ABAC Implementation Based on Key Pre-Distribution Scheme

S V Belim\textsuperscript{1,2} and S Yu Belim\textsuperscript{1}

\textsuperscript{1}Omsk State Technical University, 11, Mira ave., Omsk, 644050, Russia
\textsuperscript{2}Siberian State Automobile and Highway University, Omsk, Russia

E-mail: sbelim@mail.ru

Abstract. In the article, the KDP key pre-distribution scheme is modified for attribute access control. A scheme for implementing attribute-based access control on symmetric encryption is proposed. There are three participants in the scheme: a key distribution server, a subject, and a storage server. The key distribution server sends key content over secure channels when the system is initialized. The key materials content is determined by the attributes of the object and the subject. The storage server calculates the object’s encryption key based on the object’s key materials and attributes. The object is stored encrypted. If the subject has the necessary attributes, it can calculate the encryption key and access to object. If the subject does not have a sufficient set of attributes, the key cannot be computed and access is denied.

1. Introduction

Key pre-distribution schemes are used in distributed computer systems with a large number of subscribers. These schemes have become widespread in cloud systems. The primary purpose of key pre-distribution schemes is to reduce the key information size to be stored. Key pre-distribution schemes allow symmetric encryption keys to be calculated. Well-known key pre-distribution schemes Blom’s [1] and KDP [2] allow each subscriber to interact with each other. However, access control requirements are often present in distributed systems. The access control subsystem in such systems requires modification the pre-key distribution schemes. Modifications to the Blom and KDP schemes have been proposed for discretionary [3,4,5] and mandatory [6,7] security policies. This article proposes a modification the KDP-scheme of key pre-distribution, taking into account the attribute security policy.

KDP-scheme is based on a secure set of key materials $K=\{k_1,\ldots,k_n\}$. Key content is distributed to all users via secure channels. Key materials are distributed during system initialization. Open key materials is a subsets’ system $S=\{S_1,\ldots,S_m\}$ of set $\{1,\ldots,n\}$. $m$ is the number of users of the system. If the user $u_i$ wants to contact the user $u_j$, it extracts $S_i$ and $S_j$ from the set $S$. User $u_i$ calculates the intersection of sets $S_{ij}=S_i\cap S_j$. A common exchange key is generated based on elements of key materials set $K$ whose numbers fall into $S_{ij}$.

$$k_{ij} = \oplus k_l \quad (l \in S_{ij})$$

User $u_i$ performs the same operations. The traditional KDP scheme is built on the basis of Schpener families. The Schpener family [8] is a family of subsets $D=\{D_1,\ldots,D_n\}$ such that, if $D_i \cap D_j \subseteq D_t$, either $t = i$ or $t = j$. Set family $S$ is built on the basis of Schpener family $D$ [9]. Members of the Schpener family $D_i$ are used as intersections of subsets $S_{ij}$.
Access control systems are designed to prevent unauthorized access. Access control systems are based on one of the security models. Recently, the attribute-based access control model has been actively used in information systems. (ABAC) [10,11]. This access control model takes into account the properties (attributes) of subjects, objects, and environments. Access decisions are made by comparing attributes.

ABAC has some advantages over other security models. ABAC provides a high level of flexibility based on the use the arbitrary number of attributes. ABAC enables complex access control rules to be implemented using a simple set of rules. ABAC enables dynamic and efficient decision-making when environment parameters change. ABAC allows you to overcome some problems of the RBAC environment. [12]

The basic ideas of attribute-based access control (ABAC) are similar to role-based access control (RBAC). However, ABAC has some advantages over RBAC. The main advantage is the ability to manage based on dynamic attributes. The basic idea of ABAC is to grant or deny access requests based on user attributes and object attributes [8, 9]. This approach allows to generate a large number of access control rules [13] compared to RBAC. This advantage has made ABAC common in various applications for businesses, academia, and other organizations. There are two standards for ABAC infrastructure: Extensible Markup Language (XACML) and Next Generation AC (NGAC) [14]. In ABAC, subjects have the ability to access objects without specifying individual relationships between each subject and each object. Модель ABAC has some limitations [15]. The main limitation is due to the access decision time. Several attempts have been made to optimize ABAC. XEngine [16,17] numbers text rules and converts them to normalized numerical rules. This approach improves query processing efficiency. Tree-based data structure proposed in paper [18]. It allows binary search to be used to process requests in ABAC. XML database [19] accelerates policy analysis, but does not have thin policy control. Tiered caching mechanism based on statistical analysis [20] reorders security policy and improves its efficiency. Adaptive reordering of ABAC rules [21] allows ranking of high priority rules and improves decision-making efficiency. This paper proposes a scheme based on key pre-distribution that significantly reduces the number of calculations.

