AutoClaw on Fedora: technical manual

Installation, architecture, operation, maintenance and a balance of benefits and costs, with the state verified between 20 and 21 September 2026 on an ASUS ROG Strix G513QY running Fedora 44

José Mauricio Gómez Julián

21 September 2026

Contents

0. How to read this manual **4**

1. Executive summary **5**

1.1 What AutoClaw and OpenClaw are 5

1.2 What was obtained on Fedora 5

1.3 Verdict 6

2. Balance of benefits and costs of installing it on Fedora **6**

3. Architecture **8**

3.1 The decision everything else follows from 8

3.2 Pieces, paths and ports 9

3.3 Why the port fence is needed 10

3.4 Startup, step by step 10

3.5 The configuration translation and the own configuration file 10

3.6 The user units 11

3.7 Security invariants of the design 12

4. Installation step by step **12**

Step 0. Measure the starting point without touching anything 13

Step 1. Host packages 13

Step 2. Obtain and verify the 1.17.8 installer 13

Step 3. Install AutoClaw under Wine 14

Step 4. Copy the integration from the portable bundle 14

Step 5. Python environment and LibreOffice 14

Step 6. First opening through the original path, to sign in to Z.ai 15

Step 7. Repair the four defects specific to Fedora 15

Step 8. First full startup and acceptance . . . 16
Step 9. Silence the updater . . . 16
Step 10. Own providers, defaults and MiMo in the window . . . 16
Step 11. The document extractor patch . . . 17
Step 12. Voice of the replies (optional) . . . 17
Step 13. Dictation (optional) . . . 17

5. The three slownesses **18**
5.1 The graphics translation chain: the window draws in software . . . 18
5.2 The slowness on opening: the retry loop . . . 19
5.3 The slowness of dictation: an engine that invented text . . . 19

6. Providers and models **20**
6.1 Z.ai, the originating provider . . . 20
6.2 MiMo in the window: a custom model . . . 20
6.3 Images and documents: the agent's default models . . . 21
6.4 The MiMo token plan contract . . . 21
6.5 GitHub Copilot . . . 22
6.6 AutoClaw's credits . . . 22

7. The document tool **22**
7.1 How it treats a document . . . 22
7.2 Three cuts that said nothing . . . 22
7.3 `pdfMaxPages` at sixty . . . 23
7.4 The patch that makes the cuts visible . . . 23
7.5 The limit as a per-call budget: `parsePageRange`, repaired . . . 24

8. Who rewrites the configuration **24**
8.1 Why it had to be measured . . . 24
8.2 The established mechanism . . . 24
8.3 Blind adjudication . . . 25
8.4 Consequences for operation . . . 25

9. Voice of the replies and dictation **25**
9.1 Why pieces outside the window were needed . . . 25
9.2 The voice of the replies . . . 26
9.3 Dictation: design . . . 26
9.4 Engine and calibration . . . 26
9.5 The voice gate . . . 27
9.6 Privacy . . . 28

9.7 Cost in graphics memory . . . 28
9.8 Thermal cost and the abrupt shutdowns . . . 28

10. The working regime inside the agent **29**
10.1 The problem it solves . . . 29
10.2 The path used and its scope . . . 29
10.3 The pieces . . . 30
10.4 How it was verified . . . 30
10.5 Costs and state . . . 30

11. Updates **30**
11.1 The updater under Wine and its error dialogs . . . 30
11.2 The manual channel . . . 31
11.3 Version 1.18.1, audited and not migrated . . . 31
11.4 What an update erases and what it keeps . . . 31
11.5 Migration procedure . . . 32

12. Daily operation and traps **32**
12.1 Opening, closing and exiting . . . 32
12.2 Health checks . . . 32
12.3 Restarting the gateway . . . 32
12.4 Models, credits and configuration . . . 33
12.5 Documents . . . 33
12.6 Voice and dictation . . . 33
12.7 The gateway’s console . . . 34
12.8 Measured traps, with their lesson . . . 34

13. Open decisions and pending structural fixes **36**

14. State verified at the time of writing, paths and digests **38**
14.1 The oracles of 2026-09-14 . . . 38
14.2 What changed with respect to the logs . . . 39
14.3 Paths and digests . . . 39
14.4 What was applied on 2026-09-20 and the state it left . . . 41
14.5 The second day’s work of 2026-09-20: page integration and audio output . . . 41

15. Glossary **43**

16. Sources and earlier documents **46**

16.1 The five files of the desktop folder 46

16.2 Other sources consumed 46

16.3 Discrepancies between sources 46

16.4 In plain terms 47

0. How to read this manual

This manual gathers into a single document what was done to make AutoClaw work on Fedora 44 on an ASUS ROG Strix G513QY, whose host name is `TheCommieMachine`, between 9 and 13 September 2026, starting from the integration that had been built and validated between 2 and 4 September on an ASUS VivoBook running the same system. Since its reader is whoever has to decide whether to install AutoClaw on Fedora and, once installed, to operate and maintain it, the order follows that need: first the balance of what is gained and what is paid (chapters 1 and 2); then the architecture and the installation recipe with an oracle for each step (chapters 3 and 4); then the chapters where each mechanism is explained with its measurements (5 to 11); and finally daily operation, what remains open, the verified state with the digests of everything installed, the glossary and the sources (12 to 16). The chronology appears only where an incident teaches something reproducible, that is, as a trap with its lesson and its measured cost.

An **oracle** is a check whose result decides, without needing interpretation, whether a step came out right, and a **digest** is the SHA-256 sum of a file, which allows verifying that it is byte for byte the one described. Every material claim carries one of three states. **Measured** means it was measured on this machine, on the date given. **Cited** means it comes from a primary source that is not a measurement of our own, whether a vendor's documentation or the code of a package read directly. **Not established** means the available evidence is insufficient, and in that same place the manual says what evidence would establish it. Times are local, six hours behind coordinated universal time, unless stated otherwise. Figures for time, memory and temperature belong to this machine, with an AMD Ryzen 9 5980HX processor of 8 physical and 16 logical cores, 30 GiB of memory, a dedicated AMD Radeon RX 6800M graphics card with 12 GB of its own memory and a Radeon Vega 8 integrated graphics unit, and they do not transfer to another machine without being measured there.

Sources are cited by the short name given in table 1, together with their full path. The logs remain the laboratory notebook and keep the detail of every measurement; the manual cites them rather than copying them, and chapter 16 says which of the earlier documents has been superseded and why. Paths beginning with ~ refer to the home directory of the user `usuario`, and the memories cited live in the Claude Code memory directory, `~/.claude/projects/-home-usuario/memory/`.

Table 1

Cited sources, with their short name, path and contents

Short name	Path	Contents
installation log	`~/claude-setup/autoclaw/REGISTRO_INSTALACION_2026-09-09.md`	laboratory notebook of the installation on the ROG Strix, from 2026-09-09 to 2026-09-11, covering sessions 01 to 09
compendium	`~/claude-setup/autoclaw/legacy-escritorio-2026-09-21/COMPENDIO_AUTOCLAW_ROG_STRIX_PARA_VIVOBOOK_2026-09-10.txt`	consolidation as of 2026-09-10 at 19:04, with a reproduction recipe and appendices transcribing the scripts
replication report	`~/claude-setup/autoclaw/legacy-escritorio-2026-09-21/Informe_Replicacion_AutoClaw_OpenClaw_2026-09-04.md`	the reference installation on the VivoBook, with the root diagnosis and the architecture
portable bundle	`~/Escritorio/AutoClaw/AutoClaw_Fedora_andamiaje_portable_2026-09-21.tar.gz`	the integration's own files, with their sums
operating README	`~/.local/share/autoclaw-linux/README.md`	living documentation of the integration
regime log	`~/claude-setup/autoclaw/REGISTRO_REGIMEN_AUTOCLAW.md`	the Be Like GODmez regime inside the agent, sessions 01 to 14

Short name	Path	Contents
thermal pre-registration	`~/diagnostico_congelamiento_20260913/PREREGISTRO.md`	the 2026-09-13 test with continuous transcription on the dedicated card, with its result
thermal summary	`~/diagnostico_congelamiento_20260913/corrida/resumen.json`	maxima, controls and verdict of that run
list of open items	`~/claude-setup/autoclaw/PROMPT_SESION_10_AUTOCLAW.md`	what remained pending for dictation and voice as of 2026-09-11
shutdown memory	`setup_ntfs3_volumen_sucio_tras_apagado_forzado.md`, in the memory directory	the reboot reason recorded by the hardware and the repair of the NTFS volume
Whisper memory	`feedback_whisper_turbo_inventa_sobre_silencio.md`, in the memory directory	why the engine invents text over silence
KWin memory	`setup_kglobalaccel_por_gdbus_tumba_kwin.md`, in the memory directory	the compositor crash caused by a call to the session bus
dictation journal	`journalctl --user -u dictado.service`	phases, levels and device for each dictation
measurement of 14 Sept.	chapter 14 of this manual	the state oracles run on 2026-09-14 from 08:41
units of 20 Sept.	chapter 14.4 of this manual	what was applied on 2026-09-20 and the state it left behind

1. Executive summary

1.1 What AutoClaw and OpenClaw are

OpenClaw is an artificial intelligence agent whose core is a **gateway**: a server written for Node.js that maintains conversation sessions, calls the language models of the configured providers, runs the agent's tools (file reading and writing, browser, scheduled tasks, documents) and serves its clients over a WebSocket, a bidirectional and persistent channel over a network connection, in this case a local one. AutoClaw is the desktop client that Z.ai, the company also known as Zhipu, built with Electron 33.4.11 on top of a heavily patched fork of OpenClaw 2026.6.8; its gateway is a single 48 MB file, `gateway-bundle.mjs`, with some seventy runtime patches applied on top (replication report, chapter "Object, root diagnosis and architectural decision"; cited from the package).

OpenClaw has an official Linux distribution, whereas AutoClaw exists only for Windows and macOS. Using OpenClaw on its own is not enough, because Z.ai's credits and models arrive through an intermediary proper to AutoClaw, with credentials that only its interface renews, once per hour. From that follows the shape of the whole integration: AutoClaw's interface is kept and, on Linux, can only run under Wine, the layer that executes Windows programs by translating their system calls; the gateway, by contrast, comes out of Wine and runs natively, for the reasons given in chapter 5.2.

1.2 What was obtained on Fedora

On Fedora 44 with KDE Plasma in a Wayland session, AutoClaw 1.17.8 ended up running under Wine 11.0 Staging, with the OpenClaw gateway running natively on Linux on the official Node 24.18.0 binary and with the interface connected to it as an "external gateway". On that base the following were verified working (measured 2026-09-09; compendium, sections 2 and 3.3): OpenClaw's internal diagnostics, with 22 checks and no errors; web search through the interface's bridge, with real results; OpenClaw's native browser on Chromium; scheduled tasks over SQLite; an in-house Office bridge on LibreOffice, implementing 28 tools of AutoClaw's Office contract and passing 80 of 80 checks with the original documents intact; and the office document skills, served by a dedicated Python environment.

On top of that base the following were added, in the order a third party would need them: silencing the updater, which kept looking for a version that cannot be installed under Wine and opened error dialogs; Xiaomi's MiMo models, contracted through the provider's token plan, on the agent's three paths (conversation, images and documents), so that the last inference request to Z.ai in the gateway trace is the one at 13:16:25 on 2026-09-10 and the 49 that followed, up to 2026-09-14, all went to MiMo; a local patch to the document extractor that warns the model when a document arrived truncated; reading replies aloud, with a Río de la Plata Spanish online voice from Microsoft and a player of our own, because the window

discards the audio; a push-to-talk desktop dictation system, with local transcription on the dedicated graphics card and a gate that detects whether the recording contains speech; and the Be Like GODmez working regime loaded inside the agent. On 2026-09-13, three dictations with real speech resolved in 1.57, 1.65 and 1.70 seconds from the second keypress of the shortcut to the pasted text (measured; dictation journal; chapter 9).

1.3 Verdict

Installing AutoClaw on Fedora is worthwhile, and the integration described leaves a functionally complete agent for desktop use, provided four conditions are accepted which are the real price of operating it. The first is manual maintenance: every AutoClaw update rewrites its resources folder and with it erases the updater silencing, the Linux native modules and the extractor patch, so updating is a task with a procedure and a validation battery, not a click (chapter 11). The second is giving up the capabilities that depend on Windows binaries and are lost or substituted: optical character recognition, the side panel browser, the legal runtime, collaboration between Office panels, and automatic start with the session (table 2, row C5). The third is living with a window that draws in software, with 1.5 % of a core at rest and up to 130 % in the worst forced redraw case, because it is AutoClaw, not Wine, that disables graphics acceleration (chapter 5.1). The fourth concerns privacy and contract: everything the agent reads travels to the model provider, the text of every spoken reply travels to Microsoft to be synthesised, and the MiMo plan forbids using it in automated scripts (chapters 6 and 9).

It is not worthwhile, on the other hand, for anyone expecting unattended updates, nor for anyone who needs precisely the lost capabilities. Dictation is an addition separate from AutoClaw and has its own balance: it works and it does not take your voice off the machine, but it occupies some 4.9 GB of the dedicated card's memory while running and, under continuous transcription, drives that card's junction to its critical temperature of 100 °C in 20.2 seconds at 161 W (chapter 9.8). Furthermore, on 2026-09-13 the machine suffered an abrupt shutdown whose reason, recorded by the hardware itself, is an uncorrectable error that caused a bus flood, the same reason recorded by two shutdowns on 2026-06-29 and one on 2026-07-21, both prior to the installation of dictation and of AutoClaw; that load on the dedicated card causes it is a candidate hypothesis and **not established**. AutoClaw and video playback, measured in the thermal test of that same day, do not load the dedicated card.

2. Balance of benefits and costs of installing it on Fedora

Rows beginning with V are benefits and those beginning with C are costs. The evidence column refers to the short names of table 1 and to the section of each source; the state column distinguishes what was measured on this machine, what is cited from a primary source, and what was not established.

Table 2

Balance of benefits and costs of installing AutoClaw on Fedora, with evidence and state

Aspect	Benefit or cost, with its magnitude	Evidence	State
V1. Native gateway	answers `/health` with a median of 3.5 s and sustains scheduled tasks, device pairing and agent memory; the embedded gateway under Wine took some 70 s, retried in a loop and broke SQLite with `EBUSY`	compendium 6.3; replication report, root diagnosis	measured: ROG Strix 2026-09-10, VivoBook 2026-09-02
V2. Verified capabilities	internal diagnostics with 22 checks and 0 errors; Office 80 of 80 with originals intact; real web search; native browser; tasks over SQLite; idle shutdown	compendium 3.3	measured, 2026-09-09
V3. Inference cost	conversation, images and documents all resolve to MiMo; after 13:16:25 on 2026-09-10, 49 inference requests in the trace and none to Z.ai; 32 configuration rewrites without the defaults being lost	compendium 2 and 6.1; installation log, session 09; measurement of 14 Sept.	measured, 2026-09-10 to 2026-09-14
V4. Long documents	the patched extractor prepends a notice when a cut leaves pages out and states which range reads the rest; 22 cases with text and images byte-identical to the engine's	compendium 7.3; operating README	measured, 2026-09-10

Aspect	Benefit or cost, with its magnitude	Evidence	State
V5. Speaking to the agent	local dictation, without your voice leaving the machine; 1.57 to 1.70 s from the second keypress to the pasted text, of which transcription takes 1.20 to 1.37 s	dictation journal, 2026-09-13 from 20:00 to 20:01	measured
V6. Hearing the replies	Microsoft online voice, free and without a key, played by a player of our own	installation log, "Voice of the replies"	measured, 2026-09-11
C1. Maintenance on update	each version rewrites `resources/` (2.0 GB) and erases the updater channel, the native modules and the patch; 1.18.1 was audited and not migrated	operating README; compendium 6.5	measured and cited
C2. Defects specific to Fedora	four: Fedora's `nodejs24` crashes in `Intl.Segmenter`; a VS Code terminal inherits `ELECTRON_RUN_AS_NODE`; without `XAUTHORITY` there is no window; Calc's formatting depended on the locale	compendium 3.2	measured, 2026-09-09 and 2026-09-10
C3. Software rendering	1.5 % of a core with the window idle and 130.2 % in the worst forced case; no flag reverses it	installation log, session 01	measured, 2026-09-10
C4. Cold start	window at 10.1 s with the gateway down and at 5.4 s with the gateway alive; keeping it alive costs 466 MiB freshly started and some 655 MiB after a day's work	operating README; compendium 6.3	measured, 2026-09-10
C5. Lost capabilities	optical character recognition, side panel browser, legal runtime, collaboration between Office panels, automatic start, browser extension	operating README, "Known limitations"; compendium 3.4	cited from the code; measured in the interface log
C6. Local surface	gateway without authentication, limited to the local loopback; Z.ai and MiMo keys in plain text inside `openclaw.json`, with owner-only permissions	operating README, "Invariants"; compendium 3.3 and 5.1	measured; whether it is exploitable is not established
C7. Privacy	what the agent reads travels to the model provider; the text of every spoken reply travels to Microsoft	operating README; compendium 13.8	cited from the design
C8. MiMo plan contract	only in programming tools; forbidden in automated scripts; the plan expires on 2026-10-10	compendium 4.4, citing the provider's page	cited
C9. Dictation graphics memory	some 4.9 GB of 12 GB while the service runs; 4.27 GB occupied across the whole card on 2026-09-14 with the service loaded	installation log, dictation; measurement of 14 Sept.	measured
C10. Dictation temperature	under continuous transcription the junction reaches 100 °C in 20.2 s at 161 W; a single dictation with the card cold reaches 91 °C	thermal pre-registration; thermal summary	measured, 2026-09-13
C11. Abrupt shutdowns	reason recorded by the hardware: uncorrectable error with bus flood, on 2026-06-29 (twice), 2026-07-21 and 2026-09-13; relation to card load not established	kernel journal; shutdown memory	reason measured; cause not established
C12. Automated testing	under Wayland it is impossible to type or click programmatically inside the window	operating README	measured, 2026-09-10
C13. Procedural risk	a session bus call to the global shortcuts service aborted the compositor and killed AutoClaw and Firefox	installation log, "Serious error of our own"; KWin memory	measured, 2026-09-11
C14. Copilot	authenticated, but no model answers (`model_not_supported`)	compendium 4.3	measured, 2026-09-10

Note. The weightier rows are developed below. The magnitudes of V1, C3 and C4 belong to this machine; on one with fewer cores or less memory they change, and the sign of the balance may change with them.

On the native gateway and the cost of inference (V1 and V3). The reason for taking the gateway out of Wine is not convenience. Under Wine, the gateway AutoClaw starts with its own `node.exe` took some seventy seconds to answer its health endpoint, while AutoClaw's internal deadlines declared it dead at fifty, thirty and fifty-three seconds and relaunched it in a loop; and when it did come up, Wine failed the permission change on OpenClaw's SQLite database, which broke scheduled tasks, the task log, device pairing and agent memory. The same gateway, on native Node, comes up in three and a half seconds and loses none of those functions. As for cost, conversation resolved to MiMo from the moment the model was registered in the window, but images and documents kept spending Z.ai credits until their default models were fixed; since then the gateway trace, which is the only consumption oracle that does not confuse account queries with inference, has recorded no inference request to Z.ai.

On maintenance (C1). AutoClaw's installer rewrites the entire `C:\Program Files\AutoClaw\resources` folder, and three pieces of the integration live there: the channel file that silences the updater, the Linux native modules the gateway needs, and the patched extractor. The gateway's startup script restores the native modules on its own from a backup copy, provided their versions do not change; the channel and the patch must be restored by hand, and the patch applier refuses to write if the update changed the extractor, in which case the patch has to be derived again. The audit of 1.18.1 showed that this version expects the same six native module versions and declares the same OpenClaw version, so migrating would cost restoring and revalidating, but not obtaining new modules (chapter 11).

On the defects specific to Fedora (C2). Two of the four defects belong to the host and two to the environment from which the application is launched, and all four have in common that they fail silently or in the wrong place. Fedora 44's `nodejs24` dies with a segmentation fault when segmenting text, and a Node program that dies that way leaves no trace pointing to the cause; the official nodejs.org binary of the same version does not have the defect. A terminal born inside VS Code turns any Electron application into a Node interpreter that exits without opening a window. Without the `XAUTHORITY` variable, the interface starts with every service healthy and no window at all. And with the Spanish locale, LibreOffice read the point in `0.00` as a thousands separator. All were repaired at their root and each has its oracle in chapter 4.

On the local surface and privacy (C6 and C7). The gateway listens without authentication because AutoClaw's interface, in external gateway mode, does not negotiate credentials with it; the exception is valid only while the binding stays limited to the local loopback, which is forced both in the configuration and on the command line, and it means that any process of the same user, or of another user of the same machine, can talk to the gateway. Whether that is exploitable on a personal single-user machine was not evaluated, and a review of which commands the gateway accepts without a session would establish it. The plain-text keys are written by the interface itself, which stores Z.ai's and the custom model's that way, with owner-only permissions. On privacy, the general rule is that everything the agent reads, including the regime files authorised to it, is sent to the model provider as part of the context, and that the voice of the replies is synthesised by a Microsoft service to which the text travels; the dictation voice, by contrast, is transcribed on the machine and does not leave it.

