Research Article

Implementation of homomorphic Encryption scheme with Diffie-Hellman and RSA hybrid algorithm to improve the security in cloud networks

Dr Nelli. Chandrakala[1*], Dr. Y. Vijay Bhaskar Reddy[2], Dr. Laxmaiah Kocharla[3], Dr.M. Venkateswara rao[4]

[1*]Department of Computer science and Engineering, TKR College of engineering and Technology, Meerpet, Hyderabad, TG, India kala5136@gmail.com
[2]Department of Computer science and Engineering, Lakireddy Bali Reddy College of Engineering, L.B. Reddy Nagar, Mylavaram,AP,India yaramala.vijay@gmail.com
[3]Department of Computer science and Engineering, School of Engineering, Sreenidhi University Yamnampeta, Hyderabad, TG, India hello.laxman1235813@gmail.com
[4]Department of Computer science and Engineering, Vignana Bharathi Institute of Technology, Hyderabad, TG, India venkat2m2@gmail.com

Abstract

Cloud computing is an emerging field that provides users with different types of facilities. While cloud computing is driven by the general concept of storage, virtualization and interconnectivity and data processing power, resources are communicated between various computers and other devices via the Internet. A virtual machine is used for verification and encryption / decryption of the user in good enough condition to provide the most cloud network with security. However we have adopted hybrid cryptography to ensure secret message exchange because of the lack of key sharing and key management features. RSA uses random prime numbers P and Q, which are shared with the other side of the communication by value N after multiplication. RSA security depends on the primary factorization of N. This proposed algorithm secretly generates N's value so that security levels in the new cryptography system based on RSA are increased. When authenticating the user, we use modified Diffie-Hellman algorithm. Slowly but surely, as revealed, the authenticated user is the first part of the algorithm by random value P and Q. Create a secret key on both sides. Unknown or not sent secret keys through the communication channel, so eavesdropper has no way of gaining the value of the secret key. The secret key is multiplied to P and therefore new P cannot be identified to the attacker if the value of P is affected. This value of P and Q is actually secretly generated on the user side and is not shared via a communication channel.

Keywords: Diffie Hellman key exchange algorithm, Cloud Computing, Decryption, Encryption, Homomorphic Encryption Scheme.

1. Introduction

Cloud computing provides numerous user services, including the option available to download storage and calculate large volumes of cloud data. In order to use cloud computing, however, users should rely on the cloud service providers and share their data. One way of ensuring data privacy is before uploading to the cloud to encrypt data. But even so, if the users want to calculate the data, they should download and decrypt the data, thereby reducing the main benefits of using cloud services. In earlier cloud computing environments, developers were required to deploy licensed software, which introduced several limitations. One of the most significant challenges was the time-consuming process involved in moving applications between virtual machines. In public cloud systems, providers allocate computing and storage resources to users, making it essential to ensure data security throughout the entire lifecycle. A key objective of virtualization is to maintain system reliability, ensuring that the failure of one virtual machine does not impact the performance of others. At the same time, improving resource efficiency remains an important goal. While virtualization provides many benefits, it also introduces additional security concerns within cloud environments. Several threats must be considered, including VM sprawl, VM hopping, and insecure live migration. These vulnerabilities can Compromise system integrity if not properly managed. For example, some virtual machines may require excessive CPU resources, while hypervisor-level weaknesses can prevent secure migration between physical hosts.

Live VM migration, a core feature of virtualization, enables the transfer of a virtual machine from one physical server to another without interrupting its operation. This process is designed to be transparent to the guest operating system and end users. However, it also presents security risks during data transfer. Ensuring adaptability, maintaining system stability, and protecting all virtual machines during migration remain critical challenges in cloud computing environments.For many years, partially homomorphic encryption schemes have been known to offer the possibility of

performing an operation on ciphertexts without ever decryption such as adding or multiplying, for instance, the additively homomorphic. In [4], the authors proposed the multiplying homomorphic [5] cryptosystem. However, both addition and multiplication are used in the most frequently used calculations, for example in the areas of signal processing, statistics or economics and therefore partially homomorphic initiatives are not sufficient.

