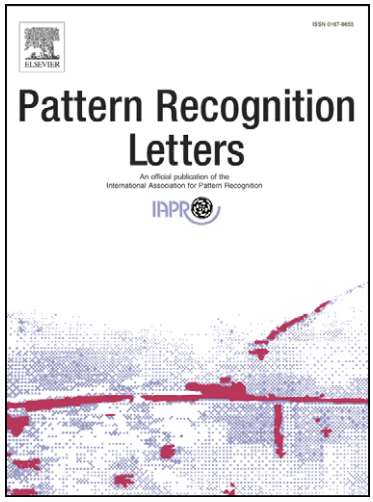


Robust Iris Verification for Key Management

Sheikh Ziauddin, Matthew N. Dailey

PII: S0167-8655(09)00370-5
DOI: 10.1016/j.patrec.2009.12.028
Reference: PATREC 4764

To appear in: *Pattern Recognition Letters*

Received Date: 22 August 2008
Revised Date: 7 November 2009
Accepted Date: 27 December 2009

Please cite this article as: Ziauddin, S., Dailey, M.N., Robust Iris Verification for Key Management, *Pattern Recognition Letters* (2009), doi: 10.1016/j.patrec.2009.12.028

Robust Iris Verification for Key Management

Sheikh Ziauddin * and Matthew N. Dailey

Computer Science and Information Management
Asian Institute of Technology
P.O. Box 4, Klong Luang, Pathumthani 12120 Thailand.

Abstract

In this paper, we present a method for secret key generation and recovery based on iris verification that, unlike typical systems which store iris templates persistently, stores recovery information on a smart card carried by the user. The scheme uses error-correcting codes to overcome the noisiness inherent in biometric readings. Our iris template incorporates reliable region selection, reliable bit estimation and one-sided masking techniques that provide for both robust verification and large key sizes. An implementation and experiments with the University of Bath iris dataset indicate that our scheme provides for generation and recovery of 260-bit keys with an EER of 0%.

Key words: iris verification, biometric authentication, error-correcting codes, reliable bit selection, one-sided masking, key management, biometric key generation

1 Introduction

Modern cryptosystems require the use of one or more large uniform random secret keys to achieve their security goals. Since it is impossible in practice to remember large keys, users typically record them on paper or store them on an easily accessible file system. In the first case, stealing a key may simply require opening a drawer or sneaking into an office, and in the second case, if the system storing a user's keys is compromised, all of the user's keys are compromised. To mitigate risk in the latter case, some key management systems store keys on a file system but encrypt them with a key derived from a user-selected

* Corresponding author.
Email addresses: zia.uddin@ait.ac.th (Sheikh Ziauddin), mdailey@cs.ait.ac.th (Matthew N. Dailey).

passphrase. Unfortunately, typical passphrases are short and may be susceptible to dictionary attacks, in which attackers repeatedly attempt decryption using the entries from a list of common passphrases. Further compounding the problem, users oftentimes reuse the same passphrase for multiple keys, so an attacker that succeeds in obtaining one passphrase may be able to compromise multiple systems. When its keys are easy to steal, any security guarantees offered by a cryptosystem are null and void.

We avoid the problem of dictionary attacks against passphrase-encrypted secret key stores using *biometric authentication and access control*. The idea is to authenticate the user using an iris scan and to only provide access to the stored key if the biometric authentication is successful. This solution has several benefits over passphrase-based systems:

- Biometric traits cannot easily be stolen, forged, or guessed.
- There is no need to remember one's biometric traits.
- Biometrics are difficult to repudiate.
- In contrast to user-selected passphrases, the entropy of a biometric template is comparable from person to person.

But systems based on biometrics have their own set of problems. The first problem is the *noisiness* of biometric scans. Readings from the same user will almost always be different; for example, independent readings of the same person's iris typically differ by 10–20% (Daugman, 2003). Consider a naive design in which we encrypt the secret with the user's enrollment-time biometric template, then decrypt the secret with the user's verification-time biometric template. With noisy biometric scans, this scheme will not work. A biometric-based key management scheme should allow a user to unlock his or her secret so long as the biometric scans presented for locking (enrollment) and unlocking (verification) are sufficiently similar.

The second problem with biometrics is that they are *irrevocable*. Losing a biometric template is akin to losing one's identity. Consider another naive design in which we first compare the verification-time biometric template with the enrollment-time biometric template, which is stored in the clear. If the two templates are sufficiently similar, we return the stored secret. This scheme is extremely dangerous. Because of the irrevocability of biometric information, an ideal biometric-based key management scheme should avoid storing biometric templates in the clear.

On the one hand, the first constraint, that we must allow for noisy matching at verification time, would be satisfied if we could store the enrollment-time template in the clear and compare it with the verification-time template. However, as discussed, it is not appropriate to store biometric templates in the clear. On the other hand, the second constraint, that we not store biometric templates

in the clear, would be satisfied if we could, as in many password authentication systems, store the hash of the scan presented at enrollment time and compare it with the hash of the scan presented at verification time. However, since each independent biometric scan is in general different, we cannot compare hashed templates.

In this paper, we present and empirically evaluate a scheme that solves both problems, using noisy biometrics for secret key management without persistent storage of biometric templates. After providing some background on previous work in this area, we introduce our scheme.

1.1 Related Work

There has been considerable research in recent years on the use of biometric information to reliably generate (not necessarily cryptographic-quality) keys that can be put to use for authentication, access control, and private communication. They fall into two groups.

The first group of approaches generate keys *directly* from the biometric measurements. Among these approaches, Tomko et al. (1997) were the first to use biometric information for cryptographic key generation. Their system generates a key directly from a fingerprint reading. As a more recent example, Goh and Ngo (2003) generate cryptographic keys by discretizing random projections of images of the user's face. Since these schemes derive keys directly from the biometric measurements, their main limitation is that they do not provide any assurance about the cryptographic quality of the generated key.

The second group of schemes recognize that uniform random keys should be generated using standard cryptographic methods. These schemes allow the use of arbitrary keys at enrollment time then use a verification-time biometric scan and some form of error correction in the key recovery process. Monrose et al. (1999) introduce the concept of hardened passwords, which combine users' passwords with information about their keystroke patterns (e.g. durations of and time intervals between keystrokes) that must be reproduced at login time to gain access to a system. In a follow up paper, Monrose et al. (2001) use similar techniques to combine passwords with information about users' voices, which are recorded as they speak their password. The hardened password approach allows for storage and recovery of an arbitrary secret, but the main limitation of the existing schemes is the relatively low entropy of the biometric information used to recover the secret (approximately 15 bits for Monrose et al. (1999) and 46 bits for Monrose et al. (2001)).

