

VeriChar Whitepaper

Verifiable charity on the XRP Ledger

Proof comes before payment.

Brianne Salato

Founder and Product Architect

Version 0.7 (Draft) · 22 October 2026

verichar.net · VeriChar@proton.me · X: [@VeriCharOnChain](https://twitter.com/VeriCharOnChain)

Document status

This is **Version 0.7**, a draft. It replaces Version 0.6 for product and business-model accuracy as of **22 October 2026**.

What changed from 0.6

Topic	v0.6	v0.7
Who pays platform fees	Free for donors and charities; 1% vendor settlement fee planned	Free for donors, charities, and vendors. No settlement fee on giving flows
How VeriChar is funded	1% vendor fee + enterprise CSR licensing	Donations to VeriChar (trackable) + enterprise SaaS
Site / demo copy	Mixed older fee language	Aligned with free-for-all + dual revenue paths

VeriChar is a **working prototype**. The public website and its four-step demonstration are **simulations**. They do not connect a wallet, submit an XRP Ledger transaction, verify a charity, accept a proof upload, or move money. A **Testnet-only API foundation** exists. The public HTTP API does not submit ledger transactions and does not touch Mainnet.

XRPL Testnet verification (automated). Backend continuous integration on GitHub Actions runs the full Node test suite against the live **XRP Ledger Testnet**, including faucet-funded wallets, conditional EscrowCreate / EscrowFinish / EscrowCancel (PREIMAGE-SHA-256), fee checks against XRPL.org, vendor balance-change confirmation, and reserve reads. As of **22 October 2026**, those runs complete successfully (for example **37 tests, 0 failures**). Public logs are available under the repository Actions tab. This is **not** Mainnet certification and **not** a claim that the public product processes donations.

The flow described in later sections is the **intended design**. Legal and security review are still required before any real (Mainnet) funds are handled.

Statements about how the XRP Ledger works were checked against XRPL.org documentation and are tagged **[S1]-[S24]** where retained from prior drafts. Design details marked **to be confirmed** are not yet verified against the XRPL specification or production operations.

This document is not legal, financial, tax, or investment advice.

What is live today

Surface	Now	Not yet
Public site	Mission, proposed flow, transparency limits, business model, four-step simulation, Enterprise Portal preview	Real donations; charity verification; published needs;

		proof uploads; vendor payments
Simulation	Sample need moves listed → locked → proof submitted → released	Wallet, XRPL transaction, verification, upload, or payment
Testnet API	Public /healthz; authenticated tenant context; validated Testnet account lookup. Bearer tokens stored only as SHA-256 hashes, bound to one tenant and explicit scopes. Browser is not given a token	Saved tenant data; product transaction submission; oracle productization; OAuth; Mainnet
Testnet integration tests (CI)	Live Testnet: conditional XRP escrow create / finish / cancel; finish fee formula; vendor balance delta; network and account reserves. Runs on every push to main	Product UI settlement; RLUSD escrow confirmation; real OCR / delivery oracle
Charity interest	Email VeriChar@proton.me . Verification will be free when it opens	Open registration; IRS lookup; any organization shown as verified

Version 0.7 exists so a reader of verichar.net and a reader of this paper see the **same product**.

Contents

1. What is live today
2. Executive summary
3. The problem
4. The VeriChar approach
5. System architecture
6. Cryptographic proof, explained
7. Donor, charity, and vendor experience
8. Planned modules
9. Business model
10. Security and risk
11. Compliance considerations
12. Roadmap
13. Conclusion
- Appendix A. Sources
- Appendix B. Open design questions

2. Executive summary

Charitable giving runs on trust, and trust is hard to verify. Donors rarely see what their gift actually bought. Charities carry the burden of proving impact after the money is already spent. Fraud, mismanagement, and vague “administrative costs” thrive in that gap.

VeriChar (Verified Charity On-Chain) is a verification layer for charitable giving, built on the XRP Ledger (XRPL).

Intended design

- Every donation funds **one specific, itemized need**.
- Funds are intended to be paid in **RLUSD** (a USD stablecoin on the XRPL) where escrow of that asset is confirmed feasible; XRP remains required for fees and reserves.

- Funds **lock in escrow** on the ledger and **release directly to the vendor** only when **two proofs** are verified: an **itemized merchant invoice** and **proof of delivery**.
- The **charity never holds the cash**.
- Settlement steps are written to a **public ledger** that anyone can audit.

That design is **not operating with real money today**. VeriChar does not currently process real donations.

Who it is free for

VeriChar is **free for donors, charities, and vendors**. There is **no settlement fee** on giving flows and **no 1% vendor fee**.

How the platform is funded

1. **Donations to VeriChar itself** — people can support the platform and, as the system matures, **track almost every dollar** of platform funding with the same transparency principles applied to charity flows.
2. **Enterprise SaaS** — licensing of audit-ready CSR / impact reporting tools to corporate customers.