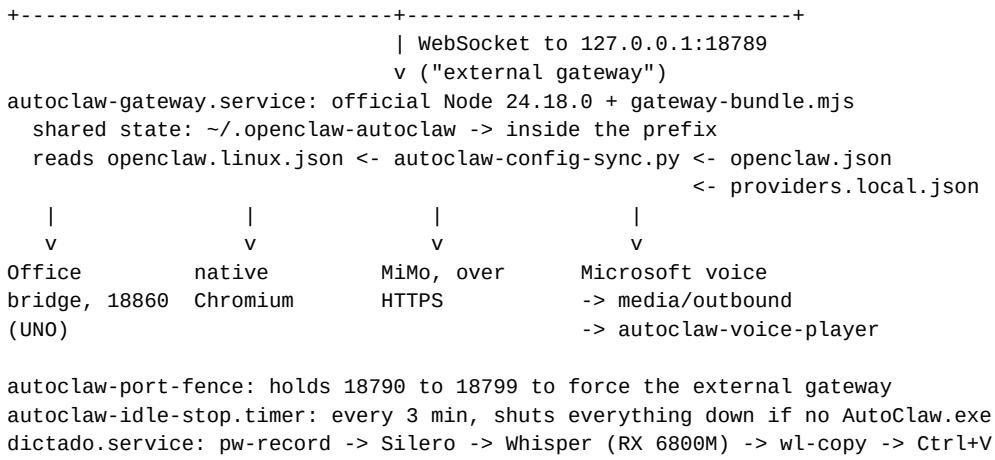
On graphics memory, temperature and shutdowns (C9, C10 and C11). All three costs belong to dictation and to the dedicated card, not to AutoClaw, and they are separated this way because their evidence has different strength. The graphics memory occupied is a measured fact and so is its practical consequence: a game or a computation needing the whole card requires stopping the service first. The temperature is a fact measured under a pre-registration: an isolated transcription with the card cold reaches 91 °C and continuous transcription reaches the critical 100 °C in twenty seconds, but real use, with pauses between dictations, lies between those two extremes and was not measured. The shutdowns are a fact measured in their reason and not in their cause: the hardware recorded the same bus flood on dates earlier than dictation, thermal tripping has its own codes which appear in no boot since 2026-06-20, and the system journal loses its last seconds before a cut, so today there is no evidence tying the 2026-09-13 shutdown to transcription. Chapter 9.8 states what measurement would establish it.

3. Architecture

3.1 The decision everything else follows from

The integration answers two facts which, taken together, leave a single way out. The first is that AutoClaw's interface cannot leave Wine, because it is a Windows program and is the only thing that renews Z.ai's credentials. The second is that the gateway that interface launches under Wine does not work, because of the retry loop and the SQLite failure described in chapter 5.2. The way out is to keep the interface under Wine and run the gateway on Linux's native Node from the very same `gateway-bundle.mjs` that AutoClaw ships, sharing the state directory with the interface, and to get the interface to adopt that gateway as an "external gateway" without trying to launch its own. Everything else (the configuration translation, the port fence, the launcher, the idle shutdown, the native modules, the Office bridge, the native browser and the Python environment) follows from that decision (replication report, "Object, root diagnosis and architectural decision").

```
Wine, prefix ~/.wine (X11 window served by XWayland)
+-----------------------------------------------------------+
| AutoClaw.exe 1.17.8: Electron interface, software rendering  |
| token server 127.0.0.1:18432 (web search, traces)            |
```



3.2 Pieces, paths and ports

Table 3

Pieces of the integration, with their path and their function

Piece	Path	Function
Launcher	`~/.local/share/autoclaw-linux/bin/autoclaw-launch.sh`	cleans the environment, starts the units, waits for `/health` and only then opens the interface under Wine
Gateway startup	`bin/autoclaw-gateway-run.sh`	restores the native modules, synchronises the configuration, exports the environment and runs Node against `gateway-bundle.mjs` on 127.0.0.1:18789
Restart without disconnection	`bin/autoclaw-gateway-restart.sh`	kills the gateway with SIGKILL so systemd relaunches it and the window reconnects on its own
Synchroniser	`bin/autoclaw-config-sync.py`	translates `openclaw.json` into `openclaw.linux.json` and adds `providers.local.json` to it
Own configuration file	`~/.local/share/autoclaw-linux/providers.local.json`	own providers, agent defaults and reply voice, out of the interface's reach
Port fence	`bin/autoclaw-port-fence.py`	holds ports 18790 to 18799
Idle shutdown	`bin/autoclaw-idle-stop.sh`	every three minutes, if no `AutoClaw.exe` process exists, stops the units
Office bridge	`office-bridge/autoclaw_office_bridge/`	HTTP server on 127.0.0.1:18860 over LibreOffice's UNO interface, with 28 tools
Gateway plugin	`extensions/libreoffice-bridge/`	fork of AutoClaw's Office plugin that talks to the bridge
LibreOffice configuration	`bin/autoclaw-libreoffice-setup.py`	adds `ooSetupConnectionURL` to the LibreOffice profile
Native modules	`native-stage/node_modules/`	backup copy of six Linux binary modules
Skills environment	`venv/`	forty frozen packages for docx, xlsx, pptx and PDF
Extractor patch	`patches/document-extractor/` and `bin/autoclaw-pdf-notice-patch.py`	applies, checks and removes the document truncation notice
Voice player	`bin/autoclaw-voice-player.py`	plays each new voice file the gateway leaves in `media/outbound/`
Provider scripts	`bin/autoclaw-set-provider-key.sh` and `bin/autoclaw-provider-login.sh`	load a key without displaying it; authenticate a provider with the gateway's environment
Official Node	`~/.local/opt/node-v24.18.0-linux-x64/bin/node`	the gateway's binary
Updater channel	`~/.wine/drive_c/Program Files/AutoClaw/resources/channel.json`	sets the `wine-manual` channel, which silences the updater
State symlink	`~/.openclaw-autoclaw`	symbolic link to `~/.wine/drive_c/users/usuario/.openclaw-autoclaw`

Piece	Path	Function
Dictation	`~/.local/share/dictado/dictado.py` and `~/.config/dictado/config.json`	resident server with the model loaded, and a client over a local socket
Regime inside the agent	`~/claude-setup/godmez/autoclaw/godmez_autoclaw_apply.py`	restores the instruction block, the skill and the read authorisation

Note. Paths beginning with `bin/`, `office-bridge/`, `extensions/`, `native-stage/`, `venv/` and `patches/` hang from `~/.local/share/autoclaw-linux/`. The script `bin/autoclaw-post-reboot-check.sh` comes in the portable bundle but its unit was not installed, because it verifies the VivoBook's memory configuration and on the ROG Strix it would fail by design.

All ports listen on the local loopback only, which is the machine's internal network interface and is not reachable from another machine: 18789 is the gateway; 18790 to 18799, the fence; 18860, the Office bridge; and 18432, 19654, 19723 and 53699 are the interface's "token server", whose primary, 18432, acts as a bridge for web search and for the model traces that the startup script points the gateway at.

3.3 Why the port fence is needed

The logic was read in the interface's `app.asar` package (replication report, "Why the port fence"; cited from the code). When the interface starts and finds port 18789 occupied by something that answers `/health`, it tries to kill the owning process, which is impossible from Wine against a Linux process, and then looks for a free port between 18790 and 18799 to launch its own gateway there. Only if none of those ten ports is free does it set `usingExternalGateway = true` and connect to 18789. The fence occupies the ten ports with minimal HTTP 503 responses and makes that path deterministic: if it cannot reserve all ten, it cancels its own startup, and the gateway, which demands it with `Requires=`, does not start either. The result is recognised in the interface's logs by the line `Startup total: outcome=external_gateway`.

3.4 Startup, step by step

When AutoClaw is opened from its shortcut, the launcher does four things in order. First it prepares the environment: it sets a restricted `PATH`, removes `SUDO_PASSWORD` and `ELECTRON_RUN_AS_NODE`, recovers `XAUTHORITY` from the session's systemd manager if it is missing, and imports the graphical variables into that manager (`DISPLAY`, `WAYLAND_DISPLAY`, `XAUTHORITY`, `DBUS_SESSION_BUS_ADDRESS`, `XDG_CURRENT_DESKTOP` and `XDG_SESSION_TYPE`), because the services open dialogs, LibreOffice and Chromium. Second, it starts the fence, the configuration watcher, the Office bridge, the gateway and the idle timer, and if any of them fails it warns through a desktop notification and exits with an error. Third, it waits up to 180 seconds for `http://127.0.0.1:18789/health` to answer, and if it does not, it stops the services and does not open the interface, so that the interface does not fall into the Wine loop. Fourth, it opens the interface with `wine 'C:\users\Public\Desktop\AutoClaw.lnk'` (replication report, "Startup flow"; operating README, ROG Strix particularities).

Inside `autoclaw-gateway.service`, the startup script verifies that the state directory and `gateway-bundle.mjs` exist; creates or validates the `~/.openclaw-autoclaw` link; restores whichever Linux native modules are missing from `resources/gateway/openclaw/node_modules`; exports the environment equivalent to the one AutoClaw would pass to its child gateway (`OPENCLAW_STATE_DIR`, `OPENCLAW_CONFIG_PATH` pointing to `openclaw.linux.json`, `OPENCLAW_SKILL_READ_ROOTS`, the web search and trace addresses on port 18432, and a compile cache of its own in `~/.cache/autoclaw-linux/compile-cache`); clears every network proxy variable; prepends the skills' Python environment to `PATH`; and runs the binary named by `AUTOCLAW_NODE_BIN`:

```
"$AUTOCLAW_NODE_BIN" --no-warnings --max-old-space-size=2048 gateway-bundle.mjs \
  gateway run --port 18789 --bind loopback --allow-unconfigured --auth none
```

3.5 The configuration translation and the own configuration file

The interface writes `openclaw.json` with Windows paths and rewrites it several times an hour (chapter 8). The synchroniser, which runs as a preparatory step of the gateway and every time the `autoclaw-config-sync.path` unit detects a

change in that file, produces `openclaw.linux.json` under these rules: it translates every string beginning with a Windows drive letter through `~/.wine/dosdevices` and rejects it if it escapes its drive; it re-expresses the state directory as `~/.openclaw-autoclaw`; it forces `gateway.mode=local`, `gateway.bind=loopback`, `gateway.auth.mode=none` and `gateway.reload.mode=hot`; it removes `browser` from `tools.deny` and configures the native browser with `/usr/bin/chromium-browser`, visible and with 45,000 and 20,000 millisecond deadlines; it adds `~/.local/share/autoclaw-linux/extensions` to `plugins.load.paths`; and it disables the original Office plugin and enables `libreoffice-bridge`. It rejects unexpected symbolic links, writes atomically and durably, leaves the state in mode 700 and both files in 600, and if the source is half-written it keeps the previous destination and exits with 75, which means temporary failure (replication report, “Configuration synchronisation”).

The `hot` reload mode is deliberate. With it, the gateway applies live those changes that allow it and records as “hot mode ignoring” those that would require restarting the process, such as the `zcode-runtime` plugin switches the interface writes on connecting; with `hybrid` mode, by contrast, that same write caused a full restart, the restart closed the WebSocket with code 1012, and on that code the interface disables its automatic reconnection and, in external gateway mode, does not reconnect on its own (operating README, “Invariants”).

On top of that translation, the synchroniser injects the own configuration file `~/.local/share/autoclaw-linux/providers.local.json`, which lives outside AutoClaw’s state so the interface neither sees it nor deletes it when renewing its credential, and so it survives an update. It accepts three sections: `providers` is poured into `models.providers` (a provider named `zai` is ignored with a warning, because that one is written by the interface); `agentDefaults` is poured with priority over `agents.defaults`; and `messages` is merged key by key with whatever the interface writes under that same section. The `.path` unit watches only `openclaw.json`, so after editing the own configuration file the synchroniser has to be run by hand. The current structure, with the key replaced, is this one (compendium 4.1; measurement of 14 Sept. of the sections and values, without reading the key):

```
{
  "providers": {
    "xiaomi-coding": {
      "api": "openai-completions",
      "baseUrl": "https://token-plan-sgp.xiaomimimo.com/v1",
      "apiKey": "<MIMO TOKEN PLAN KEY>",
      "timeoutSeconds": 600,
      "models": [
        {"id": "mimo-v2.5-pro", "name": "Xiaomi MiMo V2.5 Pro", "contextWindow": 1048576,
         "maxTokens": 131072, "input": ["text"], "reasoning": true},
        {"id": "mimo-v2.5", "name": "Xiaomi MiMo V2.5", "contextWindow": 1048576,
         "maxTokens": 131072, "input": ["text", "image"], "reasoning": true}
      ]
    }
  },
  "agentDefaults": {
    "model":      {"primary": "xiaomi-coding/mimo-v2.5-pro"},
    "imageModel": {"primary": "xiaomi-coding/mimo-v2.5"},
    "pdfModel":   {"primary": "xiaomi-coding/mimo-v2.5"},
    "pdfMaxPages": 60
  },
  "messages": {
    "tts": {"provider": "microsoft",
            "providers": {"microsoft": {"voice": "es-AR-ElenaNeural"}}}
  }
}
```

3.6 The user units

AutoClaw’s units are not enabled at login: they live only while the application is open, the launcher starts them and the idle timer stops them. That is why, after rebooting the machine, the gateway and the voice player show as inactive until AutoClaw is opened, and that is the correct state. Dictation and its virtual keyboard, by contrast, are tied to the graphical session and start with it. Every unit of the integration sets `UMask=0077` and `UnsetEnvironment=SUDO_PASSWORD`.

Table 4

systemd user units, with their activation and their function

Unit	Activation	Function and parameters that matter
`autoclaw-gateway.service`	static; started by the launcher	the gateway; `Requires=autoclaw-port-fence.service`, `Restart=always`, `RestartSec=1`, `KillMode=mixed`, `MemoryHigh=2G`, `MemoryMax=3G`, `OOMScoreAdjust=200`, `Nice=5`; drop-in `autoclaw-gateway.service.d/10-node-oficial.conf` with `AUTOCLAW_NODE_BIN`
`autoclaw-port-fence.service`	static	the fence over ports 18790 to 18799
`autoclaw-office-bridge.service`	static	the Office bridge on 18860
`autoclaw-config-sync.path` and `.service`	static	`PathChanged` on `openclaw.json` triggers the synchroniser
`autoclaw-idle-stop.timer` and `.service`	static	every three minutes stops everything if there is no `AutoClaw.exe`
`autoclaw-voice-player.service`	enabled with `WantedBy=autoclaw-gateway.service` and `PartOf=`	the voice player, which starts and stops with the gateway
`dictado.service`	enabled in `graphical-session.target`; `Requires=ydotoold.service`	the dictation server, with `venv-gpu/bin/python` and `Restart=on-failure`
`ydotoold.service`	enabled in `graphical-session.target`	the virtual keyboard through `/dev/uinput`, unprivileged, with `--mouse-off`
`godmez-autoclaw-regimen.path` and `.service`	`.path` enabled	restores the regime inside the agent when something overwrites it

Note. The machine also has `godmez-settings-guard` and `godmez-memory-mirror`, which belong to the Claude Code regime and not to AutoClaw, and are not needed to replicate the integration.

3.7 Security invariants of the design

The gateway uses `gateway.auth.mode=none`, a compatibility exception valid only while the binding stays limited to the local loopback, and the launcher forces that on the command line as well. The launcher does not open the interface if the services or `/health` do not come up healthy, the fence cancels its startup if it cannot reserve all ten ports, and the gateway requires the fence. The `~/.openclaw-autoclaw` link must point to the expected state inside Wine, and the synchroniser rejects unexpected links, escapes from Wine's drives, and incomplete JSON. The Office bridge listens on the local loopback only, uses a random token per startup, limits each request to 32 MiB and does not automatically grant permission to send data outside; its plugin accepts only discovery files that are regular, private, owned by the current user and carrying an exact local endpoint. Document titles and paths enter the model's context as untrusted metadata, normalised, bounded and quoted (operating README, "Robustness and security invariants").

4. Installation step by step

The recipe follows the order in which a third party reproduces it on Fedora 44 with KDE Plasma in a Wayland session, starting from the portable bundle and from the live files of the ROG Strix where those superseded the bundle. Each step carries its oracle. Before modifying an existing file, a backup with a dated suffix is made. On this machine the `grep` of the interactive shell is a different program, so every oracle uses `/usr/bin/grep`. Steps 1 to 11 are AutoClaw; 12 and 13, voice and dictation, are optional; and the regime inside the agent, also optional, is in chapter 10.

Step 0. Measure the starting point without touching anything

```
cat /etc/fedora-release; uname -r; echo "$XDG_SESSION_TYPE"
rpm -q wine-core wine nodejs24 python3 libreoffice-core chromium
ls /usr/lib64/wine-wow64/wine/i386-windows/ | wc -l
free -g; swapon --show
systemctl --user show-environment | /usr/bin/grep -c '^SUDO_PASSWORD='
```

Oracle. Fedora 44, a `wayland` session, and zero credentials in the session manager's environment. On the ROG Strix, on 2026-09-09, `wine-core` 11.0-3.fc44 was already in its WoW64 build (819 32-bit files, enough for the installer, which is 32-bit), the `~/.wine` prefix already existed with Windows 10 build 19045 and `native,builtin` overrides for `vcrun2019`, and memory needed no changes: 30 GiB, 40 GiB of swap and no out-of-memory kills in thirty days. On the VivoBook, with some 11 GiB, the system's memory did have to be adjusted (replication report, "System changes for memory"). A dedicated prefix is not required: with overrides and without them, the interface works.

Step 1. Host packages

```
sudo dnf install --setopt=install_weak_deps=False --assumeno \
  wine wine-mono nodejs24-npm osslsigncode wl-clipboard \
  libreoffice-core libreoffice-writer libreoffice-calc libreoffice-impress \
  libreoffice-ure libreoffice-pyuno chromium poppler-utils kdialog curl
# having read the size, repeat without --assumeno
WINEPREFIX=~/.wine wineboot -u
```

Oracle. `wine --version` prints `wine-11.0 (Staging)`; `/usr/bin/python3 -c 'import uno; print("pyuno ok")'` prints `pyuno ok`; `/usr/bin/chromium-browser` exists. On the ROG Strix only the `wine` metapackage, `wine-mono`, `nodejs24-npm` and `osslsigncode` were missing: 32 packages and 188 MiB of download. Fedora's `pyuno` module is compiled against the system Python, so the Office bridge runs with `/usr/bin/python3` and never with a virtual environment.

Step 2. Obtain and verify the 1.17.8 installer

The version installed is 1.17.8 and not the 1.18.1 that `latest.yml` announces, because the native modules, the LibreOffice plugin, the synchroniser, the fence and the patch were validated against it (chapter 11). The installer sits in the same root as `latest.yml`, `https://autoglm-public-oss.z.ai/autoclaw/updates/`.

```
cd ~/Descargas
curl -fLO https://autoglm-public-oss.z.ai/autoclaw/updates/autoclaw-1.17.8-setup.exe
stat -c %s autoclaw-1.17.8-setup.exe     # 407685392
md5sum autoclaw-1.17.8-setup.exe         # 595fbb1a1c967a45ffb61fd8f3c859b9, equal to the ETag
sha256sum autoclaw-1.17.8-setup.exe
curl -fLO https://secure.globalsign.com/cacert/codesigningrootr45.crt
openssl x509 -inform DER -in codesigningrootr45.crt -out GlobalSign_CodeSigningRootR45.pem
openssl x509 -in GlobalSign_CodeSigningRootR45.pem -noout -fingerprint -sha256
osslsigncode verify -CAfile GlobalSign_CodeSigningRootR45.pem autoclaw-1.17.8-setup.exe
```

Oracle. The SHA-256 sum is `dd00046df93ff4a02ec540fc31073a1b9b3551ef9ba6d99fd9d2064fff445ffc`; the root's fingerprint is `7B:9D:55:3E:1C:92:CB:6E:88:03:E1:37:F4:F2:87:D4:36:37:57:F5:D4:4B:37:D5:2F:9F:CA:22:FB:97:DF:86`; and `osslsigncode` says "Signature verification: ok", with the signer "JINGSHENG HENGXING TECHNOLOGY PTE. LTD." of Singapore, an extended validation certificate from "GlobalSign GCC R45 EV CodeSigning CA 2020" and a DigiCert timestamp of 2026-08-26 at 16:18:50 coordinated universal time (installation log, "Done so far", steps 8 and 9). Without the root, which Fedora's certificate bundle does not carry, the tool says "unable to get local issuer certificate" even though the digests match, and that first failure is one of trust in the root, not of file integrity. If the downloaded certificate already comes in PEM format, the `-inform DER` conversion fails and it is used as is.

