UnityFreak Security Whitepaper

Version 1.1.0 · dated 2026-10-15 · status draft

> **Draft.** This paper has not yet been independently fact-checked. Treat every statement as a self-description, not an audit finding.

1. How to read the evidence classes
2. Architecture and data flows
3. Encryption at rest
4. Encryption in transit
5. Key management and rotation
6. Backup and restore
7. Tenant isolation
8. Audit logging
9. Single sign-on and SCIM
10. Retention, deletion and legal hold
11. Open items and what this paper does not claim

This paper describes how UnityFreak protects customer data today: what is implemented, what is only partly implemented, and what is not implemented. It is a self-description by the people who operate the platform. It is not an audit report, a penetration-test result or a certification, and it makes no claim of SOC 2, ISO 27001 or CSA STAR status.

Every substantive statement carries a claim label such as C-ENC-01 and an evidence class. The classes are defined in the next section. A reviewer-only claim trace, held privately, maps each label to dated source code or to deployment and drill evidence. Where we could not gather evidence, the statement says so.

How to read the evidence classes

- **Code**: the behaviour is implemented in the platform source at the reviewed commit. This does not by itself prove that the production deployment matches the source.
- **Deployed**: the behaviour is code-verified and is also supported by dated deployment or drill evidence from the production environment.

- **Partial**: implemented, with known gaps that are listed next to the claim.
- **Not implemented**: the control does not exist. It is listed so that nobody assumes otherwise.
- **Unverified**: we believe this holds but have not gathered evidence for it.

Document control: version 1.1.0, dated 2026-10-15, status draft. The source was checked against the platform source as of commit d1c6d4f14 and against the compliance readiness table dated 2026-10-15. The backup section was additionally checked against the production backup configuration, read without changing it, on 2026-10-15. The status stays draft until an independent reviewer has traced every claim.

Architecture and data flows

UnityFreak is a project-management and version-control platform. Customers use it through a web application, a public API, a git service reachable over HTTPS and SSH, and desktop and command-line clients.

- The platform is built from a web application, an API service, a git hosting service, a real-time collaboration service and background workers. C-ARC-01 · Code
- Application data lives in a PostgreSQL database. Short-lived state, rate limiting and job queues use Redis. C-ARC-02 · Code
- Git repositories are stored on a dedicated block-storage volume. Large-file (LFS) objects and uploads are stored in object storage. C-ARC-03 · Deployed
- Browsers talk to the web application, which calls the API. Git clients talk to the git service directly. The API calls the database, the queue and the third-party services listed on the public subprocessor page. C-ARC-04 · Code
- The production environment currently runs on a single server. There is no automatic failover and no multi-region redundancy. C-ARC-05 · Deployed
- Customer-enabled integrations (for example chat, incident and design tools) send data only to the destination the customer connects. C-ARC-06 · Code

---

Encryption at rest

Encryption at rest is scoped, not blanket. The accurate picture, per data class:

- Selected secrets are encrypted in the application with AES-256-GCM, using a random initialization vector and an authentication tag per value. The platform refuses to encrypt when its key is missing, so there is no silent plaintext fallback on write. C-ENC-01 · Code
- The encrypted secret fields we verified are single sign-on client secrets, data-loss-prevention webhook secrets and stored integration credentials. C-ENC-02 · Code
- Values that do not carry the encrypted prefix are read back as legacy plaintext. Old values written before encryption was introduced would therefore remain readable as plaintext until they are rewritten. C-ENC-03 · Partial
- Passwords are stored as bcrypt hashes and SCIM provisioning tokens as one-way SHA-256 hashes. Neither is reversible. C-ENC-04 · Code
- Customer content such as tasks, comments, documents and messages is stored in PostgreSQL and is not encrypted by the application. We have not gathered evidence that the database storage itself is encrypted at rest. Do not read this paper as saying it is. C-ENC-05 · Unverified
- Git repositories are stored on a dedicated storage volume. We have not yet verified at-rest encryption for that volume or for the database. C-ENC-06 · Unverified
- Large-file objects sit in Cloudflare object storage, which encrypts stored objects at rest by default. This is the provider's control. C-ENC-07 · Deployed
- Backups are produced with restic, which always encrypts its repositories. C-ENC-08 · Deployed

