Unconditionally Secure Signatures

Ryan Amiri[1], Aysajan Abidin[2], Petros Wallden[3], Erika Andersson[1]

[1]SUPA, Institute of Photonics and Quantum Sciences, Heriot-Watt University, Edinburgh EH14 4AS, United Kingdom
[2]ESAT/COSIC, KU Leuven and iMinds, Belgium
[3]LFCS, School of Informatics, University of Edinburgh, 10 Crichton Street, Edinburgh EH8 9AB, United Kingdom
ra2@hw.ac.uk; aysajan.abidin@esat.kuleuven.be; petrol.wallden@gmail.com; e.andersson@hw.ac.uk

Abstract. Digital signatures are one of the most important cryptographic primitives. In this work we construct an information-theoretically secure signature scheme which, unlike prior schemes, enjoys a number of advantageous properties such as short signature length and high generation efficiency, to name two. In particular, we extend symmetric-key message authentication codes (MACs) based on universal hashing to make them transferable, a property absent from traditional MAC schemes. Our main results are summarised as follows.

- We construct an unconditionally secure signature scheme which, unlike prior schemes, does not rely on a trusted third party or anonymous channels. In our scheme, a sender shares with each of the remaining protocol participants (or recipients) a set of keys (or hash functions) from a family of universal hash functions. Also, the recipients share with each other a random portion of the keys that they share with the sender. A signature for a message is a vector of tags generated by applying the hash functions to the message. As such, our scheme can be viewed as an extension of MAC schemes, and therefore, the practical implementation of our scheme is straightforward.
- We prove information-theoretic security of our scheme against forging, repudiation, and non-transferability.
- We compare our schemes with existing both "classical" (not employing quantum mechanics) and quantum unconditionally secure signature schemes. The comparison shows that our new scheme has a number of unparalleled advantages over the previous schemes.
- Finally, although our scheme does not rely on trusted third parties, we discuss this, showing that having a trusted third party makes our scheme even more attractive.

Key words: Digital signatures, information-theoretic security, transferable MAC, Universal hashing.

1 Introduction

Digital signatures are one of the most widely used cryptographic primitives and are indispensable for information and communications security. Secure digital signature schemes offer authenticity and integrity, non-repudiation, and transferability of digital content. However, the public-key digital signature schemes that are currently in use, such as RSA [1], ElGamal DSA [2] and ECDSA [3], provide only computational security, and rely on unproven hardness assumptions in number theory. This implies that algorithmic breakthrough and/or the advancement in computing technologies may one day render such digital signature schemes totally insecure. Another emerging threat to the security of these schemes is from quantum computers, which can use Shor's algorithm [4] to efficiently solve the underlying "hard" problems and break all pre-quantum public-key cryptosystems. In response to this threat, the field of post-quantum cryptography is being developed. One can argue and ask whether quantum computers will ever be built. However, the National Security Agency (NSA) in the USA is taking the threat from quantum computers very seriously, and in August 2015, the NSA recommended a transition to post-quantum secure algorithms [5].

In post-quantum cryptography, there exist "quantum-safe" public-key cryptosystems which are not *yet* known to be vulnerable to quantum attacks. Such schemes range from the historical McEliece cryptosystem [6], which is based on error-correcting codes, to more recent ones based on hash functions, lattices and multivariate polynomials. The security of these "quantum-safe" alternatives is based upon (again unproven) hard problems, some of which have not yet stood the test of time[1]. We stress again that even if the underlying problems were proven to be hard to solve, the security of such schemes is still only computational, and relies on the adversary having bounded computational resources. If we want signature schemes with "everlasting" security, or if we are unsure of the resources available to our adversary, computational security may not be sufficient.

An alternative to "quantum-safe" public key signature schemes are unconditionally secure signature (USS) schemes, where security does not rely on any unproven assumptions, nor on bounds placed on the adversary's computational resources. Such a high level of security, however, comes at a cost. So far, all unconditionally secure signature schemes have

[1] In lattice-based cryptography [7] for example, it is not quite clear anymore whether all such protocols are truly quantum resistant [8, 9]

been significantly less efficient than their "quantum-safe" competitors in terms of signature length, re-usability and key sizes. A more restrictive disadvantage however, is that all unconditionally secure schemes use secret keys, rather than public keys.

USS schemes require a setup phase in which secret keys are distributed among participants before messages can be signed or verified. Therefore, they do not have the universal verifiability property inherent to standard public-key digital signature schemes. Due to this restriction, it is clear that USS schemes are not a suitable replacement for many core applications wheredigital signatures are used. Nevertheless, there may still be applications where unconditionally secure signatures may be useful, for particularly important communications, for example in high value banking transactions, when signing important legal documents, or securing sensitive government communications. Due to the requirement of distributing secret shared keys between participants, unconditionally secure signatures should not be viewed as a standalone product. Instead, it could be viewed as a complement to existing QKD systems in fixed networks environments.

In this work, we focus on unconditionally secure, also known as information-theoretically secure, signature schemes. In particular, we propose a new USS scheme based on universal hashing. Compared to the previous USS schemes in the literature, our scheme enjoys a number of favourable properties such as short secret key lengths, short signature length, and high efficiency. Before we proceed, we first briefly survey the USS schemes which are already proposed in the literature. For a detailed overview, we refer the interested reader to [10] and the references therein.

1.1 Related works

There are two lines of work on USS schemes: one on "classical" schemes (not employing quantum mechanics), and the other taking advantage of quantum-mechanical features. Although our scheme is entirely classical, it is similar to the quantum signature scheme proposed in [11].

Classical unconditionally secure signature schemes The first attempt to construct an USS scheme was suggested by Chaum and Roijakkers [12], using authenticated broadcast channels, secret authenticated channels and also utilising untraceable sending protocols. Their scheme, however, only allows signing single-bit messages, and is therefore impractical. Moreover, the Chaum-Roijakkers scheme does not offer adequate

transferability, which is crucial for a signature scheme, because the security is weakened as the message-signature pair is transferred among recipients. Pfitzmann and Waidner [13] also considered USS schemes (which they called pseudo-signatures) and constructed a scheme, somewhat related to ours, which could be used to generate information-theoretically secure Byzantine agreement. Their scheme built upon the protocol by Chaum and Roijakkers, but allowed longer messages to be signed and verified, though the scheme still required authenticated broadcast channels and untraceable sending protocols for implementation. Our scheme removes the requirement of authenticated broadcast channels by employing a method similar to secret sharing techniques [14].

Later, Hanaoka *et al.* [15] proposed an USS scheme relying on a trusted authority for key distribution, the existence of which allowed improvements both in efficiency and security over the scheme by Chaum and Roijakkers, at the cost of introducing this additional trust assumption. This scheme further improved all existing USS protocols by making the signature scheme re-usable. Nevertheless, the length of both the signature and the secret keys needed to generate signing/verification algorithms were still rather long, severely limiting its use in practice. A later variation of this scheme was proposed by Hanaoka *et al.* in [16]. This scheme sacrificed the re-usability of the previous scheme to achieve a reduction in the size of the secret keys needed to generate signing/verification algorithms by approximately a factor of 10.

