www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

Vol. 03, Issue 04, April 2023, pp : 834-845

e-ISSN : 2583-1062

Impact Factor : 5.725

A SECURE CLOUD DATA SHARING PROTOCOL FOR ENTERPRISE SUPPORTING HIERARCHICAL SEARCH

Bala Harish Kumar[1], Dharani R[2]

[1,2]Computer Science & Engineering , Sri Krishna arts and science college , India.

ABSTRACT

Cloud storage becomes the priority for storing and sharing data for enterprise users. Encrypting prior to uploading data to the cloud is the best way to protect business secrets, however, it hinders the convenient operations on plaintexts, such as searching over the cloud data. In addition, employees in an enterprise have multiple layer structures and a higher layer employee should have the privilege to monitor the lower layer employees' data to check if these users violate the regulation without letting the employees be aware of. Public key encryption with keyword search (PEKS) is a well-known cryptographic primitive suitable for secure cloud storage, which supports keyword search without decryption in public key encryption settings. Unfortunately, no existing PEKS scheme supports the monitoring function without authorization from the sender. To address this issue, we propose a variant of PEKS named Hierarchical Public Key Encryption with Keyword Search (HPEKS) and provide a semi-generic construction utilizing a public key tree (PKTree) and a PEKS scheme. To better suit for the enterprise secret data sharing, we build an advanced HPEKS scheme, named designated-tester decryptable hierarchical public key encryption with keyword search (dDHPEKS), which enjoys stronger security and integrates the public key and symmetric key encryptions. We prove our dDHPEKS scheme secure under the security definition in the random oracle model. Particularly, it satisfies the security against outside offline keyword guessing attacks and furthermore, enjoys the *transparency* property so that the sender does not need to know the internal hierarchy structure of an enterprise in order to share encrypted data to the enterprise. Theoretical evaluation and concrete experiments show that our dDHPEKS scheme has comparable running efficiency with existing PEKS schemes.

Index Terms—public key encryption, keyword search, keyword guessing attacks, secret data sharing, cloud storage.

1. INTRODUCTION

With the quick advancement of figuring and correspondence innovation, information created by endeavors or firms extend dramatically with extremely high velocity. Because of the minimal expense of information the board, the public cloud administration (laptops) turns into the first concern for most undertakings. Insights show that most endeavors utilized something like one laptops [1]. It permits clients to get to the mass of information wherever utilizing capacity restricted cell phones by means of the web. One of the administrations provided by the computers is to store and oversee messages and reports shipped off clients. Be that as it may, the gamble of a security break likewise restricts a few ventures to utilize the laptops [1]. As per the measurements, security concern is the main issue for respondents [1]. Encoding before transferring is an immediate measure to counter-go after this gamble, however it restricts the application on the information.

A. Inspiration - The requirement for various leveled structure-

The Workplace Computerization (OA) framework is one of the uses of the distributed computing. Large number of messages are created regularly from an endeavor or the public authority. In reality, the OA administrations are frequently moved to the cloud specialist organization, since it possesses an immense benefit in expenses of information stockpiling and the board. Messages in the OA framework are frequently put away as plaintexts, which make it workable for foes to get to and in like manner would prompt financial misfortune or shamefulness for the general population.For example, a venture embraced by an undertaking would be deterred by its business rival on the off chance that the connected data spills ahead of time. In another model, the spillage of some administration planned strategy would give somebody an early advantage, which is unreasonable to the remainder of the general population.To stay away from this present circumstance, there ought to be a sensible access control system, which permits approved clients to get to the comparing messages and denies the entrance from unapproved client or the less favored clients. For instance, in an undertaking, the CEO (President) directs different workers, like head working official (COO), CFO (CFO), boss data official (CIO) and boss innovation official (CTO). The COO, CFO, CIO and CTO oversee a few workers, separately. To permit the Chief's admittance to the message sent from COO to CIO helps the President really deal with the venture. To deny the CFO's admittance to the message sent from CTO to President can lessen the gamble of data spillage.Despite the fact that the OA framework has an entrance control procedure , when the framework is broken in, all the data spills. Subsequently, the ciphertext-level access control procedure is a sensible and significant method for safeguarding information protection. The essential thought of the ciphertext-level access control technique is to scramble messages with various clients' public key, in what direction regardless of

whether the OA framework is broken in, just ciphertexts are uncovered, which releases less data. For this situation, the encryption conspire shell meet the accompanying prerequisites.

1) Searchability- It is wasteful to download and decode all the ciphertexts to discover a few explicit reports, for example, records relating to the watchword "contract". To effectively track down these records without spilling data about the substance of the report, a technique for looking over ciphertexts without unscrambling is required.

2) Access control in view of client need- A message or an undertaking in reality frequently should be shipped off and dealt with by various clients. The scrambled message ought to just be unscrambled by the client who has the relating or higher access-need. Taking into account the design of the recipients, it is sensible to apply a tree-like access control technique to unscramble the ciphertexts.

3) Transparency- It isn't required for a source Alice to know the inner construction of the association the recipient is in. At the point when Alice sends a scrambled message to a collector Sway, she just has to know Weave's legitimate public key. All in all, Alice doesn't have to know Weave's need in the venture neither who in the undertaking has a higher need than Bounce.

Apparently, there is no comparing cryptographic crude gathering the above prerequisites all the while. The public key encryption with catchphrase search (PEKS) upholds looking through some watchword over ciphertexts without unscrambling, yet it comes up short on system of access control. Quality based encryption (ABE) and Progressive character based encryption (HIBE) support ciphertext-level access control. It appears to be plausible to join ABE or HIBE with PEKS to build a reasonable accessible encryption conspire, notwithstanding, in ABE and HIBE, the shipper has to realize the collector's inner association structure.

Subsequently, it is fundamental for build a new cryptographic crude reasonable for the previously mentioned cloud information sharing situation.

B. Commitments

1) Public Key Tree- To determine the issue above, we present the public key tree (PKTree) into the accessible encryption and propose another idea named progressive public key encryption with catchphrase search (HPEKS). In HPEKS, it permits clients to look over ciphertexts encoded with their public keys. Uncommonly, on the off chance that there is a gathering of clients with a various leveled structure, the client with higher access consent can look over ciphertexts shipped off clients with lower access authorization. For instance, in the event that Alice manages Sway and Carlos, Alice can perform looking through tasks over ciphertexts scrambled with Weave's and Carlos' public keys.

2) Semi-nonexclusive HPEKS Development from PEKS- To exhibit the possibility, we give a semi-conventional development HPEKS utilizing the proposed PKTree strategy along with a current bilinear-matching based PEKS conspire in the writing.

