ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 3, March 2015.

Security Mediator Using Blind Signatures and Key Rotation Algorithm

Sanket Chaudhari[1], Akanksha Singh[2], Sheetal Asopa[3]

B.E. Student, Computer Department, S.S.B.T's COET Bambhori, Jalgaon, MH-India [1,2,3]

Abstract: One of the most important issues in Network is security. When data is distributed across networks or information is transferred via public networks, it is more prone to be attack by mischievous elements. In traditional security model, all the data stored in server and the users who use that data belong to the same security level. The security service provided by the storage provider is not enough. Since the data is stored anywhere across the network, the data owner has less control over the data stored in servers. Due to the issues in security, both owners and users are advised to verify the integrity of data with Provable Data Possession (PDP) before further usage of data. However earlier methods either unnecessarily expose the identity of a data owner to untrusted servers or introduce overhead on verification metadata for preserving anonymity. The proposed system is to build a security service through a trusted third Party .The system introduces a security mediator (SEM) which does not store any data at its end. The system is also able to generate verification metadata onoutsourced data for owners

Keywords: Blind Signature, Decryption, Encryption, Key Rotation algorithm, Metadata, Provable data possession, Security, Security Mediator, Server

I. Introduction

Nowadays, number of organizations stores their large-scale data storage on servers across the network, thus it saves lots of money which is required for maintaining that data. With storage service, the members can share their data with other members easily by uploading their data to the server.

Since the stored data and operations performed on data are not visible to the data owners and security threat or unauthorized access to data is common, owners have huge concern with security of their data. To maintain data integrity, data owners can compute verification metadata on their data, and then upload both of them to the server [1].

However, the general method for protecting data integrity will conflict with another significant concern of data owner's identity privacy, or anonymity. Specifically, if digital Signatures are used to serve as verification metadata; they can only be verified with a data owner's public key. The unique binding between a public key and the identity of the data owner will inevitably reveal the owner's identity to any public verifiers [1].Refer to the example above, journalist shares the confidential data to

people agree to let other access the data stored in the storage servers across network, but is often unwilling to reveal his real identity because of his safety.

Another concern is how to allow data integrity checking without accessing whole data. Since size of stored data is huge the verifier must need access entire data and spent large amount of time and computation or communication resources .There are several methods for efficient data integrity checking but none of them is able to hide identity of owners of data to public verifiers.

Also before storing the data first it is encrypted using efficient method, then stored on servers. Usually to gain high efficiency the symmetric key algorithms are used but it does not provide the high security which is expected by data owners [2].

The rest of the paper is organized as follows; the existing security methods and proposed system are presented in the Section II Section III and IV, introduces preliminaries and detailed algorithm is given in V, Section VI gives the result and performance analysis. Finally, conclusion and future work is given in section VII

II. Related Work

An easy way to comply with the conference paper formatting requirements is to use this document as a template and s There are several data integrity methods there are efficient but major disadvantage of such techniques is they are unable to preserve identity of data owners to public verifiers. Provable data possession (PDP) proposed by Atenieseis a protocol which allows the integrity of data stored in an untrusted server to be audited without retrieving the entire data. But it leads to huge computation overhead which is performed by data owner before storing the data.

If digital signatures are used to serve as verification metadata, they can only be verified with a data owner's public key. The unique relation between a public key and the identity of the data owner will inevitably expose identity of the data owner to any public verifiers. Especially under public key infrastructure where such bindings are explicit via public key certificates [1].

Also existing key Symmetric algorithm uses same key for encrypting the all blocks of data before storing operation also uses same key while performing decryption .So such encrypted data is more prone to be attack by mischievous elements.

In Existing techniques the computation of verification metadata is at client side so it creates unnecessary overhead at client side[2].