Security notions of USS schemes are proposed and analysed in Shikata *et al.* [17] as well as Swanson and Stinson [18].

Quantum signature schemes There are also quantum signature schemes, first proposed by Gottesman and Chuang [19]. Here, security is derived from the laws of quantum physics. Lu and Feng [20] proposed a quantum signature scheme using quantum one-way functions, though it required a trusted authority (which they called an arbiter) to resolve disputes. Quantum signature schemes were first experimentally demonstrated by Clarke *et al.* [21]. While these early quantum schemes require long-term quantum memories (which are highly impractical to realise, effectively rendering these schemes unusable), the more recently proposed schemes do not require quantum memories [22, 11, 23]. Quantum digital signature schemes without quantum memories have also been experimentally demonstrated [24, 25]. Furthermore, these recent schemes and their experimental demonstrations uses the already ripe technologies required for quantum key distribution [26].

1.2 Contributions

In this work, we propose an USS scheme which naturally extends unconditionally secure message authentication schemes. The main difference between an unconditionally secure message authentication code and an unconditionally secure signature scheme is that signature schemes ensure the transferability of signed content, while authentication codes do not. We propose a simple method, similar to secret sharing [14], allowing unconditionally secure authentication codes to be transformed into USS schemes. Our method requires only minimal trust assumptions and fewer resources than in previous USS schemes. We do not assume a trusted authority, nor the existence of authenticated broadcast channels. Instead, we only require participants to share short secret keys pairwise, and that the majority of participants are honest. Our contributions can be summarised as follows.

- We construct an USS scheme that, unlike prior schemes, does not rely on a trusted authority or anonymous channels (Section 3). In our scheme, a sender shares with each of the remaining protocol participants (or recipients) a set of keys (or hash functions) from a family of universal hash functions. The recipients then share with each other a random portion of the keys that they received from the sender. A signature for a message is a vector of tags generated by applying the hash functions to the message. As such, our scheme can be viewed as an extension of MAC schemes, and therefore, the practical implementation of our scheme is straightforward and efficient.
- We prove information-theoretic security of our scheme against forging, repudiation, and non-transferability (Section 4).
- Although our scheme does not rely on trusted third parties, we show that having a trusted authority makes our scheme even more attractive (Section 5). In addition, we discuss possible extensions to our scheme.
- In Section 6 we compare our schemes with existing both classical and quantum USS schemes. The comparison shows that our new scheme has a number of unparalleled advantages over the previous schemes.

2 Preliminaries

We begin by following [27] and formally defining an unconditionally secure signature scheme.

Definition 1 *An USS protocol $\mathcal{Q}$ is an ordered set $\{\mathcal{P}, \mathcal{M}, \Sigma, L, \text{Gen}, \text{Sign}, \text{Ver}\}$ where*

- *The set $\mathcal{P} = \{P_0, P_1, ..., P_N\}$, is the set containing the signer, P_0, and the N potential receivers.*
- *$\mathcal{M}$ is the set of possible messages and Σ is the set of possible signatures.*
- Gen *is the generation algorithm that gives rise to the functions* Sign *and* Ver *that are used to generate a signature and verify its validity. More precisely, the generation algorithm specifies the instructions for the communication that takes place in the distribution stage of the protocol. Additionally, the generation algorithm instructs how to construct the functions* Sign *and* Ver *based on the data obtained during the distribution stage. The generation algorithm includes the option of outputting an instruction to abort the protocol.*
- $\text{Sign} : \mathcal{M} \to \Sigma$ *is a deterministic function that takes a message m and outputs a signature $\sigma \in \Sigma$.*
- *$L = \{-1, 0, 1, ..., l_{max}\}$ is the set of possible verification levels of a signed message. A verification level l corresponds to the minimum number of times that a signed message can be transferred sequentially to other recipients. For a given protocol, the maximum number of sequential transfers that can be guaranteed is denoted by $l_{max} \leq N$.*
- $\text{Ver} : \mathcal{M} \times \Sigma \times \mathcal{P} \times L \to \{\text{True}, \text{False}\}$ *is a deterministic function that takes a message m, a signature σ, a participant P_i and a level l, and gives a truth value depending on whether participant P_i accepts the signature as valid at the verification level l.*

Definition 2 *For a fixed participant, P_i, at a fixed verification level, l, we denote the verification function as $\text{Ver}_{i,l}(m, \sigma) := \text{Ver}(m, \sigma, i, l)$.*

Definition 3 *An USS protocol $\mathcal{Q}$ is correct if $\text{Ver}_{i,l}(m, \text{Sign}(m)) = \text{True}$ for all $m \in \mathcal{M}$, $i \in \{1, ..., N\}$, and $l \in L$.*

In Section 4 we will define what it means for an USS protocol to be secure.

The signature protocol presented in this paper utilises almost strongly universal hash function families, originally introduced in [28].

Definition 4 *Let $\mathcal{F} = \{f : \mathcal{M} \to \mathcal{T}\}$ be a set of functions such that*

1. *For any $m \in \mathcal{M}$, $t \in \mathcal{T}$, $|\{f \in \mathcal{F} : f(m) = t\}| = |\mathcal{F}|/|\mathcal{T}|$.*
2. *For any $m_1, m_2 \in \mathcal{M}$, $t_1, t_2 \in \mathcal{T}$, such that $m_1 \neq m_2$, $|\{f \in \mathcal{F} : f(m_1) = t_1 \text{ and } f(m_2) = t_2\}| \leq \epsilon \frac{|\mathcal{F}|}{|\mathcal{T}|}$.*

Then we say $\mathcal{F}$ is ϵ-ASU_2. The domain of each function in $\mathcal{F}$ is the message set, $\mathcal{M}$, and the range is the set of tags, $\mathcal{T}$.

The efficiency of our protocol relies on ability to find an ϵ-ASU_2 set which is "small".

Proposition 1 *Let $a := \log|\mathcal{M}|$ and $b := \log|\mathcal{T}|$, be the size (in bits) of the message and tag respectively[2]. Let $\mathcal{F}$ be an ϵ-ASU_2 set with $\epsilon = 2/|\mathcal{T}|$. It is possible to specify an element of $\mathcal{F}$ using y bits of data, where*

$$y = 3b + 2s \tag{1}$$

and s is such that $a = (b + s)(1 + 2^s)$.

