A new pre-authentication protocol in Kerberos 5: biometric authentication

Conference Paper · January 2015

DOI: 10.1109/RIVF.2015.7049892

4 authors:

Hoa Quoc Le
Ho Chi Minh City University of Science
1 PUBLICATION **5** CITATIONS
SEE PROFILE

Hung Phuoc Truong
Ho Chi Minh City University of Science
11 PUBLICATIONS **49** CITATIONS
SEE PROFILE

Thien Hoang Van
Ho Chi Minh City University of Technology (HUTECH)
15 PUBLICATIONS **131** CITATIONS
SEE PROFILE

Thai Hoang Le
VNU-HCM University of Science
81 PUBLICATIONS **834** CITATIONS
SEE PROFILE

A New Pre-authentication Protocol in Kerberos 5: Biometric Authentication

Hoa Quoc Le[1], Hung Phuoc Truong[1], Hoang Thien Van[2] and Thai Hoang Le[1]
[1] Faculty of Information Technology, Ho Chi Minh City University of Science, Ho Chi Minh City, Vietnam
{lqhoa, tphung, lhthai}@fit.hcmus.edu.vn
[2] Faculty of Information Technology, Ho Chi Minh City University of Technology, Ho Chi Minh City, Vietnam
vthoang@hcmhutech.edu.vn

***Abstract*—Kerberos is a well-known network authentication protocol that allows nodes to communicate over a non-secure network connection. After Kerberos is used to prove the identity of objects in client-server model, it will encrypt all of their communications in following steps to assure privacy and data integrity. In this paper, we modify the initial authentication exchange in Kerberos 5 by using biometric data and asymmetric cryptography. This proposed method creates a new pre-authentication protocol in order to make Kerberos 5 more secure. Due to the proposed method, the limitation of password-based authentication in Kerberos 5 is solved. It is too difficult for a user to repudiate having accessed to the application. Moreover, the mechanism of user authentication is more convenient. This method is a strong authentication scheme that is against several attacks.**

Keywords—Authentication, Kerberos, cryptography, biometric, fingerprint.

I. INTRODUCTION

Kerberos [1], [2] protocol provides scalable strong identity authentication over non-secure network environments. Kerberos is based on secret key encryption technology. Kerberos allows a user to authenticate once and then connect to servers within the realm of the Kerberos network, without authenticating again for a period time. It is used in business, government, military, and educational institutions. For instance, it is the native network authentication protocol in the Microsoft Windows 2000 [3] and later. Apple also integrates Kerberos into Mac OS X.

Despite of providing a strong cryptography mechanism, the weakness of Kerberos is password-based authentication. The security of the cryptographic algorithm depends on the complexity of password. We encounter three problems with password-based authentication. Firstly, a user tends to choose simple words, phrases, or easily remembered personal data, while others write the passwords down on an accessible document to avoid data loss. Therefore, attacker can easily guess users' passwords (especially based on social engineering methods). It also can be broken by simple dictionary attacks [4]. Secondly, most people use the same password across different applications. If a single password is compromised, it may open many doors. Finally, passwords are unable to provide a non-repudiation mechanism. It is the lack of direct connection between the password and the user. With a password input from a user, we do not know whether he is the legitimate user.

An approach to address the above limitation is asymmetric cryptography. Public key cryptography is used for pre-authentication in Kerberos [5], [6], [7]. Public key cryptography based authentication may replace traditional password-based identity authentication. They use public key cryptography to avoid a shared secret key between a client and the authentication server. The approach makes Kerberos more secure.

Another approach is the combination of biometric authentication with Kerberos. This solution is described by Fengling Han *et al.* [8]. It makes users access mobile commerce applications safely. They use a phone number as a username and a fingerprint biometric as a traditional password. This method focuses on the template security by obfuscating fingerprint minutiae. A watermark is embedded within the fingerprint image at the moment of acquisition. A digital watermark is calculated by hashing the mobile serial number and a timestamp. The watermark embedding key is the session key created by the Key Distribution Center. The watermark embedding corrupts fingerprint minutiae significantly but invisibly. As a result, only the valid biometrics acquired by the registered device could pass the identity authentication, then a user obtains a ticket for subsequent communications. The main contributions of this method are: (1) watermark links the mobile device to owners biometrics, and such a link offers forensic traceability; (2) watermark corrupts fingerprint minutiae, positive minutiae matching is impossible if watermark could not be removed successfully. This method bases on idea of Kerberos (not real Kerberos): the ticket granting and session key generation mechanism to create a biometric-Kerberos authentication protocol implemented in mobile commerce applications.

