

Fingerprinting Blank Paper Using Commodity Scanners

William Clarkson^{*}, Tim Weyrich^{*,†}, Adam Finkelstein^{*}, Nadia Heninger^{*},
J. Alex Halderman[‡] and Edward W. Felten^{*}

^{*}*Department of Computer Science
Princeton University
{wclarkso, af, nadiah, felten}
@cs.princeton.edu*

[†]*Department of Computer Science
University College London
t.weyrich@ucl.ac.uk*

[‡]*Department of Electrical Engineering
and Computer Science
The University of Michigan
jhalderm@eecs.umich.edu*

Abstract

This paper presents a novel technique for authenticating physical documents based on random, naturally occurring imperfections in paper texture. We introduce a new method for measuring the three-dimensional surface of a page using only a commodity scanner and without modifying the document in any way. From this physical feature, we generate a concise fingerprint that uniquely identifies the document. Our technique is secure against counterfeiting and robust to harsh handling; it can be used even before any content is printed on a page. It has a wide range of applications, including detecting forged currency and tickets, authenticating passports, and halting counterfeit goods. Document identification could also be applied maliciously to de-anonymize printed surveys and to compromise the secrecy of paper ballots.

1. Introduction and Roadmap

Viewed up close, the surface of a sheet of paper is not perfectly flat but is a tangled mat of wood fibers with a rich three-dimensional texture that is highly random and difficult to reproduce. In this study, we introduce a new method for identifying a physical document—and verifying its authenticity—by measuring this unique natural structure. We show that paper texture can be estimated using only a flatbed scanner coupled with appropriate software, and that this feature is robust against rough treatment—such as printing or scribbling on the document or soaking it in water—and adversarial counterfeiting. Under normal conditions, our technique can identify documents with near-perfect accuracy and a negligible false positive rate.

It has long been known how to authenticate the *content* printed on a page by using cryptographic methods such as digital signatures. We address a different problem: how to authenticate the paper itself. For some kinds of documents, such as currency and tickets, it matters not only that the content is unaltered but also that the document is a genuine original rather than a copy or forgery. Physical document authentication has many applications, which we discuss

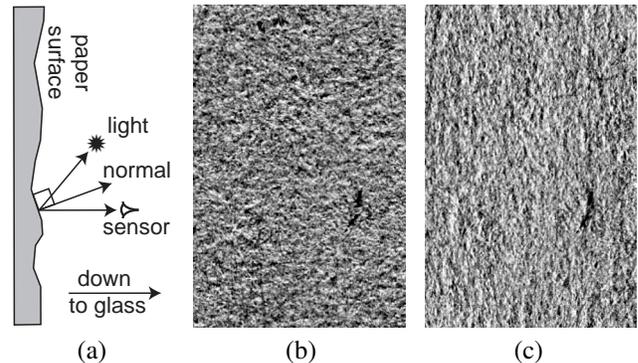


Figure 1. Since the surface of a sheet of paper is not perfectly flat, a scanner will produce a different image depending on the orientation of the page. The light reaching the sensor depends on the relative angles of the light source and surface normal, (a). A 10 mm tall region of a document scanned from top to bottom, (b), appears different from the same region scanned from left to right, (c). By combining (b) and (c) we can estimate the 3-D texture.

in Sections 7 and 8. Some of these applications may be harmful; for example, our method allows re-identification of supposedly anonymous surveys and paper ballots.

In contrast with previous efforts, our technique measures paper’s 3-D texture, requires no exotic equipment, produces a concise document fingerprint, does not require modifying the document, and may be applied to blank paper before content is printed. Previous systems lack one or more of these properties. For example, Laser Surface Authentication [1] requires a costly laser microscope to image paper texture, while the technique proposed by Zhu et al. [2], which focuses on ink splatter caused by randomness in the printing process, requires the paper to be printed with known content prior to fingerprinting. We discuss these and other related work in Section 2.

The physical document authentication technique we propose is a three-stage process culminating in a robust and secure fingerprint. In the first stage, we scan the original

document and estimate its surface texture. Scanners normally measure only the color of a document, but by scanning the paper several times at different orientations, we can estimate the shape of the surface (see Figure 1). In the second stage, we condense the surface texture into a concise feature vector, which robustly identifies the page. The third and final stage uses a secure sketch to generate a fingerprint that reveals little information about the feature vector. The fingerprint can be printed on the document (e.g., as a 2-D bar code) or stored in a database. The verification procedure is similar to the fingerprinting process, with a different final stage that verifies that the generated feature vector is correct. We describe each of these stages in detail in Section 3.

We designed our technique to satisfy several security and usability goals:

- **Uniqueness** Every document should be identifiable and distinguishable from all others.
- **Consistency** A fingerprint should be verifiable by multiple parties over the lifetime of the document.
- **Conciseness** Document fingerprints should be short and easily computable.
- **Robustness** It should be possible to verify a fingerprinted document even if it has been subjected to harsh treatment.
- **Resistance to Forgery** It should be very difficult or costly for an adversary to forge a document by coercing a second document to express the same fingerprint as an original.

Sections 4–6 evaluate our system in terms of these goals.

The most recent version of this paper can be found on our web site, <http://citp.princeton.edu/paper/>.

2. Related Work

The Fiberfingerprint system of Metois et al. first introduced the notion of using surface texture to uniquely identify a document [3]. Employing a custom device, Fiberfingerprint measures “inhomogeneities in the substrate” of a document, from which a unique identifier is derived. The system employs alignment marks that are added to the document in order to orient the verification system, and requires a specialized hardware device rather than a commodity scanner.

Laser Surface Authentication is a technique that measures the texture of a page using a high-powered laser microscope [1]. Creating and verifying fingerprints in their system requires access to this specialized device, which may be prohibitively expensive for many users. Their system also requires that the verifier be online, which may rule out applications such as third-party ticket verification.

A recent patent application by Cowburn and Buchanan describes using a commodity scanner to identify documents [4]. This method does not measure the normal vector field of a document, but rather uses scans from multiple orientations

in order to extract other additional information. The feature vector used by Cowburn and Buchanan is not concise, and their fingerprint is not secure. An adversary with access to the fingerprint is able to easily discover the surface texture of the document, possibly making forgery less difficult.

Zhu et al. focus on identifying “non-repeatable randomness existing in the printing process” [2]. They generate a fingerprint from the random ink splatter that occurs around the edges of any features printed on a page. Unlike our scheme, their method can only be applied after a document has been printed. Furthermore, their implementation requires modifying the original document by printing a known target pattern.

Our method is an improvement over previous work because we measure the surface texture of a document without the requirement of expensive equipment. We utilize the unique fiber structure as identified and relied upon by Metois et al., Cowburn and Buchanan, and Zhu et al. but do so without modifying the document in any way. Our method allows documents to be fingerprinted before or after content is printed. In fact, fingerprinting and tracking using our system can begin during the paper manufacturing process. We have also developed methods for hiding the target feature vector through the use of a secure sketch. This means a potential counterfeiter cannot learn what features he needs to reproduce from the fingerprint alone but would need access to the original document to even attempt a forgery.

3. Fingerprinting Process

Our fingerprinting process allows for registration and validation of a sheet of paper without the participation of a central registration authority. Depending on the application, a document’s fingerprint can be stored in a database for future verification, or it can be printed on the document along with a digital signature, creating a self-verifying original. The fingerprint can be used to ascertain whether two documents share the same feature vector without revealing the registered feature vector itself.

The registration and validation processes are quite similar, as shown in Figure 2. In the registration process, we scan a document, estimate its three-dimensional surface texture, and generate a feature vector \mathbf{V} that represents the document’s unique texture. We consider two documents to be the same if they have similar feature vectors. To protect the feature vector and inhibit forgeries that might seek to reproduce an exact feature vector, the fingerprint contains only a one-way hash $H(\mathbf{V})$ of the extracted feature vector. To achieve robustness against measurement errors in the feature vector, the registration process derives error-correction information from \mathbf{V} and stores it in the fingerprint in the form of a secure sketch. The fingerprint also contains a random seed to initialize the pseudorandom number generator used to compute the feature vector, as described in Section 3.2.

