Randomized Radon Transforms for Biometric Authentication via Fingerprint Hashing

Mariusz H. Jakubowski and Ramarathnam Venkatesan
Microsoft Research
Redmond, WA, USA
{mariuszj, venkie}@microsoft.com

ABSTRACT

We present a new technique for generating biometric fingerprint hashes, or summaries of information contained in human fingerprints. Our method calculates and aggregates various key-determined metrics over fingerprint images, producing short hash strings that cannot be used to reconstruct the source fingerprints without knowledge of the key. This can be considered a randomized form of the Radon transform, where a custom metric replaces the standard line-based metric. Resistant to minor distortions and noise, the resulting fingerprint hashes are useful for secure biometric authentication, either augmenting or replacing traditional password hashes. This approach can help increase the security and usability of Web services and other client-server systems.

Categories and Subject Descriptors

I.4.9 [**Computing Methodologies**]: Image Processing and Computer Vision—*Applications*; D.2.11 [**Software Engineering**]: Software Architectures—*Information hiding*; E.3 [**Data**]: Data Encryption

General Terms

Algorithms, Human Factors, Security

Keywords

Biometrics, authentication, fingerprints, hashing, Radon transform

1. INTRODUCTION

Password-protected Web accounts and other secure sites have recently proliferated, requiring users to create and remember large quantities of passwords. Many users have addressed the resulting hassles with a variety of insecure tactics, such as choosing easily guessed passwords, as well as reusing and writing down secrets. While software exists to manage password-protected lists of passwords [19], this may be unwieldy and dangerous if master passwords are leaked or lost.

Biometric methods [20, 1, 12] have been proposed recently to alleviate the "too-many-passwords" problem, as well as to help with user authentication in general. Human features such as fingerprints, veins, and retinas can provide reasonably unique and robust identifiers for secure authentication. While biometrics has been used for high-security applications in the past, such methods have been implemented mainly for highly specialized, closed systems. Open systems such as PCs and the Web have somewhat different requirements, particularly in terms of fast verification that can be incorporated into relatively lightweight authentication protocols.

This paper presents a methodology for using human fingerprints (FPs) [10] for biometric authentication suitable for systems such as networked PCs. Our scheme involves quick computation of fingerprint hashes, or short strings that contain much of a fingerprint's uniqueness or entropy. With some modifications, the methods also apply to other human attributes, such as blood-vessel patterns in retinas or hands [8].

Our hashing method enables fingerprint matching without the need to store actual fingerprints or information useful for reconstructing them. Like the "secure sketches" produced by fuzzy extractors [4], fingerprint hashes capture the essence or entropy of fingerprint images, but act more like keyed cryptographic hashes. Secure fingerprint matching is also possible via other approaches, such as "chaffing" of fingerprint data [2]; our techniques are complementary and potentially useful more generally in other applications.

2. HASH GENERATION

2.1 General methodology

To produce an FP hash, our general method performs two main actions:

1. Preprocess the FP image into canonical form.
2. Compute a vector of various metrics over the FP image.

The preprocessing step aims to produce a canonical FP image suitable for reliable metric computation. We typically use low-pass and median filters along with thresholding to convert a noisy color or gray-scale FP image into a "clean"

DRM'07, October 29, 2007, Alexandria, Virginia, USA.

two-tone version. For better reliability, more involved techniques are helpful [15], particularly methods used for fingerprint scanning and forensics.

The metric-computation step essentially performs a one-way compression of the FP image into a short vector of pseudorandom numbers. Each element of this vector is a specially chosen metric evaluated over the canonical fingerprint image. A secret key provides the source of randomness used for determining metric types and their parameters. This also helps to enforce the one-way property, since an adversary lacking the key is unable to extract much nontrivial fingerprint information from the hash.

Examples of metrics suitable for FPs include the following:

- Number of crossings and tangents a line or curve segment makes with FP curves and whorls
- Number of FP minutiae [9] contained within a rectangular or circular FP region
- Area of the convex hull of minutiae contained within a given region

Our metric computation is a generalized form of the Radon transform [7, 3] that uses custom metrics to compute projections onto randomized lines. The standard Radon transform converts a two-dimensional image $I(x, y)$ into a matrix $R(m, b)$, where m and b denote slopes and y-intercepts of lines, respectively. A line with parameters (m, b) in $I(x, y)$ will lead to a high value of the coefficient $R(m, b)$.

