Security Enhancement of Visual Hashes Through Key Dependent Wavelet Transformations*

Albert Meixner[1] and Andreas Uhl[2]

[1] Department of Computer Science,Duke University, USA
[2] Department of Scientific Computing,Salzburg University, Austria

Abstract. Parameterized wavelet filters and wavelet packet subband structures are discussed to be used as key dependent wavelet transforms in order to enhance the security of wavelet based hashing schemes. Experiments show that key dependency and keyspace of the hashing scheme considered have been significantly improved. The attack resistance could only be slightly enhanced by using parametrized wavelet filters.

1 Introduction

The use of robust hash functions for image authentication has become a popular field of research. A key problem in the construction of secure hash values is the selection of image features that are resistant to common transformations. To ensure the algorithm's security [7], these features are required to be key dependent and not computable without knowledge of the key used for hash construction. For example, the Visual Hash Function (VHF) [3] projects image blocks onto key dependent patterns to achieve this goal.

In recent work [5] we have shown a simple attack against a wavelet-based robust hashing scheme introduced by Venkatesan et al. [9]. In this context we have pointed out that a key-dependent parameterized wavelet transform could serve as a generic way to improve the security of such algorithms. A similar approach has been proven to enhance the security of various watermarking schemes, using both key-dependent filter parameterization [2] and key-dependent wavelet packet subband structures [1].

In this paper two different methods of adding key dependency to the wavelet transformation are proposed. In the experiments, these wavelet transformations are evaluated with respect to their sensibility to changes in the key material and the available keyspace when used in the context of the hahsing scheme of Venkatesan et al. [9]. Finally we test the usefulness of those schemes to counter the attack [5] against this algorithm.

2 Key-Dependency Schemes

Pseudo Random Partitioning. A common approach to generate secret image features is to first create a pseudo-random partitioning of the image and compute

* This work has been partially supported by the Austrian Science Fund FWF, project no. P15170.

F. Roli and S. Vitulano (Eds.): ICIAP 2005, LNCS 3617, pp. 543–550, 2005.

features independently for every partition. The exact values of the features can not be computed without knowledge of the key used to seed the PRNG, because the regions on which the features are computed are not known.

Random paritioning is used as original key-dependency scheme in the hash algorithm of Venkatesan et al. [9]. Its use is orthogonal to the following two schemes and can be easily combined with either of them to further increase security (which will be done in our experiments).

Random Wavelet Packet Decomposition. In the classical wavelet transformation only the low-low-sub-band can be further decomposed, resulting in the typical pyramidal structure. Wavelet packet decomposition [1] removes this constraint and allows to further decompose any sub-band. The decision which sub-bands are decomposed is either determined by a given structure or based on some measure of optimality.

By using a pseudo random number generator to decide, if a sub-band should be further decomposed, we can make the decomposition structure key dependent. This approach has been shown to be effective in selective image encryption [6] and in securing watermarking schemes [1].

Parameterized Filters. Wavelet decomposition typically uses fixed, well known filters, such as the Daubechies filters. There are also methods to generate families of wavelet filters from a number of parameters, that can be freely chosen (we employ a familiy of parameterized orthogonal Daubechies wavelet filters [8]). If these parameters are kept secret, they can be used as a key for the decomposition. Similar to the wavelet packet case, this type of key-dependency has been used before in selective image encryption [4] and watermarking [2].

3 Experiments and Results

We have tested both proposed schemes by including them into a authentication hash algorithm introduced by Venkatesan et al. [9]. The original algorithm achieves key dependency through random partitioning. We use this algorithm as a base case:

- The image is transformed, using a 3-level pyramidal wavelet transformation
- For each of the resulting subbands a feature vector F_i is calculated. This is done by randomly partitioning the subband and calculating a statistical measure for each region.

  For the approximation the statistical measure used is the arithmetic average, for all other subbands the variance is computed.
- The real number elements of each F_i are projected to $\{0 \ldots 7\}$ using randomized rounding. The resulting values are concatenated to form a preliminary hash string H_p.
- The hash string H_p is shortened by feeding it into the decode stage of a Reed-Muller error correcting code. This does not only shorten the hash string, but also improves robustness.

