# Homomorphic Encryption with CCA Security[*]

Manoj Prabhakaran[†] Mike Rosulek[†]

May 24, 2008

**Abstract**

We address the problem of constructing public-key encryption schemes that meaningfully combine useful *computability features* with *non-malleability.* In particular, we investigate schemes in which anyone can change an encryption of an unknown message $m$ into an encryption of $T(m)$ (as a *feature*), for a specific set of allowed functions $T$, but the scheme is "non-malleable" with respect to all other operations. We formulate precise definitions that capture these intuitive requirements and also show relationships among our new definitions and other more standard ones (IND-CCA, gCCA, and RCCA). We further justify our definitions by showing their equivalence to a natural formulation of security in the Universally Composable framework. We also consider extending the definitions to features which combine *multiple* ciphertexts, and show that a natural definition is unattainable for a useful class of features. Finally, we describe a new family of encryption schemes that satisfy our definitions for a wide variety of allowed transformations $T$, and which are secure under the standard Decisional Diffie-Hellman (DDH) assumption.

[*] An extended abstract of this work appears in *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008*, Springer-Verlag, 2008. This full version is available from `http://eprint.iacr.org/2008/079`.

[†] Department of Computer Science, University of Illinois, Urbana-Champaign. {`mmp,rosulek`}`@uiuc.edu`. Partially supported by NSF grant CNS 07-47027.

# Contents

# 1 Introduction

A recurring theme in cryptography is the tension between achieving powerful functionality and making strong security guarantees. In the case of encryption, IND-CCA security is well-accepted as a sufficiently strong security guarantee. On the other hand, for encryption to be useful in sophisticated applications (such as voting or mix-nets), the scheme should have features which allow *computation* on encrypted messages (e.g., features like rerandomizability [20, 21], proxy re-encryption [6, 9], searchability [35, 11] and different kinds of homomorphism properties [18, 26]). However, IND-CCA security rules out *any* such feature which operates on encrypted messages, while the other extreme of IND-CPA security does not exclude the possibility that a scheme may have additional "unforeseen features" that an adversary can exploit when the scheme is used in a larger application. Is it possible to express (and achieve via a construction) a security requirement capturing *the best of both worlds*: to be malleable enough to allow rich features, but non-malleable enough to rule out "everything else?"

In this work we address this question in the context of *homomorphic* public-key encryption schemes — those which allow anyone to change encryptions of unknown messages $m_1, \ldots, m_k$ into an encryption of $T(m_1, \ldots, m_k)$, for some allowed set of functions $T$. Such schemes have been extensively studied for a long time and have a wide variety of applications (cf. [5, 32, 12, 33, 22, 24, 15, 20, 23, 14]). Homomorphic encryption schemes have additional utility in that ciphertexts hide not only the underlying plaintext, but also the way in which the ciphertext was derived (i.e., as a regular encryption, or via some homomorphic operation applied to some other ciphertexts). We explicitly formalize this requirement, which we call *unlinkability.*

**Challenges and Related Work.** The first challenge is formally defining (in a convincing way) the intuitive requirement that a scheme "allow particular features but forbid all others." Security notions for regular encryption developed and matured over many years [19, 25, 31, 4, 17, 7], while arguably security definitions for homomorphic encryptions have lagged behind — to date, homomorphic encryptions are almost exclusively held to the weak standard of IND-CPA security. In some applications (e.g., [14]) CPA security is indeed sufficient, but for others (e.g. [16]) it is not. Very little work has addressed the possibility of homomorphic encryption schemes having "unforeseen features" beyond the prescribed operations; one exception is Wikström [36], who addresses this question in a simpler setting for El Gamal.

Benignly-malleable (gCCA) security [34, 1] was proposed as a relaxation of CCA security, and was further relaxed in the definition of Replayable-CCA (RCCA) security [10]. RCCA security allows a scheme to have homomorphic operations which preserve the underlying plaintext, but enforces non-malleability "everywhere else." However, relaxing CCA security in the same way does not yield an acceptable level of security when applied to more expressive homomorphic operations (see Section 3.1); a new approach to defining security is needed.

The second challenge is achieving the desired security with a construction based on standard assumptions — i.e., an encryption scheme that has a particular set of (unlinkable) homomorphic operations, but is non-malleable with respect to all other operations. Note that even if the set of allowed operations is very simple, supporting it can be very involved. Indeed, the problem of unlinkable (rerandomizable) RCCA encryption considered in a recent series of works [10, 21, 29] corresponds to arguably the simplest special case of our definitions.

**Our Results.** We give several new security definitions to precisely capture the desired requirements in the case of *unary* homomorphic operations (those which transform a single encryption of $m$ to an encryption of $T(m)$, for a particular set of functions $T$). We provide two new indistinguishability-based security definitions: one formalizing the unlinkability requirement and one formalizing the intuition of "non-malleability except for certain prescribed operations." To justify this last security definition, we show that it subsumes the standard IND-CCA, gCCA, and RCCA security definitions (Theorem 1). We further show that our two new security requirements imply a natural definition of security in the Universal Composition framework (Theorem 2). Using the UC framework to define security of encryption schemes was already considered in [7, 10, 27, 8].

We also consider extending our definitions to the case of *binary* homomorphic operations (those which combine pairs of ciphertexts). We show that the natural generalization of our UC security definition to this scenario is unachievable for a large class of useful homomorphic operations (Theorem 5).

Finally, we describe a family of encryption schemes which achieves our definitions for a wide range of allowed (unary) homomorphism operations. The construction, which is a careful generalization of the rerandomizable RCCA-secure scheme of [29], is secure under the standard DDH assumption, and supports the group operation as a homomorphic feature (as well as several related operations).

## 2 Homomorphic Encryption Preliminaries

$$\begin{tikzcd}
m \arrow[r, shift left, "\mathsf{Enc}"] \arrow[d, "T"'] & \zeta \arrow[l, shift left, "\mathsf{Dec}"] \arrow[d, "\mathsf{CTrans}(\cdot, T)"] \\
m' & \zeta' \arrow[l, "\mathsf{Dec}"]
\end{tikzcd}$$

Figure 1: Illustration of the syntax and correctness of a homomorphic encryption scheme.

We call a function $\nu$ *negligible in $k$* if it asymptotically approaches zero faster than any inverse polynomial in $k$ — i.e., $\nu(k) = k^{-\omega(1)}$. A probability is *overwhelming* if it is $1 - \nu(k)$ for a negligible $\nu$ ($k$ being the security parameter). In all the encryption schemes we consider, the security parameter is the number of bits needed to represent an element from the underlying cyclic group.

Let $\mathcal{M}$ be a space of plaintext messages, let $\perp$ be a special error indicator symbol not in $\mathcal{M}$, and let $\mathcal{T}$ be a "transformation space" — i.e., a set of polynomial-time computable functions from $(\mathcal{M} \cup \{\perp\})^k$ to $\mathcal{M} \cup \{\perp\}$. We call the elements of $\mathcal{T}$ the *allowable transformations.*

An encryption scheme consists of three polynomial-time (polynomial in the implicit security parameter) algorithms: KeyGen, Enc and Dec.

A *$\mathcal{T}$-homomorphic encryption scheme* comes with an additional algorithm CTrans, the homomorphic operation feature: a randomized algorithm which takes $k$ ciphertexts and (the description of) a *transformation* from $\mathcal{T}$, and outputs another ciphertext.[1]

We mostly restrict attention to the case where $k = 1$; i.e., when the homomorphic operation is *unary.*

**Correctness Properties.** Below we give the correctness properties for unary homomorphic encryption. These requirements can be slightly relaxed (e.g., to hold only with overwhelming proba-

[1] Allowing CTrans to take the public key as additional input would also be a meaningful relaxation, but may not be suitable in some applications. See Section 7.1.

bility over key generation), but our construction achieves these simpler requirements.

For all key pairs $(PK, SK)$ in the support of $\mathsf{KeyGen}$, we require the following:

1. For every plaintext $\mathsf{msg} \in \mathcal{M}$, we require $\mathsf{Dec}_{SK}(\mathsf{Enc}_{PK}(\mathsf{msg})) = \mathsf{msg}$, with probability 1 over the randomness of $\mathsf{Enc}$.
2. For every purported ciphertext $\zeta$ and every $T \in \mathcal{T}$, we require $\mathsf{Dec}_{SK}(\mathsf{CTrans}(\zeta, T)) = T(\mathsf{Dec}_{SK}(\zeta))$, with probability 1 over the randomness of $\mathsf{CTrans}$.

The following property is desirable and achieved by our construction, though our security definitions (Definition 3) require only a weaker condition:

**Definition 1.** *A scheme is* perfectly rerandomizable *if for all messages* $\mathsf{msg} \in \mathcal{M}$*, all ciphertexts* $\zeta$ *in the support of* $\mathsf{Enc}_{PK}(\mathsf{msg})$*, and all* $T \in \mathcal{T}$*, the distribution of* $\mathsf{CTrans}(\zeta, T)$ *is identical to that of* $\mathsf{Enc}_{PK}(T(\mathsf{msg}))$*.*

### 2.1 Decisional Diffie-Hellman (DDH) Assumption

Let $\mathbb{G}$ be a (multiplicative) cyclic group of prime order $p$. The *Decisional Diffie-Hellman (DDH) assumption in* $\mathbb{G}$ is that the following two distributions are computationally indistinguishable:

$$\{(g, g^a, g^b, g^{ab})\}_{g \leftarrow \mathbb{G}; a,b \leftarrow \mathbb{Z}_p} \quad \text{and} \quad \{(g, g^a, g^b, g^c)\}_{g \leftarrow \mathbb{G}; a,b,c \leftarrow \mathbb{Z}_p}.$$

Here, $x \leftarrow X$ denotes that $x$ is drawn uniformly at random from a set $X$.

## 3 Defining Security

In this section we present our formal security definitions. The first two are traditional indistinguishability-based definitions, while the third is a definition in the Universal Composition framework.

### 3.1 Homomorphic-CCA (HCCA) Security

Our first indistinguishability-based security definition formalizes the intuitive notions of message privacy and "non-malleability other than certain operations."

Existing non-malleability definitions such as IND-CCA, benignly-malleable (a.k.a. gCCA) security [34, 1] and Replayable CCA (RCCA) security [10] share a similar structure, in which an experimenter encrypts one of two adversarially chosen plaintexts and provides a decryption oracle to the adversary, whose task it is to guess which plaintext was encrypted. Since the adversary could simply ask to decrypt the challenge ciphertext itself, the decryption oracles are guarded to not decrypt ciphertexts which may be "derivatives" of the challenge ciphertext. In CCA security, the only derivative is the challenge ciphertext itself; in gCCA, derivatives are those which satisfy a particular binary relation with the challenge ciphertext; in RCCA, derivatives are those which decrypt to either of the two adversarially-chosen plaintexts.

However, in the case of more general homomorphic encryption, it may be legal (i.e., possible via a feature of the scheme) to change the underlying plaintext of a ciphertext to any other possible plaintext. Indeed, in some instantiations of our construction, *every ciphertext* in the support of the $\mathsf{Enc}$ operation is a possible derivative of every other such ciphertext. Following the IND-CCA paradigm here would weaken it essentially to IND-CCA1 (i.e., "lunchtime attack") security.

Our approach to identifying "derivative" ciphertexts is completely different, and as a result our definition initially appears incomparable to these other standard definitions. However, Theorem 1 demonstrates that our new definition gives a generic notion of non-malleability which subsumes these existing definitions.

The formal definition, which we call *Homomorphic-CCA (HCCA) security*, appears below. Informally, in the security experiment we identify derivative ciphertexts not for regular encryptions, but for special "rigged" ciphertexts that carry no message. In other words, there should be a procedure $\mathsf{RigEnc}_{PK}$ which outputs a rigged ciphertext $\zeta$ and some auxiliary information $S$, such that $\zeta$ is indistinguishable from a normal ciphertext. There should also be a corresponding procedure $\mathsf{RigExtract}_{SK}$ which, when given another ciphertext $\zeta'$ and auxiliary information $S$, determines whether $\zeta'$ was obtained by applying a transformation to $\zeta$, and if so, outputs that transformation.

Intuitively, the transformations output by $\mathsf{RigExtract}$ constitute all the ways a ciphertext's message can depend on another ciphertext in the scheme, so restricting the range of $\mathsf{RigExtract}$ restricts the malleability of the scheme.

**Definition 2.** *A homomorphic encryption scheme is* Homomorphic-CCA (HCCA) secure *with respect to $\mathcal{T}$ if there are PPT algorithms* $\mathsf{RigEnc}$ *and* $\mathsf{RigExtract}$*, where the range of* $\mathsf{RigExtract}$ *is $\mathcal{T} \cup \{\bot\}$, and such that for all PPT adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the following IND-HCCA experiment is negligible:*

1. **Setup:** *Pick $(PK, SK) \leftarrow \mathsf{KeyGen}$ and give $PK$ to $\mathcal{A}$.*

2. **Phase I:** *$\mathcal{A}$ gets access to the $\mathsf{Dec}_{SK}(\cdot)$ oracle and the following two "guarded"* $\mathsf{RigEnc}$ *and* $\mathsf{RigExtract}$ *oracles:*

$$\begin{aligned} \mathsf{GRigEnc}_{PK}() &= \zeta_i, \;\; \textit{where } (\zeta_i, S_i) \leftarrow \mathsf{RigEnc}_{PK}, \;\; \textit{when called for the ith time} \\ \mathsf{GRigExtract}_{SK}(\zeta, i) &= \mathsf{RigExtract}_{SK}(\zeta, S_i) \end{aligned}$$

3. **Challenge:** *$\mathcal{A}$ outputs a plaintext $\mathsf{msg}^*$. We privately flip a coin $b \leftarrow \{0,1\}$. If $b = 0$, we compute $\zeta^* \leftarrow \mathsf{Enc}_{PK}(\mathsf{msg}^*)$. If $b = 1$, we compute $(\zeta^*, S^*) \leftarrow \mathsf{RigEnc}_{PK}$. In both cases, we give $\zeta^*$ to $\mathcal{A}$.*

4. **Phase II:** *$\mathcal{A}$ gets access to the same* $\mathsf{GRigEnc}$ *and* $\mathsf{GRigExtract}$ *oracles as in Phase I, as well as a "rigged" version of the decryption oracle* $\mathsf{RigDec}$*. When $b = 0$,* $\mathsf{RigDec}$ *is simply the normal decryption oracle $\mathsf{Dec}_{SK}(\cdot)$. When $b = 1$,* $\mathsf{RigDec}$ *is implemented as follows:*

$$\mathsf{RigDec}_{SK}(\zeta) = \begin{cases} T(\mathsf{msg}^*) & \textit{if } \bot \neq T \leftarrow \mathsf{RigExtract}_{SK}(\zeta, S^*) \\ \mathsf{Dec}_{SK}(\zeta) & \textit{otherwise} \end{cases}.$$

5. **Output:** *$\mathcal{A}$ outputs a bit $b'$. The* advantage *of $\mathcal{A}$ in this experiment is* $\Pr[b' = b] - \frac{1}{2}$.

We immediately observe that in order to achieve HCCA security, $\mathcal{T}$ must be closed under composition (or at least approximately so). $\mathcal{T}$ must also contain the identity function, since the adversary can simply submit the challenge ciphertext $\zeta^*$ to the $\mathsf{RigDec}$ oracle.

## 3.2 Unlinkability

There indeed is some tension between the HCCA definition given above and the intuitive notion of unlinkability that we desire. HCCA security implies that it is possible to track transformations applied to rigged ciphertexts, while unlinkability demands that ciphertexts not leak whether they were generated via a transformation. To reconcile this, we require unlinkability only on *ciphertexts that succesfully decrypt under a private key chosen by the challenger.* This excludes linkability via the $\mathsf{RigEnc}$ and $\mathsf{RigExtract}$ procedures, since tracking ciphertexts using $\mathsf{RigExtract}$ in general requires the tracking party to know the private key.

Our formal definition of unlinkability is given below. We highlight that the adversary has access to a decryption oracle in the experiment, making it meaningful for modeling chosen-ciphertext attacks.

**Definition 3.** *A homomorphic encryption scheme is* unlinkably homomorphic *with respect to $\mathcal{T}$ if for all PPT adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the following experiment is negligible:*

1. **Setup:** *Pick $(PK, SK) \leftarrow \mathsf{KeyGen}$ and give $PK$ to $\mathcal{A}$.*
2. **Phase I:** *$\mathcal{A}$ is given access to the decryption oracle $\mathsf{Dec}_{SK}(\cdot)$.*
3. **Challenge:** *Flip a coin $b \leftarrow \{0,1\}$. Receive from $\mathcal{A}$ a ciphertext $\zeta$ and a transformation $T \in \mathcal{T}$. If $\mathsf{Dec}_{SK}(\zeta) = \bot$, do nothing. Else give $\zeta^*$ to $\mathcal{A}$ where*

$$\zeta^* \leftarrow \begin{cases} \mathsf{Enc}_{PK}(T(\mathsf{Dec}_{SK}(\zeta))) & \textit{if } b = 0 \\ \mathsf{CTrans}(\zeta, T) & \textit{if } b = 1 \end{cases}$$

4. **Phase II:** *$\mathcal{A}$ is given access to the decryption oracle $\mathsf{Dec}_{SK}(\cdot)$.*
5. **Output:** *$\mathcal{A}$ outputs a bit $b'$. The* advantage *of $\mathcal{A}$ in this experiment is* $\Pr[b' = b] - \frac{1}{2}$.

Note that unlinkability is a security guarantee (involving maliciously crafted ciphertexts) and is not implied by the perfect rerandomization property (Definition 1), which only involves honestly generated ciphertexts.

We have defined unlinkability with the goal that a scheme can be both $\mathcal{T}$-unlinkably homomorphic, and $\mathcal{T}$-HCCA-secure (for the same $\mathcal{T}$). Indeed, it is easy to see that if a scheme is $\mathcal{T}$-unlinkably homomorphic and $\mathcal{T}'$-HCCA-secure, then $\mathcal{T} \subseteq \mathcal{T}'$. For simplicity, and to highlight the compatibility and sharp tradeoff between these two definitions, we only focus on schemes which satisfy them both with respect to the same transformation space $\mathcal{T}$.

## 3.3 UC Definition: Homomorphic Message Posting

We also define the "Homomorphic Message Posting" functionality $\mathcal{F}_{\text{HMP}}^{\mathcal{T}}$ in the framework of Universally Composable security [7, 28] as a natural security definition encompassing both unlinkability and our desired notion of non-malleability. The complete definition appears in Figure 2.

