Efficient Privacy-Preserving Blueprints for Threshold Comparison

Pratyush Ranjan Tiwari[1], Harry Eldridge[2], and Matthew Green[2]

[1] Eternis Labs
pratyush@eternis.ai
[2] Johns Hopkins University
{hme, mgreen}@cs.jhu.edu

Abstract. Privacy-Preserving Blueprints (PPBs), introduced by Kohlweiss *et al.* in in EUROCRYPT 2023, offer a method for balancing user privacy and bad-actor detection in private cryptocurrencies. A PPB scheme allows a user to append a verifiable *escrow* to their transactions which reveals some identifying information to an authority in the case that the user misbehaved. A natural PPB functionality is for escrows to reveal user information if the user sends an amount of currency over a certain threshold. However, prior works constructing PPBs for such a functionality have severe limitations when it comes to efficiency: escrows are either computationally infeasible to compute, or too large to be plausibly stored on a large-scale distributed ledger. We address these gaps by constructing a space and computation-efficient PPB for threshold comparison, producing escrows under 2kb that can be computed in seconds. The scheme can be instantiated using well-known cryptographic primitives, namely variants of the ElGamal encryption scheme and generic non-interactive zero-knowledge proofs. As an additional contribution, we implement one of the theoretical generic PPB constructions originally proposed by Kohlweiss *et al.* and find that it performs surprisingly well in practice. For the threshold comparison functionality it requires approximately 14kb escrows, and can be computed in around 12 seconds.

1 Introduction

Cryptocurrencies, protocols and exchanges have recently faced stringent regulatory pressure due to their potential misuse by terrorist organizations and for money laundering purposes in the US [33], EU [30], UAE [8] and Japan [36]. This problem has become more serious due to the recent deployment of asset-backed *stablecoins*, which use decentralized networks to conduct transactions denominated in regulated financial assets. A common solution to this problem employs blockchain tracing [21], which depends on the fact that many cryptocurrency systems use *non-private* transactions, where all details are visible to the public. Unfortunately, this results in a level of transaction privacy dramatically below that of the traditional financial system. Several technologies have sought to improve privacy by deploying techniques that make transaction details confidential

to non-participants [29, 25, 28, 19]. The lack of visibility in these systems has led to a series of conflicts with regulatory authorities, culminating in the U.S. Treasury applying sanctions to the Tornado Cash smart contracts [34].

Over the past several years, several proposals have sought to achieve privacy while also satisfying the requirements of government regulators. One family of solutions aims to preserve the confidentiality of transactions from the general public while also providing some degree of visibility to a designated *compliance authority* (*e.g.,* a stablecoin issuer). The most straightforward approach to this problem makes *all* transaction data visible to the authority, for example by encrypting transaction details under the authority's public key. However, this level of visibility can pose a risk to users in the event that the authority becomes compromised. An alternative approach is related to the area of *pre-constrained encryption* [2]: in these protocols a designated authority learns only appropriate functions of an individual's financial activity, while keeping other aspects fully private. For example, individuals who remain within an envelope of compliance parameters would experience full transaction privacy, while those who exceed these parameters would have some or all of their transaction data revealed. Most critically, this guarantee should hold *even if the authority is fully compromised and behaves maliciously*, which rules out solutions that simply collect all transaction data and then filter what is made available to regulators.

Privacy-preserving blueprints. In service of this goal, Kohlweiss et al. [16] introduced *Privacy-Preserving Blueprints* (PPBs). In a PPB scheme, an auditor publishes a function $f(\cdot,\cdot)$, along with some secret and committed input x chosen by the auditor. Each time a user authors a transaction, it adds an encrypted *escrow* field that contains the value $f(x, v)$, where v represents some input from the user's transaction. Finally, this escrow field can be decrypted by the auditor. This primitive has several applications. For example, $f(x,\cdot)$ can realize a *watchlist* functionality [16, 13]: here $x = \{A_1, \ldots, A_n\}$ contains a secret list of suspicious accounts, and $v = \{A_{\mathsf{sender}}, A_{\mathsf{recipient}}\}$ are the addresses of the parties involved in the transaction. The function $f(x, v)$ would reveal v to the auditor iff $v \cap x \neq \emptyset$ and would reveal an empty value $\perp$ in all other cases. A critical feature of PPBs is that the watchlist x remains confidential from the transaction authors.[3]

PPBs for threshold comparison. One particularly useful functionality for PPBs is *threshold comparison*, which allows authorities to place threshold limits on transaction values. Here v contains the amount of a transaction, and the auditor learns some message m iff $v > x$.[4] This functionality is a critical component of implementing traditional Anti-Money Laundering (AML) functionalities. For

[3] In practice, the auditor's input x can be placed within a public commitment: this ensures that the contents of the watchlist are not entirely arbitrary, *e.g.,* outside parties can be allowed to examine the content of the watchlist and ensure that this watchlist is in use on the network.

[4] In practice m may be identifying information for the user, or a key that encrypts other values of interest.

example, the U.S. Bank Secrecy Act both mandates the reporting of cash transactions over $10,000 [22], and that banks file secret Suspicious Activity Reports (SARs) for any transaction that may represent "structuring" intended to evade the published limits. PPBs for threshold comparison (over individual transaction amounts, or cumulative transaction amounts over a period) represents one way to implement SARs on a private chain.

Building threshold-comparison PPBs for use on large-scale distributed ledgers imposes a number of restrictions on constructions. Any constructions must be efficient in both computation time and particularly in bandwidth (*i.e.,* the size of the escrow field), since the escrow must be stored on the ledger. Next, PPBs require that escrows be *publicly verifiable*. While the auditor checks and decrypts values, the validators maintaining the ledger should be capable of verifying that any given escrow is valid as a condition for accepting transactions on chain. Finally, the auditor must be able to decrypt an escrow field non-interactively, *i.e.,* it should not require further interaction with the transaction author to obtain the escrow data.

Limitations of previous work. Previous works addressing threshold-comparison PPBs or related problems fall short on one or more of these metrics. Kohlweiss *et al.* propose a solution based on fully-homomorphic encryption (FHE) [16], which can realize general functionalities and is space-efficient. However, the scheme requires the transactor to formulate a zero-knowledge proof over the FHE evaluation process, which remains computationally infeasible [35]. Subsequent work [13] improves efficiency, but supports only the more limited set membership ("watchlist") functionality discussed above.

A second line of work by David *et al.* [10] produces a computationally-efficient *updatable* PPB that can be adapted to fit the threshold comparison functionality. In their scheme, users submit transactions with values up to some limit d, and the auditor becomes able to decrypt an escrow when the sum of the transaction values passes the threshold t. Unfortunately, their solution is notably inefficient when it comes to the escrow size: escrows require a number of group elements that scales *linearly* with d. As a concrete example, to allow up to 1000 units of currency per transaction would require each escrow to consume 385KB using the instantiation from [10, §8].

The PPB offering the most practical solution for threshold-comparison is a second construction in Kohlweiss *et al.* [16] which uses non-interactive secure computation (NISC). In this construction, escrows consist of the sender message in a NISC protocol (typically instantiated with oblivious-transfer (OT) ciphertexts and a garbled circuit) along with a proof that the NISC was computed honestly. While computing a "proof of garbling" requires proving over cryptographic primitives, and escrows must contain the full garbled truth tables, this solution remains plausible as a deployable PPB, though to the best of the authors' knowledge no implementations currently exist. One contribution of this work is to explore and optimize this approach.

Table 1: A comparison of escrow sizes and proof generation time for threshold comparison of 32-bit integers. Best-case FHE escrow size was estimated as TFHE [7] ciphertexts compressed using the TFHE-rs library [26]. The d in the escrow size for [10] scales linearly with the amount of currency permitted per-transaction, and would be impractically large for implementations at $d = 2^{32}$ in this example.

Scheme	Practical?	Generation time	Estimated Escrow Size (bytes)
Kohlweiss *et al.* [16] FHE	No	n/a	2,252
David *et al.* [10]	No	n/a	$384d + 1{,}152$
Kohlweiss *et al.* [16] NISC	Yes	11.95 seconds (see §5.)	13,472
This work	Yes	4.42 seconds	1,600

Our contributions. In this work we investigate the problem of constructing concretely efficient privacy-preserving blueprints for threshold comparison. We investigate two different approaches to this question:

1. **Realizing and optimizing NISC for PPBs.** As a baseline, we revisit the NISC approach mentioned by Kohlweiss *et al.* [16]. While the authors describe this construction as "theoretical" for general functionalities, we show that careful engineering allows us to realize the specific comparison functionality practically. The challenge here is careful circuit optimization, as well as the need to realize a ZK-friendly implementation of a garbled circuit, something that requires non-trivial optimization and design choices. We use this construction as a baseline to compare against later contributions.
2. **Developing a novel PPB based on ambiguous encryption.** To improve upon this result, we next develop a novel PPB construction that allows the auditor to perform an oblivious variant of the classic string comparison algorithm using "lossy" and ambiguous encryption [3]. Our construction makes use of the fact that, in our intended application, the threshold comparison functionality already leaks information about the user's input amount *in the specific case where the comparison is satisfied.* The resulting escrows require only $4 \log_b(\ell) + 7$ group elements, where ℓ is the maximum value of the threshold t and b is an arbitrary base chosen at setup time.

To verify the efficiency of our techniques, we implement both approaches under the assumption of a malicious user, and we present detailed performance and bandwidth numbers for each construction. We provide a comparison of estimated escrow sizes and prover times between this and prior work in Table 1.

In the next section we provide a high-level technical overview of our contributions.

1.1 Technical Overview

We are interested in a PPB for the following functionality:

$$f(t,(v,m)) = \begin{cases} m \text{ if } v > t \\ \perp \text{ otherwise} \end{cases}$$

In other words, our goal is to construct an encryption scheme where a ciphertext ct can only be decrypted if some associated value v is greater than a private threshold t, which is encoded in the public key.

Exploring and optimizing the NISC-based construction Kohlweiss *et al.* observe that a "generic" construction of PPBs can be derived from the use of non-interactive secure computation (NISC) and an appropriate ZK proof-of-knowledge [16]. However, they do not implement this construction, and mainly treat it as a "theoretical" alternative to their direct construction of a different functionality. Thus, as a baseline we wish to determine if this protocol can work in practice for the specific threshold comparison functionality we are interested in.

In practice, realizing this construction requires solving a number of engineering challenges. NISC constructions traditionally involve the use of a garbled circuit combined with a re-usable two-message OT scheme. The first challenge, therefore, is to identify an efficient circuit for threshold comparison that maximizes the advantages of efficient garbling schemes, such as the ability to calculate XOR gates for "free" [18]. A more imposing challenge is the need to construct efficient zero-knowledge proofs ensuring that the garbled circuit has been constructed correctly. We accomplish this using efficient zkSNARKs [14]. We show that even complex garbling optimizations such as Free-XOR [18] and Half-Gates [39] can be realized with careful optimizations, by storing wire labels and circuit tables as field elements, and by using zero-knowledge-friendly hash functions such as Poseidon [12].[5] While we use this result mainly as a baseline for comparing our novel scheme below, we find that the resulting implementation is borderline practical. As shown in Table 1, for 32-bit comparisons we are able to realize our escrows in less than 14KB, with approximately 12 seconds of proving time.

Our Scheme Garbled circuits by definition hide the full state of the computation up to the circuit's output. In the context of a PPB for threshold comparison, this means that the state of the comparison between t and v must remain completely hidden, even though m may *fully reveal* v, along with other information

[5] A surprising difficulty in this work is that the free-XOR technique cannot be natively employed on field elements, since the field addition operation is not its own inverse. The practical impact is that we must frequently convert lables between bit-string and field-element representation in our zk circuit, adding to the prover's computational overhead.

about the user. Our scheme improves upon the baseline by allowing information about v to be leaked in the case that $v > t$. Specifically, we build a PPB for the following functionality:

$$f_{thr}(t,(v,m)) = \begin{cases} (m, f'_b(v,t)) \text{ if } v > t \\ \perp \text{ otherwise} \end{cases}$$

where f'_b represents some extra information about v in relation to t. We will fully describe f'_b after going over the basics of our construction.

We now describe our main construction. Let ℓ be the maximum possible value of t. We can construct a PPB for f_{thr} as follows:

- The auditor publishes a public key consisting of $\ell + 1$ public keys for an underlying encryption scheme: $pk_0, \ldots, pk_\ell$, the first t of which are *lossy*, i.e. cannot decrypt any messages encrypted to them.
- The escrow is computed as $ct \leftarrow \mathsf{Enc}(pk_v, m)$, along with a zero-knowledge proof that ct was computed correctly.

If $v \leq t$, then pk_v is lossy, and so ct cannot be decrypted. Correspondingly, if $v > t$ then pk_v is non-lossy, and the message can be recovered. To decrypt the receiver tries all of their non-lossy secret keys until a decryption succeeds.

Unfortunately, this construction requires an impractically large public key that scales linearly with the maximum threshold. To solve this issue, we alter our construction to allow for a tradeoff between public key and ciphertext size. We achieve this tradeoff by using multiple copies of the above construction, where each copy, or "row," represents one digit of t in some base b. For example, if we are using base 10, max threshold $\ell = 99$, and $t = 42$, then the public key would consist of two rows of keys: $pk_0^0, \ldots, pk_0^{10}$ and $pk_1^0, \ldots, pk_1^{10}$, where $\{pk_0^0, pk_0^1, pk_0^2, pk_0^3, pk_0^4\}$ and $\{pk_1^0, pk_1^1, pk_1^2\}$ are lossy. Going forward, let $t[i]$ denote the digit of t at position i.

Say our value $v = 48$. To encrypt, we can start by computing the value $ct_0^{\mathsf{R}} \leftarrow \mathsf{Enc}(pk_4^0, m)$, as if $t[0] < 4$, then v is clearly larger than t. But what if $t[0] = 4$? In this case, we would like decryption to be possible if and only if $t[0] = 4$ *and* $t[1] < 8$. To achieve this, we can sample a one-time-pad (OTP) α, and compute the following two ciphertexts:

- $ct_0^{\mathsf{M}} \leftarrow \mathsf{Enc}(pk_0^5, \alpha)$
- $ct_1^{\mathsf{R}} \leftarrow \alpha \circ \mathsf{Enc}(pk_1^8, m)$

where $\alpha \circ x$ denotes blinding x with the one-time-pad. The final ciphertext is $ct = (ct_0^{\mathsf{R}}, ct_0^{\mathsf{M}}, ct_1^{\mathsf{R}})$. To decrypt, the receiver first attempts to decrypt ct_0^{R} with each non-lossy sk_0^i. If none of these succeed, they compute $ct_1^{\mathsf{R}'} \leftarrow \mathsf{Dec}(sk_0^5, ct_0^{\mathsf{M}}) \circ ct_1^{\mathsf{R}}$, attempt to decrypt $ct_1^{\mathsf{R}'}$ with each non-lossy sk_1^i, and if none of those succeed output $\perp$. We refer to ct_i^M as a "match" ciphertext, as it needs to be decrypted when a digit of v exactly equals a digit of t, and ct_i^R as a "reveal" ciphertext, as it directly reveals the message.

We note here the conflicting requirements between the match and reveal ciphertexts. Reveal ciphertexts must indicate whether a decryption succeeds in

order for the receiver to know that they have successfully decrypted the message, i.e. they must be *robust*. However, match ciphertexts must *hide* whether decryption was successful, as revealing so would also reveal information about the value v, even if $v < t$. Practically, this means that we must use different encryption schemes for the match and reveal ciphertexts. Additionally, both schemes must satisfy some form of *ambiguity*, which ensures that the ciphertexts don't reveal what public keys they were encrypted under.

The above technique can be extended to thresholds of any number of digits. To do so, a match ciphertext after the first encrypts not just a new one-time pad α_i, but instead $\alpha_i \circ \alpha_{i-1}$, i.e. a pad blinded by the *previous* OTP. This ensures that a reveal ciphertext can only be decrypted if *all* previous digits have been matched.

We can now see what $f'_b(v, t)$ reveals about v. The auditor learns the longest (base b) prefix of v such that each digit is greater than or equal to the corresponding digit of t.

2 Preliminaries

We use $[x, y]$ to denote the set $\{x, x+1, \ldots, y\}$, where $x, y \in \mathbb{Z}$ and $x \leq y$. We use $[x]$ as shorthand to denote $[1, x]$. When $x > y$ we allow $[x, y]$ to denote $\{x, x-1, \ldots, y\}$. For an integer x we use $x[i]_b$ to denote the 1-indexed i'th digit of x in base b, e.g. for the base-10 value $x = 357$, $x[1]_{10} = 3$. We may drop the subscript when the base is clear from context. We denote concatenation between the strings x and y with $x||y$, and the empty string as ε. For any algorithm $\mathcal{A}$, we use $y := \mathcal{A}(x; r)$ to denote that the algorithm runs on input x and random tape r. In most cases, we do not denote the random tape explicitly and simply write $y \leftarrow \mathcal{A}(x)$. We may write $y \xleftarrow{r} \mathcal{A}(x)$ to indicate sampling r as appropriate and recording it for later use. We use λ as the computational security parameter, and $\mathsf{negl}(\lambda)$ to denote negligible functions. For two probability ensembles $\{A_i\}_i$ and $\{B_i\}_i$, we use $\{A_i\}_i \stackrel{\text{p}}{=} \{B_i\}_i$ to denote that the ensembles are identical, $\{A_i\}_i \stackrel{\text{s}}{\approx} \{B_i\}_i$ to denote that the ensembles are statistically close, and $\{A_i\}_i \stackrel{\text{c}}{\approx} \{B_i\}_i$ to denote that the ensembles are computationally indistinguishable. We use $\Pr[E : A]$ to denote the probability of an event E in an experiment defined by executing A. For a finite set $\mathcal{X}$, we will use $\mathcal{U}_{\mathcal{X}}$ to denote the uniform distribution over $\mathcal{X}$. For a distribution $\mathcal{D}$ we will use $x \leftarrow \mathcal{D}$ to indicate sampling a value according to the distribution. We will use f_{thr} to denote the following function:

$$f_{thr}(t, (v, m)) = \begin{cases} (m, f'_b(v, t)) \text{ if } v > t \\ \perp \text{ otherwise} \end{cases}$$

Where $f'_b(v, t)$ is defined as follows.

$f'_b(v,t)$

$x = \varepsilon$
for $i \in [\lceil \log_b(t) \rceil]$
  if $v[i]_b \geq t[i]_b$ **then** $x = x || v[i]_b$ **else break**
return x

We provide descriptions of the Diffie-Hellman assumption, the ElGamal encryption scheme, one-time pads, and non-interactive zero-knowledge proofs in Appendix A.

2.1 Lossy Public-Key Encryption

Definition 1. *A lossy public-key encryption scheme (PKE) [27] for a message space $\mathcal{M}$ is a tuple of algorithms* $\mathsf{PKE} = (\mathsf{Setup}, \mathsf{KGen}, \mathsf{KGen}_{\mathsf{loss}}, \mathsf{Enc}, \mathsf{Dec})$ *defined as:*

- $\mathsf{Setup}(1^\lambda) \to \Lambda_k$*: outputs public parameters* Λ_k*.*
- $\mathsf{KGen}(\Lambda_k) \to (sk, pk)$*: generates a keypair* (sk, pk)*.*
- $\mathsf{KGen}_{\mathsf{loss}}(\Lambda_k) \to pk$*: generates a lossy key* pk*.*
- $\mathsf{Enc}(\Lambda_k, pk, m) \to ct$*: takes as input a public key* pk *and a plaintext message* $m \in \mathcal{M}$ *and outputs a ciphertext* ct*.*
- $\mathsf{Dec}(\Lambda_k, sk, ct) \to m$*: takes as input a secret key* sk *and a ciphertext* ct *and either outputs* $\perp$ *or the message* m*.*

We defer the definitions of standard PKE properties to Appendix A.6, and informally describe lossiness along with some additional properties we will require of our PKE schemes here. When invoking the PKE algorithms we may drop the Λ_k input for brevity when the value is clear from context.

Lossiness. We require that messages encrypted to keys generated by $\mathsf{KGen}_{\mathsf{loss}}$ are unrecoverable, even by the party that created the keys, and refer to such keys as *lossy*. We refer to non-lossy keys as *injective*. We additionally require that adversaries should be unable to distinguish between lossy and injective keys. We formalize these notions in Appendix A.6.

Robustness and Collision Freeness. We will require that the encryption schemes used in this work satisfy properties related to their behavior when a ciphertext is decrypted with the *wrong* key (i.e. a secret key unrelated to the public key the ciphertext was encrypted under). A *robust* encryption scheme [1, 23] requires that decrypting with the wrong key results in the output $\perp$. The weaker *collision-free* [23] requirement stipulates that decrypting an encryption of some message m with the wrong key results in an output $m' \neq m$. We give formal definitions of robustness and collision-freeness in Appendix A.7.

