Yale University
Department of Computer Science

Private Eyes: Secure Remote Biometric Authentication

Ewa Syta Michael J. Fischer David Wolinsky
Abraham Silberschatz Gina Gallegos-García
Bryan Ford

YALEU/DCS/TR-1510
March 15, 2015

Private Eyes: Secure Remote Biometric Authentication*

Ewa Syta[†] Michael J. Fischer [†] David Wolinsky[†] Abraham Silberschatz[†] Gina Gallegos-García[‡]
Bryan Ford[†]

Abstract

Authentication using biometrics in place of passwords is attractive, both for user convenience and for enhanced security. However, unlike passwords, biometric data cannot be changed, making it useless as an authentication factor if ever compromised.

We propose an efficient remote biometric authentication protocol that gives strong protection to the user's biometric data in case of two common kinds of security breaches: (1) loss or theft of the user's token (smart card, handheld device, etc.), giving the attacker full access to any secrets embedded within it; (2) total penetration of the server. Only if both client *and* server are simultaneously compromised is the user's biometric data vulnerable to exposure.

The protocol works by encrypting the user's biometric template in a special way that allows it to be used for authentication without being decrypted by either token or server. Further, the encrypted template never leaves the token, and only the server has the information that would enable it to be decrypted.

We have implemented our protocol using two iris recognition libraries and evaluated its performance. The overall efficiency and recognition performance is essentially the same compared to an unprotected biometric system.

1 Introduction

A major problem facing the Internet is "the reliance on passwords to authenticate users" [16]. The drawbacks of passwords have long been known [29]. Weak passwords are easily guessed; strong passwords are difficult for users to remember and to supply when required. In addition, username and password data must be somehow protected when sent over the network and stored on a server since once compromised, they are sufficient to impersonate the legitimate user.

Combining biometrics with cryptographic authentication schemes is an attractive alternative to password authentication [17], an approach supported by the FIDO (Fast IDentity Online) Alliance, a non-profit organization formed to promote easier to use and stronger authentication. FIDO works closely with dozens of prominent industry partners (e.g., Bank

*This material is based upon work supported by the Defense Advanced Research Agency (DARPA) and SPAWAR Systems Center Pacific, Contract No. N66001-11-C-4018.

[†]Department of Computer Science, Yale University, CT

[‡]Department of Computer Science, National Polytechnic Institute of Mexico

of America, Google, Visa, RSA) and strives to reflect the current needs and expectations of the authentication process, from clients and services alike. In their approach, a secret key, stored on the user's local device, is used with a challenge-response protocol to authenticate securely to the server. Biometrics are used to prevent the local device from being activated by any but the legitimate user. However, the user's device stores secret information, which becomes problematic when the device is compromised.

Especially sensitive in any biometric scheme is the user's biometric data which, if compromised, can subsequently be used by an attacker to impersonate the individual to whom it belongs. Unlike passwords, biometric data cannot be changed, so once compromised, it becomes useless as an authentication factor.

Many techniques exist for extracting data stored on a device's internal memory, even from so-called "tamper-resistant" devices [1, 2]. One must assume that an attacker who steals or otherwise gains physical possession of the user's device also obtains the entire contents of the device's internal memory, including any secret keys and biometric data stored therein. Therefore, the user's biometric data must not be stored on the user's device in any form that would allow an attacker to reconstruct it. Similar considerations apply to the server, which also must protect the user's biometric data even in the face of a total compromise.

2 Our Contribution

We propose an efficient remote biometric authentication protocol that gives strong protection to the user's biometric data in case of two common kinds of security breaches: a full client compromise or a full server compromise. Our scheme also allows the creation of multiple unlinkable personas in much the same way as with passwords. (See Section 8.1.)

Our two-factor protocol can be combined with a broad class of existing biometric authentication schemes to protect the privacy of the user's biometric data and the template derived from it. It works with any biometric scheme where the result of feature extraction can be represented by a binary feature vector, and the matching criterion for two feature vectors is based on their Hamming distance.[1] The user's device (or *token*) can insist that a scan of a biometric feature be performed each time before beginning an authentication round, ensuring that the presented biometric template is fresh.

The token stores only an encrypted form of the reference biometric template, which we call the *blinded template*. This keeps the biometric data safe even if the token is compromised. The encryption is performed by computing the XOR of the biometric template with a random *blinding factor* that is stored only on the server. How this is accomplished is the heart of our protocol and is described in Section 4. Since the blinding factor is random and carries no information about the actual biometric template, the biometric data is safe even if the server is compromised. Only if *both* token *and* server are simultaneously compromised is the user's biometric data vulnerable to exposure.

To authenticate, the user provides a fresh biometric sample. A new template is constructed immediately from the fresh scan and then XORed with the blinded template. The result is a blinded difference vector, that is, a vector that is the XOR of the original template,

[1] The Hamming distance between two bit vectors is the number of indices in which the two vectors differ.

the fresh template, and the blinding factor. The blinded template is then sent to the server, which unblinds it to reveal the difference vector between the two templates. The server then counts the number of "1" bits in the difference vector to obtain the Hamming distance between the two templates, which it then compares against the acceptance threshold of the underlying biometric scheme in order to establish the user's claim of identity.

Note that the blinded template itself never leaves the token. It is used only to contruct the blinded difference vector, which is sent to the server. Thus, even a compromised server that watches a valid authentication attempt can learn only the difference between the two templates; not either of the templates themselves.

The scheme just described is not secure against replay attacks. An eavesdropper who records the blinded difference vector could subsequently present it to the server and have it accepted as valid.

To overcome this problem, our protocol changes the blinding factor after each successful authentication, similar to the way a rolling code used to protect some keyless entry systems from replay attacks [44]. The idea is that both token and server use synchronized pseudorandom number generators to produce a sequence of blinding factors that changes after each authentication. Only if the current blinding factors match can the authentication succeed.

The rest of the paper is structured as follows. Section 3 summarizes other solutions to biometric authentication. Section 4 presents our protocol, and Section 5 analyzes its security properties. Section 6 discusses deployment issues. Section 7 describes a prototype implementation and a performance evaluation of our protocol. Section 8 mentions some further extensions to the protocol and Section 9 concludes.

3 Related Work

Many biometric authentication protocols offer protection of biometric data. Unlike our protocol, however, those protocols frequently protect the biometric data at the cost of a degraded recognition performance, higher complexity, or a lack of mechanisms to create unlinkable personas.

Standard cryptographic solutions for protecting passwords or other secrets, such as encryption or hashing, are difficult to use for protecting biometric templates because even if two templates are generated using two samples of the same biometric characteristic, they are never exactly the same. Homomorphic encryption [19] and secure two-party computation techniques [32] offer good security and privacy guarantees but they normally come at a high performance cost.

