Akheel Mohammed, 2025, 13:4
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology

An Open Access Journal

Encrypted cloud storage using a dual system encryption better architecture

[1]Akheel Mohammed , [2]Sameera Khanam, [3]Ayesha
[1,2]Dept. of CSE,DRVRKWECT, Hyderabad,
[3]Software Engineer at Sonata Software Pvt

Abstract- This research introduces a Secure Cloud Data architecture that secures cloud storage and data using dual-system encryption and selective-proofing. Traditional techniques have proven that the recommended solution, which permits any standard access structure inside a composite bilinear group, is adaptively CCA secure while preserving access policy expressiveness. This study aims to improve the model's key generation and re-encryption efficiency. Proxy Re-Encryption (PRE) allows data owners provide other companies access to encrypted cloud data without an innocent- looking cloud server prying. It streamlines data sharing by letting data owners with weaker hardware (like mobile devices) use the cloud for much of their work. Since its beginning, PRE has been proposed and supported. Sec RBAC-Based Proxy Re-Encryption (SecRBAC-ABPRE) uses PRE technology in the attribute-based encryption cryptographic architecture to allow the proxy to switch between access policies. PRE is Proxy Re-Encryption. Secure data exchange in network or cloud applications is one way CP-ABPRE may be utilised with real-time network devices.

Keywords - the cloud, data security, privacy protection, and cryptography based on an individual's identification.

I. INTRODUCTION

A distributed computational paradigm, cloud computing makes use of a wide variety of shared virtualised resources, such as storage, processing power, applications, and services, and has attracted a lot of attention from academic and business research organisations. Users of a cloud computing service may provide and release resources on demand. Similar to publicly provided utilities like water and power, this innovative computational paradigm represents a new way of thinking about the delivery of computer services. There are several benefits that customers may get from using cloud computing. Users can minimise capital expenditure on hardware, software, and services by paying only for what they use. They can also enjoy reduced management overhead and instant access to a diverse array of applications. Lastly, users are no longer limited to their computers when it comes to retrieving data; they can do it from any location with network connectivity. The broad use of cloud computing is, however, impeded by a number of issues. Oracle has polled cloud users, and 87% of them are worried about security. The survey used data from the International Data Corporation's business panel. Since customers no longer physically store their data and so give up control over it, a major security concern for cloud users is the integrity of their outsourced files. Additionally, there is no guarantee that the cloud server would reveal instances of data loss, and the cloud server is not completely dependable. The Cloud Security Alliance (CSA) published an analysis of cloud vulnerability

incidents to assess the reliability of cloud computing. According to the report [2], data loss and leakage accounted for 25% of all incidents, placing it second only to "Insecure Interfaces & APIs." Take Amazon's disastrous cloud service outage as an example.The second. A major failure of Amazon's EC2 cloud services in 2011 caused some customers to lose data permanently. Even if the amount of data lost was little in comparison to the total data saved, every website manager can tell you how terrifying data loss is. Data corruption detection during access isn't always enough since restoring corrupted data could be an impossible task. This means that cloud users need to check the security of their data on a frequent basis [4].

According to the same ranges of possibilities, there is a lack of knowledge about the whereabouts of cloud resources, co-tenants share resources, and physical security is compromised [2]. Cloud Service Providers (CSPs) host user assets, making them vulnerable to a range of security issues; moreover, there is a lack of visibility into the security state of these assets and inadequate monitoring capabilities [3, 4]. Users' awareness of the importance of asset monitoring and the hazards associated with cloud computing's benefits has grown substantially over the years. As a result, CSPs have developed strategies to increase the usage of cloud services by providing consumers with powerful monitoring tools. In order to monitor service availability, detect service interruptions quickly, and evaluate performance indicators, CSPs provide dashboards [5]. Even while these initiatives by CSPs fulfil functional and performance requirements, they fail to persuade end-users, particularly those who are inclined to monitor data linked to security, to use them.