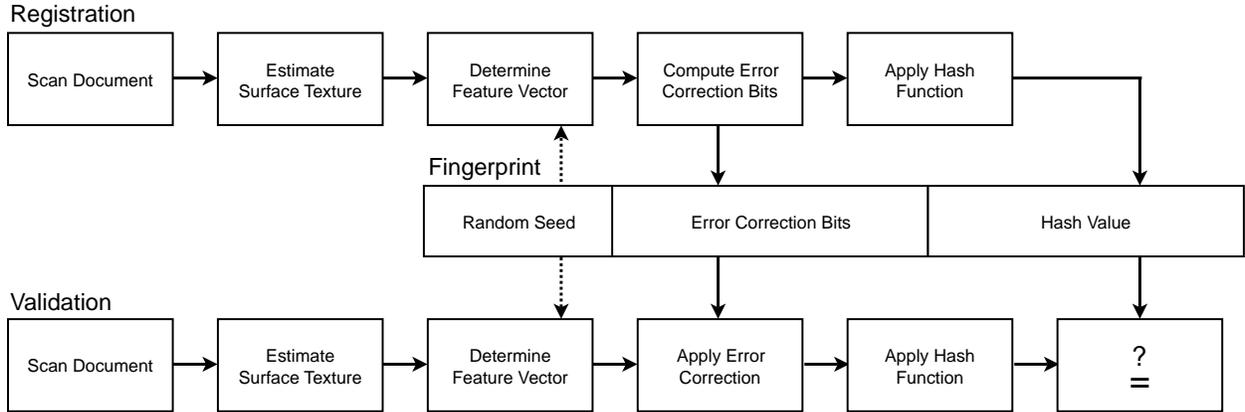


Figure 2. Registration and validation pipelines. *Registration*: Our method creates a fingerprint that consists of a hash value, error correction information, and a random seed. *Validation*: A document is authenticated if a newly computed hash value is identical to the one stored in the fingerprint. The stored error correction information is used to correct potentially faulty bits in the feature vector.

The validation process has no access to the original feature vector. Validating a document requires determining a document’s feature vector anew, using the seed stored in the fingerprint. Validation assumes a potentially flawed raw feature vector $\tilde{\mathbf{V}}'$ and uses the secure sketch to obtain an error corrected $\tilde{\mathbf{V}}$, as described in Section 3.3. The candidate document is considered valid if this feature vector maps to the same hash value stored in the fingerprint—that is, if $H(\tilde{\mathbf{V}}) = H(\mathbf{V})$. The remainder of this section discusses the registration and validation pipeline in detail.

3.1. Estimating document surface texture

To capture the surface texture of a document, we scan it at four orientations: 0° , 90° , 180° , and 270° . This allows recovery of the surface orientation for every sampled surface point. Our procedure assumes that paper is perfectly diffuse, which is an assumption that largely holds for near-orthogonal observation. Diffuse materials reflect a portion of the incident light that is proportional to the cosine of the angle between the direction of incidence and the surface normal—that is, proportional to the dot product of these normalized directions. This property is commonly exploited in *photometric stereo* to reconstruct surface normals from multiple images under varying illumination by a point light source [5]. Similarly, we apply photometric stereo to the four captured scans. Flatbed scanners, however, contain a linear light source, rather than a point source, which disallows the application of traditional photometric stereo. Brown et al. recently demonstrated how normals can be derived from flatbed scans under multiple orientations [6]. Their method, however, relies on an extensive calibration procedure, which would make it impractical for authentication purposes. Instead, we will



Figure 3. Difference image between two 1200 DPI scans showing the surface texture measured by the scanner in the y direction. Actual size: “sum”.

derive a novel photometric stereo solution for flatbed scanners, which provides us with information on surface orientation without the need for dedicated calibration.

Let us define a coordinate system for the paper and the scanner so that the paper lies in the xy -plane, the z -axis points away from the flatbed of the scanner, and the scanner’s linear light source is parallel to the x -axis. We approximate this light source by a line segment extending from x_1 to x_2 . We further assume that the light source is offset with respect to the line on the paper imaged by the CCD sensor (see Figure 1(a)) by o_y in the y -direction and by o_z in the z -direction.

Each point on the paper has a normal \mathbf{n} and a diffuse color, or *albedo*, ρ . Without loss of generality, we concentrate on a surface point at the origin of our coordinate system. The observed intensity of such a surface point is then:

$$I = \rho \int_{x_1}^{x_2} \left\langle \mathbf{n}, \frac{(x, o_y, o_z)^\top}{\|(x, o_y, o_z)^\top\|^3} \right\rangle dx, \quad (1)$$

which is the integral over all light diffusely reflected off that surface point and originating from points $(x, o_y, o_z)^\top$ along

the linear light source. As every flatbed scanner is designed for even illumination, any limiting effects near ends of the light source are negligible and we shall ignore the integral limits in the remainder of this discussion.

Scanning the same surface point a second time with the paper rotated by 180° displaces the light source from o_y to $-o_y$. Subtracting the resulting two scans I_{0° and I_{180° from each other leads to:

$$\begin{aligned}
 d_y &= I_{0^\circ} - I_{180^\circ} \\
 &= \rho \int \left\langle \mathbf{n}, \frac{(x, o_y, o_z)^\top}{\|(x, o_y, o_z)^\top\|^3} - \frac{(x, -o_y, o_z)^\top}{\|(x, -o_y, o_z)^\top\|^3} \right\rangle dx \\
 &= \rho \int \left\langle \mathbf{n}, \frac{(0, 2o_y, 0)^\top}{\|(x, o_y, o_z)^\top\|^3} \right\rangle dx \\
 &= n_y \rho \int \frac{2o_y}{\|(x, o_y, o_z)^\top\|^3} dx \\
 &= n_y \rho s. \tag{2}
 \end{aligned}$$

That is, the difference d_y directly yields the y component n_y of the surface normal \mathbf{n} , multiplied by the albedo ρ and a fixed constant s that is dependent on the scanner geometry only. Analogously, $d_x = I_{270^\circ} - I_{90^\circ} = n_x \rho s$. With four scans we can determine each surface normal's projection into to xy -plane, $\mathbf{n}_2 = (n_x, n_y)$, up to a scale. The factor s is assumed to be fairly constant across the page, and the remaining scale is given by the local surface reflectance ρ of the paper at any given location.

Application of equation (2) requires precise alignment of each surface point across all scans. To reduce the effect of alignment imprecision and to isolate frequencies of the document that are stable across scans and different scanners, we apply a low-pass filter to the document and down-sample it. In our experiments we scanned each document at 1200 SPI (samples per inch) and down-sampled it by a factor of eight, resulting in an effective resolution of 150 SPI.

After processing the four scans of a document, we recover the surface texture as a two-dimensional vector field with $\mathbf{d} = (d_x, d_y)^\top = \rho s \mathbf{n}_2$ defined at each location of the document.

3.2. Computing the feature vector

From this vector field \mathbf{d} we determine the feature vector of the document. A good feature vector captures unique characteristics of a document, while at the same time being concise. We model the feature vector \mathbf{V} as an N -bit vector of independent bits f_i whose values are derived from the surface normals of the document.

In contrast to previous approaches, we do not extract a feature vector from a single region of the document, but we compute the feature vector from a collection of representative subsections, *patches*, of the document. For documents down-sampled to 150 SPI, we choose square patches of 8×8 samples, centered at a series of random locations \mathbf{p}_i . For even

spacing we draw these locations from a Voronoi distribution [7]: we use the random seed stored in the fingerprint to initialize P pseudorandom start locations on the page and use Lloyd's Voronoi relaxation to obtain a set of locations distributed evenly across the document, as shown in Figure 4.

