





# Simple Two-Message OT in the Explicit Isogeny Model

Emmanuela Orsini<sup>a,1</sup>  and Riccardo Zanotto<sup>b,2</sup> 

<sup>1</sup> Bocconi University, Department of Computing Sciences, Milan, Italy

<sup>2</sup> CISPA Helmholtz Center for Information Security, Saarbrücken, Germany

**Abstract.** In this work we study algebraic and generic models for group actions, and extend them to the universal composability (UC) framework of Canetti (FOCS 2001). We revisit the constructions of Duman et al. (PKC 2023) integrating the type-safe model by Zhandry (Crypto 2022), adapted to the group action setting, and formally define an *algebraic action model* (AAM). This model restricts the power of the adversary in a similar fashion to the algebraic group model (AGM). By imposing algebraic behaviour to the adversary and environment of the UC framework, we construct the UC-AAM. Finally, we instantiate UC-AAM with isogeny-based assumptions, in particular the CSIDH action with twists, obtaining the *explicit isogeny* model, UC-EI; we observe that, under certain assumptions, this model is “closer” to standard UC than the UC-AGM, even though there still exists an important separation.

We demonstrate the utility of our definitions by proving UC-EI security for the passive-secure oblivious transfer protocol described by Lai et al. (Eurocrypt 2021), hence providing the first concretely efficient two-message isogeny-based OT protocol in the random oracle model against malicious adversaries.

**Keywords:** Oblivious Transfer · Isogenies · CSIDH · Group Action Model

## 1 Introduction

Oblivious transfer (OT), introduced by Rabin [Rab05], is a fundamental cryptographic primitive that plays a central role in modern cryptography. In particular, OT is both sufficient and necessary for secure two-party and multiparty computation, and it is widely deployed in a number of efficient protocols [BLN<sup>+</sup>21, KOS16] and applications ranging from private set intersection [DCW13, PSZ14] to contract signing [EGL82]. The most commonly studied form of oblivious transfer is 1-out-of-2 OT, where a sender  $P_S$  holds two messages  $m_0, m_1$  and a receiver  $P_R$  holds a bit  $b$  corresponding to the sender’s message  $m_b$  that it will receive as output of the protocol. The security requirement is that  $P_R$  should obtain  $m_b$  without learning any information about the other message  $m_{1-b}$  and  $P_S$  should learn nothing about the choice bit  $b$ . Oblivious transfer can be constructed from various assumptions: number-theoretic assumptions like Decisional Diffie-Hellman (DDH) [BM90, NP01, AIR01, PVW08, ZLWR13, CO15], and quadratic-residuosity (QR) [HK12]; and also from (presumed) post-quantum assumptions like coding-theory

---

E-mail: [emmanuela.orsini@unibocconi.it](mailto:emmanuela.orsini@unibocconi.it) (Emmanuela Orsini), [riccardo.zanotto@cispa.de](mailto:riccardo.zanotto@cispa.de) (Riccardo Zanotto)

<sup>a</sup>Emmanuela Orsini was supported by CyberSecurity Research Flanders with reference number VR20192203. Her work was primarily carried out while she was affiliated with imec-COSIC.

<sup>b</sup>Funded by the European Union (ERC, LACONIC, 101041207). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.



related assumptions [DvMN08, DDN14, DGH<sup>+</sup>20], lattice-based assumptions [PVW08, BD18, MS20], and isogeny-based assumptions [Vit19, BDGM19, DOPS20, LGD21].

*Security and complexity.* The efficiency of oblivious transfer protocols is a very important metric, as it directly impacts the efficiency of larger cryptographic systems in which OT is used; in addition, even if there exists a variety of security models for OT, ideally we would like to achieve security in the *universal composability* (UC) framework described by Canetti [Can01], where security is maintained under concurrent general composition. Unfortunately, achieving efficient UC-secure OT protocols is not an easy task, especially if we want a protocol with a low number of messages exchanged by the two parties. A straightforward approach to achieve fully secure OT is by using zero-knowledge proofs to transform a passively secure protocol into an active one. However, this technique often increases the number of interactions, which may not be desirable if the goal is to design a protocol with few interactions, such as a two-message protocol, where  $P_R$  sends a single “request” message, and  $P_S$  answers with a single “response” message. Therefore, alternative approaches are necessary, focusing on two-message protocols that are already actively secure.

A first barrier is given by the impossibility result of Goldreich and Oren [GO94], which states that achieving two-message OT with simulation-based security is impossible in the plain model; therefore, it is necessary to rely on setup assumptions such as a *common reference string* (CRS) or a *random oracle* (RO). In a simulation-based proof, the main challenge is to extract the receiver’s input bit and then argue that the message  $m_{1-b}$  remains hidden.

When considering (presumed) post-quantum assumptions exclusively, only a few constructions for two-message maliciously secure OT are known. The first one is the general framework described by Peikert, Vaikuntanathan and Waters [PVW08], where they instantiate dual mode cryptosystems from LWE (and also DDH and QR), and obtain an efficient *bit* OT. We also have OT from LPN via the compiler described by Döttling et al. [DGH<sup>+</sup>20], but this construction is more involved and requires garbled circuits.

Hence, the following natural question arises:

*Can we describe a concretely efficient UC-secure two-message oblivious transfer protocol from (presumed) post-quantum secure assumptions, and in particular from isogeny-based assumptions?*

*Why isogeny-based OT.* Isogeny-based cryptography is a very recent and active field from which there have been many proposed schemes for post-quantum secure key exchange, public key encryption, signatures and much more. One of the main advantage of these constructions, compared to other post-quantum constructions, is their relatively small keys.

The first isogeny-based cryptosystem was given by Couveignes [Cou06] and Rostovtsev and Stolbunov [RS06]. They described a non-interactive key-exchange protocol based on the theory of complex multiplication of ordinary elliptic curves over  $\mathbb{F}_q$ , whose endomorphism ring is a given quadratic order  $\mathcal{O}$ . In particular, they observed that the commutative action of the ideal class group  $Cl(\mathcal{O})$  naturally leads to a key-exchange procedure à la Diffie-Hellman. Later, Jao and De Feo [JD11] proposed the SIDH key-exchange scheme using supersingular elliptic curves. Unfortunately, this scheme and related variants, like the NIST candidate SIKE [JAC<sup>+</sup>20], have been recently broken [CD23, MMP<sup>+</sup>23, Rob23] using Kani’s reducibility criterion [Kan97]. These attacks have no effect on another class of isogeny-based schemes, namely CSIDH-based constructions.

CSIDH stands for Commutative-SIDH and it was introduced by Castryck et al. [CLM<sup>+</sup>18], as an adaptation of the original Couveignes-Rostovtsev-Stolbunov protocol with supersingular curves, and is now one of the main tools of isogeny-based cryptography.

In particular, it sparked new interest in the general framework of group actions, which can be seen as a generalization of the exponentiation property of Diffie-Hellman.

The challenging question, which we will address, is then if it is possible to construct UC-secure efficient OT protocols from the CSIDH group action, similarly to how the Chou and Orlandi [CO15] protocol can be seen as a variation of two DDH-based key-exchanges.

## 1.1 Our Contributions

In this work, we address the question above by focusing on the isogeny-based two-message OT protocol described by Lai et al. [LGD21] (Figure 2), which was originally proven secure only against semi-honest adversaries.

We first introduce a general framework for group action-based cryptography, and formally define an Algebraic Action Model, following [Zha22] and [DHK<sup>+</sup>23]. Inspired by the UC-AGM described by Abdalla et al. [ABK<sup>+</sup>21], we then extend the Algebraic Action Model to the UC framework, obtaining the UC-AAM. Although our UC-AAM is actually less expressive than standard UC, as it is for the UC-AGM, we provide some evidence to suggest that our UC-AAM might be “closer” to plain UC, and we believe that this model can be of independent interest.

We then specialize the UC-AAM to the context of CSIDH and refer to it as the *UC Explicit-Isogeny* (UC-EI) model; this new model incorporates twists into a slightly bigger group action, offering a novel perspective on the CSIDH action. Using the UC-EI model, we finally show how to prove malicious security for the two-message protocol described by Lai et al. [LGD21]. We conclude with a thoughtful comparison between several existing constructions of OT protocols from isogeny-based assumptions in Section 5.3.

## 1.2 Technical overview

*Background.* In [CO15], Chou and Orlandi proposed a simple and efficient OT protocol, hereafter known as the CO protocol (we refer to Section B.1 for further details), based on DDH in the random oracle model. The protocol follows the Bellare-Micali [BM90] paradigm, where there is a public element  $C \in \mathbb{Z}_p^*$  with unknown discrete logarithm relative to a generator  $g$ , and the receiver creates two public keys  $(g^r, C/g^r)$  so that they can only know the secret key relative to one of them.

This scheme has been used as a blueprint for many subsequent constructions based on different hardness assumptions. However, proving its UC security is problematic: the main issue is that a corrupt receiver might never query the random oracle to get the decryption key for the ciphertext, and in this case the simulator cannot extract the receiver’s choice bit and thus cannot query the functionality for the corresponding sender’s message and finish the simulation.

One solution to this problem is to require some sort of “proof of decryption”: some protocols, like [BDD<sup>+</sup>17], add one or two messages between the parties to allow the simulator to extract and to complete the simulation of the corresponding functionality. This same approach was taken by Lai et al. [LGD21] to describe a concretely efficient 4-message isogeny-based UC-secure oblivious transfer.

Another solution, proposed by Abdalla et al. [ABK<sup>+</sup>21], is given by using a different model of computation, in which the extraction of the input bit is made almost automatic by the behaviour imposed on the adversary by the model. More concretely, the authors introduced the UC-AGM, obtained by extending the *algebraic group model* (AGM) within the UC framework, and showed that some important protocols, like CO, can be proven secure in this model. We recall that the AGM, introduced by Fuchsbauer, Kiltz and Loss [FKL18], takes the middle ground between the standard model and the *generic group model* (GGM), which has been proposed in different and not necessarily equivalent variants, like the ones described by Shoup [Sho97] and Maurer [Mau05].

Roughly, adversaries in the AGM are allowed to see and use the structure of the group, but they are required to behave “algebraically”, i.e. obtain new group elements only via group multiplications from known elements. More precisely, an adversary  $\mathcal{A}$  is *algebraic* if for every  $h \in G$  that it outputs, it also provides a vector of exponents  $(x_1, \dots, x_n)$  such that  $h = \prod_{i=1}^n g_i^{x_i}$ , where the  $g_i$  are all the group elements already seen by the adversary  $\mathcal{A}$ .

*Drawbacks of the AGM.* While the AGM (and UC-AGM) enables easier proofs of security for protocols, the model also shows some limitations and gives rise to concerns. For example, in the AGM we are not able to sample group elements without access to an external oracle.

Another important issue, highlighted concurrently by Katz et al. [ZZK22] and Zhandry [Zha22], is that the AGM is incomparable to Shoup’s GGM, due to the not fully formal definition of the AGM. Katz et al. focus on the problems of encoding group elements and what it means to “output a group element”; in particular, the authors prove that an algebraic adversary can be used to reduce a generic-hard problem to a generic-easy problem, thus showing a key weakness in the AGM’s definition. In the paper by Zhandry, a solution to this issue is proposed by formally defining the AGM as a compiler for games in the *Type-Safe* (TS) generic-group model, which is a variant of Maurer’s GGM; the TS model restricts the type of games that can be described, and among those the AGM requirement is met. Additional issues are encountered when one tries to combine computational models and the UC framework; for example, [Zha22] showed that security in the TS model is equivalent to security in Shoup’s model for *single-stage* games, which however cannot describe UC security.

A more fundamental problem, specific to the UC-AGM [ABK<sup>+</sup>21], is that a composition theorem holds only if the algebraic adversary “does not mix” protocols, i.e., if it explains group elements of some sub-protocol using only group elements from the same sub-protocol as a base. This means that we can safely compose protocols only if they operate on independent groups.

*Our models.* Informally, a group action  $\mathcal{G} = (G, X, \star)$  consists of a group  $G$  and a set  $X$ , along with an *action*  $\star : G \times X \rightarrow X$ , such that for any  $g, g' \in G$  and  $x \in X$ , it holds that  $(gg') \star x = g \star (g' \star x)$ . This definition allows us to capture the exponentiation-only property of Diffie-Hellman, but without imposing any structure on the set  $X$  so that we can instantiate it with Shor-resistant constructions. Similarly to the group setting, the most studied hardness assumption for group actions is the “discrete logarithm”, or *vectorization*, problem: it is easy to compute  $y = g \star x$  given  $g$  and  $x$ , while it is assumed to be hard computing  $g$  from only  $x, y$ . Group actions have long been considered for cryptography [BY91, GS10, Cou06], and the isogeny-based key exchange CSIDH [CLM<sup>+</sup>18] highlighted the potential of this framework, which now has different instantiations, like the lattice isomorphism problem [Dv22] and the general linear group action on tensors [JQSY19]; the problems of code equivalence and graph isomorphism can also be viewed as discrete logarithm problems with respect to an appropriate group action.

In a seminal paper for group action-based cryptography, Alamati et al. [ADMP20] introduced the notions of *effective-GA* (EGA) and *restricted-EGA* (REGA) to model cryptographic group actions and in particular CSIDH, where computing the group action efficiently is not possible for all group elements  $g \in G$ . We propose a new generalization of EGA and REGA to *hint-effective GA* (HEGA), which simply means that it is possible to attach an efficient way to evaluate the action to sampled group elements. We then introduce a type-safe model for group actions, which is a generalization of the generic group model for the group action setting. Informally, a circuit is said to be *type-safe* with respect to an action  $\mathcal{G}$  if it has two types of wires: *bit* wires and *element* wires. Bit wires can be combined together in any way with usual boolean gates, while element wires can only be used through special gates: an *action* gate, whose input are a bunch of bit wires

representing a group element  $g \in G$  and an element wire representing a set element  $x \in X$ , and it outputs an element wire representing  $g \star x$ ; and an *equality* gate, which outputs a bit wire from two element wires. Notice that the only way to get bits from elements is through equality gates, and this resolves the problems highlighted in [ZZK22, Zha22].

We can view our type-safe model as a “Maurer-style” generic model, while the Generic Group Action Model proposed in [DHK<sup>+</sup>23] is a “Shoup-style” model. We will need this formalization in order to avoid the problems in the AGM with the encoding of group elements. Notice however that like [DHK<sup>+</sup>23] we allow “generic” algorithms to have full access to the group structure, in contrast to the model proposed in [MZ22], where even the group operations are computed through an oracle. Using the type-safe model, and following [Zha22], we can define the Algebraic Action Model (AAM) as a “compiler” for certain games in the TS model. Loosely speaking, in our setting an adversary is said to be *algebraic* if, every time they output a set element  $y \in X$ , they must *explain* it as a group action  $y = g \star x$ , where  $x \in X$  was a set element already known; the adversary moreover has complete access to the group structure and representation. Then, given a type-safe protocol  $\pi$  and an algebraic adversary  $\mathcal{A}$ , we can “plug” it in:  $\mathcal{A}$  only has bit wires, but we can use the explanations it provides to connect its “element” outputs to the element wires of the protocol  $\pi$  as shown in Figure 1.

*UC emulation in the AAM.* We follow [ABK<sup>+</sup>21] in defining a restricted UC framework where we impose algebraic behaviour; as in [Zha22], we limit ourselves to type-safe protocols in order to properly define the algebraic framework, and in particular what it means to “output a set element”. Indeed, given a type-safe protocol  $\pi$  and a pair  $(\mathcal{A}, \mathcal{E})$  of algebraic adversary and environment, we can connect them to the protocol  $\pi$  as in the usual UC framework using the AA compiler described above. Informally, the UC-AAM has the same definitions of the standard UC framework, but we are restricting the adversary-environment pair to be algebraic, i.e. for every  $y \in X$  that they output towards the protocol, they must explain it with  $g \in G, x \in X$  such that  $y = g \star x$ .

