Seven Bindings: A Machine-Checked Map of the Assumptions Behind Lightning's BOLT 12

Rong Qian[1], Mengrun Chen[1], Yu Cheng[1], Yuchang Zhang[2], and Zengli Guo[3]

[1] Department of Cyberspace Security, Beijing Electronic Science and Technology Institute, Beijing, China,
rqian@besti.edu.cn, 13015310107@163.com, 2863394654@qq.com
Corresponding authors: Mengrun Chen and Yu Cheng
[2] Graduate Department of North China Institute of Computer Systems Engineering,
zhangyuchang26@mails.ucas.ac.cn
[3] Faculty of Information Science and Engineering, Ocean University of China, Qingdao 266100, China,
guozengli@stu.ouc.edu.cn

Abstract. Lightning's BOLT 12 (offers, onion messages, blinded paths) is the largest addition to the Lightning protocol suite, and it is rolling out now. The specification changed three times in 2026, the community is actively debating its privacy posture, and its newest mechanism, Payer Proofs, landed in July 2026 without any analysis. We give the offer family its first formal treatment, covering the full lifecycle from request to receipt, pinned to specification commits `e20a924/1528972` and machine-checked in twenty-two ProVerif models that each converge within seconds. The results read as a map rather than a checklist: the family's privacy and integrity rest on seven bindings, removable assumptions whose removal consequences we check row by row. Receiver anonymity and payer-payee unlinkability hold under unbounded sessions and corrupted path nodes, and every protocol-level attack we exhibit lands on an assumption the specification already leans on out-of-band. The map's structure is itself the finding. The blinding key chain, designed for unlinkability, authenticates recipient-side transit for free, while the payer side's delivery binding rests on implementation discipline alone. The newest part of the specification, its payment receipts, turns out to hide a precise disclosure boundary. We close with six testable recommendations for the BOLTs repository.

Keywords: Lightning Network · BOLT 12 · formal verification · ProVerif · unlinkability · route blinding.

1 Introduction

The Lightning Network's BOLT 12 specification [1] is the largest addition to the Lightning protocol suite since its inception. Offers replace single-use invoices with reusable payment codes; onion messages provide a lightweight transport

requiring no channel liquidity; blinded paths hide a recipient's position in the network topology. Deployment is now in full swing: Core Lightning and Eclair shipped offers by default, LDK reached full support in 2025–2026, and LND, the largest implementation, only began landing BOLT 12 functionality in 2026.[4] The specification is still moving: three substantive changes landed in 2026 alone, the newest of which—the *Payer Proofs* selective-disclosure mechanism, July 2026—is its privacy-relevant frontier.[5]

And the security posture is being fought over live. A months-long 2026 discussion documented a bounce-amplification jamming attack in which a single onion message is reflected between honest victims more than 750 times [15]; invoice `description` fields were shown to leak order metadata to invoice-processing infrastructure [16]; latency-correlation concerns target the blinded-path mechanism itself [13,14]. Implementers keep patching privacy-relevant behaviors the specification leaves open: in June 2026, LDK had to bind returned invoices to the reply path they arrived on, after it emerged that an attacker could relay one payment's invoice over another's path. Elsewhere, this class of question has been institutionalized: the TLS working group now runs a formal-analysis triage that gates standardization changes.[6] For BOLT 12, no such analysis existed: formal treatments of Lightning stop at the core channel protocol (funds safety in Why3 [3] and TLA+ [4], congestion-era channel security in Spin [22], transport-layer onion-message attacks and mitigations [5]), and the equivalence-type properties that motivate blinded paths had never been machine-checked on Lightning.

This paper. We analyze the offer family as pinned to specification commits `e20a924` (BOLT 12) and `1528972` (BOLT 4), end to end: request, invoice, error, transit, delivery of the reply, and—with the 2026-07 addition—the receipt a payer can exhibit afterwards. The analysis is organized around the resulting map of assumptions rather than a list of verdicts: seven *bindings* (Table 2), each checked as one or two rows of an eighteen-row table (Table 3). Our contributions:

- **The binding map, and the asymmetry it reveals (Section 5).** Seven bindings carry the architecture, each a removable assumption whose removal consequence a row or two machine-checks. Read as assumptions rather than channels, the rows expose a structural asymmetry between the two sides of the protocol: the blinding chain, built for unlinkability, authenticates recipient-side transit for free (Theorem 7), while the payer side's delivery binding rests on implementation discipline alone (Attack 5, Theorem 12).
- **The map's evidence base: machine-checked anonymity (rows 1–4).** Extending the classical equivalence program for onion routing [10,2] to route blinding: receiver anonymity (identity-anchored, Theorem 2) and

[4] Implementation status per the public trackers: LND BOLT 12 epic #10736 (`github.com/lightningnetwork/lnd`), `lightningdevkit.org` (LDK), the CLN release notes (`docs.corelightning.org`), and the community portal `bolt12.org`.
[5] Commits `34455ff`, `3111193`, `e20a924` in `https://github.com/lightning/bolts`.
[6] TLS Formal Analysis Triage Team, `github.com/tlswg/tls-fatt` (2024).

payer-payee unlinkability in the 2×2 swap of two payers and two offers (Theorem 3), stated in replicated (unbounded-session) form, with robustness against corrupted path nodes per property and position: receiver anonymity against a corrupted introduction node and a corrupted middle hop (Theorem 4), unlinkability against a corrupted introduction node (Theorem 3, M7 clause).

- **The receipt plane: Payer Proofs (rows 12–14).** A verified payer proof is a *named receipt*: it certifies simultaneously that the honest issuer signed the disclosed invoice content, that the named payer key requested it, and that the preimage of the paid invoice's hash was released (Theorem 10). Its privacy hangs on elision: when the payer's request carries `invreq_paths` (payer-supplied reply paths, spec-legal [B12:808–809]), a full-disclosure proof names their introduction node in the clear (Lemma 1); a minimum-disclosure proof is machine-checked unlinkable to it, in single-session and replicated forms (Theorem 11); and the specification's leaf-nonce construction is load-bearing—remove the nonce and even minimum disclosure breaks (Lemma 5), as does payer-key reuse (Lemma 6).
- **Two boundary pairs with attack consequences (rows 5–8).** Invoice authentication is *exactly* as strong as the offer distribution channel: the correspondence holds under an authenticated channel (Theorem 5), while under a public channel a Dolev-Yao adversary executes full payment redirection (Attack 1)—sharpening, and in the adversarial setting contradicting, the specification's recorded justification for unsigned offers [B12:340–343], which addresses accidental errors. And although `invoice_error` carries no signature and its reader semantics are literally `FIXME!` [B12:895–896], the error envelope inherits authenticity from the reply path's end-to-end encryption exactly when the arrival binding [B12:813] is enforced (Theorem 6); the deviation mutant spoofs (Attack 2), localizing spoofability to the un-enforced binding.
- **Methodology (Section 6).** To our knowledge (search protocol, Appendix D), the first machine-checked equivalence for an onion-routed protocol with *proactive* blinding, under a route-blinding equational theory of two-directional fidelity (Theorem 1); a falsifiability-mutant discipline (every encoding ships with a sanity mutant that must break); correspondence vacuity control—an audit that caught a vacuity in our own earlier artifact; and compositional modeling that converges in seconds where full-protocol equivalence diverges [7].