Lossy Ambiguity. An *ambiguous* PKE [3] requires that a decryptor be unable to tell which key a ciphertext was encrypted under. We will require two variants of this definition in relation to *lossy* keys. *Weak* lossy ambiguity will require ambiguity between ciphertexts encrypted under lossy keys, while the *strong* variant will require that ambiguity still hold even between lossy and injective keys. We formalize this notion in Appendix B.

Lossy Correlation Resistance. In order to preserve soundness, we will need a client to be unable to cause the PPB decryption process to terminate early. This in turn means that they should be unable to craft lossy ciphertexts such that their decryptions appear to be valid ciphertexts for some non-lossy key. We refer to this property as *lossy correlation resistance*, and define it in Appendix C.1.

2.2 Statistically Hiding Commitments

Definition 2. *A statistically hiding non-interactive commitment scheme is a tuple of algorithms* $\mathsf{C} = (\mathsf{CSetup}, \mathsf{Commit})$ *defined over a message space* $\mathbb{M}$ *and randomness space* $\mathbb{R}$ *defined as:*

- $\mathsf{CSetup}(1^\lambda) \to \Lambda_C$*: outputs public parameters* Λ_C*.*
- $\mathsf{Commit}(\Lambda_C, m; r) \to C$*: takes as input a message* $m \in \mathbb{M}$ *and some randomness* $r \in \mathbb{R}$ *and outputs a commitment* C*.*

We require that the above algorithms satisfy the following properties.

- ***Statistical Hiding****: For any* Λ_C *output by* $\mathsf{CSetup}(1^\lambda)$ *and for any* $m_0, m_1 \in \mathbb{M}$ *is should hold that*

$$\{\mathsf{Commit}(\Lambda_C, m_0; r) \ : \ r \leftarrow \mathbb{R}\} \stackrel{s}{\approx} \{\mathsf{Commit}(\Lambda_C, m_1; r) \ : \ r \leftarrow \mathbb{R}\}$$

- ***Computational Binding****: For all PPT adversaries* $\mathcal{A}$ *is should hold that*

$$\Pr\left[\begin{matrix}\mathsf{Commit}(\Lambda_C, m_0; r_0) = \mathsf{Commit}(\Lambda_C, m_1, r_1) \\ \wedge\, m_0 \neq m_1\end{matrix} \ : \ \begin{matrix}\Lambda_C \leftarrow \mathsf{CSetup}(1^\lambda) \\ (m_0, r_0, m_1, r_1) \leftarrow \mathcal{A}(\Lambda_C)\end{matrix}\right] \leq \mathsf{negl}(\lambda)$$

2.3 Privacy Preserving Blueprints

We reproduce here the PPB definition put forth in [10], adapted to the non-updatable case.

Definition 3. *Let* $\mathsf{C} = (\mathsf{CSetup}, \mathsf{Commit})$ *be a statistically hiding non-interactive commitment scheme defined over* $(\mathbb{M}, \mathbb{R})$*, and* $f : \mathbb{M} \times \mathbb{M} \to \mathbb{M}$ *be a function. A Privacy Preserving Blueprint (*f*-PPB) scheme is a tuple of algorithms* $(\mathsf{Setup}, \mathsf{KGen}, \mathsf{VerPK}, \mathsf{Escrow}, \mathsf{VerEscrow}, \mathsf{Dec})$ *defined as:*

- $\mathsf{Setup}(1^\lambda, \Lambda_C) \to (\Lambda, \mathsf{td})$*: takes as input the security parameter and the public parameters of the commitment scheme, and outputs parameters for the scheme* Λ*, which will contain* Λ_C*, along with a trapdoor* td*.*
- $\mathsf{KGen}(\Lambda, t, r) \to (sk, pk)$*: takes as input values* $t \in \mathbb{M}$ *and* $r \in \mathbb{R}$ *and outputs a key pair* (sk, pk)*.* (t, r) *define a commitment* C_{KG}*.*
- $\mathsf{VerPK}(\Lambda, pk, C_{KG}) \to b$*: takes as input a public key and a commitment, and outputs a bit* $b \in \{0, 1\}$ *verifying the validity of* pk*.*
- $\mathsf{Escrow}(\Lambda, pk, m, r) \to Z/\perp$*: takes as input a public key, a message, and some randomness, and returns an escrow value* Z *for the commitment* C_E *defined by* (m, r)*, or* $\perp$*.*
- $\mathsf{VerEscrow}(\Lambda, pk, C_E, Z) \to b$*: takes as input a public key, a commitment, and an escrow, and outputs a bit* $b \in \{0, 1\}$ *verifying the validity of the escrow.*
- $\mathsf{Dec}(\Lambda, sk, C_E, Z) \to m'/\perp$*: takes as input a secret key, a commitment, and an escrow, and returns* $m' = f(t, m) \in \mathbb{M}$ *or* $\perp$*.*

We require that a PPB satisfy the following properties:

- ***Correctness****: For all* $\Lambda_C \leftarrow \mathsf{CSetup}(1^\lambda)$*,* $\Lambda \leftarrow \mathsf{Setup}(1^\lambda, \Lambda_C)$*, all* $t, m \in \mathbb{M}$*, and all* $r \in \mathbb{R}$*:*

$$\Pr\left[\mathsf{Dec}(\Lambda, sk, C, Z) \neq f(t, m) \;:\; \begin{array}{c}(sk, pk) \leftarrow \mathsf{KGen}(\Lambda, t) \\ Z \leftarrow \mathsf{Escrow}(\Lambda, pk, m, r) \\ C \leftarrow \mathsf{Commit}(\Lambda_C, m; r)\end{array}\right] \leq \mathsf{negl}(\lambda)$$

- ***Soundness****: For all PPT adversaries* $\mathcal{A}$*, the adversary's advantage in the following game is negligible:*

$\mathsf{Sound}(\mathsf{PPB}, \mathcal{A}, \lambda)$

$\Lambda_C \leftarrow \mathsf{CSetup}(1^\lambda)$
$(\Lambda, \cdot) \leftarrow \mathsf{Setup}(1^\lambda, \Lambda_C)$
$t, r_k \leftarrow \mathcal{A}(1^\lambda, \Lambda)$
$(pk, sk) \leftarrow \mathsf{KGen}(\Lambda, t, r_k)$
$(C_E, m, r_E, Z) \leftarrow \mathcal{A}(pk)$
return $[C_E = \mathsf{Commit}(\Lambda_C, m; r_E) \wedge$
$\quad \mathsf{VerEscrow}(\Lambda, pk, C_E, Z) \wedge \mathsf{Dec}(\Lambda, sk, C_E, Z) \neq f(t, m)]$

The $\mathsf{Sound}(\mathsf{PPB}, \mathcal{A}, \lambda)$ *advantage of* $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{Sound}}_{\mathsf{PPB}, \mathcal{A}}(\lambda) = \Pr\left[\mathsf{Sound}(\mathsf{PPB}, \mathcal{A}, \lambda) \Rightarrow 1\right]$$

- ***Threshold Hiding****: For* $b \in \{0, 1\}$ *let* $\mathsf{ThrHide}_b(\mathsf{PPB}, \mathcal{A}, \lambda)$ *denote the result of the following probabilistic experiment:*

$\mathsf{ThrHide}_b(\mathsf{PPB}, \mathcal{A}, \lambda)$

$\Lambda_C \leftarrow \mathsf{CSetup}(1^\lambda)$
$(\Lambda, \cdot) \leftarrow \mathsf{Setup}(1^\lambda, \Lambda_C)$
$(t_0, t_1) \leftarrow \mathcal{A}(\Lambda)$
$r \leftarrow\!\!\$\ \mathbb{R}$
$(\cdot, pk) \leftarrow \mathsf{KGen}(\Lambda, t_b, r)$
$B \leftarrow \mathcal{A}(pk)$

The $\mathsf{ThrHide}(\mathsf{PPB}, \mathcal{A}, \lambda)$ *advantage of* $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{ThrHide}}_{\mathsf{PPB},\mathcal{A}}(\lambda) = |\Pr[\mathsf{ThrHide}_0(\mathsf{PPB}, \mathcal{A}, \lambda) \Rightarrow 1] - \Pr[\mathsf{ThrHide}_1(\mathsf{PPB}, \mathcal{A}, \lambda) \Rightarrow 1]|$$

A PPB satisfies threshold hiding if for all PPT adversaries $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{ThrHide}}_{\mathsf{PPB},\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$.

- ***Escrow Hiding****: For* $b \in \{0, 1\}$*, let* $\mathsf{EscHide}_b(\mathsf{PPB}, \mathcal{A}, \lambda)$ *denote the result of the following probabilistic experiment with a simulator* Sim*:*

$\mathsf{EscHide}_b(\mathsf{PPB}, \mathcal{A}, \mathsf{Sim}, \lambda)$

$\Lambda_C \leftarrow \mathsf{CSetup}(1^\lambda)$
$(\Lambda, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\lambda, \Lambda_C)$
$(t, r_{KG}, pk, x) \leftarrow \mathcal{A}(\Lambda)$
if $\mathsf{VerPK}(\Lambda, pk, \mathsf{Commit}(\Lambda_C, t; r_{KG})) = 0$: **return** $\perp$
$r \leftarrow\!\!\$\ \mathbb{R}$
$Z \leftarrow$ **if** $b = 0$ **then** $\mathsf{Escrow}(\Lambda, pk, x, r)$ **else** $\mathsf{Sim}(\mathsf{td}, pk, f(t, x))$
$B \leftarrow \mathcal{A}(Z)$
return B

The $\mathsf{EscHide}(\mathsf{PPB}, \mathcal{A}, \lambda)$ *advantage of* $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{EscHide}}_{\mathsf{PPB},\mathcal{A}}(\lambda) = |\Pr[\mathsf{EscHide}_0(\mathsf{PPB}, \mathcal{A}, \lambda) \Rightarrow 1] - \Pr[\mathsf{EscHide}_1(\mathsf{PPB}, \mathcal{A}, \lambda) \Rightarrow 1]|$$

A PPB satisfies escrow hiding if for all PPT adversaries $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{EscHide}}_{\mathsf{PPB},\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$.

3 An Efficient PPB For Threshold Comparison

We present here our main construction of an efficient PPB for threshold comparison. Let PKE_R and PKE_M be lossy public-key encryption schemes, and OTP, OTP' be one-time pad schemes, where $\mathsf{OTP}.\epsilon$ is some fixed key. Define the following two relations:

$$\mathcal{R}_{KG} = \left\{ ((pk', C), (t, sk', r_k, r_c)) \ : \ \begin{array}{c} (sk', pk') = \mathsf{KGen}'(\lambda, \Lambda_k^{\mathsf{R}}, \Lambda_k^{\mathsf{M}}, b, n, t; r_k) \\ \wedge\, C = \mathsf{Commit}(\Lambda_C, t; r_c) \end{array} \right\}$$

$$\mathcal{R}_{E} = \left\{ ((Z', C, pk), (v, m, r_e, r_c)) \ : \ \begin{array}{c} Z' = \mathsf{Escrow}'(\Lambda_k^{\mathsf{R}}, \Lambda_k^{\mathsf{M}}, n, pk, v, m; r_e) \\ \wedge\, C = \mathsf{Commit}(\Lambda_C, (v, m); r_c) \end{array} \right\}$$

Let $\mathcal{L}_{KG}$ and $\mathcal{L}_E$ be the corresponding languages to these relations, and Π_{KG}, Π_E be corresponding NIZKs for these languages. Our f_{thr}-PPB construction can then be seen in Figure 1 and Figure 2. To help provide intuition we give examples of our construction's output and computation in Appendix F.

We can now state our main theorem.

Theorem 1. *If* OTP' *is a secure OTP scheme,* OTP *is an abelian OTP scheme with message space* $\mathcal{O}$*,* $\mathsf{PKE}_{\mathsf{R}}$ *is a weakly robust, lossy, weak-lossy-ambiguous public-key encryption scheme,* $\mathsf{PKE}_{\mathsf{M}}$ *is a lossy,* $\mathcal{U}_{\mathcal{O}}$*-strong-lossy-ambiguous public-key encryption scheme,* $(\mathsf{PKE}_{\mathsf{R}}, \mathsf{PKE}_{\mathsf{M}}, \mathsf{OTP})$ *satisfy lossy correlation resistance,* $\mathsf{C} = (\mathsf{CSetup}, \mathsf{Commit})$ *is a secure commitment scheme,* Π_{KG} *is zero-knowledge and strong simulation extractable, and* Π_E *is zero-knowledge and satisfies knowledge soundness, then* (ℓ, b)*-TCPPB is a secure* f_{thr}*-PPB scheme.*

Proof of Theorem 1 can be found in Appendix C.

3.1 Concrete Instantiation

We describe here a concrete instantiation of our PPB scheme using a lossy variant of the ElGamal encryption scheme [11]. Our lossy variant uses an algorithm $\mathcal{H}$ that takes as input a security parameter 1^λ and outputs a group description with *two* generators $G = (\mathbb{G}, p, g, h)$, where the discrete log of h with respect to g is unknown. The scheme generates lossy keys by running the traditional ElGamal key generation algorithm with the alternate generator h. As the first element of a ciphertext remains g^r, computational lossiness follows from DDH. We give a full description of this scheme and prove prove that it satisfies collision-freeness, computational lossiness, and $\mathcal{U}_{\mathcal{O}}$-strong-lossy-ambiguity in Appendix D.

Our scheme additionally requires a PKE with weak-robustness and weak-lossy-ambiguity. We therefore give a generic transformation from a collision-free PKE to a weakly robust PKE that preserves lossiness and weak-lossy-ambiguity. The transform uses two public keys, and then has ciphertexts encrypt a public flag value ϕ under the second key. The transform can be seen in the Appendix in Figure 5. The transform clearly preserves lossiness, and we prove that it provides weak-robustness while preserving weak-lossy-ambiguity in Appendix D.

The final piece we need to instantiate our construction is suitable one-time-pad schemes. For OTP', we will use the natural scheme where a key α consists of a randomly sampled group element and messages are blinded via the group operation. For completeness, a description of this scheme can be seen in the Appendix in Figure 6. The other OTP scheme is applied to two objects in our

$\mathsf{Setup}(1^\lambda, \Lambda_C)$

$(\mathsf{crs}_{KG}, \mathsf{td}_{KG}) \leftarrow \mathsf{S}_{KG}(1^\lambda)$
$(\mathsf{crs}_E, \mathsf{td}_E) \leftarrow \mathsf{S}_E(1^\lambda)$
$\Lambda_k^{\mathsf{R}} \leftarrow \mathsf{PKE}_{\mathsf{R}}.\mathsf{Setup}(1^\lambda)$
$\Lambda_k^{\mathsf{M}} \leftarrow \mathsf{PKE}_{\mathsf{M}}.\mathsf{Setup}(1^\lambda)$
$n = \lceil \log_b(\ell) \rceil$
$\Lambda = (\lambda, \Lambda_C, \Lambda_k^{\mathsf{R}}, \Lambda_k^{\mathsf{M}}, \mathsf{crs}_{KG}, \mathsf{crs}_E, n)$
return $(\Lambda, (\mathsf{td}_{KG}, \mathsf{td}_E))$

$\mathsf{KGen}'\left(\lambda, \Lambda_k^{\mathsf{R}}, \Lambda_k^{\mathsf{M}}, n, t\right)$

for $i \in [n]$
  for $j \in [0, b]$
    if $j \leq t[i]$
      $rpk_i^j \leftarrow \mathsf{PKE}_{\mathsf{R}}.\mathsf{KGen}_{\mathsf{loss}}(\Lambda_k^{\mathsf{R}})$
      $mpk_i^j \leftarrow \mathsf{PKE}_{\mathsf{M}}.\mathsf{KGen}_{\mathsf{loss}}(\Lambda_k^{\mathsf{M}})$
      $\overline{sk}_i^j \leftarrow \perp$
    else
      $rsk_i^j, rpk_i^j \leftarrow \mathsf{PKE}_{\mathsf{R}}.\mathsf{KGen}(\Lambda_k^{\mathsf{R}})$
      $msk_i^j, mpk_i^j \leftarrow \mathsf{PKE}_{\mathsf{M}}.\mathsf{KGen}(\Lambda_k^{\mathsf{M}})$
      $\overline{sk}_i^j \leftarrow (rsk_i^j, msk_i^j)$
$sk' = \{\overline{sk}_i^j\}_{i \in [n], j \in [0, b-1]}$
$pk' = \{\overline{pk}_i^j\}_{i \in [n], j \in [0, b-1]}$
return (sk', pk')

$\mathsf{Escrow}(\Lambda, pk, (v, m), r_E)$

parse $(\lambda, \Lambda_C, \Lambda_k^{\mathsf{R}}, \Lambda_k^{\mathsf{M}}, \mathsf{crs}_{KG}, \mathsf{crs}_E, n) = \Lambda$
parse $(pk', C_{KG}, \pi_{KG}) = pk$
if $\mathsf{VerPK}(\Lambda, pk, C_{KG}) = 0$ **then return** $\perp$
$Z' \xleftarrow{r_e} \mathsf{Escrow}'(\Lambda_k^{\mathsf{R}}, \Lambda_k^{\mathsf{M}}, n, pk', v, m)$
$C_E \leftarrow \mathsf{Commit}(\Lambda_C, (v, m); r_E)$
$\pi_E \leftarrow \mathsf{P}_E(\mathsf{crs}_E, (Z', C_E, pk'), (v, m, r_e, r_E))$
return (Z', π_E)

$\mathsf{KGen}\,(\Lambda, t, r_{KG})$

parse $(\lambda, \Lambda_C, \Lambda_k^{\mathsf{R}}, \Lambda_k^{\mathsf{M}}, \mathsf{crs}_{KG}, \mathsf{crs}_E, n) = \Lambda$
$(sk', pk') \xleftarrow{r_k} \mathsf{KGen}'(\lambda, \Lambda_k^{\mathsf{R}}, \Lambda_k^{\mathsf{M}}, n, t)$
$C_{KG} \leftarrow \mathsf{Commit}(\Lambda_C, t; r_{KG})$
$\pi_{KG} \leftarrow \mathsf{P}_{KG}(\mathsf{crs}_{KG}, (pk', C_{KG}), (t, sk', r_k, r_{KG}))$
$pk \leftarrow (pk', C_{KG}, \pi_{KG})$
$sk \leftarrow (sk', pk)$
return (sk, pk)

$\mathsf{Escrow}'(\Lambda_k^{\mathsf{R}}, \Lambda_k^{\mathsf{M}}, n, pk, v, m)$

$\alpha^* \leftarrow \mathsf{OTP}'.\mathsf{KGen}()$
$\beta^* \leftarrow \mathsf{OTP}'.\mathsf{Enc}(\alpha^*, m)$
$\alpha_1, \ldots, \alpha_{n-1} \leftarrow \mathsf{OTP}.\mathsf{KGen}()$
$\alpha_0 \leftarrow \mathsf{OTP}.\varepsilon$
for $i \in [n]$
  $ct_i^{\mathsf{R}} \leftarrow \mathsf{PKE}_{\mathsf{R}}.\mathsf{Enc}(\Lambda_k^{\mathsf{R}}, rpk_i^{v[i]}, \alpha^*)$
  $\beta_i^{\mathsf{R}} \leftarrow \mathsf{OTP}.\mathsf{Enc}(\alpha_{i-1}, ct_i^{\mathsf{R}})$
  if $i < n$
    $\beta_i^{\mathsf{M}} \leftarrow \mathsf{OTP}.\mathsf{Enc}(\alpha_{i-1}, \alpha_i)$
    $ct_i^{\mathsf{M}} \leftarrow \mathsf{PKE}_{\mathsf{M}}.\mathsf{Enc}(\Lambda_k^{\mathsf{M}}, mpk_i^{v[i]+1}, \beta_i^{\mathsf{M}})$
  else
    $ct_i^{\mathsf{M}} \leftarrow \perp$
  $\overline{ct}_i := (\beta_i^{\mathsf{R}}, ct_i^{\mathsf{M}})$
return $(\{\overline{ct}_i\}_{i \in [n]}, \beta^*)$

Fig. 1: Our f_{thr}-PPB construction (ℓ, b)-TCPBB. (ℓ, b) are assumed to be implicit inputs to each algorithm.

