a public key encryption with keyword search scheme based on pseudo-inverse matrix

Thuc D. Nguyen1 – Peeter Laud2 – Van H. Dang1

1University of Science, HCMC

2University of Tartu

**Abstract**

We study mechanism of Public Key Encryption with keyword Search, or searchable encryption, which enables a recipient to give a the third party the ability to test whether a give word is a keyword in a large message but server should learn nothing else about the keyword and the message. In this paper, we propose an effective and security scheme which based on the pseudo-inverse matrices.

1. **Introduction**

***1.2 Searchable encryption and related works***

Search encryption is a way which enables a recipient to give a un-trust server the ability to test whether W is a keyword in a large message M but server should learn nothing else about the keyword W and the message M.

[Bre04] proposed two schemes for creating searchable encrypted entries for audit log: Searchable Symmetric Key Scheme (SSKE) and Searchable Asymmetric Key Scheme. In the symmetric key scheme, the authors used keyed pseudorandom functions (such as HMAC-SHA1), shared secrete, symmetric encryption and random value to create searchable encrypted entry to the audit log. However, this scheme cannot prevent attacks in case the server is compromised. It is because all secret information is stored in the server. The second scheme, asymmetric scheme, uses Identity-Based-Encryption [Bon01] (IBE, which is proved to have a strong security property – namely key-privacy [Bell01]) to encrypt the mixture (by XOR operation) of keywords and symmetric encryption key (the key necessary for decrypting the message). The user (if he approves) must have the master secret shared with the server and keyword (public key) to compute private key (the key need for IBE – decrypt) and search for a keyword. The authors also discussed three ways of optimization: (1) reuse result of pairing operation of IBE encryption; (2) index when servers may collect queries into “blocks” to be sent to the audit log all at once and keywords are repeated among several audit log entries within a block. However this optimization may leak partial information about the frequency of keywords presented in one single block; (3) reuse random value.

[Son00] proposed Sequential Scan Scheme for searches on encrypted data with advantages: provable secrecy, controlled searching, hidden queries, and query isolation. Un-trusted server cannot know anything about the plaintext corresponding to cipher-text which are stored in server. Besides, it is just able to search for a word with the user’s authentication. It can search for a secret word without knowing the word. Even a single secret key is revealed, no extra information is leaked beyond the ability to identify the position of other words. The authors used pseudorandom generators, pseudorandom functions, pseudorandom permutations, flexible keys, deterministic encryption algorithm which depends only on plaintext (word), not on the position of the word in document to provide the above features. However, sequential scan may not be efficient enough when the data size is large. The authors suggested to use a pre-computed index to speed up searching in large size data.

[Golle04] proposed a scheme which supported secure consecutive search. The security of their scheme relies on the Decisional Diffie-Hellman (DDH) assumption.

***1.2 Our contributions***

*General searchable encryption*

We consider the general searchable encryption of D. Boneh et al. [BOP04]. In this scheme, a sender B who want to send a secrete message to a recipient A via a un-trust server. The scheme is briefly described as the following:

* The sender B encrypts his message using a standard public key system. He then appends to the resulting cipher-text a public key encryption with keyword search (PEKS) for each keyword. B send a message M with keywords W1,…,Wp to C: EApub(M)||PEKS(Apub,W1)||…||PEKS(Apub,Wp), where Apub is A’s public key and E a encryption function
* The recipient A gives the 3rd party C a certain trapdoor TW that enables C to test whether one of the keywords associated with the message is equal to the work W of A’s choice: given PEKS(Apub,W’) and TW, C can test if W=W’

**Definition 1:** *A public key encryption with keyword search scheme is a tuple of probabilistic polynomial time algorithms* (KeyGen*;* PEKS*;* Trapdoor*;* Test)*, such that*

