Bi-Deniable Public-Key Encryption

Adam O'Neill[*] Chris Peikert[†] Brent Waters[‡]

September 15, 2011

Abstract

In CRYPTO 1997, Canetti *et al.*put forward the intruiging notion of *deniable encryption*, which (informally) allows a sender and/or receiver, having already performed some encrypted communication, to produce 'fake' (but legitimate-looking) random coins that open the ciphertext to another message. Deniability is a powerful notion for both practice and theory: apart from its inherent utility for resisting coercion, a deniable scheme is also noncommitting (a useful property in constructing adaptively secure protocols) and secure under selective-opening attacks on whichever parties can equivocate. To date, however, known constructions have achieved only limited forms of deniability, requiring at least one party to withhold its randomness, and in some cases using an interactive protocol or external parties.

In this work we construct *bi-deniable* public-key cryptosystems, in which both the sender and receiver can simultaneously equivocate; we stress that the schemes are noninteractive and involve no third parties. One of our systems is based generically on "simulatable encryption" as defined by Damgård and Nielsen (CRYPTO 2000), while the other is lattice-based and builds upon the results of Gentry, Peikert and Vaikuntanathan (STOC 2008) with techniques that may be of independent interest. Both schemes work in the so-called "multi-distributional" model, in which the parties run alternative key-generation and encryption algorithms for equivocable communication, but claim under coercion to have run the prescribed algorithms. Although multi-distributional deniability has not attracted much attention, we argue that it is meaningful and useful because it provides credible coercion resistance in certain settings, and suffices for all of the related properties mentioned above.

Keywords. Deniable encryption, noncommitting encryption, simulatable encryption, lattice cryptography.

1 Introduction

Suppose that Eve has two children: Alice, who is away at college, and a young Bob, who still lives at home. The siblings are planning a surprise party for Eve, so to keep their plans secret, they communicate using public-key encryption. Eve, however, has taken note of their encrypted communications and grows suspicious. Using her inherent parental authority, she demands that Alice and Bob reveal their secret decryption keys, as

[*]Department of Computer Science, University of Texas at Austin. Most of this work was completed while at the Georgia Institute of Technology.

[†]School of Computer Science, Georgia Institute of Technology. This material is based upon work supported by the National Science Foundation under Grant CNS-0716786 and CAREER Award CCF-1054495, and by the Alfred P. Sloan Foundation. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

[‡]Department of Computer Science, University of Texas at Austin. Supported by NSF CNS-0915361 and CNS-0952692, AFOSR Grant No: FA9550-08-1-0352, DARPA PROCEED, DARPA N11AP20006, Google Faculty Research award, the Alfred P. Sloan Foundation, and Microsoft Faculty Fellowship.

well as any of the encryption randomness they might have retained. Is there any way for Alice and Bob to comply, without spoiling the surprise? The answer seems to be obviously no: using the secret keys, Eve can simply decrypt their messages and learn about the party.

However, the above argument misses a subtle point: if Alice and Bob are able to produce *alternative* keys and randomness that are consistent with their ciphertexts so far, then they might be able to fool Eve into thinking that they are communicating about something else (or at least not alert her to the party). A scheme that makes this possible is said to be *deniable*, a notion formally introduced by Canetti, Dwork, Naor, and Ostrovsky [CDNO97]. (Deniability is related to, but different from Benaloh and Tuinstra's notion of *uncoercible communication* [BT94], in which a sender is able to undetectably indicate to a receiver than he is being coerced to send a particular message.)

In practice, deniable encryption has been sought by users whose legitimate activities may not always be protected from subpoenas or legal coercion, e.g., journalists and whistleblowers, or lawyers and activists in repressive regimes. Indeed, several commercial and open-source storage encryption products claim limited forms of deniability (see, for example, [Rub, Tru], and further references in [Wik10]), though without formal definitions or supporting security proofs. More worryingly, these products only allow for denying the *existence* of messages on a storage medium, not for *equivocating* those messages. This is insufficient in a communications setting, where the mere exchange of messages between parties indicates that they are communicating in some form.

Deniability is also a compelling property for theoretical reasons: in particular, deniable encryption schemes are *noncommitting* (a fundamental concept in the design of adaptively secure protocols) [CFGN96, DN00, CDSMW09], secure against *selective-opening* attacks [DNRS99, BHY09], and imply incoercible multiparty computation [CG96]. We point out that deniable encryption is stronger than noncommitting encryption, because equivocable ciphertexts actually decrypt to the intended messages, and users of the system (not just a simulator) can themselves produce such ciphertexts.

Canetti *et al.* distinguish between two different models of deniability. The first is *full* deniability, in which the parties always run the prescribed key-generation and encryption algorithms, and can equivocate their messages later on if they so choose. The second model is called *multi-distributional* deniability, in which there exist alternative "deniable" algorithms whose outputs can be equivocated, so that it appears as if the *prescribed* algorithms had been used all along. Whether these models are useful in various settings has been the subject of some debate over the years; we discuss these issues in Section 1.2 below. We also discuss some recent developments and related work in Section 1.3.

Under standard assumptions, Canetti *et al.* construct a multi-distributional *sender*-deniable scheme (i.e., one that remains secure if only the sender is coerced), and give a *fully* sender-deniable scheme where the coercer's distinguishing advantage between a 'real' and 'fake' opening is an inverse polynomial that depends on the public key size. They also construct a receiver-deniable scheme that requires an additional round of interaction, and a sender- and receiver-deniable protocol that relies on third parties, at least one of whom must remain uncoerced. In particular, up to this point there have not been any noninteractive schemes offering receiver-deniability, nor any schemes (interactive or not) in which all the parties can be coerced simultaneously, in either of the two models (full or multi-distributional deniability).

1.1 Our Contributions

Bideniable public-key encryption. Our main results are the first known *bideniable* (that is, simultaneously sender- and receiver-deniable) public-key encryption schemes, in the multi-distributional model. We stress that the schemes are noninteractive, require no third parties, and are immediately noncommitting and secure under selective-opening attacks.

We give two qualitatively different constructions. The first is built generically, using a combinatorial construction with somewhat large overhead, from any "simulatable" encryption scheme in the sense of Damgård and Nielsen [DN00]. This shows (perhaps surprisingly) that simulatability is sufficient not only for noncommitting encryption, but for a form of deniability as well. The scheme is presented in Section 4.

Our second scheme is based on worst-case lattice problems via "learning with errors" [Reg05], and builds upon the trapdoor function and identity-based encryption techniques of Gentry, Peikert, and Vaikuntanathan [GPV08]. In particular, it exploits a unique property of the GPV IBE, namely its negligible chance of oblivious decryption error when the secret key vector and the error vector in the ciphertext are too highly "aligned." Our scheme relies on the ability of the receiver to resample a fresh secret key that is highly correlated with the ciphertext error term. Proving that this secret key "looks real" relies on a symmetry between correlated (discrete) Gaussians, which we believe may be of independent interest and application in lattice cryptography. Interestingly, the deniable scheme is essentially identical to the GPV cryptosystem, and it appears to be the first known cryptosystem that "naturally" supports receiver-deniability without any substantial changes. It is also essentially as efficient for the receiver as the GPV system, but it is less efficient for the sender because she must encrypt each bit separately (rather than amortizing). The details of the system are described in Section 6.

In addition to our public-key schemes, we also define notions of deniability for the identity-based setting, and show how our techniques immediately adapt to it as well. As we discuss below, multi-distributional deniability (especially for the receiver) may be more palatable in this setting because the receiver does not run a different key-generation algorithm (indeed, there is no such algorithm). We also remark that to be meaningful, the identity-based setting inherently requires any solution to be noninteractive.

Plan-ahead bideniability with short keys. A simple information-theoretic argument reveals that in any noninteractive receiver-deniable encryption scheme, the secret key must be at least as long as the message: a fixed ciphertext can only decrypt to N different plaintexts if there are at least N distinct secret keys for the public key (see also [Nie02] and the recent work [BNNO11]). We circumvent this constraint by designing a scheme offering *"plan-ahead"* bideniability (a notion also introduced in [CDNO97]) that can encrypt arbitrarily long messages using *fixed-sized* keys. In plan-ahead deniability, the sender must choose at encryption time a bounded number of potential 'fake' messages, to which the parties may later equivocate. This may be seen as the deniable analogue of "somewhat noncommitting encryption," introduced by Garay, Wichs and Zhou [GWZ09]. In many cases, this model would seem to be sufficient for coercion resistance, since the sender can just include one "innocuous" message along with the real one.

Our plan-ahead scheme is a hybrid system that reduces the deniable encryption of an arbitrary-length message to that of a short symmetric key. For example, when combined with the moderately good efficiency of our GPV-style bideniable scheme, the overall system is potentially usable in practice. (Though as noted above, our scheme is not able to amortize many bits into one ciphertext as the GPV scheme does, so our bandwidth requirements are larger.)

Relations among notions. We clarify and study relations among the various types of deniability introduced in [CDNO97]. Our main contribution is that any type of *multi-distributional* deniability suffices to obtain the corresponding type of *full* deniability, with an inverse-polynomial distinguishing advantage related to the size of the public key. This further reinforces the usefulness of multi-distributional deniability itself. We also observe that for multi-distributional schemes, bideniability implies sender-deniability, but perhaps surprisingly, it may not imply receiver-deniability alone. That is, bideniability relies on the sender to correctly run the deniable encryption algorithm.

1.2 Discussion

Is (multi-distributional) deniability useful? The ideal deniable encryption scheme would be noninteractive and fully deniable for both parties. Unfortunately, it has been recently shown [BNNO11] that these properties cannot all be achieved at once, even for receiver deniability alone (see related work below). So to obtain a noninteractive scheme we must work in the multi-distributional model.

A common objection to multi-distributional deniability is that, since there are alternative deniable algorithms (for encryption and key generation) that are strictly more powerful than the normal ones, why would anyone ever run the normal algorithms? And given this situation, why would a coercer ever accept a transcript corresponding to the normal algorithms? Whether this is a significant problem will depend on the setting in which coercion happens, and what recourse the coercer has in response to the users' claims.

For example, if there is a prescribed legal process (such as a subpoena or search warrant) by which parties are forced to reveal their transcripts, then multi-distributional deniability may be sufficient to protect the users. Even if the coercer asks for a transcript corresponding to the deniable algorithms, the users can simply assert that they did not run those algorithms, and so cannot produce coins for them. The users' claims might also gain in credibility via "safety in numbers," if a deployed implementation defaults to the normal algorithms — which do make up an operational cryptosystem, after all — or by formally standardizing on the normal algorithms within an organization. Since the coercer would only have *reason to believe* — but not any actual *evidence* — that the deniable algorithms were used in a particular instance, imposing a sanction seems fundamentally unjust, and might even be ruled out by the prescribed process. If, on the other hand, the coercer is able to punish the users until they "tell him what he wants to hear," then multi-distributional deniability might not be enough to protect the users — but neither might full deniability! After all, in either model the coercer has no reason to believe what the users have revealed. Anticipating potential punishment, the users of a multi-distributional scheme might retain the coins of the deniable algorithms as a "backup plan," just in case they might later want to reveal a convincing proof for the true message (e.g., if the punishment becomes too severe). But even a fully deniable scheme allows for a similar backup plan and proof of the true message, by using "verifiably random" coins such as the digits of π or the output of a pseudorandom generator.

In the identity-based setting, multi-distributional deniability may be useful as well, especially for the receiver. Here the receiver does not run a key-generation algorithm at all, but instead gets his secret key from an authority who possesses a 'master' secret key for all users. Our model allows the receiver to ask the authority for a fake (but real-looking) secret key that causes a particular ciphertext to decrypt to any desired message. This could be useful if the authority is out of the coercer's jurisdiction (e.g., if the receiver is travelling in another country), or if it can argue that exposing its master secret key would harm the privacy of too many other users.

In summary, the purpose of deniability is not at all to 'convince' the coercer that the surrendered transcripts are real; indeed, it is common knowledge that they can easily be faked. Instead, the goal is to *preempt coercion* in the first place by making it useless, since parties who "stick to their stories" can never be pinned down to the real message. At the same time, neither form of deniability seems appropriate if a user might eventually want to convincingly reveal the true plaintext, e.g., to sell her vote in an election. The main significant difference we see between the two models relates not to security, but usability: multi-distributional deniability requires the users to know in advance which messages they might want to equivocate, whereas full deniability allows the user to decide afterward.

Why not erase? At first glance, erasures appear to provide a very simple way of achieving deniability: the parties can just tell the coercer that they deliberately erased their coins (perhaps according to a published schedule), and therefore cannot surrender them. For sender deniability, this claim might be credible, since there is no point in the sender keeping her ephemeral encryption coins. (And indeed, none of our results preclude using erasures on the sender side.) For receiver deniability, erasures seem much less credible, since the receiver must store some form of decryption key in order to decrypt messages, and at some point in time this key can be subject to coercion. Certain regulatory regimes (e.g., in the financial sector) might also mandate retention of all data for a certain time, for potential audit. The existence of deniable encryption means that such requirements would still not necessarily guarantee compliance with the intent of the regulations. In any case, we contend that there is a significant qualitative difference between deniable schemes that use erasures, and those that do not. In the former, the coerced parties must resort to the claim that they no longer have the desired evidence, even though they once did. In the latter, the parties can credibly claim to have provided the coercer with all the evidence they have ever had, and yet still equivocate.

1.3 Other Related Work

Subsequent to the initial dissemination of this work in mid-2010, there has been additional work on deniable encryption that complements our own. Dürmuth and Freeman [DF11b] announced an interactive, fully sender-deniable encryption protocol (i.e., one with a single encryption protocol and negligible detection advantage). However, following its publication Peikert and Waters found a complete break of the system's deniability property (and a corresponding flaw in the claimed security proof); see [DF11a] for details. In particular, the problem of constructing a fully deniable encryption scheme remains an intriguing open question. Bendlin *et al.* [BNNO11] recently showed that a noninteractive public-key scheme having key size σ can be fully receiver-deniable (or bideniable) only with non-negligible $\Omega(1/\sigma)$ detection advantage. In particular, their result implies that our use of the multi-distributional model is necessary to achieve a noninteractive receiver-deniable scheme.

Deniability can be seen as a type of security that holds true even when secret information is revealed to the adversary. In the case of break-ins, one relevant notion is "forward security" (see, e.g., [BM99, CHK03]), which relies on secure erasures to update secret state over time. In the case of side-channel or memory attacks, relevant notions include the "bounded retrieval model" and "leakage-resilient" cryptography, which limit the amount of secret information the adversary may learn (see the recent survey [ADW09] and references therein). In contrast, deniability ensures security in contexts where the adversary obtains the *entire, unchanging* secret key and/or encryption randomness, but cannot tell whether those values came from the actual execution of the system.

1.4 Organization

In Section 3 we motivate and formally define various types of deniable encryption, with a focus on multi-distributional bideniability. In Section 4 we give a generic construction based on any simulatable encryption scheme. In Section 5 we define a new abstraction called a "bitranslucent set," which leads directly to a bideniable encryption scheme. In Section 6 we give an instantiation of bitranslucent sets from lattices. In Section 7 we show how to obtain plan-ahead bideniability with short keys.

2 Preliminaries

We denote the set of all binary strings by $\{0,1\}^*$ and the set of all binary strings of length i by $\{0,1\}^i$. The length of a string s is denoted by $|s|$. By $s_1 \| s_2$ we denote an encoding of strings s_1, s_2 from which the two strings are unambiguously recoverable. (If the lengths of s_1, s_2 are known, then concatenation suffices.) For $i \in \mathbb{N}$ we let $[i]$ be the set $\{1, \ldots, i\}$. A nonnegative function $f \colon \mathbb{N} \to \mathbb{R}$ is called *negligible*, written $f(n) = \mathrm{negl}(n)$, if it vanishes faster than any inverse polynomial, i.e., $f(n) = o(n^{-c})$ for every constant $c \geq 0$. A function $g \colon \mathbb{N} \to [0, 1]$ is called *overwhelming* if it is $1 - \mathrm{negl}(n)$.

The *statistical distance* between distributions X, Y with the same range is $1/2 \sum_x |\Pr[X = x] - \Pr[Y = x]|$. We say that X, Y are *statistically indistinguishable* if their statistical distance is negligible. We say that X, Y are *computationally indistinguishable* if for all efficient (PPT) distinguishers D, $|\Pr[D(X) = 1] - \Pr[D(Y) = 1]|$ is negligible. Note that when X, Y are statistically indistinguishable, the latter holds for all *unbounded* distinguishers.

For a finite (or compact) set S, $x \leftarrow S$ denotes that x is sampled according to the uniform distribution on S. For a (possibly randomized) algorithm A, the expression $z \leftarrow A(\cdots)$ denotes that A is executed on the elided inputs and z is assigned its output. We write $z \leftarrow A(\cdots; r)$ to make A's random coins r explicit. All algorithms we consider may be randomized unless indicated otherwise.

We say an algorithm A with input space $\mathcal{X}$ has *invertible sampling* [DN00] if there is an efficient inverting algorithm, denoted I_A, such that for all $x \in \mathcal{X}$ the outputs of the following two experiments are indistinguishable (either computationally, or statistically):

$$\begin{array}{r|l} y \leftarrow A(x; r) & y \leftarrow A(x; r) \\ & r' \leftarrow I_A(x, y) \\ \text{Return } (x, y, r) & \text{Return } (x, y, r') \end{array}$$

In other words, given just an input-output pair of A, it is possible to efficiently generate appropriately distributed randomness that "explains" it. It may also be the case that I_A requires some "trapdoor" information about x in order to do so. Namely, we say that A has *trapdoor* invertible sampling if we replace the second line in the right-hand experiment above with "$r' \leftarrow I_A(td_x, x, y)$," where td_x is a trapdoor corresponding to x (we think of x and td_x as being sampled jointly as the setup to both of the above experiments).

