A Framework for Token and Biometrics Based Authentication in Computer Systems

Jian De Zheng
School of Information Science and Technology, Xiamen University, Xiamen, China
Email: zhengjd@xmu.edu.cn

***Abstract*—User authentication is of vital importance to the security of computer systems. This paper proposes a new framework for multifactor authentication using token and various biometrics, such as fingerprint, retina scan, hand geometry and face pattern, which allows the authenticator, usually runs as a sever, to store only cipher texts, instead of the plaintexts of the biometrics templates, so as to reduce the risk of disclosing personal data of users. Another advantage of the framework is that the biometrics templates are bound to the private key inside the token therefore cannot be modified by changing server-resident data. The framework is based on a special challenge-response protocol, which is used to authenticate the token and decrypt the cipher texts of biometrics at the same time, such that live biometrics samples collected from the token owners can be matched to the recovered templates. Besides principles and architecture, a cryptographic study of the framework is also presented, which focuses on a formal proof for the security of the new protocol, under the Random Oracle Model.**

***Index Terms*—Computer security; Authentication protocol; Token; Biometrics**

I. INTRODUCTION

User authentication is of vital importance to the security of computer systems. It is well known that there are three basic methods for user authentication in computer systems, namely the password-based authentication, the token-based authentication and the biometrics-based authentication (FIPS PUB 190). Traditional password-based authentication has become inadequate for demanding applications; since the entropy of a password is inherently limited by the capability of human memory, while the computing power used by dictionary attackers is growing steadily. Token-based authentication is much stronger, since the secret key inside the token is usually as big as hundreds or thousands of bits. Furthermore, it is generated at random, and well protected by modern electronic technologies. However, tokens may be used by unauthorized people, in case they are lost or stolen. So in recent years, people have also turned to authentication methods using automated biometrics, such as fingerprint, retina scan, hand geometry and face pattern [1]. Since the above biological characteristics are intrinsic properties of individual people, they are difficult to be transferred or duplicated. Biometrics-based authentication also suffers a major setback, as it requires users to register their biometrics data to the authenticator in advance. This requirement could create a big ethical problem in case the data are disclosed on a large scale [2]. Meanwhile, biological measurement is statistical in general, and not very accurate in many cases, which may reduce the reliability of the authentication systems.

Multifactor user authentication is a security system in which more than one form of authentication is implemented to verify the legitimacy of a user. For examples, most tokens make use of passwords to activate the private keys inside, resulting in so called Chip and PIN authentication [3], while attempts have also been made to turn biometrics data of the users into keys for cryptographic authentication protocols, resulting in so called biometric cryptosystems [4], to name just a few.

This paper proposes a new framework for multifactor authentication using token and various biometrics, which allows the authenticator to store cipher texts, instead of the plaintexts of the biometrics templates, so as to reduce the risk of disclosing personal data of users. The core of this framework is a special challenge- response protocol, which is used to authenticate the token and decrypt the cipher texts at the same time. The recovered templates are next used to match live biometrics samples collected from the token owners. Besides principles and architecture, a cryptographic study of the framework is also presented, which focuses on a formal proof for the security of the new protocol, under the Random Oracle Model (ROM).

II. THE AUTHENTICATION FRAMEWORK

A. The Identity Recoverable Authentication Protocol

A straightforward way to combine token-based authentication with biometrics-based authentication is to compute keys, whether symmetric secret keys or asymmetric private/public keys, from the biometrics of the token owners as mentioned above, and use them with some token-based authentication protocol, including the classic challenge-response protocol. Such a combination would be valid if the biometrics data obtained in different times remain unchanged for the same user; however, this is not always the case. In fact, biometrics readings tend to vary from time to time, which obviously causes a big problem for the implementation of the above scheme. Theoretically, it is possible to solve the problem using so called fuzzy extractor, which is supposed to be capable of extracting a unique ID string from a biometrics input in such a way that a certain amount of error is allowed for, i.e., the extracted ID will be the same if the input changes slightly [5]. However, the method may not work well in practice, as noise often contributes significantly to the biometrics measurement.

