Hindawi Publishing Corporation
EURASIP Journal on Information Security
Volume 2008, Article ID 895174, 19 pages
doi:10.1155/2008/895174

Research Article

Key-Dependent JPEG2000-Based Robust Hashing for Secure Image Authentication

Gerold Laimer and Andreas Uhl

Department of Computer Sciences, University of Salzburg, Jakob-Haringerstaße 2, 5020 Salzburg, Austria

Correspondence should be addressed to Andreas Uhl, uhl@cosy.sbg.ac.at

Received 31 May 2007; Accepted 12 December 2007

Recommended by S. Voloshynovskiy

We discuss a robust image authentication scheme based on a hash string constructed from leading JPEG2000 packet data. Motivated by attacks against the approach, key-dependency is added by means of employing a parameterized lifting scheme in the wavelet decomposition stage. Attacks can be prevented effectively in this manner and the security of the scheme in terms of unicity distance is assumed to be high. Key-dependency however can lead to reduced sensitivity of the scheme. This effect has to be compensated by an increase of the hash length which in turn decreases robustness.

Copyright © 2008 G. Laimer and A. Uhl. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

1. INTRODUCTION

The widespread availability of digital image and video data has opened a wide range of possibilities to manipulate these data. Compression algorithms usually change image and video data without leaving perceptual traces. Additionally, different image processing and image manipulation tools offer a variety of possibilities to alter image data without leaving traces which are recognizable by the human visual system.

In order to ensure the integrity and authenticity of digital visual data, algorithms have to be designed which consider the special properties of such data types. On the one hand, such an algorithm should be robust against compression and format conversion, since such operations are a very integral part of handling digital data (therefore, such techniques are termed “robust authentication,” “soft authentication,” or “semifragile authentication”). On the other hand, such an algorithm should be able to detect a large amount of different intentional manipulations to such data.

Classical cryptographic tools to check for data integrity like the cryptographic hash functions MD-5 or SHA are designed to be strongly dependent on every single bit of the input data. While this property is important for a big class of digital data (e.g., compressed text, executables, etc.), classical hash functions cannot provide any form of robustness and are therefore not suited for typical multimedia data.

To account for these properties, new techniques are required which do not assure the integrity of the digital representation of visual data but its visual appearance or perceptual content. In the area of multimedia security, two types of approaches have been proposed so far: semifragile watermarking and robust/perceptual/visual multimedia hashes.

The use of robust hash algorithms for media authentication has been extensively researched in recent years. A number of different algorithms [1–9] have been proposed and discussed in literature.

Similar to cryptographic hash functions, robust hash functions for image authentication should satisfy 4 major requirements [10] (where P denotes probability, H is the hash function, $X, \hat{X}, Y$ are images, α and β are hash values, and $\{0/1\}^L$ represents binary strings of length L) as follows.

(1) Equal distribution of hash values holds

$$P[H(X) = \alpha] \approx \frac{1}{2^L}, \quad \forall \alpha \in \{0/1\}^L. \tag{1}$$

(2) Pairwise independence for visually different images X and Y: $\forall\ \alpha, \beta \in \{0/1\}^L$ holds

$$P[H(X) = \alpha \mid H(Y) = \beta] \approx P[H(X) = \alpha]. \tag{2}$$

(3) Invariance for visually similar images X and $\hat{X}$ holds

$$P[H(X) = H(\hat{X})] \approx 1. \tag{3}$$

To fulfill this requirement, most proposed algorithms try to extract image features which are invariant to slight global modifications like compression or filtering.

(4) Distinction of visually different images X and Y holds

$$P[H(X) = H(Y)] \approx 0. \tag{4}$$

This final requirement also means that given an image X, it is almost impossible to find a visually different image Y with $H(X) = H(Y)$ (or even $H(X) \approx H(Y)$). In other words, it should be impossible to create a forgery which results in the same hash value as the original image.

A robust visual hashing scheme usually relies on a technique for feature extraction as the initial processing stage, often transformations like DCT or wavelet transform [7] are used for this purpose. Subsequently, the features (e.g., a set of carefully selected transform coefficients) are further processed to increase robustness and/or reduce dimensionality (e.g., decoding stages of error-correcting codes are often used for this purpose). Note that the visual features selected according to requirement (3) are usually publicly known and can therefore be modified. This might threaten security, as the hash value could be adjusted maliciously to match that of another image.

For this reason, security has always been a major design and evaluation criterion [3, 9, 11] for these algorithms. Several attacks on popular algorithms have been proposed and countermeasures to these attacks have been developed. A key problem in the construction of secure hash values is the selection of image features that are resistant to common transformations. In order to ensure the algorithms' security, these features are required to be key-dependent and must not be computable without knowledge of the key used for hash construction. Key-dependency schemes used in the construction of robust hashes include key-dependent transformations [1, 4, 12], pseudorandom permutation of the data [13], randomized statistical features [8–10], and randomized quantization/clustering [14]. The majority of these approaches adds key-dependency to the feature extraction stage, only the latter technique randomizes the actual hash string generation stage. Nevertheless, even key-dependent robust hashing schemes have been successfully attacked. For example, the visual hash function (VHF) [1] projects image blocks onto key-dependent patterns to achieve key-dependency. A security weakness of VHF has been pointed out and resolved by adding block interdependencies to the algorithm [6]. As a second example, we mention the strategy to achieve key-dependency by pseudorandom partitioning of wavelet subbands before the computation of statistical features [9]. An attack against this scheme has been demonstrated [15] which can be resolved by employing key-dependent wavelet transforms [12] or the use of overlapping and nondisjoint tiling. Recently, generic ways to assess the security of visual hash functions have been proposed based on *differential entropy* [8] and *unicity distance* [16].

In this work, we investigate the security of a JPEG2000-based robust hashing scheme which has been proposed in earlier works [17, 18]. We describe severe attacks against the original scheme and propose a key-dependent lifting parameterization in the wavelet transform stage of JPEG2000 encoding as key-dependency scheme for the JPEG2000-based robust hashing scheme. We discuss robustness and sensitivity of the resulting approach and show the improved attack resistance of the key-dependent scheme. Note that we restrict our investigations to the features extracted from the JPEG2000 bitstream themselves and treat them as actual hash string even though a final processing stage eliminating redundancy, and so forth, has not yet been applied. After reviewing JPEG2000 basics, Section 2 discusses various aspects and sorts of JPEG2000-based hashing schemes and presents the attack against the approach covered in this work. In Section 3, the employed lifting parameterization is shortly described. Subsequently, we discuss properties of the key-dependent hashing approach and provide experimental evidence for its improved attack resistance. Also, its actual key-dependency and unicity distance is discussed. Section 4 concludes this paper.

2. JPEG2000-BASED (ROBUST) HASHING

Most robust hashing techniques use a custom and dedicated procedure for hash generation which differs substantially from one technique to the other. Several techniques have been proposed using the wavelet transform as a first stage in feature extraction (e.g., [3, 9, 10]). The employment of a standardized image coding technique like JPEG2000 (based on a wavelet transform as well) for feature extraction offers certain advantages as follows.

(1) Widespread knowledge on properties of the corresponding bitstream is available.

(2) A vast hardware (e.g., Analog Devices ADV202 chip) and software (official reference implementations like JJ2000 or Jasper and additional commercial codecs) repository is available.

(3) In case visual data is already given in JPEG2000 format, the hash value may be extracted with negligible effort (parsing the bitstream and extracting the hash data). In case any other visual data format is given, simply JPEG2000 compression has to be applied before extracting the features from the bitstream (this is the usual way JPEG2000-based hashing is applied).

2.1. JPEG2000 basics

The JPEG2000 [19] image coding standard uses the wavelet transform as energy compaction method. JPEG2000 may be operated in lossy and lossless mode (using a reversible integer transform in the latter case) and also the wavelet decomposition depth may be defined. The major difference between previously proposed zerotree wavelet-based image compression algorithms such as EZW or SPIHT is that JPEG2000 operates on independent, nonoverlapping blocks of transform coefficients ("codeblocks"). After the wavelet

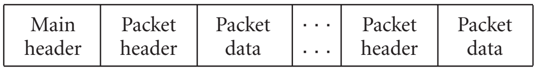


Figure 1: JPEG2000 bitstream structure.

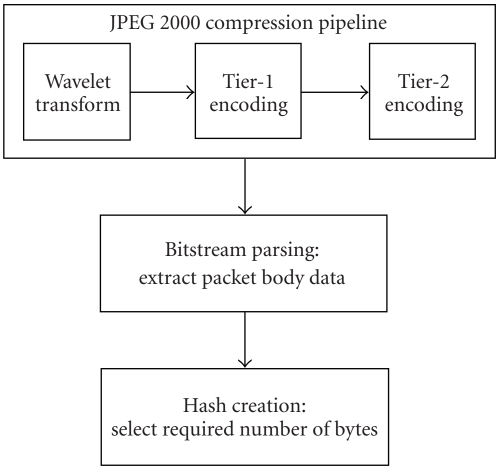


Figure 2: Block-diagram of the JPEG2000 PBHash.

transform, the coefficients are (optionally) quantized and encoded on a codeblock basis using the EBCOT scheme, which renders distortion scalability possible. Thereby, the coefficients are grouped into codeblocks and these are encoded bitplane by bitplane, each with three coding passes (except the first bitplane). While the arithmetic encoding of the codeblock is called Tier-1 coding, the generation of the rate-distortion optimal final bitstream with its scalable structure is called Tier-2 coding (see also Figure 2). The codeblock size can be chosen arbitrarily with certain restrictions.

The final JPEG2000 bitstream (see Figure 1) is organized as follows. The main header is followed by packets of data (packet bodies) each of which is preceded by a packet header. A packet body contains CCPs (codeblock contribution to packet) of codeblocks that belong to the same image resolution (wavelet decomposition level) and layer (which roughly stand for successive quality levels). Depending on the arrangement of the packets, different progression orders may be specified. Resolution and layer progression order are the most important progression orders for grayscale images.

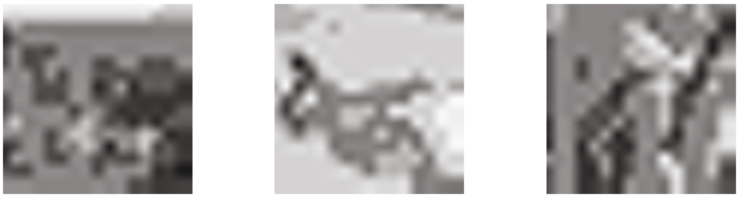
2.2. JPEG2000 authentication and hashing

Authentication of the JPEG2000 bitstream has been described in previous work. In [20], it is proposed to apply SHA-1 onto all packet data and to append the resulting hash value after the final termination marker to the JPEG2000 bitstream. Contrasting to this approach, when focusing onto robust authentication, it turns out to be difficult to insert the hash value directly into the codestream itself (e.g., after termination markers), since, in any operation which involves decoding and recompression, the original hash value would be lost. The only applications which do not destroy the hash value are purely bitstream-oriented like rate adaptation transcoding by simply dropping parts of the packet data. As a consequence, a possible solution to this dilemma would be to use a robust watermarking scheme to embed the hash value into the codestream, provided that the embedding does not change the features involved in computing the hash value. A different solution would be to signal the hash value in the context of a JPSEC [21] description. An elegant technical solution of how authentication can be applied to the entire codestream while it remains valid also for parts of it (e.g., scaled versions) has been derived using Merkle hash trees [22] (and tested with MD-5 and RSA).

