# Enhancing security and privacy in biometrics-based authentication systems

by N. K. Ratha
J. H. Connell
R. M. Bolle

***Because biometrics-based authentication offers several advantages over other authentication methods, there has been a significant surge in the use of biometrics for user authentication in recent years. It is important that such biometrics-based authentication systems be designed to withstand attacks when employed in security-critical applications, especially in unattended remote applications such as e-commerce. In this paper we outline the inherent strengths of biometrics-based authentication, identify the weak links in systems employing biometrics-based authentication, and present new solutions for eliminating some of these weak links. Although, for illustration purposes, fingerprint authentication is used throughout, our analysis extends to other biometrics-based methods.***

Reliable user authentication is becoming an increasingly important task in the Web-enabled world. The consequences of an insecure authentication system in a corporate or enterprise environment can be catastrophic, and may include loss of confidential information, denial of service, and compromised data integrity. The value of reliable user authentication is not limited to just computer or network access. Many other applications in everyday life also require user authentication, such as banking, e-commerce, and physical access control to computer resources, and could benefit from enhanced security.

The prevailing techniques of user authentication, which involve the use of either passwords and user IDs (identifiers), or identification cards and PINs (personal identification numbers), suffer from several limitations. Passwords and PINs can be illicitly acquired by direct covert observation. Once an intruder acquires the user ID and the password, the intruder has total access to the user's resources. In addition, there is no way to positively link the usage of the system or service to the actual user, that is, there is no protection against repudiation by the user ID owner. For example, when a user ID and password is shared with a colleague there is no way for the system to know who the actual user is. A similar situation arises when a transaction involving a credit card number is conducted on the Web. Even though the data are sent over the Web using secure encryption methods, current systems are not capable of assuring that the transaction was initiated by the rightful owner of the credit card. In the modern distributed systems environment, the traditional authentication policy based on a simple combination of user ID and password has become inadequate.

Fortunately, automated biometrics in general, and fingerprint technology in particular, can provide a much more accurate and reliable user authentication method. Biometrics is a rapidly advancing field that is concerned with identifying a person based on his or her physiological or behavioral characteristics. Examples of automated biometrics include fingerprint, face, iris, and speech recognition. User authentication methods can be broadly classified into three categories[1] as shown in Table 1. Because a biometric

0018-8670/01/$5.00 © 2001 IBM

Table 1 Existing user authentication techniques

| Method | Examples | Properties |
|---|---|---|
| What you know | User ID<br>Password<br>PIN | Shared<br>Many passwords easy to guess<br>Forgotten |
| What you have | Cards<br>Badges<br>Keys | Shared<br>Can be duplicated<br>Lost or stolen |
| What you know and what you have | ATM card + PIN | Shared<br>PIN a weak link<br>(Writing the PIN on the card) |
| Something unique about the user | Fingerprint<br>Face<br>Iris<br>Voice print | Not possible to share<br>Repudiation unlikely<br>Forging difficult<br>Cannot be lost or stolen |

property is an intrinsic property of an individual, it is difficult to surreptitiously duplicate and nearly impossible to share. Additionally, a biometric property of an individual can be lost only in case of serious accident.

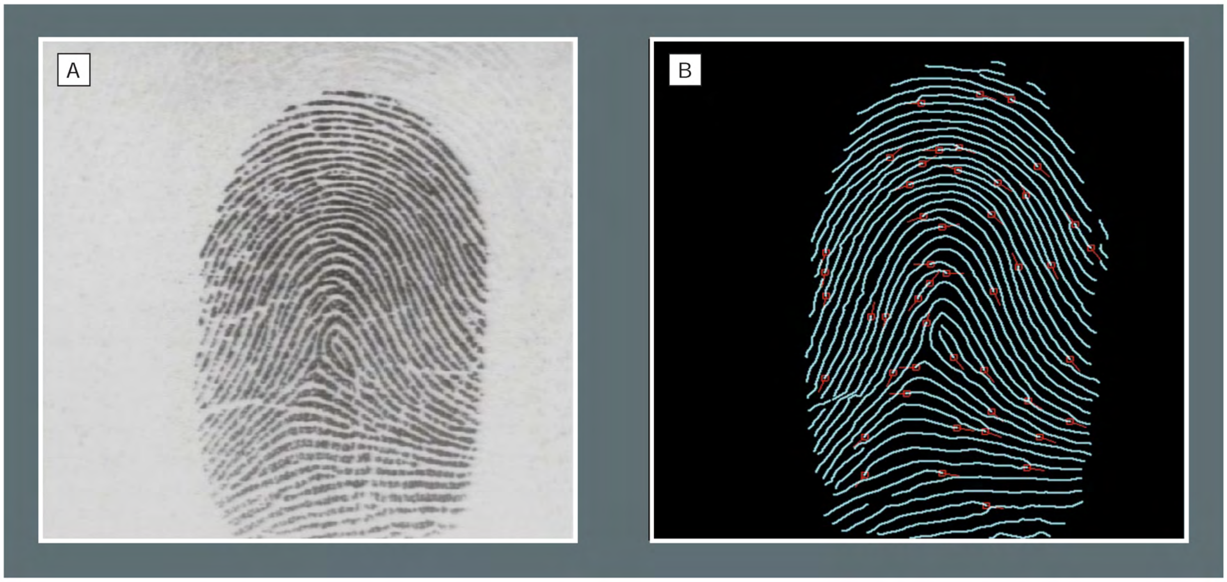
Biometric readings, which range from several hundred bytes to over a megabyte, have the advantage that their information content is usually higher than that of a password or a pass phrase. Simply extending the length of passwords to get equivalent bit strength presents significant usability problems. It is nearly impossible to remember a 2K phrase, and it would take an annoyingly long time to type such a phrase (especially without errors). Fortunately, automated biometrics can provide the security advantages of long passwords while retaining the speed and characteristic simplicity of short passwords.

Even though automated biometrics can help alleviate the problems associated with the existing methods of user authentication, hackers will still find there are weak points in the system, vulnerable to attack. Password systems are prone to brute force dictionary attacks. Biometric systems, on the other hand, require substantially more effort for mounting such an attack. Yet there are several new types of attacks possible in the biometrics domain. This may not apply if biometrics is used as a supervised authentication tool. But in remote, unattended applications, such as Web-based e-commerce applications, hackers may have the opportunity and enough time to make several attempts, or even physically violate the integrity of a remote client, before detection.

A problem with biometric authentication systems arises when the data associated with a biometric feature has been compromised. For authentication systems based on physical tokens such as keys and badges, a compromised token can be easily canceled and the user can be assigned a new token. Similarly, user IDs and passwords can be changed as often as required. Yet, the user only has a limited number of biometric features (one face, ten fingers, two eyes). If the biometric data are compromised, the user may quickly run out of biometric features to be used for authentication.

In this paper, we discuss in more detail the problems unique to biometric authentication systems and propose solutions to several of these problems. Although we focus on fingerprint recognition throughout this paper, our analysis can be extended to other biometric authentication methods. In the next section, "Fingerprint authentication," we detail the stages of the fingerprint authentication process. In the following section, "Vulnerable points of a biometric system," we use a pattern recognition framework for a generic biometric system to help identify the possible attack points. The section "Brute force attack directed at matching fingerprint minutiae" analyzes the resilience of a minutiae-based fingerprint authentication system in terms of the probability of a successful brute force attack. The next two sections, "WSQ-based data hiding" and "Image-based challenge/response method," propose two methods that address some of the vulnerable points of a biometric system. The section "Cancelable biometrics" introduces the concept of "cancelable biometrics"

Figure 1 Fingerprint recognition; (A) input image, (B) features



and discusses its application to authentication. Finally, the section "Conclusions" recapitulates the issues discussed and summarizes the proposed new approaches.

## Fingerprint authentication

We present here a brief introduction to fingerprint authentication. Readers familiar with fingerprint authentication may skip to the next section.

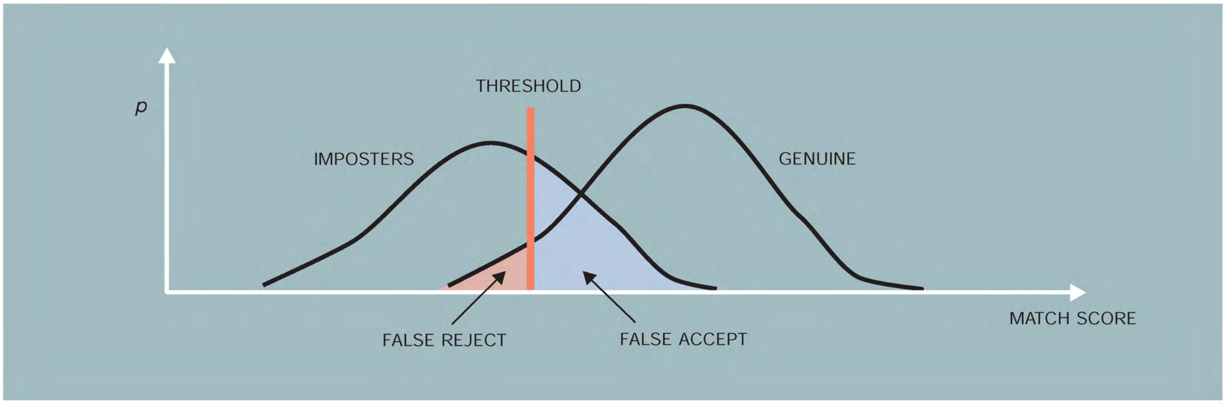
Fingerprints are a distinctive feature and remain invariant over the lifetime of a subject, except for cuts and bruises. As the first step in the authentication process, a fingerprint impression is acquired, typically using an inkless scanner. Several such scanning technologies are available.[2] Figure 1A shows a fingerprint obtained with a scanner using an optical sensor. A typical scanner digitizes the fingerprint impression at 500 dots per inch (dpi) with 256 gray levels per pixel. The digital image of the fingerprint includes several unique features in terms of ridge bifurcations and ridge endings, collectively referred to as *minutiae*.

The next step is to locate these features in the fingerprint image, as shown in Figure 1B, using an automatic feature extraction algorithm. Each feature is commonly represented by its location $(x, y)$ and the ridge direction at that location $(\theta)$. However, due to sensor noise and other variability in the imaging process, the feature extraction stage may miss some minutiae and may generate spurious minutiae. Further, due to the elasticity of the human skin, the relationship between minutiae may be randomly distorted from one impression to the next.

