Cybersecurity

RESEARCH Open Access

Verifiable image revision from chameleon hashes

Junpeng Xu, Haixia Chen*, Xu Yang, Wei Wu and Yongcheng Song

Abstract
In a digital society, the rapid development of computer science and the Internet has greatly facilitated image applications. However, one of the public network also brings risks to both image tampering and privacy exposure. Image authentication is the most important approaches to verify image integrity and authenticity. However, it has been challenging for image authentication to address both issues of tampering detection and privacy protection. One aspect, image authentication requires image contents not be changed to detect tampering. The other, privacy protection needs to remove sensitive information from images, and as a result, the contents should be changed. In this paper, we propose a practical image authentication scheme constructed from chameleon hashes combined with ordinary digital signatures to make tradeoff between tampering detection and privacy protection. Our scheme allows legitimate users to modify contents of authenticated images with a privacy-aware purpose (for example, cover some sensitive areas with mosaics) according to specific rules and verify the authenticity without interaction with the original authenticator. The security of our scheme is guaranteed by the security of the underlying cryptographic primitives. Experiment results show that our scheme is efficient and practical. We believe that our work will facilitate image applications where both authentication and privacy protection are desirable.

Keywords: Image authentication, Privacy protection, Cryptography, Chameleon hashes, Digital signatures

Introduction

Nowadays, we have stepped into an information era where digital data plays an important role in people's life. As the most common visual data, digital images are widely used in almost all aspects to convey crucial information. The development of information technology has contributed to significant advancements in image processing, since a digital image can be edited flexibly in computers with powerful image processing software (e.g., the Photoshop). However, flexible image editing tools also make malicious attacks (tampering, forgery, and so forth) towards images more accessible and imperceptible, which brings potential risks to image applications, especially in case if a digital image is used as a legal evidence and processing towards it should be controllable and authorizable.

Image authentication is the technology of verifying image origin, integrity and authenticity, and it is significant to security-relevant image applications for the purpose of tampering detection. Digital watermarks (Al-Otum 2014; Ur-Rehman and Zivic 2017; Lu and Liao 2001; Wang et al. 2018) and perceptual hashes (Venkatesan et al. 2000; Pun et al. 2018; Jiang and Pang 2018; Du et al. 2020) are important techniques for image authentication. Digital watermarks embed invisible information called watermarks by transforming original images at perception-tolerable levels, and verification is the extraction of the watermarks with reverse transformation.

Digital watermarks can be classified into fragile watermarks, robust watermarks, and semi-fragile watermarks according to their robustness. Fragile watermarking (Sreenivas and Prasad 2018) is mainly used in integrity detection, and it does not tolerate any editing of the

*Correspondence: tummy7882@hotmail.com
College of Computer and Cyber Security, Fujian Normal University, Fuzhou, China

image. Robust watermarking (Agarwal et al. 2019) is generally used for copyright protection, which can prove the origin of the image, however, it can not distinguish rational operations from malicious ones. Semi-fragile (Peng et al. 2010; Lin and Chang 2000; Chen et al. 2017) watermarking can support editing detection, but its universality is not satisfying. Moreover, the embedding of the watermark needs to modify the original image, which sometimes brings negative impacts on image applications.

Perceptual hashes firstly extract features based on image contents. The extracted features are quantized, compressed, and encoded into a binary vector to compute a hash value, and the hash value is used as a digital “fingerprint” to authenticate the image. Since the extracted features are robust to reasonable editing operations which do not cause content distortion, perceptual hashes are available to provide image content authentication. However, when the contents of an authenticated image need to be changed even for a rational purpose (for example, a privacy-aware editing by removing sensitive information), the perceptual hashes do not work well. In addition, computation of perceptual hash is relative to methods of feature extraction and its universality is also not satisfying.

Overall, ordinary image authentication techniques, such as semi-fragile watermarking and perceptual hashing, tend to authenticate image contents instead of image operations, and have shortages in distinguishing malicious image operations from permissible and authorized ones accurately.

Motivations and contributions

Let us begin with an example. When there is a traffic accident, images/video captured by the dashcam can be used as important evidence by the Officer to make a responsibility-confirmation report. The raw image captured by hardware may contain some sensitive information that is not suitable to be disclosed. When the image is used as evidence in a court, the sensitive area of the image should be hidden away (covered with some mosaics) for privacy protection. In this case, an image authentication scheme is required to ensure that images are still authentic after undergoing some privacy-aware local revisions (Fig. 1). However, we think that even an authorized editor could have dishonest motives to edit the image. In our design, the image is divided into editable and non-editable parts. The non-editable parts which are hashed by an ordinary hash function are important and necessary. These parts refer to core contents of the image which should not be changed to avoid excessive content distortion. If the entire image is editable and the editor can change all the contents at will, we think it is not so practical in some real-world applications.

In this paper, we propose a practical image authentication scheme for privacy protection (Fig. 1). Our scheme allows legitimate users to modify the image contents with the privacy-aware purpose (for example, cover some sensitive areas with mosaic) according to specified rules. The verification of the modified image does not need any interaction with the original authenticator. In our design, the signer and editor are distinct entities. The signer is the image copyright owner and an editor is an image user who is authorized to edit the image limitedly. The owner’s purpose of authenticating an image is to allow a specific user to edit the image in controllable ways, but not deny his behavior. Our contributions mainly consist of the following two aspects.

Flexible but verifiable image revision

In our scheme, we use chameleon hashing (Ateniese and de Medeiros 2004) combined with digital signatures to design an image authentication scheme, to achieve both flexible and verifiable revision. In our scheme, the original image is blocked and divided into several areas. An image producer authenticates the image and defines an editing rule. The rule denotes which areas in the authenticated image can be masked/edited. These editable areas are hashed with chameleon hashes to compute authentication codes (Fig. 2). A specific user (trapdoor holder of the

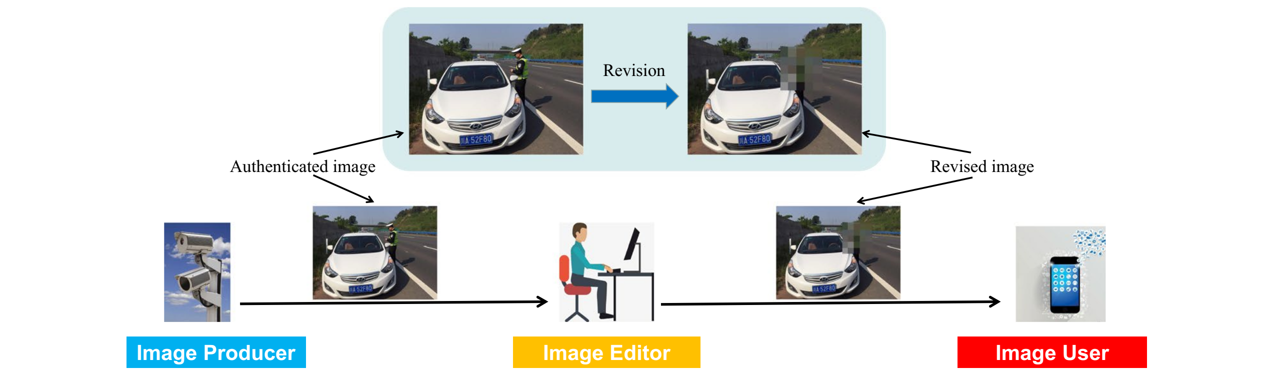

Fig. 1 The workflow of a privacy-aware image authentication system

chameleon hash) edits/revises the image by replacing the editable areas with arbitrary contents but not invalidate the codes. This provides a flexible image revision.

