Scan2Tray

Vulnerability & Threat Management Procedure

Scan2Tray Ltd (Company No. 17096824) · Version 1.0 · 24 September 2026 · Owner: Paul Hinett, Director · hello@scan2tray.com

Supplement to the Scan2Tray Information Security Policy.

1. Purpose

This procedure describes how Scan2Tray finds, assesses and fixes security vulnerabilities in its software, dependencies and infrastructure, and how it stays aware of threats relevant to a small live-commerce fulfilment service.

2. Sources of vulnerability information

Source	What it covers
GitHub Dependabot alerts and automated security updates	Known vulnerabilities (CVE / GHSA) in every dependency manifest: .NET (NuGet) for the API, npm for the web app, marketing site and desktop shell, Cargo for the desktop's Rust core, and GitHub Actions. Alerts go to the security owner; weekly update pull requests are raised automatically.
Operating-system security updates	Production hosts apply OS security patches automatically (unattended upgrades). Staff endpoints apply OS and browser updates automatically.
Platform and vendor advisories	Release notes and security advisories from .NET, Tauri, Node.js, PostgreSQL, the deployment platform, the hosting provider and marketplace partner developer programmes (including TikTok Shop Partner Center notices) are reviewed as they are published.
External reports	Anyone can report a vulnerability to hello@scan2tray.com (published on scan2tray.com/security). Good-faith reports are acknowledged within 2 working days and handled under this procedure.
Internal findings	Issues found during code review, testing, log review or incident response are logged and triaged the same way.

3. Assessment and remediation targets

Each finding is assigned a severity based on CVSS where available, adjusted for whether the vulnerable code path is actually reachable in Scan2Tray and what data it could expose.

Severity	Typical examples	Fix or mitigate within
Critical	Remote code execution, authentication bypass, exposure of marketplace tokens or personal data	48 hours (mitigate immediately, e.g. disable the feature or block at the proxy)
High	Privilege escalation between organisations, injection, exploitable dependency in a reachable path	7 days
Medium	Vulnerability in an unreachable path, denial-of-service, information disclosure without personal data	30 days
Low	Hardening opportunities, best-practice deviations	Next scheduled release

If a fix cannot be applied within the target, a compensating control is put in place and the exception is recorded with an owner and a review date.

4. Verification and release

- Fixes are made on a branch, reviewed, and must pass the automated test suite before merging to the main branch.
- Production deploys automatically from the main branch, so a fix reaches production within minutes of merge; desktop fixes ship as a signed auto-update.
- After deployment the fix is confirmed against the original finding (re-test, log review or re-scan) and the record is closed.

5. Ongoing threat awareness and controls

- **Perimeter and host:** cloud firewall allowing only required ports, host firewall with default-deny inbound, key-only SSH, unattended OS security updates.
- **Application:** per-IP rate limiting on authentication and device pairing, HMAC verification and de-duplication of inbound marketplace webhooks, secrets held only in runtime configuration, organisation-scoped data access.
- **Monitoring:** proxy access logs and API security logs are reviewed when alerts fire; database health alerts and abuse alerts go to the security owner; per-IP blocking rules are applied without a deployment.
- **Threat review:** the security owner reviews the threat picture (new attack patterns against live-commerce sellers, marketplace API changes, supplier incidents) at least quarterly and after any incident, and adjusts controls or priorities accordingly.
- **Supplier risk:** hosting, payment, email and marketplace providers are chosen for recognised security certifications; their advisories and status pages are monitored.

6. Records and review

Findings, severities, decisions, exceptions and fix dates are recorded in the repository's issue tracker and Dependabot history. This procedure is reviewed with the Information Security Policy at least annually.