This paper offers another solution. Instead of adapting biometrics measurements painfully to cryptographic algorithms, we propose a new scheme that incorporates both techniques in a natural way. Our basic idea is to develop a protocol that can authenticate a user based on the token he/she presents, while recovers the biometrics identity of the user, which is made of one or more biometrics templates, from messages exchanged during authentication, so that the token owner can be authenticated by more than one factor. Such a protocol is called identity recoverable authentication protocol or IRAP in short. It can be implemented by revising the classic challenge-response protocol based on a signature scheme. Let *a* represent the public key of the signer, ξ represent the challenge code, and η represent the response code, which is now the signature of ξ, this kind of implementations typically has the following features:

(a) The signature verification algorithm can be represented by

$$a = v(\xi, \eta), \quad (1)$$

where *v* is a polynomial time function, it allows the public key to be computed from ξ and η.

(b) The public key *a* is computed as

$$a = h(u \,||\, b_1 \,||\, b_2 ... \,||\, b_t), \quad (2)$$

where $b_1, b_2, ... b_t$ represent registered biometrics templates of the user, *h* represent a one-way collision free hash function, and "||" represent appending operation.

(c) The public key *a* is actually turned into a secret key, and used to encrypt biometrics templates of the user. The cipher texts are kept by the system as help data for the recovery of biometrics templates.

(d) The recovered templates are verified for their integrity using (2).

Note that in order to make use of the prescribed public key, an identity-based (IB) signature scheme should be employed to sign ξ.

B. The Archetecture

Based on the IRCR protocol introduced above, a multifactor authentication framework can be created, which provides a unified interface for integrating a token with various biometrics authentication subsystems, referred to as authentication service providers. The framework includes an authenticator, several authentication service providers, one or more private key generators (PKGs), the tokens of users, and miscellaneous hardware devices such as token reader/writer and biometrics scanners, the architecture of which is illustrated by figure 1. Note that such architecture is applicable not only to distributed systems, but also to personal computer (PC) systems. Since the authenticator and the private key generator are independent of the biometrics, they can be put into the firmware or operating systems of PCs by manufacturers as part of a standard configuration, while the authentication service providers, i.e., the biometrics authentication modules can be installed later by the users.

C. Basic Components

1. *The authenticator*. The authenticator is also called authentication manager due to the similarity between our model and the standard "manager versus service providers" model used in many software systems; it can be further divided into three modules. The first module implements an IRAP, which will be discussed in details later. The second module manages the cipher-texts of biometrics templates, which are given by $\sigma_i = E(\lambda, b_i)$, *i*=1,2,…*t*, where *E* represents a symmetric encryption algorithm, while the secret key λ is computed from *a* as $\lambda = h(s, a)$, where *s* represents a system secret. Note that in the framework *a* is no longer published, nor is it stored in the authenticator, so as to keep the confidentiality of the sensitive biometrics data. Finally, the third module of the authenticator invokes authentication service providers with respective templates. It is also responsible to make the final decision as to accept or reject a user, using the feedbacks from the service providers.

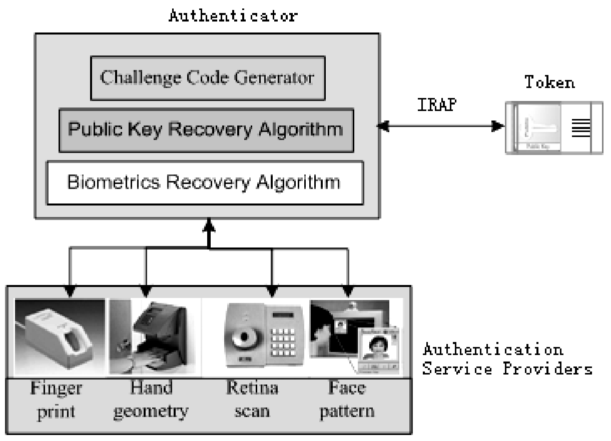

Figure 1. An illustration of the authentication framework

2. *The service providers*. The authentication service providers in the framework collect live samples of biometric characteristics from the users, compare them with biometrics templates received from the authenticator, and return the matching results back to it. In this paper, we assume that the same interface can be provided for the authenticator to invoke any of them. The interface is represented by the C-like pseudo code

```
bool BioFn ( String tplStr ),
```

where the argument *tplStr* represents a biometrics template. Therefore each authentication service provider can be called by the following VC++ codes:

```
HINSTANCE hinst = LoadLibrary (BioModule);
FARPROC lpfn = GetProcAddress (hinst, "BioFn");
BOOL f = (BioFn) lpfn;
BOOL authResult = false;
authResult = f(tplStr);
```