In the final stage, the matcher subsystem attempts to arrive at a degree of similarity between the two sets of features after compensating for the rotation, translation, and scale. This similarity is often expressed as a score. Based on this score, a final decision of match or no-match is made. A decision threshold is first selected. If the score is below the threshold, the fingerprints are determined not to match; if the score is above the threshold, a correct match is declared. Often the score is simply a count of the number of the minutiae that are in correspondence. In a number of countries, 12 to 16 correspondences (performed by a human expert) are considered legally binding evidence of identity.

The operational issues in an automated fingerprint identification system (AFIS) are somewhat different from those in a more traditional password-based system. First, there is a system performance issue known as the "fail to enroll" rate to be considered. Some people have very faint fingerprints, or no fingers at

Figure 2 Error trade-off in a biometric system



all, which makes the system unusable for them. A related issue is a "Reject" option in the system based on input image quality. A poor quality input is not accepted by the system during enrollment and authentication. Note that poor quality inputs can be caused by noncooperative users, improper usage, dirt on the finger, or bad input scanners. This has no analog in a password system. Then there is the fact that in a biometric system the matching decision is not clear-cut. A password system always provides a correct response—if the passwords match, it grants access but otherwise refuses access. However, in a biometric system, the overall accuracy depends on the quality of input and enrollment data along with the basic characteristics of the underlying feature extraction and matching algorithm.

For fingerprints, and biometrics in general, there are two basic types of recognition errors, namely the false accept rate (FAR) and the false reject rate (FRR). If a nonmatching pair of fingerprints is accepted as a match, it is called a false accept. On the other hand, if a matching pair of fingerprints is rejected by the system, it is called a false reject. The error rates are a function of the threshold as shown in Figure 2. Often the interplay between the two errors is presented by plotting FAR against FRR with the decision threshold as the free variable. This plot is called the ROC (Receiver Operating Characteristic) curve. The two errors are complementary in the sense that if one makes an effort to lower one of the errors by varying the threshold, the other error rate automatically increases.

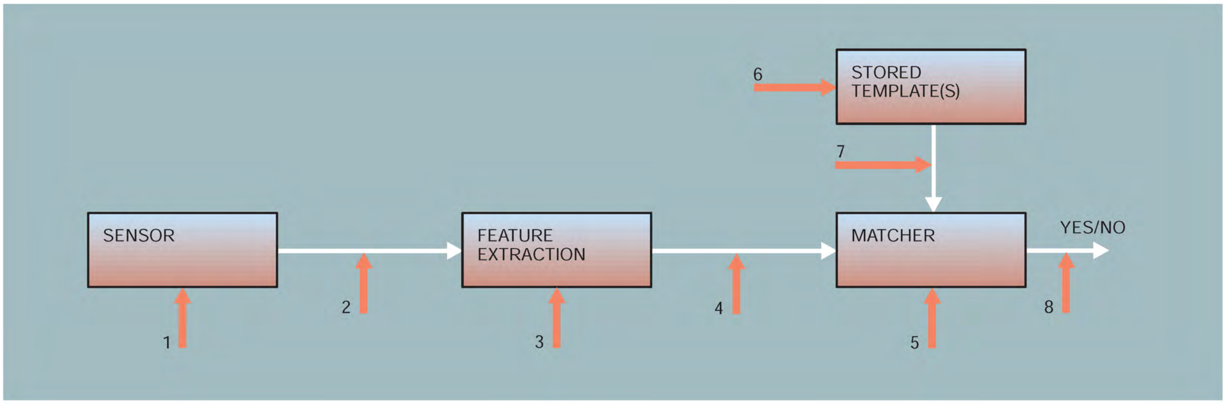
In a biometric authentication system, the relative false accept and false reject rates can be set by choosing a particular operating point (i.e., a detection threshold). Very low (close to zero) error rates for both errors (FAR and FRR) at the same time are not possible. By setting a high threshold, the FAR error can be close to zero, and similarly by setting a significantly low threshold, the FRR rate can be close to zero. A meaningful operating point for the threshold is decided based on the application requirements, and the FAR versus FRR error rates at that operating point may be quite different. To provide high security, biometric systems operate at a low FAR instead of the commonly recommended equal error rate (EER) operating point where FAR = FRR. High-performance fingerprint recognition systems can support error rates in the range of $10^{-6}$ for false accept and $10^{-4}$ for false reject.[3] The performance numbers reported by vendors are based on test results using private databases and, in general, tend to be much better than what can be achieved in practice. Nevertheless, the probability that the fingerprint signal is supplied by the right person, given a good matching score, is quite high. This confidence level generally provides better nonrepudiation support than passwords.

## Vulnerable points of a biometric system

A generic biometric system can be cast in the framework of a pattern recognition system. The stages of such a generic system are shown in Figure 3. Excellent introductions to automated biometric systems can be found in References 1 and 4.

The first stage involves biometric signal acquisition from the user (e.g., the inkless fingerprint scan). The acquired signal typically varies significantly from pre-

Figure 3 Possible attack points in a generic biometrics-based system



sentation to presentation; hence, pure pixel-based matching techniques do not work reliably. For this reason, the second signal processing stage attempts to construct a more invariant representation of this basic input signal (e.g., in terms of fingerprint minutiae). The invariant representation is often a spatial domain characteristic or a transform (frequency) domain characteristic, depending on the particular biometric.

During enrollment of a subject in a biometric authentication system, an invariant template is stored in a database that represents the particular individual. To authenticate the user against a given ID, the corresponding template is retrieved from the database and matched against the template derived from a newly acquired input signal. The matcher arrives at a decision based on the closeness of these two templates while taking into account geometry, lighting, and other signal acquisition variables.

Note that password-based authentication systems can also be set in this framework. The keyboard becomes the input device. The password encryptor can be viewed as the feature extractor and the comparator as the matcher. The template database is equivalent to the encrypted password database.

We identified eight places in the generic biometric system of Figure 3 where attacks may occur. In addition, Schneier[5] describes several types of abuses of biometrics. The numbers in Figure 3 correspond to the items in the following list.

1. Presenting fake biometrics at the sensor: In this mode of attack, a possible reproduction of the biometric feature is presented as input to the system. Examples include a fake finger, a copy of a signature, or a face mask.
2. Resubmitting previously stored digitized biometrics signals: In this mode of attack, a recorded signal is replayed to the system, bypassing the sensor. Examples include the presentation of an old copy of a fingerprint image or the presentation of a previously recorded audio signal.
3. Overriding the feature extraction process: The feature extractor is attacked using a Trojan horse, so that it produces feature sets preselected by the intruder.
4. Tampering with the biometric feature representation: The features extracted from the input signal are replaced with a different, fraudulent feature set (assuming the representation method is known). Often the two stages of feature extraction and matcher are inseparable and this mode of attack is extremely difficult. However, if minutiae are transmitted to a remote matcher (say, over the Internet) this threat is very real. One could "snoop" on the TCP/IP (Transmission Control Protocol/Internet Protocol) stack and alter certain packets.
5. Corrupting the matcher: The matcher is attacked and corrupted so that it produces preselected match scores.
6. Tampering with stored templates: The database of stored templates could be either local or remote. The data might be distributed over several servers. Here the attacker could try to modify one or more templates in the database, which could result either in authorizing a fraudulent individual or denying service to the persons associated with the corrupted template. A smartcard-based

authentication system,[6] where the template is stored in the smartcard and presented to the authentication system, is particularly vulnerable to this type of attack.

7. Attacking the channel between the stored templates and the matcher: The stored templates are sent to the matcher through a communication channel. The data traveling through this channel could be intercepted and modified.
8. Overriding the final decision: If the final match decision can be overridden by the hacker, then the authentication system has been disabled. Even if the actual pattern recognition framework has excellent performance characteristics, it has been rendered useless by the simple exercise of overriding the match result.

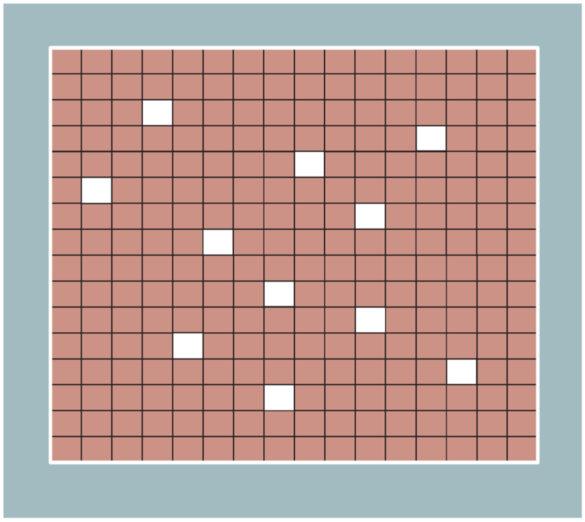
There exist several security techniques to thwart attacks at these various points. For instance, finger conductivity or fingerprint pulse at the sensor can stop simple attacks at point 1. Encrypted communication channels[7] can eliminate at least remote attacks at point 4. However, even if the hacker cannot penetrate the feature extraction module, the system is still vulnerable. The simplest way to stop attacks at points 5, 6, and 7 is to have the matcher and the database reside at a secure location. Of course, even this cannot prevent attacks in which there is collusion. Use of cryptography[8] prevents attacks at point 8.

We observe that the threats outlined in Figure 3 are quite similar to the threats to password-based authentication systems. For instance, all the channel attacks are similar. One difference is that there is no "fake password" equivalent to the fake biometric attack at point 1 (although, perhaps if the password was in some standard dictionary it could be deemed "fake"). Furthermore, in a password- or token-based authentication system, no attempt is made to thwart replay attacks (since there is no expected variation of the "signal" from one presentation to another). However, in an automated biometric-based authentication system, one can check the liveness of the entity originating the input signal.

## Brute force attack directed at matching fingerprint minutiae

In this section we attempt to analyze the probability that a brute force attack at point 4 of Figure 3, involving a set of fraudulent fingerprint minutiae, will succeed in matching a given stored template. Figure 4 shows one such randomly generated minutiae set. In a smart card system where the biometrics template is stored in the card and presented to the authentication system, a hacker could present these random sets to the authentication system assuming that the hacker has no information about the stored templates. Note that an attack at point 2 of Figure 3, which involves generating all possible fingerprint images in order to match a valid fingerprint image, would have an even larger search space and consequently would be much more difficult.