The architecture survives its first formal examination: every protocol-level attack we exhibit lands on an assumption the specification already leans on out-of-band, not on a broken mechanism, and the six recommendations of Section 8 turn each soft spot into a testable specification statement.

Roadmap. Section 2 fixes notation, pins the specification, and frames the lifecycle; Section 3 positions the work; Section 4 defines the symbolic model and blinding theory; Section 5 walks the binding map row by row; Section 6 discusses methodology; Section 7 registers limitations.

2 Background: BOLT 12 and Route Blinding

The offer flow. A merchant (the *offer node*) publishes an *offer*: a bech32 string (prefix `lno`) carrying TLV fields—amount, description, validity, and either a long-term node key (`offer_issuer_id`) or one or more *blinded paths*. Offers are intentionally unsigned; the specification's rationale is that "a signature is unnecessary. . . if the offer has an error, no invoice will be given since the request includes all the non-signature fields" [B12:340–343][7] That statement concerns *accidental* errors; its adversarial counterpart we examine in Section 5. To pay, a wallet (the *payer*) constructs an `invoice_request` mirroring the offer's non-signature fields, adds a *transient* signing key (`invreq_payer_id`, fresh per request [B12:476–477]), signs under it, and sends the request as an onion message along the offer's blinded path. The offer node returns an *invoice* mirroring the request, carrying a payment hash, and signed under `invoice_node_id`—the final *blinded* node id when no `offer_issuer_id` was published [B12:733–736]; the invoice must return along the reply path supplied in the request [B12:813]. Signatures are BIP-340 Schnorr [17] over the Merkle root of the TLV stream. Error replies (`invoice_error`) carry *no* signature [B12:862–896]; the spec's reader requirements for them are literally `FIXME!` [B12:895–896].

Route blinding. A blinded path [B4:444–543] is created by a node that samples a blinding secret chain $e_0, \ldots, e_r$ (each e_{i+1} derived from E_i and the per-hop secret), publishes $E_0 = e_0 \cdot G$ and, for each path node $N_i = k_i \cdot G$, a *blinded node id* $B_i = \mathrm{HMAC}(\text{"blinded_node_id"}, ss_i) \cdot N_i$ with $ss_i = H(e_i \cdot N_i)$. Each path node derives its blinded private key from its static key and the incoming path key; no other party can link B_i to N_i. The first node id and the hop count are *public* (dummy padding hops are a MAY [B4:515]). Onion messages are forwarded over these paths with no authentication and no delivery guarantees [B4:1715–1717].

Payer proofs. Added to the specification in July 2026 [B12:910–1098], a *payer proof* (`lnp`. . .) is a receipt of payment the payer can exhibit to third parties. It is built on the invoice's TLV stream: mandatory fields in clear (the payer key `invreq_payer_id`, the payment hash, the signing `invoice_node_id`, the invoice signature), the payment *preimage* (`proof_preimage`, mandatory, since the normal Lightning proof of payment demonstrates only *that* an invoice was paid, not by whom [B12:62–66]); and, for every further field, either the clear value or a marker plus the minimal Merkle hashes that keep the invoice signature verifiable. Each included field also carries a *leaf nonce* $H(\text{"LnNonce"} \,\|\, T_0 \,\|\, \mathit{type})$ derived from the requester's secret `invreq_metadata` (T_0), which the specification bans from the proof ("that is the hashing nonce, thus allowing brute-force of omitted fields"); a `proof_signature` under the transient payer key then binds the payer to the receipt. The rationale states that some fields "may compromise privacy (such as `invreq_payer_note`. . .)" and may be elided, but gives no criterion for which elisions suffice; rows 12–14 draw that boundary.

[7] Bracket notation `[B12:n]` refers to line numbers of the pinned BOLT 12 snapshot (Section 2); `[B4:n]` likewise.

Table 1. Closest prior work.

Work	Object	Method	Differentiation
[3]	BOLT 1/2/3/5	Why3	channels only; funds safety
[4]	payment protocol	TLA+	channels only; balance safety
[22]	BOLT 2/5 (congestion)	Spin	channel layer; no offers
[5]	onion messages	attack+mitigation	transport layer; no offer semantics, no symbolic equivalence
[6]	blinded-path routing	design+empirical	no BOLT 12 formalization
[7]	smartcard payment	manual bisim.+PV	no onion routing; divergence data
[2]	onion routing	definitions	predates proactive blinding
[8]	EMV contactless	game-based	not symbolic; different protocol
[10]	unlinkability defs.	appl. pi equivalence	definitions; case studies; no blinding
[11]	PCN privacy	defns+impossibility	relationship below

Notation. [B12 : n] and [B4 : n] refer to line numbers of the pinned snapshots `e20a924/1528972` (archived in the artifact). *Version pin:* we model BOLT 12 as of `e20a924` (2026-07-27, including Payer Proofs) and BOLT 4 as of `1528972` (2026-08-26); the Payer Proofs flow is modeled in rows 12–14.

Specification archaeology. Two bookkeeping observations from pinning: the canonical specification file has always been `12-offer-encoding.md`—the community filename `12-offers.md` never existed in the repository, which silently breaks literature searches that use it (ours included, initially); and the draft-era fields `offer_id` and `blinded_xor`, which still circulate in informal descriptions, do not exist in the current specification.

The lifecycle and its bindings. This paper covers the offer family end to end; Figure 1 shows the message flow and where each of the seven bindings of Table 2 acts; color encodes how each is enforced.

3 Related Work

The formal-methods line on Lightning covers channel funds safety only [3,4] (Table 1); the offer family is unanalyzed. At the transport layer, the short paper of Bashiri and Khabbazian [5] analyzes onion-message anonymity against path-aborting adversaries and proposes "leashes" (blinding-point restrictions). It is the closest prior analysis of BOLT 4 onion messages, and complementary to ours: it targets the transport, not the offer family's semantics, uses attack-and-mitigation rather than symbolic equivalence, and does not treat offers, invoices, or receipts. Corcoran and Spasić generalize blinded paths for path-planning with

