Authentication and Key Management Automation in Decentralized Secure Email and Messaging via Low-entropy Secrets

Itzel Vazquez Sandoval[1], Arash Atashpendar[1,2] and Gabriele Lenzini[1]

[1]*SnT, University of Luxembourg, Luxembourg*
[2]*itrust Consulting, Luxembourg*

Keywords: Authentication, Key Management, Secure Email and Messaging, Password-Authenticated Key Exchange.

Abstract: We revisit the problem of entity authentication in decentralized end-to-end encrypted email and secure messaging to propose a practical and self-sustaining cryptographic solution based on password-authenticated key exchange (PAKE). This not only allows users to authenticate each other via shared low-entropy secrets, e.g., memorable words, without a public key infrastructure or a trusted third party, but it also paves the way for automation and a series of cryptographic enhancements; improves security by minimizing the impact of human error and potentially improves usability. First, we study a few vulnerabilities in voice-based out-of-band authentication, in particular a combinatorial attack against lazy users, which we analyze in the context of a secure email solution. Next, we propose solving the problem of *secure equality test* using PAKE to achieve entity authentication and to establish a shared high-entropy secret key. Our solution lends itself to offline settings, compatible with the inherently asynchronous nature of email and modern messaging systems. The suggested approach enables enhancements in key management such as automated key renewal and future key pair authentications, multi-device synchronization, secure secret storage and retrieval, and the possibility of post-quantum security as well as facilitating forward secrecy and deniability in a primarily symmetric-key setting. We also discuss the use of auditable PAKEs for mitigating a class of online guess and abort attacks in authentication protocols.

1 INTRODUCTION

The use of email and instant messaging (IM) has become pervasive and entrenched in the fabric of modern communication. Thanks to cryptography, modern messaging tools have reached a considerable degree of sophistication (e.g., Signal) and offer advanced security features ranging from end-to-end encryption to forward secrecy and deniability. For these reasons, coupled with better usability, although email has a long history and remains undeniably popular with hundreds of billions of emails exchanged on a daily basis (Clark et al., 2018), secure messaging has often been recommended by security experts as the go-to tool for secure communication. Yet, secure messaging and email share two long-standing challenges, namely entity authentication and key management.

The primary concern is entity authentication, which invariably involves a mechanism that associates some cryptographic material with an identity, e.g., public key authentication. Key management, affecting email more acutely, is intertwined with authentication and the need for automating it has been known for a

long time, e.g., see (Ruoti et al., 2018).

Over the years, several methods have been established to tackle authentication, and indirectly key management: manual validation, web of trust, public key infrastructure (PKI) and hierarchical validation, public key directories as well as server-derived public keys such as identity-based encryption (IBE). The set of viable candidates becomes much smaller once we consider a decentralized setting, i.e., without a PKI or a trusted third party (TTP). For this scenario, the body of work on key authentication contains hundreds of works focusing on methods based on the use of out-of-band (OOB) channels and short authentication string (SAS) comparisons, see Section 1.1. However, when it comes to schemes that rely on low-entropy shared secrets, which is what we address here, the only work that to the best of our knowledge proposes a solution is by (Alexander and Goldberg, 2007). They use a modified solution to the socialist millionaires' problem (SMP) by (Boudot et al., 2001), also known as secure equality test, for authentication in the off-the-record messaging (OTR) protocol.

Due to the required user interaction in most of

these approaches, e.g., for verifying the authenticity of an interlocutor's public key, usability plays a key role in achieving authentication. Reducing the gap between security and usability, by finding optimal trade-offs, has been a central theme for decades with a plethora of long-standing open problems, e.g., see (Unger et al., 2015; Clark et al., 2018).

Here we revisit the problem of authenticating public keys in a decentralized setting and propose a user-friendly and robust approach based on password-authenticated key exchange (PAKE) to solve SMP via low-entropy secrets. These secrets are not expected to be sampled from a large, uniformly distributed space, but rather from a small set of values, e.g., typical human-memorable passwords or pin numbers. The task of SMP boils down to two parties verifying equality of their inputs $\pi_{\mathcal{A}}$ and $\pi_{\mathcal{B}}$ in a zero-knowledge manner such that by the end they learn nothing but the boolean result of the test. By solving SMP via PAKE, we also establish a shared cryptographically strong secret key, making further cryptographic enhancements possible. Furthermore, this approach would not require any understanding of cryptographic concepts from the user, e.g., knowing about public-keys and fingerprints.

We also show how the suggested approach would not only work naturally in the context of secure messaging, but also in the inherently asynchronous setting of email. Apart from offering improved usability properties and eliminating a host of vulnerabilities present in OOB-based protocols, as discussed in Section 3, we show how the PAKE-generated secret key can be used to pave the path towards providing a series of enhancements in secure email and messaging. These include inattentive user resistance, automated key renewal, automated future key pair authentication and multi-device synchronization, along with security properties such as deniability, forward secrecy, post-quantum security, auditability for detecting guess and abort attacks, secure secret storage and retrieval with applications in email and secure messaging.

By applying PAKE to this problem, we advance the state-of-the-art in the use of shared low-entropy secrets for entity authentication, an idea considered only in (Alexander and Goldberg, 2007). Also note that while SMP is a subproblem solved naturally by PAKE, the latter has not been applied to the problem of authenticating public keys in decentralized settings.

Motivation. Entity authentication in decentralized, non-PKI environments is generally brushed aside. Solutions that do consider this problem typically rely on users correctly executing a manual comparison and even tend to keep this feature rather hidden, e.g., Signal. Our incentive for replacing OOB authentication with a cryptographic protocol is the impact of failures occurring in methods highly-dependent on user behavior, which could completely jeopardize security.

Our motivation for using PAKE—a method that does not seem to have enjoyed enough recognition due to a lack of mature implementations, reluctance towards client side crypto, patent-encumbered designs and perhaps even unawareness of its usefulness—is grounded not only in its independence from a PKI or a TTP, but also in its provision of a zero-knowledge (ZK) solution for the secure equality test problem using a low number of rounds, compatible with asynchronous settings, and in the fact that it enables additional cryptographic enhancements.

We were also motivated by two open problems stressed by (Unger et al., 2015; Clark et al., 2018): bridging the gap between known theoretical results and real-world solutions, and the need for more robust authentication methods that also improve the trade-off between security and usability in secure solutions. Finally, the need for addressing common challenges such as key management automation and device synchronization also spurred us on.

Contributions and Structure. After a brief review of the state-of-the-art in Section 1.1, we cover background concepts in Section 2. In Section 3, we focus on a few vulnerabilities in the use of OOB channels for authentication, including a partial preimage attack aimed at lazy users, which we analyze in the context of the p≡p secure email solution. In Section 4, we present an efficient PAKE-based solution for authentication in secure messaging and email via low-entropy secrets, which enables further cryptographic enhancements. We provide a concrete illustrative scheme along with an analysis of various PAKE constructions and properties relevant for our work. We show how our proposal can be used to achieve additional cryptographic tasks and properties such as automation in key management and key renewal, forward secrecy in a symmetric-key setting, deniability, post-quantum security, secure secret retrieval, and auditability for mitigating a certain class of online guess and abort attacks. We briefly analyze network transport mechanisms and security. Section 5 concludes with remarks on future work.

