

Waxseal: Sealed-Bid Tenders and Auctions with Zero-Knowledge Winner Proofs on Midnight

The Waxseal Project

waxseal.xyz · Version 1.3, September 2026

ABSTRACT

Sealed-bid tenders and auctions are the standard instrument for competitive procurement and for selling scarce goods, yet their integrity rests on a fragile assumption: that whoever opens the envelopes is honest, and that the envelopes cannot be opened early. We present Waxseal, a protocol and open implementation that removes this assumption using programmable privacy on the Midnight network. A bid never appears on chain. Each bidder publishes a cryptographic commitment together with a zero-knowledge proof that the hidden bid is valid: within the announced range, backed by a shielded deposit and, if required, made by an eligible bidder. The winner is not declared by anyone; it is proven. In *clock mode* a public price clock moves step by step after bidding closes and the first bidder whose sealed bid the clock reaches proves it, so no party ever sees any bid. In *committee mode* bids are additionally encrypted to a designated committee, with an in-circuit proof that the ciphertext and the commitment hold the same values; the committee decrypts off-chain and submits one proof that the chosen bid is the best against every commitment on chain. Losing bids are never revealed, losers obtain a private proof that they did not win, and an auditor can be granted access on chain. We state six security invariants, show how the protocol meets each one, discuss the residual assumptions, and describe the implementation deployed on the Midnight Preprod test network.

Keywords: sealed-bid auction, procurement, zero-knowledge proofs, commitment schemes, clock auction, threshold custody, Midnight, Compact.

1. Introduction

A sealed-bid tender asks every participant to submit a single, secret price before a deadline; the best price wins. The design is attractive because it is simple and because, in theory, it induces bidders to reveal their true valuation rather than react to rivals [1, 2]. In practice the guarantee is only as good as the custody of the envelopes. A procurement officer who can peek at a bid can leak it to a favoured supplier; a platform operator who stores bids in a database can be breached or compelled; and after opening, every losing price becomes public information that weakens

losers in the next round. Digital tender platforms have replaced paper envelopes with server-side secrecy, which merely moves the trusted party.

Public blockchains give tamper-evidence but not confidentiality: a bid written to a public ledger is public. Prior work on blockchain auctions therefore layers commitments, homomorphic encryption or multi-party computation on top of a transparent chain [5, 6]. These constructions are workable but expensive, and they usually still end with a reveal phase in which every bid is opened.

Waxseal takes a different route. It is built on Midnight, a Cardano partner chain whose smart contracts are written in the Compact language and executed as zero-knowledge circuits [7, 8]. A Compact circuit accepts private inputs (*witnesses*) that never leave the caller's machine; the chain receives only the public state transition and a proof that it was computed correctly. This lets us design a tender in which the ledger holds nothing but commitments and proofs, and in which the statement “bidder k has the best bid” is itself proven rather than asserted.

The contributions of this paper are:

1. A protocol for sealed-bid tenders in which bids are committed, validated and ranked without ever being revealed to the chain, the operator or, in clock mode, to anyone at all (Section 4).
2. Two winner-selection mechanisms sharing one core: a trustless price clock, and a committee mechanism with proof of correct encryption and an exactly- N winner proof, so that a committee that sees bids still cannot misreport or omit one (Sections 4.3 and 4.4).
3. A security model with six explicit invariants, an argument for each, and an honest account of what remains an assumption (Section 5).
4. An open implementation, deployed on the Midnight Preprod network, with contracts, an SDK, a web application and a committee tool (Section 6).

2. Background

2.1 Commitments and zero-knowledge proofs

A *commitment scheme* lets a party fix a value now and open it later [3]. It is *hiding* (the commitment reveals nothing about the value) and *binding* (the value cannot be changed afterwards). Waxseal uses a hash-based commitment over a bid, a random salt and a bidder identifier. A *zero-knowledge proof* convinces a verifier that a statement is true without revealing why [4]. Here the statements are of the form “I know an opening of commitment c whose bid lies in the range $[min, max]$ ” or “that bid is at most the current clock price”.

