Efficient Implementation of Password-based Authenticated Key Exchange from RLWE and Post-Quantum TLS

Xinwei Gao[1], Jintai Ding[2], Lin Li[1], Saraswathy RV[2], and Jiqiang Liu[1]
(Corresponding author: Jintai Ding and Lin Li)

Beijing Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing Jiaotong University[1]
No.3 ShangYuanCun, Haidian District, Beijing, 100044, P.R.China
Department of Mathematical Sciences, University of Cincinnati[2]
French Hall, 2815 Commons Way, Cincinnati, Ohio 45219, United States
(Email: jintai.ding@gmail.com; lilin@bjtu.edu.cn)

(Received Aug. 6, 2017; revised and accepted Nov. 21, 2017)

Abstract

Two post-quantum password-based authenticated key exchange (PAKE) protocols were proposed at CT-RSA 2017. Following this work, we give much more efficient and portable C++ implementation of these two protocols. We also choose more compact parameters providing 200-bit security. Compared with original implementation, we achieve 21.5x and 18.5x speedup for RLWE-PAK and RLWE-PPK respectively. Compare with quantum-vulnerable J-PAKE protocol, we achieve nearly 8x speedup. We also integrate RLWE-PPK into TLS to construct a post-quantum TLS ciphersuite. This allows simpler key management, mutual authentication and resistant to phishing attack. Benchmark shows that our ciphersuite is indeed practical.

Keywords: Authenticated Key Exchange; Implementation; Post-quantum; RLWE; TLS

1 Introduction

1.1 Key Exchange and Post-Quantum World

With the groundbreaking work "New Directions in Cryptography" from Diffie and Hellman in 1976 [9], the idea of key exchange (KE) and public key cryptography come into reality. Key exchange is a very important cryptographic primitive. With properly designed protocols, two or more parties can agree on same session key for message or data encryption using symmetric encryption algorithms over adversary-controlled network. Well known protocols including Diffie-Hellman key exchange (DH), elliptic curve DH (ECDH) *etc.* However, these protocols cannot authenticate user's identity, *i.e.*, man-in-the-middle (MITM) attack can compromise the security of communication. Authenticated key exchange (AKE) is a solution to this problem. AKE protocols can negotiate key and authenticate identity of communicating parties simultaneously. Well studied protocols including HMQV [20], authenticated DH *etc.* As long as each party has certified public key, two parties can use various explicit or implicit authentication mechanisms to verify the identity of communicating party. The widely-deployed approach is combining unauthenticated key exchange with digital signatures and trusted certificates [21]. This is also known as Public Key Infrastructure (PKI)-based authentication.

Another important line of AKE is password-based AKE (PAKE). PAKE utilizes human-memorable password (or passphrase) which is cryptographically insecure to authenticate and negotiate symmetric session key. PAKE is very strong in the sense that user does not reveal passwords to others [7]. In PAKE, password (or its variant) is pre-shared by both parties. Since only communicating parties know this password, it can be intuitively used as authentication mechanism. Advantages of PAKE include simpler key management since PAKE does not rely on certificates and signatures to authenticate, secure against webpage spoofing, phishing and man-in-the-middle (MITM) attacks since attacker does not know the password, user-friendly authentication (using human-memorable password), prevents offline dictionary attack *etc.* Plenty of PAKE protocols have been standardized and deployed in various applications. Examples and real-world applications of PAKE include PAK & PPK [5], Password Authenticated Key Exchange by Juggling (J-PAKE) [16], secure Pre-Shared Key (PSK) authentication for Internet Key Exchange (IKE) protocol in RFC 6617 [17], elliptic curve J-PAKE ciphersuites in TLS [8], Secure Remote Password protocol (SRP) in RFC 2945 [29] and patented protocols like Encrypted Key Exchange (EKE) [3], Simple Password Exponential Key Exchange (SPEKE) [18] *etc.* Also OpenSSL supports J-PAKE and Firefox Sync service adopted J-PAKE for authentication and key exchange. Some other works related to key exchange include [12, 13, 24] *etc.*

With the advent of quantum computers during past decades, people have realized the untapped potential from quantum computers and huge threats to current cryptographic constructions, especially public key algorithms. Two best-known attacks from quantum computers are Shor's algorithm [28] and Grover's algorithm [14]. Shor's quantum algorithm is widely conjectured to be effective against all mainstream public-key algorithms that designed based on integer factorization problem, discrete logarithm problem *etc.*, including RSA, Diffie-Hellman and elliptic curve-based ones. If large quantum computer exists, it is believed that most public key algorithms can be broken very efficiently. For Grover's quantum algorithm, it attacks symmetric encryption algorithms. Result shows that n-bit key provides $n/2$-bit security on quantum computers. Quantum brute force can be defeated by doubling key size without switching to new algorithms. Larger key size works for symmetric and hash algorithms but current public key algorithms will be broken regardless of key size. In 2017, D-Wave 2000Q quantum computer breaks the 2000-qubit barrier. Despite there are controversies around D-Wave on whether they are building truly quantum computer or not, they show the potential of building practical and very powerful quantum computers within coming years. In 2015, National Security Agency (NSA) announced plan to switch to quantum-resistant cryptography in near future. At PQCrypto 2016 conference, NIST announced their call for quantum-resistant cryptographic algorithms for future and plans for post-quantum cryptography standards. Therefore, it is imperative to build quantum-resistant and efficient algorithms, implementations and gain real-world deployment.