We have the usual notions of a protocol  $\pi$  UC-AAM emulating a protocol  $\phi$  (denoted by  $\pi \sim \phi$ ) if an algebraic environment cannot distinguish the two executions, and we can prove transitivity of emulation. We can also define execution in an  $\mathcal{F}$ -hybrid model, where  $\mathcal{F}$  is any type-safe functionality, and prove that if  $\pi$  UC-AAM emulates  $\mathcal{F}$ , then also  $\rho^\pi \sim \rho^\mathcal{F}$ . However, like in the UC-AGM, the composition theorem needs extra care: since we have that  $\pi \sim \mathcal{F}$ , it means that the adversary uses set elements from protocols  $\pi$  as the “base” for outputting algebraic explanations; when proving that  $\rho^\pi \sim \rho^\mathcal{F}$  we can only get emulation with respect to algebraic adversaries that use set elements from  $\pi$  instead of the whole  $\rho^\pi$ . This means that even if we prove that  $\rho^\mathcal{F} \sim \mathcal{F}'$ , we cannot immediately obtain that  $\rho^\pi \sim \mathcal{F}'$ , since for the second part of the emulation we are also using set elements from  $\rho$  as the base for algebraic explanations. In particular, in order to get to a full composition theorem  $\rho^\pi \sim \mathcal{F}'$  we need to restrict to “non-mixing” adversaries, i.e. they explain set elements they output to  $\rho$  with elements previously seen in  $\rho$ , and set elements going into  $\pi$  with set elements coming from  $\pi$ . As noted in [ABK<sup>+</sup>21], this should not necessarily be seen as a limitation of UC-AGM (and UC-AAM), but rather as a limitation of proofs in idealized models. More discussion on this point can be found on their paper.

Ways to overcome this issue, and consider also “mixing adversaries”, could be either proving multiple protocol executions simultaneously or proving security in extended settings like GUC (UC with global setup) [CDPW07].

*The UC-EI model.* We finally introduce the UC-EI (Explicit Isogeny) model, which is simply the specialization of the UC-AAM to the group action of CSIDH [CLM<sup>+</sup>18]. We recall that CSIDH uses the action of the class group  $Cl(\mathcal{O})$  on the set of supersingular elliptic curves having  $\mathcal{O}$  as endomorphism ring defined by  $\mathfrak{a} \star E = E/E[\mathfrak{a}]$ , which is a commutative and regular action.



Since it is possible to compute the *twist*  $E^t$  of an elliptic curve, the models should consider this operation as well; the authors of [DHK<sup>+</sup>23] model twists externally in their CEGAT and AGAM<sup>T</sup>, while we don't need any new definition. In particular, since the twisting operation satisfies the relation  $(\mathfrak{a} \star E)^t = \mathfrak{a}^{-1} \star E^t$ , we notice that this translates into an action ( $\mathcal{G}_{\text{tw}}$  in Proposition 3) of the group  $CU(\mathcal{O}) \rtimes \mathbb{Z}/2\mathbb{Z}$ ; this results in a non-commutative and non-regular group action, but neither of the properties are of fundamental importance.

Moreover, the EI model doesn't seem to actually restrict real adversaries, given our current knowledge. Indeed, the only efficient way we know to generate a supersingular curve is via an isogeny walk or with twisting: a well known open problem in isogeny-based cryptography is in fact how to sample random supersingular elliptic curves without taking a random isogeny walk from an already known curve. The sampling problem has recently gained some attention [MMP22, BBD<sup>+</sup>22], but so far all the attempts to solve it have failed.

Assuming that the problem is hard implies that the CSIDH action group  $\mathcal{G}_{\text{tw}}$  is actually an unsampleable HEGA, and therefore restricting algorithms to be algebraic does not restrict the model. Notice that this is different to what happens in AGM, where algebraic algorithms have to provide a representation of any group element they output, but it is not true that the only way for them to output a new group element is to derive it using group multiplication from known group elements.

*The 2-message isogeny-based OT.* Once we have established our UC-EI model, proving full malicious security of the two-message OT protocol given in [LGD21] is a relatively easy task. Indeed, the algebraic behaviour of the adversary allows us to immediately extract the choice bit of a corrupted receiver, thus we can conclude the simulation. The algebraic explanations give also an easy reduction to the group action discrete logarithm problem simply by computing some equations, exactly as it usually happens in the AGM.

In this way, we obtain an efficient two-message protocol in the ROM with a trusted setup curve (TSC) based on CSIDH. In addition, we show how to eliminate the TSC requirement, but at the cost of adding an extra message of communication, so the resulting scheme is not optimal. This latter protocol needs one message less than the maliciously secure protocol described in [LGD21] with less computation and without TSC.

### 1.3 Other Related Work

Another important line of work aims to construct two-message oblivious transfer with a slightly weaker form of security, namely statistically sender-private OT (SSP OT) [NP01, AIR01]. In this setting, different schemes based on different quantum secure assumptions are known, like [BD18, DGI<sup>+</sup>19, MS20, ADMP20].

The recent work [BMM<sup>+</sup>23] introduces some new OT protocols based on isogenies; in particular, the authors show how to build UC-secure round-optimal protocols based on the computational CSIDH assumption, both in the plain model and in the setup model (i.e. respectively 4-message and 2-message). This is an important theoretical result, since round-optimal protocols were known only from the decisional CSIDH problem [ADMP20]. However, both constructions require a number of isogeny computations linear in the security parameter, which is highly inefficient.

More details about their constructions and efficiency of theirs and other isogeny-based OT protocols can be found in Section 5.3.

## 2 Preliminaries

For a set  $S$ , we denote by  $a \leftarrow S$  the process of drawing  $a$  from  $S$  with a uniform distribution on  $S$ . If  $\mathcal{D}$  is a probability distribution, we denote by  $a \leftarrow \mathcal{D}$  the process of

drawing  $a$  with the given probability distribution. For a probabilistic algorithm  $\mathcal{A}$ , we denote by  $a \leftarrow_s \mathcal{A}$  the process of assigning  $a$  the output of algorithm  $\mathcal{A}$ , with the underlying probability distribution being determined by the random coins of  $\mathcal{A}$ . We write  $\approx$  (resp.  $\approx_s$ ) to denote computational (resp. statistical) indistinguishability between probabilistic distributions.

In this work we assume familiarity with the UC framework, of which we provide an informal overview in Appendix A.3; in Appendix A.4 we describe our functionalities. In Appendix B, we also give a brief description of UC-AGM [ABK<sup>+</sup>21].

## 2.1 Cryptographic Group Actions

Group actions have recently been getting a lot of attention for their use in cryptography, starting from the ‘‘Hard Homogeneous Spaces’’ by Couveignes [Cou06], which is a precursor of CSIDH. Here we will follow the formalization by Alapati et al. [ADMP20].

**Definition 1** (Group action). A group  $G$  is said to *act* on a set  $X$  if there is a map  $\star : G \times X \rightarrow X$  that satisfies the following two properties:

IDENTITY. If  $e$  is the identity of  $G$ , then,  $\forall x \in X$ , we have  $e \star x = x$

COMPATIBILITY. For any  $g, h \in G$  and any  $x \in X$ , then  $(gh) \star x = g \star (h \star x)$ .

A group action as described in the previous definition is usually denoted by  $\mathcal{G} = (G, X, \star)$ . The standard notion of cryptographic group actions is given by *effective* group action (EGA). Roughly, an EGA  $(G, X, \star)$  is such that all the well-defined group operations and group action operations are efficiently computable, sampling random group elements is efficient and set elements are uniquely represented. It is also possible to endow group actions with different hardness assumptions like one-way EGA (ow-EGA), weak-unpredictable EGA (wUP-EGA), weak pseudorandom EGA (wPR-EGA) [ADMP20].

EGA is usually too powerful to capture isogeny-based assumptions, therefore, to model isogeny-based protocols, [ADMP20] provides the definition of *restricted effective group action* (REGA), where it is possible to only evaluate the action of a generating set of small cardinality.

## 2.2 CSIDH

In [CLM<sup>+</sup>18], the authors propose an efficient post-quantum abelian group action, as an adaptation of the Couveignes-Rostovtsev-Stolbunov scheme, from which they derive a key-exchange primitive, called CSIDH.

They consider a supersingular elliptic curve  $E$  over  $\mathbb{F}_p$ , so that its  $\mathbb{F}_p$ -rational endomorphism ring  $\text{End}_p(E)$  is an order  $\mathcal{O}$  in a quadratic imaginary field. If  $\mathfrak{a}$  is a non-zero ideal in  $\mathcal{O}$ , then it defines a kernel subgroup  $E[\mathfrak{a}]$ . We can then consider the quotient isogeny  $\psi : E \rightarrow E' = E/E[\mathfrak{a}]$  with kernel  $E[\mathfrak{a}]$ . This isogeny, as well as its codomain, is well-defined up to isomorphism. If  $\mathfrak{a} = (\alpha)$  is a principal ideal, then  $\psi \cong \alpha$  and  $E/E[\mathfrak{a}] \cong E$ . Denoting by  $\mathcal{E} = \text{Ell}_p(\mathcal{O})$  the set of curves having  $\mathcal{O}$  as their  $\mathbb{F}_p$ -endomorphism ring, we have a free and transitive action of the class group  $Cl(\mathcal{O}) \times \mathcal{E} \rightarrow \mathcal{E}$  given by  $\mathfrak{a} \star E := E/E[\mathfrak{a}]$ .

The main idea of CSIDH is to pick a prime of the form  $p = 4\ell_1 \dots \ell_n - 1$ , with  $\ell_i$  small odd primes. In addition, they fix  $E_0 : y^2 = x^3 + x$ , which is supersingular when  $p \equiv 3 \pmod{4}$ . This curve has  $\text{End}_p(E_0) \cong \mathbb{Z}[\sqrt{-p}]$ .

Since the characteristic polynomial of the Frobenius map is  $\pi^2 + p = 0$ , when reduced modulo  $\ell_i$  it becomes  $\pi^2 - 1 \equiv 0 \pmod{\ell_i}$ , given that  $p \equiv -1 \pmod{\ell_i}$ . In particular, this means that  $\ell$  splits as the product of  $\mathfrak{l}_i = (\ell_i, \pi - 1)$  and  $\bar{\mathfrak{l}}_i = (\ell_i, \pi + 1)$  inside  $\mathcal{O}$  (primes that are split are called *Elkies* primes).

The very peculiar choice of the prime  $p$  implies that the evaluation of the action  $\mathfrak{l}_i \star E$  is very easy: the kernel of the corresponding isogeny  $E[\mathfrak{l}_i]$  is the intersection of  $\ker(\ell_i)$  and

$\ker(\pi - 1)$ , which is the  $\mathbb{F}_p$ -rational  $\ell_i$ -torsion subgroup. By computing it and using Vélú's formulas, we can compute the isogeny  $\varphi_{\ell_i}$ .

For computing  $\iota_i^{-1} \star E$  we can either compute the  $\mathbb{F}_{p^2}$ -rational  $\ell_i$ -torsion or we can use the fact that  $(\mathfrak{a} \star E)^t \cong \mathfrak{a}^{-1} \star E^t$ , where  $E^t$  is the quadratic twist of  $E$ . For more details on evaluating the class group action, we refer to [CLM<sup>+</sup>18, section 8].

Another key aspect of CSIDH is that it uses curves in Montgomery form. Indeed, we have the following proposition.

**Proposition 1** ([CLM<sup>+</sup>18, Proposition 8]). *Let  $p \geq 5$  be a prime with  $p \equiv 3 \pmod{8}$ , and let  $E/\mathbb{F}_p$  be a supersingular elliptic curve. Then  $\text{End}_p(E) = \mathbb{Z}[\pi]$  if and only if there exists  $A \in \mathbb{F}_p$  such that  $E$  is  $\mathbb{F}_p$ -isomorphic to the curve  $E_A : y^2 = x^3 + Ax^2 + x$ . Moreover, such  $A$  is unique.*

This means that it is sufficient to use the  $A$  coefficient as the public key, instead of a  $j$ -invariant and then having to check that it has the correct endomorphism ring. The only check needed is that  $A \notin \{\pm 2\}$  (otherwise  $E_A$  is not even smooth), and that  $E_A$  is supersingular. Furthermore, it is very easy to see that in this setting the twist of  $E_A$  is simply  $E_A^t \cong E_{-A}$ .

**CSIDH assumptions.** We list below the main hardness assumptions we use in our protocols; notice that these are the direct translation of respectively the discrete logarithm and the computational Diffie-Hellman problems to the CSIDH setting.

**Problem 1** (Vectorization (Vec-CSIDH)). *Given curves  $(E, r \star E)$  with  $E \in \mathcal{E}, r \in \text{Cl}(\mathcal{O})$ , the problem asks to find said element  $r$ .*

Notice that since the action is regular, the generic vectorization problem is equivalent to the one with  $E = E_0$ .

**Problem 2** (Computational CSIDH (Comp-CSIDH)). *Given curves  $(E, r \star E, s \star E)$  in  $\mathcal{E}$  with  $r, s \in \text{Cl}(\mathcal{O})$ , the problem asks to find  $E' \in \mathcal{E}$  such that  $E' = rs \star E$ .*

Additional hard problems and some discussion can be found in Section A.2.

**Sampling ideals in the class group.** One of the issues with CSIDH is that we know how to *efficiently* evaluate only the action of the ideals  $\mathfrak{l}_i$  corresponding to the Elkies primes  $\ell_i$  of the factorization of  $p + 1$ . This is why CSIDH has been defined as a REGA [ADMP20], which actually means that it describes an action of the additive group  $\mathbb{Z}^n$  on the set  $\mathcal{E}$ . Indeed, we have a morphism  $\Pi : \mathbb{Z}^n \rightarrow \text{Cl}(\mathcal{O})$  given by  $\Pi(e_1, \dots, e_n) = \prod_{i=1}^n \mathfrak{l}_i^{e_i}$ , and we can define  $v \star E = \Pi(v) \star E$  for any integer vector  $v \in \mathbb{Z}^n$ .

Therefore, even if we could in theory pick a uniformly random element of the source group  $\text{Cl}(\mathcal{O})$  (which is what many protocols would require), we can only efficiently represent and operate with it using the integer vector representation.

We thus pose the following assumption in order to deal with this issue.

**Assumption 1.** *Let  $\mathcal{D}_m$  be the distribution on  $\text{Cl}(\mathcal{O})$  given by sampling  $(e_1, \dots, e_n)$  uniformly at random from  $[-m, m]^n$  and outputting  $\mathfrak{a} = \Pi(e_1, \dots, e_n) = \prod_{i=1}^n \mathfrak{l}_i^{e_i}$ . Then, if  $(2m + 1)^n \approx |\text{Cl}(\mathcal{O})| \approx \sqrt{p}$ , for any  $E \in \mathcal{E}$ , the distribution<sup>1</sup>  $\mathcal{D}_m \star E$  is computationally indistinguishable from the uniform distribution on  $\mathcal{E}$ .*

This assumption is strongly motivated by the following heuristic from [CLM<sup>+</sup>18, Section 7.1]: choosing  $m$  to be the smallest integer such that  $(2m + 1)^n \geq |\text{Cl}(\mathcal{O})|$ , for any  $\mathfrak{a} \in \text{Cl}(\mathcal{O})$ , the size of  $\Pi^{-1}(\mathfrak{a}) \cap [-m, m]^n$  is “small”; this means that the min-entropy of the uniform distribution on  $\text{Cl}(\mathcal{O})$  and the one of  $\mathcal{D}_m$  differ just by a few bits.

<sup>1</sup>The distribution  $\mathcal{D}_m \star E$  is defined by sampling  $\mathfrak{a}$  from  $\mathcal{D}_m$  and then outputting  $\mathfrak{a} \star E$ .



*Remark 1.* In the case of CSIDH-512, the structure of the class group has been fully computed [BKV19], which means that we know  $\Lambda = \ker \Pi$ , and thus can compute unique representatives. This implies that we can consider the CSIDH-512 action as a full blown EGA, and allow for uniform sampling without heuristic assumptions. Notice however that this is at the cost of solving a CVP problem in  $\Lambda$  each time that we want to evaluate the action, as highlighted by Panny [Pan23].

Additional definitions and preliminaries about isogenies and computational assumptions are given in Appendix A.

### 3 Type-Safe and Algebraic Models for Group Actions

In this section, we first propose a generalization of both EGAs and REGAs, called *hint-effective group action* (HEGA), and secondly introduce a variant of Zhandry’s type-safe (TS) model for the setting of group actions, which corresponds to the generic group model. We then describe the *algebraic action model* (AAM) as a compiler, similarly to the AGM but for group actions, where any adversary must explain any set element that it outputs with the group element that has generated it. Finally, we study this model in the UC framework.

A generic (and algebraic) group action model has been proposed by Duman et al. [DHK<sup>+</sup>23], where the authors prove generic hardness results and equivalence between different assumptions in a quantum setting; in particular, their algebraic group action model is almost identical to the one proposed by us, and they both differ from the one first proposed by [MZ22] in the sense that we both only encode set elements and otherwise give full access to the group to the adversary. However, we follow [Zha22] to first describe a type-safe generic model which formally delimits the games that can be described in our algebraic action model, while [DHK<sup>+</sup>23] poses the same informal constraint as in the AGM that “input that is not a set element  $x \in X$  does not depend on set elements”.

