# Cryptanalysis of 'FS-PEKS: Lattice-based Forward Secure Public-key Encryption with Keyword Search for Cloud-assisted Industrial Internet of Things'

Zi-Yuan Liu, Yi-Fan Tseng*, and Raylin Tso

Department of Computer Science, National Chengchi University, Taipei, Taiwan
{zyliu, yftseng, raylin}@cs.nccu.edu.tw

**Abstract.** In this manuscript, we review lattice-based public-key encryption with the keyword search against inside keyword guess attacks (IKGAs) proposed by Zhang *et al.* in IEEE Transactions on Dependable and Secure Computing in 2019. We demonstrate that this scheme is insecure for IKGAs, although Zhang *et al.* demonstrated a secure proof.

**Keywords:** Public-key Encryption with Keyword Search · Lattice-based · Insider Keyword Guessing Attack · Cryptanalysis

## 1 Introduction

Public-key encryption with keyword search (PEKS), which was first proposed by Boneh *et al.* [3], allows a data sender to generate a ciphertext for a specific keyword, and then only the data receiver can generate a valid trapdoor for testing. Since third parties cannot obtain any information about the keyword from the ciphertext; thus, the PEKS scheme is suitable for cloud storage scenarios.

However, Byun [4] demonstrated that a malicious insider (*e.g.,* cloud server) may offline guess the keyword from the trapdoor, which is referred to as insider keyword guessing attacks (IKGA). The malicious server can adaptively encrypt a random keyword to generate a searchable ciphertext and test it using the trapdoor received from the data receiver. If the test passes, the malicious insider can obtain the keyword selected by the data receiver.

Recently, Zhang *et al.* proposed forward-secure public-key encryption with searchable encryption (FS-PEKS) that resists IKGA [12]. Unfortunately, we found some flaws in their work that made their scheme actually cannot withstand IKGA. More preciously, although the security proof is provided in the work, the IKGA security model defined by Zhang *et al.* cannot capture the concept of IKGA; thus, the adversary can still obtain the information about keyword from the trapdoor. In this manuscript, we present the steps to attack Zhang *et al.*'s scheme [12] and demonstrate that their scheme cannot resist IKGA.

The remainder of this manuscript is organized as follows. In Section 2, we review basic preliminary knowledge, including notations, lattice, and the definition of FS-PEKS. In Section 3, we summarize the PS-PEKS scheme proposed by Zhang *et al.* In Sections 4 and 5, we demonstrate that this scheme is susceptible to IKGA and discuss the reasons for this susceptibility, respectively. Finally, conclusion are presented in Section 6.

## 2 Preliminaries

Here, we review notations, the lattice concept, and the definition of FS-PEKS.

### 2.1 Notations

Let $\mathbb{Z}$ and $\mathbb{R}$ denotes a set of integer and rational numbers, respectively. For prime $q$, $\mathbb{Z}_q$ denotes a finite field (or Galois field) with order $q$. For an element $e$ and finite set $S$, $e \leftarrow S$ indicates that $e$ is selected uniformly at random from $S$. Moreover, for $a \in \mathbb{R}$, $\lfloor a \rfloor$ is rounded down to the closest integer of $a$. Finally, $||A||$ represents the $l_2$ norm of $A$.

* Corresponding author.

### 2.2 Lattice

Here, we briefly summarize the lattice concept.

Given $n, m, q \in \mathbb{Z}$ and $A \in \mathbb{Z}_q^{n\times m}$, we can define two lattices as follows:

- $\Lambda_q(A) = \{y \in \mathbb{Z}_q^m | \exists z \in \mathbb{Z}_q^n, y = A^\top z \bmod q\}$;
- $\Lambda_q^\perp(A) = \{e \in \mathbb{Z}_q^m | Ae = 0 \bmod q\}$.

In addition, we can further define a coset

$$\Lambda_q^u(A) = \{e \in \mathbb{Z}^m | Ae = u \bmod q\},$$

where $u \in \mathbb{Z}_q^n$.

