INFORMATICA, 2020, Vol. 31, No. 4, 751–768

DOI: https://doi.org/10.15388/20-INFOR427

Group Key Establishment in a Quantum-Future Scenario

María Isabel GONZÁLEZ VASCO[1,*], Ángel L. PÉREZ DEL POZO[1], Rainer STEINWANDT[2]

[1] *MACIMTE, Universidad Rey Juan Carlos, Spain*
[2] *Dept. of Mathematical Sciences, Florida Atlantic University, USA*
e-mail: mariaisabel.vasco@urjc.es, angel.perez@urjc.es, rsteinwa@fau.edu

Received: February 2020; accepted: August 2020

Abstract. In cryptography, key establishment protocols are often the starting point paving the way towards secure execution of different tasks. Namely, the parties seeking to achieve some cryptographic task, often start by establishing a common high-entropy secret that will eventually be used to secure their communication. In this paper, we put forward a security model for group key establishment (GAKE) with an adversary that may execute efficient quantum algorithms, yet only once the execution of the protocol has concluded. This captures a situation in which keys are to be established in the present, while security guarantees must still be provided in the future when quantum resources may be accessible to a potential adversary.

Further, we propose a protocol design that can be proven secure in this model. Our proposal uses password authentication and builds upon efficient and reasonably well understood primitives: a message authentication code and a post-quantum key encapsulation mechanism. The hybrid structure dodges potential efficiency downsides, like large signatures, of some "true" post-quantum authentication techniques, making our protocol a potentially interesting fit for current applications with long-term security needs.

Key words: Group Key Exchange, post-quantum cryptography, QUANTUM-future cryptography.

1. Introduction

The advent of quantum computing has had a great effect in cryptographic developments, giving rise to different active lines of work. Some efforts focus on finding new constructions exploiting the great potential of quantum technology (*Quantum Key Distribution* schemes being the flagship example), while others target design strategies transitioning from classical to quantum resistant schemes.

In this contribution, we focus on group key exchange protocols (GAKE), which are cryptographic constructions allowing a group of $n \geqslant 2$ participants to agree upon a high-entropy secret key. Communication is carried out over an insecure channel, and thus legitimate participants need to authenticate themselves (if not necessarily as specific individuals, at least as legitimate group members). It is typical to assume in this context

*Corresponding author.

that the network is fully under adversarial control, and thus a potential adversary may not only eavesdrop, but also delay, suppress, or insert messages at will. On top of the standard security challenges encountered in this framework, significant difficulties arise when considering adversaries that have access to quantum computing—the so-called post-quantum setting. Basic building blocks behind a post-quantum GAKE (such as encryption or commitment schemes) should be proven secure in this new restricted scenario, where primitives based on the hardness of factoring or computing discrete logarithms in certain groups can no longer be trusted. While a number of primitives for *post-quantum* cryptographic tasks are available, restricting to this kind of tools comes at a prize in terms of computational cost, memory, bandwidth, etc. The question arises whether it is possible to put off some of the cost that comes with an immediate transition to a "full-fledged post-quantum design" without jeopardizing the long-term security of established session keys. This is where a *future-quantum* scenario comes in.

Related work

Two-party constructions. A number of two-party key establishment protocols have been proposed taking into account quantum adversaries with diverse security models and levels of formalism. Many of these proposals are not contributory key exchange protocols, but key encapsulation mechanisms (KEMs), allowing one party to send a high-entropy key to another one, which can be later used to secure their two-party communication; submissions to NIST's ongoing standardization effort provide various examples of current candidates for post-quantum KEMs National Institute of Standards and Technology (2019).

When it comes to secure joint key generation of two-party keys, fewer proposals are available in the literature. In Bos *et al.* (2015), an unauthenticated Diffie-Hellman-like key exchange protocol is proposed, based on the ring learning with errors (RLWE) problem, and the authors demonstrate its practicality, integrating it in TLS cipher suits. Their protocol is only secure for passive adversaries, and classical (non-quantum resistant) authentication means—such as RSA signatures—are suggested in order to dodge active attacks by standard adversaries. Also considering different levels of quantum-precautions with respect to authentication, Bindel *et al.* (2018) builds a hybrid key exchange protocol, in the two party scenario, which uses (post-quantum) KEMs as a fundamental building block. More precisely, they present a compiler for authenticated key exchange that can be built from a passively secure KEM, a signature scheme, a message authentication code and a secure key derivation function. Depending on the security of these building blocks, different guarantees are proven with respect to two-stage adversaries.[1]

Ding et al.'s recent work Ding *et al.* (2019b) is worth mentioning, too. They gave a somewhat informal proposal for two-party key exchange constructions based on the short integer solution problem and learning with errors. In a similar fashion, two-party constructions using the ring learning with errors problem as a base can be found in Ding *et al.* (2019a). Finally, in a paper presented at PKC 2018, Benhamouda *et al.* (2018) refine

[1] We later mimic their approach, where adversaries are modelled differently in two well-defined attack stages, where the quantum/classical capabilities may differ.

prior work by Katz and Vaikuntanathan (2009) to obtain a very strong type of *smooth projective hash functions* over lattices. As a byproduct, they present a one-round two-party password-authenticated key establishment.

Group constructions. Already Ding *et al.* (2012) gave a proposal for mimicking Diffie-Hellman constructions for key exchange using different variants of the *learning with errors* problem. However, no security proof was provided for the group version of their protocol. Recently, in Apon *et al.* (2019), a constant-round protocol for group key exchange is proposed and proven secure in a passive scenario. Using a post-quantum signature scheme this construction can be made secure in the presence of active adversaries, by means of a well known compiler from Katz and Yung (see Katz and Yung, 2007). This construction adapts to the Ring-LWE scenario a well known circular design introduced by Burmester and Desmedt (2005). While being a major contribution, this proposal carries over many of the hardships of a lattice-based construction; in particular, a bound on the number of maximum parties the protocol may support for both correctness and security (depending on the ring, the noise distributions and the security parameters). Finally, here the (unfortunately, only theoretical) work of Boneh *et al.* (2018) should be mentioned, where first steps towards a post-quantum secure group key establishment construction based on isogenies are taken.

Going from 2-party to group. Instead of using a direct design like ours, one possible approach for deriving group key establishment with some security guarantees in the presence of a quantum adversary is to use the compiler of Abdalla *et al.* (2007) from a secure two-party construction. However, complexity drawbacks of a post-quantum authentication method can make such a construction rather inefficient. As it is password-based, a compiled protocol from the two-party PAKE of Benhamouda *et al.* (2018) will not be hindered by this, while due to the (rather sophisticated) tools used in this construction, a thorough analysis would be needed to be able to understand the efficiency of such a design. For other two-party schemes, such as the one presented in Bindel *et al.* (2018, Section 4), it is the round-efficiency where our construction surpasses this compiling approach.

Our contribution

In this work we focus on a scenario that tries to capture today's reality: Participants engage in a GAKE execution today, assuming no quantum-adversary is present, and establish a common secret key that should remain secure even if, in the future, an adversary eventually obtains access to quantum computing capabilities. We adapt the (by now standard) security model for GAKE to capture this kind of evolution of adversarial capabilities, and put forward a protocol design that can be proven secure in this model. Our proposal uses password-based authentication and builds on rather non-expensive primitives: a message authentication code and a post-quantum key encapsulation mechanism.