where BioModule represents the name of the biometrics module, assuming it is provided in the form of Dynamic Link Library (DLL); or by the following Java codes, which make use of the reflection property of the classes:

```
Class c = BioClass.getClass();
Class parameters[] = new Class[] { String };
Object[] arguments = new Object[]{ tplStr };
Method m = c.getMethod ( "BioFn",  parameters);
boolean authResult = false;
try { authResult = m. invoke ( BioClass, arguments ); }
catch (Exception e){}
```

where BioClass represents the name of the class that is the entry point of the biometrics module. A true return code signifies the acceptance of the user by the subsystem, while a false return code signifies the user being rejected.

3. *The private key generator*. Since an identity-based signature scheme is used to generate and verify response codes in the framework, a PKG is required to issue a private key for each user. There is no essential difference between the PKG in a classic IB signature scheme, and the PKG in the authentication framework.

In general, the PKG can be included in the authenticator as an additional module. However, more PKGs are also allowed to run in the framework, each of which can be maintained by a trusted third party, while its parameters can be certified using a digital certificate.

4. *The basic devices.* The tokens, token reader/writer and biometrics scanners are basic hardware devices in the authentication framework. The tokens primarily store the private keys; they are issued to the users, while the token reader/writer and the biometrics scanners are connected to the authenticator and the authentication service providers respectively through secure channels. We assume the existence of such channels; their construction is beyond this paper.

During authentication, the challenge code is written to the token, the response code is read from the token, while live biometrics of its owner are collected through the biometrics scanners, and checked for authenticity of his/her identity. We also assume that proper collecting process is enforced in one way or another, so as to guarantee the reliability of the collected data.

C. How to work with the framewok

To begin an authentication under the framework, the user inserts the token into the token reader/writer, which reads initial information, including account name of the user, from the token and passes them to the authenticator. The authenticator in turn, starts the IRAP, which authenticates the token, while recovers the biometrics templates of the token owner.

After the token is successfully authenticated, the authenticator starts another authentication process based on biometrics, which includes the following steps:

Step 1, the authenticator invokes the authentication service providers with respective templates recovered above; the templates are cached by the authentication service providers.

Step 2, the authentication service providers collect biometrics samples from the user on spot, match them to the cached templates.

Step 3, the authentication service providers return the match results to the authenticator, while removing the templates from their memories.

Step 4, the authenticator makes final decision with regards to the authenticity of the user identity, based on the feedbacks from authentication service providers, and a predefined decision policy.

III. An implementaion of IRAP

The following presents an implementation of the IRAP, which consists of three phases, namely, the setup phase, the token issuing phase and the token authentication phase. Like classic challenge-response protocol, one may use different signature schemes to generate response codes, resulting in different implementations. However, this paper studies only one implementation, where the algorithms for generating and verifying response codes are derived from the IB signature scheme proposed by Shamir [6]. For simplicity, this only implementation is also referred to by the general term “IRAP” without further classification.

A. The Setup Phase

The setup phase of the protocol consists of three steps, assuming that the PKG is included in the authenticator as an additional module:

Step 1, the PKG generates two large prime numbers p and q, which make the master key, and calculates $n = pq$.

Step 2, the PKG finds large numbers e and d, which satisfies

$$ed = 1 \bmod (p-1)(q-1) .$$

Step 3, the PKG keeps d as the master key, and publishes (n, e).

B. The token issuing phase

The token issuing phase of the protocol consists of four steps:

Step 1, the authenticator collects biometrics of the user, with the help of biometrics service providers, and passes the data to the PKG.

Step 2, the PKG computes the public key a using (2).

Step 3, the PKG runs key extraction algorithm to find the private key β that satisfies the following equation:

$$a = \beta^e \pmod{n}, \quad (3)$$

which can be solved using the following equation:

$$\beta = a^d \pmod{n} .$$

Step 4, the private key β is saved in the token and issued to the user.

C. The Token Authentication Phase

The token authentication phase consists of six steps. The first five steps make a revised version of the classic challenge-response protocol, which will be proved later to be secure, regardless of how biometrics-based authentications are carried on, and whatever their results are.

Below is a description of the steps:

Step 1, the authenticator generates a random number ξ, and sends it as the challenge to the token.