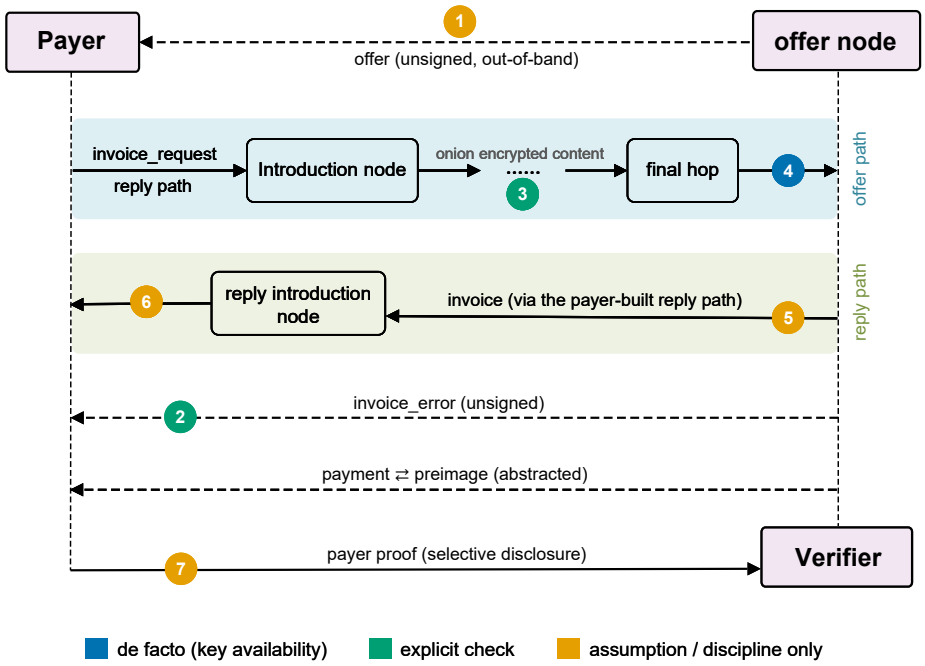

Fig. 1. The offer family's lifecycle and its seven bindings. Numbered tags mark where each binding (Table 2) acts: **1** offer distribution, **2** error arrival, **3** path content, **4** path transit, **5** reply-exit origin, **6** reply delivery, **7** receipt disclosure. Color encodes enforcement, and each status maps to machine-checked verdicts: **blue** = enforced de facto by key availability (the blinding chain, binding 4); **green** = enforced by an explicit, specification-consistent check whose sufficiency is machine-verified (bindings 2, 3); **orange** = assumed out-of-band or, where a check exists, left to implementation discipline by the specification (bindings 1, 5, 6, 7). The invoice returns over the payer-built reply path—its own introduction node, a blinded route separate from the offer's blinded path; dashed = out-of-band, unsigned, or abstracted traffic (offer, error, payment).

an empirical evaluation [6] but do not formalize BOLT 12 or prove unlinkability. Arapinis, Chothia, Ritter, and Ryan defined unlinkability and anonymity via observational equivalence in the applied pi calculus [10], and Mauw et al. formalized anonymity of classical onion routing [2]; our blinding equational theory and identity anchoring extend that program to route blinding. Malavolta et al. define relationship anonymity and value-locking for PCNs and prove an impossibility for multi-hop locks [11]. Our guarantees concern the *recipient side* of a single logical hop (offer→invoice over a blinded path) and are complementary: their impossibility lives in multi-hop concurrency, ours in the offer family's own fields and bindings. Machine-checked unlinkability methodology is due to Blanchet's ProVerif line [19,20] over the applied pi calculus [18]; the desynchronized-biprocess technique [21] is the natural tool for our machine-inconclusive cases (Section 6). The

UTX analysis of a deployed payment protocol [7] is our closest methodological precedent, including its documentation that full-protocol equivalence diverges in three tools—the calibration behind our compositional strategy. Empirical Lightning privacy work [12,13,14] measures the timing/topology channels we register as out of symbolic scope. The probing attack family has an implementer record (LDK issues #3117 and #2688) and a classical counterpart in Tor guard discovery [9]; our contribution is the first formal treatment, the content-vs-transit boundary, and the verified hardening. Beyond Lightning, the same feedback loop has reached TLS: the ECH privacy attacks were "accounted for" in the next draft [23], and the working group's formal-analysis triage now gates last calls; Payer Proofs landed in BOLT 12 with no such analysis—this paper supplies it, recommendations included (Section 8). For the receipt plane, the auditable-payments literature *constructs* receipt-bearing schemes (e.g., AQQUA [24]) and receipt aggregation has been formalized with ideal functionalities [25]; the native receipt mechanism of a deployed protocol had not been analyzed.

4 Symbolic Model

We work in the applied pi calculus [18] as implemented by ProVerif 2.05 [19]. Public-key encryption models per-hop onion layers; signatures are message-recovery style (conservative for our queries: every signed tuple is available in clear to the verifier inside the onion). The core of the encoding is the *blinding equation*:

$$\mathrm{blind}(e, \mathrm{pk}(k)) \;=\; \mathrm{pk}(\mathrm{blindsk}(e, k)). \tag{1}$$

Theorem 1 (Fidelity of the blinding theory). *Equation* (1) *is faithful in both directions: (i) every derivation the honest parties perform in BOLT 4 is an instance of the constructor or equation side; (ii) no derivation available to the symbolic attacker yields* e, k, *or* $\mathrm{blindsk}(e, k)$ *from public inputs* N, $E = e \cdot G$, *and* B *(no destructor exists; recombination from publics is not derivable)—the symbolic counterpart of CDH-hardness of the HMAC-over-ECDH construction.*

Proof sketch. (i) By inspection of [B4:493–543]: the creator computes B_i from e_i and N_i; the node derives its blinded private key from its static key and the incoming path key—both are constructor/equation instances. (ii) The equational theory presents no destructor for blind or blindsk; terms over public inputs are constructor combinations, which the mutants F1/F2 witness to be leak-detecting whenever a real leak exists (identity substitution is caught). □

The mutants F1/F2/F1c (below) witness Theorem 1(ii) from the leak-detection side: identity substitutions detectable in reality are detectable in the encoding.

Identity anchoring (non-vacuity). Candidate payee nodes carry static keys announced with *distinct public aliases* (gossip), and distinct offers carry *distinct public tags* (amounts). Anchoring closes both symmetry axes along which an equivalence can be vacuous: no renaming of static keys (aliases are world-invariant

constants) and no renaming of ephemeral blinding keys (tags bind them to public positions) maps one world onto the other. We developed this discipline after our first baseline proved vacuously true (Section 6).

Adversary. The Dolev-Yao adversary controls the public channel, forges anything unprotected, and *is* the corrupted introduction node or middle hop in the corresponding variants (its forwarding key is a free name). Corrupt-hop capability and view: it peels exactly its onion layer, sees the opaque inner ciphertext and the addressee blinded id B (public anyway), knows its predecessor and successor, and can drop, delay, replay, and inject; it cannot read inner layers or link B to any node. Intermediate-hop blinding derivation is modeled as direct key possession (sound: it grants the hop exactly its real power; the equation is exercised at the final node).

Core encoding (excerpt). Listing 1 shows the frozen base that every model in the artifact extends.

