

Simple Two-Message OT in the Explicit Isogeny Model

E. Orsini, R. Zanotto

ArticCrypt | 07/07/2025



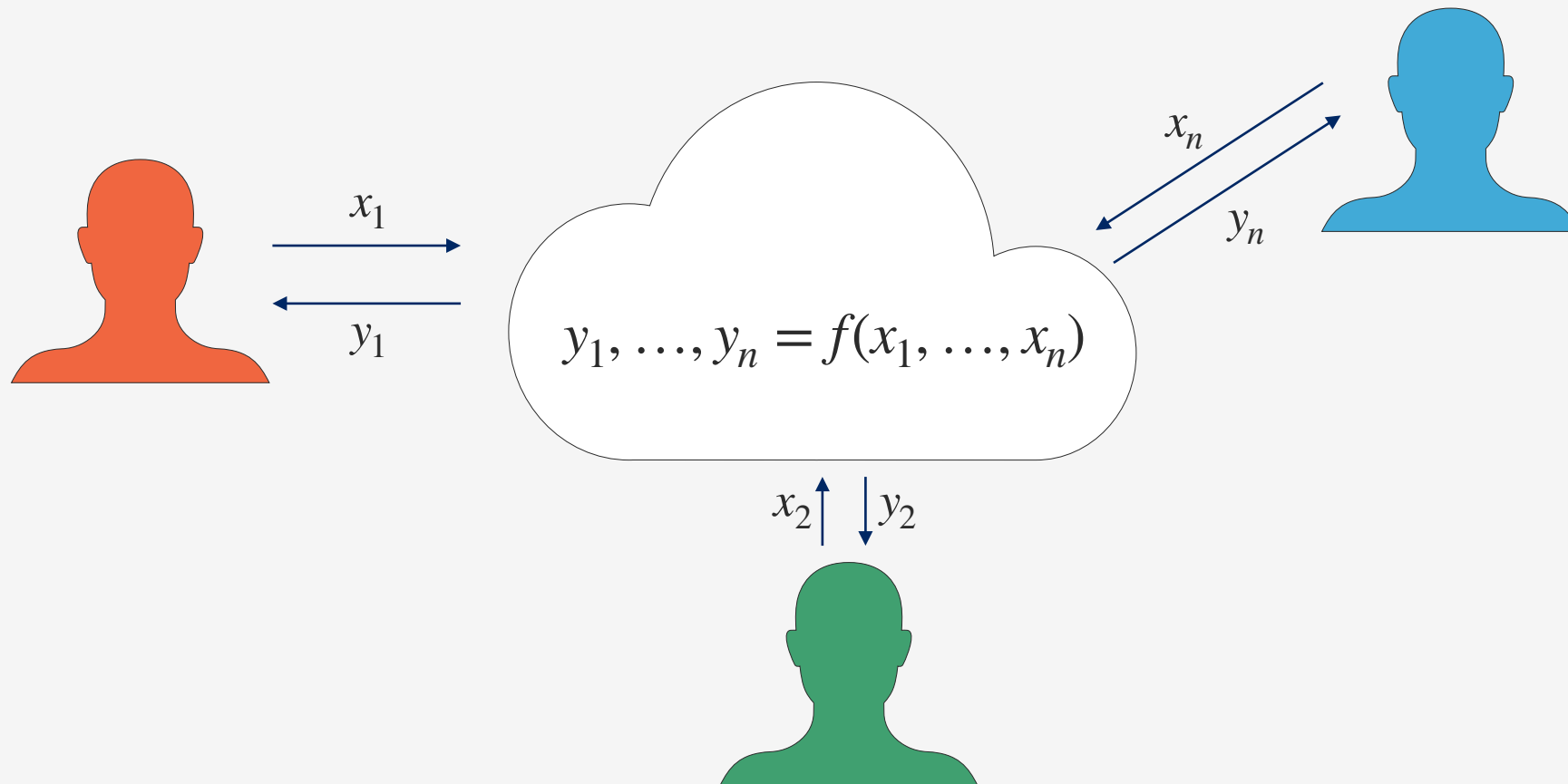
Contents

- I. Introduction & motivation
- II. Background
 - Oblivious transfer
 - Group actions
 - Known OT protocols from GA
- III. Our contributions
 - Algebraic group action model
 - Explicit isogeny model
 - Malicious two-message OT from isogenies
- IV. Conclusions



Introduction

- Goal of modern cryptography: computing on private data (**MPC**, SFE)
- Tools to achieve this: FHE, ZK, **OT**





Motivation

- OT is **sufficient** for 2PC, via garbled circuits
- Different **metrics** to optimise: round complexity, rate
- Optimal is 2-round, rate-1. Very hard to achieve
- Different security assumptions: we know (generic) constructions of 2-round OT from CDH, LPN, LWE. What about isogenies?
- *Can we get fully malicious MPC from isogenies - **efficiently**?*



