

Threshold Fully Deniable Interactive Encryption

Navid Abapour

Surrey Centre for Cyber Security, University of Surrey, UK
n.abapour@surrey.ac.uk

Abstract. Deniable Encryption (Canetti et al., CRYPTO 1997) enables parties to produce fake internal states making any transcript consistent with any plaintext of their choice. Fully Deniable Encryption (FDE) (Canetti et al., CRYPTO 2020) achieves the strongest two-party guarantee since both sender and receiver can independently fabricate randomness, even with mutually inconsistent claimed plaintexts, and anyone can fake the receiver’s side. However, FDE addresses only post-execution coercion and remains confined to the two-party setting, which leaves both *adaptive coercion* (where parties may be coerced at any round during execution) and *threshold deniability* (where secrets are distributed among n parties) as open problems.

In this work, we resolve the threshold gap and take a definitional step toward the adaptive one. We design *Threshold Fully Deniable Interactive Encryption* (TFDE) via FDE with a universal thresholdizer, equivocable commitments, and non-committing encryption via a four-layer equivocation pipeline so that any coalition of fewer than t_S senders and t_R receivers can produce per party fake states indistinguishable from honest executions, while satisfying all standard threshold public key encryption properties; like FDE itself, which is a feasibility result. Moreover, we formalize *Strong Off-the-Record Deniability* (SORD), a new notion that captures adaptive mid-execution coercion and consistent fake claims by both parties.

Keywords: Cryptographic Distributed Protocols, Secure Multi-party Computation, Privacy-preserving Protocols, Fully Deniable Encryption

1 Introduction

The ability to communicate secretly over public channels without previously shared secrets is one of the fundamental and promising achievements [1]. Yet traditional secure communication provides secrecy only while the parties’ internal states remain concealed [2] since if either the sender or the receiver is later coerced or bribed into disclosing its secret keys and randomness, conventional encryption offers no protection [3], and the communication log effectively binds the parties to the disclosed message, which makes any claim that another message was sent non-credible.

Deniable Encryption. Introduced by Canetti et al. [4], Deniable Encryption addresses this limitation by enabling parties to produce fake internal states that make any transcript appear consistent with any plaintext of their choice [4, 5]. Deniable encryption allows a user to decrypt the same ciphertext to different plausible plaintexts depending on which secret is used, so a coerced party can reveal a “decoy” message while

hiding the real one [5]. The need for plausible deniability arises in a wide range of deployed systems, from smartphones [6] and Signal Messenger [7] to IoT and wearable devices [8, 9], FractalCrypt [10], and VeraCrypt [11, 12]; these systems motivate the *notion* of deniability. The defining feature of deniable encryption is that an adversary cannot distinguish whether the parties revealed their true randomness and plaintext or provided fabricated alternatives [4, 5]. This guarantee holds even when the adversary observes the entire communication and later coerces the parties.

Fully Deniable Encryption. Deniable encryption developed incrementally through partial results: in a sender-deniable protocol [13] only the sender can produce fake randomness; in a receiver-deniable protocol [13, 14], only the receiver; in a bideniable protocol [14], both can fake simultaneously; in Fully Deniable Encryption (FDE) [15], both can fake their randomness after execution, even when their claimed plaintexts are mutually inconsistent, and anyone can fake the receiver’s side.

Problem 1: Stronger Deniability. The construction of Canetti et al. [15] addresses only post-execution coercion, where adversaries coerce parties after the protocol completes. The more challenging setting of *adaptive coercion*, where parties may be coerced at any point during protocol execution, was identified in [15] as an open problem.

Threshold Cryptosystems. As computation expands from distributed systems [16] to LLMs [17] and federated learning [18], the need for distributed, multi-party threshold security in privacy-preserving protocols keeps growing, and NIST has recently called for the standardization of Threshold Cryptography [19, 20]. A threshold cryptosystem distributes a secret across N parties such that any subset of at least t parties can collectively reconstruct or operate on the secret [21], while fewer than t parties gain no information about it; a Threshold public key Encryption (TPKE) scheme is one where decryption requires the cooperation of at least t out of N keyholders, with no single party holding the full secret key [22]. Real-world deployments of threshold cryptosystems range from Microsoft ElectionGuard [23, 24] and AWS CloudHSM [25] to DigiCert [26] and PySyft [27] of OpenMined.

Problem 2: Threshold Fully Deniable Protocol. Bringing deniable encryption together with the threshold setting, one notices that no Threshold Fully Deniable Interactive Encryption (TFDE) protocol exists to date. The question is:

Given a two-party FDE, can one construct a TFDE in which any coalition of fewer than t_S senders and fewer than t_R receivers can jointly encrypt, decrypt, and produce fake internal states that are indistinguishable from honest executions under any plaintext?

Figure 1 illustrates why this question does not answer itself. On the left, in two-party FDE, both secrecy and deniability hinge on the *monolithic* sender and receiver randomness s and r : the faking algorithms rewrite exactly this randomness. On the right, in a threshold system, no single party ever holds s , r , or m ; each party holds only a share, and a coercer may target any sub-threshold subset on either side. Bridging the two sides of the figure therefore requires faking at the granularity of *individual parties*, which raises several challenges: (I) the existing FDE faking [15] operates on monolithic randomness, not on individual shares; (II) a single fake randomness s' does not, by itself, yield fake shares, partial evaluations, commitment openings, and channel

randomness at the individual-party level; (III) simultaneously coerced parties across both sides must present mutually consistent fake views.

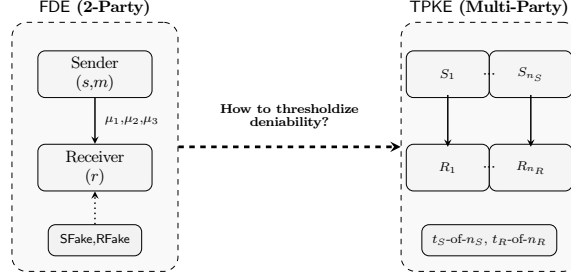


Fig. 1: The TFDE problem, where FDE is with sender randomness s , receiver randomness r , plaintext $m \in \mathcal{M}$, transcript (μ_1, μ_2, μ_3) , and faking algorithms SFake , RFake ; TPKE is over n_S sender parties $S_1, \dots, S_{n_S}$ with threshold t_S and n_R receiver parties $R_1, \dots, R_{n_R}$ with threshold t_R .

1.1 Contributions

Our contributions are in two parts, and we state which claims rest on constructions with proofs and which are definitional.

Constructive contribution: resolving Problem 2. We introduce *Threshold Fully Deniable Interactive Encryption* (TFDE, Definition 5), a multi-party protocol in which any coalition of fewer than t_S senders and t_R receivers can produce per party fake internal states, indistinguishable from honest executions under any plaintext. Our construction composes the two-party FDE of [15], a universal thresholdizer [22, 28, 29], equivocable commitments [30], and non-committing encryption [31] via a four-layer equivocation pipeline. We formalize the threshold analogues of the FDE deniability guarantees (Definitions 6–9) and prove that our construction satisfies them (Theorems 1–4), as well as the standard TPKE properties (Theorems 5–8); it also inherits from FDE resistance against accumulating attacks [32]. We emphasize that this is a *feasibility result* as the underlying FDE requires subexponentially secure indistinguishability obfuscation, but the threshold layer we add costs only polynomially in the number of parties (Section 4.1).

Definitional contribution: progress on Problem 1. We formalize *Strong Off-the-Record Deniability* (SORD, Definition 10), which strengthens off-the-record deniability [15] along two aspects: (i) it captures *adaptive coercion*, where parties may be coerced at any round during execution, and (ii) it adds a fourth indistinguishability case covering both parties deviating with *consistent* fake claims. Our TFDE, like FDE, handles post-execution coercion only, and Section 5 explains why the faking machinery of [15] does not extend to mid-execution coercion. Eventually, we leave realizing SORD, or proving it impossible, is left as an open problem.

1.2 Related Works

As Figure 2 summarizes, deniable encryption was introduced by Canetti, Dwork, Naor, and Ostrovsky [4], who established the foundational definitions and initial constructions; their scheme achieved only an inverse-polynomial distinguishing advantage, which requires superpolynomially long ciphertexts, a limitation that persisted for nearly two decades.

Sahai and Waters [13] achieved the first sender-deniable encryption with polynomial-length ciphertexts via indistinguishability obfuscation, and their scheme converts to receiver-deniable encryption via the generic transformation from [4], which yields a three-message protocol handling adversaries coercing either party separately, but not both simultaneously.

Impossibility results shaped the boundaries. Bendlin, Nielsen, Nordholt, and Orlandi [32] proved two-message receiver-deniable encryption impossible via information-theoretic compressibility arguments, establishing three-round necessity for bideniable schemes; Nielsen [33] showed non-committing encryption permits only bounded public key reuse, which implies deniable communication must be interactive; Dachman-Soled [2] established black-box limitations for sender-deniable encryption from simulatable public key encryption.

O’Neill, Peikert, and Waters [14] introduced multi-distributional bideniability under restricted settings, though without achieving standard bideniability where adversaries know the scheme and see all random coins. This remained open until Canetti, Park, and Poburinnaya [15] presented a three-message protocol achieving bideniability and off-the-record deniability (protecting parties with inconsistent claimed states); their construction requires subexponentially secure indistinguishability obfuscation and one-way functions, operates in the CRS model, addresses post-execution coercion, and explicitly identifies adaptive coercion and weaker-assumption constructions as open problems.

Extensions to richer functionalities include deniable attribute-based encryption from LWE [3], deniable functional encryption denying both plaintexts and computed functions [5], and deniable editing of encrypted data [1].

Of course, deniable security is not limited to encryption schemes: Jiang et al. [34] proposed a formal framework for deniable authenticated key exchange, which makes sure both session-key indistinguishability and offline deniability against any computationally bounded adversary; also, Canetti et al. [35] proposed a deniable secret sharing scheme, which allows shareholders to produce fake-but-plausible shares that open to any chosen secret, thus concealing participation under coercion. Nevertheless, to the best of our knowledge, no prior work achieves a threshold encryption structure with (full) deniability.

2 Preliminaries

Notation. We write λ for the security parameter, $\text{negl}(\lambda)$ for an unspecified negligible function, $[n] = \{1, \dots, n\}$, $x \leftarrow \$ X$ for uniform sampling, and $\approx_c$ for computational indistinguishability. A *probabilistic polynomial-time (PPT) algorithm* is an algorithm that may toss random coins and runs in time polynomial in its input length; *PPT adversaries* and *PPT simulators* are PPT algorithms playing the corresponding roles in a security experiment. We write $\mathcal{A}(x;\rho)$ when the random coins ρ of $\mathcal{A}$ are made explicit. Throughout, $\sigma \in \{S, R\}$ indexes the sender/receiver side, so n_σ , t_σ , A_σ , C_σ denote the number of parties, the threshold, the active set, and the coerced set of side σ .

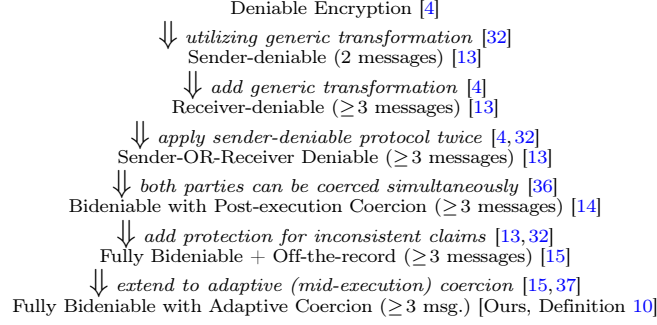


Fig. 2: The evolution of deniability notions for interactive encryption.

Game-based definitions. Our security definitions are stated as *games* between a challenger and an adversary $\mathcal{A}$: the challenger secretly samples $b \leftarrow \{0, \dots, k-1\}$ among k possible worlds (e.g., real versus fake views), prepares $\mathcal{A}$'s input accordingly, and $\mathcal{A}$ outputs a *guess* b' of b , *winning* if $b' = b$. There is no guarantee that $\mathcal{A}$ wins, a uniformly random guess already wins with probability $1/k$, and security requires that no PPT adversary does noticeably better: the *advantage* $|\Pr[b' = b] - 1/k|$ must be negligible in λ . An adversary winning with non-negligible advantage *breaks* the notion (it distinguishes real from fake internal states, so coercion becomes effective); the scheme satisfies the notion precisely when no such adversary exists.

Access structures and circuits. For a set of parties $\mathcal{P} = \{P_1, \dots, P_N\}$, a *monotone access structure* is a collection $\mathbb{A} \subseteq 2^{\mathcal{P}}$ of *authorized sets* such that $S \in \mathbb{A}$ and $S \subseteq S'$ imply $S' \in \mathbb{A}$. An access structure is *efficient* if membership can be decided by a polynomial-size boolean circuit $f_{\mathbb{A}}: \{0,1\}^N \rightarrow \{0,1\}$; we write $\mathbb{S}$ for a class of efficient access structures. Our main instantiation is the threshold structure $\mathbb{A}_t = \{S \subseteq \mathcal{P} : |S| \geq t\}$. We use boolean circuits $\mathbb{C}: \{0,1\}^k \rightarrow \{0,1\}$ of depth at most d .

Definition 1 (Equivocable Commitment). An equivocable commitment scheme over a domain $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda \in \mathbb{N}}$ is a 5-tuple $\text{EQ} = (\text{EQ.Setup}, \text{EQ.Commit}, \text{EQ.Decommit}, \text{EQ.ECom}, \text{EQ.Equiv})$ of PPT algorithms:

- $(\text{crs}, \text{td}) \leftarrow \text{EQ.Setup}(1^\lambda)$: Takes the security parameter; outputs a common reference string crs and an equivocation trapdoor td .
- $(c, d) \leftarrow \text{EQ.Commit}(\text{crs}, x)$: Commits to $x \in \mathcal{X}_\lambda$; outputs a commitment c and a decommitment d .
- $x/\perp \leftarrow \text{EQ.Decommit}(\text{crs}, c, d)$: Recovers $x \in \mathcal{X}_\lambda$ from (c, d) , or outputs the rejection symbol $\perp$.
- $(\hat{c}, \text{st}) \leftarrow \text{EQ.ECom}(\text{crs}, \text{td})$: Using the trapdoor, outputs an equivocable commitment $\hat{c}$ and a state st .
- $\hat{d} \leftarrow \text{EQ.Equiv}(\text{crs}, \text{td}, \text{st}, x)$: Using the trapdoor and state st , outputs a decommitment $\hat{d}$ that opens $\hat{c}$ to any given $x \in \mathcal{X}_\lambda$.

Following [30], EQ satisfies three properties. *Correctness*: for any $x \in \mathcal{X}_\lambda$, running $(\text{crs}, \text{td}) \leftarrow \text{EQ.Setup}(1^\lambda)$ and $(c, d) \leftarrow \text{EQ.Commit}(\text{crs}, x)$ yields $\text{EQ.Decommit}(\text{crs}, c, d) = x$ with probability 1. *Equivocability*: for any $x \in \mathcal{X}_\lambda$ chosen by a PPT adversary,

the tuple $(\text{crs}, \hat{c}, \hat{d})$ produced by running $(\hat{c}, \text{st}) \leftarrow \text{EQ.ECom}(\text{crs}, \text{td})$ and then $\hat{d} \leftarrow \text{EQ.Equiv}(\text{crs}, \text{td}, \text{st}, x)$ is computationally indistinguishable from an honest (crs, c, d) with $(c, d) \leftarrow \text{EQ.Commit}(\text{crs}, x)$, while still satisfying $\text{EQ.Decommit}(\text{crs}, \hat{c}, \hat{d}) = x$; a single global crs supports arbitrarily many equivocable commitments. *Binding*: no PPT adversary, given only crs (but not td), can produce (c, d, d', x, x') with $x \neq x'$ s.t. $\text{EQ.Decommit}(\text{crs}, c, d) = x$ and $\text{EQ.Decommit}(\text{crs}, c, d') = x'$, except with negligible probability.

Definition 2 (Non-Committing Encryption). A non-committing encryption (NCE) scheme is a tuple of PPT algorithms $\text{NCE} = (\text{Gen}, \text{Enc}, \text{Dec}, \text{Sim}_1, \text{Sim}_2)$:

- $(\text{pk}, \text{sk}) \leftarrow \text{Gen}(1^\lambda; r_G)$: Generates a key pair from explicit random coins r_G .
- $\text{ct} \leftarrow \text{Enc}(\text{pk}, M; r_E)$: Encrypts M under pk with explicit random coins r_E .
- $M/\perp \leftarrow \text{Dec}(\text{sk}, \text{ct})$: Decrypts ct ; outputs a message M or the rejection symbol $\perp$.
- $(\text{pk}, \text{ct}, \text{st}) \leftarrow \text{Sim}_1(1^\lambda)$: Outputs a simulated public key, a simulated ciphertext, and an internal state st .
- $(r_G, r_E) \leftarrow \text{Sim}_2(M, \text{st})$: Outputs key-generation and encryption coins (r_G, r_E) equivocating the simulated (pk, ct) to M .