$\mathcal{F}_{\text{HMP}}^{\mathcal{T}}$ allows parties to post private messages for other parties, as on a bulletin board, represented by abstract *handles* which reveal no information about the message (they are generated by the adversary without knowledge of the message). Only the designated receiver is allowed to obtain the corresponding message for a handle. To model the homomorphic features, the functionality

**Setup:** On receiving a command $\textsc{setup}$ from a party $P$: If a previous $\textsc{setup}$ command has been processed, abort. Else, send $(\textsc{id-req}, P)$ to the adversary, and expect in response a string $\mathsf{id}$. Broadcast $(\textsc{id-announce}, P, \mathsf{id})$ to all other parties.

**Message posting:** On receiving a command $(\textsc{post}, \mathsf{msg})$ from a party $\mathsf{sender}$: If $\mathsf{msg} \notin \mathcal{M}$, ignore the request. If $P$ is corrupt, send $(\textsc{handle-req}, \mathsf{sender}, \mathsf{msg})$ to the adversary; otherwise send $(\textsc{handle-req}, \mathsf{sender})$ to the adversary. In both cases expect a string $\mathsf{handle}$ in return. If $\mathsf{handle}$ has been previously used, abort; else internally record $(\mathsf{handle}, \mathsf{msg})$ and broadcast $(\textsc{handle-announce}, \mathsf{handle})$ to all parties.

**Dummy handles:** On receiving a command $(\textsc{dummy}, \mathsf{handle})$ from a *corrupt* party, internally record $(\mathsf{handle}, \perp)$ and broadcast $(\textsc{handle-announce}, \mathsf{handle})$ to all parties.

**Homomorphic reposting:** On receiving a command $(\textsc{repost}, \mathsf{handle}, T)$ from a party $\mathsf{sender}$: If $\mathsf{handle}$ is not recorded internally or $T \notin \mathcal{T}$, ignore the request. Otherwise, suppose $(\mathsf{handle}, \mathsf{msg})$ is recorded internally. If $\mathsf{msg} \neq \perp$, then do the same as if $(\textsc{post}, T(\mathsf{msg}))$ were received. Otherwise, send $(\textsc{handle-req}, \mathsf{sender}, \mathsf{handle}, T)$ to the adversary and expect a string $\mathsf{handle}'$ in return. If $\mathsf{handle}'$ has been previously used, abort; else record $(\mathsf{handle}', T(\mathsf{msg}))$ internally and send $(\textsc{handle-announce}, \mathsf{handle}')$ to all parties.

**Message reading:** On receiving a command $(\textsc{get}, \mathsf{handle})$ from party $P$ (and only party $P$): If a record $(\mathsf{handle}, \mathsf{msg})$ is recorded internally, give $\mathsf{msg}$ to $P$; otherwise ignore this request.

Figure 2: UC ideal functionality $\mathcal{F}_{\text{HMP}}^{\mathcal{T}}$.

allows parties to post messages derived from other handles. The functionality is parameterized by the set of allowed transformations $\mathcal{T}$. When a party provides a previously posted handle and a transformation $T \in \mathcal{T}$, the functionality retrieves the message $m$ corresponding to the handle and then acts as if the party had actually requested $T(m)$ to be posted. The sender does not need to know, nor is it told, the underlying message $m$ of the existing handle.

$\mathcal{F}_{\text{HMP}}^{\mathcal{T}}$ models the non-malleability we require, since the *only* way a posted message can influence a subsequent message is via an allowed transformation.

The functionality also models unlinkability by internally behaving identically (in particular, in its interaction with the adversary) for the two different kinds of posts. The only exception is that corrupt parties may generate "dummy" handles which look like normal handles but do not contain any message. When a party derives a new handle from such a dummy handle, the adversary learns the transformation. This apparent slight weakness is natural[2] and it mirrors the tradeoff between our indistinguishability definitions. In our security proofs, this additional dummy handle feature is crucial.

**Homomorphic Encryption Schemes and Protocols for $\mathcal{F}_{\text{HMP}}^{\mathcal{T}}$.** The UC framework defines when a protocol is said to *securely realize* the functionality $\mathcal{F}_{\text{HMP}}^{\mathcal{T}}$: for every PPT adversary in the real world interaction (using the protocol), there exists a PPT simulator in the ideal world interaction

[2] For example, an adversary may broadcast a single encryption under a public key that he keeps hidden. The ciphertext will be meaningless to the recipient, but if the adversary later encounters another ciphertext that decrypts under this same key, he can deduce that the it was derived from his previous ciphertext.

with $\mathcal{F}^{\mathcal{T}}_{\text{HMP}}$, such that no PPT environment can distinguish between the two interactions.

We associate homomorphic encryption schemes with candidate protocols for $\mathcal{F}^{\mathcal{T}}_{\text{HMP}}$ in the following natural way (for simplicity assume all communication is on an authenticated broadcast channel). To setup an instance of $\mathcal{F}^{\mathcal{T}}_{\text{HMP}}$, a party generates a key pair and broadcasts the public key. To post a message, a party encrypts it under the public key and broadcasts the resulting ciphertext. The "derived post" feature is implemented via the $\mathsf{CTrans}$ procedure. To retrieve a message from a handle, the receiver decrypts it using the private key.

# 4 Relationships Among Security Definitions

To justify our new security definitions, we prove some relationships among them and among the more established definitions of IND-CCA, gCCA [1, 34], and RCCA [10] security.

## 4.1 HCCA Generalizes CCA, gCCA, RCCA

**Theorem 1.** *CCA, gCCA, and RCCA security can be obtained as special cases of the HCCA definition, by appropriately restricting* $\mathsf{RigEnc}$ *and* $\mathsf{RigExtract}$*.*

*Proof.* The restrictions on $\mathsf{RigExtract}$ are progressively relaxed as we go from CCA to gCCA to RCCA, making it explicit that the non-malleability requirements get weaker in that order.

First, consider a variant of the traditional IND-CCA definition, in which the adversary provides only one of the two challenge plaintexts $m_1$, while the other challenge plaintext $m_0$ is fixed and publicly known. This variant realizes the same level of security as the original IND-CCA definition (in which the adversary chooses both plaintexts).[3] We can further modify the experiment cosmetically so that when the adversary submits the challenge ciphertext to the decryption oracle, the response is $m_1$ (regardless of whether $m_0$ or $m_1$ was chosen), instead of an error.

This modified IND-CCA experiment can be directly obtained as a special case of IND-HCCA as follows: $\mathsf{RigEnc}$ generates an encryption of $m_0$, and uses the ciphertext itself as the auxiliary information. $\mathsf{RigExtract}$ simply checks if an input ciphertext is identical to the auxiliary information; if so, it outputs the identity transformation (indicating that the given ciphertext encodes the same plaintext as the output of $\mathsf{RigEnc}$); otherwise, it outputs $\perp$. The auxiliary information shared between $\mathsf{RigEnc}$ and $\mathsf{RigExtract}$ is public, and $\mathsf{RigExtract}$ does not use the private key, thus separate oracle access to $\mathsf{GRigEnc}$ and $\mathsf{GRigExtract}$ is redundant. Removing these redundant oracles, what remains is the modified IND-CCA experiment, with $\mathsf{RigDec}$ acting as the decryption oracle.

Similarly, *benignly non-malleable* (or gCCA) security [34, 1] is obtained if $\mathsf{RigExtract}$ is allowed to compute an arbitrary equivalence relation among the two ciphertexts (but still without being given the private key). *Replayable CCA* (RCCA) security [10] is obtained if $\mathsf{RigEnc}$ encrypts a random plaintext (using the plaintext as the auxiliary information), and $\mathsf{RigExtract}$ simply decrypts the given ciphertext and checks whether the result equals the auxiliary information. In this case, $\mathsf{RigExtract}$ only uses the private key to use $\mathsf{Dec}$ as a black box, so separate oracle access to $\mathsf{GRigEnc}$ and $\mathsf{GRigExtract}$ is again redundant.

Note that in all these cases $\mathsf{RigExtract}$ is allowed to output only $\perp$ or the identity transformation. This highlights the fact that schemes satisfying these security definitions are not malleable in ways which alter the message. □

[3] If an adversary can successfully distinguish between encryptions of $m$ versus $m'$ in the IND-CCA experiment, then a related adversary can successfully distinguish between either $(m, m_0)$ or $(m', m_0)$.

We note that all of these special cases of HCCA involve a $\mathsf{RigEnc}$ procedure which simply encrypt a plaintext as normal. In our construction (Section 5), we exploit the flexibility of the full HCCA definition to achieve larger classes of transformations, by letting $\mathsf{RigEnc}$ generate a “ciphertext” that is not in the range of $\mathsf{Enc}(\cdot)$.

### 4.2 HCCA and Unlinkability Imply UC Security

**Theorem 2.** *Every $\mathcal{T}$-homomorphic encryption scheme which is HCCA-secure, unlinkably homomorphic (with respect to $\mathcal{T}$) and satisfies the correctness properties, is a secure realization of $\mathcal{F}^{\mathcal{T}}_{\text{HMP}}$ in the standard UC model, where the adversary is allowed to make only static (non-adaptive) corruptions.*

Let $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{CTrans})$ be an unlinkably homomorphic, HCCA-secure encryption scheme (with allowable homomorphisms $\mathcal{T}$). To prove Theorem 2, for any real-world adversary $\mathcal{A}$, we must demonstrate an ideal-world adversary (simulator) $\mathcal{S}$, so that for all PPT environments $\mathcal{Z}$, $\text{REAL}^{\Pi}_{\mathcal{A},\mathcal{Z}} \approx \text{IDEAL}^{\mathcal{F}^{\mathcal{T}}_{\text{HMP}}}_{\mathcal{S},\mathcal{Z}}$.

In the case where the recipient $P$ is corrupt, the simulation is trivial. Each time it is asked to generate a handle, it is given the underlying message. Each time the adversary itself outputs a ciphertext, the simulator can register it as a dummy handle, after which it is notified each time that handle is REPOST'ed. We now focus on the case where $P$ is not corrupt.

We construct $\mathcal{S}$ in a sequence of hybrids, starting from the real-world interactions and altering it step by step to get an ideal-world adversary, at every stage ensuring that each change remains indistinguishable to all environments. All the simulators below exist in the ideal world, but are also given (progressively less) information about the inputs to the honest parties. We conveniently model this access to extra information using modified functionalities.

**$\mathcal{S}_0$ and $\mathcal{F}_0$ (Correctness):** $\mathcal{F}_0$ behaves exactly like $\mathcal{F}^{\mathcal{T}}_{\text{HMP}}$ except that in its HANDLE-REQ interactions with the adversary, it reveals the message and whether the handle is being requested for a repost. Thus $\mathcal{S}_0$ effectively learns all the honest parties' inputs to $\mathcal{F}_0$. $\mathcal{S}_0$ internally simulates the encryption scheme algorithms for all honest parties, and lets the adversary $\mathcal{A}$ interact with these simulated parties and directly with the environment, as follows:

1. When an honest party $P$ sends a SETUP command to $\mathcal{F}_0$, the functionality sends $(\text{ID-REQ}, P)$ to $\mathcal{S}_0$ and expects an ID in return. $\mathcal{S}_0$ generates a key pair $(PK, SK) \leftarrow \mathsf{KeyGen}$ and uses $PK$ as the ID string. It also internally simulates to $\mathcal{A}$ that $P$ broadcast the public key.

2. When an honest party $\mathsf{sender}$ sends a command $(\text{POST}, \mathsf{msg})$ to $\mathcal{F}_0$, the functionality sends $(\text{HANDLE-REQ}, \mathsf{sender}, \mathsf{msg})$ to $\mathcal{S}_0$ and expects a handle in return. $\mathcal{S}_0$ computes $\mathsf{handle} \leftarrow \mathsf{Enc}_{PK}(\mathsf{msg})$ and uses it as the handle. It also internally simulates to $\mathcal{A}$ that $\mathsf{sender}$ broadcast $\mathsf{handle}$.

3. When an honest party $\mathsf{sender}$ sends a command $(\text{REPOST}, \mathsf{handle}, T)$ to $\mathcal{F}_0$, and $\mathsf{handle}$ is internally recorded, the functionality sends $(\text{HANDLE-REQ}, \mathsf{sender}, \mathsf{handle}, T)$ to $\mathcal{S}_0$ and expects a handle in return. $\mathcal{S}_0$ computes $\mathsf{handle}' \leftarrow \mathsf{CTrans}(\mathsf{handle}, T)$ and uses it as the handle. It also internally simulates to $\mathcal{A}$ that $\mathsf{sender}$ broadcast $\mathsf{handle}'$.

4. When the adversary broadcasts a ciphertext $\zeta$, $\mathcal{S}_0$ does the following:

- If $\mathsf{Dec}_{SK}(\zeta) = \mathsf{msg} \neq \perp$, then $\mathcal{S}_0$ sends $(\textsc{post}, \mathsf{msg})$ to the functionality on behalf of $\mathcal{A}$. It uses $\zeta$ as the corresponding handle.
- Otherwise, $\mathcal{S}_0$ sends $(\textsc{dummy}, \zeta)$ to $\mathcal{F}_0$.

We denote the output of an environment $\mathcal{Z}$ when interacting with $\mathcal{S}_0$ and honest parties who interact with $\mathcal{F}_0$ by $\textsc{ideal}^{\mathcal{F}_0}_{\mathcal{S}_0,\mathcal{Z}}$.

**Claim 1.** *For any given PPT adversary, let $\mathcal{F}_0$ and $\mathcal{S}_0$ be as described above. Then for all PPT environments $\mathcal{Z}$, $\textsc{real}^{\Pi}_{\mathcal{A},\mathcal{Z}} \approx \textsc{ideal}^{\mathcal{F}_0}_{\mathcal{S}_0,\mathcal{Z}}$.*

*Proof.* This follows from the correctness properties of encryption scheme $\Pi$, and the fact that $\mathcal{S}_0$ exactly emulates the real world actions of all parties. □

**$\mathcal{S}_1$ and $\mathcal{F}_1$ (Unlinkable Homomorphism):** $\mathcal{F}_1$ is identical to $\mathcal{F}_0$ except that it does not tell the adversary whether a $\textsc{handle-req}$ was the result of a $\textsc{post}$ or $\textsc{repost}$ command, except for dummy handles. The exact differences are as follows:

1. When an honest party $\mathsf{sender}$ sends a command $(\textsc{repost}, \mathsf{handle}, T)$ to $\mathcal{F}_1$, and $(\mathsf{handle}, \mathsf{msg})$ is internally recorded, and $\mathsf{msg} \neq \perp$, the functionality now does the same thing as if $\mathsf{sender}$ had given the command $(\textsc{post}, \mathsf{id}, T(\mathsf{msg}))$ command. If $\mathsf{msg} = \perp$, the functionality sends $(\textsc{handle-req}, \mathsf{sender}, \mathsf{handle}, T)$ to the simulator just as in $\mathcal{F}_0$.
2. $\mathcal{S}_1$ and $\mathcal{S}_0$ are identical, although they receive different types of $\textsc{handle-req}$ requests when interacting with $\mathcal{F}_1$ instead of $\mathcal{F}_0$.

**Claim 2.** *For any given PPT adversary $\mathcal{A}$, let $\mathcal{S}_0$, $\mathcal{F}_0$, $\mathcal{S}_1$ and $\mathcal{F}_1$ be as described above. Then for all PPT environments $\mathcal{Z}$, $\textsc{ideal}^{\mathcal{F}_0}_{\mathcal{S}_0,\mathcal{Z}} \approx \textsc{ideal}^{\mathcal{F}_1}_{\mathcal{S}_1,\mathcal{Z}}$.*

*Proof.* This follows from the unlinkable homomorphism property of the encryption scheme. The only manner in which the adversary's view differs in the two executions is in whether certain ciphertexts are generated via a transformation (as $\mathcal{S}_0$ does on receiving a $(\textsc{handle-req}, \mathsf{sender}, \mathsf{handle}, T)$ request) or as a fresh encryptions of the appropriate message (as $\mathcal{S}_1$ does on receiving a $(\textsc{handle-req}, \mathsf{sender}, T(\mathsf{msg}$ request). We note that $\mathcal{F}_1$ only behaves differently when $\mathsf{msg} \neq \perp$. Thus the difference between executions only involves ciphertexts which were either honestly generated by the simulator, or adversarially generated ciphertexts that successfully decrypted under $PK$. The unlinkable homomorphism property of the scheme implies that the difference in these interactions' outcomes is negligible.

More formally, we can only apply the unlinkable homomorphism property to one ciphertext at a time. It is straightforward to construct a sequence of hybrid simulators where the difference between successive hybrids is in whether a single handle was freshly re-encrypted or had a transformation applied, and such that the rest of the interaction can be carried out within the unlinkability experiment. Thus $\textsc{ideal}^{\mathcal{F}_0}_{\mathcal{S}_0,\mathcal{Z}} \approx \textsc{ideal}^{\mathcal{F}_1}_{\mathcal{S}_1,\mathcal{Z}}$. □

**$\mathcal{S}_2$ and $\mathcal{F}_2$ (HCCA security):** $\mathcal{F}_2$ is identical to $\mathcal{F}_1$ except that it does not tell the adversary the contents of the posted messages when the receiver is not corrupted. $\mathcal{S}_2$ differs from $\mathcal{S}_1$ accordingly, and uses the $\mathsf{RigEnc}$ and $\mathsf{RigExtract}$ features guaranteed by HCCA security. The exact differences are as follows:

1. When $P$ is not corrupt and $\mathcal{F}_1$ would send ($\textsc{handle-req}, \mathsf{sender}, \mathsf{msg}$) to the simulator (i.e, when a party posts or reposts a non-dummy handle), $\mathcal{F}_2$ instead sends ($\textsc{handle-req}, \mathsf{sender}$).
2. When $\mathcal{S}_2$ receives a request of the form ($\textsc{handle-req}, \mathsf{sender}$) from $\mathcal{F}_2$, it computes $(\mathsf{handle}, S) \leftarrow \mathsf{RigEnc}_{PK}$ and uses $\mathsf{handle}$ as the message's handle. It internally keeps track of $(\mathsf{handle}, S)$ for later use.
3. When the adversary broadcasts a ciphertext $\zeta$, $\mathcal{S}_2$ does the following: For each $(\mathsf{handle}, S)$ recorded above, $\mathcal{S}_2$ checks if $\mathsf{RigExtract}_{SK}(\zeta, S) = T \neq \perp$. If so, $\mathcal{S}_2$ sends ($\textsc{repost}, \mathsf{handle}, T$) to $\mathcal{F}_2$ and uses $\zeta$ as the handle. If none of these $\mathsf{RigExtract}$ calls succeed, then $\mathcal{S}_2$ proceeds just as $\mathcal{S}_1$ (i.e, attempts to decrypt $\zeta$ under $SK$ and so on).