Figure 4 Example of a randomly generated minutiae set

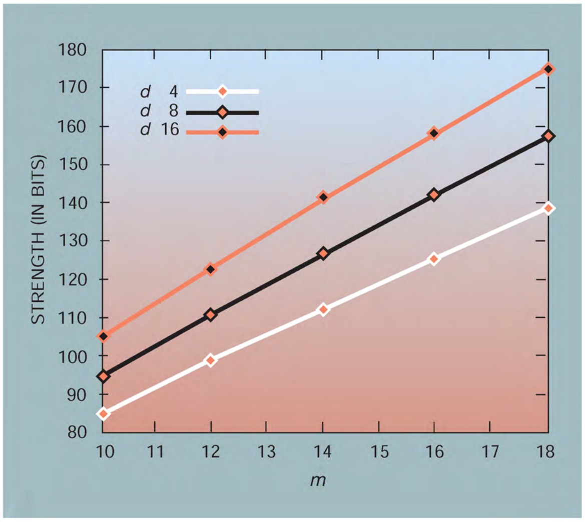


**A naive model.** For the purpose of analyzing the "naive" matching minutiae attack, we assume the following.

- The system uses a minutia-based matching method and the number of paired minutiae reflects the degree of match.
- The image size $S = 300$ pixels $\times$ 300 pixels.
- A ridge plus valley spread $T = 15$ pixels.
- The total number of possible minutiae sites ($K = S/(T^2)) = 20 \times 20 = 400$.
- The number of orientations allowed for the ridge angle at a minutia point $d = 4, 8, 16$.
- The minimum number of corresponding minutiae in query and reference template $m = 10, 12, 14, 16, 18$.

These values are based on a standard fingerprint scanner with 500 dpi scanning resolution covering an area 0.6 × 0.6 inches. A ridge and valley can span

Figure 5 Bit strength in the naive model



about 15 pixels on average at this scanning resolution. The other two variables *d* and *m* are being used as parameters to study the brute force attack. We start with 10 matching minutiae since often a threshold of 12 minutiae is used in matching fingerprints in manual systems. Ridge angles in an automated system can be quantized depending on the tolerance supported in the matcher. A minimum of four quantization levels provides a 45 degree tolerance, while 16 levels provides roughly an 11 degree tolerance.

Then, the number of possible ways to place *m* minutiae in *K* possible locations is

$$\binom{K}{m} \tag{1}$$

and, the number of possible ways to assign directions to the minutiae is $d^m$.

Hence, the total number of possible minutiae combinations equals

$$\binom{K}{m} \times (d^m) \tag{2}$$

Note that it is assumed that the matcher will tolerate shifts between query and reference minutiae of at most a ridge and valley pixel width, and an angular difference of up to half a quantization bin (±45 degrees for $d = 4$).

Plugging these values into Equation 2, for $d = 4$ and $m = 10$, the probability of randomly guessing the exact feature set is $3.6 \times 10^{-26} = 2^{-84.5}$. The $\log_2$ of the probability of randomly guessing a correct feature set *through a brute force attack* for different values of *d* and *m* is plotted in Figure 5. We refer to this measure (in bits) as "strength," and it represents the equivalent number of bits in a password authentication system. This should convince the reader that a brute force attack in the form of a random image or a random template attempting to impersonate an authorized individual will, on average, require a very large number of attempts before succeeding.

The foregoing analysis assumes that each fingerprint has exactly *m* minutiae, that only *m* minutiae are generated, and that all of these minutiae have to match. A realistic strength is much lower because one can generate more than *m* query minutiae, say $N_{total}$, and only some fraction of these must match *m* minutiae of the reference fingerprint. This leads to a factor of about $\binom{N_{total}}{m}^2$ or a loss of nearly 64 bits in strength for $m = 10$ with $N_{total} = 50$. The equivalent strength thus is closer to 20 bits for this parameter set. A more realistic model, which carefully incorporates this effect, is described below.

**A more realistic model.** In the naive approach, we made several simplifying assumptions. In this more realistic model, we will make assumptions that are more realistic and will analyze the brute force attack model in more detail.

Let the reference print have $N_r$ minutiae, and let each feature of the minutiae include a ridge direction which takes *d* possible values, and a location which takes *K* possible values. Then the probability that a randomly generated minutia will match one of the minutiae in the reference print in both location and direction can be approximated as:

$$p_{est} = \frac{N_r}{K \times d} \tag{3}$$

A more accurate model would require that we consider the probability of a minutiae site being populated as a function of the distance to the center of the print (they are more likely in the middle). In addition, such a model would require that the directional proclivities depend on location (they tend to swirl around the core). In this model, however, we ignore such dependencies and use the simpler formulation.

While the expression above is valid for the first generated minutia, when creating the full synthetic set it is undesirable to generate two minutiae with the same location. So after $j - 1$ minutiae have been generated, the probability that the $j$th minutia will match (assuming the previous $j - 1$ minutiae all fail to match) is bounded from above by:

$$\frac{N_r}{(K - j + 1)d} \tag{4}$$

Thus, while generating $N_q$ random minutiae we can conservatively assume each minutia has matching probability:

$$p = p_{hi} = \frac{N_r}{(K - N_q + 1)d} \tag{5}$$

Typical parameter values are $K = 400$, $N_q = N_r = 50$ and $d = 4$. Note that brute force attacks with $N_q$ excessively large (close to the value $K$) would be easy to detect and reject out of hand. For this reason there is an upper bound on $N_q$ that still enables an attacker to generate the facsimile of a real finger. Using the values above we find $p_{est} = 0.03125$ while $p_{hi} = 0.03561$ (14 percent higher). This is a relatively small effect in itself, but important in the overall calculation.

Therefore, the probability of getting *exactly t* of $N_q$ generated minutiae to match is about:

$$P_{thresh} = p^t(1 - p)^{N_q - t} \tag{6}$$

This derivation breaks down for small $K$ because the minutiae matching probability changes depending on how many other minutiae have already been generated as well as on how many of those minutiae have matched. However, for the large values of $K$ typically encountered (e.g., 400) it is reasonably close.

Now there are a number of ways of selecting which $t$ out of the $N_r$ minutiae in the reference print are the ones that match. Thus, the total match probability becomes:

$$P_{exact} = \binom{N_r}{t} p^t(1 - p)^{N_q - t} \tag{7}$$

But matches of $m$ or *more* minutiae typically count as a verification, so we get:

$$P_{ver} = \sum_{t=m}^{N_q} \binom{N_r}{t} p^t(1 - p)^{N_q - t} \tag{8}$$

For convenience, let us assume that $N_q = N_r = N$, so the above equation can be rewritten as:

$$P_{ver} = \sum_{t=m}^{N} \binom{N}{t} p^t(1 - p)^{N - t} \tag{9}$$

Since $p$ is fairly small in our case, we can use the Poisson approximation to the above binomial probability density function:

$$P_{ver} = \sum_{t=m}^{N} \frac{(Np)^t e^{-Np}}{t!} \tag{10}$$

This summation is usually dominated by its first term (where $t = m$). For typical parameter values the second term is 10 to 20 times smaller than the first. Neglecting all but the first term may make the overall estimate approximately 20 percent lower, but for order-of-magnitude calculations this is fine. Thus, we rewrite the expression as simply:

$$P_{ver} = \frac{(Np)^m e^{-Np}}{m!} \tag{11}$$

Because $m$ is moderately large, we can use Stirling's approximation for the factorial and further rewrite the equation as:

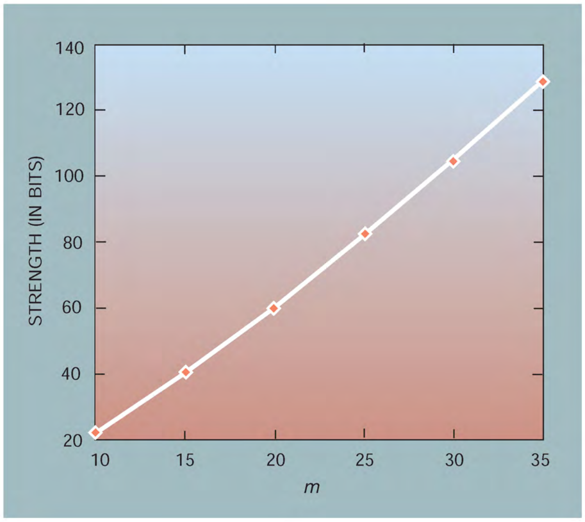
$$P_{ver} = \frac{(Np)^m e^{-Np}}{\sqrt{(2\pi m)}\, e^{-m} m^m} \tag{12}$$

and regrouping to emphasize the exponential dependency:

$$P_{ver} = \frac{e^{-Np}}{\sqrt{2\pi m}} \left(\frac{eNp}{m}\right)^m \tag{13}$$

The $\log_2$ of $P_{ver}$ (bit strength) is plotted in Figure 6 for $N = 40$, $d = 4$, $K = 400$ with $m$ (the number of minutiae required to match) between 10 and 35. For a value of $m = 10$, we have about 22 bits of information (close to the prediction of the revised

Figure 6 Bit strength in the more realistic model



naive model). For the legal threshold of $m = 15$, we have around 40 bits of information (representing a number of distinct binary values equal to about 140 times the population of the earth). For a more typical value of $m = 25$, we have roughly 82 bits of information content in this representation. This is equivalent to a 16-character nonsense password (such as "m4yus78xpmks3bc9").

Studies similar to ours have been reported in the literature, and these studies evaluate the individuality of a fingerprint based on the minutiae information.[9,10] These analyses were based on the minutiae frequency data collected and interpreted by a human expert and involving a small set of fingers. Furthermore, these studies used all the ten types of Galton characteristics,[11] whereas our study is based on just one type of feature (with no differentiation between ridge endings and bifurcations). The purpose of these studies was to quantify the information content of a fingerprint (similar to our naive method) rather than set thresholds for matching in the face of brute force attacks.

Examining the final equation (Equation 13), we make two important observations. First, in both the naive and the more realistic model, it can be seen that adding extra feature information at every minutia (e.g., raising $d$) increases significantly the strength of the system. Similarly, if the spatial domain extent is increased or the number of minutiae sites $K$ are increased, the strength also increases. Both these factors directly affect $p$, the single minutia matching probability, which shows up inside the exponential term of $P_{ver}$. Second, there is a strong dependence on $N$, the overall number of minutiae in a fingerprint. For high security, this number needs to be kept as low as possible. This is one reason why the probability of break-ins is much smaller when good quality fingerprint images are enrolled as opposed to using poor quality images with many spurious minutiae (yielding a higher overall $N$). Often practical systems reject a bad quality fingerprint image for this reason instead of taking a hit on the accuracy of the system.

