

study the security of randomized image features. They also suggest that the security of a hashing scheme should be measured by the conditional entropy of the hashing key, when the hashing algorithm, and pairs of images and their hashes are known. We formalize this notion and show that such information-theoretic security is impossible.

The robust hash in [3] is specifically designed for an authentication application. There are existing works that analyze security models for authentication application scenarios. Some examples of proposed scenarios are as follows. [4] analyzes a scenario where A sends a message and an approximate message authentication code (AMAC) to B for verification. The AMAC serves as a keyed similarity-preserving function. Another scenario as discussed in [5] is when A sends a message, some helper data and a cryptographic MAC to B , where the noise during communication can be corrected using the helper data. In both cases, A and B share a secret key. If public-private key pairs are used instead of secret keys in these scenarios, it would become a digital signature. Another scenario focusing on key extraction is discussed in [6] where A and B have access to a content X and its corrupted version X' respectively. A extracts a key K_a and some helper data which is sent to B . B uses X' and the helper data to generate another key k_b , and $k_a = k_b$ if X and X' are close enough. In all the above three scenarios there is need for communication between A and B . In the scenario proposed in this paper there is no communication between A and B .

We highlight that the scenario considered in this work represents a more general applicability of robust hash. In fact [7] also proposes a robust hashing algorithm to extract a consistent key from a content but for watermarking application. Note that we also extract a consistent key, which can be used for authentication. Hence it is interesting to study the security of robust hash under the proposed scenario.

3. DEFINITIONS AND NOTATIONS

Let \mathcal{M} be the message space. We do not impose any constraint on \mathcal{M} . For example, it can be the set of all possible feature vectors from all images. The distribution of the message can be either discrete or continuous. Here we mainly use discrete distributions and entropies as examples. Similar definitions and proofs based on differential entropies can be easily adapted. Let D be a distance function $D : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}^+$ defined over \mathcal{M} . We consider the following definitions of hash functions and their properties.

Definition 1 (Hash Function) A hash function \mathcal{H} is a efficiently computable¹ function $\mathcal{H} : \mathcal{M} \times \{0, 1\}^k \rightarrow \{0, 1\}^{poly(k)}$ for some $k \in \mathbb{Z}^+$ and positive polynomial $poly(\cdot)$.

Definition 2 (Robustness) A hash function \mathcal{H} is $(D, \delta_r, \epsilon_r)$ -robust if for any $X, Y \in \mathcal{M}$ such that $D(X, Y) \leq \delta_r$, it holds that $\Pr[\mathcal{H}(X, K) = \mathcal{H}(Y, K)] \geq 1 - \epsilon_r$.

Definition 3 (Sensitivity) A hash function \mathcal{H} is $(D, \delta_s, \epsilon_s)$ -sensitive if for random $K \in \{0, 1\}^k$ and random $X, Y \in \mathcal{M}$ such that $D(X, Y) \geq \delta_s$, we have $\Pr[\mathcal{H}(X, K) \neq \mathcal{H}(Y, K)] \geq 1 - \epsilon_s$.

Sensitivity of a hash function is equivalent to collision resistance for random attackers. Sometimes it is desirable to consider stronger security against a type of preimage attack as below, where a smart attacker tries to find a preimage of a hash value of a given message.

¹Here being ‘‘efficiently computable’’ means that \mathcal{H} can be computed with a polynomial time deterministic algorithm.

Definition 4 (Negligible Function) A function $\sigma : \mathbb{Z} \rightarrow \mathbb{R}^+$ is negligible if for any positive polynomial $poly(\cdot)$ and any sufficiently large $n \in \mathbb{Z}$, it holds that $\sigma(n) < 1/poly(n)$.

When we say a task is computationally infeasible w.r.t. a parameter k , we mean that the probability that this task can be done is a negligible function of k .

Definition 5 (Preimage Resistance) A hash function \mathcal{H} is (D, δ) -preimage-resistant if for random $K \in \{0, 1\}^k$, and a given $X \in \mathcal{M}$, it is computationally infeasible for any polynomial time probabilistic algorithm A to find another Y such that $D(X, Y) > \delta$, yet $\mathcal{H}(X, K) = \mathcal{H}(Y, K)$.

It is often desirable to make it difficult to forge hash values of noisy data without the secret key. Furthermore, we consider chosen message attacks, where the attacker has the access to polynomial number of messages and their corresponding hash values, and the goal of the attacker is to find the hash value for another message that is not similar to any of the messages the attacker has seen. It is clear that security against forgery attacks for a randomly chosen message implies resistance to preimage attacks.

Ideally, we would prefer information-theoretic security where it is possible, which is defined as the following.