III. PROPOSED SYSTEM

Propose system introduces a simple, efficient, and publicly-verifiable approach called Security Mediator (SEM) to ensure data integrity without sacrificing the anonymity of data owners and requiring significant overhead. Security-mediator (SEM), is also able to generate verification metadata (i.e., signatures) on outsourced data for data owner.
The SEM provides security services to data owners by generating signatures on data for owners before these data are outsourced to the server [1].

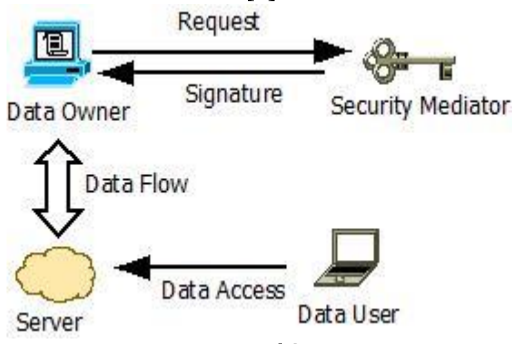

Fig. Proposed System

IV. PRELIMINARIES

This section gives all the structures and schemes used in this paper

A. *Blind Signatures*

Blind signatures were proposed by Chaum in 1982[3]. They define an interactive signature protocol between a user and a signer, guaranteeing that the signed data, and even the resulting signature, are unknown to the signer; this property is called blindness. More precisely, if the signer runs several executions of the protocol leading to several data/signature pairs, he cannot link a pair to a specific execution: the view of the signer is unlinkable to the resulting data/signature pair[3]. In detail data owner blinds the content of the data and sends the blinded data to the signer i.e. Security Mediator. After receiving the blinded message, the SEM generates a signature on the blinded message and returns it to the data owner. The message owner is able to recover and output the signature on the original data based on the result returned by the signer and the blinding factor owner has chosen [1].

There are several applications of blind signatures. Following are impotent applications of blind signature scheme.
1 E-Voting
2 E-Cash
3 Data aggregation in networks

B. *Data Structures*

There are several data structures which are required for implementation of key rotation algorithm [2].

1) Block Status Table(BST):-
The Block Status Table (BST) is a data structure used to access the encrypted file from the server. It consists of two columns such as, sequence number of dataa block in the file and the data block number. The BST can be implemented using linked list.

Sequence Number	Block Number
1	1
2	2
3	3
4	4

.Fig.Block Status Table

2) Key Chooser (KC):
The key chooser is a important component which defines the criteria for selecting key character.
3) Circular Array(CA):
The circular array is used in both encryption and decryption process. The circular array is used for shift operations on both character and on a key character.
4) CA Inverter (CAI):
The CA Inverter inverts the circular array for high degree of security.
5) CA Shifter (CAS):
The CA shifter shifts or rotates the circular array.

V. ALGORITHMS

Process starts with obtaining the blind signature on blocks of data from Security Mediator. The data owner first send the data blocks to Security Mediator, then Security Mediator computes blind signature on data.
Algorithm for Blind Signature is given below [3].

A. ***Algorithm for Blind Signature***

A blind signature scheme BS is a 4-tuple of polynomial time algorithm (Setup; KeyGen, <S, U>, Verify) in [3]:
1) Setup (1^{λ}), where λ is the security parameter, generates the global parameters param.
2) KeyGen(param) generates a pair of keys (vk,sk):
3) Signature Issuing is an interactive protocol between the algorithms S(sk) and U(vk;m), for a
Message m∑M. It generates an output σ for the user: σ← <S(sk), U(vk,;m)>.
4) Verify (vk,m,σ) outputs 1 if the signature σ is valid w.r.t. m and vk, 0 otherwise.

```
$\mathsf{Exp}^{euf}_{Sig,\mathcal{A}}(\lambda)$
1. $\mathsf{param} \leftarrow \mathsf{Setup}(1^{\lambda})$
2. $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$
3. $(m^*, \sigma^*) \leftarrow \mathcal{A}(\mathsf{vk} : \mathtt{OSign}(\mathsf{sk}, \cdot))$
4. $b \leftarrow \mathsf{Verify}(\mathsf{vk}, m^*, \sigma^*)$
5. IF $m^* \in \mathcal{SM}$ THEN RETURN 0
6. ELSE RETURN $b$
```