A *public-key cryptosystem* PKC with message space $\mathcal{M}$ consists of three algorithms: The *key generation algorithm* $\mathsf{Gen}(1^n; r_R)$ outputs a public key pk, and the randomness r_R is used as the associated secret decryption key. (This convention is natural in the context of deniability, where we might even consider coercing the receiver to reveal the random coins r_R it used to generate its public key. This is without loss of generality, since the stored "secret key," whatever its form, can always be computed from r_R.) The *encryption algorithm* $\mathsf{Enc}(pk, m; r_S)$ outputs a ciphertext c. The deterministic *decryption algorithm* $\mathsf{Dec}(pk, r_R, c)$ outputs a message m or $\perp$. For correctness, we require the probability that $m' \neq m$ be negligible for all messages $m \in \mathcal{M}$, over the experiment $pk \leftarrow \mathsf{Gen}(1^n; r_R)$, $c \leftarrow \mathsf{Enc}(pk, m)$, $m' \leftarrow \mathsf{Dec}(pk, r_R, c)$. To distinguish between the above notion and deniable encryption as defined in Section 3, we sometimes refer to the former as *normal* encryption. We say that PKC is *semantically secure* (or IND-CPA-secure) if for every $m_0, m_1 \in \mathcal{M}$ the distribution of (pk, c_0) is computationally indistinguishable from (pk, c_1) where $c_0 \leftarrow \mathsf{Enc}(pk, m_0)$, $c_1 \leftarrow \mathsf{Enc}(pk, m_1)$, and in both distributions $pk \leftarrow \mathsf{Gen}(1^n; r_R)$.

We defer the relevant background on lattice-based cryptography to Section 6.

3 Bideniable Encryption

Here we formally define bideniable encryption and its security properties, along with some weaker variants. (We use "bideniable" as shorthand for "sender-and-receiver deniable" in the language of Canetti *et al.* [CDNO97].) Following the definitions, we discuss further how our notion relates to the variants of deniable encryption introduced in [CDNO97].

As with normal encryption, bideniable encryption allows a sender in possession of the receiver's public key to communicate a message to the latter, confidentially. Additionally, if the parties are later coerced to reveal all their secret data — namely, the coins used by the sender to encrypt her message and/or those used by the receiver to generate her key — bideniable encryption allows them to do so as if *any desired message* (possibly chosen as late as at the time of coercion) had been encrypted.

In a *multi-distributional* deniable encryption scheme, there are 'normal' key generation, encryption, and decryption algorithms that can be run as usual — though the resulting communication may not be equivocable later on. In addition, there are 'deniable' key generation and encryption algorithms that can be used for equivocable communication. Associated with these deniable algorithms are 'faking' algorithms, which can generate secret coins that open a deniably generated public key and ciphertext to any desired message, as if the *normal* algorithms had been used to generate them. Note that the ability to generate fake *random coins* for the parties yields the strongest definition, since such coins can be used to compute whatever locally stored values (e.g., a secret key of some form) the coercer might expect. We now give a formal definition.

Definition 3.1 (Deniable encryption)**.** A *multi-distributional sender-*, *receiver-*, or *bi-deniable* encryption scheme DEN with message space $\mathcal{M}$ is made up of the following algorithms:

- The *normal* key-generation, encryption, and decryption algorithms $\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}$ are defined as usual for public-key encryption (see Section 2). These algorithms make up the *induced normal scheme*.
- The *deniable* key-generation algorithm $\mathsf{DenGen}(1^n)$ outputs (pk, fk), where fk is the *faking key*.[1] We also extend Dec to so that it can decrypt using fk in lieu of the usual receiver randomness r_R.
- The *deniable* encryption algorithm DenEnc has the same interface as the normal encryption algorithm.
- The *sender faking algorithm* $\mathsf{SendFake}(pk, r_S, m', m)$, given a public key pk, original coins r_S and message m' of DenEnc, and desired message m, outputs faked random coins r_S^* for Enc.
- The *receiver faking algorithm* $\mathsf{RecFake}(pk, fk, c, m)$, given the public and faking keys pk and fk (respectively), a ciphertext c, and a desired message m, outputs faked random coins r_R^* for Gen.

We require the following properties:

1. *Correctness.* Any triplet $(\mathsf{G}, \mathsf{E}, \mathsf{Dec})$, where $\mathsf{G} \in \{\mathsf{Gen}, \mathsf{DenGen}\}$ and $\mathsf{E} \in \{\mathsf{Enc}, \mathsf{DenEnc}\}$, should form a correct public-key encryption scheme.
2. *Multi-distributional deniability.* Let $m, m' \in \mathcal{M}$ be arbitrary messages, not necessarily different. The appropriate experiment below, which represents equivocation (by the appropriate party/parties) of an encrypted m' as m, should be computationally indistinguishable from the following 'honest opening' experiment: let $pk \leftarrow \mathsf{Gen}(1^n; r_R)$, $c \leftarrow \mathsf{Enc}(pk, m; r_S)$, and output pk, c, and whichever of r_R, r_S are appropriate to the type of deniability under consideration.

Sender-Deniable	Receiver-Deniable	Bi-Deniable
$pk \leftarrow \mathsf{Gen}(1^n; r_R)$	$(pk, fk) \leftarrow \mathsf{DenGen}(1^n)$	$(pk, fk) \leftarrow \mathsf{DenGen}(1^n)$
$c \leftarrow \mathsf{DenEnc}(pk, m'; r_S)$	$c \leftarrow \mathsf{Enc}(pk, m'; r_S)$	$c \leftarrow \mathsf{DenEnc}(pk, m'; r_S)$
	$r_R^* \leftarrow \mathsf{RecFake}(pk, fk, c, m)$	$r_R^* \leftarrow \mathsf{RecFake}(pk, fk, c, b)$
$r_S^* \leftarrow \mathsf{SendFake}(pk, r_S, m', m)$		$r_S^* \leftarrow \mathsf{SendFake}(pk, r_S, m', m)$
Return (pk, c, r_S^*)	Return (pk, c, r_R^*)	Return (pk, c, r_R^*, r_S^*)

[1]Without loss of generality, we could replace fk with the randomness of DenGen, but since this randomness will never be exposed to the adversary, we elect to define a distinguished faking key.

Multi-distributional bideniability is a particularly strong theoretical notion. For example, it immediately implies non-committing encryption as defined in [CFGN96] — but in addition, equivocable ciphertexts actually decrypt to the intended messages, and can be produced by the regular users of the scheme, not just by a simulator. Bideniability is also important in practice; in particular, each party's security does not depend upon the incoercibility of the other.

Note that we did not explicitly require DEN to satisfy the standard notion of indistinguishability under chosen-plaintext attack; this is because it is implied by any of the above notions of deniability.

Proposition 3.2. *Suppose that DEN satisfies any of sender-, receiver-, or bideniability. Then any triplet* $(\mathsf{G}, \mathsf{E}, \mathsf{Dec})$*, where* $\mathsf{G} \in \{\mathsf{Gen}, \mathsf{DenGen}\}$ *and* $\mathsf{E} \in \{\mathsf{Enc}, \mathsf{DenEnc}\}$*, is semantically secure.*

Proof. We give a proof for the induced normal scheme of a bideniable scheme; the proofs for the other cases are similar. We need to show that (pk, c_0) and (pk, c_1) are indistinguishable, where $pk \leftarrow \mathsf{Gen}(1^n; r_R)$, $c_i \leftarrow \mathsf{Enc}(pk, m_i)$ for any $m_0, m_1 \in \mathcal{M}$. First, by taking $m = m_0$ and $m' = m_1$, it follows by restricting the outputs of the deniability experiments that (pk, c_0) is indistinguishable from (pk', c'_1), where $(pk', fk) \leftarrow \mathsf{DenGen}(1^n)$, $c'_1 \leftarrow \mathsf{DenEnc}(pk', m_1)$. Now by taking $m = m' = m_1$ and invoking deniability again, we have that (pk', c'_1) is indistinguishable from (pk, c_1). $\square$

While our focus is on multi-distributional bideniability, we also briefly examine interrelations among the other types. We start with a basic question: for a particular deniable encryption scheme DEN, which notions of deniability imply which others? (This question is also important in practice, since an encryption scheme may not always be used in the ways it is intended.) First, we show that bideniablility implies sender deniability.

Proposition 3.3. *If DEN is bideniable, then DEN is sender-deniable.*

Proof. By assumption, we know that the distributions (pk, c, r_R, r_S) and (pk, c, r_R^*, r_S^*) are computationally indistinguishable, as produced by the two bideniability experiments. Clearly, the distributions (pk, c, r_S) and (pk, c, r_S^*) are indistinguishable when produced by the same two experiments, where we can now omit the generation of r_R^* in the faking experiment. This modified bideniable faking experiment producing (pk, c, r_S^*) differs from the sender-deniable faking experiment only its use of DenGen instead of Gen. Because neither experiment uses fk, all we need is for the pks output by DenGen and by Gen to be indistinguishable. But this follows directly from bideniability, by restricting the outputs of the two experiments to just pk. $\square$

One might also expect bideniability to imply receiver-deniability. Perhaps surprisingly, at least in the multi-distributional setting this appears not to be the case! For example, in our abstract scheme from Section 5, the receiver can equivocate a *normal* ciphertext in one direction (from 1 to 0), but apparently not from 0 to 1. In general, the problem is that to allow the receiver to equivocate a message, DenEnc may need to produce special ciphertexts that Enc would never (or very rarely) output. In other words, the receiver's ability to equivocate a message may depend crucially the sender's use of DenEnc.

In the reverse direction, it appears that a scheme that is both sender-deniable and (separately) receiver-deniable still might not be bideniable. Indeed, the fake randomness produced by $\mathsf{SendFake}$ and $\mathsf{RecFake}$ (which depend on the ciphertext c) might be obviously correlated in such a way that exposing both together is easily detected, whereas exposing only one is safe. Constructing a concrete example along these lines to demonstrate a formal separation remains an interesting problem.

4 Bideniable Encryption from Simulatable Encryption

Here we give a bideniable public-key encryption scheme from any *simulatable* one, in the sense of Damgård and Nielsen [DN00]. In particular, this shows that simulatable encryption suffices to realize not just a noncommitting encryption scheme, but also a (multi-distributional) bideniable one.

Simulatable public-key encryption. We recall the notion of a simulatable public-key encryption scheme from [DN00]. Intuitively, this is a scheme in which it is possible to 'obliviously' sample a public key without knowing the secret key, and to 'obliviously' sample the encryption of a random message without knowing the message.

Definition 4.1 (Simulatable PKE [DN00]). A *simulatable public-key encryption scheme* PKC-SIM with message space $\mathcal{M}$ is made up of the following algorithms:

- The *normal* key-generation, encryption, and decryption algorithms $\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}$ are defined as usual for a public-key encryption scheme (see Section 2).
- The *oblivious* key-generation algorithm $\mathsf{OGen}(1^n; r_{\mathsf{OGen}})$ outputs a public key opk, and has invertible sampling via algorithm I_{OGen}.
- The *oblivious* encryption algorithm $\mathsf{OEnc}(pk)$ outputs a ciphertext oc, and has invertible sampling via algorithm I_{OEnc}.

We require that PKC-SIM is semantically (IND-CPA) secure, and in addition require the following properties:

1. *Oblivious key generation.* The distribution of pk, where $pk \leftarrow \mathsf{Gen}(1^n; r_R)$, should be computationally indistinguishable from opk, where $opk \leftarrow \mathsf{OGen}(1^n; r_{\mathsf{OGen}})$.
2. *Oblivious ciphertext generation.* The output distributions of the following two experiments should be computationally indistinguishable:

$$\begin{array}{l|l} pk \leftarrow \mathsf{Gen}(1^n; r_R) & pk \leftarrow \mathsf{Gen}(1^n; r_R) \\ m \leftarrow \mathcal{M} & \\ c \leftarrow \mathsf{Enc}(pk, m) & oc \leftarrow \mathsf{OEnc}(pk) \\ \text{Return } (pk, r_R, c) & \text{Return } (pk, r_R, oc) \end{array}$$

Note that the above conditions are non-trivial in light of the the fact that $\mathsf{OGen}, \mathsf{OEnc}$ are required to have invertible sampling.

Simulatable encryption can be realized under a variety of standard computational assumptions such as DDH and RSA [DN00], as well as worst-case lattice assumptions [DN00, GPV08] (though we later show a more efficient bideniable encryption scheme based on the latter using an entirely different approach). We also note that Choi *et al.* [CDSMW09] introduced a relaxation of simulatability called "trapdoor" simulatability (and gave a construction based on factoring) and showed it yields noncommitting encryption, but this notion will not be sufficient in our construction of bideniable encryption below (though our scheme draws on and extends their techniques in other ways, as explained in more detail later).

4.1 A “Coordinated” Scheme

Overview. To get at the technical core of our construction, we first present a *coordinated* scheme in which the faked random coins r_R^* for the receiver and r_S^* for the sender are outputs of the *same* algorithm $\mathsf{FakeCoins}(pk, fk, r_S, b', b)$, the interpretation being that sender and receiver coordinate their faked coins upon being coerced. We later describe how the coordination can be removed for our specific scheme. For simplicity, we present a scheme that encrypts one-bit messages, but a scheme that encrypts $\mathrm{poly}(n)$-bit messages then follows generically by parallel repetition with independent keys.

We start with an informal description. For *normal* key generation, the receiver chooses a random size-n subset $\mathcal{R} \subset [5n]$ and generates $5n$ public keys pk_i (of the underlying simulatable encryption scheme), with knowledge of the corresponding secret key sk_i for $i \in \mathcal{R}$, and obliviously for $i \notin \mathcal{R}$. For normal encryption of a bit b, the sender chooses a random size-n subset $\mathcal{S} \subset [5n]$ and generates $5n$ ciphertexts c_i as encryptions of b under pk_i for $i \in \mathcal{S}$, and oblivious encryptions (of random bits) under pk_i for $i \notin \mathcal{S}$. To decrypt, the receiver decrypts c_i for $i \in \mathcal{R}$ and takes a majority vote, which will be correct with high probability due to the expected significant overlap between the receiver’s set $\mathcal{R}$ and the sender’s set $\mathcal{S}$.

For *deniable* key generation, the receiver generates the $5n$ public keys with knowledge of all of their secret keys. For deniable encryption of a bit b', the sender first chooses random pairwise disjoint size-n subsets $\mathcal{S}_0$, $\mathcal{S}_1$, $\mathcal{Y}$ of $[5n]$. It then encrypts 0s, 1s, and b's at positions in $\mathcal{S}_0$, $\mathcal{S}_1$, and $\mathcal{Y}$ (respectively), and elsewhere encrypts random bits, obliviously. Correctness holds with high probability because the set $\mathcal{Y}$ causes the majority encrypted bit to be b'.

For *coordinated faking* of a deniably encrypted bit b' as a normally encrypted b, the sender simply reveals the set $\mathcal{S}^* = \mathcal{S}_b$ and the randomness for the corresponding encryptions of b, claiming the remaining ciphertexts were generated obliviously (as in normal encryption). How the receiver chooses $\mathcal{R}^*$ is more subtle, because a random subset of decryption keys will tend to reveal too many b' plaintexts. Instead, it chooses $\mathcal{R}^*$ to have an appropriate random number of elements in common with $\mathcal{S}_b$, and then chooses the remainder of $\mathcal{R}^*$ as random indices from $[5n] \setminus (\mathcal{S}_0 \cup \mathcal{S}_1 \cup \mathcal{Y})$, which by construction all correspond to obliviously generated ciphertexts. The receiver reveals the secret keys corresponding to the indices in $\mathcal{R}^*$, claiming the remaining keys were generated obliviously (as in normal key generation).

Comparison to the scheme of Choi *et al.* [CDSMW09] The high-level idea for our scheme, namely running many parallel instances of an underlying simulatible encryption scheme such that in ‘normal’ operation the receiver “knows” the secret keys for only a random subset, and the sender “knows” the plaintexts and associated encryption randomness for only an independent random subset (which encrypts the actual message), was previously used by Choi *et al.* [CDSMW09] (generalizing the scheme of Damgård and Nielsen [DN00]) to construct noncommitting encryption. In ‘non-normal’ operation (which in our case means deniable, whereas for [CDSMW09] means simulated), the receiver knows all the secret keys and the sender knows all the plaintexts and associated encryption randomness. The main difference is in how the sender chooses the plaintexts in the latter case. In our scheme, it ‘plants’ two sets of 0s and 1s that it can later open, as well as a ‘biasing’ set of the true message that is never opened. This still ensures correct decryption. On the other hand, in the scheme of [CDSMW09] there is no ‘biasing’ set planted, and indeed this would not ensure correct decryption since a different decryption rule than majority vote is used.

The scheme. Formally, let PKC-SIM be simulatable public-key encryption scheme as per Definition 4.1 with message space $\{0, 1\}$. In Figure 1, we define the scheme BI-DEN[PKC-SIM] with message space $\{0, 1\}$. There and in what follows, for a set $\mathcal{X}$ we denote by $\mathcal{P}(\mathcal{X})$ the set of all subsets (i.e., the power set) of $\mathcal{X}$, and by $\mathcal{P}_i(\mathcal{X})$ the set of all size-i subsets of $\mathcal{X}$. Additionally, in algorithm $\mathsf{FakeCoins}$ we sample from what is known as the *hypergeometric distribution*. For nonnegative integers $x, y, N \geq M$, let $P_{\mathsf{HGD}}(x; N, M, y)$ denote the probability that exactly x values from $[M]$ are chosen after a total of y values are drawn uniformly at random from $[N]$, without replacement. That is,

$$P_{\mathsf{HGD}}(x; N, M, y) := \frac{\binom{M}{x} \cdot \binom{N-M}{y-x}}{\binom{N}{y}} .$$

We denote by $\mathsf{HGD}(N, M, y)$ the *hypergeometric distribution on* $[N]$ *with parameters* M, y that assigns to each $x \in \{0, \ldots, M\}$ the probability $P_{\mathsf{HGD}}(x; N, M, y)$. The expectation of a random variable with distribution $\mathsf{HGD}(N, M, y)$ is easily seen to be $y \cdot M/N$. Below we will use parameters N, M, y that are polynomial in the security parameter, so we can sample efficiently from the hypergeometric distribution simply by running the sampling experiment. The scheme BI-DEN[PKC-SIM] with message space $\{0, 1\}$ is defined in Figure 1.

4.2 Correctness and Security

Theorem 4.2. *Let PKC-SIM be a simulatable public-key encryption scheme. Then BI-DEN*[*PKC-SIM*] *is correct.*

We will use the following (lower and upper) Chernoff-like tail inequalities for the hypergeometric distribution, which follow from [Chv79].