Proof. See [29]. □

3 The Protocol

The following protocol is inspired by protocol P2, first introduced in [11] and subsequently generalised to the N-party case in [27]. The protocol contains $N + 1$ participants: a sender P_0 and N receivers, $P_1, ..., P_N$. Before the protocol, all participants agree on an ϵ-ASU_2 family of functions, $\mathcal{F}$, where $\epsilon = 2/|\mathcal{T}|$. The basic idea is for the sender to give each recipient a number of keys (hash functions) which will be used in future to authenticate a message by appending tags (hash values) to the message being sent. To check the signature, participants will apply their hash functions to the message, and check that the outcome matches the tags appended to the message by the sender. They will count the number of mismatches between their hash values and the appended tags, and only accept the message if they find less than a threshold amount of mismatches. However, if the sender knows which hash functions are held by which participant, he could choose to append appropriate tags such that one recipient accepts the message while another does not, thereby breaking transferability of the scheme. To ensure transferability then, each recipient will group the hash functions received from the sender into N equally sized sets (of size k), and send a set (over secret channels) to each recipient, keeping one for himself. The situation is further complicated if the sender is in collusion with some of the recipients. In that case, the sender can have partial knowledge on who holds which keys, which forces us to define levels of transferability.

[2] In this paper all logarithms are taken to base 2.

Levels of transferability are perhaps confusing, so here we will try to highlight the need for such levels. Imagine that a sender is in collusion with a single recipient. In this case, the sender knows k of the keys held by honest recipient H_1, and k of the keys held by honest recipient H_2 - namely he knows the keys that were forwarded by his dishonest partner. The sender can attach tags that are correct for H_1, and are incorrect for H_2. Therefore, based on the number of colluding adversaries, the sender is able to bias the number of mismatches found between each honest party. To ensure transferability then, we require that the second verifier accepts a message as authentic even if it contains a higher number of mismatches than the first. Of course, to ensure security against forging, we cannot allow messages-signature pairs containing too many errors to be accepted, and so there must be a cap on the highest level of mismatches acceptable by anyone. This leads to levels of verification, and a limit on the number of times a message is guaranteed to be transferable in sequence. For clarity, suppose then there are three levels of verification, l_0, l_1 and l_2. Accepting a message at any of these levels means the message is guaranteed to have originated with the claimed sender. If H_1 accepts a message at level l_2 (the highest verification level, i.e. the level with the fewest errors in the signature), then he can forward it to H_2, who will first try to accept the message at level l_2. If he finds too many mismatches for the message to be accepted at level l_2, he will instead try to verify at level l_1. The protocol ensures that if H_1 found the message to be valid at level l_2, then H_2 will find the message to be valid at level l_1 with high probability. Therefore, with three verification levels, accepting the message at level l_2 guarantees that the message can be transferred at least twice more. In practice, the message may be transferred many more times, since with honest participants it is highly likely that H_2 will also find the message valid at level l_2 and they will not need to move to the next verification level.

With this in mind, to begin the protocol we must first decide the maximum fraction of dishonest participants we want our protocol to be able to tolerate (which, as per the proceeding paragraph, will impact our verification levels). We set this to be $d_c < 1/2$, since if $d_c \geq 1/2$ the protocol cannot be made secure using the majority vote dispute resolution process (see Definition 6 below). Under this assumption we know that the number of honest participants is at least n_h, with $n_h = N(1 - d_c)$. As in previous protocols, there are two stages - the distribution stage and the messaging stage.

3.1 Distribution Stage

1. The sender uniformly and randomly selects (with replacement) N^2k functions from the set $\mathcal{F}$, where k is a security parameter. We denote these functions by $(f_1, ..., f_{N^2k})$ and will refer to them as the *signature functions*.
2. To each recipient, P_i, the sender uses secret classical channels to transmit the functions $(f_{(i-1)Nk+1}, ..., f_{iNk})$. This requires the sender to share Nky secret bits with each recipient.
3. Each recipient P_i randomly splits the set $\{(i-1)Nk+1, ..., iNk\}$ into N disjoint subsets of size k, which we denote $R_{i\to 1}, ..., R_{i\to N}$. He then uses the secret classical channels to send $R_{i\to j}$ and $F_{i\to j} := \{f_r : r \in R_{i\to j}\}$ to recipient P_j. To securely transmit the signature functions and their positions requires each pair of participants to share $ky + k\log(Nk)$ secret bits. Following this symmetrisation, participant P_i holds the Nk functions given by $F_i := \bigcup_{j=1}^{N} F_{j\to i}$ and their positions given by $R_i := \bigcup_{j=1}^{N} R_{j\to i}$. We refer to these as the *key functions* and *function positions* of participant P_i. The participants will use these to check a future signature declaration.

3.2 Messaging Stage

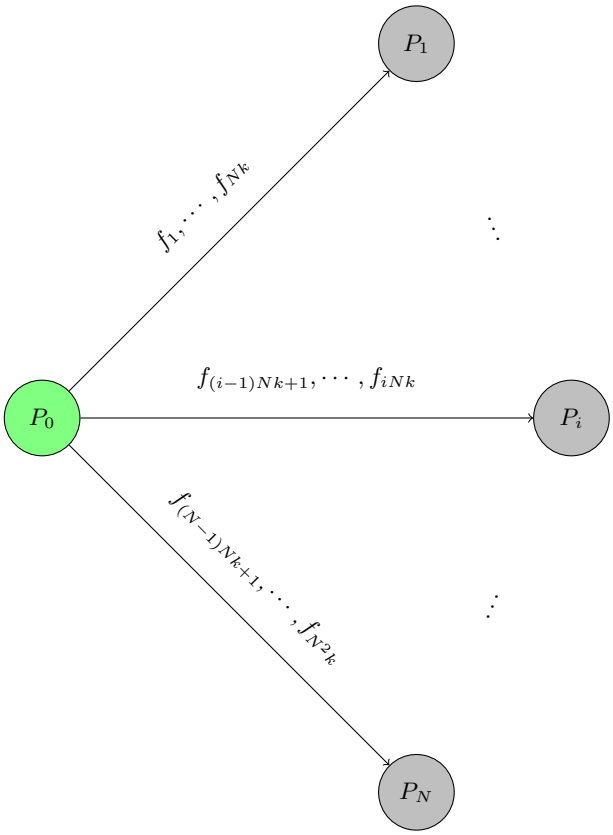
1. To send message $m \in \mathcal{M}$ to P_i, the sender sends (m, Sig_m), where

$$\mathsf{Sig}_m := (f_1(m), f_2(m), \ldots, f_{N^2k}(m)) = (t_1, \ldots, t_{N^2k}).$$