**Claim 3.** *For any given PPT adversary $\mathcal{A}$, let $\mathcal{S}_1$, $\mathcal{F}_1$, $\mathcal{S}_2$ and $\mathcal{F}_2$ be as described above. Then for all PPT environments $\mathcal{Z}$, $\textsc{ideal}^{\mathcal{F}_1}_{\mathcal{S}_1,\mathcal{Z}} \approx \textsc{ideal}^{\mathcal{F}_2}_{\mathcal{S}_2,\mathcal{Z}}$.*

*Proof.* This follows from the HCCA security of the scheme. Intuitively, the only way the two executions differ is in whether the simulator provides honest ciphertexts (as $\mathcal{S}_1$ does) or "rigged" ciphertexts (as $\mathcal{S}_2$ does).

More formally, we can only apply the HCCA guarantee to one ciphertext at a time. It is straightforward to construct a sequence of hybrid simulators where the difference between successive hybrids is in whether a single handle was encrypted with the correct message or else via $\mathsf{RigEnc}$, and such that the rest of the interaction can be carried out within the HCCA experiment. We note that in most hybrids, there must be calls to $\mathsf{RigEnc}$ and $\mathsf{RigExtract}$ for other ciphertexts, so the $\mathsf{GRigEnc}$ and $\mathsf{GRigExtract}$ oracles are crucial in the HCCA definition. □

**Concluding the proof.** Combining the above claims we get that for all adversaries $\mathcal{A}$, there exists a simulator $\mathcal{S}_2$ such that $\textsc{real}^{\Pi}_{\mathcal{A},\mathcal{Z}} \approx \textsc{ideal}^{\mathcal{F}_2}_{\mathcal{S}_2,\mathcal{Z}}$ for all environments $\mathcal{Z}$. Note that $\mathcal{F}_2$ is in fact identical to $\mathcal{F}^{\mathcal{T}}_{\text{HMP}}$. So letting $\mathcal{S} = \mathcal{S}_2$ completes the proof.

## 4.3 Obtaining RCCA from HCCA

In general, one cannot easily modify a $\mathcal{T}_1$-unlinkable-HCCA-secure scheme into a $\mathcal{T}_2$-unlinkable-HCCA-secure scheme, even if $\mathcal{T}_2 \subseteq \mathcal{T}_1$. The problem of "disabling" the transformations in $\mathcal{T}_1 \setminus \mathcal{T}_2$ while at the same time maintaining those in $\mathcal{T}_2$ appears just as challenging as constructing a $\mathcal{T}_2$-unlinkable-HCCA scheme from scratch. However, such a generic reduction is possible for the special case of unlinkable (a.k.a. rerandomizable) RCCA security, where the only allowed transformation is the identity function:

**Theorem 3.** *Given a $\mathcal{T}$-unlinkable-HCCA-secure scheme and a (not necessarily unlinkable) RCCA-secure scheme, it is possible to construct an unlinkable RCCA-secure scheme.*

Note that RCCA security without unlinkability is a weaker requirement than CCA security [10]. Thus, for example, an unlinkable HCCA-secure scheme along with a plain CCA-secure encryption scheme will yield an unlinkable RCCA-secure encryption scheme.

*Proof.* Let $(\mathsf{KeyGen}^H, \mathsf{Enc}^H, \mathsf{Dec}^H, \mathsf{CTrans}^H)$ be the unlinkable, $\mathcal{T}$-HCCA-secure scheme, and let $(\mathsf{KeyGen}^R, \mathsf{Enc}^R, \mathsf{Dec}^R)$ be the RCCA-secure scheme. The new scheme simply encapsulates the RCCA-secure scheme inside the HCCA-secure one. Formally:

- $\mathsf{KeyGen}$: Run $(hpk, hsk) \leftarrow \mathsf{KeyGen}^H$; $(rpk, rsk) \leftarrow \mathsf{KeyGen}^R$. Output $(PK = (hpk, rpk), SK = (hsk, rsk))$.
- $\mathsf{Enc}_{hpk,rpk}(\mathsf{msg}) = \mathsf{Enc}^H_{hpk}(\mathsf{Enc}^R_{rpk}(\mathsf{msg}))$.
- $\mathsf{Dec}_{hsk,rsk}(\zeta) = \mathsf{Dec}^R_{rsk}(\mathsf{Dec}^H_{hsk}(\zeta))$, where we let $\mathsf{Dec}^R(\bot) = \bot$ for simplicity.
- $\mathsf{CTrans} = \mathsf{CTrans}^H$.

Note that in $\mathsf{CTrans}$, the only allowed transformation is the identity function, while for $\mathsf{CTrans}^H$, there may be other allowed transformations. We note that to achieve HCCA security with respect to $\mathcal{T}$, $\mathcal{T}$ must indeed contain the identity function (see Section 3.1). It is easy to see that the unlinkability of the outer scheme (with respect to the identity transformation) is preserved by the construction.

To show RCCA security (HCCA security with the identity function as the only allowed transformation), we must demonstrate appropriate $\mathsf{RigEnc}$ and $\mathsf{RigExtract}$ procedures for the new scheme. Let $(\mathsf{RigEnc}^H, \mathsf{RigExtract}^H)$ and $(\mathsf{RigEnc}^R, \mathsf{RigExtract}^R)$ be the procedures guaranteed by the two schemes, respectively. Then the new scheme satisfies HCCA security with the following procedures:

- $\mathsf{RigEnc}_{hpk,rpk}$ does the following: Run $(\zeta, S_H) \leftarrow \mathsf{RigEnc}^H_{hpk}$ and $(\zeta_R, S_R) \leftarrow \mathsf{RigEnc}^R_{rpk}$. Set $S = (S_H, \zeta_R, S_R)$ and output $(\zeta, S)$.
- $\mathsf{RigExtract}_{hsk,rsk}(\zeta, S)$ does the following: Parse $S$ as $(S_H, \zeta_R, S_R)$. Run $T \leftarrow \mathsf{RigExtract}^H_{hsk}(\zeta, S_H)$. If $T = \bot$, output $\bot$; otherwise output $\mathsf{RigExtract}^R_{rsk}(T(\zeta_R), S_R)$, which must be either the identity function or $\bot$, by the RCCA security of the inner scheme.

Consider a hybrid HCCA experiment where the challenge ciphertext is generated from $\mathsf{msg}^*$ as:

- Run $(\zeta^*, S^*) \leftarrow \mathsf{RigEnc}^H_{hpk}$ and $\zeta^*_R \leftarrow \mathsf{Enc}^R_{rpk}(\mathsf{msg}^*)$. Remember $\zeta^*_R$ and output $\zeta^*$.

and $\mathsf{RigDec}$ in step 4 is implemented as:

$$\mathsf{RigDec}_{hsk,rsk}(\zeta) = \begin{cases} \mathsf{Dec}^R_{rsk}(T(\zeta^*_R)) & \text{if } \bot \neq T \leftarrow \mathsf{RigExtract}^H_{hsk}(\zeta, S^*) \\ \mathsf{Dec}^R_{rsk}(\mathsf{Dec}^H_{hsk}(\zeta)) & \text{otherwise.} \end{cases}$$

It is straight-forward to verify that this hybrid experiment is indistinguishable from both the $b = 0$ and $b = 1$ branches of the HCCA experiment instantiated with the new scheme and its $\mathsf{RigEnc}$ and $\mathsf{RigExtract}$ procedures described above. □

# 5 Achieving Security

Our main result is a family of construtions which achieves both HCCA security and unlinkable homomorphism, with respect to a wide range of message transformations, under the standard DDH assumption in two related groups.

Our construction is based on the rerandomizable RCCA scheme of Prabhakaran and Rosulek [29]. Recall that rerandomizable RCCA security is a special case of unlinkable HCCA security where the only allowed transformation is the identity function. Indeed, for the appropriate choice of parameters, our construction coincides with the one presented in [29].

**Requirements.** As in [29], our construction requires two (multiplicative) cyclic groups with a specific relationship: $\mathbb{G}$ of prime order $p$, and $\widehat{\mathbb{G}}$ of prime order $q$, where $\widehat{\mathbb{G}}$ is a subgroup of $\mathbb{Z}_p^*$. We require the DDH assumption to hold in both groups (with respect to the same security parameter). Given a sequence of primes $q, 2q+1, 4q+3$ (a *Cunningham chain* of the first kind of length 3 [2]), the two quadratic-residue groups $\widehat{\mathbb{G}} = \mathbb{QR}^*_{2q+1}$ and $\mathbb{G} = \mathbb{QR}^*_{4q+3}$, in which the DDH assumption is believed to hold, represent a suitable choice.

**Notation and Supported Transformations.** Let "$*$" denote the group operation in the product group $\mathbb{G}^n$ defined by $(\alpha_1, \ldots, \alpha_n) * (\beta_1, \ldots, \beta_n) = (\alpha_1\beta, \ldots \alpha_n\beta_n)$.

For $\tau \in \mathbb{G}^n$, define $T_\tau$ to be the "multiplication by $\tau$" transformation in $\mathbb{G}^n$; i.e., $T_\tau(m) = \tau * m$. We also let $T_\tau(\perp) = \perp$ for simplicity. Now let $\mathbb{H}$ be a subgroup of $\mathbb{G}^n$. Our construction provides a scheme whose message space is $\mathbb{G}^n$, and whose set of allowable transformations is $\mathcal{T}_{\mathbb{H}} = \{T_\tau \mid \tau \in \mathbb{H}\}$. By chosing $\mathbb{H}$ appropriately, we can obtain the following notable classes $\mathcal{T}_{\mathbb{H}}$:

- The identity function alone (i.e., rerandomizable RCCA security), by setting $\mathbb{H} = \{1\}$.
- All transformations $T_\tau$ (that is, all component-wise multiplications in $\mathbb{G}^n$), by setting $\mathbb{H} = \mathbb{G}^n$.
- "Scalar multiplication" of tuples in $\mathbb{G}^n$ by coefficients in $\mathbb{G}$, by setting $\mathbb{H} = \{(s, \ldots, s) \mid s \in \mathbb{G}\}$.

## 5.1 The Construction

**Double-strand Malleable Encryption Scheme.** We now define a homomorphic encryption scheme which, following [29], we call the "Double-strand malleable encryption" (DSME), and which we use as a component in our main construction.

**System parameters.** A cyclic multiplicative group $\widehat{\mathbb{G}}$ of prime order $q$. $\widehat{\mathbb{G}}$ also acts as the message space for this scheme. Similar to above, we denote $T_\sigma$ as the multiplication-by-$\sigma$ transformation in $\widehat{\mathbb{G}}$, with $T_\sigma(\perp) = \perp$.

**Key generation.** Pick random generators $\widehat{g}_1, \widehat{g}_2$ from $\widehat{\mathbb{G}}$, and random $\vec{a} = (a_1, a_2), \vec{b} = (b_1, b_2)$ from $(\mathbb{Z}_q)^2$. The private key is $(\vec{a}, \vec{b})$. The public key consists of $\widehat{g}_1$, $\widehat{g}_2$, and the following values:

$$A = \textstyle\prod_{j=1}^2 \widehat{g}_j^{a_j}; \qquad B = \prod_{j=1}^2 \widehat{g}_j^{b_j}$$

**Encryption (MEnc):** To encrypt $u \in \widehat{\mathbb{G}}$ under public key $(\widehat{g}_1, \widehat{g}_2, A, B)$, first pick random $v \in \mathbb{Z}_q$, $w \in \mathbb{Z}_q^*$. Output

$$(\widehat{g}_1^v, \widehat{g}_2^v, uA^v, B^v; \widehat{g}_1^w, \widehat{g}_2^w, A^w, B^w).$$

**Decryption (MDec):** To decrypt $U = (V_1, V_2, A_V, B_V; W_1, W_2, A_W, B_W)$ under private key $(\vec{a}, \vec{b})$: First, if $W_1 = W_2 = 1$, then output $\perp$. Then check the following constraints:

$$A_W \stackrel{?}{=} \textstyle\prod_{j=1}^2 W_j^{a_j}; \qquad B_V \stackrel{?}{=} \prod_{j=1}^2 V_j^{b_j}; \qquad B_W \stackrel{?}{=} \prod_{j=1}^2 W_j^{b_j};$$

If any fail, output $\perp$. Otherwise, output $u = A_V / \prod_{j=1}^2 V_j^{a_j}$.

**Ciphertext transformation (MCTrans):** To apply transformation $T_\sigma$ to the ciphertext $U = (\vec{V}, A_V, B_V; \vec{W}, A_W, B_W)$ choose random $s \in \mathbb{Z}_q$, $t \in \mathbb{Z}_q^*$ and output

$$(V_1 W_1^s, V_2 W_2^s, \sigma A_V A_W^s, B_V B_W^s; W_1^t, W_2^t, A_W^t, B_W^t)$$

It is not hard to see that if $U$ is in the support of $\mathsf{MEnc}_{\widehat{PK}}(u)$ (with random choices $v$ and $w$), then the $\mathsf{MCTrans}(U, T_\sigma)$ is in the support of $\mathsf{MEnc}_{\widehat{PK}}(\sigma u)$, corresponding to random choices $v' = v + sw$ and $w' = tw$.

We emphasize that this DSME scheme does not achieve our desired definitions of a multiplicative homomorphic scheme, because given an encryption of $u$ and a value $r \in \mathbb{Z}_q$, one can easily construct an encryption of $u^r$, and exponentiation by $r$ is not an allowed transformation. Our main construction uses only the multiplicative transformation of DSME as a feature, although the security analysis accounts for the fact that other types of transformation are possible.

**Main Construction.** We now present our main construction, which uses the previous DSME scheme as a component.

**System parameters.** A cyclic multiplicative group $\mathbb{G}$ of prime order $p$. A space of messages. We also require a secure DSME scheme over a group $\widehat{\mathbb{G}}$ of prime order $q$, where $\widehat{\mathbb{G}}$ is also a subgroup of $\mathbb{Z}_p^*$. This relationship is crucial, as the ciphertext transformation $\mathsf{MCTrans}$ of the DSME scheme must coincide with multiplication in the exponent in $\mathbb{G}$.

As described above, we let $\mathbb{G}^n$ be the message space, for any parameter $n$. The scheme is also parametrized by the subgroup $\mathbb{H}$ of allowed transformations. For $m \in \mathbb{G}^n$, let $f(m)$ be an injective encoding (say, an arbitrary representative) of the coset $m * \mathbb{H}$ to $\{0,1\}^*$. Let $\mathcal{H}$ be a family of collision-resistant hash functions from $\{0,1\}^*$ to $\mathbb{Z}_p$.[4] Finally, we require any fixed vector $\vec{z} \in (\mathbb{Z}_p)^4$, which is not a scalar multiple of the all-ones vector.

**Key generation ($\mathsf{KeyGen}$).** Generate a keypair $(\widehat{PK}, \widehat{SK})$ for the DSME scheme in $\widehat{\mathbb{G}}$. Pick random generators $g_1, \ldots, g_4$ from $\mathbb{G}$. For $i \in [n]$, choose random $\vec{c_i} = (c_{i,1}, \ldots, c_{i,4})$ from $(\mathbb{Z}_p)^4$ and compute $C_i = \prod_{j=1}^4 g_j^{c_{i,j}}$. Choose random $\vec{d} = (d_1, \ldots, d_4), \vec{e} = (e_1, \ldots, e_4)$ from $(\mathbb{Z}_p)^4$ and compute $D = \prod_{j=1}^4 g_j^{d_j}$ and $E = \prod_{j=1}^4 g_j^{e_j}$. Choose a random hash $\mathsf{H} \leftarrow \mathcal{H}$.

The private key for the scheme is $(\widehat{SK}, \vec{c}_1, \ldots, \vec{c}_n, \vec{d}, \vec{e})$. The public key is $(\widehat{PK}, g_1, \ldots, g_4, C_1, \ldots, C_n, D, E, \mathsf{H})$.

**Encryption ($\mathsf{Enc}$):** To encrypt $(m_1, \ldots, m_n) \in \mathbb{G}^n$ under a public key of the preceding form, first compute $\mu = \mathsf{H}(f(m_1, \ldots, m_n))$. Then pick random $x \in \mathbb{Z}_p$, $y \in \mathbb{Z}_p^*$ and random $u \in \widehat{\mathbb{G}}$, and output

$$\begin{array}{llllllll} g_1^{(x+z_1)u}, & \ldots, & g_4^{(x+z_4)u}, & m_1 C_1^x, & \ldots, & m_n C_n^x, & (DE^\mu)^x; & \\ g_1^{yu}, & \ldots, & g_4^{yu}, & C_1^y, & \ldots, & C_n^y, & (DE^\mu)^y; & \mathsf{MEnc}_{\widehat{PK}}(u) \end{array}$$

**Decryption ($\mathsf{Dec}$):** Let $\zeta$ be a ciphertext of the preceding form, say, $\zeta = (\vec{X}, \vec{C}_X, P_X; \vec{Y}, \vec{C}_Y, P_Y; U)$, where

$$\begin{array}{ll} \vec{X} = (X_1, \ldots, X_4) & \vec{C}_X = (C_{X,1}, \ldots, C_{X,n}) \\ \vec{Y} = (Y_1, \ldots, Y_4) & \vec{C}_Y = (C_{Y,1}, \ldots, C_{Y,n}) \end{array}$$

[4] Using the same technique as in the Cramer-Shoup scheme [13], our use of a hash can be removed, but at the expense of longer public keys.

First compute $u = \mathsf{MDec}_{\widehat{SK}}(U)$. If $u = \perp$, output $\perp$. Otherwise, strip off $u$ and $\vec{z}$ from the exponents as follows: For $j = 1, \ldots, 4$: set $\overline{X}_j = X_j^{1/u} g_j^{-z_j}$ and $\overline{Y}_j = Y_j^{1/u}$.

Compute the purported plaintext $(m_1, \ldots, m_n)$ via $m_i = C_{X,i} / \prod_{j=1}^{4} \overline{X}_j^{c_{i,j}}$, and then compute $\mu = \mathsf{H}(f(m_1, \ldots, m_n))$. Finally, check the integrity of the ciphertext in the following way. If $Y_1 = \cdots = Y_4 = 1$ (the identity element in $\mathbb{G}$), output $\perp$. Check the following conditions:

$$C_{Y,i} \stackrel{?}{=} \prod_{j=1}^{4} \overline{Y}_j^{c_{i,j}} \text{ (for each } i \in [n]); \qquad P_X \stackrel{?}{=} \prod_{j=1}^{4} \overline{X}_j^{d_j + \mu e_j}; \qquad P_Y \stackrel{?}{=} \prod_{j=1}^{4} \overline{Y}_j^{d_j + \mu e_j}$$

If any checks fail, output $\perp$, otherwise output $(m_1, \ldots, m_n)$.