Similarly, a line may be defined by an angle θ (slope) and a distance ρ (from the origin). As an example, fig. 5 shows such a Radon transform of the fingerprint in fig. 2. Displayed as shades of brightness, high values of coefficients (θ, ρ) indicate presence of lines with slopes θ and distances ρ in the fingerprint image.

Our biometric transform also computes projections of lines, but we use a small set of randomized line distances ρ and angles θ. Also, instead of a standard line-based metric, we use the count of crossings that a line makes with a fingerprint image, as well as other metrics suitable for hash computation on biometric data. Key-derived randomization is important to prevent an adversary from using crossing counts and other metrics to determine nontrivial information about the fingerprint. Unlike two-dimensional images, a fingerprint's features appear to be closer to one-dimensional; our transform is designed around this notion.

Since FP scans are subject to distortions and scanner-dependent artifacts, FP hashes may be inexact. For determining whether two FP hashes originated from the same FP, we may need to use some measure of distance between the hashes (e.g., Euclidean distance). In addition, we can enhance hash robustness by performing aggregation or error correction on the vector of metrics. This is similar to image hashing [21, 22, 14], but we specifically choose metrics that produce good results on FP images.

For an FP hash to be considered effective, hashes of two distinct FPs should be usually distinct or dissimilar, while hashes of an FP and its distorted version should be equal or close in distance. The experimental results we present in section 3 provide evidence that our scheme satisfies these requirements.

FP-based methods are subject to an entropy problem: Since there are approximately 2^{33} human beings, the entropy of all fingerprints may not be much more than 33 bits, especially given anecdotal forensic evidence of individuals possessing similar fingerprints. Thus, we need to randomize explicitly in order to achieve higher entropy in the hash values. This will be important to improve the accuracy of identification and security.

2.2 The Radon transform

We now motivate usage of the Radon transform for our construction.

Assume we have a smooth function f with a compact support. Now consider a transform

$$H \mapsto \hat{f}(H)$$

where H is a hyperplane, and $\hat{f}(H)$ is the average value of the function over the hyperplane H. The idea behind the Radon transform is that if one knows the values of $\hat{f}(H)$ as a function of H, then one can effectively reconstruct the function f. The values $\hat{f}(H)$ can be considered analogues of the frequency coefficients in the Fourier-transform domain. This has generalizations to arbitrary dimensions. For concreteness, we now recall the formulae in two dimensions for the forward and inverse Radon transforms of a function $f(x, y)$:

$$R(m, b)[f(x, y)] = \int_{-\infty}^{\infty} f(x, b + mx)dx \tag{1}$$

$$f(x, y) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \frac{d}{dy} Z[U(m, y - mx)]dm \tag{2}$$

In the above equations, the parameters m and b represent line slopes and y-intercepts, respectively. Z denotes a Hilbert transform [5], and $U(m, b) \equiv R(m, b)[f(x, y)]$.

The collection of hyperplanes naturally forms a projective space, where they can be given a topology, and thus one can vary H continuously. Also, the map $f \mapsto \hat{f}(H)$ needs a measure on the plane to do the integration. We change these two aspects to define our Radon-based transform.

We will pick our H randomly. The idea is that if we pick enough hyperplanes, the function will be uniquely defined. We will not study the invertibility aspect here. Secondly, the objects we integrate are not two-dimensional in nature. If they were (e.g., like images), then one may use a randomization akin to randlets [11], which uses a Gaussian and its derivatives as integration kernels. One can invert this transform using a Gram-Schmidt-type procedure called *pursuit algorithms*. We can imagine a fingerprint as a collection of curves with one-dimensional parametrization (to a first approximation). Thus, we choose lines for our hyperplanes, along with a *counting measure*, which simply counts how many times a (random) line intersects the curve.

The Radon transform has numerous applications, including computerized tomography. For a mathematical treatment, we refer the reader to [5].

2.3 Algorithms

The following is an example algorithm based on the above principles:

1. Preprocess the FP image to produce a two-tone version.
2. Using a cryptographic pseudorandom generator (e.g., the RC4 stream cipher [13, 18]), choose N line segments that cross the image. Let $s_1, s_2, ..., s_N$ denote these segments.