In principle one could now directly compare the patches of a document A to corresponding patches in a document B in order to verify two documents as being the same. The disadvantages are that this requires access to the patches of B when verifying A , which would require an amount of storage prohibitive for offline applications, and, more importantly, that it would reveal the original document's structure to a forger. Hence, we derive a compressed feature vector and store its hash along with a secure sketch to hide the feature vector from an adversary.

Each patch contains 64 2-D samples \mathbf{d}_i , $i = 1, \dots, 64$, which we stack to create a patch vector $\mathbf{p} \in \mathbb{R}^{128}$. Each patch contributes T bits to the feature vector. We compute these feature bits f_i , $i = 1, \dots, T$, by subsequently comparing the patch vector to T *template vectors* \mathbf{t}_i . The template vectors are a set of pseudorandomly chosen orthonormal vectors in \mathbb{R}^{128} generated using the same seed that is used to determine patch locations: the \mathbf{t}_i are initialized with vector components drawn from a $N(0, 1)$ distribution, followed by Gram-Schmidt orthonormalization. Each template vector can be interpreted as a template patch of 8×8 2-vectors denoting surface orientation.

The comparison is performed by correlating the patch vector \mathbf{p} and each template vector \mathbf{t}_i ; i.e., by computing the dot product $\langle \mathbf{p}, \mathbf{t}_i \rangle$. Positive correlation means that surface orientations in the patch and the template patch

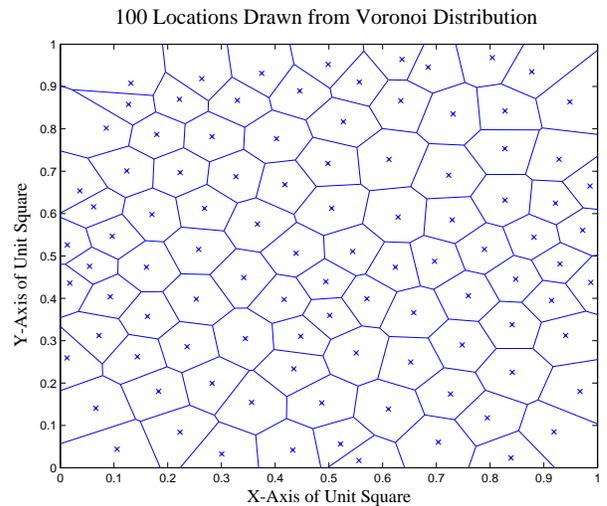


Figure 4. Sample Voronoi distribution of 100 points in the unit square. Voronoi distributions give relatively uniform coverage of a region, while simultaneously ensuring no overlap of patches.

agree; negative correlation denotes mostly opposing surface orientations. The respective feature bit is determined by the sign of the correlation:

$$f_i = \frac{1 + \text{sign}(\langle \mathbf{p}, \mathbf{t}_i \rangle)}{2}. \quad (3)$$

See Algorithm 1 for further illustration.

```

V = bool[PT]
Retrieve surface orientation vectors of document
Extract  $P$  patches based on Voronoi distribution
for  $p = 1$  to  $P$  do
  template = generate new set of  $T$  pseudo-
             random orthonormal template vectors
  for  $i = 1$  to  $T$  do
     $c = \langle \text{patch}[p], \text{template}[i] \rangle$ 
     $f_{(p-1)P+i} = \text{TRUE}$  if  $c > 0$ 
  end for
end for

```

Algorithm 1: Feature vector generation.

The number of independent bits that can be extracted from a patch in this way is limited and depends on the amount of information contained in a patch. A standard tool to characterize this amount of information is principal component analysis (PCA) [8]. We performed PCA on a large set of randomly chosen patches from different documents. The results show that for 8×8 -patches 75% of the information can be expressed with only 32 principal components; that is, within a 32-dimensional subspace. We hence decided to restrict ourselves to $T = 32$ of 128 possible orthonormal template vectors, as additional template vectors are likely to produce increasingly correlated feature bits. We further choose 100 patches, $P = 100$, leading to a feature vector of 3,200 bits for each document.

3.3. Creating the document fingerprint

From the feature vector we can create a document fingerprint that can be used to authenticate the document without revealing information about the document. The fingerprinting method should be both concise and robust to errors. This situation is similar to that of biometrics, where a user provides a value $\tilde{\mathbf{V}}$ which is close to, but not identical to the registered value \mathbf{V} (e.g., the Hamming distance is relatively small). Additionally, providing an adversary with the full feature vector may not be desirable, as it provides a blueprint for potential forgery.

A document fingerprint consists of a hash of the feature vector $H(\mathbf{V})$, where H is a collision-resistant cryptographic hash function, along with a secure sketch $ss(\mathbf{V})$ following the ideas of Dodis et al. [9] and Juels and Wattenberg [10]. The secure sketch allows the system to correct any errors that may occur in the candidate $\tilde{\mathbf{V}}$, assuming $\tilde{\mathbf{V}}$ is close enough

to \mathbf{V} , without revealing \mathbf{V} to an adversary who does not have any information about \mathbf{V} .

Suppose the registered value for a document is an N -bit value \mathbf{V} , and we wish to accept any $\tilde{\mathbf{V}}$ within Hamming distance δN of \mathbf{V} . The secure sketch proposed by Juels and Wattenberg chooses a random codeword x from an error-correcting code of length N that can correct δN errors, and stores $ss(\mathbf{V}) = \mathbf{V} \oplus x$. To recover \mathbf{V} from a candidate $\tilde{\mathbf{V}}$, the system calculates $\hat{x} = ss(\mathbf{V}) \oplus \tilde{\mathbf{V}}$, corrects \hat{x} to the nearest codeword, and verifies that $H(\mathbf{V}) = H(x \oplus ss(\mathbf{V}))$. If \mathbf{V} and $\tilde{\mathbf{V}}$ have Hamming distance less than δN , the system correctly outputs \mathbf{V} .

Dodis et al. [9] show that the number of bits of information about the fingerprint revealed by the secure sketch is $N - k$, where $k = \log K$ is the dimension of the error-correcting code used in the secure sketch when it has K codewords. Thus, in order to maximize the security of the system for a fixed N , the error-correcting code should have as high a dimension k as possible.

Low-Density Parity Check (LDPC) codes, along with turbo codes, are among the strongest error-correcting codes in use, thanks to efficient decoding algorithms developed in the last two decades. In our implementation, we used the LDPC library written by Neal [11]. LDPC codes are well-suited to this application because they work well on large block sizes and in practice can often correctly decode beyond the minimum distance of the code [12]. In addition, the LDPC decoding algorithm can take into account a confidence level specified for each individual bit to further improve the performance of the code. In our case, this confidence level can be calculated from the magnitude of the dot product of

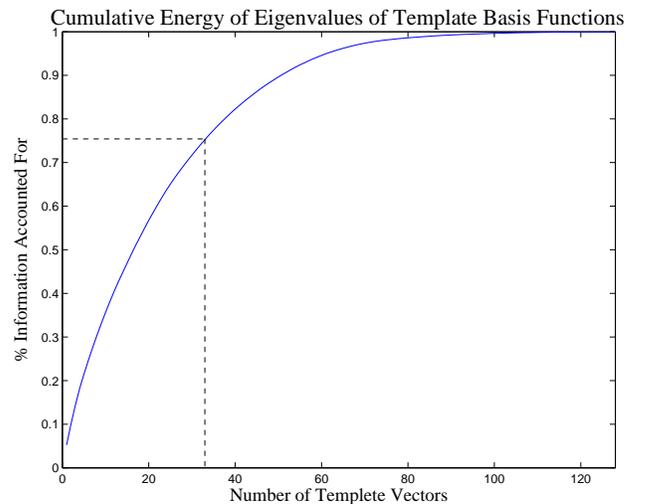


Figure 5. Principal component analysis of a large number of 8×8 -patches shows that 75% of the information has been extracted after 32 components.