---

Encryption in transit

- The web application sends an HTTP Strict Transport Security header on every response, with a two-year lifetime, subdomain coverage and the preload flag. C-TLS-01 · Code
- Git is served over HTTPS and SSH. C-TLS-02 · Code
- Certificate management and TLS termination for the production hosts are configured outside the application source. This review did not independently verify the production TLS configuration, protocol versions or certificate renewal. C-TLS-03 · Unverified

---

Key management and rotation

- The application encryption key is a 256-bit value supplied to the service through its environment. The service validates its format and will not encrypt without it. C-KEY-01 · Code

- Ciphertext does not record a key identifier, and the service holds one active key. Rotating the key therefore means re-encrypting stored values by hand. There is no automated rotation, no managed key service and no hardware security module. C-KEY-02 · Not implemented
- The backup encryption passphrase is generated and held by the operators outside the source repository. A documented custody and rotation procedure was not reviewed. The database archive uses a second, separate encryption key, also held by the operators outside the production server and the source repository; its custody is recorded, and a rotation procedure has not been reviewed. C-KEY-03 · Unverified
- Sessions can be revoked on demand, and refresh tokens rotate with reuse detection that revokes the whole session family. C-KEY-04 · Code

---

Backup and restore

- Repositories and a logical dump of the database are backed up every hour into an encrypted restic repository. Retention keeps the most recent 48 hourly snapshots, then 30 daily, 12 weekly and 12 monthly generations, and a one percent sample of the stored data is read back and verified on each run. C-BAK-01 · Deployed
- Large-file objects are synchronised every hour, as part of the same backup run, to a separate backup bucket with its own credentials. The copy is compared by checksum after each run, and objects deleted or replaced in live storage are moved to a retained-history area instead of being removed from the backup at once. C-BAK-02 · Deployed
- Backup freshness is checked every five minutes. An alert fires when the repository or large-file success marker is older than two hours, or the database base-backup marker is older than 26 hours; continuous database archiving has its own failure and backlog checks. The alert path was exercised deliberately when the freshness check was introduced in July 2026, against the longer threshold used at that time; we have not recorded a repeat at the current thresholds. C-BAK-03 · Deployed
- Recovery point objective is up to about one hour for repositories and large-file objects. For the database it is about five to ten minutes through continuous archiving, provided the archive encryption key can be recovered; if it cannot, the fallback is the hourly dump, about one hour. These are design and measurement figures, not contractual commitments. C-BAK-04 · Code

- A full restore drill was first run on 2026-07-13, on a separate server with a small pre-launch dataset. It was run again on 2026-09-30 on a separate server, restoring from the current backup repository (about 470 MB of data in 20 repositories). Each run restored the repositories and the database and passed repository integrity checks, clone tests and row reconciliation; the 2026-09-30 run took about a minute. No later drill is on record, and the quarterly repeat planned for October 2026 has not yet been recorded. The data is still small and pre-launch, so restore time at production scale has not been measured. A configured backup is not the same as a proven restore, and only these small drills support this claim. C-BAK-05 · Partial
- Recovery means provisioning a replacement server and restoring from backup. There is no live standby. C-BAK-06 · Deployed
- Point-in-time recovery of the database is available. Database write-ahead log segments are archived continuously, at least every five minutes, encrypted with a key held by the operators before they leave the server, and a base backup is taken daily. Base backups are kept for a shorter window than repository snapshots. A timed point-in-time restore from the encrypted archive onto a separate server succeeded on 2026-09-30. Recovery depends on the operators being able to supply the archive key, and it has been exercised only on the current small dataset. C-BAK-07 · Deployed

---

Tenant isolation

- Customer separation is enforced in application code on each request, by checking organization and project membership and role permissions before data is returned or changed. C-ISO-01 · Code
- The database does not use row-level security policies, so isolation rests on the application checks and not on a second database-level barrier. C-ISO-02 · Code
- Six organization roles with fine-grained permissions are defined, and project access resolves custom-role permissions. C-ISO-03 · Code
- Authorization has known defects. The readiness table lists escalation and bypass paths in custom roles, and an export that bypasses some project permissions. Fixes for the defects found in the 2026-09-27 closeout review are in the source but the production deployment of those fixes is still pending. C-ISO-04 · Partial
- A systematic authorization review of the whole codebase and an independent penetration test have not been completed. C-ISO-05 · Not implemented

