Track me if you can: Ephemeral coin tracing

Ignacio Amores-Sesar[1][0000−0002−1751−1515], Christian Cachin[2][0000−0001−8967−9213], Rohit Chatterjee[3][0009−0004−1970−1779], Luiza Soezima[1][0000−0002−4766−1506], François-Xavier Wicht[2][0009−0005−6090−7901], and Michelle Yeo[1,4][0009−0001−3676−4809]

[1] Aarhus University, Denmark
{amores-sesar,lbrsoezima}@cs.au.dk
[2] University of Bern, Switzerland
{christian.cachin,francois-xavier.wicht}@unibe.ch
[3] National University of Singapore, Singapore
rochat@nus.edu.sg
[4] Nanyang Technological University, Singapore
michellexyeo@gmail.com

Abstract. Privacy-preserving payment systems are well understood, yet concerns about their misuse for financial crime have led to only limited adoption in regulated settings such as central bank digital currencies (CBDCs) and institutional stablecoins. Tracing is one tool for addressing these concerns: acting on external evidence implicating a user, law enforcement follows the suspect's funds through the ledger to uncover laundering routes and accomplices. Existing coin-tracing schemes, however, provide no cryptographic bound on tracing reach: once initiated, a trace may propagate indefinitely through the transaction graph or persist across all future transactions of a targeted user. Keeping surveillance targeted and temporary therefore depends on the restraint of the authority or a committee.

We introduce *ephemeral coin tracing* (ECT), a primitive that bounds tracing reach by construction. Each account carries an encrypted tag that records which traced identifiers its funds carry while hiding its tracing status from users. When funds move, the sender's tag degrades and merges with the recipient's tag. Each tracing contribution expires independently after a policy-defined number of hops and then becomes unrecoverable even to the tracing authority, without affecting other live contributions in the same tag. Public parameters also bound how many identifiers a tag can distinguish simultaneously. We formalize ECT and give constructions based on exponential ElGamal, Damgård–Jurik encryption, and Ring-LWE, the last providing post-quantum security.

Keywords: Ephemeral Tracing · Coin Tracing · Transaction Privacy · Digital Payment Systems

Table of Contents

1 Introduction . . . 3
  1.1 Our contribution . . . 3
  1.2 Technical approach . . . 4
2 Related work . . . 6
3 Preliminaries . . . 7
  3.1 Cryptographic notions . . . 8
  3.2 System model . . . 8
4 Ephemeral tracing scheme . . . 10
  4.1 Characterization . . . 10
  4.2 Security . . . 12
5 Constructions . . . 15
  5.1 Approaches from first principles . . . 15
  5.2 Nilpotent degradation . . . 16
  5.3 Exponential ElGamal instantiation . . . 16
  5.4 Damgård–Jurik instantiation . . . 21
  5.5 Ring-LWE instantiation . . . 25
  5.6 Hardness of Ring-LWE over the extended ring . . . 33
  5.7 Parameter selection for the Ring-LWE instantiation . . . 36
  5.8 Efficiency and comparison . . . 38
6 Evaluation . . . 40
  6.1 Prototype measurements . . . 40
7 Discussion and applications . . . 40
8 Concluding remarks and outlook . . . 44
A Standard cryptographic notions . . . 48
  A.1 Public key encryption . . . 48
  A.2 Signature scheme . . . 49
  A.3 Non-interactive zero-knowledge (NIZK) proof system . . . 50

1 Introduction

Cryptography has long made private payments possible, from Chaum's ecash [18] to decentralized systems such as Zcash [4] and Monero [39]. In regulated settings, including central bank digital currencies (CBDCs) and institutional stablecoins, payment privacy must coexist with the ability to investigate financial crime. Most regulated systems support such investigations through trusted intermediaries with broad access to transaction histories. This access exposes the activities of ordinary users alongside those of suspects and can be abused for commercial exploitation or surveillance of political dissidents and journalists.

Investigative tools should therefore limit their reach and the information they expose. We focus on *targeted tracing*, which complements suspect identification and measures such as account freezes, transaction limits, or mandatory disclosure. Acting on outside evidence, such as a financial intelligence report, law enforcement actively embeds a *tracing tag* in a suspect's funds. This tag encrypts tracing information and travels with the funds, letting the authority uncover laundering routes and accomplices. Crucially, such tracing must not deanonymize users unrelated to the investigation.

The coin tracing schemes proposed in the literature [32,37] support such targeted investigations, but grant the authority tracing capabilities that are *unbounded*: once initiated, surveillance propagates indefinitely through the transaction graph, affecting potentially all users [32], and continues through all future transactions of a targeted user [37]. These schemes provide no cryptographic guarantee that tracing remains targeted and temporary; it rests instead on the authority's goodwill. A common mitigation distributes the capability: in PEReDi [37] and UTT [55], tracing requires a threshold of committee members to cooperate. Distributing trust in this way bounds *who* may trace, not *how much* may be traced, and in practice the shares concentrate in a handful of entities, often state-operated, whose collusion restores unrestricted surveillance.

Secure MPC could enforce these bounds [19], but running it for every transaction is too costly at ledger scale. We therefore seek lightweight tracing whose reach is fixed by public parameters and cannot be silently extended by the authority. We ask:

> *Can we design a simple and efficient protocol for targeted tracing whose reach is cryptographically bounded, even against the authority itself?*

1.1 Our contribution

We introduce *ephemeral coin tracing* (ECT), a primitive for targeted tracing with cryptographically enforced expiry. Its central idea is to attach encrypted tracing metadata to every account state and update that metadata as funds move. A trace follows tagged funds prospectively for a bounded number of policy-defined hops; expired contributions disappear while unrelated account data remains private.

Account tags and expiry. Every account includes an encrypted tracing tag in its authenticated state. The ledger checks that payments update these tags according to the protocol: each payment degrades the sender's tag and merges the result into the recipient's. The authority initiates a trace by issuing a tag that encrypts an identifier it assigns to the suspect; untargeted users are assigned zero. Encryption hides both the tracing status and the remaining hop budget from users. Each tracing contribution, that is, each identifier issued into a tag, expires after its assigned number of *hops*, bounded by the public maximum h fixed at setup. At expiry, it becomes a zero contribution that not even the authority can recover, without affecting other live contributions in the same tag.

Hop policy and evasion. The ledger policy determines when a hop occurs. It may degrade a tag on every outgoing transfer, once per epoch or block-height interval, or only when a transfer meets a condition such as a minimum amount. A user can exhaust a contribution's budget only by triggering enough such hops before moving the funds of interest. If every transfer counts, cheap "breadcrumb" payments may suffice; minimum amounts, rate limits, or counterparty requirements can instead make exhaustion consume time, meaningful value, or identified counterparties. Until the budget is exhausted, the trace remains live, and the evasion pattern may itself justify stronger measures or a newly recorded tag. Moving the funds immediately avoids this delay but risks making the incriminating transfer while the tag is still active.

Capacity and merging. ECT also exposes an identifier-slot bound: supporting more distinguishable identifiers requires larger public parameters and tags. This bound remains compatible with recurring investigations: a later warrant may start a new contribution, while each contribution retains its own hop budget. ECT merges tags issued at different times, and therefore at different degradation depths, homomorphically while preserving each identifier at its current depth. To the best of our knowledge, ECT is the first coin-tracing scheme to handle such merging efficiently.

Scope and applications. We use anonymous account-based payment systems as our running model [37,58], but ECT also applies to authenticated payment states that carry, update, and merge encrypted tags, including CBDCs, regulated stablecoins, custodial wallets, and interbank networks. Since degradation is public and requires no secret, ECT is flexible: deployments can assign it to account holders, custodians, validators, or other designated parties without interaction with the authority.

1.2 Technical approach

In short, ECT uses additively homomorphic encryption over a plaintext ring containing a publicly applicable nilpotent element ν of index h, such that $\nu^h = 0$ and $\nu^{h-1} \neq 0$. A tag encrypts an identifier, degradation multiplies its plaintext

by ν, and merging adds plaintexts. After h degradations, every contribution becomes zero and remains zero, so expired contributions are neutral under merging and do not affect live ones.

Setup and group structure. Our first construction uses exponential ElGamal [31] in a cyclic group $\mathbb{G} = \langle g \rangle$ of order q^h, where q is prime and the public parameter h is the hop budget. The prime-power order is essential: multiplication by q in the exponent is non-invertible and erases all information after h applications. We also instantiate the template with Damgård–Jurik encryption [24], where exponentiation by the composite modulus implements degradation and base-N digits separate hop depths, and with Ring-LWE, where BGV [9] ciphertexts over the ring $R = \mathbb{Z}[X, Y]/(X^{N_0} + 1, Y^h)$ use powers of Y for hop depth and the X-slots for identifiers. Multiplication by Y degrades a tag, ciphertext addition merges tags, and the construction provides post-quantum security. Among these constructions, exponential ElGamal is typically the best choice for deployments that seek to cap the authority's tracing power at a small set of identifiers. We evaluate it on a commodity laptop at 128-bit security and $h = 10$ hops: it adds a few kilobytes per transaction and traces 28 identifiers in 1.33 s. Deployments requiring a larger set of distinguishable identifiers can use the other constructions at the cost of larger tags: our Damgård–Jurik and Ring-LWE parameters support 341 and thousands of identifier slots, respectively. Any increase in capacity requires a public parameter change that the authority cannot make silently.

User identifiers and the base-r encoding. User id_i receives the plaintext identifier $m_i = r^{i-1}$, for a base $r \geq 2$ satisfying $r^n < q$, so each identifier occupies one base-r position and the complete encoding fits within one base-q digit. To trace id_i, the authority encrypts m_i; a dummy tag encrypts zero.

Degradation via exponentiation. Each degradation raises the ciphertext components to the q-th power, changing an encryption of mq^t into one of mq^{t+1}. The plaintext map $m \mapsto qm \bmod q^h$ is nilpotent: after h applications, $q^h m \equiv 0 \pmod{q^h}$, and zero remains zero, so expiry is exact and permanent. Semantic security hides whether a tag contains zero or an identifier and also hides its current depth; the user therefore learns neither its status nor the remaining budget.

Merging across identifiers and depths. Multiplying ElGamal ciphertexts adds their plaintexts. Within a depth, the base-r representation separates identifiers: the coefficient at position $i - 1$ records the multiplicity of id_i, and r is chosen to prevent overflows. Across depths, the base-q representation provides a second layer of separation: a contribution $q^t m_i$ occupies digit t, so contributions at different depths coexist without collision. The authority recovers the traced set by reading the base-q digits and then decoding each in base r.

Separation from ledger verification. ECT defines tag operations and their security properties; the deployment determines when and how tags enter accounts, while the ledger binds them to states, authenticates fresh entries, and prevents

reuse. Limits on tagged accounts are a separate deployment policy. The scheme bounds the number of distinguishable identifier slots, but the authority assigns them: users sharing a slot are indistinguishable to Trace, and reassigning one before its old contributions expire risks obscuring the trail for up to h hops.

Organization. Section 2 surveys related work, and Section 3 introduces the model and notation. Section 4 defines ECT and its security properties. Section 5 presents the nilpotent template, its three instantiations with their security proofs, and a comparison of the three. Section 6 evaluates our exponential ElGamal implementation. Section 7 discusses deployment choices and applications, and Section 8 draws a conclusion. Appendix A recalls standard cryptographic notions.

2 Related work

ECT lies at the intersection of privacy-preserving payment systems, anti-money-laundering mechanisms, and cryptographic tracing primitives. We first review approaches to reconciling transaction privacy with regulatory oversight, including controlled disclosure, privacy budgets, and collaborative analysis, and then compare ECT with prior coin-tracing, watermarking, and tracing techniques.

Compliant payment systems. Research on reconciling privacy with regulation dates back to the origins of ecash [17]. Chaum [18] introduced blind signatures for untraceable payments, which von Solms and Naccache [53] showed can shield extortion and other perfect crimes, and Sander and Ta-Shma [50] added a mechanism for auditing the total amount in circulation. More recent permissioned designs provide auditable token payments [2,42,15,16] or add auditability to Quisquis [28,43], allowing selective disclosure to regulators without revealing information to other users. A parallel line of work makes the authority accountable for each decryption through on-chain logs [49,41,40], collaboration among several regulators [61], or self-incriminating proofs that expose rogue decryptors [20]. A further line shifts compliance to the users themselves, who prove that their funds avoid illicit sources without revealing their history [11]. These mechanisms constrain who may decrypt and record each decryption; ECT instead bounds the reach of the resulting trace.

Privacy budgets. Privacy-budget mechanisms enforce compliance through value thresholds, an idea that goes back to e-cash that is anonymous only up to a spending limit [13]. A user transacts anonymously below a limit defined per transaction [57], per period [55], or for incoming value [58]; above the limit, disclosure to the regulator is mandatory. The same threshold applies to all users. Thus, a user with high transaction volume, such as a currency exchange, may be exposed indefinitely. Moreover, inflation can cause an increasing fraction of users to exceed a fixed threshold without any change in policy. ECT instead bounds each trace in hops rather than in value.

Collaborative anti-money-laundering. Money-laundering routes often cross institutional boundaries, but banks generally cannot pool their records. Secure risk propagation over an interbank network [27], AMLChain [35], and homomorphic graph inference [26] privately compute risk scores or compliance predicates from records held by several institutions. ECT is complementary: once such an analysis, or conventional off-ledger evidence, identifies a target, ECT prospectively follows the associated funds across participating banks. It does not require a joint computation over the entire interbank graph on the critical path of every payment.

Coin tracing. Fair e-cash lets designated trustees lift the anonymity of otherwise untraceable payments [54,10,14,29,25,36], which bounds who may revoke but not how much follows once revocation is triggered. Garman *et al.* [32] mark individual coins in decentralized anonymous payment schemes [4]: traceability propagates to every descendant coin until a regulated entity clears it, so a trace has no horizon of its own and the tracing payload grows linearly in the number of inputs. Kiayias *et al.* [37] and Tomescu *et al.* [55] improve accountability with a committee-initiated scheme in which each of a user's transactions carries a pseudorandom tag derived from a hidden per-user tracing key, which the committee reconstructs from shares to locate those transactions; a trace, once started, still covers every transaction the user makes. ECT differs from both in bounding the reach of a trace cryptographically rather than procedurally.

Watermarking and tracing primitives. Tag embedding resembles cryptographic watermarking, and more broadly steganography [12], but a watermark is meant to persist, whereas our tags must merge with one another and expire at a fixed budget. Undetectable watermarks for language models [22] embed a single bit for a single user; Cohen *et al.* [23] reach the multi-user setting but not merging, for the reason given in Section 5; Zhao *et al.* [60] survey the area. Traitor tracing [21,7,6,59] and traceable signatures [38,30,34] trace the party responsible for a forbidden or disputed action, whereas ECT traces the ordinary transactions of a user designated in advance.

3 Preliminaries

Notation. For $n \in \mathbb{N}$, let $[n] = \{1, \ldots, n\}$. We write $x \xleftarrow{\$} S$ when x is sampled uniformly from S. A function $f : \mathbb{N} \to \mathbb{R}$ is negligible, written $f(\lambda) = \mathsf{negl}(\lambda)$, if $f(\lambda) < 1/p(\lambda)$ for every polynomial p and all sufficiently large λ. PPT denotes probabilistic polynomial time.

To support tag merging in the plaintext ring, we use the following positional encoding: a multiset is represented by an integer whose base-r digits record element multiplicities, allowing the multiset to be recovered after encoded elements are added.

Definition 1 (Base-r encoding). *Let $n \in \mathbb{N}$ and $r \geq 2$. Assign index $i \in [n]$ the value $m_i = r^{i-1}$. For a multiset T over $[n]$ with multiplicities $c(i) \in$*

$\{0, \ldots, r-1\}$, *define* $v_T = \sum_{i=1}^{n} c(i) r^{i-1}$. *Because* $c(i) < r$, *no digit carries into the next position. Hence* $v_T \leq r^n - 1$ *and uniquely determines* T.

When tracing tags are merged, the coefficient of each base-r position records the multiplicity of the corresponding identifier. These coefficients must remain below r to prevent overflows into adjacent positions; the ledger's bound on transaction fan-out ensures that they do.

Lemma 1 (Bounded-fan-out multiplicity). *Suppose a transaction copies each input-tag contribution to at most d output tags. Then an identifier has multiplicity at most d^t at depth t, and at most d^{h-1} at any live depth $t < h$.*

Proof. The multiplicity is one at depth 0. If it is at most d^t at depth t, copying each occurrence to at most d outputs gives multiplicity at most d^{t+1} at depth $t+1$. Induction proves the first bound; the maximum over $t < h$ gives the second. □

In our two-state account model, $d = 2$: the sender's tag propagates to both the change state and the recipient's new state, giving multiplicity at most 2^{h-1}. An incoming payment carries the recipient's existing contributions to only one successor state. Thus, only outgoing transfers increase multiplicity (Section 4).

3.1 Cryptographic notions

The ECT primitive uses additively homomorphic encryption and signatures. We require signature correctness and existential unforgeability under chosen-message attacks (EUF-CMA). A deployment may additionally use a NIZK, direct public verification, or a trusted-execution attestation to enforce its state-transition policy, but that verification mechanism is not part of the ECT primitive. Appendix A recalls the cryptographic notions used in the paper. We state homomorphic encryption explicitly because its algebraic properties determine merging and degradation.

Homomorphic encryption. An *additively homomorphic encryption scheme* over a message group $\mathbb{M}$ consists of PPT algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Rerand})$ satisfying correctness and IND-CPA security. The ciphertext space forms a group under $\cdot$, and for $c_i = \mathsf{Enc}(\mathsf{pk}, m_i)$, $\mathsf{Dec}(\mathsf{sk}, c_1 \cdot c_2) = m_1 + m_2$. Rerandomization computes $c \cdot \mathsf{Enc}(\mathsf{pk}, 0)$ and produces a ciphertext computationally indistinguishable from a fresh encryption of the same message. Homomorphic addition implements merging; rerandomization hides the link between a tag and its degraded successor.

3.2 System model

ECT is a cryptographic layer attached to a host ledger. The model separates the primitive's tag operations from the deployment mechanisms that authenticate account states, enforce transitions, and determine when tags are issued or degraded.

Ledger and accounts. All parties are computationally bounded. The system consists of users, a tracing authority holding $\mathsf{sk}_{\mathsf{trace}}$, and a permissioned ledger with at least $3f + 1$ validators, of which at most f are Byzantine. Users enroll under real-world identities and receive credentials. The authority knows these identities but cannot link them to ledger states except at system boundaries. As in Platypus [58] and PayOff [3], an account state commits to a balance, account secret, tracing tag, and epoch. A payment consumes the sender's and recipient's states and creates two new states. Its proof establishes membership, correct nullifiers, balance conservation, enrollment, and the required tag transition. Each tag is a ciphertext under the authority's key and is bound to its published state.

Tag lifecycle. Time is divided into epochs. At the start of each epoch, the authority publishes a signed *tag board* with one new ciphertext for every enrolled identity. The entry encrypts zero unless the authority issues or renews a warrant for that identity in this epoch, in which case it encrypts a tracing identifier assigned to that identity; semantic security hides which entries are nonzero. These authenticated board entries are the only source of new tag contributions. Each account merges its board entry into its tag, so that existing and newly issued contributions remain active together; under our default policy, the account first degrades its existing tag, placing contributions issued in different epochs at different depths. The transaction proof hides which board entry was selected while enforcing this update rule and its binding to the account owner's credential; a per-identity, per-epoch nullifier prevents the same entry from being ingested more than once. The policy also determines when accounts must ingest entries and whether dormant accounts consume hop budget. To renew a warrant, the authority issues a fresh identifier in a later epoch, that is, one never issued before or one whose earlier contributions have expired, for instance because every account degrades at each ingestion and h epochs have passed. The authority records the identity behind each identifier, so renewal does not affect attribution. Each identifier therefore has at most one live issuance, as Lemma 1 requires. The warrant controls when fresh contributions may be issued, while ECT limits how far each contribution can propagate.