Fig. Blind Signature Algorithm

After computing the blind signature the data owner verifies the data blocks and finally stores the data to server. Before storing data, it is encrypted using key rotation algorithm.[2]

Steps of Storing data are given below
1) The data owner splits the source file in to blocks of 128 characters and encrypts all the blocks using key rotation algorithm. And prepare the Block Status Table (BST) for encrypted blocks, and then send the encrypted file, key, BST to the Security Mediator.
2) The SEM calculate the combined hash values for BST (TH) and encrypted file (FH),then send only encrypted file and BST to the server for storage
3) The authorized user sends the data access request to both SEM and server
4) SEM verifies the authorized user, if the user is verified then, it sends the authorization signal to the server.
5) SEM sends the hash values of BST (TH) and encrypted file (FH) to the requested user.
6) Server sends the BST and encrypted file to the users.
7) User calculate the hash values of BST and encrypted file received from the server then verifies with hash values received from the SEM.
8) If both values are verified then user gets a data decryption key and decrypt the data blocks.

The Overall process is represented in diagram

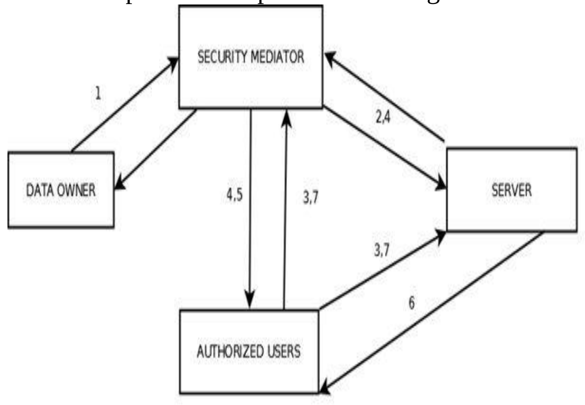

Fig.Overall System Model

B. Key Rotation Algorithm
1) Algorithm for Data Encryption [2]
Algorithm Data Encryption (Source file (F), X, Decrypted file)
Input: File for encryption, X- Encryption Key
Output:CF- Ciphertext of a file F
1) Split the characters of the file/string into blocks.
2) Get the mapped values from Em.
3) Get the binary equivalent of the current character.
4) Remove the first character from the binary value and store it into First Character and consider rest of binary value for shift operations (this is done to avoid having case the resultant value 00001 or 01111 after, shift operations this would result into forgetting a bit as 0001 can also represented as 1).
5) Store binary value into CA for bit operations.
6) The key chooser component selects a key character from ith position, such a way that if chunk character is selected for encryption then select the first character of the key, so i is selected chunk character > size of key (|X|) gets the modulus of the chunk character position.
7) Add the selected ith key character int value and its previous (i-1)th character int value instance if 1st key character is selected then the previous character would the 16th character (key stored into circular array double or way linked list for this operation).
8) The CA Inverter component complements Circular array (ICA) if the summed result is even per the equation (10).
9) Add the selected ith key character and (i+2)th key character.
10) The CA shifter shifts the Circular to find SCA as per the equation (11).
11) Add the stripped off first Character to the resultant binary value of Circular Array obtained previous step.
12) Get the character equivalent of the binary value which is the cipher character.
13) Repeat steps 2 through 12 for all the chunks with different Key (by shifting the key character by using Circular Array a double way linked list.
2) Algorithm for Data Decryption [2]
Algorithm Data Decryption (Ciphertext of a file (F), X, Source file)
Input: CF- Ciphertext of a file, X- Encryption Key
Output: Source file (F)
1) Split the characters of the file/string into blocks.
2) Get the binary equivalent of the current ciphertext.
3) Remove the first character from the binary value and store it into First Character and consider rest of binary value for shift operations.
4) Store binary value into CA.
5) Select a key character from ith position such a way that if 1st chunk character is selected for encryption then select the first character of the key, so i=1. If selected chunk character > size of key (|X|) get the modulus of chunk character position.
6) Add the stripped off first Character to the resultant binary value of CA after bit operations.
7) Add the selected |X|i and |X|i+2 key character.
8) The CA shifter shifts the Circular to find SCA as per the equation (11).
9) Add the selected ith key character int value and its previous (i-1)th character int value instance if 1st key character is selected then the previous character would the 16th character (key stored into circular array double or way linked list for this operation).
10) Get the character equivalent of the binary value.
11) Get the mapped value from Em, which is the decrypted value of the character.
12) Repeat steps 2 through 12 for all the chunks with different Key (by shifting the key character by using Circular Array a double way linked list).