Definition 6 (Perfect Forgery Resistance) A hash function \mathcal{H} is information-theoretically secure against forgery under chosen message attacks if for all sufficiently large k , any positive polynomial $p(\cdot)$, any messages $X_1, \dots, X_{p(k)} \in \mathcal{M}$, such that $D(X, X_i) > \delta_c$ for all $1 \leq i \leq p(k)$, and for K uniformly distributed over $\{0, 1\}^k$, it holds that

$$H(b_1 \mid X_1, (X_2, b_2), \dots, (X_{p(k)}, b_{p(k)})) \geq s$$

where $H(A \mid B)$ denotes the conditional entropy of A given B , $b_i \triangleq \mathcal{H}(X_i, K)$, and s is a security parameter.

In other words, give the observations of $p(k) - 1$ dissimilar messages (which could be chosen by the attacker) and their corresponding hash values, there should still be sufficient amount of uncertainty about the hash value of another message X_1 that is dissimilar to all the observations. It should be noted that if a robust hash scheme is secure by the above definition, it would also imply

$$H(K \mid (X_2, b_2), \dots, (X_{p(k)}, b_{p(k)})) \geq s \quad (1)$$

That is, it should also be difficult to find the key K , since otherwise the attacker could just work out the key first and then compute the hash value of X_1 .

Correspondingly, we can define the security in terms of computationally bounded attackers as follows.

Definition 7 (Computational Forgery Resistance) A hash function \mathcal{H} w.r.t. distance function D is computationally secure against forgery under chosen message attacks if, for any positive polynomial $p(\cdot)$, uniformly random $K \in \{0, 1\}^k$, and any poly-time probabilistic algorithm A , given any $X_1 \in \mathcal{M}$, the probability

$$p = \Pr[A(X_1, (X_2, b_2), \dots, (X_{p(k)}, b_{p(k)})) = b_1]$$

is a negligible function of k , where $b_i \triangleq \mathcal{H}(X_i, K)$, $D(X_i, X_j) > \delta_c$ for all $1 \leq i, j \leq p(k)$, and the probability is taken over random choices of the key K , and random decisions made by A .

In other words, it should be computationally infeasible for any attacker to come up with the correct hash value of a new message (which may be chosen by the attacker), even with observations of polynomial number of random message/hash pairs.

4. IMPOSSIBILITY OF INFORMATION-THEORETIC FORGERY RESISTANCE

If the attacker has unbounded computation power and is able to observe previous random message/hash pairs, we can see that it is not possible to have a robust hash scheme that is secure against forgery. In particular, we have the following result.

Theorem 1 *For any hash function \mathcal{H} , given any independently chosen $X_1, \dots, X_{p(k)} \in \mathcal{M}$, any positive polynomial $p(\cdot)$, uniformly chosen key $K \in \{0, 1\}^k$, and any positive constant s ($s \leq k$), if*

$$H(K | X_1, \dots, X_{p(k)}, b_1, \dots, b_{p(k)}) \geq s \quad (2)$$

where $b_i \triangleq \mathcal{H}(X_i, K)$, it holds that

$$H(b_1 | X_1, \dots, X_{p(k)}, b_2, \dots, b_{p(k)}) \leq (k - s)/p(k). \quad (3)$$

Proof: Since the key is independently chosen,

$$\begin{aligned} H(K) &= H(K | X_1, \dots, X_{p(k)}) \\ &= H(K, b_1, \dots, b_{p(k)} | X_1, \dots, X_{p(k)}) \\ &= H(b_1, \dots, b_{p(k)} | X_1, \dots, X_{p(k)}) \\ &\quad + H(K | X_1, \dots, X_{p(k)}, b_1, \dots, b_{p(k)}) \end{aligned} \quad (4)$$

The second equality is due to the fact that the hash values are deterministically computed from the messages and the key, and the third equality is obtained by applying the chain rule. Since $H(K) = k$ and (2), we have

$$H(b_1, \dots, b_{p(k)} | X_1, \dots, X_{p(k)}) \leq k - s. \quad (5)$$

By applying the chain rule, we have

$$\begin{aligned} &H(b_1, \dots, b_{p(k)} | X_1, \dots, X_{p(k)}) \\ &= H(b_1 | b_2, \dots, b_{p(k)}, X_1, \dots, X_{p(k)}) \\ &= H(b_2 | b_3, \dots, b_{p(k)}, X_1, \dots, X_{p(k)}) + \dots \\ &\quad + H(b_{p(k)} | X_1, \dots, X_{p(k)}) \end{aligned} \quad (6)$$

Clearly, the first term in the summation is no larger than the rest of the terms. Hence, we have proved (3). \square

In other words, no matter how we choose the parameters k and s , as long as the attacker has enough number of independent observations (i.e., large enough $p(k)$), the entropy of the hash value $\mathcal{H}(X_1, K)$ can be reduced arbitrarily. Hence the scheme cannot be information-theoretically secure against forgery under chosen message attacks. As mentioned earlier, a similar proof using differential entropy for continuous distributions can be easily adapted.