$\mathsf{VerPK}(\Lambda, pk, C_{KG})$

parse $(\lambda, \Lambda_C, \mathsf{crs}_{KG}, \mathsf{crs}_E, \Lambda_k, \ell, b, n) = \Lambda$
parse $(pk', C'_{KG}, \pi_{KG}) = pk$
return $\mathsf{V}_{KG}(\mathsf{crs}_{KG}, \pi_{KG}, (pk', C_{KG}))$
  $\wedge\, (C'_{KG} = C_{KG})$

$\mathsf{VerEscrow}(\Lambda, pk, C_E, Z = (Z', \pi_E))$

parse $(\lambda, \Lambda_C, \mathsf{crs}_{KG}, \mathsf{crs}_E, \Lambda_k, \ell, b, n) = \Lambda$
parse $(pk', C'_{KG}, \pi_{KG}) = pk$
return $\mathsf{VerPK}(\Lambda, pk, C_{KG})$
  $\wedge\ \mathsf{V}_E((Z', C_E, pk'), \pi_E)$

$\mathsf{Dec}(\Lambda, sk, C_E, Z = (Z', \pi_E))$

parse $(\lambda, \Lambda_C, \Lambda_k^{\mathsf{R}}, \Lambda_k^{\mathsf{M}}, \mathsf{crs}_{KG}, \mathsf{crs}_E, n) = \Lambda$
parse $(\{(rsk_i^j, msk_i^j)\}_{i\in[n], j\in[0,b]}, pk)$
  $= sk$
if $\mathsf{VerEscrow}(\Lambda, pk, C_E, Z) = 0$ **return** $\perp$
parse $(\{\overline{ct}_i\}_{i\in[n]}, \beta^*) = Z'$
$\alpha'_0 := \mathsf{OTP}.\varepsilon$
for $i \in [n]$
  $(\beta_i^{\mathsf{R}}, ct_i^{\mathsf{M}}) \leftarrow \overline{ct}_i$
  $ct_i^{\mathsf{R}} \leftarrow \mathsf{OTP.Dec}(\alpha'_{i-1}, \beta_i^{\mathsf{R}})$
  for $j \in [t[i]+1, b]$
    $\alpha^{*\prime} = \mathsf{PKE_R.Dec}(\Lambda_k^{\mathsf{R}}, rsk_i^j, ct_i^{\mathsf{R}})$
    if $\alpha^{*\prime} \neq \perp$ **return** $\mathsf{OTP.Dec}(\alpha^{*\prime}, \beta^*)$
  if $i < n$
    $\beta_i^{\mathsf{M}} \leftarrow \mathsf{PKE_M.Dec}(\Lambda_k^{\mathsf{M}}, msk_i^{t[i]+1}, ct_i^{\mathsf{M}})$
    $\alpha'_i \leftarrow \mathsf{OTP.Dec}(\alpha'_{i-1}, \beta_i^{\mathsf{M}})$
return $\perp$

Fig. 2: Our f_{thr}-PPB construction continued.

construction: $\mathsf{PKE_R}$ ciphertexts and OTP keys. In Section 3.2 we will show an optimization that allows $\mathsf{PKE_R}$ ciphertexts to be represented with only two group elements. Therefore, we will have OTP consist of two copies of OTP' run in parallel.

We can then instantiate the scheme in Figure 1 as follows, where the base scheme is the lossy ElGamal variant described above:

- $\mathsf{PKE_R}$ is the scheme that results from applying the weak-robust transform to the base scheme.
- $\mathsf{PKE_M}$ is two copies of the base scheme run in parallel.
- OTP' is the scheme described above and shown in Figure 6, and OTP is two copies of OTP' run in parallel.

3.2 Optimizations

We describe here optimizations that reduce the public-key and escrow size of the concrete PPB scheme described in Section 3.1 while preserving security.

Randomness Reuse. As written, a $\mathsf{PKE_R}$ public-key would consist of two sub-keys $pk_0 = g^{x_0}, pk_1 = g^{x_1}$, and a ciphertext would consist of two ElGamal encryptions of the form $((g^y, g^{x_0 y} \cdot m), (g^z, g^{x_1 z} \cdot \phi))$. We can compress this ciphertext via randomness re-use. ElGamal encryption allows an encryptor to re-use the same g^r value across encryptions under different public-keys. This was first shown to be secure in [20], and later tested securely under a more stringent security definition in [4]. We can leverage this to use the same g^{r_R} value across *all* reveal ciphertexts, as each is under a different key. $\mathsf{PKE_R}$ ciphertexts now take the form: $(g^{x_0 r_\mathsf{R}} \cdot m, g^{x_1 r_\mathsf{R}} \cdot \phi)$, with g^{r_R} included separately in the escrow value.

Shared Public Keys. As our instantiated OTP scheme has keys α that consist of two group elements (α_1, α_2), match ciphertexts ct^M must consist of *two* ElGamal encryptions. For these, we can use the same (pk_0, pk_1) present in the $\mathsf{PKE_R}$ public key. We can additionally use the same randomness-reuse trick as before, using the same random value r_M for all match ciphertexts. A match ciphertext ct^M encrypting an OTP value $\alpha = (\alpha_0, \alpha_1)$ then has the form: $(g^{x_0 r_\mathsf{M}} \cdot \alpha_0, g^{x_1 r_\mathsf{M}} \cdot \alpha_1)$, with r_M included separately in the escrow value. With this optimization the scheme's public-key only needs to include two ElGamal public-keys in each cell, rather than four as before.

4 Implementation

As the goal of our construction is an efficient Escrow algorithm, our implementation and experiments focus on this portion of the schemes. Code for our implementations can be found at `https://github.com/anonfc2026/efficient-ppb`.

Our implementation for (ℓ, b)-TCPPB uses the concrete algorithm instantiations described in Section 3.1 and the optimizations described in Section 3.2.

For proof generation we implement Escrow' in Circom [6] version 2.2.2, using the Baby JubJub elliptic curve [37] as the underlying group for the encryption schemes. Proofs were generated using snarkjs [31].

The baseline scheme we compare against is the NISC-based PPB described in [16]. We instantiate the NISC with the well-known construction based on garbled circuits [38]. In this scheme, the auditor's public key consists of n oblivious transfer (OT) receiver messages, where n is bit length of the integers to be compared. An escrow is then a garbled n-bit comparison circuit, the labels corresponding to the user's input, and the OT sender messages encoding the remaining labels. We use the Free-XOR optimized comparator circuit from [17]. For completeness, we give a description of OT, garbled circuits, and this scheme in Appendix E.

A notable downside of this construction is that proving over the garbling algorithm requires non-black-box use of random oracles due to optimizations like Free-XOR [18] and Half-Gates [39]. Despite this caveat we have implemented this approach to provide the closest possible alternative to our main PPB scheme, giving it the best opportunity to serve as a viable comparison. We implement standard garbling with the Free-XOR and Half-Gates optimizations, and use Poseidon [12] as our hash function. Wire labels are stored as 254-bit arrays and converted to field elements before hashes are performed, with the resulting digest then converted back to a 254-bit array.

5 Experiments

Experiments were run on a desktop computer with an AMD Ryzen 5 5600X CPU and 32GB of RAM. We test our schemes on implementations for comparing integers up 2^{32} and 2^{64}. For (ℓ, b)-TCPPB we use $b = 41$, which results in $n = 6$ in the first case and $n = 12$ in the second.[6] The results of our experiments can be seen in Table 2. Our (ℓ, b)-TCPPB gives a $\geq 8\times$ improvement in escrow size, along with a $\geq 2\times$ improvement in prover time as compared to the baseline scheme.

Table 2: Escrow sizes and prover times of the baseline NISC-based PPB and the TCPPB presented in this work. The TCPPB uses base $b = 41$.

Scheme	ℓ	Escrow Size (bytes)	Prover Time (seconds)
Garbled Circuit	2^{32}	13,472	11.95
	2^{64}	26,784	30.06
TCPPB	2^{32}	1,600	4.42
	2^{64}	3,136	7.86

[6] We set $b = 41$ as it represents a reasonable tipping point for the escrow size: $\lceil \log_{40}(2^{32}) \rceil = 7$, $\lceil \log_{41}(2^{32}) \rceil = 6$.

References

[1] Michel Abdalla, Mihir Bellare, and Gregory Neven. "Robust Encryption". In: *TCC 2010*. Ed. by Daniele Micciancio. Vol. 5978. LNCS. Springer, Berlin, Heidelberg, Feb. 2010, pp. 480–497. DOI: 10.1007/978-3-642-11799-2_28.

[2] Prabhanjan Ananth and Abhishek Jain. "Pre-constrained encryption". In: *13th Innovations in Theoretical Computer Science Conference*. 2022.

[3] Gabrielle Beck et al. "Fuzzy Message Detection". In: *ACM CCS 2021*. Ed. by Giovanni Vigna and Elaine Shi. ACM Press, Nov. 2021, pp. 1507–1528. DOI: 10.1145/3460120.3484545.

[4] Mihir Bellare, Alexandra Boldyreva, and Jessica Staddon. "Randomness Re-use in Multi-recipient Encryption Schemeas". In: *PKC 2003*. Ed. by Yvo Desmedt. Vol. 2567. LNCS. Springer, Berlin, Heidelberg, Jan. 2003, pp. 85–99. DOI: 10.1007/3-540-36288-6_7.

[5] Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. "Foundations of garbled circuits". In: *Proceedings of the 2012 ACM conference on Computer and communications security*. 2012, pp. 784–796.

[6] Marta Bellés-Muñoz et al. "Circom: A circuit description language for building zero-knowledge applications". In: *IEEE Transactions on Dependable and Secure Computing* 20.6 (2022), pp. 4733–4751.

[7] Ilaria Chillotti et al. "TFHE: fast fully homomorphic encryption over the torus". In: *Journal of Cryptology* 33.1 (2020), pp. 34–91.

[8] CoinDesk. *Dubai Prohibits Privacy Coins Like Monero Under New Crypto Rules*. https://www.coindesk.com/policy/2023/02/08/dubai-prohibits-privacy-coins-under-new-crypto-rules/. 2023.

[9] Ronald Cramer and Victor Shoup. "A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack". In: *CRYPTO'98*. Ed. by Hugo Krawczyk. Vol. 1462. LNCS. Springer, Berlin, Heidelberg, Aug. 1998, pp. 13–25. DOI: 10.1007/BFb0055717.

[10] Bernardo David et al. "Updatable Privacy-Preserving Blueprints". In: *ASIACRYPT 2024, Part I*. Ed. by Kai-Min Chung and Yu Sasaki. Vol. 15484. LNCS. Springer, Singapore, Dec. 2024, pp. 105–139. DOI: 10.1007/978-981-96-0875-1_4.

[11] Taher ElGamal. "A public key cryptosystem and a signature scheme based on discrete logarithms". In: *IEEE transactions on information theory* 31.4 (1985), pp. 469–472.

[12] Lorenzo Grassi et al. "Poseidon: A new hash function for {Zero-Knowledge} proof systems". In: *30th USENIX Security Symposium (USENIX Security 21)*. 2021, pp. 519–535.

[13] Scott Griffy et al. *Privacy-Preserving Blueprints via Succinctly Verifiable Computation over Additively-Homomorphically Encrypted Data*. Cryptology ePrint Archive, Report 2024/675. 2024. URL: https://eprint.iacr.org/2024/675.

[14] Jens Groth. "On the Size of Pairing-Based Non-interactive Arguments". In: *EUROCRYPT 2016, Part II*. Ed. by Marc Fischlin and Jean-Sébastien

Coron. Vol. 9666. LNCS. Springer, Berlin, Heidelberg, May 2016, pp. 305–326. DOI: 10.1007/978-3-662-49896-5_11.

[15] Shai Halevi and Yael Tauman Kalai. "Smooth projective hashing and two-message oblivious transfer". In: *Journal of Cryptology* 25.1 (2012), pp. 158–193.

[16] Markulf Kohlweiss, Anna Lysyanskaya, and An Nguyen. "Privacy-Preserving Blueprints". In: *EUROCRYPT 2023, Part II*. Ed. by Carmit Hazay and Martijn Stam. Vol. 14005. LNCS. Springer, Cham, Apr. 2023, pp. 594–625. DOI: 10.1007/978-3-031-30617-4_20.

[17] Vladimir Kolesnikov, Ahmad-Reza Sadeghi, and Thomas Schneider. "Improved garbled circuit building blocks and applications to auctions and computing minima". In: *International Conference on Cryptology and Network Security*. Springer. 2009, pp. 1–20.

[18] Vladimir Kolesnikov and Thomas Schneider. "Improved garbled circuit: Free XOR gates and applications". In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2008, pp. 486–498.

[19] Ahmed Kosba et al. "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts". In: *2016 IEEE symposium on security and privacy (SP)*. IEEE. 2016, pp. 839–858.

[20] Kaoru Kurosawa. "Multi-recipient Public-Key Encryption with Shortened Ciphertext". In: *PKC 2002*. Ed. by David Naccache and Pascal Paillier. Vol. 2274. LNCS. Springer, Berlin, Heidelberg, Feb. 2002, pp. 48–63. DOI: 10.1007/3-540-45664-3_4.

[21] Sarah Meiklejohn et al. "A fistful of bitcoins: characterizing payments among men with no names". In: *Proceedings of the 2013 conference on Internet measurement conference*. 2013, pp. 127–140.

[22] P. E. Meltzer. "Keeping Drug Money from Reaching the Wash Cycle: A Guide to the Bank Secrecy Act". In: *Banking Law Journal* 108.3 (May 1991). NCJ Number: 132406, pp. 230–255.

[23] Payman Mohassel. "A Closer Look at Anonymity and Robustness in Encryption Schemes". In: *ASIACRYPT 2010*. Ed. by Masayuki Abe. Vol. 6477. LNCS. Springer, Berlin, Heidelberg, Dec. 2010, pp. 501–518. DOI: 10.1007/978-3-642-17373-8_29.

[24] Moni Naor and Omer Reingold. "Number-theoretic constructions of efficient pseudo-random functions". In: *Journal of the ACM* 51.2 (Mar. 2004), pp. 231–262. DOI: 10.1145/972639.972643.

[25] Shen Noether, Adam Mackenzie, et al. "Ring confidential transactions". In: *Ledger* 1 (2016), pp. 1–18.

[26] Jean-Baptiste Orfila, Arthur Meyre, and Agnes Leroy. *TFHE-rs v0.7: Ciphertext Compression, Multi-GPU Support and More*. Zama blog. urlhttps://www.zama.ai/post/tfhe-rs-v0-7-ciphertext-compression-multi-gpu-support-and-more. July 2024.

[27] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. "A framework for efficient and composable oblivious transfer". In: *Annual international cryptology conference*. Springer. 2008, pp. 554–571.

[28] Alexey Pertsev, Roman Semenov, and Roman Storm. "Tornado cash privacy solution version 1.4". In: *Tornado cash privacy solution version* 1.6 (2019).
[29] Eli Ben Sasson et al. "Zerocash: Decentralized anonymous payments from bitcoin". In: *2014 IEEE symposium on security and privacy*. IEEE. 2014, pp. 459–474.
[30] Jack Schickler. "Privacy-Enhancing Crypto Coins Could Be Banned Under Leaked EU Plans". In: *CoinDesk* (2022). URL: https://www.coindesk.com/policy/2022/11/15/privacy-enhancing-crypto-coins-could-be-banned-under-leaked-eu-plans/.
[31] *snarkjs*. Version 0.7.5. 2019. URL: https://github.com/iden3/snarkjs.
[32] Yiannis Tsiounis and Moti Yung. "On the Security of ElGamal Based Encryption". In: *Public Key Cryptography, First International Workshop on Practice and Theory in Public Key Cryptography, PKC '98, Pacifico Yokohama, Japan, February 5-6, 1998, Proceedings*. Ed. by Hideki Imai and Yuliang Zheng. Vol. 1431. Lecture Notes in Computer Science. Springer, 1998, pp. 117–134.
[33] U.S. Department of the Treasury. *U.S. Treasury Sanctions Notorious Virtual Currency Mixer Tornado Cash*. https://home.treasury.gov/news/press-releases/jy0916. Press release. 2022.
[34] U.S. Department of the Treasury. *U.S. Treasury Sanctions Notorious Virtual Currency Mixer Tornado Cash*. Press Release. Aug. 2022. URL: https://home.treasury.gov/news/press-releases/jy0916.
[35] Alexander Viand, Christian Knabenhans, and Anwar Hithnawi. "Verifiable fully homomorphic encryption". In: *arXiv preprint arXiv:2301.07041* (2023).
[36] Robert Viglione. "Japan's Ban Is a Wake-Up Call to Defend Privacy Coins". In: *CoinDesk* (2018). URL: https://www.coindesk.com/markets/2018/05/29/japans-ban-is-a-wake-up-call-to-defend-privacy-coins/.
[37] Barry WhiteHat, Jordi Baylina, and Marta Bellés. "Baby Jubjub elliptic curve". In: *Ethereum Improvement Proposal, EIP-2494* 29 (2020), p. 27.
[38] Andrew C Yao. "Protocols for secure computations". In: *23rd annual symposium on foundations of computer science (sfcs 1982)*. IEEE. 1982, pp. 160–164.
[39] Samee Zahur, Mike Rosulek, and David Evans. "Two halves make a whole: Reducing data transfer in garbled circuits using half gates". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2015, pp. 220–250.

A Preliminaries

A.1 Hardness of Discrete Log

Let $\mathbb{G}$ be a group of prime order and g be a generator. Let the variable $\mathsf{DL}(g, q, \mathcal{A})$ denote the result of the following probabilistic experiment:

$\underline{\mathsf{DL}(\mathbb{G}, g, q, \mathcal{A})}$
Sample $s \leftarrow \mathbb{Z}_q$
$s' \leftarrow \mathcal{A}(g, g^s)$
return $s' = s$

The $\mathsf{DL}(\mathbb{G}, g, q, \mathcal{A})$ advantage of $\mathcal{A}$ is defined as

$$\mathsf{Adv}^{\mathsf{DL}}_{\mathbb{G},\mathcal{A}}(\lambda) = \Pr[\mathsf{DL}(\mathbb{G}, g, q, \mathcal{A}) \Rightarrow 1]$$

If for all adversaries $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{DL}}_{\mathbb{G},\mathcal{A}}(\lambda)$ is negligible in λ, we say that discrete log is hard in $\mathbb{G}$.

A.2 The Decisional Diffie-Hellman Assumption

Let $\mathbb{G}$ be a group of prime order q and g be a generator. The tuple $(g, h_1, h_2, h_3) \in \mathbb{G}^4$ is called a Diffie-Hellman tuple if there exists $\alpha \in \mathbb{Z}_q$ such that $h_1 = g^\alpha$ and $h_3 = h_2^\alpha$. The decisional Diffie-Hellman (DDH) assumption in such a group $\mathbb{G}$ claims that it is computationally difficult to distinguish a random Diffie-Hellman (DH) tuple from uniformly random group elements. This notion is formalized as a bit-guessing game below.

Definition 4 (DDH Game). *Let the variable* $\mathsf{DDH}_b(\mathbb{G}, g, q, \mathcal{A})$ *where* $\mathbb{G}$ *is a group of prime order* q *with generator* g *and* $b \in \{0, 1\}$ *denote the result of the following probabilistic experiment:*

$\underline{\mathsf{DDH}_b(\mathbb{G}, g, q, \mathcal{A})}$
if $b = 0$:
  Sample $u, v \leftarrow \mathbb{Z}_q$
  $B \leftarrow \mathcal{A}(g, g^u, g^v, g^{uv})$
else
  Sample $u, v, w \leftarrow \mathbb{Z}_q$
  $B \leftarrow \mathcal{A}(g, g^u, g^v, g^w)$
return B

The $\mathsf{DDH}(\mathbb{G}, g, q, \mathcal{A})$ *advantage of* $\mathcal{A}$ *is defined as*

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathbb{G},\mathcal{A}}(\lambda) = \left|\Pr\left[\mathsf{DDH}_0(\mathbb{G}, g, q, \mathcal{A}) \Rightarrow 1\right] - \Pr\left[\mathsf{DDH}_1(\mathbb{G}, g, q, \mathcal{A}) \Rightarrow 1\right]\right|$$

As discussed in [27], DDH can be equivalently phrased as distinguishing between tuples of the form (g, h, g^y, h^y) and (g, h, g^y, h^z) for random generators g and h and random exponents y and z. We will use this form of the assumption in later proofs.

A.3 The ElGamal Encryption Scheme

The ElGamal public-key encryption scheme [11] in groups of prime order has been proven [32, 9, 24] secure against chosen-plaintext attacks under the assumption that the decisional Diffie-Hellman (DDH) problem is hard. Let $\mathcal{G}$ be an algorithm that takes as input a security parameter 1^λ and outputs a group description $G = (\mathbb{G}, p, g)$, where $\mathbb{G}$ is a cyclic group of prime order p and g is a generator of $\mathbb{G}$. The message space $\mathcal{M}$ of the scheme will be the generated group itself. A description of the scheme can be seen in Figure 3.