Examples of security contraventions happen more frequently by cloud service providers [6, 7]. Users demand that their data remain secure through the CSP and they must have a powerful guarantee from the cloud servers that the CSP stores data correctly without being disturbed or partially deleted, though because the cloud users are not aware of the internal operational details of the service providers. Therefore, an efficient and effective and secure cloud data storage system must be able to ensure data integrity and confidentiality. The confidentiality problem can be dealt with by encrypting data before cloud storage. Checking data integrity is a challenging task, however, without having a local data copy or being retrieved from the server. For this kind of reason, simple cryptographic primitives cannot be used for the protection of outsourced data directly. Furthermore, it is naive to check the integrity of data in data storage to download the stored data in order to validate its integrity, which is inefficient at exorbitant I/O costs, a high network overhead of communication and restricted computing capacity. Efficient and reliable processes are therefore needed to protect the confidentiality and integrity of user data with absolute minimum overhead computation, communication and storage. How to really secure and efficient design to incorporate these two critical data storage components seamlessly remains an open challenge in Cloud Computing.

In this article, we advocate a cost effective and secure protocol to guarantee the confidentiality and integrity of cloud computing data storage. Furthermore, the Diffie-Hellman algorithm is still computer-intensive, dramatically increasing the time complexity when generating public keys to resolve the suggested algorithm. This study focuses on existing virtual machine migration techniques and the security challenges associated with data transfer. It also examines the risks, vulnerabilities, and the need for stronger security mechanisms during VM relocation Finally the paper gives a comparative analysis on Diffie-Hellman and the proposed algorithm approach to time complexity, which we have included for hybrid cryptography.

The main contributions and organization of this paper are summarized as follows: In section II we describe literature review of searchable encryption schemes. The section III proposed work. The section IV results and discussion. Finally in section V we concluded the paper.

1.1 Related work

The searchable encryption systems have been extensively examined and can be divided into two kinds: symmetric key encryption and public key encryption.

In [18], the authors suggested a study of the author's concept. And introduced the wild card search concept. The disadvantage of this scheme is that bloom filters can reintroduce false positive effects. In [9], the authors proposed a scheme to search for multiple keywords with one encrypted query. But somehow this arrangement is not practical. The authors proposed a concept of symmetric Searchable Encryption (SSE) in [8] and later suggested an expanded version of the SSE (Dynamic SSE (DSSE)), in which documents may be added and deleted in an index table. In [25], the authors proposed the use of bucketization techniques to support SQL queries. The disadvantage is that the system does not support scalability with respect to the bucketization technique.

In [19], the authors proposed a multi-dimensional data bucketization strategy that would also support interfaces. The limitation is the same as in the scalability Hacigumus scheme. Schemes based on order preserving encryption (OPE) help range queries. The disadvantage with OPE support schemes is to show the order relationship between cipher text data. In [26] the authors proposed a CryptDB technique for database protection. CryptDB uses the proxy server as a database interface. Uses interfaces and homomorphic encryption methods such as Paillier's scheme for the search and computation of encrypted data. The major downside is that the proxy server is trusted. All these schemes are user-based [24].

The authors advocated the first public key encryption with those keyword search (PEKS) in [15]. The strategy endures from a trapdoor encryption inference attack. In [13] and [20], the authors improved the safety of the system. In [22], the author’s strategy keyword searches for the encrypted data by a single user using a unique key. The strategy is subject

to key management. In [16][17] the authors addressed functional encryption and connection searches, interval queries and sub-set queries. In the [12] system conveyed by the authors, the scheme is an updated version of [16] and discussed the predicate encryption for internal products. In [21] the authors proposed a symmetrical key-based system using predicate encryption. In [28], the authors proposed a scheme that used Hidden Vector Encryption (HVE) to disseminate search capabilities to users using several trusted agencies. The disadvantages of these strategies include key management and complex methodologies. In [11], the authors discussed the multi-keyword search system. Here the owner of the database constructs a very outrageously expensive index table that cannot be altered algorithmically. Every one of these schemes are based on semiconductors [24].