Listing 1. The frozen core encoding.

```
fun blind(skey, pkey): pkey. fun blindsk(skey, skey): skey.
equation forall e: skey, k: skey; blind(e, pk(k)) = pk(blindsk(e, k)).
let creator(e: skey, kc: skey, B: pkey, amto: bitstring) =
  out(c, (amto, pk(e), B)); (* unsigned offer *)
  in(c, x: bitstring);
  let (invreq: bitstring, Bp: pkey) = adec(x, blindsk(e, kc)) in
  let (meta,a,Bm,vkp,sigp) = invreq in
  if Bm = B then if a = amto then (* mirror check *)
  let m = open(sigp, vkp) in if m = (meta,a,Bm,vkp) then
  new ph: bitstring; let invoice = ((meta,a,Bm,vkp),ph,amto) in
  out(c, aenc((invoice, sign(invoice, blindsk(e, kc))), Bp)).
let payerP(B: pkey, amto: bitstring, Intro: pkey) =
  new kp; new bp; new meta; new tok;
  let invreq = (meta,amto,B,pk(kp),
              sign((meta,amto,B,pk(kp)),kp)) in
  out(c, aenc((tok, (* onion to intro *)
        aenc((invreq, pk(bp)), B)), Intro));
  in(c, env); let (f,sig) = adec(env,bp) in
  let m = open(sig, B) in if m = f then event PayerAccepts(f, B).
```

Error semantics (rows 7–8): the payer parses three cases—valid invoice (accept), well-formed error envelope (abort), anything non-verifiable (silent drop); this is the conservative reading of the `FIXME!` reader semantics, and row 7's theorem is stated relative to it. The receipt plane (rows 12–14) adds three one-way constructors: leaf commitments $lh(\mathit{field}, \mathit{nonce})$, and leaf nonces $\mathit{nonce}_M, \mathit{nonce}_O$—one-way functions of the requester's secret T_0; the invoice signature covers the commitment pair (the Merkle root), and exhibited proofs carry the nonces for included fields so that verifiers recompute the leaves. The full abstraction table (modeled-faithful vs. abstracted, per-entry rationale) is in Appendix A.

Table 2. The binding map: seven assumptions carry the offer family's privacy and integrity. Five removals land as machine-checked attack traces—bindings 1, 2, 3, 4, 6 (rows 6, 8, 18, 10, 15); binding 7's removal consequence is machine-inconclusive with a manual lemma (row 13) while its elision defense is machine-checked (row 14); binding 3 additionally carries its enforced defense machine-checked (row 17); binding 5 is soft by design with a verified hardening (row 11).

#	Binding	Removed ⇒	Enforcement
1	offer distribution channel	payment redirection (row 6)	out-of-band
2	error arrival [B12:813]	error spoofing (row 8)	enforce binding (row 7)
3	path *content*	fresh-route probe (row 18)	check (row 17)
4	path *transit*	direct-delivery probe (row 10)	chain, de facto (row 9)
5	reply-exit origin	origin observability (soft)	hardening (row 11)
6	reply *delivery*	cross-path delivery (row 15)	payer check only (row 16)
7	receipt disclosure policy	payer attribution (row 13)	elision + nonces (row 14)

5 Results: The Binding Map

The eighteen rows below are best read as a map of assumptions: the offer family's privacy and integrity rest on seven *bindings* (Table 2), each a removable assumption whose removal consequence one or two rows machine-check. Read as assumptions rather than channels, the rows expose a structural asymmetry between the two sides of the protocol. The blinding key chain—designed for unlinkability—authenticates recipient-side transit de facto, by key availability: the arrival binding the specification demands [B12:521–524] but never shows how to test is enforced for free (Theorem 7). The reply's delivery binding [B12:808–813] gets no such help: the specification's MUST references no test, the natural test sits unused in an optional BOLT 4 mechanism (`path_id`, [B4:1793]) that BOLT 12 never cites, and the gap is exploitable (Attack 5). The subsections walk the rows in evidence order; Figure 1 fixes the lifecycle stages.

Definition 1 (Receiver anonymity). *Fix two candidate nodes with static keys k_1, k_2 announced under distinct public aliases. The* anchored worlds *differ only in which key backs an anonymous offer's blinded id: creator*$(\mathrm{choice}[k_1, k_2])$ *versus creator*$(\mathrm{choice}[k_2, k_1])$ *in ProVerif biprocess notation, with announcements and (in the replicated variants) both role processes under replication. Receiver anonymity holds iff the biprocess is observationally equivalent.*

Definition 2 (Payer-payee unlinkability). *With distinct public tags $amt_1 \neq amt_2$ and payers bound by process parameter to offers B_1 resp. B_2 (pairings fixed across worlds; both role families replicated), where $B_1 = \mathrm{blind}(e_1, \mathrm{pk}(\mathrm{choice}[k_1, k_2]))$ and $B_2 = \mathrm{blind}(e_2, \mathrm{pk}(\mathrm{choice}[k_2, k_1]))$, payer-payee unlinkability holds iff the biprocess is observationally equivalent.*

Theorem 2 (Receiver anonymity, unbounded sessions). *In the symbolic model of Section 4, with the introduction node distinct from the candidate payees*

Table 3. Leak-channel table (result rows; — continues the plane above). Verdicts: MC = machine-checked; MI = machine-inconclusive with manual lemma (ProVerif 2.05 emits no certified-false for equivalence); trace = machine-checked attack trace. Model IDs map to the artifact (Appendix B); the sanity-mutant suite and registered non-symbolic channels follow in Table 4.

#	Plane	Variant (model)	Verdict	Reading
1	blinded id	anchored swap, repl. (M3)	MC true	Thm. 2
2	2×2 swap	tagged swap, repl. (M4); +corrupted intro (M7)	MC true	Thm. 3
3	blinded id	corrupted intro, repl. (M5)	MC true	Thm. 4, M5 case
4	blinded id	corrupted middle hop, repl. (M6)	MC true	Thm. 4, M6 case
5	invoice auth.	offer via auth. channel (M2b1)	MC true	Thm. 5
6	offer unsigned	offer via public channel (M2b2)	trace	Attack 1
7	`invoice_error`	spec-faithful (M2d-v2)	MC true	Thm. 6
8	—	deviation: unbound errors (M2d-dev)	trace	Attack 2
9	transit plane	chained keys, direct-delivery adversary (M8e)	MC true	Thm. 7
10	—	sanity mutant: final-hop key public (M8c)	trace	Attack 3
11	reply exit	reverse-path hardening (M8d)	MC true	Thm. 9
12	receipt plane	payer-proof authentication (M10a)	MC true	Thm. 10
13	—	full disclosure: `invreq_paths` in (M10c-full)	MI break	Lemma 1
14	—	minimum disclosure: elided (M10c-min)	MC true	Thm. 11
15	delivery binding	deviation: no arrival check (M9a)	trace	Attack 5
16	—	spec-faithful: binding enforced (M9b)	MC true	Thm. 12
17	path content	spec-faithful: check enforced (M11a)	MC true	Thm. 8
18	—	removal: fresh-route probe serviced (M11b)	trace	Attack 4

and at least one blinded hop beyond it, receiver anonymity (Definition 1) holds under unbounded sessions of both roles (M3).

Table 4. Sanity-mutant suite (each introduces a known leak and must break) and registered non-symbolic channels.

ID	Mutant / channel	Verdict	Reading
F0	issuer_id exposure, 1×1 (M1neg)	MI break	Lemma 2
F1	identity instead of blinding (F1)	MI break	Lemma 3
F2	issuer_id in 2×2 (F2)	MI break	Lemma 4
F1c	F1 under corruption+repl. (F1c)	MI break	Lemma 3
F3	deterministic leaf hashes (M10c-F2)	MI break	Lemma 5
F4	payer-key reuse (M10b-F1)	MI break	Lemma 6
R1	description plaintext	registered	[16]
R2	hop count/first node	registered	public by constr.
R3	latency/size	registered	[13,14,12]

Theorem 3 (Payer-payee unlinkability). *Payer-payee unlinkability (Definition 2) holds under unbounded sessions with an honest introduction relay (M4), and under unbounded sessions against a corrupted introduction node (M7). Offer-level linkability (which offer a payer queries) is by design and not covered.*

Theorem 4 (Corrupted-hop anonymity). *Receiver anonymity is preserved under unbounded sessions when the adversary occupies the introduction node (M5) or a middle hop of a three-node path (M6)—the strongest on-path positions short of the final node.*

Theorem 5 (Invoice authentication). *Under an authenticated offer distribution channel, the correspondence*

$$\texttt{query}\ x, y;\ event(PayerAccepts(x, y)) \Rightarrow event(Issued(x, y))$$

is machine-checked true (M2b1): every invoice the payer accepts under the offer's blinded key was issued under that key.

