
mcz convert ~/Downloads/Assets.7z /media/backup/Assets.mcz --overwrite
```

`7z` writes a 7z file that 7-Zip, NanaZip, p7zip, the Archive Manager and other 7z tools open, using every core. It is smaller than a zip of the same files but takes longer to make, and an `.mcz` is faster still, so use it when the other side uses 7-Zip. `-l 0..9` sets the level, the same scale as 7-Zip's (default 5, 0 = store, 9 = smallest and slowest; levels 7 to 9 need a lot of memory on many cores, and MCZ uses fewer threads when memory is short); `-t N` sets the threads; `--overwrite` and `--fail-unreadable` work as for `c`. Photos, videos and other archives are stored as they are. Names keep their accents and other scripts, and file times are kept. On the Compress tab, choose **7z (7-Zip compatible)** in the format list or end the output name with `.7z`.

`convert` turns a 7z (or a zip, 4.8) into a `.mcz` in the same way as `fromzip`: named after the source and next to it unless you give a name, unpacked into a temporary folder beside the new archive (the drive needs free space for the unpacked files; MCZ checks first), then compressed with the active profile, and the temporary folder is deleted. The 7z itself is never changed. 7z files made by 7-Zip with its usual settings convert, at any level, as do ones that use PPMd, Deflate, Deflate64 or the BCJ, BCJ2, ARM64 and Delta filters. Password-protected 7z files, and the rare ones that use other methods (such as BZip2), are refused with a message, and so is a name inside that would land outside the folder. Split (multi-volume) 7z files cannot be converted; join them first.

To unpack a 7z without making a `.mcz`, run `mcz x Assets.7z <folder>`: the files go straight into the folder, as for a zip (4.8).

4.10 Self-extracting archives

A self-extracting archive is an MCZ archive inside a small program: the person you send it to runs the program and the files come out, on every core and with every file checked, without installing MCZ. It is free, with or without a premium key.

```
mcz c ~/share/Photos.run ~/Pictures/Holiday --sfx
mcz sfx /media/backup/Project.mcz
mcz sfx /media/backup/Project.mcz ~/share/Project.run --overwrite
```

`c ... --sfx` compresses with the same settings and profile as an ordinary `c`, into a temporary `.mcz` beside the program that is deleted afterwards. `sfx` turns an existing archive into a program, named after it with `.run` unless you give a name; the archive is kept. The program is the archive plus about 1 MB, and `x`, `t` and `l` read it like an `.mcz`. Files cannot be added to a self-extracting program: add them to the `.mcz` and run `sfx` again. On the Compress tab, choose **Self-extracting** in the format list or end the output name with `.run`. `--stub <file>` uses another copy of the extract program, for example the Windows one (`mcz-sfx.exe` from the Windows installer) to make an `.exe` on Linux.

Running one. In a terminal, `./Photos.run` extracts beside the program (into a folder named after it, or straight there when everything in the archive is already in one folder). A copy downloaded with a browser or saved from mail loses the execute bit: run `chmod +x Photos.run` first. Switches:

```
-o <folder>   extract into this folder
-y            replace files that already exist (otherwise nothing is written when any
would be replaced)
-l            list the files
-t            test: decode and check every file, write nothing
-threads N    threads to use (default: all)
```

Exit codes: 0 done, 1 failed, 2 unknown switch. The program never connects to the network, needs no administrator rights and changes nothing outside the folder it extracts into.

4.11 Signed archives

A signed archive tells the person you send it to that it comes from you and that nothing in it changed on the way: not a file name, not a byte of data. Signing is free, and so is checking.

```
mcz key new "Pete Walton"
mcz key export
mcz c ~/share/Release.mcz ~/build/Release --sign
mcz sign ~/share/Older.mcz
mcz key trust MCZPK1-AQUYakkmm6kqlE7g8ChY7SO6dGLa4HV5nOZmqqfr_ig "Pete Walton"
mcz x ~/Downloads/Release.mcz ~/Release --require-signer "Pete Walton"
```

Your key. `key new` makes a key pair; `--passphrase` locks it with a passphrase you are asked for whenever you sign (scripts can set `MCZ_KEY_PASSPHRASE`). Without one the key is protected by file permissions (owner only). `key export` prints your public key, one line starting `MCZPK1-`, and its fingerprint (four groups of four characters). Send the public key to the people you send archives to, and read the fingerprint out to them another way (phone, in person) so they know the key is really yours. Back up the keys folder (2.3): a lost key cannot be made again, and there is no way to cancel one; if a key

is lost or stolen, make a new one and ask people to remove the old one from their trusted list. `key list` and `key delete` manage your keys.