In this paper, we propose the method combining biometric authentication with Kerberos 5 for computer systems. We use biometric data for pre-authentication in Kerberos 5. The biometric authentication system replaces the password-based authentication system. Biometric authentication [9], [10] refers to verifying individuals based on their physiological and

behavioral characteristics such as face, fingerprint, hand geometry, iris, keystroke, signature, voice, etc. It is more reliable than password-based authentication. Biometric characteristics cannot be lost or forgotten. They are extremely difficult to copy, share, and distribute. They require the person being authenticated to be present at the time. It is difficult to forge biometrics (it requires more time, money, experience, and access privileges). It is unlikely for a user to repudiate having accessed the application using biometrics. Biometric authentication supports the facet of identification, authentication and non-repudiation in information security. In this method, we use fingerprint biometric data that are stored in the authentication server. We also use asymmetric cryptography to make the authentication process more secure. The encryption and biometrics with the authentication server and the ticket granting server makes this method unique. It can guard against almost all kind of possible threats. We take care of privacy of the user, trust between a user and an authentication server and network security related issues [11].

In computer security, general access control includes identification, authentication and authorization (see Fig. 1). The proposed method focuses on the authentication stage. Identification and authorization stages still work on principle of the basic Kerberos.

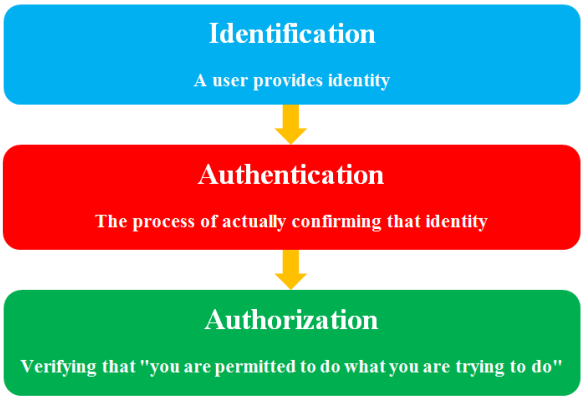

Fig. 1. Access control

The rest of the paper is organized as follows: Section II introduces the background. A new pre-authentication protocol is proposed in Section III. Section IV is the discussion. Security and privacy provided by the proposed method is presented in Section V. Section VI simulates and shows the experimental results for the proposed protocol. Section VII concludes the result.

II. BACKGROUND

A. The Basic Kerberos 5 Authentication Protocol

Kerberos was originally developed for the distributed computing environment that MIT deployed in the 1980s as Project Athena. It is a secure authentication mechanism designed for distributed severs, which assumes the network is unsafe. The first public release was Kerberos version 4, which leads to the actual version (v5) in 1993 after a wide public review. It followed the IETF standard process and its specifications are defined in Internet RFC 1510 (made obsolete by RFC 4120 in 2005). Originally designed for UNIX, it is now available for all major operating systems.

