Efficient unconditionally secure signatures using universal hashing

Efficient unconditionally secure signatures using universal hashing

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]imec-COSIC KU Leuven, 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; petros.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.
- We prove information-theoretic security of our scheme against forging, repudiation, and non-transferability.
- We compare our scheme with existing both "classical" (not employing quantum mechanics) and quantum unconditionally secure signature schemes. The comparison shows that our new scheme, despite requiring fewer resources, is also much more efficient than all 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. Large companies such as Google, Microsoft and IBM certainly seem to think it's possible, and are allocating significant resources to research in this area. Furthermore, the National Security Agency (NSA) in the USA is also 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. Instead, these schemes provide information-theoretic security. Such a high level of security, however, comes at a cost. So far, all USS schemes have 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 USS 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 where digital signatures are used. Nevertheless, there may still be applications where USS schemes are 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, USS schemes should not be viewed as a standalone product. Instead, it should be viewed as a complement to existing QKD systems in fixed networks environments.

In this work, 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

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

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 USS scheme proposed in Ref. [11].

Classical USS 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 using untraceable sending protocols. Their scheme, however, only allows users to sign 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 lengths 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 classical USS schemes are proposed and analysed in Shikata *et al.* [17] as well as Swanson and Stinson [18].

Quantum USS schemes. There are also quantum USS schemes, first proposed by Gottesman and Chuang [19], in which security is derived from the laws of quantum physics. Lu and Feng [20] proposed a quantum USS scheme using quantum one-way functions, though it required a trusted authority (which they called an arbiter) to resolve disputes. Quantum USS 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 [22, 11, 23]. Quantum USS schemes without quantum

memories have also been experimentally demonstrated [24, 25]. Furthermore, these recent schemes and their experimental demonstrations use 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 USS 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 previous USS schemes. We do not assume a trusted authority, nor the existence anonymous channels or 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.

1. We construct an USS scheme that, unlike prior schemes, does not rely on a trusted authority, anonymous channels or broadcast channels (Section 3).
2. We prove information-theoretic security of our scheme against forging, repudiation, and non-transferability (Section 4).
3. We compare our scheme 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 (Section 5).

The distribution stage of our scheme derives from the Generalised P2 protocol described in Ref. [27]. However, instead of participants distributing bits, in our scheme a sender shares with each of the remaining protocol participants (or recipients) a set of keys (hash functions) from a family of universal hash functions. This conceptual difference leads to vast efficiency improvements (see Section 5) as it allows the distribution stage to be performed only once for all possible future messages, as opposed to Generalised P2 in which the distribution stage is performed independently *for each* future message. This is because, in our scheme, a signature for a message is a vector of tags generated by applying the hash functions to the message. Our scheme can be viewed as an extension of MAC schemes, and therefore its practical implementation is straightforward and efficient.

2 Preliminaries

We begin by formally defining an USS scheme.

Definition 1 ([27]) *An USS scheme $\mathcal{Q}$ is an ordered set $\{\mathcal{P}, \mathcal{M}, \Sigma, L,$ Gen, Sign, 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.*

- Σ *is the set of possible signatures.*
- *Gen is the generation algorithm that gives rise to the functions Sign and Ver, used respectively 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. Based on the data obtained during the distribution stage, the generation algorithm instructs how to construct the functions Sign and Ver. The generation algorithm includes the option of outputting an instruction to abort the protocol.*
- *Sign*: $\mathcal{M} \to \Sigma$ *is a deterministic function that takes a message* $m \in \mathcal{M}$ *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* $l_{max} \leq N$.
- *Ver*: $\mathcal{M} \times \Sigma \times \mathcal{P} \times L \to \{$*True*, *False*$\}$ *is a deterministic function that takes a message* m*, a signature* σ*, a participant* P_i *and a level* l*, and gives a boolean 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* $\mathrm{Ver}_{i,l}(m, \sigma) := \mathrm{Ver}(m, \sigma, i, l)$.

Definition 3 *A signature* σ *on a message* m *is* i*-acceptable if* $\mathrm{Ver}_{i,0}(m, \sigma) = \mathrm{True}$.

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

The signature protocol presented in this paper uses almost strongly universal hash function families.

Definition 5 ([28]) *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 the ability to find an ϵ-ASU_2 set which is "small".

Proposition 1 ([29]) *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* $a = (b + s)(1 + 2^s)$.

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

3 The Protocol

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 were to know which hash functions are held by which participant, she 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 one set (using secret channels) to each other recipient, keeping one for himself. The recipients test each of the N sets independently.