$\mathsf{Setup}(1^\lambda)$

1 : **return** $\mathcal{G}(1^\lambda)$

$\mathsf{KGen}(\Lambda_k)$	$\mathsf{Enc}(\Lambda_k, pk, M)$	$\mathsf{Dec}(\Lambda_k, sk, ct)$
1 : **parse** $(\mathbb{G}, g, p) = \Lambda_k$	1 : **parse** $(\mathbb{G}, g, p) = \Lambda_k$	1 : $(Y, W) \leftarrow ct$
2 : $x \leftarrow\!\!\$\ \mathbb{Z}_p$	2 : $y \leftarrow\!\!\$\ \mathbb{Z}_p$	2 : $T \leftarrow Y^x$
3 : $X \leftarrow g^x$	3 : $Y \leftarrow g^y$	3 : $M \leftarrow WT^{-1}$
4 : $pk \leftarrow (p, g, X)$	4 : $T \leftarrow X^y$	4 : **return** M
5 : $sk \leftarrow (p, g, x)$	5 : $W \leftarrow TM$	
6 : **return** (sk, pk)	6 : **return** (Y, W)	

Fig. 3: The ElGamal Encryption Scheme

A.4 One-Time Pad

Definition 5. *A one-time pad scheme over a space $\mathcal{O}$ is a tuple of algorithms* $\mathsf{OTP} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *defined as:*

- $\mathsf{KGen}() \to \alpha$*: outputs a key* $\alpha \in \mathcal{O}$*.*
- $\mathsf{Enc}(\alpha, m) \to \beta$*: takes a key and message* $\alpha, m \in \mathcal{O}$ *and outputs a ciphertext* $\beta \in \mathcal{O}$*.*
- $\mathsf{Dec}(\alpha, \beta) \to m \in \mathcal{O}$ *takes a key and ciphertext and outputs a message.*

We require that the above algorithms satisfy the following properties:

- ***Correctness****: For all messages m it should hold that*

$$\Pr\left[m = \mathsf{Dec}(\alpha, \mathsf{Enc}(\alpha, m)) \ : \ \alpha \leftarrow \mathsf{KGen}()\right] = 1$$

- ***Perfect Security****: For all m it should hold that*

$$\mathsf{Enc}(\mathsf{KGen}(), m) \stackrel{p}{=} \mathcal{U}_{\mathcal{O}}$$

A common method for building an OTP scheme is to have $\mathcal{O}$ be an abelian group, KGen sample an element of the group uniformly at random, and have $\mathsf{Enc}(\alpha, m) = \alpha \cdot m$ and $\mathsf{Dec}(\alpha, \beta) = \alpha^{-1} \cdot \beta$. The OTP scheme used in this work has such a structure, which we refer to as an *abelian* OTP.

A.5 Non-Interactive Zero-Knowledge

We reproduce here the NIZK definition from [10]. Let $\mathcal{R}$ be a polynomial-time verifiable binary relation. For a pair $(\mathbb{x}, \mathbb{w}) \in \mathcal{R}$, we refer to $\mathbb{x}$ as the statement and $\mathbb{w}$ as the witness. Let $\mathcal{L} = \{\mathbb{x} \mid \exists \mathbb{w} : (\mathbb{x}, \mathbb{w}) \in \mathcal{R}\}$.

Definition 6. *A NIZK proof system for a language $\mathcal{L}$ is a tuple of algorithms $\Pi = (\mathsf{S}, \mathsf{P}, \mathsf{V})$ defined as:*

- $\mathsf{S}(1^\lambda) \to (\mathsf{crs}, \mathsf{td})$*: outputs a common reference string* crs *and trapdoor* td*.*
- $\mathsf{P}(\mathsf{crs}, \mathbb{x}, \mathbb{w}) \to \pi$*: outputs a proof π for $(\mathbb{x}, \mathbb{w}) \in \mathcal{R}$.*
- $\mathsf{V}(\mathsf{crs}, \pi, \mathbb{x}) \to b$*: outputs $b \in \{0, 1\}$, verifying the proof π w.r.t. the public statement $\mathbb{x}$.*

A NIZK requires that the above algorithms satisfy the following properties:

- ***Perfect Completeness****: For* $\mathsf{crs} \leftarrow \mathsf{S}(1^\lambda)$*, and all $(\mathbb{x}, \mathbb{w}) \in \mathcal{R}$, it should hold that*
$$\Pr\left[\mathsf{V}(\mathsf{crs}, \mathsf{P}(\mathsf{crs}, \mathbb{x}, \mathbb{w}), \mathbb{x}) = 1\right] = 1$$
where the randomness is over the random coins of P*.*
- ***Soundness****: For all $\mathbb{x} \notin \mathcal{L}$ and all PPT adversaries $\mathcal{A}$, it should hold that*
$$\Pr\left[\mathsf{V}(\mathsf{crs}, \pi, \mathbb{x}) = 1 \ : \ \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{S}(1^\lambda) \\ \pi \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right] \leq \mathsf{negl}(\lambda)$$
where the randomness is over the random coins of S *and* $\mathcal{A}$*.*
- ***Zero-Knowledge****. There exists a PPT simulator* Sim *such that for all $(\mathbb{x}, \mathbb{w}) \in \mathcal{R}$*
$$\left\{(\mathsf{crs}, \pi) \;\middle|\; \begin{array}{c} (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{S}(1^\lambda) \\ \pi \leftarrow \mathsf{Sim}(\mathsf{crs}, \mathsf{td}, \mathbb{x}) \end{array}\right\} \stackrel{p}{=} \left\{(\mathsf{crs}, \pi) \;\middle|\; \begin{array}{c} (\mathsf{crs}, \cdot) \leftarrow \mathsf{S}(1^\lambda) \\ \pi \leftarrow \mathsf{P}(\mathsf{crs}, \mathbb{x}, \mathbb{w}) \end{array}\right\}$$

We may also require that a NIZK system satisfy one of the following properties.

- ***Knowledge Soundness****. For all PPT $\mathcal{A}$ there exists an extractor* $\mathsf{Ext}_\mathcal{A}$ *such that*
$$\Pr\left[\begin{array}{c} \mathsf{V}(\mathsf{crs}, \pi, \mathbb{x}) = 1 \wedge \\ (\mathbb{x}, \mathbb{w}) \notin \mathcal{R} \end{array} \ : \ \begin{array}{c} (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{S}(1^\lambda) \\ (\mathbb{x}, \pi) \leftarrow \mathcal{A}(\mathsf{crs}) \\ \mathbb{w} \leftarrow \mathsf{Ext}_\mathcal{A}(\mathsf{trans}_\mathcal{A}) \end{array}\right] \leq \mathsf{negl}(\lambda)$$
where $\mathsf{trans}_\mathcal{A}$ *is a list containing all of $\mathcal{A}$'s inputs as outputs.*

- ***Strong Simulation-Extractability***. *There exists an extractor* Ext *such that for all PPT* $\mathcal{A}$

$$\Pr\left[\begin{array}{c} \\ \mathsf{V}(\mathsf{crs},\pi,\mathbb{x})=1 \wedge \\ (\mathbb{x},\pi)\notin Q \wedge (\mathbb{x},\mathbb{w})\notin \mathcal{R} \\ \\ \end{array} : \begin{array}{c} (\mathsf{crs},\mathsf{td})\leftarrow \mathsf{S}(1^\lambda) \\ Q\leftarrow \emptyset \\ (\mathbb{x},\pi)\leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Sim}(\cdot,\cdot)}}(\mathsf{crs}) \\ \mathbb{w}\leftarrow \mathsf{Ext}(\mathsf{crs},\mathsf{td},\pi) \end{array}\right] \leq \mathsf{negl}(\lambda)$$

Where $\mathcal{O}_{\mathsf{Sim}(\mathbb{x},\mathbb{w})}$ *behaves as follows: it first checks whether* $(\mathbb{x},\mathbb{w})\in\mathcal{R}$*, and if not, aborts; otherwise it generates* $\pi\leftarrow\mathsf{Sim}(\mathsf{td},\mathbb{x})$*, sets* $Q\leftarrow Q\cup\{(\mathbb{x},\pi)\}$*, and returns* π*.*

A.6 Lossy Encryption

A lossy public-key encryption scheme has the interface described in Section 2.1, and should satisfy the following properties.

- **Correctness on real keys**: For all messages m, it should hold that

$$\Pr\left[m=\mathsf{Dec}(sk,\mathsf{Enc}(pk,m)) \;:\; (pk,sk)\leftarrow\mathsf{KGen}\left(1^\lambda\right)\right]=1$$

- **Lossiness on lossy keys**: For all $pk\leftarrow\mathsf{KGen}_{\mathsf{loss}}\left(1^\lambda\right)$ and m_0,m_1 such that $|m_0|=|m_1|$, *statistical lossiness* requires that

$$\mathsf{Enc}\left(pk,m_0\right)\overset{s}{\approx}\mathsf{Enc}\left(pk,m_1\right)$$

- **Indistinguishability of keys**: No PPT adversary can distinguish between the public keys output by the above algorithms, i.e. given $\Lambda_k\leftarrow\mathsf{Setup}(1^\lambda)$:

$$\mathsf{KGen}\left(\Lambda_k\right)\overset{c}{\approx}\mathsf{KGen}_{\mathsf{loss}}\left(\Lambda_k\right)$$

At times we may relax the lossiness property to require only *computational* lossiness. We define this below.

Let $\mathsf{PKE}=(\mathsf{Setup},\mathsf{KGen},\mathsf{KGen}_{\mathsf{loss}},\mathsf{Enc},\mathsf{Dec})$ be a lossy public key encryption scheme and let the variable $\mathsf{CompLoss}_b(\mathsf{PKE},\mathcal{A},\lambda)$ where $b\in\{0,1\}$ and $\lambda\in\mathbb{N}$ denote the result of the following probabilistic experiment:

$\underline{\mathsf{CompLoss}_b(\mathsf{PKE},\mathcal{A},\lambda)}$
$\Lambda_k\leftarrow\mathsf{Setup}(1^\lambda)$
$r,m_0,m_1\leftarrow\mathcal{A}(\Lambda_k)$
$pk\leftarrow\mathsf{KGen}_{\mathsf{loss}}(\Lambda_k;r)$
$C\leftarrow\mathsf{Enc}(\Lambda_k,pk,m_b)$
$B\leftarrow\mathcal{A}(C)$
return B

The $\mathsf{CompLoss}(\mathsf{PKE},\mathcal{A},\lambda)$ advantage of $\mathcal{A}$ is defined as

$$\mathsf{Adv}^{\mathsf{CompLoss}}_{\mathsf{PKE},\mathcal{A}}(\lambda)=\left|\Pr\left[\mathsf{CompLoss}_0(\mathsf{PKE},\mathcal{A},\lambda)\Rightarrow 1\right]-\Pr\left[\mathsf{CompLoss}_1(\mathsf{PKE},\mathcal{A},\lambda)\Rightarrow 1\right]\right|$$

A lossy encryption scheme PKE satisfies computational lossiness if for all PPT adversaries $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{CompLoss}}_{\mathsf{PKE},\mathcal{A}}(\lambda)$ is negligible.

A.7 Robust and Collision Free Encryption

Robustness. Weak robustness [1, 23] for a PKE scheme requires that a ciphertext does not decrypt to a valid plaintext under two different secret keys that correspond to distinct public keys. We repeat a simplified form of the definition below.

Definition 7 ((Weak) Robustness). *A PKE scheme satisfies weak robustness if for $\lambda \in \mathbb{N}$, it holds true that a PPT adversary's advantage in the following game is negligible:*

$\mathsf{Rob}(\mathsf{PKE}, \mathcal{A}, \lambda)$

$\Lambda_k \leftarrow \mathsf{Setup}()$
$\{sk_i, pk_i\} \leftarrow \mathsf{KGen}(\Lambda_k)\ \forall i \in [n]$
$(j, k, m, r) \leftarrow \mathcal{A}(1^\lambda, \{pk_i\}_{i\in[n]})$, *where* $0 \leq j, k < n$
$ct := \mathsf{Enc}(pk_j, m; r)$
if $j \neq k \wedge \mathsf{Dec}(sk_k, ct) \neq \perp$
  return 1
else
  return 0

The $\mathsf{Rob}(\mathsf{PKE}, \mathcal{A}, \lambda)$ *advantage of* $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{Rob}}_{\mathsf{PKE},\mathcal{A}}(\lambda) = \Pr\left[\mathsf{Rob}(\mathsf{PKE}, \mathcal{A}, \lambda) \Rightarrow 1\right]$$

An encryption scheme PKE *satisfies weak robustness if for all PPT adversaries* $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{Rob}}_{\mathsf{PKE},\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$.

Collision-Freeness. Mohassel [23] defines a relaxation of robustness called collision-freeness for PKE. Intuitively, collision-freeness requires that a ciphertext decrypts to two different plaintexts when decrypted using distinct secret keys. We define the corresponding variant with adversarial randomness below.

Definition 8 ((Weak) Collision-Freeness). *A PKE scheme satisfies collision-freeness with adversarial randomness if for $\lambda \in \mathbb{N}$, it holds true that a PPT adversary's advantage in the following game is negligible:*

$\mathsf{CollFree}(\mathsf{PKE}, \mathcal{A}, \lambda)$

$\Lambda_k \leftarrow \mathsf{Setup}()$
$\{sk_i, pk_i\} \leftarrow \mathsf{KGen}(\Lambda_k)\ \forall i \in [n]$
$(j, k, m, r) \leftarrow \mathcal{A}(1^\lambda, \{pk_i\}_{i\in[n]})$, *where* $0 \leq j, k < n$
$ct := \mathsf{Enc}(pk_j, m; r)$
$m' := \mathsf{Dec}(sk_k, ct)$
$b \leftarrow j \neq k \wedge \mathsf{IsEqual}(m, m')$
return b

The $\mathsf{CollFree}(\mathrm{PKE}, \mathcal{A}, \lambda)$ *advantage of* $\mathcal{A}$ *is defined as:*

$$\mathsf{Adv}^{\mathsf{CollFree}}_{\mathsf{PKE},\mathcal{A}}(\lambda) = \Pr\left[\mathsf{CollFree}(\mathsf{PKE}, \mathcal{A}, \lambda) \Rightarrow 1\right]$$

An encryption scheme PKE *satisfies weak collision-freeness if for all PPT adversaries* $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{CollFree}}_{\mathsf{PKE},\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$.

ElGamal was shown to satisfy collision-freeness in [23].

Theorem 2 ([23, Claim 51]). *ElGamal satisfies collision-freeness.*

B Lossy Ambiguous Encryption

Ambiguous encryption, originally defined in [3], requires that for certain distributions of messages an adversary cannot distinguish which public key a ciphertext was generated under, even with access to the corresponding secret keys. Our PPB construction will require a minor twist on this property: ambiguity with respect to *lossy* keys. Match ciphertexts must be ambiguous even between lossy and non-lossy keys, a property we call *strong-lossy-ambiguity*, while reveal ciphertexts only require ambiguity between lossy keys, which we call *weak-lossy-ambiguity*. As in [3], strong-lossy-ambiguity will be defined relative to a distribution $\mathcal{D}$. We define these properties formally below.

Definition 9 (Strong Lossy Ambiguous Encryption). *Let* $\mathsf{PKE} = (\mathsf{Setup}, \mathsf{KGen}, \mathsf{KGen}_{\mathsf{loss}}, \mathsf{Enc}, \mathsf{Dec})$ *be a lossy public-key encryption scheme for the message space* $\mathcal{M}$ *and let the random variable* $\mathcal{D}\text{-}\mathsf{SLAMB}_b(\mathsf{PKE}, \mathcal{A}, \mathsf{Dec}, \lambda)$ *where* $b \in \{0,1\}, \mathcal{D}$ *is a probability distribution over* $\mathcal{M}$*, and* $\lambda \in \mathbb{N}$*, denote the result of the following probabilistic experiment:*

$\mathcal{D}\text{-}\mathsf{SLAMB}_b(\mathsf{PKE}, \mathcal{A}, \lambda)$

$\Lambda_k \leftarrow \mathsf{Setup}(1^\lambda)$
$r_0, r_1 \leftarrow \mathcal{A}(\mathcal{D}, \Lambda_k)$
$(sk_0, pk_0) := \mathsf{KGen}(\Lambda_k; r_0)$
$pk_1 := \mathsf{KGen}_{\mathsf{loss}}(\Lambda_k; r_1)$
$m \leftarrow \mathcal{D}$
$C^* \leftarrow \mathsf{Enc}(pk_b, m)$
$B \leftarrow \mathcal{A}(\mathcal{D}, C^*)$
return B

The $\mathcal{D}\text{-}\mathsf{SLAMB}_b(\mathsf{PKE}, \mathcal{A}, \lambda)$ *advantage of an adversary* $\mathcal{A}$ *is defined as*

$$\mathsf{Adv}^{\mathcal{D}\text{-}\mathsf{SLAMB}}_{\mathsf{PKE},\mathcal{A}}(\lambda) = \left|\Pr\left[\mathcal{D}\text{-}\mathsf{SLAMB}_0(\mathsf{PKE}, \mathcal{A}, \lambda) \Rightarrow 1\right] - \Pr\left[\mathcal{D}\text{-}\mathsf{SLAMB}_1(\mathsf{PKE}, \mathcal{A}, \lambda) \Rightarrow 1\right]\right|$$

An encryption scheme PKE *is* $\mathcal{D}$-SLAMB *secure, i.e. satisfies* $\mathcal{D}$*-strong-lossy-ambiguity, if for all PPT adversaries* $\mathcal{A}$, $\mathsf{Adv}^{\mathcal{D}\text{-}\mathsf{SLAMB}}_{\mathsf{PKE},\mathcal{A}}(\lambda)$ *is negligible.*

Definition 10 (Weak Lossy Ambiguous Encryption). *Let* $\mathsf{PKE} = (\mathsf{Setup}, \mathsf{KGen}, \mathsf{KGen}_{\mathsf{loss}}, \mathsf{Enc}, \mathsf{Dec})$ *be a lossy public-key encryption scheme for the message space* $\mathcal{M}$ *and let the random variable* $\mathsf{WLAMB}_b(\mathsf{PKE}, \mathcal{A}, \mathsf{Dec}, \lambda)$*, where* $b \in \{0, 1\}$*, denote the result of the following probabilistic experiment:*

$\mathsf{WLAMB}_b(\mathsf{PKE}, \mathcal{A}, \lambda)$

$\Lambda_k \leftarrow \mathsf{Setup}(1^\lambda)$

$r_0, r_1, m \leftarrow \mathcal{A}(\Lambda_k)$

$pk_0 := \mathsf{KGen}_{\mathsf{loss}}(\Lambda_k; r_0)$

$pk_1 := \mathsf{KGen}_{\mathsf{loss}}(\Lambda_k; r_1)$

$C^* \leftarrow \mathsf{Enc}(pk_b, m)$

$B \leftarrow \mathcal{A}(C^*)$

return B

The $\mathsf{WLAMB}_b(\mathsf{PKE}, \mathcal{A}, \lambda)$ *advantage of an adversary* $\mathcal{A}$ *is defined as*

$$\mathsf{Adv}^{\mathsf{WLAMB}}_{\mathsf{PKE},\mathcal{A}}(\lambda) = |\Pr[\mathsf{WLAMB}_0(\mathsf{PKE}, \mathcal{A}, \lambda) \Rightarrow 1] - \Pr[\mathsf{WLAMB}_1(\mathsf{PKE}, \mathcal{A}, \lambda) \Rightarrow 1]|$$

An encryption scheme PKE *is* WLAMB *secure, i.e. satisfies weak-lossy-ambiguity, if for all PPT adversaries* $\mathcal{A}$*,* $\mathsf{Adv}^{\mathsf{WLAMB}}_{\mathsf{PKE},\mathcal{A}}(\lambda)$ *is negligible.*

We now show that strong-lossy-ambiguity is strictly stronger than weak-lossy-ambiguity.

Theorem 3. *Strong-lossy-ambiguity is strictly stronger than weak-lossy-ambiguity.*

Proof. We begin by showing that a scheme satisfying $\mathcal{D}$-strong-lossy-ambiguity for any $\mathcal{D}$ also satisfies weak-lossy-ambiguity. Let PKE be a $\mathcal{D}$-strong-lossy-ambiguous public key encryption scheme, and consider the following sequence of games. For a game Game_i, let $\mathcal{G}_i = \Pr[\mathsf{Game}_i(\mathsf{PKE}, \mathcal{A}, \lambda) \Rightarrow 1]$.

- Game_0: This game is identical to $\mathsf{WLAMB}_0(\mathsf{PKE}, \mathcal{A}, \lambda)$.
- Game_1: Sample $m' \leftarrow \mathcal{D}$, and replace C^* with $\mathsf{Enc}(pk_0, m')$. By the lossiness of PKE, $\mathcal{G}_0 \overset{c}{\approx} \mathcal{G}_1$.
- Game_2: In this game, pk_1 is computed using the non-lossy KGen algorithm, and $\mathcal{A}$ receives $\mathsf{Enc}(pk_1, m)$. By the $\mathcal{D}$-strong-lossy-ambiguity property of PKE, $\mathcal{G}_1 \overset{c}{\approx} \mathcal{G}_2$.
- Game_3: In this game pk_1 is computed with the lossy KGen algorithm. We have that $\mathcal{G}_2 \overset{c}{\approx} \mathcal{G}_3$ again by the strong-lossy-ambiguity of PKE.
- Game_4: This game is identical to $\mathsf{WLAMB}_1(\mathsf{PKE}, \mathcal{A}, \lambda)$. $\mathcal{G}_3 \overset{c}{\approx} \mathcal{G}_4$ again by the lossiness of PKE.