In [10], the authors proposed public user key encryption. In [14], the authors proposed a multi-user scheme based on bilinear maps. The disadvantage is the computer complexity of bilinear maps. In [27], the authors proposed a symmetrical data encryption (SDE) scheme that depends on proxy encryption. This system is efficient in performing encrypted data searches. The authors proposed keyword search re-encryption proxy. This system involves the management of keys which render this system impractical. In [24], the authors supported a multi-use settings system and a highly complex query search. This system provides SQL, such as queries for encrypted databases. Here the queries are shown with the tree data structure. All of these schemes are multi-user based [24].

1.2 Proposed framework

Virtual machine (VM) migration introduces new challenges in designing the underlying communication infrastructure, particularly in terms of addressing schemes and bandwidth allocation to support seamless VM mobility. Additionally, developing efficient VM migration strategies is complex because it requires balancing the significant benefits of migration against its associated costs, such as communication overhead, service disruption, and system management complexity.VM migration refers to the process of transferring a virtual machine from one physical host (source server) to another (destination server). In the case of live (online) migration, the VM continues to operate and respond to user requests throughout the migration process, ensuring minimal disruption and maintaining a transparent user experience. Users can continue their activities, including running applications and interacting with virtual devices, without noticing the migration. However, challenges arise in cloud environments when multiple VMs are dynamically created or migrated within a short period. This can lead to uneven load distribution, where some physical servers become overloaded while others remain underutilized. Such imbalance may result in inefficient resource utilization and degraded performance. Although load balancing strategies aim to optimize resource allocation, they must consider that different VMs have varying resource requirements. For example, some VMs are CPU-intensive and demand significant processing power, while others may require more memory or storage. Therefore, effective VM placement and migration decisions must account for these diverse workloads to prevent scenarios where high-demand VMs are assigned to underpowered hosts, leaving other resources idle.

Overall, achieving efficient VM migration requires careful consideration of resource heterogeneity, workload characteristics, and system constraints, while ensuring minimal downtime and uninterrupted service delivery.

A virtual machine is designed to continuously respond to user requests while maintaining a seamless and transparent computing environment. Users can perform normal operations such as running applications, modifying system settings, and interacting with virtual devices without interruption. However, challenges arise in cloud environments when multiple virtual machines are created or migrated within a short period. This can lead to an excessive number of VMs being assigned to certain physical servers, while others remain underutilized, resulting in load imbalance. Uneven distribution of workloads across physical servers reduces overall system efficiency. Although resource balancing strategies aim to optimize performance, they must account for the fact that virtual machines have different resource requirements. For instance, some VMs are CPU-intensive and require high processing power, while others may rely more on memory or storage resources. Therefore, effective load management must consider the diverse resource demands of virtual machines. Proper allocation ensures that CPU-intensive VMs are not placed on underpowered servers, while avoiding situations where storage or other resources remain unused. This balanced approach improves system performance and resource utilization in cloud computing environments.

The Computer Network is a group of separate computing nodes that use protocols with mutually acceptable rules and conventions to communicate with one another in a predictable way and require a permit the sharing of resources between a wide range of users. Communication has a significant impact on the business today and is expected to

communicate highly secure information. The rapid development of network technology also increases internet attacks, and traditional encryption algorithms are not enough for current Internet security information, so we propose this hybrid encryption algorithm.