2. Model

Our modelling and construction follow the approach of recent work by Bindel *et al.* for signature schemes (Bindel *et al.*, 2017) and KEMs (Bindel *et al.*, 2018), who consider

security against adversaries with different levels of quantum-computing capabilities over time. Our adversaries will be merely classical during the protocol run, but may take advantage of quantum computing once the execution under attack is finished and accepted by the involved participant. Users are modelled as probabilistic polynomial time (ppt) Turing machines. We build on classical models for group key establishment as introduced in Bellare and Rogaway (1994), Bellare *et al.* (2000), Bresson *et al.* (2001). The potential set of users $\mathcal{U}$ is assumed to be of polynomial size (in the security parameter 1^{ℓ}), and each user $U \in \mathcal{U}$ may execute a polynomial number of protocol *instances* concurrently. To refer to instance s_i of a user $U_i \in \mathcal{U}$ we use the notation $\Pi_i^{s_i}$ ($i \in \mathbb{N}$).

Protocol instances. A single instance $\Pi_i^{s_i}$ can be taken for a process executed by U_i. To each instance we assign seven variables:

$\mathsf{used}_i^{s_i}$ indicates whether this instance is or has been used for a protocol run. The $\mathsf{used}_i^{s_i}$ flag can only be set through a protocol message received by the instance due to a call to the Execute- or the Send-oracle (see below);
$\mathsf{state}_i^{s_i}$ keeps the state information needed during the protocol execution;
$\mathsf{term}_i^{s_i}$ shows if the execution has terminated;
$\mathsf{sid}_i^{s_i}$ denotes a possibly public session identifier that can serve as identifier for the session key $\mathsf{sk}_i^{s_i}$;
$\mathsf{pid}_i^{s_i}$ stores the set of identities of those users that $\Pi_i^{s_i}$ aims at establishing a key with—including U_i himself;[2]
$\mathsf{acc}_i^{s_i}$ indicates if the protocol instance was successful, i.e. the user accepted the session key;
$\mathsf{sk}_i^{s_i}$ stores the session key once it is accepted by $\Pi_i^{s_i}$. Before acceptance, it stores a distinguished NULL value.

For more details on the usage of the variables we refer to Bellare *et al.* (2000).

Communication network. Arbitrary point-to-point (peer-to-peer) connections among the users are assumed to be available. Thus, the network topology is that of a complete graph. We assume the network to be non-private, however, and fully asynchronous. More specifically, it is controlled by the adversary, who may alter, delay, insert, and delete messages at will.

Adversarial capabilities. Our adversaries are only capable of executing tasks in probabilistic polynomial time, and they are restricted to classical algorithms. The capabilities of an adversary $\mathcal{A}$ are made explicit through a number of *oracles* allowing $\mathcal{A}$ to communicate with protocol instances run by the users, and we use an oracle to capture future quantum capabilities of $\mathcal{A}$:

[2] Dealing with authentication through a shared password exclusively, we do not consider key establishments among strict subsets of $\mathcal{U}$. With $\mathsf{pid}_i^{s_i} := \mathcal{U}$ being the only case of interest, in the sequel we do not make explicit use of $\mathsf{pid}_i^{s_i}$ when defining partnering, integrity, etc.

$\mathsf{Send}(U_i, s_i, M)$ This sends message M to the instance $\Pi_i^{s_i}$ and returns the reply generated by this instance. If $\mathcal{A}$ queries this oracle with an unused instance $\Pi_i^{s_i}$ and M being the string "Start", the $\mathsf{used}_i^{s_i}$-flag is set, and the initial protocol message of $\Pi_i^{s_i}$ is returned.

$\mathsf{Execute}(\{\Pi_{u_1}^{s_{u_1}}, \ldots, \Pi_{u_\mu}^{s_{u_\mu}}\})$ This executes a complete protocol run among the specified unused instances of the respective users. The adversary obtains a transcript of all messages sent over the network. A query to the Execute oracle is supposed to reflect a passive eavesdropping. In particular, no online-guess for the secret password can be implemented with this oracle.

$\mathsf{Reveal}(U_i, s_i)$ yields the session key $\mathsf{sk}_i^{s_i}$ along with its corresponding session identifier $\mathsf{sid}_i^{s_i}$.

$\mathsf{Test}(U_i, s_i)$ Only one query of this form is allowed for an active adversary $\mathcal{A}$. Provided that $\mathsf{sk}_i^{s_i}$ is defined, (i.e. $\mathsf{acc}_i^{s_i} = \mathsf{true}$ and $\mathsf{sk}_i^{s_i} \neq \text{NULL}$), $\mathcal{A}$ can execute this oracle query at any time when being activated. Then with probability 1/2 the session key $\mathsf{sk}_i^{s_i}$ and with probability 1/2 a uniformly chosen random session key is returned.

$\mathsf{Corrupt}(U_i)$ This oracle returns all long-term secrets held by user U_i (e.g. a password or static keys for an authentication mechanism). The Corrupt-oracle's only purpose is to model forward secrecy.

$\mathsf{QCom}(c)$ This oracle is used to capture future quantum computation abilities. It accepts a polynomial-size description c (a quantum circuit) of a quantum computation that can be executed in polynomial time, e.g. computing a discrete logarithm. The oracle returns a classical value, representing the result of the specified quantum computation.

2.1. *Correctness, Integrity and Secrecy*

Before we define correctness, integrity, and secrecy, we introduce *partnering* to express which instances are associated in a common protocol session.

Partnering. We adopt the notion of partnering from Bohli *et al.* (2007). Namely, we refer to instances $\Pi_i^{s_i}$, $\Pi_j^{s_j}$ as being *partnered* if both $\mathsf{sid}_i^{s_i} = \mathsf{sid}_j^{s_j}$ and $\mathsf{acc}_i^{s_i} = \mathsf{acc}_j^{s_j} = \mathsf{true}$.

To avoid trivial cases, we assume that an instance $\Pi_i^{s_i}$ always accepts the session key constructed at the end of the corresponding protocol run if no deviation from the protocol specification occurs. Moreover, all users in the same protocol session should come up with the same session key, and we capture this in the subsequent notion of correctness.

Correctness. We call a group key establishment protocol $\mathcal{P}$ *correct*, if in the presence of a passive adversary $\mathcal{A}$—i.e. $\mathcal{A}$ must not use the Send oracle—the following holds: for all i, j with both $\mathsf{sid}_i^{s_i} = \mathsf{sid}_j^{s_j}$ and $\mathsf{acc}_i^{s_i} = \mathsf{acc}_j^{s_j} = \mathsf{true}$, we have $\mathsf{sk}_i^{s_i} = \mathsf{sk}_j^{s_j} \neq \text{NULL}$.

Key integrity. Correctness takes only passive attacks into account, whereas *key integrity* does not restrict the adversary's oracle access: a correct group key establishment protocol fulfills *key integrity*, if with overwhelming probability all instances of users that have accepted with the same session identifier $\mathsf{sid}_j^{s_j}$ hold identical session keys $\mathsf{sk}_j^{s_j}$.

Next, for detailing the security definition, we will have to specify under which conditions a Test-query may be executed.