Privacy and adversaries. We assume that hiding commitments and membership proofs over the full state set provide ledger confidentiality, unlinkability, and untraceability [37,56]. ECT reveals only live tags within their hop window; dummy and expired tags decrypt to zero. Identities are recovered only at system boundaries or through later enforcement. The primitive guarantees that no party, including the tracing authority, can recover a contribution after its hop budget and that a tag can distinguish at most $\lfloor \log_r q \rfloor$ identifier slots without larger parameters. The ledger integration is responsible for rejecting malformed transitions, binding tags to states, preventing reuse, and enforcing any issuance quota or slot-uniqueness policy. We exclude out-of-band disclosure of tracing results. The two-state account model supplies the fan-out bound required by Lemma 1.

With this model in place, we are ready to define the ephemeral tracing scheme.

4 Ephemeral tracing scheme

An ephemeral tracing scheme tracks identifiers through encrypted tags that move toward expiry whenever they are degraded. We call each application of degradation a *hop*. ECT specifies the effect of a hop and the number of hops a tag survives, but the ledger policy determines which events trigger one. A hop might be charged for every outgoing transfer, only for transfers above a value threshold, once per epoch or block-height interval, or upon interaction with an enrolled counterparty. This choice determines how costly it is to evade tracing: charging every transfer may let a target exhaust the budget through many negligible payments, whereas thresholds, rate limits, or counterparty requirements can make exhaustion consume time, meaningful value, or identified counterparties (see Section 1.2). A tag's *hop budget* is the number of such policy-defined steps it survives; once that budget is exhausted, the tag becomes untraceable even to the authority. When several payment states are combined, their tags can be merged without resetting their progress toward expiry, so the resulting tag preserves each live identifier at its current degradation depth.

4.1 Characterization

An ephemeral tracing scheme consists of the following six algorithms:

$\mathsf{Setup}(1^\lambda, n, h, r) \to (pp, \mathsf{sk}_{\mathsf{trace}})$ Takes as input the security parameter λ, the number of identifier slots n (the maximum number of identifiers that a tag can distinguish at once), the *maximum* hop budget h, and the encoding base r. Outputs public parameters pp, which fix h publicly, and a tracing secret key $\mathsf{sk}_{\mathsf{trace}}$. The public parameters pp are implicit inputs to the remaining algorithms.

$\mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}, b, h') \to (w, \sigma)$ Takes as input the tracing key, target identifier id, trace flag $b \in \{0, 1\}$, and a per-tag hop budget $h' \leq h$. Returns a tag w embedding id if $b = 1$ or zero if $b = 0$, together with an authority signature σ on w. The tag expires after exactly h' degradation hops.

$\mathsf{Verify}(w, \sigma) \to \{0, 1\}$ Verifies that σ is a valid authority signature on the freshly issued tag w.

$\mathsf{Degrade}(w) \to (w', \xi)$ Takes as input a tag w. Returns a rerandomized tag w' after one degradation step and the private transition witness ξ.

$\mathsf{Merge}(w_1, \ldots, w_m) \to w'$ Takes as input tags $w_1, \ldots, w_m$. Returns the merged tag $w' = \prod_j w_j$, which contains the union of the traced identifiers across all inputs.

$\mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w) \to S$ **or** $\perp$ Takes as input the tracing key and a tag w. Outputs a set S of recovered identifiers, or $\perp$ if the coin is untraced (dummy tag) or fully degraded.

The authority runs Setup, Tag, and Trace, while validators use Verify to authenticate freshly issued tags. In contrast, Degrade and Merge are public operations that require no secret key, so the ledger policy may assign them to account holders, custodial intermediaries, validators, or another designated party. Under our default payment policy, an outgoing payment advances the contributions in the sender's tag by one hop but does not advance contributions already held by the recipient. Given the sender's tag w_S and the recipient's tag w_R, the party constructing the transaction computes $(\hat{w}_S, \xi) \leftarrow \mathsf{Degrade}(w_S)$. It assigns $\hat{w}_S$ to the sender's successor state and merges $\hat{w}_S$ with w_R for the recipient's successor state; that is, $w'_S = \hat{w}_S$ and $w'_R = \mathsf{Merge}(\hat{w}_S, w_R)$. The party retains ξ as a witness for the host transaction proof, which binds the tag update to the hidden input state. Alternatively, a trusted execution environment may attest the update, or validators may apply a deterministic, publicly recomputable degradation. Charging a hop only to the sender's tag prevents third parties from exhausting a target's budget through unsolicited incoming payments. Under the default epoch-ingestion policy, an account similarly degrades its existing tag before merging its new board entry, placing contributions issued in successive epochs at different depths (Section 3.2). Figure 1 illustrates propagation, merging, and per-contribution expiry under the default policy in which spending advances the sender's tag. Figure 2 summarizes the resulting flow between the

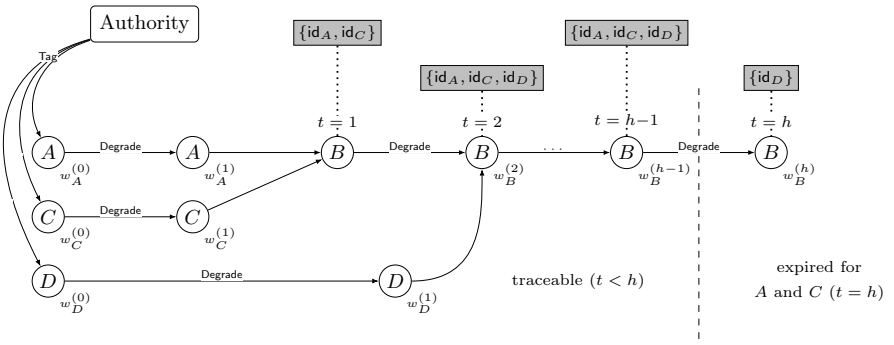


Fig. 1. Tag propagation, merging, and expiry, with t counting hops and h the budget. The authority issues tags with Tag, and every outgoing payment applies Degrade to the spent state. At $t = 1$, A and C pay B, and their degraded tags merge into B's state; from then on B spends once per step, advancing every contribution it carries. D stays dormant until it pays B at $t = 2$, so its contribution joins one hop younger and a single tag then carries three identifiers at two depths. The gray boxes give the set returned by Trace. The contributions of A and C reach depth h at $t = h$ and decrypt to zero, while that of D remains recoverable: expiry is per contribution and exact at the budget.

authority, users, and validators. This assignment is illustrative: ECT specifies the tag operations but leaves transition enforcement to the deployment.

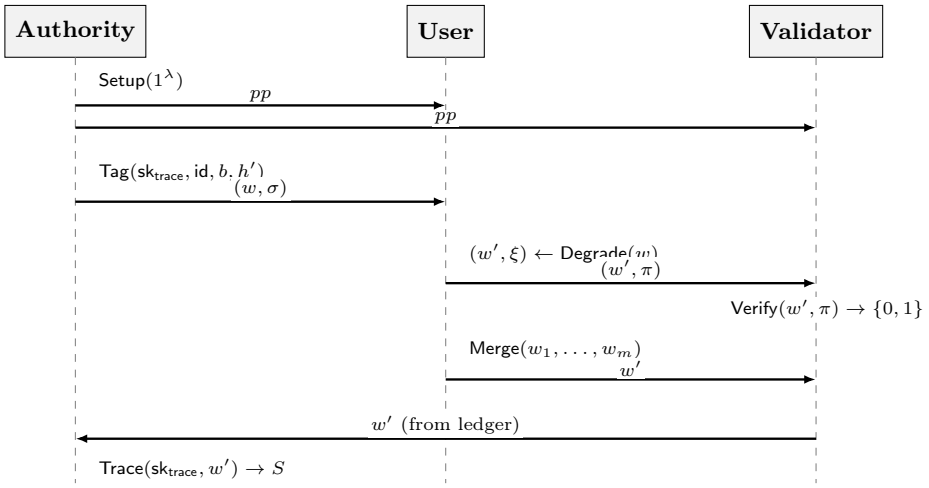


Fig. 2. Protocol overview: interactions between the authority, users, and validators. The authority runs Setup and Tag (producing (w, σ)) and Trace (requiring $\mathsf{sk}_{\mathsf{trace}}$). Users run Degrade and Merge when constructing payments. Validators run the host system's Verify algorithm on the transaction-validity proof, whose witness includes ξ.

4.2 Security

We require two correctness conditions and four security properties, which we define formally below. *Trace correctness* (Definition 2) requires a live tag to reveal its identifier. *Merge correctness* (Definition 3) requires a merged tag to reveal exactly its live contributions; dummy and expired contributions must be neutral. *Indistinguishability* (Definition 4) prevents a user with access to the tagging oracle from distinguishing a traced tag from a dummy tag. *Ephemeral traceability* (Definition 5) makes expiry irrevocable even for an authority holding $\mathsf{sk}_{\mathsf{trace}}$. *Unforgeability* (Definition 6) prevents an adversary from framing an identifier that the authority never tagged; because encryption is public, the notion only concerns tags with a valid issuance and transition history. *Unlinkability* (Definition 7) prevents correlation across degradation steps. For the games below, Setup produces public parameters pp and tracing key $\mathsf{sk}_{\mathsf{trace}}$. The authority calls $(w, \sigma) \leftarrow \mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}, b, h')$, where $b = 1$ embeds id, $b = 0$ embeds zero, and $h' \leq h$ is the contribution's budget; $\mathsf{Verify}(w, \sigma)$ authenticates this fresh issuance. Public degradation returns $(w', \xi) \leftarrow \mathsf{Degrade}(w)$, with private witness ξ, public merging returns $\mathsf{Merge}(w_1, \ldots, w_m)$, and $\mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w)$ returns the live identifiers or $\perp$. We suppress the public identifier-slot bound n, maximum budget h, and encoding base r, and write $\mathsf{Degrade}^k(w)$ for k sequential applications while omitting the retained witnesses. Under the two-output payment policy, a contribution occurs at most 2^t times at depth t; hence $r > 2^{h-1}$ prevents carries at every live depth and makes the identifier multiset uniquely decodable.

The following games define trace correctness, merge correctness, indistinguishability, ephemeral traceability, unforgeability, and unlinkability, in that order. Trace correctness captures the basic functionality of the scheme: a tag issued for an identifier and degraded within its hop budget must remain recoverable by the authority.

Definition 2 (Trace correctness). *An ECT scheme satisfies* trace correctness *if for every identifier* id*, every per-tag budget* $h' \leq h$*, and every* $k < h'$*:*

$$\Pr\left[\begin{array}{l}(pp, \mathsf{sk}_{\mathsf{trace}}) \leftarrow \mathsf{Setup}(1^\lambda);\\ (w^{(0)}, \sigma) \leftarrow \mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}, 1, h');\\ w^{(k)} \leftarrow \mathsf{Degrade}^k(w^{(0)});\\ \mathsf{id} \in \mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w^{(k)})\end{array}\right] \geq 1 - \mathsf{negl}(\lambda),$$

where $\mathsf{Degrade}^k(w^{(0)})$ *denotes* k *sequential applications of the* $\mathsf{Degrade}$ *operation.*

Merge correctness extends trace correctness to the multi-coin setting: after merging, the authority must recover exactly the identifiers whose contributions are still within budget. Expired and dummy contributions must act as neutral elements of merging, vanishing without disturbing live ones.

Definition 3 (Merge correctness). *An ECT scheme satisfies* merge correctness *if for every* $m \in \mathbb{N}$*, identifiers* $\mathsf{id}_1, \ldots, \mathsf{id}_m$*, flags* $b_1, \ldots, b_m \in \{0,1\}$*, per-tag budgets* $h'_1, \ldots, h'_m \leq h$*, and degradation counts* $t_1, \ldots, t_m \geq 0$*, writing* $L = \{i : b_i = 1 \wedge t_i < h'_i\}$ *for the live inputs and* $S_L = \{\mathsf{id}_i : i \in L\}$ *if* $L \neq \emptyset$ *and* $S_L = \perp$ *otherwise, the following holds whenever each identifier occurs at most* 2^{h-1} *times among the live inputs at any one depth* $h - h'_i + t_i$*:*

$$\Pr\left[\begin{array}{l}(pp, \mathsf{sk}_{\mathsf{trace}}) \leftarrow \mathsf{Setup}(1^\lambda);\\ (w_i^{(0)}, \sigma_i) \leftarrow \mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}_i, b_i, h'_i),\ i \in [m];\\ w_i \leftarrow \mathsf{Degrade}^{t_i}(w_i^{(0)}),\ i \in [m];\\ w' \leftarrow \mathsf{Merge}(w_1, \ldots, w_m);\\ \mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w') = S_L\end{array}\right] \geq 1 - \mathsf{negl}(\lambda),$$

where $\mathsf{Degrade}^{t_i}(w_i^{(0)})$ *denotes* t_i *sequential applications of* $\mathsf{Degrade}$*. The multiplicity condition follows from a valid history under the two-output payment policy with at most one live issuance per identifier (Lemma 1); it is necessary because arbitrary repetitions can overflow an identifier slot.*

Indistinguishability ensures that a user cannot determine whether their coin carries a traceable or a dummy tag, even given arbitrarily many other tags of its choice.

Definition 4 (Indistinguishability). *An ECT scheme satisfies* indistinguishability *if for every PPT adversary $\mathcal{A}$:*

$$\Pr\left[\begin{array}{l}(pp, \mathsf{sk}_{\mathsf{trace}}) \leftarrow \mathsf{Setup}(1^\lambda);\\ (\mathsf{id}, h') \leftarrow \mathcal{A}^{\mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \cdot, \cdot, \cdot)}(pp);\\ b \xleftarrow{\$} \{0,1\};\\ (w, \sigma) \leftarrow \mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}, b, h');\\ b' \leftarrow \mathcal{A}^{\mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \cdot, \cdot, \cdot)}(w, \sigma);\\ b' = b\end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

Ephemeral traceability is the central guarantee of the scheme: once a tag has been degraded beyond its hop budget, the authority irrevocably loses the ability to recover the traced identifier, even in possession of $\mathsf{sk}_{\mathsf{trace}}$. In the exponential ElGamal and Damgård–Jurik instantiations the guarantee is stronger still: degradation rerandomizes with fresh uniform randomness, so a tag whose contributions have all expired is distributed exactly as a dummy tag, even to the holder of $\mathsf{sk}_{\mathsf{trace}}$.

Definition 5 (Ephemeral traceability). *An ECT scheme satisfies* ephemeral traceability *if for every identifier* id*, every per-tag budget $h' \leq h$, and every $k \geq h'$:*

$$\Pr\left[\begin{array}{l}(pp, \mathsf{sk}_{\mathsf{trace}}) \leftarrow \mathsf{Setup}(1^\lambda);\\ (w^{(0)}, \sigma) \leftarrow \mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}, 1, h');\\ w^{(k)} \leftarrow \mathsf{Degrade}^k(w^{(0)});\\ \mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w^{(k)}) \neq \perp\end{array}\right] \leq \mathsf{negl}(\lambda),$$

where $\mathsf{Degrade}^k(w^{(0)})$ *again denotes k sequential applications of* $\mathsf{Degrade}$.

Unforgeability rules out framing: an adversary cannot derive a tag with a valid history that traces to an identifier not tagged by the authority. The notion cannot be stated for bare ciphertexts, because public encryption allows anyone to encrypt arbitrary identifiers; it only concerns tags with a valid history. Let R_{hist} contain every (pp, w^*, H) such that H derives w^* from issued tags through the payment and ingestion transitions of Section 4. Every issuance in H must satisfy $\mathsf{Verify}(w, \sigma) = 1$. A payment degrades a sender tag w_S to $\hat{w}_S$, reproducibly from its witness ξ, and outputs $\hat{w}_S$ and $\mathsf{Merge}(\hat{w}_S, w_R)$ for a recipient tag w_R. An ingestion degrades an account tag in the same way and merges one issued tag into the result. No tag may be consumed twice. A ledger proof with soundness and nullifiers is one way to ensure that every accepted tag has such a history; direct verification or an attested execution environment may provide the same guarantee.

Definition 6 (Unforgeability). *An ECT scheme satisfies* unforgeability *if for every PPT adversary $\mathcal{A}$, letting $\mathcal{Q}$ denote the set of identifiers queried to the* Tag

oracle with flag $b = 1$, where $\mathcal{A}$ queries each identifier at most once with $b = 1$:

$$\Pr\left[\begin{array}{l}(pp, \mathsf{sk}_{\mathsf{trace}}) \leftarrow \mathsf{Setup}(1^\lambda);\\ (w^*, H) \leftarrow \mathcal{A}^{\mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \cdot, \cdot, \cdot)}(pp);\\ (pp, w^*, H) \in R_{\mathsf{hist}} \ \wedge\\ \mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w^*) \neq \bot \ \wedge\\ \quad \exists\, \mathsf{id} \in \mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w^*) : \ \mathsf{id} \notin \mathcal{Q}\end{array}\right] \leq \mathsf{negl}(\lambda).$$

The game allows one nonzero issuance per identifier because Lemma 1 bounds the copies of a single contribution. This matches the tag lifecycle of Section 3.2, where a renewal issues a fresh identifier; reusing a slot after all its earlier contributions have expired is equivalent, since expired contributions are zero. Unlinkability prevents any party without the tracing key from correlating a tag before and after a degradation step. The game below isolates the protection supplied by rerandomization; in a deployment the host ledger's zero-knowledge transaction proof additionally hides which committed state was consumed.

Definition 7 (Unlinkability). *An ECT scheme satisfies* unlinkability *if for every PPT adversary $\mathcal{A}$:*

$$\Pr\left[\begin{array}{l}(pp, \mathsf{sk}_{\mathsf{trace}}) \leftarrow \mathsf{Setup}(1^\lambda);\\ (w_0, w_1) \leftarrow \mathcal{A}^{\mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \cdot, \cdot, \cdot)}(pp);\\ b \xleftarrow{\$} \{0, 1\};\\ (w', \xi) \leftarrow \mathsf{Degrade}(w_b);\\ b' \leftarrow \mathcal{A}(w_0, w_1, w');\\ b' = b\end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

Having defined the interface and the security properties of ECT, we now give three constructions that satisfy all six properties.

5 Constructions

Standard properties of homomorphic encryption and signatures provide most of the required guarantees: additive homomorphism enables merging, IND-CPA security hides whether a tag is traced, rerandomization gives unlinkability, and EUF-CMA signatures authenticate issuance. Transition validation is handled separately by the deployment. The main challenge within ECT is *ephemeral traceability*. After exactly h steps, a tag must become indistinguishable from a dummy tag without destroying live contributions when tags are merged. We first examine two natural approaches that each satisfy only one side of this requirement.

5.1 Approaches from first principles

One approach is *noisy rerandomization*. Each degradation step adds bounded noise $\delta \in [-\Delta, \Delta]$ to the encoded identifier. Once enough noise has accumulated,

decoding fails and Trace returns $\perp$ with overwhelming probability. Decoding failure, however, is not the same as expiry. Even after h hops, the decryption of a traced tag remains concentrated around $m_i \cdot M$, whereas a dummy tag decrypts to zero. The authority can therefore still distinguish expired traced tags from dummy tags, and the tracing window does not close at a sharp threshold.

