SECURITY AUDIT

dunno

Version 1.3.0 · Build 36

Privacy-Focused Android Messaging — Direct Download Release

Document	DUNNO-SEC-2026-006
Date	08 September 2026
Classification	Confidential
Scope	Full source code audit (Kotlin / 233 production files) + practical device audit + static release-APK audit
Perspective	Published release build (.apk) for direct download from the dunno website

Complete security audit of dunno v1.3.0 (build 36), from the perspective of the published release build (.aab / .apk) for Google Play, supplemented by a practical on-device audit.

Table of Contents

1. Executive Summary

dunno v1.3.0 (build 36) is the direct-download release of this privacy-focused Android messaging application, distributed as a signed .apk from the dunno website. This report presents a security audit of the release build (.apk) from source and static analysis of the artifact, supplemented by the practical device audit of the identical core documented in DUNNO-SEC-2026-005 §6.

The core application — text chat, PTT walkie-talkie, and file attachments — maintains its strong, multi-layered security posture from v1.1.0/v1.2.0. All messaging is end-to-end encrypted (NaCl/libsodium) over the Tor network. Local storage is protected by SQLCipher with hardware-backed Keystore passphrases. The UserDataKey remains device-bound via DeviceKeyWrapper. Input validation, rate limiting, and signature verification remain consistent across all core protocols.

This version introduces the direct-download distribution layer — a new attack surface, assessed here: an offline license gate (DirectLicense) validating an Ed25519-signed license blob in the APK Signing Block, and an optional, opt-in update check (disabled by default; only runs after the user explicitly enables it in Settings or presses the manual check action). Both are **clean**, with one consciously accepted limitation (time-limited license backstop).

The core messaging functionality contains no CRITICAL or HIGH findings. Two findings (1 MEDIUM, 1 LOW) of limited impact are documented — both carry-overs from v1.1.0/v1.2.0 in unchanged source files.

The practical device audit of the identical core (DUNNO-SEC-2026-005 §6) found no successful exfiltration or read vector whatsoever: not debuggable, backup blocked, no external storage, no vulnerable exported components, FLAG_SECURE works in practice, notifications leak nothing.

The two optional WebRTC-based features — Fast Call (audio) and Video Call — remain unchanged from v1.2.0. Their privacy trade-offs are protected by mandatory user consent and documented separately in Section 7.

2. Scope & Methodology

In Scope

- Full source code review of 233 Kotlin/Java production files
- All files modified for v1.3.0 (licensing/update layer, video-call FGS, restore flow)
- AndroidManifest.xml, build.gradle.kts, proguard-rules.pro, network_security_config.xml
- Direct distribution layer: DirectLicense, ApkSigningBlock, UpdateChecker, UpdateVerifier, UpdateManifest, UpdatePrefs, LicenseGateScreens
- Practical device audit of the identical core (see DUNNO-SEC-2026-005 §6)
- Static analysis of the release build (.apk, release dex + merged manifest)

- Verification that all previous fixes remain intact

Out of Scope

- Server-side infrastructure (TURN/STUN relay, video signalling server on the VPS, the website's update-manifest signer)
- Build configuration not present in the published artifact
- Vulnerabilities in third-party libraries (unless caused by misconfiguration)
- Physical device security, social engineering
- Google Play Billing/subscription code (not present in this variant)

Methodology

Manual code review with architectural analysis. OWASP Mobile Top 10 (2024) reference. Severity: CRITICAL, HIGH, MEDIUM, LOW.

3. Changes in v1.3.0

Version 1.3.0 introduces the direct-download distribution layer (license gate + opt-in update check) and a video-call foreground service. No changes were made to core cryptographic, networking, storage, or authentication source files.

3.1 Direct Licensing (new)

- `license/` layer: DirectLicense (Ed25519-signed license blob in APK Signing Block), DirectLicenseGate.
- Offline activation with install window + 365-day build backstop. `BuildConfig.DEBUG` bypass is debug-only, absent from the release artifact.
- No network license check — activation is fully offline after install; no phone-home.

3.2 Update Check (new, opt-in)

- UpdateChecker fetches a signed update manifest over Tor (.onion first, HTTPS-via-Tor fallback — never cleartext). Disabled by default; user opts in via Settings or the manual check action.
- UpdateVerifier requires the update APK's signer to match the installed app; an update from any other key is rejected before install.