2.2 Midnight and Compact

Midnight is a blockchain whose contracts separate public ledger state from private witness data. A contract is written in Compact and compiled into circuits; calling a circuit produces a transaction carrying the new public state and a succinct proof [7, 8]. Values that must be published are marked explicitly with `disclose()`; anything else derived from a witness stays private by construction. The network also supports *shielded* tokens whose ownership and amounts are hidden, which Waxseal uses for deposits so that taking part in a tender is not visible in a bidder's wallet activity. Proofs are generated by a proof server that sees the witnesses;

Waxseal requires this server to be the bidder's own, local or bundled with the wallet, and never operates a shared one for real bids.

2.3 Clock auctions

A clock (Dutch or English) auction replaces sealed submission with a public price that moves monotonically until a bidder accepts it [2]. It reveals nothing about bids the clock never reaches. Waxseal combines both worlds: bids are sealed *before* the deadline, so nobody can react to them, and the clock is used only afterwards as a device for selecting the winner without opening the envelopes.

3. System Model and Goals

3.1 Parties

An *organiser* creates a tender. *Bidders* commit sealed bids. In committee mode a *committee* holds a decryption key. An *auditor* may be granted access after settlement. The *operator* (Waxseal) runs a public website and an API that mirrors public ledger state and stores public tender metadata; it holds no secrets. The *chain* and its *indexer* are public.

3.2 Adversaries

We consider a public observer who reads everything on chain and in the operator's systems; a rival bidder who can also submit and time transactions; a malicious organiser who sets parameters; a malicious committee; a malicious or compromised operator; a holder of a contract's maintenance key, who on Midnight can replace a deployed contract's verifier keys and so its rules; and a compromised software dependency in the web application. A compromised bidder machine and breaks of the underlying cryptography are out of scope.

3.3 Security invariants

The protocol is designed and tested against six invariants, which no release may weaken.

I1 (Binding). A bid cannot be changed after it is committed.

I2 (Correct winner). A published winner has the best bid against every commitment on chain, under the tender's rule.

I3 (Completeness). A committee cannot omit a commitment; the winner circuit consumes exactly the number of bids recorded on chain.

I4 (Unlinkable participation). Deposits and fees are shielded; participation is not inferable from wallet activity.

I5 (Losing bids stay sealed). A losing bid is never revealed on chain, in logs, in the indexer, in the API or in the user interface.

I6 (Witness locality). The bid and its salt never leave the bidder's machine.