Another difference is that [DHK<sup>+</sup>23] only considers abelian group actions (in particular the concrete group  $\mathbb{Z}/n\mathbb{Z}$ ), and the authors have to model externally the twisting property of CSIDH, while we embed it in a bigger group action, which is however neither abelian nor regular.

#### 3.1 Hint-Effective Group Actions

**Definition 2** (HEGA). A group action  $(G, X, \star)$  is *hint-effective* if the following properties are satisfied:

1. The group  $G$  is finite, and there is an efficiently sampleable distribution  $\mathcal{D}_G$  on  $G$ . Moreover, sampling from this distribution produces also a *hint*  $e$ , and we write that as  $(g, e) \leftarrow \mathcal{D}_G$ .
2. The set  $X$  is finite, and there are efficient algorithms for membership testing and computing a unique representation.
3. There is a distinguished and known element  $x_0 \in X$ , called the *origin*.
4. There exists an efficient algorithm such that for any  $(g, e) \leftarrow \mathcal{D}_G$  and any  $x \in X$  computes  $g \star x$ , eventually using the hint  $e$ .

We collect these parameters in  $\mathcal{G} = (G, X, \star, \mathcal{D}_G, x_0)$ .

Notice that an EGA is an HEGA where the hints are empty and group operations on  $G$  are also efficient, while a REGA is an HEGA where the hint is the exponent vector with respect to the chosen generating set. In addition, all protocols of [ADMP20] built from

EGAs can also be instantiated from HEGAs, as soon as  $\mathcal{D}_G$  is statistically close to the uniform distribution on  $G$ .

**Computational assumptions on HEGAs.** We can define the one-way, weak unpredictable and weak pseudorandom hardness assumptions on HEGAs exactly as for EGAs in [ADMP20]. More explicitly, we can define the “discrete logarithm” analogue for group actions as follows. Note that we do not restrict ourselves to regular actions, so there might be more than one possible answer.

**Problem 3** (DLog-HEGA). *Given  $x = g \star x_0$  for  $(g, e) \leftarrow \mathcal{D}_G$ , compute any  $g' \in G$  such that  $x = g' \star x_0$ .*

We finally describe an important property of a group action, namely the inability of sample directly from  $X$  without using group elements.

**Definition 3** (Unsampleable HEGA). An HEGA  $\mathcal{G} = (G, X, \star, \mathcal{D}_G, x_0)$  is said to be *unsampleable* if for any PPT algorithm  $\mathcal{A}_G(x_1, \dots, x_n)$  that outputs a set element  $x \in X$ , there exists a PPT algorithm  $\mathcal{A}'_G$  that outputs  $(g, e)$  such that  $x = g \star x_i$  for some  $i \in \{0, 1, \dots, n\}$ .

### 3.2 The Type-Safe Model

Following Zhandry’s type-safe model [Zha22], we define a similar model in the context of group actions, and in particular HEGAs.

**Definition 4.** Let  $\mathcal{G} = (G, X, \star, \mathcal{D}_G, x_0)$  be an HEGA. An algorithm  $\mathcal{A}$ , given as a circuit, is said to be *type-safe w.r.t.  $\mathcal{G}$*  (written as  $TS_{\mathcal{G}}$ ) if

- It has two types of wires, *bit* wires and *element* wires. Element wires should be thought as containing/hiding values  $x \in X$ .
- There always exists a given element wire containing the origin  $x_0$  of the action.
- The only gates allowed are the following:

BOOLEAN GATE: It only has bit wires for input and output, and can perform any classical boolean function.

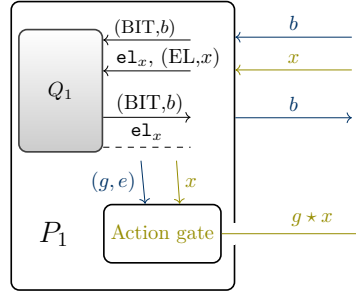
ACTION GATE: It has as inputs some bit wires that encode the group element  $g$  with an hint  $e$ , and an element wire containing  $x$ . Its output is an element wire which contains  $g \star x$ .

EQUALITY GATE: It has two element wires  $x, y$  as inputs, and outputs a bit wire that is 1 if  $x = y$ , and 0 otherwise.

This “type-safe” model can be seen as a possible definition of a *generic group action*; in [MZ22] the authors briefly propose a generic group action framework based on Shoup’s model, stating that it can also model quantum adversaries. Unfortunately, since AGM is incompatible with Shoup’s GGM, for our purposes we decided to follow the type-safe approach. Moreover, we allow the adversary to access the group structure in any case: the hardness of a group action should derive from hiding the group inside the set, and not from the group having a “generic” structure.

### 3.3 The Algebraic Action Model

We now introduce the *algebraic action model* (AAM) as a relaxation of the TS model, where adversaries can behave arbitrarily, but must explain the set elements they produce. First we define what it means for an adversary to behave algebraically.



**Figure 1:** Compiled protocol  $AA(\pi, \{Q_1\})$

**Definition 5.** Let  $\mathcal{G}$  be an HEGA, and  $\mathcal{A}$  a PPT algorithm<sup>2</sup>. We say that  $\mathcal{A}$  uses algebraic actions w.r.t.  $\mathcal{G}$  (denoted by  $AA_{\mathcal{G}}$ ) if

- It receives two types of input messages:
  - (BIT,  $b$ ), where  $b$  is a bit-string
  - (EL,  $x$ ), where  $x \in X$  is the representation of some element
- It sends two types of output messages:
  - (BIT,  $b$ ), where  $b$  is a bit-string
  - (EL,  $(g, e, x)$ ), where  $g \in G$ ,  $e$  is some hint and  $x$  is one of the previously received EL messages.

The meaning of this definition is that if an algebraic adversary wants to output a set element  $y \in X$ , it must explain it as some  $y = g * x$  for some  $x \in X$  that it has already seen. Notice that any given  $TS_{\mathcal{G}}$  algorithm can be translated into an  $AA_{\mathcal{G}}$  algorithm, and thus we can use the AA model as a compiler in the following way.

**Definition 6.** Let  $\pi$  be a  $TS_{\mathcal{G}}$  protocol between parties  $P_1, \dots, P_n$ , and let  $Q = \{Q_{i_1}, \dots, Q_{i_k}\}$  a set of algebraic algorithms, with  $S = \{i_1, \dots, i_k\} \subset \{1, \dots, n\}$ .

Then  $AA(\pi, Q)$  is a protocol where each party  $P_s$  for  $s \in S$  is replaced by  $Q_s$  and its communication wires have been transformed as follows:

- Any incoming *bit* wire to  $P_s$  gets translated to a BIT message for  $Q_s$ .
- Any incoming *element* wire to  $P_s$  gets translated into a (EL,  $x$ ) message for  $Q_s$ , and the element wire gets labelled as  $e\mathbf{l}_x$ .
- Any outgoing BIT message from  $Q_s$  gets translated into a bit wire from  $P_s$ .
- Any outgoing (EL,  $(g, e, x)$ ) message from  $Q_s$  gets translated back into an element wire from  $P_s$ , by applying an action gate on  $e\mathbf{l}_x$  with group element  $(g, e)$ .

The AA compilation on a party  $P_1$  being substituted by an algebraic  $Q_1$  can be seen in Figure 1. What this compiler is doing is transforming a fixed subset of parties in a TS protocol into algebraic machines, forcing them to output group elements with which they have generated the element  $y$  they are sending. The goal of this transformation is being able to give a definition of what it means to be “secure in the algebraic action model”. The intuition here is that a protocol  $\pi$  is secure in the AAM model if its compiled version  $AA(\pi)$  is secure in the standard model. This type of approach is exactly the one used by [Zha22] for formally (re)defining security in the AGM.

We conclude the discussion of the AAM with the following informal statement about unsampleable HEGA.

<sup>2</sup>Remark that we don’t impose the use of *element* wires to  $\mathcal{A}$ .

*Remark 2.* Let  $\mathcal{G}$  be an unsampleable HEGA, and let  $\mathcal{A}$  be any PPT algorithm that receives inputs and sends outputs of the form  $(\text{BIT}, b)$  and  $(\text{EL}, x)$ . Then, if the BIT inputs give no information about the group action  $\mathcal{G}$ , there exists an *algebraic* algorithm  $\mathcal{A}'$  that upon the same inputs of  $\mathcal{A}$  gives the same outputs, giving also an explanation to the EL messages that it outputs.

This gives the intuition that for an unsampleable HEGA, all possible adversaries are actually algebraic; however, this is the closest we can get to a formal result, which cannot exist since it's not clear what it means that BIT messages don't reveal anything about  $\mathcal{G}$ . This is the fundamental issue with the AGM, as highlighted in [ZZK22]; the best resolution of this issue is some kind of type-safe model, but there will always be a gap between the TS model and the standard model.

### 3.4 UC Emulation in the Algebraic Action Model

We now formalize the definition of security in the AA model within the UC framework. First, we use the previous compiler to define the execution of a type-safe protocol against algebraic adversaries.

**Definition 7.** Let  $\pi$  be a  $TS_{\mathcal{G}}$  protocol, and  $\mathcal{A}, \mathcal{Z}$  be  $AA_{\mathcal{G}}$  algorithms.

Denote by  $UC(\pi)$  the  $TS_{\mathcal{G}}$  protocol defined by adding a “dummy” type-safe adversary and environment  $\mathcal{A}', \mathcal{Z}'$ . In particular, we have that:

- The environment  $\mathcal{Z}'$  can send input messages to the main parties of  $\pi$ , and read their output messages, all of which consist of both a bit wire and an element wire.
- The environment  $\mathcal{Z}'$  and the adversary  $\mathcal{A}'$  can communicate freely.
- The adversary  $\mathcal{A}'$  has a *backdoor* channel towards the parties, which also has both bit wires and element wires.

We define  $\text{EXEC}_{\pi, \mathcal{A}, \mathcal{Z}}(z)$  as the output bit of the environment  $\mathcal{Z}$  in an execution of the protocol  $AA(UC(\pi), \{\mathcal{A}, \mathcal{Z}\})$  with input  $z$ , i.e. we are compiling the “dummy” adversary and environment into meaningful algebraic ones.

With this definition we are modelling a similar setting of UC-AGM, while using the type-safe model to give a precise definition on the condition “when the adversary outputs a group element” given in UC-AGM [ABK<sup>+</sup>21].

Notice that with our definition we are allowing  $\pi$  to have hybrid type-safe functionalities, and also to have element wires as input/output. For example,  $\mathcal{F}_{\text{RO}}$  is a  $TS_{\mathcal{G}}$  algorithm that keeps a list of pairs (element wire, bit-string). When activated with an element wire input, it checks with equality gates if that input is present; if yes, it answers with the bit-string, otherwise it samples a bit-string, answers with it and stores it in the list with the input element wire.

We can finally define what an algebraic emulation of a protocol is, and consequently define an algebraic realization of a type-safe functionality.

**Definition 8.** Let  $\pi, \phi$  be  $TS_{\mathcal{G}}$  protocols. We say that  $\pi$  *UC-AA emulates*  $\phi$  if for any  $AA_{\mathcal{G}}$  adversary  $\mathcal{A}$  there is an  $AA_{\mathcal{G}}$  simulator  $\mathcal{S}$  such that for all  $AA_{\mathcal{G}}$  environments  $\mathcal{Z}$  it holds that

$$\text{EXEC}_{\pi, \mathcal{A}, \mathcal{Z}} \approx \text{EXEC}_{\phi, \mathcal{S}, \mathcal{Z}}.$$

If  $\mathcal{F}$  is a  $TS_{\mathcal{G}}$  functionality, we say that a  $TS_{\mathcal{G}}$  protocol  $\pi$  *UC-AA realizes*  $\mathcal{F}$  if  $\pi$  UC-AA emulates  $\text{IDEAL}_{\mathcal{F}}$ , where  $\text{IDEAL}_{\mathcal{F}}$  is the protocol in which all the parties forward their input to the ideal functionality  $\mathcal{F}$ , and output the returned value from  $\mathcal{F}$ .

We can also prove a composition theorem, but, as in UC-AGM, there is an important limitation. In order to obtain emulation, the adversary must give its explanations relative to elements of the same sub-protocol where it's sending the output. Concretely, we say that an  $AA_{\mathcal{G}}$  adversary against a  $TS_{\mathcal{G}}$  composed protocol  $\rho^\pi$  is *non-mixing* if the explanations of messages to be sent to  $\rho$  only use element wires from  $\rho$ , and analogously with  $\pi$ . Therefore, we can state the composition theorem respect to non-mixing adversaries as follow. Note the proof of this theorem is exactly the same as the one given in [ABK<sup>+</sup>21].

**Theorem 1** (Informal). *Let  $\mathcal{F}_1, \mathcal{F}_2$  be TS functionalities. Let  $\pi$  be a TS protocol that UC-AA realizes  $\mathcal{F}_2$ , and  $\rho$  a protocol that UC-AA realizes  $\mathcal{F}_1$  in the  $\mathcal{F}_2$ -hybrid model. Then protocol  $\rho^\pi$  UC-AA realizes  $\mathcal{F}_1$  against non-mixing algebraic adversaries.*

## 4 The Explicit Isogeny Model

In this section we will introduce the Explicit Isogeny model of computation, which we will then use to prove the security against malicious adversaries of the two-message OT protocol proposed in [LGD21]. Concretely, the UC-EI model is nothing else than UC-AA instantiated with the action given by CSIDH, where we also incorporate twists for a more accurate model. This means that for any curve  $E$  the adversary must output the secret isogeny path that it used to generate  $E$ .

We will then argue that, given the difficulty of the “hash-to-curve” problem, the CSIDH action is an example of an *unsampleable HEGA*, so that this new model of computation is really close to the standard model.

### 4.1 The CSIDH Action with Twists

Recall that the CSIDH action is  $\star : Cl(\mathcal{O}) \times \mathcal{E} \rightarrow \mathcal{E}$ , where  $\mathcal{O}$  is the order  $\mathbb{Z}[\sqrt{-p}]$  and  $\mathcal{E} = Ell_p(\mathcal{O})$ . However, to fully capture the structure of the  $\mathbb{F}_p$ -rational supersingular isogeny graph it is necessary to also consider twists. This has also been proposed and used by [AEK<sup>+</sup>22] to construct *password authentication key-exchange* (PAKE) protocols and to prove their (in)security. Moreover, also Duman et al. [DHK<sup>+</sup>23] consider twists in their AGAM<sup>T</sup>, by modelling them as an external construction. However, we notice that twists can still be described internally to the group action framework, by introducing the appropriate semidirect product.

We recall that a semidirect product of two groups is given by the following definition.

**Proposition 2** (Semidirect product). *Let  $H, N$  be groups, and  $\phi : H \rightarrow \text{Aut}(N)$  (where  $\text{Aut}(N)$  denotes the group of all automorphisms of  $N$ ). Then the cartesian product  $N \times H$  equipped by the operation*

$$(n_1, h_1)(n_2, h_2) = (n_1\phi(h_1)(n_2), h_1h_2)$$

*is a group, denoted by  $N \rtimes_{\phi} H$ .*

This construction is useful because of the following classical result in the theory of group actions.

**Theorem 2.** *Let  $H$  be a group and  $N$  a  $H$ -group, meaning that there is a map  $\phi : H \rightarrow \text{Aut}(N)$ . Suppose that both  $H$  and  $N$  act on the same set  $X$  (with actions denoted by  $\star_H, \star_N$  respectively) in a compatible way, i.e.  $h \star_H (n \star_N x) = \phi_h(n) \star_N (h \star_H x)$ . Then there is a well-defined action of  $N \rtimes_{\phi} H$  on the set  $X$  given by  $(n, h) \star x = n \star_N (h \star_H x)$ .*

We will apply the theorem in our setting, where  $H = \mathbb{Z}/2\mathbb{Z}, N = Cl(\mathcal{O}), X = \mathcal{E}$ . The action of  $\mathbb{Z}/2\mathbb{Z}$  on  $\mathcal{E}$  is exactly twisting, in particular  $0 \cdot E = E$  and  $1 \cdot E = E^t$ . Moreover,  $\mathbb{Z}/2\mathbb{Z}$  has also a natural action on  $Cl(\mathcal{O})$  given by  $1 \cdot \mathfrak{a} = \mathfrak{a}^{-1}$ , which corresponds to the map



$\iota : \mathbb{Z}/2\mathbb{Z} \rightarrow \text{Aut}(Cl(\mathcal{O}))$  where  $\iota(1) : \mathfrak{a} \mapsto \mathfrak{a}^{-1}$ . It is now easy to see that the compatibility requirement is exactly the fact that  $(\mathfrak{a} \star E)^t = \mathfrak{a}^{-1} \star E^t$ . We can then construct a new action on  $\mathcal{E}$ , which automatically includes twists in its description.

