International Journal of Soft Computing 2 (3): 367-372, 2007

Adaptive Secured Service and Trust Management with Broker Architecture in a Grid Environment

S. Srinivasan, T.C. Rangarajan and V. Prasanna
Anna University, Chennai, Tamilnadu, India

Abstract: In broker architecture in the grid, providing a secured service is still a great challenge. Further, the trust values of the service provider need to be shared confidentially. As a grid contains a wide variety of services, the level of security too needs to be flexible with the type of service. Hence, in this study, the authors propose an adaptive security mechanism for such broker architecture. This multi-level security mechanism is mainly to ensure that a highly sensitive transaction be treated severely with security so that its confidentiality and the value of the data, being transmitted doesn't get lost. This is achieved with the shared key agreement with tickets for authentication and using a rotating key mechanism from there on.

Key words: Rotating key, adaptive security, tickets, trust value, grid, mechanism, broker

INTRODUCTION

Broker architecture (Varalakshmi *et al*., 2005; Harry, 2003) is one, in which a broker system mediates the transactions between the client and the service provider. In each domain cloud of the grid, there may be one or more brokers. Each broker has details about the Service Providers (SP) in his own domain and brokers in other domains. These details include the name of the SP, his location and his trust rank rated by the clients. When a client needs a service, he approaches the broker with the service request. The broker identifies the appropriate SP for that request by collaborating with the brokers in other domains. This may be done through the evaluation of the trust ranks of the SPs'. Then he informs the client, of the best SP he could transact with. The client then finishes the transaction and gives the feedback to the broker. The broker records it for future use. The weight of this rating is taken based on the details of the service. This study extends the existing broker architecture, adding the element of security to it Abadi (2002) which is adaptive in nature.

PROBLEM

The data exchanged between the client and the SP and also the trust values that the clients send to the broker are the real sensitive ones and they need to be secured. Furthermore, the sensitivity of the data too varies with the type of the service. So, each set of data needs to be given an adaptive treatment (Agostini *et al*., 2004; Agrawal *et al*., 1998; Xynogalas *et al*., 2004; Harry, 2003) even between the same parties. Also, the clients need to be validated by some mechanism, so that he does not misuses the services of the SPs. The data communication should also be able to protect the integrity (Jutla, 2001) and sensitivity of the data, evading eavesdropping and passive attacks.

RELATED WORK

Broker architecture in grid environment is one particular area, which has seen a lot of research activities, recently. Many architectures (Varalakshmi *et al*., 2005; Foster *et al*., 1998; Farag and Muthucumaru, 2002) involving the idea of broker, have been suggested. All these architectures have some common features like trusted broker service, ranking, trust based on feedback values etc. But, none of them dealt with the security issues involved in such architectures. The security of trust values and authentication mechanisms has not been considered in any studies. We propose a security model, an adaptive one (Agostini *et al*., 2004; Agrawal *et al*., 1998; Xynogalas *et al*., 2004) which can be used to secure the Broker architecture.

The CBC-MAC mechanism (Mihir *et al*., 1999; Daniel, 2005) for authentication purposes, gave us an idea of what is called the rotating key, which is discussed in the later part of the study.

ASSUMPTIONS

- Every time, a SP joins the Domain and registers himself under a Broker, the SP and the Broker have a Shared Key (SK) established between them, which

Corresponding Author: S. Srinivasan, No 55-C/1, Bharathi Nagar Main Road, Zamin Pallavaram, Chennai- 600043, India

will be used for future data exchange between them. This SK can be shared by using Diffie-Hellman Key Exchange mechanism or by any other similar means.

- Broker is trust worthy. This is the basic assumption in any architecture involving a third party.
- Every broker has a public-private key pair associated with him.

TERMINOLOGIES

Tickets serve the purpose of authenticating the client for the service provider. It is supplied to the client by the broker and the same must be submitted by the client, to the service provider, for utilizing the service. The *ticket* contains details like the client's name, the service he has been allowed to get from the SP, etc. Apart from these, it also contains a value, which remains unknown to the client. This is because, the client should be prevented from being able to create any duplicate ticket by himself to mislead the sp.This value is known, only to the broker and the sp.

Trust Value is the feedback provided by the client to the broker, about the service of the SP. This trust value is of great importance, since it is the basis for the ranking of the SP by the broker and for subsequent allocation of clients to the Sp.