In addition, for non-editable areas, image blocks are hashed by ordinary hashes. Any change (even 1-bit change) of the non-editable areas will invalidate the authentication. This provides a verifiable revision to the authenticated image.

In summary, editable areas in authenticated images can be replaced with arbitrary contents while the rest areas should be the same as before. That is, our scheme is verifiable by limiting image editing. Therefore, the flexible revision is useful for privacy-aware editing and the verifiable revision contributes to detecting tampering.

Authority

Image editor in our scheme is authorizable. In our scheme, nobody except the trapdoor holder of chameleon hashing can make collisions for the revised values. That is, our scheme achieves user control by specifying the editor of the image. Concretely, no one except a specific user can revise an image without invalidating the original authentication. If an image is revised and verified as true, the reviser can not repudiate his/her behavior.

Organization

The rest of the paper is organized as follows. In "Related work", we briefly review the literature on image editing authentication . In "Preliminaries", we present the relevant knowledge required by this paper. In "Definition", we present the formal definitions of our scheme. In "Construction", we describe the scheme and analyze its security in "Security analysis". In "Experiments and evaluation", theoretical analysis and experimental simulation of the scheme are given and we conclude this paper in "Conclusions" section.

Related work

As mentioned in the introduction, digital watermarks and digital signatures are important technology for image authentication. However, both schemes are expected to be very sensitive to any malicious modification used on the image. A good image authentication system should be able to tolerate modifications that are sufficiently needed. In order to make the image authentication system tolerate reasonable modification, various schemes in the field of watermarks and digital signatures have been actively investigated and proposed. We introduce some related designs as follows.

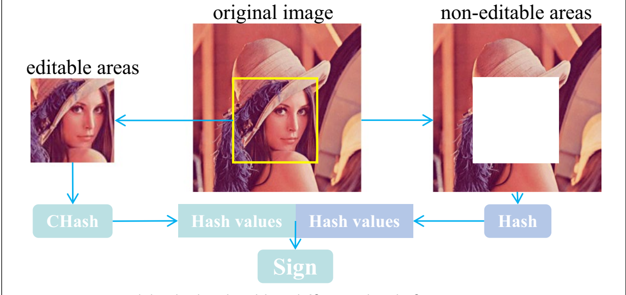

Fig. 2 Image blocks hashed by different hash functions

The semi-fragile watermark is introduced in image authentication to protect copyright and tolerate some normal image processing. In Yu et al. (2017), a detection method is proposed to calculate and measure the error between watermark image and tampered image. A novel wavelet domain image authentication scheme is proposed in Al-Otum (2014), which uses a semi-fragile watermark to detect and locate malicious tampering accurately in images. An algorithm with both robust watermarking and semi-fragile watermarking is proposed in Fridrich et al. (2002). The algorithm can realize the dual functions of copyright protection and integrity authentication. Though semi-fragile watermarks provide solutions for image authentication with robustness, it is difficult to prove the security of a watermark-based image authentication scheme. Therefore, it is difficult to decide if a specific editing (such as add a logo or covered with mosaics) is an authorized operation.

Another approach based on digital signatures in image authentication was proposed in Zhu and Hu (2008). In the proposed scheme, a image is authenticated by signing features extracted from the image. The scheme allows the image to be modified by setting a threshold value, and the modified image can be verified successfully as long as the difference value between the modified image and the signature image is less than the predefined threshold value. The scheme can tolerate some operations but is not suitable for privacy-aware authenticating methods of the image since its sensitivity of image content alternation.

To solve the problem of privacy protection, some attractive designs are proposed in Chen et al. (2018), Chen et al. (2020), Kim et al. (2017), Chen et al. (2020).

Chen et.al proposed privacy-aware image authentication from commitments (Chen et al. 2018) and accumulators (Chen et al. 2020). These schemes allow users to crop the image according to the signer's predefined rules and output a valid signature. The disadvantage of this scheme is that it does not support authority. Any user can edit the image, which may lead to repudiation.

Kim et al. proposed a privacy-aware security signature scheme using chameleon hashing (Kim et al. 2017). In the proposed scheme, users can delete the object of the image legally, and the original signature is still valid. This scheme supports privacy-aware image processing operations. However, there is no editing rule defined in the scheme and the editor can edit the whole image at will, even replacing the image with a new one entirely.

Overall, existing designs have at least one of the following two drawbacks: (1) Sensitive to content revision: most of the existing schemes based on watermarks and perceptual hashes are used to provide content authentication. They are robust to operations which do not cause content distortion, but sensitive to privacy-aware operations which would change contents of the images. (2) The authority is not well addressed. Image authentication from cryptographic primitives (accumulators and commitments) detects operations but does not identify its users. Anyone is considered as an honest user if the processing operations do not invalidate the authentication, which could lead to repudiation of the revision behavior.

Preliminaries

In this section, we will present the cryptographic tools used in our scheme.

Digital Signature Schemes

A digital signature scheme ($\mathsf{D}SS$) consists of three polynomial-time algorithms: $\mathsf{DSS}=(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$:

- KeyGen: The key generation algorithm takes a security parameter 1^λ as an input, and outputs a public key pk_{sig} and a private key sk_{sig}. That is $(pk_{\mathsf{sig}}, sk_{\mathsf{sig}}) \leftarrow \mathsf{KeyGen}(1^\lambda)$.
- Sign: The signing algorithm takes a private key sk_{sig} and a message M as input, and outputs a σ. That is $\sigma \leftarrow \mathsf{Sign}(sk_{\mathsf{sig}}, M)$.
- Verify: The verifying algorithm takes a public key pk_{sig}, a signature σ, and a message M as input, and outputs a bit $b \in \{0,1\}$. That is $b \leftarrow \mathsf{Verify}(pk_{\mathsf{sig}}, M, \sigma)$.

Security requirements of DSS

It should be computationally infeasible for any adversary to compute a valid signature with a fixed public key pk_{sig} and the access to signing oracle $\mathsf{Sign}(sk_{\mathsf{sig}}, .)$ to obtain signatures of messages chosen by himself. We give the formalized definition of this security property as follows.

