Secure Secret-Key Management of Kerberos Service

Lai-Cheng Cao*

School of Computer and Communication
Lanzhou University of Technology
Lanzhou 730050, China
caolch@lut.cn

Abstract. The Kerberos protocol has promoted the development of new techniques to support various kinds of distributed applications. However, the secret-key management is security core in the whole system. Using symmetric encryption algorithm Rijndael of AES (Advanced Encryption Standard), all secret-keys of the client were encrypted by the secret-key of the authentication server and stored in the database. The secret-key of the authentication server was protected by distributing its shares to the router, Ticket-granting Server (TGS) and the Web server. The authentication server did not store its secret-key in system, when the system needed this secret-key, the authentication server could synthesize it by distributed shares. Security analysis shows that this secret-key management has fault-tolerant and no-information leakage; it also defends collusive attack and cracking the secret-key attack.

Keywords: The secret-key, Kerberos protocol, Rijndael encryption algorithm, fault-tolerant.

1 Introduction

The Kerberos protocol developed by MIT has been widely used in the security authentication service for users and network connection. It bases on symmetric encryption algorithm, the communication uses share-secret-key, and these share-secret-keys of clients and server are stored into the database. Once the intruder unauthorized accesses the database, all secret-keys can be leaked, there is no security to speak of the whole system. The paper [1] extends an ideal functionality for symmetric and public-key encryption proposed in previous work by a mechanism for key derivation. Ke Jia etc [2] propose the public key encryption algorithm based on braid groups to improve Kerberos protocol. Liu ke-long etc [3] present an improved way using Yaksha security system of ElGamal Public Key algorithm based on the original protocol framework to overcome the Kerberos' limitations. All these methods have extended public key algorithm based on symmetric encryption algorithm, it makes key management more complex and system speed slower. In this paper, we put

* This work is supported by the National Natural Science Foundation of China under Grant No. 60972078; the Gansu Provincial Natural Science Foundation of China under Grant No. 0916RJZA015.

H. Deng et al. (Eds.): AICI 2011, CCIS 237, pp. 76–83, 2011.

forward scheme to protect the secret-keys of the client's users based on symmetric encryption algorithm Rijndael of AES, all secret-keys except the identity (ID) of users are encrypted by the secret-key of the authentication server and stored in the database. The secret-key of the authentication server is protected by distributing its shares to the router (or the gateway), Ticket-granting Server (TGS) and the Web server.

The remainder of this work is organized as follows. In section 2 we describe the Kerberos protocol. In Section 3 we present a scheme to protect the secret-key of the users. In section 4 point out a scheme to protect the secret-key of the authentication server. In section 5 we present the correctness proof of this scheme. In section 6 we finish security analysis about our method. The conclusion is presented in section 7.

2 The Kerberos Protocol

In the distributed environment, in order to protect user information and server resources, we require that client systems authenticate themselves to the server, but trust the client system concerning the identity of its user. We describe the Kerberos protocol [4-6] (as shown in Fig. 1); it includes the gateway, Authentication Server (*AS*), Ticket-granting Server (*TGS*), web server. Authentication Server keeps a database containing the secret-keys of the users and the router and TGS. The client and the Server can communicate each other based on TCP/IP protocol.

- *Client*: Requires gaining access to Web server in the distributed environment.
- *AS*: Authenticates servers to the client.
- *TGS*: Grants service-granting ticket to the client.
- *Web server*: Stores resource and data for the web users.