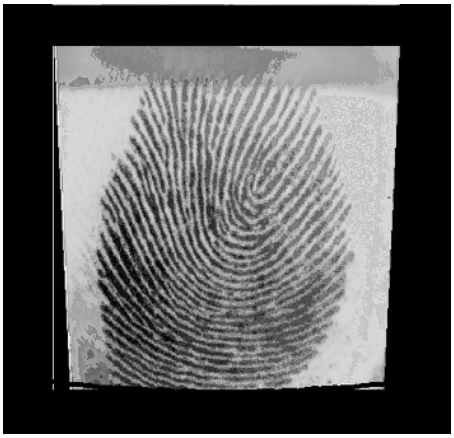

Figure 1: Original FP image.

Figure 2: Cleaned FP image.

Figure 3: Slightly distorted FP image.

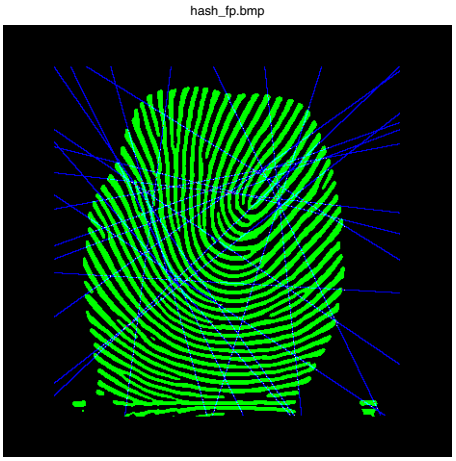

Figure 4: FP image showing lines for computing crossing counts.

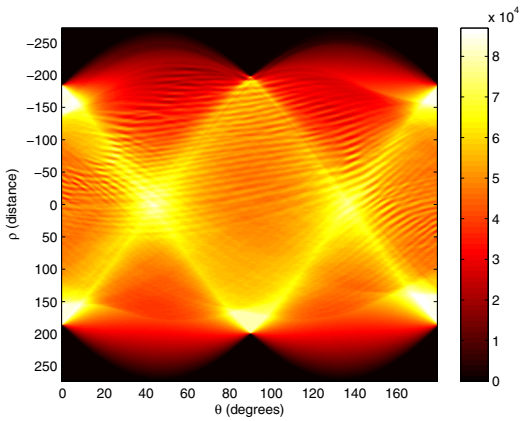

Figure 5: Radon transform of cleaned FP image.

3. For each segment s_i, compute the number of crossings and tangents with shapes in the FP image. Let c_i denote this number.

4. Return the hash vector $V = (c_1, c_2, ..., c_N)$.

Figs. 1–4 show the steps of this procedure on a sample FP distorted by simulated scanning. The original FP in fig. 1 is filtered and cleaned using VeriFinger software [15] to yield the FP in fig. 2, which undergoes StirMark [16] distortions to produce the FP in fig. 3. (Although StirMark is intended as an anti-watermark tool, we have found some of its transformations useful to approximate fingerprint-scanner distortions.) Fig. 4 shows the FP with a number of random line segments used for computing the crossing counts that comprise a hash vector.

This scheme is easy to implement and appears to work well with standard human FPs, as we show in section 3. Many variants are possible; for example, we may replace line segments with ellipses, parabolas, and other shapes. The precise choice of metrics may depend on the characteristics of FPs and FP scanners.

2.4 Hash usage

FP hashes may either augment or replace traditional password hashes in a variety of popular scenarios, such as system log-ons and Web-based authentication. In addition, FP hashes can increase security whenever a person's physical identity needs to be confirmed, such as for passport issuance and verification, secure access to buildings, purchase of restricted goods, and air travel. Such methodology can help verify FPs much like via zero-knowledge schemes [6], with minimum amount of FP-information leakage. The one-way nature of FP hashes also helps to alleviate potential privacy issues [17].

As in standard password management, a server can use a password file to store a list of user IDs and their corresponding FP hashes. For authentication, a user scans his FP, while the system computes its FP hash and matches this against all hashes in the password file. If the FP hashes are inexact, the system can match hashes based on minimum distance instead of absolute equality. The key used to compute each FP may be secretly fixed; alternately, for more security via two-factor authentication, each user may be required to enter a PIN or pass phrase to produce a key.