It should be pointed out that the brute force attack break-in probability is not dependent in any way on the FAR. That is, if the FAR is $10^{-6}$, this does not mean that, on average, the system is broken into after 500000 trials. The FAR is estimated using actual human fingers and is typically attributable to errors in feature extraction (extra or missing features) and, to a lesser extent, to changes in geometry such as finger rolling or skin deformations due to twisting. The statistics governing the occurrence of these types of errors are different from those describing a brute force attack.

## WSQ-based data hiding

In both Web-based and other on-line transaction processing systems, it is undesirable to send uncompressed fingerprint images to the server due to bandwidth limitations. A typical fingerprint image is of the order of 512 × 512 pixels with 256 gray levels, resulting in a file size of 256 Kbytes. This would take nearly 40 seconds to transmit at 53 Kbaud. Unfortunately, many standard compression methods, such as JPEG (Joint Photographic Experts Group), have a tendency to distort the high-frequency spatial and structural ridge features of a fingerprint image. This has led to several research proposals regarding domain-specific compression methods. As a result, an open Wavelet Scalar Quantization (WSQ) image compression scheme proposed by the FBI[12] has become the *de facto* standard in the industry, because of its low image distortion even at high-compression ratios (over 10:1).

Typically, the compressed image is transmitted over a standard encrypted channel as a replacement for (or in addition to) the user's PIN. Yet, because of the open compression standard, transmitting a WSQ compressed image over the Internet is not particularly secure. If a compressed fingerprint image bitstream can be freely intercepted (and decrypted), it can be decompressed using readily available software. This potentially allows the signal to be saved and fraudulently reused (attack point 2 in Figure 3).

One way to enhance security is to use data-hiding techniques to embed additional information directly in compressed fingerprint images. For instance, if the embedding algorithm remains unknown, the service provider can look for the appropriate standard watermark to check that a submitted image was indeed generated by a trusted machine (or sensor). Several techniques have been proposed in the literature for hiding digital watermarks in images.[13,14] Bender et al.[15] and Swanson et al.[16] present excellent surveys of data-hiding techniques. Petitcolas et al.[14] provide a nice survey and taxonomy of information-hiding techniques. Hsu and Wu[17] describe a method for hiding watermarks in JPEG compressed images. Most of the research, however, addresses issues involved in resolving piracy or copyright issues, not authentication. An exception is the invisible watermarking technique for fingerprints proposed by Yeung and Pankanti.[18] Their study involves examining the accuracy after an invisible watermark is inserted in the image domain. Our proposed solution is different because, first, it operates directly in the compressed domain and, second, it causes no performance degradation.

The approach is motivated by the desire to create on-line fingerprint authentication systems for commercial transactions that are secure against replay attacks. To achieve this, the service provider issues a different verification string for each transaction. The string is mixed in with the fingerprint image before transmission. When the image is received by the service provider it is decompressed and the image is checked for the presence of the correct one-time verification string. The method we propose here hides such messages with minimal impact on the appearance of the decompressed image. Moreover, the message is not hidden in a fixed location (which would make it more vulnerable to discovery) but is, instead, deposited in different places *based on the structure of the image itself*. Although our approach is presented in the framework of fingerprint image compression, it can be easily extended to other biometrics such as wavelet-based compression of facial images.

Our information hiding scheme works in conjunction with the WSQ fingerprint image encoder and decoder, which are shown in Figures 7A and 7B, respectively. In the first step of the WSQ compression, the input image is decomposed into 64 spatial frequency subbands using perfect reconstruction multirate filter banks[19] based on discrete wavelet transformation filters. The filters are implemented as a pair of separable 1D filters. The two filters specified for encoder 1 of the FBI standard are plotted in Figures 7C and 7D. The subbands are the filter outputs obtained after a desired level of cascading of the filters as described in the standard. For example, subband 25 corresponds to the cascading path of "00, 10, 00, 11" through the filter bank. The first digit in each binary pair represents the row operation index. A zero specifies low pass filtering using h0 on the row (column) while a one specifies high pass filtering using h1 on the row (column). Thus for the 25th subband, the image is first low pass filtered in both row and column; followed by high pass filtering in rows, then low pass filtering in columns; the output of which is then low pass filtered in rows and columns; and ending with high pass filtering in rows and columns. Note that there is appropriate down sampling and the symmetric extension transform is applied at every stage as specified in the standard. The 64 subbands of the gray-scale fingerprint image shown in Figure 8A are shown in Figure 8C.

There are two more stages to WSQ compression. The second stage is a quantization process where the Discrete Wavelet Transform (DWT) coefficients are transformed into integers with a small number of discrete values. This is accomplished by uniform scalar quantization for each subband. There are two characteristics for each band: the zero of the band ($Z_k$) and the width of the bins ($Q_k$). These parameters must be chosen carefully to achieve a good compression ratio without introducing significant information loss that will result in distortions of the images. The $Z_k$ and $Q_k$ for each band are transmitted directly to the decoder. The third and final stage is Huffman coding of the integer indices for the DWT coefficients. For this purpose, the bands are grouped into three blocks. In each block, the integer coefficients are remapped to numbers between 0–255 prescribed by the translation table described in the standard. This translation table encodes run lengths of zeros and large values. Negative coefficients are translated in a similar way by this table.

Our data-hiding algorithm works on the quantized indices before this final translation (i.e., between stages

Figure 7A, B WSQ algorithm; (A) compression, (B) decompression

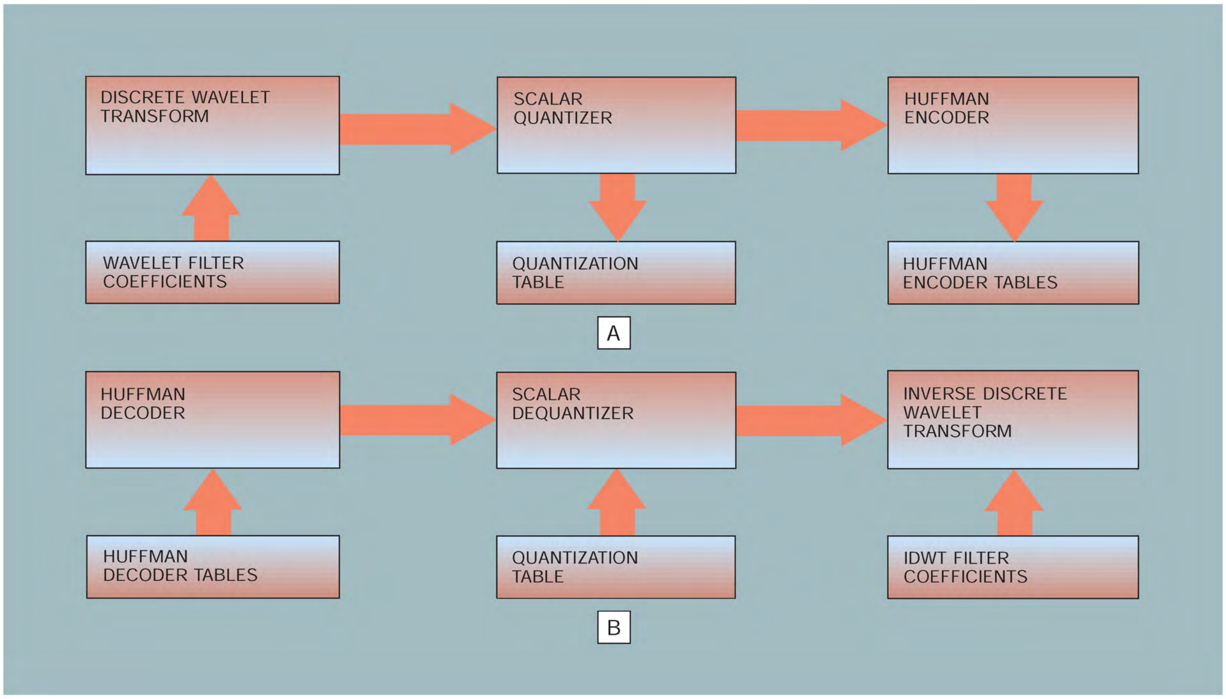


Figure 7C Analysis filter *h*0

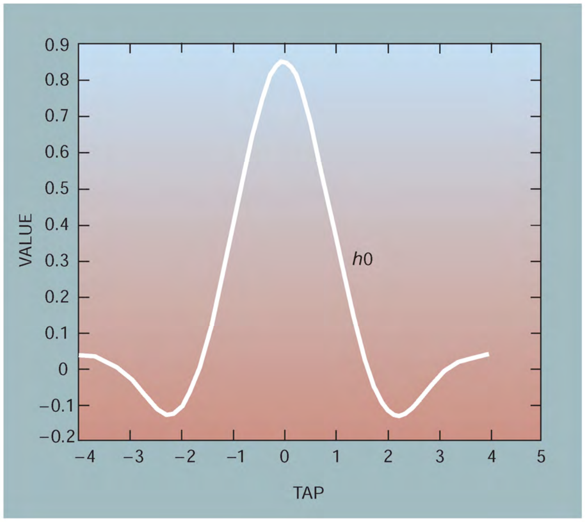
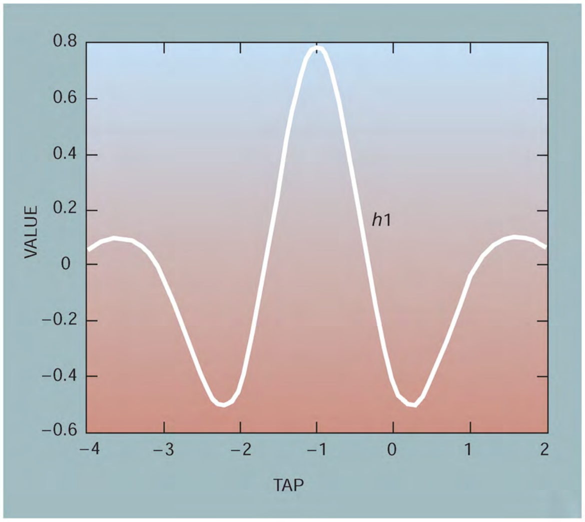


Figure 7D Analysis filter *h*1



2 and 3). We assume the message size is very small compared to the image size (or, equivalently, the number of DWT coefficients). Note, however, that the Huffman coding characteristics and tables are not changed; the tables are computed as for the original coefficients, not after the coefficient altering steps described next.

