

An Improved Kerberos Scheme Based on Dynamic Password

Wen Lei, Hai Cao, Xingjian Liang, Hong Zhang
School of Computer Science, Sichuan University of Science & Engineering, Zigong, Sichuan, China
suselwen@126.com

Abstract—By studying the Kerberos authentication scheme, an improved authentication scheme is raised, which is based on Dynamic Password Method. In the improved scheme, user's password can be effectively protected, and the authentication is double between users and servers. Also, the scheme can resist jacking connection attack. The improved scheme is more secure and more practical than the original one.

Index Terms—identity authentication; dynamic password; security; improvement

I. INTRODUCTION

When we communicate in a distributed network, there are some security threats:

- (1) If a user accesses a certain specific workstation, he can impersonate another user to operate the workstation.
- (2) A user can change the workstation network addresses to make its issued request seemingly come from a disguise workstation.
- (3) A user can wiretap the message exchange process, and use the replay attack to access the server or interrupt the communication.

For solving the above problems, Kerberos[1] was designed to implement the identity authentication when a user wants to access a workstation. Kerberos is a Network Authentication Protocol, which was created by MIT for Athena Project based on the deformation of symmetric Needham-Schroeder protocol, using symmetry encryption scheme. Its goal is to provide strong authentication services for client/server applications programs through the key systems. This process does not depend on the authentication of the host operating system, without the trust based on the host address, no requirement for the physical security of all the hosts in the networks and assuming the data carried in networks can be arbitrarily read, modified and inserted. As the trusted third party, Kerberos implements the authentication services through the traditional encryption technique. Its design thought influenced a lot of later identity authentication process, such as Krypto Knight and SE-SAME, and its successful application is a big boost for the development of the identity authentication protocol.

Since Kerberos was put forward, it has experienced 5

versions. Among them, versions 1-3 occurred only internally at MIT. When version 4 was designed, it has been extensively identified and used off MIT campus. With the spread of versions 4, some of its limitations and weaknesses were gradually discovered, for example, applicable network environment is limited, encryption process is redundant, etc. According to those feedbacks, MIT expanded and improved the version 4. Now, a more complete version 5 is formed[2]. Until January 2000, the latest version is krb5 -1.1[3].

However, Kerberos is combining with the environment of MIT, there are some limitations to extend it as a standard use in all kinds of existing computer communication network[4]. Aiming at the Kerberos security issues in actual network authentication applications, at home and abroad, many specialists have studied a lot and put forward some improved security measures[5-14]. However, some of these measures will change the deployment structure of Kerberos, some will make the system calculation significantly increase. In this paper, on the basis of the Kerberos main structure, an improved Kerberos scheme will be given out, which is based on the dynamic password techniques. This scheme can improve its security performance significantly with a small computational cost.

II. KERBEROS AUTHENTICATION PROGRAM

The authentication's aim is to confirm that the communication object is the original rather than an impostor. The Kerberos's fundamental can be illustrated as: a KDC (key distribution center) that consists of two logically separate parts: an Authentication Server (AS) and a Ticket Granting Server (TGS) is established to centrally preserve usernames and passwords, and is used to authenticate the users' identity and grant authorization to users. Any service ticket must be granted by the KDC, and then the servers that provide various kinds of service do not directly authenticate the users' identity and not grant any authorization, but provide corresponding services according to granted tickets. The communications between users, KDC and servers are encrypted with DES (data encryption standard).

The process of authentication is shown in Fig.1, which consists of three stages, including six steps. The process is stated in the following.

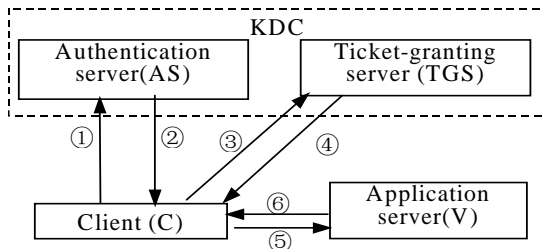


Figure 1. The process of Kerberos authentication

The symbols among the later are illustrated as follows:

- ID***: *** ID identity
- TS_x: Timestamp, x is one of 1,2,3,4 and 5
- K_{tgs}: The shared secret key between AS and TGS
- K_v: The shared secret key between TGS and V
- E_{xxx}[*]: Using the key xxx to encrypt *
- K_{c,v}: The session key shared by C and V
- K_{c,tgs}: The session key shared by C and TGS

