A Compact Post-quantum Strong Designated Verifier Signature Scheme from Isogenies

Farzin Renan[1], Wendi Gao[2], and Jason T. LeGrow[2]

[1]Independent Researcher
[2]Department of Mathematics, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, USA
farzin.renan@gmail.com, wendig@vt.edu, jlegrow@vt.edu

Abstract

Strong designated verifier signatures are a variant of digital signatures in which only one party—the designated verifier—is able to verify the authenticity of a signature. We propose a new strong designated verifier signature scheme based on (abelian) cryptographic group actions.

Keywords: Post-quantum cryptography, Isogeny-based cryptography, Strong designated verifier signatures

Acknowledgements. Jason T. LeGrow was partially supported by National Science Foundation grant NSF CNS 2528910. We would like to express our gratitude to Sarah Arpin, Frederik Vercauteren, and the Isocrypt Study Group for their constructive feedback, which helped improve this paper. We are also especially grateful to Péter Kutas for his insightful discussions during the early stages of the draft.

1 Introduction

Digital signatures are cryptographic protocols used to ensure data integrity and authenticity. Ordinary digital signatures provide *non-repudiation* by default: once Alice signs a message and publishes the signature, she can not later deny having signed the message, since anyone with access to her public key can verify that the signature is hers. While non-repudiation is a useful—or at worst, benign—feature in many applications, it is not always desirable. In 1996, Jakobsson, Sako, and Impagliazzo [1] considered the task of authenticated off-the-record messaging, in which Alice wishes to authenticate herself and her messages to Bob, but in a way that does not allow Bob to convince any third party that the messages really came from Alice. Clearly, an ordinary digital signature would not work in this application.

To achieve this, Jakobsson et al. [1] and Chaum [2] independently introduced the notion of *Designated Verifier Signatures* (DVS) in 1996. In a DVS scheme, only a specific verifier—chosen by the signer at the time of signing—is capable of

verifying the authenticity of a signature, while any third party remains unconvinced. This is achieved by ensuring that the designated verifier is capable of *simulating* signatures that are computationally indistinguishable from authentic ones. When Bob receives a designated verifier signature from Alice, he knows that he did *not* construct it, and so he is convinced it is valid. But if he sends the signature to a third party, that third party cannot tell whether it was produced by Alice (valid) or by Bob (invalid). This crucial feature—called *non-transferability*—ensures that only the designated verifier is convinced of a signature's authenticity.

Building on preliminary work of Jakobson et al., in 2004 Laguillaumie and Vergnaud [3] introduced *Strong Designated Verifier Signatures* (SDVS) as an extension of designated verifier signatures that offers enhanced privacy guarantees. SDVS schemes provide additional privacy guarantees: in particular, they must satisfy the *Privacy of Signer's Identity* property, which says that given a valid signature and two candidate signing public keys, it is computationally infeasible for anyone other than the designated verifier to determine which key was used.

1.1 Related Work

Strong designated verifier signature schemes have been extensively studied in both classical and post-quantum cryptographic settings.

Classical Constructions. Independent of the foundational work by Laguillaumie et al. [3], Saeednia et al. [4] proposed an efficient SDVS scheme by adapting the Schnorr signature and incorporating the designated verifier's secret key into the verification process. Subsequently, Huang et al. [5] and Kang et al. [6] introduced identity-based SDVS (IBSDVS) schemes and proved their unforgeability under the Bilinear Diffie–Hellman (BDH) assumption. Unfortunately, these constructions are rendered insecure in the quantum setting, due to Shor's algorithm [7].

Post-Quantum Constructions. There are a number of SDVS proposals from post-quantum assumptions.

Code-based SDVS constructions were proposed independently by Assar [8], Ren et al. [9], and Shooshtari et al. [10], based on the hardness of the bounded syndrome decoding problem. Unfortunately, it was shown in [11] that the constructions by Ren et al. and Shooshtari et al. fail to satisfy the critical non-transferability property, which is central to the SDVS notion.

Wang et al. [22] propose a lattice-based SDVS using *preimage sampleable functions* (PSFs) and Bonsai Trees. Their work builds on the framework of Gentry, Peikert and Vaikunthanathan [21], which introduced PSFs as a primitive for lattice-based signatures. Although the construction offers provable security in the random oracle model, it suffers from efficiency issues due to large key and signature sizes. In 2016, Noh and Jeong [23] proposed a scheme in the standard model based on LWE-based encryption and a lattice-based chameleon hash function. To improve practicality, Cai et al. [24] introduced a more efficient scheme based on the Ring-SIS assumption, using rejection sampling to reduce signature size while maintaining security. More recently, Zhang et al. [25] proposed an SDVS construction combining

SIS and LWE assumptions, further contributing to the development of quantum-secure SDVS protocols. These lattice-based schemes incur key and signature sizes with complexity $\mathcal{O}(\lambda^2)$ [22, 23, 24, 25], which may limit their suitability for resource-constrained environments.

In the isogeny-based setting, Sun et al.[26] proposed an SDVS scheme built on the Supersingular Isogeny Diffie–Hellman (SIDH) protocol [15]. This construction was rendered insecure by the landmark attacks against SIDH [16, 17, 18] in 2022. In 2025, Khuc et al. [12] proposed an SDVS scheme based on the CSIDH group action [19], which is still believed to be secure.

1.2 Our Contribution

We present CSI-SDVS, an efficient post-quantum SDVS scheme that also uses the CSIDH group action. Under the group action decisional Diffie-Hellman assumption (GA-DDH), the scheme satisfies all security properties required by an SDVS scheme, while offering very compact key and signature sizes and efficient signing and verification. We also prove that in the algebraic group action model (AGAM), the security of our scheme can be reduced to a computational (rather than decisional) assumption.

Outline. The next section recalls definition of an SDVS scheme and its security requirements. We then review the mathematical preliminaries and the group action-based hardness assumptions underlying our construction. In Section 3, we present our proposed scheme, CSI-SDVS. We establish the security of our construction in the random oracle model (ROM) in Section 4, under the GA-DDH assumption. We then prove reductions to GAIP in Section 5. Finally, in Section 6 we analyze the efficiency of the construction.

2 Background

Notation. We denote the security parameter by λ. For a finite set $\mathbb{X}$, the notation $x \xleftarrow{\$} \mathbb{X}$ indicates that x is sampled uniformly at random from $\mathbb{X}$. A (deterministic or randomized) algorithm A that outputs a value x on input y is denoted by $x \leftarrow A(y)$. $\mathcal{B}^{\mathcal{A}}$ denotes that algorithm $\mathcal{B}$ has oracle access to algorithm $\mathcal{A}$ during its execution. A function $\mathsf{negl} : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is said to be *negligible* if for every positive polynomial $p(\cdot)$, there exists a threshold $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$, it holds that $\mathsf{negl}(n) < \frac{1}{p(n)}$. The concatenation of two bit strings s_1 and s_2 is written as $s_1 \,||\, s_2$. For any positive integer n, we define $\mathbb{Z}_n := \mathbb{Z}/n\mathbb{Z}$.

2.1 Strong Designated Verifier Signatures (SDVS)

An SDVS scheme involves two parties: a *signer* S and a designated *verifier* V, who have secret key/public key pairs $(\mathsf{sk}_S, \mathsf{pk}_S)$ and $(\mathsf{sk}_V, \mathsf{pk}_V)$, respectively. The signer S generates a signature σ on a message m using its secret key and the public key of the designated verifier. The resulting signature σ can be verified only by V using its secret key sk_V, ensuring that no third party can verify the signature or be convinced of its validity. The designated verifier is equipped with a simulation

algorithm Simul that can generate a simulated signature σ' on any message m' using sk_V and the signer's public key—the existence of the simulation algorithm prevents the designated verifier from convincing any third party of the validity of a signature, even if the verifier reveals their secret key.

The formal syntax of an SDVS scheme is presented below, following the framework introduced in [26].

Definition 1 (SDVS Scheme)**.** A strong designated verifier signature scheme is defined by the following six polynomial-time algorithms:

$\mathsf{Setup}(1^\lambda) \xrightarrow{\$} \mathsf{pp}$**:** A PPT algorithm that, on inputs 1^λ, outputs the public parameters pp.

$\mathsf{SKeyGen}(\mathsf{pp}) \xrightarrow{\$} (\mathsf{sk}_S, \mathsf{pk}_S)$**:** A PPT algorithm that, on input pp, generates a secret/public key pair $(\mathsf{sk}_S, \mathsf{pk}_S)$ for the signer S.

$\mathsf{VKeyGen}(\mathsf{pp}) \xrightarrow{\$} (\mathsf{sk}_V, \mathsf{pk}_V)$**:** A PPT algorithm that, on input pp, generates a secret/public key pair $(\mathsf{sk}_V, \mathsf{pk}_V)$ for the designated verifier V.

$\mathsf{Sign}(\mathsf{sk}_S, \mathsf{pk}_V, m) \xrightarrow{\$} \sigma$**:** A PPT (or DPT) algorithm that, given the signer's secret key sk_S, the public key pk_V of the designated verifier, and a message $m \in \mathcal{M}$, outputs a real designated verifier signature $\sigma \in \Sigma$.

$\mathsf{Verify}(\mathsf{sk}_V, \mathsf{pk}_S, m, \sigma) \to \{0,1\}$**:** A DPT algorithm that, given the designated verifier's secret key sk_V, the signer's public key pk_S, a message $m \in \mathcal{M}$, and a signature $\sigma \in \Sigma$, outputs a decision bit $b \in \{0,1\}$, where $b = 1$ indicates acceptance and $b = 0$ indicates rejection.

$\mathsf{Simul}(\mathsf{sk}_V, \mathsf{pk}_S, m) \xrightarrow{\$} \sigma'$**:** A PPT (or DPT) algorithm that, given the designated verifier's secret key sk_V, the signer's public key pk_S, and a message $m \in \mathcal{M}$, outputs a simulated designated verifier signature $\sigma' \in \Sigma$.

An SDVS scheme is required to satisfy the following notion of correctness. Let pp be the public parameters generated by $\mathsf{Setup}(1^\lambda)$, and let $(\mathsf{sk}_S, \mathsf{pk}_S)$ and $(\mathsf{sk}_V, \mathsf{pk}_V)$ be the key pairs generated by $\mathsf{SKeyGen}(\mathsf{pp})$ and $\mathsf{VKeyGen}(\mathsf{pp})$, respectively. The correctness of an SDVS scheme requires that, for any message m, the following conditions hold with probability 1:

$$\mathsf{Verify}(\mathsf{sk}_V, \mathsf{pk}_S, m, \mathsf{Sign}(\mathsf{sk}_S, \mathsf{pk}_V, m)) = 1,$$

for a real designated verifier signature, and

$$\mathsf{Verify}(\mathsf{sk}_V, \mathsf{pk}_S, m, \mathsf{Simul}(\mathsf{sk}_V, \mathsf{pk}_S, m)) = 1,$$

for a simulated designated verifier signature. These conditions ensure that honestly generated real and simulated designated verifier signatures are both valid and accepted by the verifier.

2.2 Security Model

An SDVS scheme must satisfy three security notions: Non-Transferability (NT), Strong Unforgeability under Chosen-Message Attacks (SUF-CMA), and Privacy of Signer's Identity (PSI). The formal security definitions corresponding to these properties are presented below.

Non-Transferability (NT). In an SDVS scheme, the notion of non-transferability (NT) ensures that the designated verifier cannot convince any third party of the authenticity of a valid signature. This is achieved by enabling the verifier to use the Simul algorithm to generate simulated signatures that are computationally indistinguishable from genuine ones produced by the signer. As a result, any conviction about the validity of a signature cannot be reliably transferred to others. The NT security of our SDVS construction is formalized as follows.

Definition 2 (NT Security)**.** A strong designated verifier signature (SDVS) scheme is said to satisfy non-transferability (NT) if, for any PPT adversary $\mathcal{A}$, its advantage

$$\mathsf{Adv}^{\mathsf{NT}}_{\mathsf{SDVS},\mathcal{A}}(\lambda) := \left| \mathbb{P}\left[\mathsf{Exp}^{\mathsf{NT}}_{\mathsf{SDVS},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2} \right|,$$

is negligible. Here, $\mathsf{Exp}^{\mathsf{NT}}_{\mathsf{SDVS},\mathcal{A}}(\lambda)$ denotes the NT experiment between a challenger $\mathcal{C}$ and the adversary $\mathcal{A}$, depicted in Fig. 1. The probability is taken over the randomness of $\mathcal{A}$ and the algorithms Sign and Simul.

$\mathsf{Exp}^{\mathsf{NT}}_{\mathsf{SDVS},\mathcal{A}}(1^\lambda)$

101 : $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
102 : $(\mathsf{sk}_S, \mathsf{pk}_S) \leftarrow \mathsf{SKeyGen}(\mathsf{pp})$
103 : $(\mathsf{sk}_V, \mathsf{pk}_V) \leftarrow \mathsf{VKeyGen}(\mathsf{pp})$
104 : $m^* \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathsf{Sign}}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}}}(\mathsf{pk}_S, \mathsf{pk}_V)$
105 : $\mathsf{ID} \xleftarrow{\$} \{S, V\}$
106 : **if** $\mathsf{ID} = S$
107 : $\quad \sigma^* \leftarrow \mathsf{Sign}(\mathsf{sk}_S, \mathsf{pk}_V, m^*)$
108 : **else**
109 : $\quad \sigma^* \leftarrow \mathsf{Simul}(\mathsf{sk}_V, \mathsf{pk}_S, m^*)$
110 : $\mathsf{ID}' \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathsf{Sign}}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}}}(\mathsf{pk}_S, \mathsf{pk}_V, m^*, \sigma^*)$
111 : **return** $[\mathsf{ID} = \mathsf{ID}']$

$\mathcal{O}_{\mathsf{Sign}}(m)$

201 : $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}_S, \mathsf{pk}_V, m)$
202 : **return** σ

$\mathcal{O}_{\mathsf{Simul}}(m)$

301 : $\sigma \leftarrow \mathsf{Simul}(\mathsf{sk}_V, \mathsf{pk}_S, m)$
302 : **return** σ

$\mathcal{O}_{\mathsf{Verify}}(m, \sigma)$

401 : **return** $\mathsf{Verify}(\mathsf{sk}_V, \mathsf{pk}_S, m, \sigma)$

Fig. 1: The Non-Transferability (NT) experiment.

Strong Unforgeability under Chosen-Message Attacks (SUF-CMA). This notion guarantees that no PPT adversary (without access to the secret key of either the signer or the designated verifier) can generate a valid designated verifier signature, even on messages that were previously queried via either the signing or simulating oracles. The formal definition is given below.

Definition 3 (SUF-CMA Security)**.** A strong designated verifier signature (SDVS) scheme is said to be strongly unforgeable under chosen message attacks (SUF-CMA) if, for any PPT adversary $\mathcal{A}$, its advantage, defined by

$$\mathsf{Adv}^{\mathsf{SUF\text{-}CMA}}_{\mathsf{SDVS},\mathcal{A}}(\lambda) = \mathbb{P}\left[\mathsf{Exp}^{\mathsf{SUF\text{-}CMA}}_{\mathsf{SDVS},\mathcal{A}}(\lambda) = 1\right],$$

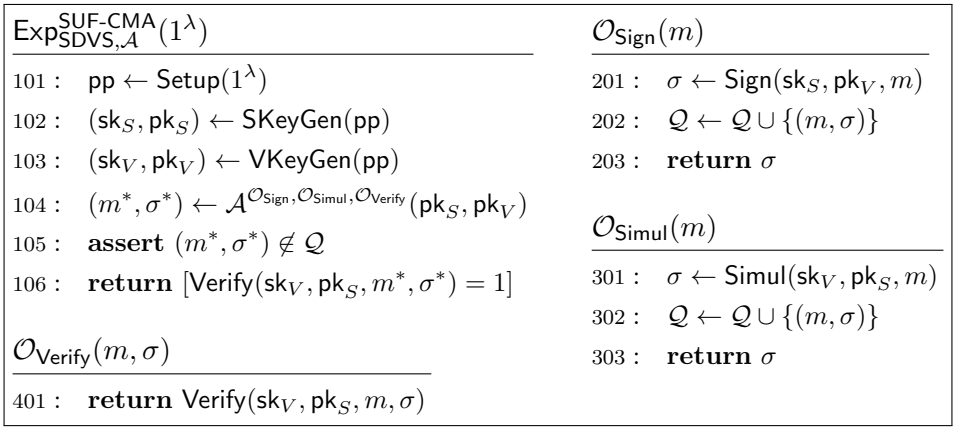


Fig. 2: The SUF-CMA experiment.

is negligible. Here, $\mathsf{Exp}^{\mathsf{SUF\text{-}CMA}}_{\mathsf{SDVS},\mathcal{A}}(\lambda)$ denotes the SUF-CMA experiment between a challenger $\mathcal{C}$ and the adversary $\mathcal{A}$, as illustrated in Fig. 2.
The probability is taken over the randomness of $\mathcal{A}$ and the algorithms Sign and Simul.

Privacy of Signer's Identity (PSI). The privacy of signer's identity (PSI) security notion ensures that whenever there are two or more possible signers for a given message, the identity of the signer remains hidden. Within an SDVS scheme, even when a signature and two valid signing key pairs are available, no adversary observing the communication between the signer and the designated verifier can determine—except with negligible probability—which of the two secret keys was used to generate the signature.

Definition 4 (PSI Security)**.** A strong designated verifier signature (SDVS) scheme is said to satisfy privacy of the signer's identity (PSI) if, for any PPT adversary $\mathcal{A}$, its advantage, defined by

$$\mathsf{Adv}^{\mathsf{PSI}}_{\mathsf{SDVS},\mathcal{A}}(\lambda) := \left| \mathbb{P}\left[\mathsf{Exp}^{\mathsf{PSI}}_{\mathsf{SDVS},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2} \right|,$$

is negligible. Here, $\mathsf{Exp}^{\mathsf{PSI}}_{\mathsf{SDVS},\mathcal{A}}(\lambda)$ denotes the PSI experiment between a challenger $\mathcal{C}$ and the adversary $\mathcal{A}$, as shown in Fig. 3. The probability is taken over the randomness of $\mathcal{A}$ and the algorithms Sign and Simul.

2.3 Cryptographic Group Actions

We recall the notion of cryptographic group actions following [29, Section 3]. Let $\mathcal{G}$ be a group and $\mathcal{X}$ be a set. A *group action* is a triple $(\mathcal{G}, \mathcal{X}, *)$, where $* : \mathcal{G} \times \mathcal{X} \to \mathcal{X}$ satisfies

1. Identity: $1_\mathcal{G} * X = X$ for all $X \in \mathcal{X}$.
2. Compatibility: $g * (h * X) = (gh) * X$ for all $g, h \in \mathcal{G}$ and all $X \in \mathcal{X}$.