It is also worth to note that even if the requirement (2) is relaxed such that the key K is only required to have a conditional entropy the same as the hash value, the conclusion that the conditional entropy of the hash value can be reduced arbitrarily remains the same.

It is also noted that this proof does not depend on the robustness of the hash function. It is analogous to the known result that perfect secrecy cannot be achieved with fixed length keys.

5. A COMPUTATIONALLY SECURE SCHEME

In this section, we consider input space \mathcal{M} to be real vectors of size n . This is a natural representation of many types of features that can be extracted from multimedia data.

Let Q be a random composite scalar quantizer, which consists of two sub-quantizers Q_0 and Q_1 . Both sub-quantizers are uniform quantizers with the same step size 2λ , but the quantization levels are interleaved such that the minimum distance between the quantization levels of Q_0 and that of Q_1 is λ . The quantization levels of both sub-quantizers are labeled as interleaving 0's and 1's, and the output of the sub-quantizers will be the label of the nearest quantization level. The quantizer Q takes in two inputs, namely a random bit r and an x to be quantized. Based on the random bit r , Q either quantize x with Q_0 or Q_1 . That is, $Q(r, x) = Q_r(x)$.

We assume that there exists an error-correcting code \mathcal{C} that allows us to correct t errors in a binary string of length n , where t is a parameter depending on the number of bit flips to be tolerated for the required robustness.

We further assume that there exists an encryption scheme $(\mathcal{E}, \mathcal{D})$ with encryption algorithm \mathcal{E} and decryption algorithm \mathcal{D} , and the encryption scheme is secure against chosen ciphertext attacks.

Given $X = (x_1, \dots, x_n)$ and random key $K = (k_1, k_2, k_3)$ consisting of three parts, the function \mathcal{H} performs the following steps.

1. Randomize X by multiplying X with a random n by n matrix R generated using a pseudo-random number generator with seed k_1 . Let $Y = RX = (y_1, y_2, \dots, y_n)$.
2. Suppose $|k_2| = n$. Let the bits in k_2 be denoted as $r_1, \dots, r_n \in \{0, 1\}$. Quantize Y by applying random quantizer Q on each y_i using randomness r_i . Let $\hat{Y} = (\hat{y}_1, \dots, \hat{y}_n)$ where $\hat{y}_i = Q(r_i, y_i)$.
3. Decode \hat{Y} using the error-correcting code \mathcal{C} to obtain a binary string W .
4. Apply the decryption function \mathcal{D} with key k_3 on W to obtain the final hash value h . That is, $h = \mathcal{D}(k_3, W)$.

The first two steps are commonly used techniques to handle permissible noise, such that at the end of the second step, we would obtain a binary string that is close to the original, measured by some distance metric, say, Hamming distance. The error-correcting code in the third step is to correct a small number of bit flips in the binary string. The design of such an error-correcting code can be challenging if the length of \hat{Y} is large, and the errors can be bursty. For typical applications, we can employ the two-layer error-correcting technique as in [8], or a suitable LDPC code [9] such as in [10, 11]. If the distance metric is not Hamming, the error-correcting code should be adapted accordingly (such as in the case of [7]). The last step of decryption provides the actual security property that we need.

5.1. Security Analysis

The security of the hash function as stated at the beginning of this section largely depends on the security of the underlying encryption scheme. In particular, we require the encryption scheme to be secure under chosen ciphertext attacks.

Roughly speaking, an encryption scheme is secure if for any given ciphertext, it is computationally infeasible for any attacker to compute the corresponding plaintext. Furthermore, it is secure against chosen ciphertext attacks if the security can be maintained even the attackers observed *a priori* the plaintext corresponding to polynomial number of ciphertext at his choice.

Theorem 2 *The aforementioned robust hash \mathcal{H} is computationally secure against forgery under chosen message attacks if it is (D, δ) -preimage-resistant, and the encryption function $(\mathcal{E}, \mathcal{D})$ is secure against chosen ciphertext attacks.*

Proof: The proof is a standard reduction proof, which is quite straight forward. We only give the sketch of the proof here.

Suppose on the contrary that the hash function \mathcal{H} is not secure against forgery under chosen message attackers, we can then use the attacker's algorithm A to break the underlying encryption scheme.

The main idea is that, we can randomly select keys k_1 and k_2 , and for any message X chosen by the algorithm A , we can perform steps 1 to 3 to obtain a binary string W , which we then send for decryption as our "chosen ciphertext".

In the end, for the given ciphertext c , we can invert the error-correcting code (by encoding c), the quantization (by finding any data that would be quantized to the encoded c), and the randomization (by multiplying with the inverse of R).