We therefore conclude that $\mathcal{G}_0 \overset{c}{\approx} \mathcal{G}_4$, and thus PKE satisfies weak-lossy-ambiguity.

Finally, consider a PKE scheme that satisfies weak-lossy-ambiguity, and modify it so that encryption appends a copy of the encryption key to the message before computing a ciphertext. This scheme clearly maintains weak-lossy-ambiguity

as ciphertexts encrypted under lossy keys reveal no information about the underlying message, while it leads to a trivial attack against strong-lossy-ambiguity by an adversary that decrypts a ciphertext and checks for the non-lossy public key.

C Main Construction Proofs

We give here the correctness and security proofs of our main PPB construction shown in Figure 1 and Figure 2.

We begin with describing the intuition regarding the soundness proof, as it will require an additional property of the encryption schemes used in the construction. To show soundness we will need to show that $\mathsf{PKE_R}$ both decrypts to the correct message at the proper reveal index when $v > t$, and that it *never* (except with negligible probability) decrypts to any message when $v < t$. The former follows straightforwardly from the correctness of the decryption schemes, while the latter is more subtle. Proving the latter is challenging as the adversary has control over selecting the randomness used in encryption. Therefore, it is plausible that they could select some combination of $\mathsf{PKE_R}$, $\mathsf{PKE_M}$, and OTP encryption randomness such that decrypting a match ciphertext with the wrong key leads to some custom reveal ciphertext that the decryption algorithm recognizes as legitimate. The key observation here is that as $v < t$, there must be some index for which the match ciphertext is encrypted under a *lossy* key. We need that the lossiness of the encryption scheme prevents the adversary from carefully selecting its randomness to produce the bad case discussed above, and with this property soundness can be proved. We call the needed property "lossy correlation resistance," and describe it in the following subsection.

C.1 Lossy Correlation Resistance

Here we define a property that we will require of the encryption and one-time-pad schemes used by our construction, and then prove that our concrete instantiations satisfy this property. Intuitively, the property will require that an adversary cannot bias a lossy ciphertext in a particular way.

Let $\mathsf{PKE_R}$ and $\mathsf{PKE_M}$ be lossy public-key encryption schemes, and OTP be a one-time pad scheme. Let the variable $\mathsf{LossyCor}(\mathsf{PKE_R}, \mathsf{PKE_M}, \mathsf{OTP}, \mathcal{A}, \lambda)$ where $\lambda \in \mathbb{N}$ denote the result of the following probabilistic experiment:

$\mathsf{LossyCor}(\mathsf{PKE_R}, \mathsf{PKE_M}, \mathsf{OTP}, \mathcal{A}, \lambda)$

$\Lambda_k^{\mathsf{M}} \leftarrow \mathsf{PKE_M.Setup}(1^\lambda)$

$\Lambda_k^{\mathsf{R}} \leftarrow \mathsf{PKE_R.Setup}(1^\lambda)$

$mpk_\ell := \mathsf{PKE_M.KGen_{loss}}(\Lambda_k^{\mathsf{M}})$

$(msk, mpk) := \mathsf{PKE_M.KGen}(\Lambda_k^{\mathsf{M}})$

$(rsk_0, rpk_0) := \mathsf{PKE_R.KGen}(\Lambda_k^{\mathsf{R}})$

$(rsk_1, rpk_1) := \mathsf{PKE_R.KGen}(\Lambda_k^{\mathsf{R}})$

$(\epsilon^{\mathsf{M}}, \epsilon^{\mathsf{R}}, \kappa_0^{\mathsf{O}}, \kappa_1^{\mathsf{O}}, \kappa_2^{\mathsf{O}}, \kappa_3^{\mathsf{O}}) \leftarrow \mathcal{A}(\Lambda_k^{\mathsf{M}}, \Lambda_k^{\mathsf{R}}, mpk_\ell, mpk, rpk_0, rpk_1)$

for $i \in [0,3]:\ \alpha_i := \mathsf{OTP.KGen}(;\kappa_i^{\mathsf{O}})$

return $\mathsf{PKE_R.Dec}(rsk_1,$

$\mathsf{OTP.Dec}($

$\mathsf{OTP.Dec}(\alpha_0, \mathsf{PKE_M.Dec}(msk, \mathsf{PKE_M.Enc}(\Lambda_k, mpk, \alpha_2; \epsilon^{\mathsf{M}})),$

$\mathsf{OTP.Enc}(\alpha_1, \mathsf{PKE_R.Enc}(rpk_0, \alpha_3; \epsilon^{\mathsf{R}}))$

$)) \neq \perp$

The $\mathsf{LossyCor}(\mathsf{PKE_R}, \mathsf{PKE_M}, \mathsf{OTP}, \mathcal{A}, \lambda)$ of $\mathcal{A}$ is defined as:

$$\mathsf{Adv}^{\mathsf{LossyCor}}_{\mathsf{PKE_R},\mathsf{PKE_M},\mathsf{OTP},\mathcal{A}}(\lambda) = \Pr\left[\mathsf{LossyCor}(\mathsf{PKE_R}, \mathsf{PKE_M}, \mathsf{OTP}, \mathcal{A}, \lambda) = 1\right]$$

We say that schemes $(\mathsf{PKE_R}, \mathsf{PKE_M}, \mathsf{OTP})$ together satisfy lossy-correlation-resistance if for all PPT adversaries $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{LossyCor}}_{\mathsf{PKE_R},\mathsf{PKE_M},\mathsf{OTP},\mathcal{A}}(\lambda)$ is negligible.

Lemma 1. *The instantiations of* $\mathsf{PKE_R}$, $\mathsf{PKE_M}$, *and* OTP *described in Section 3.1, satisfy lossy-correlation-resistance if the discrete logarithm problem is hard in* $\mathbb{G}$.

Proof. We begin by recalling from Section 3.2 that $\mathsf{PKE_R}$ and $\mathsf{PKE_M}$ are both made up of two copies of an underlying ElGamal PKE scheme over a group $\mathbb{G}$ run in parallel. We will abuse notation and use the same variables above to describe only the *second* copy of the scheme (i.e. the one in the $\mathsf{PKE_R}$ construction that encrypts ϕ). For readability will also replace the OTP operations with group notation, and refrain from listing the g^y portion of ElGamal ciphertexts.

Assume that there exists an adversary $\mathcal{A}$ such that $\mathsf{Adv}^{\mathsf{LossyCor}}_{\mathsf{PKE_R},\mathsf{PKE_M},\mathsf{OTP},\mathcal{A}}(\lambda)$ is non-negligible. We can then construct a reduction $\mathcal{B}$ to the discrete log problem in $\mathbb{G}$. $\mathcal{B}$ receives (g, h), samples $(\kappa_0^{\mathsf{R}}, \kappa_1^{\mathsf{R}}, \kappa_0^{\mathsf{M}}, \kappa_1^{\mathsf{M}}) \leftarrow\$ \mathbb{Z}$, invokes $\mathcal{A}$ on $((g,h), h^{\kappa_1^{\mathsf{M}}}, g^{\kappa_0^{\mathsf{M}}}, g^{\kappa_0^{\mathsf{R}}}, g^{\kappa_1^{\mathsf{R}}})$, and receives $(\epsilon^{\mathsf{M}}, \epsilon^{\mathsf{R}}, \kappa_0^{\mathsf{O}}, \kappa_1^{\mathsf{O}}, \kappa_2^{\mathsf{O}}, \kappa_3^{\mathsf{O}})$. $\mathcal{B}$ then outputs $\kappa_0^{\mathsf{O}} + \kappa_1^{\mathsf{O}} + \kappa_3^{\mathsf{O}} + \kappa_1^{\mathsf{M}}\epsilon^{\mathsf{M}} + \kappa_0^{\mathsf{R}}\epsilon^{\mathsf{R}} - \kappa_1^{\mathsf{R}}\epsilon^{\mathsf{R}} - \kappa_2^{\mathsf{O}} - \kappa_0^{\mathsf{M}}\epsilon^{\mathsf{M}}$.

As $\mathcal{A}$ has non-negligible advantage, we must have that with non-negligible probability:

$$\mathsf{PKE_R.Dec}(rsk, (\alpha_0^{-1} \cdot \mathsf{PKE_M.Dec}(msk, ct^{\mathsf{M}}))^{-1} \cdot (\alpha_1 \cdot \mathsf{PKE_R.Enc}(rpk, \alpha_3; \epsilon^{\mathsf{R}}))) \neq \perp$$

$$(\alpha_0^{-1} \cdot \mathsf{PKE_M.Dec}(msk, ct^{\mathsf{M}}))^{-1} \cdot (\alpha_1 \cdot \mathsf{PKE_R.Enc}(rpk, \alpha_3; \epsilon^{\mathsf{R}})) = g^{\kappa_1^{\mathsf{R}}\epsilon^{\mathsf{R}}}$$

(Definition of $\mathsf{PKE_R}$)

$$(\alpha_0^{-1} \cdot \frac{h^{\kappa_0^{\mathsf{M}}\epsilon^{\mathsf{M}}}\alpha_2}{g^{\kappa_1^{\mathsf{M}}\epsilon^{\mathsf{M}}}})^{-1} \cdot (\alpha_1 \cdot g^{\kappa_0^{\mathsf{R}}\epsilon^{\mathsf{R}}}\alpha_3) = g^{\kappa_1^{\mathsf{R}}\epsilon^{\mathsf{R}}}$$

(Definitions of $\mathsf{PKE_R}$ and $\mathsf{PKE_M}$)

$$\frac{\alpha_0\alpha_1\alpha_3 g^{\kappa_1^{\mathsf{M}}\epsilon^{\mathsf{M}}} g^{\kappa_0^{\mathsf{R}}\epsilon^{\mathsf{R}}}}{\alpha_2 h^{\kappa_0^{\mathsf{M}}\epsilon^{\mathsf{M}}}} = g^{\kappa_1^{\mathsf{R}}\epsilon^{\mathsf{R}}}$$

$$\frac{\alpha_0\alpha_1\alpha_3 g^{\kappa_1^{\mathsf{M}}\epsilon^{\mathsf{M}}} g^{\kappa_0^{\mathsf{R}}\epsilon^{\mathsf{R}}}}{\alpha_2 g^{\kappa_1^{\mathsf{R}}\epsilon^{\mathsf{R}}}} = h^{\kappa_0^{\mathsf{M}}\epsilon^{\mathsf{M}}}$$

$$g^{\kappa_0^{\mathsf{O}}+\kappa_1^{\mathsf{O}}+\kappa_3^{\mathsf{O}}+\kappa_1^{\mathsf{M}}\epsilon^{\mathsf{M}}+\kappa_0^{\mathsf{R}}\epsilon^{\mathsf{R}}-\kappa_1^{\mathsf{R}}\epsilon^{\mathsf{R}}-\kappa_2^{\mathsf{O}}} = h^{\kappa_0^{\mathsf{M}}\epsilon^{\mathsf{M}}}$$

$$\kappa_0^{\mathsf{O}} + \kappa_1^{\mathsf{O}} + \kappa_3^{\mathsf{O}} + \kappa_1^{\mathsf{M}}\epsilon^{\mathsf{M}} + \kappa_0^{\mathsf{R}}\epsilon^{\mathsf{R}} - \kappa_1^{\mathsf{R}}\epsilon^{\mathsf{R}} - \kappa_2^{\mathsf{O}} - \kappa_0^{\mathsf{M}}\epsilon^{\mathsf{M}} = s$$

Where s is the discrete log of h with respect to g. Therefore $\mathcal{B}$ succeeds with non-negligible probability, which contradicts our assumption that discrete log is hard in $\mathbb{G}$, and so $(\mathsf{PKE_R}, \mathsf{PKE_M}, \mathsf{OTP})$ satisfy lossy-correlation-resistance.

C.2 Iterative OTP Decryption

We now introduce some notation for iteratively decrypting a series of OTP ciphertexts. For an OTP scheme OTP, define $\mathsf{OTP.RDec}(x, y_1, \ldots, y_n)$ as follows:

$\underline{\mathsf{OTP.RDec}(x, y_1, \ldots, y_n)}$
$\alpha := x$
for $i \in [n]$
  $\alpha := \mathsf{OTP.Dec}(\alpha, y_i)$
return α

Lemma 2. *For an abelian OTP scheme* OTP *as described in Appendix A.4 over a group with generator g, given* $r_0, r_1, \ldots, r_n$*, one can efficiently compute a value* r' *such that* $\mathsf{OTP.RDec}(g^{r_0}, g^{r_1}, \ldots, g^{r_n}) = \mathsf{OTP.Dec}(g^{r'}, g^{r_1})$.

Proof. Let $f(r_0, r_2, \ldots, r_n) = \sum_{i \in \{0,2,\ldots,n\}}((\prod_{n-i} -1)r_i)$

We then have that:

$$\mathsf{OTP.RDec}(g^{r_0}, g^{r_1}, \ldots, g^{r_n}) =$$
$$(g^{r_0})^{(\prod_n -1)} \cdot (g^{r_1})^{(\prod_{n-1} -1)} \cdot \ldots \cdot g^{r_n} =$$
$$g^{f(r_0, r_2, \ldots, r^n)} \cdot (g^{r_1})^{(\prod_{n-1} -1)}$$

Then, if n is even we have that:

$$\begin{aligned} g^{f(r_0,r_2,\ldots,r^n)} \cdot (g^{r_1})^{(\prod_{n-1}-1)} &= g^{f(r_0,r_2,\ldots,r^n)} \cdot g^{r_1} \\ &= (g^{-f(r_0,r_2,\ldots,r^n)})^{-1} \cdot g^{r_1} \end{aligned}$$

and so $r' = -f(r_0, r_2, \ldots, r^n)$.

In the case that n is odd we have that:

$$\begin{aligned} g^{f(r_0,r_2,\ldots,r^n)} \cdot (g^{r_1})^{(\prod_{n-1}-1)} &= g^{f(r_0,r_2,\ldots,r^n)} \cdot g^{-r_1} \\ &= g^{f(r_0,r_2,\ldots,r^n)} \cdot g^{-r_1} g^{r_1} g^{-r_1} \\ &= g^{f(r_0,r_2,\ldots,r^n)-2r_1} g^{r_1} \\ &= (g^{-f(r_0,r_2,\ldots,r^n)+2r_1})^{-1} g^{r_1} \end{aligned}$$

and so $r' = -f(r_0, r_2, \ldots, r^n) + 2r_1$.

C.3 Main Theorem

We provide here proof of Theorem 1.

As our construction uses the generic commit-and-prove technique from [16], the proofs of correctness of $\mathsf{VerEscrow}$ and VerPK, follow from the same arguments, and we omit the proofs here.

Lemma 3. *If Π_E is a knowledge-sound NIZK, C is a binding commitment scheme, $\mathsf{PKE_R}$ satisfies weak-robustness, OTP' is correct, and $\mathsf{PKE_R}$, $\mathsf{PKE_M}$, and OTP are correct and satisfy lossy correlation resistance, then (ℓ, b)-TCPPB satisfies soundness.*

Proof. Suppose for the sake of contradiction that there exists an adversary $\mathcal{A}$ such that $\mathsf{Adv}^{\mathsf{Sound}}_{\mathsf{TCPPB},\mathcal{A}}(\lambda) = \nu(\lambda)$ is non-negligible. In the adversary's first stage it outputs (t, r_{KG}), and the challenger sets $(sk, pk) \leftarrow \mathsf{KGen}(\Lambda, t, r_{KG})$. In its second stage, the adversary $\mathcal{A}$ outputs $(C_E, (v, m), r_E, Z = (Z', \pi_E))$. We can divide our analysis into the following three cases depending on the properties of these outputs.

1. There exist some r'_E and $(v', m') \neq (v, m)$ such that $\mathsf{Commit}((v, m), r_E) = \mathsf{Commit}((v', m'), r'_E)$ and $Z = \mathsf{Escrow}(\Lambda, pk, (v', m'), r'_E)$.
2. $Z' \neq \mathsf{Escrow}'(\Lambda^{\mathsf{R}}_k, \Lambda^{\mathsf{M}}_k, n, pk, v, m)$ and (1) does not hold.
3. Neither (1) nor (2) hold (i.e. $\mathcal{A}$'s output is well-formed).

Let the probability of $\mathcal{A}$'s outputs satisfying property j be denoted by $\nu_j(\lambda)$. If $\nu(\lambda)$ is non-negligible, then at least one of $\nu_1(\lambda)$, $\nu_2(\lambda)$, $\nu_3(\lambda)$ must be non-negligible.

Suppose that $\nu_2(\lambda)$ is non-negligible. In this case $\mathcal{A}$ produced a proof of a false statement and we can construct a reduction $\mathcal{B}$ to the knowledge-soundness of Π_E.

$\mathcal{B}$ runs $\mathcal{A}$, simulates the generation of pk, receives $(C_E, (v, m), r_E, Z = (Z', \pi_E))$, and outputs $((Z', C_E, pk), \pi_E)$. As the statement is false any extractor must fail, meaning $\Pr[\mathcal{B} \text{ wins}]$ is non-negligible, thus contradicting our assumption that Π_E satisfies knowledge soundness. Therefore, $\nu_2(\lambda)$ is negligible.

As $\nu_2(\lambda)$ is negligible we can assume there exists an extractor $\mathsf{Ext}_{\mathcal{A}}$ that given $\mathsf{trans}_{\mathcal{A}}$ outputs (v', m', r_e, r_E) such that $C_E = \mathsf{Commit}(\Lambda_C, (v', m'); r_E)$ and $Z' = \mathsf{Escrow}'(\Lambda_k^{\mathsf{R}}, \Lambda_k^{\mathsf{M}}, n, pk, v', m'; r_e)$.

Suppose $\nu_1(\lambda)$ is non-negligible. In this case we can break the computational binding property of the commitment scheme by using a reduction that uses the extractor and outputs (m, r_E, m', r_c). Therefore, $\nu_1(\lambda)$ must be negligible.

Suppose $\nu_3(\lambda)$ is non-negligible. In this case we can break a property of $\mathsf{PKE}_{\mathsf{R}}$, $\mathsf{PKE}_{\mathsf{M}}$, or OTP. In all cases, the reduction begins by using the extractor to learn (v', m', r_e, r_E). The reduction can then use r_e and r_k to infer the randomness used to generate each individual value in Escrow'. We define notation for these values as follows:

- κ_i^{O} is the randomness used at iteration i of Escrow' for $\mathsf{OTP.KGen}$.
- ϵ_i^{R} and ϵ_i^{M} are the randomness used at iteration i of Escrow' for $\mathsf{PKE}_{\mathsf{R}}.\mathsf{Enc}$ and $\mathsf{PKE}_{\mathsf{M}}.\mathsf{Enc}$, respectively.

Let $\nu_3^{>}(\lambda)$ denote the probability that when $\nu_3(\lambda)$ is true, $v' > t$. Let $\nu_3^{<}(\lambda)$ be the case that $v' < t$, and $\nu_3^{=}(\lambda)$ be the case that $v' = t$. If $\nu_3(\lambda)$ is non-negligible, then at least one of $\nu_3^{>}(\lambda)$, $\nu_3^{<}(\lambda)$, $\nu_3^{=}(\lambda)$ must be non-negligible.

Suppose $\nu_3^{>}(\lambda)$ is non-negligible. In this case, let d be the index for which $v'[d] > t[d]$ and $\forall i < d$, $v'[i] = t[i]$. We can split this into further cases depending on the behavior of Dec. To do so, we define the following notation.

- Let $\nu_{3,i}^{>,\mathsf{M}}(\lambda)$ denote the probability that $\mathsf{PKE}_{\mathsf{M}}.\mathsf{Dec}(\Lambda_k^{\mathsf{M}}, msk_i^{t[i]+1}, ct_i^{\mathsf{M}}) \neq \mathsf{OTP.Enc}(\alpha_{i-1}, \alpha_i)$ for some $i \in [d-1]$.
- Let $\nu_{3,i}^{>,\mathsf{OM}}(\lambda)$ denote the probability that $\alpha_i' \neq \alpha_i$ for some $i \in [0, d-1]$
- Let $\nu_{3,i}^{>,\mathsf{OR}}(\lambda)$ denote the probability that $ct_i^{\mathsf{R}} \neq \mathsf{PKE}_{\mathsf{R}}.\mathsf{Enc}(\Lambda_k^{\mathsf{R}}, rpk_i^{v'[i]}, \alpha^*; \epsilon_i^{\mathsf{R}})$ for some $i \in [d]$.
- Let $\nu_{3,i,j}^{>,\mathsf{R}}(\lambda)$ denote the probability that $\mathsf{PKE}_{\mathsf{R}}.\mathsf{Dec}(\Lambda_k^{\mathsf{R}}, rsk_i^j, ct_i^{\mathsf{R}}) \neq \perp$ for some $i \in [d-1]$ and $j \in [t[i]+1, b]$, or for $i = d$ and $j \in [t[i]+1, v'[d]-1]$.
- Let $\nu_{3,d,v'[d]}^{>,*}(\lambda)$ denote the probability that $\mathsf{PKE}_{\mathsf{R}}.\mathsf{Dec}(\Lambda_k^{\mathsf{R}}, rsk_d^{v'[d]}, ct_d^{\mathsf{R}}) \neq \alpha^*$.

