CROOT: Code-based Round-Optimal Oblivious Transfer*

Nicolas Aragon, Olivier Blazy, Neals Fournaise and Philippe Gaborit
Université de Limoges, XLIM-DMI, 123, Av. Albert Thomas, 87060 Limoges Cedex, France

Keywords: Oblivious Transfer, Code-based Crypto, Random-Oracle, Rank Metric.

Abstract: In this paper, we present a new functionality for 1-out-of-2 Oblivious Transfer. This functionality lives between the classical and the weak-Oblivious Transfer ones. We motivate this functionality to adapt and prove a formerly proposed (and retracted) framework that was shown to be unprovable with the classical OT functionality. Our functionality still remains reasonably close to natural expectation for Oblivious Transfer. Once our global framework is proven in the new functionality, we propose two instantiations using recent code-based candidates of the NIST post-quantum standardization process. We show that the resulting instantiations are both efficient and secure. Our new functionality opens the way to round-optimal oblivious transfer instantiations.

1 INTRODUCTION

The concept of Oblivious Transfer (OT) was introduced by Rabin in 1981 (Rabin, 1981). In the simplest form of the protocol, the 1-out-of-2 OT, a sender possesses two messages m_0, m_1 and a receiver chooses a bit b. After a successful execution of the protocol, the receiver obtains message m_b while learning nothing about m_{1-b} and the sender does not know which of the two messages has been requested. Such a scheme has been shown to be *complete* (Kilian, 1988; Ishai et al., 2008) in the sense that secure Multi-Party Computation (MPC) can be obtained directly from it.

The work of (Canetti et al., 2002) motivated the use of the Universal-Composability (UC) framework of (Canetti, 2001) when constructing OT schemes. Indeed, proving a scheme secure in the UC model guarantees that composing it in a larger protocol will preserve security. As OT is regarded as a building block for MPC and as such is ran under composition, it is of importance to have it in the UC model. Another important notion that divides many works in the area of OT is the adversary corruption model which can be *static* or *adaptive*. In the former model, corrupted parties are decided at the start of the protocol while adaptive security is a stronger model in which the adversary can corrupt honest parties at any time during the protocol. It captures better real-world scenarios where corruption can happen at anytime e.g. if an adversary acquires the control of a computer during execution of the protocol.

Many OT instantiations have been proposed to reach greater efficiency in terms of round, computation or communication costs (Naor and Pinkas, 2001; Naor and Pinkas, 2005; Halevi and Kalai, 2012; Chou and Orlandi, 2015). In (Horvitz and Katz, 2007), the authors achieved the first round-optimal OT while (Peikert et al., 2008) propose a general framework for round-optimal UC-secure OT, both are in the static setting. In the adaptive security model, (Blazy and Chevalier, 2016; Blazy et al., 2017; Choi et al., 2013; Garay et al., 2009; Barreto et al., 2017) proposals lack efficiency and round-optimality.

In a recent work from (Byali et al., 2017), the authors present the first round-optimal adaptively UC-secure construction of OT relying on the Decisional Diffie-Hellman (DDH) problem. Their approach is closely related to that of (Peikert et al., 2008) but it does not seem to be instantiable using post-quantum hardness assumption. Our work follows up on this in the sense that our framework can be instantiated, as we show, from Rank Metric Codes-based cryptographic schemes which are believed to be quantum-computer resistant .

1.1 Related Work

Since the original paper (Rabin, 1981), several instantiations of OT protocols have appeared in the literature, including proposals in the UC framework. Some instantiations tried to achieve low communication

*This work was partially funded by French DGA.

Aragon, N., Blazy, O., Fournaise, N. and Gaborit, P.
CROOT: Code-based Round-Optimal Oblivious Transfer.
DOI: 10.5220/0009776700760085
In *Proceedings of the 17th International Joint Conference on e-Business and Telecommunications (ICETE 2020) - SECRYPT*, pages 76-85
ISBN: 978-989-758-446-6

costs (Peikert et al., 2008), while others like Choi et al. (Choi et al., 2013; Blazy and Chevalier, 2015) proposed a generic method to achieve Oblivious Transfer with adaptive corruptions. However, among all the existing articles, only the schemes from (Peikert et al., 2008) (an ad-hoc construction based on lattices) and (Blazy and Chevalier, 2015) (a generic construction relying on (Katz and Vaikuntanathan, 2009)), or (Blazy et al., 2019) (a variant using (Benhamouda et al., 2018)) are UC-Secure in a Post-Quantum setting, and they rely on lattice-based cryptography. Unfortunately, the first construction only fulfills static security. The other one offers adaptive security, but relies on standard-model lattice-based SPHF (Smooth Projective Hash Functions) constructions with costly decryption procedure (either with repetitions or a super polynomial modulus)

Some code-based Oblivious Transfers exist like (Kobara et al., 2008; Barreto et al., 2017), however none of them managed to be proven in the UC setting up to now.