VI. RESULT

Now we evaluate the performance of scheme. The experiment is carried out on text files with different size. The algorithms are implemented in JAVA. The Tomcat Server forms the complete execution environment.

As compare to previous scheme we use Security Mediator for generate signature. But in this scheme data owner waste little amount of time while generating signature on data which is in milliseconds. And computation cost of generation of signature is n(x+3) + 2nPair. Where n is no of signature andx is no element in block of data.

The data file is encrypted before storing in server. If File has N blocks and if every block has n characters then inversion and shifting is performed by $N*n$ *operations*. And therefore key rotation algorithm has complexity of $O(N*n)$.

VII. CONCLUSION

Security Mediator reliefs the client from maintaining any type of key information and allowing the client for using any browser enabled device to access the services. Thus the client can share the data securely with specific group of people without any overhead. Proposed System Approach separates the anonymous protection mechanism from the PDP mechanism via the use of security mediator. This Solution not only minimizes the calculations and requirement of this mediator, but also minimizes the trust placed on it in terms of data privacy and identity privacy. Future Scope will beto enhance the more security. Also the multi-SEM model can be implemented to reduce chances of failure and reduce the overhead of network traffic can be anotherarea of research.

ACKNOWLEDGMENT

The heading of this material is based on work supported by **Mr Manoj L. Mahajan** and H.O.D of Computer Department **Prof. Girish K. Patnaik** from SSBT’s COET, Jalgaon.

REFERENCES

[1] Boyang Wang, Sherman S. M. Chow, Ming Li and Hui Li, Storing Shared Data on the Cloud via Security-Mediator", ICDCS'13, 2013
[2] Prakash G L ,Dr. Manish Prateek and Dr. Inder Singh, "Data Encryption and Decryption Algorithms using Key Rotations for International Journal Of Engineering And Computer Science, 4 April,2014.
[3] Olivier Blazy, Georg Fuchsbauer, David Pointcheval and Damien Vergnaud,"Short Blind Signatures", Journal of Computer Security, 2013.
[4] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H.Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin,I. Stoica, and M. Zaharia, “A View of Cloud Computing,”*Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.
[5] K. Ren, C. Wang, and Q. Wang, “Security Challenges for the Public Cloud,” *IEEE Internet Computing*, vol. 16, no. 1, pp. 69–73, 2012
[6] A Novel Multi owner Data sharing Group key protocol K.U.V.Padma, J.Anitha, K.Balaji.
[7] Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. Structure-preserving signatures and commitments to group elements. In CRYPTO 2010, volume 6223 of LNCS, pages 209{236. Springer, August 2010.
[8] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, “Provable Data Possession at Untrusted Stores,” in *ACM CCS*, 2007, pp. 598–610.
[9] D. Chaum and E. van Heyst, “Group Signatures,” Proc. Int’l Conf. Theory and Applications of Cryptographic Techniques (EUROCRYPT), pp. 257-265, 1991.
[10] B. Wang, B. Li, and H. Li, “Oruta: Privacy-Preserving Public Auditing for Shared Data in the Cloud,” in *IEEE Cloud*, June 2012, pp. 295–302.