Definition 1 ($\mathsf{EUF-CMA}$). A signature scheme $\mathsf{DSS}=(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ is existentially unforgeable under adaptive chosen-message attacks ($\mathsf{EUF-CMA}$) if for all probabilistic polynomial-time adversaries $\mathcal{A}$, there is a negligible function negl with a secure parameter λ such that:

$$\Pr\left[\begin{array}{c}(sk_{\mathsf{sig}}, pk_{\mathsf{sig}}) \leftarrow \mathsf{KeyGen}(1^\lambda),\\ (m^*, \sigma^*) \leftarrow \mathcal{A}(pk_{\mathsf{sig}} | \mathsf{Sign}(sk_{\mathsf{sig}}, \cdot)),\\ \mathsf{Verify}(pk_{\mathsf{sig}}, m^*, \sigma^*) = true \wedge m^* \notin \Omega\end{array}\right] \leq \mathsf{negl}(\lambda). \tag{1}$$

where Ω is a set of messages which has been inquired to sign oracle $\mathsf{Sign}(\mathsf{sk_{sig}}, \cdot)$ by $\mathcal{A}$.

Chameleon hashing schemes

A chameleon hash function is a trapdoor collision-resistant hash function with a key pair (pk, sk) (Ateniese and de Medeiros 2004; Chen et al. 2004). Anyone who knows the public key pk can efficiently compute the hash value for each input. Besides, there is an efficient algorithm for the holder of the private key sk, called a trapdoor, to find collisions for any given input. Formally, a chameleon hash scheme ($\mathsf{C}HS$) consists of three algorithms:

- Key Generation: KeyGen takes a security parameter λ as input. It returns a key pair (pk, sk), where pk is a public key and sk is a secret key of the user to find collisions for any given input.
- Hash Computation: CHash that takes a public key pk, a message m, and a randomness $r \in Z_q^*$ as input, returns a hash value $h = \mathsf{CHash}(pk, m, r)$.
- Collision Computation: Adapt that takes the secret key sk of the user, a message m, a randomness $r \in Z_{q^*}$, and another message m' as input, outputs an integer $r' = \mathsf{Adapt}(sk, M, r, M')$ that satisfies

$$\mathsf{C}Hash(m, r) = \mathsf{CHash}(m', r'). \tag{2}$$

Security of chameleon hashes

The security requirements of a chameleon hash is Collision-resistance (Ateniese and de Medeiros 2004): By giving only public key pk, message m and randomness r, there is no efficient algorithm that can find a second pair (m', r') such that a hash value $C = Hash(pk, m, r)$ with more than negligible probability.

Definition

This section focuses on formal definitions of verifiable image revision from chameleon hashes.

Definition of our scheme

Our scheme consists of the following six algorithms: $\mathsf{KeyGen}, \mathsf{OrigAuth}, \mathsf{Edit}, \mathsf{Verify}, \mathsf{Proof}$, and Judge (Fig. 3).

- KeyGen: The key generation algorithm takes a security parameter λ as input and returns two key pairs $(pk_{\mathsf{sig}}, sk_{\mathsf{sig}})$ and $(pk_{\mathsf{ed}}, sk_{\mathsf{ed}})$. That is: $((pk_{\mathsf{sig}}, sk_{\mathsf{sig}}), (pk_{\mathsf{ed}}, sk_{\mathsf{ed}})) \leftarrow \mathsf{KeyGen}(1^\lambda)$.
- $\mathsf{OrigAuth}$: The authentication algorithm $\mathsf{OrigAuth}$ takes a signer's private key sk_{sig}, an editor's public key pk_{ed}, an editing rule ER, and an image M

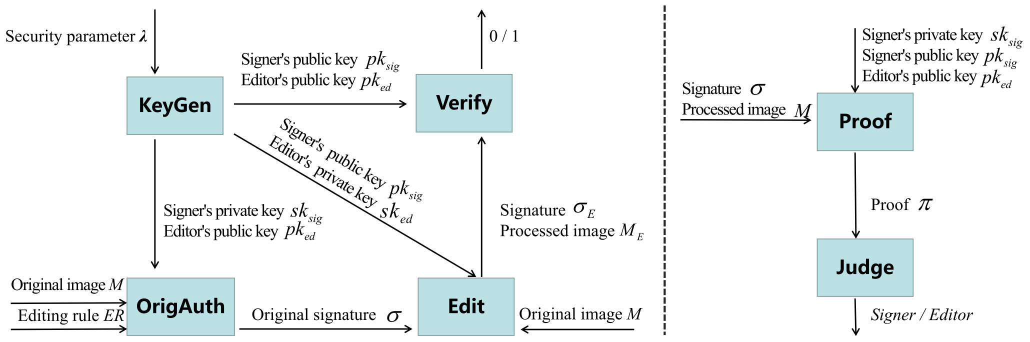
Security parameter λ
KeyGen
Signer's public key pk_{sig}
Editor's public key pk_{ed}
Verify
0 / 1
Signer's public key pk_{sig}
Editor's private key sk_{ed}
Signer's private key sk_{sig}
Editor's public key pk_{ed}
Signature σ_E
Processed image M_E
Original image M
Editing rule ER
OrigAuth
Original signature σ
Edit
Original image M
Signature σ
Processed image M
Signer's private key sk_{sig}
Signer's public key pk_{sig}
Editor's public key pk_{ed}
Proof
Proof π
Judge
Signer / Editor

Fig. 3 Frame of image authentication scheme supporting verifiable image revision

as input. It outputs an original signature σ. That is: $\sigma \leftarrow \mathsf{OrigAuth}(sk_{\mathsf{sig}}, pk_{\mathsf{ed}}, M, ER)$.

- Edit: The editing algorithm Edit takes the signer's public key pk_{sig}, the editor's private key sk_{ed}, an image M, an editing strategy ES, and an original signature σ as input. It outputs a signature σ_E together with the processed image M_E. That is $(\sigma_E, M_E) \leftarrow \mathsf{Edit}(pk_{\mathsf{sig}}, sk_{\mathsf{ed}}, M, \sigma, ES)$.
- Verify: The verification algorithm Verify takes the signer's public key pk_{sig}, the editor's public key pk_{ed}, a signature σ, an image M, and an editing rule ER as input. It outputs a bit $b \in \{0, 1\}$. That is: $b \leftarrow \mathsf{Verify}(pk_{\mathsf{sig}}, pk_{\mathsf{ed}}, M, \sigma, ER)$.
- Proof: The proof algorithm Proof takes the signer's private key sk_{sig}, the signer's public key pk_{sig}, the editor's public key pk_{ed}, and an image/signature pair (M, σ) which has been obtained by a polynomial entity as input. It outputs a proof π. That is $\pi \leftarrow \mathsf{Proof}(sk_{\mathsf{sig}}, pk_{\mathsf{sig}}, pk_{\mathsf{ed}}, M, \sigma)$. This algorithm is issued by the signer.
- `Judge`: The judge algorithm Judge takes the signer's public key pk_{sig}, the editor's public key pk_{ed}, the signature σ, a proof π, and an image M as input. It outputs $d \in \{Signer, Editor, \perp\}$. That is $d \leftarrow \mathsf{Judge}(pk_{\mathsf{sig}}, pk_{\mathsf{ed}}, M, \sigma, \pi)$.

