Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 5, May : 2024

ENHANCING CLOUD STORAGE SECURITY THROUGH VERIFIED KEY MANAGEMENT

Mrs. R. Deepika, Assistant professor CSE, Vaagdevi College of Engineering(Autonomous),India
Konatham Sindhuja College of Engineering(Autonomous),India
Kengarala Nagendra babu College of Engineering(Autonomous),India
Kodumagulla Vishwaksena, UG Student, CSE, Vaagdevi College of Engineering(Autonomous),India
Mohammad Javed, UG Student, CSE, Vaagdevi College of Engineering(Autonomous),India

ABSTRACT

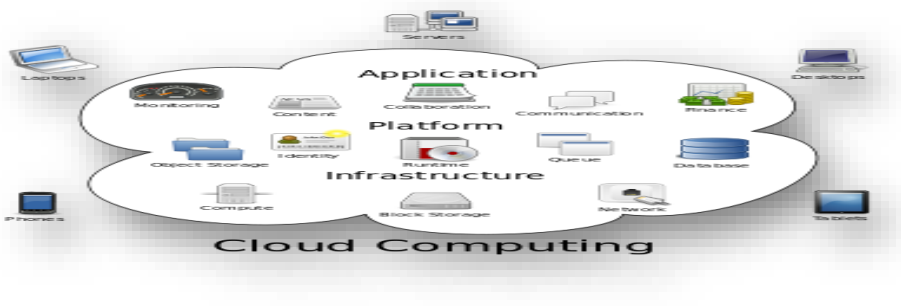
Key-exposure resistance has always been an important issue for in-depth cyber defence in many security applications. Recently, how to deal with the key exposure problem in the settings of cloud storage auditing has been proposed and studied. To address the challenge, existing solutions all require the client to update his secret keys in every time period, which may inevitably bring in new local burdens to the client, especially those with limited computation resources, such as mobile phones. In this paper, we focus on how to make the key updates as transparent as possible for the client and propose a new paradigm called cloud storage auditing with verifiable outsourcing of key updates. In this paradigm, key updates can be safely outsourced to some authorized party, and thus the key-update burden on the client will be kept minimal. In particular, we leverage the third party auditor (TPA) in many existing public auditing designs, let it play the role of authorized party in our case, and make it in charge of both the storage auditing and the secure key updates for key-exposure resistance. In our design, TPA only needs to hold an encrypted version of the client's secret key while doing all these burdensome tasks on behalf of the client. The client only needs to download the encrypted secret key from the TPA when uploading new files to cloud. Besides, our design also equips the client with capability to further verify the validity of the encrypted secret keys provided by the TPA. All these salient features are carefully designed to make the whole auditing procedure with key exposure resistance as transparent as possible for the client. We formalize the definition and the security model of this paradigm. The security proof and the performance simulation show that our detailed design instantiations are secure and efficient

1. INTRODUCTION

1.1 INTRODUCTION TO PROJECT:

What is cloud computing?

Cloud computingis the use of computing resources (hardware and software) that are delivered as a service over a network (typically the Internet). The name comes from the common use of a cloud-shaped symbol as an abstraction for the complex infrastructure it contains in system diagrams[1]. Cloud computing entrusts remote services with a user's data, software and computation. Cloud computing consists of hardware and software resources made available on the Internet as managed third-party services. These services typically provide access to advanced software applications and high-end networks of server computers[2].



Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 5, May : 2024

Structure of cloud computing

How Cloud Computing Works?

The goal of cloud computing is to apply traditional supercomputing, or high-performance computing power, normally used by military and research facilities, to perform tens of trillions of computations per second, in consumer-oriented applications such as financial portfolios, to deliver personalized information, to provide data storage or to power large, immersive computer games.

The cloud computing uses networks of large groups of servers typically running low-cost consumer PC technology[3] with specialized connections to spread data-processing chores across them. This shared IT infrastructure contains large pools of systems that are linked together. Often, virtualization techniques are used to maximize the power of cloud computing[4].

Characteristics and Services Models:

The salient characteristics of cloud computing based on the definitions provided by the National Institute of Standards and Terminology (NIST) are outlined below:

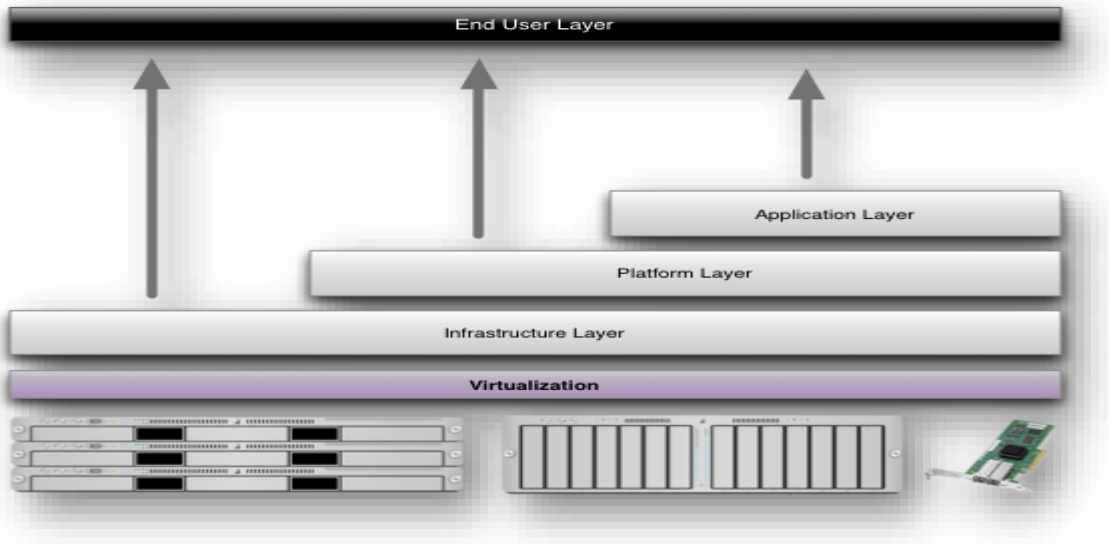
- **On-demand self-service**: A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service's provider.
- **Broad network access**: Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, laptops, and PDAs) [5].
- **Resource pooling**: The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location-independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or data center)[6]. Examples of resources include storage, processing, memory, network bandwidth, and virtual machines.
- **Rapid elasticity**: Capabilities can be rapidly and elastically provisioned, in some cases automatically, to quickly scale out and rapidly released to quickly scale in[7]. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be purchased in any quantity at any time[8].
- **Measured service**: Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be managed, controlled, and reported providing transparency for both the provider and consumer of the utilized service[9].

Characteristics of cloud computing

Services Models:

Cloud Computing comprises three different service models, namely Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS) [10]. The three service models or layer are completed by an end user layer that encapsulates the end user perspective on cloud

services[11]. The model is shown in figure below. If a cloud user accesses services on the infrastructure layer, for instance, she can run her own applications on the resources of a cloud infrastructure and remain responsible for the support, maintenance, and security of these applications herself. If she accesses a service on the application layer, these tasks are normally taken care of by the cloud service provider[12].



Structure of service models

Benefits of cloud computing:

1. **Achieve economies of scale** – increase volume output or productivity with fewer people. Your cost per unit, project or product plummets.
2. **Reduce spending on technology infrastructure.** Maintain easy access to your information with minimal upfront spending. Pay as you go (weekly, quarterly or yearly), based on demand.
3. **Globalize your workforce on the cheap.** People worldwide can access the cloud, provided they have an Internet connection.
4. **Streamline processes.** Get more work done in less time with less people.
5. **Reduce capital costs.** There's no need to spend big money on hardware, software or licensing fees.
6. **Improve accessibility.** You have access anytime, anywhere, making your life so much easier!
7. **Monitor projects more effectively.** Stay within budget and ahead of completion cycle times.
8. **Less personnel training is needed.** It takes fewer people to do more work on a cloud, with a minimal learning curve on hardware and software issues.
9. **Minimize licensing new software.** Stretch and grow without the need to buy expensive software licenses or programs.
10. **Improve flexibility.** You can change direction without serious "people" or "financial" issues at stake.

Advantages:

1. **Price:** Pay for only the resources used.
2. **Security**: Cloud instances are isolated in the network from other instances for improved security.
3. **Performance:** Instances can be added instantly for improved performance. Clients have access to the total resources of the Cloud's core hardware.
4. **Scalability:** Auto-deploy cloud instances when needed.
5. **Uptime:** Uses multiple servers for maximum redundancies. In case of server failure, instances can be automatically created on another server.
6. **Control:** Able to login from any location. Server snapshot and a software library lets you deploy custom instances.

7. **Traffic:** Deals with spike in traffic with quick deployment of additional instances to handle the load.

1.2 PURPOSE OF THE SYSTEM

Data Leakage Detection project proposes data allocation strategies that improve the probability of identifying leakages [13]. In some cases, we can also inject "realistic but fake", data records[14] to further improve our chances of detecting leakage and identifying the guilty party**.**

2. LITERATURE SURVEY

We investigate the outsourcing of numerical and scientific computations using the following framework: A customer who needs computations done but lacks the computational resources (computing power, appropriate software, or programming expertise) to do these locally, would like to use an external agent to perform these computations [15]. This currently arises in many practical situations, including the financial services and petroleum services industries. The outsourcing is secure if it is done without revealing to the external agent either the actual data or the actual answer to the computations. The general idea is for the customer to do some carefully designed local preprocessing (disguising) of the problem [16] and/or data before sending it to the agent, and also some local postprocessing of the answer returned to extract the trues answer. The disguise process should be as lightweight as possible, e.g., take time proportional to the size of the input and answer. The disguise preprocessing that that the customer performs locally to "hide" the real computation can change the numerical properties of the computational performance. We present a framework for disguising scientific computations and discuss their costs, numerical properties, and levels of security. These disguise techniques can be embedded in a very high level, easy-to-use system (problem solving environment) that hides their complexity [17].