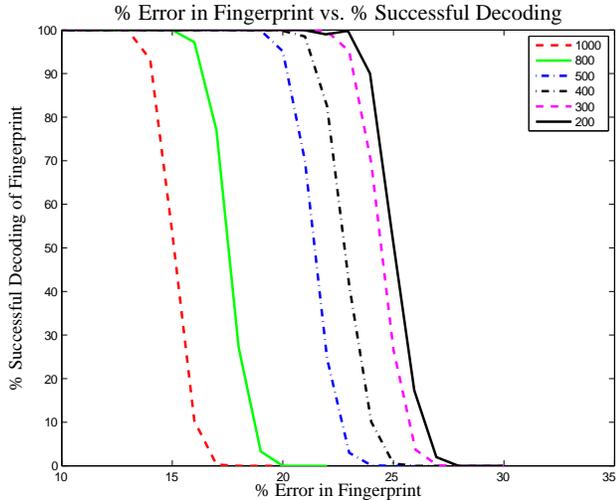


Figure 6. Fraction of fingerprints successfully decoded for varying fingerprint error rates using LDPC codes of different dimensions k .

the template vector with the patch vector. The correspondence between the two is graphed in Figure 7.

The length of our feature vector is $N = 3200$ bits. We experimented with codes of suitable dimension to correct bit error rates between $\delta = 10\%$, allowing correct identification of all types of paper we experimented with under ideal conditions (see Figure 8), and $\delta = 30\%$, suitable to identify documents under less-ideal conditions such as soaking and scribbling (see Figure 9). For the case of decoding under ideal conditions, a code of dimension $k = 1000$ and $N = 3200$ is sufficient to correctly verify all test documents, with no false positives. For the case of decoding under less ideal conditions, a code of dimension $k = 300$ and $N = 3200$ sufficed to correctly verify 95% of all test documents, with no false positives. See Figure 6 for a summary of these results.

The feature vector length can be adjusted to suit the needs of the application (and the expected document treatment conditions) by increasing or reducing the number of patches. Longer feature vectors provide a higher level of accuracy when distinguishing two documents, especially under harsh treatment, but require increased storage. We chose $N = 3200$ bits as our feature vector length to ensure that it would fit in a 2-D barcode.

4. Robustness

Section 3 describes a process for registration and validation of a document fingerprint. In this section we evaluate document fingerprints across different types of paper, including normal copy paper (Boise Aspen 50), university letterhead with a visible watermark, and index cards. We also evaluate

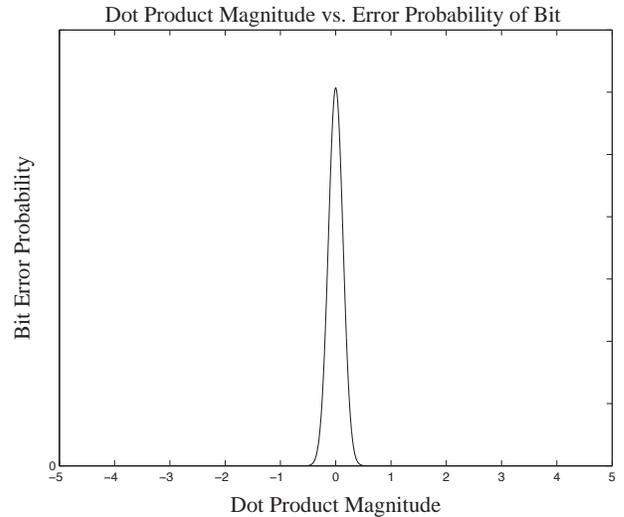


Figure 7. Correspondence between dot product magnitude and error probability during validation. (Fit to normal distribution with $\mu=0$ and $\sigma=0.1314$.)

the fragility of a document fingerprint under various treatment conditions. Our goal is to test whether our technique typically validates different observations of the same document (true positives) and rejects pairs of observations of different documents (true negatives), while rarely validating pairs of observations of different documents (false positives) or rejecting different observations of the same document (false negatives).

Our experiments show that a fingerprint can be found for a variety of different types of paper. Unless otherwise noted, each experiment began with a document scanned at 1200 DPI on an Epson Perfection v700 scanner. Each test focused on a 3×3 inch square in the center of the page.¹

For each test, we captured five observations of a set of five documents, for a total of 25 observations. Each observation consisted of four scans taken at 0° , 90° , 180° , and 270° that we used to estimate surface normals. We expect no two scans of the same document to be exactly alike due to slight variations in placement on the scanner, random noise, and other uncontrollable variables in the process.

The amount of error tolerated in a matching fingerprint can be adjusted by choosing an appropriate error-correcting code during the fingerprinting process described in Section 3. The number of bits that can be corrected by the code should be determined by the needs of the application, as it establishes

1. In our robustness experiments, we used a printed box on the test pages to identify the region to be fingerprinted and to align the different scans. However, alignment could be accomplished by other means, such as by relying on the boundaries of the page or other printed material, or by simply recording a few patches at high resolution [13]. Different sets of patches should be used for alignment and verification, because using the same patches could increase the false positive rate.

a tradeoff between the security of the system and the relative likelihood (or harm) of a false positive or false negative.

4.1. Ideal handling conditions

As a baseline test, we measured the frequency of correct and incorrect validation and rejection under ideal handling conditions, when we expect no document degradation.

We began with 25 observations (5 from each of 5 documents). We chose 40 random seeds and sampled a fingerprint from each observation for each seed, following the process described in Section 3. We made $\binom{25}{2} = 300$ comparisons for each seed, yielding a total of 12,000 comparisons.

For each comparison, we computed the Hamming distance between the two fingerprints. These distances are summarized for the 12,000 comparisons by the histogram shown in the top graph in Figure 8. Under non-adversarial conditions, document fingerprints of normal copy paper differ, on average, in only 99 (3.1%) of the 3200 bits. In contrast, as one would expect, the average Hamming distance for fingerprints made from observations of *different* documents is 50% of the bits. These distributions are well-separated; the maximum Hamming distance between feature vectors from the same document is 177, while the minimum distance between feature vectors of different documents is 1490. An error tolerance anywhere in this range should give no false positives and no false negatives for these tests. We found similar results for index cards and university letterhead; see Figure 8.

The distributions for the “same” fingerprint comparison tests and “different” fingerprint comparison tests seem to be reasonably approximated by a normal distribution. Fitting Gaussian curves to this data, we can find a summary statistic, Egan’s *sensitivity index* for Gaussian distributions of signal and noise with unequal variances, given by:

$$d_s = 2(\mu_2 - \mu_1)/(\sigma_1 + \sigma_2) \quad (4)$$

where μ_1 , μ_2 , σ_1 and σ_2 are the means and standard deviations of the distributions [14]. For this experiment, $d_s = 52.0$. To give some intuition about the significance of this statistic, the two Gaussians intersect at a Hamming distance of 731 bits; the heights of the curves are such that the chance of a single comparison resulting in either a false positive or false negative is 1 in 10^{148} . If we reduce the feature vector length from $N = 3200$ bits to 1600, 800, or 400 bits, the probability of such errors is 1 in 10^{96} , 1 in 10^{57} , or 1 in 10^{35} , respectively.

We repeated these experiments on different scanner models and found similar results. When comparing a document fingerprinted on one model and verified on another, results are slightly worse.²

2. We chose several of the parameters of our algorithm (e.g., the downsample factor and size of the patches) based on preliminary experiments using the Epson v700 scanner. The optimal settings for verification of documents using other scanner models may vary.

4.2. Non-ideal handling conditions

The previous experiments were performed under ideal handling conditions. We performed additional tests to ascertain the robustness of fingerprints when the document is subjected to less-than-ideal conditions. These tests included scribbling on the paper, printing on it with ink, and soaking the page in water.