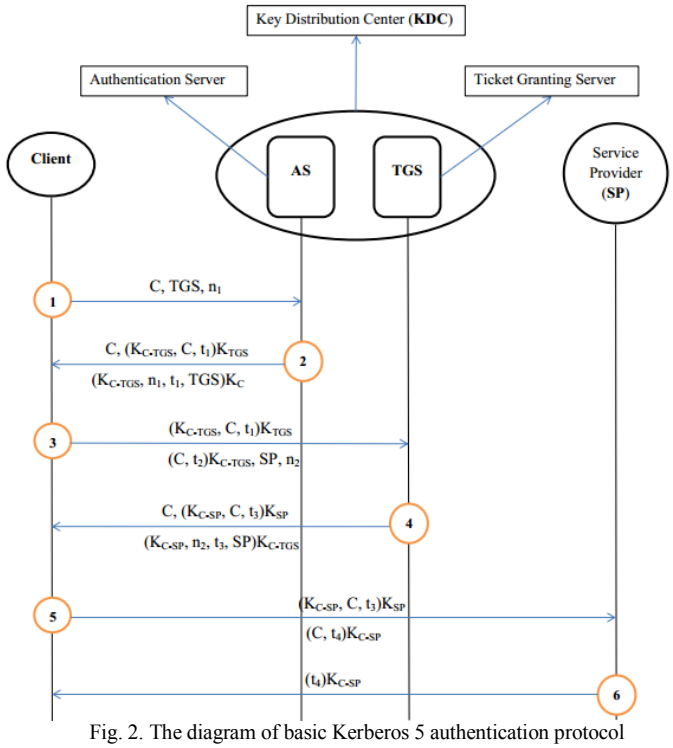

Fig. 2. The diagram of basic Kerberos 5 authentication protocol

C: Client
n: Nonce
t: Timestamp
$\mathbf{K_C}$: Client's symmetric key
$\mathbf{K_{TGS}}$: TGS's symmetric key
$\mathbf{K_{SP}}$: SP's symmetric key
$\mathbf{K_{C\text{-}TGS}}$: Session key between Client and TGS
$\mathbf{K_{C\text{-}SP}}$: Session key between Client and SP

These steps in Fig. 2 are presented in detail below:

Step 1: Client **C** sends a service request to the **AS**. This contains the name of client **C**, the name of the **TGS** and a nonce n_1.

Step 2: After recognizing **C**, the **AS** replies with a message containing two encrypted components: $(K_{C\text{-}TGS}, C, t_1)K_{TGS}$ and $(K_{C\text{-}TGS}, n_1, t_1, TGS)K_C$. The first component is the ticket granting ticket (**TGT**). The second component contains a session key $K_{C\text{-}TGS}$, a nonce n_1, a timestamp t_1 and the name of **TGS**. The key K_C used to encrypt the second component is a long term secret between **C** and the **AS** derived from the user's password.

Step 3: Client **C** transmits the cache ticket $(K_{C\text{-}TGS}, C, t_1)K_{TGS}$ and the name of the service provider (**SP**) together with a newly generated nonce n_2 and the authenticator $(C, t_2)K_{C\text{-}TGS}$, where t_2 is a timestamp. The authenticator proves to the **TGS** that **C** actually knows the session key $K_{C\text{-}TGS}$.

Step 4: The **TGS** decrypts the ticket and the authenticator, verifies request then creates the session key $K_{C\text{-}SP}$. It is

encrypted with the session key $K_{C\text{-}TGS}$. The **TGS** also issue the service ticket $(K_{C\text{-}SP}, C, t_3)K_{SP}$ for requested **SP**, encrypts it with K_{SP}. The TGS sends three components $[C, (K_{C\text{-}SP}, C, t_3)K_{SP}, (K_{C\text{-}SP}, n_2, t_3, SP)K_{C\text{-}TGS}]$ back to client.

Step 5: This exchange takes place each time the client initiates a new session with the **SP**. With a service ticket in hand, client **C** contacts the **SP** with this ticket $(K_{C\text{-}SP}, C, t_3)K_{SP}$ and an authenticator $(C, t_4)K_{C\text{-}SP}$.

Step 6: The **SP** verifies that ticket and authenticator match, and then grant access to service. If mutual authentication required, **SP** returns the timestamp t_4 that client **C** included in its request. Timestamp t_4 is encrypted with the session key $K_{C\text{-}SP}$.

B. The Pulic Key Kerberos 5 Authentication Protocol

PKINIT is known as an extension to Kerberos 5, which uses public key cryptography to avoid a shared secret key between a client and the authentication server [6], [7]. It modifies the initial authentication exchange. However, the remaining parts of the Kerberos 5 protocol (step 3 to step 6) are unchanged. This protocol extension adds complexity to Kerberos 5. It retains symmetric encryption in the later rounds but relies on asymmetric encryption in the first round. The first message pairs (the first round) exchanged in Fig. 3 are the client contacting the **AS** to request a ticket. The second message pairs (the second round) use the session key to request a ticket from a ticket granting server (**TGS**). The third message pairs (the third round) are to access a service from the resource using this ticket [8].