Beyond hardened passwords, all of the recent biometric key management schemes published thus far make use of the seminal theoretical contributions

of the fuzzy commitment technique by Juels and Wattenberg (1999) and the fuzzy vault technique by Juels and Sudan (2002). Uludag et al. (2005) implement a fuzzy vault using fingerprint biometrics. Their system requires images that are aligned by a human expert. It allows a 128-bit key size with a FRR of 21%. Hao et al. (2006) present a smart card-based system using fuzzy commitment and iris scans. For error correction, they use a concatenated coding scheme comprised of Reed-Solomon and Hadamard codes. They allow key sizes up to 140 bits with a FRR of 0.45%. Santos et al. (2006) give a handwritten signature-based implementation of the fuzzy vault scheme. They perform automatic alignment of signatures (in contrast to the manual alignment of minutiae points by Uludag et al. (2005)). The system allows 128-bit keys with a FRR of 57%. The FAR is 1.18% for skilled forgeries and 0.32% for random forgeries. In a more recent work, Yang and Verbauwhede (2007) present a system based on iris scans and fuzzy commitment. They allow 92-bit keys with a FRR of 0.8%.

1.2 Our Scheme

Although modern key management schemes based on iris scanning achieve impressively low error rates, they are still limited by short key lengths. For example, Advanced Encryption Standard (NIST, 2001), the international standard cipher, provides for key sizes of 128, 192 or 256 bits. Yang and Verbauwhede (2007) generate 92-bit keys, which are too short for use with AES. Though Hao et al. (2006) produce 140-bit keys, which are sufficient for AES-128, they are not long enough for AES-192 or AES-256. More importantly, in the case of smart card theft, these systems' *effective* key sizes are much lower than their actual key length, making brute force attacks nearly tractable. This vulnerability will be further discussed in section 6.

Why is it difficult to build biometric key management systems that allow storage of long keys? For schemes based on error-correcting codes, the answer lies in an intriguing set of tradeoffs between error correction capability, key size, and biometric verification accuracy. We will discuss these issues in detail in the rest of this paper, but intuitively, regardless of the code, as the key size grows, the error correction region for each codeword necessarily shrinks, requiring a less noisy biometric to maintain the same biometric accuracy.

To address this problem, we propose a method for cryptographic key management using biometrics that achieves substantially greater key lengths than previous systems without compromising biometric verification accuracy. Our scheme relies first on an error-correcting code that allows fine-tuning of the key size in relation to the error correction capability and second on new methods for reducing biometric noise that include reliable region selection, reliable

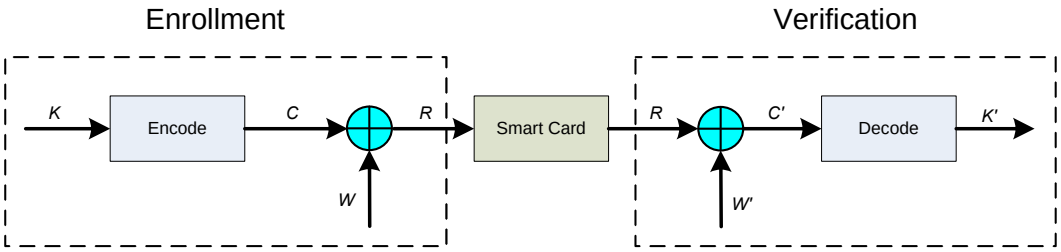


Fig. 1. Overview of proposed system. K is the secret key, W and W' are biometric readings at enrollment and verification times respectively. R represents secret recovery string. K and K' are equal if W and W' are from the same person.

bit selection and one-sided masking. The combined scheme allows the use of 260-bit keys with no errors on the University of Bath iris data set (University of Bath, 2004).

The system generates a secret key using traditional cryptographic methods and then encodes it using the user's iris template. The key and the template are bound monolithically in such a way that it is difficult to retrieve either the key or the template from the bound value. We encode the key using a *Bose-Chaudhuri-Hocquenghem (BCH)* code and this encoded key is XORed with the biometric template as suggested by the *fuzzy commitment* scheme of Juels and Wattenberg (1999). We store the output on a smart card, and at verification time, we use the data stored on the smart card to recover the original key. Figure 1 gives an overview of the system.

In our scheme, obtaining long keys without sacrificing verification accuracy means that biometric noise must be minimized. As in many pattern recognition problems, feature selection can reduce the effects of biometric noise and increase biometric verification accuracy. Many systems thus mask out corrupted or unreliable regions of the biometric pattern at enrollment as well as verification time. But since BCH codes require full-length codewords for decoding, we cannot apply this approach directly. We therefore introduce one-sided masking, in which masks are calculated and incorporated for enrollment images but not for verification images.

The rest of this paper is organized as follows. In Section 2, we introduce readers to error correction coding, provide an overview of BCH codes, and describe the fuzzy commitment scheme of Juels and Wattenberg (1999). The proposed system is presented in Section 3. Section 4 describes an experimental evaluation of our scheme. Section 5 gives a comparison of our scheme with existing work. In Section 6, we analyze the security of our scheme. In Section 7, we introduce a slight modification of the proposed scheme for use on more challenging datasets. The practical implementation issues are discussed in Section 8. Finally, Section 9 concludes the paper and discusses directions for further research.

2 Background

2.1 Error-Correcting Codes for Biometrics

Error-correcting codes are widely used in the field of telecommunications. Their role is normally to correct errors introduced during transmission over a noisy channel. Researchers in biometrics and cryptography have suggested the use of error-correcting codes for overcoming the noisiness typical of biometric data. Biometric readings acquired from the same user at enrollment and verification time can be treated as data transmitted and received, respectively, over a noisy channel. Depending on the characteristics of the noise for a particular biometric measurement, error-correcting codes may be useful for eliminating the differences between enrollment and verification readings.

In coding theory, the original data to be transmitted is called an *information word* or a *message*. Before transmission, some redundancy is added to each information word, resulting in a larger *codeword*. This redundant data helps in reconstructing the transmitted codeword from the received codeword. Each error-correcting code has a bound on the number of errors it can correct in a received codeword. This bound is a code's *error correction capability* or *error correction distance*. Any block code can be described by a tuple (n, k, t), where n is the size of each codeword, k is the size of each information word, and t is the error correction capability of the code.

Bose-Chaudhuri-Hocquenghem (BCH) codes form an important class of cyclic block error-correcting codes that can correct multiple random errors in a received codeword. BCH codes are among the most widely used block codes because their algebraic structure makes encoding and decoding simple. Efficient decoding algorithms exist for BCH codes; the most popular is the *Berlekamp-Massey algorithm*. BCH codes exist for a wide range of parameters, and they typically perform better than other block codes when the code rate k/n is large. For a detailed treatment of error-correcting codes, there are many good texts, e.g., Berlekamp (1984); Blahut (1983); Lin and Costello (1983).

2.2 Fuzzy Commitment

The *fuzzy commitment* scheme Juels and Wattenberg (1999) is an error-tolerant version of conventional cryptographic bit commitment schemes. In a bit commitment scheme, a sender possesses a secret bit and generates an encryption or *commitment* of that bit using a key called a *witness*. The receiver can retrieve the bit from the commitment (*open* it) using the witness. A cryptographic bit commitment scheme is *concealing* if it is difficult for the

receiver to open the commitment without the witness. A scheme is *binding* if it is difficult to provide a witness that, when paired with a commitment, allows opening to a bit different from the originally committed bit.