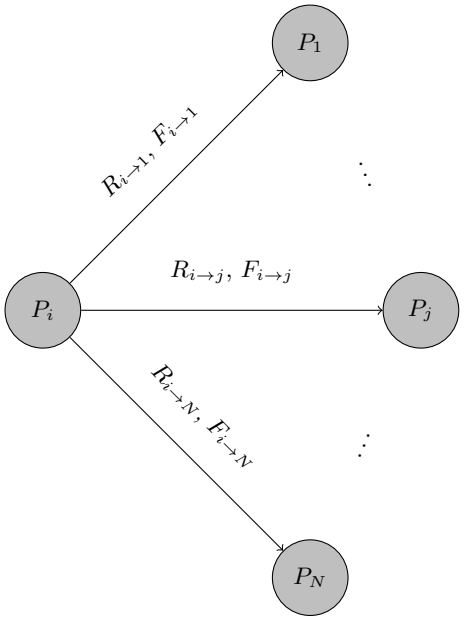
   Since the tags have size b, the signature is N^2kb bits in size.
2. For message m and the signature elements t_r such that $r \in R_{j\to i}$, participant P_i defines the following test

$$T^m_{i,j,l} = \begin{cases} 1 & \text{if } \sum_{r\in R_{j\to i}} g(t_r, f_r(m)) < s_l k \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

   where s_l is a fraction defined by the protocol, such that $1/2 > s_{-1} > s_0 > ... > s_{l_{max}}$, and $g(.,.)$ is a function of two inputs which returns 0 if the inputs are equal, and 1 if the inputs are different. For each fixed l, if the outcome of the test is 1, we say that the test is passed at level l. Essentially, this test checks whether the signature matches what the recipient expects to receive, but allows for a certain number, $s_l k$, of errors. For each verification level, all participants will perform N such tests, one for each $j = 1, ..., N$. Note that participant P_i knows all signature functions $f_{i'}$ with $i' \in R_i$ and so can perform all tests *without* interaction with any other participant.



(a) Step 2 of the distribution stage - sender key distribution



(b) Step 2 of the distribution stage - recipient key exchange

Fig. 1: In the distribution stage, (a) the sender P_0 shares distinct sets of keys with all the receivers P_i, $i = 1, \cdots, N$, and (b) the receivers exchange a randomly selected portions of their keys with each other.

3. Participant P_i will accept (m, Sig_m) as valid at level l if

$$\sum_{j=1}^{N} T_{i,j,l}^{m} > N\delta_l \tag{3}$$

That is, participant P_i accepts the signature at level l if more than a fraction of δ_l of the tests are passed, where δ_l is a threshold given by $\delta_l = 1/2 + (l+1)d_c$. Therefore, we see that each participant can accept/reject a message without interacting with any other participant in the messaging stage.
4. To forward a message, participant P_i simply forwards on (m, Sig_m) to the desired recipient.

4 Security

It is clear from the definition of the tests in Eq. (2) that the protocol works correctly if everyone is honest. In Sections 4.2, 4.3 and 4.4 we further prove that the protocol is secure against forging, non-transferability and repudiation. Before proving security of the protocol we first formally define what it means for a protocol to be secure. Again, we follow [27] in making these definitions.

4.1 Security Definitions

Definition 5 *A signature σ on a message m is i-acceptable if* $\mathrm{Ver}_{i,0}(m, \sigma) =$ True.

The meaning of this definition is that participant P_i will accept (m, σ) as a valid message signature pair at the lowest verification level, $l = 0$.

Definition 6 *When the validity of a message-signature pair (m, σ) is in dispute, we invoke a majority vote dispute resolution method* $\mathrm{MV}(m, \sigma)$*, defined by the following rule:*

1. $\mathrm{MV}(m, \sigma) =$ Valid *if* $\mathrm{Ver}_{(i,-1)}(m, \sigma) =$ True *for more than half of the users.*
2. $\mathrm{MV}(m, \sigma) =$ Invalid *otherwise*

Where $\mathrm{Ver}_{(i,-1)}(m, \sigma)$ *is the verification function at level* $l = -1$.

The $l = -1$ verification level is only used in dispute resolution, and not in normal runs of the protocol. The dispute resolution process is expensive, as it requires all participants to communicate to decide whether the

message is valid or not. This process would not be part of the regular protocol, and would only arise if participants lied about the outcome of their tests in step 3 of the messaging stage. It is expected that even dishonest participants would not lie, since forcing the expensive dispute resolution process would presumably come with consequences, and the procedure ensures the honest participants prevail as long as they are in the majority (which we assume that they are). Dispute resolution should be thought of as akin to taking legal action; in the vast majority of cases it does not happen, but its existence is necessary to prevent dishonesty. It is reasonable to assume that there would be significant disadvantages associated with forcing the dispute resolution procedure, and so parties would normally be discouraged from pursuing this route.

Signature schemes must be secure against three types of security threat – forging, repudiation and non-transferability.

Definition 7 (Forging) *Let $\mathcal{Q}$ be an USS protocol and let $C \subset \mathcal{P}$ be a coalition of malevolent parties, not including the signer P_0. Suppose that the coalition holds any valid message-signature pair (m, σ) and can use this to output a message-signature pair (m', σ') with $m' \neq m$. We define* Forging *to be the function:*

$$Forg_C(\mathcal{Q}, m', \sigma') = \begin{cases} 1 & \text{if } (m', \sigma') \text{ is } i\text{-acceptable for some } P_i \notin C \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Definition 8 (Non-Transferability) *Let $\mathcal{Q}$ be an USS protocol and $C \subset \mathcal{P}$ a coalition of malevolent participants including the signer P_0. Suppose that C outputs a message-signature pair (m, σ) and a verification level l. We define* Non-Transferability *to be the function:*

$$NonTrans_C(\mathcal{Q}, m, \sigma, l) = \begin{cases} 1 & \text{if } \mathrm{Ver}_{(i,l)}(m, \sigma) = \text{True for some } P_i \notin C \text{ and} \\ & \mathrm{Ver}_{(j,l')}(m, \sigma) = \text{False for some } 0 \leq l' < l \\ & \text{and some } j \neq i, P_j \notin C \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

Definition 9 (Repudiation) *Let $\mathcal{Q}$ be an USS protocol and $C \subset \mathcal{P}$ a coalition of malevolent participants including the signer P_0. Suppose that C outputs a message-signature pair (m, σ) and a verification level l. We*

define Repudiation *to be the function:*

$$Rep_C(\mathcal{Q}, \mathrm{MV}, m, \sigma) = \begin{cases} 1 & \textit{if } (m,\sigma) \textit{ is } i\textit{-acceptable for some } P_i \notin C \textit{ and} \\ & \mathrm{MV}(m,\sigma) = \text{Invalid} \\ 0 & \textit{otherwise} \end{cases} \tag{6}$$

We say that the protocol is secure against forging/non-transferability/repudiation if the probability of a dishonest coalition being successful decays exponentially fast with respect to some security parameter.

4.2 Forging

In order to forge, a coalition, C, (which does not include the signer) must output a message-signature pair (m, Sig_m) that will be accepted (at any level $l \geq 0$) by a participant $P_i \notin C$. We consider forging successful if the coalition can deceive any (i.e. at least one) honest participant.

