Homomorphic Secret-Sharing with Certified Deletion

Nikhil Pappu[†]

[†]Portland State University
nikpappu@pdx.edu

October 14, 2024

Abstract

Contents

1 Preliminaries **3**

2 HSS with Certified Deletion **3**
2.1 HSS-CD Syntax . . . 3
2.2 Additive HSS-CD Syntax . . . 4
2.3 Security Definitions . . . 4

3 Feasibility Results **5**
3.1 FHE-CD based Construction . . . 5
3.2 Impossibility Results . . . 7

1 Preliminaries

2 HSS with Certified Deletion

Unless otherwise specified, we will consider the following kind of HSS schemes by default:

- Those that work for all PPT computable circuits.
- Are 2-out-of-2 secret-sharing schemes.
- Allow evaluation for a single secret.

An HSS scheme with certified deletion must have the following syntax and correctness requirements:

2.1 HSS-CD Syntax

A scheme satisfying the HSS-CD syntax is a tuple of 5 algorithms $\mathsf{HSS\text{-}CD} = \mathsf{HSS\text{-}CD}.(\mathcal{Share}, \mathcal{Eval}, \mathcal{Del}, \mathsf{Vrfy}, \mathcal{Rec})$ with the following properties:

Syntax:

$\mathcal{Share}(s) \to (sh_0^0, \mathsf{vk}_0), (sh_1^0, \mathsf{vk}_1)$: The sharing algorithm outputs quantum (possibly-entangled) secret-shares sh_0^0, sh_1^0 encoding an input secret s. It also outputs the corresponding classical verification keys $\mathsf{vk}_0, \mathsf{vk}_1$.

$\mathcal{Eval}(C_j, i, sh_i^{j-1}) \to sh_i^j$: The evaluation algorithm takes the description of a PPT computable circuit C_j, an index $i \in \{0,1\}$, and an input share sh_i^{j-1}. It outputs a possibly-altered output share sh_i^j.

$\mathcal{Del}(i, sh_i^j) \to \mathsf{cert}_i$: The deletion algorithm takes an index $i \in \{0,1\}$, a corresponding share sh_i^j, and produces a deletion certificate cert_i.

$\mathsf{Vrfy}(i, \mathsf{vk}_i, \mathsf{cert}_i) \to \top/\bot$: The verification algorithm takes an index $i \in \{0,1\}$, the corresponding verification key vk_i and a certificate cert_i. It outputs $\top$ or $\bot$.

$\mathcal{Rec}(sh_0^q, sh_1^q) \to (d_1, \cdots, d_q)$: The reconstruction algorithm takes two evaluated input shares sh_0^q, sh_1^q and outputs a q-tuple $(d_1, \cdots, d_q)$.

Evaluation Correctness: $\forall$ PPT C, the following condition holds for all $q = \mathrm{poly}(\lambda)$:

$$\Pr\left[(d_1, \cdots, d_q) = (C_1(s), \cdots, C_q(s)) \ : \ \begin{array}{l} (sh_0^0, \mathsf{vk}_0), (sh_1^0, \mathsf{vk}_1) \leftarrow \mathcal{Share}(s) \\ \forall i, j \in \{0,1\} \times [q] : sh_i^j \leftarrow \mathcal{Eval}(C_j, i, sh_i^{j-1}) \\ (d_1, \cdots, d_q) \leftarrow \mathcal{Rec}(sh_0^q, sh_1^q) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

Deletion Correctness: The following condition holds for all $i \in \{0,1\}$ and $q = \mathrm{poly}(\lambda)$:

$$\Pr\left[\mathsf{Vrfy}(i, \mathsf{vk}_i, \mathsf{cert}_i) \to \top \ : \ \begin{array}{l} (sh_0^0, \mathsf{vk}_0), (sh_1^0, \mathsf{vk}_1) \leftarrow \mathcal{Share}(s) \\ \forall j \in [q] : sh_i^j \leftarrow \mathcal{Eval}(C_j, i, sh_i^{j-1}) \\ \mathsf{cert}_i \leftarrow \mathcal{Del}(i, sh_i^q) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

