

Silent Payments, Formally: Security, Scanning Costs, and Blind Collaborative Payments

Rong Qian¹, Yu Cheng¹, Mengrun Chen¹, Yuchang Zhang², and Zengli Guo³

¹ Beijing Electronic Science and Technology Institute, Beijing, China
rqian@besti.edu.cn, 2863394654@qq.com, 13015310107@163.com

² Graduate Department of North China Institute of Computer Systems Engineering,
Beijing, China zhangyuchang26@mails.ucas.ac.cn

³ Faculty of Information Science and Engineering, Ocean University of China,
Qingdao, China guozengli@stu.ouc.edu.cn

Abstract. Silent payments (BIP-352) let Bitcoin users publish one static address and receive at fresh, unlinkable outputs. The protocol already ships in major wallets, yet its own specification concedes it has no formal security proof—and 2026 brought a worst-case scanning attack, a funds-locking hardware-wallet disclosure, and an omitted-payment bug. We supply the missing analysis. Receiver and co-transactor unlinkability are proved under DDH, with the label system as a corollary that includes an executable linking attack and its fix. We then quantify what scanning costs and what an attacker can buy with it. The community’s $K_{\max}$ cap turns quadratic attack amplification into linear; every compliant scanner needs $\Omega(N+KM)$ work; our set-probing scanner meets this bound and runs $39\times$ faster in our measurements; and a year of fee data prices a full-block attack as low as 0.0015 BTC. For collaborative transactions we show what the specification’s warning was about (the transaction assembler can always identify the recipient’s output, and malformed partial secrets burn payer funds, the failure class disclosed in August 2026), and we construct the variant the warning called for: blind collaborative payments with information-theoretic payer privacy. Delegation, finally, offers no safe default: scanning servers either learn too much or can hide payments. A twelve-experiment benchmark suite accompanies the paper.

Keywords: Silent payments · BIP-352 · Bitcoin · Unlinkability · Scanning cost · CoinJoin · Light clients

1 Introduction

Bitcoin’s privacy tooling has contracted under regulatory pressure: the Samourai Whirlpool coordinators were convicted in 2025, Wasabi Wallet shut down its coordinator, and the surviving coinjoin infrastructure operates under explicit legal risk [42]. Against this backdrop, *silent payments* [20] have emerged as the deployment-ready privacy mechanism: a receiver publishes a single static address

Corresponding authors: Yu Cheng and Mengrun Chen.

$(B_{\text{scan}}, B_{\text{spend}})$, each sender derives a fresh Taproot output key from an ECDH shared secret computed over the transaction’s input keys, and no on-chain artifact distinguishes the output from any other Taproot spend. The protocol has been merged since 2024, its specification reached status Complete in 2026, and sending or receiving support ships in major wallets, including hardware wallets with dedicated DLEQ flows.

The specification itself is frank about the gap: it warns that collaborative transactions such as coinjoins are “not recommended . . . due to a lack of a formal security proof”; it defers light-client support to “an area of open research”; it asks how filter-based light clients affect “the security assumptions of BIP-158”; and it notes that MuSig/FROST inputs might require DLEQ proofs—“no security proof yet,” as a developer meeting minuted it [4]. In February 2026 a worst-case scanning attack (one transaction with 23,230 outputs all paying the same receiver) forced an emergency per-group recipient cap K_{max} [13,5]; in August 2026 a hardware-wallet vendor disclosed that a malicious host could lock silent-payment funds at unintended addresses, recoverable only with the attacker’s cooperation [10]; and a wallet vendor has shipped a bug in which server-side data omission silently hid incoming payments [29].

This paper. We give the deployed BIP-352 specification what is, to our knowledge, its first formal security analysis, covering the four surfaces the specification itself flags. Relative to prior formal treatments of stealth-address-style protocols, our object is Bitcoin-native and already deployed: zkBSA [52] formalizes Ethereum’s ERC-5564 with compliance machinery and does not consider BIP-352’s input-key aggregation, label system, or full-node scanning; the stealth-signature and publicly-derived-key lines [35,34,32] supply our definitional vocabulary. Table 3 summarizes the differences: none of the four features that drive our analysis (scan/spend key separation, the label system, receiver-side scan cost, collaborative settings) exists in the abstract template, which instead treats tracking data generated by the receiver; in silent payments the “tracking information” is adversarial chain data chosen by senders. That inversion shapes most of what follows.

1. **Definitions and proofs for the core protocol.** A five-tier adversary matrix (§2), game-based unlinkability for observers and co-transactors, and a DDH reduction with an explicit embedding lemma for adversarially chosen input aggregates (§3). Label composition follows as a corollary, together with the executable same-nonce linking attack and its fix.
2. **The scanning tax, two-sided.** We formalize receiver-side scan work as an attackable resource: the per-sat optimal attack makes every purchased output a real payment; the 2026 K_{max} cap provably converts quadratic amplification into linear; and any compliant scanner needs $\Omega(N+KM)$ operations, which separates the specification’s subtraction-based algorithm from the information-theoretic floor by a factor N/M . We propose SPECS, a set-probing scanner that meets the bound and measures $39\times$ faster on the exact worst case (§4).

3. **Collaborative payments, decoded and repaired.** We prove the specification’s warning precisely: plain joint ECDH lets the assembler identify the recipient’s output with certainty; witness-malleable inputs break detection completeness, and the spec’s four engineering footnotes are instances of one invariant we isolate; unsigned partial ECDH silently burns payer funds (the failure class of the August 2026 disclosure), and BIP-375’s mandatory DLEQ is the necessary countermeasure, which we prove. We then construct the missing secure variant: *blind collaborative payments*, combining payer-blinded scan keys à la Somsen’s blind Diffie–Hellman [41] with arbitrary-base DLEQ [44], and prove correctness, information-theoretic payer privacy against co-spenders, and DDH blindness of the assembler (§5).
4. **The delegation spectrum.** For constrained devices we characterize the two delegation modes and their complementary failures: scan-key delegation (server-side scanning) inherently leaks same-label linkability, which we prove via a perfect-simulation leakage profile and partially mitigate in-syntax with epoch-rotated labels; tweak-data delegation (light clients) admits selective omission and a crafted-match deanonymization measured at $\sim 2,200\times$ signal-to-noise on real-block parameters (§6).

In total, we answer four of the five open questions the BIP-352 specification acknowledges about its own security (collaborative transactions; light-client support; trustless tweak-data sourcing; the BIP-158 interaction), with proofs, constructions, or restorations. An open benchmark suite of twelve experiments, including end-to-end attack–fix demonstrations for every negative result, accompanies the paper.

2 Preliminaries and Protocol

Notation. $\mathbb{G}$ is secp256k1 of order q with generator G ; scalars are lowercase, points uppercase. Tagged hashes H_{In} , H_{SS} , H_{Lbl} are modeled as independent random oracles. $x(\cdot)$ denotes the x-only encoding of a point. We write N for the number of Taproot outputs of a transaction and M for the label count of a wallet ($m=0$ is the reserved change label).