Step 3. Install AutoClaw under Wine

```
cd ~/Descargas && WINEPREFIX=~/.wine wine autoclaw-1.17.8-setup.exe /S
tail -3 ~/.wine/drive_c/users/$USER/.openclaw-autoclaw/logs/win-installer.log
ls -l ~/.wine/drive_c/"Program Files"/AutoClaw/resources/gateway/openclaw/gateway-bundle.mjs
```

Oracle. The installer log ends with `result=success` and `process_exit=0`, and `gateway-bundle.mjs` measures 48,621,000 bytes. On the ROG Strix the silent installation took 133 seconds and left 2.0 GB in `C:\Program Files\AutoClaw`; on the VivoBook, nine minutes. The installer tries to register Windows Defender exclusions and a firewall rule, which under Wine are harmless simulations, and extracts the gateway with its own `7z.exe` because Wine does not ship `tar.exe`.

Step 4. Copy the integration from the portable bundle

```
sha256sum ~/Escritorio/AutoClaw/AutoClaw_Fedora_andamiaje_portable_2026-09-21.tar.gz
mkdir -p ~/claude-setup/autoclaw/portable-2026-09-21
tar -xzf ~/Escritorio/AutoClaw/AutoClaw_Fedora_andamiaje_portable_2026-09-21.tar.gz \
    -C ~/claude-setup/autoclaw/portable-2026-09-21
PORTABLE=$(dirname "$(find ~/claude-setup/autoclaw/portable-2026-09-21 -name SHA256SUMS | head -1)")
cd "$PORTABLE" && sha256sum -c --quiet SHA256SUMS && echo "portable sums: ok"

mkdir -p ~/.local/share/autoclaw-linux ~/.config/systemd/user
cp -a "$PORTABLE/autoclaw-linux/." ~/.local/share/autoclaw-linux/
chmod 755 ~/.local/share/autoclaw-linux/bin/*
cd "$PORTABLE/systemd-user" && cp -a autoclaw-gateway.service autoclaw-port-fence.service \
    autoclaw-office-bridge.service autoclaw-config-sync.path autoclaw-config-sync.service \
    autoclaw-idle-stop.service autoclaw-idle-stop.timer ~/.config/systemd/user/
systemctl --user daemon-reload

GW=~/.wine/drive_c/"Program Files"/AutoClaw/resources/gateway/openclaw
ln -s ~/.wine/drive_c/users/$USER/.openclaw-autoclaw ~/.openclaw-autoclaw
for rel in @img/sharp-linux-x64 @img/sharp-libvips-linux-x64 @lydell/node-pty-linux-x64 \
           @napi-rs/canvas-linux-x64-gnu @mariozechner/clipboard-linux-x64-gnu sqlite-vec-linux-x64; do
  mkdir -p "$GW/node_modules/$(dirname "$rel")"
  cp -a ~/.local/share/autoclaw-linux/native-stage/node_modules/$rel "$GW/node_modules/$rel"
done

A=~/.local/share/applications/wine/Programs/AutoClaw.desktop
cp "$A" "$A.orig-wine" && cp "$PORTABLE/accesos-directos/menu-wine-Programs-AutoClaw.desktop" "$A"
```

Oracle. The bundle's sum is `2a159add9ee86a32e125bcde7873f126dc3e18d89286b78fb0e49a0f3f903d74` and its files match `SHA256SUMS`; the seven units show as `static` in `systemctl --user list-unit-files`; and `/usr/bin/grep '^Exec=' ~/.local/share/applications/wine/Programs/AutoClaw.desktop` points to `bin/autoclaw-launch.sh`. The six Windows native module versions in the bundle (`sharp` 0.34.5, `sharp-libvips` 1.2.4, `node-pty` 1.2.0-beta.12, `canvas` 0.1.100, `clipboard` 0.3.2 and `sqlite-vec` 0.1.9) must be the same as the Linux ones in the portable bundle; if they differ, the Linux variants with the new versions are installed inside `native-stage` with `npm install --no-save`. If the user is not named `usuario`, that path has to be replaced in the `Documentation=` lines of three units and in the shortcuts. The post-reboot checker's units are not copied, for the reason given in the note to table 3.

Step 5. Python environment and LibreOffice

```
/usr/bin/python3 -m venv ~/.local/share/autoclaw-linux/venv
R=~/.local/share/autoclaw-linux/venv-requirements-congelado.txt
~/.local/share/autoclaw-linux/venv/bin/pip install --dry-run -r "$R"
~/.local/share/autoclaw-linux/venv/bin/pip install -r "$R"
/usr/bin/python3 ~/.local/share/autoclaw-linux/bin/autoclaw-libreoffice-setup.py   # with LibreOffice
↪ closed
```

```
/usr/bin/python3 ~/.local/share/autoclaw-linux/tests/test_regressions.py
node ~/.local/share/autoclaw-linux/tests/test_runtime_snapshot.mjs
```

Oracle. The dry run resolves the forty exact versions of the frozen list; the installation takes some 371 MB; `test_regressions.py` gives 5 of 5 and the other prints `runtime_snapshot_security=ok`. The frozen list and the system Python are used, rather than the direct package list, in order to reproduce the validated environment without depending on another Python version. The LibreOffice script refuses to run if a `soffice.bin` is alive, because LibreOffice overwrites its profile on closing, and it leaves a timestamped backup.

Step 6. First opening through the original path, to sign in to Z.ai

```
cd ~/.wine/drive_c/"Program Files"/AutoClaw/resources/gateway/openclaw
env -u ELECTRON_RUN_AS_NODE WINEPREFIX=~/.wine setsid -f wine 'C:\users\Public\Desktop\AutoClaw.lnk'
```

Sign in to Z.ai from the window and, once `openclaw.json` exists, exit the application completely from the tray menu; if any process remains, `WINEPREFIX=~/.wine wineserver -k`. On this first opening the interface uses its `node.exe` under Wine and may be slow or fail, which is expected and inconsequential, because all that is needed is the session signed in. **Oracle.** `~/.openclaw-autoclaw/openclaw.json` exists with permissions `600`.

Step 7. Repair the four defects specific to Fedora

(a) The launcher: `ELECTRON_RUN_AS_NODE` and `XAUTHORITY`. The bundle's launcher carries neither repair. Either install the ROG Strix's live launcher (digest in chapter 14) or apply the diff from appendix A.1 of the compendium: `unset ELECTRON_RUN_AS_NODE` at the beginning, `env -u ELECTRON_RUN_AS_NODE` on the line that opens the interface, and the recovery of `XAUTHORITY` from `systemctl --user show-environment` when it is missing. **Oracle.** From a VS Code terminal, where `echo $ELECTRON_RUN_AS_NODE` prints `1`, the launcher opens the window all the same; and from an environment without `XAUTHORITY` the window also appears, at ten seconds in the measurement of 2026-09-10.

(b) The gateway's Node.

```
node -e 'console.log([...new Intl.Segmenter("en",{granularity:"grapheme"}).segment("ab")].length)'; echo
 ↪  $?
# "2" and 0: healthy.  No output and 139: segmentation fault, continue:
mkdir -p ~/.local/opt && cd ~/.local/opt
curl -fLO https://nodejs.org/dist/v24.18.0/node-v24.18.0-linux-x64.tar.xz
curl -fLO https://nodejs.org/dist/v24.18.0/SHASUMS256.txt
/usr/bin/grep ' node-v24.18.0-linux-x64.tar.xz$' SHASUMS256.txt | sha256sum -c -
tar -xJf node-v24.18.0-linux-x64.tar.xz
mkdir -p ~/.config/systemd/user/autoclaw-gateway.service.d
printf '[Service]\nEnvironment=AUTOCLAW_NODE_BIN=%s\n' \
  "$HOME/.local/opt/node-v24.18.0-linux-x64/bin/node" \
  > ~/.config/systemd/user/autoclaw-gateway.service.d/10-node-oficial.conf
systemctl --user daemon-reload
```

The gateway's startup script has to read the binary from `AUTOCLAW_NODE_BIN`, defaulting to `/usr/bin/node`, which is the diff of appendix A.2 of the compendium or the ROG Strix's live script. **Oracle.** The same `Intl.Segmenter` line with `~/.local/opt/node-v24.18.0-linux-x64/bin/node` prints `2`, and that binary's digest on the ROG Strix is the one in chapter 14. On the ROG Strix, with `nodejs24` 1:24.18.0-1.fc44, the line crashes with any granularity, language and locale, and the dump points at `v8::internal::JSSegments::Create`; the official binary, with the same Node and internationalisation library versions, does not. The gateway bundle uses `Intl.Segmenter` in four places, and the defect showed up as internal diagnostics and a search provider listing that died with code 139 without saying anything (Fedora Node memory; compendium 3.2).

(c) Calc's number formatting. In `office-bridge/autoclaw_office_bridge/calc.py`, the conversion of the Office contract's format strings is done with the `en-US` locale set explicitly, and the general format is identified by its standard

key rather than by its translated name (diff of appendix A.4 of the compendium). The reason is that the contract expresses formats in Excel's English convention, so the conversion's locale is part of the contract; with the system's, `es_ES.UTF-8`, LibreOffice stored `000` where the contract asked for `0.00`. **Oracle.** With copies of the three documents in `tests/docs/` open in LibreOffice, `/usr/bin/python3 tests/test_bridge.py` gives 80 of 80 and the originals keep their sums.

Step 8. First full startup and acceptance

```
~/.local/share/autoclaw-linux/bin/autoclaw-launch.sh
curl -s http://127.0.0.1:18789/health
ss -ltnp | /usr/bin/grep ':18789'
pgrep -fa 'node\.exe' || echo "no node.exe"
ss -tn state established | /usr/bin/grep -c ':18789'
/usr/bin/grep -rhF 'outcome=' --exclude='autoclaw-auth*' ~/.openclaw-autoclaw/logs/ | tail -1

export OPENCLAW_STATE_DIR=~/.openclaw-autoclaw \
       OPENCLAW_CONFIG_PATH=~/.openclaw-autoclaw/openclaw.linux.json
cd ~/.wine/drive_c/"Program Files"/AutoClaw/resources/gateway/openclaw
~/.local/opt/node-v24.18.0-linux-x64/bin/node --no-warnings gateway-bundle.mjs doctor --lint
 ↪  --non-interactive
```

Oracle. `/health` answers (at four seconds on the ROG Strix) and the window appears (at seven); the owner of 18789 is the `openclaw` process and not `wineserver`; there is no `node.exe`; there are two established connections against 18789; the interface's logs say `outcome=external_gateway`; and the internal diagnostics give 22 checks, 0 errors, 6 warnings and 3 notes. The warnings describe the design and not a defect: Z.ai keys in plain text, which the interface writes, and skills permitted but not available. The full battery of 2026-09-09 added web search through the 18432 bridge with three real results, the native browser with a real snapshot of example.com, scheduled tasks created and deleted against an intact SQLite database, idle shutdown with the interface closed, and a gateway restart with the script and automatic reconnection (compendium 3.3).

Step 9. Silence the updater

```
curl -sI https://autoglm-public-oss.z.ai/autoclaw/updates/channels/wine-manual/latest.yml | head -1
printf '{\n  "channel": "wine-manual"\n}\n' > ~/.wine/drive_c/"Program
 ↪  Files"/AutoClaw/resources/channel.json
```

Oracle. Before writing the file, the `wine-manual` channel answers 404. After restarting the interface, no new "Found version" or signature verification line appears in `~/.openclaw-autoclaw/logs/autoclaw-compat.log`, and the updater's pending directory stays empty. The file `AppData/Roaming/autoclaw/channel.json` is the business channel, says `official` and is not touched. The mechanism is described in chapter 11.

Step 10. Own providers, defaults and MiMo in the window

The bundle's synchroniser does not know about the own configuration file. Install the ROG Strix's live synchroniser, which incorporates the diff of appendix A.3 of the compendium and the merging of the `messages` section, create `providers.local.json` with the structure of chapter 3.5 and an empty key, and load the key with an editor, so that it passes through neither the shell history nor any conversation:

```
umask 077
${EDITOR:-nano} ~/.local/share/autoclaw-linux/providers.local.json
chmod 600 ~/.local/share/autoclaw-linux/providers.local.json
/usr/bin/python3 ~/.local/share/autoclaw-linux/bin/autoclaw-config-sync.py
~/.local/share/autoclaw-linux/bin/autoclaw-gateway-restart.sh
```

The script `bin/autoclaw-set-provider-key.sh` serves this purpose and, since 2026-09-20, accepts both shapes of the own configuration file, the flat one and the sectioned one; with a second argument it writes to a copy and does not synchronise, which is how `tests/test_set_provider_key.py` tests it (chapter 14.4). MiMo is then registered in the window with the values of table 5 (chapter 6.2). **Oracle.** The defaults comparison of chapter 12.4 prints `IGUAL` on all four keys; the trace of one window turn shows `provider=custom__…` and `model.id=mimo-v2.5-pro` towards `token-plan-sgp.xiaomimimo.com` with status 200, with no requests to `autoglm-api` in those minutes. The image check is a single interactive call, because of the plan's contract (chapter 6.4).

Step 11. The document extractor patch

```
X=~/.wine/drive_c/"Program
 ↪  Files"/AutoClaw/resources/gateway/openclaw/dist/extensions/document-extract/document-extractor.js
sha256sum "$X"
P=~/.local/share/autoclaw-linux/bin/autoclaw-pdf-notice-patch.py
python3 "$P" --check; echo $?     # 1: original installed
python3 "$P"
python3 "$P" --check; echo $?     # 0: patched
~/.local/share/autoclaw-linux/bin/autoclaw-gateway-restart.sh
```

It requires `patches/document-extractor/original.js` and `patched.js` next to the applier, which are copied from the ROG Strix or rebuilt from appendices D.1 to D.3 of the compendium. **Oracle.** Before patching, the installed extractor gives `8a7e77a9cab9b4019f8735f905b2a293e02d5b32d8aa93022effffc06cf542c`; if it gives anything else, the installation is not the same 1.17.8 and the patch is not applied. Afterwards, `--check` gives 0 and the extractor gives `fe307e1122ee5c62870dc8495a4850444641d6eb143efe590cabb94779d152d6`.

Step 12. Voice of the replies (optional)

With the `messages.tts` section already set in the own configuration file and synchronised, the voice preferences are turned on from the gateway's console, which writes `~/.openclaw-autoclaw/settings/tts.json` without touching the configuration, and the player is installed as a unit tied to the gateway:

```
cd ~/.wine/drive_c/"Program Files"/AutoClaw/resources/gateway/openclaw
N=~/.local/opt/node-v24.18.0-linux-x64/bin/node
$N --no-warnings gateway-bundle.mjs infer tts set-provider --local --provider microsoft
$N --no-warnings gateway-bundle.mjs infer tts enable --local
systemctl --user daemon-reload && systemctl --user enable autoclaw-voice-player.service
pactl get-sink-mute "$(pactl get-default-sink)"
```

(Both `infer` commands are run with the environment exported in step 8.) **Oracle.** `tts.json` contains `{"tts": {"provider": "microsoft", "auto": "always"}}`; `infer tts status --gateway` reports it; after a turn, a file appears in `~/.openclaw-autoclaw/media/outbound/` and the player's journal says it played it; and the audio output is not muted (`Mute: no`).

Step 13. Dictation (optional)

```
sudo dnf install ydotool
getfacl /dev/uinput | /usr/bin/grep "user:$USER"
python3 -c 'import torch; print(torch.__version__, torch.cuda.is_available())'
/usr/bin/python3 -m venv --system-site-packages ~/.local/share/dictado/venv-gpu
V=~/.local/share/dictado/venv-gpu/bin/pip
$V install --no-deps openai-whisper==20250625
$V install tiktoken==0.14.0 numba==0.67.0 onnxruntime==1.30.0
```

Dictation requires the system Python to already have PyTorch with ROCm working on the dedicated card; on the ROG Strix that is torch 2.9.1 with ROCm 7.1, and the dictation environment sees it because it is created with `--system-site-packages`. `openai-whisper` is installed without automatic dependencies because its installer wants to pull in `triton`, which is NVIDIA's and would clash with ROCm; the dependencies the system did not provide were `tiktoken` and `numba`, and `onnxruntime` runs the voice detector on the processor. The Silero v6 detector is the same file distributed by `faster-whisper`, copied to `~/.local/share/dictado/silero_vad_v6.onnx`. Then `dictado.py`, its configuration, the units of table 4 and the launcher `~/.local/share/applications/net.local.dictado.desktop` are installed, the latter's `Exec=` line being `/usr/bin/python3 /home/usuario/.local/share/dictado/dictado.py toggle`:

```
{
  "engine": "openai-whisper",
  "model": "turbo",
  "device": "cuda",
  "language": "es",
  "beam_size": 5,
  "insert": "paste",
  "keep_recording": false
}
```

```
systemctl --user daemon-reload
systemctl --user enable --now ydotoold.service dictado.service
journalctl --user -u dictado.service -b --no-pager | /usr/bin/grep -F 'voice gate on'
```

The shortcut is assigned **by hand** in System Settings, Keyboard, Keyboard Shortcuts, "Add New", "Command or Script", with that same command and the Control+Alt+Space combination. It is never assigned from a terminal against the live compositor, because of the trap in chapter 12.8. **Oracle.** The journal says `voice gate on (silero_vad_v6)`; `dictado.py transcribe:<file>` over 9.8 seconds of silence answers `[no speech: 0.0 s over 9.8 s of audio]` within a few tenths; and audio of known text is transcribed in full. On the ROG Strix `keep_recording` returned to `false` on 2026-09-20, which is the installation value. The first load takes some 8.8 seconds and a 2.2 second warm-up absorbs the compilation of the card's kernels; the model ends up in `~/.cache/dictado/modelos/large-v3-turbo.pt`, of 1,617,941,637 bytes.

5. The three slownesses

The expression "the translator's slowness" admits three readings among the things measured, and all three deserve explanation because they have different causes and different remedies. The first is the graphics translation chain under Wine, which is why the window draws in software. The second is the slowness of opening the application, whose trigger was AutoClaw's retry loop. The third is the slowness of dictation, which turned out not to be slowness at all but an engine inventing text over recordings without speech.

5.1 The graphics translation chain: the window draws in software

AutoClaw is Chromium packaged by Electron, and under Wine an accelerated window would have to travel a five-layer chain: Chromium draws with ANGLE, which translates OpenGL into Direct3D 11; Direct3D 11 is served by Wine's own translator, which in this prefix is Wine's Direct3D and not DXVK, because neither `d3d11.dll`, nor `dxgi.dll`, nor `d3d9.dll`, nor `d3d12.dll` in `system32` contains the string "dxvk"; and Wine's translator ends in OpenGL or Vulkan over Mesa, the card's free driver (installation log, session 01, unit 2; measured 2026-09-10). In practice that chain is never used, and the reason is not an incompatibility in any of its layers: the graphics process starts with `--use-gl=disabled` because AutoClaw disables acceleration by its own decision.

The string `use-gl` appears zero times in the 309 MB of `app.asar`, so the flag is a consequence and not a literal. The cause is in `out/main/index.js`, inside that package, in a block repeated identically for Windows and for Linux: it calls `app.disableHardwareAcceleration()` and adds the flags `disable-gpu`, `disable-gpu-sandbox`, `no-sandbox`, `disable-gpu-compositing` and `disable-software-rasterizer`. Since the call happens before the graphics unit is initialised, no command-line flag reverses it, and enabling acceleration would require modifying the vendor's package. It would be

possible, because the Electron fuses burned into `AutoClaw.exe` state that neither is `app.asar` integrity validated nor is the program obliged to load only from it, but there is a trap: the same block is the one that passes `no-sandbox`, and neutralising it whole leaves Chromium raising its security sandbox under Wine, which is the shortest path to the window not opening at all. A correct patch would have to remove only the four graphics flags and the call, preserving the two sandbox ones.

The cost of having no acceleration was measured with a sampler validated against a load of exactly one core, which registered 100.0 %. With the window open and idle, the interface's four processes add up to 1.5 % of a core, with a maximum of 1.7 % across fifteen consecutive twenty-second windows. Forcing the worst case with 179 full resizes in 22 seconds, some eight full-surface redraws per second, the interface reached 130.2 % of a core (70.1 the compositing process, 54.6 the one that draws and 5.6 the main one), plus 13.6 for XWayland and 11.8 for the Wine server; on a sixteen-thread machine that is 8 % of total capacity and only while redrawing at a rate no real use produces. Throughout the exercise the dedicated card sat at 0 % and the integrated one averaged 9.8 %, and there is a structural reason for that: the display hangs off the integrated graphics, which is where the compositor runs, so drawing on the dedicated one would additionally require copying every frame between the two cards.

The decision was to touch nothing on the graphics side. The wait one feels when using AutoClaw lies in the gateway's startup, five or six seconds from cold, and in the model's response, 3.9 to 6.9 seconds for minimal turns, and both are waiting on network and remote service, where the processor is not drawing but waiting. The only thing acceleration could offload is that 8 % peak during a forced redraw, in exchange for patching a signed binary, redoing the patch on every version, dodging the sandbox trap and betting on a five-layer chain working.