Table 1. Hamming distances among a set of images

	baboon	barb	boat	jet	lena	peppers	truck	zelda
baboon	0.00	0.43	0.46	0.32	0.35	0.32	0.37	0.38
barb	0.43	0.00	0.35	0.39	0.37	0.44	0.47	0.40
jet	0.32	0.39	0.39	0.00	0.31	0.40	0.48	0.34
lena	0.35	0.37	0.40	0.31	0.00	0.36	0.45	0.30

– In the final step a Linear Code algorithm is applied to the hash, again both shortening it and increasing robustness.

To obtain an initial estimate and upper bound of the Hamming distance threshold for considering an image untampered, a set of different images is compared.

The Hamming distance between two independent images is consistently below the optimal distance of $\frac{1}{2}$. This is mainly a result of the fixed values used in the randomized rounding procedure, which favor the lower and upper bounds, and a non uniform distribution of features values. For more detailed results and some improvements of the alorithm see [5].

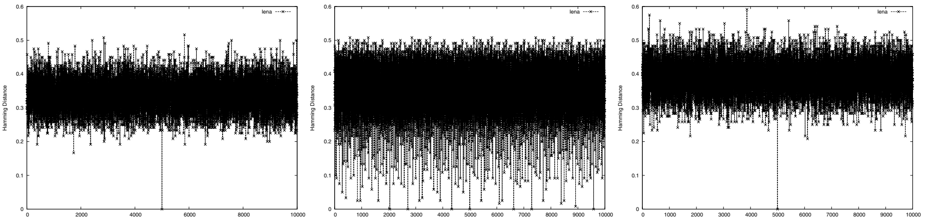
3.1 Key Dependency

A key dependency scheme can only improve security if the choice of the key has a significant impact on the resulting hash value. All following figures show the normalized Hamming distance of a hash created with some fixed key value to other hashes, produced with varying other key values. Key values are displayed along the ordinate, resulting Hamming distances along the abscissa.

The random partitioning approach, though vulnerable by a simple attack (see [5] and subsection 3.3), is very effective in adding key dependency, with average Hamming distance 0.336 and very few keys reaching values below 0.2 (see Fig. 1(a)). The figure shows the results 10000 different partitions, compared to a fixed key at position 5000. A similar phenonemon (i.e. security weaknesses in spite of a key-dependent hash) was pointed out by Radhakrishnan et al. [7] for the block-based VHF. This contradictory behaviour was improved by adding block inter-dependencies to VHF.

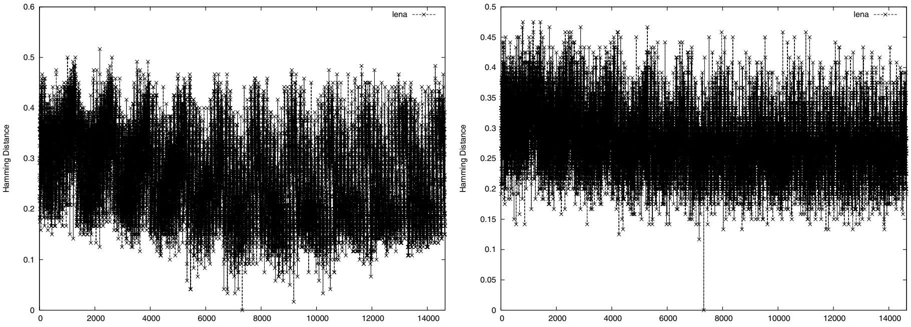
Random wavelet packet decompositions with a constant decomposition probability for all subbands makes shallow trees far more likely than deep trees. This increases the chance of collisions, especially for shallow trees. Following a previous suggestion [6], we use a higher initial decomposition probability for the first decomposition level and decrease it subsequently for every subsequent decomposition recursion (we use a base value of 0.9 ($p = 0.55$) and a change factor of -0.1 [6]). The obtained average Hamming distance (Fig. 1(b)) is 0.3570 and about 0.73% of all distances are below 0.1. However, we result in 20 “almost” correct keys (distance < 0.05) which makes the approach less reliable.