Freshness. A Test-query should only be allowed to those instances holding a key that is not for trivial reasons known to the adversary. To this aim, an instance $\Pi_i^{s_i}$ is called *fresh* in case none of the events below occurred:

- A $\mathsf{Corrupt}(U_j)$ query is executed before a query of the form $\mathsf{Send}(U_k, s_k, *)$ takes place. (One could consider to restrict to $U_j, U_k \in \mathsf{pid}_i^{s_i}$, but as indicated in footnote 2, $\mathsf{pid}_i^{s_i} = \mathcal{U}$ is the only case of interest.)
- A Send query is performed, after a QCom query took place.
- A $\mathsf{Reveal}(U_j, s_j)$ query is executed with $\Pi_i^{s_i}$ and $\Pi_j^{s_j}$ being partnered.

As usual, the idea is that revealing a session key from an instance $\Pi_i^{s_i}$ trivially yields the session key of all instances partnered with $\Pi_i^{s_i}$, and hence this kind of "attack" will be excluded in the security definition. Similarly, if an adversary eventually succeeds in corrupting a legitimate group member and controls him fully while the protocol is being executed, he will of course know the resulting session key, and that situation is also excluded through our freshness definition. Further, this definition formalizes our exclusion of on-line quantum attacks; namely, we also restrict Test queries to those sessions for which no quantum process has been executed during the actual protocol run.

In our construction we will focus on password-based authentication. Thus, we must assume users select their passwords from a given public dictionary $\mathcal{D}$, of polynomial size in the security parameter ℓ. Groups are as a result defined by a set of users which use the same password in $\mathcal{D}$ for authentication. The security goal aimed at is defined in terms of the advantage $\mathsf{Adv}_{\mathcal{A}}$ of an adversary $\mathcal{A}$ in attacking protocol $\mathcal{P}$. This advantage is a function in the security parameter ℓ, defined as

$$\mathsf{Adv}_{\mathcal{A}} := |2 \cdot \mathsf{Succ} - 1|.$$

Here Succ is the probability that the adversary queries Test on a fresh instance $\Pi_i^{s_i}$ and guesses correctly the bit b used by the Test oracle in a moment when $\Pi_i^{s_i}$ is still fresh.

Definition 1. A group key exchange protocol $\mathcal{P}$ provides *quantum-future key secrecy*, if for every dictionary $\mathcal{D}$ and every ppt adversary $\mathcal{A}$ querying the Send-oracle with at most q different protocol instances, the following inequality holds for some negligible function $\mathrm{negl}(\ell)$:

$$\mathsf{Adv}_{\mathcal{A}}(\ell) \leqslant \varepsilon(\ell, q) + \mathrm{negl}(\ell),$$

where ℓ is the security parameter and ε is a function which is at most linear in q.

Remark 1. The above definition follows the standard approach in password authenticated key exchange, where typically the function ε is a constant multiple of $\frac{q}{|\mathcal{D}|}$ plus some negligible term (thus, it is assumed that passwords are chosen uniformly at random from the dictionary and that the adversary can test a constant number of passwords on each Send-query).

3. Tools

Our construction invokes two well-studied cryptographic primitives: a key encapsulation mechanism (KEM) and a message authentication code (MAC). We use them in a black-box way, in the sense that specific details of the primitives do not affect our security claims, as long as the required security properties are fulfilled. The KEM needs to be resistant against quantum adversaries and NIST's ongoing standardization effort offers various candidates (National Institute of Standards and Technology, 2019), whose security relies on the intractability of several families of mathematical problems. For instance, CRYSTALS-Kyber (Bos *et al.*, 2018), Frodo (Bos *et al.*, 2016), NewHope (Alkim *et al.*, 2016) or NTRU Prime (Bernstein *et al.*, 2017b) are lattice-based KEMs, BIKE (Aragon *et al.*, 2017) or Classic McEliece (Bernstein *et al.*, 2017a) are code-based, while SIKE (Azarderakhsh *et al.*, 2017) is an isogeny-based proposal. On the other hand, for the choice of the MAC one can rely on a popular construction like Poly1305 (Bernstein, 2005).

3.1. *Key Encapsulation Mechanisms*

Key encapsulation mechanisms are public key algorithms suited for the generation and transfer of a high entropy key for later use. To achieve this, first, a pair of keys is generated, one of them being public while the other must be kept secret. Any entity holding the public key is able to run an *encapsulation* algorithm, which outputs a fresh, high entropy, key and a ciphertext which "encapsulates" it. The user holding the secret key can, upon reception of this ciphertext, run a *decapsulation* algorithm, which outputs the same fresh key, shared, from that moment, between both users.

More formally, a *key encapsulation mechanism* (KEM) is a triple of algorithms $\mathcal{K} = (\mathcal{K}.\mathsf{KeyGen}, \mathsf{Encaps}, \mathsf{Decaps})$, where:

- The probabilistic *key generation* algorithm $\mathcal{K}.\mathsf{KeyGen}(1^\ell)$ takes as input the security parameter and outputs a key pair (pk, sk).
- The probabilistic *encapsulation* algorithm $\mathsf{Encaps}(pk, 1^\ell)$ takes as input a public key pk and outputs a ciphertext c and a key $k \in \{0,1\}^{p(\ell)}$, where $p(\ell)$ is a polynomial function of the security parameter.
- The deterministic *decapsulation* algorithm $\mathsf{Decaps}(sk, c, 1^\ell)$ takes as input a secret key sk and a ciphertext c and outputs a key k or $\perp$.

A KEM $\mathcal{K}$ is *correct* if for all $(pk, sk) \leftarrow \mathcal{K}.\mathsf{KeyGen}(1^\ell)$ and $(c, k) \leftarrow \mathsf{Encaps}(pk, 1^\ell)$, we have $\mathsf{Decaps}(sk, c, 1^\ell) = k$.

As a security requirement, we adopt IND-CPA security against fully quantum adversaries from Bindel *et al.* (2017). This intuitively means that any eavesdropper, that may have access to quantum computation, is unable to obtain any information about the shared fresh key, which is, from its point of view, indistinguishable from a key chosen uniformly at random from the key space. In more detail, an IND-CPA experiment is defined, where a challenger $\mathcal{C}$ generates $(pk, sk) \leftarrow \mathcal{K}.\mathsf{KeyGen}(1^\ell)$ and $(c^*, k_0^*) \leftarrow \mathsf{Encaps}(pk, 1^\ell)$, chooses uniformly at random a key $k_1^* \in \{0,1\}^{p(\ell)}$ and a bit $b \in \{0,1\}$. Then the adversary

$\mathcal{A}$, which is treated as a quantum algorithm, is given (pk, c^*, k_b^*) and produces an output $b' \in \{0, 1\}$ as the guess for b. With $\mathsf{Succ}_{\mathcal{A}}$ being the probability that $\mathcal{A}$'s output equals b, we define $\mathsf{Adv}_{\mathcal{A},\mathtt{KEM}} := |2 \cdot \mathsf{Succ}_{\mathcal{A}} - 1|$ and say that the KEM is IND-CPA secure against fully quantum adversaries if for every polynomial-time bounded $\mathsf{Adv}_{\mathcal{A},\mathtt{KEM}}$, the advantage $\mathsf{Adv}_{\mathcal{A},\mathtt{KEM}}$ is negligible.

3.2. *Message Authentication Codes*

A message authentication code is a symmetric key primitive whose purpose is to preserve information integrity. To achieve this, whenever two users holding the same secret key wish to communicate, the one sending the message produces an authentication tag computed from the message and the secret key. When the other user receives the message together with the tag, the secret key allows him to check the validity of the tag with respect to the message. This procedure protects against an adversary modifying the message without being detected, because if the MAC is secure he will be unable to produce a valid tag for any different message.