3) Advanced HPEKS Plan dDHPEKS- To oppose outside disconnected catchphrase speculating assaults, we propose a high level HPEKS plot, named assigned analyzer decryptable progressive public key encryption with watchword search (dDHPEKS), in which just the assigned waiter can look for clients. Besides, our proposition coordinates PEKS and PKE, and that implies it upholds catchphrase search as well as decoding. Our dDHPEKS conspire partakes in a fascinating property, straightforwardness, implying that the shipper doesn't have to realize the inward ordered progression design of the association that the recipient is in prior to encoding a watchword for the beneficiary. All things being equal, the source needs to just encode the watchword under the public key of the planned recipient. Interestingly, HIBE or ABE coordinated with PEKS doesn't uphold this property.

4) Security and Proficiency- We give formal security definition for dDHPEKS, including secret key security, watchword protection and plaintext protection, and demonstrate our dDHPEKS conspire secure under the given security definitions. We dissect the running above of dDHPEKS hypothetically and execute it using C language and PBC library [2]. The investigation and trial results show that our dDHPEKS conspire has tantamount running above with existing PEKS plans.C. Related WorksThe thought of accessible symmetric encryption (SSE) and the principal accessible encryption conspire was proposed by Tune et al. [3] in 2000. It permits a client who has the looking through key to look through some watchword over ciphertexts without decoding. Nonetheless, Tune et al's. plot and different plans [4][5][6] in view of it have a typical limitation that it is difficult to impart the looking through key to other people, consequently are just reasonable for information proprietor itself to look over the ciphertexts. Boneh et al. [7] brought SE into the public key settings and proposed the main public key SE plot named Public Key Encryption with Watchword Search (PEKS), what breaks the obstruction of information sharing cutoff. In PEKS, an information proprietor (encrypter) plays out the encryption calculation utilizing a recipient's (searcher's) public key, and in converse, the collector creates a hidden entrance to look through some catchphrase over ciphertexts utilizing its mystery key.Different issues in PEKS stand out. Park et al. what's more, Golle et al. proposed plans named public key

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

Vol. 03, Issue 04, April 2023, pp : 834-845

e-ISSN : 2583-1062

Impact Factor : 5.725

encryption with conjunctive catchphrase search (PECKS) [8][9] which support beneficiaries to look for records containing every one of a few watchwords in just a solitary question. To empower a positioned rundown of the looking through outcome, another thought called multi-watchword positioned search (MRSE) was proposed [10] [11][12]. Bao et al. [13] and others [14] [15][16] proposed accessible encryption plans in multireceiver settings. To help fine-access control in PEKS, trait based encryption with catchphrase search (ABEKS) was proposed [17] [14]. Byun et al. [18] and Yau et al. [19] brought up that Boneh et al's. PEKS plot and other existing PEKS plans are weak under the new assault named disconnected watchword speculating assaults (KGA). They additionally underscored that practically all the current PEKS can't avoid the disconnected watchword speculating assaults given by inside enemies (IKGA), for example KGA given via looking through server. Tooth et al. [20] proposed a securechannel free PEKS conspire witch opposes the KGA effectively, in any case, can't avoid IKGA. Huang and Li proposed another idea called public key confirmed encryption with watchword search (PAEKS) [21][22] which opposes IKGA by denying enemies the capacity to encode catchphrases. He et al. [23] and Li et al. [24] bring PAEKS into Certificateless public key encryption and personality based encryption settings, individually. Chen et al. [25] and Chen et al. [26] additionally tackled the IKGA issue by using two server in the framework model.

c. Paper Association

In the following area, we momentarily present a few primers and the framework model of our HEPKS. We present the public key tree structure and its development in Segment III. We then propose our HPEKS plans in Segment IV, and dissect the security in Area V. We examine about the proficiency of our plan and contrast our plan and a few other related plans in Segment VI, lastly finish up the paper in Section

2. PERLIMENARIES

A. Bilinear Matching-Allow G1,G2 and GT to be three gatherings with same prime request p, g and h be any two generators of G1 and G2, separately. There is a guide eˆ : G1×G2 → GT from G1 and G2 to GT. We say eˆ is a bilinear matching guide [27] in the event that the accompanying circumstances are fulfilled:

• Bilinearity: ∀x,y ∈ Zp, eˆ(gx,hy) = eˆ(g,h)x·y.

• Non-decline: eˆ(g,h) 6= 1.

• Calculability: eˆ(g,h) is effectively processable.

B. Computational Diffie-Hellman (CDH) Presumption-Definition 1 (CDH Issue): Let eˆ : G1 × G2 → GT be a bilinear matching guide. Given g,gx,gy ∈ G1, the CDH issue is to compute gxy.

Definition 2 (CDH Suspicion [28]): The CDH supposition that will be that the likelihood that any probabilistic polynomial time (PPT) foe takes care of the CDH issue is irrelevant .

C. Decisional Direct (DLIN) Supposition-Let eˆ : G1 ×G2 → GT be a bilinear matching. Given u,ux ∈ G1,v,vy,h ∈ G2, the DLIN supposition that will be that the upside of recognizing hx+y from an irregular hr ∈ G2 is immaterial for any PPT foe [29]:

Pr[A(u,ux,v,vy,h,hr) = 1]| ≤ negl(λ).

D. Decisional Bilinear Diffie-Hellman (DBDH) Presumption-Definition 3 (DBDH Issue): Let eˆ : G1 × G2 → GT be a bilinear matching guide. Given g,gx,gy ∈ G1,h,hy,hz ∈ G2, the DBDH issue is to recognize eˆ(g,h)xyz from an irregular component R ∈ GT, where x,y,z ∈ Zp are obscure.

Definition 4 (DBDH Supposition): The DBDH supposition that will be that the upside of taking care of the DBDH issue for any PPT enemy is insignificant [28]:

AdvADBDH = |Pr[A(g,gx,gy,h,hy,hz,eˆ(g,h)xyz) = 1]− Pr[A(g,gx,gy,h,hy,hz,R) = 1]| ≤ negl(λ).

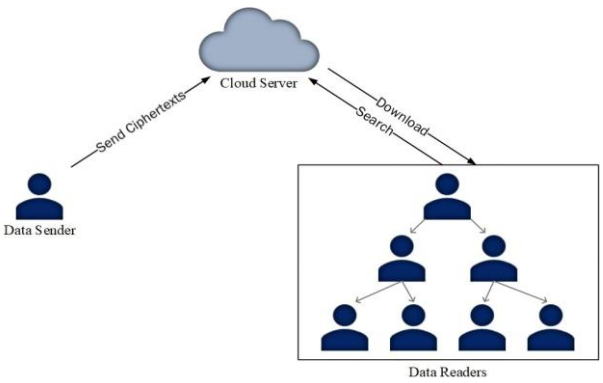

Fig. 1. Framework Model of Tree-based Progressive Multi-recipient Accessible Encryption

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

www.ijprems.com
editor@ijprems.com

Vol. 03, Issue 04, April 2023, pp : 834-845

e-ISSN : 2583-1062

Impact Factor : 5.725

E. Framework Model

Here we present the framework model of the HPEKS framework (Figure 1). There are three substances in the framework, meant by information source, information beneficiaries and cloud server, separately.