If $\nu_3^{>}(\lambda)$ is non-negligible, then at least one of $\nu_{3,i,j}^{>,\mathsf{R}}(\lambda)$, $\nu_{3,d,v'[d]}^{>,*}(\lambda)$ must be non-negligible, as by the perfect correctness of OTP' we must have $\alpha^{*\prime} = \mathsf{PKE}_{\mathsf{R}}.\mathsf{Dec}(\Lambda_k^{\mathsf{R}}, rsk_d^{v'[d]}, ct_d^{\mathsf{R}}) \neq \alpha^*$ for the output of Dec to not equal m.

Suppose that $\nu_{3,i}^{>,\mathsf{M}}(\lambda) > 0$ for some $i \in [n-1]$. Then, we can break the correctness property of $\mathsf{PKE}_{\mathsf{M}}$, as

$$\mathsf{PKE}_{\mathsf{M}}.\mathsf{Dec}(\Lambda_k^{\mathsf{M}}, sk, \mathsf{PKE}_{\mathsf{M}}.\mathsf{Enc}(\Lambda_k^{\mathsf{M}}, pk, m'; \epsilon_i^{\mathsf{M}})) \neq m'$$

Therefore, $\nu_{3,i}^{>,\mathsf{M}}(\lambda) = 0$ for all $i \in [d-1]$.

Next, it is clearly the case that $\nu_{3,0}^{>,\mathsf{OM}}(\lambda) = 0$, as α'_0 is set identically to how it is set in Escrow'. We now show that for all $i \in [n-1]$, if $\nu_{3,i-1}^{>,\mathsf{M}}(\lambda) = 0$ then $\nu_{3,i}^{>,\mathsf{M}}(\lambda) = 0$. As $\nu_{3,i-1}^{>,\mathsf{M}}(\lambda) = 0$, we have that $\beta_i^{\mathsf{M}} = \mathsf{OTP.Enc}(\alpha_{i-1}, \alpha_i)$. Then, by definition and our assumption we have that $\alpha'_i = \mathsf{OTP.Dec}(\alpha_{i-1}, \mathsf{OTP.Enc}(\alpha_{i-1}, \alpha_i))$. Therefore, if $\alpha'_i \neq \alpha_i$ we have broken the correctness property of OTP. Thus, by induction, we have that $\nu_{3,i}^{>,\mathsf{OM}}(\lambda) = 0$ for all $i \in [0, d-1]$.

Next, suppose that $\nu_{3,i}^{>,\mathsf{OR}}(\lambda) > 0$ for some i. This is equivalent to stating that $\mathsf{OTP.Dec}(\alpha'_{i-1}, \beta_i^{\mathsf{R}}) \neq \mathsf{PKE_R.Enc}(\Lambda_k^{\mathsf{R}}, rpk_i^{v'[i]}, \alpha^*; \epsilon_i^{\mathsf{R}})$. Then, by the definition of β_i^{R} and the fact that $\nu_{3,i-1}^{>,\mathsf{OM}}(\lambda) = 0$, we have:

$$\mathsf{OTP.Dec}(\alpha_{i-1}, \mathsf{OTP.Enc}(\alpha_{i-1}, \mathsf{PKE_R.Enc}(\Lambda_k^{\mathsf{R}}, rpk_i^{v'[i]}, \alpha^*; \epsilon_i^{\mathsf{R}}))) \neq \mathsf{PKE_R.Enc}(\Lambda_k^{\mathsf{R}}, rpk_i^{v'[i]}, \alpha^*; \epsilon_i^{\mathsf{R}}).$$

This breaks the correctness of OTP, and we therefore have that $\nu_{3,i}^{>,\mathsf{OR}}(\lambda) = 0$ for all $i \in [d]$.

Next, suppose that $\nu_{3,i,j}^{>,\mathsf{R}}(\lambda)$ is non-negligible for some (i, j). Then, we have that:

$$\begin{aligned}
\mathsf{PKE_R.Dec}(\Lambda_k^{\mathsf{R}}, rsk_i^j, ct_i^{\mathsf{R}}) &\neq \bot \\
\mathsf{PKE_R.Dec}(\Lambda_k^{\mathsf{R}}, rsk_i^j, \mathsf{PKE_R.Enc}(\Lambda_k^{\mathsf{R}}, rpk_i^{v'[i]}, \alpha^*)) &\neq \bot \qquad (\nu_{3,i}^{>,\mathsf{OR}}(\lambda) = 0) \\
\mathsf{PKE_R.Dec}(\Lambda_k^{\mathsf{R}}, rsk_i^j, \mathsf{PKE_R.Enc}(\Lambda_k^{\mathsf{R}}, rpk_i^{v'[i]}, \alpha^*; \epsilon_i^{\mathsf{R}})) &\neq \bot \qquad (\text{Definition of } \mathsf{Enc})
\end{aligned}$$

Then, we can write a reduction $\mathcal{B}$ that breaks the weak-robustness property of $\mathsf{PKE_R}$. The reduction outputs $(0, 1, \alpha^*, \epsilon_i^{\mathsf{R}})$ and wins as $v'[i] < j$. Therefore, $\nu_{3,i,j}^{>,\mathsf{R}}(\lambda)$ must be negligible for all $i \in [d-1]$ and $j \in [t[i]+1, b]$ or when $i = d$ and $j \in [t[i]+1, v'[d]-1]$.

Finally, suppose that $\nu_{3,d,v'[d]}^{>,*}(\lambda) > 0$. Then, we have that:

$$\begin{aligned}
\mathsf{PKE_R.Dec}(\Lambda_k^{\mathsf{R}}, rsk_i^{v'[i]}, ct_i^{\mathsf{R}}) &\neq \alpha^* \\
\mathsf{PKE_R.Dec}(\Lambda_k^{\mathsf{R}}, rsk_i^{v'[i]}, \mathsf{PKE_R.Enc}(\Lambda_k^{\mathsf{R}}, rpk_i^{v'[i]}, \alpha^*)) &\neq \alpha^* \qquad (\nu_{3,i}^{>,\mathsf{OR}}(\lambda) = 0)
\end{aligned}$$

This breaks the correctness of $\mathsf{PKE_R}$, and therefore we have that $\nu_{3,d,v'[d]}^{>,*}(\lambda) = 0$, and that $\nu_3^{>}(\lambda)$ is negligible.

Suppose $\nu_3^{<}(\lambda)$ is non-negligible. In this case, let d be the index for which $v'[d] < t[d]$ and $\forall i < d$, $v'[i] = t[i]$. Let $\nu_{3,i,j}^{<,\mathsf{R}}(\lambda)$ denote the probability that $\mathsf{PKE_R.Dec}(\Lambda_k^{\mathsf{R}}, rsk_i^j, ct_i^{\mathsf{R}}) \neq \bot$ for some $i \in [n]$ and $j \in [t[i]+1, b]$. If $\nu_3^{<}(\lambda)$ is non-negligible, then at least one of $\nu_{3,i,j}^{<,\mathsf{R}}(\lambda)$ must be non-negligible. By the same logic as above, we have that $\nu_{3,i,j}^{<,\mathsf{R}}(\lambda) = 0$ for all $i \in [d]$.

Now, suppose that $\nu^{<,\mathsf{R}}_{3,i,j}(\lambda)$ is non-negligible for some $i > d$. By definition we have that there is a non-negligible probability that $\mathsf{PKE_R.Dec}(rsk_i^j, ct_i^{\mathsf{R}}) \neq \perp$.

We then have:

$$\mathsf{PKE_R.Dec}(rsk_i^j, ct_i^{\mathsf{R}}) \neq \perp$$
$$\mathsf{PKE_R.Dec}(rsk_i^j, \mathsf{OTP.Dec}(\alpha'_{i-1}, \beta_i^{\mathsf{R}})) \neq \perp$$

$$\mathsf{PKE_R.Dec}(rsk_i^j, \mathsf{OTP.Dec}(\mathsf{OTP.RDec}(\alpha_{d-1}, \mathsf{PKE_M.Dec}(msk_d^{t[d]+1}, ct_d^{\mathsf{M}}), \ldots, \beta_{i-1}^{\mathsf{M}}), \beta_i^{\mathsf{R}})) \neq \perp$$

$$\mathsf{PKE_R.Dec}(rsk_i^j, \mathsf{OTP.Dec}(\mathsf{OTP.Dec}(g^{r'}, \mathsf{PKE_M.Dec}(msk_d^{t[d]+1}, ct_d^{\mathsf{M}})), \beta_i^{\mathsf{R}})) \neq \perp \quad \text{(Lemma 2)}$$

As ct_d^{M} was encrypted under a lossy key (as $v[d] < t[d]$), we can then break the lossy-correlation-resistance property of $(\mathsf{PKE_R}, \mathsf{PKE_M}, \mathsf{OTP})$ with a reduction that outputs

$$(\epsilon_d^{\mathsf{M}}, \epsilon_i^{\mathsf{R}}, r', \kappa_{i-1}^{\mathsf{O}}, \kappa_{d-1}^{\mathsf{O}} + \kappa_d^{\mathsf{O}}, \kappa_*^{\mathsf{O}})$$

Therefore, $\nu^{<,\mathsf{R}}_{3,i,j}(\lambda) \leq \mathsf{negl}(\lambda)$ for all $i \in [d+1, n]$ and for all $j \in [t[i]+1, b]$, and so $\nu_3^{<}(\lambda)$ must be negligible.

Finally, we have already shown that $\nu^{>,\mathsf{R}}_{3,i,j}(\lambda)$ must be negligible at all positions where $v'[i] = t[i]$. Therefore, $\nu_3^{=}(\lambda)$ must be negligible. In turn, $\nu_3(\lambda)$ and then $\nu(\lambda)$ must be negligible, and therefore the scheme satisfies soundness.

Lemma 4. *If Π_{KG} and Π_E are complete and* $\mathsf{PKE_R}$, $\mathsf{PKE_M}$, OTP, *and* OTP' *are correct, then (ℓ, b)-TCPPB satisfies correctness of* Dec.

Proof. We first consider the case that $v > t$. In this case, there is some index d for which $v[d] > t[d]$, and $v[i] = t[i] \;\; \forall i < d$. For $i < d$, as $v[i] = t[i]$, the match decryption $\beta_i^{\mathsf{M}} \leftarrow \mathsf{PKE_M.Dec}(\cdot)$ succeeds (as $mpk_i^{v[i]+1}$ must be injective), which in turn implies that the OTP key α'_i is computed correctly. Therefore, the ciphertext ct_d^{R} is properly formed. Finally, as $v[d] > t[d]$, the auditor will attempt to decrypt ct_d^{R} with $rsk_d^{v[d]}$, which will correctly return α^* and then m.

Next we turn to the case that $v < t$. In this case there is some index d for which $v[d] < t[d]$, and $v[i] = t[i] \;\; \forall i < d$. As $v[i] = t[i]$ for all $i < d$, $\mathsf{PKE_R.Dec}(\cdot)$ will output $\perp$ for each tried rsk_i^j by the weak robustness property. Finally, at index d the match decryption will *fail*, resulting in all future reveal ciphertexts being blinded, and therefore $\mathsf{PKE_R.Dec}(\cdot)$ outputting $\perp$ by the lossy correlation resistance of $(\mathsf{PKE_R}, \mathsf{PKE_M}, \mathsf{OTP})$. As all reveal decrypt attempts fail, Dec will therefore output $\perp$ as required.

Finally, we examine the case that $v = t$. In this case all reveal ciphertexts are encrypted under lossy keys, and therefore Dec will always output $\perp$ by the weak robustness property of $\mathsf{PKE_R}$.

Lemma 5. *If* PKE_R *and* PKE_M *are lossy encryption schemes satisfing indistinguishability of keys and* Π_{KG} *satisfies zero-knowledge, then* (ℓ, b)*-TCPPB satisfies threshold hiding.*

Proof. We prove security using a sequence of games. For a game Game_i, let $\mathcal{G}_i = \Pr[\mathsf{Game}_i(\mathsf{PPB}, \mathcal{A}, \lambda) \Rightarrow 1]$.

- Game_0: This game is identical to $\mathsf{ThrHide}_0$.
- Game_1: In this game the proof π_{KG} output by $\mathsf{KGen}(\cdot)$ is generated by the Π_{KG} simulator Sim. By the zero-knowledge property of Π_{KG}, $\mathcal{G}_0 \stackrel{\text{p}}{=} \mathcal{G}_1$.
- Game_2: In this game steps 1 through 14 of KGen' are replaced with the following:

```
1 :  for i ∈ [n]
2 :    for j ∈ [0, b − 1]
3 :      if j ≤ t0[i]
4 :        rpk_i^j ← PKE_R.KGen_loss(Λ_k^R, 1^λ)
5 :      else
6 :        rsk_i^j, rpk_i^j ← PKE_R.KGen(Λ_k^R, 1^λ)
7 :      if j ≤ t0[i]
8 :        mpk_i^j ← PKE_M.KGen_loss(Λ_k^M, 1^λ)
9 :      else
10 :       msk_i^j, mpk_i^j ← PKE_M.KGen(Λ_k^M, 1^λ)
11 :     p̄k_i^j ← (rpk_i^j, mpk_i^j)
12 :     s̄k_i^j ← ⊥
```

  As $\mathcal{A}$ does not have access to sk, this game behaves identically to Game_1 from the adversary's point of view, and so $\mathcal{G}_1 \stackrel{\text{p}}{=} \mathcal{G}_2$.
- Game_3: In this game step 3 of KGen' is changed from $j \leq t_0[i]$ to $j \leq t_1[i]$. We show that $\mathcal{G}_2 \stackrel{\text{c}}{\approx} \mathcal{G}_3$ via a hybrid argument. In hybrid $\mathcal{H}_{k,\ell}$ for $k \in [0, n]$ and $\ell \in [0, b-1]$, step 3 is changed as described at loop iteration (k, ℓ). Clearly $\mathcal{H}_{0,0}$ is equivalent to Game_2, and $\mathcal{H}_{n,b-1}$ is equivalent to Game_3.
  In the case that $(\ell \leq t_0[k]) = (\ell \leq t_1[k])$, the two hybrids are identical. In the case that $(\ell \leq t_0[k]) \neq (\ell \leq t_1[k])$, indistinguishability follows from the key indistinguishability property of PKE_R. We therefore have that $\mathcal{H}_{k,\ell} \stackrel{\text{c}}{\approx} \mathcal{H}_{k,\ell+1}$, and $\mathcal{H}_{k,b} \stackrel{\text{c}}{\approx} \mathcal{H}_{k+1,0}$ for all k, ℓ[7], and therefore that $\mathcal{G}_2 \stackrel{\text{c}}{\approx} \mathcal{G}_3$.
- Game_4: In this game step 7 of KGen' is changed from $j \leq t_0[i]$ to $j \leq t_1[i]$. $\mathcal{G}_3 \stackrel{\text{c}}{\approx} \mathcal{G}_4$ follows identically as before from the key indistinguishability property of PKE_M.
- Game_5 In this game KGen' is returned to its original formulation, with all comparisons performed against t_1. As $\mathcal{A}$ does not have access to sk, this game behaves identically to Game_4, and so $\mathcal{G}_4 \stackrel{\text{p}}{=} \mathcal{G}_5$.

[7] We abuse notation here and allow $\mathcal{H}_{k,\ell}$ to also denote $\Pr[\mathcal{H}_{k,\ell}(\mathsf{PPB}, \mathcal{A}, \lambda) \Rightarrow 1]$. We will continue to abuse notation in this way throughout the remainder of the paper.

– Game_6: This game is identical to $\mathsf{ThrHide}_1$. By the zero-knowledge property of Π_{KG} we have that $\mathcal{G}_5 \stackrel{\text{p}}{=} \mathcal{G}_6$.

We therefore conclude that $\mathcal{G}_0 \stackrel{\text{c}}{\approx} \mathcal{G}_6$, and thus (ℓ, b)-TCPPB satisfies threshold hiding.

Lemma 6. *If* PKE_R *is a lossy and weak-lossy-ambiguous public-key encryption scheme,* PKE_M *is a lossy and* $\mathcal{U}_{\mathsf{OTP}.\mathcal{O}}$*-strong-lossy-ambiguous public-key encryption scheme,* OTP *is an abelian OTP scheme,* OTP' *is a secure OTP scheme,* $(\mathsf{CSetup}, \mathsf{Commit})$ *satisfies statistical hiding and computational binding,* Π_E *satisfies zero-knowledge, and* Π_{KG} *satisfies strong simulation extractability, then* (ℓ, b)*-TCPPB satisfies escrow hiding.*

Proof. We start by describing our simulator Sim, and use use a sequence of games to show that:

$$\left\{\mathsf{Escrow}(\Lambda, pk, x, r)\right\} \stackrel{\text{c}}{\approx} \left\{\mathsf{Sim}(\Lambda, \mathsf{td}, pk, f(t, x))\right\}$$

Our simulator behaves as follows:

$\mathsf{Sim}(\Lambda, \mathsf{td}, pk, z = f(t, x))$

1 : **parse** $(\lambda, \Lambda_C, \Lambda_k^\mathsf{R}, \Lambda_k^\mathsf{M}, \mathsf{crs}_{KG}, \mathsf{crs}_E, n) = \Lambda$
2 : **if** $z = (m, v')$ **return** $\mathsf{Escrow}(\Lambda, pk, (v' || 0^{n-|v'|}, m))$
3 : **else if** $z = \perp$ **return** $\mathsf{Escrow}(\Lambda, pk, (0, \epsilon))$

We begin our analysis with the case that $z = \perp$. Additionally, as $z = \perp$, it is the case that $v \leq t$. Therefore, there either exists some index d such that $\forall i < d$, $v[i] = t[i]$, and $v[d] < t[d]$, or $v = t$. We first prove security in the case that $v < t$ via the following sequence of games. Let $\mathcal{G}_i = \Pr\left[\mathsf{Game}_i(\mathsf{PPB}, \mathcal{A}, \lambda) \Rightarrow 1\right]$.

– Game_0: The original escrow hiding game when $b = 0$, i.e. $\mathsf{Escrow}(\Lambda, pk, x, r)$.
– Game_1: The proof π_E is simulated using the trapdoor td_E. By the zero-knowledge property of Π_E, $\mathcal{G}_0 \stackrel{\text{p}}{=} \mathcal{G}_1$.
– Game_2: Replace C_E with $\mathsf{Commit}(\Lambda_C, (0, \epsilon))$. By the statistical hiding property of the commitment scheme scheme, $\mathcal{G}_1 \stackrel{\text{s}}{\approx} \mathcal{G}_2$.
– Game_3: Replace β_d^M with an element sampled uniformly at random from $\mathsf{OTP}.\mathcal{O}$. To show that $\mathcal{G}_2 \stackrel{\text{c}}{\approx} \mathcal{G}_3$ we need to show that for any PPT adversary $\mathcal{A}$, $|\mathcal{G}_2 - \mathcal{G}_3| = \nu(\lambda)$ is negligible.
  The adversary $\mathcal{A}$ distinguishing between the two games outputs $(t, r_{KG}, pk = (pk', C_{KG}, \pi_{KG}), x)$ in its first stage. We can divide our analysis into the following three cases depending on the properties of these outputs.
  1. There exist some r'_{KG} and $t' \neq t$ such that $\mathsf{Commit}(t, r_{KG}) = \mathsf{Commit}(t', r'_{KG})$ and $(\cdot, pk') = \mathsf{KGen}'(\lambda, \Lambda_k^\mathsf{R}, \Lambda_k^\mathsf{M}, n, t')$.
  2. $(\cdot, pk') \neq \mathsf{KGen}'(\lambda, \Lambda_k^\mathsf{R}, \Lambda_k^\mathsf{M}, n, t)$ and (1) does not hold.
  3. Neither (1) nor (2) holds.

Let the probability of the outputs satisfying property j be denoted by $\nu_j(\lambda)$. If $\nu(\lambda)$ is non-negligible, then at least one of $\nu_1(\lambda)$, $\nu_2(\lambda)$, $\nu_3(\lambda)$ must be non-negligible.

Suppose that $\nu_2(\lambda)$ is non-negligible. In this case $\mathcal{A}$ produced a proof of a false statement and we can construct a reduction $\mathcal{B}$ to the simulation-extractability of Π_{KG}. $\mathcal{B}$ runs $\mathcal{A}$, receives $(t, r_{KG}, pk = (pk', C_{KG}, \pi_{KG}), x)$, and outputs $((pk', C_{KG}), \pi_{KG})$. As the statement is false the extractor must fail, meaning $\Pr[\mathcal{B} \text{ wins}]$ is non-negligible, thus contradicting our assumption that Π_{KG} satisfies strong simulation-extractability. Therefore, $\nu_2(\lambda)$ is negligible.