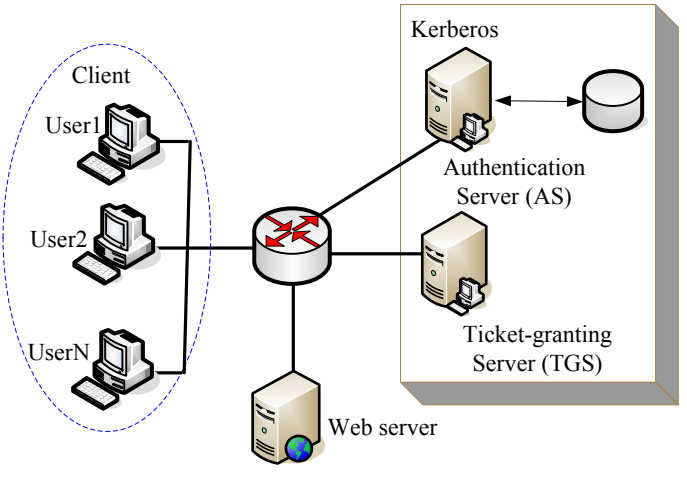

Fig. 1. The architecture about the Kerberos protocol

If the client requests to access Web server, this server will authenticate the client. Similarly, if Web server requests to access client, this client authenticates web server.

3 Protecting the Secret-Key of the Users

All secret-keys of the users, the secret-keys of TGS and the router are encrypted by the secret-key of the authentication server based on AES encryption algorithm. Unencrypted users' ID and their encrypted secret-key, unencrypted TGSs' ID and its encrypted secret-key, unencrypted router's ID and its encrypted secret-key are stored in the database; this database table is showed in Table 1 as follows.

Table 1. Encrypted secret-key OF database table

Primary key	Encrypted secret-key	Index
TGS_id	**********	*1*
Router_id	**********	*2*
User1_id	**********	*3*
User2_id	**********	*4*
……	……	…
UserN_id	**********	*N+2*

When the system needs a secret-key, first it searches the ID of this secret-key in this database table, and then decrypt this encrypted secret-key with the secret-key of the authentication server. Of course, the security of the secret-key of the authentication server is a key and core of security of entire system, thus, we put forward that the shares of the secret-key of the authentication server are distributed to store in the router (or the gateway), Ticket-granting Server (TGS) and the Web server, and the authentication server does not save its secret-key, when the authentication server needs this secret-key, it can be synthesized by these shares, and the authentication server throws away it after it is used.

4 Protecting Secret-Key of the Authentication Server

4.1 Generating Pre-datum of the Shares

Referring to the paper [7-8], we divide the secret-key of the authentication server into three shares, they are stored in four share servers, these four share servers are the authentication server, the router, Ticket-granting Server (TGS) and the Web server. The secret-key of the authentication server can be synthesized by three shares of three servers which are selected from four servers, the shares are designed for two groups to increase fault tolerance. The authentication server produces pre-datum of the share before these shares are generated. Table 2 shows these pre-datum which are belong to four share servers, d_{11} and d_{21} are pre-datum which shall generate the shares of the authentication server, and they each belong to group 1 and group 2, d_{12} and d_{21} are pre-datum which shall generate the shares of the router, and they each belong to group

1 and group 2, d_{13} and d_{22} are pre-datum which shall generate the shares of TGS, and they each belong to group 1 and group 2, d_{13} and d_{23} are pre-datum which shall generate the shares of the Web server, and they each belong to group 1 and group 2. These pre-datum can compound: $d=d_{11}+ d_{12}+ d_{13} = d_{21}+ d_{22}+ d_{23}$.

Table 2. The pre-datum of the share belong to share server

Group number	Authentication server	Router	TGS	Web server
Group 1	d_{11}	d_{12}	d_{13}	d_{13}
Group 2	d_{21}	d_{21}	d_{22}	d_{23}
	Compounding 1: $d=d_{11}+ d_{12}+ d_{13}$			
	Compounding 2: $d=d_{21}+ d_{22}+ d_{23}$			

4.2 The Method of Distributing Shares

Referring to the paper [9-10], the method of distributing shares is presented the follow.

1) Choosing a polynomial:

$$f(x)=\sum_{k=0}^{3} a_k x^k \tag{1}$$

2) Generating 4 random numbers x_i (i=0,1,...,3), and computing:

$$f(x_i)=\sum_{k=0}^{3} a_k x_i^k$$

3) When *i=0*, taking $d_0 = f(x_0)$ as the pre-datum of synthesizing share of the authentication server.

