ENHANCING SECURITY OF RESOURCE MATCHMAKING BY PARTITIONING FILES ACROSS MULTIPLE CLOUDS

Pranamitha Nanda,
Assistant Professor,
Department of Computer Science and Engineering,
Velammal Institute of Technology,
Chennai,Tamilnadu.

B.Daniel Nepoleon,
Department of Computer Science and Engineering,
Velammal Institute of Technology,
Chennai,Tamilnadu.

M.Mohamed Faizal Shariff,
Department of Computer Science and Engineering,
Velammal Institute of Technology,
Chennai,Tamilnadu.

K.Nethajhi,
Department of Computer Science and Engineering,
Velammal Institute of Technology,
Chennai,Tamilnadu.

Abstract: This paper proposes a security enhancement option for resource matchmaking across multiple clouds. Through analysis of the integrated relationship between users, dealers, and service resources, this paper proposes a scheme of hiding resources that can effectively reduce the user's load and improve reliability of the system. Based on multidimensional resource service operators, we model the problem of security by hiding files by partitioning them into many files and encrypting them with a key. This adaptive approach can overcome the limitations of traditional schemes, whereby the resources are shared with service providers who are anonymous to the user. As a resultthe broker can efficiently and accurately prepare the most trusted resources in advance, and thus provide more dependable resources to users. Our experiments yield interesting and meaningful observations that can facilitate the effective utilization of hiding files in a large-scale multi-cloud environment.

Index Terms—Cloud broker, multi-cloud environment, secure cloud, resource matchmaking

1.INTRODUCTION

Regardless of the developments in the field of computer science, security remains as a great concern for the user and other actors involved. User will have to send sensitive information to centers of the cloud, which is based on the relationship between the faith of users and service providers. The lack of security among users, and cloud and severely hinders any universal use of outsourced resources to cloud computing[3].

Although several scholars have been attracted by the trust and security question of cloud service, and many studies have been carried out [3], [4], [5], [6], a universal and expanded trust cum security scheme designed specifically for a multi-cloud computing environment is still lacking, and previous studies have some key limitations. We are quoting the research extensively done by done by Xiaoyong Li, Huadong Ma, Feng Zhou, and XiaolinGui in their paper[1] stressing on the need for trust.

(1) Few studies have focused on a trust-aware brokering framework for multi-cloud environments. Cloud brokers can provide intermediation and aggregation capabilities to enable providers to deploy their virtual infrastructures across multiple clouds [7]. The future of cloud computing will be permeatedwith the emergence of cloud brokers acting as intermediaries between cloud providers and users to negotiate and allocate resources among multiple data centers. Therefore, to increase the adoption of cloud services, a cloud broker should establish and provide trust management capacity to alleviate the worries of their users.

(2) Few studies have focused on an expanded trust model based on dynamic service factors of a cloud resource. From a user's perspective, trust is a comprehensive index for service guarantee and there are several trust factors in a system, i.e., security, availability, and reliability [5], [6]. As observed in [8], trust is beyond security, and an expanded trust model should incorporate security, reliability, and availability factors (and other factors if possible) into a trust vector. This data can be imported from existing models to form a comprehensive trust model. This highlights the fact that the level of trust in service resources should be objectively evaluated by dynamic service operators.

The idea of hiding files can be implemented in all clouds. This idea is mainly in tandem with the proposed idea of Xiaoyong Li, Huadong Ma, Feng Zhou, and XiaolinGui in their paper [1] on service operator aware trust scheme. Inspired by their idea of trust evaluation of various service providers in a multi-cloud environment we define a scheme of partitioning the resources uploaded. The evaluation of trust is left to cloud broker withrespect to the security, availability, and reliability of a resource within several specified time windows. The broker is attached to many registered resources.

II.RELATED WORK

This paper is mainly based on findings ofXiaoyong Li, Huadong Ma, Feng Zhou, and XiaolinGui who proposed a scheme of trust evaluation. They analyzed issues of trust through the built-in relationship between the user, the broker and the service resources. So far, many innovative trust schemes for cloud computing have been proposed by researchers.

III. ARCHITECTURE

This paper mainly focuses on the security of resources uploaded by the user coupled with the trust evaluation feature mention in the paper [1]; thus, we also assume that all cloud sites have a security option to prevent attacks from anonymous and malicious providers.

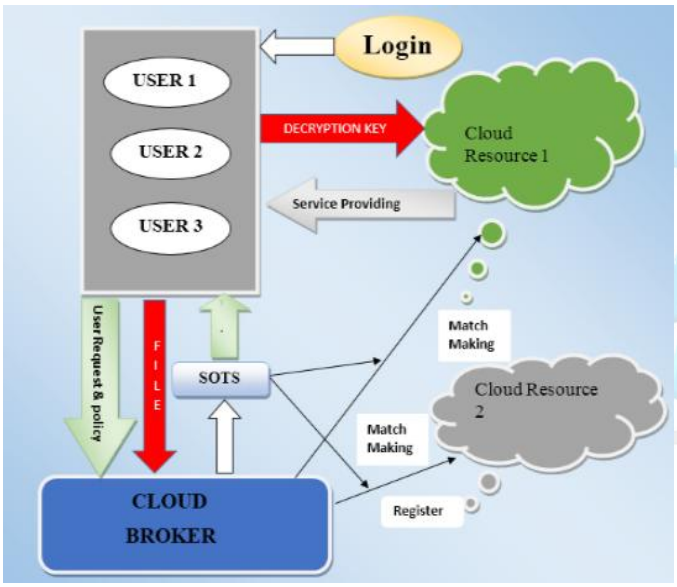

Figure 1: Architecture of partitioning file.