BIP-352 in brief. A receiver holds $(b_{\text{scan}}, B_{\text{scan}})$ online and $(b_{\text{spend}}, B_{\text{spend}})$ offline. For a transaction with eligible input public keys $A_1, \dots, A_n$ (P2TR/P2WPKH/P2SH-P2WPKH/P2PKH, with taproot inputs always contributing their output key), let $A = \sum_i A_i$ (even-Y normalized) and $ih = H_{\text{In}}(\text{outpoint}_L \| A)$ for the lexicographically smallest outpoint. The sender computes $S = ih \cdot a \cdot B_{\text{scan}}$, $t_k = H_{\text{SS}}(x(S) \| k)$, and pays to $P_k = B_m + t_k G$ where $B_m = B_{\text{spend}} + H_{\text{Lbl}}(b_{\text{scan}} \| m)G$ is a published (labeled) spend key. The receiver scans every eligible transaction: compute $S' = ih \cdot b_{\text{scan}} \cdot A$; for $k = 0, 1, \dots$ compare $P_k = B_{\text{spend}} + t_k G$ against all Taproot outputs; on a match, remove it and test for labels via $\text{out} - P_k$ against the precomputed set $\{\ell_m G\}$ with the x-only $\pm Y$ retry; *stop at the first k without a match*, or at $k = K_{\text{max}}=2323$. We pin the specification, its test vectors, and the adjacent BIP-374/375/376 drafts by content hash in our artifact.

Adversary tiers. We distinguish five tiers, which the rest of the paper treats uniformly:

Tier	Holds	Question answered
T1 observer	chain, addresses	which outputs pay a published address (§3)
T2 co-transactor	T1 + own inputs	linking own payment to later ones (§3)
T3 scan-key delegate	T1 + b_{scan}	same-label linkability across transactions (§6)
T4 co-spender	T2 in a shared tx	identifying a co-spender's payee; burning it (§5)
T5 serving layer	tweak/filter server	omitting payments; deanonymizing clients (§6)

3 Security of the Core Protocol

3.1 Syntax and games

A silent-payment tracking scheme is $(\text{Setup}, \text{LabelAddr}, \text{Send}, \text{Scan})$; Setup outputs $(b_{\text{scan}}, B_{\text{scan}}, b_{\text{spend}}, B_{\text{spend}})$, and $\text{LabelAddr}(m)$ publishes (B_{scan}, B_m) . The Bitcoin-native surface, absent from prior stealth formalizations, is that the *context* of a payment (the input aggregate A , the anchoring outpoint, the candidate output set) is public, attacker-influenced chain data.

A *transaction context* is $ctx = (\{A_i\}, \text{outpoint}_L, \text{slots})$; it is *valid* if $A \neq \infty$ and H_{In} yields a legal scalar, matching the specification's fail/skip rules. Our games (Algorithm 1) are two-phase: the adversary queries an honest-send oracle $\mathcal{O}_{\text{send}}(i, ctx)$ for either address, then submits a fresh, valid challenge context ctx^* (never previously queried, and the challenge sender's input slot is challenger-generated—a forced modeling choice, since any legal payment requires the sender to know a discrete log); the challenger embeds a payment to addr_b among the adversary's filler outputs, and the adversary guesses b after oracle closure. Tier T2 extends learning with the adversary's own inputs and partial secrets $\mathcal{O}_{\text{part}}(i, ctx) = (ih \cdot a_j)B_{\text{scan}}^i$.

Definition 1 (Unlinkability, UNL-OBS/UNL-CO). *For the two-address game above (and its co-transactor extension), the scheme is unlinkable if for all PPT adversaries $\text{Adv}_{\mathcal{A}}^{\text{UNL}}(\lambda) = |\Pr[b'=b] - \frac{1}{2}|$ is negligible.*

3.2 Reduction and lemmas

Lemma 1 (DDH embedding with adversarial aggregate). *Given a DDH challenge $(X=xG, Y=yG, T)$, a reduction can answer UNL games with $B_{\text{scan}} := Y$ and the challenger-generated sender slot set to X , programming $H_{\text{In}}(\text{outpoint}_L^* \| A^*)$ with value ih and setting $S^* := T^{ih}$. In the real case $T = xyG$ this is exactly the honest $S^* = (ih \cdot x)Y$; in the random case S^* is uniform. The adversary's own input contributions $(ih \cdot a_j)Y$ are computable by it and cancel no information, and the adversary cannot force a known discrete log on $A^* = X + \sum A_j$ without solving for x .*

Algorithm 1 Games UNL-OBS (observer) and its co-transactor extension UNL-CO (lines marked $\triangle$).

- 1: $(\text{sk}_i, \text{addr}_i) \leftarrow \text{Setup}(1^\lambda)$, $i \in \{0, 1\}$; $b \leftarrow \{0, 1\}$; give $\text{addr}_0, \text{addr}_1$ to $\mathcal{A}$
Learning phase (adaptive):
 - 2: $\mathcal{O}_{\text{send}}(i, \text{ctx})$: if ctx invalid, return $\perp$; with a fresh sender secret, return the honest output keys for addr_i in ctx
 - 3: $\triangle \mathcal{A}$ may include its own inputs (a_j, A_j) in any ctx , and query $\mathcal{O}_{\text{part}}(i, \text{ctx}, j) = (ih(\text{ctx}) \cdot a_j)B_{\text{scan}}^i$
Challenge (single):
 - 4: $\mathcal{A}$ outputs valid ctx^* (*fresh*: distinct from every queried ctx) and filler outputs
 - 5: challenger generates the sender slot a^* , embeds the payment to addr_b among the fillers, returns the output list; oracles close
 - 6: $\mathcal{A}$ outputs b' ; win iff $b' = b$
-

Theorem 1 (Observer and co-transactor unlinkability, concrete). *In the ROM, for every adversary $\mathcal{A}$ posing at most q_H random-oracle queries, q_S send-oracle queries, and running in time t ,*

$$\text{Adv}_{\mathcal{A}}^{\text{UNL-OBS}} \leq (2q_H + 2) \cdot \text{Adv}_{\mathcal{B}}^{\text{DDH}}(t') + \frac{q_S^2 + 3q_H}{2^\lambda},$$

in time $t' \approx t$; the same bound holds for UNL-CO in both its same-transaction and cross-transaction variants (the latter by H_{in} domain separation). The loss is the two hybrids; the q_H factor is the standard random-oracle programming-collision term. At mainnet scale ($q_H \leq 2^{60}$, $\lambda=256$) the additive term is $\approx 2^{-193}$: the binding constraint is DDH itself, not the bound.