**Discrete Gaussian** Let $\sigma$ be a positive real number and $x \in \mathbb{Z}^m$. Here, we define the Gaussian distribution of $\mathcal{D}_\sigma$ with parameter $\sigma$ by the probability distribution function $\rho_\sigma(x) = \exp(-\pi \cdot \|x\|^2/\sigma^2)$. Furthermore, for any set $\mathcal{L} \subset \mathbb{Z}^m$, we define $\rho_\sigma(\mathcal{L}) = \sum_{x\in\mathcal{L}} \rho_\sigma$. Then, the discrete Gaussian distribution over $\mathcal{L}$ with parameter $\sigma$ is defined as follows: for all $x \in \mathcal{L}$, $\mathcal{D}_{\mathcal{L},\sigma} = \rho_\sigma(x)/\rho_\sigma(\mathcal{L})$.

**Lattice with Trapdoors** Here, we introduce the algorithms related to the lattice trapdoor [1,2,5] used in Zhang *et al.*'s scheme [12].

1. $\mathsf{TrapGen}(1^n, 1^m, q) \to (A, T_A)$: For input $n, m, q \in \mathbb{Z}$, this algorithm outputs matrix $A \in \mathbb{Z}_q^{n\times m}$ with its corresponding trapdoor $T_A \in \mathbb{Z}_q^{m\times m}$, and the following property holds:
$$\{A : (A, T_A) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)\} \approx \{A : A \leftarrow \mathbb{Z}_q^{n\times m}\}.$$
2. $\mathsf{SamplePre}(A, T_A, \mu, \sigma) \to t$: For an input matrix $A \in \mathbb{Z}_q^{n\times m}$ and its trapdoor $T_A \in \mathbb{Z}_q^{m\times m}$, a vector $\mu \in \mathbb{Z}_q^n$, and parameter $\sigma \in \mathbb{R}$, this algorithm outputs sample $t \in \mathbb{Z}_q^m$ such that $At = \mu$ and $t$ is distributed in $\mathcal{D}_{\mathbb{Z}^m,\sigma}$.
3. $\mathsf{NewBasisDel}(A, R, T_A, \delta) \to T_B$: For an input matrix $A \in \mathbb{Z}_q^{n\times m}$, $\mathbb{Z}_q$-invertible matrix $R \leftarrow \mathcal{D}_{m\times m}$, trapdoor $T_A$, and parameter $\delta \in \mathbb{R}$, this algorithm outputs random matrix $T_B \in \Lambda_q^\perp(B)$, where $B = AR^{-1}$.

### 2.3 Forward Secure Public-key Encryption with Keyword Search

Here, we summarize the definition of the FS-PEKS scheme that resists IKGA [12]. The FS-PEKS scheme comprises five algorithms, i.e., the Setup, KeyUpdate, PEKS, Trapdoor, and Test algorithms, and three entities, i.e., a cloud server, the data owner, and the data receiver. These algorithms and entities are described as follows.

- Setup: For an input secure parameter $\kappa$, this probabilistic polynomial-time (PPT) algorithm outputs system parameter $\Sigma$ and the initial public/private key pair of the data sender $(PK_{s\|1}, SK_{s\|1})$ and data receiver $(PK_{r\|1}, SK_{r\|1})$, respectively. Note that the system parameter $\Sigma$ will implicit include in the following algorithms.
- KeyUpdate: For an input key pair $(PK_{s\|i}, SK_{s\|i})$ of the data sender, time period $i$, this PPT algorithm outputs an updated key pair $(PK_{s\|j}, SK_{s\|j})$ in time period $j$, where $i < j$. Note that the data receiver can also update their key pair using the KeyUpdate algorithm.
- PEKS: For an input public/private key pair of the data sender $(PK_{s\|j}, SK_{s\|j})$, the public key of the data receiver $PK_{r\|j}$ , current time period $j$, keyword $w$, this PPT algorithm outputs a forward-secure PEKS ciphertext $CT_j$ associated with keyword $w$.
- Trapdoor: For an input public/private key pair $(PK_{r\|j},\ SK_{r\|j})$ of the data receiver in current time period $j$ and a keyword $w$, this PPT algorithm outputs a trapdoor $t_{w\|j}$ for keyword $w$.
- Test: For an input trapdoor $t_{w\|j}$ in time period $j$ and forward-secure PEKS ciphertext $CT_j$, this deterministic polynomial-time algorithm outputs 1 if $CT_j$ and $t_{w\|j}$ contain the same keyword $w$; otherwise, this algorithm outputs 0.

