Anonymous Biometric Identifiers – Revisited

Rudolf M Bolle
Scheveningen, the Netherlands
bolle@alumni.brown.edu

December 1, 2015

Abstract

Biometrics is about identification and verification of identities, establishing trusted communication between a person and an application – such as, bank account, email account, and so on. Biometrics based authentication has numerous advantages over conventional password and PIN authentication. Truly accurate biometric authentication systems would mean that the nightmare of forgotten passwords will behind us. However, one only has finite biometric identifiers and if a biometric identifier is compromised multiple times one may have run out of biometric identifiers. A solution to this is *A*nonymous Biometrics, which takes care of the anonymity of the representation. One more thing *R*evocability, is needed when a biometric is compromised, it should be possible to issue a new biometric. Things are complicated because of the randomness of biometric identifiers. Conventional encryption works very well on deterministic signals – i.e., text strings, numbers. There are problems encrypting biometric identifiers because of their randomness and they are often not encrypted. This results in lack of anonymity and revocability of biometric identifiers and limits acceptance. A biometric identifier is not anonymous, because they are not encrypted due to their stochastic nature. We propose a means to thoroughly encrypt biometric identifiers, where we focus on fingerprint images. The two-dimensional nature in the form of minutiae scattered through the print is converted to a one-dimensional signal using exhaustive enumeration of features. A representation of a processed fingerprint as a sparse binary string does not require registration of two fingerprints prior to matching. Verifying fingerprints amounts to comparing binary strings. *R*andom Permutation is discussed, which assigns a unique identifier to each enrolled subject and permutes the binary string based on the identifiers for randomized matching. The described technique is an anonymous and revocable fingerprint representation that does not require registration before matching. We introduce modeling at 2 resolutions, which is just concatenation of 2 sparse strings 1 obtained from the coarse resolution, 1 from the fine.

1 Introduction

The term 'biometrics' is taken to mean the identification of individuals based on a physical characteristic. Biometrics-based authentication schemes using fingerprints, iris recognition, face recognition, and so on, offer elusive usability advantages over password based authentication. It is not necessary to have to remember dozens of passwords and one cannot lose a biometric – like it is to forget a password. Well implemented biometric authentication could reliable replace traditional authentication schemes like passwords or ID cards. Successful implementation of a biometric authentication scheme would mean that the nightmare of forgotten passwords a thing most of us suffer from – will be over.

A biometric identifier is a sample of one of human's pertinent characteristics. The biometric should be unique to each person. There is a host of biometrics, from iris, to blood vein patterns, hand biometric (measurements take of the hand), etc. [**?**, 4]. However, there are large differences in the uniqueness of these different biometrics. The hand geometry biometric, for instance, is hardly discriminating and some say it is not a biometric [**?**]. The two biometric identifiers that perhaps show most promise are fingerprint and iris – never have two people been found with the same fingerprint [3]; for iris recognition, never a false accept has occurred [1]. The biometric identifier associated with a person can be compromised but, at least not forgotten. A biometric does not change and therefore it is like having a password that cannot be changed, as opposed to (say) credit cards that can be canceled and a new credit card number is issued. Despite these drawbacks, proponents of biometric authentication tout biometrics as secure authentication because (1) they are unique over the population, and (2) they do not change over the lifespan of an individual. In most papers on biometric authentication (identification or verification), the above two properties of a biometric are given as an argument for the use of biometrics.

We concentrate on fingerprints which satisfy the many conditions of a biometric (see, for example [**?**]). Among other things, (1) fingerprints are more or less constant over a persons lifespan – barring, for example, as often mentioned, the fingerprints of a brick layer and other types of damaged fingerprints; and, (2) for "good" fingers, impressions are relatively easy to acquire. For a certain percentage of the population, the fingers are dry or wet but that can be remedied. The ridges, however, could be too tiny to produce good prints [3]. Fingerprint authentication has an enormous install base mainly for forensic purposes; the situation in the consumer market is unclear. Financial institutions, health care providers, retailers, and governments are all looking into ways to become more secure using biometrics. This is the new frontier: mainstream biometric identification is so cutting-edge that there is no universally-agreed-to standard to build upon. Initial adopters of personal biometric security, such as Apples TouchID and Samsungs biometric sensor, wanted to alleviate their customers concerns over private and public-sector biometric collection, by opting to store biometrics locally on the device. Unfortunately, this method can be a nightmare for both the company and the end-user due to lack of security on individual devices. Recently, hackers declared they can remotely hack into Android devices and hijack the device-stored fingerprint. Whether its remote hacking, or the theft of the actual device, once a hacker has access to the device, they also have a lot of the data about who the person is.

