

The Legacy Patch Suite: A Natural History

Every adaptation the repackaging applies on `main`, from acquiring the upstream bytes to patching the bundle to shipping the package: why each exists, how it survived eighteen months of upstream re-minification, and what the official Linux build and the `v3.0.0` rebase do to it.

DOCUMENT	CDL-ANT-0009 · Rev B · RELEASED
SUBJECT	Dossier of the legacy Linux adaptation layer on <code>main</code> , unit by unit
FOCUS	Mechanism, origin, revision history, and fate under the official- <code>.deb</code> rebase for every unit
SCOPE	<code>main @ 0c4e73f</code> (upstream 1.17377.1) vs the official <code>claude-desktop_1.17377.2_amd64.deb</code> ; rebase branch <code>rebase/official-deb @ cafb4cc</code>
TOOLS	<code>git pickaxe</code> + <code>gh issue/PR</code> archaeology, CDL-ANT-0008 byte checks, a repo-wide environment catalog, and an 18-unit contrarian-gated research workflow
HEADLINE	Ten of eighteen units die in the <code>v3.0.0</code> rebase, redundant against the official Linux build; three survive, one is parked, one awaits a reproduction, and three are reworked rather than removed.
AUTHOR	Claude Fable 5 · July 3, 2026
REVIEWED BY	Aaddrick Williams
REV B	Live-hardware settlement of the addendum's open checks, July 3–4, 2026 (\$21)

Eighteen Months of Patches, Read Back in Full

Since its first commit on December 26, 2024, `claude-desktop-debian` has done one thing at its core: take the `app.asar` out of Anthropic's Windows installer and make it behave like a native Linux application. The instrument for that is the **patch suite**: nine shell modules under `scripts/patches/` totaling 2,167 lines, plus 3,870 lines of injected JavaScript (`frame-fix-wrapper.js` at 997 lines, `cowork-vm-service.js` at 2,766, `claude-native-stub.js` at 107). Together they form thirteen patch units that rewrite minified upstream code by regex, stand in for Windows-only native binaries, and impersonate services the Windows app expects to find. The suite has been touched by **272 commits**, tracked **79 upstream version bumps**, and shipped in 147 tagged releases. This report widens the frame past the patch suite proper to the whole adaptation layer that surrounds it: the acquisition pipeline that fetches the upstream bytes, the launcher and doctor that decide runtime policy, and the packaging-time staging that shapes the shipped artifact, **eighteen units in all**.

On June 30, 2026, Anthropic shipped the first official Claude Desktop for Linux `.deb`, torn down in report **CDL-ANT-0008**. Two days later the project locked the decision to rebase v3.0.0 onto that official package, and the patch-necessity audit returned its verdict on the `app.asar` suite: **11 delete, 2 survivor candidates, 1 behavioral check, 1 parked** across the matrix's fifteen rows. A follow-on accessibility reassessment, re-verified against pristine official 1.18286.0, then annotated three of those calls with residual runtime risk the `.deb` bytes cannot settle, moving `frame-fix`, `wco-shim`, and `quick-window` to a *verify* disposition without reversing any shipped code (section 21). Read under that reassessment and extended to the full adaptation layer this report covers, the settled disposition lands at the unit level as **eight units deleted outright, two survivors, four behavioral holds, one parked subsystem, and three reworked** rather than removed (the chassis, the launcher and doctor policy, and the acquisition pipeline). The deletions are not an admission of wasted work; in most cases the official build independently converged on the same fix the community had already derived, which makes the suite's history a record of correct diagnoses.

This report is the suite's dossier, written at the moment of its retirement. For every patch unit it answers four questions: what the patch does mechanically, why it was created (with the issues and pull requests that motivated it), how it was revised across eighteen months of upstream re-minification, and what the official build and the new `rebase/official-deb` build script do about it. Section 03 covers the shared chassis every patch runs on; sections 04–13 walk the ten `app.asar` units; sections 14–18 cover the surrounding layer this revision brings fully into scope, the acquisition pipeline, the launcher and doctor, and three packaging-time pieces; section 19 compresses everything into a fate matrix; section 20 closes with what the history means.

Eighteen Dossiers, Each Through a Contrarian Gate

The findings here were produced by a multi-agent research workflow with adversarial verification at every step. One historian agent per patch unit reconstructed mechanism, origin, and revision history from the `main` branch: code was read via `git show main:...`, origins were traced with pickaxe searches (`git log -S`) so that logic which migrated across the `build.sh` refactors is attributed to its true introduction rather than a file move, and every issue and pull request reference was resolved live against GitHub with `gh`. Each dossier then had to pass a **con-**

trarian gate: an adversarial reviewer that re-verified every cited issue number and title against reality, every verdict against the patch-necessity matrix in docs/learnings/official-deb-rebase-verification.md, every mechanism claim against the main-branch bytes, and every causal story against the commit or issue that states the cause. Dossiers that failed were revised and re-gated; all eighteen passed, the stubbornest after two rounds. A completeness critic swept main for machinery no dossier covered; its findings, three packaging-time pieces, together with two subsystems an earlier pass had scoped out (the acquisition pipeline, and the launcher and doctor policy), are the five units in sections 14–18. Each unit’s **Affects** line reverse-looks-up its own issues and pull requests against a repo-wide environment catalog compiled from every issue and PR body, so distro, desktop, and compositor sensitivity is catalog-backed rather than asserted.

Two facts were additionally byte-verified for this report against the pinned official package (claude-desktop_1.17377.2_amd64.deb, SHA-256 ec08d41...212ab, re-downloaded and hash-checked on July 3): the official tree ships its locale JSONs loose at resources/*.json, settling the fate of the i18n copy machinery, and it contains **no claude-ssh artifact anywhere**, which bears on the staged SSH helpers in section 14. Claims about the rebase side rest on reads of the checked-out rebase/official-deb working tree at cafb4cc.

Scope. This report covers the whole Linux adaptation layer on main: the acquisition pipeline that fetches the upstream bytes (section 14), the app.asar patch suite and its injected files that mutate or stand in for application code (sections 03–13), the launcher and doctor that set runtime policy (section 15), and the packaging-time staging that shapes the shipped artifact (sections 16–18). Where a verdict rests on evidence this project has not yet reproduced at runtime (the Quick Entry focus bug, the mcpServers overwrite, the in-app topbar question, the claude-ssh capability), the text says so plainly.

03 • CHASSIS

The Chassis: Extract, Patch by Regex, Repack

MECHANISM

Before any behavior patch runs, the chassis unpacks the upstream archive: `asar extract app.asar app.asar.contents`, per-feature patches, then `asar pack ... --unpack '**/*.node'` in `finalize_app_asar` (scripts/staging/electron.sh). The `--unpack` flag is load-bearing: Electron’s `asar-to-.unpacked` redirect fires only when the manifest entry is annotated as unpacked; without it, `require()` of native helpers returns `MODULE_NOT_FOUND`. Around the extract/repack cycle, `patch_app_asar` (scripts/patches/app-asar.sh) performs the **layout normalization** the Windows-extracted tree demanded: it rewrites `package.json` so `main` points at `frame-fix-entry.js` (a two-line shim that loads `frame-fix-wrapper.js` before the original entry point), sets `desktopName` to `claude-desktop.desktop` (or the `io.github.aaddrick` ID for `Applmage`, since Electron derives the `Wayland app_id` from this field, #561, #562), copies the locale JSONs from beside the `asar` into `resources/i18n/` where the app actually resolves them (#23, `en-US.json` not found), and copies `Tray*` icons into the `asar` so both packaged and unpackaged (Nix, `isPackaged=false`) lookups succeed. It also fails fast if upstream ever renames `productName`: Electron ignores `--class=` and derives `X11 WM_CLASS` from `productName`, so `readonly WM_CLASS='Claude'` in `build.sh` is the single source of truth and a mismatch aborts the build.

Because upstream re-minifies every release, the chassis also carries `extract_electron_variable` in `scripts/patches/_common.sh`, which recovers the electron module variable at build time:

```
electron_var=$(grep -oP '[$\w]+(?:\s*=\s*require\("electron"\))' \
"$index_js" | head -1)
```

with a fallback anchor on new X.Tray and a hard exit if both fail. Its sibling `fix_native_theme_references` auto-repairs a recurring upstream minifier bug where the bundle references `nativeTheme` through the wrong alias.

ORIGIN

The chassis predates every feature patch: `extract`, `tray-icon copy`, and `repack` are already present in `d8f4bbe`, the repository's December 2024 initial commit (`build-deb.sh`). Everything layered on since is a normalization or repair forced by the Windows-extracted tree or by upstream re-minification: the `i18n` copy from Coldsewoo's #23 report, the `main` rewrite from speleoalex's #127 (windows appeared undecorated on Linux window managers), dynamic identifier extraction from #220 (fixing #218/#219), and the `productName`/`WM_CLASS` fail-fast from #655.

REVISION HISTORY

WHEN	REFS	EVENT
2024-12	d8f4bbe	Origin: initial <code>build-deb.sh</code> commit already contains the <code>asar extract/repack</code> scaffolding and an in- <code>asar</code> <code>tray-icon copy</code> .
2025-03	#23, #26 466705d; #25 ee7e4bc	<code>i18n</code> locale JSON copy into <code>resources/i18n/</code> added, motivated by Coldsewoo's #23 (<code>en-US.json</code> not found inside the <code>asar</code>); #26 landed the surviving <code>*-*.json</code> glob first, #25 merged later as the documented fix (the #23 link for #26 is inferred from identical code, not recorded).
2025-11	#127, 7882635	<code>package.json</code> <code>main</code> redirected through <code>frame-fix-entry.js</code> (speleoalex): Claude Desktop windows appeared without borders or decorations on Linux window managers.
2025-11	#122, 47c1889	Tray icons pulled back out of the <code>asar</code> for a filesystem-only copy: "Tray icons must be in filesystem (not inside <code>asar</code>) for Electron Tray API to access them."
2026-02	#218/#219, 7bd2f38 (PR #220)	Anchor rot 1: upstream v1.1.2321 re-minified the <code>electron</code> variable <code>oe</code> to <code>Ae</code> , breaking every hardcoded <code>tray sed</code> ; hardcoded lookups replaced with <code>extract_electron_variable</code> plus <code>fix_native_theme_references</code> .
2026-02	#252, 546f845 (PR #253)	Anchor rot 2: v1.1.4088 minified the <code>electron</code> variable to <code>\$e</code> ; <code>\w</code> does not match <code>\$</code> , so extraction captured <code>e</code> and <code>seds</code> spliced code mid-name, a launch-time <code>SyntaxError</code> ; <code>electron_var_re</code> introduced for regex-escaped <code>sed</code> use.
2026-02	573f052 (PR #266)	Tray icons copied back into the <code>asar</code> so unpackaged builds (Nix, <code>isPackaged=false</code>) can find them, reversing #122's filesystem-only move.
2026-04	#421, 3150477	<code>--unpack '**/*.node'</code> added to the <code>repack</code> after <code>node-pty require()</code> returned <code>MODULE_NOT_FOUND</code> : the <code>asar-to-.unpacked</code> redirect needs the manifest entry annotated as unpacked.
2026-04	ff4821e	The 2124-line <code>build.sh</code> split into modules, creating <code>scripts/patches/_common.sh</code> , <code>scripts/patches/app-asar.sh</code> , and <code>scripts/staging/{electron,locales}.sh</code> (pure move, function bodies verbatim).
2026-05	#561/#562, 5a98854	<code>desktopName</code> edit added (ApplImage-specific ID): Electron derives the Wayland <code>app_id</code> from this field, and KDE Wayland grouped the pinned launcher separately from the running window.
2026-05	#644, b40441c	Anchor rot 3: <code>\w</code> silently truncated mid- <code>\$</code> names like <code>i\$A</code> ; <code>[\$\w]+</code> hardened repo-wide (3 sites in <code>_common.sh</code>), the same <code>\$</code> -capture trap having recurred in #421 and #555 before this convention stuck.
2026-05	#648, 76a5a21	<code>WM_CLASS</code> and <code>StartupWMClass</code> aligned to <code>claude-desktop</code> across all formats: the wrong direction, superseded two days later.
2026-05	#652/#655, e7e6475/73c9b8f	Reversal and centralization: Electron ignores <code>--class=</code> and derives <code>WM_CLASS</code> from <code>productName</code> (Claude); users confirmed via <code>/proc cmdline</code> plus <code>xprop</code> that the flag was silently ignored. Added <code>readonly WM_CLASS='Claude'</code> and the <code>productName</code> fail-fast.

Fate under the rebase. **REWORKED** The chassis is reworked, not deleted. With the official .deb a conventional electron-forge Linux tree, most normalization has nothing left to fix: `_common.sh` is gone (`grep -rn electron_var scripts/` returns nothing; both survivors self-anchor on literals), the `frame-fix` main edit is deleted (“the only `frame:!1` sites are the Quick Entry popup and two transparent overlay windows”), the `desktopName` edit and the `tray-icon` copy are deleted (upstream ships `TrayIconLinux(-Dark).png` natively), and the `i18n` copy is safely dead: byte-checked against the pinned official 1.17377.2 deb, locale JSONs ship loose at `usr/lib/claude-desktop/resources/*.json` (plus `resources/ion-dist/i18n/`). What remains on `rebase/official-deb` is a thin orchestrator with `active_patches=(patch_quick_window patch_org_plugins_path)`; an empty array means the official `app.asar` ships **byte-identical**, no extract, no repack. When patches are active, the `--unpack` expression is derived from the shipped `app.asar.unpacked` tree, folded into a single brace glob, and guarded by a set-equality check that hard-fails if the repacked unpacked set diverges. The one piece that survives upgraded is the `productName/WM_CLASS` fail-fast, now run unconditionally via `asar extract-file` so the assertion holds even on a patch-zero build that never unpacks the `asar`.