1.2 Contributions

In this paper, we first present a very efficient implementation of two RLWE-based post-quantum password-based authenticated key exchange protocols proposed at CT-RSA 2017 (denoted as PAKE17) [10]. We also choose more compact parameters providing at least 200-bit security. Our implementation achieve 21.5x and 18.5x performance improvement over original implementation of RLWE-PAK and RLWE-PPK protocol respectively. We also compare performance with J-PAKE, which is deployed in real-world applications but vulnerable to quantum computers. Our implementation is more efficient and achieves 8.5x and 7.4x speedup for RLWE-PAK and RLWE-PPK respectively. Benchmark proves that our work is indeed efficient, which is even faster than current widely deployed and quantum-vulnerable PAKE protocol. Our implementation is a portable C++ implementation and does not rely on new instruction set (*eg*, AVX2) to achieve high performance.

Second, we introduce a post-quantum TLS ciphersuite and present our proof-of-concept implementation. We integrate our efficient RLWE-PPK implementation into TLS ciphersuite in a similar way as pre-shared key ciphersuites. Pre-shared password in RLWE-PPK in this context is a pre-shared key. Advantages of our ciphersuite more convenient key management compared with PKI-based authentication, resistant to phishing attacks and mutual authentication. Benchmark of our implementation shows that our post-quantum TLS ciphersuite is truly practical.

1.3 Organization

We recall necessary background knowledge in Section 2. In Section 3, we revisit PAKE17 protocol and introduce new parameter choice with security level estimation, much more efficient implementation, comparison with original work and analysis. In Section 4, we introduce our post-quantum TLS ciphersuite based on RLWE-PPK, proof-of-concept implementation, performance and discussions. We conclude the paper in Section 5.

2 Preliminary

2.1 Post-Quantum Cryptography

Due to Shor's algorithm, major public key cryptosystems nowadays (RSA, Diffie-Hellman, ECDH *etc.*) are no longer secure when large quantum computer is available. Constructions built on these hard problems: integer factorization problem, discrete logarithm problem or elliptic-curve discrete logarithm problem can be broken on sufficient large quantum computer. Although it is hard to predict the exact time that efficient quantum computers can be built, most scientists believe that they will be built within decades.

Post-quantum cryptography refers to designing and building cryptosystems that can resist attacks from quantum computers. Generally, quantum-resistant cryptosystems can be achieved by these approaches: lattice-based, multivariate-based, hash-based, code-based and symmetric ciphers with larger key size. In this paper, we focus on lattice-based ones since they have strong provable security, high efficiency, simple structure and much smaller key sizes compared with other approaches. A lattice is a set of points in an n-dimensional space with periodic structure. The lattice $L(b_1, \cdots, b_n) = \sum_{i=1}^{n} x_i b_i : x_i \in Z$ is formed by linear combinations of n linearly independent vectors $b_1, \cdots, b_n \in R^n$. These vectors are called "lattice basis". With the groundbreaking work of Ajtai [1], cryptographic constructions based on lattice come to existence. The security of lattice-based constructions can be reduced to hard lattice problems, including Shortest Vector Problem (SVP), Closest Vector Problem (CVP) *etc.* Till now, no known efficient classical or quantum algorithm can solve these lattice problems. Lattice-based cryptosystem is one of the most promising constructions for post-quantum world.

2.2 Learning with Errors and Ring Variant

In 2005, Oded Regev showed a problem called Learning with Errors (LWE) [26]. LWE is proven to be as hard as solving several worst-case lattice problems when LWE instances are properly initiated. Regev and Peikert [25] showed both quantum and classical reductions from LWE to standard lattice problems. Properly chosen error term keeps LWE problem very hard to solve. Decision version of LWE is to distinguish truly uniform generated samples from $b_i = a_i \cdot s + e_i$, where s is secret key, e_i is the error term. Typically, s and e_i are sampled from discrete Gaussian distribution, a_i is uniformly random generated. Search version is to recover secret s given multiple LWE samples. If one can solve LWE problem, then underlying lattice problem can be solved as well due to reduction theorem. Till now, there are no public known efficient algorithms (both on classical and quantum computers) that can solve lattice problems.

Ring Learning with Errors (RLWE) problem [22] is the analogue of LWE in ring setting. The hardness of RLWE can be reduced to hard problems in ideal lattice. Compared with LWE, main attraction of RLWE problem mainly lies on its high efficiency since schemes based on RLWE reduces quadratic overhead in LWE. Solving RLWE problem can be reduced to finding shortest or closest vectors in ideal lattice. Until now, there are no efficient attacks that especially takes advantage of the ring structure, therefore attacks work for LWE may also work for RLWE.