**Definition 1 (Correctness of FS-PEKS).** *An FS-PEKS scheme is correct if, for all security parameters $\kappa$, ($\Sigma$, $PK_{r\|1}$, $SK_{r\|1}$, $PK_{s\|1}$, $SK_{s\|1}$) $\leftarrow$ Setup($\kappa$), any time period $j$, and any keyword $w$, the following requirement holds:*

$$\mathsf{Test}(t_{w\|j}, CT_j) = 1,$$

*where $t_{w\|j} \leftarrow$ Trapdoor($w$, $SK_{r\|j}$, $PK_{r\|j}$, $j$), $CT_j \leftarrow$ PEKS($PK_{s\|j}$, $SK_{s\|j}$, $PK_{r\|j}$, $j$, $w$), and $(PK_{r\|j}, SK_{r\|j})$ and $(PK_{s\|j}, SK_{s\|j})$ are updated from $(PK_{r\|i}, SK_{r\|i})$ and $(PK_{s\|i}, SK_{s\|i})$ for some time period $i < j$ using the* KeyUpdate *algorithm.*

**Security Requirements** A secure FS-PEKS scheme must satisfy ciphertext indistinguishability under the chosen keyword attacks. This security captures the keyword hiding from the ciphertext. In other words, any polynomial-time adversary cannot obtain any information about the keyword from the ciphertext. For further security requirements, PEKS should consider IKGA. More precisely, after receiving the trapdoor from the data receiver, a malicious insider (*e.g.,* a cloud server) may adaptively encrypt the keyword using the data receiver's public key, and then test whether the ciphertext and trapdoor match. As discussed in the literature [4], in real-world scenarios, the keyword space is not sufficient large; thus, there is a high probability that a malicious insider can obtain keyword information from the trapdoor.

**IKGA Security Model in Zhang *et al.*'s Scheme** Here, we briefly describe the IKGA security model defined by Zhang *et al.* [12]. This security model is a game operated interactively by a challenger $\mathcal{C}$ and adversary $\mathcal{A}$.

**Game - IKGA:**

- **Setup Phase**: $\mathcal{C}$ runs the Setup algorithm to generate system parameter $\Sigma$ and the public/private key pairs of the data sender and data receiver. Then, $\mathcal{C}$ sends the public keys of the data sender and receiver to $\mathcal{A}$.
- **Query Phase**: In this phase, $\mathcal{A}$ can adaptively query the following oracles.
  - Hash oracle: $\mathcal{A}$ is allowed to query hash oracles in time period $j = 1, \cdots, \eta$, where $\eta$ is the total number of time periods. Then, $\mathcal{C}$ responds with the corresponding hash value.
  - Trapdoor oracle: $\mathcal{A}$ can query this oracle for any keyword $w$ in time period $j$. $\mathcal{C}$ executes the Trapdoor algorithm to generate a valid trapdoor, and returns it to $\mathcal{A}$.
  - Searchable ciphertext oracle: $\mathcal{A}$ can query this oracle for any keyword $w$ in time period $j$. Here, to achieve forward security, the only restriction is $j > j^*$, where $j^*$ is the break-in time. $\mathcal{C}$ executes the PEKS algorithm to generate a valid ciphertext and returns it to $\mathcal{A}$.
- **Break-in Phase**: In this phase, $\mathcal{C}$ sends the private keys for time period $k = j^* + 1, \cdots, \eta$ to $\mathcal{A}$. Here, $j^*$ is the break-in time period.
- **Forgery Phase**: In this phase, $\mathcal{A}$ outputs a forged searchable ciphertext associated with $w^*$ in time period $j^*$, which could pass the testing process.

In this security model, the adversary is considered to have won the game if and only if the adversary is able to generate a valid searchable ciphertext, rather than just get the information about the keywords. Thus, we emphasize that the security model in the scheme proposed by Zhang *et al.* cannot capture the IKGA concept.

## 3 The FS-PEKS scheme by Zhang *et al.*

In this section, we review the FS-PEKS scheme proposed by Zhang *et al.* to resist IKGA [12].