Fuzzy commitment differs from the conventional cryptographic bit commitment in two ways:

(1) The scheme commits a binary *string* instead of a single bit.
(2) The commitment can be opened with a witness that is *sufficiently close* to the original witness rather than identical to it.

The fuzzy commitment scheme contains two functions: the commitment function F and the decommitment function f. The commitment function is defined by

$$F(c, x) = (h(c), (c \oplus x)) = (h(c), \delta)$$

where $c \in \{0,1\}^n$ is a randomly selected codeword from an error-correcting code (secret), $x \in \{0,1\}^n$ is the biometric reading of the user (witness), $h(c)$ is the hashed value of the secret obtained through some hash function $h : \{0,1\}^n \mapsto \{0,1\}^l$ and δ is the commitment of the secret. The decommitment function is defined by

$$f(x', \delta) = f(x' \oplus \delta) = f(c \oplus (x \oplus x') = c'$$

where x' is the new witness. The decommitment is successful if the values $h(c)$ and $h(c')$ are equal.

3 Proposed System

We propose a system for biometric key management where we avoid cleartext storage of both keys and biometric templates. Instead we generate secret recovery information and store it on a smart card. In our system, the user has two secrets, 1) the secret recovery information stored on the smart card, and 2) a biometric secret. This makes it difficult for an adversary to obtain the user's key unless he or she acquires both of these secrets.

3.1 Biometric Enrollment Process

During the enrollment phase, four important procedures are performed. First, we generate iris templates using existing template generation methods. Sec-

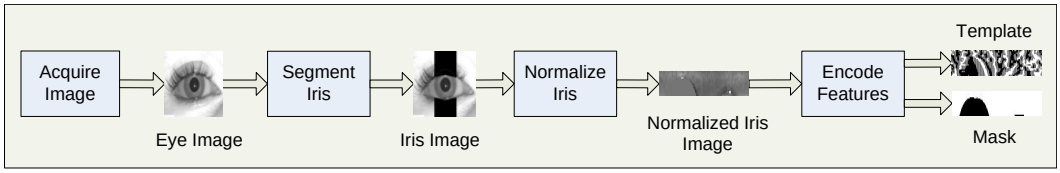


Fig. 2. Iris template generation process. The eye image is from the Bath dataset.

ond, we mask out those regions of the template that correspond to unreliable parts of the iris image. Third, we analyze the reliability of the individual bits in the template. Fourth, we generate a key using existing cryptographic techniques then encode it using a BCH code, the iris templates, and the reliable bit information. We describe each of these procedures in turn.

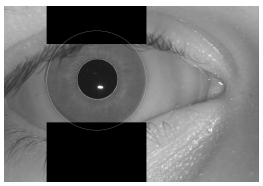
3.1.1 Iris Template Generation

Our scheme uses iris scans to encode keys. In an iris recognition system, the user presents his or her eye to an iris sensor, which images the user's iris and generates a template from this image. Most iris scanners use near infrared illumination with normal monochrome CMOS or CCD camera sensors that are sensitive to near infrared light.

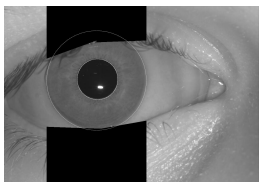
After image acquisition, we use Masek and Kovesi's algorithm (Masek, 2003; Masek and Kovesi, 2003) for iris template generation with some modifications. The algorithm is based primarily on the methods given by Daugman (2003). Generating an iris template from a raw eye image involves three steps, namely iris segmentation, iris normalization, and iris feature encoding. Figure 2 shows the iris template generation process schematically.

The goal of the iris segmentation procedure is to segment the annular iris region from the rest of the image. First, it finds the circular inner and outer boundaries (the iris-pupil and iris-sclera boundaries) of the iris using a circular Hough transform. Then it marks the region of the annular iris ring that is not visible due to eyelids and eyelashes. Masek and Kovesi's method uses a linear Hough transform to find the eyelids. They find the single best-fitting lines for the upper and lower eyelids. We, on the other hand, find two lines for each of the upper and lower eyelids. We first find the area of interest for the upper eyelid and then divide it horizontally into two equal-width parts. For each of these parts, we find the best-fitting line and join the two lines. We repeat the same process for the bottom eyelid. Figure 3 illustrates the two methods on the same iris image. Eyelashes covering the iris region are detected using a simple thresholding technique.

The iris normalization procedure transforms the segmented iris region into a rectangular shape with fixed dimensions. Iris images taken under different conditions can have different sizes due to differences in camera focal length,



(a) Masek and Kovesi's method



(b) Proposed method

Fig. 3. Eyelid detection.

distance to the subject, contraction or expansion of the pupil, or orientation of the eye with respect to the camera. To eliminate the effects of these factors, iris normalization transforms the iris annular region to a rectangular grid of fixed size using a polar to Cartesian transformation and bilinear interpolation. In our work, we use a rectangular grid of 240×20 pixels. Since the subsequent steps described below include convolution with Gaussian and log-Gabor kernels, to minimize the effect of detected corrupted regions on convolution results for neighboring valid pixels, we set the intensity of corrupted regions to the neighborhood's mean intensity.

The feature encoding procedure encodes the normalized iris image as a binary string. To encode the normalized iris image, we first apply Gaussian blur. We find that blurring the image at this stage reduces intra-class as well as inter-class Hamming distances. This is a desirable property in biometric key generation systems because lower error correction thresholds mean larger key sizes. At each pixel in the 240×20 image, the feature encoding procedure extracts information about local texture characteristics as described by convolution with a horizontal one-dimensional log-Gabor wavelet filter [1] whose real and imaginary components form a quadrature pair. Field (1987) describes the frequency response of a log-Gabor function as

$$G(f) = \exp\left(\frac{-(\log(f/f_0))^2}{2(\log(\sigma/f_0))^2}\right)$$

where f_0 is the base frequency and σ is the bandwidth of the filter. Figure 4 shows an example of a 1D log-Gabor wavelet filter.

Following Daugman (2003), the method discards the amplitude of the complex-valued response and quantizes the phase so that only its quadrant in the complex plane is retained, using a two-bit grey code. The result is a 9600-bit template ($240 \times 20 \times 2$), which is combined with the corrupted bit mask

[1] Log-Gabor filters are Gaussian on a logarithmic spatial frequency scale whereas Gabor filters (Daugman, 1985) are Gaussian on a linear spatial frequency scale (Daugman, 1985; Field, 1987). Masek uses the log-Gabor filter mainly to eliminate the Gabor filter's DC response.