Considering [31] and [36], an NCE satisfies two properties. *Correctness*: for all M , $\Pr[\text{Dec}(\text{sk}, \text{Enc}(\text{pk}, M; r_E)) = M] \geq 1 - \text{negl}(\lambda)$ over $(\text{pk}, \text{sk}) \leftarrow \text{Gen}(1^\lambda; r_G)$ and uniform r_G, r_E . *Simulatability*: the real distribution $\{(\text{pk}, \text{ct}, r_G, r_E) : (\text{pk}, \text{sk}) \leftarrow \text{Gen}(1^\lambda; r_G), \text{ct} \leftarrow \text{Enc}(\text{pk}, M; r_E)\}$ is computationally indistinguishable from the ideal distribution $\{(\text{pk}, \text{ct}, r_G, r_E) : (\text{pk}, \text{ct}, \text{st}) \leftarrow \text{Sim}_1(1^\lambda), (r_G, r_E) \leftarrow \text{Sim}_2(M, \text{st})\}$, where in both worlds $M \leftarrow \mathcal{A}(\text{pk})$ for any PPT adversary $\mathcal{A}$.

Definition 3 (Universal Thresholdizer). Consider an efficient access structure $\mathbb{A}$ over N parties and circuits of depth at most d . A universal thresholdizer UT consists of the algorithms:

- $(\text{pp}, s_1, \dots, s_N) \leftarrow \text{Setup}(1^\lambda, 1^d, (\mathbb{A}, N), x)$: Given a depth bound 1^d , an efficient access structure $(\mathbb{A}, N)$, and a message $x \in \{0, 1\}^k$, outputs public parameters pp and shares $s_1, \dots, s_N$.
- $y_i \leftarrow \text{Eval}(\text{pp}, s_i, \mathbb{C})$: Given a share s_i and a circuit $\mathbb{C}: \{0, 1\}^k \rightarrow \{0, 1\}$ of depth at most d , produces a partial evaluation y_i .
- $\{0, 1\} \leftarrow \text{Verify}(\text{pp}, y_i, \mathbb{C})$: Outputs a bit indicating whether the purported partial evaluation y_i is valid.
- $y \leftarrow \text{Combine}(\text{pp}, B)$: Outputs the final evaluation y from the partial evaluations $B = \{y_i\}_{i \in S}$ of an authorized set S .

According to [22], a UT satisfies the following properties. *Compactness*: the size of each partial evaluation $y_i \leftarrow \text{UT.Eval}(\text{pp}, s_i, \mathbb{C})$ is bounded by $\text{poly}(\lambda, d, N)$, independent of the secret x . *Evaluation correctness*: for any authorized set $S \in \mathbb{A}$, $\text{UT.Combine}(\text{pp}, \{\text{UT.Eval}(\text{pp}, s_i, \mathbb{C})\}_{i \in S}) = \mathbb{C}(x)$ with overwhelming probability. *Verification correctness*: for any honestly computed y_i , $\text{UT.Verify}(\text{pp}, y_i, \mathbb{C}) = 1$ with overwhelming probability. *Security*: the shares $\{s_i\}_{i \in S^*}$ and evaluation shares $\{y_i\}_{i \in S^*}$ for any unauthorized $S^* \notin \mathbb{A}$ are simulatable given only $\mathbb{C}$ and $\mathbb{C}(x)$, i.e., no PPT adversary can distinguish the real experiment from an ideal one in which shares and partial evaluations are replaced by the output of a PPT simulator $\mathcal{S}(\text{pp}, \mathbb{C}, \mathbb{C}(x), S^*)$. *Robustness*: no PPT adversary, given all shares $s_1, \dots, s_N$, can produce a fake partial evaluation $y_i^* \neq \text{UT.Eval}(\text{pp}, s_i, \mathbb{C})$ that passes UT.Verify , except with negligible probability.

Definition 4 (Fully Deniable Interactive Encryption). A 3-message fully deniable interactive encryption (FDE) scheme for message space $\mathcal{M}$ in the CRS model is a tuple of PPT algorithms $\text{FDE} = (\text{Setup}, P_1, P_2, P_3, \text{Dec}, \text{SFake}, \text{RFake})$:

- $\text{CRS} \leftarrow \text{Setup}(1^\lambda)$: Outputs a common reference string CRS containing the descriptions of the six algorithms below.
- $\mu_1 \leftarrow P_1(s, m)$: From sender randomness s and plaintext $m \in \mathcal{M}$, outputs the first protocol message μ_1 .
- $\mu_2 \leftarrow P_2(r, \mu_1)$: From receiver randomness r and μ_1 , outputs the second protocol message μ_2 .
- $\mu_3 \leftarrow P_3(s, m, \mu_1, \mu_2)$: Outputs the third protocol message μ_3 .
- $m/\perp \leftarrow \text{Dec}(r, \mu_1, \mu_2, \mu_3)$: From r and the transcript (μ_1, μ_2, μ_3) , outputs a plaintext $m \in \mathcal{M}$ or $\perp$.
- $s' \leftarrow \text{SFake}(s, m, \hat{m}, \mu_1, \mu_2, \mu_3)$: From the real (s, m) , a fake plaintext $\hat{m} \in \mathcal{M}$, and the transcript, outputs fake sender randomness s' making the transcript look consistent with $\hat{m}$.
- $r' \leftarrow \text{RFake}(\hat{m}, \mu_1, \mu_2, \mu_3; \rho)$: From a fake plaintext $\hat{m} \in \mathcal{M}$, the transcript, and fresh coins ρ , outputs fake receiver randomness r' making the transcript look consistent with $\hat{m}$; note that RFake requires neither the real receiver randomness r nor the real plaintext m (public receiver deniability).

As formalized in [15], a FDE has multiple properties, where we write $\text{tr}(s, r, m)$ for the transcript (μ_1, μ_2, μ_3) of an execution with sender randomness s , receiver randomness r , and plaintext m . *Correctness* requires that $\text{Dec}(r, \text{tr}(s, r, m)) = m$ except with negligible probability over uniform s, r . *Bideniability* requires that no PPT adversary can distinguish a real execution (s^*, r^*, tr^*) for plaintext m_0 from a faked execution (s', r', tr') for plaintext m_1 , where $\text{tr}^* = \text{tr}(s^*, r^*, m_1)$, $s' \leftarrow \text{SFake}(s^*, m_1, m_0, \text{tr}^*)$, and $r' \leftarrow \text{RFake}(m_0, \text{tr}^*; \rho_R)$; i.e., $(\text{tr}(s, r, m_0), s, r) \approx_c (\text{tr}(s, r, m_1), s', r')$. *Off-the-record deniability* strengthens this by requiring that no PPT adversary can distinguish among three cases: (i) sender tells the truth and receiver fakes $((s^*, r', \text{tr}^*)$ with $\text{tr}^* = \text{tr}(s^*, r^*, m_0)$, $r' \leftarrow \text{RFake}(m_1, \text{tr}^*; \rho_R)$); (ii) sender fakes and receiver tells the truth $((s', r^*, \text{tr}^*)$ with $\text{tr}^* = \text{tr}(s^*, r^*, m_1)$, $s' \leftarrow \text{SFake}(s^*, m_1, m_0, \text{tr}^*)$); and (iii) both fake $((s', r', \text{tr}^*)$ with $\text{tr}^* = \text{tr}(s^*, r^*, m_2)$ for any $m_2 \in \mathcal{M}$, $s' \leftarrow \text{SFake}(s^*, m_2, m_0, \text{tr}^*)$, $r' \leftarrow \text{RFake}(m_1, \text{tr}^*; \rho_R)$), ensuring protection even when parties' claimed plaintexts are mutually inconsistent. Finally, *Public Receiver Deniability* holds if RFake does not require the true receiver randomness r^* or the true plaintext as input, i.e., it has the syntax $\text{RFake}(\hat{m}, \text{tr}; \rho)$ of Definition 4, meaning any party (not just the receiver) can produce valid fake receiver coins.

3 Thresholdizing Deniable Encryption

We now present our construction, which lifts the *two-party* FDE protocol of Canetti et al. [15] (Definition 4) to a multi-party threshold setting by composing it with a universal thresholdizer, equivocable commitments, and non-committing encryption. Figure 3 gives the overall architecture; the pseudocode of the faking procedures appears in Figure 4, and that of the remaining algorithms in Figures 5–7 (Appendix A.2). Let $\text{FDE} = (\text{FDE.Setup}, P_1, P_2, P_3, \text{FDE.Dec}, \text{SFake}, \text{RFake})$ be a 3-message fully deniable interactive encryption scheme (Definition 4), let $\text{UT} = (\text{UT.Setup}, \text{UT.Eval}, \text{UT.Verify}, \text{UT.Combine})$ be a universal thresholdizer (Definition 3), let $\text{EQ} = (\text{EQ.Setup}, \text{EQ.Commit}, \text{EQ.Decommit},$

EQ.ECom, EQ.Equiv) be an equivocable commitment scheme (Definition 1), and NCE = (NCE.Gen, NCE.Enc, NCE.Dec, NCE.Sim₁, NCE.Sim₂) be a non-committing encryption scheme (Definition 2).

The intuition behind the composition is as follows. The UT replaces the monolithic sender secret $(s||m)$ and receiver secret r of FDE by shares, and lets any authorized coalition evaluate the FDE round functions $P_1, P_2, P_3, \text{FDE.Dec}$ as circuits over the shared secrets, so the transcript of our protocol is *exactly* an FDE transcript, and every FDE guarantee over the transcript is preserved. What UT alone cannot provide is deniability of the per party state that the threshold execution creates: shares, partial evaluations, exchanged commitments, and channel randomness. Hence every piece of state is wrapped in one of two equivocation mechanisms, equivocable commitments (EQ) and non-committing channels (NCE), so that under coercion *each* layer of a party's view can be re-explained toward a fake plaintext (Figure 3, bottom). Correctness is inherited round by round: by UT evaluation correctness, each combined message μ_ℓ equals exactly the corresponding FDE round-function output, so threshold decryption recovers m whenever two-party FDE decryption would (Theorem 6, Appendix A).

Definition 5 (Threshold Fully Deniable Interactive Encryption TFDE). A Threshold Fully Deniable Interactive Encryption *scheme* for message space $\mathcal{M}$, sender access structure $(\mathbb{A}_S, n_S)$ with threshold t_S , and receiver access structure $(\mathbb{A}_R, n_R)$ with threshold t_R , in the CRS model, is a tuple of PPT algorithms $\text{TFDE} = (\text{Setup}, \text{ShareGen}, \text{ThreshP}_1, \text{ThreshP}_2, \text{ThreshP}_3, \text{ThreshDec}, \text{ThreshSFake}, \text{ThreshRFake})$ whose faking procedures are given in Figure 4 and whose remaining algorithms are walked through below, with pseudocode in Figures 5–7 (Appendix A.2).

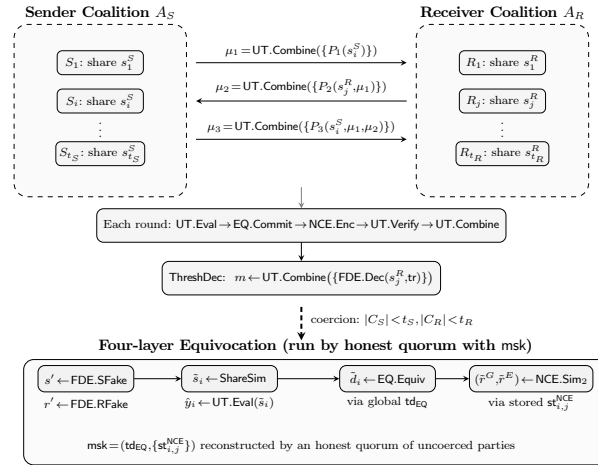


Fig. 3: Overview of our TFDE construction. The sender coalition (t_S -of- n_S) and receiver coalition (t_R -of- n_R) execute three interactive rounds, each following the pattern $\text{UT.Eval} \rightarrow \text{EQ.Commit} \rightarrow \text{NCE.Enc} \rightarrow \text{UT.Verify} \rightarrow \text{UT.Combine}$. Under coercion of $|C_S| < t_S$ senders and $|C_R| < t_R$ receivers, the honest quorum reconstructs msk and activates the four-layer equivocation pipeline (bottom): (1) FDE-level faking produces fake master randomness (s', r') ; (2) ShareSim re-explains coerced shares; (3) EQ.Equiv equivocates commitments; (4) NCE.Sim_2 equivocates channel ciphertexts.

Setup. Protocol initializes $\text{CRS}_{\text{FDE}} \leftarrow \text{FDE.Setup}(1^\lambda)$ and $(\text{crs}_{\text{EQ}}, \text{td}_{\text{EQ}}) \leftarrow \text{EQ.Setup}(1^\lambda)$, and generates an NCE key pair $(\text{pk}_{i,j}, \text{sk}_{i,j}) \leftarrow \text{NCE.Gen}(1^\lambda; r_{i,j}^G)$ for every *channel* (i,j) : intra-side pairs $(i,j \in [n_S] \text{ or } i,j \in [n_R], i \neq j)$, cross-side pairs $(i \in [n_S], j \in [n_R])$, and dealer channels $(i=0)$; sender and receiver index sets are treated as disjoint. The *public* compound CRS is $\text{CRS}_{\text{TFDE}} = (\text{CRS}_{\text{FDE}}, \text{crs}_{\text{EQ}}, \{\text{pk}_{i,j}\}_{i,j})$; each secret key $\text{sk}_{i,j}$ goes only to the receiving endpoint j . Since NCE keys tolerate only bounded reuse [33], an independent pair is provisioned per channel and round; we suppress the round index for readability. Setup also fixes the *equivocation master key* $\text{msk} = (\text{td}_{\text{EQ}}, \{\text{st}_{i,j}^{\text{NCE}}\}_{i,j})$, where $\text{st}_{i,j}^{\text{NCE}}$ is the simulation state that NCE.Sim_1 associates with channel (i,j) for later equivocation via NCE.Sim_2 ; by NCE simulatability, channels prepared in simulation mode are indistinguishable from honestly generated ones (made formal in hybrid H_4 of the proof of Theorem 1). No single party holds msk : it is secret-shared across all $n_S + n_R$ parties so that only an honest quorum (Figure 3) can reconstruct it for faking; since no coercible (sub-threshold) coalition can recover td_{EQ} , the binding of EQ is preserved against exactly the coalitions our definitions quantify over.

Share Generation. Given plaintext m , sender master randomness s , and receiver master randomness r , a *trusted dealer* forms the sender secret $x_S = (s \| m)$ and runs $(\text{pp}_S, s_1^S, \dots, s_{n_S}^S) \leftarrow \text{UT.Setup}(1^\lambda, 1^d, (\mathbb{A}_S, n_S), x_S)$, and analogously forms $x_R = r$ and runs $(\text{pp}_R, s_1^R, \dots, s_{n_R}^R) \leftarrow \text{UT.Setup}(1^\lambda, 1^d, (\mathbb{A}_R, n_R), x_R)$. Each share s_i^S (resp. s_j^R) is delivered to its holder over the corresponding dealer NCE channel, so that all transmitted material is non-committingly encrypted and hence equivocable from the outset. The trusted dealer is a standard assumption in threshold cryptography; removing it via distributed key generation is right to our contribution (cf. the silent-setup [38–41] discussion in Section 5).

Threshold Encryption Rounds. The three interactive rounds of the underlying FDE are executed in threshold structure, each following a uniform *commit-exchange-verify-combine* pattern. In Round 1, each active sender $i \in A_S$ ($|A_S| \geq t_S$) encodes P_1 as a circuit $\mathbb{C}_1$ satisfying $\mathbb{C}_1(x_S) = P_1(s, m)$, computes a partial evaluation $y_i^1 \leftarrow \text{UT.Eval}(\text{pp}_S, s_i^S, \mathbb{C}_1)$, commits via $(c_i^1, d_i^1) \leftarrow \text{EQ.Commit}(\text{crs}_{\text{EQ}}, (s_i^S, y_i^1))$, broadcasts c_i^1 , and transmits y_i^1 to every co-sender over intra-sender NCE channels. After all parties verify received partials via UT.Verify , any party combines to obtain $\mu_1 \leftarrow \text{UT.Combine}(\text{pp}_S, \{y_i^1\}_{i \in A_S})$.

Round 2 mirrors this on the receiver side: each active receiver $j \in A_R$ ($|A_R| \geq t_R$) encodes P_2 with μ_1 hardcoded as circuit $\mathbb{C}_2$, threshold-evaluates, commits, exchanges over intra-receiver NCE channels, verifies, and combines to produce $\mu_2 \leftarrow \text{UT.Combine}(\text{pp}_R, \{y_j^2\}_{j \in A_R})$; a designated combiner $j^* \in A_R$ transmits μ_2 to each sender over cross-side NCE channels. Round 3 repeats the sender-side pattern with circuit $\mathbb{C}_3$ encoding $P_3(s, m, \mu_1, \mu_2)$, yielding μ_3 , which a combiner $i^* \in A_S$ sends to the receiver coalition over cross-side NCE channels. At every step, each party's internal state is locked behind an equivocable commitment and every transmission is wrapped in NCE, which creates a double layer of equivocability.