LWE and RLWE are extremely versatile with available cryptography constructions including encryption, signature, key exchange, identity-based encryption, attribute-based encryption, function encryption, homomorphic encryption *etc.*

3 Efficient PAKE17 Implementation

In this section, we first briefly revisit the password-based authenticated key exchange proposed at CT-RSA 2017 in [10]. Two protocols in PAKE17 can be regarded as RLWE analogues of classical PAK and PPK protocol. Second, we introduce our efficient and portable C++ implementation for both three-pass explicit authenticated key exchange protocol (RLWE-PAK) and two-pass implicit authenticated key exchange protocol (RLWE-PPK). We also give more compact parameter choice, analysis of security level, efficient implementation, benchmark and comparisons with original work and J-PAKE protocol.

3.1 Revisit PAKE17

PAKE17 can be categorized to an important line of AKE protocols that takes advantage of human-memorable password to achieve authentication. Since only communicating parties share same password (or equivalents), it can be utilized to construct password-based AKE protocols. To the best of our knowledge, only [19] and PAKE17 provide post-quantum PAKE solutions. [19] can be viewed as a general lattice-based construction but it is based on Common Reference String (CRS). CRS-based protocols are more complex and weaker in security proofs. PAKE17 is based on Random Oracle Model (ROM) and this work gives two RLWE-based PAKE protocols: RLWE-PAK and RLWE-PPK. This work follows the idea and structure of PAK & PPK [5] but they are constructed based on RLWE key exchange of [11] (denoted as DING12). They also prove the security of both protocols following [3] with new techniques to adapt to RLWE setting. Proof-of-concept implementation shows that two protocols in PAKE17 are efficient. This is the first work that gives practical and provably secure RLWE-based PAKE constructions.

Figure 1 recalls three-pass explicit authenticated protocol RLWE-PAK. Figure 2 recalls two-pass implicit authenticated protocol RLWE-PPK.

3.2 New Parameter Choice and Security Estimation

We first choose new and more compact parameters for PAKE17 considering correctness, security and implementation efficiency. Parameters are given as follows: We choose prime $q = 1073479681$ (approximately 30 bits). q can ensure the correctness of PAKE17 by following theorem 3 in Section 3.2 of PAKE17 and it holds $q \equiv 1 \mod 2n$ that can realize NTT efficiently. We also remark that our prime p choice is more compact than original work (modulus $q = 2^{32}-1$ and it is not a prime). $n = 1024$ and standard deviation $\sigma = 8/\sqrt{2\pi} \approx 3.192$ are identical to original work.

We also analyze security level of our parameter choice. We use LWE estimator proposed in [2] to get security level of our parameter choice. LWE estimator gives a thorough security estimation for both LWE and RLWE-based cryptosystems. It evaluates security level of cryptosystems by computing attack complexity of exhaustive search, coded Blum-Kalai-Wassermann algorithm (BKW), lattice reduction, decoding, reducing BDD to unique-SVP, standard and dual embedding attacks *etc.* This estimation approach is considered as one of the most standardized and complete security estimation tool for LWE and RLWE-based cryptosystems. Authors of [2] also suggested it can be used to evaluate RLWE instances since no known attacks take advantage of ring structure.

Commands for executing security estimation script are given as follows:

- load("https://bitbucket.org/malb/lwe-estimator/

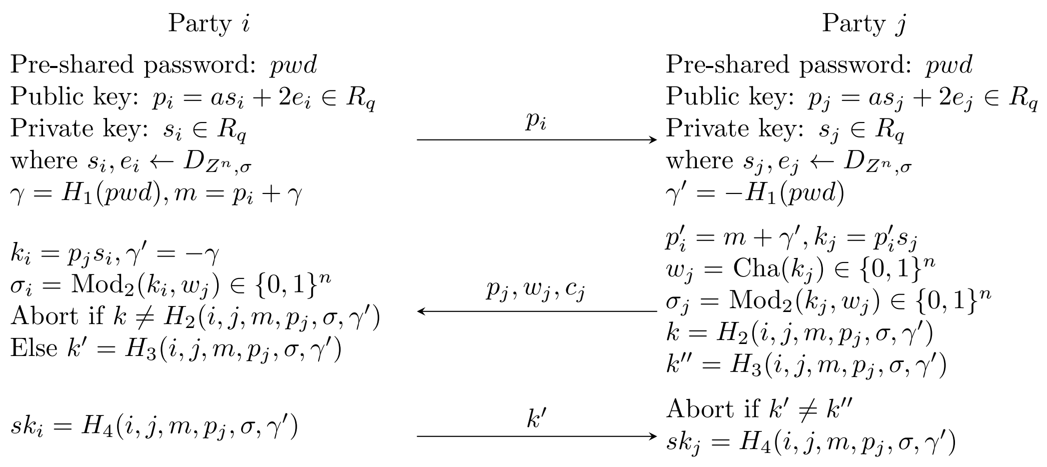
Party i
Party j
Pre-shared password: pwd
Public key: $p_i = as_i + 2e_i \in R_q$
Private key: $s_i \in R_q$
where $s_i, e_i \leftarrow D_{Z^n,\sigma}$
$\gamma = H_1(pwd), m = p_i + \gamma$
p_i
Pre-shared password: pwd
Public key: $p_j = as_j + 2e_j \in R_q$
Private key: $s_j \in R_q$
where $s_j, e_j \leftarrow D_{Z^n,\sigma}$
$\gamma' = -H_1(pwd)$
$p'_i = m + \gamma', k_j = p'_i s_j$
$w_j = \mathrm{Cha}(k_j) \in \{0,1\}^n$
$\sigma_j = \mathrm{Mod}_2(k_j, w_j) \in \{0,1\}^n$
$k = H_2(i, j, m, p_j, \sigma, \gamma')$
$k'' = H_3(i, j, m, p_j, \sigma, \gamma')$
p_j, w_j, c_j
$k_i = p_j s_i, \gamma' = -\gamma$
$\sigma_i = \mathrm{Mod}_2(k_i, w_j) \in \{0,1\}^n$
Abort if $k \neq H_2(i, j, m, p_j, \sigma, \gamma')$
Else $k' = H_3(i, j, m, p_j, \sigma, \gamma')$
$sk_i = H_4(i, j, m, p_j, \sigma, \gamma')$
k'
Abort if $k' \neq k''$
$sk_j = H_4(i, j, m, p_j, \sigma, \gamma')$