Step 2, the token generates a nonce, denoted *k*, calculates the response code as $\eta = (x, y)$ using the following equations:

$$x = k^e (\bmod\ n), \quad (4)$$

$$y = \beta^{-1} k^{h(\xi,x)} (\bmod\ n), \quad (5)$$

and sends it back to the authenticator.

Step 3, the authenticator recovers *a* from ξ, x, y using the following equation:

$$a = y^{-e} x^{h(\xi,x)} (\bmod\ n), \quad (6)$$

which is a revised version of the original equation given below:

$$a y^e = x^{h(\xi,x)} (\bmod\ n).$$

Step 4, the authenticator retrieves the cipher texts of the biometrics templates associated to the given user account, decrypts them using $\lambda = h(s, a)$ as the secret key:

$$b_i = E^{-1}(\lambda, \sigma_i), \ i=1,2,\ldots t.$$

Step 5, the authenticator tests the recovered biometrics templates with (2); if the equation is not satisfied, the authenticator rejects the user and stops the protocol.

Step 6, the authenticator outputs the biometrics templates b_i, *i*=1, 2,…*t*.

Note that in the above description, we assume the account name of the user is sent to the authenticator before it starts the authentication, although one may also include this step in the protocol.

IV. Security analysis

As mentioned above, the purpose of this paper is to provide a new combination of token-based authentication with biometrics-based authentication, so that an adversary can not break into a computer system without or even with a legitimate token stolen from some user. This section conducts a study to determine whether the goal is achievable using the framework designed above. Note that under our framework biometrics-based authentications are technically decoupled from token-based authentication, but cryptographically bound to the latter. This architecture allows different authentications to be carried out separately, while ensures that none of them can be circumvented by an imposter.

In the following discussions, we first provide a formal proof for the security of token-based authentication using IRAP. Integrity and confidentiality of biometrics templates under the framework are studied next, using the result of previous study as a given condition.

A. Security of token-based authentication

As discussed above, the IRAP uses an RSA related signature scheme proposed by Shamir. Security of such schemes depends on so called RSA assumption [7], which is elaborated below for convenience of the readers:

Assumption1 (RSA assumption) Given an RSA modulus *n=pq*, an exponent *e*,

$$\gcd[e, (p-1)(q-1)] = 1, \quad (7)$$

and a random number $y \in Z_n^*$, it is hard to find $x \in Z_n^*$, such that $x^e = y$, assuming *p* and *q* are not known.

A lemma can be derived from the above assumption immediately, which is given below:

Lemma 2. Given an RSA modulus *n=pq*, an exponent *e* that satisfies (7), and a random number set $\Omega \subset Z_n^*$, if the RSA assumption holds true, and the order of Ω is polynomial, it is hard to find $x \in Z_n^*$ such that $x^e \in \Omega$, assuming *p* and *q* are not known.

In the following discussions, we borrow the concepts of "no message attack" and "adaptively chosen message attack" together with two additional lemmas, namely lemma 2 and lemma 4, from [8], and call them as the Forking Lemma and the Probability Lemma respectively. With the help of the above assumptions and lemmas, we are ready to prove the theorem given below:

Theorem 3. Token-based authentication using the IRAP is secure against no-message attacks under the random oracle model (ROM).

Proof. In order to deceive the authenticator, an imposter should generate a response code (*x*, *y*), such that the biometrics templates recovered are valid. We are going to prove that no such response code could be forged without knowing a legitimate private key. This is done by showing that a contradiction will be created should the imposter be able to accomplish the above task.

First, suppose an attacker uses a Turing machine with a random tape to forge the signature of the challenge, and an oracle to generate the hash value of the recovered templates. If it succeeds after a polynomial number of queries, the queries will make a set, denoted by

$$Q = \{u \,||\, b_1 \,||\, b_2 \ldots ||\, b_t : b_i \in B_i, i = 1,2,\ldots t\},$$

where B_i represents the set containing valid templates of respective biometrics, while the corresponding answers make another set, denoted by *A*, which contains random numbers output by the oracle as the hash values of $u \,||\, b_1 \,||\, b_2 \ldots ||\, b_t$. Using the Probability Lemma, there is a good subset of *A*, denoted Ω, if $h(u \,||\, b_1 \,||\, b_2 \ldots ||\, b_t)$ is assigned a value in Ω evenly at random, the attacker would also succeed with non-negligible probability.