Signing. `--sign` on `c`, `a` and `convert` signs with your only key; with several, `--key <name>` chooses one. `sign` signs an archive you already have, or signs it again (another key replaces the signature). Adding files to an archive you signed signs the new archive with the same key; an archive someone else signed loses that signature when you add to it, and MCZ says so. To send a signed self-extracting program, sign the `.mcz` first; the signature travels inside the program and is checked when it runs. Signing adds about 1 KB plus 32 bytes per MB of archive, and the time to read the archive once more.

Checking. `x` and `t` check a signed archive before writing anything, and every `l`, `x` and `t` prints a `signature:` line: *signed by Pete Walton (trusted)*, *signed by an unknown signer* with the key and the command to trust it, *not signed*, or *SIGNATURE INVALID* with the reason. An invalid signature stops the extraction; `--allow-invalid-signature` extracts anyway (the usual checksums still apply). Because anyone can remove a signature and send the archive unsigned, use `--require-signed` or `--require-signer <name|fingerprint|public key>` when it matters who made it (a name or fingerprint only matches a signer you trust; a full MCZPK1- public key matches that key). Your own keys count as trusted; `key trust`, `key untrust` and `key trusted` manage the others. A signature says who signed the archive, not that its contents are safe, and the signing time is the signer's own clock. Older MCZ versions open signed archives as usual; they just do not check the signature.

4.12 Converting MCZ archives to zip, 7z or tarballs

To send files to someone without MCZ, turn an archive into a standard zip, 7z, tar.gz or tar.xz. The `.mcz` is kept.

```
mcz convert ~/archives/Holiday.mcz ~/share/Holiday.zip
mcz convert ~/archives/Holiday.mcz ~/share/Holiday.7z -l 9 --overwrite
```

The new file's extension picks the format. The archive is extracted into a temporary folder beside the new file (the drive needs free space for the files; MCZ checks first), packed on every core as `mcz zip`, `mcz 7z` or `mcz tar` would (`-l` is the format's own level: zip 6, 7z 5, tar.gz and tar.xz 6 by default), and the temporary folder is deleted, also when the run fails or is cancelled. Self-extracting MCZ programs convert the same way. In the GUI, use **Convert MCZ to zip, 7z or tar...** on the Compress tab or *Convert to Zip* / *Convert to 7z* / *Convert to tar.gz* on an archive's right-click menu.

> **Signed archives lose their signature.** The signature is checked while the archive is extracted, and a changed archive is not converted. But zip, 7z and tar files have no place for an MCZ signature, so the new file is **not** signed: the people you send it to cannot check who made it or that it was not changed. The GUI asks before converting a signed archive, and `mcz convert` prints a warning when it is done. Send the `.mcz` itself when the signature matters.

4.13 Tarballs (.tar.gz, .tar.xz, .tar.bz2)

Tarballs are the usual archives of Linux and macOS. MCZ unpacks them, converts them to `.mcz` and writes them, free, with or without a premium key. It reads `.tar`, `.tar.gz` / `.tgz`, `.tar.xz` / `.txz` and `.tar.bz2` / `.tbz2` made by GNU tar, bsdtar (macOS and Windows' own `tar.exe`), 7-Zip or Python, including long names, accented names and files over 8 GB. A `.gz`, `.xz` or `.bz2` that holds one plain file instead of a tar unpacks to that file (`notes.txt.gz` gives `notes.txt`).