Core principle. Proof comes before payment, not after. Trust should be enforced by code, not demanded by slogans. The code that would enforce full settlement is not finished. Until it is, the public demonstration says so on the screen.

3. The problem

Today's giving systems ask donors to take organizations at their word. Four structural problems follow.

Opacity. Donations are usually pooled into general funds. A donor cannot trace a specific dollar to a specific purchase.

Diversion and fraud. Unverified organizations, misused funds, and inflated costs are difficult to detect. When money moves first and reports follow later, problems surface only after the damage is done.

The burden of proof. Honest charities must assemble reports and receipts to demonstrate impact. The process is slow, manual, and hard for an outsider to audit.

Privacy versus transparency. Many donors want to give without being tracked, yet still want proof that their gift was used properly. Existing platforms usually force a choice between the two.

4. The VeriChar approach

VeriChar changes the order of events. Instead of paying first and reporting later, funds would be **locked first** and **released only when the evidence exists**.

The sequence below is the **intended** flow. The public simulation shows these steps with sample data and stops there.

1. **Donor** funds one specific, itemized need.
2. **Escrow** locks funds on the XRPL with the **vendor** as the fixed destination.
3. **Charity** submits an itemized invoice and proof of delivery.
4. **Review** (oracle / verification service) checks both proofs. Neither is optional.
5. **Vendor** receives funds only if both proofs pass; otherwise funds can return to the sender after expiry.

Building blocks (intended)

Verified charities. The intended check binds an organization to its EIN (or equivalent) and to a wallet address. The XRPL supports a two-way link between an account and a domain: the account's Domain field plus an `xrp-ledger.toml` file hosted at that domain **[S12]**. The EIN check itself happens off-ledger. XRPL Credentials are another possible fit: an issuer creates a credential for a subject account, and the subject must accept it **[S19]** (*to be confirmed for VeriChar*). **None of this runs on the website today**. There is no IRS lookup, and no charity is marked verified.

Itemized needs. A charity would list a specific need with a price and a named vendor, for example “4 cases of water — \$20.” Donors would fund that item, not a general pool. **Need publishing is not live.**

Escrow. The donation would lock in an escrow on the XRP Ledger. An escrow locks XRP or fungible tokens that cannot be used or destroyed until conditions are met [S17]. The escrow names its recipient (the vendor) when it is created; finishing it delivers the funds to that recipient [S17]. **VeriChar does not yet create escrows in production.** Backend modules exist for Testnet experimentation (conditions, escrow helpers, reserves, monitoring patterns).

Two proofs. The charity would submit an itemized invoice and proof of delivery. Delivery proof may be a time-stamped, GPS-checked photo, a carrier confirmation, or a signed receipt, depending on the item. **The site does not accept uploads.** The dual-layer oracle in code is a **structured stub** (always fails until real OCR / EXIF / carrier checks are implemented).

Direct settlement. When both proofs pass, the escrow would be finished and the funds would go straight to the vendor. The charity would not touch the money. **Vendor payment is not live.**

5. System architecture

VeriChar has four layers, each with a single job. Only a thin slice of the application layer and none of the live ledger settlement layer are productized.

Ledger layer: the XRP Ledger

VeriChar plans to use the XRPL’s **native escrow** feature rather than custom smart-contract code, to keep the attack surface small. The XRPL is a decentralized blockchain [S1]. Consensus is designed so a new ledger version is typically produced in about **4-6 seconds** [S1]. Transaction costs are small amounts of XRP quoted in **drops** (1,000,000 drops = 1 XRP) [S9].

Escrow is a native payment type that locks XRP or fungible tokens until conditions are met [S17]. Types include time-based, conditional, and combination escrow [S17]. VeriChar’s design uses a **conditional escrow**: a cryptographic condition is the lock; the matching fulfillment releases the funds [S17].

Lifecycle

- **Create (EscrowCreate).** Defines amount, conditions, and recipient. A conditional escrow with no FinishAfter **must** have an expiry (CancelAfter) [S17]. Since fix1571, EscrowCreate **must** include a Condition or a FinishAfter [S20].
- **Finish (EscrowFinish).** Delivers funds to the recipient if conditions are met [S17]. For a conditional escrow, anyone can submit the finish, but only with the correct fulfillment [S17].
- **Expire and cancel (EscrowCancel).** After CancelAfter , the escrow can no longer be finished; anyone can cancel it, returning funds to the sender [S17]. An escrow with no CancelAfter never expires and cannot be canceled [S17].

Implications for VeriChar

- Every VeriChar escrow **must** carry an expiry, or failed delivery could lock funds permanently.
- The **fulfillment must be guarded**; whoever holds it can trigger release [S17].
- The only supported crypto-condition type is **PREIMAGE-SHA-256** [S17].
- Time fields follow ledger close times (roughly ± 5 seconds) and cannot be set in the past [S17].
- Smart Escrows (WebAssembly release rules) are not assumed [S20].