Correctness of our scheme

For any key correctly generated by the KeyGen, any signature σ generated by the OrigAuth and σ_E generated by the Edit should be accepted by the Verify algorithm. That is :

$$\begin{aligned}
&\forall((pk_{\mathsf{sig}}, sk_{\mathsf{sig}}), (pk_{\mathsf{ed}}, sk_{\mathsf{ed}})) \leftarrow \mathsf{KeyGen}(1^\lambda)\\
&\wedge \forall\, \sigma \leftarrow \mathsf{OrigAuth}(sk_{\mathsf{sig}}, pk_{\mathsf{ed}}, M, ER)\\
&\wedge \forall\, (\sigma_E, M_E) \leftarrow \mathsf{Edit}(pk_{\mathsf{sig}}, sk_{\mathsf{ed}}, M, ES, \sigma)\\
&\Rightarrow \mathsf{Verify}(pk_{\mathsf{sig}}, pk_{\mathsf{ed}}, M, \sigma) = \mathsf{Verify}(pk_{\mathsf{sig}}, pk_{\mathsf{ed}}, M_E, \sigma_E) = 1.
\end{aligned} \tag{3}$$

Security model

Given free access to two oracles $\mathcal{O}_{OA}$ and $\mathcal{O}_{ED}$, our scheme requires that these is no PPT adversary can compute a valid signature for an image which has been edited against the editing rule, or a new image which is completely different from the original one. The security is defined as *unforgeability* and formalized with the following game *EXPU*:

```
$EXPU(\lambda)$
  $(pk_{\mathsf{sig}}, sk_{\mathsf{sig}}), (pk_{\mathsf{ed}}, sk_{\mathsf{ed}}) \leftarrow KeyGen(1^\lambda)$;
  $\Omega \leftarrow \emptyset$;
  $(M, \sigma) \leftarrow \mathcal{O}_{OA}(sk_{\mathsf{sig}}, ., ., .)$;
    $On\ input(M_i, ER_i), \mathcal{O}_{OA}\ is\ defined\ as$
      $\sigma_i \leftarrow OrigAuth(sk_{\mathsf{sig}}, pk_{\mathsf{ed}}, M_i, ER_i)$;
      $\Omega \leftarrow \Omega \cup M_i$;
      $return\ \sigma_i\ to\ \mathcal{A}$;
  $\Omega_E \leftarrow \emptyset$;
  $(M_E, \sigma_E) \leftarrow \mathcal{O}_{ED}(sk_{\mathsf{ed}}, ., ., .)$
    $On\ input(M_i, ES_i), \mathcal{O}_{ED}\ is\ defined\ as$
      $(M_E^i, \sigma_E^i) \leftarrow Edit(pk_{\mathsf{sig}}, sk_{\mathsf{ed}}, M_i, \sigma_i, ES_i)$;
      $\Omega_E \leftarrow \Omega_E \cup (M_E^i, \sigma_E^i)$;
      $return\ (M_E^i, \sigma_E^i)\ to\ \mathcal{A}$;
  $(M^*, \sigma^*, ER^*) \leftarrow \mathcal{A}$;
  $b \leftarrow \mathsf{Verify}(pk_{\mathsf{sig}}, pk_{\mathsf{ed}}, M^*, \sigma^*, ER^*)$
  $if\ b = 1 \wedge \forall i : M^* \neq M_E^i$
    $return\ 1$
  $else$
    $return\ 0.$
```

Definition 2 Our scheme is unforgeable if any probabilistic polynomial time (PPT) adversary has a negligible success probability: $\Pr[EXPU(\lambda) = 1] \leq neg(\lambda)$, here *neg* is a negligible function of λ.

Construction

We present a design based on previous achievements of existing works (Ateniese and de Medeiros 2004, 2004; Ateniese et al. 2005; Chabanne et al. 2017; Guo et al. 2016). When an image needs to be edited in some areas, image pixels are modified and even substituted by arbitrary other pixels. Motivated by sanitizable signature schemes based on chameleon hashes, our design overview is shown in Fig. 4. The signer uses OrigAuth algorithm to authenticate the image, in which the chameleon hashing algorithm is used to calculate the hash value of editable image block and the collision-resistance hashing function is used to calculate the hash value of non-editable region. The editor uses the Edit algorithm to modify the image and update the randomness so that the original signature remains valid. The verifier can use the Verify algorithm to verify the authenticity of the image. We give the technical details as follows:

Algorithm 1 KeyGen: The key generation algorithm

```
Input:
    The security parameter $\lambda$;
Output:
    Two key pairs: $(pk_{\text{sig}}, sk_{\text{sig}})$, $(pk_{\text{ed}}, sk_{\text{ed}})$;
1: $(pk_{\text{sig}}, sk_{\text{sig}}) \leftarrow$ DSS.KeyGen$(1^\lambda)$;
2: $(pk_{\text{ed}}, sk_{\text{ed}}) \leftarrow$ CHS.KeyGen$(1^\lambda)$;
3: return $((pk_{\text{sig}}, sk_{\text{sig}}), (pk_{\text{ed}}, sk_{\text{ed}}))$;
```

In the key generation algorithm KeyGen, on input the secure parameter λ, it outputs two key pairs. Here DSS is an underlying digital signature scheme and CHS is a chameleon hash scheme. $(pk_{\text{sig}}, sk_{\text{sig}})$ are keys for the original authenticator (signer) and $(pk_{\text{ed}}, sk_{\text{ed}})$ are keys for an editor.

Algorithm 2 OrigAuth: The original authentication algorighm

```
Input:
    A signer's private key $sk_{\text{sig}}$;
    An editor's public key $pk_{\text{ed}}$;
    An image $M$;
    An image editing rule denoted by $ER$;
Output:
    An original signature $\sigma$;
1: Divide $M$ into $n$ non-overlapping image blocks with the same sizes denoted by $\{M_i\}_{1 \le i \le n}$;
2: $\mathcal{H} : \{0,1\}^* \to \{0,1\}^q$;
3: for all $i = 1$ to $n$ do
4:     Draw randomness $r_i \in Z_q$;
5:     if $i \in ER$ then
6:         $h_i = \mathcal{H}(M_i, r_i)$;
7:     else
8:         $h_i \leftarrow$ CHS.Hash$(pk_{\text{ed}}, M_i, r_i)$;
9:     end if
10: end for
11: $t \leftarrow$ DSS.Sign$(h_1 || h_2 || \ldots || h_n || ER, sk_{\text{sig}})$;
12: $\sigma \leftarrow (t, < r_i >_{1 \le i \le n}, ER)$;
13: return $\sigma$;
```

In Algorithm 2, on input sk_{sig}, pk_{ed}, image M and an editing rule ER, it outputs a signature σ. The function of the algorithm is to set the editable area of the image and the authenticated image is run by the signer.

