Journal Homepage: - www.journalijar.com
INTERNATIONAL JOURNAL OF ADVANCED RESEARCH (IJAR)
Article DOI: 10.21474/IJAR01/2911
DOI URL: http://dx.doi.org/10.21474/IJAR01/2911

RESEARCH ARTICLE

ENHANCED SECURE CLOUD DATA PROTECTION MODEL BY INTEGRATING DUAL SYSTEM ENCRYPTION TECHNOLOGY WITH SELECTIVE PROOF TECHNIQUE.

Sadeer Dheyaa Abdulameer.
Faculty of computer science, Cihan university, sulaimaniya Kurdistan, Iraq.

Manuscript Info

Manuscript History

Received: 24 November 2016
Final Accepted: 25 December 2016
Published: January 2017

Key words:-
Cloud storage, data integrity, privacy preserving, identity-based cryptography.

Abstract

This paper, proposes a new Secure Cloud Data as an enhancement for the framework model in data security and cloud storage model by integrating the dual system encryption technology with selective proof technique. While the introduced scheme supporting any standard access structures is built in the composite structure bilinear group, it is verified adaptively CCA secure in the standard technique without threatening the expressiveness of access policy. In this paper, we attempt in addition to make an enhancement for the model to obtain more efficiency in the re-encryption key generation and re-encryption phases. Proxy Re-Encryption (PRE) is an effective cryptographic essential model that permits a data owner to nominee the access rights of the encrypted data which are stored on a cloud storage system to remaining entities without leaking the information of the data to the honest-but-curious cloud server. It implements the effectiveness for data sharing as the data owner even working with limited resource devices (e.g. mobile devices) can offload most of the computational activity to the cloud. Since its establishment many variants of PRE have been recommended and proposed. SecRBAC Based Proxy Re-Encryption (SecRBAC -ABPRE), which is observed as a regular approach for PRE, engages the PRE technology in the attribute-based encryption cryptographic framework as like that the proxy is granted to make change an encryption down an access policy to another encryption under a new access policy. CP-ABPRE is suitable to numerous real time network appliances, like sharing secure data in the network or cloud applications.

Introduction:-

Cloud computing [1], which has received considerable attention from research communities in academia as well as industry, is a distributed computation model over a large pool of shared-virtualized computing resources, such as storage, processing power, applications and services. Cloud users are provisioned and release recourses as they want in cloud computing environment. This kind of new computation model represents a new vision of providing computing services as public utilities like water and electricity. Cloud computing brings a number of benefits for cloud users. For example, (1) Users can reduce capital expenditure on hardware, software and services because they pay only for what they use; (2) Users can enjoy low management overhead and immediate access to a wide range of applications; and (3) Users can access their data wherever they have a network, rather than having to stay nearby their computers. However, there is a vast variety of barriers before cloud computing can be widely deployed.

Corresponding Author:- Sadeer Dheyaa Abdulameer
Address:- Faculty of computer science, Cihan university, sulaimaniya Kurdistan, Iraq .

A recent survey by Oracle referred the data source from international data corporation enterprise panel, showing that security represents 87% of cloud users' fears1. One of the major security concerns of cloud users is the integrity of their outsourced files since they no longer physically possess their data and thus lose the control over their data. Moreover, the cloud server is not fully trusted and it is not mandatory for the cloud server to report data loss incidents. Indeed, to ascertain cloud computing reliability, the cloud security alliance (CSA) published an analysis of cloud vulnerability incidents. The investigation [2] revealed that the incident of data Loss & Leakage accounted for 25% of all incidents, ranked second only to "Insecure Interfaces & APIs". Take Amazon's cloud crash disaster as an example2. In 2011, Amazon's huge EC2 cloud services crash permanently destroyed some data of cloud users. The data loss was apparently small relative to the total data stored, but anyone who runs a website can immediately understand how terrifying a prospect any data loss is. Sometimes it is insufficient to detect data corruption when accessing the data because it might be too late to recover the corrupted data. As a result, it is necessary for cloud users to frequently check if their outsourced data are stored properly [4].

