SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

December 2020, Vol. 63 222301:1–222301:18
https://doi.org/10.1007/s11432-020-3007-5

Efficient and secure image authentication with robustness and versatility

Haixia CHEN, Xinyi HUANG*, Wei WU & Yi MU

Fujian Provincial Key Laboratory of Network Security and Cryptology, Center for Applied Mathematics of Fujian Province, College of Mathematics and Informatics, Fujian Normal University, Fuzhou 350117*, China*

Received 5 June 2020/Revised 1 July 2020/Accepted 2 November 2020/Published online 11 November 2020

Abstract Image authentication is the technology of verifying image origin, integrity and authenticity. A rich stream of research on image authentication has shown various trade-off among four favorable features, namely robustness, security, versatility and efficiency. Image data authentication has the highest level of security but provides no robustness/versatility. Image content authentication from robust hashing keeps robust to limited types of operations, and as a result its versatility is not satisfactory. Existing designs of image content authentication from advanced cryptographic primitives achieve robustness, security and versatility, at the cost of low efficiency. In this paper, we present a new design of image authentication with an improved trade-off among the aforementioned features. Our versatile design is robust to a number of predefined image processing operations. Its security can be reduced to q-strong Diffie-Hellman (q-SDH), a complexity problem used by existing cryptographic algorithms. From the aspect of efficiency, the new design has a constant-size authentication overhead ($\leqslant$ 2 kB) and a constant verification time (around 0.05 s). While the time of generating authentication overhead increases linearly with the number of permissible editing operations, it only takes around 0.33 s for 1000 types of permissible operations. We believe the new design will facilitate image applications where trustworthy image editing is required.

Keywords image authentication, image processing, cryptography, bilinear maps

Citation Chen H X, Huang X Y, Wu W, et al. Efficient and secure image authentication with robustness and versatility. Sci China Inf Sci, 2020, 63(12): 222301, https://doi.org/10.1007/s11432-020-3007-5

1 Introduction

After the mechanical and electronic eras, human society has entered a new era of information technology [1]. The rapid development of information technology facilitates image processing with powerful image-editing software such as the Photoshop[1)]. Hence, anyone who holds a small amount of ability in computer application can edit an image at will, which brings challenges to make sure that digital images are processed sincerely.

Image authentication is the technology of verifying image origin, integrity and authenticity [2]. It is commonly used in image-relevant applications for tampering detection. A subtle issue in image authentication is to distinguish image-data authentication from image-content authentication. Image data includes the exact values of all image bits/pixels, and image content refers to image semantics. Image content distortion will lead to image data modification, but the converse is not necessarily true. For example, if an image is JPEG-compressed (Joint Photographic Experts Group), the value of its pixels is changed but the literal meaning, i.e., the image content, could remain the same.

* Corresponding author (email: xyhuang@fjnu.edu.cn)
1) https://www.adobe.com/products/photoshop.html.

One of the earliest studies [3] realizes image-data authentication with digital signatures [4]. A bit change on the image data (pixel values) will lead to authentication failure. This is guaranteed by the unforgeability of digital signatures. As a result, images authenticated in this way disallow any image processing operations, even these operations are "reasonable" and necessary. In general, image-data authentication has the highest level of security, but provides no robustness. This drawback is alleviated by image-content authentication, the focus of our paper.

It is the goal of image-content authentication to tolerate "reasonable" image processing. Perpetual/Robust hashing [5–7] is one of the most popular image-content authentication techniques. In Perpetual/Robust hashing schemes, ordinary cryptographic hash functions are used to compute the authentication code of an image. Rather than taking the whole image data as the input, the hash is applied to carefully selected features extracted from the image. These features will keep intact under specific operations, e.g., universal JPEG compression [8], sharpening and denoising. Therefore, proper image processing operations would not invalidate the authentication code, and the "robustness" is achieved.

It is also clear that one can circumvent the protection in image-content authentication from Perpetual/Robust hashing, since the two images could differ in contents but have the same feature. For instance, histogram is a common feature in image-content authentication, with the purpose to support rotation transformation on authenticated images. However, a mirrored image has the same histogram as the original one and can also successfully pass the authentication. It would be a security breach if mirror transformation is not considered as a reasonable operation by the image holder. In this case, another image feature, which is sensitive to mirror but robust to rotation, must be selected in image authentication. This uncertainty of security calls for further study on image-content authentication, which can tolerate various permissible image processing with a higher level of security protection.

1.1 Motivations and applications

Let us begin with a specific scenario with three entities.

• **Image owner.** This entity is the image owner/producer who generates an original authentication code of an image. This is an honest entity and defines reasonable permissible operations on the image.

• **Image editor.** This entity could be any untrusty image holder who alters an authenticated image. Given the authenticated image and its image authentication code, this entity edits the image and computes the supplementary proof for the processed image. Producing supplementary proof does not take any interaction with the image owner.

• **Image user.** This entity verifies the validity of an image using the attached authentication code/proof.

As we can see, the aforementioned scenario needs a practical image authentication with four favorable features. (1) Robustness: An authenticated image after reasonable image processing can still convince image users of its validity. (2) Versatility: An image authentication scheme is robust under various kinds of reasonable image processing. (3) Security: An authenticated image after malicious image processing will lead to authentication failure. (4) Efficiency: The overall computation and communication, especially at the stage of image original authentication (which could be carried out by an online device such as a surveillance system), are practical for real-world applications. While there is a rich stream of research on image authentication, existing designs cannot provide a satisfactory trade-off on the aforementioned four features.

1.2 Contributions

We present a new design of image authentication based on advanced cryptographic primitives. Our construction yields an improved trade-off among robustness, versatility, security and efficiency. We give a summary about the comparison of our construction to existing schemes in Table 1 [3, 9–13].

(1) Robustness. In our scheme, a set of processing operations is defined by the image owner for permissible image processing. After a permissible image processing, the image editor can compute a supplementary proof of the altered image. The proof will convince the image user of the image's validity.

Table 1 Comparison of our construction to existing schemes[a)]

Scheme	Primitive	Versatility	Security	Robustness				Efficiency	
			Forgery probability	Enhancement	Geometric transformation	Filter	JPEG head	Overtime	Total
[3]	DSS	None	Negligible	×	×	×	×	$\mathsf{O}(1)$	$\mathsf{O}(1)$
[9]	PHF	Restricted	<10%	Sensitive	Sensitive	Sensitive	√	$\mathsf{O}(1)$	$\mathsf{O}(n)$
[10]	PHF	Restricted	<10%	√	√	√	√	$\mathsf{O}(1)$	$\mathsf{O}(n)$
[11]	DSS+CH	Restricted	Negligible	×	Restricted	×	×	$\mathsf{O}(n)$	$\mathsf{O}(n)$
[12]	DSS+CS	Restricted	Negligible	×	Restricted	×	×	$\mathsf{O}(n)$	$\mathsf{O}(n)$
[13]	DSS+SNARKs	Flexible	Negligible	√	√	√	√	$\mathsf{O}(1)$	$\mathsf{O}(mn)$
Our scheme	Bilinear pairing	Flexible	Negligible	√	√	√	√	$\mathsf{O}(1)$	$\mathsf{O}(m)$

a) DSS: Digital signatures; PHF: Perpetual hash functions; CS: Commitment schemes; CH: Chameleon Hashes; SNARKs [14]: Succinct non-interactive zero-knowledge proofs; n: Number of image pixels; m: Number of permissible operations. Particularly, $m = 0$ in [3] and $m = 1$ in [11, 12].

Table 2 Efficiency of our scheme with 80-bit security[a)]

Image size (pixels)	Num	Time (s)		Memory (kB)	
		AC+SP Computation	Verification	Key size	Overhead
256×256	200	≈ 0.14	≈ 0.05	$\leqslant 6$	$\leqslant 2$
256×256	500	≈ 0.24	≈ 0.05	$\leqslant 15$	$\leqslant 2$
512×512	1000	≈ 0.33	≈ 0.05	$\leqslant 30$	$\leqslant 2$

a) Num: number of permissible image operations; AC: the authentication code; SP: the supplementary proof.

(2) Versatility. Our scheme is a versatile scheme which supports a large number of image processing operations which can be defined with functions. It is compatible with not only image-content authentication but also image-data authentication. In our scheme, versatility is efficiently achieved by a succinct and constant-size cryptographic proof for all permissible operations defined by the image owner.

(3) Security. Image editor could alter images in a malicious way. However, it is infeasible to compute the supplementary proof. As a result, images altered maliciously would not convince any image user. In our scheme, this is guaranteed by a rigorous analysis under the q-strong Diffie-Hellman (q-SDH) assumption.

(4) Efficiency. Our scheme outputs a constant-size original authentication code (AC) and the supplementary proof (SP), while the time of AC/SP generation increases linearly with the number of permissible editing operations (but still efficient). The verification time is also a constant, and independent of permissible editing operations or image size. Evaluation is done with an ordinary desktop of a 2-core 3.4 GHz Intel i5-7500 processor and 8 G of RAM. Table 2 gives a brief summary, and a detailed analysis is given in Subsection 6.2.

1.3 Organization

The rest of the paper is organized as follows. We describe some related work in Section 2. In Section 3, we present the preliminaries required in this paper. Section 4 is devoted to formal definitions of an efficient and secure image authentication scheme. In Section 5 we give the construction of our scheme and prove its security. We evaluate the new scheme in Section 6 and conclude this paper in Section 7.