- Setup: For an input security parameter $\kappa$, this algorithm runs the following steps:
  - Initialize a discrete Gaussian distribution $\chi$;
  - Select security Gaussian parameters $\delta = (\delta_1, \cdots, \delta_\eta)$, $\sigma = (\sigma_1, \cdots, \sigma_\eta)$ for each time period $(1, \cdots, \eta)$;
  - Randomly select a uniformly vector $\mu \leftarrow \mathbb{Z}_q^n$;
  - Select three secure hash functions $H_1 : \mathbb{Z}_q^{n\times m} \times \{0, \cdots, \eta\} \to \mathbb{Z}_q^{m\times m}$, $H_2 : \{0,1\}^{\ell_1} \times \{0, \cdots, \eta\} \to \mathbb{Z}_q^{n\times m}$, $H_3 : \mathbb{Z}_q^{m\times \ell} \times \{0,1\}^{\ell} \to \mathbb{Z}_q^n$, where the output of $H_1$ and $H_2$ are distributed in $\mathcal{D}_{m\times m}$;
  - Generate the data receiver's public-private key pair $(A_r, T_r) \leftarrow \mathsf{TrapGen}(^n, 1^m, q)$;
  - Generate the data sender's public-private key pair $(A_s, T_s) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$;
  - Output the system parameter
  $$\Sigma = (\mu, H_1, H_2, H_3, \chi, \delta, \sigma)$$
  and public-private key pair of data sender $(A_s, T_s)$ and data receiver $(A_r, T_r)$.
- KeyUpdate: For an input public/private key pair $(A_r, T_r)$ in the previous time period $i$ and current time period $j$, the data receiver runs the following steps:
  - Compute $R_{r\|i} = H_1(A_r\|i) + \cdots + H_1(A_r\|1) \in \mathbb{Z}_q^{m\times m}$, and $A_{r\|i} = A_r(R_{r\|i})^{-1} \in \mathbb{Z}_q^{n\times m}$;
  - Compute $R_{r\|i\to j} = H_1(A_r\|j) + \cdots + H_1(A_r\|i+1) \in \mathbb{Z}_q^{m\times m}$;
  - Compute $T_{r\|j} = \mathsf{NewBasisDel}(A_{r\|i}, R_{r\|i\to j}, T_{r\|i}, \delta_j)$, where $A_{r\|j} = A_{r\|i}(R_{r\|i\to j})^{-1} = A_r(R_{r\|j})^{-1} \in \mathbb{Z}_q^{n\times m}$;
  - Output the data receiver's public/private key pair $(A_{r\|j}, T_{r\|j})$ for time period $j$.

  Note that the data sender can use the same steps to generate their public/private key pair $(A_{s\|j}, T_{s\|j})$ for time period $j$.

– PEKS: For an input data sender's public/private key pair $(A_{s\|j}, T_{r\|j})$ for time period $j$, the data receiver's public key $A_{r\|j}$ for time period $j$, the current time period $j$, and keyword $w \in \{0,1\}^{\ell_1}$, the data sender runs the following steps to generate a searchable ciphertext:
  - Choose a random binary string $\gamma_j = (\gamma_{j_1}, \cdots, \gamma_{j_\ell}) \in \{0,1\}^{\ell}$;
  - Randomly select a uniform matrix $B_j \leftarrow \mathbb{Z}_q^{n\times\ell}$;
  - Select noise $e_j = (e_{j_1}, \cdots, e_{j_\ell})$, where $e_{j_1}, \cdots, e_{j_\ell} \leftarrow \mathbb{Z}_q$ according to $\chi$;
  - Select noise $v_j = (v_{j_1}, \cdots, v_{j_\ell})$, where $v_{j_1}, \cdots, v_{j_\ell} \leftarrow \mathbb{Z}_q^m$ according to $\chi$;
  - Compute $\beta_j = H_2(w\|j)$;
  - Compute $CT_{j_1} = \mu^\top B_j + e_j + (\gamma_{j_1}, \cdots, \gamma_{j_\ell})\lfloor q/2 \rfloor$, $CT_{j_2} = (A_{r\|j}\beta_j^{-1})B_j + v_j$;
  - Compute $h_j = H_3(CT_{j_2}\|\gamma_j) \in \mathbb{Z}_q^n$, and generate $\zeta_j \leftarrow \mathsf{SamplePre}(A_{s\|j}, T_{s\|j}, h_j, \sigma_j)$;
  - Output searchable ciphertext
$$CT_j = (CT_{j_1}, CT_{j_2}, \zeta_j).$$
– Trapdoor: For an input public/private key pair $(A_{r\|j}, T_{r\|j})$ of the data receiver in the current time period $j$ and a keyword $w$, the data receiver runs the following steps to generate a trapdoor:
  - Compute $\beta_j = H_2(w\|j)$, and compute $T_{w\|j} \leftarrow \mathsf{NewBasisDel}(A_{r\|j}, \beta_j, T_{r\|j}, \delta_j)$;
  - Compute $t_{w\|j} \leftarrow \mathsf{SamplePre}(A_{r\|j}\beta_j^{-1}, T_{w\|j}, \mu, \sigma_j) \in \mathbb{Z}_q^m$;
  - Output a trapdoor $t_{w\|j}$ for keyword $w$ and time period $j$.
– Test: For an input trapdoor $t_{w\|j}$ from the data receiver and a ciphertext $CT_j$ in the current time period $j$, the cloud server runs as follows:
  - Compute $\gamma_j = (\gamma_{j_1}, \cdots, \gamma_{j_\ell}) \leftarrow CT_{j_1} - t_{w\|j}^\top CT_{j_2}$;
  - For $k = 1, \cdots, \ell$, if $|\gamma_{j_k} - \lfloor q/2 \rfloor| < \lfloor q/4 \rfloor$, set $\gamma_{j_k} = 1$; otherwise, set $r_{j_k} = 0$. Then, update $\gamma_j$;
  - Compute $h_j = H_3(CT_{j_2}\|\gamma_j) \in \mathbb{Z}_q^n$;
  - If $A_{s\|j}\zeta_j = h_j$ and $\zeta_j$ is distributed in $\mathcal{D}_{\Lambda_q^{h_j}(A_{s\|j}),\sigma_j}$, the cloud server outputs 1; otherwise, it outputs 0.