Biometric cryptosystems (BC), such as fuzzy extractors [14, 15], fuzzy vaults [27] and fuzzy commitments [28], use a template as well as helper data to extract a cryptographic key, with the resulting key validated by verifying its correctness. While BC offers additional features such as reliable cryptographic key generation, they come at the cost of performance and complexity. They heavily rely on error correction codes, which limits their recognition performance to the error-correcting capability of the employed code [25, 40]. Furthermore, to achieve reusability and unlinkable personas, BC schemes must be strengthened by adding auxiliary information, for example passwords [3, 36]. This adds to their complexity, limits user convenience, and in some cases may still be insufficient [22].

Other template protection schemes use a transformation function, either invertible (BioHashing [26]) or non-invertible (cancelable biometrics [39]), and apply it to biometric data during the enrollment phase. For the authentication phase, they apply the same transformation and compare the resulting template against the reference template. In case of invertible transformations, users need to supply, and therefore remember or keep secure, a password or a key, which impacts their convenience. A compromise of this additional information can yield further vulnerabilities [31, 33]. This is in contrast to our protocol where a compromise of the user's token does not expose her biometric data. In the case of non-invertible transformations, the recognition performance is affected because the matching is applied to degraded transformed templates [40]. However, unlinkable personas can be achieved [25, 35].

4 Protocol Description

In this section, we give a fuller description of the protocol that is described informally in Section 2. In particular, we describe the enrollment phase, the authentication phases, and the resynchronization method that is needed to recover from failed authentication attempts.

The authentication process is performed over a network between an authenticating party (Peggy, the user) and a verifying party (Victor, the authentication server).

The protocol requires a pseudorandom number generator $G = (m, S, \iota, \delta, \rho, n)$ whose arguments are the length of the seed m, a finite set of internal states S, an initial-state function ι which maps a seed z to a state $s_0 \in S$, a next-state function δ, and an output function ρ, which produces n-bit values. We assume that δ is a permutation on S. G is used during the protocol to generate a sequence $s_0, s_1, s_2, \ldots$ of states and a corresponding sequence $r_0, r_1, \ldots$ of pseudorandom numbers. These in turn are used to generate a sequence of blinding factors $R_0, R_1, R_2, \ldots$. More formally, $s_0 = \iota(z)$, and for each $k \geqslant 0$, $r_k = \rho(s_k)$, $R_k = \bigoplus_{j=0}^{k} r_j$, and $s_{k+1} = \delta(s_k)$.

4.1 Enrollment Phase

During the enrollment phase, Peggy and Victor cooperate to create Peggy's credentials and to establish the shared authentication information.

The public information includes the choice of a biometric characteristic, a feature extractor that produces a biometric template, an appropriate matching metric on templates, and an appropriate pseudorandom number generator G. Our protocol assumes the template is described by a Boolean vector, and the matching metric is a function of the difference between templates. It assumes that G is cryptographically secure and backtracking resistant. (See Section 5.1 for definitions.)

Since our protocol is a two-factor scheme, Peggy needs to obtain a token on which to store her blinded biometric template and the state of the pseudorandom number generator. Section 6.2 discusses the issue of tokens and obtaining them.

Peggy and Victor also need to obtain a shared secret z to be used as the seed for G. The seed needs to be established in a secure manner in order to keep the sequences of blinding factors secret and the blinded template secure. How the seed is obtained will depend on the enrollment method. With face-to-face enrollment, Victor can generate the seed and give it

to Peggy. For remote enrollment, the secret seed can be exchanged using one of the schemes to establish a shared secret, for example a key agreement protocol [21]. Alternatively, Victor can generate the seed and send it to Peggy through a secure channel.

To continue the enrollment, Peggy and Victor initialize G using seed z. Peggy obtains a biometric sample using an external sensor or a sensor built into the token depending on the kind of token she chose as described in Section 6.2. She next generates a reference template P_{ref}. She then blinds P_{ref} with the first blinding factor $R_0 = r_0$ generated using G to produce the blinded template $T_0 = P_{\text{ref}} \oplus R_0$. This process binds Peggy's biometric identity to the secret seed z established with Victor.

Victor meanwhile uses G to generate the blinding factor R_0, which he stores for future use. Both Peggy and Victor store the next state s_1 of G. Finally, Peggy securely erases her raw biometric data, the unprotected template, the secret z, the first blinding factor r_0, and the first state s_0 of G. Similarly, Victor securely erases the secret z and the first state s_0 of G.

Algorithm 1 shows the steps of the enrollment process in detail.

Algorithm 1 Enrollment Phase

1. Peggy obtains a token. Peggy and Victor agree on non-secret authentication information: the biometric recognition protocol and the choice of $G = (m, S, \iota, \delta, \rho, n)$, where m defines the length of the seed, S is the finite set of states of the generator, ι is the initial-state function, δ is the next-state function, ρ is the output function, and n is the length of biometric templates.

2. Peggy and Victor securely exchange a random seed $z \in \{0,1\}^m$.

3. Peggy and Victor both initialize their generator G to the initial state $s_0 = \iota(z)$. They use G to generate the first random number $r_0 = \rho(s_0)$ and the next state $s_1 = \delta(s_0)$ of G.

4. Peggy obtains a biometric template P_{ref}, computes the first blinding factor $R_0 = r_0$, and creates a blinded template $T_0 = P_{\text{ref}} \oplus R_0$, where $\oplus$ is the bit-wise exclusive-OR operation. She securely erases z, P_{ref}, R_0, r_0, and s_0. She keeps T_0 and s_1 on her token.

5. Victor computes the first blinding factor $R_0 = r_0$. He securely erases z, r_0, and s_0. He retains R_0 and s_1 in private storage.

To summarize, after the enrollment phase:

- Peggy's token stores $T_0 = P_{\text{ref}} \oplus R_0$ and s_1.
- Victor retains $R_0 = r_0$ and s_1.

4.2 Authentication Phase

Peggy and Victor run the authentication phase each time Peggy wishes to prove her identity to Victor. We number the authentication phases in sequence, starting with 1, and we denote the current phase number by k. Neither Peggy nor Victor need to know the current phase number in order to carry out the protocol, but we use the phase number to distinguish the values available to Peggy and Victor during the phase. Thus, at the start of phase k,

- Peggy's token stores $T_{k-1} = P_{\text{ref}} \oplus R_{k-1}$ and s_k.
- Victor stores $R_{k-1} = \bigoplus_{j=0}^{k-1} r_j$ and s_k.