1.1 Related Work

The works of (Unger et al., 2015) and (Clark et al., 2018) provide extensive systematic surveys on secure messaging and email covering numerous aspects. We limit ourselves to the decentralized setting without elaborating on the drawbacks of web of trust approaches covered in the above mentioned works.

The literature contains a sizeable body of work on OOB-based approaches, considered first by (Rivest and Shamir, 1984), many of which are inspired by the original work of (Vaudenay, 2005) based on SAS comparisons, e.g., (Nguyen and Roscoe, 2011; Kainda et al., 2009; Kainda et al., 2010; Tan et al., 2017), to name a few. This area has also been investigated by the formal methods community, see e.g. (Delaune et al., 2017) for a recent formal analysis of SAS-based schemes in the symbolic model.

As for low-entropy secret-based authentication, to the best of our knowledge, in the only work in the literature, (Alexander and Goldberg, 2007) use a modified version of a solution to SMP (Boudot et al., 2001), which is mainly suitable for synchronous settings, to improve authentication in OTR (Borisov et al., 2004).

2 FRAMEWORK AND PRELIMINARIES

We use $\mathcal{A}$ and $\mathcal{B}$ to refer to honest parties Alice and Bob, and $\mathcal{M}$ for the adversary, Mallory. We use $\leftarrow_\$$ to denote an element sampled uniformly at random, and $\|$ to denote concatenation. We denote low-entropy secrets provided by users with π.

Security Model. We consider the standard Dolev-Yao model (Dolev and Yao, 1981). We do not assume any trusted infrastructure. In one of our proposed methods for transport protocol, we assume the existence of untrusted buffer/relay servers, somewhat akin to the ones used in the design of Signal or OTR4 (see Section 4.3). Regarding PAKEs, we will consider various constructions in Section 4, largely proven secure in the so-called BPR model (Bellare et al., 2000) under various hardness assumptions.

Cryptographic Notions. For space reasons, we assume familiarity with common cryptographic concepts, in particular with Diffie-Hellman (DH)-based computational hardness assumptions.

We discuss schemes based on the Ring Learning With Errors (RLWE) problem, a special case of the Learning With Errors (LWE) problem whose security may be reducible to the hardness of solving the Shortest Vector Problem (SVP) in lattices, for which no efficient quantum algorithms are known, thus conjectured to be quantum-secure. Post-quantum (PQ) cryptography encompasses schemes that are considered to be safe against adversaries equipped with scalable, cryptographically relevant quantum computers.

We use $\mathsf{KDF}(s)$ to denote a key derivation function that takes a source s of keying material, typically with a fair amount of entropy but not uniformly distributed, and produces one or more cryptographically strong secret keys, see (Krawczyk, 2010) for details. We denote with $\mathsf{MAC}(k, m)$ a keyed message authentication code scheme that computes a tag on m under key k.

System Requirements. We assume standard requirements for email transfer as our proposal does not require any format modifications and preserves compatibility between existing systems. As for secure messaging, we do not introduce any extra trust assumptions and no additional infrastructure would be required. Any exchanges relayed or buffered by intermediate servers can be done by untrusted ones.

Socialist Millionaires' Problem. In the realm of secure multi-party computation (MPC), Yao's millionaires' problem (Yao, 1982) is a famous example in which two parties want to find out whose input is greater without revealing any more information on the actual value. SMP is a variant of this and a ZK proof of knowledge protocol, with the difference that the parties only wish to know if their inputs are equal.

A series of works have been dedicated to solving SMP, including a well-known solution by (Boudot et al., 2001) that provides a fair and efficient protocol, where fairness roughly means that no party can evaluate the function and walk away with the result without the other party learning the output.

(Garay et al., 2004) showed that the fairness and the security definition of (Boudot et al., 2001) are not compatible with the simulation paradigm and that their solution would not be secure when composed concurrently; they present a construction that can be composed arbitrarily, with similar complexity results.

PAKE. Password-authenticated key exchange (PAKE) protocols enable the establishment of secure channels without the need for a PKI, TTPs or empirical OOB channels. They allow two parties who share only a low-entropy secret, hereafter password, to agree on a cryptographically strong shared secret key, using the password for authentication.

Since the seminal work of (Bellovin and Merritt, 1992), numerous PAKE protocols have been proposed, which largely fall into the two categories of balanced (symmetric) and augmented (or asymmetric), referred to as aPAKE. The latter stores one-way mappings of passwords on the server side in client-server settings.

Intuitively, a core property of PAKE is that a run of the protocol should not leak any information about the password. Moreover, they should be resistant to offline dictionary attacks; an online guessing attack with at most one test per run should be the optimal attack strategy for an active $\mathcal{M}$ interacting with a party. Similar to SMP, $\mathcal{M}$ can mask failed guessing attempts as network failures, thus allowing numerous attempts without raising suspicion. This is in general unavoidable, however, we will see in Section 4 how a recent

work by (Roscoe and Ryan, 2017) can mitigate this.

3 PITFALLS IN OUT-OF-BAND AUTHENTICATION

In OOB authentication, users compare some representation of a cryptographic hash (fingerprint) of their partners' public keys via a separate authenticated channel. This representation is usually in the form of a list of words, numbers or images.

Strong security and usability properties can be achieved if users execute the manual verification correctly. Yet, the difficulty of having users do the assigned tasks correctly while finding the right balance between usability and security is the root cause of security pitfalls, which have been amply discussed by research on fingerprint and SAS comparison via OOB channels (see Section 1.1). Usability studies encourage the replacement of manual comparisons by automated software whenever possible (Tan et al., 2017).

Selection of an Adequate OOB Channel. In practice, the theoretical and strong authentication requirements of OOB methods are not easy to satisfy. While face-to-face conversations provide a strong authenticated channel (Nguyen and Roscoe, 2011), they are often not viable. It is usually assumed that an OOB channel cannot be forged, but it can be blocked, overheard, delayed or replayed. Typical instantiations are done via voice-based channels, e.g. a phone call. However, some already consider voice-based SAS comparison to be obsolete from a security perspective (Unger et al., 2015) as nowadays messages can be forged by voice synthesizers with a small sample of the victim's voice. Indeed, a voice impersonation attack on users comparing PGP words (Shirvanian and Saxena, 2014) reported the fake voice to be indistinguishable in about 50% of the cases.

Social Engineering Attacks. There are multiple ways for humans to interact via OOB, but with few indications about secure, privacy-preserving, or fair ways to do it, e.g., without knowing the authentication value, $\mathcal{M}$ can fool $\mathcal{A}$ by pretending to be $\mathcal{B}$, asking her to read the words first, and confirming a full match.

Inattentive and Lazy Users. Users misreading words (inattentive) or comparing only subsets of them (lazy). A recent paper by (Naor et al., 2018) analyzes approaches based on SAS authentication that are vulnerable to MITM attacks w.r.t. lazy users. For instance, the approach in WhatsApp and Signal would be flawed if users compared only either the first or the second half of the value, since it would amount to verifying only one peer's fingerprint. To fix this, the authors propose an influence spreading technique in which every bit of the value to be authenticated influences the generation of each element of the OOB representation.