More formally, a *message authentication code* (MAC) (Dodis *et al.*, 2012) is a triple of algorithms $\mathcal{M} = (\mathcal{M}.\mathsf{KeyGen}, \mathsf{Tag}, \mathsf{Vf})$ where:

- The probabilistic *key generation* algorithm $\mathcal{M}.\mathsf{KeyGen}(1^\ell)$ takes as input the security parameter and outputs a key $k \in \{0, 1\}^{m(\ell)}$ for a suitable polynomial $m(\ell)$.[3]
- The probabilistic *authentication* algorithm $\mathsf{Tag}(k, M)$ takes as input a key k and a message M and outputs a tag t.
- The deterministic *verification* algorithm $\mathsf{Vf}(k, M, t)$ takes as input a key k, a message M and a tag t and outputs a decision: 1 (accept) or 0 (reject).

The standard security notion for a MAC is *unforgeability under chosen message and chosen verification queries attack* (UF-CMVA) (Dodis *et al.*, 2012). This essentially means that an adversary cannot produce a valid tag for a message of his choice, even if he has access to tags of other messages of his choice and is able to check validity of pairs message-tag (via the so called *verification oracle*). To formally capture this notion, an experiment is defined where a challenger $\mathcal{C}$ generates $k \leftarrow \mathcal{M}.\mathsf{KeyGen}(1^\ell)$ and the adversary $\mathcal{A}$ is granted oracle access to $\mathsf{Tag}(k, \cdot)$ and $\mathsf{Vf}(k, \cdot, \cdot)$. The adversary wins if $\mathcal{A}$ makes a query (M^*, t^*) to $\mathsf{Vf}(k, \cdot, \cdot)$ such that the output is 1 and M^* has not been queried to $\mathsf{Tag}(k, \cdot)$. The MAC is said to be UF-CMVA secure if for all ppt adversaries $\mathcal{A}$, the probability $\mathsf{Succ}_{\mathcal{A}}$ of the adversary winning the previous experiment is negligible in the security parameter ℓ.

If Tag is a deterministic algorithm, then Vf does not need to be explicitly defined, since it is specified by $\mathsf{Vf}(k, M, t) = 1$ if and only if $\mathsf{Tag}(k, M) = t$.

[3] In the sequel, for the sake of simplicity, we will assume $\mathcal{M}.\mathsf{KeyGen}(1^\ell)$ actually selects k uniformly at random in $\{0, 1\}^{m(\ell)}$.

3.3. *Deterministic Randomness Extraction*

In the protocol to be discussed in Section 4, we use a prime order group G in which the Decision-Diffie-Hellman assumption holds, and from (uniform random) elements in G, we want to extract (uniform random) bit-strings. To realize this without the introduction of additional technical machinery or a random oracle, work in Chevassut *et al.* (2006) comes in handy. Specifically, if we let G be the group of quadratic residues in $\mathbb{Z}^{\times}_{2|G|+1}$ with a Sophie-Germain prime $|G|$ close to a power of 2, simple truncation of the binary representation is an efficient extractor (Chevassut *et al.*, 2006, Section 3.2). Subsequently, for a (uniform random) group element $g \in G$, we denote by $[g]$ (statistically close to uniform random) bits extracted deterministically from g. When extracting two independent (half-length) bit-strings from g, we take $[g] = [g]_L||[g]_R$ as concatenation of two (half length) bit-strings.

Remark 2. An elegant and more efficient approach to randomness extraction, which enables the use of elliptic curves over prime fields $\mathbb{F}_p$ with p being close to a power of 2, is provided by Chevassut et al.'s *Twist-AUgmented (TAU)* technique. When trying to optimize the performance of the protocol below, a possible adoption of the TAU approach is natural to explore. Similar to Chevassut *et al.* (2006, Fig. 1) one could—at the bandwidth cost of essentially two parallel Diffie-Hellman executions—try to work on a curve and its twist simultaneously, eventually leveraging the message authentication code to select the correct protocol outcome.

4. The Proposed Protocol

Let $\mathcal{D}$ be the (polynomial-size) password dictionary, and let G be a group of prime order q. We assume that $\mathcal{D} \subseteq G$ through some public and efficiently computable injection $\mathcal{D} \hookrightarrow G$. For instance, if $G = \langle g \rangle$ is the group of quadratic residues in $\mathbb{Z}_{2|G|+1}$ with a Sophie-Germain prime $|G|$, we can identify the binary representation of (length-restricted) passwords with elements in $\mathbb{Z}_{|G|}$, and then map passwords uniquely into G by raising the generator g to the appropriate power.[4] Finally, let ℓ denote the security parameter and $p(\ell)$ denote the bit-length of session keys. We impose a Decisional Diffie Hellman assumption as follows:

Assumption 1 (*Decisional Diffie Hellman for* $(G, \mathcal{D})$). Let G be a finite group of prime order and $\mathcal{D}$ a dictionary with an efficient injection $\iota : \mathcal{D} \hookrightarrow G$. Then, the Decisional Diffie Hellman assumption for $(G, \mathcal{D})$ states that, for every $g \in \iota(\mathcal{D})$, the probability distributions (g^a, g^b, g^{ab}) for $(a, b) \leftarrow \mathbb{Z}^2_{|G|}$ and (g^a, g^b, h) for $(a, b, h) \leftarrow \mathbb{Z}^2_{|G|} \times G$ are computationally indistinguishable. In other words, no ppt algorithm can tell them apart with non-negligible probability in the security parameter ℓ.

[4] In view of Definition 1, simply squaring the password pw, seen as group element, has the downside that with any incorrect guess pw' for pw, an adversary can exclude $-\mathsf{pw}'$, too.

4.1. *Protocol Specification*

Let $U_0, U_1, \ldots, U_n$ be the users running the protocol and assume they share password pw. Further, we will assume that every user is aware of his index and the indices of the rest of participants. Our construction is depicted in Fig. 1, while in Fig. 2 we give a somewhat simplified description for the case of four users.

The basic idea is a simple key transport from U_0 to all the other parties, with the actual session key k being masked (for each of them) with an ephemeral key obtained from the key encapsulation. To ensure (password-based) authentication, in parallel each user establishes a Diffie-Hellman key with U_0, with the shared password fixing a generator for the Diffie-Hellman group. The latter keys are used to compute "after the fact" authentication tags on protocol messages. Once a player is convinced that all protocol messages are legitimate, the session key is accepted. More precisely:

- In Round I, U_0 broadcasts a group element, g_0, obtained as an encoding of his password, which is his "Diffie-Hellman contribution" for the authentication tags. Any other user U_i broadcasts similarly a group element g_i, and his freshly generated public key pk_i for the key encapsulation mechanism.
- In Round II, user U_0 generates for each user U_j a KEM key, and masks with it a (freshly chosen for each run) bitstring k (which will eventually be the session key). This message is authenticated using a bit-string extracted from $g_{0,j}$. Furthermore, users also broadcast confirmation tags pairwise. Namely, U_i and U_j extract two different MAC keys from the shared element $g_{i,j}$, that is, U_i uses $[g_{i,j}]_L$ for users "on his left" (i.e. for $j > i$) and $[g_{i,j}]_R$ for users "on his right" ($i < j$).
- Finally, all tags are checked, and if they are successful then each U_i ($i > 0$) decapsulates the bitstring k and extracts from it the session key and its corresponding session identifier. Note that if a user U_i is inserting an invalid password, the two party Diffie Hellman key $g_{i,j}$ he will be able to construct will (with overwhelming probability) not match the one constructed by U_j, hence it will be detected as a tag-mismatch.