As $\nu_2(\lambda)$ is negligible, we can assume that there exists an extractor Ext that given π_{KG} outputs (t', sk', r_k, r_c) such that $C_{KG} = \mathsf{Commit}(\Lambda_C, t; r_c)$ and $(sk', pk') = \mathsf{KGen}'(\lambda, \Lambda_k^{\mathsf{R}}, \Lambda_k^{\mathsf{M}}, b, n, t; r_k)$.

Suppose $\nu_1(\lambda)$ is non-negligible. In this case we can break the computational binding property of the commitment scheme by using a reduction that uses the extractor and outputs (t, r_{KG}, t', r_c). Therefore, $\nu_1(\lambda)$ must be negligible.

Suppose $\nu_3(\lambda)$ is non-negligible. In this case we can construct a reduction $\mathcal{B}$ to the computational lossiness of $\mathsf{PKE}_{\mathsf{M}}$. $\mathcal{B}$ runs $\mathcal{A}$, receives $(t, r_{KG}, pk = (pk', C_{KG}, \pi_{KG}), x)$, and uses Ext to extract (t, sk', r_k, r_c). $\mathcal{B}$ then uses r_k to infer r such that $(\cdot, mpk_d^{v[d]}) = \mathsf{PKE}_{\mathsf{M}}.\mathsf{KGen}_{\mathsf{loss}}(\Lambda_k^{\mathsf{M}}; r)$ (the key must be lossy as $v[d] < t[d]$), samples β' uniformly at random from $\mathsf{OTP}.\mathcal{O}$, and sends $(r, \beta_d^{\mathsf{M}}, \beta')$ to its challenger. $\mathcal{B}$ receives C, sends to $\mathcal{A}$ an escrow value identical to the one produced in Game_3 with the exception that ct_d^{M} is replaced by C, and outputs what $\mathcal{A}$ outputs.
In the case that $\mathcal{B}$ is in $\mathsf{CompLoss}_0$, $\mathcal{A}$'s view is identical to game Game_2, and in $\mathsf{CompLoss}_1$ $\mathcal{A}$'s view is identical to Game_3. We therefore conclude that $\mathcal{B}$'s advantage is equal to $\nu_3(\lambda)$, and so $\nu_3(\lambda)$ must be negligible.

As $\nu_1(\lambda)$, $\nu_2(\lambda)$, and $\nu_3(\lambda)$ are all negligible, $\nu(\lambda)$ must be negligible. Therefore, we have that $\mathcal{G}_2 \overset{c}{\approx} \mathcal{G}_3$.
Identical logic showing that $\nu_0(\lambda)$ and $\nu_1(\lambda)$ must be negligible will be implicitly used in future arguments that depend upon properties of pk'. For brevity we will assume the existence of an extractor outputting (t, sk', r_k, r_c) and focus only on $\nu_3(\lambda)$ without repeating the full proof.

– Game_4: In this game β_i^{M} is replaced by an element sampled uniformly at random from $\mathsf{OTP}.\mathcal{O}$ for all $i \in [d+1, n-1]$. We show that $\mathcal{G}_3 \overset{p}{=} \mathcal{G}_4$ via a hybrid argument. In hybrid $\mathcal{H}_k$ for $k \in [d+1, n-1]$, replace β_k^{M} as described. Clearly $\mathcal{H}_d$ is equivalent to Game_3 and $\mathcal{H}_{n-1}$ is equivalent to Game_4. As the output contains no information regarding α_{k-1} (as β_{k-1}^{M} has been replaced with a random element in the previous hybrid), $\mathcal{H}_k \overset{c}{\approx} \mathcal{H}_{k+1}$ for all k by the perfect security of OTP, and so $\mathcal{G}_3 \overset{p}{=} \mathcal{G}_4$.

– Game_5: In this game β_i^{M} is replaced by an element sampled uniformly at random from $\mathsf{OTP}.\mathcal{O}$ for all $i \in [d-1, 1]$. We show that $\mathcal{G}_4 \overset{\mathrm{p}}{=} \mathcal{G}_5$ via a hybrid argument. In hybrid $\mathcal{H}_k$ for $k \in [d-1, 1]$, replace β_k^{M} as described.
In $\mathcal{H}_k$ we replace $\mathsf{OTP}.\mathsf{Enc}(\alpha_{k-1}, \alpha_k)$ with a random element. Note that $\mathsf{OTP}.\mathsf{Enc}(\alpha_{k-1}, \alpha_k) = \mathsf{OTP}.\mathsf{Enc}(\alpha_k, \alpha_{k-1})$ as OTP is abelian. Then, as the output contains no information regarding α_k (as β_{k+1}^{M} has been replaced with a random element in the previous hybrid, or in Game_3 for $k = d-1$), $\mathcal{H}_k \overset{\mathrm{p}}{=} \mathcal{H}_{k-1}$ for all k by the perfect security of OTP.

Note that at this point all all values β_i^{M} have been replaced with uniformly random elements of $\mathsf{OTP}.\mathcal{O}$.

– Game_6: In this game ct_i^{M} is replaced by $\mathsf{PKE}_{\mathsf{M}}.\mathsf{Enc}(\Lambda_k^{\mathsf{M}}, mpk_i^0, \beta_i^{\mathsf{M}})$ for all $i \in [d-1]$. We show that $\mathcal{G}_5 \overset{\mathrm{c}}{\approx} \mathcal{G}_6$ via a hybrid argument. In hybrid $\mathcal{H}_k$ for $k \in [d-1]$, replace ct_k^{M} as described. In the case that $t[k] = v[k] = 0$, $\mathcal{H}_k$ is identical to the previous one. In the case that $t[k] = v[k] > 0$, note that $mpk_i^{v[i]}$ is injective and mpk_i^0 is lossy. Skipping the logic gone over in Game_3, we will now show that in this case $\nu_3(\lambda)$ is negligible. To do so, we will construct a reduction $\mathcal{B}$ to the strong lossy ambiguity property of $\mathsf{PKE}_{\mathsf{M}}$.

$\mathcal{B}$ runs $\mathcal{A}$, receives $(t, r_{KG}, pk = (pk', C_{KG}, \pi_{KG}), x)$, and uses Ext to extract (t, sk', r_k, r_c). $\mathcal{B}$ then uses r_k to infer r_0, r_v such that $(\cdot, mpk_k^0) = \mathsf{PKE}_{\mathsf{M}}.\mathsf{KGen}_{\mathsf{loss}}(\Lambda_k^{\mathsf{M}}, 1^\lambda; r_0)$ and $(\cdot, mpk_k^{v[i]}) = \mathsf{PKE}_{\mathsf{M}}.\mathsf{KGen}(\Lambda_k^{\mathsf{M}}, 1^\lambda; r_v)$, and sends (r_v, r_0) to its challenger. $\mathcal{B}$ receives C^*, sends to $\mathcal{A}$ an escrow value identical to the one produced in $\mathcal{H}_k$ with the exception that ct_k^{M} is replaced by C^*, and outputs what $\mathcal{A}$ outputs. In the case that $\mathcal{B}$ is in $\mathcal{D}\text{-}\mathsf{SLAMB}_0$, $\mathcal{A}$'s view is identical to hybrid $\mathcal{H}_{k-1}$, and in $\mathcal{D}\text{-}\mathsf{SLAMB}_1$ $\mathcal{A}$'s view is identical to hybrid $\mathcal{H}_k$. We therefore conclude that $\mathcal{B}$'s advantage is equal to $\nu_3(\lambda)$, and so $\nu_3(\lambda)$ must be negligible. Therefore, we have that $\mathcal{H}_k \overset{\mathrm{c}}{\approx} \mathcal{H}_{k+1}$ for all k, and so $\mathcal{G}_5 \overset{\mathrm{c}}{\approx} \mathcal{G}_6$.

Future transitions that rely on the weak/strong lossy ambiguity of $\mathsf{PKE}_{\mathsf{M}}/\mathsf{PKE}_{\mathsf{R}}$ will proceed in the same way, and so we will omit the full proofs for brevity.

– Game_7: In this game we replace ct_d^{M} with $\mathsf{PKE}_{\mathsf{M}}.\mathsf{Enc}(\Lambda_k^{\mathsf{M}}, mpk_d^0, \beta_d^{\mathsf{M}})$. In the case that $v[d] = 0$, this game is identical to Game_6^d. Otherwise, as $v[d] < t[d]$, we have that $mpk_d^{v[d]+1}$ is lossy. As such, by the weak lossy ambiguity property of $\mathsf{PKE}_{\mathsf{M}}$, $\mathcal{G}_6 \overset{\mathrm{c}}{\approx} \mathcal{G}_7$.
– Game_8: In this game we replace ct_i^{M} with $\mathsf{PKE}_{\mathsf{M}}.\mathsf{Enc}(\Lambda_k, mpk_i^0, \beta_i^{\mathsf{M}r})$ for all $i \in [d+1, n-1]$. We show that $\mathcal{G}_7 \overset{\mathrm{c}}{\approx} \mathcal{G}_8$ via a hybrid argument. In hybrid $\mathcal{H}_k$ for $k \in [d+1, n-1]$, replace ct_k^{M} as described.
In the case that $v[k] = 0$, the hybrid is identical to the previous hybrid. Otherwise, we can split the possibilities into several cases.
  1. When $v[k] \geq t[k] > 0$, we have that $mpk_k^{v[k]+1}$ is injective while mpk_k^0 is lossy. Indistinguishability therefore follows from the strong lossy ambiguity property of $\mathsf{PKE}_{\mathsf{M}}$.

2. When $t[k] > v[k] > 0$ we have that $mpk_k^{v[k]+1}$ and mpk_k^0 are both lossy. Indistinguishability therefore follows from the weak lossy ambiguity property of $\mathsf{PKE_M}$.

We therefore have that $\mathcal{G}_7 \overset{c}{\approx} \mathcal{G}_8$ by the strong lossy ambiguity of $\mathsf{PKE_M}$.

Note that at this point ct_i^{M} is encrypted under mpk_i^0 for all i.

- Game_9: Replace ct_i^{R} with $\mathsf{PKE_R.Enc}(\Lambda_k^{\mathsf{R}}, rpk_i^0, \alpha^*)$ for all $i \in [d]$. We show that $\mathcal{G}_8 \overset{c}{\approx} \mathcal{G}_9$ via a hybrid argument. In hybrid $\mathcal{H}_k$ for $k \in [d]$, replace ct_k^{R} as described. In the case that $v[k] = 0$ the hybrid is identical to the previous hybrid. Otherwise, as $t[k] \geq v[k]$ we have that $rpk_k^{v[k]}$ is lossy, and so $\mathcal{H}_k \overset{c}{\approx} \mathcal{H}_{k+1}$ for all k by the weak lossy ambiguity of $\mathsf{PKE_R}$.
- Game_{10}: Replace β_i^{R} with an element sampled uniformly at random from $\mathsf{OTP}.\mathcal{O}$ for all $i \in [d+1, n]$. We can show that $\mathcal{G}_9 \overset{p}{=} \mathcal{G}_{10}$ via a hybrid argument, where at hybrid $\mathcal{H}_k$ we replace β_k^{R} as described. As the output contains no information regarding α_{k-1} (as β_{k-1}^{M} has been replaced with a random element), $\mathcal{H}_k \overset{p}{=} \mathcal{H}_{k+1}$ by the perfect security of OTP. Therefore $\mathcal{G}_9 \overset{p}{=} \mathcal{G}_{10}$.
- Game_{11}: Replace ct_i^{R} with $\mathsf{PKE_R.Enc}(\Lambda_k^{\mathsf{R}}, rpk_i^0, \epsilon)$ for all $i \in [d+1, n]$. As the output contains no information regarding ct_i^{R} (as β_i^{R} was previously replaced with a random element), we have that $\mathcal{G}_{10} \overset{p}{=} \mathcal{G}_{11}$.
- Game_{12}: Replace ct_i^{R} with $\mathsf{PKE_R.Enc}(\Lambda_k^{\mathsf{R}}, rpk_i^0, \epsilon)$ for all $i \in [d]$. We can show that $\mathcal{G}_{11} \overset{c}{\approx} \mathcal{G}_{12}$ via a hybrid argument, where at hybrid $\mathcal{H}_k$ we replace ct_k^{R} as described. As rpk_k^0 is always lossy, and $rpk_k^{v[k]}$ is lossy by our assumption that $v \leq t$, by the computational lossiness property of $\mathsf{PKE_R}$ we have that $\mathcal{H}_k \overset{c}{\approx} \mathcal{H}_{k+1}$ for all k, and so $\mathcal{G}_{11} \overset{c}{\approx} \mathcal{G}_{12}$.

Note that at this point ct_i^{R} encrypts ϵ under rpk_i^0 for all i.

- Game_{13}: Replace β^* with $\mathsf{OTP'.Enc}(\alpha^*, \epsilon)$. As the output contains no information about α^* (as all ct_i^{R} now encrypt ϵ), we have that $\mathcal{G}_{12} \overset{p}{=} \mathcal{G}_{13}$ by the perfect security of $\mathsf{OTP'}$.
- Game_{14}: This game is identical to the functioning of the simulator. We have that $\mathcal{G}_{13} \overset{c}{\approx} \mathcal{G}_{14}$ via a series of hybrids that reverts the changes made in games so far as necessary. Indistinguishability follows from the same justifications as above.

We therefore conclude that

$$\left\{\mathsf{Escrow}(\Lambda, pk, (v, m), r)\right\} \overset{c}{\approx} \left\{\mathsf{Sim}(\Lambda, \mathsf{td}, pk, f(t, (v, m))\right\}$$

in the case that $v < t$.

We now turn to the case that $v = t$, and consider the following sequence of games.

- Game_0: The original escrow hiding game when $b = 0$, i.e. $\mathsf{Escrow}(\Lambda, pk, x, r)$.
- Game_1: The proof π_E is simulated using the trapdoor td_E. By the zero-knowledge property of Π_E, $\mathcal{G}_0 \overset{p}{=} \mathcal{G}_1$.

- Game_2: Replace C_E with $\mathsf{Commit}(\Lambda_C, (0, \epsilon))$. By the statistical hiding property of the commitment scheme scheme, $\mathcal{G}_1 \overset{\text{s}}{\approx} \mathcal{G}_2$.
- Game_3: Replace ct_i^{R} with $\mathsf{PKE_R.Enc}(\Lambda_k^{\mathsf{R}}, rpk_i^0, \alpha^*)$ for all $i \in [n]$. As $rpk_i^{v[i]}$ is lossy for all i (as $v = t$), we have that $\mathcal{G}_2 \overset{\text{c}}{\approx} \mathcal{G}_3$ via a hybrid argument and the weak-lossy-ambiguity of $\mathsf{PKE_R}$.
- Game_4: Replace ct_i^{R} with $\mathsf{PKE_R.Enc}(\Lambda_k^{\mathsf{R}}, rpk_i^0, \epsilon)$ for all $i \in n$. As rpk_i^0 is lossy for all i, we have that $\mathcal{G}_3 \overset{\text{c}}{\approx} \mathcal{G}_4$ via a hybrid argument and the computational lossiness of $\mathsf{PKE_R}$.
- Game_5: Replace β^* with $\mathsf{OTP'.Enc}(\alpha^*, \epsilon)$. As the output contains no information about α^*, we have that $\mathcal{G}_4 \overset{\text{p}}{=} \mathcal{G}_5$ by the perfect security of $\mathsf{OTP'}$.
- Game_6: Replace ct_i^{M} with $\mathsf{PKE_M.Enc}(\Lambda_k^{\mathsf{M}}, mpk_i^0, \beta_i^{\mathsf{M}})$ for all $i \in [n]$. As $mpk_i^{v[i]+1}$ is injective for all i (as $v = t$), we have that $\mathcal{G}_5 \overset{\text{c}}{\approx} \mathcal{G}_6$ via a hybrid argument and the strong-lossy-ambiguity of $\mathsf{PKE_M}$.
- Game_7: The simulator honestly generates the proof π_E. By the zero-knowledge property of Π_E, $\mathcal{G}_6 \overset{\text{p}}{=} \mathcal{G}_7$.

Note that $\mathcal{G}_7$ is identical to the functioning of the simulator.

Finally, we consider the case that $v > t$. If $v[i] \geq t[i]$ for all $i \in [n]$ then $v' = v$, and so the simulation is clearly perfect. Otherwise, there must be some digit d for which $v[d] < t$. We can then use an identical sequence of games as in the case that $v < t$ to show that the output of the simulator is indistinguishable to the output of Escrow.

We therefore conclude that

$$\left\{\mathsf{Escrow}(\Lambda, pk, (v, m), r)\right\} \overset{\text{c}}{\approx} \left\{\mathsf{Sim}(\Lambda, \mathsf{td}, pk, f(t, (v, m))\right\}$$

for $v < t$, $v = t$, and $v > t$, and therefore (ℓ, b)-TCPPB satisfies escrow hiding.

Theorem 1 then follows directly from Lemma 3, Lemma 4, Lemma 5, and Lemma 6.

D Encryption Schemes

We give here constructions and proofs related to the encryption schemes used in this work. Our lossy encryption scheme can be seen in Figure 4. The transformation from a weak collision free scheme to a weak robust scheme can be seen in Figure 5. The one-time-pad scheme can be seen in Figure 6.

D.1 Proofs for $\mathsf{PKE_M}$

Theorem 4. *If the Decisional-Diffie-Hellman assumption holds in $\mathbb{G}$, then the scheme shown in Figure 4 satisfies computational lossiness.*

Proof. We prove security using a sequence of games. For a game Game_i, let $\mathcal{G}_i = \Pr\left[\mathsf{Game}_i(\mathsf{PKE}, \mathcal{A}, \lambda) \Rightarrow 1\right]$.

$\mathsf{Setup}(1^\lambda)$

1 : **return** $\mathcal{H}(1^\lambda)$

$\mathsf{KGen}(\Lambda_k)$

1 : $(\mathbb{G}, p, g, h) \leftarrow \Lambda_k$
2 : $x \leftarrow\!\!\$\ \mathbb{Z}_p$
3 : $X \leftarrow g^x$
4 : $pk \leftarrow (p, g, X)$
5 : $sk \leftarrow (p, g, x)$
6 : **return** (sk, pk)

$\mathsf{Enc}(\Lambda_k, pk, M)$

1 : $(\mathbb{G}, p, g, h) \leftarrow \Lambda_k$
2 : $y \leftarrow\!\!\$\ \mathbb{Z}_p$
3 : $Y \leftarrow g^y$
4 : $T \leftarrow X^y$
5 : $W \leftarrow TM$
6 : **return** (Y, W)

$\mathsf{Dec}(\Lambda_k, sk, ct)$

1 : $(\mathbb{G}, p, g, h) \leftarrow \Lambda_k$
2 : $(Y, W) \leftarrow ct$
3 : $T \leftarrow Y^x$
4 : $M \leftarrow WT^{-1}$
5 : **return** M

$\mathsf{KGen}_{\mathsf{loss}}(\Lambda_k)$

1 : $(\mathbb{G}, p, g, h) \leftarrow \Lambda_k$
2 : $r \leftarrow\!\!\$\ \{0,1\}^\lambda$
3 : $X \leftarrow h^r$
4 : $pk \leftarrow (p, g, X)$
5 : **return** pk

Fig. 4: A lossy variant of ElGamal encryption. It uses a second generator h to produce lossy keys.

$\mathsf{KGen}'(1^\lambda, \phi)$

1 : $(sk_0, pk_0) \leftarrow \mathsf{KGen}(1^\lambda)$
2 : $(sk_1, pk_1) \leftarrow \mathsf{KGen}(1^\lambda)$
3 : $\overline{sk} = (sk_0, sk_1)$
4 : $\overline{pk} = (pk_0, pk_1)$
5 : **return** $(\overline{sk}, \overline{pk})$

$\mathsf{KGen}'_{\mathsf{loss}}(1^\lambda, \phi)$

1 : $pk_0 \leftarrow \mathsf{KGen}_{\mathsf{loss}}(1^\lambda)$
2 : $pk_1 \leftarrow \mathsf{KGen}_{\mathsf{loss}}(1^\lambda)$
3 : $\overline{pk} = (pk_0, pk_1)$
4 : **return** $\overline{pk}$

$\mathsf{Enc}'(pk, m, \phi)$

1 : $(pk_0, pk_1) \leftarrow pk$
2 : $ct_0 \leftarrow \mathsf{Enc}(pk_0, m)$
3 : $ct_1 \leftarrow \mathsf{Enc}(pk_1, \phi)$
4 : **return** (ct_0, ct_1)

$\mathsf{Dec}'(sk, ct, \phi)$

1 : $(sk_0, sk_1) \leftarrow sk$
2 : $(ct_0, ct_1) \leftarrow ct$
3 : **if** $\mathsf{Dec}(sk_1, ct_1) \neq \phi$
4 : **return** $\perp$
5 : **else**
6 : **return** $\mathsf{Dec}(sk_0, ct_0)$

Fig. 5: Our transform from a collision-free $\mathsf{PKE} = (\mathsf{Setup}, \mathsf{KGen}, \mathsf{KGen}_{\mathsf{loss}}, \mathsf{Enc}, \mathsf{Dec})$ to a weakly-robust one.