Type of Service (ToS) denotes the kind of service requested by the client from the service provider. The broker selects the service provider, appropriate to the service requested for, by analyzing the rank values of SPs.

PROPOSED SECURITY MODEL

A representation of our proposed security model is given in Fig. 1, where a broker shares key between the active clients and the SP's. The figure represents scenario in one single domain and many such domains can co-exist.

The mechanism broadly consists of two steps

- Key agreement between the client and the SP. (Fig. 2).
- Data transfer between the client and the SP with rotating key mechanism (Fig. 3 and 4).

KEY AGREEMENT BETWEEN THE CLIENT AND THE SP

The message sequence diagram for key agreement between the client and the SP is shown in Fig. 2. Whenever a client is in need of a service, he sends a request message, REQ and a Symmetric key, Dx encrypted with the public key of the broker (K_{pub}) to the broker, i.e.,

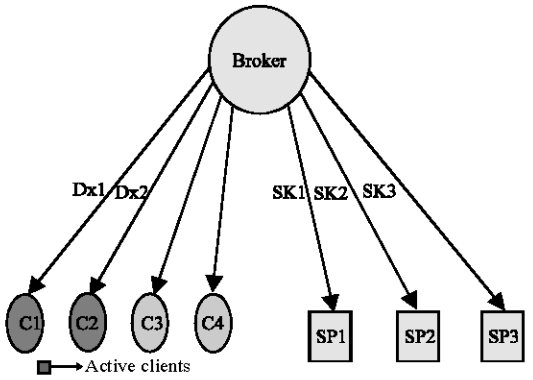

Fig. 1: Key storage involved at broker

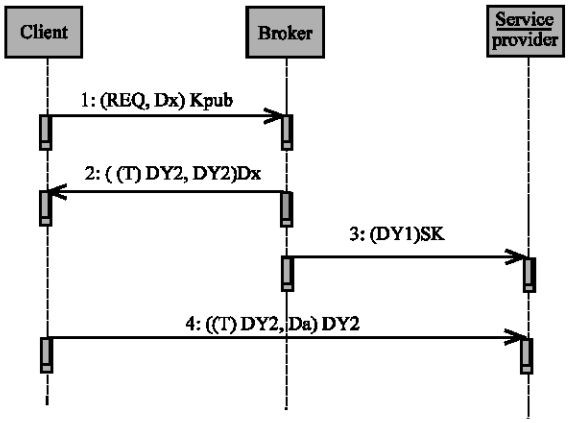

Fig. 2: Message sequence diagram depicting the key agreement mechanism

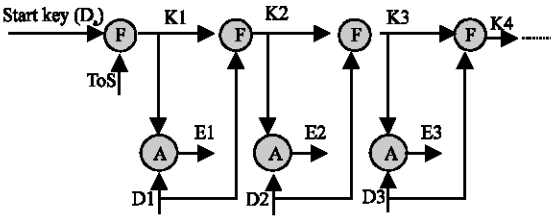

Fig. 3: Rotational key encryption

Service request message:

$$\text{Client->Broker: (REQ, } D_x \text{) } K_{pub}$$

The REQ message includes the Type of Service (TOS), the Client is in need for, from the SP. The key D_x, generated by the client, is used as a shared key between the client and the broker. This shared key is required to secure the trust value, which the client sends to the broker after completing the transactions.

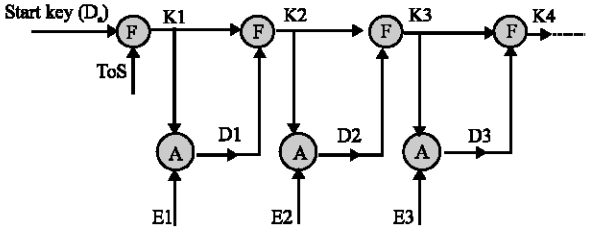

Fig. 4: Rotational key decryption

Service response message:

Broker-> Client: ((T) Dy_2, Dy_2) D_x

Broker decrypts this message using his private key (K_{pri}) and finds the type of request. He then finds the appropriate SP based on the trust value, generates a public-private key pair Dy_2 (public) and Dy_1 (private), respectively and sends the client two entities,

- A ticket to the SP encrypted with Dy_2. The ticket is a similar to a receipt, which the client should give the sp. in order to prove his authenticity. The unknown value in the ticket, discussed above, is taken as the Dy1, the private key itself.
- Dy_2 itself.