Attack 1 (Payment redirection) *Under a public offer channel (M2b2), the Dolev-Yao adversary: (1) publishes a forged offer naming its own key as introduction and its own blinded id; (2) receives the payer's onion and peels the outer layer (it is the intro); (3) peels the inner layer (it holds the blinded secret); (4) reads the mirrored invreq and the payer's reply key; (5) returns a self-signed invoice wrapped for the reply key; (6) the payer verifies it under the forged offer's blinded key and accepts—payment is redirected to the attacker. The attack sharpens the spec's recorded rationale for unsigned offers [B12:340–343], which addresses accidental errors: under adversarial substitution an invoice* does *issue and verify.*

Theorem 6 (Error authenticity by construction). *In the spec-faithful model, where the error envelope travels inside the reply path's end-to-end encryption and the payer enforces the arrival binding [B12:813] for envelopes (three-case parsing, conservative `FIXME!` reading), the error-honesty correspondence below is machine-checked true (M2d-v2):*

$$event(PayerAborted) \Rightarrow event(NodeSentError)$$

The unsigned `invoice_error` fields inherit authenticity from the reply-path encryption.

Attack 2 (Error spoofing under deviation) *In the deviation mutant (M2d-dev) where the payer additionally processes a plaintext well-formed error message—implementations that skip the arrival binding—the adversary outputs a syntactically valid `invoice_error`, and the payer aborts although the node sent nothing (machine-checked trace). The deviation class is plausible—implementation-level BOLT 12 bugs are documented (e.g., the LDK denial-of-service fix for crafted `Bolt12Offers`[8]), though no public implementation is known to skip the arrival binding specifically.*

Probing: content, transit, and the reply exit (rows 9–11, 17–18). The specification's route-blinding rationale [B4:569–578] contemplates the probing adversary ("a malicious sender could create different blinded routes to all the nodes that they suspect could be the real recipient and try them until one accepts") and prescribes the `path_id` context check [B4:545]; implementers responded to a probing report ("Blinded Paths are probe-able", LDK issue #3117, 2024) by binding the offer's authentication nonce into the blinded path (LDK PR #3139). These are *content* defenses, and rows 17–18 machine-check that they carry the weight: with the check, the fresh-route probe is rejected (Theorem 8); without it, the prober's probe is serviced and the answering node identifies the receiver (Attack 4). What they do not even attempt is authenticating *transit*—and here the specification demands more than it explains: [B12:521–524] requires ignoring requests that did not arrive via the offer's paths, yet prescribes no test. Our first model assumed the missing test was really missing: it keyed the final hop on the *public* first path key, and a verbatim-transit probe then broke receiver anonymity (Attack 3)—reported below as the sanity mutant for the theorem that follows. A closer reading of the chain construction refuted that attack, and the refutation is the finding:

Attack 3 (Sanity mutant: final-hop key public) *In the variant that publishes the* final-hop *path key (as our first model did, conflating it with the public first path key), the adversary (1) copies the path key, the final-hop blob, and the blinded id verbatim from the offer; (2) encrypts its own `invoice_request` to the*

[8] LDK security fix in the `rust-lightning` 0.1.3 release notes (`https://github.com/lightningdevkit/rust-lightning`); CLN behavior per `fetchinvoice(7)`/`invoicerequest(7)` (`https://docs.corelightning.org`).

blinded id; (3) delivers the one-hop final onion directly to each suspect node. Every content check passes; the suspect that services the request reveals itself by the origin of its response: machine-checked trace (M8c, with the attribution control M8c-v2 isolating the break to the direct-delivery branch)—receiver anonymity is broken in this variant, while the faithful chain of row 9 resists the same probe.

Theorem 7 (Transit authentication by the chain). *With the chain faithful to [B4:502–505,532,541]—the final hop's incoming key E_1 derived from the introduction node's secret, never published; only the first path key public—receiver anonymity holds against the direct-delivery adversary: observational equivalence is machine-checked true (M8e), with genuine transit-routed traffic serviced throughout. The arrival binding of [B12:521–524], which has no message-layer test, is enforced de facto by key availability. Attack 3 is the sanity mutant: publish the final-hop key and the same encoding breaks.*

Theorem 8 (Content check rejects fresh-route probing). *With the path-content check modeled as the recipient-bound nonce context and the transit chain faithful throughout, receiver anonymity holds against the fresh-route prober: observational equivalence is machine-checked true (M11a), with genuine transit-routed traffic serviced. The prober's maximal delivery capability is granted (fresh routes modeled as direct-to-endpoint delivery); only the content check turns it away. The check is keyed under the prober-supplied incoming key, so the barrier is the content credential the prober lacks rather than the offer's own path key.*

Attack 4 (Fresh-route probe under removal) *Remove the content check and the probing adversary of the route-blinding rationale [B4:569–578] succeeds: machine-checked trace (M11b). The prober mirrors an offer's blinded id in probes delivered to both anchored candidates; only the offer's owner services the probe and answers on its own channel—the answering channel identifies the receiver. The equivalence-form encoding cannot express this attack (the choice-symmetric structure masks it; the attempt is archived in the artifact), so we carry the consequence as a reachability trace.*

What the chain does not protect is the reply exit: the response travels to the requester-chosen `first_node_id`, so its origin is observable to that node and, depending on the implementation's routing, on-path—an honest-flow, implementation-dependent leak (the payment-phase cousin is LDK #2688). The hardening is on the reply side:

Theorem 9 (Reverse-path hardening of the reply exit). *If the responder returns the invoice through its* own *introduction node (reverse-path reply), the reply's observable origin becomes the offer's introduction node—already public; with final-hop acceptance additionally bound to the introduction-node peer, observational equivalence is machine-checked true (M8d). This is defense in depth for the one plane the chain leaves soft: not a repair of a broken protocol, but a verified hardening of the reply exit.*

The attack family is implementer-known (LDK #3117); the content/transit boundary and the machine-checked hardening are new. The classical counterpart is Tor guard discovery [9], whose deployed answer—the Vanguards defense—dedicates fixed, purpose-chosen guards to the anonymity-critical connection roles; our reverse-path hardening is the same principle transplanted to BOLT 12 (dedicate the reply exit to the already-public introduction node) and verified mechanically for this protocol.