Partial Preimage Attack. (Dechand et al., 2016) study an attack aimed at finding a partial preimage for a fingerprint verified by lazy users; specifically, they assume that subsets of bits at the boundaries and in the middle are checked. Let p denote the probability of finding a partial preimage for a given fingerprint f and q its complementary event. To calculate $p = 1 - q$, we work out q (i.e., the absence of partial preimages for a specific bit permutation). Let b be the length of the fingerprint f and assuming that r consecutive boundary bits are fixed (checked by the user), in this case, the leftmost and rightmost bits of f, we let ℓ denote the number of remaining bits in the middle from which a possible variation of u bits could be fixed, i.e., checked by the user. Thus, we have $2 \cdot r + u$ fixed bits that the adversary cannot invert without the user noticing. Valid preimages can thus be obtained by flipping up to $t = \ell - u$ bits within the middle bits; by removing these from the total space of size 2^b, we obtain the number of invalid ones. With k denoting a given number of positions to modify, the valid strings are then given by $\binom{\ell}{k}$ choices of positions to flip. Thus, q is given by

$$q = \frac{2^b - \sum_{k=1}^{t} \binom{\ell}{k}}{2^b}. \tag{1}$$

Expressing p as a function of the computational effort in terms of e brute-force attempts, we have $p = 1 - q^e$. To estimate the number of steps needed for finding partial preimages with a success probability $\geq p$, we simply compute $e = \log_q(1 - p)$. Expressing e in base 2 gives results comparable to (Dechand et al., 2016).

3.1 Case Study

Pretty Easy Privacy (p≡p) is a software aimed at providing usable privacy-by-default in email via end-to-end opportunistic encryption. The tool largely automates initial key generation and storage. The public key of a user is attached to outgoing emails when a key of the recipient has not been stored. Received keys are automatically stored for future use (*trust-on-first-use*) and outgoing emails are automatically encrypted when a public key of the intended receiver is available. This approach requires neither a PKI nor a TTP.

Similar to the PGP word list, *p≡p trustwords* (Birk et al., 2019) are natural language words that two users compare via a low-bandwidth OOB authenticated channel to prevent man-in-the-middle (MITM) attacks. The trustwords generation algorithm tws($\cdot$) is a deterministic algorithm that runs locally taking as input the public key of the peer obtained by email and the user's own public key. Informally, tws($\cdot$) performs

an XOR over the fingerprints of each of the input arguments, and then maps each block of 16 bits from the resulting 160-bit long string to a word in a predefined dictionary of size 2^{16}, thus yielding a list of ten words.

To encourage users to perform the OOB authentication, by default p≡p shows only five words; this means that the peers compare the first 80 out of the 160 bits of a PGP fingerprint, assuming that they check all the words. Since an "influence spreading" property, similar to Naor et al.'s, is already present, the best adversarial strategy is a brute-force attack over the public key space requiring $O\left(2^{80}\right)$ steps to find a key k such that the first 80 bits of $\mathsf{fpr}(k)$ are equal to those of $\mathsf{fpr}(\mathsf{pk}_B)$, with pk_B being the public key of $\mathcal{B}$.

We consider lazy users and compute estimates for partial preimage attacks similar to the one presented above. We consider the two cases where, out of five words, the user verifies (*i*) the first and last words as well as two from the middle (*ii*) the first and last words, along with one of the three in the middle. Let $b = 80$, $\ell = 48$ and for (*i*) we have $u = 32$ and we get $e \approx 2^{38}$; for (*ii*), with $u = 16$, we get $e \approx 2^{32}$. These results show that $\mathcal{M}$ would succeed with costs equal to and lower than the computational power estimated for an average adversary (Dechand et al., 2016).

Clearly the decision to show five words instead of ten by default needs to be reconsidered. Users might feel less annoyed by having to compare fewer words, however, its adverse effect on security is considerable as it practically renders brute-force attacks viable.

4 AUTHENTICATION IN EMAIL AND MESSAGING VIA PAKE

We now show how PAKE can be used to perform a secure equality test and thereby authentication. Compared to OTR that uses a modified solution to the SMP protocol, we show that our PAKE-based approach yields a more efficient solution with better security guarantees and enables further cryptographic features. **Trust Establishment using Low-entropy Secrets.** For $\mathcal{A}$ and $\mathcal{B}$ to mutually authenticate, for now we assume that they share a low-entropy secret—e.g., a short password—either agreed upon beforehand or decided by posing and answering a question. Intuitively, the goal is to perform a secure equality test such that upon termination of the protocol, $\mathcal{A}$ and $\mathcal{B}$ would only learn whether or not their respective secrets π_A and π_B were the same, thus authenticating each other on the basis of knowing the same secret.

In other words, $\mathcal{A}$ and $\mathcal{B}$ wish to authenticate their public keys via a secure equality test of their secrets without revealing any information about the latter,

hence the need for a zero-knowledge protocol. This means that the resulting transcript of their exchanges should not leak any information on π_A and π_B to $\mathcal{M}$. Also, it should not be possible for $\mathcal{M}$ to brute-force the password via offline dictionary attacks. Thus, $\mathcal{M}$'s only strategy would amount to making online attempts.

4.1 Public Key Authentication via PAKE

To determine at the end of a PAKE run whether the user secrets π_A and π_B are equal, without revealing anything else, we suggest the enforcement of explicit authentication using key confirmation (KC) after the key establishment phase. While this step may be optional in the general case for PAKE protocols, here it would be necessary in order to bind the cryptographic material with an identity. The information that $\mathcal{A}$ and $\mathcal{B}$ wish to authenticate—e.g., public keys for email addresses in p≡p or key fingerprints for phone numbers in Signal—can be incorporated either into the KC phase or into the initial user secrets.

Next we show using a concrete example how this can be constructed. For the moment, we do not focus on engineering aspects related to (a)synchronicity and message transport mechanisms, but we will come back to these in Section 4.3. The literature contains several well-studied instances of PAKE and for this reason, we first pick a candidate to demonstrate how it can be used for public key authentication, and then compare a few prominent schemes according to specific properties of interest, as shown in Table 1.

4.1.1 An Instantiation based on SPAKE2

For illustration, in Figure 1 we propose an extension of SPAKE2, a one-round protocol, with a KC step to achieve explicit authentication, thus binding a public key to an entity. This yields a 2-round scheme, the minimum when KC is enforced; see (Katz and Vaikuntanathan, 2011) for optimal-round PAKEs. For KC we can use the generic refresh-then-MAC transformation. Despite its long history and popularity, this transform was only recently proved secure (Fischlin et al., 2016).

With $\mathbb{G}$ being a finite cyclic group of prime order p, generated by an element g, let $\mathbb{G}, g, p, M \leftarrow_\$ \mathbb{G}, N \leftarrow_\$ \mathbb{G}$ and hash function $H(\cdot)$ denote public parameters and $\pi \in \mathbb{Z}_p$ the private low-entropy secret, with the user password assumed to be appropriately mapped to an element in $\mathbb{Z}_p$. The parties perform the key exchange phase, as shown in Figure 1, which concludes with the generation of a symmetric key. Upon termination of the key establishment, $\mathcal{A}$ and $\mathcal{B}$ each use the symmetric

key to carry out a key-refreshing step via a key derivation function in order to generate fresh MAC keys (for both parties), along with a new session key, K, which will be the secret shared key. Next, under the freshly generated keys, they each compute a MAC on the fingerprints of both parties' public keys. The authentication now amounts to exchanging and verifying the obtained tags τ^a and τ^b.