Compactness: The following condition holds for all $i \in \{0,1\}$ and $q = \mathrm{poly}(\lambda)$, where l_q denotes the output length of the circuit C_q:

$$\left[|sh_i^q| - |sh_i^{q-1}| = \mathrm{poly}(1^\lambda, l_q) \ : \ \begin{array}{l} (sh_0^0, \mathsf{vk}_0), (sh_1^0, \mathsf{vk}_1) \leftarrow \mathcal{Share}(s) \\ \forall j \in [q] : sh_i^j \leftarrow \mathcal{Eval}(C_j, i, sh_i^{j-1}) \end{array}\right]$$

2.2 Additive HSS-CD Syntax

A scheme satisfying the additive HSS-CD syntax is a tuple of 4 algorithms $\mathsf{HSS\text{-}CD} = \mathsf{HSS\text{-}CD}.(\mathcal{Share}, \mathcal{Eval}, \mathcal{Del}, \mathsf{Vrfy})$ with the following properties:

Syntax:

$\mathcal{Share}(s) \to (\mathcal{sh}_0^0, \mathsf{vk}_0), (\mathcal{sh}_1^0, \mathsf{vk}_1)$: The sharing algorithm outputs quantum (possibly-entangled) secret-shares $\mathcal{sh}_0^0, \mathcal{sh}_1^0$ encoding an input secret s. It also outputs the corresponding classical verification keys $\mathsf{vk}_0, \mathsf{vk}_1$.

$\mathcal{Eval}(C_j, i, (\mathsf{sh}_i^{j-1}, \mathcal{sh}_i^{j-1})) \to (\mathsf{sh}_i^j, \mathcal{sh}_i^j)$: The evaluation algorithm takes the description of a PPT computable circuit C_j, an index $i \in \{0,1\}$, a quantum input share $\mathcal{sh}_i^{j-1}$, and a classical input share sh_i^{j-1} where $\mathsf{sh}_i^0 = \bot$. It outputs a quantum output share $\mathcal{sh}_i^j$ and a classical output share sh_i^j.

$\mathcal{Del}(i, \mathcal{sh}_i^j) \to \mathsf{cert}_i$: The deletion algorithm takes an index $i \in \{0,1\}$, a corresponding quantum share $\mathcal{sh}_i^j$, and produces a deletion certificate cert_i.

$\mathsf{Vrfy}(i, \mathsf{vk}_i, \mathsf{cert}_i) \to \top/\bot$: The verification algorithm takes an index $i \in \{0,1\}$, the corresponding verification key vk_i and a certificate cert_i. It outputs $\top$ or $\bot$.

Evaluation Correctness: $\forall$ PPT C, the following condition holds for all $q = \mathrm{poly}(\lambda)$:

$$\Pr\left[\mathsf{sh}_0^q \oplus \mathsf{sh}_1^q = C_1(s)\| \cdots \|C_q(s) \ :\ \begin{array}{l} (\mathcal{sh}_0^0, \mathsf{vk}_0), (\mathcal{sh}_1^0, \mathsf{vk}_1) \leftarrow \mathcal{Share}(s) \\ \forall i, j \in \{0,1\} \times [q] : (\mathsf{sh}_i^j, \mathcal{sh}_i^j) \leftarrow \mathcal{Eval}(C_j, i, (\mathsf{sh}_i^{j-1}, \mathcal{sh}_i^{j-1})) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

In the case of additive HSS-CD, we will consider the following deletion guarantee by default, which is weaker than the standard deletion correctness guarantee:

Delete-before-Eval Correctness: The following condition holds for all $i \in \{0,1\}$:

$$\Pr\left[\mathsf{Vrfy}(i, \mathsf{vk}_i, \mathsf{cert}_i) \to \top \ :\ \begin{array}{l} (\mathcal{sh}_0^0, \mathsf{vk}_0), (\mathcal{sh}_1^0, \mathsf{vk}_1) \leftarrow \mathcal{Share}(s) \\ \mathsf{cert}_i \leftarrow \mathcal{Del}(i, \mathcal{sh}_i^0) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

