The Third Can of Paint: Fair Exchange Without a Trusted Third Party

Wings
Riff Labs
wings@riff.cc

January 2026

Abstract

We present a construction that achieves fair exchange between two parties without requiring a trusted third party (TTP). The classical result of Pagnia and Gärtner (1999) states that fair exchange is impossible without a TTP. We show this impossibility can be circumvented through *bilateral construction*: a shared artifact that emerges only when both parties contribute, and cannot exist otherwise. We call this the "third can of paint" construction.

Our concrete instantiation uses *bilateral Schnorr signatures*: a four-phase protocol (C $\rightarrow$ D $\rightarrow$ T $\rightarrow$ S) where the Schnorr challenge $e = H(R, P, m, \text{attack\_key})$ is bound to an *attack key* that can only be computed after both parties complete the bilateral T-exchange. Since neither party can compute the challenge without both T proofs, neither can compute their partial signature first. The signature *emerges* from bilateral completion—it cannot exist otherwise.

Our construction achieves: (1) no party "goes first"—partial signatures are impossible until bilateral completion; (2) symmetric outcomes under all channel conditions including total failure; (3) MuSig-style key aggregation preventing rogue-key attacks. Reference implementation in Rust with formal proofs in Lean 4 (zero `sorry` statements). This work extends the Two Generals Protocol (TGP), applying the same bilateral construction principle to fair exchange.

1 Introduction

1.1 The Fair Exchange Problem

Fair exchange is a fundamental problem in distributed systems and cryptographic protocols. Two parties, Alice and Bob, wish to exchange items such that either both receive what they want, or neither does. The canonical examples include:

- **Contract signing**: Neither party wants to be bound while the other is not
- **Atomic swaps**: Neither party wants to lose their cryptocurrency while the other keeps theirs
- **Certified email**: The sender wants proof of receipt; the receiver wants the content

1.2 The Classical Impossibility

Pagnia and Gärtner [1] proved that fair exchange is impossible without a trusted third party (TTP). Their argument is intuitive: someone must "go first," and whoever goes first can be cheated. Therefore, an external arbiter is required to hold both items and release them atomically.

The TTP requirement is problematic:

- **Single point of failure**: If the TTP fails, no exchange
- **Single point of trust**: All parties must trust the TTP
- **Single point of attack**: Compromising the TTP compromises all exchanges

1.3 Our Contribution

We show that the impossibility can be circumvented by *not having anyone go first*. Instead of Alice sending her item, then Bob sending his, we construct a protocol where the exchange item *emerges* from bilateral contribution.

The key insight: if neither party holds the exchange item until both have contributed, there is nothing to cheat with.

2 The Third Can of Paint

2.1 The Metaphor

Consider Alice with red paint and Bob with blue paint. They want to create purple paint together, but neither wants to give their paint first (the trust problem).

Traditional solution: A trusted third party holds both cans, mixes them, and distributes the purple paint.

Our solution: The purple paint is not "held" by anyone. It *emerges* when both colors are mixed. Neither party gives anything to a holder—they contribute to a shared construction.

The "third can" (the purple paint) has special properties:

1. It doesn't exist until both contribute

2. Neither party holds it alone
3. When it exists, both can access it
4. Mathematics serves as the "trusted" mixer

2.2 Formal Construction

We model the bilateral state with four boolean variables:

$$\begin{aligned} d_a &: \text{Alice's contribution arrived} \\ d_b &: \text{Bob's contribution arrived} \\ a\_responds &: \text{Alice responded to the bilateral state} \\ b\_responds &: \text{Bob responded to the bilateral state} \end{aligned}$$

The shared construct V emerges only when both contributions exist:

$$V\_emerges(d_a, d_b) = \begin{cases} \text{Some}(v) & \text{if } d_a \wedge d_b \\ \text{None} & \text{otherwise} \end{cases}$$

Each party's response requires V to exist:

$$alice\_response(V, d_b) = \begin{cases} \text{Some}(r_a) & \text{if } V = \text{Some}(\_) \wedge d_b \\ \text{None} & \text{otherwise} \end{cases}$$

$$bob\_response(V, d_a) = \begin{cases} \text{Some}(r_b) & \text{if } V = \text{Some}(\_) \wedge d_a \\ \text{None} & \text{otherwise} \end{cases}$$

The **third can** (shared artifact) emerges only when all three components exist:

$$third\_can(V, r_a, r_b) = \begin{cases} \text{Some}(artifact) & \text{if all three are Some} \\ \text{None} & \text{otherwise} \end{cases}$$