These same ranges of possibilities imply surrender of control over physical security, sharing resources with cotenants and lacking the knowledge of where cloud resources are hosted [2]. In a nutshell, user assets become vulnerable to various security issues and challenges that are exacerbated by the lack of monitoring capabilities and decreased visibility into the security status of assets being hosted by the Cloud Service Providers (CSPs) [3, 4]. Nevertheless, users have over the years gained appreciable understanding of the need to monitor assets and the threats associated with the benefits of cloud computing, and to some extents, methods have been devised by CSPs to effectively provide users with monitoring capabilities so as to increase the adoption of cloud services. For instance, CSPs offer dashboards for tracking service availability, timely discovery of service outages and performance metrics [5]. These attempts by CSPs although sufficient for performance and functional requirements, cannot be said to be sufficiently convincing for end-users to rely on, particularly whose proclivity is the monitoring of security related metrics.

Data de duplication enables data storage systems to find and remove duplication within data without compromising its availability. The goal of data de duplication is to store more data in less space by storing and maintaining files (blocks in fine-grained de duplication manner) into a single copy, where the redundant copies of data are replaced by a reference to this copy [11]. It means that data de duplication storage system could reduce the storage size of u clients, who share the same data copy m, from O(u - jmj) to O(u + jmj) if some implementation-dependent constants are hidden [6]. Also, clients do not need to upload their data to the cloud storage server when there has been one copy stored, which will not only greatly reduce the communication cost of clients and cloud server, but also save the network bandwidth [7].

Since the data from different clients is encrypted with different secret keys, it is difficult to conduct cipher text data de duplication among clients. A secure cross-client de duplication scheme should enable a storage server to detect data de duplication over the data encrypted by different clients, and efficiently prevent the practical attacks [10], [13], [14] from poor de duplication. Douceur et al. [21] proposed the first solution for secure and efficient data de duplication, and they call it convergent encryption. This idea promoted many significant applications, where various schemes [15], [16] are implemented or designed based on convergent encryption. Recently, Bellare et al. [17] defined a new primitive, Message- Locked Encryption (MLE), which brought rigor to security de duplication, and captured various security aspects of MLE. Also, they constructed several schemes and provided some detailed analysis over them [18].

To strengthen the notions of security by considering plaintext distributions depending on the public parameter, two approaches (fully random scheme and deterministic scheme) that are secure even for lock-dependent message in realistic. It answered the question: Can message-locked encryption be secure for lock dependent message [19]. The tag randomization design makes the fully random scheme, R-MLE2 for short, satisfy the standard secure notion of data confidentiality. Also, the overhead in the length of the cipher text is only additive and independent of the message length [20].

Contribution and plan of this paper:-

The work in this paper seeks to unravel the issue of tool based cloud security monitoring by discerning the approaches and tools for enabling users attain security visibility in the cloud. It augments existing literatures in the area of cloud security by using a systematic approach that reflects on real-life requirements to help cloud users address one of the most pressing concerns associated with gaining visibility. We present a real-world case study to evaluate the ability of the tools and approaches to meet the security requirements based on published properties and functionality. We believe that this paper contributes to existing work by forming the first step to creating a framework for identifying cloud security monitoring requirements and implementing solutions to meet such requirements.

The paper is structured as follows: section two discusses related works and section three covers problem definition and the parameters behind it. Section four presents the motivations of this project. Section five presents the proposed system and tools that can be employed for cloud monitoring and section six presents the implementation of the proposed system. Section seven draws conclusions and outlines future work.

Literature survey:-

There are several existing works in the area of cloud monitoring. Alhamazani et al [6] reviewed commercial cloud monitoring tools by considering applications within the various cloud layers. Aceto et al [7] analyzed the properties that are key to monitoring cloud systems and adopted a methodology that analyses state of the art cloud monitoring techniques. Similarly, Fatema et al [8] identified essential features that are desired for operational monitoring in the cloud, reviewed and analyzed a broad range of monitoring tools that are being used to observe cloud functional resources.