Affects. All Linux; packaging-layer normalization, format-general (deb/Applmage/Nix). The Wayland `app_id` derivation (#561/#562) is the one environment-specific edge.

frame-fix-wrapper.js: The Patch That Became a Platform

MECHANISM

Alone in the legacy suite, this unit performs **no sed edits on the minified bundle at all**. It is pure runtime interception, injected at build time by `scripts/patches/app-asar.sh`: the 997-line `scripts/frame-fix-wrapper.js` is copied into the `asar` contents, a two-line `frame-fix-entry.js` is generated by heredoc (it exists nowhere in the repo as a file), and the package’s `main` field is swapped to the entry stub, which loads the wrapper before the original `main`. At runtime the wrapper monkey-patches `Module.prototype.require` and, when the `id` is `electron`, returns the module wrapped in a `Proxy`, required because `electron`’s exports are non-configurable getters that silently swallow direct reassignment. The `Proxy`’s `get` traps substitute a `PatchedBrowserWindow` subclass, a `Menu` interceptor that re-hides the menu bar and smuggles in a hidden F11 fullscreen accelerator, a logging shim over `powerSaveBlocker`, and, on Linux, an `autoUpdater` no-op: a chainable `Proxy` where every property access returns a function returning the `Proxy` itself, with `then/catch/finally` and the coercion symbols masked so V8’s `thenable` check cannot hang an `await` on it. The constructor decides frames via `isPopupWindow()`: explicit `frame:false` means popup, a `parent` option means child modal, and `titleBarStyle` of `empty` or `hiddenInset` without a `minWidth` catches the About dialog; main windows get `frame:true` under the `CLAUDE_TITLEBAR_STYLE` modes (hybrid default, native, and a hidden mode documented in-file as broken on X11). The `Proxy`’s closure even bit the project’s own tooling: `docs/learnings/test-harness-electron-hooks.md` records that constructor-level `BrowserWindow` test hooks are silently bypassed by the `get` trap, and only `prototype-method` hooks survive.

ORIGIN

The wrapper began far smaller. In November 2025, external contributor speleoalex landed PR #127 (native window decorations) against the then-current Windows-installer `asar`, whose main window shipped `frame:false` with a custom titlebar; the fix was a wrapper heredoc embedded in `build.sh` plus blanket `frame:!1-to-frame:true` seds. No pre-existing issue motivated it: the PR body is the problem statement. The heredoc became a real file

during the #179 build-script refactor in January 2026, and in February 2026 vboi's PR #228 rework added popup detection (the blanket `frame:true` had given Quick Entry an unwanted frame, #223), scrollbar CSS, and the #149 attention-flash fix. The review series that followed (#231, landed as PR #232) culminated in the architectural pivot of commit `0f776a1`: the module-level Proxy replaced direct assignment, and the sed's were deleted from `build.sh` entirely.

REVISION HISTORY

WHEN	REFS	EVENT
2025-11	#127, 7882635/281847b	Origin: speleoalex's PR landed the wrapper as a <code>build.sh</code> heredoc plus blanket <code>frame:!1-to-frame:true</code> sed's against the Windows-installer asar's frameless main window; no pre-existing issue, the PR body is the problem statement.
2026-01	#179, 4bf5986	Heredoc extracted to <code>scripts/frame-fix-wrapper.js</code> as part of the build-script refactor.
2026-01	#155, 4174c20 (PR #169)	Menu-interception layer added: the app re-shows the hidden menu bar by calling <code>setApplicationMenu</code> asynchronously after window creation, so the wrapper intercepts it and re-hides.
2026-02	#223/#149, 83cbb9a (PR #228)	vboi's rework: popup and Quick Entry detection after the blanket <code>frame:true</code> had given Quick Entry an unwanted frame (#223), scrollbar CSS injection, and the KDE attention-flash fix (#149).
2026-02	#231, 0f776a1 (PR #232)	Architectural pivot: the review series against PR #228 culminated in a module-level Proxy replacing direct <code>module.BrowserWindow</code> assignment, silently swallowed by electron's non-configurable getters; the sed's were deleted from <code>build.sh</code> entirely.
2026-04	#321	Global Ctrl+Q shortcut added to give GNOME users a quit path without a tray icon.
2026-04	#399/#474, PR #484	The global Ctrl+Q grab stole Ctrl+Q from every application on the system (#399) and swallowed Ctrl+A on AZERTY layouts (#474); PR #484 rescope'd it to the focused window via <code>before-input-event</code> .
2026-04	#128, PR #450	XDG Autostart shim: <code>setLoginItemSettings</code> rerouted since Electron's <code>openAtLogin</code> is a Linux no-op.
2026-04	#448, PR #451	Close-to-tray: a <code>CLAUDE_QUIT_ON_CLOSE=1</code> opt-out added for in-app schedulers broken by quit-on-close; the opt-out did nothing, because the bundled close handler hardcodes hide-on-close on non-Windows.
2026-05	#623, PR #624	phelps-matthew's fix: an active <code>app.quit()</code> branch made the <code>CLAUDE_QUIT_ON_CLOSE</code> opt-out actually quit.
2026-05	#416, PR #589	Hover-raise under focus-follows-mouse got a <code>webContents.focus()</code> suppressor.
2026-05	#605, PR #645	The sleep inhibitor that never released got the <code>CLAUDE_KEEP_AWAKE</code> escape hatch.
2026-05	#630, PR #642	Alt toggle moved to keyup so Alt+Shift and Alt+F4 could complete.
2026-05	#481, PR #489	Upstream's migration of the About window to <code>titleBarStyle: 'hiddenInset'</code> stopped <code>isPopupWindow()</code> matching, showing a minified-JS error; Hayao0819's PR carried the fix.
2026-05	PR #538	Hybrid-titlebar machinery added (<code>CLAUDE_TITLEBAR_STYLE</code>).
2026-05	PR #564	In-place-upgrade watcher on <code>app.asar</code> added.
2026-05	#567	<code>autoUpdater</code> no-op Proxy added, defending against what its issue called a happy accident.

Fate under the rebase. **DELETE** (frame core) **VERIFY** (accreted fixes) The accessibility reassessment splits the verdict. The frame-core slice is byte-moot and its deletion stands: “The only frame:!1 sites are the Quick Entry popup and two transparent overlay windows, intentionally frameless on every platform. The main window omits frame (system frame),” and the updater bootstrap early-returns with apt_channel_pending. But the same 997-line file carried roughly eighteen accreted Electron-runtime fixes whose root cause is *unfixed upstream*, not a mis-sourced Windows asar: hover-raise under focus-follows-mouse (#416), the sleep inhibitor that never releases (#605), openAtLogin non-persistence (#128), and the CLAUDE_QUIT_ON_CLOSE quit hatch (#623). Re-verified against pristine official 1.18286.0, probe_frame_fix() in tools/patch-necessity-audit.sh returns check, not a clean delete (“frame:!1 occurs 3x, confirm Linux reachability”), so the accreted slice moves to **VERIFY**: open check **FF-1** (unpatched official on a focus-follows-mouse WM) gates whether those fixes must return. This annotates residual risk; it does not un-ship the deletion. On rebase/official-deb the wrapper, entry heredoc, and main swap were all deleted in d9cef9e; app-asar.sh runs only the two survivors, and the doctor’s _check_legacy_env() warns on CLAUDE_TITLEBAR_STYLE, CLAUDE_MENU_BAR, and CLAUDE_KEEP_AWAKE but still omits CLAUDE_QUIT_ON_CLOSE (follow-on LD-2).

Affects. All desktops: GNOME and KDE, X11 and Wayland, wlroots compositors (Hyprrland, Niri, Sway). The suite’s broadest footprint; FF-1’s populations are Sway, i3, Hyprrland, and Niri (raise-on-hover) plus all GNOME and KDE (sleep inhibitor, quit accessibility).

The Native-Binding Stub: Eighteen Months of Pretending

MECHANISM

Unlike the sed-based patches, scripts/claude-native-stub.js never touches the minified bundle. It is a **module-resolution shadow**: the bundle’s require("@ant/claude-native") is satisfied by a drop-in index.js that the build copies into both load contexts, inside the asar (scripts/patches/app-asar.sh) and into the unpacked tree (scripts/staging/electron.sh), because the Windows .node binary at that path cannot load on Linux. No anchors, no identifier extraction; idempotency is mkdir -p plus an overwriting cp. The coupling: upstream null-guards only the module, never individual methods, so the stub must export everything upstream calls or a call throws during top-level execution.

ORIGIN

The stub is as old as the repository, born in the initial commit of December 2024 as two heredocs in build-deb.sh: a frozen 19-entry KeyboardKey enum, a fixed getWindowsVersion: () => "10.0.0" spoof, and pure no-ops otherwise. It predates the tracker; the README credits k3d3’s flake for the bindings insight (the specific key codes matching k3d3’s work is inference, not a verified byte-trace). Growth came in waves: the AuthRequest class arrived to force login into the system browser (#121, Google login spinner), PR #143 (jacobfrantz1) retargeted the stub to the @ant/ namespace, and PR #228 (@milog1994) later made three methods genuinely functional.

REVISION HISTORY

WHEN	REFS	EVENT
2024-12	d8f4bbe	Origin: initial commit embeds the stub twice as heredocs in build-deb.sh, targeting the un-namespaced node_modules/claude-native/index.js; KeyboardKey enum, getWindowsVersion spoof, pure no-ops, no getWindow(), no AuthRequest.

REVISION HISTORY CONTINUED

WHEN	REFS	EVENT
2025-11	#121, ae6d3a3	The AuthRequest stub (isAvailable() false, throwing start()) forces Google login to fall back to the system browser, fixing the indefinite loading spinner; direct commit to main, no associated PR.
2025-11	PR #143, 5c5eb39 (@jacobfrantz1)	Both heredoc destinations retargeted from node_modules/claude-native/ to node_modules/@ant/claude-native/, tracking upstream 1.0.1307's package rename; same-day follow-up 1405e1c fixed a missing mkdir -p for the new namespaced directory.
2026-01	PR #180/#181, 4bf5986	Heredoc extracted to the standalone scripts/claude-native-stub.js, duplicated second definition removed; one source file thereafter, copied to both destinations.
2026-02	PR #228, 83cbb9a (@milog1994)	getIsMaximized, flashFrame, and setProgressBar made genuinely functional via a new getWindow() helper, fixing #149 (KDE Plasma attention flash).
2026-02	PR #232, 2245808, 75841f0	Hardening series: destroyed/invisible windows filtered out of the getWindow() fallback, then the isVisible() filter deliberately dropped so flashFrame() keeps working on minimized windows, its primary use case.
2026-04	ff4821e	Pure move: injection logic lands in scripts/patches/app-asar.sh (patch_app_asar) and scripts/staging/electron.sh (finalize_app_asar); no behavior change.
2026-06	#729, PR #737, 295d71b	Upstream ≥ 1.13576.0 calls readRegistryValues() and getWindowsElevationType() unconditionally at startup; the old stub threw, an empty uncaughtException handler swallowed it, and the app hung windowless. PR #737 consolidated #734 (@chrisw1005) and #730 (@colonelpanic8) into five neutral registry no-ops plus tests/claude-native-stub.bats, crediting both authors.

Fate under the rebase. **DELETE** The matrix verdict is unconditional: “**delete** ...Real Rust NAPI ELF at resources/app.asar.unpacked/node_modules/@ant/claude-native/claude-native-binding.node.” That binding is a 1.65 MB NAPI-RS Rust ELF with genuine X11 input injection (enigo plus x11rb/XTEST), rustix/openat2 containment, and SO_PEERCREDS peer auth, capabilities a JS stub could never fake; tools/patch-necessity-audit.sh confirms the ELF and reports not-needed. At d9cef9e the stub and its bats suite are deleted, neither injection site survives, and the wholesale ar p | tar extraction plus a repack that hard-fails on unpacked-set divergence ships the binding untouched. Known-stale by design: tests/test-artifact-common.sh still asserts index.js exists; that rework belongs to @sabiut, with the test-artifacts CI jobs expected red until it lands.

Affects. All Linux; a module-resolution shadow, DE-agnostic.

node-pty: Compiling What Windows Would Not Ship

MECHANISM

This unit is **build-time provisioning, not a bundle sed patch**: it touches no minified JavaScript. Its job on main is to put a Linux-native pty.node where the Windows-installer app.asar expects one, so the Code tab’s terminal (startShellPty, which dynamically imports node-pty as an optional dependency) can spawn shells. Four pieces cooperate: install_node_pty() in cwork.sh either consumes a pre-built tree via --node-pty-dir or npm-installs into an isolated directory; patch_app_asar() injects optionalDependencies['node-pty'] so the import resolves; finalize_app_asar() packs with --unpack '*/*.node'; and nix/node-pty.nix builds node-pty v1.1.0 from source with three commented upstream workarounds.

ORIGIN

The origin is #152 (add node-pty for terminal integration), merged in January 2026: macOS shipped node-pty as an optional dependency, but the repackaged Windows app.asar carried no Linux pty.node. No motivating issue exists; the PR reads as self-initiated, though that is inference. @typedrat's Nix flake PR #266 (March 2026) briefly consumed a nixpkgs node-pty that turned out not to exist, replaced within the PR by nix/node-pty.nix and the --node-pty-dir wiring.

