TRUST CENTER

Encryption Policy

Cryptographic standards, algorithms, and key-management practices.

Effective: May 21, 2026 Owner: Howell & Gibbs LLC Status: v1 – initial publication

Review cadence: Semi-annual, or after any material change to cryptographic systems

---

1. Purpose

This policy defines the cryptographic standards SendTax uses to protect customer data, and the key management practices that govern how those cryptographic operations are controlled. It supports the commitments made in the **Information Security Policy** and aligns with SOC 2 Common Criteria CC6.1 and CC6.7.

2. Scope

This policy applies to all cryptographic operations performed by or on behalf of SendTax, including:

- Encryption of customer data at rest (documents, database storage, backups)
- Encryption of data in transit (between clients, application servers, and sub-processors)
- Hashing of credentials and other sensitive values
- Cryptographic key generation, storage, use, rotation, and destruction
- Cryptographic implementations in source code authored by SendTax

This policy applies regardless of whether the cryptographic operation is performed by SendTax application code or by a managed sub-processor on SendTax's behalf. It does not govern the internal cryptographic implementations of our sub-processors (Clerk, Fly.io, Google Cloud, Cloudflare, Resend, Sentry, Modal), which are governed by the providers' own security programs and compliance certifications. Where

SendTax relies on a sub-processor for a cryptographic primitive — most notably KEK custody and TLS termination — that reliance is explicit in the relevant section below.

3. Definitions

Term	Meaning
AEAD	Authenticated Encryption with Associated Data. Provides confidentiality and integrity in a single primitive.
AAD	Additional Authenticated Data. Public context (e.g., document and filer IDs) cryptographically bound to a ciphertext at encryption time.
DEK	Data Encryption Key. A symmetric key used to encrypt a single unit of data (e.g., one document).
KEK	Key Encryption Key. A long-lived key used to wrap (encrypt) DEKs. KEKs are not used to encrypt data directly.
Envelope encryption	Pattern in which data is encrypted by a per-record DEK, and the DEK itself is encrypted by a KEK.
HSM	Hardware Security Module. A tamper-resistant device that performs cryptographic operations without exposing key material.
KMS	Key Management Service. A managed service that custodies KEKs and performs wrap/unwrap operations. SendTax uses Google Cloud KMS.

4. Approved algorithms and minimum strengths

SendTax uses only modern, widely reviewed cryptographic algorithms at minimum strengths consistent with current NIST guidance.

Purpose	Approved primitive	Minimum strength
Symmetric authenticated encryption	AES-GCM (12-byte nonce, AEAD with AAD binding)	256-bit key

Key wrapping	GCP Cloud KMS (AES-256-GCM under the hood, HSM-backed available)	256-bit key
Transport encryption	TLS	1.2 or higher (TLS 1.3 preferred)
Password hashing	bcryptEncryption Policy	Provider default (Clerk)
Token signing	RS256 / ES256 (handled by Clerk for authentication JWTs)	n/a
Cryptographic hashing	SHA-256 or SHA-512	256-bit output
Random number generation	OS-provided CSPRNG (Python `secrets`, Node `crypto.randomBytes`, `cryptography` library secure RNG)	n/a

Selection rationale: AES-256-GCM provides authenticated encryption in a single primitive and is FIPS-approved. New cryptographic code must use one of the primitives in this table.

5. Prohibited algorithms

The following algorithms and protocols are not used in any production code path:

- **MD5** (collision-broken; not approved even as a non-cryptographic checksum in SendTax code, to avoid confusion)
- **SHA-1** and HMAC-SHA-1 (collision-broken; non-cryptographic uses such as integrity checksums in package manifests are acceptable)
- **DES**, **3DES / Triple DES** (insufficient key length, deprecated by NIST)
- **RC4** (biased keystream)
- **TLS 1.0** and **TLS 1.1** (deprecated by IETF RFC 8996)
- **SSL** (all versions)
- **ECB mode** for any block cipher
- **AES-CBC without integrity** — if AES-CBC is required by a third-party integration, it must be accompanied by an explicit HMAC

As of the effective date of this policy, no prohibited algorithm appears in any SendTax source code or production configuration. This was confirmed by a codebase audit on May 21, 2026 and will be re-verified at each policy review and whenever a new third-party library with cryptographic dependencies is added.