The addition of the KC step increases the number of rounds and flows to 2 and 4, respectively. Note that this is merely an illustrative example and as already mentioned, other possibilities for KC do exist, some of which offer additional properties. For instance, (Becerra et al., 2018) showed that a modified version of SPAKE2, called PFS-SPAKE2, coupled with a KC step can achieve perfect forward secrecy (PFS) at the cost of increasing the number of rounds from 1 to 3. More recently, (Abdalla and Barbosa, 2019) showed that SPAKE2 does indeed satisfy PFS even without KC under a different hardness assumption. They also prove a version with a KC step (yielding a better bound) almost identical to the one given in Figure 1, except that the protocol has one less flow.

Alternatively, the public key fingerprints can be embedded in the secret π, but note that even in that case, the KC step cannot be skipped as an explicit authentication of the public keys would still be needed. More precisely, we would let $\pi = \pi' \| \mathsf{fpr}(\mathsf{pk}_{\mathcal{A}}) \| \mathsf{fpr}(\mathsf{pk}_{\mathcal{B}})$, where π' denotes the original user provided secrets, and we would compute the tags as $\tau^a \leftarrow \mathsf{MAC}(k^a_{\mathrm{MAC}}, \mathsf{sid})$, where the identifier sid is computed over the transcript, with τ^b computed similarly. Similar one round KC methods for explicit authentication can be found in IETF internet-drafts for SPAKE2[1] and J-PAKE[2].

4.1.2 Choice of PAKE

We consider a number of representative PAKE protocols and analyze their properties w.r.t. our use case: SPAKE2 (Abdalla and Pointcheval, 2005), OPAQUE (Jarecki et al., 2018), PFS-SPAKE2 (Becerra et al., 2018), J-PAKE (Hao and Ryan, 2010), KV-SPOKE (Katz and Vaikuntanathan, 2011), RLWE-PAK and PPK (Ding et al., 2017). PAKEs are typically evaluated according to the security model in which they are proven secure, support for forward secrecy, the number of rounds, along with their communication and computational complexity. The complexity related aspects become more relevant in a client-server setting wherein a server has to process a high number of requests and sessions. In a decentralized peer-to-peer setting, such properties no longer play a major role.

---

[1] https://tools.ietf.org/id/draft-irtf-cfrg-spake2-08.html
[2] https://tools.ietf.org/html/rfc8236

Table 1: Comparison of PAKE protocols.

Protocol	Rounds/ Flows	KC	FS	Security model	Hardness assump.
SPAKE2	1/2	✗	✓	ROM	CDH
PFS-SPAKE2	3/3	✓	✓	ROM	CDH
OPAQUE	2/3	✓	✓	ROM	OMDH
J-PAKE	2/4	✗	✓	ROM-AAM	DSDH
KV-SPOKE	1/2	✗	-	CRS	DDH
RLWE-PAK	3/3	✓	✓	ROM	RLWE
RLWE-PPK	2/2	✗	✓	ROM	RLWE

ROM: Random Oracle Model; AAM: Algebraic Adversary Model; CRS: Common Reference String
DH: Diffie-Hellman; CDH: Computational DH; DDH: Decisional DH; DSDH: Decision Square DH; OMDH: One-More DH; RLWE: Ring Learning With Errors

In Table 1, we present some of the relevant properties of the said constructions. Note that except for RLWE-PAK and RLWE-PPK that make use of lattice-based cryptography, all other schemes are Diffie-Hellman-based. In terms of PQ security, an immediate implication of this is that the latter cases would not be safe against quantum adversaries, whereas the first two would provide conjectured quantum-security due to the underlying RLWE problem.

Minimizing the number of rounds becomes more important for secure email than for messaging, especially if the transport mechanism is based on attachments or hidden emails (see Section 4.3). As for secure messaging, this may be equally relevant for solutions that do not operate in a purely decentralized and peer-to-peer setting in which one may wish to reduce the load on relay or buffer servers, e.g., Signal or OTR4, but the number of rounds would in general be arguably less of a concern. Note that KC can be added to schemes that do not have it by default at the cost of an extra round.

Intuitively, the notion of forward secrecy (FS) captures the requirement that a long-term secret compromise should not result in prior session keys getting compromised and consequently the corresponding exchanges. Weak FS (wFS) refers to those schemes satisfying FS against passive adversaries who did not interfere in the previous sessions and perfect FS to those achieving the same against active adversaries. We will come back to this in Section 4.2.2.

We limit the discussion on security models to practical considerations. In the random oracle model (ROM), an ideal truly random function being accessible to the parties through oracle calls is typically instantiated using cryptographic hash functions, and the common reference string (CRS) model implies the accessibility of a random string to all parties, generated in a trusted way. The latter may be less obvious to implement in the case of email due to the constraints of decentralization given that the generation of the CRS

Public parameters: $\mathbb{G}, g, p, M \leftarrow_s \mathbb{G}, N \leftarrow_s \mathbb{G}, H$; private parameter: $\pi \in \mathbb{Z}_p$ the low-entropy secret.

Alice	**Bob**

$x \leftarrow_s Z_p; X \leftarrow g^x$ $y \leftarrow_s Z_p; Y \leftarrow g^y$

$X^* \leftarrow X \cdot M^\pi$ $Y^* \leftarrow Y \cdot N^\pi$

$$\xrightarrow{\quad A, \mathsf{pk}_A, X^* \quad}$$

$$\xleftarrow{\quad B, \mathsf{pk}_B, Y^* \quad}$$

$K_A \leftarrow \left(\dfrac{Y^*}{N^\pi}\right)^x$ $K_B \leftarrow \left(\dfrac{X^*}{M^\pi}\right)^y$

$\mathsf{sk}_A \leftarrow H(A, B, X^*, Y^*, \pi, K_A)$ $\mathsf{sk}_B \leftarrow H(A, B, X^*, Y^*, \pi, K_B)$

.... Key exchange complete; key confirmation and binding follow ....