Threshold Decryption. Upon receiving the complete transcript (μ_1, μ_2, μ_3) , the receiver coalition encodes FDE.Dec with the full transcript hardcoded as circuit $\mathbb{C}_{\text{dec}}$ and follows the same commit-exchange-verify-combine pattern: each receiver $j \in A_R$ computes $y_j^{\text{dec}} \leftarrow \text{UT.Eval}(\text{pp}_R, s_j^R, \mathbb{C}_{\text{dec}})$, commits, exchanges, verifies, and combines to recover $m/\perp \leftarrow \text{UT.Combine}(\text{pp}_R, \{y_j^{\text{dec}}\}_{j \in A_R})$.

Deniability under Coercion. When a coercer targets a subset of senders $C_S \subseteq A_S$ ($|C_S| < t_S$) or receivers $C_R \subseteq A_R$ ($|C_R| < t_R$), the faking procedures (Figure 4) activate the four coordinated layers of equivocation depicted at the bottom of Figure 3. *Layer 1*, at the FDE level, FDE.SFake (resp. FDE.RFake) produces fake master randomness s' (resp. r') consistent with a cover message $\hat{m}$. *Layer 2*, at the share level, the fake master secret $\hat{x}_S = (s' || \hat{m})$ (resp. $\hat{x}_R = r'$) is used to re-simulate each coerced party's share via ShareSim_σ , and each coerced party recomputes its partial evaluations under the fake shares; here $\text{ShareSim}_\sigma(C_\sigma, \hat{x}_\sigma)$ denotes the share simulator implied by the security property of UT (Definition 3), which samples the restriction to the unauthorized set C_σ of a fresh sharing of the target secret $\hat{x}_\sigma$; since $|C_\sigma| < t_\sigma$, the information-theoretic privacy of the underlying secret sharing [22] guarantees that the restricted shares are identically distributed regardless of the shared secret. *Layer 3*, at the commitment level, each coerced party's commitments are replaced by equivocable ones under the *same* global crs_{EQ} , produced via $\text{EQ.ECom}/\text{EQ.Equiv}$ with the trapdoor $\text{td}_{\text{EQ}} \in \text{msk}$, opening to the fake shares and fake partial evaluations. *Layer 4*, at the channel level, the randomness of each NCE ciphertext involving a coerced party is re-explained via NCE.Sim_2 to be consistent with the fake data. **ThreshRFake** requires neither the real receiver randomness r , the real plaintext m , nor the real receiver shares, which preserves public receiver deniability: any entity holding msk can produce fake receiver views given only the public transcript and the fake plaintext.

TFDE.ThreshSFake($\text{CRS}_{\text{TFDE}}, \text{msk}, \text{pp}_S, C_S, s, m, \hat{m}, \mu_1, \mu_2, \mu_3$)	TFDE.ThreshRFake($\text{CRS}_{\text{TFDE}}, \text{msk}, \text{pp}_R, C_R, \hat{m}, \mu_1, \mu_2, \mu_3; \rho$)
1: $\parallel C_S \subseteq A_S, C_S < t_S; s$ reconstructed by an authorized quorum	1: $\parallel C_R \subseteq A_R, C_R < t_R$
2: parse $\text{msk} = (\text{td}_{\text{EQ}}, \{\text{st}_{i,j}^{\text{NCE}}\})$	2: parse $\text{msk} = (\text{td}_{\text{EQ}}, \{\text{st}_{i,j}^{\text{NCE}}\})$
3: $\parallel$ Layer 1: FDE-level sender faking	3: $\parallel$ Layer 1: FDE-level receiver faking (public: no r , no m)
4: $s' \leftarrow \text{FDE.SFake}(s, m, \hat{m}, \mu_1, \mu_2, \mu_3)$	4: $r' \leftarrow \text{FDE.RFake}(\hat{m}, \mu_1, \mu_2, \mu_3; \rho)$
5: $\parallel$ Layer 2: fake shares and partial evaluations	5: $\parallel$ Layer 2: fake shares and partial evaluations
6: $\hat{x}_S \leftarrow (s' \hat{m}); \{s_i^{\hat{S}'}\}_{i \in C_S} \leftarrow \text{ShareSim}_S(C_S, \hat{x}_S)$	6: $\hat{x}_R \leftarrow r'; \{s_j^{\hat{R}'}\}_{j \in C_R} \leftarrow \text{ShareSim}_R(C_R, \hat{x}_R)$
7: for $i \in C_S, \ell \in \{1, 3\}$ do $y_i^\ell \leftarrow \text{UT.Eval}(\text{pp}_S, s_i^{\hat{S}'}, \mathbb{C}_\ell)$	7: for $j \in C_R, \ell \in \{2, \text{dec}\}$ do $y_j^\ell \leftarrow \text{UT.Eval}(\text{pp}_R, s_j^{\hat{R}'}, \mathbb{C}_\ell)$
8: $\parallel$ Layer 3: equivocate commitments (global crs_{EQ})	8: $\parallel$ Layer 3: equivocate commitments (global crs_{EQ})
9: for $i \in C_S, \ell \in \{1, 3\}$ do	9: for $j \in C_R, \ell \in \{2, \text{dec}\}$ do
10: $(c_i^\ell, \text{st}_i^\ell) \leftarrow \text{EQ.ECom}(\text{crs}_{\text{EQ}}, \text{td}_{\text{EQ}})$	10: $(c_j^\ell, \text{st}_j^\ell) \leftarrow \text{EQ.ECom}(\text{crs}_{\text{EQ}}, \text{td}_{\text{EQ}})$
11: $d_i^\ell \leftarrow \text{EQ.Equiv}(\text{crs}_{\text{EQ}}, \text{td}_{\text{EQ}}, \text{st}_i^\ell, (s_i^{\hat{S}'}, y_i^\ell))$	11: $d_j^\ell \leftarrow \text{EQ.Equiv}(\text{crs}_{\text{EQ}}, \text{td}_{\text{EQ}}, \text{st}_j^\ell, (s_j^{\hat{R}'}, y_j^\ell))$
12: $\parallel$ Layer 4: equivocate NCE channels	12: $\parallel$ Layer 4: equivocate NCE channels
13: for $i \in C_S, \ell \in \{1, 3\}$, NCE channel $(i, \cdot)$ used in round ℓ do	13: for $j \in C_R, \ell \in \{2, \text{dec}\}$, NCE channel $(j, \cdot)$ used in round ℓ do
14: $(\hat{r}_{i,\ell}^G, \hat{r}_{i,\ell}^E, \hat{r}_{i,\ell}^U, \hat{r}_{i,\ell}^V, \hat{r}_{i,\ell}^W, \hat{r}_{i,\ell}^X, \hat{r}_{i,\ell}^Y, \hat{r}_{i,\ell}^Z, \hat{r}_{i,\ell}^{\text{dec}}) \leftarrow \text{NCE.Sim}_2(y_i^\ell, \text{st}_{i,\ell}^{\text{NCE}})$	14: $(\hat{r}_{j,\ell}^G, \hat{r}_{j,\ell}^E, \hat{r}_{j,\ell}^U, \hat{r}_{j,\ell}^V, \hat{r}_{j,\ell}^W, \hat{r}_{j,\ell}^X, \hat{r}_{j,\ell}^Y, \hat{r}_{j,\ell}^Z, \hat{r}_{j,\ell}^{\text{dec}}) \leftarrow \text{NCE.Sim}_2(y_j^\ell, \text{st}_{j,\ell}^{\text{NCE}})$
15: return $\{(s_i^{\hat{S}'}, \hat{r}_{i,\ell}^G, \hat{r}_{i,\ell}^E, \hat{r}_{i,\ell}^U, \hat{r}_{i,\ell}^V, \hat{r}_{i,\ell}^W, \hat{r}_{i,\ell}^X, \hat{r}_{i,\ell}^Y, \hat{r}_{i,\ell}^Z, \hat{r}_{i,\ell}^{\text{dec}})\}_{i \in C_S}$	15: return $\{(s_j^{\hat{R}'}, \hat{r}_{j,\ell}^G, \hat{r}_{j,\ell}^E, \hat{r}_{j,\ell}^U, \hat{r}_{j,\ell}^V, \hat{r}_{j,\ell}^W, \hat{r}_{j,\ell}^X, \hat{r}_{j,\ell}^Y, \hat{r}_{j,\ell}^Z, \hat{r}_{j,\ell}^{\text{dec}})\}_{j \in C_R}$

Fig. 4: Threshold Sender Faking and Threshold Receiver Faking, each activating the four equivocation layers of Figure 3. Both take the equivocation master key msk reconstructed by an honest quorum. **ThreshSFake** additionally needs the sender master randomness s , reconstructable only by an authorized sender quorum (mirroring FDE.SFake, which needs s). **ThreshRFake** takes neither the real receiver randomness r , the real plaintext m , nor the real receiver shares $\{s_j^R\}$, which preserves public receiver deniability.

4 Security Analysis of TFDE

As a threshold cryptosystem that is fully deniable, TFDE must satisfy the properties of both TPKE (Appendix A.5) and FDE (Section 2). One gap remains: TFDE must

capture deniability at the *threshold granularity*, where the adversary sees views of individual parties, not monolithic randomness. To address this, we now define what it means for the composition to be deniable.

A coercer might target subsets of parties, see partial internal states (shares, partial evaluations, commitment openings, NCE randomness), and try to distinguish real from fake at the per party level. The FDE guarantee that “fake s is indistinguishable from real s ” does not automatically imply “the fake share of s held by party i , together with party i ’s commitment openings and NCE channel randomness, is indistinguishable from the real version.” So, we lift the properties of FDE to threshold equivalent. Below, TFDE = (Setup, ShareGen, ThreshP₁, ThreshP₂, ThreshP₃, ThreshDec, ThreshSFake, ThreshRFake) is a TFDE protocol with sender access structure $(\mathbb{A}_S, n_S)$, threshold t_S , receiver access structure $(\mathbb{A}_R, n_R)$, threshold t_R , and message space $\mathcal{M}$; $A_S \subseteq [n_S]$, $A_R \subseteq [n_R]$ are the active sender and receiver sets. For a set of parties S , we write view_S for the joint view of the parties in S , their UT shares, partial evaluations, commitment openings, NCE ciphertexts and randomness, and all protocol messages received, and $\widetilde{\text{view}}_S$ for the fake view produced by ThreshSFake or ThreshRFake.

Definition 6 (Threshold Bideniability). TFDE is threshold bideniable if no PPT adversary $\mathcal{A}$ has more than negligible advantage in the following game, for any $m_0, m_1 \in \mathcal{M}$:

1. The challenger runs $(\text{CRS}_{\text{TFDE}}, \text{msk}) \leftarrow \text{Setup}(1^\lambda)$, samples master randomness (s^*, r^*) , generates shares via ShareGen, and chooses $b \leftarrow \{0, 1\}$.
2. The challenger executes the three threshold rounds with plaintext m_b to obtain transcript $\text{tr}^* = (\mu_1, \mu_2, \mu_3)$ and all parties’ internal states.
3. The challenger sends $(\text{CRS}_{\text{TFDE}}, \text{pp}_S, \text{pp}_R, \text{tr}^*)$ to $\mathcal{A}$. $\mathcal{A}$ outputs coercion sets (C_S, C_R) with $|C_S| < t_S$, $|C_R| < t_R$.
4. – If $b = 0$: The challenger gives $\mathcal{A}$ the true views $(\text{view}_{C_S}, \text{view}_{C_R})$ from the execution with plaintext m_0 .
 – If $b = 1$: The challenger computes fake views

$$\{\widetilde{\text{view}}_i\}_{i \in C_S} \leftarrow \text{ThreshSFake}(\text{CRS}_{\text{TFDE}}, \text{msk}, \text{pp}_S, C_S, s^*, m_1, m_0, \text{tr}^*),$$

$$\{\widetilde{\text{view}}_j\}_{j \in C_R} \leftarrow \text{ThreshRFake}(\text{CRS}_{\text{TFDE}}, \text{msk}, \text{pp}_R, C_R, m_0, \text{tr}^*; \rho),$$
 and gives $\mathcal{A}$ the fake views $(\{\widetilde{\text{view}}_i\}_{i \in C_S}, \{\widetilde{\text{view}}_j\}_{j \in C_R})$.
5. $\mathcal{A}$ outputs a guess b' and wins if $b' = b$.

For all PPT $\mathcal{A}$, all $m_0, m_1 \in \mathcal{M}$: $|\Pr[b' = b] - \frac{1}{2}| \leq \text{negl}(\lambda)$.

Definition 6 subsumes several natural weaker statements: *adaptive party selection* (C_S, C_R are chosen after seeing tr^* and the public parameters); *intra-side fake view indistinguishability* (setting $C_R = \emptyset$, resp. $C_S = \emptyset$, recovers the per-side statement); and *cross-side fake consistency* (the challenger produces $\widetilde{\text{view}}_{C_S}$ and $\widetilde{\text{view}}_{C_R}$ jointly using the shared msk , so mutual consistency of simultaneously coerced parties’ fake views is a correctness requirement on the faking procedures, not a separate security property). Moreover, switching a coerced party’s threshold-layer state (share, partial evaluations, openings, channel randomness) from real to fake is an internal hybrid step in the proof of Theorem 1, not an externally observable property.

Definition 7 (Threshold Off-the-Record Deniability). TFDE is threshold off-the-record deniable if no PPT adversary $\mathcal{A}$ has more than negligible advantage in the following game, for any $m_0, m_1, m_2 \in \mathcal{M}$:

1. The challenger runs $(\text{CRS}_{\text{TFDE}}, \text{msk}) \leftarrow \text{Setup}(1^\lambda)$, samples master randomness (s^*, r^*) , generates shares, and chooses $b \leftarrow \{0, 1, 2\}$.
2. Let $m^* = m_b$ be the real plaintext determined by b . The challenger executes the protocol with plaintext m^* to obtain tr^* .
3. The challenger sends $(\text{CRS}_{\text{TFDE}}, \text{pp}_S, \text{pp}_R, \text{tr}^*)$ to $\mathcal{A}$. $\mathcal{A}$ outputs (C_S, C_R) with $|C_S| < t_S, |C_R| < t_R$.
4.
 - $b=0$ (sender truthful, receiver deviates): Give $\mathcal{A}$ true sender views view_{C_S} and fake receiver views $\{\widetilde{\text{view}}_j\}_{j \in C_R} \leftarrow \text{ThreshRFake}(\dots, m_1, \text{tr}^*; \rho)$.
 - $b=1$ (sender deviates, receiver truthful): Give $\mathcal{A}$ fake sender views $\{\widetilde{\text{view}}_i\}_{i \in C_S} \leftarrow \text{ThreshSFake}(\dots, m_1, m_0, \text{tr}^*)$ and true receiver views view_{C_R} .
 - $b=2$ (both deviate, possibly inconsistently): Give $\mathcal{A}$ the fake sender views $\{\widetilde{\text{view}}_i\}_{i \in C_S} \leftarrow \text{ThreshSFake}(\dots, m_2, m_0, \text{tr}^*)$ and the fake receiver views $\{\widetilde{\text{view}}_j\}_{j \in C_R} \leftarrow \text{ThreshRFake}(\dots, m_1, \text{tr}^*; \rho)$.
5. $\mathcal{A}$ outputs b' and wins if $b' = b$.

For all PPT $\mathcal{A}$, all $m_0, m_1, m_2 \in \mathcal{M}$: $|\Pr[b' = b] - \frac{1}{3}| \leq \text{negl}(\lambda)$.

As in Definition 6, adaptive party selection is built in, and cross-side consistency is enforced by the shared msk rather than stated as a separate property.

Definition 8 (Threshold Public Receiver Deniability). TFDE has threshold public receiver deniability if ThreshRFake has the following restricted syntax:

$$\{\widetilde{\text{view}}_j\}_{j \in C_R} \leftarrow \text{ThreshRFake}(\text{CRS}_{\text{TFDE}}, \text{msk}, \text{pp}_R, C_R, \hat{m}, \mu_1, \mu_2, \mu_3; \rho),$$

where msk is system-level secret state (equivocation trapdoor and NCE simulation states), not receiver-specific secret state. In particular, ThreshRFake takes as input neither the real receiver master randomness r , nor the real plaintext m , nor the real receiver shares $\{s_j^R\}_{j \in C_R}$. This ensures that any entity with access to msk , not just the coerced receivers, can produce fake receiver views without knowledge of any receiver-specific secret.

Definition 9 (Full Threshold Deniability). A TFDE protocol is fully threshold deniable if it satisfies threshold bideniability (Definition 6), threshold off-the-record deniability (Definition 7), and threshold public receiver deniability (Definition 8), in addition to the standard TPKE properties (Definitions 13, 14, 15, and 16).

The following theorems state that the proposed TFDE protocol is fully threshold deniable. Due to space constraint, here we give proof sketches; the complete formal proofs, together with the standard TPKE definitions and theorems (Definitions 12–16, Theorems 5–8), appear in Appendix A.

Theorem 1. The TFDE protocol (Definition 5) satisfies threshold bideniability (Definition 6) under the assumptions that FDE satisfies bideniability [15], UT supports share simulation relative to fixed public parameters, EQ satisfies equivocability, and NCE satisfies simulatability.