## 4 Cryptanalysis of FS-PEKS Scheme

In this section, we describe how the FS-PEKS scheme proposed by Zhang *et al.* [12] is susceptible to IKGA.

**Lemma 1.** *The FS-PEKS is vulnerable to IKGA.*

*Proof.* In this proof, we describe the steps performed by a malicious insider $\mathcal{A}$.

– **Step 1:** After receiving trapdoor $t_{w\|j}$ from the data receiver, $\mathcal{A}$ first randomly selects a binary string $\gamma \in \{0,1\}^{\ell}$ and other parameters required by the PEKS algorithm.
– **Step 2:** $\mathcal{A}$ selects a possible keyword $w'$ and generates $CT_{j_1}$ and $CT_{j_2}$ as described in Section 3, except to generate $\zeta_j$.
– **Step 3:** Using the trapdoor, $\mathcal{A}$ generates a random binary string $\gamma_j'$ by computing $CT_{j_1} = t_{w\|j}^\top CT_{j_2}$.
– **Step 4:** If $\gamma_j = \gamma_j'$, the guessed keyword $w'$ is a valid keyword $w$; otherwise, $\mathcal{A}$ returns to Step 2 and continues to select and test other keywords.

Therefore, if the keyword is common, there is a high probability that $\mathcal{A}$ will obtain the information of the keyword hidden by the trapdoor. □

## 5 Discussion and Future Work

Recently, Huang and Li [7] proposed an effective method against IKGA, called "public-key authenticated encryption with keyword search". In this method, the data sender encrypts the keyword, and authenticates the ciphertext. For the Test algorithm, the trapdoor tests whether its keyword corresponds to the ciphertext generated by the sender, and tests that the ciphertext was generated by the data sender. Therefore, a malicious insider cannot generate a ciphertext that can be verified to guess the keywords selected by the data receiver.

In the following, we discuss why Zhang *et al.*'s work [12] cannot withstand IKGA. At a high level, to withstand against IKGA, Zhang *et al.*'s used the same concepts as authenticated encryption [7] to counter IKGA. In other words, the data sender must generate a "signature" $\zeta_j$ using the SamplePre function to authenticate the ciphertext $CT_{j_2}$ and the random binary string $\gamma_j$ they selected. Unfortunately, in Zhang *et al.*'s scheme [12], encryption and authentication operate independently; thus, their scheme only prevents a malicious insider from generating a valid $\zeta_j$, *i.e.,* the malicious insider can still test the trapdoor by recovering $\gamma_j$. Although Zhang *et al.* provided a security proof and showed that their scheme can withstand IKGA, the security model they proposed does not apply to IKGA scenarios. In their IKGA security model (Section 2.3), the adversary must forge a ciphertext that can pass the Test algorithm for a given challenge keyword. However, even if the adversary cannot forge such ciphertext (Section 4), they can still guess the keywords used by the trapdoor offline.

