

Bugpacker: security and data handling

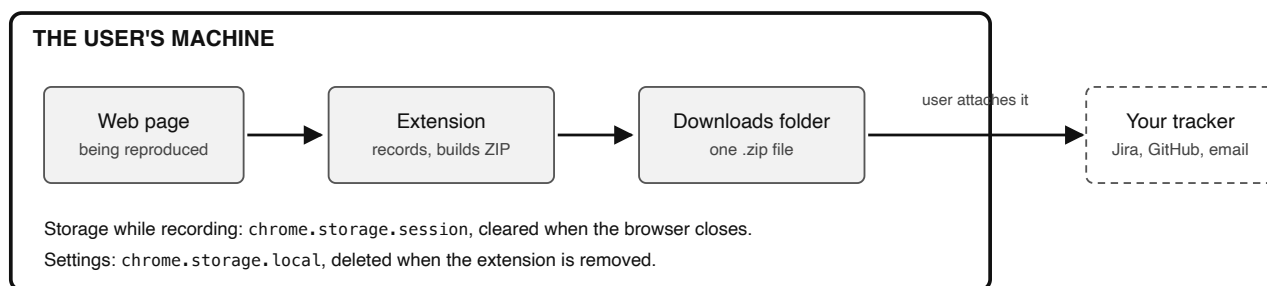
Prepared 12 September 2026 for vendor and security review, and reviewed quarterly. Describes the version then published on the Chrome Web Store. The authoritative and current copy is at bugpacker.com/security.

Bugpacker is a Chrome extension that records a bug while somebody reproduces it, then builds a ZIP containing screenshots, console output, network activity, a page snapshot, form state and reproduction steps. The user saves that file and attaches it to their own issue tracker. There is no Bugpacker service, because we do not operate a server.

THE SHORT ANSWERS

Where does captured data go?	Nowhere. It is written to the user's own downloads folder.
What does the extension send?	Nothing. It makes no network requests of any kind.
Which hosts can it reach?	None. It declares no host_permissions.
Is there an account or login?	No.
Analytics, telemetry, crash reporting?	None of any kind, including anonymized forms.
Is there a data processing agreement to sign?	No. We process nothing.
Sub-processors?	None.
What happens to our data in three years?	It was always only ever on your own disk.
What happens if the vendor disappears?	The installed extension keeps working. It needs nothing from us.
Cost, and what happens if we stop paying?	Free. There is nothing to stop paying.
Vulnerability disclosure contact?	hello@bugpacker.com. No public tracker; the repository is private.
Attestation reports or certifications held?	None. See the last section, which says so plainly.

DATA FLOW



Nothing crosses this boundary. There is no outbound request at any point above.

The dashed box is the only thing that leaves the machine, and a person moves it deliberately by attaching a file they have already reviewed.

HOW TO CHECK THAT THERE ARE NO OUTBOUND REQUESTS

The extension declares **no host_permissions at all**. Chrome will not permit a request to a host an extension has not declared, so the absence of that key is enforced by the browser itself. Open the extension's entry in `chrome://extensions`, or unzip the package and read `manifest.json`: the key is absent entirely.

PERMISSIONS, AND WHY EACH IS HELD

PERMISSION	WHY IT IS HELD
activeTab	Access to the one tab the user invoked Bugpacker on, granted by their click. This is what lets it work without requesting access to every site at install.
scripting	Draws the region selector, records console and network activity from inside the page, reads the snapshot and form state when recording stops. Injected only into the tab being recorded.
storage	The two local stores shown in the diagram.
downloads	Writes the finished ZIP. This is the extension's only means of delivering output.
sidePanel	The capture interface.
alarms	Stops a forgotten recording after a configurable limit. Service workers are terminated between events and cannot hold timers.
webNavigation	Re-attaches the recorder when the page navigates mid-recording, so a reproduction that moves between pages on the same origin is captured completely.
webRequest	Observes whether requests succeeded or failed. Observation only: nothing is blocked, redirected or modified.

Two things about the install prompt, because both get raised in review.

Chrome renders `webNavigation` as **"Read your browsing history"**. That phrase is Chrome's name for the permission. Bugpacker uses it to notice that the recorded tab navigated, so the recorder can re-attach. No history is read, stored or transmitted.