4) When *i=1,2,3*, using LaGrange formula to computer:

$$f(x)=\sum_{i=1}^{3}\left(f(x_i)\sum_{j=1,3}^{j\neq i}\frac{x-x_j}{x_i-x_j}\right) \tag{2}$$

$$\text{Let } x=0:\ f(0)=\sum_{i=1}^{3}\left(f(x_i)\sum_{j=1,3}^{j\neq i}\frac{-x_j}{x_i-x_j}\right) \tag{3}$$

5) Taking m (m=1,2) group pre-datum of the shares:

$$d_{mi} = f(x_i)\sum_{j=1,3}^{j\neq i}\frac{-x_j}{x_i-x_j} \quad (i=1,2,3) \tag{4}$$

6) Computing pre-datum d of the shares:

$$d=d_0+\sum_{i=1}^{3} d_{mi} \tag{5}$$

7) Then the authentication server distributes pre-datum d_{mi} ($m=1,2$; $i=1,2,3$) of the shares to the share severs.

4.3 Generating the Shares of Share Servers

We adopt the computational infeasibility of discrete logarithms to generate the shares of share servers. Table 3 shows the group of these shares S_{mi} ($m=1, 2$; $i=1, 2, 3$). When any one share server loses its shares, we can adopt the remaining three share servers to synthesize the secret-key, and the system can continuously work. When any two share severs are attacked, any information of the secret-key can not be leaked.

Table 3. The key share of 3 server-from-4 server

Group number	Authentication server	Router	TGS	Web server
Group 1	S_{11}	S_{12}	S_{13}	S_{13}
Group 2	S_{21}	S_{21}	S_{22}	S_{23}

The whole system chooses two big prime number p and q.

1) The authentication server selects an integer $x_k \in [1, p-1]$ and computes:

$$y_k = p^{x_k} \bmod q \tag{6}$$

2) The authentication server broadcasts yk to the share severs.

3) The share severs compute own shares:

$$S_{mi} = y_k^{d_{mi}} \bmod q \quad (m=1, 2;\ i=1, 2, 3) \tag{7}$$

4.4 Synthesizing the Secret-Key of the Authentication Server

When the authentication server needs its secret-key, it selects one group shares whose numbers are m1, m2, m3 (m=1, 2) from three of the four servers, and then synthesize the secret-key as follows.

1) The authentication server computes:

$$S_0 = y_k^{d_0} \bmod q \tag{8}$$

2) When the authentication server receives the shares of share server, it synthesizes its secret-key: $\text{Secret-Key} = S_0 \prod_{i=1}^{3} S_{mi}$ (9)

The left figure in Fig. 2 shows that the authentication server uses the first group shares to synthesize the secret-key based on the authentication server, the router and TGS. The right figure in Fig. 2 shows same result based on the authentication server, the router and the Web server.