Figure 1: Explicitly authenticated PAKE17 protocol: RLWE-PAK

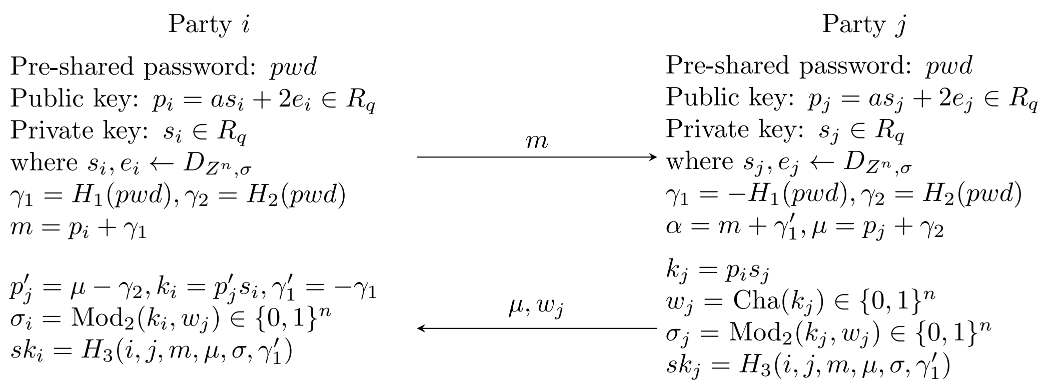
Party i
Party j
Pre-shared password: pwd
Public key: $p_i = as_i + 2e_i \in R_q$
Private key: $s_i \in R_q$
where $s_i, e_i \leftarrow D_{Z^n,\sigma}$
$\gamma_1 = H_1(pwd), \gamma_2 = H_2(pwd)$
$m = p_i + \gamma_1$
m
Pre-shared password: pwd
Public key: $p_j = as_j + 2e_j \in R_q$
Private key: $s_j \in R_q$
where $s_j, e_j \leftarrow D_{Z^n,\sigma}$
$\gamma_1 = -H_1(pwd), \gamma_2 = H_2(pwd)$
$\alpha = m + \gamma'_1, \mu = p_j + \gamma_2$
$k_j = p_i s_j$
$w_j = \mathrm{Cha}(k_j) \in \{0,1\}^n$
$\sigma_j = \mathrm{Mod}_2(k_j, w_j) \in \{0,1\}^n$
$sk_j = H_3(i, j, m, \mu, \sigma, \gamma'_1)$
μ, w_j
$p'_j = \mu - \gamma_2, k_i = p'_j s_i, \gamma'_1 = -\gamma_1$
$\sigma_i = \mathrm{Mod}_2(k_i, w_j) \in \{0,1\}^n$
$sk_i = H_3(i, j, m, \mu, \sigma, \gamma'_1)$

Figure 2: Implicitly authenticated PAKE17 protocol: RLWE-PPK

raw/HEAD/estimator.py");

- n, α, $q = 1024$, $f(8, 1073479681)$;
- set_verbose(1);
- _ = estimate_lwe(n, α, q, skip=["arora-gb"]).

We remark that our approach is the same as [4]. With above estimation approach, our parameter choice offers at least 200-bit security. Since LWE estimator is constantly updated, our result might slightly different from estimation using latest version of LWE estimator script.

3.3 Implementation and Performance

We use NFLlib library [23] to implement RLWE-PAK and RLWE-PPK protocols. NFLlib is a very efficient Number Theoretic Transform (NTT)-based C++ library dedicated to RLWE-based cryptography implementation. It contains various algorithms and programming optimizations for lattice cryptography. NTT, inverse-NTT and sampling from discrete Gaussian operations are very efficient. For fast polynomial multiplication, we adopt NTT and inverse NTT since our parameter q is chosen to instantiate NTT efficiently. Note that NFLlib provides non-constant time implementation and takes advantage of SSE/AVX2 instruction set to optimize NTT and inverse NTT computation.