JPEG2000-related information has been suggested recently to be used for content-based image search and retrieval in the context of JPSearch, a recent standardization effort of the JPEG committee. General wavelet-based features have been proposed for image indexing and retrieval which can be computed during JPEG2000 compression (cf. [23]). However, this strategy does not take advantage of the particular information available in JPEG2000 codestreams. The packet header information is specific to the visual content, and it is specific enough to be used as a fingerprint/hash for content search. Some suggestions have been made in this direction in the context of indexing, retrieval, and classification. In [23] the number of bytes spent on coding each subband (“information content”) is used for texture classification. Similarly, in [24] a set of classifiers based on the packet header (codeblock entropy) and packet body data (wavelet coefficient distribution) is used to retrieve specified textures from JPEG2000 image databases. In [25] the number of leading bitplanes is used (means and variances of the number of nonzero bitplanes in the codeblocks of each subband are computed) as a fingerprint to retrieve specific images. Finally, in [26] the same authors additionally propose to use significance bitmaps of the coefficients and significant bits histograms.

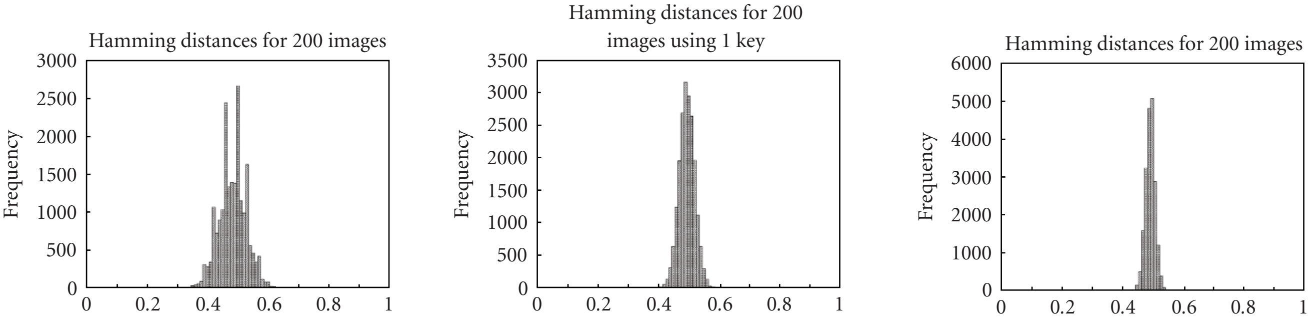
In the following, we restrict the attention to a robust hashing scheme proposed in earlier work [17, 18] which employs parts of the JPEG2000 packet body data as robust hash—we denote this approach JPEG2000 PBHash (Packet Body Hash). An image given in arbitrary format is converted into raw pixel data and compressed into JPEG2000 format. Due to the embeddedness property of the JPEG2000 bitstream, the perceptually more relevant bitstream parts are positioned at the very beginning of the file. Consequently, the bitstream is scanned from the very beginning to the end, and the data of each data packet—as they appear in the bitstream, excluding any header structures—are collected sequentially and concatenated to be then used as visual feature values (see Figure 2).

Note that it is not required to actually perform the entire JPEG2000 compression process—as soon as the amount of data required for hash generation has been output by the encoder, compression may be stopped. JPEG2000 PBHash has been demonstrated to exhibit high robustness against JPEG2000 recompression and JPEG compression [17] and provides satisfying sensitivity with respect to intentional local image modifications [18]. As it is expected due to properties of the wavelet transform, also high sensitivity



(a) (b) (c)

Figure 3: 50-byte images of the test images Goldhill, Plane, and Lena.



(a) (b) (c)

Figure 4: Hamming distances among 200 uncorrelated images.

against global geometric alterations and rescaling has been reported [18] (as determined using the Stirmark [27] attack suite). While the latter properties are prohibitive for the use of JPEG2000 PBHash in the content search scenario, these specific robustness limitations are less critical for authentication purposes. In this scenario, a specific image size can be enforced (e.g., by image interpolation) before the hash is applied; and in a nonautomated scenario, image registration may be conducted before the actual authentication process.

The visual information contained in the hash string (i.e., concatenated packet body data) may be visualized by decoding the corresponding part of the bitstream by a JPEG2000 decoder (including the header information for providing the required context information to the decoder). Figure 3 shows the visual information corresponding to a hash length of 50 bytes of the images displayed in Figures 5–7 (in fact, the images shown are severely compressed JPEG2000 images).

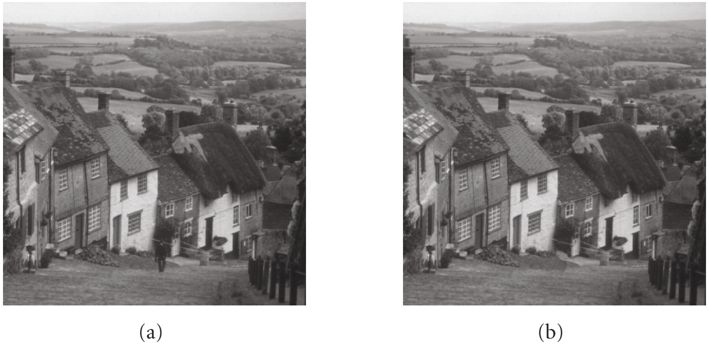
Unless noted otherwise, we use JPEG2000 with layer progression order, output bitrate set to 1 bit per pixel, and wavelet decomposition level 5 to generate the hash string. The length of the hash and the wavelet decomposition depth employed can be used as parameters to control the tradeoff between robustness and sensitivity of the hashing scheme [14]—obviously a shorter hash leads to increased robustness and decreased sensitivity (see [17, 18] for detailed results). A shallow decomposition depth is not at all suited for the JPEG2000 PBHash application since settings of this type lead to a large LL subband. For a large LL band, the hash only consists of coefficient data of the LL band corresponding to the upper part of the image (due to the size of the subband and the raster-scan order used in the bitstream assembly stage). Therefore, a certain minimal decomposition depth (e.g., down to decomposition level 3) is a must and a short hash string requires a higher decomposition depth for sensible employment of the JPEG2000 PBHash in order to avoid the phenomenon described before.

In Figure 4, we visualize the distribution of the Hamming distances computed among hashes of 200 uncorrelated images (i.e., perceptually entirely unrelated) for three parameter settings: hash-length 16 bytes with decomposition level 7, hash-length 50 bytes with decomposition level 5, and hash-length 128 bytes with decomposition level 6.

It can be observed that the distributions of the Hamming distances are centered around 0.5 as desired. The variance of the distribution is larger for the more robust settings, which is also to be expected. The influence of the wavelet decomposition level may not be immediately derived from these results but it is known from earlier experiments [18] that there is a trend to result in higher robustness for a lower decomposition level value (please refer also to the results in Section 3.2 on this issue). The reason is obvious—low-decomposition depth causes the hash string to be mainly consisting of low frequency coefficient data while differences caused by subtle image modifications are found in higher frequency coefficient data.

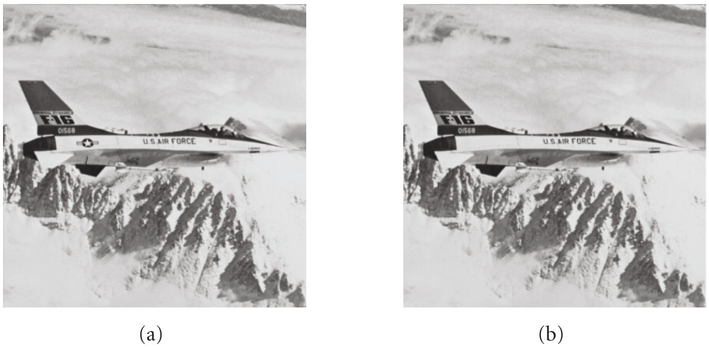
2.3. Attacks against the JPEG2000 PBHash

In order to demonstrate the definite need for key-dependency in the JPEG2000 PBHash procedure, we conduct attacks



(a) (b)

Figure 5: Test image Goldhill (original and with man removed).



(a) (b)

Figure 6: Test image Plane (original and with flag removed).

against the approach using the sightly modified images as displayed in Figures 5–7.

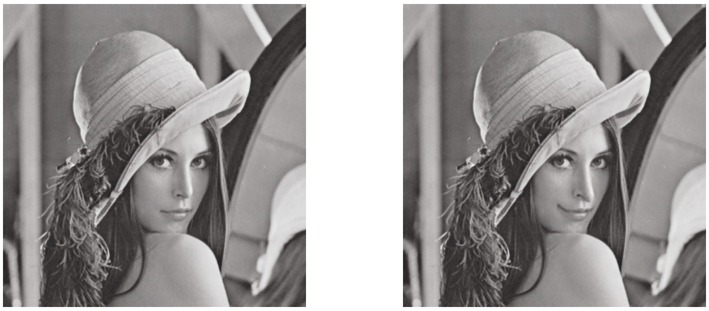
With the standard hash settings (length 50 bytes with decomposition level 5), the Hamming distance between original and modified images is 0.2 for Goldhill, 0.255 for Plane, and 0.1575 for Lena. Clearly, these modifications are detected when the modification threshold is set to a sensible value.

A possible attacker aims at maliciously tampering the modified image in a way that the hash string becomes similar or even identical to the hash string of the original image while preserving the visual content (this is the attacked image). In this way, the attacked image would be rated as being authentic by the hashing algorithm.

The attack actually conducted works as follows. Both the original and the modified images are considered in a JPEG2000 representation matching the parameters used for the JPEG2000 PBHash (if they do not match this condition, they are converted to JPEG2000). Now the first part of the bitstream of the original image (corresponding to the packet body data used for hashing) is exchanged with the corresponding part of the bitstream of the modified image resulting in the attacked image. Obviously, if the attacked image remains in JPEG2000 format, its hash exactly matches that of the original. But even if both the original and the attacked images are converted back to their source format (e.g., PNG) and the JPEG2000 PBHash is applied subsequently it turns out that the hash strings are still identical. Figure 8 shows the corresponding attacked Goldhill and Lena images. Their hash strings are identical to those of the respective originals.

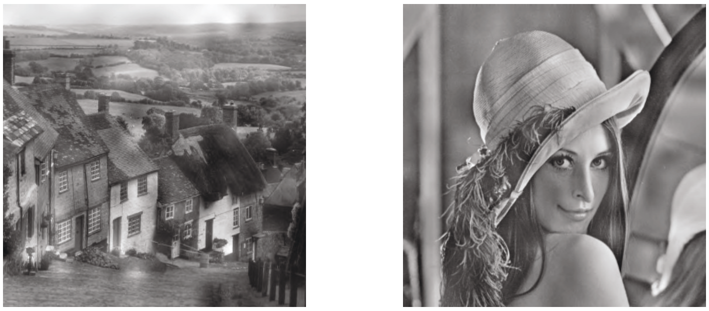
This attack is even more severe when we do not apply it to an original image and a slightly modified version as before but to completely different images. In this case we denote the attack as “collision attack” since we generate two visually entirely distinct images exhibiting an identical JPEG2000 PBHash using the same approach. Two arbitrary images (an original image and an attacked image) are either converted or already given in corresponding JPEG2000 representation. The attacked image should be modified to have a similar hash as the original image. To accomplish this, the first part of the bitstream of the attacked image is replaced by the first part of the bitstream of the original image. Figure 9 visualizes the result for the Plane and Lena image, respectively. In case the images have been present in JPEG2000 format already and remain in this format, the first image exhibits a hash string identical to that of the Lena image and the second images hash is identical to the one of the Plane image. Obviously, this does not correspond to visual perception.

