Data Auditing in Cloud Environment using Message Authentication Code

K. Govinda
SCSE, VIT University, Vellore, India

E.Sathiyamoorthy
SITE, VIT University, Vellore, India

ABSTRACT
Cloud computing can be defined as a platform which provides different users an opportunity to store their data in the cloud. This data can be retrieved by any of the users or the service provider of that cloud. Data integrity is one of the major aspects of cloud computing services that prevails in a cloud environment. A cloud user needs to consider about data integrity apart from the confidentiality of data , that is data is not modified or corrupted by the service provider or other users. In this paper, we are going to propose a method to ensure data integrity in cloud environment. We will assure that the data in the cloud has not been altered by using Message Authentication Code (MAC).

General Terms
Security Management, Auditing, Cryptography and Data Integrity

Keywords
Integrity, Cloud, MAC, XOR, Diffie-Hellman.

1. INTRODUCTION
Since its origin, Cloud Computing has been growing in its web-based services and taking over the IT industry**.** Cloud computing services are being availed by customers around the world who are enjoying the use of its multiple virtual server-based resources over the internet. It is a virtualized OS that runs on a virtual hardware[1]. Google, Microsoft and Amazon are some of the vendors for public cloud computing while VMware, Citrix and many more facilitate private cloud computing**.** The software's ability to manage applications, store data and provide online services to the user through any network on a PC, mobile or iPad has increased its consumption. The consumers can gain a fast access to applications and improve the resources of their infrastructure. External users are provided with this new scalable IT model on demand at a low cost, whereas the service itself is managed by the provider. But with such advantages there are always consequences and in this case the consequences relate to security risks. The concerns for the security of data and information of the user are growing and questions are being raised as to its authenticity. Cloud computing has a public sharing environment and the users need to be aware of the safety of their data on the software. The IT community is now focusing on the cloud computing security which has been discovered to possess some loopholes. Talking about some of the main security issues, the consumer cannot be sure of the privacy of his data stored in the cloud**.** Questions need to be asked about the access control to the data, since the information gets transferred to external sources and is managed by a third party.
The computing resources are shared and data security depends on the service providers**.** You need to know who has access to your data and how far do the boundaries extend for privacy privilege [2]. The service surpasses the in-house personal safety and information needs to be gained on who is controlling your data and how it is being managed. If the vendors try to avoid providing guarantees or going through audits then the security of your data might be at risk. The user should be able to scrutinize the situation and be aware of the existing breaches in privacy policies. The network control is managed by the third parties of cloud computing providers and the data might be available to other members of the party, including other cloud clients as shown in Fig1. The data can be tampered with while on a local machine in transit which could result in loss and theft.

The paper is organized as follows, section1 describes the introduction, section2 describes the literature review, the proposed method with DH, XOR and SHA1 is discussed in section3 followed by conclusion.

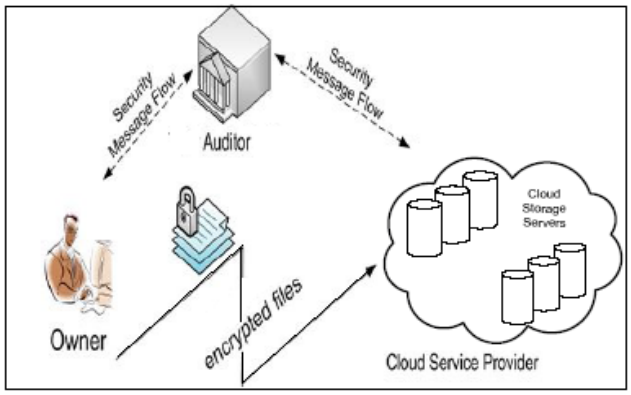

Figure 1.The Cloud Access Scenario

2. LITERATURE REVIEW
HMAC is a symmetric process that uses a secret key and a hash algorithm such as SHA to generate a message authentication code, or digest. This authentication code securely provides data integrity and authenticity because the secret key is required to reproduce the code. Digests for normal hash functions can be reproduced with no such constraint[5]. HMAC can protect against man-in-the-middle attacks on the message, but it is not designed to encrypt the message itself. The HMAC function was published by Bellare et al. (1996), which includes analysis and a proof of the function's security, and it is standardized in FIPS PUB 198 (NIST, 2002)[6-7]. Any hash algorithm can be used with HMAC including MD5, SHA-1, SHA-256, etc.

The output of HMAC is a binary authentication code equal in length to the hash function digest. The security of HMAC is directly related to the underlying hash function used, so it is

weaker with MD5 and stronger with SHA-512. Forgery and key recovery attacks threaten HMAC, but typically require a large number of message/digest pairs for analysis. The HMAC functions used in the implementation of the HTEE scheme are based on the SHA-1 hash algorithm. The use of HMAC SHA1 specifies some data sizes that are important in the HTEE implementation such as a 64 byte key size 20 byte digest output size.