The client and the authentication server have independent public/private key pairs. The certificate issued by a Public Key Infrastructure (PKI) is independent from Kerberos. It is used to testify the binding between each principal and its public key. Authentication decisions can be reached based on the trust **AS** that holds in a few known certificates. Fig. 3 shows the process of authentication:

Step 1: This shows the relevant parts of the request that the client **C** sends to the **AS**. The last part of the message (C, TGS, n_1) is the same as in the basic Kerberos 5. It contains the name of client **C**, the name of the **TGS**, and a nonce n_1. The "boxed" parts are added by PKINIT and contain the certificate of client $Cert_C$ and his signature (with the private key Pri_C) over a timestamp t_1 and another nonce n_2.

Step 2: This shows the formalization of **AS**'s response, which is more complex than that of basic Kerberos 5. The last part of the message $(K_{C\text{-}TGS}, n_1, t_2, TGS)K, C, (K_{C\text{-}TGS}, C, t_2)K_{TGS}$ is very similar to AS's reply in the basic Kerberos 5. The difference of the first part is the symmetric key K protecting $K_{C\text{-}TGS}$. K is now freshly generated by the **AS** and not a long-term shared key. It is freshly generated for the reply. It must be informed to client **C** before client **C** can learn $K_{C\text{-}TGS}$. PKINIT does this by adding the boxed message $[Cert_{AS}, (K, cksum)Pri_{AS}]Pub_C$. This contains **AS**'s certificate and its signature (with its private key Pri_{AS}), over K and the checksum *cksum* (*cksum = Hash (C's request)*). The boxed message $Cert_{AS}, (K, cksum)Pri_{AS}$ is encrypted under **C**'s public key Pub_C.

Step 3 to step 6: Similar to the basic Kerberos 5.

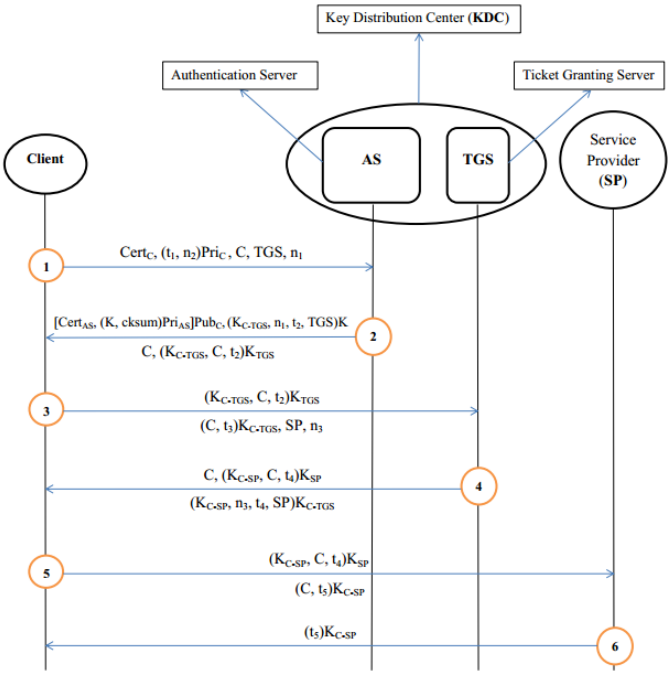

Fig. 3. The diagram of public key Kerberos 5 authentication protocol (PKINIT-27 [7])

III. A New Pre-authentication Protocol

To prepare for demonstrating the proposed protocol, we assume following conditions: (1) the authentication server has already stored fingerprint references of clients; (2) the client C and the authentication server have independent public/private key pairs. We use MCC Software Development Kit (SDK) Version 1.4 for biometric authentication (see Fig. 4). MCC SDK is a .Net DLL library that enables to develop fingerprint verification applications using the Minutia Cylinder Code (MCC) algorithms [12], [13], [14]. P-MCC (Protect Minutia Cylinder Code) [14] is a fingerprint verification method that ensure template security.

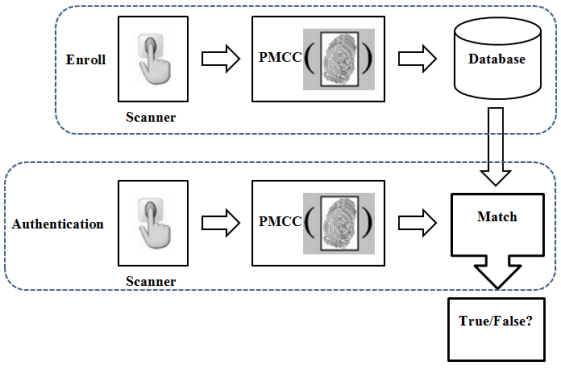

Fig. 4. Fingerprint biometric authentication

The process of authentication follows the steps shown in Fig. 5. Firstly, client authenticates with the authentication server. If client authenticates successfully, the authentication server will sends the ticket to client to contact the ticket granting server. Then, the ticket granting server sends the ticket to client to communicate with the service provider.