In order to authenticate, Peggy obtains a fresh biometric sample and generates a new template P_k from it. She then calculates an authentication message

$$W_k = P_k \oplus T_{k-1} = (P_k \oplus P_{\text{ref}}) \oplus R_{k-1}.$$

Peggy sends W_i to Victor for verification.

Without waiting for a response from Victor, she immediately updates her token. She uses G to compute $r_k = \rho(s_k)$ and $s_{k+1} = \delta(s_k)$. She then computes

$$T_k = T_{k-1} \oplus r_k = P_{\text{ref}} \oplus R_{k-1} \oplus r_k = P_{\text{ref}} \oplus R_k.$$

Finally, she securely replaces T_{k-1} with T_k and s_k with s_{k+1}, and she securely erases W_k and all other temporary data from memory. By updating after each authentication attempt, successful or not, she ensures that the same blinding factor is never used more than once.[2]

Upon receiving W_k from Peggy, Victor removes the blinding factor R_{k-1} to obtain the difference vector $V_k = P_k \oplus P_{\text{ref}}$. He applies the matching metric of the underlying biometric system to V_k in order to decide whether or not to accept Peggy's authentication attempt as valid.

If valid, he updates his stored information. Using G, he computes $r_k = \rho(s_k)$ and $s_{k+1} = \delta(s_k)$. He then computes $R_k = R_{k-1} \oplus r_k$. Finally, he securely replaces R_{k-1} with R_k and s_k with s_{k+1}, and he securely erases all temporary data from memory.

4.3 Resynchronization

A legitimate authentication attempt might fail for many reasons, for example, because Peggy's message never reaches Victor, or because of poor feature extraction by Peggy, or because of Victor's not storing the updated blinding factor before going offline, or because of intentional malicious authentication attempts by an adversary. In such cases, Peggy advances her generator but Victor does not. This will leave Victor unable to unblind Peggy's future messages.

Peggy's and Victor's generators must be resynchronized in order for authentications to continue. Because Peggy updates her blinded template T_k after each authentication attempt and Victor does so only after a successful authentication, if the generators are out

[2]This prevents an attacker from obtaining useful information from differencing two authentication messages W_i and W_j. If they both used the same blinding factor T, the blinding factor would cancel out, and an attacker could compute $W_i \oplus W_j = (P_i \oplus T) \oplus (P_j \oplus T) = P_i \oplus P_j$.

of sync, Peggy's generator will be ahead of Victor's. A simple solution is for Victor to search forward in the sequence produced by G for some limited predefined distance looking for a blinding factor $R_{k'}$ that leads to a successful authentication using Peggy's current authentication message W_k. After finding the correct value of T_k, both generators will again be in sync. Such a scheme is called a rolling code and is widely used in keyless entry systems [44]. Alternatively, Peggy and Victor can both keep track of the current stage of their generators, and Peggy can send it to Victor along with her authentication message. Both of these schemes can be exploited by an adversary if the distance that Victor is allowed to advance during resynchronization is not reasonably limited.

Algorithm 2 shows the steps of the authentication process in detail.

Algorithm 2 Authentication Phase

1. Peggy obtains a biometric sample and generates a fresh biometric template P_k.
2. Peggy calculates $W_k = P_k \oplus T_{k-1}$ and sends W_i to Victor.
3. Peggy uses G to compute $r_k = \rho(s_k)$ and $s_{k+1} = \delta(s_k)$. She then computes $T_k = T_{k-1} \oplus r_k$. She securely replaces T_{k-1} on her token with T_k and s_k with s_{k+1}, and she securely erases P_k, W_k, r_k, and s_k from her token.
4. Victor, upon receiving W_k, computes the difference vector $V_k = W_k \oplus R_k$. He passes V_k to the matching algorithm and accepts Peggy's authentication attempt if the match is sufficiently good.
5. If the authentication attempt succeeds, Victor uses G to compute $r_k = \rho(s_k)$ and $s_{k+1} = \delta(s_k)$. He then computes $R_k = R_{k-1} \oplus r_k$. He securely replaces R_{k-1} with R_k and s_k with s_{k+1} in memory, and he securely erases r_k, and s_k.

To summarize, at the end of authentication phase k:

- Peggy's token stores $T_k = P_{\text{ref}} \oplus R_k$ and s_{k+1}.
- Victor retains $R_k = \oplus_{j=0}^{k} r_j$ and s_{k+1}.

5 Security Properties

Our main goal and concern is the security of biometric data, not only under normal use of the protocol, but also in case of complete compromise of either Peggy's token or Victor's entire internal state. In addition, our protocol prevents an attacker who compromises Peggy's token from impersonating her to Victor.

We note that if an attacker compromises both Peggy and Victor, then Peggy's biometric template is easily obtained. Peggy's token contains her blinded template; Victor has the blinding factor. It is needed to unblind the difference vector, but it will also unblind the template stored on the token.

5.1 Assumptions

Peggy and Victor interact over a network, possibly in the presence of a computationally bounded adversary (Mallory, the malicious adversary). In addition to eavesdropping on all communication between Peggy and Victor, we assume that Mallory can talk directly to Victor in an attempt to impersonate Peggy. In addition, Mallory might actively attack either Peggy or Victor but not both.

In an attack on Peggy, Mallory takes possession of Peggy's token and obtains access to all of the data stored on it. Peggy detects the attack since her token is physically gone. Nevertheless, our protocol guarantees that Peggy's biometric data remains secret and Mallory cannot impersonate Peggy to Victor.

In an attack on Victor, Mallory compromises Victor and gains access to all of his data. Victor does not necessarily detect the intrusion. He continues processing authentication requests, and Mallory sees everything that happens on the server. In this case, Peggy's biometric data still remains secret, but Mallory can easily impersonate Peggy to Victor. This can be prevented by using digital signatures as described in Section 8.2.

We assume that all communication occurs over an unsecured channel, so after k authentication attempts, Mallory knows the authentication messages $W_1, \ldots, W_k$, which are blinded differences between pairs of biometric templates. Thus, Mallory has the following information.

$$\begin{aligned}
W_1 &= P_1 \oplus T_0 && = P_1 \oplus P_{\text{ref}} \oplus r_0 \\
W_2 &= P_2 \oplus T_1 && = P_2 \oplus P_{\text{ref}} \oplus r_0 \oplus r_1 \\
W_3 &= P_3 \oplus T_2 && = P_3 \oplus P_{\text{ref}} \oplus r_0 \oplus r_1 \oplus r_2 \\
&\cdots && \cdots \\
W_k &= P_k \oplus T_{k-1} && = P_k \oplus P_{\text{ref}} \oplus r_0 \oplus \cdots \oplus r_{k-1}
\end{aligned}$$

Furthermore, we assume that the sensor Peggy uses to obtain biometric samples does not directly reveal her biometric data to Mallory, prior to his possible compromise of Peggy's token. We also assume that Peggy does not use her token after it has been compromised. Similarly, we assume that the communication channel between the sensor and the token is trusted.