• Information shipper: It encodes information with a beneficiary's public boundary and sends the ciphertext to the collector by means of the cloud server.

• Information collectors: The information recipients have a tree-based progressive construction. Every information collector is signified by a hub in the tree. An information collector can look over ciphertexts shipped off itself and to every one of its kids, not the other way around.

• Cloud server: It gives the putting away and registering administrations for information recipients. The public boundaries of the gathering of recipients and the got ciphertexts will be put away by the cloud server. It additionally upholds recipients to run calculations to look over the ciphertexts

3. THE PUBLIC KEY TREE

To fabricate HPEKS conspire, we present another idea named public key tree (PKTree), which is a multi-way tree. Every hub of a PKTree contains a client's public data, and we say the client who has the comparing secret key is the proprietor of the hub. As referenced above, there is a gathering of information recipients. There is a manager Rt of the gathering, which at first arrangements the framework and holds the expert mystery key msk. Utilizing msk, Rt creates the mystery key skRt of itself and the root hub containing Rt's public data. After the instatement, Rt ought to add somewhere around one kid hub. For example, Rt will add a hub named Nodei for collector Ri.First and foremost, Rt assembles the public data IDi of Ri and creates a mystery key ski and a public boundary Pubi forRi utilizing (IDi,skRt). Furthermore, Rt signs Pubi to create a mark σRt(Pubi). At long last, Rt sends ski to Ri through a protected channel and adds the hub Nodei, containing Pubi and σRt(Pubi), into the PKTree. Notice that, to oppose enemies from manufacturing a kid hub, a mark for the substance is fundamental. Crafted by producing Nodei's youngster hubs is then moved to Ri. Along these lines, every collector can add its own youngster hubs itself, what shares the responsibility of building a PKTree. One more component of the PKTree is that a recipient can likewise add a grandkid hub straightforwardly. Extraordinarily, the boss Rt can add a kid hub for any of its relative hubs. Be that as it may, proprietors of two hubs in various branch can't add a kid hub for one another, regardless of which hub is nearer to the root than the other.Figure 2 shows an illustration of the PKTree. There is an undertaking and the proprietor of the endeavor regulates this PKTree. The proprietor arrangements the framework, distributes the framework boundary GP and holds the key msk covertly. The root hub of the PKTree will be alloted to the President of the venture (Level 1 in Fig. 2). The root hub contains the Chief's public data PubCEO and a mark Sigmsk(PubCEO) delivered by the venture proprietor. Notice that, PubCEO is produced by the venture proprietor and incorporates the President's character data, and the Sigmsk(PubCEO) fills in as a proof on the legitimacy of PubCEO.Like the Chief, the COO, CFO and CTO will be alloted with hubs by the President (Level 2 in Fig. 2). Every hub in the PKTree contains a worker's public data. For example, the CTO's public data is produced and endorsed by the President. The youngster hubs of the CTO's hub will be created by CTO rather than Chief. Along these lines, it permits each representative of the undertaking to add hubs into the PKTree and share the responsibility of building the tree. As referenced over, the CFO can add youngster hubs for NodeAlice and NodeBob, however not for NodeCarter nor NodeDavid.

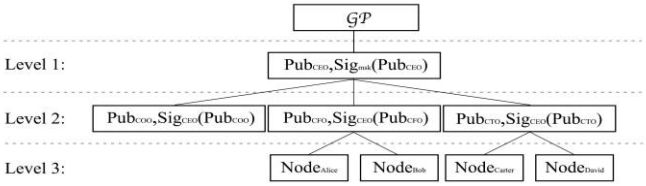

Fig. 2. An Illustration of Public Key Tree

A. Meaning of PKTree

A public key tree (PKTree) comprises of the accompanying four calculations.

• Setup(1λ): Given the security boundary 1λ, this calculation yields an expert mystery key for the framework msk and the worldwide boundary GP.

• Build Tree(GP,msk,IDRt,τ): Given the worldwide boundary GP, the expert mystery key msk, the personality IDRt of the root recipient Rt, and a period stamp τ, this calculation yields the root hub Node root of a public key Tree and the mystery key skroot of the hub. The Node root contains the personality IDRt and a mark σmsk endorsed with the expert mystery key msk. The mark will be utilized to confirm in the event that the Node root is substantial and bound to the personality of the root collector Rt.

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

Vol. 03, Issue 04, April 2023, pp : 834-845

e-ISSN : 2583-1062

Impact Factor : 5.725

• Add Node(GP,Tree,ski,IDj,τj): Given GP,Tree, the mystery key ski of hub Nodei, a character IDj and a period stamp τ, this calculation yields a youngster hub Nodej of Nodei and Nodej's relating secret key skj. Notice that the time stamp remembered for a youngster hub is substantial if and provided that it is more current than that in the parent hub. Like the Build Tree calculation, the recently added kid hub Nodej contains the character IDj and a mark σi,j endorsed with the parent hub's mystery key ski.

• Delete Node(GP,Tree,ski,τi0): Given GP,Tree, the mystery key ski of hub Nodei and another time stamp τi0, this calculation erases all the youngster hubs of Nodei and recharge the Nodei utilizing the new time stamp τi0. Since the τi0 will be fresher than any of the youngster hubs, all the old kid hubs will be invalid. Consequently, it erases the youngster hubs.

• VerifyTree(GP,Tree): Given GP and Tree, this calculation yields 1 assuming the time stamp of each and every hub is more up to date than its parent hub and the mark of each and every hub is truly endorsed by its parent hub (the root hub is endorsed by the expert key msk), and yields 0 in any case.

B. The proposed PKTree

In view of the bilinear matching, we give a substantial PKTree which will be used to fabricate the HPEKS plans.

• Setup(1λ): Select three gatherings G1,G2,GT with a similar prime request p and a bilinear matching eˆ : G1 × G2 → GT. Haphazardly select an expert mystery key msk = k ∈ Zp and register the expert public key MPK = gk ∈ G1, where g is a generator of G1.