REVISION HISTORY

WHEN	REFS	EVENT
2026-01	#152, 3591392	Origin: install_node_pty() and the optionalDependencies['node-pty'] edit land directly in build.sh; motivation is the macOS optional-dependency precedent, since the repackaged Windows app.asar shipped no Linux pty.node; no motivating issue number, reads as self-initiated.
2026-03	#266, caa58ca	Nix flake integration (@typedrat): a nixpkgs node-pty input is added, then removed within the same PR after proving not to exist; nix/node-pty.nix built from source instead (v1.1.0, three commented workarounds), wired through --node-pty-dir.
2026-04	#421, 3150477	Asar-manifest unpack fix (@Joost-Maker): no manifest entry for node-pty/build/ meant Electron's .unpacked redirect never fired, and the require died with MODULE_NOT_FOUND on every Code-tab shell session; install_node_pty() now stages build/, and finalize_app_asar() packs with --unpack '**/*.node'.
2026-04	#432/#438, 50b10ed, e92aea1	Nix permissions (@typedrat): the #421 --unpack pass preserved Nix-store read-only bits, breaking the follow-up copy; #432 chmods the extracted files, #438 retires the chmod for cp --no-preserve=mode at the source.
2026-05	#401, #598/#597	Fedora ships Windows node-pty binaries (#401): the origin's soft-warning design let a failed npm install through, shipping winpty.dll and PE32+ binaries unchanged; @JoshuaVlantis closes both halves via #598 (fail loudly on install failure) and #597 (wipe the upstream Windows tree before staging).
2026-06	#727/#728, #761	Open (@EtherAura): node-pty 1.1.0 also forks the shell through a separate build/Release/spawn-helper executable the legacy build never shipped (#727), and the bundle hardcodes powershell.exe (#728). @LiukScot's #761 targets nix/node-pty.nix and cwork.sh, files the rebase deletes or parks; its disposition is unrecorded.

Fate under the rebase. **DELETE** The matrix verdict is an unconditional "delete": "Prebuilt prebuilds/linux-x64/pty.node ships in app.asar.unpacked." The official 1.17377.2 .deb carries a pre-built Linux ELF, in node-pty's newer prebuilds/ layout rather than the legacy build/Release/ path, verified by probe_node_pty() in the audit tool. Commit d9cef9e deletes the whole unit: nix/node-pty.nix gone, install_node_pty nowhere, --node-pty-dir dropped. The new repack derives its --unpack glob from the shipped app.asar.unpacked tree with a post-pack set-equality check, preserving the official placement. One hedge: the delete rests on byte presence of the ELF, not a live shell-spawn test, so whether the official build moots #727, #728 is unverified.

Affects. All Linux; build-time provisioning, DE-agnostic, format-general (deb, RPM, Nix).

Tray: Five Anchor-Rot Cycles Toward Upstream's Own Design

MECHANISM

scripts/patches/tray.sh is four functions against the minified main-process bundle, three of them tray

proper. `patch_tray_menu_handler` finds the rebuild function through the `on("menuBarEnabled", ()=>{FUNC()})` anchor, rewrites it `async`, injects a **trailing-edge mutex** (`_running/_pending`, 1500ms) so a rebuild request arriving mid-flight is remembered and the final `nativeTheme` value wins, and adds a 250ms `post-destroy()` sleep so the `StatusNotifierItem` can unregister before a new `Tray` registers. `patch_tray_icon_selection` rewrites the hardcoded `TrayIconTemplate.png` assignment into a `shouldUseDarkColors` conditional (the icons themselves were made opaque at build time, since the originals are macOS ~20%-opacity templates Linux never colorizes). `patch_tray_inplace_update` extracts five minified locals and splices a fast-path ahead of the destroy-and-recreate block: if a tray exists and the feature is enabled, `setImage` plus `setContextMenu` in place, return. The fourth resident, `patch_menu_bar_default`, captures the preference variable off the "menuBarEnabled" string literal and rewrites the gate's `,!VAR)` coercion to `VAR!==false`, so a missing config key means tray on rather than `!!undefined` meaning tray off. Two `_common.sh` helpers, `extract_electron_variable` and `fix_native_theme_references`, serve the family; the latter exists to repair an *upstream* minifier bug class where the bundle referenced `oe.nativeTheme` while electron was bound to `Ae`.

```
if(TRAY && ENABLED!==false){
  TRAY.setImage(EL.nativeImage.createFromPath(PATH));
  process.platform!="darwin" && TRAY.setContextMenu(BUILDER());
  return
}
```

ORIGIN

The tray arc opened in November 2025 with #135 (Wayland Dbus "already exported", a non-functional tray menu): commit `1bb05df` added the mutex and delay with hardcoded names, and `c660bf1` made them dynamic the same day. The `menubar-default` patch is the family's quieter passenger, born in `7bd2f38` against #218's twin complaint: the `!!undefined` gate plus updaters dropping the key from `config.json`. Across the life of the file the suite absorbed **five anchor-rot re-derivations**, called out below.

REVISION HISTORY

WHEN	REFS	EVENT
2025-11	#135, #138 1bb05df; #139 c660bf1	Origin: <code>async</code> rewrite, mutex, and post-destroy delay against the Dbus double-export; hardcoded minified names made dynamic the same day.
2026-01	#158, bed0cc3	Anchor rot 1: upstream 1.0.3218 added an early return before the first <code>const</code> ; the mutex insertion re-anchored on the function brace (@lizthegrey).
2026-01	#163, e5d58f2, 6916a9e	Icon selection plus build-time opacity processing against invisible template icons; the light/dark logic inverted on day one and corrected the next; mutex widened to 1500ms and the post-destroy delay to 250ms after a first attempt (#164) leaned on a Chromium flag that "doesn't exist in Electron/Chromium".
2026-02	#218, #219, 7bd2f38	Upstream's own <code>oe/Ae</code> minifier bug in v1.1.2321; birthed both <code>fix_native_theme_references</code> and the <code>menuBarEnabled</code> default patch.
2026-02	#252, 546f845	Anchor rot 2: v1.1.4088 minified the electron binding to <code>\$e</code> and <code>\w</code> -only capture spliced code mid-name, a launch-time <code>SyntaxError</code> ; anchors widened to accept dollar identifiers.
2026-04	#515, cf2b0fc	The in-place <code>setImage/setContextMenu</code> fast-path ahead of destroy-and-recreate: the destroy-plus-delay race is structural on KDE Plasma, the old SNI staying registered while the new one appears, two icons until logout (@IliyaBrook; diagnosis preserved in <code>docs/learnings/tray-rebuild-race.md</code>).

REVISION HISTORY CONTINUED

WHEN	REFS	EVENT
2026-05	#625, #627 6219d5a; #644 b40441c	Anchor rot 3: 1.8089.1's i\$A handler beat PCRE \w; [\$\w]+ then became convention repo-wide in the #644 regex audit (@typedrat).
2026-05	#656, e38066e	Anchor rot 4: 1.9255.0 reshuffled declarations; the tray variable re-anchored on the Tray constructor literal.
2026-06	#679, #680, e13e331	Mutex converted to trailing-edge after dark-GNOME startups latched a stale shouldUseDarkColors=false and stuck the icon black (@LiukScot).
2026-06	#750, 6091615, eb12ad8	Anchor rot 5: the 1.13576+ "yukonSilver" rebuild refactor silently re-armed the #515 race through green CI; the resolver learned to skip the setContextMenu(null) decoy and assert exactly one TRAY.destroy() anchor, and the menubar patch learned the new upstream defaults map, warning loudly on a genuine anchor miss instead of skipping.

Fate under the rebase. **DELETE** Both rows are unconditional deletes against official 1.17377.2. The tray verdict: "Official rebuild takes an in-place setImage branch keyed on icon-path change; Tray.destroy() only runs when the user disables the tray. No SNI re-registration gap exists," and the Linux branch "natively selects TrayIconLinux(-Dark).png" (the TrayIconTemplate.png anchor survives only in the macOS branch). Upstream independently converged on exactly what patch_tray_inplace_update injected, purpose-made opaque icons included: a validation of the community diagnosis. The menubar row: "Defaults map ships menuBarEnabled:!0," the same Bpt={menuBarEnabled:!0} bytes first captured in #750, so an unset preference already resolves true. Commit d9cef9e on rebase/official-deb deleted tray.sh, _common.sh, tests/tray-patches.bats, and the ImageMagick icon pipeline; active_patches carries no tray entry, and #750 is mooted by deleting the patch it tracks. One hedge from the verification record: fix_native_theme_references has no explicit matrix row or audit probe, so its deletion rests on tray.sh's wholesale removal rather than a byte-level check that 1.17377.2 is free of the #218-class bug.

Affects. KDE Plasma and GNOME (SNI-tray desktops); Wayland and X11, where the DBus/SNI re-registration race lived.

Quick Entry: The One-Line Hack That Refused to Die

MECHANISM

patch_quick_window in scripts/patches/quick-window.sh is a two-part runtime patch of the minified main-process bundle (app.asar.contents/.vite/build/index.js) that works around Electron's stale BrowserWindow.isFocused() on Linux/KDE, so the main window actually reappears after a Quick Entry submit. Part one anchors on the unique setAlwaysOnTop(!0, "pop-up-menu") call to extract the popup's minified variable, then rewrites the ||VAR.hide() short-circuit into a desktop-environment ternary: when XDG_CURRENT_DESKTOP contains kde, blur() runs before hide(), forcing isFocused() honest. Part two anchors on two surviving log strings, 'Navigating to existing chat' and 'Creating new chat with submit_quick_entry', and within 1500 characters of each swaps upstream's **don't-show-if-already-focused gate** from the focus check to a visibility check, again KDE-only. Both halves carry idempotency guards, and a failed symbol extraction exits nonzero so the build prints a warning rather than silently passing (hardened in May 2026).

```
||((process.env.XDG_CURRENT_DESKTOP||"").toLowerCase().includes("kde")
  ? (VAR.blur(),VAR.hide()) : VAR.hide())
```

ORIGIN

The origin was six lines. In December 2025, @malins reported #144 (Quick Entry submit silently disappears, Kubuntu/KDE); jacobfrantz1’s PR #147, merged in January 2026, added a literal `s/e.hide()/e.blur(),e.hide()/` to `build.sh`, hardcoding the minified variable `e`.

REVISION HISTORY

WHEN	REFS	EVENT
2026-01–2026-04	#144; #147	That seed rotted: an unrecorded upstream re-minification renamed <code>e</code> , and the patch silently no-oped, quietly regressing #144.
2026-04	PR #390, 32660be	Rewrote the patch with dynamic symbol extraction and added the second, <code>show()</code> -site half, then promptly regressed GNOME.
2026-04	#393; PR #406, ab33960	@Andrej730 confirmed in #393 (Quick Entry broken on Ubuntu 24.04) that removing the patch restored Quick Entry there, so PR #406 gated both halves to KDE; the commit called it “a temporary gate” pending a bisection of which half regresses GNOME. No such bisection landed: #393 remains open, and the gate persisted as-is.
2026-04	PR #420; PR #496 (fixes #495), d4db728	@Andrej730 stayed the unit’s steward: a readability refactor in PR #420, then PR #496 fixing #495 (patch partially failing) after upstream 1.3883.0’s minifier hoisted a <code>var e;</code> declaration that broke <code>visFnRe</code> ; the fix made the prefix optional, verified end-to-end on Nobara KDE Plasma 6 Wayland.
2026-05	PR #644, b40441c	Identifier hardening closed the arc, applying the guidelines chronicled in the chassis section: <code>\w+</code> became <code>[\$\w]+</code> , and whitespace tolerance joined the <code> hide()</code> anchors.

Fate under the rebase. **VERIFY** The accessibility reassessment moves this from survivor candidate to **VERIFY** (keep-pending repro). Re-verified against pristine official 1.18286.0, the quick-window variable is now `ms`: the `||VAR.hide()` anchor is present with the `VAR.blur()` count still zero, so the `stale-isFocused()` condition is structurally unchanged in the official Linux build (the `var` was `Ns` on 1.17377.2; the anchor survived the bump). `probe_quick_window` in `tools/patch-necessity-audit.sh` reproduces this mechanically. The defect lives in Electron’s Linux `isFocused()`, not in app bytes, so the bytes cannot prove it still reproduces: whether the KDE stale-focus bug actually fires against the official build is a static inference, not a runtime observation. The patch therefore stays wired as one of exactly two survivors in `active_patches` on `rebase/official-deb` (byte-identical to `main`; the Phase 2 build against 1.17377.2 amd64 extracted the symbol plus both `show()` anchors and applied cleanly); open check **QW-1** (KDE Plasma X11 and Wayland repro) decides keep-versus-drop. Deletion is clearly wrong, the #144 regression being historically confirmed; the open question is keep-still-helps versus keep-now-a-liability if Electron has quietly made `isFocused()` honest.

Affects. KDE Plasma only by design (the Electron `stale-isFocused()` bug); GNOME regressed and was gated out (#393/#406).

Claude Code: Two Lines in a Platform Switch

MECHANISM

`scripts/patches/claude-code.sh` is a single 29-line function, `patch_linux_claude_code`, invoked from `patch_app_asar` under the comment `# Add Linux Claude Code support`. Its target is upstream’s `getHostPlatform()` resolver in the minified main-process bundle, the switch that decides which Claude Code agent binary to fetch for the Code tab. Historically that switch returned only `darwin-*` and `win32-*` and threw `Unsupported platform: linux-x64` on Linux; the patch **splices Linux return branches into the switch**. An

idempotency guard greps for an existing Linux branch first; then a "new format" path anchors on the arch-aware win32 return, captures the minified arch variable with (`[[:alnum:]]_+$`), and inserts before the `throw`:

```
| if(process.platform==="linux")return \1==="arm64"? "linux-arm64": "linux-x64";
```

A legacy path handles the older single-win32-x64 shape using `process.arch` directly, and a non-fatal warning fires if neither anchor matches (leaving the Code tab broken at runtime rather than aborting the build).