Lemma 4.3. *Let X be distributed according to* $\mathsf{HGD}(N, M, y)$*. Then for any $t \geq 0$,*

$$\Pr[\, X \leq \mathrm{E}[X] - ty = y(M/N - t)\,] \leq e^{-2t^2 y},$$
$$\Pr[\, X \geq \mathrm{E}[X] + ty = y(M/N + t)\,] \leq e^{-2t^2 y}.$$

Proof of Theorem 4.2. The proof follows directly by some straightforward (but tedious) applications of Chernoff bounds and Lemma 4.3. Fix any $b \in \{0, 1\}$. We first show correctness of BI-DEN.Dec for the normal encryption algorithm BI-DEN.Enc, then later for the deniable encryption algorithm BI-DEN.DenEnc. In both cases, we assume that the normal key-generation algorithm BI-DEN.Gen is used; using BI-DEN.DenGen instead does not affect correctness because it produces an functionally identical keypair from the perspective of BI-DEN.Dec. Additionally, by Condition 2 of Definition 4.1 it suffices to argue correctness in the case where oblivious ciphertexts are formed by encrypting truly random bits.

In the correctness experiment, let I be the random variable taking the value $|\mathcal{S} \cap \mathcal{R}|$, and D take the number of $i \in \mathcal{R} \setminus \mathcal{S}$ such that $d_i = b$ (the message bit) as chosen by BI-DEN.Enc. Note that a decryption error occurs only if $I \leq n/2$ and $D \leq n/2 - I = (n - I)/2 - I/2$. Let us also assume for the moment that $n/10 < I \leq n/2$, which we will justify below. Then, writing D as a sum of $n - I$ indicator random variables and applying a standard Chernoff bound, we have

$$\Pr[\, D \leq (n - I)/2 - I/2\,] \leq \Pr\left[\, D \leq (1 - \tfrac{1}{9}) \cdot E[D]\,\right] \leq e^{-(n-I)/324} \leq e^{-n/648}.$$

where the first inequality uses $n/10 < I$ and the last uses $I \leq n/2$. To justify the assumption that $n/10 < I$ with high probability, note that I is distributed according to $\mathsf{HGD}(5n, n, n)$ with expectation $n/5$. Applying the first part of Lemma 4.3 with $t = 1/10$ we get

$$\Pr[\, I \leq n/10\,] \leq e^{-n/50}.$$

```
BI-DEN.Gen$(1^n)$:
    $\mathcal{R} \leftarrow \mathcal{P}_n([5n])$
    For $i = 1$ to $5n$ do:
        If $i \in \mathcal{R}$ then
            $pk_i \leftarrow \mathsf{Gen}(1^n; r_{R,i})$
        Else $pk_i \leftarrow \mathsf{OGen}(1^n; r_{R,i})$
    $pk \leftarrow pk_1 \| \dots \| pk_{5n}$
    Return $pk$
```

```
BI-DEN.Enc$(pk, b)$:
    $\mathcal{S} \leftarrow \mathcal{P}_n([5n])$
    For $i = 1$ to $5n$ do:
        If $i \in \mathcal{S}$ then
            $c_i \leftarrow \mathsf{Enc}(pk_i, b; r_{S,i})$
        Else $c_i \leftarrow \mathsf{OEnc}(pk_i; r_{S,i})$
    $c \leftarrow c_1 \| \dots \| c_{5n}$
    Return $c$
```

```
BI-DEN.Dec$((\mathcal{R}, r_R), c)$:
    For all $i \in \mathcal{R}$ do:
        $d_i \leftarrow \mathsf{Dec}(r_{R,i}, c_i)$
    If most $d_i$'s are 1 then
        Return 1
        Else return 0
```

```
BI-DEN.DenGen$(1^n)$:
    $\mathcal{R} \leftarrow \mathcal{P}_n([5n])$
    For $i = 1$ to $5n$ do:
        $pk_i \leftarrow \mathsf{Gen}(1^n; r_{R,i})$
    $pk \leftarrow pk_1 \| \dots \| pk_{5n}$
    $r \leftarrow r_{R,1} \| \dots \| r_{R,5n}$
    Return $(pk, (\mathcal{R}, r))$
```

```
BI-DEN.DenEnc$(pk, b')$:
    $\mathcal{S}_0 \leftarrow \mathcal{P}_n([5n])$
    $\mathcal{S}_1 \leftarrow \mathcal{P}_n([5n] \setminus \mathcal{S}_0)$
    $\mathcal{Y} \leftarrow \mathcal{P}_n([5n] \setminus (\mathcal{S}_0 \cup \mathcal{S}_1))$
    For $i = 1$ to $5n$ do:
        If $i \in \mathcal{S}_0$ then $c_i \leftarrow \mathsf{Enc}(pk_i, 0; r_{S,i})$
        If $i \in \mathcal{S}_1$ then $c_i \leftarrow \mathsf{Enc}(pk_i, 1; r_{S,i})$
        If $i \in \mathcal{Y}$ then $c_i \leftarrow \mathsf{Enc}(pk_i, b'; r_{S,i})$
        Else $c_i \leftarrow \mathsf{OEnc}(pk; r_{S,i})$
    $c \leftarrow c_1 \| \dots \| c_{5n}$
    Return $c$
```

```
BI-DEN.FakeCoins$(pk, fk, r_S, b', b)$:
    $c \leftarrow$ BI-DEN.Enc$(pk, b'; r_S)$
    $z \leftarrow \mathsf{HGD}(5n, n, n)$
    $\mathcal{Z} \leftarrow \mathcal{P}_z(\mathcal{S}_b)$
    $\mathcal{Z}' \leftarrow \mathcal{P}_{n-z}([5n] \setminus (\mathcal{S}_0 \cup \mathcal{S}_1 \cup \mathcal{Y}))$
    $\mathcal{R}^* \leftarrow \mathcal{Z} \cup \mathcal{Z}'$
    $\mathcal{S}^* \leftarrow \mathcal{S}_b$
    For $i = 1$ to $5n$ do:
        If $i \in \mathcal{S}^*$ then $r^*_{S,i} \leftarrow r_{S,i}$
        Else $r^*_{S,i} \leftarrow I_{\mathsf{OEnc}}(pk_i, c_i)$
        If $i \in \mathcal{R}^*$ then $r^*_{R,i} \leftarrow r_{R,i}$
        Else $r^*_{R,i} \leftarrow I_{\mathsf{OGen}}(pk_i)$
    $r^*_S \leftarrow r^*_{S,1} \| \dots \| r^*_{S,5n}$ ; $r^*_R \leftarrow r^*_{R,1} \| \dots \| r^*_{R,5n}$
    Return $(r^*_S, r^*_R)$
```

Figure 1: Algorithms of the “coordinated” bindeniable scheme BI-DEN[PKC-SIM].

Thus, the overall probabililty of a decryption error is exponentially small in n.

To show correctness when the deniable encryption algorithm BI-DEN.DenEnc is used, we extend the above argument slightly. Now, in the correctness experiment, let I be the random variable taking the value $|\mathcal{S}_b \cap \mathcal{Y} \cap \mathcal{R}|$, let B take the value $|\mathcal{R} \cap \mathcal{S}_{1-b}|$, and let D take the number of $i \in \mathcal{R} \setminus (\mathcal{S}_b \cap \mathcal{S}_{1-b} \cap \mathcal{Y})$ such that $d_i = b$. Similarly to before, decryption error occurs only if $I \leq n/2$ and $D \leq n/2 - I = (n - I - B)/2 + B/2 - I/2$. Let us also assume for the moment that $19n/50 < I \leq n/2$ and $B < 11n/50$, which as before we will justify below. Then, writing D as a sum of $n - I - B$ indicator random variables and applying a standard Chernoff bound, we have

$$\Pr\left[\, D \leq (n - I - B)/2 + B/2 - I/2 \,\right] \leq \Pr\left[\, D \leq (1 - \tfrac{3}{4}) \cdot E[D] \,\right] \leq e^{-9(n-I-B)/64} \leq e^{-9n/160}$$

where the first inequality uses $19n/50 < I$ and $B < 11n/50$, and the last uses $I \leq n/2$ and $B < 11n/50$. To justify the assumption that $19/50n \leq I$ with high probability, note that I is distributed according to $\mathsf{HGD}(5n, 2n, n)$ with expectation $2n/5$. Applying the first part of Lemma 4.3 with $t = 1/50$ we get

$$\Pr\left[\, I \leq 19n/50 \,\right] \leq e^{-n/1225}.$$

Similarly, to justify the assumption that $B \leq 11n/50$ with high probability, we can apply the second part of Lemma 4.3 with $t = 1/50$. Thus, the overall probabililty of a decryption error is exponentially small in n, concluding the proof of correctness. □

Theorem 4.4. *Let PKC-SIM be a simulatable public-key encryption scheme. Then BI-DEN[PKC-SIM] satisfies bideniability under chosen-plaintext attacks.*

Proof of Theorem 4.4. The proof is via a sequence of hybrid experiments G_0, G_1, G_2, G_3, defined formally in Figure 2. We give an intuitive overview of these hybrids as follows:

- Experiment G_0 is exactly the left-hand experiment of Definition 3.1; that is, it corresponds to the adversary's view of an honest opening of an honest encryption of b. In particular, it chooses the sets $\mathcal{S}_b, \mathcal{R} \subset [5n]$ at random and generates the corresponding keypairs and ciphertexts accordingly. The experiment also chooses sets $\mathcal{S}_{1-b}$ and $\mathcal{Y}$ to be disjoint from each other and from $\mathcal{S}_b$ and $\mathcal{R}$, but they are never used in the experiment; they will come into play in later hybrids.

- Experiment G_1 chooses $\mathcal{S}_b$ as before but chooses $\mathcal{R}$ in a different way, first choosing its intersection with S_b at random to be appropriate (HGD-distributed) size, then choosing the rest of its elements at random to be disjoint from S_b. After, it chooses $\mathcal{S}_{1-b}$ and $\mathcal{Y}$ as before. The output of this experiment is distributed identically to that of G_0, by definition of the hypergeometric distribution.

- Experiment G_2 chooses $\mathcal{S}_b$ as before, but changes the order in which $\mathcal{R}$, $\mathcal{S}_{1-b}$, and $\mathcal{Y}$ are chosen, such that they are chosen as in the faking experiment when faking an encryption of b' as one of b. Namely, (after choosing $\mathcal{S}_b$) it first chooses $\mathcal{S}_{1-b}$ at random disjoint from $\mathcal{S}_b$, then $\mathcal{Y}$ at random disjoint from $\mathcal{S}_b$ and $\mathcal{S}_{1-b}$, and finally $\mathcal{R}$ to have an intersection with $\mathcal{S}_b$ of appropriate size and the rest of its elements to be disjoint from $\mathcal{S}_b, \mathcal{S}_{1-b}$, and $\mathcal{Y}$. The output of this experiment is distributed identically to that of G_1, as shown in Lemma 4.6.

- Finally, experiment G_3 changes the way in which the keys and ciphertexts are generated, so that they are generated as in the faking experiment when faking an encryption of b' as one of b. Specifically, it generates all keypairs as in the deniable key generation algorithm, and then 'retroactively' claims that the public keys corresponding to the set $[5n] \setminus \mathcal{R}$ were generated obliviously (via algorithm I_{OGen})

as in the receiver faking algorithm. Additionally, it forms ciphertexts corresponding to the sets $\mathcal{Y}$ and $\mathcal{S}_{1-b}$ not as oblivious ciphertexts, but according to the DenEnc algorithm on bit b'. Again, these are 'retroactively' claimed to be oblivious ciphertexts. Using simulatability of PKC-SIM, the output distribution of this experiment is computationally indistinguishable from that of G_2, as shown in Lemma 4.7.

The theorem follows. □

Lemma 4.5. *Experiments G_0 and G_1 have the same output distribution.*

Proof. This follows directly from the definition of the hypergeometric distribution. □

Lemma 4.6. *Experiments G_1 and G_2 have the same output distribution.*

Proof. This simply follows from the fact that $\mathcal{S}_b$, $\mathcal{Z}'$, $\mathcal{S}_{1-b}$, and $\mathcal{Y}$ are all random pairwise disjoint subsets of $[5n]$ (of certain sizes), so it does not matter in what order they are chosen. (The latter can be shown using a simple counting argument.) □

Lemma 4.7. *The output distributions of G_2 and G_3 are computationally indistinguishable if PKC-SIM is simulatable.*

Proof. For the proof, we expand the transition between G_2 and G_3 via a sequence of intermediate hybrids. The idea is to have the hybrids change one-by-one how the keypairs whose indices lie outside $\mathcal{R}$ are generated, and then change one-by-one how the ciphertexts whose indices lie in $\mathcal{S}_{1-b}$ or $\mathcal{Y}$ are generated. Namely, we consider the sequence of hybrids $(G_2, K_1, \ldots, K_{4n}, C_1, \ldots, C_{2n} = G_3)$ defined as follows. For all $j \in [4n]$, hybrid K_j changes the generation of the jth keypair and its 'claimed' coins whose index (say i_j) satisfies $i_j \in [5n] \setminus \mathcal{R}$ from

$$pk_{i_j} \leftarrow \mathsf{OGen}(1^n; r_{R,i_j}) \ ; \ r^*_{R,i_j} \leftarrow r_{R,i_j}$$

to

$$pk_{i_j} \leftarrow \mathsf{Gen}(1^n; r_{R,i_j}) \ ; \ r^*_{R,i_j} \leftarrow I_{\mathsf{OGen}}(pk_{i_j}).$$

Indistinguishability of the adjacent hybrids follows by a straightforward reduction using Property 1 of Definition 4.1.

For all $k \in [2n]$, hybrid C_k changes the generation of the kth ciphertext and its 'claimed' coins whose index (say i_k) satisfies $i_k \in \mathcal{S}_{1-b} \cup \mathcal{Y}$, in a way depending on whether $i_k \in \mathcal{S}_{1-b}$ or $i_k \in \mathcal{Y}$. If $i_k \in \mathcal{S}_{1-b}$ then the change is from

$$c_{i_k} \leftarrow \mathsf{OEnc}(pk_{i_k}; r_{S,i_k}) \ ; \ r^*_{S,i_k} \leftarrow r_{S,i_k}$$

to

$$c_{i_k} \leftarrow \mathsf{Enc}(pk_{i_k}, 1-b; r_{S,i_k}) \ ; \ r^*_{S,i_k} \leftarrow I_{\mathsf{OEnc}}(pk_{i_k}, c_{i_k}).$$

We can show indistinguishability of C_{k-1} and C_k (where for simplicity we define $K_{4n} = C_0$) by considering another intermediate one, say C'_k, where the above is replaced by

$$c_{i_k} \leftarrow \mathsf{Enc}(pk_{i_k}, d; r_{S,i_k}) \ ; \ r^*_{S,i_k} \leftarrow I_{\mathsf{OEnc}}(pk_{i_k}, c_{i_k})$$

for an independent random bit d. Then C_{k-1} and C'_k are indistinguishable by Property 2 of Definition 4.1, and C'_k and C_k are indistinguishable by semantic security of PKC-SIM (which is required by Definition 4.1). In the case that $i_k \in \mathcal{Y}$, then $1-b$ above is replaced with b' and the argument is the same. Note that $C_{2n} = G_3$, completing the proof. □

```
Experiment $G_0$:
    $\mathcal{S}_b \leftarrow \mathcal{P}_n([5n])$
    $\mathcal{R} \leftarrow \mathcal{P}_n([5n])$
    $\mathcal{S}_{1-b} \leftarrow \mathcal{P}_n([5n] \setminus (\mathcal{S}_b \cup \mathcal{R}))$
    $\mathcal{Y} \leftarrow \mathcal{P}_n([5n] \setminus (\mathcal{S}_b \cup \mathcal{S}_{1-b} \cup \mathcal{R}))$
    For $i = 1$ to $5n$ do:
        If $i \in \mathcal{R}$ then $pk_i \leftarrow \mathsf{Gen}(1^n; r_{R,i})$
        Else $pk_i \leftarrow \mathsf{OGen}(1^n; r_{R,i})$
        If $i \in \mathcal{S}_b$ then $c_i \leftarrow \mathsf{Enc}(pk_i, b; r_{S,i})$
        Else $c_i \leftarrow \mathsf{OEnc}(pk_i; r_{S,i})$
    Return $(pk, c, (\mathcal{R}, r_R), (\mathcal{S}_b, r_S))$
```

```
Experiment $G_1$:
    $\mathcal{S}_b \leftarrow \mathcal{P}_n([5n])$
    $z \leftarrow \mathsf{HGD}(5n, n, n)$
    $\mathcal{Z} \leftarrow \mathcal{P}_z(\mathcal{S}_b)$
    $\mathcal{Z}' \leftarrow \mathcal{P}_{n-z}([5n] \setminus \mathcal{S}_b)$
    $\mathcal{R} \leftarrow \mathcal{Z} \cup \mathcal{Z}'$
    $\mathcal{S}_{1-b} \leftarrow \mathcal{P}_n([5n] \setminus (\mathcal{S}_b \cup \mathcal{R}))$
    $\mathcal{Y} \leftarrow \mathcal{P}_n([5n] \setminus (\mathcal{S}_b \cup \mathcal{S}_{1-b} \cup \mathcal{R}))$
    For $i = 1$ to $5n$ do:
        If $i \in \mathcal{R}$ then $pk_i \leftarrow \mathsf{Gen}(1^n; r_{R,i})$
        Else $pk_i \leftarrow \mathsf{OGen}(1^n; r_{R,i})$
        If $i \in \mathcal{S}_b$ then $c_i \leftarrow \mathsf{Enc}(pk_i, b; r_{S,i})$
        Else $c_i \leftarrow \mathsf{OEnc}(pk_i; r_{S,i})$
    Return $(pk, c, (\mathcal{R}, r_R), (\mathcal{S}_b, r_S))$
```

```
Experiment $G_2$:
    $\mathcal{S}_b \leftarrow \mathcal{P}_n([5n])$
    $\mathcal{S}_{1-b} \leftarrow \mathcal{P}_n([5n] \setminus \mathcal{S}_b)$
    $\mathcal{Y} \leftarrow \mathcal{P}_n([5n] \setminus \mathcal{S}_{1-b} \cup \mathcal{S}_b)$
    $z \leftarrow \mathsf{HGD}(5n, n, n)$
    $\mathcal{Z} \leftarrow \mathcal{P}_z(\mathcal{S}_b)$
    $\mathcal{Z}' \leftarrow \mathcal{P}_{n-z}([5n] \setminus (\mathcal{S}_b \cup \mathcal{S}_{1-b} \cup \mathcal{Y}))$
    $\mathcal{R} \leftarrow \mathcal{Z} \cup \mathcal{Z}'$
    For $i = 1$ to $5n$ do:
        If $i \in \mathcal{R}$ then $pk_i \leftarrow \mathsf{Gen}(1^n; r_{R,i})$
        Else $pk_i \leftarrow \mathsf{OGen}(1^n; r_{R,i})$
        If $i \in \mathcal{S}_b$ then $c_i \leftarrow \mathsf{Enc}(pk_i, b; r_{S,i})$
        Else $c_i \leftarrow \mathsf{OEnc}(pk_i; r_{S,i})$
    Return $(pk, c, (\mathcal{R}, r_R), (\mathcal{S}_b, r_S))$
```

```
Experiment $G_2$:
    $\mathcal{S}_b \leftarrow \mathcal{P}_n([5n])$
    $\mathcal{S}_{1-b} \leftarrow \mathcal{P}_n([5n] \setminus \mathcal{S}_b)$
    $\mathcal{Y} \leftarrow \mathcal{P}_n([5n] \setminus \mathcal{S}_{1-b} \cup \mathcal{S}_b)$
    $z \leftarrow \mathsf{HGD}(5n, n, n)$
    $\mathcal{Z} \leftarrow \mathcal{P}_z(\mathcal{S}_b)$
    $\mathcal{Z}' \leftarrow \mathcal{P}_{n-z}([5n] \setminus (\mathcal{S}_b \cup \mathcal{S}_{1-b} \cup \mathcal{Y}))$
    $\mathcal{R} \leftarrow \mathcal{Z} \cup \mathcal{Z}'$
    For $i = 1$ to $5n$ do:
        $pk_i \leftarrow \mathsf{Gen}(1^n; r_{R,i})$
        If $i \in \mathcal{R}$ then $r^*_{R,i} \leftarrow r_{R,i}$
        Else $r^*_{R,i} \leftarrow I_{\mathsf{OGen}}(pk_i)$
        If $i \in \mathcal{S}_b$ then
            $c_i \leftarrow \mathsf{Enc}(pk_i, b; r_{S,i})$
            $r^*_{S,i} \leftarrow r_{S,i}$
        Else if $i \in \mathcal{S}_{1-b}$ then
            $c_i \leftarrow \mathsf{Enc}(pk_i, 1-b; r_{S,i})$
            $r^*_{S,i} \leftarrow I_{\mathsf{OEnc}}(pk_i, c_i)$
        Else if $i \in \mathcal{Y}$ then
            $c_i \leftarrow \mathsf{Enc}(pk_i, b'; r_{S,i})$
            $r^*_{S,i} \leftarrow I_{\mathsf{OEnc}}(pk_i, c_i)$
        Else
            $c_i \leftarrow \mathsf{OEnc}(pk_i; r_{S,i})$
            $r^*_{S,i} \leftarrow r_{S,i}$
    Return $(pk, c, (\mathcal{R}, r^*_R), (\mathcal{S}_b, r^*_S))$
```

Figure 2: Hybrid experiments for showing bideniability of BI-DEN[PKC-SIM]. Shaded areas indicate the only syntactic differences with the previous hybrid.