Transferability levels. 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. 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. For these known keys, 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 and the number of incorrect sets found between each honest party. To ensure transferability then, we require that the second verifier accepts a message as authentic even if each set contains a higher number of mismatches, and there are more invalid sets than found by the first verifier. Of course, to ensure security against forging, we cannot allow message-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 number 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 ω such that $\omega < (N+1)/2$, since the protocol cannot be made secure using the majority vote dispute resolution process if more than half of the participants are dishonest. We also define the notation $d_R := (\omega - 1)/N$, where d_R is the maximum fraction of dishonest *recipients* possible when the sender is part of the coalition. As in previous protocols, there are two stages – the distribution stage and the messaging stage.

3.1 Distribution Stage

1. The sender independently and uniformly at random 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 that 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 any verification level, the recipient will perform N such tests, one for each $j = 1, ..., N$. Note that participant P_i knows all of the signature functions $f_{i'}$ with $i' \in R_i$ and so can perform all tests *without* interaction with any other participant.

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 $\delta_l = 1/2 + (l+1)d_R$. Therefore, we see that each participant can accept/reject a message without interacting with any other recipient in the messaging stage.

4. To forward a message, participant P_i simply forwards (m, Sig_m) to the desired recipient.

Note that the number of dishonest participants the protocol is able to tolerate is directly related to the number of allowed transferability levels, according to the parameter $\delta_l = 1/2 + (l+1)d_R$. Specifically, the maximum transferability level for a given number of dishonest participants is set by

$$(l_{\max} + 1)d_R < 1/2. \tag{4}$$

4 Security

Informally, USS schemes must provide the following three security guarantees [18]:

1. Unforgeability: Except with negligible probability, it should not be possible for an adversary to create a valid signature.
2. Transferability: If a verifier accepts a signature, he should be confident that any other verifier would also accept the signature.
3. Non-repudiation: Except with negligible probability, a signer should be unable to repudiate a legitimate signature that he has created.

Formal security definitions covering both quantum and classical USS schemes were first provided in Ref. [27]. For completeness, the definitions are reproduced in Appendix A. Below we prove that the scheme presented in Section 3 is secure against each type of dishonest behaviour. The security analysis for transferability and non-repudiation is similar to the one provided in Ref. [27], and as such it is presented in Appendix B.

Theorem 1. *The protocol defined in Section 3 is secure against forging attempts. Letting H_2 denote the binary entropy, we find*

$$\mathbb{P}(\textit{Forge}) \leq (N - \omega)^2 \, 2^{-k(1-H_2(s_0))}. \tag{5}$$

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

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 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 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 $N\delta_0 + 1$ of the N tests that P_i performs in step 2 of the messaging stage, where $m' \neq m$ and $\delta_0 = 1/2 + d_R$, meaning $N\delta_0 + 1 = N/2 + \omega$. By the definition of the protocol, the number of members in a coalition is at most ω. The coalition knows $F_{j\to i}$ and $R_{j\to i}$ for all $P_j \in C$, so they can use this knowledge to trivially ensure that P_i passes ω of the N tests performed at level $l = 0$. To pass the required $N\delta_0 + 1$ tests, the coalition must pass a further $N/2$ tests out of the $N - \omega$ remaining tests. The first step in computing this probability is to calculate the probability of the coalition being able to create a signature such that, for a single $P_j \notin C$, $T^{m'}_{i,j,0} = 1$, i.e. the probability that the coalition can guess the tags forwarded from a single honest recipient P_j to P_i.

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 $F_{j\to i}$ and $R_{j\to i}$ 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 5 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 5, 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 independently true for each of the k unknown functions.

Let X be the random variable that counts how many incorrect tags the coalition declares. Then X follows a binomial distribution and we have

$$p_t = \mathbb{P}(X < ks_0) = \sum_{v=0}^{ks_0-1} \binom{k}{v} \left(\frac{2}{|\mathcal{T}|}\right)^{k-v} \left(1 - \frac{2}{|\mathcal{T}|}\right)^{v}. \tag{6}$$

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

$$p_t = \sum_{v=0}^{ks_0-1} \binom{k}{v} \left(\frac{1}{2}\right)^k \approx 2^{-k(1-H_2(s_0))}. \tag{7}$$

Of course, 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 $N-\omega$ tests, which is given by

$$P(\text{FixedForge}) \leq 1 - (1-p_t)^{N-\omega} \approx (N-\omega)p_t, \tag{8}$$

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

The total number of honest recipients is $N-\omega$ 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

$$P(\text{Forge}) = 1 - (1 - P(\text{FixedForge}))^{N-\omega} \approx (N-\omega)^2 p_t, \tag{9}$$

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.