2 Related work

2.1 Image authentication from cryptographic hash

Collision-resistant (cryptographic) hash is the cornerstone of image authentication. Informally, a cryptographic hash function H is collision-resistant if it is computationally infeasible for any probabilistic polynomial-time (PPT) algorithm to find a collision of H [15].

One of the earliest study in image authentication was proposed by Friedman [3] in 1993. In [3], the image authentication code is a digital signature [16] on the hash of the whole image. The drawback of this scheme is that even a bit change in any pixel of the authenticated image will invalidate the authentication code. To accomplish more versatile image authentication, image features are hashed instead of the whole image data. This method is introduced in [5] and named as perceptual/robust-hash (PHF). In perceptual-hash-based image authentication, image features (such as mean, variance, and histogram) are extracted and hashed before the computation of image authentication code.

In perceptual-hash-based image authentication, static image features are usually chosen to ensure the robustness of authentication. Xie et al. [17] introduced the notion of approximate image message authentication codes (IMACs) from the means of image blocks. The evaluation in [17] shows that the proposed scheme can tolerate small/moderate image compression and detect/locate tampering. Monga and Evans [6] proposed an image hashing paradigm using visually significant feature points, which are largely invariant under perceptually insignificant distortions. Tang et al. [18] proposed an image hashing scheme by dividing the original image into different rings and computing hash value according to ring-based entropies. All aforementioned schemes can withstand standard benchmark image processing operations such as compression and small-angle rotation, but are sensitive to content changing manipulations.

There are also situations in which reasonable image content changes are allowed. Histogram, another kind of image feature, is used in perpetual-hash-based image authentication schemes to tolerate reasonable content changing manipulations such as rotation and pixel shuffling. Xiang et al. [19] proposed a scheme by using the invariance of the image histogram shape to tolerate geometric distortions. Choi and Park [20] then improved the robustness of [19] by generating adaptive hash string according to the length of histogram bin.

However, image authentication based on image features, such as histogram, has a potential risk. Adversaries are able to change image content and convey wrong information, while the histogram is the same. To further enhance the security, histogram-based image hashing algorithms with dual perceptual image hash functions are proposed by Tang et al. [21] and Gharde et al. [22]. Nonetheless, these improvements can alleviate but not eliminate the aforementioned attacks.

Other features are also used in image authentication to tolerate various image processing operations. For instance, in JPEG compression, the relationship between discrete cosine transform (DCT) coefficients in the same position of different image blocks remains the same, and even the image is compressed with different ratios. Such a property can be used to design image authentication schemes tolerating JPEG compression. In this line of research, a number of attractive studies have been proposed to support JPEG compression [9,23–27], filtering [28,29], demosaicking [30], rotation [10,31] and geometric transformations [32,33].

To summarize, existing designs of hash-based image authentication have at least one of the following two drawbacks. (1) Versatility is not satisfactory: Swiss-Knife image authentication robust to a wide range of image processing operations is certainly desirable, but most existing designs are only designed for very limited types of operations. As a result, an image authentication scheme for compression may not support geometric transformations. (2) Tradeoff between robustness and security is not optimal. Image data authentication provides a very high level of security protection but does not allow any bit change. With perpetual-hash approach, image-content authentication will keep robust to certain image processing operations, but one can also make the use of this robustness to alter images maliciously without being detected.

2.2 Image authentication from advanced cryptographic primitives

In this subsection, we shall give a brief review on image authentication from advanced cryptographic primitives, including zero-knowledge proof [13], trapdoor/chameleon hashes [11], and commitments [12].

In 2016, Naveh and Tromer [13] proposed an image authentication scheme supporting permissible operations. In their scheme, specific codes called "photoproofs" are generated using zero-knowledge proof [34] in cryptography. These codes are attached to modified images to authenticate corresponding

image operations. The most attractive advantage of their scheme is that it not only provides robustness to multiple image processing operations, but also achieves a high level of security. Although time costs are independent of which operation is implemented on the image, they increase dramatically with the image size. In addition, since each editing must be verified with an additional public key, the size of public keys increases linearly with the number of image processing operations. For example, proving of a 128×128 image needs a time of 306 s, and its verification needs a 2.6 GB public key [13]. In 2017, Kim et al. [11] proposed a privacy-aware image authentication scheme using a chameleon hash [35]. The proposed scheme allows users to delete objects from an authenticated image and provides more efficient verification. However, their scheme needs to sign hash value for every data block separately. As a result, the signature size is rather long. Furthermore, there exist potential security risks since the key exposure issues in chameleon hashes [36–38] are not addressed in the proposed scheme. In 2018, Chen et al. [12] proposed an image authentication scheme for permissible cropping, using a digital signature scheme together with a commitment scheme. The security of this scheme is guaranteed by the unforgeability of digital signatures and the binding property of commitment schemes. However, the scheme also requires a long signature size, because the signature contains the commitments of all pixels.

Overall, compared to hash-based image authentication schemes, the aforementioned schemes have a higher level of security, more satisfactory robustness and versatility, at the cost of low efficiency.

3 Preliminaries

From the discussion above, existing image authentication schemes cannot provide an efficient solution for image authentication achieving both security and robustness. We will investigate a new image authentication scheme which produces an obvious tradeoff among security, robustness and efficiency. In this section, we will give a brief introduction on the preliminaries required in this paper, including the notations and cryptographic primitives.

3.1 Notations

3.1.1 *Negligible function*

A negligible function is asymptotically smaller than any inverse polynomial function.

Definition 1 (Negligible function [39])**.** A function f from the natural numbers to the non-negative real number is negligible, if for every positive polynomial p there is an N such that for all integer $n \leqslant N$ it holds that $f(n) \leqslant \frac{1}{p(n)}$.

We denote this negligible function by $\mathsf{negl}(N)$.

3.1.2 *Probabilistic polynomial-time (PPT)*

Definition 2 (Probabilistic polynomial-time (PPT))**.** A probabilistic polynomial-time algorithm is a randomized algorithm whose running time grows as a polynomial function of the size of its input.

3.1.3 *Computational security*

For practical purpose, computational security is used in modern cryptography. This security definition takes into account computation limits on the attacker, and allows for a small and negligible probability of failure. A definition of computational security is formally defined as follows.

Definition 3 (Computational security [40])**.** A scheme is computational secure if for every PPT adversary carrying out an attack of some formally specified type, there exists an integer N suth that the probability of the adversary's success in its attack is not larger than $\mathsf{negl}(N)$.

Table 3 Image processing operations

Operations (f_i)	Parameter (aux_i)	Parameter description	Editing tool
Sharpening	{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}	Pixels	Photoshop
Blurring	{1, 2, 3, 4, 5}	Pixels	Photoshop
Brightness adjustment	{±30, ±20, ±10}	Scale	Photoshop
Contrast adjustment	{±50, ±30, ±20}	Scale	Photoshop
Rotation	{±90°}	Rotation angle	Photoshop
Mirror	Horizontal, Vertical	Mirror axis	Photoshop
Scaling	{0.5, 2, 4}	Scaling values	Photoshop
Cropping	(x_1, y_1, x_2, y_2)	Rectangle cropping	Photoshop
Median filtering	{3, 5, 7, 9}	Window size of filter	OpenCV
JPEG compression	{10, 20, 30, 40, 50, 60, 70, 80, 90, 100}	Quality factor	OpenCV

3.2 Bilinear pairing and q-SDH assumption

The verification of our scheme requires a bilinear pairing. We give a brief review of bilinear pairings and pairing groups by following the standard notation in [41].

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic groups generated by g_1 and g_2 respectively, whose orders are primer p. Let $\mathbb{G}_T$ be a cyclic multiplicative group with the same primer order p, and let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a bilinear pairing with the following properties.

- Bilinearity: $e(aX, bY) = e(X, Y)^{ab}$ for all $X \in \mathbb{G}_1, Y \in \mathbb{G}_2$, and $a, b \in \mathbb{Z}_p$.
- Non-degeneracy: $e(g_1, g_2) \neq 1$.
- Computability: There is an efficient algorithm to compute $e(X, Y)$ for all $X \in \mathbb{G}_1$ and $Y \in \mathbb{G}_2$.

Here, if $\mathbb{G}_1 = \mathbb{G}_2$, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is symmetric pairing, and otherwise is asymmetric pairing. The proof of security requires an efficiently computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ with $\psi(g_2) = g_1$.

The q-SDH assumption comes from an assumption introduced by Mitsunari et al. [42] to construct traitor tracing schemes before being well stated by Boneh and Boyen [43].

Definition 4 (q-SDH assumption)**.** Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic groups of primer order p with length λ, where possibly $\mathbb{G}_1 = \mathbb{G}_2$. Let g_1 be a generator of $\mathbb{G}_1$, g_2 be a generator of $\mathbb{G}_2$ and $g_1 = \psi(g_2)$. For a randomness s chosen from $\mathbb{Z}_p^*$, an integer $x \in \mathbb{Z}_p$, $q > 0$ and every PPT algorithm $\mathcal{A}$, it holds that

$$\Pr\left[\mathcal{A}(g_1, g_2, g_2^s, \ldots, g^{s^q}) \to \left(x, g_1^{\frac{1}{x+s}}\right)\right] \leqslant \mathsf{negl}(\lambda). \tag{1}$$

4 Definitions

4.1 Definition of image editing

Most existing image authentication schemes can tolerate specific image editing (attacks) constricted by perpetual-based image quality [44, 45], providing image content authentication. In our scheme, more kinds of image operations are considered to provide both content and data authentication, and as a result, versatility is achieved. Image editing operations in our paper refer to operations using image processing tools, such as the Adobe PhotoShop and OpenCV libraries[2)], to edit images.