Stage 1. User authentication and asking for the Ticket_{tgs} of TGS:

- ① C → AS : ID_c || ID_{tgs} || TS1 ;
- ② AS → C : ID_c || Ticket_{tgs} || E_{K_c} [K_{c,tgs} || TS2 || ID_{tgs}],

Notes: Ticket_{tgs} = E_{K_{tgs}} [K_{c,tgs} || ID_c || TS2 || ID_{tgs}] ;

Firstly, the user asks for a TGT (Ticket Granting Ticket) at a client. When logging, the user will be asked to enter his username, then the client will send a cleartext message to the AS requesting services, which includes the user ID and TGS ID.

Secondly, the AS grants a TGT to user. After receiving the user's request, the AS will check the user's ID. If the user is valid, the AS will generate a Session Key K_{c,tgs} for protecting the communication between Client and TGS, and create a TGT Ticket_{tgs}, then encrypt them by K_c and send to Client. In particular, the Ticket_{tgs} includes the client ID, the TGS ID, client network address, the current time, ticket validity period, and the client/TGS session key; the K_c is derived from the user's password, and only can be known by the user and the AS.

Stage 2. Asking for the credential Ticket_v for accessing application server:

- ③ C → TGS: ID_v || Ticket_{tgs} || Auth_{c,tgs} ,
- Notes: Auth_{c,tgs} = E_{K_{c,tgs}} [ID_c || TS3];
- ④ TGS → C: ID_c || Ticket_v || E_{K_{c,tgs}} [K_{c,v} || TS4 || ID_{tgs}],

Notes: Ticket_v = E_{K_v} [K_{c,v} || ID_c || ID_v || TS4];

Thirdly, asking for the credential Ticket_v. After receiving the message from TGS, the user is asked to provide his password. The client changes the password to K_c, and decrypts the message to obtain the Ticket_{tgs} and K_{c,tgs}, and saves them for later communication. For security, the password and K_c will be removed. If the login time is out of the ticket validity period, the user is asked for a new Ticket_{tgs} request. Certainly, user can check the current status of his tokens.

A ticket is only corresponding to a specific service, thus the user needs to separately ask for the granting server tickets (Ticket_v) for each application server (V), the

Ticket_v can be obtained from TGS. When asking for a Ticket_v, the client must send a request message to TGS, which includes V's ID, the Ticket_{tgs} and the encrypted authenticator (Auth_{c,tgs}) by using K_{c,tgs}.

Fourthly, TGS grants Ticket_v to user. Upon receiving the user's request message, the TGS decrypts Ticket_{tgs} using its own secret key to get K_{c,tgs}. Then using K_{c,tgs}, the TGS decrypts message Auth_{c,tgs}, thus it can confirm the user through the decrypted message. If the user is valid, TGS will generate a session key K_{c,v} for the communication between C and V, then create a ticket Ticket_v, which includes C's ID, V's ID, network address, the current time, Ticket_v validity period and K_{c,v}. The Ticket_v validity period is the shortest one between the Ticket_{tgs} remaining valid time and the service default valid time.

Then TGS encrypts Ticket_v using V's secret key and session key K_{c,v} using K_{c,tgs}, and sends them to C. C can decrypt the reply message by using K_{c,tgs} to obtain Ticket_v and K_{c,v}.

Stage 3. The stage of accessing the application server:

- ⑤ C → V: Ticket_v || Auth_{c,v} ,
- Notes: Auth_{c,v} = E_{K_{c,v}} [ID_c || TS5];
- ⑥ V → C: E_{K_{c,v}} [TS5].

Fifthly, user can access application server. This step is similar with the third step, only the server changes from TGS to V, so it is not discussed here.

Sixthly, service responding. Upon receiving messages including Ticket_v and Auth_{c,v}, sent by user, V decrypts them separately, and judges whether the requests is effective by comparing the username, network addresses, validity period and other information. The synchronized time must limit in a few minutes between C and V, if there is a big deviation between the time of request and the current system time, the request is decided as invalid. To prevent against replay attacks, V usually keeps a recently received effective requests list, if a receiving request has the same time with someone in the list, it is considered as an invalid request.

If C needs to verify the identity of V, V can add the received timestamp by 1, and encrypt it using K_{c,v}, then send to C. C can confirm V by decrypting and checking the message.