Adoption of any prohibited algorithm — for example, to integrate with a third party that mandates one — requires a documented exception under §11.

6. Encryption in transit

6.1 Public-facing traffic

All traffic between end-user clients, SendTax application services, and public APIs is encrypted using **TLS 1.2 or higher**, with TLS 1.3 preferred where supported by the client. *(Verified 2026-05-26 at the Fly.io edge: TLS 1.3 is negotiated on every customer-facing host — send.tax, app.send.tax, pro.send.tax, admin.send.tax — via `openssl s_client -tls1_3` probes; TLS 1.2 remains the floor.)* TLS termination is performed at the Fly.io edge.

HTTP Strict Transport Security (HSTS) is configured in every Next.js application with `Strict-Transport-Security: max-age=63072000; includeSubDomains; preload` (see `st-apps/apps/*/next.config.js`), and HTTP requests are redirected to HTTPS. *(Verified 2026-05-26: the header is observed on the `send.tax` apex and on direct content responses from each Fly-hosted app; submission of `send.tax` to https://hstspreload.org is the remaining Phase-2 task so a cold browser receives the policy without first visiting the apex.)*

Certificates are issued and rotated automatically by our hosting provider; SendTax does not manage long-lived certificate private keys directly.

6.2 Service-to-service traffic

- Communication between SendTax application services occurs over Fly.io's private network (`fly-internal`) and is not exposed to the public internet.
- Database connections from the application to PostgreSQL use TLS within the Fly.io private network.

- Outbound calls to managed sub-processors (Clerk, Google Cloud KMS, Cloudflare R2, Resend, Sentry, Modal) use the providers' published HTTPS endpoints with TLS 1.2 or higher.

6.3 Configuration

SendTax does not pin specific cipher suites at the application layer; we rely on the secure defaults of our TLS implementations (Fly.io edge, Python `ssl`, Node.js `tls`), which exclude all algorithms listed in §5. Underlying runtimes (Python 3.12 standard library, libpq, Node 20+) enforce TLS 1.2 or higher by default.

7. Encryption at rest

Customer data is protected by multiple layers of encryption at rest.

7.1 Application-layer envelope encryption (customer documents)

Customer documents — including tax forms, identity documents, financial statements, and prepared returns — are encrypted by SendTax application code **before** they are written to object storage. The scheme is as follows:

1. A unique 256-bit **DEK** is generated for each document using a cryptographically secure random source.
2. The document is encrypted with the DEK using AES-256-GCM, with a freshly generated 12-byte nonce.
3. The DEK is wrapped (encrypted) by the production **KEK** held in Google Cloud KMS.
4. The wrap operation is bound to **Additional Authenticated Data (AAD)** of the form `sendtax-doc/v2/{document_id}/{filer_id}`. KMS refuses to unwrap the DEK if a different AAD is supplied at decrypt time, and the AEAD ciphertext itself also incorporates the AAD — making ciphertext substitution across documents detectable.
5. The wrapped DEK is stored in the database alongside the document record, with a versioned magic prefix (`\x00sendtax-v2\x00`) so future format upgrades can be made transparently.

The same envelope-encryption scheme is used for other categories of sensitive data with appropriately scoped AAD: Friends & Family upload blobs

(`sendtax-ff/v2/{upload_id}`) and anonymized training data contributions (`sendtax-training/v2/{contribution_id}`).

This scheme is implemented using the `cryptography` library's `AESGCM` primitive (an authenticated encryption mode reviewed and recommended by cryptographers).