ORIGIN

The patch was born with upstream 1.0.1307, the release that introduced the Code preview. In #143 (Code-preview support, merged November 2025) jacobfrantz1 added the original inline guard-plus-sed to `build.sh`; no motivating issue exists, and the rationale is reconstructed from the PR body alone.

REVISION HISTORY

WHEN	REFS	EVENT
2025-11	#143, 5c5eb39	Origin: inline guard-plus-sed spliced Linux return branches into <code>getHostPlatform()</code> inside <code>build.sh</code> , adapting to upstream 1.0.1307's new Code preview (jacobfrantz1).
2026-02	#241, #243 db11bd0	Anchor rot: upstream 1.1.3541 refactored <code>getHostPlatform()</code> for Windows arm64 support, rotting the origin anchor so the sed silently no-matched; @noctuum reported the Code tab completely broken. @IliyaBrook's fix introduced the dual-format detection, a broader idempotency guard, and the warning fallback.
2026-05	#579, 3506c14	Test harness pinned the <code>linux-arm64</code> fingerprint as a build regression check.
2026-05	#644, b40441c	Regex-hardening audit: <code>\w+</code> widened to <code>[\$\w]+</code> so dollar-containing minified identifiers are not truncated.

Fate under the rebase. DELETE The matrix verdict is unconditional: "delete: `getHostPlatform` has a native `linux-x64/linux-arm64` branch." The recorded 2026-07-02 audit of official `.deb` 1.17377.2 found the official bytes satisfying the legacy patch's own idempotency-guard string, `process.platform==="linux".*linux-arm64.*linux-x64`, natively; the teardown likewise records prebuilt `node-pty` (`linux-x64`) shipping for Code/agent shells. On `rebase/official-deb` the file is gone, deleted in `d9cef9e` among the eleven condemned patches, with no `claude-code` entry in `active_patches` and no launcher or doctor compensation anywhere.

Affects. All Linux; the Code-tab platform switch, DE-agnostic.

Cowork: Impersonating a VM Service, and Parking It

MECHANISM

Cowork is the suite's largest unit and the only one with a runtime component of its own. `patch_cowork_linux()` in `scripts/patches/cowork.sh` runs a single embedded node heredoc against the minified `index.js`, version-guarded on the literal `vmClient` (TypeScript), and lands roughly a dozen numbered patches that **reroute the Windows VM client to Linux**: the `startVM` support gate is widened past the `yukon-Silver` feature check (FATAL on miss), the support evaluator is flipped so the renderer un-grays the tab, the multi-GB rootfs download is held disabled, the Windows named-pipe string becomes a ternary pointing at `$XDG_RUNTIME_DIR/cowork-vm-service.sock`, empty `linux:{x64:[],arm64:[]}` manifest arrays ex-

`exploit [].every()` vacuous truth, and the keystone patch forks a daemon from `app.asar.unpacked` on `ENOENT/ECONNREFUSED` with a 10-second respawn cooldown, plus a quit handler that `SIGTERMs` it. The other half is `scripts/cowork-vm-service.js`, a 2,766-line daemon that **impersonates the Windows VM service** over that Unix socket, speaking the same 4-byte length-prefixed JSON protocol as the named pipe: a `VMManager` over `Host`, `Bwrap`, and `KVM` backends, `bubblewrap` by default and `KVM` opt-in via `COWORK_VM_BACKEND=kvm`, unpacked because `child_process.fork` cannot execute inside an `asar`.

The same file carries two smaller passengers, the `asar-path` guards, addressing a shared root cause: all four legacy launchers redundantly appended the `app.asar` path to Electron's `argv`, and Electron's ASAR virtual-filesystem shim reports archives as directories, so the app dispatched its own bundle to Cowork as a folder drop. `patch_asar_path_filter` rewrites the directory-check helper, capturing function, parameter, and `fs` variable dynamically:

```

  /function\s+([\w$]+)\s*\(\s*([\w$]+)\s*\)\s*\{\s*try\s*\{\s*
  return\s+([\w$]+)\.statSync\(\s*\2\s*\)\.isDirectory\(\)/

```

`patch_asar_argv_file_drop_guard` covers the second-instance `argv` collector, which had no equivalent guard of its own: it injects the same `.asar`-suffix check ahead of an `existsSync` call, guarded by a uniqueness assertion and a TSV verification marker so a future anchor rot fails loudly instead of silently.

ORIGIN

The reroute's origin is a same-day pivot. A `@ant/claude-swift` stub by `@chukfinley` landed via #198 (Cowork stub PR) on 2026-02-16, shipping a documented KNOWN ISSUE (spawned-process stdout never fired, so Cowork loaded but showed nothing); hours later commit `25e7932` deleted the stub and introduced the daemon-plus-patch architecture that still stands. Upstream Cowork was macOS/Windows-only, so without the unit the mode was dead on Linux. `@RayCharlizard` is the subsystem owner, with a dedicated learning page, `docs/learnings/cowork-vm-daemon.md`.

REVISION HISTORY

WHEN	REFS	EVENT
2026-02	#198 b8b2893; 25e7932	Origin: <code>@chukfinley</code> 's <code>claude-swift</code> stub merged, then deleted hours later for the <code>daemon-plus-patch_cowork_linux</code> architecture (upstream Cowork was macOS/Windows-only).
2026-02	#259, 2017011	Anchor rot: <code>v1.1.4173</code> renamed the platform-gate anchor from <code>Unsupported platform</code> → <code>unsupported_platform</code> , crashing the app at startup; Patch 1 made a hard build error (<code>process.exit(1)</code>) on anchor miss, a property the June re-derivation below depends on.
2026-02	#269, 4929dde	Refactor: the monolithic VM manager split into <code>Host/Bwrap/Kvm</code> backends, <code>COWORK_VM_BACKEND</code> override added, plus a <code>--doctor</code> Cowork section.
2026-03	#265, d7a4606	Guest-path fix (<code>@leomayer</code>): the daemon had been stripping VM guest paths from <code>--plugin-dir/--add-dir</code> , hiding skills and plugins; <code>translateGuestPath()</code> and <code>buildMountMap()</code> translate them instead.
2026-03	#288, PR #300	KVM wire-protocol series (<code>af1f2e3</code> through <code>473b0ba</code> , hardened by <code>932044d</code>): <code>vsock</code> direction/port, <code>virtiofsd</code> readiness, session disk plus <code>smol-bin</code> drive, guest RPC protocol, and a <code>virtio-9p</code> fallback got real VMs booting.
2026-03	#329/#332/#334, aa6b87d; #337, a3190c3	Manifest fix: real Linux CDN checksums caused an infinite download-retry loop as they drifted from CDN content; superseded a day later by empty <code>linux: {x64: [], arm64: []}</code> manifest arrays exploiting <code> [].every()</code> vacuous truth.

WHEN	REFS	EVENT
2026-04	#418, PR #421 2f6194f (@Joost-Maker)	Identifier widening: Claude 1.3109.0 or later renamed the win32 fs var <code>e</code> → <code>\$e</code> , breaking Patch 9 with <code>TypeError: e.existsSync is not a function</code> ; all six extraction regexes widened from <code>(\w+)</code> to <code>[\$\w]+</code> .
2026-06	yukonSilver, PR #736 83ea637; #742, 7327c95	Re-derivation: upstream's yukonSilver VM refactor stalled Patch 1's anchor; because Patch 1 is FATAL, main stayed red from the 1.13576.0 bump on June 17 until the June 23 re-derivation, with support-evaluator patches 1b/1c following two days later.
2026-04-05	#383, #622, #632	Symptom cluster: the ASAR virtual-filesystem shim reports <code>app.asar</code> as a directory, so the launchers' redundant <code>argv</code> path triggered a permission dialog on every launch (#383), forced Cowork mode on every reopen (#622), and a fatal <code>--add-dir</code> error in bundled Claude Code (#632).
2026-05	PR #640, 6bfb296 (@aaddrick)	<code>patch_asar_path_filter</code> lands: rewrites the directory-check helper so an <code>.asar</code> -suffix test short-circuits before <code>statSync().isDirectory()</code> .
2026-05	#668 (@MitchSchwartz); PR #669, 623f1b0	Incomplete-fix regression: the second-instance <code>argv</code> collector had no equivalent guard; <code>patch_asar_argv_file_drop_guard</code> adds one, with a uniqueness assertion and a TSV verification marker.
2026-06	PR #700, ab17b69 (@emandel82)	Root-cause fix: launchers stop passing <code>app.asar</code> on Electron's <code>argv</code> at all; the guards stay on main as defense-in-depth, load-bearing status on the <code>deb/rpm</code> global-Electron fallback left as unverified inference.

Fate under the rebase. **PARKED** The guards are gone and the reroute is shelved, for opposite reasons. The matrix marks the guards **delete**: the official `statSync().isDirectory()` helpers "still exist (3 anchors, no upstream `.asar` guard), but the official launcher is a bare ELF symlink, no `app.asar` `argv` ever reaches them." Commit `d9cef9e` deleted both guard functions and the marker TSV; the new launchers exec the official binary directly, the global-Electron fallback is gone, and zero `endsWith(".asar")` matches remain under `scripts/`. The reroute itself is **park (3.1 track)**: official Cowork is a Go `coworkd` plus QEMU/KVM over a `SO_PEERCRED` Unix socket, so the named-pipe client every anchor greps for no longer exists, and "a `bwrap` fallback now means impersonating that protocol, off the 3.0.0 critical path." The stack moved intact to `scripts/cowork-fallback/` (the reroute, a byte-identical daemon, three bats suites, and a README stating nothing there is executed or patched into any artifact); 3.0.0 ships KVM-only with doctor guidance via `_check_kvm`, `_check_vhost_vsock`, and `_check_cowork_stack`; the launcher keeps `cleanup_orphaned_cowork_daemon()` to reap leftover 2.x daemons; and the `cowork-bwrapd` investigation against the official protocol is a 3.1 item owned by @RayCharlizard.

Affects. KVM-capable hosts; OVMF/AAVMF firmware-path sensitive (a hardcoded probe list, not distro-portable). The `asar-path` guards were launcher-`argv` defense, all Linux.

Org Plugins: The Two-Commit Survivor

MECHANISM

`scripts/patches/org-plugins.sh` is a single function, `patch_org_plugins_path`, that fixes a platform switch upstream never finished. The minified `index.js` resolves the MDM-managed plugin-marketplace directory (used in third-party inference mode) via `switch(process.platform)` with only `darwin` and `win32` cases; default: returns `null`, which every downstream caller reads as **feature off**. The patch splices a Linux case returning `/etc/claude/org-plugins`, a path the in-file comment defends as FHS-correct for MDM configuration and consistent with Claude Code's `/etc/claude-code/`. Three steps run against `app.asar.contents/.vite/build/index.js`:

an idempotency guard that skips if the injected case already exists ("upstream may add one in the future"), a presence probe on the darwin string Application Support/Claude/org-plugins that warns to stderr and skips gracefully if absent, and the compound insertion anchor, unique to this switch, with \s* tolerating minified and beautified spacing:

```
| sed -i -E 's/(\"org-plugins\")\\s*;\s*(default\s*:\s*return\s+null)/\\1case\"linux\":return\"\\etc\\/claude\\/org-plugins
```

Every anchor is a string literal or keyword, so the patch needs **no dynamic identifier extraction** and is immune to identifier re-minification by construction.

ORIGIN

Born May 24, 2026 (commit 337e9a4) from #607 (org-plugins has no Linux default), opened ten days earlier by @johncrn, who ran upstream 1.7196.0 in 3P inference mode, read the code himself, and asked for exactly this fix; even a symlink to the mac-style folder left the marketplace undetectable. The triage bot corroborated: resolver pD() with no case "linux", and the /etc/claude-code precedent for the path choice. Note the disambiguation: docs/learnings/plugin-install.md covers the adjacent remote-marketplace install gate (#396, Anthropic & Partners plugins), not this unit.

REVISION HISTORY

WHEN	REFS	EVENT
2026-05	#607/#639, 337e9a4	Origin: Linux case spliced into the platform switch, closing #607 the same minute; shipped in v2.0.13.
2026-05	428777a	Same-day fixup (also PR #639): redirected the two skip-path warnings to stderr; rationale unrecorded, plausibly stream discipline.
2026-05–2026-07	#677	Zero anchor repairs ever: the string-literal anchors survived every re-minification from 1.7196.0 through official 1.17377.2; a build log attached to #677 confirms success on 1.9659.2 in passing.

Fate under the rebase. **SURVIVOR** The reassessment firms this from survivor candidate to a settled **SURVIVOR**. Re-verified against pristine official 1.18286.0, the platform switch reads "org-plugins");default:return null with **no linux case** (count 0), so MDM org plugins are dead on Linux upstream and keeping the patch preserves our /etc/claude/org-plugins behavior; keep-cost is near zero (a self-defusing idempotency guard). An earlier audit that reported a “native linux case” was a *patched-tree* false positive, the probe matching our own injected case, which is why the pristine .deb is the byte of record. Commit d9cef9e wired patch_org_plugins_path into the new active_patches array while the module itself stays byte-identical to main. Retirement is designed in: if Anthropic ever adds a Linux case, the idempotency guard and a not-needed probe verdict retire it automatically. One caveat: the app-asar.sh comment says “(filed upstream)”, but no such filing is traceable anywhere; the report remains planned, not proven.