3. EXPERIMENTAL RESULTS

Using a small FP database, we have tested hash-generation schemes that use the previously described line-crossing metric. To evaluate hash effectiveness, we computed Euclidean distances between each hash and all other hashes of distinct FPs. We compare these results with the distances between each FP hash and the hash of the FP's distorted version. To simulate scanner artifacts, we used StirMark software [16] to perform random bending, noise addition, and other minor distortions.

Figs. 6 and 7 show distances between hashes of different FPs, along with distances between hashes of an FP and its distorted version. The horizontal axis shows the FP number $N = 1..23$; the vertical axis shows the distances between hashes. In each column, the diamond-shaped points show the distances between an FP hash and all other FP hashes; the square-shaped point shows the distance between the FP hash and the hash of the distorted FP.

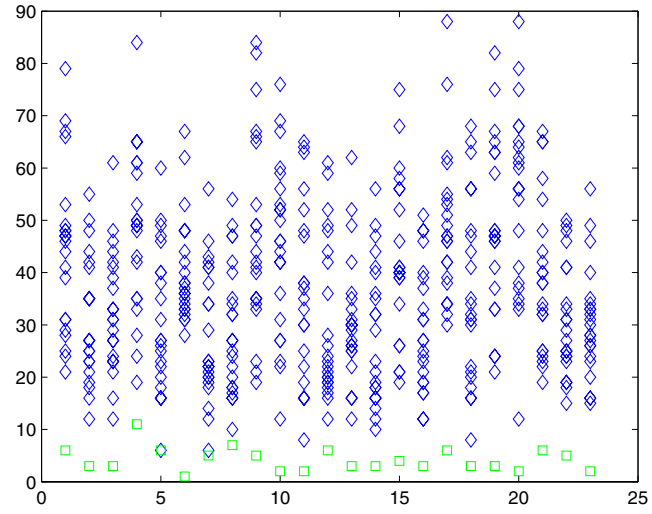

Figure 6: Hash distances for crossing counts computed over $N = 5$ random lines. The x-axis denotes fingerprint number $(1 - 23)$ from our set of samples, and the y-axis shows a simple distance metric between two fingerprints. The bottom (square) points indicate distances between each FP and its distorted version; the top (diamond) points indicate distances between the FP and other distinct FPs.

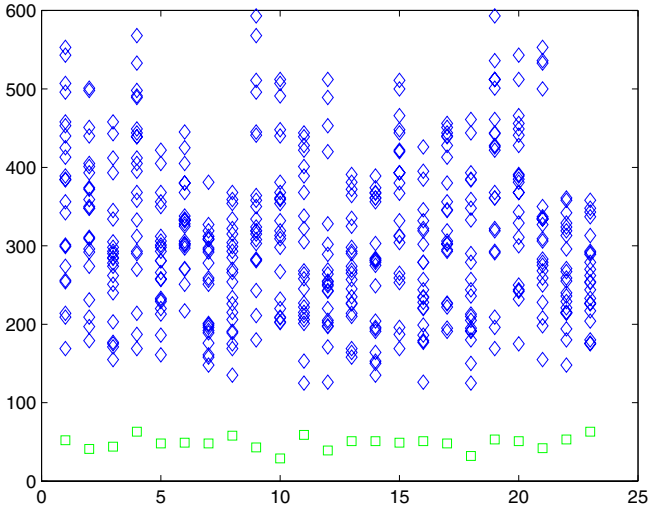

Figure 7: Hash distances for crossing counts computed over $N = 50$ random lines. The x-axis denotes fingerprint number $(1 - 23)$ from our set of samples, and the y-axis shows a simple distance metric between two fingerprints. The bottom (square) points indicate distances between each FP and its distorted version; the top (diamond) points indicate distances between the FP and other distinct FPs.

To distinguish FPs well, the diamonds should be well separated from the squares. In general, as we increase N, the results improve. $N = 5$ is not enough, as fig. 6 attests. Around $N = 50$, we can distinguish between different FPs reasonably well, as fig. 7 shows. These results are for only one particular sample of forensic FPs, but our experiments have worked similarly on several others. In practice, we choose N empirically to strike a balance between computational performance and diminishing returns as N increases. Future work may yield analytical methods to determine appropriate values for this parameter.