Even with random decomposition in place, the key of the standard algorithm required to create partitions for extracting localized feature vectors may



(a) Random partitioning (b) Random decomposition (c) Random decomposition & partitioning

Fig. 1. Key dependency test: Hamming distances between hashes generated with different keys



(a) Random filters (b) Random filters & partitioning

Fig. 2. Key dependency test: Hamming distances between hashes generated with different keys

be varied as well, thus increasing the key space and possibly overall security. Fig. 1(c) shows key dependency results for varying both keys. The average distance for this setup increases to 0.3884 with no incorrect keys reaching distances below 0.1. Combining both strategies obviously significantly increases the keyspace while maintaining the high sensitivity to key variations of the original standalone random partitioning scheme.

Experiments concerning filter parametrization are based on a parameterized filter with 4 parameters $(1.0, 1.5, -2.0, -1.0)$, all parameters were modified in a range of ± 1.0 in steps of 0.2, resulting in $11^4 = 14641$ combinations. The correct key for this test is 7320. The results for parameterized filters are almost as good as the random partition scheme, with an average of 0.265 and only 0.53% of the keys below 0.1 (see Fig. 2(a)).

Similar to the random decomposition, using parameterized filters adds key dependency to the decomposition stage. Thus, the parameterization key can also be combined with the standard partitioning key used during a later stage of the scheme. When both keys are used, the average hamming distance increases slightly to 0.2795, additionally there are no more incorrect keys reaching values below 0.1 (see Fig. 2(b)). Again, combining the two schemes maintaines sensitivity towards key alterations while increasing the keyspace.

3.2 Key Space

A major concern of any key dependent algorithm is the number of distinct keys that can be used. If the number of keys is too small, the scheme is vulnerable to brute force attacks. The discrete key space of both random partitioning and random decomposition grows exponentially with a free algorithm parameter (e.g., following the formula given in [6], a decomposition depth of 5 leads to $\approx 2^{1043}$ different keys in random decomposition). Thus the size of the key space can be easily adjusted and it seems that a suitable number of keys is available for any level of security desired. However, a bigger number of keys may have some undesired side effect on the overall algorithm.

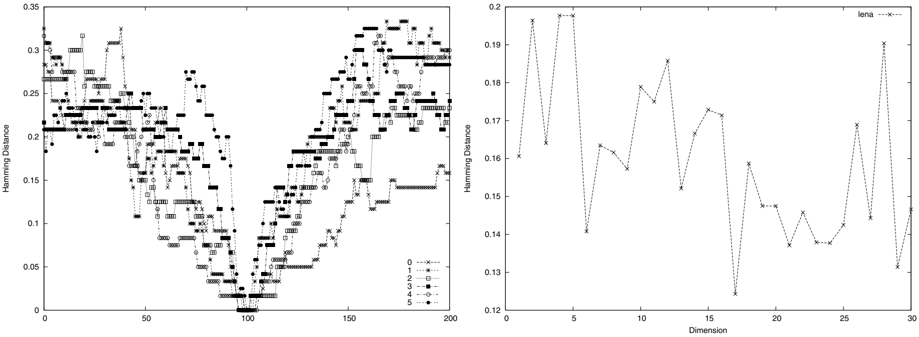
In random partitioning, the areas get smaller with an increasing number of keys. This makes the hash more sensitive to minor image modifications and many keys will produce fairly similar results. Random decompositions suffers from the fact, that high decomposition depth leads to a big number of very similar tree structures, which lead to identical hash values. Therefore, the keyspace needs to be set to some sensible compromise in this two cases (e.g. decomposition depth 5 is a good choice for random decomposition).

Contrasting to the previous cases, the key values are continuous rather than discrete for filter parametrization. Therefore, a quantization must be defined to determine the number of possible keys. This can be done by defining a range of valid parameters ($d_{min} \dots d_{max}$) and quantization function $Q(d) = \left\lfloor \frac{d}{q} \right\rfloor$. Now the the number of keys $f(n)$ for a filter with n parameters can be calculated: $f(n) = \left\lfloor \frac{d_{max} - d_{min}}{q} \right\rfloor^n$. The filter parametrization used is based on trigonometric functions (sin, cos). Thus, the parameters have a range of $(-\pi \dots \pi)$.

n	Keys	
1	125	$\approx 2^7$
2	15625	$\approx 2^{14}$
3	1953125	$\approx 2^{21}$
4		$\approx 2^{28}$
5		$\approx 2^{35}$
6		$\approx 2^{42}$
7		$\approx 2^{49}$
8		$\approx 2^{56}$
9		$\approx 2^{63}$