Background

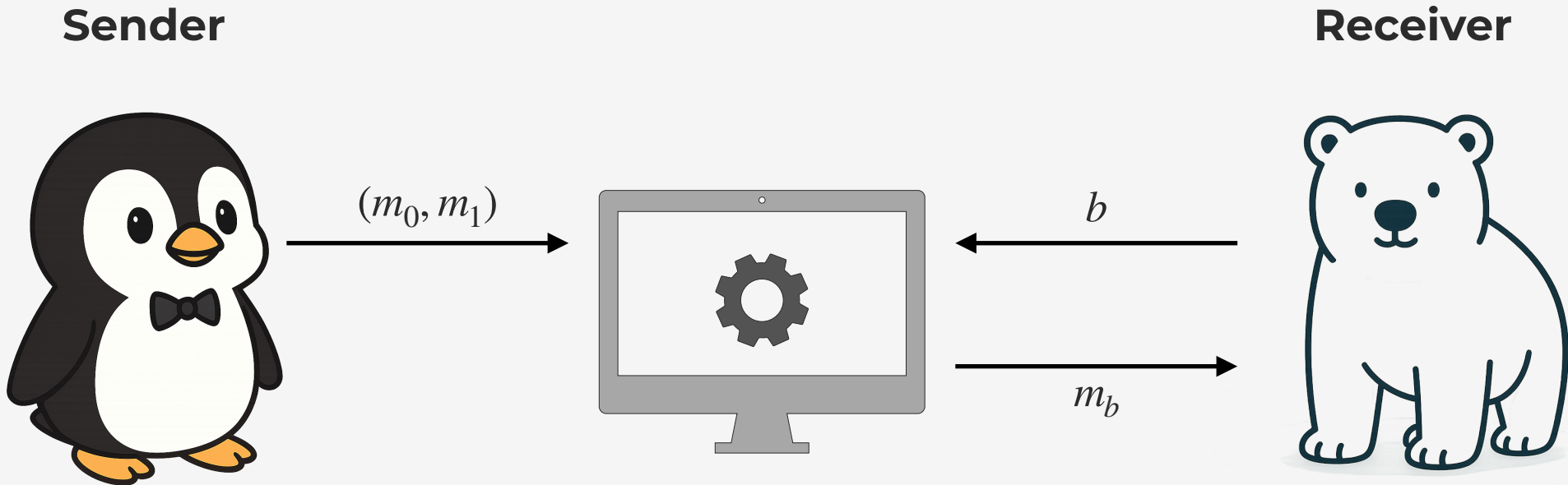
- I. Oblivious transfer
- II. Group actions
- III. Known OT protocols
from group actions



What is oblivious transfer?

- Oblivious transfer (OT) is a 2-party functionality defined by

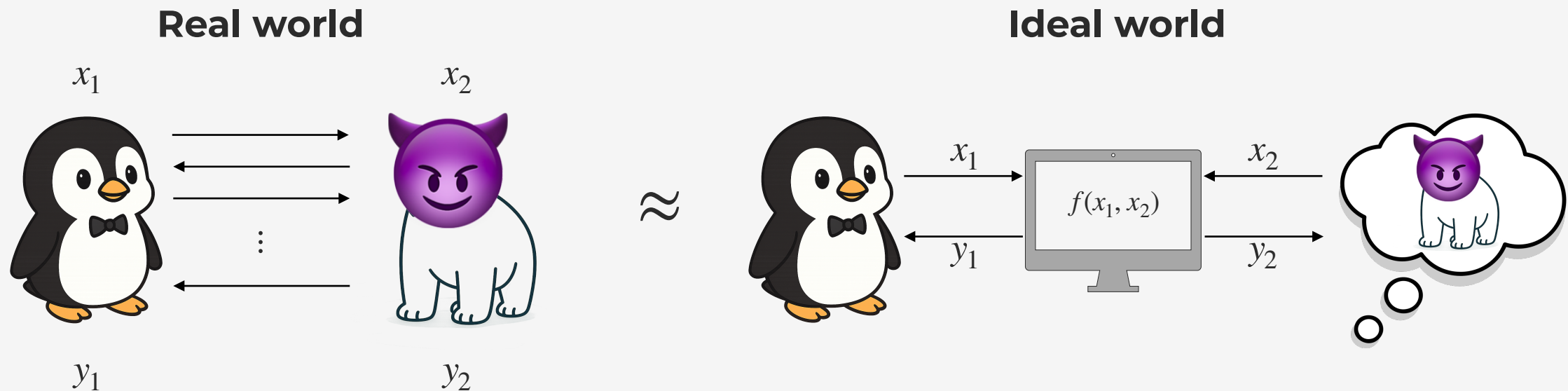
$$f((m_0, m_1), b) = (\emptyset, m_b)$$





Security models for MPC protocols

- Simulation based: executing the protocol should reveal “**no more information**” than just party’s own input/output
- Adversarial models:
 - Semi-honest: follows the protocol, just tries to learn more information
 - Malicious: arbitrary deviations from the protocol