* ***KeyGen(s)****: takes a security parameter s and return a pair of keys (Apub,Apriv)*
* ***PEKS(Apub,W)****: for a Apub and W, produces a searchable encryption of W*
* ***Trapdoor(Apriv,W)****: returns a trapdoor Tw*
* ***Test(Apub,S,Tw)****: given Apub, a searchable encryption S=PEKS(Apub,W’), and a trapdoor Tw=Trapdoor(Apriv,W’), outputs if W=W’*

*Contributions and organization*

1. We propose a searchable encryption scheme, say PPEKS, which uses only linearly operations. The complexity of the linear matrix operations is very low. The detail of PPEKS will be presented in Section 3.
2. The proposed scheme bases on the notations of pseudo-inverse matrices which will be presented in the next section. From pseudo-inverse matrices, we obtain the secure encrypted keywords and the secure trapdoors. Section 4 will present a comparison PPEKS with the PEKS based on Decisional Diffie-Hellman (DDH) assumption.
3. **Preliminaries**
   1. ***Pseudo-inverse matrix***

Consider the linear equation system:

(2.1) Ax = b,

where A ∈ ℝm×n, x ∈ ℝn, and b ∈ ℝm.

Moore [Moo20] and Penrose [Pen55], which was independently worked, showed that there is a general solution to (2.1) of the form x = A+b. The matrix A+ is called pseudo-inverse matrix of the matrix A. And they proved that this matrix is the unique matrix that satisfies all of the following four criteria:

* AA+A = A
* A+AA+ = A+
* (AA+)T = AA+
* (A+A)T = A+A

where MT is the transpose matrix of M: M = (mij), MT = (mji).

* 1. ***Properties of pseudo-inverse matrices***

For the implementation purpose, we recall two results that the proofs can be found in [GC96].

**Proposition 1:**

* If A is invertible, then A+ = A-1
* If **O** is a zero matrix**,** then **O**+ = **O**T
* (A+)+ = A
* (AT)+ = (A+)T

**Proposition 2 (computing pseudo-inverse via a limiting process)**:

(2.4) 

Note that these limits exist even if (AAT)-1 and (ATA)-1 do not exist [GC96].

In some special cases, we have also two results which the proofs can be found in [BT03]:

**Proposition 3 (the explicit formula of full rank matrices):**

* If the columns of A are linearly independent, then ATA is invertible. In this case,

(2.5) A+ = (ATA)-1AT

* If the rows of A are linearly independent, then AAT is invertible. In this case,

(2.6) A+ = AT(AAT)-1

**Proposition 4:**

1. If the pseudo-inverse of ATA is already known, then

(2.2) A+ = (ATA)+AT

* If the pseudo-inverse of (AAT) is already known, then

(2.3) A+ = AT(AAT)+

1. **A scheme for searchable encryption**

In this section, we reuse scheme of D. Boneh et al [BOP04], as represented above. However, our proposed scheme will be based on the pseudo-inverse matrices instead of the Decisional Diffie-Hellman (DDH) assumption and bilinear maps.

Let

H1: {0,1}\* → {0,1}m

and

H2: {0,1}m → {0,1}k

be two one-way hash functions, where k, m ∈ ℕ\{0}.

Let X ∈ {0,1}k×n and X+ ∈ {0,1}n×k be the pseudo-inverse of X; and let Y ∈ {0,1}n×m and Y+ ∈ {0,1}m×n be the pseudo-inverse of Y, where n ∈ ℕ\{0}.

Proposed public key encryption with keyword search, PPEKS, is defined as the followings:

**PPEKS = <KeyGen, PPEKS, Trapdoor, Test>,**

where:

* **KeyGen(k,m,n):** returns Apub= (XY,X+XY) and Apriv= (X,Y).
* **PEKS(Apub,W):** generates by randomly a non-singular matrix Q ∈ {0,1}m×m; computes H1(W) and generates a non-singular matrix U∈ {0,1}k×k using H1(W); returns A=UXYQ; B=X+XYQ.
* **Trapdoor(Apriv,W):** generates randomly a non-singular matrix R ∈ {0,1}m×m; computes V = H2(H1(W)R) and uses H1(W) to generate the inverse matrix U-1 of U; returns TW =(C,D), where C=VU-1 and D=VX
* **Test(Apub,S,TW):** Let S = (A,B) and T = (C,D). If CA = DB then returns true; false otherwise.