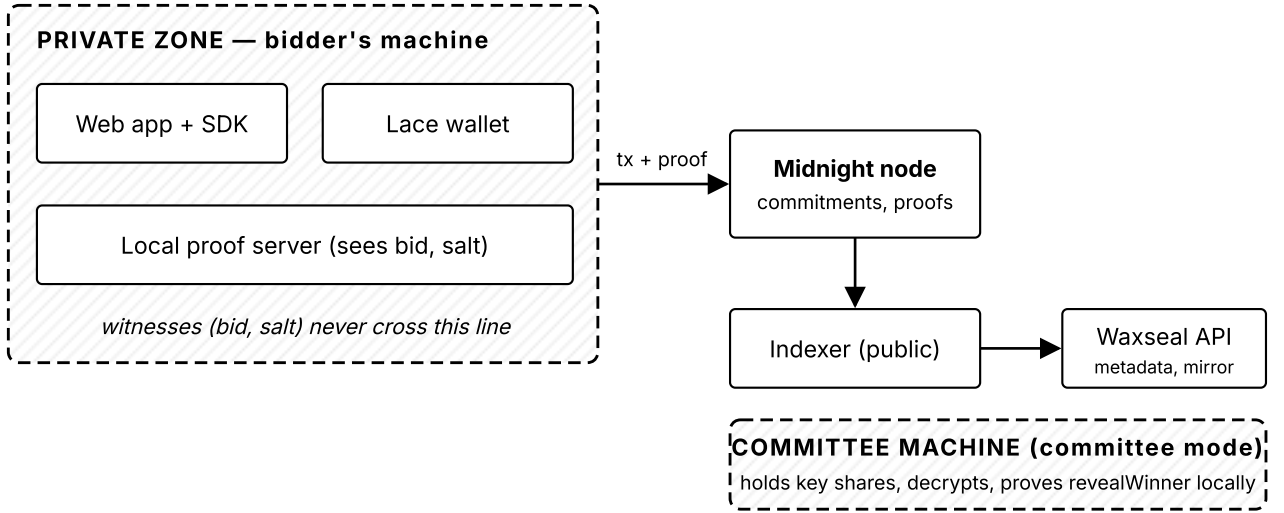


Figure 1. Trust zones. Secrets exist only in the two hatched, dashed regions. Every other component handles public parameters, commitments, ciphertexts and proofs.

4. Protocol

4.1 Tender creation

The organiser deploys a contract instance with public parameters: the valid bid range $[min, max]$, the close time T_c , the required deposit, the winner rule (lowest or highest bid wins), whether an eligibility list is enforced, and mode-specific parameters: the clock step s and round length ρ in clock mode, or the committee public key and a reveal deadline in committee mode. Human-readable metadata (title, description, document hashes) is kept off chain by the operator, but its canonical hash is stored in the contract, so it can neither be altered later nor attached to a contract by an impostor.

Eligibility, when enabled, is an append-only Merkle tree of keys. Each key is derived from a bidder secret and the organiser's identifier, so the same person is not linkable across organisers, and membership is proven in zero knowledge at commitment time.

4.2 Sealed commitment

A bidder holds a long-term secret σ and chooses a bid b and a fresh random salt r . The bidder identifier is scoped to the tender, and the commitment binds all three values under domain-separated hashes:

$$id = H(\text{"waxseal:bidder-id:v1"} \parallel \sigma \parallel tender) \quad (1)$$

$$c = H(\text{"waxseal:commitment:v1"} \parallel b \parallel r \parallel id) \quad (2)$$

The `commitBid` circuit takes (b, r, σ) as witnesses and proves, without disclosing them, that $min \leq b \leq max$, that the bidder is in the eligibility tree if one is enforced, and that a shielded coin of the required amount is locked in the contract's escrow under id . Only id and c are written to the ledger. One commitment per identifier per tender is accepted, and every later circuit that uses the bid recomputes (2) from the witness and asserts equality with the stored value; this is the mechanism behind I1.

In committee mode the same circuit additionally computes a ciphertext of (b, r) under the committee key (Section 4.4). Because ciphertext and commitment are computed *in the same circuit from the same witnesses*, the proof guarantees that they open to identical values; a bidder cannot commit to one bid and encrypt another.

4.3 Clock mode: winner selection without a trusted party

After T_c the contract enters the clock phase. Round $r \in \{0, \dots, R\}$ starts at $T_c + r \cdot \rho$ and has a public price

$$\begin{aligned} p(r) &= \min + \min(r \cdot s, \max - \min) \quad (\text{lowest wins}), \\ p(r) &= \max - \min(r \cdot s, \max - \min) \quad (\text{highest wins}). \end{aligned} \tag{3}$$

A bidder calls `claimAtPrice(r)` during round r . The circuit re-derives id and c from the witnesses, checks them against the ledger, and asserts two boolean facts, disclosing only their truth:

$$b \leq p(r) \quad \text{and} \quad \neg(b \leq p(r-1)) \quad (\text{lowest wins; mirrored for highest}). \tag{4}$$

The second conjunct is the key design decision. A bid may be claimed *only in its own round*, the first round whose price reaches it. Waiting for a better price is impossible, so the first valid claim necessarily comes from the best committed bid, unless that bidder stays silent. The first accepted claim settles the tender: the ledger records the winner's identifier, the round price and the round, and no further claims are accepted. The winning *price* is the clock price, not the bid itself; the bid remains sealed, and the result is within one step s of the best bid.