The security of our protocol depends critically on the pseudorandom number generator G, which we assume is cryptographically secure and backtracking resistant. We also assume that the secret seed z and the unprotected reference template P_{ref} are securely erased after enrollment and are not available to Mallory.

To be *cryptographically secure* means that the sequence of outputs are computationally indistinguishable from a similar sequence of truly random numbers. The notion of computational indistinguishability, introduced by Yao [45], means that any probabilistic polynomial-time algorithm behaves essentially the same whether supplied with pseudorandom inputs or truly random inputs. See Goldreich [20] for further details.

To be *backtracking resistant* means that it is not feasible to run G backwards from a given state to recover previously-generated values.[3] More formally, it means that the sequence $r_0, \ldots, r_k$ is computationally indistinguishable from a truly random sequence of the same form, where the distinguishing judge also has access to s_{k+1}. Cryptographically strong

[3]Note that the previous values are well defined since we assume the next-state function δ is a permutation on S. Among other things, backtracking resistance implies that δ is a one-way permutation.

pseudorandom number generators that are resistant to a previous-outputs backtracking attack exist and are proven to be secure [4].

For our protocol, backtracking resistance protects the biometric reference template from an attack where Mallory obtains the token and has access to the blinded reference template T_k and the state s_{k+1} of G. Backtracking resistance prevents Mallory from running the generator backwards from s_{k+1} to obtain $r_k, r_{k-1}, \ldots, r_0$ from which the blinding factor R_k and the reference template $P_{\text{ref}} = T_k \oplus R_k$ could be computed. The same protection holds even if Mallory has prior knowledge of $r_0, r_1, \ldots, r_{k-1}$ (which she might) but not r_k.

5.2 Security of Biometric Templates

5.2.1 Mallory compromises Victor

We assume that Mallory can compromise Victor at any time and remain undetected. If she compromises him at phase k, she learns the current blinding factor R_k and the next state of the generator s_{k+1}. This enables her to compute the future random numbers $r_{k+1}, r_{k+2}, \ldots$ and the future blinding factors $R_{k+1}, R_{k+2}, \ldots$. Clearly, she learns the most by compromising Victor at the very beginning, in which case she recovers the exact same information that Victor receives from Peggy. From this, she can learn all of the difference vectors, $V_1, V_2, \ldots$.

The usefulness of the difference vectors depends on the underlying feature extractor. Ideally, we want a feature extractor that produces templates on repeated scans of the same biometric that lead to small false rejection rates. When the match function is based on the Hamming distance between two templates, small false rejection rates imply that most difference vectors approximate the zero vector. Hence, Mallory only learns vectors in the neighborhood of 0. In any case, we can say that Mallory has no other information about the template since Peggy sends nothing else besides the blinded difference vectors.

5.2.2 Mallory compromises Peggy

When Mallory obtains access to Peggy's token, she learns the current blinded reference template

$$T_k = P \oplus r_0 \oplus r_1 \oplus \cdots \oplus r_{k-1} \oplus r_k$$

and the next state of the generator s_{k+1}. This new information is in addition to all authentication messages $W_1, \ldots, W_k$ sent up to that point which we assume she already knew.

T_k looks random to Mallory because of the blinding factor r_k, which she does not know. It was securely erased from the token when T_k was updated, and it was never included in any of the messages sent. Additionally, r_k cannot be recovered using the stored state s_{k+1} of G and $r_0, \ldots, r_{k-1}$ (assuming they are known) since G is backtracking resistant and cryptographically secure. Thus, Mallory cannot obtain any information about the blinding factor R_k, so neither T_k nor s_{k+1} give Mallory any information about P_{ref}.

We assume that Peggy's token is not compromised at the moment she is using it, since for a brief interval, the token contains her unprotected biometric template as well as data from both stage $k-1$ and stage k.

5.3 Impersonation

5.3.1 Mallory compromises Victor

As before, when Mallory compromises Victor, she gets R_k and s_{k+1}. R_k is the blinding factor needed to unblind the next authentication message. Therefore, Mallory might be able to prepare a fake message W'_k so that verification will succeed from Victor's point of view. Section 8.2 discusses a practical defense against this attack.

5.3.2 Mallory compromises Peggy

As before, when Mallory compromises Peggy, she gets T_k and s_{k+1}. We argued in Section 5.2.2 that her compromise of Peggy's token does not give her any information about P_{ref} or R_k. Hence, she gets no information that would allow her to impersonate Peggy.

5.4 Leakage of Information

During each authentication attempt, Victor receives a difference between two biometric templates. If he has been compromised, then Mallory also receives this information. Unlike the case of a compromise of Peggy, we assume that the compromise of Victor might be undetected so that Mallory can collect data over time from legitimate authentication requests.

After a number of such authentications, Mallory has a set of differences between Peggy's templates. Those differences are binary vectors of differences between Peggy's reference template and the sample template used on a given authentication. A difference bit of 1 indicates a discrepancy between the reference and the sample templates.

The frequency of 1's in any given bit position represents the unreliability in that position of the template. A low-frequency position indicates a reliable bit; a high frequency position means that little useful information is being carried by that bit. Mallory can compute these frequencies and thereby learn about the reliability of each bit in the template. What information these frequencies carry about the actual reference template or Peggy's raw biometric data depends in detailed properties of the sensor as well as the feature exaction algorithm. Although analyzing this kind of information leakage for any particular sensor and feature extractor is beyond the scope of this paper, it is well to keep in mind this possibility in designing biometric systems.

Low-frequency bits can arise equally well from 0's in both reference and sample templates or from 1's in both, so knowing that it is low frequency says little about the actual template bit. With a good feature extractor, we expect most difference bits to be low-frequency, so information leakage would seem to be minimized with good quality biometric systems. Section 8.3 discusses an approach to further minimize any information leakage.

6 Usage Considerations

6.1 Suitable Biometrics

There are two main categories of biometric characteristics used in biometric systems: physiological (e.g., a fingerprint or iris pattern) and behavioral (e.g., voice print or signature).

Characteristics must be universal (everyone has it), unique (different for every person), permanent (it does not change with time), and collectable (it can be quantitatively measured) [10]. In practice, fingerprints, face geometry, and iris patterns have been popular choices as they can be obtained easily and non-intrusively using a simple camera. Fingerprints tend to be prone to spoofing, however, and the accuracy of facial recognition may be impacted by pose, expression, or lighting [12,24]. An iris, on the other hand, exhibits many highly desirable properties. Its pattern varies greatly among different people, even identical twins, and persists over a lifetime. Iris-based recognition systems have been widely deployed by many organizations including British Telecom, Panasonic, LG and IBM Schiphol Group [7,11].