4.3 Removing the Coordination

Removing the coordination from the scheme is based on the observation that, in the FakeCoins algorithm, while the choice of the set $\mathcal{R}^*$ depends on the choices of $\mathcal{S}_0, \mathcal{S}_1, \mathcal{Y}$, the reverse is not true; namely the choice of the set $\mathcal{S}^*$ is independent of the choice of $\mathcal{R}$ and $\mathcal{R}^*$. Thus, the necessary communication from sender to receiver can be done "in-band" via a separate instance of the given simulatable cryptosystem, by encrypting (in the deniable case) the choices of $\mathcal{S}_0, \mathcal{S}_1$ and $\mathcal{Y}$ the sender may later use to equivocate. (Note that we assume the encoding of these sets lies in the message space of the simulatable encryption scheme; the latter can always be expanded by parallel repetition using independent public keys.) The normal key generation and encryption algorithms simply generate an oblivious public key and ciphertext for this component of the system.

More specifically, for the scheme without coordination, the modified key-generation, encryption, and faking algorithms work as follows:

```
BI-DEN.DenGen'$(1^n)$:
    $(pk, (\mathcal{R}, r)) \leftarrow$ BI-DEN.Gen$(1^n)$
    $pk' \leftarrow$ Gen$(1^n; r'_R)$
    Return $(pk \| pk', (\mathcal{R}, r \| r'_R))$
```

```
BI-DEN.DenEnc'$(pk \| pk', b)$:
    $c \leftarrow$ BI-DEN.DenEnc$(pk, b)$
    $c' \leftarrow$ Enc$(pk', \langle \mathcal{S}_0 \rangle \| \langle \mathcal{S}_1 \rangle \| \langle \mathcal{Y} \rangle; r'_S)$
    Return $c \| c'$
```

```
BI-DEN.SendFake$(pk, r_S, b', b)$:
    $\mathcal{S}^* \leftarrow \mathcal{S}_b$
    For $i = 1$ to $5n$ do:
        If $i \in \mathcal{S}^*$ then $r^*_{S,i} \leftarrow r_{S,i}$
        Else $r^*_{S,i} \leftarrow I_{\mathsf{OEnc}}(pk_i, c_i)$
    $r'_S \leftarrow I_{\mathsf{OEnc}}(pk', c')$
    $r^*_S \leftarrow r^*_{S,1} \| \dots \| r^*_{S,5n} \| r'_S$
    Return $r^*_S$
```

```
BI-DEN.RecFake$(pk, fk, c \| c', b)$:
    $\langle \mathcal{S}_0 \rangle \| \langle \mathcal{S}_1 \rangle \| \langle \mathcal{Y} \rangle \leftarrow$ Dec$(r'_R, c')$
    $z \leftarrow$ HGD$(5n, n, n)$
    $\mathcal{Z} \leftarrow \mathcal{P}_z(\mathcal{S}_b)$
    $\mathcal{Z}' \leftarrow \mathcal{P}_{n-z}([5n] \setminus (\mathcal{S}_0 \cup \mathcal{S}_1 \cup \mathcal{Y}))$
    $\mathcal{R}^* \leftarrow \mathcal{Z} \cup \mathcal{Z}'$
    For $i = 1$ to $5n$ do:
        If $i \in \mathcal{R}^*$ then $r^*_{R,i} \leftarrow r_{R,i}$
        Else $r^*_{R,i} \leftarrow I_{\mathsf{OGen}}(pk_i)$
    $r'_R \leftarrow I_{\mathsf{OGen}}(pk')$
    $r^*_R \leftarrow r^*_{R,1} \| \dots \| r^*_{R,5n} \| r'_R$
    Return $r^*_R$
```

Above, $\langle \mathcal{X} \rangle$ denotes the cannonical encoding of a set $\mathcal{X}$ as a bitstring. The normal key-generation, encryption, and decryption algorithms of the modified scheme are defined in the corresponding manner. Bideniability of the modified scheme follows straightfowardly in light of Theorem 4.4, under the same assumption.

5 Bideniable Encryption from Bitranslucent Sets

Here we construct a bideniable encryption scheme based on a new primitive we call a *bitranslucent set*, which extends the notion of a translucent set from [CDNO97]. Whereas translucent sets can be constructed straightforwardly from any trapdoor permutation (and other specific assumptions) [CDNO97], bitranslucent sets appear much more challenging to realize. In Section 6 we give a novel construction based on lattices.

5.1 Bitranslucent Sets

Informally, a *translucent set* is a subset P of a universe U, which can be sampled efficiently using only public information, and which is pseudorandom unless its associated secret key is known (in which case it is easily distinguishable from uniform over the universe).

Here we strengthen the notion of a translucent set in two ways. The first, main strengthening essentially preserves the pseudorandomness of P *even if the secret key is revealed*. Of course this is impossible as just stated, because the secret key makes P 'transparent' by design. Instead, we introduce an algorithm that, given a 'faking' key for the set system, and some c drawn from the pseudorandom set P, is able to *resample* a new, 'good-looking' secret key for which c appears uniformly random. We stress that such keys are necessarily very rare, because a typical secret key should correctly recognize P with high probability. Nevertheless, it can still be the case that for any $c \in P$, there are a few rare keys that misclassify c (without making it apparent that they are doing so). A bitranslucent set scheme is able to use the faking key find such keys.

Looking ahead to our bideniable encryption scheme, bitranslucency will allow a coerced sender and receiver both to plausibly claim that a value $c \in P$ is actually uniform: the sender simply declares that c was chosen uniformly from U (by claiming c itself as the random coins), and the receiver resamples a secret key that also makes c appear uniform.

The second strengthening, which yields qualitative improvements in efficiency and may have independent applications, allows for multiple translucent sets to share a single, fixed-size faking key. Essentially, this makes the bitranslucent set 'identity-based' (although we do not refer to identities explicitly): each translucent set has its own public and secret keys for generating and distinguishing P- and U-samples, and the master faking key makes it possible to generate a good-looking secret key that equivocates a given P-sample as a U-sample. Interestingly, this implies a bideniable encryption scheme in which the deniable key generator's faking key is a *fixed* size independent of the message length, despite the information-theoretic bound that *normal* secret keys must exceed the message length.

Definition 5.1 (Bitranslucent Set Scheme (BTS))**.** A *bitranslucent set scheme* BTS is made up of the following algorithms:

- The *normal setup* procedure $\mathsf{Setup}(1^n; r_{\mathsf{Setup}})$ outputs a public parameter pp.

  We require that Setup has invertible sampling via an algorithm I_{Setup}.
- The *deniable setup* procedure $\mathsf{DenSetup}(1^n)$ outputs a public parameter pp and a *faking key* fk.
- The *key generator* $\mathsf{Gen}(pp; r_R)$ outputs a public key pk, whose associated secret key is the randomness r_R (without loss of generality).

  We require that Gen has trapdoor invertible sampling via an algorithm I_{Gen} and trapdoor fk, where $(pp, fk) \leftarrow \mathsf{DenSetup}(1^n)$.
- The P- and U*-samplers* $\mathsf{SampleP}(pp, pk; r_S)$ and $\mathsf{SampleU}(pp, pk; r_S)$ each output some $c \in \{0,1\}^*$.
- The P*-tester* $\mathsf{TestP}(pp, r_R, c)$ either accepts or rejects.
- The *sender-coins faker* $\mathsf{FakeSCoins}(pp, pk, r_S)$ outputs some coins r_S^* for the U-sampler.[2]
- The *receiver-coins faker* $\mathsf{FakeRCoins}(pp, fk, pk, c)$ outputs some coins r_R^* for the key generator.

We require:

[2]In some realizations, including our own, FakeSCoins can directly compute coins r_S^* given just a ciphertext $c \leftarrow \mathsf{SampleP}(pp, pk; r_S)$, not r_S (or even pp, pk). We retain the more relaxed definition above for generality.

1. *(Correctness.)* The following experiment should accept (respectively, reject) with overwhelming probability over all its randomness: let $pp \leftarrow \mathsf{Setup}(1^n)$, $pk \leftarrow \mathsf{Gen}(pp; r_R)$, $c \leftarrow \mathsf{SampleP}(pp, pk; r_S)$ (resp., $c \leftarrow \mathsf{SampleU}(pp, pk; r_S)$), and output $\mathsf{TestP}(pp, r_R, c)$.

   We also require correctness for the faking algorithms: with overwhelming probability over all the random choices, letting $(pp, fk) \leftarrow \mathsf{DenSetup}(1^n)$ and $pk \leftarrow \mathsf{Gen}(pp; r_R)$, we should have

$$\begin{aligned} \mathsf{SampleU}(pp, pk; \mathsf{FakeSCoins}(pk, r_S)) &= \mathsf{SampleP}(pp, pk; r_S) \\ \mathsf{Gen}(pp; \mathsf{FakeRCoins}(pp, fk, pk, \mathsf{SampleP}(pp, pk; r_S))) &= pk. \end{aligned}$$

2. *(Indistinguishable public parameters.)* The public parameters pp as produced by the two setup procedures $pp \leftarrow \mathsf{Setup}(1^n; r_{\mathsf{Setup}})$ and $(pp, fk) \leftarrow \mathsf{DenSetup}(1^n)$ should be indistinguishable (either statistically or computationally).

3. *(Bideniability.)* The following two experiments should be computationally indistinguishable:

$$\begin{array}{l|l} (pp, fk) \leftarrow \mathsf{DenSetup}(1^n) & (pp, fk) \leftarrow \mathsf{DenSetup}(1^n) \\ pk \leftarrow \mathsf{Gen}(pp; r_R) & pk \leftarrow \mathsf{Gen}(pp; r_R) \\ c \leftarrow \mathsf{SampleU}(pp, pk; r_S) & c \leftarrow \mathsf{SampleP}(pp, pk; r_S) \\ & r_R^* \leftarrow \mathsf{FakeRCoins}(pp, fk, c) \\ & r_S^* \leftarrow \mathsf{FakeSCoins}(pk, r_S) \\ \text{Return } (pp, r_R, r_S) & \text{Return } (pp, r_R^*, r_S^*) \end{array}$$

Remark 5.2. For correctness, it suffices (and is more convenient) to require that when $c \leftarrow \mathsf{SampleU}(pp, pk)$, $\mathsf{TestP}(pp, r_R, c)$ just rejects with probability at least (say) $1/2$. The error probability can then be made negligible by parallel repetition of Gen and $\mathsf{SampleU}$, using the same public parameters pp.

The definition is designed to allow for the use of many public keys pk_i and associated secret keys $r_{R,i}$ under the same public parameter pp. This lets the sender and receiver exchange multiple bits more efficiently, and have shorter keys/randomness, than if the entire system were parallelized. In addition, in deniable mode, the receiver can perform all its secret-key operations (for multiple public keys pk_i) using just the single faking key fk, without keeping any of the random strings $r_{R,i}$ that were used to generate the pk_i, by just resampling $r_{R,i}$ as needed using $I_{\mathsf{Gen}}(fk, pp, pk_i)$. This trivially allows the receiver to decrypt, and to open P-samples and U-samples 'honestly' (without equivocation), in deniable mode.

Note that in the bideniability property above, both experiments use the *deniable* setup algorithm $\mathsf{DenSetup}$, rather than using the normal setup in the left-hand experiment. However, the choice of setup in the left experiment is essentially arbitrary, and the definition would be equivalent if we replaced the first line of the left-hand experiment with a normal setup $pp \leftarrow \mathsf{Setup}(1^n; r_{\mathsf{Setup}})$. This is because the faking key fk is never used, the public parameters are indistinguishable, and Setup has invertible sampling. We chose the presentation above because it yields a more modular proof that one can securely equivocate many independent public key/ciphertext pairs (pk_i, c_i) under a single public parameter pp: first, the bideniability property allows us to replace each pair of calls to the faking algorithms, one by one, with their normal counterparts. Then, finally, the deniable setup can be replaced with a normal setup, as just described.

5.2 Construction of Deniable Encryption

Canetti *et al.* [CDNO97] described a simple encoding trick to construct a multi-distributional sender-deniable encryption scheme from a translucent set: the normal encryption algorithm encodes a 0 message as "UU"

whereas the deniable encryption algorithm encodes it as "PP;" both algorithms encode 1 as "UP." Thus, the sender can always open a deniably generated ciphertext as any message bit, by equivocating zero or more Ps as Us.

The same encoding lets us construct a multi-distributional *bideniable* encryption scheme from a bi-translucent set, since now the sender and receiver are both able to produce 'fake' coins that simultaneously equivocate a P as a U. Using the security properties of BTS, the proof of the following theorem (which we given in the full version) is routine.

Theorem 5.3. *Existence of a bitranslucent set scheme implies existence of a bideniable encryption scheme, secure under chosen-plaintext attacks.*

Canetti *et al.* [CDNO97] also construct a *fully* sender-deniable scheme from a translucent set, where the 'fake' coins can be distinguished with an inverse-polynomial probability related to the public key size. The basic idea is that to encrypt a bit b, the encryption algorithm Enc creates an ℓ-bit sharing $b = \bigoplus_{i\in[\ell]} b_i$ and encodes each $b_i = 0$ as a U-sample and each $b_i = 1$ as a P-sample (all under distinct public keys). To fake, $\mathsf{SendFake}$ chooses a random i such that $b_i = 1$ (which exists with overwhelming probability) and 'flips' it by equivocating the corresponding P-sample in the ciphertext as a U-sample.

For full bideniability, an analogous construction from a bitranslucent set also works, with just one subtlety: the sender and receiver need to equivocate the same share of the message bit. This is trivially achieved in a model in which the sender and receiver can 'coordinate' their fake coins, as described in Section 4. Since it is enough for the sender to decide which share should be flipped, the coordination can be removed by using simulatable public-key encryption, also in exactly the manner described in Section 4.

6 Lattice-Based Bitranslucent Set

In this section we construct a bideniable translucent set scheme based on lattice problems.

6.1 Overview

Here we describe the main ideas behind our construction. To start, it will help to consider a basic mechanism for a (sender-deniable) translucent set. The public key is the description of a lattice Λ, and the secret key is a *short* lattice vector $\mathbf{z} \in \Lambda$. (A lattice is a periodic "grid" of points — formally, a discrete additive subgroup — in $\mathbb{R}^m$.) A P-sample is a point very close to the *dual lattice* Λ^*, while a U-sample is a uniformly random point in a certain large region. (The dual lattice Λ^* is the set of points $\mathbf{v} \in \operatorname{span}(\Lambda)$ for which $\langle \mathbf{z}, \mathbf{v} \rangle \in \mathbb{Z}$ for every $\mathbf{z} \in \Lambda$.) With knowledge of $\mathbf{z}$, it is possible to distinguish these two types of points $\mathbf{c}$ by computing the inner product $\langle \mathbf{z}, \mathbf{c} \rangle$: when $\mathbf{c}$ is close to some $\mathbf{v} \in \Lambda^*$, the inner product $\langle \mathbf{z}, \mathbf{c} \rangle \approx \langle \mathbf{z}, \mathbf{v} \rangle \in \mathbb{Z}$ is close to an integer. On the other hand, when $\mathbf{c}$ is uniform, the inner product is uniformly random modulo $\mathbb{Z}$. Moreover, distinguishing between P-samples and U-samples (given only the public information) is essentially the decision-LWE problem, when the lattice Λ is defined appropriately in relation to the LWE instance. All of this is so far very similar to the structure of lattice-based encryption going back to the seminal scheme of Ajtai and Dwork [AD97], but in that system, the public key uniquely defines a secret key while the ciphertext does not uniquely define the encryption randomness. By contrast, here we have essentially described the 'dual' cryptosystem of Gentry, Peikert, and Vaikuntanathan [GPV08], where there are many short vectors in Λ and hence many valid secret keys, and the ciphertext uniquely specifies the encryption randomness.