Silence is penalised rather than prevented. A loser refunds their deposit by proving, again in zero knowledge, that their bid was *not* reachable before the winning round. A better bidder who declined to claim cannot produce this proof and forfeits the deposit, which the organiser may collect after the refund deadline. Consequently the I2 guarantee in clock mode is economic: it holds when the deposit is meaningful. A zero-deposit clock tender gives no such guarantee, and the interface warns accordingly. Ties within a round are resolved by transaction order; same-round runners-up refund normally. If the clock runs out without a valid claim, anyone may mark the tender expired and all deposits become refundable.

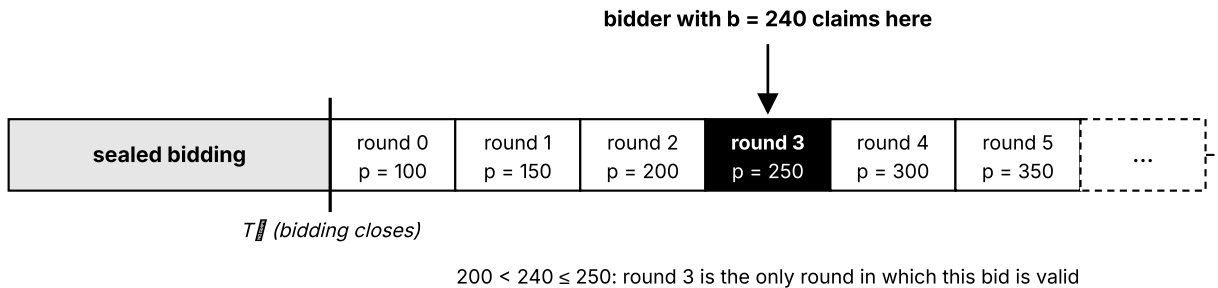


Figure 2. Clock mode with $\min = 100$, step $s = 50$, lowest bid wins. A sealed bid of 240 can be claimed only in round 3. Earlier claims fail the price test; later claims fail the own-round test.

4.4 Committee mode: verifiable evaluation by a designated party

Business procurement often requires an evaluation committee that must read the bids. Waxseal accommodates this without letting the committee control the outcome. At creation the organiser publishes a committee public key $K = k \cdot G$ on the Jubjub curve; the contract checks that K is not the identity and lies in the prime-order subgroup. At commitment the bidder samples an ephemeral scalar e and computes a hashed-ElGamal ciphertext

$$R = e \cdot G, \quad S = e \cdot K, \quad ct = (b + \kappa_1, r + \kappa_2), \quad \kappa_i = H(\text{"keystream"} \parallel S \parallel i) \text{ over the scalar field} \quad (5)$$

inside the same circuit that computes c , with $R \neq \text{identity}$ enforced. The ledger stores (id, c, R, ct) and an insertion index. The committee, holding k , recovers $S = k \cdot R$ and decrypts every bid off chain.

After the close the committee calls `revealWinner` with all openings (b_i, r_i, id_i) for $i < n$ and an index k as witnesses. The circuit is fixed at $N = 16$ slots and walks the on-chain insertion order: for every $i < n$ it asserts that the opening belongs to the i -th recorded identifier and hashes to the i -th commitment, and that

$$b_k < b_i \text{ or } (b_k = b_i \text{ and } k \leq i) \text{ (lowest wins; mirrored for highest).} \quad (6)$$

Only the winner's identifier and winning bid are disclosed. Because the loop is bound to the ledger's own count and order, the committee cannot skip, duplicate or reorder a commitment (I3), and because the openings must match the commitments produced with the proof of correct encryption, the committee cannot substitute a value it did not decrypt (I1, I2). Ties resolve deterministically to the earliest bid. If the committee fails to reveal before the deadline, the tender expires and every deposit is refundable; the organiser can collect only the winner's bond, never a loser's deposit.

The committee key may be split with Shamir secret sharing [9] into n shares with threshold t , so that no single member can read bids alone. Reconstruction happens on one machine at reveal time; this is threshold *custody*, not threshold decryption, and we state it as such.