2. Key Distribution scheme
We are considering a basic version of the attribute security policy. $A$ is a set of attributes in the system.

$$A = \{a_1, a_2, ..., a_N\}.$$ Attributes in set $A$ are ordered. Attributes are properties of the objects and subjects in the system. Attributes of object $O$ denote $O.a$, attributes of subject $S$ denote $S.a$. The access rule is based on the attribute relationship of the subject and object. Subject $S$ has the right to access object $O$ if the relationship $S.a \sqsupseteq O.a$ is executed. Such security policy does not impose restrictions on the attribute values of subjects and objects.

We implement this security policy with a modified KDP-scheme of key pre-distribution. We form open and closed key materials. The calculations are performed in ring $\mathbb{Z}_p$. We map each attribute $a_i$ to $k_i \in \mathbb{Z}_p$.

$$K = \{k_1, k_2, ..., k_N\}.$$ A set of key materials $K$ are stored in secret on the key distribution server. The objects are stored encrypted on the data server. The encryption key $K(O)$ for the object $O$ is calculated based on the key materials corresponding to the attributes of the object.

$$K(O) = \bigoplus_{a_i \in O.a} k_i \oplus H(Id).$$
O.id is object’s ID. \( H() \) is a function that returns a binary string of the same length as the key material element length. \( \oplus \) - Bits operation XOR. This key is unique to each object. The value of the key depends on the object’s attributes. The storage server calculates the key value and encrypts the object. The storage server requests key materials from the key distribution server based on the object’s attributes. Key materials are transmitted over a secure channel. Upon receiving the materials, the data server calculates the key and encrypts the object. In the future, the data server provides access to objects and information about object attributes at the request from subjects.

Subject attributes show information about its access rights. Subject requests key materials from the key distribution server. The key distribution server checks the attributes of the subject and provides it with only those key materials that match its attributes. Key materials are transmitted over a secure channel. Subject \( S \) keeps a set of key materials \( K(S) \) in secret.

\[
K(S) = \{ k_i \mid a_i \in S.a \}
\]

If subject \( S \) wants to access the object \( O \), then sends a request to the storage server. The query contains the object ID and the subject attributes. If subject’s attributes \( S.a \) include attributes of the object \( O.a \), then storage server grants access to the object. Upon accessing the object, the subject calculates the encryption key. The subject owns all the necessary data to calculate the key. The subject can then decrypt the object. The storage server does not verify the truth of the data presented by the subject. If the subject provided false information, it cannot read the object because it cannot calculate the encryption key.

The interaction diagram of the system participants is shown in Figure 1.

**Figure 1.** System participants interaction diagram.

The diagram contains symbols for the main stages of the system operation.
1) Storage server queries key content for object \( O \).
2) The key distribution server sends key content over a secure channel.
3) Storage server encrypts object \( O \).
4) Subject request key material.
5) The key distribution server sends key materials to the subject via a secure channel.
6) Subject sends a request to access the object to the storage server.
7) The storage server sends the access link to the entity.
8) The subject accesses to the object.

We record the formal protocol for interaction between the participants in the scheme. We enter designations.
\( SK \) – Server of keys distribution.
SO – Data storage server.
S – Subject.
S.a – Subject’s attribute.
O – Object.
O.a – Object’s attribute.
O.id – Object’s identifier.
O’ – Encrypted object.
E() – Symmetric encryption.
D() – Symmetric decryption.
K={k_i} – Key materials.
⊕ - Bits operation XOR.

There are two stages in the operation of the system. The system initializes in the first step. System initialization includes the distribution of key materials. The subject accesses the object in the second step. The access step includes verifying access rights.

1. The storage server is initialized on the system.
1) The storage server reports the object attributes list to the key distribution server.
SO → SK: O.a.

2) The key distribution server returns the key materials set to the storage server over a secure channel.
SK → SO: {k_i | a_i ∈ O.a}.

3) The storage server calculates the encryption key for the object based on its attributes.
SO:
\[ k = \left( \bigoplus_{a_i \in O.a} k_i \right) \oplus H(O.id). \]

4) The storage server encrypts the file.
SO: O’=E_k(O).

2. The subject is initialized in the system.
1) The subject reports its attribute set to the key distribution server.
S → SK: S.a.

2) The key distribution server returns the key materials set to the subject over a secure channel.
SK → S: {k_i | a_i ∈ S.a}.

3) The subject requests access to the object.
1) The subject sends a request to the data server to access the object.
S → SO: O.id, S.a.

2) The storage server compares the attributes set of the subject and the object.
SO:
\[ R = \left( O.a \subseteq S.a \right). \]

3) The storage server returns a denial of access when the subject's attribute set is insufficient.
if R=FALSE then Denial.
4) The storage server provides access to the object with a sufficient set of subject attributes.
if R=TRUE then SO → S: O.a, O’.

5) The subject calculates the encryption key based on the attributes of the object.
S:
\[ k = \left( \bigoplus_{a_i \in O.a} k_i \right) \oplus H(O.id). \]

6) The subject decrypts the object and accesses it.
S:
\[ k = \left( \bigoplus_{a_i \in O.a} k_i \right) \oplus H(O.id), O=D_k(O’). \]

The server works with only one object in the presented protocol. The data server stores a large number of objects in real systems. The data server requests key materials for all objects. The set of server attributes is equal to combining the set of attributes for individual objects.
\[ SO.a = \bigcup_{O \in SO} O.a. \]
The storage server can be considered a single object in the initialization step. Key materials received from the key distribution server are distributed among individual objects at the encryption stage.