Our choice for hash function is SHA3-256. We also utilize an extendable-output function (XOF) in our implementation-SHAKE-128. In RLWE-PAK, we have H_1 = SHAKE-128(pwd, 4096), H_2 = SHA3-256(C, S, m, μ, σ, γ'). In RLWE-PPK, we have H_1 = SHAKE-128(pwd, 4096), H_2 = SHAKE-128(SHA3-256(pwd), 4096), H_i = SHA3-256(H_{i-1}) ($i = 3, 4$). We safely set statistical distance from sampled distribution to discrete Gaussian distribution as 2^{-128} to preserve high statistical quality and security. More implementation techniques are introduced in Section 3.4. For Cha() and Mod$_2$(), since input is an polynomial in R_q, we take each coefficient in the polynomial and compute its corresponding value.

We test our performance of new implementation, original work and J-PAKE on same server equipped with a 3.4GHz Intel Xeon E5-2687W v2 processor and 64GB memory running 64-bit Ubuntu 14.04. Both PAKE17 implementations are compiled by g++ version 4.8.4 with '-O3 -fomit-frame-pointer -march=native -m64' options to achieve maximum performance. Note that this CPU does not support AVX2 instruction set, therefore the performance of our implementation only benefits from SSE instruction set. Recall that the only difference between our parameter choice and PAKE17 is modulus q since we choose same dimension n and standard deviation σ for sampling.

We also compare performance of our implementation with J-PAKE, which is known to be vulnerable to quantum computers. J-PAKE implementation we use is later integrated in OpenSSL and is compiled by gcc version 4.8.4 with '-O3 -fomit-frame-pointer -march=native -m64' options. All implementation codes only runs on single core and does not utilizes parallel computing mechanisms. We report average runtime over 10,000 executions of our implementation, original work and J-PAKE in Table 1.

Table 1: Performance of this work, original PAKE17 and J-PAKE implementation

		Client (ms)	**Server (ms)**
This work	RLWE-PAK	0.176	0.175
	RLWE-PPK	0.203	0.203
Original implementation	RLWE-PAK	3.472 (19.7x)	4.053 (23.2x)
	RLWE-PPK	3.488 (17.2x)	4.041 (19.9)
J-PAKE	-	1.499	1.495

Number in parentheses is number of times of corresponding runtime for original implementation compared with this work as baseline. Our implementation achieves 21.5x and 18.5x speedup over original implementation for both RLWE-PAK and RLWE-PPK. Compared with J-PAKE, our implementation is 8.5x and 7.4x faster for RLWE-PAKE and RLWE-PPK respectively. This result highlights that our post-quantum PAKE implementation is much more efficient and even much faster than legacy PAKE protocol implementation that is vulnerable to quantum computers.

3.4 Discussion

Compared with original PAKE17 implementation, we believe following improvements and optimizations make this work much more efficient:

1) Efficiency from NFLlib library. We use this library to implement major functions related to PAKE17 and this contributes to good performance. NTT, inverse-NTT and sampling computation costs 0.008ms, 0.010ms, and 0.011ms respectively and they are 48x, 45.6x and 13.9x faster than original implementation. There are 2 sampling, 2 NTT and 1 inverse NTT computation operations for RLWE-PAK and RLWE-PPK for both party i and j. NTT and inverse NTT functions take advantage the power of SSE instruction set, therefore it is much more efficient. In original work, they use NTL library which is much slower than NFLlib (performance comparison of NFLlib and NTL can be found in [23]). In original work, they adopted FFT for polynomial multiplication and implementation of discrete Gaussian sampler is less efficient. NTT, inverse-NTT and sampling take up around 25% of total running time in our implementation.

2) We adopt a much smarter hashing strategy combined with efficient hashing implementation. We choose SHA3-256 as hash function and SHAKE-128 as XOF. These efforts deliver much faster hashing computations. In original implementation, when adding hashed value $\gamma = \mathrm{H}(pwd)$ to public key p_i ($m = p_i + \gamma$), they hash a long and randomly generated number using SHA2-256 for $\frac{1024}{256/32} = 128$ rounds. Since SHA2-256 has 256-bit output and each coefficient of polynomial takes up 32-bit, therefore in each SHA2-256 hashing computation round, $\frac{256}{32} = 8$ polynomial coefficients are generated. By repeating 128 times, all 1024 coefficients are derived. We use an alternative and more efficient approach to achieve this. Since output of SHAKE family functions can be extended to any desired length, we adopt SHAKE-128 and extend the output to 4096 bytes to initiate 1024 polynomial coefficients. In our efficient implementation, generate $\gamma = \mathrm{H}(pwd)$=SHAKE-128(pwd,4096) only costs 0.027ms. There is 1 γ computation and 2 γ computations in RLWE-PAK and RLWE-PPK respectively. In original work, it costs total of 1.578ms and 3.125ms to generate γ for RLWE-PAK and RLWE-PPK respectively, which take up more than 50% of total running time. Our approach is much faster than original implementation by 58.4x (0.027ms) and 57.9x (0.054ms) for RLWE-PAK and RLWE-PPK respectively. Since γ computation is very costly, our approach has a significant improvement over original implementation. γ computation takes up around 20% of total running time in our implementation.