Proof (Proof sketch). Two hybrids move the challenge payment first to a uniformly random output key and then to the other address; each step invokes Lemma 1 once. With S^* uniform, $t^* = H_{SS}(x(S^*)||k)$ is a fresh random-oracle output, so $P^* = B_m + t^*G$ is a uniformly random point, information-theoretically hidden among the adversary's fillers. The full proof, including random-oracle bookkeeping and the cross-transaction argument, is in Appendix A; the additive term counts send-oracle context collisions ($q_S^2/2^\lambda$) and per-hybrid programming collisions.

Corollary 1 (Label composition, with attack-fix pair). *Joint unlinkability of outputs to distinct labels of one wallet holds under DDH given the specification's nonce-increment rule. Without it—two labels paid in one transaction with the same k —the difference of the two output keys equals the difference of the two published addresses, $P_{m_1,k} - P_{m_2,k} = B_{m_1} - B_{m_2}$, a public subtraction test linking outputs to the address pair; our artifact executes this test, and its blinded ($k++$) counterpart fails it. The specification's footnote prescribing the increment is therefore load-bearing.*

Proposition 1 (Detection soundness and liveness). *A non-payment output is falsely detected with probability at most $NK_{\text{max}}(M+1)2^{-255}$ per transaction; Scan detects every honest output if and only if the sender respects group*

size $\leq K_{\max}$ and contiguous k —the compliance predicate whose violation (a specification-conforming sender exceeding the receiver’s cap) was precisely the funds-loss edge documented in the 2026 cap revision.

4 The Scanning Tax

Every silent-payments wallet must inspect every eligible transaction before it can stop. We treat this receiver-side work as an attackable resource and characterize it from both sides.

4.1 Baseline and the optimal attack

Proposition 2 (Baseline tax). *An eligible transaction with N Taproot outputs costs every wallet 2 scalar multiplications plus $\Theta(N)$ point operations before any attack. Measured over 19 mainnet blocks (94,895 transactions), the eligible fraction is 4.5% (Wilson 95% CI [4.40, 4.66]; per-block range 0.8–14.2%; P2SH-P2WPKH inputs excluded by our detection, so this is a lower bound), averaging 2.09 Taproot outputs per eligible transaction—about 0.17 CPU-hours per wallet-year. Secondary sources report Taproot shares up to 35–45% under different denominators and windows; our direct measurement is the scan-relevant statistic, and the attack below does not depend on it.*

Proposition 3 (Optimal amplification). *For a fixed per-transaction budget of N outputs, the specification’s scan loop performs $W(N, K) = (K+1)N - \binom{K+1}{2}$ label checks, which is strictly increasing in K on $[0, N]$: the per-sat optimum makes every purchased output a real payment ($K=N$), yielding $\Theta(N^2)$ work. This is exactly the shape of the signet demonstration transaction that motivated $K_{\max}$ [13].*

Theorem 2 ($K_{\max}$ converts quadratic amplification into linear). *With the receiver-side cap $K \leq K_{\max}$, worst-case work is $\Theta(NK_{\max})$ —linear in the attacker budget; without it, $\Theta(N^2)$. Quantitatively, our model predicts the ratio $\frac{W(N, N)}{W(N, K_{\max})} = 5.27$ for the signet worst case and the measured ratio is 5.26 [13]; the ratio cancels the calibration constant, so the comparison is independent of our per-check cost estimate. The absolute anchor (31.4s for $N=23,230$ in a C implementation) is used as a unit conversion; Table 4 in the Appendix collects these reconciliations.*

Attack economics. A full block of such outputs costs the attacker its fees only. Over a one-year fee series (Fig. 2), the full-block attack cost ranged from 144,783 to 7,001,155 sats (median 2.0M): at the observed minimum, roughly 0.0015 BTC buys 51.3M forced label checks—about 31.4 CPU-seconds—*per wallet*, simultaneously for the entire user population, and again on every rescan. Recent blocks are essentially full (median free space ≈ 1.6 kB per virtual megabyte in our sample), so a full-block attack *displaces* fee payers and prices at the high-priority margin: our median-rate series is a lower bound, and in-block 90th-percentile rates put the one-shot cost at 24M–273M sats.

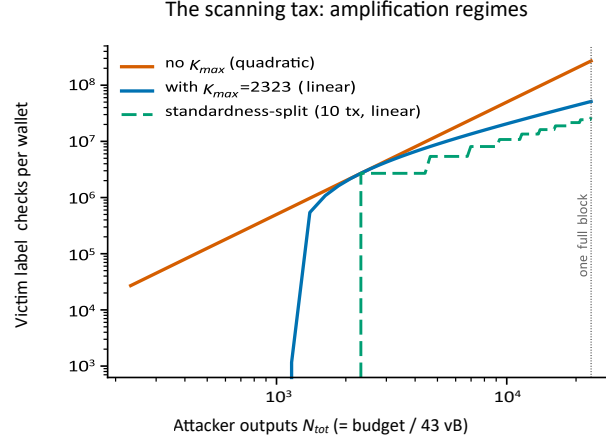


Fig. 1. Worst-case victim work per wallet against attacker budget. Without the cap, the optimal all-real attack is quadratic (orange solid); $K_{\max}$ makes it linear (blue solid); splitting across standardness-constrained transactions is also linear, at lower efficiency and in discrete per-transaction steps (green dashed).

4.2 A two-sided bound and the SPECS scanner

Lemma 2 (One probe, one hypothesis). *Under the protocol’s output distribution, a single membership test (hash-set probe or point comparison) rules out at most one payment hypothesis (k, m) : candidate keys $P_k + \ell_m G$ are pairwise independent and uniform, so a probe hits a second hypothesis with probability 2^{-255} . Batch inversion accelerates checks within one hypothesis and does not reduce hypothesis coverage.*

Theorem 3 (Scan lower bound). *Any scanner that detects all payments of an adversarial transaction with negligible false positives performs $\Omega(N + KM)$ group operations in expectation, under both uniform-random and worst-case output placement ($M \geq 1$ counting the change label).*

Proof. (Theorem 2) Arithmetic from Proposition 3: the cap yields $W(N, K_{\max}) = (K_{\max} + 1)N - \binom{K_{\max} + 1}{2} = \Theta(NK_{\max})$ while the uncapped optimum is $W(N, N) = \Theta(N^2)$; the stated ratio is $W(23\,255, 23\,255)/W(23\,255, 2\,323)$. $\square$

Table 1 situates these results against the community’s engineering record.

Proof. (Theorem 3) Place the payment uniformly at random. (i) If a scanner leaves an output unread, that output carries the payment with probability proportional to its share, so expected unread-loss forces $\Omega(N)$ reads under both uniform and worst-order placement. (ii) By Lemma 2, each membership test rules out at most one hypothesis (k, m) ; a payment uniform over the $KM + 1$ hypotheses is missed with probability at least $|U|/(KM + 1)$ when $|U|$ hypotheses are never probed, forcing $\Omega(KM)$ expected probes. The bounds add. $\square$

Table 1. Engineering record vs. formal results added here.