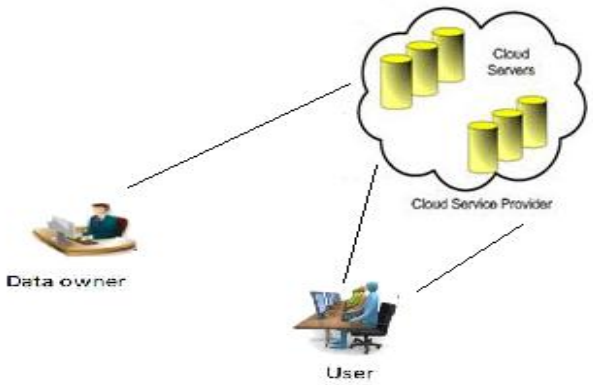


Fig. 1. Cloud data storage architecture

In our work, we consider a model of cloud data storage system, which consisting of three main entities as illustrated in Fig. 1: Data Owner, Cloud Service Provider (CSP) and Authorized Users. The typical interactions between these three entities of the system (see Fig. 1) are follows:

1) The data owner wants to outsource the set of files on the cloud server in encrypted form while still keeping the capability to search them through keyword for effective data utilization reasons.
2) When an authorized user wants to retrieve the file collection, send a search request to the CSP.
3) Then, the CSP search the files and returns set of files and hash values of files to the user.
4) Finally, the authorized user verifies the integrity and decrypts the files and gets the corresponding plaintext.

1.3 Need for Homomorphic Encryption (using RSA and Diffie -Hellman Approach) :

Homomorphic encryption is a cryptographic technique that allows specific computations to be performed directly on encrypted data. The result of these operations, when decrypted, matches the outcome as if the computations had been carried out on the original plaintext. In traditional cloud computing scenarios, data must typically be decrypted before processing, which can expose sensitive information to the service provider. Homomorphic encryption overcomes this limitation by enabling secure computation without requiring access to the original data or private keys. This ensures that data remains confidential even while being processed on remote servers. For instance, certain cryptographic systems, such as the RSA algorithm, demonstrate properties that support limited forms of such operations.

In multi-user environments, key management becomes a challenge. If n users need to communicate securely, a total of $n(n-1)/2$ unique keys would be required for pairwise communication. To address this issue, key exchange protocols like Diffie–Hellman are used, allowing two parties to securely generate a shared session key over an insecure channel without prior key distribution.

Let me know if you want this rewritten in simpler terms or expanded with examples for better understanding.