Remark 3. Note that our solution is not contributory (for U_0 fully determines the session key established by the execution). Still, as it often happens in this type of construtions users are assumed to be honest and thus the security definition does not impose that all parties influence the value of the session key. Moreover, if a contributory solution is preferred, one could, e.g. deterministically extract random bits from the g_i-values in Round I, and exclusive-or these with the session key.

The following theorem establishes security of the proposed protocol in the sense of Definition 1. In our formal analysis, the fact that legitimate users are authenticated as such comes from the strength of the MAC, as can be seen in Game 1 in the proof below. Note also that MAC keys are generated using passwords as fresh inputs for Diffie-Hellman exponents, thus we also take into account the probability of a password guess and the hardness of the Decisional Diffie Hellman problem in our reduction. Further, note that the session key is freshly generated (uniformly at random) by U_0 on each execution

Round I:

- `Computation:`
  - For $0 \leqslant i \leqslant n$, U_i chooses $\beta_i \leftarrow \mathbb{Z}_q$ and computes $g_i = \iota(\mathsf{pw})^{\beta_i}$. U_0 sets $M_0 := (g_0)$.
  - For $1 \leqslant i \leqslant n$, U_i computes $(pk_i, sk_i) \leftarrow \mathcal{K}.\mathsf{KeyGen}(1^\ell)$, and sets $M_i := (pk_i, g_i)$.
- `Communication:`
  - For $0 \leqslant i \leqslant n$, U_i broadcasts M_i.

Round II:

- `Computation:`
  - Keying material:
    - U_0 chooses $k \leftarrow \{0,1\}^{p(\ell)}$, and for each $1 \leqslant j \leqslant n$ computes

$$(c_j, k_j) \leftarrow \mathsf{Encaps}(pk_j, 1^\ell)$$

      and sets $d_j := k \oplus k_j$, $m_{0,j} := (d_j, c_j)$.
    - For $0 \leqslant i \leqslant n$, U_i sets $g_{i,j} := g_j^{\beta_i}$ for each $j \neq i$.
  - Tags:
    - U_0 computes $t_{0,j} := \mathsf{Tag}([g_{0,j}]_L, U_0||m_{0,j}||M_0||\ldots||M_n)$ for each $1 \leqslant j \leqslant n$.
    - For $1 \leqslant i \leqslant n$, U_i computes

$$t_{i,j} := \begin{cases} \mathsf{Tag}([g_{i,j}]_L, U_i||M_0||\ldots||M_n), & \text{for } 1 \leqslant i < j \leqslant n. \\ \mathsf{Tag}([g_{i,j}]_R, U_i||M_0||\ldots||M_n), & \text{for } 0 \leqslant j < i \leqslant n. \end{cases}$$

- `Communication:`
  - U_0 sends $(m_{0,j}, t_{0,j})$ to U_j $(1 \leqslant j \leqslant n)$.
  - For $1 \leqslant i \leqslant n$,

$$U_i \text{ sends } t_{i,j} \text{ to } U_j \ (0 \leqslant j \leqslant n,\ j \neq i)$$

    .
- `Verification and key computation:`
  - All users verify all tags:
    - For $1 \leqslant j \leqslant n$, U_j runs $\mathsf{Vf}(([g_{0,j}]_L, U_0||m_{0,j}||M_0||\ldots||M_n), t_{0,j})$.
    - For $0 \leqslant j \leqslant n$, U_j runs $\begin{cases} \mathsf{Vf}([g_{j,i}]_L, U_i||M_0||\ldots||M_n, t_{i,j}), & \text{for } 1 \leqslant i < j \leqslant n. \\ \mathsf{Vf}([g_{j,i}]_R, U_i||M_0||\ldots||M_n, t_{i,j}) & \text{for } 0 \leqslant j < i \leqslant n. \end{cases}$
  - If all checks are successful, U_0 sets

$$\mathsf{ssk}_0 = [k]_L, \mathsf{sid}_0 := [k]_R$$

    .
  - For $1 \leqslant i \leqslant n$, if all checks are successful, U_i runs $\mathsf{Decaps}(sk_i, c_i)$ to obtain k_i, computes $K_i := d_i \oplus k_i$, and sets

$$\mathsf{ssk}_i := [K_i]_L \text{ and } \mathsf{sid}_i := [K_i]_R.$$

Fig. 1. Proposed password-based group-key establishment.

(in the proof, this results in the fact that Game 1 and Game 2 are indistinguishable unless the security of the underlying KEM is compromised).

Round I:

- `Computation`: For each $i = 0, \ldots, 3$, each user U_i selects a group element g_i by encoding his password and raising it to a fresh random exponent β_i:

$$g_0 := \iota(\mathsf{pw})^{\beta_0}, g_1 := \iota(\mathsf{pw})^{\beta_1}, g_2 := \iota(\mathsf{pw})^{\beta_2}, g_3 := \iota(\mathsf{pw})^{\beta_3}.$$

  Further, U_i freshly generates a key pair (pk_i, sk_i) for the KEM.
- `Communication`: For $0 \leqslant i \leqslant 3$, U_i broadcasts M_i:

$$M_0 := (g_0), M_1 := (pk_1, g_1), M_2 := (pk_2, g_2), M_3 := (pk_3, g_3).$$

Round II:

- `Computation`:
  - Keying material:
    - U_0 chooses $k \leftarrow \{0,1\}^{p(\ell)}$. Further, he encapsulates a key k_j as c_j for $j = 1, 2, 3$, and prepares messages

$$m_{0,1} = (k \oplus k_1, c_1), m_{0,2} = (k \oplus k_2, c_2), m_{0,3} = (k \oplus k_3, c_3).$$

    - For $0 \leqslant i \leqslant 3$, U_i sets an (ephemeral) Diffie-Hellman key with U_j,

$$g_{i,j} := g_j^{\beta_i} \text{ for each } j \neq i.$$

  - Tags: Tags will be computed for message flows including the sender's identifier with keys obtained from the exchanged Diffie-Hellman keys (e. g., the message from U_1 to U_2 is tagged with the half most-significant bits of $g_{1,2}$.)
    - U_0 will construct tags $t_{0,1}$, $t_{0,2}$, $t_{0,3}$ for all messages he constructed for U_1, U_2 and U_3.
    - Each other user computes tags for the messages sent in Round I.
- `Communication`:
  - U_0 sends $(m_{0,j}, t_{0,j})$ to U_j $(1 \leqslant j \leqslant 3)$.
  - For $0 \leqslant i \leqslant 3$, U_i sends $t_{i,j}$ to U_j $(0 \leqslant j \leqslant 3,\ j \neq i)$.
- `Verification and key computation`:
  - All users verify all received tags. If all checks go through, session keys and identifiers are set as follows:
    - U_0 sets $\mathsf{ssk}_0 = [k]_L$, $\mathsf{sid}_0 := [k]_R$.
    - For $1 \leqslant i \leqslant 3$, U_i runs $\mathsf{Decaps}(sk_i, c_i)$ to obtain k_i and retrieve

$$K_i := d_i \oplus k_i.$$

    Later, keys and session identifiers are extracted as:

$$\mathsf{ssk}_i := [K_i]_L \text{ and } \mathsf{sid}_i := [K_i]_R$$

Fig. 2. Simplified description of our protocol with four users.

