

Vendor security questionnaire

The questions a supplier review asks, answered — including the ones where the answer is unfavourable. Signing, penetration testing, certifications and logging are all in here.

Answers as at Oxofolio 0.2.1. Where the answer is unfavourable it is given plainly; a security pack that only contains good news is not usable by the person who has to sign it off.

Is customer data transmitted to the supplier or any third party?

No. The application makes no network connections. The supplier receives nothing, including no telemetry or usage data.

Where is customer data stored?

On the customer's own machine, in the customer's own folders. The supplier operates no infrastructure that holds customer data.

Is data encrypted in transit?

Not applicable: there is no transit. Nothing leaves the machine.

Is data encrypted at rest?

Documents are left as the customer stores them, so at-rest encryption is whatever the customer's disk encryption provides — full-disk encryption is recommended and the application reports whether it is enabled. Document passwords the operator asks the application to remember are encrypted with the operating system keystore (Keychain on macOS, DPAPI on Windows) and are not written at all if the OS cannot encrypt them.

What authentication does the application use?

None. It has no accounts. It runs as the logged-in operating system user and inherits that user's file permissions.

Are there sub-processors?

No. There is no processing off the machine, so there is nothing to sub-contract.

What third-party components are used?

Listed in full in the accompanying inventory, with the licence each declares.

How are vulnerabilities in dependencies handled?

Dependencies are pinned. Updates are applied on a release cycle and after any advisory affecting a component in the shipped inventory. The inventory in this pack is generated from the committed lockfile, with the integrity hash of every resolved package, so it can be diffed against an advisory feed and describes the same components whichever platform built the release.

Has the application been penetration tested?

Not independently, as at this release. Stated plainly rather than omitted.

Do you hold SOC 2 or ISO 27001?

No. Neither certification is held.

Is the application code signed?

Per platform, because the answer differs. **macOS: yes.** Signed with a Developer ID Application certificate, with the hardened runtime enabled, notarised by Apple and stapled — `spctl` reports it as a Notarized Developer ID build, and the release check fails if the disk image is not stapled. **Windows: yes.** The installer and everything inside it are signed as CHADURA TECH PRIVATE LIMITED and timestamped, so the signatures outlive the certificate rather than expiring with it. That includes the nested binaries — the sidecar, the bundled OCR engine and the DLLs beside them — not only the outer installer, because an unsigned binary inside a signed installer is what an endpoint agent quarantines. `check:signing` verifies every one against Windows itself and fails the build under `OXOFOLIO_RELEASE=1` if any is unsigned or untimestamped. SmartScreen builds reputation per certificate, so downloads may still warn until that reputation accrues. **Linux: the .rpm is GPG-signed** and verified against the published key; a customer should check the signature before installing.

What logging does the application produce?

A processing log written into the output directory, recording what was processed, when, by whom, on which machine, and the SHA-256 of every input and output. It is written for the customer, into the customer's folder. Document passwords are deliberately excluded from it.

How is an incident reported?

To the supplier's security contact. Because the application transmits nothing, a defect in it cannot expose customer data to the supplier or to any third party; the realistic incident classes are local — a document written to the wrong place, or an output that fails to redact what it claimed to.

Can the application be run without administrative rights?

Yes. It needs no elevated privileges and installs per-user.

Does the application execute anything it downloads?

It downloads nothing. The Python sidecar and the OCR engine are bundled in the installer and are the only executables it launches.

What infrastructure does the supplier operate?

One thing: a private S3 bucket of installers behind a CloudFront distribution, so the software can be downloaded. The bucket is not public — Block Public Access is on and only the distribution can read it, through an Origin Access Control. It holds the supplier's own installers and no customer data of any kind.

Does the supplier log who downloads the software?

Partly, and the distinction matters. **The CDN does not:** CloudFront access logging is switched off deliberately, configured in `scripts/setup-downloads.sh` with the reasoning recorded beside the setting. **The website does:** requesting a build through `oxofolio.com` records the platform, the release version, the time, the IP address and the browser user-agent, so the supplier knows which releases are in use. That makes the supplier a controller for those records, which the published privacy policy states. They are website access records and contain no customer document data of any kind, and they are not linked to any licence or

purchase.

How can a downloaded installer be verified?

Every release is published with a manifest listing the SHA-256 of each file, and the download page prints those hashes with the command to check them. A hash that does not match means the file is not the one that was published.

What happens to data when the customer stops using the product?

Nothing needs to happen at the supplier's end, because the supplier holds nothing. The application includes a retention clock and a certified erasure function for the customer's own copies.

1. Enforced rather than asserted

Some of the answers above are checked by scripts that run on every release build, not merely believed:

- `npm run lint:noNetwork && npm run security:check && npm run test:js && npm run preflight`
- The no-network scan rejects 7 networking patterns across 3 source trees; the patterns themselves are listed in the security overview.
- The preflight gate exercises 62 of the 62 sidecar methods against the frozen binary before a build is allowed, which is all of them.
- This document is regenerated from the source and the build fails if it is stale.
- The release pipeline refuses to publish an unsigned build: publishing runs the signing check first and stops on failure.