Krizanic et al [33] performed a review and categorization of monitoring tools according to Operating Systems (OS), notification and other services being supported by the cloud, while Rimal et al [34] presented taxonomy of cloud services based on comparative study of different CSPs and their systems. Although, our work does not include a systematic literature review of the tools, it is a significant contribution and differs from other literatures by using a case study to reflect on how the tools can fulfill real-life monitoring requirements. It also focuses on cloud security monitoring in addition to a selection criteria of suitable tools.

Bellare et al. [27] formalized this primitive as message locked encryption, and explored its application in space efficient secure outsourced storage. An MLE scheme MLE = (P; K; E; D; T) is composed of five polynomial time algorithms. In MLE, the parameter generation algorithm P is used to generate the public parameter. The key generation algorithm K is used to generate the message-derived key. On inputting a key and a message the encryption algorithm E outputs the cipher text. The decryption algorithm D reverses the process, whose output is used to compute the cipher text/plaintext, and the tag generation algorithm T is used to generate the tag of the cipher text. In the scheme, tag generation maps the cipher text to a tag and identical plaintext result in one equal tag.

To enhance the security of de duplication and protect the data confidentiality, Bellare et al. [25] showed how to protect the data confidentiality by transforming the predictable message into an unpredictable message. In their system, a third party called key server is introduced to generate the file tag for duplication check. Li et al. [26] addressed the key management issue in block-level de duplication by distributing these keys across multiple servers after encrypting the files. Li et al. [29] considered the hybrid cloud architecture consisting of a public cloud and a private cloud and efficiently solved the problem of de duplication with differential privileges. Yuan et al. [30] proposed a de duplication system in the cloud storage to reduce the storage size of the tags for integrity check. Recently, Bellare and Keelveedhi [31] proposed a new primitive iMLE, which adopted interaction as a new ingredient to provide privacy for messages that are both correlated and dependent on the public system parameters.

Abadi et al. [28] provided stronger security guarantee for secure de duplication. The first approach was to avoid using tags that are derived deterministically from the message. They designed a fully randomized scheme that supported equality test over cipher text. More precisely, there were three components in the fully randomized scheme, namely a payload, a tag and a proof of consistency. The second approach was a deterministic scheme. It was made secure subject to the condition where the distributions were efficiently more sampler using at most q queries to the random oracle. Thus, the security of the second approach was guaranteed by limiting the computational power of the adversarial message distributions.

Problem Statement:-

Problem Definition:-

In cloud data storage, the client stores the data in cloud via cloud service provider. Once data moves to cloud he has no control over it i.e. no security for outsourced data stored in cloud, even if Cloud Service Provider (CSP) provides some standard security mechanism to protect the data from attackers but still there is a possibility threats from attackers to cloud data storage, since it is under the control of third party provider, such as data leakage, data corruption and data loss. We note that the client can verify the integrity of data stored in cloud without having a local copy of data and any knowledge of the entire data. In case clients do not have the time to verify the security of data stored in cloud, they can assign this task to trusted Third Party Auditor (TPA). The TPA verifies the integrity of data on behalf of clients using their public key.

System Architecture:-
The network representation architecture for cloud data storage, which consists four parts: those are Client, Cloud Service Provider (CSP), Third Party Auditors (TPAs) and SUBTPAS. Clients are those who have data to be stored, and accessing the data with help of Cloud Service Provider (CSP). They are typically desktop computers, laptops, mobile phones, tablet computers, *etc*. Cloud Service Provider (CSP):- Cloud Service Providers (CSPs) are those who have major resources and expertise in building, managing distributed cloud storage servers and provide applications, infrastructure, hardware, enabling technology to customers via internet as a service. Third Party Auditor (TPA):- Third Party Auditor (TPA) who has expertise and capabilities that users may not have and he verify the security of cloud data storage on behalf of users. SUBTPAS: the SUBTPAs verifies the integrity of data concurrently under the control of TPA

Security Threats:-
The cloud data storage mainly facing data corruption challenge: Data Corruption: cloud service provider or malicious cloud user or other unauthorized users are self interested to alter the user data or deleting.