$\mathsf{OTP.KGen}()$	$\mathsf{OTP.Enc}(\alpha, m)$	$\mathsf{OTP.Dec}(\alpha, \beta)$
$r \leftarrow\!\!\$\ \mathbb{Z}_p$	**return** $\alpha \cdot m$	**return** β/α
return g^r		

Fig. 6: Our instantiated OTP scheme over $\mathcal{O} = \mathbb{G}$, a group with prime order p and generator g.

- Game_0: This game is identical to $\mathsf{CompLoss}_0$.
- Game_1: In this game $C = (g^y, (h^{ry} m_0))$ is replaced with $(g^y, (h^{rz} m_0))$ for $z \leftarrow\!\!\$\ \mathbb{Z}_p$. We then have that $\mathcal{G}_0 \overset{\text{c}}{\approx} \mathcal{G}_1$, as otherwise we could build a reduction $\mathcal{B}$ to DDH as follows. $\mathcal{B}$ receives (g, h, g^y, c) from the challenger, and (r, m_0, m_1) from $\mathcal{A}$. $\mathcal{B}$ sends $(g^y, c^r m_0)$ to $\mathcal{A}$, and outputs what $\mathcal{A}$ outputs. In the case that $c = h^y$, $\mathcal{A}$'s view is exactly as it is in Game_0, while when $c = h^z$, $\mathcal{A}$'s view is exactly as it is in Game_1.
- Game_2: In this game C is replaced with $(g^y, (h^{rz} m_1))$. As z is a uniformly random element of $\mathbb{Z}_p$ the message is fully masked, and so $\mathcal{G}_1 \overset{\text{p}}{=} \mathcal{G}_2$.
- Game_3: In this game C is replaced with $(g^y, (h^{ry} m_1))$. We have that $\mathcal{G}_2 \overset{\text{c}}{\approx} \mathcal{G}_3$ from DDH as before. Note that Game_3 is identical to $\mathsf{CompLoss}_1$.

We therefore have that $\mathcal{G}_0 \overset{\text{c}}{\approx} \mathcal{G}_3$, and so the scheme satisfies computational lossiness.

Theorem 5. *The scheme shown in Figure 4 satisfies $\mathcal{U}_{\mathbb{G}}$-strong-lossy-ambiguity.*

Proof. We prove security using a sequence of games. For a game Game_i, let $\mathcal{G}_i = \Pr\left[\mathsf{Game}_i(\mathsf{PKE}, \mathcal{A}, \lambda) \Rightarrow 1\right]$.

- Game_0: This game is identical to $\mathcal{D}$-SLAMB_0.
- Game_1: In this game $C = (g^y, pk_0^y \cdot m)$ is replaced with $C = (g^y, pk_1^y \cdot m)$. As m is sampled uniformly at random from $\mathbb{G}$, we have that $\mathcal{G}_0 \overset{\text{p}}{=} \mathcal{G}_1$.

Then, as Game_1 is identical to $\mathcal{D}$-SLAMB_1, we have that the scheme satisfies $\mathcal{U}_{\mathbb{G}}$-strong-lossy-ambiguity.

D.2 Proofs for $\mathsf{PKE_R}$

Theorem 6. *If* PKE *satisfies collision-freeness, then the scheme* PKE' *shown in Figure 5 satisfies weak-robustness.*

Proof. Assume there exists an adversary $\mathcal{A}$ such that $\mathsf{Adv}^{\mathsf{Rob}}_{\mathsf{PKE}',\mathcal{A}}(\lambda)$ is non-negligible. We can then construct a reduction $\mathcal{B}$ to the collision-freeness of PKE as follows.

$\mathcal{B}$ receives $\{pk_i\}_{i\in[n]}$, computes $\{pk'_i\}_{i\in[n]} \leftarrow \mathsf{KGen}(1^\lambda)$, runs $\mathcal{A}(1^\lambda, \{(pk'_i, pk_i)\}_{i\in[n]})$, receives $(j, k, m, r = (r_0, r_1))$, and outputs (j, k, ϕ, r_1). As $\mathcal{A}$'s advantage is non-negligible, it must be the case that $\mathsf{Dec}(sk_k, \mathsf{Enc}(pk_j, \phi; r'_1)) = \phi$ with non-negligible probability. Therefore $\mathcal{B}$'s advantage is non-negligible, which is a contradiction, and so the scheme satisfies weak-robustness.

Theorem 7. *If* PKE *satisfies weak-lossy-ambiguity, then the scheme* PKE' *shown in Figure 5 satisfies weak-lossy-ambiguity.*

Proof. We prove security using a sequence of games. For a game Game_i, let $\mathcal{G}_i = \Pr\left[\mathsf{Game}_i(\mathsf{PKE}', \mathcal{A}, \lambda) \Rightarrow 1\right]$.

- Game_0: This game is identical to WLAMB_0.
- Game_1: In this game $ct_0 \leftarrow \mathsf{Enc}(pk_{0,0}, m)$ is replaced with $\mathsf{Enc}(pk_{1,0}, m)$. We then have that $\mathcal{G}_0 \overset{c}{\approx} \mathcal{G}_1$ as otherwise we could build a reduction $\mathcal{B}$ to the weak-lossy-ambiguity of PKE as follows. $\mathcal{B}$ receives $(r_0 = (r_{0,0}, r_{0,1}), r_1 = (r_{1,0}, r_{1,1}), m)$ from $\mathcal{A}$, and sends $(r_{0,0}, r_{1,0}, m)$ to the challenger, receiving C^*. $\mathcal{B}$ then sends $(C^*, \mathsf{Enc}(\mathsf{KGen}_{\mathsf{loss}}(; r_{0,1}), \phi))$ to $\mathcal{A}$, and outputs whatever $\mathcal{A}$ outputs.
- Game_2: In this game ct_1 is replaced by $\mathsf{Enc}(pk_{1,1}, \phi)$. We again have that $\mathcal{G}_1 \overset{c}{\approx} \mathcal{G}_2$ by the weak-lossy-ambiguity of PKE.

Then, as Game_2 is identical to WLAMB_1, we have that $\mathcal{G}_0 \overset{c}{\approx} \mathcal{G}_2$ and PKE' satisfies weak-lossy-ambiguity.

E Baseline PPB

Before describing the baseline PPB instantiation, we first recall the interfaces of oblivious transfer and garbling schemes.

E.1 Oblivious Transfer

A one-out-of-two Oblivious Transfer (OT) protocol is a tuple of algorithms $\mathsf{OT} = (\mathsf{Init}, \mathsf{Send}, \mathsf{Receive})$ defined as:

- $\mathsf{Init}(1^\lambda, b)$ takes as input a bit $b \in \{0,1\}$ and outputs a secret key/public key pair (sk, pk).
- $\mathsf{Send}(pk, x_0, x_1)$ takes as input a public key and two values and outputs a ciphertext ct.
- $\mathsf{Receive}(sk, ct)$ takes as input a secret key and a ciphertext and outputs a value x'.

Informally, we require that the receiver gets x_b while learning nothing about x_{b-1}. We refer the reader to other texts discussing oblivious transfer in more depth such as [15] for a complete definition.

E.2 Garbling Schemes

Definition 11. *A garbling scheme is a tuple of algorithms* $\mathsf{GC} = (\mathsf{GCircuit}, \mathsf{GInput}, \mathsf{Eval})$ *defined as:*

- $\mathsf{GCircuit}(1^\lambda, C)$ *takes as input a circuit* $C : \{0,1\}^n \to \{0,1\}^m$ *and outputs the garbled circuit* $\tilde{C}$ *and key* k.

- $\mathsf{GInput}(k, x)$ *takes as input a key* k *and string* $x \in \{0,1\}^n$ *and outputs a garbled input* $\tilde{x}$.
- $\mathsf{Eval}(\tilde{C}, \tilde{x})$ *takes as input a garbled circuit and input, and outputs* $y \in \{0,1\}^m$.

We say that a garbling scheme is projective *if each bit of the garbled input* $\tilde{x}$ *depends on one bit of the actual input* x*, i.e. each bit of the input is garbled independently of the other bits. We will let* $\mathsf{GInput}'(k, x_i)$ *denote the algorithm to garble a single bit of input, i.e.* $\mathsf{GInput}(k, x) = \{\mathsf{GInput}'(k, i, x[i])\}_{i \in [n]}$ *for* $x \in \{0,1\}^n$.

As with OT, we refer the reader to [5] for a full description of the properties required of garbling schemes.

E.3 Construction

Define the following two relations:

$$\mathcal{R}_{KG}^{\lambda,\Lambda_C,n} = \left\{ ((pk', C), (t, sk', r_k, r_c)) \ : \ \begin{array}{c} (sk', pk') = \mathsf{KGen}'(\lambda, t; r_k) \\ \wedge\, C = \mathsf{Commit}(\Lambda_C, t; r_c) \end{array} \right\}$$

$$\mathcal{R}_{E}^{\lambda,\Lambda_C,n} = \left\{ ((Z', C, pk'), (v, m, r_e, r_c)) \ : \ \begin{array}{c} Z' = \mathsf{Escrow}'(\lambda, pk', (v, m); r_e) \\ \wedge\, C = \mathsf{Commit}(\Lambda_C, (v, m); r_c) \end{array} \right\}$$

Let $\mathcal{L}_{KG}^{\lambda,\Lambda_C}$ and $\mathcal{L}_{E}^{\lambda,\Lambda_C}$ be the corresponding languages to these relations, and $\Pi_{KG}^{\lambda,\Lambda_C}$, $\Pi_{E}^{\lambda,\Lambda_C}$ be corresponding NIZKs for these languages. We may drop the superscripts when the values are clear from context. The generic NISC-based PPB construction instantiated with garbled circuits and OT can be seen in Figure 7 and Figure 8.

F Examples

We provide here some examples of our construction's output and computation in order to provide intuition. All examples use $b = 10$ and $\ell = 9999$, and use $\alpha \circ x$ as shorthand for $\mathsf{OTP.Enc}(\alpha, x)$ and $(\alpha^{-1}) \circ x$ as shorthand for $\mathsf{OTP.Dec}(\alpha, x)$. Table 3 represents the output of KGen in table form. Figure 9 gives an example of the output of Escrow and the Dec computation for a value under the auditor's threshold, while Figure 10 gives an example for a value above the threshold.

$\mathsf{Setup}(1^\lambda, \Lambda_C)$

1 : $(\mathsf{crs}_{KG}, \mathsf{td}_{KG}) \leftarrow \mathsf{S}_{KG}(1^\lambda)$
2 : $(\mathsf{crs}_E, \mathsf{td}_E) \leftarrow \mathsf{S}_E(1^\lambda)$
3 : $\Lambda = (\lambda, \Lambda_C, \mathsf{crs}_{KG}, \mathsf{crs}_E)$
4 : **return** $(\Lambda, (\mathsf{td}_{KG}, \mathsf{td}_E))$

$\mathsf{KGen}'(\lambda, x)$

1 : $n := |x|$
2 : **for** $i \in [n]$ **do**
3 : $\quad (sk_i, pk_i) \leftarrow \mathsf{Init}(1^\lambda, x[i])$
4 : $sk' \leftarrow \{sk_i\}_{i\in[n]}$
5 : $pk' \leftarrow \{pk_i\}_{i\in[n]}$
6 : **return** (sk', pk')

$\mathsf{KGen}(\Lambda, x, r_{KG})$

1 : **parse** $(\lambda, \Lambda_C, \mathsf{crs}_{KG}, \mathsf{crs}_E) = \Lambda$
2 : $(sk', pk') \xleftarrow{r_k} \mathsf{KGen}'(\lambda, x)$
3 : $C_{KG} \leftarrow \mathsf{Commit}(\Lambda_C, x; r_{KG})$
4 : $\pi_{KG} \leftarrow \mathsf{P}_{KG}(\mathsf{crs}_{KG}, (pk', C), (t, sk', r_k, r_c))$
5 : $pk \leftarrow (pk', C_{KG}, \pi_{KG})$
6 : $sk \leftarrow (sk', pk)$
7 : **return** (sk, pk)

$\mathsf{VerPK}(\Lambda, pk, C_{KG})$

1 : **parse** $(\lambda, \Lambda_C, \mathsf{crs}_{KG}, \mathsf{crs}_E) = \Lambda$
2 : **parse** $(pk', C'_{KG}, \pi_{KG}) = pk$
3 : **return** $\mathsf{V}_{KG}(\mathsf{crs}_{KG}, \pi_{KG}, (pk', C_{KG}))$
4 : $\quad \wedge (C'_{KG} = C_{KG})$

$\mathsf{VerEscrow}(\Lambda, pk, C_E, Z = (Z', \pi_E))$

1 : **parse** $(\lambda, \Lambda_C, \mathsf{crs}_{KG}, \mathsf{crs}_E) = \Lambda$
2 : **parse** $(pk', C'_{KG}, \pi_{KG}) = pk$
3 : **return** $\mathsf{VerPK}(\Lambda, pk, C_{KG})$
4 : $\quad \mathsf{V}_E((Z', C_E, pk'), \pi_E)$

Fig. 7: A generic NISC-based PPB construction

$\mathsf{Escrow}'(\lambda, pk', v)$

1 : **parse** $pk' = \{pk_i\}_{i\in[n]}$
2 : $(k, \tilde{C}) \leftarrow \mathsf{GCircuit}(1^\lambda, C)$
3 : **for** $i \in [n]$ **do**
4 : $\quad ct_i \leftarrow \mathsf{Send}(pk_i, \mathsf{GInput}'(k, i, 0), \mathsf{GInput}'(k, i, 1))$
5 : **for** $i \in [|v|]$ **do**
6 : $\quad \tilde{x}_{n+i} \leftarrow \mathsf{GInput}'(k, n+i, v[i])$
7 : $ct' := \{ct_1, \ldots, ct_n\}$
8 : $\tilde{x}' := \{\tilde{x}_{n+1}, \ldots, \tilde{x}_{n+|v|}\}$
9 : **return** $(\tilde{C}, ct', \tilde{x}')$

$\mathsf{Escrow}(\Lambda, pk, v, r_E)$

1 : **parse** $(\lambda, \Lambda_C, \mathsf{crs}_{KG}, \mathsf{crs}_E) = \Lambda$
2 : **parse** $(pk', C_{KG}, \pi_{KG}) = pk$
3 : $Z' \xleftarrow{r_e} \mathsf{Escrow}'(\lambda, pk', v)$
4 : $C_E \leftarrow \mathsf{Commit}(\Lambda_C, v; r_E)$
5 : $\pi_E \leftarrow \mathsf{P}_E(\mathsf{crs}_E, (Z', C_E, pk'), (v))$
6 : **return** (Z', π_E)

$\mathsf{Dec}(\Lambda, sk, C_E, Z = (Z', \pi_E))$

1 : **parse** $(\lambda, \Lambda_C, \mathsf{crs}_{KG}, \mathsf{crs}_E) = \Lambda$
2 : **parse** $(\{sk_i\}_{i\in[n]}, pk) = sk$
3 : **if** $\mathsf{VerEscrow}(\Lambda, pk, C_E, Z) = 0$ **then return** $\perp$
4 : **parse** $(\tilde{C}, \{ct_1, \ldots, ct_n\}, \{\tilde{x}_{n+1}, \ldots, \tilde{x}_{n+|v|}\}) = Z'$
5 : **for** $i \in [n]$ **do**
6 : $\quad \tilde{x}_i \leftarrow \mathsf{Receive}(sk_i, ct_i)$
7 : $\tilde{x} := \{\tilde{x}_i\}_{i\in[n+|v|]}$
8 : **return** $\mathsf{Eval}(\tilde{C}, \tilde{x})$

Fig. 8: A generic NISC-based PPB construction, continued.

k_1^0	k_1^1	k_1^2	k_1^3	k_1^4	k_1^5	k_1^6	k_1^7	k_1^8	k_1^9	k_1^{10}
k_2^0	k_2^1	k_2^2	k_2^3	k_2^4	k_2^5	k_2^6	k_2^7	k_2^8	k_2^9	k_2^{10}
k_3^0	k_3^1	k_3^2	k_3^3	k_3^4	k_3^5	k_3^6	k_3^7	k_3^8	k_3^9	k_3^{10}
k_4^0	k_4^1	k_4^2	k_4^3	k_4^4	k_4^5	k_4^6	k_4^7	k_4^8	k_4^9	k_4^{10}

Table 3: Example KGen output for $b = 10$, $\ell = 9999$, and $t = 1486$. Here, $k_i^j = ((\perp, rpk_j^i), (\perp, mpk_j^i))$ (lossy) and $k_i^j = ((rsk_j^i, rpk_j^i), (msk_j^i, mpk_j^i))$ (injective).

$\beta_1^{\mathsf{R}} = \varepsilon \circ \mathsf{PKE_R.Enc}(rpk_1^1, m)$	$ct_1^{\mathsf{M}} = \mathsf{PKE_M.Enc}(mpk_1^2, \varepsilon \circ \alpha_1)$
$\beta_2^{\mathsf{R}} = \alpha_1 \circ \mathsf{PKE_R.Enc}(rpk_2^4, m)$	$ct_2^{\mathsf{M}} = \mathsf{PKE_M.Enc}(mpk_2^5, \alpha_2 \circ \alpha_1)$
$\beta_3^{\mathsf{R}} = \alpha_2 \circ \mathsf{PKE_R.Enc}(rpk_3^2, m)$	$ct_3^{\mathsf{M}} = \mathsf{PKE_M.Enc}(mpk_3^3, \alpha_3 \circ \alpha_2)$
$\beta_4^{\mathsf{R}} = \alpha_3 \circ \mathsf{PKE_R.Enc}(rpk_4^8, m)$	$\perp$

(a) Example Escrow for $v = 1427$.

$\perp = \mathsf{PKE_R.Dec}(rsk_1^j, \epsilon^{-1} \circ \beta_1^{\mathsf{R}})$	$\alpha_1' = \epsilon^{-1} \circ \mathsf{PKE_M.Dec}(msk_1^2, ct_1^{\mathsf{M}}) = \alpha_1$
$\perp = \mathsf{PKE_R.Dec}(rsk_2^j, (\alpha_1')^{-1} \circ \beta_2^{\mathsf{R}})$	$\alpha_2' = (\alpha_1')^{-1} \circ \mathsf{PKE_M.Dec}(msk_2^5, ct_2^{\mathsf{M}}) = \alpha_2$
$\perp = \mathsf{PKE_R.Dec}(rsk_3^j, (\alpha_2')^{-1} \circ \beta_3^{\mathsf{R}})$	$\alpha_3' = (\alpha_2')^{-1} \circ \mathsf{PKE_M.Dec}(msk_3^9, ct_3^{\mathsf{M}}) \neq \alpha_3$
$\perp = \mathsf{PKE_R.Dec}(rsk_4^j, (\alpha_3')^{-1} \circ \beta_4^{\mathsf{R}})$	

(b) Example failed Dec for $v = 1427 < t = 1486$. Red indicates a failed decryption, while Blue indicates a successful one. Note that the decryptor cannot tell whether a decrypted α_i' is correct.

Fig. 9: An example of an Escrow and failed Dec for a value $v < t$.

$\beta_1^{\mathsf{R}} = \varepsilon \circ \mathsf{PKE_R.Enc}(rpk_1^1, m)$	$ct_1^{\mathsf{M}} = \mathsf{PKE_M.Enc}(mpk_1^2, \varepsilon \circ \alpha_1)$
$\beta_2^{\mathsf{R}} = \alpha_1 \circ \mathsf{PKE_R.Enc}(rpk_2^7, m)$	$ct_2^{\mathsf{M}} = \mathsf{PKE_M.Enc}(mpk_2^8, \alpha_2 \circ \alpha_1)$
$\beta_3^{\mathsf{R}} = \alpha_2 \circ \mathsf{PKE_R.Enc}(rpk_3^0, m)$	$ct_3^{\mathsf{M}} = \mathsf{PKE_M.Enc}(mpk_3^1, \alpha_3 \circ \alpha_2)$
$\beta_4^{\mathsf{R}} = \alpha_3 \circ \mathsf{PKE_R.Enc}(rpk_4^2, m)$	$\perp$

(a) Example Escrow for $v = 1702$.

$\perp = \mathsf{PKE_R.Dec}(rsk_1^j, \epsilon^{-1} \circ \beta_1^{\mathsf{R}})$	$\alpha_1' = \epsilon^{-1} \circ \mathsf{PKE_M.Dec}(msk_1^2, ct_1^{\mathsf{M}}) = \alpha_1$
$\alpha^* = \mathsf{PKE_R.Dec}(rsk_2^7, (\alpha_1')^{-1} \circ \beta_2^{\mathsf{R}})$ $m = \alpha^* \circ \beta^*$	

(b) Example successful Dec for $v = 1702 > t = 1486$. Red indicates a failed decryption, while Blue indicates a successful one.

Fig. 10: An example of an Escrow and successful Dec for a value $v > t$.