5.2 The slowness on opening: the retry loop

The original slowness on opening AutoClaw, diagnosed on the VivoBook on 2026-09-02, had two layers (replication report, "Object, root diagnosis and architectural decision"; measured there). In the first, the gateway AutoClaw starts with its embedded `node.exe` under Wine took some seventy seconds to answer `/health`, while AutoClaw's internal deadlines, of fifty, thirty and fifty-three seconds across three successive attempts, declared it dead and relaunched it in a loop, with "runtime patch" steps of 217 and 132 seconds and a memory peak of 2.27 GB; and even when it did come up, Wine failed the permission change on OpenClaw's SQLite database (`EBUSY` on `openclaw.sqlite-shm`), which broke scheduled tasks, pairing and agent memory, without the internal diagnostics being able to repair it, because the root lay in Wine and not in the configuration. In the second layer, the VivoBook had its 11 GiB of memory chronically saturated, and the retry loop was the trigger that overflowed it. The ROG Strix, with 30 GiB, did not have the second layer.

The repair of the first layer is the entire architecture of chapter 3: the native gateway comes up in a median of 3.546 seconds to `/health`, and the port fence guarantees the interface will not launch its own again. What remains of the wait on opening was measured on 2026-09-10 across three rounds: with the gateway down, the window appears at 10.1 seconds (10.179, 9.948 and 10.130), of which 3.8 belong to the gateway, because the launcher waits for `/health` with a one-second loop before opening the interface; with the gateway already alive, at 5.4 (operating README). The difference of 4.7 seconds per opening is what disabling or lengthening the idle timer would save, in exchange for permanently occupying the 466 MiB that the gateway's control group measures at rest when freshly started (654.6 after a day's work), under its 2 GiB limit. The decision was to leave the timer as it is.

There are two life-cycle details that get mistaken for slowness. Closing the window shuts nothing down: AutoClaw stays in the system tray with its four processes alive, and since the timer decides by processes and not by windows, the gateway stays alive and the next opening is warm; the cold start is paid only when the application is truly exited. And a leftover rendering process from an earlier instance, which appears when the main process is terminated with a signal, is enough to keep the timer from ever shutting the gateway down (chapter 12.8).

5.3 The slowness of dictation: an engine that invented text

On 2026-09-11, between 10:26 and 10:31, the first five real dictations felt slow and poor. The service journal recorded 9.8 seconds of audio transcribed in 1.0 second yielding 5 characters; 1.3 in 1.1 with 8; 4.5 in 1.1 with 25; 14.5 in 1.1 with 18; and 33.0 in 2.5 with 56. Transcription was not the slow part; what was anomalous was the scarcity of text (installation log, "First real use of dictation").

The initial hypothesis, a microphone set too low, was refuted, and the secondary one turned out to be the entire mechanism. The control that decided it was to feed the installed engine pure digital silence, in the same format dictation records: over

9.8 seconds of silence it returned “Gracias por ver el video” in 1.09 seconds, and over 33 seconds, a production company’s copyright line repeated, in 2.11. The delays reproduce those of the earlier dictations within one and four tenths, and the amount of text is of the same order. What read as slowness was the signature of an engine that receives a recording without speech and, rather than saying so, invents a short phrase (installation log, session 09; Whisper memory).

The engine does not know it heard nothing because the chosen model has a degenerate no-speech probability in what it publishes per segment: `large-v3-turbo` gives 1.07×10^{-10} over silence at half precision on the card, 1.06×10^{-10} at single precision and 8.38×10^{-11} with another implementation on the processor, while the `small` model gives 0.83 and 0.95 over the same silence. It is not the precision, because single gives the same; it is not the implementation, because the two agree; it is the model, whose no-speech head was left uncalibrated, and with it its internal silence filters never fire. The microphone, measured, works: two recordings with the gain at 23 % gave root-mean-square levels of −19.9 and −20.7 dBFS.

The repair was a voice detection gate in front of the engine, described in chapter 9.5, which resolves an empty recording in 0.197 seconds instead of transcribing it, and reports “Dictado: no se escuchó tu voz”. With real speech, on 2026-09-13 from 20:00 to 20:01, three dictations from the internal microphone of 19.9, 22.5 and 7.2 seconds, with 2.98, 5.95 and 6.46 seconds of detected speech and root-mean-square levels of −21.7, −23.1 and −12.3 dBFS, took 1.57, 1.65 and 1.70 seconds from the second keypress to the pasted text, of which transcription took 1.20, 1.30 and 1.37 and pasting 0.20 (measured; dictation journal). The cause of the speechless recording of 2026-09-11 at 10:26 remains **not established**, because the microphone works and it did not recur; what would establish it is an empty dictation with its journal line, which states the level and the device, and a level of −240 dBFS would mean a stream that delivered no samples at all, not a quiet voice.

6. Providers and models

6.1 Z.ai, the originating provider

On signing in to Z.ai from the window, with a Google account in this installation, the credential goes from `unknown` to `healthy` and the interface writes the `zai` provider’s models into `openclaw.json`: seven as of 2026-09-09 and six since 03:54 on 2026-09-11, when Z.ai changed its catalogue and the interface rewrote it (compendium 3.3; installation log, session 09). The interface stores those credentials in plain text and renews them hourly, and the gateway receives them through the translated configuration and applies them live (chapter 8). Before the own models were fixed, images and documents resolved to `zai/zai_auto-fast` and spent AutoClaw credits.

The interface additionally queries, on its own account, Z.ai’s account service (`/userapi/v1/user-info` and `/agentpay/v1/assistant/product-info`, with the `[AuthAPI]` prefix in its log). Those queries are not inference, do not spend credits and do not go through the gateway, which is why the service name in the interface’s log is useless for knowing which model served a turn: the consumption oracle is the gateway’s `AutoClawRequestTrace` (compendium 8.3).

6.2 MiMo in the window: a custom model

The interface sends its own model on every request and ignores `agents.defaults.model.primary`, so the gateway’s default only governs when the agent runs from the console. For the window to converse with MiMo, it has to be registered in the window itself, under “Models and API”, “Custom models”, “Add custom model”; the form’s field names and validators were read in the interface’s package and a screenshot confirmed them (compendium 5.1; measured and cited from the code on 2026-09-10).

Table 5

Custom model form values for MiMo

Field	Value	Remark
Provider	Custom	internal value `custom`; always the last option
Model ID	`mimo-v2.5-pro`	free text even though the grey hint says “Select a model”; the validator accepts letters, digits and `-._/:`
Display name	MiMo V2.5 Pro	

Field	Value	Remark
API Key	the token plan key	begins with the plan's prefix; not transcribed here
API protocol	OpenAI	internal value `openai-completions`
Base URL	`https://token-plan-sgp.xiaomimimo.com/v1`	
Context length	1048576	the model's published specification; the form offers 200000 as filler
Maximum output	131072	the model's published specification; the form offers 32000 as filler

Note. With the filler values, the agent would be left with a fifth of the context the model admits. The two magnitudes are not knobs: they are the specification the provider publishes.

The program's own "Connectivity test" returned "Test successful" against `/v1/chat/completions` before registration. The interface publishes the registration as a provider `custom__<identifier>` in `openclaw.json`, in this installation `custom__xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx`, with the key in plain text, and the synchroniser keeps it and adds the `xiaomi-coding` of the own configuration file on top: both reach the same service and do not collide. The cost accepted is that the key also exists inside the Wine prefix, with owner-only permissions. The oracle was satisfied by a window turn at 13:56 on 2026-09-10, which left in the gateway's journal the request to `token-plan-sgp` with `model.id=mimo-v2.5-pro` and the response with status 200 at 6.876 seconds, with no request to `autoglm-api` in the preceding twenty minutes. The registration survived four cycles of truly exiting the application, shutting down the five units and reopening, and nine gateway restarts (compendium 6.2).

6.3 Images and documents: the agent's default models

The model the window converses with has no image input, so the gateway falls back on `agents.defaults.imageModel` for images and `agents.defaults.pdfModel` for documents. Each model's vision capability was measured against the service rather than read: an instrument draws a freshly drawn four-digit number, whose probability of being guessed by chance is one in ten thousand, and repeats the request without the image as a control. `mimo-v2.5` read the exact digits at 360 by 180 and at 2000 by 2000 pixels; `mimo-v2.5-pro` returned 404 with "No endpoints found that support image input" (compendium 6.1; measured 2026-09-10).

By decision of 2026-09-10, both point to `xiaomi-coding/mimo-v2.5`, fixed in `agentDefaults` of the own configuration file, with the backup `providers.local.json.bak-2026-09-10-s02`. It was verified that `infer image describe` without an explicit model resolved to `xiaomi-coding/mimo-v2.5` with a trace to `token-plan-sgp`, status 200 and the exact digits; that the document tool, without an explicit model, got right the number in a figure with no text layer; and that the window inherits the setting, because a turn with a genuinely pasted image, at 15:10, left four requests, all to `token-plan-sgp`, alternating `xiaomi-coding/mimo-v2.5` to see the image and the custom model to converse.

From that architecture follows an effect worth knowing: the model that converses never sees the image. The gateway sends the image to the image model, receives a description in text and passes that to the main model, which reasons over the text; in the measured turn, the main model worked on a description of the kind "An animated, muscular warrior with long, spiky blonde hair…" and hesitated between two characters over the colour of the outfit. A turn with an image may additionally cost two vision passes, because the agent asks for a second one on its own. This is not specific to MiMo but the way the gateway compensates for the main model's lack of vision, and it was the same with Z.ai; if the model needed to look at the image itself, a model with vision would have to be made the agent's primary.

6.4 The MiMo token plan contract

The plan's page (`mimo.mi.com/docs/en-US/price/token-plan`; cited in compendium 4.4) states that the quota "can only be used in programming tools (such as OpenClaw, OpenCode, etc.)" and that it is "prohibited from being used in the form of API calls for request behaviors in clearly non-Coding scenarios such as automated scripts and custom application backends", warning that using it outside that scope "will be considered a violation or abuse". The restriction concerns the context of use and not the modality, and OpenClaw is named by the provider. From that follows the rule governing this

installation: the MiMo plan is not used for automated work, and every measurement of the document tool is made against the engine and the extractor, with no model. As of 2026-09-10 the plan held 4,100,000,000 credits and expired on 2026-10-10; the later balance was not measured.

6.5 GitHub Copilot

GitHub Copilot was authenticated with `bin/autoclaw-provider-login.sh`, which uses the device code flow with the gateway's environment, but none of its models answers, neither Anthropic's nor OpenAI's: the server replies `model_not_supported`. It was decided on 2026-09-10 to leave it at that and spend no more on it (compendium 4.3).

6.6 AutoClaw's credits

The interface writes its balance as `[IPC] Wallet v2 display items: reward:N` in `~/.openclaw-autoclaw/logs/autoclaw-compat.log`, with the time in coordinated universal time, so the series can be followed without looking at the window. The balance also rises through promotions, as on 2026-09-10 at 14:19:43, when it went from 4,892 to 5,292 without anyone doing anything, and that is why consumption is decided by the gateway's trace. That same day the interface restarted itself at 14:20:49, one minute after the increase; with a single occurrence, that is temporal coincidence and not an established cause.

7. The document tool

7.1 How it treats a document

The document tool does not do what its name suggests (compendium 6.1; cited from the code and measured 2026-09-10). If the model's provider does not declare native documents, which Anthropic and Google do and none of this installation's providers does, the document is extracted locally with the `clawpdf` engine and what travels to the model is the result: text and page images. If the text exceeds a minimum number of characters, nothing is rasterised. And with a single document, if its first three pages add up to fewer than two hundred characters, the tool replies that it appears to be scanned and refers to the optical character recognition skill, which in this installation has no bridge, without calling any model. So, with one document, page images never reach the model, and vision only matters with two or more documents and at least one without a text layer. The package's magnitudes are ten megabytes per document, twenty pages by default, ten documents per call, a two-hundred-character threshold, a budget of four million pixels and a maximum dimension of ten thousand pixels.

7.2 Three cuts that said nothing

Table 6

Cuts made by the document tool, measured against the engine and with no model

Cut	Magnitude and origin	What it leaves out	When it bites
Page limit	`DEFAULT_MAX_PAGES = 20`, configurable with `agents.defaults.pdfMaxPages`, today 60	the pages exceeding the amount admitted per call; until the patch of 2026-09-20 it additionally discarded every page beyond the limit (chapter 7.5)	with pages of little text
Character limit	`MAX_EXTRACTED_TEXT_CHARS = 2e5`, a package constant with no key	cuts at exactly 200,000 characters, mid-page if need be	with dense text, near page 66 at some 3,000 characters per page
Pixel budget	four million pixels in total per document, not per page	a sixty-page letter-sized scan arrives as four full pages, a fifth shrunk and three remnants of a few pixels	with two or more documents, when one has no text layer

Note. The two remaining limits fail noisily and do not deceive: the maximum size raises `File exceeds N bytes` and going past ten documents returns `Too many PDFs`. The pixel budget governs page rasterisation and not attached images: a nine-megapixel image passed and the model read its digits (operating README; compendium 7.1 and 7.2).

Until the patch of chapter 7.4, none of the three cuts carried a warning, and the model answered about a mutilated document believing it whole. In the test, with a twenty-five-page document hiding a key on page 25, the agent gave as its answer the last thing it saw, the number 20, instead of reporting that it could not find it: the question returned “8274 20” with the limit at twenty and “8274 2403”, both exact keys, with the limit at sixty. A document whose text the character limit cuts can be read in full by asking for ranges with the `pages` argument, because each call has its own budget.

7.3 `pdfMaxPages` at sixty

The page limit has been at sixty since 2026-09-10 by the owner’s decision, fixed in `agentDefaults` of the own configuration file with the backup `providers.local.json.bak-2026-09-10-s03`. The value comes from aligning the two limits: at the density of an academic article, some three thousand characters per page, the character limit falls near page sixty-six, so a lower page limit bites before the character one and a higher one does not enlarge what the model receives. Raising the limit does not enlarge a scan’s pixel load either, which weighed the same 150,208 bytes at twenty and at sixty.

The cost of a higher limit was measured against the installed extractor, with limits of 20, 60, 120, 250 and 500 and four documents (compendium 7.4). In a document with text, the engine stops where the two hundred thousand characters bite and a higher limit costs nothing: in the two real books, it walked 107 and 79 pages in 0.32 and 0.51 seconds, with peak memory below 210 MiB. In a scan, the engine walks every admitted page at some 0.16 milliseconds per page. The only cost the limit bounds is that of a very sparse text document, at 7.2 milliseconds per page, with a declared limitation: the figure was derived from two books and not from a population.

7.4 The patch that makes the cuts visible

The cause of the silence lies in a precise layer. The `clawpdf` engine reports the page count, the character cut, the pixel budget cut and the pages walked, but the gateway’s extractor returned only the text and the images. The only channel reaching the model without touching the 48 MB package is the text, so the patch lives in `resources/gateway/openclaw/dist/extensions/document-extract/document-extractor.js` and prepends to the text a `[PDF notice: …]` line stating which pages were left out and with what `pages` range the rest can be read; the limit clause appears only when the selection reaches the limit (compendium 7.3; decision of 2026-09-10).

The design respects an invariant that is not obvious. The extractor’s text has four consumers and three of them decide by its length: the scan guard, which answers “appears to be scanned” if its three-page probe brings fewer than two hundred characters; the message assembly, which discards empty text; and the branch for models without vision, which raises an error if no document carries text. That is why no notice is added to a text of fewer than two hundred characters carrying no images, and the value is the guard’s literal threshold, so the guard decides the same with the patch and without it.

Verification used twenty-two cases, sixteen fabricated and six real documents (two books cut by characters, one cut by pages, two scanned comics and a one-page certificate as a control). In all twenty-two, the text after the notice and the images came out byte-identical to the engine’s, and the guard decided the same. Twenty cases came out coherent, and the two incoherent ones were foreseen and are caused by `parsePageRange` (chapter 7.5). Three controls showed that the checks were capable of failing: the pristine extractor gave zero notices and six incoherences across six cuts; a mutant without the suppression rule took the boundary document’s probe from 124 to 256 characters and bent the guard; and that boundary document, with real text below the threshold, is the only case that exposes the guard. The applier was validated on copies, 21 of 21, and on installing it, on 2026-09-10 at 16:39:15, the gateway restarted with the script came up healthy at 5.06 seconds and the window reconnected on its own.

The patch is not edited by hand. In `~/.local/share/autoclaw-linux/patches/document-extractor/` live `original.js`, `patched.js` and the backups, and `bin/autoclaw-pdf-notice-patch.py` applies it, checks it with `--check` (0 patched, 1 original, 2 foreign file) and removes it with `--revert`. It only replaces the exact file it was derived against, identified by its digest, and before any other it refuses without writing, which is what will happen if an update changes the extractor.

7.5 The limit as a per-call budget: `parsePageRange`, repaired

Until 2026-09-20 one silent case remained whose cause lay in the package and not in the extractor. `parsePageRange` discarded every page above `pdfMaxPages` before the request reached the engine, so an entire range above the limit, such as `61-80`, arrived empty and without a notice; a request such as `1,70` arrived as `[1]`; and with a single document, asking only for pages above the limit made the tool answer that the document appears to be scanned, which is false. The engine never shared that reading: `clawpdf` applies the limit as an amount, with `options.pages.slice(0, limit)`.

The local patch of 2026-09-20 aligns the gateway with the engine. The function expands the requested range without trimming by page number and rejects with an explicit error any selection holding more pages than the limit, checking that inside the range branch before enumerating it and over the union at the end, so that an enormous range fails before occupying memory. The function lived duplicated, with different wording, in `gateway-bundle.mjs` and in `dist/openclaw-tools-Bm2s_FnJ.js`, and both are patched. The extractor's truncation notice, which claimed that pages beyond the limit could not be read "not even with the `pages` parameter", now states how many were processed and with what range the following ones can be read; its condition went from including the limit number to counting the selected amount, which additionally eliminates the false notice a range such as `55-60` used to receive.

It is applied by `bin/autoclaw-page-range-patch.py`, in the same shape as the notice applier: it identifies each block by its exact text, refuses with 2 before a file other than the one it was derived against, backs up before writing and never restarts anything. Its bench, `tests/test_page_range_patch.py`, tests three layers on copies, each with a negative control that fails over the pristine text (chapter 14.4).

The entire chain, except the call to the model, was measured on 2026-09-20 against the installed extractor and the real engine, with an eighty-page document whose per-page marks come from system entropy and are verified with `pdftotext`, a tool foreign to the package under measurement. With the limit at sixty, the range `61-80` returns the twenty pages with their marks, `1,70` returns both, `55-60` returns six and no notice, and a selection of sixty-one fails with an error naming the amount and the limit; over the pristine package, mounted outside the vendor's tree, the first two come back empty and as page one, and with a single document the tool would answer that it appears to be scanned. The only thing still unmeasured is the interactive turn, which consumes the token plan and requires a person (chapter 14.5).

8. Who rewrites the configuration

8.1 Why it had to be measured

The interface and the synchroniser write over the same pair of files, and the practical question was whether the own defaults survive the interface's rewrites, something that cannot be answered by seeing them looking fine at one instant. To answer it, two watchers were written, the second of which takes the `agentDefaults` keys at every check, records the changed paths without their values and notes the last renewal and the last remote query, both validated on copies in both directions; plus an instrument for attributing writes. They all live in `~/claude-setup/autoclaw/diag/` and are measurement, not installation: a replica does not need them to work (compendium 8.1).

8.2 The established mechanism

- The interface renews its Z.ai session every 3,600 seconds counted from the renewal it starts with, with residuals of one or two milliseconds, and additionally on demand for causes that were not identified and that do not move the hourly grid.
- Every successful renewal writes `openclaw.json` with the new credential in the `X-Authorization` header of Z.ai's models (a class R write, paths `meta.lastTouchedAt` and `models.providers.zai.models`), and the gateway applies it live with `config hot reload applied (models.providers.zai.models)`.
- The first subsequent remote query, `[RemoteConfig] Fetched N/13 configs`, which happens every five minutes, rewrites it again changing only `meta.lastTouchedAt` (a class F write).
- Remote queries also write when they bring something new: on 2026-09-11, one rewrote Z.ai's model catalogue and two others restored `agents.defaults.model.primary` (chapter 8.3).
- An interface startup additionally writes two `zcode-runtime` plugin switches, which the gateway ignores in hot mode.

- The synchroniser carries every write over to `openclaw.linux.json` and only rewrites when the content changes; when it runs as a preparatory step of the gateway, it records in the gateway's journal and not in its own.
- The header regular expression of the interface's main process, `BREAKING_IGNORE_PATTERNS`, is not a list of reloadable fields: it is what its configuration guardian excludes when deciding whether a foreign change to `openclaw.json` breaks the connection, in which case it reinjects its credential into the gateway at most every 120 seconds; the interface's own writes do not pass through that guardian.
- The per-write oracle is the gateway's journal, with its lines `[reload] config change detected; evaluating reload (<paths>)`, and not a sampling of file timestamps, which merges bursts into a single write.