We treat the result as an original message X and pass it to algorithm A and ask A to find a forged hash value, which would be exactly the plaintext corresponding to c . If A is a successful attacker, we would also be successful in attacking the underlying encryption scheme under chosen ciphertext attacks, which contradicts with the assumption. \square

5.2. Further Discussions on Security

It should be noted that in many real applications, we are not given directly a sequence of numbers, but rather images, audio or video data. As a result, the security of the actual system depends not only on the security of the robust hash, but also depends on the *quality* of the features, in the sense that it should be difficult to find another multimedia object that is very different yet yields the same features. However, the similarity metric largely depends on the application scenario. How to select such good features is out of the scope of this paper.

It is also worth noting that we apply an error-correcting code to extract a consistent string from the noisy data. In practice, such codes are usually not perfect (or have large gaps between ϵ_1 and ϵ_2). As a result, to achieve required robustness, sometimes it would correct more errors than necessary. In this case, the attacker might be able to exploit the property of the error-correcting code to create a collision on a chosen message X by constructing dissimilar objects X' that have features that fall within the error-correcting capability, hence creating a forgery on the constructed object X' by re-using the hash value of X . In our scheme, however, we include a randomization step (the first step) that will help to diffuse the feature coefficients in a key-dependent manner, such that it would be difficult for the attackers to make use of the error-correcting code.

6. CONCLUSIONS

A robust hash allows the extraction of a consistent key from noisy data, such as images. This can be useful in many application scenarios, ranging from authentication to session key agreement. An example of such applications is watermarking schemes resistant to copy attacks, where the watermarks are generated from a key extracted from the content, so that directly copying the watermark makes it useless. Another example is key distribution with noisy data, where two or more parties agree on a consistent key based on a common

noisy random source, without the need for communication as other key agreement protocols do.

In this paper we study the security of robust hash functions against forgery, under chosen message attacks. In these attacks, the attacker is allowed to observe or probe the system to access polynomial number of message/hash pairs, and the goal of the attacker is to compute the hash of another given message that is dissimilar to all previous observations.

We give formal definitions of the security of robust hash against forgery under chosen message attacks (both information-theoretic and computational). We show that information-theoretic security is not possible. This answers one of the open questions stated in [3]. That is, it is not possible for the hash value in question to have a conditional entropy that is not negligible, while keeping enough entropy for the secret key. Furthermore, we give a scheme that is computationally secure.

7. REFERENCES

- [1] M. Kutter, S. Voloshynovskiy, and A. Herrigel, "The watermark copy attack," in *Proceedings of the SPIE, Security and Watermarking of Multimedia Contents II*, 2000, vol. 3971.
- [2] S. Craver, N. Memon, B.L. Yeo, and M.M. Yeung, "Resolving rightful ownerships with invisible watermarking techniques: Limitations, attacks, and implications," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 4, pp. 573–586, 1998.
- [3] A. Swaminathan, Y. Mao, and M. Wu, "Robust and secure image hashing," *IEEE Trans. Information Forensics and Security*, vol. 1, no. 2, pp. 215–230, 2006.
- [4] R. Ge, G. R. Arce, and G. DiCrescenzo, "Approximate message authentication codes for n -ary alphabets," *IEEE Trans. on Information Forensics and Security*, vol. 1, no. 1, pp. 56–67, 2006.
- [5] Q. Li and E.-C. Chang, "Robust, short and sensitive authentication tags using secure sketch," in *ACM Multimedia and Security Workshop*, 2006, Short paper.
- [6] Y. Dodis, L. Reyzin, and A. Smith, "Fuzzy extractors: How to generate strong keys from biometrics and other noisy data," in *Eurocrypt*, 2004, vol. 3027 of *LNCS*, pp. 523–540.
- [7] M.K. Mihcak and R. Venkatesan, "A perceptual audio hashing algorithm: A tool for robust audio identification and information hiding," in *Information Hiding Workshop*, 2001, vol. 2137 of *LNCS*, pp. 51–65.
- [8] F. Hao, R. Anderson, and J. Daugman, "Combining crypto with biometrics effectively," *IEEE Trans. Computers*, vol. 55, no. 9, pp. 1081–1088, 2006.
- [9] D.J.C. MacKay and R.M. Neal, "Near shannon limit performance of low density parity check codes," *Electronics Letters*, vol. 33, no. 6, pp. 457–458, 1997.
- [10] E. Martinian, S. Yekhanin, and J.S. Yedidia, "Secure biometrics via syndromes," in *Allerton Conf. on Communications, Control, and Computing*, 2005.
- [11] S.C. Draper, A. Khisti, E. Martinian, A. Vetro, and J.S. Yedidia, "Secure storage of fingerprint biometrics using slepian-wolf codes," in *Information Theory and Applications Workshop*, January 2007.