This attack facilitates the modification of a given original image in a way that its hash matches that of an arbitrary different image while the visual appearance of the attacked image stays close to the original. This can be considered an extremely serious threat to the reliability of the hashing



(a) (b)

Figure 7: Test image Lena (original and with a grin).



(a) (b)

Figure 8: Attacked Goldhill and Lena images.

scheme. However, the hash values can only be made identical in case no format conversion is applied. If the attacked and original images have to be converted back to a different source format, the resulting Hamming. distances between the original and attacked versions are 0.235 and 0.113, This is in contrast to the previous case when originals and slightly modified versions have been considered. Still, those differences are significantly below the values observed among uncorrelated images (cf. Figure 4).

The demonstrated attack shows that the JPEG2000 PBHash is highly insecure in its original form and requires a significant security improvement to be useful as a reliable authentication hashing scheme.

3. KEY-DEPENDENT JPEG2000 PBHash

The concept of secret transform domains has been exploited as a key-dependency scheme to some degree in the area of multimedia security during the last years. Fridrich [28, 29] introduced the concept of DCT-type key-dependent basis functions in order to protect a watermark from hostile attacks. Unnikrishnan and Singh [30] suggest to use secret fractional Fourier domains to encrypt visual data, a technique which was also used to embed watermarks in an unknown domain [31]. The many degrees of freedom available to design a wavelet transform have also been exploited in similar manner for image and video encryption [32, 33] and to secure watermarking copy-protection [34, 35] and authentication [36] schemes.

In recent works [12, 15, 37], we have proposed to use Pollens' orthogonal filter parameterization as a generic key-dependency scheme for wavelet-based visual hash functions. In the case of an authentication hash, this strategy proved to be successful [12, 15] while it did not work out for a CBIR hash [37] due to the high robustness of the original scheme. Since the orthogonal Pollen parameterization does not easily integrate with lifting-based biorthogonal JPEG2000 filters, we propose to use a different strategy in this work, compliant to the JPEG2000 Part 2 compression pipeline. JPEG2000 Part 2 allows to extend JPEG2000 in various ways. One possibility is to employ different wavelet filters as specified in Part 1 of the standard (e.g., user designed filters) and to vary the filters during decomposition, which is discussed to be used as key-dependency scheme in the following subsection.

Using a key-dependent hashing scheme, the advantage of the JPEG2000 PBHash to generate hash strings from already JPEG2000-encoded visual data by simple parsing and concatenation is lost. An image present as JPEG2000 file needs to be JPEG2000-decoded (with the standard filters) into raw pixel data and reencoded into the key-dependent JPEG2000 domain (with the key-dependent filters) for generating the corresponding hash string.

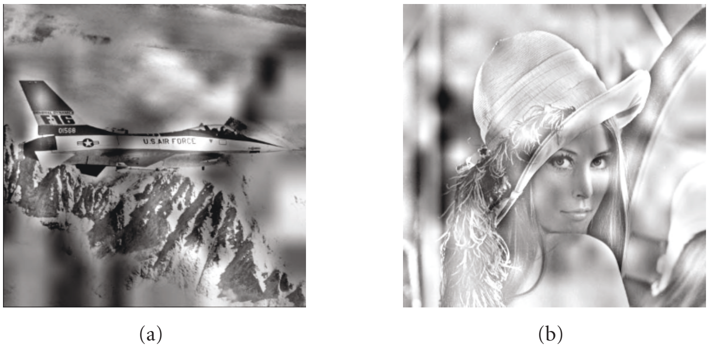


Figure 9: Collision attack: attacked Plane and Lena images.

3.1. Wavelet lifting parametrization

We use a lifting parameterization of the CDF 9/7 wavelet filter, which is described in [32] based on the work of Zhong, et al. [38], Daubechies and Sweldens [39] as well as Cohen, et al. [40]. The following conditions for the lowpass and highpass filter taps h and g are formulated [40] as follows:

$$
\begin{gathered}
h_0 + 2\sum_{n=1}^{4} h_n = \sqrt{2}, \qquad g_0 + 2\sum_{n=1}^{3} g_n = \sqrt{2}, \\
h_0 + 2\sum_{n=1}^{4} (-1)^n h_n = 0, \\
g_0 + 2\sum_{n=1}^{3} (-1)^n g_n = 0 \\
2\sum_{n=1}^{3} n^2 (-1)^n g_n = 0.
\end{gathered} \tag{5}
$$

A possible transformation of the CDF 9/7 wavelet into lifting steps, as described in [39] looks like

$$
\begin{aligned}
s_n^{(0)} &= x_{2n}, \\
d_n^{(0)} &= x_{2n+1}, \\
d_n^{(1)} &= d_n^{(0)} + \alpha\left(s_n^{(0)} + s_{n+1}^{(0)}\right), \\
s_n^{(1)} &= s_n^{(0)} + \beta\left(d_n^{(1)} + d_{n-1}^{(1)}\right), \\
d_n^{(2)} &= d_n^{(1)} + \gamma\left(s_n^{(1)} + s_{n+1}^{(1)}\right), \\
s_n^{(2)} &= s_n^{(1)} + \delta\left(d_n^{(2)} + d_{n-1}^{(2)}\right), \\
s_n &= \zeta s_n^{(2)}, \\
d_n &= \frac{d_n^{(2)}}{\zeta}.
\end{aligned} \tag{6}
$$

These lifting steps can be used to express the filter taps of h and g as functions of the four parameters α, β, γ, δ, and a scaling factor ζ. A parameterization which is only dependent on a single parameter α can be derived from these lifting steps together with condition (5) as described in [38]:

$$
\begin{aligned}
\beta &= \frac{-1}{4(1+2\alpha)^2}, \\
\gamma &= \frac{-1-4\alpha-4\alpha^2}{1+4\alpha}, \\
\delta &= \frac{1}{16}\left(4 - \frac{2+4\alpha}{(1+2\alpha)^4} + \frac{1-8\alpha}{(1+2\alpha)^2}\right), \\
\zeta &= \frac{2\sqrt{2}(1+2\alpha)}{1+4\alpha}.
\end{aligned} \tag{7}
$$

For $\alpha = -1.58613\ldots$, the original CDF 9/7 filter is obtained. The parameterization comes at virtually no additional computational cost, only the functions (7) have to be evaluated, and the lowpass and highpass synthesis filter taps for normalization have to be calculated. For a discussion on the applicability of certain parts of the range of α and on the resulting keyspace see [32]; here, we restrict the range of admissible α values to $[-6, -1.4]$.

We do not only use one single key-dependent wavelet filter in the decomposition. Instead, different key-dependent filters are used at each decomposition level of the wavelet transform and for each decomposition orientation (i.e., horizontal and vertical). These techniques originate from content adaptive image compression [41] and are denoted as "nonstationary" and "inhomogeneous" multiresolution analyses. Consequently, we actually employ $2k$ filters during a k-level wavelet decomposition—the corresponding $2k$ α's are all generated by a pseudorandom number generator from a single seed denoted as "key." However, in fact all $2k$ α's serve as potential key-material for our key-dependent JPEG2000 PBHash and especially the approximation subband data depends on all $2k$ α's.

In the following, we investigate the impact of choosing different keys on the resulting hash string, that is, whether the resulting hash is really sufficiently dependent on the key used during JPEG2000 compression. We take an image and generate its hash string with specified settings (i.e., fixed number of bytes extracted from the JPEG2000 bitstream and a certain wavelet decomposition depth)—this procedure is repeated for 100 randomly chosen keys and the Hamming

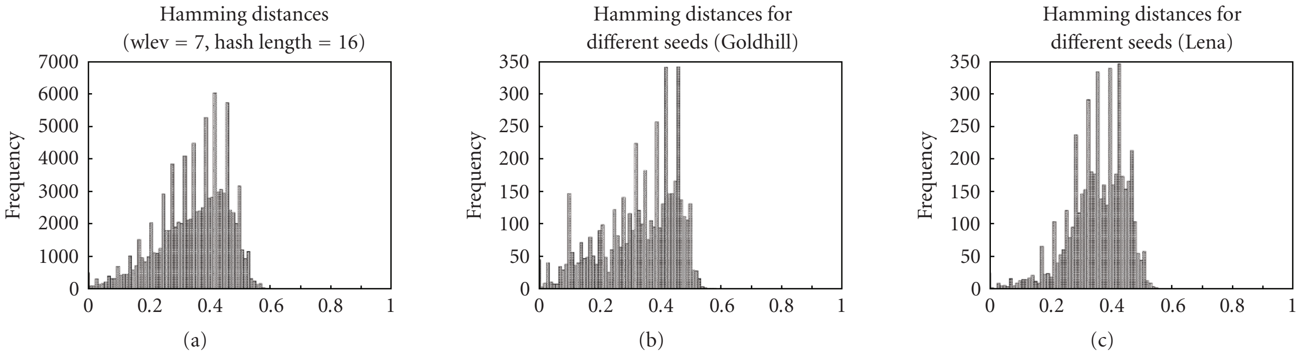


Figure 10: Hamming distances among 16-byte hashes (decomposition depth 7) generated with 100 random keys (accumulation of 20 images, Goldhill, Lena).

distance among all hash strings is computed. Figure 10 shows the resulting Hamming distance histograms for the images Goldhill and Lena where the hash string is only 16 bytes long and decomposition depth 7 is selected. The first plot in Figure 10 displays the Hamming distances among the hash strings of 100 randomly chosen keys where all corresponding distances of 20 test images are accumulated (this set of images includes Goldhill, Lena, Plane, Mandrill, Barbara, Boats, and several other test images).

It is obvious that the key-dependency scheme works in principle, however, there are several hash strings resulting in distances below 0.1. Especially when compared to the corresponding Hamming distance histogram for entirely different images (see Figure 4 left), the distribution is shifted to the left, is much broader, and exhibits many small values. The situation is much improved when increasing the hash length to 50 bytes as displayed in Figure 11. This corresponds well to our expectations since in the longer hash string more high-frequency coefficient data is included which reflects the differences among different filters much more significantly as compared to the smoothed approximation subband data. The Hamming distance histograms are shown in accumulated manner for the same set of 20 test images as before varying the wavelet decomposition depth during hash generation.

The histograms do hardly contain Hamming distances below 0.2 for all three decomposition depths with this hash length. Increasing the hash length even further to 128 bytes with a decomposition depth 6 as shown in Figure 12 for the Goldhill and Lena images and the set of 20 test images even resolves the undesired effects seen before. Most distance values are clearly above 0.3 and the histograms are clearly unimodal. Still, the distributions of the Hamming distances among different images in Figure 4 are centered better and have a lower variance. As a consequence, we recommend to use a hash length of at least 50 bytes when key-dependency of the resulting hash string is important.

3.2. Properties: sensitivity and robustness

Sensitivity is the property of a hashing scheme to detect image alterations—for the JPEG2000 PBHash, high sensitivity means that a low number of packet body bytes are required to detect image manipulations. Robustness on the other hand is the property of a hashing scheme to maintain an identical hash string even under common image processing manipulations like compression—for the JPEG2000 PBHash, high robustness means that a high number of packet body bytes are required to detect such types of manipulations. While sensitivity against intentional image modifications and robustness with respect to image compression has been discussed in detail for the key-independent JPEG2000 PBHash in previous work [17, 18], the impact of the different filters used in the key-dependency scheme on these properties of the hashing scheme is not clear yet. Therefore, we conduct several experiments on these issues.

