A Secure and Verifiable Access ControlScheme for Big Data Storage in Clouds

V. Satish Kumar[1], C. Preethi[2]

[1]Assistant Professor, Dept. of CSE, Dr.K.V.Subba Reddy College of Engineering for Women, Kurnool, A.P
[2]M.Tech Student, Dept. of CSE, Dr.K.V.Subba Reddy College of Engineering for Women, Kurnool, A.P

Abstract: We propose a secure and verifiable access control scheme based on the NTRU cryptosystem for big data storage in clouds. We first propose a new NTRU decryption algorithm to overcome the decryption failures of the original NTRU, and then detail our scheme and analyze its correctness, security strengths, and computational efficiency. Our scheme allows the cloud server to efficiently update the cipher text when a new access policy is specified by the data owner, who is also able to validate the update to counter against cheating behaviors of the cloud. It also enables **(i)** the data owner and eligible users to effectively verify the legitimacy of a user for accessing the data, and (ii) a user to validate the information provided by other users for correct plaintext recovery. Rigorous analysis indicates that our scheme can prevent eligible users from cheating and resist various attacks such as the collusion attack.

Keywords: NTRU Cryptosystem, computational efficiency, data owner, eligible users, collusion attack.

1. Introduction

BIG data is a high volume, and/or high velocity, high variety information asset, which requires new forms of processing to enable enhanced decision making, insight discovery, and process optimization [1]. Due to its complexity and large volume, managing big data using on hand database management tools is difficult.

An effective solution is to outsource the data to a cloud server that has the capabilities of storing big data and processing users' access requests in an efficient manner. For example, in ehealth applications, the genome information should be securely stored in an e-health cloud as a single sequenced human genome is around 140 gigabytes in size [2], [3]. However, when a data owner outsources its data to a cloud, sensitive information may be disclosed because the cloud server is not trusted; therefore, typically the cipher text of the data is stored in the cloud. But how to update the cipher text stored in a cloud when a new access policy is designated by the data owner and how to verify the legitimacy of a user who intends to access the data are still of great concerns.

Most existing approaches for securing the outsourced big data in clouds are based on either attributed-based encryption (ABE) or secret sharing. ABE based approaches [4]–[11] provide the flexibility for a data owner to predefine the set of users who are eligible for accessing the data but they suffer from the high complexity of efficiently updating the access policy and cipher text. Secret sharing [11]–[17] mechanisms allow a secret to be shared and reconstructed by certain number of cooperative users but they typically employ asymmetric public key cryptography such as RSA for users' legitimacy verification, which incur high computational overhead. Moreover, it is also a challenging issue to dynamically and efficiently update the access policies according to the new requirements of the data owners in secret sharing approaches.

As a data owner typically does not backup its data locally after outsourcing the data to a cloud, it cannot easily manage the data stored in the cloud. Besides, as more and more companies and organizations are using clouds to store their data, it becomes more challenging and critical to deal with the issue of access policy update for enhancing security and dealing with the dynamism caused by the users' join and leave activities. To the best of our knowledge, policy update for outsourced big data storage in clouds has never been considered by the existing research [13], [17], [18].

The considerations mentioned above motivate us to develop a verifiable access control scheme for securing the big data stored in clouds, tackling the challenges of the following security services:

- *Security.* The proposed scheme should be able to defend against various attacks such as the collusion attack. Meanwhile, access policy update should not break the security of the data storage, disclose sensitive information about the data owner, and cause any new security problem.
- *Verification.* When a user needs to decrypt a stored cipher text, its access legitimacy should be verified by other participating users and the secret shares obtained from other users must be validated for correct recovery.
- *Authorization.* To reduce the risk of information leakage, a user should obtain authorization from the data owner for accessing the encrypted data.

A. Issues in existing system

In this information era, companies and organizations are facing a challenging problem of effectively managing their complex data. As the development of cloud storage, outsourcing the data to a cloud is an appropriate approach. Generally speaking, clouds can be classified into two major categories. Public clouds with each being a multi-tenant environment shared with a number of other tenants, and private clouds with each being a single-tenant environment dedicated to a single tenant. For example, the IBM cloud was proposed as a public one for the data management of banking. When the bank stores its data in the could server as shown in Fig. 1, [3].