In the public record before this work	New here
worst-case shape ($K=N$), timing 165.2s/31.4s [13]	closed-form $W(N, K)$; ratio 5.27 vs. measured 5.26 (calibration-free)
cap turns “minutes into tens of seconds” [13,5]	quadratic-to-linear theorem with the exact coefficient
amortized-cost concern raised in discussion	$\Omega(N+KM)$ lower bound: no compliant scanner, shared or local, beats linear-in-budget while remaining complete
LabelSet-style probing sketched as an alternative [14]	SPECS : specification-grade algorithm meeting the bound, measured 39 $\times$, with migration rule
fee-cost intuition	one-year price series; displacement pricing band

The specification’s algorithm is $\Theta(NK)$: for label-less wallets it is a factor $\Theta(N)$ above the floor. The set-probing alternative already discussed by practitioners [14,13] meets the bound; we specify it as a drop-in replacement.

Construction 1 (SPECS: set-probing efficient coin scan) *Build a hash set of the transaction’s output keys (N inserts). For each k (one scalar multiplication), test membership of P_k and of $P_k + \ell_m G$ for each label m . Stop at the first childless k , exactly as the specification prescribes. Work: $(1+K)c_{\text{mul}} + KM c_{\text{add}} + (N+KM)c_{\text{probe}}$, i.e. $O(N+KM)$, meeting Theorem 3 up to constants; detection results and stop semantics are identical to the specification’s algorithm (same relation set, enumerated by hypothesis instead of by output), so security statements carry over verbatim.*

On the exact standardness-regime worst case ($N=K=2323$), measured end-to-end in our libsecp256k1-backed harness (Fig. 3): specification loop 2.37s vs. SPECS 0.061s—a 39 $\times$ measured speedup, driver-invariant, growing linearly with N as the bound predicts (6.8 $\times$ at $N=200$, 12.8 $\times$ at $N=400$). Under an implementation charging full elliptic-curve work per label check, the gap reaches $\sim 500\times$ (Appendix B). For wallets with many labels the crossover $M^* \approx N$ favors the specification’s algorithm—consistent with the specification’s own recovery guidance of precomputed label sets, which our recovery quantification confirms (Appendix B).

4.3 Population and mitigation

Amplification composes linearly across wallets with no amortization: each wallet’s shared secret is an independent scalar multiple, so a scanning server performs the full per-wallet work itself. Every mitigation that lowers per-wallet



Fig. 2. Weekly median fee sampling over one year: the price of a full-block scanning attack (blue solid), with the minimum (green dashed) and median (orange dashed) annotated. Median-rate pricing is a lower bound—full blocks price displacement at high-priority rates (see text); the attack is cheapest exactly when benign fee pressure is lowest.

attack cost (tighter $K_{\max}$, SPECS, hardware acceleration) simultaneously lowers the incentive of constrained devices to delegate scanning—a privacy-relevant channel we return to in §6.

5 Collaborative Payments

The specification’s warning against collaborative transactions is its most explicit admission.

5.1 What the warning says

Proposition 4 (Assembler omniscience). *In the plain construction, any party that assembles the partial secrets $S_i = (ih \cdot a_i)_{B_{\text{scan}}}$ learns $S = \sum_i S_i$ and identifies the recipient’s output among the outputs with certainty. Co-spenders holding all but one partial cannot: their view is an instance of the co-transactor tier, covered by Theorem 1.*

Theorem 4 (Witness-independence $\Leftrightarrow$ detection completeness). *Call an input type witness-independent if every legal spend of the same UTXO yields the same derivation key. If all eligible inputs are witness-independent, Scan is complete for collaborative transactions regardless of adversarial re-signing; conversely, any eligible input admitting two witnesses with distinct derivation keys admits a burning strategy in which a malicious co-spenders derives with one key set and broadcasts with the other, making the payer’s funds undiscoverable.*

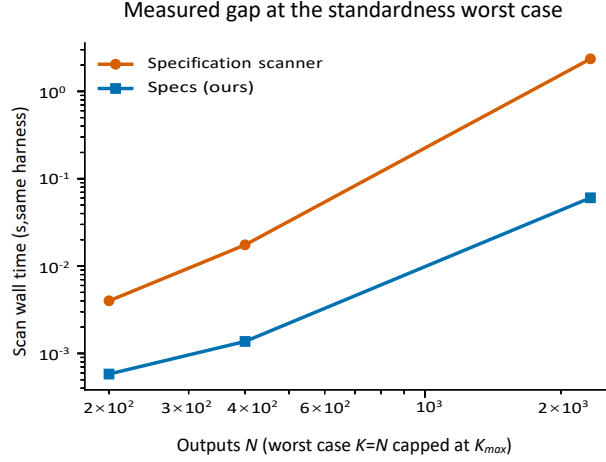


Fig. 3. Measured scan time on the worst case under the same libsecp256k1-backed harness: the specification scanner (orange, circles) vs. SPECS (blue, squares). The gap grows linearly in N as Theorem 3 predicts; at the standardness worst case ($N=K=2,323$) it measures $39\times=2.37$ s vs. 61 ms.

Algorithm 2 Game G-PSBT (signer–host coordination).

- 1: parties: honest signer(s) that see only the PSBT fields they receive; adversary $\mathcal{H}$ = host, which constructs, updates, assembles, and broadcasts; declared payee address public
 - 2: $\mathcal{H}$ adaptively supplies inputs, outputs, and ECDH shares C'_i (each optionally with a $\text{DLEQ}_{G,B'}$ proof), and collects signatures
 - 3: a signer accepts a share iff its DLEQ verifies *and* the proven key point equals the expected $ih \cdot A_i$; all scripts must be final and locked before any signature
 - 4: $\mathcal{H}$ broadcasts tx ; $\mathcal{H}$ wins if the payee's Scan of tx misses the payment, or the paid output differs from the declared payee's derivation
-

The specification's four engineering footnotes—excluding conditional-branch inputs, always using the taproot output key across key- and script-path spends, parsing non-standard P2PKH scriptSigs, and forbidding ANYONECANPAY—are instances of this single invariant; Theorem 4 turns a patch list into a characterization.

Proposition 5 (Unsigned partial ECDH burns payer funds). *With MuSig/FROST-aggregated inputs, a malicious shard may submit $S_{\text{mal}} \neq a_m B_{\text{scan}}$; the assembled output is then derived from a corrupted sum, the receiver's scan misses, and the payment is irrecoverable. Verification that each partial is correctly formed requires a DLEQ argument; unsigned partials admit no such check.*

This is the failure class of the August 2026 hardware-wallet disclosure, in which a malicious host locked silent-payment funds at unintended addresses recoverable only with attacker cooperation [10]; our artifact executes both the burn

and its DLEQ interception. BIP-375, the PSBT extension for silent payments, already mandates a BIP-374 DLEQ proof on every ECDH share; Theorem 5 makes that mandate provably necessary.