III. THE KERBEROS SECURITY ANALYSIS

Although Kerberos is widely used in the identity authentication, literature [15] also is analyzed by using BAN and proved its security. There are some weaknesses in Kerberos due to the limitation in design [4].

A. Easily to suffer guessing attack

Although it does not require the user to transfer plain text passwords, but Password Guessing Attacks are not solved by Kerberos. Though the user can obtain the sharing key with AS by entering his password on client, and avoid the password frequently being used, but after all, the password must be provided, this is not safe enough. From Kerberos authentication process perspective, the AS cannot verify user's identity, but only confirm him by encrypting the messages sent to users. In the second data packet, E_{K_c} [K_{c,tgs} || TS2 || ID_{tgs}], in

Kerberos Protocol, session key is encrypted by K_c , which is the key derived from user password, and it is not necessary to verify the authenticity of the user when the servers are answering. But it is assumed that only legitimate users have the password. In practice, however, many users do not attach importance to choosing a strong password; on the contrary, in order to make them easy to remember, they often choose simple passwords. If a user chooses a poor password, it is possible for an attacker to successfully mount an offline dictionary attack by repeatedly attempting to decrypt, with successive entries from a dictionary, messages obtained which are encrypted under a key derived from the user's password. Once the attacker intercepted a response, password attack is easily to be formed.

B. Lacking of mutual authentication between client and server

If Kerberos fail to realize interactive authentication between the user and the server, the authentication server is likely to be attacked by Denial of Service [11]. At the same time, the attacker can also deceive user by forging authentication server [8,11].

C. Easily suffer jacking connection attack

Once the users pass the authentication, as long as Ticket, tickets are in validity, they can have access to any services. By sharing encryption in Kerberos, once the attackers intercept a cipher text, and get to know the plaintext, they can easily get $K_{c,v}$, thus to steal server information by disguising as a legitimate user.

D. Time synchronization problem is a weakness

In Kerberos V5, time is often used, such as the validity period of tickets and timestamps in authenticator, etc. If it is difficult to keep the times consistent, which are on local-distributed hosts, the Kerberos authentication system is hard to keep normal operation. Although Kerberos can tolerate certain time deviation between each host (the maximum deviation is defined by system administrator), and take it into account in programming, as each host clock cannot well keep walking accurate, the deviation will increase. If there is a bigger deviation between the local-distributed hosts' physical clock, a certain time synchronization protocol is necessary to be used, however this not only increases the complexity of Kerberos authentication system, but also brings out new security problems (which are caused by the clock synchronization protocol). In addition, if a host time is changed, the host cannot use Kerberos protocol. Furthermore, if there is something wrong with the Kerberos server time, the whole authentication system will be broken down. And in fact, it is difficult to synchronize clock with each other in a distributed network with various terminals.

E. The cost is too great by using timestamp to prevent replay attack

A ticket with a long validity period is easily to suffer replay attack. Even it has short validity period, if an attacker is brilliant enough, the ticket still has the

opportunity to be replay attack, besides the attacker may damage the system clock synchronicity. In order to use timestamp to prevent the replay attack, the server must achieve the following jobs:

① Comparing the actual client IP with the IP in ticket to check their consistency. ② Saving all effective triples (C, V, TS). ③ Comparing each fresh received timestamp with the saved triples, If some are the same, the fresh is a replay attack. And also those outdated triples must be removed. ④ As there may be many service processes running at the same time, the above operations of triples must be held mutexes by a certain mechanism.

Thus, if this server is too busy, its running speed will be significantly reduced.

IV. THE IMPROVEMENT OF KERBEROS SCHEME BASED ON DYNAMIC PASSWORD

A. The idea of scheme design

In the original Kerberos system, the reasons caused the security problems mentioned as the above, mainly include the following: ① in the user authentication stage, user's identification is only on the basis of his static password, and there are not mutual authentication between users and AS; ② in the service access stage, the communications between client and application server are lack of authentication mechanism. So, based on the thought of dynamic password, the improved Kerberos scheme can be drawn up as Fig.2. During the user initial authentication, the user authentication is not simply based on his password, but a big random number (called as salt) is used together to generate a security password, thus can prevent the Password Guessing. And using exchanging random number between the client and the server, the mutual authentication can be achieved. In addition, in the communication period between the client and application server, a connection authentication factor can be used to prevent the replay attacks.