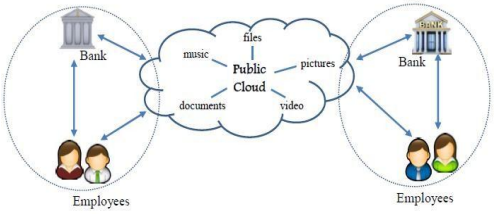


Fig. 1. An example application of big data storage in banking

Big data frequently contains a huge amount of personal identifiable information, how to securely store the data and how to provide access control over the stored data are two biggest challenges. In this subsection, we mainly summarize the state of-the-art of securing the big data stored in clouds.

In outsourcing, the secure mechanism should be established between a data owner and a cloud. The cloud to perform operations over the encrypted data, "Fully Homomorphic Encryption" (FHE) was usually adopted, which allows the direct addition and multiplication operations over the ciphertexts while preserving decryptability. Nevertheless, it is an immature cryptosystem, and is extremely inefficient in practice, which renders it hardly applicable in real world applications. Securely outsourcing big data computations to the clouds was also extensively studied.

Delegation is a popular approach for policy update. In [7], a user generates a new private key using its previous private key, and then delegates the new private key to a local authority for access policy update. A procedure called "cipher text delegation" was designed for the third party to 're-encrypt' the cipher text to a more restrictive policy using only public information.

2. Proposed system

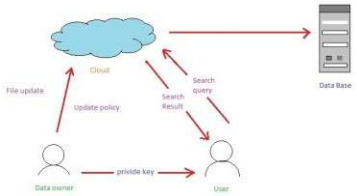


Fig. 2. System model

We consider a cloud storage system that is applicable for both public and private clouds as shown in Fig. 2. It consists of the following three types of entities: cloud server, data owner (owners), and data user (users).

A. Owner Module

In this module, owner first register their basic information that are username, password, mobile no, emailed and address etc. and according to the registration they login to the site through valid user name and password. In that owner site page, he/she will upload the different types of files that are video, audio, document and image file, etc. Each file has multiple access policy details that are separated by commas. Owner can view all download file details from server.

B. Access Policy Module

In this access policy module, if the owner may want to view or change their file access policy that also possible. First of all, they select their file id and it will display the file name and old access policy and we can enter new access policy that will be updated in the server.

C. Encryption & Decryption Module

Depend on owner name and access policy, file key will automatically have generated. Based on the file key, file content will be encrypted and stored into cloud server. If the user may want to download the file, he/she will provide valid file key and then encrypted content will be decrypted to the user at lastthey download it.

D. Key Distribute Module

Owner will view all requested file details and they decide to distribute that requested file key to corresponding requested user email id. According to the file key, user will decrypt and download the file from the server.

E. Cloud Module

In this cloud module, cloud admin can view owner file access policy and if the owner changes their file access policy that privilege is accepted by cloud admin. Based the updated file access policy, cloud admin can regenerate the file key and re- encrypt their file content again it will be stored in the server. This file key is send to the

corresponding updated file owner email-id. If any user request file from server. That request details are first accepted by cloud admin and then it will be accepted by file owner. Cloud admin can view owner and user details and who update user authorized status. Another process of cloud admin is user private key is generated by admin and it will automatically send to corresponding user email-id. Finally, who may view upload and download file details.

F. User Module

In this module, user first register their basic information that are username, password, mobile no, emailed and address etc. and according to the registration they login to the site through valid user name and password. In that user site page, if the user searches the file first of all it checks private key and file access policy all are correct, it will display corresponding file details and user may request the file that details are send to the cloud admin.

G. File Request Module

If the user needs the file from the cloud server, he/she will request to the cloud admin. Then cloud admin will send that details to the user. If both members are accepted, then file key will send to the user email-id.

H. Download Module

In this module, if the user may want to download the file from server, site will check their private key and request file details including given file id all are valid then file concept will be decrypt and send the user site page. Additionally, user can view all download file details in their site page.

3. System architecture

A system architecture or systems architecture is the computational design that defines the structure and/or behavior of a system as shown in Fig 3.

An architecture description is a formal description of a system, organized in a way that supports reasoning about the structural properties of the system. It defines the system components or building blocks and provides a plan from which products can be procured, and systems developed, that will work together to implement the overall system.