There are two types of attackers are disturbing the data storage in cloud: 1) Internal Attackers: malicious cloud user, malicious third party user (either cloud provider or customer organizations) are self interested to altering the user's personal data or deleting the user data stored in cloud. Moreover they decide to hide the data loss by server hacks or Byzantine Failure to maintain its reputation. 2) External Attackers: we assume that an external attacker can compromise all storage servers, so that he can intentionally modify or delete the user's data as long as they are internally consistent.

Goals:-
In order to address the data integrity stored in cloud computing, we propose an Efficient Distribution Verification Protocol for ensuring data storage integrity to achieve the following goals: Integrity: the data stored safely in cloud and maintain all the time in cloud without any alteration. Low-Overhead: the proposed scheme verifies the security of data stored in cloud with less overhead.

Motivation:-
Unfortunately, despite being free from secret key distribution, PEKS schemes suffer from an inherent security problem regarding the keyword privacy, namely (inside) off-line Keyword Guessing Attack (KGA). Specifically, given a trapdoor, the adversarial server can choose a guessing keyword from the keyword space and then use the keyword to generate a PEKS cipher text. The server then can test whether the guessing keyword is the one underlying the trapdoor. This guessing-then-testing procedure can be repeated until the correct keyword is found. As the keyword always could leak some sensitive information of the user data, it is therefore of practical importance to overcome this security threat for secured and searchable encrypted data outsourcing.

Proposed system:-
We first provide the basic RDPC scheme only for static data integrity checking. Furthermore, we show the advanced RDPC scheme supporting fully dynamic block operations based on ORT. A. Basic RDPC Scheme We use the homomorphism hash function defined in [20] to construct our basic RDPC scheme.

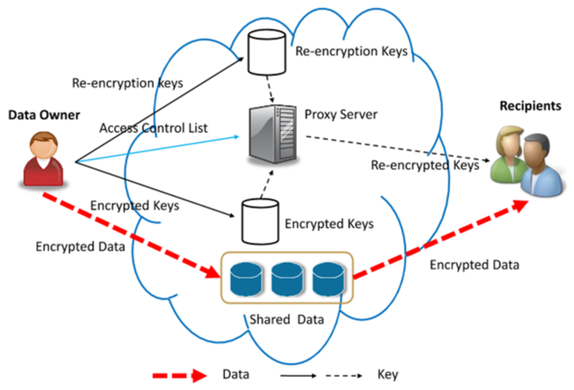

Figure 1:- Proposed system architecture

System model:-

System Components and its Security consists of six algorithms namely Setup, Extract, Tag Gen, Challenge, Proof Gen and Proof Check are involved in a CP-ABPRE system.

Setup (1k) is a probabilistic algorithm run by the KGC. It takes a security parameter k as input and outputs the system parameters PARAM and the master secret key MSK.

Extract (PARAM; MSK; ID) is a probabilistic algorithm run by the KGC. It takes the system parameters PARAM, the master secret key MSK and a user's identity ID as input, outputs the secret key SKID that corresponds to the identity ID.

Tag Gen (PARAM; F; SKID) is a probabilistic algorithm run by the data owner with identity ID. It takes the system parameters PARAM, the secret key of the user SKID and a file F 2 f0; 1g to store as input, outputs the tags = (1; SKID; n) of each file block mi, which will be stored on the cloud together with the file F.

Challenge (PARAM; FN; ID) is a randomized algorithm run by the TPA. It takes the system parameters *PARAM*, the data owner's identity ID, and a unique file name *FN* as input, outputs a challenge *CHAL* for the file named Fn on behalf of the user ID.