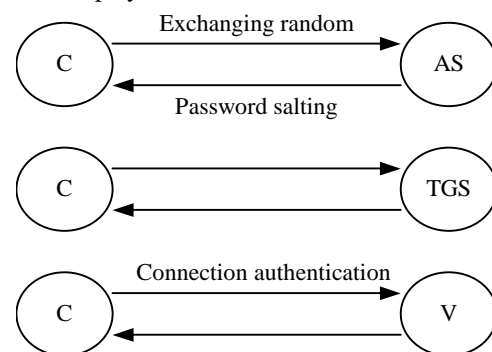


Figure 2. The Schematic of improved scheme

B. Brief of dynamic password

Dynamic Password (DP) is also called One Time Password (OTP), which is used to solve the traditional problems which appear when the static Password authentication cannot cope with eavesdropping and replaying, forging, guessing, etc.

By means of DP, uncertainties will be considered in authentication information during the process of lodging

to make authentication information different every time, which can improve the security of information in the process of lodging. This technology can effectively avoid replay attack, and solve the problems that the static password is likely to be stolen in transmission and database. Dynamic password authentication mainly realizes two schemes: S/KEY dynamic password authentication and asymmetric password authentication, which can better go against intercept replay attack. But S/KEY cannot withstand impersonating server attack and tracking attack, and at the same time there are some defects in safety, such as "stealing server information+speculation attack" and "jacking connection attack". In reference[17],an effective improved DP scheme was schemed out.

Taking the identity authentication process in communication between Alice and a Server as an example, the reference[17]'s dynamic password scheme can be stated in the following:

Some symbols among the later are illustrated as follows:

T_a, T_s : Alice and server's timestamps, should be as accurate as 1 millisecond;

PS : Alice's authentication evidence stored in server;

B : A large random number generated by Alice;

1) *The process of user registering :*

Alice \rightarrow Server: $E_{K_p}(PIN \parallel (PS \oplus B)) \parallel E_{K_p}(B)$

Step 1: Alice chooses her identity symbol PIN (Personal Identification Number) and password PW , selects a large random number B . The client computes $PS = g^{PIN \parallel PW \parallel B}$, and figures out $PS \oplus B$, then sends $E_{K_p}(PIN \parallel (PS \oplus B))$ and $E_{K_p}(B)$ to server to ask for registering (H refers to hash function; K_p is as the public key of sever; $E_{K_p}(\bullet)$ refers to encrypt the information in the parentheses with K_p);

Step 2: When receiving the registration information, the server uses its private key K_R to obtain $PIN, PS \oplus B$ and B to check the uniqueness of PIN . If registration is found, Alice is required to choose another PIN . Otherwise, the server calculates: $PS = (PS \oplus B) \oplus B$, and finally stores PIN, PS and B in safety database. So far, the registration process is over.

2) *The process of asking for authentication:*

① Alice \rightarrow Server: $PIN \parallel E_{K_p}(N_1)$

② Server \rightarrow Alice: $P \oplus g^{N_2} \parallel g^{N_1} \oplus B \parallel PS \oplus Ts \parallel H(N_1)$

③ Alice \rightarrow Server: $E_K\{Ts - Ta, g^{N_2}\}$

Notes: $K = (g^{N_2})^{PIN \parallel PW \parallel B}$

④ Server \rightarrow Alice: acknowledgement for connection request

Step 1: Alice inputs her PIN and PW , the client generates a random number N_1 and computes out g^{N_1} , then sends PIN and $E_{K_p}(N_1)$ to server to request certification and establish connection;

Step 2: The server queries her PIN in user database, if there is not, then terminates the authentication, else the

server computes g^{N_1} , queries the PS and B according the PIN , and generates a random number N_2 , then computes out g^{N_2} and $K = (PS)^{N_2}$, sends $PS \oplus g^{N_2}, g^{N_1} \oplus B, PS \oplus Ts$ and $H(N_1)$ to the client;

Step 3: The client computes $H(N_1)$, and compares it with the $H(N_1)$ received from the server, if not equal, then terminates the authentication, else it computes out B and $PS = g^{PIN \parallel PW \parallel B}$. Using the PS , the client can compute $PS \oplus (PS \oplus g^{N_2})$ and $PS \oplus Ts$ to obtain g^{N_2} and Ts , respectively, and generate a K as $K = (g^{N_2})^{PIN \parallel PW \parallel B}$, then sends the authentication information $E_K\{Ts - Ta, g^{N_2}\}$ to server.