We give protocols for the secure and private outsourcing of linear algebra computations, that enable a client to securely outsource expensive algebraic computations (like the multiplication of huge matrices) to two remote servers, such that the servers learn nothing about the customer's private input or the result of the computation, and any attempted corruption of the answer by the servers is detected with high probability. The computational work done locally by the client is linear in the size of its input and does not require the client to carry out locally any expensive encryptions of such input. The computational burden on the servers is proportional to the time complexity of the current practically used algorithms [18] for solving the algebraic problem (e.g., proportional to n^3 for multiplying two n x n matrices). If the servers were to collude against the client,then they would only find out the client's private inputs, but they would not be able to corrupt the answer without detection by the client.

Cloud computing enables customers with limited computational resources to outsource large-scale computational tasks to the cloud, where massive computational power [19] can be easily utilized in a pay-per-use manner. However, security is the major concern that prevents the wide adoption of computation outsourcing in the cloud, especially when end-user's confidential data [20] are processed and produced during the computation. Thus, secure outsourcing mechanisms are in great need to not only protect sensitive information by enabling computations with encrypted data, but also protect customers from malicious behaviors by validating the computation result. Such a mechanism of general secure computation outsourcing was recently shown to be feasible in theory, but to design mechanisms that are practically efficient remains a very challenging problem. Focusing on engineering computing and optimization tasks, this paper investigates secure outsourcing of widely applicable linear programming (LP) [21] computations. In order to achieve practical efficiency, our mechanism design explicitly decomposes the LP computation outsourcing into public LP solvers running on the cloud and private LP parameters owned by the customer. The resulting flexibility allows us to explore appropriate security/efficiency tradeoff via higher-level abstraction of LP computations than the general circuit representation. In particular, by formulating private data owned by the customer for LP problem as a set of matrices and vectors, we are able to develop a set of efficient privacy-preserving

problem transformation techniques, which allow customers to transform original LP problem into some random one while protecting sensitive input/output information. To validate the computation result, we further explore the fundamental duality theorem of LP computation and derive the necessary and sufficient conditions that correct result must satisfy. Such result verification mechanism is extremely efficient and incurs close-to-zero additional cost on both cloud server and customers. Extensive security analysis and experiment results show the immediate practicability of our mechanism design [22].

With the rapid development of cloud services, the techniques for securely outsourcing the prohibitively expensive computations to untrusted servers are getting more and more attention in the scientific community. Exponentiations modulo [23] a large prime have been considered the most expensive operations in discrete-logarithm-based cryptographic protocols, and they may be burdensome for the resource-limited devices such as RFID tags or smartcards. Therefore, it is important to present an efficient method to securely outsource such operations to (untrusted) cloud servers. In this paper, we propose a new secure outsourcing algorithm for (variable-exponent, variable-base) exponentiation modulo a prime in the two untrusted program model. Compared with the state-of-the-art algorithm, the proposed algorithm is superior in both efficiency and checkability. Based on this algorithm, we show how to achieve outsource-secure Cramer-Shoup encryptions and Schnorr signatures. We then propose the first efficient outsource-secure algorithm for simultaneous modular exponentiations. Finally, we provide the experimental evaluation that demonstrates the efficiency and effectiveness of the proposed outsourcing algorithms and schemes.

We introduce a model for *provable data possession* (PDP) [24] that allows a client that has stored data at an untrusted server to verify that the server possesses the original data without retrieving it. The model generates probabilistic proofs [25] of possession by sampling random sets of blocks from the server, which drastically reduces I/O costs. The client maintains a constant amount of metadata to verify the proof. The challenge/response protocol transmits a small, constant amount of data, which minimizes network communication. Thus, the PDP model for remote data checking supports large data sets in widely-distributed storage system. We present two provably-secure PDP schemes that are more efficient than previous solutions, even when compared with schemes that achieve weaker guarantees. In particular, the overhead at the server is low (or even constant), as opposed to linear in the size of the data. Experiments using our implementation verify the practicality of PDP and reveal that the performance of PDP is bounded by disk I/O and not by cryptographic computation.

3. PROBLEM STATEMENT

Yu *et al* [26]. constructed a cloud storage auditing protocol with key-exposure resilience by updating the user's secret keys periodically. In this way, the damage of key exposure in cloud storage auditing can be reduced. But it also brings in new local burdens for the client because the client has to execute the key update algorithm in each time period to make his secret key move forward. For some clients with limited computation resources, they might not like doing such extra computations by themselves in each time period. It would be obviously more attractive to make key updates as transparent as possible for the client, especially in frequent key update scenarios.