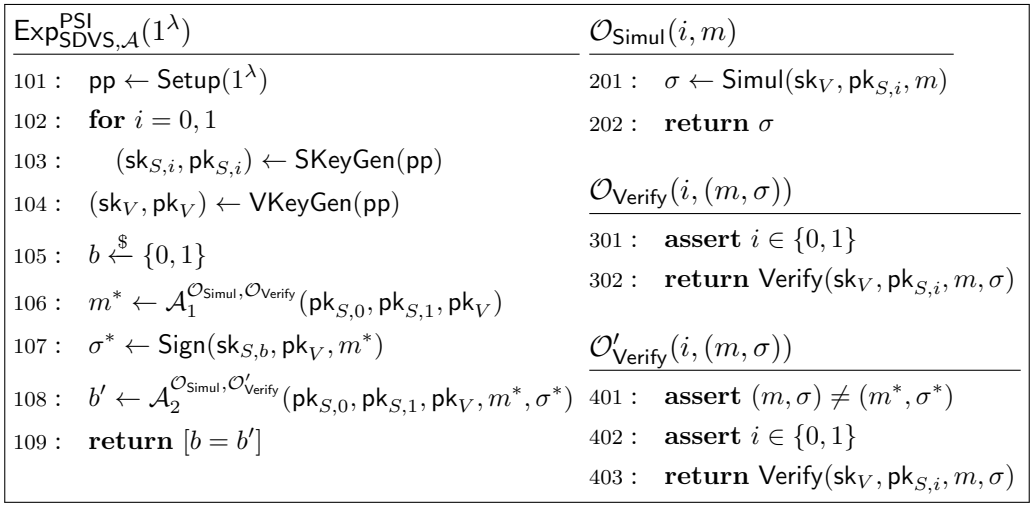


Fig. 3: The Privacy of Signer's Identity (PSI) experiment.

The action is said to be *free* if $g * X \neq X$ for every $g \in \mathcal{G} \setminus \{1_\mathcal{G}\}$ and every $X \in \mathcal{X}$, and *transitive* if for every $X_1, X_2 \in \mathcal{X}$, there exists $g \in \mathcal{G}$ such that $g * X_1 = X_2$. Of particular importance in cryptographic applications are *effective* group actions.

Definition 5 (Effective Group Action [29, Definition 3.4])**.** A group action $(\mathcal{G}, \mathcal{X}, *)$ is *effective* if:

1. $\mathcal{G}$ and $\mathcal{X}$ are finite.
2. There are PPT algorithms for the following problems in $\mathcal{G}$: membership testing, equality testing, sampling, the group operations, and inversion.
3. There is a PPT algorithm for membership testing in $\mathcal{X}$, and each element of $\mathcal{X}$ admits an efficiently-computable unique representaiton.
4. There is a distinguished element X_0 of $\mathcal{X}$—the *origin*—which is publicly-known.
5. There is a PPT algorithm for computing $g * X$ given any $g \in \mathcal{G}$ and $X \in \mathcal{X}$.

The protocols in this work derive their security from the difficulty of three problems: the *group action inverse problem* (GAIP), the *group action computational Diffie-Hellman* (GA-CDH) problem, and the *group action decisional Diffie-Hellman* (GA-DDH) problem. These problems are the group action analogues of the classical discrete logarithm problem, computational Diffie-Hellman problem, and decisional Diffie-Hellman problem, respectively.

Definition 6 (Group Action Inverse Problem (GAIP))**.** Let $(\mathcal{G}, \mathcal{X}, *)$ be an effective group action with origin X_0. Let $g \xleftarrow{\$} \mathcal{G}$, and $X_1 := g * X_0$. The GAIP problem is, given (X_0, X_1) to find $g' \in \mathcal{G}$ such that $g' * X_0 = X_1$.

Remark 1. We note that when $(\mathcal{G}, \mathcal{X}, *)$ is free, the solution to GAIP is unique.

Definition 7 (Group Action Computational Diffie-Hellman (GA-CDH) Problem)**.** Let $(\mathcal{G}, \mathcal{X}, *)$ be an effective group action with origin X_0. Let $g, h \xleftarrow{\$} \mathcal{G}$, $X_1 := g * X_0$, and $X_2 := h * X_0$. The GA-CDH problem is, given (X_1, X_2), to find $(gh) * X_0$.

Definition 8 (Group Action Decisional Diffie-Hellman (GA-DDH) Problem)**.** Let $(\mathcal{G}, \mathcal{X}, *)$ be an effective group action with origin X_0. Let $g, h \xleftarrow{\$} \mathcal{G}$, $X_1 := g * X_0$, and $X_2 := h * X_0$. Let X_3 be constructed as follows: sample $b \in \{0, 1\}$ and $z \in \mathcal{G}$ uniformly at random, and define

$$X_3 = \begin{cases} (gh) * X_0 & \text{if } b = 1 \\ z * X_0 & \text{if } b = 0 \end{cases}$$

The GA-DDH problem is, given (X_1, X_2, X_3), to determine the value of b.

An effective group action is said to be a *cyclic effective group action* (of known order) if there exists a group isomorphism $\iota \colon \mathbb{Z}_N \to \mathcal{G}$ for which both ι and ι^{-1} are efficiently computable. We recall the Algebraic Group Action Model (AGAM) for such group actions in Section 2.3.1; it forms the basis for the SUF-CMA and PSI security proofs presented in Section 5.

2.3.1 The Algebraic Group Action Model

The algebraic group action model (AGAM) was introduced by Duman et al. [13] as a generalization of the algebraic group model (AGM) [14] to the group action setting. In this model, we restrict our attention to *algebraic* adversaries, as defined in Definition 9.

Definition 9 (Algebraic Adversary)**.** Let $(\mathcal{G}, \mathcal{X}, *)$ be a cyclic effective group action, let Exp be a security experiment, and let $\mathcal{A}$ be an adversary for Exp. Suppose that $\mathcal{A}$ has received the set elements $(X_i)_{i=0}^n \in \mathcal{X}^{n+1}$ so far. We say that $\mathcal{A}$ is *algebraic* if, for every $Y \in \mathcal{X}$ appearing in an output of $\mathcal{A}$ or in an oracle query issued by $\mathcal{A}$, it additionally provides a pair $(a, i) \in \mathbb{Z}_N \times \{0, \ldots, n\}$ such that $Y = \iota(a) * X_i$.

Notation. For a set element Y output by $\mathcal{A}$ or used in an oracle query issued by $\mathcal{A}$, let

$$(a, i) \leftarrow \mathsf{GetRep}(Y)$$

denote the representation provided by $\mathcal{A}$ for that occurrence of Y; that is, $\iota(a) * X_i = Y$, where X_i is a set element previously received by $\mathcal{A}$ and $i \in \{0, \ldots, n\}$. If Y is neither output by $\mathcal{A}$ nor used in an oracle query issued by $\mathcal{A}$, we define $\mathsf{GetRep}(Y) := \perp$.

2.4 Elliptic Curves, Isogenies, and the CSIDH Group Action

Here we recall some fundamental properties of elliptic curves. For a comprehensive treatment of the topic, see [27, 28].

Let $k := \mathbb{F}_q$ denote a finite field, where $q := p^n$ for a prime $p > 3$ and a positive integer n. An *elliptic curve* E over k is a smooth, projective, genus-1 curve defined over k with a distinguished rational point $\mathcal{O}_E$ serving as the identity element for the group structure on E. For a positive integer ℓ, the *ℓ-torsion subgroup* of E is

defined as $E[\ell] := \{P \in E(\overline{k}) \mid [\ell]P = \mathcal{O}_E\}$. An elliptic curve is called *supersingular* if its group of p-torsion points is trivial over $\overline{\mathbb{F}}_p$, i.e., $E[p] = \{\mathcal{O}_E\}$. In particular, if $E/\mathbb{F}_p$ is supersingular, then $\#E(\mathbb{F}_p) = p + 1$.

An *isogeny* between elliptic curves is a surjective morphism that preserves the group structure. Two curves E_1 and E_2 defined over $\mathbb{F}_q$ are said to be *isogenous* if there exists an isogeny between them. For an isogeny $\phi : E \to E'$, its *degree*, denoted $\deg(\phi)$, is the degree of the function field extension $[k(E) : \phi^*(k(E'))]$, where ϕ^* is the pullback of ϕ defined as $\phi^* : k(E') \to k(E)$, with $\phi^*(f) := f \circ \phi$ for $f \in k(E')$. We call ϕ *separable* if the extension $k(E)/\phi^*(k(E'))$ is separable, a property ensuring that its degree exactly equals the size of its kernel, $\deg(\phi) = \#\ker(\phi)$. Consequently, for any finite subgroup $G \subset E(\mathbb{F}_q)$, there exists a unique separable isogeny (up to isomorphism) $\phi : E \to E' := E/G$ such that $\ker(\phi) = G$.

An *endomorphism* is an isogeny from an elliptic curve E to itself. The set of all endomorphisms of E, along with the zero map, forms a ring known as the *endomorphism ring* of E, denoted $\mathrm{End}(E)$. If E is a supersingular elliptic curve defined over $\mathbb{F}_p$, the ring of $\mathbb{F}_p$-rational endomorphisms, denoted by $\mathrm{End}_p(E)$, is isomorphic to an order in an imaginary quadratic field. Following [19], we choose parameters such that $\mathrm{End}_p(E)$ is an order in $\mathbb{Q}(\sqrt{-p})$.

Let $\mathfrak{O} \subset \mathbb{Q}(\sqrt{-p})$ be an order. The *ideal class group* $\mathcal{C}\ell(\mathfrak{O})$ is defined as the quotient $\mathcal{I}_\mathfrak{O}/\mathcal{P}_\mathfrak{O}$, where $\mathcal{I}_\mathfrak{O}$ is the group of invertible fractional ideals of $\mathfrak{O}$ and $\mathcal{P}_\mathfrak{O}$ is the subgroup of principal fractional ideals. The class group $\mathcal{C}\ell(\mathfrak{O})$ acts on the set $\mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O})$ of $\mathbb{F}_p$-isomorphism classes of supersingular elliptic curves defined over $\mathbb{F}_p$ with endomorphism ring isomorphic to $\mathfrak{O}$ as follows. Let $[\mathfrak{a}] \in \mathcal{C}\ell(\mathfrak{O})$ be an ideal class, and suppose without loss of generality that $\mathfrak{a}$ is integral. Let E be an elliptic curve with $\mathrm{End}(E) \cong \mathfrak{O}$. Through this isomorphism we can view the elements of $\mathfrak{a}$ as endomorphisms of E, and hence define the subgroup

$$S_\mathfrak{a} := \bigcap_{\alpha \in \mathfrak{a}} \ker(\alpha).$$

This $S_\mathfrak{a}$ is a finite subgroup of E, and hence it is the kernel of a unique (up to $\mathbb{F}_p$-isomorphism) isogeny $\psi_\mathfrak{a}$. We define the action $* : \mathcal{C}\ell(\mathfrak{O}) \times \mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O}) \to \mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O})$ by

$$[\mathfrak{a}] * E := \psi_\mathfrak{a}(E).$$

This action is both free and transitive. Moreover, recent work [20] demonstrates that this group action is effective[1], and the relevant computational problems (GAIP, GA-CDH, and GA-DDH) are generally believed to be quantum-safe. We call this the Commutative Supersingular Isogeny Diffie-Hellman (CSIDH) group action, after the paper in which it first appeared [19] in the cryptographic context.

3 Commutative Supersingular Isogeny Strong Designated Verifier Signatures: CSI-SDVS

We present our commutative supersingular isogeny-based strong designated verifier signature scheme, denoted by CSI-SDVS, built over a CSIDH-style group action. In

[1]Earlier works using this group action do not require the group action to be effective; for many protocols, the weaker notion of *restricted* effective group action [29, Definition 3.8] is sufficient.

Fig. 4, we introduce the CSI-SDVS. Our scheme's public parameter setup follows that of CSIDH [19], which we include here for completeness. A trusted party executes the setup procedure to generate the public parameters pp as follows. First, it selects a large prime $p = 4\ell_1\ell_2 \dots \ell_n - 1$, where the ℓ_i are small, distinct, odd primes. CSIDH-512 takes $n = 74$, $\ell_1, \dots, \ell_{73}$ to be the first 73 odd primes, and $\ell_{74} = 587$. Next, it fixes a base curve $E : y^2 = x^3 + x$ defined over $\mathbb{F}_p$, such that $E \in \mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O})$, and chooses a generator $\mathfrak{g}$ of the class group $\mathcal{G} := \mathcal{C}\ell(\mathfrak{O})$ with class number $N \approx p^{1/2}$, where $\mathfrak{O} := \mathbb{Z}[\sqrt{-p}]$. It then samples a cryptographically secure hash function $\mathcal{H} : \{0,1\}^* \to \{0,1\}^{2\lambda}$, where λ is a security parameter—in the security proof, we model this as a random oracle. Finally, it outputs the public parameters $\mathsf{pp} := (p, \mathfrak{g}, N, E, \mathcal{H})$.

The remaining algorithms of our protocol are described in Fig. 4.

CSI-SDVS.SKeyGen(pp)

101 : **parse** $(p, [\mathfrak{g}], N, E, \mathcal{H}) \leftarrow \mathsf{pp}$
102 : $\mathsf{sk}_S \xleftarrow{\$} \mathbb{Z}_N$
103 : $\mathsf{pk}_S \leftarrow [\mathfrak{g}^{\mathsf{sk}_S}] * E$
104 : **return** $(\mathsf{sk}_S, \mathsf{pk}_S)$

CSI-SDVS.VKeyGen(pp)

201 : **parse** $(p, [\mathfrak{g}], N, E, \mathcal{H}) \leftarrow \mathsf{pp}$
202 : $\mathsf{sk}_V \xleftarrow{\$} \mathbb{Z}_N$
203 : $\mathsf{pk}_V \leftarrow [\mathfrak{g}^{\mathsf{sk}_V}] * E$
204 : **return** $(\mathsf{sk}_V, \mathsf{pk}_V)$

CSI-SDVS.Sign($\mathsf{sk}_S, \mathsf{pk}_V, m$)

301 : $z \xleftarrow{\$} \mathbb{Z}_N$
302 : $Y \leftarrow [\mathfrak{g}^{z+\mathsf{sk}_S}] * \mathsf{pk}_V$
303 : $h \leftarrow \mathcal{H}(Y \,\|\, m)$
304 : **return** $\sigma := (h, z)$

CSI-SDVS.Simul($\mathsf{sk}_V, \mathsf{pk}_S, m$)

401 : $z \xleftarrow{\$} \mathbb{Z}_N$
402 : $Y \leftarrow [\mathfrak{g}^{z+\mathsf{sk}_V}] * \mathsf{pk}_S$
403 : $h \leftarrow \mathcal{H}(Y \,\|\, m)$
404 : **return** $\sigma := (h, z)$

CSI-SDVS.Verify($\mathsf{sk}_V, \mathsf{pk}_S, m, \sigma$)

501 : **parse** $(h, z) \leftarrow \sigma$
502 : **return** $[h = \mathcal{H}([\mathfrak{g}^{z+\mathsf{sk}_V}] * \mathsf{pk}_S \,\|\, m)]$

CSI-SDVS.Verify′($\mathsf{sk}_S, \mathsf{pk}_V, m, \sigma$)

601 : **parse** $(h, z) \leftarrow \sigma$
602 : **return** $[h = \mathcal{H}([\mathfrak{g}^{z+\mathsf{sk}_S}] * \mathsf{pk}_V \,\|\, m)]$

Fig. 4: CSI-SDVS: Commutative Supersingular Isogeny Strong Designated Verifier Signatures.

The correctness of our scheme follows directly from the commutativity of the underlying class group action, since

$$[\mathfrak{g}^{z+\mathsf{sk}_V}] * \mathsf{pk}_S = [\mathfrak{g}^{z+\mathsf{sk}_S}] * \mathsf{pk}_V$$

for any $\mathsf{sk}_S, \mathsf{sk}_V \in \mathbb{Z}_N$.

Although the correctness definition only requires that the verifier be able to verify signatures from the signer, we note that the signer can verify by checking

$$h \stackrel{?}{=} \mathcal{H}\big([\mathfrak{g}^{z+\mathsf{sk}_S}] * \mathsf{pk}_V \,\|\, m\big).$$

We denote this "signer-side verification" as Verify', and we include it in Fig. 4 to simplify notation in later proofs.

4 Security Proofs

In this section, we establish the security properties of the proposed scheme, CSI-SDVS. We first prove that that is satisfies NT security unconditionally, and we then establish its SUF-CMA and PSI security properties by reducing them to the GA-DDH assumption in the ROM. In the next section, we show that CSI-SDVS achieves SUF-CMA security and PSI under a computational (rather than decisional) assumption, in the algebraic group action model (AGAM).

Theorem 1. *For the scheme* CSI-SDVS*, for all public parameters* pp*, valid signer and designated verifier key pairs* $(\mathsf{sk}_S, \mathsf{pk}_S)$ *and* $(\mathsf{sk}_V, \mathsf{pk}_V)$*, and every message* $m \in \mathcal{M}$*, the outputs of* $\mathsf{CSI\text{-}SDVS.Sign}(\mathsf{sk}_S, \mathsf{pk}_V, m)$ *and* $\mathsf{CSI\text{-}SDVS.Simul}(\mathsf{sk}_V, \mathsf{pk}_S, m)$ *are identically distributed.*

Proof. Outputs of $\mathsf{CSI\text{-}SDVS.Sign}(\mathsf{sk}_S, \mathsf{pk}_V, m)$ and $\mathsf{CSI\text{-}SDVS.Simul}(\mathsf{sk}_V, \mathsf{pk}_S, m)$ take the form (h, z), where $h \in \{0,1\}^{2\lambda}$ and $z \in \mathbb{Z}_N$. Note first that the distribution of the z component of both algorithms is identical: uniformly random in $\mathbb{Z}_N$. For a fixed value of z, the hash value computed in $\mathsf{CSI\text{-}SDVS.Sign}(\mathsf{sk}_S, \mathsf{pk}_V, m)$ is

$$h = \mathcal{H}\big([\mathfrak{g}^{z+\mathsf{sk}_S}] * \mathsf{pk}_V \,\|\, m\big) = \mathcal{H}\big([\mathfrak{g}^{z+\mathsf{sk}_S+\mathsf{sk}_V}] * E \,\|\, m\big)$$

while in $\mathsf{CSI\text{-}SDVS.Simul}(\mathsf{sk}_V, \mathsf{pk}_S, m)$ it is

$$h = \mathcal{H}\big([\mathfrak{g}^{z+\mathsf{sk}_V}] * \mathsf{pk}_S \,\|\, m\big) = \mathcal{H}\big([\mathfrak{g}^{z+\mathsf{sk}_V+\mathsf{sk}_S}] * E \,\|\, m\big).$$

We see that the hash values are the same in both cases. Thus, the output distributions of the two algorithms (with a fixed message and fixed signer and verifier keys) are identical. □

Corollary 2. *The SDVS scheme* CSI-SDVS *is NT secure.*

Theorem 3. *Under the GA-DDH assumption (Definition 8), the proposed scheme,* CSI-SDVS*, is SUF-CMA secure (as per Definition 3) in the random oracle model.*

The proof of Theorem 3 follows by a standard sequence of games argument. At a high level, the games proceed as follows:

Game_0: The SUF-CMA game, as illustrated in Fig. 2.

Game_1: The challenger chooses uniformly-random curve R, and whenever an oracle would compute $[\mathfrak{g}^{z+\mathsf{sk}_V}] * \mathsf{pk}_S$ or $[\mathfrak{g}^{z+\mathsf{sk}_S}] * \mathsf{pk}_V$ for some $z \in \mathbb{Z}_N$, it instead computes $[\mathfrak{g}^z] * R$. Under the GA-DDH assumption, any adversary's advantage changes only negligibly when moving from Game_0 to Game_1.

Game_2: The game aborts and $\mathcal{A}$ loses if two random oracle queries on different inputs return the same output. For any PPT adversary $\mathcal{A}$, this happens with negligible probability by the birthday bound.

Game_3: The game aborts and $\mathcal{A}$ loses if, when $\mathcal{A}$ returns the forgery $(m^*, \sigma^* = (h^*, z^*))$, the random oracle has not been queried at $([\mathfrak{g}^{z^*}] * R \,\|\, m^*)$. In Game_2, such a forgery would pass verification only with negligible probability, so any adversary's advantage changes only negligibly when moving from Game_2 to Game_3.