To simplify our denotation, let $M_i \leftarrow f_i(M, \text{aux}_i)$ denote each operation. Here, M is the original image, f_i is a specific operation (e.g., JPEG compression), aux_i is an auxiliary instruction on processing parameters and values (e.g., quality factor: 60), and M_i is a new (processed) image. Our scheme supports a number of image editing operations, and some of which are given in Table 3. We denote these operations by an ordered set $\mathcal{F} = \{(f_1, \text{aux}_1), (f_2, \text{aux}_2), \ldots, (f_l, \text{aux}_l)\}$ where l is the number of set members.

2) https://opencv.org.

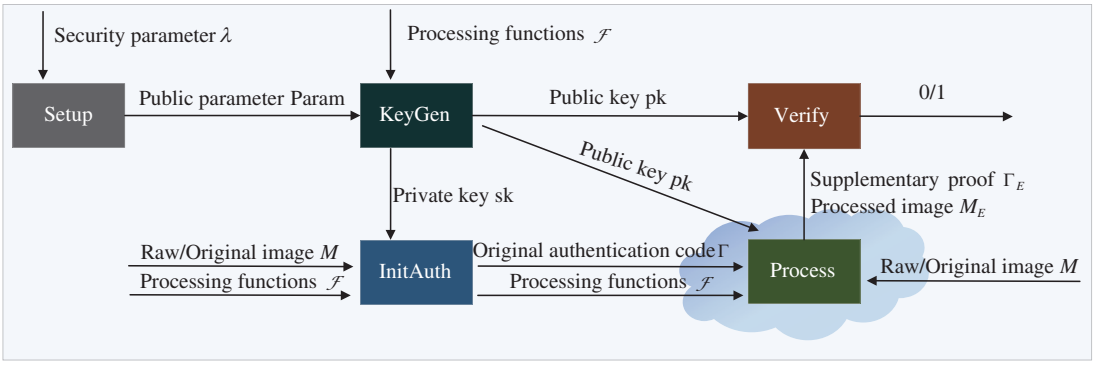

Figure 1 (Color online) The work-flow of our scheme.

4.2 Definition of our scheme

In this subsection, we introduce the formal definition of our scheme. In the proposed scheme, the image producer can allow anyone to edit image in a controllable way. Our scheme involves three entities mentioned in the system description (defined in Subsection 1.1): the image owner, the image editor, and the image user.

Our scheme consists of the following five polynomial-time algorithms: Setup, KeyGen, InitAuth, Process, and Verify. The work-flow of the scheme is presented in Figure 1.

• Setup: The setup algorithm Setup takes a security parameter λ as input. It outputs a public parameter Param. That is $\mathsf{Param} \leftarrow \mathsf{Setup}(1^\lambda)$. This algorithm is run by the image owner.

• KeyGen: The key generation algorithm KeyGen takes as input a public parameter Param and an integer n. It outputs a key pair $(\mathrm{pk}_{\mathsf{Auth}}, \mathrm{sk}_{\mathsf{Auth}})$ for the authenticator. That is $(\mathrm{sk}_{\mathsf{Auth}}, \mathrm{pk}_{\mathsf{Auth}}) \leftarrow \mathsf{KeyGen}(\mathsf{Param}, n)$. This algorithm is still run by the image owner.

• InitAuth: The authentication algorithm InitAuth takes as input the private key $\mathrm{sk}_{\mathsf{Auth}}$, an image M, and an ordered set $\mathcal{F}$. It outputs an authentication code Γ. That is $\Gamma \leftarrow \mathsf{InitAuth}(\mathrm{sk}_{\mathsf{Auth}}, M, \mathcal{F})$. Here, $\mathcal{F}$ contains image processing functions which are allowed to be implemented by users, and the up bound of $\mathcal{F}$ is n. That is, $|\mathcal{F}| \leqslant n$. This algorithm is also run by the image owner.

• Process: The processing algorithm Process takes as input the public key $\mathrm{pk}_{\mathsf{Auth}}$, an image M and an authentication code Γ. It outputs an image M_E together with a new authentication code Γ_E which contains an original AC and an SP. That is $(M_E, \Gamma_E) \leftarrow \mathsf{Process}(\mathrm{pk}_{\mathsf{Auth}}, M, \Gamma)$. This algorithm is run by the image editor.

• Verify: The verification algorithm Verify takes the public key $\mathrm{pk}_{\mathsf{Auth}}$, an image M' and an authentication code Γ'_E as input. It outputs a bit $b \in \{0, 1\}$ to verify the authenticity of the image. That is $b \leftarrow \mathsf{Verify}(\mathrm{pk}_{\mathsf{Auth}}, M', \Gamma'_E)$. This algorithm is run by the verifier.

Correctness of our scheme. For design, we require the correctness property must hold, i.e., for any public parameter reasonably chosen by Setup, any key correctly generated by the KeyGen, any authentication code Γ generated by the InitAuth, and any supplementary proof generated by Process should be accepted by the Verify algorithm. That is

$$\begin{aligned} &\forall\ \mathsf{Param} \leftarrow \mathsf{Setup}(1^\lambda) \wedge (\mathrm{sk}_{\mathsf{Auth}}, \mathrm{pk}_{\mathsf{Auth}}) \leftarrow \mathsf{KeyGen}(\mathsf{Param}, n) \\ &\wedge \forall\ \Gamma \leftarrow \mathsf{InitAuth}(\mathrm{sk}_{\mathsf{Auth}}, M, \mathcal{F}) \wedge\ \forall\ (M_E, \Gamma_E) \leftarrow \mathsf{Process}(\mathrm{pk}_{\mathsf{Auth}}, M, \Gamma) \\ &\Rightarrow \mathsf{Verify}(\mathrm{pk}_{\mathsf{Auth}}, M_E, \Gamma_E) = 1. \end{aligned} \tag{2}$$

4.3 Definition of security model

Given the editing operations specified by a set $\mathcal{F}$ and an authenticated image M provided by the original image authenticator, an image holder could attempt to produce a valid authentication code for an illegitimate image M^* edited by operations not in $\mathcal{F}$, or a totally new image.

We follow the standard way to define the security of our scheme, using a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. The challenger $\mathcal{C}$ creates the scheme, generates the authenticator's private key and answers $\mathcal{A}$'s queries. $\mathcal{A}$ aims to break the scheme by producing a valid authentication code for an illegitimate image.

Setup. $\mathcal{C}$ runs the Setup algorithm to generate the public parameter Param, runs the KeyGen algorithm to generate a key pair $(\mathrm{sk}_{\mathsf{Auth}}, \mathrm{pk}_{\mathsf{Auth}})$, and then sends the public parameter Param to $\mathcal{A}$ together with the public key pk_{Auth}.

Authentication query. $\mathcal{C}$ responds to the authenticating queries made by $\mathcal{A}$ as follows. $\mathcal{A}$ adaptively chooses a sequence of pairs $Q = \{(M_i, \mathcal{F}_i)\}_{1 \leqslant i \leqslant q_s}$ to obtain a set of authentication codes $T = \{\Gamma_i\}_{1 \leqslant i \leqslant q_s}$ for all the queries in Q.

Output. After receiving $T = \{\Gamma_i\}_{1 \leqslant i \leqslant q_s}$, the adversary $\mathcal{A}$ generates an authentication code Γ_E^* for an image M_E^*.

We say $\mathcal{A}$ succeeds in the above game when for all $i \in [1, q_s]$ and $i_j \in [1, |\mathcal{F}_i|]$ it satisfies that

$$\mathsf{Verify}(\mathrm{pk}_{\mathsf{Auth}}, M_E^*, \Gamma_E^*) = 1 \wedge ((M_E^* \neq M_i) \wedge (\forall f_{i_j} \in \mathcal{F}_i, M_E^* \neq f_{i_j}(M_i, \mathrm{aux}_i))). \tag{3}$$

Definition 5 (Editing-unforgeability)**.** Our scheme is unforgeable if every PPT adversary has a negligible success probability in the aforementioned game.

5 The construction

In this section, we present the technological details of our scheme which is built from the bilinear pairing (defined in Subsection 3.2) and the cryptographic tools introduced in [43, 46, 47].

The layout of our construction is as follows.

• **Algorithm 1** Setup: This algorithm takes as input a security parameter λ. It outputs the public parameter Param.

Algorithm 1 Setup: the parameter initialization algorithm

Require: A security parameter λ;
Ensure: The public parameter Param;
1: Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be cyclic groups with prime order p and e denote the bilinear pairing $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$;
2: Randomly pick a generator $g_2 \in \mathbb{G}_2$, and set $g_1 = \psi(g_2)$;
3: Pick two hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}_p$ and $H_2 : \mathbb{G}_1 \to \mathbb{Z}_p$;
4: Set $\mathsf{Param} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, p, e, H_1, H_2)$;
5: **return** Param;

• **Algorithm 2** KeyGen: This algorithm takes as input the public parameter Param and an integer n. It computes the public key $\mathrm{pk}_{\mathsf{Auth}}$ and private key $\mathrm{sk}_{\mathsf{Auth}}$.