The receipt plane: authentication and the disclosure boundary (rows 12–14). After a payment completes, the payer can exhibit a payer proof (Section 2). Rows 13 and F4 confirm that the specification's own warnings are tight; rows 14 and F3 draw the boundary the specification never states. The authentication core is a three-way correspondence:

Theorem 10 (Named-receipt authentication). *Whenever an honest verifier accepts a payer proof—preimage matching the payment hash, the issuer's invoice signature over the Merkle commitments, the payer's proof signature—the conjunction below is machine-checked true (M10a):*

$$event(ProofVerified) \Rightarrow event(Issued) \wedge event(Requested) \wedge event(Paid).$$

A single artifact gives both sides non-repudiation: the issuer signed the disclosed invoice content under the offer's blinded key, the payer key named in the proof signed the corresponding request, and the payment completed (payment abstracted at preimage release; Appendix A). The Requested *event fires when the issuer validates the invreq signature, so the theorem certifies exactly what the two signatures certify—key possession on both ends.*

What the receipt *discloses* is a privacy question the specification leaves open—its rationale warns that some fields are not useful and one "may compromise privacy" [B12:1086] without delimiting which disclosures are safe. The privacy-critical optional field family is the mirrored `invreq_paths`: *payer-supplied* blinded paths for the invoice's return, a specification-legal flow with its own reader branch [B12:808–809], whose first node id (the payer's chosen introduction node) is public by construction. (In the pure offer flow, the reply path is an onion-layer object that never enters the invoice TLV stream, so such proofs carry no path field at all—the disclosure question arises exactly when the payer uses `invreq_paths`.) Including them names the payer's network neighborhood; a verifier colluding with introduction nodes ties the receipt to the payer adjacent to the named node:

Lemma 1 (Full disclosure names the reply-path introducer). *If the exhibited proof carries the mirrored* `invreq_paths` *identity core in the clear, the two worlds (receipt of a payer whose* `invreq_paths` *enter via introduction node I_A vs. I_B) are not observationally equivalent: the proofs contain the public values $id(I_A)$ resp. $id(I_B)$, and a syntactic test against the announced introduction ids distinguishes them (M10c-full; machine-inconclusive under ProVerif's no-certified-false equivalence, carried by this lemma).* □

Theorem 11 (Minimum disclosure unlinks). *With the* `invreq_paths` *field elided per the specification's selective disclosure, the exhibited payer proof is machine-checked observationally equivalent across the two worlds, in single-session and replicated (unbounded-session) forms (M10c-min): every published component is fresh-random (payment hash, transient payer key, preimage, the T_0-keyed leaf commitments) or symmetric (the offer's blinded id, the amount). The elision is exactly what stands between a colluding verifier-and-introducers pair and payer attribution of the receipt.*

The boundary is not an artifact of generous cryptography: it rests on the leaf-nonce construction, and the mutant that removes it (deterministic leaf hashes) breaks even minimum disclosure (Lemma 5); likewise, reusing the transient payer key across payments links two exhibited receipts (Lemma 6). The specification's own design choices—T_0-keyed nonces with T_0 banned from the proof, single-use payer keys—are load-bearing for the receipt's unlinkability; the mutants machine-check the negative side, and the fresh-key positive direction is carried by Theorem 11's equivalence itself.

Delivery binding of the reply (rows 15–16). The recipient side of transit is authenticated de facto by the blinding chain (Theorem 7). The payer side of the *reply* is not. The specification demands that the payer reject an invoice "if it did not arrive via invoice request `onionmsg_tlv reply_path`" [B12:808–813], an arrival binding whose MUST references no test. The natural test exists one specification away, and only as an option: BOLT 4 lets a reply-path publisher store a `path_id` secret in the final payload "so it can recognize use of this `reply_path`" (a MAY, [B4:1793,1815–1832]), the same `path_id` context check the offer side prescribes ([B4:545])—but BOLT 12's binding MUST never cites it, and the deployed fix binds instead via the payment identifier in the payer metadata. Nothing in the invoice's content identifies the channel it traveled: the signatures bind the issuer and the mirrored fields, not the reply path.

Attack 5 (Cross-path invoice delivery) *One payer holds two outstanding payments to the same issuer, each with its own reply path through its own (corrupted, attacker-relayed) introduction node. The Dolev-Yao adversary listens on payment A's reply path, captures A's invoice onion—an opaque end-to-end blob—and replays it onto payment B's reply path. The payer without an arrival check decrypts it (both endpoint keys are its own, per-path derived from its static key) and accepts the mis-delivered invoice into the wrong payment context: machine-checked trace (M9a). This is the surface LDK fixed in June 2026 by checking the payment identifier recovered from the invoice's payer metadata against the arrival path's context, "preventing an attacker from reusing one payment's blinded path to deliver another's invoice and correlate the two as ours."*[9]

Theorem 12 (Delivery binding). *With the arrival binding enforced—the payload's path tag must equal the arrival channel's own path tag and the invoice fields*

[9] LDK PR #4685, merged 2026-06-15 (`github.com/lightningdevkit/rust-lightning`).

must match that tag's payment context (the `path_id` *mechanism of [B4:1793] made mandatory; the payment-identifier semantics of the LDK fix)—the correspondence event*$(PayerAccepts(t)) \Rightarrow$ *event*$(OwnDelivery(t))$ *is machine-checked true (M9b) against the fully relaying, replaying network of Attack 5. The payer model tries every (arrival channel, endpoint key) pair, so the check is the only separator between own-path and cross-path acceptance.*

The chain gives the recipient side delivery authentication for free; the reply path's own cryptography gives the payer side nothing of the kind—whoever observes its structure can deliver over it. Our model's public delivery channels grant the adversary both corrupted introductions and full relay control, so the theorem holds under the stronger adversary.

Reconciliation. Every table row maps to one artifact model identifier (Appendix B); wall times and encoding sizes are tabulated there.

6 Methodology Notes

Vacuity control. Equivalence queries are vacuously true whenever the two worlds are related by a renaming of restricted names—an easy trap. The standard remedy applies: anchor identity on both the static-key and ephemeral axes (Section 4), and pair each encoding with a sanity mutant that introduces a known leak and must break. Correspondence queries carry the dual hazard that the concluding event may be unreachable, making the implication vacuously true; every correspondence model therefore ships with an event-reachability query (this discipline was added after an audit found our own invoice-authentication model (M2b1) had left the payer role unspawned; the repaired model verifies non-vacuously). ProVerif's equivalence procedure emits no certified-false verdicts; our negative rows are machine-inconclusive and are carried by written frame-distinguisher lemmas (Appendix C). Two further encoding lessons from this study: biprocess *late choices* (publishing $choice[x_1, x_2]$ over values received on private channels) sit in a blind spot of ProVerif's equivalence procedure and had to be restructured into in-process choices (the anchored-swap idiom); and abstraction fidelity cuts both ways: an early payer-proof encoding omitted the leaf nonces from the exhibited proof, which admitted a fake-fields-with-genuine-signature combination impossible in the real construction (the Merkle root binds leaves, leaves bind nonce and field).

Compositional modeling. Minimum core first, incremental fields, one property per sub-model—calibrated against the UTX report that full-protocol equivalence diverged in three tools [7]. Our core converged in seconds (wall times above); every extension is a small delta on a frozen base, which is also what made the same-day remediation of two review-caught encoding defects possible. On second-tool triangulation: we rebuilt the desynchronized-biprocess tool of [21] and verified it on its own examples; its front-end accepts an equation-based dialect without destructor rules, events, or choice macros, so porting our encodings would be a re-encoding exercise; we leave it as future work.