**Ciphertext transformation (CTrans):** Let $\zeta$ be a ciphertext of the preceding form. To apply transformation $T_{(\tau_1, \ldots, \tau_n)}$ to $\zeta$, choose random $\sigma \in \widehat{\mathbb{G}}$ and random $s \in \mathbb{Z}_p$, $t \in \mathbb{Z}_p^*$. Output:

$$\begin{array}{cccccccc} (X_1 Y_1^s)^\sigma, & \ldots, & (X_4 Y_4^s)^\sigma, & \tau_1 C_{X,1} C_{Y,1}^s, & \ldots, & \tau_n C_{X,n} C_{Y,n}^s, & P_X P_Y^s; & \\ Y_1^{t\sigma}, & \ldots, & Y_4^{t\sigma}, & C_{Y,1}^t, & \ldots, & C_{Y,n}^t, & P_Y^t; & \mathsf{MCTrans}(U, T_\sigma) \end{array}$$

It is not hard to see that if $\zeta$ is in the support of $\mathsf{Enc}_{PK}(m_1, \ldots, m_n)$, say, with random choices $x$, $y$, and $u$, then the above ciphertext is in the support of $\mathsf{Enc}_{PK}(\tau_1 m_1, \ldots, \tau_n m_n)$, corresponding to random choices $x' = x + sy$, $y' = ty$, and $u' = \sigma u$.

### 5.2 High-level Overview

Disregarding $\vec{z}$ and $u$, ciphertexts in our scheme resemble those in the original Cramer-Shoup scheme [13]. Two similar-looking "strands" are given, with only the first one directly carrying the message. This allows us to refresh the randomness $x$ and $y$ and achieve unlinkability when applying a transformation to the ciphertext. A similar "double-strand" paradigm was used by Golle, et al. [20], applied to the ElGamal encryption scheme to achieve a rerandomizable and anonymous CPA-secure scheme.

Without the additional random value $u$ appearing in the exponents of some of the ciphertext components, the second strand's components would be completely independent of the first strand's. Thus, the scheme would be malleable via an attack which combined the first strand of one ciphertext and the second strand of another – the combination would result in a valid ciphertext if and only if the two ciphertexts shared the same $\mu$ value. The addition of $u$ and its encryption under $\mathsf{MEnc}$ correlates the two strands, leaves $u$ hidden from eavesdroppers, yet still allows for the random choice of $u$ to be refreshed.

In our "double-strand" paradigm of achieving unlinkability, $x$'s randomness is refreshed additively (as $x + sy$) and $y$'s multiplicatively (as $ty$). However, without the $\vec{z}$ vector perturbing the randomness in $x$, there is a possible attack whereby $x$ can be rerandomized *multiplicatively* and still result in a valid ciphertext (say, by squaring each component of the first strand). By adding $\vec{z}$, (intuitively) any attack that would multiply $x$ would also multiply $\vec{z}$ as well. The decryption procedure only strips away one copy of $\vec{z}$, so $x$ would remain perturbed for the Cramer-Shoup-like integrity checks on the ciphertext. In our analysis, it is important that $\vec{z}$ is linearly independent of the all-ones vector, so that an adversary would not be able to successfully compensate for additional perturbances in the Cramer-Shoup integrity checks.

We achieve our desired level of non-malleabiliy by a technique similar to the Cramer-Shoup CCA-secure scheme [13]. It uses a ciphertext component of the form $(DE^\mu)^x$, where $D, E$ are parts of the public key, $x$ is a random value used in encryption, and $\mu$ is a hash of the ciphertext's prefix. The rerandomizable RCCA scheme of [29] uses the same paradigm, except that the value $\mu$ is a direct encoding of the plaintext (in rerandomizable RCCA, ciphertexts are malleable but only in ways which preserve the plaintext). In our HCCA-secure scheme, $\mu$ is a hash of an encoding of the coset $m * \mathbb{H}$. Intuitively, our scheme can therefore only be malleable in ways which preserve the $\mathbb{H}$-coset of the plaintext.

**Theorem 4.** *The construction satisfies the correctness requirements, HCCA security, and unlinkable homomorphism properties with respect to $\mathcal{T}_{\mathbb{H}}$, for any subgroup $\mathbb{H}$ of $\mathbb{G}^n$, under the DDH assumption in the two cyclic groups.*

The lengthy proof follows in Appendix A, and is a careful generalization of that of [29].

# 6 Beyond Unary Transformations

Many interesting applications of homomorphic encryptions involve (at least) *binary* operations — those which accept encryptions of plaintexts $m_0$ and $m_1$ and output a ciphertext encoding $T(m_0, m_1)$. A common example is ElGamal encryption, where $T$ is the group operation of the underlying cyclic group. In this section, we examine the possibility of extending our results to schemes with *binary* transformations. We show some simple positive results, and also an impossibility for the specific case of group operations.

## 6.1 Extending Definitions

It is straight-forward to extend the UC definition of $\mathcal{F}^{\mathcal{T}}_{\text{HMP}}$ to handle binary transformations. We define $\mathcal{F}^{\mathcal{T}}_{\text{2-HMP}}$ to act like $\mathcal{F}^{\mathcal{T}}_{\text{HMP}}$, except that $\mathcal{T}$ is a set of allowed *binary* transformations. Honest parties may then generate a derived post by giving *two* handles an an allowed transformation $T$. Naturally, the functionality internally acts as if the party had requested a post of $T(m_1, m_2)$, where $m_1, m_2$ are the messages corresponding to the given handles.

It is not clear what is the most appropriate behavior for $\mathcal{F}^{\mathcal{T}}_{\text{2-HMP}}$ when one or both of the given handles is a dummy handle. Our impossibility results in this section do not depend on any particular behavior of $\mathcal{F}^{\mathcal{T}}_{\text{2-HMP}}$ in such a case, so we opt to make the definition as weak as possible. When such a request is made, we let the adversary learn the transformation and the two handles.

Defining an analog of our indistinguishability definition, however, appears to be much more difficult task. Indeed, it is not clear how to appropriately handle the case where the adversary applies a transformation to a pair of ciphertexts where one or both were generated via (independent calls to) $\mathsf{RigEnc}$.

Below we show that it is impossible to securely realize $\mathcal{F}^{\mathcal{T}}_{\text{2-HMP}}$ (i.e., achieve unlinkability and an HCCA-like definition for binary transformations) for a large class of useful transformations $\mathcal{T}$. Still, one may be willing to relax the unlinkability requirement (e.g., as in [33]) and still demand some non-malleability. Thus we leave it as an important open problem to give a meaningful generalization of HCCA security, for transformations that combine multiple ciphertexts.

## 6.2 A Simple Positive Result

$\mathcal{F}^{\mathcal{T}}_{\text{2-HMP}}$ can be achieved for some simple transformation spaces, by appropriately composing an HCCA-secure, unlinkably homomorphic (unary) scheme $\Pi$. To see this, construct a new scheme whose ciphertexts are $k$-tuples of completely independent ciphertexts from $\Pi$. The (binary) transformation operations in the new scheme have the following form: Given two tuples of $k$ ciphertexts each, choose $k$ components from among these $2k$ ciphertexts, apply an allowed transformation to each separately, and output those $k$ ciphertexts in a tuple.

The resulting scheme's transformations simply "mix and match" the independent components of the two tuples of ciphertexts to form a new tuple. The unlinkability of the transformations applied to each individual component implies the unlinkability of the new scheme. It is easy to see that such a scheme securely realizes of $\mathcal{F}^{\mathcal{T}}_{\text{2-HMP}}$ with respect to the appropriate transformation space.

### 6.3 Negative Results

The positive result presented above appears to be much less sophisticated than, say, a scheme that is homomorphic with respect to a group operation. Indeed, this limitation turns out to be inherent in securely realizing $\mathcal{F}^{\mathcal{T}}_{\text{2-HMP}}$.

**Theorem 5.** *There is no secure realization of $\mathcal{F}^{\mathcal{T}}_{\text{2-HMP}}$ via a homomorphic encryption scheme, when $\mathcal{T}$ contains a quasigroup operation*[5] *on the message space.*

The main observation is that each handle (ciphertext) must have a bounded length independent of its "history" (i.e., whether it was generated via the homomorphic operation and if so, which operations applied to which existing handles), and thus can only encode a bounded amount of information about its history. We show that any simulator for $\mathcal{F}^{\mathcal{T}}_{\text{2-HMP}}$ must be able to extract a reasonble history from any handle output by the adversary.

However, when a quasigroup operation is an allowed transformation, there can be far more possible histories than can be encoded in a single handle. We use this fact to construct an environment and adversary which can distinguish between the real world and the ideal world with any simulator, contradicting the security definition.

To show Theorem 5, we establish a technical lemma which demonstrates the desired contradiction whenever the space of "possible histories" for a transition space grows large enough. We note that the proof does not rely on complete unlinkability as specified in $\mathcal{F}^{\mathcal{T}}_{\text{2-HMP}}$. Instead, it only uses the fact that ciphertexts in the scheme are bounded in length, independent of the number of transformations that were used to generate them. Thus Theorem 5 applies even when slight relaxations of unlinkability are considered.

**Definition 4.** *Let $\mathcal{T}$ be a set of* binary *message transformations over a message space $\mathcal{M}$, and let $x_1, \ldots$ be variables over $\mathcal{M}$. Define $\mathcal{C}_{\mathcal{T}}$ inductively as follows:*

- *For all $x_i$, the selection function $(x_1, \ldots) \mapsto x_i$ is in $\mathcal{C}_{\mathcal{T}}$.*
- *For all $m \in \mathcal{M}$, the constant function $(x_1, \ldots) \mapsto m$ is in $\mathcal{C}_{\mathcal{T}}$.*
- *For all $T \in \mathcal{T}$, if $f, g \in \mathcal{C}_{\mathcal{T}}$, then the function $(x_1, \ldots) \mapsto T(f(x_1, \ldots), g(x_1, \ldots))$ is also in $\mathcal{C}_{\mathcal{T}}$.*

[5] A quasigroup operation $\star$ on a set $X$ is an operation such that fixing any two values in the equation $x \star y = z$ *uniquely* determines the third value.

*Furthermore, define $\mathcal{C}_{\mathcal{T}}^d$ as the subset of $\mathcal{C}_{\mathcal{T}}$ of functions that can be equivalently written as a function of only the variables $x_1, \ldots, x_d$.*

$\mathcal{C}_{\mathcal{T}}^d$ represents all the functions of $d$ (unknown) ciphertexts that can be obtained "legally" via the operations of the encryption scheme.

We write $f \approx g$ to denote that the functions $f$ and $g$ agree for an overwhelming fraction of their inputs. When $I = \{i_1, \ldots, i_k\} \subseteq [d]$ and $f \in \mathcal{C}_{\mathcal{T}}^k$, for $k < d$, then we denote $f^I$ as the following function in $\mathcal{C}_{\mathcal{T}}^d$:

$$f^I(x_1, \ldots, x_d) = f(x_{i_1}, \ldots, x_{i_k}).$$

In other words, $f^I$ is simply $f$ evaluated on the variables indexed by $I \subseteq [d]$.

Theorem 5 is then a special case of the following lemma:

**Lemma 1.** *If for some $\mathcal{T}$, $\mathcal{F}_{\text{2-HMP}}^{\mathcal{T}}$ is securely realizable by an encryption scheme (i.e., a non-interactive protocol), then there is a constant $d > 0$ such that for all $f \in \mathcal{C}_{\mathcal{T}}^d$, there is an $g \in \mathcal{C}_{\mathcal{T}}^{d-1}$ and $I \subseteq [d]$, with $|I| = d - 1$ such that $f \approx g^I$.*

In other words, for some $d$, every legal operation on $d$ ciphertexts is actually approximately equivalent to an operation that only depends on $d - 1$ ciphertexts. We obtain Theorem 5 by observing that for a quasigroup operation $\star$, the product $x_1 \star \cdots \star x_d$ (parenthesized in any way — $\star$ need not be associative) is not approximately equal to any function on fewer than $d$ variables. After fixing any $d - 1$ variables in the product, there is a unique setting of the free variable that causes the product to evaluate to any element of the quasigroup.

*Proof.* Suppose $\mathcal{F}_{\text{2-HMP}}^{\mathcal{T}}$ is securely realizable by an encryption scheme, and for the sake of contradiction that the above condition is false. We will construct a family of sets $\{S_d\}_d$, where $S_d \subseteq \mathcal{C}_{\mathcal{T}}^d$, with the following properties: $|S_d| = 2^{\Omega(d)}$; every the functions in $S_d$ are pairwise non-equivalent under the $\approx$ relation; and $S_d$ can be efficiently sampled (polynomial time in $d$).

Using such $S_d$, we obtain a contradiction in the following way. Consider an environment which instantiates $\mathcal{F}_{\text{2-HMP}}^{\mathcal{T}}$ with two honest parties Alice and Bob, and a dummy adversary Carol. The environment chooses $d$ random messages $m_1, \ldots, m_d \leftarrow \mathcal{M}$, where $d$ is a parameter to be fixed later, and instructs Alice to POST them for Bob. Then, the environment chooses a random $f \leftarrow S_d$ and internally applies it to the $d$ resulting handles (ciphertexts). The environment can do this on behalf of the adversary without any interaction, because this protocol implements the REPOST operation via a non-interactive procedure CTrans. Finally, the environment instructs the adversary to broadcast the resulting handle/ciphertext, and the environment checks that Bob received $f(m_1, \ldots, m_d)$ correctly.

The contradiction is obtained as follows: First, due to the security of the encryption scheme, there must exist a simulator for this adversary. Because of how $\mathcal{F}_{\text{2-HMP}}^{\mathcal{T}}$ models non-malleability, the simulator must request the new handle according to one of the "legal" functions $\mathcal{C}_{\mathcal{T}}^d$. In the real world, Bob always receives the correct output, so the simulator must specify a function $f' \in \mathcal{C}_{\mathcal{T}}^d$ such that $f(m_1, \ldots, m_d) = f'(m_1, \ldots, m_d)$ with overwhelming probability. From the definition of $\mathcal{F}_{\text{2-HMP}}^{\mathcal{T}}$, the simulator's view is statistically independent of the choice of $m_1, \ldots, m_d$, so that the simulator must in fact specify an $f' \in \mathcal{C}_{\mathcal{T}}^d$ such that $f' \approx f$, with overwhelming probability.

Finally, observe that in any realization of $\mathcal{F}_{\text{2-HMP}}^{\mathcal{T}}$, all handles must be bounded in length by some fixed polynomial $\ell(k)$, where $k$ is the security parameter (else the unlinkability requirement will be broken). Since $|S_d| = 2^{\Omega(d)}$, we can choose $d = \text{poly}(k)$ such that $|S_d| > 2^{\ell(k)+2}$. The handle output by the adversary (to the simulator) is at most $\ell(k)$ bits long, so the simulator's view has at least 1 bit of uncertainty about the $f$ chosen by the environment. Furthermore, $S_d$ is constructed so that for any $f'$, there is at most one $f \in S_d$ such that $f \approx f'$. Thus with probability at least $1/2$, the $f'$ chosen by the simulator will disagree with $f$ on a non-negligible fraction of inputs; a contradiction.

It suffices then to construct $\{S_d\}$ with the desired properties. For each $d$, there is some function $f_d \in \mathcal{C}^d_{\mathcal{T}}$ such that $f_d \not\approx g^I$ for all $g \in \mathcal{C}^{d-1}_{\mathcal{T}}$ and indices $I \subseteq [d]$, $|I| = d-1$. We define $S_d = \{(f_{|I|})^I : I \subseteq [d]\}$, i.e., all ways to evaluate one of the $f_i$'s on a subset of variables of the appropriate size. Clearly $|S_d| = 2^{\Omega(d)}$.

Next, we claim that the elements in $S_d$ are pairwise non-equivalent under the $\approx$ relation. To see this, take $g \neq g' \in S_d$. By the definition of $S_d$, each of these is simply an evaluation of some $f_i$ on $i$ of its variables. Say, $g$ has $f_i$ as its underlying function and $g'$ has $f_j$. If $i \neq j$, then without loss of generality let $i < j$. By its construction, $f_j$ is not approximately equal to $f_i \in \mathcal{C}^i_{\mathcal{T}}$ evaluated on any subset of variables. Thus $i = j$, and since they are different functions, $g$ and $g'$ must evaluate $f_i = f_j$ on different sets of variables. We may fix all variables randomly except one so that $g$ is fixed and $g'$ has one remaining free variable. With overwhelming probability, the residual $g'$ must not be approximately equal to a constant function, otherwise $f_j$ is approximately equal to a function on $j-1$ variables, a contradiction.

Finally, we observe that $S_d$ can be fully specified (and uniformly sampled from) given just descriptions of the $d$ functions $\{f_i\}$. □

# 7 Extensions

## 7.1 Anonymity

In some applications, it is useful for an encryption scheme to have the additional property of *receiver-anonymity* (also known as *key-privacy*), as introduced by Bellare et al. [3]. Receiver-anonymity means, essentially, that in addition to hiding the underlying plaintext message, a ciphertext does not reveal the public key under which it was encrypted. Encryption schemes with this property are important tools in the design of many systems [30]. The special case of rerandomizable, anonymous, RCCA-secure encryption has interesting applications in mix-nets [20] and anonymous P2P routing [29].

The way we have defined the syntax of the CTrans feature of a homomorphic encryption scheme (i.e, so that it does not require the "correct" public key in addition to the ciphertext), it remains a meaningful feature even in an environment where receiever-anonymity is utilized.

To model the property of receiver-anonymity for HCCA schemes, we consider an anonymous, multi-user variant of the $\mathcal{F}^{\mathcal{T}}_{\text{HMP}}$ UC functionality. This variant allows multiple users to register IDs, and senders to post messages destined for a particular ID. The functionality does not reveal the handle's recipient in its HANDLE-ANNOUNCE broadcasts (or in its HANDLE-REQ requests to the adversary).

Our indistinguishability-based security definitions can also be extended in a simple way to account for receiver-anonymity. We call a homomorphic encryption scheme *HCCA-anonymous* if it is HCCA secure and if the RigEnc and RigExtract procedures from the HCCA security definition can be implemented without the public or private keys (i.e, RigEnc takes no arguments and RigExtract takes only a ciphertext and a saved state).

We also consider an additional correctness requirement on schemes, which is natural in the context of multiple users: With overwhelming probability over $(PK, SK) \leftarrow \mathsf{KeyGen}$ and $(PK', SK') \leftarrow \mathsf{KeyGen}$, we require that $\mathsf{Dec}_{SK'}(\mathsf{Enc}_{PK}(\mathsf{msg})) = \bot$ for every $\mathsf{msg} \in \mathcal{M}$, with probability 1 over the randomness of $\mathsf{Enc}$. In other words, ciphertexts honestly encrypted for one user do not successfully decrypt for another user.

Via a similar argument to the proof of Theorem 2, it can be seen that any HCCA-anonymous, unlinkable scheme which satisfies the additional correctness property is a secure realization of the anonymous variant of $\mathcal{F}^{\mathcal{T}}_{\text{HMP}}$.