Next we assume that an oracle is used to generate the hash value of (ξ, x), while the hash values of the templates are picked evenly at random from Ω. Using the Forking Lemma, a replay of this Turing machine would output two valid response codes $(x, y^{(1)})$ and $(x, y^{(2)})$ such that

$$h^{(1)}(\xi, x) \neq h^{(2)}(\xi, x),$$

while the following equations are satisfied with non-negligible probability:

$$(y^{(1)})^e a^{(1)} = x^{h^{(1)}(\xi,x)} (\text{mod } n), \quad (8)$$
$$(y^{(2)})^e a^{(2)} = x^{h^{(2)}(\xi,x)} (\text{mod } n),$$

where $a^{(1)}, a^{(2)} \in \Omega$. Note that the size of Ω is polynomial, so with non-negligible probability, we have

$$a^{(1)} = a^{(2)} = a . \quad (9)$$

Let

$$\theta_1 = h^{(1)}(\xi, x),$$
$$\theta_2 = h^{(2)}(\xi, x), \quad (10)$$
$$z_1 = (y^{(1)})^{\theta_2} (\text{mod } n),$$
$$z_2 = (y^{(2)})^{\theta_1} (\text{mod } n),$$

Combining (8), (9) and (10) gives

$$z_1^e a^{\theta_2} = z_2^e a^{\theta_1} (\text{mod } n)$$

which can be rewritten as

$$z_3{}^e = a^{\theta_2 - \theta_1} (\text{mod } n), \quad (11)$$

where

$$z_3 = z_1 z_2^{-1} (\text{mod } n) \in Z_n .$$

Also with non-negligible probability,

$$\gcd(e, \theta_1 - \theta_2) = 1,$$

so we conclude from (11) that the adversary is able to find certain $z_4 \in Z_n$ in polynomial time [6], such that

$$z_4^e (\text{mod } n) = a \in \Omega ,$$

which contradicts to lemma 2. End of the proof.

Furthermore, we also have the following theorem:

Theorem 4. Token-based authentication using the IRAP is secure against adaptively chosen message attacks under ROM, if and only if it is secure against no-message attacks.

A proof for the above theorem can be provided by imitating theorem 12 in [8]. The basic idea of which is to simulate the output of the token by a simulator who does not have the secret key, and to make the adaptively chosen message attacker and the simulator collude in order to form a no-message attacker. The only thing required to fill the gap between the original theorem and our theorem is the counterpart of lemma 11 in [8], which is given below:

Lemma 5. The output of the token in the IRAP can be simulated with an indistinguishable distribution under ROM.

Proof. Similar to the lemma 11 in [8], the key ingredient of our proof is as follows: values returned by the random oracle can be freely computed and have no correlation with the challenge code. We begin with getting a random output from the oracle for the value of $h(\xi, x)$, or h for short, which satisfies $\gcd(h, e) = 1$ with non-negligible probability. We next find $M, N \in Z_n$, such that $Mh = Ne + 1$, which can be accomplished in polynomial time, and compute x, y as $x = a^M (\text{mod } n)$, and $y = a^N (\text{mod } n)$, where a is the target public key. A simple calculation then shows that (6) is satisfied by the random oracle output h, the target public key a, and (x, y) computed above, which can not be distinguished from the output of the legitimate token of the user. That completes the proof.

B. Security of biometrics templates

We first study the integrity of the biometrics templates. Since the IRAP has been proved to be secure, whenever a token or rather the private key inside the token is authenticated by the protocol, a reliable digest is provided for the biometrics templates, which is given by

$$h(u \| b_1 \| b_2 ... \| b_t) = \beta^e = a \ (\text{mod } n) .$$

In case the templates are modified one way or another, the above equation will be violated; consequently, the token owner will be rejected in step 5 during authentication.

We next study the confidentiality of the biometrics templates. Since the authenticator keeps only the cipher texts of the biometrics templates, while the tokens supply only the information required to decrypt the cipher texts during authentication, an adversary can not get the biometrics data either by breaking into the authenticator, or by capturing the inputs and outputs of the tokens. Even if the adversary is capable of attacking in both ways, there is still the last barrier to be crossed, which is created by the system secret *s*. Unlike the cipher texts of the templates, which are stored in some databases, and kept in the hard disks, the system secret can be sealed in some tamper resistant cryptographic module, therefore, is much harder to break.