A processed fingerprint is a collection of minutiae $\{m, m_1, m_2, ...\}$. This has become the *d*efacto standard for the representation of a fingerprint (see Figure 1, a list of minutiae associated with properties, the most well known representation is (x, y, θ), the location and orientation of the minutiae. Now all invariant triangles of minutiae are converted to an new index, $((s_1), s_2, \alpha_1, \alpha_2)$. This is shown in Figure 1, the fingerprint a collection of triplets of minutiae from which an invariant index I is computed and de fingerprint representation is $\{I_1, ..., I_M\}$ with M the number of triangles.

1in

Figure 1: A triplet of minutiae is encoded so that the triplet is invariant to translation and rotation.

The size of the image is finite and the triangles have to satisfy certain criteria; a simple criteria is that a triplet lie within de bounds of the image. The representation of a fingerprint is long sparse binary string – but triangles have certain criteria and therefore only a portion of a string with all possible triangles is occupied. More about this can be found in Section . There are some subtle problems when encrypting stochastic biometric identifiers like fingerprints, because they are slightly different for each acquisition – they are simply stochastic variables. And it is hard to encrypt such signals. There is a large body of encryption techniques, having in common that encryption works very well on deterministic signals – i.e., text strings, numbers. But because of ???, conventional encryption is impossible. Therefore a biometric identifier is not anonymous. The anonymity is lost because the fingerprint is out in the open and can be matched with other fingerprints. Further, one only has finite biometric identifiers and if a biometric identifier is compromised one may have run out of biometric identifiers. So the biometric representation cannot be replaced by issuing another biometric.

A biometric identifier, that is, a machine representation of an acquired biometric signal for access control needs to be encrypted. In traditional password based authentication, a password is not decrypted and upon an authentication request, matching between the encrypted input password and a stored encrypted password is performed in the encrypted domain, without decrypting. Ideally, like password matching, matching of biometric identifier signals, which are stochastic this time should somehow be done in the encrypted domain. A representation of a processed fingerprint as a long sparse binary string is presented. This is achieved by enumerating triplets of minutiae.

We need another type of encryption than conventional encryption because the hash function is random too. We call this *random permutation*. We propose a means to thoroughly encrypt biometric identifiers, where we focus on fingerprint images.

Fingerprint images are represented each by one sparse binary string. The two-dimensional nature in the form of minutiae scattered through the print is converted to a one-dimensional signal using exhaustive enumeration of features of the minutiae. A novel representation of fingerprints as binary strings, which do not require registration or alignment of fingerprints prior to verification. Then, verifying fingerprints amounts to comparing binary strings using some sort of distance measure. To the best of our knowledge, our technique is the first anonymous and revocable fingerprint representation that does not require any registration before matching. Using the binary string representation of the fingerprint, we describe techniques for creating anonymous and revocable representations. The performance is evaluated using a large database of fingerprint images. If a representation is compromised, it is computationally infeasible to invert it to the original fingerprint, thereby preserving privacy of the fingerprint.

The random permutation encryption of a fingerprint introduced here, is achieved by randomly permuting the long binary string that represents a fingerprint. Lack of anonymity and revocability of biometric identifiers limit their acceptance. We show that randomized encryption, which also leads to randomized search (matching) and leads to encryption that takes an infeasible amount of computation to decrypt. Biometrics-based authentication schemes using fingerprints, face recognition, etc., offer usability advantages. However, despite its obvious advantages, the use of biometrics raises several security and privacy concerns. Unlike passwords and cryptographic keys, biometrics are not concealed and can be easily misused without the user's consent. In addition, unlike PINS and credit card numbers, biometrics are permanently associated with an individual and cannot be canceled and reset if compromised. This leads to the necessity of anonymous biometric [**?**] systems which could afford the usability advantages of biometrics in addition to the security and privacy advantages of the conventional key-based systems. This would require *cancelable biometrics* (also known as revocable biometrics) instead of the plain and obvious ones. Conceptual frameworks for cancelable biometrics have been presented in [**?**, **?**, **?**] and a comprehensive review can be found in [**?**].