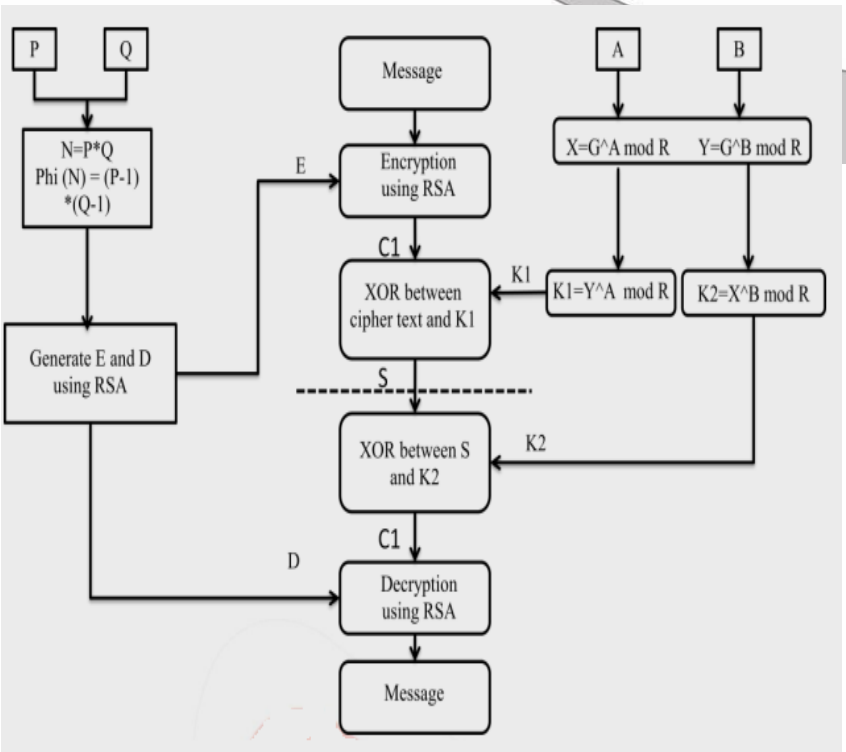


Fig.2 Proposed Hybrid framework

As shown in Fig.2, the RSA algorithm has been used to ensure message security by encryption and decryption, and the Hellman differential algorithm is used to generate the sender and receiver secret key for communicating. In this process you add another operation that is bit by bit *XOR*. For this kind of logical reason, the uniqueness of the message can be increased. This is done when the message is converted to cipher text. We first choose two primary numbers and find the key exponents for encryption and decryption that will be used for encryption and decryption in this proposed framework. We select *A* and B for the Diffie Hellman algorithm. *R* is a random primary number generated probably automatically by the system. The Diffie Hellman algorithm generates the public number. This public number allows us to generate *KA* and *KB* secret keys. These are used for carrying out *XOR* operation.

A. Diffie-Hellman Algorithm:

One of the major issues of public-key encryption is to address the problem of key distribution. The motivation for running the Diffie-Hellman protocol is to implement a secure session over an insecure network connection.

Steps involved:

- **Step1:** The two users are and . Both agree upon two positive integers, P and g where P is a prime number and g is a group generator.
- **Step2:** User randomly chooses a positive integer *X*, smaller than P and it's called as's private key. Also chooses its own private key Y.
- **Step3:** and compute public keys using
  Y = ((g) ^ X) mod P (1)
  Y = ((g) ^ X) mod P (2)
- **Step4:** They exchange public keys through a communication channel.
- **Step5:** On receiving these keys, they will compute shared key K, using
  Ka =(Y) ^x mod P (3)
  Kb = (Y) ^y mod P (4)
- **Step6:** In the last step we observe that x and y both are raised to the generator which indicates that two exponentiations are involved in this case.

Where Ka and Kb at user and user are same.
In this way secret key is shared between users.

B. RSA algorithm

Steps of RSA algorithm as follows.

1. Choose P and Q any random prime numbers.
2. Compute N=P . (5)
3. Compute ϕ (N) = (P-1) (6)

Choose e such that (1 < e < ϕ (N) and e and n are co-prime.

4. Compute a value for d such that (d ϕ
5. (N) =1
6. Public key is (e, n) Private Key is (d, n)
7. The encryption of m is
8. c=m^e mod N (7)
9. The decryption of c is
10. m=c^d mod N (8)

Our proposed hybrid algorithm shared into two parts. Paramount part is based on modified Diffie-Hellman key exchange. Succeeding part uses RSA approach to encrypt and decrypt the message but both side two keys are generated with RSA approach and keys are called as sender key for encryption of message and receive key to decrypt incoming message.

A. Secret key generation

Secret key generation is modified Diffie-Hellman algorithm.
Steps are as follows

1. User Generates Random Prime Number P P prime number generated at user
   User make P as P1=P+P (that is twice P) and send to user.
2. User Generates Random Prime Number
   User receives Number P1 and calculates P=P1/2
   Q prime number generated at user
   User make Q as Q1 =P + Q (that is add with P)
   And send to user.
3. User
   Receive number from user and subtract P from Q1
   User get Q= O1-P
   User sends this value of Q1 back to Q
4. User Authenticate and Send Public key to user receive value of Q1
   Q' = Q1-P
   Compare value with Q If Q==Q' then go ahead.
   Select Radom Private Prime Number Pb.
   If above value Q matches then User generate this public key
   PubB is sent to User
5. User Receive Public key of User and generate own public key
   Pub is received.
   Select random prime number as private key Pa
   Pub is sent to user.
6. User Receive Public key of User and Secret Key Generation Process (9)
7. User Secret Key Generation Process (10)
8. Equation 9 and 10 are same as equation 3 and 4 but generated with modified Diffie-Hellman algorithm.