V. On biometrics-based authentication

We have completed major part of our work, which develops a framework that can enforce both token based-authentication and various biometrics-based authentications. The following are discussions on typical examples of potential service providers, the policy that should be used to summarize their authentication results, as well as the scenario where biometrics-based authentications are delegated to trusted third parties (TTPs).

A. Examples of service providers

The most practical authentication service provider is the fingerprint subsystem, which uses fine features, also known as minutiae to match fingerprints. The minutiae include ridge bifurcations and ridge terminations etc., each of which is characterized by its type, coordinates and orientation. So the parameter string would be translated into (x_j, y_j, μ_j, ν_j) , *j*=1, 2, …t, where x_j, y_j represent the *j*th location of the feature with respect to horizontal and vertical coordinates, μ_j represents its type, while ν_j represents its orientation. Typically [9], we have $x_j, y_j < 32$, $\mu_j < 4$, $\nu_j < 16$. Suppose *t*=100, i.e., there are 100 features in a fingerprint template, including false ones, they can be encoded by 1600 bits of data in total. In practice, a finger print may also be characterized by its gross features. They include the basic patterns referred to as arches, loops and whorls, which are useful for fixing the minutiae into a correct coordination. Therefore, the total size of each template is often over 3200 bits, or 400 bytes. It is obviously too big to make a public key without compression in most cryptographic systems, such as those based on integer factorization problem (IFP) or digital logarithm problem (DLP)[10]. This is another reason why (2) is used to compute the public key. Ironically, the noise that contributes to the size of the template also contributes to the entropy of which, and the increased entropy will prevent an adversary from obtaining the user's personal information out of the public key by brute force attacks.

Another potential authentication service provider to be discussed in particular is the DNA typing system [11], which can give extremely reliable and accurate authentication result, regardless of many setbacks. Theoretically, a DNA fingerprint is characterized by the pattern of short tandem repeats (STRs) at different sized alleles in highly polymorphic regions of DNA. and the template of which consists of duplets denoted by (τ_j, μ_j) , *j*=1, 2,…*t*, where *t* represents the number of alleles analyzed, which is usually between 10 and 40, τ_j identifies the locus of each allele, μ_j represents the number of its tandem repeat, which varies from 5 to 50 or more. Suppose each STR pattern is encoded into 16 bits, the size of an ideal DNA template is 16*t*, or 208 bits if *t*=13. It is so concise that the entropy contained in which may not be big enough to protect the personal information from brute force attacks. In this case, we have better combine a DNA template with one or more templates of other biometrics, such as the fingerprint minutiae, before turning it into a public key.

Note that at present it takes quite long time to test a DNA sample, and the actual DNA templates are pictorial rather than digital, if transformed into data, the size of which would be much bigger then the ideal one mentioned above. However, since biometrics authentication is well decoupled from token-based authentication in our framework, these facts will not prevent the exclusive technology from finding applications in new areas other than forensics.

B. Decision making policy

Since the authentication results of most biometrics-based authentications are probabilistic by nature, a decision making policy is necessary if multiple biometrics authentication service providers are integrated under the framework. For example, suppose there are three biometrics subsystems, an "AND" policy requires that the users pass the checks by all the biometrics subsystems, which minimizes the combined false acceptance rate (FAR) while maximizes the combined false reject rate (FRR); an "OR" policy requires that the users pass the check by only one of the biometrics subsystems; which maximizes the combined FAR while minimizes the combined FRR; a "Majority" policy requires that the users pass the checks by at least two of the biometrics subsystems, which makes a trade-off between the combined FAR and combined FRR.

As mentioned above, if an imposter acquires the token of a user illegally, and passes the token-based authentication with it, the system will depend on biometrics-based authentications to stop the imposter from going further. Since the templates used for these authentications are bound to the token, which are registered by the legitimate user, the probability that the imposter could succeed on penetrating into the system is slim if the combined FAR can be brought sufficiently close to 0.

C. One more disccusion

Given the sensitivity and exclusivity of some biometrics technologies, it is worthwhile to have one more discussion on the scenario where biometrics-based authentications are delegated to TTPs, or in another word, the biometrics service providers are moved to remote hosts, therefore have better to keep the biometrics templates on their own databases.