**Corollary 1.** *There is an action of  $G_{\text{tw}} := Cl(\mathcal{O}) \rtimes_{\iota} \mathbb{Z}/2\mathbb{Z}$  on  $\mathcal{E}$  given by  $(\mathfrak{a}, 0) \star_{\text{tw}} E = \mathfrak{a} \star E$  and  $(\mathfrak{a}, 1) \star_{\text{tw}} E = \mathfrak{a} \star E^t$ .*

Since  $|G_{\text{tw}}| = 2|Cl(\mathcal{O})|$ , the action will not be regular anymore; indeed, for any  $E = \mathfrak{a} \star E_0$  it's easy to see that its stabilizer is  $\text{Stab}(E) = \{(1, 0), (\mathfrak{a}^2, 1)\}$ .

We conclude by showing that what we have constructed is still an HEGA.

**Proposition 3.** *Let  $\star_{\text{tw}} : G_{\text{tw}} \times \mathcal{E} \rightarrow \mathcal{E}$  the action defined above. Then the group action  $\mathcal{G}_{\text{tw}} = (G_{\text{tw}}, \mathcal{E}, \star_{\text{tw}}, \mathcal{D}_{G_{\text{tw}}}, E_0)$  such that:*

- $E_0$  is the same as in the CSIDH action, namely the curve  $y^2 = x^3 + x$
- $\mathcal{D}_{G_{\text{tw}}}$  is defined by independently sampling  $(\mathfrak{a}, e)$  from  $Cl(\mathcal{O})$  as in CSIDH, and choosing a random bit  $b \in \mathbb{Z}/2\mathbb{Z}$ ; the group element is  $(\mathfrak{a}, b)$  and the hint is  $e' = (e, b)$

is an HEGA. Moreover, if Assumption 1 holds,  $\mathcal{D}_{G_{\text{tw}}}$  is close to the uniform distribution.

*Proof.* The last claim follows directly from the assumption that the CSIDH sampling is close to the uniform distribution on  $Cl(\mathcal{O})$ . The fact that  $\mathcal{G}_{\text{tw}}$  is a HEGA directly follows from the fact that CSIDH is a HEGA (indeed, the hint allows us to evaluate the action of sampled group elements).  $\square$

## 4.2 The UC-EI Model

We are now ready to define the Explicit Isogeny model, which is an instantiation of UC-AA with  $\mathcal{G}_{\text{tw}}$ . More concretely, we see that our compiler turns the adversaries into EI-adversaries.

**Definition 9.** Let  $\mathcal{G}_{\text{tw}}$  be the HEGA defined before. We say that an algorithm  $\mathcal{A}$  uses *Explicit Isogenies* (EI) if its communication tapes have messages of the type  $(\text{bit}, m)$  and  $(\text{curve}, E)$ , where  $m$  is a bit-string and  $E$  is an element of  $\mathcal{E}$ , i.e., a supersingular curve over  $\mathbb{F}_p$ . Moreover, for any outgoing message  $(\text{curve}, E)$ ,  $\mathcal{A}$  must also send an explanation  $(\mathfrak{a}, e, E')$  such that  $E = \mathfrak{a} \star E'$ , where  $E'$  is one of the previous incoming curves or its twist.

Notice that for our UC-AA definition to make sense, it's necessary that the original protocol is type-safe w.r.t.  $\mathcal{G}_{\text{tw}}$ , so incorporating twists into the action is a fundamental step of the construction, otherwise we couldn't describe protocols that use twists, such as the one proposed by Lai et al. [LGD21].

Our EI model can thus be seen as the formalized type-safe variant of the AGAM<sup>T</sup> model by [DHK<sup>+</sup>23]. We also restate our definition of UC emulation, which is exactly that given for UC-AA for the specific case of  $\mathcal{G}_{\text{tw}}$  protocols against EI adversaries.

**Definition 10.** Given two  $TS_{\mathcal{G}_{\text{tw}}}$  protocols  $\pi$  and  $\phi$ , we say that  $\pi$  *UC-EI emulates*  $\phi$  if for any efficient EI-adversary  $\mathcal{A}$  there is an efficient EI-simulator  $\mathcal{S}$  such that for any efficient EI-environment  $\mathcal{Z}$  we have that

$$\text{EXEC}_{\phi, \mathcal{S}, \mathcal{Z}} \approx \text{EXEC}_{\pi, \mathcal{A}, \mathcal{Z}}.$$

If  $\mathcal{F}$  is an ideal functionality and  $\phi = \text{IDEAL}_{\mathcal{F}}$ , we say that  $\pi$  *UC-EI realizes*  $\mathcal{F}$ .

Like UC-AGM, also UC-EI limits the capabilities of the adversary, so it may seem that it's less expressive than the plain model of UC security. However, as we will discuss in the next sections, there is strong evidence suggesting that any PPT adversary  $\mathcal{A}$  behaves like in the explicit isogeny model, in particular given the fact that “hashing” to a supersingular curve seems to be very hard.

### 4.3 The EI Model and the Sampling Problem

One of the main open problems of isogeny-based cryptography is how to sample a supersingular curve without taking a random isogeny walk from another known curve; this is also called the “hashing-to-curve” problem, which can be roughly stated as follows.

**Problem 4 (Informal).** *Given a prime  $p$ , compute a supersingular curve  $E/\mathbb{F}_{p^2}$ , without revealing anything about  $\text{End}(E)$ , or any information that helps solving the isogeny path problem from  $E$ .*

More concretely, the sampling problem is defined and studied in [MMP22], where it is called the cSRS (cryptographic Supersingular Random Sampling) problem. In this work and another comprehensive study of the problem [BBD<sup>+</sup>22], the authors review some known methods to generate supersingular elliptic curves and explore possible ways to hash into the isogeny graph, but the conclusion of both these works is that the best algorithms we know for this task still have exponential complexity.

We can then introduce a new knowledge assumption based on this problem.

**Assumption 2.** *For any PPT algorithm  $\mathcal{A}$  that receives as input a prime number  $p$  and outputs a supersingular  $j$ -invariant  $j \in \mathbb{F}_{p^2}$ , there exists a PPT algorithm  $\mathcal{A}'$  that outputs the pair  $(j, R)$ , where  $R = \text{End}(E)$  is the endomorphism ring of a curve  $E$  with  $j(E) = j$ .*

We can also use a slightly different variant of the problem, that only deals with isogeny paths and not endomorphism rings.

**Assumption 3.** *For any PPT algorithm  $\mathcal{A}(p, j_0, j_1, \dots, j_n)$  that outputs a supersingular  $j$ -invariant  $j \in \mathbb{F}_{p^2}$  knowing a list of some  $j$ -invariants, there exists a PPT algorithm  $\mathcal{A}'$  that outputs a computable isogeny  $\phi : E_i \rightarrow E$ , where  $j(E) = j$  and  $j(E_i) = j_i$ .*

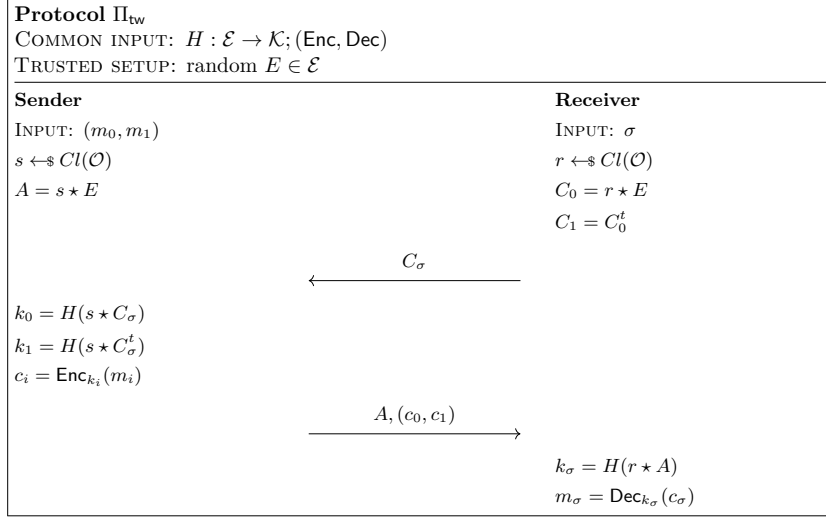
It is important to notice that the two different Assumptions 2 and 3 are not equivalent: knowing an isogeny  $\phi : E_0 \rightarrow E$  is equivalent to knowing  $\text{End}(E)$  only if we can compute  $\text{End}(E_0)$ , but in an interactive protocol a party usually receives a supersingular curve from other parties, and cannot know its endomorphism ring. The second assumption is thus trying to model exactly the multi-party computation setting, by forcing any party to generate new curves only by walking in the isogeny graph, starting from curves that have been sent to it. Moreover, in [BBD<sup>+</sup>22] the authors highlight some variants for the hashing problem, in particular the problem of sampling  $\mathbb{F}_p$ -rational supersingular curves. It is related to the general problem, and it likely seems as difficult, but it is very hard to prove an equivalence between them. However, we currently don’t know better algorithms to hash into the  $\mathbb{F}_p$ -graph than those that hash into the  $\mathbb{F}_{p^2}$ -graph.

We will then pose another assumption, specific to the CSIDH setting.

**Assumption 4.** *For any PPT algorithm  $\mathcal{A}(p, E_0, E_1, \dots, E_n)$  that outputs a supersingular curve  $E/\mathbb{F}_p$  knowing a list of some supersingular curves  $E_i/\mathbb{F}_p$ , there exists a PPT algorithm  $\mathcal{A}'$  that outputs a computable  $\mathbb{F}_p$ -rational isogeny  $\phi : E'_i \rightarrow E$ , where  $E'_i$  is  $E_i$  or one of its twists.*

Notice that this is a quite literal translation of Assumption 3 into the  $\mathbb{F}_p$ -isogeny graph, where we have to use curves instead of  $j$ -invariants: any supersingular invariant  $j \in \mathbb{F}_p$  will correspond to multiple curves which are  $\overline{\mathbb{F}}_p$ -isomorphic, but not  $\mathbb{F}_p$ -isomorphic, i.e. all the twists. Moreover, this assumption also captures the fact that twists are easy to compute, so an algorithm can easily generate  $E^t$  from  $E$ , even without knowing an isogeny from  $E$  to  $E^t$ .

This new assumption is actually needed, and it doesn’t seem to follow directly from the other two. Indeed, Assumption 2 and Assumption 4 are not related for the exact same reason for which Assumption 2 and Assumption 3 are not equivalent. Trying to directly



**Figure 2:** The twist OT protocol by Lai et al.

relate Assumptions 3 and 4 means instead translating  $\mathbb{F}_{p^2}$ -isogenies into  $\mathbb{F}_p$ -isogenies, which is a very interesting problem, but still open for now.

We conclude the section by describing what those assumptions mean for the EI model, starting from the following immediate result.

**Proposition 4.** *Under Assumption 4,  $\mathcal{G}_{\text{tw}}$  is an unsampleable HEGA.*

In conjunction with Remark 2, this means that our UC-EI model does not seem to impose any actual restrictions, and can almost be thought as equivalent to the plain UC model, with all the caveats of that remark.

## 5 Actively-Secure 2-Message Isogeny-Based OT

In this section we describe a 2-message OT protocol secure against malicious adversaries and give its security proof in the UC-EI model. We describe in Section 5.2 a variant of this protocol with 3 messages, but without the need of a trusted setup. Finally, a comparison of different isogeny-based OT protocols can be found in Section 5.3.

### 5.1 The Twist OT Protocol

The protocol  $\Pi_{\text{tw}}$ , described in Figure 2, is exactly the 2-message protocol proposed by Lai et al. [LGD21].

The protocol is actually type-safe w.r.t.  $\mathcal{G}_{\text{tw}}$ , since the only operations we do on curves are twists and CSIDH actions. In addition, both functionalities  $\mathcal{F}_{\text{TSC}}$  and  $\mathcal{F}_{\text{RO}}$  can be easily seen to be type-safe.<sup>3</sup>

This protocol was only proved to be semi-honest secure in the UC framework by Lai et al.; in the same paper, the authors give a maliciously-secure version of it that requires two additional messages to allow the extraction of the input of a malicious receiver in the UC proof. We now show that this is not needed in our UC-EI setting.

<sup>3</sup> $\mathcal{F}_{\text{TSC}}$  just samples  $(t, e)$  from  $\mathcal{D}_{\mathcal{G}_{\text{tw}}}$ , and then computes  $t \star x_0$  via an action gate.  $\mathcal{F}_{\text{RO}}$  can be emulated by lazily sampling a table of pairs  $(x, m_x)$ , where  $m_x$  is the random answer to the query  $x$ , and can be implemented using only equality gates.

**Theorem 3.** *The protocol  $\Pi_{\text{tw}}$ , described in Figure 2, UC-EI realizes the functionality  $\mathcal{F}_{\text{OT}}$  (Figure 4) in the  $(\mathcal{F}_{\text{RO}}, \mathcal{F}_{\text{TSC}})$ -hybrid model in the presence of malicious adversaries and static corruptions, if the encryption scheme  $(\text{KeyGen}, \text{Enc}, \text{Dec})$  is IND-CPA secure, the CSIDH vectorization problem is hard, and Assumption 1 holds.*

*Proof.* We distinguish the main two cases of honest  $P_S$  and corrupt  $P_R$  and corrupt  $P_S$  and honest  $P_R$ . Proving security in the remaining two cases is straightforward.

**HONEST SENDER AND CORRUPT RECEIVER.** We first describe the simulator  $\mathcal{S}$ , and in particular how it can extract the input of the corrupt receiver to forward to the OT functionality.

*Simulation.* Throughout the execution,  $\mathcal{S}$  simulates the random oracle  $H$  by answering every new query with a random value from the relevant set and maintaining a list of past queries to answer repeated queries consistently. More concretely,  $\mathcal{S}$  keeps a list  $L$  in  $\mathcal{E} \times \mathcal{K}$  in which it stores all the past queries. It initializes the random oracle with an empty list, then for each query  $X \in \mathcal{E}$  it checks whether  $(X, k) \in L$  for some  $k$ : if this is the case, returns that  $k$ , otherwise it samples a random  $k \leftarrow \mathcal{K}$ , adds  $(X, k)$  to  $L$  and returns  $k$ . The simulator is defined by the following instructions:

- Emulate the trusted setup step, defining the curve  $E = t \star E_0$ , with a randomly sampled  $t \in \text{Cl}(\mathcal{O})$ .
- Set its public key as the honest sender  $A = s \star E$ , for a random  $s \leftarrow \mathcal{K}$ .
- When receiving the curve  $C$  from the adversary, also obtain an explanation  $C = x \star E_R$ , with  $E_R = E_0, E$  or  $E^t$ . If  $E_R = E$  set  $\sigma = 0$ , if  $E_R = E^t$  set  $\sigma = 1$ , otherwise set  $\sigma = -1$ . If  $\sigma \neq -1$ , query the functionality and get the message  $m_\sigma$ .
- Sample two random keys  $k_i \leftarrow \text{KeyGen}()$  and for any additional query to  $H$  proceed as follows:
  - If query is  $s \star C$ , then if  $\sigma \neq 0$  send abort to  $\mathcal{F}_{\text{OT}}$ , otherwise returns  $k_0$
  - If query is  $s \star C^t$ , then if  $\sigma \neq 1$  send abort to  $\mathcal{F}_{\text{OT}}$ , otherwise returns  $k_1$
- Set any  $m_i$  that it doesn't know to 0 (i.e.  $m_{1-\sigma}$  if  $\sigma \in \{0, 1\}$ , both  $m_0, m_1$  otherwise); then it computes  $c_i = \text{Enc}_{k_i}(m_i)$ .
- Finally, send  $A, c_0, c_1$  to the adversary.