Scribbling. We first scanned a set of five blank documents, then scanned them again after scribbling over them with a pen. In each document the scribble was unique, covering an average of 8% of the test region. In this test, 25 pre-scribble observations were compared against their 25 post-scribble counterparts, for a total of 625 pairs. We used 40 different fingerprint samples per document to yield a total of 25,000 comparisons. The Hamming distances resulting from these comparisons are plotted in the top graph in Figure 9. The sensitivity index in this case is lower ($d_s = 28.8$), although the curves remain quite well-separated. With a decision threshold of 1130 bit errors in the fingerprint, the chance of a false positive or false negative is 1 in 10^{47} .

Printing. In this experiment we printed single-spaced text in 12 pt. Times New Roman lettering over the test region, covering approximately 13% of the area with ink. The distributions shown in the middle graph of Figure 9 were obtained as in the scribble test. Even in this experiment, in which most patches used for the fingerprint were partially covered by ink, the sensitivity index is 26.1 and the chance of a false positive or false negative at the crossover is 1 in 10^{38} .

Wetting and drying. The bottom graph in Figure 9 shows the resiliency of document fingerprints after the document was submerged in water for five minutes. We dried each test document and ironed it until it was as flat as possible. Using the same evaluation protocol as for the scribble and printing tests, we found that documents could still be validated with 100% reliability, even with this fairly extreme mistreatment of the page ($d_s = 33.3$).

These experiments demonstrate that our fingerprinting method is robust when a document is handled under certain rough conditions. The ability to identify a document before and after it is printed on, scribbled on, or soaked in water has many potential applications.

5. Security

The security of our method relies on the inability of an attacker to reproduce the document’s surface, either because he does not know what features to produce or because he cannot recreate the normal vectors at the required scale. The threat model of each application is determined by several

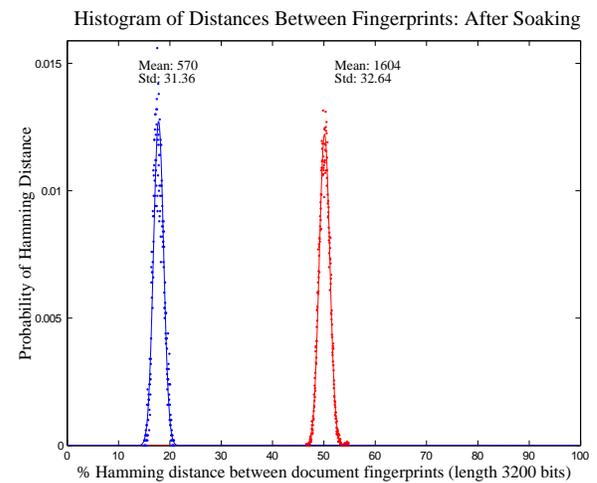
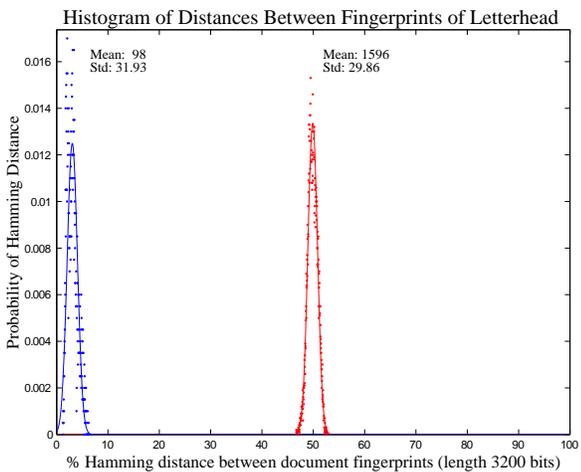
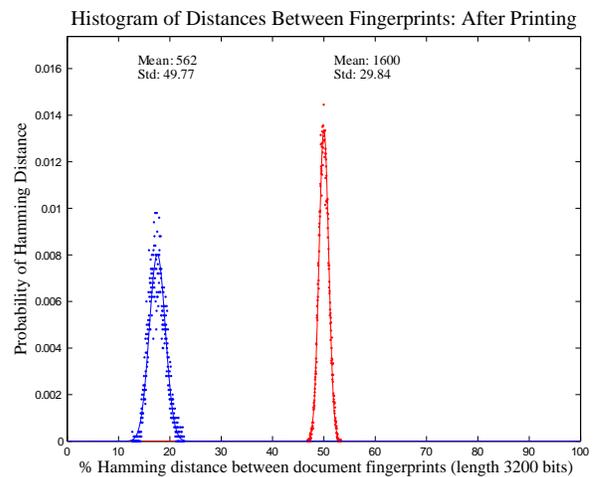
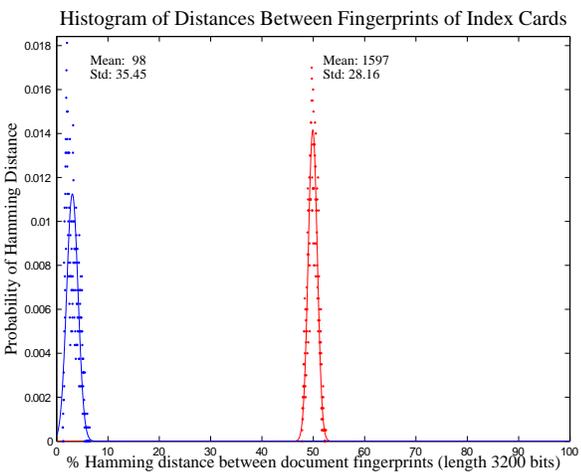
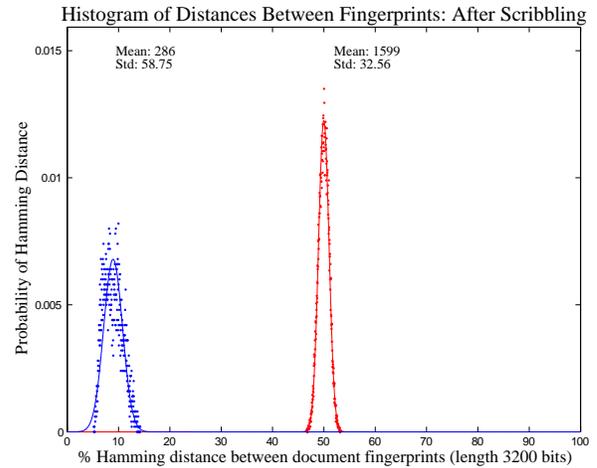
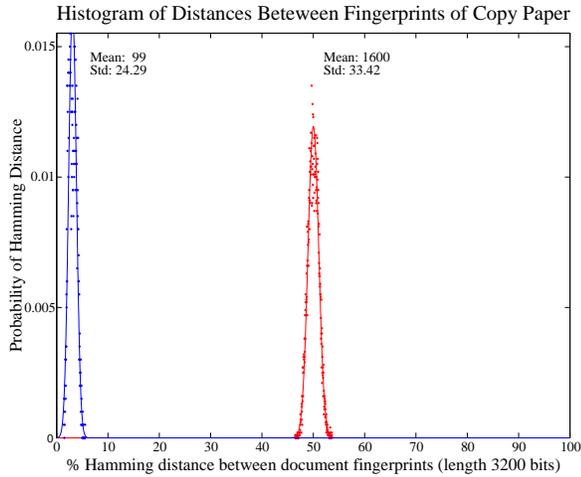


Figure 8. Distributions of Hamming distances between fingerprints for three paper types: copy paper (*top*), index cards (*middle*), and letterhead (*bottom*). In all graphs, the curve on the left depicts the distribution for scans of the same document, while the curve on the right gives the distribution for different documents.