Table 2. Parameterized Filters Key Space

In the following, we determine the quantization function by a simple experiment. Fig. 3(a) shows the results, if only one parameter of a 6 dimensional parameterization is modified in the range of ± 1.0 with a step size of 0.01. There is a curve for every one of the six single parameters. The graph's values change in multiple steps, suggesting that key values within about 0.05 produce the same hash. Thus, when generating parameters from the key the granularity should be $0.05 - 0.10$ (the parameters used to create the graph were $(1.0, 1.5, -2.0, -1.0, 0.0, 0.5)$). To be on the safe side, we limit the the distance in a single parameter between two keys to be no smaller than 0.1. Using these values, the number of available keys can be calculated as: $f(n) = \left\lfloor \frac{\pi - (-\pi)}{0.1} \right\rfloor^n = \lfloor 20.0 \cdot \pi \rfloor^n \approx 62.8^n$. The number of resulting keys dependent on n is shown in Table 2.



(a) Varying one out of six parameters

(b) Average distances for varying the first parameter

Fig. 3. Hamming distances

The granularity q is very important for the security of the scheme and might be dependent also on the number of parameters n. It seems intuitive, that the influence of a single parameter on the overall result will decrease for a higher numer of parameters. This, however, is not the case as shown in Fig. 3(b). For every filter dimension shown on the x-axis, the average Hamming distance between the hash for a fixed parameter vector and all hashes resulting from the *first* parameter of this vector being changed in the range of ±1.0 is shown on the y-axis. This average distance indicates how much influence a single parameter has on the resulting hash value – it varies significantly from 0.12 to almost 0.2 without any clear trend up or downwards for an increasing number of dimensions. Thus, d does not have to be selected dependent on n.

3.3 Attack Resistance

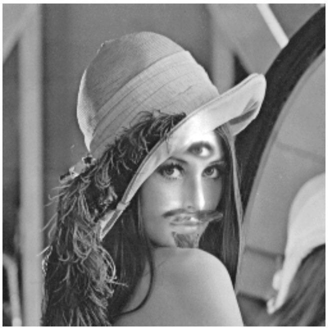
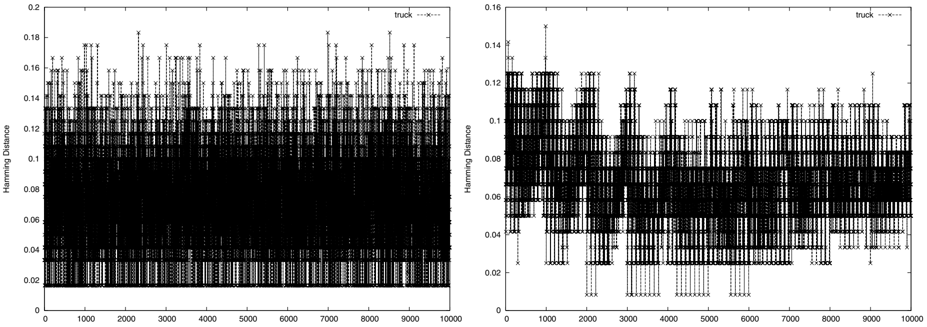


Fig. 4. Forged & attacked Lena image

The reason for the idea of enhancing the original partitioning scheme with a key dependent wavelet transformation is its vulnerability to the simple attack shown in [5]. The major problem of the use of variance and average as basis of the hash value is that both are publicly available and very easy to modify [3]. Both average and variance mostly change gradually within an image, so that if the measures of two images match within a certain partition, they will at least be similar within any other partition covering approximately the same area as well. This is exploited by the referenced attack.

Fig. 4 shows a forged and attacked version of the Lena image with a Hamming distance of 0.01 to the original. The image modification without attack mounted exhibits Hamming distance 0.12 to the original which would have been detected as forgery of course. Since this value is significantly below the Hamming distance of that between the original and a JPEG compressed version with high quality, the picture would be rated identical to the original by the hashing algorithm. This example shows the severness of this attack drastically. See [5] for more details on the attack and corresponding results.