7.2 Storage-layer encryption

Customer documents are also encrypted at rest by Cloudflare R2 using AES-256 by default. This is independent of, and applied in addition to, the application-layer envelope encryption described in §7.1 — customer documents are therefore encrypted twice.

Database storage and backups are encrypted at rest by Fly.io Managed Postgres. Backup cadence, retention, point-in-time recovery, and encryption are provided transparently by the managed service per Fly.io's published terms. SendTax does not maintain a separate backup pipeline outside the managed service.

7.3 Endpoint encryption

Devices used by SendTax personnel to access customer data have full-disk encryption enabled (Apple FileVault). See the **Access Control Policy** §5.6.

7.4 Application secrets

Application secrets (API keys, database credentials, KMS configuration) are stored in:

- **Fly.io encrypted secrets** — for runtime-injected secrets, scoped per environment (development, staging, production)
- **1Password vaults** — for human-accessible operator credentials, segmented by environment (Clerk-Dev, Clerk-Staging, Clerk-Production, and equivalents)

No production secrets are stored in source control. Automated secret-scanning runs in CI on every change.

8. Key management

8.1 Key generation

- **DEKs** are generated by the application using the `cryptography` library's secure RNG, which delegates to the OS CSPRNG.
- **KEKs** are generated by Google Cloud KMS within its own cryptographic boundary. SendTax never sees raw KEK material.

8.2 Key storage and access

- **Production KEKs** are stored exclusively in Google Cloud KMS. The underlying key material does not leave the KMS boundary; only wrap and unwrap operations are exposed to SendTax services.
- **Hardware Security Module (HSM)** backing for production KEKs is supported by the implementation via dual-key configuration. SendTax will adopt HSM-backed keys when customer or regulatory requirements warrant.
- **Development KEKs** are local AES-256 keys configured via the `ENCRYPTION_LOCAL_KEK_HEX` environment variable. Local keys are used only against development databases containing synthetic data.

8.3 Authentication to the key management service

SendTax services authenticate to Google Cloud KMS using **Workload Identity Federation**. No long-lived service-account key files are stored on application servers. A long-lived service-account key (`GCP_SA_KEY_B64`) is supported as a fallback authentication mode but is not the production default.

8.4 Key rotation

- **KEKs.** Google Cloud KMS supports both operator-initiated and automatic key rotation. Because DEKs are wrapped — not encrypted directly — by the KEK, the KEK can be rotated without re-encrypting customer documents: new DEKs are wrapped under the current KEK version, while existing wrapped DEKs continue to be unwrappable under their original KEK version. SendTax is moving toward **annual automatic KEK rotation** in GCP KMS. *(2026-05-26 status: automatic rotation is a 30-second console toggle pending in GCP KMS for the production keyring — project `sendtax-prod`, location `us-east1`. Until*

that toggle is flipped, KEK rotation is operator-initiated and follows the scheduled review cadence below.)

- **DEKs.** A new DEK is generated for every document. DEKs are used exactly once and are never reused across documents or rotated periodically; the per-document scheme makes periodic DEK rotation unnecessary.
- **Application secrets.** Rotated as part of the secret-rotation procedure documented in internal runbooks, and additionally whenever a leak is suspected or a team member with access offboards.

8.5 Key destruction

- **DEKs in memory** are released by the `cryptography` library's standard lifecycle handling and are not persisted in plaintext.
- **Wrapped DEKs at rest** exist only in the database row associated with their document. When a document is deleted (§4.3 of the Data Retention & Deletion Policy), the row is removed and the wrapped DEK is destroyed with it. Because each DEK is unique to its document, this is sufficient to render the document's ciphertext unrecoverable from SendTax systems, independent of R2 object propagation.
- **Retired KEK versions** in Google Cloud KMS may be destroyed according to GCP KMS's destruction lifecycle (24-hour soft-delete followed by destruction). SendTax does not currently use KMS key-version destruction as a routine disposal mechanism, since per-document DEK uniqueness already provides per-document unrecoverability. KMS key-version destruction remains available for bulk-deletion scenarios that may arise (for example, mass tenant offboarding) and would render all documents wrapped under that KEK version simultaneously unrecoverable.
- **Application secrets** are revoked at the source (provider console or 1Password) on rotation; references in Fly.io encrypted secrets are overwritten in the same change.