Game_4: The game no longer makes consistency checks between the oracle calls. We enumerate the ways that this could change the behaviour of the game, and show that for PPT adversaries, they all happen with negligible probability.

Finally, in Game_4, the adversary must make a query of the form $\mathcal{H}([\mathfrak{g}^{z^*}] * R \,\|\, m^*)$, where $(m^*, \sigma^* = (h^*, z^*))$ is its forgery. In Game_4, the adversary does not receive any information about R—either as input from the challenger or in the oracle responses—and so can make such a query with only negligible probability.

The first four games are depicted in Fig. 5.

$\mathsf{Game}_0(1^\lambda)$ $\mathsf{Game}_1(1^\lambda)$ $\mathsf{Game}_2(1^\lambda)$ $\mathsf{Game}_3(1^\lambda)$

101 : $\mathcal{Q} \leftarrow \varnothing, L \leftarrow [\,]$
102 : $(\mathsf{sk}_S, \mathsf{pk}_S) \leftarrow \mathsf{CSI\text{-}SDVS.SKeyGen}(\mathsf{pp})$
103 : $(\mathsf{sk}_V, \mathsf{pk}_V) \leftarrow \mathsf{CSI\text{-}SDVS.VKeyGen}(\mathsf{pp})$
104 : $r \xleftarrow{\$} \mathbb{Z}_N$
105 : $R \leftarrow [\mathfrak{g}^r] * E$
106 : $\mathsf{oracles} \leftarrow (\mathcal{H}, \mathcal{O}_{\mathsf{Sign}}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}})$
107 : $(m^*, \sigma^* = (h^*, z^*)) \leftarrow \mathcal{A}^{\mathsf{oracles}}(\mathsf{pk}_S, \mathsf{pk}_V)$
108 : **assert** $\nexists x \neq x'$ s.t. $L[x] = L[x'] \neq \bot$
109 : **assert** $L[[\mathfrak{g}^{z^*}] * R \,\|\, m^*] \neq \bot$
110 : **assert** $(m^*, \sigma^*) \notin \mathcal{Q}$
111 : **if** $h^* = \mathcal{H}([\mathfrak{g}^{z^*}] * R \,\|\, m^*)$
112 : **return** 1
113 : **return** 0
114 : **return** $\mathsf{CSI\text{-}SDVS.Verify}(\mathsf{sk}_V, \mathsf{pk}_S, m^*, \sigma^*)$

$\mathcal{O}_{\mathsf{Sign}}(m)$

201 : $z \xleftarrow{\$} \mathbb{Z}_N$
202 : $Y \leftarrow [\mathfrak{g}^{z+\mathsf{sk}_S}] * \mathsf{pk}_V$
203 : $Y \leftarrow [\mathfrak{g}^z] * R$
204 : $h \leftarrow \mathcal{H}(Y \,\|\, m)$
205 : $\sigma \leftarrow (h, z)$
206 : $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(m, \sigma)\}$
207 : **return** σ

$\mathcal{O}_{\mathsf{Simul}}(m)$

301 : $z \xleftarrow{\$} \mathbb{Z}_N$
302 : $Y \leftarrow [\mathfrak{g}^{z+\mathsf{sk}_V}] * \mathsf{pk}_S$
303 : $Y \leftarrow [\mathfrak{g}^z] * R$
304 : $h \leftarrow \mathcal{H}(Y \,\|\, m)$
305 : $\sigma \leftarrow (h, z)$
306 : $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(m, \sigma)\}$
307 : **return** σ

$\mathcal{H}(x)$

401 : **if** $L[x] = \bot$
402 : $L[x] \xleftarrow{\$} \{0,1\}^{2\lambda}$
403 : **return** $L[x]$

$\mathcal{O}_{\mathsf{Verify}}(m, \sigma)$

501 : **parse** $(h, z) \leftarrow \sigma$
502 : $h' \leftarrow \mathcal{H}([\mathfrak{g}^{z+\mathsf{sk}_V}] * \mathsf{pk}_S \,\|\, m)$
503 : $h' \leftarrow \mathcal{H}([\mathfrak{g}^z] * R \,\|\, m)$
504 : **return** $[h = h']$

Fig. 5: Games Game_0–Game_3 used in the proof of Theorem 3.

Claim. For any adversary $\mathcal{A}$ running in time T, there exists an algorithm $\mathcal{B}$ running in time $T + \mathsf{poly}(\lambda)$ such that

$$\mathsf{Adv}^{\mathsf{Game}_0}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{Game}_1}(\mathcal{A}) + \mathsf{Adv}^{\text{GA-DDH}}(\mathcal{B}).$$

Proof. Consider the GA-DDH adversary $\mathcal{B}$ defined in Fig. 6.

$\mathcal{B}^{\mathcal{A}}(X_1, X_2, X_3)$

101 : $\mathcal{Q} \leftarrow \varnothing, L \leftarrow [\,]$
102 : $(\mathsf{pk}_S, \mathsf{pk}_V, R) \leftarrow (X_1, X_2, X_3)$
103 : $\mathsf{oracles} \leftarrow (\mathcal{H}, \mathcal{O}_{\mathsf{Sign}}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}})$
104 : $(m^*, \sigma^* = (h^*, z^*)) \leftarrow \mathcal{A}^{\mathsf{oracles}}(\mathsf{pk}_S, \mathsf{pk}_V)$
105 : **assert** $(m^*, \sigma^*) \notin \mathcal{Q}$
106 : **if** $h^* = \mathcal{H}([\mathfrak{g}^{z^*}] * R \,\|\, m^*)$
107 : **return** 1
108 : **return** 0

$\mathcal{H}(x)$

401 : **if** $L[x] = \perp$
402 : $L[x] \xleftarrow{\$} \{0,1\}^{2\lambda}$
403 : **return** $L[x]$

$\mathcal{O}_{\mathsf{Verify}}(m, \sigma)$

501 : **parse** $(h, z) \leftarrow \sigma$
502 : $h' \leftarrow \mathcal{H}([\mathfrak{g}^z] * R \,\|\, m)$
503 : **return** $[h = h']$

$\mathcal{O}_{\mathsf{Sign}}(m)$

201 : $z \xleftarrow{\$} \mathbb{Z}_N$
202 : $Y \leftarrow [\mathfrak{g}^z] * R$
203 : $h \leftarrow \mathcal{H}(Y \,\|\, m)$
204 : $\sigma \leftarrow (h, z)$
205 : $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(m, \sigma)\}$
206 : **return** σ

$\mathcal{O}_{\mathsf{Simul}}(m)$

301 : $z \xleftarrow{\$} \mathbb{Z}_N$
302 : $Y \leftarrow [\mathfrak{g}^z] * R$
303 : $h \leftarrow \mathcal{H}(Y \,\|\, m)$
304 : $\sigma \leftarrow (h, z)$
305 : $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(m, \sigma)\}$
306 : **return** σ

Fig. 6: A GA-DDH distinguisher $\mathcal{B}$. Its input (X_1, X_2, X_3) is drawn uniformly at random from either the group action Diffie-Hellman distribution on $(\mathcal{G}, \mathcal{X}, *)$, or the uniform distribution on $\mathcal{X}^3$, each with probability $\frac{1}{2}$.

Observe that when (X_1, X_2, X_3) is sampled from the Diffie-Hellman distribution on $(\mathcal{G}, \mathcal{X}, *)$, $\mathcal{B}$ is honestly playing Game_0 with $\mathcal{A}$, and when (X_1, X_2, X_3) is sampled uniformly from $\mathcal{X}^3$, $\mathcal{B}$ is honestly playing Game_1 with $\mathcal{A}$. Thus,

$$|\mathsf{Adv}^{\mathsf{Game}_0}(\mathcal{A}) - \mathsf{Adv}^{\mathsf{Game}_1}(\mathcal{A})| \leq \mathsf{Adv}^{\text{GA-DDH}}(\mathcal{B}). \qquad \square$$

Claim. For any algorithm $\mathcal{A}$ that makes at most q total queries to $\mathcal{H}, \mathcal{O}_{\mathsf{Sign}}, \mathcal{O}_{\mathsf{Simul}}$, and $\mathcal{O}_{\mathsf{Verify}}$, we have

$$\mathsf{Adv}^{\mathsf{Game}_1}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{Game}_2}(\mathcal{A}) + \frac{q^2}{2^{2\lambda}}.$$

Proof. The only difference between Game_1 and Game_2 is on line 108, and so the difference in advantages is at most the probability that the assertion there fails. The assertion fails if and only if there have been distinct queries $x \neq x'$ to $\mathcal{H}$ (either explicit by the adversary, or implicit through queries to other oracles) such that $\mathcal{H}(x) = \mathcal{H}(x')$. Since $\mathcal{H}$ is queried at at most q inputs, and each output of $\mathcal{H}$ is chosen independently and uniformly at random, this probability is bounded above by $\frac{q^2}{2^{2\lambda}}$ by the birthday bound. The claim follows. $\square$

Claim. For any algorithm $\mathcal{A}$, we have $\mathsf{Adv}^{\mathsf{Game}_2}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{Game}_3}(\mathcal{A}) + \frac{1}{2^{2\lambda}}$.

Proof. The only difference between Game_2 and Game_3 occurs on line 109. The differences in advantages between the games is bounded by the probability that $\mathcal{A}$ wins the game conditioned on the assertion failing; that is,

$$|\mathsf{Adv}^{\mathsf{Game}_2}(\mathcal{A}) - \mathsf{Adv}^{\mathsf{Game}_3}(\mathcal{A})| \leq \mathbb{P}[h^* = \mathcal{H}([\mathfrak{g}^{z^*}] * R \,\|\, m^*) \mid L[[g^{z^*}] * R \,\|\, m^*] = \perp].$$

Since $L[[g^{z^*}] * R \,\|\, m^*] = \perp$, we know that $\mathcal{H}$ has not been queried (directly or by a query to $\mathcal{O}_{\mathsf{Sign}}$, $\mathcal{O}_{\mathsf{Simul}}$, or $\mathcal{O}_{\mathsf{Verify}}$) at $([g^{z^*}] * R \,\|\, m^*)$. Thus, when the challenger verifies (m^*, σ^*) on line 111, the value of $\mathcal{H}(([g^{z^*}] * R \,\|\, m^*)$ will be chosen uniformly at random on line 402. Then, we have $h^* = \mathcal{H}([\mathfrak{g}^{z^*}] * R \,\|\, m^*)$ with probability $\frac{1}{2^{2\lambda}}$, as required. □

Finally, before proceeding to the proof of Theorem 3, we consider the Game_4, depicted in Fig. 7.

$\mathsf{Game}_4(1^\lambda)$	$\mathcal{O}_{\mathsf{Sign}}(m)$, $\mathcal{O}_{\mathsf{Simul}}(m)$
101 : $\mathcal{Q} \leftarrow \varnothing, L \leftarrow [\,]$	201 : $z \xleftarrow{\$} \mathbb{Z}_N$
102 : $(\mathsf{sk}_S, \mathsf{pk}_S) \leftarrow \mathsf{CSI\text{-}SDVS.SKeyGen}(\mathsf{pp})$	202 : $h \xleftarrow{\$} \{0,1\}^{2\lambda}$
103 : $(\mathsf{sk}_V, \mathsf{pk}_V) \leftarrow \mathsf{CSI\text{-}SDVS.VKeyGen}(\mathsf{pp})$	203 : $\sigma \leftarrow (h, z)$
104 : $r \xleftarrow{\$} \mathbb{Z}_N$	204 : $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(m, \sigma)\}$
105 : $R \leftarrow [\mathfrak{g}^r] * E$	205 : **return** σ
106 : $\mathsf{oracles} \leftarrow (\mathcal{H}, \mathcal{O}_{\mathsf{Sign}}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}})$	$\mathcal{H}(x)$
107 : $(m^*, \sigma^* = (h^*, z^*)) \leftarrow \mathcal{A}^{\mathsf{oracles}}(\mathsf{pk}_S, \mathsf{pk}_V)$	301 : **if** $L[x] = \perp$
108 : **assert** $L[[\mathfrak{g}^{z^*}] * R \,\|\, m^*] \neq \perp$	302 : $L[x] \xleftarrow{\$} \{0,1\}^{2\lambda}$
109 : **assert** $\nexists x \neq x'$ s.t. $L[x] = L[x'] \neq \perp$	303 : **return** $L[x]$
110 : **assert** $(m^*, \sigma^*) \notin \mathcal{Q}$	$\mathcal{O}_{\mathsf{Verify}}(m, \sigma)$
111 : **if** $h^* = \mathcal{H}([\mathfrak{g}^{z^*}] * R \,\|\, m^*)$	401 : **return** $[(m, \sigma) \in \mathcal{Q}]$
112 : **return** 1	
113 : **return** 0	

Fig. 7: The Game_4 used in the proof of Theorem 3.

Claim. For any algorithm $\mathcal{A}$ that makes at most $q_{\mathcal{H}}$ (respectively, $q_{\mathsf{Sign}}, q_{\mathsf{Simul}}, q_{\mathsf{Verify}}$) queries to $\mathcal{H}$ (respectively $\mathcal{O}_{\mathsf{Sign}}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}}$), we have

$$\mathsf{Adv}^{\mathsf{Game}_3}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{Game}_4}(\mathcal{A}) + \frac{(q_{\mathsf{Sign}} + q_{\mathsf{Simul}} + 2q_{\mathcal{H}})(q_{\mathsf{Sign}} + q_{\mathsf{Simul}}) + q_{\mathcal{H}} q_{\mathsf{Verify}}}{N} + \frac{q_{\mathcal{H}} q_{\mathsf{Verify}}}{2^{2\lambda}} + \frac{q_{\mathsf{Verify}}(q_{\mathsf{Sign}} + q_{\mathsf{Simul}})}{2^{2\lambda} N}.$$

Proof. From the adversary's perspective, Game_3 is identical to Game_4 unless the result of an oracle query is contradicted by a later oracle query. In particular, the games are identical unless one of the following events happens in Game_4:

$\mathcal{O}_{\mathsf{Sign}}/\mathcal{O}_{\mathsf{Simul}}$ contradicts $\mathcal{O}_{\mathsf{Sign}}/\mathcal{O}_{\mathsf{Simul}}$: Distinct queries to either $\mathcal{O}_{\mathsf{Sign}}$ or $\mathcal{O}_{\mathsf{Simul}}$ with the same message m have the same value of z and different values of h.

Since each call to $\mathcal{O}_{\mathsf{Sign}}$ and $\mathcal{O}_{\mathsf{Simul}}$ generates an independent, uniformly random value of z, and there are at most $q_{\mathsf{Sign}} + q_{\mathsf{Simul}}$ total queries to these oracles, this event happens with probability at most $\frac{(q_{\mathsf{Sign}} + q_{\mathsf{Simul}})^2}{N}$ by the birthday bound.

$\mathcal{O}_{\mathsf{Verify}}$ **contradicts** $\mathcal{O}_{\mathsf{Sign}}/\mathcal{O}_{\mathsf{Simul}}$**:** $\mathcal{O}_{\mathsf{Verify}}$ rejects a signature generated by $\mathcal{O}_{\mathsf{Sign}}$ or $\mathcal{O}_{\mathsf{Simul}}$.

By construction, this never happens.

$\mathcal{H}$ **contradicts** $\mathcal{O}_{\mathsf{Sign}}/\mathcal{O}_{\mathsf{Simul}}$**:** The adversary queries $\mathcal{O}_{\mathsf{Sign}}$ or $\mathcal{O}_{\mathsf{Simul}}$ at m and it returns (h, z). Later, the adversary queries $\mathcal{H}$ at $([\mathfrak{g}^z] * R \,\|\, m)$ and it returns $h' \neq h$.

Up until such a query to $\mathcal{H}$ occurs, R is information-theoretically hidden from the adversary, and so such a query cannot be made except by random chance. There are N possible values of R, and the adversary makes at most $q_{\mathcal{H}}$ queries to $\mathcal{H}$ and at most $q_{\mathsf{Sign}} + q_{\mathsf{Simul}}$ queries to $\mathcal{O}_{\mathsf{Sign}}$ and $\mathcal{O}_{\mathsf{Simul}}$ in total. Thus, this event occurs with probability at most $\frac{q_H(q_{\mathsf{Sign}}+q_{\mathsf{Simul}})}{N}$.

$\mathcal{O}_{\mathsf{Sign}}/\mathcal{O}_{\mathsf{Simul}}$ **contradicts** $\mathcal{O}_{\mathsf{Verify}}$**:** The adversary queries $\mathcal{O}_{\mathsf{Verify}}$ at some $(m, \sigma = (h, z))$ and it is rejected; later, the adversary queries $\mathcal{O}_{\mathsf{Sign}}$ or $\mathcal{O}_{\mathsf{Simul}}$ at m and the oracle returns (h, z).

Since each call to $\mathcal{O}_{\mathsf{Sign}}$ and $\mathcal{O}_{\mathsf{Simul}}$ generates independent, uniformly random values of z and h, and there are at most $q_{\mathsf{Sign}} + q_{\mathsf{Simul}}$ total queries to $\mathcal{O}_{\mathsf{Sign}}$ and $\mathcal{O}_{\mathsf{Simul}}$ and q_{Verify} queries to $\mathcal{O}_{\mathsf{Verify}}$, this event happens with probability at most $\frac{q_{\mathsf{Verify}}(q_{\mathsf{Sign}}+q_{\mathsf{Simul}})}{2^{2\lambda}N}$.

$\mathcal{O}_{\mathsf{Verify}}$ **contradicts** $\mathcal{O}_{\mathsf{Verify}}$**:** The adversary queries $\mathcal{O}_{\mathsf{Verify}}$ at some (m, σ) twice and it is accepted once and rejected once.

This can only happen in the following way: (m, σ) is rejected because it is not in $\mathcal{Q}$, then a query to $\mathcal{O}_{\mathsf{Sign}}$ or $\mathcal{O}_{\mathsf{Simul}}$ adds (m, σ) to $\mathcal{Q}$, and then the adversary queries $\mathcal{O}_{\mathsf{Verify}}$ at (m, σ) again. This case is subsumed by the previous one.

$\mathcal{H}$ **contradicts** $\mathcal{O}_{\mathsf{Verify}}$**:** The adversary queries $\mathcal{O}_{\mathsf{Verify}}$ at a pair $(m, \sigma = (h, z)) \notin \mathcal{Q}$ (and so, it is rejected). Later, the adversary queries $\mathcal{H}([\mathfrak{g}^z] * R \,\|\, m)$ and it returns h.

Since $\mathcal{H}([\mathfrak{g}^z] * R \,\|\, m)$ is set uniformly at random on line 402, and the adversary makes at most $q_{\mathcal{H}}$ queries to $\mathcal{H}$ and q_{Verify} queries to $\mathcal{O}_{\mathsf{Verify}}$, this occurs with probability at most $\frac{q_{\mathcal{H}} q_{\mathsf{Verify}}}{2^{2\lambda}}$.

$\mathcal{O}_{\mathsf{Sign}}/\mathcal{O}_{\mathsf{Simul}}$ **contradicts** $\mathcal{H}$**:** The adversary queries $\mathcal{H}([\mathfrak{g}^z] * R \,\|\, m)$ and it returns h. Later, the adversary queries $\mathcal{O}_{\mathsf{Sign}}$ or $\mathcal{O}_{\mathsf{Simul}}$ at m and the oracle returns $\sigma = (h', z)$ with $h' \neq h$.