How to achieve fully malicious OT

- Achieving 2-message UC-secure OT in plain model is **impossible** (simulator is also real world adversary)
- Main problem in proofs with malicious adversaries: **extracting** their input
- Usual approach: add CRS and/or RO
 - NIZK proofs
 - Dual-mode encryption [PVW08]
 - *Elementary OT* compiler [DGH+20]
- Alternative approach: restrict adversary's power (e.g. algebraic group model, under which [CO15] is UC secure [ABK+21])



Background

- I. Oblivious transfer
- II. Group actions
- III. Known OT protocols
from group actions



Group actions

- Post-quantum generalisation of group exponentiation
- Defined by a tuple $(G, X, \star)$
 - G is a group
 - X is a set
 - $\star : G \times X \rightarrow X$ is the map defining the action; we denote it by $g \star x$
- We need that $g \star (h \star x) = (gh) \star x$
- For example, $\mathbb{Z}_p$ acts on cyclic groups via $n \star t = t^n$

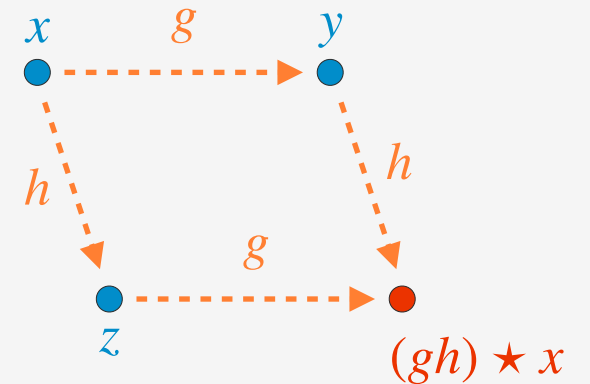
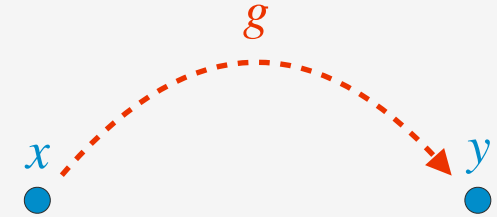


(Picture from Wikipedia)



Cryptographic properties of group actions

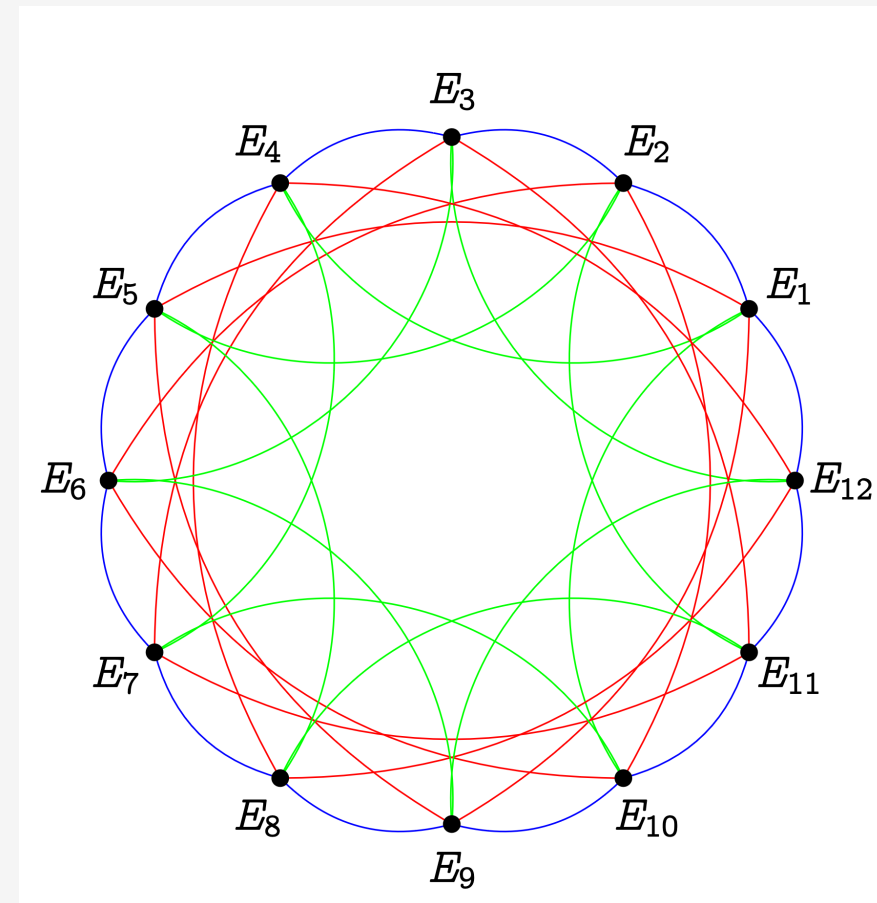
- In order to do cryptography, we need to have a complexity gap [AFMP20]
- Easy problems:
 - Computing group operations
 - Computing the action $g \star x$ for any g, x
- Hard problems:
 - **Vectorization**: given $x, y = g \star x$ compute g
 - **Parallelization**: given $x, y = g \star x, z = h \star x$ compute $(gh) \star x$
- We are **hiding** group elements into set elements