1.2 Contributions

- First, we revisit a framework for round-optimal OT already seen in the literature which requires a certain property on the public keys. Informally, given a random value $\mathcal{T}$, it should not be possible to have two public keys of an asymmetric encryption scheme $\mathsf{pk}_0, \mathsf{pk}_1$ satisfying the relation $\mathcal{T} = \mathsf{pk}_0 \star \mathsf{pk}_1$. To prove this framework adaptively UC-secure, we revise the classical ideal functionality of OT.
- Secondly, we propose multiple instantiations of this framework using rank metric code-based encryption scheme (NIST competitors) such as HQC (Aguilar Melchor et al., 2017a) and RQC (Aguilar Melchor et al., 2017b) with performance evaluation.

We thus propose the first construction of adaptively UC-secure round-optimal OT using quantum-resistant hardness assumptions.

1.3 Organizations of the Paper

After recalling basic definitions in Section 2, we propose our revisited framework in Section 3. We then propose a new functionality for Oblivious Transfer, that is closer to real-life protocols, in Section 4 and prove the security of the framework using it. Finally in Section 5, we instantiate this framework using some code-based NIST candidates and show the efficiency of the corresponding implementations in Section 6.

2 PRELIMINARIES

2.1 Universal Composability

We are going to prove our protocol in the universal composability framework. This framework was introduced in (Canetti, 2001).

In the context of multi-party computation, one wants several users P_i with inputs x_i to be able to compute a specific function $f(x_1,\ldots,x_n) = (y_1,\ldots,y_n)$ without learning anything except y_i. This approach was seen for example in Yao's Millionaires' problem (Yao, 1982), where two millionaires want to know who is richer without revealing their respective wealth. So here, x_i is the wealth of the millionaire i, and f simply returns which one is richer (in this specific case $y_1 = y_i = y_n$).

Instead of following the classical approach which aims to list exhaustively all the expected properties, Canetti did something else and tried to define how a protocol should ideally work.

For that, he divided the world into two spaces, the real world, where the protocol is run with some possible attack, and the ideal world where everything would go smoothly. For a good protocol, it should be impossible to distinguish the real world from the ideal one.

In the ideal world there is an incorruptible entity named the ideal functionality, to which players can send their inputs privately, and then receive the corresponding output without any kind of communication between the players. This way the functionality can be set to be correct, without revealing anything except what is expected.

A protocol, in the real world with an adversary, should create an execution similar to the one obtained by the ideal functionality. This means that the communication between the players should not give more information than the functionality description, and its output. In this case the protocol runs not really against the adversary but against the environment who picks the inputs given to the players, and obtains the outputs. After the interaction the environment should output a bit saying whether he is in the real world.

The main constraint is that the adversary is now free to interact with the environment whenever he wants which prevents the simulator from rewinding when needed. The adversary has access to the communication between the players but not their inputs/outputs, while the environment has only access to the inputs/outputs.

To prove that a protocol realizes the ideal functionality, we consider an environment $\mathcal{Z}$ which can choose inputs given to all the users and whose goal is

to distinguish in which case he receives outputs from the real world execution with an adversary $\mathcal{A}$, and in which case they come from an ideal execution with an ideal adversary $\mathcal{S}$ who interacts solely with the functionality. Such protocol realizes the functionality if for all polynomial adversary $\mathcal{A}$, there exists a polynomial simulator $\mathcal{S}$ such that no environment $\mathcal{Z}$ can distinguish the real world from the ideal one with a non-negligible probability.

In the adaptive corruption setting, the adversary can get complete access to the private credentials and the internal memory of an honest player, and then get control of it, at any time.

More precisely, we will work in the UC framework with joint state proposed by Canetti and Rabin (Canetti and Rabin, 2003) (for the CRS). Informally, this allows different protocols to have some common states while preserving their security. Basically for a given session identifier sid we also define sub-session identifier ssid, and so we have a functionality, possibly generated on the fly, for each ssid.

2.2 Random Oracle

Our constructions will rely on the random oracle model (Bellare and Rogaway, 1993), which can be modelled in the UC framework as the $\mathcal{F}_{\mathsf{RO}}$ -hybrid model. The random oracle functionality is presented in Figure 1.

2.3 UC-secure Oblivious Transfer

The ideal functionality of an Oblivious Transfer (OT) protocol is depicted in Figure 2. It is inspired from (Choi et al., 2013; Abdalla et al., 2013; Byali et al., 2017).

3 FRAMEWORK

We now introduce a framework to build Oblivious Transfer protocols in the Random Oracle Model.

3.1 Star Product

To describe our framework, we need to use a Public-Key Encryption scheme whose keys have the following properties:

Definition 1 (Star Product)**.** *Let $\mathcal{T}$ be a random value, $(\mathsf{pk}_1, \mathsf{pk}_2)$ be two values lying in the public key space of the considered PKE, and $\star$ be on operation between two elements of the public key space.*

If $\mathsf{pk}_1 \star \mathsf{pk}_2 = \mathcal{T}$, we want the following properties :

- ***Sender Security.*** *The Receiver must not be able to know the secret keys associated to both of the public keys*
- ***Receiver Security.*** *The Sender must not be able to tell which secret key is know to the receiver*

3.2 Framework

In figure 3, we present a framework from the earliest version of (Barreto et al., 2017), that we adapt (in particular we add erasures at the critical steps[1]) to achieve **Practical** Oblivious-Transfer. We present later a new UC functionality in Figure 4 and show this framework can achieve it. Erasure means we can voluntarily remove a data from memory so that it's inaccessible in case of a corruption.