A second approach makes an expired tag look uniformly random. For example, pseudorandom-code constructions add noise from the ambient space until the codeword becomes computationally indistinguishable from uniform and its identifier can no longer be decoded, as in the construction of Cohen *et al.* [23]. This gives a stronger notion of expiry, but it is incompatible with merging. A uniformly random expired tag is not a neutral element: merging it with a live tag randomizes the result and destroys the live identifier as well.

These approaches expose the two requirements that degradation must satisfy simultaneously. Expired tags must become indistinguishable from dummy tags, and they must behave neutrally when merged with tags that remain live. The nilpotent construction below achieves both by making every expired contribution equal to zero.

5.2 Nilpotent degradation

The template requires an additively homomorphic encryption scheme whose plaintext ring R contains a publicly applicable element ν that is nilpotent of index h: $\nu^h = 0$ and $\nu^{h-1} \neq 0$. A tag encrypts an identifier, degradation multiplies its plaintext by ν and rerandomizes the ciphertext, and merging adds plaintexts homomorphically. After exactly h degradations, every contribution becomes zero and remains neutral under merging. Before then, the encoding keeps live contributions separable. Degradation requires no secret key; Section 7 further discusses which party may apply it.

We instantiate the template with exponential ElGamal over $\mathbb{Z}_{q^h}$, using $\nu = q$ and $c \mapsto c^q$ (Section 5.3); Damgård–Jurik over $\mathbb{Z}_{N^h}$, using $\nu = N$ and $c \mapsto c^N$ (Section 5.4); and Ring-LWE over $R_0[Y]/(Y^h)$, using $\nu = Y$ (Section 5.5). ElGamal gives the smallest tags, Damgård–Jurik gives polynomial-time tracing under DCR, and Ring-LWE provides post-quantum security using Chinese remainder theorem (CRT) slots instead of positional encoding.

5.3 Exponential ElGamal instantiation

Exponential ElGamal gives compact tags and is designed for tags containing a small set of distinguishable live identifiers, since tracing becomes expensive as this set grows. The public parameters bound the identifier-slot capacity, but bounding the number of tagged accounts requires a separate issuance policy. The construction uses a cyclic group $\mathbb{G}$ of order q^h for prime q; this prime-power order requires a non-standard DDH assumption and new parameters for each hop budget.

Definition 8 (Prime-power-order group). *A* cyclic group of prime-power order q^h *is a group* $\mathbb{G} = \langle g \rangle$ *in which* $g^{q^h} = 1$ *and* $g^k \neq 1$ *for* $0 < k < q^h$.

Exponentiation by q realizes nilpotent degradation: applying $x \mapsto x^q$ exactly h times gives $x^{q^h} = 1$. Thus, q is the nilpotent element of Section 5.2. Gennaro, Krawczyk, and Rabin reduce DDH in cyclic groups to their maximal prime-power-order components, but not further to prime-order subgroups [33]. We therefore state the required assumption directly.

Definition 9 (DDH in prime-power-order groups). *Let $\mathbb{G}$ be a cyclic group of order q^h with generator g. The* Decisional Diffie-Hellman (DDH) assumption *holds in $\mathbb{G}$ if for all PPT distinguishers $\mathcal{D}$, where $a, b, c \xleftarrow{\$} \mathbb{Z}_{q^h}$,*

$$\left|\Pr[\mathcal{D}(g,\, g^a,\, g^b,\, g^{ab}) = 1] - \Pr[\mathcal{D}(g,\, g^a,\, g^b,\, g^c) = 1]\right| \leq \mathsf{negl}(\lambda).$$

Projection by $x \mapsto x^{q^{h-1}}$ shows that prime-order DDH is necessary, but the converse is not known. Boneh's random self-reduction gives worst-case-to-average-case equivalence when q is super-polynomial [5], while Shoup bounds generic advantage after m operations by $O(m^2/q)$ [51]. Hence, generic attacks require $\Omega(\sqrt{q})$ operations, matching Pohlig–Hellman's $O(h\sqrt{q})$ cost up to h [48]. These results support, but do not prove, our assumption. One realization is the order-q^h subgroup of $\mathbb{Z}_p^*$ for a prime p such that $q^h \mid p - 1$.

With the algebraic setting and security assumption fixed, we now present the six algorithms of the exponential-ElGamal construction in lifecycle order. We begin with Setup (Alg. 1), which generates an ElGamal key pair for tag encryption and a signature key pair for authenticating issued tags.

Algorithm 1. Setup.

$\mathsf{Setup}(1^\lambda, n, h, r) \to (pp, \mathsf{sk}_{\mathsf{trace}})$
- Let r be an integer with $r > 2^{h-1}$; `// encoding base, Lemma 1`
- Choose prime q with $q > r^n$ and $q \geq 2^{2\lambda}$, $q \neq r$;
- Construct cyclic group $\mathbb{G}$ of order q^h with generator g;
- $(\mathsf{pk}_{\mathrm{enc}}, \mathsf{sk}_{\mathrm{enc}}) \leftarrow \mathsf{KeyGen}(\mathbb{G})$;
- $(\mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{sig}}) \leftarrow \mathsf{SigKeyGen}(1^\lambda)$;
- Assign $m_i \leftarrow r^{i-1}$ for each user id_i, $i \in [n]$;
- $pp \leftarrow (\mathbb{G}, q, h, n, r, \mathsf{pk}_{\mathrm{enc}}, \mathsf{pk}_{\mathrm{sig}})$, $\mathsf{sk}_{\mathsf{trace}} \leftarrow (\mathsf{sk}_{\mathrm{enc}}, \mathsf{sk}_{\mathrm{sig}})$;
- **return** $(pp, \mathsf{sk}_{\mathsf{trace}})$;

The public parameters make both bounds auditable. For $\mathbb{G} \subseteq \mathbb{Z}_p^*$, the conditions $q^h \mid p - 1$, $g^{q^h} = 1$, and $g^{q^{h-1}} \neq 1$ establish the hop budget h. They also give capacity $n_{\max} = \lfloor \log_r q \rfloor$; larger encodings cross a base-q digit and require running Setup with new parameters.

Tag (Alg. 2) realizes budget $h' \leq h$ by issuing at depth $h - h'$. Setting $b = 0$ encrypts zero. The authority signs each issued ciphertext, and Verify (Alg. 3) checks the signature under $\mathsf{pk}_{\mathrm{sig}}$.

Algorithm 2. Tag issuance.

$\mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}_i, b, h') \to (w, \sigma)$
- $m \leftarrow q^{h-h'} \cdot r^{i-1} \cdot b$;
- $w \leftarrow \mathsf{Enc}(\mathsf{pk}_{\mathrm{enc}}, m)$;
- $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}_{\mathrm{sig}}, w)$;
- **return** (w, σ);

Algorithm 3. Verification.

$\mathsf{Verify}(w, \sigma) \to \{0, 1\}$
- **return** $\mathsf{Ver}(\mathsf{pk}_{sig}, w, \sigma)$;

$\mathsf{Degrade}$ (Alg. 4) raises both components to q and rerandomizes, implementing $m \mapsto qm \bmod q^h$. It returns the rerandomization exponent as ξ. The merging algorithm (Alg. 5) multiplies ciphertexts componentwise, thereby adding their plaintexts. The transaction statement enforces correctness.

Algorithm 4. Degradation.

$\mathsf{Degrade}(w) \to (w', \xi)$
- Parse $w = (A, B)$;
- $r' \xleftarrow{\$} \mathbb{Z}_{q^h}$;
- $w' \leftarrow \left(A^q \cdot g^{r'},\ B^q \cdot \mathsf{pk}_{\mathrm{enc}}^{r'}\right)$;
- **return** $(w', \xi = r')$;

Algorithm 5. Merging.

$\mathsf{Merge}(w_1, \ldots, w_m) \to w'$
- $w' \leftarrow \prod_{j=1}^{m} w_j$;
- **return** w';

Trace (Alg. 6) recovers $v \in \mathbb{Z}_{q^h}$ from g^v: base q separates degradation depths, and base r separates identifiers within each depth. Let $I \subseteq [n]$ contain the K identifier slots issued so far, which the authority records with $\mathsf{sk}_{\mathsf{trace}}$, split as $I = I_0 \sqcup I_1$. For $b \in \{0, 1\}$, let D_b contain all sums $\sum_{i \in I_b} c_i r^{i-1}$ with $0 \le c_i \le 2^{h-1}$, and set $D = D_0 + D_1 \subseteq \mathbb{Z}_q$. Lemma 1 gives the coefficient bound, and $|D| \le r^K$ because $r > 2^{h-1}$.

As in Pohlig–Hellman [48], tracing extracts one digit at a time. With $\gamma = g^{q^{h-1}}$ and $y_t = g^{\sum_{j \ge t} d_j q^j}$, the equation $y_t^{q^{h-1-t}} = \gamma^{d_t}$ determines $d_t \in D$; multiplying by $g^{-d_t q^t}$ removes it. A meet-in-the-middle search over D_0 and D_1 finds each digit in $O(r^{K/2})$ time and memory. Overall, tracing costs $O(hr^{K/2})$ group operations and $O(r^{K/2})$ memory, so polynomial-time tracing requires $K = O(\log \lambda)$. Damgård–Jurik removes this restriction (Section 5.4). If no valid digit exists, Trace returns $\perp$.

Algorithm 6. Tracing.

$\mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w) \to S$ *or* $\perp$
  Parse $w = (A, B)$ and set $y_0 \leftarrow B/A^{\mathsf{sk}_{\mathrm{enc}}}$; // $y_0 = g^v$
  if $y_0 = 1$ **then return** $\perp$;
  $\gamma \leftarrow g^{q^{h-1}}$;
  for $t = 0, \ldots, h-1$ **do**
    if *no* $d \in D$ *satisfies* $\gamma^d = y_t^{q^{h-1-t}}$ **then return** $\perp$;
    $d_t \leftarrow$ the $d \in D$ with $\gamma^d = y_t^{q^{h-1-t}}$;
    $y_{t+1} \leftarrow y_t \cdot g^{-d_t q^t}$;
  Write $d_t = \sum_{i \in I} c_{t,i}\, r^{i-1}$ for each t;
  return $S \leftarrow \{\mathsf{id}_i : c_{t,i} > 0$ *for some* $t\}$;

Under DDH in $\mathbb{G}$ and EUF-CMA security of the issuance signature, the construction satisfies the six properties of Section 4.2, as we now show.

Theorem 1 (Trace correctness). *For any* $\mathsf{id}_i \in [n]$*, any per-tag budget* $h' \leq h$*, and any* $t < h'$*, let* $w \leftarrow \mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}_i, 1, h')$ *and* w' *be obtained by applying* Degrade *exactly* t *times. Then* $\mathsf{id}_i \in \mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w')$ *with probability* $1 - \mathsf{negl}(\lambda)$.

Proof. By the ElGamal homomorphism, raising each ciphertext component to the q-th power multiplies the plaintext by q in $\mathbb{Z}_{q^h}$. The tag $\mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}_i, 1, h')$ encrypts $q^{h-h'} r^{i-1}$ (issuance depth $h - h'$); after t applications of Degrade it encrypts $q^{h-h'+t} \cdot r^{i-1} \bmod q^h$. Since $r^{i-1} < q$ (by $q > r^n$) and $h - h' + t < h$ for $t < h'$, this value is nonzero. Writing it in base q, the digit at position $h - h' + t$ is r^{i-1} and all others are zero; decomposing r^{i-1} in base r gives a unit coefficient at position $i - 1$, so Trace returns id_i. □

Theorem 2 (Merge correctness). *For* $i \in [m]$*, let* $w_i \leftarrow \mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}_i, b_i, h_i')$*, and suppose each* w_i *has been degraded* $t_i \geq 0$ *times before merging; write* $\tau_i = (h - h_i') + t_i$ *for its current depth and let* $L = \{i : b_i = 1 \wedge \tau_i < h\}$ *be the live inputs. Write* $m(\mathsf{id}_i)$ *for the encoding assigned to the identifier of input* i*, allowing identifiers to repeat among inputs. Assume the multiplicity condition of Definition 3, and let* $w' \leftarrow \mathsf{Merge}(w_1, \ldots, w_m)$*. Then the merged ciphertext encrypts* $\sum_{i \in L} q^{\tau_i} \cdot m(\mathsf{id}_i)$*, and* $\Pr[\mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w') = S_L] \geq 1 - \mathsf{negl}(\lambda)$*, with* S_L *as in that definition.*

Proof. The tag $\mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}_i, b_i, h_i')$ is issued at depth $h - h_i'$, so after t_i applications of Degrade it encrypts $q^{\tau_i} \cdot m(\mathsf{id}_i) \bmod q^h$ by the ElGamal homomorphism. For $\tau_i \geq h$ this value is zero modulo q^h, so expired inputs, like dummy inputs, contribute nothing. Merging via component-wise multiplication therefore encrypts $v = \sum_{i \in L} q^{\tau_i} \cdot m(\mathsf{id}_i) \pmod{q^h}$. If $L = \emptyset$, then $v = 0$ and Trace returns $\perp = S_L$. Otherwise, let $c_{\tau,j}$ count the live inputs with identifier id_j at depth τ. The partial sum at that depth is $d_\tau = \sum_{j=1}^{n} c_{\tau,j} r^{j-1}$. By the stated multiplicity condition, $c_{\tau,j} \leq 2^{h-1} < r$, so $d_\tau \leq \sum_{j=1}^{n} (r-1) r^{j-1} = r^n - 1 < q$. Hence no

base-q carry occurs across depths. The digit at position τ is exactly d_τ, whose base-r digits uniquely identify the live contributions at that depth. Taking the union over all τ yields S_L. □

Theorem 3 (Indistinguishability). *Under DDH in $\mathbb{G}$, the ECT scheme satisfies indistinguishability.*

Proof. Let $m_1 = q^{h-h'} r^{i-1}$ and $m_0 = 0$. On input b, Tag encrypts m_b and signs the resulting ciphertext. Suppose $\mathcal{D}$ distinguishes the two cases with advantage ϵ. Construct an IND-CPA adversary $\mathcal{B}$: on input $\mathsf{pk}_{\mathrm{enc}}$, generate an independent signing key pair $(\mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{sig}})$, answer tagging queries by encrypting under $\mathsf{pk}_{\mathrm{enc}}$ and signing with $\mathsf{sk}_{\mathrm{sig}}$, and submit (m_1, m_0) to the IND-CPA challenger. Upon receiving the challenge ciphertext w, compute $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}_{\mathrm{sig}}, w)$ and forward (w, σ) to $\mathcal{D}$. Since $\mathsf{sk}_{\mathrm{sig}}$ is independent of the IND-CPA challenge, (w, σ) has the same distribution as in the real game. Thus, $\mathcal{B}$ achieves advantage ϵ against IND-CPA. By IND-CPA security of ElGamal under DDH in $\mathbb{G}$, $\epsilon \leq \mathsf{negl}(\lambda)$. □

Theorem 4 (Ephemeral traceability). *The ECT scheme satisfies ephemeral traceability unconditionally. Specifically, for any $\mathsf{id}_i \in [n]$ and any per-tag budget $h' \leq h$, if $(w^{(0)}, \sigma) \leftarrow \mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}_i, 1, h')$ and $w^{(k)}$ is obtained by applying $\mathsf{Degrade}$ at least h' times, then $\mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w^{(k)}) = \perp$.*

Proof. The tag is issued at depth $h - h'$, so after h' applications of $\mathsf{Degrade}$ it encrypts $q^h \cdot m_i \equiv 0 \pmod{q^h}$, since the group has order q^h. As 0 is a fixed point of $m \mapsto qm \bmod q^h$, all further degradations leave the plaintext at 0. Decryption yields 0, so Trace returns $\perp$. The argument is information-theoretic; no computational assumption is used. □

Theorem 5 (Unforgeability). *Under EUF-CMA security of the signature scheme, the exponential ElGamal instantiation satisfies unforgeability: every tag with a valid history traces only to identifiers issued by the authority. A deployment inherits unforgeability when its verification mechanism accepts only tags possessing such a history.*

Proof. Let (w^*, H) be the adversary's output with $(pp, w^*, H) \in R_{\mathsf{hist}}$. A reduction $\mathcal{B}$ against EUF-CMA receives $\mathsf{pk}_{\mathrm{sig}}$ and a signing oracle, generates the encryption key pair itself, and simulates the Tag oracle by encrypting the prescribed plaintext and querying the signing oracle. An issued tag in H that the oracle never produced is a valid forgery, so, except with negligible probability, every issued tag in H encrypts $q^{h-h'} r^{i-1}$ with $\mathsf{id}_i \in \mathcal{Q}$, or zero. We argue by induction along H that every derived tag encrypts a value of the form $\sum_j \kappa_j q^{t_j} r^{i_j - 1}$ with $\mathsf{id}_{i_j} \in \mathcal{Q}$: degradation multiplies by q, a payment copies the degraded sender contribution into at most two outputs, ingestion merges one issued contribution at its issuance depth, and no tag is consumed twice. Since each identifier is issued with $b = 1$ at most once, it has multiplicity at most $2^t \leq 2^{h-1}$ at every live depth t. The choices $r > 2^{h-1}$ and $q > r^n$ prevent all carries, so $\mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w^*)$ returns only identifiers in $\mathcal{Q}$. □

Theorem 6 (Unlinkability). *Under the DDH assumption in $\mathbb{G}$, the ECT scheme satisfies unlinkability. In a ledger deployment, zero knowledge of the host transaction proof additionally hides the consumed account states.*

Proof. Let $\mathcal{A}$ be a PPT adversary submitting (w_0, w_1) and receiving $w' \leftarrow \mathsf{Degrade}(w_b)$ for a uniformly random b. A tag parses as a pair of elements of $\mathbb{G}$, membership being publicly checkable via $x^{q^h} = 1$, so we may assume $w_0, w_1 \in \mathbb{G} \times \mathbb{G}$. Write $w_b = (A_b, B_b)$. Degradation returns $w' = \big(A_b^q \cdot g^{r'},\ B_b^q \cdot \mathsf{pk}_{\mathrm{enc}}^{r'}\big)$ for uniform $r' \in \mathbb{Z}_{q^h}$, that is, w_b raised componentwise to the q-th power and multiplied by the rerandomization factor $(g^{r'}, \mathsf{pk}_{\mathrm{enc}}^{r'})$. Construct a distinguisher $\mathcal{B}$ for DDH: on challenge (g, X, Y, Z), set $\mathsf{pk}_{\mathrm{enc}} \leftarrow X$, answer tagging queries by encrypting under X and signing with a self-generated key, sample b, and give $\mathcal{A}$ the challenge $w' = (A_b^q \cdot Y,\ B_b^q \cdot Z)$; output 1 if $\mathcal{A}$ guesses b. If (g, X, Y, Z) is a Diffie–Hellman tuple, then $Y = g^{r'}$ and $Z = X^{r'}$ for uniform r', and w' is distributed exactly as $\mathsf{Degrade}(w_b)$. If Z is instead uniform, then (Y, Z) is uniform on $\mathbb{G} \times \mathbb{G}$, so w' is uniform and independent of b, and $\mathcal{A}$ guesses b with probability exactly $\frac{1}{2}$. The advantage of $\mathcal{A}$ is therefore bounded by the DDH advantage of $\mathcal{B}$, and $\Pr[b' = b] \le \frac{1}{2} + \mathsf{negl}(\lambda)$. □

5.4 Damgård–Jurik instantiation

When a deployment requires more simultaneous traces than the ElGamal construction efficiently supports, Damgård–Jurik is a natural choice. It trades larger tags for a larger plaintext space, polynomial-time tracing, and security under DCR. Its plaintext ring is $\mathbb{Z}_{N^s}$, where N is an RSA modulus, and the ciphertext space is $\mathbb{Z}^*_{N^{s+1}}$. The modulus itself plays the role of the nilpotent element: raising a ciphertext to the power N multiplies its plaintext by N modulo N^s,

$$c^N = (1+N)^{Nm}\,\rho^{N \cdot N^s} = (1+N)^{Nm}\,(\rho^N)^{N^s} = \mathsf{Enc}\big(N,\ Nm \bmod N^s;\ \rho^N\big),$$

so setting $s = h$ gives exact expiry after h hops. A tag is a single element of $\mathbb{Z}^*_{N^{h+1}}$, degradation is $w \mapsto w^N \rho^{N^h} \bmod N^{h+1}$ for a fresh ρ, and merging is multiplication modulo N^{h+1}. Tracing decrypts to $v \in \mathbb{Z}_{N^h}$, reads the base-N digits as degradation depths and the base-r digits within each as identifiers, and returns $\perp$ if the digit bound $d_t < r^n$ fails.