For these reasons, we chose to use an iris-based template for our implementation, described in Section 7. These templates typically consist of 2048 bits to represent the iris pattern with any bit equally likely to be either 1 or 0. On average half of all the bits will disagree between the templates of two different people. A study [11] based on 9.1 million comparisons between different pairings of iris images concluded that it is extremely improbable that two different irises might disagree in fewer than a third of all bits. Consequently, if a difference score is less than 0.32, then a positive match is statistically guaranteed.

Difference Score	False Match
0.26	1 in 10^{13}
0.27	1 in 10^{12}
0.28	1 in 10^{11}
0.29	1 in 13 billion
0.30	1 in 1.5 billion
0.31	1 in 185 million
0.32	1 in 26 million
0.33	1 in 4 million
0.34	1 in 690,000
0.35	1 in 133,000

Table 1: The relation between the difference score and odds of a false match [11]

An iris-based template encodes an iris pattern as a binary vector. Fingerprint templates use the fingerprint texture as a real-valued fixed length vector. Finally face-based templates use facial features represented again as a real-valued fixed length vector. A match can be performed by calculating the Hamming distance (or alternatively a fractional Hamming distance) for binary vectors, while a Euclidian distance for real-valued vectors with the points defined by the set difference. Our current protocol assumes binary biometric templates, which are suitable for all iris-based templates, however, a binarization technique [38] can convert other types of templates into a binary vector [8,30,38] and make it usable in our protocol but likely trading off some recognition performance for the ability to use diverse types of templates. Another promising approach is the use of private set intersection techniques, as briefly outlined in Section 8.3.

6.2 Enrollment Process and Tokens

The main drawback of possession-based authentication is the need to obtain and manage tokens. Eddie, an enrolling agent, can be responsible for issuing tokens and performing the enrollment phase, ensuring a successful bond between a token and a biometric identity. Depending on the application-specific security requirements, Eddie can be an independent, trusted enrollment center, Victor can assume Eddie's role, or Eddie's role can be delegated to users. In the first scenario, Eddie's services can be offered by an organization such as VeriSign [43]. This approach would provide a good way to issue and manage a variety of tokens. VeriSign already provides similar services and issues security credentials (VIP Security Token or Card [42]).

Tokens and central databases are two most popular locations for storing biometric templates [24,37]. A token allows users to physically secure their biometric templates and gives them a sense of control over their personal data. However, issues arise when tokens are lost or stolen. A central database makes it possible for users to authenticate from multiple locations easily as templates are constantly available for verification. On the other hand, the database may become a target of attacks because of its valuable content and central storage of templates raises privacy concerns because all authentication attempts go through a single point.

Our protocol has been designed with security of biometric data in mind, we use tokens to store biometric templates but ensure that their content is protected in case of loss or theft.

There are two different approaches to utilizing tokens depending on the token's ability to obtain biometric samples.

Using a token with a built-in sensor removes security concerns related to the sensor and the communication channel between the token and the sensor. Mobile devices are an obvious choice for such tokens; they are equipped with a high resolution camera capable of capturing images suitable for authentication using several biometric characteristics such as a fingerprint, facial geometry, or iris pattern [24]. Additionally, mobile devices make it possible to take advantage of less frequently utilized characteristics like voiceprint, keystroke or handwriting patterns, service utilization [9] or even gait [13].

A token without a sensor is only used to store authentication information and to perform computations. It must be paired with an external sensor to obtain a biometric sample. This implies certain level of trust that the sensor is not compromised and the channel between the token and sensor is secure. However, such tokens are inexpensive and make it possible to utilize virtually any biometric characteristic. Smart cards are the most obvious choice for such tokens. They have been extensively used for authentication as they offer enough computational power and are relatively cheap, small, and convenient to use [41].

6.3 Template Generation

A biometric template is a representation of the features from a biometric sample. A feature extractor is a component of a biometric system responsible for generating templates. During the feature extraction process, key features of the biometric sample are located, selected, measured, encoded and then stored in form of a template. The template quality directly impacts the performance of a biometric system. We require that a feature extractor

produces templates of high quality. More specifically, we assume that two templates created based on a biometric sample from the same user are "sufficiently" similar to be suitable for authentication purposes. Similarly, we require that two templates created based on biometric samples from different users are "sufficiently" different. The goal is to achieve an acceptable false rejection rate (FRR) and more importantly a low false acceptance rate (FAR).

6.4 Verification Decision

In biometric systems, a matcher and decision module are the two components directly involved in making the verification decision.

A matcher takes two biometric templates, the reference template created in the enrollment phase and the freshly obtain template from the verification phase, as input. Then, it calculates a *match* score which shows how similar the two templates are [24]. In case of our protocol, the matcher functionality is embedded into the protocol. The verifying party calculates $V_i = T_{s_{i-1}} \oplus W_i$ which defines $\Delta(P, P'_i)$, the difference between two biometric templates. Therefore, the value of V_i defines the *difference score*. In other biometric authentication protocols, the authentication decision is based on the match score while in our it is based on the difference score. To express the difference score in terms of the match score we can say that the smaller the difference V_i, the higher the match score is.

The decision module takes a match score (in our case a difference score) as input and based on a predefined threshold parameter τ decides whether the two templates were created based on biometric samples from the same person. If the match score is greater than a predefined threshold τ, user's identity if verified. In our protocol, if the difference score if lower than τ, then the two templates are accepted as coming from the same user.

Choosing a proper value for τ is a challenging task. To have a high level of confidence that two templates were created based on samples from the same user, the difference should be very low. Hence, the value chosen for τ should reflect the desired level of security as well as the sensor and feature extractor's capabilities to create accurate templates. The goal is to balance the false rejection and false acceptance rates while ensuring a proper level of security.

7 Evaluation

We evaluate our prototype implementation to observe the performance characteristics of our protocol in comparison to using unprotected templates. We then analyze the behavior of the feature extraction libraries to determine their usefulness and potential overheads in this scheme. We used the CASIA Iris Image Database [23] as input into our system.

7.1 Implementation Details

We have implemented our biometric authentication system in C++ using the Qt framework and Crypto++ cryptographic libraries. For feature extraction, we have employed two different iris recognition libraries: Project Iris [6] and Libor Masek's Iris Recognition [34],

both of which utilize John Daugman's approach [11] to produce an iris template. Project Iris uses C++ and the Qt framework; however, for Masek's library, we constructed a C++ to Octave[4] interface. In evaluation figures, we denote Project Iris [6] feature extraction library as C++ and Masek's library [34] as Octave.