4 A NEW MODEL

The classical functionality for Oblivious Transfer has been achieved in many cases, however it often artificially complicates the construction.

Inspired by the work of (Byali et al., 2017), we present an ideal functionality for Oblivious Transfer protocols in figure 4. Our framework which appears to be similar as in the earliest version of (Barreto et al., 2017), as shown in (Li and Micciancio, 2018), cannot be proven UC-secure according to this functionality due to a "timing bug".

In this paper, we propose a new functionality more adapted to the real world. At a high level, we propose to split the `Receive` element into two parts. An $(\texttt{Emit}, \mathsf{sid}, \mathsf{ssid}, P_i, P_j)$ which models that the server is seen sending a message, and a new $(\texttt{Receive}, \mathsf{sid}, \mathsf{ssid}, P_i, P_j, s)$ which models the fact the user receives **and interprets** the flow.

It should be noted that every protocol proven secure in the classical OT functionality is also secure in the one presented in Figure 4. In the other direction, every secure protocol within this functionality is also a secure weak-Oblivious Transfer (Maurer and Ribeiro, 2016)[2].

[1] Without them it would not be possible to prove the framework within our functionality, in particular in case of the server corruption after its flow.

[2] This is trivial, as weak Oblivious Transfer do not require s in the `Receive` message, hence are encompassed by `Emit`.

The Functionality $\mathcal{F}_{\mathsf{RO}}$ is parametrized by a range $\mathcal{D}$, it keeps an initially empty list L of pairs of values (to remember the calls):

- **Upon receiving an input** $(\mathbf{sid}, \mathbf{ssid}, x)$ **from a party** P_i, if there is already a pair (x,h) stored in L, it sets $h_x = h$, otherwise it picks $h_x \xleftarrow{\$} \mathcal{D}$, and stores (x,h_x) in L. In both cases, it replies to its activator with $(\mathsf{sid}, \mathsf{ssid}, h_x)$.

Figure 1: Ideal Functionality for Random Oracle queries $\mathcal{F}_{\mathsf{RO}}$.

The functionality $\mathcal{F}_{(1,2)\text{-OT}}$ is parametrized by a security parameter λ. It interacts with an adversary $\mathcal{S}$ and a set of parties $P_1,\ldots,P_n$ via the following queries:

- **Upon receiving an input** $(\texttt{Send}, \mathbf{sid}, \mathbf{ssid}, P_i, P_j, (m_0,m_1))$ **from party** P_i, with $m_i \in \{0,1\}^\lambda$: record the tuple $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, (m_0,m_1))$ and reveal $(\texttt{Send}, \mathsf{sid}, \mathsf{ssid}, P_i, P_j)$ to the adversary $\mathcal{S}$. Ignore further `Send`-message with the same ssid from P_i.
- **Upon receiving an input** $(\texttt{Receive}, \mathbf{sid}, \mathbf{ssid}, P_i, P_j, s)$ **from party** P_j**,** with $s \in \{0,1\}$: record the tuple $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, s)$, and reveal $(\texttt{Receive}, \mathsf{sid}, \mathsf{ssid}, P_i, P_j)$ to the adversary $\mathcal{S}$. Ignore further `Receive`-message with the same ssid from P_j.
- **Upon receiving a message** $(\texttt{Sent}, \mathbf{sid}, \mathbf{ssid}, P_i, P_j)$ **from the adversary** $\mathcal{S}$: ignore the message if $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, (m_0,m_1))$ or $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, s)$ is not recorded; otherwise send $(\texttt{Sent}, \mathsf{sid}, \mathsf{ssid}, P_i, P_j)$ to P_i and ignore further `Sent`-message with the same ssid from the adversary.
- **Upon receiving a message** $(\texttt{Received}, \mathbf{sid}, \mathbf{ssid}, P_i, P_j)$ **from the adversary** $\mathcal{S}$: ignore the message if $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, (m_0,m_1))$ or $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, s)$ is not recorded; otherwise send $(\texttt{Received}, \mathsf{sid}, \mathsf{ssid}, P_i, P_j, m_s)$ to P_j and ignore further `Received`-message with the same ssid from the adversary.

Figure 2: Ideal Functionality for 1-out-of-2 Oblivious Transfer $\mathcal{F}_{(1,2)\text{-OT}}$.

4.1 Security Proof in this New Functionality

Theorem 2. *The framework explained in Figure 3 achieves the functionality presented in Figure 4 in the ROM, under the security of the star product.*

To prove this theorem, we exhibit a sequence of games. The sequence starts from the real game, where the adversary $\mathcal{A}$ interacts with real players and ends with the ideal game, where we have built a simulator $\mathcal{S}$ that makes the interface between the ideal functionality $\mathcal{F}$ and the adversary $\mathcal{A}$. We prove the adaptive version of the protocol. The proof of the static version can be obtained by removing the parts related to adaptive version from the proof below. We denote as P_i the sender (the server) and P_j the receiver (the user).