3 Main Results

Theorem 1 (No Unilateral Completion)**.** *Neither party alone can create the shared artifact:*

$$\begin{aligned} &\forall d_b, a\_responds, b\_responds: \quad make\_state(false, d_b, a\_responds, b\_responds) = None \\ &\forall d_a, a\_responds, b\_responds: \quad make\_state(d_a, false, a\_responds, b\_responds) = None \end{aligned}$$

Proof. By construction, $V\_emerges$ requires $d_a \wedge d_b$. If either is false, $V =$ None, and therefore $third\_can =$ None. □

Theorem 2 (Atomicity)**.** *The shared artifact exists if and only if all four conditions hold:*

$$(make\_state(d_a, d_b, a\_responds, b\_responds)).isSome \iff d_a \wedge d_b \wedge a\_responds \wedge b\_responds$$

Proof. Verified by exhaustive case analysis in Lean 4 (16 cases). □

Theorem 3 (Symmetric Outcomes)**.** *The outcome is always symmetric—both parties succeed or both fail:*

$$\forall d_a, d_b, a\_responds, b\_responds : \quad outcome \in \{BothSucceed, BothFail\}$$

Proof. The outcome type has only two constructors. By construction, there is no "AliceSucceeds_BobFails" or vice versa. □

Theorem 4 (Fair Exchange Without TTP)**.** *The bilateral construction satisfies the fair exchange specification:*

1. ***Fairness****: If the artifact exists, both contributed*
2. ***Atomicity****: Artifact exists iff full bilateral completion*
3. ***No TTP****: Computation is purely deterministic from inputs*
4. ***Termination****: Every input produces a definite outcome*

Proof. See `FairExchangeStandalone.lean` for the complete formal proof. □

4 Relation to the Two Generals Problem

This construction derives from the Two Generals Protocol (TGP) [2], which solves the coordinated attack problem through the same principle.

In TGP:

- d_a: Alice's double-proof (D_A) delivered to Bob
- d_b: Bob's double-proof (D_B) delivered to Alice
- The "attack key" is the third can—it exists only when both generals have completed the bilateral construction

The generals don't "decide" to attack. The attack capability *emerges* from their collaboration. If either fails to contribute, the capability doesn't exist, and both abort.

5 Concrete Instantiation: Bilateral Schnorr Signatures

5.1 The Irrevocability Problem

The abstract bilateral construction assumes the "shared artifact" doesn't exist until both parties contribute. But in practice, when Alice computes a partial Schnorr signature, that partial *exists*—it's irrevocable. If she sends it to Bob before knowing Bob will reciprocate, Bob could complete the signature alone. How do you create a signature protocol where neither party can compute their partial until bilateral completion?

5.2 The Key Insight: Bilateral Challenge Binding

In standard MuSig-style Schnorr multisignatures, each party computes:

$$s_i = k_i + e \cdot a_i \cdot x_i$$

where k_i is the nonce, e is the challenge, a_i is the key coefficient, and x_i is the private key. The challenge e is typically:

$$e = H(R, P, m)$$

where R is the combined nonce point, P is the aggregated public key, and m is the message.

The problem: Once Alice knows R and P (after nonce exchange), she can compute e and therefore her partial s_A. She could send s_A before Bob sends s_B, creating asymmetry.

Our solution: Bind the challenge to the *attack key*, which can only be computed after bilateral T-proof exchange:

$$e = H(R, P, m, \textbf{attack\_key})$$

where

$$\textbf{attack\_key} = H(T_A \| T_B)$$

Since the attack key requires *both* T_A and T_B, and T_B can only exist if Bob completed the bilateral D-exchange, **neither party can compute the challenge e until bilateral completion**.

No challenge ⇒ no partial signature ⇒ nothing to "go first" with.

5.3 The Four-Phase Protocol

The bilateral Schnorr construction uses four phases:

Phase	Artifacts	What It Achieves
C	C_A, C_B	Unilateral commitments: public keys and nonce points
D	D_A, D_B	Bilateral at C level: $D_X = \text{Sign}_X(C_X, C_Y)$
T	T_A, T_B	Bilateral at D level: $T_X = \text{Sign}_X(D_X, D_Y)$ — **THE KNOT**
S	S_A, S_B	Partial signatures with attack key binding

Phase C: Each party floods their commitment containing their public key P_i and nonce point $R_i = k_i \cdot G$. No bilateral state yet.

Phase D: Upon receiving the counterparty's C, each party constructs a signed double-proof embedding both commitments:

$$D_A = \text{Sign}_A(C_A \| C_B) \qquad D_B = \text{Sign}_B(C_B \| C_A)$$

D proves "I saw your commitment." D embeds the counterparty's C—bilateral at the C level.