Proof Sketch. The proof proceeds through a hybrid argument: (i) switch the FDE-level randomness from real to fake via FDE bideniability, (ii) re-simulate coerced parties' shares to be consistent with the fake secret via UT share simulation, (iii) equivocate their commitments via EQ equivocability, and (iv) equivocate their NCE ciphertexts via NCE simulatability; each step incurs only negligible distinguishing advantage. $\square$

Theorem 2. *Our TFDE (Definition 5) satisfies threshold off-the-record deniability (Definition 7) under the same assumptions as Theorem 1, with the additional requirement that FDE satisfies off-the-record deniability [15].*

Proof Sketch. The three cases ($b \in \{0,1,2\}$) are reduced to the underlying FDE off-the-record deniability game by first invoking UT share simulation, EQ equivocability, and NCE simulatability to replace each coerced party's threshold-layer state, then appealing to the FDE-level three-way indistinguishability to conclude. $\square$

Theorem 3. *TFDE protocol (Definition 5) satisfies threshold public receiver deniability (Definition 8).*

Proof Sketch. By inspection of ThreshRFake (Figure 4), its inputs are exactly those of the restricted syntax in Definition 8: it invokes $\text{FDE.RFake}(\hat{m}, \mu_1, \mu_2, \mu_3; \rho)$, which by the public receiver deniability of FDE [15] requires neither r nor m ; and $\text{ShareSim}_R(C_R, \hat{x}_R)$ requires only the target secret and the coerced set, not the real shares. $\square$

Theorem 4. *The TFDE (Definition 5) is fully threshold deniable (Definition 9).*

Proof Sketch. This follows directly from Theorems 5–8 (standard TPKE properties, Appendix A) and Theorems 1–3 (threshold deniability properties). $\square$

4.1 Complexity and Feasibility

We quantify the overhead our threshold layer adds over the two-party FDE, as a function of the thresholds. The dominant cost is inherited from FDE itself: its CRS consists of six obfuscated programs under subexponentially secure iO [15]. Everything our threshold layer adds, lattice-based UT and NCE, group-based EQ, is polynomial and implementable with existing tools. Hence TFDE is a feasibility result whose concrete efficiency worth future progress on practical FDE.

Rounds. After a one-time dealer round (ShareGen), the protocol retains the three-message flow of FDE, each flow message being produced by one internal *commit-exchange-verify-combine* round within a coalition; threshold decryption adds one more such round. Round complexity is thus constant, four coalition rounds, independent of n_S, n_R .

Channels and keys. Setup provisions one NCE key pair per channel: $n_S n_R$ cross-side, $\binom{n_S}{2} + \binom{n_R}{2}$ intra-side, and $n_S + n_R$ dealer channels, $\Theta((n_S + n_R)^2)$ key pairs in total; bounded NCE key reuse (one pair per channel per round, at most four rounds) multiplies this by a constant.

Communication and sizes. By UT compactness (Definition 3), every share and partial evaluation has size at most $\text{poly}(\lambda, d, N)$, independent of the secret. In a sender round, each $i \in A_S$ broadcasts one commitment and sends $|A_S| - 1$ ciphertexts; receiver rounds are analogous, and the designated combiner sends $|A_S|$ (resp. $|A_R|$) cross-side ciphertexts, for total communication $O((|A_S|^2 + |A_R|^2) \cdot \text{poly}(\lambda, d, N))$ per execution. In fact, the *ciphertext* of TFDE, the transcript tr^* , has the same $\text{poly}(\lambda)$ size as the two-party FDE transcript, independent of n_S, n_R (Theorem 5).

Concrete configuration. For 3-of-5 senders and 3-of-5 receivers ($n_S = n_R = 5$, $t_S = t_R = 3$, $|A_S| = |A_R| = 3$): Setup provisions 25 cross-side, 10+10 intra-side, and 10 dealer channels (55 per round used). Per execution, the four coalition rounds involve $4 \cdot 3 = 12$ broadcast commitments and $4 \cdot 6 = 24$ intra-coalition plus $3 + 3 = 6$ cross-side NCE ciphertexts, each carrying a $\text{poly}(\lambda, d, N)$ -size payload.

5 Strong Off-the-Record Deniability

This section presents our definition toward Problem 1. Off-the-record deniability, the three-way indistinguishability among (i) sender truthful/receiver faking, (ii) sender faking/receiver truthful, and (iii) both faking with inconsistent claims, sketched in Section 2 and recalled formally as Definition 11 and Figure 8 in Appendix A.3, leaves unaddressed an important case: both parties coordinating on a *consistent* fake plaintext that differs from the true message, e.g., when both are coerced simultaneously and provide the same fabricated explanation to avoid detection of their deception [15]. Standard bideniability [14] covers this scenario but assumes both parties execute the protocol honestly and only lie afterward; off-the-record deniability considers arbitrary deviations but protects only inconsistent claims. The gap matters wherever parties can coordinate their responses under coercion or face incentives to present a unified false narrative: without a guarantee for consistent fake claims, the mere consistency of the parties' claims could itself leak information about the true communication.

Therefore, to fill this gap, we introduce Strong Off-the-record Deniability (SORD), which strengthens Definition 11 along two axes: it requires indistinguishability under *adaptive* (mid-execution) coercion, and it adds a fourth indistinguishability case where both parties deviate with consistent claims. SORD ensures that no adversary can distinguish whether the observed consistency in the parties' claims reflects truth, coordination, or independent fabrication. Combined with bideniability, SORD protects all possible coercion scenarios and response patterns.

Definition 10 (Strong Off-the-Record Deniability). Let $\pi = (\text{Setup}, P_1, P_2, P_3, \text{Dec}, \text{SFake}, \text{RFake})$ be a three-message interactive encryption scheme with message space $\mathcal{M}$. We say that π satisfies strong off-the-record deniability (SORD) if no PPT adversary $\mathcal{A}$ wins with more than negligible advantage in the following game, for any $m_0, m_1, m_2 \in \mathcal{M}$.

1. The challenger generates $\text{CRS} \leftarrow \text{Setup}(r_{\text{Setup}})$ for random r_{Setup} , and chooses $b \leftarrow \{0, 1, 2, 3\}$.
2. The challenger initializes the protocol execution with random s^*, r^* and tracks the current round $\tau \in \{1, 2, 3\}$.

3. $\mathcal{A}$ adaptively issues coercion queries (party, τ) where $\text{party} \in \{\text{sender}, \text{receiver}\}$ and $\tau \in \{1, 2, 3, 4\}$ indicates the round at which coercion occurs (with $\tau = 4$ denoting post-execution coercion). Let τ_S and τ_R denote the coercion times for the sender and receiver, respectively (adaptive coercion phase).
4. Based on b and the coercion pattern (τ_S, τ_R) , the challenger responds as follows:
 - Sender truthful, Receiver deviates ($b=0$): generate $\text{tr}^* \leftarrow \pi(s^*, r^*, m_0)$; upon sender coercion at round τ_S , give the real partial state $\text{st}_S^{\tau_S}(s^*, m_0)$; upon receiver coercion at round τ_R , give the fake partial state $\text{st}_R^{\tau_R} \leftarrow \text{RFake}_{\tau_R}(r^*, m_0, m_1, \text{tr}_{<\tau_R}; \rho_R)$, where $\text{tr}_{<\tau_R}$ denotes the partial transcript up to round τ_R .
 - Sender deviates, Receiver truthful ($b=1$): generate $\text{tr}^* \leftarrow \pi(s^*, r^*, m_1)$; upon sender coercion, give the fake state $\text{st}_S^{\tau_S} \leftarrow \text{SFake}_{\tau_S}(s^*, m_1, m_0, \text{tr}_{<\tau_S}; \rho_S)$; upon receiver coercion, give the real state $\text{st}_R^{\tau_R}(r^*, m_1)$.
 - Both deviate with inconsistent claims ($b=2$): generate $\text{tr}^* \leftarrow \pi(s^*, r^*, m_2)$; upon coercion, give the fake states $\text{st}_S^{\tau_S} \leftarrow \text{SFake}_{\tau_S}(s^*, m_2, m_0, \text{tr}_{<\tau_S}; \rho_S)$ and $\text{st}_R^{\tau_R} \leftarrow \text{RFake}_{\tau_R}(r^*, m_2, m_1, \text{tr}_{<\tau_R}; \rho_R)$.
 - Both deviate with consistent fake claims ($b=3$): generate $\text{tr}^* \leftarrow \pi(s^*, r^*, m_2)$; upon coercion, give the fake states $\text{st}_S^{\tau_S} \leftarrow \text{SFake}_{\tau_S}(s^*, m_2, m_0, \text{tr}_{<\tau_S}; \rho_S)$ and $\text{st}_R^{\tau_R} \leftarrow \text{RFake}_{\tau_R}(r^*, m_2, m_0, \text{tr}_{<\tau_R}; \rho_R)$, both claiming m_0 .
5. After coercion, the challenger continues the (not yet complete) execution using the provided states, for a mid-execution coerced party, its fake state, while maintaining consistency with already-sent messages.
6. $\mathcal{A}$ outputs $b' \in \{0, 1, 2, 3\}$ and wins if $b' = b$.

Here, st_P^τ denotes the internal state of party $P \in \{\text{sender}, \text{receiver}\}$ at round τ : all random coins used up to that point, all messages sent and received, and any intermediate results. The faking algorithms SFake_τ and RFake_τ are round-specific variants of the faking algorithms of Definition 4, producing fake states at any round $\tau \in \{1, 2, 3, 4\}$ consistent with the partial transcript so far (ρ_S, ρ_R are their fresh coins; for schemes with public receiver deniability, RFake_τ ignores r^*). The advantage of $\mathcal{A}$ is $|\Pr[\mathcal{A} \text{ wins}] - 1/4|$, and SORD requires it to be negligible for all PPT $\mathcal{A}$.

On realizing SORD. To clarify, our TFDE construction does *not* achieve SORD, and the obstruction is instructive. First, the faking algorithms of [15] are inherently *post-execution*: they consume the complete transcript, whereas a party coerced at round $\tau < 4$ must produce a fake state consistent with only $\text{tr}_{<\tau}$, and must then *continue* the execution from it (step 5 of Definition 10). Faking must thus *commute* with protocol continuation, which the obfuscated programs of [15] were not designed for: their hidden level structure supports one faking operation on a finished transcript, and interleaving faking with execution runs into the bounded-length obstacles underlying accumulating attacks [32]. Second, in the threshold setting, a coerced party's fake share must remain functional for the remaining UT evaluations, which our ShareSim-based faking does not guarantee. We are not aware of an impossibility result either; we leave realizing SORD and separating it from off-the-record deniability as open problems.

Future Directions. Designing a protocol that achieves SORD is the natural next step; a protocol satisfying both bideniability and SORD could be called *Strongly Fully Deniable Encryption*. Also, the trusted dealer of our construction could be eliminated via silent-setup threshold cryptosystems [38–41]; however, current constructions rely on witness encryption [39], which remains inefficient [40, 41], and a more efficient *silent-setup* threshold public key scheme is an open problem.

A Appendix

A.1 Computational Assumptions

One-Way Functions [42]. A function $f : \{0,1\}^* \rightarrow \{0,1\}^*$ is a *one-way function* if it is polynomial-time computable, yet for every probabilistic polynomial-time adversary $\mathcal{A}$ and all sufficiently large $n \in \mathbb{N}$, $\Pr[\mathcal{A}(1^n, f(x)) \in f^{-1}(f(x))] \leq \text{negl}(\lambda)(n)$, where the probability is taken over $x \leftarrow \{0,1\}^n$ and the internal randomness of $\mathcal{A}$.

Subexponentially One-Way Functions [43]. A function $f : \{0,1\}^* \rightarrow \{0,1\}^*$ is a *subexponentially one-way function* if it is polynomial-time computable, yet for every adversary $\mathcal{A}$ running in time 2^{n^ϵ} and some fixed $\epsilon > 0$, and all sufficiently large $n \in \mathbb{N}$, $\Pr[\mathcal{A}(1^n, f(x)) \in f^{-1}(f(x))] \leq 2^{-n^\epsilon}$, where the probability is taken over $x \leftarrow \{0,1\}^n$ and the internal randomness of $\mathcal{A}$.

Learning With Errors [44]. Let n, m, q be positive integers with q prime, and let χ be an error distribution over $\mathbb{Z}_q$. The $\text{LWE}_{n,m,q,\chi}$ *decision* problem requires distinguishing, with non-negligible advantage, the distribution $\{(\mathbf{a}_i, \mathbf{a}_i^\top \mathbf{s} + e_i)\}$ from $\{(\mathbf{a}_i, u_i)\}$, where $\mathbf{a}_i \leftarrow \mathbb{Z}_q^n$, $\mathbf{s} \leftarrow \mathbb{Z}_q^m$, $e_i \leftarrow \chi$, and $u_i \leftarrow \mathbb{Z}_q$.

Ring Short Integer Solution [45]. Let $R_q = \mathbb{Z}_q[X]/(f(X))$ for an irreducible polynomial $f \in \mathbb{Z}[X]$ of degree n . The $\text{RSIS}_{n,q,\beta}$ problem requires finding a non-zero element $z \in R_q$ satisfying $a \cdot z = 0$ in R_q and $\|z\| \leq \beta$, given $a \leftarrow \mathbb{Z}_q^n$, where $\|\cdot\|$ denotes the Euclidean norm of the coefficient vector.

A.2 Deferred Pseudocode

Figure 5 presents the pseudocode of setup and share generation, and Figures 6 and 7 that of the four threshold rounds, all walked through in Section 3; the rounds follow the same commit-exchange-verify-combine pattern.

TFDE.Setup(1^λ)	TFDE.ShareGen($\text{CRS}_{\text{TFDE}}, m, s, r$)
1: $\text{CRS}_{\text{FDE}} \leftarrow \text{FDE.Setup}(1^\lambda)$	1: // Run by the trusted dealer holding (m, s, r)
2: $(\text{crs}_{\text{EQ}}, \text{td}_{\text{EQ}}) \leftarrow \text{EQ.Setup}(1^\lambda)$	2: // Sender-side thresholdization
3: $\text{Chan} \leftarrow ([n_S] \times [n_R]) \cup \binom{[n_S]}{2} \cup \binom{[n_R]}{2} \cup (\{0\} \times ([n_S] \cup [n_R]))$	3: $x_S \leftarrow (s \ m)$; $(\text{pp}_S, s_1^S, \dots, s_{n_S}^S) \leftarrow \text{UT.Setup}(1^\lambda, 1^d, (\mathbb{A}_S, n_S), x_S)$
4: for $(i, j) \in \text{Chan}$ do	4: // Receiver-side thresholdization
5: $r_{i,j}^G \leftarrow \{0,1\}^\lambda$; $(\text{pk}_{i,j}, \text{sk}_{i,j}) \leftarrow \text{NCE.Gen}(1^\lambda, r_{i,j}^G)$	5: $x_R \leftarrow r$; $(\text{pp}_R, s_1^R, \dots, s_{n_R}^R) \leftarrow \text{UT.Setup}(1^\lambda, 1^d, (\mathbb{A}_R, n_R), x_R)$
6: give $\text{sk}_{i,j}$ to endpoint j ; $\text{st}_{i,j}^{\text{NCE}} \leftarrow \text{sim. state of channel } (i, j)$	6: // Distribute shares over dealer NCE channels
7: $\text{CRS}_{\text{TFDE}} \leftarrow (\text{CRS}_{\text{FDE}}, \text{crs}_{\text{EQ}}, \{\text{pk}_{i,j}\}_{(i,j) \in \text{Chan}})$	7: for $i \in [n_S]$ do $\text{ct}_i^S \leftarrow \text{NCE.Enc}(\text{pk}_{0,i}, s_i^S, r_{0,i}^E) \rightarrow i$
8: $\text{msk} \leftarrow (\text{td}_{\text{EQ}}, \{\text{st}_{i,j}^{\text{NCE}}\}_{(i,j) \in \text{Chan}})$; secret-share msk	8: for $j \in [n_R]$ do $\text{ct}_j^R \leftarrow \text{NCE.Enc}(\text{pk}_{0,j}, s_j^R, r_{0,j}^E) \rightarrow j$
9: return $(\text{CRS}_{\text{TFDE}}, \text{msk})$	9: // Party i (resp. j) decrypts its share with $\text{sk}_{0,i}$ (resp. $\text{sk}_{0,j}$)
	10: return $(\text{pp}_S, \{s_i^S\}_{i \in [n_S]}, \text{pp}_R, \{s_j^R\}_{j \in [n_R]})$

Fig. 5: TFDE Setup and Share Generation. The setup produces a compound CRS from FDE, EQ, and a full mesh of NCE key pairs, and secret-shares the equivocation master key $\text{msk} = (\text{td}_{\text{EQ}}, \{\text{st}_{i,j}^{\text{NCE}}\})$ used only by the faking procedures (Figure 4). ShareGen thresholdizes the sender secret $x_S = (s \| m)$ and receiver secret $x_R = r$ via UT and distributes the shares over the dealer NCE channels.