The goal of the proposed new schemes is to eliminate feature correlations between transformations computed with different key values. Though some parameters apparently result in the exact same hash value, overall hash values strongly depend on the selected parameters as we have seen in the previous subsections. Attempting an attack gets very hard without knowledge of the transform domain used for creating the original hash. The underlying assumption of the attack is, that it is operating on a transformed image identical to the one used to calculate the hash value. Only if this is the case, adjusting the forgery's features to match those of the original has the desired effect on the hash value. By using a tranform domain with an incorrect set of parameters, this assumption is weakened. The adjusted forgery's features will only match those of the original for the filter chosen for the attack. This does not necessarily make them more alike for any other filter. Fig. 5 shows the results of the attack using both techniques and various decomposition keys.



(a) Wavelet Packets

(b) Parameterized Filters

Fig. 5. Attack resistance of the key dependency schemes

The Hamming distance for the correct key in the random decomposition case after the attack has been mounted is 0.0166. The average distance after the attack for all random decompositions considered is inceased to 0.0728, however, the large number of "correct" keys (i.e. leading to the same result as the key used to compute the original hash) makes the scheme unreliable (Fig. 5(a)). This corresponds well to the results with respect to key dependency displayed in Fig. 1(b).

Given the key dependency tests (Fig. 2(a)), filter parameterization seemes more promising than random decomposition. Though only a small number of filters renders the attack completely useless, its effects are attenuated considerably, thus improving the scheme's overall security. The average distance of 0.0666 after the attack, compared to 0.0083 for the correct key, is a definite improvement (see Fig. 5(b)). The number of successful attacks (i.e. equally successful as without filter parametrization) is negligible. However, considering the high number of key values with still rather low Hamming distances, the effects of the attack can only said to be weakened to some extent.

4 Conclusion

We have discussed the use of key dependent wavelet transforms as a means to enhance the security of wavelet based hashing schemes. Whereas key dependency and keyspace of the hashing scheme considered in experiments have been significantly improved, the attack resistance has been improved by using parametrized wavelet filters to a small extent only.

References

[1] W. M. Dietl and A. Uhl. Robustness against unauthorized watermark removal attacks via key-dependent wavelet packet subband structures. In *Proceedings of the IEEE International Conference on Multimedia and Expo, ICME '04*, Taipei, Taiwan, June 2004.

[2] Werner Dietl, Peter Meerwald, and Andreas Uhl. Protection of wavelet-based watermarking systems using filter parametrization. *Signal Processing (Special Issue on Security of Data Hiding Technologies)*, 83:2095–2116, 2003.

[3] Jiri Fridrich and Miroslav Goljan. Robust hash functions for digital watermarking. In *Proceedings of the IEEE International Conference on Information Technology: Coding and Computing*, Las Vegas, NV, USA, March 2000.

[4] T. Köckerbauer, M. Kumar, and A. Uhl. Lightweight JPEG 2000 confidentiality for mobile environments. In *Proceedings of the IEEE International Conference on Multimedia and Expo, ICME '04*, Taipei, Taiwan, June 2004.

[5] Albert Meixner and Andreas Uhl. Analysis of a wavelet-based robust hash algorithm. In Edward J. Delp and Ping W. Wong, editors, *Security, Steganography, and Watermarking of Multimedia Contents VI*, volume 5306 of *Proceedings of SPIE*, pages 772–783, San Jose, CA, USA, January 2004. SPIE.

[6] A. Pommer and A. Uhl. Selective encryption of wavelet-packet encoded image data — efficiency and security. *ACM Multimedia Systems (Special issue on Multimedia Security)*, 9(3):279–287, 2003.

[7] R. Radhakrishnan, Z. Xiong, and N. D. Memom. Security of visual hash functions. In Ping Wah Wong and Edward J. Delp, editors, *Proceedings of SPIE, Electronic Imaging, Security and Watermarking of Multimedia Contents V*, volume 5020, Santa Clara, CA, USA, January 2003. SPIE.

[8] J. Schneid and S. Pittner. On the parametrization of the coefficients of dilation equations for compactly supported wavelets. *Computing*, 51:165–173, May 1993.

[9] Ramarathnam Venkatesan, S.-M. Koon, Mariusz H. Jakubowski, and Pierre Moulin. Robust image hashing. In *Proceedings of the IEEE International Conference on Image Processing (ICIP'00)*, Vancouver, Canada, September 2000.