Proof Gen (PARAM; ID; CHAL; F) is a probabilistic algorithm run by the cloud server. It takes the system parameters *PARAM*, the challenge *CHAL*, the data owner's identity ID, the tag(a), the file F and its name Fn as input, outputs a data possession proof P of the challenged blocks.

Proof Check (PARAM; ID; CHAL; P; FN) is a deterministic algorithm run by the TPA. It takes the system parameters *CHAL*, the challenge *CHAL*, the data owner's identity ID, the file name Fn and an alleged data possession proof P as input, outputs 1 or 0 to indicate if the file F keeps intact.

Pseudo code of proposed Method:-

Step 1:-

```
Start Procedure
If User is registered then go to Step1
Else go for User registration
```

Step 2:-

```
User is going for user logs in
If User go for LDAP Authentication
If spoofed IP detected then, Block user until admin authentication.

IF user is authorized by cloud admin go to step 2
Else block user permanently by cloud admin
Else forward user and go to Step 3
Else User forward to user logs in go to step2
```

Step 3:-
Compress the user data
IF User data is sensitive then *go to* step 4
Else go for cloud storage and *go to* step 5

Step 4:-
Apply two way encryption algorithms

Step 5:-
Stored user data in cloud storage

Step 6:-
End procedure
We consider three security properties namely completeness, security against a malicious server (soundness), and privacy against the TPA (perfect data privacy) in identity-based remote data integrity checking protocols. Following the security notions due to Shacham and Waters [7], an identity-based RDIC scheme is called secure against a server if there exist no polynomial-time algorithm that can cheat the TPA with non-negligible probability and there exists a polynomial-time extractor that can recover the file by running the challenges response protocols multiple times. Completeness states that when interacting with a valid cloud server, the algorithm of Proof Check will accept the proof. Soundness says that a cheating identifier who can convince the TPA it is storing the data file is actually storing that file. We now formalize the security model of soundness for identity-based remote data integrity checking below, where an adversary who plays the role of the un trusted server and a challenger who represents a data owner are involved.

Implementation:-

The proposed system is divided into three major modules and described as below.
1. Key distribution
2. Verification process
3. Validating integrity

Key Distribution:-
In key distribution, the TPA generates the random key and distributes it to his CP-ABPRE as follows: The TPA first generates the Random key by using SOBOL Random Function. After that TPA chooses CP-ABPRE and distributes *n* pieces to them. The procedure of key distribution is given in algorithm 1.

Algorithm 1: Key Distribution:-
```
1. Generates a random key K using SOBOL Sequence
$K= f* I* k$
2. Then, the TPA partition the K into n pieces using (m, n) secret sharing scheme
3. TPA select the Number of SUBTPA$_s$: n, and threshold value m;
4. for I=1 to n do
5. TPA sends K$_I$ to the all SUBTPA$_I$ s
6. end for
7. end
```

Verification Process:-
In verification process, all SUBTPAs verify the Integrity of data and give results to the TPA; if *m* SUBTPAs responses meet the threshold value then TPA says that Integrity of data is valid. At a high level, the protocol operates like this: A TPA assigns a local timestamp to every SUBTPA of its operations. Then, every SUBTPA maintains a timestamp vector T in its trusted memory. At SUBTPA$_I$, entry T[j] is equal to the timestamp of the most recently executed operation by SUBTPA$_I$ in some view of SUBTPA$_I$. To verify the Integrity of data, each SUBTPA creates a challenge and sends to the CSP as follows: first SUBTPA generates set of Random indices **c** of set [1, n] using SOBOL Random Permutation (SRP) with random key j (c) K j $=\pi$ (4) Where local and key(a) is a SOBOL Random Permutation (SRP), which is indexed under key: $\{0,1\}\log2(l)$ ×key–$\{0,1\}$ $\log2(l)$. Next, each SUBTPA

also chooses a fresh random key RJ, where RJ= (a) 2**f***l*. Then, creates a challenge *CHAL*= {j, RJ} is pairs of random indices and random values.