The plaintext space is a full residue ring rather than a discrete-log space, so tracing costs $O(nh)$ modular operations instead of a bounded search, and the capacity grows with $\log_r N$ rather than with $\log_r q$. The same six properties hold: correctness and ephemeral traceability by the same digit argument as above, indistinguishability and unlinkability under DCR in place of DDH, and unforgeability again under EUF-CMA alone. We now give the assumption, the six algorithms, and the proofs. Identifiers use $m_i = r^{i-1}$ with $r > 2^{h-1}$ and $N > r^n$, so base N separates degradation depths and base r separates identifiers within each depth.

Definition 10 (Decisional composite residuosity). *Let $N = p_1 p_2$ be an RSA modulus and $s \geq 1$. The* decisional composite residuosity (DCR) assumption *holds if for all PPT distinguishers $\mathcal{D}$:*

$$\left|\Pr[\mathcal{D}(N, z) = 1] - \Pr[\mathcal{D}(N, \rho^{N^s} \bmod N^{s+1}) = 1]\right| \leq \mathsf{negl}(\lambda)$$

where $z \xleftarrow{\$} \mathbb{Z}^*_{N^{s+1}}$ and $\rho \xleftarrow{\$} \mathbb{Z}^*_{N^{s+1}}$, that is, a uniform element of $\mathbb{Z}^*_{N^{s+1}}$ cannot be distinguished from a uniform N^s-th residue.

Damgård–Jurik is IND-CPA secure under DCR for every s, with the assumption equivalent to the $s = 1$ case [24]. We again use $m_i = r^{i-1}$, choose $r > 2^{h-1}$ and $N > r^n$, and let base N separate depths. Setup (Alg. 7) samples the modulus and fixes $s = h$.

Algorithm 7. Setup.

$\mathsf{Setup}(1^\lambda, n, h, r) \to (pp, \mathsf{sk}_{\mathsf{trace}})$
Let r be an integer with $r > 2^{h-1}$; `// prevents multiplicity carries`
Sample distinct primes p_1, p_2 and set $N \leftarrow p_1 p_2$, with $N > r^n$ and $|N|$ sized for λ-bit factoring security;
$s \leftarrow h$; `// plaintext ring` $\mathbb{Z}_{N^h}$`, ciphertext ring` $\mathbb{Z}^*_{N^{h+1}}$
$\mathsf{pk}_{\mathrm{enc}} \leftarrow N, \quad \mathsf{sk}_{\mathrm{enc}} \leftarrow \lambda_N = \mathrm{lcm}(p_1 - 1, p_2 - 1)$;
$(\mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{sig}}) \leftarrow \mathsf{SigKeyGen}(1^\lambda)$;
Assign $m_i \leftarrow r^{i-1}$ for each user id_i, $i \in [n]$;
$pp \leftarrow (N, h, n, r, \mathsf{pk}_{\mathrm{sig}}), \quad \mathsf{sk}_{\mathsf{trace}} \leftarrow (\mathsf{sk}_{\mathrm{enc}}, \mathsf{sk}_{\mathrm{sig}})$;
return $(pp, \mathsf{sk}_{\mathsf{trace}})$;

The public tag space $\mathbb{Z}_{N^{h+1}}$ exposes the hop budget, while $n_{\max} = \lfloor \log_r N \rfloor$ exposes the identifier-slot capacity. Enlarging either bound enlarges every tag.

Tag (Alg. 8) issues at depth $h - h'$, encrypts zero for a dummy, and signs the resulting ciphertext, which Verify (Alg. 9) checks against the authority's issuance key.

Algorithm 8. Tag issuance.

$\mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}_i, b, h') \to (w, \sigma)$
$m \leftarrow N^{h-h'} \cdot r^{i-1} \cdot b$;
$\rho \xleftarrow{\$} \mathbb{Z}^*_{N^{h+1}}$;
$w \leftarrow (1+N)^m \rho^{N^h} \bmod N^{h+1}$;
$\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}_{\mathrm{sig}}, w)$;
return (w, σ);

Algorithm 9. Verification.

$\mathsf{Verify}(w, \sigma) \to \{0, 1\}$
return $\mathsf{Ver}(\mathsf{pk}_{sig}, w, \sigma)$;

Degrade (Alg. 10) implements $m \mapsto Nm \bmod N^h$ and returns the rerandomization value ρ as ξ. Merge (Alg. 11) multiplies ciphertexts, adding their plaintexts modulo N^h.

Algorithm 10. Degradation.

$\mathsf{Degrade}(w) \to (w', \xi)$
- $\rho \xleftarrow{\$} \mathbb{Z}^*_{N^{h+1}}$;
- $w' \leftarrow w^N \cdot \rho^{N^h} \bmod N^{h+1}$;
- **return** $(w', \xi = \rho)$;

Algorithm 11. Merging.

$\mathsf{Merge}(w_1, \ldots, w_m) \to w'$
- $w' \leftarrow \prod_{j=1}^{m} w_j \bmod N^{h+1}$;
- **return** w';

In the ledger, the transaction statement enforces $w' = w^N \rho^{N^h} \bmod N^{h+1}$ for each degradation and enforces multiplication for each merge.

Trace (Alg. 12) decrypts v directly, then reads base-N depths and base-r identifiers in $O(nh)$ digit operations. The value $v = 0$ denotes either a dummy or an expired tag. A digit $d_t \geq r^n$ cannot arise from a valid transition history, and Trace then returns $\perp$.

Algorithm 12. Tracing.

$\mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w) \to S$ *or* $\perp$
- $v \leftarrow \mathsf{Dec}(\mathsf{sk}_{\mathrm{enc}}, w) \in \mathbb{Z}_{N^h}$;
- **if** $v = 0$ **then return** $\perp$;
- Write $v = \sum_{t=0}^{h-1} d_t \cdot N^t$ in base N;
- **if** $d_t \geq r^n$ *for some* t **then return** $\perp$;
- **foreach** t *with* $d_t \neq 0$ **do**
  - Decompose $d_t = \sum_{i=1}^{n} c_{t,i} \cdot r^{i-1}$ in base r;
  - $S_t \leftarrow \{\mathsf{id}_i : i \in [n],\, c_{t,i} > 0\}$;
- **return** $\bigcup_t S_t$;

Unforgeability uses EUF-CMA signatures; the other security proofs use DCR.

Theorem 7 (Trace correctness, Damgård–Jurik). *For any* $\mathsf{id}_i \in [n]$*, any per-tag budget* $h' \leq h$*, and any* $t < h'$*, let* $(w, \sigma) \leftarrow \mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}_i, 1, h')$ *and let* w' *be obtained by applying* Degrade *exactly* t *times. Then* $\mathsf{id}_i \in \mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w')$ *with probability* $1 - \mathsf{negl}(\lambda)$.

Proof. Raising a ciphertext to N multiplies the plaintext by N in $\mathbb{Z}_{N^h}$, so the tag, which encrypts $N^{h-h'} r^{i-1}$, encrypts $N^{h-h'+t} r^{i-1} \bmod N^h$ after t degradations. Since $r^{i-1} \leq r^{n-1} < N$ by the constraint $N > r^n$, and $h - h' + t < h$, the integer $N^{h-h'+t} r^{i-1}$ is positive and strictly below $N^{h-h'+1} \leq N^h$, hence nonzero in $\mathbb{Z}_{N^h}$. Its base-N digit at position $h - h' + t$ is r^{i-1} and all other digits vanish;

decomposing that digit in base r gives a unit coefficient at position $i-1$, so Trace returns id_i. □

Theorem 8 (Merge correctness, Damgård–Jurik). *For $i \in [m]$, let $w_i \leftarrow \mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}_i, b_i, h'_i)$, and suppose each w_i has been degraded $t_i \geq 0$ times before merging; write $\tau_i = (h - h'_i) + t_i$ for its current depth and let $L = \{i : b_i = 1 \wedge \tau_i < h\}$ be the live inputs. Write $m(\mathsf{id}_i)$ for the encoding assigned to the identifier of input i, allowing identifiers to repeat among inputs. Assume the multiplicity condition of Definition 3, and let $w' \leftarrow \mathsf{Merge}(w_1, \ldots, w_m)$. Then w' encrypts $\sum_{i \in L} N^{\tau_i} m(\mathsf{id}_i) \bmod N^h$, and*

$$\Pr[\mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w') = S_L] \geq 1 - \mathsf{negl}(\lambda),$$

with S_L as in that definition.

Proof. Componentwise multiplication adds plaintexts modulo N^h. An input with $\tau_i \geq h$ contributes zero modulo N^h, so expired inputs, like dummy inputs, vanish, which gives the stated value v. If $L = \emptyset$, then $v = 0$ and Trace returns $\perp = S_L$. Otherwise, let $c_{\tau,j}$ count the live inputs with identifier id_j at depth τ. Their contribution at that depth is $d_\tau = \sum_{j=1}^{n} c_{\tau,j} r^{j-1}$. The multiplicity condition gives $c_{\tau,j} \leq 2^{h-1} < r$, and therefore $d_\tau \leq \sum_{j=1}^{n} (r-1) r^{j-1} = r^n - 1 < N$. Every depth occupies one base-N digit with no carry, and the base-r digits uniquely identify its contributions. Taking the union over τ yields S_L. □

Theorem 9 (Indistinguishability, Damgård–Jurik). *Under DCR (Definition 10), the Damgård–Jurik instantiation satisfies indistinguishability.*

Proof. A tag issued with $b = 1$ is $\mathsf{Enc}(N, N^{h-h'} r^{i-1})$ and a tag issued with $b = 0$ is $\mathsf{Enc}(N, 0)$, so a distinguisher between them yields an IND-CPA adversary. On input the public key N, the adversary $\mathcal{B}$ generates an independent signing key pair, answers tagging queries by encrypting under N and signing with $\mathsf{sk}_{\mathrm{sig}}$, submits the challenge messages $(N^{h-h'} r^{i-1}, 0)$, signs the challenge ciphertext w with $\mathsf{sk}_{\mathrm{sig}}$, and forwards (w, σ) to the distinguisher. The signing key is independent of the challenge, so the simulated pair is distributed as in the real game and $\mathcal{B}$ inherits the distinguisher's advantage. Damgård–Jurik is IND-CPA secure under DCR [24], so that advantage is negligible. □

Theorem 10 (Ephemeral traceability, Damgård–Jurik). *For any $\mathsf{id}_i \in [n]$ and any per-tag budget $h' \leq h$, if $(w^{(0)}, \sigma) \leftarrow \mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}_i, 1, h')$ and $w^{(k)}$ is obtained by applying $\mathsf{Degrade}$ at least h' times, then $\mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w^{(k)}) = \perp$. The statement holds unconditionally and for every modulus N.*

Proof. The tag is issued at depth $h - h'$, so after h' degradations it encrypts $N^h m_i \equiv 0 \pmod{N^h}$, and 0 is a fixed point of $m \mapsto Nm \bmod N^h$, so every further degradation leaves the plaintext at 0 and Trace returns $\perp$. The identity $N^h \equiv 0 \pmod{N^h}$ holds for every integer N, so the argument is information-theoretic and uses no property of the public parameters beyond the value of h fixed by the ciphertext ring. □

Theorem 11 (Unforgeability, Damgård–Jurik). *Under EUF-CMA security of the signature scheme, the Damgård–Jurik instantiation satisfies unforgeability: every tag with a valid history traces only to identifiers issued by the authority. A deployment inherits unforgeability when its verification mechanism accepts only tags possessing such a history.*

Proof. A reduction against EUF-CMA simulates the Tag oracle with the signing oracle and its own encryption keys. Thus, for every $(pp, w^*, H) \in R_{\mathsf{hist}}$, each issued tag in H was, except with negligible probability, produced by the oracle and encrypts $N^{h-h'} r^{i-1}$ with $\mathsf{id}_i \in \mathcal{Q}$, or zero. Induction along H shows that w^* encrypts a sum $\sum_j \kappa_j N^{t_j} r^{i_j - 1}$ with every $\mathsf{id}_{i_j} \in \mathcal{Q}$: degradation multiplies by N, a payment copies the degraded sender contribution into at most two outputs, ingestion merges one issued contribution at its issuance depth, and no tag is consumed twice. Since each identifier is issued with $b = 1$ at most once, it has multiplicity at most $2^{h-1} < r$ at every live depth. Together with $N > r^n$, this prevents all digit carries, so tracing returns only identifiers in $\mathcal{Q}$. □

Theorem 12 (Unlinkability, Damgård–Jurik). *Under DCR (Definition 10), the Damgård–Jurik instantiation satisfies unlinkability. In a ledger deployment, zero knowledge of the host transaction proof additionally hides the consumed account states.*

Proof. The adversary submits (w_0, w_1) and receives $w' \leftarrow \mathsf{Degrade}(w_b)$ for a uniform bit b. A tag parses as an element of $\mathbb{Z}^*_{N^{h+1}}$, membership being publicly checkable via a gcd with N, so we may assume $w_0, w_1 \in \mathbb{Z}^*_{N^{h+1}}$. Degradation returns $w' = w_b^N \rho^{N^h} \bmod N^{h+1}$ for uniform $\rho \in \mathbb{Z}^*_{N^{h+1}}$. Construct a distinguisher $\mathcal{B}$ for DCR with $s = h$: on challenge (N, z), answer tagging queries by encrypting under N and signing with a self-generated key, sample b, give $\mathcal{A}$ the challenge $w' = w_b^N \cdot z \bmod N^{h+1}$, and output 1 if $\mathcal{A}$ guesses b. If $z = \rho^{N^h}$ is a random N^h-th residue, w' is distributed exactly as $\mathsf{Degrade}(w_b)$. If z is uniform in $\mathbb{Z}^*_{N^{h+1}}$, then $w' = w_b^N \cdot z$ is uniform in $\mathbb{Z}^*_{N^{h+1}}$ and independent of b, since multiplication by the fixed unit w_b^N permutes the group; $\mathcal{A}$ then guesses b with probability exactly $\frac{1}{2}$. The advantage of $\mathcal{A}$ is therefore bounded by the DCR advantage of $\mathcal{B}$, and $\Pr[b' = b] \le \frac{1}{2} + \mathsf{negl}(\lambda)$. □

5.5 Ring-LWE instantiation

Ring-LWE is the post-quantum alternative and, like Damgård–Jurik, supports more distinguishable live identifiers per tag than the ElGamal construction. Its larger identifier-slot capacity and post-quantum security come at the cost of substantially larger tags. The construction works over

$$R \;=\; \mathbb{Z}[X,Y] \big/ \big(X^{N_0} + 1,\; Y^h\big) \;=\; R_0[Y]/(Y^h), \qquad R_0 = \mathbb{Z}[X]/(X^{N_0} + 1),$$

where the second variable supplies the nilpotent element outright: $Y^h = 0$ holds in R by construction, so the hop budget is baked into the public ring rather than into the order of a group. Tags are BGV ciphertexts over R, degradation is

$(c_0, c_1) \mapsto (Yc_0,\, Yc_1) + \mathsf{Enc}(\mathsf{pk}_{\mathrm{enc}}, 0)$, and merging is ciphertext addition. Only additive homomorphism and multiplication by the public constant Y are ever used, so plain IND-CPA BGV suffices, with neither key switching nor modulus switching.

Two features distinguish this instantiation from the previous two. First, choosing the plaintext modulus such that $t \equiv 1 \pmod{2N_0}$ splits the plaintext ring into N_0 CRT slots, one per identifier. Thus, the positional base-r encoding of Definition 1 is unnecessary: the capacity is $n_{\max} = N_0$, and the only remaining constraint is that no slot wraps, namely $t > 2^{h-1}$. Second, nilpotency erases noise together with messages. After degradation, the invariant noise is $Y \cdot e_{\mathrm{old}} + e_{\mathrm{fresh}}$. Noise older than h hops therefore vanishes, and a degradation chain of any length contains at most h live rerandomization terms. Hardness is stated as decision RLWE over R, which is not cyclotomic but decomposes over the cyclotomic ring R_0. We now give the ring, the assumption, the algorithms, and the proofs. Our Ring-LWE instantiation works over the nonstandard ring

$$R := \frac{\mathbb{Z}[X]}{(X^{N_0}+1)} \otimes_{\mathbb{Z}} \frac{\mathbb{Z}[Y]}{(Y^h)} \;\cong\; \mathbb{Z}[X,Y]/(X^{N_0}+1, Y^h).$$

We write $R_0 := \mathbb{Z}[X]/(X^{N_0}+1)$, so that $R = R_0[Y]/(Y^h)$. Every element has a unique representation $\sum_{i=0}^{h-1} c_i Y^i$ with $c_i \in R_0$. Multiplication by Y shifts these coefficients and maps every plaintext to zero after h applications. Identifiers are encoded in CRT slots of the X component, rather than by the base-r positional encoding used in the two preceding instantiations.

We use the coefficient ℓ_∞ norm on R, written $\|a\|_\infty$ or simply $\|a\|$. Let δ_R denote an expansion bound satisfying $\|ab\|_\infty \le \delta_R \|a\|_\infty \|b\|_\infty$ for the products considered below.

As an R_0-module, R is free of rank h with basis $1, Y, \ldots, Y^{h-1}$, so every element can be written uniquely as $u = \sum_{j<h} u_j Y^j$. In the following we will refer to the coefficients $u_j \in R_0$ as the *blocks* of u.

We use two elementary facts about R.

Lemma 2 (Nilpotent shift). *Let $\mu_Y : R \to R$ be the map $m \mapsto Ym$. For every nonzero $m_0 \in R_0$ and $0 \le d < h$, the value $\mu_Y^d(m_0) = m_0 Y^d$ is nonzero and lies in block d. Moreover, μ_Y^h is the zero map and $\mu_Y(0) = 0$. Thus μ_Y is nilpotent of index exactly h, and zero is absorbing.*

Proof. The blocks of an element of Y-degree $< h$ are its canonical coordinates, so $m_0 Y^d \neq 0$ for $d < h$; and $m_0 Y^h = m_0 \cdot 0 = 0$ since $Y^h = 0$ in R. The final claim, that zero is absorbing, is immediate.

Lemma 3 (Degradation). *In $R = R_0[Y]/(Y^h)$, for every $e \in R$ and $d \ge 0$: $\|Y^d e\| \le \|e\|$.*

Proof. Consider the case for $d = 1$. Write $e = \sum_{j=0}^{h-1} e_j Y^j$ with blocks $e_j \in R_0$. Then $Ye = \sum_{j=0}^{h-2} e_j Y^{j+1}$: the blocks shift up by one and the top block goes to 0 (by Lemma 2). Hence $\|Ye\| = \max_{j \le h-2} \|e_j\| \le \|e\|$. The case for general d follows by induction.