Algorithm 2 KeyGen: the key generation algorithm

Require: The public parameter Param, and an integer n;
Ensure: A key pair $(\mathrm{sk}_{\mathsf{Auth}}, \mathrm{pk}_{\mathsf{Auth}})$;
1: Randomly choose $\alpha \in \mathbb{Z}_p^*$;
2: **for all** $i = 0$ to $n-1$ **do**
3: Compute $y_i = g_1^{\alpha^i}$;
4: **end for**
5: Randomly choose $\beta \in \mathbb{Z}_p^*$, and compute $u = g_2^{\beta}$;
6: Set $\mathrm{sk}_{\mathsf{Auth}} = (\alpha, \beta)$ and $\mathrm{pk}_{\mathsf{Auth}} = (\{y_i\}_{0 \leqslant i \leqslant n-1}, u, g_2^{\alpha})$;
7: **return** $(\mathrm{sk}_{\mathsf{Auth}}, \mathrm{pk}_{\mathsf{Auth}})$;

• **Algorithm 3** InitAuth: On input the private key $\mathrm{sk}_{\mathsf{Auth}}$, the original image M, and a set $\mathcal{F}$, it outputs an original authentication code Γ for M. Here, $\mathcal{F}$ is a set of processing functions with l ($l \leqslant n$) members denoted by $\mathcal{F} = \{(f_1, \mathrm{aux}_1), (f_2, \mathrm{aux}_2), \ldots, (f_i, \mathrm{aux}_i), \ldots, (f_l, \mathrm{aux}_l)\}$. As defined in Subsection 4.1, f_i is

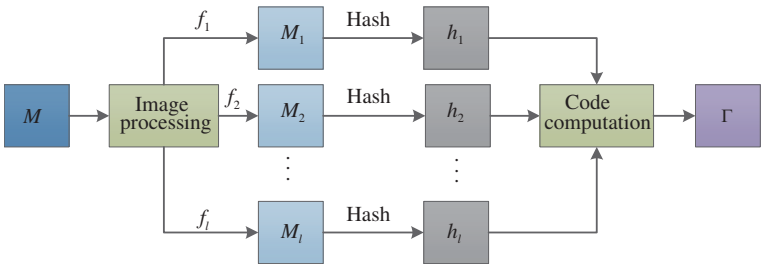

Figure 2 (Color online) The workflow of authenticating an image.

an image processing function with input of an image M and an auxiliary parameter aux_i.

Algorithm 3 InitAuth: the authentication algorithm

Require: The private key sk_{Auth}, the original image M, and a set of processing functions $\mathcal{F}$;
Ensure: The original authentication code Γ;
1: Take out α and β from sk_{Auth};
2: $l = |\mathcal{F}|$;
3: **for all** $i = 1$ to l **do**
4: Compute $M_i = f_i(M, \text{aux}_i)$ and $h_i = H_1(M_i)$;
5: **end for**
6: Draw a random $r \in \mathbb{Z}_p^*$, compute $\delta = g_1^{r \cdot \prod_{i=1}^{l}(h_i+\alpha)}$, $\pi = g_1^{1/(\beta+H_2(\delta))}$, and $\Gamma = (\delta, r, \pi, \mathcal{F})$;
7: **return** Γ;

Notice that, in the authentication phase, the image owner defines the set $\mathcal{F}$ of processing methods which are allowed to be implemented by a subsequent processor. The workflow of the authentication algorithm is presented in Figure 2 where each M_i is the data of a (permissibly) processed image.

• **Algorithm 4** Process: The processing algorithm Process is used to edit an authenticated image by any image holder. On input an image M, the public key pk_{Auth}, and an original authentication code Γ, it outputs a processed image M_E together a new authentication code Γ_E. The workflow of the Process algorithm is presented in Figure 3.

Algorithm 4 Process: the processing algorithm

Require:
The public key pk_{Auth};
The original image M;
The original authentication code Γ;
Ensure:
The processed image-code pair (M_E, Γ_E);
1: Take out $\delta, r, \pi, \mathcal{F}$ from Γ;
2: $l = |\mathcal{F}|$;
3: **for all** $i = 1$ to l **do**
4: Compute $M_i = f_i(M, \text{aux}_i)$ and $h_i = H_1(M_i)$;
5: **end for**
6: Choose a processing function f_k from $\mathcal{F}$;
7: Compute $M_E = f_k(M, \text{aux}_k)$, $\pi_E = g_1^{r \prod_{j=1, j\neq k}^{l}(h_j+\alpha)}$, and $\Gamma_E = (\delta, \pi, \pi_E)$;
8: **return** (M_E, Γ_E);

In Algorithm 4, π_E is computed with the public key but not the private key α. We implement the computation of π_E as follows.

Let $F_i(x)$ be the polynomial $F_i(x) = \Pi_{j=1, j\neq i}^{l}(h_j + x) = \Sigma_{i=0}^{l-1} a_i x^j$. Here, $a_i \in \mathbb{Z}_p$ for $i = 0, \ldots, l-1$ are the coefficients of the polynomial $F_i(x)$. Therefore, π_E can be computed using the public key as follows:

$$\pi_E = \left(\Pi_{i=0}^{l-1} y_i^{a_i}\right)^r = g_1^{r \cdot \Sigma_{i=0}^{l-1} a_i \alpha^i} = g_1^{r \cdot F_i(\alpha)} = g_1^{r \Pi_{j=1, j\neq i}^{l}(h_j+\alpha)}. \tag{4}$$

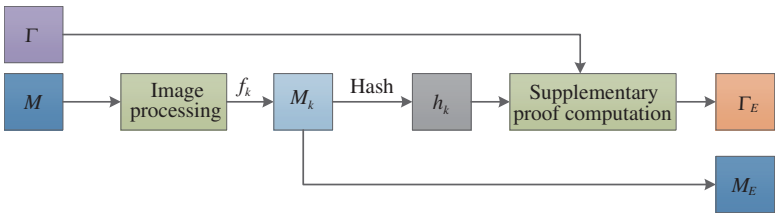

Figure 3 (Color online) The workflow of image processing.

Notice that, to support the editing of verifiable images, the computation of tag π_E needs to know all hash values of images allowed to be edited. While this would introduce extra computation load, the image editor can obtain a succinct and constant-size tag π_E, which shall improve the efficiency of follow-up verification and reduce communication costs. This is the reason why we do not use hash-based methods, such as the Merkle Tree [4], to compute the new authentication tag.

- **Algorithm 5** Verify: On input an image M', a public key $\mathrm{pk}_{\mathsf{Auth}}$ and an authentication code Γ'_E, the algorithm Verify outputs a bit $b \in \{0,1\}$.

Algorithm 5 Verify: the verification algorithm

Require: The public key $\mathrm{pk}_{\mathsf{Auth}}$, the image M', and the authentication code Γ'_E;
Ensure: A bit $b \in \{0,1\}$;
1: Take out δ', π' and π'_E from Γ'_E, and take out g_2^{α} from $\mathrm{pk}_{\mathsf{Auth}}$;
2: Compute $h = H_2(\delta')$ and $v = ug_2^h$;
3: **if** $e(\pi', v) \neq e(g_1, g_2)$ **then**
4: $\quad b = 0$;
5: $\quad$ **return** b;
6: **end if**
7: Compute $h_E = H_1(M_E)$;
8: **if** $e(\delta', g_2) = e(\pi'_E, g_2^{h_E} g_2^{\alpha})$ **then**
9: $\quad b = 1$;
10: **else**
11: $\quad b = 0$;
12: **end if**
13: **return** b;

5.1 Correctness

In the scheme, if the public parameter is correctly set by the Setup, the keys are correctly generated by the KeyGen and authentication codes/proofs are correctly computed by the InitAuth and Process, we hold that

$$\begin{aligned} e(\delta, g_2) &= e\left(g_1^{r\cdot\prod_{i=1}^{l}(h_i+\alpha)}, g_2\right) = e\left(g_1^{r\prod_{j=1,j\neq k}^{l}(h_j+\alpha)}, g_2^{h_k+\alpha}\right) = e\left(g_1^{r\prod_{j=1,j\neq k}^{l}(h_j+\alpha)}, g_2^{H_1(f_k(M,\mathrm{aux}_k))+\alpha}\right) \\ &= e\left(g_1^{r\prod_{j=1,j\neq k}^{l}(h_j+\alpha)}, g_2^{H_1(M_E)+\alpha}\right) = e\left(\pi_E, g_2^{h_E} g_2^{\alpha}\right) \end{aligned} \tag{5}$$

and

$$e(\pi, ug_2^{H_2(\delta)}) = e(\pi, g_2^{\beta}\cdot g_2^{H_2(\delta)}) = e\left(g_1^{1/(\beta+H_2(\delta))}, g_2^{(\beta+H_2(\delta))}\right) = e(g_1, g_2). \tag{6}$$

That is,

$$\mathsf{Verify}(\mathrm{pk}_{\mathsf{Auth}}, M_E, \Gamma_E) = 1. \tag{7}$$

Therefore, the authentication codes are accepted by the Verify algorithm. This completes the correctness analysis.

5.2 Security proof

In this subsection, we prove the security of our proposed scheme under the q-SDH assumption defined in Section 3. The approach used in our poof is security reduction [48], a standard method in the security analysis of cryptographic algorithms.

Theorem 1. Our scheme is editing-unforgeable in the random oracle model, if q-SDH assumption holds.

Proof. Suppose the adversary $\mathcal{A}$ who can win the game defined in Subsection 4.3 by making q_H queries to the random oracle model. We construct a simulator $\mathcal{B}$ to solve the q-SDH problem.