Theorem 1. *The protocol defined in Section 3 is secure against forging attempts.*

Proof. It is easiest for the coalition to forge a message at the lowest verification level $l = 0$, so we consider this case in what follows. We further assume that the coalition hold a valid message-signature pair (m, Sig_m). We first restrict our attention to the coalition trying to deceive a fixed participant, and we will prove that this probability decays exponentially fast with the parameter k. We can then use this to bound the general case where the target is not a fixed participant. Therefore, for now, we fix the recipient that the coalition wants to deceive to be $P_i \notin C$. To successfully forge, the coalition should output a message signature pair, $(m', \mathrm{Sig}_{m'})$, that passes at least $N\delta_0$ of the N tests that P_i performs in step 2 of the messaging stage, where $\delta_0 = 1/2 + d_c$ and $m' \neq m$. By the definition of the protocol, the number of members in a coalition is at most Nd_c. The coalition knows $F_{j\to i}$, $R_{j\to i}$ for all $P_j \in C$, so they can use this knowledge to trivially ensure P_i passes Nd_c of the N tests performed at level $l = 0$. To pass the required $N\delta_0$ tests, the coalition must pass a further $N/2$ tests out of the $N(1 - d_c)$ remaining tests. The first step in computing the probability that they can do this is to calculate the probability of passing a single test for $P_j \notin C$.

Let p_t denote the probability that the coalition can force $T^{m'}_{i,j,0} = 1$, when they have no access to $(F_{j\to i}, R_{j\to i})$, i.e. p_t is the probability that the

coalition can create a message-signature pair that will pass the test performed by P_i for the functions received from $P_j \notin C$. As per the protocol, P_j sent $(F_{j\to i}, R_{j\to i})$ to P_i using secure channels and therefore $F_{j\to i}, R_{j\to i}$ are unknown to the coalition. However, we assume the coalition possess a valid message-signature pair (m, Sig_m), from which they can gain partial information on $(F_{j\to i}, R_{j\to i})$. Let us denote the k unknown functions in $F_{j\to i}$ by $u_1, ..., u_k$, and consider how the coalition might try to guess the value of $t_1' := u_1(m')$, given $t_1 := u_1(m)$, where $m' \neq m$. Since $\mathcal{F}$ is ϵ-ASU$_2$, using Definition 4 the coalition immediately knows u_1 is in a set $\mathcal{F}_1 \subset \mathcal{F}$ which has size $|\mathcal{F}|/|\mathcal{T}|$. Upon receiving message m', P_i will be expecting to find tag t_1' in the signature. The coalition does not know t_1' though, so the best they can do is to pick a random function in $\mathcal{F}_1$, and hope that this function also maps m' to the unknown t_1'. Again by Definition 4, the fraction of functions in $\mathcal{F}_1$ that map m' to t_1' is at most $2/|\mathcal{T}|$. Therefore, the probability that the coalition chooses a function that gives the correct tag for message m' is $2/|\mathcal{T}|$. This is true for each of the k unknown functions independently. Let X be the random variable that counts how many incorrect tags the coalition declares. Then X follows a binomial distribution and we have

$$p_t = \mathbb{P}(X < ks_0) = \sum_{v=0}^{ks_0-1} \binom{k}{v} \left(\frac{2}{|\mathcal{T}|}\right)^{k-v} \left(1 - \frac{2}{|\mathcal{T}|}\right)^{v} \tag{7}$$

This decays exponentially fast with the parameter k. For example, it may be desirable to choose a small tag length in order to minimise the length of the signature. For $|\mathcal{T}| = 4$ the signature is $2N^2k$ bits in size and we have

$$p_t = \sum_{v=0}^{ks_0-1} \binom{k}{v} \left(\frac{1}{2}\right)^{k} \approx 2^{-k(1-H_2(s_0))} \tag{8}$$

Where H_2 is the binary entropy function. Obviously, choosing a larger tag size will increase security against forging.

We will now give an upper bound for the probability of forging against a fixed participant. We start by computing the probability of passing at least one of the unknown $N(1-d_c)$ tests, which is given by

$$P(\text{FixedForge}) \leq 1 - (1-p_t)^{N(1-d_c)} \approx N(1-d_c)p_t \tag{9}$$

where we have used the fact that $p_t \ll 1$ in the approximation.

The total number of honest participants is $N(1-d_c)$ and for successful forging we only require that any one of them is deceived. Using

the probability of forging against a fixed participant, we can bound the probability of deceiving any honest participant as

$$P(\text{Forge}) = 1 - (1 - P(\text{FixedForge}))^{N(1-d_c)} \approx N^2(1-d_c)^2 p_t \qquad (10)$$

where we have used the fact that $P(\text{FixedForge}) \ll 1$ in the approximation. We again note that this probability decays exponentially fast with parameter k, and thus the protocol is secure against forging attempts. □

4.3 Transferability

In order to break the transferability of the protocol, a coalition C which includes the signer P_0 must generate a signature that is accepted by recipient $P_i \notin C$ at level l, while also being rejected by another recipient $P_j \notin C$ at a level $l' < l$.

Theorem 2. *The protocol defined in Section 3 is secure against non-transferability attempts.*

Proof. The task of the coalition is easiest if $l' = l - 1$ and so we consider this case in what follows. To provide an upper bound, we allow for the biggest coalition C that includes Nd_c recipients and the sender, i.e. all the dishonest participants. For simplicity, again we will fix the participants whom the coalition is trying to deceive to be the honest participants P_i and P_j, while all other honest participants are labelled with the index h. In general, according to the definitions in [27], transferability fails if the coalition forms a signature that is not transferable for at least one pair of honest participants (P_i, P_j). Therefore, we should take into account all possible pairs of honest participants. Here, we first focus on the case of a fixed pair of participants, and at the end we give the more general expressions.

The first step is to compute $p_{m_{l,l-1}}$, which is the probability that the test $T^m_{i,h,l}$ is passed (i.e. the part sent from honest participant P_h to recipient P_i is accepted at level l) and the test $T^m_{j,h,l-1}$ fails (i.e. the part sent from honest participant P_h to recipient P_j is rejected at level $l-1$). Since the sender, P_0 is dishonest, it can be assumed that the coalition know all the signature functions. However, they are unaware of the sets $R_{h\to i}$ and $R_{h\to j}$. Therefore, the coalition can control the number of mismatches the signature will make with the signature functions originally sent to P_h, but they cannot separately control the number of mismatches the signature will make with the functions in $F_{h\to i}$ and $F_{h\to j}$. Therefore, when participants P_i and P_j test the functions sent to them by honest participant

P_h they will both have the same expected fraction of mismatches; we will call this fraction p_e.

To compute a bound on the probability of both P_i accepting at level l and P_j rejecting at level $l-1$ we will consider

$$\begin{aligned} p_{m_{l,l-1}} &= \mathbb{P}(P_i \text{ passes test at level } l \text{ AND } P_j \text{ fails test at level } l-1) \\ &\leq \min\{\mathbb{P}(P_i \text{ passes test at level } l), \mathbb{P}(P_j \text{ fails test at level } l-1)\} \end{aligned} \tag{11}$$

The probability of passing the test at level l with an average error $p_e > s_l$ can be bounded using Hoeffding's inequalities to be below

$$\exp(-2(p_e - s_l)^2 k) \tag{12}$$

The probability of failing the test at level $l-1$ with average errors $p_e < s_{l-1}$ can similarly be bounded to be smaller than

$$\exp(-2(s_{l-1} - p_e)^2 k) \tag{13}$$

As in [27], since we are taking the minimum over both cases, the best choice for the coalition is to have these probabilities equal each other. This is achieved by using a fraction of errors $p_e = (s_l + s_{l-1})/2$ and in that case we obtain the bound

$$p_{m_{l,l-1}} \leq \exp\left(-\frac{(s_{l-1} - s_l)^2}{2} k\right) \tag{14}$$

which decays exponentially with k.