*Indistinguishability.* We now prove indistinguishability between the real and ideal execution. Let  $\mathcal{Z}_S$  denote  $\text{EXEC}_{\mathcal{F}, \mathcal{S}, \mathcal{Z}}$ , while  $\mathcal{Z}_\pi$  denote  $\text{EXEC}_{\pi, \mathcal{A}, \mathcal{Z}}$ . Let **Ab** be the event that  $\mathcal{S}$  aborts in an ideal execution. Write

$$s = \Pr[\mathcal{Z}_S = 1 \mid \text{Ab}], \quad s' = \Pr[\mathcal{Z}_S = 1 \mid \neg \text{Ab}], \quad p = \Pr[\mathcal{Z}_\pi = 1] \quad a = \Pr[\text{Ab}].$$

Then we have

$$\begin{aligned} \left| \Pr[\mathcal{Z}_S = 1] - \Pr[\mathcal{Z}_\pi = 1] \right| &= \left| sa + s'(1-a) - p \right| = \left| a(s-s') + s' - p \right| \\ &\leq 2a + |s' - p| = 2\Pr[\text{Ab}] + \left| \Pr[\mathcal{Z}_S = 1 \mid \neg \text{Ab}] - \Pr[\mathcal{Z}_\pi = 1] \right|. \end{aligned}$$

The theorem will then follow from the fact that both quantities are negligible. Indeed if  $\mathcal{S}$  aborts, then we can solve a vectorization CSIDH problem, while if  $\mathcal{Z}$  can distinguish we can break the IND-CPA property of the encryption scheme. More concretely, suppose that  $\mathcal{S}$  does not abort, then we can construct an adversary  $\mathcal{D}$  for the IND-CPA game as we describe in what follows.  $\mathcal{D}$  internally runs  $\mathcal{Z}$  against  $\mathcal{S}$ , but stopping the execution before the simulator computes  $c_{1-\sigma}$ . Then  $\mathcal{D}$  takes the input  $(m_0, m_1)$  for the honest sender, and sends to the IND-CPA oracle the pair of messages  $(0, m_{1-\sigma})$ , which returns

**Table 1:** The computable solutions to the CSIDH problem

|                                | $y \star E_0$   | $y \star E$       | $y \star E^t$       | $y \star A$        | $y \star A^t$        |
|--------------------------------|-----------------|-------------------|---------------------|--------------------|----------------------|
| $C = x \star E_0, s \star C$   | $s = yx^{-1}$   | $s = ytx^{-1}$    | $s = yt^{-1}x^{-1}$ | $t = xy^{-1}$      | $s^2 = x^{-1}t^{-1}$ |
| $C = x \star E_0, s \star C^t$ | $s = yx$        | $s = ytx$         | $s = yt^{-1}x$      | $t = x^{-1}y^{-1}$ | $s^2 = xt^{-1}$      |
| $C = x \star E, s \star C^t$   | $s = xy t$      | $s = xy t^2$      | $s = xy$            | $t = x^{-1}y^{-1}$ | $s^2 = xy$           |
| $C = x \star E^t, s \star C$   | $s = x^{-1}y t$ | $s = x^{-1}y t^2$ | $s = x^{-1}y$       | $t = xy^{-1}$      | $s^2 = x^{-1}y$      |

a ciphertext  $c$ . At this point  $\mathcal{D}$  resumes the execution, but it sets  $c_{1-\sigma} = c$ . Finally  $\mathcal{D}$  outputs whatever  $\mathcal{Z}$  outputs. Notice that when the bit  $b$  of the IND-CPA oracle is 1,  $\mathcal{D}$  runs a perfect emulation of the real protocol, while if  $b = 0$ ,  $\mathcal{D}$  is running  $\mathcal{S}$ . This means that

$$\text{Adv}_{\mathcal{D}, \mathcal{E}}^{\text{IND-CPA}} = \left| \Pr[\mathcal{D} = 1 \mid b = 0] - \Pr[\mathcal{D} = 1 \mid b = 1] \right| = \left| \Pr[\mathcal{Z}_{\mathcal{S}} = 1 \mid \neg \text{Ab}] - \Pr[\mathcal{Z}_{\pi} = 1] \right|.$$

In particular  $\mathcal{Z}$  cannot distinguish  $\mathcal{S}$  and the real world if the encryption scheme  $\mathcal{E}$  is IND-CPA, in the case that  $\mathcal{S}$  doesn't abort. We now estimate the probability of  $\mathcal{S}$  aborting. Suppose then that we have a CSIDH problem  $E_1 = a \star E_0$  we want to solve. We create two possible solvers for this problem:

- Algorithm  $\mathcal{D}_1$  will run  $\mathcal{S}$  with  $E_1$  as trusted setup. Then it will check if it can compute  $a$  from the queries and explanations that  $\mathcal{Z}$  makes to the random oracle.
- Algorithm  $\mathcal{D}_2$  will run  $\mathcal{S}$  with  $b \star E_0$  as trusted setup and  $b \star E_1$  as sender's public key. Then it will check queries to compute a value  $a'$  such that  $a' \star (b \star E_0) = b \star E_1$ , which means  $a' = a$ .

In Table 1, we show how  $\mathcal{D}_1$  and  $\mathcal{D}_2$  can compute the solutions from the query. The rows are indexed by the explanation of the curve  $C$  and the query, while the columns are the explanation of the query. We now compute the probability that the simulator aborts,  $\Pr[\mathcal{S} \text{ aborts}]$ , and estimate it with the advantages of  $\mathcal{D}_1, \mathcal{D}_2$  for the CSIDH problem. Let  $T_1$  be the event that  $\mathcal{Z}$  makes one of the "forbidden" queries and explains it as  $y \star A$  (so that  $t$  can be computed); let  $T_2$  be the event that a forbidden query is made and is explained differently from  $y \star A$  (in which case  $s$  can be computed). Notice that  $\mathcal{S}$  only aborts when a forbidden query is made, so we have that  $\Pr[\mathcal{S} \text{ aborts}] = \Pr[T_1] + \Pr[T_2]$ . Moreover  $\mathcal{D}_i$  wins with probability 1 if event  $T_i$  happens, so we have that

$$\text{Adv}_{\mathcal{D}_i}^{\text{csidh}} \geq 1 \cdot \Pr[T_i] + \frac{1}{\#\text{Cl}(\mathcal{O})} \Pr[\neg T_i] \geq \Pr[T_i].$$

In particular, we get that  $\Pr[\mathcal{S} \text{ aborts}] \leq \text{Adv}_{\mathcal{D}_0}^{\text{csidh}} + \text{Adv}_{\mathcal{D}_1}^{\text{csidh}}$ , from which we can finally conclude

$$\left| \Pr[\mathcal{Z}_{\mathcal{S}} = 1] - \Pr[\mathcal{Z}_{\pi} = 1] \right| \leq \text{Adv}_{\mathcal{D}}^{\text{IND-CPA}} + \text{Adv}_{\mathcal{D}_0}^{\text{csidh}} + \text{Adv}_{\mathcal{D}_1}^{\text{csidh}},$$

which proves indistinguishability, provided that the encryption scheme is IND-CPA and the CSIDH problem is hard.

**CORRUPT SENDER AND HONEST RECEIVER.** As in the previous case, we first describe the simulator and then we argue indistinguishability between the real and ideal execution.

*Simulation.* The simulator  $\mathcal{S}$  handles random oracles queries as in the previous case and does the following.

- Backdoor the trusted setup as before: sample  $t \leftarrow_{\$} \text{Cl}(\mathcal{O})$  and set  $E = t \star E_0$
- Sample  $r \leftarrow_{\$} \text{Cl}(\mathcal{O})$  and compute  $C = r \star E$ . Set  $\sigma = 0$  and send  $C$  to  $\mathcal{A}$ . Then proceed as an honest party would do.



- If, at some point,  $\mathcal{A}$  sends `abort`, then forward `abort` to the OT functionality.
- If receive  $(A, c_0, c_1)$ , together with an explanation for  $A$ , from  $\mathcal{A}$ , using  $t$  can recover both the keys  $k_0$  and  $k_1$  as follows:  $k_0 = H(r \star A)$  and  $k_1 = H(r^{-1}t^{-2} \star A)$ ; then decrypt the two messages with the computed keys  $m_i = \text{Dec}_{k_i}(c_i)$ . Send  $m_0, m_1$  to the functionality.
- Output whatever the adversary outputs and halt.

*Indistinguishability.* In the trusted setup, the simulator backdoors the public curve, but this is unnoticeable to the adversary. After the setup phase, in the real protocol the curve  $C$  sent by the receiver is either  $r \star E$  or  $(r \star E)^t$  depending on whether  $\sigma = 0$  or  $\sigma = 1$ , respectively. In the former case the messages received by  $\mathcal{A}$  are identically distributed in the two executions, in the latter case, by Assumption 1 the messages are computationally indistinguishable, since both  $r \star E$  and  $(r \star E)^t$  are sampled from distributions that are computationally indistinguishable from uniform (notice that the twist of the uniform distribution is again uniform). Finally, using its knowledge of  $t$  it is straightforward to see that  $\mathcal{S}$  is able to correctly extract the input of  $\mathcal{A}$ . Therefore we can conclude that the two executions are indistinguishable.  $\square$

In conclusion, by proving UC-EI security for the protocol  $\Pi_{\text{tw}}$ , we can claim security against malicious adversaries of a two-round OT protocol that only requires 3 isogeny computations for the sender and 2 for the receiver. Removing one of the two assumptions, i.e., either ROM or TSC, without compromising the efficiency of the protocol, remains a fascinating open question, whose answer will probably require a completely different approach.

## 5.2 A 3-message variant

We can apply our model to prove UC-EI security of other isogeny-based OT protocols. We decided to only show this for a variant of  $\Pi_{\text{tw}}$  that does not require a trusted setup, but unfortunately needs 3 messages, and thus it's not optimal. The protocol  $\Pi_{\text{tw}^3}$ , described in Figure 3, is similar to  $\Pi_{\text{tw}}$ , except that now it is the sender  $P_S$  that generates the curve  $E$  and sends it to the receiver  $P_R$ .

**Theorem 4.** *The protocol  $\Pi_{\text{tw}^3}$ , described in Figure 3, UC-EI realizes the functionality  $\mathcal{F}_{\text{OT}}$  in the random oracle model if the CSIDH vectorization problem is hard and the encryption scheme (KeyGen, Enc, Dec) is IND-CPA secure.*

*Sketch.* As before we distinguish between the two main cases of corrupt sender and honest receiver and honest sender and corrupt receiver.

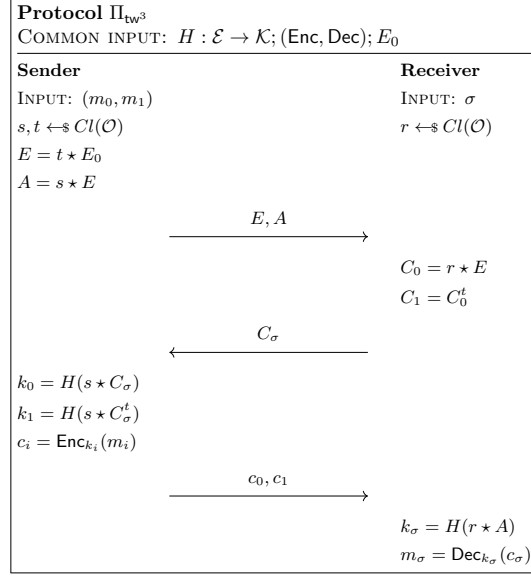
**CORRUPT SENDER AND HONEST RECEIVER.** As in the other proof, the twisting statistically hides the choice bit, since the lemma holds for any starting curve, even possibly ones that have been maliciously generated.

**HONEST SENDER AND CORRUPT RECEIVER.** This also works as in the other proof, since by simulating the honest sender we know  $t$  such that  $E = t \star E_0$ , which otherwise we knew by simulating the trusted setup functionality.  $\square$

## 5.3 Efficiency and Comparison

In Table 2, we give a detailed comparison between the protocol  $\Pi_{\text{tw}}$  presented in Section 5.1 and other isogeny-based OT protocols that have been proposed in the last few years; we considered only the ones whose security relies on CSIDH or variants.

In the table, we report the number of messages  $\#M$ , the number of isogenies computed by the sender and the receiver, respectively,  $\#(\text{PK}_S, \text{PK}_R)$ , the power of the adversary,



**Figure 3:** The twist protocol without trusted setup

**Table 2:** Comparison of some properties of proposed OT protocols

| Reference                                       | # M | # (PK <sub>S</sub> , PK <sub>R</sub> ) | Adversary   | Proof      | Model   | Assumption     |
|---|-----|--|-------------|------------|---------|----------------|
| [DOPS20] I                                      | 2   | 3, 2                                   | Semi-honest | UC         | ROM     | ParallelEither |
| [LGD21] I                                       | 2   | 3, 2                                   | Semi-honest | UC         | ROM+TSC | Inv-CSIDH      |
| [ADMP20]  | 2   | $4\lambda, \lambda + 3$                | Malicious   | SSP        | Plain   | wPR-EGA        |
| [ADMP20] + [PVW08]                              | 2   | $4\lambda, \lambda$                    | Malicious   | UC         | CRS     | wPR-EGA        |
| [DOPS20] + [DGH <sup>+</sup> 20]                | 2   | $\text{poly}(\lambda)$                 | Malicious   | UC         | ROM+TSC | ParallelDouble |
| [LGD21] II                                      | 4   | 5, 6                                   | Malicious   | UC         | ROM+TSC | Rec-CSIDH      |
| [BMM <sup>+</sup> 23] I                         | 2   | $O(\lambda), O(\lambda)$               | Malicious   | UC         | ROM+CRS | Comp-CSIDH     |
| [BMM <sup>+</sup> 23] II                        | 4   | $O(\lambda), O(\lambda)$               | Malicious   | Simulation | Plain   | Comp-CSIDH     |
| $\Pi_{\text{tw}}$ ([LGD21] + <b>this work</b> ) | 2   | 3, 2                                   | Malicious   | UC-EI      | ROM+TSC | Vec-CSIDH      |
| $\Pi_{\text{tw}^3}$ <b>this work</b>            | 3   | 4, 2                                   | Malicious   | UC-EI      | ROM     | Vec-CSIDH      |

the proof framework, the model of computation, and the underlying hardness assumption on which the security of the protocol is based on. We denote the security parameter by  $\lambda$ .

A brief description of some of the works cited in Table 2 was already given in [LGD21]. We recall here their main properties for completeness. In the first rows, we have semi-honest secure protocols.

In [DOPS20], the authors introduce the concept of *masking*, which generalizes the one of hard homogeneous spaces. The paper contains two passively secure OT protocols, one with two rounds, derived from the Shamir-3-Pass key transportation scheme, and the other with three rounds derived from the CO protocol. In addition, the authors prove that their two-round protocol can be extended to be secure against malicious adversaries using a transformation by Döttling et al. [DGH<sup>+</sup>20], which increases the complexity of the protocol as a side effect. As mentioned before, the protocols are based on masking assumptions, ParallelEither, ParallelBoth and ParallelDouble, that can be instantiated with isogeny-based assumptions. The ParallelEither asks for either  $g^{ab}$  or  $g^{a/b}$  given  $g^a, g^b$ ; the ParallelBoth asks for  $g^{ba_0/a_1}$  or  $g^{ba_1/a_0}$  given  $g^{a_0}, g^{a_1}, g^b$ ; the ParallelDouble asks for  $g^{ac}$  and  $g^{bc}$  given  $g^a, g^b, g^c$  and a one-time access to an oracle that exponentiates by  $c$ . We refer to [DOPS20] for additional details.

In [ADMP20], the authors introduce a new framework based on group actions, from which they derive new cryptographic primitives based on CSIDH. In particular, they

construct a “dual-mode encryption scheme” which allows them to use the framework by Peikert et al. [PVW08] to produce an OT protocol that is UC-secure against malicious adversaries. The resulting protocols has only two rounds, but it needs to generate  $O(\lambda)$  public keys and compute  $O(\lambda)$  isogenies. They also directly build a statistically sender-private OT protocol, which still needs  $O(\lambda)$  public key operations. The security of these protocols relies on the weak pseudorandomness of the underlying EGA.

The recent paper [BMM<sup>+</sup>23] constructs a 4-round maliciously secure protocol in the plain model, and a 2-round UC-secure protocol in the ROM+CRS model. They start from a semi-honest protocol and add a proof of knowledge to show that the messages are well-formed. Both protocols are based on the computational CSIDH problem, but need a number of isogeny computation that is linear in the security parameter  $\lambda$ , which mainly comes from the PoK.

Lai et al. [LGD21] describe the 2-round semi-honest protocol that we prove to be maliciously secure in the UC-EI model. This protocol is particularly efficient compared to other isogeny-based protocols as it requires only 2 round of communications and a constant, very low number of isogeny computations. The protocol requires both a random oracle and a trusted setup.