Costs of escrow. Create plus finish (or cancel) are two transactions. Fulfillment-bearing EscrowFinish costs more than ordinary payments: documentation describes a minimum on the order of **330 drops + 10 drops per 16 bytes of fulfillment** at a 10-drop reference fee, scaling with fee voting; multi-signing adds cost [S17]. While an escrow is

incomplete, the **sender** is responsible for its **owner reserve [S17]**. These costs must be measured on Testnet against real gift sizes before any minimum gift size is set.

Ledger facts that shape the design

- **Finality.** Count only **validated** results; require `tesSUCCESS` ; prefer `meta.delivered_amount` over `Amount` (partial-payment risk) **[S8][S15]**.
- **Reserves.** Accounts hold a base reserve; each owned object (including an incomplete escrow) adds an owner reserve. Values can change by validator vote **[S10]**. Who owns each escrow and who funds the reserve is an open product decision.
- **Accounts and keys.** Funding a valid address creates the account; the secret key controls it **[S4]**. No seed or private key is stored in the public site. API bearer tokens are stored only as hashes.
- **Test networks.** The backend allowlists official **Testnet** WebSocket hosts and refuses Mainnet URLs at startup.
- **Servers.** Production should prefer operated `xrpld` / Clio infrastructure; public servers can misreport state **[S6]**.
- **RLUSD.** Intended settlement asset where feasible. Issuer trust lines, reserves, clawback, and **whether RLUSD can be escrowed** (issuer `AllowTrustLineLocking` / `TokenEscrow`) remain **to be confirmed on Testnet** before building further **[S17][S22][S23]**.
- **Credentials.** Possible on-ledger marker for verified-charity status after off-ledger EIN review **[S19][S20]** (*to be confirmed*).

Verification layer: dual-layer oracle

An application “oracle” (not XRPL price-oracle objects) would run two checks that **both** must pass:

1. **Invoice (supplier log).** Itemized invoice matches the listed need (vendor, lines, price).
2. **Delivery (proof of delivery).** Time/location or carrier confirmation appropriate to the good.

Only then would the oracle use the guarded fulfillment to finish the escrow **[S17]**. **Current code:** stubs that refuse release until real verification is implemented.

Privacy layer

Donors may give as guests with no VeriChar account. An XRPL address is still required to sign a payment **[S4][S8]**. Tax-receipt email, if collected, would be stored off-chain, encrypted, never written to the ledger. XRPL transactions are public **[S15]**. Stronger “stealth” designs need careful engineering and legal review and are **not built**.

Application layer

Three interfaces are planned: donor experience, charity portal, and enterprise CSR / impact dashboard. Today: marketing site, simulation, enterprise **preview** (sample data only), and a thin Testnet API.

6. Cryptographic proof, explained

Concept	Plain meaning
Hash (digital fingerprint)	A fixed-size digest of data. Change one byte, the fingerprint changes. Useful for detecting tampering of uploaded evidence records.
Digital signature	Proves a transaction was authorized by a key holder without revealing the key.
Public ledger	After validation, the record is widely replicated and hard to rewrite unilaterally.
Crypto-condition / fulfillment	PREIMAGE-SHA-256 pair: the condition is published in the escrow; the secret fulfillment unlocks it [S17] .

Important limit. Cryptographic proof shows a record is authentic and unaltered. It **cannot** prove the underlying real-world event was honest. VeriChar is designed to make fraud **harder and easier to spot**, not impossible.

7. Donor, charity, and vendor experience

Donors (intended). Browse verified charities and itemized needs; fund one need; follow escrow state on a public explorer; optional receipt path. **Free.**

Charities (intended). Apply for verification; publish itemized needs; submit invoice and delivery evidence; never custody donor funds. **Free.** Registration is not open yet.

Vendors (intended). Named as escrow destination; receive funds only after approved proofs. **Free to participate** — no percentage taken from the payout as a platform settlement fee.

Enterprises (intended). Subscribe to SaaS tools for audit-ready impact / CSR reporting. Sample portal on the site is illustrative only, not certified, and not legal or tax advice.

8. Planned modules

Module	Role	Status
Charity verification	EIN / registry checks, domain linkage, optional Credentials	Design; email interest only
Need catalog	Itemized needs with vendor and price	Not live
Escrow service	Conditional XRPL escrow create / finish / cancel	Testnet CI passed for helper module; not productized on the public site
Dual-layer oracle	Invoice + delivery verification; fulfillment custody	Stub only
Donor UI	Fund need, track state	Simulation only
Enterprise SaaS	CSR / impact exports and dashboards	Portal preview (sample data)
Platform funding ledger	Transparent accounting for donations to VeriChar	Product principle; implementation TBD

9. Business model

Free for the giving network

Donors, charities, and vendors pay \$0 in platform settlement fees. There is **no 1% vendor fee** and no take-rate on escrow release for ordinary giving flows.

