www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 4 Issue 1 January 2015, Page No. 10184-10186

Identity Based Secured Distributed Data Storage Schemes

Dhanshri Patil, Tejal Pardeshi[1], Nikita Mane[2], Akshaykumar Thakur[3],

[1]N.M.I.E.T. Department of Computer, Pune University,
Talegaon Dabhade
6nikitamane@gmail.com

[2]N.M.I.E.T. Department of Computer, Pune University,
Talegaon Dabhade
09tejalp@gmail.com

[3]N.M.I.E.T. Department of Computer, Pune University,
Talegaon Dabhade
akshay20thakur@gmail.com

Abstract: ***Identity based secured distributed data storage is such a scheme that reduces the load of an owner to maintain excessive number of files. Hence, it reduces the ramification of the owner to take care of all the files that he has with him. In this, the owner passes his files to the mediator between the owner and the user that is the proxy server. The proxy server plays an important role in this scheme. It encrypts the files that are given to him by the owner that are encrypted once, without having any cognition about the contents of the files. The proxy server also stores these encrypted files and gives the files to the authenticated users. When these files are given to the proxy server, the owner deletes those files from his system for the benefit of space efficiency. Hence, it is necessary to carefully examine the issues of confidentiality and integrity of the outsourced files. IBSDDS schemes possesses the properties: firstly, the file owner can decide access permission independently for each file and secondly, a user can access only one file for a single query. Resultantly, by using this scheme, the collusion attacks, means the disambiguation of data can be avoided.***

Keywords: DDS, IBS, AC, Security.

1. Introduction

Nowadays, users are especially concerned on the issues of confidentiality and integrity when they store their files on cloud using cloud computing. To store the files of the users on cloud, the DAS (database as a service) scheme [1][2] was put forward. But this scheme was not able to fulfill the requirements of the users about the security of their files. It means the user was unable to guarantee that his files are accessed by the authorized users only and also the files are not being modified by the proxy server when they are outsourced by the owner. Also there was a problem that how to guarantee that the requested file by the receiver is received to him correctly by the proxy server because the proxy server only maintains the cipher texts of the files.

2. Related Work

In this section we describe the architecture of our system. The overall architecture of the system can be subdivided into three main modules: (1) Owner, (2) Proxy Server, and (3) Receiver. Figure.1 shows the system architecture of the question and answering system. Each module is described.

In the networked file systems, proxy servers are assumed to be intimate. They certify receivers and confirm access permissions. The communication between the proxy servers and receivers is executed in a protected channel. Therefore, these systems cannot afford an end-to-end data security, namely they cannot assure the secrecy of the data stored at the proxy server [3]. In these schemes, a receiver authenticates himself to the proxy server using his password. Then, the proxy server sends the certificated result to the file owner. The owner will make avenue permission according to the received information.

In the storage based intrusion detection systems, an intrusion detection scheme is embedded in proxy servers or the file owner to detect the intruder's behaviors, such as adding backdoors, inserting Trojan horses and tampering with audit logs. These schemes can be classified into two types: host-based system and network-based system. In the host-based systems, an intrusion detection scheme is embedded in the host to detect the local intrusion actions [1]. On the contrary, in network-based systems, an intrusion detection scheme is embedded in the proxy servers to detect the external intruder's actions. The main advantage of these systems is that proxy servers can still detect the intrusion actions even if the host is compromised as the proxy server are independent from the host [4], [5].

3. Existing System

The Cloud computing provides users with a convenient mechanism to manage their personal files with the notion called database-as-a-service (DAS). In DAS schemes, user can give his encrypted files to proxy servers. Proxy servers can carry out some operations on the cipher texts without having any understanding about the commencing files. Lamentably, this approach has not been implemented broadly. The main reason is that users are especially anxious on the issues of familiarity, forthrightness of their data. Also the users are lot more worried about the query of the outsourced files because cloud computing is a lot more problematic than the local data storage systems, as the cloud is administered by an untrusted third party. After sending the files to proxy servers, the user will delete them from his confined machine. Therefore, a question arises that how to assure that the outsourced files are not achieved by the unauthorized users and not modified by proxy servers. Also how to ensure that an authorized user can query the outsourced files from proxy servers is another concern as the proxy server only maintains the outsourced encrypted files.