Essentially, one first send random values for $\mathcal{T}$, and equivocate using the ROM if need be, then, under the IND-CPA property of the Encryption scheme, one can send dummy values for both u_i. Finally at the time of decryption, one can program the ROM so that when queried a hash on K_b, one can deduce the knowledge of sk_b and so decide that the queried $s = b$, and use the ROM to lead to m_s. This allows to simulate the `Send`, `Emit`, `Receive`-queries respectively to the ideal functionality.

More details follow:

Game $\mathbf{G_0}$: This is the real game.

Game $\mathbf{G_1}$: In this game, the simulator generates correctly every flow from the honest players, as they would do themselves, knowing the inputs (m_0,m_1) and s sent by the environment to the sender and the receiver. In case of corruption, the simulator can give the internal data generated on behalf of the honest players.

Game $\mathbf{G_2}$: We first deal with **honest senders** P_i: when receiving a commitment t, pk_0, the simulator, instead of computing the mask u_b, for $b = 0,1$ with the key K_b, it chooses u_b at random.

When queried on $\mathcal{H}_2(K_b, \mathsf{sid})$, the ROM returns $u_b \oplus m_b$. This game is indistinguishable from the previous one.

In case of corruption, everything has been erased. This game is thus indistinguishable from the previous one under the ROM.

Game $\mathbf{G_3}$: We now deal with **honest receivers** P_j: we replace all the commitments t, pk_0 in Step 1 of the index query phase of honest receivers by honestly generating pk_0 and pk_1 knowing their respective secret keys $\mathsf{sk}_0, \mathsf{sk}_1$, and programming the ROM so that $\mathcal{T} = \mathsf{pk}_0 \star \mathsf{pk}_1$. We then store

Assuming a $\mathsf{crs} = \mathcal{H}_1, \mathcal{H}_2$
Index query on b:
Receiver will generate two encryption keys knowing only one secret key sk_b

- Picks a random seed t, set $\mathcal{T} = \mathcal{H}_1(t, \mathsf{sid}, \mathsf{ssid})$
- For chosen b, run $\mathsf{pk}_b = \mathsf{KeyGen}(1^k, \mathsf{crs})$
- If $b = 1$, sets $\mathsf{pk}_0 = \mathcal{T} \oplus \mathsf{pk}_1$
- Sends (t, pk_0), keeps b, sk_b and erases the rest

Database input (m_0, m_1):
Sender will encrypt both messages m_0, m_1 under their respective keys

- Upon receiving (t, pk_0)
- Regenerates $\mathsf{pk}_1 = \mathsf{pk}_0 \oplus \mathcal{T}$ with t, pk_0
- For each $b \in \{0,1\}$
  - Picks a random $K_b \in \{0,1\}^\lambda$, queries $K'_b = \mathcal{H}_2(K_b, \mathsf{sid})$
  - Compute $C_b = \mathsf{Encrypt}(\mathsf{pk}_b, K_b; r_b)$
  - Computes $u_b = m_b \oplus K'_b$
- Sends (u_0, u_1, C_0, C_1), erases everything afterwards

Data recovery:
Receiver recovers the wanted line m_b

- Upon receiving (u_1, u_2, C_0, C_1)
- For the chosen b, computes $K_b = \mathsf{Decrypt}(\mathsf{sk}_b, C_b)$
- Recovers $m_b = u_b \oplus \mathcal{H}_2(K_b, \mathsf{sid})$

Figure 3: A simple-OT achieving the Practical-OT Functionality.

$(\mathsf{sk}_0, \mathsf{sk}_1)$.

Under the difficulty of distinguishing an honestly generated $\mathcal{T}$ from a crafted one, this game is similar to the precedent.

In case of corruptions, one reveals the valid sk_b and erases the other one.

Game G_4: We now deal with **honest receivers P_j, receiving**: we simply recover m_b by using the appropriate sk_b as usual.

Game G_5: We can now make use of the functionality, which leads to the following simulator:

- when receiving a `Send`-message from the ideal functionality, which means that an honest receiver has sent a first flow, the simulator generates secret keys, and programs the ROM so that $\mathcal{T}$ is the star product of the associated public keys, and sends the corresponding flow;
- after receiving the said flow (from an honest or a corrupted sender) and a `Emitted`-message from the ideal functionality, which means that an honest receiver has sent an index query, the simulator sends honest C_b and random u_b, and uses it to send the corresponding `Emit`-message to the ideal functionality;
- when receiving (u_0, u_1, C_0, C_1) from the adversary (a corrupted sender), the simulator uses both $\mathsf{sk}_0, \mathsf{sk}_1$ to recover u_0, u_1. It uses them to send a `Receive`-message to the ideal functionality.
- when receiving a `Received`-message from the ideal functionality, together with m_s, on behalf of a corrupted receiver, from the extracted s, instead of proceeding as the sender would do on (m_0, m_1), the simulator proceeds on answering $u_s \oplus m_s$ when queried on $\mathcal{H}_2(K_b, \mathsf{sid})$;
- when receiving a flow, generated by an honest sender (by the simulator itself), the simulator proceeds by answering randomly for every $\mathcal{H}_2$ queries

Any corruption before the end does not alter the simulation, a corruption at the `Receive` phase reveals s, and alters one answer of the random oracle... In all cases, a corruption is undetectable as all traces of simulations are erased.