The first experiment investigates the sensitivity against the modification of the Goldhill image shown in Figure 5. We apply the JPEG2000 PBHash to the original and the modified Goldhill images with the same key, and record the number of bytes required to detect the modification (i.e., starting from the beginning of the two hash strings, the position/number of the first unequal byte is recorded). This procedure is repeated for 100 different random keys and the results for four different decomposition depths and are shown in Figure 13 (only two different decomposition depths are shown in Figures 14 and 15). The solid line represents the value obtained with the key-independent JPEG2000 PBHash while the dots represent 100 key-dependent results. Note that (unrealistically) long hashes with 1000 bytes are used in this experiments in order to be able to capture the corresponding behavior well.

First, it is obvious that, in the plots in Figure 13, sensitivity varies among the different keys employed. Second, there is no clear trend with respect to the sensitivity of the "standard" JPEG2000 filter as compared to the parameterized versions. While for decomposition depths 4 and 5 it seems that most parameterized filters degrade sensitivity (i.e., more bytes are required to detect the modifications), decomposition depths 6 and 8 show improvements but also degradations in sensitivity of the parameterized filters as compared to the standard filter. It has to be noted that the different results for different decomposition depths discussed are specific for the

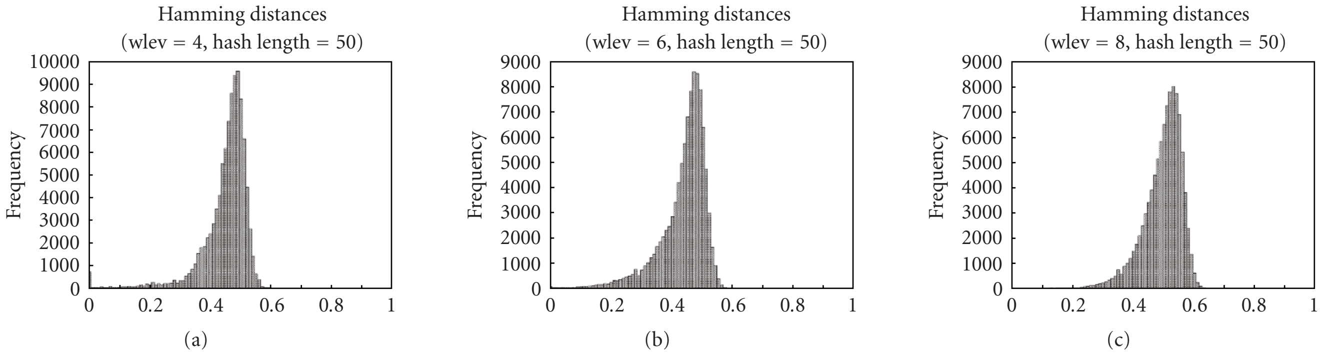


Figure 11: Hamming distances among 50-byte hashes generated with 100 random keys (decomposition depths 4, 6, and 8), accumulated over 20 images.

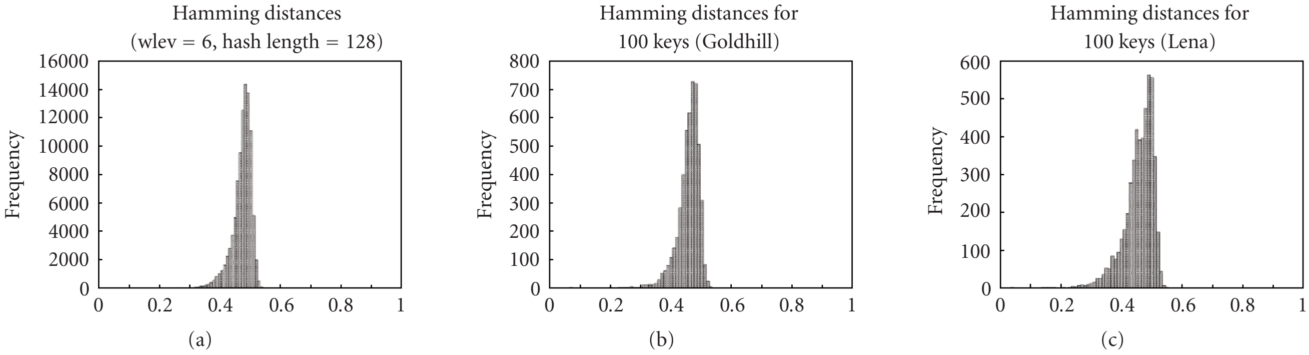


Figure 12: Hamming distances among 128-byte hashes (decomposition depth 6) generated with 100 random keys (accumulation of 20 images, Goldhill, Lena).

Goldhill image and its modification and depend significantly on the kind and severeness of the modification performed (e.g., for decomposition depth 5, we notice a sensitivity decrease for the Goldhill image; but for the Lena image as shown in Figure 15, we observe both improvements as well as degradations). In fact, it is clear that there are variations and that the "standard" filter is just one out of many other filters with no specific properties with respect to sensitivity.

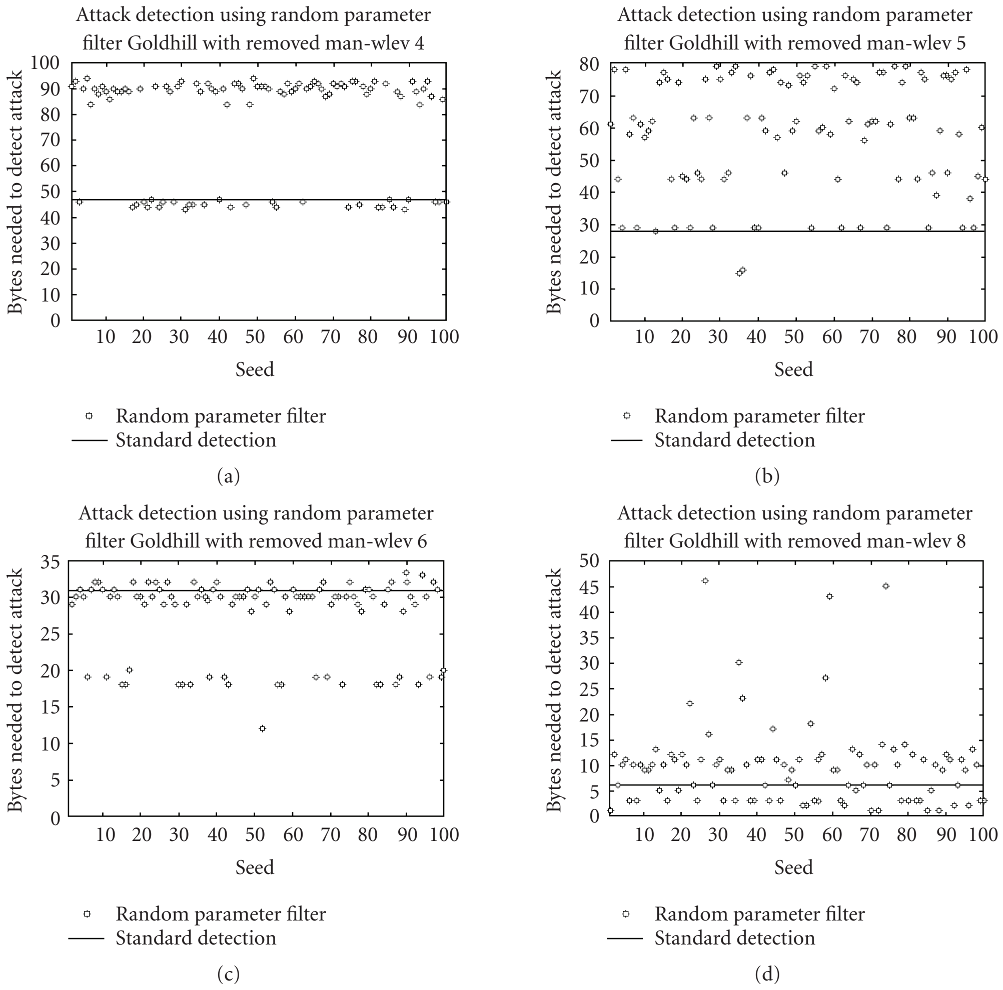
Figure 14 displays the results for decomposition depths 6 and 8 for the Plane image. While decomposition depth 6 seems to improve sensitivity, for depth 8, we notice improvements as well as degradations as compared to the standard filter.

Similarly, in Figure 15 we both observe improvements as well as degradations with respect to sensitivity for both decomposition depths considered.

The second experiment regarding sensitivity relates the variations caused by the different filters to the type and severeness of the modifications as shown in Figures 5–7. We use the JPEG2000 PBHash with 128 bytes and decomposition depth 6 and compute the Hamming distances between the original and modified images for 200 random keys (identical keys for original and modification are used). Figure 16 shows the corresponding results.

The modification performed on the Plane image is rich in contrast and affects a considerable area in the image. This modification is clearly detected for all keys assuming a detection threshold of 0.15 or lower as displayed by the middle histogram. The modification of the Goldhill image also affects a considerable number of pixels, but the contrast in this area is not changed that much. Therefore, the detection threshold had been set to 0.04 to detect the modification for all filters (which in turn negatively influences robustness of course). Finally, the modification done to Lena image affects only few pixels and hardly changes the contrast in the areas modified. Consequently, for some filter parameters, the modification is not detected at all (i.e., the Hamming distance between the hash strings is 0). Similar to the key-independent JPEG2000 PBHash, sensitivity can be controlled by setting the hash length accordingly. In the key-dependent scheme, the variations among different filters need to be considered additionally which means that longer hash strings as compared to the key-independent scheme should be used to guarantee sufficient sensitivity for all filters. Overall, employing the key-dependent hashing scheme with different filters on the same image (see Figures 10–12) results in larger Hamming distances as compared to using it with the same filters on an original and a slightly modified image (Figure 16).

The second property investigated in this subsection is robustness to common image transformations. As a typical example, we select JPEG2000 compression. We apply the



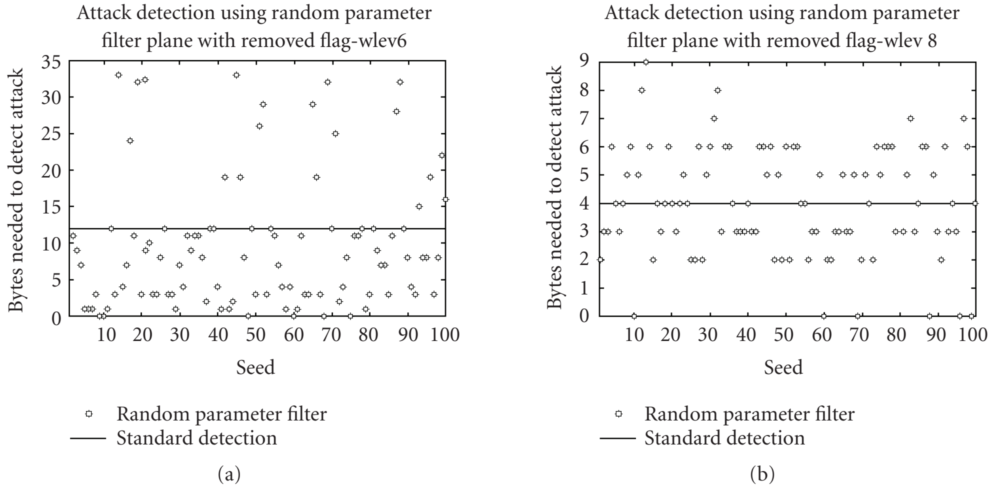
(a)

(b)

(c)

(d)

Figure 13: Number of hash bytes required to detect the removed man in the Goldhill image (hash strings generated with 100 random keys versus "standard" JPEG2000 PBHash, decomposition depths 4, 5, 6, and 8).