At last, this calculation yields the worldwide boundary GP = {G1,G2,GT,p,g,h,eˆ MPK,H∗,H1,H2} in which h is a generator of G2 and H∗ : {0,1}∗ → G2,H1 : {0,1}∗ → G2,H2 : {0,1}∗ → Zp are cryptographic hash capabilities.

• Build Tree(GP,msk,IDRt,τ): Given GP,msk, Rt's character IDRt and a period stamp τ, Rt creates the primary hub of the tree as follow.

Haphazardly select a salt ς, and produce Rt's public

boundary PubRt = (pkRt,IDRt) and secret key

msk

skRt = H1(IDRt) , where pkRtand

IDRt = (IDRt,ς).

Sign IDRt with the expert mystery key msk to produce a mark σ0,Rt = Sigmsk(PubRt) =

(σ0,Rt,1,σ0,Rt,2) = (H∗(PubRtkτ)msk,τ).

Set the root hub of the PKTree as NodeRt =

(PubRt,σRt) and transfer Tree = {(NodeRt,P0)} to the cloud. Here P0 demonstrates the place of the NodeRt, for example the root, in the Tree.

Comment. The salt ς is utilized to randomize the beneficiary's public boundary. Clients can confirm the legitimacy of the root hub with a money order whether the accompanying condition holds:

eˆ(g,σ0,Rt,1) = eˆ(MPK,H∗(PubRtkσ0,Rt,2)).

• Add Node(GP,Tree,ski,IDj,τj): This calculation is controlled by a recipient I, who claims a public boundary in the Tree, to produce a youngster hub for another collector j. Given GP, Tree, the beneficiary I's mystery key ski, the lower layer collector j's character IDj and the time stamp τj, the calculation runs the accompanying advances and results the new hub Nodej and its relating secret key skj:

-Haphazardly select a salt ςj and produce people in general

boundary Pubj = (pkj,IDj) and secret key sk =

H1(IDj)H2(ski), where pkj = gH2(skj)

(IDj,ςj);

Sign IDj with ski and produce a mark

σi,j = Sigski(Pubj) = (σi,j,1,σi,j,2) = (H∗(Pubjkτj)H2(ski),τj);

Set the kid hub as Nodej = (Pubj,σi,j).

Comment. The time stamp τj of the new hub Nodej should be more up to date than that of the parent hub Nodei. Clients can confirm the legitimacy of this hub with a money order whether the accompanying condition holds:

eˆ(g,σi,j,1) = eˆ(pki,H∗(Pubjkσi,j,2)).

• Delete Node(GP,ski,Node : This calculation disavows all the youngster hubs of Nodej. Given GP, the beneficiary's mystery key ski, the youngster hub Nodej, this calculation runs as follows:

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

Vol. 03, Issue 04, April 2023, pp : 834-845

e-ISSN : 2583-1062

Impact Factor : 5.725

Get the ongoing time stamp τj0 and create another mark σi,j0 = Sigski(Pubj) =

Produce another Hub ;

Supplant Nodej with Node0j in the Tree and erase all the youngster hubs of Nodej. Note that, the mystery key of the Node0j isn't changed and same as that of Nodej.

Comment. A credulous technique to repudiate the kid hubs is to straightforwardly erase them from Nodej. Notwithstanding, it can't be demonstrated that the kid hubs have been without a doubt erased, in light of the fact that the marks connected to them and the time stamps of the youngster hubs are as yet substantial, which might deceive different clients. An improved answer for the issue is to refresh the time stamp of the parent hub (for example Nodej). As per our plan of PKTree, time stamps of youngster hubs ought to be fresher than that of the parent hub. Subsequently, in the event that we update the parent hub's time stamp with the most recent one (as well as the relating mark on the new time stamp), all the kid hubs are discredited right away. Nonetheless, by and large we don't have to erase all the youngster hubs. Consequently, we plan a two-layer-per-beneficiary construction, for example every recipient has two layer hubs. The upper-layer hub is alloted by the beneficiary's parent hub and the lower-layer hubs are produced by the actual collector. Each lower-layer hub is related with a youngster hub. In the event that the beneficiary necessities erase one of the kid hubs, it essentially refreshes the related lower-layer hub, to make the time stamp of the lower-layer hub more current than the kid hub. In such a manner, it finishes the erasure of the to-be-erased kid hub, and keeps the other kid hubs unaffected.

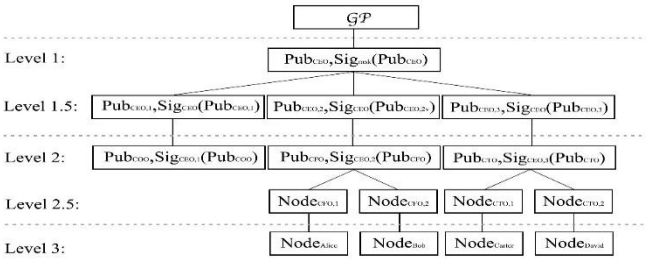

Fig. 3. The Substantial PKTree

• Verify Tree(GP,Tree): The Tree is substantial if and provided that, for all hubs in the tree, the conditions eˆ(g,σ0,Rt,1) = eˆ(MPK,H∗(PubRtkσ0,Rt,2))

eˆ(g,σi,j,1) = eˆ(pki,H∗(Pubjkσi,j,2))

hold, where Rt is the addendum of the root hub and i,j show the addendums of each sets of parent-youngster hubs.

C. Use of PK Tree Figure 3 shows the development of the PK Tree in reality situation. Prior to building the PK Tree, the worldwide boundary GP ought to be given by the undertaking proprietor. Somewhat not quite the same as the previously mentioned model, a recipient could possess different hubs, for example the Chief possesses a Level-1 hub (doled out by the venture proprietor) and three Level-1.5 hubs (relegated without help from anyone else) in the PK Tree.To erase a hub, for example the CTO's hub, as displayed in Fig. 4, the President runs the Delete Node calculation to refresh the Level1.5 hub NodeCEO,3 which is the parent hub of the CTO's hub. The public boundary PubCEO,3 in the refreshed hub NodeCEO,3 isn't changed however the relating mark SigCEOnew is refreshed with the most recent time stamp . Since the is fresher than the time stamps of its youngster hubs, all it

4. THE PROPOSED HPEKS PLANS

In this segment, in light of the proposed PK Tree, we acquaint how with fabricate a HPEKS plot by giving a semi-nonexclusive HPEKS conspire. To keep the integrality of the encoded catchphrases and plaintext, we give an upgraded concrete HPEKS conspire which supports decoding ciphertexts.