Figure 8A WSQ data-hiding results; (A) original image

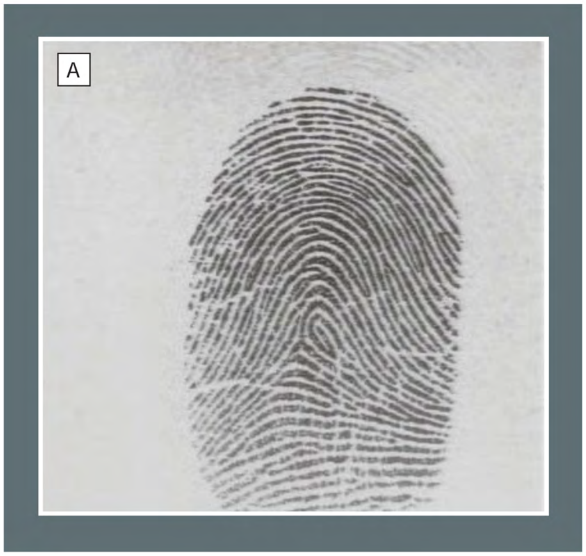
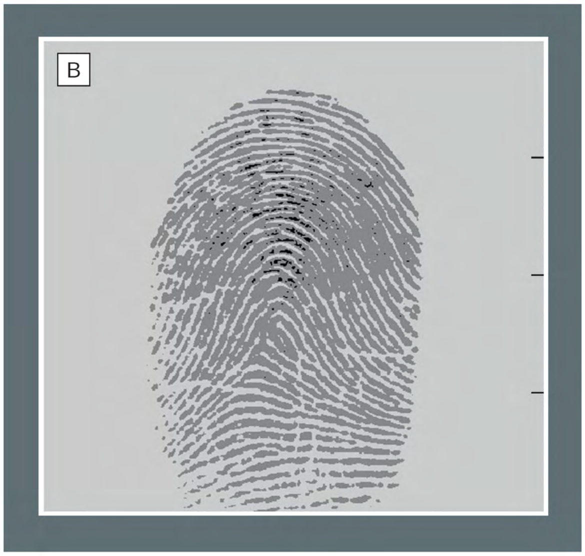


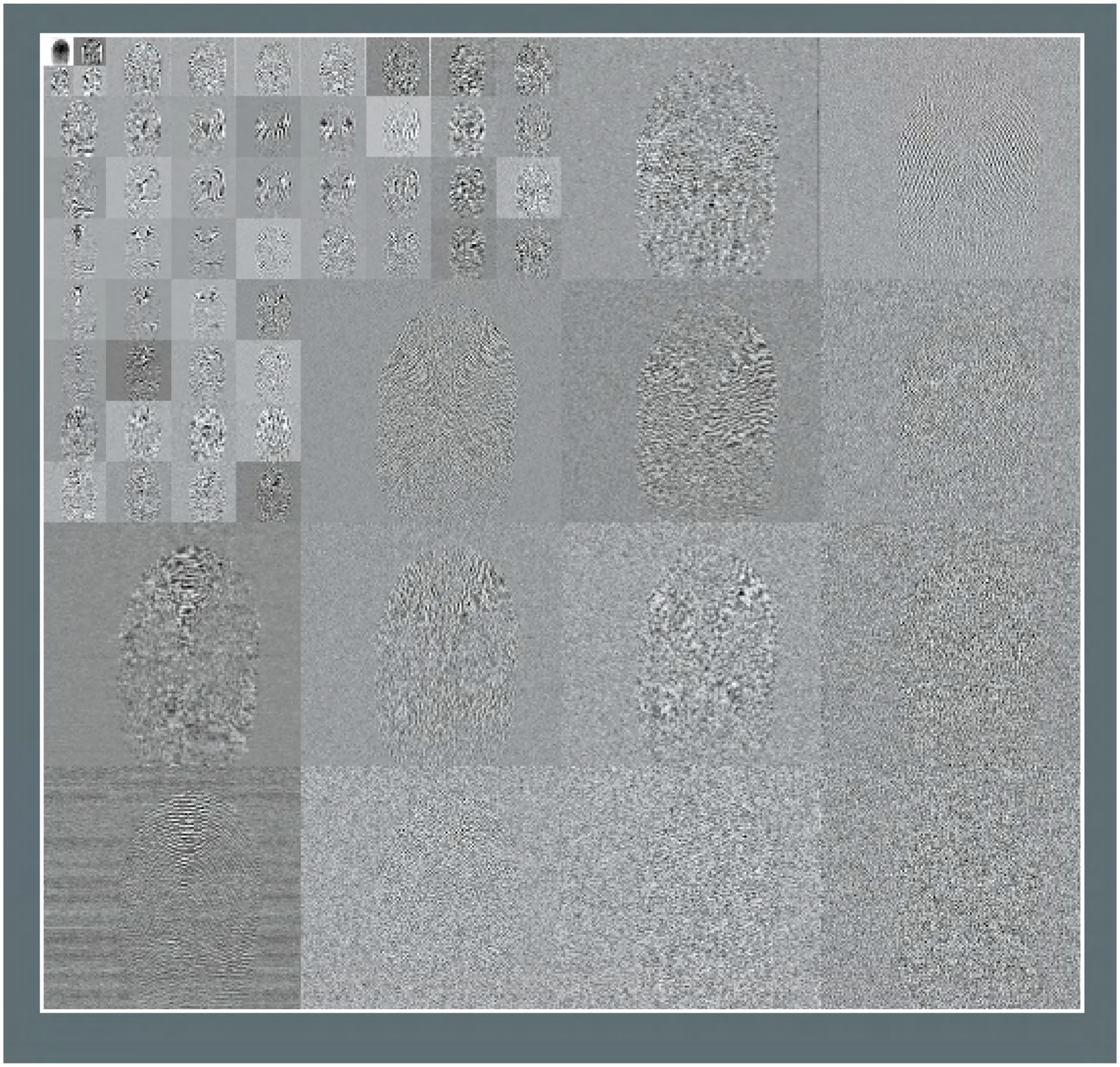
Figure 8B WSQ data-hiding results; (B) reconstructed image



As mentioned, our method is intended for messages which are very small (in terms of bits) compared to the number of pixels in the image. The basic principle is to find and slightly alter certain of the DWT coefficients. However, care must be taken to avoid corrupting the reconstructed image. To hide a message during the image encoding process, we perform three (or, optionally, four) basic steps:

- Selecting a set of sites $S$: Given the partially converted quantized integer indices, this stage collects the indices of all possible coefficient sites where a change in the least significant bit is tolerable. Typically, all sites in the low frequency bands are excluded. Even small changes in these coefficients can affect large regions of the image because of the low frequencies. For the higher frequencies, candidate sites are selected if they have coefficients of large magnitude. Making small changes to the larger coefficients leads to relatively small percentage changes in the values and hence minimal degradation of the image. Note that among the quantizer indices there are special codes to represent run lengths of zeros, large integer values, and other control sequences. All coefficient sites incorporated into these values are avoided. In our implementation, we only select sites with translated indices ranging from 107 to 254, but excluding 180 (an invalid code).
- Generating a seed for random number generation and then choosing sites for modification: Sites from the candidate set $S$, that are modified, are selected in a pseudorandom fashion. To ensure that the encoder actions are invertible in the decoder, the seed for the random number generator is based on the subbands that are not considered for alteration. For example, in the selection process the contents of subbands 0–6 are left unchanged in order to minimize distortion. Typically, fixed sites within these bands are selected, although in principle any statistic from these bands may be computed and used as the seed. Selecting the seed in this way ensures that the message is embedded at varying locations (based on the image content). It further ensures that the embedded message can only be read if the proper seed selection algorithm is known by the decoder.
- Hiding the message at selected sites by bit setting: The message to be hidden is translated into a sequence of bits. Each bit will be incorporated into a site chosen pseudorandomly by a random number generator seeded as described above. That is, for each bit a site is selected from the set $S$ based on the next output of the seeded pseudorandom number generator. If the selected site has already been used, the next randomly generated site is chosen instead. The low order bit of the value at the selected site is changed to be identical to the cur-

Figure 8C 64 subbands of the image in Figure 8A



rent message bit. On average, half the time this results in no change at all of the coefficient value.

- Appending the bits to the coded image: Optionally, all the original low order bits can be saved and appended to the compressed bit stream as a user comment field (an appendix). The appended bits are a product of randomly selected low-order coefficient bits and hence these bits are uncorrelated with the hidden message.

The steps performed by the decoder correspond to the encoder steps above. The first two steps are identical to the first steps of the encoder. These steps construct the same set *S* and compute the same seed for the random number generator. The third step uses the pseudorandom number generator to select specific sites in *S* in the prescribed order. The least significant bits of the values at these sites are extracted and concatenated to recover the original message.

If the appendix restoration is to be included, the decoder can optionally restore the original low-order bits while reconstructing the message. This allows perfect reconstruction of the image (up to the original compression) despite the embedded message. Because the modification sites $S$ are carefully selected, the decompressed image even with the message still embedded will be nearly the same as the restored decompressed image. In practice, the error due to the embedded message is not perceptually significant and does not affect subsequent processing and authentication. Figures 8A and 8B show the original and the reconstructed images, respectively.

Using this process only a specialized decoder can locate and extract the message from the compressed image during the decoding process. This message might be a fixed authentication stamp, personal ID information which must match some other part of the record (which might have been sent in the clear), or some time stamp. Thus, if the bit stream does not contain an embedded message or the bit stream is improperly coded, the specialized decoder will fail to extract the expected message and will thus reject the image. If instead an unencoded WSQ compressed fingerprint image is submitted to the special decoder, it will still extract a garbage message which can be rejected by the server.

Many implementations of the same algorithm are possible by using different random number generators or partial seeds. This means it is possible to make every implementation unique without much effort; the output of one encoder need not be compatible with another version of the decoder. This has the advantage that cracking one version will not compromise any other version.

This method can also be extended to other biometric signals using a wavelet compression scheme, such as facial images or speech. While the filters and the quantizer in the WSQ standard have been designed to suit the characteristics of fingerprint images, wavelet-based compression schemes for other signals are also available.[20] It is relatively straightforward to design techniques similar to ours for such schemes.