It is trivial for the coalition to make two recipients disagree in any way they wish for the results of a test that involves a member of the coalition, i.e. they can make $T^m_{i,c,l}$ and $T^m_{j,c,l-1}$ take any values they wish. However, the number of those tests are at most Nd_c, which is the maximum number of recipients in the coalition. For the participant P_i to accept a message at level l, he needs a fraction greater than δ_l of the tests to pass at this level. On the other hand, for the participant P_j to reject the message at level $l-1$, a fraction less than or equal to δ_{l-1} of tests must pass at this level. Therefore, since it holds that $\delta_l = \delta_{l-1} + d_c$, in order for the non-transferability to be successful, the honest participants P_i and P_j need to disagree on at least $Nd_c + 1$ tests. As we saw, the coalition can easily make them disagree on the Nd_c tests originating from them, but they still have to disagree on at least one more test originating from an honest participant. There are $N(\delta_l - d_c) + 1$ such tests (tests originating from

an honest participant that were passed), and the participants need only disagree on one of them for the coalition to succeed. Therefore, we have

$$\begin{aligned}\mathbb{P}(\text{Fixed Non-Transferability}) &\leq 1-(1-p_{m_{l,l-1}})^{N(\delta_l-d_c)+1}\\ &\approx (N(\delta_l-d_c)+1)p_{m_{l,l-1}}\end{aligned} \tag{15}$$

Lastly, we consider the general case, where the participants P_i and P_j are not fixed. Here, as in [27], we find

$$\begin{aligned}\mathbb{P}(\text{Non-Transferability}) &\leq 1-(1-\mathbb{P}(\text{Fixed Non-Transferability}))^{N_p}\\ &\approx N_p(N(\delta_l-d_c)+1)p_{m_{l,l-1}}\end{aligned} \tag{16}$$

where $N_p := [(N(1-d_c)][N(1-d_c)-1]/2$. Again, this decays exponentially with k, and thus the protocol is secure against non-transferability. □

4.4 Repudiation

Security against repudiation can be reduced to the special case of non-transferability from level $l = 0$ to level $l = -1$, thus we have the following:

Theorem 3. *The protocol defined in Section 3 is secure against repudiation attempts.*

Proof. The proof is a special case of non-transferability, see Section V A of [27]. We find

$$\mathbb{P}(\text{Rep}) \leq N_p(N(\delta_l-d_c)+1)p_{m_{0,-1}} \tag{17}$$

As for non-transferability, this goes to zero exponentially fast with k, and thus the protocol is secure against repudiation. □

We note here that equations (10), (16) and (17) are independent of the message size, meaning the signature size will be constant with respect to the size of the message being sent.

5 Protocol extensions

5.1 Reusability

A desirable extension of the current protocol would be to make the distributed keys reusable. Ideally, following the distribution stage the sender would be able to use the distributed keys to sign more than one message. With the current protocol, this is not possible – the definition of an ϵ-ASU$_2$ set means that to maintain security against forging attempts,

once the keys have been used to sign a message they must be discarded. Reusability could be obtained in two different ways. The first method would simply be to perform the distribution stage n times before moving on to the messaging stage. In this way, the sender would trivially be able to send n different messages in the future. The second method would be to distribute functions from an ϵ-ASU$_n$ set, instead of an ϵ-ASU$_2$ set as described above.

Definition 10 [30] *A hash function family $\mathcal{F}$ of functions from $\mathcal{M}$ to $\mathcal{T}$ is ϵ-ASU$_n$ provided that for all distinct elements $m_1, ..., m_n \in \mathcal{M}$ and for all (not necessarily distinct) $t_1, ..., t_n \in \mathcal{T}$ we have*

$$|\{f \in \mathcal{F} : f(x_i) = y_i, 1 \leq i \leq n\}| \leq \epsilon \times |\{f \in \mathcal{F} : f(x_i) = y_i, 1 \leq i \leq n-1\}|$$

It is shown in [30] that there exists a family of functions $\mathcal{F}$ that can be used to securely authenticate $n-1$ messages. The number of bits required to specify a function in this $\mathcal{F}$ is $u = (n^2-n+1)b+ns$, where s is an integer such that $a \leq ((n-1)b+s)(1+2^s)$, with a, b defined as in Proposition 1. Note that for $n = 2$ this reduces to the key length in (1).

For simplicity, we will have the sender perform the distribution stage n times, rather than using ϵ-ASU$_n$ functions. Note that for fixed tag length, b, and large s (with $n \ll s$), both methods require distribution of $O(ns)$ secret keys to leading order, where $a \approx s2^s$, and so in our case there is little advantage in using ϵ-ASU$_n$ functions as opposed to simply performing the distribution stage n times.

5.2 Latecomers

We might wonder whether it is possible for a new participant to enter the protocol after the distribution stage. In fact it is, but it requires either a trusted authority (see below), or for the new participant to communicate with all existing participants in the protocol. More concretely, to join, the sender would give the new participant $(N+1)k$ functions from the ϵ-ASU$_2$ set. The participant would then send k of the functions to each of the other recipients and keep k for himself. The other participants would each randomly select k of the Nk functions they hold and send them over secure channels to the latecomer. Following this, security follows in a very similar manner as before.

5.3 Designated sender

For practical applications of signature schemes, it is often useful for any participant to be able to sign a message, rather than having a desig-

nated sender. This can trivially be introduced to our protocol by having the participants perform the distribution stage $N + 1$ times, where each participant acts as the sender in one of the distribution stages.

5.4 Trusted authority

Our scheme requires participants to communicate pairwise with all other participants, as well as for secret keys to be distributed pairwise. For some applications, this may be too cumbersome a requirement, especially when all future participants are not known. In those situations, it is possible to greatly increase the efficiency of the protocol, at the expense of introducing a trusted authority. In the distribution stage, the signer would send Nk functions to the trusted authority, where N is an arbitrarily large number chosen to be the maximum number of participants able to verify the senders signature. When the sender wants to send a signed message, the trusted authority randomly (and secretly) sends k of the Nk functions to the recipient. Recipients could either obtain their k functions at the start of the protocol (i.e. have a distribution stage), or simply request the functions from the trusted authority as and when needed. Security against forging would follow as before from the properties of ϵ-ASU_2 sets, while security against repudiation would come from the fact that the trusted authority distributes the functions out at random, so each honest participant would have the same expected number of mismatches with any signature declaration. This would simplify the protocol in that all participants would only need to share a short secret key with the trusted authority, rather than requiring pairwise secret shared keys. Thus, as well as removing the need for pairwise communication between all parties, the total number of secret shared bits needed to generate the signing and verification algorithms would scale as $O(N)$, rather than $O(N^2)$ as in the unmodified protocol. Further benefits are that messages would be transferable an unlimited number of times between participants, and that if the sender gives an excess of keys to the trusted authority, latecomers can easily join by communicating solely with the trusted authority, who would send the latecomer k of the unused functions.