Theorem 2. *The protocol defined in Section 3 is secure against non-transferability attempts. Defining* $N_p := [(N(1-d_R)][N(1-d_R)-1]/2$*, we find*

$$\mathbb{P}(\textit{Non-Transferability}) \leq N_p(N(\delta_l - d_R) + 1) \exp\left(-\frac{(s_{l-1}-s_l)^2}{2}k\right). \tag{10}$$

Proof. See Appendix B.

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

$$\mathbb{P}(\textit{Rep}) \leq N_p(N(\delta_l - d_R) + 1) \exp\left(-\frac{(s_{-1}-s_0)^2}{2}k\right). \tag{11}$$

Proof. See Appendix B.

We note here that equations (5), (10) and (11) are independent of the message size, meaning the signature size will be constant with respect to the size of the message being sent.

5 Comparisons

5.1 Classical USS schemes

In this section we compare the performance of our protocol to the one proposed in [15] constructed using polynomials over a finite field. We will refer to this protocol as the

HSZI scheme. Since the HSZI scheme allows all participants to send multiple messages, we extend our protocol to facilitate a comparison.

Consider the protocol described in Section 3, except that now each participant performs the distribution stage ψ times in the role of the sender. Trivially, this extended distribution stage allows all participants in the scheme to send up to ψ messages securely in the role of sender. We call this the *extended protocol* and all comparisons are made with reference to this scheme.

This extended scheme still enjoys a number of advantages when compared to the HSZI scheme. Namely,

1. We require fewer trust assumptions – our scheme does not require a trusted authority.
2. Security in the 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 by participants.

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

$$|\Sigma| = q^{(\omega+1)}, \quad |\mathcal{S}| = q^{(\omega+1)(\psi+1)}, \quad |\mathcal{V}| = q^{\omega+(N+1)(\psi+1)}, \tag{12}$$

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.

Signature length. 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, 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 $q = |\mathcal{M}|$. Eq. (12) implies that $(\omega + 1)\log(|\mathcal{M}|)$ is the bit-length of the signature. The authors also show that the HSZI scheme provides security proportional to $1/|\mathcal{M}|$.

Immediately we see that both the size of the signature and the security level depend on the size of the message to be sent. On the other hand, in our scheme the signature length is $2N^2k$ bits, regardless of the message length. The security level of our scheme depends on the parameter k, but is independent of the length of the message being signed. This allows our scheme to bypass the optimality results presented in Ref. [15]. Specifically, the authors show that the signature generated by the HSZI scheme is optimally small *for a given security level*. By decoupling the security level from the size of the message being sent, we are able to generate smaller signatures while maintaining security.

Secret key requirements. 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, the trusted authority must hold

$$\big[(\omega + 1)(\psi + 1) + \omega + (N + 1)(\psi + 1)\big] \log(|\mathcal{M}|) = O(N\psi \log |\mathcal{M}|) \tag{13}$$

secret shared bits with each participant (as implied by Eq. (12)).

For the hash scheme, 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: (i) Nky secret bits with each of the N recipients for the ψ rounds in which he is the sender; and (ii) Nky bits with the sender and $ky + k\log(Nk))$ secret bits with each of the $(N-1)$ other recipients for each of the $N\psi$ rounds when he is not the sender. This is a total of

$$\begin{aligned} N^2k\psi y + N\psi\big[Nky + k(N-1)(y+\log(Nk))\big] \\ &= Nk\psi(3N-1)y + N(N-1)k\psi\log(Nk) \\ &= Nk\psi(3N-1)(6+2s) + N(N-1)k\psi\log(Nk) \\ &= O\big(N^2k\psi(\log\log|\mathcal{M}| + \log Nk)\big) \end{aligned} \tag{14}$$

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 [30]. The results are summarised in Table 1 below.

The table shows that the signature length in our scheme is constant with respect to the size of the message to be signed. On the other hand, the signature length in the HSZI scheme increases linearly with the bit-length of the message to be signed. Similarly, the secret shared key required by our scheme increases logarithmically with the bit-length of the message, whereas the increase in the HSZI scheme is linear in the bit-length of the message.

The fact that our scheme scales unfavourably with respect to the number of participants is due to the lack of a trusted authority, meaning participants must perform the pairwise exchange process. As discussed below, this N^2 scaling can be removed from the hash scheme by introducing a trusted authority.