8.3 Blind adjudication

The mechanism was submitted to predictions frozen before the data was seen. The instrument `diag/atribuye_escrituras_config.py`, frozen with the digest `2cc5491729d3bdffaa887a5496390c5e1eb08d813b3ae0a59210c9e851b420cc` and backed by a validator that gave 17 of 17 against eleven mutants (deliberately broken versions, each of which bends the prediction it names), adjudicates seven predictions: P0, an hourly grid with a one-second tolerance, whose probability of being hit by chance is 2/3600; P1, one R write per interval; P2, an F write following the R and preceded by the first query; P3, no write of any other class; P4, no line requiring a restart; C1, every write preceded by one from the synchroniser; and C2, no unknown line type. Over the first blind interval, on 2026-09-10, all seven passed (compendium 8.2).

The twenty-four-hour series ended earlier than planned for an external reason: the compositor crash of 2026-09-11 at 08:44:31 killed the interface, and the cut-off was fixed at that instant and written down before the result was known. The adjudication, in session 09, gave the following (installation log, session 09). In the literal run, P3, P4, C1 and C2 pass; P1 and P2 fail; and P0 is not adjudicated, because without the interface alive there is no grid anchor. In the run over the union of the forty-one snapshots, with 747 events, P0 passes with fifteen grid points, and P1 and P2 keep failing in the same two intervals, so the widened coverage does not explain them. The gap check counted 180 queries, no gap larger than 600 seconds and no loss of coverage.

The two failures share a cause: the anomalous writes follow remote configuration queries and not credential renewals. P1 fails because at 03:54:54.689 there was a second class R write, 1.2 seconds after the remote query of 03:54:52.860, whose paths were not the authorisation headers but the catalogue (identifier, name, context window and compatibility of several models): Z.ai went from seven models to six and the interface rewrote it. The pre-registered classification calls R every write touching `models.providers.zai.models`, so it lumps the credential renewal together with the catalogue update; the write is legitimate and it is the prediction that failed to contemplate it. P2 fails because in the 07:44:50 interval there were four F writes instead of one, the last three at 8.6, 22.0 and 2.0 seconds from remote queries, and two of them carried `agents.defaults.model.primary`, which the synchroniser restored both times. Meanwhile the watcher recorded thirty-two rewrites, all with the defaults surviving, and no request to Z.ai, which is what the mechanism claims at bottom. The structural fix identified is to split class R in two, the authorisation header and the remote catalogue, and to admit that a remote query also writes (chapter 13).

8.4 Consequences for operation

Three practical rules follow from all of the above which do not depend on the instrument's details. Own values go in `providers.local.json` and never by hand into `openclaw.linux.json`, which is regenerated, and during a burst of writes a change made in the window may be rejected without harm. The hourly renewals of Z.ai's credential and the rewrites that follow remote queries are normal and the gateway absorbs them live. And the question of which model served a turn is answered with the gateway's trace, never with the interface's log.

9. Voice of the replies and dictation

9.1 Why pieces outside the window were needed

The window's code neither captures a microphone nor synthesises voice: `getUserMedia`, `MediaRecorder`, `SpeechRecognition` and `speechSynthesis` appear zero times across its nineteen bundles. OpenClaw's gateway, by contrast, ships

speech synthesis in the `messages.tts` section, off by default, with the providers `microsoft` (Edge's online voice through `node-edge-tts`, free and without a key), `xiaomi` (MiMo-V2.5-TTS) and others requiring a key (installation log, "Voice of the replies"; cited from the code on 2026-09-11). From that follows the division of labour: the voice of the replies is generated by the gateway and played by a player of our own, and dictation lives outside AutoClaw and pastes text into whichever field has focus, which may be AutoClaw's or any other.

9.2 The voice of the replies

The voice is configured in two places. The `messages.tts` section of the own configuration file sets the provider `microsoft` and the voice `es-AR-ElenaNeural`, and the synchroniser merges it key by key with whatever the interface writes. Whether it always speaks is decided by `~/.openclaw-autoclaw/settings/tts.json`, with `auto: always`, which is changed by typing `/tts off` or `/tts on` in the window; that file does not store the voice, and the Edge provider does not read environment variables, so without the configuration section the voice is the default English one.

The gateway delivers the voice as a message of its own, with the text "Audio reply" and an audio attachment in `media/outbound/`, and the window discards it, because it only renders structured attachments in the user's messages. Fixing that inside the window would require patching its package, and instead `bin/autoclaw-voice-player.py` was written, which watches `media/outbound` with inotify, on the close-after-write and moved-in events, plays each new voice file with `pw-play` once and in order, ignores those already present and never deletes, because the gateway prunes its media after 120 seconds. On the bench, in simulation mode, it played the three expected voices and ignored the pre-existing one, the foreign name and the rewrite; in the real run it played 4.60 seconds for an audio file of 4.56. Its unit starts and stops with the gateway without restarting it. The oracle that the applied voice is the configured one cannot be the file, because Edge is not deterministic in its bytes: over the decoded sound, the synthesis with the configuration correlates 1.000 in envelope with an explicit synthesis of Elena and 0.614 with the English voice, and lasts 5.04 seconds as Elena against the English voice's 5.81.

That the sound really leaves the device was measured on 2026-09-20, recording the sink's monitor and never the microphone. A short phrase synthesised by the gateway's console, delivered to the watched directory, gave a single playback with status zero in the player's journal, and the capture returned the same signal as the file, with a peak of −5.72 dBFS against the reference's −5.71, while the control taken in silence immediately beforehand stayed in exact digital silence, without a single non-zero sample in twenty seconds. The verdict is given not by the level, which any notification would reach, but by the identity: the reference's envelope correlates 0.9983 with the capture against 0.7527 for its own null. What that check lacks is a matter of ear and not of instrument: that the owner confirm he heard it complete and only once (chapter 14.5).

9.3 Dictation: design

Dictation was designed so one can speak to the agent without anything listening to other people's conversations, and that is why it is push-to-talk, with no open microphone and no wake word: one press of the shortcut starts recording, the second ends it, transcription is local and the text is pasted into the focused field without sending Enter (installation log, "Dictation, installed and measured"). Recording is `pw-record` at 16 kilohertz in mono. Insertion uses `wl-copy` and the Control+V combination through `ydotool`, because with the Latin American keyboard layout typing letter by letter with `ydotool`'s codes would break the accents. `/dev/uinput` has a `uaccess` access control list for the seat's user, so `ydotoold` runs unprivileged. The server, `dictado.py serve`, keeps the model loaded and serves the client, `dictado.py toggle`, over a local socket at `$XDG_RUNTIME_DIR/dictado.sock`.

9.4 Engine and calibration

The engine was chosen by measurement. Twelve audio files of known text, three phrases with vocabulary typical of the intended use in four Río de la Plata Spanish Edge voices, seven to eight seconds each, were transcribed with four configurations, and the word error rate was computed, that is, the proportion of words that have to be substituted, deleted or inserted to reach the correct text.

Table 7

Accuracy and delay of four Whisper configurations over twelve audio files of known text

Engine, model and device	Mean error	Worst case	Median per phrase	Load
faster-whisper `small`, processor	11.0 %	22.7 %	2.85 s	1.2 s
faster-whisper `medium`, processor	9.8 %	22.7 %	6.19 s	1.6 s
faster-whisper `large-v3-turbo`, processor	9.6 %	13.6 %	6.39 s	1.9 s
openai-whisper `turbo`, RX 6800M	9.2 %	13.6 %	2.24 s (the first, 10.3)	8.1 s and 4.9 GB of graphics memory

Note. Measured on 2026-09-11 (installation log, “Dictation, installed and measured”). The audio files are synthetic and cleaner than a real microphone, so the rates underestimate the error with a real voice.

The dedicated card dominates in both accuracy and delay, and was chosen. The remaining errors concern accentuation of the *voseo*, such as “fíjate” for “fijate”, and figures written as digits. The first use on the card compiles compute kernels, and a warm-up with one second of silence at startup absorbs that: the measured load is 8.8 seconds and the warm-up 2.2.

9.5 The voice gate

Since the model invents text over silence (chapter 5.3), the server measures each recording’s level, decides with the Silero voice activity detector, in its sixth version and on the processor, whether there is speech, and only then transcribes, handing the engine only the stretches with speech. The detector marks a window as speech when its probability reaches 0.5 and stops marking it when it falls below 0.35, so that a weak syllable does not cut a word, and the gate demands at least 0.30 seconds of speech. When there is no speech, it does not transcribe, reports “Dictado: no se escuchó tu voz” with the seconds recorded and the level, and touches neither the clipboard nor any key (installation log, session 09; values from the code of `dictado.py`).

Calibration measured both directions of error through the live server, which is the installed path: twelve negatives (silence of one to thirty-three seconds and white and brown noise from −20 to −50 dBFS) and twenty-eight positives (the twelve audio files of known text and sixteen degraded ones, attenuated by twenty and thirty decibels or mixed with noise at −40 and −30 dBFS). The twelve negatives gave 0.00 seconds of speech and zero characters, without exception; the twenty-eight positives, between 7.30 and 9.14 seconds of speech, and all of them produced text. The separation is complete, and the 0.30-second threshold was not fitted to that data but verified against it: it falls inside the gap, twenty-four times away from the weakest positive, and not pressed against the edge. The results are in `~/claude-setup/autoclaw/diag/voz/CALIBRACION_PORTON_VOZ.json`.

The repair made nothing measurably worse: over the twelve clean audio files, the error rate stayed at 0.0920 on average and 0.1364 in the worst case, identical to the previous ones, and the median delay fell from 2.24 to 2.15 seconds; over the sixteen degraded ones, the mean error is 0.1108 and the worst 0.2273, that of the audio attenuated by twenty decibels over noise at −30 dBFS. The detector costs 0.101 seconds in median, and a recording without speech is resolved in 0.197 seconds instead of being transcribed for one or two.

After the second keypress, the server records the time of each phase (audio load, measurement, gate, transcription, clipboard, paste and notification) and the total, which is the number one feels; it also records the root-mean-square level, the seconds of speech and the device it recorded from, because the default source changes by itself when headphones are connected or disconnected. The diagnostic commands `transcribe:<file>` and `probe:<file>` walk the same path as a dictation, gate included; over silence they answer `[no speech: 0.0 s over 9.8 s of audio]` within a tenth of a second. An end-to-end cycle without microphone or keyboard, with a PipeWire null sink as source, gave over an empty recording a level of −240.0 dBFS, zero characters and a total of 0.11 seconds, and with audio played to the virtual device the gate detected 6.32 seconds of speech and let it through. The two journal lines of 2026-09-11 at 18:19 and 18:20 with −240 dBFS and 0.11 seconds match that test cycle in time and figures, and are not real dictations.

The internal microphone’s gain stays at 23 %, which in raw terms is 14906 and amounts to −38.59 dB, over an analogue gain of the sound card, an ALC294 codec, of 51 steps out of 63, that is, 21.00 dB. With that gain, the microphone delivers a root-mean-square level close to −20 dBFS with peaks at full scale and transcribes well; at 100 %, the level rises to −2.3 dBFS, 22.557 % of the samples sit at full scale and transcription degenerates into repeating one phrase four times.

9.6 Privacy

The dictation voice does not leave the machine, because transcription is local; the pasted text goes wherever the focused field sends it, and if that is AutoClaw, to the model's provider. The text of every spoken reply, by contrast, travels to Microsoft's service to be synthesised.

There was one privacy failure which is declared here because it is a real risk for anyone replicating the measurements. On 2026-09-11, two six-second recordings taken to measure the microphone's noise floor captured conversation in the room, and transcription made it legible. The three files were deleted as soon as this was noticed, before anything was written about their contents, and ambient sound was never recorded again: later tests used synthetic silence, synthetic noise, the calibration audio files and a virtual audio device. The rule that remains is that the microphone is not opened for measurement without first warning whoever is in the room.

The `keep_recording` option of `~/.config/dictado/config.json` keeps the audio and the measurements of each dictation in `~/.cache/dictado/diagnostico/`. It was on in order to diagnose the empty recording and returned to `false` on 2026-09-20, with the three recordings of 2026-09-13 deleted without being opened; the journal still records the device, the root-mean-square level and the seconds of speech, which is what distinguishes a stream with no samples from a quiet microphone. Since the server reads its configuration only once at startup, the change takes effect from the service's next start. How to turn it off is in chapter 12.6.

9.7 Cost in graphics memory

The dictation server occupies some 4.9 GB of the dedicated card's memory while running, measured on the bench of 2026-09-11. On 2026-09-14 at 08:41, with the service loaded and AutoClaw closed, the dedicated card had 4,273,455,104 bytes occupied of 12,868,124,672, that is 33 %, in a driver reading that does not break it down per process; the integrated one, which drives the display, had 484 MiB occupied of 512. The practical consequence is that a game or a computation needing the whole card requires stopping the service first with `systemctl --user stop dictado.service`. On disk, the model occupies 1.62 GB.

9.8 Thermal cost and the abrupt shutdowns

On 2026-09-13, between 18:21:32 and 18:21:57, the machine shut down abruptly while voice was being recorded for AutoClaw. To find out whether transcription could cause it, a pre-registration was written before anything was run (thermal pre-registration). The hypothesis was that dictation transcription on the dedicated card, under the conditions of that moment, can freeze the machine. The conditions reproduced were the dictation server with the model loaded receiving `transcribe: <file>`, AutoClaw open through its launcher, a VP9 video of 1920 by 1080 at 30 frames looping in Firefox with hardware decoding, continuous microphone capture, the performance power profile with the machine plugged in, `amdgpu.runpm=0`, and the disk `/mnt/kingston` unmounted so that another cut would not dirty it. The sample size was derived from the 67 previous inferences without a freeze: if the shutdown was caused by number 68, the point rate is 1/68, and seeing at least one freeze with probability 0.95 requires $\ln(0.05)/\ln(1 - 1/68) = 202.3$ transcriptions, rounded to seventeen full rounds of twelve recordings, 204; with the declared limitation that this rate comes from a single event. It aborted if the card reached any of the critical thresholds its own driver declares: 100 °C at the edge, 100 °C at the junction and 105 °C in memory.

The result was verdict R4, thermal abort, applied literally. The conditions were verified and the three controls came out green: the card was working (median maximum load 99 % and 158 W per transcription, against 0 % and 6 W without transcription), the path transcribed correctly (median error rate 0.0714 and maximum 0.1364) and the conditions were in place. At 20.2 seconds of continuous transcription, on the tenth transcription, the junction reached 100 °C at 161 W, with no freeze and no new kernel message (thermal summary). Since only ten transcriptions were made, the 95 % upper bound on the per-transcription freeze probability is 0.259, which excludes nothing; the 204 planned would have taken it to 0.0146. The hypothesis was left without a verdict.

What was measured is useful all the same. The junction temperature, which is that of the chip's hottest point, follows power almost instantly: between 93 and 100 °C at peaks of 155 to 161 W and between 70 and 77 °C in the 85 W stretches, in samples half a second apart, while the edge only rose from 53 to 65 °C, with a gap of 33 to 36 °C at the peaks. The first transcription, with the card cold, reached 91 °C, so an isolated dictation does not reach the critical point. Without transcription, with AutoClaw and the video running, the dedicated card sat at 0 % and 6 W, so neither AutoClaw nor the

video loads it; the processor, by contrast, registered up to 92 °C without the conditions, 95 °C with them and 96 °C with transcription. The instrument had one defect: the cool-down phase is skipped when there is an abort request, and it was replaced with three manual readings (junction 55 °C, processor from 85 down to 69 °C, fan from 3,800 to 4,600 revolutions per minute).

The reason for the shutdown was recorded by the hardware itself, and it does not point to temperature. On booting, the kernel printed `x86/amd: Previous system reset reason [0x08000800]: an uncorrected error caused a data fabric sync flood event`, that is, an uncorrectable error caused a synchronisation flood of the bus, which is how the platform halts in the face of a hardware error it cannot contain. The same reason appears twice on 2026-06-29 and once on 2026-07-21, and thermal tripping, which has codes of its own, appears in no boot since 2026-06-20 (measured 2026-09-14 in the kernel journal). Walking the kernel journal of every boot since that date, there is a single error message from the card or the bus: `amdgpu 0000:03:00.0: device lost from bus!`, on 2026-06-29 at 19:08:15, under local inference load on the dedicated card, followed by the first shutdown of that day some thirty seconds later, according to the boot time corrected for the journal's six-hour stamp. That episode led that same day to `amdgpu.runpm=0`, which was active during the test and is active today. Before the shutdowns of 2026-07-21 and 2026-09-13 the journal records no message from the card, but that rules nothing out, because the journal loses its last seconds before a cut (on 2026-09-13 its last line is from 18:21:13, and there were files written up to 18:21:32) and persistent kernel message storage has no destination configured.

On that evidence, the thermal hypothesis is contradicted as a direct mechanism, and that of a bus error of the dedicated card under load is a candidate, supported by the sequence of 2026-06-29 and **not established** for 2026-07-21 or for 2026-09-13. It would be established by a persistent capture of kernel messages, with a destination configured for that storage or a kernel dump on the error, showing what the system said in the seconds before the next bus flood; and by a controlled reproduction with the card under load, with transcriptions at the real cadence of use, recording whether the flood follows. After the shutdown, `/mnt/kingston` failed to mount because it had been marked dirty and was repaired with the procedure of chapter 12.8; as of 2026-09-14 it is mounted read-write.

In plain terms, dictation can be used, and a single dictation leaves the card far from its limit; what takes it to the limit is transcribing without pause. This machine's abrupt shutdowns, by contrast, began before dictation and before AutoClaw, the first of them followed a crash of the dedicated card under load, and until the system saves what happens in its last seconds there is no way to know whether the card is behind the others as well.

10. The working regime inside the agent

10.1 The problem it solves

AutoClaw's agent is OpenClaw's agent running in the gateway, and in every session it receives the startup files of its workspace, `~/.openclaw-autoclaw/workspace/`, among them `AGENTS.md`. Making that agent work under a regime of instructions living outside its workspace collides with a restriction the interface itself writes into the configuration: `tools.fs.workspaceOnly` is true, and with that key the reading and writing tools are bounded to a list of roots. Three apparent ways out were discarded by measurement: a symbolic link inside the workspace pointing outside fails with "Symlink escapes sandbox root"; the official hook for injecting additional startup files resolves its paths against the workspace and goes through the same guard; and there is no configuration key pointing to an external instructions file (compendium 13.1 and 13.2; regime log, session 01). The technique serves any regime of instructions, and this machine uses it for Be Like GODmez.

10.2 The path used and its scope

AutoClaw added a mechanism of its own to the guard: `~/.openclaw-autoclaw/agent-project-read-roots.json` declares, per workspace, additional read roots, and the guard reads it on every tool call, without restarting anything. A root may be an exact file, and that finding sets the scope: exactly ten files are authorised, the adapter `~/claude-setup/godmez/AGENTS_be_like_godmez.md` and the regime's nine modules, and not the whole memory directory, which would expose every project memory to an agent connected to a remote service. The workspace comparison does not resolve links, so the declared value is literally `/home/usuario/.openclaw-autoclaw/workspace`.

10.3 The pieces

The applier `~/claude-setup/godmez/autoclaw/godmez_autoclaw_apply.py` is idempotent, contains no rules and only restores three pointers. The first is a block delimited by `<!-- godmez:be-like-godmez -->` and `<!-- /godmez:be-like-godmez -->` at the **beginning** of `AGENTS.md`, which orders the adapter and the core to be read by absolute path, sets the language in its first paragraph and carries a load check that can only be answered by reading the core. It goes at the beginning because the gateway creates the startup files without ever overwriting them and the interface's five functions that touch `AGENTS.md` append their blocks at the end, so the beginning is the only zone nothing foreign reaches; besides, `AGENTS.md` is one of the two startup files that subagents also receive. The second pointer is a skill, `<workspace>/skills/be-like-godmez/SKILL.md`, which repeats the notice by way of the skills list. The third is the read authorisation for the ten files, preserving any entry the interface may have placed for another workspace.

Without arguments, the applier exits with 0 and says whether it restored anything; with `--check`, it exits with 0 if everything is in place and with 1 if not; and it exits with 2, writing nothing, if any of the ten files or the workspace is missing. Two units of its own, `godmez-autoclaw-regimen.path`, which watches `AGENTS.md` and the authorisation file, and `godmez-autoclaw-regimen.service`, which runs the applier, restore everything when something overwrites it. They are needed because the interface rebuilds the authorisation when starting the gateway and empties it completely when unlinking an external folder; since the applier does not write when nothing changed, the watching converges in two triggers.

10.4 How it was verified

Before touching anything, ten scenarios were tested on a bench, with a single block, idempotence and AutoClaw's content preserved byte for byte in all of them. Restoration was measured by provoking the failure: with the authorisation emptied by hand, it returned to its sum in under four seconds, and with the block deleted, it came back exact. Across six agent turns, the oracle was not what the agent said it had read but what the session transcript records in `~/.openclaw-autoclaw/agents/main/sessions/<id>.jsonl`, and the difference mattered, because on one round the answer mentioned no reading at all and the transcript showed eight read calls. The test question was designed so that it could not be answered from the block, and the agent returned the date and the verbatim quotation that exist only in the core. Three defects of the block were measured and repaired: the language stated inside an enumeration, which the agent quoted and disobeyed; a load check that could not fail because the block itself carried the answer; and a criterion of "substantive work" that depended on the size of the request and not on what the answer carries inside (compendium 13.5 and 13.6).