Next we define the Ring-LWE problem over rings of the above kind.

Definition 11 (Decision Ring-LWE over a ring). *Let R be a commutative ring and Q a positive integer such that $R_Q = R/QR$ is finite. Let χ_s, χ_e be distributions on R. The* decision RLWE assumption $\mathrm{RLWE}_{R,Q,\chi_s,\chi_e}$ *holds if for every PPT distinguisher $\mathcal{D}$ and every $m = \mathrm{poly}(\lambda)$,*

$$\left|\Pr\big[\mathcal{D}\big(\{(a_i,\ a_i s + e_i)\}_{i\in[m]}\big) = 1\big] - \Pr\big[\mathcal{D}\big(\{(a_i,\ u_i)\}_{i\in[m]}\big) = 1\big]\right| \le \mathsf{negl}(\lambda),$$

where $s \leftarrow \chi_s$ is drawn once and $a_i, u_i \xleftarrow{\$} R_Q$ and $e_i \leftarrow \chi_e$ are drawn independently for each i.

Remark 1. Standard Ring-LWE is typically defined over a cyclotomic ring, where its hardness is well studied. Although R is not cyclotomic, it is built over the cyclotomic ring R_0. Section 5.6 uses this structure to relate the assumption over R to standard Ring-LWE over R_0.

Remark 2. Classical hardness reductions use a uniform secret and a suitably narrow discrete Gaussian error; standard transformations also support short secrets. Efficient implementations often use a uniform ternary secret together with discrete Gaussian error. Our proofs require only the stated Ring-LWE assumption and blockwise independence, while the concrete estimates follow the ternary-secret parameterization of [1].

Related variants in the literature. Our assumption is closest to multivariate Ring-LWE, introduced and developed by Pedrouzo-Ulloa, Troncoso-Pastoriza, and Pérez-González [45,46,47,44]. They consider rings of the form $\mathbb{Z}_q[x,y]/(f(x), g(y))$, where $f(x) = x^{n_1} + 1$ and $g(y) = y^{n_2} + 1$ are power-of-two cyclotomic polynomials. This structure was claimed to provide hardness equivalent to RLWE over $\mathbb{Z}_q[z]/(z^{n_1 n_2} + 1)$. Bootland, Castryck, and Vercauteren showed that this claim is false when n_2 divides n_1 [8]: evaluating y at suitable roots in $\mathbb{Z}_q[x]/(x^{n_1} + 1)$ maps multivariate samples to native dimension-n_1 RLWE samples with only $O(\sqrt{n_2})$ error growth.

Our second component $\mathbb{Z}[Y]/(Y^h)$ is instead non-cyclotomic and has only the repeated root 0. The corresponding evaluation map is the projection $Y \mapsto 0$, which exposes the cyclotomic base component but not the remaining Y-blocks. This explains both the connection to standard Ring-LWE and why the multivariate attack above does not directly recover the full secret in our ring.

The encryption scheme. For this instantiation, we use the BGV encryption scheme [9], which supports fully homomorphic encryption. We need only additive homomorphism and multiplication by a fixed public ring element, so we describe the simplified scheme below.

The scheme is defined over a ring R with moduli Q and a parameter t satisfying $\gcd(t, Q) = 1$, with χ_s and χ_e as above. Here R_Q serves as the ciphertext ring, while $P = R/tR$ is the plaintext ring. We then have: We write $\mathsf{Enc}(\mathsf{pk}, m; \xi)$ when

Algorithm 13. BGV, simplified.

$\mathsf{KeyGen}(R, Q, t)$: $s \leftarrow \chi_s$ (ternary); $a \leftarrow R_Q$ uniform; $e \leftarrow \chi_e$
$\mathsf{pk} \leftarrow (p_0, p_1) := (-(as + te),\ a) \in R_Q^2$; $\mathsf{sk} \leftarrow s$
$\mathsf{Enc}(\mathsf{pk}, m)$: for $m \in P$
$u \leftarrow \chi_s$; $e_0, e_1 \leftarrow \chi_e$
$\mathrm{ct} \leftarrow (c_0, c_1) := (p_0 u + te_0 + m,\ p_1 u + te_1) \in R_Q^2$
$\mathsf{Dec}(\mathsf{sk}, c = (c_0, c_1))$:
$\mu \leftarrow [c_0 + c_1 \cdot s]_Q$; output $\mu \bmod t$

the encryption randomness $\xi = (u, e_0, e_1)$ is made explicit, and $\Omega_{\mathsf{Enc}} = \chi_s \times \chi_e^2$ for the distribution it is drawn from. A pair $c = (c_0, c_1) \in R_Q^2$ is a valid encryption of $m \in P$ when we have $c_0 + c_1 s \equiv m + te \pmod{Q}$ for some small *invariant noise* $e \in R$, and decryption is correct as long as $\|m + te\|_\infty < Q/2$. Adding ciphertexts adds plaintexts and adds noise; multiplying by a public $\alpha \in R$ componentwise, $\mathsf{CMul}(\alpha, c) = (\alpha c_0, \alpha c_1)$, multiplies both the plaintext and the noise term by α.

Representation of the plaintext ring. We set the plaintext modulus t to be a prime with $t \equiv 1 \pmod{2N_0}$. Then the polynomial $X^{N_0} + 1$ splits into N_0 distinct linear factors modulo t, and thus the Chinese remainder theorem gives a ring isomorphism $\mathbb{Z}_t[X]/(X^{N_0} + 1) \cong \mathbb{Z}_t^{N_0}$ [52], and so the plaintext ring can be reinterpreted as

$$P \ \cong\ \left(\mathbb{Z}_t[Y]/(Y^h)\right)^{N_0}. \tag{1}$$

Each factor on the right is a local ring with maximal nilpotent ideal (Y), and we associate one factor with each identifier. Identifier id_i is assigned slot i, and $\mathsf{sl}(i) \in \mathbb{Z}_t[X]/(X^{N_0} + 1)$ denotes the plaintext that is 1 in slot i and 0 elsewhere. We write $\mathsf{SlotDecode}$ for the CRT map of Equation (1), applied blockwise: on the depth-d block it is the isomorphism $\mathbb{Z}_t[X]/(X^{N_0} + 1) \cong \mathbb{Z}_t^{N_0}$. Since this is a ring isomorphism, addition acts on slots coordinatewise and independently, so a merged tag holds in slot i precisely the count of appearances of id_i in the merged ciphertext. The only constraint needed to keep the count accurate now is that no slot wraps modulo t, which is the condition $t > 2^{h-1}$. Next we describe in sequence the various algorithms in this instantiation of our tracing scheme.

The algorithms. The setup algorithm Setup (Alg. 14) fixes the ring and generates a BGV key pair for tag encryption, along with a signature key pair for authenticating freshly issued tags. The parameters N_0, t, Q are chosen jointly: N_0 and the error distributions determine Ring-LWE security, while Q must exceed twice the largest plaintext-plus-noise norm permitted by the deployment's fan-in and correctness horizon. Section 5.7 gives the constraints in full, together with concrete parameters. Both tracing bounds are auditable from *pp*. The relation $Y^h = 0$ is part of the public ring and fixes the hop budget, while the degree of $X^{N_0} + 1$ fixes the identifier-slot capacity $n_{\max} = N_0$; any entity receiving *pp* can check both. Changing h changes the ring and therefore requires fresh

Algorithm 14. Setup.

$\mathsf{Setup}(1^\lambda, n, h) \to (pp, \mathsf{sk}_{\mathsf{trace}})$
Choose N_0 a power of two with $N_0 \geq n$;
Choose a suitably large prime t with $t \equiv 1 \pmod{2N_0}$;
Choose Q with $Q \equiv 1 \pmod{2N_0}$ and $\gcd(t, Q) = 1$, sized per Section 5.7;
$R \leftarrow \mathbb{Z}[X, Y]/(X^{N_0} + 1, Y^h)$; // $P = R/tR$, ciphertexts in R_Q^2
$(\mathsf{pk}_{\mathsf{enc}}, \mathsf{sk}_{\mathsf{enc}}) \leftarrow \mathsf{KeyGen}(R, Q, t)$;
$(\mathsf{pk}_{\mathsf{sig}}, \mathsf{sk}_{\mathsf{sig}}) \leftarrow \mathsf{SigKeyGen}(1^\lambda)$;
Assign slot i to identifier id_i for each $i \in [n]$;
$pp \leftarrow (R, Q, t, h, n, N_0, \mathsf{pk}_{\mathsf{enc}}, \mathsf{pk}_{\mathsf{sig}})$, $\mathsf{sk}_{\mathsf{trace}} \leftarrow (\mathsf{sk}_{\mathsf{enc}}, \mathsf{sk}_{\mathsf{sig}})$;
return $(pp, \mathsf{sk}_{\mathsf{trace}})$;

public parameters. Unlike exponential ElGamal, however, the hop budget does not enter the hardness assumption directly: by Theorem 19, security is governed by the base-ring dimension N_0 and the modulus Q. It enters indirectly, since $t > 2^{h-1}$ raises the plaintext modulus and hence Q, so holding the security level fixed calls for a larger N_0 in turn; Section 5.7 makes this dependence explicit. A larger h also increases tag size and arithmetic cost. The tag algorithm Tag (Alg. 15) realizes the budget $h' \leq h$ by issuing at depth $h - h'$, exactly as in the two previous instantiations. As before, the authority signs the ciphertext to anchor its ledger history, and Verify (Alg. 16) checks this signature.

Algorithm 15. Tag issuance.

$\mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}_i, b, h') \to (w, \sigma)$
$m \leftarrow Y^{h-h'} \cdot \mathsf{sl}(i) \cdot b$;
$w \leftarrow \mathsf{Enc}(\mathsf{pk}_{\mathsf{enc}}, m)$;
$\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}_{\mathsf{sig}}, w)$;
return (w, σ);

Algorithm 16. Verification.

$\mathsf{Verify}(w, \sigma) \to \{0, 1\}$
return $\mathsf{Ver}(\mathsf{pk}_{\mathsf{sig}}, w, \sigma)$;

The degrade algorithm $\mathsf{Degrade}$ (Alg. 17) multiplies both ciphertext components by Y and rerandomizes, implementing $m \mapsto Ym$ in P. It returns the randomness of the fresh encryption of zero as ξ. The algorithm uses no secret and, by Lemma 3, adds no noise beyond that rerandomization, whose level Section 5.7 fixes. The merging algorithm Merge (Alg. 18) adds ciphertexts componentwise in R_Q^2, thereby adding their plaintexts in P. Its correctness is enforced by the transaction statement.

Algorithm 17. Degradation.

$\mathsf{Degrade}(w) \to (w', \xi)$
  Parse $w = (c_0, c_1)$;
  $\xi \xleftarrow{\$} \Omega_{\mathsf{Enc}}$;
  $w' \leftarrow (Yc_0,\ Yc_1) + \mathsf{Enc}(\mathsf{pk}_{\mathsf{enc}}, 0; \xi)$;
  return (w', ξ);

Algorithm 18. Merging.

$\mathsf{Merge}(w_1, \ldots, w_m) \to w'$
  $w' \leftarrow \sum_{j=1}^{m} w_j$;
  return w';

The tracing algorithm Trace (Alg. 19) decrypts and then checks the result: nonempty block indices record the degradation depth of various trackers, and nonempty slots record the tracked identifier. A tag decrypting to 0 denotes either a dummy or a fully expired tag. Tracing costs one decryption together with h inverse transforms modulo t, hence $O(N \log N)$ operations in total, and unlike Section 5.3 its cost does not depend on how many identifiers are being tracked in total. Finally, we establish the six properties of Section 4.2 for this instantiation.

Algorithm 19. Tracing.

$\mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w) \to S$ *or* $\perp$
  $v \leftarrow \mathsf{Dec}(\mathsf{sk}_{\mathsf{enc}}, w) \in P$;
  if $v = 0$ **then return** $\perp$;
  Write $v = \sum_{d=0}^{h-1} v_d Y^d$ with $v_d \in \mathbb{Z}_t[X]/(X^{N_0} + 1)$;
  foreach d *with* $v_d \neq 0$ **do**
    $(c_{d,1}, \ldots, c_{d,n}) \leftarrow \mathsf{SlotDecode}(v_d)$;
    if $c_{d,i} > 2^{h-1}$ *for some* $i \in [n]$ **then return** $\perp$;
    $S_d \leftarrow \{\mathsf{id}_i : i \in [n],\ c_{d,i} \neq 0\}$;
  return $\bigcup_d S_d$;

Unforgeability relies on EUF-CMA security of the issuance signature; the other security arguments rely on Ring-LWE over R (Definition 11). Decryption is correct only for tags whose history stays within the correctness horizon of Setup, that is, the bounds on transitions and fan-in of Section 5.7; merge correctness and unforgeability are stated relative to this horizon.

Theorem 13 (Trace correctness, Ring-LWE). *For any* id_i *with* $i \in [n]$*, any per-tag budget* $h' \leq h$ *and any* $k < h'$*, let* $(w, \sigma) \leftarrow \mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}_i, 1, h')$ *and let* w' *be obtained by applying* Degrade *exactly* k *times. Then* $\mathsf{id}_i \in \mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w')$ *with probability* $1 - \mathsf{negl}(\lambda)$.

Proof. Componentwise multiplication by Y multiplies the plaintext by Y in P, and adding an encryption of zero leaves it unchanged, so after k degradations the tag encrypts $Y^{\,h-h'+k}\mathsf{sl}(i)$. Since $k < h'$ we have $h - h' + k < h$, so this element is nonzero: the corresponding block at depth $h - h' + k$ is $\mathsf{sl}(i)$ and all other blocks

are zero. Lemma 3 shows that multiplication by Y does not increase noise. The bound on Q from Section 5.7 covers the at most h live rerandomization terms except with negligible probability, so decryption returns $v = Y^{h-h'+k}\mathsf{sl}(i)$. Slot decoding of $v_{h-h'+k}$ yields $c_{d,i} = 1$ and $c_{d,i'} = 0$ for $i' \neq i$, so Trace returns id_i.

Theorem 14 (Merge correctness, Ring-LWE). *For $i \in [m]$, let $w_i \leftarrow \mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}_i, b_i, h'_i)$ and suppose that w_i is degraded $k_i \geq 0$ times before merging. Write $L = \{i : b_i = 1 \wedge k_i < h'_i\}$ and $\tau_i = (h - h'_i) + k_i$ for $i \in L$. Assume the multiplicity condition of Definition 3. If $w' \leftarrow \mathsf{Merge}(w_1, \ldots, w_m)$ and the history of w' stays within the correctness horizon, then w' encrypts $\sum_{i \in L} Y^{\tau_i}\mathsf{sl}(\mathsf{id}_i)$, where $\mathsf{sl}(\mathsf{id}_i)$ denotes the slot assigned to the identifier of input i, and*

$$\Pr[\mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w') = S_L] \geq 1 - \mathsf{negl}(\lambda),$$

with S_L as in Definition 3.

Proof. Every dummy input contributes zero, as does every input with $k_i \geq h'_i$ because its plaintext contains a factor Y^h. Componentwise addition therefore gives the stated plaintext v, and within the correctness horizon decryption returns v except with negligible probability. If $L = \emptyset$, then $v = 0$ and Trace returns $\perp = S_L$, so assume $L \neq \emptyset$. Fix a depth τ and let $c_{\tau,j}$ denote the number of live inputs with identifier id_j at that depth, so that the block v_τ equals $\sum_{j=1}^{n} c_{\tau,j}\,\mathsf{sl}(j)$ and slot j in it holds $c_{\tau,j}$ reduced modulo t. We now observe that: (i) for a given block, its contents can only be driven up in depth till it degrades to zero, but not otherwise modified, and (ii) for a given slot, the value held in it can only be driven up in multiplicity by merging (till it wraps around mod t), but not otherwise modified. Two separations act independently here. The blocks are module coordinates: multiplication by Y shifts them and addition never carries between them, so contributions at different depths cannot collide. The slots are components of the ring isomorphism (1): addition acts coordinatewise, so contributions of different identifiers cannot collide either. It therefore suffices that no slots wrap, that is, $c_{\tau,j} < t$ for every $j \in [n]$. The stated multiplicity condition gives $c_{\tau,j} \leq 2^{h-1} < t$. Hence $\mathsf{SlotDecode}(v_\tau)$ returns the exact multiplicities and uniquely determines $S_\tau = \{\mathsf{id}_i : i \in L,\ \tau_i = \tau\}$. Taking the union over all depths yields S_L.

Note that no analogue of the constraint $r^n < q$ is required: only the per-slot multiplicity is bounded, so capacity and multiplicity are decoupled and the plaintext modulus can stay small however many identifiers are traced at once.

Theorem 15 (Indistinguishability, Ring-LWE). *The Ring-LWE-based tracing scheme satisfies indistinguishability assuming that* $\mathrm{RLWE}_{R,Q,\chi_s,\chi_e}$ *(Definition 11) holds.*

Proof. The algorithm Tag returns (w, σ) where $w = \mathsf{Enc}(\mathsf{pk}_{\mathsf{enc}}, Y^{h-h'}\mathsf{sl}(i))$ when $b = 1$, and returns an encryption of 0 when $b = 0$. Suppose there is a PPT distinguisher $\mathcal{D}$ that distinguishes the two with advantage ϵ. Construct an IND-CPA adversary $\mathcal{B}$: on input $\mathsf{pk}_{\mathsf{enc}}$, generate an independent signing key pair

$(\mathsf{pk}_{\mathsf{sig}}, \mathsf{sk}_{\mathsf{sig}})$, submit the challenge messages $(Y^{\,h-h'}\mathsf{sl}(i), 0)$, receive the challenge ciphertext w, compute $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}_{\mathsf{sig}}, w)$ and forward (w, σ) to $\mathcal{D}$. Since $\mathsf{sk}_{\mathsf{sig}}$ is independent of the challenge, the simulated pair is distributed exactly as in the real game and $\mathcal{B}$ preserves the advantage ϵ. It remains to note that BGV is IND-CPA secure under Definition 11. The public key $(-(as+te), a)$ is a Ring-LWE sample up to relabelling: since $\gcd(t, Q) = 1$, the bijection $(a, b) \mapsto (t^{-1}a,\ t^{-1}b)$ of R_Q^2 carries $(a, as+te)$ to $(a', a's+e)$ with a' uniform, and negation preserves uniformity. Given a uniform (p_0, p_1), the ciphertext $(p_0u + te_0 + m,\ p_1u + te_1)$ consists of two Ring-LWE samples with secret u, shifted by the known constant $(m, 0)$, hence is indistinguishable from uniform. Combining these observations yields $\epsilon \leq \mathsf{negl}(\lambda)$ since the $\mathrm{RLWE}_{R,Q,\chi_s,\chi_e}$ assumption holds.

Theorem 16 (Ephemeral traceability, Ring-LWE). *For any id_i with $i \in [n]$ and any per-tag budget $h' \leq h$, if $(w, \sigma) \leftarrow \mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}_i, 1, h')$ and $w^{(k)}$ is obtained by applying $\mathsf{Degrade}$ at least h' times, then the plaintext of $w^{(k)}$ is identically $0 \in P$, and $\mathsf{Trace}(\mathsf{sk}_{\mathsf{trace}}, w^{(k)}) = \perp$ with probability $1 - \mathsf{negl}(\lambda)$ under the setup correctness bound. The plaintext-erasure claim itself is unconditional and independent of N_0, Q, and t.*

Proof. The tag is issued at depth $h - h'$, so after $k \geq h'$ degradations it encrypts $Y^{\,h-h'+k}\mathsf{sl}(i)$ with $h - h' + k \geq h$, and $Y^h = 0$ in R. Zero is a fixed point of $m \mapsto Ym$, so all further degradations leave the plaintext at 0. Erasure is therefore information-theoretic. Under the setup noise bound, decryption returns 0 with overwhelming probability and Trace returns $\perp$.