Note that this notion of anonymity is a chosen-ciphertext and not a chosen-plaintext (simple ciphertext indistinguishability) one. Our construction does not achieve HCCA-anonymity, since it is possible to combine a ciphertext with a public key and obtain a valid ciphertext if and only if the original ciphertext was encrypted under that public key. We consider it an interesting and important open problem to construct an anonymous, unlinkably homomorphic HCCA encryption scheme, for any $\mathcal{T}$.

## 7.2 Alternate UC Security Definition

For simplicity, we have defined our ideal UC functionality $\mathcal{F}^{\mathcal{T}}_{\text{HMP}}$ in such a way that the adversary is notified on-line every time a handle is generated. As pointed out in [27], this paradigm does not allow the most flexibility. A more general-purpose functionality would be one in which parties privately generate new handles (without the adversary being notified), and can have arbitrary control over how the handles are sent to other parties. If a handle never reaches the adversary, the adversary should not know that it was ever generated. To model this, the functionality can be modified so that the adversary is not notified each time a new handle is generated; instead, following [27], the adversary supplies a handle-generating algorithm during the set-up phase so that handles can be generated without the adversary's intervention/notification.

To securely realize such a functionality via a homomorphic encryption scheme, we must ensure that the scheme satisfies an additional security property; namely, that ciphertexts reveal (even to the receiver) at most the *cumulative effect* of all the transformations that have been applied — in particular, the ciphertext does not reveal which particular sequence of transformations has been applied. This property can be specified more formally as a security experiment, where an adversary supplies a ciphertext $\zeta$ and two transformations $T_1$ and $T_2$. The challenger flips a fair coin and returns either either $\mathsf{CTrans}(\zeta, T_2 \circ T_1)$ or $\mathsf{CTrans}(\mathsf{CTrans}(\zeta, T_1), T_2)$, correspondingly. We insist that the adversary cannot correctly guess the coin with nonnegligible advantage.

With this additional security requirement, an analog of Theorem 2 holds for this non-broadcasting definition of $\mathcal{F}^{\mathcal{T}}_{\text{HMP}}$. Our construction does indeed satisfy this additional property, since the two distributions $\mathsf{CTrans}(\zeta, T_2 \circ T_1)$ and $\mathsf{CTrans}(\mathsf{CTrans}(\zeta, T_1), T_2)$ are identical.

## 7.3 Relaxing the Definition of Unlinkability

The definition of unlinkable homomorphism given in Section 3 is very strong, in that it expresses a requirement even for adversarially-generated ciphertexts. There is a relaxation of this definition that only expresses a requirement for honestly generated ciphertexts, and is still meaningful:

**Definition 5.** *We say that a homomorphic encryption scheme is* weakly unlinkably hommormophic *if for every PPT adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ in the following experiment is negligible:*

1. **Setup:** *Pick $(PK, SK) \leftarrow \mathsf{KeyGen}$ and give $PK$ to $\mathcal{A}$.*

2. **Phase I:** $\mathcal{A}$ *gets access to the decryption oracle* $\mathsf{Dec}_{SK}(\cdot)$*.*

3. **Challenge:** $\mathcal{A}$ *outputs a plaintext* $\mathsf{msg}$ *and a transformation* $T \in \mathcal{T}$*. We privately flip a coin* $b \leftarrow \{0,1\}$*.*

   - *If* $b = 0$*, we give the adversary* $\zeta_0^* \leftarrow \mathsf{Enc}_{PK}(\mathsf{msg})$ *and* $\zeta_1^* \leftarrow \mathsf{Enc}_{PK}(h(\mathsf{msg}))$*.*
   - *If* $b = 1$*, we give the adversary* $\zeta_0^* \leftarrow \mathsf{Enc}_{PK}(\mathsf{msg})$ *and* $\zeta_1^* \leftarrow \mathsf{CTrans}(\zeta_0^*, T)$*.*

4. **Phase II:** $\mathcal{A}$ *gets access to the decryption oracle* $\mathsf{Dec}_{SK}(\cdot)$*.*

5. **Output:** $\mathcal{A}$ *outputs a bit* $b'$*. The* advantage *of* $\mathcal{A}$ *in this experiment is* $\Pr[b' = b] - \frac{1}{2}$*.*

Unlike the stronger variant, weak unlinkability is implied by the perfect rerandomizing property (Definition 1). Relaxing the unlinkability property in this way corresponds to a slight relaxation of the UC functionality the scheme can realize. In the $\mathcal{F}_{\text{HMP}}^{\mathcal{T}}$ UC functionality, call a handle *adversarially influenced* if:

- it is the result of a POST or REPOST command issued by a corrupted party,
- or it is the result of a (REPOST, $\mathsf{handle}$) command, where $\mathsf{handle}$ is adversarially influenced.

An encryption scheme which is HCCA secure and only weakly unlinkably homomorphic is a secure realization of a variant of $\mathcal{F}_{\text{HMP}}^{\mathcal{T}}$, in which the adversary is notified every time an adversarially influenced handle is reposted (in the same way it is notified when its dummy handles are reposted). The proof is very similar to that of Theorem 2, except that unlinkability is only applied to handles which are not adversarially influenced.

### 7.4 Repost-test

In $\mathcal{F}_{\text{HMP}}^{\mathcal{T}}$, when an honest party Alice receives a post from Bob and then another from Carl, Alice has no way of knowing if Carl's message was derived from Bob's (via $\mathcal{F}_{\text{HMP}}^{\mathcal{T}}$'s REPOST feature), or via an independent POST command. In fact, the only time $\mathcal{F}_{\text{HMP}}^{\mathcal{T}}$ informs a recipient that a REPOST occurred is for the adversary's dummy handles.

We can easily modify our schemes and $\mathcal{F}_{\text{HMP}}^{\mathcal{T}}$ to provide such a *feature* for honest parties. We call this feature *repost-test.* In this variant of $\mathcal{F}_{\text{HMP}}^{\mathcal{T}}$, the recipient may issue an additional command (TEST, $\mathsf{handle}_1, \mathsf{handle}_2$). The functionality returns a boolean indicating whether the two handles were the result of reposting a common handle (it keeps extra book-keeping to track the ancestor of each REPOST-generated handle).

To realize this modified functionality, we start with a realization of $\mathcal{F}_{\text{HMP}}^{\mathcal{T}}$ on message space $\mathcal{M}^{n+1}$, where $\mathcal{M}$ has superpolynomial size. Suppose every $T \in \mathcal{T}$ always preserves the $(n+1)$th component of the message. Then let $\mathcal{T}'$ be the restrictions of $T \in \mathcal{T}$ to the first $n$ components.

We may then use $\mathcal{F}_{\text{HMP}}^{\mathcal{T}}$ to obtain a secure realization of $\mathcal{F}_{\text{HMP}}^{\mathcal{T}'}$ with repost-test feature in the following way: To post a message $(m_1, \ldots, m_n) \in \mathcal{M}^n$, choose a random $m_{n+1} \leftarrow \mathcal{M}$ and post $(m_1, \ldots, m_{n+1})$ to $\mathcal{F}_{\text{HMP}}^{\mathcal{T}}$. When reading a message, ignore the last component. To perform the repost-test on two handles, simply check whether the last components of their corresponding messages are equal.

### 7.5 Triviality of HCCA without Unlinkability

Without an unlinkability requirement, it is relatively trivial to construct a scheme that is HCCA-secure with respect to any space of message transformations $\mathcal{T}$. Consider modifying any CCA-secure encryption scheme by considering an additional kind of ciphertext of the form $(\zeta, T)$, where $\zeta$ is a ciphertext in the original scheme and $T$ is a description of a transformation in $\mathcal{T}$. To decrypt a ciphertext of this new form, first decrypt $\zeta$ and then if $T \in \mathcal{T}$, apply $T$ to the result. The scheme has a homomorphic transformation procedure: $\mathsf{CTrans}(\zeta, T) = (\zeta, T)$, and $\mathsf{CTrans}((\zeta, T), T') = (\zeta, T' \circ T)$.

It is not hard to see that such a scheme achieves HCCA security with respect to $\mathcal{T}$. $\mathsf{RigEnc}$ should encrypt some fixed message and use the ciphertext itself as the auxiliary information $S$. Then on input $(\zeta, T), S$, the $\mathsf{RigExtract}$ procedure should return $T$ if $T \in \mathcal{T}$ and $\zeta = S$, and return $\perp$ otherwise.

## Acknowledgments

We thank Rui Xue for helpful discussions regarding Theorem 3, as well as the feedback from the anonymous referees.

## References

[1] J. H. An, Y. Dodis, and T. Rabin. On the security of joint signature and encryption. In L. R. Knudsen, editor, *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 83–107. Springer, 2002.

[2] J. K. Andersen and E. W. Weisstein. Cunningham chain. From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/CunninghamChain.html, 2005.

[3] M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In C. Boyd, editor, *ASIACRYPT*, volume 2248 of *Lecture Notes in Computer Science*, pages 566–582. Springer, 2001.

[4] M. Bellare and A. Sahai. Non-malleable encryption: Equivalence between two notions, and an indistinguishability-based characterization. In M. J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 519–536. Springer, 1999.

[5] J. Benaloh. *Verifiable Secret-Ballot Elections.* PhD thesis, Department of Computer Science, Yale University, 1987.

[6] M. Blaze, G. Bleumer, and M. Strauss. Divertible protocols and atomic proxy cryptography. In K. Nyberg, editor, *EUROCRYPT*, volume 1403 of *Lecture Notes in Computer Science*, pages 127–144. Springer, 1998.

[7] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. Cryptology ePrint Archive, Report 2000/067, 2005.

[8] R. Canetti and J. Herzog. Universally composable symbolic analysis of mutual authentication and key-exchange protocols. In S. Halevi and T. Rabin, editors, *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pages 380–403. Springer, 2006.

[9] R. Canetti and S. Hohenberger. Chosen-ciphertext secure proxy re-encryption. In *ACM Computer and Communication Security (CCS)*, 2007.

[10] R. Canetti, H. Krawczyk, and J. B. Nielsen. Relaxing chosen-ciphertext security. In D. Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 565–582. Springer, 2003.

[11] B. Chor, N. Gilboa, and M. Naor. Private information retrieval by keywords. TR CS0917, Department of Computer Science, Technion, 1997.

[12] R. Cramer, M. K. Franklin, B. Schoenmakers, and M. Yung. Multi-autority secret-ballot elections with linear work. In *EUROCRYPT*, pages 72–83, 1996.

[13] R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In H. Krawczyk, editor, *CRYPTO*, volume 1462 of *Lecture Notes in Computer Science*. Springer, 1998.

[14] I. Damgård, N. Fazio, and A. Nicolosi. Non-interactive zero-knowledge from homomorphic encryption. In S. Halevi and T. Rabin, editors, *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pages 41–59. Springer, 2006.

[15] I. Damgård and J. B. Nielsen. Universally composable efficient multiparty computation from threshold homomorphic encryption. In D. Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 247–264. Springer, 2003.

[16] G. Danezis. Breaking four mix-related schemes based on universal re-encryption. In *Proc. Information Security Conference*. Springer-Verlag, September 2006.

[17] D. Dolev, C. Dwork, and M. Naor. Nonmalleable cryptography. *SIAM J. Comput.*, 30(2):391–437 (electronic), 2000. Preliminary version in STOC 1991.

[18] T. E. Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *CRYPTO*, pages 10–18, 1984.

[19] S. Goldwasser and S. Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, Apr. 1984. Preliminary version appeared in STOC' 82.

[20] P. Golle, M. Jakobsson, A. Juels, and P. Syverson. Universal re-encryption for mixnets. In *Proceedings of the 2004 RSA Conference, Cryptographer's track*, San Francisco, USA, February 2004.

[21] J. Groth. Rerandomizable and replayable adaptive chosen ciphertext attack secure cryptosystems. In M. Naor, editor, *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 152–170. Springer, 2004.

[22] M. Hirt and K. Sako. Efficient receipt-free voting based on homomorphic encryption. In *EUROCRYPT*, pages 539–556, 2000.

[23] Y. Ishai, E. Kushilevitz, and R. Ostrovsky. Sufficient conditions for collision-resistant hashing. In J. Kilian, editor, *TCC*, volume 3378 of *Lecture Notes in Computer Science*, pages 445–456. Springer, 2005.

[24] M. J. Jurik. *Extensions to the Paillier Cryptosystem with Applications to Cryptological Protocols*. PhD thesis, BRICS, 2003.

[25] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *STOC*, pages 427–437. ACM, 1990.

[26] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT*, pages 223–238, 1999.

[27] A. Patil. On symbolic analysis of cryptographic protocols. Master's thesis, Massachusetts Institute of Technology, 2005.

[28] B. Pfitzmann and M. Waidner. Composition and integrity preservation of secure reactive systems. In *ACM Conference on Computer and Communications Security*, pages 245–254, 2000.

[29] M. Prabhakaran and M. Rosulek. Rerandomizable RCCA encryption. In A. Menezes, editor, *CRYPTO*, volume 4622 of *Lecture Notes in Computer Science*. Springer, 2007. To appear.

[30] F. H. Project. Anonymity bibliography. `http://freehaven.net/anonbib/`, 2006.

[31] C. Rackoff and D. R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In J. Feigenbaum, editor, *CRYPTO*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444. Springer, 1991.

[32] K. Sako and J. Kilian. Secure voting using partially compatible homomorphisms. In Y. Desmedt, editor, *CRYPTO*, volume 839 of *Lecture Notes in Computer Science*, pages 411–424. Springer, 1994.

[33] T. Sander, A. Young, and M. Yung. Non-interactive cryptocomputing for $\mathrm{NC}^1$. In *FOCS*, pages 554–567, 1999.

[34] V. Shoup. A proposal for an ISO standard for public key encryption. Cryptology ePrint Archive, Report 2001/112, 2001. `http://eprint.iacr.org/`.

[35] D. X. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. In *IEEE Symposium on Security and Privacy*, pages 44–55, 2000.

[36] D. Wikström. A note on the malleability of the El Gamal cryptosystem. In A. Menezes and P. Sarkar, editors, *INDOCRYPT*, volume 2551 of *Lecture Notes in Computer Science*, pages 176–184. Springer, 2002.

# A Security Proof

**Theorem 4.** *Our construction (Section 5) satisfies the correctness properties for a homomorphic encryption scheme, is unlinkably homomorphic and is HCCA-secure, under the DDH assumption in $\mathbb{G}$ and $\widehat{\mathbb{G}}$.*

An overview of the (somewhat lengthy) proof is provided below. Then the full details of the proof are carried out in the following sections.

*Proof Sketch.* To show HCCA security, we must demonstrate appropriate $\mathsf{RigEnc}$ and $\mathsf{RigExtract}$ procedures. Our $\mathsf{RigEnc}$ encrypts a fixed dummy plaintext and uses a randomly chosen $\mu$ value instead of one derived from the plaintext. Our $\mathsf{RigExtract}$ similarly checks the integrity of the ciphertext using the same random $\mu$ value, and checks that the dummy plaintext was altered by an allowed transformation.

To show the suitability of these procedures in the HCCA experiment, we first describe an alternate encryption procedure which is implemented using the private key instead of the public key. When this procedure is used in place of $\mathsf{Enc}$ or $\mathsf{RigEnc}$ to generate the challenge ciphertext $\zeta^*$ in the HCCA security experiment, it follows from the DDH assumption that the difference is indistinguishable to any adversary. The ciphertexts produced by this alternate procedure are information-theoretically independent of the secret coin flip in the HCCA experiment, as well as some internal randomness used to generate the ciphertext.

Next, we show that given a fixed view of an adversary in the HCCA experiment, any ciphertext which is not in the support of $\mathsf{Enc}_{PK}(\cdot)$ or $\mathsf{CTrans}(\zeta^*, \cdot)$ is rejected by the decryption oracles $\mathsf{Dec}$ and $\mathsf{RigDec}$ (i.e, they output $\perp$ for such ciphertexts) with overwhelming probability over the remaining randomness in the experiment (which is independent of the adversary's view). This is the most delicate part of our proof, and it roughly follows that of [29]. It uses a linear-algebraic characterization of our scheme, and relies on the fact that certain quantities in the challenge ciphertext are distributed independently of the adversary's view. We also show an analagous statement for the $\mathsf{GRigExtract}$ oracles.

From the previous observation, we may replace the $\mathsf{Dec}$, $\mathsf{RigDec}$, and $\mathsf{GRigExtract}$ oracles (which use the secret key) with oracles that can be implemented using only information that is public to the adversary (e.g, the public key and challenge ciphertext). These oracles are computationally unbounded, as they exhaustively search the supports of $\mathsf{Enc}_{PK}(\cdot)$ and $\mathsf{CTrans}(\zeta^*, \cdot)$. Only with negligible probability do these alternate oracles give an answer which disagrees with the original oracles.

Finally, we conclude that with these two modifications — alternate encryption and decryption procedures — the adversary's entire view (the public key, challenge ciphertext and responses from the oracles) in the HCCA security experiment is independent of the secret bit $b$, and so the adversary's advantage is zero. Furthermore, this modified experiment is indistinguishable to the original experiment for any PPT adversary, so the HCCA security claim follows.

The correctness and unlinkable homomorphism properties are a direct consequence of the lemmas needed to prove the HCCA security. □

## A.1 Rigged Encryption and Extraction

To show HCCA security, we must demonstrate $\mathsf{RigEnc}$ and $\mathsf{RigExtract}$ procedures. First, we factor out some subroutines that are common to the "rigged" and non-rigged encryption procedures:

**Ciphertext generation.** $\mathsf{GenCiph}_{PK}((m_1, \dots, m_n), \mu)$: Pick random $x \in \mathbb{Z}_p$, $y \in \mathbb{Z}_p^*$ and random $u \in \widehat{\mathbb{G}}$, and output

$$\begin{array}{llllllll} g_1^{(x+z_1)u}, & \dots, & g_4^{(x+z_4)u}, & m_1 C_1^x, & \dots, & m_n C_n^x, & (DE^\mu)^x; & \\ g_1^{yu}, & \dots, & g_4^{yu}, & C_1^y, & \dots, & C_n^y, & (DE^\mu)^y; & \mathsf{MEnc}_{\widehat{PK}}(u) \end{array}$$

**Deriving purported plaintext.** $\mathsf{PurpMsg}_{SK}(\zeta, u)$: Strip off $u$ and $\vec{z}$ from the exponents as follows: For $j = 1, \dots, 4$: set $\overline{X}_j = X_j^{1/u} g_j^{-z_j}$ and $\overline{Y}_j = Y_j^{1/u}$. Output $(m_1, \dots, m_n)$, where $m_i = C_{X,i} / \prod_{j=1}^4 \overline{X}_j^{c_{i,j}}$.