Data deduplication is a technique that data storage systems use to find and remove duplicate data without compromising accessibility. By merging many files (blocks in a fine-grained deduplication method) into one copy and then replacing duplicate data instances with references to this copy, data deduplication aims to maximise storage capacity [11]. If certain implementation-dependent constants are hidden, the data deduplication storage system has the potential to reduce the storage capacity for u clients—all of which share the same data copy m—from O(u - jmj) to O(u + jmj) [6]. In addition, once a client has stored a copy of their data, they are not obligated to upload it to the cloud storage server. This feature helps to reduce network bandwidth consumption and communication costs between clients and the cloud server [7]. Encrypting data from several clients with different secret keys makes ciphertext data deduplication more complicated. To successfully mitigate practical risks resulting from insufficient deduplication, a strong cross-client deduplication method should enable a storage server to detect data deduplication across data encrypted by many clients. Convergent encryption, first proposed by Douceur et al. [21], is the gold standard for efficient and secure data deduplication. Multiple approaches based on convergent encryption have been either implemented or created thanks to this principle, which has allowed various major applications [15], [16]. Improved security deduplication and other aspects of Message-Locked Encryption (MLE) were presented by Bellare et al. [17] as a new primitive. Further, they came up with a plethora of plans and provided in-depth evaluations of each one [18]. To improve security principles In actual settings, even for messages that rely on locks, two methods—a completely random scheme and a deterministic scheme—that take into consideration plaintext distributions depending on the public parameter manage to maintain security. The question of whether message-locked encryption is secure for communications that rely on a lock was answered in it [19]. Tag randomisation guarantees that R-MLE2, the fully random system, follows the specified safe data confidentiality level. The amount of cost due to the ciphertext's length is cumulative and unrelated to the message's length [20].

Project Purpose and Importance

The purpose of this study is to shed light on the difficulties of cloud security monitoring tools by cataloguing the processes and resources that help users get insight into cloud security. Helping cloud users address a crucial visibility problem, this improves the existing literature on cloud security via the use of a systematic approach that takes practical demands into account. We provide a real-world example to evaluate how well the approaches and

tools meet the security requirements according to the features and functions that have been defined. By laying the groundwork for identifying cloud security monitoring requirements and offering related solutions, this study contributes to the existing body of knowledge.

This is the structure of the document: Section two discusses relevant literature, while Section 3 defines the problem and its parameters. In section four, we lay forth the rationale for this undertaking. Section five lays out the recommended framework and tools for cloud monitoring, and Section six describes how to put that plan into action. The seventh section provides the results and lays out the plans for the future.

II. LITERATURE REVIEW

Numerous studies exist in the domain of cloud monitoring. Alhamazani et al. [6] evaluated commercial cloud monitoring solutions by examining their applicability across different cloud tiers. Aceto et al. [7] examined the essential qualities for monitoring cloud systems and used a methodology that evaluates contemporary cloud monitoring approaches. Fatema et al. [8] highlighted critical aspects necessary for operational monitoring in the cloud and conducted a comprehensive study and analysis of various monitoring systems used for observing cloud functional resources.

Krizanic et al. [33] conducted a review and classification of monitoring tools based on Operating Systems (OS), notifications, and other services enabled by the cloud, while Rimal et al. [34] proposed a taxonomy of cloud services derived from a comparative analysis of various Cloud Service Providers (CSPs) and their systems. While our work does not include a thorough literature evaluation of the tools, it represents a substantial addition and distinguishes itself from existing material by using a case study to illustrate how the tools might meet practical monitoring needs. It further emphasises cloud security monitoring with criteria for selecting appropriate technologies.

Bellare et al. [27] formalised this concept as message-locked encryption and examined its applicability in space-efficient secure outsourced storage. A maximum likelihood estimation approach MLE = (P; K; E; D; T) comprises five polynomial-time algorithms. In MLE, the parameter generation method P is used to produce the public parameter. The key generation algorithm K is used to produce the message-derived key. Upon entering a key and a message, the encryption algorithm E produces the ciphertext. The decryption algorithm D reverses the process, producing output used to calculate the ciphertext/plaintext, while the tag creation method T generates the tag for the ciphertext. In the technique, tag creation associates the ciphertext with a tag, ensuring that identical plaintext produces a singular, identical tag.