Since each call to $\mathcal{O}_{\mathsf{Sign}}$ and $\mathcal{O}_{\mathsf{Simul}}$ generates an independent, uniformly random value of z, and there are at most $q_{\mathcal{H}}$ queries to $\mathcal{H}$ and $q_{\mathsf{Sign}} + q_{\mathsf{Simul}}$ total queries to $\mathcal{O}_{\mathsf{Sign}}$ and $\mathcal{O}_{\mathsf{Simul}}$, this event happens with probability at most $\frac{q_{\mathcal{H}}(q_{\mathsf{Sign}}+q_{\mathsf{Simul}})}{N}$.

$\mathcal{O}_{\mathsf{Verify}}$ **contradicts** $\mathcal{H}$**:** The adversary queries $\mathcal{H}$ at some $([\mathfrak{g}^z] * R \,\|\, m)$ and it returns h. Later, the adversary queries $\mathcal{O}_{\mathsf{Verify}}$ at $(m, \sigma = (h, z))$ and the pair is rejected.

Up until such a query occurs, R is information-theoretically hidden from the adversary, and so such a query cannot be made except by random chance.

There are N possible values of R, and the adversary makes at most $q_{\mathcal{H}}$ queries to $\mathcal{H}$ and at most q_{Verify} queries to $\mathcal{O}_{\mathsf{Verify}}$. Thus, this event occurs with probability at most $\frac{q_H q_{\mathsf{Verify}}}{N}$.

$\mathcal{H}$ contradicts $\mathcal{H}$: The adversary queries $\mathcal{H}$ twice at the same input x and receives two different outputs. By construction, this never happens.

The result follows by summing the probabilities of these events. □

Claim. For any adversary $\mathcal{A}$ that makes at most $q_{\mathcal{H}}$ queries to $\mathcal{H}$, we have $\mathsf{Adv}^{\mathsf{Game}_4}(\mathcal{A}) \leq \frac{q_{\mathcal{H}}}{N}$.

Proof. In Game_4, each oracle's output is independent of the value of R. Thus, for any z^*, the probability that $\mathcal{H}$ has been queried at $([\mathfrak{g}^{z^*}] * R \,\|\, m)$ for some m is at most $\frac{q_H}{N}$. If no such query has been made, then $\mathcal{A}$ loses the game because of the assertion on line 109. The result follows. □

Proof (of Theorem 3). Altogether, for any adversary $\mathcal{A}$ running in time T and making at most $q_{\mathcal{H}}$ (respectively, $q_{\mathsf{Sign}}, q_{\mathsf{Simul}}, q_{\mathsf{Verify}}$) queries to $\mathcal{H}$ (respectively $\mathcal{O}_{\mathsf{Sign}}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}}$), there is an adversary $\mathcal{B}$ that runs in time $T + \mathsf{poly}(\lambda)$ such that

$$\mathsf{Adv}^{\mathsf{SUF\text{-}CMA}}_{\mathsf{CSI\text{-}SDVS}}(\mathcal{A}) \leq \mathsf{Adv}^{\mathrm{DDH}}(\mathcal{B}) + \frac{q^2 + q_{\mathcal{H}} q_{\mathsf{Verify}} + 1}{2^{2\lambda}} + \frac{q_{\mathsf{Verify}}(q_{\mathsf{Sign}} + q_{\mathsf{Simul}})}{2^{2\lambda} N} + \frac{(q_{\mathsf{Sign}} + q_{\mathsf{Simul}} + 2q_{\mathcal{H}})(q_{\mathsf{Sign}} + q_{\mathsf{Simul}}) + q_{\mathcal{H}} q_{\mathsf{Verify}}}{N},$$

where $q = q_{\mathcal{H}} + q_{\mathsf{Sign}} + q_{\mathsf{Simul}} + q_{\mathsf{Verify}}$. Thus, when $\mathcal{A}$ is PPT (so that $T = \mathsf{poly}(\lambda)$ and $q = \mathsf{poly}(\lambda)$), $\mathsf{Adv}^{\mathsf{SUF\text{-}CMA}}_{\mathsf{CSI\text{-}SDVS}}(\mathcal{A}) = \mathsf{negl}(\lambda)$ under the GA-DDH assumption, as required. □

Theorem 4. *Under the GA-DDH assumption (Definition 8), the proposed scheme,* CSI-SDVS*, is PSI secure (as per Definition 4) in the random oracle model.*

Like the proof of Theorem 3, for Theorem 4 we proceed by a sequence of games.

Game_0: The PSI game, as shown in Fig. 3.

Game_1: The challenger chooses uniformly-random curve R_0, and whenever an oracle would compute $[\mathfrak{g}^{z+\mathsf{sk}_V}] * \mathsf{pk}_{S,0}$ or $[\mathfrak{g}^{z+\mathsf{sk}_{S,0}}] * \mathsf{pk}_V$ for some $z \in \mathbb{Z}_N$, it instead computes $[\mathfrak{g}^z] * R_0$. Under the GA-DDH assumption, any adversary's advantage changes only negligibly when moving from Game_0 to Game_1.

Game_2: The challenger chooses uniformly-random curve R_1, and whenever an oracle would compute $[\mathfrak{g}^{z+\mathsf{sk}_V}] * \mathsf{pk}_{S,1}$ or $[\mathfrak{g}^{z+\mathsf{sk}_{S,1}}] * \mathsf{pk}_V$ for some $z \in \mathbb{Z}_N$, it instead computes $[\mathfrak{g}^z] * R_1$. Under the GA-DDH assumption, any adversary's advantage changes only negligibly when moving from Game_1 to Game_2.

Game_3: The game aborts and $\mathcal{A}$ loses if two random oracle queries on different inputs return the same output. For any PPT adversary $\mathcal{A}$, this happens with negligible probability by the birthday bound.

Game_4: The game no longer makes consistency checks between the oracle calls. We enumerate the ways that this could change the behaviour of the game, and show that for PPT adversaries, they all happen with negligible probability.

Finally, we show that no adversary can win Game_4 with probability different from $\frac{1}{2}$, completing the proof.

Games $\mathsf{Game}_0, \mathsf{Game}_1, \mathsf{Game}_2$, and Game_3 are depicted in Fig. 8. These game hopes closely resemble ones made in the proof of Theorem 3, and so we omit many of the details.

$\mathsf{Game}_0(1^\lambda)$ $\mathsf{Game}_1(1^\lambda)$ $\mathsf{Game}_2(1^\lambda)$ $\mathsf{Game}_3(1^\lambda)$

101 : $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
102 : **for** $i = 0, 1$
103 : $\quad (\mathsf{sk}_{S,i}, \mathsf{pk}_{S,i}) \leftarrow \mathsf{SKeyGen}(\mathsf{pp})$
104 : $(\mathsf{sk}_V, \mathsf{pk}_V) \leftarrow \mathsf{VKeyGen}(\mathsf{pp})$
105 : $b \xleftarrow{\$} \{0,1\}$
106 : $R_0 \leftarrow [\mathfrak{g}^{\mathsf{sk}_V}] * \mathsf{pk}_{S,0}$
107 : $R_1 \leftarrow [\mathfrak{g}^{\mathsf{sk}_V}] * \mathsf{pk}_{S,1}$
108 : $r_0 \xleftarrow{\$} \mathbb{Z}_N$
109 : $R_0 \leftarrow [\mathfrak{g}^{r_0}] * E$
110 : $r_1 \xleftarrow{\$} \mathbb{Z}_N$
111 : $R_1 \leftarrow [\mathfrak{g}^{r_1}] * E$
112 : $m^* \leftarrow \mathcal{A}_1^{\mathcal{H}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}}}(\mathsf{pk}_{S,0}, \mathsf{pk}_{S,1}, \mathsf{pk}_V)$
113 : $z^* \xleftarrow{\$} \mathbb{Z}_N$
114 : $h^* \leftarrow \mathcal{H}([\mathfrak{g}^{z^*}] * R_b \,\|\, m^*)$
115 : $\sigma^* \leftarrow (h^*, z^*)$
116 : **assert** $\nexists x \neq x'$ s.t. $L[x] = L[x'] \neq \perp$
117 : $b' \leftarrow \mathcal{A}_2^{\mathcal{H}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}'_{\mathsf{Verify}}}(\mathsf{pk}_{S,0}, \mathsf{pk}_{S,1}, \mathsf{pk}_V, m^*, \sigma^*)$
118 : **return** $[b = b']$

$\mathcal{O}_{\mathsf{Simul}}(i, m)$

201 : $z \xleftarrow{\$} \mathbb{Z}_N$
202 : $h \leftarrow \mathcal{H}([\mathfrak{g}^z] * R_i \,\|\, m)$
203 : $\sigma \leftarrow (h, z)$
204 : **return** σ

$\mathcal{H}(x)$

301 : **if** $L[x] = \perp$
302 : $\quad L[x] \xleftarrow{\$} \{0,1\}^{2\lambda}$
303 : **return** $L[x]$

$\mathcal{O}_{\mathsf{Verify}}(i, (m, \sigma))$

401 : **assert** $i \in \{0,1\}$
402 : **parse** $(h, z) \leftarrow \sigma$
403 : $h' \leftarrow \mathcal{H}([\mathfrak{g}^z] * R_i \,\|\, m)$
404 : **return** $[h = h']$

$\mathcal{O}'_{\mathsf{Verify}}(i, (m, \sigma))$

501 : **assert** $(m, \sigma) \neq (m^*, \sigma^*)$
502 : **assert** $i \in \{0,1\}$
503 : **parse** $(h, z) \leftarrow \sigma$
504 : $h' \leftarrow \mathcal{H}([\mathfrak{g}^z] * R_i \,\|\, m)$
505 : **return** $[h = h']$

Fig. 8: The games $\mathsf{Game}_0, \mathsf{Game}_1, \mathsf{Game}_2$, and Game_3 used in the proof of Theorem 4.

Claim. For any adversary $\mathcal{A}$ running in time T, there exists an algorithm $\mathcal{B}$ running in time $T + \mathsf{poly}(\lambda)$ such that

$$\mathsf{Adv}^{\mathsf{Game}_0}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{Game}_2}(\mathcal{A}) + 2\mathsf{Adv}^{\text{GA-DDH}}(\mathcal{B}).$$

Proof. Using an analogous GA-DDH distinguisher to the one depicted in Fig. 6, it is straightforward to verify that for any PPT adversary $\mathcal{A}$ running in time T, there are PPT adversaries $\mathcal{B}_1$ and $\mathcal{B}_2$, each running in time $T + \mathsf{poly}(\lambda)$, such that

$$\mathsf{Adv}^{\mathsf{Game}_0}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{Game}_1}(\mathcal{A}) + \mathsf{Adv}^{\text{GA-DDH}}(\mathcal{B}_1)$$
$$\mathsf{Adv}^{\mathsf{Game}_1}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{Game}_2}(\mathcal{A}) + \mathsf{Adv}^{\text{GA-DDH}}(\mathcal{B}_2).$$

Taking $\mathcal{B}$ to be whichever of $\mathcal{B}_1$ and $\mathcal{B}_2$ has greater advantage yields the claim. □

Claim. For any algorithm $\mathcal{A}$ that makes at most q total queries to $\mathcal{H}, \mathcal{O}_{\mathsf{Sign}}, \mathcal{O}_{\mathsf{Simul}}$, and $\mathcal{O}_{\mathsf{Verify}}$, we have

$$\mathsf{Adv}^{\mathsf{Game}_2}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{Game}_3}(\mathcal{A}) + \frac{(q+1)^2}{2^{2\lambda}}.$$

Proof. As in the proof of Theorem 3, this follows by the birthday bound, noting that at most $q+1$ hash values are defined: q by the adversary's queries, and one by the challenge signature. □

We now proceed to Game_4, depicted in Fig. 9.

$\mathsf{Game}_4(1^\lambda)$

101 : $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
102 : **for** $i = 0, 1$
103 : $\quad (\mathsf{sk}_{S,i}, \mathsf{pk}_{S,i}) \leftarrow \mathsf{SKeyGen}(\mathsf{pp})$
104 : $(\mathsf{sk}_V, \mathsf{pk}_V) \leftarrow \mathsf{VKeyGen}(\mathsf{pp})$
105 : $b \xleftarrow{\$} \{0,1\}$
106 : $r_0 \xleftarrow{\$} \mathbb{Z}_N$
107 : $R_0 \leftarrow [\mathfrak{g}^{r_0}] * E$
108 : $r_1 \xleftarrow{\$} \mathbb{Z}_N$
109 : $R_1 \leftarrow [\mathfrak{g}^{r_1}] * E$
110 : $m^* \leftarrow \mathcal{A}_1^{\mathcal{H}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}}}(\mathsf{pk}_{S,0}, \mathsf{pk}_{S,1}, \mathsf{pk}_V)$
111 : $\sigma^* := (h^*, z^*) \xleftarrow{\$} \{0,1\}^{2\lambda} \times \mathbb{Z}_N$
112 : **assert** $\nexists x \neq x'$ s.t. $L[x] = L[x'] \neq \bot$
113 : $b' \leftarrow \mathcal{A}_2^{\mathcal{H}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}'_{\mathsf{Verify}}}(\mathsf{pk}_{S,0}, \mathsf{pk}_{S,1}, \mathsf{pk}_V, m^*, \sigma^*)$
114 : **return** $[b = b']$

$\mathcal{O}_{\mathsf{Simul}}(i, m)$

201 : $z \xleftarrow{\$} \mathbb{Z}_N$
202 : $h \xleftarrow{\$} \{0,1\}^{2\lambda}$
203 : $\sigma \leftarrow (h, z)$
204 : $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(i, m, \sigma)\}$
205 : **return** σ

$\mathcal{H}(x)$

301 : **if** $L[x] = \bot$
302 : $\quad L[x] \xleftarrow{\$} \{0,1\}^{2\lambda}$
303 : **return** $L[x]$

$\mathcal{O}_{\mathsf{Verify}}(i, (m, \sigma))$

401 : **return** $[(i, m, \sigma) \in \mathcal{Q}]$

$\mathcal{O}'_{\mathsf{Verify}}(i, (m, \sigma))$

501 : **assert** $(m, \sigma) \neq (m^*, \sigma^*)$
502 : **return** $[(i, m, \sigma) \in \mathcal{Q}]$

Fig. 9: The Game_4 used in the proof of Theorem 4.

Claim. For any algorithm $\mathcal{A}$ that makes at most $q_{\mathcal{H}}$ (respectively, $q_{\mathsf{Simul}}, q_{\mathsf{Verify}}$) queries to $\mathcal{H}$ (respectively $\mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}}$), we have

$$\mathsf{Adv}^{\mathsf{Game}_3}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{Game}_4}(\mathcal{A}) + \frac{q_{\mathcal{H}} q_{\mathsf{Verify}}}{2^{2\lambda}} + \frac{q_{\mathsf{Simul}} q_{\mathsf{Verify}}}{2^{2\lambda} N} + \frac{(q_{\mathcal{H}} + q_{\mathsf{Simul}})(2q_{\mathsf{Simul}} + q_{\mathsf{Verify}}) + 2(q_{\mathcal{H}} + q_{\mathsf{Simul}} + q_{\mathsf{Verify}})}{N}.$$

Proof. From the adversary's perspective, Game_3 is identical to Game_4 unless the result of an oracle query is contradicted by a later oracle query, or the challenge message/signature pair (m^*, σ^*) contradicts or is later contradicted by the output of an oracle query.

We first enumerate the ways that the oracles can contradict one another:

$\mathcal{O}_{\mathsf{Simul}}$ **contradicts** $\mathcal{O}_{\mathsf{Simul}}$**:** The adversary queries $\mathcal{O}_{\mathsf{Simul}}$ twice at the same pair (i, m), and receives two signatures with the same value of z and different values of h. Or, the adversary makes queries of the form $\mathcal{O}_{\mathsf{Simul}}(0, m)$ and $\mathcal{O}_{\mathsf{Simul}}(1, m)$ and receives $\sigma_0 = (h_0, z_0)$ and $\sigma_1 = (h_1, z_1)$, respectively, with $h_1 \neq h_0$ and $z_1 = z_0 + r_0 - r_1$.

Each query to $\mathcal{O}_{\mathsf{Simul}}$ uses a fresh, uniformly-random value of z. Thus, the probability that two calls to $\mathcal{O}_{\mathsf{Simul}}$ use the same value of z is at most $\frac{q_{\mathsf{Simul}}^2}{N}$ by the birthday bound. Similarly, the probability that two calls to $\mathcal{O}_{\mathsf{Simul}}$ use values of z that differ by $r_0 - r_1$ is at most $\frac{q_{\mathsf{Simul}}^2}{N}$, again by the birthday bound. Thus, this event occurs with probability at most $\frac{2q_{\mathsf{Simul}}^2}{N}$.

$\mathcal{O}_{\mathsf{Verify}}/\mathcal{O}'_{\mathsf{Verify}}$ **contradicts** $\mathcal{O}_{\mathsf{Simul}}$**:** $\mathcal{O}_{\mathsf{Verify}}(i, (m, \sigma))$ returns 0 after $\mathcal{O}_{\mathsf{Simul}}(i, m)$ has returned σ. Or, $\mathcal{O}_{\mathsf{Verify}}(i, (m, \sigma_i = (h, z_i))$ returns 0 after $\mathcal{O}_{\mathsf{Simul}}(1 - i, m)$ has returned $\sigma_{1-i} = (h, z_{1-i})$, where $z_{1-i} = z_i + r_i - r_{1-i}$.

By construction, the first possibility never happens. For the second possibility, until such a query occurs, the value of $r_0 - r_1$ is information-theoretically hidden from the adversary, and so such a query can only be made by chance. Since there are N possible values of $r_0 - r_1$ and the adversary makes at most q_{Verify} queries to $\mathcal{O}_{\mathsf{Verify}}$ and $\mathcal{O}'_{\mathsf{Verify}}$ together and at most q_{Simul} queries to $\mathcal{O}_{\mathsf{Simul}}$, this occurs with probability at most $\frac{q_{\mathsf{Verify}} q_{\mathsf{Simul}}}{N}$.

$\mathcal{H}$ **contradicts** $\mathcal{O}_{\mathsf{Simul}}$**:** The adversary queries $\mathcal{O}_{\mathsf{Simul}}$ at (i, m), and it returns a signature $\sigma = (h, z)$. Later, the adversary queries $\mathcal{H}$ at $([g^z] * R_i \,\|\, m)$ and it returns $h' \neq h$.

Up until such a query occurs, R_0 and R_1 are information-theoretically hidden from the adversary, and so such a query cannot be made except by random chance. There are N possible values of R_0 (and R_1), and the adversary makes at most $q_{\mathcal{H}}$ queries to $\mathcal{H}$ and at most q_{Simul} queries to $\mathcal{O}_{\mathsf{Simul}}$. Thus, this event occurs with probability at most $\frac{q_H q_{\mathsf{Simul}}}{N}$.

$\mathcal{O}_{\mathsf{Simul}}$ **contradicts** $\mathcal{O}_{\mathsf{Verify}}$**:** The adversary queries $\mathcal{O}_{\mathsf{Verify}}$ at some $(i, (m, \sigma = (h, z_i)))$ and it is rejected. Later, the adversary queries $\mathcal{O}_{\mathsf{Simul}}$ at (i, m) and the oracle returns (h, z_i), or at $(1 - i, m)$ and it returns $(h, z_i + r_i - r_{1-i})$.