3.3 Video-call Foreground Service

- VideoCallForegroundService with foregroundServiceType camera|microphone for active video calls.

3.4 Restore Flow

- Restart after PIN unlock with OK dialog; restore flags survive process restart.

3.5 Verification of Previous Fixes

All fixes from v1.1.0/v1.2.0 remain intact: (a) FailedAttemptManager timestamp, (b) ProGuard OkHttp keep-rules, (c) FLAG_SECURE + recents blackout, (d) all notifications VISIBILITY_PRIVATE, (e) PinKeyDeriver zeroing.

4. Findings — Core Application

The core application has no CRITICAL or HIGH findings. Two findings of limited impact. Both are carry-overs from v1.1.0; their source files were not modified in v1.3.0.

Ref	Severity	Component	Title
M-01	MEDIUM	Networking	Custom HTTP/1.1 parser in OnionHttpClient
L-01	LOW	Networking	XOR obfuscation of relay .onion (self-documented)

No new findings in the core application for v1.3.0.

MEDIUM — M-01: Custom HTTP/1.1 parser in OnionHttpClient

Finding: OnionHttpClient implements a hand-written HTTP/1.1 parser instead of a proven library. Input validation is present (64 header cap, 128-char status line, 8192-char header line, 64 KiB response body). Parsing logic itself is hand-rolled. Unchanged from v1.1.0.

Impact: Limited to interactions with malicious Tor hidden services. All peers are trusted contacts verified via Ed25519 fingerprints.

Recommendation: Consider fuzz-testing the parser as defense-in-depth.

LOW — L-01: XOR obfuscation of relay .onion in TurnCredentialClient

Finding: Rolling XOR used to hide TURN relay .onion in the APK. Code explicitly states this is not a security measure. Self-documented. Unchanged from v1.1.0.

Impact: Negligible. Address trivially extractable from the APK.

Recommendation: Acceptable as-is.

5. Security Strengths

All security strengths from v1.1.0/v1.2.0 remain applicable. Key highlights:

STRONG — S-01: Transport v2 + Inbox v4: E2EE with minimal RAM exposure

ECDH-based per-message keys via cryptoSecretBoxEasy. UserDataKey zeroed after unlock. SessionKey-only in RAM. TEE compromise alone cannot decrypt stored messages.

STRONG — S-02: Hardware-backed key protection

DeviceKeyWrapper binds UserDataKey to physical device (non-exportable Keystore key). IdentityKeyWrapper protects identity secrets with user authentication. Keystore aliases cover DB passphrase, identity-wrap, device-bind, attachment-wrap and biometric keys.

STRONG — S-03: Comprehensive screen & notification privacy

FLAG_SECURE + recents blackout on incoming call/PTT activities. All notifications consistently use VISIBILITY_PRIVATE on lock screen.

STRONG — S-04: DeviceIntegrity + ProGuard anti-tamper

6 anti-Frida vectors (maps, TracerPid/ptrace, threads, port, artifacts, root/Magisk) trigger session key wipe on threat. Crypto classes deliberately obfuscated in release builds.

STRONG — S-05: BridgeConfig + EmergencyWipeManager

12 embedded obfs4 bridges with auto-rotation. Secure delete with zero-overwrite. Tor HS key and Keystore alias cleanup. Crash recovery via persistent markers. Wipe runs 13 phases (0–12, incl. sub-steps), idempotent.

STRONG — S-06: Robust FGS fallback

CallForegroundService and FastCallForegroundService handle SecurityException when microphone FGS type is blocked. Falls back gracefully without crashing.

STRONG — S-07: Backup protection (v1.3.0 verified)

`allowBackup=false` — no data exfiltration via Android backup; verified in the practical device audit of the identical core.

6. Practical Device Audit (USB, physical access)

Goal: attempt to read messages/data via realistic attack vectors on a non-rooted device. The vectors below were executed on a release build of the identical core (Samsung A16, Play variant v1.2.3; same core security as v1.3.0,

documented in DUNNO-SEC-2026-005 §6). server-04 differs only in the distribution layer (§7), which none of these vectors exercise.

