

Security Concept LocalTranscript

LocalTranscript · Documentation for IT and data protection

Status: version 0.6.65, September 2026

Purpose and Scope

This security concept describes the technical and organisational measures of LocalTranscript, a locally installed desktop application for recording, transcription, speaker recognition, and AI-assisted summarization of meetings. It is addressed to the IT department and the data protection officers of your organisation as part of a security or procurement review. Its scope covers the Windows and macOS editions of the application from version 0.6.65 onward.

Architecture in One Paragraph

LocalTranscript processes audio entirely on the workstation: speech recognition (whisper.cpp), speaker recognition (pyannote, natively via ONNX Runtime), and AI actions (Gemma via a local server listening only on 127.0.0.1) run without any cloud connection. The UI (React, in an embedded WebView) communicates with the application core (Rust) exclusively via an internal interface, not over the network. Meetings, transcripts, and notes reside in a local SQLite database and as files in the user profile. Every network connection originating from the application core – cloud provider, model download, calendar fetch, feedback, update check – is required to pass through a single checkpoint, which logs every attempt and, in Hard Offline Mode, refuses it before any name resolution.

Protection Goals and Measures

Protection Goal	Measure	Status
Confidentiality	Local processing with no cloud requirement; no user account; no telemetry	Implemented
Confidentiality	Network chokepoint with logging, Hard Offline Mode, and administrative policy	Implemented; hardening against redirects/proxies and update checking in offline mode has been implemented, tamper-evident chaining of the audit log (hash chain) is planned
Confidentiality	Secrets (cloud API keys, license, calendar token) encrypted in the Windows user profile (DPAPI) or macOS Keychain	Implemented
Confidentiality	Voiceprints encrypted, capture only with consent recorded per contact, no automatic background capture	Implemented
Confidentiality	Local network services (MCP integration, local AI server) protected by an access token newly generated per session –	Implemented

	relevant on shared computers/terminal servers	
Integrity	Signed installation packages (Authenticode, "Jenzer Innovations") for the Stable channel	Implemented for the Stable channel; Beta channel unsigned (see "What we do not offer today")
Integrity	Downloaded models and runtime components are checksum-verified before use	Implemented
Integrity	Software bill of materials (SBOM) and checksums published per release; automated dependency scanning in the development pipeline	Implemented
Availability	Fully offline-capable – no operation depends on our availability; update cadence controllable by your organisation itself (channel, timing)	Implemented
Availability	Deletion only through deliberate user action ("Delete All Data", deleting individual meetings) or after explicitly configured retention periods; default is unlimited retention	Implemented
Traceability	Network audit log, MCP audit log, consent log, each exportable	Implemented; tamper-evident chaining of the network audit log is planned
Traceability	Logs content-scrubbed since 0.6.65 (no transcript text, no contact names)	Largely implemented; the final logging level for released versions is still in progress

Responsibilities (Shared Responsibility)

Security here comes from the interplay of two sides. The following measures rest with **your organisation**, because the application itself has no access at the operating system or network level: disk encryption (BitLocker on Windows, FileVault on macOS – the database is today protected by the operating system user profile, but not additionally encrypted at the application level, see "What we do not offer today"), multi-factor authentication at Windows sign-in (e.g. Windows Hello for Business, FIDO2), an AppLocker publisher rule based on the certificate, and rolling out the administrative policy (`policy.json`).

LocalTranscript is responsible for: local processing with no cloud requirement, the network chokepoint including audit log and Hard Offline Mode, signed and checksum-verified installation packages, encrypted storage of secrets and voiceprints, content-scrubbed logs, retention and deletion functions, and the automated dependency scanning in the development pipeline.

Vulnerability Reporting

We accept security-relevant reports at hello@localtranscript.com and respond within two business days. A formal `SECURITY.md` / `security.txt` publication with a binding response time is planned but does not yet exist as of this document's date.

What We Do Not Offer Today

- **No encryption of the database/audio files at rest.** Meeting data is protected by the operating system user profile (access rights), not by dedicated application-level encryption. We recommend BitLocker/FileVault as a compensating measure. Application-level encryption is planned as a second development stage. Voiceprints are excepted from this – they are already encrypted today.
- **The Beta channel is unsigned.** Institutional installations should use only the signed Stable channel; the administrative policy can enforce this (autoUpdate control, see our document "Administrative policy").
- **No BSI C5, no ISO 27001, no SOC 2.** These certifications presuppose an operated cloud service; LocalTranscript operates none. In their place is verifiability: full network disclosure, exportable audit log, Hard Offline Mode, signed packages.
- **On the roadmap:** tamper-evident chaining of the network audit log, additional administrative policy sources (Windows registry, macOS "Managed Preferences"), policy-enforceable retention periods, application-level encryption at rest, formal SECURITY.md / security.txt publication.

Jenzer Innovations · Markgräflerstrasse · 4057 Basel, Switzerland UID CHE-334.396.708 ·
hello@localtranscript.com