Algorithm 3 Edit: The revision algorithm

```
Input:
    A signer's public key $pk_{\text{sig}}$;
    A editor's private key $sk_{\text{ed}}$;
    An image $M$;
    The original signature $\sigma$;
    An image editing strategy denoted by $ES$;
Output:
    A revised image $M_E$;
    A signature for edited image $\sigma_E$;
1: Divide $M$ into $n$ non-overlapping image blocks with the same sizes denoted by $\{M_i\}_{1\leq i\leq n}$;
2: Parse $\sigma \leftarrow (t, < r_i >_{1\leq i\leq n}, ER)$;
3: $\mathcal{H} : \{0,1\}^* \rightarrow \{0,1\}^q$;
4: for $i = 1$ to $n$ do
5:     if $i \in ER$ then
6:         $h_i = \mathcal{H}(M_i, r_i)$;
7:     else
8:         $h_i \leftarrow$ CHS.Hash$(pk_{\text{ed}}, M_i, r_i)$;
9:     end if
10: end for
11: $b \leftarrow$ DSS.Verify$(pk_{\text{sig}}, h_1||h_2||\ldots||h_n||ER, t)$;
12: if $b = 0$ then
13:     return $\perp$;
14: end if
15: for $j \in ES$ do
16:     $r_j^{'} \leftarrow$ CHS.Adapt$(sk_{\text{ed}}, M_j, r_j, M_j')$;
17:     $M_j \leftarrow M_j'$;
18:     $r_j \leftarrow r_j'$;
19: end for
20: $M_E \leftarrow M$;
21: $\sigma_E \leftarrow (t, < r_i >_{1\leq i\leq n}, ER)$;
22: return $(M_E, \sigma_E)$;
```

Algorithm 3 takes the signer's public key pk_{sig}, the editor's private key sk_{ed}, an image M, an original signature σ, and an editing strategy denoted by ES as input. It outputs a signature σ_E together with the edited image M_E. The algorithm is run by the signer to modify the image and produce a valid signature for the modified image.

Algorithm 4 Verify: The verification algorithm

```
Input:
    A signer's public key $sk_{\text{sig}}$;
    A editor's public key $pk_{\text{ed}}$;
    An image $M$;
    The original signature $\sigma$;
    An image editing rule denoted by $ER$;
Output:
    A bit $b \in \{0,1\}$;
1: Divide $M$ into $n$ non-overlapping image blocks with the same sizes denoted by $\{M_i\}_{1\leq i\leq n}$;
2: for $i = 1$ to $n$ do
3:     if $i \in ER$ then
4:         $h_i = \mathcal{H}(M_i, r_i)$;
5:     else
6:         $h_i \leftarrow$ CHS.Hash$(pk_{\text{ed}}, M_i, r_i)$;
7:     end if
8: end for
9: $b \leftarrow$ DSS.Verify$(pk_{\text{sig}}, h_1||h_2||\ldots||h_n||ER, t)$;
10: return $b$;
```

Algorithm 4 takes the signer's public key pk_{sig}, the editor's public key pk_{ed}, an image M, a signature σ and an editing rule ER as input. It outputs a bit $b \in \{0,1\}$. The algorithm is run by a verifier to detect the authenticity of the image. If $b = 1$, the image is authentic; otherwise, the image is untrusted.

Algorithm 6 takes the signer's public key pk_{sig}, the editor's public key pk_{ed}, an image M, the signature σ and a proof π as input. It outputs *Signer* or *Editor*. The algorithm is run by a verifier to determine who has output the signature.

Algorithm 5 Proof: The proving algorithm

```
Input:
    A signer's private key $sk_{\mathsf{sig}}$;
    A signer's public key $pk_{\mathsf{sig}}$;
    A editor's public key $pk_{\mathsf{ed}}$;
    An image $M$;
    The original signature $\sigma$;
Output:
    A proof $\pi$;
1: if Verify$(pk_{\mathsf{sig}}, pk_{\mathsf{ed}}, M, \sigma) = 0$ then
2:     $\pi \leftarrow \perp$;
3: else
4:     provide the original message-signature pair $(M', \sigma')$ with $sk_{\mathsf{sig}}$;
5: end if
6: if Verify$(pk_{\mathsf{sig}}, pk_{\mathsf{ed}}, M', \sigma') = 1 \wedge M' \neq M \wedge \sigma' = \sigma$ then
7:     $\pi \leftarrow (M', \sigma')$;
8: else
9:     $\pi \leftarrow \perp$;
10: end if
11: return $\pi$;
```

Algorithm 5 takes the signer's private key sk_{sig}, the signer's public key pk_{sig}, the editor's public key pk_{ed}, and an image/signature pair (M, σ) as input. It outputs a proof π. That is $\pi \leftarrow \mathsf{Proof}(sk_{\mathsf{sig}}, pk_{\mathsf{sig}}, pk_{\mathsf{ed}}, M, \sigma)$. This algorithm is run by the signer to provide evidence that the editor has modified the image.

Security analysis

In our paper, the original image is blocked and divided into several non-overlapped parts. The image owner decides which parts can be modified but which cannot, which we called editing rule denoted by ER. To handle the rule, the non-editable image blocks are hashed by

Algorithm 6 Judge: The judging algorithm

```
Input:
    A signer's public key $pk_{\mathsf{sig}}$;
    A editor's public key $pk_{\mathsf{ed}}$;
    An image $M$;
    The original signature $\sigma$;
    A proof $\pi$;
Output:
    A result $Editor/Signer/\perp$;
1: if Verify$(pk_{\mathsf{sig}}, pk_{\mathsf{ed}}, M, \sigma) \neq 1$ then
2:     return $\perp$;
3: else
4:     if $\pi = \perp$ then
5:         return $Signer$;
6:     else
7:         Take out $(M', \sigma')$ from $\pi$;
8:     end if
9: end if
10: if Verify$(pk_{\mathsf{sig}}, pk_{\mathsf{ed}}, M', \sigma') = 1 \wedge M' \neq M \wedge \sigma' = \sigma$ then
11:     return $Editor$;
12: else
13:     return $Signer$;
14: end if
```

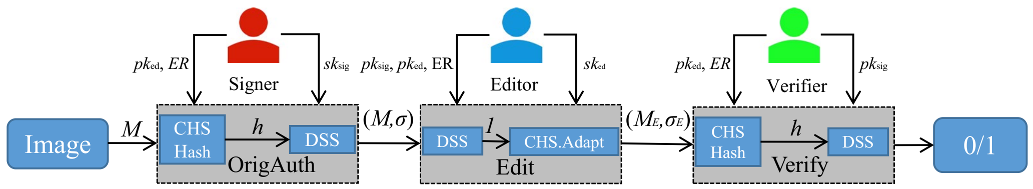

Fig. 4 Design overview

collision-resistance hash functions, while the editable image blocks are hashed by chameleon hash functions. These hash values are used to compute a signature. If the editor modifies the non-editable areas, the hash values computed by the collision-resistance hash functions will be changed, which will invalidate the signature. Therefore, the editor can only modify the content of editable image blocks hashed by the chameleon hash functions.