## References

- [ABK<sup>+</sup>21] Michel Abdalla, Manuel Barbosa, Jonathan Katz, Julian Loss, and Jiayu Xu. Algebraic adversaries in the universal composability framework. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part III*, volume 13092 of *LNCS*, pages 311–341. Springer, Heidelberg, December 2021. [doi:10.1007/978-3-030-92078-4\\_11](https://doi.org/10.1007/978-3-030-92078-4_11).
- [ADMP20] Navid Alamati, Luca De Feo, Hart Montgomery, and Sikhar Patranabis. Cryptographic group actions and applications. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 411–439. Springer, Heidelberg, December 2020. [doi:10.1007/978-3-030-64834-3\\_14](https://doi.org/10.1007/978-3-030-64834-3_14).
- [AEK<sup>+</sup>22] Michel Abdalla, Thorsten Eisenhofer, Eike Kiltz, Sabrina Kunzweiler, and Doreen Riepel. Password-authenticated key exchange from group actions. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part II*, volume 13508 of *LNCS*, pages 699–728. Springer, Heidelberg, August 2022. [doi:10.1007/978-3-031-15979-4\\_24](https://doi.org/10.1007/978-3-031-15979-4_24).
- [AIR01] William Aiello, Yuval Ishai, and Omer Reingold. Priced oblivious transfer: How to sell digital goods. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 119–135. Springer, Heidelberg, May 2001. [doi:10.1007/3-540-44987-6\\_8](https://doi.org/10.1007/3-540-44987-6_8).
- [BBD<sup>+</sup>22] Jeremy Booher, Ross Bowden, Javad Doliskani, Tako Boris Fouotsa, Steven D. Galbraith, Sabrina Kunzweiler, Simon-Philipp Merz, Christophe Petit, Benjamin Smith, Katherine E. Stange, Yan Bo Ti, Christelle Vincent, José Felipe Voloch, Charlotte Weitkämper, and Lukas Zobernig. Failing to hash into supersingular isogeny graphs. Cryptology ePrint Archive, Report 2022/518, 2022. <https://eprint.iacr.org/2022/518>.
- [BD18] Zvika Brakerski and Nico Döttling. Two-message statistically sender-private OT from LWE. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part II*, volume 11240 of *LNCS*, pages 370–390. Springer, Heidelberg, November 2018. [doi:10.1007/978-3-030-03810-6\\_14](https://doi.org/10.1007/978-3-030-03810-6_14).

- [BDD<sup>+</sup>17] Paulo S. L. M. Barreto, Bernardo David, Rafael Dowsley, Kirill Morozov, and Anderson C. A. Nascimento. A framework for efficient adaptively secure composable oblivious transfer in the ROM. Cryptology ePrint Archive, Report 2017/993, 2017. <https://eprint.iacr.org/2017/993>.
- [BDGM19] Pedro Branco, Jintai Ding, Manuel Goulão, and Paulo Mateus. A framework for universally composable oblivious transfer from one-round key-exchange. In Martin Albrecht, editor, *17th IMA International Conference on Cryptography and Coding*, volume 11929 of *LNCS*, pages 78–101. Springer, Heidelberg, December 2019. [doi:10.1007/978-3-030-35199-1\\_5](https://doi.org/10.1007/978-3-030-35199-1_5).
- [BKV19] Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. CSI-FiSh: Efficient isogeny based signatures through class group computations. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 227–247. Springer, Heidelberg, December 2019. [doi:10.1007/978-3-030-34578-5\\_9](https://doi.org/10.1007/978-3-030-34578-5_9).
- [BLN<sup>+</sup>21] Sai Sheshank Burra, Enrique Larraia, Jesper Buus Nielsen, Peter Sebastian Nordholt, Claudio Orlandi, Emmanuela Orsini, Peter Scholl, and Nigel P. Smart. High-performance multi-party computation for binary circuits based on oblivious transfer. *Journal of Cryptology*, 34(3):34, July 2021. [doi:10.1007/s00145-021-09403-1](https://doi.org/10.1007/s00145-021-09403-1).
- [BM90] Mihir Bellare and Silvio Micali. Non-interactive oblivious transfer and applications. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 547–557. Springer, Heidelberg, August 1990. [doi:10.1007/0-387-34805-0\\_48](https://doi.org/10.1007/0-387-34805-0_48).
- [BMM<sup>+</sup>23] Saikrishna Badrinarayanan, Daniel Masny, Pratyay Mukherjee, Sikhar Patranabis, Srinivasan Raghuraman, and Pratik Sarkar. Round-optimal oblivious transfer and MPC from computational CSIDH. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 376–405. Springer, Heidelberg, May 2023. [doi:10.1007/978-3-031-31368-4\\_14](https://doi.org/10.1007/978-3-031-31368-4_14).
- [BY91] Gilles Brassard and Moti Yung. One-way group actions. In Alfred J. Menezes and Scott A. Vanstone, editors, *CRYPTO'90*, volume 537 of *LNCS*, pages 94–107. Springer, Heidelberg, August 1991. [doi:10.1007/3-540-38424-3\\_7](https://doi.org/10.1007/3-540-38424-3_7).
- [Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145. IEEE Computer Society Press, October 2001. [doi:10.1109/SFCS.2001.959888](https://doi.org/10.1109/SFCS.2001.959888).
- [CD23] Wouter Castryck and Thomas Decru. An efficient key recovery attack on SIDH. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 423–447. Springer, Heidelberg, April 2023. [doi:10.1007/978-3-031-30589-4\\_15](https://doi.org/10.1007/978-3-031-30589-4_15).
- [CDPW07] Ran Canetti, Yevgeniy Dodis, Rafael Pass, and Shabsi Walfish. Universally composable security with global setup. In Salil P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 61–85. Springer, Heidelberg, February 2007. [doi:10.1007/978-3-540-70936-7\\_4](https://doi.org/10.1007/978-3-540-70936-7_4).
- [CLM<sup>+</sup>18] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part III*, volume 11274 of *LNCS*, pages 395–427. Springer, Heidelberg, December 2018. [doi:10.1007/978-3-030-03332-3\\_15](https://doi.org/10.1007/978-3-030-03332-3_15).

- [CO15] Tung Chou and Claudio Orlandi. The simplest protocol for oblivious transfer. In Kristin E. Lauter and Francisco Rodríguez-Henríquez, editors, *LATIN-CRYPT 2015*, volume 9230 of *LNCS*, pages 40–58. Springer, Heidelberg, August 2015. doi:10.1007/978-3-319-22174-8\_3.
- [Cou06] Jean-Marc Couveignes. Hard homogeneous spaces. Cryptology ePrint Archive, Report 2006/291, 2006. <https://eprint.iacr.org/2006/291>.
- [CPV20] Wouter Castryck, Lorenz Panny, and Frederik Vercauteren. Rational isogenies from irrational endomorphisms. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 523–548. Springer, Heidelberg, May 2020. doi:10.1007/978-3-030-45724-2\_18.
- [DCW13] Changyu Dong, Liqun Chen, and Zikai Wen. When private set intersection meets big data: an efficient and scalable protocol. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013*, pages 789–800. ACM Press, November 2013. doi:10.1145/2508859.2516701.
- [DDN14] Bernardo David, Rafael Dowsley, and Anderson C. A. Nascimento. Universally composable oblivious transfer based on a variant of LPN. In Dimitris Gritzalis, Aggelos Kiayias, and Ioannis G. Askoxylakis, editors, *CANS 14*, volume 8813 of *LNCS*, pages 143–158. Springer, Heidelberg, October 2014. doi:10.1007/978-3-319-12280-9\_10.
- [DGH<sup>+</sup>20] Nico Döttling, Sanjam Garg, Mohammad Hajiabadi, Daniel Masny, and Daniel Wichs. Two-round oblivious transfer from CDH or LPN. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 768–797. Springer, Heidelberg, May 2020. doi:10.1007/978-3-030-45724-2\_26.
- [DGI<sup>+</sup>19] Nico Döttling, Sanjam Garg, Yuval Ishai, Giulio Malavolta, Tamer Mour, and Rafail Ostrovsky. Trapdoor hash functions and their applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 3–32. Springer, Heidelberg, August 2019. doi:10.1007/978-3-030-26954-8\_1.
- [DHK<sup>+</sup>23] Julien Duman, Dominik Hartmann, Eike Kiltz, Sabrina Kunzweiler, Jonas Lehmann, and Doreen Riepel. Generic models for group actions. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 406–435. Springer, Heidelberg, May 2023. doi:10.1007/978-3-031-31368-4\_15.
- [DOPS20] Cyprien Delpech de Saint Guilhem, Emmanuela Orsini, Christophe Petit, and Nigel P. Smart. Semi-commutative masking: A framework for isogeny-based protocols, with an application to fully secure two-round isogeny-based OT. In Stephan Krenn, Haya Shulman, and Serge Vaudenay, editors, *CANS 20*, volume 12579 of *LNCS*, pages 235–258. Springer, Heidelberg, December 2020. doi:10.1007/978-3-030-65411-5\_12.
- [Dv22] Léo Ducas and Wessel P. J. van Woerden. On the lattice isomorphism problem, quadratic forms, remarkable lattices, and cryptography. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part III*, volume 13277 of *LNCS*, pages 643–673. Springer, Heidelberg, May / June 2022. doi:10.1007/978-3-031-07082-2\_23.



- [DvMN08] Rafael Dowsley, Jeroen van de Graaf, Jörn Müller-Quade, and Anderson C. A. Nascimento. Oblivious transfer based on the McEliece assumptions. In Reihaneh Safavi-Naini, editor, *ICITS 08*, volume 5155 of *LNCS*, pages 107–117. Springer, Heidelberg, August 2008. doi:10.1007/978-3-540-85093-9\_11.
- [EGL82] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *CRYPTO'82*, pages 205–210. Plenum Press, New York, USA, 1982.
- [Fel19] Joël Felderhoff. Hard Homogenous Spaces and Commutative Supersingular Isogeny based Diffie-Hellman. Internship report, LIX, Ecole polytechnique ; ENS de Lyon, August 2019. URL: <https://hal.archives-ouvertes.fr/hal-02373179>.
- [Feo17] Luca De Feo. Mathematics of isogeny based cryptography. *CoRR*, abs/1711.04062, 2017. URL: <http://arxiv.org/abs/1711.04062>.
- [FKL18] Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 33–62. Springer, Heidelberg, August 2018. doi:10.1007/978-3-319-96881-0\_2.
- [GO94] Oded Goldreich and Yair Oren. Definitions and properties of zero-knowledge proof systems. *Journal of Cryptology*, 7(1):1–32, December 1994. doi:10.1007/BF00195207.
- [GPSV21] Steven Galbraith, Lorenz Panny, Benjamin Smith, and Frederik Vercauteren. Quantum equivalence of the DLP and CDHP for group actions. *Mathematical Cryptology*, 1(1):40–44, Jun. 2021. URL: <https://journals.flvc.org/mathcryptology/article/view/122741>.
- [GS10] Dima Grigoriev and Vladimir Shpilrain. Authentication schemes from actions on graphs, groups, or rings. *Annals of Pure and Applied Logic*, 162(3):194–200, 2010. Special Issue: Dedicated to Nikolai Alexandrovich Shanin on the occasion of his 90th birthday. URL: <https://www.sciencedirect.com/science/article/pii/S0168007210001181>, doi:10.1016/j.apal.2010.09.004.
- [HK12] Shai Halevi and Yael Tauman Kalai. Smooth projective hashing and two-message oblivious transfer. *Journal of Cryptology*, 25(1):158–193, January 2012. doi:10.1007/s00145-010-9092-8.
- [JAC<sup>+</sup>20] David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, Vladimir Soukharev, David Urbanik, Geovandro Pereira, Koray Karabina, and Aaron Hutchinson. SIKE. Technical report, National Institute of Standards and Technology, 2020. available at <https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions>.
- [JD11] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Bo-Yin Yang, editor, *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011*, pages 19–34. Springer, Heidelberg, November / December 2011. doi:10.1007/978-3-642-25405-5\_2.

- [JQSY19] Zhengfeng Ji, Youming Qiao, Fang Song, and Aaram Yun. General linear group action on tensors: A candidate for post-quantum cryptography. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019, Part I*, volume 11891 of *LNCS*, pages 251–281. Springer, Heidelberg, December 2019. doi:[10.1007/978-3-030-36030-6\\_11](https://doi.org/10.1007/978-3-030-36030-6_11).
- [Kan97] Ernst Kani. The number of curves of genus two with elliptic differentials. *Journal für die reine und angewandte Mathematik (Crelles Journal)*, 1997:122–93, 1997.
- [KOS16] Marcel Keller, Emmanuela Orsini, and Peter Scholl. MASCOT: Faster malicious arithmetic secure computation with oblivious transfer. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016*, pages 830–842. ACM Press, October 2016. doi:[10.1145/2976749.2978357](https://doi.org/10.1145/2976749.2978357).
- [LGD21] Yi-Fu Lai, Steven D. Galbraith, and Cyprien Delpech de Saint Guilhem. Compact, efficient and UC-secure isogeny-based oblivious transfer. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 213–241. Springer, Heidelberg, October 2021. doi:[10.1007/978-3-030-77870-5\\_8](https://doi.org/10.1007/978-3-030-77870-5_8).
- [Mau05] Ueli M. Maurer. Abstract models of computation in cryptography (invited paper). In Nigel P. Smart, editor, *10th IMA International Conference on Cryptography and Coding*, volume 3796 of *LNCS*, pages 1–12. Springer, Heidelberg, December 2005.
- [MMP22] Marzio Mula, Nadir Murru, and Federico Pintore. Random sampling of supersingular elliptic curves. Cryptology ePrint Archive, Report 2022/528, 2022. <https://eprint.iacr.org/2022/528>.
- [MMP<sup>+</sup>23] Luciano Maino, Chloe Martindale, Lorenz Panny, Giacomo Pope, and Benjamin Wesolowski. A direct key recovery attack on SIDH. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 448–471. Springer, Heidelberg, April 2023. doi:[10.1007/978-3-031-30589-4\\_16](https://doi.org/10.1007/978-3-031-30589-4_16).
- [MS20] Daniele Micciancio and Jessica Sorrell. Simpler statistically sender private oblivious transfer from ideals of cyclotomic integers. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 381–407. Springer, Heidelberg, December 2020. doi:[10.1007/978-3-030-64834-3\\_13](https://doi.org/10.1007/978-3-030-64834-3_13).
- [MZ22] Hart Montgomery and Mark Zhandry. Full quantum equivalence of group action DLog and CDH, and more. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part I*, volume 13791 of *LNCS*, pages 3–32. Springer, Heidelberg, December 2022. doi:[10.1007/978-3-031-22963-3\\_1](https://doi.org/10.1007/978-3-031-22963-3_1).
- [NP01] Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In S. Rao Kosaraju, editor, *12th SODA*, pages 448–457. ACM-SIAM, January 2001.
- [Pan23] Lorenz Panny. CSI-FiSh really isn’t polynomial-time, 2023. URL: <https://yx7.cc/blah/2023-04-14.html>.
- [PSZ14] Benny Pinkas, Thomas Schneider, and Michael Zohner. Faster private set intersection based on OT extension. In Kevin Fu and Jaeyeon Jung, editors, *USENIX Security 2014*, pages 797–812. USENIX Association, August 2014.