LIMITATION OF SYSTEM

Existing system don't like auditing protocol with verifiable outsourcing of key updates. Third party has the access to see client's secret key without encryption [27]. No verification system available for client's for to check validity of the encrypted secret keys when downloading them from the TPA. All existing auditing protocols are all built on the assumption that the secret key of the client is absolutely secure and would not be exposed. Wang *et al*. proposed a public privacy-preserving auditing protocol [28]. They used the random masking technique to make the protocol achieve privacy preserving property.

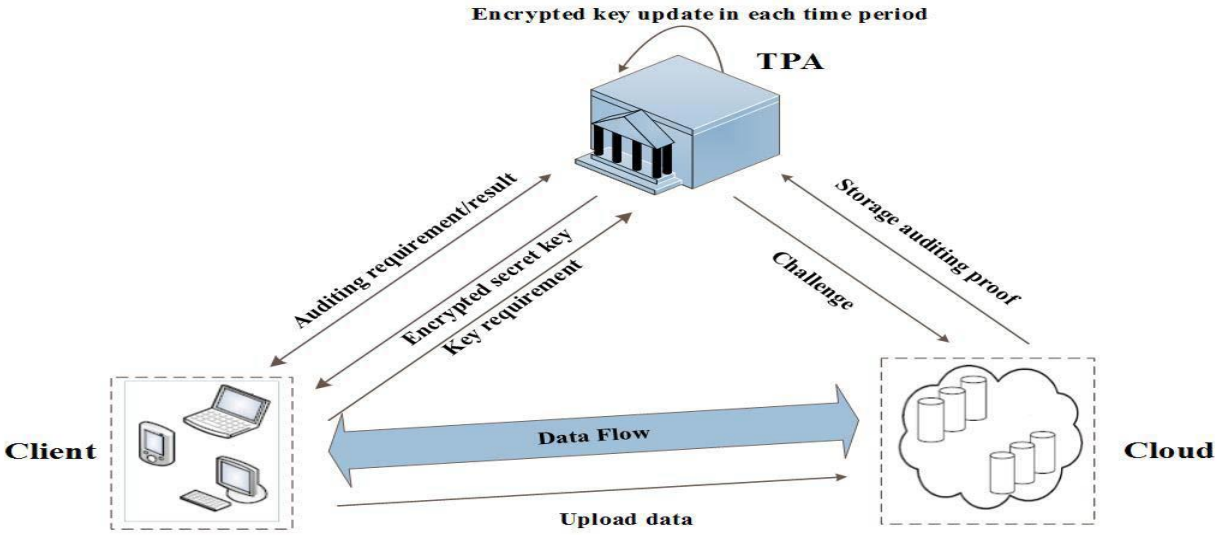
4. PROPOSED SYSTEM

The main contributions are as follows: We propose a new paradigm called cloud storage auditing with verifiable outsourcing of key updates. In this new paradigm, key-update operations are not performed by the client, but by an authorized party. The authorized party holds an encrypted secret key of the client for cloud storage auditing and updates it under the encrypted state in each time period. The client downloads the encrypted secret key from the authorized party and decrypts it only when he would like to upload new files to cloud. In addition, the client can verify the validity of the encrypted secret key. We design the first cloud storage auditing protocol with verifiable outsourcing of key updates. In our design, the third party auditor (TPA) [28] plays the role of the authorized party who is in charge of key updates. We formalize the definition and the security model of the cloud storage auditing protocol with verifiable outsourcing of key updates. We also prove the security of our protocol in the formalized security model and justify its performance by concrete implementation.

4.1 BENFITS OF PROPOSED SYSTEM

The TPA does not know the real secret key of the client for cloud storage auditing, but only holds an encrypted version. In the detailed protocol, we use the blinding technique with homomorphic property to form the encryption algorithm to encryp [29]t the secret keys held by the TPA. It makes our protocol secure and the decryption operation efficient. Meanwhile, the TPA can complete key updates under the encrypted state. The client can verify the validity of the encrypted secret key when he retrieves it from the TPA. The client downloads the encrypted secret key from the authorized party and decrypts it only when he would like to upload new files to cloud. In addition, the client can verify the validity of the encrypted secret key. Cloud storage auditing protocol [30] with verifiable outsourcing of key updates [31]. The client can verify the validity of the encrypted secret key when he retrieves it from the TPA.

5 .SYSTEM ARCHITECTURE



6. IMPLEMENTATION

6.1 Client Module

This module includes the Client registration and client login details. Every Client need to register while accessing to the cloud [32]. Every Client will be activated by the Cloud. After Cloud activated, every Client need to provide time stamp upload key to upload a new files into cloud. Time stamp upload key will be provided by third party auditor. Client need to download the time stamp upload key when client uploading new files into cloud. Client can view file details and download the file using time stamp file key provided by TPA.

6.2 Time stamp upload key:

Time stamp upload key will be provided by TPA. Client can download the upload key each time client uploading new file into cloud and they need not to give request key from TPA. At the time of client

downloading the time stamp upload key, the request will send in directly to TPA and update according to time by TPA and send encrypted upload secret key to client. And finally, client can decrypt download the upload secret key. After getting decrypt upload secret key, now Client can upload a new file into cloud.