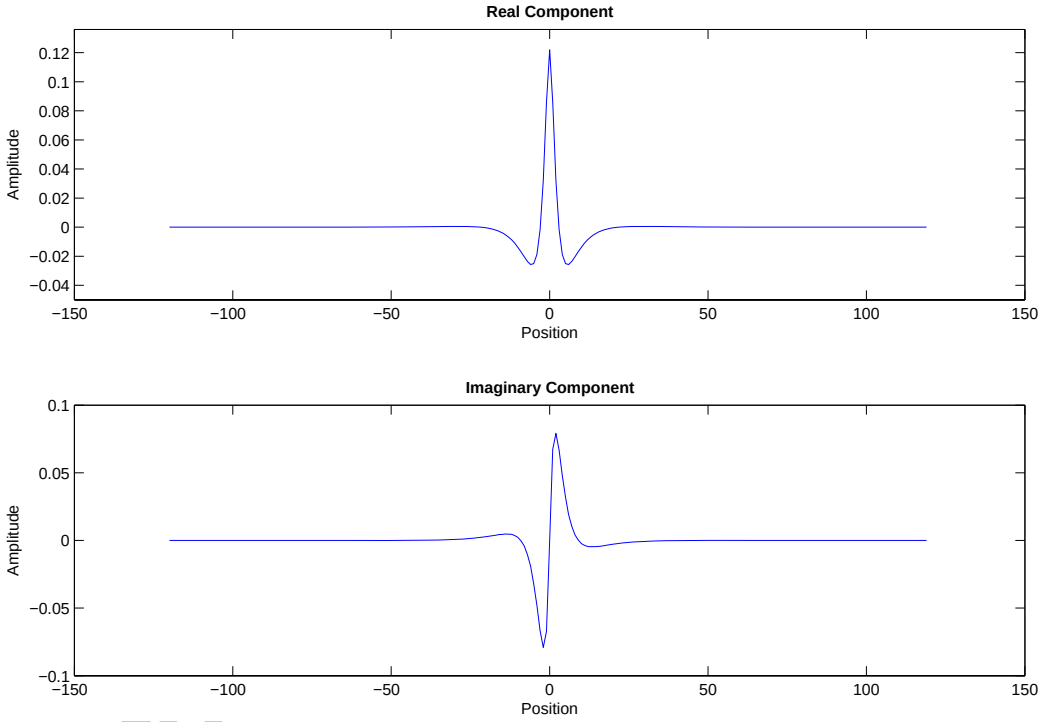


Fig. 4. Real and imaginary components of a 1D log-Gabor wavelet.

computed in the segmentation stage.

3.1.2 Iris Template Comparison

For iris template matching, the most common metric used is the *Hamming* metric. The normalized Hamming distance between two binary strings is the number of positions in which the strings differ from each other divided by the size of each string (Daugman, 2003):

$$HD(T_1, T_2) = \frac{\|T_1 \oplus T_2\|_1}{9600}, \tag{1}$$

where T_1 and T_2 represent the two templates to be matched and $\|\cdot\|_1$ represents the L_1 norm.

When information about masked (corrupted) bits is incorporated, the Hamming distance excludes bits in the template that are marked as corrupted by the corresponding mask:

$$HD(T_1, M_1, T_2, M_2) = \frac{\|(T_1 \oplus T_2) \cap M_1 \cap M_2\|_1}{\|M_1 \cap M_2\|_1}, \tag{2}$$

where T_1 and T_2 represent the two templates to be matched, M_1 and M_2 represent corresponding masks, and $\|\cdot\|_1$ represents the L_1 norm. Our system does not incorporate the Hamming metric directly, but as we shall see in Section 4.4, it is straightforward to relate a particular chosen BCH code's error correction capability to a threshold on the Hamming distance between enrollment and verification templates.

3.1.3 Selection of Reliable Regions

Some regions of a given person's iris tend to be more reliable than others for template generation. Less reliable regions include those that are close to the pupil-iris boundary and the iris-sclera boundary. These regions are unreliable due to imperfect detection of the inner and outer iris circles. Iris regions close to the upper and lower eyelids are also less reliable because they are often occluded by eyelids and eyelashes. During the reliable region selection phase, we add these less reliable regions to the corrupt bit mask previously calculated during the template generation stage. Figure 5 shows an image with less reliable regions marked.

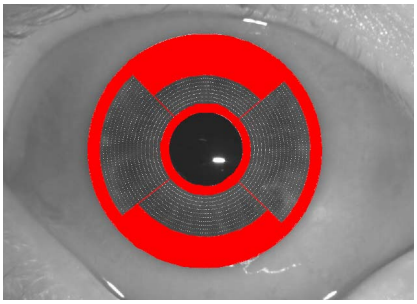


Fig. 5. Reliable region selection. Unreliable regions are masked out and shown in red. Regions close to the pupil circle, iris circle, upper eyelid and lower eyelid are not reliable.

3.1.4 Selection of Reliable Bits

When a user wishes to enroll in the system, he or she presents n independent iris scans. In our experiments, we use $n = 3$. For each iris scan, we generate a 9600-bit template and mask as described in Sections 3.1.1 and 3.1.3. We call these n templates *base templates*.

To derive a unified *final template* W, we first determine which bits are *reliable* in that they are both unmasked (not corrupted) and equal over all n base templates. Yang and Verbauwhede (2007) use a similar technique without considering any corrupt bit information. We found that the recognition performance is significantly improved when masking information is incorporated in reliable bit selection. Among these reliable bits, we select the first 4095 bits for the final template. In addition to final template, we create a 9600-bit *flag vector* F indicating the positions of these reliable bits.

After the system creates a user's final template W and flag vector F, it stores the flag along with an encoded version of the final template (see next section) on the user's smart card. The reliable bit selection process is illustrated in Figure 6.

3.1.5 Key encoding using BCH and the iris template

As previously discussed, our scheme provides for generation and recovery of 260-bit keys. At enrollment time, we pick 260 bits at random as the user's secret key K. We then compute the hash $H \leftarrow H(K)$, where $H()$ is a collision-resistant hash function, and store H on the user's smart card. H is used to verify K when it is later regenerated.

In parallel, we compute $C \leftarrow E(K)$, where $E()$ is a $(4095, 260, 696)$-BCH encoder. This BCH code has an information word size of 260 bits, a codeword size of 4095 bits, and an error correction distance of 696 bits. We XOR the 4095-bit encoded value C with the final template W resulting in a recovery

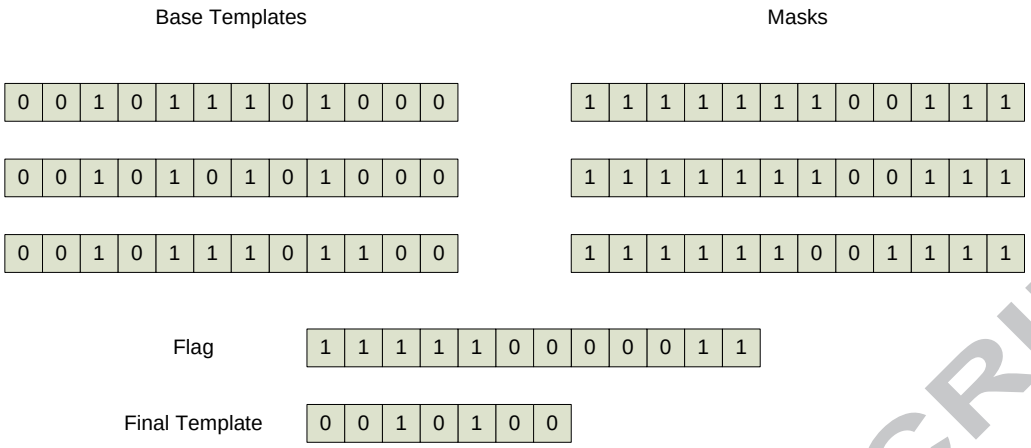


Fig. 6. Generation of final template. For purpose of illustration, we use 12-bit templates. There are 10 bits that are identical in all three base templates. Among these 10 bits, 3 are corrupted (as indicated by a value of 0 in the mask) in at least one of the scans. Therefore the final template contains 7 reliable bits and the flag specifies the locations of these reliable bits.