Since each call to $\mathcal{O}_{\mathsf{Simul}}$ generates independent, uniformly random values of z and h, and there are at most q_{Simul} queries to $\mathcal{O}_{\mathsf{Simul}}$ and q_{Verify} queries to $\mathcal{O}_{\mathsf{Verify}}$, this event happens with probability at most $\frac{q_{\mathsf{Verify}} q_{\mathsf{Simul}}}{2^{2\lambda} N}$.

$\mathcal{O}_{\mathsf{Verify}}$ **contradicts** $\mathcal{O}_{\mathsf{Verify}}$**:** The adversary queries $\mathcal{O}_{\mathsf{Verify}}$ at some $(i, (m, \sigma))$ twice, and one query returns 0 while the other returns 1. Or, the adversary queries $\mathcal{O}_{\mathsf{Verify}}(0, (m, \sigma_0 = (h, z_0)))$ and $\mathcal{O}_{\mathsf{Verify}}(1, (m, \sigma_1 = (h, z_1)))$ with $z_1 = z_0 + r_0 - r_1$, and one query returns 0 while the other returns 1.

The first possibility can only happen in the following way: (i, m, σ) is rejected because it is not in $\mathcal{Q}$, then a query to $\mathcal{O}_{\mathsf{Simul}}$ adds (i, m, σ) to $\mathcal{Q}$, and then the adversary queries $\mathcal{O}_{\mathsf{Verify}}$ at (i, m, σ) again. This case is subsumed by the previous one.

The second possibility can only happen in two ways:

1. For some $i \in \{0,1\}$, $(i,(m,\sigma_i))$ is rejected because it is not in $\mathcal{Q}$, then a query to $\mathcal{O}_{\mathsf{Simul}}$ adds $(1-i,m,\sigma_{1-i})$ to $\mathcal{Q}$, and then the adversary queries $\mathcal{O}_{\mathsf{Verify}}$ at $(1-i,(m,\sigma_{1-i}))$. This case is subsumed by the previous one.
2. For some $i \in \{0,1\}$, a query to $\mathcal{O}_{\mathsf{Simul}}$ adds $(1-i,m,\sigma_{1-i})$ to $\mathcal{Q}$, and then the adversary queries $\mathcal{O}_{\mathsf{Simul}}$ at both $(1-i,(m,\sigma_{1-i}))$ and $(i,(m,\sigma_i))$ (in some order) *without* querying $\mathcal{O}_{\mathsf{Simul}}(i,m)$ and receiving σ_i. This is subsumed by the earlier case "$\mathcal{O}_{\mathsf{Verify}}$ contradicts $\mathcal{O}_{\mathsf{Simul}}$".

Altogether, this case is subsumed entirely by earlier ones.

$\mathcal{H}$ contradicts $\mathcal{O}_{\mathsf{Verify}}$: The adversary queries $\mathcal{O}_{\mathsf{Verify}}$ at some $(i,(m,\sigma=(h,z)))$ and it is rejected. Later, the adversary queries $\mathcal{H}$ at $([\mathfrak{g}^z] * R_i \,\|\, m)$ and it returns h.

Since $\mathcal{H}([\mathfrak{g}^z] * R_i \;||\; m)$ is set uniformly at random when the query to $\mathcal{H}$ is made, and the adversary makes at most $q_{\mathcal{H}}$ queries to $\mathcal{H}$ and q_{Verify} queries to $\mathcal{O}_{\mathsf{Verify}}$, this occurs with probability at most $\frac{q_{\mathcal{H}} q_{\mathsf{Verify}}}{2^{2\lambda}}$.

$\mathcal{O}_{\mathsf{Simul}}$ contradicts $\mathcal{H}$: The adversary queries $\mathcal{H}(Z \,\|\, m)$ for some curve Z, and it returns h. Later, the adversary queries $\mathcal{O}_{\mathsf{Simul}}$ at (i,m) and the oracle returns $\sigma=(h',z)$ with $h' \neq h$ and $[g^z] * R_i = Z$.

Since the action used in CSI-FiSh is free, for each (i,Z) there is exactly one value of z such that $Z = [g^z] * R_i$. Since each call to $\mathcal{O}_{\mathsf{Simul}}$ generates an independent, uniformly random value of z, and there are at most $q_{\mathcal{H}}$ queries to $\mathcal{H}$ and q_{Simul} queries to $\mathcal{O}_{\mathsf{Simul}}$, this event happens with probability at most $\frac{q_{\mathcal{H}} q_{\mathsf{Simul}}}{N}$.

$\mathcal{O}_{\mathsf{Verify}}$ contradicts $\mathcal{H}$: The adversary queries $\mathcal{H}$ at some $(Z \,\|\, m)$ and it returns h. Later, the adversary queries $\mathcal{O}_{\mathsf{Verify}}$ at $(i,(m,\sigma=(h,z)))$ such that $Z = [g^z] * R_i$ and the oracle returns 0.

Up until such a query occurs, R_0 and R_1 are information-theoretically hidden from the adversary, and so such a query cannot be made except by random chance. There are N possible values of each R_i, and the adversary makes at most $q_{\mathcal{H}}$ queries to $\mathcal{H}$ and at most q_{Verify} queries to $\mathcal{O}_{\mathsf{Verify}}$. Thus, this event occurs with probability at most $\frac{q_H q_{\mathsf{Verify}}}{N}$.

$\mathcal{H}$ contradicts $\mathcal{H}$: The adversary queries $\mathcal{H}$ twice at the same input x and receives two different outputs. By construction, this never happens.

Now we must consider the ways that the challenge message/signature pair (m^*,σ^*) can contradict or be contradicted by an oracle query. To begin, (m^*,σ^*) implicitly defines exactly one hash value: $\mathcal{H}([g^{z^*}] * R_b \,\|\, m) = h^*$. Since z^* is chosen uniformly at random when the challenge signature is constructed, $[\mathfrak{g}^{z^*}] * R_b$ is a uniformly random curve; thus, the probability that this contradicts an existing hash value is at most $\frac{q_H + q_{\mathsf{Simul}} + q_{\mathsf{Verify}}}{N}$.

Later oracle queries may contradict the challenge message/signature pair as follows:

$\mathcal{O}_{\mathsf{Simul}}$ **contradicts** (m^*, σ^*)**:** The adversary queries $\mathcal{O}_{\mathsf{Simul}}$ at some (i, m^*), and it returns $\sigma = (h^*, z)$ where $[\mathfrak{g}^z] * R_i = [\mathfrak{g}^{z^*}] * R_b$.

Since the group action in CSI-FiSh is free, for each i there is exactly one value of z such that $[\mathfrak{g}^z] * R_i = [\mathfrak{g}^{z^*}] * R_b$. Since z is chosen uniformly at random in each call to $\mathcal{O}_{\mathsf{Simul}}$, this event occurs with probability at most $\frac{q_{\mathsf{Simul}}}{N}$.

$\mathcal{O}_{\mathsf{Verify}}$ **contradicts** (m^*, σ^*)**:** The adversary queries $\mathcal{O}_{\mathsf{Verify}}$ at $(1-b, (m^*, \sigma = (h^*, z)))$, where $z = z^* + r_b - r_{1-b}$, and the oracle returns 0.

Until this $\mathcal{O}_{\mathsf{Verify}}$ query is made, $r_b - r_{1-b}$ is information-theoretically hidden from the adversary. Thus, such a query can be made only by chance. Thus, the probability that such a query is made is at most $\frac{q_{\mathsf{Verify}}}{N}$.

$\mathcal{H}$ **contradicts** (m^*, σ^*)**:** The adversary queries $\mathcal{H}$ at $([g^{z^*}] * R_b \,\|\, m)$ and it returns $h \neq h^*$.

Until this $\mathcal{H}$ query is made, R_b is information-theoretically hidden from the adversary. Thus, such a query can be made only by chance. Thus, the probability that such a query is made is at most $\frac{q_{\mathcal{H}}}{N}$.

Summing all of the individual probabilities, we find that

$$\mathsf{Adv}^{\mathsf{Game}_3}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{Game}_4}(\mathcal{A}) + \frac{q_{\mathcal{H}} q_{\mathsf{Verify}}}{2^{2\lambda}} + \frac{q_{\mathsf{Simul}} q_{\mathsf{Verify}}}{2^{2\lambda} N} + \frac{(q_{\mathcal{H}} + q_{\mathsf{Simul}})(2q_{\mathsf{Simul}} + q_{\mathsf{Verify}}) + 2(q_{\mathcal{H}} + q_{\mathsf{Simul}} + q_{\mathsf{Verify}})}{N}.$$

□

Claim. For any adversary $\mathcal{A}$, we have $\mathsf{Adv}^{\mathsf{Game}_4}(\mathcal{A}) = 0$.

Proof. In Game_4, the challenge signature and all oracle responses are independent of the value of b chosen by the adversary. Since b is chosen uniformly at random from $\{0, 1\}$, regardless of how $\mathcal{A}$ chooses b', the probability that $b' = b$ is $\frac{1}{2}$, and $\mathcal{A}$'s advantage is 0. □

Proof (of Theorem 4). Altogether, for any adversary $\mathcal{A}$ running in time T and making at most $q_{\mathcal{H}}$ (respectively, $q_{\mathsf{Simul}}, q_{\mathsf{Verify}}$) queries to $\mathcal{H}$ (respectively $\mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}}$), there is an adversary $\mathcal{B}$ that runs in time $T + \mathsf{poly}(\lambda)$ such that

$$\mathsf{Adv}^{\mathsf{PSI}}_{\mathsf{CSI\text{-}SDVS}}(\mathcal{A}) \leq 2\mathsf{Adv}^{\text{GA-DDH}}(\mathcal{B}) + \frac{q^2 + q_{\mathcal{H}} q_{\mathsf{Verify}}}{2^{2\lambda}} + \frac{q_{\mathsf{Simul}} q_{\mathsf{Verify}}}{2^{2\lambda} N} + \frac{(q_{\mathcal{H}} + q_{\mathsf{Simul}})(2q_{\mathsf{Simul}} + q_{\mathsf{Verify}}) + 2(q_{\mathcal{H}} + q_{\mathsf{Simul}} + q_{\mathsf{Verify}})}{N}$$

where $q = q_{\mathcal{H}} + q_{\mathsf{Simul}} + q_{\mathsf{Verify}}$. Thus, when $\mathcal{A}$ is PPT (so that $T = \mathsf{poly}(\lambda)$ and $q = \mathsf{poly}(\lambda)$), $\mathsf{Adv}^{\mathsf{PSI}}_{\mathsf{CSI\text{-}SDVS}}(\mathcal{A}) = \mathsf{negl}(\lambda)$ under the GA-DDH assumption, as required. □

5 Tight Reductions to GAIP in the AGAM

As shown in Section 4, our security proofs rely on the GA-DDH assumption. It is desirable to base such reductions on computational problems, such as GAIP or GA-CDH, as in [12]. In this section, we show that, in the AGAM (see Section 2.3.1) and the random oracle model, the SUF-CMA and PSI security of CSI-SDVS reduce tightly to the hardness of GAIP. We first establish SUF-CMA security.

Theorem 5. *Assuming the hardness of GAIP (Definition 6), the proposed scheme,* CSI-SDVS*, is SUF-CMA secure, as defined in Definition 3, in the AGAM and the random oracle model.*

At a high level, the idea of the proof is as follows: we embed the GAIP instance X into the game either by setting $\mathsf{pk}_S := X$ or $\mathsf{pk}_V := X$. In either case, letting $\mathcal{A}$ denote an adversary in the AGAM and ROM who wins the SUF-CMA game with non-negligible probability, we examine the message/signature pair $(m, \sigma = (h, z))$ that it produces. With high probability, we will have $h = \mathcal{H}(Y \,\|\, m)$ for a unique $Y \in \mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O})$. Then, because we work in the AGAM, $\mathcal{A}$ will have provided us with a explanation $Y = [\mathfrak{g}^a] * A$ for Y, where A is among the curves we provided to $\mathcal{A}$ initially; in particular, $A \in \{E, \mathsf{pk}_S, \mathsf{pk}_V\}$. Then, if $A \neq X$, we can extract a solution to the GAIP instance. Since X is embedded uniformly at random into pk_S or pk_V, and since the reductions behave identically from the perspective of $\mathcal{A}$ in those two cases, we will be able to extract the GAIP instance solution with non-negligible probability.

The reduction is depicted in Figs. 10, 11, and 12.

$\mathcal{B}(X)$

101 : $\mathsf{ID} \xleftarrow{\$} \{S, V\}$

102 : $x \leftarrow \mathcal{B}^{\mathsf{ID}}(X)$

103 : **return** x

Fig. 10: The reduction $\mathcal{B}$ from GAIP to SUF-CMA security of CSI-SDVS.

For the proof of Theorem 5, we proceed through a sequence of claims.

Claim. Let $\mathsf{pp} = (p, \mathfrak{g}, N, E)$ be the pubic parameters of a GAIP instance X. If $\perp \neq x \leftarrow \mathcal{B}(X)$, then $[\mathfrak{g}^x] * E = X$; that is, x is the solution to the GAIP instance.

Proof. To begin, we address the case that $\mathsf{ID} = S$. We note that in this case, $\mathcal{B}$ will only return $x \neq \perp$ as a result of lines 112 and 113 of Figure 11. Suppose that x is returned on line 112, so that $x = a - z - \mathsf{sk}_V$. Then, since all 3 assert statements of $\mathcal{B}^S$ have passed, we know that there is a unique $Y \in \mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O})$ such that

$$\mathcal{H}(Y \,\|\, m) = h^* = \mathcal{H}([\mathfrak{g}^{z^* + \mathsf{sk}_V}] * \mathsf{pk}_S \,\|\, m).$$

Since hash values are only defined by queries to $\mathcal{H}$ (including those made during signing, simulating, and verification), we necessarily have $Y = [\mathfrak{g}^{z^* + \mathsf{sk}_V}] * \mathsf{pk}_S$. Moreover, since we are working in the AGAM, we can also construct from $\mathcal{A}$'s queries a pair (a, A) such that $a \in \mathbb{Z}_N$, $A \in \{\mathsf{pk}_S, \mathsf{pk}_V, E\}$ and $Y = [\mathfrak{g}^a] * A$. If

$\mathcal{B}^S(X)$

101 : $\mathcal{Q} \leftarrow \varnothing, L \leftarrow [\,]$
102 : $(\mathsf{sk}_V, \mathsf{pk}_V) \leftarrow \mathsf{CSI\text{-}SDVS.VKeyGen}(\mathsf{pp})$
103 : $\mathsf{pk}_S \leftarrow X$
104 : $\mathsf{oracles} \leftarrow (\mathcal{H}, \mathcal{O}_{\mathsf{Sign}}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}})$
105 : $(m^*, \sigma^* = (h^*, z^*)) \leftarrow \mathcal{A}^{\mathsf{oracles}}(\mathsf{pk}_S, \mathsf{pk}_V)$
106 : **assert** $(m^*, \sigma^*) \notin \mathcal{Q}$
107 : **assert** $\exists!\ Y \in \mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O})$ s.t. $h^* = L[Y \,\|\, m^*]$
108 : **assert** $\mathsf{CSI\text{-}SDVS.Verify}(\mathsf{sk}_V, \mathsf{pk}_S, m^*, \sigma^*)$
109 : **assert** $\mathsf{GetRep}(Y) \neq \perp$
110 : $(a, A) \leftarrow \mathsf{GetRep}(Y)$
111 : **if** $A = \mathsf{pk}_S$: **return** $\perp$
112 : **if** $A = E$: **return** $a - z^* - \mathsf{sk}_V$
113 : **if** $A = \mathsf{pk}_V$: **return** $a - z^*$

$\mathcal{O}_{\mathsf{Sign}}(m)$, $\mathcal{O}_{\mathsf{Simul}}(m)$

201 : $z \xleftarrow{\$} \mathbb{Z}_N$
202 : $h \leftarrow \mathcal{H}([g^{z+\mathsf{sk}_V}] * \mathsf{pk}_S \,\|\, m)$
203 : $\sigma \leftarrow (h, z)$
204 : $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(m, \sigma)\}$
205 : **return** σ

$\mathcal{H}(x)$

301 : **if** $L[x] = \perp$
302 : $\quad L[x] \xleftarrow{\$} \{0,1\}^\lambda$
303 : **return** $L[x]$

$\mathcal{O}_{\mathsf{Verify}}(i, m, \sigma)$

401 : **parse** $(h, z) \leftarrow \sigma$
402 : $h' \leftarrow \mathcal{H}([\mathfrak{g}^{z+\mathsf{sk}_V}] * \mathsf{pk}_S \,\|\, m)$
403 : **return** $[h = h']$

Fig. 11: The reduction when the GAIP instance in embedded in pk_S.

$\mathcal{B}^V(X)$

101 : $\mathcal{Q} \leftarrow \varnothing, L \leftarrow [\,]$
102 : $(\mathsf{sk}_S, \mathsf{pk}_S) \leftarrow \mathsf{CSI\text{-}SDVS.VKeyGen}(\mathsf{pp})$
103 : $\mathsf{pk}_V \leftarrow X$
104 : $\mathsf{oracles} \leftarrow (\mathcal{H}, \mathcal{O}_{\mathsf{Sign}}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}})$
105 : $(m^*, \sigma^* = (h^*, z^*)) \leftarrow \mathcal{A}^{\mathsf{oracles}}(\mathsf{pk}_S, \mathsf{pk}_V)$
106 : **assert** $(m^*, \sigma^*) \notin \mathcal{Q}$
107 : **assert** $\exists!\ Y \in \mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O})$ s.t. $h = L[Y \,\|\, m^*]$
108 : **assert** $\mathsf{CSI\text{-}SDVS.Verify}'(\mathsf{sk}_V, \mathsf{pk}_S, m^*, \sigma^*)$
109 : $(a, A) \leftarrow \mathsf{GetRep}(Y)$
110 : **if** $A = \mathsf{pk}_S$: **return** $a - z^*$
111 : **if** $A = E$: **return** $a - z^* - \mathsf{sk}_S$
112 : **if** $A = \mathsf{pk}_V$: **return** $\perp$

$\mathcal{O}_{\mathsf{Sign}}(m)$, $\mathcal{O}_{\mathsf{Simul}}(m)$

201 : $z \xleftarrow{\$} \mathbb{Z}_N$
202 : $h \leftarrow \mathcal{H}([g^{z+\mathsf{sk}_S}] * \mathsf{pk}_V \,\|\, m)$
203 : $\sigma \leftarrow (h, z)$
204 : $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(m, \sigma)\}$
205 : **return** σ

$\mathcal{H}(x)$

301 : **if** $L[x] = \perp$
302 : $\quad L[x] \xleftarrow{\$} \{0,1\}^\lambda$
303 : **return** $L[x]$

$\mathcal{O}_{\mathsf{Verify}}(i, m, \sigma)$

401 : **parse** $(h, z) \leftarrow \sigma$
402 : $h' \leftarrow \mathcal{H}([\mathfrak{g}^{z+\mathsf{sk}_S}] * \mathsf{pk}_V \,\|\, m)$
403 : **return** $[h = h']$

Fig. 12: The reduction when the GAIP instance in embedded in pk_V.

$A = E$, by line 112 we have $x = a - z^* - \mathsf{sk}_V$, and moreover we have (recalling that $\mathsf{pk}_S = X$ in this case):

$$[\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_S = [\mathfrak{g}^a] * E \implies X = [\mathfrak{g}^{a-z^*-\mathsf{sk}_V}] * E = [\mathfrak{g}^x] * E$$

so that, indeed, x is the solution to the GAIP instance X. Similarly, if $A = \mathsf{pk}_V$, we have $x = a - z^*$, and then

$$[\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_S = [\mathfrak{g}^a] * \mathsf{pk}_V \implies X = [\mathfrak{g}^{a-z^*}] * E = [\mathfrak{g}^x] * E$$

(where we have used the identity $\mathsf{pk}_V = [\mathfrak{g}^{\mathsf{sk}_V}] * E$) so that, again, x is a solution to the GAIP instance X.