3) We avoid slower multiplication, division and modular operations by make full use of bitwise operations to boost performance. We also make full use of macros and various programming techniques to reduce unnecessary overhead, expensive memory copy operations *etc.* to improve efficiency. All these approaches also contribute to the efficiency of our implementation.

Moreover, we remark that our implementation is portable. Our code does not take advantage the power of new instruction sets (*eg*, AVX2) to speedup performance, therefore our implementation is portable and can run on more outdated devices.

We note that most time-consuming computation of our PAKE17 implementation is hashing six-tuple $(i, j, m, \mu, \sigma, \gamma')$ using SHA3-256 since they take up more than 8KB of storage. Hashing this tuple costs more time compared with other operations. It costs 0.077ms which takes up 43.75% and 36.95% of total running time for RLWE-PAK and RLWE-PPK respectively.

We believe that PAKE17 and our implementation are indeed practical for post-quantum world.

4 Post-quantum TLS Ciphersuite and Implementation

We construct our post-quantum TLS ciphersuite by integrating RLWE-PPK into TLS. It can be regarded as a post-quantum variant of DHE_PSK and ECDHE_PSK. We build our ciphersuite based on DHE_PSK in TLS v1.2 and adapt the notion of password in RLWE-PPK to a key for symmetric encryption which is also pre-shared by two parties (same as definition of PSK in DHE_PSK). Advantages of TLS and PAKE are well-inherited and integrated in our ciphersuite, including easier key management (without PKE-based certificates), mutual authentication, better performance (efficient implementation and no signatures), anti-phishing attack, simplified message flow (inherit from TLS and RLWE-PPK) *etc.* We introduce cryptographic primitive combination of our ciphersuite, implementation based on Botan open source C++ library, benchmarks and discussions.

4.1 Introduction

TLS is designed to ensure secure communications over adversary controlled network, providing secrecy and data integrity between two communicating parties. TLS is consisted with two major components: handshake protocol and record protocol. In handshake protocol, two parties negotiate and establish secure connection. In record protocol, two parties transmit encrypted and authenticated data securely. TLS is widely deployed in real world with applications like HTTPS (HTTP over TLS), IMAPS (IMAP over TLS), SMTPS (SMTP over TLS) *etc.* and it has already comprised more than half of total web traffic. It supports various methods for key exchange (Diffie-Hellman and elliptic curve variant, RSA *etc.*), authentication (pre-shared key, RSA, ECDSA *etc.*), encryption (AES, stream ciphers *etc.*) and message authentication code (MAC) algorithms. Two parties can agree on a premaster key using various key exchange algorithms. Other keys are generated through premaster key. For authentication, certificates and signatures are more preferred and widely deployed.

As TLS is so important and we are moving into a post-quantum world, we consider TLS should also adopt post-quantum cryptographic primitives. However, ciphersuites in the latest version of TLS fail to meet the demands since available key exchange and signature algorithms can be broken by quantum computers. Recently, a project called "Open Quantum Safety" (OQS) was launched [27]. It provides prototype open-source software implementation which integrate several unauthenticated post-quantum key exchange protocols into TLS. By combining post-quantum key exchange and classical signatures (RSA, ECDSA *etc.*), OQS project suggests several post-quantum TLS ciphersuites. Google did experiments on integrating a post-quantum key exchange into Chrome browser in canary channel and few Google domain TLS servers with ciphersuite called "CECPQ1" [6]. These works show that post-quantum cryptographic primitives are very promising for real-world applications.

4.2 Our Post-Quantum TLS Ciphersuite

Project OQS and Google's effort are based on adopting unauthenticated key exchange protocol with RSA/ECDSA signature to achieve authentication. We want to achieve authentication in a way that discards quantum-insecure signatures. Authenticate two parties using pre-shared key is a very practical and efficient approach as pointed out in [15]. In TLS v1.2, there are various ciphersuites that supports authentication using pre-shared key (PSK), where PSK is a pre-shared symmetric key for encryption and authentication. PSK ciphersuites including standalone PSK, DHE_PSK (Diffie-Hellman ephemeral for key exchange+PSK for authentication, ECDHE_PSK (elliptic curve DHE+PSK), RSA_PSK (RSA for key exchange+PSK) *etc.* Advantages for PSK-based ciphersuites including avoid expensive public key computations for authentication (signing and verifying), mutual authentication, avoid phishing attacks, simpler key management *etc.* It can be established with various methods and it is not the focus of this work.

As suggested in [15], integrating PAKE protocols in TLS does not require too much changes in TLS standards, therefore we can safely take this approach. To the best of our knowledge, there are no existing works that integrate post-quantum PAKE protocols into TLS. Since RLWE-PPK is a two-pass implicit authenticated key exchange protocol and it shares very similar structure with Diffie-Hellman, it can be regarded as post-quantum alternative for pre-shared key ciphersuite in TLS. Analogously, we can replace DHE_PSK in standard TLS with RLWE-PPK without modify structure of TLS and message flow significantly. As mentioned before, PSK is a pre-shared symmetric encryption key for authentication, therefore we pre-share key for both parties in our implementation.