$(K, k_{\mathrm{MAC}}^a, k_{\mathrm{MAC}}^b) = \mathsf{KDF}(\mathsf{sk}_A)$ $(K, k_{\mathrm{MAC}}^a, k_{\mathrm{MAC}}^b) = \mathsf{KDF}(\mathsf{sk}_B)$

$\tau^a \leftarrow \mathsf{MAC}(k_{\mathrm{MAC}}^a, \mathsf{fpr}(\mathsf{pk}_A)\|\mathsf{fpr}(\mathsf{pk}_B)\|\mathsf{sid})$ $\tau^b \leftarrow \mathsf{MAC}(k_{\mathrm{MAC}}^b, \mathsf{fpr}(\mathsf{pk}_A)\|\mathsf{fpr}(\mathsf{pk}_B)\|\mathsf{sid})$

$$\xrightarrow{\quad \tau^a \quad}$$

$$\xleftarrow{\quad \tau^b \quad}$$

Abort if τ^b invalid; else output key K Abort if τ^a invalid; else output key K
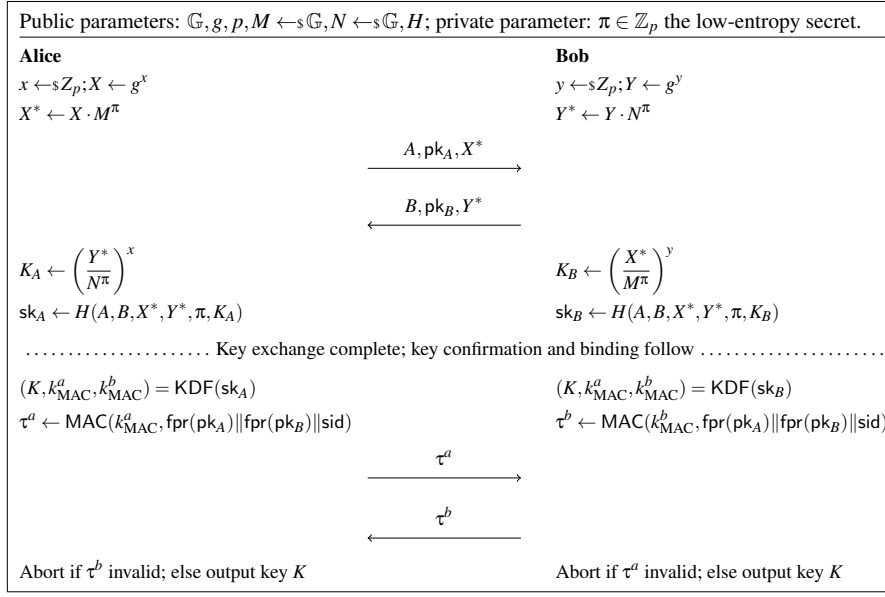

Figure 1: pk authentication using SPAKE2 with refresh-then-MAC key confirmation for entity binding.

would be typically done by a trusted party or via a secure MPC protocol, see e.g., (Sasson et al., 2014) for an example of CRS generation in a decentralized setting. Finally, regarding the RLWE-based schemes, their proofs are unfortunately in the ROM, as opposed to the quantum ROM (QROM), which would allow adversaries to query the random oracle in superposition.

4.2 Cryptographic Enhancements

We first show how a number of key properties related to key management automation and error resilience that have been identified in the literature (Unger et al., 2015) are satisfied and improved upon by our approach. We then present novel uses of PAKEs in secure email and messaging. Note that once a PAKE-generated key is established, subsequent PAKE instances can be run automatically via a chaining self-sustaining mechanism. While we mainly focus on enhancements for existing paradigms that depend on public-keys, we also consider possibilities for shifting to entirely symmetric-key solutions. Indeed, once a PAKE-generated shared symmetric key has been established, not only a wide range of well-understood techniques become possible, but one could also consider the benefits of transitioning to symmetric-key constructions, e.g., MAC-based authentication and symmetric-key encryption schemes.

In Table 2, we compare our proposal with a select set of approaches extracted from (Unger et al., 2015). Due to space reasons, we refer the reader to the cited source for details on the properties therein.

4.2.1 Key Management Automation

Automation of Future Key Pair Authentications. This is the underlying feature facilitating the achievement of some of the subsequent properties. Once authentication between $\mathcal{A}$ and $\mathcal{B}$ is bootstrapped from an initial PAKE, the authentication of new key pairs from either $\mathcal{A}$ or $\mathcal{B}$ can be automated using the PAKE-generated shared keys without prompting the users to yet again enter new secrets. For instance, in the case of email (e.g., p≡p), authentication due to key pair generations can be triggered whenever a new key pair needs to be associated with an existing identity, or for binding a new email of $\mathcal{A}$ or $\mathcal{B}$ to a new key pair or when keys expire. Note that upon each future authentication, the PAKE-generated symmetric keys can be refreshed by automatically carrying out a new PAKE.

Immediate Enrolment. This property refers to a user's keys being reinitialized in such a way that other parties can verify and use them immediately. The PAKE-generated key allows to automate the new key exchange and the corresponding authentication.

Alert-less Key Renewal. Complementing the previous property, this one refers to users not receiving alerts or warnings prompting them to take action when other parties renew their public keys. This would be automated similarly to immediate enrolment.

Low Key Maintenance. This property pertains to the amount of user effort required for maintaining keys, e.g., tasks such as signing keys, renewing expired keys. For instance, while the p≡p client does automate key generation and renewal, the established trust le-

vel disappears with every key refreshment; key maintenance can be improved with PAKEs as explained above.

Multi-device Syncing. Another quite natural application of PAKE is in the realm of device pairing and multi-device synchronization. These typically rely on a human interactive security protocol (HISP) and OOB techniques requiring manual intervention, which can give rise to new and subtle attacks. The application of PAKEs in other contexts for device pairing has been considered before; it is thus natural to consider incorporating them in multi-device syncing of email agents and secure messaging systems.

$\mathcal{A}$ can enter a password in both devices to be paired, D_1 and D_2, triggering a PAKE protocol that establishes a secure channel between them for synchronization; alternatively, this can even be done asynchronously without the two devices being online: D_1 pushes its state (e.g., key store, chat or email archive) to a server in encrypted form and later D_2 retrieves the secrets stored on the server in an oblivious manner w.r.t. the server, see Section 4.2.3 for more details. For example, the current implementation of p≡p resorts to an ad-hoc pairing technique for key synchronization that could benefit from such a PAKE-based solution.

Inattentive User Resistance. As discussed earlier, manual OOB key/fingerprint verification methods are susceptible to human error and inattentiveness. In the PAKE-based approach, even if the users enter the wrong value, the result would not be as catastrophic as trusting a key prepared by the adversary. At worst, it would be inconvenient as the authentication would fail prompting the user to eventually repeat the process.

4.2.2 Cryptographic Properties

Perfect Forward Secrecy (PFS). Once, more popular in the context of secure messaging (e.g., Signal and OTR), PFS is now a requirement for cipher suites supported in TLS 1.3. PFS means that in the event of a password disclosure, prior derived session keys remain secure. Clearly, a compromised PAKE-generated key would have to be discarded and refreshed via a new PAKE instance. A PAKE-chaining mechanism that automatically performs key rotations and periodically refreshes the long-term key would provide limited windows of opportunity for $\mathcal{M}$, after which the resulting key would be secure again.

Several PAKE constructions provide PFS by default, some of which are listed in Table 1; it is known that PFS can be obtained by adding explicit authentication via a KC step (Bellare et al., 2000). This paradigm would be more relevant when such PAKE-based approaches are used for synchronization purposes, be it device-to-device or device-to-server where PAKE

can be used to both authenticate and establish a secure channel, thus providing PFS for the session keys used for syncing. For more efficiency, a symmetric-key scheme with PFS such as SAKE (Avoine et al., 2020) can be bootstrapped using PAKE.