Step 4: The server uses the K to decrypt $E_K\{Ts - Ta, g^{N_2}\}$ to obtain $Ts - Ta$ and g^{N_2} , then comparing the received g^{N_2} with the sent, if they are equal, the server accepts the user authentication, and sets a connection authentication counter i as 1, then stores the $Ts - Ta$ and i , else server terminates the authentication.

3) *The process of services communication*

Alice \rightarrow Server: Service request $\parallel E_K\{(Ts - Ta) \oplus i\}$

Server \rightarrow Alice: Service response

Step 1: Once passing the initial authentication, the client will store the $Ts - Ta$ and set i as 1. Then during the communication with server, the client must regularly (internals t) send the connection authentication factor $E_K\{(Ts - Ta) \oplus i\}$ to the server, each sending, i should be added by 1.

Step 2: When receiving the i th connection authentication information, the server decrypts $E_K\{(Ts - Ta) \oplus i\}$ to obtain $(Ts - Ta) \oplus i$, then compares it with the stored. If the result is same, the server keeps the connection with the client, and adds i by 1. Otherwise, if the connection authentication information is not received in time or is incorrect, the server will terminate the connection, once Alice needs to access the server again, she must start a new request for connection authentication.

4) *The security analysis of the scheme*

① In the scheme, with the introduction of big random factor B , the flexibility in modifying authentication basis can be strengthened. Normally, in order to keep user account safe, the password must be altered regularly, if a user thinks it is troublesome to remember a new password, he may generate a new PS through altering the random number without remembering a new password; ② The scheme can effectively resist speculating password. The scheme uses dynamic password without directly transmitting password to authenticate, computing out g^{N_1} as secret key with N_1 . By the difficulty of discrete logarithm problem, the attacker cannot obtain B , thus ensures the security of PS ; ③ In the scheme, by exchanging random numbers, the mutual authentication between the client and the server AS can be realized, thus can prevent against replay attacks; ④ Through regularly

(internals t) sending $E_K\{Ts-Ta \oplus i\}$ to resist the jacking connection attacks. Because t is small, even if it is attacked, the attacker can just visit V in a t interval at most, thus effectively ensure the security of information application.

C. The Kerberos improvement

In this section, an improved Kerberos authentication scheme will be put forward by means of DP method.

1) *Improvement of register*: Firstly, a pair of keys (K_P, K_R) is chosen for the authentication server. Among them, K_P is public, and K_R is private. In an authentication domain, the authentication server only needs this pair of keys.

When registering, each user chooses identity symbol IDc and password, and selects a large random Number B, the client computes the credential $PS=g^{IDc/PW/B}$, the calculate $PS \oplus B$, send $E_{K_P}(IDc/(PS \oplus B))$ and $E_{K_P}(B)$ to the authentication server and require registration (H refers to hash function; $E_{K_P}(\cdot)$ refers to encrypt the information in the parentheses with K_P). When receiving the registration information, the server uses private key K_R to obtains IDc, $PS \oplus B$ and B to check the uniqueness of IDc. If registration is found, the user is required to re-choose an IDc. Otherwise, the server calculates: $PS=(PS \oplus B) \oplus B$, and finally stores IDc, PS and B in safety database. So far, the registration process is over.

2) *The improvement of authentication process*: In the process of authentication, random factor is introduced to realize two-way authentication. In applications accessing stage, connected authentication is increased to resist jacking connection attack. Specific improvement process is the following:

- ① $C \rightarrow AS: ID_c \parallel ID_{TGS} \parallel E_{K_P}(N_I) \parallel TS1$;
- ② $AS \rightarrow C: ID_c \parallel g^{N_I} \oplus B \parallel E_{P_S}(N) \parallel H(N_I) \parallel Ticket_{TGS} \parallel E_{P_S}[K_{c,TGS} \parallel TS2 \parallel ID_{TGS}]$,
Notes: $Ticket_{TGS} = E_{K_{TGS}}(K_{c,TGS} \parallel ID_c \parallel H^{n+1}(N) \parallel TS2 \parallel ID_{TGS})$;
- ③ $C \rightarrow TGS: ID_v \parallel Ticket_{TGS} \parallel Auth_{c,TGS}$,
Notes: $Auth_{c,TGS} = E_{K_{c,TGS}}(ID_c \parallel TS3)$;
- ④ $TGS \rightarrow C: ID_c \parallel Ticket_v \parallel E_{K_{c,TGS}}(K_{c,v} \parallel TS4 \parallel ID_{TGS})$,
Notes: $Ticket_v = E_{K_v}(K_{c,v} \parallel ID_c \parallel ID_v \parallel TS4)$;
- ⑤ $C \rightarrow V: Ticket_v \parallel H^n(N) \parallel Auth_{c,v}$,
Notes: $Auth_{c,v} = E_{K_{c,v}}(ID_c \parallel TS5)$;
- ⑥ $V \rightarrow C: E_{K_{c,v}}(TS5)$.