Group action from isogenies: CSIDH

- Given $\mathcal{O} = \mathbb{Z}[\sqrt{-p}]$, there is a regular action of $Cl(\mathcal{O})$ on the set of supersingular elliptic curves over $\mathbb{F}_p$ (curves with $\#E(\mathbb{F}_p) = p + 1$)
- We can abstract and use it as an action of $\mathbb{Z}^d$ on supersingular curves
 - Given $v = (v_1, \dots, v_d) \in \mathbb{Z}^d$ and a curve E , we get another curve $v \star E$
 - Composition is $(v + w) \star E = v \star (w \star E)$
- In picture, $d = 3$, the edges are actions of basis vectors $(1,0,0)$ (blue), $(0,1,0)$ (red), $(0,0,1)$ (green)



(Picture by Luca De Feo)



The CSIDH action with twists

- The **twist** of an elliptic curve $E/\mathbb{F}_p$ is a curve $E^t/\mathbb{F}_p$ such that E, E^t have the same j -invariant, but are not isomorphic over $\mathbb{F}_p$
- In the CSIDH setting, the twist of $y^2 = x^3 + Ax^2 + x$ is $y^2 = x^3 - Ax^2 + x$
- Twisting is an *involution*, since $(E^t)^t = E$
- Compatible with the inversion automorphism of $Cl(\mathcal{O})$:

$$(\mathfrak{a} \star E)^t = \mathfrak{a}^{-1} \star E^t$$



Background

- I. Oblivious transfer
- II. Group actions
- III. Known OT protocols
from group actions



What can we achieve?

Reference	# M	# GAs	Adversary
[LGd21] I	2	5	Semi-honest
[DOPS20] I	2	5	Semi-honest
[ADMP20] + [PVW08]	2	$O(\lambda)$	Malicious
[DOPS20] + [DGH+20]	2	$O(\lambda)$	Malicious
[BMM+23]	2	$O(\lambda)$	Malicious
[LGd21] II	4	11	Malicious
[LGd21] + this work	2	5	Malicious



2-message OT protocol [LGd21]

- Lai, Galbraith and de Saint Guilhem proposed a 2-message OT protocol
- They proved UC-security against **semi-honest** adversaries
- It uses the *twisting* operation on elliptic curves
- Communication is 2 curves, 2 ciphertexts
- Computation is 5 CSIDH operations

Setup: H is a RO; E is output of CRS

Sender (m_0, m_1)

$$s \leftarrow G$$

$$A = s \star E$$

$$k_0 = H(s \star C_b)$$

$$k_1 = H(s \star C_b^t)$$

$$c_i = \text{Enc}_{k_i}(m_i)$$

$\xrightarrow{A, c_0, c_1}$

Receiver (b)

$$r \leftarrow G$$

$$C_0 = r \star E$$

$$C_1 = C_0^t$$

C_b

$\longleftarrow$

$$k_b = H(r \star A)$$

$$m_b = \text{Dec}_{k_b}(c_b)$$



Our contributions

- I. Algebraic group action model
- II. Explicit isogeny model
- III. Malicious two-message OT from isogenies