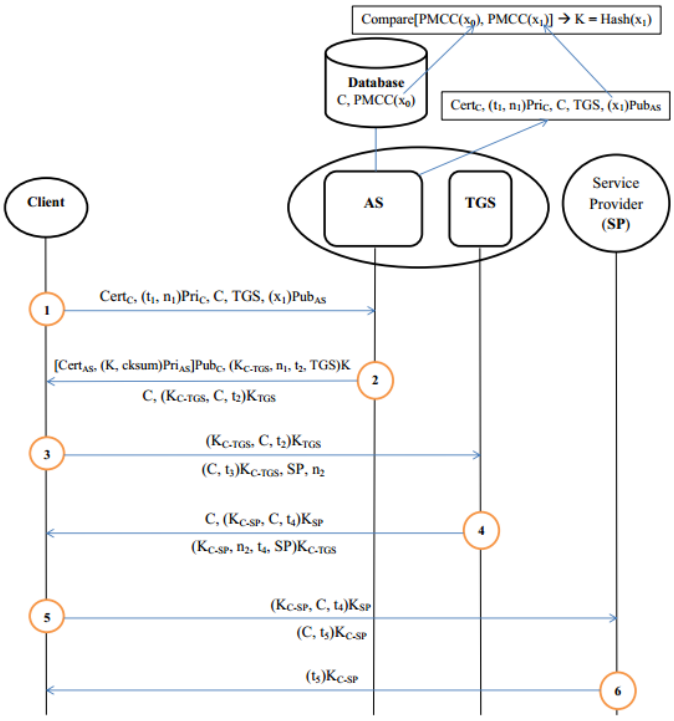

Fig. 5. A new pre-authentication protocol

Step 1: Client sends an identity **C**, the name of **TGS**, a fingerprint x_1 encrypted with the **AS**'s public key Pub_{AS}, the client's certificate $Cert_C$ and his signature (with the private key Pri_C) over a timestamp t_1 and a nonce n_1 to the **AS**.

Step 2: The **AS** decrypts the $(x_1)Pub_{AS}$ to obtain fingerprint x_1. The **AS** compares a $P\text{-}MCC(x_1)$ with a $P\text{-}MCC(x_0)$ stored in database (The comparison is implemented by MCC SDK). If there is a match, the **AS** creates random key K from x_1 by the hash function ($K = Hash(x_1)$). Because fingerprint x_1 is changed in every acquisition times, value of key K is created differently by the hash function. Then the **AS** signs over a random key K and checksum *cksum* ($cksum = Hash\ [Cert_C, (t_1, n_1)Pri_C, C, TGS, (x_1)Pub_{AS}]$) with its private key Pri_{AS}. The **AS** encrypts them and the **AS**'s certificate $Cert_{AS}$ with the client's public key Pub_C. The next, the **AS** creates the ticket granting ticket and a session key $K_{C\text{-}TGS}$. A timestamp t_2, a nonce n_1, a session key $K_{C\text{-}TGS}$ and the name of TGS are encrypted under random key K. Finally, all of this is sent back to client.

Step 3: Client **C** receives message from the **AS** in step 2. Client **C** uses his private key to decrypt $[Cert_{AS}, (K, cksum)Pri_{AS}]Pub_C$. Client C obtains $Cert_{AS}$ and $(K, cksum)Pri_{AS}$. Then, client **C** decrypts $(K, cksum)Pri_{AS}$ with the **AS**'s public key to obtain random key K and checksum *cksum*. Client **C** compares checksum *cksum* with checksum of message in step 1 to detect man-in-middle attack. Finally, client **C** transmits the cache ticket granting ticket and the name of **SP** together with a newly generated nonce n_2, and the authenticator $(C, t_3)K_{C\text{-}TGS}$, where t_3 is a timestamp.