4.5 Settlement and disclosure

Settlement makes exactly two facts public: the winner's identifier and the winning price. Each loser can generate a private "you did not win" proof from their own witnesses; it discloses a single boolean and nothing about other bidders. Losers' deposits are refunded from shielded escrow; the winner's deposit remains locked as a performance bond for the organiser. In committee mode the organiser may record an auditor grant on chain, after which the committee can re-encrypt the openings to the auditor's key. Grants are public, so disclosure to third parties is itself auditable.

Table 1. Comparison of the two winner-selection modes.

Property	Clock mode	Committee mode
Who can read bids	Nobody	The committee (key optionally split <i>t-of-n</i>)
Trust assumption	None; deposit makes silence costly	Committee confidentiality; result integrity is proven
Published price	Clock price (within one step of the best bid)	Exact winning bid
Winner proof	Own-commitment claim in own round	One proof over all $n \leq 16$ commitments
Bidder actions after close	Claim in own round; refund	None; refund if expired
Suited to	Auctions, price-only procurement	Evaluated procurement with a named panel

5. Security Analysis

We argue each invariant informally, name the mechanism that enforces it and the residual assumption, if any. Table 2 lists how each is tested in the implementation.

I1, binding. Commitment (2) is a collision-resistant hash over bid, salt and identifier. The ledger accepts one commitment per identifier per tender, and every bid-consuming circuit recomputes and compares the hash. Changing b or r after the fact requires a hash collision; using another bidder's opening fails because id is derived from the caller's own secret.

I2, correct winner. In committee mode the comparison (6) is asserted for every recorded commitment, so a false winner cannot be proven. In clock mode the own-round rule ensures that the first valid claim is the best bid among bidders who act; the deposit forfeit makes not acting costly. The residual assumption is economic and stated in Section 4.3.

I3, completeness. The reveal circuit iterates to the ledger's own counter over the ledger's own index, not over a witness-supplied list. There is no execution path in which a recorded commitment is left out.

I4, unlinkable participation. Deposits are shielded coins locked and refunded through the contract's escrow; amounts and owners are not visible in the public transaction. Two limits apply. A contract-held coin is public ledger state and the standard library derives a refund's output nonce from it, so a party who already knows a bidder's shielded address can test the refund transaction against it and learn that the bidder took part, though never what they bid; the next contract release builds refund outputs with a witness-derived nonce to close this. And transaction fees on Midnight are paid in a resource generated from a registered stake; whether fee payment can link a transaction to a bidder's public address is still under investigation. We

therefore state I4 against the public, and treat unlinkability against a party holding a bidder's address as an assumption for now.

I5, losing bids stay sealed. Every `disclose()` in the contracts is enumerated in review; the only disclosed bid value is the committee-mode winning bid. The API's schema has no column that could hold a bid or salt and its handlers reject any request body containing such keys. A continuous-integration check fails on any log or analytics statement that mentions these values. The refund proof in clock mode leaks one bit by design: an unrefunded deposit suggests its bid was better than the price of the round preceding the winner's.

I6, witness locality. All witness-taking SDK functions live in a module named `private/` and are never imported by server code. Proofs are generated on the user's machine: by the wallet, by the prover compiled to WebAssembly in the browser, or by a proof server on loopback. The web application places no private value in a URL, cookie, storage key or analytics event; the local bid vault is encrypted at rest under a user passphrase.

Committee assumptions. The committee can read every bid as soon as it is committed and could leak one to a favoured late bidder, or let the tender expire rather than reveal. Neither can be prevented by code; the protocol limits the damage (the result cannot be falsified, expiry refunds everyone) and reduces the risk of a single insider through key splitting. Because a committee tender holds at most 16 bids and the committee contract has no eligibility list yet, and non-winner deposits are refunded, the slots could be filled with worthless bids at the cost of fees; real procurement should run with invited suppliers until the allow list is added, and use a fresh committee key per tender, since ciphertexts stay on chain.

Immutable rules. A Midnight contract is deployed with a maintenance authority that can replace its verifier keys, and with them what the contract accepts. The default deployment gives the deployer a single key, which would let an organiser change the rules of a running tender. Every Waxseal tender therefore renounces its authority in the block after deployment: a signed maintenance update installs an authority with no keys and a threshold of one, which no one can ever satisfy. The renunciation is itself on the record, and a verifier can check that the verifier keys stored on chain equal those of a published release and never changed. An end-to-end test confirms that the ledger rejects a later update signed with the former key.