Given a q-SDH instance $(g_1', g_2', (g_2')^s, \ldots, (g_2')^{s^q})$ with some unknown $s \in \mathbb{Z}_p$ where $\mathbb{G}_1$ and $\mathbb{G}_2$ are two cyclic groups of primer order p with length λ. Here, $g_2' \in \mathbb{G}_2$, $g_1' \in \mathbb{G}_1$, and there exists an efficiently computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ with $\psi(g_2') = g_1'$. In addition, the proof requires that $q = \max\{q_H, n\}$ where n is the input of KeyGen algorithm in our scheme. The simulator $\mathcal{B}$ controls the random oracle H_2, runs $\mathcal{A}$ and aims to output a pair $(x, (g_1')^{\frac{1}{x+s}})$ for an integer x where $x \in \mathbb{Z}_p$. To simplify our description, we denote $(g_2')^{s^i}$ by Y_i for $i = 0, \ldots, q$ in the following discussion.

Suppose that $\mathcal{A}$ has forged a valid authentication code $\Gamma_E^* = (\delta^*, \pi^*, \pi_E^*)$ for an image M^*. Let $Q = \{(M_i, \mathcal{F}_i)\}_{1 \leqslant i \leqslant q_H}$ be a sequence of pairs of $\mathcal{A}$'s queries to the authenticating oracle $\mathcal{O}_{\mathsf{InitAuth}}(\mathrm{sk}_{\mathsf{Auth}}, \cdot, \cdot)$ where $|\mathcal{F}_i| = l \leqslant n$ for all $i = 1, \ldots, q_H$. With the output $\{\Gamma_i = (\delta_i, r_i, \pi_i, \mathcal{F}_i)\}_{1 \leqslant i \leqslant q_H}$ of the oracle, $\mathcal{A}$ outputs a forgery $\Gamma_E^* = (\delta^*, \pi^*, \pi_E^*)$. According to the value of δ^* in Γ_E^*, we divide the output instance into the following two cases.

Case 1. $\forall i \in [1, q]$, $\delta^* \neq \delta_i$.

Case 2. $\exists i \in [1, q]$, $\delta^* = \delta_i$.

We will discuss the simulation of the two cases as follows.

Case 1.

Setup. $\mathcal{B}$ randomly chooses $\omega_1, \omega_2, \ldots, \omega_{q_H}$ from $\mathbb{Z}_p^*$. Let $G(x)$ be the polynomial $G(x) = \Pi_{i=1}^{q_H}(\omega_i + x)$. Expand $G(x)$ and write $G(x) = \Sigma_{i=0}^{q_H} b_i x^i$. Here, $b_i \in \mathbb{Z}_p$ for $i = 0, \ldots, q_H$ are the coefficients of the polynomial $G(x)$.

Firstly, $\mathcal{B}$ sets $\beta = s$ and computes

$$g_2 = (g_2')^{G(s)} = (g_2')^{\Pi_{i=1}^{q_H}(\omega_i + s)} = (g_2')^{\Sigma_{i=0}^{q_H} b_i x^i} = \prod_{i=0}^{q_H} (Y_i)^{b_i} \tag{8}$$

and

$$u = g_2^{\beta} = g_2^s = (g_2')^{s \cdot G(s)} = \prod_{i=1}^{q_H} (Y_{i-1})^{b_i}. \tag{9}$$

Then, $\mathcal{B}$ sets $g_1 = \psi(g_2)$, randomly chooses $\alpha \in \mathbb{Z}_p$ and computes $y_i = g_1^{\alpha^i}$ for all $i \in [0, l-1]$.

Lastly, $\mathcal{B}$ chooses $H_1 : \{0,1\}^* \to \mathbb{Z}_p$ and sends the public parameter $\mathsf{Param} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, H_1, p, g_1, g_2)$ together with the public key $\mathrm{pk}_{\mathsf{Auth}} = (\{y_i\}_{0 \leqslant i \leqslant l-1}, u, g_2^{\alpha})$ to $\mathcal{A}$.

H_2-Query. At the very beginning, $\mathcal{B}$ creates an empty hash list and randomly chooses an $i^* \in [1, q_H]$ and an integer $\omega^* \in \mathbb{Z}_p$.

Let the i-th hash query be δ_i, and $\mathcal{B}$ sets the hash values as follows:

$$H_2(\delta_i) = \begin{cases} \omega^*, & i = i^*, \\ \omega_i, & i \neq i^*. \end{cases} \tag{10}$$

$\mathcal{B}$ responds with $H_2(\delta_i)$ and adds (δ_i, ω_i) or (δ_i, ω^*) to the hash list.

Authentication query. $\mathcal{B}$ responds $\mathcal{A}$'s authentication queries in this phase. For any query, if $(M_i, \mathcal{F}_i)$ is the i^*-th query image in the hash list, abort. Otherwise, $\mathcal{B}$ randomly chooses $r_i \in \mathbb{Z}_p^*$ and computes the authentication code $\Gamma_i = (\delta_i, r_i, \pi_i, \mathcal{F}_i)$ using the hash list, the problem instance and $\omega_1, \omega_2, \ldots, \omega_{q_H}$ together with corresponding keys.

Firstly, $\mathcal{B}$ computes

$$\delta_i = g_1^{r_i \Pi_{j=1}^{l} (h_{i_j} + \alpha)} \in \mathbb{G}_1, \tag{11}$$

where $h_{i_j} = H_1(f_{i_j}(M_i, \mathrm{aux}_{i_j}))$ for all $(f_{i_j}, \mathrm{aux}_{i_j}) \in \mathcal{F}_i$ and $j \in [1, l]$. Notice that, α is a randomness chosen by $\mathcal{B}$.

To obtain the hash value of δ_i, $\mathcal{B}$ checks the hash list. If δ_i is in the hash list, $\mathcal{B}$ takes out its hash value ω_i. Otherwise, $\mathcal{B}$ queries the H_2 oracle again with δ_i as input to get ω_i and adds (δ_i, ω_i) to the hash list.

Let $G_i(x)$ be the polynomial $G_i(x) = \Pi_{j=1 \wedge j \neq i}^{q_H} (\omega_j + x)$. Expand $G_i(x)$ and write $G_i(x) = \Sigma_{j=0}^{q_H - 1} c_j x^j$. Here, $c_j \in \mathbb{Z}_p$ for $j = 0, \ldots, q_H - 1$ are the coefficients of the polynomial $G_i(x)$. Then, $\mathcal{B}$ computes

$$S_i = \prod_{i=0}^{q_H - 1} (Y_i)^{c_i} = (g_2')^{G_i(s)} = (g_2')^{\Sigma_{i=0}^{q_H - 1} c_i s^i} = (g_2')^{\frac{G(s)}{s + w_i}} = g_2^{\frac{1}{s + w_i}} = g_2^{\frac{1}{\beta + w_i}} \in \mathbb{G}_2 \tag{12}$$

and sets

$$\pi_i = \psi(S_i) = (\psi(g_2))^{\frac{1}{\beta + w_i}} = g_1^{\frac{1}{\beta + w_i}} \in \mathbb{G}_1. \tag{13}$$

According to the AuthInit definition, $\Gamma_i = (\delta_i, r_i, \pi_i, \mathcal{F}_i)$ is a valid authentication code of M_i and the simulator $\mathcal{B}$ responds to the authentication query with Γ_i.

Output. The adversary outputs a forged authentication code $\Gamma_E^* = (\delta^*, \pi^*, \pi_E^*)$. According to the InitAuth definition and simulation, we have

$$e\left(\pi^*, u g_2^{H_2(\delta^*)}\right) = e\left(\pi^*, g_2^{\beta} \cdot g_2^{H_2(\delta^*)}\right) = e(g_1, g_2). \tag{14}$$

Then, we have

$$\pi^* = g_1^{1/(\beta + H_2(\delta^*))}. \tag{15}$$

According to the H_2-query, $\mathcal{B}$ can get $H_2(\delta^*) = \omega^*$ with a non-negiligible probability of $\frac{1}{q_H}$ such that

$$\pi^* = g_1^{1/(\beta + H_2(\delta^*))} = g_1^{1/(\beta + \omega^*)}. \tag{16}$$

Since $g_1 = \psi(g_2) = \psi((g_2')^{G(s)}) = (\psi(g_2'))^{G(s)} = (g_1')^{G(s)}$ and $\beta = s$, then

$$\pi^* = g_1^{1/(\beta + \omega^*)} = (g_1')^{\frac{G(\beta)}{\beta + \omega^*}} = (g_1')^{\frac{G(s)}{s + \omega^*}} = (g_1')^{\frac{(s + \omega_1) \cdots (s + \omega_j) \cdots (s + \omega_{q_H})}{(s + \omega^*)}}. \tag{17}$$

Since we are in Case 1, that is, $\forall i \in [1, q_H]$, $\delta^* \neq \delta_i$. Therefore, $\forall i \in [1, q_H]$, $\omega^* \neq \omega_i$. That is,

$$\frac{(s + \omega_1) \cdots (s + \omega_j) \cdots (s + \omega_{q_H})}{(s + \omega^*)} \tag{18}$$

cannot be reduced by $(s + \omega^*)$, and π^* can be written as

$$\pi^* = g_1^{1/(s + H_2(\delta^*))} = (g_1')^{\frac{G(s)}{s + \omega^*}} = (g_1')^{G_{-1}(s) + \frac{\zeta}{(s + \omega^*)}}, \tag{19}$$

where $G_{-1}(s)$ is a $(q_H - 1)$-degree polynomial function in s and ζ is a nonzero integer.

At last, $\mathcal{B}$ computes

$$\left(\frac{\pi^*}{\psi\left((g_2')^{G_{-1}(s)}\right)}\right)^{\frac{1}{\zeta}} = \left(\frac{(g_1')^{G_{-1}(s) + \frac{\zeta}{(s + \omega^*)}}}{(g_1')^{G_{-1}(s)}}\right)^{\frac{1}{\zeta}} = (g_1')^{\frac{1}{s + \omega^*}} \in \mathbb{G}_1 \tag{20}$$

and outputs $(\omega^*, (g_1')^{\frac{1}{s + \omega^*}})$ as the solution to the q-SDH problem instance for $\omega^* \in \mathbb{Z}_p$.