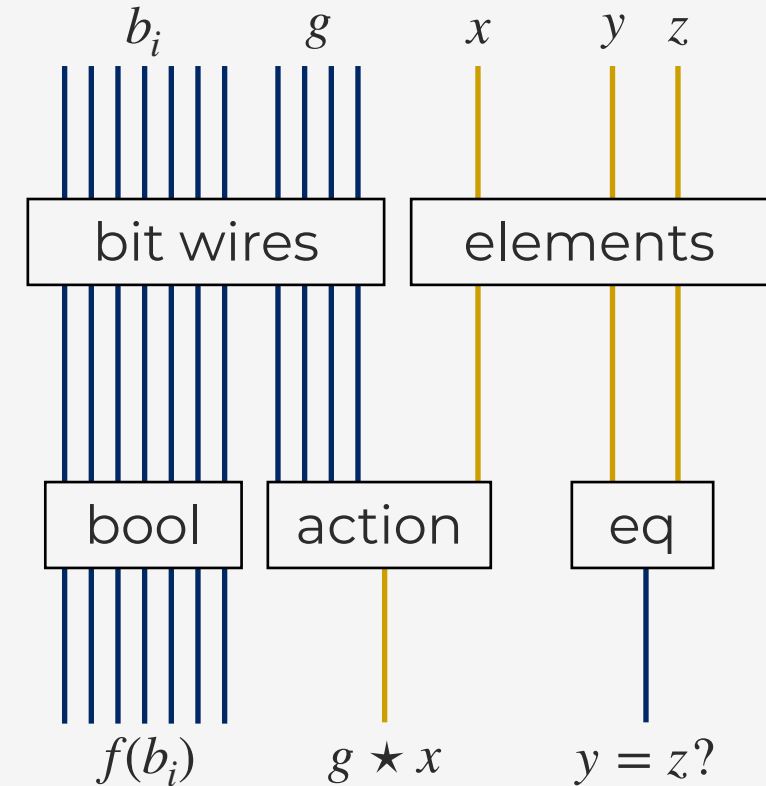
Algebraic models

- Algebraic models **restrict** the class of the adversaries, by imposing additional behaviour
- It's like **knowledge assumptions** baked in the model of computation
- Algebraic Group Model: for classical groups. Every time adversary outputs an $h \in G$, they must **explain** it as $h = g_1^{e_1} \cdots g_k^{e_k}$, where the g_i are previously seen elements
- Slightly informal in the definition: what does it mean to “**output** a group element”?
- Algebraic Action Model: for the **group action** setting. When the adversary outputs $y = g \star x$, it must provide g and a previously seen element x



The Type-Safe model for group actions

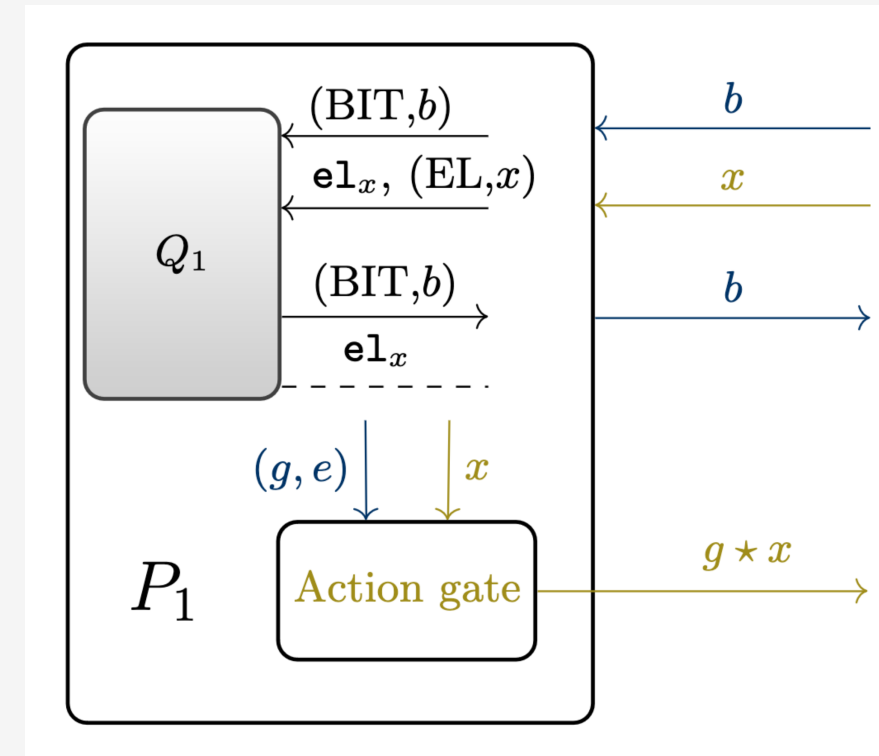
- We introduce a Type-Safe model [Zha22] also for group actions
- Fix a group action $(G, X, \star)$
- Each party in the protocol is a circuit with **two types of wires**:
 - Bit wires
 - Element wires, for elements $x \in X$
- The wires interact via **gates**:
 - Boolean gates
 - Action gate
 - Equality gate





The Algebraic Action Model

- An algorithm is *algebraic* w.r.t. an action $(G, X, \star)$ if for any element $y \in X$ that they output, they provide an **explanation** $y = g \star x$, where x is a previously seen set element
- We can formally define the meaning of “output a set element” via the wire distinction of the type-safe model
- We can **compile** type-safe protocols with algebraic algorithms and **define** security against algebraic adversaries





Our contributions

- I. Algebraic group action model
- II. Explicit isogeny model
- III. Malicious two-message OT from isogenies



The CSIDH action with twists, revisited

- We model the CSIDH group action with twists using the group $\mathcal{G}_{tw} := Cl(\mathcal{O}) \rtimes \mathbb{Z}/2\mathbb{Z}$, again acting on the set of supersingular curves
- The action of $(\mathfrak{a}, b)$ on a curve E is $\mathfrak{a} \star E$ or $\mathfrak{a} \star E^t$ depending on $b=0$ or 1
- Notice that acting with $(\mathfrak{a}', 1)$ on $(\mathfrak{a}, 0) \tilde{\star} E = \mathfrak{a} \star E$ results in

$$\mathfrak{a}' \star (\mathfrak{a} \star E)^t = \mathfrak{a}' \mathfrak{a}^{-1} \star E^t$$

which is also the result of acting with $(\mathfrak{a}' \mathfrak{a}^{-1}, 1)$ on E