4. Proposed System.

We are putting forth two identity based secure distributed data storage(IBSDDS) schemes in canonical model,in which the receiver can only access one of the owner's files, for one query, instead of all files. We can also say that an access permission is bound not only to the authentication of the receiver but also to the file. The access permission can be adjudged by the owner, instead of the trusted party (PKG). Also, our schemes are secure against the collusion attacks.

It contains two security schemes: CPA secure and CCA secure respectively. In order to achieve stronger security and implement file-based access control, the owner must be online to authenticate legal users and also to generate access permission for them. Therefore, the owner in our schemes needs to do more operations than that in PRE schemes. PRE schemes can provide the similar functionalities like our schemes. But when the owner only has multiple files, these are not flexible and practical.

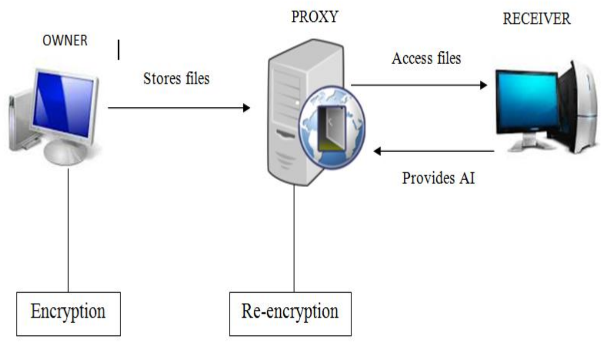

Fig.1. System Architecture

5. Mathematical Model

1. Let S be a system that describes Triple DES.
   S = {…}

2. Identify input as I.
   S = {I,..}
   Let I={i1,i2,i3,….i_d}
   The input will be file name and secret key.

3. Identify output as O
   S = {I,O,…}
   O = The receiver will receive the file when authenticated.

4. Identify the processes as P
   S={I,O,P,..}
   P={E,D}
   E={parameter,id,msg}
   D={parameter,Skid,CTi}

5. Identify failure cases as F
   S={I,O,P,F}
   S=When data is accessed by authorized user.

6. Identify the initial condition as Ic
   S={I,O,P,F,Ic}

7. Identify success cases as s.
   S={I,O,P,F,Ic}

6. Identity-based Secure Distributed Data Storage (IBSDDS)

There are four identity-based secure distributed data storage(IBSDDS) schemes: the private key generator(PKG), the data owner, the proxy server and the receiver. The PKG validates the users' identities and issues secret keys to them. The data owner encrypts his data and outsources the ciphertexts to the proxy servers. Proxy server stores the encrypted data and transfers the ciphertext for owner to the ciphertext for the receiver when they obtain an access permission(re-encryption key) from the owner. The receiver authenticates himself to the owner and decrypts the re-encrypted ciphertext to obtain the data.

7. Algorithm

We need so many algorithms. But the basic and important algorithms of all algorithms we are using here is:

1. **Setup Algorithm:** The setup algorithm takes as input a security parameter 1_, and outputs the public parameters and a master secret MSK.

2. **KeyGen:** The key generation algorithm takes as input the public parameters, an identity ID and the Master Secret Key MSK, and outputs a secret key S for the identity ID.

3. **Encryption**: This algorithm Suppose that there are k messages {M1,M2, ・・・,Mk}. To encrypt the message Mi, the encryption algorithm takes as input the public parameters, the identity ID and the message Mi, and outputs the cipher text= 1, 2, ・・・, k. It sends the cipher texts to the proxy servers.

4. **Query algorithm**: It takes as input the receiver's identity ID_, the receiver's secrete key and the cipher text, and outputs an authentication information AI. It sends (ID_,AI,CT) to the proxy server. The proxy server redirects (ID_,AI,Ci,2) to the owner with identity ID.