- [PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 554–571. Springer, Heidelberg, August 2008. doi:10.1007/978-3-540-85174-5\_31.
- [Rab05] Michael O. Rabin. How to exchange secrets with oblivious transfer. Cryptology ePrint Archive, Report 2005/187, 2005. <https://eprint.iacr.org/2005/187>.
- [Rob23] Damien Robert. Breaking SIDH in polynomial time. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 472–503. Springer, Heidelberg, April 2023. doi:10.1007/978-3-031-30589-4\_17.
- [RS06] Alexander Rostovtsev and Anton Stolbunov. Public-Key Cryptosystem Based On Isogenies. Cryptology ePrint Archive, Report 2006/145, 2006. <https://eprint.iacr.org/2006/145>.
- [Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 256–266. Springer, Heidelberg, May 1997. doi:10.1007/3-540-69053-0\_18.
- [Sil09] J.H. Silverman. *The Arithmetic of Elliptic Curves*. Graduate Texts in Mathematics. Springer New York, 2009. URL: [https://books.google.it/books?id=Z9OCA\\_EUCCkC](https://books.google.it/books?id=Z9OCA_EUCCkC).
- [Vél71] Jacques Vélou. Isogénies entre courbes elliptiques. *Comptes Rendus de l'Académie des Sciences de Paris*, 273:238–241, July 1971. URL: <https://gallica.bnf.fr/ark:/12148/bpt6k56191248/f52.item>.
- [Vit19] Vanessa Vitse. Simple oblivious transfer protocols compatible with supersingular isogenies. In Johannes Buchmann, Abderrahmane Nitaj, and Tajje eddine Rachidi, editors, *AFRICACRYPT 19*, volume 11627 of *LNCS*, pages 56–78. Springer, Heidelberg, July 2019. doi:10.1007/978-3-030-23696-0\_4.
- [Wes22a] Benjamin Wesolowski. Orientations and the supersingular endomorphism ring problem. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part III*, volume 13277 of *LNCS*, pages 345–371. Springer, Heidelberg, May / June 2022. doi:10.1007/978-3-031-07082-2\_13.
- [Wes22b] Benjamin Wesolowski. The supersingular isogeny path and endomorphism ring problems are equivalent. In *62nd FOCS*, pages 1100–1111. IEEE Computer Society Press, February 2022. doi:10.1109/FOCS52979.2021.00109.
- [Zha22] Mark Zhandry. To label, or not to label (in generic groups). In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part III*, volume 13509 of *LNCS*, pages 66–96. Springer, Heidelberg, August 2022. doi:10.1007/978-3-031-15982-4\_3.
- [ZLWR13] Bingsheng Zhang, Helger Lipmaa, Cong Wang, and Kui Ren. Practical fully simulatable oblivious transfer with sublinear communication. In Ahmad-Reza Sadeghi, editor, *FC 2013*, volume 7859 of *LNCS*, pages 78–95. Springer, Heidelberg, April 2013. doi:10.1007/978-3-642-39884-1\_8.
- [ZZK22] Cong Zhang, Hong-Sheng Zhou, and Jonathan Katz. An analysis of the algebraic group model. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part IV*, volume 13794 of *LNCS*, pages 310–322. Springer, Heidelberg, December 2022. doi:10.1007/978-3-031-22972-5\_11.

## Auxiliary Supporting Material

### A Additional Preliminaries

#### A.1 Elliptic Curves, Isogenies, Endomorphisms

An elliptic curve is a smooth curve of genus one with a distinguished rational point. More concretely, an elliptic curve  $E$  defined over a field  $K$  (denoted  $E/K$ ) is the set of points satisfying the Weierstrass equation  $y^2 = x^3 + ax + b$ , with  $a, b \in K$ , with an additional “point at infinity”. The points of an elliptic curve, denoted by  $E(K)$ , have an abelian and algebraic group structure given by intersecting lines with the curve. Given two elliptic curves  $E_1$  and  $E_2$  over  $K$ , an *isogeny*  $\psi$  between  $E_1$  and  $E_2$  is a non-constant morphism between them. The degree of an isogeny is its degree as a rational map; an isogeny is said to be *separable* if its degree is equal to the size of its kernel. Given a finite subgroup  $G$  of  $E$ , there exists a separable isogeny  $\psi : E \rightarrow E/G$ , with  $\ker \psi = G$ , which is unique up to isomorphism. Both the isogeny  $\psi$  and image  $E/G$  can be computed from the kernel using Vélu’s formulas [Vél71], whose efficiency depends on the smoothness of the isogeny degree.

An *endomorphism* of an elliptic curve  $E$  is an isogeny from  $E$  to itself. The set  $\text{End}(E)$  of endomorphisms of  $E$ , together with the zero map, is a ring. When  $E$  is defined over a finite field, the endomorphism ring of  $E$  is either an order in a quadratic field, in which case we say  $E$  is *ordinary*, or a maximal order in a quaternion algebra and  $E$  is called *supersingular*. For more background on elliptic curves, isogenies and their use in cryptography we refer to standard resources [Sil09, Feo17].

#### A.2 Other Computational Assumptions

The two main problems of isogeny based cryptography are the *Endomorphism Ring Problem* (**EndRing**) and the *Isogeny Path Problem* (**IsogenyPath**), which can be stated as follows.

**Problem 5** (**IsogenyPath**). *Given two isogenous supersingular curves  $E, E'$ , compute an isogeny  $\phi : E \rightarrow E'$ .*

This problem deals with the computational complexity of finding a path in the isogeny graph of all supersingular curves. Indeed, we know that for any pair of curves having the same number of points, there exists an isogeny between them; the problem then asks to explicitly compute it.

**Problem 6** (**EndRing**). *Given a supersingular elliptic curve  $E$  over  $\mathbb{F}_{p^2}$ , compute  $\text{End}(E)$ .*

Since  $\text{End}(E)$  is a rank 4 lattice for supersingular curves, finding even one extra endomorphisms that is not “trivial” (i.e. a combination of scalar multiplications and the Frobenius) seems to be very hard.

One of the most important results for isogeny based cryptography is the following theorem that relates **EndRing** to **IsogenyPath**.

**Theorem 5** ([Wes22b]). *The problems **EndRing** and **IsogenyPath** are equivalent, under the generalised Riemann hypothesis (GRH).*

We can also try to understand the relationship between CSIDH, where curves and isogenies are constrained to be  $\mathbb{F}_p$ -rational, and the general **EndRing** problem. In [CPV20], the authors describe an efficient algorithm that, given two curves  $E_1, E_2$  over  $\mathbb{F}_p$  and their full endomorphism rings, outputs a class group element  $\mathfrak{a}$  such that  $\mathfrak{a} \cdot E_1 = E_2$ . The problem is that this ideal usually doesn’t have a smooth norm, so its action cannot be efficiently computed. In order to smoothen the norm, we need to find relations in  $Cl(\mathcal{O})$  and then run lattice reduction algorithms, which greatly increase complexity. This previous result can be made effective, and we actually have the following equivalence.

**Theorem 6** ([Wes22a]). *EndRing and the effective CSIDH vectorization problem are equivalent, under GRH.*

Other computational problems related to CSIDH are the following; these are mainly defined and studied in [Fel19] and [LGD21].

**Problem 7** (Computational Inverse CSIDH (Inv-CSIDH)). *Given curves  $E, r \star E$  in  $\mathcal{E}$  with  $s \in \text{Cl}(\mathcal{O})$ , the problem asks to find  $E' \in \mathcal{E}$  such that  $E' = r^{-1} \star E$ .*

Given that  $E_0^t = E_0$ , we have that  $(\mathfrak{a} \star E_0)^t = \mathfrak{a}^{-1} \star E_0$ , so the above problem is easy in the special case of  $E = E_0$ .

**Problem 8** (Computational Reciprocal CSIDH (Rec-CSIDH)). *Given  $E \in \mathcal{E}$ , first the adversary chooses and commits to  $X \in \mathcal{E}$ , then it receives the challenge  $s \star E, s \in \text{Cl}(\mathcal{O})$ . The adversary wins if it can compute  $(s \star X, s^{-1} \star X)$  w.r.t.  $X$ .*

We have the following reductions between these problems.

**Proposition 5.** *The computational reciprocal CSIDH problem is equivalent to the computational inverse CSIDH problem.*

**Proposition 6** ([Fel19], [LGD21]). *If the group  $\text{Cl}(\mathcal{O})$  has known order, the computational CSIDH problem is equivalent to the computational inverse problem.*

This also means that the two problems are *quantumly* equivalent, since the computation of the class group can be done in quantum polynomial time.

Another fundamental quantum equivalence is the following.

**Theorem 7** ([GPSV21], [Wes22a], [MZ22]). *The vectorization problem and the computational CSIDH problem are quantumly equivalent.*

### A.3 Overview of the UC Framework

We present here a semi-formal overview of the universally composable (UC) model of security established by Canetti [Can01]. The UC framework allows for defining security properties tasks so that security is maintained under general composition with unbounded number of instances of arbitrary protocols running concurrently.

Protocols that aim to achieve security in this model are defined in three steps. First, the protocol and its execution in the presence of an adversary are formalized, this represents the real-life model which we also call the *real world*. Next, an ideal process for executing the task is defined; its role is to act as a trusted party by separately receiving the input of each party, honestly computing the result of the protocol internally and returning the output assigned to each party. In this ideal world, the parties do not communicate with one another but instead solely rely on the ideal functionality to provide them with their output. Finally, we say that the protocol in question UC-realizes the ideal functionality if running the protocol is equivalent to emulating the ideal functionality. Below, we provide a brief discussion with additional formal details.

*Real model.* An execution of a protocol  $\pi$  in the real model consists of  $n$  PPT *interactive Turing machines* (ITMs)  $P_1, \dots, P_n$  representing the computing parties. Each party is uniquely determined by a *party identifier* (ID) that is used to distinguish between participants of the same protocol instance. We also have two additional ITMs, an adversary  $\mathcal{A}$ , describing the behaviour of the corrupted parties and an environment  $\mathcal{Z}$ , representing the external network environment in which the protocol operates. The environment gives inputs to the honest parties, receives their outputs, and can communicate with the adversary at any point during the execution. The adversary controls the operations of the corrupted parties and the delivery of messages between the parties. In more details, when



the environment is activated, it can read the output tapes of all honest parties and  $\mathcal{A}$ , and it can activate a subset of the parties and  $\mathcal{A}$  by writing input messages on their input tapes. Parties, once activated, can perform local computation, write on their output tape, send messages to other parties and send subroutine output messages to  $\mathcal{Z}$ . The adversary  $\mathcal{A}$  can send backdoor messages to  $\mathcal{Z}$  and all parties, and receive backdoor messages from all parties. We say that the adversary is *passive* (or *semi-honest*) if it always instructs the corrupt parties to follow the protocol; while we say that the adversary is malicious if it may instruct the corrupt parties to arbitrarily deviate from the protocol's instructions. We let  $\text{EXEC}_{\pi, \mathcal{A}, \mathcal{Z}}(\kappa, z, \mathbf{r})$  denote  $\mathcal{Z}$ 's output on input  $z$  and security parameter  $\kappa$ , after interacting with  $\mathcal{A}, P_1, \dots, P_n$  running the protocol  $\Pi$  with random tape  $\mathbf{r}$ .

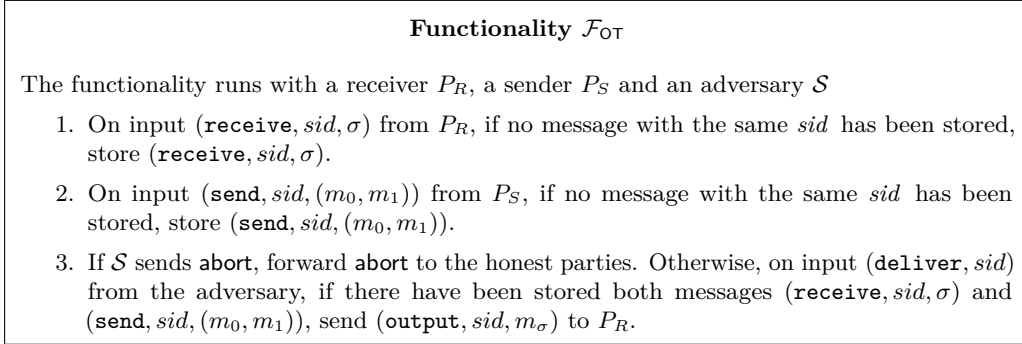
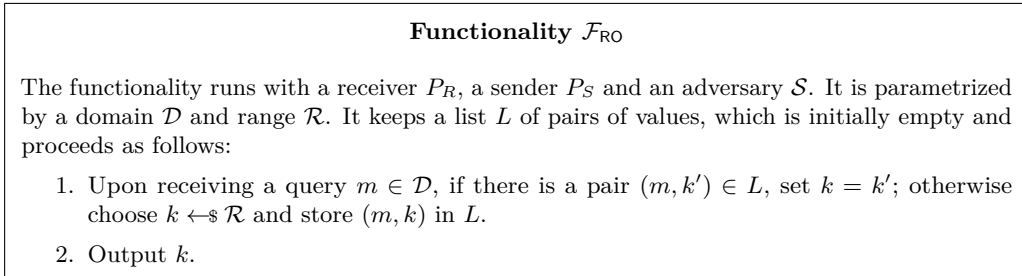
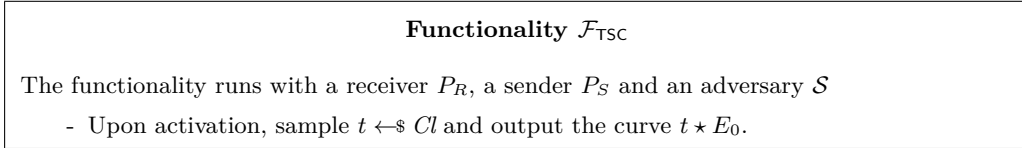
*Ideal model.* In the ideal protocol  $\text{IDEAL}_{\mathcal{F}}$ , we have  $n$  dummy parties  $P_1, \dots, P_n$  which interact with an ideal functionality  $\mathcal{F}$  in a simple way: they pass their private inputs to  $\mathcal{F}$  and wait for it to return their assigned output. There is also an ideal-adversary  $\mathcal{S}$  which is responsible for the delivery of messages. The ideal functionality  $\mathcal{F}$  defines the desired behaviour of the computation, playing the role of a trusted third party, and can communicate with the ideal adversary  $\mathcal{S}$  by providing and receiving backdoor information. Finally, the same environment  $\mathcal{Z}$  is present in the ideal world.  $\mathcal{Z}$  also prescribes the inputs and observes the outputs of all parties. We only consider *static* corruptions, hence the set of corrupt parties is fixed before the start of the computation and is known to  $\mathcal{F}, \mathcal{Z}$  and  $\mathcal{S}$ . We let  $\text{EXEC}_{\mathcal{F}, \mathcal{S}, \mathcal{Z}}(\kappa, z, \mathbf{r})$  denote  $\mathcal{Z}$ 's output on input  $z$  and security parameter  $\kappa$ , after interacting with  $\mathcal{S}$  and dummy parties  $P_1, \dots, P_n$  that interact with  $\mathcal{F}$  using random tape  $\mathbf{r}$ .

*UC emulation.* One of the main concept of the UC framework is that of UC-emulation. Informally, it involves two protocols, say  $\pi$  and  $\phi$ , and we say that  $\pi$  *UC-emulates*  $\phi$  ( $\pi \sim \phi$ ), if for every efficient adversary  $\mathcal{A}$  in an execution of  $\pi$ , there is an efficient ideal adversary  $\mathcal{S}$  in an execution of  $\phi$ , such that no efficient environment  $\mathcal{Z}$  can distinguish an execution of  $\pi$  with  $\mathcal{A}$  and an execution of  $\phi$  with  $\mathcal{S}$ , i.e.  $\text{EXEC}_{\pi, \mathcal{A}, \mathcal{Z}} \approx \text{EXEC}_{\phi, \mathcal{S}, \mathcal{Z}}$ . In particular, it implies that  $\pi$  can be safely used on behalf of  $\phi$  without compromising security.

Since the security of protocols is defined by comparing the “real” protocol execution with an “ideal” one, we can instantiate the concept of protocol emulation with the very special case of  $\phi$  being an ideal protocol  $\text{IDEAL}_{\mathcal{F}}$  for the functionality  $\mathcal{F}$ . Therefore, we say that  $\pi$  UC-emulates  $\mathcal{F}$  if we can infer that  $\pi$  does not leak any other information to an adversary than  $\mathcal{F}$  would have, and hence *securely realizes* the given task no matter how many other instances of  $\pi$  and/or other protocols are executed concurrently. In this case we write  $\text{EXEC}_{\pi, \mathcal{A}, \mathcal{Z}} \approx \text{EXEC}_{\mathcal{F}, \mathcal{S}, \mathcal{Z}}$ .

**Definition 11.** We say that a protocol  $\pi$  *UC-realizes* an ideal functionality  $\mathcal{F}$  if  $\pi$  UC-emulates the ideal protocol  $\text{IDEAL}_{\mathcal{F}}$ .

*Hybrid model.* When a protocol  $\pi$  uses an ideal functionality  $\mathcal{G}$  as a sub-routine, the UC-framework considers the  $\mathcal{G}$ -hybrid model. In this case the parties, in both real and ideal world, have access to a copy of the ideal functionality  $\mathcal{G}$ . In the real world, this is an independent trusted party that executes the functionality honestly. In the ideal world,  $\mathcal{S}$  executes an internal copy of the functionality  $\mathcal{G}$  and only interacts with  $\mathcal{F}$ . An important property of the UC framework is that the ideal functionality  $\mathcal{G}$  in a  $\mathcal{G}$ -hybrid model can be replaced with a protocol  $\rho$  that UC-realizes  $\mathcal{G}$ . More concretely, let  $\rho$  be a protocol that securely realizes  $\mathcal{G}$  and let  $\pi^\rho$  be identical to  $\pi$  with the exception that the interaction with each copy of  $\mathcal{G}$  is replaced with an interaction of a separate instance of  $\rho$ . Then  $\pi$  and  $\pi^\rho$  have essentially the same input/output behaviour. In particular, if  $\pi$  securely realizes  $\mathcal{F}$  in the  $\mathcal{G}$ -hybrid model, then  $\pi^\rho$  securely realizes  $\mathcal{F}$  in the standard model, i.e., without access to  $\mathcal{G}$  and any other functionalities. In the following, we informally state the composition theorem.

