

The SolBazaar Network: A Decentralized, Uncensorable Marketplace with On-Chain Escrow, Reputation, and Order Settlement

Lukas Saul* Robert Gludo† Jonathan Gludo‡ Ian Spiro§

September 28, 2026

Abstract

SolBazaar (or Solzaar) is a Solana-native decentralized marketplace protocol. It allows merchants to register storefronts, list products, and settle orders through an external escrow program that enforces deposit, completion, and refund conditions on-chain. Reputation is accrued through buyer-submitted reviews gated by escrow completion, and order state is synchronized with product inventory through deterministic, escrow-driven transitions. This whitepaper describes the protocol’s membership model, product lifecycle, order settlement, reputation accrual, and dispute-handling interfaces. It situates the design within a longer literary and technical lineage that runs from science fiction’s imagined anarcho-capitalist markets through OpenBazaar’s empirical disappointments to a contract-layer response.

1 Introduction: From the Micro Era to the Bazaar

The dream of a marketplace without a central operator is older than the blockchain and has appeared in literature. In Zheng Zhaocai’s *Micro Era: San Francisco* (2026), a post-scarcity San Francisco hosts a barter market where exchange is governed not by price but by “value resonance”—items carry life stories, and their exchangeability depends on the narratives they embody. The novel’s tension is not material scarcity but the contest over meaning and control: when survival is solved, ownership becomes a question of identity rather than utility.

Bruce Sterling’s *Bicycle Repairman* (1996) made this speculation into a functioning anarchist commune. Its protagonist repairs bicycles in a free zone that maintains its autonomy through voluntary exchange, reputation, and parallel institutions. Sterling’s novelette, which won the 1997 Hugo Award for Best Novelette, treats the decentralized market not as a utopian endpoint but as an ongoing coordination problem: trust must be produced, disputes must be resolved, and central authority must be kept at arm’s length without collapsing into disorder.

These and other literary antecedents frame the central question that OpenBazaar attempted to answer in software. Launched in 2014 as “DarkMarket” at a Toronto Bitcoin hackathon and renamed by Brian Hoffman, OpenBazaar was a peer-to-peer marketplace built on Bitcoin multisignature

*合肥工业大学, Vermont Secure Computing

†Vermont Secure Computing

‡Vermont Secure Computing

§Vermont Secure Computing

escrow: buyer, seller, and a chosen arbiter each held a key, and two of three signatures released funds. It raised \$1 million in seed funding from Andreessen Horowitz and Union Square Ventures and launched its first full release in April 2016. The promise was explicit: “The Internet has decentralized communications, and Bitcoin has decentralized money, but online commerce still relies on numerous intermediaries. OpenBazaar removes the intermediaries and gives everyone ... the ability to directly engage in trade with anyone in the world, for free.”

The empirical outcome was sobering. A 14-month study by Arps and Christin spanning 2018–2019 found that despite 24,000+ product listings and 1,000+ sellers, fewer than 200 sellers had any documented sale, and total observable sales amounted to approximately \$86,000—roughly \$6,000 per month. By comparison, Silk Road generated \$10–15 million per month in 2013. The researchers attributed the failure to two factors: the built-in search engine filtered illegal items, forcing users to third-party tools, and the peer-to-peer architecture required dedicated software rather than a browser. More than two-thirds of users stayed less than a day.

SolBazaar is a contract-layer response to that diagnosis. It does not attempt to solve the user-experience problem that OpenBazaar could not overcome. Instead, it addresses the settlement and trust problems that OpenBazaar left to manual arbitration and off-chain reputation. The protocol treats escrow as an external program with a deterministic state machine, reputation as a protocol-native account updated only on verified completion, and inventory as a resource synchronized with escrow-driven order transitions. The result is a marketplace where the hard parts of exchange—custody, completion, refund, and review—are enforced by program logic rather than by intermediary discretion.