(a)

(b)

Figure 14: Number of hash bytes required to detect the removed flag in the Plane image (hash strings generated with 100 random keys versus "standard" JPEG2000 PBHash, decomposition depths 6 and 8).

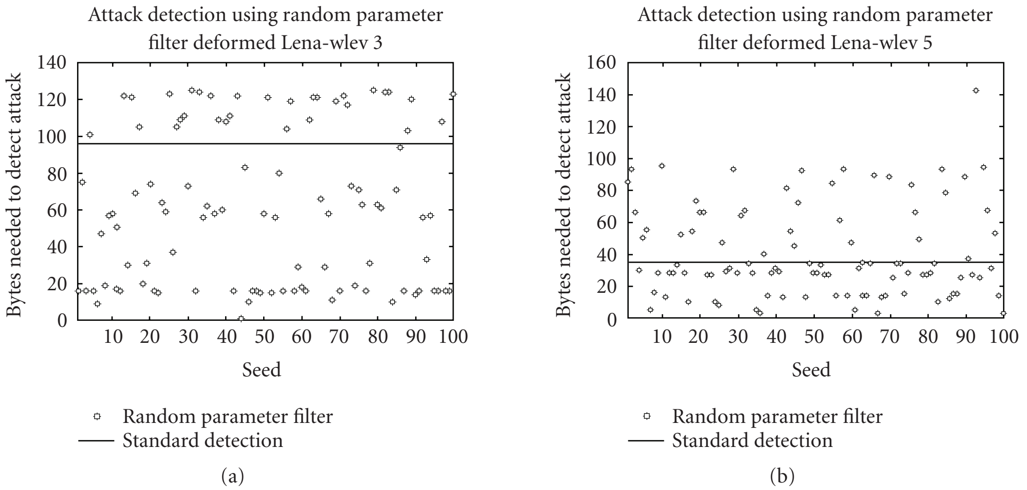


Figure 15: Number of hash bytes required to detect Lena's grin (hash strings generated with 100 random keys versus "standard" JPEG2000 PBHash, decomposition depths 3 and 5).

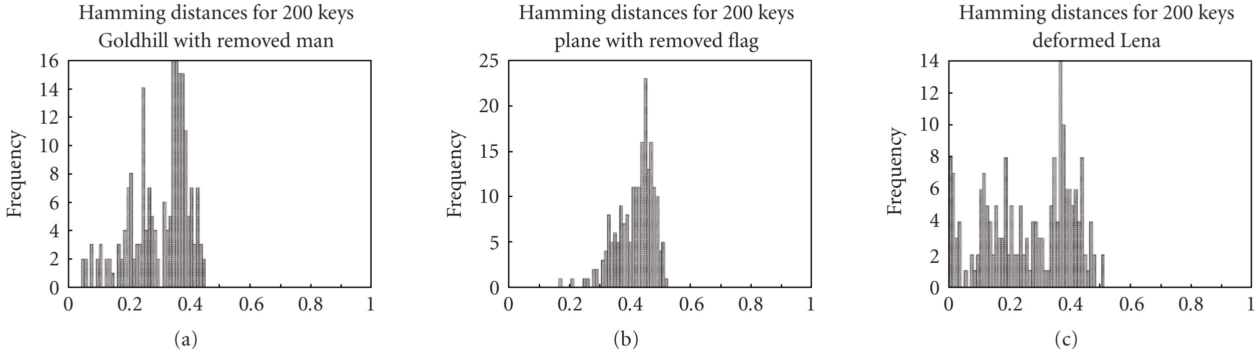


Figure 16: Hamming distances among 128-byte hashes (decomposition depth 6) generated with 200 random keys: hash of the original image is compared to the modified image, both use the same key (Goldhill, Plane, Lena).

JPEG2000 PBHash to the original and compressed Plane images (bitrate 0.5 bpp) with the same key and record the number of bytes required to detect the modification (i.e., starting from the beginning of the two hash strings, the position/number of the first unequal byte is recorded). This procedure is repeated for 100 different random keys and the results for four different decomposition depths are shown in Figure 17 (only two in Figure 18). The solid line represents the value obtained with the key-independent JPEG2000 PBHash while the dots represent 100 key-dependent results.

Similar to the investigations on sensitivity, we notice varying robustness for the parameterized filters (also concerning the relation to the robustness of the "standard" JPEG2000 filter) and inconsistent results for the different decomposition levels. However, the differences are not as pronounced as in the case of sensitivity and the results are similar for different images.

The second experiment regarding robustness relates the variations caused by the different filters to the target bitrate used for compression and the length of the hash string. We use the JPEG2000 PBHash with 16 and 128 bytes and decomposition depths 7 and 6 and compute the Hamming distances between the original and compressed images for 100 random keys (identical keys for original and compressed are used). Figure 19 shows the corresponding results for target bitrate 0.5 bpp.

At this bitrate, the 16-byte JPEG2000 PBHash provides good robustness for almost all keys (i.e., almost all Hamming distances are 0). The 128-byte hash string on the other hand produces differences up to 0.5 for some keys so that for this setting, compression robustness cannot be provided. Figure 20 shows corresponding results for a target bitrate of 0.05 bpp. At this low rate, even the 16-byte hash generates differences up to 0.5 and the 128-byte hash results in a histogram distribution similar to the case when different images have been used as input.

To summarize, we may conclude that the key-dependency introduced into the JPEG2000 PBHash has undesired effects on sensitivity and robustness. Caused by the varying sensitivity for different filters used in the hashing scheme, the length of the hash string has to be increased as compared to the key-independent scheme to detect even

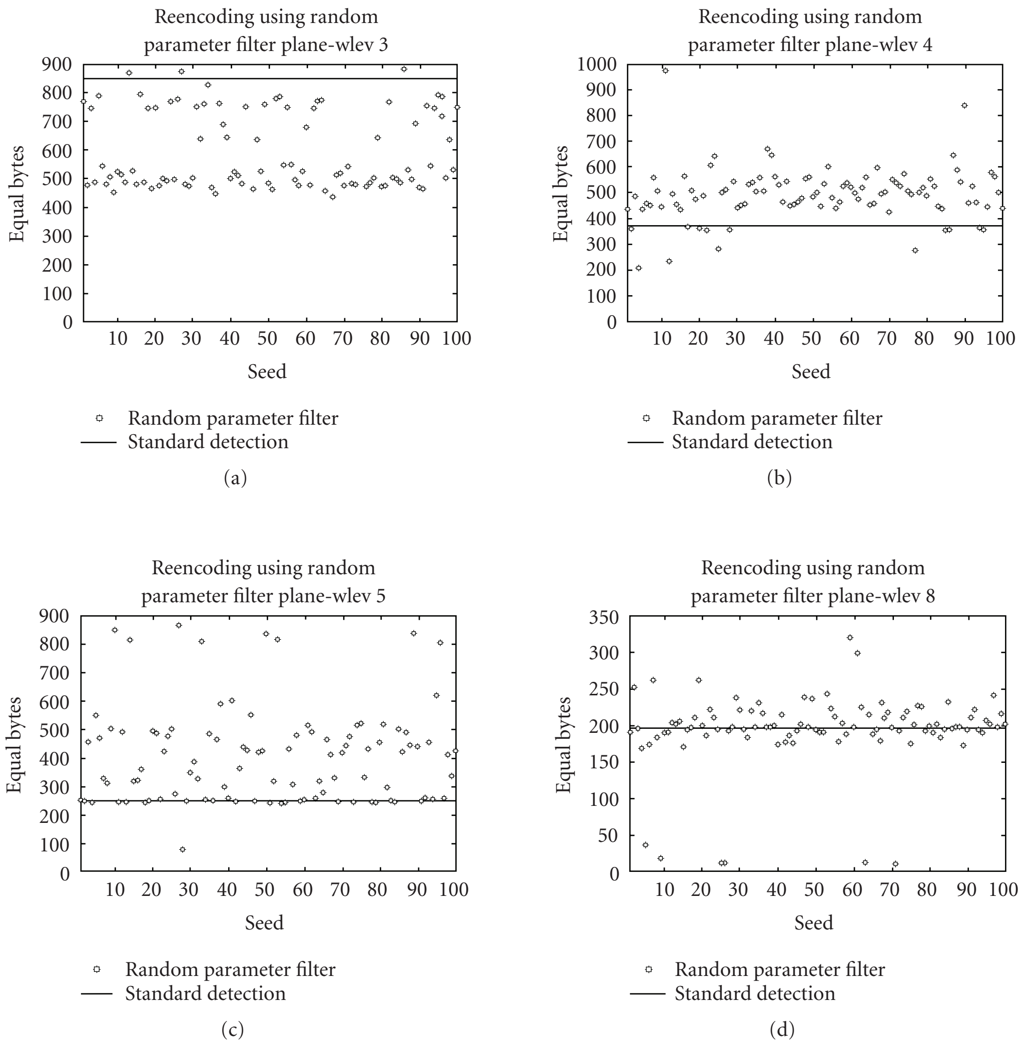


FIGURE 17: Number of hash bytes required to detect that the Plane image got compressed to 0.5 bpp (hash strings generated with 100 random keys versus "standard" JPEG2000 PBHash, decomposition depths 3, 4, 5, and 8).

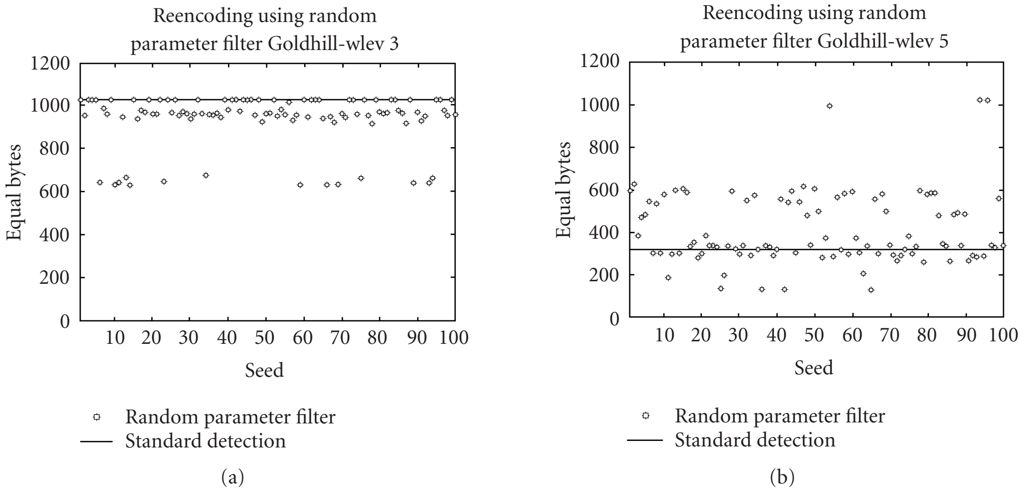
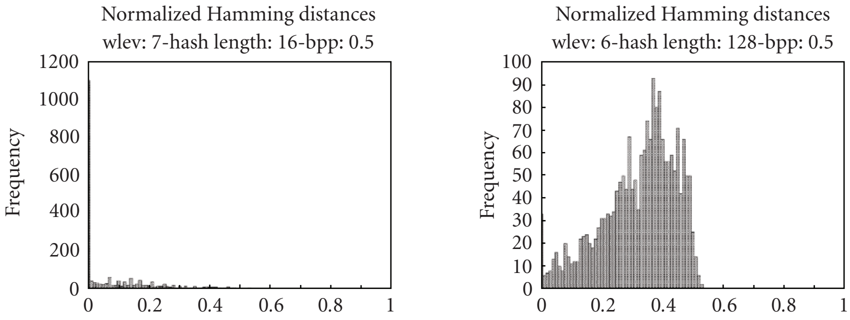
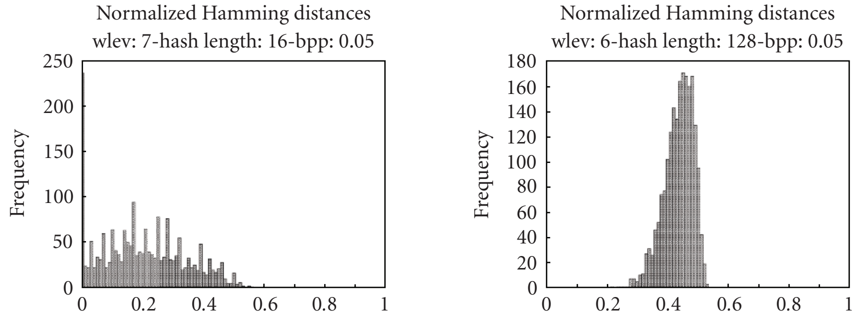