This protocol does not address subject and storage server authentication. Authentication can be implemented by any of the known protocols. Selecting an authentication protocol will not affect the reliability of the system.

3. Conclusion

We compare the proposed scheme with other ABAC implementations. ABAC can be implemented based on the access control service. In this case, each access request must activate the security subsystem and compare the attributes of the subject and object. The security subsystem must decide whether to allow or deny access based on the result of the attribute comparison. With this implementation, there is an additional load on the system. This load can be critical in distributed systems. In distributed systems, a single access control subsystem is not feasible. It requires a large number of the messages and leads to an unacceptable increase in the processing time of requests.

In the present system, a single key distribution server for all participants operates only at the system initialization stage. Permission or denial of access occurs automatically because the ability or inability to decrypt the object. As a disadvantage of the proposed system, it is possible to indicate the need to implement secure channels for transmission key materials. However, secure channels are also needed in a centralized access control system to transmit access authorization overhead messages. This protection is needed to prevent attacks on availability. The creation a secure channel is necessary at each access request, not only at system initialization.

Attribute encryption is used as an alternative to a single service in modern systems [22-27]. In these algorithms, the encryption key is also generated based on the attributes of the subject and object. But attribute encryption systems use public-key cryptography based on the power-to-power operation in the final ring. Such algorithms are labor-intensive and slow to execute. The proposed scheme uses symmetric cryptography. Symmetric encryption algorithms work significantly faster than asymmetric ones.

The proposed scheme has flexibility. The set of attributes can change dynamically while the system is running. The key distribution server registers new attributes. Key materials are generated for the new attributes. The storage server requests key materials for new attributes if necessary. If the object attributes change within a set known to the server, the old key materials are used.

Thus, the ABAC implementation scheme proposed in this article has advantages over other implementations.

References

[1] Blom R 1985 *Lecture Notes in Computer Science* **209** 335
[2] Mitchell C J and Piper C 1988 *Discrete and Applied Math.* **21** 215
[3] Belim S V and Belim S Yu 2016 *Automatic Control and Computer Sciences* **50** 8 773
[4] Belim S V and Belim S Yu 2019 *Journal of Physics: Conf. Series* **1210** 012009
[5] Belim S V and Belim S Yu 2018 *Automatic Control and Computer Sciences* **52** 8 1124
[6] Belim S V and Belim S Yu 2018 *Automatic Control and Computer Sciences* **52** 8 1134
[7] Belim S V and Belim S Yu 2019 *Journal of Physics: Conf. Series* **1210** 012008.
[8] Dyer M, Fenner T, Frieze A, Thomason A 1995 *J. Cryptology* **8** 189
[9] O’Keefe C M 1993 *Australas. J. Combin.*, 7 195
[10] Hu V C, Ferraiolo D, Kuhn R et al. 2013 *NIST special publication* **800** 162
[11] Yuan E and Tong J 2005 *IEEE International Conference on Web Services (ICWS’05)*
[12] Sandhu R, Coyne E, Feinstein H and Youman C 1996 *IEEE Computer* **29** 2 38
[13] Hu, V C, Kuhn V and Ferraiolo D F 2018 *Attribute-Based Access Control* Norwood: Artech Hous
[14] Kuhn D R, Coyne E J and Weil T R 2010 *Computer* **43**(6) 79
[15] Crampton J 2003 *Proceedings of the 10th ACM conference on Computer and communications security*
[16] Liu A X, Chen F, Hwang J. Xie T 2008 *SIGMETRICS 2008*
[17] Liu A X, Chen F, Hwang, J. Xie T 2011 *IEEE Trans. Comput.* **60** 1802
[18] Ros S P, Lischka M, Marmol F G 2011 *Proceedings of the 17th ACM*
[19] Wang Y Z, Feng D G, Zhang L W, Zhang M *J. Softw.* **22** 323.
[20] Niu D H, Ma J F, Ma Z, Li C N, Wang L 2014 *J. Commun.* **35** 206
[21] Qi Y, Chen J, Li Q M 2016 *Comput. Sci.* **43** 163
[22] Sandhu R, Ferraiolo D and Kuhn R 2000 ACM workshop on Role-based access control.
[23] De Capitani di Vimercati, Paraboschi S S and Samarati P 2003 *Software: Practice and Experience* **33**(5) 397.
[24] Ali M, Mohajeri J, Sadeghi M R and Liu X M 2020 *Theoretical Computer Science* **815** 25
[25] Cui H, Deng R H, Qin B D and Weng J 2020 *Information Science* **517** 217
[26] Rana S and Mishra D 2020 Journal of Medical System 2020 **44**(5) 97
[27] Sumathi M and Sangeetha S 2020 *International Journal of Cloud Applications and Computing* **10** 2 77