Step 4 to step 6: Similar to the basic Kerberos 5.

IV. Discussion

In our proposed method, the authentication between client and server is based on not only public key cryptography but also biometric data. They increase the complexity of the pre-authentication stage in Kerberos 5. It is noticed that our proposed method has some advantages listed as below:

(1) The safety of the public key Kerberos 5 protocol (see Fig. 3) depends on the safety of client's private key. Client's private key is stored in a place that does not belong to human (e.g. smart card); so it could be stolen or copied. Biometric data is added in our proposed method to prevent hacker from attacking the system easily.

(2) According to Fengling Han *et al.*[8], we find that authors used public key cryptography to exchange the information (a random number, a mobile number, a timestamp, and a session key) between client and the authentication server. They ignored the process of pubic key exchange because of implementing the protocol in the mobile environment that required the length of packet as short as possible. As we know, the computational capabilities of computer is higher than mobile's, so our proposed method keeps the process of public key exchange (round 1 in Fig. 5). Despite the longer packet, the processing time may be acceptable due to the high computational capabilities of computer.

(3) Due to public key cryptography, the important information (a biometric data and a random key) is exchanged between client and the authentication server securely. With having the private key, only client and the authentication server can decrypt and read the important information to be able to perform these next steps. The combination of public key cryptography and biometric data make Kerberos more strong. In fact, if a hacker steals the biometric data then he needs the client's private key to decrypt the information sent from the AS. Else if a hacker steals the client's private key then the biometric data is required to pass the biometric authentication step in the AS. Probability a hacker can steal both the biometric data and the client's private key is very low.

(4) Biometric samples at sampling times will be different, which creates unstable data. We find advantages from the limitation by creating various keys K through hashing biometric samples in every session.

V. Security and Privacy Provided by the Proposed Method

We analyze most of cases where the system is attacked. We see how the security risks are handled by the proposed method. Firstly, the database containing user's information and biometric template is at risk. Secondly, we are using an unsecure network.

A. Hacker Attacks on the Biometric Template Database on the Server Side

In this case, we know that the biometric template is stored in the P-MCC template. The P-MMC template is noninvertible. It is a protected template containing only bit vectors. The spatial location and direction of the minutiae are

not stored in it. Even if the hacker replaces the user's P-MCC template with his P-MCC template in order to authenticate successfully with his biometric data then he has to get the user's private key to understand the information sent from the authentication server.

B. *Hacker Steals the User's Biometric on the Client Side*

We are considering the advantages of not only the biometric authentication but also the security of public key cryptography. If hacker gets the user's biometric he cannot do anything because the user's private key is required to decrypt the information sent from the authentication server.

C. *Network Security*

The network can be secured by cryptographic methods like symmetric or asymmetric cryptography. All traffic is encrypted either by client's public key or random session key.

D. *Replay Attack*

Replay attack will usually result in the attacker assuming the victim's identity without actually recovering the password. Replay attacks against Kerberos are targeted on the final message transferred from the client to the service provider. An attacker will attempt to capture this message and reuse its data to authenticate himself as the victim. If successful, the attacker will have full access to the same service the victim accessed. This attack requires that traffic from the victim to the server is subverted to the attacker's network address. The use of timestamp, nonce, session key and ticket prevents the replay attacks.

E. *Password Attack*

Password attack can be based on any text encrypted with the key derived from the victim's password, and will result in exposure of the plaintext password. This attack is solved by proposed method because of the use biometric data and public key cryptography.

F. *Man-in-middle Attack*

The man-in-middle attack is a form of active eavesdropping in which the attacker makes independent connections with the victims (client and the AS) and relays messages between them, making them believe that they are talking directly to each other over a private connection, when in fact the entire conversation is controlled by the attacker. The attacker must be able to intercept all messages going between the two victims and inject new ones. This is a deterministic attack. The attacker must be a legal user. The checksum-based approach in the proposed method can solve it.

G. *Forwardable Tickets*

Let's suppose we have a work session on a machine with the related TGT and wish to login from it onto another machine, keeping the ticket. Forwardable tickets are the solution to this problem. A ticket forwarded from one host to another is in itself forwardable, thus once authenticated it is possible to access the login on all the desired machines without having to re-enter any password.