Figure 9. Distributions of Hamming distances after subjecting documents to non-ideal treatments: scribbling (*top*), printing (*middle*), and soaking in water (*bottom*). The curves remain well separated even under these adverse conditions.

factors: the availability of an original to the attacker, whether verification is performed online or offline, and whether the verification device is trusted. Under the most common threat models, our method should prevent an attacker from forging a copy of an original.

Performing verification online or offline results in different considerations. Here “online” means that the verification device can communicate with a remote trusted server which can store data and perform computations; “offline” means that the verification device cannot communicate, although it can be preprogrammed with a limited amount of information such as a constant number of cryptographic keys. Online verification of a document has a straightforward solution, while offline verification requires security tradeoffs.

5.1. Online verification

Online verification need not reveal in advance the patch locations that will be analyzed. This forces an attacker to reproduce the entire surface of a document before presenting it for verification. In one approach, the verification server requests complete raw scans of the document at each of four orientations, which the server uses to perform the verification algorithm. Under this construction, the verification server does not reveal the chosen patches.

In an alternative approach, the verification server provides a fresh pseudorandom challenge to the client, and the client uses the challenge to seed a pseudorandom generator which is used to pick the patches and templates used in the verification algorithm. The client then computes the feature vector and sends it to the server. The server, having computed the same feature vector on its stored scans of the original document, verifies that the two feature vectors are similar enough.

In this threat model an attacker does not know *a priori* which patch locations on a document will be sampled. This forces an attacker to reproduce the surface texture of the document at each sample point in order to pass a counterfeit as an original.

5.2. Offline verification

The security of offline verification depends on whether the verification client is trusted and on the availability of an original to the attacker. In the offline case, we assume that the fingerprint of the legitimate original document is either pre-stored on the client or is printed onto the document (perhaps as a 2-D barcode) along with the authority’s digital signature of the fingerprint. In either case, the client device checks the document against a known fingerprint.

5.2.1. Offline: trusted device. Currency and ticket counterfeit detection at banks and concerts are two important examples of offline verification with a trusted device. By

“trusted” we mean that the device outputs a Boolean match/no-match result but does not leak any other information.

The secret information stored in the device could be the public key of the registration entity. The seed stored in the document fingerprint could be encrypted under the secret key of the registration entity. Therefore, knowledge of the fingerprint for a document does not reveal the patch locations. The hash of the feature vector could also be signed by the registration entity, preferably using a separate key. This allows only trusted devices to determine patch locations and verify the authenticity of a document. No access to the registration entity is required, provided that the device has knowledge of the decryption and verification keys of the registration entity.

In this threat model the adversary does not know which patches will be analyzed. This forces the attacker to recreate the surface normals across the entire document to ensure verification of the document.

5.2.2. Offline: untrusted device, no access to original.

In the next case, the verification device is offline and untrusted (i.e., it might leak everything it knows to the attacker) and the attacker has not seen the original document that he is trying to forge. In this case, the attacker cannot forge the document because he does not know anything useful about the normal field that he must create. At most, he knows the fingerprint (if it is stored in the device) but this does not help him because the fingerprint is a secure sketch.

5.2.3. Offline: untrusted device, access to original.

The final case is the most challenging one, where the verification device is offline and untrusted, and the attacker has access to an original document that he wants to copy. Because there are no secrets from the attacker—he sees the original document including anything printed on it, and he knows the full state of the device—the attacker knows exactly which patches will be used for verification and how the feature vector will be computed from those patches. The attacker’s only task is to make a document that will generate a feature vector close enough to the original. Whether the attacker can do this is the subject of the next section.

6. Forging a Document

Suppose an attacker has seen an original document and wants to create a second document that will pass verification as the original. The attacker will start with an arbitrary piece of paper which, by assumption, will have a very different feature vector from the original. He will then try to modify the target paper so that its feature vector is close to that of the original document. To do this, the attacker needs to make fine-grained modifications to the document’s surface normals. This might be done via lithography or photographic techniques, but these will be expensive and will probably require special equipment. The most effective, economical

way to control the surface, we believe, is to print on the document.

Equation (2) shows that the fingerprinted vector (d_x, d_y) contains additional factors s and ρ . The scanner-dependent factor s can be assumed to be fairly constant across the page and hence has no influence on the sign of the correlation results in the feature vector generation. The remaining scale is given by the local surface reflectance ρ of the paper at a given location, which should be stable across multiple scans. On empty paper it is nearly constant; in the presence of print, ρ is greatly attenuated, which lessens the influence of the printed portion onto the correlation result. The adversary can try to control bits of the feature vector by printing dark ink at selected points in order to reduce their influence in the correlation calculations. Besides reducing ρ , printing at a point tends to flatten the document surface, as shown in Figure 3.

An adversary who aims at forging a document might try to leverage these effects by printing a set of carefully placed dots, either to cause the surface texture of a candidate document to express the same fingerprint as an original, or to down-weight unfavorable contributions to the patch correlation. To do this the forger must overcome two hurdles: printing dots on the page at desirable locations and/or printing dots with favorable surface normal vectors. Dark ink on a document would directly affect reflectivity, while light ink might solely change the normal vectors at a specific location. We assume that the adversary uses commercially available equipment and is able to print dots in any color from black to white. He has less control over the exact shape of the dots, which varies by printing technology and type of paper.

We conducted experiments to characterize the ability of a forger to precisely control the effect of a printed pattern. We measured the effective resolution—the number of distinct printable dots—for a high-end office printer, a Xerox Phaser 8550, with a nominal resolution of 2400x2400 DPI. The effective resolution is limited by dot gain, which causes printed dots to become larger than intended (see Figure 10) due to factors such as the viscosity of the ink and the absorbency of the paper. The smallest dots that the test printer could produce on a normal piece of copy paper are 1/240 inch, or 20x20 samples when scanned at 4800 SPI. This limits the effective resolution to 240 DPI. On the other hand, the positional accuracy of the printer seems closer to the rated 2400 DPI. We conclude that a forger could use commodity printers to print dots with positional accuracy similar to what commodity scanners can measure but size much greater than the scanner’s pixels.

Because printed dots typically span more than one sample in a patch, printing a dot at a specific location affects the neighboring surface normal vectors in unpredictable and uncontrollable ways. Due to paper variations as well as limited precision in the placement and viscosity of ink, the forger does not have precise control over the normal vectors

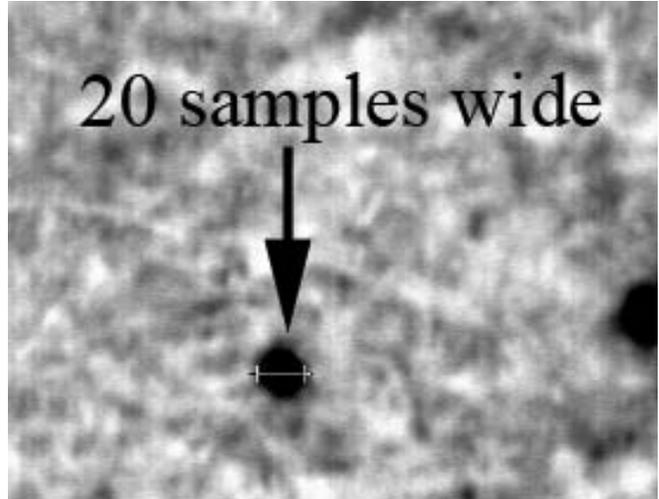


Figure 10. The smallest dots that can be produced by our test printer are 1/240 inch—20 samples wide in this 4800 SPI scan—despite the printer’s nominal 2400 DPI positional accuracy.

caused by a dot. We performed an experiment where we printed a series of black dots in a region of the document. We identified the black dots and measured the normal vectors in the surrounding region. For each printed dot, the desired normal vector of a location occurred on only one point of the surface.

The bottom-line question is how many degrees of freedom the adversary has in controllably modifying the normal vector field in a patch. Given the linear transformation used to determine each feature vector bit, the adversary will likely be able to achieve a desired set of feature vector values if he has enough degrees of freedom.