Phase T: Upon receiving the counterparty's D, each party constructs a signed triple-proof embedding both D proofs:

$$T_A = \text{Sign}_A(D_A \| D_B) \qquad T_B = \text{Sign}_B(D_B \| D_A)$$

T embeds both D proofs—bilateral at the D level. This is **the knot**. Crucially: *no partial signatures have been computed yet.*

Phase S: Upon receiving the counterparty's T, each party:

1. Computes the attack key: attack_key $= H(T_A \| T_B)$
2. Computes the bilateral challenge: $e = H(R, P, m, \text{attack\_key})$
3. Computes their partial signature: $s_i = k_i + e \cdot a_i \cdot x_i$
4. Constructs S_i containing their T, the counterparty's T, and the partial

Completion: Upon receiving the counterparty's S, each party combines:

$$s = s_A + s_B \qquad R = R_A + R_B$$

The final signature (R, s) verifies against the aggregated public key P.

5.4 Why This Achieves Fair Exchange

The construction guarantees bilateral determination:

Theorem 5 (Bilateral Challenge Binding)**.** *Neither party can compute the Schnorr challenge e without possessing both T_A and T_B.*

Proof. The challenge is $e = H(R, P, m, \text{attack\_key})$ where attack_key $= H(T_A \| T_B)$. Without both T proofs, the attack key cannot be computed, and therefore e cannot be computed. □

Theorem 6 (No Unilateral Partial)**.** *Neither party can compute their partial signature s_i without the counterparty having completed the T-phase.*

Proof. $s_i = k_i + e \cdot a_i \cdot x_i$ requires knowing e. By the previous theorem, e requires both T proofs. T_B can only exist if Bob had D_A, which proves Bob completed the D-exchange. Therefore Alice cannot compute s_A unless Bob reached phase T. □

Theorem 7 (Symmetric Outcomes)**.** *Under any channel behavior (including total failure), the outcome is symmetric: both parties obtain the signature, or neither does.*

Proof. If T_B never reaches Alice: Alice cannot compute attack_key, cannot compute e, cannot compute s_A, cannot attack.

If T_A never reaches Bob: Bob cannot compute attack_key, cannot compute e, cannot compute s_B, cannot attack.

If S_B never reaches Alice: Alice has attack_key but not s_B, cannot combine, cannot complete.

If S_A never reaches Bob: Bob has attack_key but not s_A, cannot combine, cannot complete.

In all cases, failure is symmetric. □

5.5 MuSig-Style Key Aggregation

To prevent rogue-key attacks (where a malicious party chooses their public key as a function of the honest party's key), we use MuSig-style key coefficients:

$$L = H(\text{“MUSIG\_KEYSET”} \| P_A \| P_B)$$

$$a_i = H(\text{“MUSIG\_COEF”} \| L \| P_i)$$

The aggregated public key is:

$$P = a_A \cdot P_A + a_B \cdot P_B$$

Each party's partial signature uses their coefficient:

$$s_i = k_i + e \cdot a_i \cdot x_i$$

The combined signature verifies as:

$$s \cdot G \stackrel{?}{=} R + e \cdot P$$

5.6 The Attack Key IS the Consent

In TGP, the attack key answers: "Can we attack?"

In fair exchange, the attack key answers: "Did both parties consent?"

Same construction, different interpretation. The attack key is not just *evidence* of consent—it *is* consent, crystallized into a cryptographic binding that makes the signature possible.

The signature doesn't prove that consent happened. The signature *couldn't exist* without consent happening.

5.7 Comparison with Prior Approaches

Approach	Mechanism	Limitation
2-of-2 threshold	Both parties must combine simultaneously	Requires synchronous finalization
Adaptor signatures	Partial becomes valid when secret revealed	One party knows secret first
HTLCs	Hash locks with timeouts	Free option problem
Bilateral Schnorr	Challenge bound to bilateral state	**Neither can compute partial first**

5.8 Implementation

A reference implementation in Rust is available at `github.com/riff-labs/thirdcan` under AGPLv3. The implementation uses the secp256k1 curve (k256 crate) with SHA-256 for all hash functions.

Key implementation details:

- All D, T, S proofs carry real Schnorr signatures that are verified
- Canonical ordering ensures both parties compute identical attack keys
- Transcript consistency checks prevent proof substitution attacks
- The emergence proof $H(S_A \| S_B)$ is embedded in the final signature

6 Applications

6.1 Contract Signing

Alice and Bob want to sign a contract. Neither wants to be bound while the other is not.

Using the third can construction:

1. Alice sends her commitment (proof she will sign)

2. Bob sends his commitment (proof he will sign)
3. The signed contract *emerges* when both commitments exist
4. Neither holds a signed contract until both do

6.2 Atomic Swaps

Alice has BTC, Bob has ETH. They want to swap without a centralized exchange.