#	Vector	Attempt	Result
1	Debug/root access	`run-as`, reading `/data/data`	✗ Blocked — not debuggable, Permission denied
2	adb backup (exfil)	`adb backup` + on-device confirmation	✗ No data — `allowBackup=false`, empty backup also after confirmation
3	Shared storage	`/sdcard/Android/data`, `/obb`	✗ Does not exist — app writes nothing to external storage
4	Exported components	Manifest analysis	✓ Safe — LAUNCHER / system broadcasts / framework services only
5	FileProvider	Path configuration	✓ Cache subfolders only, exported=false, grant-URI only
6	Hardcoded secrets	Dex strings	✓ No API/private keys
7	Cleartext network	network_security_config	✓ Cleartext only `.onion` + localhost
8	Test/debug backdoor	Dex scan	✓ No BuildConfig.DEBUG/testOnly/backdoor
9	Screenshots	adb-screencap during app	✓ FLAG_SECURE works — image ~99% black
10	Recents/task snapshot	dumpsys	✓ No content leak
11	Notification lockscreen	dumpsys notification	✓ VISIBILITY_PRIVATE, no content
12	External intents/deep links	Manifest + code	✓ No dangerous triggers

Conclusion device audit: the messages are **not readable**. No exfiltration or read vector succeeded on the release build of the identical core. The app data (SQLCipher + hardware Keystore + PIN/Argon2id) is unreachable without root + PIN.

7. Direct Distribution & Licensing Layer (new in v1.3.0)

This variant is distributed as a signed .apk from the dunno website instead of Google Play. It replaces the Play Billing layer with two mechanisms: an offline license gate and an opt-in update check (disabled by default, Tor-only when enabled).

7.1 DirectLicense — APK license blob

- Every distributed .apk carries a license blob (`DUNL`, 137 bytes) in its APK Signing Block, covered by the v2 signature — a per-download watermark.
- On first run the app verifies the Ed25519 signature against an embedded public key and requires install within an install window of the signed issue day.
- Once activated, the license stays valid locally; the only remaining limit is a 365-day build backstop that ages out every copy — after it, messaging stops while local data and backup/export remain available.

7.2 Update check — optional, opt-in, disabled by default

- No automatic network activity by default: the preference defaults to `false`; the user must explicitly enable it in Settings or press the manual check action.
- Only when enabled does UpdateChecker fetch the manifest from a hard-coded v3 .onion host first (HTTPS-via-Tor fallback). Never cleartext — fail-closed.
- UpdateManifest verifies an Ed25519 signature before any version info is trusted; UpdateVerifier requires the update APK's first signer SHA-256 to match the installed app.

7.3 Distribution-layer findings

Ref	Severity	Subject	Status
D1	LOW	License signer key embedded in app	Necessary for offline verification; compromise would let an attacker mint licenses — key must stay offline on the signing server
D2	LOW	365-day build backstop	Consciously accepted. Every copy ages out; local data and backup/export always remain available
D3	INFO	Update check (opt-in) over .onion	Verified: disabled by default; when enabled, onion-first, no cleartext fallback
D4	LOW	APK Signing Block parsing	Custom reader for the v2 block; bounded reads of the app's own APK. Recommendation: keep parser minimal if extended

Extra verified points:

- License gate placement: DirectLicenseGate wraps the app outside AppLockGate — blocking states render instead of content; EXPIRED lets the user through to their own data (messaging stopped, data intact).

Conclusion distribution layer: clean. Ed25519-signed license and update manifest, signer-matched updates, Tor-only update fetch, consciously accepted time backstop. No CRITICAL or HIGH findings.

8. Optional Features — WebRTC Audio & Video Calling

NOTE: These features are OPTIONAL and operate in a reduced-security environment. An explicit security warning is shown before use. User consent is mandatory. No changes were made to WebRTC source files in v1.3.0 — all findings from v1.2.0 remain applicable.

Ref	Severity	Component	Title
C-01	CRITICAL	WebRTC Audio	ICE ALL exposes IP via STUN
C-02	CRITICAL	WebRTC Video	Clearnet signaling bypasses Tor
M-02	MEDIUM	WebRTC Video	Server auth without nonce

CRITICAL — C-01: ICE policy ALL may expose real IP via STUN (WebRTC Audio)

Finding: WebRtcCallSession.kt line 201: iceTransportsType = ALL. STUN candidates reveal user's public IP to TURN/STUN server operator.