3. PROPOSED METHOD

Data integrity is one of the important aspect of cloud computing. Maintaining data integrity in the cloud is a major challenge that is faced by today's cloud users. Data integrity refers to the assurance by the user that the data is not modified or corrupted by the service provider or other users. In this paper, we are going to propose a method to ensure data integrity in the cloud using message authentication code (MAC).The data will be stored in the cloud by a user and the integrity of the data will be checked by his auditor. For ensuring the integrity of that data, we will be using three approaches in this paper.

- Diffie-Hellman(DH) algorithm used to generate the symmetric key that is known by both, the user and the auditor.
- Exclusive- OR(XOR) to perform an xor operation between the message and the key generated using DH algorithm.
- Secured Hashing Algorithm(SHA1) – to generate a separate key using a one-way hash function by passing the original message to the hash function. This is done by both the user and the auditor and the value obtained from the hash function by both of them is compared and hence the data integrity is verified.

HMAC is a symmetric process that uses a secret key and a hash algorithm such as SHA to generate a message authentication code, or digest. This authentication code securely provides data integrity and authenticity because the secret key is required to reproduce the code. Digests for normal hash functions can be reproduced with no such constraint[8]. HMAC can protect against man-in-the-middle attacks on the message, but it is not designed to encrypt the message itself. The HMAC function was published by Bellare et al. (1996), which includes analysis and a proof of the function's security, and it is standardized in FIPS PUB 198 (NIST, 2002). Any hash algorithm can be used with HMAC including MD5, SHA-1, SHA-256, etc.

The output of HMAC is a binary authentication code equal in length to the hash function digest. The security of HMAC is directly related to the underlying hash function used, so it is weaker with MD5 and stronger with SHA-512. Forgery and key recovery attacks threaten HMAC, but typically require a large number of message/digest pairs for analysis. The HMAC functions used in the implementation of the HTEE scheme are based on the SHA-1 hash algorithm. The use of HMAC SHA1 specifies some data sizes that are important in the HTEE implementation such as a 64 byte key size 20 byte digest output size.

3.1 Diffie Hellman algorithm

In DH algorithm, both the parties agree over a common key which is found using this algorithm[3]. The steps for executing the algorithm are:-

- A number 'p' ,which is prime number and 'g' be the generator of p.
- Alice picks a number 'a' which is known only to her and Bob picks a number 'b' which is known only to her. Alice computes her DH value (A) using the formula $A=g^{a} \bmod p$ and Bob computes her DH value (B) using the formula $d2=g^{b} \bmod p$. The DH values are exchanged between both.
- Alice calculates her key value k1 as $k1=B^{a} \bmod p$ and Bob calculates her key value k2 as $k2=A^{b} \bmod p$.

The keys k1 and k2 will be equal as shown in Fig2[10-12].

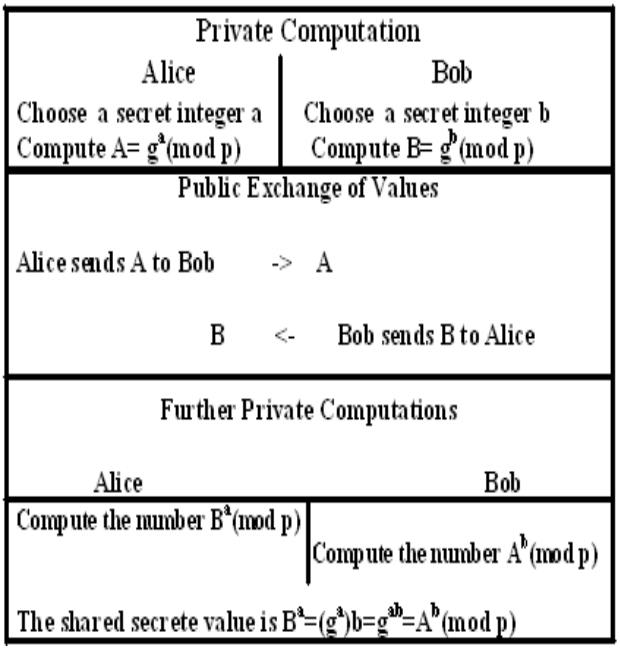

Fig2. Key Exchange

3.2 Exclusive OR Operation

XOR is an operation in which same bits produce a resultant bit as 0 whereas different bits produce a resultant bit as 1.In this, an XOR operation is done on the original text along with the secret value which gives a cipher text. The original text can be obtained back by performing an XOR[4] operation between the secret key and the resultant cipher text.

3.3 Secured Hashing Algorithm (SHA1)

This is an algorithm that uses one way hashing function. It is used to generate a hash digest which is used as a fingerprint for the data. The data is given to the function which produces a value (h1), a hash value. This value (h1) is sent to the receiver. On the other side, the receiver also passes the message to the function to generates a value(h2),a hash value.

The receiver compares the values h1 and h2 as shown in Fig3. If both the values,h1 and h2 are same then the data is safe otherwise the data is tampered.