TFDE.ThreshP ₁ (CRS _{TFDE} , pp _S , {s _i ^S } _{i ∈ A_S} , m)	TFDE.ThreshP ₂ (CRS _{TFDE} , pp _R , {s _j ^R } _{j ∈ A_R} , μ ₁)
1: $\parallel A_S \subseteq [n_S], A_S \geq t_S$	1: $\parallel A_R \subseteq [n_R], A_R \geq t_R$
2: Define $\mathbb{C}_1(\cdot)$ s.t. $\mathbb{C}_1(x_S) = P_1(s, m)$ where $x_S = (s \ m)$	2: Define $\mathbb{C}_2(\cdot)$ s.t. $\mathbb{C}_2(x_R) = P_2(r, \mu_1)$ with μ_1 hardcoded
3: for $i \in A_S$ do	3: for $j \in A_R$ do
4: $y_i^1 \leftarrow \text{UT.Eval}(\text{pp}_S, s_i^S, \mathbb{C}_1)$	4: $y_j^2 \leftarrow \text{UT.Eval}(\text{pp}_R, s_j^R, \mathbb{C}_2)$
5: $(c_i^1, d_i^1) \leftarrow \text{EQ.Commit}(\text{crseq}, (s_i^S, y_i^1))$	5: $(c_j^2, d_j^2) \leftarrow \text{EQ.Commit}(\text{crseq}, (s_j^R, y_j^2))$; broadcast c_j^2
6: broadcast c_i^1	6: for $j' \in A_R \setminus \{j\}$ do NCE.Enc(pk _{j,j'} , y _j ² ; r _{j,j'} ^E , 2) → j'
7: for $i' \in A_S \setminus \{i\}$ do NCE.Enc(pk _{i,i'} , y _i ¹ ; r _{i,i'} ^E , 1) → i'	7: for $j \in A_R$ do
8: for $i \in A_S$ do	8: for $j' \in A_R \setminus \{j\}$ do
9: for $i' \in A_S \setminus \{i\}$ do	9: assert UT.Verify(pp _R , y _{j'} ² , C ₂) = 1
10: assert UT.Verify(pp _S , y _i ¹ , C ₁) = 1	10: $\mu_2 \leftarrow \text{UT.Combine}(\text{pp}_R, \{y_j^2\}_{j \in A_R})$
11: $\mu_1 \leftarrow \text{UT.Combine}(\text{pp}_S, \{y_i^1\}_{i \in A_S})$	11: $\parallel$ Transmit μ_2 to sender coalition via NCE
12: return μ_1	12: for $i \in A_S$ do NCE.Enc(pk _{j*,i} , μ ₂ ; r _{j*,i} ^E , 2) → i; return μ_2

Fig. 6: TFDE Threshold Rounds 1 and 2. Each party evaluates the FDE round function via UT.Eval, commits with EQ.Commit, exchanges partial evaluations over intra-side NCE channels, verifies via UT.Verify, and combines via UT.Combine. After Round 2, the combiner $j^* \in A_R$ transmits μ_2 to the sender coalition over cross-side NCE channels.

TFDE.ThreshP ₃ (CRS _{TFDE} , pp _S , {s _i ^S } _{i ∈ A_S} , m, μ ₁ , μ ₂)	TFDE.ThreshDec(CRS _{TFDE} , pp _R , {s _j ^R } _{j ∈ A_R} , μ ₁ , μ ₂ , μ ₃)
1: Define $\mathbb{C}_3(\cdot)$ s.t. $\mathbb{C}_3(x_S) = P_3(s, m, \mu_1, \mu_2)$ with μ_1, μ_2 hardcoded	1: Define $\mathbb{C}_{\text{dec}}(\cdot)$ s.t. $\mathbb{C}_{\text{dec}}(x_R) = \text{FDE.Dec}(r, \mu_1, \mu_2, \mu_3)$
2: for $i \in A_S$ do	2: with (μ_1, μ_2, μ_3) hardcoded
3: $y_i^3 \leftarrow \text{UT.Eval}(\text{pp}_S, s_i^S, \mathbb{C}_3)$	3: for $j \in A_R$ do
4: $(c_i^3, d_i^3) \leftarrow \text{EQ.Commit}(\text{crseq}, (s_i^S, y_i^3))$	4: $y_j^{\text{dec}} \leftarrow \text{UT.Eval}(\text{pp}_R, s_j^R, \mathbb{C}_{\text{dec}})$
5: broadcast c_i^3	5: $(c_j^{\text{dec}}, d_j^{\text{dec}}) \leftarrow \text{EQ.Commit}(\text{crseq}, (s_j^R, y_j^{\text{dec}}))$
6: for $i' \in A_S \setminus \{i\}$ do NCE.Enc(pk _{i,i'} , y _i ³ ; r _{i,i'} ^E , 3) → i'	6: broadcast c_j^{dec}
7: for $i \in A_S$ do	7: for $j' \in A_R \setminus \{j\}$ do NCE.Enc(pk _{j,j'} , y _j ^{dec} ; r _{j,j'} ^E , dec) → j'
8: for $i' \in A_S \setminus \{i\}$ do	8: for $j \in A_R$ do
9: assert UT.Verify(pp _S , y _i ³ , C ₃) = 1	9: for $j' \in A_R \setminus \{j\}$ do
10: $\mu_3 \leftarrow \text{UT.Combine}(\text{pp}_S, \{y_i^3\}_{i \in A_S})$	10: assert UT.Verify(pp _R , y _{j'} ^{dec} , C _{dec}) = 1
11: $\parallel$ Transmit μ_3 to receiver coalition via NCE	11: $m/\perp \leftarrow \text{UT.Combine}(\text{pp}_R, \{y_j^{\text{dec}}\}_{j \in A_R})$
12: for $j \in A_R$ do NCE.Enc(pk _{j*,j} , μ ₃ ; r _{j*,j} ^E , 3) → j	12: return $m/\perp$
13: return μ_3	

Fig. 7: TFDE Round 3 and Decryption. Round 3 mirrors Round 1 on the sender side with circuit $\mathbb{C}_3$ encoding P_3 ; the combiner $i^* \in A_S$ sends μ_3 to receivers. Decryption encodes FDE.Dec as $\mathbb{C}_{\text{dec}}$ and follows the same commit-exchange-verify-combine procedure.

A.3 Off-the-Record Deniability Experiments

For completeness, we recall the off-the-record deniability notion of [15] referenced in Section 5.

Definition 11 (Off-the-Record Deniability). *An interactive deniable encryption scheme Π for message space $\mathcal{M}$ satisfies off-the-record deniability if for any PPT adversary $\mathcal{A}$, the advantage*

$$\text{Adv}_{\mathcal{A}}^{\text{OTR}}(\lambda) = \max_{i \neq j \in \{0,1,2\}} \left| \Pr[\text{Expt}_{\mathcal{A},i}^{\text{OTR}}(1^\lambda) = 1] - \Pr[\text{Expt}_{\mathcal{A},j}^{\text{OTR}}(1^\lambda) = 1] \right|$$

is negligible in λ , where the experiments $\text{Expt}_{\mathcal{A},b}^{\text{OTR}}(1^\lambda)$ for $b \in \{0,1,2\}$ are defined in Figure 8.

$\text{Expt}_{A,0}^{\text{OTR}}(1^\lambda, m_0, m_1, m_2)$	$\text{Expt}_{A,1}^{\text{OTR}}(1^\lambda, m_0, m_1, m_2)$	$\text{Expt}_{A,2}^{\text{OTR}}(1^\lambda, m_0, m_1, m_2)$
1: $b \leftarrow 0$	1: $b' \leftarrow 1$	1: $b \leftarrow 2$
2: $r_{\text{setup}} \leftarrow \{0,1\}^{ r_{\text{setup}} }$	2: $r_{\text{setup}} \leftarrow \{0,1\}^{ r_{\text{setup}} }$	2: $r_{\text{setup}} \leftarrow \{0,1\}^{ r_{\text{setup}} }$
3: $\text{CRS} \leftarrow \text{Setup}(1^\lambda; r_{\text{setup}})$	3: $\text{CRS} \leftarrow \text{Setup}(1^\lambda; r_{\text{setup}})$	3: $\text{CRS} \leftarrow \text{Setup}(1^\lambda; r_{\text{setup}})$
4: $s^* \leftarrow \{0,1\}^{ s }; \quad r^* \leftarrow \{0,1\}^{ r }$	4: $s^* \leftarrow \{0,1\}^{ s }; \quad r^* \leftarrow \{0,1\}^{ r }$	4: $s^* \leftarrow \{0,1\}^{ s }; \quad r^* \leftarrow \{0,1\}^{ r }$
5: $\mu_1^* \leftarrow P_1(s^*, m_0); \quad \mu_2^* \leftarrow P_2(r^*, \mu_1^*)$	5: $\mu_1^* \leftarrow P_1(s^*, m_1); \quad \mu_2^* \leftarrow P_2(r^*, \mu_1^*)$	5: $\mu_1^* \leftarrow P_1(s^*, m_2); \quad \mu_2^* \leftarrow P_2(r^*, \mu_1^*)$
6: $\mu_3^* \leftarrow P_3(s^*, m_0, \mu_1^*, \mu_2^*)$	6: $\mu_3^* \leftarrow P_3(s^*, m_1, \mu_1^*, \mu_2^*)$	6: $\mu_3^* \leftarrow P_3(s^*, m_2, \mu_1^*, \mu_2^*)$
7: $\text{tr}^* \leftarrow (\mu_1^*, \mu_2^*, \mu_3^*)$	7: $\text{tr}^* \leftarrow (\mu_1^*, \mu_2^*, \mu_3^*)$	7: $\text{tr}^* \leftarrow (\mu_1^*, \mu_2^*, \mu_3^*)$
8: $\rho_R \leftarrow \{0,1\}^{ \rho_R }$	8: $s' \leftarrow \text{SFake}(s^*, m_1, m_0, \text{tr}^*)$	8: $\rho_R \leftarrow \{0,1\}^{ \rho_R }$
9: $r' \leftarrow \text{RFake}(m_1, \text{tr}^*; \rho_R)$	9: $b' \leftarrow \mathcal{A}(\text{CRS}, m_0, m_1, m_2, s', r', \text{tr}^*)$	9: $s' \leftarrow \text{SFake}(s^*, m_2, m_0, \text{tr}^*)$
10: $b' \leftarrow \mathcal{A}(\text{CRS}, m_0, m_1, m_2, s^*, r', \text{tr}^*)$	10: return b'	10: $r' \leftarrow \text{RFake}(m_1, \text{tr}^*; \rho_R)$
11: return b'		11: $b' \leftarrow \mathcal{A}(\text{CRS}, m_0, m_1, m_2, s', r', \text{tr}^*)$
		12: return b'

Fig. 8: Interactive Deniable Encryption’s Off-the-Record Deniability Experiments. We state the experiments with the public-receiver syntax $\text{RFake}(\hat{m}, \text{tr}; \rho)$ of Definition 4; for schemes whose receiver faking additionally requires the real receiver randomness r^* , RFake takes r^* as an extra input and the experiments are otherwise unchanged.

A.4 Instantiations and Parameters

Universal Thresholdizer. We adopt the UT from Boneh et al. [22], built upon a Linear Secret Sharing Scheme $\{0,1\}$ -LSSS [22], Fully Homomorphic Encryption FHE [46] (which together yield TFHE), a Pre-processing Zero-knowledge Proof PZK [47], and a Commitment scheme Com [48]; this line of work merges the lattice-based UT of [28] with the construction of TFHE for arbitrary access structures by Jain et al. [29]. Figure 9 formalizes how this UT operates, instantiated with $n_S, n_R \leq N$, a depth bound d sufficient for the FDE circuits $\mathbb{C}_1, \mathbb{C}_2, \mathbb{C}_3, \mathbb{C}_{\text{dec}}$, and share sizes $|s_i| \leq \text{poly}(\lambda, d, N)$, achieving λ -bit security under the $\text{LWE}_{n,m,q,\chi}$ [44] assumption.

Non-Committing Encryption We instantiate NCE from Brakerski, Branco, Döttling, Garg, and Malavolta [49], which achieves constant ciphertext-rate $O(1)$ in the plain model (no trusted setup) from the hardness of $\text{LWE}_{k,n,q,\chi}$ with superpolynomial modulus-to-noise ratio $q/B = \lambda^{\omega(1)}$ [50], instantiated with message length ℓ , packed Regev public key $(A \in \mathbb{Z}_q^{k \times n}, \{v_i\}_{i \in [\ell]})$, compressed ciphertext $(c_1 \in \mathbb{Z}_q^n, \{c_{2,i}\}_{i \in [\ell]} \in \{0,1\}^\ell, z \in \mathbb{Z}_q)$, error-correcting code of constant rate $R = O(1)$ and error rate $1/2 - \delta$ for fixed $\delta > 0$ [51], subset sizes $|I_R| = |I_S| = \ell/8$, and share size $|I_g| = 3\ell/4$, achieving λ -bit security and overall ciphertext-rate $O(1)$ as $\ell \rightarrow \infty$.

Fully Deniable Encryption We adopt the FDE of Canetti et al. [15], which operates in the CRS model and achieves full deniability for a single-bit message space $\mathcal{M} = \{0,1\}$ under subexponentially secure indistinguishability obfuscation (iO) and subexponentially secure one-way functions [13], with a CRS consisting of six obfuscated programs and a three-message interaction, where the receiver’s deniability is public (i.e., RFake does not require the true receiver randomness r^*) and a single CRS suffices for an unbounded number of encryptions.

Equivocal Commitment. For EQ, we instantiate an equivocal commitment in the style of Rotem and Segev [30] in a 256-bit prime-order group $\mathbb{G}$ (Ristretto255 [52]),

UT.Setup ($1^\lambda, 1^d, \mathbb{A}, x$)	UT.Eval (pp, s_i, C)
1: $(\text{tfhepk}, \text{tfhesk}_1, \dots, \text{tfhesk}_N) \leftarrow \text{TFHE.Setup}(1^\lambda, 1^d, \mathbb{A})$ 2: $\forall i \in [k]. \text{ct}_i \leftarrow \text{TFHE.Encrypt}(\text{tfhepk}, x_i)$ 3: $\forall i \in [N]. (\sigma_{V,i}, \sigma_{P,i}) \leftarrow \text{PZK.Pre}(1^\lambda)$ 4: $\forall i \in [N]. r_i \xleftarrow{\mathbb{R}} \{0,1\}^\lambda$ 5: $\forall i \in [N]. \text{com}_i \leftarrow \text{Com}(\text{tfhesk}_i; r_i)$ 6: $pp \leftarrow (\text{tfhepk}, \{\text{ct}_i\}_{i \in [k]}, \{\sigma_{V,i}\}_{i \in [N]}, \{\text{com}_i\}_{i \in [N]})$ 7: $\forall i \in [N]. s_i \leftarrow (\text{tfhesk}_i, \sigma_{P,i}, r_i)$ 8: return $(pp, s_1, \dots, s_N)$	1: $\hat{\text{ct}} \leftarrow \text{TFHE.Eval}(\text{tfhepk}, C, \{\text{ct}_i\}_{i=1}^k)$ 2: $p_i \leftarrow \text{TFHE.PartDec}(\text{tfhepk}, \hat{\text{ct}}, \text{tfhesk}_i)$ 3: $\Psi_i : \exists(\text{tfhesk}_i, r_i) \mid \text{com}_i \leftarrow \text{Com}(\text{tfhesk}_i; r_i)$ $\wedge p_i \leftarrow \text{TFHE.PartDec}(\text{tfhepk}, \hat{\text{ct}}, \text{tfhesk}_i)$ 4: $\pi_i \leftarrow \text{PZK.Prove}(\sigma_{P,i}, \Psi_i, (\text{tfhesk}_i, r_i))$ 5: $y_i \leftarrow (p_i, \pi_i)$ 6: return y_i
UT.Combine (pp, B)	UT.Verify (pp, y_i, C)
1: $B \leftarrow \{y_i\}_{i \in S} \mid (S \subseteq \{1, \dots, N\}) \vee (S \in \mathbb{A})$ 2: $\forall i \in S: y_i \leftarrow (p_i, \pi_i)$ 3: $y \leftarrow \text{TFHE.FinDec}(\text{tfhepk}, \{p_i\}_{i \in S})$ 4: return y	1: $\hat{\text{ct}} \leftarrow \text{TFHE.Eval}(\text{tfhepk}, C, \{\text{ct}_i\}_{i=1}^k)$ 2: $\Psi_i : \exists(\text{tfhesk}_i, r_i) \mid \text{com}_i \leftarrow \text{Com}(\text{tfhesk}_i; r_i)$ $\wedge p_i \leftarrow \text{TFHE.PartDec}(\text{tfhepk}, \hat{\text{ct}}, \text{tfhesk}_i)$ 3: $y_i \leftarrow (p_i, \pi_i)$ 4: return $\text{PZK.Verify}(\sigma_{V,i}, \Psi_i, \pi_i)$

Fig. 9: Universal Thresholdizer from TFHE = (Setup, Encrypt, Eval, PartDec, FinDec) [46], PZK = (PZK.Pre, PZK.Prove, PZK.Verify) [47], and Com = (Com) [48]

where the common-reference string consists of 3 group elements, each commitment is a single group element of 32 bytes, and the discrete-logarithm relation among the CRS elements serves as the global equivocation trapdoor td_{EQ} (as in Definition 1, td_{EQ} permits creating commitments that can later be opened to any value under the same CRS). This achieves $\lambda = 128$ bits of security under the discrete logarithm [53, 54] assumption.