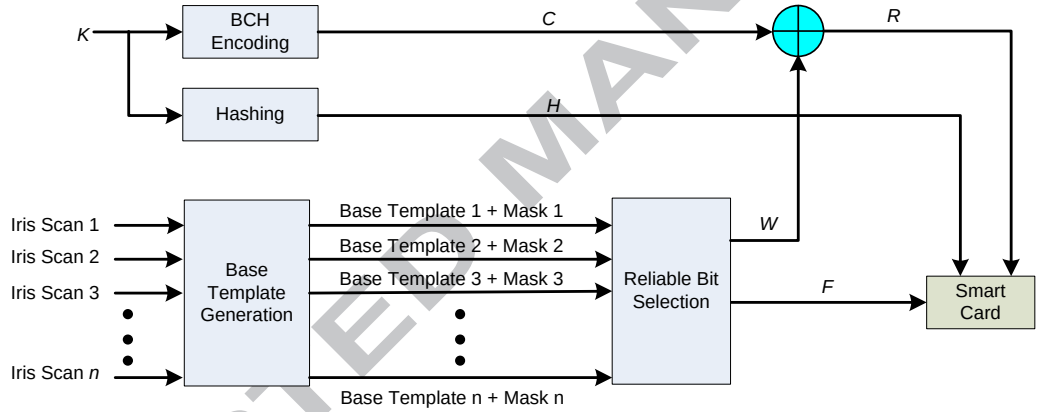


Fig. 7. Enrollment process. The user's secret key K is hashed to obtain H and encoded to obtain C. C is XORed with iris template W to obtain recovery information R. H and R are stored on the user's smart card with reliable bit flag vector F.

string R that is used to regenerate K at verification time using a fresh iris scan. The resulting string R is stored on the smart card along with F and H. Figure 7 shows the entire enrollment process schematically.

3.2 Biometric Verification Process

The verification process is in some ways similar to the enrollment process. The user presents his or her eye to the iris sensor along with his or her smart card. We generate a verification-time iris template W' from the user's iris image. Then we XOR W' with the recovery information R, decode it to get a

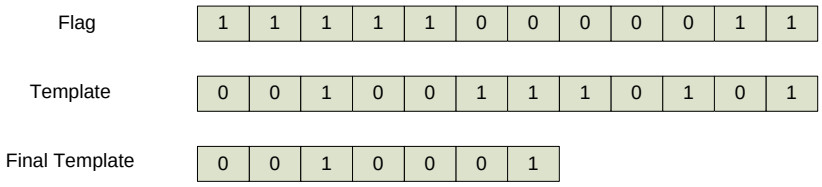


Fig. 8. Reliable bit selection during verification. We copy bits from the base template to the final template W' if the corresponding flag bit is 1.

candidate key K', and finally verify that $K' = K$ using the stored hash H.

To compute W', we first generate a base template from the user's iris scan using the procedure described in Section 3.1.1. However, the reliable bit selection process is necessarily different from the process used for enrollment. The user only presents one template, and no mask is generated. The reason we do not use the verification-time mask is that the recovery string R stored on the smart card was generated using the enrollment-time reliable bits and cannot be changed at verification time. Therefore, to align the verification template bits with the bits of R, we simply select 4095 reliable bits from the 9600-bit verification-time base template as indicated by the flag vector F stored on the smart card. The procedure for verification-time reliable bit selection is demonstrated in Figure 8.

To recover the user's secret key, we XOR W' with R to get the string C' and compute $K' \leftarrow D(C')$, where $D()$ is a $(4095, 260, 696)$-BCH decoder. We next compute $H' \leftarrow H(K')$. Finally, we compare H' with the enrollment-time hash H stored on the smart card. If $H' = H$, the user is accepted; otherwise he or she is rejected. If the user is accepted, K' can be used in the targeted cryptosystem. It must be noted that head tilt and camera angle often cause rotational inconsistencies. To overcome this problem, we perform 8 rotations of 2 bits each in the clockwise as well as the counterclockwise direction on the verification-time iris template and, each time, try to generate the hashed secret H after performing reliable bit selection, BCH decoding and hashing. If there is no rotation for which $H = H'$, we reject the user.

Since our BCH code is able to correct up to 696 corrupted bits, and the iris template W is 4095 bits, the error correction capability of our system is 17%. This means that if the Hamming distance between two iris templates is less than 17%, our system will treat the two templates as belonging to the same user. If the Hamming distance between two templates is more than 17%, the system will be unable to recover K from C' and will reject the user as an intruder. Figure 9 shows the verification process schematically.

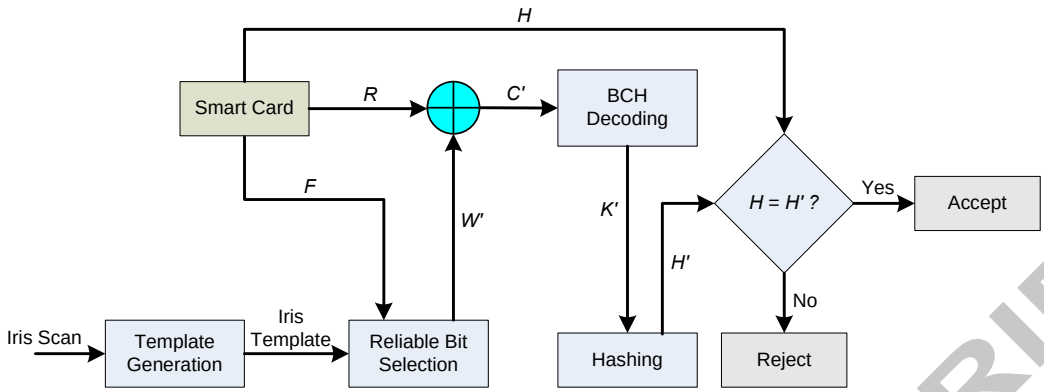


Fig. 9. Verification process. The reliable bit flag vector F, recovery information R, and hashed key H are stored on the user's smart card. W' is constructed from the user's iris scan and F. W' is XORed with R to obtain codeword C'. The recovered key K' is hashed to obtain H', which is compared to H for verification. The user is accepted and K' is usable if $H' = H$.

4 Experimental Evaluation

4.1 Iris Database

We performed an experiment using the University of Bath iris dataset (University of Bath, 2004) to evaluate the key size and biometric accuracy of our scheme. The free version of the University of Bath iris dataset consists of 1000 high resolution iris images acquired from 25 subjects. There are 40 images for each subject, taken in one session, 20 each for both left and right eyes. We tested our algorithm on the right eye images of all 25 subjects. For each subject, we used the first three images for training and the remaining 17 images for testing. Therefore our training set contains 75 images while our testing set has 425 images.