**Checking ciphertext integrity.** $\mathsf{Integrity}_{SK}(\zeta, u, \mu)$: Strip off $u$ and $\vec{z}$ from the exponents as follows: For $j = 1, \dots, 4$: set $\overline{X}_j = X_j^{1/u} g_j^{-z_j}$ and $\overline{Y}_j = Y_j^{1/u}$. If $\overline{Y}_1 = \cdots = \overline{Y}_4 = 1$ (the identity element in $\mathbb{G}$), output 0. Otherwise, check the following constraints:

$$C_{Y,i} \stackrel{?}{=} \prod_{j=1}^4 \overline{Y}_j^{c_{i,j}} \text{ (for each } i \in [n]); \qquad P_X \stackrel{?}{=} \prod_{j=1}^4 \overline{X}_j^{d_j + \mu e_j}; \qquad P_Y \stackrel{?}{=} \prod_{j=1}^4 \overline{Y}_j^{d_j + \mu e_j}$$

If any fail, output 0, otherwise output 1.

We can view the scheme's $\mathsf{Enc}$ and $\mathsf{Dec}$ routines as using these subroutines:

$\mathsf{Enc}_{PK}(m_1, \dots, m_n)$**:** Output $\mathsf{GenCiph}_{PK}((m_1, \dots, m_n), \mathsf{H}(f(m_1, \dots, m_n)))$.

$\mathsf{Dec}_{SK}(\zeta)$**:** Compute $u \leftarrow \mathsf{MDec}_{\widehat{SK}}(U)$. If $u = \perp$, output $\perp$. Otherwise, set $(m_1, \dots, m_n) = \mathsf{PurpMsg}_{SK}(\zeta, u)$. If $\mathsf{Integrity}_{SK}(\zeta, u, \mathsf{H}(f(m_1, \dots, m_n))) = 1$, output $(m_1, \dots, m_n)$; otherwise output $\perp$.

Now, we define our $\mathsf{RigEnc}$ and $\mathsf{RigExtract}$ features:

$\mathsf{RigEnc}_{PK}()$**:** Pick random $\mu \leftarrow \mathbb{Z}_p$. Generate $\zeta \leftarrow \mathsf{GenCiph}_{PK}((1, \dots, 1), \mu)$, and output $(\zeta, S = \mu)$.

$\mathsf{RigExtract}_{SK}(\zeta, S)$**:** Compute $u \leftarrow \mathsf{MDec}_{\widehat{SK}}(U)$. If $u = \perp$, output $\perp$. Otherwise, set $(m_1, \dots, m_n) = \mathsf{PurpMsg}_{SK}(\zeta, u)$. If $\mathsf{Integrity}_{SK}(\zeta, u, S) = 1$ and $T_{(m_1, \dots, m_n)}$ is an allowed transformation, then output $T_{(m_1, \dots, m_n)}$; otherwise output $\perp$.

## A.2 Encryption and Decryption as Linear Algebra

In this section we characterize our construction using linear algebra, which will be useful in the security proof.

**Public key constraints.** First we examine what information is revealed to the adversary about the private key by the public key.

The following constaints relate the private keys to public keys (the first equation is in the field of order $q$, and the second is in the field of order $p$):

$$\begin{bmatrix} \vec{1} & 0 \\ 0 & \vec{1} \end{bmatrix} \begin{bmatrix} \hat{G} & 0 \\ 0 & \hat{G} \end{bmatrix} \begin{bmatrix} \vec{a}^\top \\ \vec{b}^\top \end{bmatrix} = \begin{bmatrix} \log A \\ \log B \end{bmatrix}, \text{ where } \hat{G} = \begin{bmatrix} \log \widehat{g}_1 & 0 \\ 0 & \log \widehat{g}_2 \end{bmatrix}$$

$$\begin{bmatrix} \vec{1} & & \\ & \ddots & \\ & & \vec{1} \end{bmatrix} \begin{bmatrix} G & & \\ & \ddots & \\ & & G \end{bmatrix} \begin{bmatrix} \vec{c}_1^\top \\ \vdots \\ \vec{c}_n^\top \\ \vec{d}^\top \\ \vec{e}^\top \end{bmatrix} = \begin{bmatrix} \log C_1 \\ \vdots \\ \log C_n \\ \log D \\ \log E \end{bmatrix}, \text{ where } G = \begin{bmatrix} \log g_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \log g_4 \end{bmatrix} \tag{1}$$

We call these constraints the *public-key constraints.*

**Definition 6.** *Let $U = (\vec{V}, A_V, B_V, \vec{W}, A_W, B_W)$ be a DSME ciphertext. The two* DSME strands *of $U$ with respect to a public key $(\widehat{g}_1, \widehat{g}_2, A, B)$ are:*

$$\vec{v} = (v_1, v_2), \text{ where } v_j = \log_{\widehat{g}_j} V_j$$
$$\vec{w} = (w_1, w_2), \text{ where } w_j = \log_{\widehat{g}_j} W_j$$

Observe that applying $\mathsf{MCTrans}(U, T_\sigma)$ gives a ciphertext whose two strands are $\vec{v} + r\vec{w}$ and $s\vec{w}$, for random $r \in \mathbb{Z}_q, s \in \mathbb{Z}_q^*$. In ciphertexts generated by $\mathsf{MEnc}$, both strands are scalar multiples of the all-ones vector.

For DSCS ciphertexts, we define a similar notion of strands. However, in a DSCS ciphertext, the first strand is "masked" by $u$ and $z_i$'s, and the second strand is masked by $u$.

**Definition 7.** *Let $\zeta = (\vec{X}, \vec{C}_X, P_X; \vec{Y}, \vec{C}_Y, P_Y; U)$ be a DSCS ciphertext. The* DSCS strands *of $\zeta$ with respect to a public key $(g_1, \ldots, g_4, \ldots)$ and a value $u \in \widehat{\mathbb{G}}$ are:*

$$\vec{x} = (x_1, \ldots, x_4), \text{ where } x_i = (\log_{g_i} X_i)/u - z_i$$
$$\vec{y} = (y_1, \ldots, y_4), \text{ where } y_i = (\log_{g_i} Y_i)/u$$

As above, applying $\mathsf{CTrans}(\zeta, T_{\vec{\tau}})$ gives a ciphertext whose two strands are $\vec{x} + r\vec{y}$ and $s\vec{y}$, for random $r \in \mathbb{Z}_p, s \in \mathbb{Z}_p^*$. In ciphertexts generated by $\mathsf{Enc}$, both strands are scalar multiples of the all-ones vector.

**Decryption constraints.** Let $\widehat{SK} = (\vec{a}, \vec{b})$ be a DSME private key, let $U = (\vec{V}, A_V, B_V, \vec{W}, A_W, B_W)$ be a DSME ciphertext, and let $\vec{v}, \vec{w}$ be its two strands with respect to the corresponding public key. Then $\mathsf{MDec}_{\widehat{SK}}(U) = u \neq \perp$ if and only if $\vec{w}$ is a nonzero vector and the following constraints hold in the field of order $q$:

$$\begin{bmatrix} \vec{v} & 0 \\ \vec{w} & 0 \\ 0 & \vec{v} \\ 0 & \vec{w} \end{bmatrix} \begin{bmatrix} \hat{G} & 0 \\ 0 & \hat{G} \end{bmatrix} \begin{bmatrix} \vec{a}^\top \\ \vec{b}^\top \end{bmatrix} = \begin{bmatrix} \log(A_V/u) \\ \log A_W \\ \log B_V \\ \log B_W \end{bmatrix} \tag{2}$$

The logarithms are with respect to any fixed generator of $\widehat{\mathbb{G}}$.

Similarly, let $SK = (\widehat{SK}, \vec{c}_1, \ldots, \vec{c}_n, \vec{d}, \vec{e})$ be a DSCS private key and let $\zeta = (\vec{X}, \vec{C}_X, P_X; \vec{Y}, \vec{C}_Y, P_Y; U)$ be a DSCS ciphertext, such that $\mathsf{MDec}_{\widehat{SK}}(U) = u \neq \bot$. Let $\vec{x}$ and $\vec{y}$ denote the strands of $\zeta^*$ with respect to the public key and $u$.

Then $\mathsf{PurpMsg}_{SK}(\zeta, u) = (m_1, \ldots, m_n)$ and $\mathsf{Integrity}_{SK}(\zeta, u, \mu) = 1$ if and only if $\vec{y}$ is a nonzero vector and the following constraints hold in the field of order $p$:

$$\begin{bmatrix} \vec{x} & & & & \\ & \ddots & & 0 & 0 \\ & & \vec{x} & & \\ \vec{y} & & & & \\ & \ddots & & 0 & 0 \\ & & \vec{y} & & \\ & 0 & & \vec{x} & \mu\vec{x} \\ & 0 & & \vec{y} & \mu\vec{y} \end{bmatrix} \begin{bmatrix} G & & \\ & \ddots & \\ & & G \end{bmatrix} \begin{bmatrix} \vec{c}_1^\top \\ \vdots \\ \vec{c}_n^\top \\ \vec{d}^\top \\ \vec{e}^\top \end{bmatrix} = \begin{bmatrix} \log(C_{X,1}/m_1) \\ \vdots \\ \log(C_{X,n}/m_n) \\ \log C_{Y,1} \\ \vdots \\ \log C_{Y,n} \\ \log P_X \\ \log P_Y \end{bmatrix} \qquad (3)$$

The logarithms are with respect to any fixed generator of $\mathbb{G}$.

We call each constraint in these systems of equations the *decryption constraints*, and refer to them by the name of the ciphertext component that is involved ($A_V$, $A_W$, $P_X$, etc.).

The adversary receives ciphertexts throughout the HCCA experiment from $\mathsf{GRigEnc}$ oracles and as the challenge $\zeta^*$. These ciphertexts gives the adversary's view additional constraints on the private key.

**Observation:** Ciphertexts that are generated by $\mathsf{GenCiph}$ induce constraints that are linearly dependent on the constraints given by the public key, because their strands are scalar multiples of the appropriate all-ones vectors.

**Lemma 2.** *For all key pairs $(\widehat{PK}, \widehat{SK})$, all (purported) ciphertexts $U$, and all $U'$ in the support of $\mathsf{MCTrans}(U, T_\sigma)$, we have $\mathsf{MDec}_{\widehat{SK}}(U') = T_\sigma(\mathsf{MDec}_{\widehat{SK}}(U))$.*

*Proof.* If $\vec{v}$ and $\vec{w}$ are the two strands of $U$, then the two strands of $U'$ are $\vec{v} + r\vec{w}$ and $s\vec{w}$ for some $r \in \mathbb{Z}_q$, $s \in \mathbb{Z}_q^*$. The two strands of $U'$ are linear combinations of the strands of $U$, and the three values checked by a decryption constraint ($B_V$,$A_W$,$B_W$) are the corresponding combinations of the values from $U$. Thus, a particular decryption constraint fails on $U'$ if and only if it fails for $U$. Furthermore, as $s \neq 0$, so $\vec{w}$ is a nonzero vector if and only if $s\vec{w}$ is a nonzero vector. Finally, it is straight-forward to check that the purported plaintext of $U'$ is $\sigma$ times the purported plaintext of $U$. □

**Lemma 3.** *For all key pairs $(PK, SK)$, all (purported) ciphertexts $\zeta$, and all $\zeta'$ in the support of $\mathsf{CTrans}(\zeta, T_{\vec{\tau}})$, we have $\mathsf{Dec}_{SK}(\zeta') = T_{\vec{\tau}}(\mathsf{Dec}_{SK}(\zeta))$.*

*Proof.* First, by the above lemma, the DSME component of $\zeta$ will fail to decrypt if and only if the DSME component of $\zeta'$ fails to decrypt.

Otherwise, the two strands of $\zeta'$ (with respect to the decryption of its DSME component) are linear combinations of the strands of $\zeta$ (with respect to the decryption of its DSME component). A similar argument to above shows that a decryption check fails on $\zeta'$ if and only if the same check fails on $\zeta$; and the ratios of the purported plaintexts are $\vec{\tau}$. □

## A.3 Decisional Diffie-Hellman Assumption

We now describe a more intricate indistinguishability assumption, which is implied by the standard DDH assumption in $\mathbb{G}$ and $\widehat{\mathbb{G}}$.

First, consider the following two distributions:

- $\mathsf{DDH}(\mathbb{G}, j)$ **distribution.** Pick random elements $g_1, \ldots, g_j \leftarrow \mathbb{G}$, and pick a random $v \leftarrow \mathbb{Z}_p$, where $|\mathbb{G}| = p$. Output $(g_1, \ldots, g_j, g_1^v, \ldots, g_j^v)$.
- $\mathsf{Rand}(\mathbb{G}, j)$ **distribution.** Pick random elements $g_1, \ldots, g_j \leftarrow \mathbb{G}$, and pick random $v_1, \ldots, v_j \leftarrow \mathbb{Z}_p$, where $|\mathbb{G}| = p$. Output $(g_1, \ldots, g_j, g_1^{v_1}, \ldots, g_j^{v_j})$.

We will require distributions of this form with $j = 2$ and $j = 4$, in different groups. Note that for fixed $n$, the standard DDH assumption in $\mathbb{G}$ (which is the special case of $j = 2$) implies that the above distributions are indistinguishable. To see this, consider a hybrid distribution in which the first $k$ exponents are randomly chosen, and the remaining exponents are all equal. The standard DDH assumption is easily seen to imply that the $k$th hybrid distribution is indistinguishable from the $(k + 1)$st.

Now consider the following two “double-strand” distributions:

- $\mathsf{DS\text{-}DDH}(\mathbb{G}, j)$ **distribution.** Pick random elements $g_1, \ldots, g_j \leftarrow \mathbb{G}$, and pick random $v, w \leftarrow \mathbb{Z}_p$, where $|\mathbb{G}| = p$. Output $(g_1, \ldots, g_j, g_1^v, \ldots, g_j^v, g_1^w, \ldots, g_j^w)$.
- $\mathsf{DS\text{-}Rand}(\mathbb{G}, j)$ **distribution.** Pick random elements $g_1, \ldots, g_j \leftarrow \mathbb{G}$, and pick random $v_1, \ldots, v_j, w_1, \ldots, w_j \leftarrow \mathbb{Z}_p$, where $|\mathbb{G}| = p$. Output $(g_1, \ldots, g_j, g_1^{v_1}, \ldots, g_j^{v_j}, g_1^{w_1}, \ldots, g_j^{w_j})$.

Again, a simple hybrid argument shows that if the $\mathsf{DDH}(\mathbb{G}, j)$ and $\mathsf{Rand}(\mathbb{G}, j)$ distributions are indistinguishable, then so are $\mathsf{DS\text{-}DDH}(\mathbb{G}, j)$ and $\mathsf{DS\text{-}Rand}(\mathbb{G}, j)$. We call elements in the support of these distributions *double-strand tuples of length j.*

Finally, our security proofs rely on the indistinguishability of the following two distributions:

- Pick $K_0 \leftarrow \mathsf{DS\text{-}DDH}(\mathbb{G}, 4)$, and pick $K_1 \leftarrow \mathsf{DDH}(\widehat{\mathbb{G}}, 2)$. Output $(K_0, K_1)$.
- Pick $K_0 \leftarrow \mathsf{DS\text{-}Rand}(\mathbb{G}, 4)$, and pick $K_1 \leftarrow \mathsf{Rand}(\widehat{\mathbb{G}}, 2)$. Output $(K_0, K_1)$.

A final hybrid argument shows that if $\mathsf{DS\text{-}DDH}(\mathbb{G}, 4)$ and $\mathsf{DS\text{-}Rand}(\mathbb{G}, 4)$ are indistinguishable, and $\mathsf{DDH}(\widehat{\mathbb{G}}, 2)$ and $\mathsf{Rand}(\widehat{\mathbb{G}}, 2)$ are also indistinguishable, then the above two distributions are indistinguishable.

## A.4 The Alternate Encryption Procedure

We now describe the alternate method of generating ciphertexts $\mathsf{AltGenCiph}$. As a component, it uses $\mathsf{AltMEnc}$, an alternate encryption procedure for the DSME scheme. Both of these procedures use the secret keys instead of the public keys to generate ciphertexts.

**DSME alternate encryption:** $\mathsf{AltMEnc}_{\widehat{SK}}(u)$**.**

- Pick random $v_1, v_2 \in \mathbb{Z}_q$ and $w \in \mathbb{Z}_q^*$. For $j = 1, 2$ let $V_j = \widehat{g}_j^{v_j}$ and $W_j = \widehat{g}_j^{w}$ (alternatively, in the analysis below we also consider $V_1, V_2$ as inputs instead).

- Output $(\vec{V}, A_V, B_V, \vec{W}, A_W, B_W)$, where

$$\begin{array}{lll} \vec{V} = (V_1, V_2) & A_V = u \cdot \prod_{j=1}^{2} V_j^{a_j} & B_V = \prod_{j=1}^{2} V_j^{b_j} \\ \vec{W} = (W_1, W_2) & A_W = \prod_{j=1}^{2} W_j^{a_j} & B_W = \prod_{j=1}^{2} W_j^{b_j} \end{array}$$

**DSCS alternate ciphertexts:** $\mathsf{AltGenCiph}_{SK}((m_1, \ldots, m_n), \mu)$**.**

- Pick random $x_1, \ldots, x_4, y_1, \ldots, y_4 \in \mathbb{Z}_p$. For $j = 1, \ldots, 4$, set $\overline{X}_j = g_j^{x_j}$ and $\overline{Y}_j = g_j^{y_j}$, (alternatively, in the analysis below we also consider $\overline{X}_j, \overline{Y}_j$ as inputs instead).
- Pick random $u \in \widehat{\mathbb{G}}$, set $U \leftarrow \mathsf{AltMEnc}_{\widehat{SK}}(u)$, Compute:

$$\begin{array}{lll} X_j = (\overline{X}_j g_j^{z_j})^u; & C_{X,i} = m_i \prod_{j=1}^{4} \overline{X}_j^{c_{i,j}}; & P_X = \prod_{j=1}^{4} \overline{X}_j^{d_j + \mu e_j}; \\ Y_j = \overline{Y}_j^u; & C_{Y,i} = \prod_{j=1}^{4} \overline{Y}_j^{c_{i,j}}; & P_Y = \prod_{j=1}^{4} \overline{Y}_j^{d_j + \mu e_j}; \end{array}$$

- Finally, output $\zeta = (\vec{X}, \vec{C}_X, P_X; \vec{Y}, \vec{C}_Y, P_Y; U)$

These alternate encryption procedures differ from the normal encryption procedures in that they generate ciphertexts whose decryption constraints are not linearly dependent on the public key constraints. The DSME alternate encryption generates a ciphertext whose first strand is random, and the DSCS alternate encryption generates a ciphertext whose two strands are both random. The remainder of the ciphertexts are constructed using the *private keys* to ensure that the decryption constraints are satisfied.