Specific steps as the follows:

- The client C chooses random N_I and gets $E_{K_P}(N_I)$. Because K_P is the public key, only the private key can decrypt it and get correct N_I . Therefore, the client and verified AS at the same time.
- AS sends $g^{N_I} \oplus B$ to the user, based on the difficulty of solving the discrete logarithm problem, only through knowing N_I can it get B, only with B can it calculate: $PS=g^{IDc/PW/B}$, and N and $[K_{c,TGS} \parallel TS2 \parallel ID_{TGS}]$ can be obtained by using PS. By this way, the authentication is realized and N is the computing seed to link the authentication.

- By sharing K_v , TGS sends the initial connection authentication information $H^{n+1}(N)$ to the application server. In the process of visiting, the client sends authentication information $H^{n-i}(N)$ regularly (interval t) to maintain the connection with the server. The initial value of i is 1, and i is added by 1 each time when sending connection authentication information. After receiving the i th $H^{n-i}(N)$, V gets $H^{n-(i-1)}(N)$, and compares it with stored $H^{n-(i-1)}(N)$. If they are equal, authentication goes through and changes the stored value into $H^{n-i}(N)$. If the server does not receive user's authentication information timely and correctly, the connection with the user will be over.

The third, forth and sixth step in the improved program is the same with those in the original agreement.

V. THE ANALYSIS OF THE IMPROVED SCHEME

A. The feasibility analysis of the improved scheme

The improved scheme keeps the authentication main structure of the original one; it only adds some security protection fields in the data structure of the first, second and the fifth step. All added fields do not change the process of the original scheme, so the improved scheme is feasible.

Comparing to the original scheme, the amounts of computing have increased in the improved scheme, which are shown in Table 1. But in practice, it is very fast to do Hash, thus cannot cause an obvious decrease in authentication.

TABLE I. THE CONTRAST OF COMPUTING

| Operation | Counts | |
|-----------------------|--------------|--------------|
| | The original | The improved |
| Public key encryption | 0 | 1 |
| Symmetric encryption | 2 | 2 |
| Discrete logarithm | 0 | 2 |
| Hash | 0 | n |

B. The analysis of the security of improved scheme

The improved Kerberos scheme can effectively prevent the attack from password speculation, realize the two-way certification functions between the client and the server AS, and better resist the jacking connection attack. The specific instructions are below.

a) In the improvement scheme, with the introduction of big random factor B, the security can be enhanced effectively when the user uses a shorter password. Due to the authentication reference: $PS=g^{IDc/PW/B}$, even if the user uses a shorter password (at least 64Bits), he also can obtain strong authentication basis. At the same time, the flexibility in modifying authentication basis can be strengthened, and the user can generate the new PS through setting the new PW. Besides, the new random B is also generated to obtain new PS without memorizing the new password. This can well meet the "password change regularly" rule.

b) In the improvement scheme, PS is used as the encryption key instead of Kc in the original scheme. By

c) using the difficulty of discrete logarithm of DH as the security basis and introducing random N_I as the authentication factors, it is difficult for the attacker to guess the N_I , thus he cannot obtain B and calculate PS . Therefore the attacker can not obtain $K_{c,tgs}$, and there is no risk of speculating password[8,9].

d) By using random N_I , the mutual authentication is achieved between the client and the server.