Impact: Real IP visible to TURN/STUN infrastructure. Mitigated by mandatory user consent.

Recommendation: Consider RELAY-only ICE as optional privacy mode.

CRITICAL — C-02: Clearnet signaling bypasses Tor (WebRTC Video)

Finding: VideoCallServer.kt connects to wss://assistant.eelke-sp.cc via clearnet — outside Tor. STUN uses clearnet endpoint.

Impact: IP, call timing, metadata visible to VPS operator. Mitigated by mandatory user consent.

Recommendation: Consider certificate pinning or Tor-based signaling.

MEDIUM — M-02: Server auth without nonce (WebRTC Video)

Finding: VideoCallServer.sendAuth() signs (contactId + timestamp) without random nonce. Replay possible within timestamp window.

Impact: Intercepted auth message can be replayed.

Recommendation: Add random nonce to auth message.

Privacy Comparison

Property	Chat + PTT	Fast Call (Audio)	Video Call
Transport	Tor (.onion)	Tor + STUN/TURN	Clearnet (WSS+STUN)
IP visible	No	Yes (STUN)	Yes (VPS+STUN)
Signaling	Tor (E2EE)	Tor (E2EE)	Clearnet WSS
Media	N/A	DTLS-SRTP	DTLS-SRTP
Metadata protected	Yes	Partial	No
User warning	No	Yes	Yes
Privacy level	Maximum	Reduced	Minimal

Distribution note: this variant has no Play Billing. The only optional network contact besides Tor/messaging is the opt-in update check (§7.2), disabled by default. No analytics, no trackers, no phone-home licensing.

9. Conclusion & Recommendations

9.1 Overall Assessment

dunno v1.3.0 (build 36) maintains the strong security posture of v1.1.0/v1.2.0. The changes — direct licensing, opt-in update check, video-call FGS, restore flow — introduced no security regressions in the core application.

The core messaging functionality has no CRITICAL or HIGH findings. Two low-impact findings carry over from v1.1.0 in unchanged files. The new distribution layer is clean: Ed25519-signed license and update manifest, signer-matched updates, Tor-only update fetch, consciously accepted time backstop. The practical device audit of the identical core confirmed no exfiltration or read vector succeeds on a release build.

The application's security architecture — proven cryptography, Tor-exclusive transport with bridge support, hardware-backed key storage, runtime integrity monitoring, and comprehensive screen/notification privacy — places it among the strongest privacy-focused communication tools available.

9.2 Recommendations by Priority

- [HIGH] Fix M-02: add random nonce to Video Call auth message.
- [HIGH] Implement certificate pinning for Video Call WSS connection.
- [HIGH] Consider RELAY-only ICE as optional privacy mode for Fast Call.
- [MEDIUM] Consider fuzz-testing the OnionHttpClient HTTP/1.1 parser.
- [LOW] Operations: keep the license-signing key offline; ensure every distributed .apk carries a fresh license blob with a correct issue day (install window).

9.3 Complete Findings Table

Ref	Severity	Scope	Component	Title
C-01	CRITICAL	WebRTC	Audio	ICE ALL exposes IP via STUN
C-02	CRITICAL	WebRTC	Video	Clearnet signaling bypasses Tor
M-01	MEDIUM	Core	Networking	Custom HTTP/1.1 parser
M-02	MEDIUM	WebRTC	Video	Server auth without nonce
L-01	LOW	Core	Networking	XOR obfuscation (self-documented)
D1–D4	LOW/INFO	Distribution	Licensing/Updates	Distribution findings (see §7.3)

Total: 5 core/WebRTC findings (2 CRITICAL, 0 HIGH, 2 MEDIUM, 1 LOW) + 4 distribution items (LOW/INFO). Core application: 2 findings (1 MEDIUM, 1 LOW). No new core findings in v1.3.0. Optional WebRTC features: 3 findings (2 CRITICAL, 1 MEDIUM) — unchanged. Both CRITICAL findings protected by mandatory user consent. Distribution layer: clean, no CRITICAL/HIGH. All previously resolved issues remain fixed. No regressions.

End of report. Classification: Confidential — for authorized recipients only.