To augment the security of deduplication and safeguard data confidentiality, Bellare et al. [25] demonstrated a method to secure data secrecy by converting a predictable message into an unexpected one. A third party, referred to as the key server, is included into their system to produce the file tag for duplicate verification. Li et al. [26] tackled the key management challenge in block-level deduplication by dispersing the keys among many servers subsequent to file encryption. Li et al. [29] examined the hybrid cloud architecture, which comprises both a public cloud and a private cloud, and effectively addressed the issue of deduplication with varying privileges. Yuan et al. [30] presented a deduplication technique in cloud storage to minimise the storage capacity of tags for integrity verification. Recently, Bellare and Keelveedhi [31] introduced a novel primitive, iMLE, which incorporates interaction as a new element to provide privacy for messages that are linked and reliant on public system characteristics.

Abadi et al. [28] provide enhanced security assurances for safe deduplication. The first strategy was to refrain from using tags that are deterministically formed from the message. They developed a completely randomised framework that allowed equality testing on ciphertext. Specifically, the completely randomised method had three components: a payload, a tag, and a proof of

consistency. The alternative method used a deterministic framework. It was rendered secure on the condition that the distributions were sampled more efficiently utilising at most q queries to the random oracle. Consequently, the security of the second technique was ensured by limiting the processing capacity of the adversarial message distributions.

Statement of The Problem:- Definition of The Problem:

When clients use a cloud service provider (CSP) to store their data, they essentially relinquish control over that data once it is transferred to the cloud. This can lead to security concerns for the outsourced data, even with the security measures implemented by the CSP.

Although standard security mechanisms are in place to protect data from attackers, vulnerabilities in cloud data storage still exist due to its management by third-party providers. These vulnerabilities include data leakage, corruption, and loss. Clients can verify the integrity of their data stored in the cloud without needing to maintain a local copy or have detailed knowledge of the entire dataset. If clients do not have the time or resources to check the security of their cloud-stored data, they can entrust this task to a reputable Third-Party Auditor (TPA), who will use their public key to authenticate the data's integrity on behalf of the clients.

System Architecture

The network representation architecture for cloud data storage consists of four components: Client, Cloud Service Provider (CSP), Third Party Auditors (TPAs), and SUBTPAS.

Clients: These are individuals or entities who own data requiring storage and utilize a Cloud Service Provider (CSP) for data access. Clients typically use desktop computers, laptops, mobile phones, tablet PCs, and similar devices.

Cloud Service Providers (CSPs): CSPs possess significant resources and expertise in constructing and operating distributed cloud storage servers. They offer applications, infrastructure, hardware, and enabling technologies to clients via internet-based services.

Third Party Auditors (TPA): TPAs have specialized skills and capabilities that consumers may lack and are responsible for verifying the security of cloud data storage on behalf of users.

SUBTPAs: These entities simultaneously verify the integrity of data under the supervision of the TPA.

Security Vulnerabilities: The primary challenge confronting cloud data storage is data corruption.

Data Corruption: A cloud service provider, a rogue cloud user, or other unauthorized individuals may act in their own interest to alter or delete customer data.

There are two categories of attackers compromising data storage in the cloud:

- Internal Attackers: These include hostile cloud users and malicious third-party users from either the cloud provider or client organizations. Motivated by self- interest, they may alter or delete users' personal data stored in the cloud. Additionally, they may choose to conceal data loss resulting from server breaches or Byzantine failures to preserve their brand reputation.
- External Attackers: An external attacker may breach all storage servers, thereby enabling the intentional modification or deletion of user data, provided that the data remains internally consistent.

Objectives

To ensure data integrity in cloud computing, we propose an Efficient Distribution Verification Protocol that secures data storage with minimal overhead.

What drives me: - PEKS approaches eliminate secret key sharing but still face a significant security issue with keyword privacy due to the offline Keyword Guessing Attack (KGA). A malicious server can use a PEKS ciphertext to guess and test keywords iteratively until the correct one is found. This flaw needs addressing to protect user information and maintain secure, searchable encrypted data.