Remark.
It should be noted, that this split in the functionality, avoids the *trick* used in various protocols (Garay

The functionality $\mathcal{F}_{(1,2)\text{-Practical-OT}}$ is parametrized by a security parameter λ. It interacts with an adversary $\mathcal{S}$ and a set of parties $P_1,\ldots,P_n$ via the following queries:

- **Upon receiving an input** $(\texttt{Send}, \textbf{sid}, \textbf{ssid}, P_i, P_j, (m_0, m_1))$ **from party** P_i, with $m_i \in \{0,1\}^\lambda$: record the tuple $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, (m_0, m_1))$ and reveal $(\texttt{Send}, \mathsf{sid}, \mathsf{ssid}, P_i, P_j)$ to the adversary $\mathcal{S}$. Ignore further Send-message with the same ssid from P_i.
- **Upon receiving an input** $(\texttt{Emit}, \textbf{sid}, \textbf{ssid}, P_i, P_j)$ **from party** P_j**,**: record the tuple $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, \mathsf{wait})$ and reveal $(\texttt{Emit}, \mathsf{sid}, \mathsf{ssid}, P_i, P_j)$ to the adversary $\mathcal{S}$. Ignore further Emit-message with the same ssid from P_j.
- **Upon receiving an input** $(\texttt{Receive}, \textbf{sid}, \textbf{ssid}, P_i, P_j, s)$ **from party** P_j**,** with $s \in \{0,1\}$: if a $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, \mathsf{wait})$ existed supersede it with the tuple $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, s)$, and reveal $(\texttt{Receive}, \mathsf{sid}, \mathsf{ssid}, P_i, P_j)$ to the adversary $\mathcal{S}$. Ignore further Receive-message with the same ssid from P_j.
- **Upon receiving a message** $(\texttt{Sent}, \textbf{sid}, \textbf{ssid}, P_i, P_j)$ **from the adversary** $\mathcal{S}$: ignore the message if $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, (m_0, m_1))$ or $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, s)$ is not recorded; otherwise send $(\texttt{Sent}, \mathsf{sid}, \mathsf{ssid}, P_i, P_j)$ to P_i and ignore further Sent-message with the same ssid from the adversary.
- **Upon receiving a message** $(\texttt{Emitted}, \textbf{sid}, \textbf{ssid}, P_i, P_j)$ **from the adversary** $\mathcal{S}$: ignore the message if $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, (m_0, m_1))$ or $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, \mathsf{wait})$ is not recorded; otherwise send $(\texttt{Emitted}, \mathsf{sid}, \mathsf{ssid}, P_i, P_j)$ to P_j and ignore further Emitted-message with the same ssid from the adversary.
- **Upon receiving a message** $(\texttt{Received}, \textbf{sid}, \textbf{ssid}, P_i, P_j)$ **from the adversary** $\mathcal{S}$: ignore the message if $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, (m_0, m_1))$ or $(\mathsf{sid}, \mathsf{ssid}, P_i, P_j, s)$ is not recorded; otherwise send $(\texttt{Received}, \mathsf{sid}, \mathsf{ssid}, P_i, P_j, m_s)$ to P_j and ignore further Received-message with the same ssid from the adversary.

Figure 4: Ideal Functionality for 1-out-of-2 Oblivious Transfer $\mathcal{F}_{(1,2)\text{-Practical-OT}}$.

et al., 2009; Abdalla et al., 2013; Blazy and Chevalier, 2015), where they need an artificial first flow, to encrypt a hiding mask.

5 INSTANTIATIONS

In this section we propose concrete instantiations of the framework presented in Section 3 using the code-based cryptosystems HQC (Aguilar Melchor et al., 2017a) (Hamming metric) and RQC (Aguilar Melchor et al., 2017b) (rank metric). We start by defining the star product of Section 3.1 for the chosen cryptosystems and then discuss the possible bandwidth/complexity tradeoffs.

5.1 High-level Details on the Security When Instantiated with HQC or RQC

In HQC and RQC, a public key pk consists of two vectors $(h,s) \in \mathbb{F}^{2n}$, such that $s = x + hy$, where x and y are of weight w. Thus we can write $\mathcal{T}$ as $(\mathcal{T}_1, \mathcal{T}_2)$, where $\mathcal{T}_i \in \mathbb{F}^n$ and $\mathsf{pk}_1 + \mathsf{pk}_2 = \mathcal{T} \Leftrightarrow (h_1 + h_2 = \mathcal{T}_1, s_1 + s_2 = \mathcal{T}_2)$.

To prove these instantiations of the framework are secure, we need to assume that h_i cannot be chosen by the receiver during the Keygen operation. This can be achieved by setting:

- $h_1 = \mathcal{H}(\mathcal{T})$
- $h_2 = h_1 + \mathcal{T}_1$

When receiving the public keys, the sender checks that the h_i were computed legitimately.