Audit logging

- Organization-level events such as sign-ins through single sign-on, provisioning changes, role changes and audit-log exports are written to an organization audit log. C-AUD-01 · Partial
- Audit rows are protected against modification by a database trigger. This guards against application bugs. It is not tamper-proof or write-once storage, and a database administrator could alter the data. C-AUD-02 · Code
- Audit data is archived to private object storage before it is deleted. The default retention is 400 days and the minimum configurable value is 180 days. C-AUD-03 · Code
- Logs can be exported as CSV or JSON, and can be streamed to a customer system by webhook. Failed deliveries after retries do not yet notify organization owners. C-AUD-04 · Partial
- Audit-log pages and exports read with a cursor on a stable sort key rather than an offset, so long reads no longer skip events, and changes to custom fields, boards and automation rules are now written to the log. We have not enumerated every project mutation, so complete coverage of project changes is not claimed. C-AUD-05 · Partial

Single sign-on and SCIM

- SAML and OpenID Connect single sign-on are supported. SAML assertions must be signed and are verified, and the OpenID Connect flow uses PKCE, a single-use state and a nonce. C-SSO-01 · Code
- Organization administrators can require single sign-on and verify ownership of an email domain. The 2026-09-27 closeout review found and fixed enforcement races in the source; those fixes await production deployment. C-SSO-02 · Partial
- A session that was itself created by single sign-on refreshes without being blocked by the organization's require-single-sign-on policy; sessions created by password or personal sign-in are still refused at refresh in such an organization. C-SSO-03 · Code
- SCIM 2.0 provisioning is supported with a hashed token. Deprovisioning removes the person's access to the organisation and, for an account the organisation exclusively owns, revokes all of its sessions, API keys and SSH keys; a person who also has a personal account or belongs to another organisation keeps that account and only their membership is deactivated. C-SSO-04 · Code
- Two-factor authentication (TOTP) is available. The whiteboard relay checks the organization's two-factor and network policy when a connection is made and disconnects live sessions when access is lost. C-SSO-05 · Code

Retention, deletion and legal hold

- Organizations can configure a message-retention window for governed project conversations, and a scheduled job purges messages older than the window. C-RET-01 · Code
- A scheduled job applies retention to project-management content and checks legal holds before purging. C-RET-02 · Code
- Administrators can place legal holds at organization, project or user scope. Held content is preserved ahead of retention and deletion. The 2026-09-27 closeout review found a gap where task evidence could be purged under a narrower hold; the fix is in the source and awaits production deployment. C-RET-03 · Partial
- Deleting an account anonymises the profile (name, email, avatar and similar fields), releases the username, revokes sessions, API keys, linked sign-ins and SSH keys, blanks the password and blocks sign-in. The same steps run whether you delete the account yourself or an administrator processes your request; the administrator path used to do less, and the fix is in the source and awaits production deployment. While a hold placed on that user is in force, deletion is not carried out: the request is queued and runs automatically once the hold no longer applies. Neither path was found to erase content the user authored inside projects. C-RET-04 · Partial
- Backups keep deleted data until the snapshot that holds it ages out: hourly snapshots for 48 hours and monthly snapshots for up to about twelve months. Database base backups are kept for a shorter window. A copy of an earlier backup repository, retired when backups moved to their current storage in August 2026, was left in place with no expiry set, and we have not verified whether it has since been removed. C-RET-05 · Partial
- Customers can export their data. Organization exports are known to bypass some project-level permissions. C-RET-06 · Partial

Open items and what this paper does not claim

- No independent audit or certification exists. A third-party penetration test and a SOC 2 Type I assessment are planned, with no date set and no provider engaged.
- The items marked unverified or not implemented above are the ones an independent reviewer should examine first.
- This paper will be re-dated and re-versioned whenever a claim changes. It does not describe any control as effective merely because it is configured.

Version 1.1.0, 2026-10-15. Source SHA-256 e3077d0f3bd8ba4e. Generated from the reviewed source.