A.5 Security of TFDE as a Threshold PKE

Definition 12 (Threshold PKE). Let $\mathcal{P} = \{P_1, \dots, P_N\}$ denote a set of parties and $\mathbb{S}$ denote a class of efficient access structures over $\mathcal{P}$. A threshold public key encryption scheme TPKE for message space $\mathcal{M}$ and access structure class $\mathbb{S}$ consists of the following polynomial-time algorithms:

- **Setup**($1^\lambda, \mathbb{A}$) $\rightarrow (pp, ek, sk_1, \dots, sk_N)$: Takes as input the security parameter λ and an access structure $\mathbb{A} \in \mathbb{S}$, and outputs public parameters pp , an encryption key ek , and key shares $sk_1, \dots, sk_N$.
- **Enc**(ek, m) $\rightarrow ct$: Takes as input the encryption key ek and a message $m \in \mathcal{M}$, and outputs a ciphertext ct .
- **PartDec**(pp, sk_i, ct) $\rightarrow m_i$: Takes as input the public parameters pp , a key share sk_i , and a ciphertext ct , and outputs a partial decryption share m_i .
- **PartVerify**(pp, ct, m_i) $\rightarrow \{0, 1\}$: Takes as input the public parameters pp , a ciphertext ct , and a partial decryption share m_i , and outputs accept (1) or reject (0).
- **Combine**(pp, B) $\rightarrow m'$: Takes as input the public parameters pp and a set of partial decryption shares $B = \{m_i\}_{i \in S}$, and outputs a message m' .

Definition 13 (Compactness). A threshold PKE scheme for access structure class $\mathbb{S}$ satisfies compactness if there exist polynomials $\text{poly}_1(\cdot)$ and $\text{poly}_2(\cdot)$ such that for all security parameters λ and access structures $\mathbb{A} \in \mathbb{S}$, whenever $(pp, ek, sk_1, \dots, sk_N) \leftarrow \text{Setup}(1^\lambda, \mathbb{A})$ and $ct \leftarrow \text{Enc}(ek, m)$ for any message m , it holds that $|ct| \leq \text{poly}_1(\lambda)$ and $|ek| \leq \text{poly}_2(\lambda)$.

Definition 14 (Decryption Correctness). A threshold PKE scheme for access structure class $\mathbb{S}$ satisfies decryption correctness if for all security parameters λ , access structures $\mathbb{A} \in \mathbb{S}$, authorized sets $S \in \mathbb{A}$, and messages $m \in \mathcal{M}$, the following probability is overwhelming:

$$\Pr \left[\begin{array}{l} (pp, ek, sk_1, \dots, sk_N) \leftarrow \text{Setup}(1^\lambda, \mathbb{A}); \quad ct \leftarrow \text{Enc}(ek, m); \\ \{m_i \leftarrow \text{PartDec}(pp, sk_i, ct)\}_{i \in S}; \quad m' \leftarrow \text{Combine}(pp, \{m_i\}_{i \in S}); \\ \text{return } (m' = m) \end{array} \right] \geq 1 - \text{negl}(\lambda).$$

Definition 15 (Partial Verification Correctness). A threshold PKE scheme for access structure class $\mathbb{S}$ satisfies partial verification correctness if for all security parameters λ , access structures $\mathbb{A} \in \mathbb{S}$, authorized sets $S \in \mathbb{A}$, parties $i \in S$, and messages $m \in \mathcal{M}$, the following probability is overwhelming:

$$\Pr \left[\begin{array}{l} (pp, ek, sk_1, \dots, sk_N) \leftarrow \text{Setup}(1^\lambda, \mathbb{A}); \quad ct \leftarrow \text{Enc}(ek, m); \\ m_i \leftarrow \text{PartDec}(pp, sk_i, ct); \quad b \leftarrow \text{PartVerify}(pp, ct, m_i); \\ \text{return } (b = 1) \end{array} \right] \geq 1 - \text{negl}(\lambda).$$

Definition 16 (Security). A threshold PKE scheme for access structure class $\mathbb{S}$ satisfies security (IND-CCA) if for any efficient adversary $\mathcal{A}$, the advantage

$$\text{Adv}_{\mathcal{A}}^{\text{TPKE}}(\lambda) = |\Pr[\text{Expt}_{\mathcal{A},0}(1^\lambda) = 1] - \Pr[\text{Expt}_{\mathcal{A},1}(1^\lambda) = 1]|$$

is negligible in λ , where experiment $\text{Expt}_{\mathcal{A},b}(1^\lambda)$ is defined as Figure 10.

$\text{Exp}_{\mathcal{A},0}^{\text{TPKE}}(1^\lambda)$	$\text{Exp}_{\mathcal{A},1}^{\text{TPKE}}(1^\lambda)$
1: $b \leftarrow 0$	1: $b \leftarrow 1$
2: $\mathbb{A} \leftarrow \mathcal{A}(1^\lambda)$	2: $\mathbb{A} \leftarrow \mathcal{A}(1^\lambda)$
3: $(pp, ek, sk_1, \dots, sk_N) \leftarrow \text{Setup}(1^\lambda, \mathbb{A})$	3: $(pp, ek, sk_1, \dots, sk_N) \leftarrow \text{Setup}(1^\lambda, \mathbb{A})$
4: $S^* \leftarrow \mathcal{A}(pp, ek) \quad // \text{ where } S^* \notin \mathbb{A}, \text{ maximal invalid}$	4: $S^* \leftarrow \mathcal{A}(pp, ek)$
5: $b' \leftarrow \mathcal{A}^{\text{OPartDec}}(\{sk_i\}_{i \in S^*})$	5: $b' \leftarrow \mathcal{A}^{\text{OPartDec}}(\{sk_i\}_{i \in S^*})$
6: $(m_0^*, m_1^*) \leftarrow \mathcal{A}^{\text{OPartDec}}()$	6: $(m_0^*, m_1^*) \leftarrow \mathcal{A}^{\text{OPartDec}}()$
7: $ct_b^* \leftarrow \text{Enc}(ek, m_b^*)$	7: $ct_b^* \leftarrow \text{Enc}(ek, m_b^*)$
8: $b' \leftarrow \mathcal{A}^{\text{OPartDec}}(ct_b^*)$	8: $b' \leftarrow \mathcal{A}^{\text{OPartDec}}(ct_b^*)$
9: return b'	9: return b'
$\text{OPartDec}(ct, i) \quad // \quad i \in [N] \setminus S^*$ if $ct = ct_b^*$ then return $\perp$ $m_i \leftarrow \text{PartDec}(pp, sk_i, ct)$ return m_i	$\text{OPartDec}(ct, i) \quad // \quad i \in [N] \setminus S^*$ if $ct = ct_b^*$ then return $\perp$ $m_i \leftarrow \text{PartDec}(pp, sk_i, ct)$ return m_i

Fig. 10: IND-CCA Security Experiments for Threshold PKE

Theorem 5. TFDE protocol (Definition 5) satisfies compactness (Definition 13) if the underlying FDE has polynomial-size messages and UT has compact partial evaluations.

The ciphertext of TFDE is the transcript $\text{tr}^* = (\mu_1, \mu_2, \mu_3)$, which is produced by UT.Combine and is identical in size to the FDE transcript regardless of n_S, n_R , since UT.Combine outputs the evaluation $C(x)$ whose size depends only on the circuit output, not the number of parties.

Theorem 6. *TFDE (Definition 5) satisfies decryption correctness (Definition 14) if FDE satisfies correctness (Definition 4) and UT satisfies evaluation correctness (Definition 3).*

When all parties are honest and $|A_S| \geq t_S$, $|A_R| \geq t_R$, each UT.Combine invocation faithfully recovers $P_1(s, m)$, $P_2(r, \mu_1)$, $P_3(s, m, \mu_1, \mu_2)$, and $\text{FDE.Dec}(r, \mu_1, \mu_2, \mu_3)$ by evaluation correctness of UT, and the latter outputs m by correctness of FDE.

Theorem 7. *The TFDE protocol (Definition 5) satisfies partial verification correctness (Definition 15) if UT satisfies verification correctness (Definition 3).*

Every honestly computed partial evaluation $y_i = \text{UT.Eval}(\text{pp}, s_i, \mathbb{C})$ passes the value of $\text{UT.Verify}(\text{pp}, y_i, \mathbb{C}) = 1$ with overwhelming probability by verification correctness of UT.

Theorem 8. *The TFDE protocol (Definition 5) satisfies IND-CPA security against any PPT adversary controlling up to $t_S - 1$ sender parties and up to $t_R - 1$ receiver parties, if FDE is semantically secure and UT satisfies security (Definition 3).*

The adversary's view consists of $\text{pp}_S, \text{pp}_R, \text{tr}^*$, and at most $t_\sigma - 1$ shares per side; the shares reveal nothing about $x_S = (s \| m)$ or $x_R = r$ by security of UT, and the transcript tr^* hides m by semantic security of FDE (which follows from bideniability [15]).

A.6 Proof of Theorem 1

Proof. We show that no PPT adversary $\mathcal{A}$ can distinguish the real world ($b=0$) from the fake world ($b=1$) in Definition 6 with more than negligible advantage, where H_0 is the real world and H_4 is the fake world.

H_0 . The challenger runs an honest execution of TFDE with plaintext m_0 : it samples master randomness (s^*, r^*) , runs ShareGen to produce shares $\{s_i^S\}_{i \in [n_S]}, \{s_j^R\}_{j \in [n_R]}$, executes the three threshold rounds to produce transcript $\text{tr}^* = (\mu_1, \mu_2, \mu_3)$, and upon receiving the coercion sets (C_S, C_R) from $\mathcal{A}$ (with $|C_S| < t_S$, $|C_R| < t_R$), returns the *true* views $\text{view}_{C_S} \cup \text{view}_{C_R}$, each consisting of the party's share s_i^S , all partial evaluations $\{y_i^\ell\}_\ell$, commitment openings $\{(c_i^\ell, d_i^\ell)\}_\ell$, and NCE randomness $\{r_{i,\ell}^G, r_{i,\ell}^E\}$.

H_1 . Identical to H_0 except: the challenger runs the honest execution with plaintext m_1 (instead of m_0) to obtain tr^* , then computes $s' \leftarrow \text{FDE.SFake}(s^*, m_1, m_0, \text{tr}^*)$ and $r' \leftarrow \text{FDE.RFake}(\hat{m} = m_0, \text{tr}^*; \rho)$. However, it still uses the *original* shares and all threshold-layer state; only the master-level randomness that would be revealed is replaced by (s', r') .

We claim $|\Pr[\mathcal{A}(H_0) = 1] - \Pr[\mathcal{A}(H_1) = 1]| \leq \text{negl}(\lambda)$. Suppose $\mathcal{A}$ distinguishes H_0 from H_1 with non-negligible advantage ϵ . We construct an adversary $\mathcal{B}$ against the bideniability of FDE (Definition 4). $\mathcal{B}$ receives a challenge tuple $(s_\beta, r_\beta, \text{tr}^*)$ from the FDE bideniability challenger, where $\beta=0$ corresponds to real randomness for m_0 and $\beta=1$ corresponds to fake randomness consistent with m_0 while the execution used m_1 . $\mathcal{B}$ uses (s_β, r_β) as the master randomness, runs UT.Setup with $x_S = (s_\beta \| m_0)$ and $x_R = r_\beta$ to generate shares,

executes the threshold-layer machinery, and presents the resulting views to $\mathcal{A}$. If $\mathcal{A}$ outputs b' , $\mathcal{B}$ outputs b' . The advantage of $\mathcal{B}$ equals ϵ , contradicting FDE bideniability.

Note that the transcript $\text{tr}^* = (\mu_1, \mu_2, \mu_3)$ is identical in both hybrids since $\mu_\ell = \text{UT.Combine}(\text{pp}, \{y_i^\ell\}_{i \in A})$ recovers $P_\ell(\cdot)$ by UT evaluation correctness, and FDE.SFake/FDE.RFake produce randomness consistent with the *same* transcript.

H₂. Identical to **H₁** except: for each coerced sender $i \in C_S$, instead of providing the real share s_i^S derived from $x_S = (s^* \| m_1)$, the challenger computes $\hat{x}_S = (s' \| m_0)$ and samples $\tilde{s}_i^S \leftarrow \text{ShareSim}_S(C_S, \hat{x}_S)$, providing $\tilde{s}_i^S$ in place of s_i^S . Analogously, for each coerced receiver $j \in C_R$, it computes $\hat{x}_R = r'$ and samples $\tilde{s}_j^R \leftarrow \text{ShareSim}_R(C_R, \hat{x}_R)$.

We claim $|\Pr[\mathcal{A}(\text{H}_1) = 1] - \Pr[\mathcal{A}(\text{H}_2) = 1]| \leq \text{negl}(\lambda)$. This follows from the security property of UT (Definition 3). Since $|C_S| < t_S$ and $|C_R| < t_R$, the sets C_S and C_R are unauthorized. By the UT security guarantee, the joint distribution of shares $\{s_i^\sigma\}_{i \in C_\sigma}$ for any unauthorized set is simulatable: there exists a PPT simulator $\mathcal{S}$ such that for any unauthorized $S^* \notin \mathbb{A}$,

$$\{s_i\}_{i \in S^*} \approx_c \mathcal{S}(\text{pp}, S^*, \hat{x})$$

for any target secret $\hat{x}$. Concretely, the $\{0,1\}$ -LSSS underlying the thresholdizer guarantees that any $t-1$ shares are information-theoretically independent of the shared secret, so re-sampling shares consistent with $\hat{x}_S$ (resp. $\hat{x}_R$) for a sub-threshold coalition produces an identically distributed restricted view. Any distinguisher between **H₁** and **H₂** directly yields a distinguisher against UT security.

H₃. Identical to **H₂** except: for each coerced party $i \in C_S$ (resp. $j \in C_R$) and each round ℓ , instead of providing the real commitment opening (c_i^ℓ, d_i^ℓ) generated via $\text{EQ.Commit}(\text{crs}_{\text{EQ}}, (s_i^S, y_i^\ell))$, the challenger uses the equivocation mechanism under the *same* global crs_{EQ} : it runs $(\tilde{c}_i^\ell, \text{st}_i^\ell) \leftarrow \text{EQ.ECom}(\text{crs}_{\text{EQ}}, \text{td}_{\text{EQ}})$ and then $\tilde{d}_i^\ell \leftarrow \text{EQ.Equiv}(\text{crs}_{\text{EQ}}, \text{td}_{\text{EQ}}, \text{st}_i^\ell, (\tilde{s}_i^S, \hat{y}_i^\ell))$, where $\hat{y}_i^\ell = \text{UT.Eval}(\text{pp}_S, \tilde{s}_i^S, \mathbb{C}_\ell)$ is the partial evaluation under the fake share.

We claim $|\Pr[\mathcal{A}(\text{H}_2) = 1] - \Pr[\mathcal{A}(\text{H}_3) = 1]| \leq |C_S \cup C_R| \cdot L \cdot \text{negl}(\lambda)$, where $L \in \{2, 3\}$ is the number of rounds in which each party participates. For each coerced party and each round, we invoke the equivocability property of EQ (Definition 1). The equivocability guarantee states that the tuple $(\text{crs}, \hat{c}, \hat{d})$ produced by EQ.ECom and EQ.Equiv is computationally indistinguishable from an honest (crs, c, d) committing to the same value under the same crs . We apply a standard hybrid argument over the at most $(|C_S| + |C_R|) \cdot L$ commitment instances: in sub-hybrid **H_{3,k}**, the first k commitments (in some fixed ordering) use equivocated openings and the remaining use real openings. Each transition incurs negligible distinguishing advantage by EQ equivocability. By a union bound over polynomially many transitions, the total advantage is negligible.

H₄. Identical to **H₃** except: for every NCE channel $(\text{pk}_{i,j}, \text{sk}_{i,j})$ involving a coerced party (either $i \in C_S \cup C_R$ or $j \in C_S \cup C_R$), instead of providing the real randomness $(r_{i,j}^G, r_{i,j}^E)$, the challenger uses NCE simulatability: it switches channel (i, j) to simulation mode, retrieving the simulation state $\text{st}_{i,j}^{\text{NCE}}$ stored in msk at setup (where $(\text{pk}_{i,j}, \text{ct}_{i,j}, \text{st}_{i,j}^{\text{NCE}}) \leftarrow \text{NCE.Sim}_1(1^\lambda)$), and computes $(\tilde{r}_{i,j}^G, \tilde{r}_{i,j}^E) \leftarrow \text{NCE.Sim}_2(\hat{y}_i^\ell, \text{st}_{i,j}^{\text{NCE}})$.

We claim $|\Pr[\mathcal{A}(\text{H}_3) = 1] - \Pr[\mathcal{A}(\text{H}_4) = 1]| \leq \text{negl}(\lambda)$. We invoke the simulatability property of NCE (Definition 2). The simulatability guarantee states that for any PPT adversary, the real distribution $\{(\text{pk}, \text{ct}, r_G, r_E)\}$ is computationally indistinguishable from the simulated distribution $\{(\text{pk}, \text{ct}, \tilde{r}_G, \tilde{r}_E)\}$. We apply a hybrid argument over all NCE channels involving coerced parties. There are at most $(|C_S| + |C_R|) \cdot (n_S + n_R) \cdot L$

such channels (polynomial in λ). Each transition incurs negligible advantage by NCE simulatability; the total advantage is negligible by a union bound.