Case 2.

Setup. $\mathcal{B}$ sets $g_1 = g_1'$, $g_2 = g_2'$, $\alpha = s$, and randomly chooses $\beta \in \mathbb{Z}_p$. Then $\mathcal{B}$ computes

$$u = g_2^{\beta}, \tag{21}$$

and

$$g_2^{\alpha} = g_2^s = (g_2')^s. \tag{22}$$

Furthermore, $\mathcal{B}$ computes

$$y_i = g_1^{\alpha^i} = (g_1')^{s^i} = \psi\left((g_2')^{s^i}\right) = \psi(Y_i) \tag{23}$$

for all $i \in [0, l-1]$.

Lastly, $\mathcal{B}$ chooses two hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}_p$, $H_2 : \mathbb{G}_1 \to \mathbb{Z}_p$ and sends the public parameter $\mathsf{Param} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, p, e, H_1, H_2)$ together with the public key $\mathrm{pk}_{\mathsf{Auth}} = (\{y_i\}_{0 \leqslant i \leqslant l-1}, u, g_2^{\alpha})$ to $\mathcal{A}$.

Authentication query. Let q_s be the number of $\mathcal{A}$'s queries in this phase. For each query $(M_i, \mathcal{F}_i)_{1 \leqslant i \leqslant q_H}$, $\mathcal{B}$ randomly chooses $r_i \in \mathbb{Z}_p$ and let $F^{(i)}(x)$ be the polynomial $F^{(i)}(x) = \Pi_{j=1}^{l}(h_{i_j} + x)$. Here, $M_{i_j} = f_{i_j}(M_i, \mathrm{aux}_{i_j})$ and $h_{i_j} = H_1(M_{i_j})$ for all $f_{i_j} \in \mathcal{F}_i$, $j \in [1, l]$ and $i \in [1, q_H]$. We expand $F^{(i)}(x)$ and write $F^{(i)}(x) = \Sigma_{j=0}^{l} a_j^{(i)} x^j$ where $a_j^{(i)} \in \mathbb{Z}_p$ for $j = 0, \ldots, l$ are the coefficients of the polynomial $F^{(i)}(x)$ with $i \in [1, q_H]$.

$$\delta_i = \psi\left(\left(\prod_{j=0}^{l} Y_j\right)^{a_j^{(i)}}\right) = \psi\left((g_2')^{r_i \Pi_{j=0}^{l} a_j^{(i)} s_j}\right) = (g_1')^{r_i \Pi_{j=1}^{l}(h_{i_j}+s)} = g_1^{r_i \Pi_{j=1}^{l}(h_{i_j}+\alpha)}, \tag{24}$$

and

$$\pi_i = (g_1')^{1/(\beta + H_2(\delta_i))} = g_1^{1/(\beta + H_2(\delta_i))}, \tag{25}$$

where $h_{i_j} = H_1(M_{i_j})$ and $M_{i_j} = f_{i_j}(M_i)$ for all $f_{i_j} \in \mathcal{F}_i \wedge j \in [1, l]$. Lastly, $\mathcal{B}$ responds to this query with $\Gamma_i = (\delta_i, r_i, \pi_i, \mathcal{F}_i)$.

Output. The adversary outputs a forged authentication code $\Gamma_E^* = (\delta^*, \pi^*, \pi_E^*)$ for an image that M^*. According to the $\mathsf{Process}$ definition and simulation, we have

$$\pi_E^* = g_1^{r^* \prod_{j=1, h_j^* \neq h^*}^{l} (h_j^* + \alpha)} = g_1^{r^* \prod_{j=1, h_j^* \neq h^*}^{l} (h_j^* + s)}, \tag{26}$$

where $h^* = H_1(M^*)$ and $h_j^* = H_1(M_j^*)$. Furthermore, we have $M_j^* = f_j^*(M^*)$ for all $f_j^* \in \mathcal{F}^*$ and $j \in [1, l]$.

If we are in Case 2, according to the security model, the $(M^*, \Gamma^*) = (\delta^*, \Gamma^*, \Gamma_E^*)$ is a valid forgery such that for all i= 1 to q_s,

$$\forall f_{i_j} \in \mathcal{F}_i, \quad M^* \neq f_{i_j}(M_i). \tag{27}$$

Therefore, $\forall i \in [1, q_s], h^* \neq h_{i_j}$ and $F^{(i)}(s) = \prod_{j=1}^{l}(h_{i_j} + s)$ is not divisible by $h^* + s$.

Since we are in Case 2, then $\exists i \in [1, q_H]$, $\delta^* = \delta_i$. Without loss of generality, we assume that $\delta^* = \delta_k$ with an integer $k \in [1, q_H]$ where

$$\delta^* = \delta_k = g_1^{r_k \cdot \prod_{j=1}^{l}(h_{k_j}+s)} = g_1^{r_k \cdot F^{(k)}(s)}. \tag{28}$$

For a valid forgery, we have

$$\pi_E^* = (\delta^*)^{\frac{1}{h^*+\alpha}} = (\delta_k)^{\frac{1}{h^*+s}} = (g_1')^{\frac{r_k \cdot F^{(k)}(s)}{h^*+s}} = (g_1')^{r_k \cdot F_{-1}^{(k)}(s) + \frac{\eta}{h^*+s}}, \tag{29}$$

where $F_{-1}^{k}(s)$ is a $(l-1)$-degree polynomial function in s and η is a nonzero integer.

At last, $\mathcal{B}$ computes

$$\left(\frac{\delta^*}{\psi\left((g_2')^{F_{-1}^{(k)}(s)}\right)^{r_k}}\right)^{\frac{1}{\eta}} = \left(\frac{(g_1')^{r_k \cdot F_{-1}^{(k)}(s) + \frac{\eta}{(h^*+s)}}}{\left((g_1')^{F_{-1}^{(k)}(s)}\right)^{r_k}}\right)^{\frac{1}{\eta}} = (g_1')^{\frac{1}{h^*+s}} \in \mathbb{G}_1 \tag{30}$$

and outputs $(h^*, (g_1')^{\frac{1}{h^*+s}})$ as the solution to the q-SDH problem instance for $h^* \in \mathbb{Z}_p$.

After outputting the solutions in the aforementioned two cases, $\mathcal{B}$ completes its simulation. The correctness is analyzed as follows.

When $\mathcal{A}$ outputs a valid forgery, if it is in Case 1, $\mathcal{B}$ can succeed only when it can successfully guess i^* and all the authentication queries are simulatable. Therefore, in Case 1 the probability of $\mathcal{B}$'s success

Table 4 General analysis of our scheme

Algorithm	Computation cost	Communication cost
KeyGen	$n \cdot \mathsf{Exp}_{\mathbb{G}_1} + n \cdot \mathsf{Exp}_{\mathbb{Z}_p} + 2\mathsf{Exp}_{\mathbb{G}_2}$	$n \cdot \lvert\mathbb{G}_1\rvert + 2\lvert\mathbb{G}_2\rvert$
InitAuth	$2\mathsf{Exp}_{\mathbb{G}_1} + \mathsf{Hash}_{\mathbb{G}_1,\mathbb{Z}_p}$	$2\lvert\mathbb{G}_1\rvert + \lvert p\rvert + \mathsf{Sizeof}(\mathcal{F})$
Process	$\mathsf{Exp}_{\mathbb{G}_1} + l \cdot \mathsf{Mul}_{\mathbb{G}_1}$	$3\lvert\mathbb{G}_1\rvert$
Verify	$4\mathsf{Pairing}_{\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T} + \mathsf{Exp}_{\mathbb{G}_2} + \mathsf{Mul}_{\mathbb{G}_2} + \mathsf{Hash}_{\mathbb{G}_1,\mathbb{Z}_p}$	–

is $\frac{1}{q_H}$ for q_H queries made by $\mathcal{A}$. Furthermore, if in Case 2, all queries can be answered correctly with Probability 1.

Probability 1. Suppose the adversary breaks our scheme with a success probability ϵ by making q_H queries to the random oracle. Let $\Pr[S]$ be the probability that $\mathcal{B}$ solves q-SDH problem, $\Pr[C_1]$ be the probability that Case 1 occurs and $\Pr[C_2]$ be the probability that Case 2 occurs, it follows that

$$\begin{aligned}\Pr[S] &= \Pr[S|C_1]\Pr[C_1] + \Pr[S|C_2]\Pr[C_2] \\ &= \frac{\epsilon}{q_H} \cdot \Pr[C_1] + \epsilon \cdot \Pr[C_2] \\ &\geqslant \frac{\epsilon}{q_H}(\Pr[C_1] + \Pr[C_2]) = \frac{\epsilon}{q_H}. \end{aligned} \tag{31}$$

Therefore, if our scheme can be broken with a non-negligible probability ϵ, the q-SDH problem can also be solved with a non-negligible probability $\frac{\epsilon}{q_H}$. It follows that our scheme is secure under the q-SDH assumption. This completes the proof.

6 Evaluations

In this section, we demonstrate the performance of our scheme with a bunch of evaluations.