The argument when $\mathsf{ID} = V$ is similar, and so we omit it. □

Claim. $\mathcal{B}^S$ and $\mathcal{B}^V$ perfectly simulate the SUF-CMA game to $\mathcal{A}$. In particular: fix public parameters $(p, [\mathfrak{g}], N, E, \mathcal{H})$ for CSI-SDVS. Let $\mathrm{View}(\mathcal{A})$ denote the adversary's *view* in the SUF-CMA game—more precisely, $\mathrm{View}(\mathcal{A})$ is the joint distribution of the public values $\mathsf{pk}_S, \mathsf{pk}_V$ and all oracle responses given to $\mathcal{A}$ in the course of the SUF-CMA experiment. Similarly, let $\mathrm{View}_S(\mathcal{A})$ and $\mathrm{View}_V(\mathcal{A})$ denote the adversary's view in the reductions $\mathcal{B}^S$ and $\mathcal{B}^V$, respectively, when X is a uniformly-chosen GAIP instance. Then, the distributions $\mathrm{View}(\mathcal{A}), \mathrm{View}_S(\mathcal{A})$, and $\mathrm{View}_V(\mathcal{A})$ are identical.

Proof. We prove that the distributions $\mathrm{View}(\mathcal{A})$, and $\mathrm{View}_S(\mathcal{A})$ are identical; the proof for $\mathrm{View}_V(\mathcal{A})$ is similar.

To begin, note that the public values $(\mathsf{pk}_S, \mathsf{pk}_V)$ are distributed uniformly from $\mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O})^2$ in both games, as required. We then proceed to consider oracle queries. Note that $\mathcal{O}_{\mathsf{Simul}}, \mathcal{H}$, and $\mathcal{O}_{\mathsf{Verify}}$ behave identically in both games, so we need only consider $\mathcal{O}_{\mathsf{Sign}}$. Note that in $\mathcal{B}^S$, $\mathcal{O}_{\mathsf{Sign}}(m)$ actually runs $\mathsf{CSI\text{-}SDVS.Simul}(m)$; by Theorem 1, this also is exactly the same behaviour as $\mathsf{CSI\text{-}SDVS.Sign}$, which behaves identically to $\mathcal{O}_{\mathsf{Sign}}$ in the SUF-CMA game. Thus, indeed, the adversary's view is the same in both the true SUF-CMA game and in $\mathcal{B}^S$.

The proof for $\mathcal{B}^V$ is analogous. □

Consequently, the probability that $\mathcal{A}$'s response to $\mathcal{B}$ is one that would win the SUF-CMA game is exactly $\mathsf{Adv}^{\mathsf{SUF\text{-}CMA},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(\lambda)$.

Claim. Let $\mathcal{A}$ be an adversary for the SUF-CMA game against CSI-SDVS. Suppose that $\mathcal{A}$ makes at most q total queries to the oracles $\mathcal{H}, \mathcal{O}_{\mathsf{Sign}}, \mathcal{O}_{\mathsf{Simul}}$, and $\mathcal{O}_{\mathsf{Verify}}$. Define

$$\mathsf{Succ}^{\mathcal{B}^{\mathcal{A}}}(\lambda) = \mathbb{P}\left[x' \neq \bot \;\middle|\; \begin{array}{r} (p, \mathfrak{g}, N, E, \mathcal{H}) \leftarrow \mathsf{Setup}(1^\lambda) \\ x \xleftarrow{\$} \mathbb{Z}_N \\ X \leftarrow [\mathfrak{g}^x] * E \\ x' \leftarrow \mathcal{B}^{\mathcal{A}}(X) \end{array} \right].$$

Then

$$\mathsf{Succ}^{\mathcal{B}^{\mathcal{A}}}(\lambda) \geq \frac{1}{2}\mathsf{Adv}^{\mathsf{SUF\text{-}CMA},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(\lambda) - \frac{q^2 + q_{\mathsf{Verify}} + 1}{2^{2\lambda}}.$$

Proof. Since ID is sampled uniformly at random from $\{S, V\}$ in the execution of $\mathcal{B}^{\mathcal{A}}$, we have

$$\mathsf{Succ}^{\mathcal{B}^{\mathcal{A}}}(1^\lambda) = \frac{1}{2}\mathbb{P}[\mathcal{B}^{\mathcal{A}}(X) \neq \perp \mid \mathsf{ID} = S] + \frac{1}{2}\mathbb{P}[\mathcal{B}^{\mathcal{A}}(X) \neq \perp \mid \mathsf{ID} = V]$$

and so it suffices to analyze the success probability of $\mathcal{B}^S$ and $\mathcal{B}^V$ individually. Beginning with $\mathcal{B}^S$, referring to Fig. 11, it is clear that $\mathcal{B}^S$ succeeds as long as:

1. $\mathcal{A}^{\mathsf{oracles}}(\mathsf{pk}_S, \mathsf{pk}_V)$ wins the instance of the SUF-CMA game;
2. When A is constructed on line 110, we have $A \neq \mathsf{pk}_S$;
3. The assertion on line 107 does not fail; and,
4. The assertion on line 109 does not fail.

On inputs of length λ, the first condition is satisfied with probability $\mathsf{Adv}^{\mathsf{SUF\text{-}CMA},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(1^\lambda)$ by definition. From now on, assume that $\mathcal{A}$ has won the SUF-CMA game.

The cases on lines 111–113 are exhaustive; let p_S, p_E, p_V denote the probability that—in addition to $\mathcal{A}$ succeeding in the SUF-CMA game—$A = \mathsf{pk}_S$, $A = E$, and $A = \mathsf{pk}_V$, respectively. We have

$$\mathsf{Adv}^{\mathsf{SUF\text{-}CMA},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(1^\lambda) = p_S + p_E + p_V.$$

When computing $\mathbb{P}[\mathcal{B}^{\mathcal{A}}(X) \neq \perp \mid \mathsf{ID} = S]$, we must subtract p_S from this probability, because if $A = \mathsf{pk}_S$ then $\mathcal{B}^S$ fails on line 111. We must also subtract the probability of the assertion on line 107 fails. There are two ways that it can fail:

1. There does not exist any $Y \in \mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O})$ such that $h^* = L[Y \,\|\, m^*]$; or,
2. There exist two or more $Y \in \mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O})$ such that $h^* = L[Y \,\|\, m^*]$.

In the first case, $\mathcal{A}$'s signature verifies despite the fact that $\mathcal{A}$ has not queried (directly or indirectly) $\mathcal{H}(Y \,\|\, m^*)$; this occurs with probability $2^{-2\lambda}$, since $\mathcal{H}(Y \,\|\, m^*)$ would be chosen uniformly at random while verifying $\mathcal{A}$'s forgery during the SUF-CMA game. In the second case, there has been a collision in $\mathcal{H}$, which occurs with probability at most $\frac{q^2}{2^{2\lambda}}$ by the birthday bound.

We must also subtract the probability that the assertion on line 109 fails. Note that we only reach this line if the assertion on line 107 succeeds, so we may assume that there is some $Y \in \mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O})$ such that $h^* = L[Y \,\|\, m^*]$. Moreover, the assertion on line 106 must also have passed, meaning that $L[Y \,\|\, m^*]$ was not defined by a query to $\mathcal{O}_{\mathsf{Sign}}$ or $\mathcal{O}_{\mathsf{Simul}}$. Thus, the only way for the assertion on line 109 to fail is if $L[Y \,\|\, m^*]$ is defined by a query to $\mathcal{O}_{\mathsf{Verify}}$, and its value is never revealed by $\mathcal{O}_{\mathsf{Sign}}, \mathcal{O}_{\mathsf{Simul}}$, or $\mathcal{H}$. If that is the case, the probability that the assertion on line 108 passes is at most $\frac{q_{\mathsf{Verify}}+1}{2^{2\lambda}}$, since each of the q_{Verify} queries to $\mathcal{O}_{\mathsf{Verify}}$ can eliminate at most one incorrect value of $h \neq h^*$.

It follows that the probability that $\mathcal{B}^S$ succeeds is at least

$$\mathbb{P}[\mathcal{B}^{\mathcal{A}}(1^\lambda) \neq \perp \mid \mathsf{ID} = S] \geq p_E + p_V - \frac{q^2 + q_{\mathsf{Verify}} + 1}{2^{2\lambda}}.$$

A similar argument yields

$$\mathbb{P}[\mathcal{B}^{\mathcal{A}}(1^\lambda) \neq \perp \mid \mathsf{ID} = V] \geq p_E + p_S - \frac{q^2 + q_{\mathsf{Verify}} + 1}{2^{2\lambda}}$$

where we note that, from the adversary's perspective, the reductions $\mathcal{B}^S$ and $\mathcal{B}^V$ are identical, so the probabilities that $A = E, \mathsf{pk}_S$, or pk_V are the same in both reductions. Thus,

$$\mathsf{Succ}^{\mathcal{B}^{\mathcal{A}}}(1^\lambda) \geq p_E + \frac{1}{2}p_S + \frac{1}{2}p_V - \frac{q^2+1}{2^{2\lambda}} \geq \frac{1}{2}\mathsf{Adv}^{\mathsf{SUF\text{-}CMA},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(1^\lambda) - \frac{q^2 + q_{\mathsf{Verify}} + 1}{2^{2\lambda}}$$

as required.

Under the GAIP assumption, if $\mathcal{A}$ is polynomial time then $\mathsf{Succ}^{\mathcal{B}^{\mathcal{A}}}(1^\lambda)$ is negligible, as is $\frac{q^2+q_{\mathsf{Verify}}+1}{2^{2\lambda}}$; thus, $\mathsf{Adv}^{\mathsf{SUF\text{-}CMA},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(1^\lambda)$ is negligible, establishing the SUF-CMA security of CSI-SDVS. □

Theorem 6. *Assuming the hardness of GAIP (Definition 6), the proposed scheme,* CSI-SDVS*, is PSI secure, as defined in Definition 4, in the AGAM and the random oracle model.*

At a high level, the proof of Theorem 6 uses the same technique as that of Theorem 5: any algebraic adversary who wins the PSI game with non-negligible advantage must make a query to the random oracle from which we can extract a solution to a GAIP instance. The main difference arises from how we guarantee that the adversary makes a random oracle query of the correct form. While in the SUF-CMA game this was guaranteed by the check on line 108 of $\mathcal{B}^{\mathsf{ID}}$, we need a sequence of games reduction for the analogous statement in the PSI setting.

The proof proceeds in two phases: first we prove a tight relationship between any adversary's PSI advantage and the probability that they make a query of a specific form related to the challenge signature; then, we prove that we can use any adversary that makes such a query to extract a solution to a GAIP instance.

Lemma 7. *Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an algebraic adversary for the PSI experiment against* CSI-SDVS *in the ROM that takes time at most T and makes at most $q_{\mathcal{H}}$ queries to $\mathcal{H}$, q_{Simul} queries to $\mathcal{O}_{\mathsf{Simul}}$, and q_{Verify} queries to $\mathcal{O}_{\mathsf{Verify}}$ and $\mathcal{O}'_{\mathsf{Verify}}$. Then there exists an algorithm $\mathcal{C}$ for GAIP that runs in time $T + \mathsf{poly}(\lambda)$ such that*

$$\mathsf{Adv}^{\mathsf{PSI},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(1^\lambda) \leq \mathsf{Adv}^{\mathrm{GAIP},\mathcal{C}}(1^\lambda) + \frac{q_{\mathsf{Simul}} + q_{\mathsf{Verify}}}{N} + \frac{q_{\mathsf{Verify}}}{2^{2\lambda}} + \mathbb{P}[\mathcal{A} \text{ queries } \mathcal{H} \text{ at } ([\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b} \,\|\, m^*)]$$

where b is the value chosen on line 105 of Fig. 3.

Proof. Consider the games Game_0 and Game_1 depicted in Fig. 13. The two games behave identically from the adversary's perspective unless one of the adversary's queries to $\mathcal{H}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}}$, or $\mathcal{O}'_{\mathsf{Verify}}$ contradicts the definition of σ^* in Game_1. In more detail: in Game_0, the construction of σ^* defines a single random oracle value $\mathcal{H}([\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b} \,\|\, m^*) = h^*$, and this is used consistently by the other oracles. In contrast, in Game_1, this random oracle value is *not* programmed when σ^* is constructed, meaning that if the adversary queries (directly, or indirectly by a call

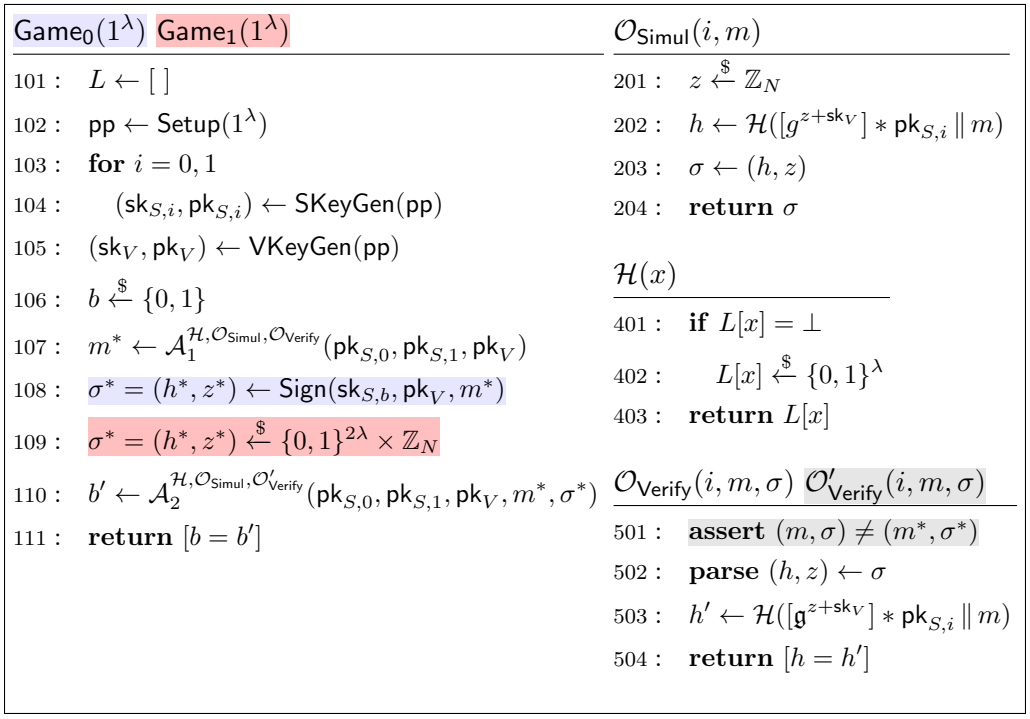


Fig. 13: Two games used in the proof of Lemma 7.

to $\mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}}$, or $\mathcal{O}'_{\mathsf{Verify}}$) $\mathcal{H}$ at this same input, the two games may differ. So we bound the probability that the adversary (directly or indirectly) queries this hash value in a way that contradicts σ^*.

We consider each oracle separately:

$\mathcal{H}$ contradicts σ^*: This can happen only if $\mathcal{A}$ queries $\mathcal{H}$ at $([g^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b} \,\|\, m^*)$, either before or after σ^* is constructed.

$\mathcal{O}_{\mathsf{Simul}}$ contradicts σ^*: This happens if and only if $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{Simul}}(i, m^*)$, the oracle chooses a value of z such that $[\mathfrak{g}^{z+\mathsf{sk}_V}] * \mathsf{pk}_{S,i} = [\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b}$, and then the random oracle output is programmed as some $h' \neq h^*$. Since the CSIDH group action is free, for each $i \in \{0,1\}$, there is a unique value of $z \in \mathbb{Z}_N$ for which this occurs: $z = z^* + \mathsf{sk}_{S,b} - \mathsf{sk}_{S,i}$. Since $\mathcal{O}_{\mathsf{Simul}}$ chooses each value of z independently and uniformly at random, this occurs with probability at most $\frac{q_{\mathsf{Simul}}}{N}$.

$\mathcal{O}_{\mathsf{Verify}}$ contradicts σ^*: In this case, $\mathcal{A}$ must have queried $\mathcal{O}_{\mathsf{Verify}}$ at some $(i, m^*, \sigma = (h, z))$ such that $[\mathfrak{g}^{z+\mathsf{sk}_V}] * \mathsf{pk}_{S,i} = [\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b}$, before σ^* was constructed. Since z^* is chosen uniformly at random on line 109 and the group action is free, this happens with probability at most $\frac{q_{\mathsf{Verify}}}{N}$.

$\mathcal{O}'_{\mathsf{Verify}}$ contradicts σ^*: A query $(i, m, \sigma = (h, z))$ to $\mathcal{O}'_{\mathsf{Verify}}$ can only contradict σ^* if $m = m^*$ and $[\mathfrak{g}^{z+\mathsf{sk}_V}] * \mathsf{pk}_{S,i} = [\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b}$. Since the group action is free, this means that $z = z^* + \mathsf{sk}_{S,b} - \mathsf{sk}_{S,i}$. We consider two cases:

1. If $i = b$, then $z = z^*$. The oracle $\mathcal{O}'_{\mathsf{Verify}}$ will not answer the query

$(i, m^*, (h^*, z^*))$, and so we may assume that $h \neq h^*$. In Game_0, any such query would return 0 because of the failed check on line 504. Thus, it suffices to bound the probability that this query returns 1. We consider two cases:

(a) A query to $\mathcal{H}$ or $\mathcal{O}_{\mathsf{Simul}}$ reveals the value of $L[[\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b} \,\|\, m^*]$. This is subsumed by the first two cases.

(b) No query to $\mathcal{H}$ or $\mathcal{O}_{\mathsf{Simul}}$ reveals the value of $L[[\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b} \,\|\, m^*]$. In this case, the value h' of $L[[\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b} \,\|\, m^*]$ is set uniformly at random on Line 503, either by a call to $\mathcal{O}_{\mathsf{Verify}}$ before σ^* is defined, or by a call to $\mathcal{O}'_{\mathsf{Verify}}$ after σ^* is defined. In any case, each query $(b, m^*, \sigma = (h, z^*))$ to $\mathcal{O}'_{\mathsf{Verify}}$ that returns 0 reveals one *incorrect* value of $L[[\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b} \,\|\, m^*]$, and no information about the correct value. After at most q_{Verify} such queries, the probability that one of these queries has had the correct value $h = h'$ (causing the oracle to output 1) is at most $\frac{q_{\mathsf{Verify}}}{2^{2\lambda}}$.

2. If $i = 1 - b$, then $z = z^* + \mathsf{sk}_{S,b} - \mathsf{sk}_{S,1-b}$. If $\mathcal{A}$ is an adversary running in time T that makes such a query with probability ε, then there exists an algorithm $\mathcal{C}$ that runs in time $T + \mathsf{poly}(\lambda)$ which solves the GAIP problem with probability λ. In particular, the adversary is given in Fig. 14.