Theorem 5 (DLEQ necessity in the signer–host model). *In the G-PSBT game (malicious host assembles; the signer sees only PSBT fields), detection completeness against the host holds if and only if every share carries a valid DLEQ proof: verified shares make S uniquely determined by the transaction’s public input set, reducing to Theorem 4’s precondition; an unproven share admits the Proposition 5 burn. BIP-375’s script-lockdown and `SIGHASH_ALL` restrictions are the coordination-layer enforcement of the same invariant.*

5.2 Blind collaborative payments

We now construct the variant whose absence the specification laments, reusing Somsen’s blind Diffie–Hellman idea [41]—the mechanism is his; the instantiation, the arbitrary-base DLEQ integration [44], and the analysis are ours.

Construction 2 (Blind collaborative payment, CJ’) *Let the payer hold input key a_0 and wish to pay $(B_{\text{scan}}, B_{\text{spend}})$ inside a collaborative transaction with co-spenders $a_1, \dots, a_{n-1}$ and assembler X .*

1. Blind: the payer draws $r \leftarrow \mathbb{Z}_q$ and broadcasts $B' = B_{\text{scan}} + rG$.
2. Blinded shares: each party i (payer included) sends $C'_i = s_i B'$ with $s_i = ih \cdot a_i$, together with $\text{DLEQ}_{G, B'}(ih \cdot A_i, C'_i)$ proving equality of discrete logs relative to the two bases; $ih \cdot A_i$ is computable by everyone.
3. Unblind and derive: the payer verifies all proofs, computes $C_i = C'_i - r(ih \cdot A_i)$, sums $S = \sum_i C_i$, derives $P_k = B_{\text{spend}} + H_{\text{SS}}(x(S) || k)G$, and hands output scripts to X among fillers.

Theorem 6 (CJ’ properties; single-session). *All properties are stated for a single collaborative session with a static, non-concurrent assembler (one challenge context); see the concurrency remark below. (i) Correctness: $C_i = s_i B_{\text{scan}}$ by algebra, so S equals the specification’s shared secret and the receiver’s scan hits. (ii) Payer privacy against co-spenders (information-theoretic): $B' = B_{\text{scan}} + rG$ is uniform and independent of B_{scan} , so co-spenders learn nothing about the payee—strictly stronger than the plain construction, which reveals B_{scan} to every co-spender. (iii) Assembler blindness (DDH): X ’s view $\{B', C'_i\}$ satisfies $\sum_i C'_i = S + r ih \cdot A_\Sigma$; separating S requires the discrete log embedded in $rG = B' - B_{\text{scan}}$, so output identification reduces to DDH exactly as in Lemma 1. (iv) Detection completeness: DLEQ-verified shares make malicious corruption of S equivalent to forging the proof, so the burn of Proposition 5 is intercepted before assembly.*

Our artifact executes all four properties end-to-end: the assembler identifies the output with certainty under plain partials and fails under blinded ones; forged shares with mismatched keys are rejected by the binding check; the victim’s scan hits the blinded derivation.

Concurrency and the ROS family. Our blindness argument is single-session; like other game-based analyses of deployed protocols [38], we claim no concurrent composition. The ROS line [2,16] concerns concurrent signing oracles, which CJ' does not instantiate—it blinds a Diffie–Hellman share—so that surface does not transfer directly; a multi-session treatment remains open.

6 Delegation and Light Clients

Constrained devices cannot scan: hardware wallets ship send-only and phones scan reluctantly. Two delegation modes exist in the wild—server-side scanning with the scan key (Frigate-style) and tweak-data serving (BlindBit-style light clients)—and they fail in complementary directions.

6.1 Scan-key delegation

Theorem 7 (Leakage profile and inherent same-label linkability). *The view of any party holding b_{scan} is perfectly simulatable from (chain data, published $\{B_m\}$, the detected set D , and the label points $\{\ell_m G\}_{m \in D}$). Consequently it extracts $\ell_m G = \text{out} - P_k$ for every detected payment, and same-label linkability across transactions is unavoidable in this delegation mode (the trivial adversary wins with advantage $1 - \text{negl}$). What remains hidden is the label index m itself (discrete log) and all spending capability; cross-address linkage is public by specification design.*

Unlike the harvest-now-decrypt-later concern [9], this needs no quantum adversary and no archival patience: today’s scanning server already links. An in-syntax mitigation is *epoch-rotated labels*: publish $B_{m,e} = B_{\text{spend}} + H_{\text{Lbl}}(b_{\text{scan}} \| m \| e)G$ per epoch e ; label points are then fresh per epoch, confining server-side same-label linkability to within-epoch windows, at the cost of a payer-side refresh window and a recovery set grown by the epoch count—wallet policy, not protocol change.

6.2 Tweak-data delegation (light clients)

The specification’s Appendix A sketch—per-transaction tweak data plus compact filters—has been instantiated by indexers and wallets. Two serving-layer attacks formalize the community’s informal concerns [7]:

Proposition 6 (Crafted-match deanonymization). *A server that controls tweak/filter serving can serve a filter containing only the target’s derived key, so that only the target fetches the block; observing fetches identifies it. Under real-block parameters (≈ 470 derived keys per block from our measurement, filter false-positive rate 2^{-20}), an honest filter makes any other client fetch with probability $\approx 4.5 \times 10^{-4}$ per block: signal-to-noise ratio $\approx 2,200$, identification within a few blocks. Mitigations (ephemeral identities; fetching full blocks from random nodes) blend traffic but do not restore indistinguishability.*

Table 2. The delegation spectrum for silent-payments receivers.

Mode	Privacy risk	Liveness risk	Cost
Self-scan	none (Thm. 1)	none	baseline + attack
Scan-key delegation	same-label link (Thm. 7)	none	low
Tweak delegation	crafted-match (Pr. 6)	omission (Pr. 7)	lowest

Table 3. Definitional surface: the abstract stealth-signature template vs. the deployed BIP-352 specification analyzed here.

Feature	Pu et al. [35]	this work
receiver-chosen tracking data	✓	— (adversarial chain data)
scan/spend key separation	—	✓
labels (address sub-derivation)	—	✓
receiver scan cost axis	—	✓
collaborative/co-spender tier	—	✓
delegated-scan (key exposure) tier	fuzzy tracking	✓ (leakage profile)

Proposition 7 (Selective omission and restoration). *A tweak server can withhold the tweak of a block containing a payment; the client derives wrong keys and the payment is silently lost—a reported real-world instance of server-side data omission appears in [29]. Querying n servers and unioning their tweak sets restores detection completeness whenever one is honest; tamper-evident per-block fingerprints [6] convert omission from undetectable to attributable after the fact, an accountability notion in the sense of [22,18].*

6.3 The delegation spectrum

Together with §4–5 this yields the trade-off matrix of Table 2. Cheaper scanning doubles as privacy protection: the migration threshold for a device slowed by factor s scales as B^*/s —tens of blocks per epoch for desktops under the 2026 cap, but ≈ 0.04 blocks at $s \approx 10^3$, roughly the hardware-wallet-companion population that already delegates.