We remark here that in our scheme, unlike in public-key schemes, the trusted authority would be omnipotent, and would be able to cheat. On the other hand, for public-key schemes the certificate authority must be trusted to correctly distribute the public key, but without knowledge of the private key the trusted authority is still unable to cheat.

5.5 Extended protocol

In the sections that follow, we include some of the above extensions into the basic protocol described in Section 3. Namely, each participant will perform the distribution stage ψ times in the role of the sender. Thus the distribution stage is performed $(N+1)\psi$ times before the messaging stage takes place. In this case, any participant would be able to send up to ψ messages in future. We will refer to this as the *extended protocol* and we compare it to existing protocols in the sections that follow. Note that we do not include a trusted authority.

6 Comparisons

6.1 Comparison to "classical" USS schemes

In this section, we compare the performance of our extended protocol to the one proposed in [15], which we will refer to as HSZI. Our scheme enjoys a number of advantages when compared to the HSZI scheme. Namely,

1. We require fewer trust assumptions – the protocol does not require a trusted authority.
2. Security in our scheme can be tuned independently of message size, resulting in shorter signature lengths.
3. Our scheme scales more efficiently (with respect to message size) in terms of the number of secret shared bits required.

Here we will look at the second and third advantages in more detail. According to Theorem 3 of [15] (translated to our notation) the HSZI scheme has

$$|\Sigma| = q^{(Nd_c+1)} \tag{18}$$

$$|\mathcal{S}| = q^{(Nd_c+1)(\psi+1)} \tag{19}$$

$$|\mathcal{V}| = q^{N(d_c+\psi+1)} \tag{20}$$

Where Σ is the set containing all possible signatures, $\mathcal{S}$ is the set containing all possible signing algorithms, $\mathcal{V}$ is the set containing all possible verification algorithms, q is the number of elements in the chosen finite field and ψ is the number of times the keys can be reused.

Let us first consider the size of the signature. Since the signature must be transmitted with the message, it is desirable to have as small a signature as possible. In the HSZI scheme the message, m, is an element of the finite field F_q, meaning the size of the finite field must be at least as big as the size of the message set, i.e. $q \geq |\mathcal{M}|$. Accordingly, in what

follows we set $q = |\mathcal{M}|$. Using (18) we find that $(Nd_c + 1)\log(|\mathcal{M}|)$ is the length of the signature in bits. For this message length, the HSZI protocol gives a security level proportional to $1/|\mathcal{M}|$. Immediately then, we see that both the size of the signature and the security level depend on the size of the message to be sent.

In our scheme, the signature length is $2N^2k$ bits, regardless of the message length. The parameter k will depend on the desired security level, but is independent of the length of the message being signed.

We now consider the number of secret shared bits required to securely distribute the signing/verification keys. In the HSZI scheme, to secretly send the signing and verification keys to all participants requires the trusted authority to have

$$\big[(Nd_c + 1)(\psi + 1) + N(d_c + \psi + 1)\big]\log(|\mathcal{M}|) = O(N\psi \log|\mathcal{M}|) \quad (21)$$

secret shared bits with each participant, as can be seen from equations (19) and (20). For our protocol, each recipient must share Nky secret bits with the sender (to receive the signature functions), and $ky + k\log(Nk)$ with every other recipient (to forward on a selection of the key functions and their positions). For the extended protocol, where the distribution stage is performed ψ times for each participant acting as sender, each participant must share Nky secret bits with each of the N recipients for the ψ rounds in which he is the sender. In the $N\psi$ rounds when he is not the sender, the participant must share Nky bits with the sender and $ky + k\log(Nk)$) secret bits with each of the $(N-1)$ other non-sender participants. This is a total of

$$\begin{aligned} N^2k\psi y + N\psi\big[Nky + k(N-1)(y + \log(Nk))\big] \\ &= Nk\psi(3N-1)y + N(N-1)k\psi\log(Nk) \\ &= Nk\psi(3N-1)(6+2s) + N(N-1)k\psi\log(Nk) \\ &= O\big(N^2k\psi(\log\log|\mathcal{M}| + \log Nk)\big) \end{aligned} \quad (22)$$

secret shared bits per recipient. The second equality follows using (1) with $b = 2$. The last equality follows using the Lambert W function to find a leading order approximation for s when s is large [31]. The results are summarised in Table I below.

We can see that, with respect to the size of the message to be signed, our scheme scales favourably in terms of the number of shared secret bits required between participants and the length of the signature. The fact that our scheme scales unfavourably with the number of participants is

due to the lack of a trusted authority meaning participants must interact pairwise. As mentioned in Section 5.4, this N^2 scaling can be removed from our scheme by introducing a trusted authority.

	Our Protocol	HSZI	QDS
Sig	$2N^2k$	$(Nd_c+1)a$	$O(N^2a)$
Key	$O\big(N^2\psi(\log a+\log N)\big)$	$O(N\psi a)$	$O\big(N^2\psi(a+\log N)\big)$

Table 1: *Comparison of the signature length and secret shared keys required for various signature protocols. It can be seen that our scheme scales favourably with respect to the message length, $a = \log|\mathcal{M}|$, both in terms of signature length and required secret shared key. The "QDS" column refers to practical quantum digital signature schemes in general. Though there are many such schemes, the above rates are applicable to the two which are at present the most efficient, namely, generalised P2 in* [27] *and the protocol presented in* [23].

The HSZI scheme enjoys a number of advantages over our scheme associated with their use of a trusted authority. These are

1. Pairwise secret shared keys between all participants are not required. Instead, each participant only needs a shared secret key with the trusted authority. This means that the HSZI scheme scales favourably with respect to the number of participants.
2. Participants are able to enter the protocol even after the distribution stage. The new participant only needs to communicate with the trusted authority to join.
3. The HSZI protocol has unlimited transferability, whereas our protocol can only guarantee transferability a finite number of times.

While these advantages are significant, they are only possible due to the existence of a trusted authority – an additional trust assumption not present in our scheme. As highlighted in Section 5.4 our scheme could easily be modified to include the trusted authority, in which case it would achieve the same three benefits above, as well as being significantly more efficient.