Our post-quantum TLS ciphersuite "RLWE _PPK _WITH _AES _256 _GCM _SHA384" is designed and implemented based on TLS v1.2. We use our efficient implementation of RLWE-PPK to perform key exchange and authentication, thus our ciphersuite can achieve mutual authentication, forward security and resistant to quantum computing attacks. We give detailed cryptographic primitive combination of our post-quantum TLS ciphersuite:

- Key exchange and authentication: We integrate RLWE-PPK protocol revisited in Section 3.1 into TLS to achieve post-quantum key exchange and authentication. This ciphersuite can realize mutual authentication in a more convenient way than PKI-based approach, where client may not have user certificate. Parameter choice for RLWE-PPK follow Section 3.2.

- Authenticated encryption: We choose AES-256-GCM. It provides confidentiality, integrity and authenticity assurances on data.
- Hash function: We choose SHA-384. Our choice followed the principle proposed by NIST of deprecating SHA-1.

4.3 Implementation and Performance

Our proof-of-concept implementation is based on Botan library. Botan is a C++ library that provides implementations of a variety of cryptographic algorithms and protocols. We integrate our RLWE-PPK implementation into Botan library according to TLS v1.2 handshake and RLWE-PPK message flow. We also implement test programs that simulate TLS session between client and server using our ciphersuite. Both client and server program are run on localhost. Server listens on port 443 and client communicates with server. We measure runtime from the very beginning of session initiation and stop after handshake completes. Test programs run on a PC equipped with a 2.7GHz Intel Core i7-6820HQ processor and 4GB RAM running Ubuntu 14.04 64-bit version. Test programs are compiled by g++ 4.8.4 with optimization flags "-O3 -m64 -fstack-protector" and execute 1,000 times using single core. Average runtime over 1,000 executions of our ciphersuite for client and server handshake is 4.83ms and 4.94ms respectively.

Although our PAKE-PPK implementation is very efficient, our ciphersuite has larger communication cost. Size of PAKE-PPK key exchange messages from client to server and server to client is 3.75KB and 3.875KB respectively and this is larger than DHE/ECDHE/RSA key exchange messages (around 1-2KB). This might be a disadvantage of our ciphersuite. Also setting up pre-shared materials securely require more works.

5 Conclusion

In this paper, we present a portable and truly efficient post-quantum PAKE implementation for RLWE-PAK and RLWE-PPK protocols. We implement these two PAKE protocols in portable C++ style so that our code can run on a variety of devices. We achieve 21.5x and 18.5x performance improvement on both RLWE-PAK and RLWE-PPK over original implementation. Performance of RLWE-PAK and RLWE-PPK in this work is also 8.5x and 7.4x faster than J-PAKE, which is known to be widely deployed but vulnerable to quantum computers. We also integrate RLWE-PPK into TLS as post-quantum TLS ciphersuite. Proof-of-concept implementation based on Botan library shows that our ciphersuite is very practical. Our work shows that post-quantum cryptographic primitives like PAKE17 and real-world post-quantum applications like our post-quantum TLS ciphersuite can be truly efficient.

Acknowledgement

This work is supported by China Scholarship Council, National Natural Science Foundation of China (Grant No. 61672092) and Fundamental Research Funds for the Central Universities (Grant No. 2017YJS038). Jintai Ding is partially supported by NSF grant DMS-1565748 and US Air Force grant FA2386-17-1-4067.

References

[1] M. Ajtai, "Generating hard instances of lattice problems," in *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, pp. 99–108, 1996.

[2] M. R. Albrecht, R. Player and S. Scott, "On the concrete hardness of learning with errors," *Journal of Mathematical Cryptology*, vol. 9, no. 3, pp. 169–203, 2015.

[3] S. M. Bellovin and M. J. Merritt, *Cryptographic Protocol for Remote Authentication*, US Patent 5,440,635, Aug. 8, 1995.

[4] J. W. Bos, C. Costello, M. Naehrig and D. Stebila, "Post-quantum key exchange for the TLS protocol from the ring learning with errors problem," in *2015 IEEE Symposium on Security and Privacy*, pp. 553–570, 2015.

[5] V. Boyko, P. MacKenzie and S. Patel, "Provably secure password-authenticated key exchange using diffie-hellman," in *Advances in CryptologyEurocrypt 2000*, pp. 156–171, 2000.

[6] M. Braithwaite, S. Engineer, "Experimenting with post-quantum cryptography" July 7, 2016.

[7] T. Y. Chang, W. P. Yang, M. S. Hwang, "Simple authenticated key agreement and protected password change protocol", *Computers & Mathematics with Applications*, vol. 49, pp. 703–714, 2005.

[8] R. Cragie, F. Hao, "Elliptic curve j-pake cipher suites for transport layer security (tls)_draft-cragie-tls-ecjpake-00," pp. 24, 2016.

[9] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[10] J. Ding, S. Alsayigh, J. Lancrenon, R.V. Saraswathy and M. Snook, "Provably secure password authenticated key exchange based on rlwe for the post-quantum world," in *Cryptographers Track at the RSA Conference*, pp. 183–204, 2017.