H_4 is exactly the fake world of Definition 6: the views presented to $\mathcal{A}$ consist of fake shares $\tilde{s}_i^\sigma$ consistent with the cover message m_0 , fake partial evaluations $\hat{y}_i^\ell$ recomputed under $\tilde{s}_i^\sigma$, equivocated commitment openings $\tilde{d}_i^\ell$, and equivocated NCE randomness, exactly the output of **ThreshSFake** and **ThreshRFake**.

By collecting the hybrid transitions we will have:

$$\begin{aligned} & |\Pr[\mathcal{A}(H_0)=1] - \Pr[\mathcal{A}(H_4)=1]| \\ & \leq \sum_{k=0}^3 |\Pr[\mathcal{A}(H_k)=1] - \Pr[\mathcal{A}(H_{k+1})=1]| \\ & \leq \text{Adv}_{\text{FDE}}^{\text{biD}}(\lambda) + \text{Adv}_{\text{UT}}^{\text{sec}}(\lambda) + (|C_S| + |C_R|) \cdot L \cdot \text{Adv}_{\text{EQ}}^{\text{equiv}}(\lambda) \\ & \quad + (|C_S| + |C_R|)(n_S + n_R)L \cdot \text{Adv}_{\text{NCE}}^{\text{sim}}(\lambda) \\ & \leq \text{negl}(\lambda), \end{aligned}$$

since each term is negligible by assumption. $\square$

A.7 Proof of Theorem 2 (Threshold Off-the-Record Deniability)

Proof. We must show that no PPT adversary $\mathcal{A}$ can distinguish among the three cases $b \in \{0, 1, 2\}$ of Definition 7 with advantage exceeding $\text{negl}(\lambda)$. It suffices to show pairwise indistinguishability: $H^{(0)} \approx_c H^{(1)} \approx_c H^{(2)}$. We show $H^{(0)} \approx_c H^{(1)}$; the argument for $H^{(1)} \approx_c H^{(2)}$ is symmetric, and $H^{(0)} \approx_c H^{(2)}$ follows by transitivity. In case $b=0$: the real plaintext is m_0 ; the sender's views are truthful; the receiver's views are faked toward m_1 . In case $b=1$: the real plaintext is m_1 ; the sender's views are faked toward m_0 ; the receiver's views are truthful.

$G_0 = H^{(0)}$. Execute with m_0 . Give $\mathcal{A}$ true sender views for C_S and fake receiver views for C_R (via **ThreshRFake**(..., m_1 , tr^* ; ρ)).

G_1 (*Equivocate receiver-side threshold layer*). For each coerced receiver $j \in C_R$: replace real share s_j^R with $\tilde{s}_j^R \leftarrow \text{ShareSim}_R(C_R, r')$ where $r' \leftarrow \text{FDE.RFake}(m_1, \text{tr}^*; \rho)$; recompute partial evaluations; equivocate commitments via **EQ.ECom**/**EQ.Equiv**; equivocate NCE channels via **NCE.Sim**₂. By the same argument as in Theorem 1 (Hybrids H_2 – H_4 applied only to the receiver side), $G_0 \approx_c G_1$ with negligible distinguishing advantage bounded by $\text{Adv}_{\text{UT}}^{\text{sec}}(\lambda) + |C_R| \cdot L \cdot \text{Adv}_{\text{EQ}}^{\text{equiv}}(\lambda) + |C_R| \cdot (n_S + n_R) \cdot L \cdot \text{Adv}_{\text{NCE}}^{\text{sim}}(\lambda)$.

Hybrid G_2 (FDE-level OTR switch). Switch the underlying execution plaintext from m_0 to m_1 , and simultaneously switch sender views from truthful to faked (toward m_0) and receiver views from faked to truthful. Concretely: the transcript is now $\text{tr}^* = \text{tr}(s^*, r^*, m_1)$; the sender views for C_S are generated via **FDE.SFake**($s^*, m_1, m_0, \text{tr}^*$); the receiver views for C_R use real (r^*, m_1).

We claim $G_1 \approx_c G_2$. At this point, G_1 corresponds to case (i) of the FDE OTR game (sender truth, receiver fakes, plaintext m_0) and G_2 corresponds to case (ii) (sender fakes, receiver truth, plaintext m_1). We construct $\mathcal{B}$ against FDE OTR as follows: $\mathcal{B}$ receives the challenge $(s_\beta, r_\beta, \text{tr}^*)$ from the FDE OTR challenger (where $\beta \in \{0, 1\}$ selects between cases (i) and (ii)), wraps the threshold layer around (s_β, r_β) by running **UT.Setup** with the appropriate secret, and presents the resulting views to $\mathcal{A}$. The threshold-layer equivocation for the coerced side (whichever is faking) is performed as in G_1 or G_3 below. Any distinguishing advantage of $\mathcal{A}$ translates directly to $\mathcal{B}$'s advantage against FDE OTR.

Hybrid G_3 (Equivocate sender-side threshold layer). For each coerced sender $i \in C_S$: replace real share s_i^S with $\tilde{s}_i^S \leftarrow \text{ShareSim}_S(C_S, \hat{x}_S)$ where $\hat{x}_S = (s' || m_0)$ and $s' \leftarrow \text{FDE.SFake}(s^*, m_1, m_0, \text{tr}^*)$; recompute partial evaluations; equivocate commitments and NCE channels. This is $H^{(1)}$.

$G_2 \approx_c G_3$ by the identical argument as $G_0 \approx_c G_1$, applied to the sender side. Case $b = 2$ uses real plaintext m_2 , with both sender and receiver views faked (sender toward m_0 , receiver toward m_1 , possibly inconsistent). The argument is analogous: first equivocate both sides' threshold layers (shares, commitments, NCE), then invoke the FDE OTR three-way indistinguishability (cases (ii) $\approx_c$ (iii)) to switch from $(m_1, \text{sender fakes}, \text{receiver truth})$ to $(m_2, \text{both fake})$.

$$\begin{aligned} & \left| \Pr[\mathcal{A} \text{ wins in } H^{(b)}] - \frac{1}{3} \right| \\ & \leq \text{Adv}_{\text{FDE}}^{\text{OTR}}(\lambda) + 2 \cdot \text{Adv}_{\text{UT}}^{\text{sec}}(\lambda) + 2(|C_S| + |C_R|) \cdot L \cdot \text{Adv}_{\text{EQ}}^{\text{equiv}}(\lambda) \\ & \quad + 2(|C_S| + |C_R|)(n_S + n_R)L \cdot \text{Adv}_{\text{NCE}}^{\text{sim}}(\lambda) \leq \text{negl}(\lambda). \end{aligned}$$

The factor of 2 accounts for equivocating both the sender and receiver threshold layers across the pairwise transitions. $\square$

A.8 Proof of Theorem 3

Proof. We must verify that **ThreshRFake** (Figure 4) satisfies the restricted syntax of Definition 8: its inputs are $(\text{CRS}_{\text{TFDE}}, \text{msk}, \text{pp}_R, C_R, \hat{m}, \mu_1, \mu_2, \mu_3; \rho)$ and it does *not* require the real receiver master randomness r , the real plaintext m , or the real receiver shares $\{s_j^R\}_{j \in C_R}$.

We verify each step of **ThreshRFake**:

FDE-level receiver faking. $r' \leftarrow \text{FDE.RFake}(\hat{m}, \mu_1, \mu_2, \mu_3; \rho)$. By the public receiver deniability of FDE [15], **FDE.RFake** has syntax **RFake** $(\hat{m}, \text{tr}; \rho)$ and does *not* take the real receiver randomness r^* or the real plaintext m as input. This is an explicit design property of the FDE construction in [15]: the receiver's faking algorithm uses only the public transcript and the target fake plaintext together with fresh random coins ρ .

Fake share construction. $\hat{x}_R \leftarrow r'$; $\tilde{s}_j^R \leftarrow \text{ShareSim}_R(C_R, \hat{x}_R)$ for $j \in C_R$. The share simulator **ShareSim_R** takes as input only the coerced set C_R and the target secret $\hat{x}_R$. It does *not* require the real shares $\{s_j^R\}$. This follows from the information-theoretic property of the $\{0,1\}$ -LSSS underlying **UT**: for any unauthorized set C_R with $|C_R| < t_R$, one can sample shares consistent with *any* target secret without knowledge of the original shares, because the conditional distribution $\{s_j^R\}_{j \in C_R} | x_R$ depends only on x_R (not on the shares of parties outside C_R). Formally, the simulator samples from the marginal distribution of the sharing of $\hat{x}_R$ restricted to coordinates in C_R .

Fake partial evaluations. $\hat{y}_j^\ell \leftarrow \text{UT.Eval}(\text{pp}_R, \tilde{s}_j^R, \mathbb{C}_\ell)$ for $j \in C_R$, $\ell \in \{2, \text{dec}\}$. This uses only the public parameters pp_R , the fake shares $\tilde{s}_j^R$ (produced in Step 2), and the circuit $\mathbb{C}_\ell$ (determined by the public transcript). No real receiver secret is required.

Commitment equivocation. $(\hat{c}_j^\ell, \text{st}_j^\ell) \leftarrow \text{EQ.ECom}(\text{crs}_{\text{EQ}}, \text{td}_{\text{EQ}})$; $\tilde{d}_j^\ell \leftarrow \text{EQ.Equiv}(\text{crs}_{\text{EQ}}, \text{td}_{\text{EQ}}, \text{st}_j^\ell, (\tilde{s}_j^R, \hat{y}_j^\ell))$. The equivocation algorithms **EQ.ECom** and **EQ.Equiv** use only the global crs_{EQ} , the equivocation trapdoor td_{EQ} embedded in msk , and the target opening value, no receiver-specific secret.

NCE channel equivocation. $(\tilde{r}_{j,\cdot}^G, \tilde{r}_{j,\cdot}^E) \leftarrow \text{NCE.Sim}_2(\hat{y}_j^\ell, \text{st}_{j,\cdot}^{\text{NCE}})$. The NCE simulation algorithm NCE.Sim_2 takes the target plaintext $\hat{y}_j^\ell$ and the simulation state $\text{st}_{j,\cdot}^{\text{NCE}}$ (which is part of msk , generated during setup by NCE.Sim_1). It does *not* require the real NCE randomness or decryption keys specific to any receiver.

As a result, the complete input set of ThreshRFake is:

1. CRS_{TFDE} : public;
2. $\text{msk} = (\text{td}_{\text{EQ}}, \{\text{st}_{i,j}^{\text{NCE}}\}_{i,j})$: system-level secret state, *not* receiver-specific;
3. pp_R : public parameters of the receiver-side thresholdizer;
4. C_R : the set of coerced receivers (indices);
5. $\hat{m}$: the fake plaintext;
6. (μ_1, μ_2, μ_3) : the public transcript;
7. ρ : fresh random coins.

None of the items r , m , or $\{s_j^R\}_{j \in C_R}$ appear in this list. Therefore, any entity possessing msk , not necessarily any of the coerced receivers, can produce valid fake receiver views. $\square$

A.9 Proof of Theorem 5

Proof. We verify Definition 13. In TFDE, the “ciphertext” presented to the receiver coalition is the transcript $\text{tr}^* = (\mu_1, \mu_2, \mu_3)$, where each $\mu_\ell = \text{UT.Combine}(\text{pp}_\sigma, \{y_i^\ell\}_{i \in A_\sigma})$ is the output of the UT combine algorithm applied to the partial evaluations of the circuit $\mathbb{C}_\ell$. By the compactness property of UT (Definition 3), each partial evaluation y_i^ℓ has size $|y_i^\ell| \leq \text{poly}(\lambda, d, N)$, independent of the secret x_σ . The output of UT.Combine is the circuit evaluation $\mathbb{C}_\ell(x_\sigma)$, which equals the FDE round function output $P_\ell(\cdot)$. By assumption, the FDE protocol messages $P_1(s, m)$, $P_2(r, \mu_1)$, $P_3(s, m, \mu_1, \mu_2)$ each have size $\text{poly}(\lambda)$ (since FDE operates on polynomial-length messages under subexponentially secure iO and OWFs [15]). Therefore $|\text{tr}^*| = |\mu_1| + |\mu_2| + |\mu_3| \leq 3 \cdot \text{poly}(\lambda) = \text{poly}_1(\lambda)$, which is independent of n_S and n_R .

The encryption key in TFDE corresponds to the compound CRS $\text{CRS}_{\text{TFDE}} = (\text{CRS}_{\text{FDE}}, \text{crs}_{\text{EQ}}, \{\text{pk}_{i,j}, \text{sk}_{i,j}\}_{i,j})$ together with pp_S, pp_R . The CRS components have size $\text{poly}(\lambda)$; the NCE key pairs number $O((n_S + n_R)^2)$, each of size $\text{poly}(\lambda)$; the public parameters pp_σ have size $\text{poly}(\lambda, d, N)$ by UT compactness. Thus $|\text{ek}| \leq \text{poly}(\lambda, d, n_S, n_R) = \text{poly}_2(\lambda)$, since d, n_S, n_R are all polynomial in λ . $\square$

A.10 Proof of Theorem 6

Proof. We verify Definition 14. Suppose all parties are honest and $|A_S| \geq t_S$, $|A_R| \geq t_R$. We trace the protocol execution.

ShareGen runs $(\text{pp}_S, s_1^S, \dots, s_{n_S}^S) \leftarrow \text{UT.Setup}(1^\lambda, 1^d, (\mathbb{A}_S, n_S), x_S)$ with $x_S = (s \| m)$, and $(\text{pp}_R, s_1^R, \dots, s_{n_R}^R) \leftarrow \text{UT.Setup}(1^\lambda, 1^d, (\mathbb{A}_R, n_R), x_R)$ with $x_R = r$.

Round 1. Each sender $i \in A_S$ computes $y_i^1 \leftarrow \text{UT.Eval}(\text{pp}_S, s_i^S, \mathbb{C}_1)$, where $\mathbb{C}_1$ is defined so that $\mathbb{C}_1(x_S) = P_1(s, m)$. Since $A_S \in \mathbb{A}_S$ (authorized), by evaluation correctness of UT: $\mu_1 = \text{UT.Combine}(\text{pp}_S, \{y_i^1\}_{i \in A_S}) = \mathbb{C}_1(x_S) = P_1(s, m)$ with overwhelming probability. The verification step $\text{UT.Verify}(\text{pp}_S, y_i^1, \mathbb{C}_1) = 1$ holds for all honest parties by verification correctness of UT, so no honest party aborts.

Round 2. Each receiver $j \in A_R$ computes $y_j^2 \leftarrow \text{UT.Eval}(\text{pp}_R, s_j^R, \mathbb{C}_2)$ where $\mathbb{C}_2(x_R) = P_2(r, \mu_1)$ with μ_1 hardcoded. By evaluation correctness:

$$\mu_2 = \text{UT.Combine}(\text{pp}_R, \{y_j^2\}_{j \in A_R}) = \mathbb{C}_2(x_R) = P_2(r, \mu_1).$$

Round 3. We will have:

$$\mu_3 = \text{UT.Combine}(\text{pp}_S, \{y_i^3\}_{i \in A_S}) = \mathbb{C}_3(x_S) = P_3(s, m, \mu_1, \mu_2).$$

The circuit $\mathbb{C}_{\text{dec}}$ is defined so that $\mathbb{C}_{\text{dec}}(x_R) = \text{FDE.Dec}(r, \mu_1, \mu_2, \mu_3)$ with the transcript hardcoded. By evaluation correctness:

$$m' = \text{UT.Combine}(\text{pp}_R, \{y_j^{\text{dec}}\}_{j \in A_R}) = \mathbb{C}_{\text{dec}}(x_R) = \text{FDE.Dec}(r, \mu_1, \mu_2, \mu_3).$$

By the correctness of FDE (Definition 4) $\text{FDE.Dec}(r, \text{tr}(s, r, m)) = m$ except with negligible probability over uniform s, r . Since each UT.Combine invocation succeeds with overwhelming probability (by evaluation correctness of UT), and FDE.Dec succeeds with overwhelming probability (by correctness of FDE), a union bound over the four UT.Combine calls and the single FDE.Dec call yields:

$$\Pr[\text{ThreshDec}(\text{CRS}, \text{pp}_R, \{s_j^R\}_{j \in A_R}, \mu_1, \mu_2, \mu_3) = m] \geq 1 - 5 \cdot \text{negl}(\lambda) = 1 - \text{negl}(\lambda). \quad \square$$

A.11 Proof of Theorem 7

Proof. We verify Definition 15. In TFDE, partial decryption corresponds to the computation of partial evaluations $y_i^\ell = \text{UT.Eval}(\text{pp}_\sigma, s_i^\sigma, \mathbb{C}_\ell)$ in each round ℓ , and partial verification corresponds to $\text{UT.Verify}(\text{pp}_\sigma, y_i^\ell, \mathbb{C}_\ell)$. Fix any round $\ell \in \{1, 2, 3, \text{dec}\}$, any honest party i in the active set A_σ , and any circuit $\mathbb{C}_\ell$ determined by the protocol. Since party i is honest, it computes $y_i^\ell \leftarrow \text{UT.Eval}(\text{pp}_\sigma, s_i^\sigma, \mathbb{C}_\ell)$ faithfully.