Independent verification. None of the arguments above should rest on the operator. A verifier, available in the browser and as a command-line tool, rebuilds a tender's history from a public indexer and checks the release, the renunciation, the deadline of each commitment, the result against the rules and the deposit book-keeping, and emits a report with a digest. The commitment format is specified byte for byte and published with test vectors; the compiled circuits, the SDK and a second implementation in standard-library Python agree on them, and deliberately wrong implementations are shown to fail. The residual trust is in the indexer read, which a disputing party can replace with its own node. Finally, shortly after each transaction the tender's state (its commitments, phase, result and the hash of the recording block) is time-stamped under RFC 3161 [11] by two independent public authorities. Anyone can rebuild the statement from the chain, so each time stamp shows, without trusting the network's clocks or Waxseal, that this state and this block existed no later than the signed time; the operator can omit a time stamp but cannot forge or backdate one, and anyone can obtain their own.

Parameter safety. All deadlines are bounded (round length 30 s to 7 days, at most 100,000 rounds, refund window 1 hour to 1 year) so that no timestamp arithmetic can overflow and lock deposits permanently.

Table 2. Invariants and test levels. C: circuit test on the compiled contract; Z: proof-level check on the compiled circuit; E: end-to-end on a local devnet; S: static or CI check; A: API integration test.

Invariant	Enforced by	Tested at
I1 Binding	Hash commitment recomputed in every circuit; one per bidder	C, Z
I2 Correct winner	Own-round claim + forfeit; exactly- N comparison	C, Z, E
I3 Completeness	Loop bound to ledger counter and insertion index	C
I4 Unlinkable participation	Shielded escrow; per-tender identifiers	E (fee linkability open)
I5 Losing bids sealed	Reviewed disclosures; API guard; log lint	C, Z, S, A, E
I6 Witness locality	private/ module boundary; local proving	S, E
Immutable rules	Maintenance authority renounced at creation; release fingerprints	E, S

6. Implementation

Waxseal is an open monorepo. The contracts are written in Compact (compiler 0.31.1, the version supported by the Preprod and Mainnet networks at the time of writing) and comprise two instances sharing one core: a clock-mode contract with circuits `commitBid`, `claimAtPrice`, `claimRefund`, `proveNotWinner`, `expire` and `ownerCollect`, and a committee-mode contract adding `revealWinner`, `grantAudit` and `ownerCollectBond`. Both are covered by adversarial circuit tests, more than seventy in total, that exercise every assertion, including wrong openings, out-of-order committee witnesses, ties and boundary rounds.

A TypeScript SDK wraps the compiled contracts, the wallet connector and the public indexer. It exposes `createTender`, `commitBid`, `claimAtPrice`, `claimRefund`, `revealWinner` and the read-only queries; every witness-taking function lives in `private/`. In the browser the SDK generates proofs on the user's machine, in the wallet (1AM or Lace) or in a Web Worker running the prover compiled to WebAssembly, and stores the bidder's sealed bids in an encrypted local vault. From that vault a bidder can check their commitment against the chain and export a receipt, or, deliberately, an opening that discloses their bid to a person of their choosing. The web application provides the public tender pages and the bidder and organiser flows; the API (Node, Fastify, PostgreSQL) stores public metadata bound to its on-chain hash, mirrors public ledger

state for fast pages and emits signed webhooks on tender events. The committee tool is a command-line application that runs entirely on a committee member's machine: it generates and splits keys, decrypts bids, produces the reveal proof with a local proof server, and exports and verifies auditor packages.