The Explicit Isogeny Model

- **Definition:** an algorithm $\mathcal{A}$ uses *Explicit Isogenies* if whenever it outputs a curve $E/\mathbb{F}_p$ it also outputs an explanation $\mathfrak{a} \in Cl(\mathcal{O})$ such that $E = \mathfrak{a} \star E'$ where E' is any other curve on the input tape of $\mathcal{A}$, or its twist
- **Observation:** for this definition to be well posed, the algorithm $\mathcal{A}$ must interact with a *type-safe protocol* for the group action $\mathcal{G}_{tw}$
- In particular, the **El model** is the Algebraic Action Model instantiated with the semidirect product $\mathcal{G}_{tw}$



The sampling problem

- Why is the EI model meaningful?
 - adversary is **only** using the group action to create new curves
 - it is unable to **sample** new curves without knowing a path to them
- The oblivious supersingular sampling problem has been studied, no efficient solutions are known
- All known methods to generate new supersingular curves **leak** a path to the starting one
- It is tempting to pose the assumption that adversaries are algebraic; would be a non-falsifiable one.



Our contributions

- I. Algebraic group action model
- II. Explicit isogeny model
- III. Malicious two-message OT from isogenies



2-message OT protocol [LGd21]

- Recall the [LGd21] 2-message OT protocol
- It is **type-safe** wrt $\mathcal{G}_{tw}$
- We have the functionalities
 - RO, *random oracle*
 - TSC, *trusted setup curve*, which generates a random supersingular curve E for the protocol

Setup: H is a RO; E is output of TSC

Sender (m_0, m_1)

Receiver (b)

$$s \leftarrow G$$

$$r \leftarrow G$$

$$A = s \star E$$

$$C_0 = r \star E$$

$$C_1 = C_0^t$$

C_b



$$k_0 = H(s \star C_b)$$

$$k_1 = H(s \star C_b^t)$$

$$c_i = \text{Enc}_{k_i}(m_i)$$

A, c_0, c_1



$$k_b = H(r \star A)$$

$$m_b = \text{Dec}_{k_b}(c_b)$$



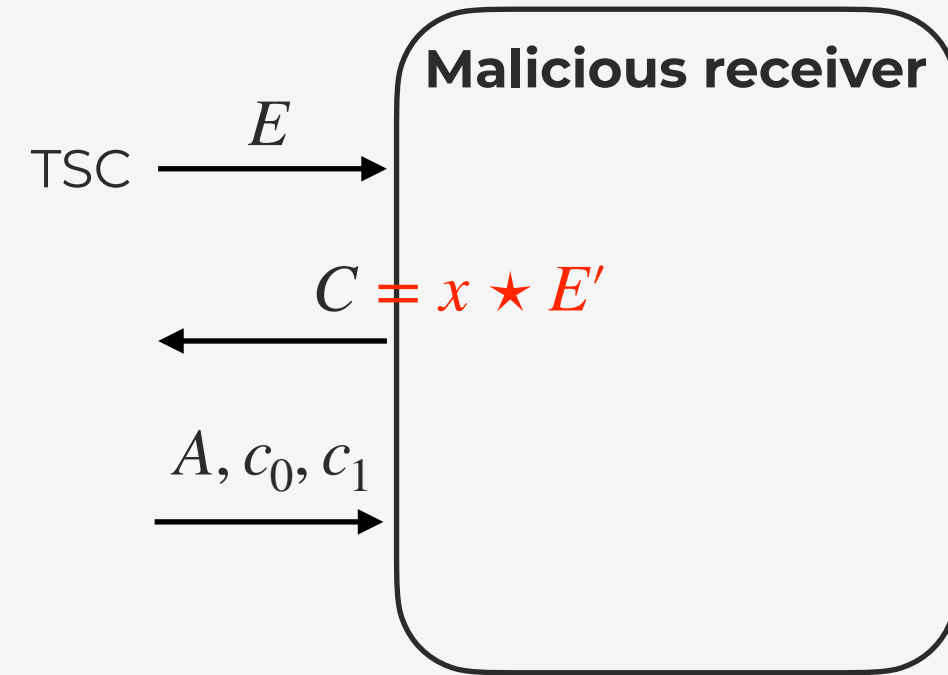
Main result

- **Theorem:** The protocol Π_{tw} UC-EI realises the OT functionality in the presence of algebraic adversaries in the (RO, TSC)-hybrid model if the encryption scheme is IND-CPA secure and the CSIDH vectorization problem is hard.
- Main tools:
 - Algebraic behaviour allows for **extraction** of malicious receiver's choice bit
 - Algebraic explanations allow to **solve** vectorisation problems
 - Twisting is statistically hidden