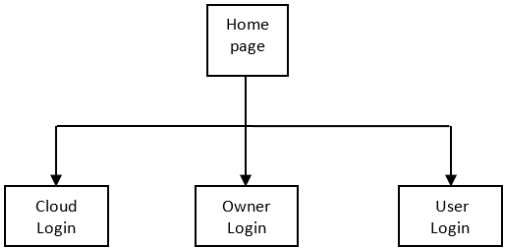


Fig 3. System architecture

4. Improved NTRU cryptosystem

In other words, the decryption process of NTRU cannot always output a correct decrypted message. To overcome this problem, we propose an improved NTRU cryptosystem in this section. To proceed, we first analyze the reasons of the decryption failures in the original NTRU. As analyzed in [48], the NTRU decryption could correctly recover the plaintext. In order to overcome the Wrap failure, we propose the following improved NTRU decryption algorithm [5], [6].

A. Algorithm 1 The Improved NTRU Decryption

The improved NTRU cryptosystem consists of the original encryption and the proposed improved decryption algorithm. Thus, proving this theorem is equivalent to showing that our improved decryption cannot reduce the security strength of the original NTRU. This can be analyzed from the following two aspects. First, similar to the original NTRU, our improved NTRU is also based on the shortest vector problem (SVP) in a lattice. Second, the improved decryption gets rid of the Gap failure and the Wrap failure to correctly recover the original message m without revealing any sensitive information as the decryptor computes the adjusting vectors and keeps them to itself, which implies that the improved decryption procedure is as secure as the original NTRU decryption. Therefore, we claim that the improved NTRU cryptosystem does not reduce the security strength of the original NTRU.

Input: cipher text e, secret key $\{f, f_p\}$.
Output: plaintext m;
The decryptor computes $a = e * f$;
$\Gamma = \max\{|\max_{0\le i\le N-1}\{a_i\}|, |\min_{0\le i\le N-1}\{a_i\}|\}$;
$\tau = \lfloor \frac{\Gamma}{q/2} \rfloor$;
If $\tau = 0$
  $m = a * f_p \pmod p$.
Else
  For $0 \le i \le N-1$,
    Compute $\gamma = \lfloor \frac{|a_i|}{q/2} \rfloor$;
    If $\gamma = 0$
      $a'_i = a_i$ and $c_i^{(1)} = c_i^{(2)} = \cdots = c_i^{(\tau)} = 0$;
    Else If $a_i \ge 0$
      $a'_i = a_i - \frac{q-1}{2}\gamma$;
      $c_i^{(1)} = c_i^{(2)} = \cdots = c_i^{(\gamma)} = \frac{q-1}{2}$;
      $c_i^{(\gamma+1)} = a'_i$;
      $c_i^{(\gamma+2)} = \cdots = c_i^{(\tau)} = 0$;
    Else
      $a'_i = a_i + \frac{q-1}{2}\gamma$;
      $c_i^{(1)} = c_i^{(2)} = \cdots = c_i^{(\gamma)} = -\frac{q-1}{2}$;
      $c_i^{(\gamma+1)} = a'_i$;
      $c_i^{(\gamma+2)} = \cdots = c_i^{(\tau)} = 0$;
    EndIf
  EndFor
  $m' = a' * f_p + c^{(1)} * f_p + \cdots + c^{(\tau)} * f_p \pmod p$;
EndIf
Output plaintext m'.

5. System implementation

Implementation is the stage in the project where the theoretical design is turned into a working system. The most critical stage is achieving a successful system and in giving confidence on the new system for the users, what it

will work efficient and effectively. It involves careful planning, investing of the current system, and its constraints on implementation, design of methods to achieve the changeover methods.

The implementation process begins with preparing a plan for the implementation of the system. According to this plan, the activities are to be carried out in these plans; discussion has been made regarding the equipment, resources and how to test activities.

The coding step translates a detail design representation into a programming language realization. Programming languages are vehicles for communication between human and computers programming language characteristics and coding style can profoundly affect software quality and maintainability. The coding is done with the following characteristics in mind.

- Ease of design to code translation.
- Code efficiency.
- Memory efficiency.

- Maintainability.

The user should be very careful while implementing a project to ensure what they have planned is properly implemented. The user should not change the purpose of project while implementing. The user should not go in a roundabout way to achieve a solution; it should be direct, crisp and clear and up to the point.

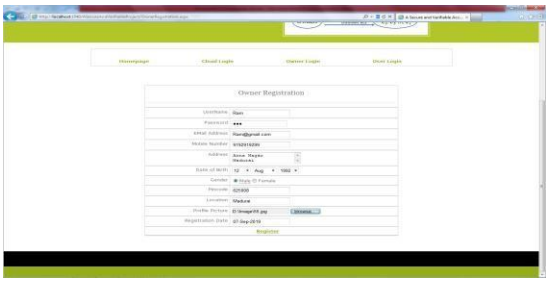
Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective.