**A hybrid HCCA experiment.** Consider a variant of the HCCA experiment, where the challenge ciphertext is generated using $\mathsf{AltGenCiph}$. That is, in the challenge phase of the experiment, the implicit call to $\mathsf{GenCiph}_{PK}$ is replaced with an identical call to $\mathsf{AltGenCiph}_{SK}$.

**Lemma 4.** *In the hybrid HCCA experiment (where $\zeta^*$ is generated using* $\mathsf{AltGenCiph}$*), conditioned on a negligible-probability event not occuring, $\zeta^*$ is distributed independently of the randomness $u$, and the bit $b$ in the experiment, even given the public key. When $b = 1$, $\zeta^*$ is also distributed independently of the random choice of $\mu$ used in* $\mathsf{RigEnc}$.

*Proof.* Given a DSME ciphertext from $\mathsf{AltMEnc}$ with first strand $\vec{v}^*$, the set $\{\vec{v}^*, \vec{1}\}$ forms a basis for the space of all DSME strands, with overwhelming probability. The adversary's view of the DSME private key $(\vec{a}, \vec{b})$ is constrained by the public key constraints in equation (1) and the decryption constraints given by $\zeta^*$ in equation (3). The constraints involving the second strand are linearly dependent on the public key constraints, as $\vec{w}$ is a scalar multiple of the all-ones vector. Combining these constraints and removing the redundant ones, we have:

$$\begin{bmatrix} \vec{1} & 0 \\ 0 & \vec{1} \\ \vec{v} & 0 \\ 0 & \vec{v} \end{bmatrix} \begin{bmatrix} \hat{G} & 0 \\ 0 & \hat{G} \end{bmatrix} \begin{bmatrix} \vec{a}^\top \\ \vec{b}^\top \end{bmatrix} = \begin{bmatrix} \log A \\ \log B \\ \log(A_V/u) \\ \log B_V \end{bmatrix},$$

The matrix on the left-hand side of this equation is nonsingular.

Similarly, in a DSCS ciphertext from $\mathsf{AltGenCiph}$, with overwhelming probability we have that for *every* $u \in \widehat{\mathbb{G}}$, $\{\vec{x}, \vec{y}, \vec{1}, \vec{z}\}$ form a basis for the space of all DSCS strands, where $\vec{x}$ and $\vec{y}$ are the strands of the challenge ciphertext with respect to $u$. Then the adversary's view of the private key is constrained as follows:

$$\begin{bmatrix}
\vec{1} & & & & \\
& \ddots & & & \\
& & & & \vec{1} \\
\vec{x} & & & & \\
& \ddots & & & \\
& & \vec{x} & & \\
\vec{y} & & & & \\
& \ddots & & & \\
& & \vec{y} & & \\
& & & \vec{x} & \mu\vec{x} \\
& & & \vec{y} & \mu\vec{y}
\end{bmatrix}
\begin{bmatrix} G & & \\ & \ddots & \\ & & G \end{bmatrix}
\begin{bmatrix} \vec{c}_1^\top \\ \vdots \\ \vec{c}_n^\top \\ \vec{d}^\top \\ \vec{e}^\top \end{bmatrix}
=
\begin{bmatrix} \log C_1 \\ \vdots \\ \log C_n \\ \log D \\ \log E \\ \log(C_{X,1}/m_1) \\ \vdots \\ \log(C_{X,n}/m_n) \\ \log C_{Y,1} \\ \vdots \\ \log C_{Y,n} \\ \log P_X \\ \log P_Y \end{bmatrix}$$

Here $(m_1, \ldots, m_n)$ and $\mu$ denote the inputs to $\mathsf{AltGenCiph}$.

Note that when $\{\vec{1}, \vec{x}, \vec{y}\}$ are linearly independent, the matrix on the left-hand side of this equation is nonsingular, for *every* $\mu \in \mathbb{Z}_p$.

Now fix a public key and challenge ciphertext (one that avoids the event that $\{\vec{1}, \vec{x}, \vec{y}\}$ is not linearly independent). When $b = 0$ in the experiment, substitute the appropriate $m_i$ and $\mu$ values in the previous system of equations. Then for every choice of $u \in \widehat{\mathbb{G}}$, there are an equal number of solutions for the private keys in this system of equations, as the matrix is nonsingular.

When $b = 1$ in the experiment, substitute $(m_1, \ldots, m_n) = (1, \ldots, 1)$. Then for every choice of $u \in \widehat{\mathbb{G}}$ and $\mu \in \mathbb{Z}_p$, there are an equal number of solutions for the private keys, as the matrix is nonsingular.

Since the underlying randomness in the HCCA experiment is the choice of these private keys, the choices of $u$, $\mu$, and $b$ are independent of each other and of the adversary's view in the experiment. □

**Lemma 5.** *For every PPT adversary, its advantage in the HCCA experiment is negligibly close to its advantage in the hybrid HCCA experiment (when the challenge ciphertext is generated using* $\mathsf{AltGenCiph}$*), if the DDH assumption holds in* $\mathbb{G}$ *and* $\widehat{\mathbb{G}}$*.*

*Proof.* If the DDH assumption holds for $\widehat{\mathbb{G}}$ and $\mathbb{G}$, then two the distributions described in Section A.3 are computationally indistinguishable.

Now consider a simulation of the HCCA experiment, where the input is from one of the above distributions. Let $(\widehat{g}_1, \widehat{g}_2, V_1, V_2)$ be the sample from either $\mathsf{DDH}(\widehat{\mathbb{G}}, 2)$ or $\mathsf{Rand}(\widehat{\mathbb{G}}, 2)$. Set $(\widehat{g}_1, \widehat{g}_2)$ as the corresponding part of the DSME public key, and generate the remainder of the keypair $(\vec{a}, \vec{b})$ honestly. To simulate the encryption of $u^*$ from the challenge ciphertext with this keypair, use $\mathsf{AltMEnc}$ with the input values $V_1, V_2$.

Similarly, let $(g_1, \ldots, g_4, \overline{X}_1, \ldots, \overline{X}_4, \overline{Y}_1, \ldots, \overline{Y}_4)$ be the sample from either $\mathsf{DS\text{-}DDH}(\mathbb{G}, 4)$ or $\mathsf{DS\text{-}Rand}(\mathbb{G}, 4)$. Set $(g_1, \ldots, g_4)$ as the corresponding part of the DSCS public key and generate the remainder of the DSCS keypairs $(\vec{c}_i,\ \vec{d},\ \vec{e})$ honestly. To simulate the encryption of the challenge ciphertext, use AltGenCiph and AltMEnc with the private keys and input values $\overline{X}_1, \dots, \overline{X}_4, \overline{Y}_1, \dots, \overline{Y}_4$.

It is straight-forward to check that when the input is sampled from the first distribution (i.e, the 2 tuples come from the appropriate DDH distributions), the ciphertext is distributed statistically close to a "normal" encryption from GenCiph and MEnc (the distribution is identical when conditioned to avoid the negligible-probability event that $\overline{Y}_1 = \cdots = \overline{Y}_4 = 1$). If the input is sampled from the second distribution (i.e, the 2 tuples comes from the appropriate random distributions), then the ciphertext is distributed identically as an encryption from AltGenCiph.

The rest of this simulation of the HCCA experiment can be implemented in polynomial time. Thus, the outcomes of the two simulations must not differ by more than a negligible amount. □

## A.5 Decryption Queries

In this section, we argue that with overwhelming probability, the only ciphertexts accepted by the decryption oracles in the HCCA experiment are ciphertexts of the "expected" form (from the supports of Enc or CTrans).

In this section, we let $\zeta^*$ denote the challenge ciphertext in the hybrid HCCA experiment, which was generated via AltGenCiph.

Our arguments in this section generally follow the same structure. Fix a set of constraints induced by a public key and challenge ciphertext $\zeta^*$ in the hybrid HCCA experiment. Call a query to the Dec, GRigExtract, or RigDec oracle *bad* if that oracle rejects (outputs $\perp$) for an overwhelming fraction of private keys which are consistent with those constraints.

We show that any ciphertext $\zeta$ not of the "expected" form is such a bad ciphertext, while on the other hand, ciphertexts which are of the expected form are actually rejected by none of the consistent private keys (and all private keys yield the same response from the oracle).

The following lemma establishes the significance of classifying ciphertexts in this way.

**Lemma 6.** *With overwhelming probability,* all *bad queries are rejected in the HCCA experiment.*

*Proof.* By definition, the response from the oracle for a non-bad query does not introduce any new constraints on the private keys, as they all yield the same oracle response.

Consider the first bad ciphertext submitted to an oracle. At that time, from the adversary's view, the private key is distributed uniformly among all keys consistent with the constraints induced by $\zeta^*$ and the public key. Thus it is only with negligible probability that the oracle will return something other than $\perp$. Conditioned on it returning $\perp$, the adversary learns that a negligible fraction of private keys are ruled out. Let $\nu$ be a negligible upper bound for this fraction. The correct private key remains distributed among the $(1 - \nu)$ fraction of remaining keys, from the adversary's view.

By a union bound, if the adversary makes $N$ *bad* queries to this decryption oracle, at least one of them is accepted with probability at most $N\nu$. Since the adversary makes a polynomial (in the security parameter) number of queries, this probability is negligible. □

The simplest way a ciphertext can be *bad* is if it one of its decryption integrity constraints (equation (2) and equation (3)) is linearly independent of the constraints given by the public key and challenge ciphertext.

### A.5.1 DSME Decryption

**Lemma 7.** *Fix a DSME public key and challenge ciphertext $U^*$ in the hybrid HCCA experiment. Let $U$ be an additional DSME ciphtertext. Suppose $u^*$ and $u$ are the purported plaintexts of $U^*$ and $U$, respectively, as computed by $\mathsf{MDec}_{\widehat{SK}}$. Then there exist fixed (with respect to the adversary's view) values $\pi = \pi(U)$ and $\sigma = \sigma(U)$ such that $u = \sigma(u^*)^\pi$.*

Note that even though in the hybrid HCCA experiment, the value of $u^*$ is independent of the adversary's view, the values $\pi$ and $\sigma$ are fixed.

*Proof.* Let $\vec{v}^*$ be the first strand of $U^*$, and let $\vec{v}$ be the first strand of $U$. We may unambiguously express $\vec{v} = \pi\vec{v}^* + \epsilon\vec{1}$ for some $\pi, \epsilon$. Then $\pi$ and $\sigma = A_V/(A_V^*)^\pi A^\epsilon$ are the values desired in the statement of the lemma.

The purported ciphertext of $U$ is:

$$u = \frac{A_V}{\prod_{j=1}^2 V_j^{a_j}} = \frac{A_V}{\left[\prod_{j=1}^2 (V_j^*)^{a_j}\right]^\pi \left[\prod_{j=1}^2 (\widehat{g}_j^*)^{a_j}\right]^\epsilon} = \frac{A_V}{\left[A_V^*/u^*\right]^\pi A^\epsilon} = \sigma(u^*)^\pi \qquad \square$$

**Lemma 8.** *If the second strand of $U$ is not a (nonzero) scalar multiple of $(1,1)$, then $\mathsf{MDec}_{SK}(U) = \perp$ for all but a negligible fraction of private keys consistent with the adversary's view.*

*Proof.* Let $\vec{w}$ be the second strand of $U$, and $\vec{v}^*$ the first strand of $U^*$. If $\vec{w}$ is not in the span of $\vec{1}$, then $\vec{w} = \alpha\vec{v}^* + \beta\vec{1}$ for some $\alpha \neq 0$. The following constraint is checked while decrypting $U$:

$$1 \overset{?}{=} \frac{A_W}{\prod_{j=1}^2 W_j^{a_j}} = \frac{A_W}{\left[\prod_{j=1}^2 (V_j^*)^{a_j}\right]^\alpha \left[\prod_{j=1}^2 \widehat{g}_j^{a_j}\right]^\beta} = \frac{A_W}{\left[A_V^*/u^*\right]^\alpha A^\beta}$$

The value of $u^*$ is independent of the adversary's view and distributed uniformly in $\widehat{\mathbb{G}}$, and thus so is $(u^*)^\alpha$. Thus equality holds only with negligible probability. Also, if $\vec{w}$ is the zero vector, then $U$ is explicitly rejected by $\mathsf{MDec}$. □

**Lemma 9.** *Let $U$ be a DSME ciphertext, with $\pi, \sigma$ as above, and suppose $\mathsf{MDec}(U) \neq \perp$ for a nonnegligible fraction of private keys consistent with the adversary's view. Then*

$$\pi = 0 \implies U \textit{ is in the support of } \mathsf{MEnc}_{\widehat{PK}}(\sigma)$$
$$\pi = 1 \implies U \textit{ is in the support of } \mathsf{MCTrans}(U^*, T_\sigma)$$

*Proof.* By the previous lemma, the second strand of $U$ must be a nonzero multiple of $\vec{1}$.

If $\pi = 0$, then the first strand of $U$ is also a multiple of $\vec{1}$. Say, $\vec{v} = v\vec{1}$ and $\vec{w} = w\vec{1}$, where $w \neq 0$. It is trivial to check that $U$ decrypts to $\sigma$ with nonnegligible probability only if $U = \mathsf{MEnc}_{\widehat{PK}}(\sigma; v, w)$.

If $\pi = 1$, then say $\vec{v} = \vec{v}^* + \beta\vec{1} = \vec{v}^* + s(\vec{w}^*)$ and $\vec{w} = \gamma\vec{1} = t(\vec{w}^*)$, for some $s \in \mathbb{Z}_q$, $t \in \mathbb{Z}_q^*$, since $\vec{w}^*$ (the second strand of $U^*$) is a nonzero scalar multiple of $\vec{1}$. Then it is trivial to check that $U$ decrypts to $\sigma u^*$ only if $U = \mathsf{MCTrans}(U^*, T_\sigma; s, t)$. □

### A.5.2 DSCS Decryption

**Lemma 10.** *Let $(\zeta, \mu)$ be an input to $\mathsf{Integrity}_{SK}$. Then it is a bad query unless there exists $\sigma \in \widehat{\mathbb{G}}$ such that one of the following cases holds:*

- *$U$ is in the support of $\mathsf{MEnc}_{\widehat{PK}}(\sigma)$; and there exists $x \in \mathbb{Z}_p, y \in \mathbb{Z}_p^*$ such that $X_j = g_j^{(x+z_j)\sigma}$ and $Y_j = g_j^{y\sigma}$, for $j = 1, \dots, 4$.*
- *$U$ is in the support of $\mathsf{MCTrans}(U^*, T_\sigma)$; and there exists $s \in \mathbb{Z}_p, t \in \mathbb{Z}_p^*$ such that $X_j = (X_j^*(Y_j^*)^s)^\sigma$ and $Y_j = (Y_j^*)^{t\sigma}$, for $j = 1, \dots, 4$; and $\mu = \mu^*$.*

We emphasize the similarity between these forms of ciphertexts and those produced by $\mathsf{Enc}$ and $\mathsf{CTrans}$.

*Proof.* As parts of the challenge ciphertext $\zeta^*$, the adversary is given the values: $X_j^* = g_j^{(x_j^*+z_j)u^*}$ and $Y_j^* = g_j^{y_j^* u^*}$, for some $u^*$ corresponding to the decryption of $U^*$ under $\widehat{SK}$. These values $\vec{X}$ and $\vec{Y}$ are fixed, even though the value of $u^*$ is distributed independently of the adversary's view.

Similarly, when submitting a ciphertext $\zeta$ to an oracle, the adversary supplies the values: $X_j = g_j^{(x_j+z_j)u}$ and $Y_j = g_j^{y_j u}$ for some $u$, where $\vec{x}$ and $\vec{y}$ are the strands of the ciphertext with respect to $u$, and $u$ is related to $u^*$ via $u = \sigma(u^*)^\pi$.

With overwhelming probability in the HCCA experiment, these fixed vectors $\{(\vec{x}^*+\vec{z})u^*, \vec{y}^* u^*, \vec{z}, \vec{1}\}$ span the space of all strands. Thus we can write the following unique linear combination:

$$(\vec{x} + \vec{z})u = \alpha\Big((\vec{x}^* + \vec{z})u^*\Big) + \beta(\vec{y}^* u^*) + \gamma\vec{1} + \delta\vec{z} \tag{4}$$

$$\vec{y}u = \alpha'\Big((\vec{x}^* + \vec{z})u^*\Big) + \beta'(\vec{y}^* u^*) + \gamma'\vec{1} + \delta'\vec{z} \tag{5}$$

Note that the coefficients of this linear combination are fixed, independent of the randomness in $u^*$. Our analysis proceeds by showing that if these coefficients are not fixed in a particular way, then the ciphertext would be rejected by $\mathsf{Dec}$ with overwhelming probability over the remaining randomness in $u^*$ and the private key.

Let $\vec{X}$ be as above, and consider the decryption constraint involving the $P_X$ component of the ciphertext. Suppose the constraint holds for a nonnegligible fraction of consistent private keys. The constraint that is checked by $\mathsf{Integrity}$ is of the following form:

$$\begin{bmatrix} \vec{x} & \mu\vec{x} \end{bmatrix} \begin{bmatrix} G & \\ & G \end{bmatrix} \begin{bmatrix} \vec{d}^\top \\ \vec{e}^\top \end{bmatrix} \overset{?}{=} \begin{bmatrix} \log P_X \end{bmatrix}$$

The public key and challenge ciphertext constrain $\vec{d}$ and $\vec{e}$ as follows:

$$\begin{bmatrix} \vec{1} & \\ & \vec{1} \\ \vec{x}^* & \mu^*\vec{x}^* \\ \vec{y}^* & \mu^*\vec{y}^* \end{bmatrix} \begin{bmatrix} G & \\ & G \end{bmatrix} \begin{bmatrix} \vec{d}^\top \\ \vec{e}^\top \end{bmatrix} = \begin{bmatrix} \log D \\ \log E \\ \log P_X^* \\ \log P_Y^* \end{bmatrix}$$

We must have that $[\vec{x}\ \mu\vec{x}]$ is a linear combination of the above set of constraints with nonnegligible probability (over $u^*$). Furthermore, the coefficients of that linear combination must be fixed with nonnegligible probability over $u^*$, otherwise the "correct" value of the constraint will be distributed randomly in a superpolynomial-size domain as $u^*$ varies.