Deletion Correctness (Optional): The following condition holds for all $i \in \{0,1\}$ and $q = \mathrm{poly}(\lambda)$:

$$\Pr\left[\mathsf{Vrfy}(i, \mathsf{vk}_i, \mathsf{cert}_i) \to \top \ :\ \begin{array}{l} (\mathcal{sh}_0^0, \mathsf{vk}_0), (\mathcal{sh}_1^0, \mathsf{vk}_1) \leftarrow \mathcal{Share}(s) \\ \forall j \in [q] : (\mathsf{sh}_i^j, \mathcal{sh}_i^j) \leftarrow \mathcal{Eval}(C_j, i, (\mathsf{sh}_i^{j-1}, \mathcal{sh}_i^{j-1})) \\ \mathsf{cert}_i \leftarrow \mathcal{Del}(i, \mathcal{sh}_i^q) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

2.3 Security Definitions

Statistical/Computational Deletion Security wrt Share j: The following security notion is defined wrt a non-local quantum adversary $(\mathcal{A}_0, \mathcal{A}_1)$:

$\mathsf{Expt}^{\mathsf{del}}_{\mathsf{HSS\text{-}CD},(\mathcal{A}_0,\mathcal{A}_1)}(1^\lambda, j, b)$**:**

1. $\mathcal{A}_0$ sends $(s_0, s_1) \in \{0,1\}^\lambda$ to the challenger.
2. The challenger runs $(\mathcal{sh}_0^0, \mathsf{vk}_0), (\mathcal{sh}_1^0, \mathsf{vk}_1) \leftarrow \mathcal{Share}(s_b)$ and sends each $\mathcal{sh}_i^0$ to party P_i.
3. $\mathcal{A}_j$ sends (cert_j, R_j) and $\mathcal{A}_{1-j}$ sends R_{1-j} where R_0, R_1 are some registers.
4. If $\mathsf{Vrfy}(j, \mathsf{vk}_j, \mathsf{cert}_j) = \top$, then output (R_1, R_2).

Statistical Deletion Security wrt Share j holds if the following holds:

$$TD\Big(\mathsf{Expt}^{\mathsf{del}}_{\mathsf{HSS\text{-}CD},(\mathcal{A}_0,\mathcal{A}_1)}(1^\lambda, j, 0), \mathsf{Expt}^{\mathsf{del}}_{\mathsf{HSS\text{-}CD},(\mathcal{A}_0,\mathcal{A}_1)}(1^\lambda, j, 1)\Big) \leq \mathsf{negl}(\lambda)$$

Computational Deletion Security wrt Share j holds if the following holds for all QPT $\mathcal{A}$:

$$\left| \Pr\left[\mathcal{A}\Big(\mathsf{Expt}^{\mathsf{del}}_{\mathsf{HSS\text{-}CD},(\mathcal{A}_0,\mathcal{A}_1)}(1^\lambda, j, 0)\Big) = 1\right] - \Pr\left[\mathcal{A}\Big(\mathsf{Expt}^{\mathsf{del}}_{\mathsf{HSS\text{-}CD},(\mathcal{A}_0,\mathcal{A}_1)}(1^\lambda, j, 1)\Big) = 1\right] \right| \leq \mathsf{negl}(\lambda)$$

Statistical/Computational Double-Deletion Security: The following security notion is defined wrt a non-local quantum adversary $(\mathcal{A}_0, \mathcal{A}_1)$:

$\mathsf{Exp}^{\mathsf{del\text{-}2}}_{\mathsf{HSS\text{-}CD},(\mathcal{A}_0,\mathcal{A}_1)}(1^\lambda, b)$**:**

1.
2. $\mathcal{A}_0$ sends $(s_0, s_1) \in \{0,1\}^\lambda$ to the challenger. The challenger runs $(\mathit{sh}_0^0, \mathsf{vk}_0), (\mathit{sh}_1^0, \mathsf{vk}_1) \leftarrow \mathcal{S}\mathit{hare}(s_b)$ and sends each sh_i^0 to party P_i.
3. $\mathcal{A}_0$ sends (cert_0, R_0) and $\mathcal{A}_1$ sends (cert_1, R_1) where R_0, R_1 are some registers.
4. If $\mathsf{Vrfy}(0, \mathsf{vk}_0, \mathsf{cert}_0) = \mathsf{Vrfy}(1, \mathsf{vk}_1, \mathsf{cert}_1) = \top$, then output (R_0, R_1).