If there are N feature vector bits, and each bit is computed as the sign of the correlation of the normal field with a random vector field, then a truly random normal field value would match all N feature vector bits with probability 2^{-N} . However, it is likely that the feature vector bits are not fully independent. Although we have some evidence about the degree of independence (see, e.g., Figure 5), we do not have a precise estimate of how much entropy is encoded in the feature vector.

We are thus left with an open question: does the amount of information in a patch, as encoded in a feature vector, exceed the adversary’s ability to inject information into the patch? If we knew the answer to this question, we could state with some confidence whether an adversary could forge a document in the most favorable case (for the adversary), where the adversary sees the original document and the verification device is offline and untrusted. Unfortunately, we have to leave this question for future work.

7. Applications

There are a large number of applications that could benefit from the ability to uniquely identify a document. Many situations where physical possession of an item must be verified or authentication of an item is required could fruitfully employ our technique. Currency, ticket, and art counterfeit detection, as well as verification of product packaging are some of the applications where physical document authentication is desirable.

Counterfeit currency detection is one obvious application. The financial impact of counterfeit currency is large. Estimates of annual global revenue loss range from \$250- to \$500 billion [15], [16]. The ability to authenticate bills could change the way currency is produced and fraud is detected. Such a system would begin during currency production. The government would generate a fingerprint for each bill. This fingerprint could be stored in a database, along with the bill's serial number, or the government could digitally sign the fingerprint and print the fingerprint and signature on the bill. Any party wishing to verify a particular bill would scan the bill and verify that the fingerprint matched the one signed by the government. The authentication of a bill could be performed offline or online. Businesses and banks accepting large cash deposits could verify the currency was legitimate before completing the transaction. Offline authentication could be performed provided that the verification device had the public key of the currency issuer.

Ticket forgery at major concerts and sporting events is another large black-market business. Counterfeit event passes were widespread at the 2008 Beijing Olympics [17], and a British website recently sold more than \$2.5 million in fake tickets [18]. The ability for purchasers to verify the authenticity of tickets prior to purchase could greatly reduce the prevalence of online ticket fraud. Trust in ticket purchases on websites such as Stub Hub and eBay could be dramatically increased if the seller had to prove access to the item being auctioned or sold. Ticket clearing houses such as Ticketmaster could maintain an online database of fingerprints for all purchased tickets. Any party selling a ticket could scan and upload the ticket to Ticketmaster and receive verification of authenticity.

Forgery of artwork is a black-market business where the application of our technique may not be initially obvious. European police estimate that over half of the works in international markets are forgeries [19]. One family of art forgers was able to make \$2 million before they were caught. The ability of art forgers to reproduce the individual brush strokes of a work makes authenticating paintings increasingly difficult. In the best forgeries, art verifiers must sometimes rely on the chain of custody of the work in order to authenticate it [20]. However, we believe that it would be difficult to duplicate features of the *canvas* (down to the detailed arrangement of the weave) upon which the work is

painted. Thus art authenticity or forgery might be detectable by applying a technique like ours to the *canvas*, most probably on the back side of the painting.

Lottery tickets are similar to currency except that players need not be aware of a fingerprinting technique at all. In order for a lottery winner to collect on their winnings, the ticket must be verified by the lottery authority. The fingerprint of a winning ticket need not be printed on the document at all. Fingerprints of all possible winning lottery tickets can be privately maintained, and any claimants can be required to produce the actual winning ticket, with correctly verified fingerprint, in order to collect their winnings.

The accurate identification of paper based product packaging could benefit from this technique as well. When inspecting cargo, customs officials often inspect the contents of packages to weed out counterfeit goods. We can increase confidence in package contents by authenticating a product's packaging. If the packaging of a product is legitimate, then the contents of the package have a much higher likelihood of being authentic.

8. Privacy Implications

The feasibility of paper-based authentication demonstrates that some undesirable attacks are possible. Because our results do not modify the paper in any way, there is no way to detect, by inspecting a piece of paper, whether its fingerprint might have been recorded in advance by an adversary. This fact violates the traditional assumption that pieces of paper cannot easily be traced without the addition of distinguishing marks. Even unopened sheaves of blank printer paper might in principle have been fingerprinted at the factory. Applications such as paper-based voting, in which the secrecy of individual ballots is important, are challenged by our results.

For example, consider an optical-scan voting system in which voters fill out paper ballots. In such a system, the secrecy of ballots relies on the assumption that individual paper ballots are indistinguishable. Our work shows that this assumption may not be valid.

A corrupt official could scan the blank ballots in advance and record a unique fingerprint for each ballot in the stack. If ballots are given out to voters in a predictable order (e.g., from the top of the stack down) and the order of voters is recorded, as it is in many polling places, or observable by the attacker, then ballots can be re-identified after the election. Even worse, because pre-scanning leaves no evidence on the ballots themselves, a mere rumor that someone might have scanned the ballots in advance would be very difficult to disprove, and such a rumor would make coercion and vote-buying more credible.

More generally, the ability to re-identify ordinary sheets of paper casts doubt on any purportedly private information gathering process that relies on paper forms. "Anonymous" surveys or reporting systems may not in fact be anonymous.

Though it has long been possible to track sheets of paper using subtle chemical markers or “invisible ink,” these methods require some level of special expertise, and the presence of markers leaves evidence of the attack. Our research shows that an attacker armed with only ordinary equipment—a commodity scanner—is able to re-identify paper reliably without leaving any telltale marks.

9. Conclusion and Future Work

Our work shows that ordinary pieces of paper can be fingerprinted and later identified using commodity desktop scanners. The technique we developed functions like a “biometric” for paper and allows original documents to be securely and reliably distinguished from copies or forgeries.

At least two questions remain to be answered in future work. First, in the threat model where the adversary has access to the original document and the fingerprint, we do not know for certain that a clever adversary cannot forge a copy of the document with a high-resolution printer. Our initial work could not determine conclusively whether an adversary who can use a good printer will have enough degrees of freedom in modifying a document to make the document match a known fingerprint. Second, while we conjecture that our method can be applied to other materials such as fabric, more testing is needed to verify this, and special methods might be needed for some materials. We leave both of these questions for future work.

Our results are a tribute to the resolution of today’s scanners. Future scanners will capture ever more detailed images of paper documents, eventually allowing individual wood fibers to be imaged clearly. The security of our methods against forgery, in cases where the adversary has full access to information, will depend ultimately on a race between the resolution of the printers that can create patterns on a page, and the resolution of the scanners that can observe patterns.

10. Acknowledgments

We thank Andrew Appel, Victoria Hill, Andrew Moore, N. J. A. Sloane, Joshua R. Smith, and the anonymous reviewers for their invaluable suggestions and assistance.