	Hash scheme	HSZI	Quantum
Signature	$2N^2k$	$(\omega+1)a$	$O(N^2a)$
Secret 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. 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 "Quantum" column refers to the two most efficient quantum USS schemes at present, described in [27] and [23].

Disadvantages. Due to the inclusion of a trusted authority, the HSZI scheme enjoys a number of advantages over our scheme. These are:

1. Pairwise secret shared keys between all participants are not required by the HSZI scheme. 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 protocol participants.

2. Participants in the HSZI scheme 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 scheme 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. 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.

A trusted authority could be included into our scheme as follows. 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.

5.2 Quantum USS schemes

A central motivating factor in the study of quantum USS schemes was that they seemed to require fewer resources than classical USS schemes. This benefit came at a cost, and all quantum USS schemes proposed have been much less efficient than classical USS schemes[3].

Until now, this decrease in efficiency had been justified by the fact that quantum protocols do not require broadcast channels, anonymous channels, or a trusted authority. Instead, the only assumption is that a limited number of the participants are dishonest, and that the participants all share a number of secret bits, which could be expanded via QKD.

However, the classical scheme presented in this paper makes *the same* trust assumptions as quantum USS schemes, and still achieves two key advantages. Namely, our scheme generates much shorter signatures and requires significantly fewer secret shared bits. One of the reasons for the increase in efficiency is that, so far, all quantum USS schemes have been of the Lamport-type, in which the distribution stage must be performed for every possible future message. On the other hand, our scheme does not follow this blueprint, and instead requires users to share hash functions in the distribution stage, which can be used to sign any future message (up to some chosen size).

[3] Although it may appear from Table 1 that quantum USS schemes scale comparably to the HSZI scheme, in fact the constant of proportionality for the quantum schemes is very large, meaning that for all practical purposes the HSZI scheme is far more efficient.

Efficiency. Here we consider the signature length and secret shared bit requirements of our scheme, and compare it to Generalised P2, the most efficient realisable quantum USS scheme. We assume that a group of $N + 1 = 51$ participants are trying to sign a 1Mb message to a security level of 10^{-10}. For comparing to quantum USS schemes, rather than considering the extended protocol, we assume the participants perform the regular distribution stage as specified in Section 3, i.e. there is a designated sender and only one message to be sent. In order to have $l_{\max} = 1$, we assume that at most $\omega = 13$ participants are dishonest meaning $d_R = 0.24$. We also choose $s_{-1} = 0.41$, $s_0 = 0.21$ and $s_1 = 0.01$ so as to have even gaps between the verification levels[4].

With these parameters, Eqs. (5), (10) and (11) show that $k = 1700$ is necessary for the message to be secure to a level of 10^{-10}. Given this value of k, the signature length is 8.50×10^6 bits and each recipient must hold a total of 7.69×10^6 secret shared bits (shared over the different participants).

When considering Generalised P2, we assume the sender signs the 1Mb message bit-by-bit, each to a level of 10^{-10}. Overall this gives a lower security level than signing the message as a whole, but makes the protocol significantly more efficient[5]. Eqs. (24), (29) and (31) of Ref. [27] can be used to show that the resulting signature length is 4.25×10^{12}, and that each recipient must hold a total of 5.96×10^{12} secret shared bits (shared over the different participants).

This example shows just how powerful our new scheme is when compared to existing quantum schemes – even for a relatively small message, our scheme is 6 orders of magnitude more efficient both in terms of signature size and secret shared bit requirements. Our results show that quantum USS schemes must either be drastically improved, or find a new source of motivation if they are to remain competitive.

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
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

[4] This choice is somewhat arbitrary, but is chosen to minimise the required signature lengths.
[5] Signing the message as a whole would require participants to share secret keys of size $O(2^{|\mathcal{M}|}) = O(2^{10^6})$, which is clearly impossible.

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. 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

A Security definitions

In this section we formally define security in USS protocols. We begin by defining the notion of a dispute resolution process.

In the messaging stage of the protocol all participants are able to check the validity of a message-signature pair without communicating with any other participant. Nevertheless, there may still be scenarios in which disagreements arise regarding whether a message is valid or not. For example, the sender may deny having ever sent a message, even though a recipient who (allegedly) followed the correct procedure found the message to be valid. In these cases, the participants need a method of deciding who is telling the truth. This is done via the dispute resolution process.

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

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

Essentially, all participants check the message-signature pair at level -1 and the majority decision prevails. 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. It is expected that even dishonest participants would not try to force dispute resolution, since losing would come with consequences and the procedure ensures that honest participants prevail as long as they are in the majority. 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.