7 Discussion and Limitations

Theorems hold in the symbolic model with the abstractions of Appendix A. Registered but unmodeled: description plaintext (R1), hop-count and first-node exposure (public by construction; dummy hops are a MAY [B4:515]), timing/-size channels (R3; the empirical literature [13,14,12] shows they can re-identify endpoints—our guarantees are conditional on those channels being inert). For the receipt plane, the payment phase is abstracted (preimage release modeled at settlement): how preimages propagate through the payment route to path observers, and multi-offer/issuer-side receipt scenarios, are out of scope; invoice caching remains a registered tension (metadata uniqueness [B12:494] vs. the caching MAY [B12:515–518]—a replayed request yields a byte-identical cached invoice, a liveness token), of which rows 15–16 formalize the delivery side. Further future work: async payments (unmerged PR #1149), multi-hop collusion, and a destructor-based blinding theory expressing a colluding node's recognition test. The artifact (twenty-two primary models with transcripts, the sanity-mutant suite, pinned spec snapshots, and run instructions) will be made available in anonymized form.

8 Conclusion

The outcome is, on balance, reassuring: within its symbolic scope, BOLT 12's architecture holds. The properties that motivate blinded paths are machine-checked under unbounded sessions and against the strongest single-position on-path corruption (Theorems 2–4); the payment receipt authenticates the way its designers would want (Theorem 10); and every protocol-level attack we exhibit lands on an assumption the specification already leans on, not on a flaw in its cryptography. What the map adds is knowing where those assumptions live and what each is worth: two of them are held up by implementation discipline alone (delivery, disclosure), one is enforced for free where nobody asked for it (transit), and the specification never says which is which.

Our recommendations to the BOLTs repository and implementors:

1. enforce the arrival binding [B12:813] for *error* envelopes as well as invoices (binding 2, rows 7–8);
2. resolve the `invoice_error` reader-semantics `FIXME!` [B12:895–896] (binding 2);
3. state offer-distribution authentication as an explicit assumption in the unsigned-offer rationale [B12:340–343] (binding 1, row 6);
4. harden the reply exit by returning invoices via the responder's own introduction node, with final-hop acceptance additionally bound to that peer (the full verified condition of Theorem 9; binding 5, row 11);
5. state the delivery binding [B12:808–813] as a testable check by referencing the BOLT 4 `path_id` mechanism [B4:1793]—the payment-identifier/path-context test implementations already apply (binding 6, rows 15–16);

6. for payer proofs, state the elision boundary—machine-checked for `invreq_paths` (rows 13–14), extended by analogy to `invoice_paths`—and keep the leaf-nonce and single-use-key requirements normative (binding 7).

For future BOLT 12 changes, the map plays the role that formal-analysis triage now plays for TLS: an analysis to cite before the next mechanism lands.

A Abstraction Table

Modeled faithfully: unsigned public offers (attacker-forgeable by design); invreq mirroring of amount and blinded id with creator-side check; transient per-request payer keys; invoice signature under the final blinded key absent `offer_issuer_id`; amount equality; invoice return via the reply path (end-to-end wrapping for the payer's reply key); per-hop onion peeling with opaque routing tokens; three-case error parsing. For the receipt plane (rows 12–14): the mandatory proof fields in clear; the leaf-nonce construction (commitments keyed by one-way functions of the requester's T_0, nonces exhibited for included fields so verifiers recompute leaves—[B12:1046–1048]); the proof signature under the transient payer key; the preimage/hash match. For the delivery plane (rows 15–16): per-path reply keys derived from one static payer key over distinct blinding epochs (the payer can decrypt either payment's payload on either channel); the payer-embedded path tag mirrored in the invoice onion's final payload; corrupted-introduction delivery as public channels. *Abstracted, with rationale:* authenticated offer distribution (M2b1) as a private channel—models out-of-band delivery (in-person QR, HTTPS), whose absence is exactly what row 6 isolates; Merkle root $\rightarrow$ sign the field commitments (auth/privacy queries depend only on coverage); the payment phase $\rightarrow$ preimage released to the payer after the payment event (payment-route propagation of preimages out of scope); a single optional field stands for the elidable field family (the boundary is about the elision mechanism, not field multiplicity); intermediate-hop derivation $\rightarrow$ direct key possession (grants exactly the hop's real power); message-recovery signatures (conservative); BOLT 8 links $\rightarrow$ public-channel visibility with content confined to onion layers (extra attacker power, declared); single logical hop with replication (path lengths beyond three nodes and MPP out of scope); the receipt-plane models omit the outer transport onion (symmetric omission for the disclosure query); fresh-route probes modeled as direct-to-endpoint delivery (the prober's maximal capability; the transport layer is binding 4's dimension); size/timing excluded (registered rows).

B Model Inventory

C Manual Distinguisher Lemmas

Lemma 2 (issuer-id exposure, 1×1). *In the variant whose offer contains* $\mathrm{pk}(\mathrm{choice}[k_1, k_2])$ *with* $\mathrm{pk}(k_1), \mathrm{pk}(k_2)$ *announced: the two frames differ on the offer term; the adversary tests the issuer field against the announcements; the*

Table 5. Artifact inventory: twenty-two primary models (twenty result models plus two stepping stones) and the sanity-mutant suite. Wall time: 41.7 s (M2 family incl. stepping stones) + 6.9 s (M3/M4) + < 1 s (M5/M6/M7/F1c) + 4.8 s (M8 family) + 21.1 s (M9/M10) + 3.6 s (M11)—under ninety seconds total; ∼2,500 lines across the 31 inventoried models (38 files including superseded evidence, preserved as such), each a small delta on the frozen core.

ID	Purpose	Verdict
M1a-v2	anchored baseline, single session (stepping stone)	true
M3	baseline, replicated	true
M4	2×2 swap, replicated	true
M7	2×2, corrupted intro, replicated	true
M5	corrupted intro, replicated	true
M6	corrupted middle hop, replicated	true
M2b1	invoice auth., auth. offer channel	true
M2b2	offer via public channel	attack trace
M2d-v2	error authenticity, spec-faithful	true
M2d-dev	error deviation mutant	attack trace
M8e	chained keys vs. direct delivery	true
M8c/M8c-v2	sanity mutant (final-hop key public)	attack trace
M8d	reverse-path hardening	true
M10a	payer-proof authentication	true
M10c-full	receipt disclosure, paths included	MI break
M10c-min	receipt disclosure, paths elided	true
M9a	delivery deviation (no arrival check)	attack trace
M9b	delivery binding enforced	true
M11a	content check vs. fresh-route prober	true
M11b	content-check removal (fresh-route probe)	attack trace
M2a-v2	2×2, single session (stepping stone)	true
M1neg/F0, F1, F2, F1c, M10c-F2, M10b-F1	sanity-mutant suite (identity/leaf-nonce/key reuse)	MI break
M1a, M2a	vacuity evidence (superseded)	documented
Superseded/smoke files (EXP-002 pair, M2a, M2d, M2c1/M2c2, pre-repair M2b1) preserved as evidence; their runs are included in the family wall times.		

test succeeds for different nodes in the two worlds. Hence not observationally equivalent. □