```
mcz x ~/Downloads/source-1.2.tar.gz ~/work/source-1.2
mcz convert ~/Downloads/source-1.2.tar.xz
mcz tar ~/share/Photos.tar.gz ~/Pictures/Holiday
mcz convert ~/archives/Holiday.mcz ~/share/Holiday.tar.xz
```

`x` on a tarball (or a zip or 7z) unpacks it straight into the folder, keeping file times and permissions. Files that already exist there are not replaced unless `--overwrite` is given; the GUI asks first. The *Extract here* and *Extract to folder* entries on a tarball's right-click menu do the same (3.11). A tar is one stream, so it unpacks on one core, and at the end MCZ checks the gzip, xz or bzip2 checksum, so a damaged download is reported. Hard links come out as copies. Symbolic links are made when they point inside the folder and left out otherwise, as are devices and other special files; the log lists what was left out. Names that would land outside the folder are refused, and so are sparse and multi-volume tars.

`convert` turns a tarball into a `.mcz` as for a zip (4.8): unpacked into a temporary folder beside the new archive, compressed with the active profile, and the temporary folder deleted. Links in the tar are not kept in the `.mcz`.

`tar` writes a tarball of files and folders; the extension picks the compression: `.tar.gz` (or `.tgz`, opened everywhere), `.tar.xz` (smaller, slower to make), `.tar.bz2` or a plain `.tar`. MCZ compresses the tar on every core, in pieces that tar, gzip, xz, bzip2, 7-Zip, macOS and Windows read as one file. `-l` sets the level (tar.gz 1..9, default 6; tar.xz 0..9, default 6; tar.bz2 1..9, default 9), `-t N` the threads; `--overwrite` and `--fail-unreadable` work as for `c`. Names keep their accents and other scripts, and file times and permissions are kept. Like a zip from MCZ, the tarball holds files and the folders they are in, but not empty folders or links. On the Compress tab, choose **tar.gz (Linux, macOS)** or **tar.xz (smaller)** or end the output name with `.tar.gz` or `.tar.xz`.

4.14 Damaged and resilient archives

Disks fail, USB sticks get pulled out mid-copy, and a backup is only as good as what comes back out of it. Every piece of an `.mcz` archive carries two checksums (CRC32 and XXH64), so damage is always found and never turns into a file that looks right but is not. What MCZ does about it:

- **Damage costs only the files it touches.** `x` and `t` do not stop at a damaged piece. Every file whose data checks out is extracted; a file that needs a damaged piece is not written (not even half of it), is listed on a `damaged:` line, and the run ends with exit code 1 so scripts notice. The GUI lists the damaged files in its error message. A part of the disk that cannot be read at all is read again in small pieces, so a bad sector costs only the files stored on it.
- **The file list is the weak point of an ordinary archive.** The index at the end of the archive says which file is where. MCZ keeps its location twice (header and footer), but there is one index: if it is damaged, no file can be found and the whole archive is lost.

A **resilient archive** closes that gap and keeps damage small:

- It keeps a **second copy of the index**, in a different disk sector from the first. If the index is damaged, MCZ reads the copy, says so in a `note:` line and carries on as usual. It finds the copy even when the header and footer are damaged as well.
- Every piece of data is **coded on its own**. An ordinary archive codes each piece against the data before it (priming), which makes it a little smaller but means one damaged piece also takes the pieces after it, up to 4 MB of the archive. In a resilient archive a bad byte costs only the files in that one piece.

- Small files are packed into **smaller shared pieces** (512 KB on average instead of 1 MB), so fewer of them share the fate of one bad byte.

Measured on a source-code tree of 605 files with one bad byte at 20 random places: an ordinary archive lost 6 files on average and 73 at worst, a resilient one 4 on average and 23 at worst. A damaged index lost all 605 files from the ordinary archive and none from the resilient one. The price was 4.8 % more space; it varies with the data.

Make one with `--resilient` on `c`, `a` or `convert`, the **Resilient archive** box on the Compress or Backup tab (one switch for both, remembered), or `resilient` on the MCP server's `mcz_compress`. Adding files to a resilient archive keeps it resilient, and so does signing it. Resilient archives open in every version of MCZ; older versions simply do not use the copy. Use them for backups, archives kept on USB disks or optical media, and anything you cannot make again. They do not replace a second backup: a resilient archive limits what damage costs, but it cannot rebuild data that was destroyed.

5. Profiles explained

The right settings depend on where the archive is going more than on the PC. Writing to a USB disk at 200 MB/s, a 32-core PC should spend its spare CPU on the strongest compression, because every megabyte not written is time saved; writing NVMe to NVMe it should use the lightest, because the CPU is then the slower side. A profile makes that decision automatically from two measurements:

- **The machine**, measured on first run (and by `mcz profile reset` or *Profile details*... in the GUI): CPU, hardware threads, RAM, GPU, and how fast this PC compresses.