B. Sender key and Receiver key generation Sender key

Key will be used for encryption. Receive key: Key will be used for decrypt received messages.

9. Calculate new P
   For user

Pnew=P …. [P is value of P from step1 in
part A and is from eq 10]
Find next prime of Pnew and which new value is of P at user P = next Prime (Pnew) (11.a)
For user
Pnew=P (secKb is from eq 9)
Find next prime of Pnew and which is new value of P at user P = nextPrime (Pnew) (11.b)

Here both equation 11.a and 11.b will be same value because secret values are same.

10. Calculate new Q For user for user

Qnew=Q …. [Q is value of Q from step1 in part A and is from eq 10]
Find next prime of Qnew and which new value is of Q at user Q = nextPrime (Qnew) (12.a)
For user
Qnew=Q ……. (secKb is from eq 9)
Find next prime of Qnew and which is new value of Q at user Q = nextPrime(Qnew) (12.b)

11. Calculate value of N and ϕ(N)
N = P * Q (13)
ϕ(N) = (P - 1) * (Q- 1) (14)
12. Choose e such that 1 < e < ϕ(N) and e and N are co-prime
13. Compute a value for d such that (d ×emod ϕ (N)
=1
14. Sender key is (e, n) Receive key is (d, n)
15. The encryption of m is (15)
16. The decryption of c is (16)

$$speedup = \frac{RSA - \text{Hybrid modified Diffie} - \text{Hellman} + \text{RSA}}{RSA} * 100$$

This is complete process of hybrid cryptosystem. In this process equation 15 and equation 16 are encryption and decryption process using sender key and receiver key respectively. Next Prime (n) is algorithms which calculate next prime number of *n.*

2. Results And Discussion

In this section, we first exhibit the numerical results of the hybrid modified Diffie-Hellman key exchange and RSA approach. All investigates conducted using C++ on system with dual core 2.7GHZ processor and 4GB RAM running Windows-7. In our implementation, we are measuring total time for computation cost of the verifier and server using hybrid modified Diffie-Hellman key exchange and RSA approach and conventional RSA respectively.

File size	Verifier side using RSA [33]	Verifier side using Our proposed
F1-10Mb	424.37 ms	336.26 ms
F2-20Mb	482.81 ms	348.26 ms
F3-30Mb	561.62 ms	358.26 ms
F4-40Mb	641.46 ms	402.26 ms
F5-50Mb	743.64 m	445.26 ms

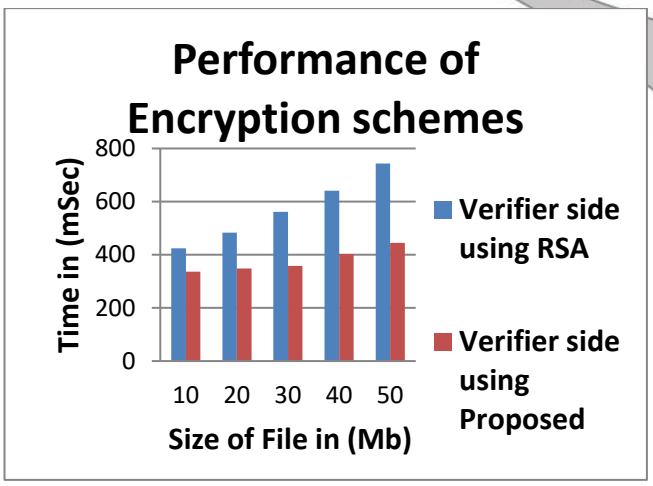


Fig 3. performance comparison between RSA and proposed system

Fig.3. shows performance by both the methods to encrypt or decrypt the data. The above comparison shows that proposed algorithm outperforms traditional RSA basic homomorphic encryption scheme for different file sizes.