The block survives a full interface cycle on its own: across five samples, from before closing to ten minutes after reopening, the sum, the modification time to the nanosecond and the inode came out identical, and the restoration unit did not even fire. The methodological lesson holds in any file audit: a constant inode rules out an atomic replacement but not a write in place, so the indicator that rules out every write is the modification time. The context budget came to some 6,150 characters of `AGENTS.md` against a cap of 20,000 per file, and some 10,300 across all startup files against a cap of 60,000.

10.5 Costs and state

The regime's cost inside the agent is one of context and of privacy: everything the agent reads travels to the model's provider, today MiMo. By decision of 2026-09-10, the identity and profile module remains among the ten readable files, against the recommendation to remove it; if one wished to exclude it, it is taken off the applier's list and the next pass withdraws it. On 2026-09-14 at 08:41, `godmez_autoclaw_apply.py --check` exited with 0. The regime's chain of sessions was closed on 2026-09-13, with no live instruction for a following session (regime log, "Closing the chain").

11. Updates

11.1 The updater under Wine and its error dialogs

AutoClaw's Electron updater kept looking for version 1.18.1 at startup and every twenty minutes, downloading the entire 389 MB installer and rejecting it at signature verification with `ERR_UPDATER_SIGNATURE_VERIFICATION_FAILED`, because that check uses PowerShell, which Wine does not provide. The rejection opened a modal error dialog, and it was the only

one: `dialog.showErrorBox` appears exactly once in the interface's entire main process, read in `out/main/index.js` inside `app.asar` (compendium 3.4; measured and cited from the code on 2026-09-10). Two other errors in the log open no dialog: the one about automatic startup, because Wine has no `schtasks`, and the one about the "OSS 0.1.6" browser extension, which finds neither Chrome nor Edge in the prefix.

11.2 The manual channel

The update address is assembled from the channel the interface reads from `resources/channel.json`, and that channel travels to the service only in the `X-Update-Channel` header, deliberately separate from the business header; changing it touches neither models, nor credits, nor account. The updater's error handler treats a 404 as "nothing new" and stays quiet. From that follows the repair, which is configuration foreseen by the vendor and not a patch: `{"channel": "wine-manual"}`, a channel Z.ai's server does not have and which answers 404, measured before applying it. Since then there has been no further "Found version 1.18.1", no downloads and no signature verifications, and the pending directory has stayed empty; the 407,685,392-byte installer the updater had left behind was deleted after checking by sum that it was the 1.17.8 one. To return to the official channel it is enough to delete the file, and the business channel, which is a different file, remains at `official`.

11.3 Version 1.18.1, audited and not migrated

The 1.18.1 installer measures 401,789,880 bytes, its MD5 sum matches the server's ETag and its SHA-512 matches the one in `latest.yml`, its SHA-256 is `f358ec32fb39c9e61bc9f61e70daea13cec3bc7b98fdd668f6dce93fc3d8360d`, and its signature verifies with the same signer and a DigiCert timestamp of 2026-09-07. The six native module versions it expects are identical to 1.17.8's, so the portable bundle's Linux modules still serve; the gateway declares the same OpenClaw 2026.6.8 and the same Node version requirement, although its bundle differs by 16,999 bytes (compendium 6.5; measured 2026-09-10). By decision of 2026-09-10 it is not migrated until there is a concrete reason, because migrating forces restoring and revalidating with no measured gain.

11.4 What an update erases and what it keeps

Table 8

Effect of an AutoClaw update on each piece of the integration

Piece	Effect of the update	How it is restored
`resources/channel.json`	erases it, because the installer rewrites `resources/`	it is written again, checking first that `wine-manual` still answers 404
Linux native modules in `resources/gateway/openclaw/node_modules`	erases them	the startup script restores them from `native-stage` if the versions did not change; if they did, `npm install --no-save` of the Linux variants
patched extractor	replaces it	`autoclaw-pdf-notice-patch.py`; if its `--check` gives 2, the version changed the extractor and the patch is derived again
`gateway-bundle.mjs` and `dist/openclaw-tools-Bm2s_FnJ.js`	replaces them	`autoclaw-page-range-patch.py`; if its `--check` gives 2, the version changed the texts and the patch is derived again
Wine shortcuts	may regenerate them	they are pointed at the launcher again
`~/.local/share/autoclaw-linux/`, with scripts, own configuration file and Python environment	does not touch it: it is outside the prefix	nothing
user units and official Node	does not touch them	nothing
state `~/.openclaw-autoclaw`, with `openclaw.json`, sessions and workspace	lives outside `resources/`; that the installer does not alter it was not measured	the regime restores itself through its unit; the custom model is checked in the window

11.5 Migration procedure

If a migration is ever done, this is the order. Back up `~/.openclaw-autoclaw` and the integration's folder. Verify the new installer as in step 2 of chapter 4, with size, ETag, sum and signature. Before installing, read the Windows native module versions it ships and compare them with `native-stage`. Install silently, recreate `channel.json` and check the channel, restore the modules if they changed, check the shortcuts and open with the launcher. Run the patch applier with `--check`: if it gives 1, apply it; if it gives 2, stop the patch migration and derive it again. And repeat the battery of step 8 (internal diagnostics, Office 80 of 80, web search, browser, tasks, idle shutdown and restart with reconnection), plus the trace of one turn towards MiMo, the defaults comparison and the regime's `--check`.

12. Daily operation and traps

12.1 Opening, closing and exiting

AutoClaw is opened from the applications menu, whose shortcut runs the launcher, or from a terminal of the graphical session with `~/.local/share/autoclaw-linux/bin/autoclaw-launch.sh`. It is never launched as a transient systemd service, because `wine` returns on handing over the shortcut and systemd would remove the control group with the interface inside it; from another session, a detached scope is used:

```
systemd-run --user --scope -- setsid -f ~/.local/share/autoclaw-linux/bin/autoclaw-launch.sh
```

Closing the window sends the application to the tray with its processes alive, and the gateway stays up; exiting from the tray menu terminates the main process, and the timer, which runs every three minutes, stops the units on its next pass (in the measurement of 2026-09-10, at thirty seconds). An orderly window close request does not terminate the process.

12.2 Health checks

```
systemctl --user is-active autoclaw-gateway autoclaw-port-fence autoclaw-office-bridge
#   active, three times, with AutoClaw open; inactive with AutoClaw closed
curl -s http://127.0.0.1:18789/health          # the gateway's health response
curl -s http://127.0.0.1:18860/v1/health       # the Office bridge's health
ss -tn state established | /usr/bin/grep -c ':18789'
#   2: window connected; 0 with the window open: press "Reconnect"
journalctl --user -u autoclaw-gateway -f       # the gateway's journal
python3 ~/.local/share/autoclaw-linux/bin/autoclaw-pdf-notice-patch.py --check; echo $?
python3 ~/.local/share/autoclaw-linux/bin/autoclaw-page-range-patch.py --check; echo $?
#   0 patched, 1 original, 2 foreign file
~/claude-setup/godmez/autoclaw/godmez_autoclaw_apply.py --check; echo $?
#   0 in place, 1 out of place, 2 a file is missing
ps -eo pid,lstart,etimes,args | /usr/bin/grep '[A]utoClaw.exe'
#   a process born BEFORE the living main one is a leftover
```

A visible window does not prove a connected window: the connection oracle is `ss`, not the picture.

12.3 Restarting the gateway

The only correct restart with the window open is `~/.local/share/autoclaw-linux/bin/autoclaw-gateway-restart.sh`, which kills the gateway with SIGKILL so that the WebSocket close is abnormal, systemd relaunches it in about a second and the interface retries on its own. `systemctl --user restart autoclaw-gateway` closes the WebSocket with code 1012, the interface takes it as an expected close, disables its reconnection and stays disconnected until someone presses "Reconnect"; on 2026-09-10 that left the window without a connection for ten minutes. After applying or removing the extractor patch, or changing the voice, the restart is done with the script.

12.4 Models, credits and configuration

```
# which model served each request: the consumption gate
journalctl --user -u autoclaw-gateway.service --since -15min --no-pager \
  | /usr/bin/grep -oE 'AutoClawRequestTrace\] (request|response)[^|]{0,170}' | tail -8

# the own defaults against the translated ones: IGUAL on all four keys
python3 - <<'EOF'
import json, os
src = json.load(open(os.path.expanduser('~/.local/share/autoclaw-
 ↪  linux/providers.local.json')))['agentDefaults']
lin = json.load(open(os.path.expanduser('~/.openclaw-
 ↪  autoclaw/openclaw.linux.json')))['agents']['defaults']
for k in src:
    print(k, 'IGUAL' if src[k] == lin.get(k) else 'DISTINTO')
EOF

# every configuration write, with its paths, according to the gateway
journalctl --user -u autoclaw-gateway.service --since today --no-pager -o short-iso-precise \
  | /usr/bin/grep -F '[reload]' | tail -20

# AutoClaw's credit balance, with the time in coordinated universal time
/usr/bin/grep -F 'Wallet v2 display items' ~/.openclaw-autoclaw/logs/autoclaw-compat.log | tail -3
```

Own values are edited in `providers.local.json` and never by hand in `openclaw.linux.json`, which is regenerated; after editing, `bin/autoclaw-config-sync.py` is run and, if the change is not applied live, the gateway is restarted with the script.

12.5 Documents

The document tool accepts only paths inside its permitted directories, so documents passed to it from the console go in `~/.openclaw-autoclaw/workspace/`. If the agent's answer begins with a `[PDF notice: …]` line, the document arrived truncated and the notice states with what `pages` range the rest can be read. Since the patch of 2026-09-20, any page is reachable by asking for ranges of up to `pdfMaxPages` pages, today sixty, and a larger selection fails with an error naming the amount requested and the limit (chapter 7.5).

12.6 Voice and dictation

```
# voice of the replies: in the window, type /tts off or /tts on
systemctl --user disable --now autoclaw-voice-player.service   # remove the player
pactl get-sink-mute "$(pactl get-default-sink)"                # Mute: no, in order to hear it

# dictation state and phases
systemctl --user is-active dictado.service ydotoold.service
journalctl --user -u dictado.service --since today --no-pager -o short-precise \
  | /usr/bin/grep -E 'recording from|s of audio|phases'

# before gaming or a computation needing the whole card
systemctl --user stop dictado.service

# test a file along the real path, gate included, without touching the keyboard
/usr/bin/python3 ~/.local/share/dictado/dictado.py transcribe:<file>
/usr/bin/python3 ~/.local/share/dictado/dictado.py probe:<file>

# check that the voice really leaves the speaker, without opening the microphone
#   a short phrase sounds in the room; it ends with 0 and leaves its report in JSON
python3 ~/claude-setup/autoclaw/diag/voz/mide_salida_audio.py
```

```
# microphone in use, volume and mute (the correct volume is 23 %)
S=$(pactl get-default-source); echo "$S"; pactl get-source-volume "$S"; pactl get-source-mute "$S"

# turn off the keeping of recordings
cp ~/.config/dictado/config.json ~/.config/dictado/config.json.bak-$(date +%F)
python3 - <<'EOF'
import json, os
p = os.path.expanduser('~/.config/dictado/config.json')
d = json.load(open(p)); d['keep_recording'] = False
open(p, 'w').write(json.dumps(d, indent=2) + '\n')
EOF
systemctl --user restart dictado.service
```

A “Dictado: no se escuchó tu voz” notice means the microphone delivered no speech, and the journal line states at what level and from what device; it does not mean the engine got it wrong.

12.7 The gateway’s console

```
export OPENCLAW_STATE_DIR=~/.openclaw-autoclaw \
       OPENCLAW_CONFIG_PATH=~/.openclaw-autoclaw/openclaw.linux.json
cd ~/.wine/drive_c/"Program Files"/AutoClaw/resources/gateway/openclaw
~/.local/opt/node-v24.18.0-linux-x64/bin/node --no-warnings gateway-bundle.mjs <command>
# verified commands: doctor --lint --non-interactive, browser start|open|tabs|snapshot|stop,
# infer web search --provider autoglm, cron add|list|rm, skills list --agent main,
# infer image describe, infer tts status --gateway
```

`cron list` hides disabled tasks unless `--all` is given. Each console invocation “migrates” the inherited mirror `cron/jobs.json` to `jobs.json.migrated.N` without the SQLite database losing anything. A turn against Z.ai through this console returns 400, because it does not send the session the intermediary demands, and that says nothing about the window.

12.8 Measured traps, with their lesson

Each trap is stated with what it costs to fall into it and with the rule that avoids it. All of them were measured on this machine.

- **Restarting with `systemctl restart` while the window is open** leaves the window disconnected until “Reconnect” is pressed; restarting is done only with the script (chapter 12.3).
- **A leftover rendering process** from an instance terminated by a signal keeps the timer from ever shutting the gateway down, because it counts processes by name without looking at which instance they belong to. It is recognised because it was born before the living main process, carries a different `--field-trial-handle`, has no window and its processor time does not grow.
- **Recording a sink’s monitor with `pw-record --target`** does not record the monitor: the session manager ignores that target and links the stream to the default source, which here is the internal microphone, so a measurement intended not to open the microphone ends up recording the room. The stream is opened with `-P '{ stream.capture.sink= true }'` and, since a flag states what was asked for and not what happened, one checks in `pactl list source-outputs` which source it actually landed on. With the monitor correctly taken, a sink with nothing playing delivers exact digital silence, and that is what distinguishes a good capture from one that carries the room.
- **`pw-record --sample-count N`** exits with code 1 while writing the complete file and leaves standard error empty. The oracle of a capture is the file’s frame count, never the exit code.
- **A terminal born inside VS Code** inherits `ELECTRON_RUN_AS_NODE=1`, and any Electron application launched from there exits with code 0 with no window and no log. Before declaring a graphical application launched from a terminal dead, look at `env | /usr/bin/grep ELECTRON`.
- **Fedora’s `nodejs24`** dies with code 139 when segmenting text; faced with a Node process dying without a trace, first run the one-line reproducer of step 7.

- **`openclaw.linux.json` is regenerated**: a change made by hand is lost, and during a burst of writes a change made in the window may be rejected with `config changed since last load`; on 2026-09-10 there were thirty `rate limit exceeded for config.patch` rejections in six minutes, without harm.
- **The "Diagnostics" button of "Models and API"** warns in its own text that it may delete custom models added by hand, and it is not pressed.
- **A session bus call to the global shortcuts service** aborted the compositor. On 2026-09-11 at 08:44:31, in order to assign the dictation shortcut, `gdbus` was used to call `org.kde.KGlobalAccel.setForeignShortcutKeys` with the signature declared by introspection; in Plasma 6 that interface is served by `kwin_wayland` itself, which died with SIGABRT. The wrapper relaunched it in two seconds and the session survived, but AutoClaw and Firefox died, and the series of measurements in progress was cut off at that instant. The cause was procedural: a never-tested call was executed against the live compositor, with an argument whose shape was known only by introspection, when a risk-free path existed. A global shortcut is assigned by hand in System Settings, and write methods of `org.kde.kglobalaccel` or of `org.kde.KWin` are not called with the session open (KWin memory).
- **A nested `kwin_wayland`** shares the bus and the configuration with the real compositor and leaves its shortcuts component inactive, so that Alt+Tab and every KWin shortcut stop responding until a restart; if it is ever used, it goes with its own `XDG_CONFIG_HOME` and `dbus-run-session`.
- **`xdotool` under Wayland** moves, resizes and closes windows, but neither types nor clicks, because the compositor does not deliver synthetic events to XWayland. The gate is `xdotool mousemove X Y` followed by `xdotool getmouselocation`: if the position does not change, any test requiring typing into the window is done by a person.
- **`ydotool` delivers the key to the focused window**, so no automated dictation test presses keys with the user's session open.
- **Whisper `large-v3-turbo` writes a phrase for any input**, silence included; a level of −240 dBFS is an absence of samples and not weak audio, and every diagnostic command has to walk the same path as real use, gate included, because one that skips it returns the invented phrase and reads as though the repair were not in place.
- **The microphone gain at 100 %** leaves 22.6 % of the samples at full scale and transcription degenerates into repeating one phrase; it stays at 23 %.
- **The default audio source changes by itself** when headphones are connected or disconnected; that is why the dictation journal records which device it recorded from.
- **Edge's synthesis is not deterministic in its bytes**: two syntheses of the same text differ, and the voice is compared by the envelope of the decoded sound.
- **Recording the room to measure captures conversation**: the microphone is not opened for measurement without first warning whoever is in the room (chapter 9.6).
- **The interface's logs rotate** on reaching one mebibyte with a single old generation, and `autoclaw-auth.log` has lines with credentials, so it is not read by hand.
- **The credit balance also rises through promotions** (from 4,892 to 5,292 at 14:19:43 on 2026-09-10 without anyone doing anything), so the consumption oracle is the gateway's trace.
- **The `grep` of the interactive shell is a different program** on this machine; the scripts and the oracles use `/usr/bin/grep`, and a pattern with backslashes is searched as a fixed string with `-F`.
- **bash's `printf` under the Spanish locale** rejects the decimal point, prints the integer part and lets the script carry on; every script that formats figures exports `LC_NUMERIC=C`.
- **`pgrep -f` and `pkill -f`** match the terminal itself if the pattern appears in its command line; one `pkill -f` killed its own terminal with exit 144.
- **A `nohup … &` launched from an assistant's terminal** may die without a trace; a process that must survive is launched with `setsid` and checked with `ps -o pid,ppid,sid` to confirm its session is its own.
- **The journal stamps each boot's first entries six hours earlier**, because the hardware clock keeps local time; whether there was a reboot is read from `uptime -s`, and the journal additionally loses its last seconds before a cut, so the time of a cut is bounded by the modification times of files.
- **`journalctl -k` walks only the current boot**; to look for kernel messages from earlier boots use `journalctl _TRANSPORT=kernel --since …`.
- **After an abrupt shutdown, `/mnt/kingston` fails to mount** because ntfs3 refuses a volume marked dirty in write mode. The procedure measured on 2026-09-13 was to mount read-only, verify by reading the raw device that the filesystem journal was entirely `0xFF`, back up the regions, run `ntfsfix -d`, which changed exactly seven bytes, and sweep the contents by status change time and not by modification time (shutdown memory).
- **Counting the regime's block with `grep -c 'godmez:be-like-godmez'`** gives two with a single block, because it counts the opening and the closing.

- **The dictation shortcut appears as `net.local.python3.desktop`** in `~/.config/kglobalshortcutsrc`, and not under the launcher's name, because System Settings names the component after the command.

13. Open decisions and pending structural fixes

Each item was re-verified on 2026-09-14 before being listed, because an inherited pending item is a claim that may have ceased to be true, and two of them had. The rows moved by the work of 2026-09-20 carry that date.

Table 9

Open and closed items, with their state verified on 2026-09-14, or on 2026-09-20 where indicated, and their recommendation

Item	Verified state	Recommendation
Definitive voice of the replies	`es-AR-ElenaNeural`, provisional	none on technical grounds: it is a matter of ear; Microsoft's Río de la Plata alternatives are `es-AR-TomasNeural`, `es-UY-ValentinaNeural` and `es-UY-MateoNeural`
Muted audio output	**closed**: `Mute: no`	nothing; the voice is heard on opening AutoClaw
`pdfMaxPages`	**closed**: stays at sixty	no raise: `parsePageRange` repaired, the limit is a per-call budget and any page is reachable by ranges
`--` in the `taskset` line of `~/claude-setup/cpu/en.sh`	**closed**: repaired on 2026-09-13 at 09:13	nothing; it does not belong to AutoClaw
`bin/autoclaw-set-provider-key.sh`	**closed**: corrected on 2026-09-20, with a bench	nothing
`parsePageRange`	**closed**: patched on 2026-09-20 across its three surfaces, with a bench, controls and integration measured against the real extractor that same day	nothing; only the interactive turn is missing, which consumes the plan
Class R of the attribution instrument	not split	split it in a successor, only if it is measured again
Dictation's `keep_recording`	**closed**: `false` since 2026-09-20 and recordings deleted	takes effect from the service's next start, which was not forced
Thermal run at the real cadence of use	proposed, not run	run it only with persistent kernel message capture and with the 100 °C abort intact
Latency of the headphones' hands-free profile	not measured	measure it when headphones are used
End-to-end acoustic test	**closed**: measured on 2026-09-20 on the sink's monitor and repeated on 2026-09-21 with another phrase, both with signal identity and a control in digital silence, and the owner confirmed by ear that it was heard complete and only once	nothing; the voice chain is verified end to end, from the synthesiser to the ear
Ordinary exit through the menu and leftovers	**closed**: measured on 2026-09-20 at 19:32; no process survives the exit and the five named units plus the voice player are left stopped	nothing; exiting through the menu is clean and the automatic shutdown does its job
Predicate of the automatic shutdown	**closed**: the `pgrep -f` defect was measured on 2026-09-20 and the successor was installed on 2026-09-21, with a dated backup of the previous one and a smoke test against real systemd	nothing; it compares `argv[0]` by equality and refuses to shut down if the executable is not on disk
Oracle of the leftovers instrument	**closed**: it waited for the timer to fire and photographed inside the shutdown, which gave a false verdict on 2026-09-20; repaired that same day, with a bench of thirteen checks	nothing; the oracle became the effect, which is the timer going inactive

Item	Verified state	Recommendation
End-to-end test with an eighty-page document	**closed**: on 2026-09-21 the owner ran the interactive turn and the agent returned `MARCA 080: 835dba07`, identical to the manifest's and unique among the eighty, with the trace against `token-plan-sgp.xiaomimimo.com` and the models `mimo-v2.5-pro` and `mimo-v2.5`	nothing; the chain was measured from the document to the model, and the reply additionally sounded through the speaker
Migration to 1.18.1	not migrated	wait for a concrete reason

Note. Every pending decision belongs to the machine's owner; none is applied without his agreement.