1. **Performance and comparison**
   1. ***Complexity of proposed searchable encryption scheme***

To analyze the performance, we will analyze the proposed public key encryption with keyword search, PPEKS, in terms of memory cost, computational cost, security and availability.

*Memory and transmission cost*

In PPEKS, the searchable encryption S = (A,B) which is stored at the un-trust server needs (k×m + n×m) = m(k + n) bits, in which k×m bits for A = UXYQ and n×m bits for B=X+XYQ.

To search an encrypted keyword, there are k+n bits of the trapdoor TW = (VU-1,VX) transferred to un-trust server.

*Computational cost*

PPEKS scheme used mainly linear matrix operations. The complexity of the linear matrix operations is very low. Therefore, the cost to generate a searchable encryption S of the algorithm PPEKS is low. Likewise, the cost to generate a trapdoor TW of a given keyword W is low, too.

The complexity of test algorithm is very low too, because, there are only two linearly matrix operation and a vector comparison operation are used.

Perhaps the complexity of PPEKS is by finding pseudo-inverse matrix using iteration algorithm by proposition 2. However, the KeyGen algorithm is done just one time, thus, this complexity can accepted. On the other hand, using the proposition 3, we can quickly generate a pseudo-inverse matrix of a given full rank matrix. The following algorithm to fast generating a pair of key <Apub,Apriv>.

To generate a singular matrix, we used results of linear algebra as the followings.

**Proposition 5:**

* Let L ∈ {0,1}p×p be a lower-triangle matrix. If L(i,i) = 1, for all i.1,…,p, then L is non-singular.
* Let U ∈ {0,1}p×p be a upper-triangle matrix. If U(i,i) = 1, for all i.1,…,p, then U is non-singular.
* If L,U ∈ {0,1}p×p are two non-singular matrices then A = PLU ∈ {0,1}p×p is non-singular, where P ∈ {0,1}p×p is a permutation matrix.

**Proof:**

* L ∈ {0,1}p×p is a lower-triangle matrix, and L(i,i) = 1, for all i=1,…,p. We have the determinant of L is defined by: . Thus, L is non-singular.
* Likewise, U is non-singular.
* L and U are two non-singular matrices. We have, det(L) ≠ 0, det U ≠ 0. P is a permutation matrix, so det(P) ≠ 0. Hence, det(A) = Det(PLU) = det(P)×det(L)×det(P) ≠ 0, or A is non-singular.

**Proposition 6:** Given a non-singular matrix A ∈ {0,1}k×k and Z ∈ {0,1}k×(n-k) is an arbitrary matrix. Let X = [A;Z] ∈ {0,1}k×n, then Rank(X) = k.

**Proof:** because A is non-singular, k rows of A are independent, or A has k rows different. Let Ai = (ai1,…,aik) and Aj =(aj1,…,ajk) be two different rows of A, then Xi = (ai1,…,aik,zi1,…,zi(n-k)) ≠ (aj1,…,ajk,zj1,…,zj(n-k)) = Xj. Thus, k rows of X are different, or Rank(X) = k.

It follows that X+ = XT(XXT)-1 by proposition 3.

**QKeyGen: generating quickly keys for PPEKS using permutation matrices P, Q**

1. Generating a non-singular matrix A ∈ {0,1}k×k using algorithm MatrixGen
2. Generating an arbitrary matrix Z ∈ {0,1}k×(n-k)
3. Let X = [A;Z] ∈ {0,1}k×n
4. Calculating X+ = XT(XXT)-1
5. Generating a non-singular matrix B ∈ {0,1}m×m using algorithm MatrixGen
6. Generating an arbitrary matrix W ∈ {0,1}(n-m)×m
7. Let ∈ {0,1}n×m

By proposition 3, outputs of QKeyGen are Apub = XY and Apriv = X.