[11] J. Ding, X. Xie and X. Lin, "A simple provably secure key exchange scheme based on the learning with errors problem," *IACR Cryptology ePrint Archive*, vol. 2012, pp. 688, 2012.

[12] M. S. Dousti and R. Jalili, "Forsakes: A forward-secure authenticated key exchange protocol based on symmetric key-evolving schemes," *Advances in Mathematics of Communications*, vol. 9, no. 4, pp. 471–514, 2015.

[13] S. Gonzlez, L. Huguet, C. Martnez and H. Villafae, "Discrete logarithm like problems and linear recurring sequences," *Advances in Mathematics of Communications*, vol. 7, no. 2, pp. 187–195, 2013.

[14] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, pp. 212–219, 1996.

[15] P. H. Griffin, "Transport layer secured password-authenticated key exchange," *Information Systems Security Association Journal*, vol. 13, no. 6, 2015.

[16] F. Hao, "J-pake: Password-authenticated key exchange by juggling," in *International Workshop on Security Protocols*, pp. 159-171, 2017.

[17] D. Harkins, "Secure pre-shared key (PSK) authentication for the internet key exchange protocol (IKE)," *Internet Engineering Task Force (IETF'12)*, RFC 6617, 2012.

[18] D. P. Jablon, "Cryptographic methods for remote authentication," May 1, 2001.

[19] J. Katz and V. Vaikuntanathan, "Round-optimal password-based authenticated key exchange," *Journal of Cryptology*, vol. 6597, no. 4, pp. 293–310, 2011.

[20] H. Krawczyk, "Hmqv: A high-performance secure diffie-hellman protocol (extended abstract)," in *Annual International Cryptology Conference*, pp. 546, 2005.

[21] C. C. Lee, M. S. Hwang, L. H. Li, "A new key authentication scheme based on discrete logarithms", *Applied Mathematics and Computation*, vol. 139, no. 2, pp. 343-349, July 2003.

[22] V. Lyubashevsky, C. Peikert and O. Regev, "On ideal lattices and learning with errors over rings," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 1–23, 2010.

[23] C. A. Melchor, J. Barrier, S. Guelton, A. Guinet, M. O. Killijian and T. Lepoint, "Nfllib: NTT-based fast lattice library," in *Cryptographer Track at the RSA Conference*, pp. 341–356, 2016.

[24] G. Micheli, "Cryptanalysis of a noncommutative key exchange protocol," *Advances in Mathematics of Communications*, vol. 9, no. 2, pp. 247–253, 2015.

[25] C. Peikert, "Public-key cryptosystems from the worst-case shortest vector problem," in *Proceedings of the Forty-first Annual ACM symposium on Theory of Computing*, pp. 333–342, 2009.

[26] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *Journal of the ACM*, vol. 56, no. 6, pp. 34, 2009.

[27] D. Stebila and M. Mosca, "Post-quantum key exchange for the internet and the open quantum safe project," *IACR Cryptology ePrint Archive*, vol. 2016, pp. 1017, 2016.

[28] P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *35th Annual Symposium on Foundations of Computer Science*, pp. 124–134, 1994.

[29] T. D. Wu *et al.*, "The secure remote password protocol," in *NDSS*, vol. 98, pp. 97–111, 1998.

Biography

Xinwei Gao received B.S. degree from Beijing Jiaotong University in 2014. He is a Ph.D. student at Beijing Key Laboratory of Security and Privacy in Intelligent Transportation at Beijing Jiaotong University. He is currently a visiting student at University of Cincinnati with sponsorship from China Scholarship Council. His research interest include post-quantum cryptography and RLWE-based key exchange.

Jintai Ding received Ph.D. degree in Yale in 1995. He is currently a professor of Mathematics at the University of Cincinnati. He was Humboldt fellow and visiting professor at TU Darmstadt in 2006-2007. He received the Zhong Jia Qing Prize from the Chinese math society in 1990. His research interest lies in post-quantum cryptography (PQC). He was a co-chair of the second international workshop on PQC. He and his colleagues invented LWE and RLWE-based key exchange protocols, Rainbow signature, "GUI" HFEV- signature and Simple Matrix encryption.

Lin Li received Ph.D. degree from Shandong University in 2007. She is currently an assistant professor at Beijing Key Laboratory of Security and Privacy in Intelligent Transportation at Beijing Jiaotong University. Her current research interests include cryptography and privacy preserving.

Saraswathy RV received bachelor degree in mathematics from University of Madras, India in 2006. She worked as a software quality assurance engineer in information technology industry for a few years till 2011 and is now a Ph.D. candidate in mathematics at University of Cincinnati. Her current research interests include lattice based cryptography, Learning with Errors and key exchange using RLWE.

Jiqiang Liu received B.S. and Ph.D. degree from Beijing Normal University in 1994 and 1999 respectively. He is currently a professor at the Beijing Key Laboratory of Security and Privacy in Intelligent Transportation at Beijing Jiaotong University. He is also the vice dean of graduation school of Beijing Jiaotong University. His research interests include security protocols, trusted computing and privacy preserving.