Finally, the approach adopted by the Sequoia-PGP project for adding FS to OpenPGP-based solutions using regular sub-key rotations would also benefit from automated authentication in case the master key, certifying the short-term sub-keys, needs to be refreshed and authenticated. For additional security, with slightly hampered usability, a separation of storage can be enforced by for example storing such PAKE long-term keys in dedicated hardware, e.g., hardware security modules or smart key storage devices such as YubiKey or Nitrokey, to protect against a device compromise; see Section 4.2.3 for more details on this.

Deniability. This is another subtle and fundamental property that has been of particular interest in recent secure messaging systems such as Signal and OTR. Deniable exchange, applied to tasks ranging from authentication to encryption, has a long and somewhat controversial history due to the subtleties in various existing security definitions. We limit ourselves to the case of key exchange and the seminal framework of (Di Raimondo et al., 2006) providing security definitions in the simulation paradigm for deniable key exchange and authentication in which message and participation repudiation are considered as requirements.

Since limited space does not allow us to elaborate, we consider only sender/receiver deniability for non-augmented PAKE, i.e., symmetric. We conjecture that such a construction would satisfy the said definition of deniability in the symmetric-key setting: in a two-party setup, a malicious party $\mathcal{M}$ would not be able to produce binding cryptographic proofs from communication transcripts, associating a party with a particular exchange, as all exchanges could have been simulated by the accusing party $\mathcal{M}$. We now observe that a simulator can be constructed as π is the only private input shared by both parties and all other parameters are public and drawn at random. Finally, assuming composability, using the PAKE-generated key with symmetric ciphers and MAC-based authentication would preserve deniability. Clearly, this and other forms of deniability for PAKE need to be studied rigorously in future work.

Post-quantum Security. As pointed out in Section 4.1.2, in the event that secure messaging and email tools transition to PQ cryptography, there are candidate PAKE constructions that provide conjectured PQ security; see Table 1. Moreover, a PQ-secure PAKE can be combined with the recent symmetric-key authenticated key exchange (SAKE) by (Avoine et al., 2020) that

provides PFS to obtain an efficient, PQ-secure and primarily symmetric key scheme with PFS. A quantum-resistant PAKE can be used once to bootstrap authentication via low-entropy secrets and to provide the initial master key needed by SAKE, which is conjectured to be PQ-secure due to its use of symmetric-key primitives, thus offering a low cost and efficient PQ-AKE suitable for settings with limited computational power.

4.2.3 Secure Secret Sharing and Retrieval

OPAQUE is a recent construction that can, among other things, serve as an aPAKE to offer protection against breaches and server password file compromises. It also offers a secret retrieval mechanism, based on oblivious pseudo-random functions, to retrieve a secret from a server, stored in encrypted form, using only a low-entropy password.

This feature is inspired by the notion of password-protected secret sharing (PPSS) schemes formalized by (Bagherzandi et al., 2011), which are (t, n)-threshold constructions wherein security is preserved against an adversary controlling up to t servers out of n. A problem that PPSS addresses is protecting $\mathcal{A}$'s secret data d (e.g., cryptographic secret key used for decryption, authentication credentials, etc) in the event of a device compromise.

Such a scheme would secret-share d among a set of n agents so that only a collusion of more than t corrupt ones would compromise the data. Secret-sharing is combined with a password-based mechanism that allows the authentication of the owner of d to the secret-share holders in order to trigger a reconstruction protocol and retrieve the secret. The private storage of d can be shared among n external network entities to protect against user device compromise. Alternatively, if $\mathcal{A}$ does not trust external entities, her device can partake in the secret-sharing by storing multiple shares instead of any other external entity, thus preventing online dictionary attacks by a network attacker and not allowing $\mathcal{M}$ to learn anything about the secret without corrupting $\mathcal{A}$'s device.

Secret retrieval would have several use cases in secure messaging. For instance, instead of retrieving contacts from the user's phone, servers could store lists of contacts in encrypted form; this would enable asynchronous syncing of contacts across multiple devices without the service provider learning the content. A general anonymity/privacy related criticism directed at messaging services has to do with the identification of users via their phone numbers. This can be dealt with by securely storing long-term identities in encrypted form on the server, accessible only to the users.

Another use case would be to secret-share user data among several of their own devices, e.g., smartphone, laptop and tablet, so that a device compromise would not provide any useful information to an attacker; this can also be used for performing key synchronization among multiple devices. All these mechanisms would work in a similar manner from the user's point of view, i.e., simply by providing a password.

4.2.4 Auditable PAKEs for Thwarting Online Guessing Attacks

As is the case for SMP in OTR, online guessing attacks are unavoidable in PAKEs. This is usually dealt with by fixing a limit on the number of failed attempts that can be tolerated before invalidating a password.

However, in certain cases, another subtle adversarial strategy aimed at sidestepping the (at most) one online test per run would be to resort to a class of guess and abort attacks in which $\mathcal{M}$ intercepts a message in a given session (or initiates a session of her own) at a crucial step of a protocol run, verifies her guess at the password and in case of an incorrect guess, drops the said message to disguise her attempt as a network communication failure.

This can be done in both directions to double the chance of discovering the password, or in parallel against many network nodes depending on the setting. Such an attack can be carried out repeatedly without raising an alarm as the honest parties may simply view this as a network failure.

We identify a similar vulnerability in the use of a modified version of SMP in OTR: just before the last phase where the parties perform their secure equality test, when $\mathcal{A}$ and $\mathcal{M}$ exchange their blinded DH terms incorporating the low-entropy password in the exponent, i.e., $(g_3^a, g_1^a g_2^{\pi_{\mathcal{A}}})$, $\mathcal{M}$ could make a guessing attempt at $\pi_{\mathcal{A}}$ and in case of obtaining 0 (not equal), drop the message and force an abort, see sections 4.2 and 4.3 in (Alexander and Goldberg, 2007). Note that the non-interactive zero-knowledge (NIZK) proofs that are attached to the messages at every exchange are not meant to protect against this type of attack.

In a relatively recent work, (Roscoe and Ryan, 2017) apply a mechanism based on commitment schemes and delay functions (e.g., timed-release encryption), originally developed by (Roscoe, 2016) for protecting against online attacks in HISPs that use SAS, to the setting of PAKEs in order to make them auditable by achieving *stochastic fair exchange*.

Roughly speaking, this is achieved by a transformation for PAKEs at the level of KC using a combination of blinding, randomization, commitments and delay functions such that a series of messages consisting of fake ones and the real intended message are exchanged and the parties will only get to know which is the *right* one until their exchange is complete. In a follow-up

Table 2: Comparison of trust establishment approaches.