3. Conclusion

In this paper, we analyze cloud computing its advantages and disadvantages. Users can save their data through the cloud. The main issue in the cloud environment is to secure the data in order to prevent an unapproved user from using or modifying the data that is stored in the cloud server. Cloud computing is a cost-effective way to improve hardware or software. We then proposed an efficient and secure approach to privacy protection based on hybrid algorithms, a modified key exchange between Diffie-Hellman and RSA, for encryption and decryption. The proposed automated system is more secure because of the secret generated message for encryption and decryption with the sender and recipient key known to send and receive. The safety of both levels is based on the hybrid encryption because it uses asymmetrical sender and recipient keys as well as a symmetrical key both user A and uses B for the encryptions and decryptions of the same key pair. We can enhance the security of messages in the proposed hybrid encryption algorithm by combining encryption and bitwise EX-OR operation, because the message complexity is also increased.

References

1. Pal AS, Pattnaik BPK. Classification of virtualization environment for cloud computing. Indian Journal of Science and Technology. 2013; 6(1):3965–71.
2. Neela TJ, Saravanan N. Privacy preserving approaches in cloud: a survey. Indian Journal of Science and Technology.2013; 6(5):4531–5.
3. Paul PK. Cloud computing based green information infrastructure: the future of ecofriendly information science practice. PIJR. 2013; 2(11):122–4.
4. P. Paillier, “Public-key cryptosystems based on composite degree residuosity classes,” in EUROCRYPT, 1999, pp. 223–238.
5. T. ElGamal, “A public key cryptosystem and a signature scheme based on discrete logarithms,” IEEE Transactions on Information Theory, vol. 31, no. 4, pp. 469–472, 1985.
6. N. Gohring, “Amazon’s S3 down for several hours,” Online at http://www.pcworld.com / business center/ article/ 142549/ amasons_down_for_sever_hours.html”, 2008.
7. J. Kincaid, “MediaMax/The Linkup Closes Its Doors”, Online at http:// www.techcrunch.com /2008/07/10/mediamax the link up-closes-its-doors/, July 2008.
8. R. Curtmola, J. A. Garay, S. Kamara, and R. Ostrovsky. Searchable symmetric encryption: improved definitions and efficient constructions, in ACM CCS, 2006, pp. 79–88.
9. P. Golle, J. Staddon, and B. Waters. Secure conjunctive keyword search over encrypted data. In M. Jakobsson, M. Yung, and J. Zhou, editors, Applied Cryptography and Network Security, Second International Conference, volume 3089 of LNCS, pages 31–45. Springer, 2004.

