# STELLAR PASSWORD MANAGER

## Security Architecture

Cloudless Software
April 2026

www.cloudlesssoftware.com
This document is intended for public distribution.

# 1. Why Local-First

The cloud-first model that dominates modern software treats centralized servers as the source of truth for user data. For most applications this is a reasonable tradeoff. For a password manager, it is a structural vulnerability.

When millions of users store credentials in a single cloud vault service, that service becomes a high-value target. The incentive for attackers scales with the concentration of data. A breach of one server can expose every customer simultaneously. The history of cloud-based password manager breaches demonstrates that this is not a theoretical risk.

Stellar takes a fundamentally different approach. It is built on local-first principles, meaning the user's device is the primary and only location where vault data exists. There is no Cloudless Software server that stores, processes, or has access to user passwords. There is no central database to breach, no API endpoint to exploit, and no authentication token that could be intercepted to gain access to someone else's data.

This is not a limitation of the product. It is the core design decision that everything else follows from.

## 1.1 Decentralized by Design

Centralized architectures create single points of failure. If the server goes down, users lose access. If the company is acquired, data policies may change. If the company ceases operations, the product may stop functioning entirely. Users of cloud-first password managers are dependent on the continued existence, goodwill, and security competence of the vendor.

Stellar eliminates this dependency. The encrypted vault is a local file on the user's device. It requires no internet connection to function. It requires no account with Cloudless Software. The core vault and all free-tier features require no subscription to remain operational. Premium features require an active subscription, but the encrypted vault itself is always accessible. If Cloudless Software ceased to exist tomorrow, every installed copy of Stellar would continue to function and every vault would remain readable.

The optional backup feature encrypts the vault and stores it in the user's own cloud storage (e.g., Google Drive via the Storage Access Framework). Cloudless Software never receives, processes, or stores the backup. The user retains full ownership and control of the encrypted file.

## 1.2 No Telemetry, No Tracking, No Data Collection

Stellar does not collect usage analytics, crash reports, device information, or any form of telemetry. There are no tracking pixels, no third-party SDKs that phone home, and no anonymous usage data. The application makes no network requests except those explicitly initiated by the user (launching a website from a saved URL, or creating a backup to their own cloud storage).

This is a deliberate architectural choice, not a marketing claim. There is no code in the application that transmits data to Cloudless Software or any third party. The privacy policy at

cloudlesssoftware.com reflects this: we do not collect, store, transmit, or sell personal data.

### 1.3 The Case for Data Decentralization

The broader trend in software is moving toward local-first and decentralized architectures. Projects like the Ink & Switch research lab have published extensive work on the benefits of local-first software: instant responsiveness, offline capability, user ownership of data, and long-term durability. These are not abstract ideals. They are practical engineering decisions that result in better software for the people who use it.

Password management is the category where these principles matter most. Credentials are the keys to a person's digital life. The argument for keeping those keys under the user's sole control, encrypted on their own hardware, with no third party in the loop, is not ideological. It is the most conservative security posture available. Stellar exists because we believe that posture should be the default, not the exception.

## 2. Encryption Architecture

Stellar uses a three-layer key hierarchy. The master password never directly encrypts vault data. This design allows master password changes without re-encrypting every record and limits the exposure of any single key.

### 2.1 Key Hierarchy

**Layer 1 — Master Password to BetaKey.** The user's master password is processed through Argon2id (3 iterations, 64 MB memory, 4 parallel lanes) with a random 128-bit salt to produce a 256-bit BetaKey. Argon2id is a memory-hard key derivation function recommended by OWASP and defined in RFC 9106. Its memory requirements make GPU and ASIC-based brute-force attacks prohibitively expensive. The BetaKey's sole purpose is to encrypt and decrypt the DataKey.

**Layer 2 — BetaKey to DataKey.** The DataKey is a random 256-bit key generated once during account setup using a cryptographically secure random number generator. It is encrypted with the BetaKey and stored in the database. On login, the BetaKey decrypts the DataKey, which is then held in memory for the session. When the master password is changed, only the DataKey's encryption is updated. The DataKey itself does not change, so no records need to be re-encrypted.

**Layer 3 — DataKey to RecordKeys.** Each vault record has its own unique 256-bit RecordKey, generated randomly at creation time. The RecordKey is encrypted with the DataKey. The record's sensitive data (username, password, URL, notes, custom fields) is encrypted with its RecordKey. This per-record isolation means that compromise of a single RecordKey exposes only that one record.

### 2.2 Encryption Algorithm

All encryption uses AES-256-GCM (Galois/Counter Mode), an authenticated encryption algorithm defined in NIST SP 800-38D. GCM provides both confidentiality and integrity: if encrypted data is tampered with, decryption fails with an authentication error rather than silently producing corrupted output. A fresh 96-bit initialization vector is generated for every encryption operation using a cryptographically secure random number generator.

### 2.3 Cryptographic Library

All cryptographic operations use Bouncy Castle Cryptography, a widely audited open-source library. Stellar does not implement any custom cryptographic algorithms. Key generation, encryption, decryption, and key derivation all delegate to Bouncy Castle primitives.

## 3. Data Protection

### 3.1 Data at Rest

Vault data is stored in a local SQLite database. Record titles and modification timestamps are encrypted with the DataKey. Record contents (usernames, passwords, URLs, notes, custom fields) are encrypted with per-record RecordKeys. The DataKey itself is encrypted with the BetaKey. Structural metadata (record identifiers used for database lookups, folder relationships) is not encrypted but contains no user credentials or personal data.

### 3.2 Data in Memory

Cryptographic keys exist in memory only while the vault is unlocked. They are held in an in-memory store that is never serialized to disk. On inactivity timeout, all keys are explicitly zeroed before the session is terminated. If the operating system kills the application process, all keys are lost and the user must re-authenticate. The master password is converted to a byte array immediately on entry, the text field is cleared, and the byte array is zeroed after key derivation completes.

### 3.3 Backups

The backup feature encrypts the entire database using AES-256-GCM with a key derived from the master password via Argon2id. The encrypted backup can be stored locally or in the user’s own cloud storage. Cloudless Software never receives or stores backup files.

## 4. Transparency

We believe that what a security product does not do is as important as what it does. Stellar does not:

- Transmit any data to Cloudless Software servers or any third-party service.
- Collect telemetry, usage analytics, crash reports, or device information.

- Store the master password. If it is lost, the vault cannot be recovered.
- Use custom or proprietary cryptographic algorithms.
- Make network requests unless explicitly initiated by the user.
- Protect against attacks on a device that is actively compromised with root-level malware while the vault is unlocked. No software-only solution can.

The master password is the single point of trust in Stellar's security model. Its strength determines the practical security of the vault. We recommend a passphrase of four or more random words, or a password of 14 or more characters with mixed character types.

# 5. Summary

Stellar's security architecture is built on the premise that user credentials should never leave the user's control. There is no cloud, no account, no server, and no third party with access to vault data. The cryptographic design uses industry-standard primitives (Argon2id, AES-256-GCM) from a vetted library (Bouncy Castle), organized in a three-layer key hierarchy with per-record isolation.

This document will be updated as the architecture evolves. Questions and responsible security disclosures can be directed to security@cloudlesssoftware.com.

Cloudless Software

1968 S Coast Hwy #2861, Laguna Beach, CA 92651

support@cloudlesssoftware.com | www.cloudlesssoftware.com