## Image-based challenge/response method

Besides interception of network traffic, more insidious attacks might be perpetrated against an automated biometric authentication system. One of these is a replay attack on the signal from the sensor (attack point 2 in Figure 3). We propose a new method to thwart such attempts based on a modified challenge/response system. Conventional challenge/response systems are based either on challenges to the user, such as requesting the user to supply the mother's maiden name, or challenges to a physical device, such as a special-purpose calculator that computes a numerical response. Our approach is based on a challenge to the sensor. The sensor is assumed to have enough intelligence to respond to the challenge. Silicon fingerprint scanners[21] can be designed to exploit the proposed method using an embedded processor.

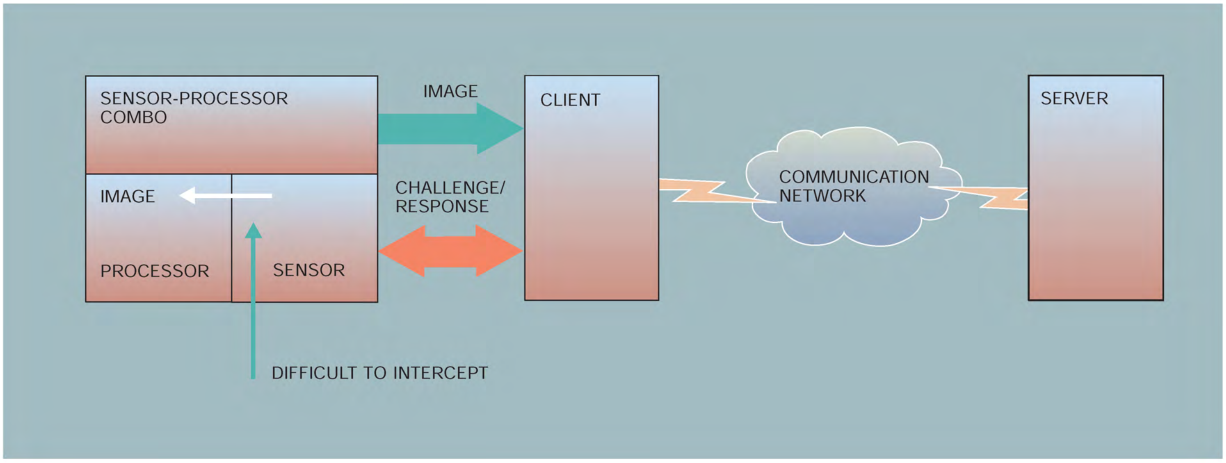
Note that standard cryptographic techniques are not a suitable substitute. While these are mathematically strong, they are also computationally intensive and could require maintaining secret keys for a large number of sensors. Moreover, the encryption techniques cannot check for liveness of a signal. A stored image could be fed to the encryptor, which will happily encrypt it. Similarly, the digital signature of a submitted signal can be used to check only for its integrity, not its liveness.

Our system computes a response string, which depends not only on the challenge string, but also on the content of the returned image. The changing challenges ensure that the image was acquired after the challenge was issued. The dependence on image pixel values guards against substitution of data after the response has been generated.

The proposed solution works as shown in Figure 9. A transaction is initiated at the user terminal or system. First, the server generates a pseudorandom challenge for the transaction and the sensor. Note that we assume that the transaction server itself is secure. The client system then passes the challenge on to the intelligent sensor. Now, the sensor acquires a new signal and computes the response to the challenge that is based in part on the newly acquired signal. Because the response processor is tightly integrated with the sensor (preferable on the same chip), the signal channel into the response processor *is assumed ironclad and inviolable*. It is difficult to intercept the true image and to inject a fake image under such circumstances.

As an example of an image-based response, consider the function "x1+" which operates by appending pixel values of the image (in scan order) to the end of the challenge string. A typical challenge might be "3, 10, 50." In response to this, the integrated pro-

Figure 9 Signal authentication based on challenge/response



cessor then selects the 3rd, 10th, and 50th pixel value from this sequence to generate an output response such as "133, 92, 176." The complete image as well as the response is then transmitted to the server where the response can be verified and checked against the image.

Other examples of responder functions include computing a checksum of a segment of the signal, a set of pseudorandom samples, a block of contiguous samples starting at a specified location and with a given size, a hash of signal values, and a specified known function of selected samples of the signal. A combination of these functions can be used to achieve arbitrarily complex responder functions. The important point is that the response depends on the challenge and the image itself.

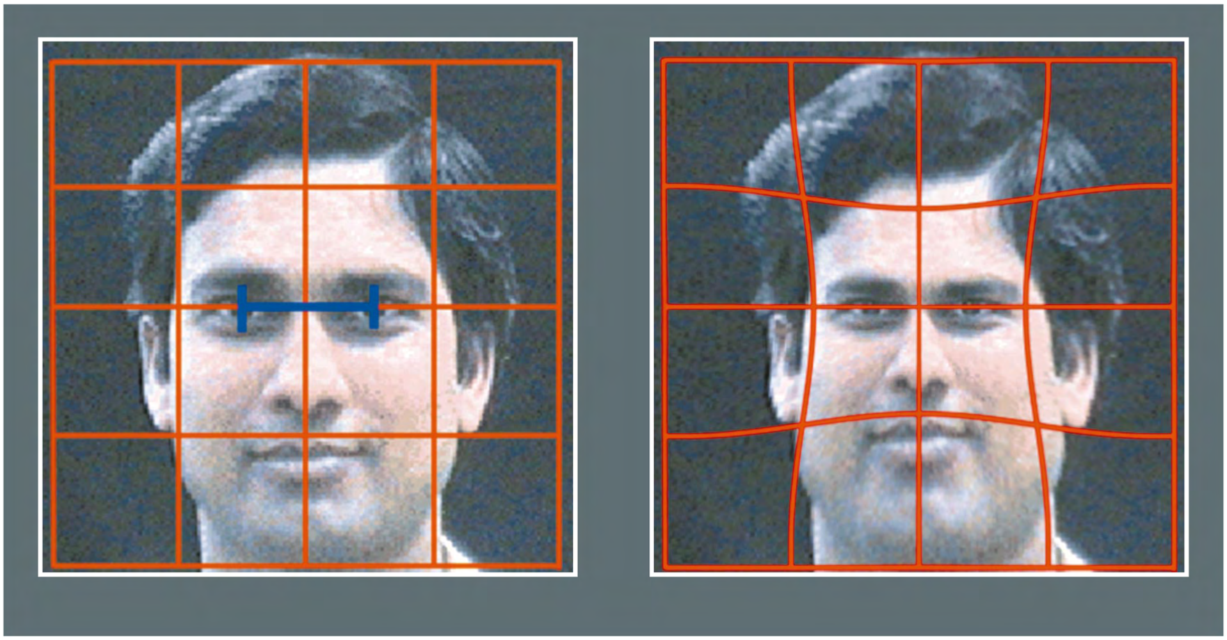
The responder can also incorporate several different response functions, which the challenger could select among. For instance, the integrated processor might be able to compute either of two selectable functions, "x1+" and "x10+." The function "x10+" is similar to "x1+" except it multiplies the requested pixel values by 10 before appending them. Financial institution *A* might use function "x1+" in all its units, while institution *B* might use "x10+" in all of its units. Alternatively, for even numbered transactions, function "x10+" might be used, and for odd numbered transactions "x1+" might be used. This variability makes it even harder to reconstruct the structure and parameters of the response function. Large numbers of such response functions are possible because we have a large number of pixels and many simple functions can be applied to these pixels.

## Cancelable biometrics

Deploying biometrics in a mass market, like credit card authorization or bank ATM access, raises additional concerns beyond the security of the transactions. One such concern is the public's perception of a possible invasion of privacy. In addition to personal information such as name and date of birth, the user is asked to surrender images of body parts, such as fingers, face, and iris. These images, or other such biometric signals, are stored in digital form in various databases. This raises the concern of possible sharing of data among law enforcement agencies, or commercial enterprises.

The public is concerned about the ever-growing body of information that is being collected about individuals in our society. The data collected encompass many applications and include medical records and biometric data. A related concern is the coordination and sharing of data from various databases. In relation to biometric data, the public is, rightfully or not, worried about data collected by private companies being matched against databases used by law enforcement agencies. Fingerprint images, for example, can be matched against the FBI or INS (Immigration and Naturalization Service) databases with ominous consequences.

Figure 10 Distortion transform based on image morphing



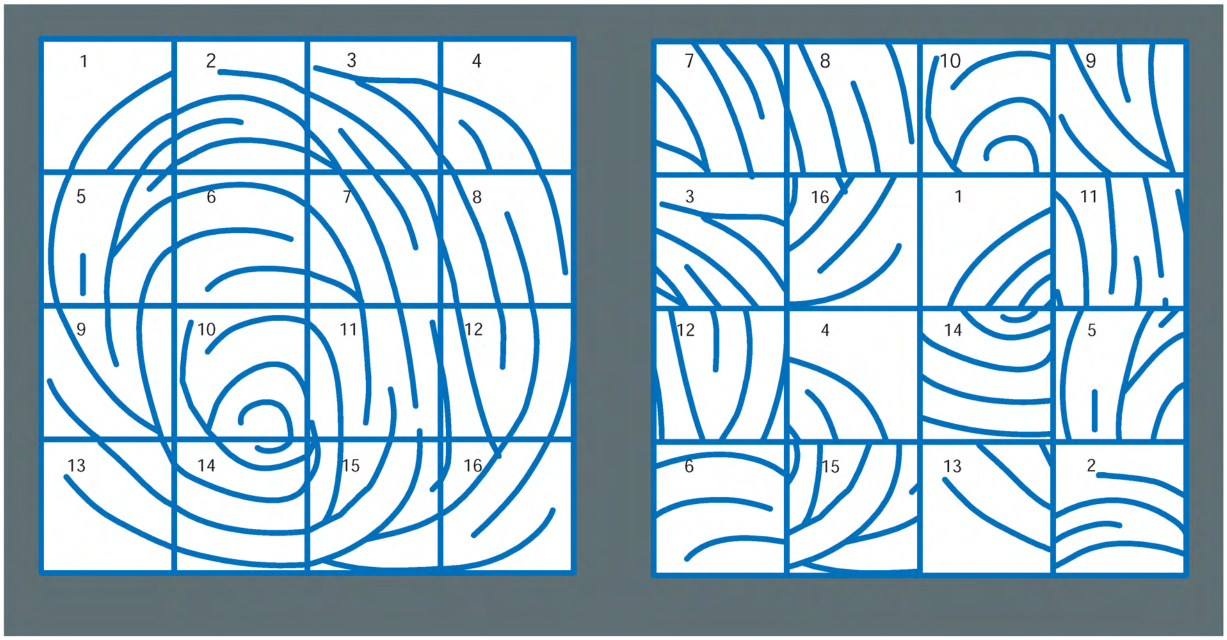
These concerns are aggravated by the fact that a person's biometric data are given and cannot be changed. One of the properties that makes biometrics so attractive for authentication purposes—their invariance over time—is also one of its liabilities. When a credit card number is compromised, the issuing bank can just assign the customer a new credit card number. When the biometric data are compromised, replacement is not possible.