The definitive voice. The provisional voice was chosen for being Río de la Plata Spanish, and changing it means putting another value in `voice` inside `messages.tts` of the own configuration file, synchronising and restarting with the script; the oracle is the envelope of the decoded sound. MiMo's voice, which the token plan includes without consuming credits for a limited time according to the provider's page, offers only predefined voices in Chinese and English, and its `default_en` voice returned 400.

The muted output and `en.sh`. The logs give both as pending. The audio output was muted on 2026-09-11 and on 2026-09-14 it is not; `en.sh` already runs `exec taskset -c "$CPUS" "$@"`, without the `--` that made `taskset` take the marker as the program's name and exit with 127, with a comment dating the measurement to 2026-09-13. They are taken as closed according to the measurement.

`pdfMaxPages`, and why it did not rise to five hundred. The earlier recommendation proposed five hundred so the agent could read books, on the assumption that the limit fixed the last reachable page. That assumption was false: the engine applies it as an amount per call, and what discarded the later pages was `parsePageRange`. With the function repaired, any page is reachable through ranges of up to sixty, raising the value would only enlarge the text sent from a sparse document and its cost against the MiMo plan, and the limit keeps its function as a budget. It stays at sixty.

The key script. Corrected on 2026-09-20 in both its defects: it wrote at the top level of the file, when providers live inside `providers`, and it printed the first three characters of the key, which contradicts its own purpose. It additionally accepts a second argument with the file's path, without which it could not be tested end to end without touching the real key (chapter 14.4).

`parsePageRange`. Repaired on 2026-09-20 across all three surfaces, with a sealed applier and a bench (chapters 7.5 and 14.4). That same day the integration was measured against the installed extractor and the real engine, with a negative control over the pristine package (chapter 14.5), so the only thing pending is the interactive turn from the window, which consumes the plan and requires a person.

The acoustic test. Measured on 2026-09-20 without opening the microphone and without calling any model: the sink's monitor picked up the phrase with the same peak as the synthesised file, the control stayed in digital silence and the signal's identity separated the playback from any other possible sound (chapter 14.5). What remains is the half no instrument answers, that the owner say whether he heard it complete and only once.

Leftovers from an exit through the menu. The script `diag/mide_remanentes_salida.py` was written on 2026-09-20 and waits internally for the timer to fire, which has a three-minute period and thirty-second accuracy, because looking immediately would confuse a shutdown in progress with a leftover. It is run with `antes` with the window open, one exits through the tray menu, and it is run with `despues`. Producing that exit requires pressing, and under Wayland synthetic input does not reach windows, so the test belongs to the owner.

`keep_recording`. Turned off on 2026-09-20 and the three recordings deleted without being opened. The server reads its configuration only once at startup, so the change takes effect from the service's next start; it was not restarted because the bus flood of 2026-09-14 happened while `dictado.service` was being stopped with the model resident on the card.

The thermal run at the real cadence. The test of 2026-09-13 measured the two extremes, an isolated dictation and continuous transcription, and left real use unmeasured. A run at the cadence of real dictations would answer whether normal use comes close to the critical point. Since the journal loses its last seconds before a cut and persistent kernel message storage has no destination configured, the run only contributes evidence about the shutdowns if that capture is configured first; raising the abort to the emergency threshold of 105 °C would loosen the pre-registration, and is not recommended.

14. State verified at the time of writing, paths and digests

14.1 The oracles of 2026-09-14

The oracles were run during the morning of 2026-09-14, from 08:41:20, with the machine booted at 08:29:19 and AutoClaw closed, and everything that follows is reading: no service was restarted, the microphone was not opened, no sound was played and no key was pressed.

Table 10

State of the installation measured on 2026-09-14; what changed afterwards is in 14.4

Check	Result
Boot and previous shutdown	boot at 08:29:19 on 2026-09-14; the previous one ended with a clean shutdown at 21:34:19 on 2026-09-13, and the current boot prints no reset reason
System	Fedora release 44; kernel `7.2.5-200.fc44` (the installation was done with `7.2.4-200.fc44`)
Wine	`wine-11.0 (Staging)`; packages `wine` and `wine-core` 11.0-3.fc44
Fedora's Node	`nodejs24` 24.18.0-1.fc44, the same build that had the defect; the gateway is still pointed at the official Node and the one-line reproducer was not run again
Other packages	`ydotool` 1.0.4-8, `xdotool` 3.20211022.1-10, `osslsigncode` 2.12-1, `chromium` 152.0.7977.82-1, `libreoffice-core` 26.2.6.3-1, `python3` 3.14.7-1
Services	`dictado.service` and `ydotoold.service` active; `autoclaw-gateway.service` and `autoclaw-voice-player.service` inactive, as befits AutoClaw being closed
AutoClaw units	seven static; `autoclaw-voice-player.service`, `dictado.service`, `ydotoold.service` and `godmez-autoclaw-regimen.path` enabled
Connections to 18789	0, with the window closed
Gateway trace	the last request to Z.ai is from 2026-09-10 at 13:16:25; after it, 49 requests, all to `token-plan-sgp.xiaomimimo.com` (47 that day and 2 on 2026-09-11), and none in the two openings of 2026-09-13
Extractor patch	`--check` with exit 0: patched
Regime inside the agent	`--check` with exit 0
Updater channel	`{"channel": "wine-manual"}`
Own configuration file	mode 600; sections `providers`, `agentDefaults` and `messages`; models `mimo-v2.5-pro` and `mimo-v2.5`; `pdfMaxPages` 60
Translated voice	`messages.tts` with provider `microsoft` and voice `es-AR-ElenaNeural`; `tts.json` with `auto: always`
Audio output	`Mute: no`
Dictation configuration	`openai-whisper`, `turbo`, `cuda`, `es`, beam of 5, insertion `paste`, `keep_recording: true` (became `false` on 2026-09-20; see 14.4)
Latest dictations	2026-09-13 at 20:00:55, 20:01:35 and 20:01:50, with totals of 1.57, 1.65 and 1.70 s
Recordings kept	three, with their measurements, all from 2026-09-13 between 20:00:55 and 20:01:50
Graphics memory	dedicated: 4,273,455,104 of 12,868,124,672 bytes occupied; integrated: 484,438,016 of 536,870,912
Card parameter	`amdgpu.runpm=0` on the kernel command line and in `/sys/module/amdgpu/parameters/runpm`
Card or bus errors in the kernel journal	one since 2026-06-20: `device lost from bus`, on 2026-06-29
Persistent kernel message storage	no destination: the `backend` parameter reads `(null)`
`/mnt/kingston`	mounted read-write with `ntfs3`
Shortcuts	the Wine menu one points at the launcher; `~/Escritorio/AutoClaw.desktop` does not exist
Dictation shortcut	`Ctrl+Alt+Space` under `[services][net.local.python3.desktop]`
Key script	still writes at the top level of the own configuration file
`~/claude-setup/cpu/en.sh`	`exec taskset -c "$CPUS" "$@"`, without `--`, modified on 2026-09-13 at 09:13

14.2 What changed with respect to the logs

Seven facts of 2026-09-14 differ from what the logs left written, and in each case the measurement governs. The audio output is no longer muted. The `en.sh` defect is already repaired. The machine went through an abrupt shutdown on 2026-09-13 at 18:21, two clean reboots that night, one of them for kernel 7.2.5, and a clean shutdown at 21:34:19. The two recordings of the 2026-09-11 test cycle are no longer in the diagnostics directory. The desktop shortcut does not exist. The real dictations of 2026-09-13 carried speech. And Z.ai's catalogue has six models and not seven.

14.3 Paths and digests

Table 11

SHA-256 digests of what is installed, measured on 2026-09-14 except where indicated

Path	SHA-256 digest or contents
`~/.local/share/autoclaw-linux/bin/autoclaw-launch.sh`	`e84e48563d62f0b03dff7ad6f29a91d02c0d4dd8bdebd3e2e28acea028ca9c79`
`bin/autoclaw-gateway-run.sh`	`ea59f4259e0941c5dc541b24599dfdaf78428b8e97994c331dac7c515f148c0b`
`bin/autoclaw-gateway-restart.sh`	`8be58c460aa907d9e502514bc67eedaa8d8b33d15e9f3c7b116bde8f39778022`
`bin/autoclaw-config-sync.py`	`124c78653b6c7ad218c380b940010b592b3a9de4f897bc75989821872f740e38`
`bin/autoclaw-port-fence.py`	`d8e74137b8e567a2eaaf8f43a0fcb37784a17309547d427033f80e2a596615cf`
`bin/autoclaw-idle-stop.sh` (successor installed on 2026-09-21; the previous one, `3698db84…`, remains in `bin/autoclaw-idle-stop.sh.bak-2026-09-21`)	`f9f0ec8d4c1d11ba35df7558193e9b381bcc233e6e191853361e038a44c79511`
`bin/autoclaw-libreoffice-setup.py`	`2e0c6aa967a248fc9bc99427ed25d585735bcde5f208913f823d8b6c5e39620c`
`bin/autoclaw-pdf-notice-patch.py`	`28d4b5a0ab6d287263bee626e87b5feb22df54250609794ccfc199d17ef581c9`
`bin/autoclaw-voice-player.py`	`e4bf96ac07e918d26bffbd9ea6486aef092b6aff3bcf1fd9180f9ce414d6fa91`
`bin/autoclaw-set-provider-key.sh` (corrected on 2026-09-20)	`7f9aa17c295af5915d0a71453a01f930ad2b47425c2ba484e78c624c0d847037`
`bin/autoclaw-page-range-patch.py`	`02732f9e9c3d6e253d6334dd5899cee8999c878c449ee6939a5abace9c4e292f`
`tests/test_set_provider_key.py`	`0c79542b1a0922e29e4a7cf6dae4e8455c11009e79bce082dc2bd33833b13ad6`
`tests/test_page_range_patch.py`	`9b1bdb3de8db623a909dbbbc6ed599483c6aed1a01ad3faa6ada03351fc7a80f`
`bin/autoclaw-provider-login.sh`	`d592de2d771e60d8a0a4f3c80171b03d7ab7add8b0ba3196b0208c21680a2603`
`bin/autoclaw-post-reboot-check.sh` (no unit)	`405262b45fcfa88400c549d33e70a9b483f5a913f62f684c2299ef547470ecfe`
`patches/document-extractor/original.js`	`8a7e77a9cab9b4019f8735f905b2a293e02d5b32d8aa93022effffc06cf542c`
`patches/document-extractor/patched.js` (as of 2026-09-14; updated on 2026-09-20 to the current extractor, with backup `.bak-2026-09-20`)	`fe307e1122ee5c62870dc8495a4850444641d6eb143efe590cabb94779d152d6`
installed extractor, `.../dist/extensions/document-extract/document-extractor.js` (as of 2026-09-14; the page-limit patch changed it, see table 13)	`fe307e1122ee5c62870dc8495a4850444641d6eb143efe590cabb94779d152d6`, equal to the patched one of that date
`.../resources/gateway/openclaw/gateway-bundle.mjs` (as of 2026-09-14; the page-limit patch changed it, see table 13)	`7389fe727d04aa235bbfdf1224d8649ce44ad2ea1f95a13c4dab43236bed13cb`
`.../resources/channel.json`	`{"channel": "wine-manual"}`

Path	SHA-256 digest or contents
`~/.local/opt/node-v24.18.0-linux-x64/bin/node`	`41a74efb34cbde5c7632cdac0cf8bd1a14d0b8d73dc1e82755014d9a9ce70f5c`
`~/.config/systemd/user/autoclaw-gateway.service`	`9aa0dcbf5dce1beee2406cc88084b06526b1860adb5d2a2eb83a8af8500d449d`
`autoclaw-gateway.service.d/10-node-oficial.conf`	`4800a3648bb8241e11cb67514dc8b2b9af91c3772eb095ee76270d317b619598`
`autoclaw-port-fence.service`	`2bbb6e41047e85b8fc45313b64091c51a23afbafb5039afeb04ea6e1e9e33669`
`autoclaw-office-bridge.service`	`401d78de362f5fa6ceadd43aa13e9b40ec5a7dc016a424857c67d213de842d47`
`autoclaw-config-sync.path`	`99b68c372f25e207576d41abaf4154201c304cc196b5e7c116ed851557e052be`
`autoclaw-config-sync.service`	`fc2552d5de6b566959f8c0d0ad7384188941ffbfd487052b14e62c89b586ea49`
`autoclaw-idle-stop.timer`	`23a8ef0d68f1ccccbd9a08f869db9649b64faf5786d7a7c8e553ac6bbc64a61d`
`autoclaw-idle-stop.service`	`ab34ca2772b6e82061666b1ccc182d01ddfe50d86cf0219205f41beb7da153e1`
`autoclaw-voice-player.service`	`763375acaf5c4f77abb85d2f43ffebe2bacd6b1b8315dadb7d8a60b0072a3e65`
`dictado.service`	`63d30e6a3a63dd2c3825afcef39d98263d586460fb89cae300a4dda37f597c6d`
`ydotoold.service`	`7a4a864f28dfd662aba44761a429322ebea31f1b4c8dfabca16ed9b8b46b64d5`
`godmez-autoclaw-regimen.path`	`6dfd48701e7d54da03c91b6e960327f3a8f8e1d4a4c1c5b7a50cab73ec667907`
`godmez-autoclaw-regimen.service`	`924830d23b1effb48183086cbb613be6799146c02bdc9c41313d16c036f68bc3`
`~/.local/share/dictado/dictado.py`	`8fc453ed38063526038b957ccf1f72aa88ac319047658345c3fe2612dfbe4d96`
`~/.local/share/dictado/silero_vad_v6.onnx`	`4cbf549b8326f60f80f2536d9eefeb450a9abe83365a098031c89719f1be17d2`
`~/.local/share/applications/net.local.dictado.desktop`	`6cba07886274a25e817d79dee32b8a885af8d9912c48611ec17d15588d19c1ca`
`~/.cache/dictado/modelos/large-v3-turbo.pt`	1,617,941,637 bytes; digest not computed
`~/claude-setup/godmez/autoclaw/godmez_autoclaw_apply.py`	`2cc73c0aece10803d9d6749dbc98897b1b2c1b2ed19b0ddf0f941d34e7c68a82`
`~/.openclaw-autoclaw/workspace/AGENTS.md` (changes when the interface appends blocks)	`5a5572bf72c8c38d2163fbebdb390d13ac7b0c57daee89e1bfbb5b3970263faf`
`~/.openclaw-autoclaw/workspace/skills/be-like-godmez/SKILL.md`	`34d45a6e507b9f63f6bb4947c5cc0204cb3e86ab2a57cff29d9fc325c792bcf0`
`~/.openclaw-autoclaw/agent-project-read-roots.json`	`e740472571274ba7aed89c3292733216a9777c0583f2d50e59505fe0626b06c0`
`~/Escritorio/AutoClaw/AutoClaw_Fedora_andamiaje_portable_2026-09-21.tar.gz`	`2a159add9ee86a32e125bcde7873f126dc3e18d89286b78fb0e49a0f3f903d74`
installer `autoclaw-1.17.8-setup.exe` (log, 2026-09-09)	`dd00046df93ff4a02ec540fc31073a1b9b3551ef9ba6d99fd9d2064fff445ffc`
installer `autoclaw-1.18.1-setup.exe` (compendium, 2026-09-10)	`f358ec32fb39c9e61bc9f61e70daea13cec3bc7b98fdd668f6dce93fc3d8360d`
intact `app.asar` of 1.17.8 (log, 2026-09-10)	`ea06809693f9c60367de577c3a66c9eb75924d101bb7e74eeba109b6e922b5da`
`~/claude-setup/autoclaw/diag/atribuye_escrituras_config.py` (compendium)	`2cc5491729d3bdffaa887a5496390c5e1eb08d813b3ae0a59210c9e851b420cc`
`~/claude-setup/autoclaw/diag/voz/calibra_porton_voz.py` (log)	`3c65fb3ef111d992ac93c9d1abe6872fbb85590b19a1c1ea965b67749aabcddc`

Note. Paths beginning with `bin/` and `patches/` hang from `~/.local/share/autoclaw-linux/`; the units, from `~/.config/systemd/user/`; and those beginning with `.../resources`, from `~/.wine/drive_c/Program Files/`

AutoClaw/. The own providers file carries no digest in this table because it contains the plan's key and its digest is of no use for verifying it on another machine.

14.4 What was applied on 2026-09-20 and the state it left

Four units, all with their oracles re-run at closing and none touching a running service. The first moved the two superseded documentary copies on the desktop to the wastebasket and turned off the keeping of dictation audio, deleting the three recordings of 2026-09-13 without opening them. The second corrected bin/autoclaw-set-provider-key.sh in both its defects and added the bench that tests it on copies. The third corrected in the operating README the two claims session 09 had left out of date. The fourth repaired parsePageRange across its three surfaces (chapter 7.5).

Table 13

State after the units of 2026-09-20

Check	Result
keep_recording	false in ~/.config/dictado/config.json, with backup .bak-2026-09-20; it takes effect from the service's next start, because the server reads its configuration only once at startup
Diagnostic recordings	none: ~/.cache/dictado/diagnostico/ was left empty
Copies withdrawn	REGISTRO_INSTALACION_2026-09-09.md and REGISTRO_REGIMEN_AUTOCLAW.md from the desktop, to the wastebasket, recoverable
Key script	accepts both shapes of the own configuration file, prints no fragment of the key and accepts a second argument for writing to a copy without synchronising
Key script bench	PASA: los cuatro casos y el control negativo
Page-limit patch	applied in all three files; autoclaw-page-range-patch.py --check with exit 0
Truncation notice patch	autoclaw-pdf-notice-patch.py --check with exit 0, after updating its reference to the current extractor
Page-limit bench	PASA: semantica, nota y aplicador, con sus tres controles
gateway-bundle.mjs	202139662eeebe46ca121ba24c0db2ea98d99769975c63dff65b9b58eba0a530
dist/openclaw-tools-Bm2s_FnJ.js	20998a21ea34391e5171fd86d2171af386855d24707d9961a835487b28bcd204
installed document-extractor.js	9fa4579704a23210fcd8d28ce98b0ced57cf60e6d354919f9a6393cc30efd6e6
Services	none was started, stopped or restarted; dictado.service keeps its start time of 09:01:40 and the gateway remains inactive with AutoClaw closed

Note. The gateway takes up the patch on opening AutoClaw. Dictation was not restarted because the bus flood of 2026-09-14 happened while dictado.service was being stopped with the model resident on the card (chapter 9.8), and forcing that restart to bring forward a configuration change does not compensate the risk. The services row describes what those four units left at 09:50; later that same day the owner opened AutoClaw and the gateway became active from 10:07:25, which is the state the following day's work built on.

14.5 The second day's work of 2026-09-20: page integration and audio output

Two checks that until that day could only be asserted in parts were measured, neither with a language model nor touching a service.