Sketch of the proof

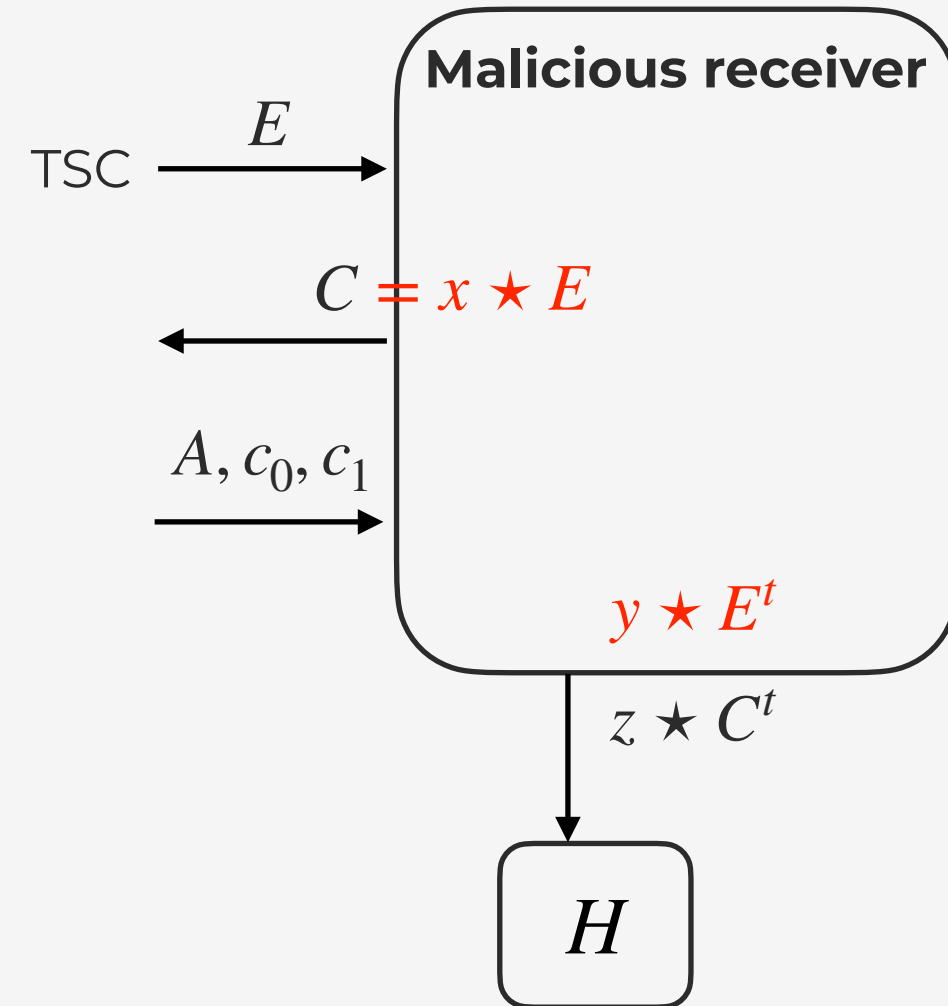
- We build a reduction to a vectorisation problem $E_1 = z \star E_0$
- Simulator computes TSC as $E = t \star E_0$ and **embeds** the challenge into the public key $A = t \star E_1 = z \star E$
- Simulator **extracts** receiver's choice bit:
 - $E' = E$, then $b=0$
 - $E' = E^t$, then $b=1$
- Suppose $C = x \star E$, so $b=0$; and suppose receiver successfully cheats learning m_1





Sketch of the proof

- In order to know m_1 , the receiver **must** query the random oracle on $Q = z \star C^t$
- But since it's algebraic, it **must** explain it as being obtained from E or A or their twists
- From any explanation we can **compute** the secret value z
 - $Q = y \star E_0 \rightarrow z = xyt$
 - $Q = y \star E \rightarrow z = xyt^2$
 - $Q = y \star E^t \rightarrow z = xy$





Conclusions



Results

- We introduce a Type-Safe model for group actions
- We formalize an **Algebraic Action Model**
- We describe the **CSIDH with twists** as a bigger group action $\mathcal{G}_{tw}$
- We instantiate the AAM using $\mathcal{G}_{tw}$, obtaining the EI model
- We prove the EI security of an **efficient two-message OT** protocol against malicious adversaries



Concurrent works

- *Generic Models for Group Actions* [DHKKLR23] also introduces generic and algebraic models
 - Observes that the generic model in the **quantum** setting is **void**: poly many queries are enough to break vectorisation, with exponential classical work
 - They prove **equivalence** of various problems with vectorisation in the quantum algebraic model
 - Twists are modelled **externally**



Thank you!

- Contact: riccardo.zanotto@cispa.de, emmanuela.orsini@unibocconi.it
- Paper: <https://cic.iacr.org/p/1/1/15>



Questions?