Theorem 1. *The protocol described in Fig. 1 achieves quantum-future key secrecy, under the Decisional Diffie Hellman Assumption in* $(G, \mathcal{D})$ *and assuming it is instantiated with a UF-CMVA-secure message authentication code and an IND-CPA-secure (post-quantum) key encapsulation mechanism.*

Proof. The proof is set up in terms of several experiments or games, where a challenger interacts with the adversary confronting it with a counterfeit Test-challenge in the spirit of the key secrecy definition from Section 2. We denote with $\mathsf{Adv}(\mathcal{A}, G_i)$ the advantage of adversary $\mathcal{A}$ in the i-th game G_i.

Game 0. This first game corresponds to a real attack, in which all the parameters are chosen as in the actual scheme. By definition, $\mathsf{Adv}(\mathcal{A}, G_0) = \mathsf{Adv}_{\mathcal{A}}$.

Game 1. This game is identical to G_0, except from the fact that it aborts with an adversarial win if $\mathcal{A}$ succeeds in producing a valid MAC for a message he has constructed (i.e. which is adversarially-generated) in Round II. There are two cases we should distinguish:

Case 1: no QCom queries called by $\mathcal{A}$ At this, we can argue that the tags $t_{i,j}$ can be replaced with bitstrings of the correct length chosen uniformly at random. Indeed, we may modify the Execute and Send oracle in such a way that all values g_i from Round I are replaced with g_i^* chosen u.a.r. from G. The games G_0 and G_1 can only be distinguished one from the other if:

- $\mathcal{A}$ correctly guesses pw and sends a properly computed value $g_i = \mathsf{pw}^{\beta_i}$ to an instance of U_j on behalf of an instance of U_i. If this is the case, $\mathcal{A}$ may correctly verify the tag $t_{j,i}$ he gets from U_j in G_0 while the verification will be unsuccessful in G_1.
- $\mathcal{A}$, not using QCom checks offline, for each password in the dictionary $\mathcal{D}$, whether the triplets $(g_i, g_j, t_{i,j})$ are all consistent for a fixed password. It is easy to see that if $\mathcal{A}$ can actually check whether such triplets $(g_i, g_j, t_{i,j})$ are consistent with a given password pw^*, the corresponding Decisional Diffie Hellman assumption in $(G, \mathcal{D})$ cannot hold. Indeed, we may construct an adversary $\mathcal{B}$ using $\mathcal{A}$ in order to solve a corresponding Decisional Diffie Hellman challenge in $(G, \mathcal{D})$. Let (g, x, y, z) be the input to $\mathcal{B}$. In order to tell whether order to distinguish whether that is a triplet of the form (g^a, g^b, g^{ab}) or z is a randomly generated element in G, $\mathcal{B}$ presents $\mathcal{A}$ with a simulated transcript (using the password encoding by g for authentication) where for a certain pair of users U_i, U_j the authenticated tags $t_{i,j}$ and $t_{j,i}$ are constructed from z.

Indeed, as the group elements g_i have been chosen uniformly at random, the MAC keys derived in Round II are also uniformly at random selected bitstrings. Thus we have,

$$|\mathsf{Adv}(\mathcal{A}, G_1) - \mathsf{Adv}(\mathcal{A}, G_0)| \leqslant \varepsilon(\ell, q) + \mathsf{Adv}_{DDH}(G, \mathcal{D}).$$

Case 2: $\mathcal{A}$ queried QCom after Execute or Send If the QCom oracle calls are only restricted as stated in the freshness definition, indeed it may be the case that the MAC keys are known to the adversary who has queried QCom; however, as he is not allowed making any Send query, he will of course not produce any valid forgery, as a result, only Case 1 is to be taken into account when bounding the distinguishing probability of $\mathcal{A}$

$$|\mathsf{Adv}(\mathcal{A}, G_1) - \mathsf{Adv}(\mathcal{A}, G_0)| \leqslant \varepsilon(\ell, q) + \mathsf{Adv}_{DDH}(G, \mathcal{D}).$$

Given that all MAC keys are at this point generated uniformly at random, this can only happen if $\mathcal{A}$ is able to produce a forgery for the MAC in use, and thus

$$\mathsf{Adv}(\mathcal{A}, G_1) = \mathsf{Adv}(\mathcal{A}, G_0) \leqslant \mathsf{Succ}_{\mathcal{A}},$$

where $\mathsf{Succ}_{\mathcal{A}}$ is the probability of an adversary winning in the UF-CMVA game described in Section 3, hence assumed to be negligible in ℓ.

Game 2. In this game the output of the Execute and Send oracles are modified as follows. For each $j \in \{1, \ldots, n\}$, the value d_j is replaced with d_j^* selected u.a.r. from $\{0,1\}^{p(\ell)}$. Then, for the computation of each tag $t_{0,j}$, $m_{0,i}$ is replaced with $m_{0,i}^* := (d_i^*, c_i)$ for $i = 1, \ldots, n$. Note that d_i is the XOR of k and the output k_i of Encaps and that k is chosen u.a.r. by U_0 and only used in the computation of the values d_i throughout a protocol run.

To argue that the only way an adversary is able to distinguish between G_1 and G_2 is breaking the IND-CPA security of the KEM, we depict how an IND-CPA adversary $\mathcal{B}$ for the KEM can be constructed from an adversary $\mathcal{A}$ who may distinguish between the two games.

Indeed, suppose we actually introduce the change in G_1 step by step. Let $\mathcal{A}$ be an adversary distinguishing between G_1 and the game resulting after step 1. Now, let $\mathcal{B}$ be presented with an IND-CPA challenge for the KEM used by user U_1, i.e. with a value (KEY, C) where KEY is either a key encapsulated in C using pk_1 or a string chosen uniformly at random. Now, fixing a session key k, fairly produce messages from Round I for each user (d_i, c_i) for $i = 1, \ldots, n$ and replace $m_{0,l}$ by $m_{0,l}^* = (KEY \oplus k, C)$, while, for the rest, follow the protocol description. Now, if KEY comes from the KEM, all messages will follow the protocol specification, while if it is selected uniformly at random indeed $KEY \oplus k$ will also be. As a result, if $\mathcal{A}$ distinguishes between G_1 and G_2 he will violate the IND-CPA security of the KEM, so indeed, replicating this argument in a step by step fashion we have

$$\left|\mathsf{Adv}(\mathcal{A}, G_2) - \mathsf{Adv}(\mathcal{A}, G_1)\right| \leqslant \mathsf{Adv}_{\mathcal{A},\mathrm{KEM}}.$$

Now note that no information correlated with the actual session key is involved in any message at this point; clearly we have $\mathsf{Adv}(\mathcal{A}, G_2) = 0$, which concludes the proof. □

Remark 4. Note that in the above proof it is crucial to restrict the QCom calls in the definition of freshness; otherwise an offline-guessing strategy can be mounted by solving the corresponding Diffie-Hellman instances $(pw, pw^{\beta_i}, pw^{\beta_j})$ for every $pw \in \mathcal{D}$, and confronting the result with the messages tagged with $t_{i,j}$, i.e. a quantum adversary may get the password from the transcript and thus we should make clear we do not allow for subsequent Send calls once QCom has been queried.