Hash Time-Locked Contracts (HTLCs) attempt this but have a critical flaw:

1. Alice creates secret s and hash $H(s)$—**Alice knows** s
2. Alice locks her BTC with hash lock $H(s)$
3. Bob locks his ETH with the same hash lock
4. Alice can reveal s anytime before timeout to claim ETH
5. Bob then uses s to claim BTC

The problem: Alice has *optionality*. She knows s from the start and can wait, watch the market, and decide whether to complete. If prices move against her, she lets the timelock expire. Bob's funds are locked the entire time. This "free option problem" makes HTLCs only *approximately* fair—and has been exploited in practice through MEV extraction and atomic swap griefing attacks [5].

True bilateral construction requires neither party to hold the unlock key until both have committed. In TGP terms: the attack key doesn't exist until $d_a \wedge d_b$—neither party "knows the secret" because the secret doesn't exist yet. A proper atomic swap would derive the unlock key from *both* parties' contributions, not from one party's pre-existing secret.

6.3 Escrow Without Escrow

Traditional escrow requires a trusted holder. The third can construction provides "escrow" without any holder:

- No party holds the funds/items alone
- The "escrow" is the bilateral construction itself
- Mathematics serves as the impartial arbiter

7 Discussion

7.1 What About the Impossibility Result?

Pagnia and Gärtner's impossibility assumes that one party must "go first"—that is, make an irrevocable commitment before the other. Our construction avoids this by having neither party make an irrevocable commitment. The commitments are *conditional* on the bilateral state, which doesn't exist until both contribute.

7.2 Relation to Gradual Release

Even, Goldreich, and Lempel [4] proposed *gradual release* protocols where parties incrementally reveal their items bit by bit. If one party stops early, both have approximately equal partial information. This achieves "approximate fairness" through incremental commitment.

Our approach differs fundamentally: rather than gradually releasing a pre-existing item, we construct an item that *doesn't exist* until both parties complete. There is no partial state—either the third can exists (both succeed) or it doesn't (both fail). This yields exact fairness rather than approximate fairness, at the cost of requiring the bilateral construction to complete.

7.3 The Role of the Channel

Our construction assumes a *fair-lossy* channel, defined precisely:

Definition 1 (Fair-Lossy Channel)**.** *A channel is* fair-lossy *if for every message m sent infinitely often, there exists a finite time t such that m is delivered by time t. Equivalently: persistent sending eventually succeeds. The channel may lose any finite prefix of transmissions, but cannot lose all transmissions of a message sent infinitely often.*

This is strictly weaker than reliable delivery (which guarantees every send succeeds) but strictly stronger than fully adversarial (which permits permanent partition). Fair-lossy captures realistic networks: packets drop, but persistent retransmission works.

Under fair-lossy, the bilateral construction completes: both parties flood continuously, so both contributions eventually arrive.

Under a fully adversarial channel (permanent partition), both parties timeout and abort—a symmetric outcome. The construction degrades gracefully: no asymmetric failures are possible regardless of channel behavior.

7.4 Mathematics as TTP

In our construction, mathematics serves the role traditionally played by a TTP:

- **Deterministic**: Same inputs always produce same outputs
- **Incorruptible**: Cannot be bribed or compromised
- **Available**: Always "online" (it's just computation)
- **Verifiable**: Anyone can check the construction

8 Conclusion

We have shown that fair exchange without a trusted third party is possible through bilateral construction. The key insight is that neither party "goes first"—the exchange item emerges from their collaboration.

The construction is simple, formally verified, and applicable to a wide range of problems including contract signing, atomic swaps, and escrow services.

The "third can of paint" is not held by Alice, not held by Bob, and not held by any third party. It emerges from mathematics itself.

References

[1] H. Pagnia and F. C. Gärtner, "On the Impossibility of Fair Exchange without a Trusted Third Party," Technical Report TUD-BS-1999-02, Darmstadt University of Technology, 1999.

[2] Wings, "The Two Generals Protocol: Deterministic Coordination Over Lossy Channels," Riff Labs Technical Report, 2026.

[3] N. Asokan, V. Shoup, and M. Waidner, "Optimistic Fair Exchange of Digital Signatures," EUROCRYPT 1998.

[4] S. Even, O. Goldreich, and A. Lempel, "A Randomized Protocol for Signing Contracts," Communications of the ACM, vol. 28, no. 6, pp. 637–647, 1985.

[5] P. Daian, S. Goldfeder, T. Kell, Y. Li, X. Zhao, I. Bentov, L. Breidenbach, and A. Juels, "Flash Boys 2.0: Frontrunning in Decentralized Exchanges, Miner Extractable Value, and Consensus Instability," IEEE Symposium on Security and Privacy, 2020.

[6] C. Komlo and I. Goldberg, "FROST: Flexible Round-Optimized Schnorr Threshold Signatures," Selected Areas in Cryptography (SAC), 2020.