Theorem 17 (Unforgeability, Ring-LWE). *Under EUF-CMA security of the signature scheme, the Ring-LWE instantiation satisfies unforgeability for histories within the correctness horizon: every tag with such a valid history traces only to identifiers issued by the authority. A deployment inherits unforgeability when its verification mechanism accepts only tags possessing such a history.*

Proof. A reduction against EUF-CMA simulates the Tag oracle with the signing oracle and its own encryption keys. Thus, for every $(pp, w^*, H) \in R_{\mathsf{hist}}$, each issued tag in H was, except with negligible probability, produced by the oracle and encrypts $Y^{\,h-h'}\mathsf{sl}(i)$ with $\mathsf{id}_i \in \mathcal{Q}$, or zero. Induction along H shows that w^* encrypts a sum $\sum_j Y^{\tau_j}\mathsf{sl}(i_j)$ with every $\mathsf{id}_{i_j} \in \mathcal{Q}$: degradation multiplies by Y, a payment copies the degraded sender contribution into at most two outputs, ingestion merges one issued contribution at its issuance depth, and no tag is consumed twice. Since each identifier is issued with $b = 1$ at most once, each slot has multiplicity at most $2^{h-1} < t$ at every live depth, so no slot wraps. Within the correctness horizon, decryption returns this plaintext except with negligible probability, so tracing returns only identifiers in $\mathcal{Q}$.

Theorem 18 (Unlinkability, Ring-LWE). *Under $\mathrm{RLWE}_{R,Q,\chi_s,\chi_e}$ (Definition 11), the Ring-LWE instantiation satisfies unlinkability. In a ledger deployment, zero knowledge of the host transaction proof additionally hides the consumed account states.*

Proof. For unlinkability, the adversary submits (w_0, w_1) and receives $w' \leftarrow \mathsf{Degrade}(w_b)$ for a uniform bit b.

Tag step. Degradation computes $w' = Y \cdot w_b + \mathsf{Enc}(\mathsf{pk}_{\mathsf{enc}}, 0; \xi)$, where ξ is sampled freshly and independently of everything the adversary has seen, in particular of the internal randomness of w_0 and w_1, which it obtained from its oracle. By the argument of Theorem 15, a fresh $\mathsf{Enc}(\mathsf{pk}_{\mathsf{enc}}, 0; \xi)$ is computationally indistinguishable from a uniform element U of R_Q^2 under $\mathrm{RLWE}_{R,Q,\chi_s,\chi_e}$. Replacing it accordingly gives $w' = Y\, w_b + U$, which is a uniform element of R_Q^2 regardless of the value of b, so in this hybrid the adversary's view is identically distributed for $b = 0$ and $b = 1$ and its guess is correct with probability exactly $\frac{1}{2}$.

Any adversary guessing b from (w_0, w_1, w') with non-negligible advantage therefore yields a distinguisher against $\mathrm{RLWE}_{R,Q,\chi_s,\chi_e}$. Hence $\Pr[b' = b] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$.

5.6 Hardness of Ring-LWE over the extended ring

The ring R has nilpotent elements and is therefore not the ring of integers of a number field, so the RLWE assumption over R might appear to require a new assumption, much as Definition 9 does in Section 5.3. We show that this is not the case: Ring-LWE over R is *equivalent* to standard Ring-LWE over R_0. Since R_0 is cyclotomic, the assumption over R_0 is the most standard form used in the literature, and in particular admits reductions to worst-case problems over ideal lattices. This both supports the starting assumption, and hence the security of this instantiation, and lets us use the standard Ring-LWE literature to estimate parameters.

In a broad sense our assumption can also be viewed as a variant of module LWE, the difference being that R is a particular module with specific structure over R_0, whereas module LWE considers a free module sampled uniformly over a base ring, and is much closer to standard LWE in that sense. The analogue of the evaluation map of Bootland, Castryck, and Vercauteren discussed above is, in our case, the projection $\pi : Y \mapsto 0$, and it forms the basis of the reduction below.

In the following we assume that the distributions χ_s and χ_e are *blockwise independent*, that is, independent across the h blocks of R, with the marginal distribution of each block equal to the corresponding distribution over R_0. This is a natural assumption, satisfied both by the standard discrete Gaussian over R, which is a product of discrete Gaussians over the blocks, and by the uniform ternary distribution.

We start with an observation about the structure of R.

Lemma 4 (Structure). *Recall that $R = R_0[Y]/(Y^h)$, and that as an R_0-module, R is of rank h with basis $1, Y, \ldots, Y^{h-1}$. In this basis, multiplication*

by any $u \in R$ is captured by the matrix

$$M(u) \;=\; \begin{pmatrix} u_0 & & & \\ u_1 & u_0 & & \\ \vdots & \ddots & \ddots & \\ u_{h-1} & \cdots & u_1 & u_0 \end{pmatrix} \in R_0^{h\times h},$$

which is a lower triangular Toeplitz matrix over R_0 with every diagonal entry equal to u_0. In particular, for $u, v \in R$, the product $w = uv$ is represented blockwise as $w_k = \sum_{i+j=k} u_i v_j$ for $k = 0, \ldots, h-1$, where $u_i, v_i, w_i \in R_0$ are the blocks of u, v, w respectively.

Lemma 5 (Projection). *The map $\pi : R \to R_0$ sending $\sum_j u_j Y^j$ to u_0 is a surjective ring homomorphism with kernel (Y). It maps the uniform distribution over R_Q to the uniform distribution over R_0/QR_0, and it carries an RLWE sample over R with secret s to one over R_0 with secret s_0 while preserving the error distribution:*

$$\pi(a \cdot s + e) \;=\; a_0 s_0 + e_0,$$

where a_0 is uniform and e_0 has the blockwise error distribution induced by χ_e.

Proof. The map π is reduction modulo the ideal (Y). It is hence a ring homomorphism, and is surjective with kernel (Y). Under the identification $R_Q \cong (R_0/QR_0)^h$ it is the projection onto the first coordinate, so it preserves uniformity. Applying it to an RLWE sample and using $\pi(as) = \pi(a)\pi(s)$ gives the displayed identity. The marginals for a_0, s_0 and e_0 are the per-block distributions and are independent, since the original distributions are blockwise independent.

Lemma 6 (Block peeling). *Given oracle access to a solver for search Ring-LWE over R_0 in dimension N_0 with modulus Q, the full secret $s \in R_Q$ of any Ring-LWE instance over R is recoverable with h oracle calls and $O(mh^2)$ multiplications in R_0/QR_0, where m is the number of samples.*

Proof. By Lemma 4, the kth block of an RLWE sample in R is $b_k = \sum_{i+j=k} a_i s_j + e_k$. For $k = 0$ this is $a_0 s_0 + e_0$, and one oracle call returns s_0 by Lemma 5. Inductively, once $s_0, \ldots, s_{k-1}$ are known, the quantity

$$b_k - \sum_{i=1}^{k} a_i s_{k-i} \;=\; a_0 s_k + e_k$$

is computable from public data and is again a standard RLWE sample in R_0/QR_0, with the same uniform a_0 and a fresh unknown s_k; one further call returns s_k. After h steps the whole secret is known, and the subtractions cost $\sum_{k<h} mk = O(mh^2)$ multiplications.

We now state the main hardness equivalence.

Theorem 19 (Hardness of Ring-LWE over R). *Let χ_s, χ_e be the standard secret and error distributions for the RLWE assumption over R_0, and let Φ_s, Φ_e be their h-blockwise repetitions, that is, blockwise independent with blockwise marginals χ_s and χ_e respectively. Then $\mathrm{RLWE}_{R,Q,\Phi_s,\Phi_e}$ over $R = R_0[Y]/(Y^h)$ with modulus Q is hard if and only if $\mathrm{RLWE}_{R_0,Q,\chi_s,\chi_e}$ over R_0 at dimension N_0 with modulus Q is hard. Specifically,*
(i) *any distinguisher against $\mathrm{RLWE}_{R_0,Q,\chi_s,\chi_e}$ yields one against $\mathrm{RLWE}_{R,Q,\Phi_s,\Phi_e}$ with the same advantage; and*
(ii) *any distinguisher against $\mathrm{RLWE}_{R,Q,\Phi_s,\Phi_e}$ with advantage ϵ yields one against $\mathrm{RLWE}_{R_0,Q,\chi_s,\chi_e}$ with advantage ϵ/h.*

Proof. (i) Given samples from the $\mathrm{RLWE}_{R,Q,\Phi_s,\Phi_e}$ challenge over R, apply π componentwise to each and invoke the distinguisher for R_0. By Lemma 5, RLWE samples map to samples from $\mathrm{RLWE}_{R_0,Q,\chi_s,\chi_e}$ and uniform samples map to uniform ones, so the reduction preserves advantage.

(ii) We use a hybrid argument. For $\ell = 0, 1, \ldots, h$ let the ℓth hybrid H_ℓ be the distribution on m pairs obtained as follows:

1. sample $s_0, \ldots, s_{\ell-1} \leftarrow \chi_s$ honestly;
2. then, independently for each sample, draw $a \xleftarrow{\$} R_Q$;
3. compute the blocks of b with indices $k < \ell$ as $\sum_{p+j=k} a_p s_j + e_k$ with $e_k \leftarrow \chi_e$;
4. draw the blocks with indices $k \geq \ell$ uniformly from R_0/QR_0.

By Lemma 4, blocks with indices $k < \ell$ involve only $s_0, \ldots, s_k$, so H_ℓ is well defined. One verifies that H_h is the distribution $\mathrm{RLWE}_{R,Q,\Phi_s,\Phi_e}$ and that H_0 is uniform.

Assume for contradiction a distinguisher $\mathcal{D}$ that takes m samples drawn either from $\mathrm{RLWE}_{R,Q,\Phi_s,\Phi_e}$ or uniformly over R_Q, and distinguishes them with advantage ϵ. By convention $\mathcal{D}$ outputs 1 for RLWE samples and 0 for uniform ones. We build a reduction $\mathcal{R}^{\mathcal{D}}$ distinguishing samples from $\mathrm{RLWE}_{R_0,Q,\chi_s,\chi_e}$ from uniform ones. Consider adjacent hybrids $H_{\ell+1}$ and H_ℓ. The reduction $\mathcal{R}^{\mathcal{D}}$ works as follows:

1. Take m challenge pairs (α, u) over R_0, drawn either from $\mathrm{RLWE}_{R_0,Q,\chi_s,\chi_e}$ with secret $s^* \leftarrow \chi_s$ or uniformly.
2. Draw $s_0, \ldots, s_{\ell-1} \leftarrow \chi_s$.
3. For each of the m samples set $a_0 := \alpha$ and $a_1, \ldots, a_{h-1} \xleftarrow{\$} R_0/QR_0$.
4. Fill the upper blocks $k > \ell$ uniformly.
5. Set
$$b_\ell \;:=\; u + \sum_{p=1}^{\ell} a_p \, s_{\ell-p}.$$
6. Feed the resulting samples to $\mathcal{D}$ and output whatever it outputs.

If $u = \alpha s^* + e$ then, writing $s_\ell := s^*$, the right-hand side equals $a_0 s_\ell + \sum_{p \geq 1} a_p s_{\ell-p} + e = \sum_{p+j=\ell} a_p s_j + e$, which is precisely the ℓth block from $\mathrm{RLWE}_{R,Q,\Phi_s,\Phi_e}$, so the samples given to $\mathcal{D}$ are distributed exactly as $H_{\ell+1}$.

If instead u is uniform, then b_ℓ is uniform and independent across samples, and the inputs to $\mathcal{D}$ are distributed as H_ℓ.

The reduction never uses the value $s_\ell = s^*$, only samples from $\mathrm{RLWE}_{R_0,Q,\chi_s,\chi_e}$, so it is well defined. Denote by p_ℓ the probability that $\mathcal{D}$ outputs 1 on inputs from H_ℓ. Replacing $\mathcal{D}$ by its complement if necessary, assume $p_h - p_0 = \epsilon > 0$. Choosing ℓ uniformly from $\{0, \ldots, h-1\}$ and running $\mathcal{R}^{\mathcal{D}}$ on the corresponding samples gives distinguishing advantage

$$\frac{1}{h}\sum_{\ell=0}^{h-1}\bigl(p_{\ell+1} - p_\ell\bigr) \;=\; \frac{p_h - p_0}{h} \;=\; \frac{\epsilon}{h}$$

for $\mathrm{RLWE}_{R_0,Q,\chi_s,\chi_e}$, which proves the assertion.

Remark 3. Hardness of RLWE over the ring R thus reduces to hardness of RLWE over R_0, a much more standard formulation, and we use the latter as a proxy for parameter estimation in the next section. The factor h loss in advantage between the two problems is immaterial in practice, since h is small and at most a tiny polynomial in the security parameter, so the loss can be absorbed without undue adjustment to hardness.

5.7 Parameter selection for the Ring-LWE instantiation

Constraints. Several conditions must hold simultaneously, as detailed in the scheme description above. We summarize them here, writing d for the maximal fan-in at each transition and L for the deployment's ciphertext-correctness horizon. The tracing signal expires after at most h degradations, but a ledger may continue to update and merge the resulting encryption of zero; thus L may exceed h.

1. **Identifier slots.** $n \le N_0$: each identifier occupies one slot of (1).
2. **Slot decomposition.** t must be prime with $t \equiv 1 \pmod{2N_0}$ so that the CRT-based slot decomposition applies, and $t > 2^{h-1}$ to prevent slot wraparound under merging (Theorem 14).
3. **Ciphertext modulus.** $\gcd(t, Q) = 1$ is required by the encryption scheme. We additionally choose $Q \equiv 1 \pmod{2N_0}$ to support efficient negacyclic transforms in the ciphertext ring.
4. **Rerandomization.** Unlinkability (Theorem 18) requires that the mask $\mathsf{Enc}(\mathsf{pk}_{\mathsf{enc}}, 0)$ added by $\mathsf{Degrade}$ be a fresh encryption of zero. Writing B_{enc} for the standard encryption noise, we take the rerandomization noise term to be $F = B_{\mathsf{enc}}$ at every hop.
5. **Correctness.** Decryption is correct while the invariant noise satisfies $\|e\|_\infty < Q/2t - \frac{1}{2}$. Degradation is free by Lemma 3, so the budget is spent only on merging and on the rerandomization step. The recurrence is $B_{j+1} = dB_j + F$ with $B_0 = F = B_{\mathsf{enc}}$, so that over a lifetime of L hops

$$B_L = B_{\mathsf{enc}}\Bigl(d^L + \frac{d^L - 1}{d - 1}\Bigr) \quad (= (L+1)B_{\mathsf{enc}} \text{ when } d = 1\,).$$

The correctness condition then gives $\log_2 Q \gtrsim 1 + \log_2 t + \log_2 B_L$.

6. Security. By Theorem 19 the parameters relevant to security are $(N_0, \log_2 Q)$. We take N_0 from the Homomorphic Encryption Security Standard [1], reproduced as Table 1, which tabulates for each dimension the largest modulus still costing 2^{128} classical operations with ternary secrets and $\sigma = 3.2$.

Table 1. Largest modulus admitting 128-bit classical security at each ring dimension, for ternary secrets and $\sigma = 3.2$ [1]. The listed values allow roughly 27 modulus bits per 1024 of dimension.

Ring dimension N_0	1024	2048	4096	8192	16384	32768
Max $\log_2 Q$	27	54	109	218	438	881

Asymptotics. We treat h, L, d and n as inputs to Setup bounded by $\mathrm{poly}(\lambda)$. For an asymptotic correctness statement, the tail cutoff for the discrete Gaussian error distribution is chosen to make failure negligible in λ. For the concrete estimates below we use the customary cutoff $B_e = \lceil 6\sigma \rceil$. A fresh encryption then has estimated noise $B_{\mathsf{enc}} = B_e(1+2\delta_R)$, where δ_R captures coefficient expansion. Using the heuristic estimate $\delta_R = O(\sqrt{N_0 h})$ for the sampled products in this scheme gives $B_{\mathsf{enc}} = O(\sigma\sqrt{N_0 h})$. Condition (2) gives lower bounds on $\log_2 t$ from both h and N_0; the following estimate additionally assumes that a suitable prime t is selected with $\log_2 t = O(h + \log_2 N_0)$. Substituting these choices into condition (5) suggests

$$\log_2 Q \;=\; O\big(h + \log_2 N_0 + L \log_2 d + \log_2(\sigma\sqrt{N_0 h})\big),$$

with $L \log_2 d$ replaced by $\log_2(L+1)$ when $d = 1$. This is a correctness and noise estimate, not an asymptotic security theorem: the standard's table supplies concrete modulus limits at fixed security level rather than a proof that $N_0 = \tilde{\Theta}(\lambda \log Q)$. For any selected (N_0, Q), the ciphertext size is $|w| = 2N_0 h \lceil \log_2 Q \rceil$ bits.

A few elements of this design differ from the previous instantiations of Sections 5.3 and 5.4. First, the hop budget h does not affect the lattice dimension N_0, and enters security only through the mild loss in advantage of the reduction. Second, the number of identifier slots affects parameters only through the condition $n \le N_0$. Condition (6) already requires a large N_0, so supporting up to N_0 slots requires no further parameter growth. In contrast, the constraints $r^n < q$ and $r^n < N$ in the two group instantiations cause the plaintext modulus, and therefore the group size, to grow with the identifier-slot capacity. Third, the merge fan-in d substantially affects the modulus and tag size, unlike in the other instantiations. Decryption requires a noise budget that supports fan-in d at every hop, which increases the required modulus size.

Concrete parameters. Table 2 illustrates the constraints at $h = 16$, $t = 65537$, $\sigma = 3.2$, a security target of 128 bits, and a correctness horizon of $L = 32$

transitions. Thus the tracing signal expires after 16 degradations, while ciphertext correctness is budgeted for 32 updates, including updates after expiry. We consider three merge profiles: a linear payment chain ($d = 1$), the two-state account model of Section 3 ($d = 2$), and a more aggressive regime with fan-in four ($d = 4$). Every transition rerandomizes with fresh encryption noise, and each row satisfies conditions (1) to (6) under the estimates above. Two observations

Table 2. Illustrative Ring-LWE parameters at $h = 16$, $t = 65537$, $\sigma = 3.2$, and a correctness horizon of $L = 32$ transitions, with fresh rerandomization at every transition. The dimension N_0 is sized against $\log_2 Q$ by Table 1, whose limit is repeated for comparison; the total degree $N = N_0 h$ affects tag size and arithmetic but not the base-ring security estimate.

Profile	N_0	d	$\log_2 Q$	max $\log_2 Q$	N	$\lvert w \rvert$
Linear chain	2048	1	36	54	32 768	0.28 MB
2 state model	4096	2	65	109	65 536	1.02 MB
Merge-heavy	8192	4	97	218	131 072	3.03 MB

emerge. First, tags are larger: at these settings a Ring-LWE tag occupies between 0.28 and 3.0 MB, compared with 0.75 KB for exponential ElGamal and 4.1 KB for Damgård–Jurik at $h = 10$. The Ring-LWE tags are therefore two to three orders of magnitude larger. In contrast, $n_{\max} = N_0$ provides thousands of identifier slots, compared with 28 for the ElGamal parameters of Section 5.3 and 341 for Damgård–Jurik. Supporting up to N_0 slots does not require a larger plaintext modulus because the security constraint already determines N_0. Tracing requires $O(N \log N) \approx 10^6$ operations, independently of the number of live identifiers, and is more efficient than in Section 5.3.