Statistical Double-Deletion Security holds if the following holds:

$$TD\Big(\mathsf{Exp}^{\mathsf{del\text{-}2}}_{\mathsf{HSS\text{-}CD},(\mathcal{A}_0,\mathcal{A}_1)}(1^\lambda, 0), \mathsf{Exp}^{\mathsf{del\text{-}2}}_{\mathsf{HSS\text{-}CD},(\mathcal{A}_0,\mathcal{A}_1)}(1^\lambda, 1)\Big) \leq \mathsf{negl}(\lambda)$$

Computational Double-Deletion Security holds if the following holds for all QPT $\mathcal{A}$:

$$\left| \Pr\left[\mathcal{A}\Big(\mathsf{Exp}^{\mathsf{del\text{-}2}}_{\mathsf{HSS\text{-}CD},(\mathcal{A}_0,\mathcal{A}_1)}(1^\lambda, 0)\Big) = 1\right] - \Pr\left[\mathcal{A}\Big(\mathsf{Exp}^{\mathsf{del\text{-}2}}_{\mathsf{HSS\text{-}CD},(\mathcal{A}_0,\mathcal{A}_1)}(1^\lambda, 1)\Big) = 1\right] \right| \leq \mathsf{negl}(\lambda)$$

Hereafter, we will use *stat* to denote statistical security and *comp* to denote computational security.

Definition 2.1 ((Additive) (X, Y)-HSS-CD scheme). *An (Additive) (X, Y)-HSS-CD scheme for X, Y* $\in \{stat, comp\}^2$ *is a scheme that satisfies the (Additive) HSS-CD syntax, the X deletion security for share* 0*, and the Y deletion security for share* 1*.*

Definition 2.2 ((Additive) (X)-HSS-CD scheme). *An (Additive) (X)-HSS-CD scheme for* $X \in \{stat, comp\}$ *is a scheme satisfying the (Additive) HSS-CD syntax and the X double-deletion security.*

Remark 2.3. Observe that a (stat, comp)-HSS-CD scheme is also a (stat)-HSS-CD scheme. Likewise, a (comp, comp)-HSS-CD scheme is also a (comp)-HSS-CD scheme.

3 Feasibility Results

3.1 FHE-CD based Construction

We construct a (stat, comp)-HSS-CD scheme $\mathsf{HSS\text{-}CD} = \mathsf{HSS\text{-}CD}.(\mathcal{S}\mathit{hare}, \mathcal{E}\mathit{val}, \mathcal{D}\mathit{el}, \mathsf{Vrfy}, \mathcal{R}\mathit{ec})$ using the following building blocks.

- Fully Homomorphic Encryption with Certified Deletion (FHE-CD) scheme $\mathsf{FHE\text{-}CD} = \mathsf{FHE\text{-}CD}.(\mathsf{Setup}, \mathcal{E}\mathit{nc}, \mathcal{D}\mathit{ec}, \mathcal{E}\mathit{val}, \mathcal{D}\mathit{el}, \mathsf{Vrfy})$.

- Secret Sharing with Certified Deletion (SS-CD) scheme $\mathsf{SS\text{-}CD} = \mathsf{SS\text{-}CD}.(\mathcal{Share}, \mathcal{Rec}, \mathcal{Del}, \mathsf{Vrfy})$.

The construction is as follows.

$\mathsf{HSS\text{-}CD}.\mathcal{Share}(s)$:

1. Generate $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE\text{-}CD.Setup}(1^\lambda)$.
2. Compute $(\mathsf{fhecd}.ct^0, \mathsf{fhecd.vk}) \leftarrow \mathsf{FHE\text{-}CD}.\mathcal{Enc}(s)$.
3. Compute $(\mathsf{sscd}.sh, \mathsf{sscd.csh}), \mathsf{sscd.vk} \leftarrow \mathsf{SS\text{-}CD}.\mathcal{Share}(\mathsf{sk})$.
4. Set $sh_0^0 := (\mathsf{fhecd.pk}, \mathsf{fhecd}.ct^0, \mathsf{sscd.csh})$ and $\mathsf{vk}_0 := \mathsf{fhecd.vk}$.
5. Set $sh_1^0 := \mathsf{sscd}.sh$ and $\mathsf{vk}_1 := \mathsf{sscd.vk}$.
6. Output $(sh_0^0, \mathsf{vk}_0), (sh_1^0, \mathsf{vk}_1)$.