Suggested Framework

At the outset, we provide the basic RDPC method just for static data integrity verification. Furthermore,

we provide the advanced RDPC method that enables ORT-based dynamic block operations. A. To build our basic RDPC system, we followed the instructions for the homomorphism hash function in [20].

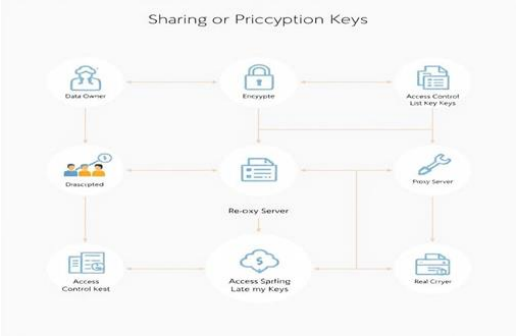

Figure 1:- Proposed system architecture

Entities and Roles in CP-ABPRE System Modelling:

- KGC: Generates the master secret key and configures system parameters. Supplies concealed keys. The data owner tags and encrypts the data with a secret key.
- Data Owner: The data owner tags and encrypts the data with a secret key before uploading it to the cloud.
- Cloud Server: Encrypted tags and data. Furnishes proof of data ownership upon request. The cloud encrypts metadata and information.
- TPA: Verifies cloud server data. Ensures the integrity and validity of the stored data. Verification of cloud servers and performing security assessments.

System Model

Configuration: Specify the security parameter k. Master Secret Key (MSK) and system specifications PARAMETER. User ID for PARAM, MSK, and key extraction. Exhibits the SK_ID user secret key.

Tag Generation: Input PARAM, SK_ID, and file F. Output tags: 1, ..., n. The system parameters PARAM, user ID, and file FN are complex.

Outcomes: CHAL challenge. Generate evidence using PARAM, user ID, challenge CHAL, file F, and file name FN. To authenticate evidence, input PARAM, user ID, challenge CHAL, proof P, and file name FN. Valid or flawed proof.

CP-ABPRE security must ensure the following:

- Only allowed individuals with secret keys are permitted to decrypt data.
- Integrity of data in the cloud.
- Authorized users are permitted to access data.
- An adversary should be incapable of substantiating a non-stored file.

Implemantaion:-

Three primary components make up the proposed system.

- Key Allocation Validation
- Verification.
- Integrity.

Important Allocation, Verification, and Integrity Implementation Issues

Validating Key Allocation: RSA or ECC can securely construct public-private key pairs. Safe random number generators give crucial randomness.

CA: Certificate Authority o Create a trustworthy CA to issue digital certificates for related companies.

Certificates need public keys, entity IDs, and validity durations.

HTTPS-secure public key distribution.

Consider a KDC for key distribution and revocation. Revocate compromised keys.

Notify entities of revoked keys using CRLs or OCSP.

Verification: Digital Signatures: ensure message integrity.

The receiver verifies messages signed privately using the sender's public key. Use SHA-256 to calculate message digests.

Compare the calculated digest against the signed message to verify data integrity. Zero-Knowledge Proofs validate assertions without disclosing sensitive information. This includes showing secret key information without exposing it.

Guaranteeing Integrity:

MACs verify communications and identify manipulation. Secret keys compute MACs, cryptochecksums.

Timestamped messages reveal creation or modification. Trust a reliable timestamp authority.

Protect sensitive data using encryption.

Use secure key management and AES encryption. Best practises and implementation problems Securely store and manage private keys.

HSMs safeguard cryptographic data.

Secure the network by preventing communication channel eavesdropping and man-in-the-middle attacks.

Encrypt and authenticate network traffic via TLS/SSL. Secure systems using the latest software and upgrades.

Limit access and authenticate users.

Audit security periodically to discover and resolve flaws.Use the NIST Cybersecurity Framework and other security best practices.

Check and repair these implementation components to protect your system, data, and connections.

Want to study zero-knowledge proofs or secure key management?

Know CP-ABPRE and Key Distribution

CP-ABPRE allows proxy servers change encrypted data across users' access restrictions without exposing anything. Sharing data with several users with different permissions needs this.

Method of Key Distribution:

TPA uses SOBOL Random Function to create K. A quasi-random number generator generates valid sequences.