In order to alleviate this problem, we introduce the concept of "cancelable biometrics." It consists of an intentional, repeatable distortion of a biometric signal based on a chosen transform. The biometric signal is distorted in the same fashion at each presentation, for enrollment and for every authentication. With this approach, every instance of enrollment can use a different transform thus rendering cross-matching impossible. Furthermore, if one variant of the transformed biometric data is compromised, then the transform function can simply be changed to create a new variant (transformed representation) for re-enrollment as, essentially, a new person. In general, the distortion transforms are selected to be noninvertible. So even if the transform function is known and the resulting transformed biometric data are known, the original (undistorted) biometrics cannot be recovered.

**Example distortion transforms.** In the proposed method, distortion transforms can be applied in either the signal domain or the feature domain. That is, either the biometric signal can be transformed directly after acquisition, or the signal can be processed as usual and the extracted features can then be transformed. Moreover, extending a template to a larger representation space via a suitable transform can further increase the bit strength of the system. Ideally the transform should be noninvertible so that the true biometric of a user cannot be recovered from one or more of the distorted versions stored by various agencies.

Examples of transforms at the signal level include grid morphing and block permutation. The transformed images cannot be successfully matched against the original images, or against similar transforms of the same image using different parameters. While a deformable template method might be able to find such a match, the residual strain energy is likely to be as high as that of matching the template to an unrelated image. In Figure 10, the original image is shown with an overlaid grid aligned with the features of the face. In the adjacent image, we show the morphed grid and the resulting distortion of the face. In Figure 11, a block structure is imposed on the image aligned with characteristic points. The

Figure 11 Distortion transform based on block scrambling



blocks in the original image are subsequently scrambled randomly but repeatably. Further examples of image morphing algorithms are described in References 22 and 23.

An example of a transform in the feature domain is a set of random, repeatable perturbations of feature points. This can be done within the same physical space as the original, or while increasing the range of the axes. The second case provides more brute force strength as was noted in Section 4 (this effectively increases the value of $K$). An example of such a transform is shown in Figure 12. Here the blocks on the left are randomly mapped onto blocks on the right, where multiple blocks can be mapped onto the same block. Such transforms are noninvertible, hence the original feature sets cannot be recovered from the distorted versions. For instance, it is impossible to tell which of the two blocks the points in composite block B, D originally came from. Consequently, the owner of the biometrics cannot be identified except through the information associated with that particular enrollment.

Note that for the transform to be repeatable, we need to have the biometric signal properly registered before the transformation. Fortunately, this problem has been partially answered by a number of techniques available in the literature (such as finding the "core" and "delta" points in a fingerprint, or eye and nose detection in a face).

**Feature domain transforms.** We present here an example of a noninvertible transform of a point pattern. Such a point pattern could, for example, be a fingerprint minutiae set

$$S = \{(x_i, y_i, \theta_i), i = 1, \ldots, M\} \tag{14}$$

However, this point set could also represent other biometrics, for example, the quantized frequencies and amplitudes of a speech pattern. A noninvertible transform maps this set $S$ into a new set $S'$ in such a fashion that the original set $S$ cannot be recovered from $S'$, i.e.,

$$S = \{(x_i, y_i, \theta_i), i = 1, \ldots, M\} \rightarrow S' = \{(X_i, Y_i, \Theta_i), i = 1, \ldots, M\} \tag{15}$$

Figure 13 shows how the $x$ coordinates of the point set $S$ can be transformed through a mapping $x \rightarrow$

Figure 12 Distortion transform based on feature perturbation

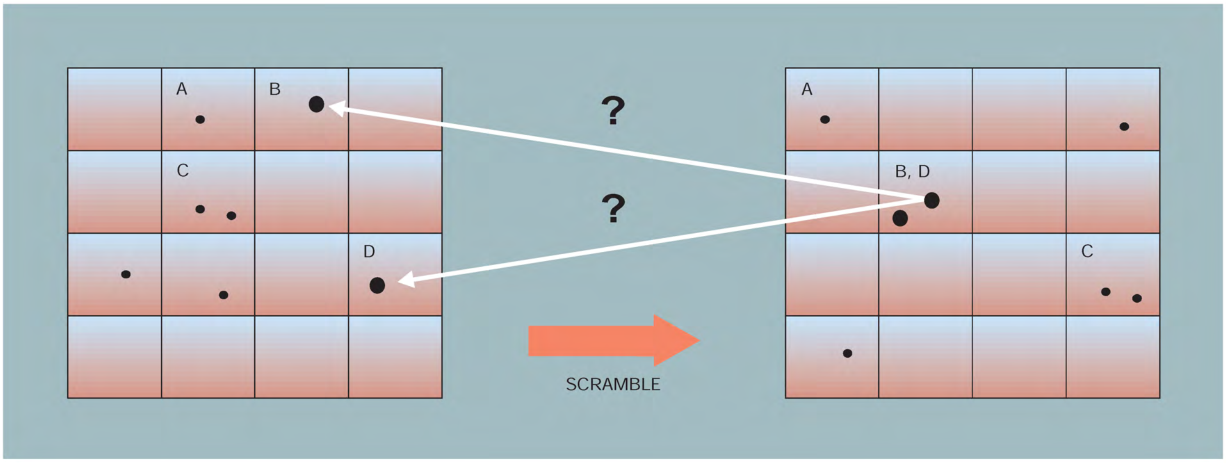
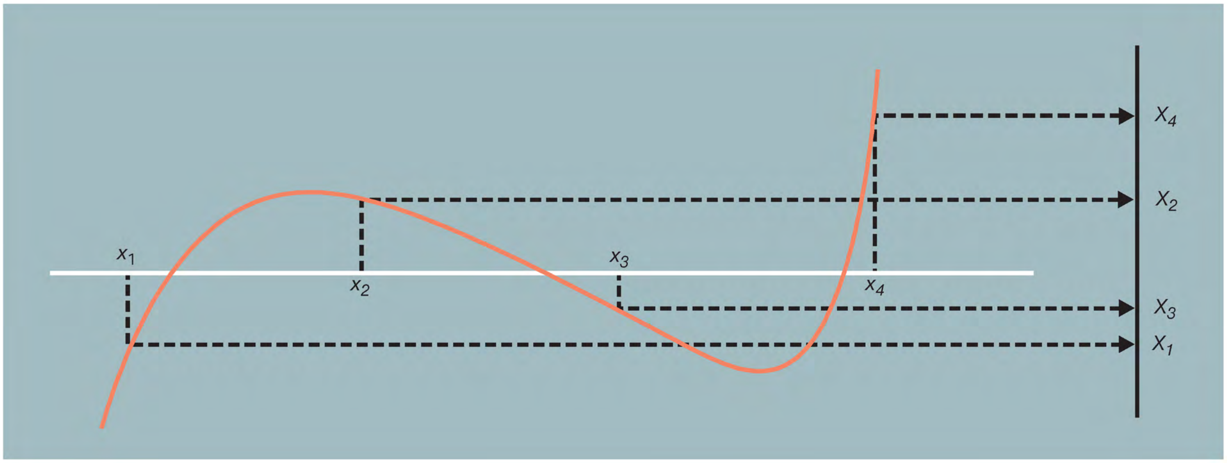


Figure 13 Example of noninvertible feature transform



$X$, or $X = F(x)$. This function of $x$ can, for example, be a high-order polynomial

$$X = F(x) = \sum_{n=0}^{N} \alpha_n x^n = \prod_{n=0}^{N} (x - \beta_n) \qquad (16)$$

The mapping $x \rightarrow X$ is one-to-one, as is seen from Figure 13. However, it is seen that the mapping $X \rightarrow x$ is one-to-many. For instance, the output value $X_1$ could be generated from three different input $x$'s. Hence, this transform is noninvertible and the original features $x$ cannot be recovered from the $X$ values.

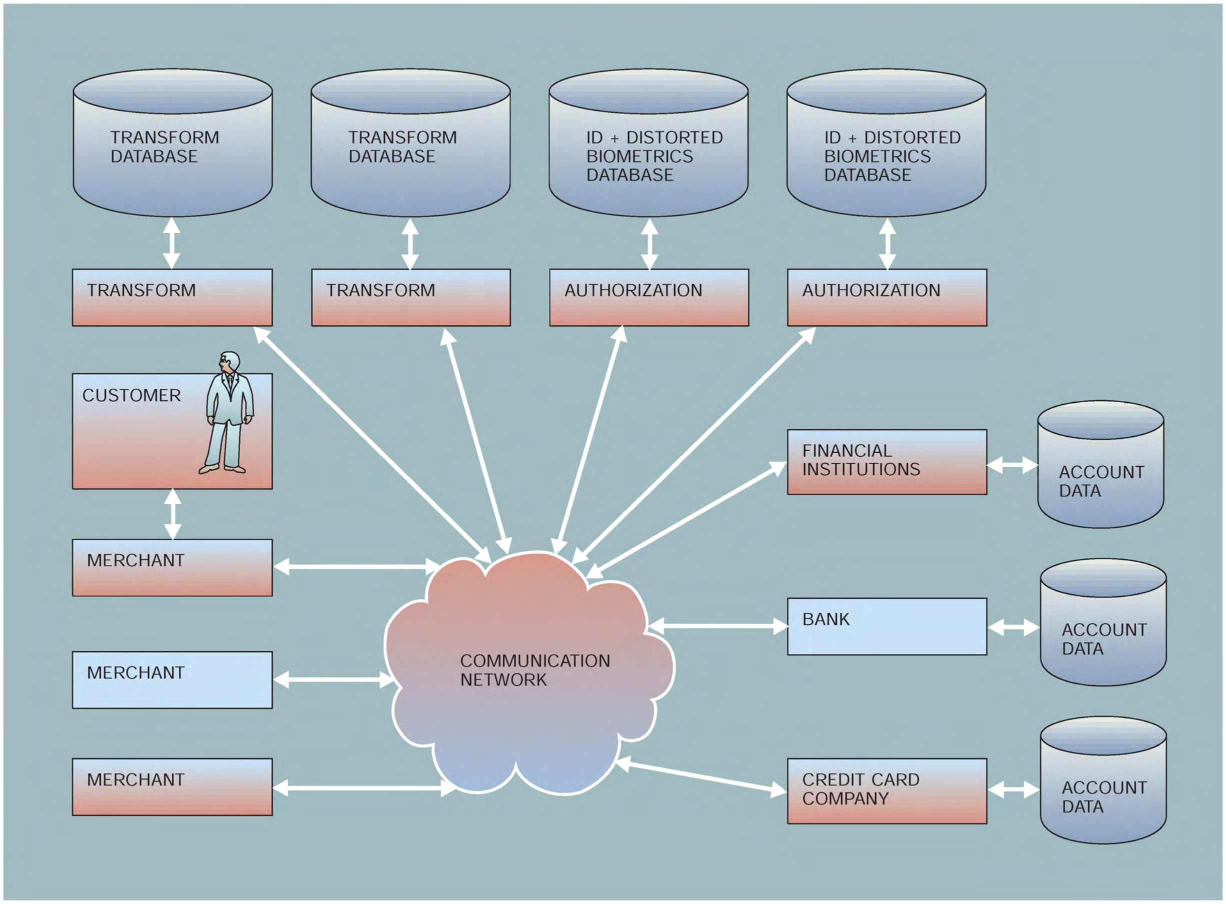
Similar polynomial noninvertible transforms