This paper presents method a technique for converting a fingerprint into a binary sparse histogram based on minutiae triplets derived from a processed fingerprint image. This binary representation is then randomly permuted, using a key assigned to every individual enrolled in a user database. This transform is not only computationally infeasible to invert, but can be revoked in case of a compromise by simply assigning a different key (which could be the seed of a PRNG). The system has been tested on an old very difficult data set called IBM99, described in Section **??** and discuss the strength of the system in Section **??**. We draw the conclusion in the final section and briefly discuss possible future work.

2 Anonymous biometrics

The importance of anonymous biometrics was pointed out in [**?**]. The one major requirements of anonymous biometrics, or rather of anonymous biometric representations is number 1 –

1. *A*nonymity: To preserve privacy a very first requirement is that, whatever the stored representation of the biometric is, it is impossible or very hard to reconstruct the original biometric from the representation. In essence, the representation needs to be completely unrelated to or uncorrelated with the original representation. This to a great extend might alleviate the privacy concerns that some people may have with biometrics. It may increase the acceptability of biometrics when properly explained to the public.

   Not mentioned in the NSF report [**?**] are the following two important issues.

2. *R*evocability: If the stored representation is compromised, it should be possible to cancel the anonymous representation and to re-enrol the individual with a new anonymous representation.

   This property of anonymous biometrics prohibits cross-matching between databases. After all, an individual can enroll with different fingerprints in different databases. It will be impossible to correlate (say) health records with insurance databases.

3. *M*ultiplicity: An individual, or rather a biometric, can enroll in different databases or enroll in the same database multiple times using different identities. The different representations of the same biometric do not match each other in the anonymous domain.

Many fingerprint matching algorithms extract minutiae from both the fingerprint (the enrolled print) and

We present a fingerprint representation that does not require to find registration points, prior to matching two prints.

The *d*e facto first step for extracting features for recognition from a print is extraction of minutiae. A fingerprint is processed by finding ridges, curved pieces of ridge with ridge endings and bifurcations, which are called minutiae, see Figure . Possible the delta of the ridge structure is also computed. The minimal fingerprint representation is a set of minutiae (x_i, y_i, θ_i), where i is de index of the i-the minutiae derived from the ridge structure, see Figure **??**. Traditional fingerprint matching amounts to registering two sets of minutiae, for example [5].

The idea is not new and is based on the fact that processed fingerprints can be represented by a set of invariant features, triplets of minutiae. From the set of minutiae, invariant features formed by triplets of minutiae features are used for matching as successfully demonstrated by Germain et al [2].

3 Revocable Biometrics

Revocability depends on the choice of invariant features for generating binary strings from the fingerprint images. Later on, we describe the transformation of the strings based on keys assigned to individuals that allows the revocability of the templates by issuing a new key.

To extract reliable recognition features from the fingerprint they should not be sensitive to any rotation and translation. This is due to the fact that there is considerable intra-class variation from one print to another. This is motivated by the early work by Germain *e*t al. [2]. We extract invariant features from minutia triplets. The geometry of the triangle in Figure **??** formed by the triplets does not change considerably under rigid transformation. There can be multiple invariants, but we limit ourselves to three sides of the triangle and the three angles of the minutia at the vertices. Figure **??** describes the 6 invariants used in the generation of the feature vector from the fingerprint.