**MatrixGen: generating a random non-singular matrix using a random permutation matrix**

* 1. Generating a non-singular lower-triangle matrix L ∈ {0,1}T×T
  2. Generating a non-singular upper-triangle matrix U ∈ {0,1}T×T
  3. Calculating A = PLU

*Security and avalability*

**Theorem 1:** for every non-singular matrix A, there exists a permutation matrix P, a lower-triangle matrix L and a upper-triangle matrix U such that PA = LU.

**Proof sketch:** let us consider solving a system of n linear equations and n unknowns. One of the most popular methods is to apply forward Gaussian elimination so that eventually we can come up with a triangular system, i.e. all entries on the diagonal, which are the pivot positions, are non-zero, and either upper or lower half of the matrix contains only zeros. Once we have the triangular system, we can first solve the equation that contains only one unknown, and then solve other equations in the backward order, which we call back-substitution process.

If the matrix is singular, then there is no way we can come up with the triangular matrix. If the matrix is nonsingular, there must be one solution to the system, and hence there will be a way to come up with a triangular with all non-zero pivot. However, during the elimination process, there might be some point at which a zero appears at the pivot position. The elimination can be continued by exchanging this rwo with another row that contains a non-zero pivot. Since the matrix is non-singular, we will always be able to find a row like this. Eventually we will come up with the triangular matrix and solve the equation system.

In summary, from a non-singular matrix A, we can always perform the elimination process so that we can come up with a product of LU, which are triangular matrices. During the elimination process, we might apply some exchanges over rows of A, or equivalently we multiply A which a permutation matrix P. In other words, PA = LU, or A=P-1LU, where P-1 is also a permutation matrix.

**Theorem 2**: the number of matrices generated by MatrixGen algorithm is .

**Proof:** by Theorem 1, the MatrixGen algorithm generates a set of binary matrices ℳ = {A ∈ {0,1}T×T| det(A) = 1}. We will calculate the cardinality of this set.

Let V be q space of T-dimensional vectors on binary field ℤ2, {e1,…,eT} be a base of V, and σ ∈ ℳ(V). Let e’i = eiσ (1 ≤ i ≤ T). Then {e’1,…,e’T} is also a base of V. Note that e’1 can be chosen randomly from 2T – 1 non-zero elements of V.

Suppose that we have chosen e’1,…,e’k (k < T). These vectors generate a sub-space S which has 2k elements. We can then choose randomly e’k+1 ∉ S. We have 2T – 2k ways to choose e’k+1. Thus, the number of possible ways to choose a set of vectors e’1,…,e’T is 

It follows that the key space generated by QKeyGen is very large in order that an exhausting algorithm is unable.

On the other hand, knowing the messages such as XY, X+XY, UXYQ; X+XYQ, VU and VX may not be helpful for recovering the secrets X and Y.

Let us consider XY. Even if Y is completely known, the probability of determining the correct value of each element of XY would be ½, which is very small by X is chosen at random. Likewise, the probability of determining the correct X, V and U from VU and VX is very small.

Now, suppose that rank(X) = r; and let us further assumes that

,

where Ir×r ∈ {0.1}r×r is an identity matrix of order r×r; and Ir×r ∈ {0.1}r×r the left-upper sub-matrix of the matrix Y. Then the probability of determining the correct Q is 2-(n-r)k.

Based on this analysis, it can be assumed that the probability of successful cracking of PPKES is 2-(n-r)k. Thus, the security of the PPKES is reasonably high for carefully chosen parameters. To ensure 2(n-r)k is a large number, n must be considerably larger than r. And this can be guaranteed by ensuring that m < n.

* 1. ***A comparison***

In this section, we present a comparison between proposed searchable encryption, PPEKS, scheme and the searchable encryption scheme based on the Decisional Diffie-Hellman (DDH) assumption, PEKS-DDH.