In order to protect the confidentiality of the templates in this scenario, we introduce the secrets of service providers, denoted s_i, *i*=1,2,…*t*, and encrypt each b_i by

$$\lambda_i = h\{s_i, h[s, h(s_i, b_i)]\}.$$

Meanwhile, an additional modification is also made on the framework by replacing (2) with

$$a = u \,||\, h(s_1, b_1) \,||\, h(s_2, b_2) ... \,||\, h(s_t, b_t).$$

When the public key is recovered during authentication, the authenticator obtains $h(s_i, b_i)$, *i*=1,2,…*t*, from which, computes each $h[s, h(s_i, b_i)]$ using the system secret, and sends it to relevant authentication service provider. The later in turn, computes λ_i using its own secret, decrypts the template, and compare it with a live biometrics sample. The authentication result is returned to the authenticator, together with the value of $h(s_i, b_i)$ computed from the decrypted template, which is used by the authenticator to verify the integrity of the biometrics data.

VI. CONCLUSUIONS

We have proposed in this paper a framework for multifactor authentication using token and biometrics. In the framework, biometrics-based authentications are technically decoupled from token-based authentication, but cryptographically bound to the latter. This architecture allows different authentications to be carried out separately, while ensures that none of them can be circumvented by an imposter.

Confidentiality of the biometrics data has brought more and more attentions recently. In our framework, it is jointly protected by the database that stores the cipher texts of the biometrics templates, the token that supply the information required to decrypt the cipher texts for each user during authentications, and the tamper resistant cryptographic module that keeps the system secret for the activation of the decryption algorithm.

We have also discussed how to move biometrics authentication service providers away from the authenticator. This is an important step toward a new configuration of the framework, where biometrics systems play the central roles, while the role of the authenticator is reduced to something like the certificate authority in a public key infrastructure, which usually runs behind the scenes. A major renovation on biometrics-based authentication could be made in this way, which further separates the usage of biometric technology from the management of biometric identities of the users.

ACKNOWLEDGMENT

The author wish to thank the reviewers for their kind advices.

This work was supported in part by the Natural Science Foundation of Fujian Province (2010J01348) and a grant from Xiamen Municipal Government.

REFERENCES

[1] International Biometric Group, Biometrics Market and Industry Report 2007-2012, Obtained through internet: http://www.biometricgroup.com, 2007

[2] A, Alterman, "A piece of yourself: Ethical issues in biometric identification", *Ethics and Information Technology*, vol. 5 (3), 2003

[3] S.Drimer, S. J Murdoch, .R. Anderson, "Thinking inside the box: system-level failures of tamper proofing", *Technical Report UCAM-CL-TR-711.*Computer Laboratory, University of Cambridge, 2008

[4] U. Uludag, S. Pankanti, S. Prabhakar et al., "Biometric cryptosystems: issues and challenges", *Proceedings of the IEEE*, vol. 92(6), pp. 948-960, 2004

[5] Y. Dodis, L. Reyzin and A. Smith, "Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data", *Proceedings from Advances in Cryptology - EuroCrypt*, 2004

[6] A Shamir, "Identity based cryptosystems and signature schemes", *LNCS 196*. New York: Springer-Verlag, pp. 47-53, 1985

[7] R. Cramer and V. Shoup, "Signature schemes based on the strong RSA assumption", *IBM Research Report RZ 3083.* 1998

[8] D. Pointcheval, and J. Stern, "Security Proofs for Signature Schemes", In *"Advances in Cryptology - Proceedings of EUROCRYPT '96"*, U. Maurer eds. LNCS 1070, Springer-Verlag, 1997, pp. 387-398

[9] N. K. Ratha, J. H. Connell, and R. M. Bolle, "Enhancing security and privacy in biometrics-based authentication systems", *IBM SYSTEMS JOURNAL*, vol. 40(3) , 2001

[10] B. Schneier, *Applied cryptography*, New York: John Wiley & Sons, 1996

[11] T. K. Lorne, *DNA Fingerprinting: An Introduction (Breakthroughs in Molecular Biology),* Oxford University Press , 1993

Jian-De Zheng received his Ph. D. from Zhejiang University in 1988. He has been a professor with School of Information Science and Technology in Xiamen University since 2003. His research interests include information security, computer network and software architecture.