7 Related Work

Stealth-address formalizations. The game-based template is due to [35]; the PDPKS/ARKG line [27,12,28,30] studies publicly derived keys abstractly (recent work [12] shows earlier notions were themselves insufficient); PDPKS theory [32] establishes equivalence with CCA-secure PKE. For *deployed* systems: zkBSA [52] formalizes ERC-5564 receivers with compliance machinery on Ethereum; Umbra’s anonymity was analyzed in [21]; attribute-driven accountability on Bitcoin appears in [34], which treats silent payments only as related work. None covers BIP-352’s input aggregation, labels, witness rules, scanning economics, or collaborative settings (Table 3); the 2026 stealth-formalization record splits cleanly

along three lines—this paper (deployed-specification analysis and scanning economics), zkBSA (Ethereum compliance), and IB-MSS accountability [34]. Earlier lineage analyses include [50,47,33].

Detection burden. Fuzzy message detection [1] and its analyses [39], multi-server variants [17], and the oblivious-retrieval line—OMR [24], PerfOMR and GroupOMR [26,25], sender-malicious DoS-resilient OMR [23], UnifOMR [15], which the Zcash Foundation has evaluated for shielded scanning [51]—study receiver-side detection cost in messaging and private chains. The deployed spectrum now spans Monero’s view tags ($p=2^{-8}$, live since v0.18 [43]), tunable FMD (p adjustable), and silent payments at $p=0$: privacy purchased with cryptography rather than cover traffic, at the price of maximal and here adversarially amplifiable detection work. Our measured baseline implies a $\sim 250\times$ per-transaction detection premium over a full-download detector. DoS-resistant OMR [23] hardens the *server* against malicious senders; our adversary instead pays the chain itself to tax *every receiver*—a complementary axis. Attack-cost-as-mispricing has formal relatives in blockspace-equilibrium and rollup-pricing analyses [48,8]; our displacement-pricing observation (§4) is the receiver-side, L1 instance.

A barrier to FMD-style repair. The detection-delegation recipe—per-server detection keys, server-side filtering, receiver-side combination—cannot repair silent-payments delegation without sender-side changes: FMD/OMR senders embed a per-message clue, whereas a BIP-352 output’s derivation data is protocol-fixed (input keys, outpoints) with no receiver-addressable flag, so any server-side filtering either holds b_{scan} (Tier-3 leakage, Theorem 7) or receives tweak data verbatim (Tier-5 attacks, Propositions 6–7). Repairing delegation within BIP-352’s send format is therefore a structural barrier rather than an engineering gap; Monero-style view tags or sender-embedded clues require a protocol fork. The mitigations of §6 accordingly stay within the current format.

Asymmetric DoS economics. Cost-asymmetry formalizations include [11,19,31,37]; the time-anonymity trade-off under DoS was studied for anonymity networks in [49], and DoS-induced funneling on Bitcoin in [3]. Our scanning tax is the receiver-side detection-work instantiation, with a deployed mitigation (K_{max}) and a matching lower bound.

Ecosystem sources. We rely on primary engineering records: the specification family [20,44,46,45], the K_{max} incident and benchmarks [13,5,14,4], the hardware-wallet disclosure [10], light-client discussions and deployments [7,6,36,40], and the quantum-exposure note [9].

8 Discussion and Limitations

Scope. Our treatment is game-based and single-session (no UC claims), as in deployed-protocol analyses like KEMTLS [38]. The migration-channel analysis is a deliberately small model (illustrative sensitivity parameters) whose purpose is to identify the exposure surface (constrained devices), not to estimate it. Our

mainnet measurement covers a 19-block window in a quiet fee regime; scan-relevant statistics vary with Taproot adoption and activity mix, and none of our attack results depend on them. Implementation-level constants carry a stated driver-overhead factor, and our headline validations are ratio-based to remain implementation-independent.

Deployment is unmeasurable. No third party can measure silent-payments usage: deciding whether an output pays a published address *is* the UNL-OBS game, so Theorem 1 rules out any public membership test. Adoption surfaces only in disclosed data and ecosystem signals—wallet release cadence, and dedicated scanning infrastructure [36,40] whose emergence is driven by the receiver-side cost quantified in §4.

Future work. Spending-side coordination (BIP-376) remains unanalyzed; the delegation spectrum’s middle ground deserves a tight characterization; SPECS awaits an upstream proposal, for which this paper supplies the security argument.

9 Conclusion

Silent payments secured real money on footnotes. This paper gave the protocol its proofs and its prices: the DDH assumption behind unlinkability, the linear scanning cost every compliant receiver must bear, what collaboration and delegation give away—and a blind construction and a cheaper scanner that claw part of it back. Our anonymized benchmark suite reproduces every number.

A Full Proofs

Theorem 1. Hybrid H_0 pays $addr_0$; H_1 pays a uniformly random Taproot key; H_2 pays $addr_1$. $H_0 \rightarrow H_1$: embed per Lemma 1 with $B_{\text{scan}}^{(0)} := Y$; if the DDH challenge is real the game is identical to H_0 , if random then S^* uniform, hence $t^* = H_{\text{SS}}(x(S^*)\|k)$ is a fresh random-oracle output (the adversary queries H_{SS} on $x(S^*)$ only if it guesses the point S^* , which requires x or z), and $P^* = B_m + t^*G$ is uniform: the embedding is information-theoretically hidden among fillers, so H_1 ’s advantage is 0. $H_1 \rightarrow H_2$ symmetrically with $B_{\text{scan}}^{(1)} := Y$. Distinguishing either pair wins DDH with the same advantage; random-oracle programming conflicts and the negligible scalar -validity abort are absorbed in **negl**. Cross-transaction unlinkability additionally uses that distinct contexts yield distinct H_{in} inputs, hence independent S values, and the adversary’s own payment yields no additional advantage beyond its partial secrets, which the reduction answers honestly. $\square$

Lemma 1. The reduction plays the challenger, sets the sender slot to X (its secret is x , unknown), and the target wallet’s scan key to Y . On the challenge context it computes $A^* = X + \sum_j A_j$ and $outpoint_L^*$, and programs or answers $H_{\text{in}}(outpoint_L^*\|x(A^*))$ with $ih \neq 0$ (abort with probability $\leq 2^{-127}$). It returns

Table 4. Model–reality reconciliation (ratio-based, calibration-free where marked).