To construct a *bitranslucent* set, we exploit and build upon additional techniques from [GPV08]. The faking key for the scheme is a *short basis* $\mathbf{T}$ of Λ (constructed using techniques from [Ajt99, AP09]). As shown in [GPV08], such a basis acts as a 'master trapdoor' that allows for efficiently sampling secret keys under a Gaussian-like distribution, and for efficiently extracting the short offset vector $\mathbf{x}$ from a P-sample $\mathbf{c} = \mathbf{v} + \mathbf{x}$, where $\mathbf{v} \in \Lambda^*$.

Using the above facts, our receiver faking algorithm works as follows: given the short basis $\mathbf{T}$ and a P-sample $\mathbf{c} = \mathbf{v} + \mathbf{x}$ that needs to be 'faked' as a U-sample, the algorithm first extracts $\mathbf{x}$, and then samples a secret key $\mathbf{z}^* \in \Lambda$ that is *highly correlated* with $\mathbf{x}$. By this we mean that $\mathbf{z}^*$ comes from a Gaussian distribution (over Λ) centered at $u \cdot \mathbf{x}$, for some random and large enough correlation coefficient $u \in \mathbb{R}$. Modulo $\mathbb{Z}$, the inner product $\langle \mathbf{z}^*, \mathbf{c} \rangle \approx \langle \mathbf{z}^*, \mathbf{x} \rangle \approx u \cdot \|\mathbf{x}\|^2$, because $\mathbf{z}^* \in \Lambda$ is short. When u is chosen from a wide enough Gaussian distribution, this inner product is uniform (modulo $\mathbb{Z}$), hence $\mathbf{c}$ "looks like" a U-sample when tested with the fake secret key $\mathbf{z}^*$. The natural question at this point is, why should it be secure to release a $\mathbf{z}^*$ that is so correlated with the secret offset vector $\mathbf{x}$? That is, why do $\mathbf{z}^*$ and $\mathbf{c}$ 'look like' an honestly generated secret key and U-sample, respectively? The first key point is that as Gaussian random variables, the correlation between $\mathbf{x}$ and $\mathbf{z}^*$ is essentially *symmetric*. That is, we can consider an alternative experiment in which $\mathbf{z}^*$ is chosen first (according to the normal key generation algorithm), and then $\mathbf{x}$ is chosen from a Gaussian centered at $v \cdot \mathbf{z}^*$ (for some appropriate random $v \in \mathbb{R}$). In both experiments, the pair $(\mathbf{z}^*, \mathbf{x})$ is jointly distributed as a nonspherical Gaussian, with the same covariance. This allows us to switch from the faking experiment to one in which the 'faked' secret key $\mathbf{z}^*$ is generated normally, followed by $\mathbf{c} = \mathbf{v} + \mathbf{x}$ where $\mathbf{v} \in \Lambda^*$ and $\mathbf{x}$ is centered at $v \cdot \mathbf{z}^*$. The second key point is that when $\mathbf{x}$ is correlated with $\mathbf{z}^*$ as described above, it may be written as the sum of two terms: the component $v \cdot \mathbf{z}^*$, and an independent (spherical) Gaussian component $\mathbf{g}$. Under the LWE assumption, we can switch from $\mathbf{c} = (\mathbf{v} + \mathbf{g}) + v \cdot \mathbf{z}^*$ to $\mathbf{c}' = \mathbf{u} + v \cdot \mathbf{z}^*$, where $\mathbf{u}$ is uniformly random. Of course, the latter quantity $\mathbf{c}'$ is truly uniform and independent of the (normally generated) secret key $\mathbf{z}^*$. This coincides exactly with the generation and subsequent honest opening of a normal secret key and U-sample, as desired.

6.2 Background

We briefly recall standard concepts, notation, and results for lattice-based cryptography, as developed in prior works such as [Ajt96, MR04, Reg05, GPV08].

6.2.1 Gaussians

The Gaussian function $\rho\colon \mathbb{R} \to (0,1]$ is defined as $\rho(x) = \exp(-\pi x^2)$; for any $r > 0$, the scaled function $\rho_r(x) = \rho(x/r)$. For any positive integer n, this extends naturally to the product function $\rho_r^n\colon \mathbb{R} \to (0,1]$, defined as $\rho_r(\mathbf{x}) = \prod_{i \in [n]} \rho_r(x_i) = \exp(-\pi \|\mathbf{x}/r\|^2)$. The Gaussian probability distribution D_r^n over $\mathbb{R}^n$ has (normalized) distribution function $r^{-n} \cdot \rho_r^n$.

For any unit vector $\mathbf{u} \in \mathbb{R}^n$, the inner product $\langle \mathbf{u}, D_r^n \rangle$ is distributed as D_r^1. For any real $t > 1$, the Gaussian tail bound says that $\Pr_{x \leftarrow D_r^1}[|x| > t \cdot r] < \exp(-\pi t^2)$; in particular, this quantity is $\text{negl}(n)$ for any $t = \omega(\sqrt{\log n})$. The sum $\mathbf{x} + \mathbf{y}$ of two independent Gaussian-distributed variables $\mathbf{x} \leftarrow D_{r_1}^n$ and $\mathbf{y} \leftarrow D_{r_2}^n$ has Gaussian distribution $D^n_{\sqrt{r_1^2 + r_2^2}}$.

6.2.2 Lattices

A (full-rank) m-dimensional lattice Λ is a discrete additive subgroup of $\mathbb{R}^m$, which is generated as the set of all integer linear combinations of some m linearly independent *basis* vectors. (Throughout this paper, the lattice dimension m is always bounded by a polynomial in the main security parameter n.) The dual lattice $\Lambda^* = \{\mathbf{x} \in \mathbb{R}^m : \langle \mathbf{x}, \mathbf{v} \rangle \in \mathbb{Z}, \forall\ \mathbf{v} \in \Lambda\}$. We are primarily interested in a certain family of *integer* lattices, i.e., subgroups of $\mathbb{Z}^m$, whose importance in cryptography was first established by Ajtai [Ajt96]. A lattice in this family is specified by a "parity check" matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ for the main security parameter n and some positive integer modulus q, as follows:

$$\Lambda^{\perp}(\mathbf{A}) = \left\{ \mathbf{x} \in \mathbb{Z}^m \; : \; \mathbf{A}\mathbf{x} = \mathbf{0} \in \mathbb{Z}_q^n \right\} \subseteq \mathbb{Z}^m.$$

It is easy to check that the dual of this lattice is

$$(\Lambda^{\perp}(\mathbf{A}))^* = \mathbb{Z}^m + \{(\mathbf{A}^t \mathbf{s})/q : \mathbf{s} \in \mathbb{Z}_q^n\}.$$

We need the following two lemmas on generating a short "trapdoor" basis for $\Lambda^{\perp}(\mathbf{A})$, and decoding on its dual under Gaussian error.

Proposition 6.1 ([AP09]). *There is a fixed constant $C > 0$ and a probabilistic polynomial-time algorithm $\mathsf{GenBasis}(1^n, 1^m, q)$ that, for $m \geq Cn \lg q$, outputs $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{T} \in \mathbb{Z}^{m \times m}$ satisfying: $\mathbf{A}$'s distribution is statistically indistinguishable from $U(\mathbb{Z}_q^{n \times m})$; $\mathbf{T}$ is a basis of $\Lambda^{\perp}(\mathbf{A})$; and $\|\widetilde{\mathbf{T}}\| \leq \widetilde{L} = O(\sqrt{n \log q})$.*

Proposition 6.2 ([GPV08]). *There is a deterministic polynomial-time algorithm $\mathsf{Invert}(\mathbf{A}, \mathbf{T}, \mathbf{b})$ that, given any $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, basis $\mathbf{T} \in \mathbb{Z}^{m \times m}$ of $\Lambda^{\perp}(\mathbf{A})$ with $\|\widetilde{\mathbf{T}}\| \cdot \omega(\sqrt{\log n}) \leq 1/\beta$ for some $\beta > 0$, and $\mathbf{b} = (\mathbf{A}^t \mathbf{s})/q + \mathbf{x}$ for arbitrary $\mathbf{s} \in \mathbb{Z}_q^n$ and random $\mathbf{x} \leftarrow D_{\beta}^m$, outputs $\mathbf{x}$ with overwhelming probability (over the choice of $\mathbf{x}$).*

6.2.3 Gaussians on Lattices

For a lattice $\Lambda \subset \mathbb{R}^m$ and an arbitrary $\mathbf{c} \in \mathbb{R}^m$, the *discrete* Gaussian probability distribution $D_{\Lambda+\mathbf{c},r}$ assigns probability proportional to $\rho_r^m(\mathbf{x})$ to each $\mathbf{x} \in \Lambda + \mathbf{c}$ (and zero elsewhere).

For a lattice Λ and a real $\epsilon > 0$, the *smoothing parameter* $\eta_\epsilon(\Lambda) > 0$ is a fundamental lattice quantity governing the behavior of Gaussians over Λ (see [MR04] for the precise definition, which we will not need). We recall several important properties of the smoothing parameter and discrete Gaussians.

Proposition 6.3. *Let $\Lambda \subset \mathbb{R}^m$ be a lattice, let $\mathbf{T}$ be any basis of Λ, and let $\mathbf{c} \in \mathbb{R}^m$ be arbitrary.*

- *[GPV08, Theorem 3.1]: For any $\omega(\sqrt{\log n})$ function, there is a negligible $\epsilon = \operatorname{negl}(n)$ such that $\eta_\epsilon(\Lambda) \leq \|\widetilde{\mathbf{T}}\| \cdot \omega(\sqrt{\log n})$.*
- *[MR04, Lemma 4.1]: If $r \geq \eta_\epsilon(\Lambda)$ for some $\epsilon = \operatorname{negl}(n)$, then each coset $\Lambda + \mathbf{c}$ is equally likely (up to $1 \pm \operatorname{negl}(n)$ multiplicative error) under D_r^m.*
- *[MR04, Lemma 4.4]: If $r \geq \eta_\epsilon(\Lambda)$ for some $\epsilon = \operatorname{negl}(n)$, then $\Pr_{\mathbf{x} \leftarrow D_{\Lambda+\mathbf{c},r}}[\|\mathbf{x}\| > r\sqrt{m}] \leq \operatorname{negl}(n)$.*
- *[GPV08, Theorem 4.1]: there is a PPT algorithm $\mathsf{SampleD}(\mathbf{T}, \mathbf{c}, r)$ that, given arbitrary $\mathbf{c} \in \mathbb{R}^m$ and $r \geq \|\widetilde{\mathbf{T}}\| \cdot \omega(\sqrt{\log n})$, generates a sample from $D_{\Lambda+\mathbf{c},r}$ (up to $\operatorname{negl}(n)$ statistical distance).*

We will also need the following lemma on 'convolutions' of continuous and discrete Gaussians.

Lemma 6.4 ([Pei10, Special case of Theorem 3.1]). *Let $\Lambda \subset \mathbb{R}^m$ be a lattice and let $r \geq \eta_\epsilon(\Lambda)$ for some $\epsilon = \operatorname{negl}(n)$. The output $\mathbf{z}$ of the experiment $\mathbf{y} \leftarrow D_r^m$, $\mathbf{z} \leftarrow \mathbf{y} + D_{\Lambda-\mathbf{y},r}$ has distribution $D_{\Lambda,r\sqrt{2}}$, up to $\operatorname{negl}(n)$ statistical distance. Moreover, the conditional distribution of $\mathbf{y}$ given $\mathbf{z} = \bar{\mathbf{z}} \in \Lambda$ is $\bar{\mathbf{z}}/2 + D_{r/\sqrt{2}}^m$, up to $\operatorname{negl}(n)$ statistical distance.*

6.2.4 Learning with Errors

Let $\mathbb{T} = \mathbb{R}/\mathbb{Z}$ be the additive group on the real interval $[0, 1)$ with modulo 1 addition. For positive integers n and $q \geq 2$, a vector $\mathbf{s} \in \mathbb{Z}_q^n$, and a probability distribution ϕ on $\mathbb{R}$, define $A_{\mathbf{s},\phi}$ to be the distribution on $\mathbb{Z}_q^n \times \mathbb{T}$ obtained by choosing a vector $\mathbf{a} \in \mathbb{Z}_q^n$ uniformly at random, choosing an error term $x \in \mathbb{R}$ according to ϕ, and outputting $(\mathbf{a}, b = \langle \mathbf{a}, \mathbf{s} \rangle / q + x)$, where the addition is performed in $\mathbb{T}$.

The decision-$\mathsf{LWE}_{q,\phi}$ assumption says that for a uniformly random (secret) $\mathbf{s} \in \mathbb{Z}_q^n$, the distribution $A_{\mathbf{s},\phi}$ is pseudorandom (i.e., indistinguishable from $U(\mathbb{Z}_q^n \times \mathbb{T})$) given any desired number of samples. We write decision-$\mathsf{LWE}_{q,\alpha}$ to abbreviate the common special case where $\phi = D_\alpha$ is a Gaussian. It is known that for certain choices of the parameters, solving decision-LWE is as hard as solving certain problems on n-dimensional lattices in the *worst case* [Reg05, Pei09].

The extended-LWE assumption. We prove the security of our BTS construction under what we call the *extended*-$\mathsf{LWE}_{q,\beta,r}$ assumption, defined here. For certain choices of the error rates α and β (and parameter r), the standard-$\mathsf{LWE}_{q,\alpha}$ assumption actually implies the extended-$\mathsf{LWE}_{q,\beta,r}$ assumption, so our results hold under an appropriate version of standard LWE. However, the ratio of β to α is quite loose (slightly super-polynomial), and so basing security on standard LWE alone involves a rather strong assumption. The advantages of using extended-LWE in our setting are twofold: first, it abstracts away a technical issue that is secondary to our main proof techniques; and second, it allows us to prove security using a much larger noise rate (and thus potentially weaker assumption) than if the proof had been based directly on standard-LWE. While we do not know of a formal reduction between standard- and extended-LWE when the ratio of β to α is only $\mathrm{poly}(n)$, it seems reasonable that extended-LWE could be substantially harder than standard-LWE when $\beta \gg \alpha$. It is also plausible that a connection to worst-case lattice problems (for small polynomial approximation factors) could be shown for extended-LWE; we leave this as an open problem.

The extended-LWE assumption is parameterized by q and β as before (we restrict to $\phi = D_\beta$ for simplicity), as well as some $r = \omega(\sqrt{\log n})$. It says that the following two games, each consisting of two stages, are computationally indistinguishable.

The first game is defined as follows. In the first stage, the distinguisher is given any desired number m of independent samples $(\mathbf{a}_i, b_i = \langle \mathbf{a}_i, \mathbf{s} \rangle / q + x_i)$ drawn from $A_{\mathbf{s},\beta}$, for uniformly random secret $\mathbf{s} \in \mathbb{Z}_q^n$. Group these samples as $(\mathbf{A}, \mathbf{b} = (\mathbf{A}^t \mathbf{s})/q + \mathbf{x})$ in the natural way. Then in the second stage, the distinguisher is given a small 'hint' about the error vector $\mathbf{x}$ in the form of a single sample $(\mathbf{z} \in \mathbb{Z}^m, \langle \mathbf{z}, \mathbf{x} \rangle + x' \bmod 1)$, where $\mathbf{z} \leftarrow D_{\mathbb{Z}^m,r}$ and $x' \leftarrow D_\beta$. Because the distinguisher can compute $\langle \mathbf{z}, \mathbf{b} \rangle$ on its own, the second phase may equivalently seen as providing $\mathbf{z}$ and, for $\mathbf{u} = \mathbf{A}\mathbf{z} \in \mathbb{Z}_q^n$, the value

$$b' = \langle \mathbf{u}, \mathbf{s} \rangle / q + x' = \langle \mathbf{z}, \mathbf{b} - \mathbf{x} \rangle + x' \in \mathbb{T}.$$

The second game is defined similarly: in the first stage, the distinguisher is given m independent samples $(\mathbf{a}_i, b_i)$ drawn from the uniform distribution $U(\mathbb{Z}_q^n \times \mathbb{T})$, and with each sample we also draw an $x_i \leftarrow D_\beta$. Then in the second stage, the distinguisher is given a single sample $(\mathbf{z}, b') \in \mathbb{Z}^m \times \mathbb{T}$, where $\mathbf{z} \leftarrow D_{\mathbb{Z}^m,r}$ and $b' = \langle \mathbf{z}, \mathbf{b} - \mathbf{x} \rangle + x' \in \mathbb{T}$ for $x' \leftarrow D_\beta$. (Note that the second stage should *not* simply choose b' uniformly, because the assumption would in fact be *false*: a distinguisher could just test whether $b' \approx \langle \mathbf{z}, \mathbf{b} \rangle$, which holds with high probability in the first game, but not when b' is uniform.)

Observe that if $\alpha = \mathrm{negl}(n)$ is any negligible function and any $\beta = 1/\mathrm{poly}(n)$, $r = \mathrm{poly}(n)$, then the standard-$\mathsf{LWE}_{q,\alpha}$ assumption implies the extended-LWE assumption via a straightforward reduction, because the D_β noise term in the second stage statistically conceals the inner product $\langle \mathbf{z}, \mathbf{x} \rangle$, which has negligible magnitude. Therefore, the reduction can simulate the second stage on its own. (A similar idea has been used in several other recent works on leakage resilience [GKPV10, ADN$^+$10].) We also remark that we could let the noise rate in the second stage be different (e.g., larger) than that of the first stage, and our construction and proof are easily adapted.

6.3 BTS Construction

We now construct a bitranslucent set scheme based on the extended-LWE problem. Our scheme involves a number of parameters, which are implicitly functions of the main security parameter n:

- the modulus q, number of samples m, and error parameter $\beta \in (0,1)$ relate to the underlying LWE problem (note that security is actually based on extended-LWE with error rate $\beta' = \beta/\sqrt{2}$; see below);
- the $\mathrm{poly}(n)$-bounded Gaussian parameter $r = \widetilde{L} \cdot \omega(\sqrt{\log n})$ is used for choosing the (real and fake) secret keys, where $\widetilde{L}$ is the quality bound of GenBasis from Proposition 6.1;
- the correlation parameter κ is used in FakeRCoins.

We first describe the scheme, then account for all the constraints that the parameters should satisfy. For convenience, we define the scheme using real numbers and continuous error terms, which can be approximated arbitrarily well using sufficient precision. Alternatively, we can use the 'convolution theorem' from [Pei10] to discretize the continuous error terms to become discrete Gaussians, and slightly adjust the receiver faking algorithms to 'smooth out' this discretization. Because this would further complicate the already heavy analysis, we opt for the simpler approach of using continuous Gaussians.