- **Each destination drive**, measured the first time an archive is written to it (or by `mcz profile probe <path>`): sequential write speed, from 256 MB written next to the archive and deleted. The probe is skipped on drives with less than 1 GB free. Drives are recognised by device id, so a USB disk is recognised wherever it is mounted.

Role	Goal	What you get
Backup (default)	Smallest archive the CPU can afford; the destination is the limit	The strongest compression that does not slow the backup down; stronger still on big machines writing to slow drives, lighter when the drive is nearly as fast as the CPU.
Fast archive	Quick local round trips	A light level with small blocks.
Share	Others pull single files out	The default level with small blocks, so single files extract quickly.

PCs with eight or fewer threads keep one core free for the desktop. The profile's own overrides (`mcz profile set ...`) win over the role's choice, and explicit command-line options win over both. Every compression prints the decision it made, for example:

```
profile 'Default' (backup), destination F: WD_18TB (exFAT) writes 200 MB/s, CPU
compresses 1007 MB/s
  -> level 5, small-file blocks up to 4 MB
```

Several named profiles can exist and one is active, so a *Backup to USB* profile with the GPU codec off and a *Quick local* fast-archive profile can be switched between in one click or one command. The measurements are shared by all profiles; each profile holds its role, its engine switches and its overrides.

6. Optimising speed and size

6.1 What sets the pace

For most backups the drives set the pace, not the processor: the profile already picks the strongest level that does not slow the backup down on your drives (section 5). Reading from a fast SSD and writing to another fast drive is where compression settings make the most difference. A second run over the same files is often quicker than the first, because Linux keeps recently read files in memory.

6.2 Smaller archives for less time

To make archives smaller, try a larger `--prime` before a higher level. On build output (compiled programs, libraries and intermediate files) it gives most of the saving of level 9 in a fraction of the time. Measured on a laptop with an 8-core Ryzen 9 5900HX, compressing 6.1 GB of Unreal Engine build files already in memory:

Settings	Time	Archive size
`-l 3 --prime 512`	53 s	24.0 % of the original
`-l 5` (default)	49 s	22.9 %
`-l 5 --prime 1024`	97 s	22.4 %
`-l 9`	360 s	22.4 %

On a 32-core Threadripper 7975WX with 12.7 GB of `Intermediate` folders from two Unreal projects, `-l 5 --prime 1024` took 22 s for an archive 5.7 % smaller than the default, which took 11 s; `-l 9` took 98 s. On a whole Unreal project, where much of the content is already compressed, the saving was under 3 %. For the smallest archive of all, use `--ultra` (section 4.2), which already uses the largest `--prime` and takes several times longer again. Results depend on the data: files that are already compressed (textures, video, audio) gain little from any setting. Because the default suits backups best, `--prime` is a command-line option only; scripts can pass it to `mcz c` for archives where size matters more than time.

6.3 Power and cooling

At level 5 and above MCZ keeps every core busy, so compression time follows how fast the processor can run all its cores for minutes at a time. That depends on power settings and cooling, far more on a laptop than on a desktop:

- **Laptops on battery** run at a much lower power limit, and a long backup can take one and a half to two times as long. Plug in for large backups.
- **Laptop performance modes.** Many laptops have a maker's mode switch (for example Silent, Performance and Turbo) that sets how much power the processor may use. The fastest mode can cut the time of a long job by a fifth or more.
- **Heat.** After a few minutes at full load a laptop warms up and slows down; in our tests a laptop's later level 9 runs took 11 % longer than its first. A cool room, a clear air path or a cooling stand helps on long jobs.

- **Power plan.** Choosing the *Performance* power profile lets the processor speed up sooner. On a desktop the gain on large jobs is usually a few percent; it matters more for short jobs.
- **Sleep.** Make sure the PC will not go to sleep in the middle of a long backup: set the sleep timer longer than your longest backup.