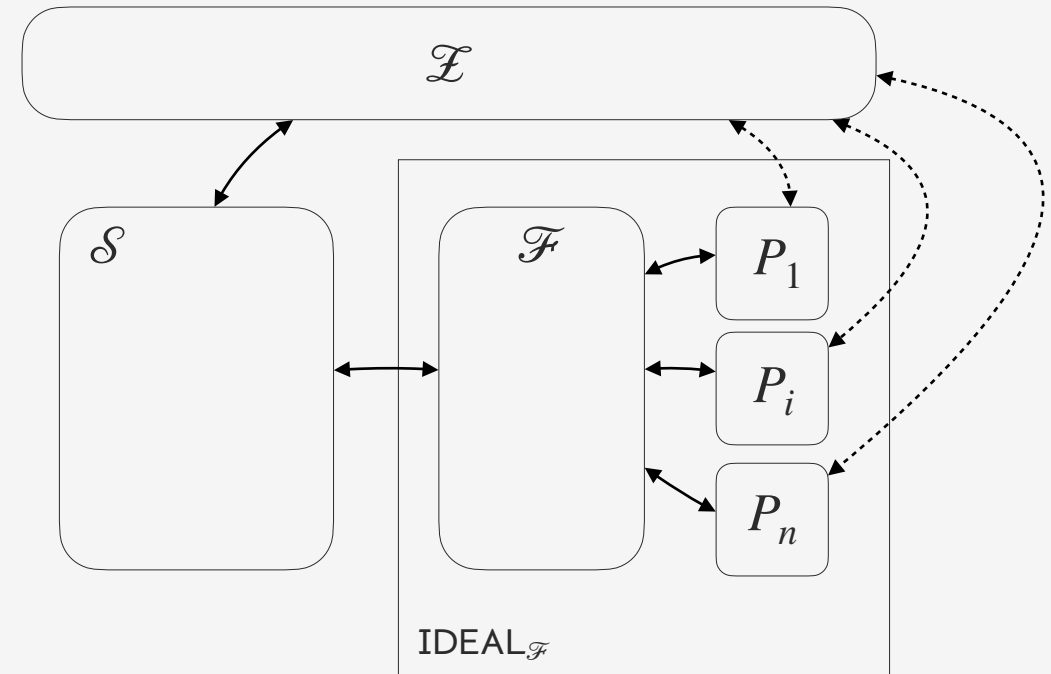
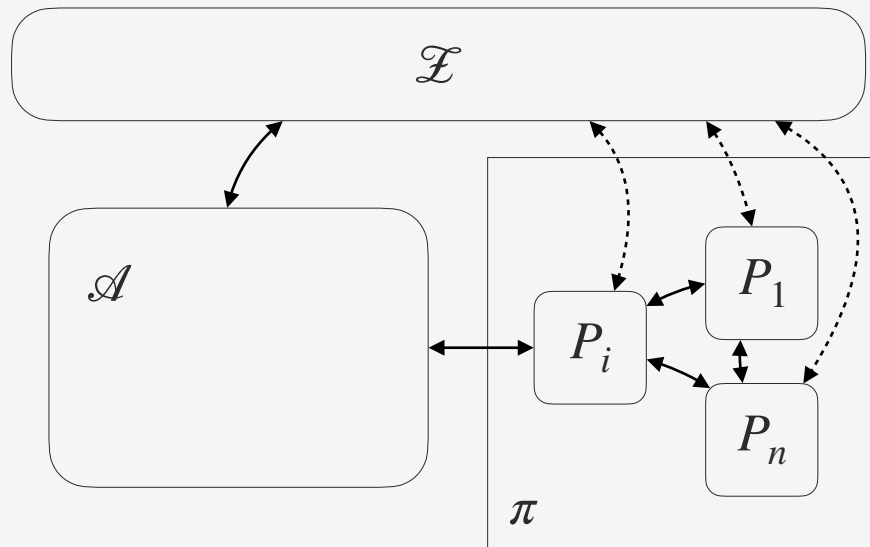






The UC framework

- In the Universal Composability framework, the distinguisher -called **environment**- between the real and ideal world is **interactive**
- Tries to capture the notion of security even with parallel executions of the protocol, or its interaction with any other cryptographic protocol





The UC-AAM

- **Definition:** given a type-safe functionality $\mathcal{F}$ and a type-safe protocol π we say that π *UC-AA emulates* $\mathcal{F}$ if for any algebraic adversary $\mathcal{A}$ there is an algebraic simulator $\mathcal{S}$ such that for any algebraic environment $\mathcal{I}$ it holds

$$\text{REAL}_{\pi, \mathcal{A}, \mathcal{I}} \approx \text{IDEAL}_{\mathcal{F}, \mathcal{S}, \mathcal{I}}$$

- **Theorem** (informal): we can *compose* protocols, if we restrict to *non-mixing* adversaries



Limitations

- The Algebraic Group Action Model is still a model, so it might be **less expressive**; though there is evidence of it not being very restrictive
- The **composition** theorem for UC type-safe/algebraic protocols is **unsatisfactory**, since it requires the adversary to explain set elements of the sub-protocol by staying inside the sub-protocol



Open questions

- Can we use the Explicit Isogeny model to prove better security of other CSIDH-based protocols?
- Can we improve the composition theorem for algebraic adversaries (issue in both UC-AGM and UC-AAM)?