6.3 Time stamp file key:

Each time client accessing and downloading the file from cloud, TPA will provide each time file update key to client registered mail Id. So same file key will not be there for same file. It will send as file time stamp update key, so corresponding client can use this file from different server without any other use of hacker or attacker [33]. If Client again login with same server or different server, same file key will not been used by Client to download the file for more security.

6.4 Third Party Auditor (TPA) Module

It acts as admin. TPA Provide time Upload secret key in Encrypted state for every client to upload new file into cloud. It will be send as in directly while Client downloading the upload key. The upload secret key, while user downloading key it will updated according to time. After cloud given auditing proof then only TPA can audit all files. And also provide the File Stamp key [34] for all files to the client request for corresponding files key [35].

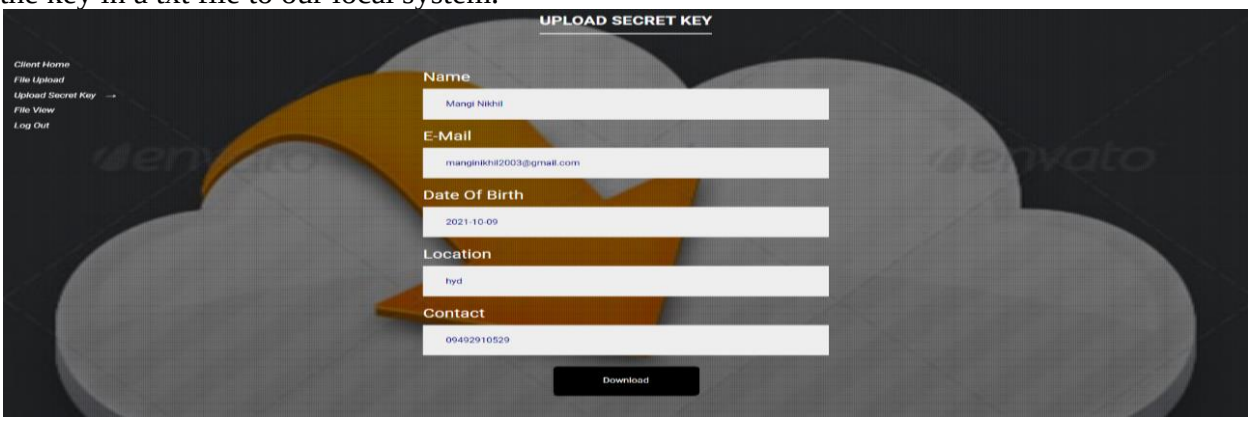
6. 5 Cloud Module

Activate data client. Cloud sends storage auditing proof [36] for all files to TPA. Cloud can view the client downloaded files from cloud.

7. EXPECTED RESULTS

Upload secret key :

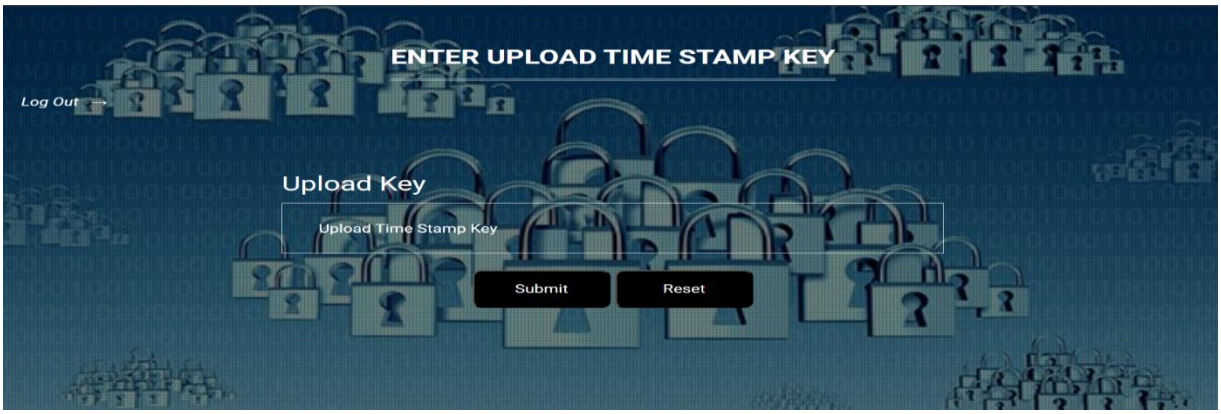
When user needs to upload user need a upload secret key to download that secret key user need to put the following asked details in the form and then it cross checks the data and then downloads the key in a txt file to our local system.



Client Side Upload Timestamp key:

When client need to upload a file to the cloud if he is a verified user by the cloud then the client will be holding a unique upload key which will be sent to client by cloud to perform the upload operation this is an extra part of verification .