Table 14

Page-limit integration and audio output, measured on 2026-09-20

Check	Result
Test document	eighty pages, ~/.openclaw-autoclaw/workspace/prueba_80_paginas_s11.pdf, digest e9d278aff117e1f15e29b923991b8cc59d570fcc34ef114c16a769a149492ff1, with one mark per page taken from system entropy and verified with pdftotext

Check	Result
`61-80` over the installed package	twenty pages with marks 61 to 80
`1,70` over the installed package	both pages, marks 1 and 70
`55-60` over the installed package	six pages and **no** truncation notice
Sixty-one pages requested	`Page selection of 61 pages exceeds the per-call limit of 60`
No range, positive control of the notice	sixty pages and the notice stating how to read the following ones
Negative control over the pristine package	`61-80` comes back empty and with a single document the tool would answer that it appears to be scanned; `1,70` comes back as page one
Page-limit bench	repaired: its layer C started from the installed tree and therefore failed when re-run with the patch in place; it now starts from the pristine backups verified against the manifest, and gives `PASA` again
Voice: player's journal	a single playback with status zero and a new file in `media/outbound/`
Voice: monitor capture	peak −5.72 dBFS against the reference's −5.71; the control in digital silence, without a non-zero sample in twenty seconds
Voice: signal identity	envelope correlation 0.9983 against the null's 0.7527
Microphone	not opened in the final measurement; the first run did open it through a target PipeWire ignored, and its captures were deleted without being opened (chapter 12.8)
Services	none was started, stopped or restarted
Exit through the menu: leftover processes	none; the four processes of the 10:07 instance disappeared with the exit
Exit through the menu: automatic shutdown	it acted at 19:32:17 and finished at 19:32:21; the gateway received `SIGTERM` and closed cleanly in 253 ms
Exit through the menu: final state	the five named units, the voice player and the timer itself were left inactive

Note. The instruments are `diag/mide_integracion_paginas.mjs`, `diag/fabrica_pdf_marcas.py`, `diag/voz/mide_salida_audio.py` and `diag/mide_remanentes_salida.py`, and their outputs were left in `diag/INTEGRACION_PAGINAS_S11.json`, `diag/voz/SALIDA_AUDIO_S11.json` and `diag/REMANENTES_SALIDA_DESPUES.json`. The manual is assembled from its five parts with `~/.cache/manual_autoclaw/ensamblar.py`, which resolves the digests of the table of earlier documents and whose reconstruction of the published file is checked with `--check`.

The exit through the menu, and the verdict the instrument failed to give. The question open since 2026-09-14 was whether an ordinary exit leaves processes capable of neutralising the automatic shutdown, and the measured answer is that it does not. The owner took the prior snapshot at 19:29:29 with the four processes of the 10:07 instance alive, exited through the tray menu and ran the following step: no process survived, the timer fired at 19:32:17, the shutdown script announced in its journal that the window was closed, the gateway received `SIGTERM` at 19:32:17.527 and closed cleanly in 253 ms, and at 19:32:21 the service finished. Three minutes later, the five named units, the voice player and the timer itself were inactive.

The instrument, however, printed the opposite: "falla del apagado", with the five units seen as active. The cause is the oracle and not the system. The reader broke its wait as soon as the timer's `LastTriggerUSec` changed, which marks the instant at which the timer **activates** the service, and photographed immediately afterwards; since stopping five units took four seconds, the snapshot fell inside the shutdown and read as a leftover what was a shutdown in progress. The repair replaced the instant with the effect: the shutdown script stops the timer as its last step, so the timer going inactive cannot be observed before the units are already stopped, and if instead two full cycles pass with the timer still active, the script ran and chose not to shut down, which is the leftover case. The bench `diag/test_mide_remanentes_salida.py` fixes the repair with thirteen checks and replays the measured race; with the old oracle reinjected, four of them fail, among them the one that again sees the five units active.

The interactive turn, which closes the whole chain. On 2026-09-21 the owner opened AutoClaw, attached the eighty-page document and asked for the mark on page eighty. The agent answered `MARCA 080: 835dba07`, which is exactly the manifest's; that mark is unique among the eighty, all distinct from one another, and `pdftotext` finds it on page eighty and on no other, so getting it right without having read it would require guessing eight hexadecimal digits. The gateway's trace shows the calls against `token-plan-sgp.xiaomimimo.com` with status 200 and the models `mimo-v2.5-pro` and `mimo-v2.5`, that is, MiMo and not Z.ai, exactly as the criterion required. The reply, moreover, sounded through the speaker without anyone arranging it, because the player comes back with the gateway.

What this turn establishes and what it does not. It establishes that page eighty reached the model, which is precisely what the limit of sixty prevented before `parsePageRange` was repaired: the negative control over the pristine package, measured the day before, returns empty for the range `61-80`. It does not establish by what path the agent asked for it, because the

document extractor leaves no line of its own in the gateway's journal and the trace shows only the calls to the model; that is a limit of the instrument and not a result.

The confirmation by ear, which closes the voice chain. On 2026-09-21 the acoustic test was repeated with another phrase, "Prueba de sonido del veintiuno de septiembre. Esta frase se oye una sola vez y completa", of 6.888 seconds, and with the player started only for the measurement and stopped afterwards, because it had been left inactive along with the gateway. The automatic verdict again gave `PASA`: a single `played` line in the journal and no error line, a new file in `media/outbound`, envelope correlation of 0.9868 against 0.5909 for the null with the reversed reference, and a prior control in absolute digital silence, without a single non-zero sample in twenty seconds. The owner confirmed by ear that he heard it, complete down to the last word and only once, with which the chain stands verified from the synthesiser to the ear and not merely as far as the device. The report is in `diag/voz/SALIDA_AUDIO_20260921.json`, which does not overwrite the previous day's.

The shutdown predicate, which counts command lines and not processes. The installed script decided with `pgrep -f 'AutoClaw\.exe'`, which matches the entire command line of every process on the system. Measured on 2026-09-20 with a negative control, a live decoy and a reversion control: a `/bin/sh` loop carrying only that path in its line is matched, so a single terminal mentioning the executable is enough for the shutdown never to happen while it lasts. The proposed successor, in `propuestas/autoclaw-idle-stop.sh`, compares `argv[0]` by equality against the installed path, which is what the four real processes under Wine carry, and additionally inverts the failure mode with a guard: `pgrep` failed towards never shutting down if the executable changed name, whereas a comparison against a fixed path would fail towards shutting down with the window open, so the successor first verifies that the executable is on disk and refuses to act if it is not. Its bench, `propuestas/test_autoclaw_idle_stop.py`, runs both predicates against the same live decoy and touches no real unit, because it substitutes `systemctl` with a double that merely records what it was asked to do; it accepts as an argument the path of the script to test, so it can be run against the installed one and not only against the copy.

It was installed on 2026-09-21, with a backup of the previous one in `bin/autoclaw-idle-stop.sh.bak-2026-09-21` (digest `3698db8495f0c3744aedb65d4ec5bcb690dbdba7fd076f7a485e5cfd2b70fd04`) and the bench re-run against the installed path, which gave zero failures. The smoke test, this time against real systemd and not against a double, measured the three cases that matter: a foreign process merely mentioning the path no longer blocks the shutdown; a process whose `argv[0]` is exactly the path stops it, which is the open-window case; and with neither of the two the script recognises the closed window and acts. No unit changed state during the test, because they were all already inactive after the previous day's exit. On the decoy for the open window a note is in order: a process can only claim an arbitrary `argv[0]` with `exec -a`, and without that the shell keeps its own, so the decoy tests the wrong case.

One declared limit remained, which the interactive turn lifted without setting out to: the test had been done with decoys and not with the real application, because AutoClaw was closed. With the window open, the same equality criterion on `argv[0]` identified the four real processes, so the new predicate recognises AutoClaw and not merely an imitation. The bench additionally incorporated that condition as an explicit precondition: with the window open almost all of its cases are unmeasurable, because they check that the script shuts down, so it now detects this, says so and exits with 2, meaning "nothing was measured", instead of exiting with 1 and passing off as a defect of the code what is nothing more than somebody working.

15. Glossary

AutoClaw Z.ai's desktop client for Windows and macOS, built with Electron on top of a fork of OpenClaw. In this manual, "the interface" or "the window" is its graphical part, which runs under Wine.

OpenClaw Artificial intelligence agent whose core is a gateway that maintains sessions, calls models and runs tools. It has an official Linux distribution.

Gateway OpenClaw's server, written for Node.js, to which the interface connects. Here it runs natively on Linux on port 18789.

External gateway The mode in which AutoClaw's interface connects to a gateway it did not launch.

Wine Compatibility layer that runs Windows programs on Linux by translating their system calls.

Wine prefix Directory simulating a Windows installation, here `~/.wine`, with its `C:` drive in `drive_c`.

WoW64 Wine's mode for running 32-bit programs inside a 64-bit installation.

Electron Framework that packages Chromium and Node.js to build desktop applications.

`app.asar` File containing the code of an Electron application; here it measures 309 MB.

Electron fuses Switches burned into the executable that enable or disable capabilities, such as running as a Node interpreter or validating `app.asar` integrity.

Chromium The free browser on which Electron and Chrome are built; here it is also the agent's native browser.

ANGLE Chromium's layer that translates OpenGL graphics calls into Direct3D.

Direct3D Windows's graphics interface. Wine ships its own Direct3D translator; DXVK is another translator, which this prefix does not use.

Mesa Free implementation of OpenGL and Vulkan serving as a user-space graphics driver.

Wayland, X11 and XWayland Wayland is the graphics protocol of the KDE Plasma session; X11 is the earlier protocol, used by Wine's windows; XWayland is the server that serves X11 windows inside a Wayland session.

KWin KDE Plasma's compositor: the program that draws windows on screen and serves global shortcuts. If it dies, the applications depending on its connection die with it.

Session bus and `gdbus` The session bus (D-Bus) is the channel through which the programs of a desktop session communicate; `gdbus` is a console tool for calling its methods.

Node.js and V8 Node.js is the JavaScript runtime outside the browser; V8 is its engine.

`Intl.Segmenter` Standard JavaScript function that splits text into visible characters, words or sentences, and which depends on the ICU internationalisation library.

Native module Library compiled for a specific operating system which a Node program loads; Windows ones do not work on Linux.

Authenticode signature and extended validation Authenticode is the digital signature of Windows executables. An extended validation certificate is one whose issuer verified the signer's legal identity through a reinforced procedure.

Timestamp Third-party signed evidence that a signature existed on a given date.

SHA-256 digest Cryptographic summary of 64 hexadecimal digits which changes if a single byte of the file changes. MD5 is an older and shorter summary, compared here only against the ETag, the version identifier a web server assigns to a file.

SQLite Database contained in a single file; OpenClaw stores tasks, pairing and memory there.

WebSocket and code 1012 A WebSocket is a persistent bidirectional channel over HTTP. Close code 1012 means "service restart", and the interface reads it as an expected close.

Local loopback The machine's internal network interface, 127.0.0.1, unreachable from other machines.

Port fence A service of our own that occupies ports 18790 to 18799 to force the interface to use the external gateway.

systemd, user unit, `.path` and `.timer` systemd is the service manager; a user unit is a service of the user rather than of the system; a `.path` unit triggers a service when a file changes, and a `.timer` one, at intervals.

Control group Grouping of processes whose memory and processor use the kernel measures and limits.

`ExecStartPre` A command systemd runs before starting the main service.

SIGKILL, SIGUSR1 and SIGABRT System signals: SIGKILL terminates a process with no say in the matter; SIGUSR1 is a free-use signal which OpenClaw uses to restart itself; SIGABRT is the one a program sends itself when aborting.

Hot reload Application of a configuration change without restarting the process.

Token In "token plan", the unit in which a model's use is counted; in "random token", a single-use or single-startup credential.

Intermediary Server standing between a client and a service; here, Z.ai's, which demands the interface's session.

Credits Balance of model usage from Z.ai inside AutoClaw.

Custom model A model the interface allows to be registered by hand with its provider, address and key.

Context window and maximum output The maximum number of tokens a model accepts as input and can produce as a reply.

Image model and document model Models the gateway falls back on when the main model does not accept images or documents.

Skill and tool A tool is an action the agent can execute, such as reading a file; a skill is a package of instructions and scripts the agent loads for a task.

Workspace, startup file and subagent The workspace is the agent's directory; the startup files, such as `AGENTS.md`, are handed to it at the beginning of each session; a subagent is a secondary agent the main one launches.

Session transcript A record, in line-delimited JSON, of every message and tool call of an agent session.

UNO LibreOffice's component programming interface, through which an external program reads and edits open documents.

Extractor and `clawpdf` `clawpdf` is the engine that reads PDF documents; the extractor is the gateway's module that calls it and assembles what travels to the model.

Rasterisation and megapixel To rasterise is to turn a page into an image; a megapixel is a million pixels.

Page range A selection of pages requested from the tool with the `pages` argument, such as `21-40`.

Scan guard The tool's rule declaring scanned any document whose three-page probe brings fewer than two hundred characters.

Patch and applier A patch is a local modification of a foreign file; the applier is the script that installs it, checks it and removes it, and refuses before a file other than the expected one.

Oracle A check whose result decides, without interpretation, whether something came out right.

Positive and negative control A positive control demonstrates that the check detects what it looks for when present; a negative one, that it does not detect it when absent.

Mutant A deliberately broken version of a program or its data, used to verify that a test fails when it should.

Pre-registration A document fixing, before measuring, the hypothesis, the sample size, the controls and what each result will mean.

Blind adjudication, hourly grid and classes R and F To adjudicate blind is to apply predictions written before seeing the data. The hourly grid is the series of instants, every 3,600 seconds, at which renewals are predicted. Class R is the write following a credential renewal, and F, the one following the first subsequent remote query.

Whisper and `large-v3-turbo` Whisper is a family of speech recognition models; `large-v3-turbo`, whose short name is `turbo`, is a version of the large model reduced by distillation.

No-speech probability A figure Whisper publishes per segment to indicate that it heard no speech; in `large-v3-turbo` it is degenerate.

Voice activity detector, Silero and gate A voice activity detector decides which stretches of audio contain speech; Silero is the one used here; the gate is dictation's rule not to transcribe if the detector found insufficient speech.

Root-mean-square level and dBFS The root-mean-square level is the square root of the mean of the squares of the signal; dBFS are decibels relative to full scale, the maximum representable value, so that 0 dBFS is the maximum and −240 dBFS amounts to no samples at all.

Word error rate The proportion of words that have to be substituted, deleted or inserted to reach the correct text.

ROCm and PyTorch ROCm is AMD's compute platform for its graphics cards; PyTorch is the machine learning library that uses it.

Graphics memory The graphics card's own memory; the RX 6800M has 12 GB.

Junction and edge temperature The junction temperature is that of the chip's hottest point; the edge one, that of a sensor on its periphery.

Uncorrectable error and bus flood An uncorrectable error is a hardware failure the system cannot repair on the fly; the data fabric sync flood is the way AMD's platform halts everything in the face of such an error, leaving evidence in a register the kernel reads at boot.

PCI Express and `amdgpu.runpm` PCI Express is the bus connecting the dedicated card; `amdgpu.runpm` is the driver parameter governing runtime power saving, disabled here with 0.

Persistent kernel message storage A mechanism (pstore) that saves the kernel's last messages in memory surviving a reboot; with no destination configured, it saves nothing.

ntfs3, dirty volume and `ntfsfix` ntfs3 is the kernel driver for the NTFS format; a dirty volume is one marked as not cleanly closed; `ntfsfix` is the tool that clears that mark.

PipeWire, `pw-record`, `pw-play` and null sink PipeWire is the audio server; `pw-record` and `pw-play` record and play through it; a null sink is a virtual output device that makes no sound.

`wl-copy`, `ydotool` and `/dev/uinput` `wl-copy` copies to Wayland's clipboard; `ydotool` simulates a keyboard through `/dev/uinput`, the kernel device for virtual input, with its `ydotoold` service.

`uaccess` access control list A permission the system automatically grants over certain devices to the user sitting at the machine.

XTEST and `xdotool` XTEST is X11's extension for synthetic keyboard and mouse events; `xdotool` uses it, and under Wayland those events do not reach windows.

inotify Kernel mechanism that tells a program when a file or a directory changes.

Be Like GODmez The machine owner's working regime, with rules of rigour, voice and conventions, loaded inside the agent as described in chapter 10.

Units of measurement MiB and GiB are powers of 1,024 (1 MiB is 1,048,576 bytes); MB and GB, of 1,000; W are watts; °C, degrees Celsius.

16. Sources and earlier documents

16.1 The five files of the desktop folder

This manual makes it unnecessary to read the five earlier files of `~/Escritorio/AutoClaw/` in order to understand the installation's state and history, but it does not render them all useless.

Table 12

Earlier files of the folder, with their state relative to this manual

File	Size, date and SHA-256 digest	State relative to the manual	Proposed destination
`AutoClaw_Fedora_andamiaje_portable_2026-09-21.tar.gz`	222,607 bytes; 2026-09-21; `2a159add9ee86a32e125bcde7873f126dc3e18d89286b78fb0e49a0f3f903d74`	current: it replaces the portable bundle of 2026-09-04, which reproduced an installation predating the patches, the voice player and the shutdown fix, and which additionally included code from AutoClaw's package	keep
`COMPENDIO_AUTOCLAW_ROG_STRIX_PARA_VIVOBOOK_2026-09-10.txt`	154,604 bytes; 2026-09-10 19:50; `756dfd73ad631efcb0b7c5185c442400fbfec139ba8cd069f6696670b938f2ea`	superseded as a description of the ROG Strix; its appendices transcribe scripts and diffs which the manual cites rather than copying, among them the corrected block of the key script	withdrawn from the desktop on 2026-09-21 and kept in `legacy-escritorio-2026-09-21/`
`Informe_Replicacion_AutoClaw_OpenClaw_2026-09-04.md`	64,396 bytes; 2026-09-04 18:47; `ae4c541d503db215f99a76ef6ef78abb76fd1d82b255072d6a6ba1ef4e08e35a`	superseded for the ROG Strix; it remains the only transcription of the original units and of the VivoBook's memory changes	withdrawn from the desktop on 2026-09-21 and kept in `legacy-escritorio-2026-09-21/`
`REGISTRO_INSTALACION_2026-09-09.md`	25,107 bytes; 2026-09-10 10:46; `90e1737ca7ed5c78e5255085f306986d20ec5c8e485f67298c02196c0094b917`	**superseded** and **moved to the wastebasket on 2026-09-20**: it was an old copy of the live log in `~/claude-setup/autoclaw/`	done; restorable from the wastebasket if needed
`REGISTRO_REGIMEN_AUTOCLAW.md`	223,151 bytes; 2026-09-13 20:08; `877de9b9b54cd84f149e320872abaee1d938292eafc86ad975b49932e88d4ae5`	byte-identical to the live log, and **moved to the wastebasket on 2026-09-20**	done; restorable from the wastebasket if needed

Note. The digests were computed on 2026-09-14 over the files in `~/Escritorio/AutoClaw/`.

16.2 Other sources consumed

Besides those of table 1, the machine's trap memories were read concerning the compositor crash, the nested compositor, `xdotool`, `ELECTRON_RUN_AS_NODE`, Fedora's Node, Whisper and the card crash of 2026-06-29; the memory of the regime inside the agent; the memory of the thermal test; and the system journal, the kernel journal and the dictation journal, together with the installed files, on 2026-09-14.

16.3 Discrepancies between sources

Where two sources disagree the later one governs, and the measurement of 2026-09-14 governs over them all; no figure was averaged.

- **The audio output.** The installation log and the list of open items give it as muted; on 2026-09-14 it is not.

- **`en.sh`.** The compendium and the list of open items give its `--` as pending a decision; the file has been repaired since 2026-09-13.
- **The number of Fedora-specific defects.** The headings of the log and of the compendium say “three defects”; the compendium itself enumerates four under that heading, because the missing `XAUTHORITY` was found on 2026-09-10, and the manual counts four.
- **Z.ai’s models.** The operating README and the compendium say seven; the session 09 log establishes that since 03:54 on 2026-09-11 there are six.
- **The series adjudication.** The compendium gives it as pending until after 16:16 on 2026-09-11; the log adjudicates it in session 09, with the cut-off at 08:44:31.
- **The regime’s chain.** The compendium gives a session 09 prompt of the regime as live; the chain was closed on 2026-09-13 with no live prompt.
- **The copies on the study disk.** The compendium places the report and the portable bundle in the Fedora settings folder of the study disk; on 2026-09-14 they are not at that path, and they are in `~/Escritorio/AutoClaw/`. The README and the log, besides, name desktop paths predating the `AutoClaw/` folder.
- **The desktop shortcut.** The report and the compendium describe it as pointing at the launcher; on 2026-09-14 it does not exist, and the menu one does point at the launcher.
- **Dictation’s graphics memory.** The 4.9 GB of the 2026-09-11 bench and the 4.27 GB of the whole card on 2026-09-14 measure different things, and both are reported.
- **The card’s repair.** The memory of the card crash takes the repair with `amdgpu.runpm=0` as confirmed; what is confirmed is that the `device lost from bus` message disappeared, while the bus floods of 2026-07-21 and 2026-09-13 occurred with that parameter active, at least in the second case.
- **The diagnostic recordings.** The dictation journal records five recordings kept; on 2026-09-14 three remained, and the two of the 2026-09-11 test cycle were gone. The three remaining ones were deleted on 2026-09-20 without being opened.

16.4 In plain terms

AutoClaw works on Fedora, and works well: it converses, reads images and documents, browses, edits LibreOffice documents, answers aloud and can be dictated to, without spending Z.ai credits and with an agent that says so when it has not read a whole document. The price is maintenance that does not exist on Windows, because every update forces restoring three pieces and revalidating; a window that draws without acceleration, with no visible consequence in use; some Windows capabilities that are lost; and accepting that what the agent reads and what it answers aloud leaves the machine. Dictation is a separate piece, useful and private as far as the voice goes, which loads the graphics card, and this machine’s abrupt shutdowns are a problem predating all of this whose cause is still not established.