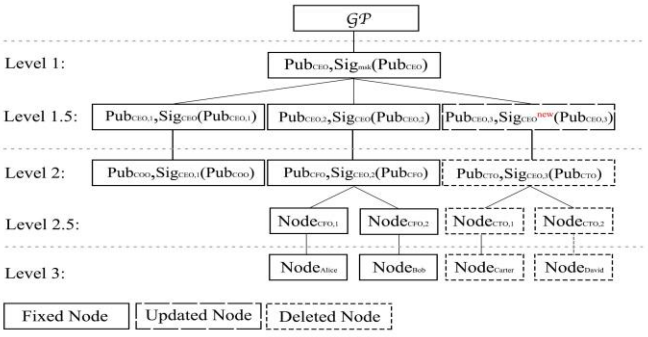

Fig. 4. Erase Hubs from the PKTree

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

Vol. 03, Issue 04, April 2023, pp : 834-845

e-ISSN : 2583-1062

Impact Factor : 5.725

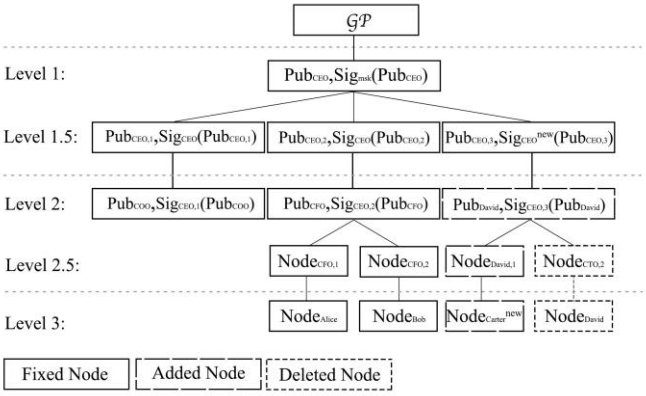

Fig. 5. Add Hubs into the PKTree

A. The Semi-Conventional HPEKS Plan from PEKS-Because of the similarity of PKTree, we can build a semi-nonexclusive HPEKS conspire from matching based PEKS plans. Let PEKS = {KeyGen0,Encrypt0,Trapdoor be a matching based PEKS conspire in which the mystery key sk is a component of Zp and the public key is of the structure gsk.

- Encrypt(GP,Tree,Nodej,w): Parse the beneficiary's hub Nodej as (pkj,IDj,σi,j) and confirm the legitimacy of the public boundary pkj,IDj and the mark σi,j. Run the PEKS Encrypt0(GP,pkj,w) calculation and return the created ciphertext Cw.
- Trapdoor(GP,skj,w): This calculation returns the hidden entrance Tw produced by PEKS Trapdoor0(GP,H2(skj),w).
- Test(GP,C,Tw): This calculation returns the hidden entryway Tw created by PEKS Test0(GP,C,Tw).

Comment. The Arrangement, BuildTree, AddNode, DeleteNode and VerifyTree calculations of the semi-nonexclusive HPEKS plot are equivalent to those in Segment III-B and in this way are discarded here.

B. assigned analyzer Decryptable HEPKS Plan-The semi-conventional HPEKS plot upholds a gathering of clients to designate the cloud server to look through some watchword over the got ciphertexts without unscrambling. Uncommonly, a recipient can look over ciphertexts shipped off beneficiaries regulated by him. Nonetheless, it depends on the current PEKS plans and the customary PEKS plans don't uphold the unscrambling capability. Albeit the PKE/PEKS plot proposed by Baek et al. [30] upholds unscrambling of ciphertexts, there is a limitation that the plaintext in the PKE/PEKS conspire is length-restricted. To determine this issue, we propose a high level plan named assigned analyzer decryptable progressive public key encryption with catchphrase search (dDHPEKS), in which plaintext of variable length can be encoded and the comparing ciphertext is decryptable. Moreover, just the assigned analyzer can test whether a ciphertext contains the very catchphrase as that in the questioned hidden entryway.

• Setup(1λ) : Given a security boundary λ, this calculation arrangements the framework by means of running the accompanying advances.

Pick three gatherings G1,G2,GT of same prime request p with a bilinear matching eˆ : G1 × G2 → GT.

Arbitrarily select two generators g ∈ G1, h ∈ G2. - Haphazardly select msk = k ∈ Zp and process MPK = gk.

Select hash capabilities H∗ : {0,1}∗ → G2, H :

{0,1}∗ → Zp, H1 : {0,1}∗ → G2, H2 : G2 → Zp, H3 : {0,1}∗ → {0,1}'1, H4 : {0,1}∗ → G2,

H5 : {0,1}∗ → {0,1}'2, where '1,'2 are two fixed lengths.

Yield the msk and the GP = (G1, G2, GT, p, eˆ, g, h, MPK, H∗, H, H1, H2, H3, H4, H5).

• KeyGenserver(GP): Given the worldwide boundary GP, this calculation arbitrarily chooses sks ∈ Zp as the assigned test server's mystery key and processes the comparing public key pks = hsks. Yield the server's public/secret key pair (pks,sks).

• Encrypt(GP,Tree,Nodej,w,M): Haphazardly select r,s ∈ Zp,K ∈ {0,1}'1 and scramble the catchphrase w and message M as underneath:

,C4 = K ⊕ H3(pksj),C5 = gs,C7 = AESEncK(M), C6 = H4(C1,C2,C3,C4,C5,H5(C7))s.

Yield the ciphertext Cw,M = (C1, C2, C3, C4, C5, C6, C7).

• Decrypt(GP,skj,C): Parse C as C = (C1,...,C7). Return ⊥ if both of the situations eˆ(C5,H3(C1,...,C5,H5(C7))) = eˆ(g,C6), eˆ(C2,h) = eˆ(g,C3)

doesn't hold, and return the plaintext M in any case, where

M = AESDec .

• Trapdoor(GP,skj,pks,w): Haphazardly select t ∈ Zp and create secret entryway as follows:

,• Test(GP,C,T,sks): Parse C as (C1,...,C7) and parse T as (T1,T2,T3). The calculation yields 1 if the condition holds, and 0 in any case.

Comment. The BuildTree, AddNode and ReplaceChild calculations are equivalent to those in the Semi-Nonexclusive HPEKS plan, and in this way overlooked here.Our plan is profitable over HIBE (with catchphrase search) as in it partakes in the property of straightforwardness. To share an encoded record to various recipients in an order, a shipper in our plan doesn't have to know the progressive system of these beneficiaries. It just has to scramble the record under the public key of the collector at the least level. Conversely, a source in HIBE has to know the personality progressive system.

5. SECURITY INVESTIGATION

In this part, we give the dangers model of HPEKS, then, at that point, give the conventional security definitions, lastly demonstrate our proposition is secure under the security definitions.