These 2 entities as a whole are encrypted using Dx.

The client decrypts the message and gets Dy_2. But, he is unable to decrypt and modify the ticket, since he doesn't know the private key Dy_1 (Clifford and Theodore, 1994). Also, the client cannot generate a ticket by himself, as he doesn't know the value of the Dy1.

Private key transmission message:

Broker-> SP: (Dy_1) SK

The broker now sends the generated private key Dy_1 to the SP encrypted through the pre-agreed shared key SK. An important point to note here is that, the SK is not constant through out the life span of the service provider. The broker changes the SK at random time intervals. The frequency of the key change depends on the *Rank* of the Service Provider. Higher the Rank, more frequent the key change is. This key change is notified to the service provider, who sends back an acknowledgement, after which the new key comes to effect. The broker deletes the public -private key pair and also the old SK, once this step gets over. As it is obvious, these are just temporary keys, which the Broker need not remember.

Connection request message:

Client->SP: ((T)Dy_2, D_a)Dy_2

The client sends the encrypted ticket and another shared key D_a to the SP, encrypted as a whole with Dy_2, the public key of SP. This symmetric key D_a is used for securing the data communication between the two parties, client and the SP. The size of the key D_a varies with the type of service request i.e., the ToS value. More sensitive the service, larger the key size. So, the level of security differs for different service types, since it takes more computational effort to break a bigger key. Hence adaptive security is achieved.

The SP decrypts the message, gets the encrypted ticket, decrypts it and recognizes that the Client is valid. He then accepts Da as the shared key between him and the client. The SP then sends an acknowledgement after which the actual data transfer takes place, encrypted using D_a and subsequent rotating keys.

After the transaction gets completed, the client sends the feedback, to the broker, encrypted by the key D_x. The feedback contains information like the identity of the client, the identity of the service provider, the trust value, the cost involved in the transaction and Type of Service (ToS).

DATA EXCHANGE BETWEEN THE CLIENT AND THE SP USING ROTATING KEY MECHANISM

Clients can request any type of service from the SPs, ranging from a simple service like printing service to a highly security-centric service like in banking applications. Each service requires different level of security treatment. A simple service which handles less sensitive data need not have high security overheads, while compromise on the security of highly sensitive data applications, cannot be made. This proposed mechanism involves a start key and a parameter called Type of Service (ToS) that decides on the level of security needed (Agostini *et al*., 2004; Agrawal *et al*., 1998; Xynogalas *et al*., 2004). In other words, ToS decides the size of the start key and it varies with different type of Client services (Bellare *et al*., 2000).

- Further, in order to protect client's data from passive attacks like dictionary attack or random key generation attack, the encryption scheme is strengthened by using the rotational key mechanism. start key is the key used initially to generate actual keys which are then used to encrypt the data. It is advancement over the Initialization Vector (IV) used

in WEP protocol in which a vector of initial values (keys) stored. Here, D_a is the Start key, which kick starts the key generating mechanism. Type of Service (ToS) is a value denoting the importance and criticality of the service.

Rotational key means that the same key is not used for encrypting the entire data. The mechanism uses different keys to encrypt different messages i.e., each key is used to encrypt not more than one message.

The mechanism is as follows:

- The data to be transmitted to the SP are divided into a number of equal-sized messages by the client.
- The ToS is then used to encode the start key to give a key K_1.
- The client then uses K_1 to the encrypt D_1.
- Now, the data D_1 and the Key $K_1$are given to a function, which results in K_2.
- Again, the client encrypts D_2 message of the data using K_2 and this mechanism repeats.

The rotating key mechanism, used above can be mathematically represented by the following

Let 'n' represent the round of the encryption.

When n = 1,

It can be easily inferred from Fig. 3 that the value of the Key K_1 is,

$$K_1=F(D_a, ToS) \quad (1)$$

For n = 2,

$$K_2=F(D_1, K_1) \quad (2)$$

And n = 3,

$$K_3=F(D_2, K_2) \quad (3)$$

Substituting from Eq. 2 in 3 we get,

$$K_3=F(D_2, F(D_1, K_1)) \quad (4)$$

Thus, it can be generalized for any n^{th} round of encryption, where n>1 the key K_n is given by,

$$K_n=F(D_{n-1}, F(D_{n-2}\ldots..F(D_1,K_1)\ldots)) \quad (5)$$

However, one should note that, the size of K_n increases with the value of the ToS parameter. Suppose in case of a critical service, the ToS will be higher and so the increased key size enhances the security for the service.