Besides, applications or users can request a new ticket to the TGS, based on a previously obtained ticket with the forwardable flag set. This option of the Kerberos 5 protocol is what makes it possible to implement single-sign-on (SSO) using Kerberos. Based on the obtainment of a single ticket, a user could request access to all services across the network, without having to re-authenticate.

VI. SIMULATION AND EXPERIMETNS

We simulate the proposed method by implementing on a client-server architecture. This application is programed by C# language in Windows operating system. This client-server socket connection is done using the TCP/IP method. We use a .Net DLL library MCC SDK for fingerprint biometric authentication. The asymmetric encryption method is the RSA 256 bits. The application is implemented on a configuration of workstation of Intel Corei3 processor, with 2GB RAM. The simulated system includes client, the authentication server, the ticket granting server, and the service provider.

Data sets used in the following experiments are:

- FVC2004 DB2: 800 fingerprints from 100 fingers (8 impressions per finger). FVC2004 databases are markedly more difficult than FVC2002 and FVC2000 ones, due to the perturbations deliberately introduced. Therefore, one should neither compare error rates among different FVC competitions, nor conclude that the state-of-the art in fingerprint matching is not improving. The properties of FVC2004 DB2 are: sensor type: optical sensor; image **s**ize: 328x364 (119 Kpixels); resolution: 500 dpi.
- FVC2006 DB2: 1680 fingerprints from 140 fingers (12 impressions per finger). Data collection in FVC2006 [15] was performed without deliberately introducing difficulties such as exaggerated distortion, large amounts of rotation and displacement, wet/dry impressions, etc. (as it was done in the previous editions), but the population is more heterogeneous and also includes manual workers and elderly people. The properties of FVC2006 DB2 are: sensor type: optical sensor; image size: 400x560 (224 Kpixels); resolution: 569 dpi.

According to Matteo Ferrara *et al.*[14], we choose $k = 64$ ($k \in \{128, 64, 32, 16\}$) because P-MCC (with $k = 64$) is better than all existing approaches and is robust against type I and type II attacks at both security levels.

In the first experiment, as proposed in [16], two scenarios have been considered: a reconstructed template is used to attack (1) the system that stores the original template (type-I attack), or (2) other systems where another impression of the same finger is enrolled (type-II attack). Both attack scenarios have been evaluated on the fingerprint recognition system: *P-MCC* (the P-MCC algorithm described in [14]). Each attack scenario has been evaluated under two security levels: (1) medium-security, where the biometric verification threshold of each system has been set to 0.1% FMR [18]; (2) high-security, where the threshold of each system has been set to 0% FMR. The attack simulations were performed on FVC2006 DB2 for P-MCC templates. Each reconstructed template was matched against all the remaining original templates of the same finger.

TABLE I. PERCENTAGE OF SUCCESSFUL ATTACKS AGAINST THE SYSTEM ON FVC2006 DB2

Level	Type-I Attack	Type-II Attack
Medium	2.4%	0.8%
High	0.0%	0.0%

The results of Table I (described in [14]) show the percentage of successful attacks under medium- and high-security levels, respectively. The results prove that the proposed pre-authentication protocol (base on P-MCC fingerprint verification method) is effective against such attacks.

In the second experiment, the evaluation of the performance of user pre-authentication has been carried out on FVC2004 DB2 and on FVC2006 DB2. The original FVC protocol is used in the literature to report results of the performance of user pre-authentication. Each template is stored (in the AS) and matched against the remaining templates of the same finger to obtain the False Non Match Rate (FNMR) (also referred as False Rejection Rate - FRR). The first template of each finger is stored (in the AS) and matched against the first template of the remaining fingers in the data set to determine the False Match Rate (FMR) (also referred as False Acceptance Rate - FAR). If template T_1 is matched against T_2, the symmetric match (T_2 against T_1) is not executed, to avoid correlation in the scores. The following performance indicators (in Table III) are considered: Equal Error Rate (EER) [18], lowest FNMR for FMR $\leq$ 0.1%, lowest FNMR for FMR = 0%, and time of pre-authentication stage.