**Figure 4:** Oblivious transfer functionality**Figure 5:** Random oracle functionality**Figure 6:** Trusted setup functionality

**Theorem 8** ([Can01]). *Let  $\pi$  be a protocol that UC-securely realizes the ideal functionality  $\mathcal{F}$  in the  $\mathcal{G}$ -hybrid model, let  $\phi$  be a protocol that UC-securely realizes the ideal functionality  $\mathcal{G}$ , then the protocol  $\pi^\phi$ , obtained by replacing each call to the ideal functionality  $\mathcal{G}$  in  $\pi$  with a call to the sub-protocol  $\phi$ , securely realizes  $\mathcal{F}$  in the standard model.*

## A.4 Functionalities

In this work we will make use of relatively standard functionalities. The main OT functionality,  $\mathcal{F}_{\text{OT}}$ , described in Figure 4, is a standard 2-party functionality.

We will also need a random oracle functionality,  $\mathcal{F}_{\text{RO}}$ , as described in Figure 5. It initially contains an empty list  $L$ , then, each time it receives a query  $m$  from a fixed domain  $\mathcal{D}$ , it checks if the queried value  $m$  is already present in the list  $l$ . If this is the case, the functionality outputs the pair  $(m, k)$  in  $L$ ; otherwise, it samples a random  $k \leftarrow \mathcal{R}$ , outputs  $(m, k)$ , and stores that pair in  $L$ .

Finally, we will make use of a trusted setup  $\mathcal{F}_{\text{TSC}}$ , described in Figure 6, that fixes a starting curve  $E \in \mathcal{E}$ . Note that it outputs  $E = t \star E_0$ , but not  $t$  which maps  $E_0$  to  $E$ .

## B The Algebraic Adversary Model

Here, we briefly recall the algebraic group model by Fuchsbauer, Kiltz and Loss [FKL18], and in particular its instantiation in the UC framework, resulting in the UC-AGM as described by Abdalla et al. [ABK<sup>+</sup>21]. We also show how the AGM enables a proof of the “simplest OT” by Chou and Orlandi [CO15].

**The UC-AGM Framework.** The setting involves a group  $\mathbb{G}$  of prime order  $p$ , with known generator  $g$ . We collect those parameters in  $\mathcal{G} = (\mathbb{G}, g, p)$ . Roughly, an *algebraic adversary*, compared to a standard adversary, has an additional auxiliary tape on which it writes the representation of any group element it outputs on some other tapes. More formally, we have the following definition.

**Definition 12.** Suppose a protocol  $\pi$  uses the group  $\mathcal{G}$  as above. A pair of environment  $\mathcal{Z}$  and adversary  $\mathcal{A}$  is said  $(\mathcal{G}, \pi)$ -algebraic if it satisfies the following conditions.

1.  $\mathcal{A}$  has a special output tape called **algebraic tape**;
2. Whenever  $\mathcal{A}$  sends a **(backdoor,  $m$ )** message to a party and  $m$  contains an element  $h \in \mathbb{G}$ , then either
  - (a) It must provide (to a special “algebraic tape”) an algebraic representation  $X$  of  $h$ , or
  - (b)  $\mathcal{A}$  has previously received such algebraic representation from  $\mathcal{Z}$ ,

where the algebraic representation of  $h$  is a list  $X = [(g_1, x_1), \dots, (g_k, x_k)]$  such that  $h = \prod_{i=1}^k g_i^{x_i}$  and  $g_i$  are group elements already seen by  $\mathcal{A}$  or  $\mathcal{Z}$  in the execution of  $\pi$ .

With this definition, it is possible to restrict standard UC-emulation to algebraic adversaries and environments.

**Definition 13.** Suppose protocols  $\pi$  and  $\phi$  involve the same group  $\mathcal{G}$ . We say that  $\pi$   $\mathcal{G}$ -AGM emulates  $\phi$  if for any efficient adversary  $\mathcal{A}$  there is an efficient simulator  $\mathcal{S}$  such that for any efficient environment  $\mathcal{Z}$  with  $(\mathcal{Z}, \mathcal{A})$  that is  $(\mathcal{G}, \pi)$ -algebraic we have that also  $(\mathcal{Z}, \mathcal{S})$  is  $(\mathcal{G}, \phi)$ -algebraic and

$$\text{EXEC}_{\phi, \mathcal{S}, \mathcal{Z}} \approx \text{EXEC}_{\pi, \mathcal{A}, \mathcal{Z}}.$$

We can apply the definition of AGM-emulations (Definition 13) to an ideal protocol  $\text{IDEAL}_{\mathcal{F}}$  and instantiate Definition 11 accordingly.

Like in standard UC, we can use dummy adversaries also in this algebraic setting; we only need to pay attention to the algebraic representations that the environment send to the adversary, because we don't want to forward them to the actual protocol.

**Definition 14.** Suppose the protocol  $\pi$  involves  $\mathcal{G}$ . An adversary  $\mathcal{D}$  is  $(\mathcal{G}, \pi)$ -algebraically dummy if it only forwards messages in this way:

- For any received message of the type **(backdoor,  $m$ )** from a party ID, it sends **(backdoor, (ID,  $m$ ))** to  $\mathcal{Z}$ .
- For any **(input, (ID,  $m$ ))** from  $\mathcal{Z}$ , it sends **(input,  $m'$ )** to ID, where  $m'$  is equal to  $m$ , but without all algebraic representations  $X$  of elements  $h \in \mathbb{G}$  which are inside  $m$ .

Using this definition of dummy adversary we then have the following theorem.

**Theorem 9.** *Suppose protocols  $\pi$  and  $\phi$  involve group  $\mathcal{G}$ . Then  $\pi$   $\mathcal{G}$ -AGM emulates  $\phi$  if and only if  $\pi$   $\mathcal{G}$ -AGM emulates  $\phi$  with respect to the dummy adversary.*

Observe also that since the dummy adversary doesn't output any algebraic representation, they must all come from the environment  $\mathcal{Z}$ .

**Composition and transitivity.** It is possible to prove the composition theorem in the UC-AGM framework stated as follows.

**Theorem 10** ([ABK<sup>+</sup>21]). *Let  $\pi$  and  $\phi$  protocols involving group  $\mathcal{G}$ , such that  $\phi$  is a sub-protocol of  $\rho^\phi$ , and  $\pi$   $\mathcal{G}$ -AGM emulates  $\phi$ . Then  $\rho^\pi$  ( $\mathcal{G}, \pi, \phi$ )-AGM emulates  $\rho^\phi$ .*

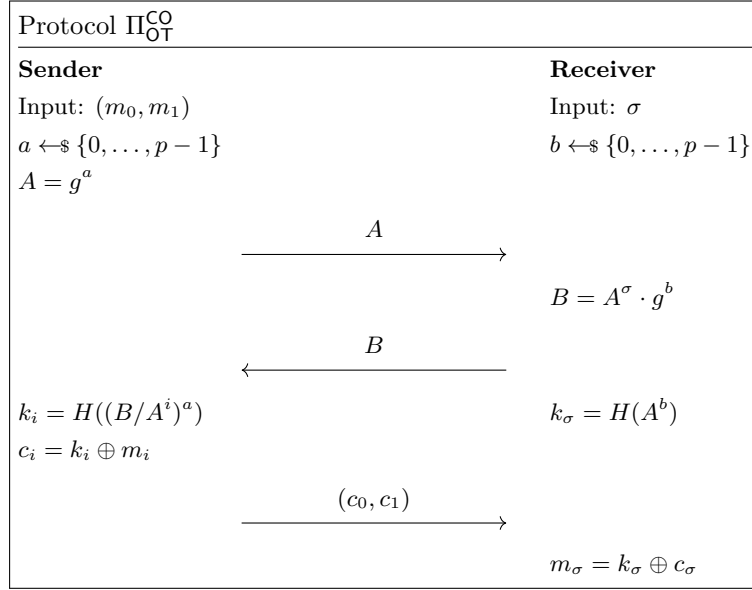
Similarly to standard UC, we have that if  $\pi, \phi'$  and  $\phi$  are protocols involving  $\mathcal{G}$ , and  $\pi$   $\mathcal{G}$ -AGM emulates  $\pi'$  (i.e.  $\pi \underset{A}{\sim} \pi'$ ) and  $\pi' \underset{A}{\sim} \phi$ , then  $\pi \underset{A}{\sim} \phi$ . However, we need to take special care when we combine the previous transitivity result with composition.

**Theorem 11** ([ABK<sup>+</sup>21]). *Suppose protocols  $\rho^{\mathcal{F}}, \pi$  and ideal functionalities  $\mathcal{F}, \mathcal{F}'$  involve the same group  $\mathcal{G}$ , such that:*

- $IDEAL_{\mathcal{F}}$  is a sub-protocol of  $\rho^{\mathcal{F}}$ ,
- $\pi$  ( $\mathcal{G}, \pi$ )-AGM realizes  $\mathcal{F}$ ,
- $\rho^{\mathcal{F}}$  ( $\mathcal{G}, \rho$ )-AGM realizes  $\mathcal{F}'$ ,

*Then, the protocol  $\rho^\pi$  AGM realizes  $\mathcal{F}'$  with respect to attackers that are both ( $\mathcal{G}, \rho$ )- and ( $\mathcal{G}, \pi$ )-algebraic.*

## B.1 The CO-OT Protocol in AGM

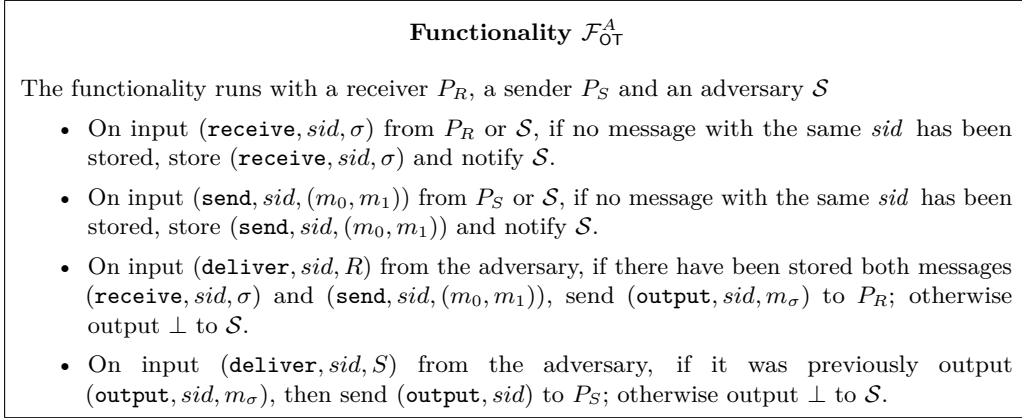


**Figure 7:** The Simplest OT protocol by Chou and Orlandi

Here, we will analyse the OT protocol proposed by Chou and Orlandi [CO15]. The core of the protocol is described in Figure 7, and needs a random oracle functionality, denoted here by  $H$ .

### B.1.1 Proof of UC-AGM security of Chou and Orlandi

The Algebraic Group Model is a key tool for proving UC security for the “simplest OT” protocol. Roughly, it uses the algebraic behaviour of the adversary both for explaining the parties’ state after adaptive corruptions and for extracting the input bit of a corrupt receiver. We will only give a sketch of the proof in the case of static corruptions. We use the protocol in Figure 7, with the functionality presented in Figure 8.



**Figure 8:** OT functionality in the AGM-UC framework

**Theorem 12.** *The protocol  $\Pi_{\text{OT}}^{\text{CO}}$  AGM-realizes the functionality  $\mathcal{F}_{\text{OT}}^A$  in the  $\mathcal{F}_{\text{RO}}$ -hybrid model under static corruptions.*

*Proof.* We construct a simulator  $\mathcal{S}$  for the dummy algebraic adversary in each of the four corruption cases. By definition, this means that all group elements output by  $\mathcal{Z}$  to  $\mathcal{S}$  must have a representation.

**CORRUPTED SENDER AND HONEST RECEIVER:** When  $\mathcal{S}$  receives  $A$  from the adversary, it also learns the value  $a$  such that  $A = g^a$ . The simulator then chooses a random  $b$ , computes  $B = g^b$ , and sends it back to  $\mathcal{A}$ .

Then it computes the key  $k = H(A^b)$ . When the adversary sends any  $(c_0, c_1)$ , the simulator sends  $(c_0 \oplus k, c_1 \oplus k)$  to the trusted party and makes it deliver to the honest receiver.

This simulates correctly since the output of the receiver is identical in both the ideal and the real execution; moreover the distributions  $g^{a\sigma+b}$  and  $g^b$  are identical, so the environment cannot distinguish between the case  $\sigma = 0$  and  $\sigma = 1$ .

**HONEST SENDER AND CORRUPTED RECEIVER:** The simulator samples  $a$ , computes  $A = g^a$  and sends it to the adversary, which responds with an arbitrary element  $B$ ; since  $\mathcal{A}$  is algebraic, it must also output a representation  $B = A^x g^y$ .

If  $x \in \{0, 1\}$ , the simulator queries the functionality with this bit, and sets  $m_x$  to the retrieved value; in all other cases (i.e.  $x = 1 - x$ , or both 0 and 1 if  $x \notin \{0, 1\}$ ), it sets  $m_i$  to null. It also samples random  $c_0, c_1$ .

The simulator also runs the random oracle, and checks the queries made to it. In particular, upon learning  $B$ , it retroactively checks all queries for the values  $B^a g^{-ia^2}$ : if  $m_i$  is null the simulator aborts, otherwise it sets  $c_i = k_i \oplus m_i$ , where  $k_i$  was the answer of the query; it also does this for future queries, this time by computing the answer as  $k_i = c_i \oplus m_i$ . This means that  $\mathcal{S}$  aborts precisely when  $x \in \{0, 1\}$  and  $\mathcal{A}$  queries for both  $B^a$  and  $B^a g^{-a^2}$ , or if  $x \notin \{0, 1\}$  and  $\mathcal{A}$  queries at least one of  $B^a$  or  $B^a g^{-a^2}$ .

The simulator concludes the simulation by sending  $(c_0, c_1)$  to the adversary.

Notice that when  $\mathcal{S}$  does not abort, the simulation is perfect. Thus, the proof follows from this claim:

**Claim.**  *$\mathcal{S}$  aborts with negligible probability if the discrete logarithm is hard.*

*Proof.* Suppose we want to solve the discrete logarithm problem  $A = g^a$ , using  $\mathcal{A}$  as an oracle. We feed  $\mathcal{A}$  the element  $A$  as coming from the simulator. Notice that now the simulator cannot check what is the query that makes it abort, so the solver for the discrete

logarithm problem analyzes all queries made by  $\mathcal{A}$ , and for each of them tries to solve the equation in  $z$  and checks if  $g^z = A$ , thus finding the secret exponent.

- Case  $x \notin \{0, 1\}$ . Suppose  $\mathcal{A}$  queries  $B^z$ . Being algebraic, it must know a representation  $B^z = A^s g^t$ . But this means that  $g^{z^2 x + zy} = (A^x g^y)^z = B^z = g^{sz+t}$ , i.e.

$$z^2 x + z(y - s) - t \equiv 0 \pmod{p}$$

from which we can compute  $z$ . In the other case we have that  $B^z g^{-z^2} = A^s g^t$ , for which the equation is  $z^2(x - 1) + zy \equiv sz + t \pmod{p}$ , which also has a solution.

- Case  $x = 0$ . The adversary  $\mathcal{A}$  has queried  $B^z g^{-z^2}$ , for which it knows a representation  $A^s g^t$ . Then it gets the equation  $zy - z^2 \equiv sz + t \pmod{p}$ , which has a solution.
- Case  $x = 1$ . The adversary  $\mathcal{A}$  has queried  $B^z$ , and represents it as  $A^s g^t$ . Then the equation is  $z^2 + zy \equiv sz + t \pmod{p}$ , which also has a solution.

This concludes the proof of the claim.  $\square$

**HONEST SENDER AND HONEST RECEIVER:** This simulation can be constructed putting together both simulations above, as we did in the proof of the toy protocol.

This concludes the proof since if the simulator doesn't abort, the simulation is perfect, given that the keys queried from the random oracle statistically hide the messages. Finally we observe that all the simulators we have constructed are algebraic themselves.  $\square$