The exact details of the algorithm are left to the discretion of the implementers, but AES will be more suitable as it has support for varying key sizes.

The decryption mechanism at the SP's side also is of a similar manner.

- The SP gets to know the service through the ticket, which he receives from the Client. So, the SP knows the ToS parameter.
- Using ToS and Da (which is the start key), the SP generates K1.
- K1 now decrypts D1 message.
- K1 and D1 are again used to find K2 which decrypts D2 and so on. This ensures data integrity (Jutla, 2001), as previous plaintext need to be correct in order to get the subsequent ones.

This idea of encrypting the data using data is taken from Mihir *et al.* (1999). The actual mechanism of circles A and F are left to the discretion of the implementers. Care should be taken to see that A and F are not similar to each other, in which case, predicting the key from the cipher text, may become possible. For instance, A can be any encryption algorithm and F can be a one way hash function.

TRUST MANAGEMENT

As already pointed out, the clients send the feedback or the Trust value (Giorgos *et al.*, 2000; Li *et al.*, 2002; Blaze *et al.*, 1999; Selcuk *et al.*, 2004). To the broker after the transaction with the service provider, based on the quality of the service he received. This Trust Value is of great importance since it is used in the selection of service providers for the forthcoming requests for the same service type.

In the proposed mechanism, the broker maintains a list of service providers, with their corresponding points.

The entry of a service provider is updated in the list every time he services a client.

The steps involved are:

The client sends the Trust value (T) to the broker using the shared key Dx. The Trust value ranges from -2 (very bad) to +2(very good). The trust value of 0 represents a nominal case. So, the client decides one, among these five values. Based on his satisfaction. Also, he sends the Cost of the transaction (C) with this message.

The broker calculates the trust points by the formula.

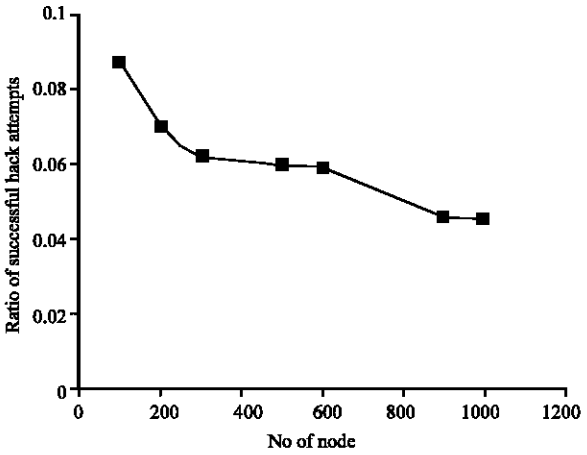

Fig. 5: Security improvement-trust management

$$TP = (C \neq T)/LF$$

Where LF is called the Limiting Factor. It can denote the No of clients in the Domain or any other value that is found appropriate.

- The broker then adds this trust point to the corresponding service provider's points in the list.
- The next time a similar service request comes, the sp. with highest points for that service type, is recommended by the broker.

PERFORMANCE EVALUATION

The proposed model was dealt with deep insight at the Grid Computing Labs of MIT, Anna University, to study its performance. It was implemented using ns-2.26 and graphs were plotted for the security in trust management (Fig. 5). The results and graphs are summarized.

RESULTS AND DISCUSSION

The scheme was compared with the existing cryptosystem standards and the number of keys and steps involved were compared and are tabulated (Table 1).

Let 'n' be the total number of nodes in the domain (no of sp. + no of clients + no of brokers).

The values in the table can be easily understood, if one can understand the symmetric and asymmetric paradigms. To have secure communication between n nodes, the number of keys that should be used are tabulated above. The table clearly shows a decrease in the number of keys involved in our mechanism (<<n). So, the no of keys involved is less on our mechanism.Similarly, the number of messages, before the actual transfer of data, for key agreement is also mentioned in Table 1. This also shows that our model is effective.

Table 1: Comparing symmetric, asymmetric and our proposed model

Factors	Symmetric key paradigm (DH)	Asymmetric key paradigm	Proposed scheme
No. of keys involved	n(n-1)/2	2n	SP+Active clients
No. of steps for key agreement	3	4	4

It is evident from the graph that, as the number of nodes increases, the ratio of successful hack attempts of the trust declines.