4.2 Base Templates

We ran Masek and Kovesi's segmentation algorithm (Masek and Kovesi, 2003) on each of our 500 Bath images. The algorithm accurately located the pupil and iris in 81% of the images. We manually located the pupil and iris in the remaining images then performed the rest of the base template extraction procedure as described in Section 3.1.1.

4.3 Final Enrollment Templates

We first computed base templates for each of the images. Then we performed reliable bit selection using each subject's 3-image training set as described in Section 3.1.4. This resulted in generation of a final template along with a flag vector for each of the 25 Bath subjects.

4.4 Verification Results

We performed all 425 possible within-class comparisons and all 10,200 possible between-class comparisons using the methods described in Section 3.2. We observed a perfect recognition rate of 100% with no false positives or false negatives.

Traditional biometric authentication systems based on iris templates compare enrollment-time and verification-time templates using Hamming distance. By varying the Hamming distance threshold at which two templates are deemed from the same person, the tradeoff between false accept rates and false reject rates can be manipulated.

In our scheme, on the other hand, we verify by generating a candidate key K', hashing it to obtain H', and comparing it with the hashed key H stored on the user's smart card, rather than comparing the enrollment and verification templates directly. However, H and H' will be equal whenever the Hamming distance between the enrollment template W and the verification template W' is below a threshold determined by the error correction capability of the specific BCH code we use.

Table 1 shows how the effective Hamming distance threshold can be manipulated by varying the error correction capability of the BCH code. The table shows that the scheme is flexible enough to allow a number of practical thresholds.

5 Comparison with Existing Systems

The results reported in the previous section compare favorably with the best existing systems. Ours is the first biometric key management scheme allowing key lengths long enough for 256-bit AES. It is the first system that achieves an EER of 0%, and the results reported here form the first experimental evaluation of a biometric key management scheme published using a public iris dataset. In contrast to previous systems, it also achieves a higher level of

Error Correction Capability (Bits)	HD Threshold	Key Size (Bits)	FRR %	FAR %
573	0.14	322	1.88	0
614	0.15	322	0.47	0
655	0.16	322	0.24	0
696	**0.17**	**260**	**0**	**0**
737	0.18	176	0	0
778	0.19	98	0	0
819	0.2	98	0	0.058
860	0.21	98	0	0.167
901	0.22	47	0	0.35

Table 1
Verification results using different error-correcting codes. The Hamming distance (HD) threshold is the ratio of error correction capability to the iris template size (4095 bits). We select error correction capabilities that correspond to desired rounded-off HD threshold values. The key size decreases as the error correction capability increases, and vice versa. We obtain the best performance, shown in bold face, with an error correction capability of 696 bits, corresponding to a HD threshold of 0.17.

Researcher(s)	Biometric Trait Used	Key Size (Bits)	FRR (%)
Monrose et al. (1999)	Keystroke patterns	15	18
Monrose et al. (2001)	Voice	46	17
Goh and Ngo (2003)	Face recognition	80	0.93
Uludag et al. (2005)	Fingerprints	128	21
Hao et al. (2006)	Iris recognition	140	0.47
Santos et al. (2006)	Handwritten signatures	128	57
Yang and Verbauwhede (2007)	Iris recognition	92	0.8
Ziauddin and Dailey (2009)	**Iris recognition**	**260**	**0**

Table 2
Comparison of proposed scheme with past work. The proposed system allows the longest key so far, with perfect verification accuracy on a publicly available dataset.

security against brute force attacks, as we will explain in Section 6. Table 2 compares our system to existing key management schemes.

In Table 2, there are three systems that use iris recognition for key generation. All three systems are based on the theory of the fuzzy commitment scheme and use error correcting codes for error removal. Hao et al. use concatenated codes, Yang and Verbauwhede use a two-block BCH code, and we use a single-block BCH code. Due to the similarity of these systems, we find it appropriate to compare our system with the other two in more detail. One major advantage of our scheme over the other two schemes is generation of much larger keys. Further advantages are listed below.

5.1 Comparison with Hao et al.

We use a single-block BCH code in contrast to the concatenated error-correction coding scheme used by Hao et al. Using a single block code is important because it gives us error correction that is uniform over the codeword and fine-grained control of the tradeoff between error correcting capability and key size. In the concatenated coding scheme of Hao et al., there are two coding layers, where the inner coding layer is dependent on the outer layer and vice versa. It is therefore more difficult to tune their scheme, and the method's actual error correction capability depends on the type and location of the errors.

To verify this claim, we implemented and conducted an experiment using Hao et al.'s concatenated coding scheme with the Bath dataset. Hao et al. state that their system is able to correct errors between enrollment- and verification-time iris templates in up to 27% of the bits. We performed a total of 475 intra-class comparisons using raw templates of size 2048 bits, without our masking or other error reduction techniques. Among these comparisons, the true positive comparison with the largest bit error rate had an error of 25.39%, while the false negative with the smallest bit error rate had an error of 19.87%. Our implementation was able to correct errors in only 5 out of 48 comparisons in which the error rate was between 25% and 27%, and in some cases, even though the error rate was far below 27%, the concatenated code was unable to correct the errors.

This experiment shows that although the Hao et al. scheme works very well for their experimental setup, the concatenated error-correcting code is not necessarily suitable for the error distributions in all datasets, at least with our implementation of the scheme. We thus argue that a single block code that provides precise control on the amount of random error correction capability is more generally applicable.

5.2 Comparison with Yang and Verbauwhede

Besides the larger key size highlighted in Table 2, the main differences between the proposed scheme and that of Yang and Verbauwhede are listed below.

(1) To reduce the intra-class bit error rate, Yang and Verbauwhede use reliable bit selection. We, in addition to reliable bit selection, use reliable region selection, Gaussian blur, and one-sided masking techniques. To gauge the relative merits of these approaches, we performed a small experiment, using 10 images for each of 5 randomly selected subjects in the Bath dataset and comparing the bit error rates output by both schemes. With no special technique used for error reduction, the mean bit error rate was 17.6%. This decreased to 9.6% with reliable bit selection and 3.6% with all of the proposed enhancements. This demonstrates that the improvement achieved by our method is substantial.

(2) Yang and Verbauwhede use a two-block BCH code whereas we use a single-block BCH code. The motivation for the two-block code is apparently efficiency and a larger code rate (k/n) but has the drawback that it provides a non-uniform error correction. Our design gives us flexibility to select code parameters to achieve a desired error threshold without making any assumption about equal error distributions in the two halves of the template.