Affects. All Linux; the MDM /etc/claude/org-plugins path, DE-agnostic (third-party inference mode).

The WCO Shim: One Commit, Zero Revisions

MECHANISM

patch_wco_shim (scripts/patches/wco-shim.sh) is the only patch in the suite that rewrites nothing: it performs

a pure prepend, inlining `scripts/wco-shim.js` at the top of `app.asar.contents/.vite/build/mainView.js`, the `BrowserView` preload, guarded by a `grep -q '__claude_wco_shim'` idempotency check and a hard `exit 1` if the target file is missing. The inline (rather than a `require`) is deliberate: sandboxed preloads can only require a fixed allowlist of modules, and a relative `require` aborts the entire preload, taking `desktopBootFeatures` down with it. At runtime the shim, Linux-only and disabled when `CLAUDE_TITLEBAR_STYLE` is native, injects a main-world script via `webFrame.executeJavaScript`. Its components: a diagnostic native-state probe, defensive `navigator.windowControlsOverlay` and `matchMedia` shims, a draggable-stripping `className` intercept, an event nudge, and **the one load-bearing piece**, a page-side `navigator.userAgent` getter that appends "Windows" so the remote `claude.ai` bundle's `isWindows()` gate flips and React renders the desktop topbar (`data-testid="topbar-windows-menu"`). The HTTP request UA is unchanged, "so analytics and anti-bot fingerprints stay honest."

ORIGIN

The unit was born whole in `5c8191e` (May 2026, #538, hybrid titlebar mode): the topbar was never in `app.asar`, since `claude.ai`'s remote React bundle renders it behind four gates, of which only Gate 3, the `/((win32|win64|windows|wince)/i` UA regex, fails on Linux (documented in `docs/learnings/linux-topbar-shim.md`). #127 (speleoalex's native-decorations PR, November 2025) had forced `frame: true`, "which definitively hid the topbar," and the upstream `frameless-plus-WCO` route was independently broken by a Chromium-level implicit drag region on frameless Linux windows (Bug C) that left topbar buttons unclickable. Hybrid mode, system frame plus page-side UA spoof, was the resolution; the same commit deleted the predecessor `patch_titlebar_detection`, which since April 2025 had stripped the `!` from `if(!isWindows && isMainWindow)` in bundled renderer assets, the fix that had closed #45 (missing hamburger menu, boyonglin), an attribution inferred from timing and the issue's closing comment. The zero-revision record that follows is the exception that proves the arms-race rule chronicled in the chassis section: because the shim overrides stable web-platform APIs instead of grepping minified identifiers, upstream re-minification could never break it.

The defensive layer also drove one contribution to Electron itself rather than a workaround in the shim: chasing why `matchMedia("(display-mode: window-controls-overlay)")` stayed false while every other WCO signal reported active, aaddrick filed [electron#51393](#) (Electron never overrides `GetDisplayMode()`, so Blink's media-query evaluator always sees `kBrowser`) and opened the fix in [electron#51396](#). Both are open as of this writing, the pull request carrying backport labels for Electron 41 through 44.

REVISION HISTORY

WHEN	REFS	EVENT
2025-04	#45, <code>d6ed2c8</code>	Predecessor: <code>patch_titlebar_detection</code> stripped the <code>!</code> from <code>if(!isWindows && isMainWindow)</code> in bundled renderer assets, closing #45 (missing hamburger menu, boyonglin; attribution inferred from timing plus the issue's closing comment).
2026-05	#538, <code>5c8191e</code>	Origin: hybrid titlebar mode; the shim's introduction and <code>patch_titlebar_detection</code> 's deletion landed in the same commit. Pickaxe (<code>git log -S</code>) confirms both <code>wco-shim.sh</code> and <code>wco-shim.js</code> carry a single-commit history on main: no anchor rot, ever.
2026-05	#538 (in-thread)	Unreproduced regression: lukedev45 reported partial topbar render on OmarchyOS plus Hyprland; aaddrick failed to reproduce across four scenarios, @typedrat confirmed working on NixOS Hyprland (also GNOME Wayland and Sway in-thread), and no follow-up issue was filed.

Fate under the rebase. **DELETE** (local WCO) **VERIFY** (remote UA gate) The accessibility reassessment splits the verdict. `probe_wco_shim()` returns not-needed, but the audit qualifies it “(mainView refs: 0)”: the official `mainView.js` has no `windowControlsOverlay/isWindows` gating and the main window omits frame entirely, so Bug C cannot arise and the local `frameless/WCO/matchMedia` defensive half is genuinely dead (delete stands). That check reaches only the *local* bundle. The one load-bearing piece, the page-side `navigator.userAgent` spoof that flips `claude.ai`’s remote `isWindows()` UA regex so the desktop topbar (`data-testid="topbar-windows-menu"`) renders at all, depends on server-delivered JS that `.deb` bytes cannot reach, so that half moves to **VERIFY**: open check **WCO-1** (topbar DOM presence on a rebased `ApplImage` or `RPM` build, *not* the official `.deb`) decides delete versus a UA-override-only survivor. This annotates residual risk; the code is already gone. Both files were deleted at `d9cef9e`, the titlebar machinery left the launcher at `cafb4cc`, and `doctor.sh` now warns that `CLAUDE_TITLEBAR_STYLE` “is set but no longer honored since the v3.0.0 rebase.” If the remote bundle still Windows-gates the topbar, v3.0.0 silently loses the hamburger/search/nav bar that v2.x hybrid mode delivered; per the tracking plan, “if missing, it’s an upstream report, NOT a shim revival.”

Affects. All Linux equally: it overrides stable web-platform APIs, so DE- and compositor-agnostic by construction.

Config Writes: One Bug, Three Patches, a Split Verdict

MECHANISM

`scripts/patches/config.sh` carries three functions, all operating on `app.asar.contents/.vite/build/index.js`. `patch_config_write_merge` anchors on the developer log string that survives every minifier pass, the central write site `await WRITE_FN(PATH, CONFIG)`, `LOGGER.info("Config file written")`, extracts the minified names with three chained `grep -oP` passes, then uses `node -e` to prepend a merge statement: **re-read the config from disk on every write**, take on-disk `mcpServers` as the base, and let in-memory entries override matching keys, so externally added servers survive writes made from the app’s stale in-memory cache. `patch_asar_trusted_folder_guard` anchors on the unminified `async addTrustedFolder(` declaration and injects `if(PARAM.endsWith(".asar"))return;` at the body head. `patch_asar_additional_dirs_guard` filters every `--add-dir` dispatch loop (for-of and `.forEach` variants) plus a best-effort session-restore self-heal keyed to the “Filtering out deleted folder from session” string.

```
| 'await \K[$\w]+(?:=\([$\w]+,\s*[$\w]+\))\s*,\s*[$\w]+\.\info\("Config file written"\))'
```

ORIGIN

The unit was born from #400 (“`claude_desktop_config.json` being reset continuously”, opened April 2026 by @davidcim): every start or mode switch rewrote the config with stale content, making MCP servers impossible to add; the same doctor paste showed `app.asar` recorded in `localAgentModeTrustedFolders`, the amplifying symptom the trusted-folder guard targets. Commit `364147e` (PR #643) created both functions, diagnosing an upstream writer that **caches parsed config and never re-reads disk before writing**. Days later @beneshengineering filed #649 (asar reaching `--add-dir` despite the #640 cowork guard): corrupted pre-#640 sessions survived restore via Electron’s ASAR VFS shim and crashed local agent mode under bundled Claude Code 2.1.111, so PR #650 added the additional-dirs guard at “the single convergence point for ALL code paths that feed additionalDirectories”.

WHEN	REFS	EVENT
2026-05	#400, 364147e (PR #643)	Origin: created patch_config_write_merge and patch_asar_trusted_folder_guard against the stale in-memory cache overwriting mcpServers on every config write (@aaddrick).
2026-05	#649, 4451694 (PR #650)	Added patch_asar_additional_dirs_guard: filters every --add-dir dispatch loop and self-heals session restore, after corrupted pre-#640 sessions kept reaching additionalDirectories despite the existing guards (@beneshengineering).
2026-06	#674/#685, 2ede75d	Anchor rot: upstream 1.10628.0 folded the trusted-folder guard's log-line anchor into a comma expression (the trailing) ; became) , , hard-failing the build; @maplefater's PR #685 re-anchored on the method declaration itself, superseding @mhentschke's identical #674, closed unmerged two minutes after #685 landed.
2026-06	#718/#722/#723, 5e4f26b	Anchor rot: upstream 1.12603.1 bundled the Claude Code SDK per-panel, producing two identical --add-dir dispatch loops and tripping the single-match assertion (#718, nix build failure, reported by @fdnt7); @typedrat's PR #723 reframed the invariant as "every unfiltered --add-dir dispatch must filter .asar paths" via a global replace, with @marveon's competing #722 credited for the diagnosis.

Fate under the rebase. Split. Both .asar guards: **DELETE** "delete", per the matrix, "addTrustedFolder(o) present without a .asar guard, but same reasoning as above: no on-disk .asar argv path exists on Linux": the official launcher is a bare ELF symlink, the reasoning first stated on main in ab17b69. The mcpServers merge: **VERIFY** "verify behaviorally", the suite's one open question. The Config file written anchor is byte-intact on official 1.17377.2 (audited 2026-07-02, reproducible via tools/patch-necessity-audit.sh), which proves structural continuity of the writer, not that the stale-cache bug persists. On rebase/official-deb (d9cef9e), config.sh shrank from 296 to 104 lines, guards and tests/config-patches.bats gone; the merge survives in the file but is kept **unwired**, absent from app-asar.sh's active_patches, pending a reproduction of #400 against a live official install, with an upstream filing due either way. Deferred, not decided.

Affects. All Linux; the config-write merge and argv guards, DE-agnostic.

Acquisition: Fetching Upstream, Reworked for the Official .deb

MECHANISM

scripts/setup/detect-host.sh's detect_architecture is a case on uname -m (x86_64/aarch64) that sets architecture (amd64/arm64) and, in the same branch, hardcodes a versioned downloads.claude.ai/releases/win32/{x64,arm64}/<ver>/Claude-<sha>.exe URL, claude_exe_sha256, and claude_exe_filename. There is no runtime resolution in the build itself: the URL and hash are literal strings, refreshed only by CI. Anything outside x86_64/aarch64 hard-exits. scripts/setup/download.sh's download_claude_installer either copies a local file (the --exe escape hatch) or wgets the hardcoded URL, then calls verify_sha256() and exits on mismatch. The .exe is 7z x-tracted twice, once for the NSIS wrapper and once for the nested .nupkg, with version parsed from the nupkg filename via:

```
| grep -oP 'AnthropicClaude-K[0-9]+\.[0-9]+\.[0-9]+(?=-full|-arm64-full)'
```

A daily cron in .github/workflows/check-claude-version.yml runs scripts/resolve-download-url.py (Playwright, headless Chromium) against the claude.ai redirect endpoints, since the bare redirect sits behind a

Cloudflare bot challenge a plain `curl/wget` cannot pass. The resolved URL and version are diffed against `detect-host.sh`; on a change the workflow `sed`-rewrites the URL/hash lines, recomputes SHA-256 (hex) and Nix SRI (base64) for both architectures, updates `nix/claude-desktop.nix`'s version/hash fields, commits, sets `CLAUDE_DESKTOP_VERSION`, and tags and pushes `v{REPO_VERSION}+claude{CLAUDE_VERSION}`, the push that triggers the release build. `nix/claude-desktop.nix` itself is a `fetchurl` per system (`x86_64-linux/aarch64-linux`) against the same URLs, each pinned with a Nix SRI hash: one resolver, two consumers.

ORIGIN

The true origin is `d8f4bbe` (2024-12-26, the project's initial commit): `build-deb.sh` hardcoded a single unversioned `storage.googleapis.com` URL, `wget`-fetched with no SHA-256 verification at all and no version automation. Every later piece, arch-aware URLs, hash verification, CI-driven bumping, Nix pins, was added afterward, not present at inception. The redirect approach had a same-day false start on 2025-11-28: `5c5eb39` (#143) switched to the `claude.ai/api/desktop/.../redirect` endpoint with a spoofed-browser `curl` header set to dodge Cloudflare, and the very next commit, `1405e1c`, reverted both changes the same day, back to the static URL and plain `wget`. The static-URL approach eventually rotted for real: `#151's 3e813c5` (2026-01-05) replaced it with Playwright-based resolution after the URLs were found "stuck at `v0.14.10`," the exact Cloudflare problem #143 had failed to solve with header spoofing five weeks earlier. SHA-256 verification did not arrive until `#310's 1fc4e83` (2026-03-20), which added `verify_sha256()` and wired it into both the installer and Node tarball fetches; before that commit a corrupted or substituted download would silently proceed into the build.