Where, $\text{Secret-Key} = S_0 \prod_{i=1}^{3} S_{1i}$

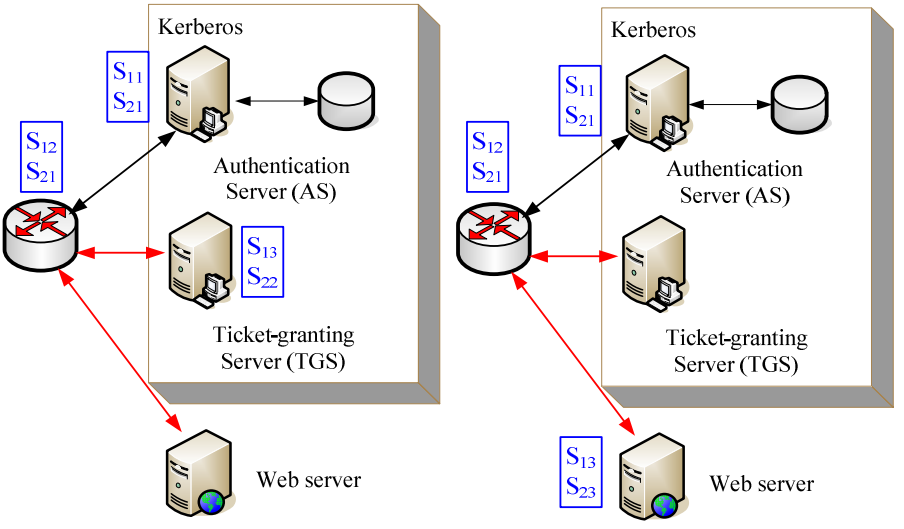

Fig. 2. Synthesizing the secret-key by first group shares

The left figure in Fig. 3 shows that the authentication server uses the second group shares to synthesize the secret-key based on the router, TGS and the Web server. The right figure in Fig. 3 shows same result based on the authentication server, TGS and the Web server. Where, $\text{Secret-Key} = S_0 \prod_{i=1}^{3} S_{2i}$

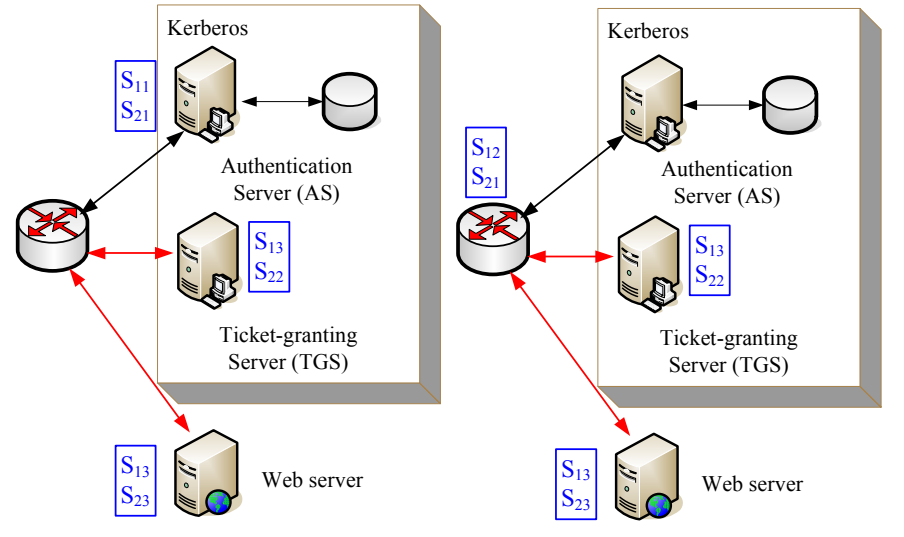

Fig. 3. Synthesizing the secret-key by the second group shares

5 Correctness Proof

Let d_{mi} and d_{li} ($m, l=1, 2; i=1, 2, 3$) as two group pre-datum, their shares are S_{mi} and S_{li}. K_{mi} and K_{li} are synthesized secret-key based on two group shares.

Because $d = d_0 + \sum_{i=1}^{3} d_{mi} = d_0 + \sum_{i=1}^{3} d_{li}$, thus

$$K_m = S_0 \prod_{i=1}^{3} S_{mi} = y_k^{d_0} \bmod q \prod_{i=1}^{3} y_k^{d_{mi}} \bmod q = y_k^{d_0 + \sum_{i=1}^{3} d_{mi}} \bmod q = y_k^{d} \bmod q$$

$$K_l = S_0 \prod_{i=1}^{3} S_{li} = y_k^{d_0} \bmod q \prod_{i=1}^{3} y_k^{d_{li}} \bmod q = y_k^{d_0 + \sum_{i=1}^{3} d_{li}} \bmod q = y_k^{d} \bmod q$$

Therefore, $K_m = K_l$.