Figure 1 shows a schematic of our architecture. The proposed middleware architecture consists of a number of core modules, including the trusted resource matchmaking and distributing module, the adaptive trust evaluation module, the agent-based service operator acquisition module, and the resource management module, among others.

Adaptive trust evaluation module. This module is the core of the trust-aware cloud computing system, and is the major focus of this paper. Using this module, the broker can dynamically sort high-performance resources by analyzing the historic resource information in terms of providing highly trusted resources.

Trusted resource matchmaking and distributing module. In general, each cloud manager registers its service resources through the cloud broker. The service user negotiates with the service broker on the service-level agreement (SLA) details [7]; they eventually prepare an SLA contract. According to this contract, the broker selects, and then presents highly trusted resources to users from the trusted resource pool.

Agent publish and service operator acquisition module. This module is used to monitor the usage of allocated resources in order to guarantee the SLA with the user. In interaction, the module monitors the resource operators and is responsible for getting run-time service operators. Another task of the module is to publish automatically the monitoring agents (MA) in a remote site when a computing task is assigned to the site. Resource register module. It manages and indexes all the resources available from multiple cloud providers, and obtains information from each particular cloud resource, acting as pricing interface for users, and updating the database when new information is available.

Trusted third party is a service provider trusted by the user. A user will trust a service resource if the matchmaker or broker states that the resources' operators will match the users request.

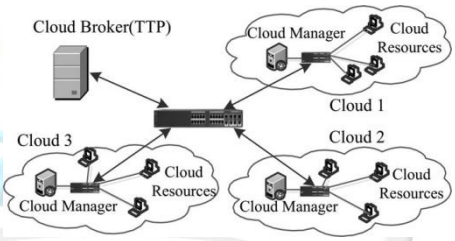

Figure 2. Multiple cloud experimental environment.

File is the resource that the user uploads on the cloud. Decryption key is the key provided by the user to the matched service provider to merge the partitioned files.

IV. ALGORITHM

The scheme to partition the file and encrypt each part consists of two algorithms. One to partition the file and the other to encrypt it.

Partition algorithm:

The algorithm for partition is simple. The file can be partitioned into three parts. The resource file is usually a text file. Hence the partition algorithm is simply framed to divide the file. A counter is initialized and set to the number of lines in the file. Then the number of characters in the file are calculated and each line is divided, thus the whole file is partitioned into three parts, incrementing the counter each time so.

AES ALGORITHM

AES is based on a design principle known as a substitution-permutation network, combination of both substitution and

permutation, and is fast in both software and hardware.Unlike its predecessor DES, AES does not use a Feistel network. AES is a variant of Rijndael which has a fixed block size of 128 bits, and a key size of 128, 192, or 256 bits. By contrast, the Rijndael specification *per se* is specified with block and key sizes that may be any multiple of 32 bits, both with a minimum of 128 and a maximum of 256 bits.

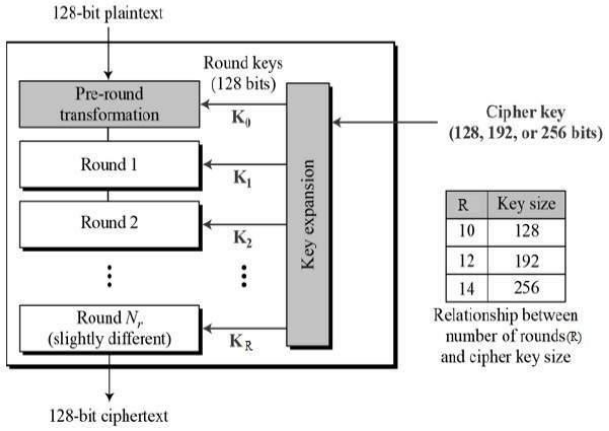

Figure 3. AES Algorithm schematic diagram

Each round comprise of four sub-processes.

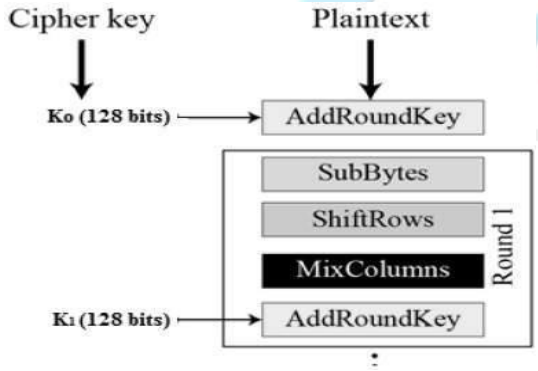

Figure 4.The first round diagram

Byte Substitution (SubBytes)

The 16 input bytes are substituted by looking up a fixed table (S-box) given in design. The result is in a matrix of four rows and four columns.