FIGURE 18: Number of hash bytes required to detect that the Goldhill image got compressed to 0.5 bpp (hash strings generated with 100 random keys versus "standard" JPEG2000 PBHash, decomposition depths 3 and 5).



(a) (b)

Figure 19: Hamming distances among hash strings generated with 100 random keys: hash of the original images is compared to a hash from a compressed image at 0.5 bpp (16-byte hash at decomposition depth 7 versus 128-byte hash at decomposition depth 6); distances are accumulated from 20 images.



(a) (b)

Figure 20: Hamming distances among hash strings generated with 100 random keys: hash of the original images is compared to a hash from a compressed image at 0.05 bpp (16-byte hash at decomposition depth 7 versus 128-byte hash at decomposition depth 6); distances are accumulated from 20 images.

small modifications reliably. For this setting, compression robustness is already hard to achieve for all filters. So, in a way, adding key-dependency to the scheme has to be paid with an aggravation of the tradeoff between sensitivity and robustness of the scheme caused by the varying respective properties of the filters used.

The sensitivity/robustness tradeoff issue has not been discussed in depth in earlier works on key-dependent wavelet transforms [12, 37] in the context of robust hashing. As already mentioned, in the CBIR scenario [37], the high robustness of the feature extraction itself prevents a satisfactory key-dependency of the hash string. In [12], parameterized (Pollen) wavelet filters as well as key-dependent wavelet packet subband structures have been investigated for their usefulness in the context of an authentication hashing scheme. Key-dependency, key-space, and attack resistance have been found to be in sensible ranges, however, the sensitivity/robustness tradeoff has not been investigated explicitly. However, the high variation in the Hamming distances found suggests varying sensitivity as found in this work. In recent work [42], we have investigated key-dependent wavelet packet subband structures as a means to add key-dependency to the JPEG2000 PBHash and found robustness to be significantly reduced as compared to the standard pyramidal subband structure, while sensitivity was found to be almost identical to the standard case. Parameterized lifting as employed in this work is clearly better suited to add key-dependency as compared to key-dependent wavelet packet structures, at least in the case of the JPEG2000 PBHash.

3.3. Attack resistance

The aim of adding key-dependency to the JPEG2000 PBHash is to prevent the attacks as described is Section 2.3. In case the key-dependent hashing scheme is used, an attacker does not know which key is used to compute the hash string for an image subject to authentication. He can just choose an arbitrary key and perform the attack as described using this key in hash generation (i.e., both the original and the modified images are JPEG2000-compressed using this particular chosen key for the attack and the part of the bitstream required for the hash is interchanged). Now, the attacker hopes that the Hamming distance between the original image and her attacked version will be small also for other keys than the single one used in her attack. Figure 21 shows an attacked version of the modified Lena image and

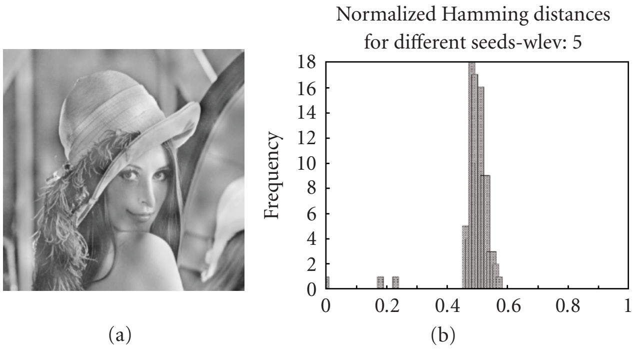


Figure 21: Attacked Lena image and Hamming distances to hash strings of the original generated with 100 random keys (50-byte hash at decomposition depth 5).

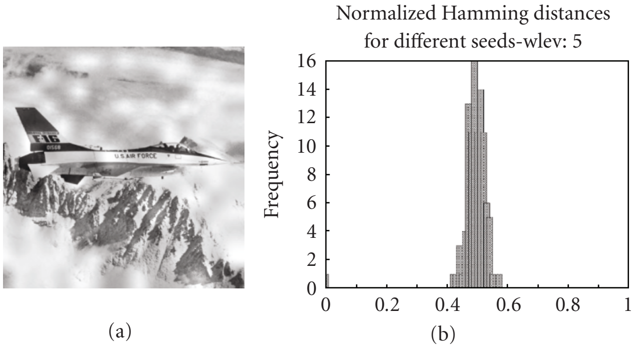


Figure 22: Attacked Plane image and Hamming distances to hash strings of the original generated with 100 random keys (50-byte hash at decomposition depth 5).

a histogram of Hamming distances between 50-byte hash strings of the original and attacked images when 100 different random keys are used in authentication (the same key is used for both original and attacked versions).

It is clearly visible that only one key results in distance 0 which is the case where the authentication key is identical to the key used for the attack. Two more distances are around 0.2, the rest is between 0.4 and 0.6. We see that the key-dependency scheme enables the JPEG2000 PBHash to identify the attacked image reliably. Figure 22 shows the same effect on the attacked Plane image where all Hamming distances are between 0.4 and 0.6 except for the single filter used in the attack.

Figure 23 verifies for the Plane image that the attack is successfully prevented with the 50-byte hash also for different decomposition depths.

When the hash length is increased to 128 bytes, the attack gets more difficult since a larger share of the bitstream data needs to be exchanged between original and modified versions possibly compromising image quality of the attacked version. Figures 24 and 25 illustrate this case for the Lena and the Goldhill images.

While the image quality of the attacked version might still be sufficient for some applications, the Hamming distance histograms clearly indicate that the attack is prevented also under these settings (in these experiments, the key used for producing the attacked versions is not included in the keys used for authentication).

In Section 2.3, we have also demonstrated the collision attack against the JPEG2000 PBHash where an image has been attacked to produce the same hash as an arbitrary original image. We cover the case of a 16-byte hash since the attacked images shown in Figure 9 hardly meet any quality requirement. In Figure 26, we visualize the attacked Lena image modified to exhibit the hash string of the Plane image. Figure 27 covers the vice versa situation. Again, the attacker has to select an arbitrary key for conducting the attack. All she can do is to hope that the hash string of the attacked image produced by other keys is similar to the string she created in the attack.

The histograms shown in Figures 26 and 27 show that again the attack can be prevented reliably. Most Hamming distances between the attacked image and the original image are >0.2 and actually all are >0.1 The hash string of the attacked image does no longer exhibit a high degree of similarity to the original image in the authentication. The same is of course true with respect to the original version of the attacked image (histograms look similar but are not shown).

The same results can be obtained for the settings corresponding to the images shown in Figure 9; however, since the visual quality of the attacked images is rather low, we do not give the plots here.

3.4. Key-dependency and security

Recently, a method for measuring the security of robust image hashing algorithms has been proposed [16]. It is based on *unicity distance*, a concept pioneered by Shannon [43] in 1949, which states that the amount of uncertainty in an encryption key reduces with each observed clear-text and cipher-text pair. This means for image hashing, that the secret key can be estimated when the key is reused multiple times on different input images. In this case, the unicity distance of a hashing scheme determines how often (i.e., for how many different images) a key can be reused, before it can be uniquely determined.

$H()$ is the image hash function, X the input image, K the secret key, and $v = H_K(X)$ the resulting hash vector. When we use the same key n-times for different input images, we get pairs of images and hash vectors $(X_1, v_1), (X_2, v_2), \ldots, (X_n, v_n)$. The conditional entropy of the secret key K can then be denoted by $E(K \mid \{(X_j, v_j)\}_{j=1}^{n})$. In general, with the increase of n, conditional entropy will decrease. To determine the unicity distance of the image-hashing algorithm, the observed image-hash pairs are taken as the input to a key estimation algorithm. The output of this algorithm (i.e., the estimated secret key) is gradually refined with the increased number of observed image-hash pairs. It is expected that the estimated key gets closer and closer to the actual key K, until they can be considered identical. The number of image-hash pairs required to recover the key K is denoted by "unicity distance."

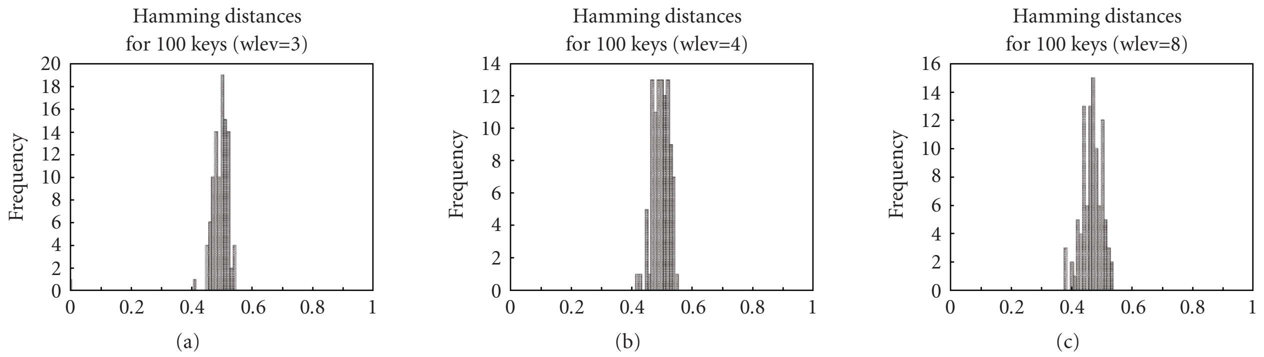


FIGURE 23: Plane image: Hamming distances between hash strings of the original and the attacked images generated with 100 random keys (50-byte hash at decomposition depths 3, 4, 8).