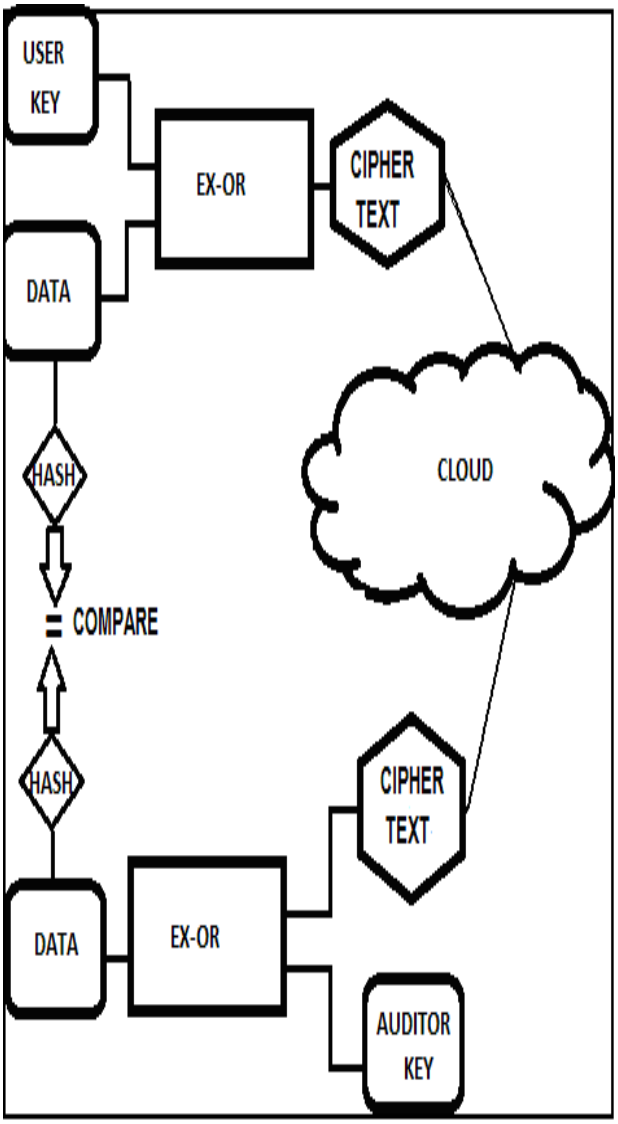

Fig3. Architecture

First of all, a secret key is generated using the DH algorithm and the same key is known by both the user and his auditor. After that an XOR operation is done between the data and the key generated. Separately the data is passed in a hash function (using SHA1) and the hash value is obtained by the sender.The auditor on behalf of the user will check the integrity of the data stored in the cloud. The auditor gets the cipher text from the cloud and performs an XOR operation with the secret key generated by the DH algorithm and gets a plain text. The auditor passes this plain text to the same hash function (using SHA1) and obtains a hash value .He then compares this hash value with the hash value received from the user .If both the values are identical then the data integrity is maintained or else the data integrity is tampered.

4. CONCLUSION

In the proposed method data integrity is maintained by using simple message authentication mechanism (MAC) like XOR and Hashing by auditing the data at regular intervals, as a future work this can be implemented by using digital signatures.

5. REFERENCES

[1] T.Mather, S.Kumarasuwamy, S.Latif ," Cloud Security and Privacy ",O'Rielly

[2] J.W.Rittinghouse, J.F.Ransome,"Cloud Computing :Implementation, Management and Security" CRC Press.

[3] Rescorla,E.,"Diffie –Hellman Key Agreement Method",RFC 2631,IETF Network Working Group.

[4] Atul Kahate,"Cryptography and Network Security".

[5] American Bankers Association, *Keyed Hash Message Authentication Code*, ANSI X9.71, Washington, D.C., 2000.

[6] National Institute of Standards and Technology, *Security Requirements for Cryptographic Modules*, Federal Information Processing Standards Publication 140-2, May 25, 2001.

[7] H. Krawczyk, M. Bellare, and R. Canetti, *HMAC: Keyed-Hashing for Message Authentication*, Internet Engineering Task Force, Request for Comments (RFC) 2104, February 1997.

[8] http://cs.uccs.edu/~gsc/pub/master/bbaker/doc/final_paper_bbaker_cs592.doc.

[9] P. Mell and T. Grance, "Draft NIST working definition of cloud computing," Referenced on June. 3rd, 2009 Online at http://csrc.nist.gov/groups/SNS/cloud-computing/index. html, 2009.

[10] 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, "Above the clouds: A berkeley view of cloud computing," University of California, Berkeley, Tech. Rep.UCB-EECS-2009-28, Feb 2009.

[11] Ian F. Blake and Theo Garefalakis, On the complexity of the Discrete Logarithm and Di_e-Hellman problems, Journal of Complexity 20 (2004), 148–170.

[12] Li Xin, An Improvement of Diffie-Hellman Protocol, Network & Computer Security, 2007,12, pp. 22–23.