TABLE II. PERFORMANCE OF THE PRE-AUTHENTICATION STAGE

Data set	EER	FNMR (FMR ≤ 0.1%)	FNMR (FMR = 0%)	Time of Pre-auth
FVC2004 DB2	0.58%	0.73%	1.48%	17.2 ms
FVC2006 DB2	0.17%	0.23%	0.42%	18.4 ms

Table II reports the performance of the proposed method that uses biometric data (with P-MCC method) for pre-authentication stage. The percentage of verification accuracy is very high. Time of pre-authentication stage (round 1 in Fig. 5) is short and acceptable (19.4ms for FVC2004 DB2 and 18.2ms for FVC2006 DB2).

The results of experiments show that our proposed method is efficient, security and can be applied in practical systems.

VII. CONCLUSION

We propose the method combining biometric authentication with Kerberos 5 for computer systems. Biometric based authentication replaces password-based authentication at pre-authentication stage. It solves the limitation of password-based authentication. A user easily authenticates his identity by using his own biometric characteristic. This method provides non-repudiation. It is the assurance that a user cannot deny having accessed the application. We also use public key cryptography to increase the complexity of the protocol. The combination of encryption, biometric authentication, the authentication server and the ticket granting server makes the system that can against most attacks.

REFERENCES

[1] C. Neuman and T. Ts'o, "Kerberos: an authentication service for computer networks," IEEE Communications Magazine, vol. 32, no. 9, pp. 33–38, September 1994.

[2] C. Neuman, T. Yu, S. Hartman, and K. Raeburn, "The Kerberos network authentication service (V5)," IETF RFC 4120, July 2005.

[3] J. Garman, Kerberos: the definitive guide, O'Reilly, 2003.

[4] D. V. Klein, "Foiling the cracker: a survey of, and improvements to, password security," in Proc. 2nd USENIX Workshop Security, pp. 5–14, 1990.

[5] Ian Downnard, "Public-key cryptography extensions into Kerberos," IEEE Potentials, vol. 21, no. 5, pp. 34-38, December 2002.

[6] L. Zhu and B. Tung, "Public Key Cryptography for Initial Authentication in Kerberos," IETF RFC 4556, June 2006.

[7] Iliano Cervesato, Aaron D. Jaggard, Andre Scedrov, Joe-Kai Tsay, and Christopher Walstad, "Breaking and Fixing Public Key Kerberos," Information and Computation, vol. 206, no. 2-4, pp. 402-424, February-April 2008.

[8] Fengling Han, Mohammed Alkhathami, and Ron Van Schyndel, "Biometric-Kerberos Authentication Scheme for Secure Mobile Computing Services," International Congress on Image and Signal Processing, vol. 3, pp. 1694–1698, September 2013.

[9] A. K. Jain, R. Bolle, S. Pankanti, and Eds., "Biometrics: Personal Identification in Networked Society," Norwell, MA: Kluwer, 1999.

[10] Jain AK, Flynn PJ, and Ross AA, Handbook of Biometrics, Springer, 2008.

[11] Cryptography and network security, William Stallings 5th edition.

[12] R. Cappelli, M. Ferrara, and D. Maltoni, "Minutia Cylinder-Code: a new representation and matching technique for fingerprint recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 12, pp. 2128-2141, December 2010.

[13] R. Cappelli, M. Ferrara, and D. Maltoni, "Fingerprint Indexing based on Minutia Cylinder Code," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 5, pp. 1051-1057, May 2011.

[14] M. Ferrara, D. Maltoni, and R. Cappelli, "Noninvertible Minutia Cylinder-Code Representation," IEEE Transactions on Information Forensics and Security, vol. 7, no. 6, pp. 1727-1737, December 2012.

[15] R. Cappelli, M. Ferrara, A. Franco, and D. Maltoni, "Fingerprint verification competition 2006," Biometric Technol. Today, vol. 15, no. 7-8, pp. 7–9, Aug. 2007.

[16] J. Feng and A. K. Jain, "Fingerprint reconstruction: From minutiae to phase," IEEE Trans. Pattern Anal. Mach. Intell., vol. 33, no. 2, pp. 209–223, Feb. 2011.

[17] R. Cappelli, M. Ferrara, D. Maltoni, and M. Tistarelli, "MCC: A baseline algorithm for fingerprint verification in FVC-onGoing," in Proc. 11th Int. Conf. Control, Automation, Robotics and Vision (ICARCV), Singapore, 2010.

[18] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, Handbook of Fingerprint Recognition, 2nd ed. New York: Springer-Verlag, 2009.