2 Protocol Overview

SolBazaar allows any participant to:

1. **Register as a merchant**—creating a storefront profile with a name, description URI, logo, banner, shipping origin, preferred contact, and a seller security deposit in basis points.
2. **List a product**—with a title, description URI, up to three image URIs, a category, a price, and a stock quantity.
3. **Update or delete a product**—modifying metadata, price, or stock, or soft-deleting the listing.
4. **Create an order record**—after an external escrow account has been funded by the buyer with the product price plus a buyer security deposit, and by the seller with the configured seller deposit.
5. **Send a message**—attaching a chat message to an escrow account, visible only to the two escrow parties.
6. **Submit a review**—once the escrow reaches completed status, the buyer may rate the merchant from 1 to 5 and leave a comment.
7. **Record a completed sale**—incrementing product and merchant sales counters when escrow status confirms completion.
8. **Restore cancelled order stock**—returning inventory to the product when escrow status confirms cancellation or refund.

The protocol is implemented as a single Anchor program on Solana. It does not hold user funds directly; all custody is delegated to an external escrow program whose account data is decoded and validated on every state transition.

3 Merchant Identity and Sybil Resistance

SolBazaar does not gate participation through invitation or stake. Any wallet may register a merchant profile by calling `create_merchant`. The profile is a PDA derived from `[b"merchant", authority]`, ensuring one profile per authority. The profile stores a `store_name` (max 64 bytes), a `description_uri` (max 200 bytes), logo and banner URIs (max 200 bytes each), a `ships_from` field (max 64 bytes), a `preferred_contact` field (max 300 bytes), a `seller_deposit_bps` value (0–10,000), and an `active` flag. A `verified` flag is reserved for future attestation but is not currently set by any instruction.

Sybil resistance in SolBazaar is economic rather than cryptographic. A merchant’s ability to attract orders depends on reputation, which is earned only through completed escrows and verified buyer reviews. A merchant who abandons orders or fails to deliver accumulates no positive reputation and loses the seller deposit escrowed per order. The cost of creating many low-quality storefronts is therefore the opportunity cost of failed escrows, not a registration fee.

4 Product Lifecycle

4.1 Creation

Any active merchant may create a product by calling `create_product`. The product must satisfy:

- **Title:** at most 64 bytes.
- **Description URI:** at most 200 bytes.
- **Images:** at most 3 URIs, each non-empty and at most 250 bytes.
- **Category:** at most 32 bytes.
- **Price:** greater than 0 lamports.
- **Stock:** a u32 quantity.

The product PDA is derived from `[b"product", authority, product_id]`. The merchant must be active at the time of creation. The product records `created_at` and `updated_at` timestamps, a `sold` counter initialized to 0, and `active` and `deleted` flags both initialized to false and false respectively.

4.2 Update

An active merchant may update a product’s metadata, price, stock, or active status via `update_product`. The product must not be soft-deleted. The `updated_at` timestamp is refreshed on every update. The `sold` counter is not modifiable by this instruction; it is updated only through escrow-confirmed settlement.

4.3 Deletion

A merchant may soft-delete a product via `delete_product`. The product’s `active` flag is set to false and `deleted` to true. The account is not closed, preserving order history references. Once deleted, the product cannot be updated or ordered.

5 Order Lifecycle and Escrow Interface

5.1 External Escrow Account

SolBazaar does not implement escrow internally. Instead, it interacts with an external escrow program identified by `ESCROW_PROGRAM_ID`. Escrow accounts are owned by that program and are deserialized by SolBazaar using a fixed layout. The discriminator `ESCROW_ACCOUNT_DISCRIMINATOR` is checked before any field is read. The decoded structure includes:

- `creator, party_a, party_b`: Pubkey
- `escrow_type, status`: u8
- `required_deposit_a, required_deposit_b, deposited_a, deposited_b, proposed_payout_a, proposed_payout_b, reference_amount, proposed_donation`: u64
- `finalization_proposer`: Pubkey
- `finalization_note, note`: bounded strings (max 200 bytes)
- `vault`: Pubkey
- `created_at, deposit_at, finalized_at`: i64

SolBazaar requires that the escrow's `note` field contains the substring `"marketplace":"solbazaar"`. This binds the escrow to the SolBazaar protocol and prevents escrows created for other purposes from being used to fabricate orders. It is a namespace tag, not a cryptographic proof, but it is sufficient to prevent accidental cross-protocol reuse.