The randomness of the rotating key mechanism or the inter-dependence between the keys is another issue. It can be clearly understood from (Daniel, 2005) that CBC-MAC is unpredictable. So, the randomness in key generation process can safely be assumed.

ADVANTAGES

The dictionary attacks are given a tough job since the keys are used for every message and not the entire data. Increased data storage to store all these cipher-texts discourages eavesdroppers. Service-oriented adaptive security. Both the clients and the sp. are authenticated (sp. by the broker and client, through the tickets). The mechanism supports the four features of the cryptographic algorithms viz. confidentiality, Non-repudiation, message integrity and privacy.

However, this model is subjected to the usual constraints like single point of failure and computational complexity. But these get overshadowed by the more evident merits of the mechanism.

CONCLUSION

In this study, the novel idea of adaptive security, for flexibility in security services, has been presented. This model can be adapted to various real time applications, where broker architecture is involved, with some minor modifications.

ACKNOWLEDGEMENT

We are grateful to Mrs ThamaraiSelvi, Incharge of the Grid Computing Labs of Anna University, for helping us to check the practicality of the model in the GC labs and evaluate the effectiveness of the same.

REFERENCES

Abadi, M., 2002. Reconciling Two Views of Cryptography (The Computational Soundness of Formal Encryption), Journal of Cryptology, Springer.

Agrawal, R., R.L. Cruz, C. Okino and R. Rajan, 1998. A Framework for Adaptive Service Guarantees, Proceedings of Allerton Conference on Communication Control and Computers, Monticello, fiji.jpl.nasa.gov.

Agostini, A., C. Bettini and N. Cesa-Bianchi *et al*., 2004. Towards Highly Adaptive Services for Mobile Computing, Proceedings of IFIP TC8, webmind.dico.unimi.it.

Bellare, M., A. Boldyreva and S. Micali, 2000. Public-key Encryption in a Multi-User Setting: Security Proofs and Improvements, Advances in Cryptology-Eurocrypt, Springer.

Blaze, M., J. Feigenbaum, J. Ioannidis and A. Keromytis, 1999. The role of trust management in distributed systems security, Secure Internet Programming-cse. buffalo.edu.

Clifford Neuman, B. and Theodore Ts'o, Kerberos, 1994. :An Authentication Service for Computer Networks, IEEE Commun., 32: 33-38.

Farag Azzedin and Muthucumaru Maheswaran A trust Brokering System and Its Application to Resource Mangement in Public-Resource Grids.

Farag Azzedin and Muthucumaru Maheswaran, 2002. Evolving and Managing Trust in Grid Computing Systems, Proceedings of the 2002 IEEE Canadian Conference on Electrical Computer Engineering.

Giorgos Zacharia, 2000. Pattie Maes, Trust Management Through Reputation Mechanisms, Applied Artificial Intelligence, 1: 881-907.

Harry Chen, 2003. An Intelligent Broker Architecture for Context-Aware Systems, A PhD. Dissertation Proposal in Computer Science at the University of Maryland Baltimore County.

Ian Foster, Carl Kesselman, Gene Tsudik and Steven Tuecke, 2003. A security architecture for computational grids, Proceedings of the 5th ACM conference on Computer and communications security, San Francisco, California, United States, pp: 83-92.

Jutla, C., 2001. Encryption modes with almost free message integrity, Advances in Cryptology-EUROCRYPT, Springer.

Li, N., J.C. Mitchell and W.H. Winsborough, 2002. Design of a role-based trust-management framework, IEEE Symposium on Security and Privacy, ieeexplore. IEEE. org.

Mihir Bellare, Joe Kilian and Phillip Rogaway, 1999. The Security of the Cipher Block Chaining Message Authentication Code, In Journal of Compter and System Sciences.

Selcuk, A.A., E. Uzun and M.R. Pariente, 2004. A reputation-based trust management system for P2P networks, IEEE Cluster Computing and the Grid, ieeexplore.ieee.org.

Varalakshmi, P., S. Thamarai Selvi and K. Narmada, 2005. A Broker Architecture Framework for Reputation based Trust Management in Grids, ICIS, ISBN., 81-7764-936-1.

Xynogalas, S.A. MK Chantzara and IC Sygkouna *et al*. 2004. Context Management for the Provision of Adaptive Services to Roaming Users, Wireless Communications, IEEE, ieeexplore. IEEE. org.