$\mathcal{C}(X_0, X_1)$

101 : $V \leftarrow \varnothing, L \leftarrow [\,]$
102 : $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
103 : **for** $i = 0, 1$
104 : $\quad \mathsf{pk}_{S,i} \leftarrow X_i$
105 : $(\mathsf{sk}_V, \mathsf{pk}_V) \leftarrow \mathsf{VKeyGen}(\mathsf{pp})$
106 : $b \xleftarrow{\$} \{0,1\}$
107 : $m^* \leftarrow \mathcal{A}_1^{\mathcal{H}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}}}(\mathsf{pk}_{S,0}, \mathsf{pk}_{S,1}, \mathsf{pk}_V)$
108 : $\sigma^* = (h^*, z^*) \leftarrow \mathsf{Simul}(\mathsf{sk}_V, \mathsf{pk}_{S,b}, m^*)$
109 : $b' \leftarrow \mathcal{A}_2^{\mathcal{H}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}'_{\mathsf{Verify}}}(\mathsf{pk}_{S,0}, \mathsf{pk}_{S,1}, \mathsf{pk}_V, m^*, \sigma^*)$
110 : **for** $z \in V$
111 : $\quad$ **if** $[\mathfrak{g}^{z^*-z}] * \mathsf{pk}_{S,b} = \mathsf{pk}_{S,1-b}$
112 : $\quad\quad$ **return** $(-1)^b(z^* - z)$
113 : **return** $\perp$

$\mathcal{O}_{\mathsf{Simul}}(i, m)$

201 : $z \xleftarrow{\$} \mathbb{Z}_N$
202 : $h \leftarrow \mathcal{H}([g^{z+\mathsf{sk}_V}] * \mathsf{pk}_{S,i} \,\|\, m)$
203 : $\sigma \leftarrow (h, z)$
204 : **return** σ

$\mathcal{H}(x)$

301 : **if** $L[x] = \perp$
302 : $\quad L[x] \xleftarrow{\$} \{0,1\}^\lambda$
303 : **return** $L[x]$

$\mathcal{O}_{\mathsf{Verify}}(i, m, \sigma)$ $\mathcal{O}'_{\mathsf{Verify}}(i, m, \sigma)$

401 : **assert** $(m, \sigma) \neq (m^*, \sigma^*)$
402 : **parse** $(h, z) \leftarrow \sigma$
403 : **if** $(i, m) = (1 - b, m^*)$
404 : $\quad V \leftarrow V \cup \{z\}$
405 : $h' \leftarrow \mathcal{H}([\mathfrak{g}^{z+\mathsf{sk}_V}] * \mathsf{pk}_{S,i} \,\|\, m)$
406 : **return** $[h = h']$

Fig. 14: A GAIP algorithm $\mathcal{C}$ using a PSI adversary $\mathcal{A}$ as a subroutine.

We make three observations about $\mathcal{C}$:

1. $\mathcal{C}$ runs in time $T + \mathsf{poly}(\lambda)$. This is clear, since $\mathcal{A}$ runs in time T, and each other part of $\mathcal{C}$'s computation runs in time $\mathsf{poly}(\lambda)$.

2. $\mathcal{C}$ simulates the PSI game perfectly to $\mathcal{A}$. We see that the public keys that $\mathcal{C}$ provides in Fig. 14 are distributed indentically to Game_0 (in particular: they are independent and uniform in $\mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O})$). The oracles behave identically except for the additional bookkeeping on lines 403–404, which does not affect $\mathcal{A}$'s view. And the challenge signature is generated using Simul instead of Sign, but for our protocol $\mathsf{CSI\text{-}SDVS}$ these algorithms' outputs follow identical distributions, so from $\mathcal{A}$'s perspective nothing changes. This means that the probability that $\mathcal{A}$ makes a query of type described in point 2. above is the same when interacting with $\mathcal{C}$ as it is when playing Game_0.

3. If $\mathcal{C}$ does not output $\perp$, then it outputs a solution to the GAIP instance (X_0, X_1). This is straightforward to see: if $b = 0$ then $\mathcal{C}$ outputs $z^* - z$ where $[\mathfrak{g}^{z^*-z}] * \mathsf{pk}_{S,0} = \mathsf{pk}_{S,1}$, which is precisely a solution to the GAIP instance (X_0, X_1) (recalling that $\mathsf{pk}_{S,b} = x_b$ for $b = 0, 1$). Similarly, if $b = 1$, then $\mathcal{C}$ outputs $z - z^*$, where $[\mathfrak{g}^{z^*-z}] * \mathsf{pk}_{S,1} = \mathsf{pk}_{S,0}$; equivalently, $[\mathfrak{g}^{z-z^*}] * \mathsf{pk}_{S,0} = \mathsf{pk}_{S,1}$, so that $z - z^*$ is the solution to the GAIP instance (X_0, X_1), as before.

Finally, note that $\mathcal{C}$ outputs something other than $\perp$ if and only if $\mathcal{A}$ makes a query to $\mathcal{O}'_{\mathsf{Verify}}$ of the form $(1 - b, m^*, \sigma = (h, z))$ in which $[\mathfrak{g}^{z^*-z}] * \mathsf{pk}_{S,b} = \mathsf{pk}_{S,1-b}$; equivalently, in which $z = z^* + \mathsf{sk}_{S,b} - \mathsf{sk}_{S,1-b}$. This is precisely the type of query whose probability we wished to bound; thus, we have

$$\mathbb{P}[\mathcal{A} \text{ queries } \mathcal{O}'_{\mathsf{Verify}}(1-b, m^*, \sigma = (h, z)), \text{ where } [\mathfrak{g}^{z^*-z}] * \mathsf{pk}_{S,b} = \mathsf{pk}_{S,1-b}] = \mathsf{Adv}^{\mathrm{GAIP},\mathcal{C}}(1^\lambda)$$

as required.

Summing the probabilities of all cases, we have that for any algebraic PSI adversary $\mathcal{A}$ running in time T and which makes at most $q_{\mathcal{H}}$ queries to $\mathcal{H}$, q_{Simul} queries to $\mathcal{O}_{\mathsf{Simul}}$, and q_{Verify} total queries to $\mathcal{O}_{\mathsf{Verify}}$ and $\mathcal{O}'_{\mathsf{Verify}}$, there exists an algorithm $\mathcal{C}$ running in time $T + \mathsf{poly}(\lambda)$ such that

$$\begin{aligned}\mathsf{Adv}^{\mathsf{Game}_0,\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(1^\lambda) \leq \mathsf{Adv}^{\mathsf{Game}_1,\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(1^\lambda) + \mathsf{Adv}^{\mathrm{GAIP},\mathcal{C}}(1^\lambda) + \frac{q_{\mathsf{Simul}} + q_{\mathsf{Verify}}}{N} + \frac{q_{\mathsf{Verify}}}{2^{2\lambda}} \\ + \mathbb{P}[\mathcal{A} \text{ queries } \mathcal{H} \text{ at } ([\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b} \| m^*)].\end{aligned}$$

Finally, we note that Game_0 is the PSI game, and that no adversary—even unbounded—can win Game_1 with probability different from $\frac{1}{2}$, since σ^* and all oracle outputs are independent of b, meaning that $\mathsf{Adv}^{\mathsf{Game}_1,\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(1^\lambda) = 0$. Thus,

$$\begin{aligned}\mathsf{Adv}^{\mathsf{PSI},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(1^\lambda) \leq \mathsf{Adv}^{\mathrm{GAIP},\mathcal{C}}(1^\lambda) + \frac{q_{\mathsf{Simul}} + q_{\mathsf{Verify}}}{N} + \frac{q_{\mathsf{Verify}}}{2^{2\lambda}} \\ + \mathbb{P}[\mathcal{A} \text{ queries } \mathcal{H} \text{ at } ([\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b} \,\|\, m^*)]. \qquad (1)\end{aligned}$$

as required. □

If $\mathcal{A}$ runs in polynomial time and we assume the GAIP problem is hard (so that $\mathsf{Adv}^{\mathrm{GAIP},\mathcal{C}}(1^\lambda) = \mathsf{negl}(\lambda)$), Equation (1) implies that

$$\mathbb{P}[\mathcal{A} \text{ queries } \mathcal{H} \text{ at } ([\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b} \,\|\, m^*)] \geq \mathsf{Adv}^{\mathsf{PSI},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(1^\lambda) - \mathsf{negl}(\lambda).$$

Intuitively: an algebraic adversary that breaks the PSI security of CSI-SDVS with non-negligible advantage makes a query of the form $\mathcal{H}([\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b} \,\|\, m^*)$ non-negligible probability.

We now proceed to prove the second lemma required for our PSI security result.

Lemma 8. *Let $\mathcal{A}$ be a algebraic adversary for the PSI game against* CSI-SDVS *that runs in time T and which makes at most $q_{\mathcal{H}}$ queries to $\mathcal{H}$, q_{Simul} queries to* Simul*, and q_{Verify} queries to $\mathcal{O}_{\mathsf{Verify}}$ and $\mathcal{O}'_{\mathsf{Verify}}$ together. Then there exists an algorithm $\mathcal{E}$ for the GAIP problem that runs in time $T + \mathsf{poly}(\lambda)$, such that*

$$\mathsf{Adv}^{\mathrm{GAIP},\mathcal{E}}(\lambda) \geq \frac{1}{3}\mathsf{Adv}^{\mathsf{PSI},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(\lambda) - \frac{(q+1)^2 + q_{\mathsf{Verify}}}{3 \cdot 2^{2\lambda}} - \frac{q_{\mathsf{Simul}} + q_{\mathsf{Verify}}}{3N}.$$

where $q = q_{\mathcal{H}} + q_{\mathsf{Simul}} + q_{\mathsf{Verify}}$.

Proof. Let $\mathcal{A}$ be PSI adversary as described in the Lemma. We define a GAIP adversary $\mathcal{D}$ in Figs. 15, 16 and 17. □

$\mathcal{D}(X)$	
101 :	$\mathsf{ID} \xleftarrow{\$} \{S, V\}$
102 :	$x \leftarrow \mathcal{D}^{\mathsf{ID}}(X)$
103 :	**return** x

Fig. 15: The reduction $\mathcal{D}$ from GAIP to PSI security of CSI-SDVS in the AGAM.

The proof of Lemma 8 proceeds in a series of claims.

Claim. Let $\mathsf{pp} = (p, \mathfrak{g}, N, E)$ be the public parameters of a GAIP instance X. If $\perp \neq x \leftarrow \mathcal{D}(X)$, then $[\mathfrak{g}^x] * E = X$; that is, x is the solution to the GAIP instance.

Proof. We begin with the case $\mathsf{ID} = S$. We note that in this case, $\mathcal{D}$ will only return $x \neq \perp$ as a result of lines 112 or 113 of Fig. 16. Suppose that $x \neq \perp$ is returned. Then, since the assertion on line 109 has passed, we know that there is a unique $Y \in \mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{D})$ such that

$$L[Y \,\|\, m^*] = h^* = L[[g^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b} \,\|\, m^*]$$

(where the second equality is because h^* is defined by the call to CSI-SDVS.Simul on line 107). Since entries in L are defined only by (direct or indirect) calls to $\mathcal{H}$, this means that $Y = [g^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b}$. Now, because the assertion on line 110 has passed, we know that $\mathcal{A}$ has provided Y as an output or oracle query, and so we have $A \in \{E, \mathsf{pk}_{S,0}, \mathsf{pk}_{S,1}, \mathsf{pk}_V\}$ and

$$[\mathfrak{g}^a] * A = Y = [\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b}. \tag{2}$$

Recall that $\mathsf{pk}_{S,b} = [\mathfrak{g}^{b \cdot r}] * X$. We now consider two cases:

1. If $A = E$, then Equation (2) implies

$$[\mathfrak{g}^a] * E = [\mathfrak{g}^{z^*+\mathsf{sk}_V}] * \mathsf{pk}_{S,b} \implies [\mathfrak{g}^{a-z^*-\mathsf{sk}_V-b\cdot r}] * E = X,$$

so that $\mathcal{D}$'s output $x = a - z^* - \mathsf{sk}_V - b \cdot r$ is a solution to the GAIP instance X.

$\mathcal{D}^S(X)$

101 : $L \leftarrow [\,]$
102 : $r \xleftarrow{\$} \mathbb{Z}_N$
103 : $(\mathsf{pk}_{S,0}, \mathsf{pk}_{S,1}) \leftarrow (X, [\mathfrak{g}^r] * X)$
104 : $(\mathsf{sk}_V, \mathsf{pk}_V) \leftarrow \mathsf{CSI\text{-}SDVS.VKeyGen}(\mathsf{pp})$
105 : $(m^*, \mathsf{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1^{\mathcal{H}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}}}(\mathsf{pk}_{S,0}, \mathsf{pk}_{S,1}, \mathsf{pk}_V)$
106 : $b \xleftarrow{\$} \{0,1\}$
107 : $\sigma^* = (h^*, z^*) \leftarrow \mathsf{CSI\text{-}SDVS.Simul}(\mathsf{sk}_V, \mathsf{pk}_{S,b}, m^*)$
108 : $b' \leftarrow \mathcal{A}_2^{\mathcal{H}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}'_{\mathsf{Verify}}}(\mathsf{pk}_{S,0}, \mathsf{pk}_{S,1}, \mathsf{pk}_V, \mathsf{st}_{\mathcal{A}})$
109 : **assert** $\exists!\ Y \in \mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O})$ s.t. $h^* = L[Y \,\|\, m^*]$
110 : **assert** $\mathsf{GetRep}(Y) \neq \perp$
111 : $(a, A) \leftarrow \mathsf{GetRep}(Y)$
112 : **if** $A = E$: **return** $a - z^* - \mathsf{sk}_V - b \cdot r$
113 : **if** $A = \mathsf{pk}_V$: **return** $a - z^* - b \cdot r$
114 : **return** $\perp$

$\mathcal{O}_{\mathsf{Simul}}(i, m)$

201 : $z \xleftarrow{\$} \mathbb{Z}_N$
202 : $h \leftarrow \mathcal{H}([g^{z+\mathsf{sk}_V}] * \mathsf{pk}_{S,i} \,\|\, m)$
203 : $\sigma \leftarrow (h, z)$
204 : **return** σ

$\mathcal{H}(x)$

301 : **if** $L[x] = \perp$
302 : $\quad L[x] \xleftarrow{\$} \{0,1\}^\lambda$
303 : **return** $L[x]$

$\mathcal{O}_{\mathsf{Verify}}(i, m, \sigma)$ $\mathcal{O}'_{\mathsf{Verify}}(i, m, \sigma)$

401 : **assert** $(m, \sigma) \neq (m^*, \sigma^*)$
402 : **parse** $(h, z) \leftarrow \sigma$
403 : $h' \leftarrow \mathcal{H}([\mathfrak{g}^{z+\mathsf{sk}_V}] * \mathsf{pk}_{S,i} \,\|\, m)$
404 : **return** $[h = h']$

Fig. 16: The reduction from GAIP to PSI security of CSI-SDVS when the GAIP instance X is embedded in $\mathsf{pk}_{S,0}$ and $\mathsf{pk}_{S,1}$.

$\mathcal{D}^V(X)$

101 : $L \leftarrow [\,]$
102 : **for** $i = 0, 1$
103 : $\quad (\mathsf{sk}_{S,i}, \mathsf{pk}_{S,i}) \leftarrow \mathsf{CSI\text{-}SDVS.SKeyGen}(\mathsf{pp})$
104 : $\mathsf{pk}_V \leftarrow X$
105 : $(m^*, \mathsf{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1^{\mathcal{H}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}}}(\mathsf{pk}_{S,0}, \mathsf{pk}_{S,1}, \mathsf{pk}_V)$
106 : $b \xleftarrow{\$} \{0,1\}$
107 : $\sigma^* = (h^*, z^*) \leftarrow \mathsf{CSI\text{-}SDVS.Sign}(\mathsf{sk}_{S,b}, \mathsf{pk}_V, m^*)$
108 : $b' \leftarrow \mathcal{A}_2^{\mathcal{H}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}'_{\mathsf{Verify}}}(\mathsf{pk}_{S,0}, \mathsf{pk}_{S,1}, \mathsf{pk}_V, \mathsf{st}_{\mathcal{A}})$
109 : **assert** $\exists!\ Y \in \mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O})$ s.t. $h^* = L[Y \,\|\, m^*]$
110 : **assert** $\mathsf{GetRep}(Y) \neq \perp$
111 : $(a, A) \leftarrow \mathsf{GetRep}(Y)$
112 : **if** $A = E$: **return** $a - z^* - \mathsf{sk}_{S,b}$
113 : **if** $A = \mathsf{pk}_{S,b}$: **return** $a - z^*$
114 : **if** $A = \mathsf{pk}_{S,1-b}$: **return** $a - z^* - \mathsf{sk}_{S,b} + \mathsf{sk}_{S,1-b}$
115 : **return** $\perp$

$\mathcal{O}_{\mathsf{Simul}}(i, m)$

201 : $z \xleftarrow{\$} \mathbb{Z}_N$
202 : $h \leftarrow \mathcal{H}([g^{z+\mathsf{sk}_{S,i}}] * \mathsf{pk}_V \,\|\, m)$
203 : $\sigma \leftarrow (h, z)$
204 : **return** σ

$\mathcal{H}(x)$

301 : **if** $L[x] = \perp$
302 : $\quad L[x] \xleftarrow{\$} \{0,1\}^\lambda$
303 : **return** $L[x]$

$\mathcal{O}_{\mathsf{Verify}}(i, m, \sigma)$ $\mathcal{O}'_{\mathsf{Verify}}(i, m, \sigma)$

401 : **assert** $(m, \sigma) \neq (m^*, \sigma^*)$
402 : **parse** $(h, z) \leftarrow \sigma$
403 : $h' \leftarrow \mathcal{H}([\mathfrak{g}^{z+\mathsf{sk}_{S,i}}] * \mathsf{pk}_V \,\|\, m)$
404 : **return** $[h = h']$

Fig. 17: The reduction from GAIP to PSI security of CSI-SDVS when the GAIP instance X is embedded in pk_V.

2. If $A = \mathsf{pk}_V$, then Equation (2) implies

$$[\mathfrak{g}^a] * \mathsf{pk}_V = [\mathfrak{g}^{z^* + \mathsf{sk}_V}] * \mathsf{pk}_{S,b} \implies [\mathfrak{g}^{a - z^* - b \cdot r}] * E = X,$$

so that $\mathcal{D}$'s output $x = a - z^* - b \cdot r$ is a solution to the GAIP instance X.

When $\mathsf{ID} = V$ the argument is similar. $\mathcal{D}$ will only return $x \neq \perp$ as a result of lines 112, 113, or 114 of Fig. 17. Suppose that $x \neq \perp$ is returned. Then, since the assertion on line 109 has passed, we know that there is a unique $Y \in \mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O})$ such that

$$L[Y \,\|\, m^*] = h^* = L[[g^{z^* + \mathsf{sk}_{S,b}}] * \mathsf{pk}_V \,\|\, m^*]$$

(where the second equality is because h^* is defined by the call to CSI-SDVS.Sign on line 107). Since entries in L are defined only by (direct or indirect) calls to $\mathcal{H}$, this means that $Y = [g^{z^* + \mathsf{sk}_{S,b}}] * \mathsf{pk}_{S,b}$. Now, because the assertion on line 110 has passed, we know that $\mathcal{A}$ has provided Y as an output or oracle query, and so we have $A \in \{E, \mathsf{pk}_{S,0}, \mathsf{pk}_{S,1}, \mathsf{pk}_V\}$ and

$$[\mathfrak{g}^a] * A = Y = [\mathfrak{g}^{z^* + \mathsf{sk}_{S,b}}] * \mathsf{pk}_V. \tag{3}$$

Recall that $\mathsf{pk}_V = X$. We now consider three cases:

1. If $A = E$, then Equation (3) implies

$$[\mathfrak{g}^a] * E = [\mathfrak{g}^{z^* + \mathsf{sk}_{S,b}}] * \mathsf{pk}_V \implies [\mathfrak{g}^{a - z^* - \mathsf{sk}_{S,b}}] * E = X,$$

so that $\mathcal{D}$'s output $x = a - z^* - \mathsf{sk}_{S,b}$ is a solution to the GAIP instance X.