The TPA divides K into n shares, K1, K2,..., Kn, using Shamir's Secret Sharing. The TPA need m shares to reproduce K.

CP-ABPRE Key Distribution: TPA securely distributes shares to n entities. Setup allocates CP-ABPRE shares.

Example: Consider a cloud storage system where users may exchange encrypted data with varying permissions. CP-ABPRE and TPA may share the master encryption key.

Set "Only users with 'admin' and 'finance' attributes can access the file."

User B asks file access.

The TPA's CP-ABPRE server re-encrypts the file using User B's access policy, "Only users with 'finance' and 'HR' attributes can access the file."

Key Points:

Shared secrets preserve the master key. CP-ABPRE servers can breach less than m shares, but the key is secured.

CP-ABPRE dynamic access control offers correct encrypted data access. Proxy re-encryption accelerates and protects data delivery.

Companies may securely transfer sensitive data and control access using CP-ABPRE and its key distribution method.

Algorithm 1: Distribution of Keys:

Understanding the Process: Key Generation and **Sharing Key Generation:**

- SOBOL Sequence: A random key K is generated using the SOBOL sequence, a quasi- random number generator. This ensures a high level of randomness and security.
- Secret Sharing: The TPA employs the (m, n) secret sharing scheme to split the key K into n shares, K1, K2, ..., Kn. This technique ensures that the original key K can only be reconstructed if at least m of these shares are combined.
- Key Distribution:
- TPA Selects Parameters: The TPA determines the number of SUBTPAs (n) and the threshold value (m).
- Key Distribution to SUBTPAs: The TPA securely distributes each share Ki to the corresponding SUBTPA i.

Security and Resilience:

- Threshold Cryptography: The (m, n) secret sharing scheme provides a robust security mechanism. Even if some SUBTPAs are compromised, the key K remains secure as long as fewer than m shares are exposed.
- Random Key Generation: The use of the SOBOL sequence ensures that the generated key K is unpredictable and resistant to attacks.

Overall, this process establishes a secure and resilient framework for key generation and distribution. It ensures that the key K is protected and can only be recovered by authorized entities.

Would you like to delve deeper into any specific aspect of this process, such as the SOBOL sequence, secret sharing, or the role of the TPA and SUBTPAs?

Verification Process:-

Verification: During this phase, each SUBTPA checks the data for integrity and reports its results to the TPA. If the total number of answers from all m SUBTPAs is more than a certain threshold, the TPA will affirm that the data is valid. Here is how the protocol works: In its operations, a TPA assigns a local timestamp to every SUBTPA. Consequently, in its encrypted memory, every SUBTPA stores a timestamp vector T. Within a certain SUBTPAI view, the item T[j] represents the timestamp of the most recent operation. To ensure the data is intact, every SUBTPA does the following and sends it to the CSP: A set of random indices c is generated from the set [1, n] by the first SUBTPA using the Sobol Random Permutation (SRP) with a random key j (c) Kj = π (4). The local variable key is indexed under key-{0,1} log2(l), and key(a) is a Sobol Random Permutation (SRP). Hence, every SUBTPA.

We also choose a new random key RJ, where RJ = (a) 2*f*l. Consequently, it becomes difficult. Pairs of arbitrary indices and values are represented by CHAL = {j, RJ}.

In response to certain SUBTPA challenges, the CSP writes a response and sends it back to the SUBTPAs. By comparing the vectors V and T and checking that V[I] = T[I], the SUBTPA first confirms the timestamp upon receiving the response message. If the server does not comply with the service's consistency, the TPA will end the process and stop. One alternative is to have SUBTPA do the action and check the stored metadata and response for correctness (integrity proof). If the information is correct, add TRUE to itstable and send a true signal to the TPA. If it is not, add FALSE and give a false signal for damaged file blocks. In the algorithm, we find the detailed method for checking Algorithm 2.

This algorithm outlines a verification process within a system likely involving multiple entities: SUBTPAs (Sub-TPAs), CSP (Cloud Service Provider), and TPA (Trusted Authority). It seems to be a security mechanism to ensure data integrity and authenticity.