The Decisional Diffie-Hellman (DDH) assumption based on difficulty of the discrete logarithm problem. The average computational complexity of the discrete logarithm problem using the best method known to date [MOV97] is O(exp(1.923+o(1))(log2p)1/3(log2 log2 p)2/3)) bit operations.

To achieve a security level complexity of 249.3 in PEKS-DDH, it is needed 200bits. Therefore, for example, in PEKS scheme of Boneh et al [BOP04], it needs 2×200 storage bits and 200 transmission bits. On the other hand, to achieve a similar level of security of 248, PPEKS is needed 48 bits. Therefore, it needs 80 storage bits and 20 transmission bits.

Likewise, to achieve security of 249.3, 259.3, 267.4 ad 274.4 in PEKS-DDH, 200, 300, 400 and 500 bits q are required. On the other hand, to achieve security of 248, 260, 270 and 275 in PPEKS, 80, 120, 150, and 189 storage bits and 20, 24, 25 and 27 transmission bits are required, respectively, corresponding to (m,n,k): (4,8,12), (5,9,15), (6,11,14) and (7,12,15), respectively.

1. **Conclusion**

We have presented the general scheme of public key encryption with keyword search which was proposed by D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano. Based on this general scheme and the notations of the pseudo-inverse matrices, we have developed an effective and security scheme for searchable encryption. The proposed scheme mainly uses the linearly matrix operations, so that the complexity is low. In particular, the storage and transmission costs of proposed scheme are very low but the security is till assured, comparing with a scheme based on the Decisional Diffie-Hellman (DDH) assumption.

**Acknowledgments**

This work is done at University of Tartu, Estonia, under the financial support of the Erasmus Mundus NordSecMob-programme in Security and Mobile Computing. The authors would like to thank Professor Antti Ylä-Jääski and Professor Eiji Kujanpää, Co/Director of NordSecMob-programme. The authors would also like to express their appreciation to all members of the Estonian Crypto Group for their supports.

**References**

[Moo20] Moore, E.H. (1920). "On the reciprocal of the general algebraic matrix". Bulletin of the American Mathematical Society **26**: 394–395.

[Pen55] Penrose, Roger (1955). "A generalized inverse for matrices". Proceedings of the Cambridge Philosophical Society **51**: 406–413.

[BT03] Ben-Israel, Adi; Thomas N.E. Greville (2003). *Generalized Inverses*. Springer-Verlag.

[GC96] Goul, Gene H., Charles F. Van Loan (1996). *Matrix computations* (3rd edition ed.). Baltimore: Johns Hopkins. pp. 257–258.

[BOP04] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano (2004). *Public-key encryption with keyword search*. In C. Cachin, editor, Proceedings of Eurocrypt 2004

[MOV97] A.J.Menezes, P.C.V. Oorschot, and S.A. Vanstone (1997). *Handbook of applied Cryptography.* CRC Press.

[Bre04] Brent R. Waters ,  Dirk Balfanz ,  Glenn Durfee ,  D. K. Smetters. *Building an Encrypted and Searchable Audit Log*. In The 11th Annual Network and Distributed System Security Symposium 2004

[Bon01] D. Boneh and M. Franklin. *Identity-based encryption from the Weil pairing*. In Proc. CRYPTO 01, pages 213–229. Springer-Verlag, 2001. LNCS 2139

[Bell01] M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. *Key-privacy in public-key encryption*. Lecture Notes in Computer Science, 2248, 2001.

[Son00] D. X. Song, D. Wagner, and A. Perrig. *Practical techniques for searches on encrypted data*. In IEEE Symposium on Security and Privacy, pages 44–55, 2000.

[Golle04] Philippe Golle , Jessica Staddon , Brent Waters. *Secure Conjunctive Keyword Search over Encrypted Data*. ACNS 04: 2nd International Conference on Applied Cryptography and Network Security, 2004.

[Chang04] Y.-C. Chang and M. Mitzenmacher. *Privacy preserving keyword searches on remote encrypted data*. Cryptology ePrint Archive, Report 2004/051, Feb 2004. http://eprint.iacr.org/2004/051/.