- $\mathsf{Setup}(1^n; r_{\mathsf{Setup}})$: using randomness r_{Setup}, output $pp = \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$.

  (Note that there is trivial sampling algorithm $I_{\mathsf{Setup}}(\mathbf{A})$ that outputs random r^*_{Setup} consistent with $\mathbf{A}$.)
- $\mathsf{DenSetup}(1^n)$: generate $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{GenBasis}(1^n, 1^m, q)$, and output $pp = \mathbf{A}$ and $fk = \mathbf{T}$.
- $\mathsf{Gen}(\mathbf{A}; r_R)$: using randomness r_R, choose $\mathbf{z} \leftarrow D_{\mathbb{Z}^m, r\sqrt{2}}$ (via $\mathsf{SampleD}(\{\mathbf{e}_i\}_{i\in[m]}, \mathbf{0}, r\sqrt{2})$), and output $pk = \mathbf{u} = \mathbf{A}\mathbf{z} \in \mathbb{Z}_q^n$. Observe that $\bar{\mathbf{A}}\bar{\mathbf{z}} = \mathbf{0} \in \mathbb{Z}_q^n$, where $\bar{\mathbf{A}} = [\mathbf{A} \mid \mathbf{u}]$ and $\bar{\mathbf{z}} = (\mathbf{z}, -1) \in \mathbb{Z}^{m+1}$. Below we identify the randomness r_R with $\mathbf{z}$.

  (The trapdoor sampling algorithm $I_{\mathsf{Gen}}(\mathbf{T}, \mathbf{u})$ just outputs $\mathbf{z} \leftarrow \mathsf{SampleD}(\mathbf{T}, \hat{\mathbf{z}}, r\sqrt{2})$, where $\hat{\mathbf{z}} \in \mathbb{Z}^m$ is any solution to $\mathbf{A}\hat{\mathbf{z}} = \mathbf{u}$.)
- $\mathsf{SampleP}(\mathbf{A}, \mathbf{u}; r_S)$: Let $\bar{\mathbf{A}} = [\mathbf{A} \mid \mathbf{u}] \in \mathbb{Z}_q^{n\times(m+1)}$. Using randomness r_S, choose $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ and $\bar{\mathbf{x}} \leftarrow D_\beta^{m+1}$, and output $\bar{\mathbf{c}} = (\bar{\mathbf{A}}^t\mathbf{s})/q + \bar{\mathbf{x}} \in \mathbb{T}^{m+1}$. Below we identify r_S with $(\mathbf{s}, \bar{\mathbf{x}})$.
- $\mathsf{SampleU}(\mathbf{A}, \mathbf{u}; r_S)$: using randomness r_S, output a uniformly random $\bar{\mathbf{c}} \leftarrow \mathbb{T}^{m+1}$.
- $\mathsf{TestP}(\mathbf{A}, \bar{\mathbf{z}}, \bar{\mathbf{c}})$: if $\langle \bar{\mathbf{z}}, \bar{\mathbf{c}} \rangle \in \mathbb{T}$ is closer to 0 (modulo 1) than to $\frac{1}{2}$, accept; otherwise, reject.
- $\mathsf{FakeSCoins}(\mathbf{A}, \mathbf{u}, (\mathbf{s}, \bar{\mathbf{x}}))$: simply output $\bar{\mathbf{c}} = \mathsf{SampleP}(\mathbf{A}, \mathbf{u}; (\mathbf{s}, \bar{\mathbf{x}})) \in \mathbb{T}^{m+1}$ as the random coins that would cause SampleU to output $\bar{\mathbf{c}}$.[3]
- $\mathsf{FakeRCoins}(\mathbf{A}, \mathbf{T}, \mathbf{u}, \bar{\mathbf{c}})$: let $\mathbf{z} \leftarrow I_{\mathsf{Gen}}(\mathbf{T}, \mathbf{u})$ be a properly distributed secret key for $\mathbf{u}$. Then:
  1. Parse $\bar{\mathbf{c}}$ as $(\mathbf{c}, c_{m+1}) \in \mathbb{T}^m \times \mathbb{T}$, and let $\mathbf{x} = \mathsf{Invert}(\mathbf{A}, \mathbf{T}, \mathbf{c})$.

[3]Note that FakeSCoins really only needs the ciphertext $\bar{\mathbf{c}}$, not the randomness $(\mathbf{s}, \bar{\mathbf{x}})$ or $pp = \mathbf{A}$, $pk = \mathbf{u}$ used to generate $\bar{\mathbf{c}}$.

2. Choose a 'correlation coefficient' $u \leftarrow D_{\kappa(r/\beta)^2}$, and choose $\mathbf{y} \leftarrow u\mathbf{x} + D^m_{\sqrt{r^2-(\beta u)^2}}$.

   (In Lemma 6.6 we prove that $(\beta u)^2 < r^2$ almost always, so $\mathbf{y}$'s distribution is well-defined.)

3. Output $\mathbf{z}^* \leftarrow \mathbf{y} + D_{\Lambda+\mathbf{z}-\mathbf{y},r}$, chosen using $\mathsf{SampleD}(\mathbf{T}, \mathbf{z}-\mathbf{y}, r)$, where $\Lambda = \Lambda^\perp(\mathbf{A})$.

We now discuss the constraints on all the parameters of the scheme. The parameter κ controls the amount of correlation in FakeRCoins, and should satisfy the following bounds:

$$\frac{\omega(\sqrt{\log n})}{mr^2} \quad \leq \quad \kappa \quad \leq \quad \frac{\beta}{r \cdot \omega(\sqrt{\log n})}. \tag{6.1}$$

The lower bound (used in Claim 6.12) ensures a large enough correlation between the faked $\mathbf{z}^*$ and the error term in a P-sample $\bar{\mathbf{c}}$, so that $\bar{\mathbf{c}}$ appears uniform when tested with $\mathbf{z}^*$. The upper bound (used in Lemma 6.6) ensures that $r > |\beta u|$ with overwhelming probability (over the choice of u), so that the 'leftover' Gaussian distribution in the choice of $\mathbf{y}$ by FakeRCoins is well-defined.

We also have the following sufficient conditions for the (inverse) error parameter β:

$$r\sqrt{2(m+1)} \cdot \omega(\sqrt{\log n}) \quad \leq \quad 1/\beta \quad \leq \quad \frac{rm}{\omega(\log n)}. \tag{6.2}$$

The lower bound (used in Lemma 6.6) ensures that TestP accepts a P-sample with high probability, and because $r \geq \widetilde{L} \cdot \omega(\sqrt{\log n})$ it also suffices for the correctness of Invert. The upper bound is needed to make the above bounds on κ satisfiable. Note that there is an $\tilde{\Omega}(\sqrt{m})$ multiplicative gap between the upper and lower bounds, so they are satisfiable for large enough m.

We base the security of our BTS on the extended-$\mathsf{LWE}_{q,\beta',r\sqrt{2}}$ assumption for an error rate $\beta' = \beta/\sqrt{2}$. With this choice, we set $1/\beta$ according to its lower bound, let $q \in [n, 2n]/\beta'$, and let $m = \Theta(n \log q) = \tilde{O}(n)$ be large enough to apply Proposition 6.1. The inverse noise parameter in our assumption is therefore $1/\beta' = \tilde{O}(n)$.

6.4 Correctness and Security

Theorem 6.5. *Under the extended-*$\mathsf{LWE}_{q,\beta',r\sqrt{2}}$ *assumption for $\beta' = \beta/\sqrt{2}$, the above algorithms form a secure bitranslucent set scheme according to Definition 5.1.*

Proof. Lemma 6.6 below demonstrates Property 1 (correctness). Property 2 (statistical indistinguishability of the public parameters $\mathbf{A}$ generated by Setup and DenSetup) follows directly by Proposition 6.1. Property 3 (bideniability) is proved in Lemma 6.7 below. □

Lemma 6.6. *For parameters satisfying the above constraints, BTS is well-defined and correct, i.e., it satisfies Property 1 of Definition 5.1.*

Proof. It is apparent that FakeSCoins and FakeRCoins produce correct outputs. To ensure that FakeRCoins is well-defined, it suffices show that $(\beta u)^2 < r^2/2$ with overwhelming probability when $u \leftarrow D_{\kappa(r/\beta)^2}$. By the Gaussian tail bound and the hypothesis that $\kappa r/\beta \leq 1/\omega(\sqrt{\log n})$,

$$|\beta u| < \kappa r^2 \cdot \omega(\sqrt{\log n})/(\beta\sqrt{2}) \leq r/\sqrt{2}$$

with overwhelming probability (where $\omega(\sqrt{\log n})$ denotes the same function in both places).

```
Experiment $H_0$:
  $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{GenBasis}(1^n, 1^m, q)$
  $\mathbf{x} \leftarrow D_\beta^m$
  $\mathbf{y} \leftarrow u\mathbf{x} + D^m_{\sqrt{r^2-(\beta u)^2}}$
    for $u \leftarrow D_{\kappa(r/\beta)^2}$
  $\mathbf{z} \leftarrow D_{\mathbb{Z}^m,r}$;  $\mathbf{u} = \mathbf{Az}$
  $\mathbf{z}^* \leftarrow \mathbf{y} + D_{\Lambda+\mathbf{z}-\mathbf{y},r}$
  $\mathbf{c} = (\mathbf{A}^t\mathbf{s})/q + \mathbf{x} \in \mathbb{T}^m$ for $\mathbf{s} \leftarrow \mathbb{Z}_q^n$
  $c_{m+1} \leftarrow \langle \mathbf{u}, \mathbf{s}\rangle/q + D_\beta \in \mathbb{T}$
  Return $(\mathbf{A}, \bar{\mathbf{c}}, \mathbf{z}^*)$
```

```
Experiment $H_1$:
  $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$
  $\mathbf{x}, \mathbf{y}$: as in $H_0$

  $\mathbf{z}^* \leftarrow \mathbf{y} + D_{\mathbb{Z}^m-\mathbf{y},r}$
  $\mathbf{u} = \mathbf{Az}^*$
  $\mathbf{c}, c_{m+1}$: as in $H_0$

  Return $(\mathbf{A}, \bar{\mathbf{c}}, \mathbf{z}^*)$
```

```
Experiment $H_2$:
  $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$
  $\mathbf{y} \leftarrow D_r^m$
  $\mathbf{x} \leftarrow D^m_{\sqrt{\beta^2-(rv)^2}} + v\mathbf{y}$
    for $v \leftarrow D_\kappa$

  $\mathbf{z}^*, \mathbf{u}, \mathbf{c}, c_{m+1}$:
    as in $H_1$

  Return $(\mathbf{A}, \bar{\mathbf{c}}, \mathbf{z}^*)$
```

```
Experiment $H_3$:
  $\mathbf{A}, v$: as in $H_2$
  $\mathbf{z}^* \leftarrow D_{\mathbb{Z}^m,r\sqrt{2}}$,  $\mathbf{u} = \mathbf{Az}^*$
  $\mathbf{y} \leftarrow \mathbf{z}^*/2 + D^m_{r/\sqrt{2}}$
  $\mathbf{x}^{(1)} \leftarrow D^m_{\beta'}$, $\mathbf{x}^{(2)} \leftarrow D^m_{\sqrt{\beta^2-\beta'^2-(rv)^2}}$
  $\mathbf{c} = (\mathbf{A}^t\mathbf{s})/q + \mathbf{x}^{(1)} + \mathbf{x}^{(2)} + v\mathbf{y} \in \mathbb{T}^m$
    for $\mathbf{s} \leftarrow \mathbb{Z}_q^n$
  $c_{m+1} \leftarrow \langle \mathbf{u}, \mathbf{s}\rangle/q + D_{\beta'} \in \mathbb{T}$
  Return $(\mathbf{A}, \bar{\mathbf{c}}, \mathbf{z}^*)$
```

```
Experiment $H_4$:
  $\mathbf{A}, v, \mathbf{z}^*, \mathbf{u}$,
    $\mathbf{y}, \mathbf{x}^{(1)}, \mathbf{x}^{(2)}$:
    as in $H_3$
  $\mathbf{b} \leftarrow U(\mathbb{T}^m)$
  $\mathbf{c} = \mathbf{b} + \mathbf{x}^{(2)} + v\mathbf{y} \in \mathbb{T}^m$
  $c_{m+1} \leftarrow$
    $\langle \mathbf{z}^*, \mathbf{b} - \mathbf{x}^{(1)}\rangle + D_\beta \in \mathbb{T}$
  Return $(\mathbf{A}, \bar{\mathbf{c}}, \mathbf{z}^*)$
```

```
Experiment $H_5$:
  $\mathbf{A}, \mathbf{z}^*, \mathbf{u}$:
    as in $H_4$

  $\mathbf{c} \leftarrow U(\mathbb{T}^m)$

  $c_{m+1} \leftarrow U(\mathbb{T})$
  Return $(\mathbf{A}, \bar{\mathbf{c}}, \mathbf{z}^*)$
```

Figure 3: Hybrid experiments for showing bideniability of BTS.

We now show that TestP is correct. First suppose that $\bar{\mathbf{c}}$ is generated by $\mathsf{SampleU}(\mathbf{A}, \mathbf{u})$, i.e., it is uniform in $\mathbb{T}^{m+1}$. Then because $\bar{\mathbf{z}} = (\mathbf{z}, -1)$ has -1 as its final coordinate, the inner product $\langle \bar{\mathbf{z}}, \bar{\mathbf{c}}\rangle \in \mathbb{T}$ is uniformly random. Thus TestP rejects with probability $1/2$ (which can be amplified via repetition, as described in Remark 5.2).

Now suppose that $\bar{\mathbf{c}}$ is generated by $\mathsf{SampleP}(\mathbf{A}, \mathbf{u})$, i.e., $\bar{\mathbf{c}} = (\bar{\mathbf{A}}^t\mathbf{s})/q + \bar{x} \bmod 1$ for $\bar{\mathbf{A}} = [\mathbf{A} \mid \mathbf{u}]$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, and $\bar{\mathbf{x}} \leftarrow D_\beta^{m+1}$. Observe that

$$\langle \bar{\mathbf{z}}, \bar{\mathbf{c}}\rangle = (\bar{\mathbf{A}}\bar{\mathbf{z}})^t\mathbf{s}/q + \langle \bar{\mathbf{z}}, \bar{\mathbf{x}}\rangle = \langle \bar{\mathbf{z}}, \bar{\mathbf{x}}\rangle \bmod 1,$$

because $\bar{\mathbf{A}}\bar{\mathbf{z}} = \mathbf{0} \bmod q$. So $\langle \bar{\mathbf{z}}, \bar{\mathbf{c}}\rangle$ is distributed as $D_{\beta\|\bar{\mathbf{z}}\|}$. By Proposition 6.3, $\|\bar{\mathbf{z}}\| \le r\sqrt{2(m+1)}$ with overwhelming probability, hence by the Gaussian tail bound and the hypothesis that $1/\beta \ge r\sqrt{2(m+1)} \cdot \omega(\sqrt{\log n})$, we conclude that $|\langle \bar{\mathbf{z}}, \bar{\mathbf{x}}\rangle| < 1/2$ with overwhelming probability, and TestP accepts. □

Lemma 6.7. *Under the extended-*$\mathsf{LWE}_{q,\beta',r\sqrt{2}}$ *assumption, BTS is bideniable, i.e., it satisfies Property 3 of Definition 5.1.*

Proof. First, notice that because SampleU simply outputs its random coins as a uniformly random $\bar{\mathbf{c}} \in \mathbb{T}^m$, we can use $\bar{\mathbf{c}}$ itself as the coins. Similarly, for the receiver's (fake) coins we may simply use $\bar{\mathbf{z}}$ or $\bar{\mathbf{z}}^*$.

We prove bideniability using a sequence of experiments H_i, which are defined precisely in Figure 3. Here we describe their main properties and the relations between adjacent hybrids.

- Experiment H_0 is statistically close to the view of the adversary in the right-hand 'faking' experiment in the definition of bideniability. We have rearranged the order of some operations to better coincide with the next experiment. We have also used the fact that Invert successfully recovers $\mathbf{x}$ from $\mathbf{c}$ with overwhelming probability over all the randomness in the experiment.

- In Experiment H_1, we no longer generate a trapdoor basis $\mathbf{T}$ with $\mathbf{A}$ for sampling the faked secret key $\mathbf{z}^*$. Instead, we directly choose $\mathbf{z}^*$ (from $\mathbb{Z}^m$) and then define $\mathbf{u}$ to match it. (The ciphertext components $\mathbf{c}$, c_{m+1} are constructed in the same way, however.) In Claim 6.8, we show that experiments H_0 and H_1 are statistically indistinguishable.

- In Experiment H_2, we choose $\mathbf{x}$ and $\mathbf{y}$ in a different order, with $\mathbf{x}$ now depending on $\mathbf{y}$ instead of the other way around. In Claim 6.9, we show that the joint distribution of $(\mathbf{x}, \mathbf{y})$ remains exactly the same (statistically), because in both cases it is a certain linear transformation of a spherical Gaussian (i.e., has the same covariance). Therefore, experiments H_1 and H_2 are statistically identical.

- In Experiment H_3, we choose $\mathbf{z}^*$ and $\mathbf{y}$ in a different order, with $\mathbf{y}$ now depending on $\mathbf{z}^*$ instead of the other way around. By Lemma 6.4, the joint distributions of $(\mathbf{y}, \mathbf{z}^*)$ in H_2 and H_3 are statistically indistinguishable. In preparation for the next experiment, we also break the error term $D^m_{\sqrt{\beta^2-(rv)^2}}$ from H_2 into two terms $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ whose sum has the same distribution. Claim 6.10 formally proves that H_2 and H_3 are statistically indistinguishable.

- In Experiment H_4, we replace the $(\mathbf{A}^t\mathbf{s})/q + \mathbf{x}^{(1)}$ component of $\mathbf{c}$ with a uniformly random $\mathbf{b} \leftarrow \mathbb{T}^m$, and replace the $\langle \mathbf{u}, \mathbf{s}\rangle/q + D_{\beta'}$ component of c_{m+1} with $\langle \mathbf{z}^*, \mathbf{b} - \mathbf{x}^{(1)}\rangle + D_{\beta'}$. (Note that c_{m+1} also has some 'leftover' Gaussian error, because $\beta' < \beta$.) Under the extended-$\mathsf{LWE}_{q,\beta'}$ assumption, this change cannot be detected by any efficient adversary, i.e., experiments H_3 and H_4 are computationally indistinguishable. The (straightforward) reduction is described in Claim 6.11.