Algorithm 2: Validation Procedure

- Procedure: Validation Process
- Timestamp T 3. Each SUBTPAI performs computations
- Calculate j(c) = π
- Generate the SOBOL random key RJ
- Transmit (CHAL=(j, RJ) as a challenge to the CSP;
- The server calculates the Proof PRI and transmits it back to the SUBTPAs.
- PRI = Receive(V);
- If (V * V[I] = T[I])
- thereafter return COMMIT
- If PRI is equal to Stored Metadata, then return TRUE; send signal (PACKETJ, TRUEI) to the TPA.
- otherwise
- return FALSE; 16. Transmit Signal, (PACKETI, FALSEI) to the TPA;
- terminate if 18. otherwise
- TERMINATE and stop the procedure
- terminate if
- Conclude

Here's a breakdown of the steps:

Timestamp T: A timestamp is generated, presumably to mark the current time for verification purposes.

SUBTPA Computation: Each SUBTPA calculates a value j(c) using a function π. This value is likely a hash or cryptographic digest of some data.

SOBOL Random Key Generation: A random key RJ is generated using the SOBOL sequence, a quasi-random number generator.

Challenge to CSP: The SUBTPA sends a challenge to the CSP, containing the calculated j(c) and the random key RJ.

CSP Proof Generation: The CSP computes a proof PRI based on the received challenge and sends it back to the SUBTPA.

Proof Verification: The SUBTPA receives the proof PRI and verifies it against the stored metadata.

Verification Outcome:

- If the proof is valid and matches the stored metadata, the SUBTPA sends a positive signal to the TPA.
- If the proof is invalid or doesn't match the metadata, the SUBTPA sends a negative signal to the TPA.
- If the verification fails completely, the process is aborted. Potential Use Cases:
- This algorithm could be used in various scenarios, such as:

- Blockchain Systems: To verify the authenticity of transactions and prevent fraudulent activity.
- Cloud Storage: To ensure data integrity and prevent unauthorized modifications.
- Secure Communication: To authenticate messages and detect tampering. Key Points:
- SOBOL Random Key: The use of a SOBOL sequence to generate the random key ensures a high level of randomness and security.
- Proof Generation and Verification: The CSP's ability to generate a valid proof based on the challenge demonstrates its knowledge of the underlying data and its commitment to data integrity.
- TPA Notification: The TPA is informed about the verification o Algorithm 2: Verification Process

Assurance of Integrity: -

An evaluation of the data's integrity will be carried out by the TPA after receiving the report from any subset m of n SUBTPAs. The TPA finds out whether the data is uncorrupted or corrupted based on whether the m SUBTPAs send the TRUE signal or not. The last step is for the TPA to provide an audit report to the customer. In Algorithm 3, we outline the procedure for verifying integrity by distributed generalising the verification protocol's integrity. Therefore, we may evaluate the method using distribution verification.

Algorithm 3 : Ensuring Data Accuracy

- Procedure: validation(i)
- TPA receives the response from the m SUBTPAs
- for I=1 to m do
- If(response==TRUE)
- Integrity of data is valid
- else if (response==FALSE)
- Integrity is not valid
- end if
- end for
- end

Experiment Results

to employ the GNU Multiple Precision Arithmetic (GMP) package and the Pairing Based Cryptography (PBC) module to evaluate our method in experiments. The Linux system evaluated here is version 2.6.35-22-generic, and it runs the C programming language with 2.00 GB of RAM and an Intel(R) Core(TM) 2 Duo CPU running at 3.33 GHz. Create the elliptic curve, we used an MNT curve with 160 and 80 jpeg values and a 159-bit base field.

Our focus is on evaluating the computational expenses related to PEKS generation, trapdoor manufacture, and testing in our schemes. In terms of PEKS generation and trapdoor building, our method's computational cost is higher than the BCOP scheme. The basic CP-ABPRE mechanism requires very minimal computing.

The computational cost is comparable to the underlying PEKS system since our solution does not introduce any additional operations during testing. The computational cost of the technique that offers a certain level of protection against offline Key-Generation Attacks is greater than that of our scheme and the PEKS scheme in all cases. Although the scheme of the Keyword Time of Testing (s) BCOP XJWW is not defined, it takes around 2 seconds to generate a PEKS ciphertext for the scheme with 50 keywords. Here is Our Strategy.