The CSP computes a response to the corresponding SUBTPA challenges and send responses back to SUBTPAs. When the SUBTPA receives the response message, first he checks the timestamp, it make sure that V-T (using vector comparison) and that V [I] = T[I]. If not, the TPA aborts the operation and halts; this means that server has violated the consistency of the service. Otherwise, the SUBTPA COMMITS the operation and check if stored metadata and response (integrity proof) is correct or not? If it is correct, then stores TRUE in its table and sends true message to TPA, otherwise store FALSE and send a false signal to the TPA for corrupted file blocks. The detailed procedure of verification processes is given in algorithm 2.

Algorithm 2: Verification Process

```
1. Procedure: Verification Process
2. Timestamp T
3. Each SUBTPA$_I$ computes
4. Compute $j(c) = \pi$
5. the Generate the SOBOL random key RJ
6. Send (CHAL=(j, RJ) as a challenge to the CSP;
7. the server computes the Proof PR$_I$ send back to the
SUBTPAs;
8. PR$_I$ = Receive(V);
9. If(V* V [I] = T[I])
10. return COMMIT then
11. if PRI equals to Stored Metadata then
12. return TRUE;
13. Send Signal, (PACKETJ, TRUEI) to the TPA
14. else
15. return FALSE;
16. Send Signal, (PACKETI, FALSEI) to the TPA;
17. end if
18. else
19. ABORT and halt the process
20. end if
21. end
```

Validating Integrity:-

To validate the Integrity of the data, the TPA will receive the report from any subset *m* out of *n* SUBTPAs and validates the Integrity. If the *m* SUBTPAs give the TRUE signal to TPA, then the TPA decides that data is not corrupted otherwise he decides that data has been corrupted. In the final step, the TPA will give an Audit result to the Client. In algorithm 3, we given the process of validating the Integrity, in which, we generalize the Integrity of the verification protocol in a distributed manner. Therefore, we can use distribution verification on scheme.

Algorithm 3: -Validating Integrity

```
1. Procedure: validation(i)
2. TPA receives the response from the m SUBTPA$_s$
3. for I=1 to m do
4. If(response==TRUE)
5. Integrity of data is valid
6. else if (response==FALSE)
7. Integrity is not valid
8. end if
9. end for
10. end
```

Experiment Results:-

To evaluate the efficiency of our scheme in experiments, we implement the scheme utilizing the GNU Multiple Precision Arithmetic (GMP) library and Pairing Based Cryptography (PBC) library. The following experiments are based on coding language C on a Linux system (more precise, 2.6.35- 22-generic version) with an Intel(R) Core(TM) 2 Duo CPU of 3.33 GHZ and 2.00 GB RAM. For the elliptic curve, we choose an MNT curve with a base field of size 159 bits, jpg=160 and jpg=80.

We mainly analyze the computation cost of PEKS generation, trapdoor generation and testing in the schemes of our scheme. The computation cost of our scheme is only slightly higher than that of the BCOP scheme in terms of PEKS generation and trapdoor generation. It is because that the computation involved in the underlying CP-ABPRE scheme is quite small. Since our solution does not introduce any additional operation in the testing phase, the corresponding computation cost remains the same as the underlying PEKS system. As for the scheme which achieves a certain level of security against off-line KGA, the computation cost is more than that of the PEKS scheme and our scheme in terms of all the operations. Particularly, it takes about 2 seconds to generate a PEKS cipher text for the scheme when the keyword number is 50, while that of the scheme of Keyword Time of Testing (s) BCOP XJWW Our Scheme.

Computation Cost of Testing in our scheme is around 0.9 second and 1 second, respectively. For the trapdoor generation, the computation is slightly higher than that of our scheme as the exponentiation in G1 is usually more expensive than the exponentiation in Z<N. To be more precise, the time of trapdoor generation for 50 keywords is about 0.12 seconds while that of our scheme is 0.08 seconds. Regarding the testing operation, the computation cost is almost twice that of our scheme. Specifically, the computation cost of testing is around 1.6 second for the scheme and 0.8 seconds for our scheme. This is because the testing requires an additional pairing computation.