To prove the security of our instantiations, we need to introduce the two following problems:

Problem 1 (Syndrome Decoding)**.** *Given a full-rank matrix $H \in \mathbb{F}^{(n-k)\times n}$, a syndrome $s \in \mathbb{F}^{n-k}$ and a weight w, it is hard to find a vector $x \in \mathbb{F}^n$ of weight lower than w such that $Hx^\top \leqslant s^T$.*

Problem 2 (Decisional SD Problem)**.** *On input $(\mathbf{H}, \mathbf{s}^\top) \in \mathbb{F}^{(n-k)\times n} \times \mathbb{F}^{(n-k)}$, the* Decisional SD Problem $\mathsf{DSD}(n,k,w)$ *asks to decide with non-negligible advantage whether $(\mathbf{H}, \mathbf{s}^\top)$ came from the $\mathsf{SD}(n,k,w)$ distribution or the uniform distribution over $\mathbb{F}^{(n-k)\times n} \times \mathbb{F}^{(n-k)}$.*

These problems can be specialized for both the Hamming metric and the rank metric: in the Hamming metric, $\mathbb{F} = \mathcal{F}$ while in the rank metric, $\mathbb{F} = \mathbb{F}_{q^m}$.

Remark.
The HQC and RQC cryptosystem use a cyclic structure to reprensent matrices in a more compact way, hence a matrix $H \in \mathbb{F}^{(n-k)\times n}$ can be represented by an associated vector $h \in \mathbb{F}^n$. We refer the reader to (Aguilar Melchor et al., 2017a) and (Aguilar Melchor et al., 2017b) for more details about this.

Proposition 3. *Under the assumption that the Syndrome Decoding (respectively Rank Syndrome Decoding) problem [1] is hard, the $+$ operation is a valid star product for the public keys of the HQC (respectively RQC) cryptosystem.*

Proof. **Sender Security.**

Finding two public keys pk_1 and pk_2 such that $\mathsf{pk}_1 + \mathsf{pk}_2 = \mathcal{T}$, knowing that $h_1 = \mathcal{H}_1(\mathcal{T})$ and $h_2 = h_1 + \mathcal{T}_1$, is equivalent to finding $(x_1, x_2, y_1, y_2, s_1, s_2)$ such that :

- $s_1 = x_1 + h_1 y_1$
- $s_2 = x_2 + h_2 y_2$
- $s_1 + s_2 = \mathcal{T}_2$

Where the weight of x_1, x_2, y_1 and y_2 is w, with w a parameter of the cryptosystem.

Solutions to this system are also solutions of the equation:

$$\mathcal{T}_2 = (x_1 + x_2) + h_1 y_1 + h_2 y_2$$

Which is an instance of the syndrome decoding problem 1 where:

- The parity check matrix matrix H is generated by $(1, h_1, h_2)$
- The error vector is $((x_1 + x_2), y_1, y_2)$
- The syndrome is $\mathcal{T}_2$

Hence we can reduce the problem of finding keys that break the sender security in our instantiation to solving these particular instances of syndrome decoding. Let $\mathcal{A}$ denote an adversary against these particular instances. We are going to show that $\mathcal{A}$ can be used to solve hard generic instances of syndrome decoding.

In the following, we are going to consider error vectors of length $3n$. The notation (w_1, w_2, w_3) means that the weight of a vector is w_1 if we take the first n coordinates, w_2 if we take the following n, and w_3 if we take the last n.

Hamming Metric.
We want to solve a random syndrome decoding instance $(H, s, 3w)$ where $H \in \mathcal{F}^{n\times 3n}$ and $s \in \mathcal{F}^n$.

First, by remarking that random instances of weight $3w$ and length $3n$ have, with good probability, a solution of the form (w, w, w), we can reduce the generic problem to these specific instances.

We then use $\mathcal{A}$ to build an adversary $\mathcal{A}'$ which, on input (H, s), proceeds as follows:

- Sample a random vector $e' \in \mathcal{F}^3 n$ of weight $(w, 0, 0)$
- Send $(H, s + He'^{\top})$ to $\mathcal{A}$. Then:
  - If $\mathcal{A}$ outputs a vector e'' then output $e'' - e'$
  - Else output $\emptyset$

$\mathcal{A}$ returns error vectors that have their weight bounded by $4w$, which is below the Gilbert-Varshamov bound. This ensures the uniqueness of the solution, which means that $e'' - e'$ is indeed a solution to the instance (H, s).

Rank Metric.
We want to solve a random rank syndrome decoding instance $(H, s, 2w)$ where $H \in \mathbb{F}_{q^m}^{n\times 2n}$ and $s \in \mathbb{F}_{q^m}^n$.

We use $\mathcal{A}$ to build an adversary $\mathcal{A}'$ which, on input (H, s), proceeds as follows:

- Sample a random vector $h_2 \in \mathbb{F}_{q^m}^n$
- Build H', the concatenation of H and the square matrix generated by the vector h_2. This results in a $n \times 3n$ matrix over $\mathbb{F}_{q^m}$.
- Sample a random vector e' of weight $(w, 0, w)$
- Send $(H', s + H'e'^{\top})$ to $\mathcal{A}$. Then:
  - If $\mathcal{A}$ outputs a vector e'' then output the first $2n$ coordinates of $e'' - e'$
  - Else output $\emptyset$

In the rank metric $\mathcal{A}$ returns error vectors that have their weight bounded by $2w$, which is again below the Gilbert-Varshamov bound. This ensures the uniqueness of the solution, which means that $e'' - e'$ is indeed a solution to the instance (H, s).