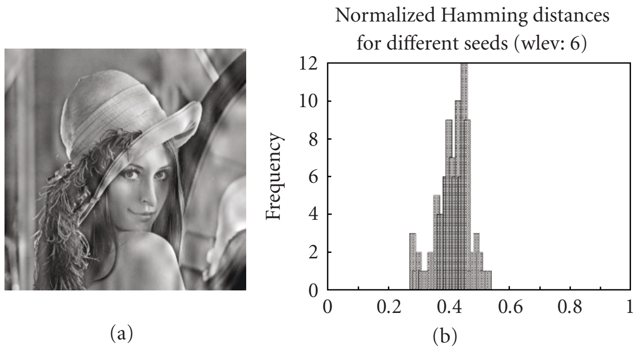


FIGURE 24: Attacked Lena image and Hamming distances to hash strings of the original generated with 100 random keys (128-byte hash at decomposition depth 6).

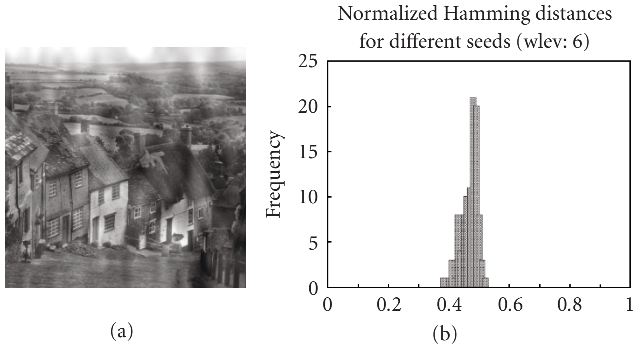


FIGURE 25: Attacked Goldhill image and Hamming distances to hash strings of the original generated with 100 random keys (128-byte hash at decomposition depth 6).

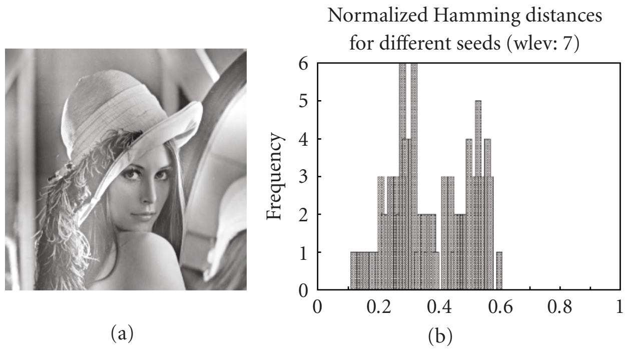


FIGURE 26: Attacked Lena image and Hamming distances to hash strings of Plane generated with 100 random keys (16-byte hash at decomposition depth 7).

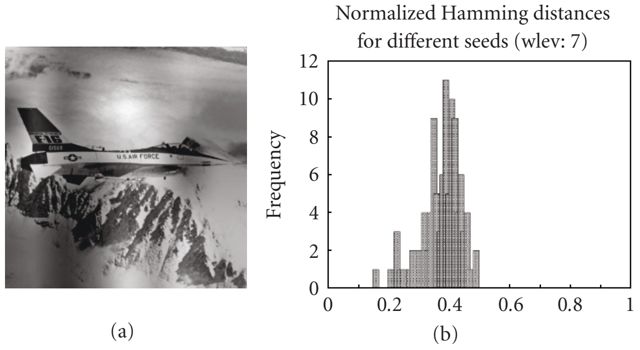
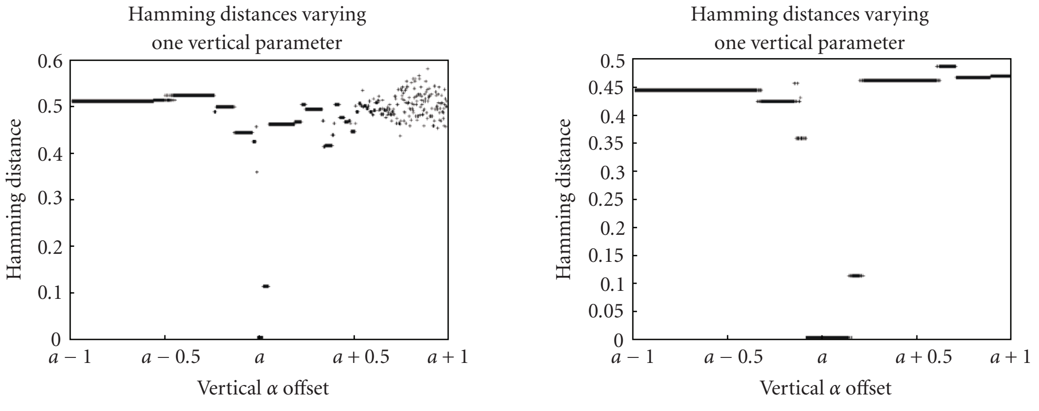


FIGURE 27: Attacked Plane image and Hamming distances to hash strings of Lena generated with 100 random keys (16 byte hash at decomposition depth 7).

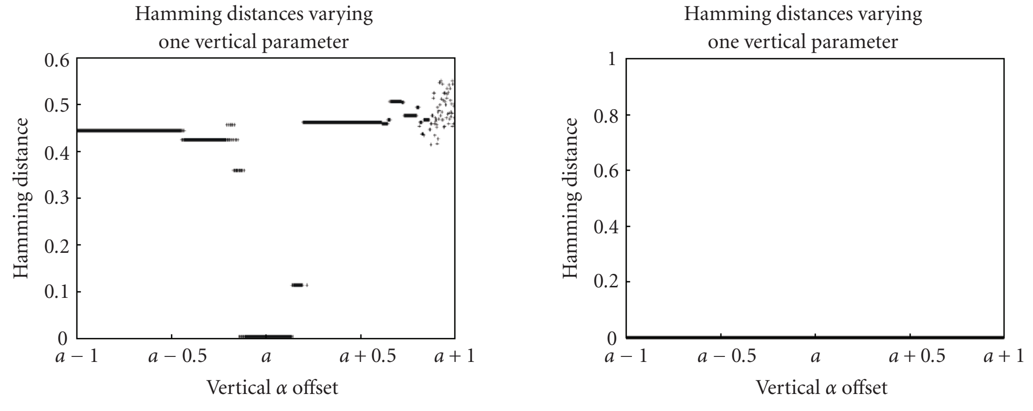
The iterative search algorithm suggested to estimate key data [16] relies on the assumption that the Hamming distances between hashes derived from similar keys get smaller the more similar the keys get. The sensitivity of the key-dependency scheme towards small changes in the key has therefore major impact on the convergence speed of this algorithm.

To investigate this issue in the context of the key-dependent JPEG2000 PBHash, we list in Table 1 the α's derived from a specific key when decomposition depth 5 is employed. We further assume that 9 out of 10 parameters used for JPEG2000 PBHash generation are already set to the correct value and only one parameter is changed slightly. Table 2 shows the Hamming distances between the hash of



(a) (b)

FIGURE 28: Hamming distances by varying one α -parameter (resolution level 1-2).



(a) (b)

FIGURE 29: Hamming distances by varying one α-parameter (resolution level 3-4).

TABLE 1: Lifting parameters derived from the key $s = 25$.

	Vertical α	Horizontal α
res. level 1	−1.4159169	−2.6140037
res. level 2	−2.1948574	−5.109782
res. level 3	−1.477038	−3.8211455
res. level 4	−5.109782	−1.7081523
res. level 5	−2.136	−1.698885

TABLE 2: Normalized Hamming distances for 10 images after varying one parameter.

Hamming distance			
Level 1	Level 2	Level 3	Level 4
$\alpha = -1.47$	$\alpha = -2.35$	$\alpha = -1.6$	$\alpha = -5.4$
0.39	0.00	0.00	0.00
0.49	0.51	0.51	0.00
0.29	0.00	0.29	0.00
0.46	0.00	0.00	0.00
0.44	0.42	0.36	0.00
0.33	0.14	0.14	0.00
0.45	0.46	0.45	0.00
0.17	0.03	0.03	0.00
0.32	0.00	0.08	0.00
0.40	0.00	0.00	0.00

length 50 bytes computed with all correct α's and a hash determined where a single vertical α is slightly incorrect (the value used as compared to Table 1 is given in the table, Hamming distances for 10 images are given).

We observe that when the resolution level-1 vertical α is slightly incorrect, all hash values still show significant Hamming differences. For incorrect level-2 and level-3 α's, some images exhibit 0 Hamming distance (e.g., 3 out of 10 at level three), others show large distances. Only at level 4 (and level 5) all images show consistently a 0 difference when all other parameters are known exactly. Note that these observations have been made under the assumption that 9 out of 10 α's are already correct without arguing how this could be achieved in an actual key-estimation algorithm.

In order to investigate the sensitivity with respect to small key changes in more detail, we determine the Hamming distances for varying the vertical parameter of each of the five resolution levels within the interval $[\alpha - 1.0, \alpha + 1.0]$ using a step size of 0.002. This leads to 1000 Hamming distances for each resolution level. Figures 28 and 29 show the results obtained for the Lena image. When varying level-1 α, we note that the Hamming distance gets 0 for an extremely small range only. Also for the level-2 and level-3 α's, only a small interval around the correct α leads to a 0 Hamming distance (i.e., $[\alpha - 0.15, \alpha + 0.15]$). Only for level 4 (and level 5—not shown) the entire range investigated leads to a 0 Hamming distance.

The assumption made so far to determine all but one α correctly is already difficult to satisfy. Considering this fact and the phenomenon that the iterative key search procedure has even problems to achieve convergence with all but one correct α at least in case the level-1 α is not yet correct makes us believe that unicity distance will be rather large for the key-dependent JPEG2000 PBHash. In fact, the assumption that the Hamming distances between hashes derived from similar keys get smaller the more similar the keys get does only hold in very small neighborhoods. Therefore, instead of an iterative key estimation technique based on successive refinement, the only way to obtain the correct key would involve a rather costly random search through a significant share of the keyspace until a configuration with small Hamming distance is found which can be systematically improved.

Consequently, we estimate the key-dependent JPEG2000 PBHash to have a rather large unicity distance.

4. CONCLUSION AND FUTURE WORK

Key-dependency is added to a JPEG2000 packet data-based hashing scheme by means of employing a parameterized lifting scheme in the wavelet decomposition stage. Attacks demonstrated against the scheme without key-dependency can be prevented effectively in this manner. Also the security of the scheme in terms of unicity distance is assumed to be high. However, key-dependency comes at a certain cost for this scheme: due to reduced sensitivity of some potentially employed filters, the hash length has to be increased as compared to the scheme without key-dependency. This leads to reduced robustness on the other hand.

In future work, we will investigate possibilities how to add key-dependency to the JPEG2000 PBHash without affecting sensitivity too much: while we have found significant variations in sensitivity among the different decompositions and filters employed, it is not yet clear if it is possible to identify subsets of the range for α where these variations could be bounded. An alternative approach is to investigate different types of key-dependency for wavelet transforms like isotropic or anisotropic wavelet packets. Additionally, we will estimate the magnitude of the keyspace available (focusing on decomposition level-dependent discretization of the α range), and we will determine the sensitivity against key modifications for the scheme in more detail to provide an approximation for an actual unicity distance value. In particular, we will investigate possibilities how to make the key-estimation procedure separable, that is, conduct key estimation for each decomposition level separately.

ACKNOWLEDGMENTS

This work has been partially supported by the Austrian Science Fund, Project no. 15170 and by the European Commission through the IST Programme under Contract IST-2002-507932 ECRYPT. The use of Dominik Engel's lifting parameterization implementation is gratefully acknowledged.