4. CONCLUSION

We have presented a new scheme for one-way biometric authentication that uses a randomized form of the Radon transform to compute fingerprint hashes. The technique can serve as a practical addition to increase the security of personal authentication, as well as to mitigate problems with forcing users to remember many passwords. Though more analysis, extensive experiments and trial runs are needed, our method has performed well in the presence of minor simulated scanner distortions and other artifacts likely to be encountered in practice.

5. REFERENCES

[1] Mikhail J. Atallah, Keith B. Frikken, Michael T. Goodrich, and Roberto Tamassia. Secure biometric authentication for weak computational devices. In *Proc. of Financial Cryptography and Data Security Conference (FC '05)*, Roseau, The Commonwealth of Dominica, February 2005.

[2] Claude Barral, Jean-Sébastien Coron, and David Naccache. Externalized fingerprint matching. Cryptology ePrint Archive, Report 2004/021, 2004. http://eprint.iacr.org/.

[3] Martin L. Brady. A fast discrete approximation algorithm for the Radon transform. *SIAM J. Comput.*, 27(1):107–119, 1998.

[4] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. Cryptology ePrint Archive, Report 2003/235, 2003. http://eprint.iacr.org/.

[5] Leon Ehrenpreis. *The Universality of the Radon Transform*. Oxford University Press, USA, 2003.

[6] Oded Goldreich. *Foundations of Cryptography*. Cambridge University Press, 2001.

[7] W.A. Gotz and H.J. Druckmuller. A fast digital Radon transform–an efficient means for evaluating the Hough transform. *PR*, 28:1985–1992, 1995.

[8] A. Jain, S. Dass, and K. Nandakumar. Soft biometric traits for personal recognition systems. In *Proc. of International Conference on Biometric Authentication*, Hong Kong, China, July 2004.

[9] A. Jain, A. Ross, and S. Prabhakar. Fingerprint matching using minutiae and texture features. In *Proc. of International Conference on Image Processing (ICIP)*, Thessaloniki, Greece, October 2001.

[10] Anil K. Jain and David Maltoni. *Handbook of Fingerprint Recognition*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2003.

[11] Michael Malkin and Ramarathnam Venkatesan. The randlet transform. In *Allerton Conference on Communication, Control and Computing*, Urbana-Champaign, IL, 2004.

[12] Vaclav Matyas and Zdenek Riha. Biometric authentication — security and usability. In *Proc. of IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security: Advanced Communications and Multimedia Security*, Portoroz, Slovenia, September 2002.

[13] Alfred J. Menezes, Scott A. Vanstone, and Paul C. Van Oorschot. *Handbook of Applied Cryptography*. CRC Press, Inc., Boca Raton, FL, USA, 1996.

[14] Mehmet Kivanc Mihcak and Ramarathnam Venkatesan. New iterative geometric methods for robust perceptual image hashing. In *DRM '01: Revised Papers from the ACM CCS-8 Workshop on Security and Privacy in Digital Rights Management*, pages 13–21, London, UK, 2002.

[15] Neurotechnologija, Inc. VeriFinger (http://www.neurotechnologija.com/verifinger.html). 2006.

[16] F. A. P. Petitcolas and M. G. Kuhn. StirMark software (available on the Web). 2003.

[17] Salil Prabhakar, Sharath Pankanti, and Anil K. Jain. Biometric recognition: Security and privacy concerns. *IEEE Security and Privacy*, 1(2):33–42, 2003.

[18] Bruce Schneier. *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. John Wiley & Sons, Inc., New York, NY, USA, 1993.

[19] Bruce Schneier. Password Safe (http://www.schneier.com/passsafe.html). 2006.

[20] Massimo Tistarelli, Josef Bigün, and Anil K. Jain, editors. *Biometric Authentication, International ECCV 2002 Workshop Copenhagen, Denmark, June 1, 2002, Proceedings*, volume 2359 of *Lecture Notes in Computer Science*. Springer, 2002.

[21] R. Venkatesan and M. H. Jakubowski. Image hashing. In *DIMACS Conf. on Intellectual Property Protection*, Piscataway, NJ (USA), April 2000.

[22] R. Venkatesan, S.-M. Koon, M. H. Jakubowski, and P. Moulin. Robust image hashing. In *Proc. of International Conference on Image Processing (ICIP)*, Vancouver, BC (CA), September 2000.