From that we deduce that finding $(\mathsf{pk}_1, \mathsf{sk}_1)$ and $(\mathsf{pk}_2, \mathsf{sk}_2)$ such that $\mathsf{pk}_1 + \mathsf{pk}_2 = \mathcal{T}$ reduces to finding a solution to an instance of the syndrome decoding problem.

Receiver Security.
To distinguish between a legitimately generated public key and some (h, s) randomly sampled, the sender would have to solve an instance of the decisional version of the syndrome decoding problem, which is easily reduced to the search version. □

Table 1: Bandwidth (in bytes) used by our instantiations of the OT framework.

Parameter	Security level	Receiver bandwidth	Sender bandwidth
hqc-128-1	128	6274	12724
hqc-192-1	192	11022	22218
hqc-256-1	256	16002	32176
RQC-III	128	2388	9360

Table 2: Timings (in millions of cycles) of our instantiations of the OT framework.

Parameter	Receiver (step 1)	Sender	Receiver (step 2)
hqc-128-1	0.20	0.67	0.45
hqc-192-1	0.35	1.18	0.76
hqc-256-1	0.53	1.85	1.20
RQC-III	0.52	1.51	3.24

1: **Input:** The 512-bit message to encrypt m, the secret key sk
2: **Output:** A ciphertext C of m
3: $(C_1, shared\_secret) \Leftarrow \mathsf{Encaps}(\mathsf{sk})$
4: $m' \Leftarrow m \oplus shared\_secret$
5: **return** $C = C_1||m'$

Figure 5: KEM to PKE conversion used in our implementation.

6 IMPLEMENTATION

6.1 Implementation Details

Since the HQC and RQC schemes are implemented as KEM schemes instead of PKE schemes, we use the conversion 5 to encrypt 512-bit messages. To decrypt, we simply use the left part of C to recover the shared secret and XOR it with the right part of C to recover m.

6.2 Bandwidth and Performances

6.2.1 Bandwidth

Our implementation uses:

- 40 bytes to represent t and 64 bytes to represent sid, hence a 104 bytes overhead over transmitting the public key
- 128 bytes to represent (u_1, u_2) and 128 bytes of overhead coming from the KEM to PKE conversion, hence a 256 bytes overhead over transmitting two ciphertexts

Using that we can compute the bandwidth used by our instantiations for security levels 128, 192 and 256. The results are described table 1. For RQC, we chose to use the parameter RQC-III as a 128 bits of security parameter to take into account the recent improvments on algebraic attacks from (Bardet et al., 2019).

6.2.2 Performances

The most costly operations in the OT framework are by far the KeyGen, $\mathsf{Encrypt}$ and $\mathsf{Decrypt}$ operations, hence the performance of the Oblivious Transfer is really close to the performances of the HQC and RQC schemes. The timings are shown table 2 and are given in million of CPU cycles. The computations were performed using an Intel® Core™ i5-7440HQ.

7 CONCLUSION

We revised the classical ideal functionality of Oblivious Transfer commonly used. Doing so, we were able to prove a round-optimal and efficient framework to be adaptively UC-secure.

We also provided concrete performance evaluation of an implementation of this generic construction using NIST competitors in the Post-Quantum Cryptography Standardization Process. This shows those quantum-resistant instantiations are both practical in communication size but also in computation time.

Our new functionality opens the way to UC secure Round-Optimal Oblivious Transfer with adaptive corruptions in a Post-Quantum setting assuming reliable erasures. An open question, would then naturally be, can we reduce the reliance on the random oracle model without conceding anything to efficiency.

ACKNOWLEDGMENTS

This work was supported in part by the French ANR projects IDFIX (ANR-16-CE39-0004) and CBCrypt (ANR-17-CE39-0007).

REFERENCES

Abdalla, M., Benhamouda, F., Blazy, O., Chevalier, C., and Pointcheval, D. (2013). SPHF-friendly non-interactive commitments. In Sako, K. and Sarkar, P., editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 214–234. Springer, Heidelberg.

Aguilar Melchor, C., Aragon, N., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.-C., Gaborit, P., Persichetti, E., and Zémor, G. (2017a). HQC. Technical report, National Institute of Standards and Technology. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.

Aguilar Melchor, C., Aragon, N., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.-C., Gaborit, P., and Zémor, G. (2017b). RQC. Technical report, National Institute of Standards and Technology. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.

Bardet, M., Briaud, P., Bros, M., Gaborit, P., Neiger, V., Ruatta, O., and Tillich, J.-P. (2019). An algebraic attack on rank metric code-based cryptosystems. *arXiv preprint cs/1910.00810*.

Barreto, P. S. L. M., David, B., Dowsley, R., Morozov, K., and Nascimento, A. C. A. (2017). A framework for efficient adaptively secure composable oblivious transfer in the ROM. Cryptology ePrint Archive, Report 2017/993. http://eprint.iacr.org/2017/993.

Bellare, M. and Rogaway, P. (1993). Random oracles are practical: A paradigm for designing efficient protocols. In Denning, D. E., Pyle, R., Ganesan, R., Sandhu, R. S., and Ashby, V., editors, *ACM CCS 93*, pages 62–73. ACM Press.