The invariants are compute to compensate fo any variations caused by the distortions during the capture of prints of a finger. Whereas too little quantization will be sensitive to slight distortions; loose or coarse quantization will result in losing the discriminative power of the invariants because too few triangles match. To strike a balance between accuracy and efficiency of the system the second configuration of quantization, *viz.* 20 pixels for the lengths and $45°$ for the angles, proved to be the best. We represent each side by 4 bits, each angle and the height by 3 bits and then append them. Thus, each triangle is represented by a total of 24 bits. We calculate the 24 bit number for all the triplets in a fingerprint and generate a histogram of length $n = 2^{24}$ for all such triangles. We filter the triangles that are too small based on a threshold that is empirically evaluated. This results in each fingerprint being represented by a histogram (say F). We make a binary histogram by a simple rule given by

(1)

Thus, for every fingerprint in the database we have a binary vector.

4 Sparse binary histogram representation

Once the binary sparse histograms B are acquired for 2 fingerprints, matching becomes a simple thing. The matching score between two templates is calculated by either defining a dissimilarity measure, like histogram intersection; or, a distance measure like Hamming distance or the L_2 norm or other distance measures. Here we only handle the Hamming distance, not much differences between similarity and dissimilarity has been observed.

The Hamming distance is the faster, and easiest to implement, using set operations. This is achieved by simply counting the positions in the binary strings that have a value 1 at the same positions, amounting to set A; counting the positions that have a 1 in the first string and a 0 in the second string, or *v*ice versa, amounts to set B. Computing the Hamming distance is than a matter of set operations, if we have two strings S_1 and S_2, the Hamming distance h is

$$h(S_1, S_2) = \mid A \mid - \bigcap \mid B \mid = A \cap B. \tag{2}$$

The enrollment of authorized individuals into a database $\mathbf{B}$ is as follows–

Each individual in the database $\mathbf{B}$ is assigned a key, which could be chronological or random. In turn, this key is used to randomly shuffle the binary histogram and *o*nly store this in some enrollment database. Hence, only an encrypted fingerprint representation, i.e., a scrambled histogram, implemented by permuting the histogram, hence the term *r*andom permutation.

For verification, a subject presents her or his key and fingerprint, minutiae are extracted and minutiae triangles are formed. This gives a sparse histogram of triangles. Permuting this histogram based upon the user's key. The operations are performed in the same order as during enrollment. This guarantees that the print presented for verification is randomly permuted in the same manner as the enrolled . This random shuffling makes the brute force attack to invert the template intractable. The scoring technique and the experimental results are explained in Section **??**.

The matching score between two templates is calculated by either defining a dissimilarity measure, like histogram intersection; or, a distance measure like Hamming distance or the L_2 norm or other distance measures. Here only deal with the Hamming distance, not much differences between similarity and dissimilarity has been observed. The Hamming distance is the faster, and easy to implement, using set operations. This is achieved by simply counting the positions in the binary strings that have a value 1 at the same positions, amounting to set A; counting the positions that have a 1 in the first string and a 0 in the second string, or *v*ice versa, amounts to set B. Computing the Hamming distance is than a matter of set operations, if we have two strings S_1 and S_2, the Hamming distance h then is

5 Security

In an actual impostor situation, an individual **A (impostor) claims to be another person B. In a verification scenario, this is simulated by matching the fingerprint of A with the fingerprint of B. But in a two-factor authentication system, to claim to be A, B must have access to the key of A. This is a situation where B has lost his key (e.g. stored in card) and A has access to it. In addition, it could be that the key is stored in a database and is retrieved during the verification by supplying a username and in this case A is supplying B's username for verification and thus the system is retrieving B's key without revealing it to A. We performed this experiment by simply comparing two non-matching pairs from the 1000 individuals transformed with the same key in Section ??. Figure ?? shows the ROC curve. The EER in this case is 1.51% which is comparable to many plain-matching algorithms in the literature.**

In case of a lost card or a database breach where the key or the transformed template is stolen, we should be able to cancel the template and they key and assign the individual a new one. For now, let us assume (before discussion in Section ?? that in case the template is lost, the original fingerprint cannot be constructed from it. Then the security of the system can be restored by assigning a new key and thus a new transformed template for the user. In order to prove this statement, we performed the following experiment. We generated two binary templates from Section ?? for one individual's matching pair. Next we assigned n different keys to this individual and transformed both binary templates (enrolled as well as test) using these n different keys. To support our claim of revocability, these n templates should be different and this can be verified by conducting the experiment in ??, this time considering each template transformed by a different key as a different individual. In this experiment again we achieved an FAR (and thus EER) of 0% which means that the templates generated are essentially no similar than different individuals. This not only vindicates the claim of revocability, it also solves the problem of cross-matching in databases. Thus, one can enroll different templates using the same finger at different physical locations.