Quantity	Model	Measured
$K_{\max}$ drop ratio	5.27	5.26 (computed from [13])
optimal attack shape	$K=N$ all-real outputs	signet tx [13]
worst-case checks ($N=K=2323$)	$\binom{2324}{2} = 2,699,326$	2,699,326 (ours)
SPECS speedup at that point	$\Theta(N)$ growth	$39\times$ measured
light-client detection SNR	$1/(N_q p_{fp})$	$\sim 2,200\times$ (real-block params)

$S^* = T^{ih}$. If $T = xyG$ then $T^{ih} = (ih \cdot x)Y$, the honest value for sender secret x . If $T = zG$ then $ih \cdot z$ is uniform, and so is S^* . The adversary’s own inputs contribute $(ih \cdot a_j)Y$, which it computes itself; forcing a known discrete log on A^* would reveal x , solving DL. $\square$

Theorem 4. ($\Rightarrow$) If every eligible input is witness-independent, the derivation key of each input is a function of the spent UTXO alone; hence A is determined by the transaction’s input set, S by $(A, \text{outpoint}_L)$, and the receiver’s scan of the final transaction matches the sender’s derivation regardless of which witnesses produced the signatures. ($\Leftarrow$) If some input admits witnesses yielding keys $X \neq Y$, a malicious co-spender participates in derivation under X and broadcasts under Y : outputs were computed for A_X , the chain carries A_Y , the receiver computes a different S and misses the outputs; the funds become undiscoverable by the receiver yet remain recoverable by the attacker, who derived the output keys—a ransom-shaped loss matching the vendor disclosure’s recovery condition [10]. $\square$

Theorem 6(iii). The assembler’s view is B' , the multiset $\{C'_i\}$, the final outputs, and public data; from these, $\sum_i C'_i = S + r' \cdot (ih \cdot A_\Sigma)$ where $r'G = B' - B_{\text{scan}}$ is public as a point. A distinguisher that identifies the output key $P_0 = B_{\text{spend}} + H_{\text{SS}}(x(S)||0)G$ among the outputs therefore separates knowledge of S ; construct the reduction exactly as Lemma 1 with the DDH challenge embedded in $(r'G, S+r'ihA_\Sigma)$: in the real case the assembler’s candidate P_0 appears among the outputs, in the random case S is uniform and P_0 is a fresh random-oracle-derived point, hidden among fillers. Honest-send oracle queries are answered with fresh randomness as in the main proof; the multi-share sum is handled by embedding the challenge in the aggregate term (all C'_i beyond the challenge pair are simulated with known discrete logs). $\square$

B Numerical Details

Calibration. Per-check cost is derived from the signet anchor; ratio-based validations (5.27 vs 5.26; worst-order vs shuffled within implementation-level factors) are constant-free. The three-point calibration across our implementations and the C anchor differs by known driver-overhead factors only. At the full-block attack point, the anchor-cost model (full elliptic-curve work per label check) prices

the specification loop at 31.5 CPU-seconds against 0.061 for SPECS ($M=1$): the $\sim 500\times$ worst-case gap of §4.

Attack-cost series. Weekly sampling: for each window we take the block-height tip back one week and use the median of per-block median fee rates over that window’s 15-block API page; attack size 1,000,165 vB (23,255 outputs at 43 vB plus transaction skeleton). The headline minimum (144,783 sats) is the *lower bound* obtained by pricing the whole attack at the cheapest weekly median; the displacement band (24M–273M sats) prices it at in-block 90th-percentile rates, since sampled blocks are essentially full (median free space ≈ 1.6 kB/vMB). Sensitivity: pricing at window *minimum* rather than median rates changes the lower bound by less than $2\times$ throughout the series.

Block-selection rule. The 19 measured blocks are a contiguous recent window taken at depth 5–24 below tip at measurement time (all cached with raw API responses); the per-block eligible fraction varies 0.8–14.2% across them, and we report the aggregate Wilson interval rather than per-block extremes. No attack result depends on η .

Recovery. 100k-label precompute ≈ 2.5 CPU-s and 3.2 MB once; per-transaction work unchanged for the specification algorithm, $K \cdot M$ probes per attack transaction for SPECS—the crossover $M^* \approx N$ favors the specification algorithm exactly in the recovery regime its guidance anticipates.

Measurements. 19 mainnet blocks, 94,895 transactions, raw API responses cached in the artifact; eligibility detection is a lower bound (P2SH-P2WPKH not distinguishable from prevout type alone).

References

1. Beck, G., Len, J., Miers, I., Green, M.: Fuzzy message detection. In: Proc. ACM CCS (2021), ePrint 2021/089. doi:10.1145/3460120.3484545
2. Benhamouda, F., Lepoint, T., Loss, J., Orrù, M., Raykova, M.: On the (in)security of ROS. In: Proc. EUROCRYPT (Best Paper) (2021)
3. Biryukov, A., Pustogarov, I.: Bitcoin over Tor isn’t a good idea. In: Proc. IEEE S&P (2015), arXiv:1410.6079
4. Bitcoin Core Dev Tech: Silent payments update (“no security proof yet”). Meeting transcript (2026), <https://btctranscripts.com/bitcoin-core-dev-tech/2026-05/silent-payments>
5. Bitcoin Optech: Newsletter #392: Limiting per-group silent payment recipients. bitcoinops.org (2026), <https://bitcoinops.org/en/newsletters/2026/02/13/>
6. Bitsaga: The silent payment server that can be caught lying. Technical blog (filter measurements; tamper-evident fingerprints) (2026), <https://bitsaga.be/insights/the-server-that-can-be-caught-lying>
7. Blagogee, S., Harding, D., et al.: Silent payments: Light client protocol. Delving Bitcoin thread #891 (2024), <https://delvingbitcoin.org/t/silent-payments-light-client-protocol/891>