5.8 Efficiency and comparison

Table 3 compares the three instantiations under a common hop budget. Here $n_{\max}$ is the public identifier-slot capacity, K is the number of live identifiers searched during ElGamal tracing, and N_0 is the number of Ring-LWE slots. For ElGamal and Damgård–Jurik, $r > 2^{h-1}$ prevents multiplicity carries. The table uses representative parameters targeting 128-bit security; exact sizes depend on the selected group, RSA modulus, and Ring-LWE parameter set.

All three constructions inherit a linear dependence on the hop budget from the nilpotent template. Indeed, nilpotency of index h gives a strict chain of plaintext subspaces $P \supsetneq \nu P \supsetneq \cdots \supsetneq \nu^h P = \{0\}$, so the plaintext must distinguish h degradation depths. The constructions differ mainly in how much additional space they require for identifier separation and in the cost of recovering those identifiers.

Exponential ElGamal minimizes the tag carried by each ledger state, but its compactness depends on keeping both the identifier-slot capacity and the live traced set modest. Increasing capacity forces $q > r^n$ and, because $q^h \mid p - 1$,

Table 3. The three instantiations at 128-bit security, using the smallest admissible base $r > 2^{h-1}$ where positional encoding is required. The assumptions are Definitions 9, 10, and 11.

	Exp. ElGamal	**Damgård–Jurik**	**Ring-LWE**
Plaintext ring	$\mathbb{Z}_{q^h}$	$\mathbb{Z}_{N^h}$	$(\mathbb{Z}_t[Y]/(Y^h))^{N_0}$
Nilpotent element	prime q	modulus N	indeterminate Y
Degradation	$c \mapsto c^q \cdot \mathsf{Enc}(0)$	$c \mapsto c^N \cdot \mathsf{Enc}(0)$	$c \mapsto Yc + \mathsf{Enc}(0)$
Digit/slot size (bits)	$\max(2\lambda,\, n \log_2 r)$	$\log_2 N$	$\log_2 t$
Tag size (bits)	$2\max(3072,\, h\,\text{digit})$	$(h+1)\log_2 N$	$2hN_0 \log_2 Q$
Tag size at $h = 10$	0.75 KB	4.1 KB	$\approx$ 0.2–2 MB
Capacity $n_{\max}$	28	341	N_0
Trace	$O(h \cdot r^{K/2})$	$O(nh)$	$O(hN_0 \log N_0)$
Assumption	DDH in $\mathbb{G}$	DCR	RLWE over R
Changing h	fresh parameters	same key	fresh parameters
Post-quantum	✗	✗	✓

enlarges the ambient modulus p by roughly h times the growth in q. Tracing additionally performs a meet-in-the-middle search whose cost is exponential in the number K of live identifiers, even when the public capacity is larger. Damgård–Jurik instead decrypts directly to an integer whose base-N digits can be decoded in $O(nh)$ time. Its tags and online exponentiations are more expensive, but capacity is substantially larger, tracing is insensitive to the number of identifiers actually present in a merge, and the same modulus can support different hop budgets under the standard DCR assumption.

This scalability comes with more expensive online arithmetic. Damgård–Jurik operates on $(h+1)\lceil \log_2 N \rceil$-bit values, and the direct rerandomization ρ^{N^h} uses an $h\lceil \log_2 N \rceil$-bit exponent. Precomputing $\zeta = \zeta_0^{N^h}$ and using ζ^ρ with $\rho \xleftarrow{\$} [0, N)$ shortens the online exponent, but restricts the randomness to $\langle \zeta \rangle$ and consequently requires a subgroup variant of DCR. We retain the direct construction so that Theorems 9 and 12 rely on standard DCR as stated.

The Ring-LWE construction removes positional encoding altogether: identifiers occupy independent CRT slots, decoupling the capacity N_0 from the slot modulus and making tracing polynomial and independent of K. It is also the only post-quantum option among the three. These benefits come with much larger tags and with parameters that must budget for noise growth under repeated rerandomization and merging. Consequently, exponential ElGamal is most attractive when bandwidth and ledger-state size dominate and only a small number of identifiers are live; Damgård–Jurik favors larger or heavily merged traced sets under a conventional assumption; and Ring-LWE favors post-quantum security and high capacity when substantially larger tags are acceptable. The measurements in Section 6 therefore evaluate the compact end of this design space.

6 Evaluation

We implemented the procedures introduced in Section 5.3 on a commodity laptop. For $h = 10$ hops and 128-bit security, we obtained primes p and q of 3072 and 257 bits, respectively, and used $r = 513$ for the identifier encoding, which gives 28 identifier slots. We implemented arithmetic in the order-q^h subgroup of $\mathbb{Z}_p^*$ using RELIC. Table 4 reports means over 100 runs for individual operations and three runs for tracing; Section 6.1 gives the experimental setup and hardware. A tag occupies 0.75 KB, so the scheme adds only a few kilobytes to each transaction, an overhead comparable to the size of a Monero transaction [39]. Issuance with its signature takes 38.1 ms and 1.44 KB; degradation with a stand-alone prototype proof takes 54.6 ms and 2.13 KB; merging two tags takes 0.02 ms. The host ledger's own transaction proof adds a cost that depends on its proof system. Tracing all 28 identifier slots takes 1.33 s. Tracing cost depends on the number K of live identifiers and on their per-depth multiplicity M: the benchmark uses $M = 1$, each identifier present once per depth, whereas the bound $O(h\, r^{K/2})$ of Section 5.3 covers the largest value $M = 2^{h-1}$ that Lemma 1 admits.

6.1 Prototype measurements

We evaluate the exponential-ElGamal instantiation over the group $\mathbb{G} \subseteq \mathbb{Z}_p^*$ of order q^h. A tag is $w = (c_1, c_2)$; Tag encrypts $b\, r^{i-1}$ and signs the fresh ciphertext, $\mathsf{Degrade}$ computes $(c_1^q g^\rho, c_2^q \mathsf{pk}_{\mathrm{enc}}^\rho)$, Merge computes $(\prod_j c_{1,j}, \prod_j c_{2,j})$, and Trace decrypts and recovers the nonzero base-q depth digits. The prototype also benchmarks fresh-tag authentication and a stand-alone proof of the tag-transition relation.

Setup and parameters. The C implementation uses RELIC for big-integer arithmetic, modular operations, and SHA-256, and SageMath 10.8 to generate the group parameters. Experiments ran on one core of a 12th-generation Intel® Core™ i7-1260P. We use $h = 10$, $r = 513$, a 257-bit prime $q \geq 2^{2\lambda}$, and a 3072-bit prime p, targeting 128-bit security. These parameters support $n_{\max} = 28$ identifier slots because $r^{28} < q < r^{29}$. We initialize the parameters once, perform one untimed warm-up, and then report the mean and sample standard deviation over 100 runs. Tracing is reported over three timed runs after the warm-up: one trace takes more than a second, so a hundred consecutive traces heat the core enough that the measurement tracks the thermal state of the laptop rather than the cost of the operation. Input generation and reset are outside each timed interval. The tracing row uses all $K = n_{\max} = 28$ identifier slots once, so its exact per-slot multiplicity bound is one.

7 Discussion and applications

ECT attaches encrypted identifier contributions to payment states. Each contribution degrades under a public hop policy, disappears after its assigned budget,

Table 4. Prototype costs, with primitive operations and transaction checks reported individually.

Operation	Output (KB)	Mean (ms)	SD (ms)	Runs
Tag (tag and signature)	1.44	38.1	0.9	100
Verify	—	15.0	1.0	100
Degrade	0.75	29.0	2.4	100
Degrade with prototype proof	2.13	54.6	0.5	100
Prototype-proof verification	—	34.3	0.3	100
Merge, $k = 2$	0.75	0.02	0.00	100
Merge, $k = 10$	0.75	0.23	0.01	100
Trace, $K = 28$	0.75	1333.1	6.0	3

and survives splits and merges until expiry. The tracing authority alone can recover live identifiers, while the ledger proof enforces valid tag transitions. This section discusses disclosure, delegation of degradation, and applications in permissioned payment systems such as central bank digital currencies, regulated stablecoins, and interbank payment networks. These applications combine ECT with institutional measures; the primitive alone neither identifies the current account holder nor authorizes an enforcement action.

On disclosure by the authority. Disclosure necessarily removes uncertainty for the identifiers that the authority names. It does not, however, help distinguish the status of an undisclosed challenge identifier, provided that the tracing key is not revealed. Tags are independently randomized encryptions, so auxiliary tagged or dummy samples are exactly what the tagging oracle of Definition 4 provides; a standard hybrid argument then reduces any remaining advantage to Theorem 3 or 9. The same reasoning permits the disclosed list to contain false entries. This observation concerns the cryptographic view only: side information about an investigation may reveal tracing status independently of the tags.

Recall that our notion of indistinguishability in Definition 4 hinges on a constrained model of interaction between a potentially malicious tracing authority and malicious users. In particular, the authority cannot reveal to the malicious users the full set of users being tracked. Here, we discuss a modification of our indistinguishability notion that accounts for full collusion between the authority and malicious users. Our modification simply enforces that the adversary is given a fixed list of ids $\mathcal{I} := \{\mathsf{id}\}$ but not the tracing secret key $\mathsf{sk}_{\mathsf{trace}}$. Instead, we assume oracle access to the tagging functionality and the adversary can call the tagging oracle $\mathsf{Tag}(\cdot)$ on any of ids in $\mathcal{I}$. The challenge tag $(w, \sigma) \leftarrow \mathsf{Tag}(\mathsf{sk}_{\mathsf{trace}}, \mathsf{id}', b, h)$ is given a fresh id $\mathsf{id}' \notin \mathcal{I}$, and the adversary has to then distinguish if this

Delegating degradation. Degradation uses no secret: it applies the public degradation map and rerandomizes, and both operations are available to any party holding *pp*. Which party advances a tag is therefore a deployment choice, subject to two conditions. That party must be able to update the structure binding tags

to account states, and the ledger must enforce exactly the degradation schedule prescribed by its hop policy. Three deployments illustrate this flexibility. *Coin holders* degrade their own tags inside the transactions they submit; in this deployment, each outgoing transfer is one hop. *Custodial intermediaries*, such as payment service providers or exchanges holding accounts on behalf of their customers, can degrade instead. They already assemble the transactions of those accounts, so the construction is unchanged, and clients with limited resources are relieved of producing the proof. Custody grants no tracing information: the intermediary manipulates ciphertexts under the authority's key and learns neither whether a tag is real nor how much of its budget remains. *Validators* can instead degrade every published tag at a fixed block height or once per epoch. If tags and their updates are public, all parties can recompute the degradation map, so no zero-knowledge proof is needed. This changes the budget from transaction hops to ledger epochs and ages dormant accounts as well. Waiting then consumes the budget, whereas holder-driven degradation preserves a dormant tag but requires transactions to exhaust it. Ledger-driven degradation requires a publicly updatable tag structure rather than owner-controlled account states; it is therefore a different ledger model, not merely a parameter choice.

Beyond coin tracing. ECT records that a ledger state descends from a tagged identifier; turning that signal into action requires the ledger or operator to associate the state with an account and apply an external policy. Possible responses include freezing the current account, imposing transaction limits, restricting counterparties, or requesting additional records under the applicable legal process. ECT can also support the investigation of structuring by preserving the tagged contribution across splits and merges, but it does not itself classify a transaction pattern as suspicious. These measures are complementary and may be applied only while the tag is traceable, or later on the basis of evidence recorded during that window.

Prospective versus retrospective tracing. ECT is *prospective* by design: Trace can recover an identifier only if the authority previously encoded it with $b = 1$. An untagged contribution may later share a state with tagged funds, but its own origin cannot be recovered retroactively, even with $\mathsf{sk}_{\mathsf{trace}}$. This separation limits retrospective mass surveillance. Deployments requiring retrospective tracing need an additional mechanism, for example a distinct account type whose transactions carry an opening trapdoor. Distributing that trapdoor among a threshold committee can prevent unilateral tracing, as in PEReDi [37] and UTT [55]; such a mechanism is complementary to, rather than an extension of, ECT.

Issuance, renewal, and hop policy. ECT makes the reach of every issued contribution explicit, and a deployment can layer its accountability guarantees on top of this bound. Policy enters at issuance: the authority's signature authenticates each board entry, and a deployment that wants to enforce a warrant's shorter budget $h' < h$, or a quota on the number of tagged accounts, against the authority itself can add a zero-knowledge proof of well-formed issuance; the public

slot capacity bounds how many identifiers a tag distinguishes, not how many accounts receive tags. Renewal issues a fresh identifier (Section 3.2), so every identifier has at most one live issuance and the no-carry guarantee of Lemma 1 holds across repeated warrants. Under the default payment policy a hop is an outgoing transfer, so dormant funds keep their contribution until they move; deployments that also want to bound elapsed time can adopt the epoch rule, under which every ingestion advances existing contributions.

User-requested protective tracing. A user facing kidnapping, extortion, or account-compromise risk may voluntarily request traceable tags for their holdings. For example, a merchant could request $b = 1$ with budget $h' = 5$ and either announce the protection as a deterrent or keep it hidden using indistinguishability. If the funds are transferred under duress, the authority can inspect subsequent ledger tags and use Trace to recognize descendants of the merchant's tagged state. Where the ledger links those states to enforceable accounts, an authorized party may then freeze or investigate them. After h' transaction hops, ECT no longer supplies that tracing signal.

Time-bounded regulatory investigation. A court order may authorize prospective tracing of a suspicious account for at most h' transaction hops. The authority issues tags with $h' \leq h$, after which degradation removes the encoded identifier without requiring the authority to revoke it. This enforces a transaction-distance bound, not a wall-clock deadline; a time-based warrant instead requires the epoch-driven variant above or an additional ledger rule.

Interbank tracing. In an interbank payment network, a financial intelligence unit or similar authority may tag funds after a participating bank reports a suspicious account. Each bank carries the encrypted tag in its authenticated payment state and enforces the agreed transition policy when funds move within or across banks. The authority can then recognize tagged funds for the prescribed number of hops, while the receiving bank can use its customer records to identify the current account holder and take any authorized action. Banks learn no tracing information from the tag itself, and unrelated customers' transaction histories remain hidden. This deployment requires an interoperable payment rail that binds tags to states, but does not require banks to pool their transaction records.

Privacy budget integration. ECT can be combined with a ledger-enforced privacy budget, such as a zero-knowledge proof that a user's monthly volume remains below a threshold. Crossing the threshold would trigger a separate policy that obtains an authority-issued traceable tag for the resulting state; implementing this trigger without revealing compliant users requires additional protocol machinery. The budget determines *when* monitoring begins, while ECT bounds how far the resulting signal propagates. Because tags survive splits and merges, the combination can support investigations of structuring, although detection of the pattern remains outside the ECT primitive.

Temporary institutional transparency. Exchanges and custodians may ask the authority to issue traceable tags for designated deposits or withdrawals. The authority's signature authenticates each initial tag, while indistinguishability hides whether it encodes an identifier. Auditors can then follow the propagation of tagged funds for a bounded number of hops, provided they receive the relevant tracing result and ledger view. This offers bounded transaction-flow accountability; it does not by itself prove solvency or correct custody, which require separate accounting commitments and proofs.

8 Concluding remarks and outlook

We answer our opening question affirmatively: a simple and efficient protocol can support targeted tracing while cryptographically bounding its reach, even against the tracing authority itself. ECT realizes this goal through encrypted identifier contributions that follow funds through splits and merges and advance toward expiry under public degradation. After its assigned number of policy-defined hops, a contribution becomes zero and cannot be recovered even by the tracing authority, without affecting other live contributions. The authority can therefore initiate a trace but cannot extend it indefinitely.

Our nilpotent-ring template makes expiry exact and expired contributions neutral under merging. Its three instantiations offer different tradeoffs: exponential ElGamal gives the smallest tags and a deliberately small identifier capacity; Damgård–Jurik provides greater capacity and polynomial-time tracing; and Ring-LWE offers post-quantum security and thousands of slots at the cost of larger tags. At 128-bit security with a ten-hop budget, our ElGamal implementation adds a few kilobytes per transaction and traces 28 identifiers in 1.33 s. In every construction, public parameters expose the hop and capacity bounds.

ECT can support warrant-based tracing in CBDCs and regulated stablecoins, cross-institution tracing without pooling banks' records, and applications such as protective tracing or temporary institutional transparency. The surrounding system must still authorize issuance, define what constitutes a hop, enforce tag transitions, and determine how to act on tracing results. Since ECT cannot recover transactions that preceded tag issuance, it complements retrospective investigative tools. Sections 5.8 and 7 detail the construction tradeoffs and deployment choices.

Several directions remain open. A zero-knowledge proof of policy-compliant issuance would let deployments enforce warrant budgets and tagging quotas against the authority itself without revealing who is tagged. Implementations of the Damgård–Jurik and Ring-LWE constructions, integrated with a complete ledger proof system, would quantify their end-to-end costs. For Ring-LWE, a sharper analysis of what the residual noise of an expired tag reveals would carry the exact expiry of the group-based constructions over to the post-quantum setting.