We ran the evaluations on a workstation computer equipped with an Intel Core i7-2600 processor with 4 cores, 16 GB of memory, and a Crucial 256GB SSD hard drive. Our software ran in single threaded mode and never exceeded 12 MB of used memory, the typical amount for an application utilizing Qt. This shows a very modest memory requirement of our implementation.

We use a Diffie-Hellman Key Agreement [21] to agree on a common key. We then use the agreed on key to seed a provably secure Blum-Blum-Shub generator [5] (PRBG). We used a SQLite database to store enrollment information. We set a minimal difference score of 0.32 to ensure a low probability of a false match (1 in 26 million) [11]. In their evaluations, Masek's scheme uses a modified hamming distance scheme that depends on a comparison between two unencrypted templates. Our system encrypts the templates, making use of this scheme incompatible and hence we use traditional hamming distance.

The two feature extraction libraries we employ use a technique to further boost the recognition performance. A single template may need to be rotated up to 8° in both directions to achieve better results. In an unprotected biometric system, the server can manipulate the template itself because it gets full access to the client's biometric template. In our system, we preprocess templates to produce these rotations and store the resulting blinded templates on the client's side. We do so because the server never gets access to unprotected biometric data and therefore cannot manipulate the templates itself. Then, during authentication, the client uses all of the saved templates as input to the protocol and forwards the result to the server. Therefore, our protocol preserves the same recognition performance as schemes employing this recognition-enhancing technique.

All CASIA database images have been converted to gray scale images. CASIA database version 1 contains 108 individuals with 7 images each. Project Iris only supports version 1 of the CASIA database, in which images have been preprocessed by replacing the pupils with a black (constant intensity) circle. Masek's iris recognition library handles CASIA database version 2, however, had trouble parsing approximately 4% of the images, though had no issue in database version 1. The libraries also differ in the resolution of their extracted features. While Project Iris extracts a 2048-bit template like Daugman [11], Masek extracts a 9600-bit template.

7.2 System Performance

To evaluate the enrollment phase, we created a client (Peggy) for each image in the CASIA database version 1 and used a single server (Victor). Clients, in no particular order, enrolled one after another. The enrollment occurred within the same process, as a result the evaluation focuses on data processing and message serialization, i.e., CPU time. Our results can be found in Figure 1.

The clients enrollment time includes both the initial enrollment request and the subsequent processing for a successful enrollment, both represented as a single, summed value.

[4]Open-source Matlab compatible system

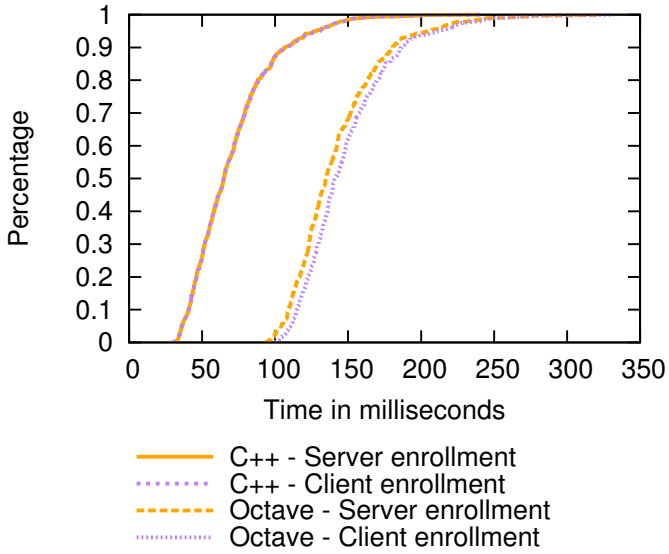

Figure 1: CPU time for enrollment

The client enrollment time is negligibly larger than the server enrollment time. The major factor in performance appears to be the size of the stored template(s) and the need to generate an appropriate amount of random blinding factors. While Octave, Masek's library, uses 17 9600-bit templates with 17 masks resulting in 40.8 KBs of PRNG work, C++, Project Iris, uses only 8.7 KBs.

To evaluate authentication time, we had each image in the database tested against every client for a total of 571,536 authentication attempts or 756 attempts per client. We separated the results, in Figure 2, into valid and invalid client and server authentications, those that our system processed, and compared them against the time a traditional template comparison would take.

7.3 Feature Extraction Reliability

To evaluate the ability of the readily available feature extraction libraries, we computed the difference scores for two images extracted from the same individual as well as different individuals and then processed them using our system. The results, as expected, were identical, though the time to do so was different, as shown earlier in Figure 2. Therefore, in this section, the evaluation primarily focuses on the recognition performance of the feature extraction libraries, as shown in Figure 3.

While both libraries were able to identify common individuals relatively easily (low false rejection rate—FRR), we were surprised to see different participants had such low difference scores in both systems, in particular in Masek's. Regardless of the reason for the discrepancy, we are satisfied that our system works equally well as the underlying feature extraction and unprotected matching scheme.

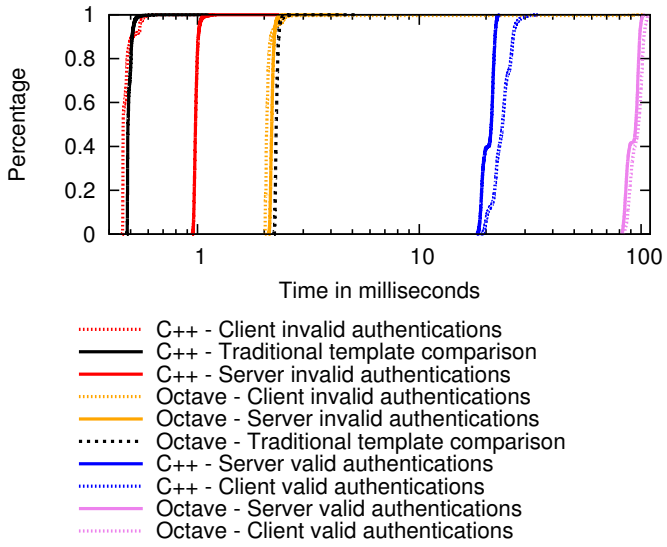

Figure 2: CPU time for authentication

7.4 Feature Extraction Timing

While our system has good response time on the orders of 10s to 100s of milliseconds, depending on the template size, we discovered that feature extraction has significant performance overheads as shown in Figure 4. This, along with the ability to capture the image, plays a critical role during the clients' authentication process.