Conclusion:-

This paper provides a general architecture based on CP-ABPRE and hardly any guiding principle that can be espoused by cloud retailer in categorize to store and progression the data steadily. A refuge architecture that provides security as a service model and this model provides to its multiple tenants and consumers of its tenants. The security as a service model while proffering a baseline safety measures to the provider to save from harm its own cloud infrastructure also make available of flexibility to tenants and have supplementary precautions functionalities that costume their security necessities.

The future of this work would be to test the tool on real prospective cloud consumers to check if it can effectively help with cloud provider selection.

Reference:-

1. T. Jiang, X. Chen, Q. Wu, J. Ma, W. Susilo, and W. Lou, “Towards efficient fully randomized message-locked encryption,” in Information Security and Privacy - 21st Australasian Conference, ACISP 2016, Melbourne, VIC, Australia, July 4-6, 2016, Proceedings, Part I, 2016, pp. 361–375.
2. Dropbox, “Dropbox,” https://www.dropbox.com/, your stuff, anywhere.
3. Google, “Google drive,” http://drive.google.com, all your files, ready where you are.
4. NetApp, “Netapp,” http://www.netapp.com/us/products/platform-os/ dedupe.aspx, universal Storage System.
5. C. Batten, K. Barr, A. Saraf, and S. Trepetin, “pstore: A secure peer-topeer backup system,” MIT Laboratory for Computer Science, progress report, 2001.
6. M. Storer, K. Greenan, D. Long, and E. Miller, “Secure data de duplication,” in Proc. of the 4th ACM International Workshop on Storage Security and Survivability, VA, USA, Oct. 2008, pp. 1–10.
7. L. Marques and C. Costa, “Secure de duplication on mobile devices,” in Proc. of the 2011 Workshop on Open Source and Design of Communication, Lisboa, Portugal, Jul. 2011, pp. 19–26.
8. D. X. Song, D.Wagner, and A. Perrig, “Practical techniques for searches on encrypted data,” in Proc. of IEEE Symposium on Security and Privacy, CA, USA, May 2000, pp. 44–55.
9. R. Curtmola, J. A. Garay, S. Kamara, and R. Ostrovsky, “Searchable symmetric encryption: improved definitions and efficient constructions,” in Proc. of the ACM Conference on Computer and Communications Security, VA, USA, Oct. 2006, pp. 79–88.