Lemma 3 (F1: identity instead of blinding). *With $B = \mathrm{pk}(k_c)$: same test as Lemma 2 against the offer's B. The argument is unchanged under corruption and replication (F1c): the corrupt hop's additional knowledge does not remove the public announcement test.* □

Lemma 4 (F2: issuer-id in the 2×2 swap). *Each tagged offer carries its backing node's static key in cleartext:* $\mathrm{pk}(\mathrm{choice}[k_1, k_2])$ *for offer 1,* $\mathrm{pk}(\mathrm{choice}[k_2, k_1])$ *for offer 2. Testing the two issuer fields against the announced keys alone distinguishes the two node assignments (a syntactic equality test on public terms).* □

Lemma 5 (F3: deterministic leaf hashes). *With the leaf commitments replaced by an unkeyed hash ($leaf_O = h(id(I))$ for the elided path field), the commitments over public introduction ids are adversary-computable: the colluding verifier evaluates h on both announced ids and matches the value carried in the minimum-disclosure proof—the two worlds of Theorem 11 become distinguishable (M10c-F2). The leaf-nonce construction (nonces keyed by the secret T_0, which the specification bans from the proof) is what blocks exactly this test.* □

Lemma 6 (F4: payer-key reuse). *If the payer signs two requests (and hence two exhibited proofs) under the same* ***invreq_payer_id*** *key, both proofs carry the same public key in clear; a pair of colluding verifiers compares the fields and links the two payments to one payer (M10b-F1). The single-use payer-key requirement [B12:476–477] is load-bearing for cross-receipt unlinkability.* □

D Search Protocol

ePrint queries (2026-09-05, re-checked 2026-09-08 and weekly through submission): "BOLT 12", "offers", "onion message", "blinded paths", "Lightning", "Lightning privacy", "discreet log contract" (adjacent line), "payer proof"/"proof of payment" (receipt line, added 2026-09-08), plus methodology-generic queries ("unlinkability ProVerif", "onion routing formal analysis", "receipt linkability selective disclosure"); arXiv equivalents; delvingbitcoin latest-index scans; DBLP cross-check for FC/ACNS 2024–2026 LN formal-analysis entries. The specification moved three times in 2026; the camera-ready will re-pin to the then-current commits and archive the corresponding snapshots.

References

1. Lightning Network developers: BOLT 12: Offer Encoding, commit `e20a924` (2026); BOLT 4: Onion Routing Protocols, commit `1528972` (2026). `https://github.com/lightning/bolts`
2. Mauw, S., Verschuren, J.H.S., de Vink, E.P.: A Formalization of Anonymity and Onion Routing. ESORICS 2004, LNCS 3193, pp. 109–124 (2004)
3. Fabiański, G., Stefański, R., Thyfronitis Litos, O.S.T.: A Formally Verified Lightning Network. FC 2025, pp. 3–20 (2025)
4. Grundmann, M., Hartenstein, H.: Security of the Lightning Network: Model Checking a Stepwise Refinement with TLA+. iFM 2025, pp. 313–335 (2025)
5. Bashiri, A., Khabbazian, M.: Short Paper: Onion Messages on Leash. FC 2024, pp. 283–292 (2024)
6. Corcoran, P., Spasić, I.: Path Planning at the Edge of Payment Channel Networks. ACM DLT 5(4), Article 39 (2026)
7. Bursuc, S., Horne, R., Mauw, S., Yurkov, S.: Provably Unlinkable Smart Card-based Payments. ACM CCS 2023, pp. 1392–1406 (2023)
8. Olivier-Anclin, C., Boureanu, I., Chen, L., Newton, C.J.P., Chothia, T., Clee, A., Kokkinis, A., Lafourcade, P.: Who Pays Whom? Anonymous EMV-Compliant Contactless Payments. USENIX Security 2025, pp. 5893–5912 (2025)

9. Biryukov, A., Pustogarov, I., Weinmann, R.-P.: Trawling for Tor Hidden Services: Detection, Measurement, Deanonymization. IEEE S&P 2013, pp. 80–94 (2013)
10. Arapinis, M., Chothia, E., Ritter, M., Ryan, M.: Analysing Unlinkability and Anonymity Using the Applied Pi Calculus. IEEE CSF 2010, pp. 107–121 (2010)
11. Malavolta, G., Moreno-Sanchez, P., Kate, A., Maffei, M., Ravi, S.: Concurrency and Privacy with Payment-Channel Networks. ACM CCS 2017, pp. 455–471 (2017)
12. Kappos, G., Yousaf, H., Piotrowska, A.M., Kanjalkar, S., Delgado-Segura, S., Miller, A., Meiklejohn, S.: An Empirical Analysis of Privacy in the Lightning Network. FC 2021, pp. 167–186 (2021)
13. Rohrer, E., Tschorsch, F.: Counting Down Thunder: Timing Attacks on Privacy in Payment Channel Networks. AFT 2020, pp. 214–227 (2020)
14. von Arx, L., Tran, K., Vanbever, L.: Revelio: A Network-Level Privacy Attack in the Lightning Network. IEEE EuroS&P 2023, pp. 942–957 (2023)
15. delvingbitcoin.org: Onion Message Jamming in the Lightning Network. Topic 2414 (2026). `https://delvingbitcoin.org/t/2414`
16. delvingbitcoin.org: E2E encryption for BOLT11/BOLT12 description/note field was this discussed before? Topic 2745 (2026). `https://delvingbitcoin.org/t/2745`
17. Wuille, P., Nick, J., Ruffing, T.: Schnorr Signatures for secp256k1 (BIP-340) (2020). `https://github.com/bitcoin/bips`
18. Abadi, M., Fournet, C.: Mobile Values, New Names, and Secure Communication. POPL 2001 (2001)
19. Blanchet, B.: Modeling and Verifying Security Protocols with the Applied Pi Calculus and ProVerif. Foundations and Trends in Privacy and Security 1(1–2), pp. 1–135 (2016)
20. Blanchet, B., Abadi, M., Fournet, C.: Automated Verification of Selected Equivalences for Security Protocols. J. Logic and Algebraic Programming 75(1), pp. 3–51 (2008)
21. Baelde, R., Debant, S., Delaune, S.: Proving Unlinkability using ProVerif through Desynchronised Bi-Processes. IEEE CSF 2023, pp. 75–90 (2023)
22. Weintraub, B., Prabhu Kumble, S., Nita-Rotaru, C., Roos, S.: Payout Races and Congested Channels: A Formal Analysis of Security in the Lightning Network. ACM CCS 2024, pp. 2562–2576 (2024)
23. Bhargavan, K., Cheval, V., Wood, C.: A Symbolic Analysis of Privacy for TLS 1.3 with Encrypted Client Hello. ACM CCS 2022 (2022)
24. Papadoulis, G., Balla, D., Grontas, P., Pagourtzis, A.: AQQUA: Augmenting Quisquis with Auditability. ACNS 2025, pp. 283–312 (2025). Full version: ePrint 2024/1181
25. Kaklamanis, I., Wang, W., Malvai, H., Yang, S., Zhang, F.: Verifiable Aggregate Receipts with Applications to User Engagement Auditing. IACR ePrint 2025/2330 (2025)