On average, the Project Iris processes images in less than 2 seconds; however, the Masek's library took an order of magnitude longer. Considering these costs, our protocol has nearly negligible overhead in comparison to the cost of feature extraction alone. Therefore, the performance of our system is very comparable to an authentication system that offers no biometric data protection.

8 Extensions

In this section we describe three possible extensions of our main protocol. First, we discuss how our protocol can be used to establish independent identities (personas). Second, we explore using digital signatures as a defense against a possible impersonation attack in case of a server compromise. Finally, we discuss further minimizing information leakage during the verification process through the use of privacy-preserving private set intersection techniques.

8.1 Personas

In case of password authentication, during enrollment, the user creates a username and password, with the username as the unique user identifier and the password as the authentication factor. To authenticate, the user presents his credentials and the server verifies

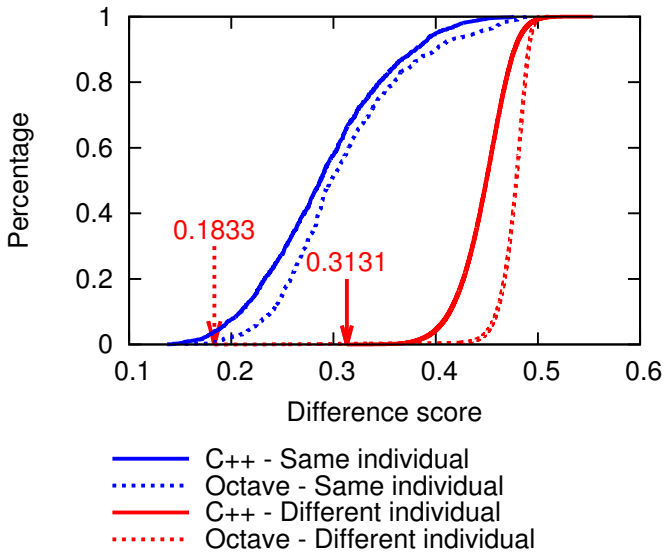

Figure 3: Difference scores using different libraries

that they match with the data presented at enrollment time. Passwords permit the user to create multiple unlinkable personas, or identities, that be used with different services. One simply chooses a different username and password for each service.

Many biometric authentication protocols create user's authentication credentials directly based on a biometric template. As a result, if a user chooses to enroll with multiple verifying parties using the same biometrics, then these verifying parties can identify the user and successfully track his activities performed using the same biometric credentials or credentials based on the same biometric features.

In our protocol, credentials are based on a user's unprotected biometric template but with respect to the blinding factors known to the verifying party. Therefore, a user can create multiple, fully independent personas. Each persona is based on the same biometric data but on a different secret shared with a verifying party. Therefore, a persona represents a user's unique identity as seen by the verifying party. Users can create different personas to deal with multiple verifying parties, or use personas for different transactions with the same verifying party. This creates a separation and *unlinkability* of biometric identities and transactions performed using those identities. The user must perform the enrollment process once for each persona, choosing a new secret for each. Policies and procedures controlling the enrollment process would determine how many and what types of personas a user may acquire.

8.2 Digital Signatures

We consider the previously described case when Mallory compromises Victor and therefore obtains his entire secret state. Specifically, Mallory gets the sequence of all blinding factors needed to unblind Peggy's next authentication message. This gives Mallory enough

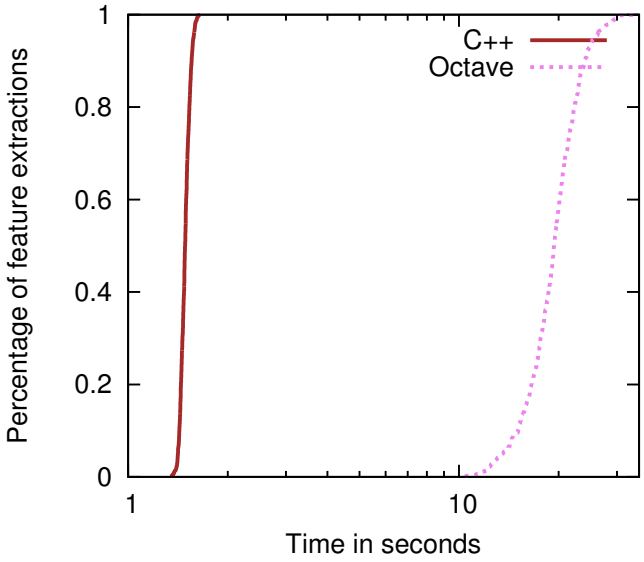

Figure 4: Time for feature extraction

information to prepare a message on Peggy's behalf and effectively impersonate her.

This attack is easily prevented by the use of digital signatures. During the enrollment phase, Victor obtains Peggy's public verification key K_{pub} which corresponds to a private signing key K_{sec}. During the authentication phase, Peggy signs her authentication message W_i using K_{sec}. Victor only processes messages that are properly signed. If Mallory chooses to send an authentication message without a valid signature, Victor will reject it and decline Mallory's request since Mallory cannot produce valid signatures without K_{sec}.

8.3 Private Set Intersection

Our protocol allows Victor to make an authentication decision based on a match function calculated from the difference vector between Peggy's reference template and a fresh template. Peggy securely computes the blinded difference vector and sends it to Victor who decrypts it and uses to make an authentication decision. This approach leaks the additional information of which positions of the difference vector differ—information that could conceivably be useful to an attacker.

We consider the use of private set intersection (PSI) techniques [18] for directly calculating the output of the match function. Private set intersection techniques allow two parties holding private input sets to calculate the intersection of their sets (elements they have in common), or the intersection set cardinality (the number of common elements), or simply whether the intersection set cardinality exceeds a fixed threshold. All this can be done without revealing anything else about their private inputs.

These techniques are directly applicable to our verification process. We envision using the PSI cardinality protocol [18] enriched with an encoding mechanism to calculate the difference score without giving Victor access to the binary difference vector. Using a PSI

protocol would offer a greater protection against even minimal information leakage but it would come at a computational cost at least $\mathcal{O}(k \log \log k)$ and communication cost $\mathcal{O}(k)$, where k is the size of the template in bits. Trading performance for enhanced protection may be justified in situations which require preventing even minimal information leakage.

9 Conclusions

Protecting sensitive biometric data is crucially important for remote biometric authentication, because once compromised, the biometric data becomes no longer useful for distinguishing between its legitimate owner and anyone else who happens to possess a copy.

We present a new method for adding strong biometric data protection to a wide class of existing biometric authentication protocols, making them an attractive alternative to password authentication. In particular, biometric data is never stored on the server, only on the user's token, and then only in encrypted form. The token itself requires no secure storage; the biometric data cannot be recovered even if an attacker has full and complete access to everything stored on the token. A lost token also cannot be used to impersonate its owner.