6.1 General analysis

Because the computation of the hash function: $H_1 : \{0,1\}^\lambda \to \mathbb{Z}_p$ and the image processing functions f_i is efficient, we only consider other more costly computations in our scheme, including following seven operations: (1) the hash computation $H_2 : \mathbb{G}_1 \to \mathbb{Z}_p$ denoted by $\mathsf{Hash}_{\mathbb{G}_1,\mathbb{Z}_p}$; (2) multiplication operation in group $\mathbb{G}_1$ denoted by $\mathsf{Mup}_{\mathbb{G}_1}$; (3) multiplication operation in group $\mathbb{G}_2$ denoted by $\mathsf{Mup}_{\mathbb{G}_2}$; (4) exponentiation operation in $\mathbb{Z}_p$ denoted by $\mathsf{Exp}_{\mathbb{Z}_p}$; (5) exponentiation operation in $\mathbb{G}_1$ denoted by $\mathsf{Exp}_{\mathbb{G}_1}$; (6) exponentiation operation in $\mathbb{G}_2$ denoted by $\mathsf{Exp}_{\mathbb{G}_2}$; (7) pairing operation in $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ denoted by $\mathsf{Pairing}_{\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T}$. The communication cost relies on the size of the group elements denoted by $|\mathbb{G}_1|$ and $|\mathbb{G}_2|$, size of the group order denoted by $|p|$, the number of predefined image processing methods denoted by $l = |\mathcal{F}|$, and the memory cost of processing function denoted by $\mathsf{Sizeof}(\mathcal{F})$. We conclude the analysis of both computation and communication costs in Table 4.

From the aspect of efficiency, our design has a constant-size authentication overhead ($\leqslant$ 2 KB) and constant verification time (around 0.15 s). While the time of generating authentication overhead increases linearly with the number of permissible editing operations, it is still efficient. We will give the detailed analysis of efficiency as follows.

6.2 Instantiation and performances

To test the efficiency of our scheme more precisely, we ran our prototype on the 100 test images and measured the average runtime of KeyGen, InitAuth, Process and Verify algorithms, using a predefined set of processing functions presented in Table 3. Because of the space limitation, we only present some test images in Figure 4.

We implement the scheme utilizing GNU Multiple Precision Arithmetic (GMP) Library[3)] and Pairing-Based Cryptography (PBC) Library[4)] by choosing two types of elliptic curves (types D and F). These

3) https://gmplib.org.
4) https://crypto.stanford.edu/pbc.

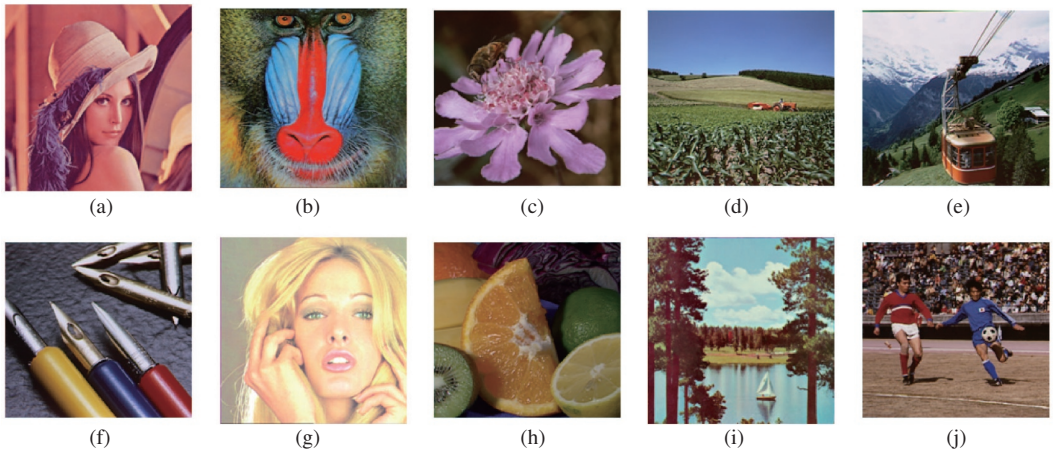

Figure 4 (Color online) Testing images: (a) Lenna (512 × 512); (b) baboon (500 × 480); (c) flower (512 × 480); (d) cornfield (512 × 480); (e) cablecar (512 × 480); (f) pens (512 × 480); (g) Tiffany (512 × 512); (h) fruits (512 × 480); (i) sailboat (512 × 512); (j) soccer (512 × 480).

Table 5 Type of pairings of our scheme

Curve type	Size of group member (bits)	Security level
D	$\lvert\mathbb{G}_1\rvert, \lvert\mathbb{G}_2\rvert, \lvert\mathbb{G}_T\rvert : (320, 960, 960)$	80-bit
F-80	$\lvert\mathbb{G}_1\rvert, \lvert\mathbb{G}_2\rvert, \lvert\mathbb{G}_T\rvert : (160, 320, 960)$	80-bit
F-112	$\lvert\mathbb{G}_1\rvert, \lvert\mathbb{G}_2\rvert, \lvert\mathbb{G}_T\rvert : (224, 448, 1324)$	112-bit
F-128	$\lvert\mathbb{G}_1\rvert, \lvert\mathbb{G}_2\rvert, \lvert\mathbb{G}_T\rvert : (256, 512, 1536)$	128-bit

two elliptic curves are asymmetric pairings. We present the size of parameters in Table 5.

Our experiment device is an ordinary desktop of a 2-core 3.4 GHz Intel i5-7500 processor and 8 G of RAM. We encode the cryptographic algorithms using C++ language combined with the OpenCV2.0, and the code is compiled by Visual Studio 2017. We use the aforementioned two types of elliptic curves with different security levels (Table 5) to implement our scheme. The average runtime of algorithms in our scheme can be found in Figure 5.

From Figure 5 we can see that when the number of processing methods $l = 100$ with a construction using the curve D with 80-bit security level, the KeyGen algorithm takes an average time about 0.2 s, the InitAuth takes only about 0.002 s, and Process takes about 0.03 s, and the Verify takes only 0.05 s. That is, when the original image is allowed to be edited using 100 types of operations, the total time cost of all the child processes in our scheme is less than 0.3 s. The evaluation shows that our scheme can be used in an efficient image authentication system.

Notice that, the KeyGen algorithm is much time-cost than InitAuth, Process and Verify algorithms. However, since the KeyGen algorithm is called only once in the lifetime of the system and we can run the KeyGen algorithm offline, the online performance of our scheme can be improved dramatically.

We can conclude that the construction based on curve D is more efficient in verification because it has a shorter group size in $\mathbb{G}_T$ where pairing computations in the Verify algorithm are implemented. While the curve F may be the better choice to achieve both efficient computation performance and higher security (128-bit security). The time-cost of our experiment results is consistent with the analysis in Table 4.

We also compare our scheme with the one in [13] in Table 6. Both schemes have robustness, security, and versatility, but ours is more efficient in computation and communication. The scheme in [13] uses costly Zero-Knowledge proof to achieve the goal of (very strong) image privacy protection. As a result, our scheme would be a better choice in the scenarios where image privacy is not a major concern.

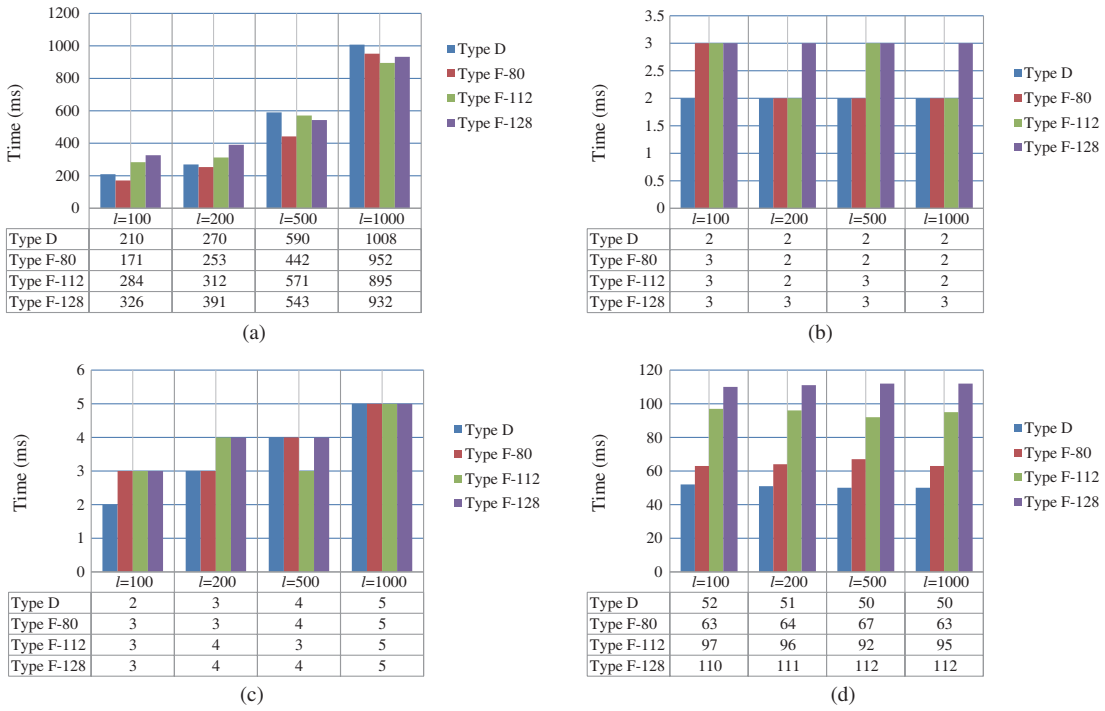

Figure 5 (Color online) The time cost of (a) KeyGen, (b) InitAuth, (c) Process, and (d) Verify in our scheme by choosing two different elliptic curves.