- In Experiment H_5, we rewrite $\mathbf{c}$ itself as uniformly random over $\mathbb{T}^m$, and focus on the $\langle \mathbf{z}^*, v\mathbf{y}\rangle = v\langle \mathbf{z}^*, \mathbf{y}\rangle$ term that moves to c_{m+1} when rewriting $\mathbf{b} = \mathbf{c} - \mathbf{x}^{(2)} - v\mathbf{y}$ in H_4. In Claim 6.12, we show that $\langle \mathbf{z}^*, \mathbf{y}\rangle = \Omega(mr^2)$ with overwhelming probability. Recalling that $\kappa = \frac{\omega(\sqrt{\log n})}{mr^2}$, it follows that for almost all choices of $\mathbf{z}^*$ and $\mathbf{y}$ (treated as fixed), the quantity $v\langle \mathbf{z}^*, \mathbf{y}\rangle$ is distributed as $D_{\omega(\sqrt{\log n})}$ *solely over the random choice of* $v \leftarrow D_\kappa$. It is therefore statistically close to uniform modulo 1 (and independent of the other variables in the adversary's view), by Proposition 6.3. We conclude that experiments H_4 and H_5 are statistically indistinguishable.

Finally, observe that experiment H_5 is exactly the 'honest opening' experiment in the definition of bideniability. This completes the proof. □

We now precisely state and prove the claims supporting the proof of Lemma 6.7.

Claim 6.8. *Experiments H_0 and H_1 are statistically indistinguishable.*

Proof. In both experiments, $\mathbf{y}$ and $\bar{\mathbf{c}}$ have the same distribution (respectively), and for fixed $\mathbf{A}$, the value $\mathbf{u}$ is statistically close to uniform over the subgroup of $\mathbb{Z}^m$ generated by the columns of $\mathbf{A}$, by Proposition 6.3. Now for analyzing both experiments, we condition on the choice of $\mathbf{u}$, and let $\mathbf{z} \in \mathbb{Z}^m$ denote *any* solution to $\mathbf{A}\mathbf{z} = \mathbf{u}$. In H_0, the vector $\mathbf{z}^* - \mathbf{y}$ is distributed as $D_{\Lambda+\mathbf{z}-\mathbf{y},r}$ by construction. In H_1, the vector $\mathbf{z}^* - \mathbf{y}$ is first chosen according to a discrete Gaussian (of parameter r) over $\mathbb{Z}^m - \mathbf{y}$, but then conditioning on the value of $\mathbf{u}$ restricts its support to $\Lambda + \mathbf{z} - \mathbf{y}$. Therefore, given $\bar{\mathbf{A}}$, the conditional distribution of $\mathbf{z}^* - \mathbf{y}$ is $D_{\Lambda+\mathbf{z}-\mathbf{y},r}$, as desired. □

Claim 6.9. *Experiments H_1 and H_2 are statistically identical.*

Proof. We first point out that in H_2, the distribution $D_{\sqrt{\beta^2-(rv)^2}}$ is well-defined with overwhelming probability over the choice of v, by the Gaussian tail bound. (The proof that $\beta^2 - (rv)^2 > \beta^2/2 = \beta'$ is virtually identical to the one in the proof of Lemma 6.6.)

The only difference between the two experiments is in the choice of $\mathbf{x}$ and $\mathbf{y}$. We show that the joint distributions of $(\mathbf{x}, \mathbf{y})$ are *identical* in both.

By construction, in both experiments the coordinates x_i of $\mathbf{x}$ are entirely independent and identically distributed, as are the coordinates y_i of $\mathbf{y}$. Therefore, it suffices to show that the joint distributions of $(x, y) = (x_i, y_i)$ for an arbitrary $i \in [m]$ are identical. We do so for any *fixed* $u \in \mathbb{R}$ and $v = \frac{u}{(r/\beta)^2}$ for which $s^2 := r^2 - (\beta u)^2 > 0$; equivalently, for $t^2 := (\beta/r)^2 \cdot s^2 = \beta^2 - (rv)^2 > 0$. Because u and v related in this way are equally likely in H_1 and H_2 (respectively), and $s^2 > 0$ with overwhelming probability over the choice of u, the claim follows by averaging over the choice of u, v.

In experiment H_1, we may write the distribution of (x, y) as $\mathbf{T}_1 \cdot D_1^2$, where $\mathbf{T}_1 = \left(\begin{smallmatrix} \beta & 0 \\ \beta u & s \end{smallmatrix}\right)$. In experiment H_2, the distribution of (x, y) is $\mathbf{T}_2 \cdot D_1^2$, where $\mathbf{T}_2 = \left(\begin{smallmatrix} t & vr \\ 0 & r \end{smallmatrix}\right)$. Observe that

$$\mathbf{T}_1\mathbf{T}_1^t = \begin{pmatrix} \beta^2 & \beta^2 u \\ \beta^2 u & (\beta u)^2 + s^2 \end{pmatrix} = \begin{pmatrix} t^2 + (rv)^2 & r^2 v \\ r^2 v & r^2 \end{pmatrix} = \mathbf{T}_2\mathbf{T}_2^t.$$

It follows that $(\mathbf{T}_2^{-1}\mathbf{T}_1)(\mathbf{T}_2^{-1}\mathbf{T}_1)^t = \mathbf{I}$, i.e., $\mathbf{T}_1 = \mathbf{T}_2\mathbf{Q}$ for some orthogonal matrix $\mathbf{Q}$. Because the spherical Gaussian D_1^2 is invariant under orthogonal (rigid) transformations, $\mathbf{T}_1 \cdot D_1^2 = \mathbf{T}_2\mathbf{Q} \cdot D_1^2 = \mathbf{T}_2 \cdot D_1^2$. □

Claim 6.10. *Experiments H_2 and H_3 are statistically indistinguishable.*

Proof. The main difference between H_2 and H_3 is in the choice of $\mathbf{y}$ and $\mathbf{z}^*$: in the former experiment, $\mathbf{z}^*$ depends on $\mathbf{y}$ and in the latter, $\mathbf{y}$ depends on $\mathbf{z}^*$. By Lemma 6.4 with $r \geq \|\widetilde{\mathbf{T}}\| \cdot \omega(\sqrt{\log n}) \geq \eta_\epsilon(\Lambda^\perp(\mathbf{A}))$ for some negligible ϵ, the joint distributions of $(\mathbf{y}, \mathbf{z}^*)$ in the two experiments are statistically indistinguishable.

In preparation for the next hybrid, we have also broken the error term $D_{\sqrt{\beta^2-(rv)^2}}$ in $\mathbf{x}$ into the sum of two independent terms $D_{\beta'}^m$, $D_{\sqrt{\beta^2-\beta'^2-(rv)^2}}$, whose sum has the desired distribution. (Also recall that $\beta^2 - (rv)^2 > \beta'^2$ with overwhelming probability over the choice of v, so both terms are well-defined.) □

Claim 6.11. *Under the extended-*$\mathsf{LWE}_{q,\beta'}$ *assumption, H_3 and H_4 are computationally indistinguishable.*

Proof. The reduction is via a straightforward simulation. The simulator plays one of the two games from the extended-LWE assumption, and emulates either H_3 or H_4 as follows:

- In the first stage of the game, draw m samples $(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^n \times \mathbb{T}$ and group them as $(\mathbf{A}, \mathbf{b}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{T}^m$ in the natural way.
- In the second stage of the game, draw one sample $(\mathbf{z}^*, b_{m+1}) \in \mathbb{Z}^m \times \mathbb{T}$, and let $\mathbf{u} = \mathbf{A}\mathbf{z}^*$.
- Choose v, $\mathbf{y}$, and $\mathbf{x}^{(2)}$ as in H_3 (and H_4), let $\mathbf{c} = \mathbf{b} + \mathbf{x}^{(2)} + v\mathbf{y} \in \mathbb{T}^m$, and let $c_{m+1} \leftarrow b_{m+1} + D_{\beta'} \in \mathbb{T}$. Output $(\bar{\mathbf{A}}, \bar{\mathbf{c}}, \mathbf{z}^*)$.

It is straightforward to check that if the simulator is playing the first extended-$\mathsf{LWE}_{q,\beta',r\sqrt{2}}$ game (i.e., $\mathbf{A}$ is uniform; $\mathbf{b} = (\mathbf{A}^t\mathbf{s})/q + \mathbf{x}^{(1)} \in \mathbb{T}$ for $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ and $\mathbf{x}^{(1)} \leftarrow D_{\beta'}^m$; $\mathbf{z}^* \leftarrow D_{\mathbb{Z}^m, r\sqrt{2}}$; and $b_{m+1} \leftarrow \langle \mathbf{u}, \mathbf{s}\rangle/q + D_{\beta'} \in \mathbb{T}$), then the simulation emulates H_3 perfectly. Similarly, if the simulator is playing the second game (i.e., $(\mathbf{A}, \mathbf{b})$ is uniform; $\mathbf{z}^* \leftarrow D_{\mathbb{Z}^m, r\sqrt{2}}$; and $b_{m+1} = \langle \mathbf{z}^*, \mathbf{b} - \mathbf{x}^{(1)}\rangle + D_{\beta'} \in \mathbb{T}$), then the simulation emulates H_4 perfectly. By assumption, the two games are computationally indistinguishable, and because the simulator is efficient, then so are H_3 and H_4. □

Claim 6.12. *Experiments H_4 and H_5 are statistically indistinguishable.*

Proof. The two experiments choose $\mathbf{A}$, $\mathbf{y}$, $\mathbf{z}^*$, and $\mathbf{u}$ in exactly the same way; the only difference is in the choice of $\bar{\mathbf{c}} = (\mathbf{c}, c_{m+1})$. In experiment H_4, we can rewrite this choice as $\mathbf{c} \leftarrow U(\mathbb{T}^m)$,

$$c_{m+1} \leftarrow \langle \mathbf{z}^*, \mathbf{c} - \mathbf{x}^{(1)} - \mathbf{x}^{(2)} - v\mathbf{y} \rangle + D_\beta = \langle \mathbf{z}^*, \mathbf{c} - \mathbf{x}^{(1)} - \mathbf{x}^{(2)} \rangle - v\langle \mathbf{z}^*, \mathbf{y} \rangle + D_\beta \in \mathbb{T},$$

where $v \leftarrow D_\kappa$. In the remainder of the proof we show that with overwhelming probability over the choice of $\mathbf{z}^*$ and $\mathbf{y}$, the $v\langle \mathbf{z}^*, \mathbf{y} \rangle$ term above is statistically close to uniform modulo 1 (and independent of $\mathbf{c}$). It follows that $\bar{\mathbf{c}} \in \mathbb{T}^{m+1}$ is (statistically close to) uniform and independent of $(\bar{\mathbf{A}}, \mathbf{z}^*)$, exactly as in experiment H_5.

We want to show that $\langle \mathbf{z}^*, \mathbf{y} \rangle = \Omega(mr^2)$ with overwhelming probability over the choice of $\mathbf{z}^*$, $\mathbf{y}$. To do this, it is more convenient to switch back to choosing $\mathbf{y} \leftarrow D_r^m$ then $\mathbf{z}^* \leftarrow \mathbf{y} + D_{\mathbb{Z}^m - \mathbf{y}, r}$, which by Lemma 6.4 introduces only negligible statistical error. Now, by the Chernoff-Hoeffding bound we know that $\langle \mathbf{y}, \mathbf{y} \rangle = \|\mathbf{y}\|^2 = \Theta(mr^2)$ with overwhelming probability, because it is the sum of m independent identically distributed terms y_i^2 each having expectation $\Theta(r^2)$, and each bounded by $r^2 \cdot \omega(\log n)$ with overwhelming probability. Next, we have $\langle \mathbf{z}^*, \mathbf{y} \rangle = \langle \mathbf{y}, \mathbf{y} \rangle + \langle D_{\mathbb{Z}^m - \mathbf{y}, r}, \mathbf{y} \rangle$. For a small enough $\nu(n) = \omega(\sqrt{\log n})$, with overwhelming probability the second term is bounded from above by

$$r \cdot \|\mathbf{y}\| \cdot \nu(n) = O(\sqrt{m} \cdot r^2) \cdot \nu(n) = o(mr^2).$$

Therefore, $\langle \mathbf{z}^*, \mathbf{y} \rangle = \Omega(mr^2)$.

Finally, observe that over the random choice of $v \leftarrow D_\kappa$, the term $v\langle \mathbf{z}^*, \mathbf{y} \rangle$ is distributed as $D_{\kappa \cdot \langle \mathbf{z}^*, \mathbf{y} \rangle}$. Since $\kappa = \frac{\omega(\sqrt{\log n})}{mr^2}$ by hypothesis, the distribution is $D_{\omega(\sqrt{\log n})}$, which is statistically close to uniform modulo 1 as desired. $\square$

7 Plan-Ahead Bideniable Hybrid Encryption for Long Messages

In pratice, it is desirable to have a (non-interactive) bideniable encryption scheme with *short keys* that can encrypt an unbounded number of messages. But, since any such scheme is non-committing, a result of Nielsen [Nie02] says that this goal is impossible to achieve. Fortunately, we can circumvent Nielsen's result via a relaxation called "plan-ahead" deniability (as first described in [CDNO97]), which still seems quite useful in practical applications. In a plan-ahead-deniable scheme, the sender decides on some bounded number of "fake" messages at the time of encryption, to which the sender and receiver may later equivocate the ciphertext.

In this section, we show how to construct an efficient plan-ahead bideniable scheme for *arbitrarily* long messages, by using a bideniable scheme for n-bit messages as a key-encapsulation mechanism. For simplicity, we focus on the case of just one fake message; an extension to any bounded number is straightforward.

Plan-ahead bideniable encryption. A *plan-ahead deniable encryption scheme* PA-DEN with message-space $\mathcal{M}$ consists of essentially the same algorithms as a deniable encryption scheme, but with the following interface changes (other algorithms remain the same):

- The *plan-ahead deniable* encryption algorithm $\mathsf{PADenEnc}(pk, m_0, m_1)$, given a public key pk and "real" and "fake" messages $m_0, m_1 \in \mathcal{M}$ (respectively), outputs a ciphertext c.
- The *plan-ahead sender faking* algorithm $\mathsf{PASendFake}(pk, r_S, m_0, m_1, b)$, given a public key pk, original coins r_S, "real" and "fake" messages $m_0, m_1 \in \mathcal{M}$ (respectively), and target bit b, outputs "faked" random coins r_S^* for Enc.

- The *plan-ahead receiver faking* algorithm $\mathsf{PARecFake}(fk, c, b)$, given a faking key fk, a ciphertext c, and a target bit b, outputs "faked" random coins r_R^* for Gen.

We make the analogous correctness requirements to Section 3. We say that a deniable encryption scheme DEN is *plan ahead bideniable* under *chosen-plaintext attack* if, for any fixed messages $m_0, m_1 \in \mathcal{M}$ and any $b \in \{0,1\}$, the output distributions of the following two experiments are computationally indistinguishable:

$$
\begin{array}{r|l}
pk \leftarrow \mathsf{Gen}(1^n; r_R) & (pk, fk) \leftarrow \mathsf{DenGen}(1^n) \\
c \leftarrow \mathsf{Enc}(pk, m_b; r_S) & c \leftarrow \mathsf{PADenEnc}(pk, m_0, m_1; r_S) \\
 & r_R^* \leftarrow \mathsf{PARecFake}(fk, c, b) \\
 & r_S^* \leftarrow \mathsf{PASendFake}(pk, r_S, m_0, m_1, b) \\
\text{Return } (pk, c, r_R, r_S) & \text{Return } (pk, c, r_R^*, r_S^*)
\end{array}
$$

Symmetric encryption. A building block we use in our construction is a *one-time pseudorandom* symmetric encryption scheme. Recall that a *symmetric encryption scheme* with message-space $\mathcal{M}$ is a pair (E, D) of algorithms (we assume wlog that the symmetric key is a uniformly random $K \in \{0,1\}^n$), where $\mathsf{E}(K, m)$ outputs a ciphertext c and $\mathsf{D}(K, c)$ outputs a message m. We say that the scheme is *one-time pseudorandom* if for any $m \in \mathcal{M}$ the output distributions of the following two experiments are computationally indistinguishable:

$$
\begin{array}{r|l}
K \leftarrow \{0,1\}^n & K \leftarrow \{0,1\}^n \\
c \leftarrow \mathsf{E}(K, m) & c \leftarrow \mathsf{E}(K, m);\ c' \leftarrow \{0,1\}^{|c|} \\
\text{Return } c & \text{Return } c'
\end{array}
$$

Note that standard block cipher-based symmetric encryption schemes are one-time pseudorandom (in fact, they are stronger than that) assuming the underlying block cipher is a PRF [BDJR97], so pseudorandomness is not an additional assumption in practice. That is, using any standard block cipher mode of operation, we obtain a one-time pseudorandom symmetric encryption scheme with short keys for arbitrarily long messages.

Our scheme. Let us first give a high-level overview of our scheme. We borrow an idea of [CDNO97] used in the symmetric setting, whereby a sender chooses "real" and "fake" symmetric keys K_0, K_1. The deniable encryption algorithm $\mathsf{PADenEnc}$ encrypts $K_0 \| K_1 \| b$ with the underlying bideniable scheme, where b is a uniformly random bit. It appends to this the symmetric encryptions of m_0 under K_0, and m_1 under K_1, *in an order determined by* b. (That is, letting, c_0,c_1 denote these two symmetric ciphertexts, respectively, it appends $c_b \| c_{1-b}$.) The normal encryption algorithm, on the other hand, encrypts $K_0 \| 0^n \| b$ under the bideniable scheme and instead of encrypting a "fake" message simply chooses a uniformly random string of appropriate length. As before, the order of the symmetric "ciphertexts" is determined by b.