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 5, May : 2024

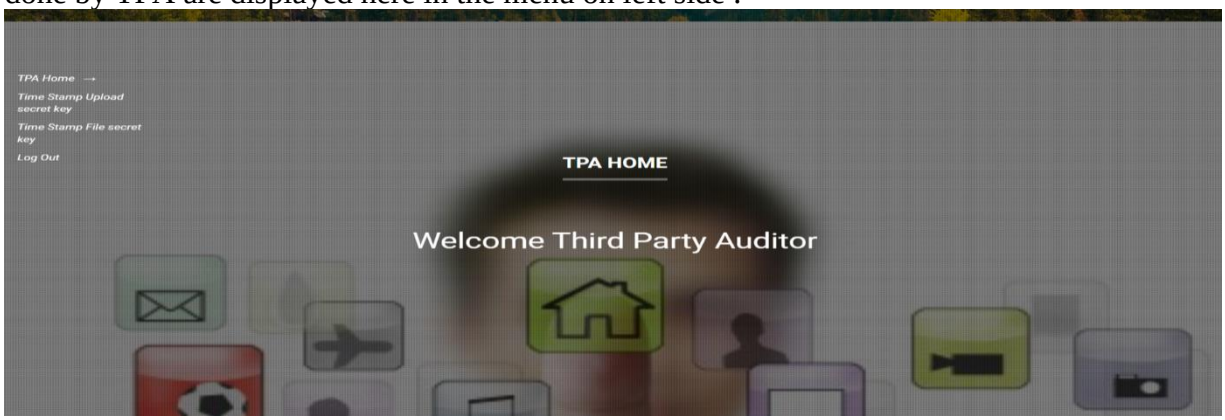


File View Client side:

This is the file view for the client. client can view all the files that have been uploaded and its details are also visible here and we can download it from here but it redirects to another page where we need to enter the secret key to download that particular file.

TPA(third party auditor) Home:

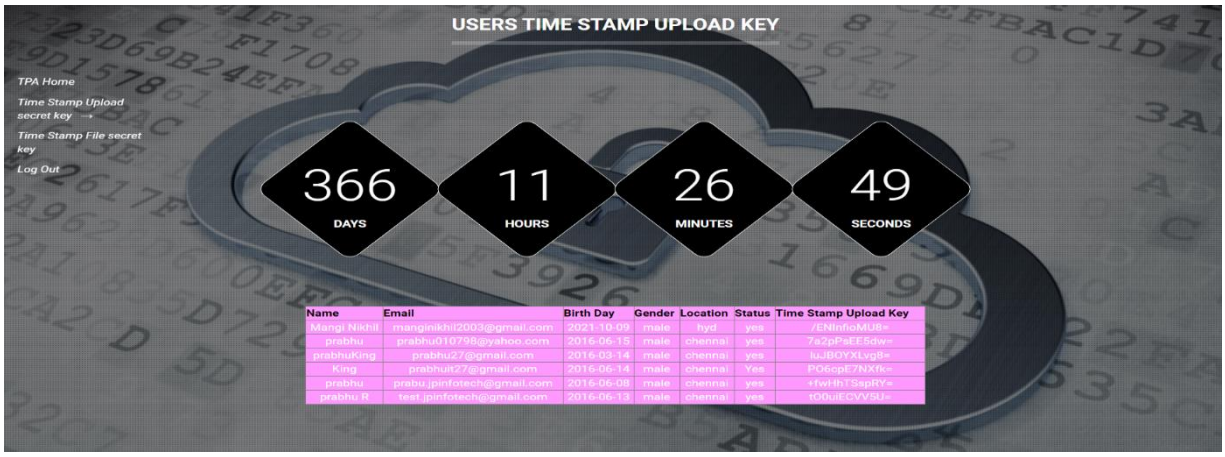
This page is for third party auditor here this is the TPA home page all the services and tasks done by TPA are displayed here in the menu on left side .



User Timestamp Upload key:

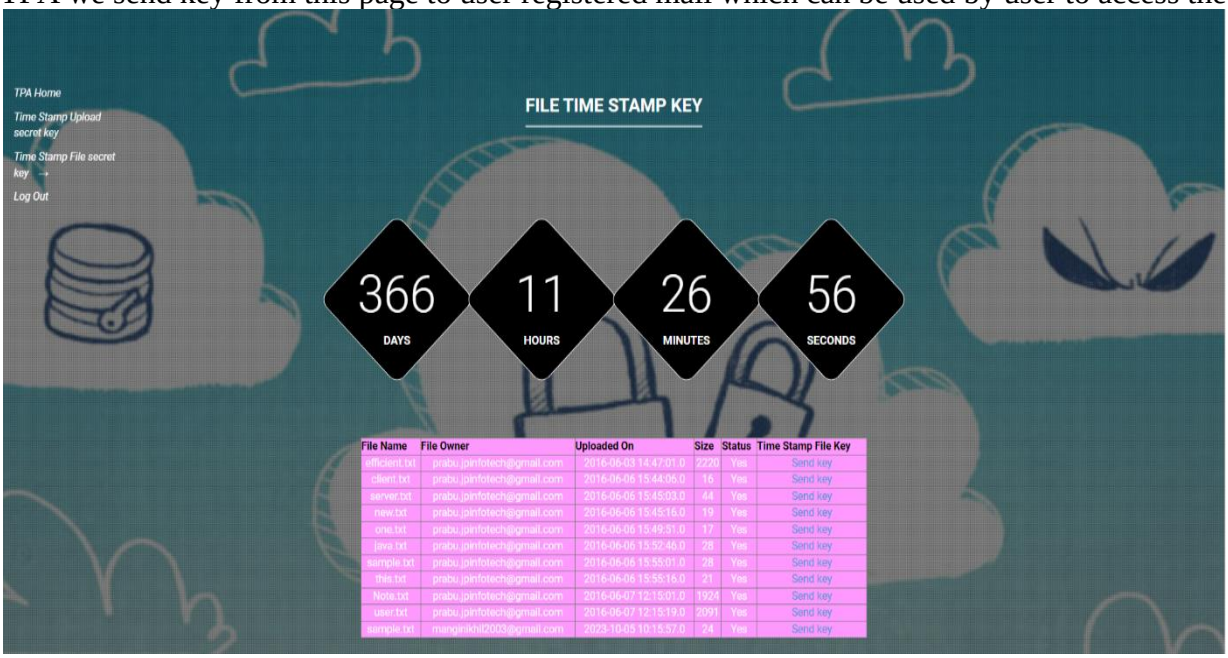
When user need to upload a file user needs users time stamp upload key this is user to verify the users identity before he can upload a file this enhances the security measures

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 5, May : 2024



Field TimeStamp Key:

In the TPA home we have a separate page for the file time stamp key [36] through this page when a user tries to access a file user need a time stamp file key which can be provided to user by TPA we send key from this page to user registered mail which can be used by user to access the key.



8. CONCLUSION

In this paper, we study on how to outsource key updates for cloud storage auditing with key-exposure resilience. We propose the first cloud storage auditing protocol with verifiable outsourcing of key updates. In this protocol, key updates are outsourced to the TPA and are transparent for the client. In addition, the TPA only sees the encrypted version of the client's secret key, while the client can further verify the validity of the encrypted secret keys when downloading them from the TPA. We give the formal security proof and the performance simulation of the proposed scheme.

9. FUTURESCOPE

In addition, the TPA only sees the encrypted version of the client's secret key, while the client can further verify the validity of the encrypted secret keys when downloading them from the TPA. We give the formal security proof and the performance simulation of the proposed scheme.

10. REFERENCES

[1] M. J. Atallah, K. N. Pantazopoulos, J. R. Rice, and E. E. Spafford, "Secure outsourcing of scientific computations," Adv. Comput., vol. 54, pp. 215–272, 2002.