10. Y. H. Hwang and P. J. Lee. Public key encryption with conjunctive keyword search and its extension to a multi-user system. In T. Takagi, editor, Pairing-Based Cryptography, First International Conference, volume 4575 of LNCS, pages 2–22. Springer, 2007.
11. C. Wang, N. Cao, J. Li, K. Ren, and W. Lou. Secure ranked keyword search over encrypted cloud data, in IEEE ICDCS, 2010, pp. 253–262.
12. G. Ateniese, S. Kamara, and J. Katz. Proofs of storage from homomorphic identification protocols. In M. Matsui, editor, Advances in Cryptology — ASIACRYPT 2009, volume 5912 of LNCS, pages 319–333. Springer, 2009.
13. J. Baek, R. Safavi-Naini, and W. Susilo. Public key encryption with keyword search revisited. In O. Gervasi, editor, Proceedings of the international conference on Computational Science and Its Applications, Part I, volume 5072 of LNCS, pages 1249–1259. Springer, 2008.
14. F. Bao, R. H. Deng, X. Ding, and Y. Yang. Private query on encrypted data in multi-user settings. In L. Chen, Y. Mu, and W. Susilo, editors, Proceedings of the 4th international conference on Information security practice and experience, volume 4991 of LNCS, pages 71–85. Springer, 2008.
15. D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public Key Encryption with Keyword Search. In C. Cachin and J. Camenisch, editors, Advances in Cryptology EUROCRYPT 2004, volume 3027 of LNCS, pages 506–522. Springer, 2004.
16. D. Boneh, A. Sahai, and B.Waters. Functional encryption: definitions and challenges. In Y. Ishai, editor, Proceedings of the 8th conference on Theory of cryptography, volume 6597 of LNCS, pages 253–273. Springer, 2011.
17. D. Boneh and B.Waters. Conjunctive, subset, and range queries on encrypted data. In S. P. Vadhan, editor, Proceedings of the 4th conference on Theory of cryptography, volume 4392 of LNCS, pages 535–554. Springer, 2007.
18. C. Bosch, Q. Tang, P. Hartel, and W. Jonker. Selective document retrieval from encrypted database. In D. Gollmann and F. C. Freiling, editors, Information Security Conference - 15th Information Security Conference (ISC 2012), volume 7483 of LNCS, pages 224–241. Springer, 2012.
19. B. Hore, S. Mehrotra, M. Canim, and M. Kantarcioglu. Secure multidimensional range queries over outsourced data. The VLDB Journal, 21(3):333–358, 2012.
20. B. Hore, S. Mehrotra, M. Canim, and M. Kantarcioglu. Secure multidimensional range queries over outsourced data. The VLDB Journal, 21(3):333–358, 2012.
21. E. Shen, E. Shi, and B. Waters. Predicate privacy in encryption systems. In O. Reingold, editor, Proceedings of the 6th Theory of Cryptography Conference on Theory of Cryptography, volume 5444 of LNCS, pages 457–473. Springer, 2009.
22. G. Yang, C. Tan, Q.Huang, and D. S.Wong. Probabilistic public key encryption with equality test. In J. Pieprzyk, editor, Topics in Cryptology—CT-RSA 2010, volume 5985 of LNCS, pages 119–131. Springer, 2010.
23. Seny Kamara, Charalampos Papamanthou, Tom Roeder. Dynamic Searchable Symmetric Encryption. IACR Cryptology ePrint Archive 2013:335, 2013.
24. Muhammad Rizwan Asghar et.al. Supporting complex queries and access policies for multi-user encrypted databases. In Proceedings of the 2013 ACM workshop on cloud computing security (CCSW'13). Pages 77-88, New York, USA, 2013.
25. H. Hacigumus, B. R. Iyer, C. Li, and S. Mehrotra. Executing sql over encrypted data in the Database-service-provider model. In Proceedings of the 2002 ACM SIGMOD international conference on Management of data, SIGMOD '02, New York, USA, pp. 216–227, ACM, 2002.
26. R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan. Cryptdb: protecting confidentiality with encrypted query processing. In Proceedings of the TwentyThird ACM Symposium on Operating Systems Principles, SOSP '11, New York, USA, pp. 85–100, ACM, 2011.
27. Changyu Dong, Giovanni Russello, Naranker Dulay. Shared and Searchable Encrypted Data for Untrusted Servers. In Proceedings of 22nd IFIP WG 11.3 Working Conference on Data and Applications Security (DBSec 08), London, UK, July 2008.provider model. In Proceedings of the 2002 ACM SIGMOD international conference on Management of data, SIGMOD '02, New York, USA, pp. 216–227, ACM, 2002.
28. R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan. Cryptdb: protecting confidentiality with
29. encrypted query processing. In Proceedings of the TwentyThird ACM Symposium on Operating Systems Principles, SOSP '11, New York, USA, pp. 85–100, ACM, 2011.
30. Changyu Dong, Giovanni Russello, Naranker Dulay. Shared and Searchable Encrypted Data for Untrusted Servers. In Proceedings of 22nd IFIP WG 11.3 Working Conference on Data and Applications Security (DBSec 08), London, UK, July 2008.