A. Security Models

1) Secret Key One-Way Protection from Intrigue: Here we consider the vital security of the proposed PKTree structure. The accompanying two games are given to demonstrate the semantic security of PKTree against low layer conspiracy and same layer arrangement assaults, separately.

Game K1: Protection from low layer arrangement assaults:

• Arrangement: The challenger C sends the foe A the worldwide boundary GP and the tested hub Nodei.

• Stage 1: A can issue all things considered qE inquiries to extricate prophet OE to get Nodei's kid hubs secret keys :

OE(IDj): Given a character, the prophet returns the relating hub Nodej and secret key skj.

• Surmise: A results a mystery key skA and dominates the match if skA = ski.

Let AdvAK1 = Pr[skA = ski] indicate the benefit of A to dominate Match K1.

Game K2: Protection from same layer plot assaults:

• Arrangement: The challenger C sends the foe A the worldwide boundary GP and the parent hub Nodei.

• Stage 1: A can issue all things considered qE questions to extricate prophet OE to get Nodei's youngster hubs secret keys :

OE(IDj): Given a character, the prophet returns the relating hub Nodej and secret key skj.

• Challenge: A picks a personality IDc as the test character and sends it to C. The imperative is that IDc can't be submitted to OE in Stage 1. C returns the relating kid hub NodeIDc of Nodei to A.

• Stage 2: An issues questions to the prophet same as in Stage 1 with the limitation that IDc can't show up in the OE.

• Surmise: A results a mystery key skA and dominates the match if skA = skIDc where ski is the mystery key of NodeIDc.

Let AdvAK2 = Pr[skA = skIDc] signify the benefit of A to dominate Match K2.

Definition 5: A dDHPEKS conspire is secure against key plot assaults if for any PPT foe A, AdvAK1 and AdvAK2 are both irrelevant.

2) Watchword Protection against Picked Catchphrase Assaults: We give the accompanying security game to characterize the watchword security against picked catchphrase assaults.

Game W1: Secret entryway protection:

• Arrangement: The challenger C sends the enemy A the worldwide boundary GP and the parent hub Nodei.

• Stage 1: A can issue all things considered qE and qT inquiries to extricate prophet OE and secret entryway prophet OT, separately.

OE(IDj): Given a character IDj, the prophet returns the relating hub Nodej and secret key skj.

OT(IDj,w): Given a character IDj and a catchphrase w, the prophet returns a comparing hidden entrance TIDj,w.

• Challenge: A picks a personality IDc and two catchphrases w0,w1. The requirement is that IDc can't be submitted to OE in Stage 1. C produces the hub NodeIDc and haphazardly chooses a piece b ∈ {0,1}. In the subsequent stage, C produces a secret entryway Twb. At long last, C returns NodeIDc and Twb to A.

• Stage 2: An issues questions to the prophets same as in Stage 1.

• Surmise: A results a piece bA and dominates the match if b0 = b.

Let AdvAW1 = Pr[wA = wc] indicate the benefit of A to dominate Match W1.

Definition 6: A dDHPEKS conspire is IND-TW-CPA secure if for any PPT enemy A, the benefit AdvAW1 is insignificant.

Game W2: Ciphertext vagary against picked catchphrase assaults:

• Arrangement: Same as that in Game W1.

• Stage 1: Same as that in Game W1.

• Challenge: A picks a character IDc, two watchwords w0,w1 and a plaintext M. The imperative is that IDc can't show up in OE in Stage 1. C produces the hub NodeIDc and haphazardly chooses a piece b ∈ {0,1}. In the following stage, C produces a ciphertext C[wb,M]. At last, C returns NodeIDc and C[wb,M] to A.

• Stage 2: A still can give questions to the prophet same as in stage 1 with the imperative that IDc can't show up in OE.

• Surmise: A results a piece b0 and dominates the match if b0 = b.

Let mean the upside of A to dominate Match W2.

Definition 7: A dDHPEKS plot is IND-CW-CPA secure if for any PPT enemy A, the benefit AdvAW2 is immaterial.

3) Plaintext Protection against Picked Ciphertext Assaults: The accompanying game is to characterize the semantic protection from picked ciphertext assaults.

Game M: Ciphertext lack of definition against picked plaintext assaults:

• Arrangement: Same as that in Game W1.

• Stage 1: A can issue all things considered qE questions to the Concentrate Prophet OE underneath.

- OE(IDj): Given a personality IDj, the prophet returns the comparing hub Nodej and secret key skj.

• Challenge: A picks a character IDc, a catchphrase w and two plaintext M0,M1. The limitation is that IDc can't be submitted to OE in Stage 1. C produces the hub NodeIDc and haphazardly chooses a piece b ∈ {0,1}. In the following stage, C produces a ciphertext C[w,Mb]. At last, C returns NodeIDc and C[w,Mb] to A.

• Stage 2: An issues questions to the prophet same as in Stage 1 with the requirements that IDc can't be submitted to OE and C[w,Mb] can't show up in OD.

• Surmise: A results a piece b0 and dominates the match if b0 = b.

Let indicate the benefit of A to dominate Match M.

Definition 8: A dDHPEKS conspire is IND-CCA secure if for any PPT enemy A, the benefit AdvAM is insignificant

6. COMPARISON AND EXPERIMENTS

We compare the proposed dDHPEKS scheme to previous searchable encryption schemes [7, 30, 31] to determine its running efficiency. The running efficiency of our proposal and the compared schemes is comparable, as shown in Table I. It is reasonable to sacrifice some running efficiency in order to attain a higher level of security and additional functionality. Our dDHPEKS scheme is the only one of the four that can integrate PKE and PEKS as well as withstand external keyword guessing attacks. Furthermore, our proposal, based on the PKTree, provides support for node supervision, something that has not been possible with previous PEKS schemes.On a laptop with a 2.3 GHz Intel Core i5 CPU and 8 GB 2133 MHz LPDDR3 memory, we used the C language and the PBC library [2] to implement our dDHPEKS scheme as well as the compared schemes referred to as BDOP-PEKS [7], BSS-PKE/PEKS [30], and HL-PEKS [31], respectively. The system was initialized with a Type-A pairing, which possesses the same level of security as a 1024-bit RSA encryption. The first step in applying the keyword search to a dataset is to extract keywords. A keyword ought to be attached to each record in the dataset. The proposed encryption method will be used to encrypt the keyword, and another method, such as AES, will be used to encrypt the record. Upset record is generally used to help effective scrambled information search. In essence, the inverted index consists of a list of (encrypted) keywords, with each keyword linked to a queue of records that include the keyword. In this framework, the catchphrases can be found in a typical word reference with a high likelihood and different bit length of watchwords hypothetically don't influence the proficiency of the program, in light of the fact that every catchphrase will be hashed into a fixed-length string before encryption. Consequently, we pick the Oxford word reference as the watchword space in the execution and every one of the words in it are taken as contribution of the connected calculations of the plan.Our dDHPEKS scheme's encryption algorithm is slower than that of the other three, as shown in Figure 6. The reason for this is that encrypting the session key K, which is not present in BDOPPEKS or HL-PEKS schemes, consumes nearly half of the computing power.TABLE I COMPARISONS WITH PEKS SCHEMES IN LITERATURE Scheme Encrypt Trapdoor