6.2 Comparison to quantum signature schemes

Quantum signature schemes [10] aim at achieving unconditionally secure signatures whilst also making fewer assumptions than the existing clas-

sical USS schemes. So far, direct quantum signature experiments have been able to achieve unconditional security at the cost of sending $O(10^{10})$ quantum states per bit to be signed [25] (or, as recently predicted, $O(10^8)$ [23]). Although it may appear from Table I that quantum signatures scale comparably with the HSZI scheme, in fact the constant of proportionality for QDS schemes is very large, meaning that for all practical purposes the HSZI scheme is far more efficient. Until now, this decrease in efficiency has been justified by the fact that quantum protocols have made significantly fewer assumptions - there is no broadcast channel or anonymous communication channels, as was assumed in [12], and no trusted authority, as was assumed in [15]. The only assumption is that the majority of participants are honest, and that the participants all share a number of secret bits, which could be generated via QKD.

In the protocol presented in this paper, we make *the same* trust assumptions as in QDS schemes, and still achieve two key advantages. Namely, our protocol requires significantly fewer secret shared bits and generates a much shorter signature, see Table I above[3]. One of the reasons for the increase in efficiency is that, so far, all QDS schemes have been of the Lamport signature type [32], in which the distribution stage must be performed for every possible future message. On the other hand, our protocol requires users to share hash functions in the distribution stage, which can be used for any future message (up to some chosen size).

Acknowledgments

R.A. and E.A. were supported by the UK Engineering and Physical Sciences Research Council (EPSRC) under EP/M013472/1 and EP/K022717/1. A.A. acknowledges the support by the European Commission through FP7 "Harmonized framework allowing a sustainable and robust identity for European Citizens (EKSISTENZ)," with grant number: 607049.

References

1. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM **21**(2) (1978) 120–126

[3] It should be noted that the QDS figures in Table I are only expected rates. Security has been proven for the case of sending a 1-bit message, and it is expected that to send longer messages the protocol can simply be iterated to sign each message bit, however one may have to be careful when defining security for such a scheme, since each message bit is signed independently

2. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: CRYPTO'84, Springer (1985) 10–18
3. Johnson, D., Menezes, A., Vanstone, S.: The elliptic curve digital signature algorithm (ecdsa). International Journal of Information Security **1**(1) (2001) 36–63
4. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In Goldwasser, S., ed.: Proceedings 35th Annual Symposium on Foundations of Computer Science. Volume 35 of SFCS '94., IEEE Computer Society (1994) 124–134
5. Agency, N.S.: Cryptography Today (August 2015). `https://www.nsa.gov/ia/programs/suiteb_cryptography/` (2015)
6. McEliece, R.J.: A public-key cryptosystem based on algebraic coding theory (1978)
7. Micciancio, D.: Lattice-based cryptography. In: Encyclopedia of Cryptography and Security. Springer (2011) 713–715
8. Song, F.: A note on quantum security for post-quantum cryptography. In: Post-Quantum Cryptography. Springer (2014) 246–265
9. Biasse, J.F., Song, F.: On the quantum attacks against schemes relying on the hardness of finding a short generator of an ideal in $Q(\zeta \mathrm{pn})$. (2015)
10. Amiri, R., Andersson, E.: Unconditionally secure quantum signatures. Entropy **17**(8) (2015) 5635–5659
11. Wallden, P., Dunjko, V., Kent, A., Andersson, E.: Quantum digital signatures with quantum-key-distribution components. Physical Review A **91**(4) (2015) 042304
12. Chaum, D., Roijakkers, S.: Unconditionally secure digital signatures. In: CRYPTO '90. LNCS, Springer-Verlag (1991) 206–214
13. Pfitzmann, B., Waidner, M.: Information-theoretic pseudosignatures and byzantine agreement for $t \geq n/3$. IBM (1996)
14. Shamir, A.: How to share a secret. Communications of the ACM **22**(11) (1979) 612–613
15. Hanaoka, G., Shikata, J., Zheng, Y., Imai, H.: Unconditionally secure digital signature schemes admitting transferability. In: ASIACRYPT 2000. Springer (2000) 130–142
16. Hanaoka, G., Shikata, J., Zheng, Y.: Efficient unconditionally secure digital signatures. IEICE transactions on fundamentals of electronics, communications and computer sciences **87**(1) (2004) 120–130
17. Shikata, J., Hanaoka, G., Zheng, Y., Imai, H.: Security notions for unconditionally secure signature schemes. In: EUROCRYPT 2002, Springer (2002) 434–449
18. Swanson, C.M., Stinson, D.R.: Unconditionally secure signature schemes revisited. In: Information Theoretic Security. LNCS, Springer (2011) 100–116
19. Gottesman, D., Chuang, I.: Quantum digital signatures. arXiv preprint quant-ph/0105032 (2001)
20. Lu, X., Feng, D.: Quantum digital signature based on quantum one-way functions. In: ICACT 2005. Volume 1., IEEE (2005) 514–517
21. Clarke, P.J., Collins, R.J., Dunjko, V., Andersson, E., Jeffers, J., Buller, G.S.: Experimental demonstration of quantum digital signatures using phase-encoded coherent states of light. Nature communications **3** (2012) 1174
22. Dunjko, V., Wallden, P., Andersson, E.: Quantum digital signatures without quantum memory. Physical review letters **112**(4) (2014) 040502
23. Amiri, R., Wallden, P., Kent, A., Andersson, E.: Quantum signatures using insecure quantum channels (2015)
24. Collins, R.J., Donaldson, R.J., Dunjko, V., Wallden, P., Clarke, P.J., Andersson, E., Jeffers, J., Buller, G.S.: Realization of quantum digital signatures without the requirement of quantum memory. Physical review letters **113**(4) (2014) 040502

25. Donaldson, R.J., Collins, R.J., Kleczkowska, K., Amiri, R., Wallden, P., Dunjko, V., Jeffers, J., Andersson, E., Buller, G.S.: Experimental demonstration of kilometer-range quantum digital signatures. Physical Review A **93**(1) (2016) 012329
26. Scarani, V., Bechmann-Pasquinucci, H., Cerf, N.J., Dušek, M., Lütkenhaus, N., Peev, M.: The security of practical quantum key distribution. Reviews of modern physics **81**(3) (2009) 1301
27. Arrazola, J.M., Wallden, P., Andersson, E.: Multiparty quantum signature schemes. Accepted to Quantum Information and Computation (2016)
28. Carter, L., Wegman, M.N.: Universal classes of hash functions. J. Comput. Syst. Sci. **18** (1979) 143–154
29. Bierbrauer, J., Johansson, T., Kabatianskii, G., Smeets, B.: On families of hash functions via geometric codes and concatenation. In Stinson, D., ed.: CRYPTO '93. Volume 773 of LNCS., Springer-Verlag (1994) 331–342
30. Atici, M., Stinson, D.R.: Universal hashing and multiple authentication. In Koblitz, N., ed.: CRYPTO '96. Volume 1109 of LNCS. (1996) 16–30
31. Abidin, A., Larsson, J.Å.: New universal hash functions. In Lucks, S., Armknecht, F., eds.: WEWoRC 2011. Volume 7242 of LNCS., Springer-Verlag (2012) 99–108
32. Lamport, L.: Constructing digital signatures from a one-way function. Technical report, Technical Report CSL-98, SRI International Palo Alto (1979)