Network participants still face **XRPL network costs** (XRP fees and reserves) when the ledger path is live; those are protocol costs, not VeriChar revenue.

Revenue path A — Donate to VeriChar

People and organizations who want the transparency layer to exist can **donate to VeriChar itself**. As capabilities mature, platform funding should be as inspectable as charity funding: **track almost every dollar** of operating support

with public or exportable records. Supporting VeriChar is participation in the **charity transparency** effort, not a purchase of investment securities.

Revenue path B — Enterprise SaaS

Corporate and institutional customers can license tools for **audit-ready CSR and impact reporting** built on verified giving records (and, over time, richer analytics). This is voluntary commercial software, separate from free public giving rails.

What is explicitly rejected

- Percentage fees on vendor payouts as the default sustenance model.
- Opaque pooling of donor funds inside VeriChar for “admin” without traceability.

Exact pricing for enterprise tiers, donation instrument design, and on-ledger vs off-ledger presentation of platform finances remain **open product decisions**.

10. Security and risk

Risk	Mitigation direction
Stolen fulfillment	Hold fulfillment only in oracle secure store; never log; never ship to browsers
Malicious or compromised oracle	Dual proofs; audit logs; eventual multi-party or policy controls <i>(TBD)</i>
Fake invoices / staged delivery photos	Real OCR, vendor registries, EXIF/GPS fences, carrier APIs — not implemented
Malicious API or XRPL server	Testnet allowlist; prefer self-hosted infrastructure in production [S6]
Key compromise	No hot seeds in public site; hashed API tokens; hardware / custody plans for Mainnet
Regulatory	No Mainnet funds until legal review
Over-claiming	Prototype labeling on site and in this paper

Cryptography reduces certain classes of error; it does not eliminate human fraud.

11. Compliance considerations

- Charity verification may interact with IRS (or local) non-profit records; **no lookup is performed today**.
- Tax receipts, if offered, need jurisdiction-specific rules.
- Enterprise CSR exports are **not** certified financial statements.
- Token and escrow features depend on issuer flags and amendments; re-check before Mainnet.
- This paper is **not** legal advice. Counsel should review before pilot.

12. Roadmap

Phase	Focus
Now	Honest prototype: simulation; Testnet API foundation; automated XRPL Testnet escrow tests passing in CI ; free-for-all model in docs

Next	Expand Testnet coverage (fee/reserve measurement at gift sizes); RLUSD escrow feasibility; oracle spike with real sample documents
Pilot	Closed pilot after legal and security review; limited charities and vendors; still no broad Mainnet marketing of live giving
Public giving rails	Verified charities, itemized needs, escrow + dual proof, free for donors/charities/vendors
Platform sustainability	Transparent donate-to-VeriChar path; enterprise SaaS generally available

Dates are intentional goals, not guarantees.

13. Conclusion

VeriChar exists to answer a simple question with evidence: **where did the money go?**

The product answer is **proof before payment** on the XRP Ledger: itemized needs, conditional escrow, dual verification, and direct vendor settlement — with the charity never holding the cash.

The economic answer is **not** a take-rate on generosity. The rails are **free for donors, charities, and vendors**. The company is sustained by people who **donate to VeriChar** and by organizations that buy **enterprise transparency tools**.

Today that vision is a **prototype**. The honest path is Testnet proof, professional review, then careful pilot — without pretending the ledger layer is already live.

Appendix A. Sources

Prior drafts cited XRPL.org and related materials as **[S11]-[S24]**, including (non-exhaustive): consensus and networks; accounts and keys; transaction costs and finality; reserves; escrow concepts and transaction types; partial payments; domain verification / `xrp-ledger.toml` ; Credentials; amendments listings; RLUSD issuer documentation.

Readers should prefer the **current** XRPL.org pages over any frozen snapshot. Re-verify amendment status, reserve amounts, and RLUSD issuer flags before implementation milestones.

Appendix B. Open design questions

1. Which account owns each escrow and who funds owner reserves?
2. Can RLUSD be escrowed under current issuer settings on Testnet/Mainnet?
3. Guest wallet creation and funding without trapping donors in key management.
4. Oracle governance: single operator vs multi-party release.
5. How platform donations to VeriChar are recorded for public tracking.
6. Enterprise data retention, export formats, and certification boundaries.
7. Minimum practical gift size given escrow transaction costs.

Some features may change or may not be built. Statements about future plans are goals, not guarantees. Nothing in this document is an offer to sell or a solicitation to buy any asset or security. VeriChar does not currently process real donations.