REVISION HISTORY

WHEN	REFS	EVENT
2024-12	n/a	<code>d8f4bbe</code> : origin; hardcoded, unversioned URL in <code>build-deb.sh</code> ; no hash check, no automation.
2025-08	closes #104	<code>a987938</code> : <code>check-claude-version.yml</code> created, daily cron, <code>CLAUDE_DESKTOP_VERSION</code> var, tag scheme.
2025-11	#143	<code>5c5eb39</code> : switched to the redirect endpoint plus spoofed-header <code>curl</code> ; <code>1405e1c</code> reverted both the same day.
2026-01	#151; #146	<code>3e813c5</code> : Playwright-based <code>resolve-download-url.py</code> added, fixing URLs stuck at <code>v0.14.10</code> ; same day, <code>#146</code> added the <code>--exe</code> escape hatch.
2026-02/03	#266	<code>ff9fd3d</code> → <code>55f7b27</code> : <code>nix/claude-desktop.nix</code> created with per-arch <code>fetchurl</code> plus SRI pins.
2026-03	#310	<code>1fc4e83</code> : <code>verify_sha256()</code> added; installer and Node tarball verified after download.
2026-04	#443	<code>ff4821e</code> plus <code>564f465</code> : <code>build.sh</code> split into <code>scripts/setup/detect-host.sh</code> plus <code>download.sh</code> ; CI paths updated in the same PR.
2026-04	#535	<code>4cc63bf</code> : workflow third-party actions pinned to commit SHAs, post <code>trivy-action</code> supply-chain incident.
2026-07	n/a	<code>9f3820c</code> : Rebase Phase 0; <code>official-deb.sh</code> written, pinned pool paths plus SHA-256, not yet wired into <code>build.sh</code> .
2026-07	n/a	<code>d9cef9e</code> : Rebase Phase 1; <code>build.sh</code> sources <code>official-deb.sh</code> ; legacy download chain deleted, Nix reduced to a stub.
2026-07	n/a	<code>cafb4cc</code> : Rebase Phase 5; <code>check-claude-version.yml</code> rewritten against the official Packages index (no Playwright); <code>mirror-official-deb</code> job added.

Fate under the rebase. **REWORKED** The acquisition target changed from a Windows .exe installer to the official Linux .deb, forcing every piece of the chain to change with it, though the shape (arch-detect, pinned URL/hash, download, verify, extract, daily CI bump, Nix pin) stayed the same. scripts/setup/official-deb.sh replaces download.sh, resolve-download-url.py, and fetch-electron-binary.js outright (commit d9cef9e: "Deleted: download.sh, resolve-download-url.py, fetch-electron-binary.js, setup_electron_asar, scripts/staging/*"). The official .deb is fetched from OFFICIAL_APT_BASE=downloads.claude.ai/claude-desktop/apt/stable, a plain-HTTPS APT pool with no bot challenge, so Playwright is gone: resolve_official_deb() is a curl plus awk parse of the Packages index. It remains SHA-256 pinned by two independent mechanisms: build-time constants checked through the same verify_sha256() helper #310 added, and CI-time reads straight from the Packages index. Extraction is deliberately ar/tar rather than dpkg-deb, so rpm-family and Arch hosts can build too. A new mirror-official-deb CI job uploads the pinned .deb for both architectures to our own GitHub Releases as insurance against the upstream pool rotating a version out. nix/claude-desktop.nix is an honest stub for now: it throws toward the old derivation in git history, with @typedrat named as the pending owner, and the version-bump workflow already detects the stub and skips the SRI step rather than failing.

Affects. Arch-specific rather than distro-specific (amd64/arm64 download URLs); the Nix SRI hashes are the one format-specific piece. DE-agnostic.

15 • LAUNCHER

The Launcher and Doctor: Runtime Policy, Cut Back to Opt-In

MECHANISM

scripts/launcher-common.sh is sourced by all four per-format launchers (scripts/packaging/deb.sh, rpm.sh, appimage.sh, and the Nix wrapper) and itself sources scripts/doctor.sh at its tail; each packaging script's heredoc calls setup_logging, setup_electron_env, detect_display_backend, and build_electron_args before exec'ing Electron. The unit's real surface is the CLAUDE_* environment-flag policy: CLAUDE_USE_WAYLAND (tri-state display-backend override), CLAUDE_PASSWORD_STORE, CLAUDE_GTK_IM_MODULE, and CLAUDE_DISABLE_GPU are resolved directly in launcher-common.sh; CLAUDE_TITLEBAR_STYLE, CLAUDE_MENU_BAR, CLAUDE_KEEP_AWAKE, and CLAUDE_QUIT_ON_CLOSE are instead enforced one layer down in frame-fix-wrapper.js, but are logged by log_session_env and surfaced by run_doctor, so they remain launcher/doctor policy even though the enforcement point sits in the preload wrapper. build_electron_args accumulates every Chromium feature request into one bash array and emits a single --enable-features= switch, with an explicit comment that Chromium honors only the *last* such switch on a command line, so independent call sites (hidden-titlebar mode, native Wayland, GlobalShortcutsPortal) must never each emit their own. _detect_password_store probed kwallet6 then gnome-libsecret then a fixed basic fallback; setup_logging/log_message write to ~/.cache/claude-desktop-debian/launcher.log, and log_session_env records every CLAUDE_* flag on each launch, unset values included, so bug reports carry context without a round trip.

ORIGIN

The Wayland default oscillated three times before settling. PR #102 flipped the launcher's default from forced X11 to native Wayland plus GlobalShortcutsPortal, betting on portal backends the commit message itself called still in development; that bet did not pay off, and fixes #141 reverted the default back to XWayland, introducing CLAUDE_USE_WAYLAND as the opt-in escape hatch. PR #690 later routed GNOME Wayland global shortcuts

through the XDG GlobalShortcuts portal and made CLAUDE_USE_WAYLAND tri-state, but deliberately did not re-flip GNOME's default to native, since the route is a no-op on GNOME 50 anyway: Chromium never calls the portal's Registry.Register(app_id) handshake, filed upstream as electron/electron#51875 and confirmed still OPEN.

REVISION HISTORY

WHEN	REFS	EVENT
2025-06	PR #97	Forces X11 and disables GPU on Wayland sessions to fix a crash; pre-refactor, lived directly in build-*.sh.
2025-08	PR #102, fixes #93	Reverses the default to native Wayland plus GlobalShortcutsPortal, betting on portal backends still "in development".
2026-01	fixes #141; part of #179	Reverts the default back to XWayland, introduces CLAUDE_USE_WAYLAND as opt-in; ee5e090 extracts scripts/launcher-common.sh from the two per-format build scripts.
2026-02	PR #228, fixes #84, #149, #172, #223, #226; PR #232; fixes #250, follow-up #251	Auto-forces native Wayland for Niri/Sway/Hyprland, then scoped back to Niri only (Sway/Hyprland already set the variable); CLAUDE_MENU_BAR introduced against KDE Plasma's Alt-toggle layout shift, with validation and doctor integration.
2026-03	PR #267	--doctor introduced (ten checks), fixing recurring support issues #247, #261/#263, #264/#216.
2026-04	PR #397; PR #475, fixes #319; PR #451; PR #538	Exports GDK_BACKEND=wayland so a system-wide override cannot blur HiDPI rendering; XRDP sessions auto-disable GPU compositing; hide-to-tray on close becomes the Linux default with CLAUDE_QUIT_ON_CLOSE=1 as the restore-old-behavior hatch; CLAUDE_TITLEBAR_STYLE/hybrid mode (detailed in the WCO-shim unit); --doctor extracted into its own file.
2026-05	PR #585, mitigates #583; PR #570, #571/#572; PR #611, fixes #593; PR #614, fixes #590; PR #624, fixes #623; PR #645, fixes #605; PR #666	CLAUDE_DISABLE_GPU opt-in plus doctor's recent-crash surfacing; log_session_env env-dump block; CLAUDE_GTK_IM_MODULE override and IBus/GTK doctor diagnostics; CLAUDE_PASSWORD_STORE plus the kwallet6/gnome-libsecret/basic keyring probe; doctor's eCryptfs NAME_MAX warning; the CLAUDE_QUIT_ON_CLOSE=1 hatch fixed to call app.quit(); CLAUDE_KEEP_AWAKE=0 plus a powerSaveBlocker logging shim; sticky GPU-FATAL auto-recovery on the next launch.
2026-05	"Ref #635"; fixes #652, refs #647	--class=Claude added but inert (Electron ignores it); WM_CLASS centralized to one build-time source of truth after duplicate taskbar icons reported on Arch and Ubuntu.
2026-06	PR #690, refs #404; PR #712, fixes #711; PR/issue #700	GNOME portal wiring, tri-state CLAUDE_USE_WAYLAND, and the --enable-features merge fix; doctor's dual-package-manager version fix; app.asar dropped from argv, re-keying the live-UI cmdline fingerprint onto --class.
2026-07	rebase Phase 1+2 (d9cef9e); Phase 4+5 (cafb4cc)	frame-fix-wrapper.js deleted wholesale; the launcher execs the official binary directly under an opt-in-only policy, and doctor gains six new checks (see fate banner).
open	PR #738	Proposes dropping --disable-software-rasterizer from the GPU-disable flag set; open and unmerged against main.

Fate under the rebase. **REWORKED** Commit `cafb4cc` (Phases 4+5) reworks this unit around a single new policy line: the packaged app is now Anthropic's official Linux build, so the launcher must not pass any default flag that shadows an official upstream code path. All four launcher variants and the doctor surface survive as live, tested code, but the policy inverts from "ship sensible Linux defaults" to opt-in-only; the launchers now exec the official binary directly rather than launching Electron against a co-located `app.asar`. The titlebar machinery (`_resolve_titlebar_style`, `CLAUDE_TITLEBAR_STYLE`, the `CustomTitlebar/WindowControlsOverlay` split) and the `_detect_password_store` auto-probe are both deleted outright; `--password-store` now emits only when `CLAUDE_PASSWORD_STORE` is explicitly set, ceding the decision to the official build's own `os_crypt` autodetection. Doctor gains six checks: `_check_kvm` and `_check_vhost_vsock` (Cowork on the official client is KVM-only), `_check_cowork_stack`, `_check_official_drift`, `_check_name_collision` (guards against both this project's and Anthropic's own APT repo being configured at once), and `_check_legacy_env`. One verified gap: `_check_legacy_env` warns on `CLAUDE_TITLEBAR_STYLE`, `CLAUDE_MENU_BAR`, and `CLAUDE_KEEP_AWAKE` being set but no longer honored, yet says nothing about `CLAUDE_QUIT_ON_CLOSE`, even though its backing logic (the `frame-fix-wrapper.js` close-event override) was deleted in the same sweep that killed the other three's; whether this is a deliberate call (the official binary may already handle close-to-tray natively) or a doctor-rework oversight was not established. The Nix launcher variant does not currently exist to inherit any of this: `nix/claude-desktop.nix` is a 23-line throw stub pending an official-deb rework, tracked as a spike for @typedrat.

Affects. All desktops (env-flag policy); the environment-specific edges are the Wayland opt-in (GNOME portal), XRDP sessions, Niri/Sway focus, and GPU-fatal recovery on Fedora.

SSH Helpers: Staged for Windows, Gone with the Pipeline

MECHANISM

`scripts/staging/ssh-helpers.sh`'s `copy_ssh_helpers` copies the upstream `claude-ssh` remote-helper binary and its `version.txt` out of the Windows-installer extraction tree (`lib/net45/resources/claude-ssh`, the Squirrel/.NET layout the pre-rebase acquisition pipeline extracted from) into `$electron_resources_dest/claude-ssh`, staging only the arch-matching `claude-ssh-linux-$architecture` binary rather than the installer's full `macOS`, `Windows`, and both-Linux-arch set, then re-asserts the exec bit with `chmod +x`. It is sourced by `build.sh` and invoked unconditionally for every build format (`deb`, `rpm`, `ApplImage`, `Nix`), with the Nix format alone branching its destination to `$app_staging_dir/nix-resources`. At runtime the staged directory resolves to `process.resourcesPath/claude-ssh/`, the exact path pre-rebase builds' SSH remote-connect flow read in order to deploy the helper onto a remote host.

ORIGIN

Origin: issue #235 ("Could not spawn `claude-ssh`", native binary missing from the Linux package), opened by @ayoahha 2026-02-17 with a decompiled trace of the failing `process.resourcesPath` lookup. A competing fix, PR #249 by @rfxlamia (2026-02-22), copied the entire `claude-ssh/` tree across every platform and arch plus a verification script; it was closed unmerged 2026-02-28 in favor of #268. PR #268 merged 2026-02-28 as `fd24511`, introducing `copy_ssh_helpers` with the narrower arch-matching-only copy, citing a stated 12–18MB per-package saving over #249's copy-everything approach.

REVISION HISTORY

WHEN	REFS	EVENT
2026-02	#235, #268, fd24511	Origin: issue #235 (@ayoahha) traced the missing binary; competing PR #249 (@rfxlamia) copied the full multi-platform tree and was closed unmerged in favor of #268, which created <code>copy_ssh_helpers()</code> with an arch-matching-only copy.
2026-02	#266, bd5a096, 6a3aae6	Nix-format support: destination first branched between the electron resources path and <code>nix-resources</code> , then collapsed back to one path by repointing <code>electron_resources_dest</code> itself for nix builds.
2026-04	#443, ff4821e	Pure move: extracted verbatim into <code>scripts/staging/ssh-helpers.sh</code> during the <code>build.sh</code> subsystem split; no logic change.
2026-07	rebase, d9cef9e	Deleted wholesale with the rest of <code>scripts/staging/*</code> in the Phase 1 acquisition swap to the official <code>.deb</code> .