$\mathsf{HSS\text{-}CD}.\mathcal{Eval}(C_j, i, sh_i^{j-1})$: If $i = 1$, set $sh_1^j := sh_1^{j-1}$. Else, execute the following:

1. Parse sh_0^{j-1} as $(\mathsf{fhecd.pk}, \mathsf{fhecd}.ct^{j-1}, \mathsf{sscd.csh})$.
2. Compute $\mathsf{fhecd}.ct^j \leftarrow \mathsf{FHE\text{-}CD}.\mathcal{Eval}(\mathsf{fhecd.pk}, C_j, \mathsf{fhecd}.ct^{j-1})$
3. Set $sh_0^j := (\mathsf{fhecd.pk}, \mathsf{fhecd}.ct^{j-1}\mathsf{sscd.csh})$.
4. Output sh_i^j.

$\mathsf{HSS\text{-}CD}.\mathcal{Del}(i, sh_i^j)$:

1. If $i = 0$, execute the following:
   (i) Parse sh_0^j as $(\mathsf{fhecd.pk}, \mathsf{fhecd}.ct^j, \mathsf{sscd.csh})$.
   (ii) Compute and output $\mathsf{cert}_0 \leftarrow \mathsf{FHE\text{-}CD}.\mathcal{Del}(\mathsf{fhecd}.ct^j)$.
2. If $i = 1$, execute the following:
   (i) Parse sh_1^j as $\mathsf{sscd}.sh$.
   (ii) Compute and output $\mathsf{cert}_1 \leftarrow \mathsf{SS\text{-}CD}.\mathcal{Del}(\mathsf{sscd}.sh)$.

$\mathsf{HSS\text{-}CD.Vrfy}(i, \mathsf{vk}_i, \mathsf{cert}_i)$:

1. If $i = 0$, output $\mathsf{ans}_0 \leftarrow \mathsf{FHE\text{-}CD.Vrfy}(\mathsf{vk}_0, \mathsf{cert}_0)$.
2. If $i = 1$, output $\mathsf{ans}_0 \leftarrow \mathsf{SS\text{-}CD.Vrfy}(\mathsf{vk}_1, \mathsf{cert}_1)$.

$\mathsf{HSS\text{-}CD}.\mathcal{Rec}(sh_0^q, sh_1^q)$:

1. Parse sh_0^q as $(\mathsf{fhecd.pk}, \mathsf{fhecd}.ct^q, \mathsf{sscd.csh})$.
2. Parse sh_1^q as $\mathsf{sscd}.sh$.
3. Compute $\mathsf{sk} \leftarrow \mathsf{SS\text{-}CD}.\mathcal{Dec}(\mathsf{sscd}.sh, \mathsf{sscd.csh})$.
4. Compute and output $(d_1, \ldots, d_q) \leftarrow \mathsf{FHE\text{-}CD}.\mathcal{Dec}(\mathsf{sk}, \mathsf{fhecd}.ct^q)$.

Theorem 3.1. *There exists a (stat, comp)-HSS-CD scheme assuming the existence of a fully homomorphic encryption scheme with certified deletion (FHE-CD), and a secret-sharing scheme with certified deletion (SS-CD).*

Proof. We will prove that the construction $\mathsf{HSS\text{-}CD}$ is a (stat, comp)-HSS-CD scheme. First, we will assume that $(\mathcal{A}_0, \mathcal{A}_1)$ is a non-local adversary that breaks the statistical deletion security of share 0. We will use this adversary to break the certified deletion security of the FHE-CD scheme $\mathsf{FHE\text{-}CD}$. Consider a QPT reduction $\mathcal{R}$ that runs as follows in the FHE-CD game:

Execution of $\mathcal{R}^{(\mathcal{A}_0,\mathcal{A}_1)}$ in $\mathsf{Exp}^{\mathsf{fhe\text{-}cd}}_{\mathsf{FHE\text{-}CD},\mathcal{R}}(1^\lambda, b)$:

1. $\mathcal{A}_0$ sends $(s_0, s_1) \in \{0,1\}^\lambda$ to $\mathcal{R}$, which $\mathcal{R}$ forwards to the challenger.
2. The challenger samples $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda)$ and sends pk to $\mathcal{R}$.
3. The challenger encrypts s_b as $ct \leftarrow \mathcal{E}nc(\mathsf{pk}, s_b)$ and sends ct to $\mathcal{R}$.
4. $\mathcal{R}$ computes $sh_0^0 := (\mathsf{pk}, ct, \mathsf{sscd.csh})$, where $\mathsf{sscd.csh} \leftarrow \mathsf{SS\text{-}CD.Sim}(1^\lambda)$.
5. $\mathcal{R}$ runs $\mathcal{A}_0$ on input sh_0^0. If $\mathcal{A}_0$ outputs (cert_0, R_0), $\mathcal{R}$ sends cert_0 to the challenger.
6. The challenger computes $\mathsf{ans} \leftarrow \mathsf{Vrfy}(\mathsf{vk}, \mathsf{cert}_0)$. If $\mathsf{ans} = \top$, it sends sk to $\mathcal{R}$. Else, it outputs $\bot$.
7. $\mathcal{R}$ computes $\mathsf{sscd}.sh$ conditioned on $(\mathsf{sscd}.sh, \mathsf{sscd.sh})$ encoding sk.
8. $\mathcal{R}$ sends $sh_1^0 := \mathsf{sscd}.sh$ to $\mathcal{A}_1$. If $\mathcal{A}_1$ outputs R_1, send (R_0, R_1) to the challenger.

We will now argue that if $(\mathcal{A}_0, \mathcal{A}_1)$ break statistical security wrt share 0, then $\mathcal{R}$ breaks the certified-deletion security of $\mathsf{FHE\text{-}CD}$. Observe that the view of $\mathcal{A}_0$ in the reduction is identically distributed to its view in $\mathsf{Expt}^{\mathsf{del}}_{\mathsf{HSS\text{-}CD},(\mathcal{A}_0,\mathcal{A}_1)}(1^\lambda, 0, b)$. Now, notice that if $\mathsf{HSS\text{-}CD.Vrfy}(0, \mathsf{vk}_0, \mathsf{cert}_0)$ passes, then $\mathsf{FHE\text{-}CD.Vrfy}(\mathsf{vk}, \mathsf{cert}_0)$ also passes. Consequently, $\mathcal{R}$ receives the secret key sk. By the information-theoretic secrecy of the scheme $\mathsf{SS\text{-}CD}$, the view of $\mathcal{A}_1$ is identically distributed to that in the original experiment. As a result, (R_0, R_1) are identically distributed to that of the HSS-CD game. By assumption, there exists an unbounded algorithm that can use (R_0, R_1) to guess b with non-negligible probability. This breaks the certified-deletion security of $\mathsf{FHE\text{-}CD}$.

Next, we will assume that $(\mathcal{A}_0, \mathcal{A}_1)$ is a non-local adversary that breaks the computational deletion security of share 1. We will use this adversary to break the certified deletion security of the SS-CD scheme $\mathsf{SS\text{-}CD}$. Consider a non-local reduction $(\mathcal{R}_0, \mathcal{R}_1)$ that runs as follows:

Execution of $(\mathcal{R}_0^{\mathcal{A}_0}, \mathcal{R}_1^{\mathcal{A}_1})$ in $\mathsf{Exp}^{\mathsf{ss\text{-}cd}}_{\mathsf{SS\text{-}CD},(\mathcal{R}_0,\mathcal{R}_1)}(1^\lambda, b)$:

1. $\mathcal{R}_0$ samples $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE\text{-}CD.Setup}(1^\lambda)$. It sets $s_0 := 0^\lambda$ and $s_1 := \mathsf{sk}$ and sends (s_0, s_1) to the challenger.
2. The challenger computes $(sh, \mathsf{csh}, \mathsf{vk}) \leftarrow \mathsf{SS\text{-}CD}.\mathcal{S}hare(s_b)$. It sends csh to $\mathcal{R}_0$ and sh to $\mathcal{R}_1$.

□

3.2 Impossibility Results

Theorem 3.2. *Any (stat, stat)-HSS-CD scheme is also an information-theoretic HSS scheme.*

Theorem 3.3. *The classical impossibility of [**?**] extends to the setting of quantum secret-shares.*

Theorem 3.4. *There does not exist a (comp)-HSS-CD scheme with additive reconstruction, given that the algorithm $\mathcal{S}hare$ only outputs shares sh_1, sh_2 that are non-entangled.*

[GVW12]

References

[GVW12] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption with bounded collusions via multi-party computation. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 162–179. Springer, Heidelberg, August 2012. (Cited on page 7.)