With our method, testing takes around 0.9 and 1 second of computing time, respectively. Since it is more expensive to exponentiate in G1 than in Z<N, the computing requirements for trapdoor creation are higher than our approach. In comparison, our method creates a trapdoor for fifty keywords in about 0.08 seconds, while the industry standard is around 0.12 seconds. The testing procedure's computing cost is twice as high as our approach. Our method has a computational cost of around 0.8 seconds, whereas the system's is 1.6 seconds. Additional calculation for pairing is necessary for testing.

III. CONCLUSION

Help cloud merchants reliably classify, store, and process data, this article introduces a generic architecture based on CP-ABPRE. The design offers minimum guiding principles. This architectural haven provides a security-as-a-service paradigm to its many occupants and the customers they serve. Tenants have flexibility and access to extra precautionary functions based on their own security

needs, all while the provider's cloud infrastructure is protected by basic safety precautions offered by the security-as-a-service concept.

The next step is to evaluate the tool on real people who would use the cloud to see how well it helps with choosing a cloud provider.

REFERENCE

1. 1.T. Jiang, X. Chen, Q. Wu, J. Ma, W. Susilo, and W. Lou, "Towards Efficient Fully Randomised Message-Locked Encryption," in Information Security and Privacy - 21st Australasian Conference, ACISP 2016, Melbourne, VIC, Australia, July 4-6, 2016, Proceedings, Part I, 2016,
2. pp. 361–375.
3. Dropbox, "Dropbox," https://www.dropbox.com/, your belongings, accessible anywhere.
4. Google, "Google Drive," http://drive.google.com, all your data accessible wherever you are.
5. 4. NetApp, "NetApp," http://www.netapp.com/us/products/platform-os/dedupe.aspxUniversal Storage System.
6. 5. C. Batten, K. Barr, A. Saraf, and S. Trepetin, "pstore: A secure peer-to-peer backup system," MIT Laboratory for Computer Science, progress report, 2001.
7. 6 M. Storer, K. Greenan, D. Long, and E. Miller, "Secure data deduplication," in Proceedings of the 4th ACM International Workshop on Storage Security and Survivability, Virginia, USA, October 2008, pp. 1–10.
8. Marques, L., & Costa, C. (2011). Secure deduplication on mobile devices. In Proceedings of the 2011 Workshop on Open Source and Design of Communication (pp. 19–26). Lisboa, Portugal.
9. Song, D. X., Wagner, D., & Perrig, A. (2000). Practical algorithms for searches on encrypted data. In Proceedings of the IEEE Symposium on Security and Privacy (pp. 44–55). California, USA.
10. R. Curtmola, J. A. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: improved definitions and efficient constructions," in Proceedings of the ACM Conference on Computer and Communications Security, Virginia, USA, October 2006, pp. 79–88.
11. 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, series Computer Science, R. Canetti and J. A. Garay, Editors. Springer, 2013, Volume 8042 of LNCS, pages 353–373.
12. S. Kamara, C. Papamanthou, and T. Roeder, "Dynamic Searchable Symmetric Encryption," in Proceedings of the ACM Conference on Computer and Communications Security, North Carolina, USA, October 2012, pp. 965–976.
13. S. Kamara and C. Papamanthou, "Parallel and dynamic searchable symmetric encryption," in Proceedings of Financial Cryptography, Okinawa, Japan, April 2013, pp. 258–274.
14. M. Naveed, M. Prabhakaran, and C. Gunter, "Dynamic Searchable Encryption via Blind Storage," in Proceedings of the IEEE Symposium on Security and Privacy, California, USA, May 2014, pp. 639–654.
15. R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order Preserving Encryption for Numeric Data," in Proceedings of ACM SIGMOD, Paris, France, June 2004, pp. 563–574.
16. H. Hacigumus, B. Iyer, C. Li, and S. Mehrotra, "Executing SQL over Encrypted Data in the Database-Service-Provider Model," in Proceedings of ACM SIGMOD, Madison, Wisconsin, June 2002, pp. 216–227.
17. H. Kadhem, T. Amagasa, and H. Kitagawa, "A secure and efficient order-preserving encryption scheme for relational databases," in Proceedings of the International Conference on Knowledge Management and Information Sharing, Valencia, Spain, October 2010, pp. 25–35.
18. R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan, "CryptDB: Safeguarding confidentiality through encrypted query processing," in Proceedings of the ACM Symposium on Operating Systems Principles, Cascais, Portugal, October 2011, pp. 85–100.
19. R. A. Popa, F. Li, and N. Zeldovich, "An ideal-security protocol for order-preserving