Second, to allow an image editor to modify the editable areas in the image, the image owner generates randomness r and computes a hash value $h = \mathsf{CHS.Hash}(b, r)$ for each block b. With security of chameleon hash functions, only an authorized editor (the trapdoor holder) can generate another randomness r' for a modified block b' which stratifies $\mathsf{CHS.Hash}(b, r) = CHS.Hash(b', r')$, which will not invalidate the signature. Since the aforementioned editing behavior is public verifiable (with a unique pubic key of the trapdoor holder), editing operation is undeniable and accountable. Furthermore, the ER is also input of the signature, so the ER is also immutable. The security of our scheme is guaranteed by the unforgeability of the digital signature and both security of collision-resistant hash functions and chameleon hash functions.

In this section, we will analyze the security of our scheme according to the security definition.

Theorem 6.1 *Our scheme is unforgeable if an ordinary signature scheme is EUF-CMA secure and a chameleon hash scheme is collision-resistance.*

Proof
Let $\mathcal{A}$ be a PPT adversary against our scheme. Suppose that $\mathcal{A}$ has forged a signature σ^* with a value t^* for an image M^*. Let M_i be $\mathcal{A}$'s ith query to the authenticating oracle $\mathcal{O}_{OA}$ and its answer is σ_i which contains t_i. We first define two events.

`Event 1`:$\forall i, t^* \neq t_i$.

`Event 2`:$\exists i, t^* = t_i$.

Let $P[Ev]$ represent the probability that the adversary $\mathcal{A}$ outputs a successful forgery. Let $P[Ev1]$ denote the probability that `Event 1` occurs and $P[Ev2]$ denote the probability of `Event 2` occurs.

$$P[Ev] = P[Ev|Ev1]P[Ev1] + P[Ev|Ev2]P[Ev2] \\ \leq P[Ev|Ev1] + P[Ev|Ev2] \tag{4}$$

In the following, we show that successful completion of `Event 1` breaks the unforgeability of the signature scheme and successful completion of `Event 2` breaks the collision resistance of the chameleon hash. If the digital signature is unforgeable and the chameleon hash function is collision-resistant, the probability of the adversary successfully completing `Event 1` and `Event 2` is negligible, so our scheme is unforgeable.

In `Event 1`, let $\mathcal{D}$ be a PPT adversary of DSS. $\mathcal{D}$ just needs to simulate the authentication oracle $\mathcal{O}_{OA}$. First, $\mathcal{D}$ generates `CHS`'s public and private key pair and sends the `CHS` public key and `DSS`'s challenge public key to adversary $\mathcal{A}$. Then, to answer $\mathcal{A}$'s i^{th} query, $\mathcal{D}$ runs the OrigAuth algorithm to compute the signature σ for $\mathcal{A}$ with the help of the signing oracle in `DSS`.

After $\mathcal{A}$ outputs its forgery (M^*, σ^*), $\mathcal{D}$ can extract message signature pairs $(h_1^*||h_2^*||\ldots||h_n^*||ER^*, t^*)$ from M^* and σ^*. Since in `Event 1`, $h_1^*||h_2^*||\ldots||h_n^*||ER^*$ is different from each M_i's $(h_1||h_2||\ldots||h_n||ER)$ which $\mathcal{D}$ has queried to its own underlying signing oracle. That is, $(h_1^*||h_2^*||\ldots||h_n^*||ER^*, t^*)$ is never queried to $\mathcal{D}$'s signing oracle, and it is a valid forgery of $\mathcal{D}$. As a result, $\mathcal{D}$ breaks the unforgeability of DSS. Assuming the underlying DSS satisfies EUF-CMA, $P[Ev|Ev1]$ is negligible.

In `Event 2`, $\mathcal{A}$ can be used to build an adversary $\mathcal{C}$ of a collision-resistant chameleon hash. First, $\mathcal{C}$ generates `DSS`'s public and private key pair and sends the `DSS` public key and `CHS`'s challenge public key to adversary $\mathcal{A}$. Then, $\mathcal{C}$ uses the underlying signing algorithm to simulate the Editing oracle $\mathcal{O}_{ED}$.

Then, to answer $\mathcal{A}$'s i^{th} query, $\mathcal{C}$ runs the E*dit* algorithm to compute the signature (M_E^i, σ_E^i) for (M_i, ES_i) to $\mathcal{A}$ with the help of the chameleon hash oracle in `CHS`.

Table 1 General analysis of our scheme

Algorithm	Computation cost	Communication cost
OrigAuth	$m \cdot P + (2m+1) \cdot E$	$l + n \cdot Q + le$
Edit	$2m \cdot P + (3m+1) \cdot E$	$l + n \cdot Q + le$
Verify	$m \cdot P + (2m+1) \cdot E$	–

Let (M^*, σ^*) be the output of $\mathcal{A}$. Let's assume that $t^* = t_k$, here $M^* \neq M_k \wedge M^* \neq M_E^k$. $\mathcal{C}$ can be calculated from the effective chameleon hash $h_i^* = h_i$ without using trapdoor with the help of (M^*, σ^*). Observe that in `Event 2`, r_i^* is different from each σ_E^k's r_i which $\mathcal{C}$ has queried to its own underlying chameleon hash oracle. This breaks the collision-resistance of the underlying chameleon hash. Assuming the underlying `CHS` satisfies collision-resistance, $P[Ev|Ev2]$ is negligible.

Therefore, if an ordinary signature scheme (`DSS`) is EUF-CMA secure and an `CHS` scheme is collision-resistance, the probability for an adversary $\mathcal{A}$ to forge a valid signature for a new image in our scheme is negligible. We complete the proof of Theorem 1.1.

□

Experiments and evaluation

In this section, we provide the simulation of our scheme.

General analysis

For a concrete instantiation, we choose the RSA signature scheme and the chameleon hash in Ateniese and de Medeiros (2004). Since the computation of the hash function: $\mathcal{H} : \{0,1\}^* \rightarrow \{0,1\}^q$ are efficient, we only consider the following two more time-consuming operations in our scheme. (1) Multiplication operation denoted by P; (2) Exponentiation operation denoted by E. The communication cost of a digital signature denoted by l, size of randomness r_i denoted by Q, size of the editing rule ER denoted by le, and the edited image M_E denoted by ME.

Table 2 Details of the test images

No.	Image name	Resolution	Size (byte)
1	Airfield2	1024 × 1024	1049654
2	Baboon	500 × 480	720054
3	Barbara	720 × 576	1244214
4	Fingerprint	256 × 256	66614
5	Pepper	512 × 512	786486

Assuming that the image is divided into n sub-images and the number of editable areas is m, we summarize the computation and communication costs in Table 1.

From the perspective of efficiency, the computational cost of our scheme increases with the number of editable areas, and the communication cost increases with grain of the image segmentation.

Instantiation and performances

We carry out our experiments in a PC (CPU: IntelCore I5 7500; Memory: 8 GB (3400 MHz)). We use C++ language coding cryptographic algorithms combined with OpenCV and the Miracl libraries, and the code is compiled by Visual Studio 2017.