2. If $A = \mathsf{pk}_{S,b}$, then Equation (3) implies

$$[\mathfrak{g}^a] * \mathsf{pk}_{S,b} = [\mathfrak{g}^{z^* + \mathsf{sk}_{S,b}}] * \mathsf{pk}_V \implies [\mathfrak{g}^{a - z^*}] * E = X,$$

so that $\mathcal{D}$'s output $x = a - z^*$ is a solution to the GAIP instance X.

3. If $A = \mathsf{pk}_{S,1-b}$, then Equation (3) implies

$$[\mathfrak{g}^a] * \mathsf{pk}_{S,1-b} = [\mathfrak{g}^{z^* + \mathsf{sk}_{S,b}}] * \mathsf{pk}_V \implies [\mathfrak{g}^{a - z^* - \mathsf{sk}_{S,b} + \mathsf{sk}_{S,1-b}}] * E = X,$$

so that $\mathcal{D}$'s output $x = a - z^* - \mathsf{sk}_{S,b} + \mathsf{sk}_{S,1-b}$ is a solution to the GAIP instance X.

This completes the proof of the claim. □

Claim. $\mathcal{D}^S$ and $\mathcal{D}^V$ perfectly simulate the SUF-CMA game to $\mathcal{A}$. In particular: fix public parameters $(p, [\mathfrak{g}], N, E, \mathcal{H})$ for CSI-SDVS. Let $\mathrm{View}(\mathcal{A})$ denote the adversary's *view* in the PSI game—more precisely, $\mathrm{View}(\mathcal{A})$ is the joint distribution of the public values $\mathsf{pk}_S, \mathsf{pk}_V$ and all oracle responses given to $\mathcal{A}$ in the course of the PSI experiment. Similarly, let $\mathrm{View}_S(\mathcal{A})$ and $\mathrm{View}_V(\mathcal{A})$ denote the adversary's view in the reductions $\mathcal{D}^S$ and $\mathcal{D}^V$, respectively, when X is a uniformly-chosen GAIP instance. Then, the distributions $\mathrm{View}(\mathcal{A})$, $\mathrm{View}_S(\mathcal{A})$, and $\mathrm{View}_V(\mathcal{A})$ are identical.

The proof of this claim is similar to the analogous claim in the proof of SUF-CMA security in the AGAM, and so we omit it. As a consequence of the claim, the adversary's behaviour in the PSI experiment is identical to its behaviour when interacting with $\mathcal{D}$.

Claim. Let $\mathcal{A}$ be an algebraic adversary for the PSI game against CSI-SDVS. Suppose that $\mathcal{A}$ runs in time T and makes at most q total queries to $\mathcal{H}, \mathcal{O}_{\mathsf{Simul}}, \mathcal{O}_{\mathsf{Verify}}$, and $\mathcal{O}'_{\mathsf{Verify}}$. Define

$$\mathsf{Succ}^{\mathcal{D}^{\mathcal{A}}}(\lambda) = \mathbb{P}\left[x' \neq \perp \;\middle|\; \begin{array}{rl} (p, \mathfrak{g}, N, E, \mathcal{H}) & \leftarrow \mathsf{Setup}(1^\lambda) \\ x & \xleftarrow{\$} \mathbb{Z}_N \\ X & \leftarrow [\mathfrak{g}^x] * E \\ x' & \leftarrow \mathcal{D}^{\mathcal{A}}(X) \end{array} \right].$$

Then there exists a GAIP algorithm $\mathcal{D}$ running in time $T + \mathsf{poly}(\lambda)$ such that

$$\mathsf{Succ}^{\mathcal{D}^{\mathcal{A}}}(\lambda) \geq \frac{1}{2}\mathsf{Adv}^{\mathsf{PSI},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(\lambda) - \frac{1}{2}\mathsf{Adv}^{\mathrm{GAIP},\mathcal{C}}(\lambda) - \frac{(q+1)^2 + q_{\mathsf{Verify}}}{2^{2\lambda+1}} - \frac{q_{\mathsf{Simul}} + q_{\mathsf{Verify}}}{2N}.$$

Proof. Since ID is sampled uniformly at random from $\{S, V\}$ in the execution of $\mathcal{D}^{\mathcal{A}}$, we have

$$\mathsf{Succ}^{\mathcal{D}^{\mathcal{A}}}(\lambda) = \frac{1}{2}\mathbb{P}[\mathcal{D}^{\mathcal{A}}(X) \neq \perp | \mathsf{ID} = S] + \frac{1}{2}\mathbb{P}[\mathcal{D}^{\mathcal{A}}(X) \neq \perp | \mathsf{ID} = V] \tag{4}$$

and so it suffices to analyze the success probability of $\mathcal{D}^S$ and $\mathcal{D}^V$ individually. Beginning with $\mathcal{D}^S$ and referring to Fig. 16, we see that $\mathcal{D}^S$ succeeds as long as:

1. The assertion on line 109 passes.
2. The assertion on line 110 passes.
3. When (a, A) is constructed on line 111, we have $A \notin \{\mathsf{pk}_{S,0}, \mathsf{pk}_{S,1}\}$.

For the first condition, first observe that some such Y must exist, because it is defined by the call to CSI-SDVS.Simul on line 107 (if not earlier). By the birthday bound, the probability that two or more such values of Y satisfy the condition is at most $\frac{(q+1)^2}{2^{2\lambda}}$, since at most $q+1$ hash values are defined (q because of $\mathcal{A}$'s queries to the various oracles, and 1 by the challenge signature construction on line 107).

For the second condition, Lemma 7 guarantees that $\mathcal{A}$ makes a query to $\mathcal{H}$ at $(Y \,\|\, m^*)$ with $Y = [\mathfrak{g}^{z^* + \mathsf{sk}_V}] * \mathsf{pk}_{S,b}$ with probability at least

$$\mathsf{Adv}^{\mathsf{PSI},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(1^\lambda) - \mathsf{Adv}^{\mathrm{GAIP},\mathcal{C}}(1^\lambda) - \frac{q_{\mathsf{Simul}} + q_{\mathsf{Verify}}}{N} - \frac{q_{\mathsf{Verify}}}{2^{2\lambda}} \tag{5}$$

where $\mathcal{C}$ is a GAIP adversary running in time $T + \mathsf{poly}(\lambda)$. Given that the assertion on line 109 passes, $\mathcal{A}$ making such a query will ensure that the assertion on line 110 passes, since the assertion on line 109, combined with the definition of the challenge signature on line 107 guarantees that the only Y such that $h^* = L[Y \,\|\, m^*]$ is precisely

$$Y = [\mathfrak{g}^{z^* + \mathsf{sk}_{S,b}}] * \mathsf{pk}_V = [\mathfrak{g}^{z^* + \mathsf{sk}_V}] * \mathsf{pk}_{S,b}$$

where the first equality from the definition of CSI-SDVS.Sign and the construction of σ^* on line 107, and the second comes from the commutativity of the group action. Thus, the probability that both assertions pass is at least

$$\mathsf{Adv}^{\mathsf{PSI},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(1^\lambda) - \mathsf{Adv}^{\mathrm{GAIP},\mathcal{C}}(1^\lambda) - \frac{q_{\mathsf{Simul}} + q_{\mathsf{Verify}}}{N} - \frac{q_{\mathsf{Verify}}}{2^{2\lambda}} - \frac{(q+1)^2}{2^{2\lambda}}.$$

When both assertions pass, we know that $A \in \{E, \mathsf{pk}_{S,0}, \mathsf{pk}_{S,1}, \mathsf{pk}_V\}$. Let $p_E, p_{S,0}, p_{S,1}$ and, p_V denote the probability that, when both assertions pass, that $A = E, \mathsf{pk}_{S,0}, \mathsf{pk}_{S,1}$, and pk_V, respectively. Then

$$\begin{aligned}\mathbb{P}[\text{Assertions pass}] &= p_E + p_{S,0} + p_{S,1} + p_V \\ &\geq \mathsf{Adv}^{\mathsf{PSI},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(1^\lambda) - \mathsf{Adv}^{\mathrm{GAIP},\mathcal{C}}(1^\lambda) - \frac{q_{\mathsf{Simul}} + q_{\mathsf{Verify}}}{N} - \frac{(q+1)^2 + q_{\mathsf{Verify}}}{2^{2\lambda}}.\end{aligned}$$

When computing $\mathbb{P}[\mathcal{D}^{\mathcal{A}}(1^\lambda) \neq \perp | \mathsf{ID} = S]$, we must subtract $p_{S,0} + p_{S,1}$, since the corresponding values of A do not allow $\mathcal{D}^S$ to succeed. What remains is

$$\mathbb{P}[\mathcal{D}^{\mathcal{A}}(1^\lambda) \neq \perp | \mathsf{ID} = S] = p_E + p_V.$$

A similar argument yields

$$\mathbb{P}[\mathcal{D}^{\mathcal{A}}(1^\lambda) \neq \perp | \mathsf{ID} = S] = p_E + p_{S,0} + p_{S,1}$$

where, as in the AGAM proof of SUF-CMA security, the probabilities $p_E, p_V, p_{S,0}, p_{S,1}$ do not change between $\mathcal{D}^S$ and $\mathcal{D}^V$. By Equations (4) and (5), we have

$$\begin{aligned}\mathsf{Succ}^{\mathcal{D}^{\mathcal{A}}}(\lambda) &= p_E + \frac{1}{2}(p_{S,0} + p_{S,1} + p_V) \\ &\geq \frac{1}{2}\mathsf{Adv}^{\mathsf{PSI},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(\lambda) - \frac{1}{2}\mathsf{Adv}^{\mathrm{GAIP},\mathcal{C}}(\lambda) - \frac{(q+1)^2 + q_{\mathsf{Verify}}}{2^{2\lambda+1}} - \frac{q_{\mathsf{Simul}} + q_{\mathsf{Verify}}}{2N}\end{aligned} \tag{6}$$

as required. □

Finally, since $\mathsf{Succ}^{\mathcal{D}^{\mathcal{A}}}(\lambda) = \mathsf{Adv}^{\mathrm{GAIP},\mathcal{D}}(\lambda)$. Equation (6) implies that

$$\mathsf{Adv}^{\mathrm{GAIP},\mathcal{D}}(\lambda) + \frac{1}{2}\mathsf{Adv}^{\mathrm{GAIP},\mathcal{C}}(\lambda) \geq \frac{1}{2}\mathsf{Adv}^{\mathsf{PSI},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(\lambda) - \frac{(q+1)^2 + q_{\mathsf{Verify}}}{2^{2\lambda+1}} - \frac{q_{\mathsf{Simul}} + q_{\mathsf{Verify}}}{2N}$$

Taking the maximum of $\mathsf{Adv}^{\mathrm{GAIP},\mathcal{D}}(\lambda)$ and $\mathsf{Adv}^{\mathrm{GAIP},\mathcal{C}}(\lambda)$, and calling the corresponding algorithm $\mathcal{E}$, we have

$$\begin{aligned}\frac{3}{2}\mathsf{Adv}^{\mathrm{GAIP},\mathcal{E}}(\lambda) &\geq \frac{1}{2}\mathsf{Adv}^{\mathsf{PSI},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(\lambda) - \frac{(q+1)^2 + q_{\mathsf{Verify}}}{2^{2\lambda+1}} - \frac{q_{\mathsf{Simul}} + q_{\mathsf{Verify}}}{2N} \\ \implies \mathsf{Adv}^{\mathrm{GAIP},\mathcal{E}}(\lambda) &\geq \frac{1}{3}\mathsf{Adv}^{\mathsf{PSI},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(\lambda) - \frac{(q+1)^2 + q_{\mathsf{Verify}}}{3 \cdot 2^{2\lambda}} - \frac{q_{\mathsf{Simul}} + q_{\mathsf{Verify}}}{3N}.\end{aligned}$$

Proof (of Theorem 6). Under the GAIP assumption, if $\mathcal{A}$ is polynomial-time, $\mathsf{Adv}^{\mathrm{GAIP},\mathcal{E}}(\lambda)$, and $\frac{(q+1)^2+q_{\mathsf{Verify}}}{3\cdot 2^{2\lambda}} + \frac{q_{\mathsf{Simul}}+q_{\mathsf{Verify}}}{3N}$ are negligible. Therefore, Lemma 8 forces that $\mathsf{Adv}^{\mathsf{PSI},\mathcal{A}}_{\mathsf{CSI\text{-}SDVS}}(\lambda)$ to be negligible. □

6 Performance

Table 1 summarizes the size complexity of our protocol when instantiated with the CSIDH-512 parameter set.

Secret Keys: The secret keys sk_S and sk_V are sampled uniformly from $\mathbb{Z}_N$, where $\log_2 N = 2\lambda$. Hence, each secret key requires 2λ bits.

Public Keys: Each elliptic curve in $\mathcal{E}\ell\ell_{\mathbb{F}_p}(\mathfrak{O})$ is represented in Montgomery form $E_A : y^2 = x^3 + Ax^2 + x$, where $A \in \mathbb{F}_p$. Since p is a 4λ-bit prime, each public key consists of a single field element in $\mathbb{F}_p$, which requires 4λ bits.

Signatures: A signature $\sigma = (h, z)$ consists of a 2λ-bit hash value and an element $z \in \mathbb{Z}_N$, where the hash output length is chosen to match the order of $\mathbb{Z}_N$. Hence, the total signature size is 4λ bits.

Table 1: Key and signature sizes (in bits) using the CSIDH-512 parameter set.

Scheme	Secret Key	Public Key	Signature	Security
Sun et al. [26]*	432	2 640	256	SS-DDH, ROM
Khuc et al. [12]	128	131 072	6 896	CSI-CDH, ROM
CSI-SDVS	**256**	**512**	**512**	**CSI-DDH, ROM** or **GAIP, AGAM, ROM**

Note. *This scheme is insecure; ROM: Random Oracle Model; GAIP: Group Action Inverse Problem; CSI-CDH: Commutative Supersingular Isogeny-Computational Diffie–Hellman Problem; SS-CDH: Supersingular Computational Diffie–Hellman Problem; SS-DDH: Supersingular Decisional Diffie–Hellman Problem.

While prior schemes exhibit large public keys or signatures, CSI-SDVS achieves balanced $\mathcal{O}(\lambda)$-size representations across all components.

References

[1] Jakobsson, M., Sako, K., Impagliazzo, R.: Designated verifier proofs and their applications. In: International Conference on the Theory and Applications of Cryptographic Techniques, pp. 143–154 (1996)

[2] Chaum, D.: Private signature and proof systems. Google Patents, US Patent 5,493,614 (1996)

[3] Laguillaumie, F., Vergnaud, D.: Designated verifier signatures: Anonymity and efficient construction from any bilinear map. In: International Conference on Security in Communication Networks, pp. 105–119 (2004)

[4] Saeednia, S., Kremer, S., Markowitch, O.: An efficient strong designated verifier signature scheme. In: International Conference on Information Security and Cryptology, pp. 40–54 (2003)

[5] Huang, X.-Y., Susilo, W., Mu, Y., Zhang, F.-T.: Short designated verifier signature scheme and its identity-based variant. International Journal of Network Security 6(1), 82–93 (2008)

[6] Kang, B., Boyd, C., Dawson, E.D.: A novel identity-based strong designated verifier signature scheme. Journal of Systems and Software 82(2), 270–273 (2009)

[7] Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Review 41(2), 303–332 (1999)

[8] Asaar, M.R.: Code-based strong designated verifier signatures: Security analysis and a new construction. Cryptology ePrint Archive (2016)

[9] Ren, Y., Wang, H., Du, J., Ma, L.: Code-based authentication with designated verifier. International Journal of Grid and Utility Computing 7(1), 61–67 (2016)

[10] Shooshtari, M.K., Ahmadian-Attari, M., Aref, M.R.: Provably secure strong designated verifier signature scheme based on coding theory. International Journal of Communication Systems 30(7), e3162 (2017)

[11] Thanalakshmi, P., Anitha, R.: A new code-based designated verifier signature scheme. International Journal of Communication Systems 31(17), e3803 (2018)

[12] Khuc, X.T., Susilo, W., Duong, D.H., Li, Y., Roy, P.S., Fukushima, K., Kiyomoto, S.: Strong designated verifier signatures from isogeny assumptions. In: International Conference on Cryptology and Network Security, pp. 46–69 (2025)

[13] Duman, J., Hartmann, D., Kiltz, E., Kunzweiler, S., Lehmann, J., Riepel, D.: Generic models for group actions. In: IACR International Conference on Public-Key Cryptography, pp. 406–435 (2023)

[14] Fuchsbauer, G., Kiltz, E., Loss, J.: The algebraic group model and its applications. In: Annual International Cryptology Conference, pp. 33–62 (2018)

[15] Jao, D., De Feo, L.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In: International Workshop on Post-Quantum Cryptography, pp. 19–34 (2011)

[16] Castryck, W., Decru, T.: An efficient key recovery attack on SIDH. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 423–447 (2023)

[17] Maino, L., Martindale, C., Panny, L., Pope, G., Wesolowski, B.: A direct key recovery attack on SIDH. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 448–471 (2023)

[18] Robert, D.: Breaking SIDH in polynomial time. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 472–503 (2023)

[19] Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: CSIDH: An efficient post-quantum commutative group action. In: Advances in Cryptology–ASIACRYPT 2018: 24th International Conference on the Theory and Application of Cryptology and Information Security, pp. 395–427 (2018)

[20] Dartois, P., Eriksen, J.K., Fouotsa, T.B., Herlédan Le Merdy, A., Invernizzi, R., Robert, D., Rueger, R., Vercauteren, F., Wesolowski, B.: PEGASIS: Practical effective class group action using 4-dimensional isogenies. In: Annual International Cryptology Conference, pp. 67–99 (2025)

[21] Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing, pp. 197–206 (2008)

[22] Wang, F., Hu, Y., Wang, B.: Lattice-based strong designate verifier signature and its applications. Malaysian Journal of Computer Science 25(1), 11–22 (2012)

[23] Noh, G., Jeong, I.R.: Strong designated verifier signature scheme from lattices in the standard model. Security and Communication Networks 9(18), 6202–6214 (2016)

[24] Cai, J., Jiang, H., Zhang, P., Zheng, Z., Lyu, G., Xu, Q.: An efficient strong designated verifier signature based on $\mathcal{R}$-SIS assumption. IEEE Access 7, 3938–3947 (2019)

[25] Zhang, Y., Susilo, W., Guo, F.: Lattice-based strong designated verifier signature with non-delegatability. Computer Standards & Interfaces 92, 103904 (2025)

[26] Sun, X., Tian, H., Wang, Y.: Toward quantum-resistant strong designated verifier signature from isogenies. In: 2012 Fourth International Conference on Intelligent Networking and Collaborative Systems, pp. 292–296 (2012)

[27] Washington, L.C.: Elliptic Curves: Number Theory and Cryptography. Chapman and Hall/CRC, Boca Raton, FL (2008)

[28] Silverman, J.H.: The Arithmetic of Elliptic Curves. Springer, New York, NY, vol. 106 (2009)

[29] Alamati, N., De Feo, L., Montgomery, H., Patranabis, S.: Cryptographic group actions and applications. In: International Conference on the Theory and Application of Cryptology and Information Security, pp. 411–439 (2020)