REFERENCES

[1] J. Fridrich, "Visual hash for oblivious watermarking," in *Security and Watermarking of Multimedia Contents II*, P. W. Wong and E. J. Delp III, Eds., vol. 3971 of *Proceedings of SPIE*, pp. 286–294, San Jose, Calif, USA, January 2000.

[2] J. Fridrich and M. Goljan, "Robust hash functions for digital watermarking," in *Proceedings of IEEE International Conference on Information Technology: Coding and Computing,*, pp. 178–183, Las Vegas, Nev, USA, March 2000.

[3] C.-S. Lu and H.-Y. M. Liao, "Structural digital signature for image authentication: an incidental distortion resistant scheme," in *Proceedings of the ACM Workshops on Multimedia*, pp. 115–118, Los Angeles, Calif, USA, October-November 2000.

[4] V. Monga and M. K. Mihçak, "Robust image hashing via non-negative matrix factorizations," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '06)*, vol. 2, pp. 225–228, Toulouse, France, May 2006.

[5] V. Monga, A. Banerjee, and B. L. Evans, "A clustering based approach to perceptual image hashing," *IEEE Transactions on Information Forensics and Security*, vol. 1, no. 1, pp. 68–79, 2006.

[6] R. Radhakrishnan, Z. Xiong, and N. D. Memon, "Security of the visual hash function," in *Security and Watermarking of Multimedia Contents V*, E. J. Delp III and P. W. Wong, Eds., vol. 5020 of *Proceedings of SPIE*, pp. 644–652, Santa Clara, Calif, USA, January 2003.

[7] C. J. Skrepth and A. Uhl, "Robust hash functions for visual data: an experimental comparison," in *Proceedings of the 1st Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA '03)*, F. J. Perales López, A. C. Campilho, N. P. de la Blanca, and A. Sanfeliu, Eds., vol. 2652 of *Lecture Notes in Computer Science*, pp. 986–993, Springer, Mallorca, Spain, June 2003.

[8] A. Swaminathan, Y. Mao, and M. Wu, "Robust and secure image hashing," *IEEE Transactions on Information Forensics and Security*, vol. 1, no. 2, pp. 215–230, 2006.

[9] R. Venkatesan, S.-M. Koon, M. H. Jakubowski, and P. Moulin, "Robust image hashing," in *Proceedings of the International Conference on Image Processing (ICIP '00)*, vol. 3, pp. 664–666, Vancouver, BC, Canada, September 2000.

[10] M. K. Mihçak and R. Venkatesan, "New iterative geometric methods for robust perceptual image hashing," in *Proceedings of the Workshop on Security and Privacy in Digital Rights Management*, vol. 2320, pp. 13–21, Philadelphia, Pa, USA, November 2001.

[11] A. Swaminathan, Y. Mao, and M. Wu, "Security of feature extraction in image hashing," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*

(ICASSP ’05), vol. 2, pp. 1041–1044, Philadelphia, Pa, USA, March 2005.

[12] A. Meixner and A. Uhl, “Security enhancement of visual hashes through key dependent wavelet transformations,” in *Proceedings of the 13th International Conference on Image Analysis and Processing (ICIAP ’05)*, F. Roli and S. Vitulano, Eds., vol. 3617 of *Lecture Notes in Computer Science*, pp. 543–550, Springer, Cagliari, Italy, September 2005.

[13] H. Özer, B. Sankur, N. Memon, and E. Anarim, “Perceptual audio hashing functions,” *EURASIP Journal on Applied Signal Processing*, vol. 2005, no. 12, pp. 1780–1793, 2005.

[14] V. Monga and B. L. Evans, “Perceptual image hashing via feature points: performance evaluation and tradeoffs,” *IEEE Transactions on Image Processing*, vol. 15, no. 11, pp. 3452–3465, 2006.

[15] A. Meixner and A. Uhl, “Analysis of a wavelet-based robust hash algorithm,” in *Security, Steganography, and Watermaking of Multimedia Contents VI*, E. J. Delp III and P. W. Wong, Eds., vol. 5306 of *Proceedings of SPIE*, pp. 772–783, San Jose, Calif, USA, January 2004.

[16] Y. Mao and M. Wu, “Unicity distance of robust image hashing,” *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 3, pp. 462–467, 2007.

[17] R. Norcen and A. Uhl, “Robust authentication of the JPEG2000 bitstream,” in *Proceedings of the 6th IEEE Nordic Signal Processing Symposium (NORSIG ’04)*, pp. 121–124, Espoo, Finland, June 2004.

[18] R. Norcen and A. Uhl, “Robust visual hashing using JPEG2000,” in *Proceedings of the 8th IFIP TC6/TC11 Conference on Communications and Multimedia Security (CMS ’04)*, D. Chadwick and B. Preneel, Eds., pp. 223–236, Springer, Lake Windermere, UK, September 2004.

[19] D. Taubman and M. W. Marcellin, *JPEG2000: Image Compression Fundamentals, Standards and Practice*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2002.

[20] R. Grosbois, P. Gerbelot, and T. Ebrahimi, “Authentication and access control in the JPEG2000 compressed domain,” in *Applications for Digital Image Processing XXIV*, vol. 4472 of *Proceedings of SPIE*, pp. 95–104, San Diego, Calif, USA, July 2001.

[21] J. Apostolopoulos, S. Wee, F. Dufaux, T. Ebrahimi, Q. Sun, and Z. Zhang, “The emerging JPEG2000 security (JPSEC) standard,” in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS ’06)*, pp. 3882–3885, Island of Kos, Greece, May 2006.

[22] C. Peng, R. H. Deng, Y. Wu, and W. Shao, “A flexible and scalable authentication scheme for JPEG2000 image codestreams,” in *Proceedings of the 11th ACM International Conference on Multimedia (MULTIMEDIA ’03)*, pp. 433–441, Berkeley, Calif, USA, November 2003.

[23] A. Tabesh, A. Bilgin, K. Krishnan, and M. W. Marcellin, “JPEG2000 and motion JPEG2000 content analysis using codestream length information,” in *Proceedings of the Data Compression Conference (DCC ’05)*, pp. 329–337, Snowbird, Utah, USA, March 2005.

[24] A. Descampe, P. Vandergheynst, C. De Vleeschouwer, and B. Macq, “Coarse-to-fine textures retrieval in the JPEG2000 compressed domain for fast browsing of large image databases,” in *Proceedings of the International Workshop on Multimedia Content Representation, Classification and Security (MRCS ’06)*, B. Günsel, A. K. Jain, A. M. Tekalp, and B. Sankur, Eds., vol. 4105 of *Lecture Notes in Computer Science*, pp. 282–289, Springer, Istanbul, Turkey, September 2006.

[25] C. Liu and M. Mandal, “Fast image indexing based on JPEG2000 packet header,” in *Proceedings of the ACM Workshops on Multimedia: Multimedia Information Retrieval*, pp. 46–49, Ottawa, Ontario, Canada, October 2001.

[26] M. K. Mandal and C. Liu, “Efficient image indexing techniques in the JPEG2000 domain,” *Journal of Electronic Imaging*, vol. 13, no. 1, pp. 182–190, 2004.

[27] F. A. P. Petitcolas, M. Steinebach, F. Raynal, J. Dittmann, C. Fontaine, and N. Fatès, “Public automated web-based evaluation service for watermarking schemes: stirMark benchmark,” in *Security and Watermarking of Multimedia Contents III*, vol. 4314 of *Proceedings of SPIE*, pp. 575–584, San Jose, Calif, USA, January 2001.

[28] J. Fridrich, A. C. Baldoza, and R. J. Simard, “Robust digital watermarking based on key-dependent basis functions,” in *Proceedings of the 2nd International Workshop on Information Hiding (IH ’98)*, D. Aucsmith, Ed., vol. 1525 of *Lecture Notes in Computer Science*, pp. 143–157, Springer, Portland, Ore, USA, April 1998.

[29] J. Fridrich, “Key-dependent random image transforms and their applications in image watermarking,” in *Proceedings of the International Conference on Imaging Science, Systems, and Technology (CISST ’99)*, pp. 237–243, Las Vegas, Nev, USA, June 1999.

[30] G. Unnikrishnan and K. Singh, “Double random fractional Fourier-domain encoding for optical security,” *Optical Engineering*, vol. 39, no. 11, pp. 2853–2859, 2000.

[31] I. Djurovic, S. Stankovic, and I. Pitas, “Digital watermarking in the fractional Fourier transformation domain,” *Journal of Network and Computer Applications*, vol. 24, no. 2, pp. 167–173, 2001.

[32] D. Engel and A. Uhl, “Parameterized biorthogonal wavelet lifting for lightweight JPEG2000 transparent encryption,” in *Proceedings of the 7th Workshop on Multimedia and Security (MM-SEC ’05)*, pp. 63–70, New York, NY, USA, August 2005.

[33] A. Pommer and A. Uhl, “Selective encryption of wavelet-packet encoded image data: efficiency and security,” *Multimedia Systems*, vol. 9, no. 3, pp. 279–287, 2003.

[34] W. M. Dietl, P. Meerwald, and A. Uhl, “Protection of wavelet-based watermarking systems using filter parametrization,” *Signal Processing*, vol. 83, no. 10, pp. 2095–2116, 2003.

[35] W. M. Dietl and A. Uhl, “Robustness against unauthorized watermark removal attacks via key-dependent wavelet packet subband structures,” in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME ’04)*, vol. 3, pp. 2043–2046, Taipei, Taiwan, June 2004.

[36] J. Huang, J. Hu, D. Huang, and Y. Q. Shi, “Improve security of fragile watermarking via parameterized wavelet,” in *Proceedings of the International Conference on Image Processing (ICIP ’04)*, vol. 2, pp. 721–724, Singapore, October 2004.

[37] A. Meixner and A. Uhl, “Robustness and security of a wavelet-based CBIR hashing algorithm,” in *Proceeding of the 8th Workshop on Multimedia and Security (MM-Sec ’06)*, pp. 140–145, Geneva, Switzerland, September 2006.

[38] G. Zhong, L. Cheng, and H. Chen, “A simple 9/7-tap wavelet filter based on lifting scheme,” in *Proceedings of the International Conference on Image Processing (ICIP ’01)*, vol. 2, pp. 249–252, Thessaloniki, Greece, October 2001.

[39] I. Daubechies and W. Sweldens, “Factoring wavelet transforms into lifting steps,” *Journal of Fourier Analysis and Applications*, vol. 4, no. 3, pp. 245–267, 1998.

[40] A. Cohen, I. Daubechies, and J.-C. Feauveau, “Biorthogonal bases of compactly supported wavelets,” *Communications on*

Pure and Applied Mathematics, vol. 45, no. 5, pp. 485–560, 1992.

[41] A. Uhl, "Image compression using non-stationary and inhomogeneous multiresolution analyses," *Image and Vision Computing*, vol. 14, no. 5, pp. 365–371, 1996.

[42] G. Laimer and A. Uhl, "Improving security of JPEG2000-based robust hashing using key-dependent wavelet packet subband structures," in *Proceedings of the 7th WSEAS International Conference on Wavelet Analysis & Multirate Systems (WAMUS '07)*, P. Dondon, V. Mladenov, S. Impedovo, and S. Cepisca, Eds., pp. 127–132, Arcachon, France, October 2007.

[43] C. E. Shannon, "Communication theory of secrecy systems," *Bell System Technical Journal*, vol. 28, no. 4, pp. 656–715, 1949.