Shiftrows

Each of the four rows of the matrix is shifted to the left. Any entries that 'fall off' are re-inserted on the right side of row. Shift is carried out as follows −

- First row is not shifted.
- Second row is shifted one (byte) position to the left.
- Third row is shifted two positions to the left.
- Fourth row is shifted three positions to the left.

The result is a new matrix consisting of the same 16 bytes but shifted with respect to each other.

MixColumns

Each column of four bytes is now transformed using a special mathematical function. This function takes as input the four bytes of one column and outputs four completely new bytes, which replace the original column. The result is another new matrix consisting of 16 new bytes. It should be noted that this step is not performed in the last round.

Addroundkey

The 16 bytes of the matrix are now considered as 128 bits and are XORed to the 128 bits of the round key. If this is the last round then the output is the ciphertext. Otherwise, the resulting 128 bits are interpreted as 16 bytes and we begin another similar round.

Decryption Process

The process of decryption of an AES ciphertext is similar to the encryption process in the reverse order. Each round consists of the four processes conducted in the reverse order

- Add round key
- Mix columns
- Shift rows

Byte substitution

Since sub-processes in each round are in reverse manner, unlike for a Feistel Cipher, the encryption and decryption algorithms needs to be separately implemented, although they are very closely related.

V. IMPLEMENTATION STRATEGY

The implementation will include operations from three sides of the cloud infrastructure. The user will post on the system a request for cloud service. Any provider wishing to give the user his services should post the details of the services it provides to the cloud broker. Then the cloud broker can evaluate the various providers accepting to give their services using the trust evaluation scheme and develop a comprehensive ranking.

The user uploads his resource file onto the cloud system whereby the file is partitioned into three and each part is encrypted. The decryption key is available only with the user and it can be accessed only by those to whom the user grants authorization. The cloud broker is also kept in dark about the key and the contents of the resource file.
Then the user can choose among the listed providers and ask for service to whom he wishes. The key is shared to the concerned provider only thereby protecting the contents of the resource.

VI.CONCLUSION AND FUTURE WORK

In this work, we propose a security enhancing strategy for the resources of the user. The proposed idea hides the file even from the middleware. We have appended our idea with the one proposed in the paper [1]. The SOTS scheme in the paper [1] yields very good results in many typical cases. However, we can still improve our security mechanism by

using a more secure encryption technology. The trust scheme can be combined with a reputation building scheme as said in the paper [1]. We can also extend the security mechanism to include more file types and using a more secure partitioning algorithm.

VII. REFERENCES

[1] Xiaoyong Li, Huadong Ma, Feng Zhou, and XiaolinGui "Service operator aware trust sheme for resource matchmaking across multiple clouds" IEEE Transactions on parallel and distributed systems, Vol 26, No.5, May 2015.

[2] K. M. Khan and Q. Malluhi, "Establishing trust in cloud computing," IT Prof., vol. 12, no. 5, pp. 20–27, Sep./Oct. 2010.

[3] K. Hwang and D. Li, "Trusted cloud computing with secure resources and data coloring," IEEE Internet Comput., vol. 14, no. 5, pp. 14–22, Sep./Oct. 2010.

[4] H. Kim, H. Lee, W. Kim, and Y. Kim, "A trust evaluation model for QoS guarantee in cloud systems," Int. J. Grid Distrib. Comput., vol. 3, no. 1, pp. 1–10, 2010.

[5] P. D. Manuel, S. ThamaraiSelvi, and M. I. A. E. Barr, "Trust management system for grid and cloud resources," in Proc. 1st Int. Conf. Adv. Comput., Dec. 2009, pp.176–181.

[6] S. M. Habib, S. Ries, and M. Muhlhauser, "Towards a trust management system for cloud computing," in Proc. IEEE 10th Int. Conf. Trust, Security Privacy Comput. Commu., 2011, pp. 933–939.

[7] S. A. de Chaves, R. B. Uriarte, C. B. Westphall, "Toward an architecture for monitoring private clouds, " IEEE Commun. Mag., vol.49, no. 2, pp. 130–137, 2011.

[8] S. Clayman, A. Galis, C. Chapman, G. Toffetti, L. Rodero-Merino, L. M. Vaquero, K. Nagin and B. Rochwerger, "Monitoring service clouds in the future internet," in Future Internet Assembly. Amsterdam, The Netherlands: IOS Press, 2010, pp. 115–126.

[9] M. Alhamad, T. Dillon and E. Chang, "Conceptual SLA framework for cloud computing," in Proc. 4th IEEE Int. Conf. Digital Ecosystems Technol., 2010, pp. 606–610.

[10] AES Algorithm referenced from http://www.tutorialspoint.com/ and www.wikipedia.com