Firstly, we demonstrate the efficiency of our scheme through simulation experiments. We use 512 × 512 images to simulate the application of the scheme. In this image, the total number of original pixels is 262144, which is divided into 16 blocks with 16384 pixels in each block. The rule ER is defined to determine which areas cannot be changed. We concatenate the immutable areas and takes them as the input of SHA-256 together with ER. The size of the original message is 6291888 bits. We use SHA-256 to compress the original image to a 256-bit message. The key size of an RSA signature is 1024 bits.

We select more standard images and adopt the control variable method in Fig. 5 to evaluate the efficiency of the scheme and the main factors affecting the efficiency. We summarize the information about these pictures in Table 2. The results are given in Tables 3, 4 and 5. Table 3

Fig. 5 Some of the test images used in our experiment

Table 3 The time cost of test images with different number of areas

No.	*n*	*m*	*s*	OrigAuth	Edit	Verify	Proof	Judge
5	2	1	512 × 256	4.8 ms	4.1 ms	2.1 ms	4.3 ms	2.2 ms
5	4	1	256 × 256	4.9 ms	3.9 ms	2.2 ms	4.6 ms	2.5 ms
5	8	1	256 × 128	5.2 ms	4.3 ms	2.8 ms	5.4 ms	2.9 ms
5	16	1	128 × 128	6.1 ms	4.8 ms	3.2 ms	6.6 ms	3.4 ms
5	32	1	128 × 64	8.1 ms	5.9 ms	3.9 ms	8.2 ms	4.2 ms

n: the number of areas, *m*: the number of edited areas, *s*: the size of the areas

Table 4 The time cost of test images with different number of immutable areas

No.	*n*	*m*	*s*	OrigAuth	Edit	Verify	Proof	Judge
5	16	1	128 × 128	6.1 ms	4.9 ms	2.9 ms	6.5 ms	3.1 ms
5	16	2	128 × 128	7.9 ms	8.8 ms	4.1 ms	9.7 ms	5.1 ms
5	16	4	128 × 128	9.2 ms	14.3 ms	6.8 ms	13.8 ms	6.8 ms
5	16	8	128 × 128	13.5 ms	26.4 ms	11.5 ms	23.3 ms	12.2 ms

n: the number of areas, *m*: the number of edited areas, *s*: the size of the areas

Table 5 The time cost of test images

No.	*n*	*m*	*s*	OrigAuth	Edit	Verify	Proof	Judge
1	16	8	256 × 256	13.4 ms	25.6 ms	10.7 ms	22.8 ms	11.8 ms
2	16	8	125 × 120	13.4 ms	25.9 ms	10.6 ms	22.9 ms	11.6 ms
3	16	8	180 × 144	13.3 ms	25.8 ms	10.6 ms	22.3 ms	11.6 ms
4	16	8	64 × 64	13.0 ms	25.3 ms	11.2 ms	23.6 ms	11.3 ms
5	16	8	128 × 128	13.5 ms	26.4 ms	11.5 ms	23.3 ms	12.2 ms

n: the number of areas, *m*: the number of edited areas, *s*: the size of the areas

Table 6 The efficiency of our scheme compared with (Chen et al. 2020)

Scheme	Image size	*n*	*s*	Authentication	Edit	Verify
Our scheme	512 × 512	16	128 × 128	13.5 ms	26.4 ms	11.5 ms
(Chen et al. 2020)	512 × 512	16	128 × 128	31 ms	6 ms	15 ms

n: the number of areas, *s*: the size of the areas

shows the results of the same image with a different number of areas. For example, when picture NO. 5 is divided into 16 blocks and the number of editing blocks is 1, the size of the areas is 128 × 128, the average time of the `Sign` algorithm is 6.1 ms, the execution time of `Edit` is 4.8 ms, and the execution time of `Verify` is 3.2 ms. The results in Table 3 also show that the more segmented areas there are, the longer the computation time will be. In other words, the smaller the size of the image block, the greater the calculation cost.

Table 4 shows the results of different numbers of editing areas in the same image, and the computation time increases linearly with the number of editing areas. When image NO. 5 is divided into 16 blocks and the number of editing blocks is 8, the average time of `Sign` algorithm is 13.5 ms, the execution time of `Edit` is 26.4 ms, and the execution time of `Verify` is 11.5 ms.

Table 5 shows the results of different images with the same editable areas. Figure 6 provides the results of different images with different editable areas. From Table 5

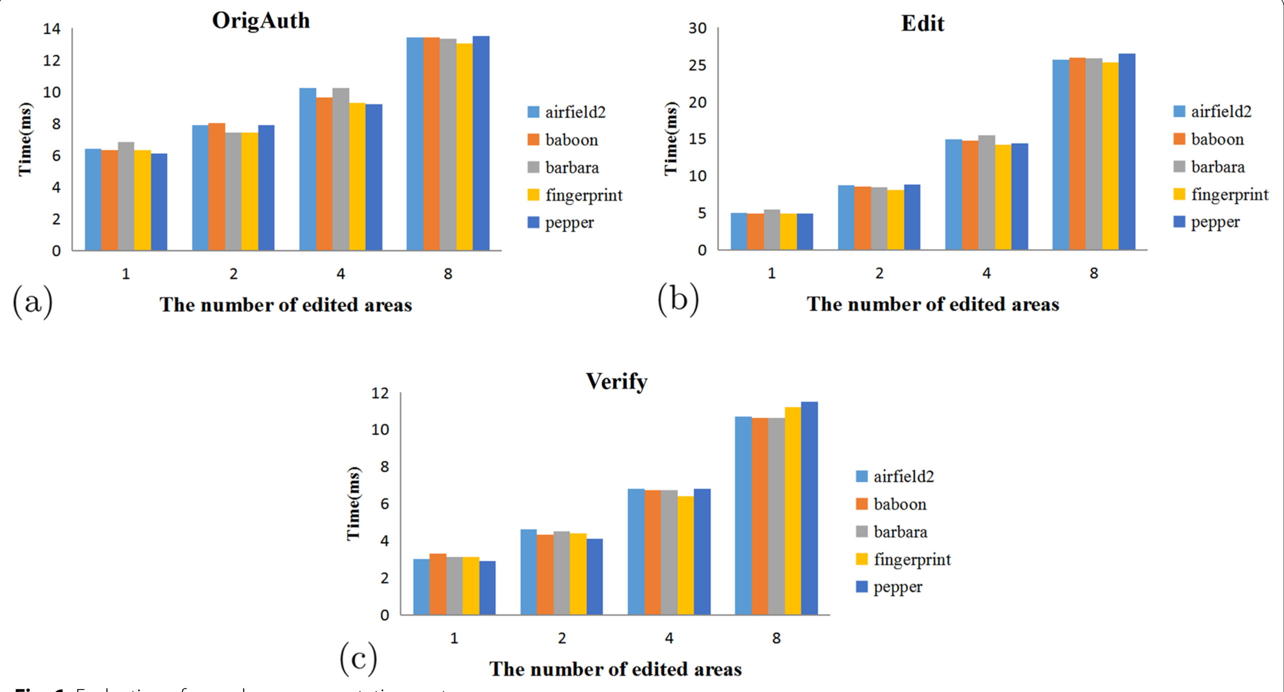

Fig. 6 Evaluation of our scheme computation cost

and Fig. 6, we can conclude that the time costs are related to the number of editable areas.