[2] D. Benjamin and M. J. Atallah, "Private and cheating-free outsourcing of algebraic computations," in Proc. 6th Annu. Conf. Privacy, Secur. Trust, 2008, pp. 240–245.
[3] C. Wang, K. Ren, and J. Wang, "Secure and practical outsourcing of linear programming in cloud computing," in Proc. IEEE INFOCOM, Apr. 2011, pp. 820–828.
[4] X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou, "New algorithms for secure outsourcing of modular exponentiations," in Proc. 17th Eur. Symp. Res. Comput. Secur., 2012, pp. 541–556.
[5] G. Ateniese et al., "Provable data possession at untrusted stores," in Proc. 14th ACM Conf. Comput. Commun. Secur., 2007, pp. 598–609.
[6] A. Juels and B. S. Kaliski, Jr., "PORs: Proofs of retrievability for large files," in Proc. 14th ACM Conf. Comput. Commun. Secur., 2007, pp. 584–597.
[7] H. Shacham and B. Waters, "Compact proofs of retrievability," in Advances in Cryptology. Berlin, Germany: Springer-Verlag, 2008, pp. 90–107.
[8] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in Proc. 4th Int. Conf. Secur. Privacy Commun. Netw., 2008, Art. ID 9.
[9] F. Sebe, J. Domingo-Ferrer, A. Martinez-balleste, Y. Deswarte, and J. Quisquater, "Efficient remote data possession checking in critical information infrastructures," IEEE Trans. Knowl. Data Eng., vol. 20, no. 8, pp. 1034–1038, Aug. 2008.
[10] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: Multiplereplica provable data possession," in Proc. 28th IEEE Int. Conf. Distrib. Comput. Syst., Jun. 2008, pp. 411–420.
[11] Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu, and S. S. Yau, "Efficient provable data possession for hybrid clouds," in Proc. 17th ACM Conf. Comput. Commun. Secur., 2010, pp. 756–758.
[12] C. Wang, K. Ren, W. Lou, and J. Li, "Toward publicly auditable secure cloud data storage services," IEEE Netw., vol. 24, no. 4, pp. 19–24, Jul./Aug. 2010.
[13] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing," IEEE Trans. Parallel Distrib. Syst., vol. 22, no. 5, pp. 847–859, May 2011.
[14] K. Yang and X. Jia, "Data storage auditing service in cloud computing: Challenges, methods and opportunities," World Wide Web, vol. 15, no. 4, pp. 409–428, 2012.
[15] Y. Zhu, G.-J. Ahn, H. Hu, S. S. Yau, H. G. An, and C.-J. Hu, "Dynamic audit services for outsourced storages in clouds," IEEE Trans. Services Comput., vol. 6, no. 2, pp. 227–238, Apr./Jun. 2013.
[16] K. Yang and X. Jia, "An efficient and secure dynamic auditing protocol for data storage in cloud computing," IEEE Trans. Parallel Distrib. Syst., vol. 24, no. 9, pp. 1717–1726, Sep. 2013.
[17] H. Wang, "Proxy provable data possession in public clouds," IEEE Trans. Services Comput., vol. 6, no. 4, pp. 551–559, Oct./Dec. 2013.
[18] C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy preserving public auditing for secure cloud storage," IEEE Trans. Comput., vol. 62, no. 2, pp. 362–375, Feb. 2013.
[19] B. Wang, B. Li, and H. Li Oruta, "Oruta: Privacy-preserving public auditing for shared data in the cloud," IEEE Trans. Cloud Comput., vol. 2, no. 1, pp. 43–56, Jan./Mar. 2014.
[20] C. Erway, A. Küpçü, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in Proc. 16th ACM Conf. Comput. Commun. Secur., 2009, pp. 213–222.
[21] B. Wang, B. Li, and H. Li, "Public auditing for shared data with efficient user revocation in the cloud," in Proc. IEEE INFOCOM, Apr. 2013, pp. 2904–2912.
[22] J. Yuan and S. Yu, "Public integrity auditing for dynamic data sharing with multiuser modification," IEEE Trans. Inf. Forensics Security, vol. 10, no. 8, pp. 1717–1726, Aug. 2015.
[23] J. Yu, K. Ren, C. Wang, and V. Varadharajan, "Enabling cloud storage auditing with key-exposure resistance," IEEE Trans. Inf. Forensics Security, vol. 10, no. 6, pp. 1167–1179, Jun. 2015.
[24] D. Chaum and T. Pedersen, "Wallet databases with observers," in Advances in Cryptology. Berlin, Germany: Springer-Verlag, 1993, pp. 89–105.
[25] B. Chevallier-Mames, J.-S. Coron, N. McCullagh, D. Naccache, and M. Scott, "Secure delegation

of elliptic-curve pairing," in Proc. CARDIS, 2010, pp. 24–35.
[26] S. Hohenberger and A. Lysyanskaya, "How to securely outsource cryptographic computations," in Proc. TCC, 2005, pp. 264–282.
[27] M. J. Atallah and J. Li, "Secure outsourcing of sequence comparisons," Int. J. Inf. Secur., vol. 4, no. 4, pp. 277–287, 2005.
[28] M. J. Atallah and K. B. Frikken, "Securely outsourcing linear algebra computations," in Proc. 5th ACM Symp. Inf., Comput. Commun. Secur., 2010, pp. 48–59.
[29] X. Chen, J. Li, X. Huang, J. Li, Y. Xiang, and D. S. Wong, "Secure outsourced attribute-based signatures," IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 12, pp. 3285–3294, Dec. 2014.
[30] F. Zhang, X. Ma, and S. Liu, "Efficient computation outsourcing for inverting a class of homomorphic functions," Inf. Sci., vol. 286, pp. 19–28, Dec. 2014.
[31] M. Sookhak, A. Gania, M. K. Khanb, and R. Buyyac, "Dynamic remote data auditing for securing big data storage in cloud computing," Inf. Sci., Sep. 2105, doi: 10.1016/j.ins.2015.09.004.
[32] C. Guan, K. Ren, F. Zhang, K. Florian, and J. Yu, "Symmetric-key based proofs of retrievability supporting public verification," in Proc. 20th Eur. Symp. Res. Comput. Secur. (ESORICS), 2015, pp. 203–223.
[33] G. Yang, J. Yu, W. Shen, Q. Su, Z. Fu, and R. Hao, "Enabling public auditing for shared data in cloud storage supporting identity privacy and traceability," J. Syst. Softw., vol. 113, pp. 130–139, Mar. 2016.
[34] J. Yu, F. Kong, X. Cheng, R. Hao, and G. Li, "One forward-secure signature scheme using bilinear maps and its applications," Inf. Sci., vol. 279, pp. 60–76, Sep. 2014.
[35] J. Yu, R. Hao, H. Zhao, M. Shu, and J. Fan, "IRIBE: Intrusionresilient identity-based encryption," Inf. Sci., vol. 329, pp. 90–104, Feb. 2016.
[36] B. Lynn. (2015). The Pairing-Based Cryptographic Library. [Online]. Available: http://crypto.Stanford.edu/pbc