Remark 5. We stress further that for the above proof (see Game 2) it is needed to impose that the KEM in use is post-quantum, as we do not restrict the usage of QCom when trying to tell apart the "real" $d_i's$ from randomly selected ones.

Table 1
Performance of our protocol compared to recent post-quantum solutions. Here, ADGK refers to the protocol in Apon *et al.* (2019) and ADGK† is an authenticated version of ADGK, obtained by applying the Katz-Yung compiler. PSS refers to the solution in Persichetti *et al.* (2019).

	Rounds	Communication	Computation	Authentication
Here	2	n^2+n elements in G, n^2 KEM public keys, n^2 MAC tags, n KEM-encapsulated keys, n masked ephem. keys	$2(n+1)$ exp. in G, n key enc. and dec., n^2+n MAC tags	Password+ MAC
ADGK	3	$2(n^2+n)$ $R_{\hat{q}}$-elements, n^2+n elements in the output space of the key reconciliation algorithm	$2(n+1)^2$ ops. in $R_{\hat{q}}$, $2n$ key rec. calls	Unauthent.
ADGK†	4	As in ADGK, plus $n+1$ nonces and $3(n^2+n)$ signatures	As above, plus $3(n^2+n)$ sign.	PKI+ Signature
PSS	3	n KEM public keys, n KEM-encapsulated keys, n^2+n masked ephem. keys n^2+n signatures	n key enc. and dec., n sign.	PKI+ Signature

4.2. *Performance*

Let us compare the performance of the above protocol with two state-of-the-art fully post-quantum solutions for group key establishment (Apon *et al.*, 2019; Persichetti *et al.*, 2019). For clarity, we assume that the number of users is $n+1$ for every protocol. Table 1 summarizes important characteristics.

- *Our protocol*. In the first round, $n+1$ elements of G and n public keys of the KEM are broadcast, yielding a total of n^2+n elements of G and n^2 public keys to be transmitted. In the second round, n^2 tags, n KEM ciphertexts, and n masked ephemeral keys are sent.
- Apon *et al.* (2019). This scheme provides security against passive adversaries. In order to transform it into a GAKE, the authors propose using the Katz-Yung compiler (Katz and Yung, 2007), which adds one round of communication (in which each participant broadcasts a nonce), and appends one signature to every sent message that should be taken into account. In the setup, a ring $R_{\hat{q}} = \mathbb{Z}_{\hat{q}}[X]/(X^{\hat{n}}+1)$ is fixed, where $\hat{q}$ is a prime and $\hat{n}$ is a power of 2 such that $\hat{q} \equiv 1 \bmod 2\hat{n}$. In the first two rounds, each user broadcasts an element of $R_{\hat{q}}$, for a total of $2(n^2+n)$ elements. In the third round, each user runs the key recovery algorithm to get a pair $(\hat{K}, \hat{k})$ and broadcasts $\hat{K}$.
- Persichetti *et al.* (2019). This scheme is also a compiler which invokes a KEM and a signature scheme. In the first round, each user sends a KEM public key to his right neighbour, while in the second one, each user sends a KEM ciphertext to his left neighbour. Finally, in the third round each user broadcasts a masked ephemeral key and a signature.

In the table we also included a column on the authentication tools used – differing from the other mentioned protocols, we opted for password-based authentication and do not assume a PKI for signatures. In terms of performance, the reduced number of rounds in our protocol is attractive. Compared to PSS, we pay a cost for the MAC tag computations and transmissions, but, as we do not involve a PKI to handle signatures and MAC computations tend to be fast, this cost seems quite reasonable.

5. Final Remarks

We present in this work a group key exchange that can be proven secure in the sense of Definition 1. As we only consider quantum adversaries to be active once the protocol execution has ended, and, moreover, our building blocks are very simple tools, we accomplish a clean and efficient design. It is indeed worth exploring other approaches towards thwarting general quantum attacks. A promising avenue is to investigate the viability of a compiled construction derived from applying the design sketched in Appendix C of Benhamouda *et al.* (2018) and the compiler from Abdalla *et al.* (2007). While being fully quantum resistant, such a design would involve sophisticated lattice-based primitives, for which, moreover, investigating the attained post-quantum security level is still a topic of active research.

Funding

This research was funded by the NATO Science for Peace and Security Programme, grant number G5448, and by MINECO under Grant MTM2016-77213-R.

References

Abdalla, M., Bohli, J., Vasco, M.I.G., Steinwandt, R. (2007). (Password) Authenticated key establishment: from 2-party to group. In: Vadhan, S.P. (Ed.), *Theory of Cryptography, 4th Theory of Cryptography Conference*, TCC 2007, Amsterdam, The Netherlands, February 21–24, 2007, Proceedings, *Lecture Notes in Computer Science*, Vol. 4392. Springer, Amsterdam, The Netherlands, pp. 499–514.

Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P. (2016). Post-quantum key exchange – a new hope. In: Holz, T., Savage, S. (Eds.), *25th USENIX Security Symposium, USENIX Security 16*, Austin, TX, USA, August 10–12, 2016. USENIX Association, pp. 327–343.

Apon, D., Dachman-Soled, D., Gong, H., Katz, J. (2019). Constant-round group key exchange from the ring-LWE assumption. In: Ding, J., Steinwandt, R. (Eds.), *Post-Quantum Cryptography – 10th International Conference* PQCrypto 2019, Chongqing, China, May 8–10, 2019, Revised Selected Papers, *Lecture Notes in Computer Science*, Vol. 11505. Springer, pp. 189–205.

Aragon, N., Barreto, P., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.-C., Gaborit, P., Gueron, S., Guneysu, T., Melchor, C.A., Misoczki, R., Persichetti, E., Sendrier, N., Tillich, J.-P., Zémor, G. (2017). BIKE: bit flipping key encapsulation. hal-01671903.

Azarderakhsh, R., Campagna, M., Costello, C., Feo, L., Hess, B., Jalali, A., Jao, D., Koziel, B., LaMacchia, B., Longa, P. et al. (2017). SIKE – Supersingular Isogeny Key Encapsulation. https://sike.org/.

Bellare, M., Rogaway, P. (1994). Entitiy authentication and key distribution. In: Stinson, D.R. (Ed.), *Advances in Cryptology – CRYPTO '93*, *Lecture Notes in Computer Science*, Vol. 773. Springer, pp. 232–249.

Bellare, M., Pointcheval, D., Rogaway, P. (2000). Authenticated key exchange secure against dictionary attacks. In: Preneel, B. (Ed.), *Advances in Cryptology – EUROCRYPT 2000*, *Lecture Notes in Computer Science*, Vol. 1807. Springer, pp. 139–155.

Benhamouda, F., Blazy, O., Ducas, L., Quach, W. (2018). Hash proof systems over lattices revisited. In: Abdalla, M., Dahab, R. (Eds.), *Public-Key Cryptography – PKC 2018 – 21st IACR International Conference on Practice and Theory of Public-Key Cryptography*. Rio de Janeiro, Brazil, March 25–29, 2018, Proceedings, Part II, *Lecture Notes in Computer Science*, Vol. 10770. Springer, pp. 644–674.

Bernstein, D.J. (2005). The Poly1305-AES message-authentication code. In: Gilbert, H., Handschuh, H. (Eds.), *Fast Software Encryption FSE 2005, Revised Selected Papers*, *Lecture Notes in Computer Science*, Vol. 3557. Springer, pp. 32–49.