(3) Most importantly, Yang and Verbauwhede do not provide any security analysis of their system. As the generated keys are intended for use in cryptographic applications including encryption, message authentication, and identity authentication schemes, key secrecy must be protected. We find two vulnerabilities in the Yang and Verbauwhede system that could potentially be exploited by an adversary.
   (a) As we will show in Section 6, an expert attacker, having obtained a user's smart card, could potentially extract the user's key in 2^{11} attempts using a brute force attack. The same attack against our scheme would require 2^{90} attempts. To translate this into time, consider an attacker able to generate and test one key per second. The attack would take 34 minutes (2^{11} seconds) for Yang and Verbauwhede's system or $3.9 * 10^7$ trillion years (2^{90} seconds) for our system.
   (b) Yang and Verbauwhede use 1096 iris bits for their system. To equate the number of iris bits with the codeword size, they append 950 zero bits to their iris template. We understand that, for datasets having large bit error rates, the error correction bounds on the binary block codes dictate the use of smaller template size as compared to the codeword size. But the zero-padding means that if an attacker gains access to the smart card, he or she can examine the recovery information (which is simply a codeword XORed with the iris template) to determine 46% of the codeword bits. This partial information about

the codeword could potentially be used by an expert attacker to retrieve the information word. To give a more secure solution to this problem, in Section 7, we present a modification of the proposed scheme for iris data containing large bit error rates.

6 Security Analysis

Here we perform an informal security analysis to establish upper bounds on the security of the proposed system.

One straightforward brute force attack against our system would be to systematically search the 260-bit key space for the correct key. This step would depend on the application; for example, if the application incorporated an encryption scheme using the information word as a key, the step might amount to encrypting an arbitrary message with the key and submitting the message to a decryption oracle.

We emphasize that as long as the recovery information stored on the smart card and the user's iris code are kept private, the brute force attack or a more sophisticated attack on the cryptosystem itself is the best an attacker can do. Since our keys are up to 260 bits and they are generated by the cryptosystem itself, the security of the cryptosystem will not be affected by our scheme. For example, if our scheme was used with 256-bit AES, we would obtain the security of 256-bit AES.

If, however, an attacker obtains the recovery information we store on the smart card, his or her job could be substantially easier. As the recovery information is the result of XORing a biometric template with a randomly-chosen codeword, an attacker can guess either of the two to extract the other. We consider a powerful attacker who has complete knowledge of the correlations present in iris template space as well as complete knowledge of the BCH codeword space.

The degrees of freedom in iris templates has been estimated to be 249 bits (Daugman, 2003). This means that 249 bits of information is sufficient to reconstruct a valid iris template for one person, and likewise, iris templates are capable of representing the eyes of no more than 2^{249} unique individuals. If an attacker had complete knowledge of the structure of the iris template space, he or she could simply search the 249 bit iris space, each time generating a valid iris template for an individual, using the generated iris template and the recovery information from the smart card to retrieve the corresponding information word, and then checking the correctness of the information word. However, since our code corrects error rates up to 17%, in the worst case, the attacker's task becomes that of finding a 249-bit string within 42 bits of the target bit

string. Following Hao et al.'s analysis based on the sphere-packing bound (also called the Hamming bound), this would require as few as 2^{90} computations. Although 2^{90} is much smaller than 2^{260}, it is far better than any previously reported system — a similar analysis for Hao et al.'s system and Yang and Verbauwhede's system results in attacks requiring only 2^{44} and 2^{11} attempts, respectively.

7 A Modified Scheme for Noisier Iris Data

In this section, we present a slight modification of the proposed scheme useful for overcoming the limitations of error control codes when the iris data is noisier. As previously discussed, in cryptographic key generation systems, it is not feasible to use masking information at verification time. This is a problem, however, because in some cases, iris scans can be severely occluded. For example, the CASIA dataset (CAS, 2004) contains images in which up to half of the iris is not visible due to occlusions caused by eyelids and eyelashes. Obviously, when iris regions that were reliable at enrollment time are compared with corresponding occluded regions at verification time, intra-class Hamming distances will be increased. It is possible to get good equivalent error rates despite large intra-class distances in conventional iris recognition systems, as long as the inter-class distances are even larger. For example, a typical threshold for differentiating intra-class and inter-class comparisons is around 40% for the CASIA dataset (Masek and Kovesi, 2003; Yang and Verbauwhede, 2007). However, it is not possible to work with such high error rates in systems using random block error correcting codes. There is no known random error correcting code, other than trivial ones ($k = 1$ or $k = 2$), that can correct even 25% bit errors in received codewords. To solve this dilemma, the current state of the art is to either to use a maximum distance separable (MDS) burst error control code to increase the error correction capability or to use a larger codeword and append fixed number of zero bits to the iris template to reduce the number of relative errors. Hao et al. use an MDS burst error control code (a Reed-Solomon code) along with the Hadamard code. But unfortunately, as mentioned earlier, their scheme is not universally applicable. Yang and Verbauwhede use the zero-padding approach but unfortunately, as mentioned earlier, this approach has repercussions from a security point of view.

In this section, we briefly present a modification of our scheme along with an experiment using CASIA version 3 dataset. Instead of appending the template with a fixed-length string of zero bits, we generate a unique random string for each user and append it to his or her template. As we will show, this technique gives the same biometric accuracy as zero-padding without exposing any secret data.

The scheme is the same as the one presented in Section 3 with one small change. At enrollment time, the n bit final template W is built by concatenation of x reliable bits selected from the base templates and an $n-x$-bit random string r generated by a pseudorandom number generator. The random string is then stored on the server along with the user id. At verification time, the n-bit final template W' is built by concatenating the x reliable bits selected from the verification time template (using the flag F) and the random string r stored on the server. The rest of the process remains the same.

In order to evaluate the performance of this technique, we conducted an experiment using the CASIA version 3 dataset (Interval). We used all 249 subjects from the dataset and conducted the experiment using all right eye images of those subjects. We used the same size for the base template and flag (9600 bits) and the same (4095, 260, 696) BCH code. We let x be 2048, making the length of r equal to 2047 bits. Our experiments resulted in a FRR of 0.79% and a FAR of 0.002%. The results show that it is practical to use the proposed scheme with noisier iris data and still achieve very good recognition accuracy.

Next we briefly discuss the security of this modified version of the proposed scheme. The user has two major secrets: the cryptographic key and the biometric template. These secrets are not stored anywhere. Instead, we store some side information which is split into two parts such that one part is stored on the smart card and the other on the server. If an attacker steals the smart card, he or she will get some partial information about the user's secrets but, as we have shown in Section 6, this information is insufficient for a practical brute-force attack against the system. On the other hand, if the attacker breaks into the server and gets the user's random string, it does not give him any useful information as the random string is uniform and independent of the key and template. Therefore, the attacker would have to not only break into the server but also steal the smart card in order to obtain useful information about the key or the template (getting both the smart card and the random string would reveal half of the codeword, then the attacker would still have to make some effort to recover the actual key).

Another possibility not requiring additional storage on the server would be to use a password and generate the random string using the password as a seed. However, this would not be user friendly, and it might introduce the possibility of successful dictionary attacks. The approach based on server-side storage of the random string allows processing of noisy iris data without requiring the user to remember any password, and it is also not susceptible to dictionary attacks.