encoding," in Proceedings of the IEEE Symposium on Security and Privacy, California, USA, May 2013, pp. 463–477.
20. Chen, X., Li, J., Ma, J., Tang, Q., & Lou, W. (2014). New algorithms for safe outsourcing of modular exponentiations. IEEE Transactions on Parallel and Distributed Systems, 25(9), 2386– 2396.
21. Chen, X., Li, J., Weng, J., Ma, J., & Lou, W. (2014). Verifiable computation across massive databases with incremental updates. In ESORICS 2014, Computer Science Series. Springer- Verlag, 2014, vol. 8712 of LNCS, pp. 148–162.
22. J. Douceur, A. Adya, W. Bolosky, D. Simon, and M. Theimer, "Reclaiming space from duplicate files in a serverless distributed file system," in Proceedings of the IEEE International Conference on Distributed Computing Systems, Macau, China, June 2002, pp. 617–624.
23. D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Side Channels in Cloud Services: Deduplication in Cloud Storage," in Proceedings of the IEEE Symposium on Security and Privacy, California, USA, January 2010, pp. 40–47.
24. M. Mulazzani, S. Schrittwieser, M. Leithner, M. Huber, and E. R. Weippl, "Dark Clouds on the Horizon: Utilising Cloud Storage as an Attack Vector and Online Slack Space," in Proceedings of the USENIX Security Symposium, California, USA, August 2011, pp. 65–76.
25. J. Stanek, A. Sorniotti, E. Androulaki, and L. Kencl, "A secure data deduplication scheme for cloud storage," in Proceedings of Financial Cryptography, CA, USA, March 2014, pp. 99–118.
26. M. Bellare, S. Keelveedhi, and T. Ristenpart, "Dupless: Server-aided encryption for deduplicated storage," in Proceedings of the USENIX Security Symposium, Washington, DC, USA, August 2013, pp. 179–194.
27. J. Li, X. Chen, M. Li, J. Li, P. Lee, and W. Lou, "Secure Deduplication with Efficient and Reliable Convergent Key Management," IEEE Transactions on Parallel and Distributed Systems, vol. 25, pp. 1615–1625, Nov. 2013.
28. Bellare, M., Keelveedhi, S., & Ristenpart, T. (2013). Message-locked encryption and safe deduplication. In T. Johansson & P. Q. Nguyen (Eds.), EUROCRYPT 2013 (Vol. 7881, pp. 296–312). Springer.
29. M. Abadi, D. Boneh, I. Mironov, A. Raghunathan, and G. Segev, "Message-locked encryption for lock-dependent messages," in CRYPTO 2013, edited by R. Canetti and J. A. Garay, Springer, 2013, vol. 8042 of LNCS, pp. 374–391.
30. J. Li, X. Chen, M. Li, J. Li, P. P. C. Lee, and W. Lou, "A hybrid cloud approach for secure authorised deduplication," IEEE Transactions on Parallel and Distributed Systems, vol. PP, pp. 1–12, April 2014.
31. J. Yuan and S. Yu, "Secure and Constant Cost Public Cloud Storage Auditing with Deduplication," in Proceedings of the IEEE Conference on Communications and Network Security, MD, USA, October 2013, pp. 145–153.
32. M. Bellare and S. Keelveedhi, "Interactive Message-Locked Encryption and Secure Deduplication," in PKC 2015, ed. J. Katz, vol. 9020 of LNCS, Springer, 2015, pp. 516–538.