The prompt does **not** include "Read and change all your data on all websites", because Bugpacker declares no host permissions at all, required or optional. Capture runs on Chrome's temporary activeTab grant, which covers the tab you invoked Bugpacker on, including same-origin navigation and reload, and ends when the tab navigates to a different origin. A recording that crosses origins stops capturing at the crossing, and the package records that it left the site.

WHAT IS NEVER CAPTURED, AS DISTINCT FROM WHAT IS REDACTED

This is the distinction that matters in a data review. Some things are redacted after capture. Some are never captured at all, so there is nothing left to fail at redacting.

Never captured

- **Field values are never recorded.** The page snapshot drops every value attribute, and the form state record keeps only the label, the validity and why the field was rejected. A password, a card number or a search term typed into a form does not enter any file, so there is nothing to redact.
- **Inline script bodies are removed** from the page snapshot, since they routinely carry configuration and tokens.
- **Cookie and storage values are not read.** Only their names appear, as environment context.

Redacted, always on, best effort

Email addresses, phone numbers, card numbers, authorization headers and tokens found in logs and network data are replaced with placeholders that keep the format and drop the content. Payment and password fields are masked in screenshots. None of this can be switched off, and there is no per-site exemption.

It is best effort, and it is described that way here and in the privacy policy. Bugpacker is a bug-reporting tool. It is not a data-loss-prevention, anonymization or compliance product. The user reviews the entire package before it is saved.

CODE AND SUPPLY CHAIN

- **No remote code.** Everything the extension runs ships inside the package. Nothing is fetched or evaluated at runtime, and `web_accessible_resources` is empty.
- **Manifest V3**, which forbids remote code by design.
- **Three direct dependencies:** React, React DOM, and fflate for building the ZIP. React DOM brings scheduler with it, so four packages of third-party code execute in the extension. No analytics SDK, no error reporting SDK, no tag manager. `THIRD-PARTY-NOTICES.txt` at the package root is generated from the build output and has eight entries: these four, whose code executes, plus vite, rollup, esbuild and tailwindcss, which leave interop code and generated CSS in the package rather than libraries. Each entry says which kind it is.
- **Source maps ship with the package**, deliberately, so the original TypeScript is readable by anyone who unpacks it.
- Distributed only through the Chrome Web Store, which reviewed and signed it.

If Chrome warned that the extension is "not trusted by Enhanced Safe Browsing"

That warning reflects the age of the publisher account. It is not an assessment of the code. Google's own position is that it takes a new developer at least a few months to become trusted, that it cannot be fixed by the developer, and that it resolves on its own for developers who follow their policies. Bugpacker was first published in September 2026. Users on Chrome's standard protection see no warning.

HOW TO VERIFY ALL OF THIS YOURSELF

None of the above requires taking our word for it. A Chrome extension is a ZIP file of readable code, and this one ships its source maps.

1. Install it, then find it under `Extensions/` in your Chrome profile directory. Or download the CRX with any of the usual tools.
2. Read `manifest.json`. It lists every permission and every host the extension may contact. Compare it against the permissions table above.
3. Search the bundled JavaScript for `http`. Every match will be a link a user can click. There is no outbound request URL in the bundle at all.
4. Watch it work. Open the network inspector on the recorded page and on the extension's own pages, and record a bug end to end. It produces no outbound request, because it holds no host permission that would let it make one.

WHAT WE CANNOT CLAIM

Bugpacker holds no attestation reports and no certifications. There is no SOC 2 report, no ISO 27001, no FedRAMP authorization and no penetration test to share. It is a small independent product.

What it offers instead is a smaller question to answer. There is no vendor processing your data, so there is no data processing agreement to negotiate, no sub-processor list to review, no breach notification path to agree and no vendor to reassess next year. The captured bug data never leaves the machine it was captured on. For many organizations that turns a vendor review into a software review.

Questions, including anything this document does not cover: hello@bugpacker.com. If your review needs something specific in writing, ask.

Authoritative and current version: bugpacker.com/security. If this copy is more than a quarter old, check there before relying on it.