AS can be realized in the Improvement scheme. The client sends $E_{K_p}(N_I)$ to AS, only having AS K_R can get N_I , then the client realizes authentication to AS through verifying $H(N_I)$; when AS sending $g^{N_I} \oplus B$ to the client, only by knowing N_I can we calculate B , thus to calculate PS , further to get the correct $K_{c,tgs}$. Finally, the authentication to the client is realized.

e) The application server realized the authentication to the client by the connection authentication computing seed N produced by AS. Based on a Hash operations on $H^n(N)$ which is send to the client, compared with $H^{n+1}(N)$ from the TGS, the authentication to the client is realized.

f) The improvement scheme can effectively resist jacking connection attack. The client sends $H^{n-i}(N)$ regularly to the application server V. After receiving it, V carried Hash operation again to get $H^{n-(i-1)}(N)$ and compare it with the stored $H^{n-(i-1)}(N)$. If they are equal, authentication gets through and $H^{n-(i-1)}(N)$ is modified to $H^{n-i}(N)$. Otherwise, if the server did not receive the correct authentication information from the user timely, the connection with the user will terminate. Because it is small, even if it is attacked, the attacker can just visit V in a t interval at most, thus effectively ensure the security of information application.

g) The requirements of timestamps are effectively decreased. Timestamp is no longer used to prevent replay attack, and it is mainly used to assist in determining the validity of tickets. So, the requirements of system time synchronization become greatly lower, and the complexity of time synchronization can be effectively decreased in a distributed network.

h) The improvement scheme can effectively prevent replay attack. During accessing application service, that the client sends $H^{n-i}(N)$ regularly to the application server can effectively prevent replay attack. Namely, if an attacker snaffles a ticket, which is in validity period, but owing to the unidirectional of Hash operations, he cannot constructe the $H^{n-i-1}(N)$ from $H^{n-i}(N)$, therefore, through the replay connection requests, the application server cannot compute a legal connect evidence, thereby it can effectively prevent against the replay attacks.

VI. CONCLUSION

Aiming at several typical security problems in Kerberos authentication program, by using dynamic password techniques, this paper made some improvements which can effectively enhance the security performance of Kerberos authentication program, which is more practical than the original scheme. However, on the basis of keeping the authentication main structure of the

original scheme, the improved scheme still exists some weaknesses:

1) Similar to the original scheme, there is such a hypothesis: KDC user database is absolutely safe, however, once the database is broken through, all user information might be exposed, it will result in certification system failure. To solve this problem, a token generating algorithm (TGA) may be considered, each time a user is authenticated successfully, the algorithm computes a token and sends it to the user for the next request, but the server doesn't keep this token. When the user asks for authentication again, the KDC does not only to verify his personal information, but also verify his token by calculating; its process is shown in Fig.3. Thus, even an attacker obtains the user information, he cannot pass through the authentication.

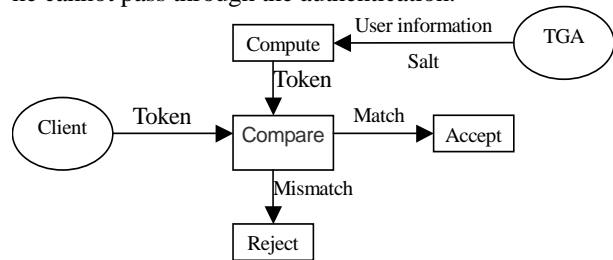


Figure 3. The authentication principle based on token

2) The session can be eavesdropped. Because this scheme uses symmetry encryption communication mechanism same as the original scheme, the session keys between the client and the server are generated by the Kerberos, therefore Kerberos can eavesdrop those session and does not burden, it is shown in Fig.4. To solve this problem, key agreement can be considered, if the session key generated by communications mutual consultation, the third-party wiretapping information may be avoided.

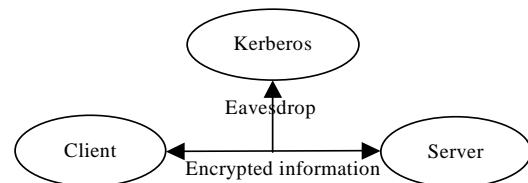


Figure 4. Kerberos eavesdrops communication

For the above problems, here only the feasible solutions are given, and the specific work needs to be studied further.

ACKNOWLEDGMENT

This work is supported by Artificial Intelligence (College) Key Laboratory of Sichuan Province Natural Science Funds (2008RK004 and 2008RK006). Thanks to all literature authors.