6 Security Analysis

The security of this secret-key management of Kerberos service can be attributed to four factors:

1) Fault-tolerant

According to equation (7) and (9), when one share sever loses its shares, the system can continuously work.

2) No-information leakage

Because equation (6), (7) and (8) base on the computational infeasibility of discrete logarithms, only $y_k = p^{x_k} \bmod q$ and $S_{mi} = y_k^{d_{mi}} \bmod q$ can reflect the information of the secret-key, it is impossible that using y_k and S_{mi} to compute d_{mi} and x_k, namely, attacker can not obtain any information of d_{mi} and x_k from filched S_{mi} and y_k.

3) Defending collusive attack

If three ones of four share severs perform collusive attack, they can select one group shares whose share numbers are sequential after they are arrayed again, but d_0 and d_{mi} ($i = 1,2,3$) are random and irrelative, and conditional information entropy $H(d_0 \mid d_{mi}) = H(d_0)$, in addition, participating x_k in synthesizing the secret-key is randomly selected in [1, p-1], therefore, collusive attack can not finish.

4) Resisting cracking the secret-key attack

Rijndael encryption algorithm uses secure S-boxes as nonlinear components [9, 10]; it supports on-the-fly subkey computation for encrypting. The operations used by Rijndael are among the easiest to defend against power and timing attacks. The use of masking techniques to provide Rijndael with some defense against these attacks does not cause significant performance degradation. So it highly resists cracking the secret-key attack.

7 Conclusion

The secret-key management is security core in the Kerberos protocol; it is also a hot research field of information security. In this paper, we adopt encryption algorithm Rijndael of AES, distributing shares of the secret-key and synthesizing the secret-key, to protect effectively the secret-key. This technology makes the whole system to have fault-tolerant and no-information leakage, and also defends collusive attack and cracking the secret-key attack.

References

1. Küsters, R., Tuengerthal, M.: Ideal Key Derivation and Encryption in Simulation-Based Security. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 161–179. Springer, Heidelberg (2011)
2. Jia, K., Chen, X., Xu, G.: The improved public key encryption algorithm of Kerberos protocol based on braid groups. In: 2008 International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM 2008), vol. 1, pp. 1–4 (2008)
3. Liu, K.-l., Qing, S.-h., Yang, M.: An Improved Way on Kerberos Protocol Based on Public-Key Algorithms. Journal of Software 12(6), 872–877 (2001)
4. Lai-Cheng, C.: Enhancing distributed web security based on kerberos authentication service. In: Wang, F.L., Gong, Z., Luo, X., Lei, J. (eds.) Web Information Systems and Mining. LNCS, vol. 6318, pp. 171–178. Springer, Heidelberg (2010)
5. Rao, G.S.V.R.K.: Threats and security of Web services - a theoretical short study. In: Proceedings of IEEE International Symposium Communications and Information Technology, vol. 2(2), pp. 783–786 (2004)
6. Seixas, N., Fonseca, J., Vieira, M.: Looking at Web Security Vulnerabilities from the Programming Language Perspective: A Field Study. Software Reliability Engineering 1, 129–135 (2009)
7. Wu, T., Malkin, M., Boneh, D.: Building intrusion-tolerant applications. In: Information Survivability Conference and Exposition, pp. 25–27. IEEE Computer Society, Los Alamitos (2000)
8. Zhang, X.-f., Liu, J.-d.: A threshold ECC Based on Intrusion Tolerance TTP Scheme. Computer Applications 24(2), 5–8 (2004)
9. Zhendong, S., Gary, W.: The essence of command injection attacks in web applications. ACM SIGPLAN Notices 41(1), 372–382 (2006)
10. Ashley, C., Wanlei, Z., Yang, X.: Protecting web services from DDOS attacks by SOTA. In: ICITA 2008, pp. 379–384 (2008)