9. Customer-managed encryption keys

SendTax does **not currently offer customer-managed encryption keys** (BYOK / CMK). All cryptographic keys protecting customer data are generated, stored, and operated by SendTax in Google Cloud KMS as described in §8.

This is a deliberate scope decision for the current product. Customer-managed key support is not on our current roadmap and would require changes to the application's key-selection layer before it could be supported.

10. Cryptographic library policy

SendTax uses only well-maintained, widely-reviewed cryptographic libraries. SendTax does not introduce custom or "rolled-our-own" cryptographic primitives. The current inventory is:

Library	Used for
`cryptography` (Python)	All application-layer encryption (`AESGCM` from `hazmat.primitives.ciphers.aead`)
`passlib[bcrypt]` (Python)	Bcrypt password hashing (residual; authentication is delegated to Clerk)
Google Cloud KMS client	Key wrap/unwrap operations
Browser / runtime native crypto	Frontend cryptographic operations (limited; no direct crypto-library dependencies on the frontend)

Dependencies are scanned automatically in CI for known vulnerabilities. Cryptographic-library security advisories are treated as high priority and patched on an expedited schedule.

11. Audit and verification

- A codebase audit on **May 21, 2026** confirmed no usage of any algorithm listed in §5.
- This audit will be re-run at each policy review (semi-annual) and any time a new third-party library is added that performs cryptographic operations.
- Findings and any required remediation are recorded internally.

12. Exceptions

Adoption of any algorithm or practice that deviates from this policy — including, but not limited to, integrating with a third party that mandates a prohibited algorithm — requires:

1. Written justification describing the business need and the threat model
2. Approval from a second SendTax operator
3. A removal target date for any prohibited algorithm, or an explicit acknowledgment that the exception is permanent for a specific bounded use case
4. Compensating controls (e.g., wrapping the prohibited primitive inside an approved authenticated layer) where technically possible

Exceptions are reviewed at the next policy review and are not allowed to become indefinite without explicit re-authorization.

13. Post-quantum cryptography

SendTax monitors developments in post-quantum cryptography and will evaluate and adopt post-quantum algorithms as NIST migration guidance and supporting library, runtime, and hosting-provider tooling mature. Symmetric encryption at the strength used by SendTax (AES-256) is expected to remain secure against foreseeable quantum attacks; the primary migration target is asymmetric cryptography used in transport-layer key exchange, which is largely handled by our hosting and sub-processor providers rather than by SendTax application code.

14. Enforcement

Violations of this policy may result in revocation of access, termination of employment or contract, and, where applicable, civil or criminal referral.

15. Related policies

- **Information Security Policy** — umbrella policy that this document supports
- **Access Control Policy** — controls on who can authorize cryptographic operations

- **Incident Response Policy** — response to suspected key compromise
- **Data Retention & Deletion Policy** — cryptographic erasure as a disposal method
- **Vendor Management Policy** — review of sub-processors that perform cryptographic operations on SendTax's behalf

16. Document control

Version	Date	Author	Notes
1.0	May 21, 2026	Holly Gibbs	Initial publication

17. Contact

Security disclosures	**security@send.tax**
Privacy inquiries	privacy@send.tax
Operating entity	Howell & Gibbs LLC (operator of SendTax)

Footnotes

1. Password hashing is performed by Clerk, our identity provider. Clerk's bcrypt cost factor and related configuration follow Clerk's published security specifications; SendTax does not store user passwords directly and does not implement password hashing in its own code. ↩