8 Practical Considerations

We implemented the BCH encoder and decoder in Java by rewriting a C implementation by Morelos-Zaragoza (2002). The implementation is quite efficient. On average, running on a Pentium 2.66 GHz processor with 2.5 GB of RAM, the encoder takes 20 ms to encode a 260-bit key while the decoder takes 240 ms to decode a 4095-bit codeword. After image segmentation, the complete enrollment and verification processes take less than one second each. Therefore, the system is practical in terms of time. The Masek Matlab segmentation implementation we used in the experiments described here is quite slow, but it could easily be replaced by a more efficient implementation, e.g., that of Ziauddin and Dailey (2009).

The key is generated using a pseudorandom number generator. For pseudorandom number generation, we use the SecureRandom class of the Java API, which uses SHA-1 as its foundation. Alternatively, we could use the well known Blum Blum Shub PRNG (Blum et al., 1986) or any of the more efficient block cipher-based PRNGs, e.g., Fortuna (Ferguson and Schneier, 2003).

On the issue of space utilization, the system stores 4095 bits of recovery information along with a 9600 bit flag on the smart card. In addition, the smart card stores the hash of the generated key. We can use any cryptographic hash function for this purpose, e.g., SHA-1 or SHA-2 (NIST, 1993). Using SHA-2 with a 256-bit message digest size, the total data storage required on the smart card is 1.7 KB. Many modern smart cards have storage capacities in hundreds of KBs so the storage requirement of the proposed scheme is quite practical.

9 Conclusion

In this paper, we presented a scheme for secret key generation and recovery based on iris verification that, rather than storing keys or iris templates persistently, stores recovery information on a smart card carried by the user. Compared to traditional key management schemes, which are susceptible to dictionary attacks, an attacker attempting to obtain a user's secret key must not only obtain the information encoded on his or her smart card, but must also forge the user's iris template. Properly integrated into a larger cryptosystem, the approach could provide the basis of a practical and convenient means to achieve authenticity and privacy for digital communications.

The system's first main contribution is its generation of 260-bit cryptographic keys which is, by far, the largest among all biometric key generation systems. Second, it is the first biometric key generation system using iris scan which

reports its results on a public iris dataset and achieves recognition accuracy of 100%. Third, we introduce reliable region selection, reliable bit selection and one-sided masking techniques that can significantly decrease the intra-class Hamming distances. Finally, we show that the system is secure if the smart card is compromised and the attacker must still expend a substantial amount of resources to recover the secret key or iris template.

In this paper, we have only considered the fuzzy commitment approach to biometric key management with iris scanning. Another promising direction for future research is the applicability of iris codes in Monrose and colleagues' hardened password scheme (Monrose et al., 1999, 2001).

Acknowledgments

This research was supported by a fellowship from the Higher Education Commission of Pakistan to SZ and grant #MRG4780209 from the Thailand Research Fund to MND. We are grateful to the University of Bath and the Institute of Automation, Chinese Academy of Sciences for allowing us to use their iris image databases. We thank Libor Masek and Peter Kovesi for their excellent open source iris template generation software. We thank Chanathip Namprempre, R.M.A. Premanandana Rajatheva, and Phan Minh Dung for helpful comments on this work. Finally, we thank Robert Morelos-Zaragoza for his excellent implementation of the binary BCH encoder/decoder.

References

Berlekamp, E. R., 1984. Algebraic Coding Theory. Aegean Park Press.

Blahut, R. E., 1983. Theory and Practice of Error Control Codes. Addison-Wesley.

Blum, L., Blum, M., Shub, M., May 1986. A simple unpredictable pseudo-random number generator. SIAM Journal on Computing 15 (2), 364–383.

CAS, 2004. CASIA iris image database. Institute of Automation, Chinese Academy of Sciences, `http://www.sinobiometrics.com`.

Daugman, J., 1985. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. Journal of the Optical Society of America A 2, 1160–1169.

Daugman, J., Feb. 2003. The importance of being random: statistical principles of iris recognition. Pattern Recognition 36 (2), 279–291.

Ferguson, N., Schneier, B., 2003. Practical Cryptography. Wiley, pub-WILEY:adr.

Field, D. J., 1987. Relations between the statistics of natural images and the

response profiles of cortical cells. Journal of the Optical Society of America A 4 (12), 2379–2394.

Goh, A., Ngo, D., 2003. Computation of cryptographic keys from face biometrics. In: Communications and Multimedia Security. pp. 1–13.

Hao, F., Anderson, R., Daugman, J., 2006. Combining crypto with biometrics effectively. IEEE Transactions on Computers 55 (9), 1081–1088.

Juels, A., Sudan, M., 2002. A fuzzy vault scheme. In: Proceedings of IEEE International Symposium on Information Theory. p. 408.

Juels, A., Wattenberg, M., 1999. A fuzzy commitment scheme. In: ACM CCS 99: 6th Conference on Computer and Communications Security. pp. 28–36.

Lin, S., Costello, D. J., 1983. Error Control Coding: Fundamentals and Applications. Prentice-Hall.

Masek, L., 2003. Recognition of human iris patterns for biometric identification. Tech. rep., The School of Computer Science and Software Engineering, The University of Western Australia, `http://www.csse.uwa.edu.au/~pk/studentprojects/libor/index.html`.

Masek, L., Kovesi, P., 2003. MATLAB Source Code for a Biometric Identification System Based on Iris Patterns. The School of Computer Science and Software Engineering, The University of Western Australia, `http://www.csse.uwa.edu.au/~pk/studentprojects/libor/sourcecode.html`.

Monrose, F., Reiter, M. K., Li, Q., Wetzel, S., 2001. Cryptographic key generation from voice. In: SP '01: Proceedings of the 2001 IEEE Symposium on Security and Privacy. IEEE Computer Society, p. 202.

Monrose, F., Reiter, M. K., Wetzel, S., 1999. Password hardening based on keystroke dynamics. In: ACM CCS 99: 6th Conference on Computer and Communications Security. pp. 73–82.

Morelos-Zaragoza, R., 2002. The Art of Error Correcting Coding. John Wiley and Sons, Inc.

NIST, 1993. Secure hash standard. National Institute of Standards and Technology (NIST) FIPS PUB 180, U.S. Department of Commerce, `http://security.isu.edu/pdf/fips180.pdf`.

NIST, 2001. Advanced encryption standard. National Institute of Standards and Technology (NIST) FIPS PUB 197, U.S. Department of Commerce, `http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf`.

Santos, F., Aguilarand, F., Garcia, O., 2006. Cryptographic key generation using handwritten signature. In: Biometric Technology for Human Identification III. SPIE, pp. 62020N.1–62020N.7.

Tomko, G., Soutar, C., Schmidt, G., 1997. Biometric controlled key generation. United States Patent 5680460.

Uludag, U., Pankanti, S., Jain, A. K., 2005. Fuzzy vault for fingerprints. In: AVBPA. pp. 310–319.

University of Bath, 2004. University of Bath iris image database. `http://www.bath.ac.uk/elec-eng/research/sipg/irisweb/`.

Yang, S., Verbauwhede, I., 2007. Secure iris verification. In: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP. pp. II–

133–II–136.

Ziauddin, S., Dailey, M. N., 2009. A robust hybrid iris localization technique. In: 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON 2009).