Paradigm	Example	Security									Usability									Adoption		
		Network MiM Prevented	Operator MiM Prevented	Operator MiM Detected	Operator Accountability	Key Revocation Possible	Privacy Preserving	Deniability Facilitated	Forward Secrecy Facilitated	Post-quantum Security	Automatic Key Initialization	Low Key Maintenance	Easy Key Discovery	Easy Key Recovery	In-Band	No Shared Secrets	Alert-less Key Renewal	Immediate Enrollment	Inattentive User Resistant	No Service Provider	Asynchronous	Multiple Key Support
Web of Trust	PGP	●	●	●	◐	◐	◐	✗	✗	-	✗	✗	◐	◐	◐	✗	✗	✗	✗	●	●	●
KD + SaL	CONIKS	●	✗	◐	●	●	●	✗	-	✗	●	●	●	●	●	●	●	●	●	●	●	●
OE + SMP	OTR	◐	◐	◐	◐	●	●	-	-	✗	✗	●	✗	✗	◐	◐	✗	●	✗	●	✗	●
OE + TOFU	TextSecure	◐	◐	◐	◐	✗	●	-	-	-	●	●	●	●	●	●	✗	●	✗	●	●	✗
OE + TOFU + OOB	p≡p	◐	◐	◐	●	◐	●	-	-	-	●	◐	●	◐	✗	●	✗	✗	✗	●	✗	✗
OE + TOFU + PAKE	-	◐	◐	◐	●	◐	◐	⊗	⊗	⊗	●	●	●	●	●	✗	●	●	●	●	●	●
KFV + OOB	SilentText	●	●	●	●	◐	●	-	-	-	✗	✗	✗	✗	✗	●	✗	✗	✗	✗	✗	✗
KFV + PAKE	-	●	●	●	●	◐	●	●	●	⊗	●	●	●	●	●	✗	●	●	●	●	●	●

Paradigm property is: ● = satisfied; ◐ = partially satisfied; ✗ = not satisfied; ⊗ = implementation dependent; - = N/A

KD = Key directory; KFV = Key fingerprint verification; OE = Opportunistic encryption; SaL = Self-auditable logs; TOFU = Trust-on-first-use

work, (Couteau et al., 2019) generalize this result to achieve ε-fair exchange using oblivious transfer and timed-release encryption.

This transformation can be used to enhance any PAKE with auditability, thus lending itself quite naturally to the authentication method suggested in this work. An important limitation here is that, due to the highly interactive design of the solution, it would be more suitable to the setting of secure messaging than email, unless a given email solution were to opt for untrusted buffer servers for transport, see Section 4.3.

Finally, note that some of the ideas in this transformation, specifically those related to enforcing fairness, have common elements with the original SMP (Boudot et al., 2001) solution aimed at providing fairness, a property that was removed from the modified version of SMP used in OTR (Alexander and Goldberg, 2007) on account of achieving efficiency.

4.3 Transport Mechanism

Email-based Approach. Given the small number of rounds required by PAKE protocols, in the case of email we can afford to use standard email attachments or specially formatted hidden emails as messages' carriers, processed by the email client in the background.

$\mathcal{A}$ can choose (via an interface option) to enter her secret $\pi_{\mathcal{A}}$ upon sending her first email, allowing the first flow of the protocol to occur via an attachment; similarly, when $\mathcal{B}$ replies, if he opts for entering his secret $\pi_{\mathcal{B}}$, the initial PAKE round would be done; the subsequent KC can be done automatically by the dedicated software.

Alternatively, one can resort to a hidden email transport model such as the one used by p≡p for multi-device key synchronization. Here, the implementations would encapsulate crypto messages in specially crafted email attachments, kept hidden from the user (e.g., archived separately) and processed automatically. Since we primarily deal with authentication, our proposal would have minimal impact in terms of communication and computational complexity as it would have to take place only once.

Untrusted Server Approach. Although early IM tools were entirely online services that maintained an active session for each conversation, modern IM tools are in fact quite similar to email in that the underlying system follows an asynchronous model. Both Signal and the latest version of OTR (OTRv4-development, 2019) achieve offline messaging by using "buffer servers" for hosting pre-key bundles that can be fetched without the other party being online.

We can use a similar mechanism to overcome transport engineering obstacles in email more elegantly, since all aspects related to the exchange of emails remain unchanged and thus interoperable. In fact, the use of an intermediate server would not introduce additional trust assumptions as the transcript of a PAKE protocol does not leak useful information to the adversary; such a server would be untrusted and any entity would be able to set up their own instance.

4.4 Security and Low-entropy Secrets

The schemes considered thus far come with proofs of security, see Table 1 for the corresponding mod-

els and assumptions. The security guarantees can be traced back to the core properties of PAKEs: they can in effect fulfill the role of ZK proof of knowledge schemes such that a run of the protocol does not leak any information on the password and upon termination only reveals whether the secrets were equal; they resist offline dictionary attacks and online ones by limiting active adversarial tests to one password per run; compromised session keys will not compromise the security of other session keys; depending on the choice of PAKE; FS would ensure that past session keys remain secure if the password is leaked.

The only way for $\mathcal{M}$ to gain knowledge about the secret would be via active online guessing attempts, typically dealt with by fixing a limit on the number of failed attempts, e.g., SMP in OTR. We discussed how the possibility of making PAKEs auditable can be used to mitigate this class of attacks by distinguishing between failed adversarial attempts and network failures to minimize the adversary's tries to one, under the assumption of correct input entry by honest users.

Low-entropy Secret Agreement. Our proposal does come with its own caveat, namely the need for either presharing or agreeing on a low-entropy secret in-band. As already discussed in (Alexander and Goldberg, 2007), the users can either share a secret over a secure channel, e.g. OOB, or agree on one via an in-band solution without revealing sensitive information about the secret itself, for instance, $\mathcal{A}$ asking $\mathcal{B}$ to use the name of their favorite restaurant. The user interface of a tool implementing this could warn users not to include the secret itself, similar to standard email warnings reminding users to attach documents in case they have mentioned it in the body of the message.

Another possibility would be to use another already authenticated and secure channel to agree on a secret. For instance, given the widespread use of tools such as Signal, the parties could simply use it to agree on a secret for a one-time entity authentication of their secure email solution. While it may not be appealing from a theoretical point of view, due to the assumption of there being an already authenticated and secure channel, practically speaking, this approach would in fact provide a realistic and usable solution.

Usability Aspects. Implementations of the approach must pay proper attention in providing an adequate interface for entering the low-entropy secret, in addition to the usual considerations for providing easy explanations and documentation for users. A lesson learned from a usability study on the OTR/SMP tool (Stedman et al., 2008) stresses the need for further research on how to guide users towards establishing a secure shared human-memorable secret.

For instance, adding a list pre-populated with questions might serve as a guide to generate similar ones or reduce user effort by allowing them to choose one from the list; the questions should not lead to evident answers or to answers belonging to very small known sets, such as "yes/no" or colors, as such cases increase the successful guessing probability of the adversary. Another measure for dealing with disparities due to letter cases would be to just convert the secret to uppercase, at the cost of reducing entropy.

5 FUTURE WORK

We foresee as a next step an implementation of our proposal, along with research on usability dedicated to assisting users with deriving low-entropy secrets, reducing mental effort and the likelihood of mistakes. We also consider an analysis of our approach applied to encrypted mailing lists. Furthermore, we expect follow-up theoretical work on all the suggested cryptographic enhancements and implementations thereof.

ACKNOWLEDGEMENTS

REFERENCES

Abdalla, M. and Barbosa, M. (2019). Perfect forward security of SPAKE2. Cryptology ePrint Archive, Report 2019/1194. https://eprint.iacr.org/2019/1194.

Abdalla, M. and Pointcheval, D. (2005). Simple password-based encrypted key exchange protocols. In *Cryptographers' track at the RSA conference*, pages 191–208. Springer.