REFERENCES

[1] B. Clifford Neuman, Theodore Ts'o. Kerberos: An Authentication Service for Computer Networks. IEEE Communications Magazine, Volume 32, Number 9, pages 33-38, September 1994.
 [2] John T. Kohl, C. Neuman : The Kerberos Network Authentication Service (V5), RFC 1510, September 1993

- [3] Neuman C., The Kerberos network authentication Service(v5), RFC 4120, 2005.
- [4] BELLOVIN SM,MERRITTM., "Limitations of the Kerberos authentication systems", Journal of ACM SIGCOMM Computer Communication Review,1990, 20 (5) :119-132.
- [5] Law L,Menezes A,Qu M., "An efficient protocol for authenticated key agreement", Designs, Codes and Cryptography, 2003,28(2):119-134.
- [6] Kwon Tackyoung,Park Young-Ho,Lee Hee Jung., "Security Analysis and Improvement of the Efficient Password-Based Authentication Protocol", IEEE Communications Letters, 2005,9(11):93-95.
- [7] YAO Gang, FENG Deng-Guo, "Pairwise Key Agreement Protocols Based on the Weil Pairing", Journal of Software,2006,17(4): 907-914.
- [8] HU Ying-song, XU peng., "An improved Kerberos algorithm based on the secure remote password and the Public Key algorithm", Journal of Computer Engineering & Science, 2007,29(5):83-85,134, (in Chinese).
- [9] WU Zhi-guang ,LU Yin-hua ,WENG Hui-yu, "Analysis of Kerberos Protocols Based on EnvironmentalS ensitive Bisimulation Method", Journal of Computer Simulation, 2007,24(2): 99-102, (in Chinese).
- [10] TANG Wei-dong,LI Wei-min, "Improving Kerberos protocol with Diffie-Hellman algorithm", Journal of Computer Engineering and Design, 2007,28(2):343-345, (in Chinese).
- [11] LI Ji-yong., "New Kerberos protocol basedon Weil pairing", Journal of Computer Engineering, 2008, 28(2): 422—423, (in Chinese).
- [12] Wu Xilan,Shu Yuanzhong,Jiang Zetao,Wu Zhihong, "An Improvement Kerberos Protocol with PKI Technology", Journal of Computer Applications and Softwar, 2009,26(2):85-87, (in Chinese).
- [13] CHEN Jia-qi, FENG Jun, HAO Yan, "Improvement of Cross-realm Kerberos with Certificateless Key Agreement Protocol", Journal of Computer Engineering, 2010,36(20):150-152, (in Chinese).
- [14] WU Chunxue,LIU Liusheng, "Improving Kerberos system based on the key agreement protocol of identity based cryptography", Journal of University of Shanghai for Science and Technology, 2010,32(4):365-368, (in Chinese).
- [15] CHEN Can, LI Jun., "Formal analysis of Kerberos protocol", Journal of Microelectronics & Computer, 2006,23(6):49-51,55, (in Chinese).
- [16] HUANG Ye-yu, CHEN Qin., "New dynamic password authentication scheme", Journal of Computer Engineering and Design, 2005,26(7):1735-1736,1739, (in Chinese).
- [17] LEI Wen,ZHAO Pan, ZHANG Hong., "Study and improve on a dynamic password authentication scheme", Journal of Sichuan University of Science & Engineering(Natural Science Edition), 2009,22(5):47-50, (in Chinese).



Wen Lei Wen Lei received his M. Eng. degree in Computer Software and Theory from University of Electronic Science and Technology of China, Chengdu, China, in 2010. He is now a lecturer with the Department of Network Engineering, School of Computer Science, Sichuan University of Science & Engineering. His research interests include information security, software design and software theory.



Hai Cao Hai Cao was granted the M. Eng. degree in Computer Software and Theory from University of Electronic Science and Technology of China, Chengdu, China, in 2009. He is an associate professor with the Department of Computer Science & Technology, School of Computer Science, Sichuan University of Science & Engineering. His research interests include computer application and information processing.



Xingjian Liang Xingjian Liang was granted the M. Eng. degree in Computer Software and Theory from University of Electronic Science and Technology of China, Chengdu, China, in 2009. As a lecturer, now he works in the Department of Network Engineering, School of Computer Science, Sichuan University of Science & Engineering. His research fields cover information system and software design.



Hong Zhang Hong Zhang received his M. Eng. degree in Computer Software and Theory from University of Electronic Science and Technology of China, Chengdu, China, in 2010. Now He is a lecturer with the Department of Network Engineering, School of Computer Science, Sichuan University of Science & Engineering. His research interests include network communication, software design and software analysis.