Our method is computationally efficient and has the same recognition performance as the underlying feature extraction scheme. It also allows the creation of independent personas provide enhanced privacy of users' actions across different verifying parties.

References

[1] R. Anderson and M. Kuhn. Tamper resistance-a cautionary note. In *Proceedings of the second Usenix workshop on electronic commerce*, volume 2, 1996.

[2] R. Anderson and M. Kuhn. Low cost attacks on tamper resistant devices. In *Security Protocols*, 1998.

[3] L. Ballard, S. Kamara, F. Monrose, and M. K. Reiter. Towards practical biometric key generation with randomized biometric templates. In *Conference on Computer and Communications Security*, 2008.

[4] M. Bellare and B. Yee. Forward-security in private-key cryptography. In *Topics in Cryptology - CT RSA*, 2003.

[5] L. Blum, M. Blum, and M. Shub. A simple unpredictable pseudo random number generator. *SIAM J. Comput.*, 1986.

[6] M. Boyd, D. Carmaciu, F. Giannaros, T. Payne, and W. Snell. Iris recognition (project iris). `http://projectiris.co.uk/`, March 2010.

[7] M. Boyd, D. Carmaciu, F. Giannaros, T. Payne, W. Snell, and D. Gillies. Iris recognition. Technical report, Department of Computer Science, Imperial College London, 2010.

[8] C. Chen and R. Veldhuis. Binary biometric representation through pairwise polar quantization. In *Advances in Biometrics*. Springer Verlag, 2009.

[9] N. Clarke and S. Furnell. Authentication of users on mobile telephones - a survey of attitudes and practices. *Computers and Security*, 2005.

[10] R. Clarke. Human identification in information systems: Management challenges and public policy issues. *Information Technology & People*, 1994.

[11] J. Daugman. How iris recognition works. *IEEE Trans. on Circuits and Systems for Video Technology*, 2002.

[12] K. Delac and M. Grgic. A survey of biometric recognition methods. In *International Symposium on Electronics in Marine*, 2004.

[13] M. Derawi, C. Nickel, P. Bours, and C. Busch. Unobtrusive user-authentication on mobile phones using biometric gait recognition. In *Intelligent Information Hiding and Multimedia Signal Processing*, 2010.

[14] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 2008.

[15] Y. Dodis, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *EUROCRYPT*, 2004.

[16] FIDO alliance: Mission. `http://fidoalliance.org/about`, September 2014.

[17] FIDO alliance: Specifications overview. `http://fidoalliance.org/specifications`, September 2014.

[18] M. J. Freedman, K. Nissim, and B. Pinkas. Efficient private matching and set intersection. In *EUROCRYPT*, 2004.

[19] C. Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009.

[20] O. Goldreich. *Foundations of Cryptography: Volume 1, Basic Tools*. Cambridge University Press, 2001.

[21] S. Goldwasser and M. Bellare. Lecture notes in cryptography, 2001.

[22] S. Hong, W. Jeon, S. Kim, D. Won, and C. Park. The vulnerabilities analysis of fuzzy vault using password. In *International Conference on Future Generation Communication and Networking*, 2008.

[23] Institute of Automation, Chinese Academy of Sciences CASIA. Iris image databases. `http://www.cbsr.ia.ac.cn/english/IrisDatabase.asp`, 11 2012.

[24] A. Jain, A. Ross, and K. Nandakumar. *Introduction to Biometrics*. Springer, 2011.

[25] A. K. Jain, K. Nandakumar, and A. Nagar. Biometric template security. *EURASIP J. Adv. Signal Process*, 2008.

[26] A. Jin, D. Ling, and A. Goh. Biohashing: two factor authentication featuring fingerprint data and tokenised random number. *Pattern Recognition*, 2004.

[27] A. Juels and M. Sudan. A fuzzy vault scheme. *Designs, Codes and Cryptography*, 2006.

[28] A. Juels and M. Wattenberg. A fuzzy commitment scheme. In *ACM Conference on Computer and Communications Security*, 1999.

[29] G. C. Kessler. Passwords – strengths and weaknesses. `http://www.garykessler.net/library/password.html`, January 1996.

[30] T. Kevenaar, G. Schrijen, M. van der Veen, A. Akkermans, and F. Zuo. Face recognition with renewable and privacy preserving binary templates. In *IEEE Workshop on Automatic Identification Advanced Technologies*, 2005.

[31] A. Kong, K.-H. Cheung, D. Zhang, M. Kamel, and J. You. An analysis of biohashing and its variants. *Pattern Recognition*, 2006.

[32] Y. Lindell and B. Pinkas. An efficient protocol for secure two-party computation in the presence of malicious adversaries. In *Advances in Cryptology-EUROCRYPT 2007*. Springer, 2007.

[33] A. Lumini and L. Nanni. An improved biohashing for human authentication. *Pattern Recognition*, 2007.

[34] L. Masek and P. Kovesi. Matlab source code for a biometric identification system based on iris patterns. Technical report, University of Western Australia, 2003.

[35] A. Nagar. *Biometric Template Security*. Dissertation, Michigan State University, 2012.

[36] K. Nandakumar, A. Nagar, and A. K. Jain. Hardening fingerprint fuzzy vault using password. In *International Conference on Advances in Biometrics*, 2007.

[37] A. Patric. Usability and acceptability of biometric security systems. In *Proceedings of the Financial Cryptography Conference (FC04)*, 2004.

[38] S. Rane, A. Nagar, and A. Vetro. Method and system for binarization of biometric data, 2010. US Patent App. 12/688,089.

[39] N. K. Ratha, J. H. Connell, and R. M. Bolle. Enhancing security and privacy in biometrics-based authentication systems. *IBM Syst. J.*, 2001.

[40] C. Rathgeb and A. Uhl. A survey on biometric cryptosystems and cancelable biometrics. *EURASIP Journal on Information Security*, 2011.

[41] Smart cards and biometrics. A Smart Card Alliance Physical Access Council White Paper, March 2011. Publication Number: PAC-11002.

[42] VeriSign. Symantec Corporation. Verisign ID protection center: Validation & ID protection (VIP). `http://idprotect.verisign.com`, November 2012.

[43] VeriSign. Symantec Corporation web site. `http://www.verisign.com/`, November 2012.

[44] Wikipedia. Rolling code. `http://en.wikipedia.org/wiki/Rolling_code`, 2015.

[45] A. C. Yao. Theory and application of trapdoor functions. In *Annual Symposium on Foundations of Computer Science*, 1982.