Alexander, C. and Goldberg, I. (2007). Improved user authentication in off-the-record messaging. In *Proceedings of the 2007 ACM workshop on Privacy in electronic society*, pages 41–47. ACM.

Avoine, G., Canard, S., and Ferreira, L. (2020). Symmetric-key authenticated key exchange (SAKE) with perfect forward secrecy. In *Cryptographers' Track at the RSA Conference*, pages 199–224. Springer.

Bagherzandi, A., Jarecki, S., Saxena, N., and Lu, Y. (2011). Password-protected secret sharing. In *Proceedings of the 18th ACM conference on Computer and Communications Security*, pages 433–444.

Becerra, J., Ostrev, D., and Škrobot, M. (2018). Forward secrecy of SPAKE2. In *International Conference on Provable Security*, pages 366–384. Springer.

Bellare, M., Pointcheval, D., and Rogaway, P. (2000). Authenticated key exchange secure against dictionary attacks. In *International conference on the theory and applications of cryptographic techniques*, pages 139–155. Springer.

Bellovin, S. M. and Merritt, M. (1992). Encrypted key exchange: password-based protocols secure against dictionary attacks. In *1992 IEEE Computer Society Symposium on Research in Security and Privacy*, pages 72–84. IEEE Computer Society.

Birk, V., Marques, H., Hoeneisen, B., and pEp Foundation (2019). Iana registration of trustword lists. https://tools.ietf.org/html/draft-birk-pep-trustwords-03.

Borisov, N., Goldberg, I., and Brewer, E. (2004). Off-the-record communication, or, why not to use PGP. In *Proceedings of the 2004 ACM workshop on Privacy in the electronic society*, pages 77–84.

Boudot, F., Schoenmakers, B., and Traore, J. (2001). A fair and efficient solution to the socialist millionaires' problem. *Discrete Applied Mathematics*, 111:23–36.

Clark, J., van Oorschot, P. C., Ruoti, S., Seamons, K., and Zappala, D. (2018). Securing email. *arXiv preprint arXiv:1804.07706*.

Couteau, G., Roscoe, A. W., and Ryan, P. Y. A. (2019). Partially-fair computation from timed-release encryption and oblivious transfer. Cryptology ePrint Archive, Report 2019/1281. https://eprint.iacr.org/2019/1281.

Dechand, S., Schürmann, D., Busse, K., Acar, Y., Fahl, S., and Smith, M. (2016). An empirical study of textual key-fingerprint representations. In *25th {USENIX} Security Symposium*, pages 193–208.

Delaune, S., Kremer, S., and Robin, L. (2017). Formal verification of protocols based on short authenticated strings. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 130–143. IEEE.

Di Raimondo, M., Gennaro, R., and Krawczyk, H. (2006). Deniable authentication and key exchange. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 400–409.

Ding, J., Alsayigh, S., Lancrenon, J., Saraswathy, R., and Snook, M. (2017). Provably secure password authenticated key exchange based on RLWE for the post-quantum world. In *Cryptographers' Track at the RSA Conference*, pages 183–204. Springer.

Dolev, D. and Yao, A. C. (1981). On the security of public key protocols. In *Proceedings of the 22Nd Annual Symposium on Foundations of Computer Science*, SFCS '81, pages 350–357. IEEE Computer Society.

Fischlin, M., Günther, F., Schmidt, B., and Warinschi, B. (2016). Key confirmation in key exchange: A formal treatment and implications for TLS 1.3. In *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE.

Garay, J. A., MacKenzie, P. D., and Yang, K. (2004). Efficient and secure multi-party computation with faulty majority and complete fairness. *IACR Cryptology ePrint Archive*, 2004:9.

Hao, F. and Ryan, P. Y. A. (2010). J-PAKE: authenticated key exchange without PKI. In *Transactions on computational science XI*, pages 192–206. Springer.

Jarecki, S., Krawczyk, H., and Xu, J. (2018). OPAQUE: an asymmetric PAKE protocol secure against pre-computation attacks. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 456–486. Springer.

Kainda, R., Flechais, I., and Roscoe, A. (2009). Usability and security of out-of-band channels in secure device pairing protocols. In *Proceedings of the 5th Symposium on Usable Privacy and Security*, page 11. ACM.

Kainda, R., Flechais, I., and Roscoe, A. (2010). Secure mobile ad-hoc interactions: reasoning about out-of-band (OOB) channels. *IWSSI/SPMU*, 2010:10–15.

Katz, J. and Vaikuntanathan, V. (2011). Round-optimal password-based authenticated key exchange. In *Theory of Cryptography Conference*, pages 293–310. Springer.

Krawczyk, H. (2010). Cryptographic extraction and key derivation: The HKDF scheme. In *Annual Cryptology Conference*, pages 631–648. Springer.

Naor, M., Rotem, L., and Segev, G. (2018). The security of lazy users in out-of-band authentication. In *Theory of Cryptography Conference*, pages 575–599. Springer.

Nguyen, L. H. and Roscoe, A. W. (2011). Authentication protocols based on low-bandwidth unspoofable channels: a comparative survey. *Journal of Computer Security*, 19(1):139–201.

OTRv4-development (2019). Specification of OTR version 4. https://github.com/otrv4/otrv4/blob/master/otrv4.md.

Rivest, R. L. and Shamir, A. (1984). How to expose an eavesdropper. *Communications of the ACM*, 27(4).

Roscoe, A. W. (2016). Detecting failed attacks on human-interactive security protocols. In *Cambridge International Workshop on Security Protocols*, pages 181–197. Springer.

Roscoe, A. W. and Ryan, P. Y. A. (2017). Auditable PAKEs: approaching fair exchange without a TTP. In *Cambridge International Workshop on Security Protocols*, pages 278–297. Springer.

Ruoti, S., Andersen, J., Monson, T., Zappala, D., and Seamons, K. (2018). A comparative usability study of key management in secure email. In *Fourteenth Symposium on Usable Privacy and Security*, pages 375–394.

Sasson, E. B., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., and Virza, M. (2014). Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474.

Shirvanian, M. and Saxena, N. (2014). Wiretapping via Mimicry: Short voice imitation man-in-the-middle attacks on crypto phones. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, page 868–879.

Stedman, R., Yoshida, K., and Goldberg, I. (2008). A user study of off-the-record messaging. In *4th symposium on Usable privacy and security*, pages 95–104.

Tan, J., Bauer, L., Bonneau, J., Cranor, L. F., Thomas, J., and Ur, B. (2017). Can unicorns help users compare crypto key fingerprints? In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 3787–3798. ACM.

Unger, N., Dechand, S., Bonneau, J., Fahl, S., Perl, H., Goldberg, I., and Smith, M. (2015). SoK: secure messaging.

In *2015 IEEE Symposium on Security and Privacy*, pages 232–249. IEEE.

Vaudenay, S. (2005). Secure communications over insecure channels based on short authenticated strings. In *Annual International Cryptology Conference*, pages 309–326. Springer.

Yao, A. C. (1982). Protocols for secure computations. In *23rd annual symposium on foundations of computer science (SFCS 1982)*, pages 160–164. IEEE.