$$Y = G(y) \quad \text{and} \quad \Theta = H(\theta) \qquad (17)$$

can be used for the other coordinates of the point set.

**Encryption and transform management.** The techniques presented here for transforming biometric sig-

Figure 14 Authentication process based on cancelable biometrics



nals differ from simple compression using signal or image processing techniques. While compression of the signal causes it to lose some of its spatial domain characteristics, it strives to preserve the overall geometry. That is, two points in a biometric signal before compression are likely to remain at comparable distance when decompressed. This is usually not the case with our distortion transforms. Our technique also differs from encryption. The purpose of encryption is to allow a legitimate party to regenerate the original signal. In contrast, distortion transforms permanently obscure the signal in a noninvertible manner.

When employing cancelable biometrics, there are several places where the transform, its parameters, and identification templates could be stored. This leads to a possible distributed process model as shown in Figure 14. The "merchant" is where the primary interaction starts in our model. Based on the customer ID, the relevant transform is first pulled from one of the transform databases and applied to the biometrics. The resulting distorted biometrics is then sent for authentication to the "authorization" server. Once the user's identity has been confirmed, the transaction is finally passed on to the relevant commercial institution for processing.

Note that an individual user may be subscribed to multiple services, such as e-commerce merchants or banks. The authentication for each transaction might be performed either by the service provider itself, or by an independent third party. Similarly, the distortion transform might be managed either by the authenticator or by still another independent agency. Alternatively, for the best privacy the transform might remain solely in the possession of the user, stored, say, on a smart card. If the card is lost or stolen, the stolen transform applied to another person's biometrics will have very little impact. However, if the transform is applied to a stored original biometrics signal of the genuine user, it will match against the stored template of the person. Hence "liveness" detection techniques (such as described earlier) should be added to prevent such misuse.

## Conclusions

Biometrics-based authentication has many usability advantages over traditional systems such as passwords. Specifically, users can never lose their biometrics, and the biometric signal is difficult to steal or forge. We have shown that the intrinsic bit strength of a biometric signal can be quite good, especially for fingerprints, when compared to conventional passwords.

Yet, any system, including a biometric system, is vulnerable when attacked by determined hackers. We have highlighted eight points of vulnerability in a generic biometric system and have discussed possible attacks. We suggested several ways to alleviate some of these security threats. Replay attacks have been addressed using data-hiding techniques to secretly embed a telltale mark directly in the compressed fingerprint image. A challenge/response method has been proposed to check the liveliness of the signal acquired from an intelligent sensor.

Finally, we have touched on the often-neglected problems of privacy and revocation of biometrics. It is somewhat ironic that the greatest strength of biometrics, the fact that the biometrics does not change over time, is at the same time its greatest liability. Once a set of biometric data has been compromised, it is compromised forever. To address this issue, we have proposed applying repeatable noninvertible distortions to the biometric signal. Cancellation simply requires the specification of a new distortion transform. Privacy is enhanced because different distortions can be used for different services and the true biometrics are never stored or revealed to the authentication server. In addition, such intentionally distorted biometrics cannot be used for searching legacy databases and will thus alleviate some privacy violation concerns.

## Cited references

1. B. Miller, "Vital Signs of Identity," *IEEE Spectrum* **31**, No. 2, 22–30 (1994).
2. L. O'Gorman, "Practical Systems for Personal Fingerprint Authentication," *IEEE Computer* **33**, No. 2, 58–60 (2000).
3. R. Germain, A. Califano, and S. Colville, "Fingerprint Matching Using Transformation Parameter Clustering," *IEEE Computational Science and Engineering* **4**, No. 4, 42–49 (1997).
4. A. Jain, L. Hong, and S. Pankanti, "Biometrics Identification," *Communications of the ACM* **43**, No. 2, 91–98 (2000).
5. B. Schneier, "The Uses and Abuses of Biometrics," *Communications of the ACM* **42**, No. 8, 136 (1999).
6. N. K. Ratha and R. M. Bolle, "Smart Card Based Authentication," in *Biometrics: Personal Identification in Networked Society*, A. K. Jain, R. M. Bolle, and S. Pankanti, Editors, Kluwer Academic Press, Boston, MA (1999), pp. 369–384.
7. B. Schneier, "Security Pitfalls in Cryptography," *Proceedings of the CardTech/SecureTech Conference*, CardTech/SecureTech, Bethesda, MD (1998), pp. 621–626.
8. B. Schneier, *Applied Cryptography*, John Wiley & Sons, Inc., New York (1996).
9. J. W. Osterburg, T. Parthasarathy, T. E. S. Raghavan, and S. L. Sclove, "Development of a Mathematical Formula for the Calculation of Fingerprint Probabilities Based on Individual Characteristics," *Journal of the American Statistical Association* **72**, 772–778 (1977).
10. S. L. Sclove, "The Occurrence of Fingerprint Characteristics as a Two Dimensional Process," *Journal of the American Statistical Association* **74**, 588–595 (1979).
11. D. A. Stoney, J. I. Thronton, and D. Crim, "A Critical Analysis of Quantitative Fingerprint Individuality Models," *Journal of Forensic Sciences* **31**, No. 4, 1187–1216 (1986).
12. *WSQ Gray-Scale Fingerprint Image Compression Specification*, *IAFIS-IC-0110v2*, Federal Bureau of Investigation, Criminal Justice Information Services Division (1993).
13. N. Memon and P. W. Wong, "Protecting Digital Media Content," *Communications of the ACM* **41**, No. 7, 35–43 (1998).
14. F. A. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information Hiding—A Survey," *Proceedings of the IEEE* **87**, No. 7, 1062–1078 (1999).
15. W. Bender, D. Gruhl, N. Morimoto, and A. Lu, "Techniques for Data Hiding," *IBM Systems Journal* **35**, Nos. 3&4, 313–336 (1996).
16. M. D. Swanson, M. Kobayashi, and A. H. Tewfik, "Multimedia Data Embedding and Watermarking Technologies," *Proceedings of the IEEE* **86**, No. 6, 1064–1087 (1998).
17. C. T. Hsu and J. L. Wu, "Hidden Digital Watermarks in Images," *IEEE Transactions on Image Processing* **8**, No. 1, 58–68 (1999).
18. M. Yeung and S. Pankanti, "Verification Watermarks on Fingerprint Recognition and Retrieval," *Journal of Electronic Imaging* **9**, No. 4, 468–476 (2000).
19. C. M. Brislawn, J. N. Bradley, R. J. Onyshczak, and T. Hopper, "The FBI Compression Standard for Digitized Fingerprint Images," *Proceedings of SPIE 2847*, 344–355 (1996).
20. M. Vetterli and J. Kovacevic, *Wavelets and Subband Coding*, Prentice Hall, Englewood Cliffs, NJ (1995).
21. T. Rowley, "Silicon Fingerprint Readers: A Solid State Approach to Biometrics," *Proceedings of the CardTech/SecureTech Conference*, CardTech/SecureTech, Bethesda, MD (1997), pp. 152–159.
22. G. Wolberg, "Image Morphing: A Survey," *The Visual Computer* **14**, 360–372 (1998).
23. T. Beier and S. Neely, "Feature-Based Image Metamorphosis," *Proceedings of SIGGRAPH*, ACM, New York (1992), pp. 35–42.

*Accepted for publication April 24, 2001.*

**Nalini K. Ratha** *IBM Research Division, Thomas J. Watson Research Center, 30 Saw Mill River Road, Hawthorne, New York 10532 (electronic mail: ratha@us.ibm.com).* Dr. Ratha is a research staff member in the Exploratory Computer Vision Group. He received his Ph.D. degree in computer science from Michigan State University in 1996, working in the Pattern Recognition and Image Processing Laboratory. His research interests include automated biometrics, computer vision, image processing, reconfigurable computing architectures, and performance evaluation.

**Jonathan H. Connell** *IBM Research Division, Thomas J. Watson Research Center, 30 Saw Mill River Road, Hawthorne, New York 10532 (electronic mail: jconnell@us.ibm.com).* Dr. Connell is a research staff member in the Exploratory Computer Vision Group. He received his Ph.D. degree in 1989 at the MIT Artificial Intelligence Laboratory, working with Rod Brooks on behavior-based mobile robot control. His research interests include robotics, vision, natural language, and complete artificial intelligence systems.

**Ruud M. Bolle** *IBM Research Division, Thomas J. Watson Research Center, 30 Saw Mill River Road, Hawthorne, New York 10532 (electronic mail: bolle@us.ibm.com).* Dr. Bolle is the founding manager of the Exploratory Computer Vision Group. He received his Ph.D. degree in electrical engineering from Brown University, Providence, Rhode Island in 1984. He is a Fellow of the IEEE and the AIPR and is a member of the IBM Academy of Technology. His research interests are focussed on video database indexing, video processing, visual human-computer interaction, and biometrics.