Solving for $\vec{x}$ in the first equation and substituting, we have:

$$[\vec{x}\ \mu\vec{x}] = \frac{\gamma}{u}[\vec{1}\ \mu\vec{1}] + \frac{u^*}{u}\Big(\alpha[\vec{x}^*\ \mu\vec{x}^*] + \beta[\vec{y}^*\ \mu\vec{y}^*]\Big) + \left(\alpha\frac{u^*}{u} + \frac{\delta}{u} - 1\right)[\vec{z}\ \mu\vec{z}]$$

Let $\pi = \pi(U)$ and $\sigma = \sigma(U)$. We consider the following cases:

- If $\pi = 0$: Then $u = \sigma$ (independent of $u^*$) while $u^*/u$ is distributed uniformly over $\widehat{\mathbb{G}}$. We must have $\alpha = \beta = 0$, otherwise the coefficients of $[\vec{x}^*\ \mu\vec{x}^*]$ and $[\vec{y}^*\ \mu\vec{y}^*]$ are distributed randomly with $u^*/u$. Furthermore, observe that $[\vec{z}\ \mu\vec{z}]$ is linearly independent of the constraints on the adversary's view for any $\mu$, since $\vec{z}$ is linearly independent of $\{\vec{1}, \vec{x}^*, \vec{y}^*\}$. Thus, its coefficient must be zero with nonnegligible probability. This happens only when $\delta = \sigma$.

  Combining everything, we must have $X_j = g_j^{(x+z_i)\sigma}$ for some $x \in \mathbb{Z}_p$.

- If $\pi = 1$: Then $u^*/u = 1/\sigma$ while $u$ (when appearing alone) is distributed uniformly over $\widehat{\mathbb{G}}$. We must have $\gamma = 0$, otherwise the coefficient of $[\vec{1}\ \mu\vec{1}]$ is distributed randomly with $u^*$. Again, we must have the coefficient of $[\vec{z}\ \mu\vec{z}]$ equal to zero with nonnegligible probability. Substituting, we get that $\delta = 0$ and $\alpha = \sigma$. Finally, we must have $\mu = \mu^*$, or else $[\vec{x}^*\ \mu\vec{x}^*]$ is linearly independent of $[\vec{x}^*\ \mu^*\vec{x}^*]$.

  Combining everything, we must have $X_j = (X_j^*(Y_j^*)^s)^\sigma$ for some $s \in \mathbb{Z}_p$, and that $\mu = \mu^*$.

- If $\pi \notin \{0, 1\}$. Below (when discussing the second strand), we show that the ciphertext would fail its integrity checks with overwhelming probability.

Similarly, we consider the second strand's $\vec{Y}$ as a linear combination (equation (4)). We then consider the integrity check on the $P_Y$ 1 component:

$$[\vec{y}\quad \mu\vec{y}]\, G \begin{bmatrix} \vec{d}^\top \\ \vec{e}^\top \end{bmatrix} \stackrel{?}{=} [\log P_Y]$$

The public key and challenge ciphertext constrain $\vec{d}$ and $\vec{e}$ in the following way:

$$\begin{bmatrix} \vec{1} & \\ & \vec{1} \\ \vec{x}^* & \mu^*\vec{x}^* \\ \vec{y}^* & \mu^*\vec{y}^* \end{bmatrix} \begin{bmatrix} G & \\ & G \end{bmatrix} \begin{bmatrix} \vec{d}^\top \\ \vec{e}^\top \end{bmatrix} = \begin{bmatrix} \log D \\ \log E \\ \log P_X^* \\ \log P_Y^* \end{bmatrix}$$

We rewrite $[\vec{y}\ \mu\vec{y}]$, substituting according to equation (4) to obtain:

$$[\vec{y}\ \mu\vec{y}] = \frac{\gamma'}{u}[\vec{1}\ \mu\vec{1}] + \frac{u^*}{u}\Big(\alpha'[\vec{x}^*\ \mu\vec{x}^*] + \beta'[\vec{y}^*\ \mu\vec{y}^*]\Big) + \left(\alpha'\frac{u^*}{u} + \frac{\delta'}{u}\right)[\vec{z}\ \mu\vec{z}]$$

Similar to the case of the first strand, we see that the coefficient $(\alpha' u^* + \delta')/u$ must be zero with nonnegligible probability over the randomness in $u^*$. This is only possible with $\alpha' = \delta' = 0$. We further consider 3 cases of $\pi$:

- If $\pi = 0$, then $u = \sigma$ and $u^*/u$ is uniform in $\widehat{\mathbb{G}}$. We see that the coefficient of $[\vec{y}^* \;\; \mu\vec{y}^*]$ is $\beta' u^*/u$, so we must have $\beta' = 0$. Then $\gamma' \neq 0$, since otherwise $\vec{y}$ is the all-zeroes vector, and the ciphertext would be unconditionally rejected by $\mathsf{Integrity}$.

  This implies that $Y_j = g_j^{y\sigma}$ for some $y \in \mathbb{Z}_p^*$.

- If $\pi = 1$, then $u^*/u = 1/\sigma$, while $u$ is uniform in $\widehat{\mathbb{G}}$ when appearing alone. We see that the coefficient of $[\vec{1} \;\; \mu\vec{1}]$ is $\gamma'/u$, so we must have $\gamma' = 0$. Then $\beta' \neq 0$, since otherwise $\vec{y}$ is the all-zeroes vector and the ciphertext would be rejected by $\mathsf{Integrity}$.

  This implies that $Y_j = (Y_j^*)^{t\sigma}$ for some $t \in \mathbb{Z}_p^*$.

- $\pi \notin \{0, 1\}$: First, observe that if $\mu \neq \mu^*$, then $[\vec{y}^* \;\; \mu\vec{y}^*]$ is independent of the view constraints, and we must have $\beta' = 0$. Then $\alpha' = \beta' = \delta' = 0$, and $[\vec{y} \;\; \mu\vec{y}] = \gamma'/u[\vec{1} \;\; \mu\vec{1}]$. Since $u$ is uniformly distributed in $\widehat{\mathbb{G}}$, we must have $\gamma' = 0$. But then $\vec{y}$ is the all-zeroes vector and the ciphertext is unconditionally rejected by $\mathsf{Integrity}$.

  Therefore we may assume $\mu = \mu^*$ if the ciphertext is to pass its integrity checks with nonnegligible probability. We consider the following two constraints simultaneously (simplifying via $\alpha' = \delta' = 0$):

$$\begin{aligned}
\begin{bmatrix} \log C_{Y,1} \\ \log P_Y \end{bmatrix} &\overset{?}{=} \begin{bmatrix} \vec{y} & & \\ & \vec{y} & \mu\vec{y} \end{bmatrix} \begin{bmatrix} G & & \\ & G & \\ & & G \end{bmatrix} \begin{bmatrix} \vec{c}_1^\top \\ \vec{d}^\top \\ \vec{e}^\top \end{bmatrix} \\
&= \begin{bmatrix} \gamma'/u & 0 & \beta' u^*/u & 0 \\ 0 & \gamma'/u & 0 & \beta' u^*/u \end{bmatrix} \begin{bmatrix} \vec{1} & & \\ & \vec{1} & \mu^*\vec{1} \\ \vec{y}^* & & \\ & \vec{y}^* & \mu^*\vec{y}^* \end{bmatrix} \begin{bmatrix} G & & \\ & G & \\ & & G \end{bmatrix} \begin{bmatrix} \vec{c}_1^\top \\ \vec{d}^\top \\ \vec{e}^\top \end{bmatrix} \\
&= \begin{bmatrix} 0 & \gamma'/u & 0 & \beta' u^*/u \\ \gamma'/u & 0 & \beta' u^*/u & 0 \end{bmatrix} \begin{bmatrix} \log C_1 \\ \log(DE^{\mu^*}) \\ \log C_{Y,1}^* \\ \log P_Y^* \end{bmatrix}
\end{aligned}$$

  If we multiply through both of these constraints and substitute according to $u = \sigma(u^*)^\pi$, we get the following two polynomials in $u^*$, which must be simultaneously zero with nonnegligible probability:

$$\begin{aligned}
(\sigma \log P_Y)(u^*)^\pi \quad &- (\beta'\sigma \log P_Y^*)u^* \quad - (\gamma' \log(DE^{\mu^*})) = 0 \\
(\sigma \log C_{Y,1})(u^*)^\pi \quad &- (\beta'\sigma \log C_{Y,1}^*)u^* \qquad - (\gamma' \log C_1) = 0
\end{aligned}$$

  Note that these are polynomials in $u^*$ of degree $\pi$, and no terms collect together, as $\pi \notin \{0, 1\}$. We now argue that these two polynomials cannot be simultaneously zero with nonnegligible probability, unless the ciphertext is degenerate and would be rejected on other grounds:

  - If one of the polynomials is not identically zero but has some coefficient equal to zero, then this polynomial is equivalent to (i.e, has the same roots as) an affine polynomial in a single variable; either $u^*$ or $(u^*)^\pi$ or $(u^*)^{\pi-1}$. Each of these variables is uniform in $\widehat{\mathbb{G}}$, so the equation is satisfied with only negligible probability.

- Otherwise, if the two polynomials have all nonzero coefficients and are identical up to scalar multiplication, then the three pairs of matching coefficients have the same ratios. In particular, we have the following equality (after cancellation):

  $$\frac{\log(DE^{\mu^*})}{\log C_1} = \frac{\log P_Y^*}{\log C_{Y,1}^*}$$

  The challenge ciphertext (including the components $P_Y^*$ and $C_{Y,1}^*$) is generated after $C_1$, $D$, $E$, and $\mu^*$ are fixed. It is only with negligible probability over the randomness of $\mathsf{AltGenCiph}$ that $C_{Y,1}^*$ an $P_Y^*$ satisfy this condition. Thus it does not affect the outcome of our analysis to condition the entire HCCA experiment on this event not happening.
- If the two polynomials have all nonzero coefficients and are not identical up to scalar multiplication, then some linear combination of them is affine either in the variable $u^*$ or in $(u^*)^\pi$. The two original polynomial equations must have been simultaneously satisfied with noticeable probability. When both equations hold, then so does any linear combination of the two. But we have demonstrated a linear combination of the equations that is affine on a single variable which is distributed uniformly over $\widehat{\mathbb{G}}$, and thus is satisfied with only negligible probability.
- Otherwise one polynomial is identically zero. It is only with negligible probability that $\mathsf{AltGenCiph}$ generates a ciphertext with $\log C_{Y,1}^* = 0$ or $\log P_Y 1^* = 0$. Thus our analysis may be conditioned on these events not happening. We must have $\beta' = 0$ to make the $u^*$ coefficient zero (since $\sigma \neq 0$ unconditionally). This makes a coefficient in the other polynomial zero as well. This case overlaps with the first case unless both polynomials are in fact identically zero. If both are identically zero, then by similar reasoning, we must have $\gamma' = 0$ ($\log C_1 = 0$ only with negligible probability over the key generation). But when $\beta' = \gamma' = 0$, $\vec{y}$ is the all-zeroes vector and the ciphertext is rejected by $\mathsf{Integrity}$. □

We now characterize precisely which queries are accepted by the decryption oracles in the HCCA experiment, to show that their outputs are independent of $b$.

- $\mathsf{RigDec}$ when $b = 0$. Then $\mathsf{RigDec}$ is the normal $\mathsf{Dec}$ oracle. The challenge ciphertext $\zeta^*$ is an encryption of $(m_1, \ldots, m_n)$ given by the adversary, and $\mu^* = \mathsf{H}(f(m_1, \ldots, m_n))$.
  - In the $\pi = 0$ case, it is straight-forward to see that the ciphertext passes its integrity check with $\mu$ computed from the purported plaintext only if $\zeta$ is in the support of $\mathsf{Enc}_{PK}(\cdot)$.
  - In the $\pi = 1$ case, let $(m_1', \ldots, m_n') = \mathsf{PurpMsg}_{SK}(\zeta, u)$. Then we must have $\mathsf{H}(f(m_1, \ldots, m_n)) = \mathsf{H}(f(m_1', \ldots, m_1'))$. By the collision-resistance of $\mathsf{H}$, this only happens when the two plaintexts are in the same $\mathbb{H}$-coset. Thus $(m_1, \ldots, m_n) * (m_1', \ldots, m_n')^{-1} \in \mathbb{H}$, and so $(m_1', \ldots, m_n')$ is an allowed transformation of $(m_1, \ldots, m_n)$. It is straight-forward to see that the remaining components' integrity checks succeed only if $\zeta$ is in the support of $\mathsf{CTrans}(\zeta^*, \cdot)$.
- $\mathsf{RigDec}$ when $b = 1$. Then $\mathsf{RigDec}$ first calls $\mathsf{RigExtract}$ using $\mu^*$, which are distributed independently of the adversary's view.

  - In the $\pi = 0$ case, ciphertexts of this form have a unique fixed (with respect to the adversary's view) value $\mu$ such that $\mathsf{Integrity}_{SK}(\zeta, \mu)$ succeeds. Since $\mu^*$ is distributed independently of the adversary's view, this happens with negligible probability.
  - In the $\pi = 1$ case, $\mathsf{RigExtract}$ accepts only if the purported plaintext of $\zeta$ is changed from $\zeta^*$ via an allowed transformation, and that the integrity constraints pass with the same $\mu^*$ value. It is straight-forward to see that these conditions hold only if $\zeta$ is in the support of $\mathsf{CTrans}(\zeta^*, \cdot)$.

Then $\mathsf{RigDec}$ calls the normal $\mathsf{Dec}$ procedure:

  - The $\pi = 0$ case is analagous to the $b = 0$ case above, it is not affected by the difference in generation of $\zeta^*$. It accepts only if $\zeta$ is in the support of $\mathsf{Enc}_{PK}(\cdot)$.
  - If $\pi = 1$, then the ciphertext's integrity is checked using $\mu$ values derived from the purported plaintext. The purported plaintext is information-theoretically fixed from the adversary's view. Only with negligible probability will $\mu$ equal the $\mu^*$ value used to generate $\zeta^*$, which is necessary for $\mathsf{Dec}$ to accept with nonnegligible probability.

Thus, $\mathsf{RigDec}$ may be replaced by an (unbounded) oracle which does the following on input $\zeta$.

- If $\zeta$ is in the support of $\mathsf{Enc}_{PK}(m_1, \ldots, m_n)$ then output $m_1, \ldots, m_n$.
- Otherwise, if $\zeta$ is in the support of $\mathsf{CTrans}(\zeta^*, T)$ for some $T \in \mathcal{T}$, then output $T(m_1^*, \ldots, m_n^*)$, where $m_1^*, \ldots, m_n^*$ is the challenge plaintext given by the adversary in the challenge phase.

By the above argument, the output of this oracle matches that of $\mathsf{RigDec}$ on *all* queries with overwhelming probability, in both the $b = 0$ and $b = 1$ branches of the HCCA experiment. Similarly, the argument for $\mathsf{RigDec}$ when $b = 0$ implies that we may also replace the $\mathsf{Dec}$ oracle given to the adversary in Phase I with an oracle which simply checks that its input is in the support of $\mathsf{Enc}_{PK}(\cdot)$.

We show a similar situation for inputs to the $\mathsf{GRigExtract}(i, \cdot)$ oracle:

- In the $\pi = 0$ case of an input ciphertext, the given ciphertext $\zeta$ is accepted only if its purported plaintext is an allowed transformation, and its integrity checks pass with respect to the same values as $\zeta_i$ (the $i$th output of $\mathsf{GRigEnc}$). It is straight-forward to see that this is only possible for $\zeta$ from the support of $\mathsf{CTrans}(\zeta_i, \cdot)$.
- In the $\pi = 1$ case of inputs, we must have $\mu^*$ equal to the $\mu$ value used to generate $\zeta_i$. Note that the $\mu$ value used to generate $\zeta_i$ is information-theoretically fixed given $\zeta_i$ and the public key (it was created using $\mathsf{GenCiph}$ instead of $\mathsf{AltGenCiph}$).

  When $b = 1$, the two sets of $\mu$ values are only equal with negligible probability, as $\mu^*$ is chosen independently of $\mu$.

  When $b = 0$, the $\mu^*$ value is computed via $\mathsf{H}(f(m_1^*, \ldots, m_n^*))$, where $m_1^*, \ldots, m_n^*$ is the challenge plaintext given by the adversary. For the adversary to be given $\zeta_i$ and subsequently be able to compute $(m_1^*, \ldots, m_n^*)$ which yield the correct $\mu$ value, the adversary must be able to compute discrete logs in $\mathbb{G}$.

  To see the reduction, suppose we are given a random pair $g, g^\mu$ as input. Then we perform 4 randomized reductions to obtain $g_j, g_j^\mu$ pairs, and generate a DSCS keypair honestly using these $g_j$ values. We can compute a public key component $E$ as well as the value $E^\mu$ needed to generate $\zeta_i$. For the output of $\mathsf{GRigEnc}$, use this value when generating the output of $\mathsf{RigEnc}$. The distribution of this ciphertext is correct, as $\mu$ is random. When the adversary gives the challenge plaintext, compute $\mu' = \mathsf{H}(f(m_1^*, \ldots, m_n^*))$. If $g^{\mu'} = g^\mu$, then we have successfully computed the discrete log. In a group where the DDH assumption holds, this can only happen with negligible probability.

Similar to above, we may replace the $\mathsf{GRigExtract}(i, \cdot)$ oracle with an unbounded oracle that checks whether its input is in the support of $\mathsf{CTrans}(\zeta_i, \cdot)$, and if so outputs the transformation. Such an oracle is clearly independent of $b$, and by the above discussion, all of its responses match that of $\mathsf{GRigExtract}(i, \cdot)$, with overwhelming probability.

**Lemma 11.** *Our construction is HCCA secure, if the DDH assumption holds in $\mathbb{G}$ and $\widehat{\mathbb{G}}$.*

*Proof.* First, by Lemma 5, any adversary's advantage in the HCCA experiment changes negligibly when $\mathsf{GenCiph}$ is replaced by $\mathsf{AltGenCiph}$ to generate the challenge ciphertext. By Lemma 4, this challenge ciphertext is distributed independently of $b$.

Next, we may replace all the oracles which use the private key with unbounded variants as described above. The entire set of responses from these oracles coincides with the original oracles, with overwhelming probability, so any adversary's advantage is only negligibly affected by this modification to the experiment. However, when these modified oracles are used, the adversary's entire view (public key, challenge ciphertext, and oracle responses) is independent of $b$. Thus the adversary has zero advantage in this modified HCCA experiment. It follows that the adversary's advantage in the original experiment is negligible. □

**Lemma 12.** *Our construction is unlinkably homomorphic.*

*Proof.* Above, we argued that in the HCCA experiment, with overwhelming probability, the only ciphertexts accepted by the $\mathsf{Dec}$ in Phase I are those which are in the support of $\mathsf{Enc}_{PK}(\cdot)$. The unlinkability experiment is a restricted special case of Phase I of the HCCA experiment where there are no $\mathsf{GRigEnc}$ or $\mathsf{GRigExtract}$ oracles. So in the unlinkability experiment also, with overwhelming probability, only ciphertexts in the support of $\mathsf{Enc}_{PK}(\cdot)$ are accepted by the $\mathsf{Dec}$. Further, the perfect rerandomization property implies that the adversary has zero advantage in the unlinkability experiment, conditioned on this event. Thus, since our construction does satisfy the perfect rerandomization property, the adversary's advantage in the unlinkability experiment must be negligible. □