Bernstein, D., Chou, T., Lange, T., von Maurich I, Misoczki, R., Niederhagen, R., Persichetti, E., Peters, C., Schwabe, P., Sendrier, N., Szefer, J., Wang, W. (2017a). Classic McEliece. https://classic.mceliece.org/.

Bernstein, D.J., Chuengsatiansup, C., Lange, T., van Vredendaal, C. (2017b). Reducing attack surface at low cost. In: Adams, C., Camenisch, J., (Eds.), Selected Areas in Cryptography – SAC 2017 – 24th International Conference, Ottawa, ON, Canada, August 16–18, 2017, Revised Selected Papers, *Lecture Notes in Computer Science*, Vol. 10719. Springer, pp. 235–260.

Bindel, N., Herath, U., McKague, M., Stebila, D. (2017). Transitioning to a quantum-resistant public key infrastructure. In: Lange, T., Takagi, T. (Eds.), *Post-Quantum Cryptography – 8th International Workshop*, PQCrypto 2017, Utrecht, The Netherlands, June 26–28, 2017, Proceedings, *Lecture Notes in Computer Science*, Vol. 10346. Springer, pp. 384–405.

Bindel, N., Brendel, J., Fischlin, M., Goncalves, B., Stebila, D. (2018). Hybrid key encapsulation mechanisms and authenticated key exchange. *IACR Cryptology ePrint Archive*, 2018, 903.

Bohli, J.M., González Vasco, M.I., Steinwandt, R. (2007). Secure group key establishment revisited. *International Journal of Information Security*, 6(4), 243–254.

Boneh, D., Glass, D., Krashen, D., Lauter, K.E., Sharif, S., Silverberg, A., Tibouchi, M., Zhandry, M. (2018). Multiparty non-interactive key exchange and more from isogenies on elliptic curves. CoRR. abs/1807.03038.

Bos, J.W., Costello, C., Naehrig, M., Stebila, D. (2015). Post-quantum key exchange for the TLS protocol from the ring learning with errors problem. In: *2015 IEEE Symposium on Security and Privacy*, SP 2015, San Jose, CA, USA, May 17–21, 2015. IEEE Computer Society, pp. 553–570.

Bos, J.W., Costello, C., Ducas, L., Mironov, I., Naehrig, M., Nikolaenko, V., Raghunathan, A., Stebila, D. (2016). Frodo: take off the ring! Practical, quantum-secure key exchange from LWE. In: Weippl, E.R., Katzenbeisser, S., Kruegel, C., Myers, A.C., Halevi, S. (Eds.), *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, Vienna, Austria, October 24–28, 2016. ACM, pp. 1006–1018.

Bos, J.W., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Seiler, G., Stehlé, D. (2018). CRYSTALS – kyber: A CCA-secure module-lattice-based KEM. In: *2018 IEEE European Symposium on Security and Privacy*, EuroS&P 2018, London, United Kingdom, April, 24–26, 2018. IEEE, pp. 353–367.

Bresson, E., Chevassut, O., Pointcheval, D., Quisquater, J. (2001). Provably authenticated group Diffie-Hellman key exchange. In: Reiter, M.K., Samarati, P. (Eds.), CCS 2001, Proceedings of the 8th ACM Conference on Computer and Communications Security, Philadelphia, Pennsylvania, USA, November 6–8, 2001. ACM, pp. 255–264.

Burmester, M., Desmedt, Y. (2005). A secure and scalable Group Key Exchange system. *Information Processing Letters*, 94(3), 137–143.

Chevassut, O., Fouque, P., Gaudry, P., Pointcheval, D. (2006). The twist-augmented technique for key exchange. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (Eds.), *Public Key Cryptography – PKC 2006 Proceedings*. *Lecture Notes in Computer Science*, Vol. 3958. Springer, pp. 410–426.

Ding, J., Xie, X., Lin, X. (2012). A simple provably secure key exchange scheme based on the learning with errors problem. *IACR Cryptology ePrint Archive*, 2012, 688.

Ding, J., Gao, X., Takagi, T., Wang, Y. (2019a). One sample ring-LWE with rounding and its application to key exchange. In: Deng, R.H., Gauthier-Umaña, V., Ochoa, M., Yung, M. (Eds.), *Applied Cryptography and Network Security – 17th International Conference*, ACNS 2019, Bogota, Colombia, June 5–7, 2019, Proceedings, *Lecture Notes in Computer Science*, Vol. 11464. Springer, pp. 323–343.

Ding, J., Schmitt, K., Zhang, Z. (2019b). A key exchange based on the short integer solution problem and the learning with errors problem. In: Carlet, C., Guilley, S., Nitaj, A., Souidi, E.M. (Eds.), *Codes, Cryptology and Information Security – Third International Conference*, C2S 2019, Rabat, Morocco, April 22–24, 2019, Proceedings – In Honor of Said El Hajji, *Lecture Notes in Computer Science*, Vol. 11445. Springer, pp. 105–117.

Dodis, Y., Kiltz, E., Pietrzak, K., Wichs, D. (2012). Message authentication. In: Pointcheval, D., Johansson, T. (Eds.), *Advances in Cryptology – EUROCRYPT 2012 – 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Cambridge, UK, April 15–19, 2012, Proceedings, *Lecture Notes in Computer Science*, Vol. 7237. Springer, pp. 355–374.

Katz, J., Yung, M. (2007). Scalable protocols for authenticated group key exchange. *Journal of Cryptology*, 20(1), 85–113.

Katz, J., Vaikuntanathan, V. (2009). Smooth projective hashing and password-based authenticated key exchange from lattices. In: Matsui, M. (Ed.), *Advances in Cryptology – ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security*, Tokyo, Japan, December 6–10, 2009, Proceedings, *Lecture Notes in Computer Science*, Vol. 5912. Springer, pp. 636–652.

National Institute of Standards and Technology (2019). Post-Quantum Cryptography; Round 2 Submissions. https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions.

Persichetti, E., Steinwandt, R., Suárez Corona, A. (2019). From key encapsulation to authenticated group key establishment – a compiler for post-quantum primitives. *Entropy*, 21(12), 1183.

M.I. González Vasco is associate professor at MACIMTE, Universidad Rey Juan Carlos, where she works since 2003. She received her diploma and PhD degree in mathematics from the Universidad de Oviedo (1999 and 2003). Her research interests include provable security for cryptographic constructions, with special focus on public key cryptographic designs for encryption and group key exchange. She is currently a member of the Board of Directors (*Junta de Gobierno*) of the Royal Spanish Mathematical Society.

A.L. Pérez del Pozo is assistant professor (*profesor ayudante doctor*) at the Universidad Rey Juan Carlos, Spain. He holds a PhD in Mathematics from Universidad Complutense de Madrid (Spain). His main research focus is cryptographic designs for key exchange in non-standard scenarios, secret sharing schemes, and applications of multi-party computation.

R. Steinwandt serves as Chair of Florida Atlantic University's Department of Mathematical Sciences. Before joining FAU, he was with the University of Karlsruhe in Germany, where he completed his MS and PhD degrees in computer science, researching topics in computer algebra. Today, his research focus is in cryptology, including quantum cryptanalysis and quantum-safe cryptography. He currently serves as director of FAU's Center for Cryptology and Information Security. His research has been funded through the Air Force Research Laboratory, the German Federal Office for Information Security, the National Science Foundation, and the NATO Science for Peace and Security program.