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:*

$$Forg_C(\mathcal{Q}, m', \sigma') = \begin{cases} 1 & \text{if } (m', \sigma') \text{ is } i\text{-acceptable for some } P_i \notin C \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

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:*

$$NonTrans_C(\mathcal{Q}, m, \sigma, l) = \begin{cases} 1 & \text{if } \mathrm{Ver}_{(i,l)}(m, \sigma) = \text{True for some } P_i \notin C \text{ and} \\ & \mathrm{Ver}_{(j,l')}(m, \sigma) = \text{False for some } 0 \leq l' < l \\ & \text{and some } j \neq i, P_j \notin C \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

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:*

$$Rep_C(\mathcal{Q}, \mathrm{MV}, m, \sigma) = \begin{cases} 1 & \text{if } (m,\sigma) \text{ is } i\text{-acceptable for some } P_i \notin C \text{ and} \\ & \mathrm{MV}(m,\sigma) = \text{Invalid} \\ 0 & \text{otherwise.} \end{cases} \tag{17}$$

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.

B Security proofs

In this section we prove Theorem 2 and Theorem 3 stated in Section 4.

B.1 Proof of Theorem 2

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 $l' < l$.

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_R 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, 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. We begin by focusing 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: (i) test $T^m_{i,h,l}$ is passed (i.e. the tags sent from honest participant P_h to recipient P_i are accepted at level l); and (ii), the test $T^m_{j,h,l-1}$ fails (i.e. the tags sent from honest participant P_h to recipient P_j are 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 bias 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 an honest participant P_h, they will both have the same expected fraction of mismatches; we call this fraction p_e.

It is helpful to use the following bound

$$\begin{aligned} p_{m_{l,l-1}} &= \mathbb{P}(P_i \text{ passes test at level } l \text{ AND } P_j \text{ fails test at level } l-1) \\ &\leq \min\{\mathbb{P}(P_i \text{ passes test at level } l), \mathbb{P}(P_j \text{ fails test at level } l-1)\}. \end{aligned} \tag{18}$$

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

$$\exp(-2(p_e - s_l)^2 k). \tag{19}$$

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

$$\exp(-2(s_{l-1} - p_e)^2 k). \tag{20}$$

Note that $s_{l-1} > s_l$ and so the above two cases cover all possible values for p_e. Since we are taking the minimum over both cases, the optimal choice for the coalition is to have these probabilities equal to each other. This is achieved by choosing $p_e = (s_l + s_{l-1})/2$. In this case we obtain the bound

$$p_{m_{l,l-1}} \leq \exp\left(-\frac{(s_{l-1} - s_l)^2}{2} k\right), \tag{21}$$

which decays exponentially with k.

For a test that involves a member of C it is trivial for the coalition to make two recipients disagree in any way they wish, 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 is at most Nd_R, which is the maximum number of recipients in the coalition. For the participant P_i to accept a message at level l, he needs strictly greater than $N\delta_l$ of the tests to pass at this level. On the other hand, for the participant P_j to reject the message at level $l-1$, less than or equal to $N\delta_{l-1}$ of tests must pass at this level. Therefore, since it holds that $\delta_l = \delta_{l-1} + d_R$, in order for the coalition to be successful, the honest participants P_i and P_j need to disagree on at least $Nd_R + 1$ tests. As we saw, the coalition can easily make them disagree on the Nd_R tests originating from coalition members, but they still have to disagree on at least one more test originating from an honest recipient. There are $N(\delta_l - d_R) + 1$ such tests (tests originating from an honest recipient that were passed by P_i), and the P_j need only reject one of them for the coalition to succeed. Therefore, we have

$$\begin{aligned} \mathbb{P}(\text{Fixed Non-Transferability}) &\leq 1 - (1 - p_{m_{l,l-1}})^{N(\delta_l - d_R)+1} \\ &\approx (N(\delta_l - d_R) + 1) p_{m_{l,l-1}}. \end{aligned} \tag{22}$$

Lastly, we consider the general case, where the participants P_i and P_j are not fixed. We find

$$\begin{aligned} \mathbb{P}(\text{Non-Transferability}) &\leq 1 - (1 - \mathbb{P}(\text{Fixed Non-Transferability}))^{N_p} \\ &\approx N_p (N(\delta_l - d_R) + 1) p_{m_{l,l-1}}, \end{aligned} \tag{23}$$

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

B.2 Proof of Theorem 3

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

$$\mathbb{P}(\text{Rep}) \leq N_p (N(\delta_0 - d_R) + 1) p_{m_{0,-1}}. \tag{24}$$

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