To the best of our knowledge, only pairing-based public-key authenticated encryption with keyword searchable has been proposed [6,7,8,9]; however, these schemes are considered vulnerable to future quantum computer attacks [10,11]. Therefore, determining how to realize an efficient lattice-based approach is an urgent problem to solve.

## 6 Conclusion

In this manuscript, we have presented cryptanalysis of the FS-PEKS scheme proposed by Zhang *et al.* [12]. The security model of Zhang *et al.*'s scheme is unsuitable for IKGA scenarios, and thus the scheme cannot withstand IKGA described in this manuscript.

## Acknowledgments

This research was supported by the Ministry of Science and Technology, Taiwan (ROC), under Project Numbers MOST 108-2218-E-004-001-, MOST 108-2218-E-004-002-MY2, and by Taiwan Information Security Center at National Sun Yat-sen University (TWISC@NSYSU).

## References

1. Agrawal, S., Boneh, D., Boyen, X.: Lattice Basis Delegation in Fixed Dimension and Shorter-ciphertext Hierarchical IBE. In: Rabin, T. (ed.) Advances in Cryptology - CRYPTO'10. pp. 98–115. Springer, Berlin, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14623-7
2. Alwen, J., Peikert, C.: Generating Shorter Bases for Hard Random Lattices. Theory Comput. Syst. **48**(3), 535–553 (2011). https://doi.org/10.1007/s00224-010-9278-3
3. Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public Key Encryption with Keyword Search. In: Cachin, C., Camenisch, J.L. (eds.) Advances in Cryptology - EUROCRYPT'04. pp. 506–522. Springer Berlin Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_30
4. Byun, J.W., Rhee, H.S., Park, H.A., Lee, D.H.: Off-line Keyword Guessing Attacks on Recent Keyword Search Schemes over Encrypted Data. In: Jonker, W., Petković, M. (eds.) Proceedings of the Third VLDB International Conference on Secure Data Management - SDM'06. pp. 75–83. Springer-Verlag, Berlin, Heidelberg (2006). https://doi.org/10.1007/11844662_6
5. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for Hard Lattices and New Cryptographic Constructions. In: Dwork, C. (ed.) Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing - STOC'08. pp. 197–206. Association for Computing Machinery, New York, NY, United States (2008). https://doi.org/10.1145/1374376.1374407
6. He, D., Ma, M., Zeadally, S., Kumar, N., Liang, K.: Certificateless Public Key Authenticated Encryption With Keyword Search for Industrial Internet of Things. IEEE Trans. Ind. Informatics **14**(8), 3618–3627 (2018). https://doi.org/10.1109/TII.2017.2771382
7. Huang, Q., Li, H.: An Efficient Public-key Searchable Encryption Scheme Secure against Inside Keyword Guessing Attacks. Inf. Sci. **403-404**, 1 – 14 (2017). https://doi.org/10.1016/j.ins.2017.03.038
8. Li, H., Huang, Q., Shen, J., Yang, G., Susilo, W.: Designated-server Identity-based Authenticated Encryption with Keyword Search for Encrypted Emails. Inf. Sci. **481**, 330–343 (2019). https://doi.org/10.1016/j.ins.2019.01.004
9. Noroozi, M., Eslami, Z.: Public Key Authenticated Encryption with Keyword Search: Revisited. IET Information Security **13**(4), 336–342 (2018). https://doi.org/10.1049/iet-ifs.2018.5315
10. Shor, P.W.: Algorithms for Quantum Computation: Discrete Logarithms and Factoring. In: Goldwasser, S. (ed.) Proceedings 35th Annual Symposium on Foundations of Computer Science - FOCS'94. pp. 124–134. IEEE Computer Society Press (1994). https://doi.org/10.1109/SFCS.1994.365700
11. Shor, P.W.: Polynomial-time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. SIAM Review **41**(2), 303–332 (1999). https://doi.org/10.1137/S0036144598347011
12. Zhang, X., Xu, C., Wang, H., Zhang, Y., Wang, S.: FS-PEKS: Lattice-based Forward Secure Public-key Encryption with Keyword Search for Cloud-assisted Industrial Internet of Things. IEEE Trans. Dependable and Secur. Comput. (2019). https://doi.org/10.1109/TDSC.2019.2914117, (early access)