7. Archive compatibility and safety

An `.mcz` archive written by any version of MCZ opens in every later version, on Windows and Linux alike; nothing in an archive depends on the PC, GPU or settings that produced it. Any single file can be extracted on its own without reading the rest of the archive, and extraction uses every core.

Everything in the archive is checked on extraction and by *Verify*, and a damaged archive is reported with the files it affects; the other files still extract. For archives that must survive bad disks, make them resilient (4.14). Data that cannot be made smaller (textures, video, already-compressed files) is stored as it is rather than made larger.

8. Backing up with scripts

Because `mcz` returns a meaningful exit code and takes its settings from the profile, a backup script is a loop of `mcz c` followed by `mcz t`:

```
#!/bin/bash
set -e
dest="/media/backup/$(date +%F)"
mkdir -p "$dest"
mcz c "$dest/Project.mcz" ~/Projects/Project
mcz t "$dest/Project.mcz"
```

With `set -e` the script stops at the first failure. Run it from cron or a systemd timer; `mcz` needs no display and prints progress to standard error. To archive only what changed, compare a stored checksum or modification time of the tree against the last run before calling `mcz c`; the PowerShell script `tools/Backup-UnrealProjects.ps1` in the source repository shows the pattern.

9. Troubleshooting

Where errors are reported. MCZ never shows pop-up error boxes. Every error appears in the log pane at the bottom of the window (or on the command line) and is also written to the system log, so it can be found later: `journalctl -t mcz` (the GUI, MCZ Copy, the command line and the network service all log their errors under that identifier; the network service's record of every connection is in `/etc/mcz/netservice.log` and `journalctl -u mcz-netsvc`).

Symptom	Cause and remedy
`mcz: command not found`	The tarball was unpacked somewhere other than `/`. Add its `bin` directory to PATH, or install the .deb, which puts both tools in `/usr/bin`.
`libMczCore.so` or a Qt library is missing	Install the .deb (it declares its dependencies) or `sudo apt install libqt6widgets6 libtbb12 libssl3`; for a tarball unpacked elsewhere, set `LD_LIBRARY_PATH` to its `lib` directory.
"The 30-day free trial of the network, job queue and MCP features has ended"	Buy a premium key on the Licence tab (3.8), which installs it for you, or at mcz.software/premium and install it with `mcz license <key>` (see 1.3). Compression and extraction keep working without one.
MCZ Copy: a PC is not found, a PIN is refused, Send is greyed out	See the troubleshooting chapter of the *MCZ Copy User Manual*.
"A commercial licence is required: send Machine ID ..." / exit code 2	The PC is joined to a domain. Buy a licence or get a free evaluation key on the GUI's Licence tab (3.8), or install a key with `mcz license <key>`; the message shows the Machine ID to quote when ordering one.
Compression is slow to start on a new drive	The first archive on a drive measures it by writing 256 MB. This happens once per drive; `mcz profile` lists the drives already measured.
Archive is larger than expected	The data was already compressed (textures, video, zips) and cannot shrink further; the log counts such data as *stored raw*. For other data check with `mcz profile` that the Backup role is in use.
Verify or extract reports damaged files	The archive is damaged (bad media, interrupted copy). Every other file was extracted; the damaged ones are listed and were not written. Re-copy the archive from its source if you have one. Verify backups on the destination disk with `mcz t` after writing them, and make important ones resilient (4.14).
"index CRC mismatch"	The archive's file list is damaged, so no file can be found. A resilient archive (4.14) reads its second copy instead; an ordinary one cannot be extracted. Re-copy it from its source.
GPU options have no effect	`mcz gpu` says which backend is available. The GPU codec needs an NVIDIA GPU with a current driver; on Linux only CUDA is available for the scan. On a 200 MB/s destination the GPU rarely matters because the disk is the limit.
Profile settings seem lost	If `profile.json` could not be read it was moved to `profile.json.bad` and a new one made; compare the two and copy anything wanted back, or recreate profiles with `mcz profile new`.
The GUI does not start	Run `mcz-gui` from a terminal to see the message. Over SSH or in a container set `QT_QPA_PLATFORM=offscreen` only for testing; a desktop session is needed for normal use.

MCZ 1.9.18 for Linux, 16 November 2026. Source and releases: github.com/peterbobwalton/Mcz. Free for personal use; see `LICENSE.txt`.