Fate under the rebase. `DELETE` `scripts/staging` was deleted wholesale on the rebase branch, including `ssh-helpers.sh`; the official 1.17377.2 amd64 `.deb` contains no `claude-ssh` artifact anywhere in `data.tar.xz` (byte-verified 2026-07-03, one unrelated `Ssh` substring hit in an asset filename aside). Static analysis of that same package's `app.asar` finds a `ClaudeSSHManager` class that lazy-downloads `claude-ssh.zst` into a writable `userData` directory at connect time rather than reading `process.resourcesPath/claude-ssh/`, which no longer occurs anywhere in the bundle; whether official Linux actually completes a live SSH-remote connection through that download path, or lacks the capability some other way, has not been verified at runtime.

Affects. Per-arch (amd64/arm64 binary staging); DE-agnostic.

Icon Staging: Extraction and Opacity, Now Shipped Upstream

MECHANISM

`scripts/staging/icons.sh` (`process_icons`) is the build-time machinery that turns the Windows installer's `claude.exe` icon resource into Linux-consumable assets. `wrestool -x -t 14 lib/net45/claude.exe -o claude.ico` pulls icon resource type 14 (`RT_GROUP_ICON`) out of the Windows PE binary; `icotool -x claude.ico` expands the multi-resolution container into individual `claude_N_SIZExSIZEx32.png` files, hard-failing the build if either tool invocation fails. A size-to-index map (16 = 13 through 256 = 6) lets `deb` and `rpm` packaging install each size into the `hicolor` icon theme tree for `.desktop` integration; `Applmage` packaging uses only the 256×256 output, copied to the `AppDir`'s `hicolor` tree, a top-level PNG, and `.DirIcon`. Separately, the function copies the bundled `TrayIconTemplate(.png|-Dark.png)` files (and `@2x/@3x` variants) into `$electron_resources_dest`, then runs `ImageMagick` (`magick`, falling back to the deprecated `convert`) with `-channel A -fx 'a>0?1:0' +channel` against each one: any non-zero alpha pixel becomes fully opaque, since the originals are macOS ~20%-opacity templates Linux never colorizes. `deb`, `rpm`, and `Applmage` packaging consume the `hicolor` and `tray` outputs directly off the filesystem; Nix additionally required a dedicated `nix-resources` destination so `process_icons` had somewhere to write, plus a second, un-opacity-processed `asar`-internal `tray-icon` copy for the `isPackaged=false` code path.

ORIGIN

Pickaxe on `wrestool/icotool` bottoms out at the true origin, `d8f4bbe` (2024-12-26, the project's initial commit, as

build-deb.sh): the extraction, size-to-index map, and per-size hicolor install were present from day one, with no opacity conversion and tray icons copied straight inside the asar. #134 (47c1889, merged 2025-11-07, fixing #122) moved the tray-icon copy out of the asar and onto the filesystem ("Tray icons must be in filesystem (not inside asar) for Electron Tray API to access them"), and created the dedicated Icon Processing section icons.sh descends from. The opacity conversion itself was born in e5d58f2 (2026-01-19, #163), a commit shared with tray.sh: this unit's half added the ImageMagick alpha pass, while the tray section covers the selection-side fix (the light/dark inversion) landed in the same diff.

REVISION HISTORY

WHEN	REFS	EVENT
2024-12	d8f4bbe	Origin: wrestool/icotool extraction, size-to-index hicolor map, deb hicolor install; tray icons copied inside the asar, in build-deb.sh's initial commit.
2025-11	#122, #134, 47c1889	Tray-icon copy moved from inside the asar to the filesystem \$electron_resources_dest: Electron's Tray API needs a real filesystem path, not an asar-internal one; dedicated Icon Processing section created.
2026-01	#163, e5d58f2; e29635f	Opacity conversion born (shared commit, tray.sh half is the selection sed), then corrected the next day: the erroneous -negate removed and the loop widened over TrayIconTemplate*.png to catch @2x/@3x variants.
2026-01	3865848	Threshold fix: the alpha formula replaced with -fx "a>0?1:0" after prior attempts left 5-20-of-255-alpha edge pixels translucent.
2026-01	29173e9; 86a1822	Extracted into named process_icons() during the 26-function build.sh refactor; RPM packaging added as a second consumer of the same extraction output.
2026-02	9fe293d; 573f052	Nix build path wired with its own nix-resources destination; a second, un-opacity-processed tray-icon copy added inside the asar for Nix's isPackaged=false code path (@typedrat).
2026-03	0e4a1e7	First integration-test coverage: tests/test-artifact-{deb,rpm,appimage}.sh check hicolor presence and tray-icon counts (@sabiut).
2026-04	ff4821e	Moved verbatim into scripts/staging/icons.sh during the module split.

Fate under the rebase. DELETE scripts/staging is deleted wholesale; the official tree ships purpose-made TrayIconLinux(-Dark).png plus a complete hicolor set at 16-256px, which the new packaging copies verbatim.

Affects. All desktops (.desktop hicolor set plus tray icons); the opacity defect specifically hit KDE and Fedora panels (#163, shared with the tray unit).

The Sandbox Shims: What the SUID Bit Survives On

MECHANISM

The deb postinst (scripts/packaging/deb.sh) locates chrome-sandbox under the installed package tree and runs chown root:root plus chmod 4755 on it, warning rather than failing if the file is missing. The comment states the reason directly: the official data.tar.xz records the SUID bit, but the project's non-root ar | tar extraction (scripts/setup/official-deb.sh, which deliberately avoids dpkg-deb so RPM-family and Arch hosts can build) strips it, so the postinst restores what extraction lost. scripts/packaging/rpm.sh takes a different shape: rather than a %post scriptlet, chmod 4755 is baked into the buildroot before %install's %files walk, or-

dered to run after the blanket find ... -exec chmod u=rwX permission-normalization pass so the 4755 bit survives it. scripts/packaging/appimage.sh passes --no-sandbox unconditionally through build_electron_args(), because a FUSE-mounted, user-mounted Applmage cannot host a root-owned, 4755-mode helper the way an installed package can: there is no SUID path available inside the AppDir, so the flag is the only option rather than a workaround. On main, nix/claude-desktop.nix repoints the stock nixpkgs electron-wrapper's CHROME_DEVEL_SANDBOX at a co-located copy of chrome-sandbox (a path-correctness fix, not a permission-loss fix, since Nix builds never go through ar | tar); on the rebase branch that derivation is currently a stub.

```
chown root:root "$SANDBOX_PATH"
chmod 4755 "$SANDBOX_PATH"
```

ORIGIN

Pickaxe resolves each shim to a distinct commit, and none share an origin despite superficially similar mechanics. The deb reassert traces to d7d4695 (2025-03-30, direct commit, no PR), reverting a same-day, ten-minutes-earlier ed2ac2d that had hardcoded --no-sandbox into the deb launcher to fix a Google-login crash, citing electron/electron#17972. The Applmage flag traces to 1f13d7a (2025-04-02, direct commit): unlike the deb flag, it was never superseded, since a portable Applmage has no postinst-style mechanism to fall back to. The Wayland-conditional --no-sandbox for deb traces to 3e43708 (#97, @delorenj), bundled with GPU flags to fix a Trace/breakpoint trap crash attributed to dmabuf/EGLImage binding errors on GNOME and KDE Plasma Wayland. The RPM shim traces to 86a1822 (2026-01-24): multi-distro support ported the deb's already-abandoned first-draft %post chown/chmod pattern into rpm.sh, not its later, current shape. That RPM shim then ran its own two-round saga: #539 (@qtlin) reported that a %post chmod silently no-ops under --noscripts or layered rpm-ostree images, shipping a non-SUID chrome-sandbox; PR #595 (@JoshuaVlantis) moved the bit into %attr(4755, root, root) in %files, which introduced a new rpmbuild "File listed twice" warning caught as #609 (aaddrick) and fixed by PR #610 (@JoshuaVlantis), which baked the chmod into the %install buildroot instead. Separately, the AppArmor userns arc opened with #351 (@hfyeh, Ubuntu 24.04 blocking Cowork's bwrap), got a manual, system-wide doc workaround in PR #434 (@RayCharlizard) that #542 later flagged as over-broad, then PR #687 (@diarized) automated a scoped profile for the Electron binary itself against a deb-on-X11-only credentials.cc crash, explicit that the sandbox "stays enabled, never --no-sandbox on the deb," and PR #694 (same author, same day) added the structurally identical scoped profile for /usr/bin/bwrap, superseding #434's manual fix.

REVISION HISTORY

WHEN	REFS	EVENT
2025-03	cc57c39; ed2ac2d; d7d4695	README documents --no-sandbox as a manual step; deb launcher hardcodes it, citing electron/electron#17972; reverted the same day by the postinst chown root:root/chmod 4755 reassert (origin of the deb shim).
2025-04	1f13d7a; 83d302a	Origin of the Applmage shim: --no-sandbox added permanently for FUSE; deb packaging logic extracted into its own script.
2025-06	#97, 3e43708; bf6c5bf	Origin of the Wayland-deb shim: --no-sandbox bundled with GPU flags against a Trace/breakpoint trap crash on GNOME and KDE Plasma Wayland (@delorenj); merged 2025-08-04.
2026-01	86a1822; ee5e090, 424e17b	Origin of the RPM shim, porting the deb's abandoned first-draft %post pattern; per-branch --no-sandbox calls consolidated into build_electron_args().
2026-03	issue #282, 99d91ac; issue #351	Wayland branch extended from deb to deb nix; @hfyeh reports Ubuntu 24.04's userns restriction blocking Cowork's bwrap.

REVISION HISTORY CONTINUED

WHEN	REFS	EVENT
2026-04	PR #368 (fixes #316), a326ea2; PR #434; issue #539	Origin of the Nix shim, a side effect of the isPackaged/resourcesPath fix; @RayCharlizard ships a manual, system-wide bwrap AppArmor profile; @qtlin reports the RPM %post chmod is scriptlet-fragile.
2026-05	#539 → PR #595, 15813ca, cf08571, c9df9e2	RPM SUID moved from %post chmod to %attr(4755, ...) in %files (@JoshuaVlantis); merged 2026-05-14.
2026-05	issue #609 → PR #610, ba8ffa1, a04ed9e	RPM SUID moved again, from the explicit %attr line to a buildroot %install chmod, silencing "File listed twice" (@JoshuaVlantis); merged 2026-05-24.
2026-06	PR #695, 5c43ccd	Blanket install-tree permission normalization for the Cowork daemon launch fix (@caidejager), sequenced to run before the chrome-sandbox-specific chmod so 4755 survives.
2026-06	PR #687, 12cc726; PR #694, 50dd1f0	Scoped AppArmor userns profile automated for the Electron binary (deb-on-X11 crash fix); a structurally identical profile for /usr/bin/bwrap the same day, superseding #434's manual fix (@diarized).
2026-07	d9cef9e	Phases 1+2: paths updated to the official package root (no more node_modules/electron/dist); AppArmor profile's attached binary path updated; Nix derivation replaced with a stub.
2026-07	cafb4cc	Phases 4+5: doctor's chrome-sandbox and User-namespaces checks moved to the official layout; tools/chromium-switch-smoke.sh added to lock the --no-sandbox scenario matrix.
open	issue #617	@sabiut's --noscripts RPM regression-test gap remains unresolved.

Fate under the rebase. **SURVIVOR** The shims survive because the official data.tar.xz records chrome-sandbox SUID (-rwsr-xr-x root/root), but the project's non-root ar | tar extraction strips that bit, so the postinst chown root:root/chmod 4755 reassert ports over unchanged in logic as the mechanism that restores what extraction lost, its path merely updated from node_modules/electron/dist/chrome-sandbox to the bare package-root chrome-sandbox in the official layout. The scoped Electron-binary AppArmor userns profile survives alongside it, re-attached to the bare official ELF, sitting next to the official postinst's own AppArmor and apt self-registration behavior in that same install step. rpm.sh's buildroot chmod and appimage.sh's FUSE --no-sandbox survive unchanged in effect; doctor.sh's chrome-sandbox and User-namespaces checks were reworked to the new paths, and a new tools/chromium-switch-smoke.sh locks the --no-sandbox scenario matrix in CI. What does not survive: the structurally identical scoped AppArmor profile for /usr/bin/bwrap (#694), parked because the official Cowork backend is coworkd plus QEMU/KVM rather than bwrap; nix/claude-desktop.nix is presently a stub, so the Nix CHROME_DEVEL_SANDBOX repoint's fate is open rather than settled.

Affects. RPM SUID payload and Ubuntu 24.04+ unprivileged-usersns restriction (AppArmor); AppImage forces --no-sandbox under FUSE.

Eighteen Units, Five Outcomes

The matrix reduces the adaptation layer to one row per unit: when it was born, the issues and pull requests that defined it, its verdict under the patch-necessity audit of official 1.17377.2 (re-verified against pristine 1.18286.0 under the accessibility reassessment of section 21), the environments it is sensitive to (reverse-looked-up from the repo-wide catalog), and what handles the problem now. **DELETE** means the official build makes the patch redundant; **SURVIVOR** means the patch stays wired in `active_patches` because upstream still has the gap; **VERIFY** means the verdict hinges on a live reproduction the `.deb` bytes cannot settle, so the unit may be deleted, kept unwired, or kept wired, but a named open check decides; **PARKED** means moved out of the build into a fallback tree for a later arc; **REWORKED** means rebuilt rather than removed. Three rows carry a **VERIFY** badge from that reassessment, `frame-fix` and `wco-shim` as delete-with-residual-risk and `quick-window` as a wired keep-pending-repro; section 21 records the reasoning.

The legacy adaptation layer at retirement, eighteen units. Born = first commit of the unit's logic. Refs = defining issues/PRs, not exhaustive. Env = catalog-derived sensitivity.