By the verification correctness property of UT (Definition 3): for any honestly computed partial evaluation $y_i = \text{UT.Eval}(\text{pp}, s_i, C)$,

$$\Pr[\text{UT.Verify}(\text{pp}, y_i, C) = 1] \geq 1 - \text{negl}(\lambda).$$

This directly applies to each partial evaluation in TFDE. Since each party produces at most $L \leq 4$ partial evaluations (rounds 1, 3 for senders; rounds 2, dec for receivers), and each passes verification with overwhelming probability, a union bound over all $|A_\sigma| \cdot L$ verification checks yields:

$$\Pr[\forall i \in A_\sigma, \forall \ell: \text{UT.Verify}(\text{pp}_\sigma, y_i^\ell, \mathbb{C}_\ell) = 1] \geq 1 - |A_\sigma| \cdot L \cdot \text{negl}(\lambda) = 1 - \text{negl}(\lambda),$$

since $|A_\sigma| \leq N = \text{poly}(\lambda)$. Moreover, the robustness property of UT ensures that no PPT adversary, even given all shares $s_1, \dots, s_N$, can produce a fake partial evaluation $y_i^* \neq \text{UT.Eval}(\text{pp}, s_i, C)$ that passes UT.Verify , except with negligible probability. This guarantees that the verification step in each round of TFDE rejects any maliciously crafted partial evaluations with overwhelming probability, preventing corrupted parties from injecting invalid partial computations. $\square$

A.12 Proof of Theorem 8

Proof. We show that no PPT adversary $\mathcal{A}$ controlling at most $t_S - 1$ sender parties and at most $t_R - 1$ receiver parties can distinguish encryptions of m_0 from encryptions of m_1 with more than negligible advantage. The proof proceeds through three hybrids.

H_0 . The challenger executes TFDE with plaintext m_0 . The adversary $\mathcal{A}$ receives CRS_{TFDE} , pp_S , pp_R , the transcript $\text{tr}^* = (\mu_1, \mu_2, \mu_3)$, and the shares of corrupted parties $\{s_i^S\}_{i \in S_S^*}, \{s_j^R\}_{j \in S_R^*}$ where $|S_S^*| < t_S$ and $|S_R^*| < t_R$.

H_1 . Identical to H_0 except: the corrupted parties' shares are replaced by simulated shares. Specifically, for the sender side, instead of providing real shares $\{s_i^S\}_{i \in S_S^*}$ derived from $x_S = (s \| m_0)$, the challenger runs the UT simulator:

$$\{\tilde{s}_i^S\}_{i \in S_S^*} \leftarrow \mathcal{S}_{\text{UT}}(\text{pp}_S, \mathbb{C}_1, \mathbb{C}_1(x_S), S_S^*).$$

Note that the simulator receives only the public parameters, the circuit, and the circuit output $\mathbb{C}_1(x_S) = \mu_1$ (not x_S itself). Analogously for the receiver side.

We claim $H_0 \approx_c H_1$. This follows directly from the security property of UT (Definition 3): since $S_S^* \notin \mathbb{A}_S$ and $S_R^* \notin \mathbb{A}_R$ (sub-threshold, hence unauthorized), no PPT adversary can distinguish real shares and partial evaluations from simulated ones. Formally, by definition:

$$(\text{pp}_\sigma, \{s_i^\sigma\}_{i \in S_\sigma^*}, \{y_i^\ell\}_{i \in S_\sigma^*}) \approx_c (\text{pp}_\sigma, \mathcal{S}_{\text{UT}}(\text{pp}_\sigma, \mathbb{C}_\ell, \mathbb{C}_\ell(x_\sigma), S_\sigma^*)).$$

The distinguishing advantage is $\leq \text{Adv}_{\text{UT}}^{\text{sec}}(\lambda) = \text{negl}(\lambda)$.

H_2 . Identical to H_1 except the execution uses plaintext m_1 instead of m_0 . Observe that in H_1 , the adversary's view consists of:

1. The transcript $\text{tr}^* = (\mu_1, \mu_2, \mu_3)$, which equals $(P_1(s, m_b), P_2(r, \mu_1), P_3(s, m_b, \mu_1, \mu_2))$;
2. Simulated shares that depend only on pp_σ and the circuit outputs $\{\mu_1, \mu_2, \mu_3, m'\}$ (where $m' = \text{FDE.Dec}(r, \text{tr}^*)$);
3. Public parameters pp_S, pp_R .

After the UT simulation step, the adversary's information about the plaintext comes *solely* through the transcript tr^* (since the simulated shares and partial evaluations depend only on the circuit outputs, not on x_S or x_R directly).

We claim $H_1 \approx_c H_2$. We construct an adversary $\mathcal{B}$ against the semantic security of FDE. $\mathcal{B}$ receives $(\text{CRS}_{\text{FDE}}, \text{tr}^*)$ from the FDE semantic security challenger, where tr^* is either an encryption of m_0 or m_1 . $\mathcal{B}$ embeds this into TFDE by constructing the remaining CRS components (crs_{EQ}, NCE keys), generating simulated shares via UT simulation (which requires only tr^* and public parameters), and presenting the resulting view to $\mathcal{A}$.

Formally, $\mathcal{B}$ sets up the UT instances with circuits $\mathbb{C}_1, \mathbb{C}_2, \mathbb{C}_3$ such that their outputs match the components of tr^* . The UT simulator $\mathcal{S}_{\text{UT}}$ can produce simulated shares given only the circuit outputs (which are the components of tr^*). Thus, $\mathcal{B}$ can construct the complete adversarial view without knowing the plaintext.

Any advantage ϵ of $\mathcal{A}$ in distinguishing H_1 from H_2 translates to an advantage ϵ of $\mathcal{B}$ against FDE semantic security. Since FDE is semantically secure (which follows from bideniability [15]: bideniability implies that the transcript hides the plaintext, as otherwise a distinguisher between real and fake randomness would exist), we have $\epsilon \leq \text{Adv}_{\text{FDE}}^{\text{sem}}(\lambda) = \text{negl}(\lambda)$.

$$\begin{aligned} \text{Adv}_{\text{TFDE}, \mathcal{A}}^{\text{IND-CPA}}(\lambda) &= |\Pr[\mathcal{A}(H_0) = 1] - \Pr[\mathcal{A}(H_2) = 1]| \\ &\leq |\Pr[\mathcal{A}(H_0) = 1] - \Pr[\mathcal{A}(H_1) = 1]| + |\Pr[\mathcal{A}(H_1) = 1] - \Pr[\mathcal{A}(H_2) = 1]| \\ &\leq 2 \cdot \text{Adv}_{\text{UT}}^{\text{sec}}(\lambda) + \text{Adv}_{\text{FDE}}^{\text{sem}}(\lambda) \\ &\leq \text{negl}(\lambda), \end{aligned}$$

where the factor of 2 for UT accounts for simulation on both sender and receiver sides. $\square$

References

1. Shafi Goldwasser, Saleet Klein, and Daniel Wichs. The edited truth. In *Theory of Cryptography: 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part I*, page 305–340, Berlin, Heidelberg, 2017. Springer-Verlag.
2. Dana Dachman-Soled. On minimal assumptions for sender-deniable public key encryption. In *Proceedings of the 17th International Conference on Public-Key Cryptography — PKC 2014 - Volume 8383*, page 574–591, Berlin, Heidelberg, 2014. Springer-Verlag.
3. Daniel Apon, Xinyu Fan, and Feng-Hao Liu. Deniable attribute based encryption for branching programs from LWE. In Martin Hirt and Adam Smith, editors, *Theory of Cryptography*, volume 9986 of *LNCS*. Springer, 2016.
4. Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In Burton S. Kaliski, editor, *Advances in Cryptology — CRYPTO ’97*, volume 1294 of *LNCS*. Springer, 1997.
5. Angelo De Caro, Vincenzo Iovino, and Adam O’Neill. Deniable functional encryption. In *Public Key Cryptography (1)*, pages 196–222. Springer, 2016.
6. Daniel Hugenroth, Alberto Sonnino, Sam Cutler, and Alastair R. Beresford. Sloth: Key stretching and deniable encryption using secure elements on smartphones. *IACR Cryptol. ePrint Arch.*, 2023:1792, 2024.
7. Katriel Cohn-Gordon, Cas Cremers, Benjamin Dowling, Luke Garratt, and Douglas Stebila. A formal security analysis of the signal messaging protocol. In *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 451–466, 2017.
8. Niusen Chen, Bo Chen, and Weisong Shi. An application-layer plausibly deniable encryption system for wearable devices. *Discover Internet of Things*, 4(1), August 2024.
9. Niusen Chen, Bo Chen, and Weisong Shi. Mobiwear: a plausibly deniable encryption system for wearable mobile devices. In *EAI International Conference on Applied Cryptography in Computer and Communications*, pages 138–154. Springer, 2021.
10. GitHub - zorggomat/FractalCryptGUI: Free cross-platform deniable encryption cryptoarchiver — github.com. <https://github.com/zorggomat/FractalCryptGUI>. [Accessed 13-04-2026].
11. VeraCrypt - Free Open source disk encryption with strong security for the Paranoid — veracrypt.io. <https://veracrypt.io/en/Home.html>. [Accessed 16-12-2025].
12. Elia Anzuoni and Tommaso Gagliardoni. Shufflecake: Plausible deniability for multiple hidden filesystems on linux. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, CCS ’23*, page 3033–3047, New York, NY, USA, 2023. Association for Computing Machinery.
13. Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 475–484, 2014.
14. Adam O’Neill, Chris Peikert, and Brent Waters. Bi-deniable public-key encryption. In *Annual Cryptology Conference*, pages 525–542. Springer, 2011.
15. Ran Canetti, Sunoo Park, and Oxana Poburinnaya. Fully deniable interactive encryption. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology — CRYPTO 2020*, volume 12170 of *LNCS*. Springer, Cham, 2020.
16. Mona Soleymani, Navid Abapour, Elham Taghizadeh, Safieh Siadat, and Rasoul Karkehabadi. Fuzzy rule-based trust management model for the security of cloud computing. *Mathematical Problems in Engineering*, 2021(1):6629449, 2021.
17. Jinglong Luo, Yehong Zhang, Zhuo Zhang, Shiyu Liu, Ye Dong, Haoran Li, Yue Yu, Hui Wang, Xun Zhou, and Zenglin Xu. Cryptography-based privacy-preserving large language models: a lifecycle survey of frameworks, methods, and future directions. *Artificial Intelligence Review*, 59(2), December 2025.

18. Yi Zhang, Kun Tian, Yunfan Lu, Fengxia Liu, Cheng Li, Zixian Gong, Zhe Hu, Jia Li, and Qun Xu. Reparable threshold paillier encryption scheme for federated learning. *Soft Computing*, 29(7):3659–3664, April 2025.
19. Multi-Party Threshold Cryptography | CSRC | CSRC — csrc.nist.gov. <https://csrc.nist.gov/projects/threshold-cryptography>. [Accessed 13-04-2026].
20. Rene Peralta. NIST Internal or Interagency Report (NISTIR) 8214C, NIST First Call for Multi-Party Threshold Schemes — csrc.nist.gov. <https://csrc.nist.gov/pubs/ir/8214/c/final>. [Accessed 13-04-2026].
21. Kiarash Sedghighadikolaei and Attila Altay Yavuz. A survey of threshold signatures: Nist standards, post-quantum cryptography, exotic techniques, and real-world applications. *ACM Comput. Surv.*, October 2025. Just Accepted.
22. Dan Boneh, Rosario Gennaro, Steven Goldfeder, Aayush Jain, Sam Kim, Peter M. R. Rasmussen, and Amit Sahai. Threshold cryptosystems from threshold fully homomorphic encryption. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018*, pages 565–596. Springer International Publishing, 2018.
23. Josh Benaloh, Michael Naehrig, Olivier Pereira, and Dan S. Wallach. ElectionGuard: a cryptographic toolkit to enable verifiable elections. In *USENIX Security*, pages 5485–5502, Philadelphia, PA, August 2024. USENIX Association.
24. ElectionGuard - Official — electionguard.vote. <https://www.electionguard.vote/spec/>. [Accessed 20-07-2024].
25. Manage quorum authentication (M of N access control) using CloudHSM CLI - AWS CloudHSM — docs.aws.amazon.com. <https://docs.aws.amazon.com/cloudhsm/latest/userguide/quorum-auth-chsm-cli.html>. [Accessed 13-04-2026].
26. DigiCert documentation — docs.digicert.com. <https://docs.digicert.com/?lang=en>. [Accessed 13-04-2026].
27. Alexander Ziller, Andrew Trask, Antonio Lopardo, Benjamin Szymkow, Bobby Wagner, Emma Bluemke, Jean-Mickael Nounahon, Jonathan Passerat-Palmbach, Kritika Prakash, Nick Rose, Théo Ryffel, Zarreen Naowal Reza, and Georgios Kaissis. *PySyft: A Library for Easy Federated Learning*, page 111–139. Springer International Publishing, 2021.
28. Dan Boneh, Rosario Gennaro, Steven Goldfeder, and Sam Kim. A lattice-based universal thresholdizer for cryptographic systems. *Cryptology ePrint Archive*, Paper 2017/251, 2017.
29. Aayush Jain, Peter M. R. Rasmussen, and Amit Sahai. Threshold fully homomorphic encryption. *Cryptology ePrint Archive*, Paper 2017/257, 2017.
30. Lior Rotem and Gil Segev. *Non-malleable Vector Commitments via Local Equivocability*, page 415–446. Springer International Publishing, 2021.
31. Zvika Brakerski, Pedro Branco, Nico Döttling, Sanjam Garg, and Giulio Malavolta. *Constant Ciphertext-Rate Non-committing Encryption from Standard Assumptions*, page 58–87. Springer International Publishing, 2020.
32. Rikke Bendlin, Jesper Buus Nielsen, Peter Sebastian Nordholt, and Claudio Orlandi. Lower and upper bounds for deniable public-key encryption. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 125–142. Springer, 2011.
33. Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In Moti Yung, editor, *Advances in Cryptology — CRYPTO 2002*, volume 2442 of *LNCS*. Springer, 2002.
34. Shaoquan Jiang, Yeow Meng Chee, San Ling, Huaxiong Wang, and Chaoping Xing. A new framework for deniable secure key exchange. *Information and Computation*, 285:104866, 2022.
35. Ran Canetti, Ivan Damgård, Sebastian Kolby, Divya Ravi, and Sophia Yakubov. Deniable secret sharing. In *Theory of Cryptography: 23rd International Conference, TCC 2025, Aarhus, Denmark, December 1–5, 2025, Proceedings, Part II*, page 399–427, Berlin, Heidelberg, 2025. Springer-Verlag.

36. Ivan Damgård and Jesper Buus Nielsen. Improved non-committing encryption schemes based on a general complexity assumption. In Mihir Bellare, editor, *Advances in Cryptology — CRYPTO 2000*, volume 1880 of *LNCS*. Springer, 2000.
37. Atsushi Waseda and Ryo Nojima. On limitation of plausible deniability. In Kazumaro Aoki and Akira Kanaoka, editors, *Advances in Information and Computer Security. IWSEC 2020*, volume 12231 of *LNCS*. Springer, Cham, 2020.
38. S. Garg et al. Threshold encryption with silent setup. In *CRYPTO*, pages 352–386. Springer, 2024.
39. B. Waters and D.J. Wu. Silent threshold cryptography from pairings: Expressive policies in the plain model. Cryptology ePrint Archive, Paper 2025/1547, 2025.
40. M. Hall-Andersen, M. Simkin, and B. Wagner. Silent threshold encryption with one-shot adaptive security. Cryptology ePrint Archive, Paper 2025/1384, 2025.
41. J. Bormet et al. BEAST-MEV: Batched threshold encryption with silent setup for MEV prevention. Cryptology ePrint Archive, Paper 2025/1419, 2025.
42. Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. volume 28, pages 1364–1396. SIAM, 1999.
43. Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing (STOC) 1985 (and published in full in SIAM J. Comput.)*, volume 18 of *SIAM Journal on Computing*, pages 186–208. SIAM, 1989.
44. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), September 2009.
45. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 1–23. Springer, 2010.
46. Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013*, volume 8042 of *LNCS*. Springer, 2013.
47. Alfredo De Santis, Silvio Micali, and Giuseppe Persiano. Non-interactive zero-knowledge with preprocessing. In Shafi Goldwasser, editor, *Advances in Cryptology — CRYPTO’88*, volume 403 of *LNCS*. Springer, New York, NY, 1990.
48. Manuel Blum. Coin flipping by telephone a protocol for solving impossible problems. *SIGACT News*, 15(1):23–27, January 1983.
49. Zvika Brakerski, Pedro Branco, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Constant ciphertext-rate non-committing encryption from standard assumptions. In *Theory of Cryptography – TCC 2020*, volume 12550 of *Lecture Notes in Computer Science*, pages 58–87. Springer, 2020.
50. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC 2005)*, pages 84–93. ACM, 2005.
51. Venkatesan Guruswami and Madhu Sudan. List decoding algorithms for certain concatenated codes. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC 2000)*, pages 181–190. ACM, 2000.
52. Mike Hamburg. Decaf: Eliminating cofactors through point compression. In *Advances in Cryptology – CRYPTO 2015*, page 705–723, Berlin, Heidelberg, 2022. Springer-Verlag.
53. John M. Pollard. Monte carlo methods for index computation (mod p). *Mathematics of Computation*, 32(143):918–924, 1978.
54. National Institute of Standards and Technology. FIPS PUB 186-5: Digital Signature Standard (DSS). Technical report, NIST, 2023.