5.2 Order Creation

A buyer creates an order record by calling `create_order_record` with a `quantity`. The instruction:

1. Verifies the product is active, not deleted, and has sufficient stock.
2. Computes `total_price = price × quantity`.
3. Verifies the escrow account is owned by `ESCROW_PROGRAM_ID` and carries the correct discriminator.
4. Verifies the escrow `note` contains the SolBazaar tag.
5. Verifies `party_a = creator = buyer`.
6. Verifies `party_b = product.merchant`.
7. Verifies `party_a ≠ party_b` and neither is the default pubkey.
8. Verifies `status = 0` (initial state).
9. Computes the seller security deposit as `total_price × seller_deposit_bps/10,000`, with a minimum of 1 lamport.
10. Verifies the escrow's `reference_amount` equals `total_price`.
11. Verifies `required_deposit_a = total_price + security_deposit`.
12. Verifies `deposited_a = required_deposit_a`.
13. Verifies `required_deposit_b = security_deposit`.
14. Decrements product stock by `quantity` and refreshes `updated_at`.
15. Creates an `OrderRecord` PDA derived from `[b"order", escrow]`, storing the escrow, product, buyer, seller, quantity, total price, and timestamp.

The order record is the canonical binding between an escrow account and a SolBazaar transaction. Its PDA derivation from the escrow key ensures that each escrow can back at most one order record.

5.3 Completion

When the escrow reaches status 3 (completed), the buyer may call `record_completed_sale`. The instruction verifies:

- Escrow owner and discriminator.
- SolBazaar tag in the escrow `note`.
- `status = 3`.
- `party_a = buyer`.
- `party_b = order_record.seller`.

It then increments `product.sold` and `merchant_profile.total_sold` by the order quantity, refreshes `updated_at`, and creates a `CompletedSale` PDA derived from `[b"completed-sale", order_record]`. This account serves as a permanent, non-forgable receipt of the completed transaction.

5.4 Cancellation and Stock Restoration

When the escrow reaches status 4 (cancelled or refunded), any signer may call `restore_cancelled_order_stock`. The instruction verifies the escrow owner, discriminator, SolBazaar tag, status 4, and that the escrow parties match the order record's buyer and seller. It then increments `product.stock` by the order quantity and creates a `StockRestoration` PDA derived from `[b"stock-restoration", order_record]`. This prevents double-restoration: the PDA can only be initialized once per order record.

6 Messaging

SolBazaar provides an on-chain messaging primitive for order-scoped communication. A party to an escrow may call `send_message` with a `message_id` and a message of at most 280 bytes. The instruction verifies:

- Escrow owner and discriminator.
- SolBazaar tag in the escrow `note`.
- Escrow parties are non-default and distinct.
- The sender is either `party_a` or `party_b`.

It then creates a `ChatMessage` PDA derived from `[b"message", escrow, message_id]`. Messages are public on-chain but semantically scoped to the escrow's two parties. This is not private communication; it is a coordination log tied to a specific transaction.

7 Reputation

7.1 Initialization

A merchant's reputation account is initialized separately from the merchant profile by calling `initialize_reputation`. The reputation PDA is derived from `[b"reputation", merchant_profile.authority]` and stores counters for total reviews, total rating sum, and counts for each star level from 1 to 5.

7.2 Review Submission

A buyer may submit a review by calling `submit_review` with a rating from 1 to 5 and a comment of at most 280 bytes. The instruction performs a layered set of checks:

1. Escrow owner, discriminator, and SolBazaar tag.
2. `status = 3` (completed).
3. `party_a = reviewer` (only the buyer may review).
4. `party_b = merchant_profile.authority` (the merchant must be the escrow seller).
5. The order record matches the reviewer, merchant, and product.

It then creates a `MerchantReview` PDA derived from `[b"review", order_record]`, updates the reputation counters, and emits the review on-chain. The PDA derivation ensures one review per order record, preventing review spam.