5. **Permission algorithm**: This Algorithm checks the authentication information AI. If the receiver is legal, this algorithm takes as inputs the public parameters, the receiver's identity ID_ and the owner's secret key, and outputs an access permission (re-encryption key) .

6. **Re-encryption**: The re-encryption algorithm takes as input the public parameters, the receiver's identity ID_, the access permission and the cipher text, and output cipher text.

7. **Decryption:** There are two algorithms. One is for the owner and the other is for the receiver.

i. **Decryption1-**The owner decryption algorithm takes as input the public parameters, the owner's secret key and the cipher text, and outputs the message Mi.

ii. **Decryption2-**The receiver decryption algorithm takes as input the public parameters, the receiver's secret key and the re-encrypted cipher text.

8. **Proxy Re-encryption Algorithm:** This algorithm is going to take the input as the public parameters, the receivers ID, and Cipher text that may be the owners (or) receivers and gives the output as Cipher text to the receivers with ID's.

8. Features

1.Unidirectional. After receiving an access permission, the proxy server can transfer a ciphertext for Alice to a ciphertext for Bob while he cannot transfer a ciphertext for Bob to a ciphertext for Alice.

2.**Non-interactive**. The access permission can be created by the file owner without any trusted third party and interaction with him.

3.**Key Optimal**. The size of the secret key of the receiver is constant and independent of the delegations which he accepts.

4.**Collusion-safe**. The secret key of the file owner is secure even if the receiver can compromise the proxy server.

9. Technologies

Java Server Pages (JSP)

It is a technology that helps software developers to create dynamically generated web pages based on HTML, XML, or other document types. JSP can be used independently or as the view component of a server side model-view-controller design, normally with Java beans as the model and Java servlets as the controller. Architecturally, JSP may be viewed as a high level abstraction of java servlets. JSP allows java code and certain pre-defined actions to be interleaved with static web mark-up content, with the resulting page being compiled and executed on the server to deliver a document. The compiled pages, as well as any dependent Java libraries, use java byte code rather than a native software format. Like any other Java program, they must be executed within a Java Virtual Machine (JVM) that integrates with the Server's host operating system to provide an abstract platform-neutral environment.

10. Conclusion

Cloud computing is a distributed system where users in different domains can share data among each other. Identity based proxy re-encryption schemes have been proposed to outsource sensitive data from the owner to an external party. But they cannot be employed in cloud computing, as security of data storage is important. Also, the security of data transfer is important. We achieve the security of data transfer by introducing the conditional proxy re-encryption. It provides many advantages like chosen ciphertext attacks, uni-directionality and collusion-resistance over the previous schemes. This scheme provides secure model of cloud storage with safe data forwarding..

References

[1] H. Hacig¨um¨us, B. R. Iyer, C. Li, and S. Mehrotra, "Executing SQL over encrypted data in the database-service-provider model," in *Proceedings: SIGMOD Conference - SIGMOD'02* (M. J. Franklin, B. Moon, and A. Ailamaki, eds.), vol. 2002, (Madison, Wisconsin, USA), pp. 216–227, ACM, Jun. 2002.

[2] L. Bouganim and P. Pucheral, "Chip-secured data access: Confidential data on untrusted servers," in *Proc. International Conference on Very Large Data Bases - VLDB'02*, (Hong Kong, China), pp. 131–142, Morgan Kaufmann, Aug. 2002.

[3] Executing SQL over encrypted data in the database-service provider model.
AUTHORS: H. Hacigumus, B. R. Iyer, C. Li, and S. Mehrotra.

[4] Improved proxy re-encryption schemes with applications to secure distributed storage.
AUTHORS: G. Ateniese, K. Fu, M. Green, and S. Hohenberger.

[5] Efficient and private access to outsourced data
AUTHORS: S. D. C. di Vimercati, S. Foresti, S. Paraboschi, G. Pelosi, and P. Samarati.