Benhamouda, F., Blazy, O., Ducas, L., and Quach, W. (2018). Hash proof systems over lattices revisited. In Abdalla, M. and Dahab, R., editors, *PKC 2018, Part II*, volume 10770 of *LNCS*, pages 644–674. Springer, Heidelberg.

Blazy, O. and Chevalier, C. (2015). Generic construction of UC-secure oblivious transfer. In Malkin, T., Kolesnikov, V., Lewko, A. B., and Polychronakis, M., editors, *ACNS 15*, volume 9092 of *LNCS*, pages 65–86. Springer, Heidelberg.

Blazy, O. and Chevalier, C. (2016). Structure-preserving smooth projective hashing. In Cheon, J. H. and Takagi, T., editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 339–369. Springer, Heidelberg.

Blazy, O., Chevalier, C., and Germouty, P. (2017). Almost optimal oblivious transfer from QA-NIZK. In Gollmann, D., Miyaji, A., and Kikuchi, H., editors, *ACNS 17*, volume 10355 of *LNCS*, pages 579–598. Springer, Heidelberg.

Blazy, O., Chevalier, C., and Vu, Q. H. (2019). Post-quantum uc-secure oblivious transfer in the standard model with adaptive corruptions. In *Proceedings of the 14th International Conference on Availability, Reliability and Security, ARES 2019, Canterbury, UK, August 26-29, 2019*, pages 28:1–28:6. ACM.

Byali, M., Patra, A., Ravi, D., and Sarkar, P. (2017). Fast and universally-composable oblivious transfer and commitment scheme with adaptive security. Cryptology ePrint Archive, Report 2017/1165. https://eprint.iacr.org/2017/1165.

Canetti, R. (2001). Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145. IEEE Computer Society Press.

Canetti, R., Lindell, Y., Ostrovsky, R., and Sahai, A. (2002). Universally composable two-party and multi-party secure computation. In *34th ACM STOC*, pages 494–503. ACM Press.

Canetti, R. and Rabin, T. (2003). Universal composition with joint state. In Boneh, D., editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 265–281. Springer, Heidelberg.

Choi, S. G., Katz, J., Wee, H., and Zhou, H.-S. (2013). Efficient, adaptively secure, and composable oblivious transfer with a single, global CRS. In Kurosawa, K. and Hanaoka, G., editors, *PKC 2013*, volume 7778 of *LNCS*, pages 73–88. Springer, Heidelberg.

Chou, T. and Orlandi, C. (2015). The simplest protocol for oblivious transfer. In Lauter, K. E. and Rodríguez-Henríquez, F., editors, *LATINCRYPT 2015*, volume 9230 of *LNCS*, pages 40–58. Springer, Heidelberg.

Garay, J. A., Wichs, D., and Zhou, H.-S. (2009). Somewhat non-committing encryption and efficient adaptively secure oblivious transfer. In Halevi, S., editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 505–523. Springer, Heidelberg.

Halevi, S. and Kalai, Y. T. (2012). Smooth projective hashing and two-message oblivious transfer. *Journal of Cryptology*, 25(1):158–193.

Horvitz, O. and Katz, J. (2007). Universally-composable two-party computation in two rounds. In Menezes, A., editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 111–129. Springer, Heidelberg.

Ishai, Y., Prabhakaran, M., and Sahai, A. (2008). Founding cryptography on oblivious transfer - efficiently. In Wagner, D., editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 572–591. Springer, Heidelberg.

Katz, J. and Vaikuntanathan, V. (2009). Smooth projective hashing and password-based authenticated key exchange from lattices. In Matsui, M., editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 636–652. Springer, Heidelberg.

Kilian, J. (1988). Founding cryptography on oblivious transfer. In *20th ACM STOC*, pages 20–31. ACM Press.

Kobara, K., Morozov, K., and Overbeck, R. (2008). *Coding-Based Oblivious Transfer*, pages 142–156. Springer Berlin Heidelberg, Berlin, Heidelberg.

Li, B. and Micciancio, D. (2018). Equational security proofs of oblivious transfer protocols. In Abdalla, M. and Dahab, R., editors, *PKC 2018, Part I*, volume 10769 of *LNCS*, pages 527–553. Springer, Heidelberg.

Maurer, U. and Ribeiro, J. (2016). New perspectives on weak oblivious transfer. In *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 790–794.

Naor, M. and Pinkas, B. (2001). Efficient oblivious transfer protocols. In Kosaraju, S. R., editor, *12th SODA*, pages 448–457. ACM-SIAM.

Naor, M. and Pinkas, B. (2005). Computationally secure oblivious transfer. *Journal of Cryptology*, 18(1):1–35.

Peikert, C., Vaikuntanathan, V., and Waters, B. (2008). A framework for efficient and composable oblivious transfer. In Wagner, D., editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 554–571. Springer, Heidelberg.

Rabin, M. O. (1981). How to exchange secrets with oblivious transfer. Technical Report TR81, Harvard University.

Yao, A. C.-C. (1982). Theory and applications of trapdoor functions (extended abstract). In *23rd FOCS*, pages 80–91. IEEE Computer Society Press.