The implementation stage involves careful planning, investigation of the existing system and it's constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

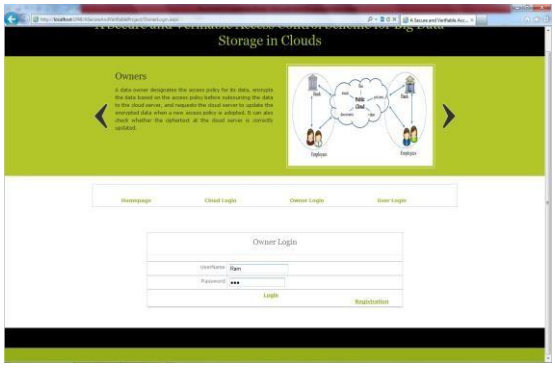
6. Result in proposed system

A. Home Page

B. New Owner Registration



C. Owner Login

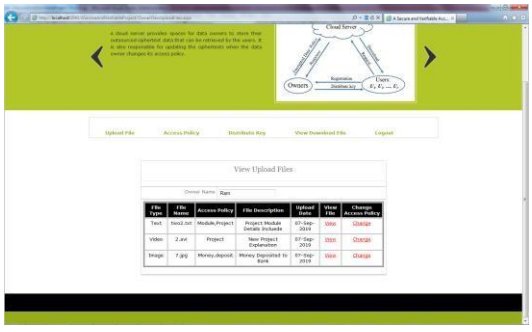


The data owner publish the his/her date in to the cloud. The cloud save file in encrypted format. And after they change the new access policy the file was re-encrypted.

D. Upload New File

The owner uploads the file or date into the cloud with file id, Accesses policy type, etc.

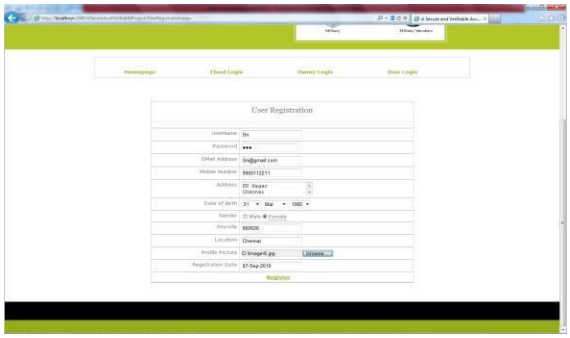
E. Owner View Upload File details



The owner can view the uploaded files.

F. Change Access Policy

G. User Registration



If data owner updates the access policy using old access policy and file id. And finally they use new access policy. When they change policy type in cloud side the updated file is re- encrypted.

A. User Login

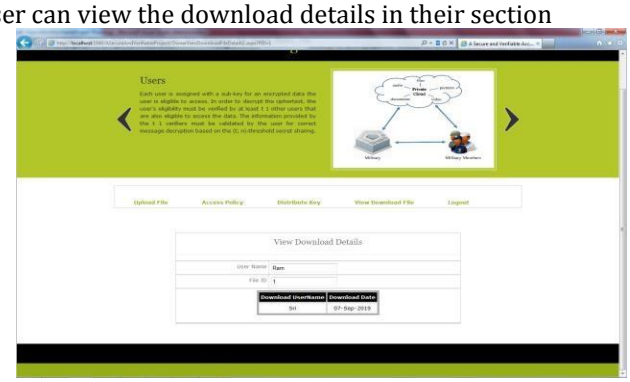
Before user login the user registers the details. The cloud verify the user then cloud provide they private key to the user. Using this key, the user can do the work.

The user searching the file. The user requests the file. Requested details are send the cloud. Then data owner distributes the key to user mail id. Using the key user download the file.

B. User View Download Details

The user downloads the file using date owner distribute key.

User can view the download details in their section



7. Conclusion and future work

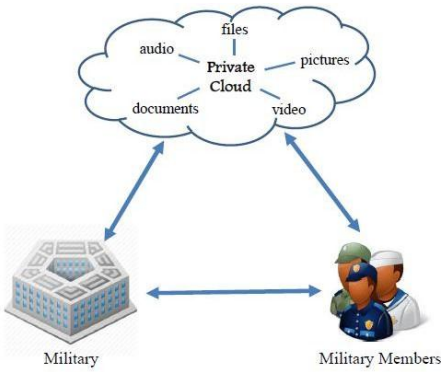
A. Conclusion

The proposed work forms a new data integrity verification protocol for cloud storage providing integrity protection of user's important data. It is proved to be secure against unauthorized users since it does not involve any trusted third party in data integrity checking operation.