References

- [1] J. D. R. Buchanan, R. P. Cowburn, A.-V. Jausovec, D. Petit, P. Seem, G. Xiong, D. Atkinson, K. Fenton, D. A. Allwood, and M. T. Bryan, “Forgery: ‘fingerprinting’ documents and packaging,” *Nature*, vol. 436, p. 475, 2005.
- [2] B. Zhu, J. Wu, and M. S. Kankanhalli, “Print signatures for document authentication,” in *Proc. 10th ACM Conference on Computer and Communications Security*, 2003, pp. 145–154.
- [3] E. Metois, P. Yarin, N. Salzman, and J. R. Smith, “FiberFingerprint identification,” in *Proc. 3rd Workshop on Automatic Identification*, 2002, pp. 147–154.
- [4] R. P. Cowburn and J. D. R. Buchanan, “Verification of authenticity,” US patent application 2007/0028093, Jul. 2006.
- [5] R. Woodham, “Photometric stereo: A reflectance map technique for determining surface orientation from image intensity,” in *Proc. 22nd SPIE Annual Technical Symposium*, vol. 155, 1978, pp. 136–143.
- [6] B. Brown, C. Toler-Franklin, D. Nehab, M. Burns, A. Vlachopoulos, C. Doumas, D. Dobkin, S. Rusinkiewicz, and T. Weyrich, “A system for high-volume acquisition and matching of fresco fragments: Reassembling Theran wall paintings,” *ACM Trans. Graphics (Proc. SIGGRAPH 2008)*, p. 84 (9 pp.), Aug. 2008.
- [7] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu, *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. Wiley, 2000.
- [8] J. E. Jackson, *A User’s Guide to Principal Component Analysis*. Wiley-Interscience, 2003.
- [9] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith, “Fuzzy extractors: How to generate strong keys from biometrics and other noisy data,” *SIAM Journal on Computing*, vol. 38, no. 1, pp. 97–137, 2008.
- [10] A. Juels and M. Wattenberg, “A fuzzy commitment scheme,” in *Proc. 6th ACM Conference on Computer and Communications Security*, 1999, pp. 28–36.
- [11] R. M. Neal. (2006, Feb.) Software for low density parity check codes. [Online]. Available: <http://www.cs.utoronto.ca/~radford/ldpc.software.html>
- [12] D. MacKay and R. Neal, “Near shannon limit performance of low density parity check codes,” *Electronics Letters*, vol. 33, no. 6, pp. 457–458, Mar. 1997.
- [13] C. Sorzano, P. Thevenaz, and M. Unser, “Elastic registration of biological images using vector-spline regularization,” *IEEE Trans. Biomedical Engineering*, vol. 52, no. 4, pp. 652–663, Apr. 2005.
- [14] D. McNicol, *A Primer on Signal Detection Theory*. Lawrence Erlbaum Assoc., 2004.
- [15] L. S. Amine and P. Magnusson, “Cost-benefit models of stakeholders in the global counterfeiting industry and marketing response strategies,” *Multinational Business Review*, vol. 15, no. 2, pp. 1–23, 2007.
- [16] U.S. Department of Commerce. Top 10 ways to protect yourself from counterfeiting and piracy. [Online]. Available: http://www.stopfakes.gov/pdf/Consumer_Tips.pdf
- [17] C. Balmer and K. Wills, “Beijing games hit by internet ticket scam,” *Reuters*, Aug. 4, 2008.
- [18] “Ticket site closed on fraud fears,” *BBC News*, Oct. 21, 2008.

- [19] J. L. Shreeve, “Art forgers: What lies beneath,” *The Independent*, Sep. 3, 2008.
- [20] R. D. Spencer, *The Expert versus the Object: Judging Fakes and False Attributions in the Visual Arts*. Oxford University Press, 2004.
- [21] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, “Information hiding—a survey,” *Proc. IEEE*, vol. 87, no. 7, pp. 1062–1078, Jul. 1999.

Appendix

Section 3 introduces a process for fingerprinting and verifying the fingerprint of a document. In this appendix we briefly outline some alternative strategies that might be desirable under different criteria for robustness or different levels of concern about forgery.

Using albedo versus normals. Because the high-resolution paper scans shown in figures throughout this paper reveal obvious color variation in addition to surface texture, perhaps a more straightforward approach would be to use the albedo (color) of the page as the basis for a fingerprint, rather than, or in addition to, the shape. Indeed, our initial implementations explored this approach, using a single scan (which combines albedo and normal information) to construct the fingerprint of a document. This approach is simpler and offers the substantial benefit that the document can be fingerprinted or verified more quickly, through a single scan.

The intensities of most of the pixels in a scanned page are modeled well by a truncated normal distribution, centered around the “white” color. To use this data as the basis for a fingerprint, we simply construct the vector \mathbf{p} as the concatenation of these intensities from a given patch. For example, an 8×8 patch would yield a vector $\mathbf{p} \in \mathbb{R}^{64}$. The fingerprint is then extracted from a collection of patches as described in Algorithm 1.

We did not pursue this approach because we believe this form of fingerprint may not resist forgers who use very light ink to print a desired pattern on the page. Another drawback is that any black ink on the page, which lies well outside the roughly-normal distribution of intensities found in blank paper, contributes to a very strong negative value in \mathbf{p} , introducing a bias in the dot products for the patch. Thus, any value outside the range of the truncated normal distribution must be zeroed out before constructing the fingerprint. This provides another opportunity for a forger to deliberately zero out regions of the patch with the goal of flipping bits towards a desired fingerprint. These attacks might be difficult to carry out in practice, since they require excellent registration in the printing process. Therefore, albedo-based fingerprints may be suitable for applications where some added risk of forgery is an acceptable tradeoff for increased speed and simplicity.

Patch-pair comparisons. Recall from Algorithm 1 that the vector \mathbf{p} contributes K bits to the overall fingerprint by taking the signs of the dot product of \mathbf{p} and a series of ortho-normal template vectors. We have also considered (and implemented) an alternate version of the algorithm where the bits of the fingerprint are taken to be the signs of the dot products of pairs of patches \mathbf{p} and \mathbf{q} . The naïve version divides the pool of patch positions into pairs and computes one bit of the feature vector from each pair. Unfortunately that approach allows an attacker to tweak each pair in turn independently. A more robust version considers bits from *all* patch pairs (p, q) where $p \neq q$. For example, for 64 patches each patch would participate in 63 bits, and this scheme could generate $\binom{64}{2} = 2016$ total bits.

In the case where a forger has a copy of an original document and therefore knows the fingerprint he is trying to reproduce (Section 5.2.3), this formulation has the advantage that the bits of the fingerprint are more tightly bound than those of the template vectors. Any attack on a single bit—for example, printing on a patch—is likely to impinge on the other (62) bits affected by that patch. Thus, a forger would have to solve an optimization problem to figure out how best to perform the attack.

However, the bits of the fingerprint generated from all patch pairs seem to be less independent than the bits generated by the template vectors. Preliminary experiments similar to those described in Section 4 indicate that “all-pairs” bits are *mostly* independent, but not as independent as the “template” bits. Since the arguments in Section 5.2.2 for security against “blind” attackers rely on bit independence, we generally prefer the “template” scheme.

Short fingerprints with no error-correcting information.

Section 3.2 describes a process for generating fingerprints composed of a hash of 3200 or more bits concatenated with some error correction bits. For some applications requiring less security, fewer feature vector bits may be used. Suppose only 100 bits are used, and further suppose that the application tends to produce fewer bit errors (say 15% or less). In such scenarios an alternate approach would be to simply record the secure hash of those bits. An attacker, without the benefit of the original, is forced to guess among 2^{100} bit sequences, checking guesses against the hash. Unfortunately, this leaves the naïve authentication process with no way to do error correction other than to guess among the roughly 10^{17} strings within Hamming distance of 15 of the sequence extracted from a page—easier, but also daunting.

Fortunately, there is a better approach for the authentication process. Recall that the bits of the fingerprint are taken as the signs of a series of dot products (patches and templates). We have observed that these dot products are well-modeled by samples from a truncated normal distribution. Moreover, we have also observed that the flipped bits mostly come from dot products near zero, and that the bit-flipping process

seems to be well-modeled by the addition of “noise” also selected from a truncated normal distribution (with smaller standard deviation than that of the “signal”). With this model in hand, the verification process can search for bit strings similar to the extracted fingerprint while taking into account which bits are more likely to have flipped. Specifically, the process flips bits at random with probability relative to the likelihood that the bit has flipped, each time checking against the secure hash. We simulated this approach and found that about 90% of the time it will find the correct string within 10^6 guesses for the example distribution described above.

The benefits of this approach are that it is simple to implement and provides *no* information to an attacker in the form of error-correction bits. The main disadvantages are that it does not scale well to longer bit sequences and that the stochastic nature of the algorithm provides only probabilistic guarantees of running time. Therefore, it would likely be used only in conjunction with other approaches. For example, an application might attempt this method for offline verification and fall back to an online method in cases when it fails.