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

Vol. 03, Issue 04, April 2023, pp : 834-845

e-ISSN : 2583-1062

Impact Factor : 5.725

Test KGA INT INT/L HMSE BDOP-PEKS[7] 2ExpG1 + 2Hash + 1Pairing 1ExpG1 + 1Hash 1Hash + 1Pairing No No No No BSS-PKE/PEKS[30] 3ExpG1 + 4Hash + 1Pairing 1ExpG1 + 1Hash 1Hash The plans are able to withstand external keyword guessing attacks;INT: The plaintext and the encrypted keyword are combined in the schemes;INT/L: The integrated plaintexts make the schemes adaptable to any length; HMSE: Hierarchical multi-user keyword search is supported by the schemes.Because it is uncertain and influenced by the length of the plaintext, the overhead of our dDHPEKS scheme's AES encryption algorithm was not included in the experiment results.Figure 7 shows that our dDHPEKS conspire claims a proficient secret entryway calculation, which is like the BDOP-PEKS and the BSS-PKE/PEKS plans and more productive than the HLPEKS plot.Figure 8 depicts the overhead of the four schemes' test algorithms. The BDOP-PEKS and BSSPKE/PEKS schemes have an average overhead of nearly 0.75 seconds, the HL-PEKS scheme takes 2.00 seconds, and our dDHPEKS scheme takes approximately 2.85 seconds to search over 1,000 ciphertexts for a given keyword. Although the first two methods are quicker, they are unable to withstand offline keyword guessing attacks from the outside. Our dDHPEKS scheme's search efficiency is comparable to that of HL-PEKS at the same security level.Take for instance the situation in which the sender transmits an encrypted keyword to a number of receivers arranged in a hierarchy. For example, receiver R2 is the child-node of receiver R1, receiver R3 is the child-node of receiver R2, and so on. In the event that we utilize a customary PEKS conspire, the source needs to scramble the watchword on different occasions (or utilize a re-encryption system to re-encode a current ciphertext to new ciphertexts under other collectors' public keys), and sends ciphertexts to the particular beneficiaries. As a result, the amount of time required for encryption is proportional to the number of receivers. However, if we apply the scenario to the dDHPEKS scheme that has been proposed, the sender only needs to encrypt the keyword for the receiver at the lowest level among all the receivers. After that, all of the receivers would be able to test the encrypted keyword because a parent node in our scheme has the privilege of accessing the encrypted data of its child-node. Therefore, regardless of the number of recipients, the sender must spend the same amount of time encrypting this message. The comparison of the encryption time in this scenario is shown in Figure 9.To look over ciphertexts for a beneficiary at a low level (say, level 5), a recipient at a significant level (say, level 1) requirements to create secret key (and the hidden entrance) for every hub along the way from the general collector to the low level beneficiary, and the computational expense is in this manner directly with the quantity of levels between the two beneficiaries. Our approach demonstrates that the additional operating expense is negligible in comparison to the encryption time. If there are 100 levels between the two receiver nodes, the time it takes to generate the additional secret keys and trapdoors is close to 0.13s, as shown in Figure 10, which is significantly less than the 55s-70s encryption time of the compared schemes. Because there are typically fewer than ten levels in an organization or business, the cost of creating the trapdoor may be overlooked.

7. CONCLUSION

In this paper, we proposed another convention named progressive public key encryption with catchphrase search (HPEKS), which upholds a client to screen its youngster clients. We presented a public key tree (PKTree) structure and utilized it to fabricate a semi-conventional HPEKS development. We likewise proposed a high level HPEKS conspire dDHPEKS, which incorporates PEKS

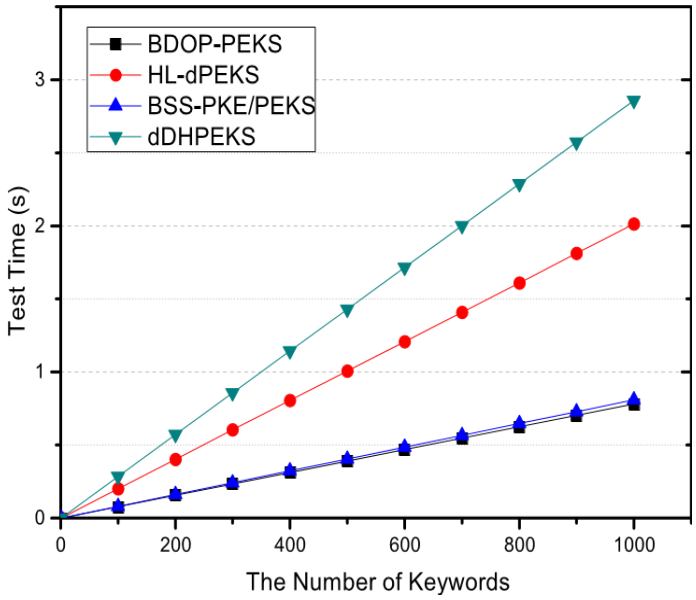

Fig. 8. Examination of Test Calculations

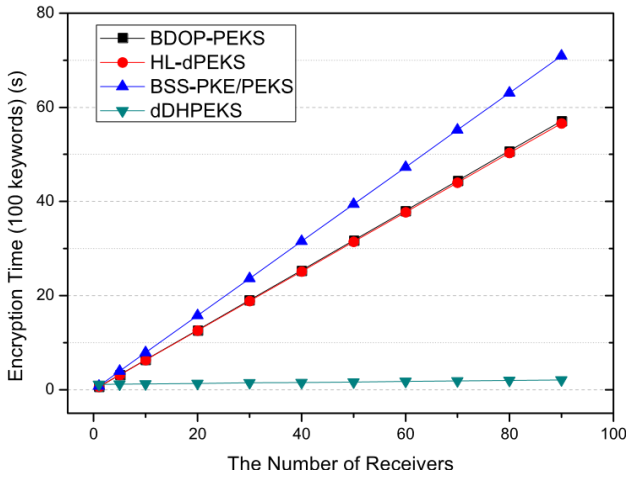

Fig. 9. Encryption Time alongside Collector Number Expanding

furthermore, PKE, and can oppose outside disconnected catchphrase speculating assaults. Tests show that our dDHPEKS conspire has practically identical effectiveness with existing related PEKS plans.