10. D. Cash, S. Jarecki, C. Jutla, H. Krawczyk, M. Rosu, and M. Steiner, “Highly-scalable searchable symmetric encryption with support for boolean queries,” in CRYPTO 2013, ser. Computer Science, R. Canetti and J. A. Garay, Eds. Springer, 2013, vol. 8042 of LNCS, pp. 353–373.
11. S. Kamara, C. Papamanthou, and T. Roeder, “Dynamic searchable symmetric encryption,” in Proc. of the ACM Conference on Computer and Communications Security, NC, USA, Oct. 2012, pp. 965–976.
12. S. Kamara and C. Papamanthou, “Parallel and dynamic searchable symmetric encryption,” in Proc. of Financial Cryptography, Okinawa, Japan, Apr. 2013, pp. 258–274.
13. M. Naveed, M. Prabhakaran, and C. Gunter, “Dynamic searchable encryption via blind storage,” in Proc. of IEEE Symposium on Security and Privacy, CA, USA, May 2014, pp. 639–654.
14. R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, “Order preserving encryption for numeric data,” in Proc. of ACM SIGMOD, Paris, France, Jun. 2004, pp. 563–574.
15. H. Hacigumus, B. Iyer, C. Li, and S. Mehrotra, “Executing sql over encrypted data in the database-service-provider model,” in Proc. of ACM SIGMOD, Madison, Wisconsin, Jun. 2002, pp. 216–227.
16. H. Kadhem, T. Amagasa, and H. Kitagawa, “A secure and efficient order preserving encryption scheme for relational databases,” in Proc. of the International Conference on Knowledge Management and Information Sharing, Valencia, Spain, Oct. 2010, pp. 25–35.
17. R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan, “Cryptdb: Protecting confidentiality with encrypted query processing,” in Proc. of ACM Symposium on Operating Systems Principles, Cascais, Portugal, Oct. 2011, pp. 85–100.
18. R. A. Popa, F. Li, and N. Zeldovich, “An ideal-security protocol for order-preserving encoding,” in Proc. of IEEE Symposium on Security and Privacy, CA, USA, May 2013, pp. 463–477.
19. X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou, “New algorithms for secure outsourcing of modular exponentiations,” IEEE Transactions on Parallel and Distributed Systems, vol. 25(9), pp. 2386–2396, Jul. 2014.
20. X. Chen, J. Li, J. Weng, J. Ma, and W. Lou, “Verifiable computation over large database with incremental updates,” in ESORICS 2014, ser. Computer Science. Springer-Verlag, 2014, vol. 8712 of LNCS, pp. 148–162.
21. J. Douceur, A. Adya, W. Bolosky, D. Simon, and M. Theimer, “Reclaiming space from duplicate files in a server less distributed file system,” in Proc. of IEEE International Conference on Distributed Computing Systems, Macau, China, Jun. 2002, pp. 617–624.
22. D. Harnik, B. Pinkas, and A. Shulman-Peleg, “Side channels in cloud services: De duplication in cloud storage,” in Proc. of IEEE Symposium on Security and Privacy, CA, USA, Jan. 2010, pp. 40–47.
23. M. Mulazzani, S. Schrittwieser, M. Leithner, M. Huber, and E. R. Weippl, “Dark clouds on the horizon: Using cloud storage as attack vector and online slack space,” in Proc. of USENIX Security Symposium, CA, USA, Aug. 2011, pp. 65–76.
24. J. Stanek, A. Sorniotti, E. Androulaki, and L. Kencl, “A secure data de duplication scheme for cloud storage,” in Proc. of Financial Cryptography, CA, USA, Mar. 2014, pp. 99–118.
25. M. Bellare, S. Keelveedhi, and T. Ristenpart, “Dupless: Server-aided encryption for de duplicated storage,” in Proc. of the USENIX Security Symposium, DC, USA, Aug. 2013, pp. 179–194.
26. J. Li, X. Chen, M. Li, J. Li, P. Lee, and W. Lou, “Secure de duplication with efficient and reliable convergent key management,” IEEE Transactions on Parallel and Distributed Systems, vol. 25, pp. 1615–1625, Nov. 2013.
27. M. Bellare, S. Keelveedhi, and T. Ristenpart, “Message-locked encryption and secure de duplication,” in EUROCRYPT 2013, ser. Computer Science, T. Johansson and P. Q. Nguyen, Eds. Springer, 2013, vol. 7881 of LNCS, pp. 296–312.
28. M. Abadi, D. Boneh, I. Mironov, A. Raghunathan, and G. Segev, “Message-locked encryption for lock-dependent messages,” in CRYPTO 2013, ser. Computer Science, R. Canetti and J. A. Garay, Eds. Springer, 2013, vol. 8042 of LNCS, pp. 374–391.
29. J. Li, X. Chen, M. Li, J. Li, P. P. C. Lee, and W. Lou, “A hybrid cloud approach for secure authorized de duplication,” IEEE Transactions on Parallel and Distributed Systems, vol. PP, pp. 1–12, Apr. 2014.
30. J. Yuan and S. Yu, “Secure and constant cost public cloud storage auditing with de duplication,” in Proc. of IEEE Conference on Communications and Network Security, MD, USA, Oct. 2013, pp. 145–153.
31. M. Bellare and S. Keelveedhi, “Interactive message-locked encryption and secure de duplication,” in PKC 2015, ser. Computer Science, J. Katz, Ed. Springer, 2015, vol. 9020 of LNCS, pp. 516–538.