The system runs on the Midnight Preprod test network, where the first public clock-mode tender was created on 23 September 2026. A full committee-mode cycle (three encrypted bids, a 2-of-3 key reconstruction, reveal and audit export) has been executed end to end on a local network with real proofs. Each release's verifier-key fingerprints are recorded in an append-only manifest checked in continuous integration, and the verifier, the specification and the conformance vectors are served at waxseal.xyz/verify and waxseal.xyz/conformance. The API obtains the RFC 3161 time stamps from DigiCert and Sectigo and verifies each one before publishing it; the verifier re-checks them against the chain. Mainnet deployment is locked in code until an independent audit is complete.

7. Limitations and Future Work

- **Clock granularity.** The clock reveals the winner's bid to within one step and publishes the round price rather than the bid. Finer steps cost more rounds and time.
- **Committee capacity.** The reveal circuit is fixed at 16 bids. Larger tenders will be batched, with a second-level proof combining batch winners; a 32-slot circuit is planned.
- **Threshold decryption.** Key shares are currently reconstructed on one machine. Distributed decryption without reconstruction would remove that single point of exposure.
- **Fee linkability.** Whether fee payment can link a transaction to a bidder's public address is an open research item (Section 5, I4).
- **Front-running and censorship.** A clock claim cannot be copied, since it is bound to the claimant's own commitment, but a block producer could delay it past the round; analysis and mitigations are future work.
- **Multi-criteria evaluation.** Committee mode currently proves the best *price*. Proving a weighted score over several sealed criteria is a natural extension.
- **Qualified time.** The RFC 3161 time stamps come from free, non-qualified authorities. Under eIDAS only a qualified time stamp carries a presumption of accurate time; adding a qualified provider for EU public procurement is future work, as is revocation checking in the built-in verifier.
- **Round length.** A claim must be proven and included inside its round. The web app enforces rounds of at least ten minutes; the contract's own minimum is lower and will be raised, with a one-round grace, in the next release.
- **Audit.** The contracts have been internally reviewed three times but not externally audited.

8. Conclusion

Sealed-bid tenders have always required someone to be trusted with the envelopes. Waxseal shows that, on a chain with programmable privacy, this trust can be replaced by proof: bids are committed and validated without being seen, the winner is proven against every commitment, and losing bids are never opened. Where a human committee is genuinely required, the protocol

confines its power to reading bids and takes away its ability to misreport or omit. The implementation is public, tested and running on a test network, and we invite scrutiny of both the protocol and the code.

References

- [1] W. Vickrey, “Counterspeculation, Auctions, and Competitive Sealed Tenders,” *The Journal of Finance*, vol. 16, no. 1, pp. 8–37, 1961.
- [2] P. Milgrom, *Putting Auction Theory to Work*. Cambridge University Press, 2004.
- [3] M. Blum, “Coin Flipping by Telephone: A Protocol for Solving Impossible Problems,” *ACM SIGACT News*, vol. 15, no. 1, pp. 23–27, 1983.
- [4] S. Goldwasser, S. Micali and C. Rackoff, “The Knowledge Complexity of Interactive Proof Systems,” *SIAM Journal on Computing*, vol. 18, no. 1, pp. 186–208, 1989.
- [5] E.-O. Blass and F. Kerschbaum, “Strain: A Secure Auction for Blockchains,” in *ESORICS 2018*, LNCS 11098, Springer, 2018.
- [6] H. S. Galal and A. M. Youssef, “Verifiable Sealed-Bid Auction on the Ethereum Blockchain,” in *Financial Cryptography and Data Security Workshops 2018*, LNCS 10958, Springer, 2019.
- [7] Midnight Network, “Midnight Developer Documentation.” <https://docs.midnight.network>
- [8] Midnight Network, “Compact Language Reference.” <https://docs.midnight.network/develop/reference/compact/>
- [9] A. Shamir, “How to Share a Secret,” *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.
- [10] T. ElGamal, “A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms,” *IEEE Transactions on Information Theory*, vol. 31, no. 4, pp. 469–472, 1985.
- [11] C. Adams, P. Cain, D. Pinkas and R. Zuccherato, “Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP),” RFC 3161, IETF, 2001.

Waxseal whitepaper v1.3, September 2026. This document describes software under development on a test network; it is not investment advice and does not describe a token. Contracts, SDK and this text are published at waxseal.xyz.