Table 6 Performance compared with [13][a)]

Scheme	Image size (pixels)	Time costs (s)			Memory cost (kB)	
		Key generation	Authentication	Verification	Key size	Size overhead
[13]	128×128	≈ 367	≈ 306	≈ 0.5	2.6×10^6	2.67
Our scheme	128×128	≈ 1	≈ 0.33	≈ 0.05	$\leqslant 30$	$\leqslant 2$

a) The evaluation of [13] is excerpted from the Table II in [13], and the evaluation of our scheme is based on a construction from the aforementioned curve D with 1000 types of permissible operations.

7 Conclusion

This paper presents a new design of image authentication. With our new scheme, the image owner can define a set of permissible image processing operations (e.g., brightness adjustment, JPEG compression and denoising) on an authenticated image. Any image holder can edit the image and generate a proof without any interaction with the image owner. The proof is publicly verifiable, which can convince image user the validity of an altered image. It is computationally infeasible to generate a valid proof of maliciously altered images, i.e., editing an image with operations not defined by the image owner. This is ensured by a rigorous theoretical analysis under the q-SDH assumption.

The authentication overhead in our scheme is a constant, and the verification time is also a constant. This is independent of permissible editing operations or image size, an improvement over previous image authentication with robustness, security and versatility. Evaluation is given on an ordinary PC. When 1000 types of permissible image processing operations are allowed, proof generation takes about 0.33 s, and the verification takes 0.05 s. As privacy would be a concern in certain situations and existing designs usually use costly Zero-Knowledge Proof to achieve privacy, it is worthwhile to study practically-usable image authentication with not only robustness, security, versatility, but also privacy in our future work.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. 61902070, 61822202, 62032005, 61872089, 61872087).

References

1 Zhang H G, Han W B, Lai X J, et al. Survey on cyberspace security. Sci China Inf Sci, 2015, 58: 110101
2 Du L, Cao X C, Zhang W, et al. Semi-fragile watermarking for image authentication based on compressive sensing. Sci China Inf Sci, 2016, 59: 059104
3 Friedman G L. The trustworthy digital camera: restoring credibility to the photographic image. IEEE Trans Consumer Electron, 1993, 39: 905–910
4 Merkle R C. A digital signature based on a conventional encryption function. In: Proceedings of Conference on the Theory and Applications of Cryptographic Techniques, Santa Barbara, 1987. 369–378
5 Venkatesan R, Koon S, Jakubowski M H, et al. Robust image hashing. In: Proceedings of the 2000 International Conference on Image Processing, Vancouver, 2000. 664–666
6 Monga V, Evans B L. Perceptual image hashing via feature points: performance evaluation and tradeoffs. IEEE Trans Image Process, 2006, 15: 3452–3465
7 Xiang S J, Yang J Q, Huang J W. Perceptual video hashing robust against geometric distortions. Sci China Inf Sci, 2012, 55: 1520–1527
8 Huang F J, Huang J W. Calibration based universal JPEG steganalysis. Sci China Ser F-Inf Sci, 2009, 52: 260–268
9 Lin C-Y, Chang S-F. A robust image authentication method distinguishing JPEG compression from malicious manipulation. IEEE Trans Circ Syst Video Technol, 2001, 11: 153–168
10 Tang Z, Zhang X, Li X, et al. Robust image hashing with ring partition and invariant vector distance. IEEE Trans Inform Forensic Secur, 2016, 11: 200–214
11 Kim J, Lee S, Yoon J, et al. PASS: privacy aware secure signature scheme for surveillance systems. In: Proceedings of the 14th IEEE International Conference on Advanced Video and Signal Based Surveillance, Lecce, 2017. 1–6
12 Chen H, Wang S, Zhang H, et al. Image authentication for permissible cropping. In: Proceedings of the 14th International Conference on Information Security and Cryptology, Fuzhou, 2018. 308–325
13 Naveh A, Tromer E. Photoproof: cryptographic image authentication for any set of permissible transformations. In: Proceedings of IEEE Symposium on Security and Privacy, San Jose, 2016. 255–271
14 Ben-Sasson E, Chiesa A, Tromer E, et al. Succinct non-interactive arguments for a von neumann architecture. IACR Cryptol ePrint Archive, 2013, 2013: 879
15 Katz J, Lindell Y. Introduction to Modern Cryptography. 2nd ed. Boca Raton: CRC Press, 2014
16 Merkle R C. A certified digital signature. In: Proceedings of the 9th Annual International Cryptology Conference, Santa Barbara, 1989. 218–238
17 Xie L H, Arce G R, Graveman R F. Approximate image message authentication codes. IEEE Trans Multimedia, 2001, 3: 242–252
18 Tang Z, Zhang X, Huang L, et al. Robust image hashing using ring-based entropies. Signal Process, 2013, 93: 2061–2069
19 Xiang S, Kim H, Huang J. Histogram-based image hashing scheme robust against geometric deformations. In: Proceedings of the 9th Workshop on Multimedia & Security, Dallas, 2007. 121–128
20 Choi Y S, Park J H. Image hash generation method using hierarchical histogram. Multimed Tools Appl, 2012, 61: 181–194
21 Tang Z, Dai Y, Zhang X, et al. Perceptual image hashing with histogram of color vector angles. In: Proceedings of the 8th International Conference Active Media Technology, Macau, 2012. 237–246
22 Gharde N D, Thounaojam D M, Soni B, et al. Robust perceptual image hashing using fuzzy color histogram. Multimed Tools Appl, 2018, 77: 30815–30840
23 Lou D-C, Liu J-L. Fault resilient and compression tolerant digital signature for image authentication. IEEE Trans Consumer Electron, 2000, 46: 31–39
24 Sun Q, Chang S F. A robust and secure media signature scheme for JPEG images. J VLSI Sign Process Syst Sign Image Video Technol, 2005, 41: 305–317
25 Kee E, Johnson M K, Farid H. Digital image authentication from JPEG headers. IEEE Trans Inform Forensic Secur, 2011, 6: 1066–1075
26 Iida K, Kiya H. Robust image identification for double-compressed and resized JPEG images. In: Proceedings of Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, Honolulu, 2018. 1968–1974
27 Yao H, Wei H, Qin C, et al. An improved first quantization matrix estimation for nonaligned double compressed JPEG images. Signal Process, 2020, 170: 107430
28 Lei Y, Wang Y, Huang J W. Robust image hash in Radon transform domain for authentication. Signal Process-Image Commun, 2011, 26: 280–288
29 Tang Z, Ling M, Yao H, et al. Robust image hashing via random Gabor filtering and DWT. Comput Materials Continua, 2018, 55: 331–344
30 Hu Y C, Lo C C, Chen W L. Probability-based reversible image authentication scheme for image demosaicking. Future Generation Comput Syst, 2016, 62: 92–103
31 Davarzani R, Mozaffari S, Yaghmaie K. Perceptual image hashing using center-symmetric local binary patterns. Multimed Tools Appl, 2016, 75: 4639–4667
32 Swaminathan A, Mao Y, Wu M. Image hashing resilient to geometric and filtering operations. In: Proceedings of IEEE 6th Workshop on Multimedia Signal Processing, 2004. 355–358
33 Swaminathan A, Mao Y, Wu M. Robust and secure image hashing. IEEE Trans Inform Forensic Secur, 2006, 1: 215–230

34 Rackoff C, Simon D R. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In: Proceedings of the 11th Annual International Cryptology Conference, Santa Barbara, 1991. 433–444
35 Krawczyk H, Rabin T. Chameleon hashing and signatures. IACR Cryptol ePrint Archive, 1998, 1998: 10
36 Chen X, Zhang F, Susilo W, et al. Identity-based chameleon hashing and signatures without key exposure. Inf Sci, 2014, 265: 198–210
37 Chen X, Zhang F, Kim K. Chameleon hashing without key exposure. In: Proceedings of the 7th International Conference Information Security, Palo Alto, 2004. 87–98
38 Ateniese G, de Medeiros B. On the key exposure problem in chameleon hashes. In: Proceedings of the 4th International Conference on Security in Communication Networks, Amalfi, 2004. 165–179
39 Bellare M. A note on negligible functions. IACR Cryptology ePrint Archive, 1997, 1997: 4
40 Pointcheval D, Stern J. Security proofs for signature schemes. In: Proceedings of International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, 1996. 387–398
41 Boneh D, Lynn B, Shacham H. Short signatures from the Weil Pairing. In: Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, 2001. 514–532
42 Mitsunari S, Sakai R, Kasahara M. A new traitor tracing. IEICE Trans Fundamentals Electron Commun Comput Sci, 2002, 85: 481–484
43 Boneh D, Boyen X. Short signatures without random oracles. In: Proceedings of International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, 2004. 56–73
44 You J, Reiter U, Hannuksela M M, et al. Perceptual-based quality assessment for audio-visual services: a survey. Signal Process-Image Commun, 2010, 25: 482–501
45 Zhai G T, Min X K. Perceptual image quality assessment: a survey. Sci China Inf Sci, 2020, 63: 211301
46 Nguyen L. Accumulators from bilinear pairings and applications. In: Proceedings of Cryptographers' Track at the RSA Conference, San Francisco, 2005. 275–292
47 Baric N, Pfitzmann B. Collision-free accumulators and fail-stop signature schemes without trees. In: Proceedings of International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, 1997. 480–494
48 Guo F, Susilo W, Mu Y. Introduction to Security Reduction. Berlin: Springer, 2018