Formally, let BI-DEN$'$ be a bidenaible encryption scheme with message-space $\{0,1\}^{2n+1}$, and let SYM be a symmetric cipher with key length n and message space $\mathcal{M}$. Define the plan-ahead hybrid bideniable encryption scheme PA-DEN[BI-DEN$'$, SYM] with message-space $\mathcal{M}$ via the algorithms in Figure 4. To simplify its description we assume that E is deterministic. We also augment the inputs to each algorithm of PA-DEN[BI-DEN$'$, SYM] to include items that are easily computable from those given to it by the above definition.

```
$\mathsf{Gen}(1^n)$:
    $(pk, sk) \leftarrow \mathsf{Gen}'(1^n)$
    Return $(pk, sk)$

$\mathsf{Enc}(pk, m)$:
    $K_0 \leftarrow \{0,1\}^n;\ b \leftarrow \{0,1\}$
    $c_{asym} \leftarrow \mathsf{Enc}'(pk, K_0 \| 0^n \| b)$
    $c_0 \leftarrow \mathsf{E}(K_0, m)$
    $c_1 \leftarrow \{0,1\}^{|c_b|}$
    Return $c_{asym} \| c_b \| c_{1-b}$

$\mathsf{Dec}(sk, c)$:
    $c_{asym} \| c_0 \| c_1 \leftarrow c$
    $K_0 \| K_1 \| b \leftarrow \mathsf{Dec}'(sk, c_{asym})$
    $m \leftarrow \mathsf{Dec}'(K_b, c_b)$
    Return $m$

$\mathsf{DenGen}(1^n)$:
    $(pk, sk, fk) \leftarrow \mathsf{Gen}'(1^n)$
    Return $(pk, sk, fk)$

$\mathsf{PADenEnc}(pk, m_0, m_1)$:
    $K_0, K_1 \leftarrow \{0,1\}^n;\ b \leftarrow \{0,1\}$
    $c_{asym} \leftarrow \mathsf{DenEnc}'(pk, K_0 \| K_1 \| b)$
    $c_0 \leftarrow \mathsf{E}(K_0, m_0)$
    $c_1 \leftarrow \mathsf{E}(K_1, m_1)$
    Return $c_{asym} \| c_b \| c_{1-b}$

$\mathsf{PARecFake}(fk, c, K_0 \| K_1 \| b', b)$:
    $c \leftarrow c_{asym} \| c_0 \| c_1$
    $x \leftarrow K_0 \| K_1 \| b';\ y \leftarrow K_b \| 0^n \| b$
    $r_R^* \leftarrow \mathsf{RecFake}'(fk, c_{asym}, x, y)$
    Return $r_R^*$

$\mathsf{PASendFake}(pk, c, r_S, b)$:
    $c \leftarrow c_{asym} \| c_0 \| c_1$
    $K_0 \| K_1 \| b' \| r \leftarrow r_S$  // $r$ are the coins for $\mathsf{DenEnc}'$
    $x \leftarrow K_0 \| K_1 \| b';\ y \leftarrow K_b \| 0^n \| b$
    $r_S^* \leftarrow \mathsf{SendFake}'(pk, c_{asym}, r, x, y)$
    Return $K_b \| r_S^* \| c_{1-b}$
```

Figure 4: Algorithms of the plan-ahead bideniable hybrid encryption scheme PA-DEN[BI-DEN$'$, SYM].

Security. The intuition for the security of our scheme is very simple: by security of the underlying bideniable scheme, the adversary cannot tell the difference between the "asymmetric" part of the ciphertexts in the two plan-ahead bideniablility experiments. But, in the right-hand experiment, the adversary never sees K_{1-b}, so we can conclude by pseudorandomness of the symmetric scheme that c_{1-b} looks random to it as well. Formally, we prove the following theorem.

Theorem 7.1. *Suppose BI-DEN$'$ is bideniable and SYM is pseudorandom. Then PA-DEN[BI-DEN$'$, SYM] defined above is plan-ahead bideniable.*

Proof. Fix m_0, m_1, b in the plan-ahead-deniablility experiments with the scheme. We define a "hybrid" game as follows:

```
$(pk, sk) \leftarrow \mathsf{Gen}'(1^n; r_R)$
$K_0, K_1 \leftarrow \{0,1\}^n;\ b \leftarrow \{0,1\}$
$c_{asym} \leftarrow \mathsf{DenEnc}'(pk, K_b \| 0^n \| b; r_S)$
$c_0 \leftarrow \mathsf{E}(K_0, m_0)$
$c_1 \leftarrow \mathsf{E}(K_1, m_1)$
Return $(pk, c_{asym} \| c_0 \| c_1, r_R, K_b \| r_S \| c_{1-b})$
```

By bideniability of BI-DEN$'$ the output distribution of the above hybrid game is indistinguishable from that of the right-side plan-ahead-deniability experiment with the scheme. By one-time pseudorandomness of SYM, it is also indistinguishable from the left-side one, finishing the proof. □

Extensions. We can make the above scheme more efficient in practice by discarding the bit b in the input to the (deniable) encryption algorithm of the underlying bideniable scheme and instead ensuring that a symmetric ciphertext decrypted under the wrong (honestly generated) key returns $\perp$. (In effect this means b is implicitly encoded, since the receiver can try encrypting c_0 and c_1 under K_0 or K_1 to determine b.) This requires adding weak robustness to the symmetric encryption scheme in the sense of [ABN10]. Fortunately, in the symmetric setting this is easily done by applying a MAC to the ciphertext.

Also note that our construction can easily be extended to achieve a notion of k-message plan-ahead bideniability (where the sender decides on $k-1$ "fake" messages at the time of encryption, to any of which the sender and receiver may later equivocate the ciphertext). However, the length of a ciphertext increases by a factor of k.

8 Bideniable Identity-Based Encryption

Here we extend our treatment of bideniability to the setting of identity-based encryption (IBE) [Sha84, BF01]. Deniability is qualitatively different in this setting than in the classical public-key one, and in particular may offer benefits over the latter in some scenarios. Moreover, we show how both of our constructions readily extend to it.

8.1 Definition

As in the PKE case, bideniable IBE augments traditional IBE with the ability for a sender and receiver to reveal their secret data to a coercer so that it appears as if any desired message had been encrypted. For the sender, this works analogously to the public-key case. For the receiver, we envisage some assistance from the master authority, who remains uncoerced. Therefore, in our definition the receiver-faking algorithm will take as input the master secret key rather than a separate 'faking key.' We further discuss this and other differences with bideniable PKE following the definition.

For simplicity (and due to the increased complexity of the definition as compared to the public-key case), we just define bideniable IBE, without separately defining sender- and receiver-deniability. The latter definitions can be obtained straightforwardly.

Definition 8.1 (Bideniable IBE). A *bi-deniable* identity-based encryption scheme IBE-DEN with message space $\mathcal{M}$ and identity-space $\mathcal{ID}$ is made up of the following algorithms:

- The *normal* setup, key-extraction, encryption, and decryption algorithms $\mathsf{Setup}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec}$ are defined as usual for identity-based encryption. These algorithms make up the *induced normal scheme*.
- The *deniable* encryption algorithm $\mathsf{DenEnc}(id \in \mathcal{ID}, m \in \mathcal{M})$ has the same interface as a normal identity-based encryption algorithm.
- The *sender faking algorithm* $\mathsf{SendFake}(pp, id, r_S, m', m)$, given public parameters pp, identity $id \in \mathcal{ID}$, original coins r_S and message m' of DenEnc, and desired message m, outputs faked random coins r_S^* for Enc with respect to id.
- The *receiver faking algorithm* $\mathsf{RecFake}(msk, id, c, m)$, given the master secret key msk (which we assume without loss of generality contains pp), an identity $id \in \mathcal{ID}$, a ciphertext c, and a desired message $m \in \mathcal{M}$, outputs faked private key sk_{id} for Ext with respect to id.

We require the following properties:

1. *Correctness.* Any triplet $(\mathsf{Setup}, \mathsf{Ext}, \mathsf{E}, \mathsf{Dec})$, where $\mathsf{E} \in \{\mathsf{Enc}, \mathsf{DenEnc}\}$, should form a correct identity-based encryption scheme.
2. *Multi-distributional bideniability.* We require that for every adversary ("coercer") $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ the output distributions of the following two experiments are computationally indistinguishable:

$(pp, msk) \leftarrow \mathsf{Setup}(1^n)$	$(pp, msk) \leftarrow \mathsf{Setup}(1^n)$
$(id^*, m, m', st) \leftarrow \mathcal{A}_1^{\mathcal{O}_1, \mathcal{O}_2}(pp)$	$(id, m, m', st) \leftarrow \mathcal{A}_1^{\mathcal{O}_1, \mathcal{O}_2}(pp)$
$(r_S, sk_{id^*}) \leftarrow \mathcal{O}_1(id^*, m', m)$	$(r_S^*, sk_{id^*}^*) \leftarrow \mathcal{O}_2(id^*, m', m)$
Return $\mathcal{A}_2^{\mathcal{O}_1, \mathcal{O}_2}(r_S, sk_{id^*}, st)$	Return $\mathcal{A}_2^{\mathcal{O}_1, \mathcal{O}_2}(r_S^*, sk_{id^*}^*, st)$

where $\mathcal{O}_1$ and $\mathcal{O}_2$ are defined as follows:

$\mathcal{O}_1(id, m_1, m_2)$:	$\mathcal{O}_2(id, m_1, m_2)$:
$c \leftarrow \mathsf{Enc}(pp, id, m_2; r)$	$c \leftarrow \mathsf{Enc}(pp, id, m_1; r)$
$sk_{id} \leftarrow \mathsf{Ext}(msk, id)$	$r^* \leftarrow \mathsf{SendFake}(pp, id, r, m_1, m_2)$
Return (c, r, sk_{id})	$sk_{id}^* \leftarrow \mathsf{RecFake}(msk, id, c, m_2)$
	Return (c, r^*, sk_{id}^*)

In the above experiments, we require that no id is queried to *both* $\mathcal{O}_1$ and $\mathcal{O}_2$ during the entire course of the experiment; *i.e.*, when counting all the queries made by $\mathcal{A}_1$ and $\mathcal{A}_2$ together. Moreover, we require that neither $\mathcal{A}_1$ nor $\mathcal{A}_2$ query id^* to either of their oracles.

Note in the above above experiments, the adversary is given access to two oracles — one that 'honestly' generates a sender's coins and a receiver's secret key, and one that generates 'faked' ones of each. (In the first case, the adversary can alternatively generate coins for honest ciphertexts itself; we include them in this oracle's output only for symmetry.) The first oracle in particular captures the usual exposure of the sender's and receiver's coins in the IBE setting, whereas the second models "coercing" a sender and receiver. By a hybrid argument, it is easy to show that the above definition is equivalent to one where the adversary chooses many challenge identities and messages. This means that the definition permits faking by an arbitrary number of senders and receivers, for the same public parameters.

Comparison with bideniable PKE. We contrast our notion of bideniabile IBE with that of bideniable PKE defined in Section 3. As we mentioned, the main functional difference is that in bideniable PKE a receiver possesses its own faking key, whereas in the IBE setting the secret key of the master authority functions as the faking key. The interpretation is that the receiver must get assistance (perhaps preemptively) from the master authority to generate a fake secret key. We assume that the master authority cannot be coerced; e.g., because it is outside the jurisdiction of the coercer. The upshot is that in this case there is no longer a need for receiver multi-distributionality, *i.e.*, a receiver need not choose one or the other key-generation algorithm to run.

It is also significant to note that in the IBE setting an interactive encryption protocol is not permitted, since the receiver is not assumed to be registered in the system at the time of encryption. Moreover, the recent impossibility proof of Bendlin *et al.* [BNNO11] readily extends to the IBE setting. Therefore, coordination between the master authority and receiver to generate the faking key appears *necessary* in order to combine the benefits of IBE with receiver deniability. (As discussion in the introduction, receiver deniability is arguably more important in practice than sender deniability.)

8.2 Constructions

Our constructions can be readily modified to work in the IBE setting. We briefly describe how this is done for each one.

The simulatable-encryption scheme. In this case, we use parallel instances of a suitable IBE scheme rather than a public-key one. That is, our IBE scheme's public parameters will consist of $5n$ independently generated public parameters $pp_1, \ldots, pp_{5n}$ of an underlying IBE scheme and the master secret key consists of the the corresponding $msk_1, \ldots, msk_{5n}$. The key derivation, encryption, and decryption algorithms are defined correspondingly, where each identity's secret key consists of secret keys of this identity for a randomly chosen n-set of the $5n$ parallel instances of the underlying IBE scheme.

Roughly, for bideniability the main property require of the underlying IBE scheme is that one can invertibly sample a ciphertext that decrypts to a random bit under a given identity. This property was recently studied by Bellare *et al.* [BWY11] (under the name "one-sided openability") in the context of IBE secure under selective-opening attacks (i.e., sender corruptions). They gave constructions achieving it, without random oracles, based on the Decision Linear Assumption in bilinear groups and the Generalized Subgroup Decision Assumption in bilinear groups of composite order. We note that since the master secret key in our security model is never revealed to the coercer, we do not need any such condition on the public parameters of the IBE scheme here. We must however, also require that its key-derivation algorithm be deterministic. This is not really an additional requirement because it can always be achieved by the master authority maintaining state or using a pseudorandom function.

The lattice-based scheme. In this case, we can transform the scheme into a bideniable IBE one exactly in the same way as for the original GPV identity-based encryption scheme [GPV08]. This involves hashing the identities using a random oracle; thus, we obtain a bideniable IBE scheme under the same assumption in the random oracle model. However, we note that the random oracle is *only* used to make the scheme identity-based and is not in any way used for its deniability properties. Indeed, we have already shown these properties for the public-key version of the scheme without using a random oracle.

9 Acknowledgments

We thank Ran Canetti and Cynthia Dwork for suggesting this research topic, and Josh Benaloh and Ari Juels for interesting comments about deniability.

References

[ABN10] M. Abdalla, M. Bellare, and G. Neven. Robust encryption. In *TCC*, pages 480–497. 2010.

[AD97] M. Ajtai and C. Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *STOC*, pages 284–293. 1997.

[ADN$^+$10] J. Alwen, Y. Dodis, M. Naor, G. Segev, S. Walfish, and D. Wichs. Public-key encryption in the bounded-retrieval model. In *EUROCRYPT*, pages 113–134. 2010.

[ADW09] J. Alwen, Y. Dodis, and D. Wichs. Survey: Leakage resilience and the bounded retrieval model. In *ICITS*. 2009.

[Ajt96] M. Ajtai. Generating hard instances of lattice problems. *Quaderni di Matematica*, 13:1–32, 2004. Preliminary version in STOC 1996.

[Ajt99] M. Ajtai. Generating hard instances of the short basis problem. In *ICALP*, pages 1–9. 1999.

[AP09] J. Alwen and C. Peikert. Generating shorter bases for hard random lattices. In *STACS*, pages 75–86. 2009.

[BDJR97] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption. In *FOCS*, pages 394–403. 1997.

[BF01] D. Boneh and M. K. Franklin. Identity-based encryption from the weil pairing. *SIAM J. Comput.*, 32(3):586–615, 2003. Preliminary version in CRYPTO 2001.

[BHY09] M. Bellare, D. Hofheinz, and S. Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In *EUROCRYPT*, pages 1–35. 2009.

[BM99] M. Bellare and S. K. Miner. A forward-secure digital signature scheme. In *CRYPTO*, pages 431–448. 1999.

[BNNO11] R. Bendlin, J. B. Nielsen, P. S. Nordholt, and C. Orlandi. Receiver-deniable public-key encryption is impossible. Cryptology ePrint Archive, Report 2011/046, 2011. `http://eprint.iacr.org/`.

[BT94] J. Benaloh and D. Tuinstra. Uncoercible communication. Technical Report TR-MCS-94-1, Clarkson University, March 1994.

[BWY11] M. Bellare, B. Waters, and S. Yilek. Identity-based encryption secure against selective opening attack. In *TCC*, pages 235–252. 2011.

[CDNO97] R. Canetti, C. Dwork, M. Naor, and R. Ostrovsky. Deniable encryption. In *CRYPTO*, pages 90–104. 1997.

[CDSMW09] S. G. Choi, D. Dachman-Soled, T. Malkin, and H. Wee. Improved non-committing encryption with applications to adaptively secure protocols. In *ASIACRYPT*, pages 287–302. 2009.

[CFGN96] R. Canetti, U. Feige, O. Goldreich, and M. Naor. Adaptively secure multi-party computation. In *STOC*, pages 639–648. 1996.

[CG96] R. Canetti and R. Gennaro. Incoercible multiparty computation (extended abstract). In *FOCS*, pages 504–513. 1996.

[CHK03] R. Canetti, S. Halevi, and J. Katz. A forward-secure public-key encryption scheme. *J. Cryptology*, 20(3):265–294, 2007. Preliminary version in EUROCRYPT 2003.

[Chv79] V. Chvátal. The tail of the hypergeometric distribution. *Discrete Math.*, 25:285–287, 1979.

[DF11a] M. Duermuth and D. M. Freeman. Deniable encryption with negligible detection probability: An interactive construction. Cryptology ePrint Archive, Report 2011/066, 2011. `http://eprint.iacr.org/`.

[DF11b] M. Dürmuth and D. M. Freeman. Deniable encryption with negligible detection probability: An interactive construction. In *EUROCRYPT*, pages 610–626. 2011.

[DN00] I. Damgård and J. B. Nielsen. Improved non-committing encryption schemes based on a general complexity assumption. In *CRYPTO*, pages 432–450. 2000.

[DNRS99] C. Dwork, M. Naor, O. Reingold, and L. J. Stockmeyer. Magic functions. *J. ACM*, 50(6):852–921, 2003. Preliminary version in FOCS 1999.

[GKPV10] S. Goldwasser, Y. Kalai, C. Peikert, and V. Vaikuntanathan. Robustness of the learning with errors assumption. In *ICS*, pages ??–?? 2010.

[GPV08] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206. 2008.

[GWZ09] J. A. Garay, D. Wichs, and H.-S. Zhou. Somewhat non-committing encryption and efficient adaptively secure oblivious transfer. In *CRYPTO*, pages 505–523. 2009.

[MR04] D. Micciancio and O. Regev. Worst-case to average-case reductions based on Gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007. Preliminary version in FOCS 2004.

[Nie02] J. B. Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In *CRYPTO*, pages 111–126. 2002.

[Pei09] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *STOC*, pages 333–342. 2009.

[Pei10] C. Peikert. An efficient and parallel Gaussian sampler for lattices. In *CRYPTO*, pages 80–97. 2010.

[Reg05] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6):1–40, 2009. Preliminary version in STOC 2005.

[Rub] The rubberhose encryption system. Internet website; accessed 9 February 2010. `http://iq.org/~proff/marutukku.org/`.

[Sha84] A. Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, pages 47–53. 1984.

[Tru] Truecrypt: Free open-source on-the-fly encryption. Internet website; accessed 9 Feburary 2010. `http://www.truecrypt.org`.

[Wik10] Wikipedia. Deniable encryption — Wikipedia, the free encyclopedia, 2010. Internet website; accessed 9 February 2010. `http://en.wikipedia.org/wiki/Deniable_encryption`.