UNIT	BORN	REFS	VERDICT	ENV	DISPOSITION UNDER V3.0.0
Chassis	2024-12	#179, #218, #219, #220	REWORKED	All Linux; packaging layer	Thin orchestrator; empty <code>active_patches</code> → byte-identical repack; derived - -unpack glob; WM_CLASS fail-fast survives upgraded.
frame-fix-wrapper	2025-11	#127, #228, #232, #451, #567	VERIFY	GNOME, KDE, wlroots; X11 + Wayland	Frame core and updater no-op are byte-moot (deleted); the audit returns check on 1.18286.0 (frame: !13x), so the ~18 accreted Electron-runtime fixes (#416/#605/#128/#623) move to verify pending FF-1.
claude-native stub	2024-12	#729	DELETE	All Linux; DE-agnostic	Real Rust NAPI ELF ships in <code>app.asar.unpacked</code> (X11 input, <code>openat2</code> , <code>SO_PEERCRED</code>).
node-pty	2026-01	#152, #421, #401	DELETE	All Linux; build-time	Prebuilt <code>linux-x64/pty</code> . <code>node</code> ships in <code>app.asar.unpacked</code> .
tray.sh	2025-11	#135, #163, #515, #679	DELETE	KDE, GNOME (SNI trays)	Official rebuild is in-place <code>setImage</code> keyed on <code>icon-path</code> change, plus purpose-made Linux icons; the community diagnosis, converged upstream.
menuBar default	2026-02	#218, #644, #750	DELETE	Any tray desktop	Official defaults map ships <code>menuBarEnabled: !0</code> .
quick-window	2025-12	#144, #147, #390, #406	VERIFY	KDE Plasma only	Anchor present in official 1.18286.0 bytes (var ms) with no <code>blur()</code> ; stays wired in <code>active_patches</code> pending KDE Plasma repro QW-1.
claude-code	2025-11	#143, #241, #243	DELETE	All Linux; DE-agnostic	Official <code>getHostPlatform</code> has native <code>linux-x64/linux-arm64</code> branches.
asar guards (cowork)	2026-05	#383, #622, #632, #668	DELETE	All Linux; launcher argv	Official launcher is a bare ELF symlink; no <code>app.asar</code> argv ever reaches the helpers.
cowork reroute + daemon	2026-02	#198, #259, #288, #337	PARKED	KVM hosts; OVMF-sensitive	Moved to <code>scripts/cowork-fallback/</code> ; 3.0.0 ships KVM-only with doctor guidance; <code>bwrap-vs-coworkd</code> protocol is the 3.1 investigation.
org-plugins	2026-05	#607, #639	SURVIVOR	All Linux; MDM path	Official switch still returns null on Linux (confirmed pristine 1.18286.0, no linux case); <code>/etc/claude/org-plugins</code> preserved; keep-cost near zero, upstream report still planned.
wco-shim	2026-05	#538	VERIFY	All Linux equally	Local WCO half dead (audit not - needed, <code>mainView</code> refs 0); the remote <code>isWindows()</code> UA gate is server-side, so the topbar half moves to verify pending WCO-1.
config writes (#400)	2026-05	#400, #649, #685, #723	VERIFY	All Linux; DE-agnostic	Both <code>.asar</code> guards die with the <code>asar-argv</code> root cause; the <code>mcpServers</code> merge is kept unwired pending a live reproduction.

FATE MATRIX CONTINUED

UNIT	BORN	REFS	VERDICT	ENV	DISPOSITION UNDER V3.0.0
acquisition	2024-12	#143, #151, #310	REWORKED	Per-arch, not per-distro	Official .deb from the apt/stable pool, SHA-256 pinned, ar/tar extraction; Playwright resolver retired; Nix side a throw stub.
launcher + doctor	2025-06	#102, #141, #690	REWORKED	All desktops; Wayland/XRDP edges	Opt-in-only policy; launchers exec the official binary; titlebar/password-store machinery deleted; six new doctor checks (_check_kvm ...).
ssh-helpers	2026-02	#235, #268	DELETE	Per-arch; DE-agnostic	No claude-ssh artifact anywhere in the official .deb; runtime capability path unverified.
icon staging	2024-12	#134, #163	DELETE	All desktops; KDE/Fedora panels	Official ships TrayIconLinux(-Dark).png plus a full hicolor set, copied verbatim.
sandbox shims	2025-03	#539, #595, #687, #694	SURVIVOR	RPM SUID; Ubuntu 24.04+ usersns	Non-root ar/tar strips the recorded SUID bit; the postinst re-assert restores it. Applmage forces --no-sandbox.

A Record of Correct Diagnoses

Read end to end, the suite’s history is dominated by one pattern: **the community found the bug first, and upstream eventually agreed**. The tray unit spent eight months converging on an in-place setImage design that the official build then shipped natively; the frame fix forced the system titlebar that the official window simply has; the menu-bar default, the Linux getHostPlatform branches, the pty binary, and the disabled updater all reappear in the official package as first-party decisions. Deleting the redundant units under the rebase discards code, not knowledge: the diagnoses were validated by the strongest reviewer available, the vendor’s own engineers arriving at the same answers.

What survives is exactly the set of problems upstream has not solved. quick-window.sh stays because the Electron-on-KDE stale-focus anchor is still present in official bytes; org-plugins.sh stays because the official platform switch still returns null on Linux, leaving MDM-managed plugin marketplaces dead upstream; the mcpServers merge waits unwired on a behavioral reproduction of #400; and the Cowork daemon is parked, not deleted, because a bwrap fallback for non-KVM hosts remains genuinely useful and now means speaking coworkd’s SO_PEERCREd protocol, a 3.1-sized investigation. The survivor suite also keeps the arms-race methodology alive: docs/learnings/patching-minified-js.md governs two patches now instead of thirteen, against an upstream that shipped 1.18286.0 one day after 1.17377.2.

The history also leaves the project a to-do list it can now file where it belongs. The suite’s remaining gaps convert into **upstream reports against a first-party Linux target**: the missing org-plugins Linux case, the possible loss of the in-app topbar behind the remote bundle’s Windows gate, the StartupWMClass mismatch, the hardcoded OVMF probe list, and the stdio MCP double-spawn already validated in CDL-ANT-0008. Open items on this report’s own ledger: the KDE Plasma reproduction that decides quick-window.sh, the #400 reproduction that decides the config merge, and a runtime answer to where, if anywhere, the official Linux build carries the claude-ssh capability. And the rebase clarifies the repository’s purpose rather than ending it: Anthropic ships Linux as a single Debian-family .deb, so the mission now is the long tail, repackaging the official build as RPM, Applmage, and Nix derivation for the distributions the first-party channel does not reach. The patch suite ends as it lived: not as a fork of Claude Desktop, but as the best available record of what it actually took to run one on Linux.

An Accessibility Reassessment, Re-Verified Against 1.18286.0

This section is an **addendum**, not a nineteenth unit: the eighteen-unit spine and every count in the overview and the fate matrix stand as written. After the retirement audit was fixed against the pinned official 1.17377.2 bytes, a second pass (**accessibility reassessment, July 3, 2026**) re-scored every verdict under a single lens, **make the Linux build work out-of-the-box on the widest range of the environments users actually report**, and ground-truthed the result against a freshly fetched pristine official **1.18286.0** .deb (sha256 8f314ad1...0536, three releases newer than the pin). The shipped code did not change; what changed is how much residual risk three of its deletions and survivor calls are now understood to carry.

THREE FLIPS, ONE FIRM

The lens moved exactly three verdicts, all in the same direction, from a confident **DELETE**/**SURVIVOR** to **VERIFY**, because in each case the load-bearing evidence lives where .deb bytes cannot reach. `frame-fix` splits: the `frame-core`, `titlebar-mode`, and `autoUpdater` no-op slices are byte-moot and stay deleted, but the roughly eighteen accreted Electron-runtime fixes track *unfixed upstream* bugs (#416 hover-raise, #605 sleep inhibitor, #128 `openAtLogin`, #623 quit hatch), and `tools/patch-necessity-audit.sh` returns check on 1.18286.0 (“`frame:!1 occurs 3x`”), not a clean delete. `wco-shim` splits the same way: the local `mainView.js` WCO half is dead (audit not-needed, “`mainView refs: 0`”), but the load-bearing page-side `isWindows()` UA spoof depends on `claude.ai`’s server-delivered bundle. `quick-window` moves from survivor candidate to **verify**: the anchor is intact on 1.18286.0 (`var ms, ||hide()` present, no `blur()`), but the defect lives in Electron’s Linux `isFocused()`, so unchanged bytes cannot distinguish “still broken” from “Electron fixed it.” One verdict firms without flipping: `org-plugins` goes from survivor candidate to a settled **SURVIVOR**, its byte-gap confirmed on pristine 1.18286.0 (`default:return null`, no linux case). A **VERIFY** badge is an annotation of residual risk, **not** a code reversal: `frame-fix` and `wco-shim` are deleted and shipped, and only `quick-window` and `org-plugins` sit in `active_patches`.

GROUND-TRUTH CORRECTIONS

The verification pass ran against the reassessment as much as against the audit, and corrected one of its claims. The reassessment framed the launcher’s dropped password-store *auto-detection* as a deleted regression and a hard blocker; that framing is wrong and is deliberately kept out of this report. `--password-store` still passes when `CLAUDE_PASSWORD_STORE` is set (a documented escape hatch citing #593), the doctor still reports the backend, and what actually changed is that the official `os_crypt` autodetect now owns the default, a deliberate documented rework. The one genuinely open part is **LD-1**: whether that autodetect persists cookies on a live KDE/kwallet6 session, which static analysis cannot settle. The pass also surfaced two byte facts the audit had not called out. The official tree is bare co-located with **no node_modules/electron/dist directory**, so the three artifact-test scripts that still asserted that on-disk path (**SB-1**) failed against the rebase packages — now repointed to the bare layout (the companion .bats references were only synthetic strings fed to path-agnostic matchers and never failed); see the settlement below. And the payload compression changed across the train, `data.tar.zst` on 1.17377.2 to `data.tar.xz` on 1.18286.0, both already handled by `_extract_deb_member`.

LIVE-HARDWARE SETTLEMENT (REV B, JULY 3--4, 2026)

The open checks were then run on the local libvirt test fleet and a real KDE Plasma/kwallet6 host against the rebased 1.18286.0 build, and the two deletions carrying the most residual risk are now confirmed. **FF-1 is re-**

solved: on a focus-follows-mouse compositor (niri) the deleted `webContents.focus()` sloppy-WM guard produces no anomalous focus-steal (#416); the #605 sleep inhibitor both registers *and* releases at the D-Bus layer on KDE (a session-bus Inhibit/UnInhibit capture paired every cookie), so the never-released-inhibitor bug does not reproduce — though keep-awake is a silent no-op on bare wlroots/i3, which expose no SessionManager or PowerManagement inhibit service (a separate functional gap, upstream candidate); openAtLogin (#128) survives a reboot; and quit-without-tray (#321/#623) is handled natively by the official build's **Settings** ▢ **General** ▢ **System Tray** toggle (off = quit-on-close), so the deleted `CLAUDE_QUIT_ON_CLOSE` hatch is superseded, not lost. **WCO-1 is resolved:** the in-app claude.ai topbar renders on both KDE and niri, so the server-delivered `isWindows()` UA gate does not hide it on Linux and `wco-shim` needs no UA-override survivor (sway draws a second, server-side titlebar over it — a decoration-suppression bug, not a topbar-absence one). **The #400 config-merge check is closed as a deliberate no-op:** the loss reproduces only when the config file is edited *while the app runs*; the normal edit-then-restart flow is safe, and extraction of the 1.18286.0 bundle showed `setMcpServers` now deletes entries programmatically (`delete i[a]`), so the retired `Object.assign` merge would resurrect a legitimately-deleted server — the patch correctly stays out of `active_patches`. **SB-1 is fixed** (repointed as above). The verification also surfaced findings outside the reassessment's frame, the load-bearing one being **LOG-1:** the launcher logged the full `argv`, so a relaunch through the login redirect wrote the `claude://login/...?code= OAuth` authorization code to `launcher.log` in plaintext — now redacted at the `log_message` chokepoint.

WHAT STILL NEEDS HARDWARE

Two checks narrow rather than close. **QW-1** confirmed the happy-path Quick Entry submit-then-raise flow on both the patched (KDE) and unpatched (niri) paths, but the KDE stale-focus raise edge — does the popup reliably reappear when the main window is hidden or visible-but-unfocused — still decides whether quick-window stays. **LD-1** passes on a KDE session with a pre-existing wallet (the official `os_crypt` autodetect seals cookies, auto-probe dropped) but leaves the fresh-no-wallet freeze edge untested; keyring-less compositors persist via the basic backend, storing the token unencrypted at rest behind an advisory prompt. Two subsystem checks stay genuinely open and off the 3.0.0 critical path: live arm64 rootfs availability, and a Cowork socket-protocol capture on a KVM host (feeding the 3.1 `cowork-bwrapd` scoping). Separately, the no-hardware follow-ons ride their own pull requests: **ACQ-1** (the Nix derivation is still a hard throw, and Nix is the single largest install channel in the catalog), **LD-2** (note `CLAUDE_QUIT_ON_CLOSE` as a no-op in the doctor's legacy-env list, now that the tray toggle supersedes it), and **AU-1/MB-1** (build tripwires on `apt_channel_pending` and `menuBarEnabled:!0` so a future upstream flip is caught at build time). The settled verdicts are the **patch-necessity matrix**; the reassessment's full reasoning is the **accessibility reassessment**.