8. REFERENCES

[1] W. Kim, "Cloud computing trends: 2018 state of the cloud survey," URL https://www.rightscale.com/blog/cloud-industryinsights/cloud-computing-trends-2018-state-cloudsurvey, [Online], 2018.

[2] B. Lynn and et al, "Pairing-based cryptography library," https://crypto.stanford.edu/pbc/, 2013.

Fig. 10. Time of Generating Child-node's Secret Keys and Trapdoors

[3] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in Proceedings of IEEE Symposium on Security and Privacy. S&P 2000, 2000, pp. 44–55.

[4] Y.-C. Chang and M. Mitzenmacher, "Privacy preserving keyword searches on remote encrypted data," in International Conference on Applied Cryptography and Network Security. Springer, 2005, pp. 442–455.

[5] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: improved definitions and efficient constructions," Journal of Computer Security, vol. 19, no. 5, pp. 895–934, 2011.

[6] K. Kurosawa and Y. Ohtaki, "Uc-secure searchable symmetric encryption," in International Conference on Financial Cryptography and Data Security. Springer, 2012, pp. 285–298.

[7] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, Public Key Encryption with Keyword Search. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 506–522.

[8] D. J. Park, K. Kim, and P. J. Lee, "Public key encryption with conjunctive field keyword search," in International Workshop on Information Security Applications. Springer, 2004, pp. 73–86.

[9] P. Golle, J. Staddon, and B. Waters, "Secure conjunctive keyword search over encrypted data." in Applied Cryptography and Network Security, Second International Conference, ACNS 2004, Yellow Mountain, China, June 8-11, 2004, Proceedings, 2004, pp. 31–45.

[10] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacypreserving multi-keyword ranked search over encrypted cloud data," IEEE Transactions on parallel and distributed systems, vol. 25, no. 1, pp. 222–233, Jan 2014.

[11] Z. Fu, X. Sun, Z. Xia, L. Zhou, and J. Shu, "Multikeyword ranked search supporting synonym query over encrypted data in cloud computing," in Performance Computing and Communications Conference (IPCCC), 2013 IEEE 32nd International. IEEE, 2013, pp. 1–8.

[12] H. Li, D. Liu, Y. Dai, T. H. Luan, and X. S. Shen, "Enabling efficient multi-keyword ranked search over encrypted mobile cloud data through blind storage," IEEE Transactions on Emerging Topics in Computing, vol. 3, no. 1, pp. 127–138, 2015.

[13] F. Bao, R. H. Deng, X. Ding, and Y. Yang, "Private query on encrypted data in multi-user settings," in International Conference on Information Security Practice and Experience. Springer, 2008, pp. 71–85.

[14] F. Zhao, T. Nishide, and K. Sakurai, "Multi-user keyword search scheme for secure data sharing with finegrained access control," in International Conference on Information Security and Cryptology. Springer, 2011, pp. 406–418.

[15] Y. Yang, H. Lu, and J. Weng, "Multi-user private keyword search for cloud computing," in Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on. IEEE, 2011, pp. 264–271.

[16] C. Van Rompay, R. Molva, and M. Onen, "Multi-user¨ searchable encryption in the cloud," in International Information Security Conference. Springer, 2015, pp. 299–316.

[17] M. Hattori, T. Hirano, T. Ito, N. Matsuda, T. Mori, Y. Sakai, and K. Ohta, "Ciphertext-policy delegatable hidden vector encryption and its application to searchable encryption in multi-user setting," in IMA International Conference on Cryptography and Coding. Springer, 2011, pp. 190–209.

[18] J. W. Byun, H. S. Rhee, H.-A. Park, and D. H. Lee, "Offline keyword guessing attacks on recent keyword search schemes over encrypted data," in Workshop on Secure Data Management. Springer, 2006, pp. 75–83.

[19] W.-C. Yau, S.-H. Heng, and B.-M. Goi, "Off-line keyword guessing attacks on recent public key encryption with keyword search schemes," in International Conference on Autonomic and Trusted Computing. Springer, 2008, pp. 100–105.

[20] L. Fang, W. Susilo, C. Ge, and J. Wang, "A secure channel free public key encryption with keyword search scheme without random oracle," in Cryptology and Network Security, International Conference, CANS 2009, Kanazawa, Japan, December 12-14, 2009. Proceedings, 2009, pp. 248–258.

[21] Q. Huang and H. Li, "An efficient public-key searchable encryption scheme secure against inside keyword guessing attacks," Information Sciences, vol. 403-404, pp. 1 – 14, 2017.

[22] An efficient public-key searchable encryption scheme secure against inside keyword guessing attacks," Cryptology ePrint Archive: Report 2018/007, 2018.

[23] D. He, M. Ma, S. Zeadally, N. Kumar, and K. Liang,Certificateless public key authenticated encryption with keyword search for industrial internet of things," IEEE Transactions on Industrial Informatics, vol. 14, no. 8, pp. 3618–3627, 2017.

[24] H. Li, Q. Huang, J. Shen, G. Yang, and W. Susilo,"Designated-server identity-based authenticated encryption with keyword search for encrypted emails," Information Sciences, vol. 481, pp. 330 – 343, 2019.

[25] R. Chen, Y. Mu, G. Yang, F. Guo, and X. Wang, "Dualserver public-key encryption with keyword search for secure cloud storage," IEEE Transactions on Information Forensics and Security, vol. 11, no. 4, pp. 789–798, April 2016.

[26] R. Chen, Y. Mu, G. Yang, F. Guo, X. Huang, X. Wang, and Y. Wang, "Server-aided public key encryption with keyword search," IEEE Transactions on Information Forensics and Security, vol. 11, no. 12, pp. 2833–2842, Dec 2016.

[27] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in Annual International Cryptology Conference. Springer, 2001, pp. 213–229.

[28] J. Katz and Y. Lindell, Introduction to modern cryptography, Second Edition. CRC press, 2014.

[29] D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," in Annual International Cryptology Conference. Springer, 2004, pp. 41–55.

[30] J. Baek, R. Safavi-Naini, and W. Susilo, On the Integration of Public Key Data Encryption and Public Key Encryption with Keyword Search. Springer Berlin Heidelberg, 2006.C. Hu and P. Liu, "A secure searchable public key encryption scheme with a designated tester against keyword guessing attacks and its extension," in International Conference on Computer Science, Environment, Ecoinformatics, and Education. Springer, 2011, pp. 131–136