References

1. Albrecht, M.R., Chase, M., Chen, H., Ding, J., Goldwasser, S., Gorbunov, S., Halevi, S., Hoffstein, J., Laine, K., Lauter, K.E., Lokam, S., Micciancio, D., Moody, D., Morrison, T., Sahai, A., Vaikuntanathan, V.: Homomorphic Encryption Standard. IACR Cryptol. ePrint Arch. **2019**, 939 (2019), https://eprint.iacr.org/2019/939
2. Androulaki, E., Camenisch, J., Caro, A.D., Dubovitskaya, M., Elkhiyaoui, K., Tackmann, B.: Privacy-preserving auditable token payments in a permissioned blockchain system. In: AFT. pp. 255–267 (2020). https://doi.org/10.1145/3419614.3423259
3. Beer, C., Zingg, S., Kostiainen, K., Wüst, K., Capkun, V., Capkun, S.: PayOff: A Regulated Central Bank Digital Currency with Private Offline Payments. In: AsiaCCS. pp. 693–714 (2026). https://doi.org/10.1145/3779208.3805963
4. Ben-Sasson, E., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., Virza, M.: Zerocash: Decentralized Anonymous Payments from Bitcoin. In: IEEE Symposium on Security and Privacy. pp. 459–474 (2014). https://doi.org/10.1109/SP.2014.36
5. Boneh, D.: The Decision Diffie-Hellman Problem. In: ANTS. pp. 48–63 (1998). https://doi.org/10.1007/BFb0054851
6. Boneh, D., Partap, A., Rotem, L.: Accountability for Misbehavior in Threshold Decryption via Threshold Traitor Tracing. In: CRYPTO (7). pp. 317–351 (2024). https://doi.org/10.1007/978-3-031-68394-7_11
7. Boneh, D., Sahai, A., Waters, B.: Fully Collusion Resistant Traitor Tracing with Short Ciphertexts and Private Keys. In: EUROCRYPT. pp. 573–592 (2006). https://doi.org/10.1007/11761679_34
8. Bootland, C., Castryck, W., Vercauteren, F.: On the security of the multivariate ring learning with errors problem. In: ANTS XIV: Proceedings of the Fourteenth Algorithmic Number Theory Symposium. The Open Book Series, vol. 4, pp. 57–71. Mathematical Sciences Publishers (2020). https://doi.org/10.2140/obs.2020.4.57
9. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) fully homomorphic encryption without bootstrapping. In: ITCS. pp. 309–325 (2012). https://doi.org/10.1145/2090236.2090262
10. Brickell, E.F., Gemmell, P., Kravitz, D.W.: Trustee-based Tracing Extensions to Anonymous Cash and the Making of Anonymous Change. In: SODA. pp. 457–466 (1995), http://dl.acm.org/citation.cfm?id=313651.313790
11. Buterin, V., Illum, J., Nadler, M., Schär, F., Soleimani, A.: Blockchain privacy and regulatory compliance: Towards a practical equilibrium. Blockchain Res. Appl. **5**(1), 100176 (2024). https://doi.org/10.1016/j.bcra.2023.100176
12. Cachin, C.: An Information-Theoretic Model for Steganography. In: Information Hiding. pp. 306–318 (1998). https://doi.org/10.1007/3-540-49380-8_21
13. Camenisch, J., Hohenberger, S., Lysyanskaya, A.: Balancing Accountability and Privacy Using E-Cash (Extended Abstract). In: SCN. pp. 141–155 (2006). https://doi.org/10.1007/11832072_10
14. Camenisch, J., Maurer, U.M., Stadler, M.: Digital Payment Systems with Passive Anonymity-Revoking Trustees. In: ESORICS. pp. 33–43 (1996). https://doi.org/10.1007/3-540-61770-1_26
15. Cecchetti, E., Zhang, F., Ji, Y., Kosba, A.E., Juels, A., Shi, E.: Solidus: Confidential Distributed Ledger Transactions via PVORM. In: CCS. pp. 701–717 (2017). https://doi.org/10.1145/3133956.3134010

16. Chatzigiannis, P., Baldimtsi, F.: MiniLedger: Compact-Sized Anonymous and Auditable Distributed Payments. In: ESORICS (1). pp. 407–429 (2021). https://doi.org/10.1007/978-3-030-88418-5_20
17. Chatzigiannis, P., Baldimtsi, F., Chalkias, K.: SoK: Auditability and Accountability in Distributed Payment Systems. In: ACNS (2). pp. 311–337 (2021). https://doi.org/10.1007/978-3-030-78375-4_13
18. Chaum, D.: Blind Signatures for Untraceable Payments. In: CRYPTO. pp. 199–203 (1982). https://doi.org/10.1007/978-1-4757-0602-4_18
19. Chaum, D., Crépeau, C., Damgård, I.: Multiparty Unconditionally Secure Protocols (Extended Abstract). In: STOC. pp. 11–19 (1988). https://doi.org/10.1145/62212.62214
20. yu Chiang, J.H., David, B., Frederiksen, T.K., Mondal, A., Yeniaras, E.: Detecting Rogue Decryption in (Threshold) Encryption via Self-Incriminating Proofs. IACR Cryptol. ePrint Arch. **2024**, 794 (2024), https://eprint.iacr.org/2024/794
21. Chor, B., Fiat, A., Naor, M., Pinkas, B.: Tracing traitors. IEEE Trans. Inf. Theory **46**(3), 893–910 (2000). https://doi.org/10.1109/18.841169
22. Christ, M., Gunn, S., Zamir, O.: Undetectable Watermarks for Language Models. In: COLT. pp. 1125–1139 (2024), https://proceedings.mlr.press/v247/christ24a.html
23. Cohen, A., Hoover, A., Schoenbach, G.: Watermarking Language Models for Many Adaptive Users. In: SP. pp. 2583–2601 (2025). https://doi.org/10.1109/SP61157.2025.00084
24. Damgård, I., Jurik, M.: A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System. In: Public Key Cryptography. pp. 119–136 (2001). https://doi.org/10.1007/3-540-44586-2_9
25. Davida, G.I., Frankel, Y., Tsiounis, Y., Yung, M.: Anonymity Control in E-Cash Systems. In: Financial Cryptography. pp. 1–16 (1997). https://doi.org/10.1007/3-540-63594-7_63
26. Effendi, F., Chattopadhyay, A.: Privacy-Preserving Graph-Based Machine Learning with Fully Homomorphic Encryption for Collaborative Anti-money Laundering. In: SPACE. pp. 80–105 (2024). https://doi.org/10.1007/978-3-031-80408-3_6
27. van Egmond, M.B., Dunning, V., van den Berg, S., Rooijakkers, T., Sangers, A., Poppe, T., Veldsink, J.: Privacy-Preserving Anti-money Laundering Using Secure Multi-party Computation. In: FC (2). pp. 331–349 (2024). https://doi.org/10.1007/978-3-031-78679-2_18
28. Fauzi, P., Meiklejohn, S., Mercer, R., Orlandi, C.: Quisquis: A New Design for Anonymous Cryptocurrencies. In: ASIACRYPT (1). pp. 649–678 (2019). https://doi.org/10.1007/978-3-030-34578-5_23
29. Frankel, Y., Tsiounis, Y., Yung, M.: "Indirect Discourse Proof": Achieving Efficient Fair Off-Line E-cash. In: ASIACRYPT. pp. 286–300 (1996). https://doi.org/10.1007/BFb0034855
30. Fujisaki, E.: Sub-linear Size Traceable Ring Signatures without Random Oracles. In: CT-RSA. pp. 393–415 (2011). https://doi.org/10.1007/978-3-642-19074-2_25
31. Gamal, T.E.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. Inf. Theory **31**(4), 469–472 (1985). https://doi.org/10.1109/TIT.1985.1057074
32. Garman, C., Green, M., Miers, I.: Accountable Privacy for Decentralized Anonymous Payments. In: Financial Cryptography. pp. 81–98 (2016). https://doi.org/10.1007/978-3-662-54970-4_5

33. Gennaro, R., Krawczyk, H., Rabin, T.: Secure Hashed Diffie-Hellman over Non-DDH Groups. In: EUROCRYPT. pp. 361–381 (2004). https://doi.org/10.1007/978-3-540-24676-3_22
34. Ghadafi, E.: Efficient Distributed Tag-Based Encryption and Its Application to Group Signatures with Efficient Distributed Traceability. In: LATINCRYPT. pp. 327–347 (2014). https://doi.org/10.1007/978-3-319-16295-9_18
35. He, Y., Chen, J.: AMLChain: Supporting Anti-money Laundering, Privacy-Preserving, Auditable Distributed Ledger. In: EISA. pp. 50–67 (2021). https://doi.org/10.1007/978-3-030-93956-4_4
36. Jakobsson, M., Yung, M.: Revokable and Versatile Electronic Money (extended abstract). In: CCS. pp. 76–87 (1996). https://doi.org/10.1145/238168.238191
37. Kiayias, A., Kohlweiss, M., Sarencheh, A.: PEReDi: Privacy-Enhanced, Regulated and Distributed Central Bank Digital Currencies. In: CCS. pp. 1739–1752 (2022). https://doi.org/10.1145/3548606.3560707
38. Kiayias, A., Tsiounis, Y., Yung, M.: Traceable Signatures. In: EUROCRYPT. pp. 571–589 (2004). https://doi.org/10.1007/978-3-540-24676-3_34
39. koe, Kurt M. Alonso, Sarang Noether: Zero to Monero: Second Edition: A Technical Guide to a Private Digital Currency; for Beginners, Amateurs, and Experts. Monero Research Lab, 2.0.0 edn. (Apr 2020), https://web.getmonero.org/library/Zero-to-Monero-2-0-0.pdf, released into the public domain
40. Li, R., Li, Y., Wang, Q., Duan, S., Wang, Q., Ryan, M.: Accountable Decryption Made Formal and Practical. IEEE Trans. Inf. Forensics Secur. **20**, 620–635 (2025). https://doi.org/10.1109/TIFS.2024.3515808
41. Li, R., Wang, Q., Liu, F., Wang, Q., Galindo, D.: An Accountable Decryption System Based on Privacy-Preserving Smart Contracts. In: ISC. pp. 372–390 (2020). https://doi.org/10.1007/978-3-030-62974-8_21
42. Narula, N., Vasquez, W., Virza, M.: zkLedger: Privacy-Preserving Auditing for Distributed Ledgers. In: NSDI. pp. 65–80 (2018), https://www.usenix.org/conference/nsdi18/presentation/narula
43. Papadoulis, G., Balla, D., Grontas, P., Pagourtzis, A.: AQQUA: Augmenting Quisquis with Auditability. In: ACNS (2). pp. 283–312 (2025). https://doi.org/10.1007/978-3-031-95764-2_12
44. Pedrouzo-Ulloa, A., Troncoso-Pastoriza, J.R., Gama, N., Georgieva, M., Pérez-González, F.: Revisiting Multivariate Ring Learning with Errors and Its Applications on Lattice-Based Cryptography. Mathematics **9**(8), 858 (2021). https://doi.org/10.3390/math9080858, preliminary version: Cryptology ePrint Archive, Report 2019/1109
45. Pedrouzo-Ulloa, A., Troncoso-Pastoriza, J.R., Pérez-González, F.: Multivariate lattices for encrypted image processing. In: ICASSP. pp. 1707–1711 (2015). https://doi.org/10.1109/ICASSP.2015.7178262
46. Pedrouzo-Ulloa, A., Troncoso-Pastoriza, J.R., Pérez-González, F.: On Ideal Lattices over the Tensor Product of Number Fields and Ring Learning with Errors over Multivariate Rings. CoRR **abs/1607.05244** (2016), http://arxiv.org/abs/1607.05244
47. Pedrouzo-Ulloa, A., Troncoso-Pastoriza, J.R., Pérez-González, F.: Multivariate Cryptosystems for Secure Processing of Multidimensional Signals. CoRR **abs/1712.00848** (2017), http://arxiv.org/abs/1712.00848
48. Pohlig, S.C., Hellman, M.E.: An improved algorithm for computing logarithms over GF(p) and its cryptographic significance (Corresp.). IEEE Trans. Inf. Theory **24**(1), 106–110 (1978). https://doi.org/10.1109/TIT.1978.1055817

49. Ryan, M.D.: Making Decryption Accountable. In: Security Protocols Workshop. pp. 93–98 (2017). https://doi.org/10.1007/978-3-319-71075-4_11
50. Sander, T., Ta-Shma, A.: Auditable, Anonymous Electronic Cash Extended Abstract. In: CRYPTO. pp. 555–572 (1999). https://doi.org/10.1007/3-540-48405-1_35
51. Shoup, V.: Lower Bounds for Discrete Logarithms and Related Problems. In: EUROCRYPT. pp. 256–266 (1997). https://doi.org/10.1007/3-540-69053-0_18
52. Smart, N.P., Vercauteren, F.: Fully homomorphic SIMD operations. Des. Codes Cryptogr. **71**(1), 57–81 (2014). https://doi.org/10.1007/s10623-012-9720-4
53. von Solms, S.H., Naccache, D.: On blind signatures and perfect crimes. Comput. Secur. **11**(6), 581–583 (1992). https://doi.org/10.1016/0167-4048(92)90193-U
54. Stadler, M., Piveteau, J.M., Camenisch, J.: Fair Blind Signatures. In: EUROCRYPT. pp. 209–219 (1995). https://doi.org/10.1007/3-540-49264-X_17
55. Tomescu, A., Bhat, A., Applebaum, B., Abraham, I., Gueta, G., Pinkas, B., Yanai, A.: UTT: Decentralized Ecash with Accountable Privacy. IACR Cryptol. ePrint Arch. **2022**, 452 (2022), https://eprint.iacr.org/2022/452
56. Wicht, F.X., Wang, Z., Le, D.V., Cachin, C.: A Transaction-Level Model for Blockchain Privacy. In: FC (2). pp. 293–310 (2024). https://doi.org/10.1007/978-3-031-78679-2_16
57. Wüst, K., Kostiainen, K., Capkun, V., Capkun, S.: PRCash: Fast, Private and Regulated Transactions for Digital Currencies. In: Financial Cryptography. pp. 158–178 (2019). https://doi.org/10.1007/978-3-030-32101-7_11
58. Wüst, K., Kostiainen, K., Delius, N., Capkun, S.: Platypus: A Central Bank Digital Currency with Unlinkable Transactions and Privacy-Preserving Regulation. In: CCS. pp. 2947–2960 (2022). https://doi.org/10.1145/3548606.3560617
59. Zhandry, M.: Optimal Traitor Tracing from Pairings. In: EUROCRYPT (3). pp. 305–335 (2025). https://doi.org/10.1007/978-3-031-91131-6_11
60. Zhao, X., Gunn, S., Christ, M., Fairoze, J., Fábrega, A., Carlini, N., Garg, S., Hong, S., Nasr, M., Tramèr, F., Jha, S., Li, L., Wang, Y.X., Song, D.: SoK: Watermarking for AI-Generated Content. In: SP. pp. 2621–2639 (2025). https://doi.org/10.1109/SP61157.2025.00178
61. Zhaolu, T., Wan, Z., Wang, H.: Division of Regulatory Power: Collaborative Regulation for Privacy-Preserving Blockchains. IEEE Trans. Inf. Forensics Secur. **19**, 2533–2548 (2024). https://doi.org/10.1109/TIFS.2023.3348268

A Standard cryptographic notions

A.1 Public key encryption

Here, we recall public key encryption and IND-CPA security.

Definition 12 (Public key encryption scheme). *A public key encryption scheme over a message space* $\mathbb{M}$ *with security parameter* 1^λ *is a tuple of 3 PPT algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *such that*

- $\mathsf{KeyGen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$ *generates a public/secret key pair.*
- $\mathsf{Enc}(\mathsf{pk}, m) \to c$ *encrypts message* $m \in \mathbb{M}$ *to ciphertext* c*.*
- $\mathsf{Dec}(\mathsf{sk}, c) \to m$ *decrypts ciphertext* c *to message* m *or* $\perp$*.*

We require correctness, *i.e., for all* $\lambda \in \mathbb{N}$, $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and* $m \in \mathbb{M}$, $\Pr[\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, m)) = m] = 1$.

We further require that our public-key encryption scheme is IND-CPA secure against computationally bounded adversaries. IND-CPA security is defined by the following indistinguishability game between an adversary $\mathcal{A}$ and challenger $\mathcal{C}$.

IND-CPA indistinguishability game $\mathsf{PubK}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(\lambda)$.

1. $\mathcal{C}$ samples $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$.
2. $\mathcal{A}$ gets oracle access to $\mathsf{Enc}(pk, \cdot)$. It outputs a pair of messages (m_0, m_1) of the same length and sends them to $\mathcal{C}$.
3. $\mathcal{C}$ samples a uniform random bit $b \in \{0, 1\}$, and computes the challenge ciphertext $c^* = \mathsf{Enc}(\mathsf{pk}, m_b)$. $\mathcal{C}$ sends the challenge ciphertext c^* to $\mathcal{A}$.
4. $\mathcal{A}$ outputs a guess bit b'. The output of the game is 1 if $b' = b$ and 0 otherwise.

Definition 13 (IND-CPA security). *A public key encryption scheme* $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is IND-CPA secure if for every PPT adversary* $\mathcal{A}$*, there exists a negligible function* negl *such that*

$$\Pr[\mathsf{PubK}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(\lambda) = 1] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

A.2 Signature scheme

Here, we recall signature schemes and EUF-CMA security.

Definition 14 (Signature scheme). *A (digital) signature scheme over a message space* $\mathbb{M}$ *with security parameter* 1^λ *is a tuple of 3 PPT algorithms* $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Ver})$ *such that*

- $\mathsf{KeyGen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$ *generates a public/secret key pair.*
- $\mathsf{Sign}(\mathsf{sk}, m) \to \sigma$ *signs a message* $m \in \mathbb{M}$ *to output a signature* σ*.*
- $\mathsf{Ver}(\mathsf{pk}, m, \sigma) \to b$ *takes as input a public key* pk*, a message* m*, and a signature* σ *and outputs a bit* b *representing if the* σ *is a valid signature on* m *or not.*

We require correctness, *i.e., for all* $\lambda \in \mathbb{N}$, $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and* $m \in \mathbb{M}$, $\Pr[\mathsf{Ver}(\mathsf{pk}, m, \mathsf{Sign}(\mathsf{sk}, m)) = 1] = 1$.

We further require that our signature scheme is EUF-CMA secure against computationally bounded adversaries. EUF-CMA security is defined by the following indistinguishability game between an adversary $\mathcal{A}$ and challenger $\mathcal{C}$.

EUF-CMA indistinguishability game $\mathsf{SigForge}^{\mathsf{EUF-CMA}}_{\mathcal{A},\Sigma}(\lambda)$.

1. $\mathcal{C}$ samples $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$.
2. $\mathcal{A}$ receives pk and gets oracle access to $\mathsf{Sign}(\mathsf{sk}, \cdot)$. $\mathcal{A}$ can repeatedly query the signing oracle $\mathsf{Sign}(\mathsf{sk}, \cdot)$ with messages $(m_1, \ldots, m_Q)$ of its choosing to get signatures $(\sigma_1, \ldots, \sigma_Q)$.
3. $\mathcal{A}$ now outputs a new message and signature pair (m^*, σ^*) such that $m^* \notin (m_1, \ldots, m_Q)$, and sends (m^*, σ^*) to $\mathcal{C}$.
4. The output of the game is 1 if $\mathsf{Ver}(\mathsf{pk}, m^*, \sigma^*) = 1$ and $m^* \notin (m_1, \ldots, m_Q)$, and 0 otherwise.

Definition 15 (EUF-CMA security). *A signature scheme* $\Sigma = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Ver})$ *is EUF-CMA secure if for every PPT adversary* $\mathcal{A}$*, there exists a negligible function* negl *such that*

$$\Pr[\mathsf{SigForge}^{\mathsf{EUF-CMA}}_{\mathcal{A},\Sigma}(\lambda) = 1] \leq \mathsf{negl}(\lambda).$$

A.3 Non-interactive zero-knowledge (NIZK) proof system

We recall NIZK proof systems and their security.

Definition 16 (NIZK proof system). *A pair of PPT algorithms* $(\mathcal{P}, \mathcal{V})$ *with security parameter* λ *is a NIZK proof system for an NP relation* R *if there exists some polynomial* poly *such that*

1. *(Completeness.) For* $x \in R \cap \{0,1\}^\lambda$*, and all witnesses* w *for* x*,*

$$\Pr[r \leftarrow \{0,1\}^{\mathsf{poly}(\lambda)}; \pi \leftarrow \mathcal{P}(r, x, w); \mathcal{V}(r, x, \pi) = 1] \geq 1 - \mathsf{negl}(\lambda).$$

2. *(Soundness.) For all* $x \in \{0,1\}^\lambda \setminus R$*, and for all algorithms* $\mathcal{P}^*$*:*

$$\Pr[r \leftarrow \{0,1\}^{\mathsf{poly}(\lambda)}; \pi \leftarrow \mathcal{P}^*(r, x); \mathcal{V}(r, x, \pi) = 1] \leq \mathsf{negl}(\lambda).$$

3. *(Zero-knowledge.) There exists a PPT algorithm* Sim *such that for any* $x \in R \cap \{0,1\}^\lambda$ *and any witness* w *for* x*, the following ensembles of random variables are computationally indistinguishable:*

$$(1)\ \{r \leftarrow \{0,1\}^{\mathsf{poly}(\lambda)}; \pi \leftarrow \mathcal{P}(r, x, w)\}_\lambda$$

$$(2)\ \{(r, \pi) \leftarrow \mathsf{Sim}(x)\}_\lambda$$