8. Chaliasos, S., et al.: Unaligned incentives: Pricing attacks against blockchain rollups. arXiv:2509.17126 (2025)
9. Conduition: Silent payments harvest-now-decrypt-later attack. Technical blog (2026), <https://conduition.io/cryptography/hndl-silent-payments/>
10. Crypto, B..S.: Bitbox 08.2026 dixence update (silent payments funds-lock fix, FW 9.26.5). Official security disclosure (2026), <https://blog.bitbox.swiss/en/bitbox-08-2026-dixence-update>
11. Ding, W., Tang, Y., Wang, Y.: Asymmetric mempool DoS security: Formal definitions and provable secure designs. In: Proc. IEEE S&P (2025)
12. Emura, K.: Key-insulated and privacy-preserving signature scheme with publicly derived public key, revisited. IACR CiC 2(3):17; ePrint 2024/2019 (2025)
13. Falbesoner, S.: BIP-352: Limiting the number of per-group recipients (k_{max}). bitcoin-dev mailing list, Feb–Apr 2026, <https://gnusha.org/pi/bitcoindev/cf64109b-4d78-4515-a68e-d427243a105cn@googlegroups.com/>
14. Falbesoner, S., et al.: Silent payments module: discussion about different scanning approaches. bitcoin-core/secp256k1 issue #1799 (2026), <https://github.com/bitcoin-core/secp256k1/issues/1799>
15. Fisch, B., Liu, Z., Wang, Y., Tromer, E.: UnifOMR: Uniform-size oblivious message retrieval. ePrint 2026/910 (2026)
16. Fuchsbaauer, G., Plouviez, A., Seurin, Y.: Blind Schnorr signatures and signed El-Gamal encryption in the algebraic group model. In: Proc. EUROCRYPT (2020)
17. Goes, C., Khalniyazova, Y., Larraia, E., Song, X.: Multi-server fuzzy message detection. IACR ePrint 2025/2072 (2025)
18. Graf, E., Küstner, D., Rausch, D.: AUC: Accountable universal composability. IACR ePrint (2022), <https://publ.sec.uni-stuttgart.de/grafkuestersrausch-iacr-2022.pdf>
19. Harris, J., Zohar, A.: Flood & loot: A systemic attack on the Lightning Network. In: Proc. AFT (2020), arXiv:2006.08513
20. josibake, Somsen, R., Falbesoner, S.: BIP-352: Silent payments. Bitcoin Improvement Proposal 352 (status: Complete, v1.1.1) (2026), <https://github.com/bitcoin/bips/blob/master/bip-0352.mediawiki>
21. Kovács, A.M., Seres, I.A.: Anonymity analysis of the Umbra stealth address scheme on Ethereum. In: Companion Proc. ACM Web Conference (2024), doi:10.1145/3589335.3651963
22. Küsters, R., Truderung, T., Vogt, A.: Accountability: Definition and relationship to verifiability. In: Proc. ACM CCS (2010), doi:10.1145/1866307.1866366
23. Liu, Z., Sotiraki, A., Tromer, E., Wang, Y.: Oblivious message retrieval resilient to malicious senders. ePrint 2024/510 (2024)
24. Liu, Z., Tromer, E.: Oblivious message retrieval. In: Proc. CRYPTO (2022), ePrint 2021/1256
25. Liu, Z., Tromer, E., Wang, Y.: Group oblivious message retrieval. In: Proc. IEEE S&P (2024)
26. Liu, Z., Tromer, E., Wang, Y.: PerfOMR: Oblivious message retrieval with reduced communication and computation. In: Proc. USENIX Security (2024)
27. Liu, Z., Yang, G., Wong, D.S., Nguyen, K., Wang, H.: Key-insulated and privacy-preserving signature scheme with publicly derived public key (2019), ePrint 2018/956
28. Lundberg, B., et al.: ARKG: Public-key authenticatable renewable privacy (ARKG). ePrint 2024/846 (2024)

29. maintainers, C.: Missing funds when receiving silent payments (issue #2395, closed). GitHub issue, opened 2025-07-17 (2025), https://github.com/cake-tech/cake_wallet/issues/2395
30. Mikić, M., Srbakoski, M., Praška, S.: Post-quantum stealth address protocols. ePrint 2025/112 (2025)
31. Mirkin, M., Papadopoulos, D., Chalkias, K., Segal, I., Spiegelman, A.: BDoS: Blockchain denial of service. arXiv:1912.07497 (2019)
32. Mizuno, R., Emura, K.: Generic construction of CCA-secure PKE from key-insulated and privacy-preserving signatures with publicly derived public key. In: Proc. ICICS (2026), ePrint 2026/922
33. Modesti, P.: Formal modelling and security analysis of Bitcoin's payment protocol (BIP-70). arXiv:2103.08436 (2021)
34. Mongardini, A.M., Friolo, D., Ateniese, G.: Stealth and beyond: Attribute-driven accountability in Bitcoin transactions. In: Proc. ACNS. pp. 371–397 (2026), ePrint 2024/1789. doi:10.1007/978-3-032-32575-4_14
35. Pu, S., Thyagarajan, S.A.K., Döttling, N., Hanzlik, L.: Post quantum fuzzy stealth signatures and applications. In: Proc. ACM CCS (2023), ePrint 2023/1148. doi:10.1145/3576915.3623148
36. Raw, C., 2140-dev: Frigate: Electrum server with server-side silent payments scanning and ephemeral client keys. Open-source project + BTrust case study (2026), <https://github.com/2140-dev/frigate>
37. Saad, M., Kim, D., Nyang, D., Mohaisen, A.: Mechanisms for countering spam attacks on blockchain memory pools. J. Network and Computer Applications (2021), arXiv:2005.04842
38. Schwabe, P., Stebila, D., Wiggers, T.: Post-quantum TLS without handshake signatures. In: Proc. ACM CCS (2020), game-based (ACCE) analysis
39. Seres, I.A., Pejó, B., Burcsi, P.: The effect of false positives: Why Fuzzy Message Detection leads to fuzzy privacy guarantees? In: Proc. Financial Cryptography (2022), arXiv:2109.06576. doi:10.1007/978-3-031-18283-9_7
40. setavenger: BlindBit oracle: BIP-352 tweak indexing server. Open-source project, <https://github.com/setavenger/blindbit-oracle>
41. Somsen, R.: Blind Diffie-Hellman key exchange (blind ecash). Technical gist (2022), <https://gist.github.com/RubenSomsen/be7a4760dd4596d06963d67baf140406>
42. Spark Research: Bitcoin privacy in 2026: Silent payments, payjoin, and the surveillance arms race. spark.money research (2026), <https://www.spark.money/research/bitcoin-privacy-landscape-2026>
43. tevador, Lab, M.R.: Pruning view tags for faster wallet scanning (MRL-73). Monero Research Lab issue #73; activated v0.18 (2021), <https://github.com/monero-project/research-lab/issues/73>
44. Toth, A.: BIP-374: Discrete log equality proofs for secp256k1. Bitcoin Improvement Proposal 374 (2025), <https://github.com/bitcoin/bips/blob/master/bip-0374.mediawiki>
45. Toth, A.: BIP-376: Spending silent payment outputs with PSBTs. Bitcoin Improvement Proposal 376 (Draft) (2026), <https://github.com/bitcoin/bips/blob/master/bip-0376.mediawiki>
46. Toth, A., Chow, A., josibake: BIP-375: Sending silent payments with PSBTs. Bitcoin Improvement Proposal 375 (Draft, v0.1.1) (2026), <https://github.com/bitcoin/bips/blob/master/bip-0375.mediawiki>
47. Vijayakumaran, S.: Analysis of CryptoNote transaction graphs using the Dulmage-Mendelsohn decomposition. In: Proc. AFT (2023), doi:10.4230/LIPIcs.AFT.2023.28

48. Wang, T., Saraf, S., Heimbach, L., Babel, A., Zhang, F.: Blockspace under pressure: Equilibrium spam in blockchain fee markets. arXiv:2604.00234 (2026)
49. Yang, M., Sassone, V.: Minimising anonymity loss in anonymity networks under DoS attacks. In: Proc. ICICS (2011)
50. Yu, J., Au, M.H., Esteves-Verissimo, P.: Re-thinking untraceability in the CryptoNote-style blockchain. In: Proc. IEEE CSF (2019), doi:10.1109/CSF.2019.00014
51. Zcash Foundation: Oblivious message retrieval for shielded scanning (engineering evaluation). zfnd.org blog, <https://zfnd.org/oblivious-message-retrieval/>
52. Zheng, S., Han, Z.: zkBSA: Auditable and compliant stealth addresses for blockchains. IACR ePrint 2026/513 (2026)