We also compared our scheme with Chen et al. (2020) in Table 6. Both of these two schemes support local image editing, but the difference is that our scheme is local coverage, while Chen et al. (2020) is local image extraction. Chen et al. (2020) is designed with an aggregator and signature. From the results, our scheme is a better choice when local information of the image needs to be deleted.

In summary, the computation and communication costs of the scheme are mainly from image hashing and signature. The number of sub-images and the number of editable regions are the main factors that affect the computation and communication cost of image hashing.

Conclusions

In this paper, we propose a practical image authentication scheme for permissible content revision, which is constructed from chameleon hashes and ordinary digital signatures. The security of our scheme is guaranteed by relevant cryptographic primitives. Our scheme allows legitimate users to revise the image contents and proves the authenticity of the revised image without interaction with the original authenticator. The experiment results show that our scheme is practical and can be used in image applications where both privacy protection and security are required. The disadvantage of our scheme is that the modification rule is block-based and users cannot change the block size, which limits usage of the authenticated image. Our future work is to design more practical authentication schemes to support more kinds of flexible image processing operations.

Abbreviations

DSS: Digital signature scheme; EUF-CMA: Existential unforgeability under adaptive chosen-messages attacks; CHS: Chameleon hash scheme; PPT: Probabilistic polynomial time.

Acknowledgements

We would like to thank the anonymous reviewers and editors for detailed comments and useful feedback.

Authors' contributions

The design of the scheme and the writing of the paper were completed by Xu and Chen. All author(s) read and approved the final manuscript.

Funding

This work was supported by National Natural Science Foundation of China (Grant Nos. 61902070, 61902289).

Availability of data and materials

All data generated or analysed during this study are included in this published article.

Declarations

Competing interests

The authors declare that they have no competing interests.

Received: 2 March 2021 Accepted: 3 August 2021
Published online: 02 October 2021

References

Agarwal N, Singh AK, Singh PK (2019) Survey of robust and imperceptible watermarking. Multim Tools Appl 78:8603–8633

Al-Otum HM (2014) Semi-fragile watermarking for grayscale image authentication and tamper detection based on an adjusted expanded-bit multiscale quantization-based technique. J Vis Commun Image Represent 25:1064–1081

Ateniese G, Chou DH, de Medeiros B, Tsudik G (2005) Sanitizable signatures. In: di Vimercati SDC, Syverson PF, Gollmann D (eds) Computer security-ESORICS 2005, 10th European symposium on research in computer security, pp 159–177

Ateniese G, de Medeiros B (2004) Identity-based chameleon hash and applications. In: Juels A (ed) 8th international conference on financial cryptography, FC 2004, pp 164–180

Ateniese G, de Medeiros B (2004) On the key exposure problem in chameleon hashes. In: Blundo C, Cimato S (eds) 4th international conference security in communication networks, SCN 2004, pp 165–179

Chabanne H, Hugel R, Keuffer J (2017) Verifiable document redacting. In: Foley SN, Gollmann D, Snekkenes E (eds) Computer security-ESORICS 2017-22nd European symposium on research in computer security, pp 334–351

Chen F, He H, Huo Y (2017) Self-embedding watermarking scheme against JPEG compression with superior imperceptibility. Multim Tools Appl 76:9681–9712

Chen H, Huang X, Wu W, Mu Y (2020) Efficient and secure image authentication with robustness and versatility. Sci China Inf Sci 63:1–18

Chen H, Huang X, Wu W, Mu Y (2020) Privacy-aware image authentication from cryptographic primitives. Comput J

Chen H, Wang S, Zhang H, Wu W (2018) Image authentication for permissible cropping. In: Guo F, Huang X, Yung M (eds) Information security and cryptology—14th international conference, Inscrypt 2018, pp 308–325

Chen X, Zhang F, Kim K (2004) Chameleon hashing without key exposure. In: Zhang K, Zheng Y (eds) Information security, 7th international conference, ISC 2004, pp 87–98

Du L, Ho ATS, Cong R (2020) Perceptual hashing for image authentication: a survey. Signal Process Image Commun 81:115713

Fridrich JJ, Goljan M, Memon ND (2002) Cryptanalysis of the Yeung—mintzer fragile watermarking technique. J Electronic Imaging 11:262–274

Guo Q, Zhang C, Zhang Y, Liu H (2016) An efficient SVD-based method for image denoising. IEEE Trans Circuits Syst Video Technol 26:868–880

Jiang C, Pang Y (2018) Perceptual image hashing based on a deep convolution neural network for content authentication. J Electron Imaging 27:043055

Kim J, Lee S, Yoon J, Ko H, Kim S, Oh H (2017) PASS: privacy aware secure signature scheme for surveillance systems. In: 14th IEEE international conference on advanced video and signal based surveillance, AVSS 2017, pp 1–6

Lin C, Chang S (2000) Semifragile watermarking for authenticating JPEG visual content. In: Wong PW, Edward JD III (eds) Security and watermarking of multimedia contents II, pp 140–151

Lu C, Liao HM (2001) Multipurpose watermarking for image authentication and protection. IEEE Trans Image Process 10:1579–1592

Peng F, Guo R, Li C, Long M (2010) A semi-fragile watermarking algorithm for authenticating 2d CAD engineering graphics based on log-polar transformation. Comput Aided Des 42:1207–1216

Pun C, Yan C, Yuan X (2018) Robust image hashing using progressive feature selection for tampering detection. Multim Tools Appl 77:11609–11633

Sreenivas K, Prasad VK (2018) Fragile watermarking schemes for image authentication: a survey. Int J Mach Learn Cybern 9:1193–1218

Ur-Rehman O, Zivic N (2017) A robust watermarking technique for image content authentication. In: Ganzha M, Maciaszek LA, Paprzycki M (eds) Communication papers of the 2017 federated conference on computer science and information systems, FedCSIS 2017, pp 223–226

Venkatesan R, Koon S, Jakubowski MH, Moulin P (2000) Robust image hashing. In: Proceedings of the 2000 international conference on image processing, ICIP 2000, pp 664–666

Wang C, Zhang H, Zhou X (2018) Review on self-embedding fragile watermarking for image authentication and self-recovery. J Inf Process Syst 14:510–522

Yu X, Wang C, Zhou X (2017) Review on semi-fragile watermarking algorithms for content authentication of digital images. Future Internet 9:56

Zhu C, Hu Y (2008) A multipurpose watermarking scheme for image authentication and copyright protection. In: Yu F, Luo Q, Chen Y, Chen Z (eds) Proceedings of the international symposium on electronic commerce and security, ISECS 2008, pp 930–933