In case a template is revealed to an adversary, a brute force attack on the transformed template will require $2^{24}!$ attempts before the actual binary template can be acquired. This is definitely computationally infeasible in real time. This is much harder than the brute force attack on a plain fingerprint template which usually would have a complexity of about 70-80 bits [?]. Thus, it is much easier for the adversary to randomly place minutia on a plain surface and generate fingerprint templates rather than use a brute force attack on this system. Even after the binary template is acquired, it is not straight forward to reconstruct the fingerprint minutia representation due to the extra noise bits added during mutation in Section ?? and the missing details during binarization of the vector in Equation 1. Additionally, to acquire a compact and secure representation of the template generated above in Section ??, we can store only the addresses of the locations where the bit is set to 1. This representation can then be secured by encrypting or hashing the addresses of these locations. Any hashing technique like MD5 or SHA-1 or a key-based encryption technique like RSA can be used. In our implementation we hashed these values using the standard MD5 algorithm that uses a 128-bit hash value and the error rates remained unchanged due to a low average collision

rate of 2^{50} in MD5. During verification, the hashed values can be compared and the scores calculated.

6 Experiments

The experiments are performed on a large data set (IBM99) that consists of $1,000$ individual fingerprints. Each fingerprint has two impressions one of which we used for enrolment and the second as the test fingerprint and *v*ice versa.

7 Conclusion

We revisited a fingerprint representation, a fingerprint matching technique where the 2-Dimensional fingerprint representation, i.e., a set of minutiae, to a sparse high-dimensional histogram or sting of minutiae triangles without the need of registration points. As noted before, there is nothing new to this. A fingerprint is represented by a set of invariant minutiae triangles, which is a just a small percentage of all possible triangles. A minutiae triangle has to satisfy certain constraints, for example, the triangle has to lie within the image borders, cannot be a equilateral and has to satisfy other constraints. This means that the representation becomes even sparser. But the number of possible triangles is finite. Exhaustively, all possible triangles are enumerated which gives a very long sparse high-dimensional histogram of minutiae triangles. Relatively few buckets are occupied. And if they are occupied it often by a 1 – indicating that the matching corresponding triangles are unique in each print; in rare occasions the buckets contain 2 or 3 or 4 triangles. These buckets are discarded, set to 0.

Matching two fingerprints now amounts to computing the distance between 2 sparse binary histograms or strings. Only the Hamming distance is used in this paper because one needs to keep the parameters constant among experiments and because the Hamming distance is fastest to compute.

Encryption of fingerprint images is achieved through permuting the fingerprint binary histogram many times. Hacking into the biometric authentication is computationally infeasible. Braking the permutation is $O(M^2)$, with M the number of possible minutiae triangles. Remember *M* is large, in the millions. Random permutation is governed by a seed, associated with an identity.

References

[1] J. G. Daugman. High confidence visual recognition of persons by a test of statistical independence. *IEEE Transanctions on Pattern Analysis and Machine Intelligence*, 15(11):1148–1161, Nov. 1993.

[2] B. Germain et al. Issues in large scale automatic biometric identification. In *IEEE Workshop on Automatic Identification Advanced Technologies*, pages 43–46, Stony Brook, NY, November 1996.

[3] FBI, U.S. Department of Justice, Washington, D.C. 20402. *The Science of Fingerprints, Classification and Uses*, 1984.

[4] A.K. Jain, R.M. Bolle, and S. Pankanti, editors. *Biometrics: Personal Identification in Networked Society*. Kluwer Academic Publishers, Boston, Mass., 1999.

[5] N.K. Ratha, V.D. Pandit, R.M. Bolle, and V. Vaish. Robust fingerprint authentication using local structural similarity. In *Fifth IEEE Workshop on Applications of Computer Vision*, pages 29–34, December 2000.