7.3 Reputation Aggregation

Reputation is stored as raw counters, not as a computed average. This choice allows any client to compute the average rating as `total_rating/total_reviews` and to inspect the distribution of star levels. The protocol does not weight reviews by purchase size, buyer reputation, or time. It is a simple, deterministic aggregation of verified buyer feedback.

8 Security Properties

The protocol’s security guarantees arise from its escrow interface, PDA-derived state, and layered validation. Table 1 summarizes the principal properties and the mechanism that enforces each.

Property	Mechanism
Escrow binding	Escrow <code>note</code> must contain the SolBazaar tag; owner and discriminator checked on every interaction.
Party authentication	Escrow <code>party_a</code> and <code>party_b</code> must match the order record and the signer’s role.
Order uniqueness	Order record PDA derived from escrow key; one order per escrow.
Review uniqueness	Review PDA derived from order record; one review per completed order.
Stock consistency	Stock decremented on order creation; restored only on escrow-confirmed cancellation.
Receipt integrity	Completed sale and stock restoration PDAs are created only once per order record.
Merchant authorization	Product updates and deletions require the merchant’s authority signature and PDA derivation.
Text bounds	All strings bounded by explicit length checks at instruction entry.

Table 1: Security properties and their enforcing mechanisms.

9 Parameters

Table 2 lists the protocol’s principal constants and limits.

Parameter	Value	Notes
Store name max length	64 bytes	
Description URI max length	200 bytes	
Logo/banner URI max length	200 bytes	
Ships-from max length	64 bytes	
Preferred contact max length	300 bytes	
Product title max length	64 bytes	
Product images max count	3	Each max 250 bytes.
Product category max length	32 bytes	
Message max length	280 bytes	
Review comment max length	280 bytes	
Rating range	1–5	
Seller deposit range	0–10,000 bps	0–100%.
Minimum seller deposit	1 lamport	If computed value is 0.
Escrow program ID	J6oe...KN5A	External escrow.
Escrow discriminator	[31,213,123,...]	8 bytes.

Table 2: Protocol parameters.

10 Conclusion

SolBazaar is a minimal, self-contained marketplace protocol. It does not attempt to solve the discovery, privacy, or user-experience problems that limited OpenBazaar’s adoption. Instead, it addresses the settlement and trust problems that OpenBazaar left to manual arbitration and off-chain reputation. By delegating custody to an external escrow program, binding orders and reviews to PDA-derived identities, and synchronizing inventory with escrow-driven state transitions, the protocol provides a verifiable, on-chain mechanism for decentralized commerce.

The literary lineage from Zheng’s post-scarcity barter to Sterling’s anarchist commune to OpenBazaar’s empirical disappointments suggests that the hard problem of uncensorable markets is not cryptographic capacity but coordination under uncertainty. SolBazaar’s contribution is to make the coordination rules explicit and enforceable, leaving the harder problems of adoption and usability to the layers above.

References

- [1] Zheng Zhaocai. *Micro Era: San Francisco*. QQ Reading, 2026.
- [2] Bruce Sterling. Bicycle Repairman. In *Intersections: The Sycamore Hill Anthology*. Tor Books, 1996. Hugo Award for Best Novelette, 1997.
- [3] James Arps and Nicolas Christin. Open Market or Ghost Town? The Curious Case of OpenBazaar. In *Financial Cryptography and Data Security*. Carnegie Mellon University CyLab, 2020.
- [4] Nasdaq. OpenBazaar Released: Decentralized Bitcoin Marketplace Now Live and Ready for Business. 2016.

- [5] Business Insider. The dream of an anonymous, uncensorable online marketplace is finally here. 2016.
- [6] Journal of Private Enterprise. Decentralized Marketplaces with Privately Enforced Contracts: A Case Study of OpenBazaar. Vol. 37, No. 4, 2022.

This whitepaper describes the SolBazaar program as deployed at the program ID declared in the source. It is not financial advice. Participants should review the source code and security policy before interacting with the protocol.