It has very good efficiency in the aspects of communication, computation and storage costs. To exploit the strengths of this technology and to overcome the drawbacks in order to ensure data integrity and consequently a big data security on the cloud. Enabling the different integrity proofs to keep the data in secure manner. So that sensitive data is encrypted and stored in cloud service provider. Insensitive data stored directly without any encryption.

B. Future Work

In future work, is focused on protocol support for data dynamics using Embedded Merkel Hash Tree (EMHT) and a hardware support for data integrity checking. And to upgrade Dynamic data updating is provided by Merkle B-tree algorithm which provide aspects of good efficiency in the communication, computation and storage costs less.



In this project future work is used in military communication, computation is take place in our project to

securely store the date in to the cloud. Designing a secure, privacy preserving, and practical scheme for big data storage in a cloud is an extremely challenging problem.

In our future research, we will further improve our scheme by combining the (t;n)-threshold secret sharing with attribute based access control, which involves an access structure that can place various requirements for a user to decrypt an outsourced cipher-text data in the cloud. Meanwhile, we will investigate the security problems when a data owner outsources its data to multi-cloud servers and consider an attribute-based access structure that can be dynamically updated, which is more applicable for practical scenarios in big data storage.

References

[1] M. A. Beyer and D. Laney, "The importance of big data: a definition," Stamford, CT: Gartner, 2012.
[2] V. Marx, "Biology: The big challenges of big data," Nature, vol. 498, no.7453, pp. 255–260, 2013.
[3] G. P. Consortium et al., "A map of human genome variation from population-scale sequencing," Nature, vol. 467, no. 7319, pp. 1061–1073, 2010.
[4] A. Sahai and B. Waters, "Fuzzy identity-based encryption," Advances in Cryptology–EUROCRYPT 2005, pp. 457–473, 2005.
[5] C. Hu, F. Zhang, X. Cheng, X. Liao, and D. Chen, "Securing communications between external users and wireless body area networks," in Proceedings of the 2nd ACM workshop on Hot topics on wireless network security and privacy. ACM, 2013, pp. 31–36.
[6] C. Hu, H. Li, Y. Huo, T. Xiang, and X. Liao, "Secure and efficient data communication protocol for wireless body area networks," IEEE Transactions on Multi-Scale Computing Systems, vol. 2, no. 2, pp. 94– 107, 2016.
[7] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in Proceedings of the 13th ACM conference on Computer and communications security. ACM, 2006, pp. 89–98.
[8] B. Waters, "Cipher text-policy attribute-based encryption: An expressive, efficient, and provably secure realization," Public Key Cryptography– PKC 2011, pp. 53–70, 2011.
[9] C. Hu, N. Zhang, H. Li, X. Cheng, and X. Liao, "Body area network security: a fuzzy attribute-based signcryption scheme," IEEE journal on selected areas in communications, vol. 31, no. 9, pp. 37–46, 2013.
[10] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," Advances in Cryptology–EUROCRYPT 2011, pp. 568–588, 2011.
[11] C. Hu, X. Cheng, Z. Tian, J. Yu, K. Akkaya, and L. Sun, "An attribute based signcryption scheme to secure attribute-defined multicast communications," in SecureComm 2015. Springer, 2015, pp. 418–435.
[12] A. Shamir, "Identity-based cryptosystems and signature schemes," in Advances in cryptology. Springer, 1985, pp. 47–53.
[13] M. Dehkordi and S. Mashhadi, "An efficient threshold verifiable multisecret sharing," Computer Standards & Interfaces, vol. 30, no. 3, pp. 187–190, 2008.
[14] Z. Eslami and J. Z. Ahmadabadi, "A verifiable multi-secret sharing scheme based on cellular automata," Information Sciences, vol. 180, no. 15, pp. 2889–2894, 2010.
[15] M. H. Dehkordi and S. Mashhadi, "New efficient and practical verifiable multi-secret sharing schemes," Information Sciences, vol. 178, no. 9, pp. 2262–2274, 2008.
[16] J. Zhao, J. Zhang, and R. Zhao, "A practical verifiable multi-secret sharing scheme," Computer Standards & Interfaces, vol. 29, no. 1, pp.138–141, 2007.
[17] http://www.sourcefordgde.com
[18] http://www.networkcomputing.com/